UNIVERSIDADE FEDERAL DE SÃO CARLOS

CENTRO DE CIÊNCIAS EXATAS E DE TECNOLOGIA

PROGRAMA DE PÓS-GRADUAÇÃO EM ESTRUTURAS E CONSTRUÇÃO CIVIL

PROGRAMA LIVRE PARA ANÁLISE DA ARMADURA LONGITUDINAL E DA TRANSVERSAL DE VIGAS PRÉ-TRACIONADAS PARA DIFERENTES SEÇÕES

Gabriel da Motta Trevizoli

São Carlos
2015

UNIVERSIDADE FEDERAL DE SÃO CARLOS

CENTRO DE CIÊNCIAS EXATAS E DE TECNOLOGIA

PROGRAMA DE PÓS-GRADUAÇÃO EM ESTRUTURAS E CONSTRUÇÃO CIVIL

PROGRAMA LIVRE PARA ANÁLISE DA ARMADURA LONGITUDINAL E DA TRANSVERSAL DE VIGAS PRÉ-TRACIONADAS PARA DIFERENTES SEÇÕES

Gabriel da Motta Trevizoli

Dissertação apresentada ao Programa de Pós-Graduação em Estruturas e Construção Civil da Universidade Federal de São Carlos como parte dos requisitos para a obtenção do Título de Mestre em Estruturas e Construção Civil.

Área de Concentração: Sistemas Construtivos

Orientador: Prof. Dr. Roberto Chust Carvalho

São Carlos
2015

Ficha catalográfica elaborada pelo DePT da Biblioteca Comunitária da UFSCar

T814pL

Trevizoli, Gabriel da Motta.
Programa livre para análise da armadura longitudinal e da transversal de vigas pré-tracionadas para diferentes seções / Gabriel da Motta Trevizoli. -- São Carlos : UFSCar, 2015.
270 f.

Dissertação (Mestrado) -- Universidade Federal de São Carlos, 2015.

1. Software livre. 2. Vigas. 3. Análise de estruturas. I. Título.

CDD: 005.1 (20ª)

UNIVERSIDADE FEDERAL DE SÃO CARLOS

Centro de Ciências Exatas e de Tecnologia
Programa de Pós-Graduação em Estruturas e Construção Civil

Folha de Aprovação

Assinaturas dos membros da comissão examinadora que avaliou e aprovou a Defesa de Dissertação de Mestrado do candidato Gabriel da Motta Trevizoli, realizada em 30/03/2015:

Prof. Dr. Roberto Chust Carvalho
UFSCar

Prof. Dr. Fernando Menezes de Almeida Filho
UFSCar

Prof. Dr. Paulo Sergio dos Santos Bastos
UNESP

AGRADECIMENTOS

Primeiramente, agradeço a Deus por me proporcionar todas as condições para realização desse trabalho.

Em seguida, a todos de minha família que sempre me apoiaram em minhas decisões e sempre estiveram presentes nos momentos difíceis ao longo dessa jornada.

Ao Prof. Dr. Roberto Chust Carvalho pela sua presença ao longo desse trabalho, atuando com muita paciência e sabedoria no desenvolvimento desse programa.

Aos amigos e companheiros de pós-graduação, com quem caminhei junto ao longo desses dois anos e com quem compartilhei muitas das dificuldades e alegrias dessa jornada. Uma menção especial a Andrew John Richter Cass, brilhante no conhecimento sobre programação, sem o qual esse trabalho nunca teria acontecido, e Leonardo Martins e Silva, pelas produtivas discussões técnicas que tanto me acrescentaram.

Aos amigos de colégio, graduação e em geral que estiveram presentes nos momentos bons e difíceis do mestrado e da vida, aconselhando e compartilhando momentos juntos.

A todos do Programa de Pós Graduação em Estruturas e Construção Civil (PPGECiv) por propiciarem um ambiente estruturado e de qualidade com o qual tanto me foi acrescentado em meus conhecimentos de engenharia.

À Coordenação de Aperfeiçoamento Pessoal de Nível Superior (CAPES) pela bolsa, que me permitiu dedicação exclusiva ao projeto.

RESUMO

TREVIZOLI, G. M. **Programa livre para análise da armadura longitudinal e transversal de vigas pré-tracionadas para diferentes seções.** 2015. 270 f. Dissertação (Mestrado em Estruturas e Construção Civil) - Universidade Federal de São Carlos, São Carlos, 2015.

Frente às necessidades em se aprimorar ferramentas que aumentem a produtividade na elaboração de projetos, aliada à confiabilidade técnica, optou-se por desenvolver um aplicativo computacional livre que realiza o cálculo e o detalhamento da armadura longitudinal, ativa e passiva, de vigas com pré-tração e biapoiadas, de forma a auxiliar profissionais da área da engenharia civil no cálculo estrutural de vigas protendidas. O programa está desenvolvido no ambiente gráfico do Lazarus, que utiliza a linguagem Pascal, de forma a utilizar programação orientada ao objeto (POO), facilitando ao usuário que deseje ter acesso aos objetos constituintes. A princípio, o usuário entra com os dados geométricos da seção transversal, assim como os carregamentos solicitantes da viga. Desta forma, o programa esta apto a pré-dimensionar sua armadura longitudinal. São realizadas verificações de tensões, em ambas as bordas, para as combinações no estado limite de serviço (ELS) e no estado limite último (ELU) em vazio, a análise de abertura de fissuras e as verificações de flechas no elemento. A verificação no estado limite último no tempo infinito é feita verificando a ruptura da seção. As perdas de protensão são devidamente calculadas segundo critérios estabelecidos na NBR6118:2014, de forma a obter valores próximos com a realidade. Por fim, verifica-se a armadura transversal para resistir à força cortante. Neste trabalho são, ainda, apresentadas as hipóteses e preceitos normativos usados no programa. São feitos diversos exemplos de validação e mostra-se como o mesmo pode se transformar em ferramenta de análise. São ainda apresentadas as listagens e indicadas, também, as diversas possibilidades de expansão do mesmo.

Palavras-chave: programa livre; pré-tração; análise estrutural

ABSTRACT

TREVIZOLI, G. M. **Free program for analysis of longitudinal and transverse reinforcement of pre-tensioned beams for different sections.** 2015. 270 f. Dissertação (Mestrado em Estruturas e Construção Civil) - Universidade Federal de São Carlos, São Carlos, 2015.

In face of the needs to improve tools that increase productivity in project design, coupled with technical reliability, it was developed a free computer application that performs the equations and details the longitudinal reinforcement of pre-tensioned simply supported prestreessed beams, so it can assist professionals of the área. The program was developed in the graphical environment of Lazarus, which uses Pascal's programming language, in order to use object-oriented programming (OOP), facilitating the user who wishes to have access to the constituent objects. Initially, the user enters with the geometric cross-section's data, as well as the loading characteristics over the beam. In this way, the software will be able to pre-size its longitudinal reinforcement. According to NBR 6118:2014 is necesary to check tensions in both edges of the beam for the serviceability state combination (SLS) and ultimate limit state (ULS) in an empty-load stage, crack control requierments, as well as checks of deflections. The checking for ULS in full-load stage is made by checking the collapse section. Prestressing losses will be properly calculated according to criteria established in NBR 6118:2014, in order to obtain values closer to reality. Finally, the software checks the transverse reinforcement to see if it can resist the efforts of the shear force. This paper also presents the assumptions and normative precepts used in the program. Several examples of validation were made and it is shown how it can turn into an analysis tool. Also presents the listings and it is also given the different possibilities of expansion.

Key words: free software, pre-tensioned, structural analysis.

SUMÁRIO

1 INTRODUÇÃO

Aliar as necessidades arquitetônicas com métodos construtivos eficazes sempre foi um dos grandes desafios encontrados na construção civil. O concreto protendido se apresenta como uma solução que vem sendo cada vez mais utilizadas tanto em edifícios de múltiplos pavimentos como em obras de infraestrutura. Esta tecnologia permite a utilização de elementos de concreto com relativa redução da seção para um mesmo vão e carregamentos solicitante quando comparados com uma viga de concreto armado convencional.

Seu conceito se baseia na ideia de armadura ativa. Isso significa que a armadura do elemento é mecanicamente tracionada e, por meio de algum mecanismo de transferência de esforços (por atrito ou pela ancoragem) direciona a tensão para o concreto. A utilização de cabos excêntricos ao centro de gravidade da seção faz com que seja induzido um momento de protensão que tende a tracionar a borda superior e comprimir a inferior, fato este que implica numa série de considerações e situações críticas a serem consideradas. Há de se ressaltar que o aço utilizado nas cordoalhas de protensão é de resistência maior que o aço de armadura passiva, isso porque ocorrem perdas de protensão ao longo da vida útil do elemento que poderiam simplesmente levar o efeito da protensão à zero caso fossem aplicadas em aço comum de armadura passiva.

Os mecanismos de transferência citados anteriormente podem ser classificados em três tipos diferentes:

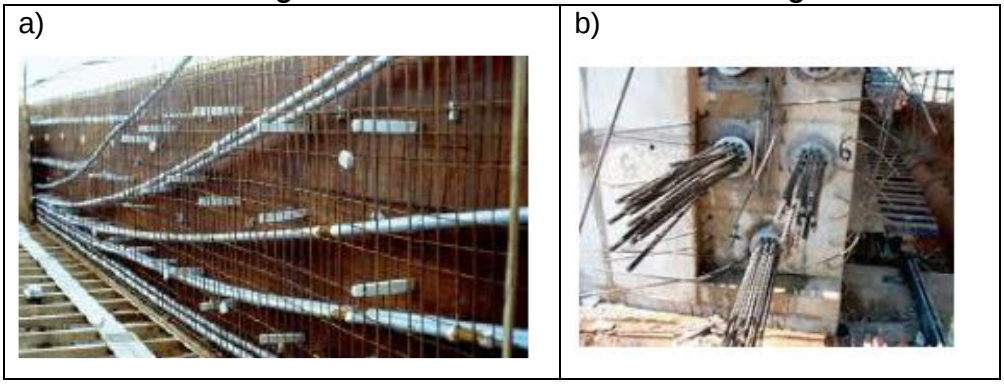
- Pré-tração com aderência (ou aderência inicial): Nesta situação a armadura de protensão é distendida antes do lançamento do concreto nas fôrmas. Quando o concreto atinge resistência adequada a armadura é cortada nas suas extremidades passando a comprimir o elemento, garantindo que a ancoragem das cordoalhas ocorra somente por aderência.

Figura 1: Sistema de pré-tração. a) Viga pré-tracionada; b) Ancoragem ativa da pista de protensão; c) Cone fêmea da ancoragem; d) Cone macho da ancoragem.

Fonte: Faleiros Junior, 2014. Arquivo pessoal.

- Pós-tração com aderência (ou aderência posterior): As bainhas (tubos plásticos corrugados) são posicionadas dentro da fôrma da viga para, em seguida, serem concretadas. Quando o concreto adquire determinada resistência, as cordoalhas são tracionadas e ancoradas nas extremidades do elemento. Finalmente, injeta-se uma nata de cimento sob pressão no interior da bainha, proporcionando a aderência do cabo com a viga. Esse tipo de sistema permite a utilização de cabos que podem acomodar uma série de cordoalhas, facilitando o posicionamento da ancoragem na extremidade da viga.

Figura 2: Sistema de pós-tração. a) Traçado curvo dos cabos no interior da viga; b) Ancoragem dos cabos na extremidade da viga.



Fonte: Faleiros Junior, 2014. Arquivo pessoal.

- Protensão sem aderência: Cita-se como exemplo a cordoalha engraxada. O processo executivo é semelhante à pós-tração com aderência posterior, no entanto a etapa de injeção da nata de cimento não ocorre. A transferência de esforços é garantida apenas pela ancoragem nas extremidades da viga.

Está claro que as diferenças entre o concreto armado convencional e protendido são inúmeras. Além das diferenças construtivas e estruturais, o modelo de cálculo e as verificações de ruptura e em serviços também possuem algumas distinções. No concreto armado convencional a situação crítica é, normalmente, a de ruptura, fato que leva os engenheiros a dimensionar nesta situação e verificá-la (quando esta é realizada) em serviço. No protendido, ambas as situações podem levar a uma situação crítica, cabendo ao responsável decidir para qual estado limite a armadura será dimensionada. Há de se ressaltar que essas verificações devem ser realizadas em dois momentos distintos da vida útil da viga: em vazio, caracterizado pela tensão máxima de protensão e carregamento mínimo (apenas peso próprio); e no tempo infinito, caracterizado pela tensão mínima nas cordoalhas (após as perdas de protensão) e carregamento máximo quando se encontra em utilização.

Outra peculiaridade do protendido está nas seções de momento máximo. Considerando o caso de uma viga biapoiada com pré-tração, as seções de extremidade estão sujeitas a momentos negativos de protensão de alta intensidade (os cabos são retos), sendo consideradas regiões com grande potencial de apresentar problemas.

Analisando todas essas questões apresentadas, fica clara a importância e a dificuldade em dimensionar uma viga protendida analisando todos os detalhes. Pode-se dizer, inclusive, que a falta de ferramentas computacionais que auxiliem no processo de cálculo e no detalhamento são fatores determinantes para o sucesso de uma empresa. Afinal, os programas tendem a agilizar, em muito, a tomada de decisões técnicas além de garantir um projeto de melhor qualidade. Também fica claro que a solução de estruturas em concreto protendido não é única, pois, só para exemplificar, é possível alterar a intensidade de protensão e combinar armadura ativa com armadura passiva. Assim, os programas para estruturas em concreto protendido exigem a atuação do profissional na sua solução final, diferente das soluções em concreto armado convencional em que os programas conseguem calcular e detalhar a armadura sem a interferência do operador do mesmo.

De forma a aumentar a competitividade dos escritórios de projeto e fábricas de pré-moldado, estas tendem a investir em ferramentas técnicas que visem não somente a fabricação de peças, mas também a produção de projetos de qualidade que sejam capazes de transmitir com clareza suas informações e propiciar economia para a empresa. Logo, os programas de cálculo utilizados por empresas e escritórios de projeto são, em sua maioria,

de natureza comercial, desenvolvidos exclusivamente para serem comercializados entre o público alvo e que permitem calcular estruturas de todos os portes. No entanto, estes não são, muitas vezes, acessíveis financeiramente o que leva o engenheiro a utilizar outras ferramentas para otimizar o processo de cálculo.

É nesse momento que se mostra a importância dos programas livres. Este se define por ser não apenas gratuito, mas também por tornar público todas as linhas de programação de forma a permitir que o usuário possa alterá-las as suas necessidades e preferências, desde que entenda a linguagem de programação do programa.

O desenvolvimento deste tipo de programa por parte das universidades permite o avanço contínuo destes ao longo do tempo. Fazendo uso de programação orientada ao objeto (POO) pode-se, por exemplo, criar novos componentes que venham a complementar determinado programa ou alterar algum componente já existente decorrente de alterações normativas. Por exemplo, a norma brasileira de concreto (NBR 6118) será atualizada a cada cinco anos. No caso de programas livres, estas podem ser corrigidas diretamente pelo programador diferentemente dos programas comerciais, onde seria necessário adquirir uma nova versão mediante pagamento. Por meio de programação orientada ao objeto um programa que faça, no presente momento, apenas o cálculo e verificação da armadura de protensão de uma viga pode, futuramente, possuir módulos que façam o cálculo da armadura transversal, verificações na situação de incêndio, detalhamento gráfico da peça, dentre uma série de outras análises. Um exemplo de programa muito utilizado para definição de esforços de uma estrutura e que vem apresentando um processo de desenvolvimento contínuo ao longo do tempo é o Ftool (gratuito, mas não é livre), que a cada versão apresenta novos módulos de análise (como consideração da temperatura, do trem-tipo longitudinal, dentre outras).

1.1 OBJETIVOS

Desenvolver um programa livre (gratuito e com os algoritmos disponibilizados) que calcula e verifica a armadura longitudinal de uma viga protendida pré-tracionada e biapoiada para os seguintes tipos de seção: seção retangular, seção “I”, seção “T” e seção “T” invertida. Em todos os casos poderá ser considerada a seção simples e a seção solidarizada (composta). Também será efetuado o cálculo das perdas de protensão, da flecha imediata, devido à fluência e total. O programa irá permitir a alteração de coeficientes normativos como os limites de tensão, simplificadores do diagrama parábola-retângulo, coeficientes redutores de resistência e majoradores de carga e os coeficientes relativos às perdas de protensão. Este será desenvolvido utilizando-se o ambiente de programação do Lazarus, que faz uso da linguagem pascal.

1.1.1 Objetivos complementares

Como objetivos secundários, visa-se criar uma ferramenta computacional que sirva, também, para analisar e refinar as diversas soluções aplicáveis em projetos de vigas pré-tracionadas. Aplicar esses conceitos em programação orientada ao objeto permitindo que este seja personalizado e implementado de maneira a melhor satisfazer o usuário.

Discussão das inovações trazidas pela NBR6118:2014, das hipóteses e dos procedimentos de cálculo utilizados e uma formação acadêmica na área de pesquisa e tecnologia em concreto protendido ao aluno.

1.2 JUSTIFICATIVA

A falta de ferramentas computacionais de qualidade que auxiliem na produção e na tomada de decisões são fatores que interferem na competitividade das empresas. A iniciativa de se criar programas livres (ou seja, programas gratuitos e com a rotina de programação disponibilizada) se traduz na necessidade de disponibilizar ferramentas de confiança que sejam uma opção aos programas comerciais existentes, muitas vezes inacessíveis financeiramente. Além disso, essas ferramentas podem substituir a utilização de planilhas eletrônicas e o tempo útil de mão de obra na elaboração gráfica de projetos, práticas usuais nos escritórios do Brasil.

O que se vê hoje são programas proprietários (desenvolvidos por empresas) de valor de licença extremamente alto e que requerem constante atualização aumentando ainda mais os custos na elaboração de projetos. As normas técnicas (pelo menos a de concreto armado e protendido) serão atualizadas a cada 5 anos, como está acontecendo neste ano de 2014. Se tratando de um programa proprietário é preciso comprar sua atualização. No caso de programas livres, em que a listagem é disponibilizada, as próprias empresas ou profissionais podem fazer mudanças ou mesmo personalizá-las com pequenas implementações, desde que entendam a linguagem de programação do programa.

Toda vez que há uma mudança normativa torna-se necessária aplicá-la nos programas computacionais. Um exemplo deste é a formulação do modelo II para o cálculo de amadura transversal. Qual o ângulo da biela do concreto mais próximo da realidade? Há o aumento ou diminuição da armadura com este novo modelo? Como ficam os esforços na biela do concreto? Estes e outros temas não são tratados nos manuais dos programas.

O Manual Munte de Projetos em Pré-fabricados de Concreto (Melo, 2007) e o livro Concreto Pré-Moldado: Fundamentos e Aplicações (El Debs, 2000) são duas literaturas bastante completas no que diz respeito ao detalhamento de elementos pré-fabricados. As informações técnicas e os detalhes construtivos apresentados nestas obras (principalmente no Manual Munte), que foram baseadas na NBR 9062:2006, são referência nacional para os

escritórios de cálculo. Desta forma, houve uma preocupação em utilizá-las para o desenvolvimento do módulo de detalhamento gráfico das armaduras.

No que diz respeito às questões teóricas de cálculo, a literatura nacional é relativamente escassa quando comparada com a Europa e os EUA. Grande parte da literatura brasileira dedicada a esse assunto é oriunda de publicações acadêmicas e algumas poucas obras didáticas que apresentam modelos de cálculo para o dimensionamento e as verificações dos elementos. Muitos desses modelos foram embasados em literatura internacional como as encontradas em Collins, 1997; Lyn, 1981; Nawy, 1995 e Elliot, 2002, este último que apresenta um modelo mais convencional de cálculo utilizando como parâmetros valores apresentado na norma Britânica e no Eurocode.

Como se pode observar, muitas das obras-base dessa dissertação são referentes ao século passado, mostrando que ainda há muito a ser desenvolvido e estudado, no que diz respeito à literatura nacional, frente aos países desenvolvidos. O minucioso estudo dessas obras se faz necessário tanto para refinamento das considerações de cálculo adotadas para o programa, como também para enriquecer a literatura nacional quanto aos modelos de cálculo.

Por ser tratar de uma linguagem orientada ao objeto, isto significa dizer que o programa é composto por uma série de componentes (por exemplo: componente para o cálculo das características geométricas ou para o cálculo das perdas de protensão) que, quando interligados de uma maneira lógica, constituem o programa final. Da mesma forma, este programa final, em um tempo futuro, pode ser observado como um componente de outro programa mais amplo que envolva este primeiro. Entendido este conceito, torna-se clara sua versatilidade, possibilitando uma reutilização e/ou ampliação dos programas ou dos componentes desenvolvidos para projetos futuros.

Como uma justificativa final e igualmente relevante, frente ao aumento da demanda por engenheiros civis no mercado e a consequente alta no número de estudantes nas universidades, os programas livres se tornam uma poderosa ferramenta de ensino, permitindo que o aluno foque mais nos conceitos e teorias do que no trabalho manual. Este programa não tem como objetivo ser apenas uma ferramenta de cálculo utilizada para dimensionar e detalhar vigas. Esta talvez seja sua aplicação mais aparente. Mas a simples ideia de utilizá-lo apenas como programa de dimensionamento não demonstra a total potencialidade que este pode oferecer. Além dessa questão prática, é importante que um programa dê ao usuário a possibilidade de testar inúmeras soluções a fim de “encaixar” esta às limitações técnicas da empresa. Sabe-se que elementos protendidos podem apresentar uma gama enorme de soluções que vão desde armadura mista até utilização de cabos de protensão na borda superior, mas também se sabe que aquilo que parece ser a melhor solução para uma empresa, não o é para a outra. Criar um programa que não leve em conta

essa liberdade de criar e testar outras soluções em consideração significa criar uma ferramenta incompleta sob o ponto de vista prático. Portanto, é de extrema importância ressaltar que este não é apenas um programa de cálculo, mas sim, também, de análise e experimentação.

1.3 METODOLOGIA

As etapas de trabalho, de forma geral, podem ser resumidas da seguinte forma: pesquisa bibliográfica, deduções analíticas, simulação e validação e comparação dos resultados. A abordagem de pesquisa apresenta um fundo aplicado ao meio técnico.

Como breve descrição do funcionamento do programa, a viga é dividida em seções espaçadas em décimos de vão e para cada seção realiza-se a verificação para o ELU em vazio, ELU no tempo infinito e ELS, segundo nível de protensão estabelecido pela norma (parcial, limitada e completa). O programa calcula, então, a armadura transversal ao longo da viga analisando a região dos apoios onde o cisalhamento é crítico. Ao término, o programa salva os dados relevantes para o detalhamento da armadura na seção transversal e longitudinal da peça. Neste detalhamento representam-se: o traçado dos cabos (ativo e passivo para ambas as bordas); a armadura transversal (inclusive o estribo exposto para seção composta), isolamento dos cabos quando existir; representação da capa de concreto para a seção composta; posição/quantidade/diâmetro de todas as barras (ativa, passiva e transversal) e a tabela com os quantitativos de aço.

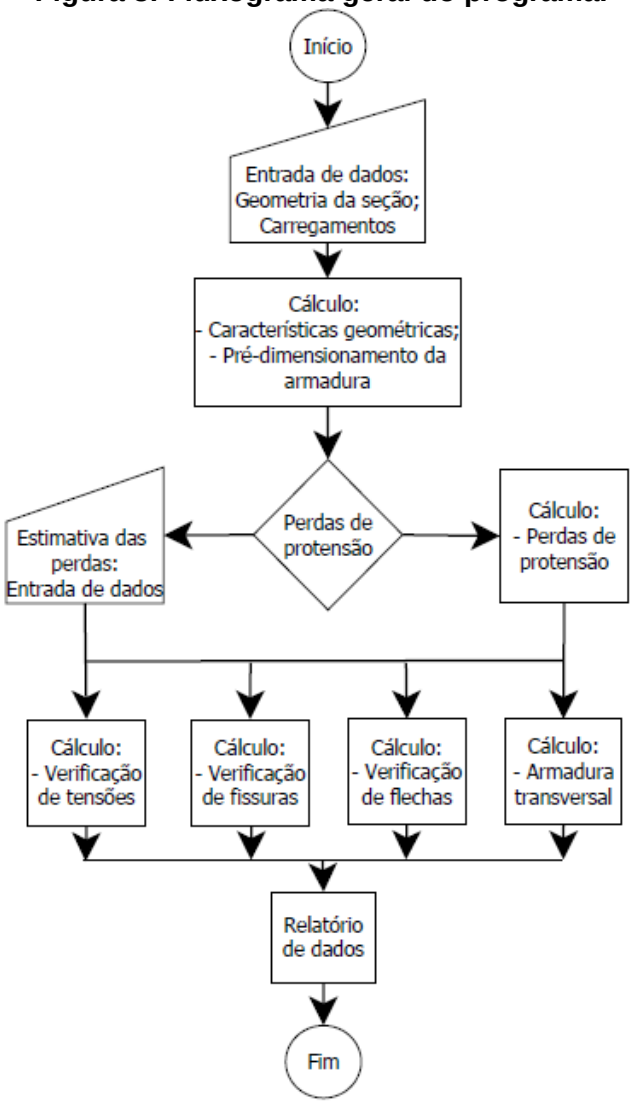
As principais hipóteses usadas nos procedimentos de cálculo e verificações foram destacadas e discutidas quanto à sua abrangência e aplicação. São analisados exemplos práticos para mostrar a aplicação dos princípios discutidos e testar também a programação desenvolvida. A partir dos exemplos são feitas amplas discussões apresentando os resultados em forma de tabelas e gráficos de maneira que o leitor possa rapidamente compreender o texto.

Há que se destacar que o desenvolvimento deste programa leva em consideração as alterações normativas apresentadas na NBR 6118:2014. Critérios como os valores simplificados do diagrama parábola-retângulo em função do tipo de concreto e os limites da razão x/d foram considerados para seu desenvolvimento.

Em relação ao programa propriamente dito, este foi desenvolvido com programação orientada ao objeto. Esta maneira de trabalhar otimizou seu desenvolvimento, permitindo o anexo de objetos previamente desenvolvidos como parte do programa, além de facilitar sua personalização às preferências do usuário.

De forma a reduzir possibilidades de erros e facilitar o trabalho de programação, foi necessário um planejamento. Isto pode ser feito com a análise do fluxograma do mesmo em que cada etapa possa ser programada separadamente, ou mesmo por programadores diferentes. O fluxograma inicial do programa é dado a seguir na Figura 3. Após iniciado o desenvolvimento do programa, algumas alterações não previstas no fluxograma foram necessárias. Essas alterações são decorrentes de fatores como facilidade de programar e atém mesmo ideias que surgiram de forma a tornar o programa mais abrangente. Por exemplo, o módulo de verificação, onde o usuário entra com todas as informações da armadura ativa e passiva, como o posicionamento na seção transversal, a tensão inicial, as perdas de protensão, e o programa retorna a tensão em cada um dos décimos de vão. Este módulo permite que o usuário compare diferentes soluções de armadura para uma mesma seção, propiciando um estudo mais detalhado dos limites máximos e mínimos que uma viga está sujeita.

Figura 3: Fluxograma geral do programa.



1.4 CONTEXTUALIZAÇÃO DO PROGRAMA: GRUPO CALCO

O presente programa faz parte de uma série de outros projetos constituintes do grupo Calco, formado por alunos de graduação e pós-graduação do Departamento de Engenharia Civil (DECiv) da UFSCar.

Precedendo este projeto pode-se citar o trabalho de Inforsato (2009) onde se desenvolveu um roteiro para determinação da armadura longitudinal e transversal de uma viga pré-tracionada. Por meio de exemplos práticos o autor realizou todas as etapas do dimensionamento da armadura de flexão e de cisalhamento de maneira bastante detalhada. Realizou, ainda, o cálculo das perdas de protensão e as verificações de tensão na seção.

Em seguida, a dissertação de Faleiros Junior (2010), que trata do dimensionamento da armadura de flexão de vigas protendidas e que apresentou uma série de exemplos práticos de dimensionamento e verificação. Neste, o autor fez uma abordagem mais detalhada das considerações normativas referentes à durabilidade e à ruptura. Muito embora essa discussão tenha sido feita considerando a norma antiga, muitas das informações mantiveram-se presentes na atual.

Finalmente, no trabalho de Scavassin (2012) o autor partiu para uma aplicação mais prática dos trabalhos de Inforsato (2009) e Faleiros Junior (2010). Neste trabalho foi desenvolvido um programa computacional para o cálculo da armadura ativa de uma viga pré-tracionada e com seção retangular composta. O autor aplica os conceitos de perdas protensão e verificação de tensão e ruptura para cada seção, aplica os conceitos de isolamento dos cabos, realiza o dimensionamento da armadura transversal e o detalhamento gráfico da viga.

1.5 ESCOPO DA DISSERTAÇÃO

No capítulo 1 apresenta-se uma introdução ao tema de protensão e seu contexto histórico no país. Também se descreve, em detalhes, os objetivos do trabalho assim como as técnicas e meios para concretizá-los.

No capítulo 2 é feita uma revisão bibliográfica abordando, principalmente, as informações técnicas já desenvolvidas e que serão utilizadas para o desenvolvimento deste projeto. Também se faz uma breve cronologia sobre os trabalhos anteriores ao grupo que este projeto se insere, demonstrando a motivação e a base técnica que precederam esta dissertação.

No capítulo 3 apresenta-se uma série de estudos que serviram de base para moldar a filosofia do programa. A partir desses estudos surgiu a motivação de se desenvolver um programa que não visasse apenas a otimização do processo de produção de projetos, mas

também de análise de soluções e sob diferentes considerações teóricas permitindo, por exemplo, sua adaptação a critérios normativos de diferentes países.

No capítulo 4 são descritos as considerações teóricas que foram feitas a fim de adaptar esses conceitos à lógica de programação. A forma como o programa deve interpretar as informações inseridas deve ser feita de maneira simples. Portanto, foi necessário se aprofundar nas teorias e verificar qual a maneira mais fácil de aplicá-las ao programa. Este capítulo descreve essas considerações e simplificações.

No capítulo 5 é demonstrado como se deve utilizar o programa. Um manual que demonstra nos mínimos detalhes e por meio de um exemplo prático como o usuário deve proceder com a entrada de informações.

No capítulo 6 são apresentados exemplos numéricos retirados ou baseados na literatura técnica com o propósito de validar e dar credibilidade ao programa, demonstrando que seus resultados são coerentes e consistentes com a realidade, dando ao usuário maior confiança na sua utilização. Nesse capítulo também é possível verificar quais foram as hipóteses e considerações utilizadas no programa por meio de exemplos que abordam sua mesma metodologia.

No capítulo 7 são feitas as conclusões por meio da análise dos resultados, propostas para outros trabalhos e limitações do programa.

No capítulo 8 é apresentado todo o material utilizado para o desenvolvimento deste projeto.

2 REVISÃO BIBLIOGRÁFICA

2.1 PRÉ-DIMENSIONAMENTO DA SEÇÃO

De acordo com Inforsato (2009), recomenda-se utilizar o coeficiente de rendimento de Guyon para determinação do tipo de seção mais eficiente para o concreto protendido.

Ainda, segundo o autor, apesar do parâmetro R indicar a seção mais vantajosa, outros fatores devem ser levados em consideração, tais como as necessidades arquitetônicas e estéticas definidas pelo departamento comercial das empresas.

De acordo com Nawy (1995), a definição da seção pode ser feita em função dos limites de tensão no concreto estabelecidos pela ACI e, também, pelos momentos decorrentes das ações atuantes.

Com as equações apresentadas pelo autor, são definidos os módulos de resistência à flexão mínimo da seção transversal da viga (admite-se um valor aproximado para o peso próprio da viga). É uma forma simples para pré-dimensionar a seção do elemento. Logicamente, devem-se realizar, em seguida, as devidas verificações.

Há de se ressaltar que esse tópico é meramente informativo. O programa não irá, a princípio, realizar o pré-dimensionamento da seção.

2.2 PRÉ-DIMENSIONAMENTO DA ARMADURA LONGITUDINAL

De acordo com Carvalho (2012, p. 265), “para pré-dimensionar ou calcular a armadura longitudinal de flexão podem ser usadas as condições de verificação no estado limite de fissuração ou a condição de estado limite último”. Ainda segundo o autor, até a década de 80, os projetistas preferiam considerar o estado limite de fissuração, onde “impondo tensões limites e conhecidos os esforços solicitantes e a excentricidade da armadura de protensão, fica determinada a força necessária de protensão". Em Nawy (1995), o autor se utiliza desse método de cálculo para pré-dimensionar a armadura ativa.

Como se pode concluir em Faleiros Junior (2010, p. 174), “nem sempre pode ser avaliada sem a realização dos cálculos, qual a situação predominante para a determinação da armadura longitudinal. Em certas situações as condições de serviço são determinantes e, em outras, o estado limite último”. Portanto, pode-se afirmar que o pré-dimensionamento

pode ser feito considerando tanto o ELS quanto o ELU desde que sejam feitas as verificações para o outro estado limite.

Durante o pré-dimensionamento da armadura longitudinal o projetista não dispõe de algumas informações necessárias para o correto dimensionamento do elemento. Dentre estas informações, pode-se citar a tensão final de protensão que é desconhecida pelo fato de não sabermos a magnitude das perdas de tensão. Neste caso, de acordo com Carvalho (2012), pode-se estipular um valor de perdas e calcular a área de armadura necessária em função do ELU ou ELS. Com a armadura pré-dimensionada, deve-se realizar um cálculo mais preciso dessas perdas, assim como as devidas verificações.

Não existe um valor estimado correto a ser adotado para as perdas. Inforsato (2009) considerou uma perda de 20% da tensão inicial, enquanto que Petrucelli (2009) adotou um valor de 25% para os tipos de concreto mais usual na construção civil.

Para seu correto cálculo, é necessário conhecer as deformações que ocorrem na cordoalha. Diferentemente do concreto armado convencional, onde a deformação da armadura ocorre apenas pelo carregamento, no concreto protendido se consideram três fontes diferentes para sua definição:

- pelo carregamento (ε_s);
- pelo pré-alongamento (ε_p); e
- até o estado de descompressão (ε_7).

Somando-se todos os 3 tipos de deformação, obtém-se seu valor total (ε_t) que será usado para definir a tensão atuante nas cordoalhas.

2.3 PERDAS DE PROTENSÃO

De acordo com Nawy (1995), não é desejável um alto grau de refinamento nas estimativas das perdas de protensão por causa da multiplicidade de fatores que as afetam. Uma estimativa mais realista, de acordo com o autor, pode ser obtida nas tabelas de perdas fixas apresentadas pela AASHTO e pelo Post-Tensioning Institute (PTI). Estão inclusas nas estimativas fixas as perdas referentes à deformação imediata do concreto, relaxação da armadura, fluência e retração do concreto. No entanto, o autor diz que a utilização de estimativas fixas devem ser aplicadas somente para projetos em condições padrões de: carregamento, concreto, controle de qualidade, procedimentos construtivos, condições ambientais. Uma análise mais detalhada deve ser feita para o caso de alguma dessas condições não serem atendidas.

De acordo com o autor, as perdas totais para elementos com pré-tração são dos seguintes tipos: deformação imediata elástica do concreto; relaxação da armadura; fluência e retração do concreto. Como se pode observar, não está incluída a perda por acomodação

da ancoragem. Não que ela não exista, mas, de acordo com o autor, esta pode ser facilmente resolvida por meio de um acréscimo de tensão durante o estiramento dos cabos.

De acordo com Lyn (1981, p. 91), “considerar as perdas tabeladas para condições usuais apresenta resultados bastante satisfatórios, mas para casos fora do comum é necessária uma estimativa mais refinada”.

Collins (1997) sugere a utilização da tabela 6-3 de seu livro, baseada nos estudos de Zia, Preston, Scott e Workman, para definição das perdas diferidas.

Para as perdas iniciais e considerando aço de baixa relaxação, o autor sugere o valor fixo de 7,5% para estruturas comuns.

A NBR 6118:2014 afirma que o projetista deve considerar as perdas de protensão em relação à tensão inicial. Para o caso da pré-tração, essas perdas são classificadas em: perdas iniciais, perdas imediatas e perdas progressivas (ou ao longo do tempo). Nessa nova versão da norma houve alteração significativa no cálculo da perda por retração além do acréscimo de novos valores relativos às novas classes de concreto.

Em Carvalho (2012), as perdas de protensão são agrupadas em perdas imediatas (deformação da ancoragem, deformação imediata do concreto e relaxação da armadura) e perdas ao longo do tempo (retração do concreto, fluência do concreto e relaxação da armadura, novamente).

A explicação para cada tipo de perda, assim como as equações, ábacos e tabelas para defini-las podem ser encontradas na norma ou em diversas obras técnicas que tratem do assunto. Estas serão descritas a seguir:

2.3.1 Perda por deformação da ancoragem

No momento de se realizar a ancoragem, ocorre um pequeno encurtamento dos cabos de protensão ocasionado pela acomodação do cone macho introduzido no cone fêmea. Para a protensão com aderência inicial este tipo de perda ocorre de maneira uniforme ao longo de toda a cordoalha no momento em que o cabo é ancorado na extremidade oposta da pista. Já para a protensão com aderência final e sem aderência, a perda ocorre de maneira não uniforme por causa da existência do atrito cabo-bainha que, de acordo com Carvalho (2012), “impede a livre movimentação do cabo para o interior da estrutura”. Assim, para essa situação, observa-se uma diminuição na perda de protensão na cordoalha até um trecho a partir do qual a tensão se mantém constante.

2.3.2 Perda por deformação imediata do concreto

A transferência dos esforços que passa das cordoalhas para o concreto causa uma compressão no elemento e, consequentemente, uma deformação. Esta deformação ocasiona um encurtamento das cordoalhas e uma perda de protensão.

Nos elementos com aderência final e sem aderência os cabos não são tracionados ao mesmo tempo. Um cabo somente será tracionado quando o anterior já estiver ancorado, o que significa que o último cabo ancorado irá causar um encurtamento no concreto e uma perda de protensão nos cabos que já estão ancorados. A perda não é a mesma para cada cabo. Como forma de simplificar os cálculos, a NBR 6118:2014 apresenta uma fórmula que calcula a perda média de protensão.

Nos casos com pré-tração, todos os cabos são previamente estirados, logo, ocorre a mesma perda em todos eles. Perda esta que varia para cada décimo de vão da viga.

2.3.3 Retração

A perda por retração, de acordo com Carvalho (2012), é um fenômeno que se inicia “logo após o lançamento do concreto” e “é devida principalmente à saída da água que não reage com o cimento”. Este ocorre como decorrência da variação do volume e consequente encurtamento do elemento estrutural. Havendo o encurtamento do concreto, consequentemente ocorrerá o encurtamento dos cabos.

Para o cálculo da perda por retração deve-se iniciar calculando o valor da deformação do concreto, devido à retração, no intervalo requerido.

2.3.4 Fluência

A fluência do concreto são as deformações que ocorrem no material sob um esforço constante (fluência pura). Por causa disso, o elemento se mantém sob constante encurtamento.

Analisando mais detalhadamente, sabe-se que, na prática, não existe a fluência pura, já que a tensão nas cordoalhas é reduzida ao longo do tempo. Mas por conveniência de cálculo e estando a favor da segurança, considera-se a fluência pura. De acordo com Petrucelli (2009, p. 48), como as ações que possuem caráter permanente são as que provocam a fluência, “pode-se considerar a combinação quase permanente da NBR 6118:2014 como a causadora da fluência e, portanto, consideram-se os efeitos de protensão, peso próprio, sobrecarga permanente e 20% da carga acidental”.

Ainda segundo a autora, calculam-se os coeficientes de fluência $\varphi(t,t0)$ para cada etapa de carregamento considerando seus efeitos isoladamente. O cálculo dos coeficientes

é feito segundo as equações do item A.2.2.3 do anexo A da NBR 6118:2014. Com os valores dos coeficientes de fluência, calcula-se a perda de protensão por fluência.

O cálculo da fluência depende dos momentos fletores de carregamento da viga, logo seu valor varia ao longo de seu comprimento.

2.3.5 Relaxação da armadura

Sempre que uma cordoalha é estirada, surgem tensões que tendem, ao longo do tempo, diminuírem como causa da relaxação do material. Assim como a fluência e a retração, calcula-se esse tipo de perda considerando-se uma relaxação pura (sob força constante nos cabos), o que não ocorre na prática sendo, contudo, favorável a segurança.

2.3.6 Simultaneidade das perdas

Entende-se que durante a ocorrência das perdas de protensão decorrentes da retração, fluência e relaxação da armadura, exista uma interação entre elas que interfere na forma e na intensidade em que estas ocorrem. Normalmente, nos raros momentos em que as perdas são calculadas (e não estimadas), o projetista dificilmente considera essa simultaneidade entre elas, superestimando, muitas vezes, os valores totais de perdas de tensão. Outro motivo que talvez leve estes projetistas a não levar a consideração da simultaneidade é que considerá-las isoladamente implica em um maior grau de segurança da estrutura. No entanto, sendo possível o refinamento de cálculo, não há porque não considerá-las.

Em Carvalho (2012), o autor apresenta algumas considerações que permitem a utilização dos processos de cálculo. A princípio, o autor relaciona a existência dessas interações com a aderência concreto-armadura.

A interação pode ser calculada por dois métodos diferentes:

a) Método simplificado para o caso de fases únicas de operação: esse caso acontece em situações em que os carregamentos permanentes e protensão ocorrem em uma única etapa.

Utiliza-se esse caso quando satisfeitas as seguintes condições:

- Quando a concretagem do elemento e a protensão ocorrem em um intervalo de tempo bem pequeno, a ponto de se desprezar os efeitos de simultaneidade entre as fases.
- A distância entre os cabos de protensão é relativamente pequena a ponto de se adotar um cabo equivalente (de área igual à soma das áreas dos cabos isolados e com centro de gravidade coincidente).

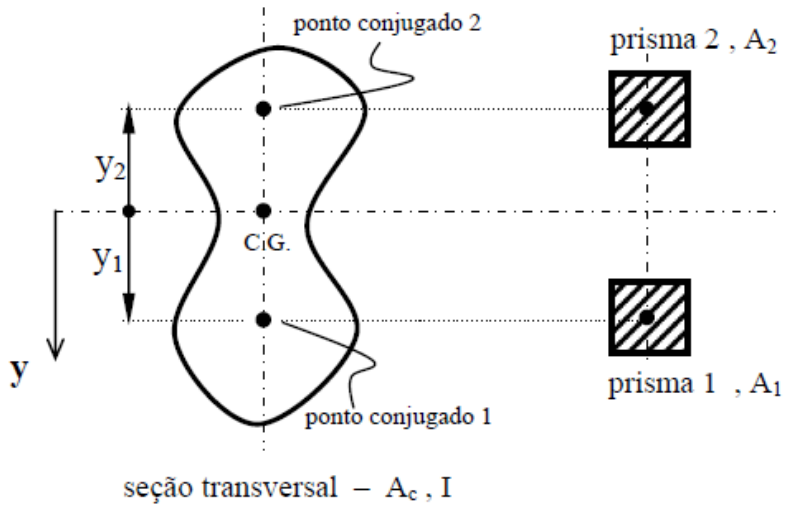
b) Método geral de cálculo: utiliza-se esse caso quando as hipóteses do método anterior não são atendidas.

Assim, quando a protensão e os carregamentos permanentes ocorrem em períodos distintos (em intervalos de tempo consideráveis) a fluência, a retração e a relaxação devem ser calculadas separadamente para cada uma dessas etapas.

2.3.7 Método dos prismas equivalentes

Pelo método dos prismas equivalentes é possível analisar a variação de tensões que ocorrem na armadura e no concreto de determinada seção possibilitando, assim, considerar os fenômenos de retração, fluência e relaxação de forma isolada. Em Silva (2003) e Araujo (2007) são apresentadas as 3 propriedades básicas que caracterizam o métodos, que serão reproduzidas nesse trabalho, assim como suas deduções matemáticas:

Figura 4: Conceito de prismas equivalentes numa seção transversal plana.

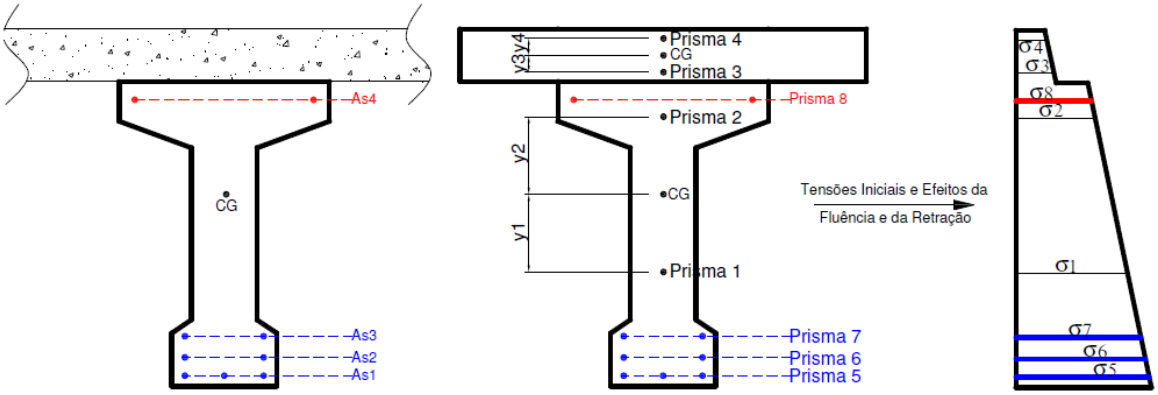


Fonte: Silva, 2003.

- 1ª propriedade: Uma força normal N_1 atuando no ponto conjugado 1 não produz tensão no ponto conjugado 2, e vice-versa, uma força normal N_2 atuando no conjugado 2 não produz tensão no ponto conjugado 1;
- 2ª propriedade: Uma força normal N atuando no centro de gravidade deve ser dividida conforme os braços de alavanca em relação aos pontos conjugados, ou seja, deve ser ponderada da mesma forma que as áreas dos prismas;
- 3ª propriedade: Para o cálculo das tensões normais provocadas por um momento fletor M aplicado à seção, basta determinar o binário correspondente ao braço de alavanca z formado pela distância entre os pontos conjugados ($z = y_1 – y_2$).

Araujo (2008) sugere a utilização deste método para calcular a redistribuição de tensões normais que surgem como consequência da retração e da deformação lenta dos elementos constituintes de uma seção composta, sabendo que o concreto de cada parte apresenta características e idades diferentes. Logo, cada parte dessa seção é substituído por um prisma, assim como cada camada de armadura mantendo-se, sempre, a simetria em relação ao centro de gravidade da peça.

Figura 5: Prismas equivalentes aplicados à redistribuição de esforços.



Fonte: Araujo, 2007.

Na Figura 5 acima se observa a esquematização da peça em prismas equivalentes assim como seu diagrama de tensões normais composto pelas tensões iniciais, fluência e retração.

O cálculo da variação da deformação decorrente da fluência, da retração e da relaxação para cada prisma corresponde aos coeficientes da reta definida pela hipótese de Navier-Bernoulli das seções planas. Por meio deste coeficiente torna-se possível calcular as redistribuições de tensões assim como as perdas progressivas na armadura.

2.4 VERIFICAÇÕES EM SERVIÇO E DIMENSIONAMENTO PARA A RUPTURA

As verificações a serem realizadas nos elementos protendidos possuem algumas particularidades em relação aos elementos com armadura passiva. No caso do protendido deve ser previsto, para o ELS, as verificações de tensões nas bordas superiores e inferiores da seção mais solicitada para o tempo infinito (quando ocorreram todas as perdas previstas, ou seja, tensão de protensão mínima e carregamento solicitante máximo).

No concreto protendido as verificações a serem analisadas em serviço são normatizadas pela NBR 6118:2014. De acordo com esta, deve-se estabelecer o tipo de protensão (pré ou pós-tração) e a classe de agressividade ambiental (CAA) da edificação.

Com isso, define-se qual o nível de protensão a ser considerado: protensão parcial, limitada ou completa.

Que fique claro que o nível de protensão não está relacionado com a intensidade do estiramento das cordoalhas. A protensão completa não implica, necessariamente, em uma aplicação de protensão maior em comparação às outras duas. A classificação está relacionada com o nível de proteção que o elemento deve possuir em relação à fissuração do concreto para determinado tipo de combinação.

A norma relaciona o nível de protensão estabelecido com as verificações e combinações a serem feitas.

Observa-se, portanto, que para o caso de pré-tração com CAA I ou pós-tração com CAA I e II (protensão parcial), deve-se impedir uma abertura de fissuras superior a 0,2 mm para a combinação frequente. Para pré-tração com CAA II ou pós-tração com CAA III e IV (protensão limitada), a norma não permite o aparecimento de fissuras para a combinação frequente (estado limite de formação de fissuras – ELS-F) e nem mesmo o surgimento de tração para a combinação quase permanente (estado limite de descompressão – ELS-D). E, finalmente, para pré-tração com CAA III e IV (protensão completa), deve-se atentar ao surgimento de fissuras na combinação rara e ao surgimento de tração na combinação quase permanente.

No item 11.8 da NBR 6118:2014 são apresentadas quais as ações a serem consideradas, assim como os coeficientes de ponderação, para cada combinação.

Na tabela 11.2 da NBR 6118:2014, estão representados os valores do coeficiente γ_{f2} para cada combinação em serviço.

Para a verificação no ELU em vazio (sem a ocorrência de perdas diferidas e somente carregamento de peso próprio) a norma brasileira estabelece algumas hipóteses:

- A resistência característica do concreto a ser considerada é o f_{ckj}, respectiva à idade do concreto no momento da protensão;
- São considerados os seguintes coeficientes de ponderação:

$\gamma_c = 1,2$ e $\gamma_p = 1,15$;

$\gamma_p = 1,0$ (p/ pré-tração) e $\gamma_p = 1,1$ (p/ pós-tração);

$\gamma_f = 1,0$ (p/ ações desfavoráveis) e $\gamma_f = 0,9$ (p/ ações favoráveis).

A norma também permite uma verificação simplificada desde que sejam adotadas algumas considerações. No item 17.2.4.3.2 da NBR 6118:2014, pode-se encontrar as condições a serem estabelecidas para a verificação simplificada. “Admite-se que a segurança em relação ao estado limite último no ato de protensão seja verificada no estádio I (concreto não fissurado e comportamento elástico linear dos materiais), desde que as seguintes condições sejam satisfeitas:

- A tensão de compressão no concreto na idade de aplicação da protensão, considerando os coeficientes de ponderação, não deve ultrapassar o valor de 70% da resistência característica do concreto nessa etapa do carregamento ($0,7*f_{ck,j}$);
- A tensão de tração no concreto deve ser menor que 1,2 da sua resistência à tração ($f_{ct,m}$) correspondente à idade de aplicação da protensão ($f_{ck,j}$);
- Quando houver tração, será prevista armadura para resistir a essas tensões calculadas para o estádio II. A força nesta armadura deverá ser igual à força resultante das tensões de tração calculadas no estádio I. A força em questão não deve causar aumento de tensão na armadura na ordem de 150 MPa (p/ fios e barras lisas) e 250 MPa (p/ barras nervuradas com $\eta_b > 1,5$).

Portanto, para cada seção e combinações analisadas, é necessária a realização de quatro verificações de tensão (compressão e tração excessivas em ambas as bordas). Abaixo podem ser analisadas as expressões para verificação na borda superior e inferior, respectivamente:

$$\sigma_s = \frac{N_p}{A} - \frac{N_p * e}{W_s} \pm \frac{\sum M}{W_s} \quad (Eq.\,2.4.1)$$

e,

$$\sigma_i = \frac{N_p}{A} + \frac{N_p * e}{W_i} \pm \frac{\sum M}{W_s} \quad (Eq.\,2.4.2)$$

Onde,

N_p – Normal de protensão;

A – Área da seçao transversal;

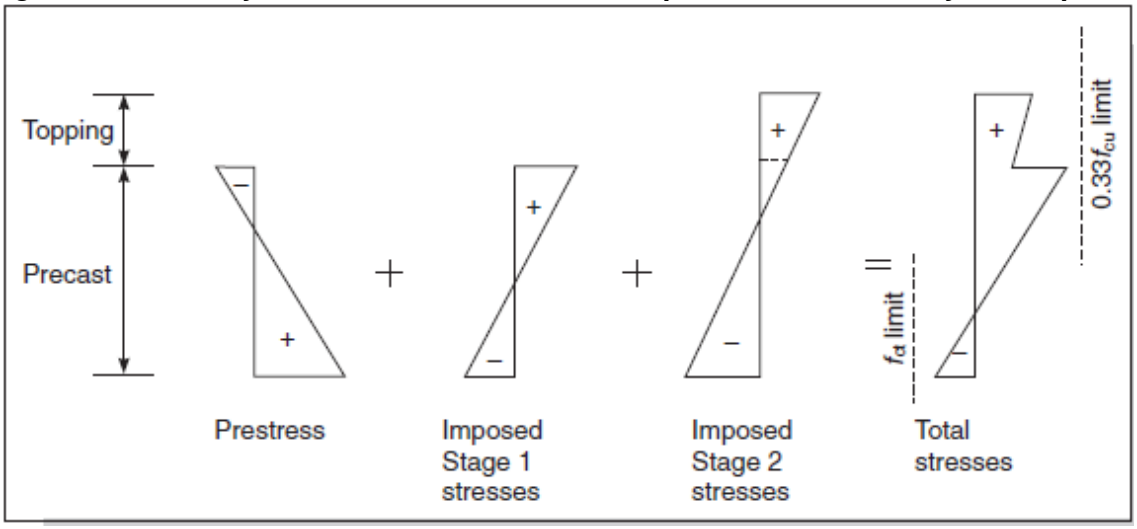
e – Excentricidade dos cabos de protensão;

W_i / W_s – Módulo de resistência à flexão da seção inferior e superior, respectivamente;

M – Momentos oriundos dos carregamentos externos.

Em Elliot (2002), o carregamento permanente atuante na viga antes do endurecimento da capa, no caso da seção composta, deve ser considerado resistido apenas pela seção pré moldada, assim como os momentos oriundos da protensão (ver Figura 6). De acordo com o autor, o acréscimo de seção auxilia apenas no carregamento existente após o endurecimento dessa capa de concreto que, normalmente, diz respeito à carga de utilização e revestimentos. Por isso que devem ser calculadas as características geométricas da seção simples e da seção composta.

Figura 6: Distribuição de tensões em elementos protendidos com seção composta.



Fonte: Elliot, 2002.

Em relação ao dimensionamento da armadura longitudinal de flexão de elementos com flexão simples, a norma estabelece algumas hipóteses básicas presentes no item 17.2.2 da mesma e aqui reproduzida:

a) as seções transversais se mantém planas após a deformação;

b) a deformação das barras passivas aderentes ou o acréscimo de deformação das barras ativas aderentes;

c) as tensões de tração no concreto, normais à seção transversal, devem ser desprezadas no ELU;

d) a distribuição de tensões no concreto se faz de acordo com o diagrama parábola-retângulo. Esse diagrama pode ser substituído pelo retângulo de profundidade y = λx, onde o valor do parâmetro λ pode ser tomado igual a:

- λ = 0,8 para f_{ck} < 50 MPa; ou
- λ = 0,8 – (f_{ck} – 50) / 400 para f_{ck} > 50 MPa.

Onde a tensão constante atuante até a profundidade y pode ser tomada igual a:

- $\alpha_c.f_{cd}$ no caso da largura da seção, medida paralelamente à linha neutra, não diminuir a partir desta para a borda comprimida;

Sendo αc definido como:

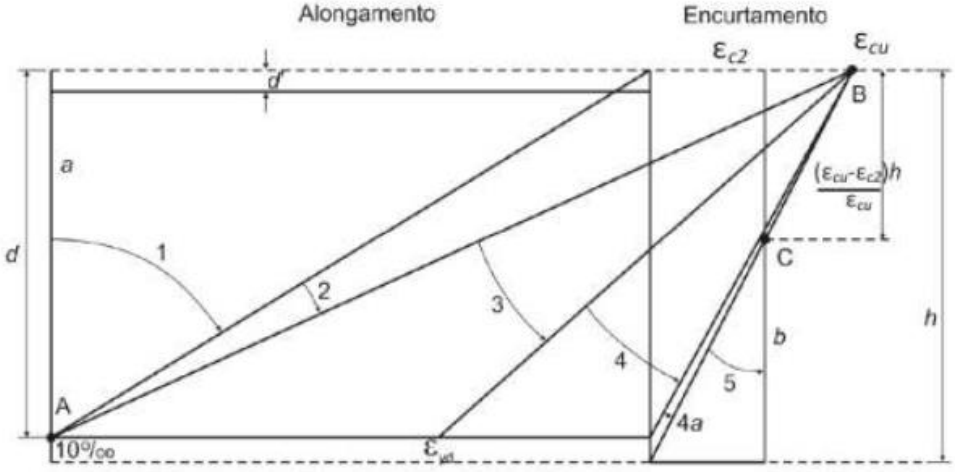
- para concretos de classes até C50: α_c = 0,85;
- para concretos de classes de C50 até C90: α_c = 0,85*[1,0 - (f_{ck} - 50) / 200].

As diferenças de resultados obtidos com esses dois diagramas são pequenas e aceitáveis, sem necessidade de coeficiente de correção adicional.

e) a tensão nas armaduras deve ser obtida a partir dos diagramas tensão-deformação;

f) o estado limite último é caracterizado quando a distribuição das deformações na seção transversal pertencer a um dos domínios definidos na Figura 7;

Figura 7: Domínios de ruptura de uma seção transversal de concreto armado.



Fonte: NBR 6118:2014.

Ruptura convencional por deformação plástica excessiva:

- Reta a: tração uniforme;
- Domínio 1: tração não uniforme, sem compressão;
- Domínio 2: flexão simples ou composta sem ruptura à compressão do concreto;

Ruptura convencional por encurtamento limite do concreto:

- Domínio 3: flexão simples (seção subarmada) ou composta com ruptura à compressão do concreto e com escoamento do aço ($\varepsilon_s \geq \varepsilon_{yd}$);
- Domínio 4: flexão simples (seção superarmada) ou composta com ruptura à compressão do concreto e aço tracionado sem escoamento ($\varepsilon_s < \varepsilon_{yd}$);
- Domínio 4a: flexão composta com armaduras comprimidas;
- Domínio 5: compressão não uniforme, sem tração;
- Reta b: compressão uniforme.

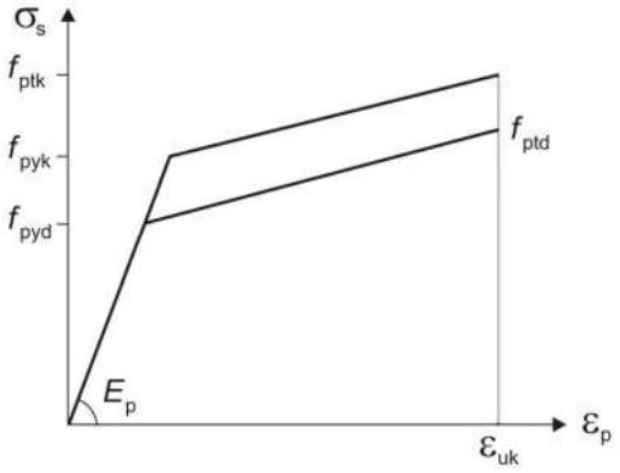
A nova revisão da norma também estabelece limites na relação x/d para que seja garantida as condições de ductilidade.

- $x/d \leq 0{,}45$, para $f_{ck} \leq 50$ MPa;
- $x/d \leq 0{,}35$, para $f_{ck} > 50$ MPa.

Segundo Carvalho (2012), a verificação para o ELU no tempo infinito se baseia na segurança à ruptura da seção transversal. Em função do momento atuante e das características da armadura de protensão (área de armadura e pré-alongamento), determina-se a posição da linha neutra (e, consequentemente, o domínio em que a seção trabalha) que leve equilíbrio à seção (Fc = Fp). Assim, têm-se o momento máximo que a seção é capaz de resistir e que deve ser comparado ao momento atuante. Se o momento atuante for menor que o momento resistente, a seção estará verificada.

Outro assunto relevante a ser descrito é em relação ao diagrama tensão-deformação para aços de armadura ativa. A NBR 6118:2014 permite a utilização do diagrama simplificado mostrado na Figura 8, válido para temperaturas entre -20°C e 150°C. É necessário informações sobre a resistência ao escoamento convencional (f_{pyk}), a resistência à tração (f_{ptk}) e o alongamento após a ruptura (ε_{uk}) das cordoalhas.

Figura 8: Diagrama tensão-deformação para aços de armadura ativa.



Fonte: NBR 6118:2014.

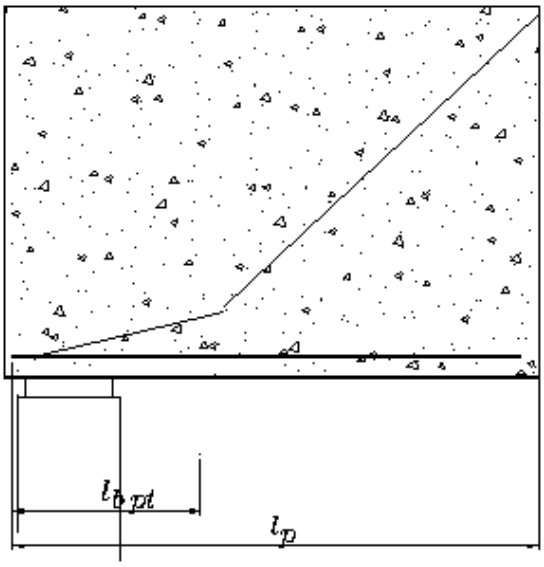
2.5 ANCORAGEM (PARA PÓS-TRAÇÃO), COMPRIMENTO DE TRANSFERÊNCIA E COMPRIMENTO DE REGULARIZAÇÃO DO ESFORÇO DE PROTENSÃO

Diferentemente do que ocorre no sistema com pós-tração, onde existe uma cunha que efetiva a ancoragem nas extremidades da viga e que transfere os esforços do cabo para o concreto, na pré-tração a transferência de esforços ocorre, exclusivamente, por meio da aderência aço-concreto, sem necessidade de qualquer sistema de ancoragem.

Em relação ao comprimento de transferência (l_{bpt}), este é definido pela NBR 6118:2014 como sendo o “comprimento necessário para transferir, por aderência, a totalidade da força de protensão ao fio” (ao fio entenda-se fio ou cordoalha). E esta pode ser calculada pelas equações apresentadas no anexo D da referida norma.

Quando da transferência de esforços entre as cordoalhas e o concreto na extremidade da viga, sabe-se que esta segue uma trajetória linear e crescente ao longo de um trecho determinado como comprimento de regularização (l_p).

Figura 9: Comprimento de transferência e de regularização.



Fonte: Carvalho (2012).

Na extremidade da viga, as tensões não são distribuídas por toda a seção. Na realidade ela atinge uma pequena região que vai crescendo proporcionalmente ao longo da viga até um trecho determinado pelo comprimento de regularização, a partir do qual a protensão é distribuída por toda a seção. Há de se ressaltar que existe uma concentração muito grande de tensões na seção de transferência de tensões. Para tanto, deve ser previsto o dimensionamento de armadura que resista a esse excesso de tensão, chamada de armadura de fretagem, que é uma armadura de confinamento. O cálculo do comprimento de regularização é definido no item 9.6.2.3 da NBR 6118:2014.

2.6 DESLOCAMENTO MÁXIMO EM VIGAS DE CONCRETO (FLECHAS)

2.6.1 Cálculo das flechas em vigas de concreto armado

Em elementos que seguem o comportamento elástico-linear, o cálculo de deslocamentos em vigas (a) sob determinado carregamento podem ser definidas pelas teorias gerais da linha elástica (Timoshenko, 1966).

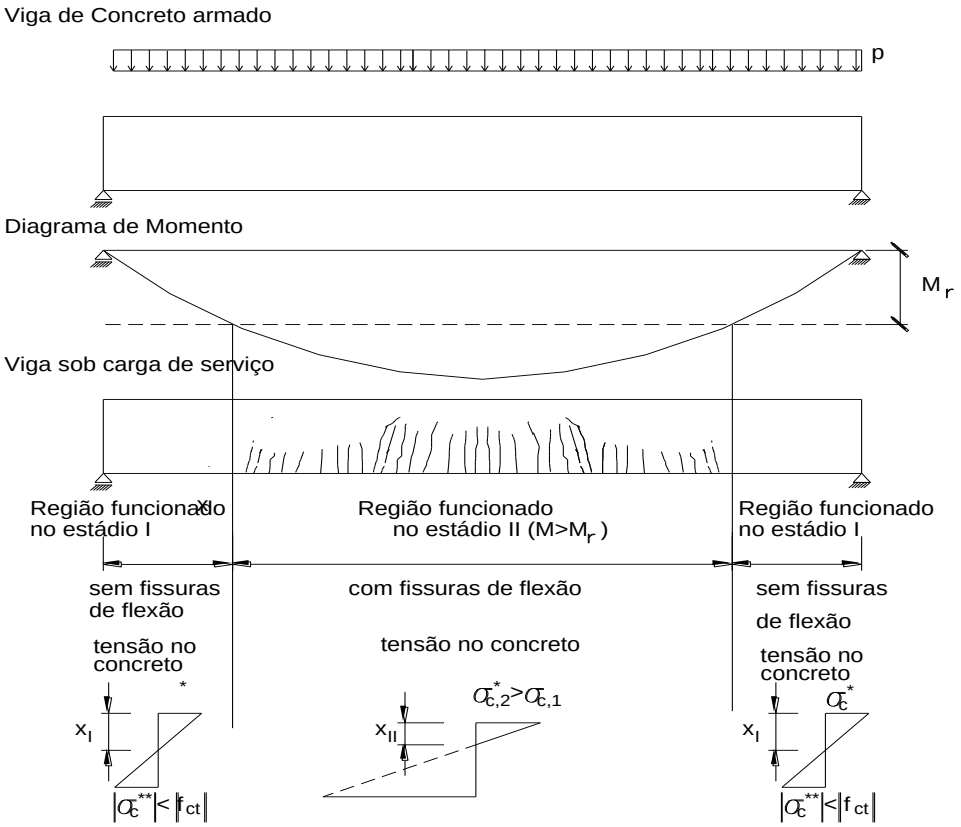
$$a = \int_0^L \frac{M_0 * M_1}{EI} dx \quad (Eq. 2.6.1.1)$$

Em Carvalho e Figueiredo Filho (2009), define-se que o maior deslocamento que surge em uma viga é denominado “flecha”. Em estruturas de concreto armado, as características não-homogêneas do material dificultam o cálculo correto da flecha. Para exemplificar tal problema neste tipo de viga, pode-se observar seu comportamento frente às

exigências estruturais. Em regiões onde o momento atuante é superior ao momento de fissuração, é possível observar o surgimento de fissuras e, consequentemente, redução do momento de inércia da seção. Diferentemente das seções onde este momento é inferior. Logo, o elemento apresenta rigidez diferente ao longo de seu comprimento.

A Figura 10, retirado de Carvalho e Figueiredo Filho (2009), ilustra esse comportamento.

Figura 10: Viga de concreto armado simplesmente apoiada sob ações de serviço.



Fonte: Carvalho e Figueiredo Filho (2009).

O fato desse tipo de elemento apresentar variações ao longo de seu comprimento, acentua seu comportamento não-linear, o que nos obriga a adotar algumas simplificações para considerar tais características no cálculo de flechas em vigas de concreto armado.

Carvalho e Figueiredo Filho (2009) definem algumas características dos estádios de deformação de uma viga de concreto:

1-) Estádio I (estado elástico):

- Diagrama tensão deformação linear ao longo da seção;
- Tensões nas fibras comprimidas são proporcionais às deformações;
- Sem fissuras visíveis.

2-) Estádio II (estado fissurado):

- Considera-se que apenas o aço resiste às tensões de tração;
- A tensão na região comprimida do concreto se mantém linear;
- Fissuras visíveis.

3-) Estádio III (estado de ruptura):

- A fibra mais comprimida do concreto começa a plastificar;
- Quase todas as fibras trabalham com sua tensão máxima;
- Fissuras se aproximam da linha neutra reduzindo a região comprimida do concreto;
- Distribuição de tensões no concreto ocorre segundo um diagrama curva-retângulo.

Portanto, as características geométricas e a rigidez da seção variam conforme o estádio de deformação no qual esta esteja. Para seções no estádio I, o momento de inércia pode ser calculado, segundo a NBR6118:2014, para a seção bruta, embora alguns autores recomendem calculá-la para a seção homogeneizada do concreto e aço.

De forma a simplificar o cálculo da flecha sem, necessariamente, considerar a variação da inércia nos estádios I e II, a NBR 6118:2014 permite utilizar o modelo proposto por Branson (1968) de uma inércia média entre essa característica dos estádios I e II. Isto facilita o cálculo da integral já que a rigidez continua a ser uma constante. Este modelo, segundo Carvalho e Figueiredo Filho (2009) se baseia em um “método semiprobabilístico, no qual toma a variação da tensão ao longo da seção transversal e ao longo do comprimento de maneira simplificada, utilizando expressões empíricas que fornecem valores médios de inércia. Dessa forma, Branson procura traduzir aproximadamente o efeito da fissuração do concreto, quando submetido à flexão, no cálculo das deformações imediatas”.

Outro efeito de relativa importância para a definição da flecha é a fluência. Esta é caracterizada por um carregamento permanente que provoca deformações (reversíveis e irreversíveis) ao longo do tempo. As parcelas de deformação da fluência são divididas em: rápida, que é irreversível e ocorre nas primeiras 24 horas do carregamento, e lenta, com uma parcela reversível e outra irreversível.

Logo, o cálculo da flecha final deve considerar, além da flecha imediata, o efeito da fluência ao longo do tempo que provoca um aumento considerável em seu valor.

2.6.2 Método dos momentos estáticos das áreas

De acordo com Timoshenko (1966), é possível calcular as deformações em uma viga sob um comportamento elástico - linear partindo de seu diagrama de momento fletor. Este é caracterizado por método dos momentos estáticos das áreas. Suas deduções podem ser analisadas na obra Resistência dos Materiais – Vol. 1, Timoshenko (1966).

Resumidamente, por meio do diagrama de momento fletor obtido pelos carregamentos e pela reação de apoio (ambos apresentados separadamente), é possível definir as deformações ao longo de um elemento linear. Ou seja:

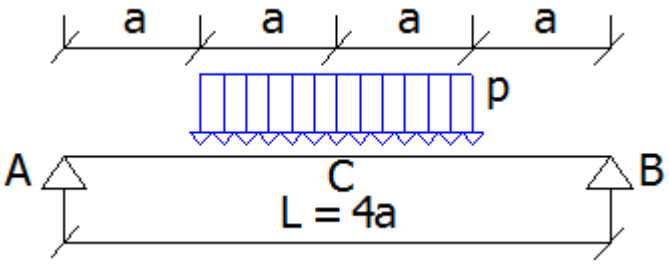
$$\delta = \int_{A}^{B} \frac{1}{EI} * xMdx \quad (Eq. 2.6.2.1)$$

Onde A é o início do trecho e B é o final.

A equação diz que a somatória do produto de cada elemento de área do diagrama de momento fletor (M*dx), multiplicado por sua distância até a reta vertical que passa pelo ponto B (ou seja, a somatória do momento estático de cada elemento de área) e dividido pela rigidez do elemento, resulta na deformação no ponto requerido.

O exemplo abaixo, retirado do livro Resistência dos Materiais – Mecânica dos Materiais, Beer, Johnston Jr e DeWolf (2006), exemplifica bem as teorias apresentadas.

Figura 11: Esquema de carregamento da viga.



Uma viga é carregada da seguinte forma e é necessário definir a deformação no meio do vão (ponto C).

As reações de apoio equivalem a:

$$R_a = R_b = pa$$

Em seguida, desenha-se o diagrama de momento fletor para a parte AC da viga. Este diagrama deve ser desenhado em partes, ou seja, um diagrama para a reação de apoio e outra para o carregamento distribuído atuante.

Figura 12: Diagrama de corpo livre de metade da viga.

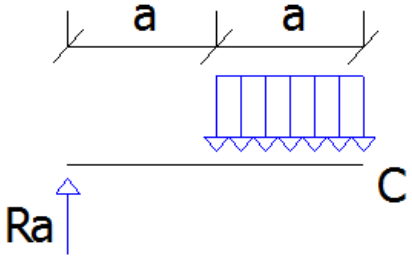
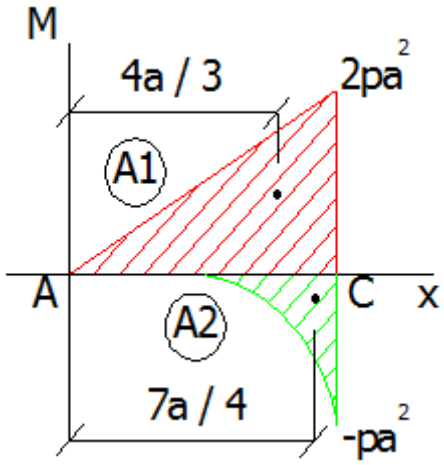


Figura 13: Diagrama de momento fletor de metade da viga para cada carregamento e reação de apoio.



As áreas de cada um dos diagramas são calculadas a seguir:

$$\mathrm{A_1} = \frac{1}{2} * 2\mathrm{pa^2} * 2\mathrm{a} = 2\mathrm{pa^3} \quad (Eq.\,2.6.2.2)$$

$$\mathrm{A_2} = -\frac{1}{3} * \frac{\mathrm{pa^2}}{2} * \mathrm{a} = -\frac{\mathrm{pa^3}}{6} \quad (Eq.\,2.6.2.3)$$

Aplicando a equação que relaciona a integral dos momentos estáticos, define-se a flecha no meio do vão ($y_{máx}$).

$$\mathrm{y_{máx}} = \frac{2\mathrm{pa^3}}{\mathrm{EI}} * \frac{4\mathrm{a}}{3} + \left(-\frac{\mathrm{pa^3}}{6\mathrm{EI}}\right) * \frac{7\mathrm{a}}{4} = \frac{19\mathrm{pa^4}}{8\mathrm{EI}} = \frac{19\mathrm{pL^4}}{2048\mathrm{EI}} \quad (Eq.\,2.6.2.4)$$

Como se pode ver, foi possível calcular a deformação no meio do vão utilizando apenas o momento estático do diagrama de momento fletor.

2.6.3 Verificação do estado limite de deformação excessiva

Segundo a NBR 6118:2014, o estado limite de deformação excessiva ocorre quando os limites de deformações são atingidos para a utilização normal. Esses limites são demonstrados na Tabela 1.

Tabela 1: Limites para deslocamentos (parte 1 de 2).

Tipo de efeito	Razão da limitação	Exemplo	Deslocamento a considerar	Deslocamento-limite
Aceitabilidade sensorial	Visual	Deslocamentos visíveis em elementos estruturais	Total	$\ell/250$
	Outro	Vibrações sentidas no piso	Devido a cargas acidentais	$\ell/350$
Efeitos estruturais em serviço	Superfícies que devem drenar água	Coberturas e varandas	Total	$\ell/250$ [a]
	Pavimentos que devem permanecer planos	Ginásios e pistas de boliche	Total	$\ell/350$+ contraflecha [b]
			Ocorrido após a construção do piso	$\ell/600$
	Elementos que suportam equipamentos sensíveis	Laboratórios	Ocorrido após nivelamento do equipamento	De acordo com recomendação do fabricante do equipamento
Efeitos em elementos não estruturais	Paredes	Alvenaria, caixilhos e revestimentos	Após a construção da parede	$\ell/500$ [c] e 10 mm e $\theta = 0{,}0017$ rad [d]
		Divisórias leves e caixilhos telescópicos	Ocorrido após a instalação da divisória	$\ell/250$ [c] e 25 mm
		Movimento lateral de edifícios	Provocado pela ação do vento para combinação frequente ($\psi_1 = 0{,}30$)	$H/1\,700$ e $H_i/850$ [e] entre pavimentos [f]
		Movimentos térmicos verticais	Provocado por diferença de temperatura	$\ell/400$ [g] e 15 mm

Fonte: NBR 6118:2014.

Tabela 2: Limites para deslocamentos (parte 2 de 2).

Tipo de efeito	Razão da limitação	Exemplo	Deslocamento a considerar	Deslocamento-limite
Efeitos em elementos não estruturais	Forros	Movimentos térmicos horizontais	Provocado por diferença de temperatura	H_i/500
		Revestimentos colados	Ocorrido após a construção do forro	ℓ/350
		Revestimentos pendurados ou com juntas	Deslocamento ocorrido após a construção do forro	ℓ/175
	Pontes rolantes	Desalinhamento de trilhos	Deslocamento provocado pelas ações decorrentes da frenação	H/400
Efeitos em elementos estruturais	Afastamento em relação às hipóteses de cálculo adotadas	Se os deslocamentos forem relevantes para o elemento considerado, seus efeitos sobre as tensões ou sobre a estabilidade da estrutura devem ser considerados, incorporando-os ao modelo estrutural adotado.		

[a] As superfícies devem ser suficientemente inclinadas ou o deslocamento previsto compensado por contra-flechas, de modo a não se ter acúmulo de água.

[b] Os deslocamentos podem ser parcialmente compensados pela especificação de contraflechas. Entretanto, a atuação isolada da contraflecha não pode ocasionar um desvio do plano maior que ℓ/350.

[c] O vão ℓ deve ser tomado na direção na qual a parede ou a divisória se desenvolve.

[d] Rotação nos elementos que suportam paredes.

[e] H é a altura total do edifício e H_i o desnível entre dois pavimentos vizinhos.

[f] Esse limite aplica-se ao deslocamento lateral entre dois pavimentos consecutivos, devido à atuação de ações horizontais. Não podem ser incluídos os deslocamentos devidos a deformações axiais nos pilares. O limite também se aplica ao deslocamento vertical relativo das extremidades de lintéis conectados a duas paredes de contraventamento, quando H_i representa o comprimento do lintel.

[g] O valor ℓ refere-se à distância entre o pilar externo e o primeiro pilar interno.

NOTAS

1 Todos os valores-limites de deslocamentos supõem elementos de vão ℓ suportados em ambas as extremidades por apoios que não se movem. Quando se tratar de balanços, o vão equivalente a ser considerado deve ser o dobro do comprimento do balanço.

2 Para o caso de elementos de superfície, os limites prescritos consideram que o valor ℓ é o menor vão, exceto em casos de verificação de paredes e divisórias, onde interessa a direção na qual a parede ou divisória se desenvolve, limitando-se esse valor a duas vezes o vão menor.

3 O deslocamento total deve ser obtido a partir da combinação das ações características ponderadas pelos coeficientes definidos na Seção 11.

4 Deslocamentos excessivos podem ser parcialmente compensados por contraflechas.

Fonte: NBR 6118:2014.

A norma estabelece que a combinação de ações a ser utilizada para essa verificação é a quase permanente e o módulo de elasticidade secante (E_{cs}). Ainda de acordo com a norma, a análise “deve ser realizada através de modelos que considerem a rigidez efetiva das seções do elemento estrutural, ou seja, que levem em consideração a presença da armadura, a existência de fissuras no concreto ao longo dessa armadura e as deformações diferidas no tempo.

Quando da existência de armadura ativa, a norma permite utilizar a rigidez da seção bruta desde que não seja ultrapassado o limite de formação de fissuras. Caso contrário, deve ser considerada a redução da inércia na seção fissurada além do efeito da protensão em seu cálculo.

2.7 VERIFICAÇÃO DE ABERTURA DE FISSURAS

A fissuração em elementos de concreto pode ocasionar problemas relacionados à sua durabilidade. Segundo Fusco (2008), “O aparecimento das fissuras obviamente traz para as armaduras um risco de agressão maior do que o naturalmente existente em virtude da porosidade do concreto da camada de cobrimento”. Portanto, seu controle é de grande importância para garantir a integridade do elemento estrutural, muito embora, de acordo com Carvalho e Figueiredo Filho (2009), outros fatores também podem vir a provocar as fissuras, como retração plástica ou térmica e expansão decorrente de reações químicas.

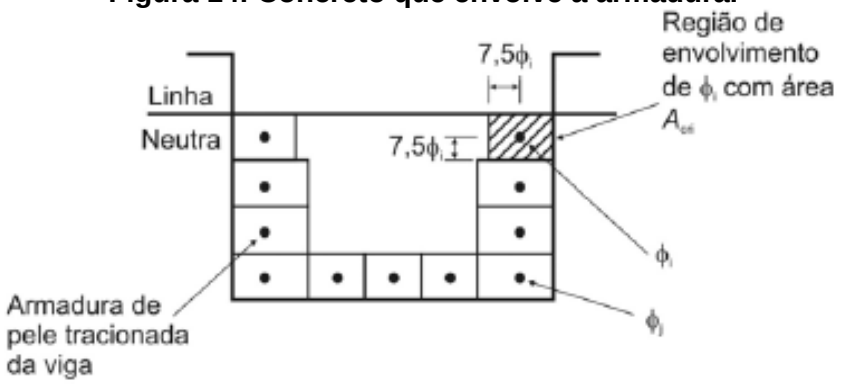
Nos elementos de concreto armado sem protensão, os limites máximos de abertura de fissura (w_k), para garantir sua durabilidade e segurança, variam conforme a classe de agressividade ambiental. Segundo a tabela 13.4 da NBR 6118:2014, esses valores, para a combinação frequente, correspondem a:

- $w_k \leq 0{,}4$ mm, para CAA I;
- $w_k \leq 0{,}3$ mm, para CAA II e III;
- $w_k \leq 0{,}2$ mm, para CAA IV;

Para o caso de armadura ativa aderente, essa análise deverá ser efetuada para quando da classe de agressividade I (ou seja, protensão parcial). Nesse caso, deve-se garantir que a abertura máxima de fissura não exceda 0,2 mm. Para as outras classes de agressividade pressupõe-se que não ocorra fissuração ao longo da viga, já que os critérios normativos não permitem que as tensões atuantes excedam os limites de formação de fissuras. Portanto, não há necessidade de verificá-las nesses casos.

No cálculo da abertura de fissuras, a NBR 6118:2014 diz que “cada elemento ou grupo de elementos das armaduras passiva e ativa aderente (excluindo-se os cabos protendidos que estejam dentro de bainhas), que controlam a fissuração do elemento estrutural, deve ser considerada uma área A_{cr} do concreto de envolvimento, constituída por um retângulo cujos lados não distem mais de 7,5 ϕ do eixo da barra da armadura”. A Figura 14 ilustra o que deve ser feito.

Figura 14: Concreto que envolve a armadura.



Fonte: NBR 6118:2014.

A norma também diz que é conveniente que a armadura de pele presente na zona tracionada da viga limite a abertura de fissura em sua região (A_{cr}), além de um espaçamento máximo de 15 cm.

Finalizando outras premissas apontadas pela norma, nos elementos estruturais protendidos, deve ser considerado o acréscimo de tensão no centro de gravidade da armadura entre os limites de descompressão e do carregamento atuante.

Esta deve, ainda, ser calculada no estádio II e considerando a relação entre os módulos de elasticidade entre o aço e o concreto (α_e) igual a 15.

2.8 DIMENSIONAMENTO DA ARMADURA TRANSVERSAL

O dimensionamento da armadura transversal se faz necessário para resistir aos esforços de cisalhamento, decorrentes do carregamento perpendicular aos eixos de um elemento. Dentre os tipos de ruptura que podem ocorrer num elemento tipo viga (flexão, flexo-cisalhamento, cisalhamento e escorregamento da armadura) a ruptura por cisalhamento ocorre de forma muito imprevisível, não apresentando sinais prévios da ocorrência do problema (como o que acontece com a deformação excessiva da armadura de flexão quando se encontra no domínio 2 ou 3). Por isso, deve-se evitá-la ao máximo.

Em Carvalho (2012, p. 397), diz-se que os efeitos da protensão em vigas protendidas deve ser levado em consideração para o dimensionamento da armadura transversal. De acordo com o autor “combater o cisalhamento passa também por evitar que a resistência da tração diagonal do concreto seja ultrapassada, ou seja, a tração que ocorre no concreto em uma direção inclinada em relação ao eixo longitudinal da peça”. Logo, o esforço de compressão reduz os efeitos da tensão de tração.

Para o caso da pós-tração, a curvatura do cabo, ocasionado pelo seu traçado não-retilíneo, proporciona uma componente vertical da força de protensão, agindo de forma a criar uma força cortante. Como, normalmente, esta força é contrária à cortante causada pelos carregamentos, ela age de forma a reduzir a tensão de cisalhamento.

O modelo de cálculo da armadura transversal em uma viga protendida é semelhante ao realizado para uma viga de concreto armado convencional. A analogia de treliça de Morsch é a base para o dimensionamento no ELU da armadura transversal.

De acordo com suas teorias, pode-se representar uma viga, quando próxima ao colapso, por uma treliça equivalente.

No entanto, deve-se ressaltar que essas hipóteses são feitas para facilitar o processo de cálculo. Na realidade, a treliça em questão é hiperestática (seus nós não devem ser consideradas uma articulação perfeita); nas regiões mais solicitadas pela cortante (próximo

aos apoios), a inclinação de fissuras e bielas é menor que 45°; uma parcela da força cortante é absorvida pelo concreto comprimido; os banzos não são paralelos entre si; bielas de concreto estão parcialmente engastados no banzo comprimido, criando uma situação de flexo-compressão que alivia as diagonais tracionadas; pelo fato de as bielas serem mais rígidas que as regiões tracionadas, estas “absorvem” uma parcela maior da cortante; e finalmente, a armadura longitudinal interfere no esforço dos estribos.

Como estes efeitos não são considerados, as expressões de cálculo levam a um dimensionamento exagerado da armadura transversal. Como soluções para isso são adotados modelos simplificados que considerem esses efeitos de forma a obter resultados menos conservadores e mais reais.

A NBR 6118:2014 estabelece dois modelos para o dimensionamento da armadura transversal. O modelo I, onde se admite que as bielas de compressão são inclinadas a 45° com o eixo longitudinal da viga, e o modelo II, onde as bielas possuem uma inclinação arbitrada pelo projetista entre 30° e 45°. A norma, no entanto, não deixa claro quando se deve utilizar um ou outro modelo. De acordo com Carvalho (2012), como nas vigas protendidas o efeito da protensão reduz a inclinação das bielas, o mais coerente seria utilizar o modelo II estimando uma inclinação.

Há de se destacar que as expressões descritas nos modelos I e II abaixo são apresentadas em função das forças cortantes. No entanto estas podem ser observadas sob a ótica de tensões de cisalhamento, bastando que o calculista divida as expressões pela área da seção transversal.

Modelo I:

Baseia-se na inclinação de 45° da biela comprimida. A verificação à ruptura da viga é garantida pelas expressões apresentadas no livro de Carvalho (2012) ou na NBR 6118:2014.

Modelo II:

Baseia-se na inclinação entre 30° e 45° da biela comprimida. Neste caso, assume-se que a parcela V_c sofra uma redução com um aumento de V_{Sd}. As expressões referentes ao cálculo da armadura pelo modelo II estão apresentadas em Carvalho (2012) e na NBR 6118:2014.

2.8.1 Quantidades mínimas e máximas de estribos

Existem casos em que a parcela resistente V_c é suficiente para resistir às forças de solicitação cortantes. No entanto, a norma estabelece que mesmo nesses casos deva existir uma armadura transversal ao longo da viga desde que $b_w > 5 * d$, caso em que este deve ser tratado como laje. Essa é definida pela NBR 6118:2014 como:

$$\rho_{sw\alpha,mín} = \frac{A_{sw}}{b_w * s * sen\alpha} = 0{,}2 * \frac{f_{ctm}}{f_{ywk}} \quad (Eq.\,2.8.1.1)$$

Quanto ao espaçamento máximo permitido, este deve atender a requisitos visando a utilização do vibrador. A norma estabelece o seguinte critério:

$$s_{máx} \leq \begin{cases} 0{,}6 * d \leq 300\ mm, \text{se } V_{sd} \leq 0{,}67 * V_{Rd2} \\ 0{,}3 * d \leq 200\ mm, \text{se } V_{sd} > 0{,}67 * V_{Rd2} \end{cases} \quad (Eq.\,2.8.1.2)$$

Em relação ao espaçamento mínimo, este deve garantir a passagem do vibrador garantindo, assim, um bom adensamento do concreto.

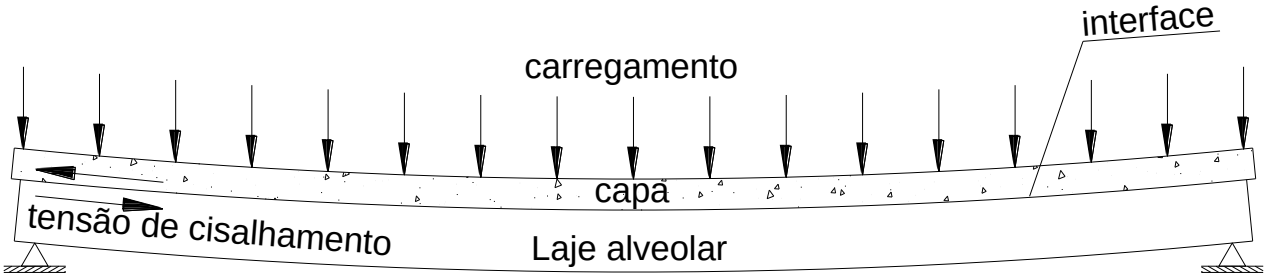
2.9 CONTROLE DO CISALHAMENTO NA INTERFACE DE UMA SEÇÃO COMPOSTA

Em elementos pré-moldados solidarizados com uma capa de concreto moldado in-loco, a existência de uma tensão de cisalhamento na interface entre o concreto pré-moldado e o concreto do capeamento deve ser estudada de forma a garantir o funcionamento dessa solidarização.

De acordo com Santos (2014), a utilização de concretos com características mecânicas diferentes faz com que seja necessária a consideração dos diferentes módulos de elasticidade em sua análise. Além disso, a idade diferente entre os concretos faz com que alguns fenômenos reológicos como a retração e a fluência provoquem efeitos desiguais entre eles. Mas, de acordo com o autor, por se tratar de uma diferença de idades pequenas, estes fenômenos podem ser desprezados.

A tensão de cisalhamento na interface surge quando um elemento sofre flexão e ocorre uma tendência de escorregamento da capa.

Figura 15: Tensão de cisalhamento na interface de um elemento fletido.



Fonte: Santos (2014) adaptado de Ibrahin (2008).

Essa transferência de cisalhamento é dividida em dois efeitos (Santos, 2014 apud El Debs, 2000):

- Por meio de superfície de contato, associado a:

1-) Adesão: Parcela que se limita a baixas tensões;

2-) Atrito: Ocorre após a adesão e depende da tensão normal na interface;

3-) Mecânica: Parcela associada às saliências da superfície.

- Por meio da armadura atuante na interface:

1-) Efeito de pino: Relacionado com a resistência ao corte da armadura;

2-) Tensão normal na interface: Ocorre pela tendência de deslocamento relativo entre os dois concretos.

Segundo o autor, dentre alguns fatores que interferem na resistência ao cisalhamento da interface, pode-se citar:

- Resistência do concreto;
- Rugosidade da superfície de contato;
- Armadura atuante na interface;
- Tensão normal na interface;
- Ações cíclicas.

Em relação à rugosidade, não há, de acordo com Santos (2014), um consenso em relação a sua classificação. Mas a mais utilizada atualmente é apresentada pela FIP:1982, onde:

- Nível 1 – superfície bastante lisa, obtida com o uso de fôrmas metálicas ou de madeiras;
- Nível 2 – superfície que foi alisada, atingindo níveis próximos ao Nível 1;
- Nível 3 – superfície que foi alisada, mas que apresenta pequenas ondulações;
- Nível 4 – superfície que foi executada com fôrmas deslizantes ou régua vibratória;
- Nível 5 – superfície que foi produzida por alguma forma de extrusão;
- Nível 6 – superfície que foi deliberadamente texturizada pelo escovamento do concreto ainda fresco;
- Nível 7 – como em 6, com maior pronunciamento da texturização;
- Nível 8 – superfície em que o concreto foi perfeitamente vibrado, sem a intenção de fazer superfície lisa, ou fazendo com que os agregados graúdos fiquem expostos;
- Nível 9 – superfície em que o concreto ainda fresco foi jateado com água ou areia para expor os agregados graúdos;
- Nível 10 – superfície propositadamente rugosa.

Resumindo esta classificação, a FIP:1982 adota uma classificação em três casos básicos:

- Superfície lisa: Níveis 1 e 2;
- Superfície naturalmente rugosa: Níveis 3 a 6;
- Superfície intencionalmente rugosa: Níveis 7 a 10.

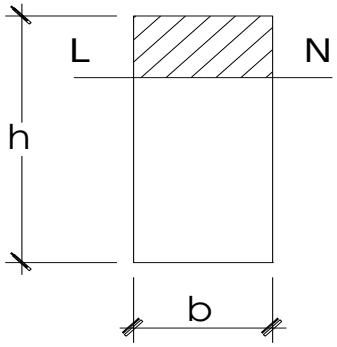
2.9.1 Cálculo da tensão de cisalhamento na interface

Em Santos (2014) apresentam-se duas formas de calcular a tensão de cisalhamento atuante na interface.

- Cálculo da tensão de referência:

Por este cálculo, o cálculo da tensão de referência corresponde às tensões de cisalhamento médio para vigas de seção transversal usual.

Figura 16: Seção transversal de uma viga usual.



Fonte: Santos, 2014 apud Beer, Johnston, 1989.

$$\tau_s = \frac{Q.M_e}{I.b} \quad (Eq.2.9.1.1)$$

Onde,

τ_s: tensão de cisalhamento na altura da linha neutra;

Q: força cortante atuante;

M_e: momento estático da região comprimida em relação à linha neutra;

I: momento de inércia da seção transversal;

b: largura da seção.

É necessário, ainda, considerar os efeitos consequentes ao estádio de funcionamento do concreto.

De acordo com o autor, para a situação não fissurada pode-se adotar a equação acima apresentada. Para a situação fissurada, no entanto, esta pode ser considerada quando da linha neutra passando dentro ou abaixo da capa.

No primeiro caso, segue-se a seguinte equação:

$$\tau_{ref} = \frac{Q}{z.b_{int}} \quad (Eq.2.9.1.2)$$

Onde,

z: braço de alavanca entre as resultantes de compressão e tração.

b_{int}: menor largura da seção na interface.

No segundo caso:

$$\tau_{ref} = \frac{Q}{z.b_{int}}\left(\frac{R_{c,loc}}{R_c}\right) \quad (Eq.2.9.1.3)$$

Onde,

$R_{c,loc}$: resultante de compressão no concreto da capa;

R_c: resultante de compressão acima da linha neutra.

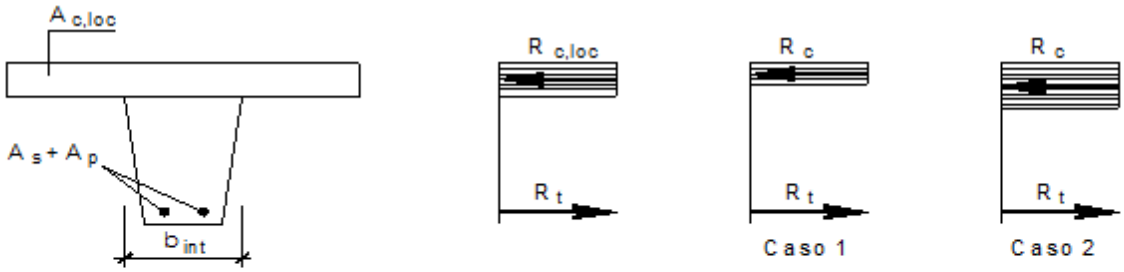
- Cálculo da tensão média:

Este caso, baseado no manual do PCI:2010, é calculado baseado na força horizontal solicitante de cálculo (F_{hd}) atuante na interface, em sua largura (b_{int}) e o comprimento relativo ao cisalhamento na interface (l_0).

$$\tau_{med} = \frac{F_{hd}}{b_{int}.l_0} \quad (Eq.2.9.1.4)$$

Para o cálculo da força horizontal, esta pode ser verificada na Figura 17.

Figura 17: Cálculo da força horizontal de cisalhamento para seção com momento fletor positivo.



$A_{c,loc}$ = área da seção transversal correspondente ao concreto moldado no local;

$R_{c,loc}$ = valor de referência da resultante de compressão na parte do concreto moldado no local, ou seja, $0,85.f_{cd}.A_{c,loc}$;

R_c = resultante de compressão;

R_t = resultante de tração (devido a $A_s + A_p$);

F_{hd} = força horizontal de cisalhamento.

Caso 1	Caso 2
$R_t = R_c < R_{c,loc}$	$R_t = R_c > R_{c,loc}$
$F_{hd} = R_c$	$F_{hd} = R_{c,loc}$

Fonte: Santos, 2014 apud PCI:2010.

O caso 1 considera a linha neutra passando acima da interface enquanto que o caso 2 a considera abaixo.

3 ESTUDOS SOBRE PRÉ-TRAÇÃO

3.1 SITUAÇÕES LIMITES DE PROTENSÃO

Ao longo do desenvolvimento do programa, constatou-se a necessidade de se estudar as situações limites de protensão para as vigas protendidas sob determinadas condições de cálculo. Existem situações onde a utilização da protensão não é possível. Geralmente, este tipo de comportamento pode ser observado em vigas delgadas com um carregamento acidental elevado. Nesses casos, a solicitação no tempo infinito é muito elevada, necessitando de bastante armadura para resistir a esses carregamentos. No entanto, quando realizada a verificação de ruptura em vazio, esta não resiste aos elevados efeitos da protensão.

Dentre as exigências normativas para o dimensionamento de um elemento protendido, uma etapa que demanda relativa parte do tempo são as verificações de tensões que devem ser realizadas na seção em análise a fim de se certificar que a viga está dentro das condições de serviço e utilização. A efetuação dos cálculos é facilmente otimizado pelo uso de planilhas ou programas, estes que, quando corretamente desenvolvidos, tornam-se ferramentas extremamente eficazes nas verificações, tão eficientes que sua análise pode facilmente ser feita por profissionais com pouca experiência no assunto.

É importante destacar, porém, que a filosofia de dimensionamento de uma viga protendida é diferente da viga de concreto armado convencional. Enquanto que para esta o fator mais importante é a quantidade mínima de armadura na seção, nas vigas protendidas esse fator pode não ser o mais determinante, variando de empresa para empresa.

Um mesmo elemento protendido pode apresentar armadura passiva na borda superior para controle da fissuração, armadura ativa na borda superior para eliminar essa fissuração ou até mesmo isolar cabos na armadura ativa inferior para reduzir a tensão na borda oposta. Se uma empresa não possui equipamentos para realizar protensão na borda superior, ela pode optar por outras soluções que resultem em um desempenho estrutural semelhante. Em outro exemplo, uma empresa pode optar, por algum motivo específico, substituir parte da armadura ativa por passiva sem perda de capacidade resistente. Há ainda uma situação particular em que a viga não apresenta solução com armadura ativa, ou seja, em vazio as tensões excedem os limites normativos e no tempo infinito elas não são suficientes para garantir os estados limites de serviço.

O que se quer dizer, afinal, é que os elementos protendidos apresentam uma gama tão variável de soluções que para se chegar a uma conclusão sobre qual a melhor delas é preciso analisar fatores intrínsecos às empresas.

Isso mostra, portanto, como é importante que o engenheiro projetista tenha conhecimento sobre todas as soluções que uma viga, com determinada seção e carregamento, apresentam.

Uma forma bastante simples em se analisar quais são as soluções cabíveis é partindo-se das equações de tensão:

- Na borda inferior:

$$\sigma_i = \frac{A_p * \sigma_p}{A_c} + \frac{A_p * \sigma_p * e_p}{W_i} - \frac{M_g}{W_i} - \frac{\Psi * M_q}{W_{i,t=\infty}} \quad (Eq. 3.1.1)$$

- Na borda superior:

$$\sigma_s = \frac{A_p * \sigma_p}{A_c} - \frac{A_p * \sigma_p * e_p}{W_s} + \frac{M_g}{W_s} + \frac{\Psi * M_q}{W_{s,t=\infty}} \quad (Eq. 3.1.2)$$

Isolando-se a variável de armadura ativa (A_p) e assumindo os valores limites para as tensões, é possível definir um intervalo de valores que é solução para a inequação. Fazendo isso para cada verificação (em vazio e no tempo infinito) se cria um “varal” de soluções cuja intersecção, quando houver, resolve o problema e transmite uma série de informações ao projetista.

As equações de equilíbrio que foram definidas a partir do processo descrito acima são apresentados abaixo aplicando-se cada um dos limites normativos de tensão. Para mais detalhes, pesquisar no artigo “Situações limites para vigas pré-fabricadas” (Trevizoli, Carvalho, Cass e Silva, 2014).

a-) Verificação de compressão excessiva no tempo zero:

Borda inferior	Borda superior
$A_p \leq \frac{0{,}7 * f_{ck,j} + \frac{M_{g1}}{W_i}}{\frac{\sigma_p}{A_c} + \frac{\sigma_p * e_p}{W_i}} \quad (Eq. 3.1.3)$	$A_p \leq \frac{0{,}7 * f_{ck,j} - \frac{M_{g1}}{W_s}}{\frac{\sigma_p}{A_c} - \frac{\sigma_p * e_p}{W_s}} \quad (Eq. 3.1.4)$

Onde o número 0,7 que multiplica o $f_{ck,j}$ é o limite estabelecido pela norma para compressão excessiva.

b-) Verificação de limite de tração no tempo zero:

Borda inferior	Borda superior
$A_p \geq \frac{-1{,}2 * f_{ct,m} + \frac{M_{g1}}{W_i}}{\frac{\sigma_p}{A_c} + \frac{\sigma_p * e_p}{W_i}} \quad (Eq.3.1.5)$	$A_p \geq \frac{-1{,}2 * f_{ct,m} - \frac{M_{g1}}{W_s}}{\frac{\sigma_p}{A_c} - \frac{\sigma_p * e_p}{W_s}} \quad (Eq.3.1.6)$

Onde o número -1,2 que multiplica o $f_{ct,m}$ é o limite estabelecido pela norma para limitar a tração na seção. O valor negativo indica tensão de tração.

c-) Verificação de tração no tempo zero:

Borda inferior	Borda superior
$A_p \geq \frac{\frac{M_{g1}}{W_i}}{\frac{\sigma_p}{A_c} + \frac{\sigma_p * e_p}{W_i}} \quad (Eq.3.1.7)$	$A_p \geq \frac{\frac{M_{g1}}{W_s}}{\frac{\sigma_p}{A_c} - \frac{\sigma_p * e_p}{W_s}} \quad (Eq.3.1.8)$

A tensão, nessa verificação, não permite o surgimento de tração, logo, equivale a zero.

d-) Verificação de compressão excessiva no tempo infinito considerando a acidental máxima e mínima (onde a acidental mínima possui valor nulo) e para as combinações frequente, quase permanente e rara:

Borda inferior	Borda superior
$A_p \leq \frac{0{,}7 * f_{ck} + \frac{M_{g,t}}{W_i} + \frac{\gamma_{f2} * M_q}{W_{i,t=\infty}}}{\frac{\sigma_p}{A_c} + \frac{\sigma_p * e_p}{W_i}} \quad (Eq.3.1.9)$	$A_p \leq \frac{0{,}7 * f_{ck} - \frac{M_{g,t}}{W_s} - \frac{\gamma_{f2} * M_q}{W_{s,t=\infty}}}{\frac{\sigma_p}{A_c} - \frac{\sigma_p * e_p}{W_s}} \quad (Eq.3.1.10)$

Onde o número 0,7 que multiplica o f_{ck} é o limite estabelecido pela norma para compressão excessiva.

e-) Verificação do estado limite de formação de fissuras (ELS-F) considerando a acidental máxima e mínima (onde a acidental mínima possui valor nulo) e para as combinações frequente e rara (protensão limitada e completa, respectivamente):

Borda inferior	Borda superior
$A_p \geq \frac{-f_{ct,m} + \frac{M_{g,t}}{W_i} + \frac{\gamma_{f2} * M_q}{W_{i,t=\infty}}}{\frac{\sigma_p}{A_c} + \frac{\sigma_p * e_p}{W_i}} \quad (Eq. 3.1.11)$	$A_p \geq \frac{-f_{ct,m} - \frac{M_{g,t}}{W_s} - \frac{\gamma_{f2} * M_q}{W_{s,t=\infty}}}{\frac{\sigma_p}{A_c} - \frac{\sigma_p * e_p}{W_s}} \quad (Eq. 3.1.12)$

Onde o número -1,0 que multiplica o $f_{ct,m}$ é o limite estabelecido pela norma para limitar a tração na seção. O valor negativo indica tensão de tração.

f-) Verificação do estado limite de descompressão (ELS-D) considerando a acidental máxima e mínima (onde a acidental mínima possui valor nulo) e para as combinações quase permanente e frequente (protensão limitada e completa, respectivamente):

Borda inferior	Borda superior
$A_p \geq \frac{\frac{M_{g,t}}{W_i} + \frac{\gamma_{f2} * M_q}{W_{i,t=\infty}}}{\frac{\sigma_p}{A_c} + \frac{\sigma_p * e_p}{W_i}} \quad (Eq. 3.1.13)$	$A_p \geq \frac{\frac{M_{g,t}}{W_s} - \frac{\gamma_{f2} * M_q}{W_{s,t=\infty}}}{\frac{\sigma_p}{A_c} - \frac{\sigma_p * e_p}{W_s}} \quad (Eq. 3.1.14)$

Na descompressão, não se permite o aparecimento de tração na seção. Logo, seu valor limite é zero.

Onde:

A_p – Área de armadura ativa;

$f_{ck,j}$ / f_{ck} / $f_{ct,m}$ – Resistência à compressão do concreto na idade de "j" dias e de 28 dias e resistência à tração média do concreto, respectivamente;

M_{g1} / $M_{g,t}$ / M_q: Momento fletor de peso próprio, permanente total e acidental, respectivamente;

W_i / W_s – Módulo de resistência à flexão da seção inferior e superior, respectivamente;

$W_{i,t=\infty}$ / $W_{s,t=\infty}$ – Módulo de resistência à flexão da seção inferior e superior final, respectivamente;

σ_p: Tensão de protensão atuante nas cordoalhas;

e_p: Excentricidade das cordoalhas inferiores;

A_c: Área da seção transversal bruta do concreto da peça pré-fabricada;

γ_{f2}: Coeficiente de ponderação das ações acidentais.

Foram feitas algumas simulações com vigas de diferentes seções e carregamentos e foi possível identificar cinco situações particulares:

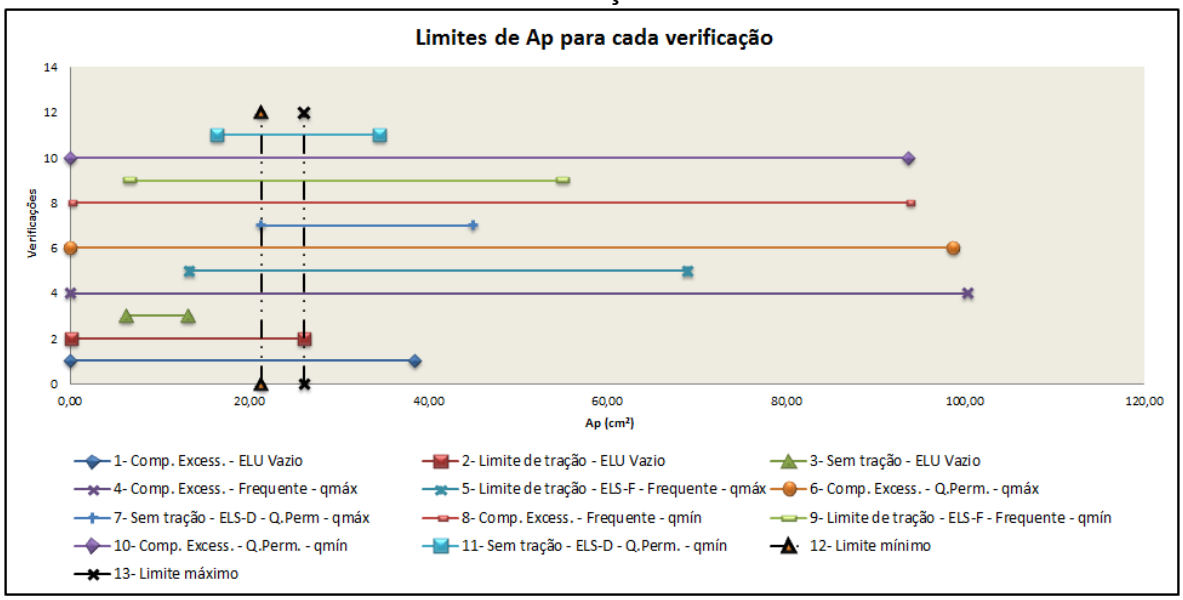
Aqui se faz uma breve descrição sobre como devem ser analisadas a

Figura 18, Figura 19 e a Figura 20. O eixo das ordenadas representa o número da verificação cuja legenda está na parte inferior da figura. O eixo das abcissas representa a

área de armadura ativa (A_p), em cm². As linhas horizontais são os intervalos solução para cada verificação. As linhas verticais tracejadas delimitam o intervalo que é solução para todas as verificações.

Alguns intervalos apresentavam valores negativos de A_p. Como isso não tem significado técnico, foram ignorados.

Figura 18: Intervalo com os limites mínimo e máximo de armadura ativa relativa às verificações.



Fonte: Adaptado de Trevizoli, Carvalho, Cass e Silva, 2014.

Observações da

Figura 18: Como se pode ver, esta viga apresenta solução pois existe um intervalo de valores de A_p que satisfaz as condições normativas. Observa-se, também, que esse intervalo não apresenta valores que satisfaçam a verificação 3 (situação sem tração na seção em vazio), o que não chega a ser determinante. Isso significa apenas que será necessário utilizar armadura passiva ou ativa para controle da tração.

Neste cenário, pode-se imaginar 3 situações distintas que está relacionado com a quantidade de A_p para satisfazer o ELU:

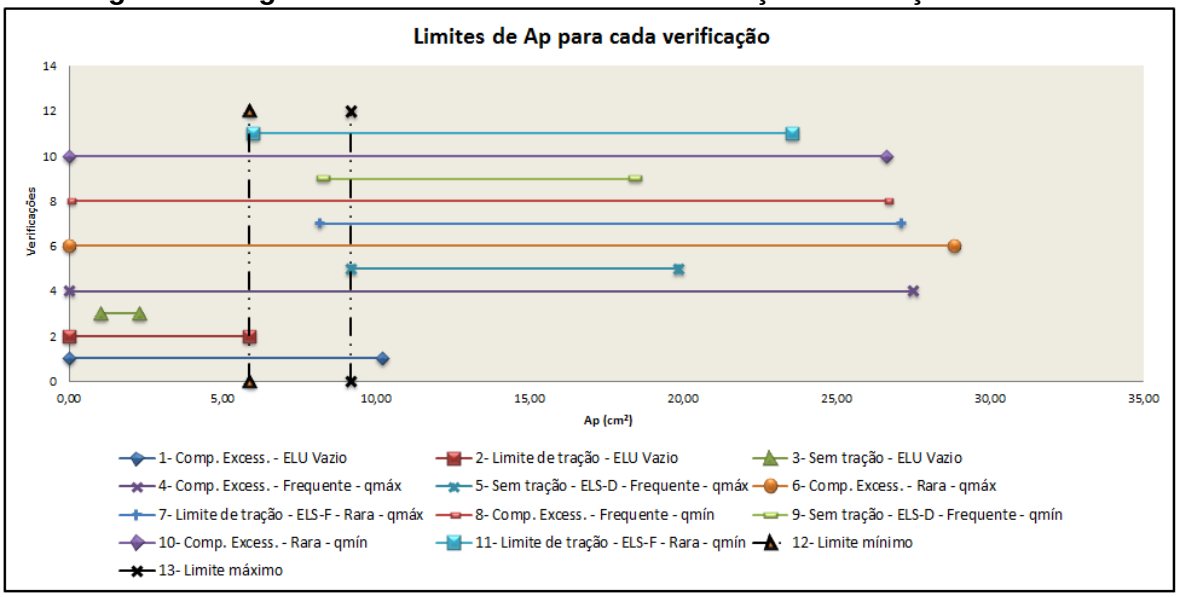
1-) Imaginando-se, primeiramente, que o A_p no ELU esteja fora do intervalo e à esquerda dele. Isso significa apenas que a quantidade de armadura no ELU não é o suficiente para satisfazer as exigências em serviço. Logo, é necessário aumentar a área de A_p.

2-) Agora, o A_p passa dentro do intervalo. Conclui-se rapidamente que não é necessário realizar alterações no número de cordoalhas.

3-) Por último, o A_p no ELU passa fora do intervalo e à direita deste. Neste caso, a armadura excedeu os limites normativos, levando a uma situação conflitante pois a redução de A_p implica em problemas de estabilidade na ruptura. Esse é um caso em que deve-se utilizar

armadura mista, ou seja, deve-se reduzir a armadura ativa até valores de utilização e completar a diferença para satisfazer os critérios de estabilidade.

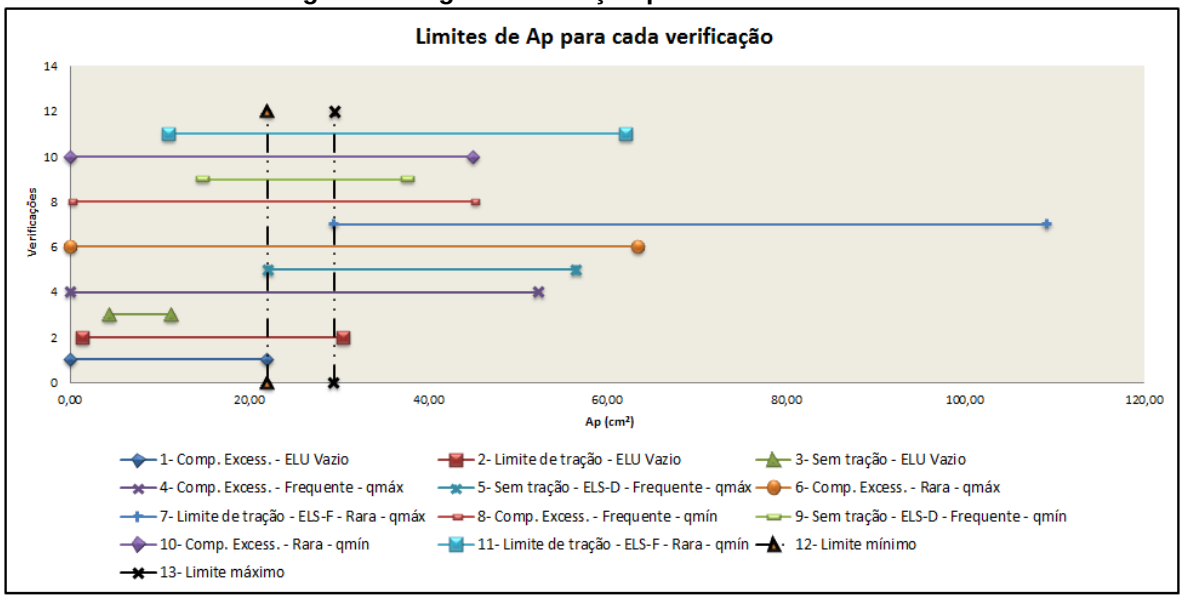
Figura 19: Viga sem intervalo comum às verificações de tração em vazio.



Fonte: Adaptado de Trevizoli, Carvalho, Cass e Silva, 2014.

A Figura 19 demonstra outra situação particular. Como se pode observar, além da verificação 3, que pode ser desconsiderada, a verificação 2 (limite de tração em vazio) não está sendo atendida. Isso significa que há tração excessiva na borda superior. Este é um caso onde, frente às condições do sistema, não há como utilizar armadura ativa apenas na borda inferior. Seria necessário prever armadura ativa na borda superior.

Figura 20: Viga sem solução para armadura ativa.



Fonte: Adaptado de Trevizoli, Carvalho, Cass e Silva, 2014.

A Figura 20 representa o quinto e último caso particular identificado. Neste, no intervalo de soluções possíveis (que está delimitado entre as verificações 7 e 2, respectivamente, o mínimo e o máximo) nenhum passa pela verificação 1 de compressão excessiva em vazio. Esta é uma situação em que a viga não apresenta solução com armadura ativa. Diminuir A_p deixaria de satisfazer as condições em serviço. Posicionar A_p na borda superior agravaria os problemas de compressão excessiva. Logo, esta é uma situação em que deve-se tratar a viga como de concreto armado convencional. Lembrando que isso vale para as condições de carregamento, geométricas e físicas estabelecidas.

Uma alternativa seria analisar a variação da excentricidade dos cabos. Mas essa variável não foi considerada nesse estudo.

O estudo dos limites de protensão mostra-se bastante relevante para a tomada de decisões. Serve de guia para que o engenheiro de projetos se oriente durante o dimensionamento de peças protendidas, permite a visualização de outras soluções para um mesmo elemento apresentando, assim, alternativas técnicas para a viga e, finalmente, define quando uma peça apresenta restrições quanto ao uso de armadura ativa.

3.2 COMPARAÇÕES NORMATIVAS ENTRE NBR 6118:2014; ACI 318M-05 (NORMA AMERICANA); NP EN 1992-1-1 (EUROCODE)

Parte importante do trabalho de pesquisa da engenharia em países em desenvolvimento está no acompanhamento das novas descobertas que são feitas por países desenvolvidos. Primeiro porque estes realizam pesquisas e trabalhos na área há muito mais tempo e, segundo, por apresentarem mais recursos destinados ao desenvolvimento técnico-científico.

Durante o desenvolvimento do programa, percebeu-se a importância de se verificar as considerações teóricas realizadas por países considerados de ponta na área de tecnologia. Nesse estudo foi possível observar algumas variações teóricas que são feitas na Europa e nos Estados Unidos como, por exemplo, a simplificação do diagrama parábola-retângulo e coeficientes de ponderação de ações que são utilizados por eles. Dessa forma, torna-se possível criar um programa que permita a customização normativa, onde o usuário pode adaptar coeficientes às normas estrangeiras e realizar diversos tipos de comparação. Para mais detalhes deste estudo, verificar o artigo “Estudo comparativo do cálculo de vigas pré-tracionadas segundo as normas brasileira, europeia e americana” (Trevizoli, Carvalho, Silva e Cass, 2014).

A tecnologia da protensão e dos pré-fabricados, em geral, vem sendo utilizados por países desenvolvidos há mais tempo que os países em desenvolvimento, cuja principal forma de construção, ainda nos dias de hoje, é a forma artesanal com a utilização de formas, escoramentos e concreto moldado no local.

Dessa forma, com o avanço de novas tecnologias e a necessidade de se aumentar a produtividade na execução, o sistema pré-fabricado vem ganhando cada vez mais espaço no Brasil. Logo, é conveniente que, além da pesquisa nacional, os pesquisadores desse tema tenham em mente a importância de se acompanhar as descobertas que vem sendo feitas nos países desenvolvidos.

Uma forma de se acompanhar o progresso dos países desenvolvidos é por meio da comparação das normas técnicas vigentes nesses países. Neste capítulo foram feitas algumas comparações entre a norma americana (ACI) e a europeia (EC) com a brasileira (NBR) a respeito dos critérios normativos para o cálculo e verificação de vigas pré-tracionadas. Foram abordados critérios como limites de tensão, coeficientes majoradores e redutores, simplificação do diagrama parábola-retângulo e combinação de ações.

3.2.1 NBR

Neste tópico serão abordados os critérios estabelecidos pela NBR. Em relação ao estado limite último, a norma estabelece que a ruína da peça ocorre com um alongamento de 1% do aço da armadura ou pelo encurtamento de 0,35% do concreto. A norma recomenda o dimensionamento no estádio III como forma de evitar o colapso de uma forma econômica.

Em relação ao diagrama parábola-retângulo, a norma permite uma simplificação deste diagrama aproximando-o de um retângulo com altura $\lambda.x$ e largura $\alpha_c.f_{cd}$, onde λ e αc equivalem a:

Tabela 3: Variação dos coeficientes simplificadores do diagrama parábola-retângulo em função do concreto.

λ	α_c	Exigência
0,8	0,85	$f_{ck} < 50$ MPa
$0,8 – (f_{ck} – 50) / 400$	$0,85 * [1,0 – (f_{ck} – 50) / 200]$	$f_{ck} > 50$ MPa

Fonte: NBR 6118:2014.

Em relação aos coeficientes redutores da resistência característica, a norma recomenda utilizar 1,4 para o concreto e 1,15 para o aço. Quanto aos coeficientes

majoradores, a norma apresenta várias tabelas que especificam o tipo de ação e a combinação analisada. Em via de regra, utiliza-se coeficientes de 1,3 à 1,35 para o carregamento permanente de elementos pré-moldados e 1,5 para o carregamento acidental (muito embora esses valores não sejam absolutos, ou seja, variam conforme as exigências).

Em relação às verificações de tensões na seção, a NBR recomenda que seja feita na situação em vazio e após as perdas de protensão, no tempo infinito. Em vazio, deve-se certificar que as tensões de compressão não excedam 0,7 da resistência do concreto no dia da aplicação da protensão ($0,7*f_{ck,j}$) e nem 1,2 de sua resistência à tração ($1,2*f_{ctm,j}$). Caso haja tração e essa não exceda o limite, a norma recomenda utilizar uma armadura passiva para controle da fissura.

No tempo infinito devem-se atender os requisitos previstos na tabela 13.3 da NBR 6118:2014, que depende da classe de agressividade ambiental.

Tabela 4: Classes de agressividade ambiental – CAA.

Classe de agressividade ambiental	Agressividade	Classificação geral do tipo de ambiente para efeito de projeto	Risco de deterioração da estrutura
I	Fraca	Rural	Insignificante
		Submersa	
II	Moderada	Urbana[1), 2)]	Pequeno
III	Forte	Marinha[1)]	Grande
		Industrial[1), 2)]	
IV	Muito forte	Industrial [1), 3)]	Elevado
		Respingos de maré	

[1)] Pode-se admitir um microclima com uma classe de agressividade mais branda (um nível acima) para ambientes internos secos (salas, dormitórios, banheiros, cozinhas e áreas de serviço de apartamentos residenciais e conjuntos comerciais ou ambientes com concreto revestido com argamassa e pintura).

[2)] Pode-se admitir uma classe de agressividade mais branda (um nível acima) em: obras em regiões de clima seco, com umidade relativa do ar menor ou igual a 65%, partes da estrutura protegidas de chuva em ambientes predominantemente secos, ou regiões onde chove raramente.

[3)] Ambientes quimicamente agressivos, tanques industriais, galvanoplastia, branqueamento em indústrias de celulose e papel, armazéns de fertilizantes, indústrias químicas.

Fonte: NBR 6118:2014.

Em função do tipo de ambiente e do tipo de protensão (pré ou pós), a norma divide as exigências e as combinações a serem utilizadas.

Tabela 5: Exigências de durabilidade relacionadas à fissuração e à proteção da armadura, em função das classes de agressividade ambiental.

Tipo de concreto estrutural	Classe de agressividade ambiental (CAA) e tipo de protensão	Exigências relativas à fissuração	Combinação de ações em serviço a utilizar
Concreto simples	CAA I a CAA IV	Não há	--
Concreto armado	CAA I	ELS-W $w_k \leq 0,4$ mm	Combinação frequente
	CAA II e CAA III	ELS-W $w_k \leq 0,3$ mm	
	CAA IV	ELS-W $w_k \leq 0,2$ mm	
Concreto protendido nível 1 (protensão parcial)	Pré-tração com CAA I ou Pós-tração com CAA I e II	ELS-W $w_k \leq 0,2$ mm	Combinação frequente
Concreto protendido nível 2 (protensão limitada)	Pré-tração com CAA II ou Pós-tração com CAA III e IV	Verificar as duas condições abaixo	
		ELS-F	Combinação frequente
		ELS-D[1)]	Combinação quase permanente
Concreto protendido nível 3 (protensão completa)	Pré-tração com CAA III e IV	Verificar as duas condições abaixo	
		ELS-F	Combinação rara
		ELS-D[1)]	Combinação frequente

[1)] A critério do projetista, o ELS-D pode ser substituído pelo ELS-DP com a_p = 25 mm (figura 3.1).

NOTAS

1 As definições de ELS-W, ELS-F e ELS-D encontram-se em 3.2.

2 Para as classes de agressividade ambiental CAA-III e IV exige-se que as cordoalhas não aderentes tenham proteção especial na região de suas ancoragens.

3 No projeto de lajes lisas e cogumelo protendidas basta ser atendido o ELS-F para a combinação frequente das ações, em todas as classes de agressividade ambiental.

Fonte: NBR 6118:2014.

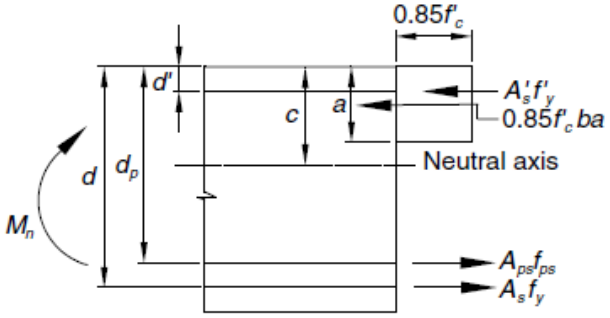
Logo, estas são algumas das exigências que a NBR 6118 estabelece para o dimensionamento da armadura de flexão de elementos pré-tracionados.

3.2.2 ACI

A norma americana apresenta critérios um pouco diferentes. No que diz respeito ao ELU, a norma parte do mesmo princípio de equilíbrio entre as forças de tração na barra e de compressão no concreto que é feito em outras normas

.

Figura 21: Momento resistente de uma seção genérica de uma viga de concreto armado.



Fonte: PCI 7th edição.

A simplificação do diagrama parábola-retângulo em um retângulo apresenta algumas diferenças. Primeiro que a largura do retângulo equivalente é de 0,85 do f_{ck} ($f'_c = f_{ck}$). Em relação à altura do retângulo (a), esse valor é obtido multiplicando a linha neutra (c) por um coeficiente β_1: $a = c * \beta_1$.

Onde o valor de β_1 varia conforme o tipo de concreto:

Tabela 6: Tabela de coeficientes β1.

f'_c, psi	β_1
3000	0.85
4000	0.85
5000	0.80
6000	0.75
7000	0.70
8000 and higher	0.65

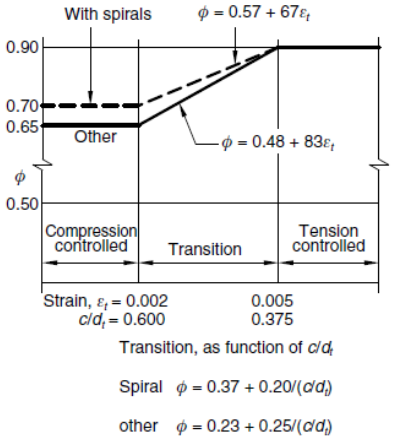
Fonte: PCI 7th edição.

Em relação aos coeficientes majoradores de carga, o ACI estabelece 1,2 para cargas permanentes e 1,6 para carregamento acidental.

O momento resistente calculado (M_n) deve ser multiplicado por um coeficiente que depende do comportamento da seção na ruptura.

- Na seção controlada por tração a deformação específica da armadura na ruptura é maior ou igual a 0,5% e apresenta ruptura dúctil;
- Na seção controlada pela compressão a deformação específica da armadura é inferior a 0,2% e apresenta ruptura frágil;
- Na seção de transição a deformação específica da armadura está entre 0,2% e 0,5%.

Figura 22: Variação do coeficiente φ pelo tipo de seção.



Fonte: PCI 7th edição.

O coeficiente φ que multiplica o momento resistente M_n é definido pela Figura 22.

Dessa forma, o momento de cálculo (M_u) deve apresentar o seguinte valor:

$$\varphi M_n = M_u \quad (Eq.3.2.2.1)$$

Em relação às verificações de tensões, a norma americana estabelece, para a situação em vazio e em serviço os valores limites de 0,6 da resistência à compressão na idade correspondente.

Para a tração, os valores são:

- Na extremidade da viga a tração não deve exceder 0,5 * $\sqrt{f_{ck,j}}$;
- Nas outras regiões da viga, o limite é de 0,25 * $\sqrt{f_{ck,j}}$.

Em relação aos critérios de durabilidade, a ACI estabelece três tipos de classe de utilização:

- Classe U (Uncracked): Seção não fissurada;
- Classe T (Transition): Seção entre a fissurada e não fissurada;
- Classe C (Cracked): Seção fissurada.

Estas que devem atender aos limites estabelecidos na tabela R18.3.3 do ACI 318M-05.

Tabela 7: Critérios de verificação em serviço.

	Prestressed			Nonprestressed
	Class U	Class T	Class C	
Assumed behavior	Uncracked	Transition between uncracked and cracked	Cracked	Cracked
Section properties for stress calculation at service loads	Gross section 18.3.4	Gross section 18.3.4	Cracked section 18.3.4	No requirement
Allowable stress at transfer	18.4.1	18.4.1	18.4.1	No requirement
Allowable compressive stress based on uncracked section properties	18.4.2	18.4.2	No requirement	No requirement
Tensile stress at service loads 18.3.3	$\leq 0.62\sqrt{f'_c}$	$0.62\sqrt{f'_c} < f_t \leq 1.0\sqrt{f'_c}$	No requirement	No requirement
Deflection calculation basis	9.5.4.1 Gross section	9.5.4.2 Cracked section, bilinear	9.5.4.2 Cracked section, bilinear	9.5.2, 9.5.3 Effective moment of inertia
Crack control	No requirement	No requirement	10.6.4 Modified by 18.4.4.1	10.6.4
Computation of Δf_{ps} or f_s for crack control	—	—	Cracked section analysis	$M/(A_s \times$ lever arm), or $0.6f_y$
Side skin reinforcement	No requirement	No requirement	10.6.7	10.6.7

Fonte: ACI 318M-05.

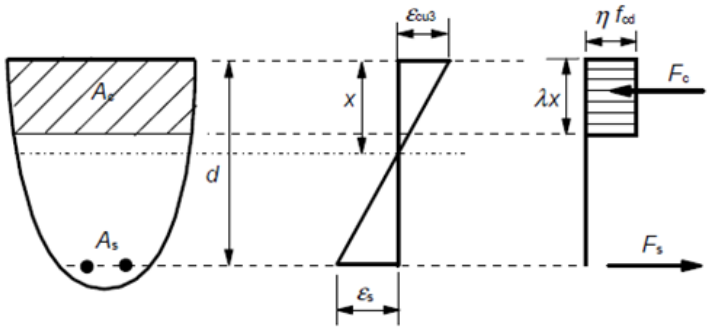
A norma também estabelece que a tensão de compressão decorrente da carga permanente não exceda 45% da resistência à compressão do concreto.

Há que ressaltar que apesar da norma estabelecer os limites de 0,6 do $f_{ck,j}$, o manual do PCI (Precast/Prestressed Concrete Institute) recomenda o valor de 0,7, ou seja, utilizando um valor menos conservador.

3.2.3 Eurocode

Em relação ao dimensionamento no ELU, as considerações do Eurocode são parecidas com a brasileira. Quanto ao diagrama parábola-retângulo, adotam-se valores parecidos para os coeficientes.

Figura 23: Distribuição retangular de tensões.



Fonte: NP EN 1992-1-1.

Onde os valores de λ e η são os apresentados na Tabela 8:

.

Tabela 8: Variação dos coeficientes simplificadores do diagrama parábola-retângulo em função do concreto.

λ	η	Exigência
0,8	1,0	f_{ck} < 50 MPa
0,8 – (f_{ck} – 50) / 400	1,0 – (f_{ck} – 50) / 200	50 MPa < f_{ck} < 90 MPa

Fonte: NP EN 1992-1-1.

Em relação aos majoradores de carga, os valores destes coeficientes são de 1,35 para carga permanente e 1,5 para a acidental.

Para os redutores de resistência, adotam-se os valores de:

Tabela 9: Coeficientes parciais relativo aos materiais para o estado limite último.

Situações de projecto	*γ_c para betão*	*γ_s para aço de armaduras para betão armado*	*γ_s para aço de armaduras de pré-esforço*
Persistentes Transitórias	*1,5*	*1,15*	*1,15*
Acidentais	*1,2*	*1,0*	*1,0*

Fonte: NP EN 1992-1-1.

Em relação às verificações de tensão, o Eurocode estabelece um limite de compressão de 0,6 do $f_{ck,j}$. No caso da tração, a norma estabelece apenas que deve ser

previsto o uso de armadura para o controle dessa tensão caso os esforços superem o valor de resistência à tração do concreto.

Para a situação em serviço, as exigências variam conforme a classe de exposição do elemento, que são estabelecidos pela Tabela 9.

Tabela 10: Classes de exposição em função das condições ambientais.

Designação da classe	Descrição do ambiente	Exemplos informativos de condições em que poderão ocorrer as classes de exposição
1 Nenhum risco de corrosão ou ataque		
X0	Para betão sem armadura ou elementos metálicos embebidos: todas as exposições excepto em situação de gelo/degelo, abrasão ou ataque químico Para betão com armadura ou elementos metálicos embebidos: muito seco	Betão no interior de edifícios com uma humidade do ar ambiente muito baixa
2 Corrosão induzida por carbonatação		
XC1	Seco ou permanentemente húmido	Betão no interior de edifícios com uma humidade do ar ambiente baixa Betão permanentemente submerso em água
XC2	Húmido, raramente seco	Superfícies de betão sujeitas a contacto prolongado com água Um grande número de fundações
XC3	Humidade moderada	Betão no interior de edifícios com uma humidade do ar ambiente moderada ou elevada Betão exterior protegido da chuva
XC4	Alternadamente húmido e seco	Superfícies de betão sujeitas a contacto com água, não incluídas na classe de exposição XC2
3 Corrosão induzida por cloretos		
XD1	Humidade moderada	Superfícies de betão expostas a cloretos transportados pelo ar
XD2	Húmido, raramente seco	Piscinas Elementos de betão expostos a águas industriais contendo cloretos
XD3	Alternadamente húmido e seco	Elementos de pontes expostos a pulverizações contendo cloretos Pavimentos Lajes de parques de estacionamento
4 Corrosão induzida por cloretos presentes na água do mar		
XS1	Exposto ao sal transportado pelo ar mas não em contacto directo com a água do mar	Estruturas próximas da costa ou na costa
XS2	Permanentemente submerso	Elementos de estruturas marítimas
XS3	Zonas sujeitas aos efeitos das marés, da rebentação e da neblina marítima	Elementos de estruturas marítimas
5 Ataque gelo/degelo		
XF1	Saturação moderada em água, sem produto descongelante	Superfícies verticais de betão expostas à chuva e ao gelo
XF2	Saturação moderada em água, com produto descongelante	Superfícies verticais de betão de estruturas rodoviárias expostas ao gelo e a produtos descongelantes transportados pelo ar
XF3	Saturação elevada em água, sem produtos descongelantes	Superfícies horizontais de betão expostas a chuva e ao gelo
XF4	Saturação elevada em água, com produtos descongelantes ou com água do mar	Estradas e tabuleiros de pontes expostos a produtos descongelantes Superfícies de betão expostas a pulverizações directas contendo produtos descongelantes e expostas ao gelo Zonas sujeitas aos efeitos da rebentação de estruturas marítimas expostas ao gelo
6 Ataque químico		
XA1	Ambiente químico ligeiramente agressivo, de acordo com a EN 206-1, ver o Quadro 2	Terrenos naturais e água do terreno
XA2	Ambiente químico moderadamente agressivo, de acordo com a EN 206-1, ver o Quadro 2	Terrenos naturais e água do terreno
XA3	Ambiente químico altamente agressivo, de acordo com a EN 206-1, ver o Quadro 2	Terrenos naturais e água do terreno

Fonte: NP EN 1992-1-1.

Finalmente, para cada tipo de classe, deve-se atender os valores normativos apresentados na Tabela 11.

Tabela 11: Valores recomendáveis de abertura de fissuras e tensão (Quadro 7.1N da NP EN 1992-1-1).

Classe de Exposição	*Elementos de betão armado e elementos de betão pré-esforçado com armaduras não aderentes*	*Elementos de betão pré-esforçado com armaduras aderentes*
	Combinação de acções quase-permanente	*Combinação de acções frequente*
X0, XC1	*0,4*[1]	*0,2*
XC2, XC3, XC4	*0,3*	*0,2*[2]
XD1, XD2, XS1, XS2, XS3		*Descompressão*
NOTA 1: *Para as classes de exposição X0 e XC1, a largura de fendas não tem influência sobre a durabilidade e este limite é estabelecido para dar em geral um aspecto aceitável. Na ausência de especificações no que respeita ao aspecto, este limite poderá ser reduzido.* ***NOTA 2:*** *Para estas classes de exposição deverá verificar-se, ainda, a descompressão para a combinação quase-permanente de acções.*		

Fonte: NP EN 1992-1-1.

Onde, para o caso de protensão aderente, as classes X0 e XC1 não podem apresentar abertura de fissuras superiores a 0,2 mm na combinação frequente. Para as classes XC2, XC3 e XC4 os elementos não podem apresentar abertura de fissuras superiores a 0,2 mm para a combinação frequente e nem tensões de tração para a combinação quase permanente. Finalmente, para as classes XD1, XD2, XS1, XS2 e XS3, a norma exige a descompressão, ou seja, não permite o aparecimento de tensões de tração para a combinação frequente.

A relevância em se apresentar os critérios estabelecidos por normas de países que apresentam larga experiência na construção civil está em verificar quais os pontos que diferem da norma nacional e entender o motivo para que haja tal discrepância. Desta forma torna-se viável aprimorar a norma brasileira de forma com que aumente credibilidade desta para com a segurança das estruturas.

O programa aqui desenvolvido dá a possibilidade de o usuário editar esses valores normativos, adaptando seus coeficientes para que se torne possível estudar o comportamento das vigas frente a esses novos dados e que, além disso, seja possível comparar os resultados de dimensionamento e verificação quando adotados normas diferentes.

4 CONSIDERAÇÕES TEÓRICAS PARA PROGRAMAÇÃO

Neste capítulo serão descritas todas as considerações técnicas utilizadas para o desenvolvimento do programa. Como se sabe, a lógica utilizada para resolver um problema por meio de programas é diferente da lógica utilizada por meios manuais. Por exemplo, para resolver um problema de engenharia o projetista muitas vezes se utiliza de ábacos e tabelas para otimizar o tempo de cálculo. No entanto, essa não é a melhor maneira de se trabalhar com linguagem programacional. Quando se utiliza ferramentas computacionais, o ideal é trabalhar diretamente com funções e equações prontas (a partir das quais foram feitas essas tabelas e os ábacos). Há, ainda, situações onde processos numéricos por tentativa são utilizados e que seria muito trabalhoso seu cálculo de forma manual como.

Além disso, para desenvolver um programa de qualidade, a aplicação direta das equações deve implicar em algumas situações de contorno que impeçam resultados absurdos ou impossíveis como, por exemplo, em divisões por zero, entrada de valores não negativos em alguns campos, dentre outros.

4.1 CÁLCULO DAS CARACTERÍSTICAS GEOMÉTRICAS E DOS ESFORÇOS SOLICITANTES

A geometria das seções transversais que o programa trabalha é: retangular, “I”, “T” e “T” invertido, além da situação de seção composta para todas elas. O cálculo da área, da inércia e do centro de gravidade da seção é feito partindo-se das coordenadas de cada um dos vértices, utilizando-se, em seguida, as equações para o cálculo. Essa teoria foi encontrada em Richardson (2009):

$$\mathrm{A} = \frac{1}{2} * \sum_{\mathrm{i}=0}^{\mathrm{n}-1} (\mathrm{x_i y_{i+1}} - \mathrm{x_{i+1} y_i}) \quad (\mathrm{Eq.}\, 4.1.1)$$

$$\mathrm{CG_x} = \frac{1}{6\mathrm{A}} * \sum_{\mathrm{i}=0}^{\mathrm{n}-1} (\mathrm{x_i} + \mathrm{x_{i+1}}) * (\mathrm{x_i y_{i+1}} - \mathrm{x_{i+1} y_i}) \quad (Eq.\, 4.1.2)$$

$$\mathrm{CG_y} = \frac{1}{6\mathrm{A}} * \sum_{\mathrm{i}=0}^{\mathrm{n}-1} (\mathrm{y_i} + \mathrm{y_{i+1}}) * (\mathrm{x_i y_{i+1}} - \mathrm{x_{i+1} y_i}) \quad (Eq.\, 4.1.3)$$

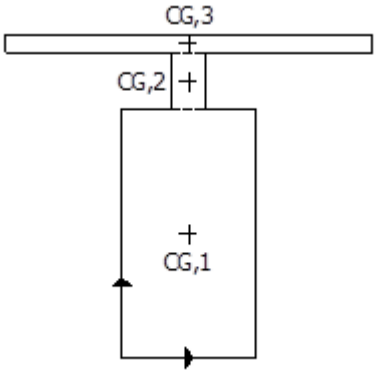
$$I_x = \frac{1}{12} * \sum_{i=0}^{n-1} ({y_i}^2 + y_i y_{i+1} + {y_{i+1}}^2) \quad (Eq. 4.1.4)$$

Com o momento de inércia e o centro de gravidade da seção, calcula-se o módulo de resistência à flexão nas fibras extremas superiores (W_s) e inferiores (W_i).

Quando se considera a seção composta o cálculo é feito com o mesmo raciocínio utilizado quando calculado manualmente

.

Figura 24: Divisão das áreas da seção composta.



Partindo-se das características das seções já conhecidas, a seção composta se baseia em dois retângulos. Logo, o cálculo da componente “y” da seção composta se dá pela seguinte equação:

$$CG_y = \frac{y_1 A_1 + y_2 A_2 + y_3 A_3}{A_1 + A_2 + A_3} \quad (Eq. 4.1.5)$$

O cálculo do momento de inércia da seção se dá pelo teorema dos eixos paralelos:

$$I_x = I_1 + A_1 * {d_1}^2 + I_2 + A_2 * {d_2}^2 + I_3 + A_3 * {d_3}^2 \quad (Eq. 4.1.6)$$

E com os novos valores de ycg e de “I” se calcula os novos valores de Ws e Wi.

Na etapa de obtenção dos valores dos esforços, utiliza-se das teorias de resistência dos materiais. Como o programa trabalha apenas com a hipótese de carregamento uniformemente distribuído em vigas biapoiadas, sabe-se que o diagrama de momento fletor resulta em uma parábola do segundo grau representada pela seguinte equação:

$$M = \frac{p}{2} * (L * x - x^2) \quad (Eq. 4.1.7)$$

O programa considera os seguintes carregamentos atuantes: peso próprio, da laje, da capa, da parede, do revestimento e da acidental máxima e mínima. O peso próprio é calculado automaticamente partindo-se da área da seção bruta de concreto e do peso específico do concreto armado de 25 kN/m³.

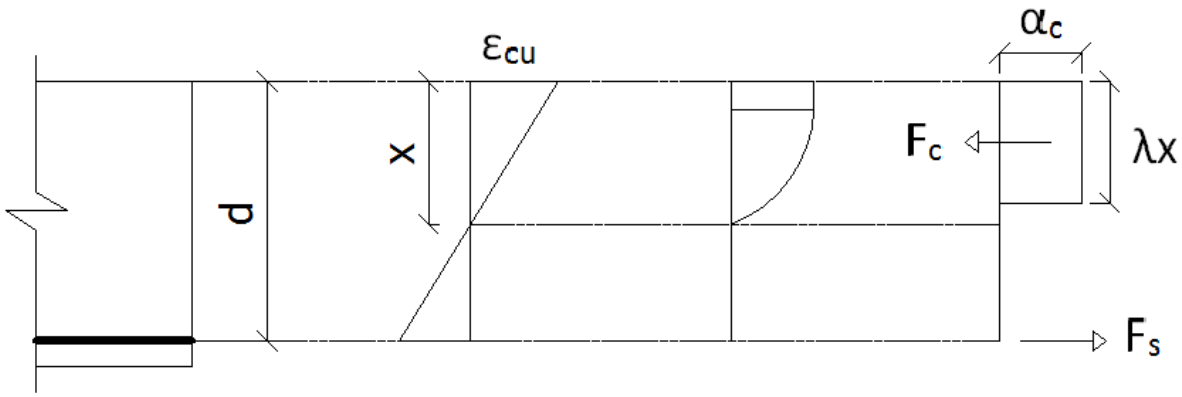
Finalmente, são salvos os momentos em cada décimo de vão e nas seções do comprimento de transferência inferior e superior.

4.2 CÁLCULO E VERIFICAÇÃO DA ARMADURA LONGITUDINAL

4.2.1 Dimensionamento

Para o dimensionamento da armadura de flexão, o programa parte do princípio de equilíbrio de forças resultantes entre as tensões do aço (F_s) e do concreto (F_c).

Figura 25: Diagrama parábola-retângulo e diagrama simplificado de tensões no estado limite último.



Fonte: Baseado em Carvalho e Figueiredo Filho (2009).

De forma a permitir a utilização dos novos critérios normativos, o programa permite a entrada desses novos coeficientes estabelecidos, de forma a aumentar a potencialidade do programa. Portanto, foi necessário deduzir as equações de dimensionamento de forma a utilizar esses valores como variáveis e não mais como constantes. Portanto:

$$F_s = F_c \quad (Eq.4.2.1.1)$$

$$F_c = (\alpha_c * f_{cd}) * b_w * (\lambda x) \quad (Eq.4.2.1.2)$$

$$z = d - \frac{\lambda x}{2} \quad (Eq.4.2.1.3)$$

Sabendo que $M_d = F_c * z$:

$$M_d = (\alpha_c * f_{cd}) * b_w * (\lambda x) * \left(d - \frac{\lambda x}{2}\right)$$

$$M_d = (\alpha_c * f_{cd}) * b_w * \left(d * \lambda x - \frac{\lambda^2 x^2}{2}\right)$$

$$\frac{M_d}{\alpha_c * f_{cd} * b_w * d^2} = \frac{\lambda x}{d} - 0{,}5 * \frac{\lambda^2 x^2}{d^2}$$

Chamando,

$$KMD, \alpha = \frac{M_d}{\alpha_c * f_{cd} * b_w * d^2} \quad (Eq.4.2.1.4)$$

$$KX, \lambda = \frac{\lambda x}{d} \quad (Eq.4.2.1.5)$$

Chega-se, a:

$$KMD,\alpha = KX,\lambda - 0{,}5 * KX,\lambda^2 \quad (Eq.\,4.2.1.6)$$

Sabendo que x varia entre 0 e d na flexão simples, determina-se os limites da equação:

p/ x = 0, KMD,α = 0;

p/ x = d, KMD,α = λ – 0,5 * λ²;

Dividindo os dois termos da equação da alavanca (z) por d, chega-se a:

$$\frac{z}{d} = 1 - 0{,}5 * \frac{\lambda x}{d}$$

$$KZ,\phi = 1 - 0{,}5 * KX,\lambda \quad (Eq.\,4.2.1.7)$$

E, finalmente, sabendo que a equação para definir a área de armadura é:

$$A_p = \frac{M_d}{z * f_s} = \frac{M_d}{KZ,\phi * d * f_s} \quad (Eq.\,4.2.1.8)$$

A equação que relaciona a altura da linha neutra com as deformações:

$$KX,\lambda = \frac{\varepsilon_c * \lambda}{\varepsilon_c + \varepsilon_s} \quad (Eq.\,4.2.1.9)$$

Logo, em função de cada domínio de deformação, obtém-se uma equação para o calculo da deformação do concreto (ε_c), para o domínio 2, e do aço (ε_s), para o domínio 3.

$$\varepsilon_c = \frac{\varepsilon_s}{\frac{\lambda}{KX,\lambda} - 1} \quad (Eq.\,4.2.1.10)$$

$$\varepsilon_s = \varepsilon_c * \left(\frac{\lambda}{KX,\lambda} - 1\right) \quad (Eq.\,4.2.1.11)$$

Com estas equações definidas, criou-se uma tabela com os valores adimensionais semelhante àquela presente em Carvalho e Figueiredo Filho (2009) com a diferença de que no cálculo dos valores adimensionais estão presentes os coeficientes α_c e λ, relativo ao tipo de concreto. Logo, esta nova tabela apresenta a seguinte forma:

Tabela 12: Valores para cálculo da armadura longitudinal de seções retangulares.

f_{ck} (MPa)			50		60		70		80		90	
			λ=0,8	α_c=0,85	λ=0,775	α_c=0,808	λ=0,75	α_c=0,765	λ=0,725	α_c=0,723	λ=0,7	α_c=0,68
KMD,α	KX,λ	KZ,ϕ	ε_c	ε_s	ε_c	ε_s	ε_c	ε_s	ε_c	ε_s	ε_c	ε_s
0,01	0,0101	0,9950	0,1272	10	0,1314	10	0,1358	10	0,1406	10	0,1457	10
0,02	0,0202	0,9899	0,2591	10	0,2677	10	0,2768	10	0,2867	10	0,2972	10
0,03	0,0305	0,9848	0,3959	10	0,4092	10	0,4234	10	0,4386	10	0,4550	10
0,04	0,0408	0,9796	0,5379	10	0,5562	10	0,5758	10	0,5968	10	0,6195	10
0,05	0,0513	0,9743	0,6854	10	0,7091	10	0,7345	10	0,7617	10	0,7911	10
0,06	0,0619	0,9690	0,8389	10	0,8683	10	0,8998	10	0,9338	10	0,9704	10
0,07	0,0726	0,9637	0,9987	10	1,0342	10	1,0724	10	1,1135	10	1,1578	10
0,08	0,0835	0,9583	1,1652	10	1,2073	10	1,2526	10	1,3014	10	1,3541	10
0,09	0,0945	0,9528	1,3389	10	1,3880	10	1,4410	10	1,4981	10	1,5600	10
0,1	0,1056	0,9472	1,5203	10	1,5771	10	1,6382	10	1,7044	10	1,7760	10
0,12	0,1282	0,9359	1,9087	10	1,9824	10	2,0621	10	2,1485	10	2,2425	10
0,14	0,1515	0,9243	2,3356	10	2,4293	10	2,5307	10	2,6411	10	2,7614	10
0,16	0,1754	0,9123	2,8078	10	2,9248	10	3,0521	10	3,1909	10	3,3430	10
0,18	0,2000	0,9000	3,3333	10	3,4783	10	3,5000	9,63	3,5000	9,19	3,5000	8,75
0,2	0,2254	0,8873	3,5000	8,92	3,5000	8,53	3,5000	8,15	3,5000	7,76	3,5000	7,37
0,22	0,2517	0,8742	3,5000	7,63	3,5000	7,28	3,5000	6,93	3,5000	6,58	3,5000	6,24
0,24	0,2789	0,8606	3,5000	6,54	3,5000	6,23	3,5000	5,91	3,5000	5,60	3,5000	5,28
0,26	0,3072	0,8464	3,5000	5,62	3,5000	5,33	3,5000	5,05	3,5000	4,76	3,5000	4,48
0,28	0,3367	0,8317	3,5000	4,82	3,5000	4,56	3,5000	4,30	3,5000	4,04	3,5000	3,78
0,3	0,3675	0,8162	3,5000	4,12								
0,34	0,4343	0,7828	3,5000	2,95								

Toda a base de cálculo está definida. Para melhorar a qualidade do programa, impedindo que alguns valores levem a conflitos matemáticos (como divisões por zero ou raiz de números negativos), é necessário verificar se algum dos coeficientes pode levar a este tipo de problema.

Resolvendo a equação de segundo grau:

$$KX,\lambda = 1 \pm \sqrt{1 - 2 * KMD,\alpha}$$

Como x deve estar contido entre 0 e d para o caso de flexão simples, pode-se eliminar a solução de soma da equação. Logo:

$$KX,\lambda = 1 - \sqrt{1 - 2 * KMD,\alpha} \quad (Eq.4.2.1.12)$$

$$1 - 2 * KMD,\alpha \geq 0$$

$$KMD,\alpha \leq 0{,}5$$

Até este momento foram apresentadas as fórmulas e as hipóteses adotadas para o dimensionamento da armadura de protensão. No entanto existe uma série de situações que tornam o problema ligeiramente mais complexo do que se aparenta. Por exemplo, essa rotina de programação deve considerar a todas as situações possíveis partindo do mais simples, como uma seção retangular simples, até a mais complexa de uma seção “I” com capa solidarizada e f_{ck} diferente. Neste último caso existe a possibilidade, ainda, de a linha neutra estar localizada na alma da peça pré-moldada.

Como se pode observar, generalizar uma rotina que englobe todas as situações não é simples. Para tanto, a lógica que melhor se enquadrou nesse problema foi a de tentativa para calcular a influência das abas e, quando resta apenas a seção retangular, utilizam-se

as equações apresentadas anteriormente. Nesse modelo, o usuário insere um valor inicial de x/d (cujo valor padrão é o de 0,45) a partir do qual o programa faz a primeira tentativa calculando toda a seção comprimida das abas. Quando todas as abas foram consideradas, utilizam-se as equações na seção retangular para definir os coeficientes e a posição da linha neutra final. Se a diferença entre a profundidade da linha neutra inicial e final for menor que 0,01 a armadura estará dimensionada.

Em relação à consideração de tipos diferentes de concreto, o programa faz uma média ponderada entre os valores de f_{ck}, α_c e γ_c da seção retangular e a utiliza nas equações.

As Figuras 26 e 27 apresentam as regiões consideradas no cálculo de uma seção com múltiplas abas.

Primeiramente, desconsidera-se a região da mísula. Em seguida define-se a região das abas em função da menor largura de cada seção. E, finalmente, considera-se a seção retangular utilizando as equações deduzidas.

Figura 26: Consideração da região comprimida de uma seção com múltiplas abas com largura mínima definida pela região compreendida pelas lajes.

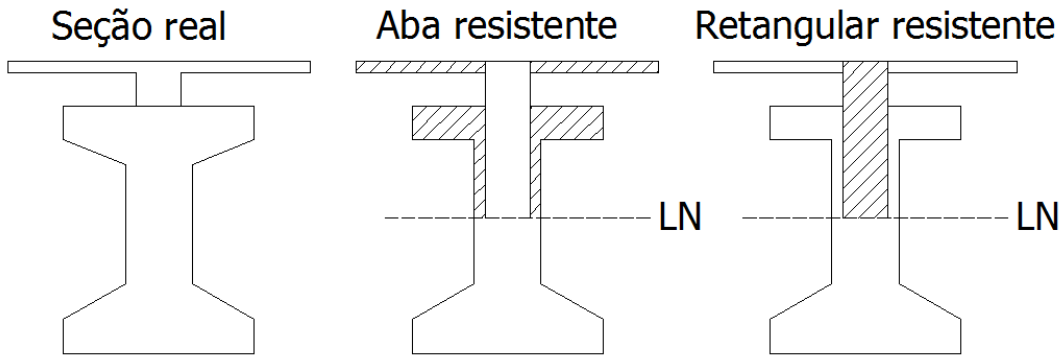
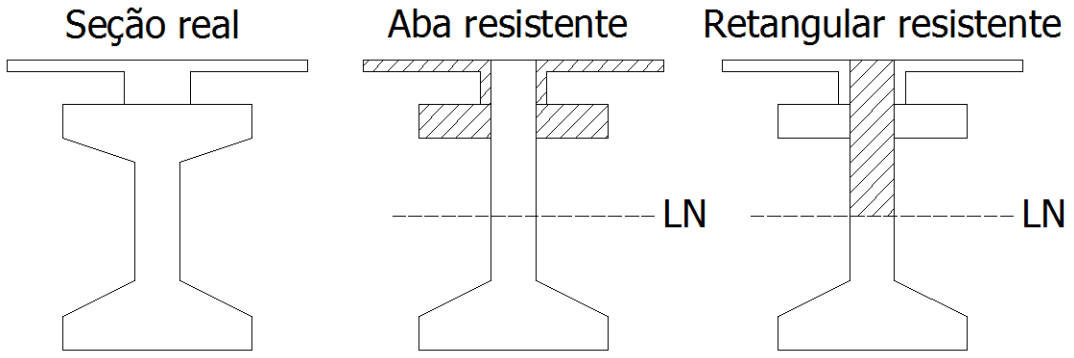


Figura 27: Consideração da região comprimida de uma seção com múltiplas abas com largura mínima definida pela alma da viga pré-moldada.



Finalizando a parte de dimensionamento destaca-se a etapa do cálculo da tensão atuante na cordoalha em função de sua deformação. Como descrito na norma, pode-se

simplificar o diagrama tensão-deformação de armaduras ativas. Em Carvalho (2012), encontram-se as equações que representam essa curva.

- p/ $\varepsilon_p < f_{pyd} / E_p$

$$\sigma_{sd} = \frac{f_{pyd} * \varepsilon_p}{\varepsilon_{yd}} \quad (Eq. 4.2.1.13)$$

- p/ $\varepsilon_p \geq f_{pyd} / E_p$

$$\sigma_{sd} = f_{pyd} + \left(\frac{f_{ptd} - f_{pyd}}{\varepsilon_u - \varepsilon_{yd}}\right) * \left(\varepsilon_p - \varepsilon_{yd}\right) \quad (Eq. 4.2.1.14)$$

O oposto, ou seja, a definição da deformação para uma dada tensão pode ser definida pelas seguintes equações:

- p/ $\sigma_p < f_{pyd}$

$$\varepsilon_p = \frac{\varepsilon_{yd} * \sigma_p}{f_{pyd}} \quad (Eq. 4.2.1.15)$$

- p/ $\sigma_p \geq f_{pyd}$

$$\varepsilon_p = \varepsilon_{pyd} + \left(\frac{\sigma_p - f_{pyd}}{f_{ptd} - f_{pyd}}\right).\left(\varepsilon_u - \varepsilon_{yd}\right) \quad (Eq. 4.2.1.16)$$

Dessa forma, possuindo os valores de tensão de escoamento característico (f_{pyk}), de resistência à tração (f_{ptk}) e o alongamento após a ruptura (ε_{uk}) de uma cordoalha, pode-se calcular as tensões decorrentes do pré-alongamento e do carregamento.

O diagrama padrão é baseado nos valores proposto por Vasconcelos (1980), adotando-se: f_{pyd} = 1460 MPa; ε_{yd} = 7,3‰; f_{ptd} = 1626 MPa e ε_u = 35‰, para cordoalha do tipo CP 190. Ressalta-se que o usuário poderá editar esses valores para adequar o dimensionamento às informações de cada projeto.

Até o momento, calcularam-se as deformações decorrentes do carregamento e do pré-alongamento. Por último, falta calcular a deformação até chegar ao estado de descompressão (ε_7), que é feito conforme a seguinte equação:

$$\varepsilon_7 = \left(\frac{N_p}{A_c} + \frac{N_p * {e_p}^2}{I}\right) * \frac{1}{E_c} \quad (Eq. 4.2.1.17)$$

O módulo de dimensionamento parte dos princípios de cálculo definidos até o momento. Por meio das informações de carregamento e de seção informadas pelo usuário, o programa calcula a área de armadura ativa necessária para satisfazer o ELU. Em seguida, o próprio usuário posiciona a armadura da maneira que lhe parecer mais conveniente. O programa possui um módulo gráfico que permite a visualização da posição da armadura no momento da inserção de seus dados. Caso o usuário deseje, o programa permite a inserção de armadura passiva junto à ativa, de forma a utilizar armadura mista.

Após finalizado o detalhamento manual passa-se a ter um novo valor de altura útil (d_{real}), logo, o programa realiza todos os cálculos novamente com este valor de altura útil.

Para definição da altura útil real, é necessário encontrar o centro de força da armadura. Como a viga protendida permite a utilização simultânea de um ou mais tipos de aços, deve-se levar em consideração essa diferença de propriedades entre elas. Por exemplo, utilizando aço CA-50 com aço de protensão CP190, o centro de força tende a se aproximar mais do aço de protensão (desde que a área de aço seja igual para ambas). A função do programa que calcula essa posição utiliza a seguinte equação de forma genérica:

$$\mathrm{CG_{yp}} = \frac{\mathrm{y_1 A_1 \sigma_1 + y_2 A_2 \sigma_2 + \cdots + y_n A_n \sigma_n}}{\mathrm{A_1 \sigma_1 + A_2 \sigma_2 + \cdots + A_n \sigma_n}} \quad (Eq.\,4.2.1.18)$$

Onde a tensão do aço de armadura passiva é sua tensão de escoamento e do aço para protensão é sua tensão final em utilização. A tensão final utilizada pelo programa é aquela inserida pelo usuário no momento de entrar com o valor da tensão de estiramento das cordoalhas e a porcentagem de perda de protensão total.

Como forma padrão do programa, o pré-dimensionamento de A_p é feito considerando uma altura útil fictícia ("$d_{fictício}$") equivalente à altura total da seção menos quatro centímetros.

$$d_{fictício} = h_{total} - 4cm \quad (Eq.\,4.2.1.19)$$

Em relação à tensão final, esta é calculada considerando-se, primeiramente, os valores de tensão inicial de protensão, estabelecidos pela norma, para a pré-tração e com aço de relaxação baixa:

$$\sigma_{pi} \leq \begin{cases} 0{,}77 * f_{ptk} \\ 0{,}85 * f_{pyk} \end{cases} \quad (Eq.\,4.2.1.20)$$

Em seguida considerou-se uma perda de protensão de 25%.

Descrita a etapa de pré-dimensionamento, há um detalhe importante a ser considerado de forma a melhorar a precisão do programa. Quando a seção transversal apresenta apenas armadura ativa, rapidamente se define a altura útil desta por meio do centro de força, afinal, considera-se que todas as cordoalhas apresentem a mesma tensão. Logo, a equação pode ser simplificada na seguinte forma:

$$\mathrm{CG_{yp}} = \frac{\mathrm{y_1 A_1 + y_2 A_2 + \cdots + y_n A_n}}{\mathrm{A_1 + A_2 + \cdots + A_n}} \quad (Eq.\,4.2.1.21)$$

A tensão não é necessária para a definição da altura útil.

Mas, considerando que existam armaduras passivas na seção, a tensão passa a ser fundamental para a correta definição do centro de gravidade do aço. Utilizar a tensão de escoamento do aço de armadura passiva é coerente, afinal, no dimensionamento, espera-se que o concreto trabalhe nos domínios 2 ou 3. No entanto, para definir a tensão atuante na armadura ativa, não basta considerar, apenas, a deformação de pré-alongamento após as perdas (ε_p). Há que se considerar, ainda, as deformações até a descompressão (ε_7) e a tensão decorrente do carregamento (ε_s). Em relação à perda de protensão decorrente da deformação de descompressão, os parâmetros utilizados são conhecidos. Contudo, para

definir a deformação provocada pelo carregamento e, consequentemente, o aumento de tensão na cordoalha, é necessário conhecer a altura útil (cuja tensão utilizada para definir considera apenas o pré-alongamento dos cabos com as perdas totais).

Portanto, após realizar os cálculos com a altura útil real inicial ($d_{real,i} = d_{real}$) recalcula-se a altura útil com a tensão atuante na cordoalha já considerando todas as deformações, chegando, finalmente, a uma altura útil final ($d_{real,f}$).

Se a diferença entre elas for superior a 0,1 cm (ou 0,001 m), o programa realiza o dimensionamento utilizando o novo valor de altura útil. Este procedimento é executado até que a diferença seja igual ou inferior a 0,1 cm. Dessa forma, garante-se maior precisão no dimensionamento da armadura.

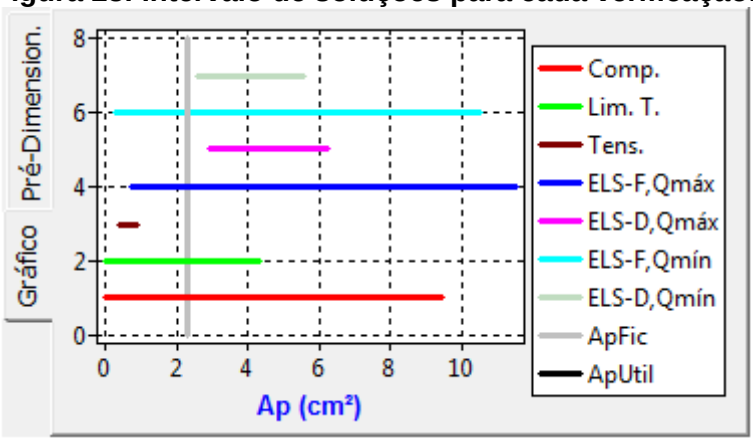
Lembrando que este procedimento é feito, apenas, quando há armaduras com tensões diferentes.

Explicado como funciona o dimensionamento da armadura, faz-se, agora, algumas considerações sobre o pré-dimensionamento das cordoalhas. Esta etapa possui o intuito de direcionar o usuário em relação à quantidade de armadura ativa necessária para o dimensionamento da peça no ELU.

Para tanto se fez uso do estudo apresentado no capítulo 3 sobre os limites de protensão em uma viga. Resumidamente, o estudo propõe a definição de uma série de intervalos de armadura ativa (A_p) que satisfaçam as equações de verificação de tensão em serviço. Fazendo isso para cada uma das verificações necessárias se restringe a quantidade de armadura ativa que pode ser utilizada para satisfazer as condições. No referido estudo foi possível definir cinco situações particulares que auxiliam o usuário a dimensionar a armadura ativa de uma viga.

O programa realiza essa função, como se pode observar na Figura 28.

Figura 28: Intervalo de soluções para cada verificação.

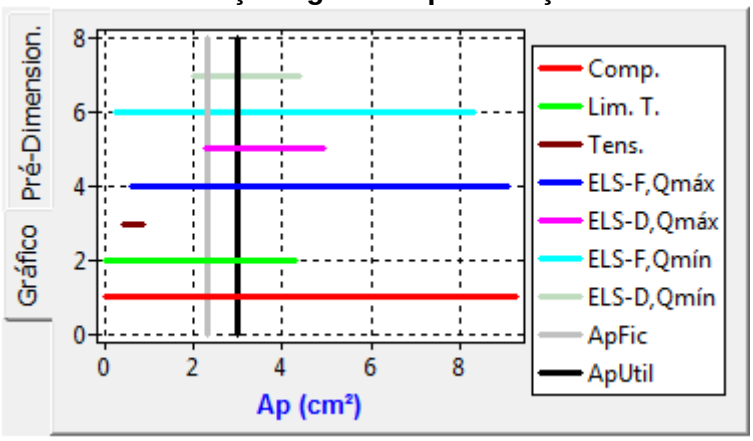


Nas ordenadas apresentam-se uma série de números cuja legenda ao lado define qual verificação está sendo calculada.

No eixo das abcissas se pode observar a área, em cm², da armadura ativa. A linha vertical (em cinza) que, de acordo com a legenda, representa o "ApFic" (A_p fictício), é a quantidade de armadura ativa necessária para satisfazer o ELU.

Durante o pré-dimensionamento o gráfico apresenta apenas uma linha vertical. Durante o processo de dimensionamento, após o usuário entrar com um arranjo de armadura ativa, o gráfico é atualizado com os novos valores de tensão e altura útil, além de uma segunda linha vertical "ApUtil" (A_p utilizado) que representa a quantidade de armadura ativa que existe na seção, conforme os dados inseridos pelo usuário. A Figura 29 apresenta o gráfico gerado pela função de dimensionamento do programa.

Figura 29: Intervalo de soluções geradas pela função de dimensionamento.



A linha vertical "ApFic" indica que é necessário utilizar, no mínimo, 2 cm² para satisfazer o ELU. Enquanto que a linha vertical "ApUtil" representa a quantidade de armadura que o usuário inseriu no detalhamento manual.

Para mais detalhes sobre este trabalho ler o capítulo 3.

Por último, há a necessidade de se considerar a ação da armadura passiva na composição da resistência da seção à ruptura. Para isso, o programa realiza a conversão da área do aço convencional posicionado na seção em área de aço de protensão (A_s para A_P). Em Carvalho (2012), apresenta-se uma forma de calcular esse tipo de situação:

$$\frac{M_d}{z} = A_p * \sigma_{pd} + A_s * f_{yd}$$

Onde,

$$A_s * f_{yd} = A_{s,25} * f_{yd,25} + A_{s,50} * f_{yd,50} + A_{s,60} * f_{yd,60} \quad (Eq.4.2.1.22)$$

Por meio do arranjo de barras e cordoalhas inseridas pelo usuário, verifica-se a igualdade de forças entre o momento atuante e a somatório da força dos aços (Fp + Fs) e, caso:

$$A_p * \sigma_{pd} + A_s * f_{yd} \geq \frac{M_d}{z} \quad (Eq. 4.2.1.23)$$

O arranjo da armadura satisfaz o ELU. Caso contrário, não o satisfaz.

4.3 VERIFICAÇÃO DE TENSÕES

As exigências de durabilidade previstas na NBR 6118:2014 estão apresentadas na Tabela 5 do texto e na tabela 13.3 da norma. Em função do tipo de protensão e da classe de agressividade ambiental (CAA), define-se quais critérios devem ser atendidos. Para a pré-tração e CAA I, deve-se atender ao limite de 0,2 mm de abertura de fissuras na combinação frequente.

Para pré-tração e CAA II, são feitas as verificações de formação de fissuras, para a combinação frequente, e a descompressão, para a combinação quase permanente.

Para pré-tração e CAA III e IV, verifica-se a descompressão, para combinação frequente, e a formação de fissuras para a combinação rara.

A verificação de tensões, além da situação em serviço, deve ser feita para a situação de vazio. Esta que, como já explicado, se faz a verificação simplificada prevista no item 17.2.4.3.2 da norma.

Os coeficientes padrão para cada uma das verificações são as seguintes:

- Verificação de tração excessiva em vazio: 1,2 $f_{ctm,j}$;
- Verificação de compressão excessiva em vazio: 0,7 $f_{ck,j}$;
- Verificação de formação de fissuras: 1,0 f_{ctm};
- Verificação de descompressão: 0,0 f_{ctm}.

Para as verificações de tensão, o programa aplica as seguintes equações para cada uma das seções de décimo de vão:

- Na borda inferior:

$$\sigma_i = \frac{n_i * A_{p,1} * \sigma_{p,i}}{A_c} + \frac{n_i * A_{p,1} * \sigma_{p,i} * e_{p,i}}{W_i} - \frac{M_{g,i}}{W_i} - \frac{M_{g,f} + \gamma_{f2} * M_q}{W_{i,t=\infty}} + \\ + \frac{n_s * A_{p,1} * \sigma_{p,s}}{A_c} - \frac{n_s * A_{p,1} * \sigma_{p,s} * e_{p,s}}{W_i} \quad (Eq. 4.3.1)$$

- Na borda superior:

$$\sigma_s = \frac{n_i * A_{p,1} * \sigma_{p,i}}{A_c} - \frac{n_i * A_{p,1} * \sigma_{p,i} * e_{p,i}}{W_s} + \frac{M_{g,i}}{W_s} + \frac{M_{g,f+}\gamma_{f2} * M_q}{W_{s,t=\infty}} + \\ + \frac{n_s * A_{p,1} * \sigma_{p,s}}{A_c} + \frac{n_s * A_{p,1} * \sigma_{p,s} * e_{p,s}}{W_s} \quad (Eq. 4.3.2)$$

Onde,

σ_i/σ_s: tensão na borda inferior e superior, respectivamente;

n_i/n_s: número de cordoalhas na borda inferior e superior, respectivamente;

$A_{p,1}$: área da seção de uma cordoalha;

$\sigma_{p,i}/\sigma_{p,s}$: tensão na cordoalha inferior e superior, respectivamente;

$e_{p,i}/e_{p,s}$: excentricidade dos cabos inferiores e superiores, respectivamente;

$M_{g,i}/M_{g,f}/M_q$: momento decorrente do carregamento permanente estrutural (viga, laje e capa), carregamento permanente não-estrutural (alvenaria e revestimento) e acidental, respectivamente;

γ_{f2}: coeficiente de ponderação das ações para o ELS, definidos pela NBR6118:2014;

A_c: área bruta de concreto da seção transversal do elemento estrutural;

W_i/W_s: módulo de rigidez à flexão da seção simples inferior e superior, respectivamente;

$W_{i,t=\infty}/W_{s,t=\infty}$: módulo de rigidez à flexão da seção composta inferior e superior, respectivamente.

Na situação em vazio, como se trabalha com a seção simples, a viga resiste apenas ao carregamento de peso próprio. Para a situação em serviço, os carregamentos permanentes, com exceção do revestimento, são resistidos pela seção simples. A carga acidental e o revestimento são resistidos pela seção composta.

Durante a verificação, o usuário também tem a possibilidade de realizar o isolamento de cabos nas seções, escolhendo-as e entrando com o número de cabos que se deseja isolar, o programa subtrai este valor do total de cordoalhas, reduzindo a tensão de protensão até a seção escolhida. O isolamento de cabos, que será mais bem explicado no tópico 4.4, é efetuado para reduzir o efeito de determinado número de cordoalhas em uma seção onde as tensões excedam os limites estabelecidos pela norma. Sua técnica consiste em revestir a cordoalha, até a seção desejada, com um material que impeça a aderência entre a cordoalha e o concreto.

Além do isolamento dos cabos, o programa ainda considera o efeito do comprimento de regularização inferior e superior das cordoalhas. Seu funcionamento, junto ao isolamento de cabos, será explicado no tópico referente a este assunto, mas seu efeito é considerado em cada uma das seções, inclusive na seção do comprimento de regularização inferior ($l_{p,i}$) e superior ($l_{p,s}$), que aparecem na aba “Resultados”, também.

Em relação às perdas de protensão consideradas para o cálculo do esforço normal atuante na seção, este é obtido na aba “Perdas de protensão”. O usuário pode inserir os valores manualmente para cada seção ou esta pode ser calculada automaticamente pelo programa.

Com os dados e posições da armadura inserida pelo usuário somada às informações de carregamento descritas anteriormente, o programa possui todos os dados necessários para obtenção da tensão atuante nas bordas e em cada seção.

Finalmente, em função das classes de agressividade ambiental e do tipo de edificação, o programa define, as exigências e combinações a serem atendidas e os coeficientes de ponderação do momento acidental (M_q). Utilizando-se uma função condicional, rapidamente obtém-se essas informações.

Os valores de tensão para cada seção é, então, comparado com os limites estabelecidos pela norma. Há que ressaltar que o usuário pode alterar os valores padrão do programa para os limites de tensão. Este utiliza a norma brasileira como padrão, mas é possível editar seus valores para outras análises e considerar outros valores. Por exemplo, a ACI (American Concrete Institute) e o EC (Eurocode) consideram que o limite de compressão em vazio é de 60% do $f_{ck,j}$, enquanto que a NBR considera o valor de 70% do $f_{ck,j}$. Como padrão do programa adota-se o valor de 70%, mas o usuário pode alterá-lo no menu “Editar” para qualquer outro valor que deseje.

4.4 DISTÂNCIA DE REGULARIZAÇÃO E ISOLAMENTO DE CABOS

Em relação ao módulo de cálculo do comprimento de regularização (lp), o usuário pode entrar com um valor aleatório nas caixas de texto Lp,sup e Lp,inf. Por outro lado, o usuário também pode permitir que o programa calcule automaticamente este valor. O cálculo padrão do comprimento de regularização considera apenas as equações normativas para cordoalhas de 3 e 7 fios.

O primeiro passo consiste em calcular os valores de f_{ctd} e f_{bpd} por meio das equações abaixo:

$$f_{ctd} = \frac{0{,}21 * \sqrt[3]{{f_{cj}}^2}}{1{,}4} \quad (Eq.\,4.4.1)$$

$$f_{bpd} = \eta_{p1} * \eta_{p1} * f_{ctd} \quad (Eq.\,4.4.2)$$

Onde η_{p1} e η_{p2} são, respectivamente, os coeficientes dependentes do tipo de cordoalha e situação de boa ou má aderência. O valor padrão do programa considera o valor de 1,2 para η_{p1} (cordoalha de três e sete fios) e 1 para η_{p2} (situação de boa aderência).

Em seguida, em função do tipo de liberação do dispositivo de tração, calcula-se o comprimento de transferência (l_{bpt}) por meio de uma das equações abaixo:

- Liberação do dispositivo de tração gradual e para cordoalhas de 3 e 7 fios:

$$l_{bpt} = \frac{7 * 0{,}5 * \phi * \sigma_{pi}}{36 * f_{bpd}} \quad (Eq.\,4.4.3)$$

- Liberação do dispositivo de tração não-gradual e para cordoalhas de 3 e 7 fios:

$$l_{bpt} = \frac{7 * 0{,}625 * \phi * \sigma_{pi}}{36 * f_{bpd}} \quad (Eq.\,4.4.4)$$

Nota-se que o comprimento de ancoragem básico para cordoalhas de 3 e 7 fios (l_{bp}) já está inserido nas equações.

Finalmente, para calcular a distância de regularização (l_p) da força de protensão, a NBR 6118:2014 estabelece a seguinte equação:

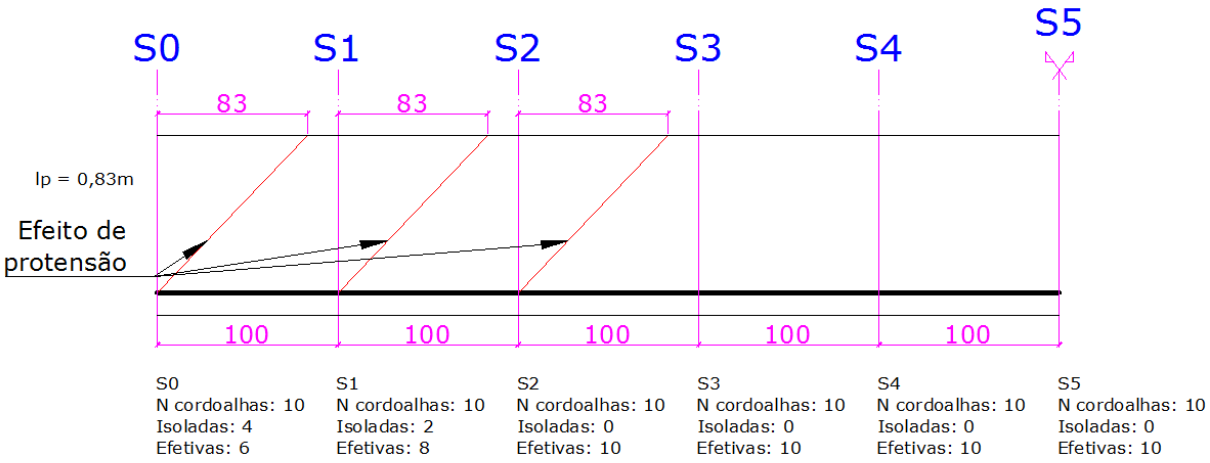
$$l_p = \sqrt{h^2 + (0{,}6 * l_{bpt})^2} \geq l_{bpt} \quad (Eq.\,4.4.5)$$

Sendo h a altura total da peça pré-moldada.

O programa permite tanto que o usuário calcule automaticamente o comprimento de regularização por meio das equações e hipóteses adotadas, assim como permite a entrada de um valor aleatório.

O comprimento de regularização, como dito anteriormente, consiste na variação linear da força de protensão até que toda a seção esteja sob seu efeito. Imaginando, portanto, os casos em que seja necessário isolar algumas cordoalhas, o comprimento de regularização passa a ocorrer Figura 30, as tensões em determinada seção é função não apenas do comprimento de regularização, mas também dos cabos que foram isolados.

Figura 30: Acréscimo de tensão das cordoalhas considerando o comprimento de regularização e o isolamento de cabos.



O isolamento de cabos é uma técnica executiva que consiste em impedir a ocorrência de aderência entre a cordoalha e o concreto, reduzindo o efeito da protensão em determinadas seções onde as tensões superem os limites estabelecidos na norma. Tem-se o costume de revestir, com fita, um determinado número de cordoalhas até a seção com excesso de tensão. Geralmente, quando uma viga apresenta carregamento acidental muito grande, necessitando, assim, de uma área de armadura ativa relativamente alta para resistir

aos esforços, isso pode implicar em tensões muito elevadas em vazio, sendo necessário isolar alguns cabos em sua extremidade.

Como se pode observar na Figura 30 acima, o comprimento de regularização deve ser considerado sempre que houver isolamento de cabos. Na seção S0, seis cordoalhas estão transmitindo esforços para a seção. Na seção S1 mais duas cordoalhas iniciam a transferência de esforços. E, finalmente, na seção S2, as duas últimas cordoalhas passam a transmitir esforços. Somente a partir da seção que está a 283 cm da extremidade é que se pode considerar o efeito das dez cordoalhas.

4.5 PERDAS DE PROTENSÃO

O módulo para o cálculo das perdas de protensão pode ser visto como um programa isolado que permite que o usuário faça uma série de tentativas e estudos sobre as influências ambientais e de tempo. O programa apresenta como resultados os valores de perda de protensão por natureza e em cada seção, que são definidas em décimos de vão e nas seções do comprimento de regularização inferior e superior. Na mesma tela ainda se pode observar os valores de tensão atuante em cada uma das seções descontada as perdas de protensão.

Caso o usuário opte por calcular as perdas conforme as exigências normativas, é necessário que ele tenha conhecimento dos coeficientes e dos parâmetros necessários para o correto cálculo, principalmente no que diz respeito às perdas por fluência e retração, estas que exigem que o usuário saiba qual o tipo de concreto, temperatura e umidade, dentre outros. Caso contrário, é possível estimar uma porcentagem de perda para cada seção e para cada borda (inferior e superior). Essas que serão utilizadas para as verificações de tensão em serviço e em vazio.

Nesse capítulo não será explicada a natureza de cada perda de protensão. Para isso recomenda-se a leitura do capítulo sobre as mesmas na revisão bibliográfica. Neste capítulo serão apresentados, apenas, as equações e a lógica de programação utilizada.

- Perda por deformação da ancoragem ($\Delta\sigma_{def,anc}$)

A perda por deformação da ancoragem é a mesma para cada seção e para cada borda. Esta depende apenas do módulo de elasticidade do aço de protensão, do comprimento da pista de protensão e do valor da acomodação da ancoragem fornecido pelo fabricante.

$$\Delta\sigma_{def,anc} = E_p * \frac{\Delta l}{L_{pista}} \quad (Eq.\ 4.5.1)$$

E_p: módulo de elasticidade do aço de protensão, em MPa;

Δl: acomodação da ancoragem fornecida pelo fabricante, em mm;

L_{pista}: comprimento da pista de protensão, em m.

O valor dessa perda é descontado da tensão de protensão inicial e, em seguida, inicia-se o cálculo relaxação da armadura com essa tensão.

- Perda por relaxação da armadura inicial ($\Delta\sigma_{rel,ini}$)

A perda por relaxação da armadura é função da tensão existente no cabo em um determinado intervalo de tempo. A perda inicial é a mesma para cada seção, considerando que no intervalo de tempo até a concretagem a tensão é constante ao longo de toda cordoalha.

Na aba características básicas é possível acessar e alterar as datas em que cada carregamento é inserido. A idade, em dias, em que se considera a perda por relaxação da armadura inicial é o referente a data em que se considera o carregamento de peso próprio da seção simples pré-moldada (PP). Logo, o programa irá calcular essa tensão em função das tensões atuantes consideradas as perdas por deformação da ancoragem e a idade imposta pelo usuário que tem, como valor padrão, a idade de 1 dia.

Outra informação que deve ser inserida pelo usuário são os valores referentes ao Ψ_{1000}, este que pode ser acessado no menu “Editar”. O padrão do programa considera os valores de Ψ_{1000} para cordoalhas com relaxação baixa, que podem ser observados na Tabela 13.

Tabela 13: Tabela padrão com os valores de Ψ_{1000}.

Tensão	Ψ_{1000}
0,5 f_{ptk}	0
0,6 f_{ptk}	1,3
0,7 f_{ptk}	2,5
0,8 f_{ptk}	3,5

Fonte: NBR 6118:2014.

Caso o usuário utilize aço com outro tipo de relaxação, fios ou barras, deve-se acessar o menu “Editar” e alterar essas informações para os desejados. A resistência à ruptura do aço de protensão (f_{ptk}) também deve ser alterado nesse menu. Nos dados da cordoalha o usuário insere uma série de valores relativos ao aço de protensão. O valor f_{ptk} é calculado em função desses dados pela equação:

$$f_{ptk} = f_{ptd} * \gamma_s \quad (Eq. 4.5.2)$$

Definidas essas informações, o módulo de cálculo se inicia pela relação entre a tensão atuante e o f_{ptk}.

$$R = \frac{\sigma_{S_i}}{f_{ptk}} \quad (Eq.4.5.3)$$

Em seguida, se define o valor de Ψ_{1000} para a relação encontrada.

Finalmente, o módulo verifica se o intervalo de idade pode ser considerado no tempo infinito ou não para o cálculo do coeficiente $\Psi(t,t_0)$. Adotou-se o critério de considerar o tempo infinito para idades igual ou superior a 10.000 dias.

Para cada um dos casos, a equação utilizada é a seguinte:

$$\Psi_{(t,t_0)} = \Psi^{1000} * \left(\frac{t - t_0}{41{,}67}\right)^{0{,}15} \quad (Eq.4.5.4)$$

Para uma idade inferior a 10000 dias.

$$\Psi_{(t,t_0)} = 2{,}5 * \Psi^{1000} \quad (Eq.4.5.5)$$

Para o tempo infinito.

Multiplicando-se o coeficiente de relaxação pela tensão na cordoalha, obtém-se a perda por relaxação:

$$\Delta\sigma_{rel} = \Psi_{(t,t_0)} * \sigma_p \quad (Eq.4.5.6)$$

Como dito anteriormente, essa perda é a mesma para as cordoalhas de uma mesma borda, logo seu resultado será o mesmo ao longo de todas as seções.

As perdas por deformação da ancoragem e relaxação inicial são, então, descontadas da tensão inicial inserida pelo usuário e, em seguida, utilizadas para o cálculo da perda por deformação inicial do concreto. Esta que deve variar para cada seção.

- Perda por deformação imediata do concreto ($\Delta\sigma_{def,conc}$)

Finalizando as perdas imediatas calcula-se a perda por deformação imediata do concreto. Seu cálculo depende do momento atuante na seção, motivo pelo qual as perdas variam de seção para seção. Além do momento, é necessário definir, também, a força normal de protensão atuante na seção, essa que também varia em função do comprimento de regularização (L_p) e do isolamento de cordoalhas, caso exista. E, finalmente, depende da excentricidade das cordoalhas, das características geométricas da peça pré-moldada e da relação entre os módulos de elasticidade ($\alpha_{p,i}$) do aço de protensão ($2x10^5$ MPa) e do concreto na idade inicial (E_{ci}).

Este cálculo é feito automaticamente considerando as equações estabelecidas na NBR 6118:2014 para quando não for realizado ensaio para definição do módulo. Caso o usuário tenha obtido o valor do módulo mediante ensaio, pode-se alterar o valor manualmente. O cálculo é feito quando o botão “Calcular” da aba “Características básicas” é ativado. Seu valor é diretamente inserido na caixa de texto “$\alpha_{p,i}$” (relação entre os módulos de elasticidade do aço de protensão e do concreto nas idades iniciais) e “$\alpha_{p,f}$” (relação entre

os módulos de elasticidade do aço de protensão e do concreto quando adquirida a resistência característica) da aba “Perdas de protensão” no sub-menu “Dados do sistema”.

O cálculo do módulo de elasticidade é feito conforme o tipo de concreto e o coeficiente α_E do tipo de jazida do cimento (padrão α_E = 0,9 para jazida de calcário). Este que pode ser alterado no menu “Editar”.

Para f_{ck} entre 20 e 50 MPa:

$$E_{ci} = \alpha_E * 5600 * \sqrt{f_{ck}} \quad (Eq.4.5.7)$$

Para f_{ck} entre 55 e 90 MPa.

$$E_{ci} = 21{,}5 * 10^3 * \alpha_E * \left(\frac{f_{ck}}{10} + 1{,}25\right)^{1/3} \quad (Eq.4.5.8)$$

Para o cálculo do módulo de elasticidade do concreto nos dias iniciais ($E_{ci,j}$), a NBR 6118:2014 recomenda a seguinte equação:

Para f_{ck} entre 20 e 50 MPa:

$$E_{ci,j} = \left[\frac{f_{ck,j}}{f_{ck}}\right]^{0,5} \quad (Eq.4.5.9)$$

Para f_{ck} entre 55 e 90 MPa.

$$E_{ci,j} = \left[\frac{f_{ck,j}}{f_{ck}}\right]^{0,3} \quad (Eq.4.5.10)$$

Com todas essas informações, as fórmulas abaixo calculam a perda por deformação imediata do concreto na borda inferior e superior, respectivamente.

$$\Delta\sigma_{def,conc,inf} = \alpha_{p,i} * \left(\frac{N_{p,i} + N_{p,s}}{A} + \frac{\left(N_{p,i} + N_{p,s}\right) * e_i - M}{I}\right) \quad (Eq.4.5.11)$$

$$\Delta\sigma_{def,conc,sup} = \alpha_{p,i} * \left(\frac{N_{p,i} + N_{p,s}}{A} + \frac{M - \left(N_{p,i} + N_{p,s}\right) * e_s}{I}\right) \quad (Eq.4.5.12)$$

Onde,

$N_{p,i}$ e $N_{p,s}$: força normal de protensão na borda inferior e superior, respectivamente, em kN;

e_i e e_s: excentricidade inferior e superior, respectivamente, em m;

M: momento atuante na seção, em kN.m;

A: área da seção transversal pré-moldada;

I: momento de inércia da seção transversal pré-moldada.

Encerrado o cálculo dessa perda finaliza-se o cálculo das perdas imediatas. Estes resultados são, então, descontados da tensão inicial e inicia-se o cálculo das perdas diferidas.

Os resultados são apresentados em uma caixa de texto de onde o programa retira os valores para calcular as verificações. Caso o usuário deseje, é possível alterar esses valores de forma arbitrária para que as verificações sejam feitas considerando valores de perda diferentes dos calculados pelo programa.

- Perda por retração ($\Delta\sigma_{retração}$)

Iniciando as perdas diferidas pela retração, esta é considerada constante para todas as seções já que seu cálculo independe dos momentos e dos esforços de protensão atuantes. Para o cálculo dessa perda é necessário saber algumas características do concreto, como seu valor de abatimento, da temperatura média e da umidade do ambiente, além do perímetro em contato com o ar e da seção transversal da peça pré-moldada, apenas. O cálculo do perímetro em contato com o ar é feito automaticamente considerando todo o perímetro da seção descontada a parte de cima que, normalmente, está em contato com a laje, parede ou outro elemento. No entanto, o usuário pode, manualmente, alterar esse valor, caso esse não represente a situação descrita.

Por meio dessas informações o módulo inicia calculando o valor de γ, válido para umidades iguais ou inferiores a 90% ($U \leq 90$) e a espessura fictícia (h_{fic}), por meio das equações:

$$\gamma = 1 + \exp(-7{,}8 + 0{,}1 * U) \quad (Eq.4.5.13)$$

$$h_{fic} = \gamma * \frac{2 * A_c}{u_{ar}} \quad (Eq.4.5.14)$$

Em seguida, inicia-se o cálculo do coeficiente dependendo da umidade relativa do ambiente e da consideração do concreto (ε_{1s}), conforme equação apresentada na norma.

$$\varepsilon_{1s} = -8{,}09 + \frac{U}{15} - \frac{U^2}{2284} + \frac{U^3}{133765} - \frac{U^4}{7608150} \quad (Eq.4.5.15)$$

Esse valor de ε_{1s} é válido para abatimentos entre 5 e 9 cm e para umidade inferior a 90%. O módulo verifica, então, se o abatimento está nesse intervalo. Caso seja inferior a ele, multiplica-se ε_{1s} por 0,75 (ou seja, 25% menor) e, caso seja superior, multiplica-se por 1,25 (25% maior).

Em seguida, calcula-se o valor de ε_{2s}, por meio da equação:

$$\varepsilon_{2s} = \frac{33 + 2 * h_{fic}}{20{,}8 + 3 * h_{fic}} \quad (Eq.4.5.16)$$

Por fim, o valor final da retração (ε_∞):

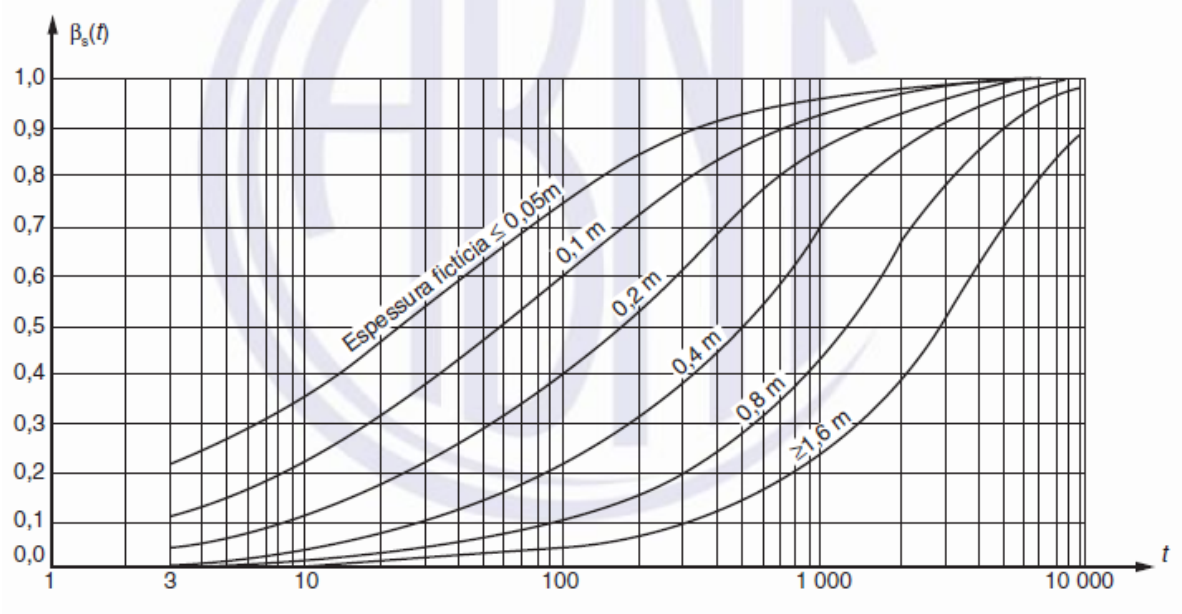
$$\varepsilon_{c\infty} = \varepsilon_{1s} * \varepsilon_{2s} \quad (Eq.4.5.17)$$

Finalmente, calcula-se os coeficientes $\beta_s(t)$ e $\beta_s(t_0)$. Para tanto, calcula-se a idade fictícia por meio da equação:

$$t = \alpha * \sum_i \frac{T_i + 10}{30} * \Delta t_{ef,i} \quad (Eq.4.5.18)$$

A variação de $\beta_s(t)$ é descrita no gráfico abaixo:

Figura 31: Variação de β_s para retração.



Fonte: NBR 6118:2014.

Como se pode observar, este depende da espessura fictícia e do intervalo entre 3 e 10000 dias fictícios. Este mesmo gráfico é descrito pelas equações abaixo:

$$\beta_s(t) = \frac{\left(\frac{t}{100}\right)^3 + A * \left(\frac{t}{100}\right)^2 + B * \left(\frac{t}{100}\right)}{\left(\frac{t}{100}\right)^3 + C * \left(\frac{t}{100}\right)^2 + D * \left(\frac{t}{100}\right) + E} \quad (Eq.\,4.5.19)$$

Onde,

$A = 40$;

$B = 116 * h^3 - 282 * h^2 + 220 * h - 4{,}8$;

$C = 2{,}5 * h^3 - 8{,}8 * h + 40{,}7$;

$D = -75 * h^3 + 585 * h^2 + 496 * h - 6{,}8$;

$E = -169 * h^4 + 88 * h^3 + 584 * h^2 - 39 * h + 0{,}8$.

A idade padrão utilizada para o cálculo dessa perda é definida, da mesma forma que para o cálculo da relaxação, por meio da idade inserida para o peso próprio do elemento pré-moldado (1 dia após o estiramento das cordoalhas). A idade no tempo infinito é imposta para 10000 dias após o estiramento do cabo.

A perda por retração é, então, calculada pela equação:

$$\Delta\sigma_{retração} = \varepsilon_{c\infty} * [\beta_s(t) - \beta_s(t_0)] \quad (Eq.\,4.5.20)$$

Caso o usuário queira fazer alguns testes e verificar um intervalo diferente do padrão do programa, ele pode acessar o menu editar e alterar o valor de T_{inf} no subitem. Desta forma, é possível verificar a retração em um intervalo diferente daquele imposto pelo programa.

- Perda por fluência ($\Delta\sigma_{fluência}$)

Como os momentos atuantes e os esforços de protensão são considerados no seu cálculo, a perda por fluência varia de seção para seção. O usuário deve conhecer, para cada concreto (pré-moldado e capeamento), as condições de temperatura ambiental média (T), a umidade relativa do ar (U), a área da seção transversal (A_c), o perímetro da seção em contato com o ar (u_{ar}) e o abatimento. Além destes, o usuário também deve saber o tipo de endurecimento do cimento (lento, normal ou rápido) e o valor da relação entre os módulos de elasticidade ($\alpha_{p,f}$) entre o aço de protensão ($2x10^5$ MPa) e do concreto quando adquire sua resistência característica (E_{ci}). As considerações de cálculo do programa estão descritas no item sobre perda por deformação imediata do concreto.

No menu “Editar” o usuário pode alterar os valores relativo ao tipo de endurecimento do concreto. Como padrão, assumiu-se os seguintes valores

Tabela 14: Coeficientes de fluência em função da velocidade de endurecimento do concreto.

Endurecimento	Fluência	s	$F_c(t_\infty)$ / $F_c(t_{28})$
Lento	1	0,38	1,433
Normal	2	0,25	1,267
Rápido	3	0,20	1,208

Fonte: NBR 6118:2014.

Definidos esses parâmetros, o programa inicia o cálculo da idade fictícia do concreto, conforme Eq. 4.5.18:

Para o intervalo de 3 e 10000 dias.

Em seguida, realiza-se o cálculo do coeficiente de fluência rápida (φ_a):

$$\beta_1 = EXP * \left\{ s * \left[1 - \left(\frac{28}{t} \right)^{1/2} \right] \right\} \quad (Eq. 4.5.21)$$

$$\varphi_a = \alpha_{c1} * \left(1 - \frac{f_c(t_0)}{f_c(t_\infty)} \right) \quad (Eq. 4.5.22)$$

Onde, α_{c1} depende do f_{ck} do concreto:

- para $f_{ck} < 50$ MPa: $\alpha_{c1} = 0,8$;
- para $f_{ck} \geq 50$ MPa: $\alpha_{c1} = 1,4$;

E o cálculo do coeficiente de deformação lenta irreversível ($\varphi_{f\infty}$):

$$\varphi_{1c} = 4,45 - 0,035 * U \quad (Eq. 4.5.23)$$

$$\varphi_{2c} = \frac{42 + h_{fic}}{20 + h_{fic}} \quad (Eq. 4.5.24)$$

$$\varphi_{f\infty} = \alpha_{c2} * \varphi_{1c} * \varphi_{2c} \quad (Eq. 4.5.25)$$

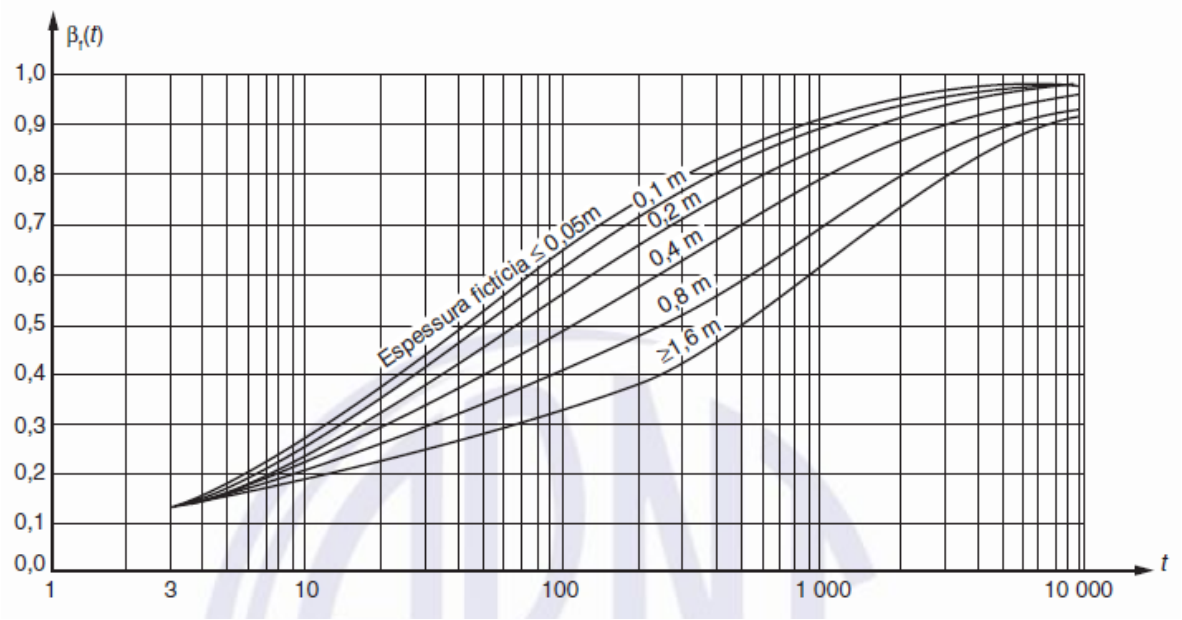
Onde, α_{c2} depende do f_{ck} do concreto:

- para $f_{ck} < 50$ MPa: $\alpha_{c2} = 1{,}0$;

- para $f_{ck} \geq 50$ MPa: $\alpha_{c2} = 0{,}45;1$

Calculam-se os valores de $\beta_s(t)$ e $\beta_s(t_0)$ de maneira semelhante à retração:

Figura 32: Variação de β_s para retração.



Fonte: NBR 6118:2014.

Cujo gráfico pode ser descrito pela equação abaixo:

$$\beta_s(t) = \frac{t^2 + A * t + B}{t^2 + C * t + D} \quad (Eq.\,4.5.26)$$

Onde,

$A = 42 * h^3 - 350 * h^2 + 588 * h + 113$;

$B = 768 * h^3 - 3060 * h^2 + 3234 * h - 23$;

$C = -200 * h^3 + 13 * h^2 + 1090 * h + 183$;

$D = 7579 * h^3 - 31916 * h^2 + 35343 * h + 1931$;

Finalmente, utilizando o coeficiente de deformação lenta reversível ($\varphi_{d\infty}$) e calculando o coeficiente de deformação lenta reversível no intervalo pré-estabelecido (β_d), chega-se a:

$$\varphi_{d_\infty} = 0{,}4$$

$$\beta_d = \frac{t - t_0 + 20}{t - t_0 + 70} \quad (Eq.\,4.5.27)$$

Com todas essas informações, calcula-se o coeficiente de fluência ($\varphi(t, t_0)$):

$$\varphi(t, t_0) = \varphi_a + \varphi_{f_\infty} * [\beta_f(t) - \beta_f(t_0)] + \varphi_{d_\infty} * \beta_d \quad (Eq.\,4.5.28)$$

A primeira parte do cálculo da perda por fluência consiste em definir os coeficientes de fluência. Este é feito para a seção pré-moldada e para a seção moldada no local

separadamente. Em seguida é feita uma média ponderada com as áreas para definir um coeficiente médio.

Finalmente, com os coeficientes de fluência médios, as características geométricas da seção e os esforços atuantes, calcula-se a perda de fluência no tempo infinito para cada borda.

$$\Delta\sigma_{fluência,inf} = \alpha_p * \left\{\left[\frac{N_p}{A} + \left(\frac{M_p - M_{PP}}{I}\right) * e_{p,i}\right] * \varphi_1 - \left(\frac{M_{laje}*\varphi_{laje} + M_{capa}*\varphi_{capa}}{I}\right) * e_{p,i} - \left(\frac{M_{parede}*\varphi_{parede} + M_{revest}*\varphi_{revest} + M_q*\varphi_q*\Psi_2}{I_{comp}}\right) * e_{p,i,comp}\right\} \quad (Eq.\ 4.5.29)$$

$$\Delta\sigma_{fluência,sup} = \alpha_p * \left\{\left[\frac{N_p}{A} + \left(\frac{-M_p + M_{PP}}{I}\right) * e_{p,s}\right] * \varphi_1 - \left(\frac{M_{laje}*\varphi_{laje} + M_{capa}*\varphi_{capa}}{I}\right) * e_{p,s} - \left(\frac{M_{parede}*\varphi_{parede} + M_{revest}*\varphi_{revest} + M_q*\varphi_q*\Psi_2}{I_{comp}}\right) * e_{p,s,comp}\right\} \quad (Eq.\ 4.5.30)$$

Da mesma forma que na retração, é possível realizar testes alterando a data final no menu “Editar”. Dessa forma é possível analisar a fluência em períodos diferentes.

- Perda por relaxação da armadura final ($\Delta\sigma_{rel,fin}$)

As equações de cálculo utilizadas para a perda por relaxação final da armadura são as mesmas que para a relaxação inicial mudando, apenas, os parâmetros de entrada. A diferença é que, agora, a tensão do cabo em cada seção é diferente, o que leva a uma perda por seção. O intervalo de tempo considerado também é diferente. Enquanto que na relaxação inicial o intervalo vai de 0 até a data de entrada do peso próprio da viga (padrão do programa de 1 dia), na relaxação final o tempo vai desde a entrada do peso próprio da viga até o tempo infinito (padrão do programa de 10000 dias), valor esse que pode ser alterado no menu “Editar” para eventuais testes.

Encerrando o cálculo da relaxação final, finalizam-se as perdas diferidas. Somando todas as perdas, o programa passa a ter os parâmetros necessários para realizar as verificações de tensão nas seções da viga. As perdas são somadas isoladamente e inseridas em uma caixa de edição que permite, ao usuário, alterar esses valores.

O programa, em seguida, realiza o cálculo das perdas de protensão considerando a simultaneidade entre elas. Caso o usuário opte por realizar as verificações considerando a simultaneidade, este deve alterar os valores nas caixas de edição.

- Perda total considerando sua simultaneidade ($\Delta\sigma_{progressiva}$)

Sabe-se que os três tipos de perdas diferidas não ocorrem de forma isolada. No entanto, como essa hipótese está a favor da segurança é muito comum considera-las isoladas. O programa apresenta um módulo onde se considera a simultaneidade entre elas refinando, assim, seu cálculo.

$$\Delta\sigma_{progressiva} = \frac{\Delta\sigma_{fluência} + \Delta\sigma_{retração} + \Delta\sigma_{rel,fin} * \chi(t,t_0)}{\chi_p + \chi_c * \alpha_p * \eta * \rho_p} \quad (Eq.\,4.5.31)$$

Onde,

$\chi_{(t,t_0)} = -ln * [1 - \Psi(t,t_0)]$;

$\Psi(t,t_0)$: é o coeficiente de relaxação;

$\chi_p = 1 + \chi_{(t,t_0)}$;

$\chi_c = 1 + 0{,}5 * \varphi_{(t,t_0)}$;

$\eta = 1 + \frac{e^2 * A_c}{I}$;

$\rho_p = \frac{A_p}{A_c}$;

Fazendo isso utilizando os parâmetros inerentes de cada seção, obtêm-se as perdas progressivas. Estas não são inseridas automaticamente para uso do módulo de verificação. O módulo de verificação utiliza as perdas consideradas isoladas. Para tanto, basta que o usuário reescreva os valores das perdas progressivas nos campos correspondentes.

4.6 CÁLCULO DA FLECHA

Como visto anteriormente, o cálculo da flecha em vigas protendidas apresenta uma série de considerações que tornam o problema mais complexo comparado a uma viga de concreto armado.

Primeiramente, é importante relembrar a relevância das fissuras na rigidez do elemento, que provocam um aumento considerável na flecha.

No entanto, uma das características do dimensionamento dos elementos protendidos são as verificações de tensão, esta que, dependendo da classe de agressividade ambiental, garante, caso seja verificado corretamente, que as tensões de tração sejam inferiores à resistência à tração do concreto, evitando o surgimento de fissuras. A própria NBR 6118:2014 afirma que caso a verificação de formação de fissuras seja atendida, pode-se considerar a seção íntegra da viga.

Esta afirmação pode ser tomada como verdadeira para os casos de protensão limitada e protensão completa, estas que contemplam a verificação de formação de fissuras (ELS-F).

No entanto, para o caso da protensão parcial (classe de agressividade ambiental I – CAAI para pré-tração), a norma permite o surgimento de fissuras, desde que controladas e limitadas a um determinado valor. Nesse caso há de ser considerada a redução da inércia na seção.

Feita esta breve introdução, explicam-se, detalhadamente, as considerações feitas no desenvolvimento do módulo de flechas adotado pelo programa.

Seu módulo inicia-se durante, opcionalmente, a etapa dos cálculos das perdas de protensão. Concomitantemente às perdas por seção, o módulo também calcula as perdas que ocorreram até determinada idade do concreto. Idade essa definida para a entrada de cada novo carregamento e para o tempo infinito.

Após definidas, essas são inseridas, em porcentagem, nas caixas de texto da aba "Flecha" podendo, portanto, serem alteradas conforme necessidade do usuário. Este cálculo por idade de carregamento é feito para a armadura ativa inferior e superior da seção do meio do vão. O esforço de protensão a ser utilizado para o cálculo da flecha é obtido pelas perdas que ocorrem no momento da transferência de esforços (ou seja, deformação da ancoragem e relaxação inicial do aço) e as perdas totais. Para as cordoalhas inferiores, utilizam-se as perdas em cada seção. Já para as cordoalhas superiores, utiliza-se a perda que ocorre no comprimento de transferência ($L_{p,s}$).

As perdas para o meio do vão foram usadas para calcular o momento de fissuração (M_r) em cada idade de carregamento, segundo a equação:

- para seção simples

$$M_r = \left(\alpha * f_{ctm} + \frac{N_p}{A}\right) * W_i + M_p \quad (Eq.\,4.6.1)$$

- para seção composta

$$M_r = \alpha * f_{ctm} * W_{i,comp} + \frac{N_p}{A} * W_i + M_p \quad (Eq.\,4.6.2)$$

Onde:

- α: coeficiente que depende do tipo de seção (1,2 para T ou duplo T e 1,5 para retangular). Este valor de α deve ser inserido pelo usuário no menu "Editar";
- f_{ctm}: resistência média à tração do concreto calculado por 0,3 * $f_{ck}^{2/3}$;
- W_i e $W_{i,comp}$: módulo de rigidez à flexão simples e composta, respectivamente;
- A: área da seção pré-moldada;
- N_p: força de protensão das cordoalhas, em kN;
- M_p: momento de protensão na seção;

O momento de fissuração é calculado para cada idade de novo carregamento e inserido em caixas de texto na aba "Flechas". Este é calculado, ainda, durante o módulo das perdas de protensão. Em seguida, ainda no mesmo módulo, calcula-se a inércia média de Branson (I_m), em função das características adquiridas até o momento.

$$I_m = \left(\frac{M_r}{M_{at}}\right)^3 * I_I + \left[1 - \left(\frac{M_r}{M_{at}}\right)^3\right] * I_{II} \leq I_I \quad (Eq.\,4.6.3)$$

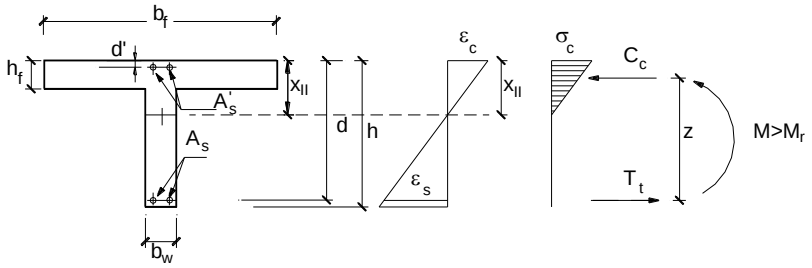
Onde:

- M_r: momento de fissuração;
- M_{at}: momento atuante na seção;

- I_I e I_{II}: momento de inércia da seção no estádio I e II;

Antes de calcular o momento de inércia no estádio II, é necessário definir a posição da linha neutra quando ocorre a fissuração (ou seja, no estádio II). Segundo Ghali e Favre (1986), as deduções se baseiam no princípio que as fibras do topo estejam comprimidas e as da base estejam tracionadas. Por causa disso, considera-se que o estádio II puro, onde todo o concreto da região fissurada é desprezado. De acordo com a NBR 6118:2014, podem-se considerar as tensões na região comprimida como linear.

Figura 33: Distribuição de tensões na seção no estádio II puro.

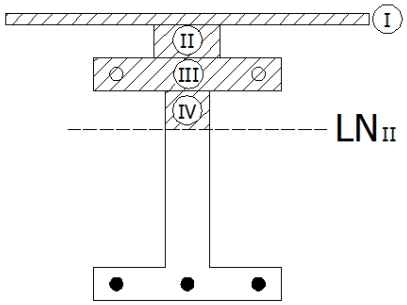


Fonte: Carvalho e Figueiredo Filho (2009).

Portanto, por meio de equações de equilíbrio de forças, podem-se deduzir as equações para cada tipo de seção considerando, inclusive, a variação do módulo de elasticidade do concreto.

Para o programa, imaginou-se a seção mais complexa. Essa que pôde ser aplicada para todas as outras seções.

Figura 34: Seção transversal comprimida no estádio II.



O programa trabalha com o princípio de tentativa, partindo de um valor mínimo para a LN até um valor suficientemente próximo da solução.

Observa-se que, nesse caso, a armadura superior (independentemente de ser passiva ou ativa, assim como a inferior) está na região comprimida do concreto, auxiliando nesse comportamento. Se a linha neutra estivesse acima, significaria que aquela armadura

estaria na região tracionada, somando-se, portanto, à armadura de tração inferior. É necessário, portanto, identificar esses casos.

No programa, a partir do momento em que a linha neutra identifica a seção pré-moldada, considera-se que esta armadura trabalha na região comprimida. E, consequentemente, se a linha neutra estiver na capa a armadura está na região tracionada.

Logo, serão apresentadas as equações para cada caso.

- Se a linha neutra estiver na seção I:

$$\frac{b_1 * x_{II} * E_{cs,1}}{2} = \left(A_p * E_p + A_s * E_s\right) * \left(\frac{d}{x_{II}} - 1\right) + \left(A_p{}' * E_p + A_s{}' * E_s\right) * \left(\frac{d'}{x_{II}} - 1\right) \quad (Eq.4.6.4)$$

Partindo de um valor mínimo e diferente de 0 para a linha neutra (x_{II} = 0,01), verifica-se a igualdade acima. Caso não esteja satisfeita, acrescenta-se o valor 0,01 até que se encontre uma solução. Quando a linha neutra atinge a altura da primeira camada e não encontra uma solução, inicia-se a equação de equilíbrio para o segundo caso.

- Se a linha neutra estiver na seção II:

$$\frac{b_1 * h_1 * E_{cs,1}}{2} * \left(2 - \frac{h_1}{x_{II}}\right) + \frac{b_2 * (x_{II} - h_1) * E_{cs,2}}{2} * \left(1 - \frac{h_1}{x_{II}}\right) = \left(A_p * E_p + A_s * E_s\right) * \left(\frac{d}{x_{II}} - 1\right) + \left(A_p{}' * E_p + A_s{}' * E_s\right) * \left(\frac{d'}{x_{II}} - 1\right) \quad (Eq.4.6.5)$$

- Se a linha neutra estiver na seção III:

$$\frac{b_1 * h_1 * E_{cs,1}}{2} * \left(2 - \frac{h_1}{x_{II}}\right) + \frac{b_2 * h_2 * E_{cs,2}}{2} * \left(2 - \frac{2 * h_1}{x_{II}} - \frac{h_2}{x_{II}}\right) + \frac{b_3 * (x_{II} - h_1 - h_2) * E_{cs,3}}{2} * \left(1 - \frac{h_1 + h_2}{x_{II}}\right) = \left(A_p * E_p + A_s * E_s\right) * \left(\frac{d}{x_{II}} - 1\right) - \left(A_p{}' * E_p + A_s{}' * E_s\right) * \left(\frac{d'}{x_{II}} - 1\right) \quad (Eq.4.6.6)$$

Observa-se, nesse caso, que o programa passa a considerar a região da armadura superior como comprimida, logo, seu sinal deve ser invertido. Além disso, deixando os módulos de elasticidade, é possível considerar valores diferentes para a capa e para a seção pré-moldada.

- Se a linha neutra estiver na seção IV:

$$\frac{b_1 * h_1 * E_{cs,1}}{2} * \left(2 - \frac{h_1}{x_{II}}\right) + \frac{b_2 * h_2 * E_{cs,2}}{2} * \left(2 - \frac{2 * h_1}{x_{II}} - \frac{h_2}{x_{II}}\right) + \frac{b_3 * h_3 * E_{cs,3}}{2} * \left(2 - \frac{2 * h_1}{x_{II}} - \frac{2 * h_2}{x_{II}} - \frac{h_3}{x_{II}}\right) + \frac{b_4 * (x_{II} - h_1 - h_2 - h_3) * E_{cs,4}}{2} * \left(1 - \frac{h_1 + h_2 + h_3}{x_{II}}\right) = \left(A_p * E_p + A_s * E_s\right) * \left(\frac{d}{x_{II}} - 1\right) - \left(A_p{}' * E_p + A_s{}' * E_s\right) * \left(\frac{d'}{x_{II}} - 1\right) \quad (Eq.\,4.6.7)$$

O programa irá procurar uma solução até que a altura da linha neutra coincida com a altura útil. Caso isso aconteça, este emite uma mensagem avisando que não foi possível definir sua posição em função dos parâmetros inseridos.

Determinado a linha neutra, o programa pode, então, calcular o momento de inércia no estádio II em relação a sua linha neutra. Sabendo em qual seção ela passa, o programa utiliza o teorema dos eixos paralelos e soma a inércia de cada retângulo acima da linha neutra. Soma, também, as inércias correspondentes às armaduras. Para estas, deve-se considerar a relação α_E entre os módulos de elasticidade entre o concreto e o aço. Quando dois concretos são considerados, realiza-se uma média ponderada utilizando a área comprimida para definir um módulo de elasticidade médio ($E_{cs,médio}$), que divide o módulo de elasticidade do aço ativo e passivo. Com essas informações, somadas ao momento atuante e ao momento de fissuração, procede-se com o cálculo da inércia de Branson.

Quando o momento atuante supera o momento de fissuração, o programa utiliza a inércia média de Branson para o cálculo da flecha. Encerrado o cálculo da inércia de Branson, este é feito para cada idade de carregamento e inserido nas caixas de texto, permitindo sua edição por parte do usuário.

Assim, encerra-se o módulo de cálculo das perdas de protensão. Note que os valores calculados foram utilizados, apenas, para preencher os campos da aba “Flecha”. Caso o usuário deseje, é possível inserir manualmente esses valores sem, necessariamente, utilizar o módulo das perdas de protensão.

Há que se fazer outra observação. Como não se sabe, ao certo, sob qual carregamento ocorrerá a fissura, é necessário que o programa realize o cálculo do momento de inércia no estádio II puro para a seção simples e, também, para a seção composta. Lembrando que a seção simples é considerada até a entrada do capeamento. Quanto às paredes, o revestimento e o carregamento acidental, estes são resistidos pela seção composta, quando houver.

Iniciando o módulo de flecha, a ideia do programa é calculá-las para cada acréscimo de carregamento. O modelo utilizado foi retirado de Timoshenko (1966) e consiste no

cálculo do momento estático das áreas dos diagramas de momento fletor até a seção desejada (no caso, a central).

Divide-se, portanto, o diagrama decorrente de cada ação atuante na viga: cordoalha superior, cordoalha inferior, carregamento atuante e reação de apoio. Cada uma dessas ações provoca um diagrama de momento fletor e, consequentemente, um momento estático dessa área em relação à extremidade da viga.

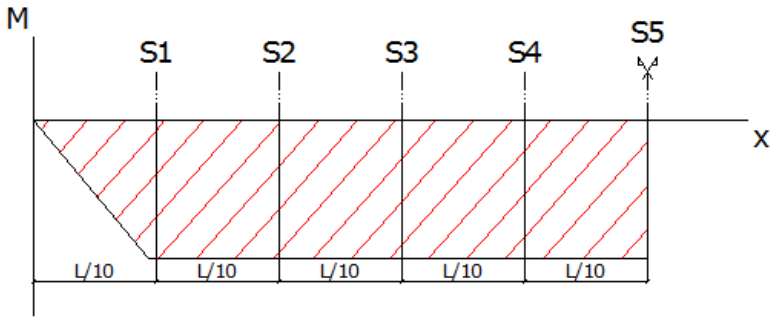
Dessa forma, somando os momentos estáticos e dividindo pela rigidez da viga (considerando a seção bruta ou fissurada e o módulo de rigidez secante do concreto - E_{cs}), obtém-se a flecha naquela seção.

$$a_{imed} = \int_{A}^{B} \frac{1}{EI} * xMdx \quad (Eq. 4.6.8)$$

Logo, para os diagramas de cada ação, foram considerados os seguintes diagramas:

- Diagrama de momento fletor ocasionado pela cordoalha superior:

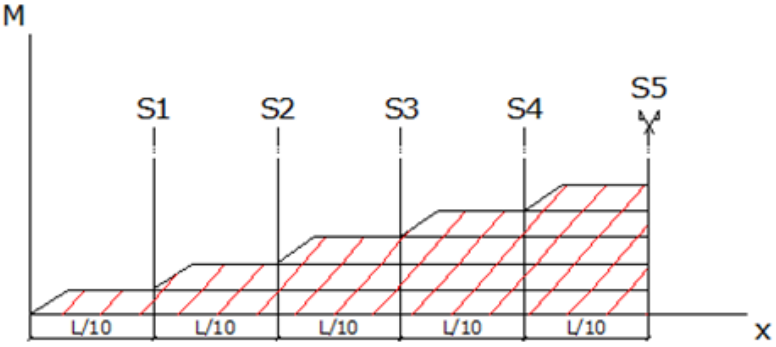
Figura 35: Diagrama de momento fletor causado pelas cordoalhas superiores.



Por este método, é possível considerar o efeito do comprimento de regularização no seu cálculo.

- Diagrama de momento fletor ocasionado pela cordoalha inferior:

Figura 36: Diagrama de momento fletor causado pelas cordoalhas inferiores.

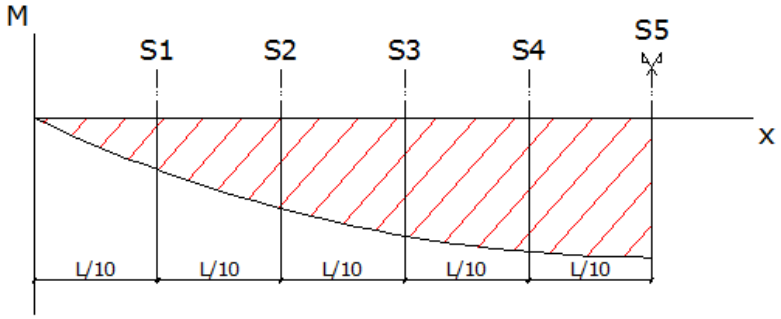


Além do comprimento de regularização, é possível, também, considerar o efeito do isolamento das cordoalhas. O diagrama, nesse caso, assume a disposição demonstrada.

Para a primeira camada, que se estende da extremidade até a seção S5 da viga, considera-se que a tensão atuante nas cordoalhas é igual a tensão inicial descontadas as perdas que ocorrem na seção S1. Para a segunda camada (ou seja, para as cordoalhas que estavam isoladas até a seção S1 e não estão na S2 em diante), considera-se a tensão de protensão descontadas as perdas na seção S2. E assim até a última camada conforme a quantidade de cordoalhas isoladas.

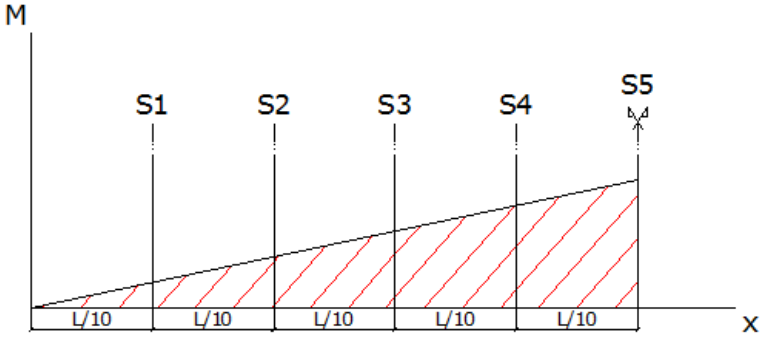
- Diagrama de momento fletor ocasionado pelo carregamento atuante:

Figura 37: Diagrama de momento fletor causado pelo carregamento atuante.



- Diagrama de momento fletor ocasionado pela reação de apoio:

Figura 38: Diagrama de momento fletor ocasionado pela reação de apoio.



Somando os momentos estáticos de cada área (o diagrama acima do eixo caracteriza uma flecha positivo e, abaixo, negativa) e dividindo pela rigidez do elemento, obtém-se a flecha imediata para cada situação. A rigidez é definida pelo produto do módulo de elasticidade secante e da inércia da seção (E_{cs}*I). Dependendo da idade do carregamento, o módulo de elasticidade varia até assumir, de acordo com a norma, um valor constante quando chega aos 28 dias. Logo, para o cálculo da flecha em cada carregamento, essa variação do módulo é considerada.

No botão onde se insere as datas e os coeficientes, na aba “Características básicas”, o usuário pode ter acesso direto a essas informações (inclusive à resistência do concreto em determinada idade). O cálculo pode ser inserido manualmente ou, caso prefira, podem-se utilizar as rotinas do programa para executá-lo.

No cálculo automático, primeiramente, define-se a resistência do concreto ($f_{ck,j}$) para determinada idade (informação que também está sob o controle do usuário).

$$f_{ck,j} = e^{s*\left(1-\sqrt{\frac{28}{t}}\right)} * f_{ck} \quad (Eq.4.6.9)$$

E, finalmente, o cálculo do módulo de elasticidade tangente (E_{ci}) e secante (E_{cs}) é feito conforme especificações da NBR 6118:2014.

- p/ concretos com f_{ck} entre 20 e 50 MPa:

$$E_{ci} = \alpha_E * 5600 * \sqrt{f_{ck}} \quad (Eq.4.6.10)$$

- p/ concretos com f_{ck} entre 55 e 90 MPa:

$$E_{ci} = 21{,}5 * 1000 * \alpha_E * \left(\frac{f_{ck}}{10} + 1{,}25\right)^{\frac{1}{3}} \quad (Eq.4.6.10)$$

Onde,

α_E: coeficiente dependente do tipo de agregado graúdo (tópico 8.2.8 da norma);

α_E = 1,2 para basalto e diabásio;

α_E = 1,0 para granito e gnaisse;

α_E = 0,9 para calcário;

α_E = 0,7 para arenito.

Como padrão do programa, utiliza-se o valor de 1,0.

Já, para o cálculo do módulo de elasticidade secante, multiplica-se o módulo de elasticidade tangente pelo coeficiente α_i.

$$E_{cs} = \alpha_i * E_{ci} \quad (Eq.4.6.11)$$

Onde,

$$\alpha_i = 0{,}8 + 0{,}2 * \frac{f_{ck}}{80} \leq 1{,}0 \quad (Eq.4.6.12)$$

Finalmente, com esses valores, é possível calcular os parâmetros necessários para cada idade, refinando, ainda mais, os cálculos.

Em seguida, é feito o cálculo da flecha considerando o efeito da fluência. Para tanto, utiliza-se o coeficiente de fluência calculado nas perdas de protensão. Estes, durante o cálculo das perdas, são inseridos em caixas de texto. O valor da flecha diferida (a_{flu}) é calculado por:

$$a_{dif} = (1 + \phi) * a_{imed} * \alpha_{Ec} \quad (Eq.4.6.13)$$

Onde,

ϕ: coeficiente de fluência;

α_{Ec}: coeficiente de ajuste que considera a idade do concreto.

Durante o cálculo dos coeficientes de fluência, a norma permite utilizar os valores de módulo de elasticidade tangencial referentes à idade de 28 dias do concreto, mesmo para os carregamentos iniciais, quando o mesmo não adquiriu sua resistência característica. No entanto, quando do cálculo das flechas imediatas, o programa considera a resistência do concreto na idade do carregamento (e, consequentemente, o módulo de elasticidade secante naquela idade). Sendo assim, acredita-se que há certa incompatibilidade durante o cálculo da flecha diferida, já que são consideradas duas resistências diferentes (aos 28 dias, referente ao coeficiente de fluência, e na idade do carregamento inferior aos 28 dias, referente à flecha imediata). Dessa forma, é conveniente multiplicar a flecha diferida por um coeficiente que leve em consideração essa variação da resistência, α_{Ec}, onde:

$$\alpha_{Ec} = \frac{E_{cs,j}}{E_{cs,28}}$$

Para concretos submetidos a ação com idade inferior aos 28 dias, esse coeficiente irá reduzir o valor da flecha diferida.

E, finalmente, a flecha total (a_t) é a somatória das flechas diferidas e imediatas.

$$a_t = a_{imed} + a_{dif} \quad (Eq.\,4.6.14)$$

Estas são somadas de maneira cumulativa de forma a obter o valor final da flecha no tempo infinito.

Finaliza-se, assim, o cálculo das flechas.

4.7 CÁLCULO DA ARMADURA TRANSVERSAL

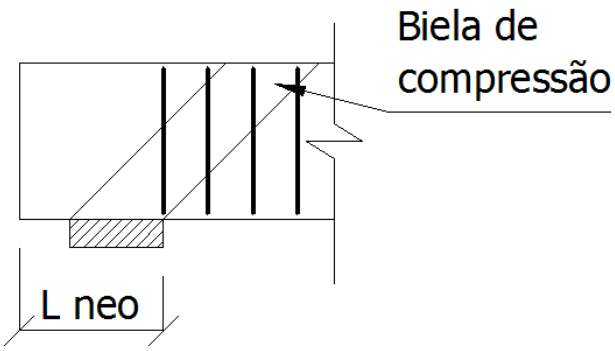
O módulo de cálculo da armadura transversal para resistir aos esforços de cisalhamento pode utilizar os dois modelos permitidos pela norma. O modelo I caracteriza o ângulo da biela em 45° enquanto que o modelo II utiliza um ângulo, pré-definido pelo usuário, entre 30° e 45°.

Dentre as informações a serem inseridas estão o tipo de aço doce (CA-25, CA-50 ou CA-60), a bitola do estribo, o ângulo de inclinação das bielas de compressão (no caso do modelo II), o ângulo de inclinação do estribo e o modelo escolhido para o cálculo.

Outras informações, como o valor do f_{ck} do concreto, características geométricas da seção, altura útil, excentricidades e esforços de momento já foram definidos previamente para o cálculo de outros módulos do programa. Estes são, simplesmente, resgatados desses módulos. Para o caso de elementos com largura variável ao longo da altura, o programa considera o de menor valor para ser usado como b_w.

Em seguida, é importante destacar que a força cortante utilizada no cálculo refere-se àquela atuando na seção após o aparelho de apoio da extremidade da viga.

Figura 39: Seção considerada para o cálculo da armadura transversal.



O valor da protensão utilizado é aquele atuante na seção S1, considerando sua maior proximidade com a seção em análise. Este valor pode ser calculado ou estimado pelo usuário na aba “Perdas de protensão” nas caixas de texto referente às perdas diferidas totais na seção S1 (tanto para armadura inferior como para a superior).

Definidos esses valores, inicia-se o cálculo do espaçamento entre os estribos para dado diâmetro da armadura.

No modelo I, o programa verifica, inicialmente, se há esmagamento da biela de compressão:

$$V_{Rd2,1} = 0{,}27 * \alpha_{V2} * f_{cd} * b_w * d \quad (Eq.4.7.1)$$

Este valor é, então, comparado com a cortante de cálculo.

Em seguida, inicia-se o cálculo da resistência complementar na treliça generalizada. O mecanismo resistente (V_c) é definido pelas seguintes equações:

$$\mathrm{V_{c0}} = 0{,}6 * \mathrm{f_{ctd}} * \mathrm{b_w} * \mathrm{d} \quad (Eq.4.7.2)$$

$$\mathrm{V_c} = \mathrm{V_{c0}} * \left(1 + \frac{\mathrm{M_0}}{\mathrm{M_{Sd}}}\right) \leq 2 * V_{c0} \quad (Eq.4.7.3)$$

Onde:

M_{Sd}: valor do momento fletor de cálculo máximo, no trecho em análise, que pode ser tomado como o de maior valor no semitramo considerado (NBR 6118:2014).

$$\mathrm{f_{ctd}} = \frac{0{,}7 * 0{,}3}{\gamma_c} * \mathrm{f_{ck}}^{2/3} \quad (Eq.4.7.4)$$

$$\mathrm{M_0} = \mathrm{W_i} * \left(\gamma_f * \frac{\mathrm{N}}{\mathrm{A}} + \frac{\mathrm{N} * \mathrm{e}}{\mathrm{W_i}}\right) \quad (Eq.4.7.5)$$

Finalmente, verifica-se a parcela da cortante que deve será resistida pela armadura (V_{sw}).

$$\mathrm{V_{sw}} = \mathrm{V_{sd}} - \mathrm{V_c} \quad (Eq.4.7.6)$$

Caso o valor de V_{sw} seja menor ou igual a zero, o programa vai direto para o cálculo da armadura considerando o espaçamento máximo. Caso contrário, calcula-se o espaçamento (s) e sua respectiva taxa ($\rho_{sw,\alpha}$):

$$s = \frac{A_{sw}}{V_{sw}} * 0{,}9 * d * f_{ywd} \quad (Eq.4.7.7)$$

$$\rho_{sw,\alpha} = \frac{A_{sw}}{b_w * s * sen\alpha} \quad (Eq.4.7.8)$$

Em seguida, verifica-se qual a taxa de armadura correspondente ao espaçamento máximo ($s_{máx}$) estabelecido pela norma para garantir a ductilidade em caso de ruptura:

$$\mathrm{s_{máx}} \leq \begin{Bmatrix} 0{,}6 * \mathrm{d} \leq 300\ \mathrm{mm, se\ V_{sd}} \leq 0{,}67 * \mathrm{V_{Rd2}} \\ 0{,}3 * \mathrm{d} \leq 200\ \mathrm{mm, se\ V_{sd}} > 0{,}67 * \mathrm{V_{Rd2}} \end{Bmatrix} \quad (Eq.4.7.9)$$

Dentre essas duas taxas, o programa adota o maior valor e o compara com a taxa mínima geométrica ($\rho_{swα,mín}$):

$$\rho_{sw\alpha,mín} = 0{,}2 * \frac{f_{ctm}}{f_{ywk}} \quad (Eq.4.7.10)$$

Onde,

$$f_{ctm} = 0{,}3 * {f_{ck}}^{2/3} \quad (Eq.4.7.11)$$

Em caso da taxa mínima geométrica for a de maior valor, calcula-se, então, o espaçamento referente a ela, manipulando algebricamente a equação de cálculo da taxa:

$$\mathrm{s} = \frac{\mathrm{A_{sw}}}{\mathrm{b_w} * \rho_{\mathrm{sw\alpha mín}} * \mathrm{sen}\alpha} \quad (Eq.4.7.12)$$

Finalizando, assim, o dimensionamento para o Modelo I.

Para o modelo II a lógica é a mesma. Calculam-se todas as taxas e verifica qual a maior, utilizando-a para o dimensionamento. A única diferença desse modelo está nas equações e no cálculo do mecanismo resistente do concreto (V_c).

Em relação ao esmagamento da biela de compressão:

$$V_{Rd2,2} = 0{,}54 * \alpha_{V2} * {f_{cd}} * b_w * d * \sin^2\theta * (\cot\alpha + \cot\theta) \quad (Eq.4.7.13)$$

No caso da parcela de cortante absorvida por mecanismos complementares ao da treliça, na flexão simples composta com compressão (V_c), calcula-se de forma similar ao modelo I:

$$\mathrm{V_c} = \mathrm{V_{c1}} * \left(1 + \frac{\mathrm{M_0}}{\mathrm{M_{Sd}}}\right) \quad (Eq.4.7.14)$$

Onde:

$$V_{c1} = V_{c0}\ quando\ V_{sd} \leq V_{c0}$$

$$V_{c1} = 0\ quando\ V_{sd} = V_{Rd2,2}$$

Logo, utilizando da cortante de cálculo, interpola-se com os valores acima e define-se o valor de V_{c1}.

Em seguida, procede-se da mesma maneira que o modelo I. Calcula-se a taxa de armadura para a situação onde o cortante resistido pela armadura transversal é superior a 0. Em seguida, a taxa de armadura para o espaçamento máximo e, finalmente, a taxa de armadura mínima estabelecida por norma. O maior valor dentre essas três taxas é utilizada para definir o espaçamento entre estribos.

Terminando o dimensionamento para a seção mais solicitada, inicia-se o detalhamento para as seções onde a força cortante é tão pequena que a armadura mínima é suficiente para satisfazê-la.

Primeiramente, define-se o espaçamento considerando a taxa mínima estabelecida por norma ($\rho_{sw\alpha,mín}$), cuja equações já foi apresentada. Em seguida, define-se o espaçamento máximo, este que já foi apresentado e, também, está prevista na norma. O menor desses dois valores é, portanto, o espaçamento a ser utilizado nas regiões de armadura mínima.

O próximo passo é determinar a força cortante máxima (V_R) na qual se pode utilizar esse detalhamento. Esta é facilmente calculada pela seguinte equação:

$$V_R = 644 * b_w * d * \left(\rho_{sw,90} * f_{ywd} + 0.1 * {f_{ck}}^{2/3}\right)$$

A distância da extremidade onde essa cortante atua ($L_{Vmáx}$) é definida por:

$$L_{Vmáx} = \frac{L}{2} - \frac{V_R}{p} \quad (Eq.4.7.15)$$

Onde,

p: é o carregamento distribuído.

4.7.1 Armadura transversal na interface da seção composta

A armadura transversal exposta à peça pré-moldada e que solidariza o concreto da capa é calculado conforme as prescrições da NBR 9062:2006. É feito, resumidamente, duas verificações de forma a comparar as tensões de cisalhamento atuantes.

$$\tau_{sd} \leq \tau_{ud} < 0{,}25\, f_{cd} \quad (Eq.4.7.1.1)$$

Onde,

τ_{sd}: tensão de cisalhamento atuante na interfaçe;

τ_{ud}: tensão de cisalhamento resistente;

A tensão de cisalhamento solicitante é obtida da seguinte forma:

$$\tau_{sd} = \frac{F_{Md}}{b * a_v} \quad (Eq.4.7.1.2)$$

Onde,

F_{Md}: é a força horizontal, decorrente do carregamento, atuando na seção;

b: menor largura da interface;

a_v: distância entre a seção de momento máximo e momento nulo (para vigas com carregamento distribuído esse valor é de L / 2, onde L é o vão).

A tensão resistente é calculada pela seguinte equação:

$$\tau_{ud} = \beta_s . \rho . f_{yd} + \beta_c . f_{ctd} \quad (Eq. 4.7.1.3)$$

Onde,

β_s e β_c: coeficientes de rugosidade que dependem da taxa de armadura;

Como se pode observar, os parâmetros de entrada são relativamente simples de serem obtidos. Iniciando pelo cálculo da tensão de cisalhamento solicitante, está pode ser obtida durante o dimensionamento da armadura, onde se obtém o valor da força normal (F_{Md}) resultante do momento de carregamento.

$$F_{Md} = \frac{M_d}{KZ * d} \quad (Eq. 4.7.1.4)$$

Para o cálculo da tensão resistente, é necessário definir os valores de β_s e β_c. Como valores padrão, segue-se a seguinte Tabela 15:

Tabela 15: Coeficientes de rugosidade.

Coeficiente	**Taxa de armadura**	
	ρ≤ 0,20	**ρ≥ 0,50**
β_s	0	0,90
β_c	0,30	0,60

Fonte: NBR 9062:2006.

Esses valores podem ser utilizados, apenas, para superfícies de ligação intencionalmente áspera com rugosidade de 0,50 cm em 3,0 cm. Para superfícies lisas ou naturalmente rugosas, esses coeficientes são obtidos por meio de ensaios. O programa permite sua edição no menu “Editar”, logo, caso o usuário tenha os valores para os outros casos, basta inseri-los nos campos adequados.

Uma premissa adotada para esse módulo se baseia na utilização da mesma bitola do estribo utilizado para resistir aos esforços cortantes do carregamento. E, como resultados, apresenta-se um intervalo de espaçamentos que é solução para a inequação.

O programa, inicialmente, considera um valor máximo de 30 cm de espaçamento. Com esse valor é possível calcular a taxa de armadura e obter os coeficientes de rugosidade para, em seguida, efetuar o cálculo da tensão resistente de cisalhamento. Esse valor é, então, comparado à tensão atuante. Enquanto a tensão resistente for inferior, a rotina é executada novamente descontando 1 cm no espaçamento até alcançar o primeiro valor que satisfaça a condição. Este valor é, então, considerado como o espaçamento

máximo ($s_{máx}$). Se o espaçamento chegar a 0 sem solução, apresenta-se uma mensagem de erro para que o usuário altere os dados.

Obtendo um valor de espaçamento que satisfaça a primeira inequação, inicia-se a segunda etapa, onde se verifica o excesso de estribos. Nesta etapa, o programa inicia com um valor hipotético de 1 cm. Para cada verificação não atendida, acrescenta-se uma unidade ao espaçamento, chegando, no máximo, até 30 cm (a partir do qual o programa emite uma mensagem de erro semelhante à anterior). Se houver um valor que satisfaça a condição, este passa a ser o espaçamento mínimo ($s_{mín}$).

Para finalizar, o programa analisa e compara os valores, que pode resultar em duas situações distintas:

$$s_{máx} \geq s_{mín}$$

Situação onde há solução e pode-se utilizar qualquer espaçamento dentro do intervalo.

$$s_{máx} < s_{mín}$$

Não há solução. O programa apresenta uma mensagem de erro para que o usuário altere os dados.

4.8 VERIFICAÇÃO DA FISSURAÇÃO

Para encerrar a parte de cálculo do programa, é feito o módulo que verifica a fissura na seção central de uma viga pré-tracionada. Conforme descrito anteriormente, esta deve ser realizada quando se considera uma classe de agressividade ambiental fraca de baixo risco para a estrutura (CAA I). Para concreto com pré-tração, não se permite que as fissuras apresentem largura superior a 0,2 mm para a combinação de ações frequentes.

Dentre os parâmetros necessários para o funcionamento do módulo estão: tensões de protensão em serviço, momento de inércia no estádio II puro, momento na combinação frequente.

As tensões de protensão são simplesmente obtidas por meio da tensão inicial (inferior e superior) descontadas as perdas de protensão calculadas (ou inseridas). As perdas consideradas estão apresentadas na aba “Perdas de protensão” nas caixas de texto referente às perdas totais. Apesar de verificar apenas a seção central, é necessário informar os valores em todas as seções, pois, como o programa considera o efeito do comprimento de regularização e do isolamento de cabos, estes podem vir a interferir no cálculo.

Em relação ao momento de inércia no estádio II, este ocorre da mesma forma que explicado no módulo de flechas. No entanto, como se considera apenas a situação em serviço, não há necessidade de se calcular esses parâmetros para a seção simples e composta. É necessário, somente, para a seção final.

Inicia-se pelo cálculo do momento de descompressão (M_0) (momento para o qual a seção volta ao estado inicial antes da protensão).

$$M_0 = \left(\frac{N_p}{A_c} + \frac{N_p * {e_p}^2}{I_I}\right) * \frac{I_I}{e_p} \quad (Eq. 4.8.1)$$

Onde,

N_p: força de protensão (em kN);

A_c: área da seção pré-moldada do concreto;

e_p: excentricidade em relação às cordoalhas inferiores;

I_I: momento de inércia da seção pré-moldada bruta;

Em seguida, calcula-se a tensão atuante (σ_i) no centro da cordoalha decorrente do carregamento. Destaca-se que não é no centro de gravidade da armadura em geral, mas sim de cada barra/cordoalha isoladamente.

$$\sigma_i = \frac{(M_{freq} - M_0)}{I_{x,II}} * y_i * \alpha_E \quad (Eq. 4.8.2)$$

Onde,

M_{freq}: é o momento para a combinação frequente;

I_{comp}: momento de inércia da seção composta;

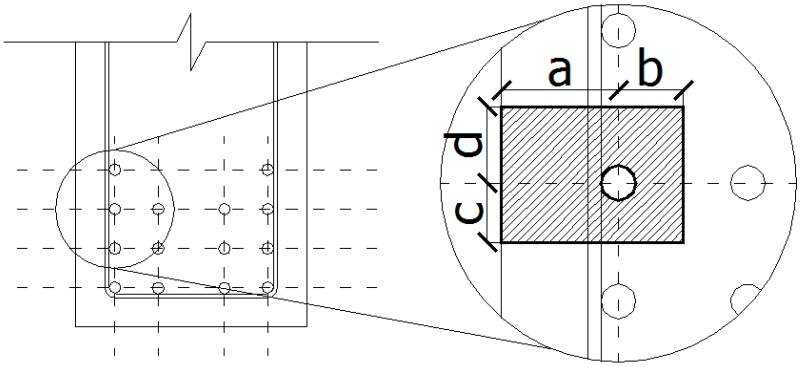
y_i: distância do centro de gravidade da seção composta até o centro da barra/cordoalha;

α_E: relação entre os módulos de elasticidade do aço (ativo e passivo, diferentemente) e o secante do concreto na idade do carregamento em serviço (E_{cs});

O programa também calcula, para cada barra, a área de concreto em seu entorno para definir sua taxa de armadura.

A definição dessa área é feita de forma simplificada. Mas, caso o usuário necessite de uma análise mais precisa, ele pode corrigir manualmente os valores da área envolvente. Primeiramente, como o usuário realiza a entrada da posição da armadura de maneira manual e por camadas, pode ser que o programa, em determinados casos, não utilize a área correta.

Figura 40: Área de concreto envolvente de uma barra para cálculo da fissuração.



A definição da área envolvente é feita dentro de cada camada lançada pelo usuário. Dessa forma, o espaçamento horizontal é obtido a partir das coordenadas dessas cordoalhas e do cobrimento. No entanto, caso o usuário entre com duas camadas na mesma altura (por exemplo, ao posicionar armadura ativa e passiva na mesma camada), o programa não irá reconhecer os tipos diferentes. Em relação à distância vertical, esta é limitada pela altura da última camada, e pela distância da armadura até a face inferior da viga. Então, para uma viga com três camadas, as barras da camada intermediária serão limitadas, na parte superior, pela maior altura dentre as três e, na parte inferior, pela distância do centro de gravidade da barra até a face inferior da viga. O programa não reconhece a camada mais baixa.

Esta forma de se considerar a área envolvente está a favor da segurança, considerando uma abertura de fissura maior do que efetivamente é. No entanto, caso esta apresente problemas, há um módulo no programa que permite ao usuário corrigir as informações utilizadas.

Se a abertura de fissuras for superior ao limite estabelecido (que pode ser alterado no menu “Editar”), o programa aponta todas as barra que apresentaram problemas, identificando-as por suas coordenadas, permitindo que o usuário altere os valores da área envolvente para números mais coerentes.

O cálculo da abertura de fissuras (w) é feito pelas equações descritas na NBR 6118:2014.

$$w \text{ é } o\ menor\ valor\ entre \begin{cases} \dfrac{\phi_i}{12{,}5 * \eta_i} * \dfrac{\sigma_i}{E_{si}} * \dfrac{3 * \sigma_i}{f_{ct,m}} \\ \dfrac{\phi_i}{12{,}5 * \eta_i} * \dfrac{\sigma_i}{E_{si}} * \left(\dfrac{4}{\rho_{ri}} + 45\right) \end{cases} \quad (Eq.\ 4.8.3)$$

Onde,

ϕ_i: diâmetro da barra, em mm;

η_i: coeficiente de conformidade superficial da armadura passiva e ativa considerada;

σ_i: tensão atuante no centro de gravidade da barra verificada, decorrente do carregamento, em MPa;

E_{si}: módulo de elasticidade do aço de protensão ou o aço doce;

$f_{ct,m}$: resistência média à tração do concreto;

ρ_{ri}: taxa de armadura na área de concreto envolvente.

$$\rho_{ri} = \frac{A_{aço}}{A_{cr}} \quad (Eq.\ 4.8.4)$$

Onde,

$$A_{cr} = (a + b) * (c + d)$$

5 MANUAL DE UTILIZAÇÃO DO PROGRAMA

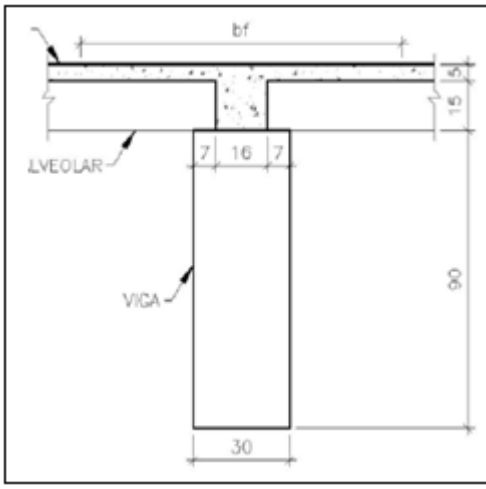
O intuito deste manual é demonstrar ao usuário como efetuar a entrada de dados no programa, além de apresentar sua interface gráfica.

Para tornar o manual mais didático, optou-se por utilizar um exemplo apresentado em Inforsato (2009). Este exemplo se constitui de uma viga com seção retangular pré-moldada e capeamento com f_{ck} diferente.

Não é intuito deste manual comparar os resultados. Esse tipo de validação já foi efetuado no respectivo capítulo. A intenção é apenas utilizar seus dados básicos (carregamento, tipo de concreto, etc.) e realizar o dimensionamento e as verificações conforme feito com qualquer outro caso.

Apresentam-se, portanto, essas informações:

Tabela 16: Tabela com os dados do exemplo.



Carregamento	Intensidade (kN/m)
Peso próprio	6,75
Laje	16,2
Capa	9
Alvenaria	5,94
Revestimento	5,76
Acidental	21,6

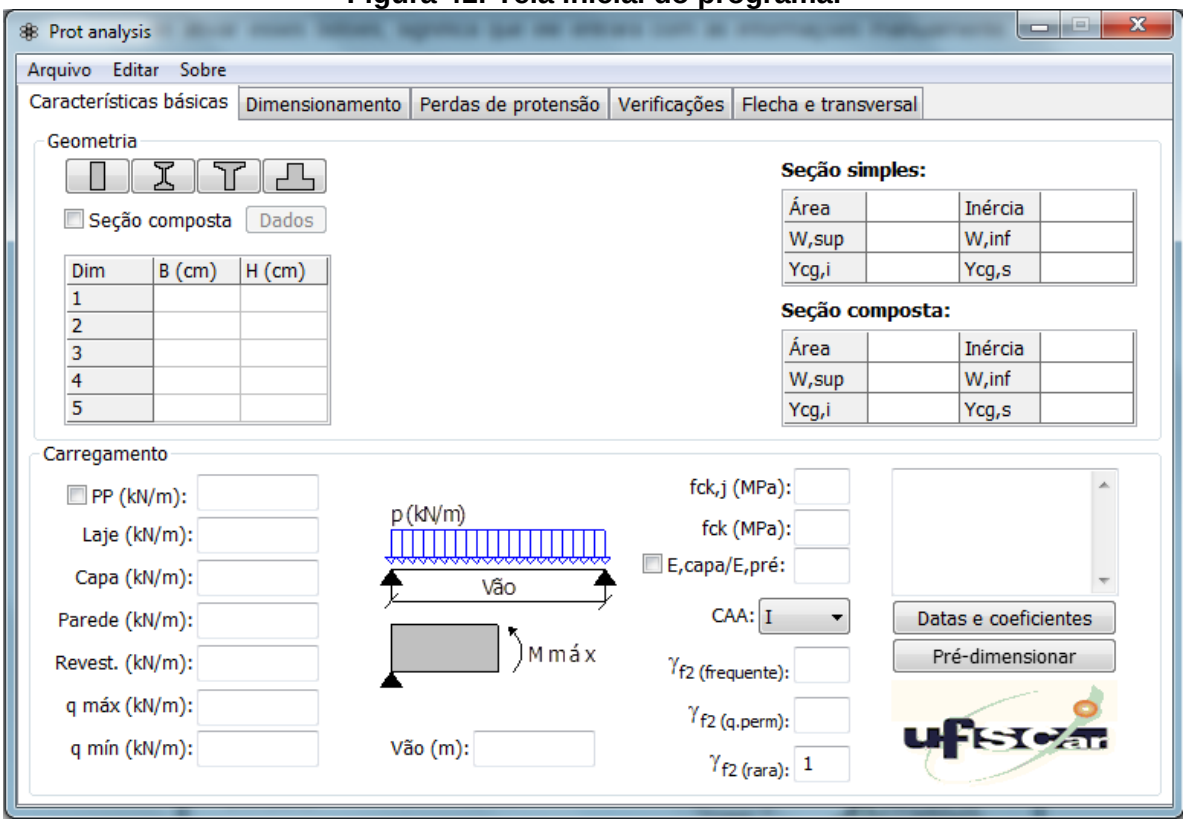
Fonte: Inforsato, 2009.

Quando o programa é executado o usuário se defronta com sua tela inicial. É nela onde se entrará com os dados básicos de qualquer projeto: carregamentos atuantes, dimensões da seção transversal, características físicas dos materiais, vão e coeficientes para as combinações de ação.

É importante destacar algumas utilidades que foram inseridas de forma a facilitar a navegação do programa pelo usuário. Por exemplo, sempre que estiver em dúvida sobre qual dado deve ser inserido, basta posicionar o ponteiro no campo sobre o qual surgirá uma breve explicação. Há, também, algumas informações onde é possível marcar ou desmarcar a opção, como se pode ver na entrada dos dados de carregamento de peso próprio (PP) e relação entre os módulos de elasticidade (E,capa / E,pré). Quando o usuário tiver a opção de ativar esses botões, significa que ele entrará com as informações manualmente. Caso estes estejam desmarcados, o próprio programa se encarrega de calcular seu preenchimento segundo seus critérios.

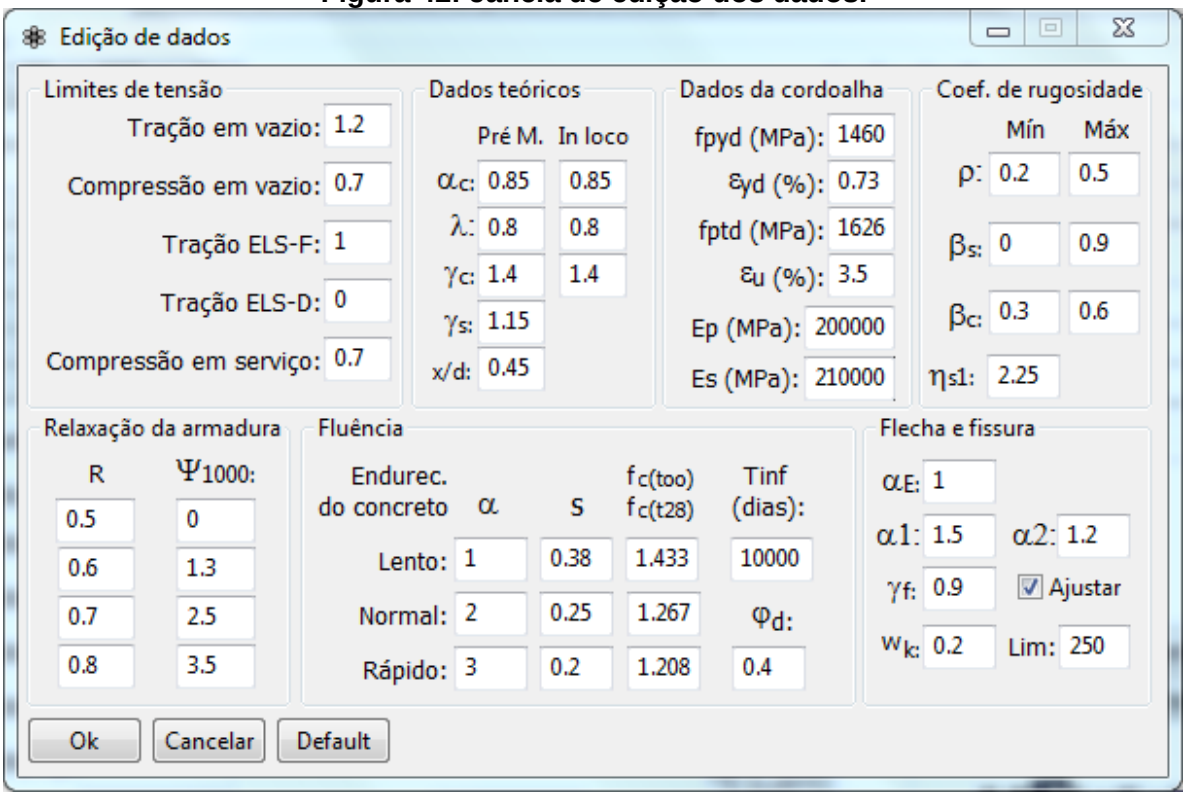
Abaixo são apresentadas as telas e o modelo analisado.

Figura 41: Tela inicial do programa.



Nesta tela, o usuário deve, inicialmente, definir os critérios de cálculo e coeficientes que serão utilizados pelo programa, tais como os limites de tensão nas verificações, os coeficientes teóricos dos materiais, os majoradores de carga e redutores de resistência, dentre outros. Para tanto, deve-se acessar o menu “Editar”. Em seguida, uma janela será aberta para que o usuário possa alterar as informações desejadas. Esta janela é apresentada na Figura 42.

Figura 42: Janela de edição dos dados.



O programa adota alguns valores como padrão. Estes podem ser restabelecidos quando executado o botão “Default” nessa mesma janela. Os valores padrão foram definidos considerando as situações mais comumentemente utilizadas em projetos ou adotados pela norma. Os grupos referentes aos limites de tensão, por exemplo, são fixados pela norma brasileira. Esses valores, no entanto, não refletem os limites utilizados por outras normas, podendo, portanto, serem alterados.

Em relação aos “Dados teóricos”, os valores padrão refletem a utilização do concreto com resistência característica igual ou inferior a 50 MPa. Logo, caso utilize um concreto com resistência superior, deve-se alterar esses valores conforme o estabelecido na norma.

O mesmo procede para os “Dados da cordoalha”. Estes são utilizados para definir o diagrama simplificado. Caso utilize um aço com outras características, estes devem ser inseridos em seus respectivos campos.

Quanto aos “Coef. de rugosidade”, “Relaxação da armadura”, “Fluência” e “Flecha e fissura”, o usuário deve ter conhecimento de cada um desses coeficientes.

Os dados do “Coef. de rugosidade” se referem ao cálculo da armadura transversal na interface entre os dois concretos. Seus valores podem ser utilizados para o caso de superfícies de ligação intencionalmente áspera com rugosidade de 0,50 cm em 3,0 cm. Para qualquer outra informação, esses valores devem ser alterados manualmente. O “η_{s1}” é o

coeficiente de conformação superficial da armadura passiva com valor referente às barras nervuradas de alta aderência (CA-50).

Para a “Relaxação da armadura”, os valores se referem para as cordoalhas com relaxação baixa.

Os valores da “Fluência” também são fixados segundo a NBR 6118:2014. Logo, esses dados, a princípio, não necessitam de alteração. O programa adota o valor de 10000 dias como idade considerando o tempo infinito, logo, nos cálculos onde a idade é um parâmetro importante, estes têm como referência a idade de 10000 dias, com a possibilidade de edição.

Finalmente, os dados de “Flecha e fissura” se referem aos coeficientes e parâmetros utilizados para o cálculo desses.

O coeficiente α_E é utilizado para o cálculo do módulo de elasticidade tangencial do concreto e, o valor padrão, se refere ao tipo de agregado utilizado na sua composição. No caso, granito e gnaisse.

Os coeficientes α_1 e α_2 são os coeficientes de forma da seção, utilizado no cálculo do momento de fissuração para a seção inicial e final, respectivamente. A norma estabelece o valor de 1,5 para seção retangular e 1,2 para seção "T". Logo, o usuário deve inserir essas informações sabendo qual tipo de seção em cada situação.

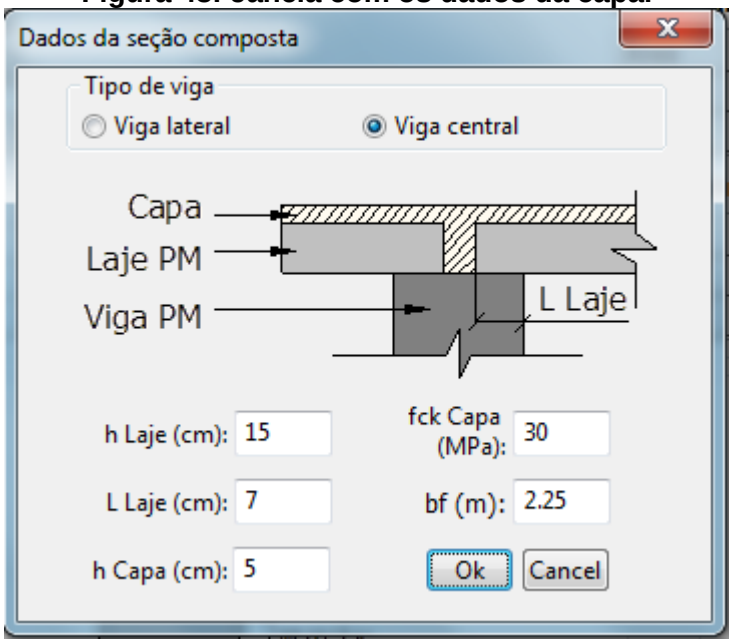
O coeficiente γ_f é utilizado no cálculo do momento de descompressão. Como a protensão é favorável a este momento fletor, a norma define um valor para reduzir sua influência como medida de segurança.

O botão “Ajustar” permite ao usuário considerar, caso ache conveniente, um coeficiente de ajuste para o cálculo da flecha. Como explicado no capítulo sobre as considerações teóricas, este coeficiente visa compensar a variação do módulo de elasticidade utilizado no cálculo das flechas imediatas quando a idade do concreto era inferior a 28 dias. Caso o usuário desmarque essa opção, o programa não irá considerar este ajuste.

Os valores “w_k” e “Lim” representam os limites normativos para a abertura de fissura e a flecha, respectivamente.

Voltando para a tela inicial, o usuário deve, primeiramente, definir qual o tipo de seção a ser utilizada. Pode-se considerar seção retangular simples ou com abas, “I” ou “T”. Para cada uma, pode-se considerar o capeamento.

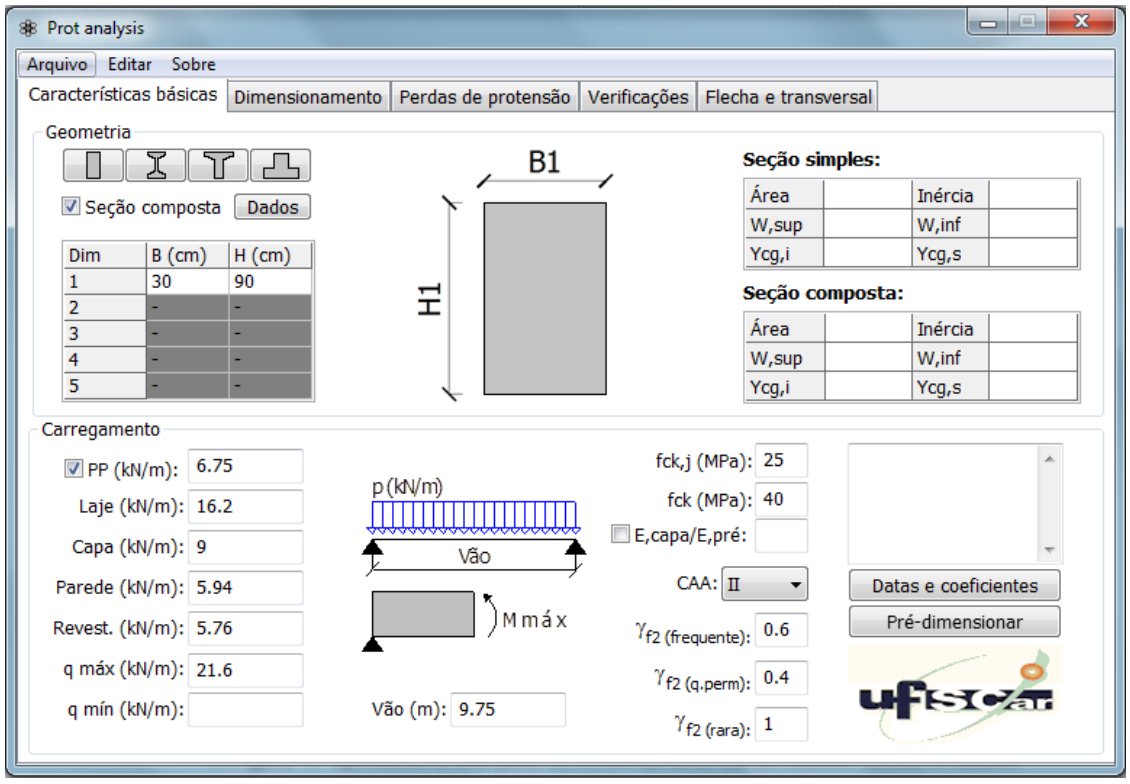
Figura 43: Janela com os dados da capa.



Para inserir os dados da capa, o usuário deve informar a altura da laje, o quanto que esta se apoia na viga, a espessura da capa, seu f_{ck} e a largura b_f a ser considerada no cálculo. Deve informar, também, se é uma viga lateral ou central.

Completando seu preenchimento, continua-se a inserção dos dados na tela inicial com os valores de carregamentos, vão, características do concreto, classe de agressividade ambiental e coeficientes de combinação de ações.

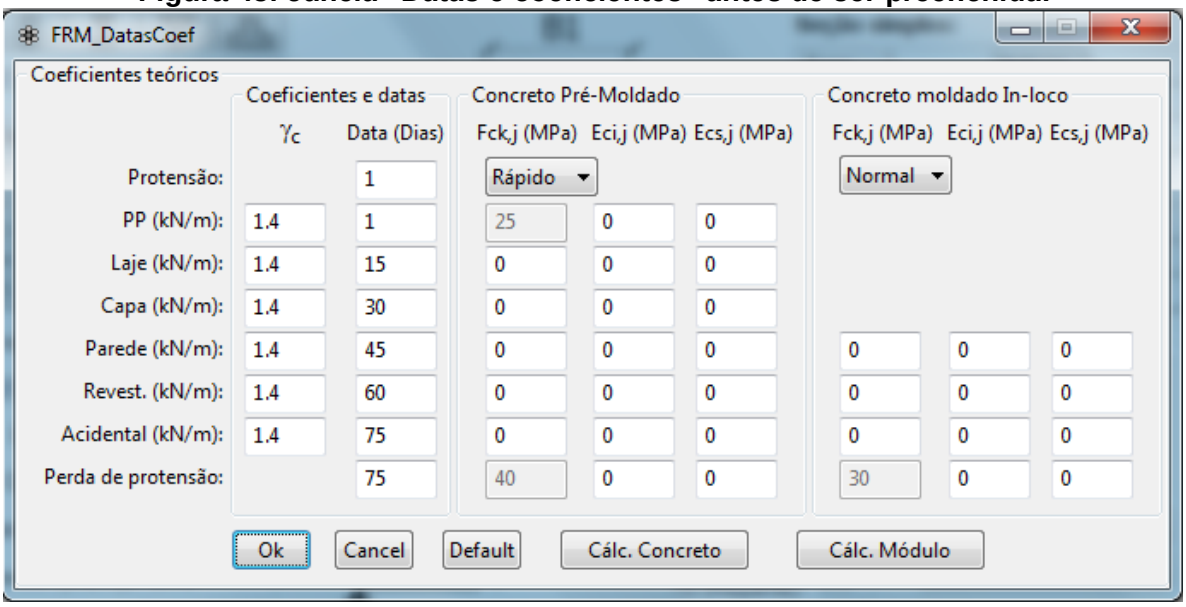
Figura 44: Aba "Características básicas" preenchidas



Observa-se que caso o usuário não queira inserir algum carregamento, é suficiente deixar a caixa de entrada em branco, como feito no campo para acidental mínima (“q mín”). Não é necessário inserir o valor “0”.

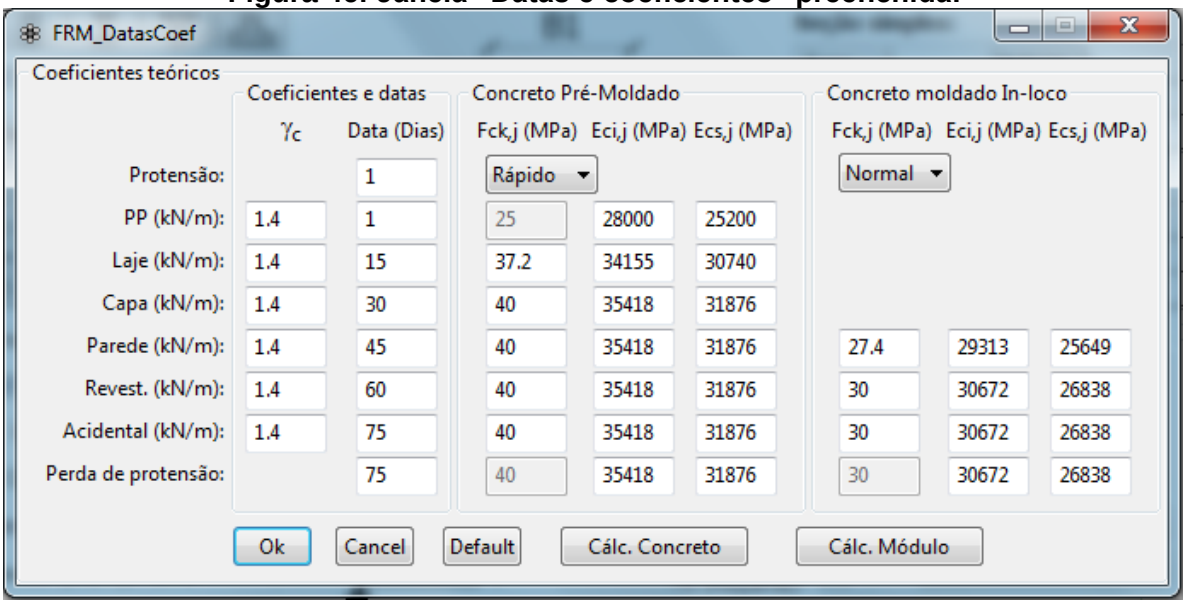
Finalizando essa tela, o usuário deve acessar o botão “Datas e coeficientes” para que sejam inseridas as últimas informações básicas.

Figura 45: Janela “Datas e coeficientes” antes de ser preenchida.



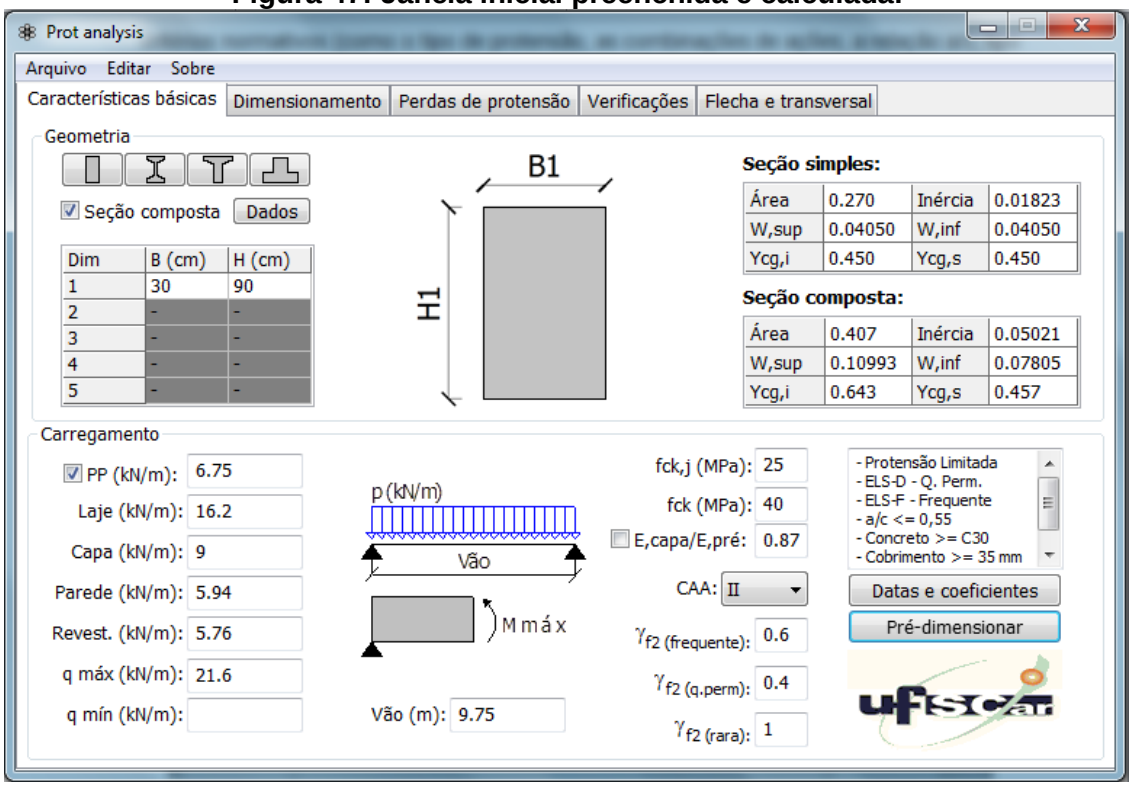
A tela se apresenta conforme a Figura 45, com os valores de majoração de cargas e idades padrão. Em seguida, devem ser inseridos os valores do módulo de elasticidade tangencial (E_{ci}) e secante (E_{cs}). Esses podem ser calculados automaticamente. O botão “Cálc. Concreto” calcula a resistência característica do concreto em função das idades. O botão “Cálc. Módulo” calcula os módulos de elasticidade (tangencial e secante) em função dessas idades, tanto para o concreto pré-fabricado como para o capeamento. Também deve ser informada a velocidade de endurecimento de cada concreto.

Figura 46: Janela “Datas e coeficientes” preenchida.



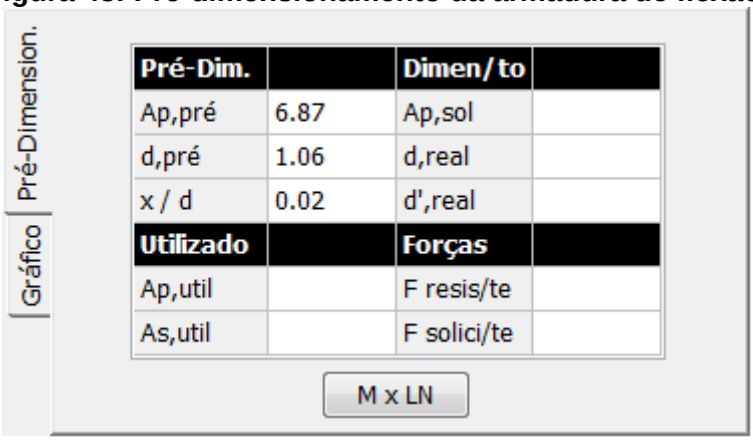
Finalmente, concluindo a inserção dos dados, o usuário pode dar continuidade na análise da viga. Ao clicar no botão “Pré-dimensionar”, o programa realiza o cálculo das características geométricas da seção transversal simples e composta, informa alguns critérios normativos (como o tipo de protensão, as combinações de ações, a relação a/c, tipo de concreto recomendado e o cobrimento mínimo) e o pré-dimensionamento da armadura ativa.

Figura 47: Janela inicial preenchida e calculada.



A Figura 48 apresenta a tela para dimensionamento da armadura. Nesta, o usuário deve inserir a armadura como desejar. O pré-dimensionamento auxilia na quantidade de armadura ativa a ser inserida. Logo, pode-se observar que, para este caso, o usuário precisa de 6,87 cm² de armadura ativa para satisfazer o estado limite último.

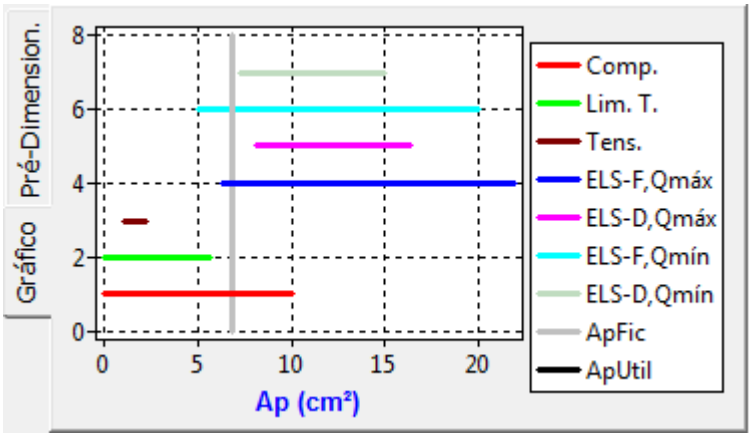
Figura 48: Pré-dimensionamento da armadura de flexão.



Clicando na aba “Gráfico”, pode-se analisar o gráfico com os intervalos de “A_p” para se guiar no dimensionamento da armadura. Por meio deste, observa-se que o limite de tração em vazio apresenta problema para a borda superior. Logo, conclui-se que será necessário utilizar armadura ativa na borda superior.

Após inseridas as armaduras na seção, o usuário deve informar a tensão inicial nas cordoalhas, assim como as perdas de protensão imediatas e totais para cada extremidade. Esses valores são, inicialmente, estimados.

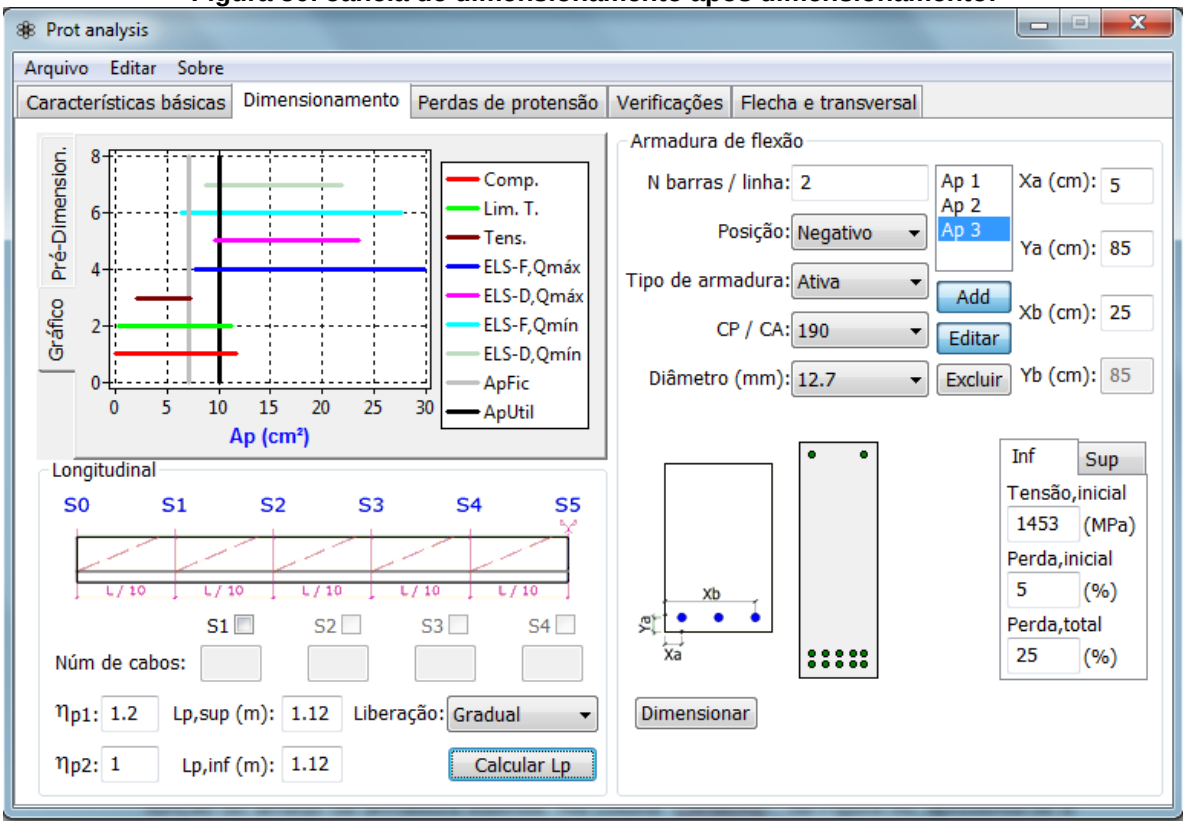
Figura 49: Gráfico com os intervalos de armadura de acordo com o pré-dimensionamento.



O arranjo final pode ser observado na Figura 50. Deve-se informar, também, os comprimentos de regularização. Estes podem ser calculados automaticamente após informar o tipo de liberação da protensão e os coeficientes de aderência das cordoalhas.

Finalmente, o usuário pode definir o isolamento das cordoalhas.

Figura 50: Janela de dimensionamento após dimensionamento.



Ao clicar no botão “Dimensionar”, o programa informa os valores de cálculo em função do arranjo da armadura inserida. Na coluna “Dimen/to”, da Figura 51, apresenta-se a área de armadura ativa necessária para satisfazer o estado limite último em função do “d,real”, “d’,real” e da tensão de protensão final. Pelo gráfico, pode-se observar que a seção central da viga está verificada para as condições em serviço e para a ruptura (Linha preta). Resta saber se as perdas de protensão estimadas estão coerentes.

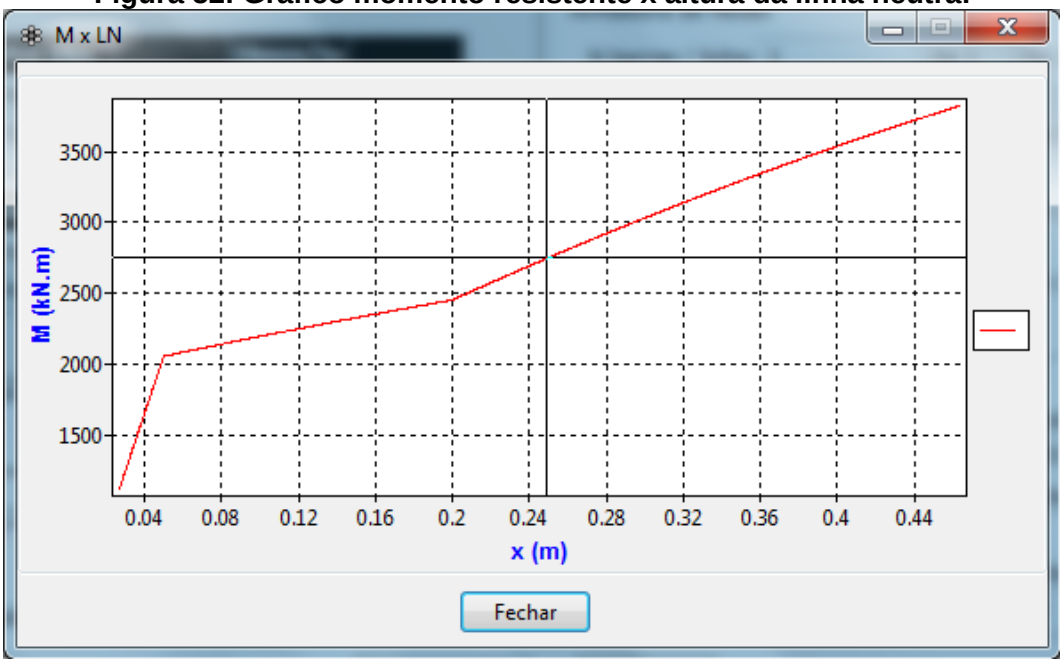
A coluna “Utilizado”, da Figura 51, mostra a quantidade de armadura ativa e passiva utilizada na borda inferior e superior.

E, finalmente, na coluna “Forças”, apresenta a força resultante (F resis/te) da armadura de flexão e a força solicitante (F solici/te) necessária. Obviamente, para garantir que não vai ruir, a força resistente da armadura deve ser igual ou superior à força solicitante (“F resis/te” $\geq$ “F solici/te”, da coluna “Forças” da Figura 51).

Figura 51: Resultados do dimensionamento.

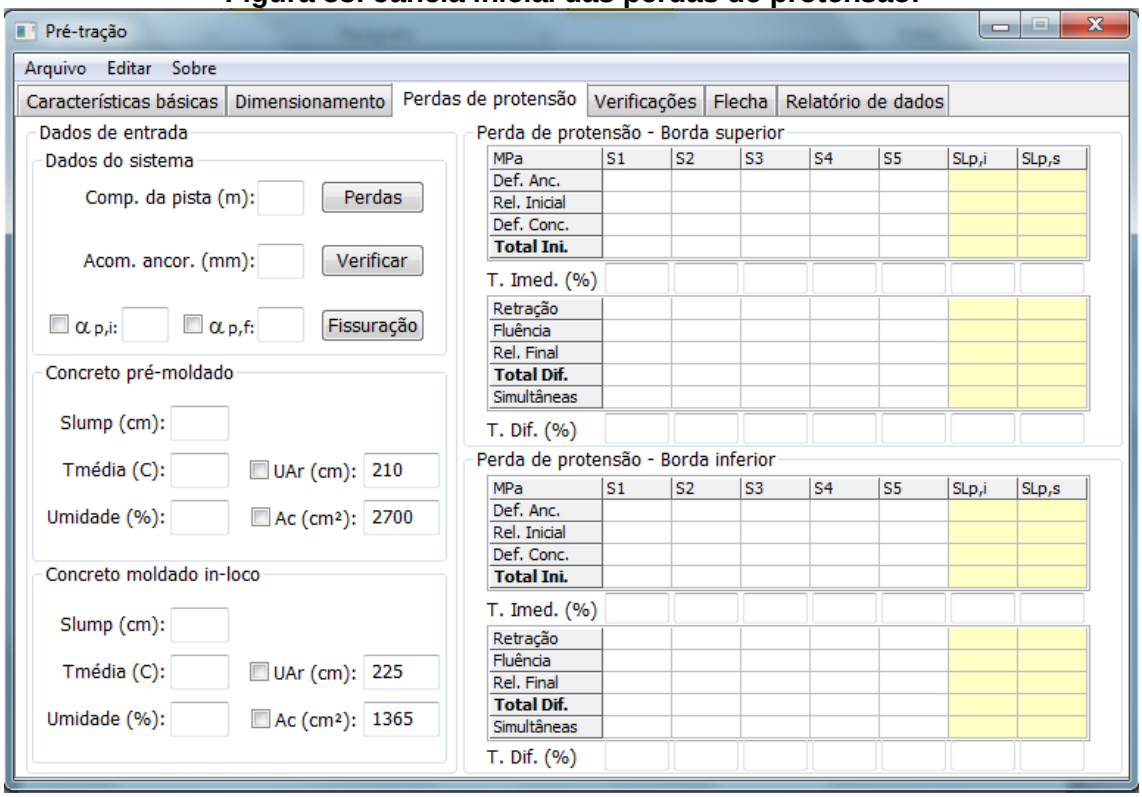
Ao clicar no botão “M x LN”, o programa apresenta um gráfico de momento resistido em função da posição da linha neutra. Dessa forma pode-se observar a variação da capacidade resistente da seção em função da variação de seção.

Figura 52: Gráfico momento resistente x altura da linha neutra.



Após efetuar o dimensionamento, recomenda-se verificar se as perdas de protensão estimadas estão coerentes e próximas das calculadas. Na aba “Perdas de protensão”, o usuário se defronta com a tela conforme Figura 53.

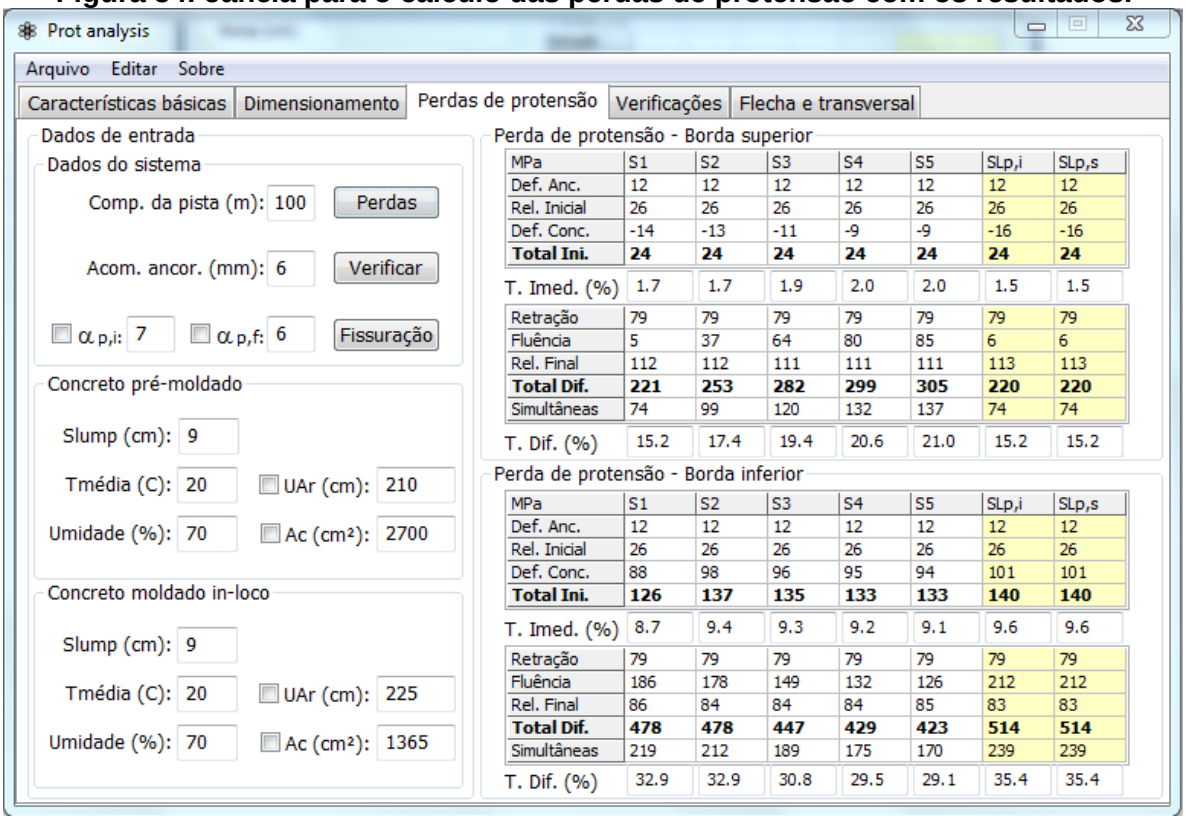
Figura 53: Janela inicial das perdas de protensão.



Para seu cálculo, é necessário inserir dados como o comprimento da pista de protensão, valor da acomodação da ancoragem fornecido pelo fabricante, valor do abatimento do tronco de cone, temperatura média e umidade relativa para os concretos da peça pré-moldada e do capeamento. Os dados do perímetro em contato com o ar e da área da seção são calculados pelo programa conforme critério estabelecido, com opção de entrada manual desses valores.

Caso o usuário não deseje calcular ou não disponha dessas informações, é possível estimar e inserir manualmente os valores de perda para cada seção nos campos referenciados. É importante que o usuário entre com esses valores para que as verificações de tensão, de fissuração e de flecha sejam corretamente calculadas.

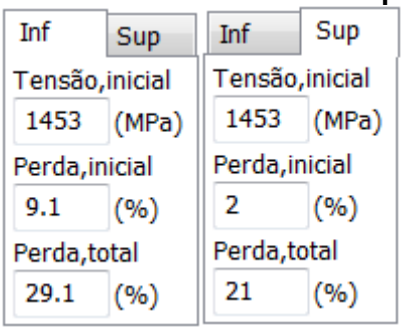
Figura 54: Janela para o cálculo das perdas de protensão com os resultados.



Depois de preenchidos os campos e com as perdas devidamente calculadas, pode-se alterar os valores utilizados no dimensionamento e ver se estão coerentes.

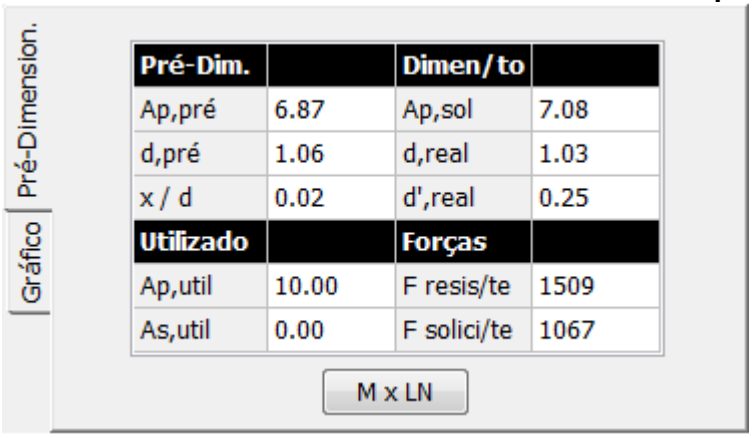
Para a seção do meio do vão, as perdas inferiores imediatas e totais foram de, respectivamente, 9,1% e 29,1% . Nas cordoalhas superiores esses valores foram de 2% e 21%.

Figura 55: Entrada dos valores calculados de perda de protensão.



Como se pode observar na Figura 56, mesmo com o aumento da perda a viga ainda está segura contra a ruptura (“F resis/te” > “F solici/te”, da coluna “Forças” da Figura 56).

Figura 56: Dimensionamento sob os novos valores de perda.

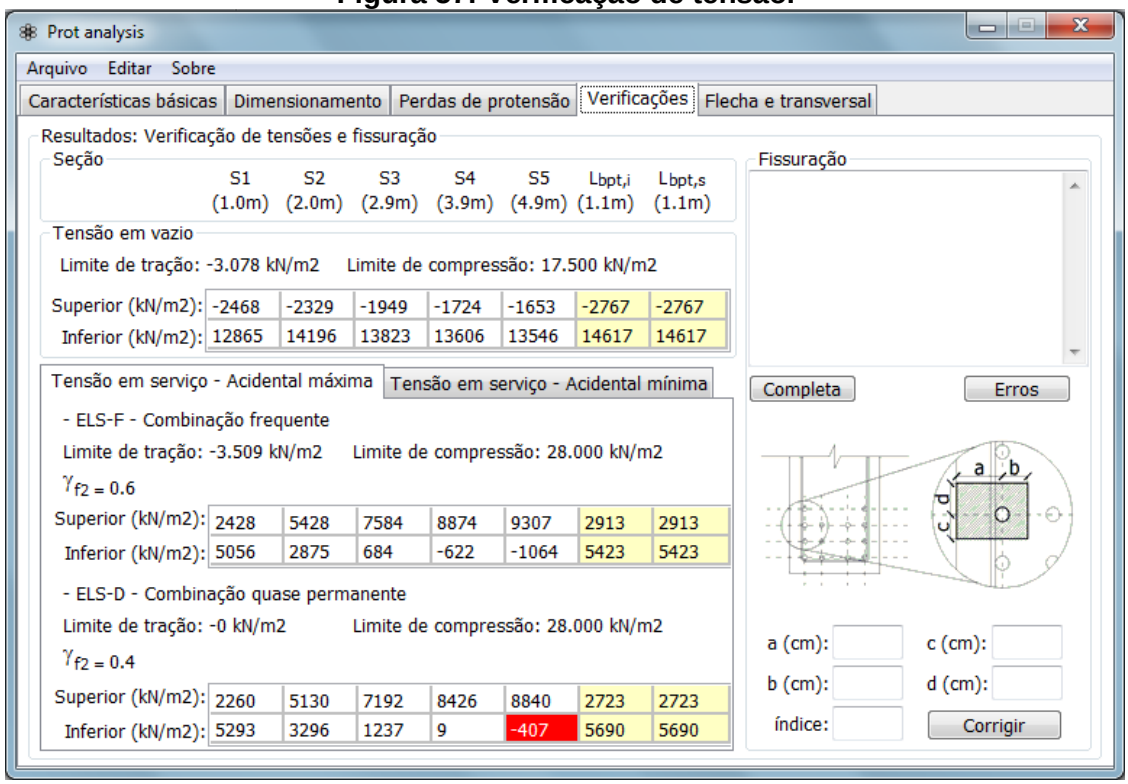


Caso a armadura fosse insuficiente com os valores calculados de perda, o usuário deveria efetuar um novo dimensionamento e calcular as perdas de protensão novamente, até que os valores fossem coerentes.

Em seguida é feita a verificação de tensão para cada décimo de vão e para as seções do comprimento de regularização (inferior e superior).

Na Figura 57, é possível verificar que a viga apresentou problema com a verificação da descompressão para a combinação quase permanente. É necessário, portanto, aumentar a protensão na borda inferior na seção central.

Figura 57: Verificação de tensão.



Foram consideradas as perdas totais isoladas, ou seja, sem considerar a simultaneidade que ocorre entre elas. Logo, como o problema de tensão é relativamente pequeno, reduzir a perda de protensão no meio do vão talvez seja o suficiente para resolver este problema. Portanto, considerando a simultaneidade das perdas diferidas, estas passam para o seguinte valor:

Cordoalha inferior: $(\Delta_{pi,imed} + \Delta_{pi,progr}) / \sigma_{p,i} = (133 + 170) / 1453 = 20,9\%$

Cordoalha superior: $(\Delta_{ps,imed} + \Delta_{ps,progr}) / \sigma_{p,s} = (24 + 137) / 1453 = 11,1\%$

Onde:

$\Delta_{pi,imed}$ / $\Delta_{ps,imed}$: Perda de protensão total imediata na borda inferior e superior, respectivamente;

$\Delta_{pi,progr}$ / $\Delta_{ps,progr}$: Perda de protensão total progressiva na borda inferior e superior, respectivamente;

Alterando esses valores manualmente na aba “Perdas de protensão” e verificando novamente as tensões, conclui-se que foi o suficiente para satisfazê-las.

Figura 58: Verificação de tensões para acidental máxima.

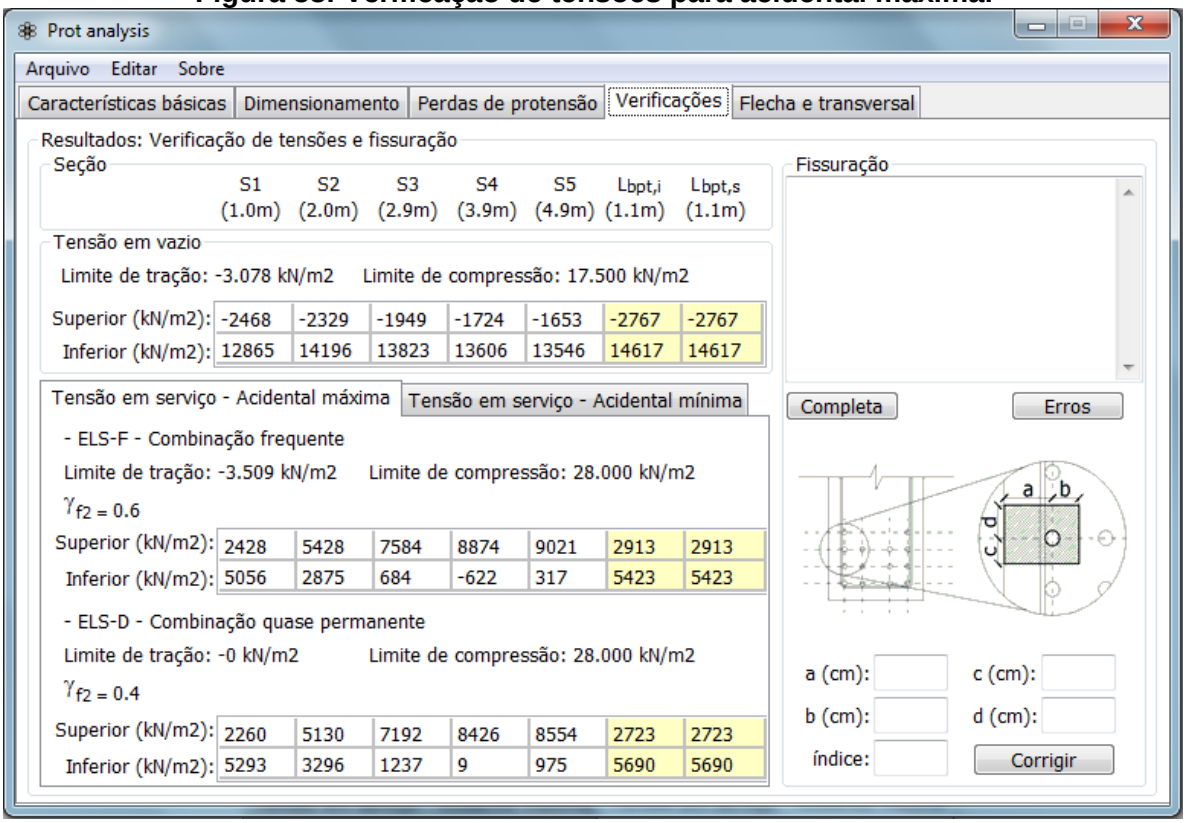
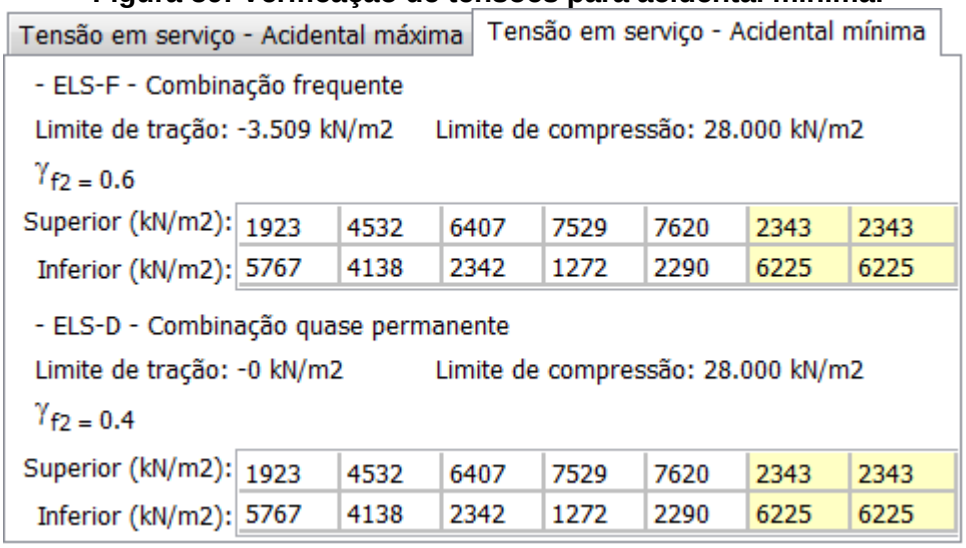
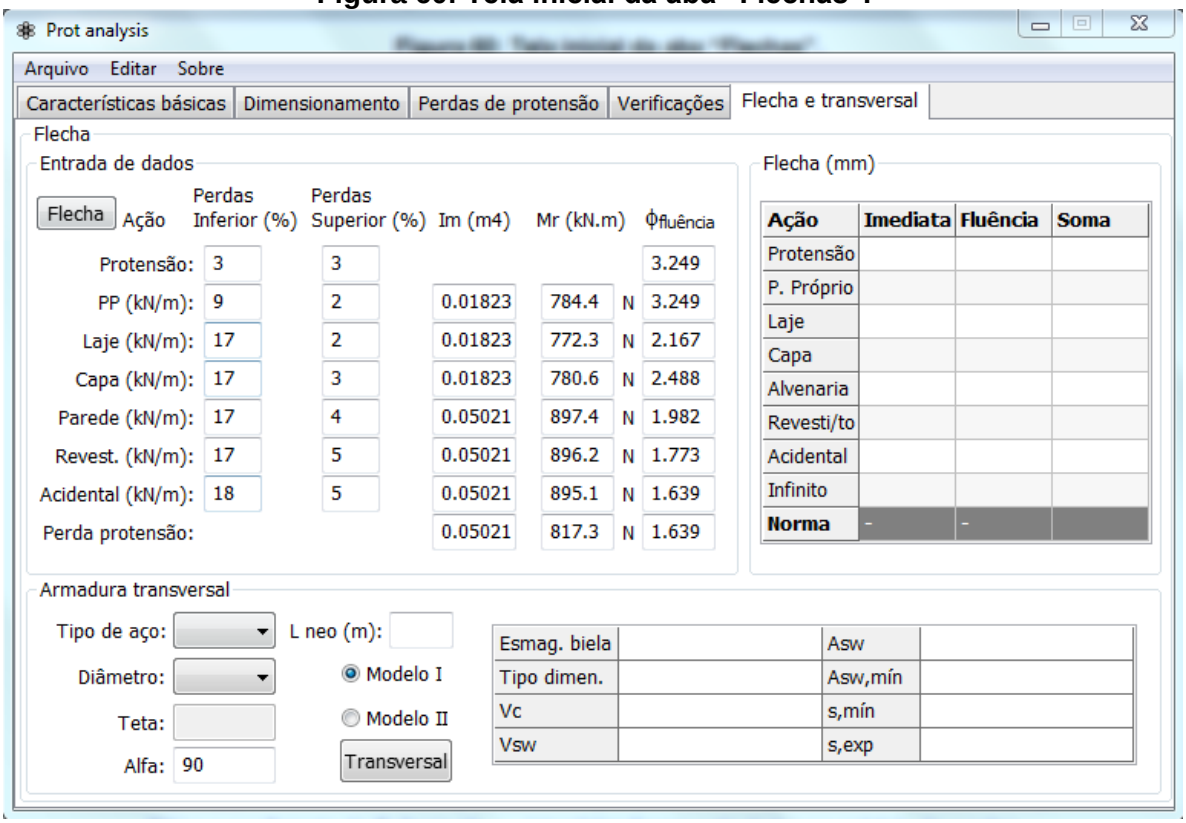


Figura 59: Verificação de tensões para acidental mínima.



Deve ser efetuada a verificação da flecha excessiva. Para tanto, é necessário definir os valores de perda de protensão totais até a idade de cada carregamento. A tela inicial do módulo de flechas pode ser analisada na Figura 60.

Figura 60: Tela inicial da aba "Flechas".



Os valores preenchidos foram inseridos automaticamente durante o cálculo das perdas de protensão. As perdas foram feitas para cada idade de carregamento, a inércia da

seção (I_m) foi calculada em função da fissuração para cada seção. O momento de fissuração (M_r) e o coeficiente de fluência ($\phi_{fluência}$) também foram calculados no módulo de perdas.

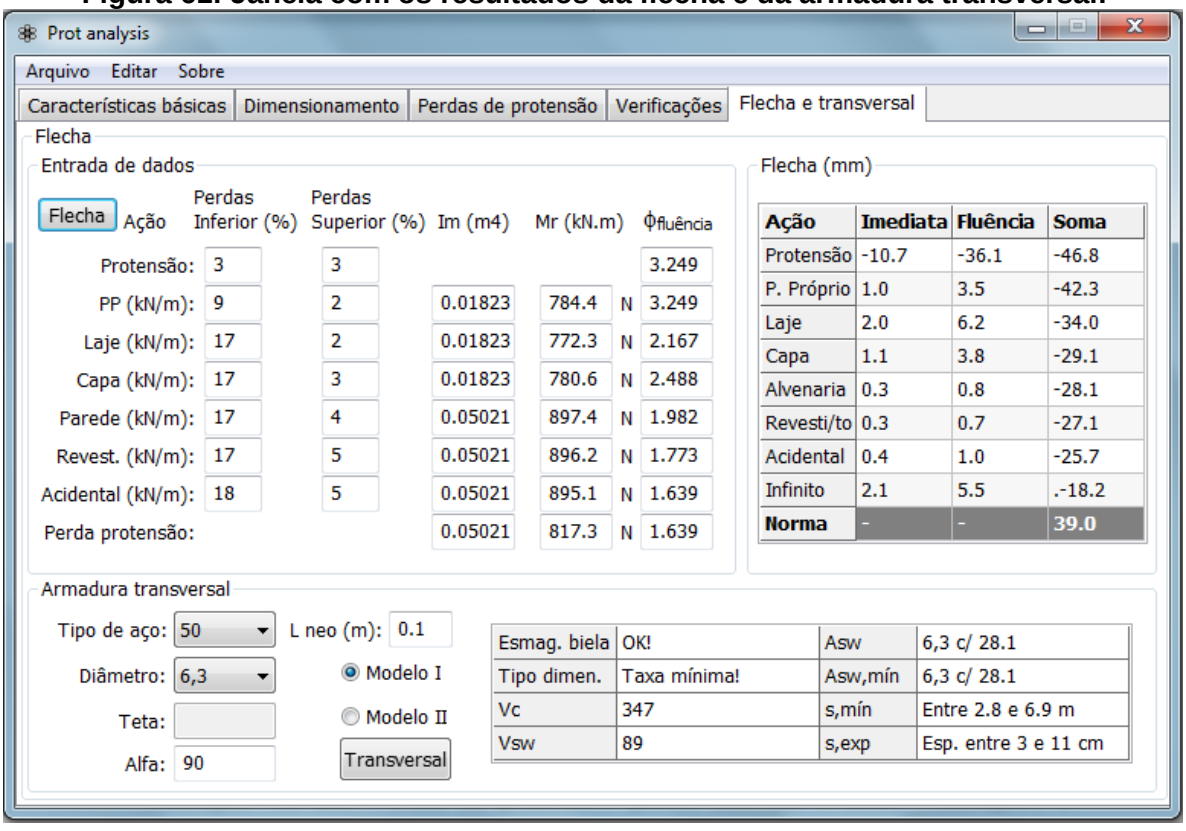
A letra entre a coluna do momento de fissuração e do coeficiente de fluência indica se houve fissuração da seção. A letra “N” indica que não houve. Caso houvesse, teria aparecido a letra “S”.

Todos esses valores podem ser editados ou inseridos manualmente conforme critério do usuário.

Nesta janela também é possível entrar com os dados da armadura transversal. Tanto a que resiste aos esforços cortantes, quanto à que resiste aos esforços de escorregamento na interface da seção pré-fabricada e da capa.

O usuário opta pelo tipo de aço de armadura passiva, seu diâmetro, a inclinação da biela de compressão (caso opte pelo cálculo no modelo II), a inclinação do estribo (com valor padrão de 90°) e a distância da extremidade da viga até o final do neoprene (caso opte por calcular na extremidade, o programa apresenta erro, pois naquela seção o momento atuante é zero, valor esse que divide o momento de descompressão).

Figura 61: Janela com os resultados da flecha e da armadura transversal.



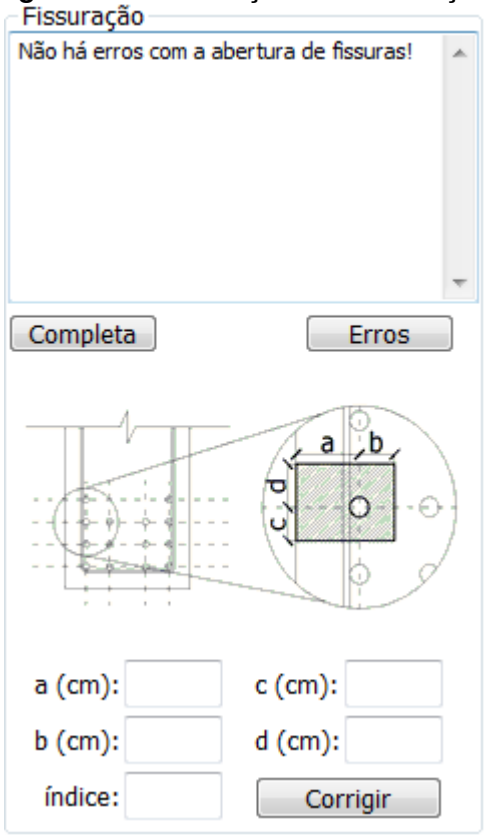
Assim que executado, o programa apresenta os resultados das flechas, assim como o valor normativo.

Quanto ao cálculo da armadura transversal, este apresenta como resultados informações sobre o esmagamento da biela, o tipo de dimensionamento (se foi feito em função da taxa mínima, do espaçamento máximo ou se foi determinada em função da resistência do estribo), a intensidade da força cortante absorvida pelos mecanismos complementares ao da treliça (V_c) e pelo aço (V_{sw}), o espaçamento da armadura, o espaçamento da armadura mínima (que nesse caso foram os mesmos), a região da viga onde se pode utilizar armadura mínima e, finalmente, o intervalo de espaçamento para a armadura exposta.

Dessa forma finaliza-se a análise e o dimensionamento da peça!

Como o exemplo modelo considera a classe de agressividade II, ou seja, não vislumbra a possibilidade de fissuração da viga, é conveniente apresentar o módulo de fissuração. Na aba “Perdas de protensão”, quando acionado o botão “Fissuração”, o programa efetua essa análise e apresenta os resultados no campo apresentado pela Figura 62.

Figura 62: Verificação de fissuração.



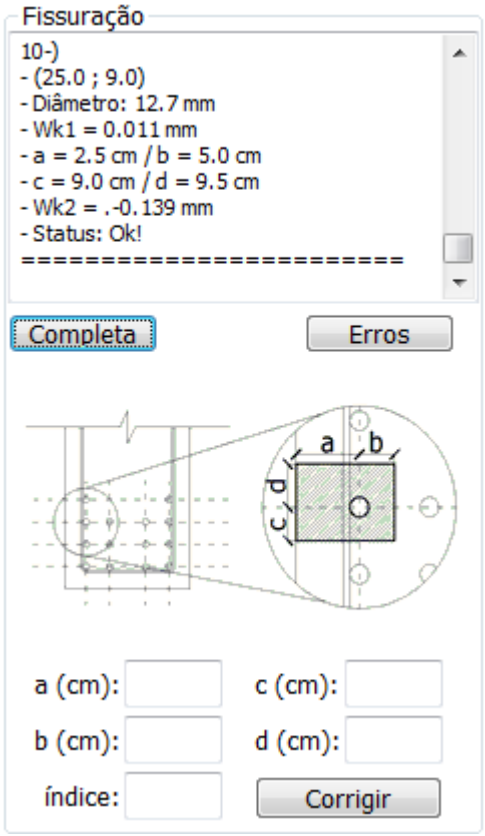
Como era de se esperar não houve problemas com a fissuração.

Como explicado anteriormente no capítulo 4, item 4.8, referente às considerações de cálculo, a verificação de fissuração é feita para cada barra ou cordoalha presente na parte inferior da seção.

Os resultados desse módulo são feitos por meio da apresentação de um relatório com os parâmetros utilizados para o cálculo. Neste relatório estão apresentados: o índice da barra, suas coordenadas, seu diâmetro, as dimensões da área de concreto que envolve a armadura, as aberturas de fissura e seu status, ou seja, se a abertura está dentro do estabelecido pela norma.

O usuário pode analisar essas informações ao clicar no botão “Completa”. Este tem o intuito de apresentar a lista completa com todas as barras e cordoalhas, conforme Figura 63. Quanto ao botão “Erros”, este retornará apenas as armaduras cuja abertura de fissura excedeu o limite.

Figura 63: Lista completa das cordoalhas verificadas



Neste caso, o usuário deve verificar se as dimensões do concreto envolvente estão corretas. Caso não estejam, deve inseri-las nos campos “a”, “b”, “c” e “d”, junto com o índice dessa armadura e clicar no botão “Corrigir”. Dessa forma o programa irá verificar se com os novos valores a abertura de fissuras está dentro dos limites.

O intuito desse manual é direcionar o usuário nos primeiros passos na sua utilização. Sendo este um programa de análise, é necessário possuir um mínimo conhecimento nas teorias relacionadas com concreto armado e protendido, assim como nas suas verificações. Espera-se que esta ferramenta seja capaz de direcionar o engenheiro na obtenção de soluções mais refinadas e confiáveis.

6 EXEMPLOS NUMÉRICOS E VALIDAÇÃO DO PROGRAMA

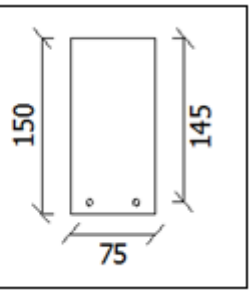
Nos exemplos numéricos têm-se o intuito de apresentar o funcionamento do programa e comparar os resultados obtidos por este com exemplos de cálculo já resolvidos na literatura técnica garantindo, assim, maior credibilidade e confiabilidade à ferramenta.

6.1 EXEMPLO 1

Neste exemplo, engloba-se o dimensionamento de uma seção retangular simples. Este foi retirado do livro de Carvalho (2012), exemplo numérico 8.2, página 321. Neste, deve-se dimensionar a armadura de protensão para a seção.

Tabela 17: Tabela com as características da seção e da viga em geral.

Ações			Concreto			Aço de protensão		
M_{g1}=	714	kN.m	f_{ck}=	40	MPa	CP190RB		
M_{g2}=	280	kN.m	$f_{ck,j}$=	20	MPa	Protensão limitada		
M_{g3}=	562	kN.m	Ψ_1=	0,4		$\sigma_{p,t=0}$=	1200	MPa
$M_{q,máx}$=	1575	kN.m	Ψ_2=	0,3		$\sigma_{p,t=\infty}$=	1000	MPa
$M_{q,mín}$=	0	kN.m				E_p=	1,95E+05	MPa



Abaixo são apresentados os resultados obtidos de forma teórica e pelo programa.

Tabela 18: Tabela comparativa dos resultados de dimensionamento do exemplo 1.

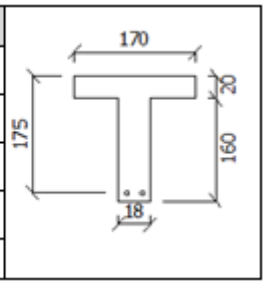
Dimensionamento							
	M_d (kN.m)	ε_s (%)	ε_p (%)	ε_7 (%)	ε_t (%)	f_{pd} (MPa)	A_p (cm²)
Teórico	4257	1,0000	0,5128	0,0254	1,5128	1507,5	20,70
Programa	4286	1,0000	0,5000	0,0254	1,5000	1506,1	20,70
Variação (%)	0,68%	0,00%	-2,56%	0,00%	-0,85%	-0,09%	0,00%

6.2 EXEMPLO 2

Neste exemplo se testa os resultados obtidos do dimensionamento de uma seção “T”. Buscou-se um exemplo em que a linha neutra passasse na alma da seção sendo necessário, portanto, considerar uma região comprimida em forma de T. Este exemplo foi retirado do livro de Carvalho (2012), exemplo numérico 6.5, página 212.

Tabela 19: Tabela com as características da seção e da viga do exemplo 2.

Dados		
M_d=	11538	kN.m
f_{ck}=	30	MPa
$\sigma_{p,t=o}$=	1200	MPa
$\sigma_{p,t=oo}$=	1000	MPa



E abaixo se apresentam os resultados:

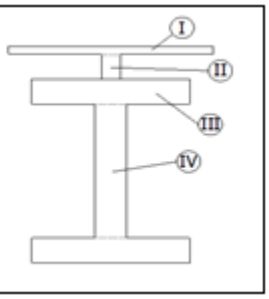
Tabela 20: Tabela comparativa dos resultados de dimensionamento do exemplo 2.

Dimensionamento							
	M_d (kN.m)	ε_s (%)	ε_7 (%)	ε_p (%)	ε_t (%)	f_{pd} (MPa)	A_p (cm²)
Teórico	11538	0,6600	0,2029	0,5120	1,3749	1501	47,50
Programa	11538	0,6580	0,2023	0,5000	1,3603	1498	47,61
Variação (%)	0,00%	-0,30%	-0,30%	-2,40%	-1,07%	-0,20%	0,23%

6.3 EXEMPLO 3

Este exemplo tem como intuito validar o módulo de dimensionamento considerando uma seção com múltiplas abas e concreto com f_{ck} diferente para a capa moldada no local e a peça pré-moldada. A lógica de programação para este módulo está apresentada no tópico 4.2.1 desse texto. Na Tabela 21 podem-se observar as características geométricas e materiais da seção transversal.

Tabela 21: Tabela com as características geométricas da seção.

Características geométricas				
Seção	b (cm)	h (cm)	f_{ck} (MPa)	
I	80	5	30	
II	10	15	30	
III	60	10	60	
IV	18	80	60	

O momento de cálculo (M_d) a ser resistido pela viga é de 2700 kN.m e a tensão de protensão é de 1450 MPa e 25% de perda total (1450 * 0,75 = 1087,5 MPa).

Supondo que a linha neutra reduzida (λx) esteja posicionada a 40 cm de profundidade, calcula-se a resistência das abas dessa viga.

$$M_I = 0{,}85 * \frac{30000}{1{,}4} * 0{,}05 * (0{,}8 - 0{,}1) * \left(1 - \frac{0{,}05}{2}\right) = 621{,}6kN.m$$

Como a seção II apresenta a menor largura dentre as quatro áreas definidas, não há abas colaborante a serem calculadas, e sua largura passa a ser o "b_w" da viga.

$$M_{III} = 0{,}808 * \frac{60000}{1{,}4} * 0{,}1 * (0{,}6 - 0{,}1) * \left(0{,}2 + \frac{0{,}1}{2}\right) = 1298{,}6kN.m$$

$$M_{IV} = 0{,}808 * \frac{60000}{1{,}4} * (0{,}4 - 0{,}3) * (0{,}18 - 0{,}1) * \left(1 - \left(0{,}3 + \frac{0{,}4 - 0{,}3}{2}\right)\right) = 180{,}1kN.m$$

Logo, o momento a ser resistido pela região retangular (M_2) é de:

$$M_2 = M_d - (M_I + M_{III} + M_{IV}) = 2700 - (621{,}6 + 1298{,}6 + 180{,}1) = 599{,}7kN.m$$

O concreto a ser considerado é uma média ponderada entre a região retangular que possui os dois tipos de concreto. Portanto:

$$f_{ck,m} = \frac{h_{I,II} * f_{ck,I,II} + h_{III,IV} * f_{ck,III,IV}}{h_{I,II} + h_{III,IV}} = \frac{0{,}2 * 30 + 0{,}2 * 60}{0{,}2 + 0{,}2} = 45\ MPa$$

$$\alpha_{c,m} = \frac{h_{I,II} * \alpha_{c,III,IV} + h_{III,IV} * \alpha_{c,III,IV}}{h_{I,II} + h_{III,IV}} = \frac{0{,}2 * 0{,}85 + 0{,}2 * 0{,}808}{0{,}2 + 0{,}2} = 0{,}829$$

E, finalmente:

$$KMD, \alpha = \frac{599{,}7}{0{,}829 * 0{,}1 * 1^2 * \frac{45000}{1{,}4}} = 0{,}225$$

$$KX, \lambda = 1 - \sqrt{1 - 2 * 0{,}225} = 0{,}258$$

$$\lambda x = 0{,}258 * 1 = 0{,}258\ m\ ou\ 25{,}8\ cm$$

Como se pode ver, a linha neutra reduzida encontrada é muito inferior à suposta (25,8 cm << 40 cm). Logo, torna-se necessário reduzir seu valor e tentar novamente. É dessa forma que o programa realiza o cálculo!

Supondo, agora, λx = 34 cm:

$$M_I = 0{,}85 * \frac{30000}{1{,}4} * 0{,}05 * (0{,}8 - 0{,}1) * \left(1 - \frac{0{,}05}{2}\right) = 621{,}6kN.m$$

$$M_{III} = 0{,}808 * \frac{60000}{1{,}4} * 0{,}1 * (0{,}6 - 0{,}1) * \left(0{,}2 + \frac{0{,}1}{2}\right) = 1298{,}6kN.m$$

$$M_{IV} = 0{,}808 * \frac{60000}{1{,}4} * (0{,}34 - 0{,}3) * (0{,}18 - 0{,}1) * \left(1 - \left(0{,}3 + \frac{0{,}34 - 0{,}3}{2}\right)\right) = 75{,}4kN.m$$

Logo, o momento a ser resistido pela região retangular (M_2) é de:

$$M_2 = M_d - (M_I + M_{III} + M_{IV}) = 2700 - (621{,}6 + 1298{,}6 + 75{,}4) = 704{,}4kN.m$$

$$f_{ck,m} = \frac{h_{I,II} * f_{ck,I,II} + h_{III,IV} * f_{ck,III,IV}}{h_{I,II} + h_{III,IV}} = \frac{0{,}2 * 30 + (0{,}1 + 0{,}04) * 60}{0{,}2 + 0{,}14} = 42{,}4\ MPa$$

$$\alpha_{c,m} = \frac{h_{I,II} * \alpha_{c,III,IV} + h_{III,IV} * \alpha_{c,III,IV}}{h_{I,II} + h_{III,IV}} = \frac{0{,}2 * 0{,}85 + (0{,}1 + 0{,}04) * 0{,}808}{0{,}2 + 0{,}14} = 0{,}833$$

$$\lambda_m = \frac{h_{I,II} * \lambda_{III,IV} + h_{III,IV} * \lambda_{III,IV}}{h_{I,II} + h_{III,IV}} = \frac{0{,}2 * 0{,}8 + (0{,}1 + 0{,}04) * 0{,}775}{0{,}2 + 0{,}14} = 0{,}790$$

$$KMD, \alpha = \frac{704{,}4}{0{,}833 * 0{,}1 * 1^2 * \frac{42400}{1{,}4}} = 0{,}279$$

$$KX, \lambda = 1 - \sqrt{1 - 2 * 0{,}279} = 0{,}335$$

$$\lambda x = 0{,}335 * 1 = 0{,}335\ m\ ou\ 33{,}5\ cm$$

O valor encontrado está bastante próximo do valor suposto inicialmente. Adota-se este valor e continua seu cálculo.

$$KZ, \phi = 1 - 0{,}5 * 0{,}335 = 0{,}8325$$

Assumindo que a ruptura ocorre no domínio 2:

$$\varepsilon_c = \frac{1{,}0}{\frac{0{,}79}{0{,}335} - 1} = 0{,}7363$$

Como a deformação foi superior a 0,35%, a ruptura não ocorreu no domínio 2. Logo:

$$\varepsilon_s = 0{,}35 * \left(\frac{0{,}79}{0{,}335} - 1\right) = 0{,}4754\ \%$$

Portanto, a deformação decorrente do carregamento é de 0,4754%.

Deformação decorrente da descompressão:

$$\varepsilon_7 = \left(\frac{23 * 137{,}75 * 1}{0{,}264} + \frac{23 * 137{,}75 * 1 * 0{,}3^2}{0{,}0595}\right) * \frac{1}{37451000} = 0{,}00045$$

$$\varepsilon_p = \frac{0{,}73 * 1087{,}5}{1460} = 0{,}5438\ \%$$

$$\varepsilon_t = 0{,}4754 + 0{,}045 + 0{,}5438 = 1{,}0237\ \%$$

$$\sigma_{sd} = 1460 + \left(\frac{1626 - 1460}{3{,}5 - 0{,}73}\right) * (1{,}0237 - 0{,}73) = 1475{,}6\ MPa$$

$$A_{p,1} = \frac{621,6}{\left(1-\frac{0,05}{2}\right)*147,5} + \frac{1298,6}{\left[1-\left(0,2+\frac{0,1}{2}\right)\right]*147,5} + \frac{75,4}{\left[1-\left(0,3+\frac{0,34}{2}\right)\right]*147,5} = 17,0\ cm^2$$

$$A_{p,2} = \frac{M_d}{KZ,\phi * d * f_s} = \frac{704,4}{0,8325*1*147,5} = 5,7\ cm^2$$

$$A_p = 17,0 + 5,7 = 22,7\ cm^2$$

Na Tabela 22, abaixo, comparam-se alguns valores emitidos pelo programa e pelo cálculo manual.

Tabela 22: Tabela comparativa entre o processo de cálculo manual e automatizado.

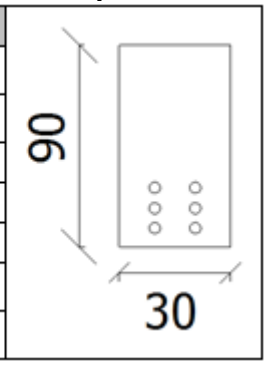
Dimensionamento										
	KMD,α_c	KX,λ	KZ,ϕ	λx (m)	ε_s (%)	ε_p (%)	ε_7 (%)	ε_t (%)	f_{pd} (MPa)	A_p (cm²)
Teórico	0,279	0,335	0,8325	0,335	4,754	5,438	0,045	10,237	1475	22,70
Programa	0,281	0,338	0,831	0,338	4,680	5,438	0,045	10,163	1479	22,49
Variação (%)	0,71%	0,89%	-0,18%	0,89%	-1,58%	0,00%	0,00%	-0,73%	0,27%	-0,93%

6.4 EXEMPLO 4

Neste exemplo mostra-se o módulo de utilização de armadura ativa em conjunto com a armadura passiva. Como dito anteriormente, por meio da armadura inserida pelo usuário, verifica-se a força resultante da armadura e compara-a com a força resultando do momento atuante.

Tabela 23: Dados do exemplo.

Dados		
M_d=	844,4	kN.m
f_{ck}=	40	MPa
1ª camada	6ϕ12,7 (CP190)	
2ª camada	3ϕ12,5 (CA-50)	
3ª camada	3ϕ12,5 (CA-25)	
$\sigma_{p,t=0}$=	1377,5	MPa
$\sigma_{p,t=\infty}$=	1087,5	MPa



Como se pode observar, a seção foi dimensionada com 3 camadas com diferentes tipos de armadura. O primeiro passo consiste, portanto, em definir o centro de gravidade da armadura e sua altura útil.

$$CG_{yp} = \frac{y_1 A_1 \sigma_1 + y_2 A_2 \sigma_2 + y_3 A_3 \sigma_3}{A_1\sigma_1 + A_2\sigma_2 + A_3\sigma_3}$$

$$CG_{yp} = \frac{4*6*1,000*108,75 + 8*3*1,227*\frac{50}{1,15} + 12*3*1,227*\frac{25}{1,15}}{6*1*108,75 + 3*1,227*\frac{50}{1,15} + 3*1,227*\frac{25}{1,15}} = 5,4cm$$

$$d = 90 - 5{,}4 = 84{,}6cm$$

$$KMD, \alpha = \frac{844{,}4}{0{,}85 * 0{,}3 * 0{,}846^2 * \frac{40000}{1{,}4}} = 0{,}162$$

$$KX, \lambda = 1 - \sqrt{1 - 2 * 0{,}137} = 0{,}178$$

$$KZ, \phi = 1 - 0{,}5 * 0{,}148 = 0{,}9111$$

$$\varepsilon_s = 1{,}0\%$$

$$\varepsilon_c = \frac{1{,}0}{\frac{0{,}8}{0{,}178} - 1} = 0{,}286$$

$$\varepsilon_7 = \left(\frac{6 * 1 * 137{,}8}{0{,}27} + \frac{6 * 1 * 137{,}8 * (0{,}45 - 0{,}04)^2}{0{,}0182}\right) * \frac{1}{31876000} = 0{,}000319$$

$$\sigma_p = 1088\, MPa \rightarrow \varepsilon_p = 0{,}5437\%$$

$$\varepsilon_t = 1 + 0{,}0335 + 0{,}5437 = 1{,}5772\% \rightarrow \sigma_{sd} = 1510{,}8\, MPa$$

De forma a tornar o cálculo mais preciso, verifica-se, novamente, qual a posição do centro de gravidade da armadura com a tensão na cordoalha considerando as deformações totais (ε_t).

$$\mathrm{CG_{yp}} = \frac{4 * 6 * 1{,}000 * 151{,}08 + 8 * 3 * 1{,}227 * \frac{50}{1{,}15} + 12 * 3 * 1{,}227 * \frac{25}{1{,}15}}{6 * 1 * 151{,}08 + 3 * 1{,}227 * \frac{50}{1{,}15} + 3 * 1{,}227 * \frac{25}{1{,}15}} = 5{,}1cm$$

$$d = 90 - 5{,}1 = 84{,}9cm$$

A diferença foi de 0,3 cm. O programa aceita uma diferença de até 0,1 cm. No entanto, ela está suficientemente próxima, considerando as dificuldades de se calcular à mão.

Finalmente, calcula-se a parcela resistente de cada armadura:

$$F_{p+s} = 6 * 1 * 151{,}08 + 3 * 1{,}227 * \frac{50}{1{,}15} + 3 * 1{,}227 * \frac{25}{1{,}15} = 1146{,}5\, kN$$

$$\frac{M_d}{z} = \frac{844{,}4}{0{,}911 * 0{,}846} = 1095{,}5\, kN$$

Como,

$$F_{p+s} > \frac{M_d}{z}$$

O estado limite último está garantido.

Tabela 24: Tabela comparativa dos resultados do exemplo.

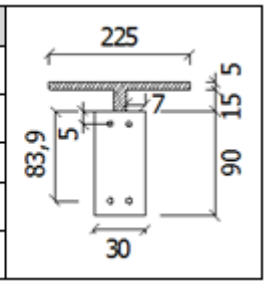
Dimensionamento							
	CGy_p (cm)	ε_s (%)	ε_p (%)	ε_7 (%)	ε_t (%)	f_{pd} (MPa)	F_s+F_p (kN)
Teórico	5,4	1,0000	0,5437	0,0335	1,5437	1510,8	1146,5
Programa	5,3	1,0000	0,5437	0,0335	1,5437	1510,8	1146,5
Variação (%)	-1,89%	0,00%	0,00%	0,00%	0,00%	0,00%	0,00%

6.5 EXEMPLO 5

Este terceiro exemplo foi retirado da dissertação de Inforsato (2009), exemplo 1. Neste, uma viga protendida foi dimensionada no ELU e suas tensões verificadas em vazio e em serviço. Considerou-se protensão limitada, portanto verificou-se a formação de fissuras para combinação frequente e descompressão para a combinação quase permanente para cada seção da viga. Neste exemplo há, ainda, o caso de seção composta e de armadura ativa em ambas as bordas. Dados:

Tabela 25: Tabela com as características geométricas e do sistema.

Dados		
M_d=	813,5	kN.m
f_{ck}=	41,37	MPa
λ=	0,75	
$\sigma_{p,t=\infty}$=	1861,6	MPa
A_p=	11,87	cm²



Abaixo são apresentados os resultados:

Tabela 26: Tabela comparativa dos resultados de dimensionamento do exemplo 5.

	M_d (kN.m)	ε_s (%)	ε_p (%)	ε_t (%)	f_{pd} (MPa)	A_p (cm²)
Teórico	1058,23	1,0000	0,5750	1,5750	1510	6,83
Programa	1058,22	1,0000	0,5596	1,5596	1509,7	6,83
Variação (%)	0,00%	0,00%	-2,75%	-0,99%	-0,02%	0,00%

Diferentemente dos exemplos 1 e 2, onde o cálculo do ε_p foi feito antes do aço escoar, neste exemplo o aço ultrapassou a deformação de escoamento e mesmo assim os resultados se apresentaram bastante próximos. O dimensionamento neste exemplo foi feito considerando-se a seção composta e, ainda assim, apresentou resultados precisos com o modelo teórico.

Tabela 27: Tabela comparativa dos resultados do cálculo das características geométricas do exemplo 5.

	Geometria				
	A (cm^2)	$W_{i,t=0}$ (m^3)	$W_{s,t=0}$ (m^3)	$W_{i,t=oo}$ (m^3)	$W_{s,t=oo}$ (m^3)
Teórico	0,270	0,0405	0,0405	0,0769	0,105
Programa	0,270	0,0405	0,0405	0,0769	0,105
Variação (%)	0,00%	0,00%	0,00%	0,00%	0,00%

O cálculo das características geométricas foi exato quando comparados os valores até certa casa decimal. Observa-se que mesmo as características considerando a seção composta foram as mesmas.

Tabela 28: Tabela comparativa dos resultados de momento fletor do exemplo 5.

Seção	**S1**	**S2**	**S3**	**S4**	**S5**
	Momento de peso próprio (kN.m)				
Teórico	28,88	51,53	67,38	77	80,21
Programa	28,88	51,33	67,37	77	80,21
Variação (%)	0,00%	-0,39%	-0,01%	0,00%	0,00%
	Momento de peso próprio + laje + capa (kN.m)				
Teórico	136,88	242,98	318,91	364,47	379,66
Programa	136,68	242,98	318,91	364,47	379,66
Variação (%)	-0,15%	0,00%	0,00%	0,00%	0,00%
	Momento de parede + revestimento (kN.m)				
Teórico	50,05	88,98	116,78	133,47	139,03
Programa	50,05	88,98	116,78	133,47	139,03
Variação (%)	0,00%	0,00%	0,00%	0,00%	0,00%
	Momento da carga acidental (kN.m)				
Teórico	92,4	164,27	215,6	246,4	256,67
Programa	92,4	164,27	215,6	246,4	256,67
Variação (%)	0,00%	0,00%	0,00%	0,00%	0,00%

Era de se esperar que os valores de momento fletor em cada seção da viga não apresentassem variações que não fossem numéricos.

Abaixo se apresentam os valores máximos de tensão em ambas as bordas e para cada décimo de vão da viga.

Tabela 29: Tabela comparativa dos resultados teóricos com o programa de tensão em vazio do exemplo 5.

Seção	S1	S2	S3	S4	S5
Tensão em vazio na borda inferior (kN/m²)					
Teórico	14855	14300	13904	13666	13587
Programa	15059	14505	14109	13871	13792
Variação (%)	1,35%	1,41%	1,45%	1,48%	1,49%
Tensão em vazio na borda superior (kN/m²)					
Teórico	-3137	-2582	-2186	-1948	-1869
Programa	-3188	-2633	-2237	-1999	-1920
Variação (%)	1,60%	1,94%	2,28%	2,55%	2,66%

Tabela 30: Tabela comparativa dos resultados teóricos com o programa de tensão em serviço do exemplo 5 (parte 1/2).

Seção	S1	S2	S3	S4	S5
	Tensão frequente na borda inferior - qmáx (kN/m²)				
Contraprova	8446	4754	2117	535	7
Programa	8621	4930	2293	711	184
Variação (%)	2,03%	3,57%	7,68%	24,75%	96,20%
	Tensão frequente na borda superior - qmáx (kN/m²)				
Contraprova	744	3812	6004	7319	7757
Programa	1132	4535	6966	8425	8911
Variação (%)	34,28%	15,94%	13,81%	13,13%	12,95%
	Tensão frequente na borda inferior - qmín (kN/m²)				
Contraprova	9167	6036	3799	2457	2010
Programa	9341	6210	3974	2633	2185
Variação (%)	1,86%	2,80%	4,40%	6,68%	8,01%
	Tensão frequente na borda superior - qmín (kN/m²)				
Contraprova	444	3280	5305	6520	6925
Programa	606	3600	5739	7022	7450
Variação (%)	26,73%	8,89%	7,56%	7,15%	7,05%

Tabela 31 Tabela comparativa dos resultados teóricos com o programa de tensão em serviço do exemplo 5 (parte 2/2).

Seção	S1	S2	S3	S4	S5
	Tensão q.perm na borda inferior - qmáx (kN/m²)				
Contraprova	8686	5181	2678	1176	675
Programa	8861	5357	2854	1352	851
Variação (%)	1,97%	3,29%	6,17%	13,02%	20,68%
	Tensão q.perm na borda superior - qmáx (kN/m²)				
Contraprova	644	3635	5771	7053	7480
Programa	954	4223	6557	7957	8424
Variação (%)	32,49%	13,92%	11,99%	11,36%	11,21%
	Tensão q.perm na borda inferior - qmín (kN/m²)				
Contraprova	9167	6036	3799	2457	2010
Programa	9341	6210	3974	2633	2185
Variação (%)	1,86%	2,80%	4,40%	6,68%	8,01%
	Tensão q.perm na borda superior - qmín (kN/m²)				
Contraprova	444	3280	5305	6520	6925
Programa	606	3600	5739	7022	7450
Variação (%)	26,73%	8,89%	7,56%	7,15%	7,05%

Como se pode observar, os resultados apresentaram variações consideráveis quando comparados os resultados do programa com os teóricos. Após uma análise mais aprofundada foi possível identificar duas razões para isso. A primeira é que o autor considerou uma área unitária de cordoalha inferior a considerada pelo programa. Em ambos os casos foi utilizada a seguinte composição: 6ϕ12,7 no positivo e 2ϕ12,7 no negativo. No entanto, o programa considera que a cordoalha de 12,7 mm possui 1,00 cm² de seção, enquanto que o autor considera um valor de 0,987 cm² para a mesma cordoalha.

Ambos os valores são permitidos pela norma. Na tabela 1 da NBR 7483:2005 podem ser verificados os valores máximos e mínimos toleráveis. A norma recomenda utilizar o valor nominal presente na referida tabela, que é equivalente a 1,009 cm².

Outro fator que levou a uma variação de resultados foi que nas verificações de serviço o autor utilizou um valor de módulo de resistência à flexão da seção composta na borda superior ($W_{s,comp}$) diferente do calculado anteriormente. Isso trouxe variações no cálculo das tensões na borda superior em serviço.

Para garantir que o módulo de verificação do programa não estava apresentando erros, foi necessário recalcular estes mesmos resultados manualmente, de forma a tirar a contraprova. Logo, isto foi feito utilizando as considerações utilizadas pelo programa. Os resultados podem ser observados na Tabela 32, na Tabela 33 e na Tabela 33.

Tabela 32: Tabela comparativa dos resultados teóricos com o programa de tensão em vazio do exemplo 5.

Seção	S1	S2	S3	S4	S5
Tensão em vazio na borda inferior (kN/m²)					
Na mão	15060	14506	14109	13872	13793
Programa	15059	14505	14109	13871	13792
Variação (%)	-0,01%	-0,01%	0,00%	-0,01%	-0,01%
Tensão em vazio na borda superior (kN/m²)					
Na mão	-3188	-2633	-2237	-2000	-1920
Programa	-3188	-2633	-2237	-1999	-1920
Variação (%)	0,00%	0,00%	0,00%	-0,05%	0,00%

Tabela 33: Tabela comparativa dos resultados da contraprova com o programa da tensão em serviço do exemplo 5 (parte 1/2).

Seção	S1	S2	S3	S4	S5
	Tensão frequente na borda inferior - qmáx (kN/m²)				
Contraprova	8607	4916	2279	697	169
Programa	8621	4930	2293	711	184
Variação (%)	0,16%	0,28%	0,61%	1,97%	8,15%
	Tensão frequente na borda superior - qmáx (kN/m²)				
Contraprova	1125	4531	6964	8424	8910
Programa	1132	4535	6966	8425	8911
Variação (%)	0,62%	0,09%	0,03%	0,01%	0,01%
	Tensão frequente na borda inferior - qmín (kN/m²)				
Contraprova	9328	6197	3961	2619	2172
Programa	9341	6210	3974	2633	2185
Variação (%)	0,14%	0,21%	0,33%	0,53%	0,59%
	Tensão frequente na borda superior - qmín (kN/m²)				
Contraprova	597	3592	5732	7016	7444
Programa	606	3600	5739	7022	7450
Variação (%)	1,49%	0,22%	0,12%	0,09%	0,08%

Tabela 34: Tabela comparativa dos resultados da contraprova com o programa da tensão em serviço do exemplo 5 (parte 2/2).

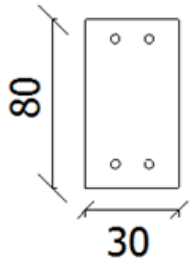
Seção	S1	S2	S3	S4	S5
	Tensão q.perm na borda inferior - qmáx (kN/m²)				
Contraprova	8848	5343	2840	1337	837
Programa	8861	5357	2854	1352	851
Variação (%)	0,15%	0,26%	0,49%	1,11%	1,65%
	Tensão q.perm na borda superior - qmáx (kN/m²)				
Contraprova	949	4219	6553	7954	8421
Programa	954	4223	6557	7957	8424
Variação (%)	0,52%	0,09%	0,06%	0,04%	0,04%
	Tensão q.perm na borda inferior - qmín (kN/m²)				
Contraprova	9328	6197	3961	2619	2172
Programa	9341	6210	3974	2633	2185
Variação (%)	0,14%	0,21%	0,33%	0,53%	0,59%
	Tensão q.perm na borda superior - qmín (kN/m²)				
Contraprova	597	3592	5732	7016	7444
Programa	606	3600	5739	7022	7450
Variação (%)	1,49%	0,22%	0,12%	0,09%	0,08%

Com os valores da contraprova se pode concluir que o módulo de verificação está funcionando suficientemente bem. Há algumas variações dos resultados, mas estas se devem a aproximações numéricas e podem ser desprezadas.

6.6 EXEMPLO 6

O exemplo 6 que analisa, exclusivamente, as verificações de tensão, tem como intuito demonstrar e validar o módulo considerando o isolamento de cabos e o comprimento de transferência, que levam em conta o crescimento linear da tensão de protensão na seção transversal. Finalmente, na Tabela 35 expressam-se os valores utilizados pelo programa e no cálculo manual.

Tabela 35: Valores dos parâmetros utilizados no cálculo.

Dados			Armadura			Tensão			
$f_{ck,j}$=	30	MPa	$A_{p,i}$=	10	cm^2		Inferior	Superior	
f_{ck}=	50	MPa	e_i=	0,325	m	$\sigma_{p,inicial}$=	1450	1450	MPa
CAA III	Prot. Completa		$A_{p,s}$=	2	cm^2	$\Delta_{p,i}$=	5	5	%
Vão=	10	m	e_s=	0,35	m	$\Delta_{p,s}$=	25	25	%
Carregamento			Seções			Cord. T	Isolado	Seção	
PP=	4,5	kN/m	S1	1	m	10	4		
Laje=	8	kN/m	S2	2	m		2		
Capa=	6	kN/m	S3	3	m		1		
Parede=	8	kN/m	S4	4	m		0		
Revest.=	5	kN/m	S5	5	m		0		
q máx=	10	kN/m	Slp_i=	1,4	m		2		
q mín=	0	kN/m	Slp_s=	1,2	m		2		

Nas colunas “Seções” estão descritas a distância de cada seção até a extremidade da viga. As seções Slp_i e Slp_s representam os comprimentos de regularização das cordoalhas inferiores e superiores, respectivamente, onde também são feitas as verificações de tensão. As colunas nomeadas como “Cord. T” e “Isolado” significam, respectivamente, a quantidade total de cordoalhas na borda inferior (afinal, o programa não contempla a possibilidade de isolar cordoalhas que estejam na borda superior) e a quantidade de cordoalhas isoladas até determinada seção. Por exemplo, da extremidade da viga até a seção S1 existem 4 cordoalhas isoladas; da S1 até a S2 existem 2 isoladas; da S2 até a S3 existe apenas 1; e da seção S4 em diante todas as cordoalhas estão aderidas ao concreto.

Primeiramente, calcularam-se os efeitos de protensão ($N_{p,i}$ e $N_{p,s}$), considerando o comprimento de regularização, em cada seção da viga para cada tempo de atuação.

- Para a seção S1 em vazio:

$$N_{p,i} = 6 * 1 * 145 * 0{,}95 * \frac{1}{1{,}4} = 590{,}4\ kN$$

Ou seja, 6 cordoalhas com 1 cm² para uma tensão de protensão de 145 kN/cm² e com 5% de perda imediata. Como o comprimento de regularização inferior é de 1,4 m e a seção S1 está a 1,0 m de distância, não se pode considerar sua tensão total, sendo necessário, portanto, considerar uma parcela linearmente proporcional, conforme previsto na norma.

$$N_{p,s} = 2 * 1 * 145 * 0{,}95 * \frac{1}{1{,}2} = 229{,}6\ kN$$

A explicação dada para o $N_{p,i}$ pode ser aplicada para o $N_{p,s}$, também. Como o comprimento de regularização da cordoalha superior é de 1,2 m, sua parcela é, portanto, referente a este comprimento.

- Para a seção S1 em serviço:

$$N_{p,i} = 6 * 1 * 145 * 0{,}75 * \frac{1}{1{,}4} = 466{,}1\ kN$$

$$N_{p,s} = 2 * 1 * 145 * 0{,}75 * \frac{1}{1{,}2} = 181{,}3\ kN$$

- Para a seção S2 em vazio:

Na seção S2, as 6 cordoalhas já estão transferindo a totalidade da protensão para a seção. Além disso, na seção S1 2 cordoalhas deixaram de estar isoladas e passaram a transferir esforços para a viga. Portanto, a seção S2 passará a receber uma parcela desses esforços, proporcional ao comprimento de regularização inferior.

$$N_{p,i} = 6 * 1 * 145 * 0{,}95 + 2 * 1 * 145 * 0{,}95 * \frac{1}{1{,}4} = 1023{,}3\ kN$$

$$N_{p,s} = 2 * 1 * 145 * 0{,}95 = 275{,}5\ kN$$

- Para a seção S2 em serviço:

$$N_{p,i} = 6 * 1 * 145 * 0{,}75 + 2 * 1 * 145 * 0{,}75 * \frac{1}{1{,}4} = 807{,}9\ kN$$

$$N_{p,s} = 2 * 1 * 145 * 0{,}75 = 217{,}5\ kN$$

- Para a seção S3 em vazio:

$$N_{p,i} = 8 * 1 * 145 * 0{,}95 + 1 * 1 * 145 * 0{,}95 * \frac{1}{1{,}4} = 1200{,}4\ kN$$

$$N_{p,s} = 2 * 1 * 145 * 0{,}95 = 275{,}5\ kN$$

- Para a seção S3 em serviço:

$$N_{p,i} = 8 * 1 * 145 * 0{,}75 + 1 * 1 * 145 * 0{,}75 * \frac{1}{1{,}4} = 947{,}7\ kN$$

$$N_{p,s} = 2 * 1 * 145 * 0{,}75 = 217{,}5\ kN$$

- Para a seção S4 em vazio:

$$N_{p,i} = 9 * 1 * 145 * 0{,}95 + 1 * 1 * 145 * 0{,}95 * \frac{1}{1{,}4} = 1338{,}1\ kN$$

$$N_{p,s} = 2 * 1 * 145 * 0{,}95 = 275{,}5\ kN$$

- Para a seção S4 em serviço:

$$N_{p,i} = 9 * 1 * 145 * 0{,}75 + 1 * 1 * 145 * 0{,}75 * \frac{1}{1{,}4} = 1056{,}4\ kN$$

$$N_{p,s} = 2 * 1 * 145 * 0{,}75 = 217{,}5\ kN$$

- Para a seção S5 em vazio:

$$N_{p,i} = 10 * 1 * 145 * 0{,}95 = 1375{,}0\ kN$$

$$N_{p,s} = 2 * 1 * 145 * 0{,}95 = 275{,}5\ kN$$

- Para a seção S5 em serviço:

$$N_{p,i} = 10 * 1 * 145 * 0{,}75 = 1087{,}5\ kN$$

$$N_{p,s} = 2 * 1 * 145 * 0{,}75 = 217{,}5\ kN$$

- Para a seção $SL_{p,i}$ em vazio:

Para este caso, nota-se que o comprimento de regularização está entre as seções S1 e S2. Logo, nessa seção consideram-se os esforços das 6 cordoalhas inferiores que não estão isoladas mais uma parcela das 2 cordoalhas inferiores que deixaram de estar isoladas na seção S1.

Em relação às cordoalhas superiores, como seu comprimento de regularização é inferior, toda sua protensão está atuando na seção $SL_{p,i}$.

$$N_{p,i} = 6 * 1 * 145 * 0{,}95 + 2 * 1 * 145 * 0{,}95 * \frac{0{,}4}{1{,}4} = 905{,}2\ kN$$

$$N_{p,s} = 2 * 1 * 145 * 0{,}95 = 275{,}5\ kN$$

- Para a seção $SL_{p,i}$ em serviço:

$$N_{p,i} = 6 * 1 * 145 * 0{,}75 + 2 * 1 * 145 * 0{,}75 * \frac{0{,}4}{1{,}4} = 714{,}6\ kN$$

$$N_{p,s} = 2 * 1 * 145 * 0{,}75 = 217{,}5\ kN$$

- Para a seção $SL_{p,s}$ em vazio:

Na seção $SL_{p,s}$, as 6 cordoalhas inferiores não transmitiram a totalidade das tensões para a viga. Assim como as 2 cordoalhas inferiores que passaram a transmitir esforços na seção S1 quando deixaram de estar isoladas.

$$N_{p,i} = 6 * 1 * 145 * 0{,}95 * \frac{1{,}2}{1{,}4} + 2 * 1 * 145 * 0{,}95 * \frac{0{,}2}{1{,}4} = 747{,}8\ kN$$

$$N_{p,s} = 2 * 1 * 145 * 0{,}95 = 275{,}5\ kN$$

- Para a seção $SL_{p,s}$ em serviço:

$$N_{p,i} = 6 * 1 * 145 * 0{,}75 * \frac{1{,}2}{1{,}4} + 2 * 1 * 145 * 0{,}75 * \frac{0{,}2}{1{,}4} = 590{,}4\ kN$$

$$N_{p,s} = 2 * 1 * 145 * 0{,}75 = 217{,}5\ kN$$

Com os esforços normais de protensão em cada borda, assim como a excentricidade inferior e superior, obtém-se os momentos de protensão em cada seção ($M_{p,i}$), pela seguinte equação:

$$M_p = N_{p,i} * e_i - N_{p,s} * e_s$$

A Tabela 36 apresenta todos os valores de M_p.

Tabela 36: Valores de M_p para cada seção em vazio e em serviço.

Momento de protensão			
	M_p (t=0)	M_p (t=∞)	
S1=	111,5	88	kN.m
S2=	236,1	186,4	kN.m
S3=	293,7	231,9	kN.m
S4=	338,5	267,2	kN.m
S5=	350,6	277,3	kN.m
SLp_i=	197,8	156,1	kN.m
SLp_s=	146,6	115,7	kN.m

De forma manual, o cálculo foi feito em função das equações de equilíbrio de tensão apresentadas abaixo:

- Inferior e superior, respectivamente, em vazio:

$$\sigma_i = \frac{N_{p,i} + N_{p,s}}{A_c} + \frac{M_p}{W_i} - \frac{M_{g1}}{W_i}$$

$$\sigma_s = \frac{N_{p,i} + N_{p,s}}{A_c} - \frac{M_p}{W_s} + \frac{M_{g1}}{W_s}$$

- Inferior e superior, respectivamente, em serviço:

$$\sigma_i = \frac{N_{p,i} + N_{p,s}}{A_c} + \frac{M_p}{W_i} - \frac{M_{g,i}}{W_i} - \frac{M_{g,f} + \gamma_{f2} * M_q}{W_{i,t=\infty}}$$

$$\sigma_s = \frac{N_{p,i} + N_{p,s}}{A_c} - \frac{M_p}{W_s} + \frac{M_{g,i}}{W_s} + \frac{M_{g,f+}\gamma_{f2} * M_q}{W_{s,t=\infty}}$$

E, finalmente, apresentam-se os resultados de tensão atuantes em cada seção, da mesma forma que fora feito no exemplo anterior, com os resultados calculados manualmente e pelo programa, assim como a variação percentual entre eles.

Tabela 37: Comparação de tensões em vazio.

Tensão em vazio na borda inferior							
	$\sigma_{i,S1}$ (kN/m²)	$\sigma_{i,S2}$ (kN/m²)	$\sigma_{i,S3}$ (kN/m²)	$\sigma_{i,S4}$ (kN/m²)	$\sigma_{i,S5}$ (kN/m²)	$\sigma_{i,SLpi}$ (kN/m²)	$\sigma_{i,SLps}$ (kN/m²)
Manual	6267	11666	13850	15613	16073	10256	8104
Programa	6268	11666	13851	15613	16107	10253	8103
Variação (%)	0,02%	0,00%	0,01%	0,00%	0,21%	-0,03%	-0,01%
Tensão em vazio na borda superior							
	$\sigma_{s,S1}$ (kN/m²)	$\sigma_{s,S2}$ (kN/m²)	$\sigma_{s,S3}$ (kN/m²)	$\sigma_{s,S4}$ (kN/m²)	$\sigma_{s,S5}$ (kN/m²)	$\sigma_{s,SLpi}$ (kN/m²)	$\sigma_{s,SLps}$ (kN/m²)
Manual	566	-843	-1551	-2166	-2323	-417	423
Programa	564	-843	-1552	-2166	-2332	-414	425
Variação (%)	-0,35%	0,00%	0,06%	0,00%	0,39%	-0,72%	0,47%

Tabela 38: Comparação de tensões para combinação frequente, no ELS-D com acidental máximo.

Comb. frequente na borda inferior - qmáx - ELS-D							
	$\sigma_{i,S1}$ (kN/m²)	$\sigma_{i,S2}$ (kN/m²)	$\sigma_{i,S3}$ (kN/m²)	$\sigma_{i,S4}$ (kN/m²)	$\sigma_{i,S5}$ (kN/m²)	$\sigma_{i,SLpi}$ (kN/m²)	$\sigma_{i,SLps}$ (kN/m²)
Manual	455	1223	451	346	235	2099	1134
Programa	456	1223	452	346	236	2085	1125
Variação (%)	0,22%	0,00%	0,22%	0,00%	0,42%	-0,67%	-0,80%
Comb. frequente na borda superior - qmáx - ELS-D							
	$\sigma_{s,S1}$ (kN/m²)	$\sigma_{s,S2}$ (kN/m²)	$\sigma_{s,S3}$ (kN/m²)	$\sigma_{s,S4}$ (kN/m²)	$\sigma_{s,S5}$ (kN/m²)	$\sigma_{s,SLpi}$ (kN/m²)	$\sigma_{s,SLps}$ (kN/m²)
Manual	4940	7321	9259	10270	10640	5668	5599
Programa	4938	7321	9257	10270	10639	5683	5607
Variação (%)	-0,04%	0,00%	-0,02%	0,00%	-0,01%	0,26%	0,14%

Tabela 39: Comparação de tensões para combinação rara, no ELS-F com acidental máximo.

Comb. rara na borda inferior - qmáx - ELS-F							
	$\sigma_{i,S1}$ (kN/m²)	$\sigma_{i,S2}$ (kN/m²)	$\sigma_{i,S3}$ (kN/m²)	$\sigma_{i,S4}$ (kN/m²)	$\sigma_{i,S5}$ (kN/m²)	$\sigma_{i,SLpi}$ (kN/m²)	$\sigma_{i,SLps}$ (kN/m²)
Manual	-389	-277	-1518	-1904	-2109	972	145
Programa	-388	-277	-1516	-1904	-2107	956	135
Variação (%)	-0,26%	0,00%	-0,13%	0,00%	-0,09%	-1,67%	-7,41%
Comb. rara na borda superior - qmáx - ELS-F							
	$\sigma_{s,S1}$ (kN/m²)	$\sigma_{s,S2}$ (kN/m²)	$\sigma_{s,S3}$ (kN/m²)	$\sigma_{s,S4}$ (kN/m²)	$\sigma_{s,S5}$ (kN/m²)	$\sigma_{s,SLpi}$ (kN/m²)	$\sigma_{s,SLps}$ (kN/m²)
Manual	5784	8821	11228	12520	12984	6795	6587
Programa	5782	8821	11226	12520	12982	6812	6597
Variação (%)	-0,03%	0,00%	-0,02%	0,00%	-0,02%	0,25%	0,15%

Tabela 40: Comparação de tensões para combinação frequente, no ELS-D com acidental mínimo.

Comb. frequente na borda inferior - qmín - ELS-D							
	$\sigma_{i,S1}$ (kN/m²)	$\sigma_{i,S2}$ (kN/m²)	$\sigma_{i,S3}$ (kN/m²)	$\sigma_{i,S4}$ (kN/m²)	$\sigma_{i,S5}$ (kN/m²)	$\sigma_{i,SLpi}$ (kN/m²)	$\sigma_{i,SLps}$ (kN/m²)
Manual	1017	2223	1763	1846	1797	2851	1792
Programa	1019	2223	1765	1846	1799	2837	1785
Variação (%)	0,20%	0,00%	0,11%	0,00%	0,11%	-0,49%	-0,39%
Comb. frequente na borda superior - qmín - ELS-D							
	$\sigma_{s,S1}$ (kN/m²)	$\sigma_{s,S2}$ (kN/m²)	$\sigma_{s,S3}$ (kN/m²)	$\sigma_{s,S4}$ (kN/m²)	$\sigma_{s,S5}$ (kN/m²)	$\sigma_{s,SLpi}$ (kN/m²)	$\sigma_{s,SLps}$ (kN/m²)
Manual	4377	6321	7946	8770	9078	4917	4940
Programa	4376	6321	7945	8770	9076	4931	4947
Variação (%)	-0,02%	0,00%	-0,01%	0,00%	-0,02%	0,28%	0,14%

Tabela 41: Comparação de tensões para combinação rara, no ELS-F com acidental mínimo.

Comb. rara na borda inferior - qmín - ELS-F							
	$\sigma_{i,S1}$ (kN/m²)	$\sigma_{i,S2}$ (kN/m²)	$\sigma_{i,S3}$ (kN/m²)	$\sigma_{i,S4}$ (kN/m²)	$\sigma_{i,S5}$ (kN/m²)	$\sigma_{i,SLpi}$ (kN/m²)	$\sigma_{i,SLps}$ (kN/m²)
Manual	1017	2223	1763	1846	1797	2851	1792
Programa	1019	2223	1765	1846	1799	2837	1785
Variação (%)	0,20%	0,00%	0,11%	0,00%	0,11%	-0,49%	-0,39%
Comb. rara na borda superior - qmín - ELS-F							
	$\sigma_{s,S1}$ (kN/m²)	$\sigma_{s,S2}$ (kN/m²)	$\sigma_{s,S3}$ (kN/m²)	$\sigma_{s,S4}$ (kN/m²)	$\sigma_{s,S5}$ (kN/m²)	$\sigma_{s,SLpi}$ (kN/m²)	$\sigma_{s,SLps}$ (kN/m²)
Manual	4377	6321	7946	8770	9078	4917	4940
Programa	4376	6321	7945	8770	9076	4931	4947
Variação (%)	-0,02%	0,00%	-0,01%	0,00%	-0,02%	0,28%	0,14%

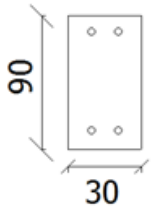
6.7 EXEMPLO 7

Este exemplo apresenta o caso em que serão comparados os resultados das perdas de protensão calculadas manualmente e pelo programa. Os cálculos contemplam as perdas por seção, de forma a verificar se estas estão coerentes para trecho definido da viga (seções em décimos de vão e nos comprimentos de regularização inferior e superior). Será, também, verificada a consideração da simultaneidade das perdas.

O exemplo contempla, ainda, os efeitos dos comprimentos de regularização e do isolamento de cordoalhas.

Tabela 42: Valores dos parâmetros utilizados no cálculo.

Dados			Armadura			Tensão e características gerais			
$f_{ck,j}$=	20	MPa	$A_{p,i}$=	10	cm^2	$\sigma_{p,inicial}$=	1450	MPa	
f_{ck}=	40	MPa	e_i=	0,385	m	Υ_{f1}=	0,4	Υ_{f2}=	0,3
CAA II	Prot. Limitada		$A_{p,s}$=	4	cm^2	Slump=	9	$T_{média}$=	20
Vão=	10	m	e_s=	0,375	m	Umidade=	70%	Cimento=	Rápido
Carregamento			Datas (dias)	Seções		Cord. T	Isolado	Seção	
PP=	6,75	kN/m	1	S1 (m)	1		4		
Laje=	12	kN/m	15	S2 (m)	2		4		
Capa=	7	kN/m	30	S3 (m)	3		0		
Parede=	5	kN/m	45	S4 (m)	4	10	0		
Revest.=	3,5	kN/m	60	S5 (m)	5		0		
q máx=	14	kN/m	75	SLp_i (m)	1,3		4		
q mín=	0	kN/m		SLp_s (m)	0,6		4		



- Perda por deformação da ancoragem

A perda por deformação da ancoragem, como dito anteriormente, é constante para todas as seções. Logo, seu cálculo é feito uma única vez e considerado para todas as seções. Considerando uma pista com 100 m de comprimento e uma acomodação da ancoragem no valor de 6 mm:

$$\Delta\sigma_{\mathrm{def,anc}} = 200000 * \frac{6}{100000} = 12\ \mathrm{MPa}$$

A perda calculada pelo programa foi igual a 12 MPa, também. Logo, a variação foi de 0%.

- Perda por relaxação inicial da armadura

Para a relaxação inicial, considera-se que a tensão na armadura já tenha perdido tensão decorrente da deformação da ancoragem. Logo, a tensão atuante nessa etapa (σ_p), ainda igual para todas as seções, equivale a:

$$\sigma_p = 1450 - 12 = 1438\ MPa$$

Como ainda não houve a entrada dos carregamentos, nesta etapa as perdas ainda são as mesmas para todas as seções. Logo, seu cálculo é realizado uma única vez:

$$R = \frac{1438}{1870} = 0{,}77$$

Para a relação entre a tensão atuante e a tensão de ruptura do aço, o valor de Ψ_{1000} para aço de cordoalha de relaxação baixa equivale a 3,2.

Como a idade de protensão é de apenas 1 dia, utiliza-se a seguinte equação:

$$\Psi_{(t,t_0)} = 3{,}2 * \left(\frac{1-0}{41{,}67}\right)^{0{,}15} = 0{,}018$$

E, finalmente, calcula-se sua perda:

$$\Delta\sigma_{rel} = 0{,}018 * 1438 = 26\ MPa$$

O programa apresentou o mesmo valor de perda. Logo, a variação foi de 0%.

- Perda por deformação imediata do concreto

Como já descrito anteriormente, a perda por deformação imediata do concreto varia para cada seção, já que se consideram os momentos atuantes em cada seção. A tensão atuante no momento anterior à perda equivale a:

$$\sigma_p = 1438 - 26{,}2 = 1411{,}8\ MPa$$

Os valores de força normal foram calculados considerando-se os efeitos do comprimento de regularização e os isolamentos de cordoalhas, de forma semelhante ao exemplo 6 sobre verificação.

- Para a seção S1:

$$N_{p,i} = 6 * 1 * 141{,}18 * \frac{1}{1{,}3} = 651{,}6\ kN$$

$$N_{p,s} = 4 * 1 * 141{,}18 = 564{,}7\ kN$$

- Para a seção S2:

$$N_{p,i} = 6 * 1 * 141{,}18 = 847{,}1\ kN$$

$$N_{p,s} = 4 * 1 * 141{,}18 = 564{,}7\ kN$$

- Para a seção S3:

$$N_{p,i} = 6 * 1 * 141{,}18 + 2 * 1 * 141{,}18 * \frac{1}{1{,}3} = 1064{,}3\ kN$$

$$N_{p,s} = 4 * 1 * 141{,}18 = 564{,}7\ kN$$

- Para a seção S4:

$$N_{p,i} = 8 * 1 * 141{,}18 + 2 * 1 * 141{,}18 * \frac{1}{1{,}3} = 1346{,}6\ kN$$

$$N_{p,s} = 4 * 1 * 141{,}18 = 564{,}7\ kN$$

- Para a seção S5:

$$N_{p,i} = 10 * 1 * 141{,}18 = 1411{,}8\ kN$$

$$N_{p,s} = 4 * 1 * 141{,}18 = 564{,}7\ kN$$

- Para a seção $SL_{p,i}$:

$$N_{p,i} = 6 * 1 * 141{,}18 = 847{,}1\ kN$$

$$N_{p,s} = 4 * 1 * 141{,}18 = 564{,}7\ kN$$

- Para a seção $SL_{p,s}$:

$$N_{p,i} = 6 * 1 * 141{,}18 * \frac{0{,}6}{1{,}3} = 391{,}0\ kN$$

$$N_{p,s} = 4 * 1 * 141{,}18 = 564{,}7\ kN$$

As equações abaixo representam a perda por seção em cada uma das bordas da viga. O valor de $\alpha_{p,i}$ equivale a 10 nesse instante, ou seja, em função do módulo de elasticidade do concreto nessa idade.

$$\Delta\sigma_{def,conc,inf} = \alpha_{p,i} * \left(\frac{N_{p,i} + N_{p,s}}{A} + \frac{\left(N_{p,i} + N_{p,s}\right) * e_i - M}{I}\right)$$

$$\Delta\sigma_{def,conc,sup} = \alpha_{p,i} * \left(\frac{N_{p,i} + N_{p,s}}{A} + \frac{M - \left(N_{p,i} + N_{p,s}\right) * e_s}{I}\right)$$

A Tabela 43 apresenta os resultados para as perdas em função das equações descritas:

Tabela 43: Comparação entre o cálculo da perda por deformação do concreto.

Deformação imediata do concreto (MPa)								
Perda		S1	S2	S3	S4	S5	SL_{pi}	SL_{ps}
Superior	Manual	43	40	34	24	22	37	52
	Programa	43	40	34	24	22	37	52
Variação (%)		0,00%	0,00%	0,00%	0,00%	0,00%	0,00%	0,00%
Inferior	Manual	47	65	87	119	126	68	18
	Programa	47	65	87	118	125	68	18
Variação (%)		0,00%	0,00%	0,00%	-0,85%	-0,80%	0,00%	0,00%

Pode-se observar uma pequena variação nas seções S4 e S5 na borda inferior. Este pode ser explicado pela aproximação numérica.

- Perda por retração

Como dito anteriormente, a perda por retração é constante para todas as seções. Logo, seu cálculo é realizado apenas uma vez e considerado para toda sua extensão.

Os parâmetros foram apresentados nos conceitos teóricos para programação e seu cálculo é feito diretamente.

Primeiramente, calculam-se os valores para obtenção do coeficiente dependente da umidade relativa (ε_{1s}):

$$\varepsilon_{1s} = -8{,}09 + \frac{70}{15} - \frac{70^2}{2284} + \frac{70^3}{133765} - \frac{70^4}{7608150} = -4{,}98$$

Em seguida, tomando o valor da altura fictícia, calcula-se o coeficiente dependente deste valor (ε_{2s})

$$\gamma = 1 + \exp(-7{,}8 + 0{,}1 * 70) = 1{,}449$$

$$h_{fic} = 1{,}449 * \frac{2 * 2700}{210} = 37{,}3\ cm$$

$$\varepsilon_{2s} = \frac{33 + 2 * 37{,}3}{20{,}8 + 3 * 37{,}3} = 0{,}811$$

Com estes dois coeficientes, calcula-se o coeficiente de retração (ε_∞);

$$\varepsilon_{c\infty} = -4{,}98 * 0{,}811 = -4{,}036$$

Definindo as idades fictícias do intervalo de tempo, dependente do tipo de endurecimento do cimento e da temperatura média diária do ambiente.

$$t = 1 * \left(\frac{20+10}{30}\right) * 1 = 1\ dias$$

No entanto, como as equações do cálculo do coeficiente $\beta_s(t)$ estão definidas apenas para o intervalo entre 3 e 10000 dias, será considerado o valor de 3 para a idade fictícia inicial e 10000 para a final. Logo:

$A = 40$;

$B = 116 * 0{,}373^3 - 282 * 0{,}373^2 + 220 * 0{,}373 - 4{,}8 = 44$;

$C = 2{,}5 * 0{,}373^3 - 8{,}8 * 0{,}373 + 40{,}7 = 37{,}5$;

$D = -75 * 0{,}373^3 + 585 * 0{,}373^2 + 496 * 0{,}373 - 6{,}8 = 255{,}4$;

$E = -169 * 0{,}373^4 + 88 * 0{,}373^3 + 584 * 0{,}373^2 - 39 * 0{,}373 + 0{,}8 = 68{,}7$.

$$\beta_s(3) = \frac{\left(\frac{3}{100}\right)^3 + 40 * \left(\frac{3}{100}\right)^2 + 44 * \left(\frac{3}{100}\right)}{\left(\frac{3}{100}\right)^3 + 37{,}5 * \left(\frac{3}{100}\right)^2 + 255{,}4 * \left(\frac{3}{100}\right) + 68{,}7} = 0{,}018$$

$$\beta_s(\infty) = \frac{\left(\frac{10000}{100}\right)^3 + 40 * \left(\frac{10000}{100}\right)^2 + 44 * \left(\frac{10000}{100}\right)}{\left(\frac{10000}{100}\right)^3 + 37{,}5 * \left(\frac{10000}{100}\right)^2 + 255{,}4 * \left(\frac{10000}{100}\right) + 68{,}7} = 1{,}002$$

Com os valores definidos, obtém-se o valor da perda por retração do concreto.

$$\Delta\sigma_{retração} = 0{,}4036 * [1{,}002 - 0{,}018] * 200000 = 79\ MPa$$

A perda por retração calculada pelo programa apresentou o mesmo resultado. Logo, não houve variações em seu cálculo.

- Perda por fluência

A perda por fluência deve ser considerada diferente para cada seção. A princípio, calculam-se os coeficientes de fluência.

A idade fictícia para cada carregamento é a idade, em dias, multiplicado por 3, este que depende do tipo de endurecimento do concreto e sua temperatura diária média.

Para concreto de endurecimento rápido, como é o caso, o coeficiente “s” equivale a 0,2.

$$\beta_1 = EXP * \left\{ s * \left[1 - \left(\frac{28}{t}\right)^{1/2} \right] \right\}$$

Utilizando a equação acima para cada idade de carregamento (não a idade fictícia!), obtém-se os valores de β_1. E, finalmente, calculam-se os valores do coeficiente de fluência rápida (φ_a).

$$\varphi_a = 0{,}8 * \left(1 - \frac{f_c(t_0)}{f_c(t_\infty)}\right)$$

Tabela 44: Valores para o cálculo do coeficiente de fluência rápida (φ_a).

Coeficiente de fluência rápida (φ_a)			
Ação	$t_{fictício}$	β_1	φ_a
Prot	3	0,424	0,519
PP	3	0,424	0,519
Laje	45	0,929	0,185
Capa	90	1,007	0,133
Parede	135	1,043	0,109
Revest	180	1,065	0,094
qMáx	225	1,081	0,084
Perda	225	1,081	0,084

Em seguida, calculam-se os parâmetros necessários para a determinação do coeficiente de deformação lenta irreversível (φ_f):

$$\varphi_{1c} = 4{,}45 - 0{,}035 * 70 = 2$$

$$\varphi_{2c} = \frac{42 + 37{,}3}{20 + 37{,}3} = 1{,}384$$

$$\varphi_{f_\infty} = 2 * 1{,}384 = 2{,}77$$

E, finalmente, os parâmetros finais para o cálculo do coeficiente de fluência:

$A = 42 * 0{,}373^3 - 350 * 0{,}373^2 + 588 * 0{,}373 + 113 = 285{,}7$;

$B = 768 * 0{,}373^3 - 3060 * 0{,}373^2 + 3234 * 0{,}373 - 23 = 797$;

$C = -200 * 0{,}373^3 + 13 * 0{,}373^2 + 1090 * 0{,}373 + 183 = 580{,}7$;

$D = 7579 * 0{,}373^3 - 31916 * 0{,}373^2 + 35343 * 0{,}373 + 1931 = 11062{,}2$;

$$\beta_s(t) = \frac{t^2 + A * t + B}{t^2 + C * t + D}$$

Os valores de $\beta_s(t)$, para cada carregamento, β_d e $\varphi(t, t_0)$ estão apresentados na Tabela 45:

$$\varphi_{d_\infty} = 0{,}4$$

Onde φ_d é o coeficiente de deformação lenta reversível.

$$\beta_d = \frac{t - t_0 + 20}{t - t_0 + 70}$$

A equação abaixo relaciona todos os parâmetros calculados e estabelece o coeficiente de fluência para cada idade de carregamento.

$$\varphi(t, t_0) = \varphi_a + \varphi_{f_\infty} * \left[\beta_f(t) - \beta_f(t_0)\right] + \varphi_{d_\infty} * \beta_d$$

Tabela 45: Dados para o cálculo do coeficiente de fluência para cada carregamento.

Coeficiente de fluência - $\varphi(t,t_0)$			
Ação	$\beta_f(t_0)$	β_d	$\varphi(t,t_0)$
Prot	0,130	0,995	3,249
PP	0,130	0,995	3,249
Laje	0,400	0,995	2,167
Capa	0,485	0,995	1,881
Parede	0,535	0,995	1,717
Revest	0,572	0,995	1,600
qMáx	0,602	0,995	1,508
Infinito	0,972	0,286	0,198

Na fluência, o valor da força normal varia de seção para seção por causa das perdas imediatas. Logo, abaixo pode ser verificado o valor de cada força normal atuando nas seções:

- Para a seção S1:

$$N_{p,i} = 6 * 1 * 136{,}49 * \frac{1}{1{,}3} = 630\ kN$$

$$N_{p,s} = 4 * 1 * 136{,}85 = 547{,}4\ kN$$

- Para a seção S2:

$$N_{p,i} = 6 * 1 * 134{,}67 = 808\ kN$$

$$N_{p,s} = 4 * 1 * 137{,}19 = 548{,}8\ kN$$

- Para a seção S3:

$$N_{p,i} = 6 * 1 * 132{,}46 + 2 * 1 * 137{,}76 * \frac{1}{1{,}3} = 998{,}5\ kN$$

$$N_{p,s} = 4 * 1 * 137{,}76 = 551\ kN$$

- Para a seção S4:

$$N_{p,i} = 8 * 1 * 129{,}33 + 2 * 1 * 141{,}18 * \frac{1}{1{,}3} = 1233{,}6\ kN$$

$$N_{p,s} = 4 * 1 * 138{,}74 = 555\ kN$$

- Para a seção S5:

$$N_{p,i} = 10 * 1 * 128{,}63 = 1286{,}3\ kN$$

$$N_{p,s} = 4 * 1 * 138{,}95 = 555{,}8\ kN$$

- Para a seção $SL_{p,i}$:

$$N_{p,i} = 6 * 1 * 134{,}34 = 806\ kN$$

$$N_{p,s} = 4 * 1 * 137{,}52 = 550{,}1\ kN$$

- Para a seção $SL_{p,s}$:

$$N_{p,i} = 6 * 1 * 139{,}34 * \frac{0{,}6}{1{,}3} = 385{,}9\ kN$$

$$N_{p,s} = 4 * 1 * 135{,}98 = 385{,}9\ kN$$

E, finalmente, por meio das equações abaixo, calcula-se a perda por fluência que ocorre em cada seção e em cada borda da viga.

$$\Delta\sigma_{fluência,inf} = \left[\frac{N_p}{A} + \left(\frac{M_p - M_{PP}}{I}\right) * e_{p,i}\right] * \varphi_1 - \left(\frac{M_{laje} * \varphi_{laje} + M_{capa} * \varphi_{capa}}{I}\right) * e_{p,i} - \left(\frac{M_{parede} * \varphi_{parede} + M_{revest} * \varphi_{revest} + M_q * \varphi_q * \Psi_2}{I_{comp}}\right) * e_{p,i,comp}$$

$$\Delta\sigma_{fluência,sup} = \left[\frac{N_p}{A} + \left(\frac{-M_p + M_{PP}}{I}\right) * e_{p,s}\right] * \varphi_1 - \left(\frac{M_{laje} * \varphi_{laje} + M_{capa} * \varphi_{capa}}{I}\right) * e_{p,s} - \left(\frac{M_{parede} * \varphi_{parede} + M_{revest} * \varphi_{revest} + M_q * \varphi_q * \Psi_2}{I_{comp}}\right) * e_{p,s,comp}$$

Tabela 46: Comparação entre o cálculo da perda por fluência do concreto.

Fluência (MPa)								
Perda		S1	S2	S3	S4	S5	SL_{pi}	SL_{ps}
Superior	Manual	135	159	171	167	168	132	138
	Programa	135	159	171	167	168	132	137
Variação (%)		0,00%	0,00%	0,00%	0,00%	0,00%	0,00%	-0,73%
Inferior	Manual	63	68	89	134	142	96	17
	Programa	63	68	89	134	142	96	18
Variação (%)		0,00%	0,00%	0,00%	0,00%	0,00%	0,00%	5,56%

Houve uma variação considerável na seção do comprimento de regularização superior na borda inferior. No entanto, considerando que seu valor numérico é baixo e os resultados foram arredondados, a diferença de um valor absoluto de apenas 1 MPa pode ser desprezado.

- Perda por relaxação final do aço

A perda por relaxação final do aço é calculada da mesma forma que a inicial. No entanto, como cada seção apresenta um valor diferente de tensão de protensão, seu cálculo deve ser feito para cada uma delas. Além disso, o período a ser considerado contempla toda a vida útil do elemento estrutural.

Calcula-se, primeiramente, a relação “R” entre a tensão após as perdas imediatas e a tensão de ruptura do aço.

$$R = \frac{\sigma_{S_i}}{f_{ptk}}$$

Em seguida, calcula-se o valor de Ψ_{1000} para essa relação.

Finalmente, como o período da relaxação final contempla toda a vida útil do elemento, a equação a ser utilizada é:

$$\Psi_{(t,t_0)} = 2{,}5 * \Psi^{1000}$$

E, concluindo, calculam-se as perdas de protensão e apresentam-se os resultados:

$$\Delta\sigma_{rel} = \Psi_{(t,t_0)} * \sigma_p$$

Tabela 47: Comparação entre o cálculo da relaxação final da armadura.

Relaxação final da armadura (MPa)								
Perda		S1	S2	S3	S4	S5	SL_{pi}	SL_{ps}
Superior	Manual	96	96	100	101	101	100	95
	Programa	96	97	99	101	102	98	94
Variação (%)		0,00%	1,03%	-1,01%	0,00%	0,98%	-2,04%	-1,06%
Inferior	Manual	96	91	86	78	77	91	105
	Programa	96	91	86	78	76	90	103
Variação (%)		0,00%	0,00%	0,00%	0,00%	-1,32%	-1,11%	-1,94%

- Simultaneidade das perdas

Considerando a simultaneidade entre as perdas diferidas, seu cálculo deve ser efetuado para cada seção considerando seus referidos parâmetros. Há alguns, no entanto, que não dependem das características de cada seção e que podem ser calculadas apenas uma vez e utilizadas de forma geral.

$$\chi_c = 1 + 0{,}5 * 3{,}249 = 2{,}624$$

$$\eta_{inf} = 1 + \frac{0{,}385^2 * 0{,}27}{0{,}0182} = 3{,}2$$

$$\eta_{sup} = 1 + \frac{0{,}375^2 * 0{,}27}{0{,}0182} = 3{,}08$$

As equações abaixo definem outros parâmetros que variam conforme alguma característica da seção. O valor de $\chi(t,t_0)$ e χ_p dependem do coeficiente de relaxação da armadura $\Psi(t,t_0)$. Já o coeficiente ρ_p depende da quantidade de armadura que há na seção. Como há cordoalhas isoladas, seu valor varia entre as seções.

$$\chi_{(t,t_0)} = -ln * [1 - \Psi(t,t_0)]$$

$$\chi_p = 1 + \chi_{(t,t_0)}$$

$$\rho_p = \frac{A_p}{A_c}$$

Tabela 48: Tabela com os parâmetros utilizados no cálculo das perdas progressivas.

	$\chi(t,t_0)$	χ_p	ρ_p
S1	0,073	1,073	0,0037
S2	0,070	1,070	0,0037
S3	0,067	1,067	0,0044
S4	0,062	1,062	0,0052
S5	0,062	1,062	0,0052
Slpi	0,070	1,070	0,0037
SLps	0,078	1,078	0,0037

Concluindo, com os parâmetros definidos, assim como as perdas de protensão consideradas isoladas, pode-se calcular a perda por seção considerando que haja interação entre elas, ou seja, de forma progressiva.

$$\Delta\sigma_{progressiva} = \frac{\Delta\sigma_{fluência} + \Delta\sigma_{retração} + \Delta\sigma_{rel,fin} * \chi(t,t_0)}{\chi_p + \chi_c * \alpha_p * \eta * \rho_p}$$

E, por fim, a Tabela 49 comparativa dos resultados.

Tabela 49: Comparação entre o cálculo das perdas considerando sua simultaneidade.

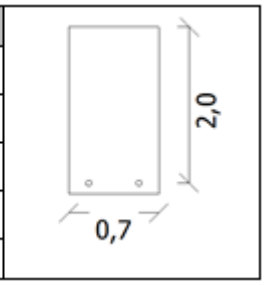
Perdas progressivas (MPa)								
Perda		**S1**	**S2**	**S3**	**S4**	**S5**	**SL_{pi}**	**SL_{ps}**
Superior	**Manual**	172	192	195	186	187	171	174
	Programa	172	191	194	185	186	170	175
Variação (%)		**0,00%**	**-0,52%**	**-0,52%**	**-0,54%**	**-0,54%**	**-0,59%**	**0,57%**
Inferior	**Manual**	116	120	131	160	166	141	81
	Programa	116	120	131	160	166	141	81
Variação (%)		**0,00%**	**0,00%**	**0,00%**	**0,00%**	**0,00%**	**0,00%**	**0,00%**

6.8 EXEMPLO 8

Este exemplo tem o intuito de validar o módulo de cálculo da armadura transversal. Primeiramente será feito o exemplo pelo modelo I e, sem seguida, pelo modelo II. Em seguida, serão apresentados os resultados comparando-os feito pelo programa e manualmente. Este exemplo foi baseado no exercício 10.8, página 423, do livro de Carvalho (2012).

Tabela 50: Tabela com os dados do sistema.

Ações			Concreto		
V_d=	2724	kN	Aço CA-50		
M_{sd}=	22765	kN.m	ϕ_{sw}=	12,5	mm
f_{ck}=	30	MPa	α=	90	°
$N_{p,inf}$=	6422	kN			



Inicia-se verificando o esmagamento da biela:

$$V_{Rd2,1} = 0{,}27 * \left(1 - \frac{30}{250}\right) * \frac{30000}{1{,}4} * 0{,}7 * 2 = 7128\ kN$$

Como $V_{Rd2,1} > V_{Sd}$, o esmagamento da biela está verificado!

Em seguida, calcula-se a parcela resistente V_c:

$$\mathrm{V_{c0}} = 0{,}6 * \frac{0{,}7 * 0{,}3}{1{,}4} * \sqrt[3]{30^2} * 1000 * 0{,}7 * 2 = 1216\ kN$$

$$\mathrm{M_0} = 1{,}22 * \left(0{,}9 * \frac{6422}{6{,}22} + \frac{6422 * 0{,}2}{1{,}22}\right) = 2418{,}1\ kN.m$$

$$\mathrm{V_c} = 1216 * \left(1 + \frac{2418{,}1}{22765}\right) = 1345{,}2 < 2 * V_{c0}$$

A parcela da cortante que será resistida pela armadura (V_{sw}) é:

$$\mathrm{V_{sw}} = 2724 - 1345{,}2 = 1378{,}8\ \mathrm{kN}$$

Logo, o espaçamento será de:

$$s = \frac{2 * 1{,}25}{1378{,}8} * 0{,}9 * 2 * \frac{50}{1{,}15} = 0{,}142\ m = 14{,}2\ cm$$

Que resulta em uma taxa de:

$$\rho_{sw,\alpha} = \frac{2 * 1{,}25}{70 * 14{,}2 * sen90} = 0{,}00252$$

Como o espaçamento foi inferior aos espaçamentos máximos estabelecidos pela norma, não há necessidade de compará-los.

Finalmente, calcula-se a taxa mínima e compara com a taxa atual:

$$\rho_{sw\alpha,mín} = 0{,}2 * \frac{0{,}3 * \sqrt[3]{30^2}}{500} = 0{,}00116$$

Como $\rho_{sw,\alpha} > \rho_{sw\alpha,mín}$, a armadura transversal deverá apresentar o seguinte arranjo: ϕ12,5 c/ 14.

Dimensiona-se a armadura para a situação de menor esforço cortante.

Verifica-se qual o espaçamento máximo especificado na norma.

$$\frac{V_{sd}}{V_{Rd2,1}} = \frac{2724}{7128} = 0{,}38 < 0{,}67 \rightarrow s_{máx} \leq 30cm$$

Em seguida, verifica-se qual o espaçamento para a taxa mínima geométrico.

$$s_{\rho,mín} = \frac{2 * 1{,}25}{70 * 0{,}00116} = 30{,}8\ cm$$

Como o menor entre os dois espaçamentos é de 30 cm, esse deve ser o valor utilizado.

E, finalmente, verifica-se a região onde se utiliza armadura mínima.

$$V_R = 644 * 0{,}7 * 2 * \left(\frac{2 * 1{,}25}{70 * 30 * sen90} * \frac{500}{1{,}15} * sen(90) * (sen(90) + \cos(90) + 0.1 * 30^{\frac{2}{3}}\right)$$

$$= 1337{,}2\ kN$$

$$L_{Vmáx} = \frac{10}{2} - \frac{1337{,}2}{544{,}8} = 2{,}5\ m$$

Dentro do intervalor de 2,5 m até 7,5 m é possível utilizar ϕ12,5 c/ 30 cm. Fora desse intervalo deve ser ϕ12,5 c/ 14 cm.

Tabela 51: Tabela comparativa dos resultados para o modelo I.

Dimensionamento - Armadura transversal - Modelo I						
	$V_{Rd2,1}$ (kN)	V_c (kN)	V_{sw} (kN)	s (m)	$x_{mín}$ (m)	$s_{máx}$ (m)
Teórico	7128,0	1345,2	1378,8	14,2	2,5	30
Programa	7128,0	1345,7	1378,3	14,2	2,5	30
Variação (%)	0,00%	0,04%	-0,04%	0,00%	0,00%	0,00%

Efetua-se, então, o cálculo desse mesmo exemplo considerando, agora, o modelo II com um ângulo de inclinação das bielas de 30 °.

Inicia-se verificando o esmagamento da biela.

$$V_{Rd2,2} = 0{,}54 * \left(1 - \frac{30}{250}\right) * \frac{30000}{1{,}4} * 0{,}7 * 2 * \sin^2 30 * (\cot 90 + \cot 30) = 6173\ kN$$

Como $V_{Rd2,1} > V_{Sd}$, o esmagamento da biela está verificado!

$$\mathrm{V_{c0}} = 0{,}6 * \frac{0{,}7 * 0{,}3}{1{,}4} * \sqrt[3]{30^2} * 1000 * 0{,}7 * 2 = 1216\ kN$$

$$\mathrm{M_0} = 1{,}22 * \left(0{,}9 * \frac{6422}{6{,}22} + \frac{6422 * 0{,}2}{1{,}22}\right) = 2378{,}4\ kN.m$$

Interpolando os valores entre:

$$V_{c1} = 1216\ quando\ V_{sd} \leq 1216\ kN$$

$$V_{c1} = 0\ quando\ V_{sd} = 6173\ kN$$

Chega-se, portanto, ao valor de V_{c1} = 846,1 kN. E:

$$\mathrm{V_c} = 846{,}1 * \left(1 + \frac{2378{,}4}{22765}\right) = 934{,}5 < 2 * V_{c0}$$

A parcela da cortante que será resistida pela armadura (V_{sw}) é:

$$\mathrm{V_{sw}} = 2724 - 934{,}5 = 1789{,}5\ \mathrm{kN}$$

Logo, o espaçamento será de:

$$s = \frac{2 * 1{,}25}{1789{,}5} * 0{,}9 * 2 * \frac{50}{1{,}15} * \cot(30) = 0{,}189\ m = 18{,}9\ cm$$

Que resulta em uma taxa de:

$$\rho_{sw,\alpha} = \frac{2 * 1{,}25}{70 * 18{,}9 * sen90} = 0{,}00189$$

Como o espaçamento foi inferior aos espaçamentos máximos estabelecidos pela norma, não há necessidade de compará-los.

Finalmente, calcula-se a taxa mínima e compara com a taxa atual:

$$\rho_{sw\alpha,mín} = 0{,}2 * \frac{0{,}3 * \sqrt[3]{30^2}}{500} = 0{,}00116$$

Como $\rho_{sw,\alpha} > \rho_{sw\alpha,mín}$, a armadura transversal deverá apresentar o seguinte arranjo: ϕ12,5 c/ 18,9.

Agora, dimensiona-se a armadura para a situação de menor esforço cortante.

Primeiramente, verifica-se qual o espaçamento máximo especificado na norma.

$$\frac{V_{sd}}{V_{Rd2,2}} = \frac{2724}{6173} = 0{,}44 < 0{,}67 \ \rightarrow s_{máx} \leq 30cm$$

Em seguida, verifica-se qual o espaçamento para a taxa mínima geométrico.

$$s_{\rho,mín} = \frac{2 * 1{,}25}{70 * 0{,}00116} = 30{,}8\ cm$$

Como o menor entre os dois espaçamentos são de 30 cm, esse deve ser o valor utilizado.

E, finalmente, verifica-se a região onde se utiliza armadura mínima.

$$V_R = 644 * 0{,}7 * 2 * \left(\frac{2 * 1{,}25}{70 * 30 * sen90} * \frac{500}{1{,}15} * sen(90) * (sen(90) + \cos(90) + 0.1 * 30^{\frac{2}{3}})\right)$$

$$= 1337{,}2\ kN$$

$$L_{Vmáx} = \frac{10}{2} - \frac{1337{,}2}{544{,}8} = 2{,}5\ m$$

Dentro do intervalor de 2,5 m até 7,5 m é possível utilizar ϕ12,5 c/ 30 cm. Fora desse intervalo deve ser ϕ12,5 c/ 19 cm.

Tabela 52: Tabela comparativa dos resultados para o modelo II.

Dimensionamento - Armadura transversal - Modelo II						
	$V_{Rd2,2}$ (kN)	V_c (kN)	V_{sw} (kN)	s (m)	$x_{mín}$ (m)	$s_{máx}$ (m)
Teórico	6173,0	934,5	1789,5	18,9	2,5	30
Programa	6173,0	936,4	1787,6	19	2,5	30
Variação (%)	0,00%	0,20%	-0,11%	0,53%	0,00%	0,00%

6.9 EXEMPLO 9

Neste exemplo, será demonstrada a metodologia utilizada pelo programa para calcular a armadura transversal que passa na interface da peça pré-moldada e na capa. Este foi adaptado de Inforsato (2009).

Primeiramente, calcula-se a tensão horizontal (τ_{sd}) atuante na interface entre os dois concretos. Considera-se uma força horizontal de 1000 kN.

$$\tau_{sd} = \frac{1000}{5 * 0{,}16} = 1250\ kN/m^2$$

Em seguida, o programa verifica, por meio de tentativas, qual o espaçamento máximo para o qual a tensão resistente (τ_{ud}) supere a tensão atuante, acima.

Iniciando por um espaçamento de 30 cm, este é reduzindo em 1 cm para cada tentativa não atendida. De forma a evitar tantas repetições, este exemplo não fará as tentativas a cada centímetro. Serão apresentadas, apenas, algumas tentativas relevantes.

Os valores são inseridos na seguinte equação. Lembrando que os valores de β_s e β_c são obtidos pela interpolação desses valores referente a esta tabela.

$$\tau_{ud} = \beta_s . \rho . f_{yd} + \beta_c . f_{ctd}$$

$$f_{ctd} = \frac{0{,}21 * \sqrt[3]{40^2}}{1{,}4} = 1{,}754\ MPa$$

- p/ espaçamento de 30 cm: ρ = 0,133 % / β_s = 0,00 / β_c = 0,30;

$$\tau_{ud} = 0 + 0{,}3 * 1754 = 526{,}3 \frac{kN}{m^2} < \tau_{sd}$$

- p/ espaçamento de 20 cm: ρ = 0,133 % / β_s = 0,00 / β_c = 0,30;

$$\tau_{ud} = 0 + 0{,}3 * 1754 = 526{,}3 \frac{kN}{m^2} < \tau_{sd}$$

- p/ espaçamento de 15 cm: ρ = 0,267 % / β_s = 0,20 / β_c = 0,367;

$$\tau_{ud} = 0{,}267 * 0{,}00267 * \frac{50}{1{,}15} + 0{,}367 * 1754 = 875{,}2 \frac{kN}{m^2} < \tau_{sd}$$

- p/ espaçamento de 12 cm: ρ = 0,333 % / β_s = 0,40 / β_c = 0,433;

$$\tau_{ud} = 0{,}4 * 0{,}00267 * \frac{50}{1{,}15} + 0{,}433 * 1754 = 1340 \frac{kN}{m^2} > \tau_{sd}$$

Finalmente, para um espaçamento de 12 cm a condição está satisfeita.

Resta saber se esta satisfaz a outra condição:

$$\tau_{ud} \leq 0{,}25 * f_{cd} = 0{,}25 * \frac{40000}{1{,}4} = 7142{,}9\ \frac{kN}{m^2}$$

E, como se pode observar, a condição está satisfeita.

O programa verifica, ainda, qual o menor valor de espaçamento para o qual essa segunda condição estará atendida. Por mais que o espaçamento mínimo não tenha sentido na economia de material, esta é explicada pelo fato de que, muitas vezes, para facilitar a execução, arredonda-se o espaçamento para um valor menor e mais fácil de ser medido. Apresentar o espaçamento mínimo garante que esta redução esteja dentro dos limites.

- p/ espaçamento de 3 cm: ρ = 1,33 % / β_s = 0,90 / β_c = 0,6;

$$\tau_{ud} = 0{,}9 * 0{,}0133 * \frac{50}{1{,}15} + 0{,}6 * 1754 = 6270\frac{kN}{m^2} > 0{,}25 * f_{cd}$$

- p/ espaçamento de 2 cm: ρ = 2,0 % / β_s = 0,90 / β_c = 0,6;

$$\tau_{ud} = 0{,}9 * 0{,}02 * \frac{50}{1{,}15} + 0{,}6 * 1754 = 8878{,}7\frac{kN}{m^2} > 0{,}25 * f_{cd}$$

Portanto, o espaçamento mínimo deve ser maior ou igual a 3 cm.

Tabela 53: Tabela comparativa do exemplo.

Dimensionamento - Armadura transversal						
	τ_{Sd} (kN/m²)	$\tau_{Ud,30}$ (kN/m²)	$\tau_{Ud,15}$ (kN/m²)	$\tau_{Ud,12}$ (kN/m²)	$\tau_{Ud,3}$ (kN/m²)	0,25.fcd (kN/m²)
Teórico	1250,0	526,3	875,2	1340	6270,0	7142,9
Programa	1250,0	526,3	875,2	1340	6270,0	7142,9
Variação (%)	0,00%	0,00%	0,00%	0,00%	0,00%	0,00%

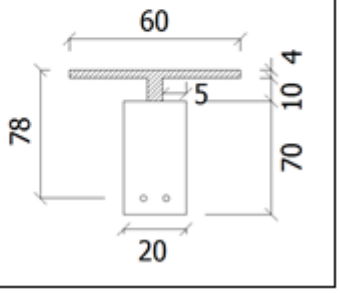
6.10 EXEMPLO 10

Este exemplo tem o objetivo de demonstrar a metodologia de cálculo utilizado pelo programa para verificação da fissuração e da flecha. Buscou-se um exemplo onde ocorra a fissuração, de forma a demonstrar uma situação mais complexa e que demonstre as potencialidades do programa.

Tabela 54: Tabela de dados do exemplo.

Dados								
$F_{ck,j}$=	20	MPa	E_p=	200000	Mpa	Vão=	10	m
F_{ck}=	40	MPa	E_s=	210000	MPa	η_p=	1,2	
$F_{ck,capa}$=	20	MPa	σ_i=	1453	MPa	η_s=	2,25	
A_p=	4 ϕ 12,7		$e_{p,i}$=	0,28	Mpa	$L_{p,i}$=	0,8	m
A_s=	2 ϕ 12,5		CAA I			Ψ_1=	0,6	

Carregamento			Perda		$\phi_{fluência}$
PP	3,5	kN/m	8	%	3,304
Laje	7	kN/m	14	%	2,225
Capa	6,8	kN/m	14	%	2,559
Parede	6,2	kN/m	13	%	2,033
Revest.	5,4	kN/m	13	%	1,825
Acid.	13,5	kN/m	12	%	1,691
Perda			19	%	1,691



Primeiramente, será demonstrado os cálculos referentes à verificação da fissuração. Este módulo inicia calculando as características geométricas no estádio II puro.

Para tanto, deve-se verificar a posição da linha neutra. O programa utiliza um método de tentativa, onde se assume uma profundidade e verifica-se se esta apresenta um resultado relativamente próximo da solução.

Aqui, portanto, não será demonstrada cada tentativa. A solução da linha neutra no estádio II apresenta-se a uma profundidade de 15,35 cm da borda superior. Portanto, está localizada na mesa da seção pré-moldada.

Utiliza-se, portanto, a seguinte equação:

$$\frac{b_1 * h_1 * E_{cs,1}}{2} * \left(2 - \frac{h_1}{x_{II}}\right) + \frac{b_2 * h_2 * E_{cs,2}}{2} * \left(2 - \frac{2 * h_1}{x_{II}} - \frac{h_2}{x_{II}}\right) + \frac{b_3 * (x_{II} - h_1 - h_2) * E_{cs,3}}{2} * \left(1 - \frac{h_1 + h_2}{x_{II}}\right) = \left(A_p * E_p + A_s * E_s\right) * \left(\frac{d}{x_{II}} - 1\right) - \left(A_p{}' * E_p + A_s{}' * E_s\right) * \left(\frac{d'}{x_{II}} - 1\right)$$

$$\frac{60 * 4 * 2128,7}{2} * \left(2 - \frac{4}{15,35}\right) + \frac{10 * 10 * 2128,7}{2} * \left(2 - \frac{2 * 4}{15,35} - \frac{4}{15,35}\right) + \frac{20 * (15,35 - 4 - 10) * 3187,6}{2} * \left(1 - \frac{4 + 10}{15,35}\right) = (4 * 20000 + 2,45 * 21000) * \left(\frac{78}{15,35} - 1\right)$$

$$536167,6 - 536504,4 = -336,8$$

Como na equação não entram os valores adimensionais da deformação, a ordem de grandeza aparenta valores elevados e não próximos. Mas se for considerá-la, a diferença entre a parcela resistente do concreto e do aço aproxima-se de zero.

Definida a posição da linha neutra, calcula-se o momento de inércia ($I_{x,II}$) da seção comprimida em relação à linha neutra.

$$I_{x,II} = \frac{b_1 * {h_1}^3}{3} + \frac{b_2 * {h_2}^3}{3} + \frac{b_3 * {x_{II}}^3}{3} + b_1 * h_1 * \left(x_{II} - \frac{h_1}{2}\right)^2 + b_2 * h_2 * \left(x_{II} - h_1 - \frac{h_2}{2}\right)^2 + b_3 * x_{II} * \left(x_{II} - h_1 - h_2 - \frac{x_{II}}{2}\right)^2 + A_p * \frac{E_p}{E_{cs,m}} * (x_{II} - d)^2 + A_s * \frac{E_p}{E_{cs,m}} * (x_{II} - d)^2$$

Onde,

$E_{cs,m}$: média ponderada do módulo de elasticidade secante entre os diferentes concretos constituintes da seção.

$$I_{x,II} = \left[\frac{60 * 4^3}{3} + \frac{10 * 10^3}{3} + \frac{20 * 15{,}35^3}{3} + 60 * 4 * \left(15{,}35 - \frac{4}{2}\right)^2 + 10 * 10 * \left(15{,}35 - 4 - \frac{10}{2}\right)^2 + 20 * 15{,}35 * \left(15{,}35 - 4 - 10 - \frac{15{,}35}{2}\right)^2 + 4 * \frac{20000}{2206{,}6} * (15{,}35 - 78)^2 + 2{,}45 * \frac{21000}{2206{,}6} * (15{,}35 - 78)^2\right] * 10^{-8} = 0{,}00282\ m^4$$

Em seguida, calcula-se o momento de descompressão (M_0):

$$M_0 = \left(\frac{435{,}9}{0{,}14} + \frac{435{,}9 * 0{,}28^2}{0{,}0057}\right) * \frac{0{,}0057}{0{,}28} = 185{,}6\ kN.m$$

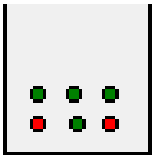
Nessa etapa, o programa segue uma rotina de repetição que aborda cada uma das barras que constituem a armadura da viga. Aqui, será demonstrada apenas uma das cordoalhas.

Calcula-se a tensão atuante no centro de gravidade da cordoalha em análise:

$$\sigma_i = \frac{(462{,}5 - 185{,}6)}{I_{x,II}} * (0{,}423 - 0{,}04) * \frac{210000}{31876} = 236{,}1\ MPa$$

É necessário, também, definir a área de concreto que envolve a armadura analisada.

Figura 64: Posição da armadura na seção transversal.



A cor verde representa armadura ativa. E a cor vermelha passiva. A primeira camada está na altura de 4 cm enquanto que a segunda está a 8 cm da borda inferior.

A cordoalha analisada é a única na primeira camada. Como se pode ver, sua área de influência deveria estar contida a uma distância média entre a segunda camada e entre as barras de aço doce. No entanto, como forma de simplificar os problemas do algoritmo, ela está envolvida verticalmente pela distância média até a segunda camada e, horizontalmente, limitada pelas faces laterais da seção. Logo, seguindo a Figura 40, as dimensões da área de concreto envolvente definidas pelo programa são de:

a = 9,5 cm / b = 9,5 cm / c = 4,0 cm / d = 4,0 cm.

E a taxa de armadura dessa área:

$$\rho_{ri} = \frac{A_p}{(a+b)*(c+d)} = 0{,}00658$$

Finalmente:

$$w \text{ é } o\ menor\ valor\ entre \begin{cases} \dfrac{12{,}7}{12{,}5*1{,}2} * \dfrac{236{,}1}{200000} * \dfrac{3*236{,}1}{3{,}51} \\ \dfrac{12{,}7}{12{,}5*1{,}2} * \dfrac{236{,}1}{200000} * \left(\dfrac{4}{0{,}00658} + 45\right) \end{cases} = \begin{cases} 0{,}202\ mm \\ 0{,}653\ mm \end{cases}$$

De acordo com a Tabela 5, o limite de abertura de fissura não pode superar 0,2 mm. Como o valor foi bem próximo, o usuário pode entrar manualmente com os valores a, b, c, d da área de concreto envolvente. Utilizando os valores reais, as dimensões passam para os seguintes valores:

a = 2,5 cm / b = 2,5 cm / c = 4,0 cm / d = 4,0 cm.

$$\rho_{ri} = 0{,}025$$

$$w_{k,2} = \frac{12{,}7}{12{,}5*1{,}2} * \frac{236{,}1}{200000} * \left(\frac{4}{0{,}025} + 45\right) = 0{,}205\ mm$$

O valor diminui bastante. O programa considera a situação mais crítica para definir a área envolvente. Mas o usuário pode refinar este cálculo manualmente.

Em seguida, inicia-se o cálculo da flecha. Sabemos que a peça fissurou, logo, deve ser definido sob qual carregamento esta ocorreu. A partir da qual se considera a inércia média de Branson.

A princípio, calculam-se os momentos de fissuração para entrada de cada carregamento.

- Para o peso próprio:

$$M_r = \left(1{,}5 * 2210 + \frac{534{,}7}{0{,}14}\right) * 0{,}0163 + 149{,}7 = 266\ kN.m$$

O momento atuante de peso próprio é de 43,75 kN.m. Logo, a peça não fissurou nessa idade.

- Para a entrada da laje:

$$M_r = \left(1{,}5 * 3343 + \frac{499{,}8}{0{,}14}\right) * 0{,}0163 + 140{,}0 = 279{,}9\ kN.m$$

O momento atuante na idade de entrada da laje é de 131,25 kN.m. Logo, a peça não fissurou nessa idade.

Observa-se que a força normal de protensão varia para cada carregamento por causa das perdas de protensão, que são consideradas.

- Para a entrada da capa:

$$M_r = \left(1{,}5 * 3509 + \frac{499{,}8}{0{,}14}\right) * 0{,}0163 + 140{,}0 = 283{,}9\ kN.m$$

O momento atuante na idade de entrada da laje é de 216,25 kN.m. Logo, a peça não fissurou nessa idade.

- Para a entrada da parede:

$$M_r = 1{,}2 * 3509 * 0{,}0244 + \frac{505{,}6}{0{,}14} * 0{,}0163 + 141{,}6 = 302{,}1\ kN.m$$

O momento atuante na idade de entrada da laje é de 293,75 kN.m. Logo, a peça não fissurou nessa idade. No cálculo, o coeficiente de forma da seção (α) muda, pois se considera a seção composta em forma de “T”.

- Para a entrada do revestimento:

$$M_r = 1{,}2 * 3509 * 0{,}0244 + \frac{505{,}6}{0{,}14} * 0{,}0163 + 141{,}6 = 302{,}1\ kN.m$$

O momento atuante na idade de entrada da laje é de 361,25 kN.m. Quando da entrada do carregamento de revestimento, o momento atuante superou o momento de fissuração. Portanto, o programa passa a considerar a inércia média de Branson para o cálculo da flecha.

$$I_m = \left(\frac{302{,}1}{361{,}25}\right)^3 * 0{,}0103 + \left[1 - \left(\frac{302{,}1}{361{,}25}\right)^3\right] * 0{,}002818 = 0{,}007194\ m^4$$

- Para a entrada da carga acidental:

$$M_r = 1{,}2 * 3509 * 0{,}0244 + \frac{511{,}5}{0{,}14} * 0{,}0163 + 143{,}2 = 304{,}4\ kN.m$$

O momento atuante na idade de entrada da laje é de 530,0 kN.m, superior ao momento de fissuração, como era de se esperar.

$$I_m = \left(\frac{304{,}4}{530}\right)^3 * 0{,}0103 + \left[1 - \left(\frac{304{,}4}{530}\right)^3\right] * 0{,}002818 = 0{,}004235\ m^4$$

- Considerando o carregamento final com as perdas de protensão no tempo infinito:

$$M_r = 1{,}2 * 3509 * 0{,}0244 + \frac{469{,}6}{0{,}14} * 0{,}0163 + 131{,}5 = 287{,}9\ kN.m$$

O momento atuante na idade de entrada da laje é de 530,0 kN.m, superior ao momento de fissuração, como era de se esperar.

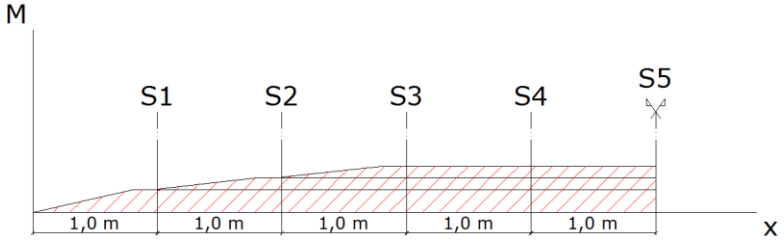
$$I_m = \left(\frac{287{,}9}{530}\right)^3 * 0{,}0103 + \left[1 - \left(\frac{287{,}9}{530}\right)^3\right] * 0{,}002818 = 0{,}004017\ m^4$$

Em seguida é feito o cálculo das flechas pelo método dos momentos estáticos das áreas dos momentos fletores.

Iniciando pela contraflecha inicial. O sistema é constituído por um comprimento de regularização de 0,8 m, 2 cordoalhas isoladas até a seção S1 e 1 até a seção S2.

Logo, o diagrama de momento é semelhante à Figura 65:

Figura 65: Diagrama de momento das cordoalhas inferiores.



$$a = (A_1 * x_1 + A_2 * x_2 + A_3 * x_3) * \frac{1}{EI}$$

$$A_1 * x_1 = 2 * \frac{145{,}3 * 0{,}97 * 0{,}28 * 0{,}8}{2} * \left(\frac{2}{3} * 0{,}8\right) + 2 * \frac{145{,}3 * 0{,}97 * 0{,}28 * 4{,}2}{2} * \left(\frac{4{,}2}{2} + 0{,}8\right)$$

$$= 978{,}2$$

$$A_2 * x_2 = \frac{145{,}3 * 0{,}97 * 0{,}28 * 0{,}8}{2} * \left(\frac{2}{3} * 0{,}8 + 1\right) + \frac{145{,}3 * 0{,}97 * 0{,}28 * 3{,}2}{2} * \left(\frac{3{,}2}{2} + 0{,}8 + 1\right)$$

$$= 453{,}6$$

$$A_3 * x_3 = \frac{145{,}3 * 0{,}97 * 0{,}28 * 0{,}8}{2} * \left(\frac{2}{3} * 0{,}8 + 2\right) + \frac{145{,}3 * 0{,}97 * 0{,}28 * 2{,}2}{2} * \left(\frac{2{,}2}{2} + 0{,}8 + 2\right)$$

$$= 378{,}6$$

$$a_0 = -\frac{(978{,}2 + 453{,}6 + 378{,}6)}{0{,}0057 * 22540000} * 1000 = -14\ mm$$

Os carregamentos distribuídos provocam um diagrama de momento fletor parabólico e um linear provocado pela reação de apoio. Portanto, para cada carregamento, calcula-se a flecha:

$$a = \left(\frac{L^3 * p}{24} * \frac{5}{16} * L - \frac{L^3 * p}{16} * \frac{L}{3}\right) * \frac{1}{EI}$$

- Carregamento de peso próprio:

$$a_1 = \left(\frac{10^3 * 3{,}5}{24} * \frac{5}{16} * 10 - \frac{10^3 * 3{,}5}{16} * \frac{10}{3}\right) * \frac{1}{0{,}0057 * 22540000} * 1000 = 2{,}1\ mm$$

- Carregamento de laje:

$$a_2 = \left(\frac{10^3 * 7}{24} * \frac{5}{16} * 10 - \frac{10^3 * 7}{16} * \frac{10}{3}\right) * \frac{1}{0{,}0057 * 30740000} * 1000 = 3{,}1\ mm$$

- Carregamento da capa:

$$a_3 = \left(\frac{10^3 * 6{,}8}{24} * \frac{5}{16} * 10 - \frac{10^3 * 6{,}8}{16} * \frac{10}{3}\right) * \frac{1}{0{,}0057 * 31876000} * 1000 = 2{,}9\ mm$$

- Carregamento da parede:

$$a_4 = \left(\frac{10^3 * 6{,}2}{24} * \frac{5}{16} * 10 - \frac{10^3 * 6{,}2}{16} * \frac{10}{3}\right) * \frac{1}{0{,}0103 * 31876000} * 1000 = 1{,}5\ mm$$

- Carregamento de revestimento:

$$a_5 = \left(\frac{10^3 * 5{,}4}{24} * \frac{5}{16} * 10 - \frac{10^3 * 5{,}4}{16} * \frac{10}{3}\right) * \frac{1}{0{,}007194 * 31876000} * 1000 = 1{,}8\ mm$$

- Carregamento acidental:

$$a_6 = \left(\frac{10^3 * 13{,}5}{24} * \frac{5}{16} * 10 - \frac{10^3 * 13{,}5}{16} * \frac{10}{3}\right) * \frac{1}{0{,}004235 * 31876000} * 1000 = 3{,}1\ mm$$

E, finalmente, a flecha decorrente da perda de protensão. É calculada de forma semelhante à contraflecha, alterando apenas o valor das perdas. Como forma de simplificar o diagrama, considera-se a perda em cada seção como constante. Portanto, para as cordoalhas que não estão isoladas, utilizam-se as perdas calculadas na seção S1. Para a cordoalha isolada que passa a "trabalhar" na seção S1, utilizam-se as perdas que ocorreram na seção S2 e, finalmente, para a cordoalha que passa a trabalhar na seção S2, adota-se as perdas da seção S3. E, assim, sucessivamente se houvesse mais cordoalhas isoladas.

$$A_1 * x_1 = 2 * \frac{145{,}3 * 0{,}832 * 0{,}28 * 0{,}8}{2} * \left(\frac{2}{3} * 0{,}8\right) + 2 * \frac{145{,}3 * 0{,}832 * 0{,}28 * 4{,}2}{2} * \left(\frac{4{,}2}{2} + 0{,}8\right)$$
$$= 839{,}0$$

$$A_2 * x_2 = \frac{145{,}3 * 0{,}828 * 0{,}28 * 0{,}8}{2} * \left(\frac{2}{3} * 0{,}8 + 1\right) + \frac{145{,}3 * 0{,}828 * 0{,}28 * 3{,}2}{2} * \left(\frac{3{,}2}{2} + 0{,}8 + 1\right)$$
$$= 387{,}2$$

$$A_3 * x_3 = \frac{145{,}3 * 0{,}813 * 0{,}28 * 0{,}8}{2} * \left(\frac{2}{3} * 0{,}8 + 2\right) + \frac{145{,}3 * 0{,}813 * 0{,}28 * 2{,}2}{2} * \left(\frac{2{,}2}{2} + 0{,}8 + 2\right)$$
$$= 317{,}3$$

$$a_7 = \frac{(839 + 387{,}2 + 317{,}3)}{0{,}004017 * 31876000} * 1000 = 12{,}1\ mm$$

Em seguida, é feito o cálculo da flecha diferida utilizando os valores dos coeficientes de fluência previamente calculados. E considerando o coeficiente de ajuste:

$$a_{0,dif} = -14 * (1 + 3{,}304) * \frac{22540}{31876} = -42{,}8\ mm$$

$$a_{1,dif} = 2{,}1 * (1 + 3{,}304) * \frac{22540}{31876} = 6{,}5\ mm$$

$$a_{2,dif} = 3{,}1 * (1 + 2{,}225) * \frac{30740}{31876} = 9{,}7\ mm$$

$$a_{3,dif} = 2{,}9 * (1 + 2{,}559) * 1 = 10{,}4\ mm$$

$$a_{4,dif} = 1{,}5 * (1 + 2{,}033) * 1 = 4{,}5\ mm$$

$$a_{5,dif} = 1{,}8 * (1 + 1{,}825) * 1 = 5{,}2\ mm$$

$$a_{6,dif} = 3{,}1 * (1 + 1{,}691) * 1 = 8{,}4\ mm$$

$$a_{7,dif} = 12{,}1 * (1 + 1{,}691) * 1 = 32{,}4\ mm$$

A Tabela 55 apresenta os resultados das flechas totais por carregamento e a somatório com a flecha final:

Tabela 55: Valores das flechas para cada carregamento e sua somatória, feitos manualmente e pelo programa.

Cálculo manual / automatizado								
	a_{imed}	a_{dif}	a_{total}	a_{soma}	a_{imed}	a_{dif}	a_{total}	a_{soma}
Contraflecha	-14	-42,8	-56,8	-56,8	-14	-42,7	-56,7	-56,7
Peso próprio	2,1	6,5	8,6	-48,2	2,1	6,5	8,6	-48,1
Laje	3,1	9,7	12,8	-35,4	3,1	9,7	12,8	-35,3
Capa	2,9	10,4	13,3	-22,1	2,9	10,4	13,3	-22,0
Parede	1,5	4,5	6,0	-16,1	1,5	4,5	6,0	-16,0
Revestimento	1,8	5,2	7,0	-9,1	1,8	5,1	6,9	-9,1
Acidental	3,1	8,4	11,5	2,4	3,1	8,4	11,5	2,4
Perda	12,1	32,4	44,5	46,9	12,0	32,3	44,3	46,7

Encerrando o exemplo, apresenta-se a tabela com a comparação entre os resultados obtidos pelo programa.

Tabela 56: Tabela comparativa dos resultados.

Variação (%)				
	a_{imed}	a_{dif}	a_{total}	a_{soma}
Contraflecha	0,0%	-0,2%	-0,2%	-0,2%
Peso próprio	0,0%	0,0%	0,0%	-0,2%
Laje	0,0%	0,0%	0,0%	-0,3%
Capa	0,0%	0,0%	0,0%	-0,5%
Parede	0,0%	0,0%	0,0%	-0,6%
Revestimento	0,0%	-2,0%	-1,4%	0,0%
Acidental	0,0%	0,0%	0,0%	0,0%
Perda	-0,8%	-0,3%	-0,5%	-0,4%

7 CONCLUSÃO

7.1 Análise dos resultados

Neste tópico serão analisados os resultados comparativos obtidos em cada módulo de cálculo desenvolvido no programa.

O exemplo 1 teve como foco analisar o módulo de dimensionamento para o caso mais simples que se pode ter: uma seção retangular simples para um carregamento distribuído. Na etapa de dimensionamento houve uma pequena diferença no cálculo do ε_p que pode ser explicado pela simplificação estabelecida na norma para o gráfico “tensão x deformação” para aço de armadura ativa. No exemplo feito no livro, este valor é baseado nos dados fornecidos por Vasconcelos (1980) e que, como mostrado na Tabela 18, apresentam pequena diferença. Apesar disso, os resultados de A_p apresentaram-se suficientemente próximos.

No exemplo 2 foi, novamente, analisado o módulo de dimensionamento. Desta vez para uma seção “T” com a linha neutra passando abaixo da mesa. Logo, seu intuito foi de analisar o dimensionamento para uma seção comprimida em forma de “T”. Neste exemplo nota-se que houve uma diferença em relação ao exemplo de referência parecida com o exemplo 1 no alongamento do ε_p, cujo motivo foi explicado no exemplo anterior. Apesar disso, o cálculo do M_d, do M_1 e do A_p foram praticamente idênticos aos valores teóricos. Logo, o módulo de identificação da seção “T” e seu respectivo dimensionamento apresentaram resultados coerentes.

O exemplo 3, novamente, verificou o módulo de dimensionamento. Desta vez para uma seção com múltiplas abas. Pode-se dizer que as pequenas diferenças apresentadas nos resultados são ocasionadas pelo processo de tentativa adotado pelo programa e por aproximações numéricas do cálculo manual.

O intuito do exemplo 4 foi o de verificar o funcionamento de uma viga que se utiliza de armadura ativa e passiva para resistir aos esforços últimos. Os resultados foram bastante precisos, apresentando uma pequena variação na posição do centro de gravidade da armadura. Como o exemplo foi desenvolvido de forma manual e o programa adota um critério de tentativas para a definição do centro de gravidade da armadura, pode-se dizer que essa diferença é decorrente da inviabilidade de se aproximar, manualmente, com a mesma eficiência que o programa.

No exemplo 5 validou-se o módulo de verificação de tensões. Este exemplo foi baseado em Inforsato (2009). Este módulo apresentou variações consideráveis quando comparado com os resultados obtidos pelo autor. No entanto, como se pode observar no exemplo, foi possível identificar as causas que levaram a isso. Em seguida, quando os problemas foram corrigidos, os resultados se apresentaram suficientemente próximos.

No exemplo 6 validaram-se as verificações de tensão, novamente, considerando a variação da força de protensão pelo comprimento de regularização e com isolamento de cordoalhas. Por meio deste exemplo foi possível entender como o programa trabalha a questão do isolamento de cabos e do comprimento de regularização. Pelos resultados, observa-se que a variação percentual entre o programa e o cálculo manual foi suficientemente próximo. Os dois únicos valores que excederam 1% de variação ocorreu nas seções SLp,i (-1,67%) e SLp,s (-7,41%) da combinação rara com acidental máxima. Analisando mais detalhadamente, observa-se que sua diferença absoluta não diferiu mais do que outros valores (algo em torno de 10 kN/m²).

O exemplo 7 demonstra a utilização e as considerações de cálculo do módulo das perdas de protensão. Analisando as tabelas comparativas, pode-se observar que as variações que surgiram para algumas perdas foram decorrentes de aproximações numéricas.

O exemplo 8 contempla o módulo de cálculo da armadura transversal de uma seção retangular simples, pelos modelos I e II. Houve algumas variações decorrentes de aproximações numéricas, mas em geral as aproximações estão suficientemente próximas.

O exemplo 9 mostra o dimensionamento da armadura transversal que passa na interface entre a peça pré-fabricada e a capa. O exemplo não apresentou variações de aproximação até a primeira casa decimal.

E, finalmente, o exemplo 10 demonstrou os módulos de fissuração e de flecha. Optou-se por um exemplo que englobasse ambas as situações para demonstrar como o programa considera a fissuração no cálculo da flecha. Houve pequenas variações de aproximação. Os valores obtidos pelo programa e pelas planilhas foram aproximados até a primeira casa decimal sem maiores variações a serem considerados.

7.2 Conclusão

Como se pode observar nas análises dos resultados, o programa vem apresentando resultados bastante próximos daqueles obtidos manualmente (com auxílio de planilhas). Buscou-se demonstrar exemplos que englobassem a maior parte dos casos de projeto, considerando-se arranjos coerentes de armadura, carregamentos e dimensões

convencionais. Partiu-se do pressuposto que o usuário tem conhecimento e experiência no cálculo de estruturas protendidas.

Destaca-se, no entanto, que seria necessário comparar seus resultados com outros programas. No entanto, pela escassa oferta de programas livres e pelas dificuldades financeiras de se utilizar programas comerciais, não foi possível efetuar essa comparação.

Este programa teve o intuito de propiciar que o usuário teste o maior número de soluções para uma viga pré-tracionada, dando total liberdade para que este altere coeficientes normativos e apresente dimensionamentos mais refinados e confiáveis. Espera-se que esta filosofia agregue tanto a profissionais que almejam diversificar as opções de solução, assim como a estudantes de engenharia, no aprendizado dos conceitos referentes a concreto protendido.

7.3 Sugestões para trabalhos futuros

Para garantir o contínuo desenvolvimento do programa aqui apresentado e do grupo Calco, sugerem-se algumas opções de trabalhos a serem desenvolvidos:

- Módulo de detalhamento gráfico, onde o programa emite as informações necessárias para que um programa gráfico realize o detalhamento da armadura longitudinal e transversal nas seções transversais e longitudinais;
- Programa semelhante para vigas protendidas pós-tracionadas (com e sem aderência). Uma das dificuldades desse programa seria a variação da excentricidade dos cabos ao longo do comprimento da viga. No entanto, tem como vantagem não precisar de isolamento de cordoalhas;
- Módulos de cálculo a serem acrescentados nesse programa que visem outros tipos de verificação, como: verificação contra incêndio, armadura de fretagem, tirantes para dente gerber;
- Programa para dimensionamento de consolos.

8 REFERÊNCIAS BIBLIOGRÁFICAS

American Concrete Institute (2004). Building Code Requirements for Structural Concrete. **ACI 318M-05.** Farmington Hills, MI.

American Concrete Institute (2014). Building Code Requirements for Structural Concrete. **ACI 318R-14.** Farmington Hills, MI.

ARAUJO, C. A. M. **Estudo de lajes alveolares pré-tracionadas com auxílio de programa computacional.** 2007. 128 f. Dissertação (Mestrado em Engenharia Civil) – Programa de Pós-Graduação em Engenharia Civil (PPGEC), Universidade Federal de Santa Catarina, Florianópolis, 2007.

ASSOCIAÇÃO BRASILEIRA DE NORMAS TÉCNICAS. **NBR-6118:2014** Projeto de estruturas de concreto - Procedimentos. Rio de Janeiro, 2004. 221 f.

ASSOCIAÇÃO BRASILEIRA DE NORMAS TÉCNICAS. **NBR-7483:2005** Cordoalhas de aço para concreto protendido. Rio de Janeiro, 2005. 12 f.

ASSOCIAÇÃO BRASILEIRA DE NORMAS TÉCNICAS. **NBR-9062:2006** Projeto e execução de estruturas de concreto pré-moldado. Rio de Janeiro, 2006. 37 f.

BEER, F. P., JOHNSTON JR, E. R. e DEWOLF, J. T. **Resistência dos materiais – Mecânica dos Materiais.** 4a Ed. São Paulo: McGraw-Hill Interamericana do Brasil Ltda., 2006. 758 f.

CARVALHO, R. C. e FIGUEIREDO FILHO, J. R. **Cálculo e detalhamento de estruturas usuais de concreto armado.** 3a Ed. São Carlos: EdUfscar, 2009. 367 f.

CARVALHO, R. C. **Estruturas em concreto protendido.** 1a Ed. São Paulo: Pini, 2012. 431 f.

COLLINS, M. P. **Prestressed concrete structures**. 1a Ed. Ontario: Copywell, 1997. 766 f.

Comité Europeu de Normalização (2010). Projecto de estruturas de betão – Parte 1-1: Regras gerais e regras para edifícios. **NP EN 1992-1-1** (Versão portuguesa). Bruxelas, Bélgica.

EL DEBS, M. K. **Concreto Pré-Moldado: Fundamentos e Aplicações.** 1ª Ed. São Carlos: EESC – USP, 2000. 456 f.

ELLIOTT, K. S. **Precast Concrete Structures**. 1ª Ed. Oxford: Butterworth-Heinemann, 2002. 375 f.

FALEIROS JUNIOR, J. H. **Procedimentos de cálculo, verificação e detalhamento de armaduras longitudinais na seção transversal em elementos protendidos.** 2010. 193 f. Dissertação (Mestrado em Engenharia Civil) – Programa de Pós-Graduação em Construção Civil (PPGECiv), Universidade Federal de São Carlos, São Carlos, 2010.

FALEIROS JUNIOR, J.H. **Arquivo pessoal**. 2014.

FUSCO, P.B. **Tecnologia do concreto estrutural.** São Paulo: PINI, 2008. 179 f.

GHALI, A. e FAVRE, R. **Concrete Structures: Stresses and Deformations**. New York: Chapman and Hall, 1986. 352 f.

HLADNI. I. **Entendendo e dominando o Delphi**. São Paulo: Universo dos Livros Editora Ltda., 2006. 576 f.

INFORSATO, T. B. **Considerações sobre o projeto, cálculo e detalhamento de vigas pré-fabricadas protendidas com aderência inicial em pavimentos de edificações**. 2009. 259 f. Dissertação (Mestrado em Engenharia Civil) – Programa de Pós-Graduação em Construção Civil (PPGECiv), Universidade Federal de São Carlos, São Carlos, 2009.

LIMA, V. S. **Projeto de superestruturas de pontes em concreto protendido aplicando a técnica de balanços progressivos.** 2011. 163 f. Dissertação (Mestrado em Engenharia Civil) – Programa de Pós-Graduação em Construção Civil (PPGECiv), Universidade Federal de São Carlos, São Carlos, 2011.

LYN, T. Y. **Design of prestressed concrete structures**. 3a Ed. New Jersey: John Wiley & Sons, 1981. 646 f.

MELO, C. E. E. **Manual Munte de Projetos em Pré-Fabricados de Concreto**. 2ª Ed. São Paulo: Pini, 2007. 540 f.

NAWY, E. G. **Prestressed concrete**. 2a Ed. New Jersey: Prentice Hall, 1995. 789 f.

PETRUCELLI, N. S. **Considerações sobre projeto e fabricação de lajes alveolares protendidas.** 2009. 126 f. Dissertação (Mestrado em Engenharia Civil) – Programa de Pós-Graduação em Construção Civil (PPGECiv), Universidade Federal de São Carlos, São Carlos, 2009.

PCI (2010). Precast & Prestressed Concrete Institute. **Design Handbook**. 7th edition. Chicago, IL.

RICHARDSON, J. **Area and Moment of Inertia of a Polygon.** Disponível em: <http://richardson.eng.ua.edu/Former_Courses//CE_331_fa09/Projects/A_and_I_of_Polygon.pdf>. Acesso em: abril 2014.

SCAVASSIN, R. M. **Programa para cálculo e detalhamento de armadura de vigas pré-tracionadas.** 2012. 193 f. Monografia – Departamento de Engenharia Civil (DECiv), Universidade Federal de São Carlos, São Carlos, 2012.

SANTOS, A. P. **Análise da continuidade em lajes alveolares: estudo teórico e experimental.** 2014. 370 f. Tese (Doutorado em Engenharia de Estruturas) – Escola de Engenharia de São Carlos, São Carlos, 2014.

SILVA, I. M. **Análise da redistribuição de esforços em vigas de concreto protendido com seções compostas.** 2003. 74 f. Dissertação (Mestrado em Engenharia Civil) – Escola Politécnica da Universidade de São Paulo, São Paulo, 2003.

SWAN. T. **Delphi 4 - Bíblia do programador**. São Paulo: Editora Berkeley Brasil, 1999. 830 f.

TREVIZOLI, G. M., CARVALHO, R. C., Silva, L. M. e CASS, A. J. R. **Estudo comparativo do cálculo de vigas pré-tracionadas segundo as normas brasileira, europeia e americana.** XXXVI Jornadas Sul Americanas de Engenharia Estrutural. Montevidéu, Uruguai, Novembro, 2014.

TREVIZOLI, G. M., CARVALHO, R. C., CASS, A. J. R. e Silva, L. M. **Situações limite para vigas pré-fabricadas.** 56° Congresso Brasileiro de Concreto – IBRACON. Natal/RN, Outubro, 2014.

TIMOSHENKO, S. P. **Resistência dos materiais - Vol. 1**. 3a Ed. Rio de Janeiro: Ao Livro Técnico SA, 1966. 451 f.

VASCONCELOS, A. C. **Manual prático para a correta utilização dos aços no concreto protendido em obediência às normas utilizadas.** Belo Horizonte: Livros técnicos e científicos; Companhia Siderúrgica Belgo-Mineira, 1980.

9 APÊNDICE A

Nesse apêndice serão apresentadas as listagens do programa principal. Estão inclusas as listagens referentes à tela principal do programa, do menu “Editar”, do botão para seção composta, para o botão “Datas e coeficientes”, do gráfico de momento fletor por profundidade da linha neutra (M x LN), a tela com os créditos e as bibliotecas de funções desenvolvidas para o programa.

9.1 LISTAGEM DA TELA PRINCIPAL DO PROGRAMA

```
var
  FRM_PTracao: TFRM_PTracao;
  n, m: integer;
  vCont, vDesl: integer;
  scale: extended;
  vFckj, vFck, vFctj, vFctm, vTParRet: extended;
  vL, vPP, vLaje, vCapa, vParede, vRevestimento, vQmax, vQmin, vPTotal,
vP2Total, vMd: extended;
  vB, vB1, vB2, vB3, vH1, vH2, vH3, vH4, vH5, vHt, vA, vACapa, vIx, vUAr,
vUArCa, vWs, vWi, vHPre: extended;
  vABarra, vTCordI, vTCordS, vgSigmaSD: extended;
  tCG, tWi, tWf, tpsi, tIxY: tPoints;
  tAp: tArmadura;
  aMd: tEsforcos;
  aVetor: array of tPoints;
  aDimensoes: array of tConcreto;
  aMomentos: array[0..8] of tEsforcos;
  aVetorInt: array of tPoint;
  aLimTen: array[0..4] of extended;
  aFissura: array of tFissura;
implementation

{$R *.lfm}

{ TFRM_PTracao }

               {ABERTURA E MENUS DO PROGRAMA}
procedure TFRM_PTracao.FormActivate(Sender: TObject);
begin
  DecimalSeparator := '.';
  //m e n são os contadores para numerar as linhas de armadura de flexão na
aba Verificação
  m := 1;
  n := 1;
  //Valores default do menu Edit
  vTVazio := 1.2; vCVazio := 0.7; vTFiss := 1.0; vTDesc := 0; vCServ :=
0.7;
```

```
  vFpyd := 1460; vEpsyd := 0.73; vFptd := 1626; vEpsu := 3.5; vEp :=
200000; vEdoce := 210000;
  vAlfaE := 1; vAlfaForma := 1.5; vAlfaForma2 := 1.2; vXD := 0.45;
  vAlfaC := 0.85; vLambida := 0.8; vGamaC := 1.4;
  vAlfaCIL := 0.85; vLambidaIL := 0.8; vGamaCIL := 1.4;
  vGamaS := 1.15; vGamaF := 0.9; vTInf := 10000; vFid := 0.4;
  aR[0] := 0.5; aR[1] := 0.6; aR[2] := 0.7; aR[3] := 0.8;
  aPsiMil[0] := 0; aPsiMil[1] := 1.3; aPsiMil[2] := 2.5; aPsiMil[3] := 3.5;
  aAlfa[0] := 1; aAlfa[1] := 2; aAlfa[2] := 3;
  aCoefS[0] := 0.38; aCoefS[1] := 0.25; aCoefS[2] := 0.2;
  aFc[0] := 1.433; aFc[1] := 1.267; aFc[2] := 1.208;
  aBeta[0].X := 0.2; aBeta[0].Y := 0.5; aBeta[1].X := 0.0;
  aBeta[1].Y := 0.9; aBeta[2].X := 0.3; aBeta[2].Y := 0.6;
  vEtaS := 2.25; vWk := 0.2; vLimFle := 250;
  //Valores default do form "Datas e coeficientes"
  vMPP := 1.4; vMLaje := 1.4; vMCapa := 1.4; vMParede := 1.4; vMRevest :=
1.4; vMAcid := 1.4;
  vDProt := 1; vDPP := 1; vDLaje := 15; vDCapa := 30; vDParede := 45;
vDRevest := 60; vDAcid := 75; vDPerda := 75;
  IMG_Fissura.Picture.Bitmap.LoadFromResourceName(hInstance, 'FISSURA');
end;

procedure TFRM_PTracao.FormClose(Sender: TObject; var CloseAction:
TCloseAction);
var
  i: integer;
begin
  for i := 0 to LBX_Linha.Count - 1 do
    tDados(LBX_Linha.Items.Objects[i]).Free;
end;

procedure TFRM_PTracao.MenuItem1Click(Sender: TObject);
begin
  UeDdADOS.FRM_EdDados.ShowModal;
end;

procedure TFRM_PTracao.MNI_CreditosClick(Sender: TObject);
begin
  ucreditos.FRM_Creditos.ShowModal;
end;

procedure TFRM_PTracao.RBT_ModeloIChange(Sender: TObject);
begin
  if RBT_ModeloI.Checked = True then
    EDB_Teta.Enabled := False
  else
    EDB_Teta.Enabled := True;
end;

               {ABA CARACTERÍSTICAS BÁSICAS}
//Ativa dados de seção composta
procedure TFRM_PTracao.CBX_CompostaClick(Sender: TObject);
begin
  if CBX_Composta.Checked = True then
     begin;
       BTN_DadosSComposta.Enabled := True;
       BTN_DadosSComposta.ShowHint := True;
     end
  else
     begin
       BTN_DadosSComposta.Enabled := False;
```

```
       BTN_DadosSComposta.ShowHint := False;
       FRM_SComposta.EDB_hCapa.Text := ''; hCapa := 0;
       FRM_SComposta.EDB_hLaje.Text := ''; hLaje := 0;
       FRM_SComposta.EDB_lLaje.Text := ''; lLaje := 0;
       FRM_SComposta.EDB_fckCapa.Text := ''; fckCapa := 0;
       FRM_SComposta.EDB_Bf.Text := ''; vBfCapa := 0;
     end
end;

//Verificação do número de entrada
procedure TFRM_PTracao.EDB_vaoKeyUp(Sender: TObject; var Key: Word;
  Shift: TShiftState);
begin
 if not isNum(TEdit(Sender).Text) then
    begin
      TEdit(Sender).Text := '';
      ShowMessage('Algarismo inválido!');
    end;
end;

procedure TFRM_PTracao.EDB_B1KeyUp(Sender: TObject; var Key: Word;
  Shift: TShiftState);
begin
  if not isNeg(TEdit(Sender).Text) then
    begin
      TEdit(Sender).Text := '';
      ShowMessage('Algarismo inválido!');
    end;
end;

procedure TFRM_PTracao.EDB_PMAcChange(Sender: TObject);
begin
  EDB_Yb.Text := EDB_Ya.Text;
end;

procedure TFRM_PTracao.SGR_DimensaoSetEditText(Sender: TObject; ACol,
  ARow: Integer; const Value: string);
var
  TextoCelula: String;
begin
  TextoCelula := SGR_Dimensao.Cells[ACol, ARow];
  if not isNeg(TextoCelula) then
    begin
      ShowMessage('Algarismo inválido!');
      SGR_Dimensao.Cells[ACol, ARow] := '';
    end;
end;

procedure TFRM_PTracao.SGR_DimensaoPrepareCanvas(sender: TObject; aCol,
  aRow: Integer; aState: TGridDrawState);
begin
  if SGR_Dimensao.Cells[aCol, aRow] = '-' then
    SGR_Dimensao.Canvas.Brush.Color := clGray;
end;

procedure TFRM_PTracao.SGR_DimensaoSelectCell(Sender: TObject; aCol,
  aRow: Integer; var CanSelect: Boolean);
begin
  CanSelect := SGR_Dimensao.Cells[aCol,aRow] <> '-';
end;
```

```
procedure TFRM_PTracao.SGR_Geom1PrepareCanvas(sender: TObject; aCol,
  aRow: Integer; aState: TGridDrawState);
begin
  if (aCol = 2) or (aCol = 0) then
    SGR_Geom1.Canvas.Brush.Color := clBtnFace;
  SGR_Geom1.GridLineColor := clGrayText;
end;

procedure TFRM_PTracao.SGR_Geom2PrepareCanvas(sender: TObject; aCol,
  aRow: Integer; aState: TGridDrawState);
begin
  if (aCol = 2) or (aCol = 0) then
    SGR_Geom2.Canvas.Brush.Color := clBtnFace;
  SGR_Geom2.GridLineColor := clGrayText;
end;

procedure TFRM_PTracao.SGR_DimensPrepareCanvas(sender: TObject; aCol,
  aRow: Integer; aState: TGridDrawState);
begin
  if (aCol = 0) or (aCol = 2) then
    SGR_Dimens.Canvas.Brush.Color := clBtnFace;
  if (aRow = 0) or (aRow = 4) then begin
    SGR_Dimens.Canvas.Brush.Color := clBlack;
    SGR_Dimens.Canvas.Font.Color := clWhite;
    SGR_Dimens.Canvas.Font.Bold := True;
  end;
end;

procedure TFRM_PTracao.SGR_PerSupIniPrepareCanvas(sender: TObject; aCol,
  aRow: Integer; aState: TGridDrawState);
begin
  if (aCol = 6) and (aRow <> 0) or (aCol = 7) and (aRow <> 0) then
    SGR_PerSupIni.Canvas.Brush.Color := $00C4FFFF;
  if aRow = 4 then
    SGR_PerSupIni.Canvas.Font.Bold := True;
end;

procedure TFRM_PTracao.SGR_PerSupDifPrepareCanvas(sender: TObject; aCol,
  aRow: Integer; aState: TGridDrawState);
begin
  if (aCol = 6) or (aCol = 7) then
    SGR_PerSupDif.Canvas.Brush.Color := $00C4FFFF;
  if aRow = 3 then
    SGR_PerSupDif.Canvas.Font.Bold := True;
end;

procedure TFRM_PTracao.SGR_PerInfIniPrepareCanvas(sender: TObject; aCol,
  aRow: Integer; aState: TGridDrawState);
begin
  if (aCol = 6) and (aRow <> 0) or (aCol = 7) and (aRow <> 0) then
    SGR_PerInfIni.Canvas.Brush.Color := $00C4FFFF;
  if aRow = 4 then
    SGR_PerInfIni.Canvas.Font.Bold := True;
end;

procedure TFRM_PTracao.SGR_PerInfDifPrepareCanvas(sender: TObject; aCol,
  aRow: Integer; aState: TGridDrawState);
begin
  if (aCol = 6) or (aCol = 7) then
    SGR_PerInfDif.Canvas.Brush.Color := $00C4FFFF;
  if aRow = 3 then
```

```
    SGR_PerInfDif.Canvas.Font.Bold := True;
end;

procedure TFRM_PTracao.SGR_TenQmax1PrepareCanvas(sender: TObject; aCol,
  aRow: Integer; aState: TGridDrawState);
begin
  if (aCol = 5) or (aCol = 6) then
    SGR_TenQmax1.Canvas.Brush.Color := $00C4FFFF;
  if (StrToFloat(SGR_TenQmax1.Cells[aCol, aRow]) < aLimTen[3]) or
     (StrToFloat(SGR_TenQmax1.Cells[aCol, aRow]) > aLimTen[2]) then begin
    SGR_TenQmax1.Canvas.Font.Color := clWhite;
    SGR_TenQmax1.Canvas.Brush.Color := clRed;
  end;
end;

procedure TFRM_PTracao.SGR_TenQmin1PrepareCanvas(sender: TObject; aCol,
  aRow: Integer; aState: TGridDrawState);
begin
  if (aCol = 5) or (aCol = 6) then
    SGR_TenQmin1.Canvas.Brush.Color := $00C4FFFF;
  if (StrToFloat(SGR_TenQmin1.Cells[aCol, aRow]) < aLimTen[3]) or
     (StrToFloat(SGR_TenQmin1.Cells[aCol, aRow]) > aLimTen[2]) then begin
    SGR_TenQmin1.Canvas.Font.Color := clWhite;
    SGR_TenQmin1.Canvas.Brush.Color := clRed;
  end;
end;

procedure TFRM_PTracao.SGR_TenQmax2PrepareCanvas(sender: TObject; aCol,
  aRow: Integer; aState: TGridDrawState);
begin
  if (aCol = 5) or (aCol = 6) then
    SGR_TenQmax2.Canvas.Brush.Color := $00C4FFFF;
  if (StrToFloat(SGR_TenQmax2.Cells[aCol, aRow]) < aLimTen[4]) or
     (StrToFloat(SGR_TenQmax2.Cells[aCol, aRow]) > aLimTen[2]) then begin
    SGR_TenQmax2.Canvas.Font.Color := clWhite;
    SGR_TenQmax2.Canvas.Brush.Color := clRed;
  end;
end;

procedure TFRM_PTracao.SGR_TenQmin2PrepareCanvas(sender: TObject; aCol,
  aRow: Integer; aState: TGridDrawState);
begin
  if (aCol = 5) or (aCol = 6) then
    SGR_TenQmin2.Canvas.Brush.Color := $00C4FFFF;
  if (StrToFloat(SGR_TenQmin2.Cells[aCol, aRow]) < aLimTen[4]) or
     (StrToFloat(SGR_TenQmin2.Cells[aCol, aRow]) > aLimTen[2]) then begin
    SGR_TenQmin2.Canvas.Font.Color := clWhite;
    SGR_TenQmin2.Canvas.Brush.Color := clRed;
  end;
end;

procedure TFRM_PTracao.SGR_TenVazioPrepareCanvas(sender: TObject; aCol,
  aRow: Integer; aState: TGridDrawState);
begin
  if (aCol = 5) or (aCol = 6) then
    SGR_TenVazio.Canvas.Brush.Color := $00C4FFFF;
  if (StrToFloat(SGR_TenVazio.Cells[aCol, aRow]) < aLimTen[1]) or
     (StrToFloat(SGR_TenVazio.Cells[aCol, aRow]) > aLimTen[0]) then begin
    SGR_TenVazio.Canvas.Font.Color := clWhite;
    SGR_TenVazio.Canvas.Brush.Color := clRed;
  end;
```

```
end;

procedure TFRM_PTracao.SGR_FlechaPrepareCanvas(sender: TObject; aCol,
  aRow: Integer; aState: TGridDrawState);
begin
  if (aRow = 9) and (aCol <> 0) then begin
    SGR_Flecha.Canvas.Brush.Color := clGray;
    SGR_Flecha.Canvas.Font.Color := clWhite;
  end;
  if (aRow = 9) or (aRow = 0) then
    SGR_Flecha.Canvas.Font.Bold := True;
end;

procedure TFRM_PTracao.SGR_EstriboPrepareCanvas(sender: TObject; aCol,
  aRow: Integer; aState: TGridDrawState);
begin
  if (aCol = 0) or (aCol = 2) then
    SGR_Estribo.Canvas.Brush.Color := clBtnFace;
  SGR_Estribo.GridLineColor := clGrayText;
end;

//Escolha do tipo de seção
procedure TFRM_PTracao.BBT_RetangularClick(Sender: TObject);
begin
  GridCleaner(SGR_Dimensao);
  SGR_Dimensao.Cells[1,2] := '-'; SGR_Dimensao.Cells[1,3] := '-';
SGR_Dimensao.Cells[1,4] := '-';
  SGR_Dimensao.Cells[1,5] := '-'; SGR_Dimensao.Cells[2,2] := '-';
SGR_Dimensao.Cells[2,3] := '-';
  SGR_Dimensao.Cells[2,4] := '-'; SGR_Dimensao.Cells[2,5] := '-';
  IMG_Secao.Picture.Bitmap.LoadFromResourceName(hInstance, 'RETANGULAR');
  IMG_SecaoArm.Picture.Bitmap.LoadFromResourceName(hinstance, 'RETANGULAR-
ARM');
  vCont := 1;
end;

procedure TFRM_PTracao.BBT_IClick(Sender: TObject);
begin
  GridCleaner(SGR_Dimensao);
  SGR_Dimensao.Cells[1,4] := '-'; SGR_Dimensao.Cells[1,5] := '-';
  IMG_Secao.Picture.Bitmap.LoadFromResourceName(hinstance, 'I');
  IMG_SecaoArm.Picture.Bitmap.LoadFromResourceName(hinstance, 'I-ARM');
  vCont := 2;
end;

procedure TFRM_PTracao.BBT_TClick(Sender: TObject);
begin
  GridCleaner(SGR_Dimensao);
  SGR_Dimensao.Cells[1,3] := '-'; SGR_Dimensao.Cells[1,4] := '-';
SGR_Dimensao.Cells[1,5] := '-';
  SGR_Dimensao.Cells[2,4] := '-'; SGR_Dimensao.Cells[2,5] := '-';
  IMG_Secao.Picture.Bitmap.LoadFromResourceName(hinstance,'T');
  IMG_SecaoArm.Picture.Bitmap.LoadFromResourceName(hinstance,'T-ARM');
  vCont := 3;
end;

procedure TFRM_PTracao.BBT_AbaClick(Sender: TObject);
begin
  GridCleaner(SGR_Dimensao);
  SGR_Dimensao.Cells[1,4] := '-'; SGR_Dimensao.Cells[1,5] := '-';
  SGR_Dimensao.Cells[2,4] := '-'; SGR_Dimensao.Cells[2,5] := '-';
```

```
  IMG_Secao.Picture.Bitmap.LoadFromResourceName(hinstance,'ABA');
  IMG_SecaoArm.Picture.Bitmap.LoadFromResourceName(hinstance,'ABA-ARM');
  vCont := 4;
end;

procedure TFRM_PTracao.BTN_DadosSCompostaClick(Sender: TObject);
begin
  USComposta.FRM_SComposta.ShowModal;
end;

procedure TFRM_PTracao.BTN_DataCoefClick(Sender: TObject);
begin
  FRM_DatasCoef.EDB_DfcPP.Text := EDB_fckj.Text;
  FRM_DatasCoef.EDB_DfcPerda.Text := EDB_fck.Text;
  FRM_DatasCoef.EDB_DfcPerda1.Text := FRM_SComposta.EDB_fckCapa.Text;
  uDatasCoef.FRM_DatasCoef.ShowModal;
end;

procedure TFRM_PTracao.BTN_MxLNClick(Sender: TObject);
begin
  UGrafico.FRM_GraficoMLN.ShowModal;
end;

//1. Evento clique do botão "Pre-dimensionamento" da aba "Características
básicas"
procedure TFRM_PTracao.BTN_PreDimClick(Sender: TObject);
var
  vTensaoFPre, vEpsp, vEpsT, vEps7, vEcEp, lEci: extended;
  vImX, vImY: integer;
  aVetorDesl: array of tPoints;
  aAp: array [0..13] of tPoints;
  t: tEstribo;
begin
  vFckj := StrToFloat(IsNil(EDB_fckj.Text));
  vFctj := -0.3 * power(vFckj, 2/3);
  vFck := StrToFloat(IsNil(EDB_fck.Text));
  vFctm := -0.3 * power(vFck, 2/3);
  vL := StrToFloat(IsNil(EDB_vao.Text));
  SetLength(aDimensoes, 4);
  if CBX_EcEp.Checked = False then begin
    EDB_rEcEp.Text := FormatFloat('###,###,##0.00', ElasticMod(vAlfaE,
fckCapa, 0, 28).X / ElasticMod(vAlfaE, vFck, 0, 28).X);
    if fckCapa = 0 then
      EDB_rEcEp.Text := '1.0';
  end;
  aDimensoes[0].X := vBfCapa;
  aDimensoes[0].Y := hCapa;
  vEcEp := StrToFloat(EDB_rEcEp.Text);
  //1.1. Cálculo das características geométricas de cada seção
  if vCont = 1 then
     begin
       SetLength(aVetor, 4);
       vB1 := StrToFloat(isNil(SGR_Dimensao.Cells[1,1])) / 100;
       vH1 := StrToFloat(isNil(SGR_Dimensao.Cells[2,1])) / 100;
       vHPre := vH1;
       vHt := vHPre + hLaje + hCapa;
       aDimensoes[1].X := vB1 - lLaje;
       if lLaje = 0 then
         aDimensoes[1].X := 0;
       aDimensoes[1].Y := hLaje;
       aDimensoes[2].X := vB1;
```

```
        aDimensoes[2].Y := vH1;
        aDimensoes[3].X := 0;
        aDimensoes[3].Y := 0;
        aVetor[0].X := 0;
        aVetor[0].Y := 0;
        aVetor[1].X := vB1;
        aVetor[1].Y := 0;
        aVetor[2].X := vB1;
        aVetor[2].Y := vH1;
        aVetor[3].X := 0;
        aVetor[3].Y := vH1;
        vB := vB1 * 100;
        vA := PolygonArea(4, aVetor);
        vACapa := aDimensoes[0].X * aDimensoes[0].Y + aDimensoes[1].X *
aDimensoes[1].Y;
        tCG := PolygonCentroid(4, aVetor, vA); //Coordenadas X e Y do
centroid da peça simples
        vIx := PolygonInertia(4, aVetor, tCG, vA);
        tWi.Y := vIx / tCG.Y;
        tWi.X := vIx / (vH1 - tCG.Y);
        tIxY := InerciaComposta((vB1 - lLaje)*vEcEp, hLaje, vBfCapa*vEcEp,
hCapa, vIx, tCG.Y, vH1, vB1, vA);
        tWf := SecaoCompostaW(tIxY.X, tIxY.Y, vH1, hLaje, hCapa);
        vUAr := PolygonPerimeter(aVetor) / 100 - vB1;
        SGR_Geom1.Cells[1,0] := FormatFloat('###,###,##0.000',vA);
        SGR_Geom1.Cells[3,0] := FormatFloat('###,###,##0.00000',vIx);
        SGR_Geom1.Cells[1,1] := FormatFloat('###,###,##0.00000',tWi.Y);
        SGR_Geom1.Cells[3,1] := FormatFloat('###,###,##0.00000',tWi.X);
        SGR_Geom1.Cells[1,2] := FormatFloat('###,###,##0.000',tCG.Y);
        SGR_Geom1.Cells[3,2] := FormatFloat('###,###,##0.000',vHPre-tCG.Y);
        SGR_Geom2.Cells[1,0] := FormatFloat('###,###,##0.000',vA+vACapa);
        SGR_Geom2.Cells[3,0] := FormatFloat('###,###,##0.00000',tIxY.X);
        SGR_Geom2.Cells[1,1] := FormatFloat('###,###,##0.00000',tWf.Y);
        SGR_Geom2.Cells[3,1] := FormatFloat('###,###,##0.00000',tWf.X);
        SGR_Geom2.Cells[1,2] := FormatFloat('###,###,##0.000',tIxY.Y);
        SGR_Geom2.Cells[3,2] := FormatFloat('###,###,##0.000',vHt-tIxY.Y);
        end
  else if vCont = 2 then
     begin
        SetLength(aVetor, 12);
        vB1 := StrToFloat(isNil(SGR_Dimensao.Cells[1,1])) / 100;
        vB2 := StrToFloat(isNil(SGR_Dimensao.Cells[1,2])) / 100;
        vB3 := StrToFloat(isNil(SGR_Dimensao.Cells[1,3])) / 100;
        vH1 := StrToFloat(isNil(SGR_Dimensao.Cells[2,1])) / 100;
        vH2 := StrToFloat(isNil(SGR_Dimensao.Cells[2,2])) / 100;
        vH3 := StrToFloat(isNil(SGR_Dimensao.Cells[2,3])) / 100;
        vH4 := StrToFloat(isNil(SGR_Dimensao.Cells[2,4])) / 100;
        vH5 := StrToFloat(isNil(SGR_Dimensao.Cells[2,5])) / 100;
        vHPre := vH1 + vH2 + vH3 + vH4 + vH5;
        vHt := vHPre + hLaje + hCapa;
        aDimensoes[1].X := vB1 - lLaje;
        if lLaje = 0 then
          aDimensoes[1].X := 0;
        aDimensoes[1].Y := hLaje;
        aDimensoes[2].X := vB1;
        aDImensoes[2].Y := vH1;
        aDimensoes[3].X := vB2;
        aDImensoes[3].Y := vH5 + vH2 + vH4;
        aVetor[0].X := 0;
        aVetor[0].Y := 0;
        aVetor[1].X := vB3;
```

```
        aVetor[1].Y := 0;
        aVetor[2].X := vB3;
        aVetor[2].Y := vH3;
        aVetor[3].X := (vB3 + vB2) / 2;
        aVetor[3].Y := vH3 + vH5;
        aVetor[4].X := (vB3 + vB2) / 2;
        aVetor[4].Y := vH3 + vH5 + vH2;
        aVetor[5].X := (vB3 + vB1) / 2;
        aVetor[5].Y := vH3 + vH5 + vH2 + vH4;
        aVetor[6].X := (vB3 + vB1) / 2;
        aVetor[6].Y := vH3 + vH5 + vH2 + vH4 + vH1;
        aVetor[7].X := (vB3 - vB1) / 2;
        aVetor[7].Y := vH3 + vH5 + vH2 + vH4 + vH1;
        aVetor[8].X := (vB3 - vB1) / 2;
        aVetor[8].Y := vH3 + vH5 + vH2 + vH4;
        aVetor[9].X := (vB3 - vB2) / 2;
        aVetor[9].Y := vH3 + vH5 + vH2;
        aVetor[10].X := (vB3 - vB2) / 2;
        aVetor[10].Y := vH3 + vH5;
        aVetor[11].X := 0;
        aVetor[11].Y := vH3;
        vB := vB3 * 100;
        vA := PolygonArea(12, aVetor);
        vACapa := aDimensoes[0].X * aDimensoes[0].Y + aDimensoes[1].X *
aDimensoes[1].Y;
        tCG := PolygonCentroid(12, aVetor, vA);
        vIx := PolygonInertia(12, aVetor, tCG, vA);
        tWi.X := vIx / tCG.Y;
        tWi.Y := vIx / (vHPre - tCG.Y);
        tIxY := InerciaComposta((vB1 - lLaje)*vEcEp, hLaje, vBfCapa*vEcEp,
hCapa, vIx, tCG.Y, vHt, vB1, vA);
        tWf := SecaoCompostaW(tIxY.X, tIxY.Y, vHt, hLaje, hCapa);
        vUAr := PolygonPerimeter(aVetor) / 100 - vB1;
        SGR_Geom1.Cells[1,0] := FormatFloat('###,###,##0.000',vA);
        SGR_Geom1.Cells[3,0] := FormatFloat('###,###,##0.00000',vIx);
        SGR_Geom1.Cells[1,1] := FormatFloat('###,###,##0.00000',tWi.Y);
        SGR_Geom1.Cells[3,1] := FormatFloat('###,###,##0.00000',tWi.X);
        SGR_Geom1.Cells[1,2] := FormatFloat('###,###,##0.000',tCG.Y);
        SGR_Geom1.Cells[3,2] := FormatFloat('###,###,##0.000',vHPre-tCG.Y);
        SGR_Geom2.Cells[1,0] := FormatFloat('###,###,##0.000',vA+vACapa);
        SGR_Geom2.Cells[3,0] := FormatFloat('###,###,##0.00000',tIxY.X);
        SGR_Geom2.Cells[1,1] := FormatFloat('###,###,##0.00000',tWf.Y);
        SGR_Geom2.Cells[3,1] := FormatFloat('###,###,##0.00000',tWf.X);
        SGR_Geom2.Cells[1,2] := FormatFloat('###,###,##0.000',tIxY.Y);
        SGR_Geom2.Cells[3,2] := FormatFloat('###,###,##0.000',vHt-tIxY.Y);
      end
  else if vCont = 3 then
      begin
        SetLength(aVetor, 8);
        vB1 := StrToFloat(isNil(SGR_Dimensao.Cells[1,1])) / 100;
        vB2 := StrToFloat(isNil(SGR_Dimensao.Cells[1,2])) / 100;
        vH1 := StrToFloat(isNil(SGR_Dimensao.Cells[2,1])) / 100;
        vH2 := StrToFloat(isNil(SGR_Dimensao.Cells[2,2])) / 100;
        vH3 := StrToFloat(isNil(SGR_Dimensao.Cells[2,3])) / 100;
        vHPre := vH1 + vH2 + vH3;
        vHt := vHPre + hLaje + hCapa;
        aDimensoes[1].X := vB1 - lLaje;
        if lLaje = 0 then
          aDimensoes[1].X := 0;
        aDimensoes[1].Y := hLaje;
        aDimensoes[2].X := vB1;
```

```
        aDImensoes[2].Y := vH3;
        aDimensoes[3].X := vB2;
        aDImensoes[3].Y := vH2 + vH3;
        aVetor[0].X := 0;
        aVetor[0].Y := 0;
        aVetor[1].X := vB2;
        aVetor[1].Y := 0;
        aVetor[2].X := vB2;
        aVetor[2].Y := vH2;
        aVetor[3].X := (vB2 + vB1) / 2;
        aVetor[3].Y := vH2 + vH3;
        aVetor[4].X := (vB2 + vB1) / 2;
        aVetor[4].Y := vH2 + vH3 + vH1;
        aVetor[5].X := (vB2 - vB1) / 2;
        aVetor[5].Y := vH2 + vH3 + vH1;
        aVetor[6].X := (vB2 - vB1) / 2;
        aVetor[6].Y := vH2 + vH3;
        aVetor[7].X := 0;
        aVetor[7].Y := vH2;
        vB := vB2 * 100;
        vA := PolygonArea(8, aVetor);
        vACapa := aDimensoes[0].X * aDimensoes[0].Y + aDimensoes[1].X *
aDimensoes[1].Y;
        tCG := PolygonCentroid(8, aVetor, vA);
        vIx := PolygonInertia(8, aVetor, tCG, vA);
        tWi.X := vIx / tCG.Y;
        tWi.Y := vIx / (vHPre - tCG.Y);
        tIxY := InerciaComposta((vB1 - lLaje)*vEcEp, hLaje, vBfCapa*vEcEp,
hCapa, vIx, tCG.Y, vHt, vB1, vA);
        tWf := SecaoCompostaW(tIxY.X, tIxY.Y, vHt, hLaje, hCapa);
        vUAr := PolygonPerimeter(aVetor) / 100 - vB1;
        SGR_Geom1.Cells[1,0] := FormatFloat('###,###,##0.000',vA);
        SGR_Geom1.Cells[3,0] := FormatFloat('###,###,##0.00000',vIx);
        SGR_Geom1.Cells[1,1] := FormatFloat('###,###,##0.00000',tWi.Y);
        SGR_Geom1.Cells[3,1] := FormatFloat('###,###,##0.00000',tWi.X);
        SGR_Geom1.Cells[1,2] := FormatFloat('###,###,##0.000',tCG.Y);
        SGR_Geom1.Cells[3,2] := FormatFloat('###,###,##0.000',vHPre-tCG.Y);
        SGR_Geom2.Cells[1,0] := FormatFloat('###,###,##0.000',vA+vACapa);
        SGR_Geom2.Cells[3,0] := FormatFloat('###,###,##0.00000',tIxY.X);
        SGR_Geom2.Cells[1,1] := FormatFloat('###,###,##0.00000',tWf.Y);
        SGR_Geom2.Cells[3,1] := FormatFloat('###,###,##0.00000',tWf.X);
        SGR_Geom2.Cells[1,2] := FormatFloat('###,###,##0.000',tIxY.Y);
        SGR_Geom2.Cells[3,2] := FormatFloat('###,###,##0.000',vHt-tIxY.Y);
      end
    else if vCont = 4 then
      begin
        SetLength(aVetor, 8);
        vB1 := StrToFloat(isNil(SGR_Dimensao.Cells[1,1])) / 100;
        vB2 := StrToFloat(isNil(SGR_Dimensao.Cells[1,2])) / 100;
        vB3 := StrToFloat(isNil(SGR_Dimensao.Cells[1,3])) / 100;
        vH1 := StrToFloat(isNil(SGR_Dimensao.Cells[2,1])) / 100;
        vH2 := StrToFloat(isNil(SGR_Dimensao.Cells[2,2])) / 100;
        vH3 := StrToFloat(isNil(SGR_Dimensao.Cells[2,3])) / 100;
        vHPre := vH3;
        vHt := vHPre + hCapa;
        aDimensoes[1].X := 0;
        aDimensoes[1].Y := 0;
        aDimensoes[2].X := vB3;
        aDimensoes[2].Y := vH3;
        aDimensoes[3].X := vB3;
        aDimensoes[3].Y := vH3;
```

```
        aVetor[0].X := 0;
        aVetor[0].Y := 0;
        aVetor[1].X := vB1 + vB2 + vB3;
        aVetor[1].Y := 0;
        aVetor[2].X := vB1 + vB2 + vB3;
        aVetor[2].Y := vH2;
        aVetor[3].X := vB1 + vB3;
        aVetor[3].Y := vH2;
        aVetor[4].X := vB1 + vB3;
        aVetor[4].Y := vH3;
        aVetor[5].X := vB1;
        aVetor[5].Y := vH3;
        aVetor[6].X := vB1;
        aVetor[6].Y := vH1;
        aVetor[7].X := 0;
        aVetor[7].Y := vH1;
        vB := (vB1 + vB2 + vB3) * 100;
        vA := PolygonArea(8, aVetor);
        vACapa := aDimensoes[0].X * aDimensoes[0].Y;
        tCG := PolygonCentroid(8, aVetor, vA);
        vIx := PolygonInertia(8, aVetor, tCG, vA);
        tWi.X := 2 * vIx / tCG.Y;
        tWi.Y := 2 * vIx / (vH3 - tCG.Y);
        tIxY := InerciaComposta(0, 0, vBfCapa*vEcEp, hCapa, vIx, tCG.Y, vH3,
vB3, vA);
        tWf := SecaoCompostaW(tIxY.X, tIxY.Y, vH3, 0, hCapa);
        vUAr := PolygonPerimeter(aVetor) / 100 - vB3;
        SGR_Geom1.Cells[1,0] := FormatFloat('###,###,##0.000',vA);
        SGR_Geom1.Cells[3,0] := FormatFloat('###,###,##0.00000',vIx);
        SGR_Geom1.Cells[1,1] := FormatFloat('###,###,##0.00000',tWi.Y);
        SGR_Geom1.Cells[3,1] := FormatFloat('###,###,##0.00000',tWi.X);
        SGR_Geom1.Cells[1,2] := FormatFloat('###,###,##0.000',tCG.Y);
        SGR_Geom1.Cells[3,2] := FormatFloat('###,###,##0.000',vHPre-tCG.Y);
        SGR_Geom2.Cells[1,0] := FormatFloat('###,###,##0.000',vA+vACapa);
        SGR_Geom2.Cells[3,0] := FormatFloat('###,###,##0.00000',tIxY.X);
        SGR_Geom2.Cells[1,1] := FormatFloat('###,###,##0.00000',tWf.Y);
        SGR_Geom2.Cells[3,1] := FormatFloat('###,###,##0.00000',tWf.X);
        SGR_Geom2.Cells[1,2] := FormatFloat('###,###,##0.000',tIxY.Y);
        SGR_Geom2.Cells[3,2] := FormatFloat('###,###,##0.000',vHt-tIxY.Y);
     end;

    //Verifica se todos os campos estão preenchidos
  if (EDB_vao.Text = '') or (EDB_fckj.Text = '') or (EDB_fck.Text = '') or
(EDB_psi2.Text = '')
  or (EDB_psi1.Text = '') or (EDB_psi2.Text = '') then begin
    ShowMessage('Ainda há campo/s obrigatório/s a serem preenchidos');
    Exit;
  end;

  //Verifica se o módulo de elasticidade e o fck estão preenchidos
  if (FRM_DatasCoef.EDB_DEciPerda.Text = '0') or
(FRM_DatasCoef.EDB_DEciPerda.Text = '') then begin
    ShowMessage('Verifique os campos no botão "Datas e coeficientes"');
    Exit;
  end;

    //Guarda os valores característicos do concreto em cada parte da seção
    aDimensoes[0].fck := fckCapa; aDimensoes[0].AlfaC := vAlfaCIL;
    aDimensoes[0].Lambida := vLambidaIL; aDimensoes[0].GamaC := vGamaCIL;
    aDimensoes[1].fck := fckCapa; aDimensoes[1].AlfaC := vAlfaCIL;
    aDimensoes[1].Lambida := vLambidaIL; aDimensoes[1].GamaC := vGamaCIL;
```

```
    aDimensoes[2].fck := vFck; aDimensoes[2].AlfaC := vAlfaC;
    aDimensoes[2].Lambida := vLambida; aDimensoes[2].GamaC := vGamaC;
    aDimensoes[3].fck := vFck; aDimensoes[3].AlfaC := vAlfaC;
    aDimensoes[3].Lambida := vLambida; aDimensoes[3].GamaC := vGamaC;
    aDimensoes[0].Ecs := aEcs[6].Y; aDimensoes[1].Ecs := aEcs[6].Y;
    aDimensoes[2].Ecs := aEcs[6].X; aDimensoes[3].Ecs := aEcs[6].X;

    //Analisa os valores do editbox
    if CBX_PP.Checked = False then
      EDB_PP.Text := FormatFloat('###,###,##0.0', vA * 25);
    if CBX_AlfaP1.Checked = False then
      EDB_AlfaP1.Text := FormatFloat('###,###,##0', vEp / aEci[0].X);
    if CBX_AlfaP2.Checked = False then
      EDB_AlfaP2.Text := FormatFloat('###,###,##0', vEp / aEci[6].X);
    if CBX_PMUar.Checked = False then
      EDB_PMUAr.Text := IntToStr(Round(vUAr * 100));
    if CBX_PMAc.Checked = False then
      EDB_PMAc.Text := IntToStr(Round(vA * 10000));
    if CBX_ILUar1.Checked = False then
      EDB_ILUAr.Text := IntToStr(Round(vBfCapa * 100));
    if CBX_ILAc1.Checked = False then
      EDB_ILAc.Text := IntToStr(Round(vACapa * 10000));

    //1.2. Desenho da seção transversal
    IMG_Secao2.Canvas.Erase;
    vImX := FRM_PTracao.IMG_Secao2.Width;
    vImY := FRM_PTracao.IMG_Secao2.Height;
    aVetorDesl := SecDispl(aVetor);
    scale := BestFit(aVetorDesl, vImX, vImY);
    vDesl := Displac(scale, aVetor);
    aVetorInt := VetorTransform(scale ,vImY, aVetorDesl);
    DrawSection(aVetorInt,IMG_Secao2,FRM_PTracao.Canvas.ClipRect);
    DrawReinforce(LBX_Linha.Count, vImY, vDesl, scale, IMG_Secao2,
LBX_Linha);

  //1.3. Cálculo do momento fletor máximo para cada carregamento
  vPP := StrToFloat(IsNil(EDB_PP.Text));
  vLaje := StrToFloat(IsNil(EDB_Laje.Text));
  vCapa := StrToFloat(IsNil(EDB_Capa.Text));
  vParede := StrToFloat(IsNil(EDB_Parede.Text));
  vRevestimento := StrToFloat(IsNil(EDB_Revestimento.Text));
  vQmax := StrToFloat(IsNil(EDB_qmax.Text));
  vQmin := StrToFloat(IsNil(EDB_qmin.Text));
  vPTotal := vPP + vLaje + vCapa;
  vP2Total := vParede + vRevestimento;
  aMomentos[0] := Momentos_Vao(vPP, vL);
  aMomentos[1] := Momentos_Vao(vLaje, vL);
  aMomentos[2] := Momentos_Vao(vCapa, vL);
  aMomentos[3] := Momentos_Vao(vParede, vL);
  aMomentos[4] := Momentos_Vao(vRevestimento, vL);
  aMomentos[5] := Momentos_Vao(vQmax, vL);
  aMomentos[6] := Momentos_Vao(vQmin, vL);
  aMomentos[7] := Momentos_Vao(vPTotal, vL);
  aMomentos[8] := Momentos_Vao(vP2Total, vL);
  aMd.S1:= aMomentos[0].S1*vMPP + aMomentos[1].S1*vMLaje +
aMomentos[2].S1*vMCapa +
           aMomentos[3].S1*vMParede + aMomentos[4].S1*vMRevest +
aMomentos[5].S1*vMAcid;
  aMd.S2:= aMomentos[0].S2*vMPP + aMomentos[1].S2*vMLaje +
aMomentos[2].S2*vMCapa +
```

```
          aMomentos[3].S2*vMParede + aMomentos[4].S2*vMRevest +
aMomentos[5].S2*vMAcid;
  aMd.S3:= aMomentos[0].S3*vMPP + aMomentos[1].S3*vMLaje +
aMomentos[2].S3*vMCapa +
          aMomentos[3].S3*vMParede + aMomentos[4].S3*vMRevest +
aMomentos[5].S3*vMAcid;
  aMd.S4:= aMomentos[0].S4*vMPP + aMomentos[1].S4*vMLaje +
aMomentos[2].S4*vMCapa +
          aMomentos[3].S4*vMParede + aMomentos[4].S4*vMRevest +
aMomentos[5].S4*vMAcid;
  aMd.S5:= aMomentos[0].S5*vMPP + aMomentos[1].S5*vMLaje +
aMomentos[2].S5*vMCapa +
          aMomentos[3].S5*vMParede + aMomentos[4].S5*vMRevest +
aMomentos[5].S5*vMAcid;
  vMd := aMd.S5;

  //1.3. Preenchimento do memo com as características normativas da
protensão
  case CBB_CAA.ItemIndex of
     0: begin
          MMO_RProtensao.Clear;
          MMO_RProtensao.Lines.Add('- Protensão Parcial');
          MMO_RProtensao.Lines.Add('- ELS-W <= 0,2 mm - Frequente');
          tPsi.X := StrToFloat(IsNil(EDB_psi1.Text));
          tPsi.Y := 0;
          MMO_RProtensao.Lines.Add('- a/c <= 0,60');
          MMO_RProtensao.Lines.Add('- Concreto >= C25');
          MMO_RProtensao.Lines.Add('- Cobrimento >= 30 mm');
        end;
     1: begin
          MMO_RProtensao.Clear;
          MMO_RProtensao.Lines.Add('- Protensão Limitada');
          MMO_RProtensao.Lines.Add('- ELS-D - Q. Perm.');
          MMO_RProtensao.Lines.Add('- ELS-F - Frequente');
          LBL_Combinacao1.Caption := '- ELS-F - Combinação frequente';
          LBL_Combinacao2.Caption := '- ELS-D - Combinação quase
permanente';
          LBL_Combinacao3.Caption := '- ELS-F - Combinação frequente';
          LBL_Combinacao4.Caption := '- ELS-D - Combinação quase
permanente';
          CHL_Serv1.Title := 'ELS-F,Qmáx';
          CHL_Serv2.Title := 'ELS-D,Qmáx';
          CHL_Serv3.Title := 'ELS-F,Qmín';
          CHL_Serv4.Title := 'ELS-D,Qmín';
          tPsi.X := StrToFloat(IsNil(EDB_psi1.Text));
          tPsi.Y := StrToFloat(IsNil(EDB_psi2.Text));
          aLimTen[3] := vFctm * vTFiss * 1000;
          aLimTen[4] := vFctm * vTDesc * 1000;
          LBL_LTracao1.Caption := 'Limite de tração: ' +
FormatFloat('###,###,##0', aLimTen[3]) + ' kN/m2';
          LBL_LTracao2.Caption := 'Limite de tração: ' +
FormatFloat('###,###,##0', aLimTen[4]) + ' kN/m2';
          LBL_LTracao3.Caption := 'Limite de tração: ' +
FormatFloat('###,###,##0', aLimTen[3]) + ' kN/m2';
          LBL_LTracao4.Caption := 'Limite de tração: ' +
FormatFloat('###,###,##0', aLimTen[4]) + ' kN/m2';
          MMO_RProtensao.Lines.Add('- a/c <= 0,55');
          MMO_RProtensao.Lines.Add('- Concreto >= C30');
          MMO_RProtensao.Lines.Add('- Cobrimento >= 35 mm');
        end;
     2: begin
```

```
          MMO_RProtensao.Clear;
          MMO_RProtensao.Lines.Add('- Protensão Completa');
          MMO_RProtensao.Lines.Add('- ELS-D - Frequente');
          MMO_RProtensao.Lines.Add('- ELS-F - Rara');
          LBL_Combinacao1.Caption := '- ELS-D - Combinação frequente';
          LBL_Combinacao2.Caption := '- ELS-F - Combinação rara';
          LBL_Combinacao3.Caption := '- ELS-D - Combinação frequente';
          LBL_Combinacao4.Caption := '- ELS-F - Combinação rara';
          CHL_Serv1.Title := 'ELS-D,Qmáx';
          CHL_Serv2.Title := 'ELS-F,Qmáx';
          CHL_Serv3.Title := 'ELS-D,Qmín';
          CHL_Serv4.Title := 'ELS-F,Qmín';
          tPsi.X := StrToFloat(IsNil(EDB_psi1.Text));
          tPsi.Y := StrToFloat(IsNil(EDB_psi3.Text));
          aLimTen[3] := vFctm * vTDesc * 1000;
          aLimTen[4] := vFctm * vTFiss * 1000;
          LBL_LTracao1.Caption := 'Limite de tração: ' +
FormatFloat('###,###,##0', aLimTen[3]) + ' kN/m2';
          LBL_LTracao2.Caption := 'Limite de tração: ' +
FormatFloat('###,###,##0', aLimTen[4]) + ' kN/m2';
          LBL_LTracao3.Caption := 'Limite de tração: ' +
FormatFloat('###,###,##0', aLimTen[3]) + ' kN/m2';
          LBL_LTracao4.Caption := 'Limite de tração: ' +
FormatFloat('###,###,##0', aLimTen[4]) + ' kN/m2';
          MMO_RProtensao.Lines.Add('- a/c <= 0,50');
          MMO_RProtensao.Lines.Add('- Concreto >= C35');
          MMO_RProtensao.Lines.Add('- Cobrimento >= 45 mm');
        end;
     3: begin
          MMO_RProtensao.Clear;
          MMO_RProtensao.Lines.Add('- Protensão Completa');
          MMO_RProtensao.Lines.Add('- ELS-D - Frequente');
          MMO_RProtensao.Lines.Add('- ELS-F - Rara');
          LBL_Combinacao1.Caption := '- ELS-D - Combinação frequente';
          LBL_Combinacao2.Caption := '- ELS-F - Combinação rara';
          LBL_Combinacao3.Caption := '- ELS-D - Combinação frequente';
          LBL_Combinacao4.Caption := '- ELS-F - Combinação rara';
          CHL_Serv1.Title := 'ELS-D,Qmáx';
          CHL_Serv2.Title := 'ELS-F,Qmáx';
          CHL_Serv3.Title := 'ELS-D,Qmín';
          CHL_Serv4.Title := 'ELS-F,Qmín';
          tPsi.X := StrToFloat(IsNil(EDB_psi1.Text));
          tPsi.Y := StrToFloat(IsNil(EDB_psi3.Text));
          aLimTen[3] := vFctm * vTDesc * 1000;
          aLimTen[4] := vFctm * vTFiss * 1000;
          LBL_LTracao1.Caption := 'Limite de tração: ' +
FormatFloat('###,###,##0', aLimTen[3]) + ' kN/m2';
          LBL_LTracao2.Caption := 'Limite de tração: ' +
FormatFloat('###,###,##0', aLimTen[4]) + ' kN/m2';
          LBL_LTracao3.Caption := 'Limite de tração: ' +
FormatFloat('###,###,##0', aLimTen[3]) + ' kN/m2';
          LBL_LTracao4.Caption := 'Limite de tração: ' +
FormatFloat('###,###,##0', aLimTen[4]) + ' kN/m2';
          MMO_RProtensao.Lines.Add('- a/c <= 0,45');
          MMO_RProtensao.Lines.Add('- Concreto >= C40');
          MMO_RProtensao.Lines.Add('- Cobrimento >= 55 mm');
        end;
  end;
  MMO_RProtensao.SelStart := 1;
```

```
  //Salvando os valores de gama f2 e atribuindo-os aos labels da aba
"Resultados"
  LBL_RGama.Caption := FloatToStr(tPsi.X);
  LBL_RGama1.Caption := FloatToStr(tPsi.Y);
  LBL_RGama2.Caption := FloatToStr(tPsi.X);
  LBL_RGama3.Caption := FloatToStr(tPsi.Y);
  aLimTen[0] := vFckj * vCVazio * 1000;
  aLimTen[1] := vFctj * vTVazio * 1000;
  aLimTen[2] := vFck * vCServ * 1000;
  LBL_LComp1.Caption := 'Limite de compressão: ' +
FormatFloat('###,###,##0', aLimTen[2]) + ' kN/m2';
  LBL_LComp2.Caption := 'Limite de compressão: ' +
FormatFloat('###,###,##0', aLimTen[2]) + ' kN/m2';
  LBL_LComp3.Caption := 'Limite de compressão: ' +
FormatFloat('###,###,##0', aLimTen[2]) + ' kN/m2';
  LBL_LComp4.Caption := 'Limite de compressão: ' +
FormatFloat('###,###,##0', aLimTen[2]) + ' kN/m2';
  LBL_LComp5.Caption := 'Limite de compressão: ' +
FormatFloat('###,###,##0', aLimTen[0]) + ' kN/m2';
  LBL_LTracao5.Caption := 'Limite de tração: ' + FormatFloat('###,###,##0',
aLimTen[1]) + ' kN/m2';

  //Pré-dimensionamento da armadura
  vTParRet := vAlfaC * vFck * 1000 / vGamaC;
  vTensaoFPre := GetITension(vFptd * vGamaS, vFpyd * vGamaS) * 0.75 / 10;
  UGrafico.FRM_GraficoMLN.CHL_Points.Clear;
  tAp := GetEps(vMd, vHt-0.04, vXD, aDimensoes,
UGrafico.FRM_GraficoMLN.CHL_Points);
  if tAp.KMD >= 0.5 then
    exit;
  vEpsp := PrestressedDeformation(vFpyd, vEpsyd, vFptd, vEpsu, vEp,
vTensaoFPre);
  vEpsT := vEpsp + tAp.Epss;
  vgSigmaSD := PrestressedTension(vFpyd, vEpsyd, vFptd, vEpsu, vEp, vEpsT)
/ 10;
  tAp.Ap := GetAp(vHt-0.04, vgSigmaSD, tAp);
  SGR_Dimens.Cells[1,1] := FormatFloat('###,###,##0.00', tAp.Ap);
  SGR_Dimens.Cells[1,2] := FormatFloat('###,###,##0.00', vHt - 0.04);
  SGR_Dimens.Cells[1,3] := FormatFloat('###,###,##0.00', tAp.X / (vHt -
0.04));

  aAp[0] := GetInfAp(0, 140, 0, 0, tCG.Y-0.04, 0, vA, tWi.X, tWf.X,
aMomentos[0].S5, 0, 0, 0, aLimTen[0]);
  aAp[1] := GetSupAp(0, 140, 0, 0, tCG.Y-0.04, 0, vA, tWi.Y, tWf.Y,
aMomentos[0].S5, 0, 0, 0, aLimTen[0]);
  aAp[2] := GetInfAp(0, 140, 0, 0, tCG.Y-0.04, 0, vA, tWi.X, tWf.X,
aMomentos[0].S5, 0, 0, 0, aLimTen[1]);
  aAp[3] := GetSupAp(0, 140, 0, 0, tCG.Y-0.04, 0, vA, tWi.Y, tWf.Y,
aMomentos[0].S5, 0, 0, 0, aLimTen[1]);
  aAp[4] := GetInfAp(0, 140, 0, 0, tCG.Y-0.04, 0, vA, tWi.X, tWf.X,
aMomentos[0].S5, 0, 0, 0, 0);
  aAp[5] := GetSupAp(0, 140, 0, 0, tCG.Y-0.04, 0, vA, tWi.Y, tWf.Y,
aMomentos[0].S5, 0, 0, 0, 0);
  aAp[6] := GetInfAp(0, 110.5, 0, 0, tCG.Y-0.04, 0, vA, tWi.X, tWf.X,
aMomentos[7].S5, aMomentos[8].S5, aMomentos[5].S5, tPsi.X, aLimTen[3]);
  aAp[7] := GetSupAp(0, 110.5, 0, 0, tCG.Y-0.04, 0, vA, tWi.Y, tWf.Y,
aMomentos[7].S5, aMomentos[8].S5, aMomentos[5].S5, tPsi.X, aLimTen[3]);
  aAp[8] := GetInfAp(0, 110.5, 0, 0, tCG.Y-0.04, 0, vA, tWi.X, tWf.X,
aMomentos[7].S5, aMomentos[8].S5, aMomentos[5].S5, tPsi.Y, aLimTen[4]);
  aAp[9] := GetSupAp(0, 110.5, 0, 0, tCG.Y-0.04, 0, vA, tWi.Y, tWf.Y,
aMomentos[7].S5, aMomentos[8].S5, aMomentos[5].S5, tPsi.Y, aLimTen[4]);
```

```
  aAp[10] := GetInfAp(0, 110.5, 0, 0, tCG.Y-0.04, 0, vA, tWi.X, tWf.X,
aMomentos[7].S5, aMomentos[8].S5, 0, 0, aLimTen[3]);
  aAp[11] := GetSupAp(0, 110.5, 0, 0, tCG.Y-0.04, 0, vA, tWi.Y, tWf.Y,
aMomentos[7].S5, aMomentos[8].S5, 0, 0, aLimTen[3]);
  aAp[12] := GetInfAp(0, 110.5, 0, 0, tCG.Y-0.04, 0, vA, tWi.X, tWf.X,
aMomentos[7].S5, aMomentos[8].S5, 0, 0, aLimTen[4]);
  aAp[13] := GetSupAp(0, 110.5, 0, 0, tCG.Y-0.04, 0, vA, tWi.Y, tWf.Y,
aMomentos[7].S5, aMomentos[8].S5, 0, 0, aLimTen[4]);
  if (aAp[1].Y = 0) or (aAp[3].Y = 0) or (aAp[5].Y = 0) or (aAp[7].Y = 0)
or
     (aAp[9].Y = 0) or (aAp[11].Y = 0) or (aAp[13].Y = 0) then
       ShowMessage('A excentricidade está muito alta! O gráfico com os
intervalos de "Ap" pode não funcionar corretamente!');

  CHL_Vazio1.Clear;
  CHL_Vazio2.Clear;
  CHL_Vazio3.Clear;
  CHL_Serv1.Clear;
  CHL_Serv2.Clear;
  CHL_Serv3.Clear;
  CHL_Serv4.Clear;
  CHL_Ap.Clear;
  CHL_ApUtil.Clear;

  CHL_Vazio1.AddXY(DrawGraphic(aAp[0], aAp[1]).X, 1);
  CHL_Vazio1.AddXY(DrawGraphic(aAp[0], aAp[1]).Y, 1);
  CHL_Vazio2.AddXY(DrawGraphic(aAp[2], aAp[3]).X, 2);
  CHL_Vazio2.AddXY(DrawGraphic(aAp[2], aAp[3]).Y, 2);
  CHL_Vazio3.AddXY(DrawGraphic(aAp[4], aAp[5]).X, 3);
  CHL_Vazio3.AddXY(DrawGraphic(aAp[4], aAp[5]).Y, 3);
  CHL_Serv1.AddXY(DrawGraphic(aAp[6], aAp[7]).X, 4);
  CHL_Serv1.AddXY(DrawGraphic(aAp[6], aAp[7]).Y, 4);
  CHL_Serv2.AddXY(DrawGraphic(aAp[8], aAp[9]).X, 5);
  CHL_Serv2.AddXY(DrawGraphic(aAp[8], aAp[9]).Y, 5);
  CHL_Serv3.AddXY(DrawGraphic(aAp[10], aAp[11]).X, 6);
  CHL_Serv3.AddXY(DrawGraphic(aAp[10], aAp[11]).Y, 6);
  CHL_Serv4.AddXY(DrawGraphic(aAp[12], aAp[13]).X, 7);
  CHL_Serv4.AddXY(DrawGraphic(aAp[12], aAp[13]).Y, 7);
  CHL_Ap.AddXY(tAp.Ap, 0);
  CHL_Ap.AddXY(tAp.Ap, 8);
end;

                    {ABA DIMENSIONAMENTO}
procedure TFRM_PTracao.EDB_S1KeyUp(Sender: TObject; var Key: Word;
  Shift: TShiftState);
begin
  if not isNeg(TEdit(Sender).Text) then begin
    TEdit(Sender).Text := '';
    ShowMessage('Algarismo inválido!');
    Exit;
  end;
end;

//Definição das características da armadura
procedure TFRM_PTracao.CBB_TBarraChange(Sender: TObject);
begin
  if CBB_TBarra.ItemIndex = 1 then
   begin
     CBB_DBarra.Clear;
     CBB_TAco.Clear;
     CBB_Taco.Items.Strings[0] := '175';
```

```
      CBB_Taco.Items.Strings[1] := '190';
      CBB_TAco.ShowHint := True;
      CBB_TAco.Hint := 'Significado do código: Aço p/ protensão; tensão de
ruptura em kN/cm².';
    end
   else if CBB_TBarra.ItemIndex = 2 then
    begin
      CBB_DBarra.Clear;
      CBB_TAco.Clear;
      CBB_Taco.Items.Strings[0] := '25';
      CBB_Taco.Items.Strings[1] := '50';
      CBB_Taco.Items.Strings[2] := '60';
      CBB_TAco.ShowHint := True;
      CBB_TAco.Hint := 'Significado do código: Aço p/ concreto armado;
tensão de escoamento em kN/cm².';
    end
   else
    begin
      CBB_DBarra.Clear;
      CBB_TAco.Clear;
      CBB_Taco.ShowHint := False;
    end;
end;

procedure TFRM_PTracao.CBB_TAcoChange(Sender: TObject);
begin
   if (CBB_TBarra.ItemIndex = 1) and (CBB_TAco.ItemIndex = 0) then
    //Aço CP175
    begin
      CBB_DBarra.Clear;
      CBB_Dbarra.Items.Strings[0] := '4.0 (1 fio)';
      CBB_Dbarra.Items.Strings[1] := '5.0 (1 fio)';
      CBB_Dbarra.Items.Strings[2] := '6.0 (1 fio)';
    end
   else if (CBB_TBarra.ItemIndex = 1) and (CBB_TAco.ItemIndex = 1) then
    //Aço CP190
    begin
      CBB_DBarra.Clear;
      CBB_Dbarra.Items.Strings[0] := '6.5 (3 fios)';
      CBB_Dbarra.Items.Strings[1] := '7.6 (3 fios)';
      CBB_Dbarra.Items.Strings[2] := '8.8 (3 fios)';
      CBB_Dbarra.Items.Strings[3] := '9.6 (3 fios)';
      CBB_Dbarra.Items.Strings[4] := '11.1 (3 fios)';
      CBB_Dbarra.Items.Strings[5] := '9.5 (7 fios)';
      CBB_Dbarra.Items.Strings[6] := '12.7 (7 fios)';
      CBB_Dbarra.Items.Strings[7] := '15.2 (7 fios)';
    end
   else if (CBB_TBarra.ItemIndex = 2) and ((CBB_TAco.ItemIndex = 0) or
(CBB_TAco.ItemIndex = 1)) then
    begin
      //Aço CA25 e CA50
      CBB_DBarra.Clear;
      CBB_Dbarra.Items.Strings[0] := '6.3';
      CBB_Dbarra.Items.Strings[1] := '8.0';
      CBB_Dbarra.Items.Strings[2] := '10.0';
      CBB_Dbarra.Items.Strings[3] := '12.5';
      CBB_Dbarra.Items.Strings[4] := '16.0';
      CBB_Dbarra.Items.Strings[5] := '20.0';
      CBB_Dbarra.Items.Strings[6] := '25.0';
      CBB_Dbarra.Items.Strings[7] := '32.0';
      CBB_Dbarra.Items.Strings[8] := '40.0';
```

```
   end
  else if (CBB_TBarra.ItemIndex = 2) and (CBB_TAco.ItemIndex = 2) then
   //Aço CA60
   begin
     CBB_DBarra.Clear;
     CBB_Dbarra.Items.Strings[0] := '4.2';
     CBB_Dbarra.Items.Strings[1] := '5.0';
     CBB_Dbarra.Items.Strings[2] := '6.0';
     CBB_Dbarra.Items.Strings[3] := '7.0';
     CBB_Dbarra.Items.Strings[4] := '8.0';
     CBB_Dbarra.Items.Strings[5] := '9.5';
   end
end;

//Salvando a área da seção das barras e cordoalhas
procedure TFRM_PTracao.CBB_DBarraChange(Sender: TObject);
begin
  if ((CBB_TBarra.ItemIndex = 1) and (CBB_TAco.ItemIndex = 0)) then begin
    //Aço CP175
    case CBB_DBarra.ItemIndex of
       0: vABarra := 0.126; //fi = 4,0 mm
       1: vABarra := 0.196; //fi = 5,0 mm
       2: vABarra := 0.283; //fi = 6,0 mm
    end;
  end
  else if ((CBB_TBarra.ItemIndex = 1) and (CBB_TAco.ItemIndex = 1)) then
begin
    //Aço CP190
    case CBB_DBarra.ItemIndex of
       0: vABarra := 0.215; //fi = 6,5 mm
       1: vABarra := 0.300; //fi = 7,6 mm
       2: vABarra := 0.376; //fi = 8,8 mm
       3: vABarra := 0.462; //fi = 9,6 mm
       4: vABarra := 0.657; //fi = 11,1 mm
       5: vABarra := 0.555; //fi = 9,5 mm
       6: vABarra := 1.000; //fi = 12,7 mm
       7: vABarra := 1.435; //fi = 15,2 mm
    end;
  end
  else if ((CBB_TBarra.ItemIndex = 2) and ((CBB_TAco.ItemIndex = 0) or
(CBB_TAco.ItemIndex = 1))) then begin
    //Aço CA25 e CA50
    case CBB_DBarra.ItemIndex of
       0: vABarra := 0.312; //fi = 6,3 mm
       1: vABarra := 0.503; //fi = 8,0 mm
       2: vABarra := 0.785; //fi = 10,0 mm
       3: vABarra := 1.227; //fi = 12,5 mm
       4: vABarra := 2.011; //fi = 16,0 mm
       5: vABarra := 3.142; //fi = 20,0 mm
       6: vABarra := 4.909; //fi = 25,0 mm
       7: vABarra := 8.042; //fi = 32,0 mm
       8: vABarra := 12.566; //fi = 40,0 mm
    end;
  end
  else if ((CBB_TBarra.ItemIndex = 2) and (CBB_TAco.ItemIndex = 2)) then
begin
    //Aço CA60
    case CBB_DBarra.ItemIndex of
       0: vABarra := 0.139; //fi = 4,2 mm
       1: vABarra := 0.196; //fi = 5,0 mm
       2: vABarra := 0.283; //fi = 6,0 mm
```

```
       3: vABarra := 0.385; //fi = 7,0 mm
       4: vABarra := 0.503; //fi = 8,0 mm
       5: vABarra := 0.709; //fi = 9,5 mm
    end;
  end
end;

//Salvando as características da armadura no ListBox
  //Atribuindo os dados de entrada nas variáveis e criando uma linha no
listbox
procedure TFRM_PTracao.TBX_AddClick(Sender: TObject);
var
  Ver_Arm: tDados;
  Barra: string;
  i, j, vImX, vImY: integer;
  aVetorDesl: array of tPoints;
begin
  if ((EDB_NBarra.Text = '') or (CBB_PBarra.ItemIndex = -1) or
(CBB_TBarra.ItemIndex = -1)
  or (CBB_TAco.ItemIndex = -1) or (CBB_DBarra.ItemIndex = -1) or
(EDB_Xa.Text = '')
  or (EDB_Ya.Text = '') or (EDB_Xb.Text = '') or (EDB_Yb.Text = '')) then
begin
    ShowMessage('Há campos a serem preenchidos!');
    Exit
  end;
  CBB_DBarra.OnChange(Ver_Arm);
  begin
    //Verifica se a armadura ativa selecionada já existe
    i := CBB_TBarra.ItemIndex;
    Barra := CBB_DBarra.Text;
    if i = 1 then
      For j := 0 to LBX_Linha.Count - 1 do begin
        Ver_Arm := tDados(LBX_Linha.Items.Objects[j]);
          if Ver_Arm.vTBarra = 1 then
            if Ver_Arm.vDBarra <> Barra then begin
              ShowMessage('A armadura ativa selecionada é diferente da
existente. O programa trabalha com o princípio de que apenas um tipo de
armadura ativa será utilizado.');
              Exit
            end;
      end;
    //Cria um objeto na ListBox com os dados da armadura
    Ver_Arm := tDados.Create;
    Ver_Arm.vNBarra := StrToInt(EDB_NBarra.Text);
    Ver_Arm.vPBarra := CBB_PBarra.ItemIndex;
    Ver_Arm.vTBarra := CBB_TBarra.ItemIndex;
    Ver_Arm.vTAco := CBB_TAco.Text;
    Ver_Arm.vTAco_index := CBB_TAco.ItemIndex;
    Ver_Arm.vDBarra := CBB_DBarra.Text;
    Ver_Arm.vDBarra_index := CBB_DBarra.ItemIndex;
    Ver_Arm.vXa := StrToFloat(EDB_Xa.Text);
    Ver_Arm.vYa := StrToFloat(EDB_Ya.Text);
    Ver_Arm.vXb := StrToFloat(EDB_Xb.Text);
    Ver_Arm.vYb := StrToFloat(EDB_Yb.Text);
    Ver_Arm.vABarra := vABarra;
    if Ver_Arm.vTBarra = 1 then begin
      LBX_Linha.AddItem('Ap ' + FloatToStr(n), Ver_Arm);
      n := n + 1;
    end;
    if Ver_Arm.vTBarra = 2 then begin
```

```
      LBX_Linha.AddItem('As ' + FloatToStr(m), Ver_Arm);
      m := m + 1;
    end;
  end;
  vImX := FRM_PTracao.IMG_Secao2.Width;
  vImY := FRM_PTracao.IMG_Secao2.Height;
  DrawReinforce(LBX_Linha.Count, vImY, vDesl, scale, IMG_Secao2,
LBX_Linha);
end;

  //Excluindo uma linha de dados do listbox
procedure TFRM_PTracao.TBX_ExcluirClick(Sender: TObject);
var
  Ver_Arm: tDados;
  i, j: integer;
  vImX, vImY: integer;
  aVetorDesl: array of tPoints;
begin
    //Apaga a linha selecionada e renumera os itens restantes
  n := 1;
  m := 1;
  i := LBX_Linha.Itemindex;
  LBX_Linha.Items.Delete(i);
  For j := 0 to LBX_Linha.Count - 1 do begin
    Ver_Arm := tDados(LBX_Linha.Items.Objects[j]);
    if Ver_Arm.vTBarra = 1 then begin
      LBX_Linha.Items.Strings[j] := 'Ap ' + FloatToStr(n);
      n := n + 1;
    end
    else if Ver_Arm.vTBarra = 2 then begin
      LBX_Linha.Items.Strings[j] := 'As ' + FloatToStr(m);
      m := m + 1;
    end;
  end;
    //Ajusta a numeração da armadura (quando for adicionar uma nova
armadura, o número segue a ordem)
  n := 0;
  m := 0;
  For j := 0 to LBX_Linha.Count - 1 do begin
    Ver_Arm := tDados(LBX_Linha.Items.Objects[j]);
    if Ver_Arm.vTBarra = 1 then
      n := n + 1
    else if Ver_Arm.vTBarra = 2 then
      m := m + 1
  end;
  if n = 0 then
    n := 1
  else
    n := n + 1;
  if m = 0 then
    m := 1
  else
    m := m + 1;
  IMG_Secao2.Canvas.Erase;
  vImX := FRM_PTracao.IMG_Secao2.Width;
  vImY := FRM_PTracao.IMG_Secao2.Height;
  aVetorDesl := SecDispl(aVetor);
  aVetorInt := VetorTransform(scale ,vImY, aVetorDesl);
  DrawSection(aVetorInt,IMG_Secao2,FRM_PTracao.Canvas.ClipRect);
  DrawReinforce(LBX_Linha.Count, vImY, vDesl, scale, IMG_Secao2,
LBX_Linha);
```

```
end;

  //Demonstrando quais os dados inseridos em uma linha do listbox
procedure TFRM_PTracao.LBX_LinhaClick(Sender: TObject);
var
  Ver_Arm: tDados;
  i: Integer;
begin
  i := LBX_Linha.ItemIndex;
  Ver_Arm := tDados(LBX_Linha.Items.Objects[i]);
  EDB_NBarra.Text := FloatToStr(Ver_Arm.vNBarra);
  CBB_PBarra.ItemIndex := Ver_Arm.vPBarra;
  CBB_TBarra.ItemIndex := Ver_Arm.vTBarra;
  CBB_TBarraChange(Sender);
  CBB_TAco.ItemIndex := Ver_Arm.vTAco_index;
  CBB_TAcoChange(Sender);
  CBB_DBarra.ItemIndex := Ver_Arm.vDBarra_index;
  EDB_Xa.Text := FloatToStr(Ver_Arm.vXa);
  EDB_Ya.Text := FloatToStr(Ver_Arm.vYa);
  EDB_Xb.Text := FloatToStr(Ver_Arm.vXb);
  EDB_Yb.Text := FloatToStr(Ver_Arm.vYb);
end;

  //Edição dos objetos da listbox
procedure TFRM_PTracao.TBX_EditClick(Sender: TObject);
var
  Ver_Arm: tDados;
  i, j, k: Integer;
  Barra: String;
  vImX, vImY: integer;
  aVetorDesl: array of tPoints;
begin
  if ((EDB_NBarra.Text = '') or (CBB_PBarra.ItemIndex = -1) or
(CBB_TBarra.ItemIndex = -1)
  or (CBB_TAco.ItemIndex = -1) or (CBB_DBarra.ItemIndex = -1) or
(EDB_Xa.Text = '')
  or (EDB_Ya.Text = '') or (EDB_Xb.Text = '') or (EDB_Yb.Text = '')) then
begin
    ShowMessage('Há campos a serem preenchidos!');
    Exit
  end;
  CBB_DBarra.OnChange(Ver_Arm);
  //Verifica se a armadura ativa selecionada já existe
  i := CBB_TBarra.ItemIndex;
  Barra := CBB_DBarra.Text;
  For j := 0 to LBX_Linha.Count - 1 do begin
    Ver_Arm := tDados(LBX_Linha.Items.Objects[j]);
    if Ver_Arm.vTBarra = 1 then
      k := k + 1;
  end;
  if (i = 1) and (k >= 2) then
    For j := 0 to LBX_Linha.Count - 1 do begin
      Ver_Arm := tDados(LBX_Linha.Items.Objects[j]);
        if Ver_Arm.vTBarra = 1 then
          if Ver_Arm.vDBarra <> Barra then begin
            ShowMessage('A armadura ativa selecionada é diferente da
existente. O programa trabalha com o princípio de que apenas um tipo de
armadura ativa será utilizado.');
            Exit
          end;
    end;
```

```
  //Inserção dos novos dados no objeto selecionado
  i := LBX_Linha.ItemIndex;
  if i >= 0 then begin
    Ver_Arm := tDados(LBX_Linha.Items.Objects[i]);
    Ver_Arm.vNBarra := StrToInt(EDB_NBarra.Text);
    Ver_Arm.vPBarra := CBB_PBarra.ItemIndex;
    Ver_Arm.vTBarra := CBB_TBarra.ItemIndex;
    Ver_Arm.vTAco := CBB_TAco.Text;
    Ver_Arm.vTAco_index := CBB_TAco.ItemIndex;
    Ver_Arm.vDBarra := CBB_DBarra.Text;
    Ver_Arm.vDBarra_index := CBB_DBarra.ItemIndex;
    Ver_Arm.vXa := StrToFloat(EDB_Xa.Text);
    Ver_Arm.vYa := StrToFloat(EDB_Ya.Text);
    Ver_Arm.vXb := StrToFloat(EDB_Xb.Text);
    Ver_Arm.vYb := StrToFloat(EDB_Yb.Text);
    Ver_Arm.vABarra := vABarra;
    LBX_Linha.Items.Objects[i] := Ver_Arm;
    n := 1;
    m := 1;
    For j := 0 to LBX_Linha.Count - 1 do begin
    Ver_Arm := tDados(LBX_Linha.Items.Objects[j]);
    if Ver_Arm.vTBarra = 1 then begin
      LBX_Linha.Items.Strings[j] := 'Ap ' + FloatToStr(n);
      n := n + 1;
    end
    else if Ver_Arm.vTBarra = 2 then begin
      LBX_Linha.Items.Strings[j] := 'As ' + FloatToStr(m);
      m := m + 1;
    end;
  end;
  end;
  IMG_Secao2.Canvas.Erase;
  vImX := FRM_PTracao.IMG_Secao2.Width;
  vImY := FRM_PTracao.IMG_Secao2.Height;
  aVetorDesl := SecDispl(aVetor);
  aVetorInt := VetorTransform(scale ,vImY, aVetorDesl);
  DrawSection(aVetorInt,IMG_Secao2,FRM_PTracao.Canvas.ClipRect);
  DrawReinforce(LBX_Linha.Count, vImY, vDesl, scale, IMG_Secao2,
LBX_Linha);
end;

//Liberar os check-box da vista longitudinal da armadura
procedure TFRM_PTracao.CBX_S1Change(Sender: TObject);
begin
  if CBX_S1.Checked = True then
    begin
      CBX_S2.Enabled := True;
      CBX_S2.ShowHint := True;
      EDB_S1.Enabled := True;
      EDB_S1.ShowHint := True;
    end
  else
    begin
      CBX_S2.Checked := False;
      CBX_S2.Enabled := False;
      CBX_S2.ShowHint := False;
      EDB_S1.Text := '';
      EDB_S1.Enabled := False;
      EDB_S1.ShowHint := True;
    end;
end;
```

```
procedure TFRM_PTracao.CBX_S2Change(Sender: TObject);
begin
  if CBX_S2.Checked = True then
    begin
      CBX_S3.Enabled := True;
      CBX_S3.ShowHint := True;
      EDB_S2.Enabled := True;
      EDB_S2.ShowHint := True;
    end
  else
    begin
      CBX_S3.Checked := False;
      CBX_S3.Enabled := False;
      CBX_S3.ShowHint := False;
      EDB_S2.Text := '';
      EDB_S2.Enabled := False;
      EDB_S2.ShowHint := True;
    end;
end;

procedure TFRM_PTracao.CBX_S3Change(Sender: TObject);
begin
  if CBX_S3.Checked = True then
    begin
      CBX_S4.Enabled := True;
      CBX_S4.ShowHint := True;
      EDB_S3.Enabled := True;
      EDB_S3.ShowHint := True;
    end
  else
    begin
      CBX_S4.Checked := False;
      CBX_S4.Enabled := False;
      CBX_S4.ShowHint := False;
      EDB_S3.Text := '';
      EDB_S3.Enabled := False;
      EDB_S3.ShowHint := True;
    end;
end;

procedure TFRM_PTracao.CBX_S4Change(Sender: TObject);
begin
  if CBX_S4.Checked = True then
    begin
      EDB_S4.Enabled := True;
      EDB_S4.ShowHint := True;
    end
  else
    begin
      EDB_S4.Text := '';
      EDB_S4.Enabled := False;
      EDB_S4.ShowHint := False;
    end;
end;

//Evento clique do botão "Dimensionar" da aba "Dimensionamento"
procedure TFRM_PTracao.BTN_DimensionarClick(Sender: TObject);
var
  j, k: integer;
```

```
  vEpsp, vEpsT, vEps7, vSigmaSd, vSigmaS2, vAAp, vAAs, vFpFs, lEci:
extended;
  vSigma, vSigmaS, vSigmaF, vSigmaI, vACord: extended;
  lTCordI, lTCordS, lPIniI, lPFinI, lPIniS, lPFinS, lEi, lEs, lNCordS5:
extended;
  tArm: tArmadura;
  Ver_Arm: tDados;
  aD: tPoints;
  aAp: array [0..13] of tPoints;
begin
  lEi := (GetExcentricity_Pos(LBX_Linha.Count, LBX_Linha, tCG)) / 100;
  lEs := (GetExcentricity_Neg(LBX_Linha.Count, LBX_Linha, tCG)) / 100;
  vAAp := GetApArea(LBX_Linha.Count, LBX_Linha, tCG).X;
  vAAs := GetAsArea(LBX_Linha.Count, LBX_Linha, tCG).X;
  lEci := StrToFloat(isNil(FRM_DatasCoef.EDB_DEciPerda.Text));
  vSigma := StrToFloat(IsNil(EDB_TensaoIniI.Text)) / 10;
  vSigmaS := StrToFloat(IsNil(EDB_TensaoIniS.Text)) / 10;
  vSigmaF := vSigma * (1 - StrToFloat(IsNil(EDB_PerdaFinI.Text)) / 100);
  vSigmaI := vSigma * (1 - StrToFloat(IsNil(EDB_PerdaIniI.Text)) / 100);
  vSigmaS2 := vSigmaS * (1 - StrToFloat(IsNil(EDB_PerdaFinS.Text)) / 100);
  aD := GetD(LBX_Linha.Count, LBX_Linha, tCG, vHt, vSigmaF, vSigmaS2);
  tAp := GetEps(vMd, aD.X, vXD, aDimensoes,
UGrafico.FRM_GraficoMLN.CHL_Points);
  if tAp.KMD >= 0.5 then
    exit;
  vEpsp := PrestressedDeformation(vFpyd, vEpsyd, vFptd, vEpsu, vEp,
vSigmaF);
  vEps7 := DescompDef(vSigmaI*vAAp, lEi, vA, tIxY.X, lEci);
  vEpsT := vEpsp + tAp.Epss + vEps7;
  vSigmaSD := PrestressedTension(vFpyd, vEpsyd, vFptd, vEpsu, vEp, vEpsT) /
10;
  aD := GetD(LBX_Linha.Count, LBX_Linha, tCG, vHt, vSigmaSD, vSigmaS2);
  if (vAAp <> 0) and (vAAs <> 0) then
    k := 0;
    while abs (aD.X - tAp.d) > 0.001 do begin
      tAp := GetEps(vMd, aD.X, vXD, aDimensoes,
UGrafico.FRM_GraficoMLN.CHL_Points);
      if tAp.KMD >= 0.5 then
        exit;
      vEpsp := PrestressedDeformation(vFpyd, vEpsyd, vFptd, vEpsu, vEp,
vSigmaF);
      vEps7 := DescompDef(vSigmaI*vAAp, lEi, vA, tIxY.X, lEci);
      vEpsT := vEpsp + tAp.Epss + vEps7;
      vSigmaSD := PrestressedTension(vFpyd, vEpsyd, vFptd, vEpsu, vEp,
vEpsT) / 10;
      aD := GetD(LBX_Linha.Count, LBX_Linha, tCG, vHt, vSigmaSD, vSigmaS2);
      k := k + 1;
    end;
  tAp.Ap := GetAp(aD.X, vSigmaSD, tAp);
  vFpFs := AsConvert(vMd, tAp.KZ, aD.X, vGamaS, vSigmaSD, LBX_Linha);
  SGR_Dimens.Cells[3,1] := FormatFloat('###,###,##0.00', tAp.Ap);
  SGR_Dimens.Cells[3,2] := FormatFloat('###,###,##0.00', aD.X);
  SGR_Dimens.Cells[3,3] := FormatFloat('###,###,##0.00', aD.Y);
  SGR_Dimens.Cells[1,5] := FormatFloat('###,###,##0.00', vAAp);
  SGR_Dimens.Cells[1,6] := FormatFloat('###,###,##0.00', vAAs);
  SGR_Dimens.Cells[3,5] := IntToStr(Round(vFpFs));
  SGR_Dimens.Cells[3,6] := IntToStr(Round(vMd / (tAp.KZ * aD.X)));

  //Obtenção dos dados para uso das funções de intervalo de Ap
  lNCordS5 := 0;
  For j := 0 to LBX_Linha.Count - 1 do begin
```

```
    Ver_Arm := tDados(LBX_Linha.Items.Objects[j]);
    if Ver_Arm.vTBarra = 1 then begin
      vACord := Ver_Arm.vABarra;
      if Ver_Arm.vPBarra = 2 then
        lNCordS5 := lNCordS5 + Ver_Arm.vNBarra;
    end;
  end;

  lTCordI := StrToFloat(IsNil(EDB_TensaoIniI.Text)) / 10;
  lTCordS := StrToFloat(IsNil(EDB_TensaoIniS.Text)) / 10;
  lPIniI := 1 - StrToFloat(IsNil(EDB_PerdaIniI.Text)) / 100;
  lPFinI := 1 - StrToFloat(IsNil(EDB_PerdaFinI.Text)) / 100;
  lPIniS := 1 - StrToFloat(IsNil(EDB_PerdaIniS.Text)) / 100;
  lPFinS := 1 - StrToFloat(IsNil(EDB_PerdaFinS.Text)) / 100;

  aAp[0] := GetInfAp(vACord, lTCordI*lPIniI, lNCordS5, lTCordS*lPIniS, lEi,
lEs, vA, tWi.X, tWf.X, aMomentos[0].S5, 0, 0, 0, aLimTen[0]);
  aAp[1] := GetSupAp(vACord, lTCordI*lPIniI, lNCordS5, lTCordS*lPIniS, lEi,
lEs, vA, tWi.Y, tWf.Y, aMomentos[0].S5, 0, 0, 0, aLimTen[0]);
  aAp[2] := GetInfAp(vACord, lTCordI*lPIniI, lNCordS5, lTCordS*lPIniS, lEi,
lEs, vA, tWi.X, tWf.X, aMomentos[0].S5, 0, 0, 0, aLimTen[1]);
  aAp[3] := GetSupAp(vACord, lTCordI*lPIniI, lNCordS5, lTCordS*lPIniS, lEi,
lEs, vA, tWi.Y, tWf.Y, aMomentos[0].S5, 0, 0, 0, aLimTen[1]);
  aAp[4] := GetInfAp(vACord, lTCordI*lPIniI, lNCordS5, lTCordS*lPIniS, lEi,
lEs, vA, tWi.X, tWf.X, aMomentos[0].S5, 0, 0, 0, 0);
  aAp[5] := GetSupAp(vACord, lTCordI*lPIniI, lNCordS5, lTCordS*lPIniS, lEi,
lEs, vA, tWi.Y, tWf.Y, aMomentos[0].S5, 0, 0, 0, 0);
  aAp[6] := GetInfAp(vACord, lTCordI*lPFinI, lNCordS5, lTCordS*lPFinS, lEi,
lEs, vA, tWi.X, tWf.X, aMomentos[7].S5, aMomentos[8].S5, aMomentos[5].S5,
tPsi.X, aLimTen[3]);
  aAp[7] := GetSupAp(vACord, lTCordI*lPFinI, lNCordS5, lTCordS*lPFinS, lEi,
lEs, vA, tWi.Y, tWf.Y, aMomentos[7].S5, aMomentos[8].S5, aMomentos[5].S5,
tPsi.X, aLimTen[3]);
  aAp[8] := GetInfAp(vACord, lTCordI*lPFinI, lNCordS5, lTCordS*lPFinS, lEi,
lEs, vA, tWi.X, tWf.X, aMomentos[7].S5, aMomentos[8].S5, aMomentos[5].S5,
tPsi.Y, aLimTen[4]);
  aAp[9] := GetSupAp(vACord, lTCordI*lPFinI, lNCordS5, lTCordS*lPFinS, lEi,
lEs, vA, tWi.Y, tWf.Y, aMomentos[7].S5, aMomentos[8].S5, aMomentos[5].S5,
tPsi.Y, aLimTen[4]);
  aAp[10] := GetInfAp(vACord, lTCordI*lPFinI, lNCordS5, lTCordS*lPFinS,
lEi, lEs, vA, tWi.X, tWf.X, aMomentos[7].S5, aMomentos[8].S5, 0, 0,
aLimTen[3]);
  aAp[11] := GetSupAp(vACord, lTCordI*lPFinI, lNCordS5, lTCordS*lPFinS,
lEi, lEs, vA, tWi.Y, tWf.Y, aMomentos[7].S5, aMomentos[8].S5, 0, 0,
aLimTen[3]);
  aAp[12] := GetInfAp(vACord, lTCordI*lPFinI, lNCordS5, lTCordS*lPFinS,
lEi, lEs, vA, tWi.X, tWf.X, aMomentos[7].S5, aMomentos[8].S5, 0, 0,
aLimTen[4]);
  aAp[13] := GetSupAp(vACord, lTCordI*lPFinI, lNCordS5, lTCordS*lPFinS,
lEi, lEs, vA, tWi.Y, tWf.Y, aMomentos[7].S5, aMomentos[8].S5, 0, 0,
aLimTen[4]);
  if (aAp[1].Y = 0) or (aAp[3].Y = 0) or (aAp[5].Y = 0) or (aAp[7].Y = 0)
or
     (aAp[9].Y = 0) or (aAp[11].Y = 0) or (aAp[13].Y = 0) then
       ShowMessage('A excentricidade está muito alta! O gráfico com os
intervalos de "Ap" pode não funcionar corretamente!');

  CHL_Vazio1.Clear;
  CHL_Vazio2.Clear;
  CHL_Vazio3.Clear;
  CHL_Serv1.Clear;
```

```
  CHL_Serv2.Clear;
  CHL_Serv3.Clear;
  CHL_Serv4.Clear;
  CHL_Ap.Clear;
  CHL_ApUtil.Clear;

  CHL_Vazio1.AddXY(DrawGraphic(aAp[0], aAp[1]).X, 1);
  CHL_Vazio1.AddXY(DrawGraphic(aAp[0], aAp[1]).Y, 1);
  CHL_Vazio2.AddXY(DrawGraphic(aAp[2], aAp[3]).X, 2);
  CHL_Vazio2.AddXY(DrawGraphic(aAp[2], aAp[3]).Y, 2);
  CHL_Vazio3.AddXY(DrawGraphic(aAp[4], aAp[5]).X, 3);
  CHL_Vazio3.AddXY(DrawGraphic(aAp[4], aAp[5]).Y, 3);
  CHL_Serv1.AddXY(DrawGraphic(aAp[6], aAp[7]).X, 4);
  CHL_Serv1.AddXY(DrawGraphic(aAp[6], aAp[7]).Y, 4);
  CHL_Serv2.AddXY(DrawGraphic(aAp[8], aAp[9]).X, 5);
  CHL_Serv2.AddXY(DrawGraphic(aAp[8], aAp[9]).Y, 5);
  CHL_Serv3.AddXY(DrawGraphic(aAp[10], aAp[11]).X, 6);
  CHL_Serv3.AddXY(DrawGraphic(aAp[10], aAp[11]).Y, 6);
  CHL_Serv4.AddXY(DrawGraphic(aAp[12], aAp[13]).X, 7);
  CHL_Serv4.AddXY(DrawGraphic(aAp[12], aAp[13]).Y, 7);
  CHL_Ap.AddXY(tAp.Ap, 0);
  CHL_Ap.AddXY(tAp.Ap, 8);
  CHL_ApUtil.AddXY(vAAp, 0);
  CHL_ApUtil.AddXY(vAAp, 8);
end;

//Evento clique do botão "Fissuração" da aba "Dimensionamento"
procedure TFRM_PTracao.BTN_FissuracaoClick(Sender: TObject);
var
  k: integer;
  vNCordS5, vNCordI5, vNCordI4, vNCordI3, vNCordI2, vNCordI1: integer;
  lSigI, lSigS, lLpi, lLps, lEi, lEtaP, lMfreq, lPsi1: extended;
  lDi, lDs: extended;
  aAp, aAs, aIxII: tPoints;
  aNp: array [0..7] of tPoints;
  aNLp: array [0..5] of integer;
  aTenInf, aTenSup: tEsforcos;
  Ver_Arm: tDados;
begin
  vNCordI1 := 0; vNCordI2 := 0; vNCordI3 := 0; vNCordI4 := 0; vNCordI5 :=
0; vNCordS5 := 0;
  //Busca o número de barras positivas e negativas totais e em cada seção
  For k := 0 to LBX_Linha.Count - 1 do begin
    Ver_Arm := tDados(LBX_Linha.Items.Objects[k]);
    if Ver_Arm.vTBarra = 1 then begin
      vABarra := Ver_Arm.vABarra;
      if Ver_Arm.vPBarra = 1 then
        vNCordI5 := vNCordI5 + Ver_Arm.vNBarra
      else if Ver_Arm.vPBarra = 2 then
        vNCordS5 := vNCordS5 + Ver_Arm.vNBarra;
    end;
  end;
  vNCordI1 := vNCordI5 - StrToInt(IsNil(EDB_S1.Text));
  vNCordI2 := vNCordI5 - StrToInt(IsNil(EDB_S2.Text));
  vNCordI3 := vNCordI5 - StrToInt(IsNil(EDB_S3.Text));
  vNCordI4 := vNCordI5 - StrToInt(IsNil(EDB_S4.Text));
  lDi := StrToFloat(SGR_Dimens.Cells[3,2]) * 100;
  lDs := StrToFloat(SGR_Dimens.Cells[3,3]) * 100;
  lLpi := StrToFloat(IsNil(EDB_LRegularizacaoI.Text));
  lLps := StrToFloat(IsNil(EDB_LRegularizacaoS.Text));
  lSigI := StrToFloat(IsNil(EDB_TensaoIniI.Text));
```

```
  lSigS := StrToFloat(IsNil(EDB_TensaoIniS.Text));
  lEtaP := StrToFloat(IsNil(EDB_Eta1.Text));
  lPsi1 := StrToFloat(IsNil(EDB_psi1.Text));
  aNLp[0] := vNCordS5;
  aNLp[1] := vNCordI5 - StrToInt(IsNil(EDB_S1.Text));
  aNLp[2] := StrToInt(IsNil(EDB_S1.Text)) - StrToInt(IsNil(EDB_S2.Text));
  aNLp[3] := StrToInt(IsNil(EDB_S2.Text)) - StrToInt(IsNil(EDB_S3.Text));
  aNLp[4] := StrToInt(IsNil(EDB_S3.Text)) - StrToInt(IsNil(EDB_S4.Text));
  aNLp[5] := 0;
  aTenInf.S1 := lSigI * (1 - StrToFloat(IsNil(EDB_PTDifI1.Text)) / 100);
  aTenInf.S2 := lSigI * (1 - StrToFloat(IsNil(EDB_PTDifI2.Text)) / 100);
  aTenInf.S3 := lSigI * (1 - StrToFloat(IsNil(EDB_PTDifI3.Text)) / 100);
  aTenInf.S4 := lSigI * (1 - StrToFloat(IsNil(EDB_PTDifI4.Text)) / 100);
  aTenInf.S5 := lSigI * (1 - StrToFloat(IsNil(EDB_PTDifI5.Text)) / 100);
  aTenInf.SLpi := lSigI * (1 - StrToFloat(IsNil(EDB_PTDifI6.Text)) / 100);
  aTenInf.SLps := lSigI * (1 - StrToFloat(IsNil(EDB_PTDifI7.Text)) / 100);
  aTenSup.S1 := lSigS * (1 - StrToFloat(IsNil(EDB_PTDifS1.Text)) / 100);
  aTenSup.S2 := lSigS * (1 - StrToFloat(IsNil(EDB_PTDifS2.Text)) / 100);
  aTenSup.S3 := lSigS * (1 - StrToFloat(IsNil(EDB_PTDifS3.Text)) / 100);
  aTenSup.S4 := lSigS * (1 - StrToFloat(IsNil(EDB_PTDifS4.Text)) / 100);
  aTenSup.S5 := lSigS * (1 - StrToFloat(IsNil(EDB_PTDifS5.Text)) / 100);
  aTenSup.SLpi := lSigS * (1 - StrToFloat(IsNil(EDB_PTDifS6.Text)) / 100);
  aTenSup.SLps := lSigS * (1 - StrToFloat(IsNil(EDB_PTDifS7.Text)) / 100);
  aNp := GetSecNp(vABarra, lLpi, lLps, vL, aNLp, aTenInf, aTenSup);
  aAs := GetAsArea(LBX_Linha.Count, LBX_Linha, tCG);
  aAp := GetApArea(LBX_Linha.Count, LBX_Linha, tCG);
  lEi := (GetExcentricity_Pos(LBX_Linha.Count, LBX_Linha, tCG)) / 100;
  aIxII := InertiaII(lDi, lDs, aAp.X, aAp.Y, aAs.X, aAs.Y, vEp, vEdoce,
aDimensoes);
  lMfreq := aMomentos[0].S5 + aMomentos[1].S5 + aMomentos[2].S5 +
aMomentos[3].S5 +
          aMomentos[4].S5 + aMomentos[5].S5 * lPsi1;
  aFissura := AbertFiss(aNp[5].X, aNp[5].Y, vA, lEi, vIx, aIxII.Y, tIxY.Y,
lEtaP, vEtaS, vEp, vEdoce, aEcs[6].X, vB, vFck, lMfreq, LBX_Linha);
  BTN_FissErros.Click;
  GRB_Resultados.Show;
end;

//Evento clique do botão "Completa" para apresentar a lista completa das
armaduras
procedure TFRM_PTracao.BTN_FissCompletaClick(Sender: TObject);
var
  i, t: integer;
  lx, ly, lfi, lWk1, lWk2, la, lb, lc, ld: string;
begin
  MMO_Fissura.Clear;
  t := Length(aFissura);
  for i := 0 to Length(aFissura) - 1 do begin
    lx := FormatFloat('###,###,##0.0', aFissura[i].X);
    ly := FormatFloat('###,###,##0.0', aFissura[i].Y);
    lfi := FormatFloat('###,###,##0.0', aFissura[i].Fi*10);
    lWk1 := FormatFloat('###,###,##0.000', aFissura[i].Wk1);
    lWk2 := FormatFloat('###,###,##0.000', aFissura[i].Wk2);
    la := FormatFloat('###,###,##0.0', aFissura[i].a);
    lb := FormatFloat('###,###,##0.0', aFissura[i].b);
    lc := FormatFloat('###,###,##0.0', aFissura[i].c);
    ld := FormatFloat('###,###,##0.0', aFissura[i].d);
    MMO_Fissura.Lines.Add(IntToStr(i+1) + '-)');
    MMO_Fissura.Lines.Add('- (' + lx + ' ; ' + ly + ')');
    MMO_Fissura.Lines.Add('- Diâmetro: ' + lfi + ' mm');
    MMO_Fissura.Lines.Add('- Wk1 = ' + lWk1 + ' mm');
```

```
    MMO_Fissura.Lines.Add('- a = ' + la + ' cm / b = ' + lb + ' cm');
    MMO_Fissura.Lines.Add('- c = ' + lc + ' cm / d = ' + ld + ' cm');
    MMO_Fissura.Lines.Add('- Wk2 = ' + lWk2 + ' mm');
    if (aFissura[i].wk1 <= vWk) or (aFissura[i].wk2 <= vWk) then
      MMO_Fissura.Lines.Add('- Status: Ok!')
    else
      MMO_Fissura.Lines.Add('- Status: Não OK! Verificar fissuras!');
    MMO_Fissura.Lines.Add('========================');
  end;
end;

procedure TFRM_PTracao.BTN_FissErrosClick(Sender: TObject);
var
  i, j: integer;
  lx, ly, lfi, lWk1, lWk2, la, lb, lc, ld: string;
begin
  MMO_Fissura.Clear;
  j := 0;
  for i := 0 to Length(aFissura) - 1 do begin
    if (aFissura[i].wk1 > vWk) and (aFissura[i].wk2 > vWk) then begin
      lx := FormatFloat('###,###,##0.0', aFissura[i].X);
      ly := FormatFloat('###,###,##0.0', aFissura[i].Y);
      lfi := FormatFloat('###,###,##0.00', aFissura[i].Fi*10);
      lWk1 := FormatFloat('###,###,##0.000', aFissura[i].Wk1);
      lWk2 := FormatFloat('###,###,##0.000', aFissura[i].Wk2);
      la := FormatFloat('###,###,##0.0', aFissura[i].a);
      lb := FormatFloat('###,###,##0.0', aFissura[i].b);
      lc := FormatFloat('###,###,##0.0', aFissura[i].c);
      ld := FormatFloat('###,###,##0.0', aFissura[i].d);
      MMO_Fissura.Lines.Add(IntToStr(i+1) + '-)');
      MMO_Fissura.Lines.Add('- (' + lx + ' ; ' + ly + ')');
      MMO_Fissura.Lines.Add('- Diâmetro: ' + lfi + ' mm');
      MMO_Fissura.Lines.Add('- Wk1 = ' + lWk1 + ' mm');
      MMO_Fissura.Lines.Add('- a = ' + la + ' cm / b = ' + lb + ' cm');
      MMO_Fissura.Lines.Add('- c = ' + lc + ' cm / d = ' + ld + ' cm');
      MMO_Fissura.Lines.Add('- Wk2 = ' + lWk2 + ' mm');
      MMO_Fissura.Lines.Add('- Status: Não OK! Verificar fissuras!');
      MMO_Fissura.Lines.Add('========================');
      j := j + 1;
    end;
  end;
  if j = 0 then
    MMO_Fissura.Lines.Add('Não há erros com a abertura de fissuras!');
end;

//Botão que corrige a área de concreto no cálculo da fissuração
procedure TFRM_PTracao.BTN_CorrigirFissClick(Sender: TObject);
var
  i: integer;
  la, lb, lc, ld, lEtaP: extended;
begin
  i := StrToInt(IsNil(EDB_indice.Text));
  if i <= 0 then begin
    ShowMessage('Indique a posição da armadura a ser corrigida!');
    Exit;
  end
  else if i > Length(aFissura) then begin
    ShowMessage('Essa posição não existe!');
    Exit;
  end
```

```
  else if (aFissura[i-1].wk1 <= vWk) or (aFissura[i-1].wk2 <= vWk) then
begin
    ShowMessage('Essa armadura não precisa ser corrigida!');
    Exit;
  end
  else begin
    la := StrToFloat(IsNil(EDB_a.Text));
    lb := StrToFloat(IsNil(EDB_b.Text));
    lc := StrToFloat(IsNil(EDB_c.Text));
    ld := StrToFloat(IsNil(EDB_d.Text));
    lEtaP := StrToFloat(IsNil(EDB_Eta1.Text));
    aFissura[i-1].a := la; aFissura[i-1].b := lb; aFissura[i-1].c := lc;
aFissura[i-1].d := ld;
    aFissura[i-1] := VerificFiss(i, lEtaP, vEtaS, vEp, vEdoce, aFissura[i-
1]);
  end;
  BTN_FissErros.Click;
end;

//Evento clique do botão "Verificar" da aba "Perdas de protensão"
procedure TFRM_PTracao.BTN_VerificarClick(Sender: TObject);
var
  Ver_Arm: tDados;
  i, j, k: integer;
  vPIniI, vPFinI, vPIniS, vPFinS, vEi, vEs: extended;
  lSigI, lSigS, lSigIniI, lSigIniS, lSigFinI, lSigFinS: extended;
  lLpi, lLps, vNCordLpi, vNCordLps: extended;
  vNCordS5, vNCordI5, vNCordI4, vNCordI3, vNCordI2, vNCordI1: integer;
  aEsfSec: array[0..7] of extended;
  aNp: array[0..7] of tPoints;
  aNLp: array[0..5] of integer;
  aTenInf, aTenSup: tEsforcos;
  begin
  vNCordI1 := 0; vNCordI2 := 0; vNCordI3 := 0; vNCordI4 := 0; vNCordI5 :=
0; vNCordS5 := 0;
  //Busca o número de barras positivas e negativas totais e em cada seção
  For k := 0 to LBX_Linha.Count - 1 do begin
    Ver_Arm := tDados(LBX_Linha.Items.Objects[k]);
    if Ver_Arm.vTBarra = 1 then begin
      vABarra := Ver_Arm.vABarra;
      if Ver_Arm.vPBarra = 1 then
        vNCordI5 := vNCordI5 + Ver_Arm.vNBarra
      else if Ver_Arm.vPBarra = 2 then
        vNCordS5 := vNCordS5 + Ver_Arm.vNBarra;
    end;
  end;
  vNCordI1 := vNCordI5 - StrToInt(IsNil(EDB_S1.Text));
  vNCordI2 := vNCordI5 - StrToInt(IsNil(EDB_S2.Text));
  vNCordI3 := vNCordI5 - StrToInt(IsNil(EDB_S3.Text));
  vNCordI4 := vNCordI5 - StrToInt(IsNil(EDB_S4.Text));
  //Momentos e dados na seção do comprimento de transferência INFERIOR e
SUPERIOR
  lLpi := StrToFloat(IsNil(EDB_LRegularizacaoI.Text));
  lLps := StrToFloat(IsNil(EDB_LRegularizacaoS.Text));
  aMomentos[0].SLpi := Momento_Lp(vPP, vL, lLpi);
  aMomentos[5].SLpi := Momento_Lp(vQmax, vL, lLpi);
  aMomentos[6].SLpi := Momento_Lp(vQmin, vL, lLpi);
  aMomentos[7].SLpi := Momento_Lp(vPTotal, vL, lLpi);
  aMomentos[8].SLpi := Momento_Lp(vP2Total, vL, lLpi);
  aMomentos[0].SLps := Momento_Lp(vPP, vL, lLps);
  aMomentos[5].SLps := Momento_Lp(vQmax, vL, lLps);
```

```
  aMomentos[6].SLps := Momento_Lp(vQmin, vL, lLps);
  aMomentos[7].SLps := Momento_Lp(vPTotal, vL, lLps);
  aMomentos[8].SLps := Momento_Lp(vP2Total, vL, lLps);
  //Salva os dados de armadura ativa necessária para utilizar a função de
tensão
  vTCordI := StrToFloat(IsNil(EDB_TensaoIniI.Text));
  vTCordS := StrToFloat(IsNil(EDB_TensaoIniS.Text));
  aNLp[0] := vNCordS5;
  aNLp[1] := vNCordI5 - StrToInt(IsNil(EDB_S1.Text));
  aNLp[2] := StrToInt(IsNil(EDB_S1.Text)) - StrToInt(IsNil(EDB_S2.Text));
  aNLp[3] := StrToInt(IsNil(EDB_S2.Text)) - StrToInt(IsNil(EDB_S3.Text));
  aNLp[4] := StrToInt(IsNil(EDB_S3.Text)) - StrToInt(IsNil(EDB_S4.Text));
  aNLp[5] := 0;
  //Cálculo das tensões
  //Armazenando os valores de força normal de protensão em vazio
considerando Lp
  aTenInf.S1 := vTCordI * (1 - StrToFloat(IsNil(EDB_PTImedI1.Text)) / 100);
  aTenInf.S2 := vTCordI * (1 - StrToFloat(IsNil(EDB_PTImedI2.Text)) / 100);
  aTenInf.S3 := vTCordI * (1 - StrToFloat(IsNil(EDB_PTImedI3.Text)) / 100);
  aTenInf.S4 := vTCordI * (1 - StrToFloat(IsNil(EDB_PTImedI4.Text)) / 100);
  aTenInf.S5 := vTCordI * (1 - StrToFloat(IsNil(EDB_PTImedI5.Text)) / 100);
  aTenInf.SLpi := vTCordI * (1 - StrToFloat(IsNil(EDB_PTImedI6.Text)) /
100);
  aTenInf.SLps := vTCordI * (1 - StrToFloat(IsNil(EDB_PTImedI7.Text)) /
100);
  aTenSup.S1 := vTCordS * (1 - StrToFloat(IsNil(EDB_PTImedS1.Text)) / 100);
  aTenSup.S2 := vTCordS * (1 - StrToFloat(IsNil(EDB_PTImedS2.Text)) / 100);
  aTenSup.S3 := vTCordS * (1 - StrToFloat(IsNil(EDB_PTImedS3.Text)) / 100);
  aTenSup.S4 := vTCordS * (1 - StrToFloat(IsNil(EDB_PTImedS4.Text)) / 100);
  aTenSup.S5 := vTCordS * (1 - StrToFloat(IsNil(EDB_PTImedS5.Text)) / 100);
  aTenSup.SLpi := vTCordS * (1 - StrToFloat(IsNil(EDB_PTImedS6.Text)) /
100);
  aTenSup.SLps := vTCordS * (1 - StrToFloat(IsNil(EDB_PTImedS7.Text)) /
100);
  vEi := (GetExcentricity_Pos(LBX_Linha.Count, LBX_Linha, tCG)) / 100;
  vEs := (GetExcentricity_Neg(LBX_Linha.Count, LBX_Linha, tCG)) / 100;
  aNp := GetSecNp(vABarra, lLpi, lLps, vL, aNLp, aTenInf, aTenSup);

  //Utilização das funções de tensão em cada seção em vazio - Borda
superior
  aEsfSec[1] := GetSupTension(aNp[1].X, aNp[1].Y, vEi, vEs, vA, tWi.Y,
tWi.Y, aMomentos[0].S1, 0, 0, 0);
  aEsfSec[2] := GetSupTension(aNp[2].X, aNp[2].Y, vEi, vEs, vA, tWi.Y,
tWi.Y, aMomentos[0].S2, 0, 0, 0);
  aEsfSec[3] := GetSupTension(aNp[3].X, aNp[3].Y, vEi, vEs, vA, tWi.Y,
tWi.Y, aMomentos[0].S3, 0, 0, 0);
  aEsfSec[4] := GetSupTension(aNp[4].X, aNp[4].Y, vEi, vEs, vA, tWi.Y,
tWi.Y, aMomentos[0].S4, 0, 0, 0);
  aEsfSec[5] := GetSupTension(aNp[5].X, aNp[5].Y, vEi, vEs, vA, tWi.Y,
tWi.Y, aMomentos[0].S5, 0, 0, 0);
  aEsfSec[6] := GetSupTension(aNp[6].X, aNp[6].Y, vEi, vEs, vA, tWi.Y,
tWi.Y, aMomentos[0].SLpi, 0, 0, 0);
  aEsfSec[7] := GetSupTension(aNp[7].X, aNp[7].Y, vEi, vEs, vA, tWi.Y,
tWi.Y, aMomentos[0].SLps, 0, 0, 0);
  SGR_TenVazio.Cells[0,0] := IntToStr(Round(aEsfSec[1]));
  SGR_TenVazio.Cells[1,0] := IntToStr(Round(aEsfSec[2]));
  SGR_TenVazio.Cells[2,0] := IntToStr(Round(aEsfSec[3]));
  SGR_TenVazio.Cells[3,0] := IntToStr(Round(aEsfSec[4]));
  SGR_TenVazio.Cells[4,0] := IntToStr(Round(aEsfSec[5]));
  SGR_TenVazio.Cells[5,0] := IntToStr(Round(aEsfSec[6]));
  SGR_TenVazio.Cells[6,0] := IntToStr(Round(aEsfSec[7]));
```

```
  //Utilização das funções de tensão em cada seção em vazio - Borda
inferior
  aEsfSec[1] := GetInfTension(aNp[1].X, aNp[1].Y, vEi, vEs, vA, tWi.X,
tWi.X, aMomentos[0].S1, 0, 0, 0);
  aEsfSec[2] := GetInfTension(aNp[2].X, aNp[2].Y, vEi, vEs, vA, tWi.X,
tWi.X, aMomentos[0].S2, 0, 0, 0);
  aEsfSec[3] := GetInfTension(aNp[3].X, aNp[3].Y, vEi, vEs, vA, tWi.X,
tWi.X, aMomentos[0].S3, 0, 0, 0);
  aEsfSec[4] := GetInfTension(aNp[4].X, aNp[4].Y, vEi, vEs, vA, tWi.X,
tWi.X, aMomentos[0].S4, 0, 0, 0);
  aEsfSec[5] := GetInfTension(aNp[5].X, aNp[5].Y, vEi, vEs, vA, tWi.X,
tWi.X, aMomentos[0].S5, 0, 0, 0);
  aEsfSec[6] := GetInfTension(aNp[6].X, aNp[6].Y, vEi, vEs, vA, tWi.X,
tWi.X, aMomentos[0].SLpi, 0, 0, 0);
  aEsfSec[7] := GetInfTension(aNp[7].X, aNp[7].Y, vEi, vEs, vA, tWi.X,
tWi.X, aMomentos[0].SLps, 0, 0, 0);
  SGR_TenVazio.Cells[0,1] := IntToStr(Round(aEsfSec[1]));
  SGR_TenVazio.Cells[1,1] := IntToStr(Round(aEsfSec[2]));
  SGR_TenVazio.Cells[2,1] := IntToStr(Round(aEsfSec[3]));
  SGR_TenVazio.Cells[3,1] := IntToStr(Round(aEsfSec[4]));
  SGR_TenVazio.Cells[4,1] := IntToStr(Round(aEsfSec[5]));
  SGR_TenVazio.Cells[5,1] := IntToStr(Round(aEsfSec[6]));
  SGR_TenVazio.Cells[6,1] := IntToStr(Round(aEsfSec[7]));
  //Armazenando os valores de força normal de protensão em serviço
considerando Lp
  aTenInf.S1 := vTCordI * (1 - StrToFloat(IsNil(EDB_PTDifI1.Text)) / 100);
  aTenInf.S2 := vTCordI * (1 - StrToFloat(IsNil(EDB_PTDifI2.Text)) / 100);
  aTenInf.S3 := vTCordI * (1 - StrToFloat(IsNil(EDB_PTDifI3.Text)) / 100);
  aTenInf.S4 := vTCordI * (1 - StrToFloat(IsNil(EDB_PTDifI4.Text)) / 100);
  aTenInf.S5 := vTCordI * (1 - StrToFloat(IsNil(EDB_PTDifI5.Text)) / 100);
  aTenInf.SLpi := vTCordI * (1 - StrToFloat(IsNil(EDB_PTDifI6.Text)) /
100);
  aTenInf.SLps := vTCordI * (1 - StrToFloat(IsNil(EDB_PTDifI7.Text)) /
100);
  aTenSup.S1 := vTCordS * (1 - StrToFloat(IsNil(EDB_PTDifS1.Text)) / 100);
  aTenSup.S2 := vTCordS * (1 - StrToFloat(IsNil(EDB_PTDifS2.Text)) / 100);
  aTenSup.S3 := vTCordS * (1 - StrToFloat(IsNil(EDB_PTDifS3.Text)) / 100);
  aTenSup.S4 := vTCordS * (1 - StrToFloat(IsNil(EDB_PTDifS4.Text)) / 100);
  aTenSup.S5 := vTCordS * (1 - StrToFloat(IsNil(EDB_PTDifS5.Text)) / 100);
  aTenSup.SLpi := vTCordS * (1 - StrToFloat(IsNil(EDB_PTDifS6.Text)) /
100);
  aTenSup.SLps := vTCordS * (1 - StrToFloat(IsNil(EDB_PTDifS7.Text)) /
100);
  aNp := GetSecNp(vABarra, lLpi, lLps, vL, aNLp, aTenInf, aTenSup);
  //Utilização das funções de tensão em cada seção no infinito e acidental
máxima - Borda superior 1
  aEsfSec[1] := GetSupTension(aNp[1].X, aNp[1].Y, vEi, vEs, vA, tWi.Y,
tWf.Y, aMomentos[7].S1, aMomentos[8].S1, aMomentos[5].S1, tPsi.X);
  aEsfSec[2] := GetSupTension(aNp[2].X, aNp[2].Y, vEi, vEs, vA, tWi.Y,
tWf.Y, aMomentos[7].S2, aMomentos[8].S2, aMomentos[5].S2, tPsi.X);
  aEsfSec[3] := GetSupTension(aNp[3].X, aNp[3].Y, vEi, vEs, vA, tWi.Y,
tWf.Y, aMomentos[7].S3, aMomentos[8].S3, aMomentos[5].S3, tPsi.X);
  aEsfSec[4] := GetSupTension(aNp[4].X, aNp[4].Y, vEi, vEs, vA, tWi.Y,
tWf.Y, aMomentos[7].S4, aMomentos[8].S4, aMomentos[5].S4, tPsi.X);
  aEsfSec[5] := GetSupTension(aNp[5].X, aNp[5].Y, vEi, vEs, vA, tWi.Y,
tWf.Y, aMomentos[7].S5, aMomentos[8].S5, aMomentos[5].S5, tPsi.X);
  aEsfSec[6] := GetSupTension(aNp[6].X, aNp[6].Y, vEi, vEs, vA, tWi.Y,
tWf.Y, aMomentos[7].SLpi, aMomentos[8].SLpi, aMomentos[5].SLpi, tPsi.X);
  aEsfSec[7] := GetSupTension(aNp[7].X, aNp[7].Y, vEi, vEs, vA, tWi.Y,
tWf.Y, aMomentos[7].SLps, aMomentos[8].SLps, aMomentos[5].SLps, tPsi.X);
  SGR_TenQmax1.Cells[0,0] := IntToStr(Round(aEsfSec[1]));
```

```
  SGR_TenQmax1.Cells[1,0] := IntToStr(Round(aEsfSec[2]));
  SGR_TenQmax1.Cells[2,0] := IntToStr(Round(aEsfSec[3]));
  SGR_TenQmax1.Cells[3,0] := IntToStr(Round(aEsfSec[4]));
  SGR_TenQmax1.Cells[4,0] := IntToStr(Round(aEsfSec[5]));
  SGR_TenQmax1.Cells[5,0] := IntToStr(Round(aEsfSec[6]));
  SGR_TenQmax1.Cells[6,0] := IntToStr(Round(aEsfSec[7]));
  //Utilização das funções de tensão em cada seção no infinito e acidental
máxima - Borda inferior 1
  aEsfSec[1] := GetInfTension(aNp[1].X, aNp[1].Y, vEi, vEs, vA, tWi.X,
tWf.X, aMomentos[7].S1, aMomentos[8].S1, aMomentos[5].S1, tPsi.X);
  aEsfSec[2] := GetInfTension(aNp[2].X, aNp[2].Y, vEi, vEs, vA, tWi.X,
tWf.X, aMomentos[7].S2, aMomentos[8].S2, aMomentos[5].S2, tPsi.X);
  aEsfSec[3] := GetInfTension(aNp[3].X, aNp[3].Y, vEi, vEs, vA, tWi.X,
tWf.X, aMomentos[7].S3, aMomentos[8].S3, aMomentos[5].S3, tPsi.X);
  aEsfSec[4] := GetInfTension(aNp[4].X, aNp[4].Y, vEi, vEs, vA, tWi.X,
tWf.X, aMomentos[7].S4, aMomentos[8].S4, aMomentos[5].S4, tPsi.X);
  aEsfSec[5] := GetInfTension(aNp[5].X, aNp[5].Y, vEi, vEs, vA, tWi.X,
tWf.X, aMomentos[7].S5, aMomentos[8].S5, aMomentos[5].S5, tPsi.X);
  aEsfSec[6] := GetInfTension(aNp[6].X, aNp[6].Y, vEi, vEs, vA, tWi.X,
tWf.X, aMomentos[7].SLpi, aMomentos[8].SLpi, aMomentos[5].SLpi, tPsi.X);
  aEsfSec[7] := GetInfTension(aNp[7].X, aNp[7].Y, vEi, vEs, vA, tWi.X,
tWf.X, aMomentos[7].SLps, aMomentos[8].SLps, aMomentos[5].SLps, tPsi.X);
  SGR_TenQmax1.Cells[0,1] := IntToStr(Round(aEsfSec[1]));
  SGR_TenQmax1.Cells[1,1] := IntToStr(Round(aEsfSec[2]));
  SGR_TenQmax1.Cells[2,1] := IntToStr(Round(aEsfSec[3]));
  SGR_TenQmax1.Cells[3,1] := IntToStr(Round(aEsfSec[4]));
  SGR_TenQmax1.Cells[4,1] := IntToStr(Round(aEsfSec[5]));
  SGR_TenQmax1.Cells[5,1] := IntToStr(Round(aEsfSec[6]));
  SGR_TenQmax1.Cells[6,1] := IntToStr(Round(aEsfSec[7]));
  //Utilização das funções de tensão em cada seção no infinito e acidental
mínima - Borda superior 1
  aEsfSec[1] := GetSupTension(aNp[1].X, aNp[1].Y, vEi, vEs, vA, tWi.Y,
tWf.Y, aMomentos[7].S1, aMomentos[8].S1, aMomentos[6].S1, tPsi.X);
  aEsfSec[2] := GetSupTension(aNp[2].X, aNp[2].Y, vEi, vEs, vA, tWi.Y,
tWf.Y, aMomentos[7].S2, aMomentos[8].S2, aMomentos[6].S2, tPsi.X);
  aEsfSec[3] := GetSupTension(aNp[3].X, aNp[3].Y, vEi, vEs, vA, tWi.Y,
tWf.Y, aMomentos[7].S3, aMomentos[8].S3, aMomentos[6].S3, tPsi.X);
  aEsfSec[4] := GetSupTension(aNp[4].X, aNp[4].Y, vEi, vEs, vA, tWi.Y,
tWf.Y, aMomentos[7].S4, aMomentos[8].S4, aMomentos[6].S4, tPsi.X);
  aEsfSec[5] := GetSupTension(aNp[5].X, aNp[5].Y, vEi, vEs, vA, tWi.Y,
tWf.Y, aMomentos[7].S5, aMomentos[8].S5, aMomentos[6].S5, tPsi.X);
  aEsfSec[6] := GetSupTension(aNp[6].X, aNp[6].Y, vEi, vEs, vA, tWi.Y,
tWf.Y, aMomentos[7].SLpi, aMomentos[8].SLpi, aMomentos[6].SLpi, tPsi.X);
  aEsfSec[7] := GetSupTension(aNp[7].X, aNp[7].Y, vEi, vEs, vA, tWi.Y,
tWf.Y, aMomentos[7].SLps, aMomentos[8].SLps, aMomentos[6].SLps, tPsi.X);
  SGR_TenQmin1.Cells[0,0] := IntToStr(Round(aEsfSec[1]));
  SGR_TenQmin1.Cells[1,0] := IntToStr(Round(aEsfSec[2]));
  SGR_TenQmin1.Cells[2,0] := IntToStr(Round(aEsfSec[3]));
  SGR_TenQmin1.Cells[3,0] := IntToStr(Round(aEsfSec[4]));
  SGR_TenQmin1.Cells[4,0] := IntToStr(Round(aEsfSec[5]));
  SGR_TenQmin1.Cells[5,0] := IntToStr(Round(aEsfSec[6]));
  SGR_TenQmin1.Cells[6,0] := IntToStr(Round(aEsfSec[7]));
  //Utilização das funções de tensão em cada seção no infinito e acidental
mínima - Borda inferior 1
  aEsfSec[1] := GetInfTension(aNp[1].X, aNp[1].Y, vEi, vEs, vA, tWi.X,
tWf.X, aMomentos[7].S1, aMomentos[8].S1, aMomentos[6].S1, tPsi.X);
  aEsfSec[2] := GetInfTension(aNp[2].X, aNp[2].Y, vEi, vEs, vA, tWi.X,
tWf.X, aMomentos[7].S2, aMomentos[8].S2, aMomentos[6].S2, tPsi.X);
  aEsfSec[3] := GetInfTension(aNp[3].X, aNp[3].Y, vEi, vEs, vA, tWi.X,
tWf.X, aMomentos[7].S3, aMomentos[8].S3, aMomentos[6].S3, tPsi.X);
```

```
  aEsfSec[4] := GetInfTension(aNp[4].X, aNp[4].Y, vEi, vEs, vA, tWi.X,
tWf.X, aMomentos[7].S4, aMomentos[8].S4, aMomentos[6].S4, tPsi.X);
  aEsfSec[5] := GetInfTension(aNp[5].X, aNp[5].Y, vEi, vEs, vA, tWi.X,
tWf.X, aMomentos[7].S5, aMomentos[8].S5, aMomentos[6].S5, tPsi.X);
  aEsfSec[6] := GetInfTension(aNp[6].X, aNp[6].Y, vEi, vEs, vA, tWi.X,
tWf.X, aMomentos[7].SLpi, aMomentos[8].SLpi, aMomentos[6].SLpi, tPsi.X);
  aEsfSec[7] := GetInfTension(aNp[7].X, aNp[7].Y, vEi, vEs, vA, tWi.X,
tWf.X, aMomentos[7].SLps, aMomentos[8].SLps, aMomentos[6].SLps, tPsi.X);
  SGR_TenQmin1.Cells[0,1] := IntToStr(Round(aEsfSec[1]));
  SGR_TenQmin1.Cells[1,1] := IntToStr(Round(aEsfSec[2]));
  SGR_TenQmin1.Cells[2,1] := IntToStr(Round(aEsfSec[3]));
  SGR_TenQmin1.Cells[3,1] := IntToStr(Round(aEsfSec[4]));
  SGR_TenQmin1.Cells[4,1] := IntToStr(Round(aEsfSec[5]));
  SGR_TenQmin1.Cells[5,1] := IntToStr(Round(aEsfSec[6]));
  SGR_TenQmin1.Cells[6,1] := IntToStr(Round(aEsfSec[7]));
  //Utilização das funções de tensão em cada seção no infinito e acidental
máxima - Borda superior 2
  aEsfSec[1] := GetSupTension(aNp[1].X, aNp[1].Y, vEi, vEs, vA, tWi.Y,
tWf.Y, aMomentos[7].S1, aMomentos[8].S1, aMomentos[5].S1, tPsi.Y);
  aEsfSec[2] := GetSupTension(aNp[2].X, aNp[2].Y, vEi, vEs, vA, tWi.Y,
tWf.Y, aMomentos[7].S2, aMomentos[8].S2, aMomentos[5].S2, tPsi.Y);
  aEsfSec[3] := GetSupTension(aNp[3].X, aNp[3].Y, vEi, vEs, vA, tWi.Y,
tWf.Y, aMomentos[7].S3, aMomentos[8].S3, aMomentos[5].S3, tPsi.Y);
  aEsfSec[4] := GetSupTension(aNp[4].X, aNp[4].Y, vEi, vEs, vA, tWi.Y,
tWf.Y, aMomentos[7].S4, aMomentos[8].S4, aMomentos[5].S4, tPsi.Y);
  aEsfSec[5] := GetSupTension(aNp[5].X, aNp[5].Y, vEi, vEs, vA, tWi.Y,
tWf.Y, aMomentos[7].S5, aMomentos[8].S5, aMomentos[5].S5, tPsi.Y);
  aEsfSec[6] := GetSupTension(aNp[6].X, aNp[6].Y, vEi, vEs, vA, tWi.Y,
tWf.Y, aMomentos[7].SLpi, aMomentos[8].SLpi, aMomentos[5].SLpi, tPsi.Y);
  aEsfSec[7] := GetSupTension(aNp[7].X, aNp[7].Y, vEi, vEs, vA, tWi.Y,
tWf.Y, aMomentos[7].SLps, aMomentos[8].SLps, aMomentos[5].SLps, tPsi.Y);
  SGR_TenQmax2.Cells[0,0] := IntToStr(Round(aEsfSec[1]));
  SGR_TenQmax2.Cells[1,0] := IntToStr(Round(aEsfSec[2]));
  SGR_TenQmax2.Cells[2,0] := IntToStr(Round(aEsfSec[3]));
  SGR_TenQmax2.Cells[3,0] := IntToStr(Round(aEsfSec[4]));
  SGR_TenQmax2.Cells[4,0] := IntToStr(Round(aEsfSec[5]));
  SGR_TenQmax2.Cells[5,0] := IntToStr(Round(aEsfSec[6]));
  SGR_TenQmax2.Cells[6,0] := IntToStr(Round(aEsfSec[7]));
  //Utilização das funções de tensão em cada seção no infinito e acidental
máxima - Borda inferior 2
  aEsfSec[1] := GetInfTension(aNp[1].X, aNp[1].Y, vEi, vEs, vA, tWi.X,
tWf.X, aMomentos[7].S1, aMomentos[8].S1, aMomentos[5].S1, tPsi.Y);
  aEsfSec[2] := GetInfTension(aNp[2].X, aNp[2].Y, vEi, vEs, vA, tWi.X,
tWf.X, aMomentos[7].S2, aMomentos[8].S2, aMomentos[5].S2, tPsi.Y);
  aEsfSec[3] := GetInfTension(aNp[3].X, aNp[3].Y, vEi, vEs, vA, tWi.X,
tWf.X, aMomentos[7].S3, aMomentos[8].S3, aMomentos[5].S3, tPsi.Y);
  aEsfSec[4] := GetInfTension(aNp[4].X, aNp[4].Y, vEi, vEs, vA, tWi.X,
tWf.X, aMomentos[7].S4, aMomentos[8].S4, aMomentos[5].S4, tPsi.Y);
  aEsfSec[5] := GetInfTension(aNp[5].X, aNp[5].Y, vEi, vEs, vA, tWi.X,
tWf.X, aMomentos[7].S5, aMomentos[8].S5, aMomentos[5].S5, tPsi.Y);
  aEsfSec[6] := GetInfTension(aNp[6].X, aNp[6].Y, vEi, vEs, vA, tWi.X,
tWf.X, aMomentos[7].SLpi, aMomentos[8].SLpi, aMomentos[5].SLpi, tPsi.Y);
  aEsfSec[7] := GetInfTension(aNp[7].X, aNp[7].Y, vEi, vEs, vA, tWi.X,
tWf.X, aMomentos[7].SLps, aMomentos[8].SLps, aMomentos[5].SLps, tPsi.Y);
  SGR_TenQmax2.Cells[0,1] := IntToStr(Round(aEsfSec[1]));
  SGR_TenQmax2.Cells[1,1] := IntToStr(Round(aEsfSec[2]));
  SGR_TenQmax2.Cells[2,1] := IntToStr(Round(aEsfSec[3]));
  SGR_TenQmax2.Cells[3,1] := IntToStr(Round(aEsfSec[4]));
  SGR_TenQmax2.Cells[4,1] := IntToStr(Round(aEsfSec[5]));
  SGR_TenQmax2.Cells[5,1] := IntToStr(Round(aEsfSec[6]));
  SGR_TenQmax2.Cells[6,1] := IntToStr(Round(aEsfSec[7]));
```

```
  //Utilização das funções de tensão em cada seção no infinito e acidental
mínima - Borda superior 2
  aEsfSec[1] := GetSupTension(aNp[1].X, aNp[1].Y, vEi, vEs, vA, tWi.Y,
tWf.Y, aMomentos[7].S1, aMomentos[8].S1, aMomentos[6].S1, tPsi.Y);
  aEsfSec[2] := GetSupTension(aNp[2].X, aNp[2].Y, vEi, vEs, vA, tWi.Y,
tWf.Y, aMomentos[7].S2, aMomentos[8].S2, aMomentos[6].S2, tPsi.Y);
  aEsfSec[3] := GetSupTension(aNp[3].X, aNp[3].Y, vEi, vEs, vA, tWi.Y,
tWf.Y, aMomentos[7].S3, aMomentos[8].S3, aMomentos[6].S3, tPsi.Y);
  aEsfSec[4] := GetSupTension(aNp[4].X, aNp[4].Y, vEi, vEs, vA, tWi.Y,
tWf.Y, aMomentos[7].S4, aMomentos[8].S4, aMomentos[6].S4, tPsi.Y);
  aEsfSec[5] := GetSupTension(aNp[5].X, aNp[5].Y, vEi, vEs, vA, tWi.Y,
tWf.Y, aMomentos[7].S5, aMomentos[8].S5, aMomentos[6].S5, tPsi.Y);
  aEsfSec[6] := GetSupTension(aNp[6].X, aNp[6].Y, vEi, vEs, vA, tWi.Y,
tWf.Y, aMomentos[7].SLpi, aMomentos[8].SLpi, aMomentos[6].SLpi, tPsi.Y);
  aEsfSec[7] := GetSupTension(aNp[7].X, aNp[7].Y, vEi, vEs, vA, tWi.Y,
tWf.Y, aMomentos[7].SLps, aMomentos[8].SLps, aMomentos[6].SLps, tPsi.Y);
  SGR_TenQmin2.Cells[0,0] := IntToStr(Round(aEsfSec[1]));
  SGR_TenQmin2.Cells[1,0] := IntToStr(Round(aEsfSec[2]));
  SGR_TenQmin2.Cells[2,0] := IntToStr(Round(aEsfSec[3]));
  SGR_TenQmin2.Cells[3,0] := IntToStr(Round(aEsfSec[4]));
  SGR_TenQmin2.Cells[4,0] := IntToStr(Round(aEsfSec[5]));
  SGR_TenQmin2.Cells[5,0] := IntToStr(Round(aEsfSec[6]));
  SGR_TenQmin2.Cells[6,0] := IntToStr(Round(aEsfSec[7]));
  //Utilização das funções de tensão em cada seção no infinito e acidental
mínima - Borda inferior 2
  aEsfSec[1] := GetInfTension(aNp[1].X, aNp[1].Y, vEi, vEs, vA, tWi.X,
tWf.X, aMomentos[7].S1, aMomentos[8].S1, aMomentos[6].S1, tPsi.Y);
  aEsfSec[2] := GetInfTension(aNp[2].X, aNp[2].Y, vEi, vEs, vA, tWi.X,
tWf.X, aMomentos[7].S2, aMomentos[8].S2, aMomentos[6].S2, tPsi.Y);
  aEsfSec[3] := GetInfTension(aNp[3].X, aNp[3].Y, vEi, vEs, vA, tWi.X,
tWf.X, aMomentos[7].S3, aMomentos[8].S3, aMomentos[6].S3, tPsi.Y);
  aEsfSec[4] := GetInfTension(aNp[4].X, aNp[4].Y, vEi, vEs, vA, tWi.X,
tWf.X, aMomentos[7].S4, aMomentos[8].S4, aMomentos[6].S4, tPsi.Y);
  aEsfSec[5] := GetInfTension(aNp[5].X, aNp[5].Y, vEi, vEs, vA, tWi.X,
tWf.X, aMomentos[7].S5, aMomentos[8].S5, aMomentos[6].S5, tPsi.Y);
  aEsfSec[6] := GetInfTension(aNp[6].X, aNp[6].Y, vEi, vEs, vA, tWi.X,
tWf.X, aMomentos[7].SLpi, aMomentos[8].SLpi, aMomentos[6].SLpi, tPsi.Y);
  aEsfSec[7] := GetInfTension(aNp[7].X, aNp[7].Y, vEi, vEs, vA, tWi.X,
tWf.X, aMomentos[7].SLps, aMomentos[8].SLps, aMomentos[6].SLps, tPsi.Y);
  SGR_TenQmin2.Cells[0,1] := IntToStr(Round(aEsfSec[1]));
  SGR_TenQmin2.Cells[1,1] := IntToStr(Round(aEsfSec[2]));
  SGR_TenQmin2.Cells[2,1] := IntToStr(Round(aEsfSec[3]));
  SGR_TenQmin2.Cells[3,1] := IntToStr(Round(aEsfSec[4]));
  SGR_TenQmin2.Cells[4,1] := IntToStr(Round(aEsfSec[5]));
  SGR_TenQmin2.Cells[5,1] := IntToStr(Round(aEsfSec[6]));
  SGR_TenQmin2.Cells[6,1] := IntToStr(Round(aEsfSec[7]));
  //Entrando com os valores de décimo de vão nos labels
  LBL_RDS1.Caption := '(' + FormatFloat('###,###,##0.0', 1 * vL / 10) +
'm)';
  LBL_RDS2.Caption := '(' + FormatFloat('###,###,##0.0', 2 * vL / 10) +
'm)';
  LBL_RDS3.Caption := '(' + FormatFloat('###,###,##0.0', 3 * vL / 10) +
'm)';
  LBL_RDS4.Caption := '(' + FormatFloat('###,###,##0.0', 4 * vL / 10) +
'm)';
  LBL_RDS5.Caption := '(' + FormatFloat('###,###,##0.0', 5 * vL / 10) +
'm)';
  LBL_RDSLi.Caption := '(' + FormatFloat('###,###,##0.0', lLpi) + 'm)';
  LBL_RDSLs.Caption := '(' + FormatFloat('###,###,##0.0', lLps) + 'm)';
  GRB_Resultados.Show;
  end;
```

```
procedure TFRM_PTracao.BTN_LpClick(Sender: TObject);
var
  vFi, vSigmaI, vSigmaS, vEta1, vEta2: extended;
  i: integer;
  aAp: tPoints;
  tArm: tDados;
begin
  for i := 0 to LBX_Linha.Count - 1 do
  begin
  tArm := tDados(LBX_Linha.Items.Objects[i]);
    if tArm.vTBarra = 1 then
      vFi := StrToFloat(tArm.vDBarra) / 1000;
  end;
  if (CBX_TLiberacao1.ItemIndex = -1) or (CBX_TLiberacao1.ItemIndex = 0)
then begin
    ShowMessage('Escolha um tipo de liberação do dispositivo');
    Exit;
  end;
  aAp := GetApArea(LBX_Linha.Count, LBX_Linha, tCG);
  vEta1 := StrToFloat(EDB_Eta1.Text);
  vEta2 := StrToFloat(EDB_Eta2.Text);
  vSigmaI := StrToFloat(EDB_TensaoIniI.Text) * (1 -
StrToFloat(EDB_PerdaIniI.Text) / 100);
  EDB_LRegularizacaoI.Text := FormatFloat('###,###,##0.00',
CompTrans(vFckj, vFi, vGamaC, vSigmaI, vEta1, vEta2, vHPre,
CBX_TLiberacao1));
  if aAp.Y > 0 then begin
    vSigmaS := StrToFloat(IsNil(EDB_TensaoIniS.Text)) * (1 -
StrToFloat(IsNil(EDB_PerdaIniS.Text)) / 100);
    EDB_LRegularizacaoS.Text := FormatFloat('###,###,##0.00',
CompTrans(vFckj, vFi, vGamaC, vSigmaS, vEta1, vEta2, vHPre,
CBX_TLiberacao1));
  end;
end;

                    {ABA PERDAS DE PROTENSÃO}
//Calculo das perdas de protensão
procedure TFRM_PTracao.BTN_CalcPerdasClick(Sender: TObject);
var
  Ver_Arm: tDados;
  i, j: integer;
  lDiaProt, lDiaInf, lPMConc, lILConc: integer;
  vNCordS5, vNCordI5, vNCordI4, vNCordI3, vNCordI2, vNCordI1, vNCordILpi,
vNCordILps: integer;
  lPMEndur, lILEndur: string;
  lLPista, lDeltaL: extended;
  lPMSlump, lPMTMedia, lPMUmidade, lPMUAr, lPMAc: extended;
  lILSlump, lILTMedia, lILUmidade, lILUAr, lILAc: extended;
  lFptk, lSigInf, lSigSup, lSigD: extended;
  lEp, lAlfaP, lAlfaPF: extended;
  lEi, lEs, lEiC, lEsC, lDi, lDs, lDiS, lDsS: extended;
  lLpi, lLps, TCordI, lPerdaI, TCordS, lPerdaS: extended;
  rDefAnc, rRelArmI, rRelArmS, rPsiRelI, rPsiRelS, rRetra: extended;
  lSigI, lSigS, lFckj, lFck, lPsi2, lD, lMr, lIm, lFlecha, lS, lMat:
extended;
  aInerciaII, aInerciaIIs, aAp, aAs: tPoints;
  aDefConInf, aDefConSup, aFluInf, aFluSup, aTenInf, aTenSup: tEsforcos;
  aRelArmFI, aRelArmFS, aPsiRelFI, aPsiRelFS, aProgrI, aProgrS: tEsforcos;
  aLDimensoes: array of tConcreto;
  aNLp: array[0..5] of integer;
```

```
  aMSecao: array[0..6] of extended;
  aCoefFlu1, aCoefFlu2, aCoefFluF, aDias, aMomento, aPerdI, aPerdS, aSigI,
aSigS, aPerdI5, aPerdS5: array[0..7] of extended;
  aNp, aPerda: array[0..7] of tPoints;
begin
  vNCordI1 := 0; vNCordI2 := 0; vNCordI3 := 0; vNCordI4 := 0; vNCordI5 :=
0; vNCordS5 := 0;
  //Busca o número de barras positivas e negativas totais e em cada seção
  For j := 0 to LBX_Linha.Count - 1 do begin
    Ver_Arm := tDados(LBX_Linha.Items.Objects[j]);
    if Ver_Arm.vTBarra = 1 then begin
      vABarra := Ver_Arm.vABarra;
      if Ver_Arm.vPBarra = 1 then
        vNCordI5 := vNCordI5 + Ver_Arm.vNBarra
      else if Ver_Arm.vPBarra = 2 then
        vNCordS5 := vNCordS5 + Ver_Arm.vNBarra;
    end;
  end;
  vNCordI1 := vNCordI5 - StrToInt(IsNil(EDB_S1.Text));
  vNCordI2 := vNCordI5 - StrToInt(IsNil(EDB_S2.Text));
  vNCordI3 := vNCordI5 - StrToInt(IsNil(EDB_S3.Text));
  vNCordI4 := vNCordI5 - StrToInt(IsNil(EDB_S4.Text));
  lLpi := StrToFloat(IsNil(EDB_LRegularizacaoI.Text));
  lLps := StrToFloat(IsNil(EDB_LRegularizacaoS.Text));
  if lLpi <= 1 * vL / 10 then
    vNCordILpi := vNCordI1
  else if lLpi <= 2 * vL / 10 then
    vNCordILpi := vNCordI2
  else if lLpi <= 3 * vL / 10 then
    vNCordILpi := vNCordI3
  else if lLpi <= 4 * vL / 10 then
    vNCordILpi := vNCordI4
  else if lLpi <= 2 * vL / 10 then
    vNCordILpi := vNCordI5;
  if lLps <= 1 * vL / 10 then
    vNCordILps := vNCordI1
  else if lLps <= 2 * vL / 10 then
    vNCordILps := vNCordI2
  else if lLps <= 3 * vL / 10 then
    vNCordILps := vNCordI3
  else if lLps <= 4 * vL / 10 then
    vNCordILps := vNCordI4
  else if lLps <= 2 * vL / 10 then
    vNCordILps := vNCordI5;
  lFckj := StrToFloat(IsNil(EDB_fckj.Text));
  lFck := StrToFloat(IsNil(EDB_fck.Text));
  if lFck < 50 then
    lPMConc := 1
  else
    lPMConc := 2;
  if fckCapa < 50 then
    lILConc := 1
  else
    lILConc := 2;
  lLPista := StrToFloat(IsNil(EDB_CPista.Text));
  lDeltaL := StrToFloat(IsNil(EDB_DeltaL.Text));
  lPMSlump := StrToFloat(IsNil(EDB_PMSlump.Text));
  lPMTMedia := StrToFloat(IsNil(EDB_PMTMedia.Text));
  lPMUmidade := StrToFloat(IsNil(EDB_PMUmidade.Text));
  lPMUAr := StrToFloat(IsNil(EDB_PMUAr.Text));
  lPMAc := StrToFloat(IsNil(EDB_PMAc.Text));
```

```
  lPMEndur := FRM_DatasCoef.CBX_PMCimento.Text;
  lILSlump := StrToFloat(IsNil(EDB_ILSlump.Text));
  lILTMedia := StrToFloat(IsNil(EDB_ILTMedia.Text));
  lILUmidade := StrToFloat(IsNil(EDB_ILUmidade.Text));
  lILUAr := StrToFloat(IsNil(EDB_ILUAr.Text));
  lILAc := StrToFloat(IsNil(EDB_ILAc.Text));
  lILEndur := FRM_DatasCoef.CBX_ILCimento.Text;
  lEp := StrToFloat(IsNil(FRM_EdDados.EDB_EDEp.Text));
  lAlfaP := StrToFloat(IsNil(EDB_AlfaP1.Text));
  lAlfaPF := StrToFloat(IsNil(EDB_AlfaP2.Text));
  lFptk := vFptd * vGamaS;
  lDiaProt := vDProt;
  lDiaInf := vTInf;
  aAs := GetAsArea(LBX_Linha.Count, LBX_Linha, tCG);
  aAp := GetApArea(LBX_Linha.Count, LBX_Linha, tCG);
  lSigInf := StrToFloat(IsNil(EDB_TensaoIniI.Text));
  if aAp.Y > 0 then
    lSigSup := StrToFloat(IsNil(EDB_TensaoIniS.Text))
  else
    lSigSup := 1;
  lSigI := lSigInf;
  lSigS := lSigSup;
  aMomentos[0].SLpi := Momento_Lp(vPP, vL, lLpi);
  aMomentos[1].SLpi := Momento_Lp(vLaje, vL, lLpi);
  aMomentos[2].SLpi := Momento_Lp(vCapa, vL, lLpi);
  aMomentos[3].SLpi := Momento_Lp(vParede, vL, lLpi);
  aMomentos[4].SLpi := Momento_Lp(vRevestimento, vL, lLpi);
  aMomentos[5].SLpi := Momento_Lp(vQmax, vL, lLpi);
  aMomentos[6].SLpi := Momento_Lp(vQmin, vL, lLpi);
  aMomentos[7].SLpi := Momento_Lp(vPTotal, vL, lLpi);
  aMomentos[8].SLpi := Momento_Lp(vP2Total, vL, lLpi);
  aMomentos[0].SLps := Momento_Lp(vPP, vL, lLps);
  aMomentos[1].SLps := Momento_Lp(vLaje, vL, lLps);
  aMomentos[2].SLps := Momento_Lp(vCapa, vL, lLps);
  aMomentos[3].SLps := Momento_Lp(vParede, vL, lLps);
  aMomentos[4].SLps := Momento_Lp(vRevestimento, vL, lLps);
  aMomentos[5].SLps := Momento_Lp(vQmax, vL, lLps);
  aMomentos[6].SLps := Momento_Lp(vQmin, vL, lLps);
  aMomentos[7].SLps := Momento_Lp(vPTotal, vL, lLps);
  aMomentos[8].SLps := Momento_Lp(vP2Total, vL, lLps);
  aTenInf.S0 := lSigInf; aTenInf.S1 := lSigInf; aTenInf.S2 := lSigInf;
aTenInf.S3 := lSigInf;
  aTenInf.S4 := lSigInf; aTenInf.S5 := lSigInf; aTenInf.SLpi := lSigInf;
aTenInf.SLps := lSigInf;
  aTenSup.S0 := lSigSup; aTenSup.S1 := lSigSup; aTenSup.S2 := lSigSup;
aTenSup.S3 := lSigSup;
  aTenSup.S4 := lSigSup; aTenSup.S5 := lSigSup; aTenSup.SLpi := lSigSup;
aTenSup.SLps := lSigSup;
  aDefConInf.S0 := 0; aDefConInf.S1 := 0; aDefConInf.S2 := 0; aDefConInf.S3
:= 0;
  aDefConInf.S4 := 0; aDefConInf.S5 := 0; aDefConInf.SLpi := 0;
aDefConInf.SLps := 0;
  aDefConSup.S0 := 0; aDefConSup.S1 := 0; aDefConSup.S2 := 0; aDefConSup.S3
:= 0;
  aDefConSup.S4 := 0; aDefConSup.S5 := 0; aDefConSup.SLpi := 0;
aDefConSup.SLps := 0;
  aFluInf.S0 := 0; aFluInf.S1 := 0; aFluInf.S2 := 0; aFluInf.S3 := 0;
  aFluInf.S4 := 0; aFluInf.S5 := 0; aFluInf.SLpi := 0; aFluInf.SLps := 0;
  aFluSup.S0 := 0; aFluSup.S1 := 0; aFluSup.S2 := 0; aFluSup.S3 := 0;
  aFluSup.S4 := 0; aFluSup.S5 := 0; aFluSup.SLpi := 0; aFluSup.SLps := 0;
  lEi := (GetExcentricity_Pos(LBX_Linha.Count, LBX_Linha, tCG)) / 100;
```

```
  lEs := (GetExcentricity_Neg(LBX_Linha.Count, LBX_Linha, tCG)) / 100;
  lEiC := (GetExcentricity_Pos(LBX_Linha.Count, LBX_Linha, tIxY)) / 100;
  lEsC := (GetExcentricity_Neg(LBX_Linha.Count, LBX_Linha, tIxY)) / 100;
  lPsi2 := StrToFloat(IsNil(EDB_psi2.Text));
  aNLp[0] := vNCordS5;
  aNLp[1] := vNCordI5 - StrToInt(IsNil(EDB_S1.Text));
  aNLp[2] := StrToInt(IsNil(EDB_S1.Text)) - StrToInt(IsNil(EDB_S2.Text));
  aNLp[3] := StrToInt(IsNil(EDB_S2.Text)) - StrToInt(IsNil(EDB_S3.Text));
  aNLp[4] := StrToInt(IsNil(EDB_S3.Text)) - StrToInt(IsNil(EDB_S4.Text));
  aNLp[5] := 0;
  //Perda por deformação da ancoragem
  rDefAnc := DefAnc(lDeltaL, lLPista, lEp);
  aTenInf := SubCons(aTenInf, rDefAnc);
  aTenSup := SubCons(aTenSup, rDefAnc);
  aPerda[0].X := rDefAnc; aPerda[1].X := rDefAnc; aPerda[2].X := rDefAnc;
aPerda[3].X := rDefAnc;
  aPerda[4].X := rDefAnc; aPerda[5].X := rDefAnc; aPerda[6].X := rDefAnc;
  aPerda[0].Y := rDefAnc; aPerda[1].Y := rDefAnc; aPerda[2].Y := rDefAnc;
aPerda[3].Y := rDefAnc;
  aPerda[4].Y := rDefAnc; aPerda[5].Y := rDefAnc; aPerda[6].Y := rDefAnc;
  SGR_PerInfIni.Cells[1,1] := IntToStr(Round(rDefAnc));
  SGR_PerInfIni.Cells[2,1] := IntToStr(Round(rDefAnc));
  SGR_PerInfIni.Cells[3,1] := IntToStr(Round(rDefAnc));
  SGR_PerInfIni.Cells[4,1] := IntToStr(Round(rDefAnc));
  SGR_PerInfIni.Cells[5,1] := IntToStr(Round(rDefAnc));
  SGR_PerInfIni.Cells[6,1] := IntToStr(Round(rDefAnc));
  SGR_PerInfIni.Cells[7,1] := IntToStr(Round(rDefAnc));
  SGR_PerSupIni.Cells[1,1] := IntToStr(Round(rDefAnc));
  SGR_PerSupIni.Cells[2,1] := IntToStr(Round(rDefAnc));
  SGR_PerSupIni.Cells[3,1] := IntToStr(Round(rDefAnc));
  SGR_PerSupIni.Cells[4,1] := IntToStr(Round(rDefAnc));
  SGR_PerSupIni.Cells[5,1] := IntToStr(Round(rDefAnc));
  SGR_PerSupIni.Cells[6,1] := IntToStr(Round(rDefAnc));
  SGR_PerSupIni.Cells[7,1] := IntToStr(Round(rDefAnc));

  //Perda inicial por relaxação da armadura
  rPsiRelI := RelArm(aTenInf.S5, lFptk, 0, vDPP, aPsiMil, aR);
  rPsiRelS := RelArm(aTenSup.S5, lFptk, 0, vDPP, aPsiMil, aR);
  rRelArmI := aTenInf.S5 * rPsiRelI;
  rRelArmS := aTenSup.S5 * rPsiRelS;
  aTenInf := SubCons(aTenInf, rRelArmI);
  aTenSup := SubCons(aTenSup, rRelArmS);
  aPerda[0].X := aPerda[0].X + rRelArmI; aPerda[1].X := aPerda[1].X +
rRelArmI;
  aPerda[2].X := aPerda[2].X + rRelArmI; aPerda[3].X := aPerda[3].X +
rRelArmI;
  aPerda[4].X := aPerda[4].X + rRelArmI; aPerda[5].X := aPerda[5].X +
rRelArmI;
  aPerda[6].X := aPerda[6].X + rRelArmI; aPerda[0].Y := aPerda[0].Y +
rRelArmS;
  aPerda[1].Y := aPerda[1].Y + rRelArmS; aPerda[2].Y := aPerda[2].Y +
rRelArmS;
  aPerda[3].Y := aPerda[3].Y + rRelArmS; aPerda[4].Y := aPerda[4].Y +
rRelArmS;
  aPerda[5].Y := aPerda[5].Y + rRelArmS; aPerda[6].Y := aPerda[6].Y +
rRelArmS;
  SGR_PerInfIni.Cells[1,2] := IntToStr(Round(rRelArmI));
  SGR_PerInfIni.Cells[2,2] := IntToStr(Round(rRelArmI));
  SGR_PerInfIni.Cells[3,2] := IntToStr(Round(rRelArmI));
  SGR_PerInfIni.Cells[4,2] := IntToStr(Round(rRelArmI));
  SGR_PerInfIni.Cells[5,2] := IntToStr(Round(rRelArmI));
```

```
  SGR_PerInfIni.Cells[6,2] := IntToStr(Round(rRelArmI));
  SGR_PerInfIni.Cells[7,2] := IntToStr(Round(rRelArmI));
  SGR_PerSupIni.Cells[1,2] := IntToStr(Round(rRelArmS));
  SGR_PerSupIni.Cells[2,2] := IntToStr(Round(rRelArmS));
  SGR_PerSupIni.Cells[3,2] := IntToStr(Round(rRelArmS));
  SGR_PerSupIni.Cells[4,2] := IntToStr(Round(rRelArmS));
  SGR_PerSupIni.Cells[5,2] := IntToStr(Round(rRelArmS));
  SGR_PerSupIni.Cells[6,2] := IntToStr(Round(rRelArmS));
  SGR_PerSupIni.Cells[7,2] := IntToStr(Round(rRelArmS));
  //Perda por deformação do concreto
  aNp := GetSecNp(vABarra, lLpi, lLps, vL, aNLp, aTenInf, aTenSup);
  aDefConInf.S0 := 0;
  aDefConInf.S1 := DefConInf(aNp[1].X, aNp[1].Y, vA, lEi, lEs, vIx, lAlfaP,
aMomentos[0].S1);
  aDefConInf.S2 := DefConInf(aNp[2].X, aNp[2].Y, vA, lEi, lEs, vIx, lAlfaP,
aMomentos[0].S2);
  aDefConInf.S3 := DefConInf(aNp[3].X, aNp[3].Y, vA, lEi, lEs, vIx, lAlfaP,
aMomentos[0].S3);
  aDefConInf.S4 := DefConInf(aNp[4].X, aNp[4].Y, vA, lEi, lEs, vIx, lAlfaP,
aMomentos[0].S4);
  aDefConInf.S5 := DefConInf(aNp[5].X, aNp[5].Y, vA, lEi, lEs, vIx, lAlfaP,
aMomentos[0].S5);
  aDefConInf.SLpi := DefConInf(aNp[6].X, aNp[6].Y, vA, lEi, lEs, vIx,
lAlfaP, aMomentos[0].SLpi);
  aDefConInf.SLps := DefConInf(aNp[7].X, aNp[7].Y, vA, lEi, lEs, vIx,
lAlfaP, aMomentos[0].SLps);
  aDefConSup.S0 := 0;
  aDefConSup.S1 := DefConSup(aNp[1].X, aNp[1].Y, vA, lEi, lEs, vIx, lAlfaP,
aMomentos[0].S1);
  aDefConSup.S2 := DefConSup(aNp[2].X, aNp[2].Y, vA, lEi, lEs, vIx, lAlfaP,
aMomentos[0].S2);
  aDefConSup.S3 := DefConSup(aNp[3].X, aNp[3].Y, vA, lEi, lEs, vIx, lAlfaP,
aMomentos[0].S3);
  aDefConSup.S4 := DefConSup(aNp[4].X, aNp[4].Y, vA, lEi, lEs, vIx, lAlfaP,
aMomentos[0].S4);
  aDefConSup.S5 := DefConSup(aNp[5].X, aNp[5].Y, vA, lEi, lEs, vIx, lAlfaP,
aMomentos[0].S5);
  aDefConSup.SLpi := DefConSup(aNp[6].X, aNp[6].Y, vA, lEi, lEs, vIx,
lAlfaP, aMomentos[0].SLpi);
  aDefConSup.SLps := DefConSup(aNp[7].X, aNp[7].Y, vA, lEi, lEs, vIx,
lAlfaP, aMomentos[0].SLps);
  aTenInf := SubEsf(aTenInf, aDefConInf);
  aTenSup := SubEsf(aTenSup, aDefConSup);
  aPerda[0].X := aPerda[0].X + aDefConInf.S1; aPerda[1].X := aPerda[1].X +
aDefConInf.S2;
  aPerda[2].X := aPerda[2].X + aDefConInf.S3; aPerda[3].X := aPerda[3].X +
aDefConInf.S4;
  aPerda[4].X := aPerda[4].X + aDefConInf.S5; aPerda[5].X := aPerda[5].X +
aDefConInf.SLpi;
  aPerda[6].X := aPerda[6].X + aDefConInf.SLps; aPerda[0].Y := aPerda[0].Y
+ aDefConSup.S1;
  aPerda[1].Y := aPerda[1].Y + aDefConSup.S2; aPerda[2].Y := aPerda[2].Y +
aDefConSup.S3;
  aPerda[3].Y := aPerda[3].Y + aDefConSup.S4; aPerda[4].Y := aPerda[4].Y +
aDefConSup.S5;
  aPerda[5].Y := aPerda[5].Y + aDefConSup.SLpi; aPerda[6].Y := aPerda[6].Y
+ aDefConSup.SLps;
  SGR_PerInfIni.Cells[1,3] := IntToStr(Round(aDefConInf.S1));
  SGR_PerInfIni.Cells[2,3] := IntToStr(Round(aDefConInf.S2));
  SGR_PerInfIni.Cells[3,3] := IntToStr(Round(aDefConInf.S3));
  SGR_PerInfIni.Cells[4,3] := IntToStr(Round(aDefConInf.S4));
```

```
  SGR_PerInfIni.Cells[5,3] := IntToStr(Round(aDefConInf.S5));
  SGR_PerInfIni.Cells[6,3] := IntToStr(Round(aDefConInf.SLpi));
  SGR_PerInfIni.Cells[7,3] := IntToStr(Round(aDefConInf.SLps));
  SGR_PerSupIni.Cells[1,3] := IntToStr(Round(aDefConSup.S1));
  SGR_PerSupIni.Cells[2,3] := IntToStr(Round(aDefConSup.S2));
  SGR_PerSupIni.Cells[3,3] := IntToStr(Round(aDefConSup.S3));
  SGR_PerSupIni.Cells[4,3] := IntToStr(Round(aDefConSup.S4));
  SGR_PerSupIni.Cells[5,3] := IntToStr(Round(aDefConSup.S5));
  SGR_PerSupIni.Cells[6,3] := IntToStr(Round(aDefConSup.SLpi));
  SGR_PerSupIni.Cells[7,3] := IntToStr(Round(aDefConSup.SLps));

  SGR_PerInfIni.Cells[1,4] := IntToStr(Round(aPerda[0].X));
  SGR_PerInfIni.Cells[2,4] := IntToStr(Round(aPerda[1].X));
  SGR_PerInfIni.Cells[3,4] := IntToStr(Round(aPerda[2].X));
  SGR_PerInfIni.Cells[4,4] := IntToStr(Round(aPerda[3].X));
  SGR_PerInfIni.Cells[5,4] := IntToStr(Round(aPerda[4].X));
  SGR_PerInfIni.Cells[6,4] := IntToStr(Round(aPerda[5].X));
  SGR_PerInfIni.Cells[7,4] := IntToStr(Round(aPerda[6].X));
  SGR_PerSupIni.Cells[1,4] := IntToStr(Round(aPerda[0].Y));
  SGR_PerSupIni.Cells[2,4] := IntToStr(Round(aPerda[0].Y));
  SGR_PerSupIni.Cells[3,4] := IntToStr(Round(aPerda[0].Y));
  SGR_PerSupIni.Cells[4,4] := IntToStr(Round(aPerda[0].Y));
  SGR_PerSupIni.Cells[5,4] := IntToStr(Round(aPerda[0].Y));
  SGR_PerSupIni.Cells[6,4] := IntToStr(Round(aPerda[0].Y));
  SGR_PerSupIni.Cells[7,4] := IntToStr(Round(aPerda[0].Y));
  EDB_PTImedI1.Text := FormatFloat('###,###,##0.0', aPerda[0].X / lSigInf *
100);
  EDB_PTImedI2.Text := FormatFloat('###,###,##0.0', aPerda[1].X / lSigInf *
100);
  EDB_PTImedI3.Text := FormatFloat('###,###,##0.0', aPerda[2].X / lSigInf *
100);
  EDB_PTImedI4.Text := FormatFloat('###,###,##0.0', aPerda[3].X / lSigInf *
100);
  EDB_PTImedI5.Text := FormatFloat('###,###,##0.0', aPerda[4].X / lSigInf *
100);
  EDB_PTImedI6.Text := FormatFloat('###,###,##0.0', aPerda[5].X / lSigInf *
100);
  EDB_PTImedI7.Text := FormatFloat('###,###,##0.0', aPerda[6].X / lSigInf *
100);
  EDB_PTImedS1.Text := FormatFloat('###,###,##0.0', aPerda[0].Y / lSigSup *
100);
  EDB_PTImedS2.Text := FormatFloat('###,###,##0.0', aPerda[1].Y / lSigSup *
100);
  EDB_PTImedS3.Text := FormatFloat('###,###,##0.0', aPerda[2].Y / lSigSup *
100);
  EDB_PTImedS4.Text := FormatFloat('###,###,##0.0', aPerda[3].Y / lSigSup *
100);
  EDB_PTImedS5.Text := FormatFloat('###,###,##0.0', aPerda[4].Y / lSigSup *
100);
  EDB_PTImedS6.Text := FormatFloat('###,###,##0.0', aPerda[5].Y / lSigSup *
100);
  EDB_PTImedS7.Text := FormatFloat('###,###,##0.0', aPerda[6].Y / lSigSup *
100);
  //Perda por retração
  aPerdI[2] := Retrac(lPMUmidade, lPMAc, lPMUAr, lPMSlump, lPMTMedia, vDPP,
vDLaje, vEp);
  aPerdI[3] := Retrac(lPMUmidade, lPMAc, lPMUAr, lPMSlump, lPMTMedia, vDPP,
vDCapa, vEp);
  aPerdI[4] := Retrac(lPMUmidade, lPMAc, lPMUAr, lPMSlump, lPMTMedia, vDPP,
vDParede, vEp);
```

```
  aPerdI[5] := Retrac(lPMUmidade, lPMAc, lPMUAr, lPMSlump, lPMTMedia, vDPP,
vDRevest, vEp);
  aPerdI[6] := Retrac(lPMUmidade, lPMAc, lPMUAr, lPMSlump, lPMTMedia, vDPP,
vDAcid, vEp);
  aPerdS[2] := Retrac(lPMUmidade, lPMAc, lPMUAr, lPMSlump, lPMTMedia, vDPP,
vDLaje, vEp);
  aPerdS[3] := Retrac(lPMUmidade, lPMAc, lPMUAr, lPMSlump, lPMTMedia, vDPP,
vDCapa, vEp);
  aPerdS[4] := Retrac(lPMUmidade, lPMAc, lPMUAr, lPMSlump, lPMTMedia, vDPP,
vDParede, vEp);
  aPerdS[5] := Retrac(lPMUmidade, lPMAc, lPMUAr, lPMSlump, lPMTMedia, vDPP,
vDRevest, vEp);
  aPerdS[6] := Retrac(lPMUmidade, lPMAc, lPMUAr, lPMSlump, lPMTMedia, vDPP,
vDAcid, vEp);
  aPerdI5[2] := Retrac(lPMUmidade, lPMAc, lPMUAr, lPMSlump, lPMTMedia,
vDPP, vDLaje, vEp);
  aPerdI5[3] := Retrac(lPMUmidade, lPMAc, lPMUAr, lPMSlump, lPMTMedia,
vDPP, vDCapa, vEp);
  aPerdI5[4] := Retrac(lPMUmidade, lPMAc, lPMUAr, lPMSlump, lPMTMedia,
vDPP, vDParede, vEp);
  aPerdI5[5] := Retrac(lPMUmidade, lPMAc, lPMUAr, lPMSlump, lPMTMedia,
vDPP, vDRevest, vEp);
  aPerdI5[6] := Retrac(lPMUmidade, lPMAc, lPMUAr, lPMSlump, lPMTMedia,
vDPP, vDAcid, vEp);
  aPerdS5[2] := Retrac(lPMUmidade, lPMAc, lPMUAr, lPMSlump, lPMTMedia,
vDPP, vDLaje, vEp);
  aPerdS5[3] := Retrac(lPMUmidade, lPMAc, lPMUAr, lPMSlump, lPMTMedia,
vDPP, vDCapa, vEp);
  aPerdS5[4] := Retrac(lPMUmidade, lPMAc, lPMUAr, lPMSlump, lPMTMedia,
vDPP, vDParede, vEp);
  aPerdS5[5] := Retrac(lPMUmidade, lPMAc, lPMUAr, lPMSlump, lPMTMedia,
vDPP, vDRevest, vEp);
  aPerdS5[6] := Retrac(lPMUmidade, lPMAc, lPMUAr, lPMSlump, lPMTMedia,
vDPP, vDAcid, vEp);
  rRetra := Retrac(lPMUmidade, lPMAc, lPMUAr, lPMSlump, lPMTMedia, vDPP,
vTInf, vEp);
  aPerda[0].X := aPerda[0].X + rRetra; aPerda[1].X := aPerda[1].X + rRetra;
  aPerda[2].X := aPerda[2].X + rRetra; aPerda[3].X := aPerda[3].X + rRetra;
  aPerda[4].X := aPerda[4].X + rRetra; aPerda[5].X := aPerda[5].X + rRetra;
  aPerda[6].X := aPerda[6].X + rRetra; aPerda[0].Y := aPerda[0].Y + rRetra;
  aPerda[1].Y := aPerda[1].Y + rRetra; aPerda[2].Y := aPerda[2].Y + rRetra;
  aPerda[3].Y := aPerda[3].Y + rRetra; aPerda[4].Y := aPerda[4].Y + rRetra;
  aPerda[5].Y := aPerda[5].Y + rRetra; aPerda[6].Y := aPerda[6].Y + rRetra;
  SGR_PerInfDif.Cells[1,0] := IntToStr(Round(rRetra));
  SGR_PerInfDif.Cells[2,0] := IntToStr(Round(rRetra));
  SGR_PerInfDif.Cells[3,0] := IntToStr(Round(rRetra));
  SGR_PerInfDif.Cells[4,0] := IntToStr(Round(rRetra));
  SGR_PerInfDif.Cells[5,0] := IntToStr(Round(rRetra));
  SGR_PerInfDif.Cells[6,0] := IntToStr(Round(rRetra));
  SGR_PerInfDif.Cells[7,0] := IntToStr(Round(rRetra));
  SGR_PerSupDif.Cells[1,0] := IntToStr(Round(rRetra));
  SGR_PerSupDif.Cells[2,0] := IntToStr(Round(rRetra));
  SGR_PerSupDif.Cells[3,0] := IntToStr(Round(rRetra));
  SGR_PerSupDif.Cells[4,0] := IntToStr(Round(rRetra));
  SGR_PerSupDif.Cells[5,0] := IntToStr(Round(rRetra));
  SGR_PerSupDif.Cells[6,0] := IntToStr(Round(rRetra));
  SGR_PerSupDif.Cells[7,0] := IntToStr(Round(rRetra));
  //Perda por fluência
  aCoefFlu1[0] := CoefFlu(lPMConc, lPMTMedia, vDProt, vTInf, lPMUmidade,
lPMAc, lPMUAr, vFid, lPMSlump, lPMEndur, aAlfa, aCoefS, aFc);
```

```
  aCoefFlu1[1] := CoefFlu(lPMConc, lPMTMedia, vDPP, vTInf, lPMUmidade,
lPMAc, lPMUAr, vFid, lPMSlump, lPMEndur, aAlfa, aCoefS, aFc);
  aCoefFlu1[2] := CoefFlu(lPMConc, lPMTMedia, vDLaje, vTInf, lPMUmidade,
lPMAc, lPMUAr, vFid, lPMSlump, lPMEndur, aAlfa, aCoefS, aFc);
  aCoefFlu1[3] := CoefFlu(lPMConc, lPMTMedia, vDCapa, vTInf, lPMUmidade,
lPMAc, lPMUAr, vFid, lPMSlump, lPMEndur, aAlfa, aCoefS, aFc);
  aCoefFlu1[4] := CoefFlu(lPMConc, lPMTMedia, vDParede, vTInf, lPMUmidade,
lPMAc, lPMUAr, vFid, lPMSlump, lPMEndur, aAlfa, aCoefS, aFc);
  aCoefFlu1[5] := CoefFlu(lPMConc, lPMTMedia, vDRevest, vTInf, lPMUmidade,
lPMAc, lPMUAr, vFid, lPMSlump, lPMEndur, aAlfa, aCoefS, aFc);
  aCoefFlu1[6] := CoefFlu(lPMConc, lPMTMedia, vDAcid, vTInf, lPMUmidade,
lPMAc, lPMUAr, vFid, lPMSlump, lPMEndur, aAlfa, aCoefS, aFc);
  aCoefFlu1[7] := CoefFlu(lPMConc, lPMTMedia, vDPerda, vTInf, lPMUmidade,
lPMAc, lPMUAr, vFid, lPMSlump, lPMEndur, aAlfa, aCoefS, aFc);
  aCoefFlu2[0] := 0;
  aCoefFlu2[1] := 0;
  aCoefFlu2[2] := 0;
  aCoefFlu2[3] := CoefFlu(lILConc, lILTMedia, 1, vTInf-vDCapa, lILUmidade,
lILAc, lILUAr, vFid, lILSlump, lILEndur, aAlfa, aCoefS, aFc);
  aCoefFlu2[4] := CoefFlu(lILConc, lILTMedia, vDParede-vDCapa, vTInf-
vDParede, lILUmidade, lILAc, lILUAr, vFid, lILSlump, lILEndur, aAlfa,
aCoefS, aFc);
  aCoefFlu2[5] := CoefFlu(lILConc, lILTMedia, vDRevest-vDCapa, vTInf-
vDRevest, lILUmidade, lILAc, lILUAr, vFid, lILSlump, lILEndur, aAlfa,
aCoefS, aFc);
  aCoefFlu2[6] := CoefFlu(lILConc, lILTMedia, vDAcid-vDCapa, vTInf-vDAcid,
lILUmidade, lILAc, lILUAr, vFid, lILSlump, lILEndur, aAlfa, aCoefS, aFc);
  aCoefFlu2[7] := CoefFlu(lILConc, lILTMedia, vDPerda-vDCapa, vTInf-
vDPerda, lILUmidade, lILAc, lILUAr, vFid, lILSlump, lILEndur, aAlfa,
aCoefS, aFc);
  aCoefFluF[0] := aCoefFlu1[0];
  aCoefFluF[1] := aCoefFlu1[1];
  aCoefFluF[2] := aCoefFlu1[2];
  aCoefFluF[3] := MedPon(lPMAc, lILAc, aCoefFlu1[3], aCoefFlu2[3]);
  aCoefFluF[4] := MedPon(lPMAc, lILAc, aCoefFlu1[4], aCoefFlu2[4]);
  aCoefFluF[5] := MedPon(lPMAc, lILAc, aCoefFlu1[5], aCoefFlu2[5]);
  aCoefFluF[6] := MedPon(lPMAc, lILAc, aCoefFlu1[6], aCoefFlu2[6]);
  aCoefFluF[7] := MedPon(lPMAc, lILAc, aCoefFlu1[7], aCoefFlu2[7]);
  EDB_FluProt.Text := FormatFloat('###,###,##0.000', aCoefFluF[0]);
  EDB_FluPP.Text := FormatFloat('###,###,##0.000', aCoefFluF[1]);
  EDB_FluLaje.Text := FormatFloat('###,###,##0.000', aCoefFluF[2]);
  EDB_FluCapa.Text := FormatFloat('###,###,##0.000', aCoefFluF[3]);
  EDB_FluParede.Text := FormatFloat('###,###,##0.000', aCoefFluF[4]);
  EDB_FluRevest.Text := FormatFloat('###,###,##0.000', aCoefFluF[5]);
  EDB_FluAcid.Text := FormatFloat('###,###,##0.000', aCoefFluF[6]);
  EDB_FluPerda.Text := FormatFloat('###,###,##0.000', aCoefFluF[7]);

  aNp := GetSecNp(vABarra, lLpi, lLps, vL, aNLp, aTenInf, aTenSup);
  aMSecao[0] := aMomentos[0].S1; aMSecao[1] := aMomentos[1].S1; aMSecao[2]
:= aMomentos[2].S1;
  aMSecao[3] := aMomentos[3].S1; aMSecao[4] := aMomentos[4].S1; aMSecao[5]
:= aMomentos[5].S1;
  aMSecao[6] := aMomentos[6].S1;
  aFluInf.S1 := PerdaFluInf(aNp[1].X, aNp[1].Y, vA, lEi, lEs, lEiC, vIx,
tIxY.X, lPsi2, lAlfaPF, aMSecao, aCoefFluF);
  aFluSup.S1 := PerdaFluSup(aNp[1].X, aNp[1].Y, vA, lEi, lEs, lEsC, vIx,
tIxY.X, lPsi2, lAlfaPF, aMSecao, aCoefFluF);
  aMSecao[0] := aMomentos[0].S2; aMSecao[1] := aMomentos[1].S2; aMSecao[2]
:= aMomentos[2].S2;
  aMSecao[3] := aMomentos[3].S2; aMSecao[4] := aMomentos[4].S2; aMSecao[5]
:= aMomentos[5].S2;
```

```
  aMSecao[6] := aMomentos[6].S2;
  aFluInf.S2 := PerdaFluInf(aNp[2].X, aNp[2].Y, vA, lEi, lEs, lEiC, vIx,
tIxY.X, lPsi2, lAlfaPF, aMSecao, aCoefFluF);
  aFluSup.S2 := PerdaFluSup(aNp[2].X, aNp[2].Y, vA, lEi, lEs, lEsC, vIx,
tIxY.X, lPsi2, lAlfaPF, aMSecao, aCoefFluF);
  aMSecao[0] := aMomentos[0].S3; aMSecao[1] := aMomentos[1].S3; aMSecao[2]
:= aMomentos[2].S3;
  aMSecao[3] := aMomentos[3].S3; aMSecao[4] := aMomentos[4].S3; aMSecao[5]
:= aMomentos[5].S3;
  aMSecao[6] := aMomentos[6].S3;
  aFluInf.S3 := PerdaFluInf(aNp[3].X, aNp[3].Y, vA, lEi, lEs, lEiC, vIx,
tIxY.X, lPsi2, lAlfaPF, aMSecao, aCoefFluF);
  aFluSup.S3 := PerdaFluSup(aNp[3].X, aNp[3].Y, vA, lEi, lEs, lEsC, vIx,
tIxY.X, lPsi2, lAlfaPF, aMSecao, aCoefFluF);
  aMSecao[0] := aMomentos[0].S4; aMSecao[1] := aMomentos[1].S4; aMSecao[2]
:= aMomentos[2].S4;
  aMSecao[3] := aMomentos[3].S4; aMSecao[4] := aMomentos[4].S4; aMSecao[5]
:= aMomentos[5].S4;
  aMSecao[6] := aMomentos[6].S4;
  aFluInf.S4 := PerdaFluInf(aNp[4].X, aNp[4].Y, vA, lEi, lEs, lEiC, vIx,
tIxY.X, lPsi2, lAlfaPF, aMSecao, aCoefFluF);
  aFluSup.S4 := PerdaFluSup(aNp[4].X, aNp[4].Y, vA, lEi, lEs, lEsC, vIx,
tIxY.X, lPsi2, lAlfaPF, aMSecao, aCoefFluF);
  aMSecao[0] := aMomentos[0].S5; aMSecao[1] := aMomentos[1].S5; aMSecao[2]
:= aMomentos[2].S5;
  aMSecao[3] := aMomentos[3].S5; aMSecao[4] := aMomentos[4].S5; aMSecao[5]
:= aMomentos[5].S5;
  aMSecao[6] := aMomentos[6].S5;
  aFluInf.S5 := PerdaFluInf(aNp[5].X, aNp[5].Y, vA, lEi, lEs, lEiC, vIx,
tIxY.X, lPsi2, lAlfaPF, aMSecao, aCoefFluF);
  aFluSup.S5 := PerdaFluSup(aNp[5].X, aNp[5].Y, vA, lEi, lEs, lEsC, vIx,
tIxY.X, lPsi2, lAlfaPF, aMSecao, aCoefFluF);
  aMSecao[0] := aMomentos[0].SLpi; aMSecao[1] := aMomentos[1].SLpi;
aMSecao[2] := aMomentos[2].SLpi;
  aMSecao[3] := aMomentos[3].SLpi; aMSecao[4] := aMomentos[4].SLpi;
aMSecao[5] := aMomentos[5].SLpi;
  aMSecao[6] := aMomentos[6].SLpi;
  aFluInf.SLpi := PerdaFluInf(aNp[6].X, aNp[6].Y, vA, lEi, lEs, lEiC, vIx,
tIxY.X, lPsi2, lAlfaPF, aMSecao, aCoefFluF);
  aFluSup.SLpi := PerdaFluSup(aNp[6].X, aNp[6].Y, vA, lEi, lEs, lEsC, vIx,
tIxY.X, lPsi2, lAlfaPF, aMSecao, aCoefFluF);
  aMSecao[0] := aMomentos[0].SLps; aMSecao[1] := aMomentos[1].SLps;
aMSecao[2] := aMomentos[2].SLps;
  aMSecao[3] := aMomentos[3].SLps; aMSecao[4] := aMomentos[4].SLps;
aMSecao[5] := aMomentos[5].SLps;
  aMSecao[6] := aMomentos[6].SLps;
  aFluInf.SLps := PerdaFluInf(aNp[7].X, aNp[7].Y, vA, lEi, lEs, lEiC, vIx,
tIxY.X, lPsi2, lAlfaPF, aMSecao, aCoefFluF);
  aFluSup.SLps := PerdaFluSup(aNp[7].X, aNp[7].Y, vA, lEi, lEs, lEsC, vIx,
tIxY.X, lPsi2, lAlfaPF, aMSecao, aCoefFluF);
  aPerda[0].X := aPerda[0].X + aFluInf.S1; aPerda[1].X := aPerda[1].X +
aFluInf.S2;
  aPerda[2].X := aPerda[2].X + aFluInf.S3; aPerda[3].X := aPerda[3].X +
aFluInf.S4;
  aPerda[4].X := aPerda[4].X + aFluInf.S5; aPerda[5].X := aPerda[5].X +
aFluInf.SLpi;
  aPerda[6].X := aPerda[6].X + aFluInf.SLps; aPerda[0].Y := aPerda[0].Y +
aFluSup.S1;
  aPerda[1].Y := aPerda[1].Y + aFluSup.S2; aPerda[2].Y := aPerda[2].Y +
aFluSup.S3;
```

```
  aPerda[3].Y := aPerda[3].Y + aFluSup.S4; aPerda[4].Y := aPerda[4].Y +
aFluSup.S5;
  aPerda[5].Y := aPerda[5].Y + aFluSup.SLpi; aPerda[6].Y := aPerda[6].Y +
aFluSup.SLps;
  SGR_PerInfDif.Cells[1,1] := IntToStr(Round(aFluInf.S1));
  SGR_PerInfDif.Cells[2,1] := IntToStr(Round(aFluInf.S2));
  SGR_PerInfDif.Cells[3,1] := IntToStr(Round(aFluInf.S3));
  SGR_PerInfDif.Cells[4,1] := IntToStr(Round(aFluInf.S4));
  SGR_PerInfDif.Cells[5,1] := IntToStr(Round(aFluInf.S5));
  SGR_PerInfDif.Cells[6,1] := IntToStr(Round(aFluInf.SLpi));
  SGR_PerInfDif.Cells[7,1] := IntToStr(Round(aFluInf.SLps));
  SGR_PerSupDif.Cells[1,1] := IntToStr(Round(aFluSup.S1));
  SGR_PerSupDif.Cells[2,1] := IntToStr(Round(aFluSup.S2));
  SGR_PerSupDif.Cells[3,1] := IntToStr(Round(aFluSup.S3));
  SGR_PerSupDif.Cells[4,1] := IntToStr(Round(aFluSup.S4));
  SGR_PerSupDif.Cells[5,1] := IntToStr(Round(aFluSup.S5));
  SGR_PerSupDif.Cells[6,1] := IntToStr(Round(aFluSup.SLpi));
  SGR_PerSupDif.Cells[7,1] := IntToStr(Round(aFluSup.SLps));
  //Perda por fluência para cada acréscimo de carregamento (para o cálculo
das flechas)
    //Para a seção do meio do vão
  aCoefFluF[0] := 0; aCoefFluF[1] := 0; aCoefFluF[2] := 0; aCoefFluF[3] :=
0; aCoefFluF[4] := 0;
  aCoefFluF[5] := 0; aCoefFluF[6] := 0; aCoefFluF[7] := 0;
      //Entrada de peso próprio + laje
  aCoefFlu1[0] := CoefFlu(lPMConc, lPMTMedia, vDProt, vDLaje, lPMUmidade,
lPMAc, lPMUAr, vFid, lPMSlump, lPMEndur, aAlfa, aCoefS, aFc);
  aCoefFlu1[1] := CoefFlu(lPMConc, lPMTMedia, vDPP, vDLaje, lPMUmidade,
lPMAc, lPMUAr, vFid, lPMSlump, lPMEndur, aAlfa, aCoefS, aFc);
  aCoefFluF[0] := aCoefFlu1[0];
  aCoefFluF[1] := aCoefFlu1[1];
  aMSecao[0] := aMomentos[0].S5; aMSecao[1] := aMomentos[1].S5;
  aMSecao[2] := 0; aMSecao[3] := 0; aMSecao[4] := 0; aMSecao[5] := 0;
aMSecao[6] := 0;
  aPerdI5[2] := aPerdI5[2] + PerdaFluInf(aNp[5].X, aNp[5].Y, vA, lEi, lEs,
lEiC, vIx, tIxY.X, 1, vEp/aEci[1].X, aMSecao, aCoefFluF);
  aPerdS5[2] := aPerdS5[2] + PerdaFluSup(aNp[5].X, aNp[5].Y, vA, lEi, lEs,
lEsC, vIx, tIxY.X, 1, vEp/aEci[1].X, aMSecao, aCoefFluF);
      //Entrada da capa
  aCoefFlu1[0] := CoefFlu(lPMConc, lPMTMedia, vDProt, vDCapa, lPMUmidade,
lPMAc, lPMUAr, vFid, lPMSlump, lPMEndur, aAlfa, aCoefS, aFc);
  aCoefFlu1[1] := CoefFlu(lPMConc, lPMTMedia, vDPP, vDCapa, lPMUmidade,
lPMAc, lPMUAr, vFid, lPMSlump, lPMEndur, aAlfa, aCoefS, aFc);
  aCoefFlu1[2] := CoefFlu(lPMConc, lPMTMedia, vDLaje, vDCapa, lPMUmidade,
lPMAc, lPMUAr, vFid, lPMSlump, lPMEndur, aAlfa, aCoefS, aFc);
  aCoefFluF[0] := aCoefFlu1[0];
  aCoefFluF[1] := aCoefFlu1[1];
  aCoefFluF[2] := aCoefFlu1[2];
  aMSecao[2] := aMomentos[2].S5;
  aPerdI5[3] := aPerdI5[3] + PerdaFluInf(aNp[5].X, aNp[5].Y, vA, lEi, lEs,
lEiC, vIx, tIxY.X, 1, vEp/aEci[2].X, aMSecao, aCoefFluF);
  aPerdS5[3] := aPerdS5[3] + PerdaFluSup(aNp[5].X, aNp[5].Y, vA, lEi, lEs,
lEsC, vIx, tIxY.X, 1, vEp/aEci[2].X, aMSecao, aCoefFluF);
      //Entrada da parede
  aCoefFlu1[0] := CoefFlu(lPMConc, lPMTMedia, vDProt, vDParede, lPMUmidade,
lPMAc, lPMUAr, vFid, lPMSlump, lPMEndur, aAlfa, aCoefS, aFc);
  aCoefFlu1[1] := CoefFlu(lPMConc, lPMTMedia, vDPP, vDParede, lPMUmidade,
lPMAc, lPMUAr, vFid, lPMSlump, lPMEndur, aAlfa, aCoefS, aFc);
  aCoefFlu1[2] := CoefFlu(lPMConc, lPMTMedia, vDLaje, vDParede, lPMUmidade,
lPMAc, lPMUAr, vFid, lPMSlump, lPMEndur, aAlfa, aCoefS, aFc);
```

```
  aCoefFlu1[3] := CoefFlu(lPMConc, lPMTMedia, vDCapa, vDParede, lPMUmidade,
lPMAc, lPMUAr, vFid, lPMSlump, lPMEndur, aAlfa, aCoefS, aFc);
  aCoefFlu2[3] := CoefFlu(lILConc, lILTMedia, 1, vDParede-vDCapa,
lILUmidade, lILAc, lILUAr, vFid, lILSlump, lILEndur, aAlfa, aCoefS, aFc);
  aCoefFluF[0] := aCoefFlu1[0];
  aCoefFluF[1] := aCoefFlu1[1];
  aCoefFluF[2] := aCoefFlu1[2];
  aCoefFluF[3] := MedPon(lPMAc, lILAc, aCoefFlu1[3], aCoefFlu2[3]);
  aMSecao[3] := aMomentos[3].S5;
  aPerdI5[4] := aPerdI5[4] + PerdaFluInf(aNp[5].X, aNp[5].Y, vA, lEi, lEs,
lEiC, vIx, tIxY.X, 1, vEp/aEci[3].X, aMSecao, aCoefFluF);
  aPerdS5[4] := aPerdS5[4] + PerdaFluSup(aNp[5].X, aNp[5].Y, vA, lEi, lEs,
lEsC, vIx, tIxY.X, 1, vEp/aEci[3].X, aMSecao, aCoefFluF);
      //Entrada do revestimento
  aCoefFlu1[0] := CoefFlu(lPMConc, lPMTMedia, vDProt, vDRevest, lPMUmidade,
lPMAc, lPMUAr, vFid, lPMSlump, lPMEndur, aAlfa, aCoefS, aFc);
  aCoefFlu1[1] := CoefFlu(lPMConc, lPMTMedia, vDPP, vDRevest, lPMUmidade,
lPMAc, lPMUAr, vFid, lPMSlump, lPMEndur, aAlfa, aCoefS, aFc);
  aCoefFlu1[2] := CoefFlu(lPMConc, lPMTMedia, vDLaje, vDRevest, lPMUmidade,
lPMAc, lPMUAr, vFid, lPMSlump, lPMEndur, aAlfa, aCoefS, aFc);
  aCoefFlu1[3] := CoefFlu(lPMConc, lPMTMedia, vDCapa, vDRevest, lPMUmidade,
lPMAc, lPMUAr, vFid, lPMSlump, lPMEndur, aAlfa, aCoefS, aFc);
  aCoefFlu1[4] := CoefFlu(lPMConc, lPMTMedia, vDParede, vDRevest,
lPMUmidade, lPMAc, lPMUAr, vFid, lPMSlump, lPMEndur, aAlfa, aCoefS, aFc);
  aCoefFlu2[3] := CoefFlu(lILConc, lILTMedia, 1, vDRevest-vDCapa,
lILUmidade, lILAc, lILUAr, vFid, lILSlump, lILEndur, aAlfa, aCoefS, aFc);
  aCoefFlu2[4] := CoefFlu(lILConc, lILTMedia, vDParede-vDCapa, vDRevest-
vDParede, lILUmidade, lILAc, lILUAr, vFid, lILSlump, lILEndur, aAlfa,
aCoefS, aFc);
  aCoefFluF[0] := aCoefFlu1[0];
  aCoefFluF[1] := aCoefFlu1[1];
  aCoefFluF[2] := aCoefFlu1[2];
  aCoefFluF[3] := MedPon(lPMAc, lILAc, aCoefFlu1[3], aCoefFlu2[3]);
  aCoefFluF[4] := MedPon(lPMAc, lILAc, aCoefFlu1[4], aCoefFlu2[4]);
  aMSecao[4] := aMomentos[4].S5;
  aPerdI5[5] := aPerdI5[5] + PerdaFluInf(aNp[5].X, aNp[5].Y, vA, lEi, lEs,
lEiC, vIx, tIxY.X, 1, vEp/aEci[4].X, aMSecao, aCoefFluF);
  aPerdS5[5] := aPerdS5[5] + PerdaFluSup(aNp[5].X, aNp[5].Y, vA, lEi, lEs,
lEsC, vIx, tIxY.X, 1, vEp/aEci[4].X, aMSecao, aCoefFluF);
      //Entrada da acidental
  aCoefFlu1[0] := CoefFlu(lPMConc, lPMTMedia, vDProt, vDAcid, lPMUmidade,
lPMAc, lPMUAr, vFid, lPMSlump, lPMEndur, aAlfa, aCoefS, aFc);
  aCoefFlu1[1] := CoefFlu(lPMConc, lPMTMedia, vDPP, vDAcid, lPMUmidade,
lPMAc, lPMUAr, vFid, lPMSlump, lPMEndur, aAlfa, aCoefS, aFc);
  aCoefFlu1[2] := CoefFlu(lPMConc, lPMTMedia, vDLaje, vDAcid, lPMUmidade,
lPMAc, lPMUAr, vFid, lPMSlump, lPMEndur, aAlfa, aCoefS, aFc);
  aCoefFlu1[3] := CoefFlu(lPMConc, lPMTMedia, vDCapa, vDAcid, lPMUmidade,
lPMAc, lPMUAr, vFid, lPMSlump, lPMEndur, aAlfa, aCoefS, aFc);
  aCoefFlu1[4] := CoefFlu(lPMConc, lPMTMedia, vDParede, vDAcid, lPMUmidade,
lPMAc, lPMUAr, vFid, lPMSlump, lPMEndur, aAlfa, aCoefS, aFc);
  aCoefFlu1[5] := CoefFlu(lPMConc, lPMTMedia, vDRevest, vDAcid, lPMUmidade,
lPMAc, lPMUAr, vFid, lPMSlump, lPMEndur, aAlfa, aCoefS, aFc);
  aCoefFlu2[3] := CoefFlu(lILConc, lILTMedia, 1, vDAcid-vDCapa, lILUmidade,
lILAc, lILUAr, vFid, lILSlump, lILEndur, aAlfa, aCoefS, aFc);
  aCoefFlu2[4] := CoefFlu(lILConc, lILTMedia, vDParede-vDCapa, vDAcid-
vDParede, lILUmidade, lILAc, lILUAr, vFid, lILSlump, lILEndur, aAlfa,
aCoefS, aFc);
  aCoefFlu2[5] := CoefFlu(lILConc, lILTMedia, vDRevest-vDCapa, vDAcid-
vDRevest, lILUmidade, lILAc, lILUAr, vFid, lILSlump, lILEndur, aAlfa,
aCoefS, aFc);
  aCoefFluF[0] := aCoefFlu1[0];
```

```
  aCoefFluF[1] := aCoefFlu1[1];
  aCoefFluF[2] := aCoefFlu1[2];
  aCoefFluF[3] := MedPon(lPMAc, lILAc, aCoefFlu1[3], aCoefFlu2[3]);
  aCoefFluF[4] := MedPon(lPMAc, lILAc, aCoefFlu1[4], aCoefFlu2[4]);
  aCoefFluF[5] := MedPon(lPMAc, lILAc, aCoefFlu1[5], aCoefFlu2[5]);
  aMSecao[5] := aMomentos[5].S5;
  aPerdI5[6] := aPerdI5[6] + PerdaFluInf(aNp[5].X, aNp[5].Y, vA, lEi, lEs,
lEiC, vIx, tIxY.X, 1, vEp/aEci[5].X, aMSecao, aCoefFluF);
  aPerdS5[6] := aPerdS5[6] + PerdaFluSup(aNp[5].X, aNp[5].Y, vA, lEi, lEs,
lEsC, vIx, tIxY.X, 1, vEp/aEci[5].X, aMSecao, aCoefFluF);
  //Perda final por relaxação da armadura
  aPerdI5[2] := aPerdI5[2] + RelArm(aTenInf.S5, lFptk, vDPP, vDLaje,
aPsiMil, aR);
  aPerdI5[3] := aPerdI5[3] + RelArm(aTenInf.S5, lFptk, vDPP, vDCapa,
aPsiMil, aR);
  aPerdI5[4] := aPerdI5[4] + RelArm(aTenInf.S5, lFptk, vDPP, vDParede,
aPsiMil, aR);
  aPerdI5[5] := aPerdI5[5] + RelArm(aTenInf.S5, lFptk, vDPP, vDRevest,
aPsiMil, aR);
  aPerdI5[6] := aPerdI5[6] + RelArm(aTenInf.S5, lFptk, vDPP, vDAcid,
aPsiMil, aR);
  aPerdS5[2] := aPerdS5[2] + RelArm(aTenSup.S5, lFptk, vDPP, vDLaje,
aPsiMil, aR);
  aPerdS5[3] := aPerdS5[3] + RelArm(aTenSup.S5, lFptk, vDPP, vDCapa,
aPsiMil, aR);
  aPerdS5[4] := aPerdS5[4] + RelArm(aTenSup.S5, lFptk, vDPP, vDParede,
aPsiMil, aR);
  aPerdS5[5] := aPerdS5[5] + RelArm(aTenSup.S5, lFptk, vDPP, vDRevest,
aPsiMil, aR);
  aPerdS5[6] := aPerdS5[6] + RelArm(aTenSup.S5, lFptk, vDPP, vDAcid,
aPsiMil, aR);
  aPsiRelFI.S1 := RelArm(aTenInf.S1, lFptk, vDPP, vTInf, aPsiMil, aR);
  aPsiRelFI.S2 := RelArm(aTenInf.S2, lFptk, vDPP, vTInf, aPsiMil, aR);
  aPsiRelFI.S3 := RelArm(aTenInf.S3, lFptk, vDPP, vTInf, aPsiMil, aR);
  aPsiRelFI.S4 := RelArm(aTenInf.S4, lFptk, vDPP, vTInf, aPsiMil, aR);
  aPsiRelFI.S5 := RelArm(aTenInf.S5, lFptk, vDPP, vTInf, aPsiMil, aR);
  aPsiRelFI.SLpi := RelArm(aTenInf.SLpi, lFptk, vDPP, vTInf, aPsiMil, aR);
  aPsiRelFI.SLps := RelArm(aTenInf.SLps, lFptk, vDPP, vTInf, aPsiMil, aR);
  aPsiRelFS.S1 := RelArm(aTenSup.S1, lFptk, vDPP, vTInf, aPsiMil, aR);
  aPsiRelFS.S2 := RelArm(aTenSup.S2, lFptk, vDPP, vTInf, aPsiMil, aR);
  aPsiRelFS.S3 := RelArm(aTenSup.S3, lFptk, vDPP, vTInf, aPsiMil, aR);
  aPsiRelFS.S4 := RelArm(aTenSup.S4, lFptk, vDPP, vTInf, aPsiMil, aR);
  aPsiRelFS.S5 := RelArm(aTenSup.S5, lFptk, vDPP, vTInf, aPsiMil, aR);
  aPsiRelFS.SLpi := RelArm(aTenSup.SLpi, lFptk, vDPP, vTInf, aPsiMil, aR);
  aPsiRelFS.SLps := RelArm(aTenSup.SLps, lFptk, vDPP, vTInf, aPsiMil, aR);
  aRelArmFI.S1 := aTenInf.S1 * aPsiRelFI.S1;
  aRelArmFI.S2 := aTenInf.S2 * aPsiRelFI.S2;
  aRelArmFI.S3 := aTenInf.S3 * aPsiRelFI.S3;
  aRelArmFI.S4 := aTenInf.S4 * aPsiRelFI.S4;
  aRelArmFI.S5 := aTenInf.S5 * aPsiRelFI.S5;
  aRelArmFI.SLpi := aTenInf.SLpi * aPsiRelFI.SLpi;
  aRelArmFI.SLps := aTenInf.SLps * aPsiRelFI.SLps;
  aRelArmFS.S1 := aTenSup.S1 * aPsiRelFS.S1;
  aRelArmFS.S2 := aTenSup.S2 * aPsiRelFS.S2;
  aRelArmFS.S3 := aTenSup.S3 * aPsiRelFS.S3;
  aRelArmFS.S4 := aTenSup.S4 * aPsiRelFS.S4;
  aRelArmFS.S5 := aTenSup.S5 * aPsiRelFS.S5;
  aRelArmFS.SLpi := aTenSup.SLpi * aPsiRelFS.SLpi;
  aRelArmFS.SLps := aTenSup.SLps * aPsiRelFS.SLps;
  aPerda[0].X := aPerda[0].X + aRelArmFI.S1; aPerda[1].X := aPerda[1].X +
aRelArmFI.S2;
```

```
  aPerda[2].X := aPerda[2].X + aRelArmFI.S3; aPerda[3].X := aPerda[3].X +
aRelArmFI.S4;
  aPerda[4].X := aPerda[4].X + aRelArmFI.S5; aPerda[5].X := aPerda[5].X +
aRelArmFI.SLpi;
  aPerda[6].X := aPerda[6].X + aRelArmFI.SLps; aPerda[0].Y := aPerda[0].Y +
aRelArmFS.S1;
  aPerda[1].Y := aPerda[1].Y + aRelArmFS.S2; aPerda[2].Y := aPerda[2].Y +
aRelArmFS.S3;
  aPerda[3].Y := aPerda[3].Y + aRelArmFS.S4; aPerda[4].Y := aPerda[4].Y +
aRelArmFS.S5;
  aPerda[5].Y := aPerda[5].Y + aRelArmFS.Slpi; aPerda[6].Y := aPerda[6].Y +
aRelArmFS.SLps;
  SGR_PerInfDif.Cells[1,2] := IntToStr(Round(aRelArmFI.S1));
  SGR_PerInfDif.Cells[2,2] := IntToStr(Round(aRelArmFI.S2));
  SGR_PerInfDif.Cells[3,2] := IntToStr(Round(aRelArmFI.S3));
  SGR_PerInfDif.Cells[4,2] := IntToStr(Round(aRelArmFI.S4));
  SGR_PerInfDif.Cells[5,2] := IntToStr(Round(aRelArmFI.S5));
  SGR_PerInfDif.Cells[6,2] := IntToStr(Round(aRelArmFI.SLpi));
  SGR_PerInfDif.Cells[7,2] := IntToStr(Round(aRelArmFI.SLps));
  SGR_PerSupDif.Cells[1,2] := IntToStr(Round(aRelArmFS.S1));
  SGR_PerSupDif.Cells[2,2] := IntToStr(Round(aRelArmFS.S2));
  SGR_PerSupDif.Cells[3,2] := IntToStr(Round(aRelArmFS.S3));
  SGR_PerSupDif.Cells[4,2] := IntToStr(Round(aRelArmFS.S4));
  SGR_PerSupDif.Cells[5,2] := IntToStr(Round(aRelArmFS.S5));
  SGR_PerSupDif.Cells[6,2] := IntToStr(Round(aRelArmFS.SLpi));
  SGR_PerSupDif.Cells[7,2] := IntToStr(Round(aRelArmFS.SLps));

  SGR_PerInfDif.Cells[1,3] := IntToStr(Round(aPerda[0].X));
  SGR_PerInfDif.Cells[2,3] := IntToStr(Round(aPerda[1].X));
  SGR_PerInfDif.Cells[3,3] := IntToStr(Round(aPerda[2].X));
  SGR_PerInfDif.Cells[4,3] := IntToStr(Round(aPerda[3].X));
  SGR_PerInfDif.Cells[5,3] := IntToStr(Round(aPerda[4].X));
  SGR_PerInfDif.Cells[6,3] := IntToStr(Round(aPerda[5].X));
  SGR_PerInfDif.Cells[7,3] := IntToStr(Round(aPerda[6].X));
  SGR_PerSupDif.Cells[1,3] := IntToStr(Round(aPerda[0].Y));
  SGR_PerSupDif.Cells[2,3] := IntToStr(Round(aPerda[1].Y));
  SGR_PerSupDif.Cells[3,3] := IntToStr(Round(aPerda[2].Y));
  SGR_PerSupDif.Cells[4,3] := IntToStr(Round(aPerda[3].Y));
  SGR_PerSupDif.Cells[5,3] := IntToStr(Round(aPerda[4].Y));
  SGR_PerSupDif.Cells[6,3] := IntToStr(Round(aPerda[5].Y));
  SGR_PerSupDif.Cells[7,3] := IntToStr(Round(aPerda[6].Y));
  EDB_PTDifI1.Text := FormatFloat('###,###,##0.0', aPerda[0].X / lSigInf *
100);
  EDB_PTDifI2.Text := FormatFloat('###,###,##0.0', aPerda[1].X / lSigInf *
100);
  EDB_PTDifI3.Text := FormatFloat('###,###,##0.0', aPerda[2].X / lSigInf *
100);
  EDB_PTDifI4.Text := FormatFloat('###,###,##0.0', aPerda[3].X / lSigInf *
100);
  EDB_PTDifI5.Text := FormatFloat('###,###,##0.0', aPerda[4].X / lSigInf *
100);
  EDB_PTDifI6.Text := FormatFloat('###,###,##0.0', aPerda[5].X / lSigInf *
100);
  EDB_PTDifI7.Text := FormatFloat('###,###,##0.0', aPerda[6].X / lSigInf *
100);
  EDB_PTDifS1.Text := FormatFloat('###,###,##0.0', aPerda[0].Y / lSigSup *
100);
  EDB_PTDifS2.Text := FormatFloat('###,###,##0.0', aPerda[1].Y / lSigSup *
100);
  EDB_PTDifS3.Text := FormatFloat('###,###,##0.0', aPerda[2].Y / lSigSup *
100);
```

```
  EDB_PTDifS4.Text := FormatFloat('###,###,##0.0', aPerda[3].Y / lSigSup *
100);
  EDB_PTDifS5.Text := FormatFloat('###,###,##0.0', aPerda[4].Y / lSigSup *
100);
  EDB_PTDifS6.Text := FormatFloat('###,###,##0.0', aPerda[5].Y / lSigSup *
100);
  EDB_PTDifS7.Text := FormatFloat('###,###,##0.0', aPerda[6].Y / lSigSup *
100);
  //Perdas progressivas
  aProgrI.S1 := Progressiva(aFluInf.S1, rRetra, aRelArmFI.S1, aPsiRelFI.S1,
aCoefFluF[0], lEi, vA, vIx, (vNCordI1+vNCordS5)*vABarra, lAlfaPF);
  aProgrI.S2 := Progressiva(aFluInf.S2, rRetra, aRelArmFI.S2, aPsiRelFI.S2,
aCoefFluF[0], lEi, vA, vIx, (vNCordI2+vNCordS5)*vABarra, lAlfaPF);
  aProgrI.S3 := Progressiva(aFluInf.S3, rRetra, aRelArmFI.S3, aPsiRelFI.S3,
aCoefFluF[0], lEi, vA, vIx, (vNCordI3+vNCordS5)*vABarra, lAlfaPF);
  aProgrI.S4 := Progressiva(aFluInf.S4, rRetra, aRelArmFI.S4, aPsiRelFI.S4,
aCoefFluF[0], lEi, vA, vIx, (vNCordI4+vNCordS5)*vABarra, lAlfaPF);
  aProgrI.S5 := Progressiva(aFluInf.S5, rRetra, aRelArmFI.S5, aPsiRelFI.S5,
aCoefFluF[0], lEi, vA, vIx, (vNCordI5+vNCordS5)*vABarra, lAlfaPF);
  aProgrI.SLpi := Progressiva(aFluInf.SLpi, rRetra, aRelArmFI.SLpi,
aPsiRelFI.SLpi, aCoefFluF[0], lEi, vA, vIx, (vNCordILpi+vNCordS5)*vABarra,
lAlfaPF);
  aProgrI.SLps := Progressiva(aFluInf.SLps, rRetra, aRelArmFI.SLps,
aPsiRelFI.SLps, aCoefFluF[0], lEi, vA, vIx, (vNCordILps+vNCordS5)*vABarra,
lAlfaPF);
  aProgrS.S1 := Progressiva(aFluSup.S1, rRetra, aRelArmFS.S1, aPsiRelFS.S1,
aCoefFluF[0], lEs, vA, vIx, (vNCordI1+vNCordS5)*vABarra, lAlfaPF);
  aProgrS.S2 := Progressiva(aFluSup.S2, rRetra, aRelArmFS.S2, aPsiRelFS.S2,
aCoefFluF[0], lEs, vA, vIx, (vNCordI2+vNCordS5)*vABarra, lAlfaPF);
  aProgrS.S3 := Progressiva(aFluSup.S3, rRetra, aRelArmFS.S3, aPsiRelFS.S3,
aCoefFluF[0], lEs, vA, vIx, (vNCordI3+vNCordS5)*vABarra, lAlfaPF);
  aProgrS.S4 := Progressiva(aFluSup.S4, rRetra, aRelArmFS.S4, aPsiRelFS.S4,
aCoefFluF[0], lEs, vA, vIx, (vNCordI4+vNCordS5)*vABarra, lAlfaPF);
  aProgrS.S5 := Progressiva(aFluSup.S5, rRetra, aRelArmFS.S5, aPsiRelFS.S5,
aCoefFluF[0], lEs, vA, vIx, (vNCordI5+vNCordS5)*vABarra, lAlfaPF);
  aProgrS.SLpi := Progressiva(aFluSup.SLpi, rRetra, aRelArmFS.SLpi,
aPsiRelFS.SLpi, aCoefFluF[0], lEs, vA, vIx, (vNCordILpi+vNCordS5)*vABarra,
lAlfaPF);
  aProgrS.SLps := Progressiva(aFluSup.SLps, rRetra, aRelArmFS.SLps,
aPsiRelFS.SLps, aCoefFluF[0], lEs, vA, vIx, (vNCordILps+vNCordS5)*vABarra,
lAlfaPF);
  SGR_PerInfDif.Cells[1,4] := IntToStr(Round(aProgrI.S1));
  SGR_PerInfDif.Cells[2,4] := IntToStr(Round(aProgrI.S2));
  SGR_PerInfDif.Cells[3,4] := IntToStr(Round(aProgrI.S3));
  SGR_PerInfDif.Cells[4,4] := IntToStr(Round(aProgrI.S4));
  SGR_PerInfDif.Cells[5,4] := IntToStr(Round(aProgrI.S5));
  SGR_PerInfDif.Cells[6,4] := IntToStr(Round(aProgrI.SLpi));
  SGR_PerInfDif.Cells[7,4] := IntToStr(Round(aProgrI.SLps));
  SGR_PerSupDif.Cells[1,4] := IntToStr(Round(aProgrS.S1));
  SGR_PerSupDif.Cells[2,4] := IntToStr(Round(aProgrS.S2));
  SGR_PerSupDif.Cells[3,4] := IntToStr(Round(aProgrS.S3));
  SGR_PerSupDif.Cells[4,4] := IntToStr(Round(aProgrS.S4));
  SGR_PerSupDif.Cells[5,4] := IntToStr(Round(aProgrS.S5));
  SGR_PerSupDif.Cells[6,4] := IntToStr(Round(aProgrS.SLpi));
  SGR_PerSupDif.Cells[7,4] := IntToStr(Round(aProgrS.SLps));

  //Perdas totais nas idades de cada carregamento
  aPerdS[0] := (rDefAnc + rRelArmS) / lSigS * 100;
  aPerdI[0] := (rDefAnc + rRelArmI) / lSigI * 100;
  aPerdI5[0] := (rDefAnc + rRelArmI) / lSigI * 100;
  aPerdI5[1] := (rDefAnc + rRelArmI + aDefConInf.S5) / lSigI * 100;
```

```
  aPerdI5[2] := aPerdI5[2] / lSigI * 100 + aPerdI5[1];
  aPerdI5[3] := aPerdI5[3] / lSigI * 100 + aPerdI5[1];
  aPerdI5[4] := aPerdI5[4] / lSigI * 100 + aPerdI5[1];
  aPerdI5[5] := aPerdI5[5] / lSigI * 100 + aPerdI5[1];
  aPerdI5[6] := aPerdI5[6] / lSigI * 100 + aPerdI5[1];
  aPerdI5[7] := aPerdI5[1] + (rRetra + aFluInf.S5 + aRelArmFI.S5) / lSigI *
100;
  aPerdS5[0] := (rDefAnc + rRelArmS) / lSigS * 100;
  aPerdS5[1] := (rDefAnc + rRelArmS + aDefConSup.S5) / lSigS * 100;
  aPerdS5[2] := aPerdS5[2] / lSigS * 100 + aPerdS5[1];
  aPerdS5[3] := aPerdS5[3] / lSigS * 100 + aPerdS5[1];
  aPerdS5[4] := aPerdS5[4] / lSigS * 100 + aPerdS5[1];
  aPerdS5[5] := aPerdS5[5] / lSigS * 100 + aPerdS5[1];
  aPerdS5[6] := aPerdS5[6] / lSigS * 100 + aPerdS5[1];
  aPerdS5[7] := aPerdS5[1] + (rRetra + aFluSup.S5 + aRelArmFS.S5) / lSigS *
100;
  //Perda de protensão para o cálculo da contraflecha (Perda por def
ancoragem + rel inicial)
  EDB_DPprotI.Text := IntToStr(Round(aPerdI[0]));
  EDB_DPprotS.Text := IntToStr(Round(aPerdS[0]));
  //Perda de protensão na seção S5
  EDB_DPppI.Text := IntToStr(Round(aPerdI5[1]));
  EDB_DPlajeI.Text := IntToStr(Round(aPerdI5[2]));
  EDB_DPcapaI.Text := IntToStr(Round(aPerdI5[3]));
  EDB_DPparedeI.Text := IntToStr(Round(aPerdI5[4]));
  EDB_DPrevestI.Text := IntToStr(Round(aPerdI5[5]));
  EDB_DPacidI.Text := IntToStr(Round(aPerdI5[6]));
  EDB_DPppS.Text := IntToStr(Round(aPerdS5[1]));
  EDB_DPlajeS.Text := IntToStr(Round(aPerdS5[2]));
  EDB_DPcapaS.Text := IntToStr(Round(aPerdS5[3]));
  EDB_DPparedeS.Text := IntToStr(Round(aPerdS5[4]));
  EDB_DPrevestS.Text := IntToStr(Round(aPerdS5[5]));
  EDB_DPacidS.Text := IntToStr(Round(aPerdS5[6]));

  //Verificando se há cordoalha superior para cálculo das perdas
  if aAp.Y = 0 then begin
    SGR_PerSupIni.Cells[1,1] := '-'; SGR_PerSupIni.Cells[2,1] := '-';
SGR_PerSupIni.Cells[3,1] := '-';
    SGR_PerSupIni.Cells[4,1] := '-'; SGR_PerSupIni.Cells[5,1] := '-';
SGR_PerSupIni.Cells[6,1] := '-';
    SGR_PerSupIni.Cells[7,1] := '-'; SGR_PerSupIni.Cells[1,2] := '-';
SGR_PerSupIni.Cells[2,2] := '-';
    SGR_PerSupIni.Cells[3,2] := '-'; SGR_PerSupIni.Cells[4,2] := '-';
SGR_PerSupIni.Cells[5,2] := '-';
    SGR_PerSupIni.Cells[6,2] := '-'; SGR_PerSupIni.Cells[7,2] := '-';
SGR_PerSupIni.Cells[1,3] := '-';
    SGR_PerSupIni.Cells[2,3] := '-'; SGR_PerSupIni.Cells[3,3] := '-';
SGR_PerSupIni.Cells[4,3] := '-';
    SGR_PerSupIni.Cells[5,3] := '-'; SGR_PerSupIni.Cells[6,3] := '-';
SGR_PerSupIni.Cells[7,3] := '-';
    SGR_PerSupIni.Cells[1,4] := '-'; SGR_PerSupIni.Cells[2,4] := '-';
SGR_PerSupIni.Cells[3,4] := '-';
    SGR_PerSupIni.Cells[4,4] := '-'; SGR_PerSupIni.Cells[5,4] := '-';
SGR_PerSupIni.Cells[6,4] := '-';
    SGR_PerSupIni.Cells[7,4] := '-'; SGR_PerSupDif.Cells[1,0] := '-';
SGR_PerSupDif.Cells[2,0] := '-';
    SGR_PerSupDif.Cells[3,0] := '-'; SGR_PerSupDif.Cells[4,0] := '-';
SGR_PerSupDif.Cells[5,0] := '-';
    SGR_PerSupDif.Cells[6,0] := '-'; SGR_PerSupDif.Cells[7,0] := '-';
SGR_PerSupDif.Cells[1,1] := '-';
```

```
    SGR_PerSupDif.Cells[2,1] := '-'; SGR_PerSupDif.Cells[3,1] := '-';
SGR_PerSupDif.Cells[4,1] := '-';
    SGR_PerSupDif.Cells[5,1] := '-'; SGR_PerSupDif.Cells[6,1] := '-';
SGR_PerSupDif.Cells[7,1] := '-';
    SGR_PerSupDif.Cells[1,2] := '-'; SGR_PerSupDif.Cells[2,2] := '-';
SGR_PerSupDif.Cells[3,2] := '-';
    SGR_PerSupDif.Cells[4,2] := '-'; SGR_PerSupDif.Cells[5,2] := '-';
SGR_PerSupDif.Cells[6,2] := '-';
    SGR_PerSupDif.Cells[7,2] := '-'; SGR_PerSupDif.Cells[1,3] := '-';
SGR_PerSupDif.Cells[2,3] := '-';
    SGR_PerSupDif.Cells[3,3] := '-'; SGR_PerSupDif.Cells[4,3] := '-';
SGR_PerSupDif.Cells[5,3] := '-';
    SGR_PerSupDif.Cells[6,3] := '-'; SGR_PerSupDif.Cells[7,3] := '-';
SGR_PerSupDif.Cells[1,4] := '-';
    SGR_PerSupDif.Cells[2,4] := '-'; SGR_PerSupDif.Cells[3,4] := '-';
SGR_PerSupDif.Cells[4,4] := '-';
    SGR_PerSupDif.Cells[5,4] := '-'; SGR_PerSupDif.Cells[6,4] := '-';
SGR_PerSupDif.Cells[7,4] := '-';
    EDB_DPprotS.Text := '';
    EDB_PTImedS1.Text := ''; EDB_PTImedS2.Text := ''; EDB_PTImedS3.Text :=
''; EDB_PTImedS4.Text := '';
    EDB_PTImedS5.Text := ''; EDB_PTImedS6.Text := ''; EDB_PTImedS7.Text :=
''; EDB_PTDifS1.Text := '';
    EDB_PTDifS2.Text := ''; EDB_PTDifS3.Text := ''; EDB_PTDifS4.Text := '';
EDB_PTDifS5.Text := '';
    EDB_PTDifS6.Text := ''; EDB_PTDifS7.Text := ''; EDB_DPppS.Text := '';
EDB_DPlajeS.Text := '';
    EDB_DPcapaS.Text := ''; EDB_DPparedeS.Text := ''; EDB_DPrevestS.Text :=
''; EDB_DPacidS.Text := '';
  end;

  //Módulo de flechas
  lSigI := StrToFloat(IsNil(EDB_TensaoIniI.Text));
  lSigS := StrToFloat(IsNil(EDB_TensaoIniS.Text));
  lPsi2 := StrToFloat(IsNil(EDB_psi2.Text));
  lDi := StrToFloat(SGR_Dimens.Cells[3,2]) * 100;
  lDs := StrToFloat(SGR_Dimens.Cells[3,3]) * 100;
  lDiS := lDi - (aDimensoes[0].Y + aDimensoes[1].Y) * 100;
  lDsS := lDs - (aDimensoes[0].Y + aDimensoes[1].Y) * 100;
  aSigI[0] := lSigI * (1 - aPerdI5[0] / 100);
  aSigI[1] := lSigI * (1 - aPerdI5[1] / 100);
  aSigI[2] := lSigI * (1 - aPerdI5[2] / 100);
  aSigI[3] := lSigI * (1 - aPerdI5[3] / 100);
  aSigI[4] := lSigI * (1 - aPerdI5[4] / 100);
  aSigI[5] := lSigI * (1 - aPerdI5[5] / 100);
  aSigI[6] := lSigI * (1 - aPerdI5[6] / 100);
  aSigI[7] := lSigI * (1 - aPerdI5[7] / 100);
  aSigS[0] := lSigS * (1 - aPerdS5[0] / 100);
  aSigS[1] := lSigS * (1 - aPerdS5[1] / 100);
  aSigS[2] := lSigS * (1 - aPerdS5[2] / 100);
  aSigS[3] := lSigS * (1 - aPerdS5[3] / 100);
  aSigS[4] := lSigS * (1 - aPerdS5[4] / 100);
  aSigS[5] := lSigS * (1 - aPerdS5[5] / 100);
  aSigS[6] := lSigS * (1 - aPerdS5[6] / 100);
  aSigS[7] := lSigS * (1 - aPerdS5[7] / 100);
  SetLength(aLDimensoes, 4);
  aLDimensoes[0] := aDimensoes[0];
  aLDimensoes[1] := aDimensoes[1];
  aLDimensoes[2] := aDimensoes[2];
  aLDimensoes[3] := aDimensoes[3];
  aLDimensoes[0].X := 0; aLDimensoes[0].Y := 0;
```

```
  aLDimensoes[1].X := 0; aLDimensoes[1].Y := 0;
  aInerciaIIs := InertiaII(lDiS, lDsS, aAp.X, aAp.Y, aAs.X, aAs.Y, vEp,
vEdoce, aLDimensoes);
  aInerciaII := InertiaII(lDi, lDs, aAp.X, aAp.Y, aAs.X, aAs.Y, vEp,
vEdoce, aDimensoes);
  //Im para carregamento de peso próprio
  aTenInf.S5 := aSigI[1];
  aTenSup.S5 := aSigS[1];
  aNp := GetSecNp(vABarra, lLpi, lLps, vL, aNLp, aTenInf, aTenSup);
  lMat := aMomentos[0].S5;
  lMr := MomFiss(aFck[0].X, aNp[5].X, aNp[5].Y, lEi, lEs, vAlfaForma,
vAlfaForma2, vA, tWi.X, tWi.X);
  lIm := Branson(lMat, lMr, vIx, aInerciaIIs.Y);
  if lMat >= lMr then
    EDB_ImPP.Text := FormatFloat('###,###,##0.00000', lIm)
  else
    EDB_ImPP.Text := FormatFloat('###,###,##0.00000', vIx);
  EDB_MrPP.Text := FormatFloat('###,###,##0.0', lMr);
  if lMat <= lMr then
    LBL_Fissura1.Caption := 'N'
  else
    LBL_Fissura1.Caption := 'S';
  //Im para carregamento de laje
  aTenInf.S5 := aSigI[2];
  aTenSup.S5 := aSigS[2];
  aNp := GetSecNp(vABarra, lLpi, lLps, vL, aNLp, aTenInf, aTenSup);
  lMat := lMat + aMomentos[1].S5;
  lMr := MomFiss(aFck[1].X, aNp[5].X, aNp[5].Y, lEi, lEs, vAlfaForma,
vAlfaForma2, vA, tWi.X, tWi.X);
  lIm := Branson(lMat, lMr, vIx, aInerciaIIs.Y);
  if lMat >= lMr then
    EDB_ImLaje.Text := FormatFloat('###,###,##0.00000', lIm)
  else
    EDB_ImLaje.Text := FormatFloat('###,###,##0.00000', vIx);
  EDB_MrLaje.Text := FormatFloat('###,###,##0.0', lMr);
  if lMat <= lMr then
    LBL_Fissura2.Caption := 'N'
  else
    LBL_Fissura2.Caption := 'S';
  //Im para carregamento de capa
  aTenInf.S5 := aSigI[3];
  aTenSup.S5 := aSigS[3];
  aNp := GetSecNp(vABarra, lLpi, lLps, vL, aNLp, aTenInf, aTenSup);
  lMat := lMat + aMomentos[2].S5;;
  lMr := MomFiss(aFck[2].X, aNp[5].X, aNp[5].Y, lEi, lEs, vAlfaForma,
vAlfaForma2, vA, tWi.X, tWi.X);
  lIm := Branson(lMat, lMr, vIx, aInerciaIIs.Y);
  EDB_ImCapa.Text := FormatFloat('###,###,##0.00000', lIm);
  if lMat >= lMr then
    EDB_ImCapa.Text := FormatFloat('###,###,##0.00000', lIm)
  else
    EDB_ImCapa.Text := FormatFloat('###,###,##0.00000', vIx);
  EDB_MrCapa.Text := FormatFloat('###,###,##0.0', lMr);
  if lMat <= lMr then
    LBL_Fissura3.Caption := 'N'
  else
    LBL_Fissura3.Caption := 'S';
  //Im para carregamento de parede
  aTenInf.S5 := aSigI[4];
  aTenSup.S5 := aSigS[4];
  aNp := GetSecNp(vABarra, lLpi, lLps, vL, aNLp, aTenInf, aTenSup);
```

```
  lMat := lMat + aMomentos[3].S5;
  lMr := MomFiss(aFck[3].X, aNp[5].X, aNp[5].Y, lEi, lEs, vAlfaForma,
vAlfaForma2, vA, tWi.X, tWf.X);
  lIm := Branson(lMat, lMr, tIxY.X, aInerciaII.Y);
  if lMat >= lMr then
    EDB_ImParede.Text := FormatFloat('###,###,##0.00000', lIm)
  else
    EDB_ImParede.Text := FormatFloat('###,###,##0.00000', tIxY.X);
  EDB_MrParede.Text := FormatFloat('###,###,##0.0', lMr);
  if lMat <= lMr then
    LBL_Fissura4.Caption := 'N'
  else
    LBL_Fissura4.Caption := 'S';
  //Im para carregamento de revestimento
  aTenInf.S5 := aSigI[5];
  aTenSup.S5 := aSigS[5];
  aNp := GetSecNp(vABarra, lLpi, lLps, vL, aNLp, aTenInf, aTenSup);
  lMat := lMat + aMomentos[4].S5;
  lMr := MomFiss(aFck[4].X, aNp[5].X, aNp[5].Y, lEi, lEs, vAlfaForma,
vAlfaForma2, vA, tWi.X, tWf.X);
  lIm := Branson(lMat, lMr, tIxY.X, aInerciaII.Y);
  if lMat >= lMr then
    EDB_ImRevest.Text := FormatFloat('###,###,##0.00000', lIm)
  else
    EDB_ImRevest.Text := FormatFloat('###,###,##0.00000', tIxY.X);
  EDB_MrRevest.Text := FormatFloat('###,###,##0.0', lMr);
  if lMat <= lMr then
    LBL_Fissura5.Caption := 'N'
  else
    LBL_Fissura5.Caption := 'S';
  //Im para carregamento de acidental
  aTenInf.S5 := aSigI[6];
  aTenSup.S5 := aSigS[6];
  aNp := GetSecNp(vABarra, lLpi, lLps, vL, aNLp, aTenInf, aTenSup);
  lMat := lMat + aMomentos[5].S5;
  lMr := MomFiss(aFck[5].X, aNp[5].X, aNp[5].Y, lEi, lEs, vAlfaForma,
vAlfaForma2, vA, tWi.X, tWf.X);
  lIm := Branson(lMat, lMr, tIxY.X, aInerciaII.Y);
  if lMat >= lMr then
    EDB_ImAcid.Text := FormatFloat('###,###,##0.00000', lIm)
  else
    EDB_ImAcid.Text := FormatFloat('###,###,##0.00000', tIxY.X);
  EDB_MrAcid.Text := FormatFloat('###,###,##0.0', lMr);
  if lMat <= lMr then
    LBL_Fissura6.Caption := 'N'
  else
    LBL_Fissura6.Caption := 'S';
  //Im para perda de protensão
  aTenInf.S5 := aSigI[7];
  aTenSup.S5 := aSigS[7];
  aNp := GetSecNp(vABarra, lLpi, lLps, vL, aNLp, aTenInf, aTenSup);
  lMr := MomFiss(aFck[6].X, aNp[5].X, aNp[5].Y, lEi, lEs, vAlfaForma,
vAlfaForma2, vA, tWi.X, tWf.X);
  lIm := Branson(lMat, lMr, tIxY.X, aInerciaII.Y);
  if lMat >= lMr then
    EDB_ImPerda.Text := FormatFloat('###,###,##0.00000', lIm)
  else
    EDB_ImPerda.Text := FormatFloat('###,###,##0.00000', tIxY.X);
  EDB_MrPerda.Text := FormatFloat('###,###,##0.0', lMr);
  if lMat <= lMr then
    LBL_Fissura7.Caption := 'N'
```

```
  else
    LBL_Fissura7.Caption := 'S';
end;

procedure TFRM_PTracao.BTN_CalcFlechaClick(Sender: TObject);
var
  Ver_Arm: tDados;
  i, j: integer;
  vABarra, vNCordI5, vNCordS5, lI1, lI2, lI3, lI4: extended;
  lFckj, lFck, lP, lPsi2, lS, lMp, lLpi, lLps, lCoefE: extended;
  lSigI, lSigS, lPerdaI, lPerdaS, lEi, lEs, lEc, lFlechaT, lFleLim:
extended;
  aNpLp: array [0..5] of extended;
  aFlechaImed, aMr, aIm, aFiFlu, aFlechaDif: array[0..7] of extended;
begin
  vNCordI5 := 0; vNCordS5 := 0;
  //Busca o número de barras positivas e negativas totais e em cada seção
  For j := 0 to LBX_Linha.Count - 1 do begin
    Ver_Arm := tDados(LBX_Linha.Items.Objects[j]);
    if Ver_Arm.vTBarra = 1 then begin
      vABarra := Ver_Arm.vABarra;
      if Ver_Arm.vPBarra = 1 then
        vNCordI5 := vNCordI5 + Ver_Arm.vNBarra
      else if Ver_Arm.vPBarra = 2 then
        vNCordS5 := vNCordS5 + Ver_Arm.vNBarra;
    end;
  end;
  lI1 := StrToFloat(IsNil(EDB_S1.Text));
  lI2 := StrToFloat(IsNil(EDB_S2.Text));
  lI3 := StrToFloat(IsNil(EDB_S3.Text));
  lI4 := StrToFloat(IsNil(EDB_S4.Text));
  lFckj := StrToFloat(IsNil(EDB_fckj.Text));
  lFck := StrToFloat(IsNil(EDB_fck.Text));
  lPsi2 := StrToFloat(IsNil(EDB_psi2.Text));
  lSigI := StrToFloat(IsNil(EDB_TensaoIniI.Text)) / 10;
  lSigS := StrToFloat(IsNil(EDB_TensaoIniS.Text)) / 10;
  lEi := (GetExcentricity_Pos(LBX_Linha.Count, LBX_Linha, tCG)) / 100;
  lEs := (GetExcentricity_Neg(LBX_Linha.Count, LBX_Linha, tCG)) / 100;
  lLpi := StrToFloat(IsNil(EDB_LRegularizacaoI.Text));
  lLps := StrToFloat(IsNil(EDB_LRegularizacaoS.Text));
  aMr[1] := StrToFloat(IsNil(EDB_MrPP.Text));
  aMr[2] := StrToFloat(IsNil(EDB_MrLaje.Text));
  aMr[3] := StrToFloat(IsNil(EDB_MrCapa.Text));
  aMr[4] := StrToFloat(IsNil(EDB_MrParede.Text));
  aMr[5] := StrToFloat(IsNil(EDB_MrRevest.Text));
  aMr[6] := StrToFloat(IsNil(EDB_MrAcid.Text));
  aMr[7] := StrToFloat(IsNil(EDB_MrPerda.Text));
  aIm[1] := StrToFloat(IsNil(EDB_ImPP.Text));
  aIm[2] := StrToFloat(IsNil(EDB_ImLaje.Text));
  aIm[3] := StrToFloat(IsNil(EDB_ImCapa.Text));
  aIm[4] := StrToFloat(IsNil(EDB_ImParede.Text));
  aIm[5] := StrToFloat(IsNil(EDB_ImRevest.Text));
  aIm[6] := StrToFloat(IsNil(EDB_ImAcid.Text));
  aIm[7] := StrToFloat(IsNil(EDB_ImPerda.Text));
  aFiFlu[0] := StrToFloat(IsNil(EDB_FluProt.Text));
  aFiFlu[1] := StrToFloat(IsNil(EDB_FluPP.Text));
  aFiFlu[2] := StrToFloat(IsNil(EDB_FluLaje.Text));
  aFiFlu[3] := StrToFloat(IsNil(EDB_FluCapa.Text));
  aFiFlu[4] := StrToFloat(IsNil(EDB_FluParede.Text));
  aFiFlu[5] := StrToFloat(IsNil(EDB_FluRevest.Text));
  aFiFlu[6] := StrToFloat(IsNil(EDB_FluAcid.Text));
```

```
  aFiFlu[7] := StrToFloat(IsNil(EDB_FluPerda.Text));
  //ContraFlecha de protensão
  lPerdaI := StrToFloat(IsNil(EDB_DPprotI.Text));
  lPerdaS := StrToFloat(IsNil(EDB_DPprotS.Text));
  aNpLp[0] := (vNCordI5 - lI1) * vABarra * lSigI * (100 - lPerdaI) / 100;
  aNpLp[1] := (lI1 - lI2) * vABarra * lSigI * (100 - lPerdaI) / 100;
  aNpLp[2] := (lI2 - lI3) * vABarra * lSigI * (100 - lPerdaI) / 100;
  aNpLp[3] := (lI3 - lI4) * vABarra * lSigI * (100 - lPerdaI) / 100;
  aNpLp[4] := (lI4 -   0) * vABarra * lSigI * (100 - lPerdaI) / 100;
  aNpLp[5] := vNCordS5 * vABarra * lSigS * (100 - lPerdaS) / 100;
  if FRM_EdDados.CBX_AjusteFlecha.Checked = True then
    lCoefE := aEcs[0].X/aEcs[6].X
  else
    lCoefE := 1;
  aFlechaImed[0] := FlechaImed(1, lLpi, lLps, lEi, lEs, lP, vL,
aEcs[0].X*1000, aIm[1], aNpLp);
  aFlechaDif[0] := FlechaDif(aFlechaImed[0], aFiFlu[0], lCoefE);
  lFlechaT := aFlechaImed[0] + aFlechaDif[0];
  SGR_Flecha.Cells[3,1] := StringReplace(FormatFloat('###,###,##0.0',
lFlechaT),'.','',[]);
  //Flecha para carregamento de peso próprio
  lP := vPP;
  if FRM_EdDados.CBX_AjusteFlecha.Checked = True then
    lCoefE := aEcs[0].X/aEcs[6].X
  else
    lCoefE := 1;
  aFlechaImed[1] := FlechaImed(2, lLpi, lLps, lEi, lEs, lP, vL,
aEcs[0].X*1000, aIm[1], aNpLp);
  aFlechaDif[1] := FlechaDif(aFlechaImed[1], aFiFlu[1], lCoefE);
  lFlechaT := lFlechaT + aFlechaImed[1] + aFlechaDif[1];
  SGR_Flecha.Cells[3,2] := StringReplace(FormatFloat('###,###,##0.0',
lFlechaT),'.','',[]);
  //Flecha para carregamento de laje
  lP := vLaje;
  if FRM_EdDados.CBX_AjusteFlecha.Checked = True then
    lCoefE := aEcs[1].X/aEcs[6].X
  else
    lCoefE := 1;
  aFlechaImed[2] := FlechaImed(2, lLpi, lLps, lEi, lEs, lP, vL,
aEcs[1].X*1000, aIm[2], aNpLp);
  aFlechaDif[2] := FlechaDif(aFlechaImed[2], aFiFlu[2], lCoefE);
  lFlechaT := lFlechaT + aFlechaImed[2] + aFlechaDif[2];
  SGR_Flecha.Cells[3,3] := StringReplace(FormatFloat('###,###,##0.0',
lFlechaT),'.','',[]);
  //Flecha para carregamento de capa
  lP := vCapa;
  if FRM_EdDados.CBX_AjusteFlecha.Checked = True then
    lCoefE := aEcs[2].X/aEcs[6].X
  else
    lCoefE := 1;
  aFlechaImed[3] := FlechaImed(2, lLpi, lLps, lEi, lEs, lP, vL,
aEcs[2].X*1000, aIm[3], aNpLp);
  aFlechaDif[3] := FlechaDif(aFlechaImed[3], aFiFlu[3], lCoefE);
  lFlechaT := lFlechaT + aFlechaImed[3] + aFlechaDif[3];
  SGR_Flecha.Cells[3,4] := StringReplace(FormatFloat('###,###,##0.0',
lFlechaT),'.','',[]);
  //Flecha para carregamento de parede
  lP := vParede;
  if FRM_EdDados.CBX_AjusteFlecha.Checked = True then
    lCoefE := aEcs[3].X/aEcs[6].X
  else
```

```
    lCoefE := 1;
  aFlechaImed[4] := FlechaImed(2, lLpi, lLps, lEi, lEs, lP, vL,
aEcs[3].X*1000, aIm[4], aNpLp);
  aFlechaDif[4] := FlechaDif(aFlechaImed[4], aFiFlu[4], lCoefE);
  lFlechaT := lFlechaT + aFlechaImed[4] + aFlechaDif[4];
  SGR_Flecha.Cells[3,5] := StringReplace(FormatFloat('###,###,##0.0',
lFlechaT),'.','',[]);
  //Flecha para carregamento de revestimento
  lP := vRevestimento;
  if FRM_EdDados.CBX_AjusteFlecha.Checked = True then
    lCoefE := aEcs[4].X/aEcs[6].X
  else
    lCoefE := 1;
  aFlechaImed[5] := FlechaImed(2, lLpi, lLps, lEi, lEs, lP, vL,
aEcs[4].X*1000, aIm[5], aNpLp);
  aFlechaDif[5] := FlechaDif(aFlechaImed[5], aFiFlu[5], lCoefE);
  lFlechaT := lFlechaT + aFlechaImed[5] + aFlechaDif[5];
  SGR_Flecha.Cells[3,6] := StringReplace(FormatFloat('###,###,##0.0',
lFlechaT),'.','',[]);
  //Flecha para carregamento e acidental
  lP := vQmax * lPsi2;
  if FRM_EdDados.CBX_AjusteFlecha.Checked = True then
    lCoefE := aEcs[5].X/aEcs[6].X
  else
    lCoefE := 1;
  aFlechaImed[6] := FlechaImed(2, lLpi, lLps, lEi, lEs, lP, vL,
aEcs[5].X*1000, aIm[6], aNpLp);
  aFlechaDif[6] := FlechaDif(aFlechaImed[6], aFiFlu[6], lCoefE);
  lFlechaT := lFlechaT + aFlechaImed[6] + aFlechaDif[6];
  SGR_Flecha.Cells[3,7] := StringReplace(FormatFloat('###,###,##0.0',
lFlechaT),'.','',[]);
  //Flecha para carregamento de perda
  lPerdaI := (100 - (StrToFloat(IsNil(EDB_PTDifI1.Text)) -
StrToFloat(IsNil(EDB_DPprotI.Text)))) / 100;
  aNpLp[0] := (vNCordI5 - lI1) * vABarra * lSigI * lPerdaI;
  lPerdaI := (100 - (StrToFloat(IsNil(EDB_PTDifI2.Text)) -
StrToFloat(IsNil(EDB_DPprotI.Text)))) / 100;
  aNpLp[1] := (lI1 - lI2) * vABarra * lSigI * lPerdaI;
  lPerdaI := (100 - (StrToFloat(IsNil(EDB_PTDifI3.Text)) -
StrToFloat(IsNil(EDB_DPprotI.Text)))) / 100;
  aNpLp[2] := (lI2 - lI3) * vABarra * lSigI * lPerdaI;
  lPerdaI := (100 - (StrToFloat(IsNil(EDB_PTDifI4.Text)) -
StrToFloat(IsNil(EDB_DPprotI.Text)))) / 100;
  aNpLp[3] := (lI3 - lI4) * vABarra * lSigI * lPerdaI;
  lPerdaI := (100 - (StrToFloat(IsNil(EDB_PTDifI5.Text)) -
StrToFloat(IsNil(EDB_DPprotI.Text)))) / 100;
  aNpLp[4] := (lI4 -   0) * vABarra * lSigI * lPerdaI;
  lPerdaS := (100 - (StrToFloat(IsNil(EDB_PTDifS1.Text)) -
StrToFloat(IsNil(EDB_DPprotS.Text)))) / 100;
  aNpLp[5] := vNCordS5 * vABarra * lSigS * lPerdaS;
  aFlechaImed[7] := - FlechaImed(1, lLpi, lLps, lEi, lEs, lP, vL,
aEcs[6].X*1000, aIm[7], aNpLp);
  aFlechaDif[7] := FlechaDif(aFlechaImed[7], aFiFlu[7], 1);
  lFlechaT := lFlechaT + aFlechaImed[7] + aFlechaDif[7];
  SGR_Flecha.Cells[3,8] := FormatFloat('###,###,##0.0', lFlechaT);

  SGR_Flecha.Cells[1,1] := StringReplace(FormatFloat('###,###,##0.0',
aFlechaImed[0]),'.','',[]);
  SGR_Flecha.Cells[1,2] := FormatFloat('###,###,##0.0', aFlechaImed[1]);
  SGR_Flecha.Cells[1,3] := FormatFloat('###,###,##0.0', aFlechaImed[2]);
  SGR_Flecha.Cells[1,4] := FormatFloat('###,###,##0.0', aFlechaImed[3]);
```

```
  SGR_Flecha.Cells[1,5] := FormatFloat('###,###,##0.0', aFlechaImed[4]);
  SGR_Flecha.Cells[1,6] := FormatFloat('###,###,##0.0', aFlechaImed[5]);
  SGR_Flecha.Cells[1,7] := FormatFloat('###,###,##0.0', aFlechaImed[6]);
  SGR_Flecha.Cells[1,8] := FormatFloat('###,###,##0.0', aFlechaImed[7]);
  SGR_Flecha.Cells[2,1] := StringReplace(FormatFloat('###,###,##0.0',
aFlechaDif[0]),'.','',[]);
  SGR_Flecha.Cells[2,2] := FormatFloat('###,###,##0.0', aFlechaDif[1]);
  SGR_Flecha.Cells[2,3] := FormatFloat('###,###,##0.0', aFlechaDif[2]);
  SGR_Flecha.Cells[2,4] := FormatFloat('###,###,##0.0', aFlechaDif[3]);
  SGR_Flecha.Cells[2,5] := FormatFloat('###,###,##0.0', aFlechaDif[4]);
  SGR_Flecha.Cells[2,6] := FormatFloat('###,###,##0.0', aFlechaDif[5]);
  SGR_Flecha.Cells[2,7] := FormatFloat('###,###,##0.0', aFlechaDif[6]);
  SGR_Flecha.Cells[2,8] := FormatFloat('###,###,##0.0', aFlechaDif[7]);
  SGR_Flecha.Cells[3,9] := FormatFloat('###,###,##0.0', vL * 1000 /
vLimFle);
end;

//Cálculo da armadura transversal
procedure TFRM_PTracao.BTN_TransversalClick(Sender: TObject);
var
  i, j: integer;
  Ver_Arm: tDados;
  lLpi, lLps, lNeo, lEi, lEs, lSigI, lSigS, lP, lMNeo, lsMin, lFhd:
extended;
  lFck, lBw, lD, lTAco, lAAco, lTeta, lAlfa: extended;
  lNCordI5, lNCordS5: integer;
  aCortante: tEsforcos;
  aSComposta: tPoints;
  aNLp: array [0..5] of integer;
  aNp: array[0..7] of tPoints;
  aVd, aTenInf, aTenSup: tEsforcos;
  tTransversal: tEstribo;
begin
  lNCordI5 := 0; lNCordS5 := 0; lBw := 10000;
  //Busca o número de barras positivas e negativas totais e em cada seção
  For j := 0 to LBX_Linha.Count - 1 do begin
    Ver_Arm := tDados(LBX_Linha.Items.Objects[j]);
    if Ver_Arm.vTBarra = 1 then begin
      vABarra := Ver_Arm.vABarra;
      if Ver_Arm.vPBarra = 1 then
        lNCordI5 := lNCordI5 + Ver_Arm.vNBarra
      else if Ver_Arm.vPBarra = 2 then
        lNCordS5 := lNCordS5 + Ver_Arm.vNBarra;
    end;
  end;
  lLpi := StrToFloat(IsNil(EDB_LRegularizacaoI.Text));
  lLps := StrToFloat(IsNil(EDB_LRegularizacaoS.Text));
  lNeo := StrToFloat(IsNil(EDB_LNeoprene.Text));
  if lNeo = 0 then begin
    ShowMessage('Insira um valor diferente de zero no campo "L neo"!');
    Exit;
  end;
  lSigI := StrToFloat(IsNil(EDB_TensaoIniI.Text));
  lSigS := StrToFloat(IsNil(EDB_TensaoIniS.Text));
  lFck := StrToFloat(IsNil(EDB_fck.Text));
  lTeta := StrToFloat(IsNil(EDB_Teta.Text));
  lAlfa := StrToFloat(IsNil(EDB_Alfa.Text));
  lD := StrToFloat(SGR_Dimens.Cells[3,2]);
  lP := vPP*vMPP + vLaje*vMLaje + vCapa*vMCapa + vParede*vMParede +
vRevestimento*vMRevest + vQmax*vMAcid;
  lMNeo := Momento_Lp(lP, vL, lNeo);
```

```
  aCortante := Cortantes_Vao(lP, vL, lLpi, lLps, lNeo);
  lFhd := vMd / (tAp.KZ * lD);
  lEi := (GetExcentricity_Pos(LBX_Linha.Count, LBX_Linha, tCG)) / 100;
  lEs := (GetExcentricity_Neg(LBX_Linha.Count, LBX_Linha, tCG)) / 100;
  for i := 0 to Length(aDimensoes) - 1 do
    if aDimensoes[i].X <> 0 then
      lBw := min(lBw, aDimensoes[i].X);
  case CBB_TAcoEstribo.ItemIndex of
     0: lTAco := 25;
     1: lTAco := 50;
     2: lTAco := 60;
  end;
  case CBB_DAcoEstribo.ItemIndex of
       0: lAAco := 0.125;
       1: lAAco := 0.240;
       2: lAAco := 0.315;
       3: lAAco := 0.500;
       4: lAAco := 0.800;
       5: lAAco := 1.250;
    end;
  aTenInf.S1 := lSigI * (1 - StrToFloat(IsNil(EDB_PTDifI1.Text)) / 100);
  aTenInf.S2 := lSigI * (1 - StrToFloat(IsNil(EDB_PTDifI2.Text)) / 100);
  aTenInf.S3 := lSigI * (1 - StrToFloat(IsNil(EDB_PTDifI3.Text)) / 100);
  aTenInf.S4 := lSigI * (1 - StrToFloat(IsNil(EDB_PTDifI4.Text)) / 100);
  aTenInf.S5 := lSigI * (1 - StrToFloat(IsNil(EDB_PTDifI5.Text)) / 100);
  aTenInf.SLpi := lSigI * (1 - StrToFloat(IsNil(EDB_PTDifI6.Text)) / 100);
  aTenInf.SLps := lSigI * (1 - StrToFloat(IsNil(EDB_PTDifI7.Text)) / 100);
  aTenSup.S1 := lSigS * (1 - StrToFloat(IsNil(EDB_PTDifS1.Text)) / 100);
  aTenSup.S2 := lSigS * (1 - StrToFloat(IsNil(EDB_PTDifS2.Text)) / 100);
  aTenSup.S3 := lSigS * (1 - StrToFloat(IsNil(EDB_PTDifS3.Text)) / 100);
  aTenSup.S4 := lSigS * (1 - StrToFloat(IsNil(EDB_PTDifS4.Text)) / 100);
  aTenSup.S5 := lSigS * (1 - StrToFloat(IsNil(EDB_PTDifS5.Text)) / 100);
  aTenSup.SLpi := lSigS * (1 - StrToFloat(IsNil(EDB_PTDifS6.Text)) / 100);
  aTenSup.SLps := lSigS * (1 - StrToFloat(IsNil(EDB_PTDifS7.Text)) / 100);
  aNLp[0] := lNCordS5;
  aNLp[1] := lNCordI5 - StrToInt(IsNil(EDB_S1.Text));
  aNLp[2] := StrToInt(IsNil(EDB_S1.Text)) - StrToInt(IsNil(EDB_S2.Text));
  aNLp[3] := StrToInt(IsNil(EDB_S2.Text)) - StrToInt(IsNil(EDB_S3.Text));
  aNLp[4] := StrToInt(IsNil(EDB_S3.Text)) - StrToInt(IsNil(EDB_S4.Text));
  aNLp[5] := 0;
  aNp := GetSecNp(vABarra, lLpi, lLps, vL, aNLp, aTenInf, aTenSup);
  if RBT_ModeloI.Checked = True then
    tTransversal := TransversalI(aNp[1].X, aNp[1].Y, vGamaF, vGamaS, lEi,
lEs, lFck, vGamaC, lBw, lD, vA, tWf.X, aCortante.SNeo, lMNeo, lAAco, lTAco,
lAlfa, vL, lP)
  else if RBT_ModeloII.Checked = True then
    tTransversal := TransversalII(aNp[1].X, aNp[1].Y, vGamaF, vGamaS, lEi,
lEs, lFck, vGamaC, lBw, lD, vA, tWf.X, aCortante.SNeo, lMNeo, lAAco, lTAco,
lAlfa, lTeta, vL, lP);
  lsMin := vL - tTransversal.xEstribo;
  aSComposta := InterfaceTrans(lFhd, vL, lFck, vGamaC, lTAco, vGamaS,
lAAco, aDimensoes, aBeta);
  SGR_Estribo.Cells[1,0] := tTransversal.Biela;
  SGR_Estribo.Cells[1,1] := tTransversal.TipoDim;
  SGR_Estribo.Cells[1,2] := IntToStr(Round(tTransversal.Vc));
  SGR_Estribo.Cells[1,3] := IntToStr(Round(tTransversal.Vsw));
  SGR_Estribo.Cells[3,0] := CBB_DAcoEstribo.Text + ' c/ ' +
FormatFloat('###,###,##0.0', tTransversal.s);
  SGR_Estribo.Cells[3,1] := CBB_DAcoEstribo.Text + ' c/ ' +
FormatFloat('###,###,##0.0', tTransversal.sMax);
```

```
  SGR_Estribo.Cells[3,2] := 'Entre ' + FormatFloat('###,###,##0.0',
tTransversal.xEstribo) + ' e ' + FormatFloat('###,###,##0.0', lsMin) + '
m';
  if CBX_Composta.Checked = True then
    SGR_Estribo.Cells[3,3] := 'Esp. entre ' + FormatFloat('###,###,##0',
aSComposta.Y)
                            + ' e ' + FormatFloat('###,###,##0',
aSComposta.X) + ' cm'
  else
    SGR_Estribo.Cells[3,3] := 'Não há estribo exposto!';
end;

//Salvando os dados em um arquivo
procedure TFRM_PTracao.MMI_SalvarClick(Sender: TObject);
var
  i: integer;
  lText: Text;
  Ver_Arm: tDados;
begin
  if Save.Execute then begin
    AssignFile(lText, Save.FileName);
    Rewrite(lText);
    //Características básicas - Save
    writeln(lText, '*****Características básicas*****');
    writeln(lText, IntToStr(vCont));
    if CBX_Composta.Checked = True then writeln(lText, 'Ch')
      else writeln(lText, 'N ch');
    if CBX_PP.Checked = True then writeln(lText, 'Ch')
      else writeln(lText, 'N ch');
    writeln(lText, SGR_Dimensao.Cells[1,1]); writeln(lText,
SGR_Dimensao.Cells[2,1]);
    writeln(lText, SGR_Dimensao.Cells[1,2]); writeln(lText,
SGR_Dimensao.Cells[2,2]);
    writeln(lText, SGR_Dimensao.Cells[1,3]); writeln(lText,
SGR_Dimensao.Cells[2,3]);
    writeln(lText, SGR_Dimensao.Cells[1,4]); writeln(lText,
SGR_Dimensao.Cells[2,4]);
    writeln(lText, SGR_Dimensao.Cells[1,5]); writeln(lText,
SGR_Dimensao.Cells[2,5]);
    writeln(lText, EDB_PP.Text);
    writeln(lText, EDB_Laje.Text);
    writeln(lText, EDB_Capa.Text);
    writeln(lText, EDB_Parede.Text);
    writeln(lText, EDB_Revestimento.Text);
    writeln(lText, EDB_qmax.Text);
    writeln(lText, EDB_qmin.Text);
    writeln(lText, EDB_vao.Text);
    writeln(lText, EDB_fckj.Text);
    writeln(lText, EDB_fck.Text);
    if CBX_EcEp.Checked = True then writeln(lText, 'Ch')
      else writeln(lText, 'N ch');
    writeln(lText, EDB_rEcEp.Text);
    writeln(lText, CBB_CAA.Text);
    writeln(lText, EDB_psi1.Text);
    writeln(lText, EDB_psi2.Text);
    writeln(lText, EDB_psi3.Text);
    //Edição de dados - Save
    writeln(lText, '*****Editar - Dados*****');
    writeln(lText, FRM_EdDados.EDB_EDTVazio.Text); writeln(lText,
FRM_EdDados.EDB_EDCVazio.Text);
```

```
    writeln(lText, FRM_EdDados.EDB_EDTFiss.Text); writeln(lText,
FRM_EdDados.EDB_EDTDesc.Text);
    writeln(lText, FRM_EdDados.EDB_EDCServ.Text);
    writeln(lText, FRM_EdDados.EDB_EDAlfaC.Text); writeln(lText,
FRM_EdDados.EDB_EDAlfaCIL.Text);
    writeln(lText, FRM_EdDados.EDB_EDLambida.Text); writeln(lText,
FRM_EdDados.EDB_EDLambidaIL.Text);
    writeln(lText, FRM_EdDados.EDB_EDGamaC.Text); writeln(lText,
FRM_EdDados.EDB_EDGamaCIL.Text);
    writeln(lText, FRM_EdDados.EDB_EDGamaS.Text); writeln(lText,
FRM_EdDados.EDB_EDxd.Text);
    writeln(lText, FRM_EdDados.EDB_EDFpyd.Text); writeln(lText,
FRM_EdDados.EDB_EDEpsyd.Text);
    writeln(lText, FRM_EdDados.EDB_EDFptd.Text); writeln(lText,
FRM_EdDados.EDB_EDEpsu.Text);
    writeln(lText, FRM_EdDados.EDB_EDEp.Text); writeln(lText,
FRM_EdDados.EDB_EDEs.Text);
    writeln(lText, FRM_EdDados.EDB_RoMin.Text); writeln(lText,
FRM_EdDados.EDB_RoMax.Text);
    writeln(lText, FRM_EdDados.EDB_BetaSMin.Text); writeln(lText,
FRM_EdDados.EDB_BetaSMax.Text);
    writeln(lText, FRM_EdDados.EDB_BetaCMin.Text); writeln(lText,
FRM_EdDados.EDB_BetaCMax.Text);
    writeln(lText, FRM_EdDados.EDB_EtaS.Text);
    writeln(lText, FRM_EdDados.EDB_R1.Text); writeln(lText,
FRM_EdDados.EDB_PsiMil1.Text);
    writeln(lText, FRM_EdDados.EDB_R2.Text); writeln(lText,
FRM_EdDados.EDB_PsiMil2.Text);
    writeln(lText, FRM_EdDados.EDB_R3.Text); writeln(lText,
FRM_EdDados.EDB_PsiMil3.Text);
    writeln(lText, FRM_EdDados.EDB_R4.Text); writeln(lText,
FRM_EdDados.EDB_PsiMil4.Text);
    writeln(lText, FRM_EdDados.EDB_Alfa1.Text); writeln(lText,
FRM_EdDados.EDB_Alfa2.Text);
    writeln(lText, FRM_EdDados.EDB_Alfa3.Text); writeln(lText,
FRM_EdDados.EDB_S1.Text);
    writeln(lText, FRM_EdDados.EDB_S2.Text); writeln(lText,
FRM_EdDados.EDB_S3.Text);
    writeln(lText, FRM_EdDados.EDB_Fc1.Text); writeln(lText,
FRM_EdDados.EDB_Fc2.Text);
    writeln(lText, FRM_EdDados.EDB_Fc3.Text); writeln(lText,
FRM_EdDados.EDB_TInf.Text);
    writeln(lText, FRM_EdDados.EDB_Fid.Text);
    writeln(lText, FRM_EdDados.EDB_EDAlfaE.Text); writeln(lText,
FRM_EdDados.EDB_EDSecaoForma.Text);
    writeln(lText, FRM_EdDados.EDB_EDSecaoForma2.Text); writeln(lText,
FRM_EdDados.EDB_EDGamaF.Text);
    if FRM_EdDados.CBX_AjusteFlecha.Checked = True then writeln(lText,
'Ch')
      else writeln(lText, 'N ch');
    writeln(lText, FRM_EdDados.EDB_Wk.Text); writeln(lText,
FRM_EdDados.EDB_EDLimFle.Text);
    //Datas e coeficientes - Save
    writeln(lText, '*****Datas e coeficientes*****');
    writeln(lText, FRM_DatasCoef.EDB_MPP.Text); writeln(lText,
FRM_DatasCoef.EDB_MLaje.Text);
    writeln(lText, FRM_DatasCoef.EDB_MCapa.Text); writeln(lText,
FRM_DatasCoef.EDB_MParede.Text);
    writeln(lText, FRM_DatasCoef.EDB_MRevest.Text); writeln(lText,
FRM_DatasCoef.EDB_MAcid.Text);
```

```
    writeln(lText, FRM_DatasCoef.EDB_DProt.Text); writeln(lText,
FRM_DatasCoef.EDB_DPP.Text);
    writeln(lText, FRM_DatasCoef.EDB_DLaje.Text); writeln(lText,
FRM_DatasCoef.EDB_DCapa.Text);
    writeln(lText, FRM_DatasCoef.EDB_DParede.Text); writeln(lText,
FRM_DatasCoef.EDB_DRevest.Text);
    writeln(lText, FRM_DatasCoef.EDB_DAcid.Text); writeln(lText,
FRM_DatasCoef.EDB_DPerda.Text);
    writeln(lText, FRM_DatasCoef.CBX_PMCimento.Text);
    writeln(lText, FRM_DatasCoef.EDB_DfcPP.Text); writeln(lText,
FRM_DatasCoef.EDB_DfcLaje.Text);
    writeln(lText, FRM_DatasCoef.EDB_DfcCapa.Text); writeln(lText,
FRM_DatasCoef.EDB_DfcParede.Text);
    writeln(lText, FRM_DatasCoef.EDB_DfcRevest.Text); writeln(lText,
FRM_DatasCoef.EDB_DfcAcid.Text);
    writeln(lText, FRM_DatasCoef.EDB_DfcPerda.Text);
    writeln(lText, FRM_DatasCoef.EDB_DEciPP.Text); writeln(lText,
FRM_DatasCoef.EDB_DEciLaje.Text);
    writeln(lText, FRM_DatasCoef.EDB_DEciCapa.Text); writeln(lText,
FRM_DatasCoef.EDB_DEciParede.Text);
    writeln(lText, FRM_DatasCoef.EDB_DEciRevest.Text); writeln(lText,
FRM_DatasCoef.EDB_DEciAcid.Text);
    writeln(lText, FRM_DatasCoef.EDB_DEciPerda.Text);
    writeln(lText, FRM_DatasCoef.EDB_DEcsPP.Text); writeln(lText,
FRM_DatasCoef.EDB_DEcsLaje.Text);
    writeln(lText, FRM_DatasCoef.EDB_DEcsCapa.Text); writeln(lText,
FRM_DatasCoef.EDB_DEcsParede.Text);
    writeln(lText, FRM_DatasCoef.EDB_DEcsRevest.Text); writeln(lText,
FRM_DatasCoef.EDB_DEcsAcid.Text);
    writeln(lText, FRM_DatasCoef.EDB_DEcsPerda.Text);
    writeln(lText, FRM_DatasCoef.CBX_ILCimento.Text);
    writeln(lText, FRM_DatasCoef.EDB_DfcParede1.Text); writeln(lText,
FRM_DatasCoef.EDB_DfcRevest1.Text);
    writeln(lText, FRM_DatasCoef.EDB_DfcAcid1.Text); writeln(lText,
FRM_DatasCoef.EDB_DfcPerda1.Text);
    writeln(lText, FRM_DatasCoef.EDB_DEciParede1.Text); writeln(lText,
FRM_DatasCoef.EDB_DEciRevest1.Text);
    writeln(lText, FRM_DatasCoef.EDB_DEciAcid1.Text); writeln(lText,
FRM_DatasCoef.EDB_DEciPerda1.Text);
    writeln(lText, FRM_DatasCoef.EDB_DEcsParede1.Text); writeln(lText,
FRM_DatasCoef.EDB_DEcsRevest1.Text);
    writeln(lText, FRM_DatasCoef.EDB_DEcsAcid1.Text); writeln(lText,
FRM_DatasCoef.EDB_DEcsPerda1.Text);
    //Dados da seção composta - Save
    writeln(lText, '*****Dados da seção composta*****');
    if FRM_SComposta.RBT_VLateral.Checked = True then writeln(lText,
'Lateral')
      else writeln(lText, 'Central');
    writeln(lText, FRM_SComposta.EDB_hLaje.Text); writeln(lText,
FRM_SComposta.EDB_lLaje.Text);
    writeln(lText, FRM_SComposta.EDB_hCapa.Text); writeln(lText,
FRM_SComposta.EDB_fckCapa.Text);
    writeln(lText, FRM_SComposta.EDB_Bf.Text);
    //Dimensionamento - Save
    writeln(lText, '*****Dimensionamento*****');
    writeln(lText, IntToStr(LBX_Linha.Count));
    for i := 0 to LBX_Linha.Count - 1 do begin
      Ver_Arm := tDados(LBX_Linha.Items.Objects[i]);
      writeln(lText, LBX_Linha.Items.Strings[i]);
      writeln(lText, IntToStr(Ver_Arm.vNBarra)); writeln(lText,
IntToStr(Ver_Arm.vPBarra));
```

```
      writeln(lText, IntToStr(Ver_Arm.vTBarra)); writeln(lText,
IntToStr(Ver_Arm.vTAco_index));
      writeln(lText, Ver_Arm.vTAco); writeln(lText, Ver_Arm.vDBarra);
      writeln(lText, IntToStr(Ver_Arm.vDBarra_index)); writeln(lText,
FloatToStr(Ver_Arm.vXa));
      writeln(lText, FloatToStr(Ver_Arm.vYa)); writeln(lText,
FloatToStr(Ver_Arm.vXb));
      writeln(lText, FloatToStr(Ver_Arm.vYb));
      writeln(lText, FloatToStr(Ver_Arm.vABarra));
    end;
    writeln(lText, EDB_TensaoIniI.Text); writeln(lText,
EDB_TensaoIniS.Text);
    writeln(lText, EDB_PerdaIniI.Text); writeln(lText, EDB_PerdaIniS.Text);
    writeln(lText, EDB_PerdaFinI.Text); writeln(lText, EDB_PerdaFinS.Text);
    if CBX_S1.Checked = True then writeln(lText, 'Ch')
      else writeln(lText, 'N ch');
    if CBX_S2.Checked = True then writeln(lText, 'Ch')
      else writeln(lText, 'N ch');
    if CBX_S3.Checked = True then writeln(lText, 'Ch')
      else writeln(lText, 'N ch');
    if CBX_S4.Checked = True then writeln(lText, 'Ch')
      else writeln(lText, 'N ch');
    writeln(lText, EDB_S1.Text); writeln(lText, EDB_S2.Text);
    writeln(lText, EDB_S3.Text); writeln(lText, EDB_S4.Text);
    writeln(lText, EDB_Eta1.Text); writeln(lText, EDB_Eta2.Text);
    writeln(lText, EDB_LRegularizacaoI.Text); writeln(lText,
EDB_LRegularizacaoS.Text);
    writeln(lText, CBX_TLiberacao1.Text);
    //Perdas de protensão - Save
    writeln(lText, '*****Perdas de protensão*****');
    writeln(lText, EDB_CPista.Text); writeln(lText, EDB_DeltaL.Text);
    writeln(lText, EDB_AlfaP1.Text); writeln(lText, EDB_AlfaP2.Text);
    if CBX_AlfaP1.Checked = True then writeln(lText, 'Ch')
      else writeln(lText, 'N ch');
    if CBX_AlfaP2.Checked = True then writeln(lText, 'Ch')
      else writeln(lText, 'N ch');
    writeln(lText, EDB_PMSlump.Text); writeln(lText, EDB_PMTMedia.Text);
    writeln(lText, EDB_PMUmidade.Text); writeln(lText, EDB_PMUAr.Text);
    writeln(lText, EDB_PMAc.Text);
    if CBX_PMUar.Checked = True then writeln(lText, 'Ch')
      else writeln(lText, 'N ch');
    if CBX_PMAc.Checked = True then writeln(lText, 'Ch')
      else writeln(lText, 'N ch');
    writeln(lText, EDB_ILSlump.Text); writeln(lText, EDB_ILTMedia.Text);
    writeln(lText, EDB_ILUmidade.Text); writeln(lText, EDB_ILUAr.Text);
    writeln(lText, EDB_ILAc.Text);
    if CBX_ILUar1.Checked = True then writeln(lText, 'Ch')
      else writeln(lText, 'N ch');
    if CBX_ILAc1.Checked = True then writeln(lText, 'Ch')
      else writeln(lText, 'N ch');
    writeln(lText, EDB_PTImedS1.Text); writeln(lText, EDB_PTImedS2.Text);
    writeln(lText, EDB_PTImedS3.Text); writeln(lText, EDB_PTImedS4.Text);
    writeln(lText, EDB_PTImedS5.Text); writeln(lText, EDB_PTImedS6.Text);
    writeln(lText, EDB_PTImedS7.Text);
    writeln(lText, EDB_PTDifS1.Text); writeln(lText, EDB_PTDifS2.Text);
    writeln(lText, EDB_PTDifS3.Text); writeln(lText, EDB_PTDifS4.Text);
    writeln(lText, EDB_PTDifS5.Text); writeln(lText, EDB_PTDifS6.Text);
    writeln(lText, EDB_PTDifS7.Text);
    writeln(lText, EDB_PTImedI1.Text); writeln(lText, EDB_PTImedI2.Text);
    writeln(lText, EDB_PTImedI3.Text); writeln(lText, EDB_PTImedI4.Text);
    writeln(lText, EDB_PTImedI5.Text); writeln(lText, EDB_PTImedI6.Text);
```

```
    writeln(lText, EDB_PTImedI7.Text);
    writeln(lText, EDB_PTDifI1.Text); writeln(lText, EDB_PTDifI2.Text);
    writeln(lText, EDB_PTDifI3.Text); writeln(lText, EDB_PTDifI4.Text);
    writeln(lText, EDB_PTDifI5.Text); writeln(lText, EDB_PTDifI6.Text);
    writeln(lText, EDB_PTDifI7.Text);
    //Flecha e transversal - Save
    writeln(lText, '*****Flecha e transversal*****');
    writeln(lText, EDB_DPprotI.Text); writeln(lText, EDB_DPprotS.Text);
    writeln(lText, EDB_DPppI.Text); writeln(lText, EDB_DPppS.Text);
    writeln(lText, EDB_DPlajeI.Text); writeln(lText, EDB_DPlajeS.Text);
    writeln(lText, EDB_DPcapaI.Text); writeln(lText, EDB_DPcapaS.Text);
    writeln(lText, EDB_DPparedeI.Text); writeln(lText, EDB_DPparedeS.Text);
    writeln(lText, EDB_DPrevestI.Text); writeln(lText, EDB_DPrevestS.Text);
    writeln(lText, EDB_DPacidI.Text); writeln(lText, EDB_DPacidS.Text);
    writeln(lText, EDB_ImPP.Text); writeln(lText, EDB_MrPP.Text);
    writeln(lText, EDB_ImLaje.Text); writeln(lText, EDB_MrLaje.Text);
    writeln(lText, EDB_ImCapa.Text); writeln(lText, EDB_MrCapa.Text);
    writeln(lText, EDB_ImParede.Text); writeln(lText, EDB_MrParede.Text);
    writeln(lText, EDB_ImRevest.Text); writeln(lText, EDB_MrRevest.Text);
    writeln(lText, EDB_ImAcid.Text); writeln(lText, EDB_MrAcid.Text);
    writeln(lText, EDB_ImPerda.Text); writeln(lText, EDB_MrPerda.Text);
    writeln(lText, EDB_FluProt.Text); writeln(lText, EDB_FluPP.Text);
    writeln(lText, EDB_FluLaje.Text); writeln(lText, EDB_FluCapa.Text);
    writeln(lText, EDB_FluParede.Text); writeln(lText, EDB_FluRevest.Text);
    writeln(lText, EDB_FluAcid.Text); writeln(lText, EDB_FluPerda.Text);
    writeln(lText, CBB_TAcoEstribo.Text); writeln(lText,
CBB_DAcoEstribo.Text);
    writeln(lText, EDB_Teta.Text); writeln(lText, EDB_Alfa.Text);
    writeln(lText, EDB_LNeoprene.Text);
    if RBT_ModeloI.Checked = True then writeln(lText, 'Modelo I')
      else writeln(lText, 'Modelo II');
    //Fissuração - Save
    writeln(lText, '*****Fissuração*****');
    writeln(lText, IntToStr(Length(aFissura)));
    for i := 0 to Length(aFissura) - 1 do begin
      writeln(lText, IntToStr(aFissura[i].Aco)); writeln(lText,
FloatToStr(aFissura[i].X));
      writeln(lText, FloatToStr(aFissura[i].Y)); writeln(lText,
FloatToStr(aFissura[i].Fi));
      writeln(lText, FloatToStr(aFissura[i].Ap)); writeln(lText,
FloatToStr(aFissura[i].SigAco));
      writeln(lText, FloatToStr(aFissura[i].Wk1)); writeln(lText,
FloatToStr(aFissura[i].Wk2));
      writeln(lText, FloatToStr(aFissura[i].a)); writeln(lText,
FloatToStr(aFissura[i].b));
      writeln(lText, FloatToStr(aFissura[i].c)); writeln(lText,
FloatToStr(aFissura[i].d));
      writeln(lText, FloatToStr(aFissura[i].Ro));
    end;
    CloseFile(lText);
  end;
end;

//Abrindo os dados de um arquivo
procedure TFRM_PTracao.MMI_AbrirClick(Sender: TObject);
var
  i: integer;
  lStr, lLinha: String;
  lText: Text;
  Ver_Arm: tDados;
begin
```

```
  if Open.Execute then begin
    AssignFile(lText, Open.FileName);
    Reset(lText);
    //Características básicas - Open
    Readln(lText, lStr);
    Readln(lText, lStr);
    case StrToInt(lStr) of
      1: BBT_Retangular.Click;
      2: BBT_I.Click;
      3: BBT_T.Click;
      4: BBT_Aba.Click;
    end;
    Readln(lText, lStr);
    if lStr = 'Ch' then CBX_Composta.Checked := True
      else CBX_Composta.Checked := False;
    Readln(lText, lStr);
    if lStr = 'Ch' then CBX_PP.Checked := True
      else CBX_PP.Checked := False;
    Readln(lText, lStr); SGR_Dimensao.Cells[1,1] := lStr; Readln(lText,
lStr); SGR_Dimensao.Cells[2,1] := lStr;
    Readln(lText, lStr); SGR_Dimensao.Cells[1,2] := lStr; Readln(lText,
lStr); SGR_Dimensao.Cells[2,2] := lStr;
    Readln(lText, lStr); SGR_Dimensao.Cells[1,3] := lStr; Readln(lText,
lStr); SGR_Dimensao.Cells[2,3] := lStr;
    Readln(lText, lStr); SGR_Dimensao.Cells[1,4] := lStr; Readln(lText,
lStr); SGR_Dimensao.Cells[2,4] := lStr;
    Readln(lText, lStr); SGR_Dimensao.Cells[1,5] := lStr; Readln(lText,
lStr); SGR_Dimensao.Cells[2,5] := lStr;
    Readln(lText, lStr); EDB_PP.Text := lStr;
    Readln(lText, lStr); EDB_Laje.Text := lStr;
    Readln(lText, lStr); EDB_Capa.Text := lStr;
    Readln(lText, lStr); EDB_Parede.Text := lStr;
    Readln(lText, lStr); EDB_Revestimento.Text := lStr;
    Readln(lText, lStr); EDB_qmax.Text := lStr;
    Readln(lText, lStr); EDB_qmin.Text := lStr;
    Readln(lText, lStr); EDB_vao.Text := lStr;
    Readln(lText, lStr); EDB_fckj.Text := lStr;
    Readln(lText, lStr); EDB_fck.Text := lStr;
    Readln(lText, lStr);
    if lStr = 'Ch' then CBX_EcEp.Checked := True
      else CBX_EcEp.Checked := False;
    Readln(lText, lStr); EDB_rEcEp.Text := lStr;
    Readln(lText, lStr); CBB_CAA.Text := lStr;
    Readln(lText, lStr); EDB_psi1.Text := lStr;
    Readln(lText, lStr); EDB_psi2.Text := lStr;
    Readln(lText, lStr); EDB_psi3.Text := lStr;
    //Edição de dados - Open
    Readln(lText, lStr);
    Readln(lText, lStr); FRM_EdDados.EDB_EDTVazio.Text := lStr;
Readln(lText, lStr); FRM_EdDados.EDB_EDCVazio.Text := lStr;
    Readln(lText, lStr); FRM_EdDados.EDB_EDTFiss.Text := lStr;
Readln(lText, lStr); FRM_EdDados.EDB_EDTDesc.Text := lStr;
    Readln(lText, lStr); FRM_EdDados.EDB_EDCServ.Text := lStr;
    Readln(lText, lStr); FRM_EdDados.EDB_EDAlfaC.Text := lStr;
Readln(lText, lStr); FRM_EdDados.EDB_EDAlfaCIL.Text := lStr;
    Readln(lText, lStr); FRM_EdDados.EDB_EDLambida.Text := lStr;
Readln(lText, lStr); FRM_EdDados.EDB_EDLambidaIL.Text := lStr;
    Readln(lText, lStr); FRM_EdDados.EDB_EDGamaC.Text := lStr;
Readln(lText, lStr); FRM_EdDados.EDB_EDGamaCIL.Text := lStr;
    Readln(lText, lStr); FRM_EdDados.EDB_EDGamaS.Text := lStr;
Readln(lText, lStr); FRM_EdDados.EDB_EDxd.Text := lStr;
```

```
    Readln(lText, lStr); FRM_EdDados.EDB_EDFpyd.Text := lStr; Readln(lText,
lStr); FRM_EdDados.EDB_EDEpsyd.Text := lStr;
    Readln(lText, lStr); FRM_EdDados.EDB_EDFptd.Text := lStr; Readln(lText,
lStr); FRM_EdDados.EDB_EDEpsu.Text := lStr;
    Readln(lText, lStr); FRM_EdDados.EDB_EDEp.Text := lStr; Readln(lText,
lStr); FRM_EdDados.EDB_EDEs.Text := lStr;
    Readln(lText, lStr); FRM_EdDados.EDB_RoMin.Text := lStr; Readln(lText,
lStr); FRM_EdDados.EDB_RoMax.Text := lStr;
    Readln(lText, lStr); FRM_EdDados.EDB_BetaSMin.Text := lStr;
Readln(lText, lStr); FRM_EdDados.EDB_BetaSMax.Text := lStr;
    Readln(lText, lStr); FRM_EdDados.EDB_BetaCMin.Text := lStr;
Readln(lText, lStr); FRM_EdDados.EDB_BetaCMax.Text := lStr;
    Readln(lText, lStr); FRM_EdDados.EDB_EtaS.Text := lStr;
    Readln(lText, lStr); FRM_EdDados.EDB_R1.Text := lStr; Readln(lText,
lStr); FRM_EdDados.EDB_PsiMil1.Text := lStr;
    Readln(lText, lStr); FRM_EdDados.EDB_R2.Text := lStr; Readln(lText,
lStr); FRM_EdDados.EDB_PsiMil2.Text := lStr;
    Readln(lText, lStr); FRM_EdDados.EDB_R3.Text := lStr; Readln(lText,
lStr); FRM_EdDados.EDB_PsiMil3.Text := lStr;
    Readln(lText, lStr); FRM_EdDados.EDB_R4.Text := lStr; Readln(lText,
lStr); FRM_EdDados.EDB_PsiMil4.Text := lStr;
    Readln(lText, lStr); FRM_EdDados.EDB_Alfa1.Text := lStr; Readln(lText,
lStr); FRM_EdDados.EDB_Alfa2.Text := lStr;
    Readln(lText, lStr); FRM_EdDados.EDB_Alfa3.Text := lStr; Readln(lText,
lStr); FRM_EdDados.EDB_S1.Text := lStr;
    Readln(lText, lStr); FRM_EdDados.EDB_S2.Text := lStr; Readln(lText,
lStr); FRM_EdDados.EDB_S3.Text := lStr;
    Readln(lText, lStr); FRM_EdDados.EDB_Fc1.Text := lStr; Readln(lText,
lStr); FRM_EdDados.EDB_Fc2.Text := lStr;
    Readln(lText, lStr); FRM_EdDados.EDB_Fc3.Text := lStr; Readln(lText,
lStr); FRM_EdDados.EDB_TInf.Text := lStr;
    Readln(lText, lStr); FRM_EdDados.EDB_Fid.Text := lStr;
    Readln(lText, lStr); FRM_EdDados.EDB_EDAlfaE.Text := lStr;
Readln(lText, lStr); FRM_EdDados.EDB_EDSecaoForma.Text := lStr;
    Readln(lText, lStr); FRM_EdDados.EDB_EDSecaoForma2.Text := lStr;
Readln(lText, lStr); FRM_EdDados.EDB_EDGamaF.Text := lStr;
    Readln(lText, lStr);
    if lStr = 'Ch' then FRM_EdDados.CBX_AjusteFlecha.Checked := True
      else FRM_EdDados.CBX_AjusteFlecha.Checked := False;
    Readln(lText, lStr); FRM_EdDados.EDB_Wk.Text := lStr; Readln(lText,
lStr); FRM_EdDados.EDB_EDLimFle.Text := lStr;
    FRM_EdDados.BTN_EDOk.Click;
    //Datas e coeficientes - Open
    Readln(lText, lStr);
    Readln(lText, lStr); FRM_DatasCoef.EDB_MPP.Text := lStr; Readln(lText,
lStr); FRM_DatasCoef.EDB_MLaje.Text := lStr;
    Readln(lText, lStr); FRM_DatasCoef.EDB_MCapa.Text := lStr;
Readln(lText, lStr); FRM_DatasCoef.EDB_MParede.Text := lStr;
    Readln(lText, lStr); FRM_DatasCoef.EDB_MRevest.Text := lStr;
Readln(lText, lStr); FRM_DatasCoef.EDB_MAcid.Text := lStr;
    Readln(lText, lStr); FRM_DatasCoef.EDB_DProt.Text := lStr;
Readln(lText, lStr); FRM_DatasCoef.EDB_DPP.Text := lStr;
    Readln(lText, lStr); FRM_DatasCoef.EDB_DLaje.Text := lStr;
Readln(lText, lStr); FRM_DatasCoef.EDB_DCapa.Text := lStr;
    Readln(lText, lStr); FRM_DatasCoef.EDB_DParede.Text := lStr;
Readln(lText, lStr); FRM_DatasCoef.EDB_DRevest.Text := lStr;
    Readln(lText, lStr); FRM_DatasCoef.EDB_DAcid.Text := lStr;
Readln(lText, lStr); FRM_DatasCoef.EDB_DPerda.Text := lStr;
    Readln(lText, lStr); FRM_DatasCoef.CBX_PMCimento.Text := lStr;
    Readln(lText, lStr); FRM_DatasCoef.EDB_DfcPP.Text := lStr;
```

```
    Readln(lText, lStr); FRM_DatasCoef.EDB_DfcLaje.Text := lStr;
Readln(lText, lStr); FRM_DatasCoef.EDB_DfcCapa.Text := lStr;
    Readln(lText, lStr); FRM_DatasCoef.EDB_DfcParede.Text := lStr;
Readln(lText, lStr); FRM_DatasCoef.EDB_DfcRevest.Text := lStr;
    Readln(lText, lStr); FRM_DatasCoef.EDB_DfcAcid.Text := lStr;
    Readln(lText, lStr); FRM_DatasCoef.EDB_DfcPerda.Text := lStr;
    Readln(lText, lStr); FRM_DatasCoef.EDB_DEciPP.Text := lStr;
Readln(lText, lStr); FRM_DatasCoef.EDB_DEciLaje.Text := lStr;
    Readln(lText, lStr); FRM_DatasCoef.EDB_DEciCapa.Text := lStr;
Readln(lText, lStr); FRM_DatasCoef.EDB_DEciParede.Text := lStr;
    Readln(lText, lStr); FRM_DatasCoef.EDB_DEciRevest.Text := lStr;
Readln(lText, lStr); FRM_DatasCoef.EDB_DEciAcid.Text := lStr;
    Readln(lText, lStr); FRM_DatasCoef.EDB_DEciPerda.Text := lStr;
    Readln(lText, lStr); FRM_DatasCoef.EDB_DEcsPP.Text := lStr;
Readln(lText, lStr); FRM_DatasCoef.EDB_DEcsLaje.Text := lStr;
    Readln(lText, lStr); FRM_DatasCoef.EDB_DEcsCapa.Text := lStr;
Readln(lText, lStr); FRM_DatasCoef.EDB_DEcsParede.Text := lStr;
    Readln(lText, lStr); FRM_DatasCoef.EDB_DEcsRevest.Text := lStr;
Readln(lText, lStr); FRM_DatasCoef.EDB_DEcsAcid.Text := lStr;
    Readln(lText, lStr); FRM_DatasCoef.EDB_DEcsPerda.Text := lStr;
    Readln(lText, lStr); FRM_DatasCoef.CBX_ILCimento.Text := lStr;
    Readln(lText, lStr); FRM_DatasCoef.EDB_DfcParede1.Text := lStr;
Readln(lText, lStr); FRM_DatasCoef.EDB_DfcRevest1.Text := lStr;
    Readln(lText, lStr); FRM_DatasCoef.EDB_DfcAcid1.Text := lStr;
    Readln(lText, lStr); FRM_DatasCoef.EDB_DfcPerda1.Text := lStr;
    Readln(lText, lStr); FRM_DatasCoef.EDB_DEciParede1.Text := lStr;
Readln(lText, lStr); FRM_DatasCoef.EDB_DEciRevest1.Text := lStr;
    Readln(lText, lStr); FRM_DatasCoef.EDB_DEciAcid1.Text := lStr;
Readln(lText, lStr); FRM_DatasCoef.EDB_DEciPerda1.Text := lStr;
    Readln(lText, lStr); FRM_DatasCoef.EDB_DEcsParede1.Text := lStr;
Readln(lText, lStr); FRM_DatasCoef.EDB_DEcsRevest1.Text := lStr;
    Readln(lText, lStr); FRM_DatasCoef.EDB_DEcsAcid1.Text := lStr;
Readln(lText, lStr); FRM_DatasCoef.EDB_DEcsPerda1.Text := lStr;
    FRM_DatasCoef.BTN_DTOk.Click;
    //Dados da seção composta - Open
    Readln(lText, lStr);
    Readln(lText, lStr);
    if lStr = 'Lateral' then FRM_SComposta.RBT_VLateral.Checked := True
      else FRM_SComposta.RBT_VCentral.Checked := True;
    Readln(lText, lStr); FRM_SComposta.EDB_hLaje.Text := lStr;
Readln(lText, lStr); FRM_SComposta.EDB_lLaje.Text := lStr;
    Readln(lText, lStr); FRM_SComposta.EDB_hCapa.Text := lStr;
Readln(lText, lStr); FRM_SComposta.EDB_fckCapa.Text := lStr;
    Readln(lText, lStr); FRM_SComposta.EDB_Bf.Text := lStr;
    FRM_SComposta.BTN_Ok.Click;
    //Dimensionamento - Open
    LBX_Linha.Clear;
    Readln(lText, lStr);
    Readln(lText, lStr);
    for i := 1 to StrToInt(lStr) do begin
      Ver_Arm := tDados.Create;
      Readln(lText, lStr); lLinha := lStr;
      Readln(lText, lStr); Ver_Arm.vNBarra := StrToInt(lStr); Readln(lText,
lStr); Ver_Arm.vPBarra := StrToInt(lStr);
      Readln(lText, lStr); Ver_Arm.vTBarra := StrToInt(lStr); Readln(lText,
lStr); Ver_Arm.vTAco_index := StrToInt(lStr);
      Readln(lText, lStr); Ver_Arm.vTAco := lStr; Readln(lText, lStr);
Ver_Arm.vDBarra := lStr;
      Readln(lText, lStr); Ver_Arm.vDBarra_index := StrToInt(lStr);
Readln(lText, lStr); Ver_Arm.vXa := StrToFloat(lStr);
```

```
      Readln(lText, lStr); Ver_Arm.vYa := StrToFloat(lStr); Readln(lText,
lStr); Ver_Arm.vXb := StrToFloat(lStr);
      Readln(lText, lStr); Ver_Arm.vYb := StrToFloat(lStr); Readln(lText,
lStr); Ver_Arm.vABarra := StrToFloat(lStr);
      LBX_Linha.AddItem(lLinha, Ver_Arm);
    end;
    BTN_PreDim.Click;
    Readln(lText, lStr); EDB_TensaoIniI.Text := lStr; Readln(lText, lStr);
EDB_TensaoIniS.Text := lStr;
    Readln(lText, lStr); EDB_PerdaIniI.Text := lStr; Readln(lText, lStr);
EDB_PerdaIniS.Text := lStr;
    Readln(lText, lStr); EDB_PerdaFinI.Text := lStr; Readln(lText, lStr);
EDB_PerdaFinS.Text := lStr;
    Readln(lText, lStr);
    if lStr = 'Ch' then CBX_S1.Checked := True
      else CBX_S1.Checked := False;
    Readln(lText, lStr);
    if lStr = 'Ch' then CBX_S2.Checked := True
      else CBX_S2.Checked := False;
    Readln(lText, lStr);
    if lStr = 'Ch' then CBX_S3.Checked := True
      else CBX_S3.Checked := False;
    Readln(lText, lStr);
    if lStr = 'Ch' then CBX_S4.Checked := True
      else CBX_S4.Checked := False;
    Readln(lText, lStr); EDB_S1.Text := lStr; Readln(lText, lStr);
EDB_S2.Text := lStr;
    Readln(lText, lStr); EDB_S3.Text := lStr; Readln(lText, lStr);
EDB_S4.Text := lStr;
    Readln(lText, lStr); EDB_Eta1.Text := lStr; Readln(lText, lStr);
EDB_Eta2.Text := lStr;
    Readln(lText, lStr); EDB_LRegularizacaoI.Text := lStr; Readln(lText,
lStr); EDB_LRegularizacaoS.Text := lStr;
    Readln(lText, lStr); CBX_TLiberacao1.Text := lStr;
    BTN_Dimensionar.Click;
    //Perdas de protensão - Open
    Readln(lText, lStr);
    Readln(lText, lStr); EDB_CPista.Text := lStr; Readln(lText, lStr);
EDB_DeltaL.Text := lStr;
    Readln(lText, lStr); EDB_AlfaP1.Text := lStr; Readln(lText, lStr);
EDB_AlfaP2.Text := lStr;
    Readln(lText, lStr);
    if lStr = 'Ch' then CBX_AlfaP1.Checked := True
      else CBX_AlfaP1.Checked := False;
    Readln(lText, lStr);
    if lStr = 'Ch' then CBX_AlfaP2.Checked := True
      else CBX_AlfaP2.Checked := False;
    Readln(lText, lStr); EDB_PMSlump.Text := lStr; Readln(lText, lStr);
EDB_PMTMedia.Text := lStr;
    Readln(lText, lStr); EDB_PMUmidade.Text := lStr; Readln(lText, lStr);
EDB_PMUAr.Text := lStr;
    Readln(lText, lStr); EDB_PMAc.Text := lStr;
    Readln(lText, lStr);
    if lStr = 'Ch' then CBX_PMUar.Checked := True
      else CBX_PMUar.Checked := False;
    Readln(lText, lStr);
    if lStr = 'Ch' then CBX_PMAc.Checked := True
      else CBX_PMAc.Checked := False;
    Readln(lText, lStr); EDB_ILSlump.Text := lStr; Readln(lText, lStr);
EDB_ILTMedia.Text := lStr;
```

```
    Readln(lText, lStr); EDB_ILUmidade.Text := lStr; Readln(lText, lStr);
EDB_ILUAr.Text := lStr;
    Readln(lText, lStr); EDB_ILAc.Text := lStr;
    Readln(lText, lStr);
    if lStr = 'Ch' then CBX_ILUar1.Checked := True
      else CBX_ILUar1.Checked := False;
    Readln(lText, lStr);
    if lStr = 'Ch' then CBX_ILAc1.Checked := True
      else CBX_ILAc1.Checked := False;
    BTN_CalcPerdas.Click;
    Readln(lText, lStr); EDB_PTImedS1.Text := lStr; Readln(lText, lStr);
EDB_PTImedS2.Text := lStr;
    Readln(lText, lStr); EDB_PTImedS3.Text := lStr; Readln(lText, lStr);
EDB_PTImedS4.Text := lStr;
    Readln(lText, lStr); EDB_PTImedS5.Text := lStr; Readln(lText, lStr);
EDB_PTImedS6.Text := lStr;
    Readln(lText, lStr); EDB_PTImedS7.Text := lStr;
    Readln(lText, lStr); EDB_PTDifS1.Text := lStr; Readln(lText, lStr);
EDB_PTDifS2.Text := lStr;
    Readln(lText, lStr); EDB_PTDifS3.Text := lStr; Readln(lText, lStr);
EDB_PTDifS4.Text := lStr;
    Readln(lText, lStr); EDB_PTDifS5.Text := lStr; Readln(lText, lStr);
EDB_PTDifS6.Text := lStr;
    Readln(lText, lStr); EDB_PTDifS7.Text := lStr;
    Readln(lText, lStr); EDB_PTImedI1.Text := lStr; Readln(lText, lStr);
EDB_PTImedI2.Text := lStr;
    Readln(lText, lStr); EDB_PTImedI3.Text := lStr; Readln(lText, lStr);
EDB_PTImedI4.Text := lStr;
    Readln(lText, lStr); EDB_PTImedI5.Text := lStr; Readln(lText, lStr);
EDB_PTImedI6.Text := lStr;
    Readln(lText, lStr); EDB_PTImedI7.Text := lStr;
    Readln(lText, lStr); EDB_PTDifI1.Text := lStr; Readln(lText, lStr);
EDB_PTDifI2.Text := lStr;
    Readln(lText, lStr); EDB_PTDifI3.Text := lStr; Readln(lText, lStr);
EDB_PTDifI4.Text := lStr;
    Readln(lText, lStr); EDB_PTDifI5.Text := lStr; Readln(lText, lStr);
EDB_PTDifI6.Text := lStr;
    Readln(lText, lStr); EDB_PTDifI7.Text := lStr;
    BTN_Verificar1.Click;
    //Flecha e transversal - Open
    Readln(lText, lStr);
    Readln(lText, lStr); EDB_DPprotI.Text := lStr; Readln(lText, lStr);
EDB_DPprotS.Text := lStr;
    Readln(lText, lStr); EDB_DPppI.Text := lStr; Readln(lText, lStr);
EDB_DPppS.Text := lStr;
    Readln(lText, lStr); EDB_DPlajeI.Text := lStr; Readln(lText, lStr);
EDB_DPlajeS.Text := lStr;
    Readln(lText, lStr); EDB_DPcapaI.Text := lStr; Readln(lText, lStr);
EDB_DPcapaS.Text := lStr;
    Readln(lText, lStr); EDB_DPparedeI.Text := lStr; Readln(lText, lStr);
EDB_DPparedeS.Text := lStr;
    Readln(lText, lStr); EDB_DPrevestI.Text := lStr; Readln(lText, lStr);
EDB_DPrevestS.Text := lStr;
    Readln(lText, lStr); EDB_DPacidI.Text := lStr; Readln(lText, lStr);
EDB_DPacidS.Text := lStr;
    Readln(lText, lStr); EDB_ImPP.Text := lStr; Readln(lText, lStr);
EDB_MrPP.Text := lStr;
    Readln(lText, lStr); EDB_ImLaje.Text := lStr; Readln(lText, lStr);
EDB_MrLaje.Text := lStr;
    Readln(lText, lStr); EDB_ImCapa.Text := lStr; Readln(lText, lStr);
EDB_MrCapa.Text := lStr;
```

```
    Readln(lText, lStr); EDB_ImParede.Text := lStr; Readln(lText, lStr);
EDB_MrParede.Text := lStr;
    Readln(lText, lStr); EDB_ImRevest.Text := lStr; Readln(lText, lStr);
EDB_MrRevest.Text := lStr;
    Readln(lText, lStr); EDB_ImAcid.Text := lStr; Readln(lText, lStr);
EDB_MrAcid.Text := lStr;
    Readln(lText, lStr); EDB_ImPerda.Text := lStr; Readln(lText, lStr);
EDB_MrPerda.Text := lStr;
    Readln(lText, lStr); EDB_FluProt.Text := lStr; Readln(lText, lStr);
EDB_FluPP.Text := lStr;
    Readln(lText, lStr); EDB_FluLaje.Text := lStr; Readln(lText, lStr);
EDB_FluCapa.Text := lStr;
    Readln(lText, lStr); EDB_FluParede.Text := lStr; Readln(lText, lStr);
EDB_FluRevest.Text := lStr;
    Readln(lText, lStr); EDB_FluAcid.Text := lStr; Readln(lText, lStr);
EDB_FluPerda.Text := lStr;
    Readln(lText, lStr); CBB_TAcoEstribo.Text := lStr; Readln(lText, lStr);
CBB_DAcoEstribo.Text := lStr;
    Readln(lText, lStr); EDB_Teta.Text := lStr; Readln(lText, lStr);
EDB_Alfa.Text := lStr;
    Readln(lText, lStr); EDB_LNeoprene.Text := lStr;
    Readln(lText, lStr);
    if lStr = 'Modelo I' then RBT_ModeloI.Checked := True
      else RBT_ModeloII.Checked := True;
    BTN_CalcFlecha.Click;
    BTN_Transversal.Click;
    //Fissuração - Open
    Readln(lText, lStr);
    Readln(lText, lStr);
    SetLength(aFissura, StrToInt(lStr));
    for i := 0 to StrToInt(lStr) - 1 do begin
      Readln(lText, lStr); aFissura[i].Aco := StrToInt(lStr); Readln(lText,
lStr); aFissura[i].X := StrToFloat(lStr);
      Readln(lText, lStr); aFissura[i].Y := StrToFloat(lStr); Readln(lText,
lStr); aFissura[i].Fi := StrToFloat(lStr);
      Readln(lText, lStr); aFissura[i].Ap := StrToFloat(lStr);
Readln(lText, lStr); aFissura[i].SigAco := StrToFloat(lStr);
      Readln(lText, lStr); aFissura[i].Wk1 := StrToFloat(lStr);
Readln(lText, lStr); aFissura[i].Wk2 := StrToFloat(lStr);
      Readln(lText, lStr); aFissura[i].a := StrToFloat(lStr); Readln(lText,
lStr); aFissura[i].b := StrToFloat(lStr);
      Readln(lText, lStr); aFissura[i].c := StrToFloat(lStr); Readln(lText,
lStr); aFissura[i].d := StrToFloat(lStr);
      Readln(lText, lStr); aFissura[i].Ro := StrToFloat(lStr);
    end;
    BTN_FissErros.Click;
    CloseFile(lText);
    TBS_GeCaFi.Show;
  end;
end;

end.
```

9.2 LISTAGEM DO MENU “EDITAR”

```
var
  FRM_EdDados: TFRM_EdDados;
  vTVazio, vCVazio, vTFiss, vTDesc, vCServ: extended;
  vAlfaE,  vAlfaForma, vAlfaForma2, vGamaS, vGamaF, vXD: extended;
```

```
  vAlfaC, vLambida, vGamaC, vAlfaCIL, vLambidaIL, vGamaCIL: extended;
  vFpyd, vEpsyd, vFptd, vEpsu, vEp, vEdoce, vFid, vEtaS, vWk, vLimFle:
extended;
  vTInf: integer;
  aR, aPsiMil: array[0..3] of extended;
  aAlfa, aCoefS, aFc: array[0..2] of extended;
  aBeta: array[0..2] of tPoints;

implementation

{$R *.lfm}

{ TFRM_EdDados }

procedure TFRM_EdDados.FormActivate(Sender: TObject);
begin
  EDB_EDTVazio.Text := FloatToStr(vTVazio);
  EDB_EDCVazio.Text := FloatToStr(vCVazio);
  EDB_EDTFiss.Text := FloatToStr(vTFiss);
  EDB_EDTDesc.Text := FloatToStr(vTDesc);
  EDB_EDCServ.Text := FloatToStr(vCServ);
  EDB_EDAlfaC.Text := FloatToStr(vAlfaC);
  EDB_EDAlfaCIL.Text := FloatToStr(vAlfaCIL);
  EDB_EDLambida.Text := FloatToStr(vLambida);
  EDB_EDLambidaIL.Text := FloatToStr(vLambidaIL);
  EDB_EDGamaC.Text := FloatToStr(vGamaC);
  EDB_EDGamaCIL.Text := FloatToStr(vGamaCIL);
  EDB_EDGamaS.Text := FloatToStr(vGamaS);
  EDB_EDGamaF.Text := FloatToStr(vGamaF);
  EDB_EDxd.Text := FloatToStr(vXD);
  EDB_EDFpyd.Text := FloatToStr(vFpyd);
  EDB_EDEpsyd.Text := FloatToStr(vEpsyd);
  EDB_EDFptd.Text := FloatToStr(vFptd);
  EDB_EDEpsu.Text := FloatToStr(vEpsu);
  EDB_EDEp.Text := FloatToStr(vEp);
  EDB_EDEs.Text := FloatToStr(vEdoce);
  EDB_R1.Text := FloatToStr(aR[0]);
  EDB_R2.Text := FloatToStr(aR[1]);
  EDB_R3.Text := FloatToStr(aR[2]);
  EDB_R4.Text := FloatToStr(aR[3]);
  EDB_PsiMil1.Text := FloatToStr(aPsiMil[0]);
  EDB_PsiMil2.Text := FloatToStr(aPsiMil[1]);
  EDB_PsiMil3.Text := FloatToStr(aPsiMil[2]);
  EDB_PsiMil4.Text := FloatToStr(aPsiMil[3]);
  EDB_Alfa1.Text := FloatToStr(aAlfa[0]);
  EDB_Alfa2.Text := FloatToStr(aAlfa[1]);
  EDB_Alfa3.Text := FloatToStr(aAlfa[2]);
  EDB_S1.Text := FloatToStr(aCoefS[0]);
  EDB_S2.Text := FloatToStr(aCoefS[1]);
  EDB_S3.Text := FloatToStr(aCoefS[2]);
  EDB_Fc1.Text := FloatToStr(aFc[0]);
  EDB_Fc2.Text := FloatToStr(aFc[1]);
  EDB_Fc3.Text := FloatToStr(aFc[2]);
  EDB_TInf.Text := IntToStr(vTInf);
  EDB_Fid.Text := FloatToStr(vFid);
  EDB_EDAlfaE.Text := FloatToStr(vAlfaE);
  EDB_EDSecaoForma.Text := FloatToStr(vAlfaForma);
  EDB_EDSecaoForma2.Text := FloatToStr(vAlfaForma2);
  EDB_RoMin.Text := FloatToStr(aBeta[0].X);
  EDB_RoMax.Text := FloatToStr(aBeta[0].Y);
  EDB_BetaSMin.Text := FloatToStr(aBeta[1].X);
```

```
  EDB_BetaSMax.Text := FloatToStr(aBeta[1].Y);
  EDB_BetaCMin.Text := FloatToStr(aBeta[2].X);
  EDB_BetaCMax.Text := FloatToStr(aBeta[2].Y);
  EDB_EtaS.Text := FloatToStr(vEtaS);
  EDB_Wk.Text := FloatToStr(vWk);
  EDB_EDLimFle.Text := FloatToStr(vLimFle);
end;

procedure TFRM_EdDados.EDB_EDTVazioKeyUp(Sender: TObject; var Key: Word;
  Shift: TShiftState);
begin
 if not isNeg(TEdit(Sender).Text) then
    begin
      TEdit(Sender).Text := '';
      ShowMessage('Algarismo inválido!');
    end;
end;

//Botão Ok
procedure TFRM_EdDados.BTN_EDOkClick(Sender: TObject);
begin
  vTVazio := (StrToFloat(IsNil(EDB_EDTVazio.Text)));
  vCVazio := (StrToFloat(IsNil(EDB_EDCVazio.Text)));
  vTFiss := (StrToFloat(IsNil(EDB_EDTFiss.Text)));
  vTDesc := (StrToFloat(IsNil(EDB_EDTDesc.Text)));
  vCServ := (StrToFloat(IsNil(EDB_EDCServ.Text)));
  vAlfaE := (StrToFloat(IsNil(EDB_EDAlfaE.Text)));
  vAlfaC := (StrToFloat(IsNil(EDB_EDAlfaC.Text)));
  vAlfaCIL := (StrToFloat(IsNil(EDB_EDAlfaCIL.Text)));
  vAlfaForma := (StrToFloat(IsNil(EDB_EDSecaoForma.Text)));
  vAlfaForma2 := (StrToFloat(IsNil(EDB_EDSecaoForma2.Text)));
  vLambida := (StrToFloat(IsNil(EDB_EDLambida.Text)));
  vLambidaIL := (StrToFloat(IsNil(EDB_EDLambidaIL.Text)));
  vGamaC := (StrToFloat(IsNil(EDB_EDGamaC.Text)));
  vGamaCIL := (StrToFloat(IsNil(EDB_EDGamaCIL.Text)));
  vGamaS := (StrToFloat(IsNil(EDB_EDGamaS.Text)));
  vGamaF := (StrToFloat(IsNil(EDB_EDGamaF.Text)));
  vXD := (StrToFloat(IsNil(EDB_EDxd.Text)));
  vFpyd := (StrToFloat(IsNil(EDB_EDFpyd.Text)));
  vEpsyd := (StrToFloat(IsNil(EDB_EDEpsyd.Text)));
  vFptd := (StrToFloat(IsNil(EDB_EDFptd.Text)));
  vEpsu := (StrToFloat(IsNil(EDB_EDEpsu.Text)));
  vEp := (StrToFloat(IsNil(EDB_EDEp.Text)));
  vEdoce := (StrToFloat(IsNil(EDB_EDEs.Text)));
  aR[0] := (StrToFloat(IsNil(EDB_R1.Text)));
  aR[1] := (StrToFloat(IsNil(EDB_R2.Text)));
  aR[2] := (StrToFloat(IsNil(EDB_R3.Text)));
  aR[3] := (StrToFloat(IsNil(EDB_R4.Text)));
  aPsiMil[0] := (StrToFloat(IsNil(EDB_PsiMil1.Text)));
  aPsiMil[1] := (StrToFloat(IsNil(EDB_PsiMil2.Text)));
  aPsiMil[2] := (StrToFloat(IsNil(EDB_PsiMil3.Text)));
  aPsiMil[3] := (StrToFloat(IsNil(EDB_PsiMil4.Text)));
  aAlfa[0] := (StrToFloat(IsNil(EDB_Alfa1.Text)));
  aAlfa[1] := (StrToFloat(IsNil(EDB_Alfa2.Text)));
  aAlfa[2] := (StrToFloat(IsNil(EDB_Alfa3.Text)));
  aCoefS[0] := (StrToFloat(IsNil(EDB_S1.Text)));
  aCoefS[1] := (StrToFloat(IsNil(EDB_S2.Text)));
  aCoefS[2] := (StrToFloat(IsNil(EDB_S3.Text)));
  aFc[0] := (StrToFloat(IsNil(EDB_Fc1.Text)));
  aFc[1] := (StrToFloat(IsNil(EDB_Fc2.Text)));
  aFc[2] := (StrToFloat(IsNil(EDB_Fc3.Text)));
```

```
  vTInf := (StrToInt(IsNil(EDB_TInf.Text)));
  vFid := (StrToFloat(IsNil(EDB_Fid.Text)));
  aBeta[0].X := (StrToFloat(IsNil(EDB_RoMin.Text)));
  aBeta[0].Y := (StrToFloat(IsNil(EDB_RoMax.Text)));
  aBeta[1].X := (StrToFloat(IsNil(EDB_BetaSMin.Text)));
  aBeta[1].Y := (StrToFloat(IsNil(EDB_BetaSMax.Text)));
  aBeta[2].X := (StrToFloat(IsNil(EDB_BetaCMin.Text)));
  aBeta[2].Y := (StrToFloat(IsNil(EDB_BetaCMax.Text)));
  vEtaS := (StrToFloat(IsNil(EDB_EtaS.Text)));
  vWk := (StrToFloat(IsNil(EDB_Wk.Text)));
  vLimFle := (StrToFloat(IsNil(EDB_EDLimFle.Text)));
  FRM_EdDados.Close;
end;

//Botão cancelar
procedure TFRM_EdDados.BTN_EDCanClick(Sender: TObject);
begin
  FRM_EdDados.Close;
end;

//Botão default
procedure TFRM_EdDados.BTN_EDDefClick(Sender: TObject);
begin
  EDB_EDTVazio.Text := '1.2';
  EDB_EDCVazio.Text := '0.7';
  EDB_EDTFiss.Text := '1';
  EDB_EDTDesc.Text := '0';
  EDB_EDCServ.Text := '0.7';
  EDB_EDAlfaE.Text := '1';
  EDB_EDAlfaC.Text := '0.85';
  EDB_EDAlfaCIL.Text := '0.85';
  EDB_EDSecaoForma.Text := '1.5';
  EDB_EDSecaoForma2.Text := '1.2';
  EDB_EDLambida.Text := '0.8';
  EDB_EDLambidaIL.Text := '0.8';
  EDB_EDGamaC.Text := '1.4';
  EDB_EDGamaCIL.Text := '1.4';
  EDB_EDGamaS.Text := '1.15';
  EDB_EDGamaF.Text := '0.9';
  EDB_EDxd.Text := '0.45';
  EDB_EDFpyd.Text := '1460';
  EDB_EDEpsyd.Text := '0.73';
  EDB_EDFptd.Text := '1626';
  EDB_EDEpsu.Text := '3.5';
  EDB_EDEp.Text := '200000';
  EDB_EDEs.Text := '210000';
  EDB_R1.Text := '0.5';
  EDB_R2.Text := '0.6';
  EDB_R3.Text := '0.7';
  EDB_R4.Text := '0.8';
  EDB_PsiMil1.Text := '0';
  EDB_PsiMil2.Text := '1.3';
  EDB_PsiMil3.Text := '2.5';
  EDB_PsiMil4.Text := '3.5';
  EDB_Alfa1.Text := '1';
  EDB_Alfa2.Text := '2';
  EDB_Alfa3.Text := '3';
  EDB_S1.Text := '0.38';
  EDB_S2.Text := '0.25';
  EDB_S3.Text := '0.2';
  EDB_Fc1.Text := '1.433';
```

```
  EDB_Fc2.Text := '1.267';
  EDB_Fc3.Text := '1.208';
  EDB_TInf.Text := '10000';
  EDB_Fid.Text := '0.4';
  EDB_RoMin.Text := '0.2';
  EDB_RoMax.Text := '0.5';
  EDB_BetaSMin.Text := '0.0';
  EDB_BetaSMax.Text := '0.9';
  EDB_BetaCMin.Text := '0.3';
  EDB_BetaCMax.Text := '0.6';
  EDB_EtaS.Text := '2.25';
  EDB_Wk.Text := '0.2';
  EDB_EDLimFle.Text := '250';
  CBX_AjusteFlecha.Checked := True;
end;

end.
```

9.3 LISTAGEM DA TELA DE ENTRADA DE DADOS DA SEÇÃO COMPOSTA

```
var
  FRM_SComposta: TFRM_SComposta;
  vCont1: integer;
  hLaje, lLaje, hCapa, fckCapa, vBfCapa: extended;

implementation

{$R *.lfm}

{ TFRM_SComposta }

procedure TFRM_SComposta.FormCreate(Sender: TObject);
begin
  RBT_VLateral.Checked := True;
end;

procedure TFRM_SComposta.RBT_VLateralClick(Sender: TObject);
begin
  if RBT_VLateral.Checked = True then
    IMG_SComposta.Picture.Bitmap.LoadFromResourceName(HINSTANCE,'VLATERAL')
end;

procedure TFRM_SComposta.RBT_VCentralClick(Sender: TObject);
begin
  if RBT_VCentral.Checked = True then
    IMG_SComposta.Picture.Bitmap.LoadFromResourceName(HINSTANCE,'VCENTRAL')
end;

procedure TFRM_SComposta.EDB_hLajeKeyUp(Sender: TObject; var Key: Word;
  Shift: TShiftState);
begin
 if not isNeg(TEdit(Sender).Text) then
    begin
      TEdit(Sender).Text := '';
      ShowMessage('Algarismo inválido!');
    end;
end;

procedure TFRM_SComposta.BTN_Ok2Click(Sender: TObject);
```

```
begin

end;

procedure TFRM_SComposta.BTN_OkClick(Sender: TObject);
begin
  if RBT_VLateral.Checked = True then begin
    lLaje := StrToFloat(IsNil(EDB_lLaje.Text)) / 100;
    vCont1 := 1;
  end
  else if RBT_VCentral.Checked = True then begin
    lLaje := StrToFloat(IsNil(EDB_lLaje.Text)) * 2 / 100;
    vCont1 := 2;
  end;
  hCapa := StrToFloat(IsNil(EDB_hCapa.Text)) / 100;
  hLaje := StrToFloat(IsNil(EDB_hLaje.Text)) / 100;
  fckCapa := StrToFloat(IsNil(EDB_fckCapa.Text));
  vBfCapa := (StrToFloat(IsNil(EDB_Bf.Text)));
  FRM_SComposta.Close;
end;

procedure TFRM_SComposta.BTN_CancelClick(Sender: TObject);
begin
  FRM_SComposta.Close;
end;

end.
```

9.4 LISTAGEM DA TELA DO BOTÃO “DATAS E COEFICIENTES”

```
var
  FRM_DatasCoef: TFRM_DatasCoef;
  vMPP, vMLaje, vMCapa, vMParede, vMRevest, vMAcid: extended;
  vDProt, vDPP, vDLaje, vDCapa, vDParede, vDRevest, vDAcid, vDPerda:
integer;
  aFck, aEci, aEcs: array[0..6] of tPoints;

implementation

{$R *.lfm}

{ TFRM_DatasCoef }

procedure TFRM_DatasCoef.FormActivate(Sender: TObject);
begin
  EDB_MPP.Text := FloatToStr(vMPP);
  EDB_MLaje.Text := FloatToStr(vMLaje);
  EDB_MCapa.Text := FloatToStr(vMCapa);
  EDB_MParede.Text := FloatToStr(vMParede);
  EDB_MRevest.Text := FloatToStr(vMRevest);
  EDB_MAcid.Text := FloatToStr(vMAcid);
  EDB_DProt.Text := FloatToStr(vDProt);
  EDB_DPP.Text := FloatToStr(vDPP);
  EDB_DLaje.Text := FloatToStr(vDLaje);
  EDB_DCapa.Text := FloatToStr(vDCapa);
  EDB_DParede.Text := FloatToStr(vDParede);
  EDB_DRevest.Text := FloatToStr(vDRevest);
  EDB_DAcid.Text := FloatToStr(vDAcid);
  EDB_DPerda.Text := FloatToStr(vDPerda);
```

```
  EDB_DfcLaje.Text := FloatToStr(aFck[1].X);
  EDB_DfcCapa.Text := FloatToStr(aFck[2].X);
  EDB_DfcParede.Text := FloatToStr(aFck[3].X); EDB_DfcParede1.Text :=
FloatToStr(aFck[3].Y);
  EDB_DfcRevest.Text := FloatToStr(aFck[4].X); EDB_DfcRevest1.Text :=
FloatToStr(aFck[4].Y);
  EDB_DfcAcid.Text := FloatToStr(aFck[5].X); EDB_DfcAcid1.Text :=
FloatToStr(aFck[5].Y);
  EDB_DEciPP.Text := FloatToStr(aEci[0].X);
  EDB_DEciLaje.Text := FloatToStr(aEci[1].X);
  EDB_DEciCapa.Text := FloatToStr(aEci[2].X);
  EDB_DEciParede.Text := FloatToStr(aEci[3].X); EDB_DEciParede1.Text :=
FloatToStr(aEci[3].Y);
  EDB_DEciRevest.Text := FloatToStr(aEci[4].X); EDB_DEciRevest1.Text :=
FloatToStr(aEci[4].Y);
  EDB_DEciAcid.Text := FloatToStr(aEci[5].X); EDB_DEciAcid1.Text :=
FloatToStr(aEci[5].Y);
  EDB_DEciPerda.Text := FloatToStr(aEci[6].X); EDB_DEciPerda1.Text :=
FloatToStr(aEci[6].Y);
  EDB_DEcsPP.Text := FloatToStr(aEcs[0].X);
  EDB_DEcsLaje.Text := FloatToStr(aEcs[1].X);
  EDB_DEcsCapa.Text := FloatToStr(aEcs[2].X);
  EDB_DEcsParede.Text := FloatToStr(aEcs[3].X); EDB_DEcsParede1.Text :=
FloatToStr(aEcs[3].Y);
  EDB_DEcsRevest.Text := FloatToStr(aEcs[4].X); EDB_DEcsRevest1.Text :=
FloatToStr(aEcs[4].Y);
  EDB_DEcsAcid.Text := FloatToStr(aEcs[5].X); EDB_DEcsAcid1.Text :=
FloatToStr(aEcs[5].Y);
  EDB_DEcsPerda.Text := FloatToStr(aEcs[6].X); EDB_DEcsPerda1.Text :=
FloatToStr(aEcs[6].Y);
end;

procedure TFRM_DatasCoef.EDB_MPPKeyUp(Sender: TObject; var Key: Word;
Shift: TShiftState
  );
begin
  if not isNeg(TEdit(Sender).Text) then
    begin
      TEdit(Sender).Text := '';
      ShowMessage('Algarismo inválido!');
    end;
end;

//Botão Ok
procedure TFRM_DatasCoef.BTN_DTOkClick(Sender: TObject);
begin
  vMPP := (StrToFloat(IsNil(EDB_MPP.Text)));
  vMLaje := (StrToFloat(IsNil(EDB_MLaje.Text)));
  vMCapa := (StrToFloat(IsNil(EDB_MCapa.Text)));
  vMParede := (StrToFloat(IsNil(EDB_MParede.Text)));
  vMRevest := (StrToFloat(IsNil(EDB_MRevest.Text)));
  vMAcid := (StrToFloat(IsNil(EDB_MAcid.Text)));
  vDProt := (StrToInt(IsNil(EDB_DProt.Text)));
  vDPP := (StrToInt(IsNil(EDB_DPP.Text)));
  vDLaje := (StrToInt(IsNil(EDB_DLaje.Text)));
  vDCapa := (StrToInt(IsNil(EDB_DCapa.Text)));
  vDParede := (StrToInt(IsNil(EDB_DParede.Text)));
  vDRevest := (StrToInt(IsNil(EDB_DRevest.Text)));
  vDAcid := (StrToInt(IsNil(EDB_DAcid.Text)));
  vDPerda := (StrToInt(IsNil(EDB_DPerda.Text)));
  aFck[0].X := (StrToFloat(IsNil(EDB_DfcPP.Text)));
```

```
  aFck[1].X := (StrToFloat(IsNil(EDB_DfcLaje.Text)));
  aFck[2].X := (StrToFloat(IsNil(EDB_DfcCapa.Text)));
  aFck[3].X := (StrToFloat(IsNil(EDB_DfcParede.Text))); aFck[3].Y :=
(StrToFloat(IsNil(EDB_DfcParede1.Text)));
  aFck[4].X := (StrToFloat(IsNil(EDB_DfcRevest.Text))); aFck[4].Y :=
(StrToFloat(IsNil(EDB_DfcRevest1.Text)));
  aFck[5].X := (StrToFloat(IsNil(EDB_DfcAcid.Text))); aFck[5].Y :=
(StrToFloat(IsNil(EDB_DfcAcid1.Text)));
  aFck[6].X := (StrToFloat(IsNil(EDB_DfcPerda.Text))); aFck[6].Y :=
(StrToFloat(IsNil(EDB_DfcPerda1.Text)));
  aEci[0].X := (StrToFloat(IsNil(EDB_DEciPP.Text)));
  aEci[1].X := (StrToFloat(IsNil(EDB_DEciLaje.Text)));
  aEci[2].X := (StrToFloat(IsNil(EDB_DEciCapa.Text)));
  aEci[3].X := (StrToFloat(IsNil(EDB_DEciParede.Text))); aEci[3].Y :=
(StrToFloat(IsNil(EDB_DEciParede1.Text)));
  aEci[4].X := (StrToFloat(IsNil(EDB_DEciRevest.Text))); aEci[4].Y :=
(StrToFloat(IsNil(EDB_DEciRevest1.Text)));
  aEci[5].X := (StrToFloat(IsNil(EDB_DEciAcid.Text))); aEci[5].Y :=
(StrToFloat(IsNil(EDB_DEciAcid1.Text)));
  aEci[6].X := (StrToFloat(IsNil(EDB_DEciPerda.Text))); aEci[6].Y :=
(StrToFloat(IsNil(EDB_DEciPerda1.Text)));
  aEcs[0].X := (StrToFloat(IsNil(EDB_DEcsPP.Text)));
  aEcs[1].X := (StrToFloat(IsNil(EDB_DEcsLaje.Text)));
  aEcs[2].X := (StrToFloat(IsNil(EDB_DEcsCapa.Text)));
  aEcs[3].X := (StrToFloat(IsNil(EDB_DEcsParede.Text))); aEcs[3].Y :=
(StrToFloat(IsNil(EDB_DEcsParede1.Text)));
  aEcs[4].X := (StrToFloat(IsNil(EDB_DEcsRevest.Text))); aEcs[4].Y :=
(StrToFloat(IsNil(EDB_DEcsRevest1.Text)));
  aEcs[5].X := (StrToFloat(IsNil(EDB_DEcsAcid.Text))); aEcs[5].Y :=
(StrToFloat(IsNil(EDB_DEcsAcid1.Text)));
  aEcs[6].X := (StrToFloat(IsNil(EDB_DEcsPerda.Text))); aEcs[6].Y :=
(StrToFloat(IsNil(EDB_DEcsPerda1.Text)));
  FRM_DatasCoef.Close;
end;

//Botão cancelar
procedure TFRM_DatasCoef.BTN_DTCancelClick(Sender: TObject);
begin
  FRM_DatasCoef.Close;
end;

//Botão default
procedure TFRM_DatasCoef.BTN_DTDefaultClick(Sender: TObject);
begin
  EDB_MPP.Text := '1.4';
  EDB_MLaje.Text := '1.4';
  EDB_MCapa.Text := '1.4';
  EDB_MParede.Text := '1.4';
  EDB_MRevest.Text := '1.4';
  EDB_MAcid.Text := '1.4';
  EDB_DProt.Text := '1';
  EDB_DPP.Text := '1';
  EDB_DLaje.Text := '15';
  EDB_DCapa.Text := '30';
  EDB_DParede.Text := '45';
  EDB_DRevest.Text := '60';
  EDB_DAcid.Text := '75';
  EDB_DPerda.Text := '75';
  EDB_DfcLaje.Text := '0';
  EDB_DfcCapa.Text := '0';
  EDB_DfcParede.Text := '0';
```

```
  EDB_DfcRevest.Text := '0';
  EDB_DfcAcid.Text := '0';
  EDB_DEciPP.Text := '0'; EDB_DEcsPP.Text := '0';
  EDB_DEciLaje.Text := '0'; EDB_DEcsLaje.Text := '0';
  EDB_DEciCapa.Text := '0'; EDB_DEcsCapa.Text := '0';
  EDB_DEciParede.Text := '0'; EDB_DEcsParede.Text := '0';
  EDB_DEciRevest.Text := '0'; EDB_DEcsRevest.Text := '0';
  EDB_DEciAcid.Text := '0'; EDB_DEcsAcid.Text := '0';
  EDB_DEciPerda.Text := '0'; EDB_DEcsPerda.Text := '0';
  EDB_DfcParede1.Text := '0';
  EDB_DfcRevest1.Text := '0';
  EDB_DfcAcid1.Text := '0';
  EDB_DEciParede1.Text := '0'; EDB_DEcsParede1.Text := '0';
  EDB_DEciRevest1.Text := '0'; EDB_DEcsRevest1.Text := '0';
  EDB_DEciAcid1.Text := '0'; EDB_DEcsAcid1.Text := '0';
  EDB_DEciPerda1.Text := '0'; EDB_DEcsPerda1.Text := '0';
end;

//Botão para calcular a resistência do concreto em determinada idade
procedure TFRM_DatasCoef.BTN_CalcFcjClick(Sender: TObject);
var
  lFck, lFckj, lFckCapa: extended;
  aDias: array[1..7] of integer;
begin
  DecimalSeparator:='.';
  aDias[1] := StrToInt(IsNil(EDB_DPP.Text));
  aDias[2] := StrToInt(IsNil(EDB_DLaje.Text));
  aDias[3] := StrToInt(IsNil(EDB_DCapa.Text));
  aDias[4] := StrToInt(IsNil(EDB_DParede.Text));
  aDias[5] := StrToInt(IsNil(EDB_DRevest.Text));
  aDias[6] := StrToInt(IsNil(EDB_DAcid.Text));
  aDias[7] := StrToInt(IsNil(EDB_DPerda.Text));
  lFckj := StrToFloat(IsNil(EDB_DfcPP.Text));
  lFck := StrToFloat(IsNil(EDB_DfcPerda.Text));
  lFckCapa := StrToFloat(IsNil(EDB_DfcPerda1.Text));
  EDB_DfcLaje.Text := FormatFloat('###,###,##0.0', GetFcj(lFck, lFckj,
aDias[2], CBX_PMCimento.Text, aCoefS));
  EDB_DfcCapa.Text := FormatFloat('###,###,##0.0', GetFcj(lFck, lFckj,
aDias[3], CBX_PMCimento.Text, aCoefS));
  EDB_DfcParede.Text := FormatFloat('###,###,##0.0', GetFcj(lFck, lFckj,
aDias[4], CBX_PMCimento.Text, aCoefS));
  EDB_DfcRevest.Text := FormatFloat('###,###,##0.0', GetFcj(lFck, lFckj,
aDias[5], CBX_PMCimento.Text, aCoefS));
  EDB_DfcAcid.Text := FormatFloat('###,###,##0.0', GetFcj(lFck, lFckj,
aDias[6], CBX_PMCimento.Text, aCoefS));
  EDB_DfcParede1.Text := FormatFloat('###,###,##0.0', GetFcj(lFckCapa, 0,
aDias[4]-aDias[3], CBX_ILCimento.Text, aCoefS));
  EDB_DfcRevest1.Text := FormatFloat('###,###,##0.0', GetFcj(lFckCapa, 0,
aDias[5]-aDias[3], CBX_ILCimento.Text, aCoefS));
  EDB_DfcAcid1.Text := FormatFloat('###,###,##0.0', GetFcj(lFckCapa, 0,
aDias[6]-aDias[3], CBX_ILCimento.Text, aCoefS));
end;

//Botão para calcular os módulos de elasticidade tangencial e secante
procedure TFRM_DatasCoef.BTN_CalcEcClick(Sender: TObject);
var
  aDias: array[1..7] of integer;
  aFc1, aFc2: array[1..7] of extended;
begin
  aDias[1] := StrToInt(IsNil(EDB_DPP.Text));
  aDias[2] := StrToInt(IsNil(EDB_DLaje.Text));
```

```
  aDias[3] := StrToInt(IsNil(EDB_DCapa.Text));
  aDias[4] := StrToInt(IsNil(EDB_DParede.Text));
  aDias[5] := StrToInt(IsNil(EDB_DRevest.Text));
  aDias[6] := StrToInt(IsNil(EDB_DAcid.Text));
  aDias[7] := StrToInt(IsNil(EDB_DPerda.Text));
  aFc1[1] := StrToFloat(IsNil(EDB_DfcPP.Text));
  aFc1[2] := StrToFloat(IsNil(EDB_DfcLaje.Text));
  aFc1[3] := StrToFloat(IsNil(EDB_DfcCapa.Text));
  aFc1[4] := StrToFloat(IsNil(EDB_DfcParede.Text));
  aFc1[5] := StrToFloat(IsNil(EDB_DfcRevest.Text));
  aFc1[6] := StrToFloat(IsNil(EDB_DfcAcid.Text));
  aFc1[7] := StrToFloat(IsNil(EDB_DfcPerda.Text));
  aFc2[4] := StrToFloat(IsNil(EDB_DfcParede1.Text));
  aFc2[5] := StrToFloat(IsNil(EDB_DfcRevest1.Text));
  aFc2[6] := StrToFloat(IsNil(EDB_DfcAcid1.Text));
  aFc2[7] := StrToFloat(IsNil(EDB_DfcPerda1.Text));
  EDB_DEciPP.Text := IntToStr(Round(ElasticMod(vAlfaE, aFc1[7], aFc1[1],
aDias[1]).X));
  EDB_DEciLaje.Text := IntToStr(Round(ElasticMod(vAlfaE, aFc1[7], aFc1[2],
aDias[2]).X));
  EDB_DEciCapa.Text := IntToStr(Round(ElasticMod(vAlfaE, aFc1[7], aFc1[3],
aDias[3]).X));
  EDB_DEciParede.Text := IntToStr(Round(ElasticMod(vAlfaE, aFc1[7],
aFc1[4], aDias[4]).X));
  EDB_DEciRevest.Text := IntToStr(Round(ElasticMod(vAlfaE, aFc1[7],
aFc1[5], aDias[5]).X));
  EDB_DEciAcid.Text := IntToStr(Round(ElasticMod(vAlfaE, aFc1[7], aFc1[6],
aDias[6]).X));
  EDB_DEciPerda.Text := IntToStr(Round(ElasticMod(vAlfaE, aFc1[7], aFc1[7],
aDias[7]).X));
  EDB_DEcsPP.Text := IntToStr(Round(ElasticMod(vAlfaE, aFc1[7], aFc1[1],
aDias[1]).Y));
  EDB_DEcsLaje.Text := IntToStr(Round(ElasticMod(vAlfaE, aFc1[7], aFc1[2],
aDias[2]).Y));
  EDB_DEcsCapa.Text := IntToStr(Round(ElasticMod(vAlfaE, aFc1[7], aFc1[3],
aDias[3]).Y));
  EDB_DEcsParede.Text := IntToStr(Round(ElasticMod(vAlfaE, aFc1[7],
aFc1[4], aDias[4]).Y));
  EDB_DEcsRevest.Text := IntToStr(Round(ElasticMod(vAlfaE, aFc1[7],
aFc1[5], aDias[5]).Y));
  EDB_DEcsAcid.Text := IntToStr(Round(ElasticMod(vAlfaE, aFc1[7], aFc1[6],
aDias[6]).Y));
  EDB_DEcsPerda.Text := IntToStr(Round(ElasticMod(vAlfaE, aFc1[7], aFc1[7],
aDias[7]).Y));
  EDB_DEciParede1.Text := IntToStr(Round(ElasticMod(vAlfaE, aFc2[7],
aFc2[4], aDias[4]-aDias[3]).X));
  EDB_DEciRevest1.Text := IntToStr(Round(ElasticMod(vAlfaE, aFc2[7],
aFc2[5], aDias[5]-aDias[3]).X));
  EDB_DEciAcid1.Text := IntToStr(Round(ElasticMod(vAlfaE, aFc2[7], aFc2[6],
aDias[6]-aDias[3]).X));
  EDB_DEciPerda1.Text := IntToStr(Round(ElasticMod(vAlfaE, aFc2[7],
aFc2[7], aDias[7]-aDias[3]).X));
  EDB_DEcsParede1.Text := IntToStr(Round(ElasticMod(vAlfaE, aFc2[7],
aFc2[4], aDias[4]-aDias[3]).Y));
  EDB_DEcsRevest1.Text := IntToStr(Round(ElasticMod(vAlfaE, aFc2[7],
aFc2[5], aDias[5]-aDias[3]).Y));
  EDB_DEcsAcid1.Text := IntToStr(Round(ElasticMod(vAlfaE, aFc2[7], aFc2[6],
aDias[6]-aDias[3]).Y));
  EDB_DEcsPerda1.Text := IntToStr(Round(ElasticMod(vAlfaE, aFc2[7],
aFc2[7], aDias[7]-aDias[3]).Y));
end;
```

```
end.
```

9.5 LISTAGEM DA TELA DO GRÁFICO “M x LN”

```
var
  FRM_GraficoMLN: TFRM_GraficoMLN;

implementation

{$R *.lfm}

{ TFRM_GraficoMLN }

procedure TFRM_GraficoMLN.BTN_FecharClick(Sender: TObject);
begin
  FRM_GraficoMLN.Close;
end;

end.
```

9.6 LISTAGEM DA TELA DOS CRÉDITOS DO PROGRAMA

```
var
  FRM_Creditos: TFRM_Creditos;

implementation

{$R *.lfm}

{ TFRM_Creditos }

procedure TFRM_Creditos.BitBtn1Click(Sender: TObject);
begin
  FRM_Creditos.Close;
end;

end.
```

9.7 LISTAGEM COM AS FUNÇÕES DA BIBLIOTECA PARA O CÁLCULO DAS CARACTERÍSTICAS GEOMÉTRICAS

```
type

//Guarda as coordenadas do CG da seção;
//Wi/Ws da seção composta
//Lp,Inf e Lp,Sup
tPoints = record
    X: Extended;
    Y: Extended;
end;

//Guarda os valores referentes às características do concreto e dimensões
das seções
tConcreto = record
```

```
    X: Extended;
    Y: Extended;
    fck: Extended;
    AlfaC: Extended;
    Lambida: Extended;
    GamaC: Extended;
    Eci: Extended;
    Ecs: Extended;
end;

//Guarda os valores dos esforços de momento e de cortante nas seções de
décimo de vão
tEsforcos = record
    S0: Extended;
    S1: Extended;
    S2: Extended;
    S3: Extended;
    S4: Extended;
    S5: Extended;
    SLpi: Extended;
    SLps: Extended;
    SNeo: Extended;
end;

//Utilizado no dimensionamento de armadura
tArmadura = record
    KMD: Extended;
    KX: Extended;
    KZ: Extended;
    Epsc: Extended;
    Epss: Extended;
    X: Extended;
    d: Extended;
    Ap: Extended;
    C1: Extended;
    M2: Extended;
end;

//Utilizado no dimensionamento do estribo
tEstribo = record
    Vc: Extended;
    Vsw: Extended;
    s: Extended;
    Ro: Extended;
    sMax: Extended;
    xEstribo: Extended;
    Biela: String;
    TipoDim: String;
end;

//Utilizado no cálculo da fissuração
tFissura = record
    Aco: integer; //1 é ativo 2 é passivo
    X, Y: Extended;
    Fi, Ap: Extended;
    SigAco: Extended;
    Wk1, Wk2: Extended;
    a, b, c, d, Ro: Extended;
end;

tPointsInt = Array of tPoint;
```

```
tPointsDesl = Array of tPoints;
tProtensao = Array [0..7] of tPoints;
tFissuracao = Array of tFissura;

//Guarda informações sobre a armadura
tDados = class(TObject)
  vNBarra, vPBarra, vTBarra, vTAco_index, vDBarra_index: Integer;
  vTAco, vDBarra: String;
  vABarra, vXa, vYa, vXb, vYb: Extended;
  end;

function PolygonArea(N:integer;Points:Array of tPoints): extended;
function PolygonCentroid(N: integer; Points:Array of tPoints; area:
extended): tPoints;
function PolygonInertia(N: Integer; Points: array of TPoints; C: tPoints;
Area: extended): extended;
function PolygonPerimeter(Points: array of tPoints): extended;
function InerciaComposta(a, b, c, d, Iviga, yviga, hviga, bviga, Aviga:
extended): tPoints;
function SecaoCompostaW(Ixcg, ycg, hviga, b, d: extended): tPoints;
function Displac(scale: extended; Points: array of tPoints): integer;
function SecDispl(Points: array of tPoints): tPointsDesl;
function BestFit(Points: array of TPoints; IMG_Vx, IMG_Vy: Integer):
extended;
function VetorTransform(scale: extended; ImY: integer; Points: Array of
tPoints): TPointsInt;
procedure DrawSection(VetorInt: TPointsInt; Image: TImage; Ret: TRect);
procedure DrawReinforce(N, ImY, Desl: integer; scale: extended; Image:
TImage; tListaArm: TListBox);
function DrawGraphic(a, b: tPoints): tPoints;

var
  Area: extended;

implementation

//Cálculo da área de uma poligonal
function PolygonArea(N: integer; Points: Array of tPoints): extended;
var
  i,j:integer;
  area:extended;
begin
  area:=0;
  For i:= 0 to N-1 do
  begin
    j:=(i + 1) mod N;
    area := area + Points[i].X * Points[j].Y - Points[j].X * Points[i].Y;
  end;
  Result := area / 2;
end;

//Cálculo do centroid de uma poligonal
function PolygonCentroid(N: integer; Points: Array of tPoints;
area:extended): tPoints;
var
  i,j:integer;
  C:tPoints;
  P:extended;
begin
    C.X := 0;
    C.Y := 0;
```

```
    For i := 0 to N-1 do
    begin
         j:=(i + 1) mod N;
         P:= Points[i].X * Points[j].Y - Points[j].X * Points[i].Y;
         C.X := C.X + (Points[i].X + Points[j].X) * P;
         C.Y := C.Y + (Points[i].Y + Points[j].Y) * P;
    end;
    C.X := C.X / (6 * area);
    C.Y := C.Y / (6 * area);
    Result := C;
end;

//Cálculo do momento de inércia em torno do eixo x de uma poligonal
function PolygonInertia(N: Integer; Points: array of tPoints; C: tPoints;
Area:extended): extended;
var
  i,j: Integer;
  Icgx: extended;
  ai, Ix: extended;
begin
  Ix := 0;
  For i:= 0 to N-1 do begin
    J:= (i + 1) mod N;
    ai:= Points[i].X*Points[j].Y-Points[j].X*Points[i].Y;
    Ix:= Ix +
(Sqr(Points[i].Y)+Points[i].Y*Points[j].Y+Sqr(Points[j].Y))*ai/12;
  end;
  Icgx := Ix-Sqr(C.Y)*Area;
  Result := Icgx;
end;

//Calcula o perímetro do polígono
function PolygonPerimeter(Points: array of tPoints): extended;
var
  dx, dy, Per: extended;
  i, j, N: integer;
begin
  N := Length(Points);
  dx := 0; dy := 0; Per := 0;
  for i := 0 to N - 1 do begin
    j := (i + 1) mod N;
    dx:= Abs(Points[j].X-Points[i].X);
    dy:= Abs(Points[j].Y-Points[i].Y);
    Per:= Per+Sqrt(Sqr(dx)+Sqr(dy));
  end;
  result := Per * 100;
end;

//Cálculo do momento de inercia da secao composta
function InerciaComposta(a, b, c, d, Iviga, yviga, hviga, bviga, Aviga:
extended): tPoints;
var
  Alaje, Acapa, ylaje, ycapa, Ilaje, Icapa: extended;
  IxY: tPoints;
begin
  Alaje := a * b;
  Acapa := (c + a) * d;
  if b = 0 then
    Acapa := (c + bviga) * d;
  ylaje := hviga + b / 2;
  ycapa := hviga + b + d / 2;
```

```
  Ilaje := a * power(b, 3) / 12;
  Icapa := c * power(d, 3) / 12;
  IxY.Y := (Aviga * yviga + Alaje * ylaje + Acapa * ycapa) / (Aviga + Alaje
+ Acapa);
  IxY.X := Iviga + Aviga * power(IxY.Y - yviga, 2) + Ilaje + Alaje *
power(IxY.Y - ylaje, 2)
  + Icapa + Acapa * power(IxY.Y - ycapa, 2);
  result := IxY;
end;

//Cálculo do Wi e Ws da seção composta
function SecaoCompostaW(Ixcg, ycg, hviga, b, d: extended): tPoints;
var
  Wf: tPoints;
begin
  Wf.X := Ixcg / ycg;                    //Inferior
  Wf.Y := Ixcg / (hviga + b + d - ycg); //Superior
  Result := Wf;
end;

//Função que retorna o deslocamento da seção
function Displac(scale: extended; Points: array of tPoints): integer;
var
  i: integer;
  Xmin: extended;
begin
  Xmin := 0;
  for i := 0 to Length(Points) - 1 do
    Xmin := Min(Points[i].X, Xmin);
  if Xmin < 0 then
    result := round((abs(Xmin) + 0.2) * scale)
  else
    result := 0;
end;

//Função para deslocar a seção
function SecDispl(Points: array of tPoints): tPointsDesl;
var
  i: integer;
  Xmin: extended;
  Vetor: array of tPoints;
begin
  SetLength(Vetor, Length(Points));
  Xmin := 0;
  for i := 0 to Length(Points) - 1 do begin
    Xmin := Min(Points[i].X, Xmin);
    Vetor[i].X := Points[i].X;
    Vetor[i].Y := Points[i].Y;
  end;
  if Xmin >= 0 then
    result := Vetor
  else begin
    for i := 0 to Length(Points) -1 do
      Vetor[i].X := Points[i].X + abs(Xmin) + 0.2;
    result := Vetor;
  end;
end;

//Função para deixar o desenho em escala
function BestFit(Points: array of TPoints; IMG_Vx, IMG_Vy: Integer):
extended;
```

```
var
  i: integer;
  XMax, YMax, XMin, YMin, vx, vy, scale: extended;
begin
  XMax := 0; YMax := 0;
  XMin := 0; YMin := 0;
  for i := 0 to Length(Points) - 1 do begin
    XMax := Max(Points[i].X, XMax);
    YMax := Max(Points[i].Y, YMax);
    XMin := Min(Points[i].X, XMin);
    YMin := Min(Points[i].Y, YMin);
  end;
  if (XMax - XMin <> 0) or (YMax - YMin <> 0) then begin
    vx := (IMG_Vx - 5) / (XMax - XMin);
    vy := (IMG_Vy - 5) / (Ymax - YMin);
    scale := min(vx, vy);
    result := scale;
  end;
end;

//Função que passa as coordenadas do formato "Extended" para "Int"
function VetorTransform(scale: extended; ImY: integer; Points: Array of
tPoints): TPointsInt;
var
  i: integer;
  Vetor: array of TPoint;
begin
  SetLength(Vetor, Length(Points));
  For i:= 0 to Length(Points)-1 do
  begin
    Vetor[i].x := round(Points[i].X * scale) + 3;
    Vetor[i].y := - (round(Points[i].Y * scale) - ImY + 3);
  end;
  Result := Vetor;
end;

//Procedure para desenhar a seção
procedure DrawSection(VetorInt: TPointsInt; Image: TImage; Ret: TRect);
begin
  Image.Canvas.Pen.Color := clBlack;
  Image.Canvas.Brush.Style := bsSolid;
  Image.Canvas.Brush.Color := clWhite;
  Image.Canvas.FillRect(Ret);
  Image.Canvas.Brush.Color := clBtnFace;
  Image.Canvas.Polygon(VetorInt);
end;

//Procedure para colocar a armadura na seção
procedure DrawReinforce(N, ImY, Desl: integer; scale: extended; Image:
TImage; tListaArm: TListBox);
var
  i, j: integer;
  CoordXa, CoordYa, CoordXb, CoordYb: integer;
  tArmadura: tDados;
begin
  Image.Canvas.Pen.Color := clBlack;
  Image.Canvas.Brush.Style := bsSolid;
  for i := 0 to N - 1 do begin
    tArmadura := tDados(tListaArm.Items.Objects[i]);
  case tArmadura.vTBarra of
    1: begin
```

```
        Image.Canvas.Brush.Color := clGreen;
        CoordXa := round(tArmadura.vXa / 100 * scale - 2.5) + 3 + Desl;
        CoordYa := - (round(tArmadura.vYa / 100 * scale + 2.5) - ImY + 3);
        CoordXb := CoordXa + 5;
        CoordYb := CoordYa + 5;
        Image.Canvas.Ellipse(CoordXa, CoordYa, CoordXb, CoordYb);
        if tArmadura.vNBarra > 1 then begin
          for j := 0 to tArmadura.vNBarra - 3 do begin
            CoordXa := CoordXa + round(((tArmadura.vXb - tArmadura.vXa) /
(tArmadura.vNBarra - 1)) / 100 * scale);
            CoordYa := - (round(tArmadura.vYa / 100 * scale + 2.5) - ImY +
3);
            CoordXb := CoordXa + 5;
            CoordYb := CoordYa + 5;
            Image.Canvas.Ellipse(CoordXa, CoordYa, CoordXb, CoordYb);
          end;
          CoordXa := round(tArmadura.vXb / 100 * scale - 2.5) + 3 + Desl;
          CoordYa := - (round(tArmadura.vYa / 100 * scale + 2.5) - ImY + 3);
          CoordXb := CoordXa + 5;
          CoordYb := CoordYa + 5;
          Image.Canvas.Ellipse(CoordXa, CoordYa, CoordXb, CoordYb);
        end;
        end;
    2: begin
        Image.Canvas.Brush.Color := clRed;
        CoordXa := round(tArmadura.vXa / 100 * scale - 2.5) + 3 + Desl;
        CoordYa := - (round(tArmadura.vYa / 100 * scale + 2.5) - ImY + 3);
        CoordXb := CoordXa + 5;
        CoordYb := CoordYa + 5;
        Image.Canvas.Ellipse(CoordXa, CoordYa, CoordXb, CoordYb);
        if tArmadura.vNBarra > 1 then begin
          for j := 0 to tArmadura.vNBarra - 3 do begin
            CoordXa := CoordXa + round(((tArmadura.vXb - tArmadura.vXa) /
(tArmadura.vNBarra - 1)) / 100 * scale);
            CoordYa := - (round(tArmadura.vYa / 100 * scale + 2.5) - ImY +
3);
            CoordXb := CoordXa + 5;
            CoordYb := CoordYa + 5;
            Image.Canvas.Ellipse(CoordXa, CoordYa, CoordXb, CoordYb);
          end;
          CoordXa := round(tArmadura.vXb / 100 * scale - 2.5) + 3 + Desl;
          CoordYa := - (round(tArmadura.vYa / 100 * scale + 2.5) - ImY + 3);
          CoordXb := CoordXa + 5;
          CoordYb := CoordYa + 5;
          Image.Canvas.Ellipse(CoordXa, CoordYa, CoordXb, CoordYb);
        end;
        end;
  end;
  end;
end;

//Salva as coordenadas para desenho do gráfico
function DrawGraphic(a, b: tPoints): tPoints;
var
  ab: tPoints;
begin
  if a.Y <> b.Y then begin
    ab.X := a.X;
    ab.Y := b.X;
  end
  else
```

```
    if a.Y = 0 then begin
      ab.X := 0;
      ab.Y := Min(a.X, b.X);
    end
    else if a.Y = 1 then begin
      ab.X := Max(a.X, b.X);
      ab.Y := Max(a.X, b.X) + 20;
    end;
  result := ab;
end;

end.
```

9.8 LISTAGEM COM AS FUNÇÕES DA BIBLIOTECA PARA VERIFICAÇÃO DOS DADOS DE ENTRADA DE ALGARISMOS

```
function IsNum(s: string): Boolean;
function IsNil(s: string): String;
function IsNeg(s:string): Boolean;
procedure LimTen(Grid: TStringGrid; CLimit, TLimit, tension: extended; Col,
Row: integer);
function SubCons(aEsforco1: tEsforcos; cons: extended): tEsforcos;
function SubEsf(aEsforco1, aEsforco2: tEsforcos): tEsforcos;
function EditOk(var editor: TEdit): boolean;
procedure GridCleaner(Grid: TStringGrid);

implementation

//Verifica se a string é um número real
function IsNum(s: string): Boolean;
begin
  try
     result := (s = '') or (s = ',') or (s = '-') or (s = '-,') or
     (StrToFloat(s) >= 0) or (StrToFloat(s) < 0);
  except
     result := False;
  end;
end;

//Verifica se uma string é vazia
function IsNil(s: string): String;
begin
  if (s = '') or (s = ',') then
     result := '0'
  else
     result := s
end;

//Verifica se uma string possui números positivos, apenas
function IsNeg(s: string): Boolean;
begin
  try
     result := (s = '') or (s = ',') or (StrToFloat(s) >= 0);
  except
     result := False;
  end;
end;
```

```
//Verifica se a tensão está fora dos limites permitidos e muda a cor do
label
procedure LimTen(Grid: TStringGrid; CLimit, TLimit, tension: extended; Col,
Row: integer);
begin
  if (tension < TLimit) or (tension > CLimit) then begin
    Grid.Font.Color := clRed;
    Grid.Cells[Col, Row] := FormatFloat('###,###,##0', tension);
  end
  else begin
    Grid.Canvas.Font.Color := clBlack;
    Grid.Cells[Col, Row] := FormatFloat('###,###,##0', tension);
  end;
end;

//Subtrai tEsforcos com extended
function SubCons(aEsforco1: tEsforcos; cons: extended): tEsforcos;
begin
  aEsforco1.S0 := aEsforco1.S0 - cons;
  aEsforco1.S1 := aEsforco1.S1 - cons;
  aEsforco1.S2 := aEsforco1.S2 - cons;
  aEsforco1.S3 := aEsforco1.S3 - cons;
  aEsforco1.S4 := aEsforco1.S4 - cons;
  aEsforco1.S5 := aEsforco1.S5 - cons;
  aEsforco1.SLpi := aEsforco1.SLpi - cons;
  aEsforco1.SLps := aEsforco1.SLps - cons;
  result := aEsforco1;
end;

//Subtrai tEsforco com tEsforco
function SubEsf(aEsforco1, aEsforco2: tEsforcos): tEsforcos;
begin
  aEsforco1.S0 := aEsforco1.S0 - aEsforco2.S0;
  aEsforco1.S1 := aEsforco1.S1 - aEsforco2.S1;
  aEsforco1.S2 := aEsforco1.S2 - aEsforco2.S2;
  aEsforco1.S3 := aEsforco1.S3 - aEsforco2.S3;
  aEsforco1.S4 := aEsforco1.S4 - aEsforco2.S4;
  aEsforco1.S5 := aEsforco1.S5 - aEsforco2.S5;
  aEsforco1.SLpi := aEsforco1.SLpi - aEsforco2.SLpi;
  aEsforco1.SLps := aEsforco1.SLps - aEsforco2.SLps;
  result := aEsforco1;
end;

//Verifica se a editbox está preenchida
function EditOk(var editor: TEdit): boolean;
begin
  result:= (Editor.Text = '') or (Editor.Text = '0');
end;

//Limpa os valores '-' do StringGrid
procedure GridCleaner(Grid: TStringGrid);
var
  i, j: integer;
begin
  for i := 1 to 2 do
    for j := 1 to 5 do
      if Grid.Cells[i,j] = '-' then
        Grid.Cells[i,j] := '';
end;

end.
```

9.9 LISTAGEM COM AS FUNÇÕES DA BIBLIOTECA DE ENGENHARIA

```
function GetFcj(fck, fck1: extended; t: integer; Endur: string; CoefS:
array of extended): extended;
function ElasticMod(AlfaE, fck, fc: extended; t: integer): tPoints;
function Momentos_Vao(p: extended; L: extended): tEsforcos;
function Momento_Lp(p, L, Lp: extended): extended;
function Cortantes_Vao(p, L, Lpi, Lps, LNeo: extended): tEsforcos;
function GetApArea(N: Integer; tLista_Arm: TListBox; Ycg: tPoints):
tPoints;
function GetAsArea(N: Integer; tLista_Arm: TListBox; Ycg: tPoints):
tPoints;
function GetExcentricity_Pos(N: Integer; tLista_Arm: TListBox; Ycg:
tPoints): Extended;
function GetExcentricity_Neg(N: Integer; tLista_Arm: TListBox; Ycg:
tPoints): Extended;
function GetD(N: Integer; tLista_Arm: TListBox; Ycg: tPoints; ht, sigmaP,
sigmaP2: extended): tPoints;
function GetNp(Ap, Sig, n, Lp, L: extended): extended;
function GetSecNp(Ap, Lpi, Lps, L: extended; aNLp: array of integer; SigI,
SigS: tEsforcos): tProtensao;
function GetSupTension(Npi, Nps, ei, es, A, Wsi, Wsf, Mp, Mp2, Mq, psi:
Extended): Extended;
function GetInfTension(Npi, Nps, ei, es, A, Wii, Wif, Mp, Mp2, Mq, psi:
Extended): Extended;
function GetSupAp(ApS, SigI, nS, SigS, ei, es, A, WsI, WsF, Mp, Mp2, Mq,
psi, TenLim: extended): tPoints;
function GetInfAp(ApS, SigI, nS, SigS, ei, es, A, WiI, WiF, Mp, Mp2, Mq,
psi, TenLim: extended): tPoints;
function PrestressedTension(fpyd, epsyd, fptd, epsu, Ep, Eps: Extended):
Extended;
function PrestressedDeformation(fpyd, epsyd, fptd, epsu, Ep, SigmaP:
Extended): Extended;
function DescompDef(Np, ep, Ac, I, Ec: Extended): Extended;
function GetEps(Md, d, limX: extended; aDim: array of tConcreto; LineS:
tLineSeries): tArmadura;
function GetAp(d, sigmaPd: extended; tArm: tArmadura): extended;
function GetITension(fptk, fpyk: Extended): Extended;
function AsConvert(Md, KZ, d, gamaS, sigmaSd: extended; tLista_Arm:
TListBox): Extended;
function CompTrans(fckj, fi, gamaC, sigma, eta1, eta2, h: extended; CBX:
TComboBox): extended;
function DefAnc(deltaL, LPista, Ep: extended): extended;
function RelArm(SigPi, fptk: extended; Dias, DiasFin: integer; aQsiMil, aR:
array of extended): extended;
function DefConInf(Npi, Nps, A, ei, es, I, AlfaP, M: extended): extended;
function DefConSup(Npi, Nps, A, ei, es, I, AlfaP, M: extended): extended;
function CoefFlu(Conc: integer; T, tDelta, tDeltaInf, U, Ac, UAr, fiD,
slump: extended; Endur: string; Alfa, CoefS, Fc: array of extended):
extended;
function PerdaFluInf(Npi, Nps, A, ei, es, eic, I, Ic, psi2, AlfaP:
extended; Momentos, Fif: array of extended): extended;
function PerdaFluSup(Npi, Nps, A, ei, es, esc, I, Ic, psi2, AlfaP:
extended; Momentos, Fif: array of extended): extended;
function MedPon(A1, A2, Fi1, Fi2: extended): extended;
function Retrac(U, Ac, UAr, slump, T, tDelta, tDeltaInf, Ep: extended):
extended;
function Progressiva(PFlu, PRet, SigRel, PsiRel, FiFlu, ep, Ac, I, Ap,
AlfaP: extended): extended;
function MomFiss(fck, Npi, Nps, EpI, EpS, alfa1, alfa2, Ac, W1, W2:
extended): extended;
```

```
function FlechaImed(k: integer; Lpi, Lps, Ei, Es, p, L, Ec, I: extended;
NpLp: array of extended): extended;
function FlechaDif(FImed, FiFlu, coefE: extended): extended;
function Branson(M, Mr, I, IxII: extended): extended;
function TransversalI(Npi, Nps, gF, gS, Ei, Es, fck, gamaC, bw, d, A, Wif,
Vsd, Msd, Asw, fywk, alfa, L, p: extended): tEstribo;
function TransversalII(Npi, Nps, gF, gS, Ei, Es, fck, gamaC, bw, d, A, Wif,
Vsd, Msd, Asw, fywk, alfa, teta, L, p: extended): tEstribo;
function InterfaceTrans(Fhd, L, Fck, gC, Fyk, gS, Asw: extended; aDim:
array of tConcreto; Beta: array of tPoints): tPoints;
function InertiaII(dI, dS, ApI, ApS, AsI, AsS, Ep, Es: extended; Sec: array
of tConcreto): tPoints;
function AbertFiss(Npi, Nps, Ac, epi, I1, I2, Ycg, EtaP, EtaS, Ep, Es, Ec,
Base, fck, Mfreq: extended; tLista_Arm: TListBox): tFissuracao;
function VerificFiss(ind: integer; EtaP, EtaS, Ep, Es: extended; Fissura:
tFissura): tFissura;

implementation

//Função para calcular a resistência do concreto
function GetFcj(fck, fck1: extended; t: integer; Endur: string; CoefS:
array of extended): extended;
var
  s: extended;
begin
  if Endur = 'Lento' then
    s := CoefS[0]
  else if Endur = 'Normal' then
    s := CoefS[1]
  else if Endur = 'Rápido' then
    s := CoefS[2];
  if t >= 28 then
    result := fck
  else if t = 1 then
    result := fck1
  else
    result := exp(s * (1 - power(28/t, 0.5))) * fck;
end;

//Função para calcular o módulo de elasticidade inicial do concreto (Eci)
function ElasticMod(AlfaE, fck, fc: extended; t: integer): tPoints;
var
  AlfaI: extended;
  Ec: tPoints;
begin
  if fck = 0 then begin
    Ec.X := 0; Ec.Y := 0;
    result := Ec;
    exit;
  end;
  //Cálculo do módulo de elasticidade tangencial (Eci) Ec.X
  if fck <= 50 then
    Ec.X := AlfaE * 5600 * power(fck, 0.5)
  else
    Ec.X := 21.5 * 1000 * AlfaE * power(fck / 10 + 1.25, 1/3);
  if (t < 28) and (fck < 50) then
    Ec.X := power(fc / fck, 0.5) * Ec.X
  else if (t < 28) and (fck >= 50) then
    Ec.X := power(fc / fck, 0.3) * Ec.X;
  //Cálculo do módulo de elasticidade secante (Ecs) Ec.Y
  AlfaI := 0.8 + 0.2 * fck / 80;
```

```
  if AlfaI > 1 then
    AlfaI := 1;
  Ec.Y := AlfaI * Ec.X;
  result := Ec;
end;

//Função que define os momentos em cada décimo de vão e retorna
function Momentos_Vao(p, L: extended): tEsforcos;
var
  momentos: tEsforcos;
begin
  momentos.S0 := 0;
  momentos.S1 := p / 2 * (L * 1 * L / 10 - power((1 * L / 10), 2));
  momentos.S2 := p / 2 * (L * 2 * L / 10 - power((2 * L / 10), 2));
  momentos.S3 := p / 2 * (L * 3 * L / 10 - power((3 * L / 10), 2));
  momentos.S4 := p / 2 * (L * 4 * L / 10 - power((4 * L / 10), 2));
  momentos.S5 := p / 2 * (L * 5 * L / 10 - power((5 * L / 10), 2));
  Result := momentos;
end;

//Função que define o momento na seçao do comprimento de transferência
function Momento_Lp(p, L, Lp: extended): extended;
begin
  result := p / 2 * (L * Lp - power(Lp, 2));
end;

//Função que define os momentos em cada décimo de vão e retorna
function Cortantes_Vao(p, L, Lpi, Lps, LNeo: extended): tEsforcos;
var
  cortantes: tEsforcos;
begin
  cortantes.S0 := p / 2 * (L - 2 * L / 10 * 0);
  cortantes.S1 := p / 2 * (L - 2 * L / 10 * 1);
  cortantes.S2 := p / 2 * (L - 2 * L / 10 * 2);
  cortantes.S3 := p / 2 * (L - 2 * L / 10 * 3);
  cortantes.S4 := p / 2 * (L - 2 * L / 10 * 4);
  cortantes.S5 := p / 2 * (L - 2 * L / 10 * 5);
  cortantes.SLpi := p / 2 * (L - 2 * Lpi);
  cortantes.SLps := p / 2 * (L - 2 * Lps);
  cortantes.SNeo := p / 2 * (L - 2 * LNeo);
  Result := cortantes;
end;

//Define a área de armadura ATIVA inferior e superior
function GetApArea(N: Integer; tLista_Arm: TListBox; Ycg: tPoints):
tPoints;
var
  i: integer;
  tArmadura: tDados;
  Ycgp: extended;
  Ap: tPoints; //Posição X Ap inferior e Y superior
begin
  Ap.X := 0; Ap.Y := 0;
  for i := 0 to N - 1 do
  begin
  tArmadura := tDados(tLista_Arm.Items.Objects[i]);
    if ((tArmadura.vPBarra = 1) and (tArmadura.vTBarra = 1)) then
      Ap.X := Ap.X + tArmadura.vNBarra * tArmadura.vABarra
    else if ((tArmadura.vPBarra = 2) and (tArmadura.vTBarra = 1)) then
      Ap.Y := Ap.Y + tArmadura.vNBarra * tArmadura.vABarra;
  end;
```

```
    Result := Ap;
end;

//Define a área de armadura PASSIVA inferior e superior
function GetAsArea(N: Integer; tLista_Arm: TListBox; Ycg: tPoints):
tPoints;
var
  i: integer;
  tArmadura: tDados;
  Ycgp: extended;
  Asd: tPoints; //Posição X As inferior e Y superior
begin
  Asd.X := 0; Asd.Y := 0;
  for i := 0 to N - 1 do
  begin
  tArmadura := tDados(tLista_Arm.Items.Objects[i]);
    if ((tArmadura.vPBarra = 1) and (tArmadura.vTBarra = 2)) then
      Asd.X := Asd.X + tArmadura.vNBarra * tArmadura.vABarra
    else if ((tArmadura.vPBarra = 2) and (tArmadura.vTBarra = 2)) then
      Asd.Y := Asd.Y + tArmadura.vNBarra * tArmadura.vABarra;
  end;
    Result := Asd;
end;

//Define o yCG da armadura ativa INFERIOR e retorna a excentricidade
function GetExcentricity_Pos(N: Integer; tLista_Arm: TListBox; Ycg:
tPoints): Extended;
var
  i: integer;
  tArmadura: tDados;
  Ycgp: extended;
  A, B: extended;
begin
  A := 0;
  B := 0;
  for i := 0 to N - 1 do
  begin
  tArmadura := tDados(tLista_Arm.Items.Objects[i]);
    if ((tArmadura.vPBarra = 1) and (tArmadura.vTBarra = 1)) then begin
      A := A + tArmadura.vNBarra * tArmadura.vABarra * tArmadura.vYa;
      B := B + tArmadura.vNBarra * tArmadura.vABarra;
    end;
  end;
  if A <> 0 then begin
    Ycgp := A / B;
    Result := Ycg.Y * 100 - Ycgp;
  end
  else
      Result := 0;
end;

//Define o yCG da armadura ativa SUPERIOR e retorna a excentricidade
function GetExcentricity_Neg(N: Integer; tLista_Arm: TListBox; Ycg:
tPoints): Extended;
var
  i: integer;
  tArmadura: tDados;
  Ycgp: extended;
  A, B: extended;
begin
  A := 0;
```

```
  B := 0;
  for i := 0 to N - 1 do
  begin
  tArmadura := tDados(tLista_Arm.Items.Objects[i]);
    if ((tArmadura.vPBarra = 2) and (tArmadura.vTBarra = 1)) then begin
      A := A + tArmadura.vNBarra * tArmadura.vABarra * tArmadura.vYa;
      B := B + tArmadura.vNBarra * tArmadura.vABarra;
    end;
  end;
  if A <> 0 then begin
    Ycgp := A / B;
    Result := Ycgp - Ycg.Y * 100;
  end
  else
    Result := 0;
end;

//Define o yCG da armadura ativa e passiva INFERIOR e retorna a
excentricidade
function GetD(N: Integer; tLista_Arm: TListBox; Ycg: tPoints; ht, sigmaP,
sigmaP2: extended): tPoints;
var
  i: integer;
  tArmadura: tDados;
  Ycgp: extended;
  A, B, Tension: extended;
  D: tPoints; //Posição X é altura útil da armadura inferior e Y da
superior
begin
  //Altura útil da armadura inferior
  A := 0;
  B := 0;
  for i := 0 to N - 1 do
  begin
  tArmadura := tDados(tLista_Arm.Items.Objects[i]);
    if (tArmadura.vPBarra = 1) then begin
      if tArmadura.vTBarra = 1 then
        Tension := sigmaP
      else if tArmadura.vTBarra = 2 then begin
        case tArmadura.vTAco_index of
          0: Tension := 25;
          1: Tension := 50;
          2: Tension := 60;
        end;
      end;
      A := A + Tension * tArmadura.vNBarra * tArmadura.vABarra *
tArmadura.vYa;
      B := B + Tension * tArmadura.vNBarra * tArmadura.vABarra;
    end;
  end;
  if A <> 0 then begin
    Ycgp := A / B;
    D.X := ht - Ycgp / 100;
  end
  else
      D.X := 0;
  //Altura útil da armadaura superior
  A := 0;
  B := 0;
  for i := 0 to N - 1 do
  begin
```

```
  tArmadura := tDados(tLista_Arm.Items.Objects[i]);
    if (tArmadura.vPBarra = 2) then begin
      if tArmadura.vTBarra = 1 then
        Tension := sigmaP2
      else if tArmadura.vTBarra = 2 then begin
        case tArmadura.vTAco_index of
          0: Tension := 25;
          1: Tension := 50;
          2: Tension := 60;
        end;
      end;
      A := A + Tension * tArmadura.vNBarra * tArmadura.vABarra *
tArmadura.vYa;
      B := B + Tension * tArmadura.vNBarra * tArmadura.vABarra;
    end;
  end;
  if A <> 0 then begin
    Ycgp := A / B;
    D.Y := ht - Ycgp / 100;
  end
  else
    D.Y := 0;
  result := D;
end;

//Retorna o valor de Np considerando o comprimento de regularização
function GetNp(Ap, Sig, n, Lp, L: extended): extended;
  begin
    if Lp = 0 then
      result := n * Ap * Sig
    else if (L / Lp < 1) then
      result := L / Lp * n * Ap * Sig
    else
      result := n * Ap * Sig;
  end;

//Retorna o valor da força normal inferior e superior em cada seção
function GetSecNp(Ap, Lpi, Lps, L: extended; aNLp: array of integer; SigI,
SigS: tEsforcos): tProtensao;
var
  i: integer;
  aNp: array [0..7] of tPoints;
begin
  aNp[1].X := GetNp(Ap, SigI.S1/10, aNLp[1], Lpi, 1*L/10);
  aNp[1].Y := GetNp(Ap, SigS.S1/10, aNLp[0], Lps, 1*L/10);
  aNp[2].X := GetNp(Ap, SigI.S2/10, aNLp[1], Lpi, 2*L/10);
  aNp[2].X := aNp[2].X + GetNp(Ap, SigI.S2/10, aNLp[2], Lpi, 1*L/10);
  aNp[2].Y := GetNp(Ap, SigS.S2/10, aNLp[0], Lps, 2*L/10);
  aNp[3].X := GetNp(Ap, SigI.S3/10, aNLp[1], Lpi, 3*L/10);
  aNp[3].X := aNp[3].X + GetNp(Ap, SigI.S3/10, aNLp[2], Lpi, 2*L/10);
  aNp[3].X := aNp[3].X + GetNp(Ap, SigI.S3/10, aNLp[3], Lpi, 1*L/10);
  aNp[3].Y := GetNp(Ap, SigS.S3/10, aNLp[0], Lps, 3*L/10);
  aNp[4].X := GetNp(Ap, SigI.S4/10, aNLp[1], Lpi, 4*L/10);
  aNp[4].X := aNp[4].X + GetNp(Ap, SigI.S4/10, aNLp[2], Lpi, 3*L/10);
  aNp[4].X := aNp[4].X + GetNp(Ap, SigI.S4/10, aNLp[3], Lpi, 2*L/10);
  aNp[4].X := aNp[4].X + GetNp(Ap, SigI.S4/10, aNLp[4], Lpi, 1*L/10);
  aNp[4].Y := GetNp(Ap, SigS.S4/10, aNLp[0], Lps, 4*L/10);
  aNp[5].X := GetNp(Ap, SigI.S5/10, aNLp[1], Lpi, 5*L/10);
  aNp[5].X := aNp[5].X + GetNp(Ap, SigI.S5/10, aNLp[2], Lpi, 4*L/10);
  aNp[5].X := aNp[5].X + GetNp(Ap, SigI.S5/10, aNLp[3], Lpi, 3*L/10);
  aNp[5].X := aNp[5].X + GetNp(Ap, SigI.S5/10, aNLp[4], Lpi, 2*L/10);
```

```
  aNp[5].X := aNp[5].X + GetNp(Ap, SigI.S5/10, aNLp[5], Lpi, 1*L/10);
  aNp[5].Y := GetNp(Ap, SigS.S5/10, aNLp[0], Lps, 5*L/10);
  i := 1;
  if Lpi > L / 10 then begin //SLpi
    aNp[6].X := GetNp(Ap, SigI.SLpi/10, aNLp[i], Lpi, Lpi);
    while Lpi > i * L / 10 do begin
      aNp[6].X := aNp[6].X + GetNp(Ap, SigI.SLpi/10, aNLp[i+1], Lpi, Lpi-
i*L/10);
      i := i + 1;
    end
  end
  else
    aNp[6].X := GetNp(Ap, SigI.SLpi/10, aNLp[i], Lpi, Lpi);
  aNp[6].Y := GetNp(Ap, SigS.SLpi/10, aNLp[0], Lps, Lpi); //SLpi
  i := 1;
  if Lps > L / 10 then begin //SLps
    aNp[7].X := GetNp(Ap, SigI.SLps/10, aNLp[i], Lpi, Lps);
    while Lps > i * L / 10 do begin
      aNp[7].X := aNp[7].X + GetNp(Ap, SigI.SLps/10, aNLp[i+1], Lpi, Lps-
i*L/10);
      i := i + 1;
    end
  end
  else
    aNp[7].X := GetNp(Ap, SigI.SLps/10, aNLp[i], Lpi, Lps);
  aNp[7].Y := GetNp(Ap, SigS.SLps/10, aNLp[0], Lps, Lps); //SLps
  result := aNp;
end;


//Define a tensão atuante na borda superior
function GetSupTension(Npi, Nps, ei, es, A, Wsi, Wsf, Mp, Mp2, Mq, psi:
extended): Extended;
  begin
    result := Npi / A - Npi * ei / Wsi + Mp / Wsi + Mp2 / Wsf + psi * Mq /
Wsf + Nps / A + Nps * es / Wsi;
  end;

//Define a tensão atuante na borda inferior
function GetInfTension(Npi, Nps, ei, es, A, Wii, Wif, Mp, Mp2, Mq, psi:
extended): Extended;
  begin
    result := Npi / A + Npi * ei / Wii - Mp / Wii - Mp2 / Wif - psi * Mq /
Wif + Nps / A - Nps * es / Wii;
  end;

//Define a quantidade de Ap de acordo com a fibra superior
function GetSupAp(ApS, SigI, nS, SigS, ei, es, A, WsI, WsF, Mp, Mp2, Mq,
psi, TenLim: extended): tPoints;
var
  tAp: tPoints;
begin
  if ei = WsI / A then begin
    ShowMessage('Erro. Excentricidade inferior assumiu um valor não
permitido. Altere-o!');
    Exit;
  end;
  tAp.X := (TenLim - SigS * ApS * nS / A - (SigS * nS * ApS * es + Mp) /
WsI - (Mp2 + psi * Mq) / WsF) / (SigI / A - SigI * ei / WsI);
  if tAp.X < 0 then
    tAp.X := 0;
```

```
  if ei > WsI / A then
    tAp.Y := 1
  else begin
    tAp.Y := 0;
  end;
  result := tAp;
end;

//Define a quantidade de Ap de acordo com a fibra inferior
function GetInfAp(ApS, SigI, nS, SigS, ei, es, A, WiI, WiF, Mp, Mp2, Mq,
psi, TenLim: extended): tPoints;
var
  tAp: tPoints;
begin
  tAp.X := (TenLim - SigS * ApS * nS / A + (SigS * nS * ApS * es + Mp) /
WiI + (Mp2 + psi * Mq) / WiF) / (SigI / A + SigI * ei / WiI);
  if tAp.X < 0 then
    tAp.X := 0;
  tAp.Y := 0;
  result := tAp;
end;

//Define a tensão na cordoalha a partir de sua deformação
function PrestressedTension(fpyd, epsyd, fptd, epsu, Ep, Eps: Extended):
Extended;
begin
  if Eps / 10 <= epsyd then
    Result := fpyd * Eps / (10 * epsyd)
  else
    Result := fpyd + ((fptd - fpyd) / (epsu - epsyd) * (Eps / 10 - epsyd));
end;

//Define a deformação na cordoalha provocada por uma tensão
function PrestressedDeformation(fpyd, epsyd, fptd, epsu, Ep, SigmaP:
Extended): Extended;
begin
  if SigmaP * 10 <= fpyd then
    Result := epsyd * 10 * SigmaP * 10 / fpyd
  else
    Result := epsyd + ((SigmaP * 10 - fpyd) / (fptd - fpyd) * (epsu -
epsyd));
end;

//Define a deformação por descompressão
function DescompDef(Np, ep, Ac, I, Ec: Extended): Extended;
begin
  result := (Np / Ac + Np * power(ep,2) / I) / Ec;
end;

//Retorna dados referentes a armadura
function GetEps(Md, d, limX: extended; aDim: array of tConcreto; LineS:
tLineSeries): tArmadura;
var
  i, j, k: integer;
  M1, MGraph: extended;
  x, b, h, ht, fckM, alfaCM, lambidaM, gamaCM: extended;
  Afck, Aalfa, Alambida, Agama: extended;
  Secao: array [0..3] of tConcreto;
  tArm: tArmadura;
begin
  j := 0;
```

```
  LineS.Clear;
  for i := 0 to Length(Secao) - 1 do
    if aDim[i].X <> 0 then begin
      Secao[j].X := aDim[i].X; Secao[j].Y := aDim[i].Y; Secao[j].fck :=
aDim[i].fck;
      Secao[j].AlfaC := aDim[i].AlfaC; Secao[j].Lambida := aDim[i].Lambida;
      Secao[j].GamaC := aDim[i].GamaC;
      j := j + 1;
    end;

  tArm.X := limX * d; x := tArm.X + 1; k := 0;
  while abs(tArm.X - x) >= 0.001 do begin
    tArm.X := limX * d - 0.0001 * k;
    if tArm.X <= 0 then begin
      ShowMessage('LN acima do limite x/d!');
      Exit;
    end;
    x := tArm.X;
    j := 0;
    b := Secao[0].X; h := Secao[0].Y; ht := 0;
    tArm.M2 := Md; tArm.C1 := 0;
    while tArm.X > h do begin
      j := j + 1;
      h := h + Secao[j].Y;
    end;

    for i := 0 to j do
      b := Min(b, Secao[i].X);

    MGraph := 0;
    for i := 0 to j - 1 do begin
      if Secao[i].X - b > 0 then begin
        M1 := Secao[i].AlfaC * Secao[i].fck * 1000 / Secao[i].GamaC *
Secao[i].Y * (Secao[i].X - b) *
              (d - (ht + Secao[i].Y / 2));
        tArm.C1 := tArm.C1 + M1 / (d - (ht + Secao[i].Y / 2));
        tArm.M2 := tArm.M2 - M1;
        MGraph := MGraph + M1;
      end;
      ht := ht + Secao[i].Y;
    end;
    if Secao[j].X - b > 0 then begin
      M1 := Secao[j].AlfaC * Secao[j].fck * 1000 / Secao[j].GamaC * (tArm.X
- ht) * (Secao[j].X - b) *
            (d - (ht + (tArm.X - ht) / 2));
      tArm.C1 := tArm.C1 + M1 / (d - (ht + (tArm.X - ht) / 2));
      tArm.M2 := tArm.M2 - M1;
      MGraph := MGraph + M1;
    end;

    Afck := 0; Aalfa := 0; Alambida := 0; Agama := 0;
    for i := 0 to j - 1 do begin
      Afck := Afck + Secao[i].Y * Secao[i].fck;
      Aalfa := Aalfa + Secao[i].Y * Secao[i].AlfaC;
      Alambida := Alambida + Secao[i].Y * Secao[i].Lambida;
      Agama := Agama + Secao[i].Y * Secao[i].GamaC;
    end;
    Afck := Afck + (tArm.X - ht) * Secao[j].fck;
    Aalfa := Aalfa + (tArm.X - ht) * Secao[j].AlfaC;
    Alambida := Alambida + (tArm.X - ht) * Secao[j].Lambida;
    Agama := Agama + (tArm.X - ht) * Secao[j].GamaC;
```

```
    fckM := Afck / tArm.X;
    alfaCM := Aalfa / tArm.X;
    lambidaM := Alambida / tArm.X;
    gamaCM := Agama / tArm.X;
    MGraph := MGraph + alfaCM * fckM * 1000 / gamaCM * b * tArm.X * (d -
tArm.X / 2);
    LineS.AddXY(tArm.X, MGraph);
    tArm.KMD := tArm.M2 / (alfaCM * fckM * 1000 / gamaCM * b * power(d,2));
    if tArm.KMD >= 0.5 then begin
      ShowMessage('Seção insuficiente para o carregamento dado. Altere seus
dados!');
      Result := tArm;
      Exit;
    end;
    tArm.KX := 1 - power(1 - 2 * tArm.KMD,0.5);
    tArm.X := tArm.KX * d;
    k := k + 1;
  end;

  tArm.d := d;
  tArm.KZ := 1 - 0.5 * tArm.KX;
  tArm.Epss := 10;
  tArm.Epsc := tArm.Epss / (lambidaM / tArm.KX - 1);
  if tArm.Epsc >= 3.5 then begin
    tArm.Epsc := 3.5;
    tArm.Epss := tArm.Epsc * (lambidaM / tArm.KX - 1);
  end;
  result := tArm;
end;

//Calcula a área de armadura ativa no ELU
function GetAp(d, sigmaPd: extended; tArm: tArmadura): extended;
begin
  Result := tArm.C1 / sigmaPd + tArm.M2 / (tArm.KZ * d * sigmaPd);
end;

//Calcula protensão inicial máxima para pré-dimensionamento da armadura
ativa
function GetITension(fptk, fpyk: extended): Extended;
begin
  result := Min(0.77 * fptk, 0.85 * fpyk);
end;

//Converte As em Ap
function AsConvert(Md, KZ, d, gamaS, sigmaSd: extended; tLista_Arm:
TListBox): Extended;
var
  i: integer;
  Tension, Fs, Fp: extended;
  tArmadura: tDados;
begin
  Fs := 0; Fp := 0;
  for i := 0 to tLista_Arm.Count - 1 do
  begin
  tArmadura := tDados(tLista_Arm.Items.Objects[i]);
    if (tArmadura.vPBarra = 1) then
      if tArmadura.vTBarra = 1 then
        Fp := Fp + tArmadura.vNBarra * tArmadura.vABarra * sigmaSD;
      if tArmadura.vTBarra = 2 then begin
        case tArmadura.vTAco_index of
          0: Tension := 25;
```

```
          1: Tension := 50;
          2: Tension := 60;
        end;
        Fs := Fs + tArmadura.vABarra * tArmadura.vNBarra * Tension / gamaS;
      end;
    end;
    result := Fp + Fs;
end;

//Cálculo do comprimento de transferência
function CompTrans(fckj, fi, gamaC, sigma, eta1, eta2, h: extended; CBX:
TComboBox): extended;
var
  fctd, fbpd, lbpt: extended;
begin
  fctd := 0.21 * power(fckj, 2/3) / gamaC;
  fbpd := eta1 * eta2 * fctd;
  case CBX.ItemIndex of
    1: lbpt := 7 * 0.5 * fi * sigma / (36 * fbpd);
    2: lbpt := 7 * 0.625 * fi * sigma / (36 * fbpd);
  end;
  result := Max(lbpt, power(power(h,2) + power(0.6*lbpt,2), 0.5));
end;

//Perda de protensão: Deformação da ancoragem
function DefAnc(deltaL, LPista, Ep: extended): extended;
begin
  if LPista = 0 then begin
    ShowMessage('O comprimento da pista deve ser maior que zero!');
    Exit;
  end
  else
    result := Ep * deltaL / LPista / 1000;
end;

//Perda de protensão: Relaxação da armadura
function RelArm(SigPi, fptk: extended; Dias, DiasFin: integer; aQsiMil, aR:
array of extended): extended;
var
  R, k: extended;
begin
  R := SigPi / fptk;
  if R <= aR[0] then
    k := 0
  else if R <= aR[1] then
    k := aQsiMil[1] - (aR[1] - R) * (aQsiMil[1] - aQsiMil[0]) / (aR[1] -
aR[0])
  else if R <= aR[2] then
    k := aQsiMil[2] - (aR[2] - R) * (aQsiMil[2] - aQsiMil[1]) / (aR[2] -
aR[1])
  else if R <= aR[3] then
    k := aQsiMil[3] - (aR[3] - R) * (aQsiMil[3] - aQsiMil[2]) / (aR[3] -
aR[2]);

  if DiasFin < 10000 then
    result := k * Power((DiasFin - Dias) / 41.67, 0.15) / 100
  else
    result := k * 2.5 / 100;
end;

//Perda de protensão: Deformação imediata do concreto - Borda inferior
```

```
function DefConInf(Npi, Nps, A, ei, es, I, AlfaP, M: extended): extended;
var
  Mp: extended;
begin
  Mp := Npi * ei - Nps * es;
  result := AlfaP * ((Npi + Nps) / A + (Mp - M) / I * ei) / 1000;
end;

//Perda de protensão: Deformação imediata do concreto - Borda superior
function DefConSup(Npi, Nps, A, ei, es, I, AlfaP, M: extended): extended;
var
  Mp: extended;
  aPerda: tEsforcos;
begin
  Mp := Npi * ei - Nps * es;
  result := AlfaP * ((Npi + Nps) / A + (- Mp + M) / I * es) / 1000;
end;

//Perda de protensão e flecha: Coeficiente de fluência
function CoefFlu(Conc: integer; T, tDelta, tDeltaInf, U, Ac, UAr, fiD,
slump: extended; Endur: string; Alfa, CoefS, Fc: array of extended):
extended;
var
  i: integer;
  A, B, C, D: extended;
  tFic, tFicInf, Beta1, BetaFt, BetaFinf, BetaD, gama, hFic, AlfaC, s,
FcTo: extended;
  fiA, fiF, fi1c, fi2c: extended;
begin
  if Endur = 'Lento' then
    i := 0
  else if Endur = 'Normal' then
    i := 1
  else if Endur = 'Rápido' then
    i := 2;
  AlfaC := Alfa[i];
  s := CoefS[i];
  FcTo := Fc[i];
  //1-Cálculo da idade fictícia
  tFic := AlfaC * ((T + 10) / 30 * tDelta);
  tFicInf := AlfaC * ((T + 10) / 30 * tDeltaInf);
  if tFic < 3 then
    tFic := 3;
  if tFicInf >= 10000 then
    tFicInf := 10000;
  //2-Cálculo do coeficiente de fluência rápida (fiA)
  Beta1 := EXP(s * (1 - power(28 / tDelta, 0.5)));
  if Conc = 1 then
    fiA := 0.8 * (1 - Beta1 / FcTo)
  else if Conc = 2 then
    fiA := 1.4 * (1 - Beta1 / FcTo);
  //3-Cálculo do coeficiente de deformação lenta irreversível (fiF)
  if Ac = 0 then
    Exit;
  gama := 1 + EXP(-7.8 + 0.1 * U);
  hFic := gama * 2 * Ac / UAr;
  fi1c := 4.45 - 0.035 * U;
  if slump <= 4 then
    fi1c := fi1c * 0.75
  else if slump >= 10 then
    fi1c := fi1c * 1.25;
```

```
  fi2c := (42 + hFic) / (20 + hFic);
  if Conc = 1 then
    fiF := fi1c * fi2c
  else if Conc = 2 then
    fiF := 0.45 * fi1c * fi2c;
  //4-Coeficiente BetaF e BetaD
  if hFic < 5 then
    hFic := 5
  else if hFic > 160 then
    hFic := 160;
  A := 42 * power(hFic/100, 3) - 350 * power(hFic/100, 2) + 588 * hFic/100
+ 113;
  B := 768 * power(hFic/100, 3) - 3060 * power(hFic/100, 2) + 3234 *
hFic/100 -23;
  C := -200 * power(hFic/100, 3) + 13 * power(hFic/100, 2) + 1090 *
hFic/100 + 183;
  D := 7579 * power (hFic/100, 3) - 31916 * power (hFic/100, 2) + 35343 *
hFic/100 + 1931;
  BetaFt := (power(tFic, 2) + A * tFic + B) / (power(tFic, 2) + C * tFic +
D);
  BetaFinf := (power(tFicInf, 2) + A * tFicInf + B) / (power(tFicInf, 2) +
C * tFicInf + D);
  BetaD := (tFicInf - tFic + 20) / (tFicInf - tFic + 70);
  //5-Cálculo do coeficiente de fluência
  result := fiA + fiF * (BetaFinf - BetaFt) + fiD * BetaD;
end;

//Perda de protensão: Fluência_Inf
function PerdaFluInf(Npi, Nps, A, ei, es, eic, I, Ic, psi2, AlfaP:
extended; Momentos, Fif: array of extended): extended;
var
  Mp, SigCG: extended;
  A1, B, C: extended;
begin
  Mp := Npi * ei - Nps * es;
  A1 := ((Npi + Nps) / A + (Mp - Momentos[0]) * ei / I) * FiF[1];
  B := (Momentos[1] * FiF[2] + Momentos[2] * FiF[3]) * ei / I;
  C := (Momentos[3] * Fif[4] + Momentos[4] * FiF[5] + Momentos[5] * FiF[6]
* psi2) * eic / Ic;
  result := AlfaP * (A1 - B - C) / 1000;
end;

//Perda de protensão: Fluência_Sup
function PerdaFluSup(Npi, Nps, A, ei, es, esc, I, Ic, psi2, AlfaP:
extended; Momentos, Fif: array of extended): extended;
var
  Mp, SigCG: extended;
  A1, B, C: extended;
begin
  Mp := Npi * ei - Nps * es;
  A1 := ((Npi + Nps) / A + (- Mp + Momentos[0]) * es / I) * FiF[1];
  B := (Momentos[1] * FiF[2] + Momentos[2] * FiF[3]) * es / I;
  C := (Momentos[3] * Fif[4] + Momentos[4] * FiF[5] + Momentos[5] * FiF[6]
* psi2) * esc / Ic;
  result := AlfaP * (A1 + B + C) / 1000;
end;

//Media ponderada dos coeficientes de fluencia
function MedPon(A1, A2, Fi1, Fi2: extended): extended;
begin
  result := (A1 * Fi1 + A2 * Fi2) / (A1 + A2);
```

```
end;

//Perda por retração
function Retrac(U, Ac, UAr, slump, T, tDelta, tDeltaInf, Ep: extended):
extended;
var
  i: integer;
  gama, hFic, eps1s, eps2s, epscs, BetaSt0, BetaSt, tFic, tFicInf, AlfaC:
extended;
  A, B, C, D, E: extended;
begin
  //Cálculo do eps1s e eps2s
  gama := 1 + EXP(-7.8 + 0.1 * U);
  hFic := gama * 2 * Ac / UAr;
  eps1s := -8.09 + (U / 15) - (power(U,2) / 2284) - (power(U,3) / 133765) +
(power(U,4) / 7608150);
  if slump <= 4 then
    eps1s := eps1s * 0.75
  else if slump >= 10 then
    eps1s := eps1s * 1.25;
  eps2s := (33 + 2 * hFic) / (20.8 + 3 * hFic);
  epscs := eps1s * eps2s;
  //Cálculo do BetaSt0 e BetaSt
  if hFic < 5 then
    hFic := 5
  else if hFic > 160 then
    hFic := 160;
  A := 40;
  B := 116 * power(hFic/100,3) - 282 * power(hFic/100,2) + 220 * hFic/100 -
4.8;
  C := 2.5 * power(hFic/100,3) - 8.8 * hFic/100 + 40.7;
  D := -75 * power(hFic/100,3) + 585 * power(hFic/100,2) + 496 * hFic/100 -
6.8;
  E := -169 * power(hFic/100,4) + 88 * power(hFic/100,3) + 584 *
power(hFic/100,2) - 39 * hFic/100 + 0.8;
  tFic := 1 * ((T + 10) / 30 * tDelta);
  tFicInf := 1 * ((T + 10) / 30 * tDeltaInf);
  if tFic < 3 then
    tFic := 3;
  if tFicInf >= 10000 then
    tFicInf := 10000;
  BetaSt0 := (power(tFic/100, 3) + A * power(tFic/100, 2) + B * tFic / 100)
/
             (power(tFic/100, 3) + C * power(tFic/100, 2) + D * tFic / 100
+ E);
  BetaSt := (power(tFicInf/100, 3) + A * power(tFicInf/100, 2) + B *
tFicInf/100) /
            (power(tFicInf/100, 3) + C * power(tFicInf/100, 2) + D *
tFicInf/100 + E);
  result := - (epscs * power(10,-4) * (BetaSt - BetaSt0) * Ep);
end;

//Cálculo das perdas de forma progressiva
function Progressiva(PFlu, PRet, SigRel, PsiRel, FiFlu, ep, Ac, I, Ap,
AlfaP: extended): extended;
var
  Xt, Xp, Xc, Eta, RoP: extended;
begin
  Xt := - Ln(1 - PsiRel);
  Xp := 1 + Xt;
  Xc := 1 + 0.5 * FiFlu;
```

```
  Eta := 1 + power(ep,2) * Ac / I;
  RoP := Ap / (Ac * 10000);
  result := (PFlu + PRet + SigRel * Xt) / (Xp + Xc * AlfaP * Eta * RoP);
end;

//Cálculo do momento de fissuração
function MomFiss(fck, Npi, Nps, EpI, EpS, alfa1, alfa2, Ac, W1, W2:
extended): extended;
var
  fctm, Mp: extended;
begin
  fctm := 0.3 * power(fck, 2/3);
  Mp := Npi * EpI - Nps * EpS;
  if W1 = W2 then
    result := (alfa1 * fctm * 1000 + (Npi + Nps) / Ac) * W1 + Mp
  else
    result := alfa2 * fctm * 1000 * W2 + (Npi + Nps) * W1 / Ac + Mp;
end;

//Calcula a flecha imediata
function FlechaImed(k: integer; Lpi, Lps, Ei, Es, p, L, Ec, I: extended;
NpLp: array of extended): extended;
var
  j: integer;
  A, C1: extended;
  MomEst: array [0..3] of extended;
begin
  if k = 1 then begin
    //Momento estático da área do momento de protensão da cordoalha
superior
    if Lps <= L / 2 then begin
      A := Lps * NpLp[5] * Es / 2;
      C1 := 2 / 3 * Lps;
      MomEst[0] := A * C1;
      A := (L / 2 - Lps) * NpLp[5] * Es;
      C1 := (L / 2 - Lps) / 2 + Lps;
      MomEst[0] := MomEst[0] + A * C1;
    end
    else begin
      A := power(L / 2,2) / Lps * NpLp[5] * Es / 2;
      C1 := 2 / 3 * L / 2;
      MomEst[0] := MomEst[0] + A * C1;
    end;
    //Momento estático da área do momento de protensão da cordoalha
inferior
    MomEst[1] := 0;
    for j := 0 to 4 do begin
      if Lpi <= L / 10 * (5 - j) then begin
        A := Lpi * NpLp[j] * Ei / 2;
        C1 := 2 / 3 * Lpi + L / 10 * j;
        MomEst[1] := MomEst[1] + A * C1;
        A := (L / 2 - Lpi - L / 10 * j) * NpLp[j] * Ei;
        C1 := (L / 2 - Lpi - L / 10 * j) / 2 + Lpi + L / 10 * j;
        MomEst[1] := MomEst[1] + A * C1;
      end
      else begin
        A := power(L / 10 * (5 - j),2) / Lpi * NpLp[j] * Ei / 2;
        C1 := 2 / 3 * L / 10 * (5 - j) + L / 2 - L / 10 * (5 - j);
        MomEst[1] := MomEst[1] + A * C1;
      end;
    end;
```

```
  result := (+MomEst[0] - MomEst[1]) / (Ec * I ) * 1000;
  end
  else if k = 2 then begin
    //Momento estático da área do momento do carregamento atuante
    A := power(L,3) * p / 24;
    C1 := 5 / 16 * L;
    MomEst[2] := A * C1;
    //Momento estático da área do momento da reação de apoio
    A := power(L,3) * p / 16;
    C1 := L / 3;
    MomEst[3] := A * C1;
    result := (-MomEst[2] + MomEst[3]) / (Ec * I ) * 1000;
  end
end;

//Cálculo da flecha diferida
function FlechaDif(FImed, FiFlu, coefE: extended): extended;
begin
  result := (1 + FiFlu) * FImed * coefE;
end;

//Cálculo da inércia de Branson
function Branson(M, Mr, I, IxII: extended): extended;
begin
  result := power(Mr / M, 3) * I + (1 - power(Mr / M, 3)) * IxII;
end;

//Cálculo da armadura transversal - Modelo I
function TransversalI(Npi, Nps, gF, gS, Ei, Es, fck, gamaC, bw, d, A, Wif,
Vsd, Msd, Asw, fywk, alfa, L, p: extended): tEstribo;
var
  Mp, VRd21, Vc0, M0, Vr, Fctm, Fctd: extended;
  Aswmin, sMax, RoMin, s, sRoMin, Ro, ARad: extended;
  Estribo: tEstribo;
begin
  //Cálculo do momento de protensão (Mp)
  Mp := Npi * Ei - Nps * Es;
  Fctm := 0.3 * power(fck,2/3);
  Fctd := 0.7 * Fctm / GamaC;
  ARad := pi * alfa / 180;
  //Verificação do esmagamento da biela de concreto
  VRd21 := 0.27 * (1 - fck / 250) * fck * 1000 / gamaC * bw * d;
  if VRd21 >= Vsd then
    Estribo.Biela := 'OK!'
  else
    Estribo.Biela := 'Não OK!';
  //Cálculo do mecanismo resistente e da parcela resistida pela armadura
  Vc0 := 0.6 * Fctd * 1000 * bw * d;
  M0 := Wif * (gF * (Npi + Nps) / A + Mp / Wif);
  Estribo.Vc := Vc0 * (1 + M0 / Msd);
  if Estribo.Vc > 2 * Vc0 then
    Estribo.Vc := 2 * Vc0;
  Estribo.Vsw := Vsd - Estribo.Vc;
  if Estribo.Vsw > 0 then begin
    s := 2 * Asw / Estribo.Vsw * 0.9 * d * fywk / gS * 100;
    Ro := 2 * Asw / (bw * 100 * s * sin(ARad));
  end
  else
    Ro := 0;
  //Cálculo da taxa da armadura mínima
  if Vsd / VRd21 <= 0.67 then begin
```

```
    sMax := 0.6 * d * 100;
    if sMax > 30 then
      sMax := 30;
  end
  else begin
    sMax := 0.3 * d * 100;
    if sMax > 20 then
      sMax := 20;
  end;
  RoMin := 2 * Asw / (bw * 100 * sMax * sin(ARad));
  //Valor do espaçamento máximo para armadura mínima e sua região
  Estribo.sMax := sMax;
  sRoMin := 2 * Asw / (bw * 100 * 0.2 * Fctm / (fywk * 10));
  Estribo.sMax := min(Estribo.sMax, sRoMin);
  Vr := 644 * bw * d * (2 * Asw / (bw*100 * Estribo.sMax * sin(ARad)) *
fywk / gS * 10 * sin(ARad) * (sin(ARad) + cos(ARad)) + 0.1 *
power(fck,2/3));
  Estribo.xEstribo := L / 2 - Vr / p;
  //Compara taxa resistida com taxa mínima
  if RoMin > Ro then
    Estribo.TipoDim := 'Espaçamento máximo!'
  else
    Estribo.TipoDim := 'Resistido pelo estribo!';
  Ro := max(Ro, RoMin);
  if Ro < 0.2 * Fctm / (10 * fywk) then begin
    Estribo.Ro:= 0.2 * Fctm / (10 * fywk);
    Estribo.TipoDim := 'Taxa mínima!';
    Estribo.s := 2 * Asw / (bw * 100 * Estribo.Ro);
    Result := Estribo;
  end
  else begin
    Estribo.Ro := Ro;
    Estribo.s := 2 * Asw / (bw * 100 * Estribo.Ro);
    Result := Estribo;
  end;
end;

//Cálculo da armadura transversal - Modelo II
function TransversalII(Npi, Nps, gF, gS, Ei, Es, fck, gamaC, bw, d, A, Wif,
Vsd, Msd, Asw, fywk, alfa, teta, L, p: extended): tEstribo;
var
  Mp, VRd22, Vc0, Vc1, M0, Vr, Fctm, Fctd: extended;
  Aswmin, sMax, RoMin, s, sRoMin, Ro, TRad, ARad: extended;
  Estribo: tEstribo;
begin
  //Cálculo do momento de protensão (Mp)
  Mp := Npi * Ei - Nps * Es;
  Fctm := 0.3 * power(fck,2/3);
  Fctd := 0.7 * Fctm / GamaC;
  TRad := pi * teta / 180;
  ARad := pi * alfa / 180;
  //Verificação do esmagamento da biela de concreto
  VRd22 := 0.54 * (1 - fck / 250) * fck * 1000 / gamaC * bw * d *
power(sin(TRad),2) * (cotan(ARad) + cotan(TRad));
    if VRd22 >= Vsd then
    Estribo.Biela := 'OK!'
  else
    Estribo.Biela := 'Não OK!';
  //Cálculo do mecanismo resistente e da parcela resistida pela armadura
  Vc0 := 0.6 * Fctd * 1000 * bw * d;
  M0 := Wif * (gF * (Npi + Nps) / A + Mp / Wif);
```

```
  Vc1 := - Vc0 * (Vsd - Vc0) / (VRd22 - Vc0) + Vc0;
  Estribo.Vc := Vc1 * (1 + M0 / Msd);
  if Estribo.Vc > 2 * Vc0 then
    Estribo.Vc := 2 * Vc0;
  Estribo.Vsw := Vsd - Estribo.Vc;
  if Estribo.Vsw > 0 then begin
    s := 2 * Asw / Estribo.Vsw * 0.9 * d * fywk / gS * 100 * cotan(TRad);
    Ro := 2 * Asw / (bw * 100 * s * sin(ARad));
  end
  else
    Ro := 0;
  //Cálculo da taxa da armadura mínima
  if Vsd / VRd22 <= 0.67 then begin
    sMax := 0.6 * d * 100;
    if sMax > 30 then
      sMax := 30;
  end
  else begin
    sMax := 0.3 * d * 100;
    if sMax > 20 then
      sMax := 20;
  end;
  RoMin := 2 * Asw / (bw * 100 * sMax * sin(ARad));
  //Valor do espaçamento máximo para armadura mínima e sua região
  Estribo.sMax := sMax;
  sRoMin := 2 * Asw / (bw * 100 * 0.2 * Fctm / (fywk * 10));
  Estribo.sMax := min(Estribo.sMax, sRoMin);
  Vr := 644 * bw * d * (2 * Asw / (bw*100 * Estribo.sMax * sin(ARad)) *
fywk / gS * 10 * sin(ARad) * (sin(ARad) + cos(ARad)) + 0.1 *
power(fck,2/3));
  Estribo.xEstribo := L / 2 - Vr / p;
  //Compara taxa resistida com taxa mínima
  if RoMin > Ro then
    Estribo.TipoDim := 'Espaçamento máximo!'
  else
    Estribo.TipoDim := 'Resistido pelo estribo!';
  Ro := max(Ro, RoMin);
  if Ro < 0.2 * Fctm / (10 * fywk) then begin
    Estribo.Ro := 0.2 * Fctm / (10 * fywk);
    Estribo.TipoDim := 'Taxa mínima!';
    Estribo.s := 2 * Asw / (bw * 100 * Estribo.Ro);
    Result := Estribo;
  end
  else begin
    Estribo.Ro := Ro;
    Estribo.s :=  2 * Asw / (bw * 100 * Estribo.Ro);
    Result := Estribo;
  end;
end;

function InterfaceTrans(Fhd, L, Fck, gC, Fyk, gS, Asw: extended; aDim:
array of tConcreto; Beta: array of tPoints): tPoints;
var
  i, sMin, sMax: integer;
  TauSd, TauUd, b, sReal, Ro, BetaS, BetaC: extended;
  sInter: tPoints;
begin
  b := aDim[2].X; TauUd := 0;
  for i := 0 to 2 do
    if aDim[i].X <> 0 then
      b := min(b, aDim[i].X);
```

```
  TauSd := Fhd / (L/2 * b);
  sMax := 30; sMin := 1;
  while TauSd >= TauUd do begin
    if sMax = 0 then begin
        ShowMessage('Não há solução para o problema! Altere as
características necessárias!');
        Exit;
    end;
    Ro := 2 * Asw / (sMax * b);
    //Beta[0] são as taxas extremas, Beta[1] são os BetaS e Beta[2] os
BetaC. X menor Y maior
    BetaS := (Beta[1].Y - Beta[1].X) / (Beta[0].Y - Beta[0].X) * (Ro -
Beta[0].X) + Beta[1].X;
    BetaC := (Beta[2].Y - Beta[2].X) / (Beta[0].Y - Beta[0].X) * (Ro -
Beta[0].X) + Beta[2].X;
    if Ro <= Beta[0].X then begin
      BetaS := Beta[1].X;
      BetaC := Beta[2].X;
    end
    else if Ro >= Beta[0].Y then begin
      BetaS := Beta[1].Y;
      BetaC := Beta[2].Y
    end;
    TauUd := BetaS * Ro / 100 * Fyk / gS * 10000 + BetaC * 0.21 *
power(Fck, 2/3) / gC * 1000;
    if TauSd <= TauUd then
      sInter.X := sMax;
    sMax := sMax - 1;
    end;

  TauUd := 0.25 * Fck / 1.4 * 1000 + 1;
  while TauUd > 0.25 * Fck / 1.4 * 1000 do begin
    if sMin = 31 then begin
        ShowMessage('Não há solução para o problema! Altere as
características necessárias!');
        Exit;
    end;
    Ro := 2 * Asw / (sMin * b);
    BetaS := (Beta[1].Y - Beta[1].X) / (Beta[0].Y - Beta[0].X) * (Ro -
Beta[0].X) + Beta[1].X;
    BetaC := (Beta[2].Y - Beta[2].X) / (Beta[0].Y - Beta[0].X) * (Ro -
Beta[0].X) + Beta[2].X;
    if Ro <= Beta[0].X then begin
      BetaS := Beta[1].X;
      BetaC := Beta[2].X;
    end
    else if Ro >= Beta[0].Y then begin
      BetaS := Beta[1].Y;
      BetaC := Beta[2].Y
    end;
    TauUd := BetaS * Ro / 100 * Fyk / gS * 10000 + BetaC * 0.21 *
power(Fck, 2/3) / gC * 1000;
    if TauUd < 0.25 * Fck / 1.4 * 1000 then
      sInter.Y := sMin;
    sMin := sMin + 1;
    end;
  if sInter.Y > sInter.X then begin
    ShowMessage('Não há solução para o problema! Altere as características
necessárias!');
    Exit;
  end;
```

```
  result := sInter;
end;
//Cálculo da inércia no estádio II (tPoints.X := xII / tPoints.Y := IxII)
function InertiaII(dI, dS, ApI, ApS, AsI, AsS, Ep, Es: extended; Sec: array
of tConcreto): tPoints;
var
  i: integer;
  xII, Fc, Fs, I2c, I2s, Ht, Eutil, Autil, EcsM: extended;
  IxII: tPoints;
begin
  Sec[0].X := Sec[0].X * 100; Sec[0].Y := Sec[0].Y * 100; Sec[0].Ecs :=
Sec[0].Ecs / 10;
  Sec[1].X := Sec[1].X * 100; Sec[1].Y := Sec[1].Y * 100; Sec[1].Ecs :=
Sec[1].Ecs / 10;
  Sec[2].X := Sec[2].X * 100; Sec[2].Y := Sec[2].Y * 100; Sec[2].Ecs :=
Sec[2].Ecs / 10;
  Sec[3].X := Sec[3].X * 100; Sec[3].Y := Sec[3].Y * 100; Sec[3].Ecs :=
Sec[3].Ecs / 10;
  Ep := Ep / 10;
  Es := Es / 10;
  xII := 0; Fc := 1000; Fs := -1000;
  while abs(Fc-Fs)>1000 do begin
    xII := xII + 0.001;
    if xII <= Sec[0].Y then begin
      Fc := Sec[0].X*xII*Sec[0].Ecs/2;
      Fs := (ApI*Ep+AsI*Es)*(dI/xII-1)+(ApS*Ep+AsS*Es)*(dS/xII-1);
    end
    else if (xII > Sec[0].Y) and (xII <= Sec[0].Y+Sec[1].Y) then begin
      Fc := Sec[0].X*Sec[0].Y*Sec[0].Ecs/2*(2-Sec[0].Y/xII)+Sec[1].X*(xII-
Sec[0].Y)*Sec[1].Ecs/2*(1-Sec[0].Y/xII);
      Fs := (ApI*Ep+AsI*Es)*(dI/xII-1)+(ApS*Ep+AsS*Es)*(dS/xII-1);
    end
    else if (xII > Sec[1].Y) and (xII <= Sec[0].Y+Sec[1].Y+Sec[2].Y) then
begin
      Fc := Sec[0].X*Sec[0].Y*Sec[0].Ecs/2*(2-
Sec[0].Y/xII)+Sec[1].X*Sec[1].Y*Sec[1].Ecs/2*(2-2*Sec[0].Y/xII-
Sec[1].Y/xII)
            +Sec[2].X*(xII-Sec[0].Y-Sec[1].Y)*Sec[2].Ecs/2*(1-
(Sec[0].Y+Sec[1].Y)/xII);
      Fs := (ApI*Ep+AsI*Es)*(dI/xII-1)-(ApS*Ep+AsS*Es)*(dS/xII-1);
    end
    else if (xII > Sec[2].Y) and (xII <=
Sec[0].Y+Sec[1].Y+Sec[2].Y+Sec[3].Y) then begin
      Fc := Sec[0].X*Sec[0].Y*Sec[0].Ecs/2*(2-
Sec[0].Y/xII)+Sec[1].X*Sec[1].Y*Sec[1].Ecs/2*(2-2*Sec[0].Y/xII-
Sec[1].Y/xII)
            +Sec[2].X*Sec[2].Y*Sec[2].Ecs/2*(2-2*Sec[0].Y/xII-
2*Sec[1].Y/xII-Sec[2].Y/xII)
            +Sec[3].X*(xII-Sec[0].Y-Sec[1].Y-Sec[2].Y)*Sec[3].Ecs/2*(1-
(Sec[0].Y+Sec[1].Y+Sec[2].Y)/xII);
      Fs := (ApI*Ep+AsI*Es)*(dI/xII-1)-(ApS*Ep+AsS*Es)*(dS/xII-1);
    end;
    if xII > dI then begin
      ShowMessage('O programa não encontrou uma LN que seja solução para o
problema!');
      Exit;
    end;
  end;
  i := 0; Ht := 0; I2c := 0; Autil := 0; Eutil := 0;
  while xII > Sec[i].Y + Ht do begin
```

```
    I2c := I2c + Sec[i].X * power(Sec[i].Y,3) / 12 + Sec[i].X * Sec[i].Y *
power(xII - (Ht + Sec[i].Y / 2), 2);
    Ht := Ht + Sec[i].Y;
    Eutil := Eutil + (Sec[i].X * Sec[i].Y) * Sec[i].Ecs;
    Autil := Autil + Sec[i].X * Sec[i].Y;
    i := i + 1;
  end;
  Eutil := Eutil + (Sec[i].X * (xII - Ht)) * Sec[i].Ecs;
  Autil := Autil + (Sec[i].X * (xII - Ht));
  EcsM := Eutil / Autil;
  I2c := I2c + Sec[i].X * power(xII - Ht, 3) / 12 + Sec[i].X * (xII - Ht) *
power((xII-Ht)/2,2);
  if xII < dS then
    I2s := power(xII-dI,2)*(ApI*Ep/EcsM + AsI*Es/EcsM) + power(xII-
dS,2)*(ApS*Ep/EcsM + AsS*Es/EcsM)
  else
    I2s := power(xII-dI,2)*(ApI*Ep/EcsM + AsI*Es/EcsM) + power(xII-
dS,2)*(ApS*(Ep/EcsM-1) + AsS*(Es/EcsM-1));
  IxII.X := xII;
  IxII.Y := (I2c + I2s) / power(10,8);
  result := IxII;
end;
//Cálculo da abertura de fissura
function AbertFiss(Npi, Nps, Ac, epi, I1, I2, Ycg, EtaP, EtaS, Ep, Es, Ec,
Base, fck, Mfreq: extended; tLista_Arm: TListBox): tFissuracao;
var
  i, j, k: integer;
  Esp, EspC, E, Eta: extended;
  M0, yMax, Fctm: extended;
  tArm: tDados;
  aFissura: array of tFissura;
begin
  yMax := 0; j := 0;
  M0 := ((Npi + Nps) / Ac + (Npi + Nps) * power(epi,2) / I1) * I1 / epi;
  Fctm := 0.3 * power(fck, 2/3);
  for i := 0 to tLista_Arm.Count - 1 do begin
    tArm := tDados(tLista_Arm.Items.Objects[i]);
    if tArm.vPBarra = 1 then
      yMax := max(yMax, tArm.vYa);
  end;
  for i := 0 to tLista_Arm.Count - 1 do begin
    tArm := tDados(tLista_Arm.Items.Objects[i]);
    EspC := 0;
    if tArm.vPBarra = 1 then begin
      if tArm.vNBarra > 1 then
        Esp := (tArm.vXb - tArm. vXa) / (tArm.vNBarra - 1);
      for k := 0 to tArm.vNBarra - 1 do begin
        j := j + 1;
        SetLength(aFissura, j);
        aFissura[j-1].X := tArm.vXa + EspC;
        EspC := EspC + Esp;
        aFissura[j-1].Y := tArm.vYa;
        aFissura[j-1].Aco := tArm.vTAco_index;
        aFissura[j-1].Fi := StrToFloat(tArm.vDBarra) / 10;
        aFissura[j-1].Ap := tArm.vABarra;
        //Guarda as dimensões
        if k = 0 then
          aFissura[j-1].a := tArm.vXa
        else
          aFissura[j-1].a := Esp / 2;
        if tArm.vNBarra = 1 then
```

```
          aFissura[j-1].b := Base - tArm.vXa
        else if k = tArm.vNBarra-1 then
          aFissura[j-1].b := Base - tArm.vXb
        else
           aFissura[j-1].b := Esp / 2;
        aFissura[j-1].c := tArm.vYa;
        aFissura[j-1].d := yMax - tArm.vYa;
        if yMax - tArm.vYa = 0 then
          aFissura[j-1].d := 7.5 * aFissura[j-1].Fi;
        //Verificação da dimensão máxima
        if aFissura[j-1].a > 7.5 * aFissura[j-1].Fi then
          aFissura[j-1].a := 7.5 * aFissura[j-1].Fi;
        if aFissura[j-1].b > 7.5 * aFissura[j-1].Fi then
          aFissura[j-1].b := 7.5 * aFissura[j-1].Fi;
        if aFissura[j-1].c > 7.5 * aFissura[j-1].Fi then
          aFissura[j-1].c := 7.5 * aFissura[j-1].Fi;
        if aFissura[j-1].d > 7.5 * aFissura[j-1].Fi then
          aFissura[j-1].d := 7.5 * aFissura[j-1].Fi;
        if tArm.vTBarra = 1 then begin
          E := Ep;
          Eta := Etap;
        end
        else if tArm.vTBarra = 2 then begin
          E := Es;
          Eta := Etas;
        end;
        aFissura[j-1].Ro := aFissura[j-1].Ap / ((aFissura[j-1].a +
aFissura[j-1].b) * (aFissura[j-1].c + aFissura[j-1].d));
        aFissura[j-1].SigAco := (Mfreq - M0) / I2 * (Ycg - aFissura[j-
1].Y/100) * E / Ec / 1000;
        aFissura[j-1].Wk1 := aFissura[j-1].Fi * 10 / (12.5 * Eta) *
aFissura[j-1].SigAco / E * 3 * aFissura[j-1].SigAco / Fctm;
        aFissura[j-1].Wk2 := aFissura[j-1].Fi * 10 / (12.5 * Eta) *
aFissura[j-1].SigAco / E * (4 / aFissura[j-1].Ro + 45);
      end;
    end;
  end;
  result := aFissura;
end;
//Cálculo da abertura de fissura para parâmetros estabelecidos
function VerificFiss(ind: integer; EtaP, EtaS, Ep, Es: extended; Fissura:
tFissura): tFissura;
var
  E, Eta: extended;
  Fiss: tFissura;
begin
  if Fissura.Aco = 1 then begin
    E := Ep;
    Eta := EtaP;
  end
  else if Fissura.Aco = 2 then begin
    E := Es;
    E := EtaS;
  end;
  Fissura.Ro := Fissura.Ap / ((Fissura.a + Fissura.b) * (Fissura.c +
Fissura.d));
  Fissura.Wk2 := Fissura.Fi * 10 / (12.5 * Eta) * Fissura.SigAco / E * (4 /
Fissura.Ro + 45);
  result := Fissura;
end;
end.
```