SERVIÇO DE PÓS-GRADUAÇÃO DO ICMC-USP

Data de Depósito:

Assinatura: _____

Felipe Hernandez Bisca

Multivariate conditional density estimation with copulas

Master dissertation submitted to the Institute of Mathematics and Computer Sciences – ICMC-USP and to the Department of Statistics – DEs-UFSCar, in partial fulfillment of the requirements for the degree of the Master Interagency Program Graduate in Statistics.

Concentration Area: Statistics

Advisor: Prof. Dr. Rafael Izbicki

USP – São Carlos
August 2021

Ficha catalográfica elaborada pela Biblioteca Prof. Achille Bassi
e Seção Técnica de Informática, ICMC/USP,
com os dados inseridos pelo(a) autor(a)

B621m

Bisca, Felipe
Multivariate density estimation with copulas / Felipe Bisca; orientador Rafael Izbicki. -- São Carlos, 2021.
49 p.

Tese (Doutorado - Programa Interinstitucional de Pós-graduação em Estatística) -- Instituto de Ciências Matemáticas e de Computação, Universidade de São Paulo, 2021.

1. Conditional Density Estimation. 2. Copula. 3. FlexCode. I. Izbicki, Rafael, orient. II. Título.

Bibliotecários responsáveis pela estrutura de catalogação da publicação de acordo com a AACR2:
Gláucia Maria Saia Cristianini - CRB - 8/4938
Juliana de Souza Moraes - CRB - 8/6176

Felipe Hernandez Bisca

Estimação de densidade condicional multivariada com cópulas

Dissertação apresentada ao Instituto de Ciências Matemáticas e de Computação – ICMC-USP e ao Departamento de Estatística – DEs-UFSCar, como parte dos requisitos para obtenção do título de Mestre em Estatística – Programa Interinstitucional de Pós-Graduação em Estatística.

Área de Concentração: Estatística

Orientador: Prof. Dr. Rafael Izbicki

USP – São Carlos
Agosto de 2021

UNIVERSIDADE FEDERAL DE SÃO CARLOS

Centro de Ciências Exatas e de Tecnologia
Programa Interinstitucional de Pós-Graduação em Estatística

Folha de Aprovação

Defesa de Dissertação de Mestrado do candidato Felipe Hernandez Bisca, realizada em 29/09/2021.

Comissão Julgadora:

Prof. Dr. Rafael Izbicki (UFSCar)

Prof. Dr. Victor Fossaluza (USP)

Prof. Dr. Anderson Luiz Ara Souza (UFBA)

O presente trabalho foi realizado com apoio da Coordenação de Aperfeiçoamento de Pessoal de Nível Superior - Brasil (CAPES) - Código de Financiamento 001.

O Relatório de Defesa assinado pelos membros da Comissão Julgadora encontra-se arquivado junto ao Programa Interinstitucional de Pós-Graduação em Estatística.

“ If I have seen further
it is by standing on the shoulders of Giants.”
(Isaac Newton)

RESUMO

BISCA, F. H. **Estimação de densidade condicional multivariada com cópulas**. 2021. 49 p. Dissertação (Mestrado em Estatística – Programa Interinstitucional de Pós-Graduação em Estatística) – Instituto de Ciências Matemáticas e de Computação, Universidade de São Paulo, São Carlos – SP, 2021.

A maioria dos modelos de regressão de aprendizado de máquina produz apenas estimativas pontuais para a resposta de uma nova observação. No entanto, ao lidar com distribuições multimodais ou assimétricas, a estimativa pontual não é suficiente para resumir toda a incerteza sobre a resposta. Uma solução para este caso é estimar toda a função de densidade condicional da resposta, condicional às características, o que é mais informativo. Por exemplo, essa densidade pode ser usada para calcular regiões de probabilidade em vez de estimativas pontuais. As densidades condicionais tornam-se especialmente úteis ao modelar respostas multivariadas, o que geralmente ocorre em campos como a cosmologia. A maioria dos estimadores de densidade condicional conhecidos são lentos computacionalmente ou não generalizam respostas multivariada. Para minimizar esses problemas, nosso método estima densidades multivariadas usando cópula para agregar estimativas de densidades condicionais univariadas fornecidas pelo FlexCode, que foi desenvolvido recentemente. Mostramos que esta solução leva a melhores resultados quando comparada com outras técnicas do estado da arte.

Palavras-chave: Estimação de densidade condicional, Cópula, FlexCode.

ABSTRACT

BISCA, F. H. **Multivariate conditional density estimation with copulas**. 2021. 49 p. Dissertação (Mestrado em Estatística – Programa Interinstitucional de Pós-Graduação em Estatística) – Instituto de Ciências Matemáticas e de Computação, Universidade de São Paulo, São Carlos – SP, 2021.

Most machine learning regression models only yield single point estimations for the label of a new observation. However, when dealing with multi-modal or asymmetric distributions, a single point estimate is not enough to summarize the full uncertainty over such label. One solution for this case is to estimate the full conditional density function of the label given the features, which is more informative. For instance, this density can be used to compute probability regions rather than single point estimates. Conditional densities become especially useful when modelling multivariate responses, which is often the case in fields such as cosmology. Most well known conditional density estimators are too slow to be computed or do not generalize to multivariate-response settings. To minimize such problems, our method estimates multivariate densities using copula to aggregate estimates of univariate conditional densities given by the recent-developed FlexCode. We show that this solution leads to improved results when compared to other state-of-the-art techniques.

Keywords: Conditional Density Estimation, Copula, FlexCode.

LIST OF FIGURES

LIST OF TABLES

CONTENTS

CHAPTER

1

INTRODUCTION

Data does not always allow us to build predictive models accurately as we desire. Estimating Z for a given X could be very challenging when dealing with asymmetric, multimodal, or heteroscedastic noise. In this case, regression models typically fail in making good predictions or estimate confidence intervals. Previous studies have shown that probability density function estimation PDF is a powerful tool to minimize systematic errors. In the cosmology field, PDF estimation is already being used to quickly extract physical properties and redshifts of galaxies mucesh2021machine. Others studies have shown great results in applications of time series models (KALDA; SIDDIQUI, 2013) and approximate Bayesian models (FAN; NOTT, 2013).

Most related work uses non-parametric kernel density to estimate conditional density (ROSENBLATT, 1969). The unconditional densities are estimated with kernel density and the conditional density is obtain by Bayes Theorem with $f(z|x) = f(z,x)/f(x)$. However, tuning all bandwidths required to perform such estimation is computationally heavy when dealing with high dimension space, even more, when dealing with multivariate response (BASHTANNYK; HYNDMAN, 2001). Some approaches have been proposed to reduce computation time to tuning bandwidths, but they often lead to a decrease in the performance of the estimator (SHARMA; LALL; TARBOTON, 1998; SHEATHER; JONES, 1991; COMANICIU, 2003). Other methods have also been proposed for conditional density estimation. For instance, Pospisil and Lee (2018) uses random forest for this task, while Izbicki and Lee (2017a) converts the problem of estimating a conditional density estimation problem to a simpler high-dimensional regression problem, a method named FlexCode.

FlexCode is a non-parametric approach that can deal easily with high dimension features and can be extended to deal with multivariate responses. However, because it is based on the tensor product on the response space, the computational required to fit it grows exponentially in the dimension of the response. In this work, our goal is to extend FlexCode so that it can deal with multivariate responses without incurring the computational burden of the standard solution. In order to achieve this, we propose to model the dependence structure of the response

variables using copula. In this way, the increase in computational time is linear with the response dimension. The copula is a model dependence structure that uses single densities distributions and leads to a join distribution. Copula received recent attention in many fields mostly in finance (JONDEAU; ROCKINGER, 2006; ROMANO, 2002). aghakouchak2010copula show copula application to multisensor precipitation estimates and Zhang and Dukic (2013) for predicting loss payments.

In this work, we present a review of standard FlexCode for conditional density estimation for univariate response and review the main idea of copulas, featuring some copula functions that are most commonly used as Gaussian and Archimedean class of copulas in chapter 2. Chapter 3 introduces our approach to estimate copula functions that have different degrees of complexity. The first one uses a parametric copula function c_θ, with θ estimated via maximum likelihood. The second one uses a local estimation of θ via maximization of a weighted likelihood function, which is a semi-parametric approach. The third uses a fully non-parametric generalization of copula with kernel density. We use copula estimation within FlexCode to create our method of estimation multivariate conditional densities, which we call Multivariate FlexCode with Copula or MFC. Chapters 4 and 5 show promising results of MFC fitting multivariate conditional densities in simulations and real data examples comparing to other two methods, Random Forest Conditional Densities, and Kernel Density Estimation. Chapter 6 concludes this work by discussing the results and ideas for future work.

CHAPTER

2

REVIEW

2.1 FlexCode

We start by reviewing the standard FlexCode for a univariate response $z \in \mathbb{R}$.

Assume we observe a i.i.d data $(\mathbf{X}_1, Z_1), ..., (\mathbf{X}_n, Z_n)$, where $\mathbf{x} \in \mathbb{R}^d$ and $Z \in \mathbb{R}$. Flexcode estimates the full density $f(z|\mathbf{x})$ by expanding it (as a function of z) into a an orthonormal basis $(\phi_i)_{i\in\mathbb{N}}$ for $L_2(R)$. Each coefficient can be estimated via a regression method. Thus, the method converts conditional density estimation problem to a simpler high-dimensional regression problem. More precisely,

$$f(z|\mathbf{x}) = \sum_{i\in\mathbb{N}} \beta_i(\mathbf{x})\phi_i(z),$$

where $(\phi_i)i \in N$ is the Fourier basis:

$$\phi_1(z) = 1;\ \phi_{2i+1}(z) = \sqrt{2}\sin(2\pi iz), i \in \mathbb{N};\ \phi_{2i}(z) = \sqrt{2}\cos(2\pi iz), i \in \mathbb{N}.$$

One can show that

$$\hat{\beta}_i(\mathbf{x}) = E[\phi_i(Z)|\mathbf{x}].$$

The FlexCode estimate of $f(z|\mathbf{x})$ is define by:

$$\hat{f}(z|\mathbf{x}) = \sum_{i=1}^{I} \hat{\beta}_i(\mathbf{x})\phi_i(z), \tag{2.1}$$

where $\hat{\beta}_i(\mathbf{x})$ are estimates obtained by regressing $\phi_i(Z)$ in $\mathbf{x}$. They model how the density varies in the covariate space. Note that any regression model can be used to estimate $\beta_i(\mathbf{x})$. So the solution to estimate conditional densities comes down to estimate I regression functions. The cutoff I is a tuning parameter that controls the bias-variance tradeoff in the final density estimate. In practice, I is estimated by splitting data into training and validation, as explained in section 2.4.

2.2 FlexCode for multivariate response

FlexCode can be easily extend to multivariate response by tensor products. For instance, if $\mathbf{Z} \in \mathbb{R}^2$, consider the basis

$$\{\phi_{i,j}(\mathbf{z}) = \phi_i(z_1)\phi_j(z_2) : i, j \in \mathbb{N}\},$$

where $\mathbf{z} = (z_1, z_2)$, and $\{\phi_i(z_1)\}_i$ and $\{\phi_j(z_2)\}_j$ are the bases for function in $L_2(R)$. Then, let

$$f(\mathbf{z}|\mathbf{x}) = \sum_{i,j \in \mathbb{N}} \beta_{i,j}(\mathbf{x})\phi_{i,j}(\mathbf{z}),$$

where the coefficients

$$\beta_{i,j}(\mathbf{x}) = E[\phi_{i,j}(z)|\mathbf{x}].$$

Note that different from the univariate case, the solution to estimate $f(.|x)$ becomes the solution of $I \times J$ regression models, where I and J are the cutoffs of the index i and j respectively. Thus, for each new dimension of $\mathbf{Z}$, the complexity to estimate growths exponentially.

2.3 Loss Function

A loss function is used to measure how good a model does in terms of being able to estimated the expected density. It is not only useful to tune parameters, but also for comparing performance models. An usual choice for loss function is the mean log-likelihood, for a testing sample $(\tilde{\mathbf{X}}_1, \tilde{Z}_1), \ldots, (\tilde{\mathbf{X}}_m, \tilde{Z}_m)$,

$$L(\widehat{f}, f) = \int \log(\widehat{f}(z|\mathbf{x}))dP(\mathbf{x})dz$$

and the estimate,

$$\widehat{L}(\widehat{f}, f) = \sum_i^n \log\left(\widehat{f}(z_i|\mathbf{x})\right).$$

However, this loss is very sensitive to regions with low density. Hall (1987) shows that maximize the log-likelihood will not lead to better goodness-of-fit. Thus, instead, we use the squared loss:

$$L(\widehat{f}, f) = \int\int \left(\widehat{f}(\mathbf{z}|\mathbf{x}) - f(\mathbf{z}|\mathbf{x})\right)^2 dP(\mathbf{x})d\mathbf{z}$$

$$= \int\int \widehat{f}^2(\mathbf{z}|\mathbf{x})dP(\mathbf{x})d\mathbf{z} - 2\int\int \widehat{f}(\mathbf{z}|\mathbf{x})\widehat{f}(\mathbf{z},\mathbf{x})d\mathbf{x}d\mathbf{z} + C,$$

where C is constant and do not depend on estimator. This loss can be estimated up to the constant C by using a testing sample $(\tilde{\mathbf{X}}_1, \tilde{Z}_1), \ldots, (\tilde{\mathbf{X}}_m, \tilde{Z}_m)$ via

$$\widehat{L}(\widehat{f}, f) = \frac{1}{m}\sum_{i=1}^m \left(\int \widehat{f}(z|\tilde{\mathbf{x}})^2 dz - 2\widehat{f}(\tilde{z}_i|\tilde{\mathbf{x}}_i)\right). \tag{2.2}$$

In practice, $\int \widehat{f}(z|\tilde{\mathbf{x}})^2 dz$ is calculated via numerical method.

2.4 Tuning Parameters

As describe in algorithm 1, the tuning parameters start by splitting the training data into two parts with the proportion of 70%/30%. The first part is called train data and is used to train the regression parameters with the standard regression loss, the minimum square error. The second part is called validation data and is used to estimated the tune parameter. For a given grid of the tuning parameter $I = 1, ..., I_{max}$, we choose the tuning of parameter I with the smallest estimated loss $\widehat{L}(\widehat{f}, f)$. We use $I_{max} = 30$.

Algorithm 1 – FlexCode

Input Training Data; Validation Data; Maximum cutoff I_0 and regression method
Output Estimator $\widehat{f}(\mathbf{z}|\mathbf{x})$

1: **for all** $i \leq I_0$ **do**
2: Compute $D = (\mathbf{X}_1, \phi_i(z)), ..., (\mathbf{X}_n, \phi_i(z))$
3: Estimate the regression $\beta_i(\mathbf{x}) = E[\phi_i(z)|\mathbf{x}]$
4: Obtain $\widehat{f}_i(z|\mathbf{x})$ ▷ Equation (2.1)
5: **end for**
6: **for all** $i \leq I_0$ **do**
7: Calculate the estimated loss $\widehat{L}(\widehat{f}_I, f)$ on the validation set ▷ Equation (2.2)
8: **end for**
9: Define $\widehat{f}(z|\mathbf{x}) = \arg\min_{\widehat{f}_i} \widehat{L}(\widehat{f}_i, f)$
10: **return** $\widehat{f}(z|\mathbf{x})$

2.5 Copula

Copulas are used to describe the structure dependence between random variables. Technically, a copula is a multivariate cumulative distribution function for which the marginal probability distribution of each variable is uniformly distributed on the interval $[0,1]$.

Sklar's Theorem 2.5.1 shows that any multivariate comulative distribution function can be written as a function of a copula.

Theorem 2.5.1. [Sklar's Theorem] Let $\mathbf{X}$ be a continuous random variable, and let $F_{\mathbf{x}}$ be its multivariate distribution function with marginals F_i, $i = 1, \ldots, d$. Then there exists a unique d-copula $C(\cdot)$ such that for all $\mathbf{x} \in \mathbb{R}^d$,

$$F_{\mathbf{x}}(x_1, \ldots, x_d) = C(F_1(x_1), \ldots, F_d(x_d)).$$

The above formula for the copula function can be rewritten to correspond to this as:

$$C(u_1, \ldots, u_d) = F(F_1^{-1}(u_1), \ldots, F_d^{-1}(u_d)).$$

where u_i is a random variable with are uniformly distributed on the interval $[0,1]$ for each $i = 1, \ldots d$. If a copula has a density, it can be obtained by:

$$c(u_1, \ldots, u_d) = \frac{\partial C(u_1, \ldots, u_d)}{\partial u_1 \ldots \partial u_d}$$

In the next section we illustrate some parametric families of copula functions.

2.5.1 Gaussian Copula

The Gaussian Copula is the copula derived from gaussian distribution. Let $\Phi_{\rho,d}$ denoted the d-dimension cumulative gaussian distribution with ρ correlation matrix and Φ denoted standard gaussian cumulative. The Gaussian n-copula with correlation ρ is written by:

$$C_\rho(u_1,\ldots,u_d) = \Phi_{\rho,d}(\Phi^{-1}(u_1),\ldots,\Phi^{-1}(u_d))$$

whose density,

$$c_\rho(u_1,\ldots,u_d) = \frac{1}{\sqrt{\det \rho}}\exp\left(-\frac{1}{2}\begin{pmatrix}\Phi^{-1}(u_1)\\ \vdots \\ \Phi^{-1}(u_d)\end{pmatrix}(\rho^{-1}-I)\begin{pmatrix}\Phi^{-1}(u_1)\\ \vdots \\ \Phi^{-1}(u_d)\end{pmatrix}\right)$$

2.5.2 Archimedean Copulas

Another class of copulas are the Archimedean copulas, which are defined by:

$$C_\theta(u_1,\ldots,u_n) = \varphi^{-1}\left(\varphi(u_1)+\cdots+\varphi(u_n)\right)$$

for some generator function φ and its generalized inverse φ^{-1} that satisfied,

- $\varphi(1) = 0$
- $\frac{d\varphi}{dx} > 0$
- $\frac{d^2\varphi}{dx^2} < 0$

Archimedean copulas are important because of their easy with which they can be constructed and the nice properties they possess. Some common Archimedean copulas are Gumbel and Clayton.

2.5.3 Gumbel Copula

Gumbel copula is an asymmetric archimedean copula that has greater dependence in the positive tail than in the negative. It uses the following generalize function:

$$\varphi(u) = (-\log(u))^\theta.$$

The copula function is written as:

$$C_\theta(u_1,\ldots,u_n) = \exp\left(-\left(\sum_i(-\log(u_i))^\theta\right)^{1/\theta}\right),$$

where $\theta \in [1, \text{inf})$. When $n = 2$, there is a directly correlation between θ and kendall's τ correlation given by:

$$\hat{\theta} = \frac{1}{1-\tau}.$$

2.5.4 Clayton Copula

Another class of asymetric archimedean copulas is the Clayton Copula. Different from Gumbel copula, Clayton copula has greater dependence in the negative tail than positive. Its generalize function is given by,

$$\varphi(u) = \frac{1}{\theta}(u^{-\theta} - 1),$$

and copula function,

$$C_\theta(u_1, \ldots, u_n) = \max\left(\left(\sum_i^n u_i^{-\theta} - 1\right)^{-1/\theta}, 0\right),$$

where $\theta \in [-1, \text{inf})$. The relationship between Kendall's τ and copula parameter θ when $n = 2$,

$$\hat{\theta} = \frac{2\tau}{1-\tau}.$$

CHAPTER

3

METHODOLOGY

In this chapter we show how copula can be used together with FlexCode for conditional distribution in order to model multivariate distributions.

3.1 Main Idea

Sklar's Theorem can be extended to conditional distributions, as the next theorem shows.

Theorem 3.1.1. [Sklar's Theorem for conditional distributions] (SKLAR, 1973) Let $(\mathbf{Z}, \mathbf{X})$ be a continuous random variable, and let $F_{\mathbf{z}|\mathbf{x}}$ be its multivariate conditional distribution function with marginals $F_{i|\mathbf{x}}$, $i = 1, \ldots, d_z$, where d_z is the dimension of z. There exists a unique conditional copula $C(\cdot|\mathbf{x})$ such that for all $\mathbf{z} \in \mathbb{R}^{d_z}$ and $\mathbf{x} \in \mathbb{R}^{d_x}$:

$$F(\mathbf{z}|\mathbf{x}) = C(F_{1|\mathbf{x}}(z_1|\mathbf{x}), \ldots, F_{d_z|\mathbf{x}}(z_{d_z}|\mathbf{x})|\mathbf{x}). \tag{3.1}$$

Equation (3.1) implies that the multivariate conditional density $f(\mathbf{z}|\mathbf{x})$ can be written as:

$$f(\mathbf{z}|\mathbf{x}) = c(F(z_1|\mathbf{x}), \ldots, F(z_{d_z}|\mathbf{x})|\mathbf{x}) f(z_1|\mathbf{x}) \ldots, f(z_{d_z}|\mathbf{x}), \tag{3.2}$$

where $c(u_1, \ldots, u_{d_z}|\mathbf{x})$ is the copula density, which is the joint conditional density of the *uniform* random variables $F_{1|\mathbf{x}}(Z_1|\mathbf{x}), \ldots, F_{d_z|\mathbf{x}}(Z_{d_z}|\mathbf{x})$.

The advantage of the characterization in Equation (3.2) is that it decomposes the problem of estimating the multivariate conditional density into the problem of estimating the univariate conditional densities plus the problem of estimating the dependency structure. More precisely, Equation (3.2) motivates the estimator:

$$\widehat{f}(\mathbf{z}|\mathbf{x}) = \widehat{c}(\widehat{F}(z_1|\mathbf{x}), \ldots, \widehat{F}(z_{d_z}|\mathbf{x})|\mathbf{x}) \widehat{f}(z_1|\mathbf{x}) \ldots, \widehat{f}(z_{d_z}|\mathbf{x}), \tag{3.3}$$

where each conditional density (distribution) $\widehat{f}(z_i|\mathbf{x})$ ($\widehat{F}(z_i|\mathbf{x})$) can be estimated via FlexCode. In section 3.2 we show three proposed methods of estimating copulas.

3.2 Copula Estimation

Once estimated each marginal density $\widehat{f}(z_i|\mathbf{x})$ and cumulative density $\widehat{F}(z_i|\mathbf{x})$ via FlexCode. We proposed three ways of estimating the conditional copula $c(\cdot|\mathbf{x})$, which have increasing degrees of complexity:

- MFC1: Assuming a parametric copula family $(c_\theta)_\theta$, with θ estimated via maximum likelihood, i.e.,

$$\begin{aligned}\widehat{\theta} &= \arg\max_\theta \sum_{i=1}^n \log\left(c_\theta(\widehat{F}(z_{i,1}|\mathbf{x}_i),\ldots,\widehat{F}(z_{i,d_z}|\mathbf{x}_i))\widehat{f}(z_{i,1}|\mathbf{x}_i)\ldots,\widehat{f}(z_{i,d_z}|\mathbf{x}_i)\right)\\ &= \arg\max_\theta \sum_{i=1}^n \log\left(c_\theta(\widehat{F}(z_{i,1}|\mathbf{x}_i),\ldots,\widehat{F}(z_{i,d_z}|\mathbf{x}_i))\right) \qquad (3.4)\end{aligned}$$

- MFC2: Assuming a parametric copula family $(c_\theta)_\theta$, with θ *locally* estimated via a maximum weighted likelihood function:

$$\begin{aligned}\widehat{\theta}(\mathbf{x}) &= \arg\max_\theta \sum_{i=1}^n w(\mathbf{x},\mathbf{x}_i)\log\left(c_\theta(\widehat{F}(z_{1,i}|\mathbf{x}_i),\ldots,\widehat{F}(z_{i,d_z}|\mathbf{x}_i))\widehat{f}(z_{1,i}|\mathbf{x}_i)\ldots,\widehat{f}(z_{1,d_z}|\mathbf{x}_i)\right)\\ &= \arg\max_\theta \sum_{i=1}^n w(\mathbf{x},\mathbf{x}_i)\log\left(c_\theta(\widehat{F}(z_{i,1}|\mathbf{x}_i),\ldots,\widehat{F}(z_{i,d_z}|\mathbf{x}_i))\right), \qquad (3.5)\end{aligned}$$

  where the weights $w(\mathbf{x},\mathbf{x}_i)$ are large when $\mathbf{x}$ is similar to $\mathbf{x}_i$. See section 3.2.1 on how to compute such weights.

- MFC3: Using a nonparametric conditional density estimator. For instance, one can use a kernel density estimator,

$$\widehat{c}(\mathbf{u}|\mathbf{x}) \propto \sum_{i=1}^n w(\mathbf{x},\mathbf{x}_i)K_{\mathbf{H}}(\mathbf{u},\mathbf{u}_i),$$

  where $\mathbf{u} = (u_1,\ldots,u_{d_z})$ and $\mathbf{u}_i = (\widehat{F}(z_{i,1}|\mathbf{x}_i),\ldots,\widehat{F}(z_{i,d_z}|\mathbf{x}_i))$. An attention point is that this method does not guarantee that the marginals are uniform.

  K can be for instance a Gaussian kernel:

$$K_{\mathbf{H}}(\mathbf{u},\mathbf{u}_i) = (2\pi)^{-d_z/2}|\mathbf{H}|^{-1/2}\exp\left(\frac{-(\mathbf{u}-\mathbf{u}_i)^{\mathbf{T}}\mathbf{H}^{-1}(\mathbf{u}-\mathbf{u}_i)}{2}\right),$$

  where $\mathbf{H}$ is the bandwidth (or smoothing) d_z x d_z matrix which is symmetric and positive definite. The bandwidth is select by plug-in methodology, this method was proposed by (SHEATHER; JONES, 1991) and is described in Section 3.6 of (WAND; JONES, 1994).

  Note that for MFC1 we are assuming a single parametric copula family for all given $\mathbf{x}$, and as we discussed in section 2.5, this represent a static joint distribution different from MFC2. In this case, θ can assume different values for a given $\mathbf{x}$, in other words the joint distribution can change depending on $\mathbf{x}$. MFC3 is even more complex and assume no parametric family for the joint distribution.

3.2.1 Weights

The objective of the weights used in our semi-parametric and non-parametric estimation is to locally estimate the copula rather than having a single copula for the entire covariate space. Therefore, $w(\mathbf{x}, \mathbf{x}_i)$ are higher when $\mathbf{x}_i$ is similar to $\mathbf{x}$. A useful metric for similarity depends on the problem. For instance, the d-dimensional euclidean distance may not be appropriate for problems where x is high-dimensional because many of its components are irrelevant for predicting Z. Thus, we propose a strategy to build w in a way that considering only relevant features. To perform such task, we choose the random forest proximity matrix to perform distance. Where proximity is the proportion how often two data points end in the same leaf node for different trees (BREIMAN, 2002). The main idea is build a Random forest model with $\mathbf{x}$ as feature and $\mathbf{z}$ as response. Therefore, trees of random forest are build using mostly relevant features of $\mathbf{x}$ to predict $\mathbf{z}$. Then proximity in this case, give higher importance for relevant features. With this, we defined the weights as function of proximity matrix by:

$$w_h(\mathbf{x}, \mathbf{x}_i) = \exp\left(-\frac{1}{hP(\mathbf{x}, \mathbf{x}_i)}\right),$$

where $P(\mathbf{x}, \mathbf{x}_i)$ is the normalize proximity of $\mathbf{x}$ with $\mathbf{x}_i$ and $h \in (0, \infty)$ is the tuning parameter. Figure 1 show for different h how w decrease with P and h shapes the decrease, where $P = 1$ is the greater similarity and $P = 0$ is the minimal similarity.

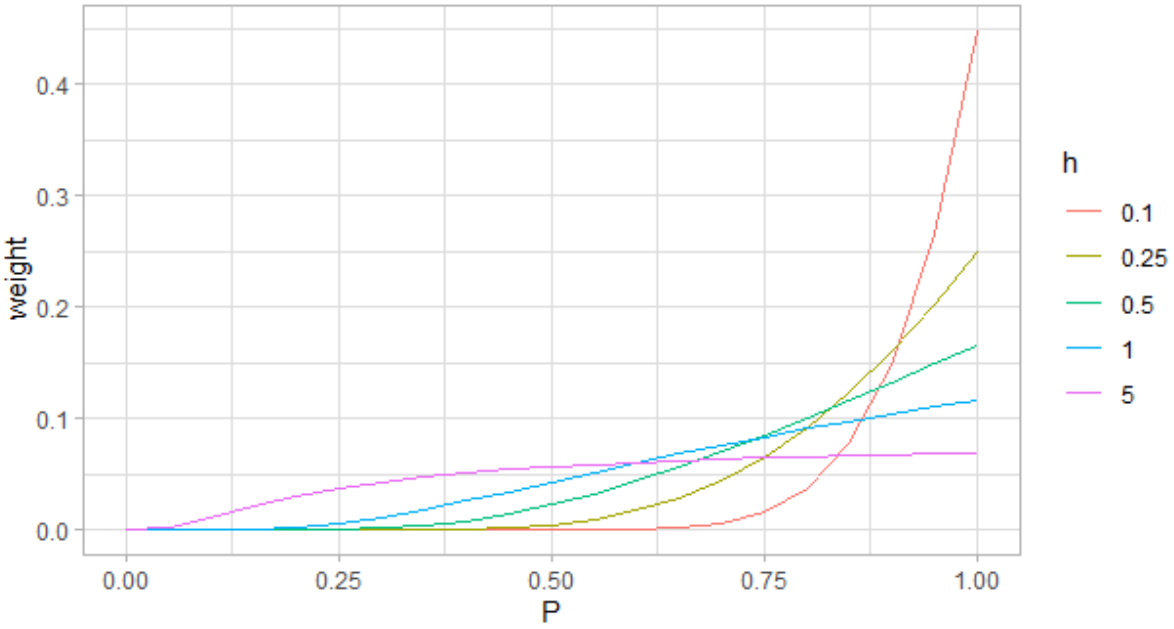


Figure 1 – Calculated weights for different parameters h.

For a sequence of h we use cross-validation to choose h that has better result for estimate copula. In section 2.3 we show how to estimated the goodness-of-fit of a model by loss function.

3.2.2 Tuning Parameters

We described in section 2.4 that to perform the tuning parameters of FlexCode data must be split into two parts, train and validation. For MFC we keep the same strategy. First, FlexCode are computed using the data split as discussed in chapter 2 and then copula is estimated as written in algorithm 2.

Notice that in cases where the weight is computed, algorithm 2 does not return $\widehat{f}(\mathbf{z}|\mathbf{x})$. Instead, it returns the marginals $\widehat{f}(z_i|\mathbf{x})$ and the weight function $w_{\hat{h}}(\mathbf{x},\mathbf{x}_i)$. This happen because for a new data $\mathbf{x}$, $\hat{\theta}(\mathbf{x})$ must be calculated.

Algorithm 2 – Multivariate FlexCode with Copulas

Input Training Data; Validation Data; weight : (true or false); Copula Function : (Gumbel, Clayton , Kernel, ...); H grid

Output Estimator $\widehat{f}(\mathbf{z}|\mathbf{x})$

for all $i \leq d$ **do** ▷ d is the dimension of $\mathbf{z}$
  Compute $\widehat{f}(z_i|\mathbf{x})$ with FlexCode ▷ Section 2.4
end for
if weight is false **then**
  Obtain θ that minimize eq. (3.4)
  Calculate $\widehat{f}(\mathbf{z}|\mathbf{x})$ ▷ Equation (3.3)
else
  Fit Random Forest with Train Data ▷ Obtain $P(\mathbf{x},\mathbf{x}_i)$)
  Split Validation Data into k-fold ▷ To find the h
  for all $h \in H$ **do**
    for all $k \leq K$ **do**
      Calculate $w_h(\mathbf{x}_k,\mathbf{x}_{k^-})$ ▷ where $\mathbf{x}_k$ is the k-fold
      ▷ and $\mathbf{x}_{k^-}$ is all the orders folds
      Estimate $\theta_h(\mathbf{x}_k)$ ▷ Equation (3.5)
      Calculate log-likelihood $l(\theta_h(\mathbf{x}_k),\mathbf{x}_k)$
    end for
    Obtain the average log-likelihood for each h.

$$l_h = \sum_{k=1}^{K} l(\theta_h(\mathbf{x}_k),\mathbf{x}_k)/K$$

  end for
  $\hat{h} = \arg\min_h l_h$
end if
return $\widehat{f}(z_i|\mathbf{x})$; $w_{\hat{h}}(\mathbf{x},\mathbf{x}_i)$

CHAPTER

4

SIMULATION

To test the predictive capability of proposed method, we simulated 3 settings: a bi-normal distribution with constant correlation matrix Σ for case 1, a bi-normal distribution with conditional variance for case 2 and conditional variance and correlation for case 3. For all cases we simulated covariate matrix X with 100 features and 1500 observations. The scenarios are detail in section 4.1.

4.1 Simulation 1

From covariate matrix X, we define:

$$\mu_1(x) = 3 + 0.5x_1 + 5x_2 - 1x_3 + 3x_4 - 1.1x_5$$
$$\mu_2(x) = 5 + 0.1x_1 + 2x_2 - 3x_3 + 3x_4 + 1.2x_5$$

Note that for all 100 features, we are using only 5 to calculated $\mu(x) = (\mu_1(x), \mu_2(x))$. We investigate the following cases:

1. $$(Z_1, Z_2)|X = x \sim \mathtt{N}_2(\mu(x), \Sigma)$$

2. $$(Z_1, Z_2)|X = x \sim \mathtt{N}_2(\mu(x), \Sigma(x)),$$

   where
   $$\Sigma(x) = \begin{pmatrix} 0.5 + x_1 & 0 \\ 0 & 0.5 + x1 \end{pmatrix}$$

3. $$(Z_1, Z_2)|X = x \sim \mathtt{N}_2(\mu(x), \Sigma(x)),$$

where

$$\Sigma(x) = \begin{pmatrix} 0.5 + x_1 & x_1/2 \\ x_1/2 & 0.5 + x1 \end{pmatrix}$$

4.2 Results

To estimate conditional densities we used FlexCode with gradient boosting. We used the following copula functions: Gumbel, Gaussian and Clayton. Because they gave similar results, we only show the results for Gumbel. The three copula approach was tested, we call parametric approach as MFC1, semi-parametric as MFC2 and non-parametric MFC3. For the sake of comparison, we also fit two other conditional methods: Kernel Density Estimator (Kernel, (SAIN, 2002)), Random Forest for Conditional Density (RF, (POSPISIL; LEE, 2018)) and FlexCode for multivariate response without copulas.

Table 1 – Estimate Loss Function of Simulation Cases 1,2 and 3

	MFC 1	MFC 2	MFC 3	RF	Kernel	FlexCode
Case 1	-0.038	-0.037	**-0.052**	-0.003	-3.9e-05	-0.011
Case 2	-0.022	-0.018	**-0.038**	-0.003	-2.5e-05	-0.010
Case 3	-0.032	-0.029	**-0.036**	-0.003	-1.3e-05	-0.011

The estimated losses in Table 1 shows that MFC in the 3 copula approaches have better results than RF, Kernel and FlexCode. Figure 2 shows an example of true density (blue) and the estimated density (red) of all six models. MFC1, MFC2 and MFC3 lead to estimates that are very close to the true density, while Random Forest, Kernel density and FlexCode lead to much wider estimates.

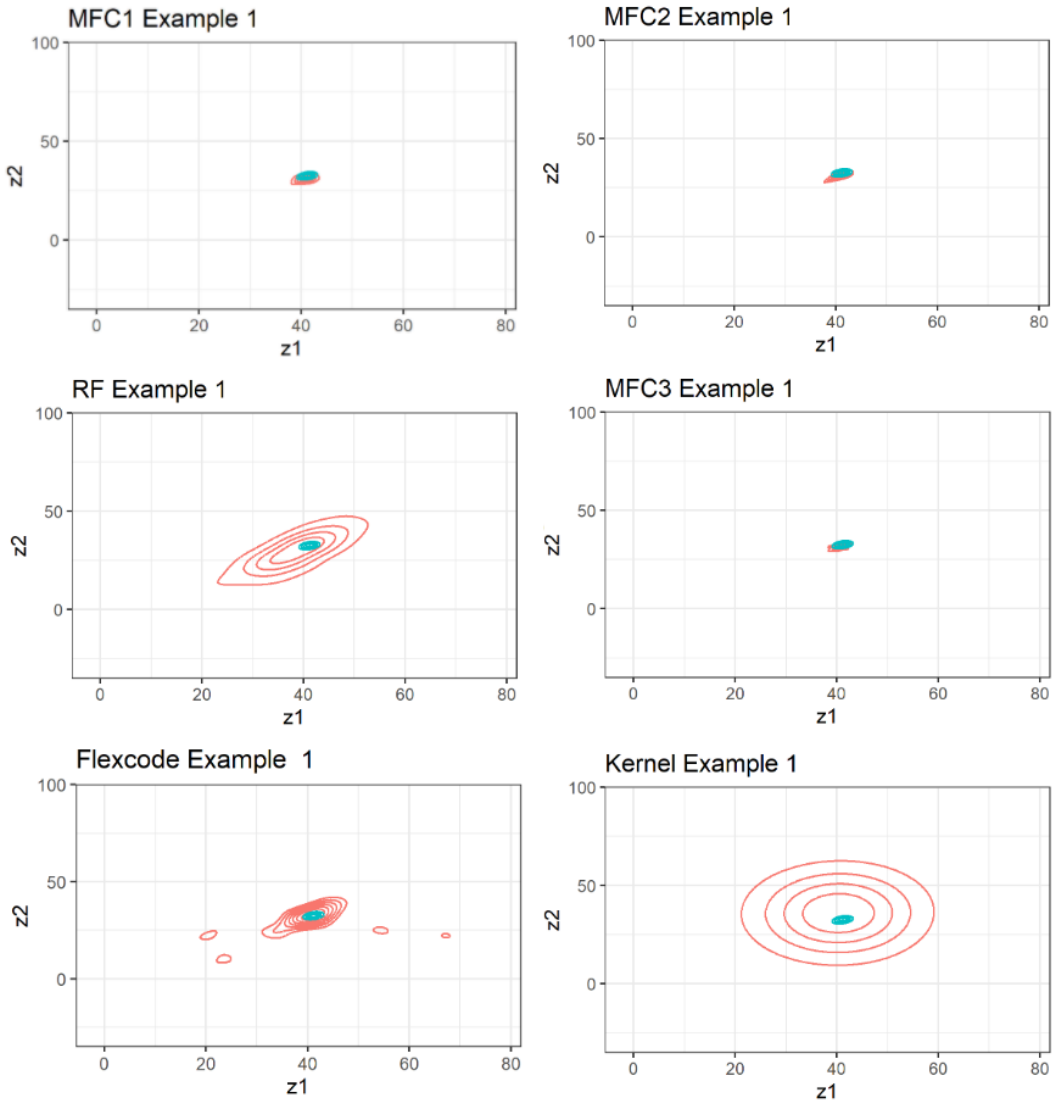


Figure 2 – Result CDE of Example 1 of Case 3.

When we compare system CPU time to fit and predict densities values in a grid of size 1000 x 1000 of the different models, we see in fig. 3 that RF performed best followed by MFC3. The FlexCode without copulas performed 5 times slower than the approach with copulas. Kernel estimation couldn't be computed because was too slow for sample size higher than 2000.

Figure 4 shows that the loss decreases faster as a function of the sample size for the MFC methods than random forests and FlexCode.

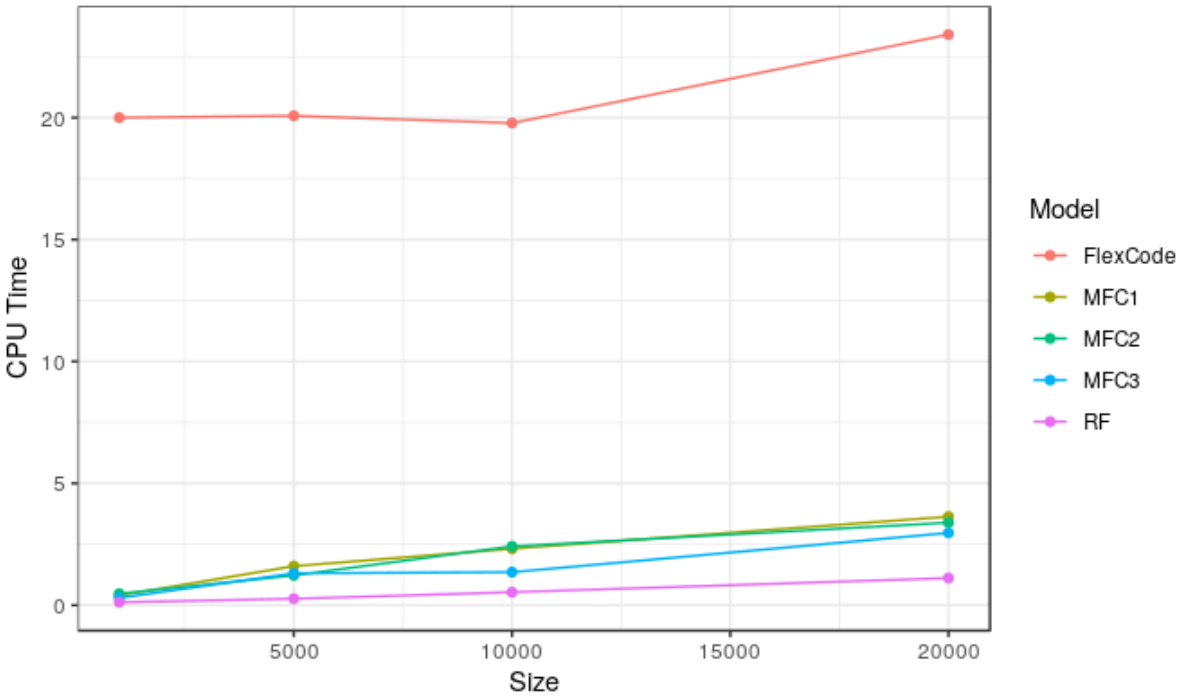


Figure 3 – Comparison of the time needed to fit each model as a function of the sample size.

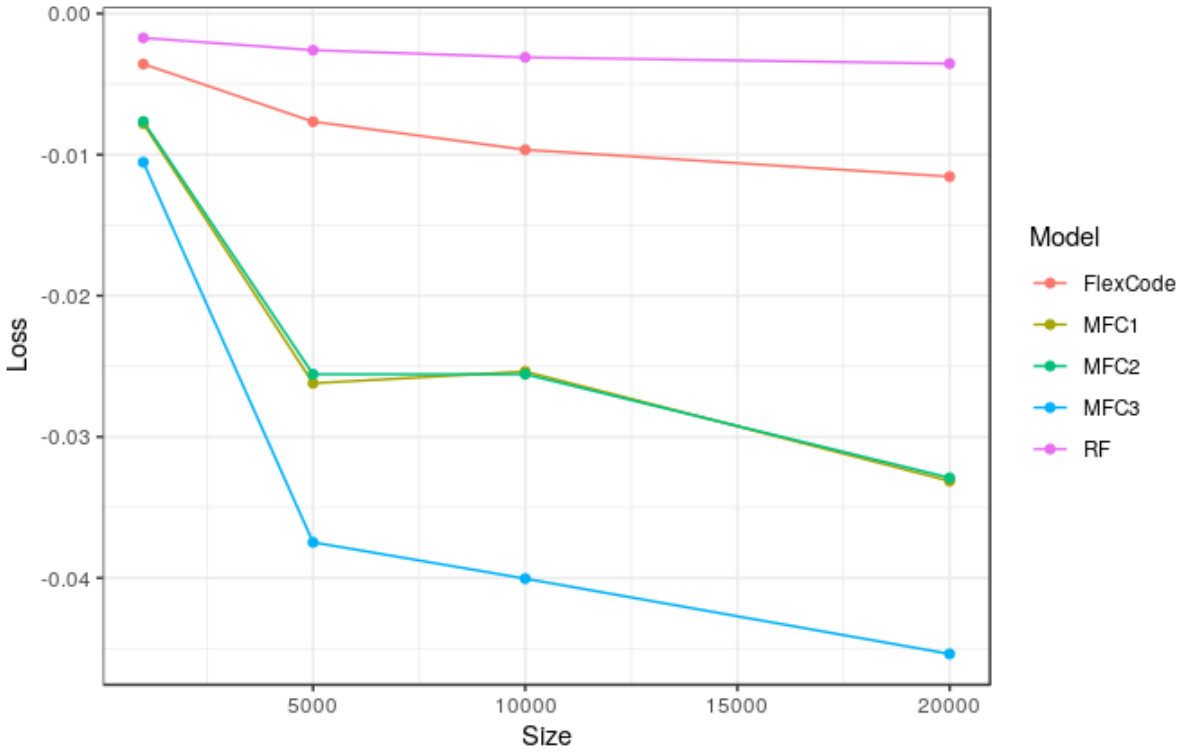


Figure 4 – Comparison of the loss functions of each model as a function of the sample size.

CHAPTER

5

REAL DATASET APPLICATIONS

We apply our methods to two real datasets. The objective was to estimate the geolocation (latitude and longitude) for two types of data. Predicting geolocation is a good to better understand how the models behave: the results are easy to interpret and usually the underlying density has asymmetric and multimodal distribution. For instance, suppose that someone did a google search "tide tables" and we want to estimate the location of this person when he did this search. We expect that the probability of this person searched close to the ocean is greater than not being close. So the CDE, in this case, is expected to have a higher value in coastal regions. As another example, assume that we want to do the same estimate of longitude and latitude from a person who tweets "it's very cold today" in July. We expect that the probability is greater in the South of the equator line than in the North because July is winter in the South and summer in the North.

To run our model we created two example with different feature types. The first is a text mining exercise with a Twitter dataset collected on July. Our features are the words of each tweet and as a response the latitude and longitude where the tweet was posted. The second is an e-commerce dataset of a marketplace that works like Amazon. The objective is to estimate the location of the next client of a specific product of a seller.

5.1 Twitter Dataset

The Twitter dataset was introduced in (IZBICKI; LEE, 2017b). The dataset contains 15000 tweets with GPS location (latitude and longitude) that were written in July in the languages Portuguese and Spanish. We picked tweets that have at least one word about climate. As explained at the beginning of this chapter, tweets about the weather in a specific season of the year are easy to interpret the location CDE.

In the process mining of the sentences, we removed the stop words and used bag-of-words with bigram and trigram tokens, for more details (MARTIN, 2009). Then, we fitted the 3 copula

approaches of MFC and compare them to Random Forest density estimation (POSPISIL; LEE, 2018). In this example, it was computationally unfeasible to fit the kernel density estimator.

Tweet: *"Hola compi de Isla y vecino de pueblo cordobá, moriremos de calor jajajaja,naah sobreviviremos (3:13am)"*

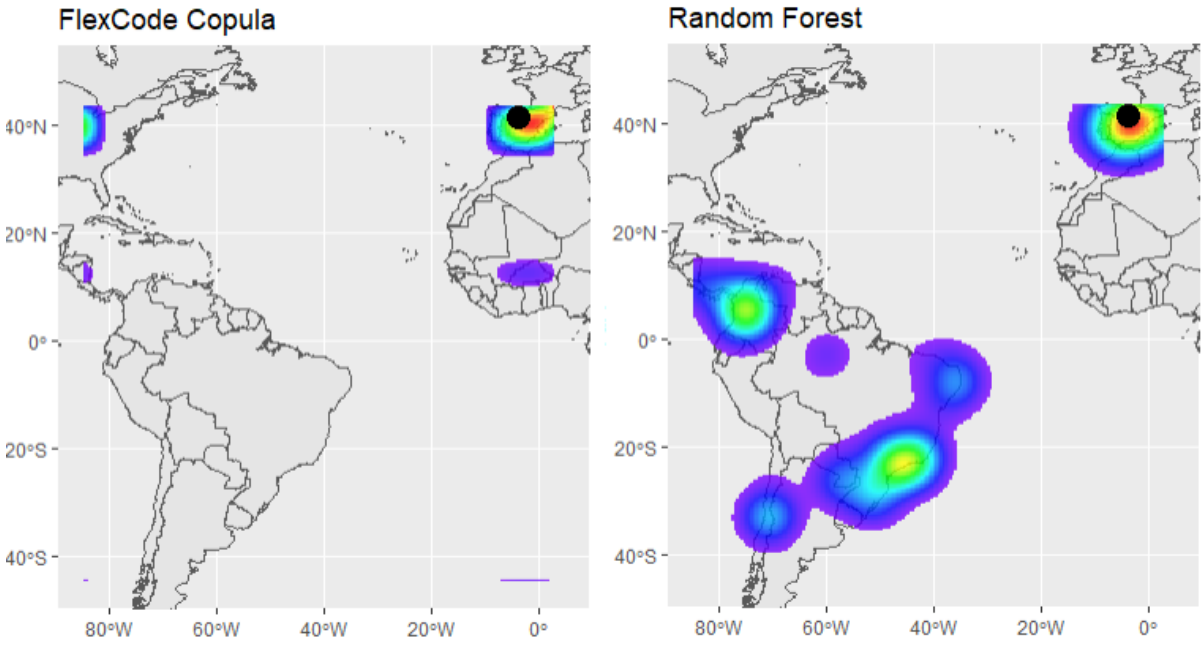


Figure 5 – CDE of MFC and RF of example 1 of twitter dataset

Tweet: *"ta muito frio"*

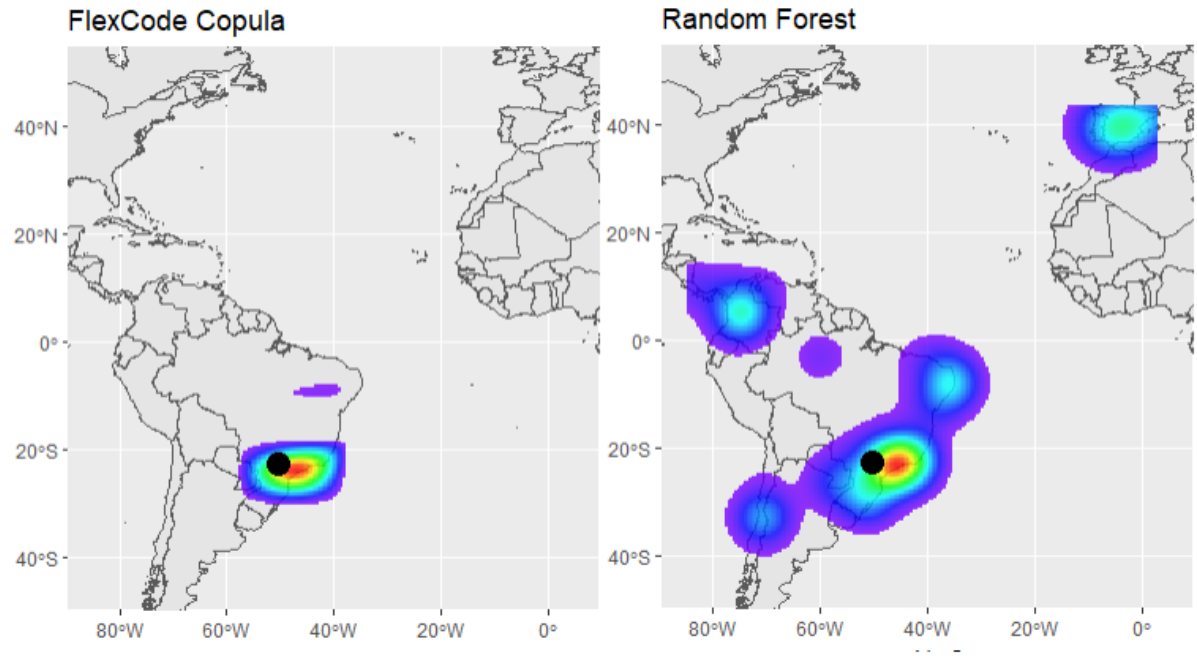


Figure 6 – CDE of MFC and RF of example 2 of twitter dataset.

Figures 5 and 6 show the heat map of conditional density estimation of MFC with semi-parametric copula approach in left and random forest in the right. The black dot represents

the true location of a tweet. In fig. 5 tweet in Spanish that mentions a city in Spain "Cordoba", both models estimated well a higher probability in Spain, but Random Forest assigned a high probability to Brazil, which is not reasonable for this example. The same behave of RF is presented in Figure 6, the tweet in Portuguese, but RF estimate densities in Chile and Bolivia.

Table 2 – Estimated Loss Function of twitter example.

MFC 1	MFC 2	MFC 3	RF
-0.0065 (0.0052)	-0.0063 (0.006)	-0.012 (0.022)	-0.0028 (5.1e-05)

Visually, MFC1 and MFC2 gives better estimates in this example, the estimated density works as expected. However, MFC3 leads to a slightly better solution with higher variability and Random Forest had the worst performance, table 2.

5.2 Brazilian E-commerce Dataset

The Brazilian E-commerce dataset has been provided by Olist on Kaggle website (Kaggle...,). It contains data about a marketplace in Brazil that connects sellers to clients with different categories of products. For the purpose of this study, we choose to use as the response the latitude and longitude of the delivery location for each order, while for the features we included the information about the seller (City and State) and the product sold (Category, Size, Weight, etc...). We included non-significant features like the size and weight of the product on purpose to evaluate how models behave. A good density model will place larger mass on places that have a higher demand for products, such as capitals and big cities.

We fitted the 3 MFC models, random forest CDE model and FlexCode with tensor products. We considered 50000 observations with 714 columns of features, (because we included cities and states of sellers throughout Brazil and treat all the category features as a dummies).

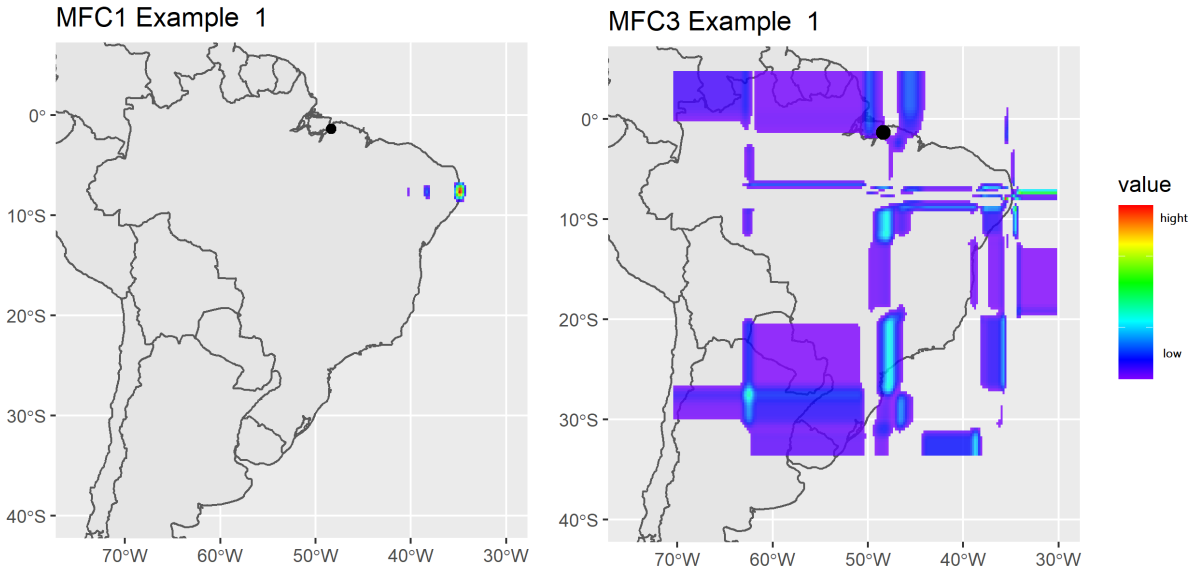


Figure 7 – CDE of observation 1 on the test group - models MFC1 and MFC3.

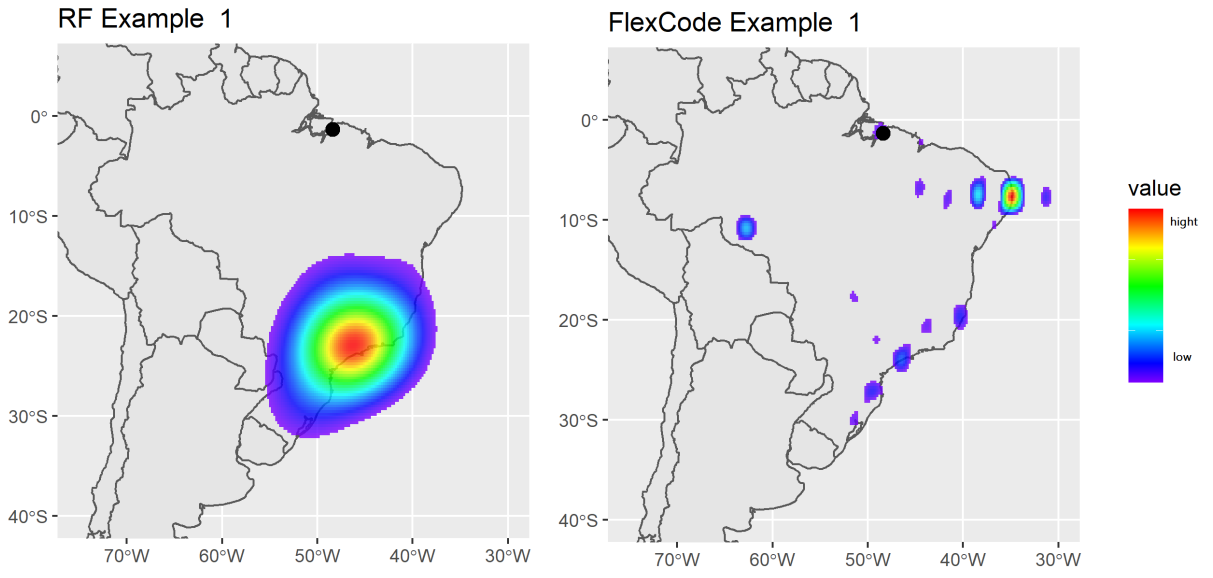


Figure 8 – CDE of observation 1 on the test group - models RF and FlexCode.

We only considered for visualization MFC1 because MFC1 and MFC2 had very similar results. Figures 7 and 8 show a CDE heat map of an observation on the test group. The real delivery location is Belem, the capital of Pará, and is mark by a dot point on the map. We have very different CDE result for each model. The MFC1 result has high values of conditional densities in the right corner of Brazil in Recife city, very far from the real value. MFC3 is always very sparse and without a clear pattern. For RF the estimation is only one big spot on the map, also very far from the real value. FlexCode presented a more interesting result having a high density next to big cities and capitals of Brazil like São Paulo, Recife, Curitiba, and Porto Alegre.

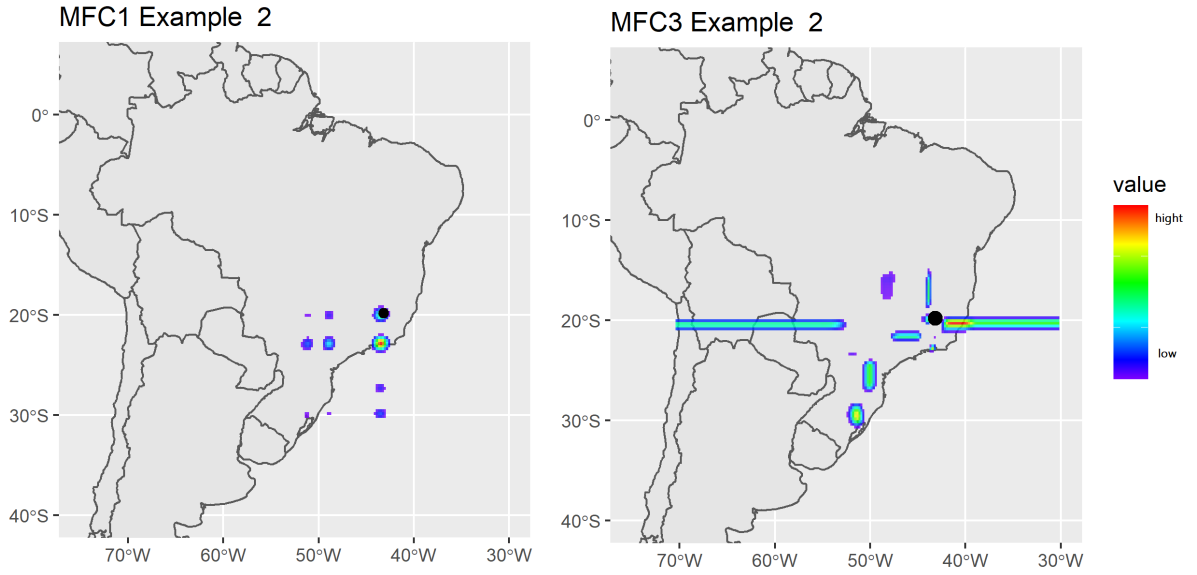


Figure 9 – CDE of MFC and RF of an example 2 of twitter dataset.

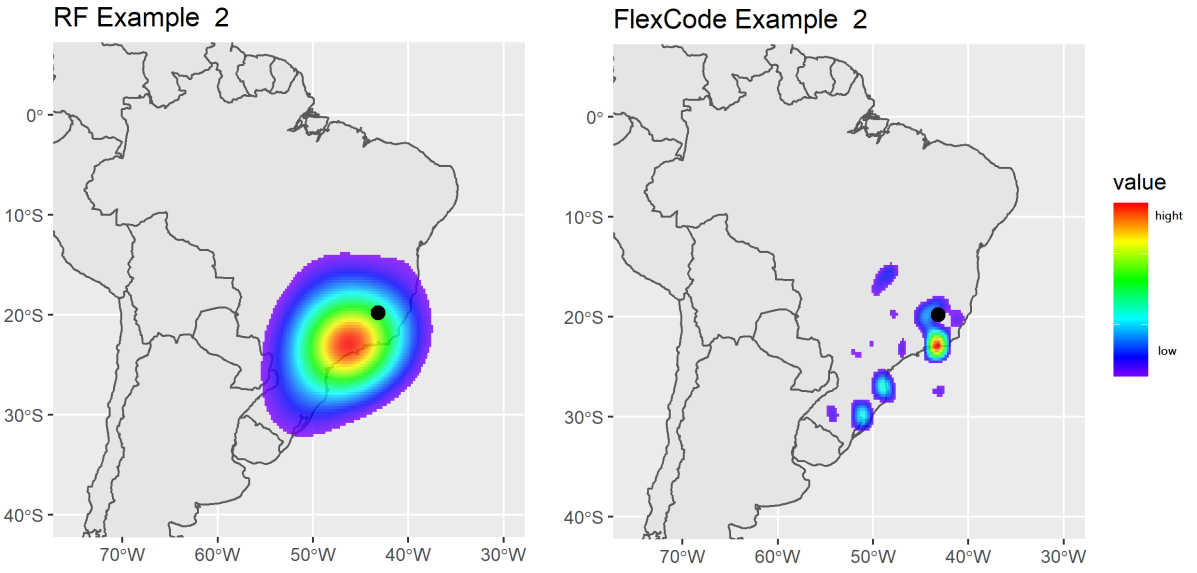


Figure 10 – CDE of MFC and RF of an example 2 of twitter dataset.

The second example represented in figures 9 and 10 is of a product delivery next to Belo Horizonte/MG by a seller in São José do Rio Preto/SP. Both MFC1 and FlexCode estimated high density around some large cities. MFC1 estimated high-density regions around São José do Rio Preto and Rio de Janeiro/RJ. FlexCode also highlighted other cities like Curitiba, Florianópolis, and Porto Alegre. RF has the same sparse result as the other example and MFC3 didn't have a clear pattern.

The estimated loss function on Table 3 indicates that the MFC1 and MFC2 have better results. The MFC1 and MFC3 took half the CPU time to fit and predict compared to FlexCode,

Table 3 – Estimated Loss Function of ecommerce example

MFC 1	MFC 2	MFC 3	RF	FlexCode
-0.2142 (0.126)	-0.2177 (0.132)	-0.0046 (0.002)	-0.0067 (4.3e-05)	-0.0621 (0.047)

but MFC2 had a reduction of only 20% of the CPU time, see Figure 11.

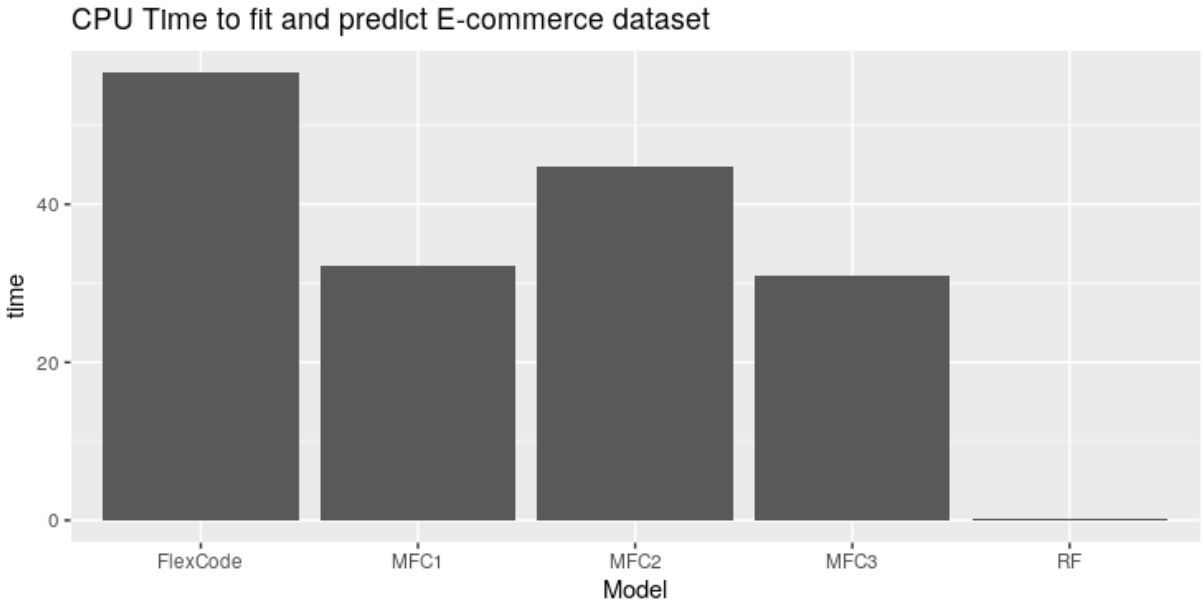


Figure 11 – CDE of MFC and RF of an example 2 of twitter dataset.

CHAPTER

6

CONCLUSION

Simulations and real cases examples showed promising results of the proposed model. In particular, the MFC model is capable of handle high dimension feature space with many non-significant features. Compared to random forest conditional density estimation, kernel density, and FlexCode without copulas, MFC had significantly better simulations results. Besides that, we had great CPU time results, MFC performed 4 times faster than FlexCode without copulas.

In real datasets, MFC also has significantly better results compared to the other models. In the Twitter example the densities estimated by MFC behaves visually as expected, yielding higher densities in the right places according to the inputs. In the E-commerce example, the MFC1 approach had better results than FlexCode with tensor products, and also has the advantage of being estimated with half the CPU time.

Copulas functions combined with FlexCode were shown to be a powerful tool for estimate conditional densities when dealing with high dimension space and multivariate response, both in terms of goodness-of-fitness as well as CPU time.

In future work, we will explore other methods to compute weights in both the semi-parametric copula approach. Our results indicate that the weights did not work as expected, leading to similar results as the parametric copula approach. Instead of using the random forest proximity matrix, we will use metrics that are directly correlated to the copula parameter θ, such as Kendall's τ. We believe that Kendall's τ could be a solution to select features that are relevant to estimate θ. The features that are important to estimate Kendall's τ may be significant to estimate θ too.

BIBLIOGRAPHY

BASHTANNYK, D. M.; HYNDMAN, R. J. Bandwidth selection for kernel conditional density estimation. **Computational Statistics & Data Analysis**, Elsevier, v. 36, n. 3, p. 279–298, 2001. Citation on page 17.

BREIMAN, L. Manual on setting up, using, and understanding random forests. Wiley Online Library, p. 18–19, 2002. Citation on page 27.

COMANICIU, D. An algorithm for data-driven bandwidth selection. **IEEE Transactions on pattern analysis and machine intelligence**, IEEE, v. 25, n. 2, p. 281–288, 2003. Citation on page 17.

FAN, Y.; NOTT. Approximate bayesian computation via regression density estimation. **Stat**, Wiley Online Library, v. 2, n. 1, p. 34–48, 2013. Citation on page 17.

HALL, P. On kullback-leibler loss and density estimation. **The Annals of Statistics**, JSTOR, p. 1491–1519, 1987. Citation on page 20.

IZBICKI, R.; LEE. Converting high-dimensional regression to high-dimensional conditional density estimation. **Electronic Journal of Statistics**, The Institute of Mathematical Statistics and the Bernoulli Society, v. 11, n. 2, p. 2800–2831, 2017. Citation on page 17.

IZBICKI, R.; LEE, A. Converting high-dimensional regression to high-dimensional conditional density estimation. **Eletronic Journal of Statistics**, 2017. Citation on page 33.

JONDEAU, E.; ROCKINGER, M. The copula-garch model of conditional dependencies: An international stock market application. **Journal of international money and finance**, Elsevier, v. 25, n. 5, p. 827–853, 2006. Citation on page 18.

Kaggle Brazilian E-commerce. <https://www.kaggle.com/olistbr/brazilian-ecommerce>. Citation on page 35.

KALDA, A.; SIDDIQUI, S. Nonparametric conditional density estimation of short-term interest rate movements: procedures, results and risk management implications. **Applied Financial Economics**, Taylor & Francis, v. 23, n. 8, p. 671–684, 2013. Citation on page 17.

MARTIN, J. H. **Speech and language processing: An introduction to natural language processing, computational linguistics, and speech recognition**. [S.l.]: Pearson/Prentice Hall, 2009. 29-35 p. Citation on page 33.

POSPISIL, T.; LEE, A. B. Rfcde: Random forests for conditional density estimation. **arXiv preprint arXiv:1804.05753**, 2018. Citations on pages 17, 30, and 34.

ROMANO, C. Calibrating and simulating copula functions: an application to the italian stock market. **Risk Management Function, Capitalia, Viale U. Tupini**, v. 180, 2002. Citation on page 18.

ROSENBLATT, M. Conditional probability density and regression estimators. **Multivariate analysis II**, Academic Press New York, v. 25, p. 31, 1969. Citation on page 17.

SAIN, S. R. Multivariate locally adaptive density estimation. **Computational Statistics & Data Analysis**, Elsevier, v. 39, n. 2, p. 165–186, 2002. Citation on page 30.

SHARMA, A.; LALL, U.; TARBOTON, D. G. Kernel bandwidth selection for a first order nonparametric streamflow simulation model. **Stochastic Hydrology and Hydraulics**, Springer, v. 12, n. 1, p. 33–52, 1998. Citation on page 17.

SHEATHER, S. J.; JONES, M. C. A reliable data-based bandwidth selection method for kernel density estimation. **Journal of the Royal Statistical Society: Series B (Methodological)**, Wiley Online Library, v. 53, n. 3, p. 683–690, 1991. Citations on pages 17 and 26.

SKLAR, A. Random variables, joint distribution functions, and copulas. **Kybernetika**, Institute of Information Theory and Automation AS CR, v. 9, n. 6, p. 449–460, 1973. Citation on page 25.

WAND, M. P.; JONES, M. C. **Kernel smoothing**. [S.l.]: Crc Press, 1994. Citation on page 26.

ZHANG, Y.; DUKIC, V. Predicting multivariate insurance loss payments under the bayesian copula framework. **Journal of Risk and Insurance**, Wiley Online Library, v. 80, n. 4, p. 891–919, 2013. Citation on page 18.

APPENDIX

A

GRAPHICS RESULTS SIMULATION EXAMPLE

In figures 4, 5 and 6 we can see more observations examples of cases 1, 2 and 3 that was describe in chapter 4.

Table 4 – Result Case 1

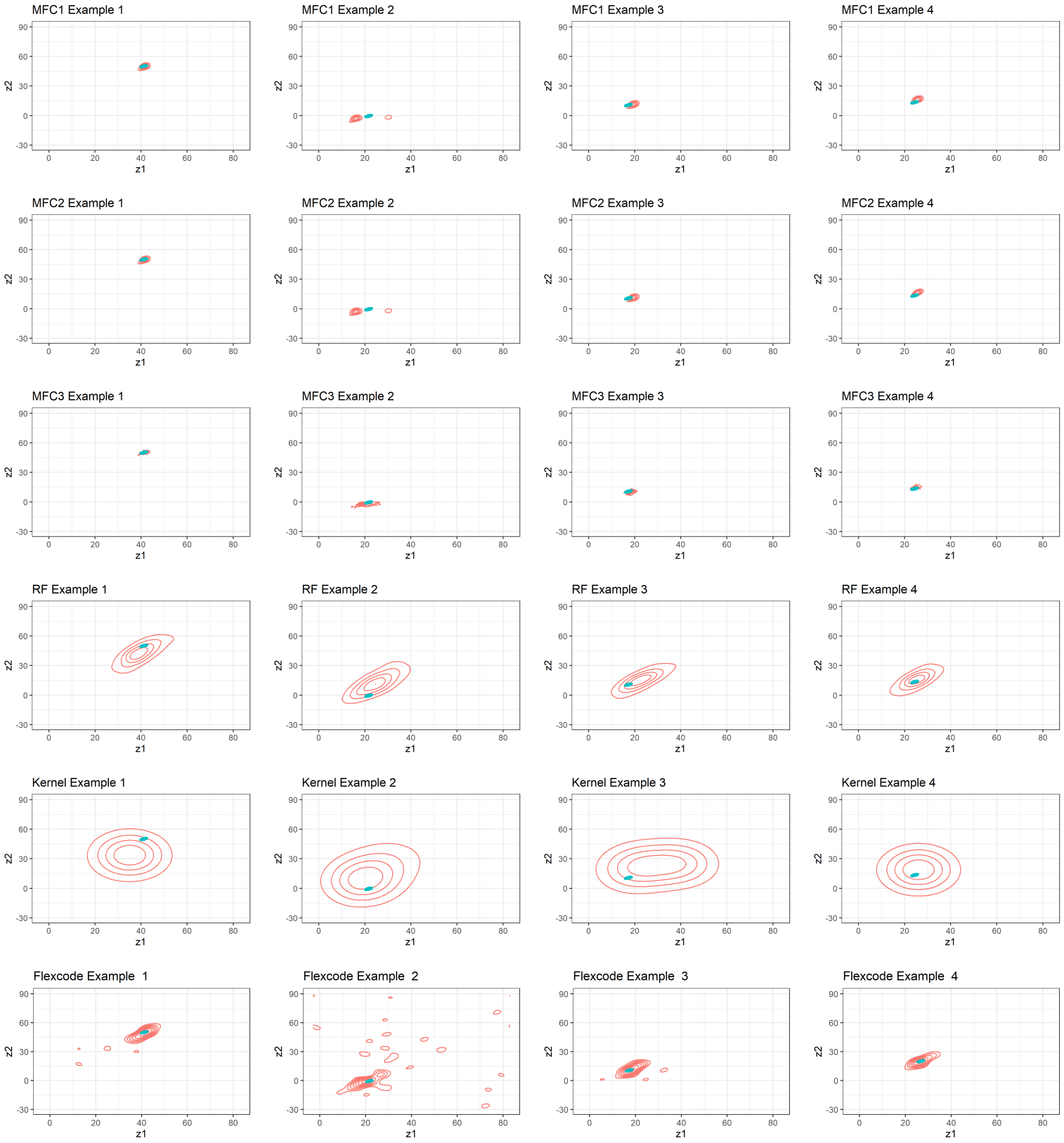


Table 5 – Result Case 2

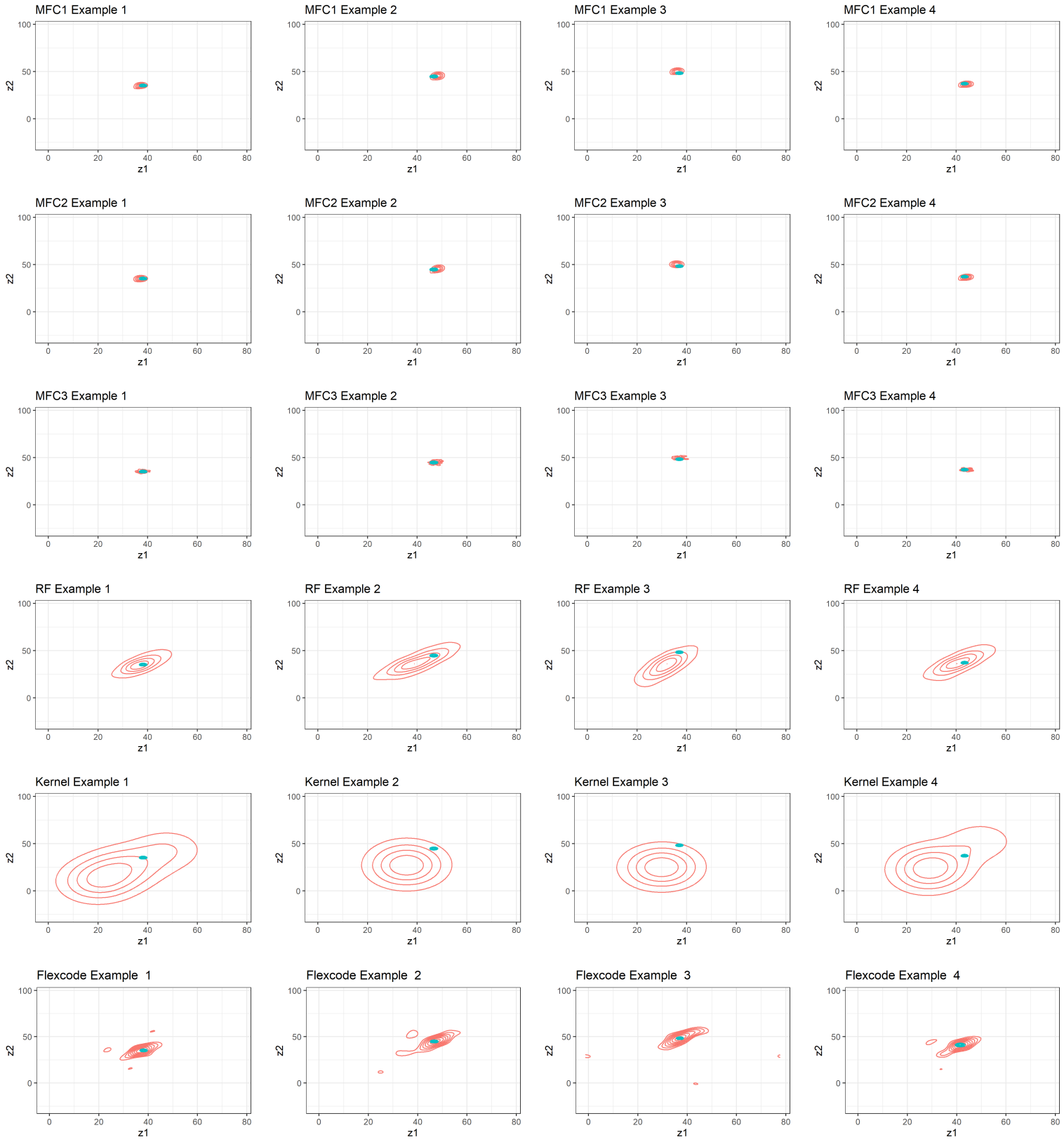
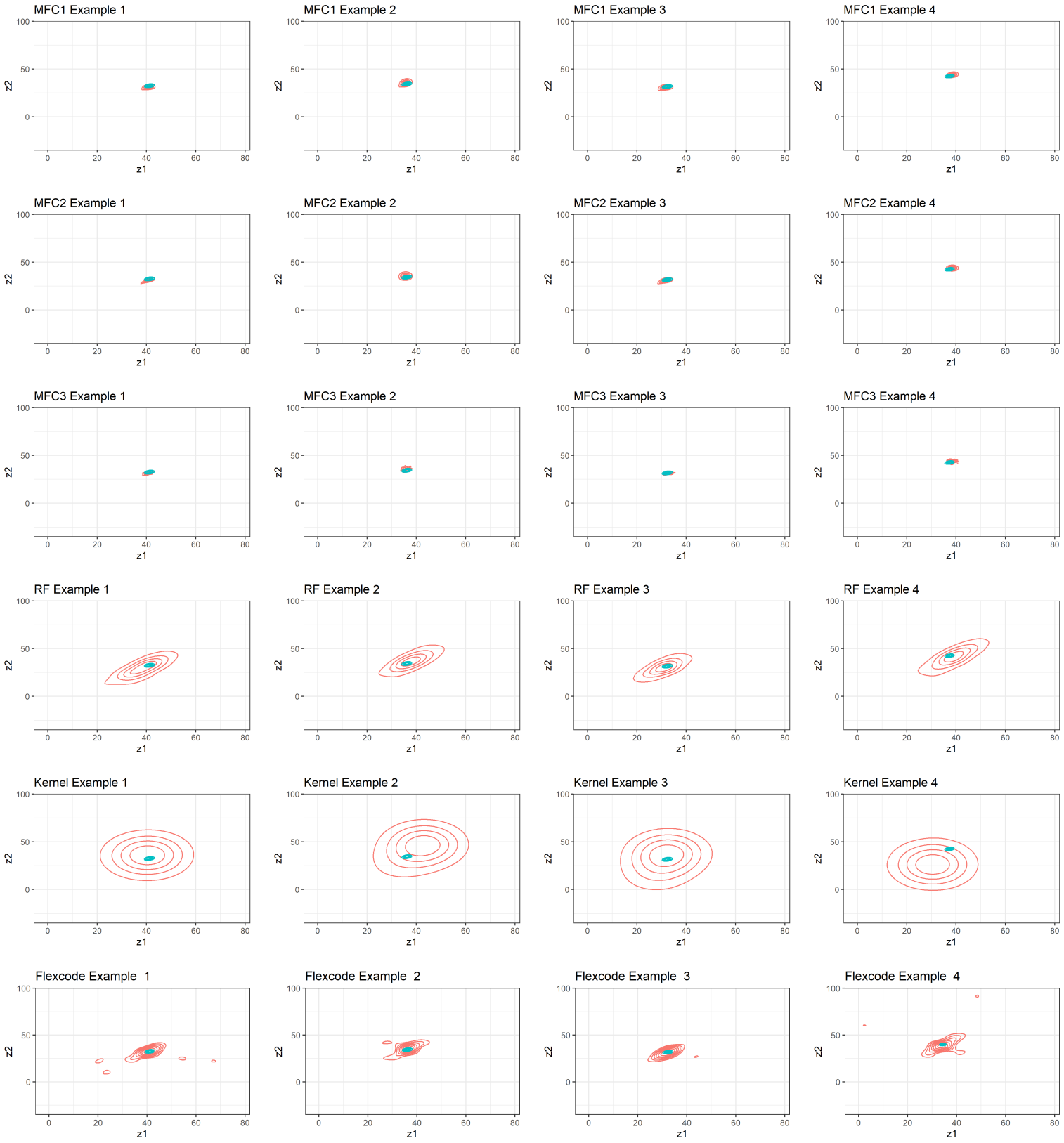


Table 6 – Result Case 3



APPENDIX

B

GRAPHICS RESULTS REAL CASE EXAMPLE

In figures 7, 8 and **??** we can see more observations examples of cases 1, 2 and 3 that was describe in chapter 5.

Table 7 – Twitter Example

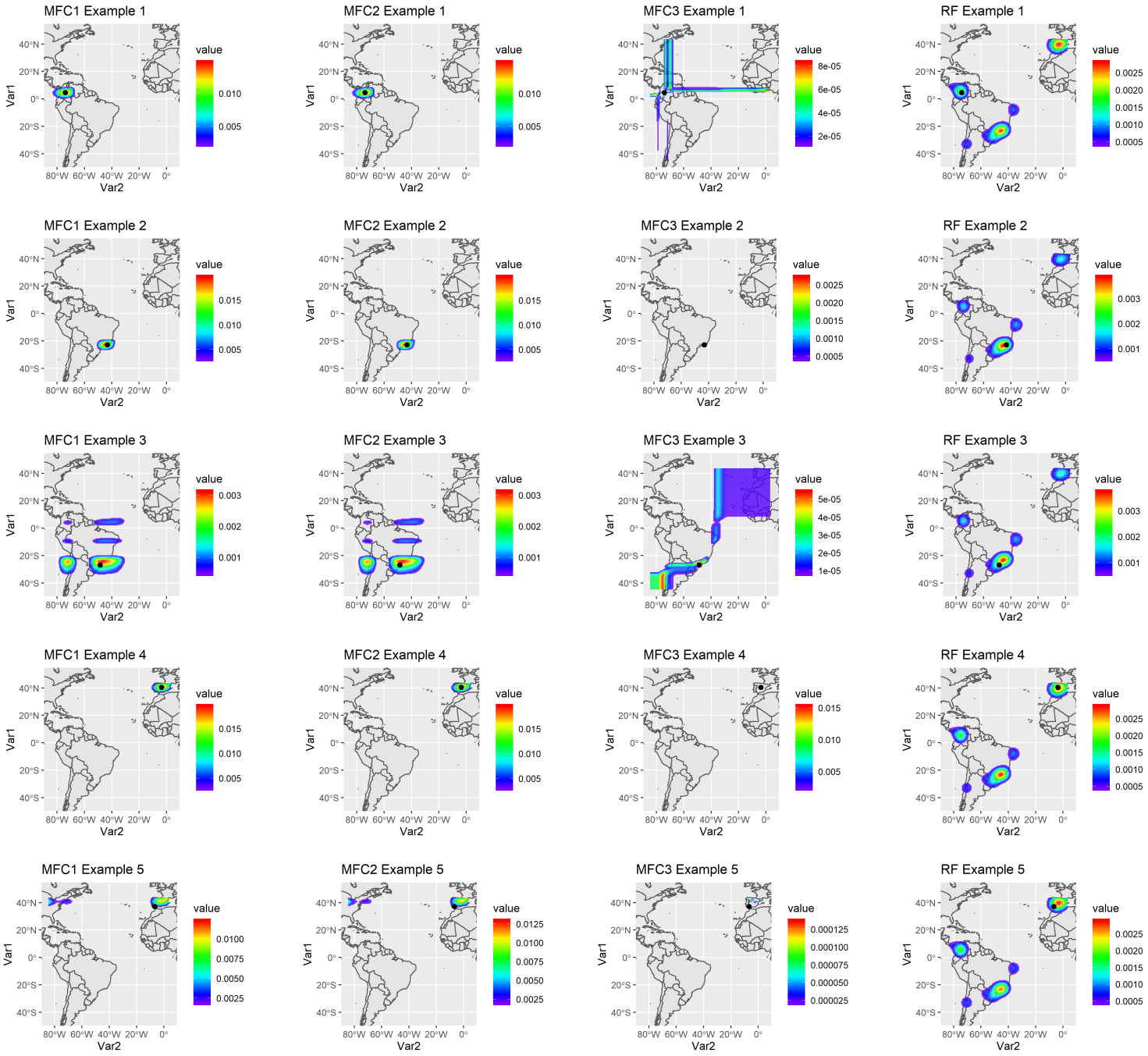


Table 8 – Ecommerce Example