UNIVERSIDADE TECNOLÓGICA FEDERAL DO PARANÁ
CAMPUS CORNÉLIO PROCÓPIO
DIRETORIA DE PESQUISA E PÓS-GRADUAÇÃO
PROGRAMA DE PÓS-GRADUAÇÃO EM INFORMÁTICA

LUIS GUSTAVO DE CARVALHO UZAI

DETECÇÃO DE PONTO DE MUDANÇA EM SÉRIES TEMPORAIS UTILIZANDO O ESPECTRO DO GRAFO

DISSERTAÇÃO DE MESTRADO

CORNÉLIO PROCÓPIO

2020

LUIS GUSTAVO DE CARVALHO UZAI

DETECÇÃO DE PONTO DE MUDANÇA EM SÉRIES TEMPORAIS UTILIZANDO O ESPECTRO DO GRAFO

Dissertação apresentada ao Programa de Pós-Graduação em Informática da Universidade Tecnológica Federal do Paraná – UTFPR como requisito parcial para a obtenção do título de "Mestre em Informática" - Área de Concentração: Inteligência Computacional.

Orientador: Prof. Dr. André Yoshiaki Kashiwabara

CORNÉLIO PROCÓPIO

2020

Dados Internacionais de Catalogação na Publicação

U99 Uzai, Luis Gustavo de Carvalho

Detecção de ponto de mudança em séries temporais utilizando o espectro do grafo / Luis Gustavo de Carvalho Uzai. - 2020.
102 f. : il. color. ; 31 cm.

Orientador: Andre Yoshiaki Kashiwabara.
Dissertação (Mestrado) – Universidade Tecnológica Federal do Paraná. Programa de Pós-Graduação em Informática, Cornélio Procópio, 2020.
Bibliografia: p. 68-72.

1. Análise de séries temporais. 2. Inteligência artificial. 3. Teoria dos grafos. 4. Informática – Dissertações. I. Kashiwabara, Andre Yoshiaki, orient. II. Universidade Tecnológica Federal do Paraná. Programa de Pós-Graduação em Informática. III. Título.

CDD (22. ed.) 004

Biblioteca da UTFPR - Câmpus Cornélio Procópio
Bibliotecário/Documentalista responsável:
Romeu Righetti de Araujo – CRB-9/1676

Ministério da Educação
Universidade Tecnológica Federal do Paraná
Câmpus Cornélio Procópio
Programa de Pós-Graduação em Informática

Título da Dissertação Nº 65:

"DETECÇÃO DE PONTO DE MUDANÇA EM SÉRIES TEMPORAIS UTILIZANDO O ESPECTRO DO GRAFO".

por

Luis Gustavo de Carvalho Uzai

Orientador: **Prof. Dr. André Yoshiaki Kashiwabara**

Esta dissertação foi apresentada como requisito parcial à obtenção do grau de MESTRE EM INFORMÁTICA – Área de Concentração: Computação Aplicada, pelo Programa de Pós-Graduação em Informática – PPGI – da Universidade Tecnológica Federal do Paraná – UTFPR – Câmpus Cornélio Procópio, às 09h30 do dia 15 de agosto de 2019. O trabalho foi _____ pela Banca Examinadora, composta pelos professores:

Prof. Dr. André Yoshiaki Kashiwabara
(Presidente – UTFPR-CP)

Prof. Dr. Fabricio Martins Lopes
(UTFPR-CP)

Prof. Dr. Sylvio Barbon Junior
(UEL)

Visto da coordenação:

Danilo Sipoli Sanches
Coordenador do Programa de Pós-Graduação em Informática
UTFPR Câmpus Cornélio Procópio

A Folha de Aprovação assinada encontra-se na Coordenação do Programa.

Agradeço primeiramente minha mãe Augusta Aparecida de Carvalho Uzai, e ao meu Pai Oswaldeci Uzai que descanse em paz, que me formaram como homem e me deram condições progredir academicamente.

Ao meu orientador, Prof. Dr. André Yoshiaki Kashiwabara que auxiliou o desenvolvimento desse trabalho, respeitando momentos de dificuldade e dor, com paciência e profissionalismo soube conduzir esse trabalho até sua conclusão.

A minha esposa que soube dividir meu tempo e ser compreensiva em momentos de ausência, sempre me apoiando e incentivando.

Aos professores e funcionários do Programa de Mestrado em Informática da UTFPR, com trabalho eficiente com colaboração e Cooperação.

RESUMO

UZAI, Luis Gustavo. DETECÇÃO DE PONTO DE MUDANÇA EM SÉRIES TEMPORAIS UTILIZANDO O ESPECTRO DO GRAFO. 103 f. Dissertação De Mestrado – Programa de Pós-graduação em Informática, Universidade Tecnológica Federal do Paraná. Cornélio Procópio, 2020.

Séries temporais são sequência de valores distribuídos ao longo do tempo. Analisar séries temporais é importante em várias áreas, incluindo médica, financeira, aeroespacial, comercial e entretenimento. Detecção de pontos de mudança é o problema em identificar a mudança no significado ou distribuição dos dados em uma série temporal. O interesse acadêmico e comercial no tema foi ampliado na ultima década devido ao aumento de potência e complexidade de sensores, além do avanço de processos tecnológicos, que, possibilitaram a captura e reconhecimento de um grande volume de dados. Muitos algoritmos amplamente utilizados na atualidade tem dificuldade para chegar em resultados ótimos quando o número de dimensões aumenta ou o volume de dados cresce exponencialmente, também existem séries temporais onde a maioria dos algoritmos não possui resultados satisfatórios (mais de 0.80 de precisão) mesmo considerando dados bidimensionais como em distribuições baseadas em contexto. A maioria dos algoritmos conhecidos é mais eficiente em cenários específicos e menos em outros, por tanto, um número maior de opções de soluções aumenta a probabilidade do usuário obter um algoritmo que atenda melhor suas necessidades. Soluções para essas questões são de grande interesse ecológico e econômico. O objetivo deste trabalho é o desenvolvimento de um algoritmo para detecção de pontos de mudança, não supervisionado que seja aplicável em séries com múltiplos pontos de mudança e com um grande volume de dados com alta precisão. Para atingir esse objetivo foi desenvolvido o novo método *SpecDetec*, um algoritmo que utiliza o agrupamento com espectro de grafo para detectar pontos de mudança. O algoritmo foi publicado em um pacote no *CRAN* como *SpecDetec* e está disponível para uso de forma irrestrita. O *SpecDetec* foi avaliado utilizando o *UCR Archive* que é uma grande base de dados de diferentes séries temporais. A performance do *SpecDetec* foi comparado com outros algoritmos que representam o estado da arte na detecção de pontos de mudança. Os resultados mostraram que agrupamento com espectro do grafo é uma técnica eficiente para detecção de pontos de mudança, pois o *SpecDetec* alcançou uma exatidão melhor em comparação ao estado da arte em alguns cenários específicos e tão eficiente quanto na maioria dos casos avaliados. Em contextos onde é possível suportar uma tolerância de até 0.05 o *SpecDetec* é recomendado pois se mostrou superior aos outros algoritmos na maioria das bases de dados.

Palavras-chave: Séries Temporais. Inteligência artificial. Detecção de ponto de mudança. Espectro do grafo.

ABSTRACT

UZAI, Luis Gustavo. TITLE. 103 f. Dissertação De Mestrado – Programa de Pós-graduação em Informática, Universidade Tecnológica Federal do Paraná. Cornélio Procópio, 2020.

Time series are a sequence of values distributed over time. Analyzing time series is important in many areas, including medical, financial, aerospace, commercial, and entertainment. Change point detection is the problem of identifying change in the meaning or distribution of data in a time series. The academic and commercial interest in the subject has been increased in the last decade due to the increase of power and complexity of sensors, as well as the advance of technological processes, which allowed the capture and recognition of a large volume of data. Many widely used algorithms find it difficult to achieve optimal results when the number of dimensions increases or the data volume grows exponentially, there are also time series where most algorithms do not have satisfactory results (over 0.80 precision) even considering two-dimensional data as in context-based distributions. Most known algorithms are more efficient in specific scenarios and less efficient in others, so a greater number of solution options increases the likelihood that the user will get an algorithm that best meets their needs. Solutions to these issues are of great ecological and economic interest. The objective of this work is the development of an unsupervised change point detection algorithm that is applicable in series with multiple change points and large data volume with high precision. To achieve this goal, the new SpecDetec method was developed, an algorithm that uses graph spectrum clustering to detect shift points. The algorithm was published in a package in *CRAN* as *SpecDetec* and is available for unrestricted use. The *SpecDetec* was evaluated using the *UCR Archive* which is a large database of different time series. The performance of *SpecDetec* has been compared with other state-of-the-art algorithms for detecting shift points. The results showed that graph spectrum clustering is an efficient technique for detecting shift points, as Spec has achieved better accuracy compared to the state of the art in some specific scenarios and is as efficient as in most cases evaluated. In contexts where it is possible to support a tolerance of up to 0.05, *SpecDetec* is recommended as it was superior to other algorithms in most databases.

Keywords: Time series data. Artificial intelligence. Change point detection. Graph spectral.

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE SIGLAS

VP	Verdadeiro Positivo
FP	Falso Positivo
FN	Falso Negativo
VN	Verdadeiro Negativo
AC	Acurácia
SB	Sensibilidade
SG	Singularidade
TFP	Taxa de Falso Positivo
TFN	Taxa de Falso Negativo
TCE	Taxa de Classificações Erradas
PC	Precisão
SIG	Significância
CM	Classificação Média

LISTA DE SÍMBOLOS

T	Série temporal
T_n	Posição de uma anotação da série temporal de 0 a *n*
y	Conjunto de anotações de uma série temporal
Q	Número máximo de pontos de mudança na série temporal
$Cj\tau$	Conjunto de pontos de mudança
$C(.)$	Medida de ajuste parametrizável - ex: função de suavização
τ^i	Posição de um potencial ponto de mudança
K	Valor constante parametrizado
$R_G(t)$	Conjunto da posição de extremidades
τ	Posição do ponto de mudança na série temporal

SUMÁRIO

1 INTRODUÇÃO

A análise de séries temporais é importante em muitas áreas, incluindo médica, financeira, aeroespacial, empresarial e meteorológica (AMINIKHANGHAHI; COOK, 2016). Séries temporais são um conjunto de valores dispersos ao longo do tempo, que descrevem um comportamento particular de um sistema dinâmico. Em pontos de tempo arbitrários uma série temporal pode alterar seu comportamento em relação ao restante da série já observada, esses pontos são chamados de *Change Points* (Pontos de Mudança) (BASSEVILLE; NIKIFOROV, 1993) .

Change Point Detection (Detecção de Ponto de Mudança/CPD) é o problema em identificar a mudança na distribuição ou significado dos dados em uma série temporal (BASSEVILLE; NIKIFOROV, 1993) (AMINIKHANGHAHI; COOK, 2016), segmentação, detecção de bordas, detecção de eventos ou de anomalias são conceitos similares que ocasionalmente são aplicados ao problema de detecção de pontos de mudança. Com o passar dos anos, diversas técnicas para a detecção de pontos de mudança foram desenvolvidas. Historicamente entre as décadas de 1920 e 1930 surgiram as primeiras técnicas de detecção de ponto de mudança (PAGE, 1955). Desde então, é uma questão de crescente interesse dado ao aumento de alcance e complexidade de sensores e avanço de processos tecnológicos, os quais possibilitam a captura e reconhecimento de um grande volume de dados (LI; XU; ZHAO, 2015).

Com o crescimento no recolhimento dos dados, aplicações estão sendo desafiadas a avaliar dados com um volume de anotações e dimensionalidades cada vez maior, como citado por (CHEN; ZHANG, 2015) o trabalho de (EAGLE; PENTLAND; LAZER, 2009) demonstrou que a conectividade de dados em redes pessoais são cada vez mais comum, incluindo e-mail, telefone, registros de chat, e conexões em redes sociais como Twitter ou *Facebook* podem ser usadas para construir uma rede de interações social entre indivíduos. Essa constatação é verdadeira em diversos cenários, como análise de textos ou sequências de *DNA* (TSIRIGOS; RIGOUTSOS, 2005) ou análise de imagens de vigilância, climatologia ou neurociência, para detectar eventos como falhas de segurança, tempestades ou atividades cerebrais. Em todos esses casos além de um grande volume de informação uma série de características devem ser observadas (várias dimensões).

Outro desafio na detecção de pontos de mudança é no cenário de *Big Data*, visto que, frequentemente os dados possuem anomalias e a maioria dos algoritmos atuais como o PELT não são resilientes a anomalias que alteram a distribuição da série (JAMES; KEJARIWAL; MATTESON, 2016). Anomalias são observações fora do padrão dos dados, como ruídos ou valores irregulares a distribuição da série que dificultam a identificação de padrões estatísticos da série temporal (JAMES; KEJARIWAL; MATTESON, 2016). Outro problema é que frequentemente os dados não possuem uma distribuição normal, o que implica na limitação da aplicação de diversos algoritmos que mostram eficiência em contextos paramétricos (KILLICK; FEARNHEAD; ECKLEY, 2012).

Existem várias abordagens para a detecção de pontos de mudança de categorias distintas. Cada método possui estratégias diferentes para contornar os problemas descritos anteriormente e detectar pontos de mudança com maior precisão (AMINIKHANGHAHI; COOK, 2016). A seleção de um método deve considerar primordialmente a natureza dos dados analisados. Algoritmos produzidos na última década tem como objetivo principal a detecção de múltiplos pontos de mudança em grandes conjuntos de dados em baixo tempo (KILLICK; FEARNHEAD; ECKLEY, 2012) (MATTESON; JAMES, 2014) (JAMES; KEJARIWAL; MATTESON, 2016). Entretanto, conjuntos de dados de alta dimensionalidade são problemáticos para a maioria dos algoritmos atuais, já que, a medida que aumenta a dimensionalidade a precisão dos algoritmos diminui e o tempo para realizar o cálculo aumenta consideravelmente (CHEN; ZHANG, 2015).

A detecção de pontos de mudança utilizando grafos foi proposto por Chen (CHEN; ZHANG, 2015) com o propósito de detectar com precisão pontos de mudança em bases de dados de alta dimensionalidade sem grande perda de precisão. O método *gSeg* proposto por Chen apresentou a mesma precisão que o estado da arte em bases de baixa dimensionalidade, contudo, obteve melhor precisão que o estado da arte em bases de média para alta dimensionalidade. O método de Chen não explora a topologia dos grafos gerados a partir das séries temporais, este trabalho apresenta a hipótese de que é possível utilizar teorias fundamentais dos grafos para recolher maiores informações da série temporal e conseguir detectar pontos de mudança de forma mais assertiva sendo mais resiliente a anomalias apresentadas na série temporal.

Este trabalho faz uso da teoria espectral do grafo com a finalidade de obter o conhecimento da topologia dos grafos gerados a partir das séries temporais multi-valoradas. A teoria espectral do grafo é descrita pelo estudo dos autovalores e autovetores de matrizes associadas a grafos (SPIELMAN, 2007). O método criado nesse trabalho *SpecDetec* (abreaviação de *Graph Spectrum*) utiliza o espectro do grafo a fim de separar os vértices de forma que o ponto limite das separações seja o ponto de mudança. O uso do espectro do grafo para o agrupamento e a separação de dados se mostrou mais eficiente que os métodos tradicionais em diversos cenários (NG; JORDAN; WEISS, 2002). Métodos tradicionais como o *K-means* tendem a falhar quando os grupos não correspondem a regiões convexas. Apesar do uso do espectro do grafo para agrupamento ser comum, no melhor do nosso conhecimento, não existem trabalhos descrevendo o uso do espectro dos grafos para detecção de pontos de mudança.

A eficiência do método proposto foi comparada com o estado da arte na detecção de pontos de mudança, *AMOC*, *BinSeg*, *SegNeigh*, *PELT* (KILLICK; ECKLEY, 2013), *EDivisive* (MATTESON; JAMES, 2014), *EDM* (JAMES; KEJARIWAL; MATTESON, 2016) , *MFT* (MESSER et al., 2014), *Breakfast* (FRYZLEWICZ, 2016) e *gSeg* (CHEN; ZHANG, 2015) o qual obteve maior precisão em diferentes cenários, como verificado nas bases de dados *BeetleFly*, *BirdChicken*, *Meat* e *ShapeletSim* onde o *SpecDetec* obteve maior assertividade na detecção de pontos de mudança. Além disso, como resultado geral, possui uma média de precisão (0.624) levemente superior a média geral de todos os algoritmos(0.613). Outro resultado interessante obtido é que, considerando uma taxa de erro na precisão de 0.05 como aceitável, o SpecDetec tem resultado médio superior a todos os outros algoritmos na maioria dos casos. Este trabalho indica experimentalmente que o espectro do grafo é uma alternativa válida para detecção de múltiplos pontos de mudança.

1.1 MOTIVAÇÃO

Originalmente a motivação deste trabalho foi identificar melhores algoritmos de detecção de pontos de mudança para o contexto de análise genômica de variação do número de cópias de genes (ZARREI et al., 2015) para detecção de doenças degenerativas. Porém, ao analisar o estado da arte foi constatado que, detecção de pontos de mudanças tem uma série de aplicações práticas (AMINIKHANGHAHI; COOK, 2016) além disso, é um tema de crescente interesse acadêmico. Em especial na última década com melhorias constantes dos métodos existentes (MATTESON; JAMES, 2014) (CHEN; ZHANG, 2015), grandes corporações como o Twitter utilizam métodos de detecção de pontos de mudança diariamente (JAMES; KEJARIWAL; MATTESON, 2016), logo, o foco do trabalho foi entender a abrangência das detecções de pontos de mudança e identificar possíveis melhorias.

Foi esclarecido ao decorrer da revisão bibliográfica quais os desafios dos métodos de detecção de pontos de mudança, grande volume de dados, multi dimensões e anomalias presentes nas séries temporais são desafios ainda presentes para os métodos, por isso, é necessário pesquisar técnicas que possam trazer melhores resultados diante desses complicantes. Então foi proposto verificar técnicas aplicadas com sucesso em outras áreas da computação, como o reconhecimento de imagem, podem ser adaptadas e utilizadas no contexto de detecção de pontos de mudança, já que, algoritmos de reconhecimento de imagem deduzem a imagem em valores numéricos que representam suas propriedades, se tornando semelhante a representação de séries temporais.

1.2 OBJETIVOS

O objetivo deste trabalho é investigar o agrupamento de espectro do grafo como metodologia computacional viável para a detecção de múltiplos pontos de mudança em uma série temporal. Deseja-se verificar a possibilidade de um método capaz de ser executado com um grande volume de dados de forma *offline* e que seus resultados possam ser facilmente medidos.

Como objetivos específicos têm-se:

- Verificar se o agrupamento por espectro do grafo pode ser uma opção eficiente para detecção de pontos de mudança em determinados contextos.
- Desenvolver de uma aplicação para o uso para detecção de pontos de mudança.
- Comparar diversos algoritmos de detecção de pontos de mudança e demonstrar o melhor para cada contexto.

1.3 ORGANIZAÇÃO DO TEXTO

O presente trabalho se organiza da seguinte forma:

- O Capítulo 2 os conceitos usados no trabalho são fundamentados, apresentando a definição de detecção de pontos de mudança, o processo de detecção comum no estado da arte, demonstração de uma série de aplicações práticas envolvendo a detecção de pontos em

diversos campos diferentes, uma explicação sobre os métodos de detecção, suas categorias e propriedades, assim como, o funcionamento dos métodos que serão comparados com o método proposto. Também são apresentados nesse capítulo uma introdução sobre a problemática de anomalias em séries temporais, por fim, uma introdução sobre os grafos e a detecção de pontos de mudança com grafos e o conceito necessário para criação do algoritmo proposto, espectro do grafo.

- No Capítulo 3 é apresentado o método proposto de detecção de pontos de mudança com o uso de agrupamentos com o espectro do grafo.

- O Capítulo 4, são apresentados todas as formulas para avaliação dos algoritmos, também são apresentados as bases de dados selecionadas e os critérios de seleção das bases e das métricas de avaliação, são demonstrados os resultados do novo método proposto em relação aos outros algoritmos, usando as métricas e bases de dados apresentadas também nesse capitulo, são realizadas discussões sobre os resultados com conclusões que serviram de orientação para os trabalhos futuros.

- No Capítulo 5, são apresentadas as conclusões baseadas nos resultados durante o desenvolvimento do trabalho, as contribuições acadêmicas e propostas para trabalhos futuros.

2 FUNDAMENTAÇÃO TEÓRICA

Nessa seção são apresentados conceitos e fundamentos relacionados ao trabalho, como detecção de pontos de mudança e agrupamento com grafos.

2.1 DETECÇÃO DE PONTOS DE MUDANÇA

O problema de detecção de pontos de mudança, caracteriza-se por detectar alterações de padrão em uma determinada série temporal, mais precisamente, o momento quando o padrão começou a mudar. Existem 4 categorias de detecção de anomalias em séries temporais, são elas (CHANDOLA; BANERJEE; KUMAR, 2009):

- Ponto de mudança: Um único ponto na série está anormal em relação ao resto dos dados. Um exemplo de ponto único é a Figura 1, onde, a série temporal é dividida em duas camadas, antes e após o ponto de mudança e, o ponto de mudança é destacado por um traço vertical.
- Mudança Contextual: Uma instância de dados que é considerada uma anomalia devido ao contexto do *dataset* (conjunto de dados), como baixa temperatura no verão, nesse tipo de problema é necessário ter informações sobre o contexto.
- Mudança de Sequência: Dados considerados fora do padrão dentro de uma determinada sequência, porém, se avaliados individualmente, não são considerados mudanças de padrão, como exemplo alta constante por vários dias na bolsa de valores. A Figura 2 exibe um exemplo de detecção de ponto de mudança por sequência, assim como, com a presença de sazonalidade representada na Figura 3.
- Mudança de Série Temporal: Série temporal considerada uma mudança de padrão considerando um conjunto de séries temporais (*dataset*).

Existem diferentes tipos de problemas envolvendo detecção de pontos de mudança, e para cada um, existe uma melhor abordagem para solução (AMINIKHANGHAHI; COOK, 2016):

- Algoritmos não supervisionadas, nos quais as mudanças não são rotuladas e a partir de medidas de similaridade os pontos discrepantes são separados pelo algoritmo
- Algoritmos supervisionados, todos os pontos fora do padrão estão rotulados e, agentes são treinados para identificar essas mudanças e aplica-las em conjuntos não rotulados.

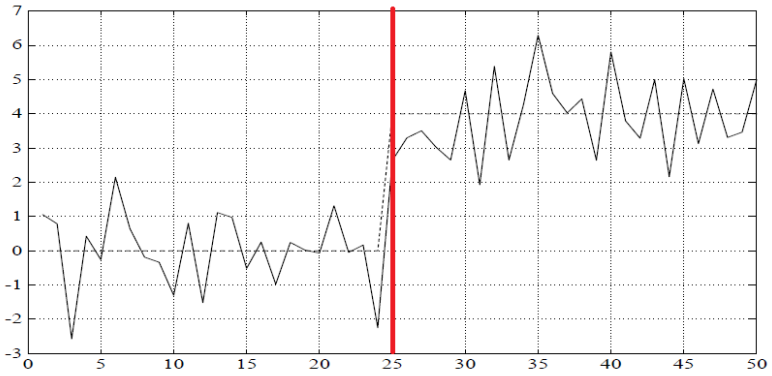


Figura 1: Série temporal com ponto de mudança único identificado.

Fonte: (BASSEVILLE; NIKIFOROV, 1993)

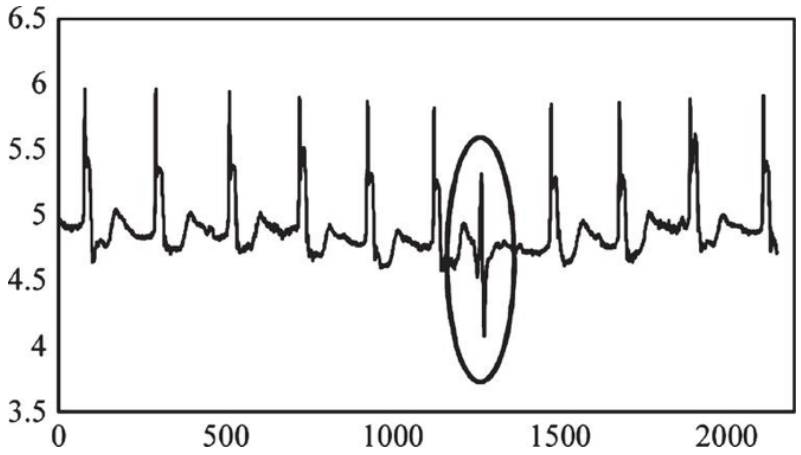


Figura 2: Série temporal com anomalia com base na sequência.

Fonte: (CETINKAYA-RUNDEL et al.,)

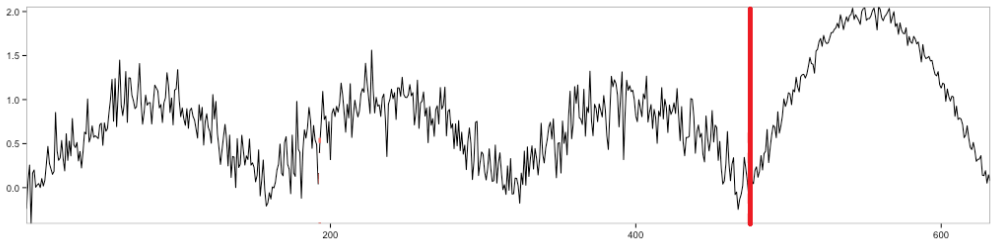


Figura 3: Detecção de ponto de mudança com séries temporal sazonais.

Fonte: (MAGAKIN, 2016)

2.1.1 PROCESSO DE DETECÇÃO

Para a detecção de pontos de mudança em qualquer categoria, comumente, são executados os seguintes passos (sequenciais) (CHEBOLI, 2010):

1. Transformação da Série Temporal: O conjunto de dados é modificado a fim das mudanças de padrão serem realçadas ou, que falsas mudanças possam ser desconsideradas.

2. Avaliação do Problema: Determinar qual técnica, algoritmo ou sistema serão melhores para o problema de aplicação.

3. Aplicação: Aplicação de uma ou mais técnicas e análise dos dados gerados, verificação das medidas de desempenho (acurácia) das técnicas selecionadas.

4. Revisão: Analise e conclusão com base nos resultados. Nesse passo, os resultados dependem diretamente da área de atuação do problema (segurança, biomédica, detecção de fraude, etc).

Existem diferentes tipos de problemas envolvendo detecção e para cada um existe uma melhor abordagem explicado na sessão 2.1.4.

2.1.2 APLICAÇÕES PRÁTICAS

Com o passar dos anos, diversas técnicas para a detecção de pontos de mudança foram desenvolvidas. Historicamente entre as décadas de 1920 e 1930 surgiram as primeiras técnicas de detecção de ponto de mudança (PAGE, 1955). Desde então, é uma questão de crescente interesse acadêmico. Esse crescimento pode ser caracterizado pelo aumento de potência e complexidade de sensores e avanço de processos tecnológicos, que possibilitam a captura e reconhecimento de um grande volume de dados, aumentando as possibilidades de pesquisa para essas áreas, soluções para essas questões são de grande interesse ecológico e econômico (BASSEVILLE; NIKIFOROV, 1993) (LI; XU; ZHAO, 2015).

Nos últimos 20 anos, houve um crescimento significante de pesquisas problemas no mundo real referente as seguintes questões (BASSEVILLE; NIKIFOROV, 1993)(AMINIKHANGHAHI; COOK, 2016):

- Detecção e diagnóstico de falhas (monitoramento);
- Controle de qualidade;
- Previsão de eventos catastróficos naturais (terremotos, tsunami, etc.);
- Monitoramento médico;
- Monitoramento de imagens em câmeras de segurança.

Uma possível solução para os problemas apresentados anteriormente é a detecção de pontos mudança, pois, mudanças de padrão frequentemente representam mudanças de estados e, identificar quando há uma mudança de estado leva a um entendimento mais profundo do problema, o que é essencial para a tomada de decisão (KAWAHARA; SUGIYAMA, 2009).

Detecção e diagnóstico de falhas: Detectar e diagnosticar falhas o mais rápido possível pode ser de grande vantagem principalmente quando considerado sistemas dinâmicos, pois, pode gerar ganho no aumento de confiabilidade do equipamento, assim como, redução de risco

de paradas não programadas de produção reduzindo perdas materiais e acidentes de trabalho (MOREIRA, 2011).

Controle de qualidade: uma das primeiras aplicações envolvendo pontos de mudança é no controle de qualidade e monitoramento da produção (BASSEVILLE; NIKIFOROV, 1993). É possível exemplificar um uso hipotético de uma fábrica de bolachas e uma câmera que analisa a cor das bolachas recém assadas em uma esteira. Naturalmente, algumas bolachas estarão mais escuras e outras mais claras, porém, caso a média das bolachas esteja mais escura que o padrão anteriormente registrado (ponto de mudança encontrado) é possível que o forno precise diminuir a temperatura. O inverso também é verdadeiro caso a média das bolachas esteja mais clara o forno possivelmente precisará aumentar a temperatura. O exemplo hipotético mencionado pode ser generalizado para uma série de cenários, por muitas razões é compreensível que atributos de determinado produto estejam dentro de padrões com desvios mínimos para a garantia da qualidade (LAI, 2014).

Previsão climática: Nas últimas décadas cada vez mais a informação climática vem sendo armazenada tornando-se necessária técnicas para minerar e entender os dados (ITOH; KURTHS, 2010). Assim, pontos de mudança são utilizados para conhecer as alterações de padrões em séries temporais de informação climática e, categorizar motivos pelos quais essas mudanças ocorrem (YONETANIL; MCCABE, 1994) (ITOH; KURTHS, 2010). Monitorar dados climáticos é de grande interesse acadêmico já que, uma grande diversidade de eventos climáticos possuem relação com as mudanças de temperatura constatadas nas últimas décadas (BONSAL et al., 2001). Nesse sentido, diversas técnicas de detecção de pontos mudanças são aplicadas para identificar oscilações de padrões climáticos, inclusive, são comparados entre si com a finalidade de indicar o melhor algoritmo para cada cenário. (REEVES et al., 2007) (ITOH; KURTHS, 2010) (DUCRÉ-ROBITAILLE; VINCENT; BOULET, 2003).

Monitoramento médico: Para garantir a saúde de um paciente frequentemente é necessário o monitoramento de uma série de sinais vitais que indicam a evolução de um quadro clínico. Os sinais vitais são variáveis fisiológicas como: frequência cardíaca, eletrocardiograma, eletroencefalograma e entre outros, podendo ser transpostos em séries temporais (AMINIKHANGHAHI; COOK, 2016). O monitoramento médico necessita de métodos que fornecem suporte rápido ao detectar mudanças sendo aplicado em casos críticos como medição cardíaca de crianças anestesiadas (YANG; DUMONT; ANSERMINO, 2006). E também, em análise mais longas e detalhadas, por exemplo, ao verificar a variação cardíaca durante o sono é necessário horas de coleta de dados para que seja possível obter resultados válidos (MALLADI; KALAMANGALAM; AAZHANG, 2013) (STAUDACHER et al., 2005) (BOSC et al., 2003).

Monitoramento de imagens em câmeras de segurança: Pesquisadores e profissionais coletam dados e imagens ou vídeos para propósitos de vigilância. A detecção de eventos abruptos, como falhas de segurança, pode ser identificada através de detecção de pontos de mudança (AMINIKHANGHAHI; COOK, 2016).

É possível verificar em (RADKE et al., 2005) que existe uma vasta variedade de técnicas e interesse acadêmico em detecção de mudanças em análise de vídeos e que, apesar do vasto número de trabalhos relacionados na área, ainda é um tema de interessante para futuras pesquisas, já que, diversas lacunas ainda não foram preenchidas e, os algoritmos precisam melhorar o tempo de execução e precisão (RADKE et al., 2005).

2.1.3 ANOMALIAS EM SÉRIES TEMPORAIS

Um problema comum com Detecção de Pontos de Mudança são anomalias presentes nas bases de dados. Anomalias podem ser representadas por mudanças abruptas na série de dados que não caracterizam pontos de mudança, pois, o padrão da série temporal volta a ser o mesmo após a anomalia (VALLIS; HOCHENBAUM; KEJARIWAL, 2014), ou, valores ruidosos na série temporal que dificultam a identificação de um padrão (CHEBOLI, 2010). Anomalias são problemáticas pois, criam um desvio na distribuição dos algoritmos dificultando a identificação dos pontos de mudança reais, também torna o algoritmo suscetível a falsos positivos, identificando a anomalia como um ponto de mudança real (VALLIS; HOCHENBAUM; KEJARIWAL, 2014). Essas anomalias também podem ser definidas como *outliers*(valores isolados não representativos) (CHEBOLI, 2010).

Anomalias em uma série temporal podem existir por diversos motivos, como erros de medição, valores aberrantes muito diferentes da média da série temporal, series com distribuição altamente variante, como anotações climáticas (DUCRÉ-ROBITAILLE; VINCENT; BOULET, 2003). A origem das anomalias pode alterar a técnica necessária para obter melhor resultados de detecção, porém, este trabalho tratará de anomalias de forma genérica.

Anomalias são normalmente tratadas através de pré-processamento na base de dados que contém as séries temporais, o pré-processamento pode aumentar consideravelmente a precisão dos algoritmos, porém, também pode deformar as séries temporais da base de dados de forma que pontos de mudança reais se tornem mais difíceis de serem detectados. Esse processo é conhecido como transformação dos dados, além do tratamento de anomalias, transformação nos dados pode ser utilizado para aumentar a performance dos algoritmos de reconhecimento de pontos de mudança (CHEBOLI, 2010).

Existem diferentes técnicas de transformação de dados para lidar com anomalias, como, agrupar um conjunto de dados similares em um único dado representante (CHAKRABARTI et al., 2002), ou, discretização, cujo o objetivo é converter dados da série temporal em valores discretos em uma sequência de um determinado alfabeto finito. Apesar dos riscos de perda de informação, é possível utilizar algoritmos que funcionam de forma mais eficiente e performática com valores discretos (WEI et al., 2005), assim como outras alternativas (MA; PERKINS, 2003).

2.1.4 MÉTODOS DE DETECÇÃO

Nessa subseção é apresentado os tipos e categorias dos métodos de detecção de pontos de mudança, assim como, o funcionamento dos métodos usados para comparação com o método proposto.

2.1.4.1 SUPERVISIONADOS E NÃO SUPERVISIONADOS

Os algoritmos de aprendizagem supervisionados utilizam de aprendizagem de máquina em um conjunto rotulado de dados para identificar características ou regras que melhor representam o problema, afim de gerar uma série de passos que possa ser aplicada em conjuntos não rotulados de dados (AMINIKHANGHAHI; COOK, 2016). Apesar de muito utilizados, principalmente para identificar os padrões de distribuição de uma série temporal, que é um das principais difi-

culdades na detecção de pontos de mudança, existe uma variedade menor de métodos supervisionados em relação aos métodos não supervisionados (POLUNCHENKO; TARTAKOVSKY, 2012) (TRIPATHY; SAHOO, 2015).

Os algoritmos de aprendizagem não supervisionados normalmente são usados para descobrir padrões em dados não rotulados. No contexto de Detecção de Pontos de Mudança, esses algoritmos podem ser usados para segmentar dados de séries temporais, encontrando pontos de mudança com base em características estatísticas dos dados. A segmentação não supervisionada é atrativa porque pode lidar com uma variedade de situações sem requerer formação prévia para cada situação (AMINIKHANGHAHI; COOK, 2016) (TRIPATHY; SAHOO, 2015). Existe uma variedade de algoritmos não supervisionados utilizados em detecção de pontos de mudança, e assim como os supervisionados, os resultados dos métodos depende da área de aplicação (AMINIKHANGHAHI; COOK, 2016).

2.1.4.2 PARAMÉTRICOS E NÃO-PARAMÉTRICOS

Uma abordagem paramétrica faz suposições sobre os parâmetros (propriedades definidoras) da distribuição dos dados (série temporal), enquanto a abordagem não paramétrica não faz premissas, algoritmos não paramétricos tem a vantagem de poder ser usados em bases sem uma distribuição normal, entretanto, devem manter todos os dados para classificar os pontos de mudança (aumentando o consumo de memória), enquanto algoritmos paramétricos podem manter apenas informações sobre os parâmetros dos dados de teste para detecção do ponto de mudança (RAYKAR, 2007).

A distinção entre abordagens paramétricas e não paramétricas é importante porque as abordagens não paramétricas demonstraram maior sucesso em conjuntos de dados massivamente grandes. Além disso, o custo computacional dos métodos paramétricos é maior que as abordagens não paramétricas e também não se dimensiona com o tamanho do conjunto de dados (CHEN; ZHANG, 2015).

2.1.4.3 ONLINE E OFFLINE

Os algoritmos de detecção de pontos de mudança são tradicionalmente classificados como *online* ou *offline*. Os algoritmos *offline* consideram todo o conjunto de dados ao mesmo tempo, e avaliam um conjunto finito e fechado de dados para determinar onde a mudança ocorreu. Em contraste, os algoritmos *online* ou *real time* (em tempo real) são executados simultaneamente com o processo que estão monitorando, processando cada ponto de dados à medida que ele se torna disponível, com o objetivo de detectar um ponto de mudança o mais rápido possível após a ocorrência, idealmente antes da próxima anotação chegar (DOWNEY, 2008).

A maioria dos algoritmos de detecção de pontos de mudança desenvolvidos atualmente são *offline* (AMINIKHANGHAHI; COOK, 2016). Um uso comum de algoritmos *offline* são em detecção de mudanças em bases biológicas, como por exemplo, análise genômica de variação do número de cópias de genes (ZARREI et al., 2015), que pode proporcionar uma série de aplicações médicas, como identificação de células cancerígenas (GIANNOUDIS et al., 2017) e doenças relacionadas a demência (BUTCHER et al., 2017).

Um exemplo de problema que exige algoritmos *online* é aproveitamento de dados da nuvem, já que diferentes estratégias de *cacheamento* (armazenamento de dados já buscados

em memória) e liberação de recursos podem influenciar a latência da rede e consequentemente a experiência dos usuários, por isso, grandes sites como o *Twitter* utilizam ferramentas para identificar mudanças na rede e alternar os recursos da nuvem para cada mudança real, no caso do *Twitter* é utilizado o algoritmo *online EDM* que toma a decisão conforme novas entradas são alimentadas (JAMES; KEJARIWAL; MATTESON, 2016).

Na prática, nenhum algoritmo de detecção de ponto de mudança opera em tempo real pois deve inspecionar novos dados antes de determinar se ocorreu um ponto de mudança entre as anotações existentes e as novas reconhecidas. No entanto, diferentes algoritmos *online* requerem diferentes quantidades de novas anotações antes que a detecção do ponto de mudança possa ocorrer.

2.1.4.4 UNI OU MULTI DIMENSIONAL

A maioria dos métodos de detecção de pontos de mudança, considera somente uma dimensão dos dados, pois, é um problema suficientemente complexo e uma série de problemas podem ser resolvidos avaliando uma só dimensão, porém, com o crescimento da complexidade e o aumento do recolhimento de dados, métodos que avaliam diversas dimensões foram desenvolvidos para obter melhor aproveitamento desse cenário (MATTESON; JAMES, 2014).

A Figura 4 exibe a detecção de pontos de mudança considerando várias dimensões de dados, as características exibidas são de diferentes mudanças em um sistema biológico que incluem genes, proteínas e metabólitos (OMRANIAN; MUELLER-ROEBER; NIKOLOSKI, 2015).

2.1.4.5 CATEGORIAS DOS ALGORITMOS

Os algoritmos de detecção de pontos de mudança podem ser classificados em categorias, os métodos supervisionados e não supervisionados possuem categorias baseadas na técnica aplicada para detecção, Aminikhanghahi em (AMINIKHANGHAHI; COOK, 2016) descreve detalhadamente cada uma das categorias citadas a seguir.

Supervisionados:

- Classificador Multi Classe;
- Classificador Classe Binária;
- Classificador Virtual.

Não Supervisionados:

- Razão de Probabilidade;
- Modelo de subespaço;
- Métodos Probabilísticos;
- Métodos baseados no núcleo;

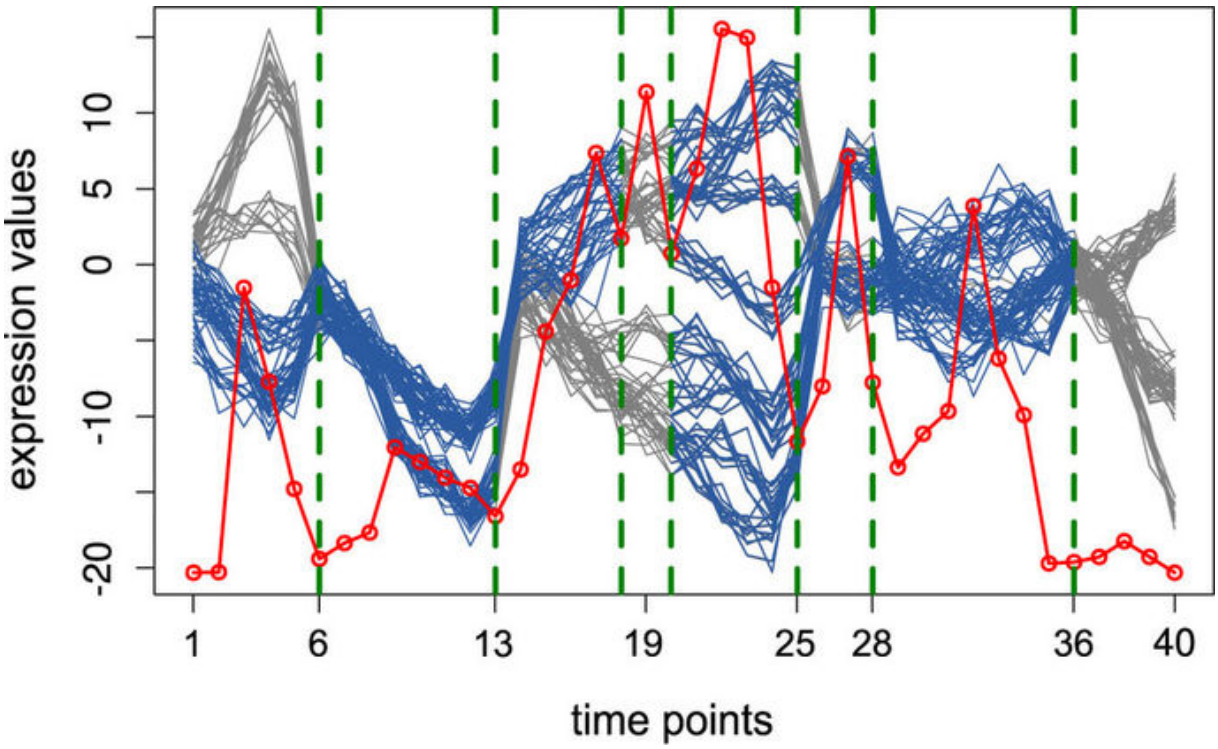


Figura 4: Detecção de Pontos de Mudança em Múltiplas Dimensões - Ilustrado como cinza as características que são ignoradas para detecção do ponto de mudança, como azul as características consideradas e a linha vermelha representa a média dos valores absolutos do coeficiente de regressão da série temporal, as linhas verticais verdes representam os pontos de mudança

Fonte: (OMRANIAN; MUELLER-ROEBER; NIKOLOSKI, 2015)

- Métodos baseado em grafos;
- Métodos de agrupamento.

2.1.4.6 ALGORITMOS RESISTENTES A ANOMALIAS

Existem sistemas desenvolvidos com a premissa de serem mais robustos a anomalias, como é o exemplo do *Breakout Detection* criado pela equipe do Twitter, foi desenvolvido para ser aplicado no contexto de *cloud data*, onde, anomalias são um problema frequente e, os algoritmos atuais não são robustos o bastante para lidar com elas (JAMES; KEJARIWAL; MATTESON, 2016). O algoritmo se mostrou promissor em detectar mudanças em base de dados com alta taxa de anomalias.

O algoritmo utilizado no *Breakout Detection* é *EDM* (E-Divisive with Medians), variação do algoritmo *E-Divisive* (MATTESON; JAMES, 2014) focado na detecção de múltiplos pontos de mudança, comparando os resultados do *EDM* com o algoritmo *PELT* (KILLICK; ECKLEY, 2013) que também resiliente a anomalias e com o *E-Divisive*, o Twitter Detection se demonstrou superior e obteve resultados promissores em um tempo computacional baixo $O(n \log n)$.

2.2 MÉTODOS AVALIADOS

A seguir serão descritos os métodos que serão avaliados em conjunto com o método proposto, descrevendo sua natureza, contexto de aplicação e vulnerabilidades, todos os algoritmos apresentados são *opensource* e podem ser adquiridos no repositório *CRAN* ou no *GitHub*, a linguagem utilizada em todos os algoritmos é R. Por se tratar de projetos ainda em evolução é possível que os métodos sofram adaptações após a data de publicação deste trabalho e o tempo computacional ou vulnerabilidades podem mudar.

Os métodos selecionados representam uma pequena amostragem do total de métodos disponíveis (AMINIKHANGHAHI; COOK, 2016), esses foram selecionados pelos trabalhos relacionados em aplicações biológicas práticas, que tem como objetivo ser resiliente a anomalias ou mais performático, ou, que seja baseado em grafos, abordagem adotada pelo algoritmo proposto.

A Figura 5 exibe a evolução histórica dos métodos, deixando claro quais métodos foram construídos com base em outros, alguns dos métodos demonstrados na figura são somente para ilustrar suas origens e não são adotados neste trabalho.

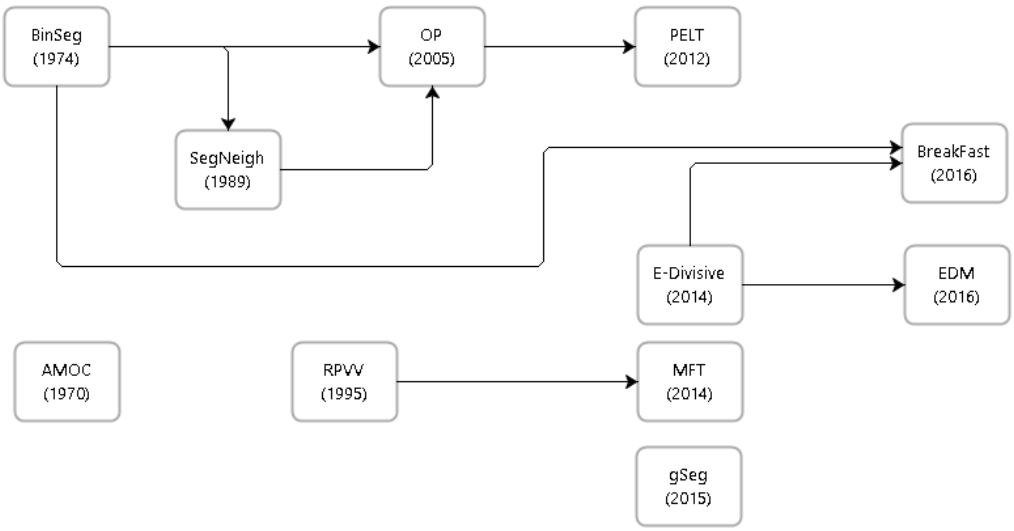
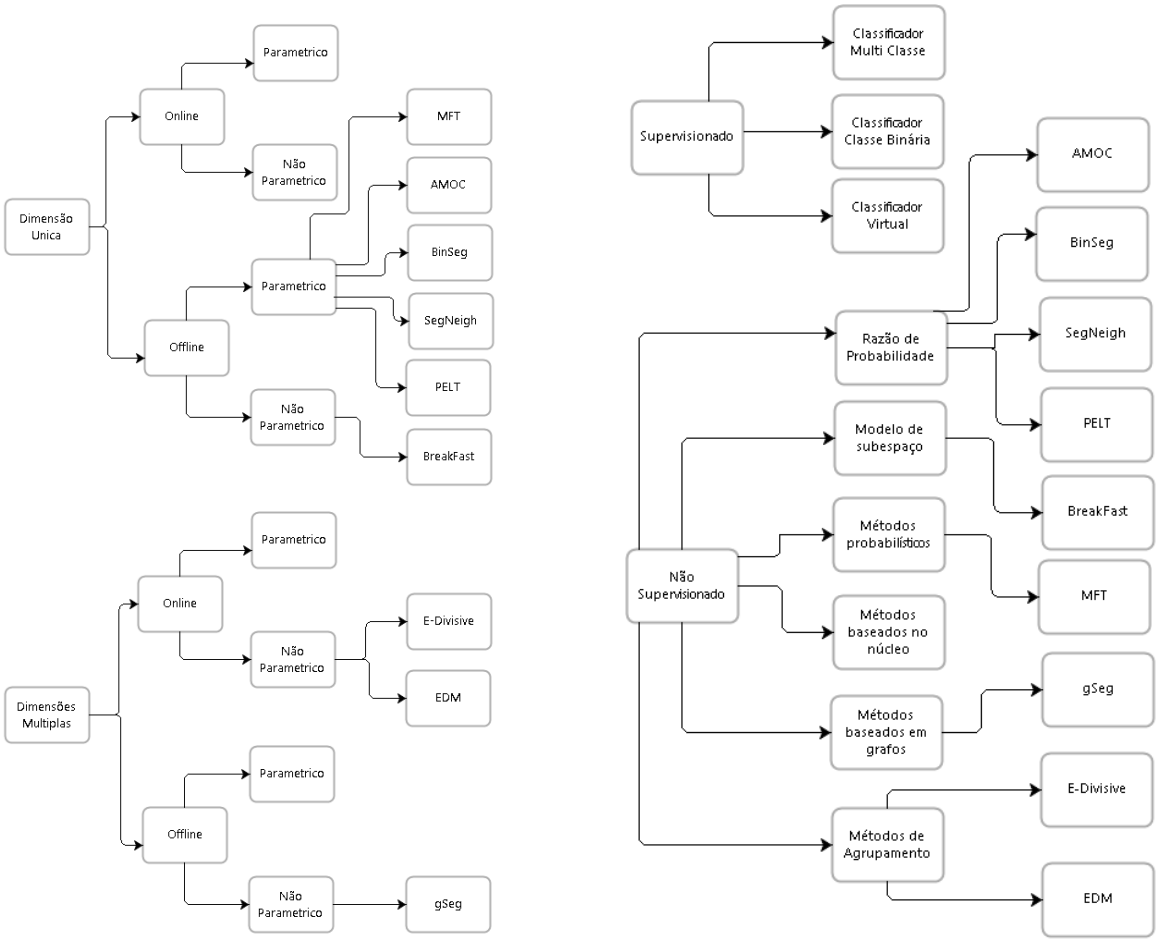


Figura 5: Evolução dos métodos selecionados através dos anos e demonstração dos métodos originados em outros

A Figura 6 - Categorias exibe a categoria de classificação dos algoritmos, como demonstrado, todos os algoritmos usados são não supervisionados.

A Figura 6 - Propriedades exibe de forma transparente a posição de cada algoritmo sobre seu desenvolvimento *Offline* ou *Online*, Multi ou Unidimensional, Paramétrico ou Não Paramétrico, apesar das posições, é possível que o algoritmo seja usado fora do seu contexto padrão, como o *PELT* em cenários de dados *Online*, porém, é possível que suas classificações sejam prejudicadas.



(a) Categorias (b) Propriedades

Figura 6: Classificação dos algoritmos usados para teste neste trabalho (AMINIKHANGHAHI; COOK, 2016)

2.2.1 AMOC

O algoritmo *AMOC*, abreviação de *At Most One Change* ou em português *Pelo Menos Uma Mudança*, da categoria *likelihood ratio* (razão de probabilística) de complexidade $\mathcal{O}(n^2)$ funciona como teste estatístico para definir que a mudança ocorreu na série temporal (T), uma determinada anotação T_n é classificada como ponto de mudança τ se o valor probabilidade máxima de $T_n(PMT_n)$ for maior do que das outras anotações na série temporal. p(.) é a função densidade de probabilidade e Θ é a razão máxima estimada por parâmetros, e y é um conjunto de anotações da série temporal $y\ \varepsilon\ T$.

A probabilidade máxima é calculada pelo resultado da seguinte formula :

$$(PMT_n) = \log p(y1 : Tn|\Theta 1) + \log p(y(Tn + 1) : n|\Theta 2) \tag{10}$$

As principais limitações do algoritmo tem relação com a base de dados que poderá ser aplicada, primeiro, espera-se que a base tenha uma distribuição normal, o que limita a aplicação em cenários online, onde, frequentemente os dados de estudo são não paramétricos (JAMES; KEJARIWAL; MATTESON, 2016). Outra limitação, como o próprio nome do algoritmo ex-

pressa, ele partirá do princípio que, existe pelo menos um ponto de mudança na distribuição, caso isso não seja verdadeiro, pelo menos um falso positivo será atribuído a série temporal, por essa limitação, é necessário que a base de dados seja pelo menos semi-rotulada, para que, se saiba com antecedência se a série temporal possui pelo menos uma mudança, mesmo que o resultados do algoritmo sejam eficientes em determinados contextos, ainda, deve-se ter conhecimento da base de dados para verificar se o algoritmo atenderá aos requisitos mínimos funcionais.

A versão em R do algoritmo pode ser encontrada pronta para uso no repositório *CRAN* no pacote *changepoint*, presente no mesmo pacote estão os algoritmos *BingSeg*, *SegNeigh* e *PELT* (KILLICK; ECKLEY, 2013).

2.2.2 BINSEG

O algoritmo *BinSeg*, abreviação de *Binary Segmenation*, da categoria *likelihood ratio*, de complexidade $\mathcal{O}(n \log n)$. A técnica utilizada no algoritmo foi descrita por (SCOTT; KNOTT, 1974) e consiste na separação da série temporal no máximo de grupos possíveis *threshold*, onde cada *threshold* tenha o mínimo de variação na razão probabilística (homogeneidade). O limite entre os *threshold* é definida como ponto de mudança τ. Assim como o *AMOC*, o *BinSeg* executa um teste estatístico para calcular a probabilidade da posição limite ser de fato um ponto de mudança.

O algoritmo é computacionalmente rápido, e é possível ser aplicado em bases de dados com um grande número de anotações, porém, devido à natureza do teste estatístico efetuado, são apresentados apenas aproximações dos pontos de mudança, o que pode prejudicar a acurácia em contextos onde a exatidão é crucial, assim como o *AMOC* é necessário que os dados sigam uma distribuição paramétrica, limitando ainda mais sua aplicação.

2.2.3 SEGNEIGH

O algoritmo *SegNeigh*, abreviação de *Segment Neighbourhood*, da categoria likelihood ratio, desenvolvido e descrito por (AUGER; LAWRENCE, 1989). O algoritmo é semelhante ao *BinSeg* porém utiliza técnicas de programação dinâmica para obter a segmentação ideal reusando a informação obtida em cada segmento para os próximas observações, assim a complexidade do algoritmo é maior que a do *BinSeg* sendo $\mathcal{O}(Qn^2)$ onde Q é o número máximo de pontos de mudança na série temporal

O algoritmo é computacionalmente mais lento que *AMOC* ou *BinSeg* do mesmo pacote, porém é mais preciso (KILLICK; ECKLEY, 2013), e consequentemente pode ser aplicado em contextos que exigem acurácia alta como identificação proteica para reconhecimento viral (AUGER; LAWRENCE, 1989). É valido ressaltar que algoritmos que sofrem com performance podem não ser aplicados em cenários online, pois identificar o ponto de mudança o mais rápido possível é o objetivo principal, também pode não ser aplicado em cenários onde a base de dados é muito grande já que o tempo para acabar de processar poderá ser considerável.

2.2.4 PELT

O algoritmo *PELT*, abreviação de *Pruned Exact Linear Time*, da categoria *likelihood ratio*, de complexidade $\mathcal{O}(n^2)$ porém, para uma penalidade linear $f(k) = k$, mudanças em escala, $\mathcal{O}(n)$. Proposto por (KILLICK; FEARNHEAD; ECKLEY, 2012) como uma solução para o problema de identificar múltiplos pontos de mudança em grandes bases de dados, foi desenvolvido como proposta para a detecção de pontos de mudança *offline*, porém, pode facilmente ser usado no contexto de detecção de pontos de mudança *online* graças a sua alta exatidão, como demonstrado em (JAMES; KEJARIWAL; MATTESON, 2016).

O algoritmo *PELT* é uma melhoria do algoritmo *OP Optimal Partitioning* descrito por (JACKSON et al., 2005), o *OP* foi desenvolvido como uma evolução do *BinSeg* e *SegNeight*, onde, espera-se resultados mais precisos em custo computacional baixo, seu funcionamento pode ser resumido pelo Algoritmo 1. Considera como $Cj\tau$ o conjunto de τ onde os valores não se repetem.

Algoritmo 1 Detecção de Mudanças com OP

Entrada: Um conjunto de dados da forma, $(y1, y2, ..., yn)$ quando, $y_i \ \varepsilon \ T$
Uma medida de ajuste $C(.)$ dependendo dos dados.
Uma constante de penalidade β que não depende do número ou localização dos pontos de mudança.

início
Inicialização: Seja n = tamanho dos dados e $F(0) = -\beta, C\tau(0) = NULL$
para $\tau^* = 1, ..., n$ **faça**
1. Calcula $F(\tau^*) = min_{0 \leq \tau < \tau^*} [F(\tau) + C(y_{(\tau+1):\tau^*}) + \beta]$.
2. Seja $\tau^i - arg\{min_{0 \leq \tau < \tau^*} [F(\tau) + C(y_{(\tau+1):\tau^*}) + \beta]\}$.
3. Atribui $Cj(\tau^*) = (Cj(t^i), t^i)$.

fim
fim
Saída: Pontos de mudança gravado em $Cj\tau$

O *PELT* considerou como um mecanismo de *Poda* poderia diminuir o custo computacional do *OP* como descrito no Algoritmo 2.

2.2.5 E-DIVISIVE

O algoritmo *E-Divisive* foi projetado para a detecção de pontos de mudança *online* (MATTESON; JAMES, 2014), sendo não paramétrico e aplicável a quase qualquer série temporal, tendo como proposta detecção de múltiplos pontos de mudança sem a necessidade de se informar a quantidade total ou aproximada dos pontos na série temporal, assim como, a possibilidade de trabalhar com dados multi-valorados, o que costuma ser problemático para a maioria dos algoritmos.

O *E-Divisive* utiliza uma nova metodologia baseada em *U-statistics*, da categoria *cluster based* (baseado em agrupamento), o algoritmo gera uma matriz contendo a distância entre todas as anotações, e então, divide a série temporal em dois grupos, cada um contendo a maior

Algoritmo 2 Detecção de Mudanças com PELT

Entrada: Um conjunto de dados da forma, $(y1, y2, ..., yn)$ quando, $y_i\ \varepsilon$
Uma medida de ajuste $C(.)$ dependendo dos dados.
Uma constante de penalidade β que não depende do número ou localização dos pontos de mudança.
A constante K

início

Inicialização: Seja n = tamanho dos dados e $F(0) = -\beta, Cj\tau(0) = NULL, R_1 = \{0\}$

para $\tau^* = 1, ..., n$ **faça**

1. Calcula $F(\tau^*) = min_{\tau\varepsilon R_{\tau^*}}\ [F(\tau) + C(y_{(\tau+1):\tau^*}) + \beta]$.
2. Seja $\tau^i - arg\{min_{\tau\varepsilon R_{\tau^*}}\ [F(\tau) + C(y_{(\tau+1):\tau^*}) + \beta]\}$.
3. Atribui $Cj\tau(\tau^*) = [Cj\tau(t^i), t^i]$.
4. Atribui $R_{\tau^*+1} = \{\tau^* \cap \{\tau\varepsilon R_{\tau^*}$ $: F(\tau) + C(y_{\tau+1:\tau^*}) + K < F(\tau^*)\}\}$.

fim

fim

Saída: Pontos de mudança gravado em $Cj\tau$

distância entre eles, esse processo se repete até que toda série temporal seja percorrida, a complexidade do método é $\mathcal{O}(n^2)$.

Após o algoritmo determinar o possível ponto de mudança, um teste de permutação é realizado para determinar a significância estatística do ponto em questão, prevendo falso positivos e criando uma resistência a anomalias. Pode ser encontrado uma versão do algoritmo implementada em R e disponível para uso no pacote *cpm* no repositório *CRAN* (MATTESON; JAMES, 2014).

2.2.6 EDM

O algoritmo *EDM*, abreviação de *E-Divisive with Medians*, da categoria *cluster based* (baseado em agrupamento), tem proposta uma melhora no algoritmo *E-Divisive* através da técnica estatística de *Moving Median* (Média Móvel) criada como alternativa ao *e-divisive* por (JAMES; KEJARIWAL; MATTESON, 2016) o nome comercial do pacote é *Twitter Breakout*. O teste de permutação apresenta diferenças, já que, é considerado a mediana das anotações e não mais a distância entre as observações, assim, o algoritmo tem complexidade de $\mathcal{O}(n\log(n))$, sendo computacionalmente mais rápido que seu antecessor.

O algoritmo foi desenvolvido pela equipe do Twitter para ser aplicado em bases em produção na nuvem no próprio *Twitter*, pois, o número alto de anomalias no cenário é frequente (*cloud big data*) e identificar o ponto de mudança real o mais cedo possível é o objetivo dos algoritmos de detecção de pontos de mudança nesse contexto, algoritmos desenvolvidos até então não possuíam assertividade suficiente para serem aplicados nesse cenário.

Apesar de ser eficiente para lidar com cenários com diversas anomalias, o algoritmo não consegue lidar com mudanças sazonais na série temporal, muito comum no cenário *offline*, não sendo a melhor opção para esses casos. O algoritmo pode ser encontrado na linguagem R no *GitHub* através do projeto *Breackout Detection* (JAMES; KEJARIWAL; MATTESON,

2016).

2.2.7 MFT

O algoritmo *MFT*, abreviação de *Multiple Filter Test*, da categoria *Probabilistic Methods* de complexidade $\mathcal{O}(n\log(n))$, baseado no algoritmo *RPVV Renewal Process with Varying Variance*, o algoritmo é destinado particularmente para mudanças de significância na série temporal ou mudança na taxa de variância em processos pontuais (MESSER et al., 2014).

A ideia principal da *MFT* é estender a técnica baseada em filtro (BASSEVILLE; NIKIFOROV, 1993) que que desliza duas janelas adjacentes de tamanho h e compara o número de eventos na janela esquerda e direita, (MESSER et al., 2014), o *MFT* gera múltiplas janelas para comparação de forma simultânea. Além disso, para ajustar o limiar de rejeição do teste, é utilizado um processo gaussiano, que surge como o limite do processo derivado do filtro.

O algoritmo pode ser encontrado na versão na linguagem R no pacote *MFT* no repositório *CRAN* (MESSER et al., 2014). Uma das vantagens do pacote é o *plot* de dados muito detalhado sobre os resultados, há informações sobre como os pontos de mudança foram encontrados e a natureza da série temporal, como exibido na Figura 7.

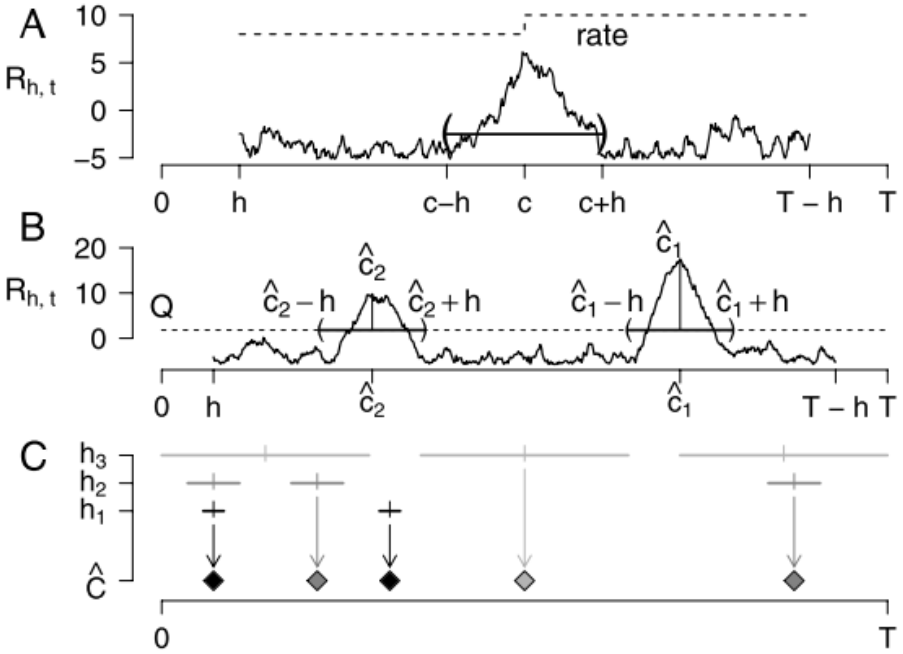


Figura 7: Detecção de pontos de mudança com *MFT* - *plot* de resultados.

Fonte: (MESSER et al., 2014)

2.2.8 BREAKFAST

O algoritmo *BreakFast* de complexidade $\mathcal{O}(T\log^2(T))$ da categoria *Subspace Modal* (Modelo de Subespaço) sendo uma proposta de detectar rapidamente diversos pontos de mudança em séries temporais de somente uma única dimensão, usando o método *Tail-Greedy Unbalanced*

Haar, explicado posteriormente. O baixo tempo computacional torna o algoritmo aplicável a cenários que exigem rápida detecção (*online*) ou grandes bases de dados (FRYZLEWICZ, 2016).

O procedimento para estimativa do número total de pontos de mudança e suas posições segue as etapas:

1. *Tail-Greedy Unbalanced Haar* (TGUH) decomposição dos dados de entrada. A transformação é multi-escala no sentido de que decompõe a série temporal em um conjunto de coeficientes de tipo-detalhe, construídos de forma adaptável, que são dispostos em uma árvore unária-binária (ou seja, que os vértices "pai"possuem 1 ou 2 "filhos"). Neste processo é construída uma base adaptativa particular *Unbalanced Haar* (UH).

2. Estágio de *Thresholding*, em que os coeficientes de detalhe cuja magnitude é inferior a um limite especificado pelo usuário são definidos como zero, desde que não prejudique a conexão da árvore de coeficiente não-zero.

3. A transformação inversa da etapa 1. Nesse estágio, o pré-estimador constante por partes τ^* é produzido, porém, é possível conter falsos pontos de mudança.

4. Etapa de pós processamento, no qual os pontos de mudança τ^* estimados com alta probabilidade de falso negativo são removidos, deixando a estimativa final de $Cj\tau$.

O algoritmo pode ser encontrado na versão na linguagem R no pacote *BreakFast* no repositório *CRAN* (FRYZLEWICZ, 2016).

2.2.9 GSEG

O algoritmo *gSeg*, abreviação de *Graph Segmenation*, da categoria *Graph Based* (Baseado em Grafos), tem como proposta a estimativa de localização de pontos de mudança (mudanças abruptas) utilizando estatísticas de varredura em uma sequência de dados, a varredura faz uso de grafos que representam a conectividade entre cada anotação da série (CHEN; ZHANG, 2015).

O algoritmo proposto é não-paramétrico sendo aplicável a quase toda série temporal desde que a medida de similaridade informada no espaço da amostra possa ser definida. É necessário fornecer ao algoritmo quais as estatísticas serão usadas na varredura baseada em grafos, assim como, as medidas de semelhança e as alternativas de intervalo. Essa abordagem mostrou melhores resultados do que o estado da arte na detecção de pontos de mudança quando a dimensionalidade dos dados é média até alta (CHEN; ZHANG, 2015).

O funcionamento do algoritmo inicia pela separação das anotações em dois grupos com um limite de observações onde $1 < n_0 \leq \tau \leq n_1 < n$, onde, o grafo é derivado de uma distância ou similaridade generalizada no espaço da amostra com extremidades que conectam observações próximas da distância. Então é executado um teste para verificar se os dois grupos são iguais em distribuição, para a identificação de múltiplos pontos de mudança o algoritmo deve ser rodado interativamente para cada novo grupo gerado.

A Figura 8 ilustra o processamento de detecção de pontos de mudança onde $R_G(t)$ é o conjunto das extremidades do grafo ou os pontos de mudança, por tanto é equivalente a $Cj\tau$. Pode ser observado na figura 8 o método sendo aplicado a um conjunto de dados artificial de

tamanho 40 com os primeiros 20 pontos desenhados de $N(0, l_2)$ e os segundos extraídos de $N((2,2)^i, l_2)$.

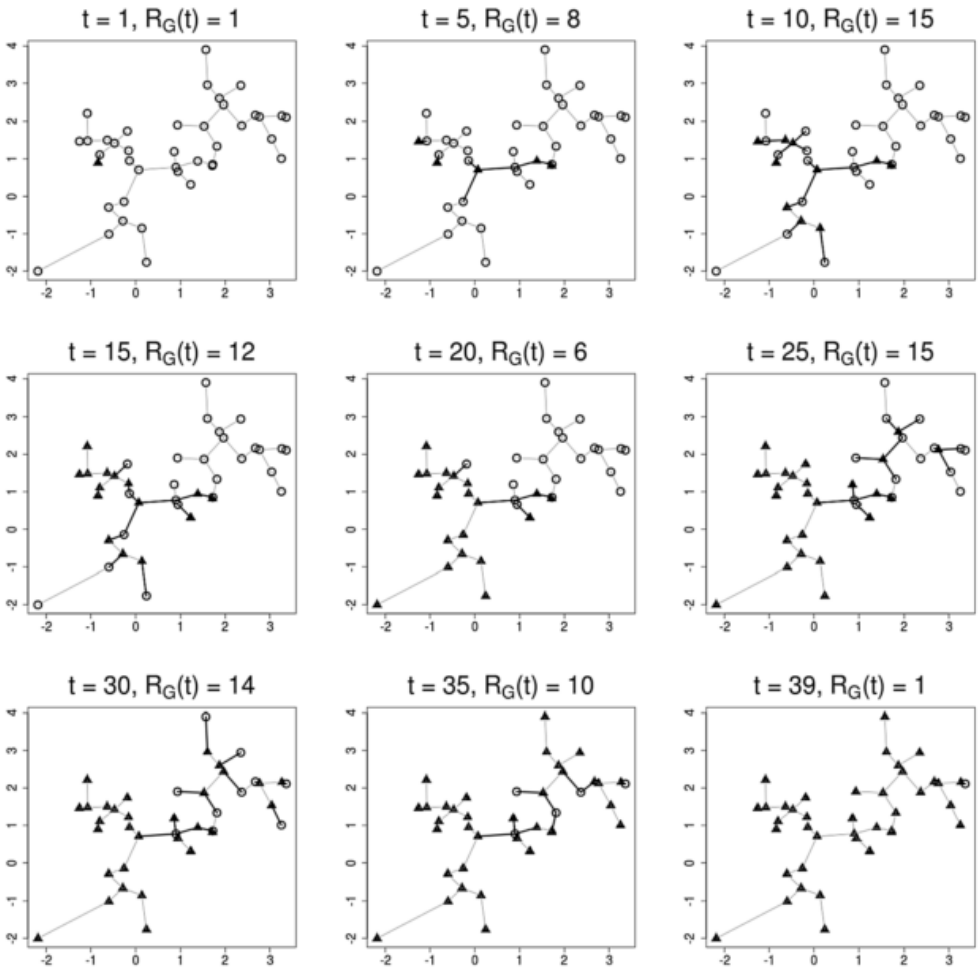


Figura 8: O cálculo de $R_G(t)$ para nove valores diferentes de t. Os dados são uma sequência de comprimento = 40, com os dois primeiros 20 pontos desenhados de $N(0, l_2)$ e os segundos extraídos de $N((2,2)^i, l_2)$. O grafo de similaridade G mostrado nas imagens é construído com Distância Euclidiana. Cada t divide as observações em dois grupos, um grupo para as observações antes de t e em t (demonstrada como triângulos) e o grupo de observações após t (demonstrada como círculos). As extremidades que conectam observações dos dois grupos diferentes (ou seja, extremidades que conectam um triângulo e um círculo) estão em negrito na imagem. Observa-se que G não muda à medida que t muda, mas as identidades de grupo de algumas observações mudam, fazendo com que $R_G(t)$ mude.

Fonte: (CHEN; ZHANG, 2015)

2.3 MÉTODOS BASEADOS EM GRAFOS

Um *Grafo* é uma estrutura G = (V, E), onde V é um conjunto finito e não vazio, cujos elementos são denominados vértices. Por sua vez, E é um conjunto de subconjuntos com dois elementos de V. Os elementos de E são chamados de arestas de G = (V, E) (FRITSCHER, 2011).

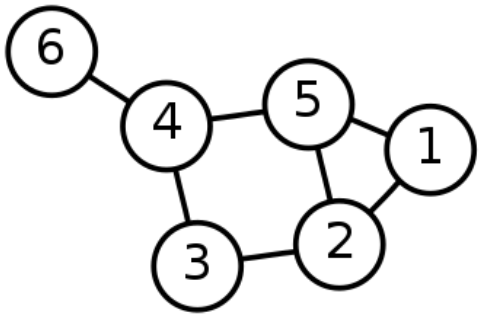


Figura 9: Exemplo de Grafo

Assim como exibido na Figura 9 os vértices de um grafo são comumente representados por um ponto ou um círculo rotulado, é comum também exibir rotulo as arestas que ligam os vértices para maior representatividade. Grafos podem ser usados para representação de uma série de cenários onde os vértices são elementos e as arestas as relações entre os elementos, por exemplo, pode representar a relação de proximidade entre cidades sendo cada cidade um vértice e as conexões entre as cidades as Arestas.

Os grafos podem ser direcionados, ou seja, quando há um caminho especifico entre os vértices através das arestas ou não direcionados quando não há restrições de caminhos e todas as conexões são válidas. Os grafos direcionados podem ser cíclicos quando há um caminho de um vértice para ele mesmo ou acíclicos quando não há (METZ, 2007).

Outra característica comum a grafos são pesos nas arestas, os pesos são úteis para diferenciar diferentes conexões, utilizando o exemplo das cidades como vértices e as arestas como a conexão entre as cidades, podem ser atribuídos pesos nas arestas para representar a distância em quilômetros entre as cidades ou ainda o tempo médio de deslocamento.

O estudo dos grafos é amplo na matemática e na área da computação, é possível verificar mais detalhes sobre o uso do grafo na computação, uma visão atualizada sobre as aplicações pode ser verificado em (PAULHEIM, 2017)

2.3.1 REDES COMPLEXAS

Redes também são uma forma de se referir a grafos, redes complexas são um tipo de grafo que apresenta uma estrutura topográfica não trivial, ou seja, estruturas que não representam um padrão regular, e tem como objetivo uma melhor representação da interação entre elementos do mundo real do que pode ser obtida com grafos simples (redes regulares) (METZ, 2007).

Ao analisar redes complexas são observadas características que não vislumbradas em grafos simples, como centralidade (vértice mais central) e conectividade (vértices com maior número de conexões) entre outras (METZ, 2007). Geralmente redes complexas estão vinculadas a ideia de grafos com muitos vértices e centenas de conexões e são exploradas principalmente para entender a natureza desses grafos (NEWMAN, 2003).

Teoria das redes complexas é a área da matemática que estuda redes complexas analisando a topografia da rede para entender as relações entre os elementos da rede, diversas propriedades podem ser medidas para elucidar como a rede foi gerada e algumas das algumas das principais características de uma rede complexa são (COSTA et al., 2007):

- *Clustering Coefficient* (Coeficiente de Aglomeração): Coeficiente de conexões ou grau de agrupamento entre os vértices vizinhos, através dele é possível verificar se os sub-grupos do grafo criar grupos de alta densidade;
- Distribuição de Graus: Graus no contexto de grafos são o número de arestas de um determinado vértice, assim, a distribuição de graus corresponde a função de distribuição probabilística dos graus de um grafo;
- Resistência: Capacidade do grafo de se manter funcional dado a remoção de vértices, a Resistência e a Distribuição de Graus está diretamente relacionada;
- Diâmetro: Maior distância de comprimento entre vértices;
- Borda: Borda ou extremidade é o ponto mais distante entre dois grafos ou entre dois sub-grafos;

Há uma série de aplicações práticas em diversas áreas com redes complexas, já que quase toda organização estrutural de relacionamento pode ser sumarizada em grafos (NEWMAN, 2003). Textos são exemplos de redes complexas, considerando as palavras como os vértices e as arestas como conexão entre vértices, é possível avaliar a qualidade de textos usando medidas estatísticas para identificar a correlação entre as palavras do texto (ANTIQUEIRA et al., 2005). Um problema crescente nas grandes cidades é a qualidade do tráfego urbano, definir as melhores rotas de tráfico também podem ser sumarizados em um problema de redes complexas (ZHAO et al., 2005).

2.3.2 ESPECTRO DO GRAFO

A matriz de adjacência é a matriz binária que se constrói naturalmente a partir das relações de adjacência entre os vértices do grafo (em caso de grafo direcionado), como demonstrada na Figura 11 (ABREU et al., 2014). A matriz de adjacência também é uma forma de representação do grafo sem pesos. O espectro do grafo são os conjuntos dos autovalores da matriz de adjacência (ABREU et al., 2014).

Autovetores e Autovalores são conceitos da álgebra linear, se uma matriz T pode ser multiplicada por um vetor vT não-nulo de forma que a matriz se torne um múltiplo linear dela mesma, então vT é denominado de autovetor e o múltiplo real λ é denominado autovalor (LEITHOLD, 1998):

$$\begin{bmatrix} 2 & 1 \\ 0 & 2 \end{bmatrix} x \begin{bmatrix} 3 & 0 \end{bmatrix} = \begin{bmatrix} 6 & 0 \end{bmatrix} \Rightarrow \lambda = 2 \tag{12}$$

Considerando a matriz de adjacência como *A*, consideramos a_{ij} a relação entre os vértices, sendo 1 caso v_i e v_j sejam adjacentes e 0 caso contrário. Para demonstração da extração da matriz adjacente de um grafo, é possível observar a Figura 10 já que o vértice *V1* se conecta com os vértices *V2*, *V4* e *V5*, logo, a construção da matriz de adjacência somente considerando o vértice *V1* é $0(V1- > V1)1(V1- > V2)0(V1 > V3)1(V1- > V4)1(V1- > V5)$, é possível visualizar a matriz de adjacência na Figura 11.

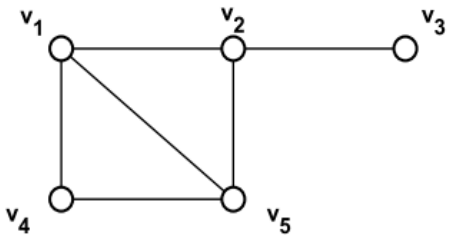


Figura 10: Grafo (G) para extração da Matriz Adjacente

Fonte: (ABREU et al., 2014)

Dado o grafo representado na Figura 10 *G*, sua matriz adjacente *A (G)* pode ser representada pela Figura 11

$$A(G) = \begin{bmatrix} 0 & 1 & 0 & 1 & 1 \\ 0 & 0 & 1 & 0 & 1 \\ 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 1 \\ 1 & 1 & 0 & 1 & 0 \end{bmatrix}$$

Figura 11: Matriz Adjacente do grafo G da Figura 10

Fonte: (ABREU et al., 2014)

O polinômio característico de uma matriz de adjacência *A(G)* do grafo *(G)*, pode ser definido por $P_G(\lambda) = det(\lambda I - A(G))$. λ é dito um autovalor do grafo G quando λ é uma raiz de $P_G(\lambda)$. Se A(G) possui s autovalores distintos $\lambda_1 > ... > \lambda_s$ com multiplicidades iguais, respectivamente, $Am(\lambda_1), ..., m(\lambda_s)$, o Espectro do Grafo, denotado Espectro(G), é definido como a matriz de $2xs$, onde a primeira linha é constituída pelos autovalores distintos de A(G) dispostos em ordem decrescente e a segunda, pelas suas respectivas multiplicidades algébricas. (ABREU et al., 2014) (FRITSCHER, 2011).

Sendo assim, o polinômio característico do grafo da Figura 10 é $P_G(\lambda) = \lambda^5 - 6\lambda^3 - 4\lambda^2 + 3\lambda + 2$ e seu espectro pode ser visualizado na Figura 12.

$$Espectro(G) = \begin{bmatrix} 2,6412 & 0,7237 & -0,5892 & -1 & -1,7757 \\ 1 & 1 & 1 & 1 & 1 \end{bmatrix}$$

Figura 12: Espectro do grafo G da Figura 10

Fonte: (ABREU et al., 2014)

O maior valor do Espectro de G é o índice de G, nesse caso, o índice do grafo da Figura 10 é $2,6412$.

É possível verificar várias propriedades dos grafos analisando seus autovalores, considerando a matriz de Adjacência como A e a_{jn} o vértice, sendo como, por exemplo (FRITSCHER, 2011):

- O número de arestas é igual a $a - a_2$, e também igual à metade da soma dos quadrados dos autovalores;
- O número de triângulos é igual a $-\frac{a_3}{2}$, e também é igual a $\frac{1}{6}tr(A^3)$;
- O elemento ij da matriz A^t indica o número de cadeias de comprimento t entre o i-ésimo e o j-ésimo vértices do grafo;
- O maior autovalor está entre o grau médio e o grau máximo, onde o grau de um vértice é o número de arestas incidentes a este;
- O quadrado do maior autovalor está limitado superiormente por $2m(1 - \frac{1}{n})$, onde *m* o número de arestas e *n* o número de vértices do grafo;
- Um grafo é bipartido se, e somente se, possui espectro simétrico em relação a zero;

2.3.2.1 APLICAÇÕES PRÁTICAS COM ESPECTRO DO GRAFO

O espectro do grafo é uma forma útil para entender sobre a natureza do grafo, também é utilizado para uma série de aplicações práticas recentemente publicadas. Foi demonstrado que as estruturas de rede funcional do cérebro e das redes regulatórias de genes são mais adequadas para caracterizar os estados do organismo do que a análise individual de suas partes. Além disso, é sabido que tais redes não são exatamente iguais mesmo entre indivíduos do mesmo grupo devido a variabilidade intrínseca (FUJITA et al., 2017).

Com a investigação das propriedades espectrais de grafos, foram introduzidos métodos estatísticos para seleção de modelos, estimativa de parâmetros e teste de hipótese para discriminar amostras, identificando a correlação conjuntos e fluxo de informação entre grafos de dados genéticos (VIDAL et al., 2017) e neuroimagens (FUJITA et al., 2017).

Foram desenvolvidos pacotes em R para utilização do espectro do grafo para a identificação de redes formadas por propriedades biológicas, como o *ANOCVA* (VIDAL et al., 2017) desenvolvido para comparar *clusters* entre grupos e *CoGA* (SANTOS et al., 2015) para identificar expressões diferenciais em conjuntos de genes.

Um dos principais problemas na genômica é a interferência automática de grupos de proteínas homólogas a partir de semelhanças nas sequências genicas. A utilização de métodos baseados em agrupamento com espectro do grafo mostraram avanços na precisão da separação em relação aos métodos formulados até então (SASIDHARAN, 2006).

Além de bases biológicas o espectro do grafo também é muito utilizado em segmentação e processamento de imagem (TUNG; WONG; CLAUSI, 2010), a segmentação é importante para a descrição e a classificação de imagens, *clusters* (Grupos) podem ser formados a partir de características como intensidade do pixel, cor, textura, localização, ou combinações destas (SURYANARAYANA; RAO; SWAMY, 2015). Existem métodos propostos a realizar uma segmentação excessiva da imagem no nível do pixel usando agrupamento espectral, e então, mesclar os segmentos usando uma combinação de consenso estocástico e agrupamentos espectrais no nível do segmento, essa abordagem de demonstrou semelhante ou superior aos outros métodos comparados em quase todos os casos (TUNG; WONG; CLAUSI, 2010).

Outra abordagem recente utilizando espectro do grafo é em mineração de dados educacionais, já que, com o rápido aumento do volume de dados dos repositórios de várias áreas edu-

cacionais, extrair informações de toda essa massa de dados é muito importante para a educação, já que, pode-se obter métricas para auxiliar o desenvolvimento acadêmico dos alunos. Há trabalhos que utilizam de características chave para predizer a performance de alunos com espectro do grafo (TRIVEDI, 2011). Ainda na área da educação, existem trabalhos para minerar o comportamento de alunos em redes sociais e comparar com seu desempenho acadêmico (OBADI et al., 2010).

Uma área importante de atuação e de grande volume de dados é a resolução de entidades, ou seja, identificar se objetos de fontes diferentes representam a mesma entidade no mundo real, esse problema se agrava em aplicações com múltiplos sistemas e grandes bases de dados sem um identificador único que represente a entidade em todos os sistemas. Foi proposto um algoritmo que seja matematicamente eficiente para resolver essa questão, o *SPAN - Spectral Neighborhood* (vizinho espectral) baseado em espectro do grafo que demonstrou ser mais rápido e escalável para grandes bases de dados além de ser mais tolerante a ruídos em relação aos métodos comparados (SHU L., 2011).

Embora algoritmos de *linkage* e algoritmos e *k-means* sejam muito populares no processamento de fala e robustos ao ruído, eles são mais adequados apenas para *clusters* linearmente separáveis e arredondados. No entanto, o agrupamento espectral é capaz de encontrar *clusters* estendidos e é mais robusto ao ruído do que os dois tipos de algoritmos citados (SURYANARAYANA; RAO; SWAMY, 2015).

3 METODOLOGIA

Nesta seção, é apresentada a metodologia para Detecção de Ponto de Mudança com o Espectro do Grafo. A Figura 13 apresenta a visão geral da metodologia proposta.

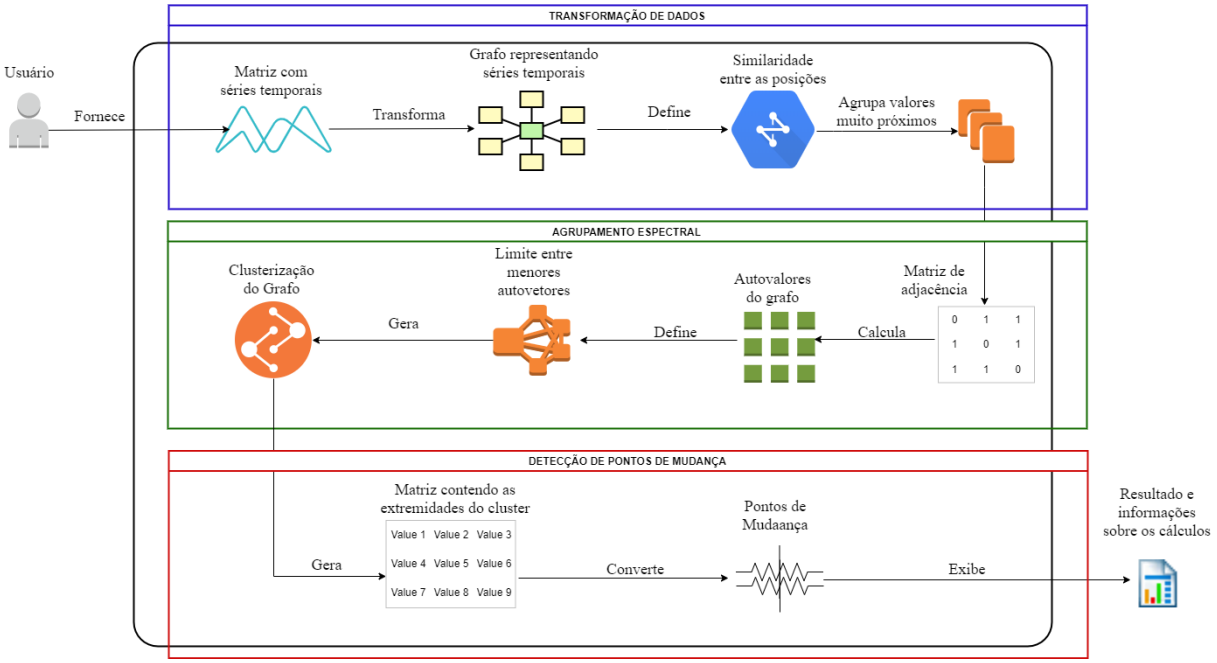


Figura 13: Explicação gráfica do processo de detecção de pontos de mudança com o espectro do grafo

O algoritmo proposto é não-paramétrico e aplicável a quase todas as séries temporais, uma vez que, a medida de similaridade relatada no espaço amostral pode ser definida. É necessário fornecer ao algoritmo quais estatísticas serão utilizadas na varredura baseada em grafos, bem como medidas de similaridade e alternativas de intervalo.

3.0.1 TRANSFORMAÇÃO DOS DADOS

O primeiro passo do método proposto é focado em transformar séries temporais multi-valoradas em um grafo. Inicialmente, cada observação de cada medida representa um vértice no grafo e o valor entre as arestas representa a distância entre cada observação. Existem várias distâncias métricas que podem ser utilizadas. Neste trabalho foi utilizada a distância euclidiana tradicional que é consolidada e trivial [Gower 1982], nesse caso a distância representa a discrepância entre o valores de uma série temporal.

Após a conversão, uma matriz de similaridade é formada sobre todas as observações

do grafo. O processo é computacionalmente pesado e uma grande quantidade de dados precisa ser armazenada em memória em caso de múltiplas medidas ou em séries muito longas. A alternativa adotada para diminuir esse entrave de alto consumo de memória e tempo, bem como reduzir o ruído entre os valores é agrupar os vértices muito próximos, assim, apenas os vértices agrupados são armazenados em memória.

Sendo assim, se um vértice A estiver relativamente próximo de um vértice B, ambos os vértices poderão ser substituídos por um único vértice C. A distância mínima pode ser parametrizada ou calculada com base na distância média entre todos os vértices da mesma série temporal. Este processo pode ser repetido até que permaneçam apenas distâncias relevantes entre os vértices.

O processo de agrupamento de vértices também pode ser considerado como uma suavização dos dados reduzindo o ruído. No entanto, a estrutura da base deve ser bem conhecida para que informações importantes não sejam perdidas. Para os testes realizados nesse trabalho foram agrupados os vértices onde a distância é menor que 0,01 da distância máxima entre vértices na mesma série e também, é menor que 0,02 da distância média na mesma série.

O agrupamento de vértices leva em consideração o armazenamento de marcações que representam as observações originais, de modo que, a transformação do gráfico em séries temporais pode ser feita diretamente por referência após a detecção dos pontos de mudança.

3.0.2 AGRUPAMENTO ESPECTRAL

O objetivo do agrupamento espectral é agrupar dados que estão conectados, mas não necessariamente agrupados em limites convexos. Para realizar o agrupamento espectral, assim como no agrupamento comum, é necessário uma medida de similaridade ou afinidade $s(x, y)$ para determinar o quão próximo os pontos x e y são um do outro (LUXBURG, 2007).

Denotando a matriz de similaridade S, sendo que $S_{ij} = s(x_i, x_j)$ dada a semelhança entre as observações x_i e x_j. Usando uma medida de similaridade dos pontos, como por exemplo a Gaussiana, que, quanto mais próximo os valores mais tendem a 1 e quanto mais distantes mais tendem a 0, computando a matriz de similaridade S e, gerando uma matriz baseado na afinidade de S denominada A, sendo A positiva e simétrica (NETO, 2013).

Possuindo a matriz de afinidade *A*, o agrupamento é resumido por um problema de partição de grafo, já que componentes conectados do grafo podem ser interpretados como *cluster* (grupos). O grafo deve ser particionado de tal forma que as arestas que conectam diferentes grupos devem ter pesos baixos e as arestas dentro do mesmo grupo deve ter valores altos.

A próxima etapa é computar a matriz Laplaciana resultante da subtração de uma matriz de grau onde cada valor diagonal é o grau do vértice respectivo e todas as outras posições são zero, da matriz de afinidade anteriormente calculada, existem variantes da matriz laplaciana que podem ser aplicadas nesse caso (LUXBURG, 2007), porém, o método foi desenvolvido utilizando a matriz laplaciana simples, conforme descrito nesse parágrafo.

Assumindo que pretende-se identificar n grupos, a próximo etapa é encontrar os n menores autovetores (ignorando o autovetor constante trivial). O espectro do autovalor possui uma lacuna que fornece o valor de n. Em (NETO, 2013) é possível verificar o exemplo do agrupamento da série que dificilmente seria agrupada pelos métodos tradicionais (*k-Means* e derivados).

Utilizar o espectro do grafo para agrupamento está sendo frequentemente utilizado em várias áreas e, obteve resultados promissores na segmentação de imagens (CASACA, 2014), porém, até o melhor do nosso conhecimento, não foi publicado um trabalho para o uso de agrupamento pelo espectro do grafo para detecção de pontos de mudança. Mesmo o uso de grafos de forma trivial para detecção de pontos de mudança tem sido pouco explorado (CHEN; ZHANG, 2015).

3.0.3 DETECÇÃO DE PONTOS DE MUDANÇA

Usando o agrupamento espectral como descrito anteriormente, as observações da série temporal são divididas em grupos distintos, onde $1 < n_0 \leq \tau \leq n_1$, sendo n_0 e n_1 dois grupos distintos e τ o valor limite entre os grupos, portanto, τ é o ponto de mudança. Essa definição pode ser extrapolada para vários grupos na série temporal até que todos os pontos de mudança sejam determinados.

Caso o contexto que o teste seja aplicado já exista conhecimento sobre a quantidade máxima de pontos de mudança na série temporal é possível informar como parâmetro o número máximo de pontos de mudança, e assim, os algoritmos de agrupamento como o *K-means* serão parametrizados para produzirem um número de *clusters* que não ultrapasse o limite parametrizado, gerando um número de pontos de mudança menor ou igual ao valor parametrizado.

Caso o conhecimento da série temporal seja notório e, exista o conhecimento exato de quantos pontos de mudança exista na base, sendo o problema em questão apenas identificar a posição dos pontos de mudança, é possível parametrizar o número de pontos de mudança, assim, o algoritmo de agrupamento como o *K-means* será parametrizado para identificar o número de *cluster* igual ao número de pontos de mudança - 1.

O Algoritmo 3 demonstra a detecção de pontos de mudança utilizando o *SpecDetec*, ilustrando tanto a transformação de dados como o processo de detecção. O Algoritmo 3 é executado em *loop* para todas as séries temporais existentes no conjunto.

3.0.4 SpecDetec

O método desenvolvido nesse trabalho foi implementado na linguagem *R* e disponibilizado no repositório *CRAN* com o nome *SpecDetec*, o nome da função principal de detecção se manteve como *Spec*, porém, o nome do pacote foi nomeado como *SpecDetec* pois os nomes de pacotes no repositório *CRAN* são únicos e não podem ser duplicados.

Como o pacote foi disponibilizado no repositório oficial, é possível fazer uso do mesmo em qualquer maquina com o serviço *R* instalado, independente do sistema operacional, o pacote pode ser instalado e colocado para uso com duas linhas de código:

```
1 install.packages('SpecDetec')
2 library(SpecDetec)
```

Código Fonte 3.1: Instalação do Package SpecDetec

Foi realizado um esforço para que o *package* seja o mais simplificado possível, para auxiliar o entendimento do *package* algumas bases de dados de teste estão acopladas ao *package* e prontas para o uso (não sendo necessário nenhum tipo de transformação prévia). No

Algoritmo 3 Detecção de Mudanças com SpecDetec

Entrada: S = série temporal multi-valorada, L = Limite para Agrupamento, Max = Número máximo de pontos de mudança, Est = Número estimado de pontos de mudança

Saída: Conjunto de posições representando os pontos de mudança

início

  (*Etapa 1: Transformação de dados*)

  para *cada* $s_i \in S$ **faça**

    O = MarcaValoresOriginais(s_i)

    D = CalculaDistanciaEntrePontos("Euclidiana", S, s_i)

  fim

  SN = AgrupaValoresProximos(s_i, S, D, L, O)

  MS = CalculaMatrizSimilaridade(D, SN)

  (*Etapa 2: Detecção por Agrupamento Espectral *)

  MA = CalculaMatrizAfinidade("Gaussiana", SN, MS)

  A = CalculaMatrizAdjacencia(SN, MA)

  AV = CalculaAutoValores(A)

  G = AgrupaConjuntoDados("KMeans", AV, G, Max, Est)

  para *cada* $gn_i \in GN$ **faça**

    P = PosicaoPontoMudanca(O, SN, gn_i)

  fim

  retorna P

fim

código fonte 3.2 é realizado uma demonstração de uso do *SpecDetec* na linguagem R com uma explicação resumida dos parâmetros.

O código do *package* é aberto, como todo o pacote inserido no *CRAN*, sendo assim, é possível customiza-lo a nível de código, não se limitando as customizações naturais provindas das diferentes combinações entre os parâmetros. Para que o código fique em um ambiente evolutivo de fácil customização e, que possa receber de críticas e comentários, foi compartilhado também em um repositório do *GitHub* (UZAI, 2019), assim, além do uso comum, também é possível fazer sugestões sobre o *package* para melhoria continua.

Uma das possíveis melhorias desejadas para o *SpecDetec* é a inserção de mais algoritmos de agrupamento, hoje conta somente com o *k-means*, é recomendado e incentivado que testes com o *package* sejam compartilhados no *GitHub* para orientar a melhorias ou correções de possíveis falhas. Considerando o uso do *k-means* o algoritmo assume a complexidade computacional exponencial, pois o *k-means* é naturalmente um NP-Difícil, em geral, o agrupamento espectral possui a complexidade $O(n^3)$.

Na página do *package SpecDetec* do *GitHub* também é possível encontrar de forma mais ampla exemplos de uso, bases de dados e *scripts* de testes do *SpecDetec* e de outros algoritmos para comparação, o intuito é fornecer insumos para que outros desenvolvedores criem novos algoritmos de detecção de pontos de mudança usando os *scripts* de teste para orientar o desenvolvimento.

```
1 #Carregamento dos dados embutidos no pacote
2 data(DEVICE)
3
```

```
#Localizacao real dos pontos de mudanca ao longo da serie
realCP <- c(which(diff(DEVICE1$Class) != 0)      #[1] 125 250

#Serie temporal utilizada para calculo
data <- DEVICE1[, 1]

#Numero maximo de vizinhos para agrupamento dos dados
neighboors <- 6

#Tolerancia do agrupamento espectral
tolerance <- 0.005

#Numero maximo de pontos de mudanca que o algoritmo ira buscar
maxCP <- 2

#Numero de pontos de mudan a estimado
estimationCP <- 2

#Executando o algoritmo pela funcao Spec, e informando  todos as entradas.
estimatelCP <- Spec(data, neighboors, tolerance, maxCP, estimationCP)
#Pontos de mudan a localizados pelo algoritmo:  [1] 107 275
```

Código Fonte 3.2: Utilizando o SpecDetec para detecção de pontos de mudança na base de dados padrão

4 RESULTADOS E DISCUSSÕES

4.1 AVALIAÇÃO DO DESEMPENHO

Para avaliação de desempenho dos algoritmos são necessárias métricas objetivas que sejam aplicadas de forma igualitária a todos os algoritmos, para calcular essas métricas as posições da série temporal devem ser rotuladas, é comum que séries sejam classificados de forma booliana, onde cada posição pode ser ou não um ponto de mudança (COOK; KRISHNAN, 2015), os algoritmos tentando classificar corretamente cada anotação, expressam os seguintes casos possíveis:

1. Verdadeiro Positivo (VP): O algoritmo classificou corretamente a anotação como um ponto de mudança.
2. Falso Positivo (FP): O algoritmo classificou como ponto de mudança uma anotação normal.
3. Falso Negativo (FN): O algoritmo classificou um ponto de mudança como uma anotação normal.
4. Verdadeiro Negativo (VN): O algoritmo classificou corretamente uma anotação normal.

É desejável por tanto, que, o algoritmo maximize o valor de Verdadeiros Positivos e Verdadeiros Negativos, ao mesmo tempo que, minimize Falsos Positivos e Falsos Negativos. Existem várias métricas presentes literatura para calcular a performance dos algoritmos de detecção de pontos de mudança, como descrito a seguir:

- Acurácia (AC): É a métrica usada mais frequentemente para determinar a performance de pontos de mudança (COOK; KRISHNAN, 2015) que representa a porcentagem de classificações corretas em uma determinada analise, obtida pela seguinte formula:

$$AC = \frac{VP + VN}{VP + FP + FN + VN}$$

  Outra medida frequentemente utilizada é a Taxa de Erro (AMINIKHANGHAHI; COOK, 2016), representando o inverso da acurácia, sendo obtida por 1 – Acurácia. A Acurácia e a Taxa de Erro uniformemente medidas podem não ser suficientes para medir a performance de um algoritmo, já que, consideram diferentes tipos de erro de igual importância, além de, desconsiderar a precisão dos acertos.

- Sensibilidade (SB): Também conhecida como Taxa de Verdadeiro Positivo, refere-se a taxa de classificações corretas de anotações como Ponto de Mudança, representado pela formula:

$$SB = \frac{VP}{VP + FN}$$

- Singularidade (SG): Também conhecida como Taxa de Verdadeiro Negativo, refere-se a taxa de classificações corretas de anotações como Ponto Normal, é uma métrica importante para medicação de séries temporais com muitos Pontos de Mudança e um risco alto de Falsos Negativos, é representada pela formula:

$$SG = \frac{VN}{VN + FP}$$

- Taxa de Falso Positivo (TFP): Representa a taxa de classificações equivocadas como Ponto de Mudança, é mais relevante no contexto offline, pois, comumente a identificação de pontos de mudança reais e de alta precisão são mais relevantes, a Taxa de Falso Positivo pode ser calculada pela formula:

$$TFP = \frac{FP}{FP + VN}$$

- Taxa de Falso Negativo (TFN): Representa a taxa de falha de encontrar os pontos de mudança reais, é uma métrica relevante no contexto online, pois, comumente existem poucos pontos de mudança detectáveis e existe uma preocupação maior em encontrar o ponto de mudança o mais rápido possível, deixar de detectar um ponto real potencialmente prejudica consideravelmente o tempo de detecção.

$$TFN = \frac{FN}{VP + FN}$$

- Taxa de Classificações Erradas (TCE): Também representada por Taxa de Erro, é uma métrica inversa a Acurácia, é uma medida relevante pois os pesos entre erros e acertos podem ser diferentes, por exemplo, em determinadas realidades é preferível que um algoritmo encontre poucos Verdadeiros Positivos e, tenha uma Taxa de Erro menor, do que, encontrar vários Verdadeiro Positivos com alta Taxa de Erro, principalmente no contexto de detecção em séries online.

$$TCE = \frac{FN + FP}{VP + FN + FP + VN}$$

- Precisão (PC): Representa a proporção de Verdadeiros Positivos entre todos os Pontos de Mudança classificados.

$$PC = \frac{VP}{VP + FP}$$

- Significância (SIG): É uma métrica de eficácia geral, resultado da combinação da Precisão e Sensibilidade, é calculada como uma proporção da importância ponderada de Precisão e

Sensibilidade, que pode conter pesos distintos em cada realidade, considerando um peso igualitário, ela segue representada pela formula:

$$SIG = \frac{2 * PC * SB}{PC + SB}$$

- Classificação Média (CM): Utilização da média aritmética entre diversas métricas com o objetivo de obter resultados mais confiáveis, somando então todos as métricas (M) utilizadas pelo número de métricas utilizadas (N), representado na formula:

$$CM = \frac{M}{N}$$

A formula a seguir representa uma Classificação Média usando todas as métricas indicadas nessa sessão (JODOIN, 2012):

$$CM = \frac{AC + SG + SIG}{3}$$

4.2 BASES DE DADOS

Para realizar comparações de forma justa e equilibrada, todos os algoritmos foram aplicados nas mesmas bases de dados, preferencialmente, as características das bases aplicadas devem cobrir uma variedade de propriedades distintas para que a natureza da base de dados não favoreça um determinado algoritmo.

Todas as bases de dados foram utilizadas são do repositório *UCR Time Series Classification Archive* (CHEN et al., 2015), as referências com mais informações sobre as extrações das bases podem ser encontradas nos arquivos originais. A seleção do repositório foi dada pelo critério de adaptação aos algoritmos, pois, alguns algoritmos tem alta complexidade e o tempo de processamento em séries temporais com mais de 10.000 anotações dificulta os testes, já que, frequentemente é necessário ajustar os parâmetros e refazer os testes diversas vezes.

A seleção dos *datasets* também levou-se em consideração a segurança e integridade dos dados, sendo assim, todos dos *datasets* selecionados são classificados, sem nenhum valor não preenchido e já utilizado em *benchmnarks* em trabalhos recentes (KEOGH; KASETTY, 2003) (RAKTHANMANON et al., 2013).

As séries temporais da maioria das bases do repositório possuem mais de uma dimensão, para que os testes pudessem acontecer com todos os algoritmos igualmente, inclusive os algoritmos que não possuem função multidimensional, essa característica não foi aproveitada, ou seja, em séries multidimensionais somente uma característica foi considerada por vez.

Todas as séries dentro de um mesmo *dataset* possuem o mesmo tamanho (número de anotações) e o mesmo número de pontos de mudança, variando somente a disposição do ponto de mudança ao longo da série. Por exemplo, todas as 720 séries temporais do *dataset RefrigerationDevices* possuem tamanho 375 e 2 pontos de mudança, A Figura 26 exibe a primeira série do *dataset*, onde os pontos de mudança são nas posições 125 e 250, olhando atentamente é possível observar que após a posição 125 os valores da série se tornam mais altos e a distância entre os picos se torna maior, após a posição 250 a distância entre os picos se torna menor nova-

Tabela 1: Informações básicas das bases de dados selecionadas para teste da biblioteca UCR Time Series Classification Archive. P.M. representa número de Pontos de Mudança e T.M representa tamanho da série temporal (número de anotações) (CHEN et al., 2015)

Tipo	Dataset	Autor	P.M.	T.M.	Qtd. Séries
Contorno de Objetos	ArrowHead	Tony Bagnall	2	175	251
	BeetleFly	Tony Bagnall	1	20	512
	BirdChicken	Tony Bagnall	1	20	512
	FaceAll	Xi	13	1690	131
	ShapeletSim	Tony Bagnall	1	180	500
	ShapesAll	Tony Bagnall	59	600	512
	WormsTwoClass	Tony Bagnall	1	181	900
	DistalPhalanx	L. Davis	2	139	80
Variação Elétrica	Computers	Tony Bagnall	1	250	720
	ElectricDevices	Tony Bagnall	44	7711	96
	LargeKitchenAppliances	Tony Bagnall	2	375	720
	ECG	Olszewski	1	100	96
	RefrigerationDevices	Tony Bagnall	2	375	720
	ScreenType	Tony Bagnall	2	375	720
	SmallKitchenAppliances	Tony Bagnall	2	375	720
Espectrógrafo	Coffee	Tony Bagnall	1	28	286
	OliveOil	Tony Bagnall	3	30	570
	Wine	Tony Bagnall	1	54	234
	Meat	Tony Bagnall	2	60	448
	Strawberry	Tony Bagnall	8	613	235
Movimento Humano	Yoga	Xi	1	3000	426
	ToeSegmentation1	L. Ye, E. Keogh	1	228	277
Meteorológico	Earthquakes	Tony Bagnall	39	322	512
Onda Sonora	Phoneme	Hossein	38	1896	1024
Artificial	Synthetic Control	Pham	5	300	60

mente e a distribuição dos valores se torna mais irregular, outras séries desse *dataset* tem pontos de mudança em outras disposições mas seguindo sempre esses mesmos padrões de mudança.

Na sequência os *datasets* são detalhados para que se possa compreender melhor as origens de dados e a importância acadêmica para cada cenário de testes.

4.2.1 Contorno de Objetos

Os dados de *ArrowHead* consistem em contornos das imagens de pontas de seta. As formas dos pontos do projétil são convertidas em séries temporais usando o método baseado em ângulo. A classificação de pontos de projétil é um tópico importante na antropologia. As classes são baseadas em distinções de forma, como a presença e localização de um entalhe na seta.

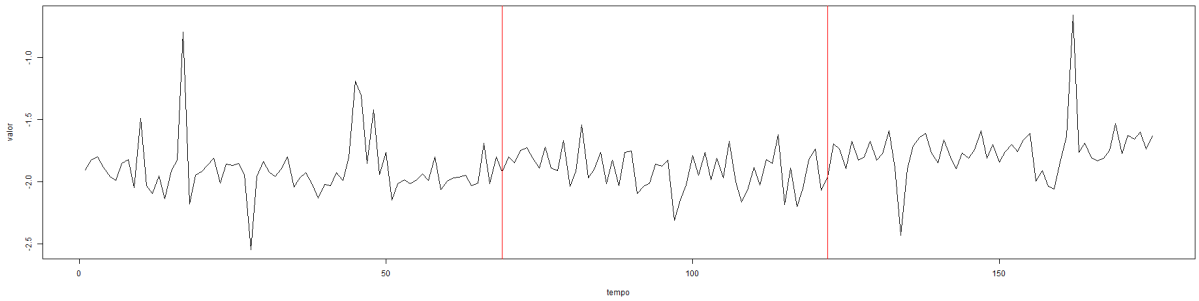


Figura 14: Série temporal ArrowHead com pontos de mudança descritos. As três classes são chamadas *"Avonlea"*, *"Clovis"* e *"Mix"*.

Os dados de *BeetleFly*, *BirdChicken*, *ShapesAll* foram extraídos de *MPEG-7 CE Shape-1 A* Parte B, um banco de dados de imagens binárias desenvolvidas para testar descritores de formato *MPEG-7* e está disponível *online* gratuitamente. É usado para testar contornos / imagens e descritores baseados em esqueletos. As classes de imagens variam amplamente e incluem classes semelhantes em forma umas às outras.

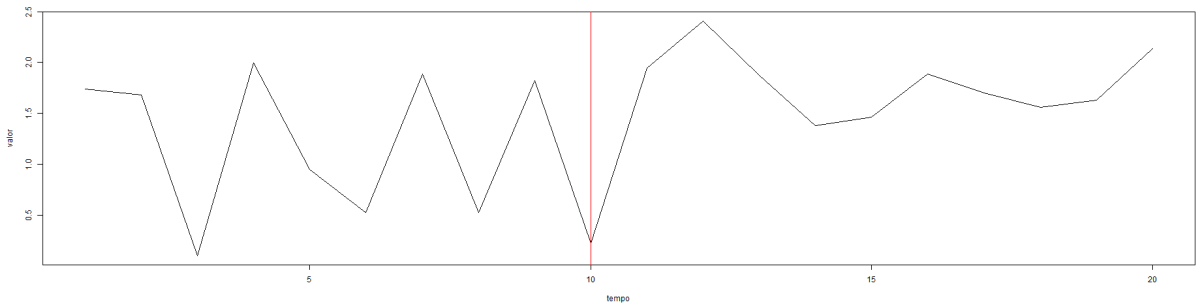


Figura 15: Série temporal BeetleFly com pontos de mudança descritos. As classes são besouro e mosca.

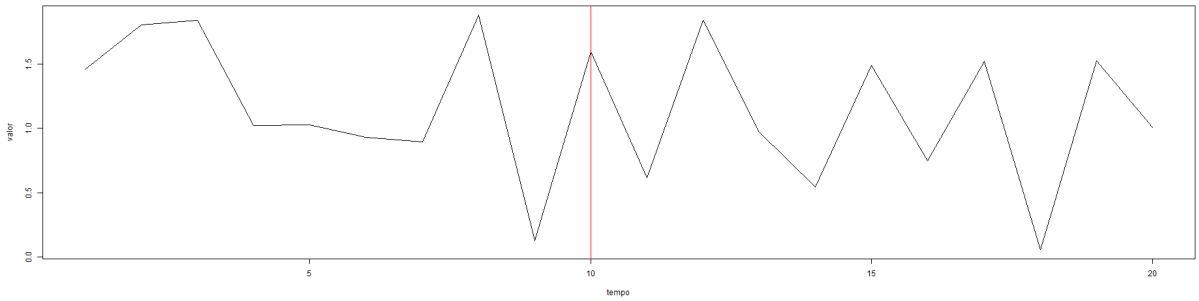


Figura 16: Série temporal BirdChicken com pontos de mudança descritos. As classes são pássaro e galinha.

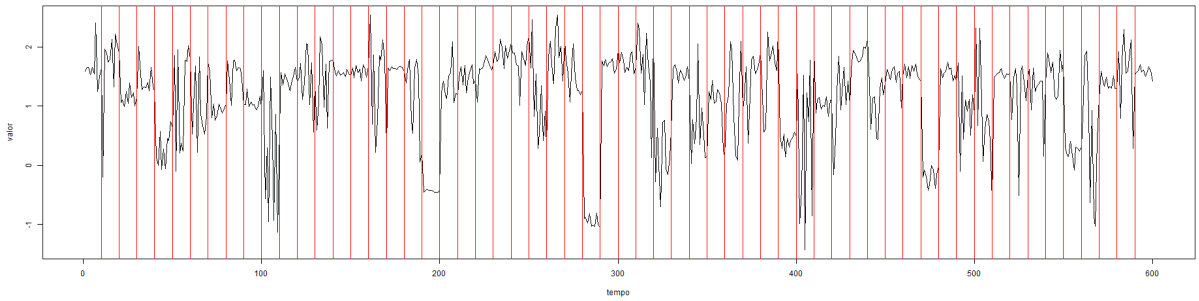


Figura 17: Série temporal ShapesAll com pontos de mudança descritos. Existem 20 instâncias de cada classe e 60 classes no total.

O *FaceAll* foi gerado no *UCR pelo Xiaopeng Xi*. Os rostos pertencem a 14 estudantes de graduação. Cada contorno facial é mapeado em uma série unidimensional.

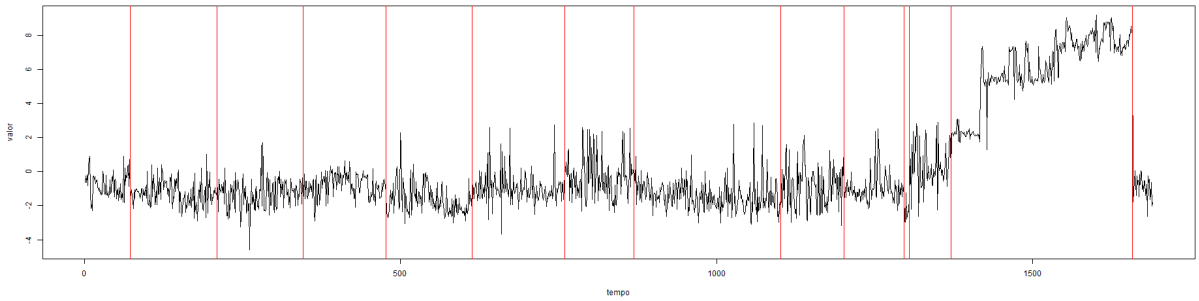


Figura 18: Série temporal FaceAll com pontos de mudança descritos.

Shapelet Sim é um conjunto de dados simulado projetado identificar formas. Uma das cinco formas é incorporada em uma posição aleatória sem ruído.

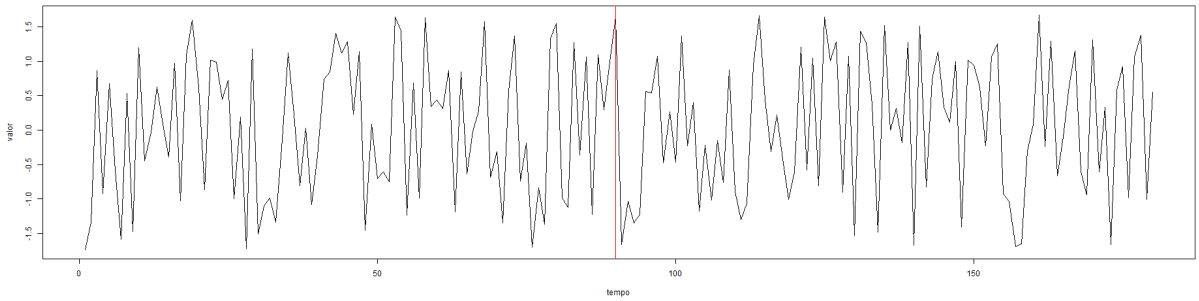


Figura 19: Série temporal ShapeletSim com pontos de mudança descritos.

Caenorhabditis elegans é um verme comumente usado como um organismo modelo no estudo da genética. Sabe-se que o movimento desses vermes é um indicador útil para entender a genética comportamental. Tem sido demonstrado que o espaço das formas que o *Caenorhabditis elegans* adota em uma placa de ágar pode ser representado por combinações de quatro formas de base, denominadas eigenworm. Uma vez que o contorno do verme é extraído, cada quadro

de movimento pode ser capturado por quatro escalares representando as amplitudes ao longo de cada dimensão quando a forma é projetada nas quatro eigenworm. Os dados referem-se a 258 traços de vermes convertidos em quatro séries de eigenworm. Os dados de eigenworm são comprimentos de 17984 a 100674 (amostrados a 30 Hz, portanto de 10 minutos a 1 hora) e em quatro dimensões (*eigwnworm* 1 a 4).

Existem cinco classes: *N2*, *goa-1*, *unc-1*, *unc-38* e *un63*. *N2* é do tipo selvagem (isto é, normal), os outros 4 são estirpes mutantes. Estes conjuntos de dados são apenas da primeira dimensão (primeira *eigenworm*). Esse *dataset* portanto possui duas classes que representam os tipos de verme selvagem ou mutante.

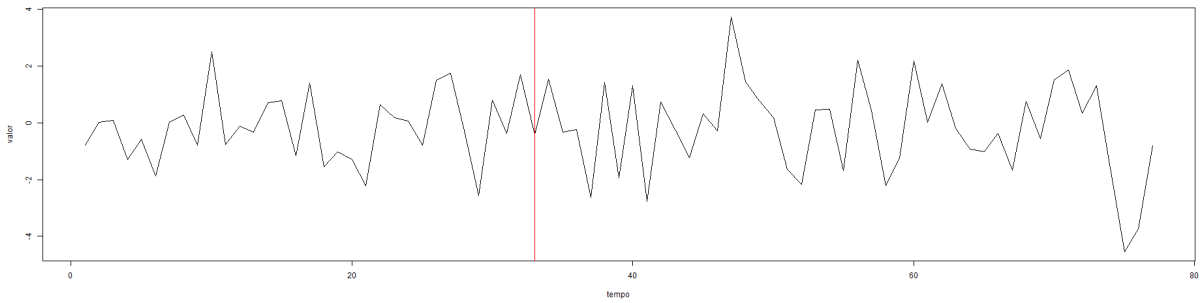


Figura 20: Série temporal WormsTwoClass com pontos de mudança descritos.

Os dados de *DistalPhalanxOutlineAgeGroup* são séries extraídas de imagens representando as medidas dos contornos dos ossos da mão, onde, as classes são grupos de idade óssea. Algoritmos extraíram automaticamente os contornos das mãos e depois os contornos de três ossos do dedo médio (falanges proximal, média e distal) em um universo de mais de 1300 imagens.

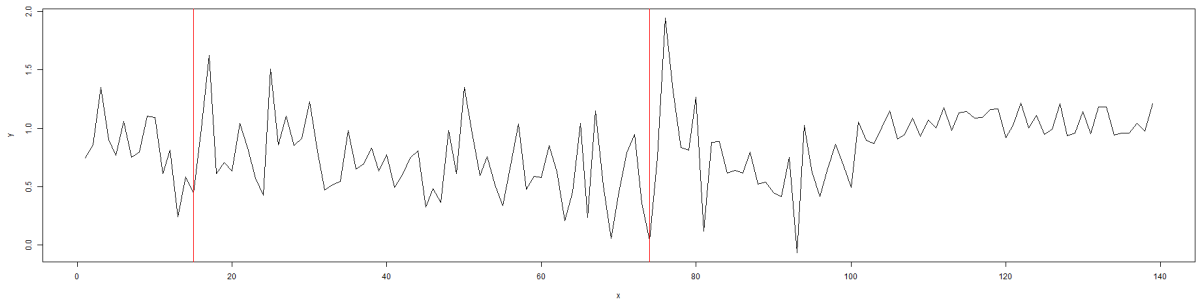


Figura 21: Série temporal DistalPhalanxOutlineAgeGroup com pontos de mudança descritos.

4.2.2 Variação Elétrica

Os dados de *Computers*, *ElectricDevices*, *LargeKitchenAppliances*, *RefrigerationDevices*, *ScreenType*, *SmallKitchenAppliances* foram obtidos por um estudo de patrocínio governamental chamado *Powering the Nation*. A intenção foi coletar dados comportamentais sobre como os consumidores usam a eletricidade dentro de casa para ajudar a reduzir o volume de carbono na atmosfera do Reino Unido. Os dados contêm leituras de 251 domicílios, amostrados em intervalos de dois minutos ao longo de um mês.

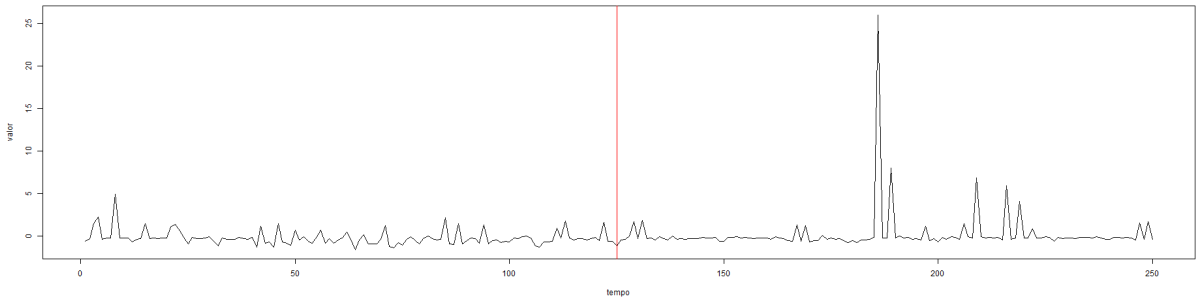


Figura 22: Série temporal Computers com pontos de mudança descritos. Classes são *Desktop* e *Laptop*.

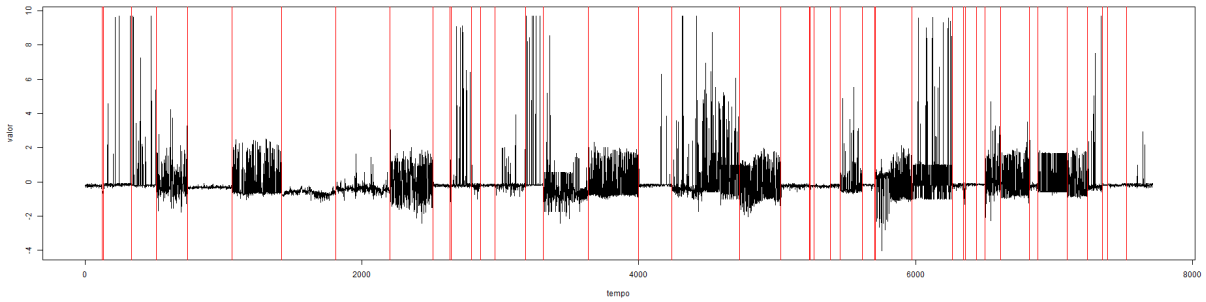


Figura 23: Série temporal ElectricDevices com pontos de mudança descritos.

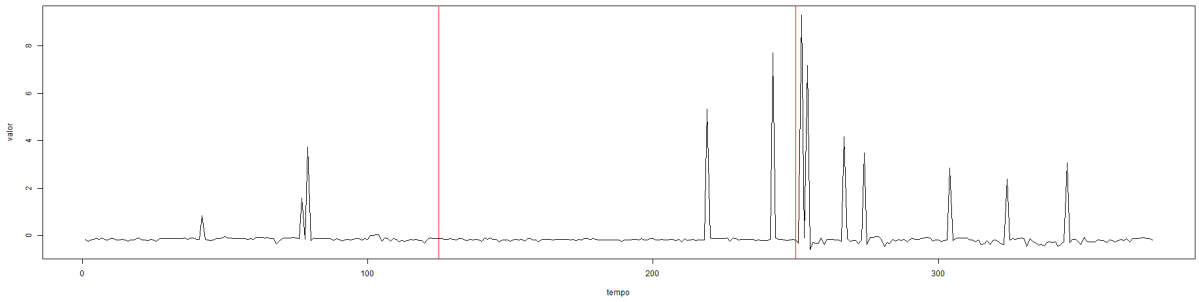


Figura 24: Série temporal LargeKitchenAppliances com pontos de mudança descritos. As classes são Lavadora, secadora e lava-louças.

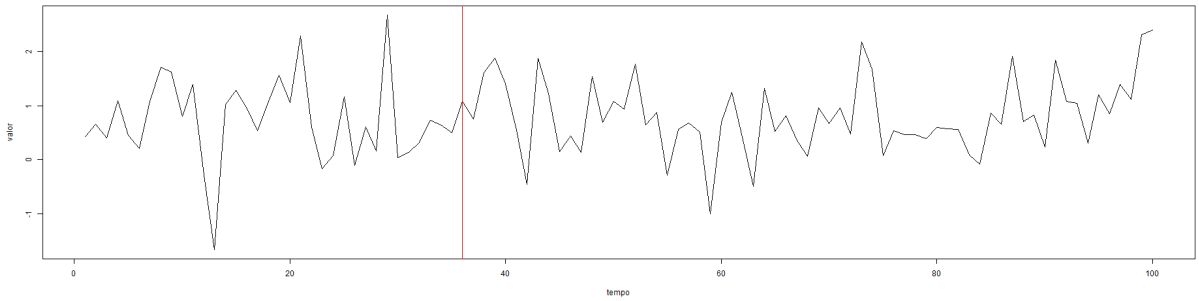


Figura 25: Série temporal ECG200 com pontos de mudança descritos. Cada série traça a atividade elétrica registrada durante uma pulsação. As duas classes são um batimento cardíaco normal e um infarto do miocárdio.

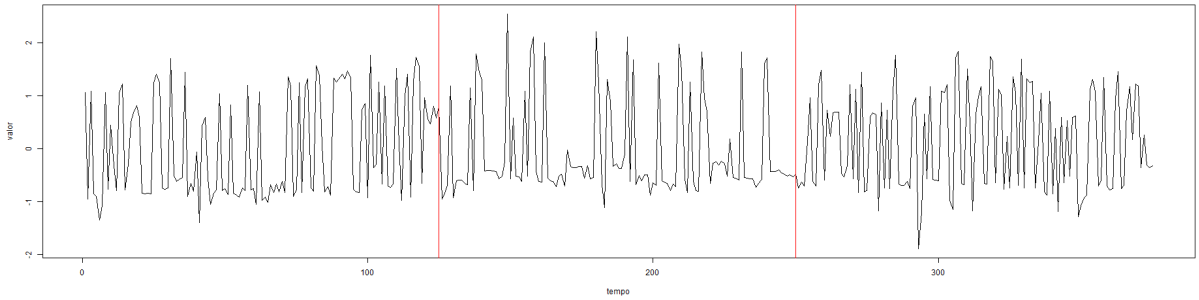


Figura 26: Série temporal RefrigerationDevices com pontos de mudança descritos, as classes são Geladeira / Freezer, Geladeira e Freezer Vertical.

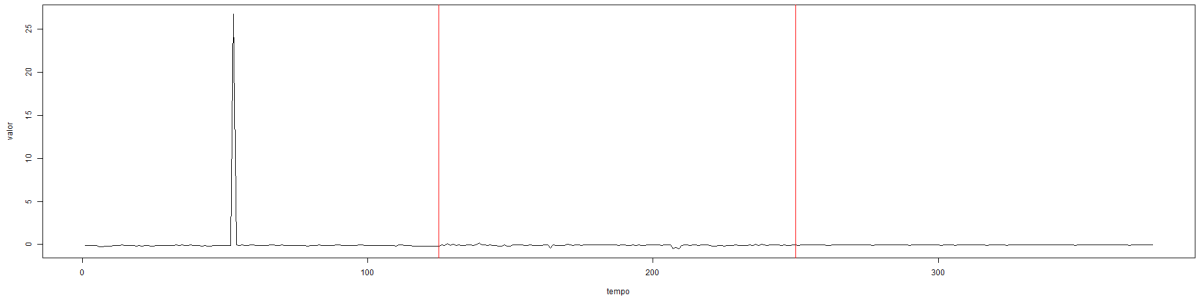


Figura 27: Série temporal SmallKitchenAppliances com pontos de mudança descritos. As classes são chaleira, micro-ondas e torradeira.

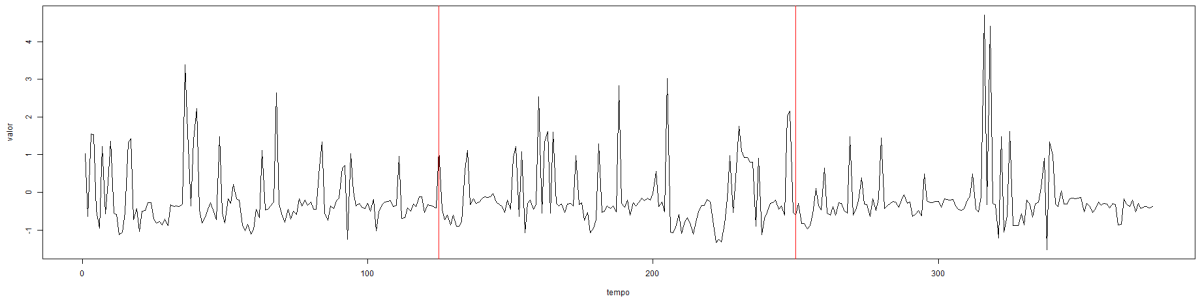


Figura 28: Série temporal ScreenType com pontos de mudança descritos. As classes são TV CRT, TV LCD e monitor de computador.

4.2.3 Espectrógrafo

Os espectrógrafos de alimentos são usados em quimometria para classificar os tipos de alimentos, uma tarefa que tem aplicações óbvias na segurança alimentar e garantia de qualidade. O *dataset Coffee* consiste em dados de café instantâneo obtidos por espectroscopia de infravermelho com transformada de *Fourier* e quimometria, possuindo duas classes para distinguir entre os grãos, são elas ”Robusta”e ”Arábica”.

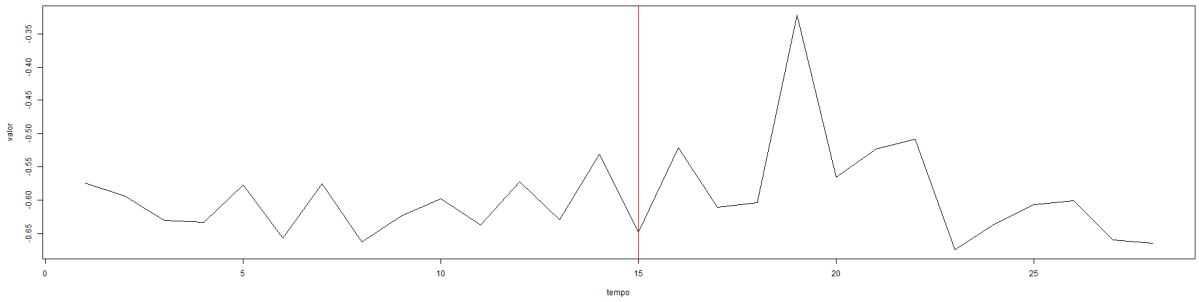


Figura 29: Série temporal Coffee com pontos de mudança descritos.

Os dados de *OliveOil* foram obtido usando espectroscopia de infravermelho com transformada de *Fourier (FTIR)*. As classes são um conjunto de azeite extra-virgem de países alternativos.

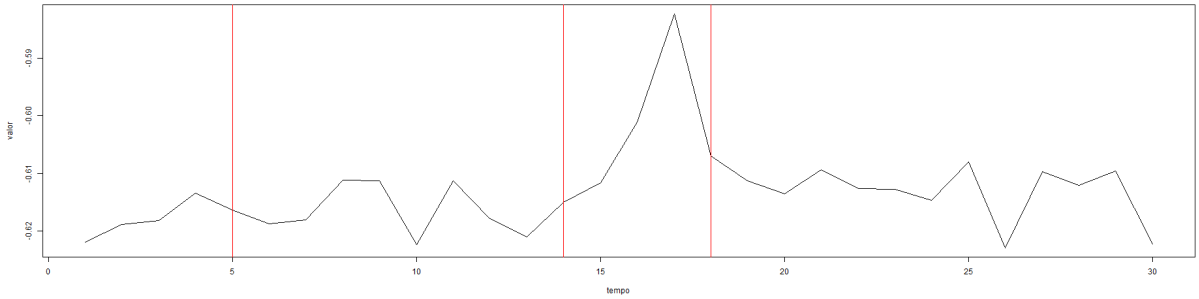


Figura 30: Série temporal OliveOil com pontos de mudança descritos.

O *dataset Wine* possui duas classes de vinhos, os dados são obtidos usando espectroscopia de infravermelho com transformada de *Fourier (FTIR)* com amostragem de refletância total atenuada (*ATR*).

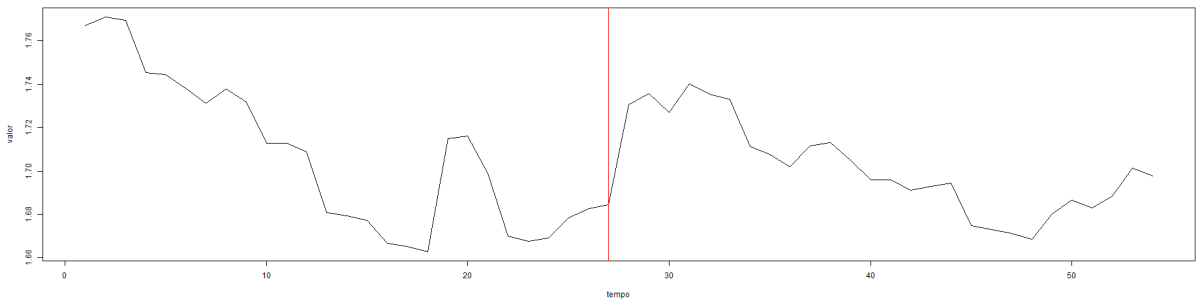


Figura 31: Série temporal Wine com pontos de mudança descritos.

Os dados de *Meat* foram obtido usando espectroscopia de infravermelho com transformada de *Fourier (FTIR)* com amostragem de refletância total atenuada *(ATR)*. As classes são de frango, porco e peru divididas em 60 amostras independentes.

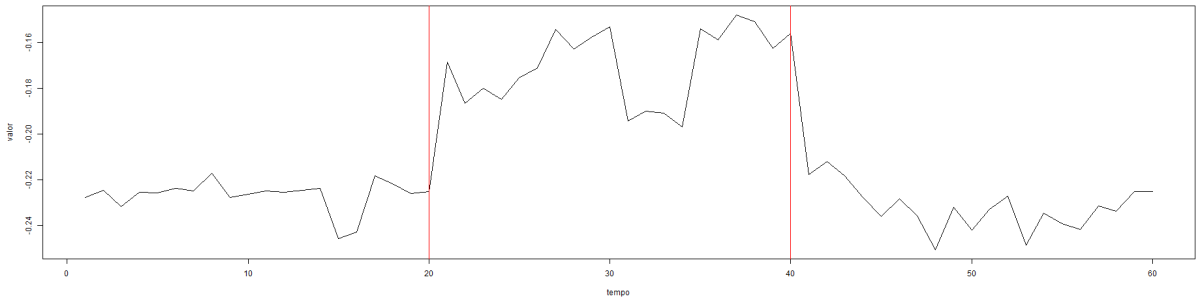


Figura 32: Série temporal *Meat* com pontos de mudança descritos.

As classes são morango (amostras autênticas) e não-morango (morangos adulterados e outras frutas). Obtido usando espectroscopia de infravermelho com transformada de *Fourier (FTIR)* com amostragem de refletância total atenuada (*ATR*).

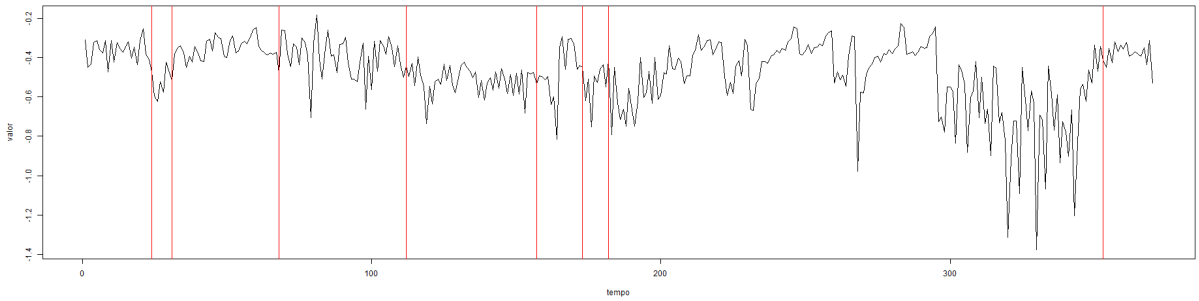


Figura 33: Série temporal Strawberry com pontos de mudança descritos.

4.2.4 Movimento Humano

O conjunto de dados foi obtido capturando dois atores que transitam entre poses de ioga em frente a uma tela verde. O *dataset* possui duas classes entre os autores (feminino e masculino). Cada imagem foi convertida em uma série unidimensional, gerando o contorno e medindo a distância do contorno ao centro da imagem.

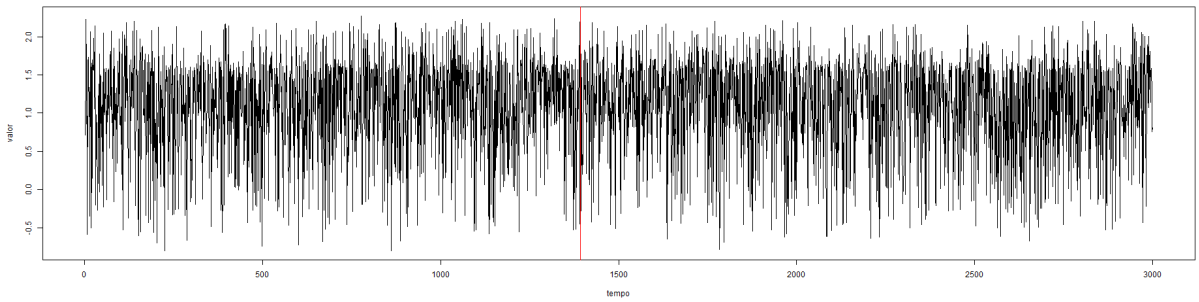


Figura 34: Série temporal Yoga com pontos de mudança descritos.

Os dados do *ToeSegmentation1* são derivados do *CMU* (*Graphics Lab Motion Capture Database*). É um *Dataset* de movimentação, que em sua definição são descrições de movimento agrupados em categorias. Existem duas classes, a primeira é a caminhada normal, com apenas andar nas descrições de movimento. A outra é a caminhada anormal, com as descrições de movimento contendo: andar mancando, andar com a perna ferida, caminhar com os dedos dobrados para a frente, machucar a perna, andar com a perna ruim ou o andar com dor no estômago. Nos passeios anormais, os atores fingem ter dificuldade em andar normalmente.

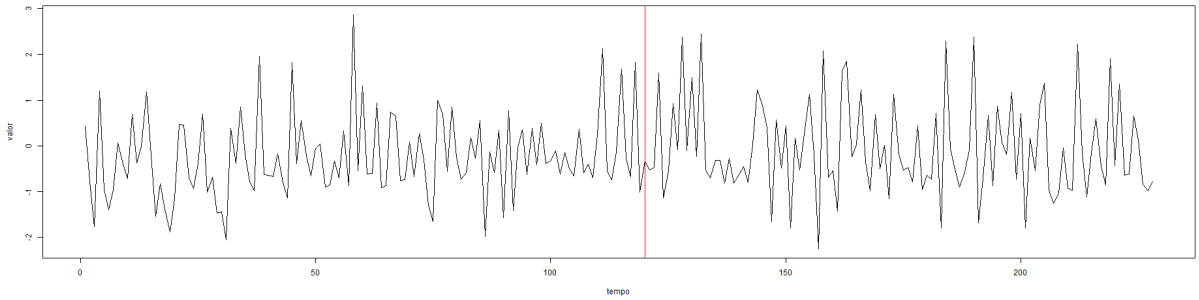


Figura 35: Série temporal ToeSegmentation1 com pontos de mudança descritos.

4.2.5 Meteorológico

O problema de classificação do terremoto envolve prever se um grande evento está prestes a ocorrer com base nas leituras mais recentes na área circundante. Os dados são obtidos do Data Center de Terremotos do Norte da Califórnia sendo cada dado uma leitura média por uma hora, com a primeira leitura em 1 de dezembro de 1967, a última em 2003. foi transformado em uma única série temporal em um problema de classificação definindo evento principal como qualquer leitura de mais de 5 na escala Richter. Os principais eventos são frequentemente seguidos por tremores secundários.

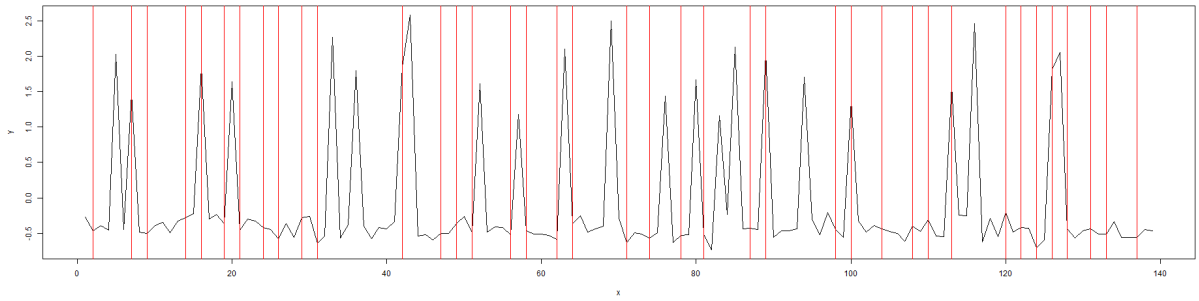


Figura 36: Série temporal Earthquakes com pontos de mudança descritos.

4.2.6 Onda Sonora

Cada série é extraída do áudio segmentado coletado do *Google Translate*, *oxforddictionaries.com* e do dicionário *online Merrriam-Webster* cada uma dessas fontes possui características diferentes. Todas as fontes possuem alto-falantes masculinos e femininos em diferentes proporções. O dicionário de *Oxford* inclui pronúncia de sotaque britânico e americano para cada palavra. Após a coleta os dados foram segmentados em formas de onda.

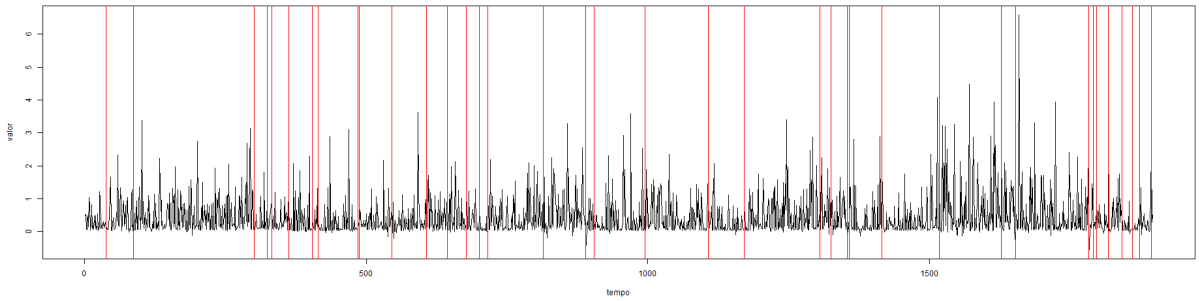


Figura 37: Série temporal Phoneme com pontos de mudança descritos.

4.2.7 Artificial

Este conjunto de dados contém exemplos de gráficos de controle gerados sinteticamente. Existem seis classes diferentes de cartas de controle: 1. Normal 2. Cíclica 3. Tendência crescente 4. Tendência decrescente 5. Mudança para cima 6. Mudança para baixo

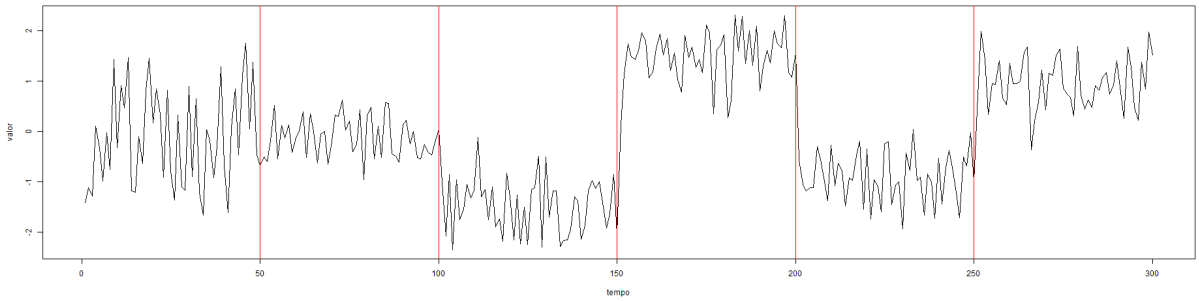


Figura 38: Série temporal SyntheticControl com pontos de mudança descritos.

4.3 TOLERÂNCIA

Para extrair outras características dos resultados comumente é aplicado uma função de tolerância (AMINIKHANGHAHI; COOK, 2016) assim é possível atribuir uma pontuação maior aos sistemas que identificaram o ponto de mudança mais próximo do real. Uma função de tolerância comum é a atribuição de pesos ao longo da série, sendo peso 1.00 no local onde se encontram os pontos de mudança e peso 0 na observação da série mais distante do ponto de mudança, os intervalos terão então pesos de 0 à 1.00 dependendo da distância que se encontram do ponto de mudança.

A técnica de tolerância demonstrada acima possui o problema de favorecer algoritmos de alta sensibilidade, pois, se identificarem muitos pontos de mudança falsos irá aumentar o resultado somando pontos ao longo da série. Uma outra técnica de tolerância é considerar como correto os pontos de mudança que estão próximos aos pontos de mudança reais, nesse caso, a partir de uma porcentagem atribuída ao tamanho da série, é selecionado a área válida e se o ponto detectado estiver dentro dessa área será considerado como verdadeiro.

Para exemplificar a técnica de tolerância, considerando uma série temporal de tamanho 60 e uma de tolerância de 0.05, a área pode ser obtida pela tolerância multiplicada pelo tamanho da série, sendo assim, $Area = 0.05 * 60$, ou seja $Area = 3$. Considerando os pontos de mudança reais da série como 20 e 40, aplicando a área de 3 os pontos considerados como verdadeiro serão 19, 20, 21 e 39, 40 e 41, como ilustrado na Figura 39.

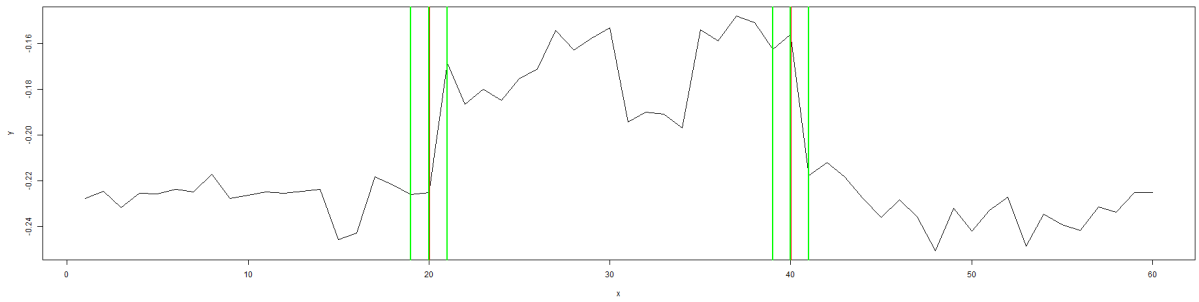


Figura 39: Pontos de Mudança do Dataset Meat e tolerância de 0.05

Caso o número da área calculada para a tolerância seja par, por exemplo, se aplicado uma tolerância de 0.10 na mesma série temporal será obtido o valor de 6 ($Area = 0.10 * 60$)

nesse caso o lado maior fica do lado esquerdo da série temporal, pois, entende-se que para a maioria dos casos é interessante que o ponto de mudança seja encontrado o mais cedo possível e os algoritmos que o fizerem devem receber maior pontuação dos que encontram o ponto tardiamente, a Figura 40 exemplifica esse caso.

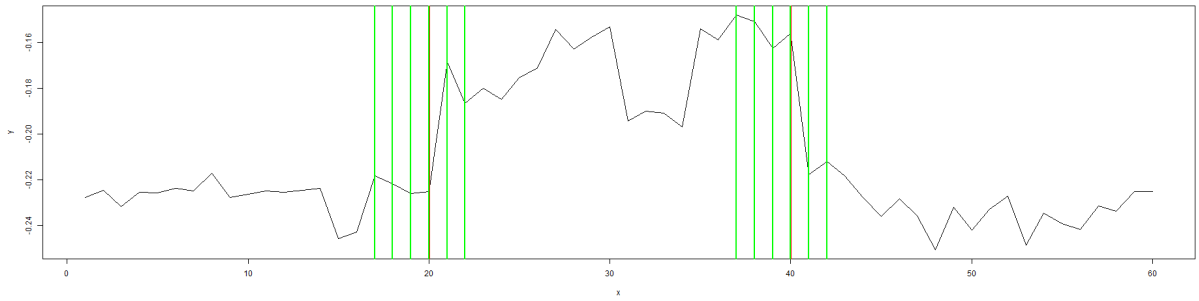


Figura 40: Pontos de Mudança do Dataset Meat e tolerância de 0.10

Para os testes desse trabalho será utilizado a segunda técnica explicada, com três porcentagens de tolerância distintas, 0.00 (sem tolerância) 0.05 e 0.10 todos os testes serão executados uma vez para cada tolerância de forma isolada para que seja possível verificar a progressão dos resultados ao aumentar o nível de tolerância.

4.4 RESULTADOS OBTIDOS

A Tabela 2 exibe uma comparação dos testes realizados em todas as bases de dados programadas nos algoritmos selecionados para esse trabalho sem tolerância, as tabelas 3 e 4 representam os resultados com 0.05 e 0.10 de tolerância respectivamente, os valores apresentação são a média aritmética de CM das séries temporais de cada *dataset*.

Para visualizar mais detalhes dos testes, é possível consultar os resultados analíticos em planilhas *excel* na página do Package no *GitHub* (UZAI, 2019) que apresenta os resultados para cada série de cada dataset sem tolerância, com tolerância de 0.05 e com tolerância de 0.10, para facilitar a visualização e discussão sobre os resultados este trabalho irá somente analisar os resultados totalizados para cada base, considerando sempre a média aritmética.

Para comparar usando todos os algoritmos, todos os testes foram feitos utilizando somente uma dimensão por vez para cada base de dados, pois, alguns algoritmos testados como o *PELT* e **EDM** não permitem calculo multidimensional, cada dimensão do algoritmo foi calculada individualmente como uma série temporal independente e seus resultados sumarizados.

Não foram usadas métricas com pesos ou ponderações, nesse caso, um Verdadeiro Negativo tem tanto peso quanto um Verdadeiro Positivo, isso pode não ser aplicável em condições reais, entretanto, é a forma mais prática para não favorecer nenhum dos algoritmos de forma arbitrária, alterações nas configurações e parâmetros de entrada do algoritmos também poderiam alterar consideravelmente os resultados, atualmente, os parâmetros e posições foram ajustados para as bases de dados mas o autor fez o esforço de deixá-los o mais próximo possível das configurações padrão.

Tabela 2: Comparação dos Resultados Obtidos pelo SpecDetec em relação a outros algoritmos - sem tolerância

	AMOC	**BinSeg**	**SegNeigh**	**PELT**	**EDivisive**	**EDM**	**MFT**	**Breakfast**	**gSeg**	**SpecDetec**	**Media**
ArrowHead	0,68	0,68	0,68	0,68	0,66	0,69	0,68	0,66	0,66	0,66	0,673
BeetleFly	0,66	0,69	0,69	0,69	0,66	0,67	0,67	0,67	0,66	**0,83**	0,689
BirdChicken	0,66	0,68	0,67	0,67	0,66	0,67	0,67	0,67	0,66	**0,88**	0,689
Coffee	0,68	0,71	0,71	0,71	0,66	0,67	0,72	0,67	0,66	**0,75**	0,694
Computers	0,68	0,67	0,67	0,66	0,66	0,67	0,67	0,67	0,67	**0,70**	0,672
DistalPhalanxOutlineAgeGroup	0,56	0,56	0,57	0,57	0,54	0,56	0,57	0,55	0,55	0,48	0,551
Earthquakes	0,64	0,65	0,66	0,69	0,64	0,64	0,64	0,64	0,64	**0,70**	0,654
ECG200	0,55	0,56	0,57	0,57	0,54	0,56	0,56	0,56	0,55	0,55	0,557
ElectricDevices	0,17	0,06	0,24	0,06	0,06	0,06	0,07	0,34	0,17	0,02	0,125
FaceAll	0,65	0,68	0,68	0,64	0,59	0,57	0,64	0,64	0,63	0,59	0,631
LargeKitchenAppliances	0,67	0,68	0,70	0,65	0,67	0,67	0,67	0,67	0,66	0,66	0,670
Meat	0,73	0,76	0,76	0,72	0,66	0,66	0,75	0,67	0,66	**0,81**	0,718
OliveOil	0,69	0,75	0,75	0,75	0,66	0,66	0,74	0,66	0,66	0,68	0,700
Phoneme	0,37	0,29	0,31	0,21	0,28	0,40	0,39	0,40	0,37	0,12	0,314
RefrigerationDevices	0,67	0,67	0,67	0,66	0,67	0,66	0,67	0,67	0,66	0,66	0,666
ScreenType	0,66	0,66	0,66	0,65	0,66	0,67	0,66	0,67	0,66	0,67	0,662
ShapeletSim	0,66	0,67	0,67	0,67	0,66	0,67	0,67	0,67	0,66	**0,72**	0,672
ShapesAll	0,63	0,65	0,66	0,77	0,62	0,67	0,65	0,63	0,63	0,73	0,664
SmallKitchenAppliances	0,67	0,67	0,66	0,67	0,66	0,68	0,70	0,67	0,66	0,66	0,670
Strawberry	0,66	0,68	0,69	0,68	0,65	0,67	0,69	0,66	0,65	0,67	0,670
syntheticControl	0,72	0,80	0,82	0,82	0,60	0,68	0,82	0,64	0,63	0,65	0,718
ToeSegmentation1	0,66	0,67	0,66	0,66	0,66	0,66	0,67	0,67	0,66	0,67	0,664
Wine	0,67	0,68	0,70	0,67	0,66	0,67	0,68	0,67	0,66	**0,91**	0,697
WormsTwoClass	0,67	0,67	0,67	0,67	0,66	0,68	0,67	0,67	0,67	0,67	0,670
Yoga	0,33	0,33	0,33	0,33	0,03	0,03	0,33	0,33	0,17	0,17	0,238
MÉDIA	**0,616**	**0,623**	**0,634**	**0,621**	**0,579**	**0,596**	**0,626**	**0,617**	**0,596**	**0,624**	**0,613**

Tabela 3: Comparação dos Resultados Obtidos pelo SpecDetec em relação a outros algoritmos - tolerância de 0.05

	AMOC	**BinSeg**	**SegNeigh**	**PELT**	**EDivisive**	**EDM**	**MFT**	**Breakfast**	**gSeg**	**SpecDetec**	**Media**
ArrowHead	0,75	0,76	0,75	0,75	0,75	0,78	0,71	0,66	0,66	**0,79**	0,736
BeetleFly	0,67	0,69	0,70	0,70	0,66	0,67	0,67	0,67	0,66	**0,92**	0,701
BirdChicken	0,66	0,68	0,68	0,68	0,66	0,67	0,67	0,67	0,66	**0,94**	0,697
Coffee	0,68	0,72	0,72	0,72	0,66	0,67	0,74	0,67	0,66	**0,91**	0,715
Computers	0,70	0,70	0,72	0,69	0,71	0,70	0,67	0,67	0,67	**0,89**	0,712
DistalPhalanxOutlineAgeGroup	0,66	0,68	0,70	0,70	0,70	0,70	0,62	0,55	0,60	**0,72**	0,663
Earthquakes	0,70	0,73	0,81	0,91	0,70	0,64	0,64	0,64	0,67	0,87	0,731
ECG200	0,58	0,58	0,63	0,64	0,61	0,60	0,57	0,56	0,60	**0,75**	0,612
ElectricDevices	0,28	0,35	0,30	0,12	0,47	0,34	0,42	0,34	0,24	0,36	0,322
FaceAll	0,68	0,80	0,78	0,78	0,80	0,71	0,81	0,64	0,68	**0,83**	0,751
LargeKitchenAppliances	0,71	0,74	0,76	0,66	0,70	0,67	0,67	0,67	0,66	**0,79**	0,703
Meat	0,75	0,80	0,80	0,76	0,66	0,66	0,79	0,67	0,66	**0,81**	0,736
OliveOil	0,71	0,81	0,82	0,82	0,66	0,66	0,80	0,66	0,66	**0,84**	0,744
Phoneme	0,44	0,55	0,53	0,60	0,51	0,40	0,42	0,40	0,42	0,47	0,474
RefrigerationDevices	0,67	0,67	0,74	0,70	0,70	0,74	0,67	0,67	0,66	0,66	0,688
ScreenType	0,68	0,70	0,71	0,68	0,69	0,70	0,67	0,67	0,66	**0,82**	0,698
ShapeletSim	0,66	0,67	0,67	0,67	0,66	0,72	0,67	0,67	0,66	**0,91**	0,696
ShapesAll	0,68	0,77	0,78	0,94	0,92	0,96	0,87	0,63	0,66	0,87	0,808
SmallKitchenAppliances	0,71	0,75	0,76	0,71	0,75	0,68	0,71	0,67	0,66	0,76	0,716
Strawberry	0,68	0,80	0,79	0,79	0,79	0,79	0,78	0,66	0,65	**0,87**	0,760
syntheticControl	0,73	0,86	0,89	0,88	0,77	0,84	0,87	0,64	0,63	0,85	0,796
ToeSegmentation1	0,66	0,67	0,67	0,67	0,66	0,71	0,67	0,67	0,66	0,87	0,691
Wine	0,67	0,69	0,72	0,68	0,66	0,67	0,68	0,67	0,66	**0,97**	0,707
WormsTwoClass	00,67	0,67	0,67	0,67	0,66	0,75	0,68	0,67	0,67	0,72	0,683
Yoga	0,33	0,33	0,33	0,33	0,18	0,65	0,33	0,33	0,21	0,44	0,346
MÉDIA	**0,644**	**0,687**	**0,697**	**0,690**	**0,668**	**0,683**	**0,672**	**0,617**	**0,611**	**0,785**	**0,675**

Tabela 4: Comparação dos Resultados Obtidos pelo SpecDetec em relação a outros algoritmos - tolerância de 0.10

	AMOC	**BinSeg**	**SegNeigh**	**PELT**	**EDivisive**	**EDM**	**MFT**	**Breakfast**	**gSeg**	**SpecDetec**	**Media**
ArrowHead	0,75	0,77	0,78	0,77	0,77	0,80	0,71	0,66	0,66	0,79	0,746
BeetleFly	0,67	0,69	0,71	0,71	0,66	0,67	0,67	0,67	0,66	**0,94**	0,705
BirdChicken	0,66	0,69	0,69	0,69	0,66	0,67	0,67	0,67	0,66	**0,96**	0,702
Coffee	0,68	0,73	0,74	0,74	0,66	0,67	0,75	0,67	0,66	**0,92**	0,722
Computers	0,71	0,71	0,75	0,70	0,73	0,72	0,67	0,67	0,67	**0,89**	0,722
DistalPhalanxOutlineAgeGroup	0,67	0,68	0,72	0,73	0,72	0,77	0,63	0,55	0,64	0,72	0,683
Earthquakes	0,74	0,78	0,87	0,93	0,77	0,64	0,65	0,64	0,70	0,87	0,759
ECG200	0,62	0,59	0,68	0,68	0,67	0,62	0,57	0,56	0,62	**0,75**	0,636
ElectricDevices	0,33	0,44	0,45	0,12	0,47	0,34	0,55	0,34	0,28	0,36	0,368
FaceAll	0,74	0,88	0,88	0,80	0,82	0,72	0,87	0,64	0,68	0,83	0,786
LargeKitchenAppliances	0,73	0,76	0,80	0,67	0,71	0,67	0,67	0,67	0,66	**0,83**	0,717
Meat	0,88	0,83	0,84	0,78	0,66	0,66	0,83	0,67	0,66	**0,89**	0,770
OliveOil	0,73	0,82	0,84	0,84	0,66	0,66	0,81	0,66	0,66	**0,88**	0,756
Phoneme	0,53	0,67	0,66	0,65	0,63	0,40	0,46	0,40	0,42	0,47	0,529
RefrigerationDevices	0,67	0,67	0,76	0,70	0,71	0,77	0,67	0,67	0,66	0,66	0,694
ScreenType	0,70	0,72	0,74	0,69	0,70	0,70	0,71	0,67	0,66	**0,88**	0,717
ShapeletSim	0,66	0,67	0,67	0,67	0,67	0,77	0,68	0,67	0,66	**0,91**	0,703
ShapesAll	0,72	0,83	0,84	0,95	0,95	0,96	0,94	0,63	0,69	0,87	0,838
SmallKitchenAppliances	0,77	0,79	0,80	0,71	0,77	0,68	0,71	0,67	0,66	0,79	0,735
Strawberry	0,71	0,85	0,84	0,81	0,86	0,85	0,84	0,66	0,78	**0,88**	0,808
syntheticControl	0,73	0,86	0,89	0,88	0,78	0,85	0,87	0,64	0,63	0,86	0,799
ToeSegmentation1	0,66	0,67	0,68	0,68	0,66	0,73	0,67	0,67	0,66	**0,89**	0,697
Wine	0,67	0,69	0,72	0,68	0,66	0,67	0,68	0,67	0,66	**0,97**	0,707
WormsTwoClass	0,67	0,67	0,68	0,68	0,66	0,78	0,69	0,67	0,67	**0,87**	0,704
Yoga	0,33	0,33	0,33	0,33	0,17	0,65	0,33	0,33	0,26	0,44	0,350
MÉDIA	**0,669**	**0,712**	**0,734**	**0,704**	**0,687**	**0,697**	**0,692**	**0,617**	**0,625**	**0,805**	**0,694**

4.4.1 TESTE DE WILCOXON

Para verificar se o *SpecDetec* e, consequentemente o Espectro do Grafo possui performance (*CM*) significativamente superior aos outros métodos, foi realizado o teste de *Wilcoxon* pareado. Foram consideradas as hipóteses nula e alternativa para os resultados cálculos em todas as tolerâncias sendo: HO (O algoritmo testado possui performance superior ao *SpecDetec*) e HA: rejeitar H0 se $p < 0,05$ (O *SpecDetec* tem performance superior ao algoritmo testado). O valor selecionado de $0,05$ é o valor de significância, ou seja, os dados representam uma mudança suficientemente significante entre si.

Os resultados sumarizados do teste estão visíveis na Tabela 5 na qual é apresentado o número de vezes que o *SpecDetec* foi superior aos outros algoritmos considerando o teste de *Wilcoxon* pareado, ou seja, em quantas bases de dados H0 foi rejeitada ou afirmada. Em sequência a tabela 6 e 7 repetem os testes considerando os resultados de 0.05 e 0.10 de tolerância respectivamente.

Tabela 5: Resultados sumarizados do teste de Wilcoxon em relação ao método proposto Spec-Detec - sem tolerância

	AMOC	**BinSeg**	**SegNeigh**	**PELT**	**EDivisive**	**EDM**	**MFT**	**Breakfast**	**gSeg**
Vence	10	10	12	11	18	11	7	7	8
Empata	0	0	0	0	0	0	0	0	0
Perde	15	15	13	14	7	14	18	18	17

Tabela 6: Resultados sumarizados do teste de Wilcoxon em relação ao método proposto Spec-Detec - tolerância de 0.05

	AMOC	**BinSeg**	**SegNeigh**	**PELT**	**EDivisive**	**EDM**	**MFT**	**Breakfast**	**gSeg**
Vence	21	18	19	18	19	20	19	21	21
Empata	0	0	0	0	0	0	0	0	0
Perde	4	7	6	7	6	5	6	4	4

Tabela 7: Resultados sumarizados do teste de Wilcoxon em relação ao método proposto Spec-Detec - tolerância de 0.10

	AMOC	**BinSeg**	**SegNeigh**	**PELT**	**EDivisive**	**EDM**	**MFT**	**Breakfast**	**gSeg**
Vence	21	18	16	17	19	18	18	22	22
Empata	0	0	0	0	0	0	0	0	0
Perde	4	7	9	8	6	7	7	3	3

4.5 ANÁLISE E DISCUSSÕES

Nesta sessão será discutido os resultados em cada dataset e sua importância em relação ao contexto da série temporal, também será discutido os resultados gerais comparáveis entre os algoritmos.

A Figura 45 exibe os resultados sumarizados do *dataset ArrowHead*, é possível verificar que sem considerar a tolerância os resultados foram próximos, sendo o mais baixo com 0.66 e o mais alto 0.69, o *SpecDetec* não se destaca em relação aos outros algoritmos tendo o mesmo resultado mínimo que Breakfast, *e-Divisive* e gSeg.

Uma informação importante que pode ser observada analisando as tolerância é que mesmo aumentando a tolerância para 0.05 ou 0.10 alguns algoritmos não obtiveram mudança no resultado, como é o caso do Breakfast e do gSeg, isso é causado na maioria dos casos porque alguns pontos de mudança não são detectados como tal para o algoritmo, sendo assim, mesmo que a tolerância aumente não haverá mudança nos resultados. Por outro lado, alguns algoritmos que são mais sensíveis a pontos de mudança se beneficiam muito da tolerância, o SpecDetec atingiu o melhor resultado (0.79) considerando tolerância de 0.05 mesmo tendo ficado entre os resultados mais baixos sem a aplicação de tolerância.

Algoritmos que, em um determinado *dataset* são muito eficientes e identificar o número de pontos de mudança em uma série temporal mas não tão eficientes para identificar a posição exata do ponto de mudança irão se beneficiar particularmente mais das tolerâncias do que algoritmos que tem precisão sobre a posição exata dos pontos de mudança mas, falham em identificar todos os pontos de mudança existentes na série temporal. Esse mesmo resultado se repete em outros datasets, mesmo com contextos de aplicações diferentes, como demonstrado nas Figuras 46, 47, 48, 49, 50, 51, 52, 53, 54, 55 e 56.

Em datasets de maior flutuação de valores e diferenças de padrão, como por exemplo eletrocardiogramas, variação de circuitos elétricos, espectrogramas de alimentos, é facilmente verificável a existência de uma mudança de padrão na série temporal, o problema para os algoritmos é encontrar o ponto onde a mudança de padrão começa a ocorrer com precisão (ponto de mudança), nesses casos a tolerância apresenta um papel ainda mais significativo na diferença dos resultados, e, o algoritmo *SpecDetec* se destaca particularmente, reforçando a hipótese de que existe a tendência de ter alta sensibilidade para detecção de mudança de padrões em séries temporais, e caso exista uma flexibilização na precisão os resultados são muito superiores a média geral como demonstrado nas Figuras 57, 58, 59, 60, 60 e 61.

É possível observar também que o em alguns datasets independente da tolerância, o SpecDetec se saiu consideravelmente superior, e nesse caso pode ser um algoritmo recomendado para uso nesses contextos, em especial na variação de contornos de insetos e aves ou nos espectrograma alimentares como demonstrado nas Figuras 62, 58, 64, 65 e 66. É importante salientar que são resultados que não podem ser atingidos com os métodos atuais e potencialmente novas possibilidades de aplicações nesses contextos podem emergir.

Em alguns casos o SpecDetec não conseguiu atingir a média ou teve resultados muito inferiores em relação ao algoritmo de melhor resultado, isso ocorre porque apesar da posição dos pontos de mudança ser evidente, se divido a série temporal em grupos é possível notar que não há uma variação da distribuição de dados tão significativa entre os grupos, e por tanto, o SpecDetec não é eficiente em detecta-los como clusters independentes e por tanto, não detecta seus pontos de mudança, esses resultados podem ser aferidos nas Figuras 67, 68 e 69.

4.5.1 ANÁLISE GERAL

Conforme demonstrado o SpecDetec obteve resultados muito superiores em algumas séries mas ficou próximo da média sem tolerância considerando o resultado geral, porém, aplicando 0.05

de tolerância foi superior aos outros algoritmos em 15 das 25 séries, também foi 0.161 superior a média geral dos outros algoritmos sendo 0.126 superior ao segundo melhor resultado como demonstrado na Figura 41. A diferença dos entre o SpecDetec e os outros algoritmos diminui ao aplicar uma tolerância de 0.10 mas ainda é superior a todos os outros em 14 das 25 séries temporais, sendo também 0.159 superior a média geral e 0.096 superior ao segundo melhor resultado.

Séries com menos pontos de mudança são mais fáceis de se detectar pois é apenas uma mudança de padrão dentro da série temporal, porém, em caso de falha é uma falha completa (0.00 de AC) em contrapartida séries temporais com vários pontos de mudança mesmo que existam falhas na detecção ainda é possível obter um índice superior a média 0.60 acertando parte dos pontos dispostos na série. Esse tipo de distinção deve ser observado ao comparar os dados em séries com números muito distintos de pontos de mudança, principalmente levando em consideração a aplicação de tolerância.

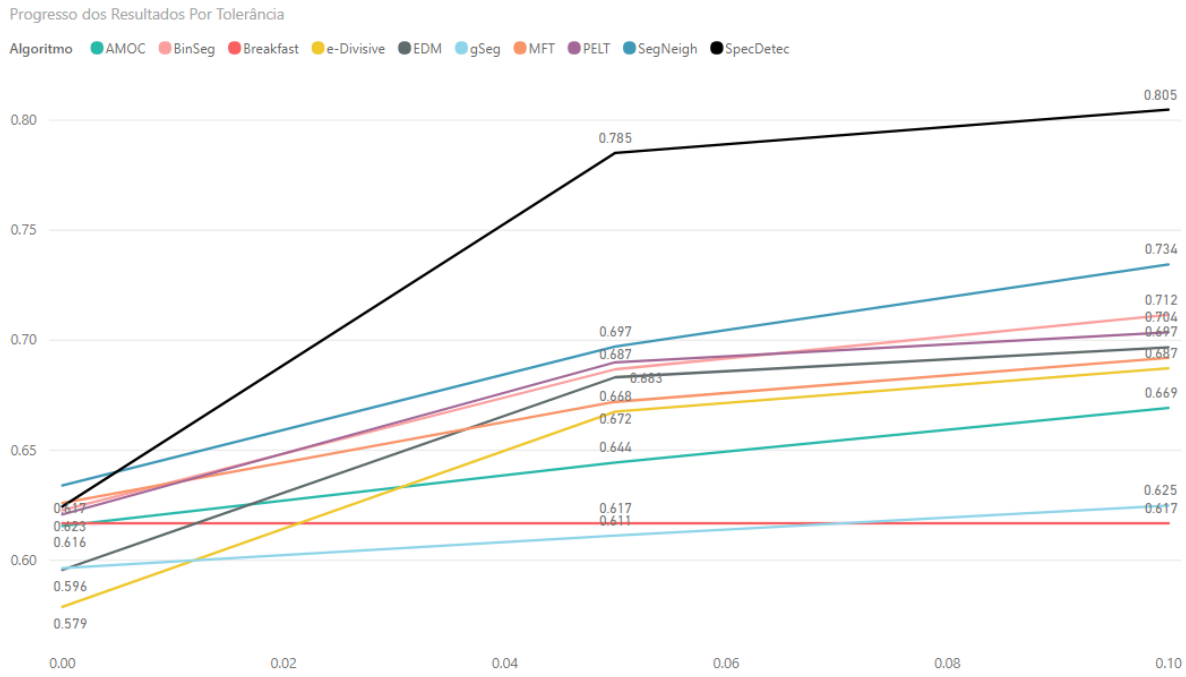


Figura 41: Progresso dos Resultados Por Tolerância

5 CONCLUSÕES

O principal objetivo deste trabalho foi investigar uma metodologia computacional para detectar múltiplos pontos de mudança em séries temporais multi-valoradas com maior precisão possível. Para resolução do problema foi investigado o estado da arte sobre detecção de pontos de mudança, todos os tipos de algoritmo existentes e quais são os algoritmos usados atualmente para setor. Constatou-se durante a revisão da literatura que métodos baseadas em grafos tiveram resultados promissores, porém, foram pouco explorados.

Foi verificado durante análise da literatura, métodos baseados em grafos tem diversas aplicações práticas na computação e tem seu destaque especial na segmentação de imagens, como o problema de detecção de pontos de mudança também pode ser resumido em um problema de segmentação (sub-dividir partes dentro de uma mesma série temporal, onde, o ponto de mudança é o extremidade entre as partes), surgiu a hipótese de que os métodos baseados em grafos e aplicados com eficiência na segmentação de imagens podem ser portados para a detecção de pontos de mudança, desde que, se considere as devidas adaptações.

Entre os métodos baseados em grafos, segmentação com o espectro do grafo foi escolhido pois a partir de funções matemáticas polinomiais é possível extrair uma série de informações sobre a topologia do grafo o que se mostrou extremamente eficiente para segmentar imagens (CASACA, 2014). Esperava-se então que com as informações sobre a topologia do grafo fosse possível identificar as variações estatísticas de segmentos diversos da série temporal e assim, identificar as ramificações de menor relação, e que essa informação poderia ser usada para identificar pontos de mudança.

Assim como na segmentação de imagens, o menor autovalor de um grafo é o valor de menor relação entre os pontos e provavelmente é a margem, esse entendimento foi estendido neste trabalho a pontos de mudança, considerando que o menor autovalor de um grafo gerado a partir de uma série temporal representa o ponto de mudança. Para aplicar esse conceito e verificar se a detecção de pontos de mudança utilizando esse método é tão eficiente ou mais que o estado da arte, foi desenvolvido um *Package* em R nomeado *SpecDetec* que, recebe como entrada uma série temporal e retorna como saída a posição dos pontos de mudança identificados ao longo da série.

Para verificar a eficiência do SpecDetec e da técnica, foram analisadas 11262 séries temporais provindas de 25 datasets de contextos e estruturas diferentes, os testes foram repetidos com outros 9 algoritmos que representam o estado da arte em detecção de pontos de mudança. Os resultados foram muito promissores, como exibido na Tabela 2 o algoritmo se equipara ao estado da arte obtendo resultado próximo a média geral na maioria dos casos, e, em alguns casos em particular como na detecção da variação de contornos de insetos e aves nos datasets BirdChicken e *BeetleFly* o resultado do SpecDetec foi consideravelmente superior a todos os outros algorítimos.

Os testes foram repetidos mais duas vezes para todos os algoritmos aplicando uma tolerância de 0.05 e 0.10 em relação ao tamanho da série, e, como exibido na Figura 41 aumentar a tolerância aumenta o resultado geral de todos os algoritmos, porém, beneficia particularmente o SpecDetec que se distancia largamente dos outros resultados com a tolerância de 0.05, isso se deve pois, apesar de em alguns casos não atingir a mesma precisão que os outros algoritmos sobre a posição do ponto de mudança, o SpecDetec é muito eficiente em identificar o real número de pontos de mudança dentro da série temporal, por isso, conforme a precisão é afrouxada pela tolerância a sensibilidade para detectar pontos de mudança influência mais nos resultados.

Dado os resultados, conclui-se que segmentação com espectro do grafo é uma alternativa valida e eficiente para detecção de pontos de mudança, o package SpecDetec é recomendado para detecção de pontos de mudança em contextos de detecção de objetos, diferenciação de alimentos por espectrograma, ou detecção de aparelhos por consumo energético entre outros cenários de contexto similar, também é recomendado para cenários onde o mais importante é detectar o número real de pontos de mudança na série temporal não sendo primordial identificar a posição do ponto de mudança com exatidão.

5.1 CONTRIBUIÇÕES

Além de indicar experimentalmente que é possível de forma eficiente a detecção de pontos de mudança com o espectro do grafo, o desenvolvimento deste trabalho possibilitou a exploração de diversos algoritmos de detecção de pontos de mudança aplicados a contexto distintos que podem ser utilizados como guia para trabalhos futuros.

Foi desenvolvido um *package* em R e disponibilizado para a detecção de pontos de mudança, foi também realizado a publicação do algoritmo e seus testes preliminares em (UZAI; KASHIWABARA, 2018) já considerando o SpecDetec como uma alternativa viável para detecção de múltiplos pontos de mudança em séries temporais. Este trabalho e a publicação indicam também que para alguns contextos se chegou a resultados que antes não eram possíveis considerando os algoritmos disponíveis, sendo possível que novos trabalham surjam se beneficiando dos novos resultados nesses contextos de aplicação.

Foram desenvolvidos diversos scripts para facilitar a reprodução dos resultados presentes nesse trabalho, todos se encontram versionados em uma conta do *GitHub* (UZAI, 2019), esses scripts também podem ser utilizados para insumo para que novos desenvolvedores usem e testem novos algoritmos de detecção de pontos de mudança, já que, scripts que preparam as bases de dados e parametrizam algoritmos de comparação já estão prontos e, mesmo que sofram adaptações não há a necessidade de desenvolver novos scripts desde o começo.

5.2 TRABALHOS FUTUROS

Para trabalhos futuros é possível que testes sejam refeitos considerando somente algoritmos de detecção de múltiplas dimensões, o SpecDetec pode ser customizado para a detecção de pontos de mudança considerando múltiplas dimensões, porém, há poucos algoritmos que suportam essa característica e os testes devem ser feitos exclusivamente com esses algoritmos.

Também é interessante que novos algoritmos de agrupamento sejam implementados como opção para o *SpecDetec* o único disponível hoje é o *k-means*, sendo que, muitos outros

poderiam ser utilizados. Outro ponto considerando o código do *package* que pode ser expandida é sobre as medidas de similaridade, que, também podem ser diversas e hoje a única diretamente implementada é a gaussiana.

Como indicado no decorrer do trabalho, o *SpecDetec* é muito eficiente em detectar a quantidade de pontos de mudança na série temporal, porém, como não possui uma precisão tão alta para a detecção da real posição dos pontos de mudança como alguns algoritmos do estado da arte. É preciso continuar modificando a implementação e realizar novos testes para tentar alcançar níveis mais altos de precisão sem perder a habilidade de detecção da quantidade real de pontos de mudança no decorrer da série temporal.

Por fim, a aplicação até então do SpecDetec é apenas teórica e, apesar de base de dados de cenários reais tenham sido utilizadas, deseja-se que o SpecDetec seja testado em cenários reais. Existe a expectativa de sua utilização em bases biológicas, em particular, análise genômica de variação do número de cópias de genes (ZARREI et al., 2015), que pode proporcionar uma série de aplicações médicas, como identificação de células cancerígenas (GIANNOUDIS et al., 2017) e doenças relacionadas a demência (BUTCHER et al., 2017).

REFERÊNCIAS

ABREU, N. M. M. de et al. Teoria Espectral de Grafos - Uma Introdução. p. 201, 2014. Disponível em: <http://mtm.ufsc.br/coloquiosul/notas_minicurso_6.pdf>.

AMINIKHANGHAHI, S.; COOK, D. J. A survey of methods for time series change point detection. *Knowledge and Information Systems*, p. 1–29, 2016. ISSN 02193116.

ANTIQUEIRA, L. et al. Modelando textos como redes complexas. In: *Anais do III Workshop em Tecnologia da Informação e da Linguagem Humana*. [S.l.: s.n.], 2005. p. 22–26.

AUGER, I. E.; LAWRENCE, C. E. Algorithms for the optimal identification of segment neighborhoods. *Bulletin of mathematical biology*, Springer, v. 51, n. 1, p. 39–54, 1989.

BASSEVILLE, M.; NIKIFOROV, I. *Detection of abrupt changes. Theory and application*. [s.n.], 1993. ISSN 09670661. ISBN 0-13-126780-9. Disponível em: <http://linkinghub.elsevier.com/retrieve/pii/0967066194901961>.

BONSAL, B. R. et al. Characteristics of daily and extreme temperatures over Canada. *Journal of Climate*, v. 14, n. 9, p. 1959–1976, 2001. ISSN 08948755.

BOSC, M. et al. Automatic change detection in multimodal serial MRI: Application to multiple sclerosis lesion evolution. *NeuroImage*, v. 20, n. 2, p. 643–656, 2003. ISSN 10538119.

BUTCHER, N. et al. Sulfotransferase 1a3/4 copy number variation is associated with neurodegenerative disease. *The pharmacogenomics journal*, Nature Publishing Group, 2017.

CASACA, W. C. d. O. *Graph laplacian for spectral clustering and seeded image segmentation*. Tese (Doutorado) — Universidade de São Paulo, 2014.

CETINKAYA-RUNDEL, M. et al. user! 2017. *Dose-response*, v. 9, p. 30am.

CHAKRABARTI, K. et al. Locally adaptive dimensionality reduction for indexing large time series databases. *ACM Transactions on Database Systems*, v. 27, n. 2, p. 188–228, 2002. ISSN 03625915. Disponível em: <http://portal.acm.org/citation.cfm?doid=568518.568520>.

CHANDOLA, V.; BANERJEE, A.; KUMAR, V. Anomaly detection: A survey. *ACM Computing Surveys (CSUR)*, v. 41, n. September, p. 1–58, 2009. ISSN 0360-0300.

CHEBOLI, D. Anomaly detection on time series. *2010 IEEE International Conference on Progress in Informatics and Computing*, v. 1, p. 603–608, 2010. Disponível em: <http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=5687485>.

CHEN, H.; ZHANG, N. Graph-based change-point detection. *Annals of Statistics*, v. 43, n. 1, p. 139–176, 2015. ISSN 00905364.

CHEN, Y. et al. *The UCR Time Series Classification Archive*. July 2015. www.cs.ucr.edu/ eamonn/time$_s eries_d ata/$.

COOK, D.; KRISHNAN, N. *Activity Learning*. [S.l.: s.n.], 2015.

COSTA, L. d. F. et al. Characterization of complex networks: A survey of measurements. *Advances in physics*, Taylor & Francis, v. 56, n. 1, p. 167–242, 2007.

DOWNEY, A. A novel changepoint detection algorithm. *arXiv preprint arXiv:0812.1237*, 2008. Disponível em: <http://arxiv.org/abs/0812.1237>.

DUCRÉ-ROBITAILLE, J. F.; VINCENT, L. A.; BOULET, G. Comparison of techniques for detection of discontinuities in temperature series. *International Journal of Climatology*, v. 23, n. 9, p. 1087–1101, 2003. ISSN 08998418.

EAGLE, N.; PENTLAND, A. S.; LAZER, D. Inferring friendship network structure by using mobile phone data. *Proceedings of the national academy of sciences*, National Acad Sciences, v. 106, n. 36, p. 15274–15278, 2009.

FRITSCHER, E. Propriedades espectrais de um grafo. 2011.

FRYZLEWICZ, P. Tail-greedy bottom-up data decompositions and fast multiple change-point detection. p. 1–33, 2016.

FUJITA, A. et al. Correlation between graphs with an application to brain network analysis. *Computational Statistics and Data Analysis*, Elsevier B.V., v. 109, n. xxxx, p. 76–92, 2017. ISSN 01679473. Disponível em: <http://dx.doi.org/10.1016/j.csda.2016.11.016>.

GIANNOUDIS, A. et al. *Abstract P2-03-04: Application of digital-PCR technology to determine c-MET copy number variation in paired primary breast cancer and brain metastases*. [S.l.]: AACR, 2017.

ITOH, N.; KURTHS, J. Change-Point Detection of Climate Time Series by Nonparametric Method. In: *Proceedings of the World Congress on Engineering and Computer Science*. [S.l.: s.n.], 2010. I, n. 1, p. 20–23. ISBN 9789881701206.

JACKSON, B. et al. An algorithm for optimal partitioning of data on an interval. *IEEE Signal Processing Letters*, IEEE, v. 12, n. 2, p. 105–108, 2005.

JAMES, N. A.; KEJARIWAL, A.; MATTESON, D. S. In: *Proceedings - 2016 IEEE International Conference on Big Data, Big Data 2016*. [S.l.: s.n.], 2016. p. 3499–3508. ISBN 9781467390040.

JODOIN, P.-M. *Challenge evaluation metrics*. Jun 2012. http://jacarini.dinf.usherbrooke.ca/resultEvaluation/.

KAWAHARA, Y.; SUGIYAMA, M. Sequential change-point detection based on direct density-ratio estimation. *Wiley Periodicals, Inc*, v. 5, n. 2, p. 114–127, 2009.

KEOGH, E.; KASETTY, S. On the need for time series data mining benchmarks: a survey and empirical demonstration. *Data Mining and knowledge discovery*, Springer, v. 7, n. 4, p. 349–371, 2003.

KILLICK, R.; ECKLEY, I. changepoint: An R Package for changepoint analysis. *Lancaster University*, v. 58, n. 3, p. 1–15, 2013. ISSN 1548-7660. Disponível em: <https://www.jstatsoft.org/article/view/v058i03/v58i03.pdf>.

KILLICK, R.; FEARNHEAD, P.; ECKLEY, I. A. Optimal detection of changepoints with a linear computational cost. *Journal of the American Statistical Association*, v. 107, n. 500, p. 1590–1598, 2012. ISSN 01621459.

LAI, T. L. Sequential changepoint detection in quality control and dynamical systems. *Journal of the Royal Statistical Society*, v. 47, n. 3, p. 429–437, 2014.

LEITHOLD, L. *El cálculo*. [S.l.]: Oxford University Press Harla, 1998.

LI, S.; XU, L. D.; ZHAO, S. The internet of things: a survey. *Information Systems Frontiers*, v. 17, n. 2, p. 243–259, 2015. ISSN 13873326.

LUXBURG, U. V. A Tutorial on Spectral Clustering. p. 1–32, 2007.

MA, J.; PERKINS, S. Time-series novelty detection using one-class support vector machines. In: *Proceedings of the International Joint Conference on Neural Networks, 2003.* [S.l.: s.n.], 2003. v. 3, p. 1741–1745. ISBN 0-7803-7898-9. ISSN 1098-7576.

MAGAKIN, M. *Change Point detection with seasonal time series*. February 2016. https://anomaly.io/change-point-detection-seasonal/.

MALLADI, R.; KALAMANGALAM, G. P.; AAZHANG, B. Online Bayesian change point detection algorithms for segmentation of epileptic activity. In: *Conference Record - Asilomar Conference on Signals, Systems and Computers*. [S.l.: s.n.], 2013. p. 1833–1837. ISBN 9781479923908. ISSN 10586393.

MATTESON, D. S.; JAMES, N. A. *Journal of the American Statistical Association*, v. 109, n. 505, p. 334–345, 2014. ISSN 1537274X.

MESSER, M. et al. A multiple filter test for the detection of rate changes in renewal processes with varying variance. *Annals of Applied Statistics*, v. 8, n. 4, p. 2027–2067, 2014. ISSN 19417330.

METZ, J. Redes complexas: conceitos e aplicações. 2007.

MOREIRA, F. d. S. Detecção de pontos de mudança em séries temporais utilizando uma formulação Neural/Fuzzy/Bayesiana: Aplicaçao na detecção de falhas. *Dissertação Mestrado - UFMG*, p. 69, 2011.

NETO, J. *Spectral Clustering*. December 2013. http://www.di.fc.ul.pt/jpn/r/spectralclustering/.

NEWMAN, M. E. The structure and function of complex networks. *SIAM review*, SIAM, v. 45, n. 2, p. 167–256, 2003.

NG, A. Y.; JORDAN, M. I.; WEISS, Y. On spectral clustering: Analysis and an algorithm. In: *Advances in neural information processing systems*. [S.l.: s.n.], 2002. p. 849–856.

OBADI, G. et al. Using spectral clustering for finding students' patterns of behavior in social networks. In: CITESEER. *DATESO*. [S.l.], 2010. p. 118–130.

OMRANIAN, N.; MUELLER-ROEBER, B.; NIKOLOSKI, Z. Segmentation of biological multivariate time-series data. *Scientific Reports*, v. 5, p. 1–6, 2015. ISSN 20452322.

PAGE, E. S. A Test for a Change in a Parameter Occurring at an Unknown Point. *Biometrika*, v. 42, n. 3/4, p. 523, 1955. ISSN 00063444. Disponível em: <http://www.jstor.org/stable/2333401?origin=crossref>.

PAULHEIM, H. Knowledge graph refinement: A survey of approaches and evaluation methods. *Semantic web*, IOS Press, v. 8, n. 3, p. 489–508, 2017.

POLUNCHENKO, A. S.; TARTAKOVSKY, A. G. *State-of-the-Art in Sequential Change-Point Detection*. [S.l.: s.n.], 2012. 649–684 p. ISSN 13875841. ISBN 1100901192565.

RADKE, R. et al. Image change detection algorithms: a systematic survey. *IEEE Transactions on Image Processing*, v. 14, n. 3, p. 294–307, 2005. ISSN 1057-7149. Disponível em: <http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=1395984>.

RAKTHANMANON, T. et al. Addressing big data time series: Mining trillions of time series subsequences under dynamic time warping. *ACM Transactions on Knowledge Discovery from Data (TKDD)*, ACM, v. 7, n. 3, p. 10, 2013.

RAYKAR, V. C. Scalable machine learning for massive datasets : Fast summation algorithms vikas chandrakant raykar doctor of philosophy , 2007 professor dr . ramani duraiswami department of computer science scalable machine learning for massive datasets : Fast summation. 2007.

REEVES, J. et al. A review and comparison of changepoint detection techniques for climate data. *Journal of Applied Meteorology and Climatology*, v. 46, n. 6, p. 900–915, 2007. ISSN 15588424.

SANTOS, S. de S. et al. Coga: an r package to identify differentially co-expressed gene sets by analyzing the graph spectra. *PloS one*, Public Library of Science, v. 10, n. 8, p. e0135831, 2015.

SASIDHARAN, R. Spectral clustering of protein sequences using sequence-profile scores. 2006.

SCOTT, a. J.; KNOTT, M. A Cluster Analysis Method for Grouping Means in the Analysis of Variance. *Biometrics*, v. 30, n. 3, p. 507–512, 1974. ISSN 0006341X. Disponível em: <http://www.jstor.org/stable/2529204%5Cnpapers2://publication/uuid/6726893B-EB81-4C7D-83E1-D39FF77376E5>.

SHU L., C. A. X. M. . M. W. Efficient spectral neighborhood blocking for entity resolution. *IEEE 27th International Conference on Data Engineeringg*, IEEE, p. 1078, 2011.

SPIELMAN, D. A. Spectral graph theory and its applications. In: IEEE. *Foundations of Computer Science, 2007. FOCS'07. 48th Annual IEEE Symposium on*. [S.l.], 2007. p. 29–38.

STAUDACHER, M. et al. A new method for change-point detection developed for on-line analysis of the heart beat variability during sleep. *Physica A: Statistical Mechanics and its Applications*, v. 349, n. 3-4, p. 582–596, 2005. ISSN 03784371.

SURYANARAYANA, S.; RAO, G. V.; SWAMY, G. V. A survey: Spectral clustering applications and its enhancements. *International Journal of Computer Science and Information Technologies*, v. 6, 2015.

TRIPATHY, S.; SAHOO, P. L. A Survey of different methods of clustering for anomaly detection. *International Journal of Scientific & Engineering Research*, v. 6, n. 1, p. 351–357, 2015.

TRIVEDI, S. Spectral clustering in educational data mining. *Educational Data Mining*, Educational Data Mining, 2011.

TSIRIGOS, A.; RIGOUTSOS, I. A new computational method for the detection of horizontal gene transfer events. *Nucleic acids research*, Oxford University Press, v. 33, n. 3, p. 922–933, 2005.

TUNG, F.; WONG, A.; CLAUSI, D. A. Enabling scalable spectral clustering for image segmentation. *Pattern Recognition*, Elsevier, v. 43, n. 12, p. 4069–4076, 2010.

UZAI, L. *Código Fonte Compartilhado no Git Hub*. January 2019. https://github.com/lguzai/Spec/.

UZAI, L. G. C.; KASHIWABARA, A. Y. Using graph spectral to solve change point detection problems. p. 716–727, 2018.

VALLIS, O.; HOCHENBAUM, J.; KEJARIWAL, A. A Novel Technique for Long-term Anomaly Detection in the Cloud. In: *Proccedings of the 6th USENIX Workshop on Hot Topics in Cloud Computing (USENIX '14)*. [S.l.: s.n.], 2014. p. 1–6.

VIDAL, M. C. et al. Anocva in r: A software to compare clusters between groups and its application to the study of autism spectrum disorder. *Frontiers in neuroscience*, Frontiers, v. 11, p. 16, 2017.

WEI, L. et al. Assumption-free anomaly detection in time series. In: *Proceedings of the 17th international conference on Scientific and statistical database management*. [S.l.: s.n.], 2005. p. 1–4. ISBN 188888111X.

YANG, P.; DUMONT, G.; ANSERMINO, J. M. Adaptive change detection in heart rate trend monitoring in anesthetized children. *IEEE Transactions on Biomedical Engineering*, v. 53, n. 11, p. 2211–2219, 2006. ISSN 00189294.

YONETANIL, T.; MCCABE, G. J. Abrupt changes in regional temperature in the conternimous United States, 1895-1989. *Climate Research*, v. 4, p. 13–23, 1994.

ZARREI, M. et al. A copy number variation map of the human genome. *Nature Reviews Genetics*, Nature Publishing Group, v. 16, n. 3, p. 172, 2015.

ZHAO, L. et al. Onset of traffic congestion in complex networks. *Physical Review E*, APS, v. 71, n. 2, p. 026125, 2005.

A APÊNDICE - ARTIGO PUBLICADO NO ENIAC 2018

Using Graph Spectral to solve Change Point Detection Problems

Luis Gustavo C. Uzai[1], **André Y. Kashiwabara**[1]

[1]Universidade Tecnológica Federal do Paraná (UTFPR)
Av. Alberto Carazzai, 1.640 – 86.300-000 – Cornélio Procópio – PR – Brasil

uzai@alunos.utfpr.edu.br, kashiwabara@utfpr.edu.br

Abstract. *Time series are sequence of values distributed over time. Analyzing time series is important in many areas including medical, financial, aerospace, commercial and entertainment. Change Point Detection is the problem of identifying changes in meaning or distribution of data in a time series. This article presents Spec, a new algorithm that uses the graph spectrum to detect change points. The Spec was evaluated using the UCR Archive which is a large database of different time series. Spec performance was compared to the PELT, ECP, EDM, and gSeg algorithms. The results showed that Spec achieved a better accuracy compared to the state of the art in some specific scenarios and as efficient as in most cases evaluated.*

Resumo. *Séries temporais são sequência de valores distribuídos ao longo do tempo. Analisar séries temporais é importante em várias áreas, incluindo médica, financeira, aeroespacial, comercial e entretenimento. Detecção de Pontos de Mudança é o problema de identificar mudanças no significado ou na distribuição de dados em uma série temporal. Este artigo apresenta o Spec, um novo algoritmo que utiliza o espectro de grafo para detectar pontos de mudança. O Spec foi avaliado utilizando o UCR Archive que é uma grande base de dados de diferentes séries temporais. A performance do Spec foi comparado com os algoritmos PELT, ECP, EDM, e gSeg. Os resultados mostraram que o Spec alcançou uma exatidão melhor em comparação ao estado da arte em alguns cenários específicos e tão eficiente quanto na maioria dos casos avaliados.*

1. Introdução

A análise de séries temporais é importante em muitas áreas, incluindo médica, financeira, aeroespacial, empresarial e meteorológica [Aminikhanghahi and Cook 2016]. Séries temporais são um conjunto de valores dispersos ao longo do tempo, que descrevem um comportamento particular de um sistema dinâmico. Em pontos de tempo arbitrários uma série temporal pode alterar seu comportamento em relação ao restante da série já observada, esses pontos são chamados de *Change Points* (Pontos de Mudança).

Change Point Detection (Detecção de Ponto de Mudança/CPD) é o problema em identificar a mudança na distribuição ou significado dos dados em uma série temporal [Basseville and Nikiforov 1993] [Aminikhanghahi and Cook 2016]. Com o passar dos anos, diversas técnicas para detecção de pontos de mudança foram desenvolvidas [Page 1955], entretanto, com o surgimento de diferentes maneiras de capturar uma grande

quantidade de dados o interesse por esse problema vem aumentando consideravelmente [Basseville and Nikiforov 1993, Li et al. 2015].

Um desafio na detecção de pontos de mudança é no cenário de *Big Data*, visto que, frequentemente os dados possuem anomalias e a maioria dos algoritmos atuais não são resilientes a anomalias [James et al. 2016]. Anomalias são observações fora do padrão dos dados, como rúidos ou valores irregulares a distribuição da série que dificultam a identificação de padrões estatísticos da série temporal [James et al. 2016]. Outro problema é que frequentemente os dados não possuem uma distribuição normal, o que implica na limitação da aplicação de diversos algoritmos que mostram eficiência em contextos paramétricos [Killick et al. 2012].

Existem várias abordagens para detecção de pontos de mudança de categorias distintas. Cada método possui estratégias diferentes para contornar os problemas descritos anteriormente e detectar pontos de mudança com maior precisão [Aminikhanghahi and Cook 2016]. A seleção de um método deve considerar primordialmente a natureza dos dados analisados. Algoritmos produzidos na última década tem como objetivo principal a detecção de múltiplos pontos de mudança em grandes conjuntos de dados em baixo tempo [Killick et al. 2012] [Matteson and James 2014] [James et al. 2016]. Entretanto, conjuntos de dados de alta dimensionalidade são problemáticos para a maioria dos algoritmos atuais, já que, a medida que aumenta a dimensionalidade a precisão dos algoritmos diminui e o tempo para realizar o cálculo aumenta consideravelmente [Chen and Zhang 2015].

A detecção de pontos de mudança utilizando grafos foi proposto por Chen [Chen and Zhang 2015] com o propósito de detectar assertivamente pontos de mudança em bases de dados de alta dimensionalidade sem grande perda de performance. O método *gSeg* proposto por Chen apresentou a mesma assertividade que o estado da arte em bases de baixa dimensionalidade, contudo, obteve melhor assertividade que o estado da arte em bases de média para alta dimensionalidade. O método de Chen não explora a topologia dos grafos gerados a partir das séries temporais, este trabalho apresenta a hipótese de que é possível utilizar teorias fundamentais dos grafos para recolher maiores informações da série temporal e conseguir detectar pontos de mudança de forma mais assertiva sendo mais resiliente a anomalias apresentadas na série temporal.

Este trabalho faz uso da teoria espectral do grafo com a finalidade de obter o conhecimento da topologia dos grafos gerados a partir das séries temporais multivaloradas. A teoria espectral do grafo é descrita pelo estudo dos autovalores e autovetores de matrizes associadas a grafos [Spielman 2007]. O método criado nesse trabalho *Spec* (abreaviação de Graph Spectrum) utiliza o espectro do grafo a fim de separar os nós de forma que o ponto limite das separações seja o ponto de mudança. O uso do espectro do grafo para agrupamento e separação de dados se mostrou mais eficiente que os métodos tradicionais em diversos cenários [Ng et al. 2002], visto que, métodos tradicionais como o K-means tendem a falhar quando os grupos não correspondem a regiões convexas. Apesar do uso do espectro do grafo para agrupamento ser comum, no melhor do nosso conhecimento, não existem trabalhos descrevendo o uso do espectro dos grafos para detecção de pontos de mudança.

A eficiência do método proposto foi comparada com o estado da arte na detec-

ção de pontos de mudança, PELT, ECP, EDM e gSeg o qual obteve maior precisão em diferentes cenários, como verificado na base de dados *Meat* onde o resultado máximo atingido pelos outros algoritmso é de 0.64 e o *Spec* obteve resultado médio de 0.80. Além disso, atingiu a mesma média de precisão (0.60) quando comparado aos outros métodos. Este trabalho demonstra que o espectro do grafo é uma alternativa válida para detecção de múltiplos pontos de mudança.

2. Trabalhos Relacionados

Detecção de pontos de mudança tem diversas aplicações, conforme descrito abaixo:

Detecção e diagnóstico de falhas: Detectar e diagnosticar falhas com antecedência pode ser de grande vantagem principalmente quando considerado sistemas dinâmicos, pois, pode gerar ganho no aumento de confiabilidade do equipamento, assim como, redução de risco de paradas não programadas de produção reduzindo perdas materiais e acidentes de trabalho [Moreira 2011].

Controle de qualidade: uma das primeiras aplicações envolvendo pontos de mudança é no controle de qualidade e monitoramento da produção [Basseville and Nikiforov 1993]. É possível exemplificar um uso hipotético de uma fábrica de bolachas e uma câmera que analisa a cor das bolachas recém assadas em uma esteira. Naturalmente, algumas bolachas estarão mais escuras e outras mais claras, porém, caso a média das bolachas esteja mais escura que o padrão anteriormente registrado (ponto de mudança encontrado) é possível que o forno precise diminuir a temperatura. O inverso também é verdadeiro caso a média das bolachas esteja mais clara o forno possivelmente precisará aumentar a temperatura. O exemplo hipotético mencionado acima pode ser generalizado para uma série de cenários, por muitas razões é compreensível que atributos de determinado produto estejam dentro de padrões com desvios mínimos para a garantia da qualidade [Lai 2014].

Previsão climática: Nas últimas décadas cada vez mais informação climática vem sendo armazenada tornando-se necessária técnicas para minerar e entender os dados [Itoh and Kurths 2010]. Assim, pontos de mudança são utilizados para conhecer as alterações de padrões em séries temporais de informação climática e, categorizar motivos pelos quais essas mudanças ocorrem [Yonetanil and Mccabe 1994] [Itoh and Kurths 2010]. Monitorar dados climáticos é de grande interesse acadêmico já que, uma grande diversidade de eventos climáticos possuem relação com as mudanças de temperatura constatadas nas últimas décadas [Bonsal et al. 2001]. Nesse sentido, diversas técnicas de detecção de pontos mudanças são aplicadas para identificar oscilações de padrões climáticos, inclusive, são comparados entre si com a finalidade de indicar o melhor algoritmo para cada cenário. [Reeves et al. 2007] [Itoh and Kurths 2010] [Ducré-Robitaille et al. 2003].

Monitoramento médico: Para garantir a saúde de um paciente frequentemente é necessário o monitoramento de uma série de sinais vitais que indícam a evolução de um quadro clínico. Os sinais vitais são variáveis fisiológicas como: frequência cardíaca, eletrocardiograma, eletroencefalograma e entre outros, podendo ser transpostos em séries temporais [Aminikhanghahi and Cook 2016]. O monitoramento médico necessita de métodos que fornecem suporte rápido ao detectar mudanças sendo aplicado em casos críticos como medição cardíaca de crianças anestesiadas [Yang et al. 2006]. E também, em análise mais longas e detalhadas, por exemplo, ao verificar a variação cardíaca durante o sono

é necessárias horas de coletas de dados para que seja possível obter resultados válidos [Malladi et al. 2013] e entre outras aplicações [Staudacher et al. 2005] [Bosc et al. 2003].

2.1. Métodos comparados

Nessa subseção serão descritos os métodos que foram comparados com o novo método proposto sendo detalhado sua natureza, contexto de aplicação e vulnerabilidades. Todos os algoritmos apresentados são opensource e podem ser adquiridos no repositório CRAN ou GitHub, a linguagem utilizada em todos os algoritmos é primordialmente R (alguns métodos internos foram implementados em C++ para ganhar performance). O fato desses pacotes ainda estarem em evolução levanta o risco dos métodos sejam alterados após a data de publicação deste trabalho e consequentemente, o tempo computacional ou vulnerabilidades variem.

O algoritmo **PELT**, abreviação de Pruned Exact Linear Time, da categoria de razão de verossimilhança e de complexidade $\mathcal{O}(n^2)$ no entanto, em caso de uma penalidade linear f (k) = k, a complexidade do algoritmo também será linear $\mathcal{O}(n)$. Proposto por [Killick et al. 2012] como uma solução para o problema de identificar múltiplos pontos de mudança em grandes conjuntos de dados offline, porém, pode ser facilmente usado no contexto de detecção de pontos de mudança online graças à sua alta precisão, como demonstrado em [James et al. 2016].

O algoritmo E-Divisive **ECP** foi projetado para detectar pontos de mudança online [Matteson and James 2014] sendo não-paramétrico e aplicável a quase todas as séries temporais. O *ECP*, possui a finalidade de detectar múltiplos pontos de mudança sem a necessidade de informar a total quantidade ou aproximação de pontos na série temporal, bem como, a possibilidade de trabalhar com dados de múltiplos valores o que geralmente é problemático para a maioria dos algoritmos.

E-Divisive utiliza uma nova metodologia baseada em *U-statistics cluster based* (baseado em agrupamento), o algoritmo gera uma matriz contendo a distância entre todas as observações e divide a série temporal em dois grupos cada um contendo a maior distância entre eles, este processo se repete até que todas as séries temporais tenham sido percorridas, logo, a complexidade do método é $\mathcal{O}(n^2)$.

O algoritmo **EDM**, abreviatura de E-Divisive with Medians da categoria *cluster based*, propôs uma melhoria no algoritmo E-Divisive pela técnica estatística de *Moving Median* [James et al. 2016]. O algoritmo foi desenvolvido pela equipe do Twitter para ser aplicado em suas bases em produção na nuvem, pois, o número alto de anomalias no cenário é frequente (cloud big data), portanto, trata-se de um cenário de detecção online, onde, o objetivo é identificar o ponto de mudança real o mais cedo possível, algoritmos desenvolvidos até então não possuíam assertividade suficiente para serem aplicados nesse contexto.

O algoritmo **gSeg**, abreviação de Graph Segmentation, da categoria *Graph Based* (Baseado em Grafos), propõe a estimativa de localização de pontos de mudança utilizando estatísticas de varredura em uma sequência de dados. A varredura faz uso de grafos que representam a verossimilhança entre cada observação da série [Chen and Zhang 2015].

3. Fundamentação Teórica

O cálculo para obtenção dos valores do espectro de um grafo se da pela matriz de adjacência. Em um grafo não direcionado, simples e sem pesos nas arestas é representado por uma matriz de zeros e uns que se constrói naturalmente a partir das relações de adjacência entre os vértices do grafo. Nomeando uma matriz adjacente de um grafo qualquer como A, consideramos a_{ij} a relação entre os vertices, sendo 1 caso v_i e v_j sejam adjacentes e 0 caso contrário [de Abreu et al. 2014]. A matriz de adjacência é uma forma de representação do grafo. O espectro do grafo são os conjuntos dos autovalores da matriz de adjacência [de Abreu et al. 2014] [Fritscher 2011].

Com a investigação das propriedades espectrais dos gráficos, foram introduzidos métodos estatísticos para a seleção do modelo, estimativa de parâmetros e testes de hipóteses para discriminar amostras, a identificação dos conjuntos de correlação e o fluxo de informação entre gráficos de dados genéticos [Vidal et al. 2017] e neuroimagens [Fujita et al. 2017] .

3.1. Agrupamento Espectral

O objetivo do agrupamento espectral é agrupar dados que estão conectados, mas, não necessariamente agrupados em limites convexos. Para realizar o agrupamento espectral assim como no agrupamento comum, é necessário uma medida de similaridade ou afinidade $s(x, y)$ para determinar o quão próximo os pontos x e y são um do outro [Luxburg 2007].

Denotando a matriz de similaridade de um grafo qualquer como S, sendo que $S_{ij} = s(x_i, x_j)$ dada a semelhança entre as observações x_i e x_j. Usando uma medida de similaridade dos pontos como a Gaussiana, que, quanto mais próximo os valores mais tendem a 1 e quanto mais distantes mais tendem a 0, computando a matriz de similaridade S e, gerando uma matriz baseado na afinidade de S denominada A, sendo A positiva e simétrica [Neto 2013].

Possuindo a matriz de afinidade, o agrupamento é substituído por um problema de partição de grafo, onde componentes conectados do grafo são interpretados como *cluster* (grupos). O grafo deve ser particionado de tal forma que as arestas que conectam diferentes grupos devem ter pesos baixos e as bordas dentro do mesmo grupo deve ter valores altos.

A próxima etapa é computar o grafo laplaciano, que é o resultado da subtração de uma matriz de grau onde cada valor diagonal é o grau do vértice respectivo e todas as outras posições são zero da matriz de afinidade anteriormente calculada, existem variantes da matriz laplaciana que podem ser aplicadas nesse caso [Luxburg 2007].

Assumindo que pretende-se identificar n grupos, a próximo etapa é encontrar os n menores autovetores (ignorando o autovetor constante trivial). O espectro do autovalor possui uma lacuna que fornece o valor de n. Em [Neto 2013] é possível verificar o exemplo do agrupamento da série que dificilmente seria agrupada pelos métodos tradicionais (k-Means e derivados).

O espectro do grafo é frequentemente utilizado para agrupamento em várias áreas e tem obtido resultados promissores na segmentação de imagens [Casaca 2014]. Entretando, no melhor do nosso conhecimento não há trabalhos que relatem o agrupamento

espectral para detecção de pontos de mudança. O uso de grafos no contexto de pontos de mudança é recente e tem sido pouco explorado [Chen and Zhang 2015].

4. Metodologia

Nesta seção, será apresentada a metodologia para Detecção de Ponto de Mudança com o Espectro do Grafo. A Figura 1 faz um resumo da metodologia utilizada para detectar pontos de mudança utilizando o espectro do grafo.

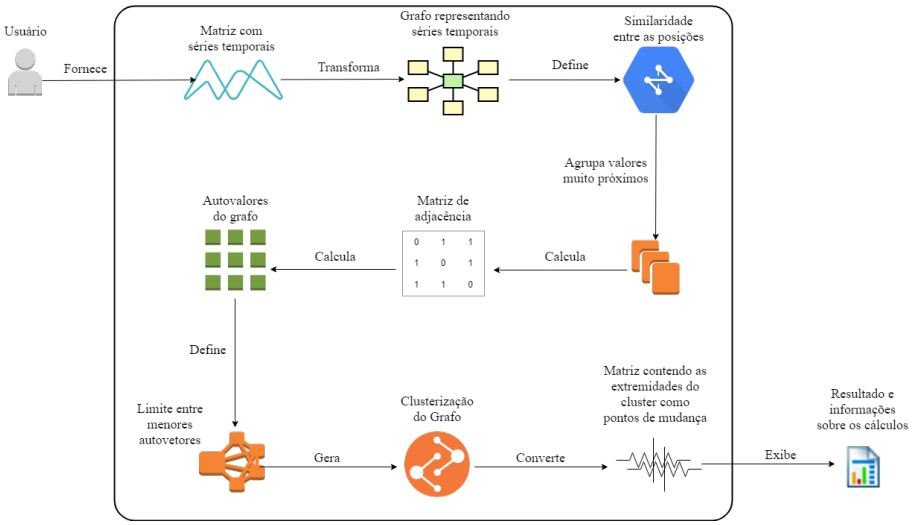


Figura 1. Explicação gráfica do processo de detecção de pontos de mudança com o espectro do grafo

O algoritmo proposto é não-paramétrico e aplicável a quase todas as séries temporais, uma vez que, a medida de similaridade relatada no espaço amostral pode ser definida. É necessário fornecer ao algoritmo quais estatísticas serão utilizadas na varredura baseada em grafos, bem como medidas de similaridade e alternativas de intervalo.

4.1. Transformação dos dados

O primeiro passo do método proposto é focado em transformar séries temporais multivaloradas em um grafo. Inicialmente, cada observação de cada medida representa um nó no grafo e o valor entre as arestas representa a distância entre cada observação. Existem várias distâncias métricas que podem ser utilizadas. Neste trabalho foi utilizada a distância euclidiana tradicional que é consolidada e trivial [Gower 1982].

Após a conversão, uma matriz de similaridade é formada sobre todas as observações do grafo. O processo é computacionalmente pesado e uma grande quantidade de dados precisa ser armazenada em memória em caso de múltiplas medidas ou em séries muito longas. A alternativa adotada para diminuir esse entrave de alto consumo de memória e tempo, bem como reduzir o ruído entre os valores é agrupar os nós muito próximos, assim, apenas os nós agrupados são armazenados em memória.

Sendo assim, se um nó A estiver relativamente próximo de um nó B, ambos os nós poderão ser substituídos por um único nó C. A distância mínima pode ser parametrizada ou calculada com base na distância média entre todos os nós da mesma série temporal. Este processo pode ser repetido até que permaneçam apenas distâncias relevantes entre os nós.

O processo de agrupamento de nós também pode ser considerado como uma suavização dos dados reduzindo o ruído. No entanto, a estrutura da base deve ser bem conhecida para que informações importantes não sejam perdidas. Para os testes realizados nesse trabalho foram agrupados os nós onde a distância é menor que 0,01 da distância máxima entre nós na mesma série e também, é menor que 0,02 da distância média na mesma série.

O agrupamento de nós leva em consideração o armazenamento de marcações que representam as observações originais, de modo que, a transformação do gráfico em séries temporais pode ser feita diretamente por referência após a detecção dos pontos de mudança.

4.2. Detecção de Pontos de Mudança

Usando o agrupamento espectral como descrito na sessão 3, as observações da série temporal são divididas em grupos distintos, onde $1 < n_0 \leq \tau \leq n_1 < n$, sendo n_0 e n_1 dois grupos distintos e τ o valor limite entre os grupos, portanto, τ é o ponto de mudança. Essa definição pode ser extrapolada para vários grupos na série temporal até que todos os pontos de mudança sejam determinados.

Tendo determinado a posição inicial dos pontos de mudança e grupos da série temporal é plausível verificar se os grupos realmente representam uma alteração no significado e na variância, ou seja, executar um teste de verificação afim de identificar falsos positivos. Várias técnicas estatísticas podem ser aplicadas para averiguar se dois grafos distintos possuem diferença de variação e significância, para este trabalho, foram utilizados os mesmos testes estatisticos de varredura aplicados no *gSeg*.

Com base nos resultados dos testes estatísticos os grupos podem ser reagrupados e os pontos de mudança desconsiderados. A tolerância dos testes deve ser definida levando em conta a natureza da distribuição dos dados bem como, o conhecimento prévio do domínio de aplicação, já que, uma tolerância muito alta podem aumentar a incidência de falsos positivos e uma tolerância baixa pode causar perda de pontos de mudança reais.

O Algoritmo 1 demonstra a detecção de pontos de mudança utilizando o Spec, ilustrando tanto a transformação de dados como o processo de detecção. O Algoritmo 1 é executado em *loop* para todas as séries temporais existentes no conjunto.

5. Resultados Experimentais

Experimentos foram implementados em R 3.4.3, sendo utilizados apenas os pacotes dos métodos para comparação com o método proposto, sem nenhum pacote de suporte.

5.1. Bases de Dados

Todas as bases de dados foram utilizadas a partir do repositório Arquivo de Classificação de Séries Temporais UCR [Chen et al. 2015]. A seleção do repositório foi dada pelos

Algoritmo 1 Detecção de Mudanças com Spec

Entrada: S = série temporal multivalorada, L = Limite para Agrupamento
Saída: Conjunto de posições representando os pontos de mudança
início
  (*Etapa 1: Transformação de dados*)
  para *cada* $s_i \in S$ **faça**
    O = MarcaValoresOriginais(s_i)
    D = CalculaDistanciaEntrePontos("Euclidiana", S, s_i)
  fim
  SN = AgrupaValoresProximos(s_i, S, D, L, O)
  MS = CalculaMatrizSimilaridade(D, SN)
  (*Etapa 2: Detecção por Agrupamento Espectral *)
  MA = CalculaMatrizAfinidade("Gaussiana", SN, MS)
  A = CalculaMatrizAdjacencia(SN, MA)
  AV = CalculaAutoValores(A)
  G = AgrupaConjuntoDados("KMeans", AV, G)
  para *cada* $g_i \in G$ **faça**
    GN = TestaValidadeGrupo(G_1, G)
  fim
  para *cada* $gn_i \in GN$ **faça**
    P = PosicaoPontoMudanca(O, SN, gn_i)
  fim
  retorna P
fim

critérios de adaptação dos algoritmos, pois alguns algoritmos possuem alta complexidade e tempo de processamento em séries temporais com mais de 10.000 observações o que tornaria as comparações inviáveis.

Para a seleção das séries temporais também foi considerado a segurança e integridade dos dados, desse modo, todas as séries apresentam as seguintes caracteristicas: sem valores não preenchidos, rotuladas e já utilizadas em trabalhos recentes em benchmnarks [Keogh and Kasetty 2003] [Rakthanmanon et al. 2013].

5.2. Avaliação de Desempenho

Para avaliar o desempenho dos algoritmos é necessário ter métricas objetivas que sejam aplicadas igualmente a todos os algoritmos. Portanto, é necessário utilizar bases rotuladas ou semi-rotuladas. Cada observação de uma série temporal é rotulada em uma classe Booleana, "Sim"onde representa que a observação é um ponto de mudança e "Não"se a observação é normal e natural na distribuição [Cook and Krishnan 2015].

- VP = Ponto de Mudança real predito como real;
- VN = Ponto de Mudança falso predito como falso;
- FP = Ponto de Mudança falso predito como real;
- FN = Ponto de Mudança real predito como falso;

$$Acc = \frac{VP + VN}{VP + FP + FN + VN}$$

Tabela 1. Descrição das Bases de Dados

Nome	Autor	Pontos de Mudança	Observações	Series
ECG	Olszewski	6	300	60
Yoga	Xi	2	3000	426
BeetleFly	Tony Bagnall	2	20	512
Computers	Tony Bagnall	2	250	720
Strawberry	Tony Bagnall	2	613	235
RefrigDevices	Tony Bagnall	3	375	720
OliveOil	Tony Bagnall	4	30	570
LgKitchenApp	Tony Bagnall	3	375	720
Meat	Tony Bagnall	3	60	448
BirdChicken	Tony Bagnall	2	20	512

$$Sp = \frac{VN}{VN + FP}$$

$$Re = \frac{VP}{VP + FN}$$

$$Pc = \frac{VP}{VP + FP}$$

$$F_1 = \frac{2 * Pc * Re}{Pc + Re}$$

$$Media = \frac{Acc + Sp + F_1}{3}$$

O fato de pesos e tolerâncias não terem sido aplicados faz com que a média dos resultados seja menor que a encontrada no estado da arte [Cheboli 2010]. Mudanças nas configurações e parâmetros de entrada dos algoritmos também podem alterar consideravelmente os resultados, atualmente, os parâmetros e posições foram ajustados para bases de dados, porém, foi feito o esforço para deixá-los o mais próximo possível da configuração padrão. O método proposto neste trabalho é denotado por *Spec* (abreviação de Spectral Clustering).

6. Conclusão

O desenvolvimento deste trabalho combinou-se em um novo pacote R capaz de identificar pontos de mudança com alta precisão em vários cenários. O uso do Espectro do Grafo já se mostrou eficiente para segmentação [Casaca 2014] e utilizá-lo na detecção de pontos de mudança se mostrou uma alternativa válida abrindo também novas possibilidades, já que, o estudo dos grafos em análise de séries temporais ainda não é muito explorado.

Em três conjunto de séries, o *Spec* obteve um resultado significativamente melhor quando comparado aos outros algoritmos:

Tabela 2. Comparação - Média dos Resultados por Método

	PELT	ECP	EDM	gSeg	Spec
ECG	0,42	0,44	0,47	0,47	0,47
Yoga	0,46	0,43	0,44	0,46	0,46
BeetleFly	0,69	0,63	0,69	0,69	0,69
Computers	0,58	0,65	0,71	0,71	0,71
Strawberry	0,49	0,46	0,46	0,53	0,53
RefrigDevices	0,57	0,62	0,55	0,65	**0,71**
OliveOil	0,44	0,55	0,59	0,59	**0,63**
LgKitchenApp	0,53	0,60	0,65	0,65	0,65
Meat	0,64	0,60	0,64	0,64	**0,80**
BirdChicken	0,58	0,63	0,69	0,69	0,69
Média	**0,54**	**0,56**	**0,59**	**0,61**	**0,63**

- Meat;
- OliveOil;
- RefrigDevices;

Explorar as características dessas séries deve ajudar a entender quando o uso do Espectro é mais vantajoso. Mesmo considerando os conjunto de séries que o algoritmo não apresentou uma melhora na média ainda se mostrou com performance similar aos outros algoritmos, mostrando sua viabilidade.

Este trabalho demonstrou que a transformação de séries em grafo torna possível aplicar o Espectro de Grafo para resolver o problema de pontos de mudança.

Referências

Aminikhanghahi, S. and Cook, D. J. (2016). A survey of methods for time series change point detection. *Knowledge and Information Systems*, pages 1–29.

Basseville, M. and Nikiforov, I. (1993). *Detection of abrupt changes. Theory and application*, volume 2.

Bonsal, B. R., Zhang, X., Vincent, L. A., and Hogg, W. D. (2001). Characteristics of daily and extreme temperatures over Canada. *Journal of Climate*, 14(9):1959–1976.

Bosc, M., Heitz, F., Armspach, J. P., Namer, I., Gounot, D., and Rumbach, L. (2003). Automatic change detection in multimodal serial MRI: Application to multiple sclerosis lesion evolution. *NeuroImage*, 20(2):643–656.

Casaca, W. C. d. O. (2014). *Graph laplacian for spectral clustering and seeded image segmentation*. PhD thesis, Universidade de São Paulo.

Cheboli, D. (2010). Anomaly detection on time series. *2010 IEEE International Conference on Progress in Informatics and Computing*, 1:603–608.

Chen, H. and Zhang, N. (2015). Graph-based change-point detection. *Annals of Statistics*, 43(1):139–176.

Chen, Y., Keogh, E., Hu, B., Begum, N., Bagnall, A., Mueen, A., and Batista, G. (2015). The ucr time series classification archive. www.cs.ucr.edu/~eamonn/time_series_data/.

Cook, D. and Krishnan, N. (2015). *Activity Learning*, volume 1.

de Abreu, N. M. M., Del-Vecchio, R., Trevisan, V., and Vinagre, C. T. M. (2014). Teoria Espectral de Grafos - Uma Introdução. page 201.

Ducré-Robitaille, J. F., Vincent, L. A., and Boulet, G. (2003). Comparison of techniques for detection of discontinuities in temperature series. *International Journal of Climatology*, 23(9):1087–1101.

Fritscher, E. (2011). Propriedades espectrais de um grafo.

Fujita, A., Takahashi, D. Y., Balardin, J. B., Vidal, M. C., and Sato, J. R. (2017). Correlation between graphs with an application to brain network analysis. *Computational Statistics and Data Analysis*, 109:76–92.

Gower, J. C. (1982). Euclidean distance geometry. *Math. Sci*, 7(1):1–14.

Itoh, N. and Kurths, J. (2010). Change-Point Detection of Climate Time Series by Nonparametric Method. In *Proceedings of the World Congress on Engineering and Computer Science*, volume I, pages 20–23.

James, N. A., Kejariwal, A., and Matteson, D. S. (2016). In *Proceedings - 2016 IEEE International Conference on Big Data, Big Data 2016*, pages 3499–3508.

Keogh, E. and Kasetty, S. (2003). On the need for time series data mining benchmarks: a survey and empirical demonstration. *Data Mining and knowledge discovery*, 7(4):349–371.

Killick, R., Fearnhead, P., and Eckley, I. A. (2012). Optimal detection of changepoints with a linear computational cost. *Journal of the American Statistical Association*, 107(500):1590–1598.

Lai, T. L. (2014). Sequential changepoint detection in quality control and dynamical systems. *Journal of the Royal Statistical Society*, 47(3):429–437.

Li, S., Xu, L. D., and Zhao, S. (2015). The internet of things: a survey. *Information Systems Frontiers*, 17(2):243–259.

Luxburg, U. V. (2007). A Tutorial on Spectral Clustering. pages 1–32.

Malladi, R., Kalamangalam, G. P., and Aazhang, B. (2013). Online Bayesian change point detection algorithms for segmentation of epileptic activity. In *Conference Record - Asilomar Conference on Signals, Systems and Computers*, pages 1833–1837.

Matteson, D. S. and James, N. A. (2014). *Journal of the American Statistical Association*, 109(505):334–345.

Moreira, F. d. S. (2011). Detecção de pontos de mudança em séries temporais utilizando uma formulação Neural/Fuzzy/Bayesiana: Aplicaçao na detecção de falhas. *Dissertação Mestrado - UFMG*, page 69.

Neto, J. (2013). Spectral clustering. http://www.di.fc.ul.pt/~jpn/r/spectralclustering/spectralclustering.html.

Ng, A. Y., Jordan, M. I., and Weiss, Y. (2002). On spectral clustering: Analysis and an algorithm. In *Advances in neural information processing systems*, pages 849–856.

Page, E. S. (1955). A Test for a Change in a Parameter Occurring at an Unknown Point. *Biometrika*, 42(3/4):523.

Rakthanmanon, T., Campana, B., Mueen, A., Batista, G., Westover, B., Zhu, Q., Zakaria, J., and Keogh, E. (2013). Addressing big data time series: Mining trillions of time series subsequences under dynamic time warping. *ACM Transactions on Knowledge Discovery from Data (TKDD)*, 7(3):10.

Reeves, J., Chen, J., Wang, X. L., Lund, R., and Lu, Q. Q. (2007). A review and comparison of changepoint detection techniques for climate data. *Journal of Applied Meteorology and Climatology*, 46(6):900–915.

Spielman, D. A. (2007). Spectral graph theory and its applications. In *Foundations of Computer Science, 2007. FOCS'07. 48th Annual IEEE Symposium on*, pages 29–38. IEEE.

Staudacher, M., Telser, S., Amann, A., Hinterhuber, H., and Ritsch-Marte, M. (2005). A new method for change-point detection developed for on-line analysis of the heart beat variability during sleep. *Physica A: Statistical Mechanics and its Applications*, 349(3-4):582–596.

Vidal, M. C., Sato, J. R., Balardin, J. B., Takahashi, D. Y., and Fujita, A. (2017). Anocva in r: A software to compare clusters between groups and its application to the study of autism spectrum disorder. *Frontiers in neuroscience*, 11:16.

Yang, P., Dumont, G., and Ansermino, J. M. (2006). Adaptive change detection in heart rate trend monitoring in anesthetized children. *IEEE Transactions on Biomedical Engineering*, 53(11):2211–2219.

Yonetanil, T. and Mccabe, G. J. (1994). Abrupt changes in regional temperature in the conternimous United States, 1895-1989. *Climate Research*, 4:13–23.

B APÊNDICE - RESULTADO DO TESTE DE WILCOXON DETALHADO

Neste Apêndice será demonstrado o detalhamento do teste de Wilcoxon realizado e sumarizado no Capítulo 4, o teste foi realizado considerando os resultados de todas as tolerâncias e o algoritmo Spec como base.

	AMOC	BinSeg	SegNeigh	PELT	EDivisive	EDM	MFT	Breakfast	gSeg
FaceAll	1	1	1	1	0,944864555	0,000179018	1	1	1
ElectricDevices	1	1	1	1	1	1	1	1	1
ECG	0,707686276	0,967157536	0,999605603	0,999238633	0,080957454	0,964084786	0,916256721	0,849198047	0,662276802
Yoga	1	1	1	1	0,5	0,5	1	1	1
Wine	1,35464E-37	2,09895E-38	5,8893E-38	1,53621E-34	5,26021E-45	2,52193E-34	1,94347E-33	2,52193E-34	2,52193E-34
Earthquakes	6,82874E-86	1,60682E-85	1,4854E-83	1,24769E-19	6,69965E-86	6,80654E-86	6,83558E-86	6,80613E-86	6,68733E-86
Coffee	7,267E-11	0,171802659	8,2688E-07	1,18487E-06	1,23932E-54	0,969924471	0,002154356	0,969924471	0,969924471
BeetleFly	3,8977E-58	2,2245E-25	3,5632E-43	2,06889E-43	1,91314E-91	1,18944E-24	1,18944E-24	1,18944E-24	1,18944E-24
WormsTwoClass	0,994858017	1	1	1	2,2575E-186	1	1	1	1
ArrowHead	1	0,912188439	0,00069808	0,003829526	8,9381E-39	1	1	1	1
ShapesAll	6,95613E-86	7,4697E-86	1,73639E-85	1	6,96256E-86	3,40191E-78	9,22715E-86	6,9391E-86	6,9391E-86
Computers	0,000734062	6,87183E-67	3,90971E-65	8,57942E-29	4,0046E-119	7,06365E-32	1	1	1
ToeSegmentation	0,699307314	1	0,999997572	0,999997571	6,87613E-59	4,87127E-39	1	1	1
Strawberry	2,81572E-18	0,996792815	0,999482197	0,999535623	6,25115E-20	0,537017037	0,999998669	1,46131E-34	1,46131E-34
RefrigerationDevices	1	1	4,80395E-52	1,22366E-34	7,20829E-27	1,39684E-73	1	1	1
Phoneme	0,997854109	1	0,997854109	1	1	0,997854109	1	0,997854109	0,997854109
OliveOil	1	1	1	1	1	1	1	1	1
SmallKitchenAppliances	1	2,71913E-32	5,92924E-69	1	2,95828E-29	1	1	1	1
LargeKitchenAppliances	1	1,87168E-07	0,97004939	9,87963E-86	1,40058E-07	1	1	1	1
ShapeletSim	3,42453E-19	1	0,9999472	0,999917601	2,93654E-95	1,22119E-75	1	1	1
Meat	2,05946E-49	1,0511E-22	1,84421E-30	3,29618E-54	3,48291E-76	4,2118E-66	3,89589E-44	4,98868E-65	5,79934E-70
BirdChicken	7,98362E-74	2,70726E-45	1,05767E-78	1,9535E-77	6,481E-93	2,44558E-52	2,44558E-52	2,44558E-52	2,44558E-52
DistalPhalanxOutlineAgeGroup	1	1	1	1	1	1	1	1	1
ScreenType	1	3,07168E-87	3,1888E-147	7,02611E-95	2,93678E-55	0,963340349	1	1	1
synthetic_control	0,999999884	1	1	1	1,22422E-05	0,991875778	1	0,724414877	6,88681E-05

Figura 42: Teste de *Wilcoxon* com base nos resultados dos algoritmos sem tolerância

	AMOC	BinSeg	SegNeigh	PELT	EDivisive	EDM	MFT	Breakfast	gSeg
FaceAll	7,75182E-24	5,23648E-18	2,66285E-21	7,85459E-22	2,71597E-21	1,54293E-23	1,76942E-08	1,44345E-25	1,44345E-25
ElectricDevices	0,5	0,5	0,5	0,5	1	0,5	1	0,5	0,5
ECG	8,50734E-18	1,12285E-17	1,3981E-17	2,16472E-17	8,46669E-18	9,10605E-18	8,51936E-18	8,46513E-18	8,46513E-18
Yoga	0,5	0,5	0,5	0,5	0,5	1	0,5	0,5	0,5
Wine	1,37309E-43	5,41338E-41	3,86968E-41	1,54429E-40	5,26021E-45	5,26021E-45	1,35896E-40	5,26021E-45	5,26021E-45
Earthquakes	6,19321E-86	6,92187E-86	7,72565E-86	1	3,81681E-86	3,66994E-86	3,85223E-86	3,64915E-86	3,64915E-86
Coffee	1,77515E-52	4,44722E-43	3,69679E-44	2,57549E-44	1,83266E-54	1,89841E-54	2,61386E-42	1,89841E-54	1,89841E-54
BeetleFly	2,07209E-81	1,70072E-78	3,69381E-80	4,02097E-80	5,31443E-90	4,48197E-86	4,48197E-86	4,48197E-86	4,48197E-86
WormsTwoClass	4,79428E-53	0,964766366	0,005544247	0,003578291	1,1913E-159	1	0,054762077	0,995651702	0,995651702
ArrowHead	1,06281E-13	3,78996E-09	4,86021E-13	2,40164E-17	1,41422E-24	0,000963964	7,27875E-37	7,92251E-57	7,92251E-57
ShapesAll	6,95579E-86	9,33541E-86	6,95982E-86	1	1	1	0,927261026	6,85913E-86	6,85913E-86
Computers	1,0224E-113	5,0215E-117	2,3557E-120	2,1114E-118	3,4518E-116	3,5302E-115	1,2724E-123	2,3409E-137	2,3409E-137
ToeSegmentation	4,57792E-51	2,20835E-54	3,30233E-51	3,30771E-51	3,06963E-56	2,31595E-43	9,99759E-56	9,99759E-56	9,99759E-56
Strawberry	1,81174E-41	9,20018E-40	3,83959E-40	1,29003E-39	1,35747E-40	1,02284E-38	5,97411E-40	2,97064E-43	2,97064E-43
RefrigerationDevices	0,996347468	1	1	1	0,16983165	1	1	1	1
Phoneme	0,052734375	1	1	1	1	0,002864688	0,003941085	0,002864688	0,002864688
OliveOil	6,46851E-87	3,14911E-11	2,11459E-14	3,05876E-17	2,4859E-102	2,4859E-102	2,60873E-33	2,4859E-102	2,4859E-102
SmallKitchenAppliances	2,4533E-13	0,34040554	0,243406063	4,45401E-53	0,525146969	7,0072E-39	1,11604E-12	2,3584E-56	2,3584E-56
LargeKitchenAppliances	1,27163E-45	4,36837E-38	9,11639E-21	4,7529E-114	6,89968E-73	1,75003E-87	1,20825E-86	1,75003E-87	1,75003E-87
ShapeletSim	4,43232E-98	1,75656E-96	1,21524E-89	1,22021E-89	9,39668E-96	2,85499E-85	3,64317E-90	3,37368E-98	3,37368E-98
Meat	1,45325E-34	0,019900847	0,000855398	1,3466E-26	3,48291E-76	4,2118E-66	1,87585E-06	4,98868E-65	5,79934E-70
BirdChicken	5,74573E-87	6,68922E-83	2,62577E-85	2,88351E-85	2,46337E-92	5,48739E-90	5,48739E-90	5,48739E-90	5,48739E-90
DistalPhalanxOutlineAgeGroup	3,92741E-15	4,57098E-12	1,95271E-05	0,00075466	1,94736E-05	0,01326756	4,75785E-15	3,82336E-15	3,82336E-15
ScreenType	6,8366E-113	7,9979E-114	2,8562E-105	9,3684E-118	8,177E-109	6,1787E-112	7,4569E-118	1,2617E-123	1,2617E-123
synthetic_control	7,48547E-12	0,988459806	0,999991362	0,999875484	1,93652E-11	0,046013453	0,974712901	5,11245E-12	5,11245E-12

Figura 43: Teste de *Wilcoxon* com base nos resultados dos algoritmos com 0.05 de tolerância

	AMOC	BinSeg	SegNeigh	PELT	EDivisive	EDM	MFT	Breakfast	gSeg
FaceAll	1,46889E-23	1	1	1,1302E-16	5,33458E-07	1,85476E-23	1	1,44345E-25	1,44345E-25
ElectricDevices	0,5	1	1	0,5	1	0,5	1	0,5	0,5
ECG	8,50734E-18	4,28373E-17	2,0834E-08	9,28071E-08	8,46669E-18	1,54916E-15	8,51936E-18	8,46513E-18	8,46513E-18
Yoga	0,5	0,5	0,5	0,5	0,5	1	0,5	0,5	0,5
Wine	1,37309E-43	5,41338E-41	3,86968E-41	1,54429E-40	5,26021E-45	5,26021E-45	1,35896E-40	5,26021E-45	5,26021E-45
Earthquakes	6,19668E-86	2,52369E-70	0,999916878	1	5,59507E-86	3,66994E-86	3,85223E-86	3,64915E-86	3,64915E-86
Coffee	7,01233E-53	1,25497E-42	9,46212E-44	6,69877E-44	1,23932E-54	1,23932E-54	1,25373E-40	1,23932E-54	1,23932E-54
BeetleFly	8,70349E-88	5,91156E-83	4,14192E-83	4,11795E-83	6,24834E-91	1,45503E-90	1,45503E-90	1,45503E-90	1,45503E-90
WormsTwoClass	1,1533E-180	6,8573E-175	3,6555E-153	3,4871E-153	3,0356E-183	4,21115E-53	1,8162E-122	1,7808E-178	1,7808E-178
ArrowHead	5,57219E-11	0,005237872	0,002695963	5,32909E-09	0,000428256	0,908075103	1,63719E-35	1,29844E-56	1,29844E-56
ShapesAll	6,95844E-86	4,83654E-54	5,85176E-62	1	1	1	1	6,85913E-86	6,85913E-86
Computers	1,7762E-110	3,3105E-116	6,514E-122	2,8997E-119	4,242E-114	1,9513E-119	1,8528E-125	3,2112E-140	3,2112E-140
ToeSegmentation	2,86288E-55	1,8614E-58	5,26908E-53	5,2774E-53	6,87802E-59	1,06651E-47	4,15265E-60	4,15265E-60	4,15265E-60
Strawberry	5,45268E-42	1,10947E-20	1,46846E-32	5,83657E-39	1,91908E-24	2,55483E-23	3,89946E-17	3,30012E-45	3,30012E-45
RefrigerationDevices	0,996347468	1	1	1	0,996958932	1	1	1	1
Phoneme	0,998046875	1	0,997854109	1	1	0,002864688	0,341350227	0,002864688	0,002864688
OliveOil	1,6544E-93	3,37217E-37	7,98189E-36	1,57067E-49	7,6252E-113	7,6252E-113	5,73833E-63	7,6252E-113	7,6252E-113
SmallKitchenAppliances	1,9478E-05	0,602222932	0,87931645	2,69936E-86	1,24493E-08	5,32092E-78	2,03574E-40	8,82686E-91	8,82686E-91
LargeKitchenAppliances	1,03884E-71	2,36631E-47	3,04588E-20	1,5312E-114	5,4197E-89	2,0687E-111	1,7805E-106	2,0687E-111	2,0687E-111
ShapeletSim	4,43232E-98	1,75656E-96	1,22332E-89	1,22787E-89	9,40323E-96	1,3062E-87	2,82424E-89	3,37368E-98	3,37368E-98
Meat	1,4923E-06	8,42463E-39	4,58233E-46	3,03265E-69	7,5636E-82	3,3667E-86	9,09611E-62	9,8692E-100	9,8692E-100
BirdChicken	9,16688E-93	2,08753E-86	9,35857E-86	9,08099E-86	6,759E-93	6,80212E-93	6,80212E-93	6,80212E-93	6,80212E-93
DistalPhalanxOutlineAgeGroup	3,92429E-15	3,38512E-07	0,578890281	0,829478593	0,114713139	0,999947302	6,32682E-14	3,82336E-15	3,82336E-15
ScreenType	3,064E-131	1,9624E-120	2,6645E-127	7,6632E-120	3,4387E-121	1,8656E-126	6,2651E-122	4,1617E-152	4,1617E-152
synthetic_control	6,57927E-12	0,980596038	0,999970766	0,999602416	7,95318E-12	0,055205076	0,954492898	3,54176E-12	3,54176E-12

Figura 44: Teste de *Wilcoxon* com base nos resultados dos algoritmos com 0.10 de tolerância

C APÊNDICE - VISUALIZAÇÃO DOS RESULTADOS DE FORMA GRÁFICA E DETALHADA

Neste Apêndice será demonstrado o detalhamento dos testes dos algoritmos e a performance em detrimento do aumento da tolerância.

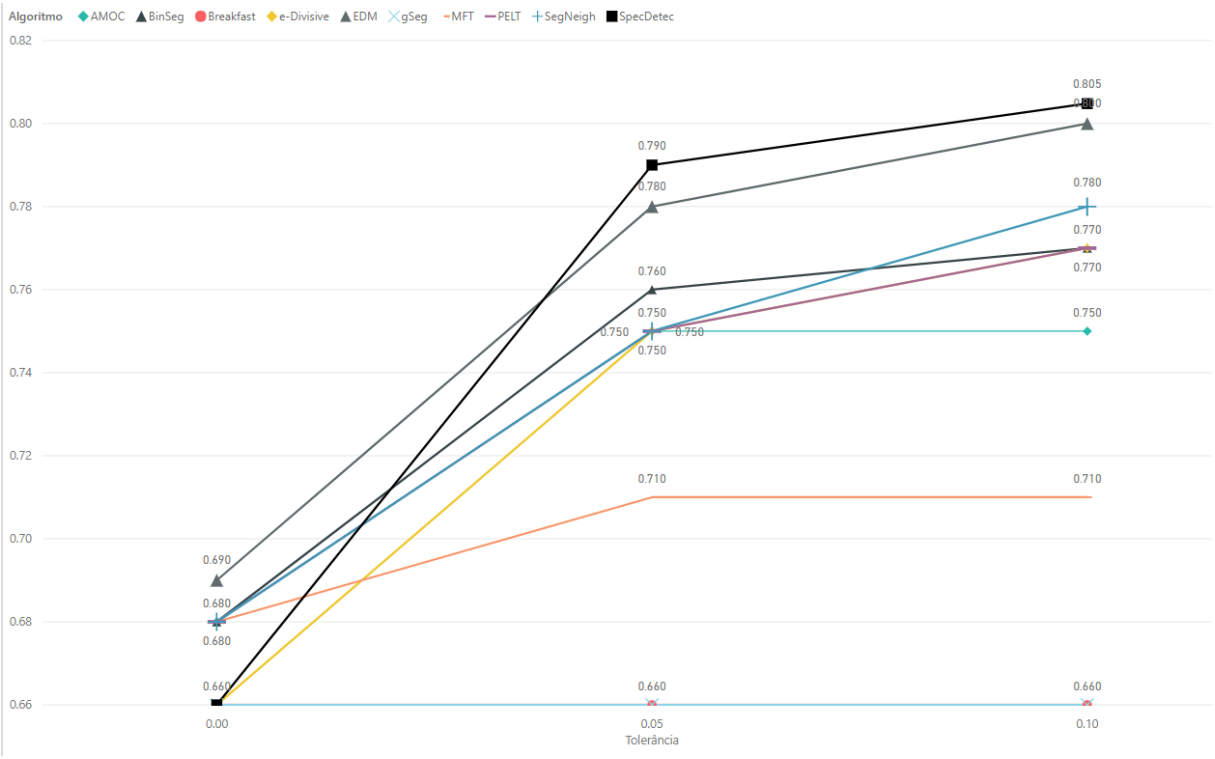


Figura 45: Comparação do Resultado do DataSet ArrowHead Por Tolerância

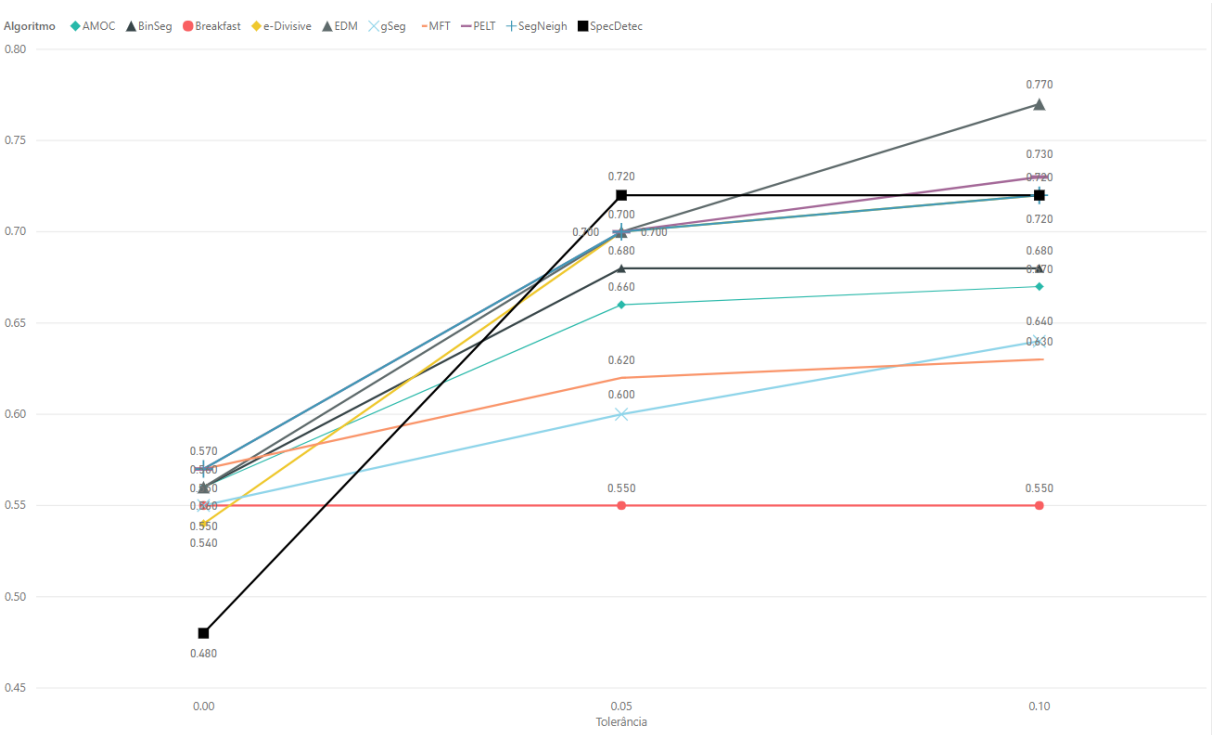


Figura 46: Comparação do Resultado do DataSet DistalPhalanxOutlineAgeGroup Por Tolerância

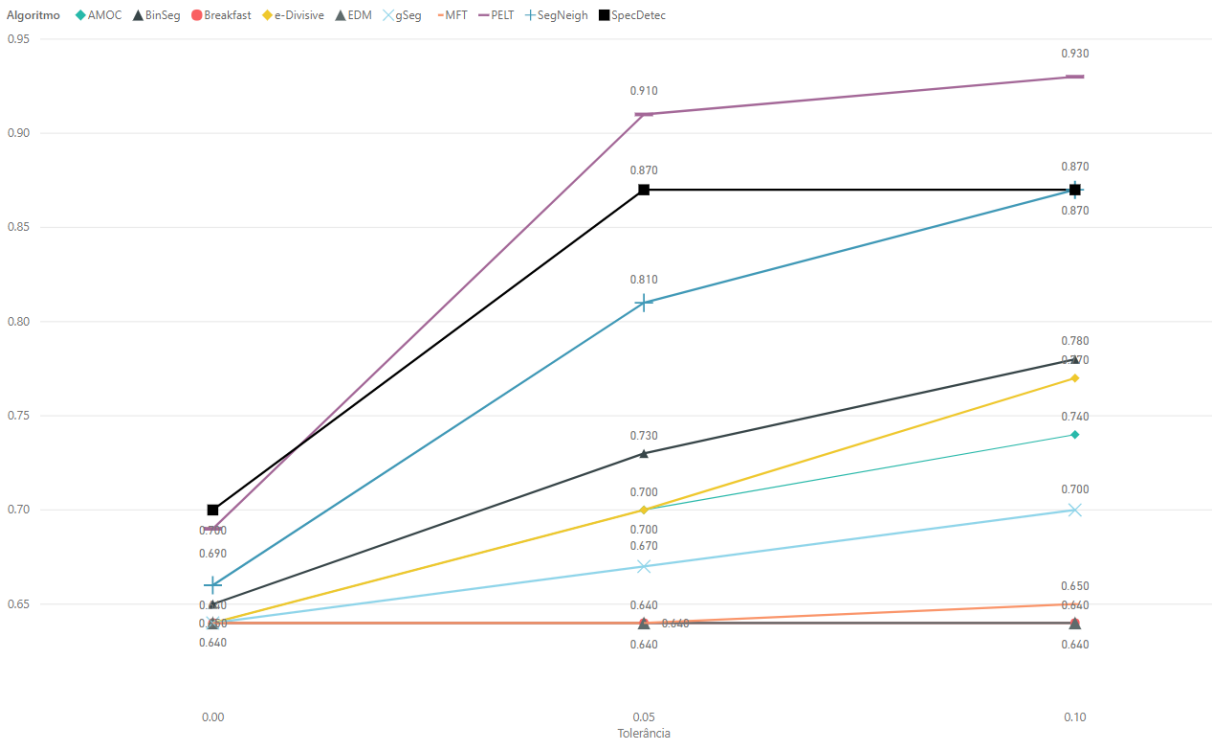


Figura 47: Comparação do Resultado do DataSet Earthquakes Por Tolerância

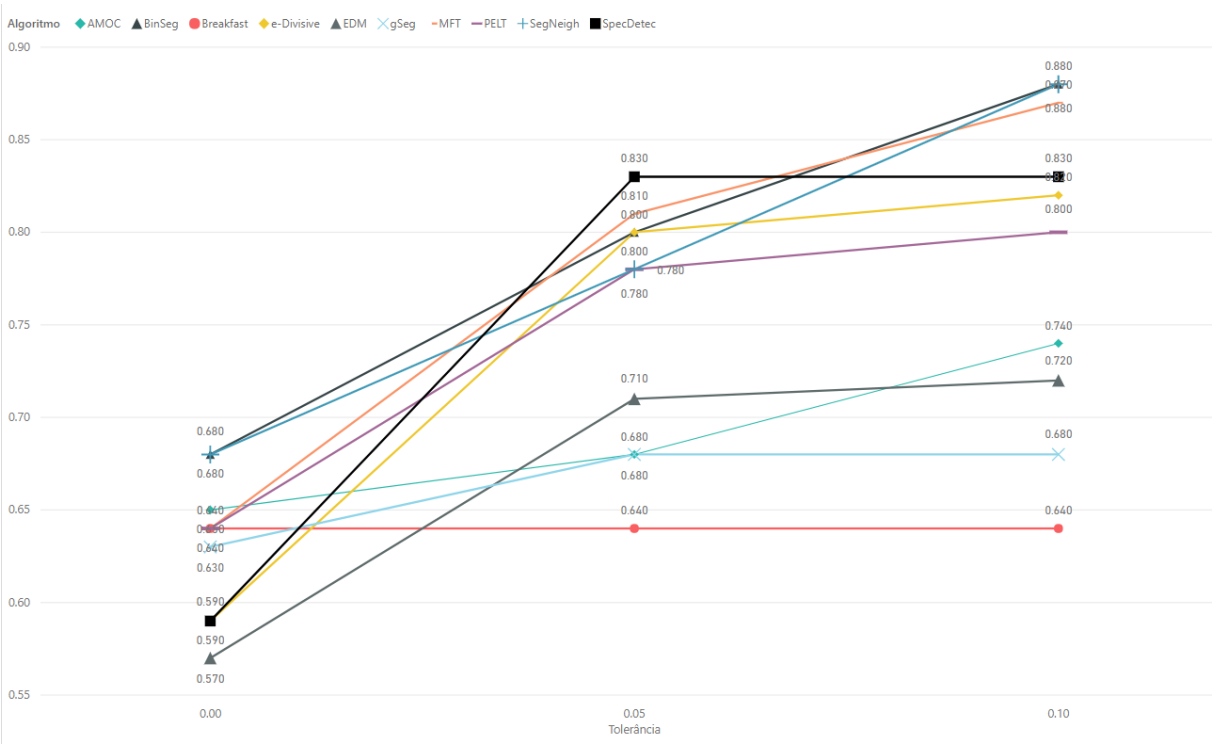


Figura 48: Comparação do Resultado do DataSet FaceAll Por Tolerância

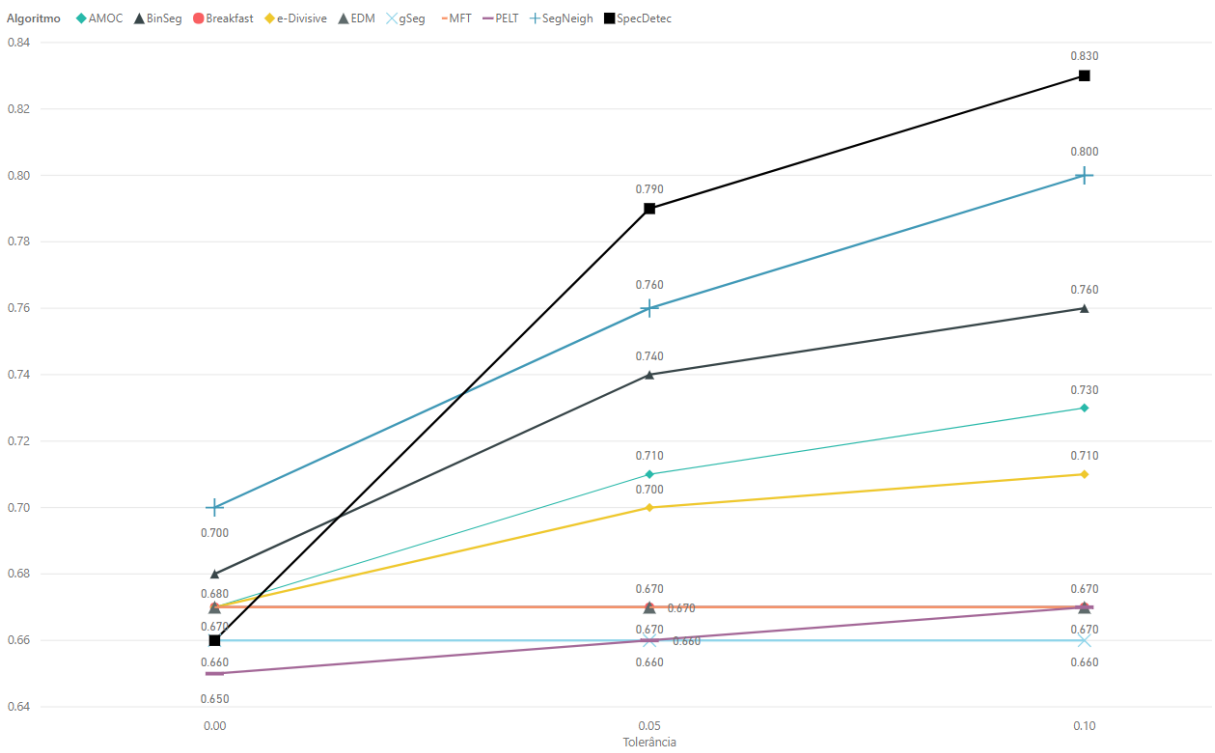


Figura 49: Comparação do Resultado do DataSet LargeKitchenAppliances Por Tolerância

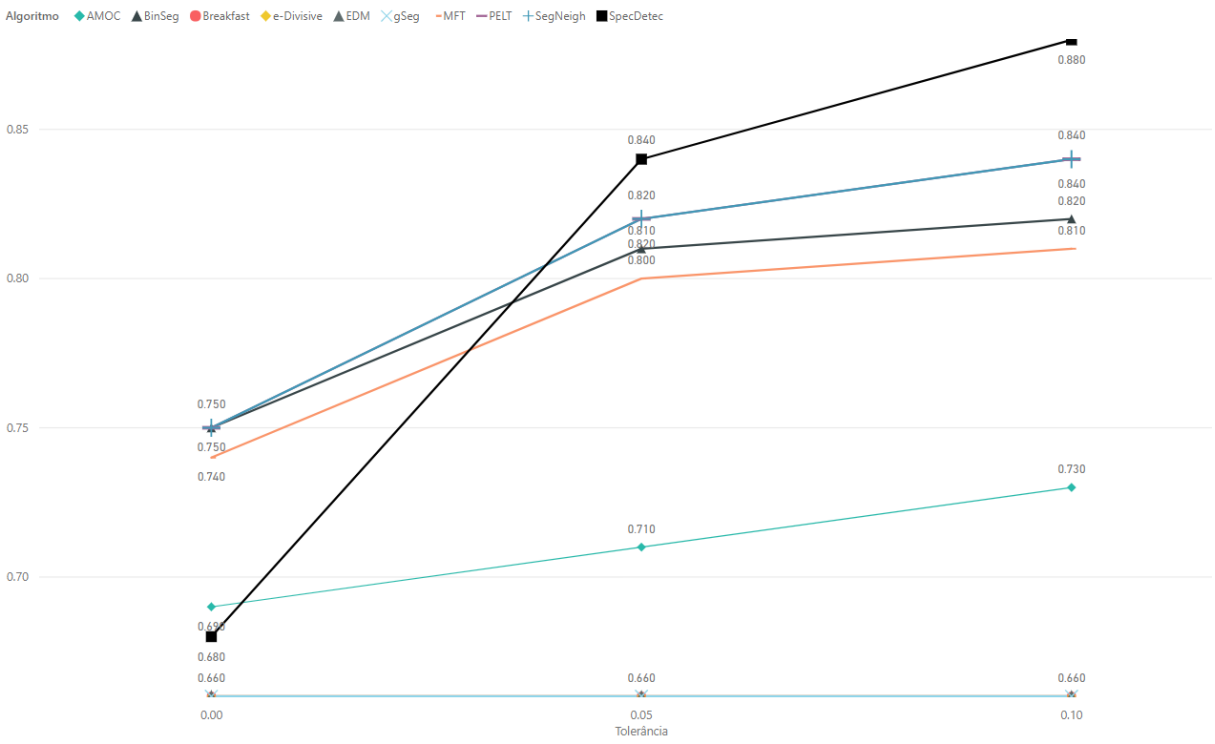


Figura 50: Comparação do Resultado do DataSet OliveOil Por Tolerância

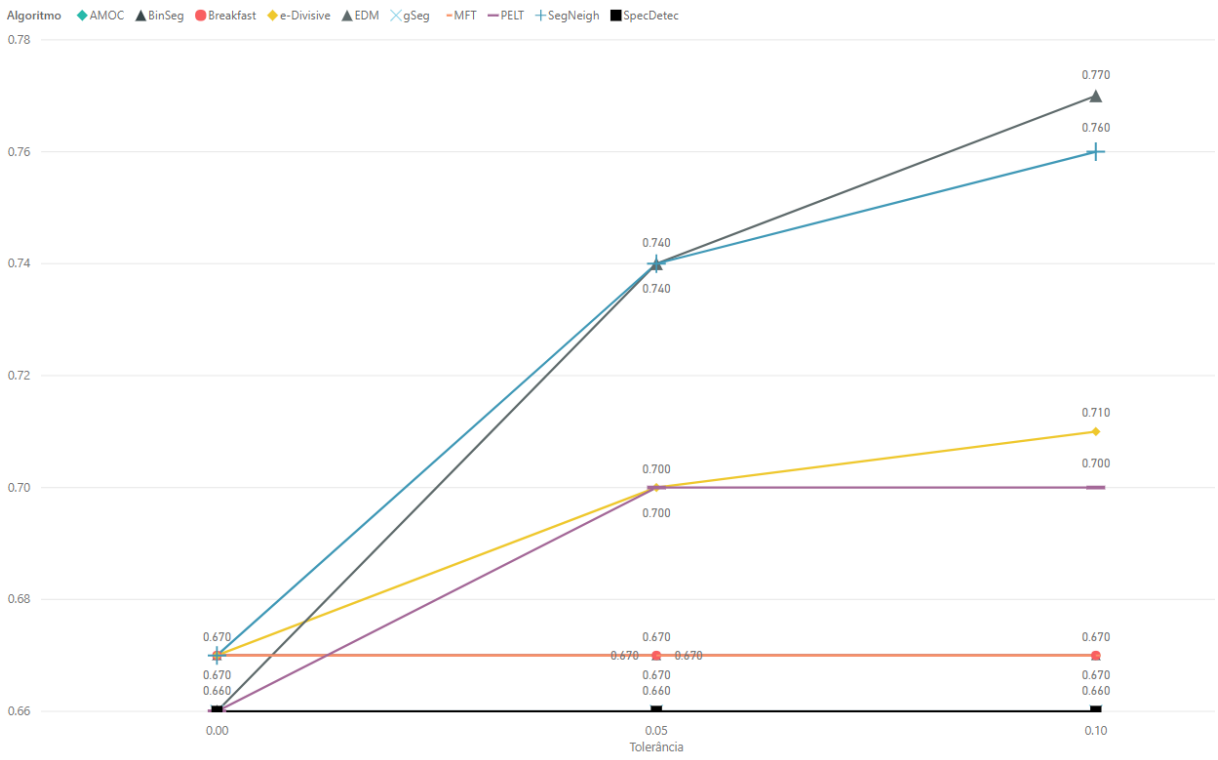


Figura 51: Comparação do Resultado do DataSet RefrigerationDevices Por Tolerância

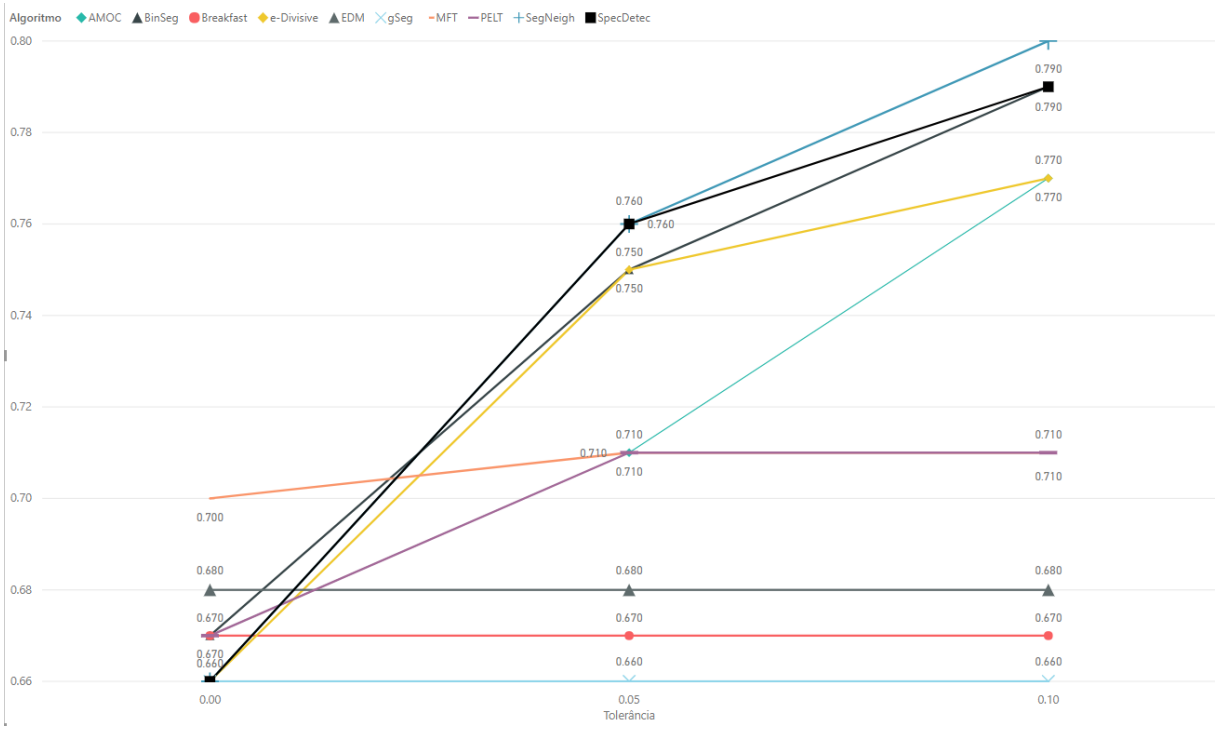


Figura 52: Comparação do Resultado do DataSet SmallKitchenAppliances Por Tolerância

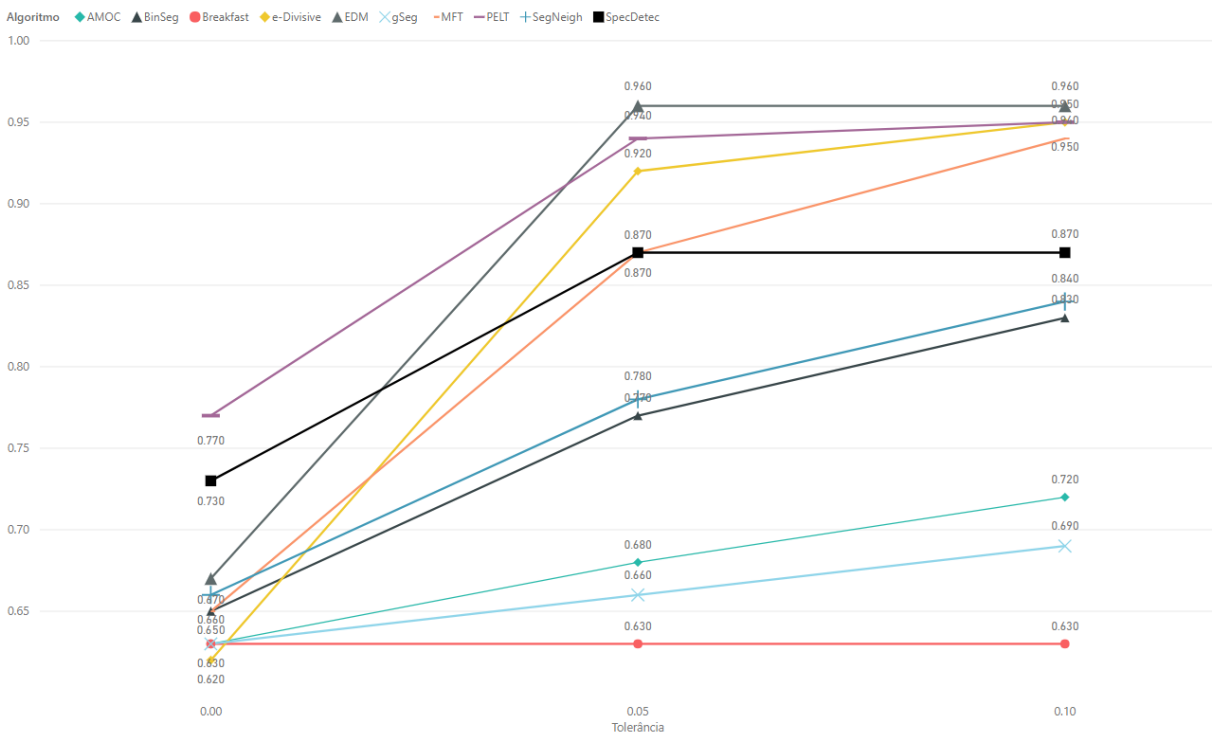


Figura 53: Comparação do Resultado do DataSet ShapesAll Por Tolerância

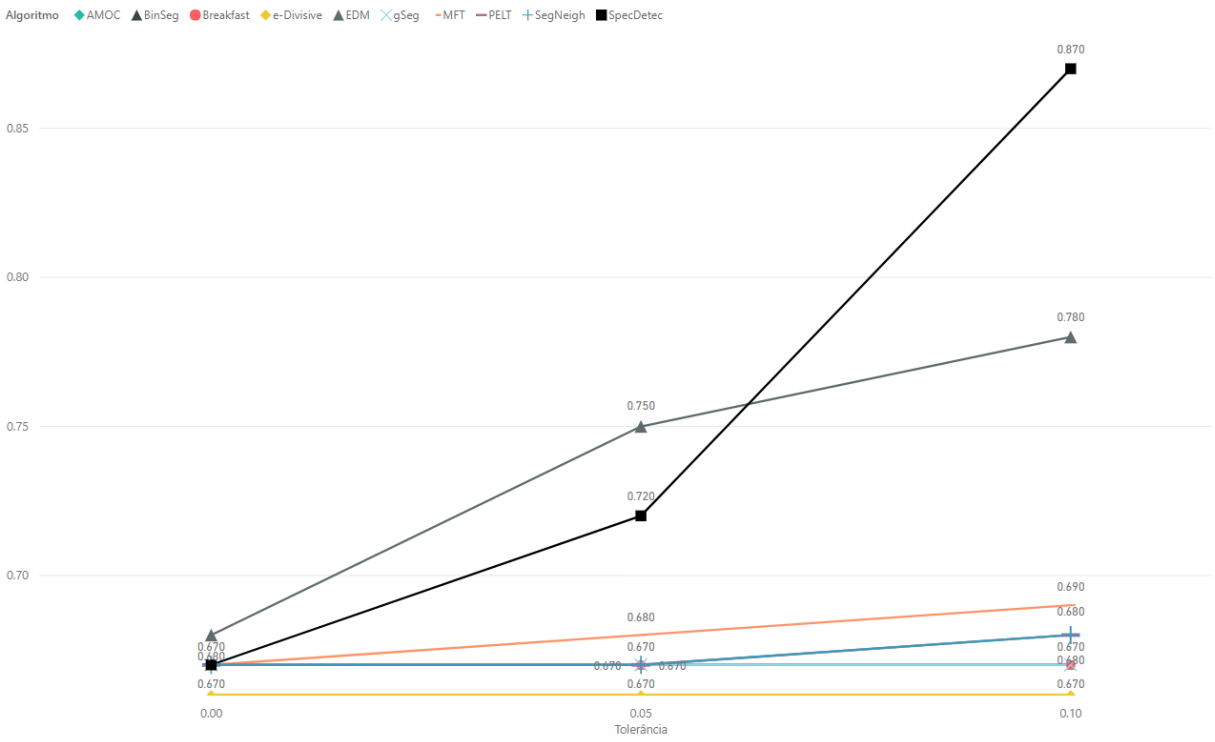


Figura 54: Comparação do Resultado do DataSet WormsTwoClass Por Tolerância

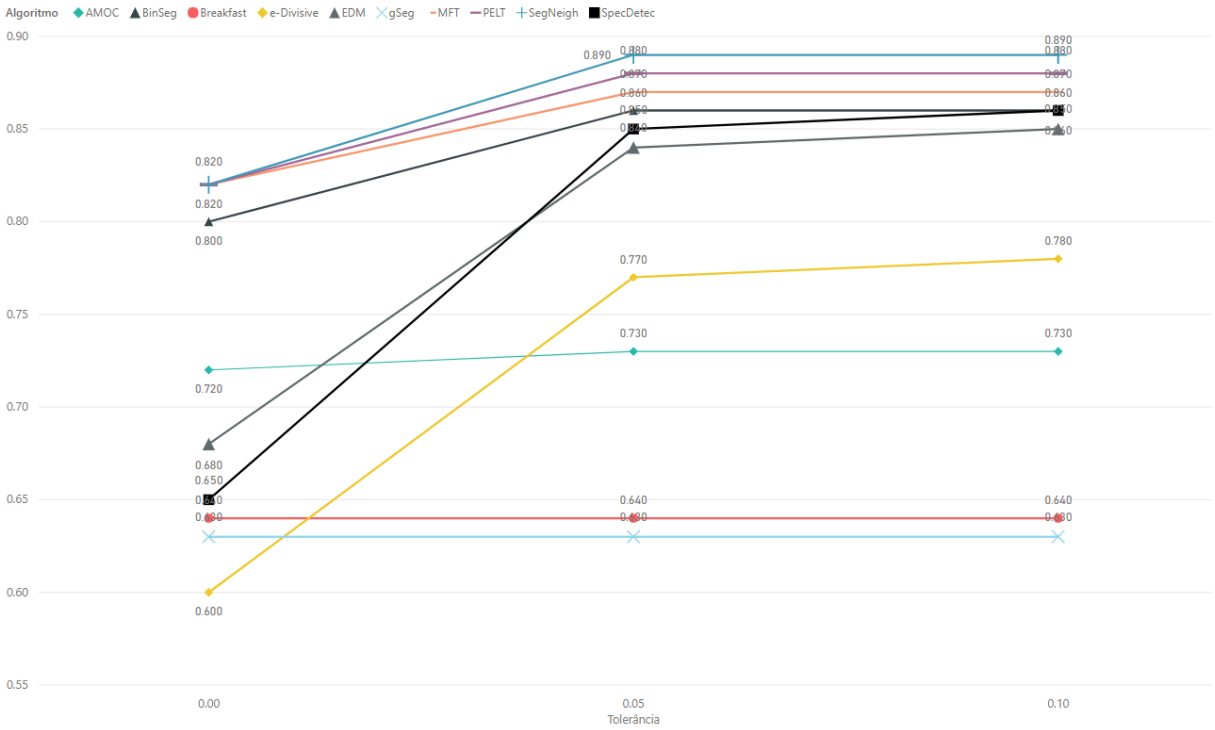


Figura 55: Comparação do Resultado do DataSet syntheticControl Por Tolerância

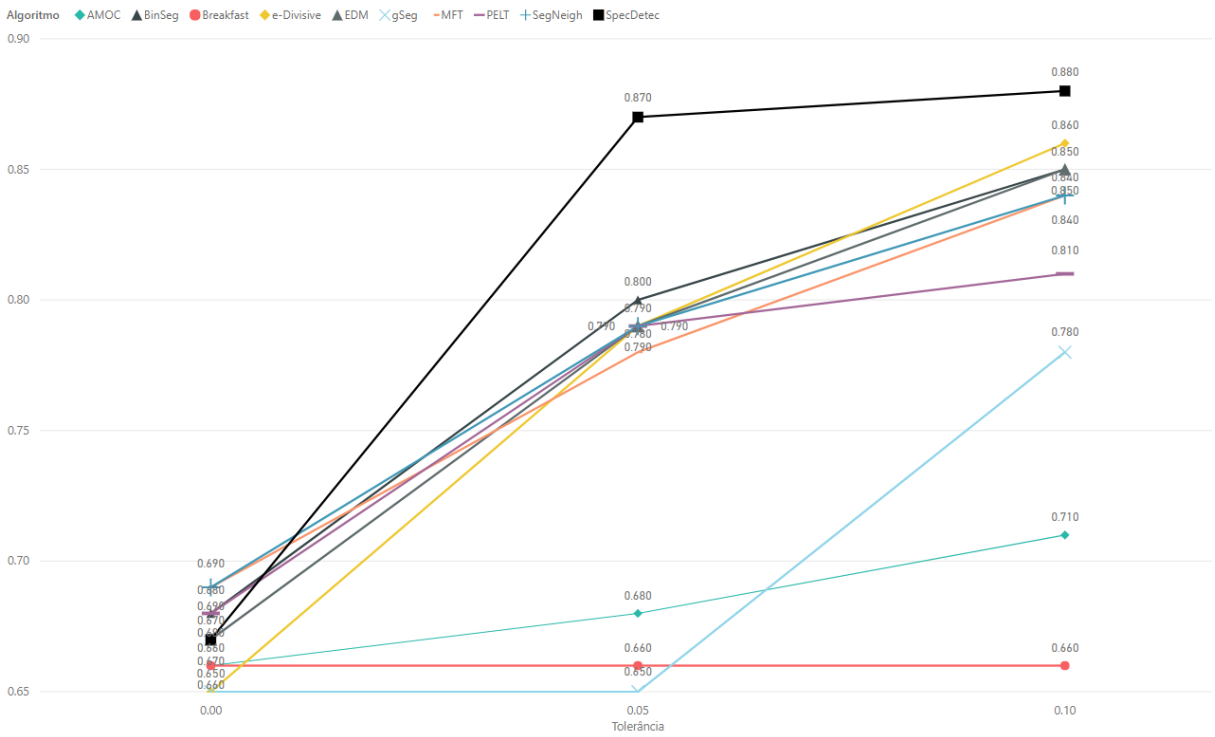


Figura 56: Comparação do Resultado do DataSet Strawberry Por Tolerância

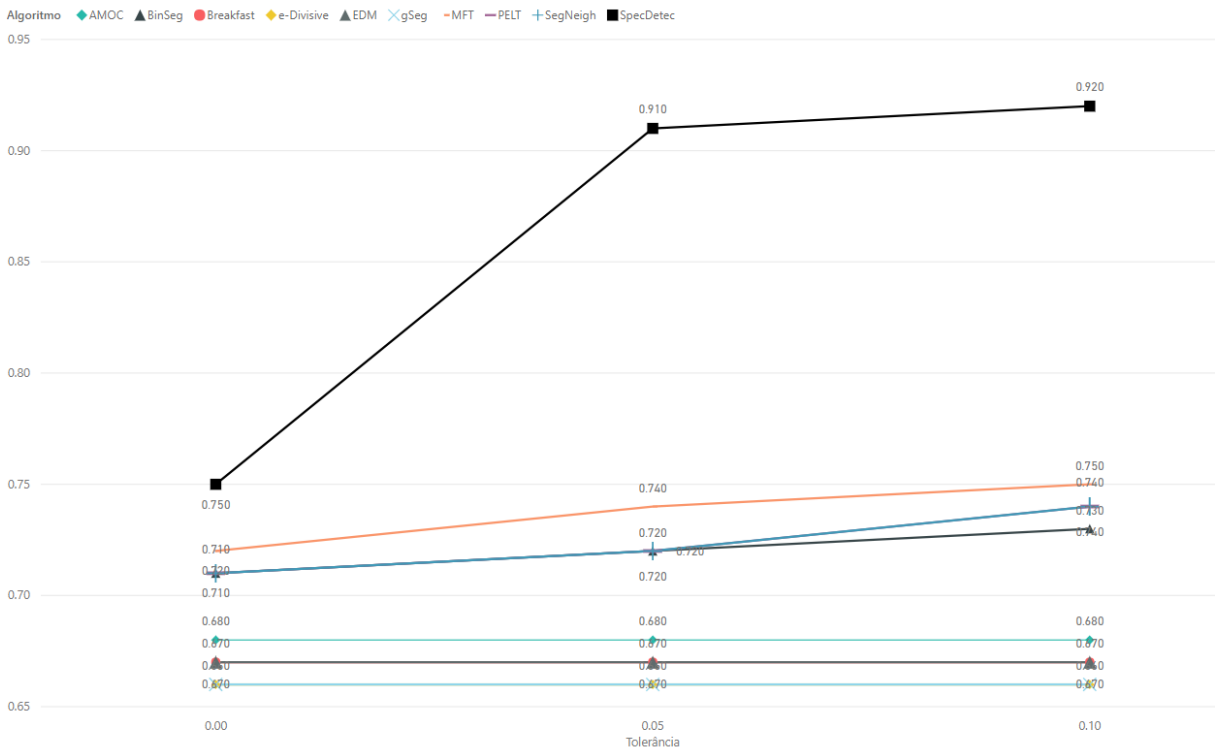


Figura 57: Comparação do Resultado do DataSet Coffee Por Tolerância

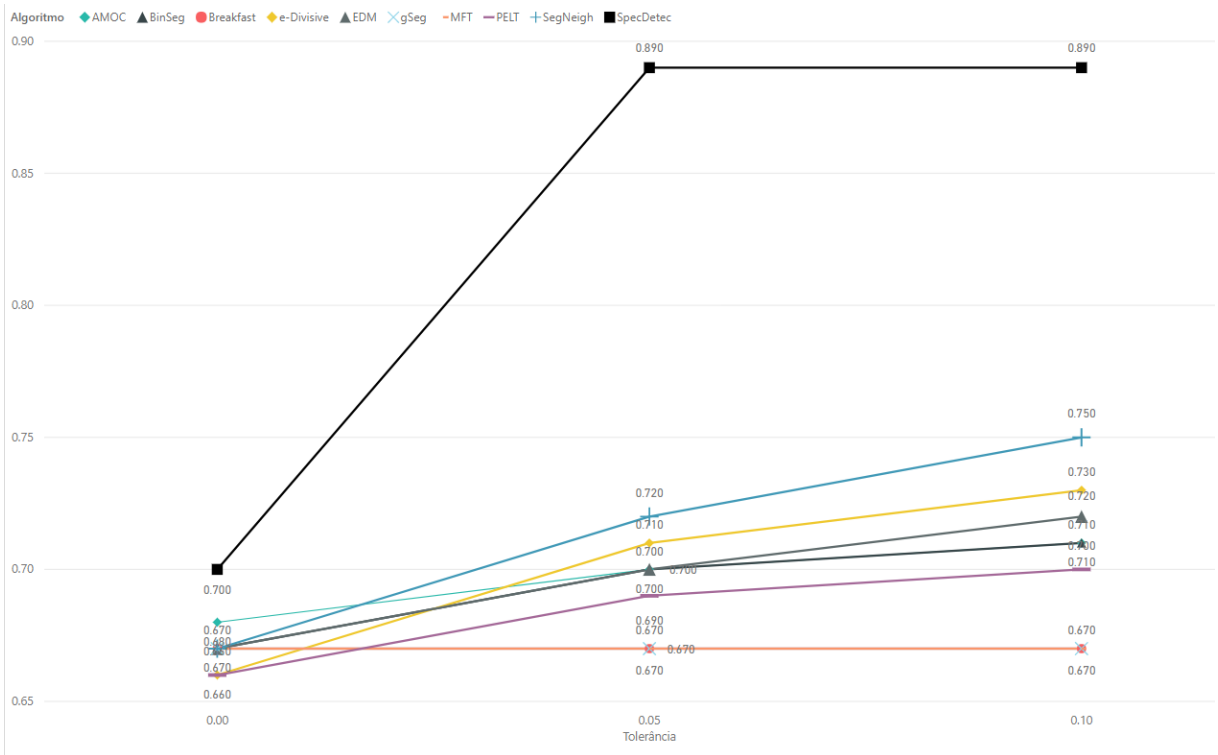


Figura 58: Comparação do Resultado do DataSet Computers Por Tolerância

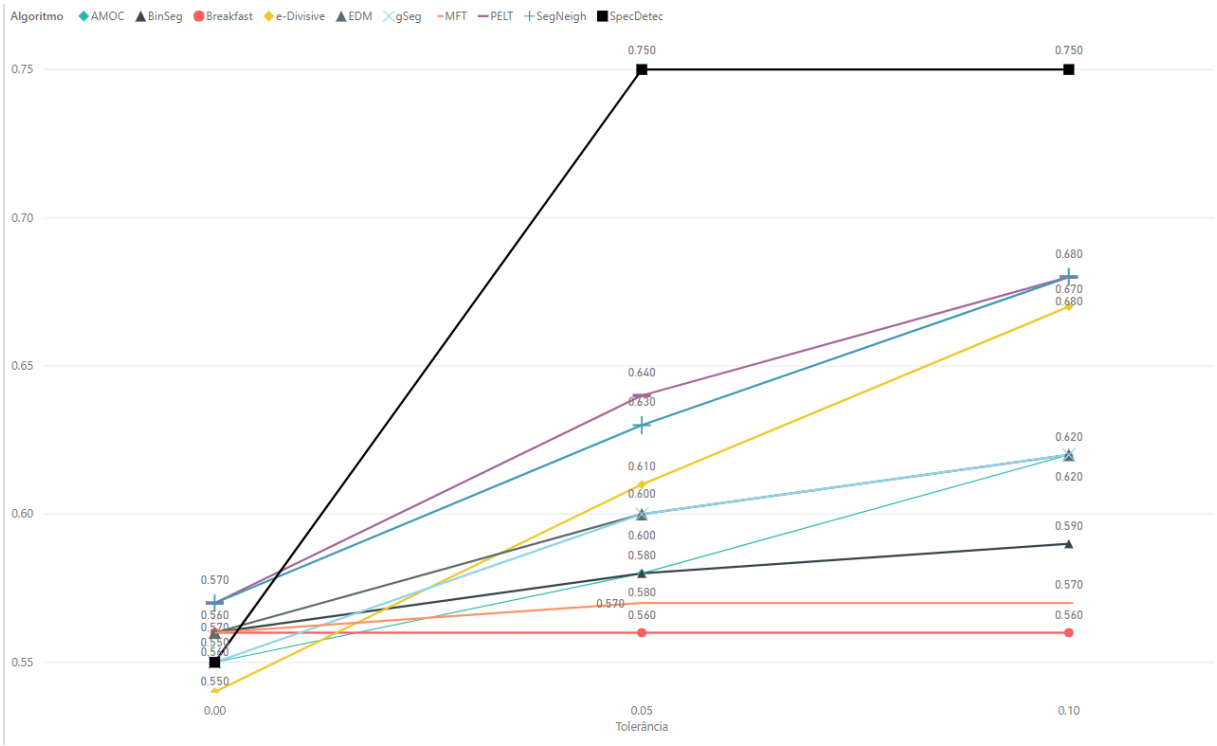


Figura 59: Comparação do Resultado do DataSet ECG200 Por Tolerância

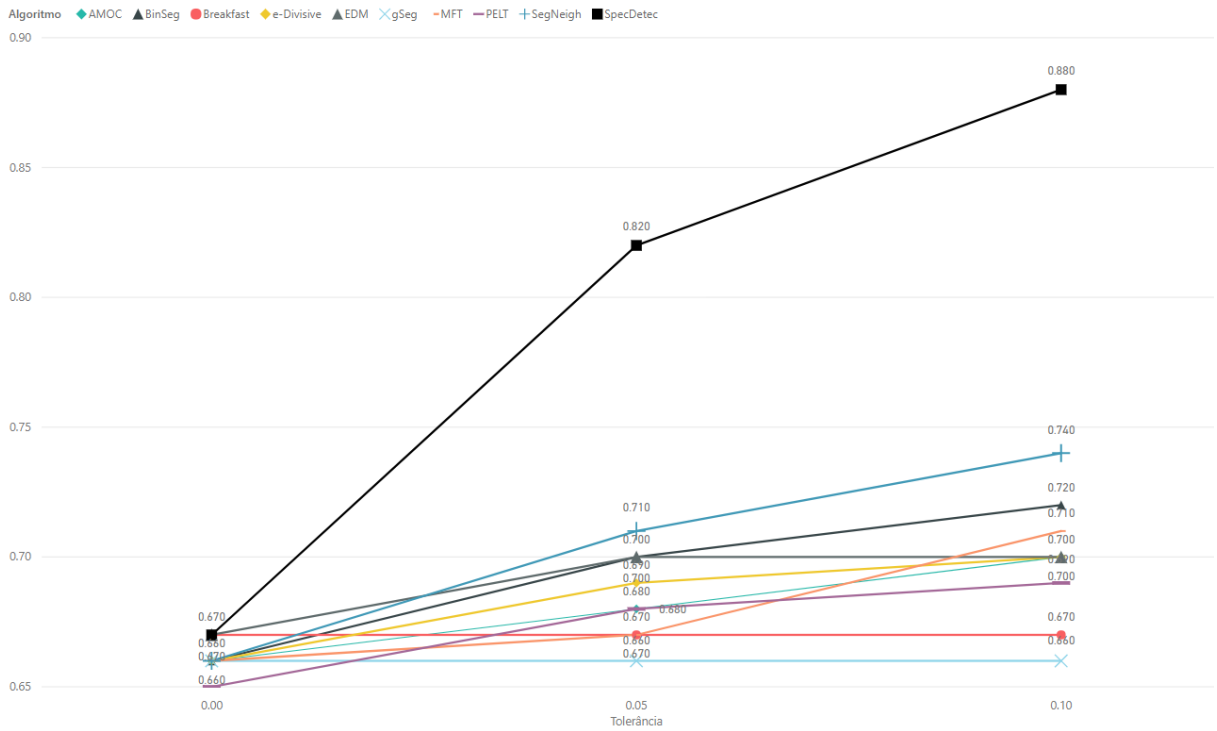


Figura 60: Comparação do Resultado do DataSet ScreenType Por Tolerância

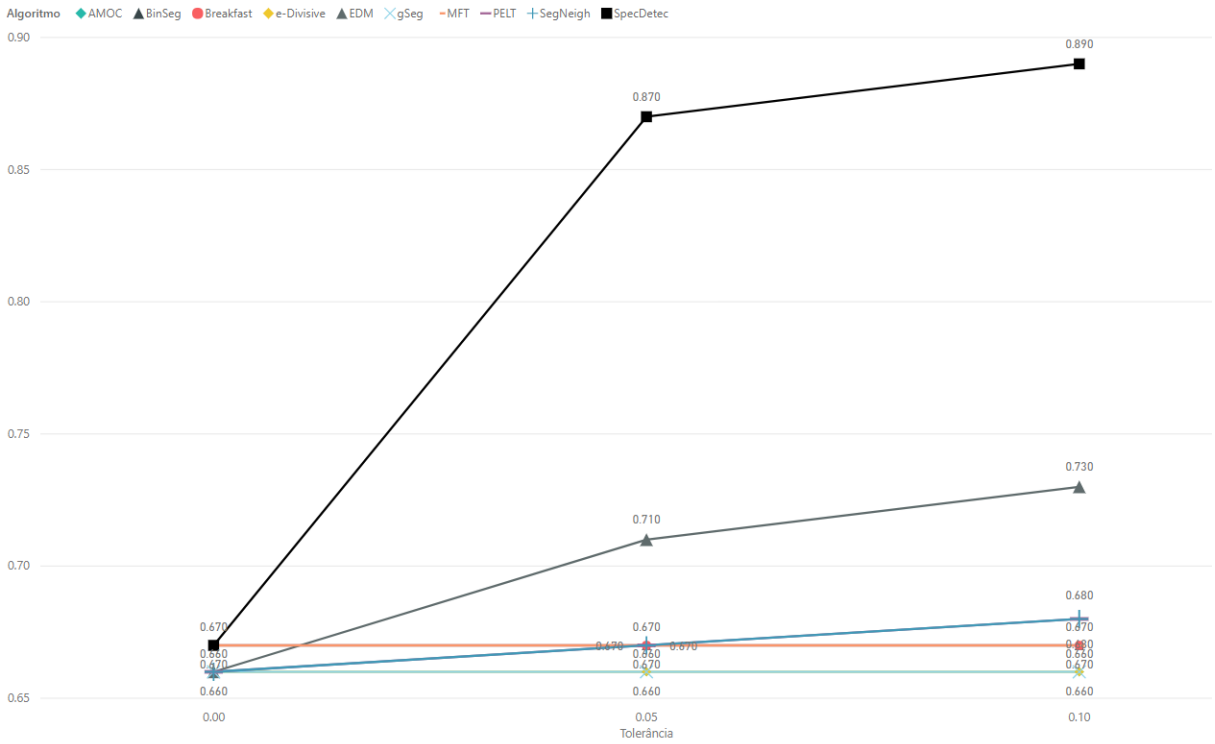


Figura 61: Comparação do Resultado do DataSet ToeSegmentation1 Por Tolerância

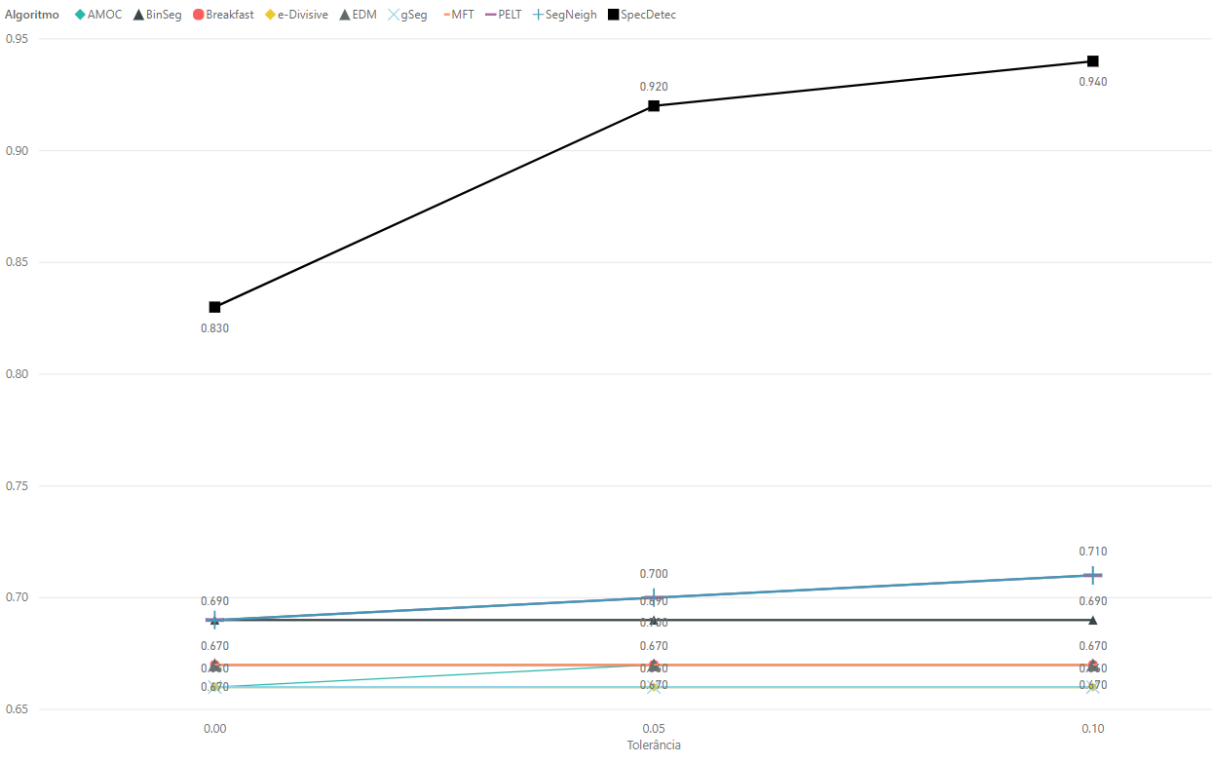


Figura 62: Comparação do Resultado do DataSet BeetleFly Por Tolerância

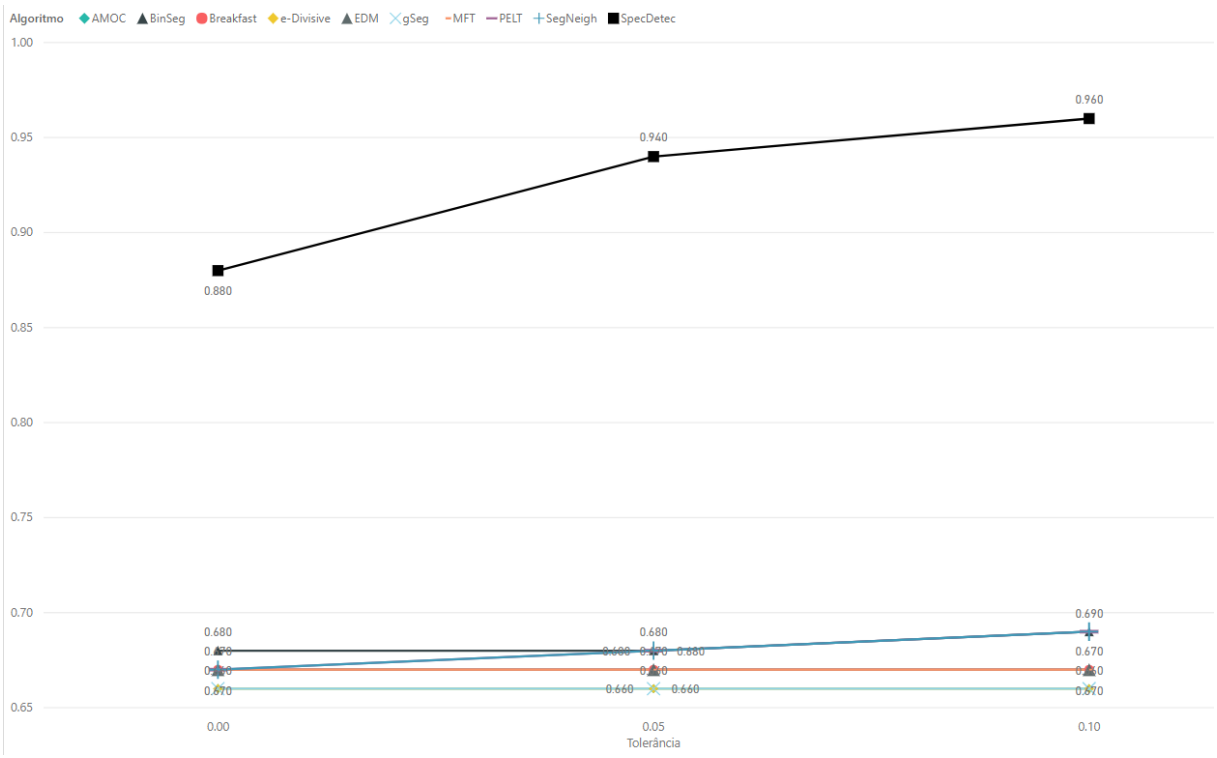


Figura 63: Comparação do Resultado do DataSet BirdChicken Por Tolerância

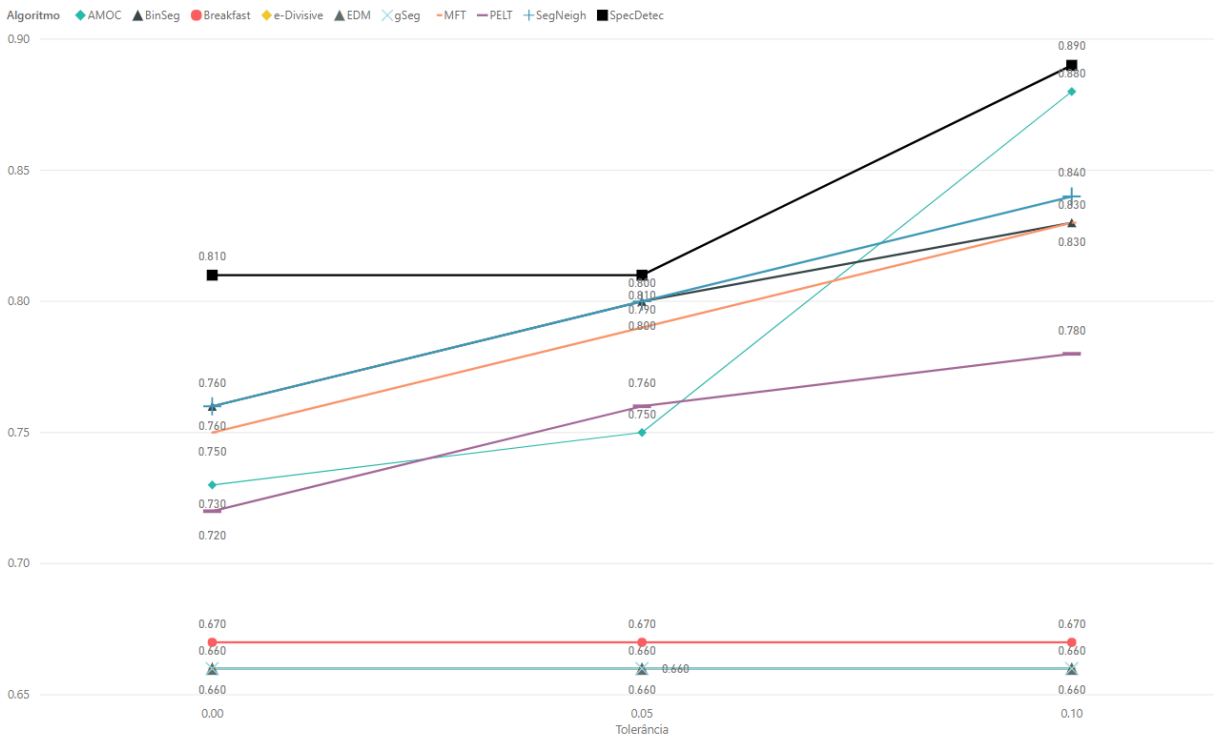


Figura 64: Comparação do Resultado do DataSet Meat Por Tolerância

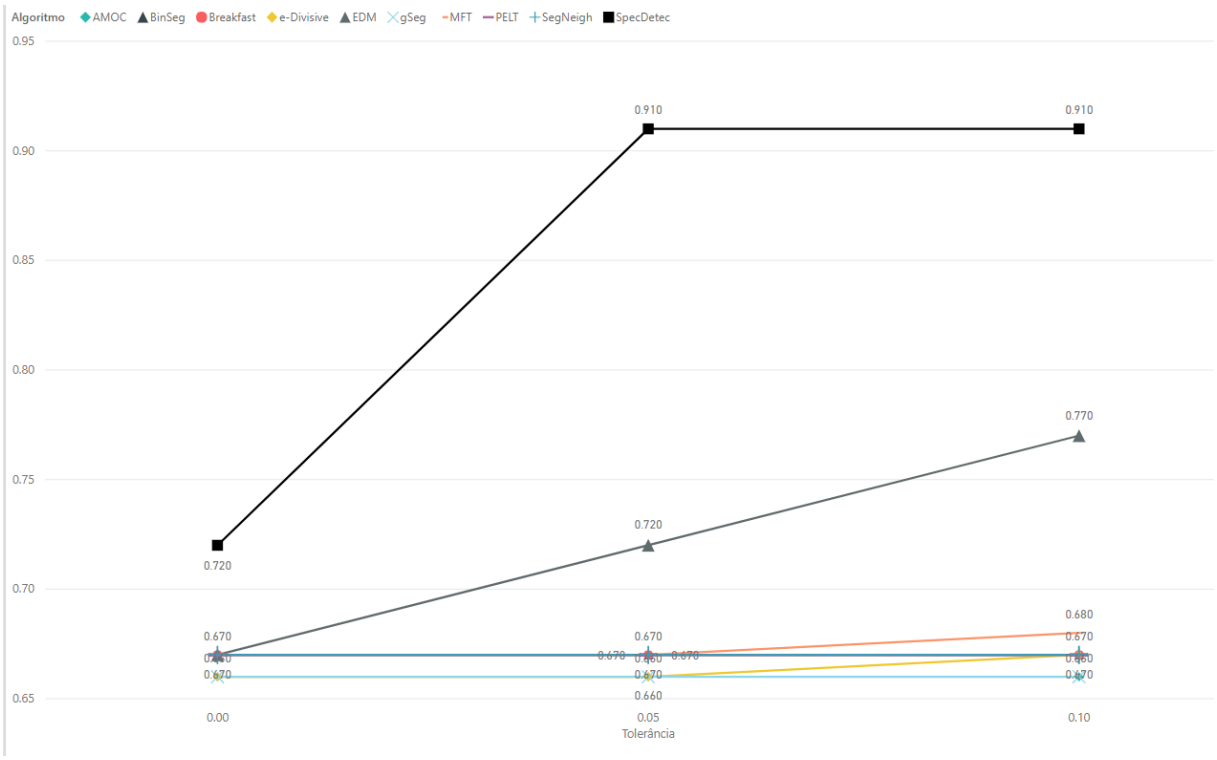


Figura 65: Comparação do Resultado do DataSet ShapeletSim Por Tolerância

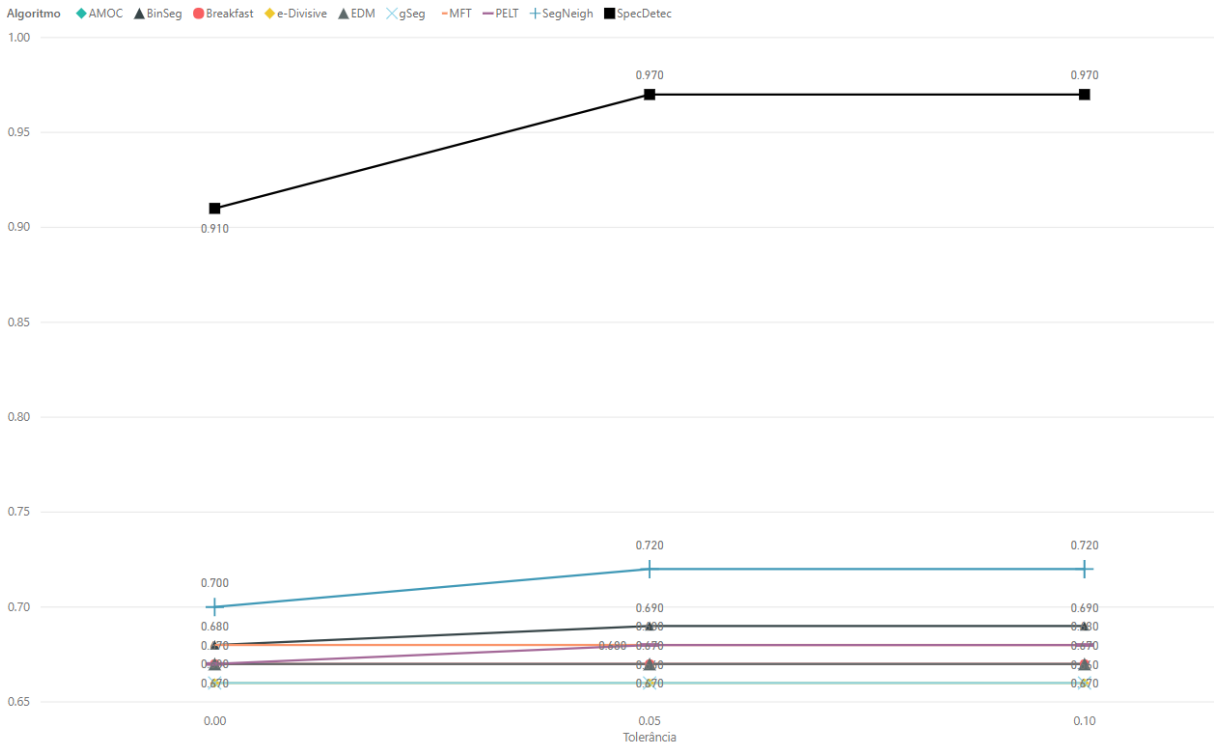


Figura 66: Comparação do Resultado do DataSet Wine Por Tolerância

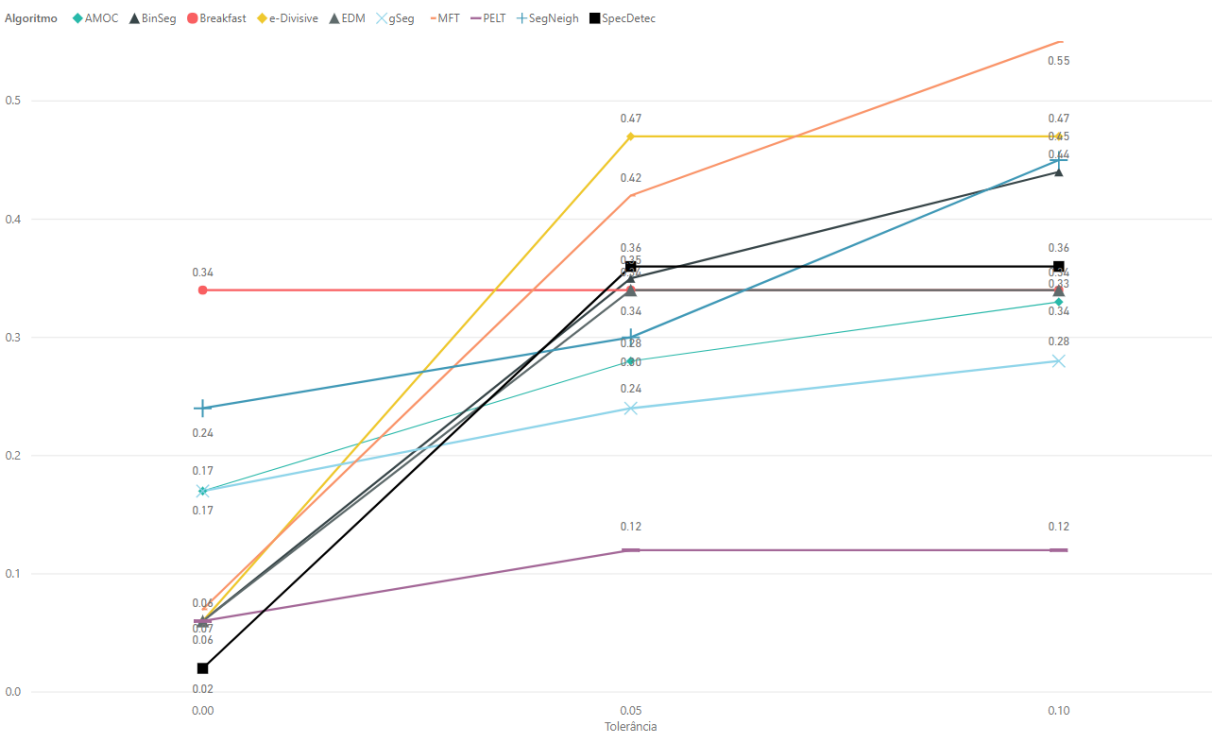


Figura 67: Comparação do Resultado do DataSet ElectricDevices Por Tolerância

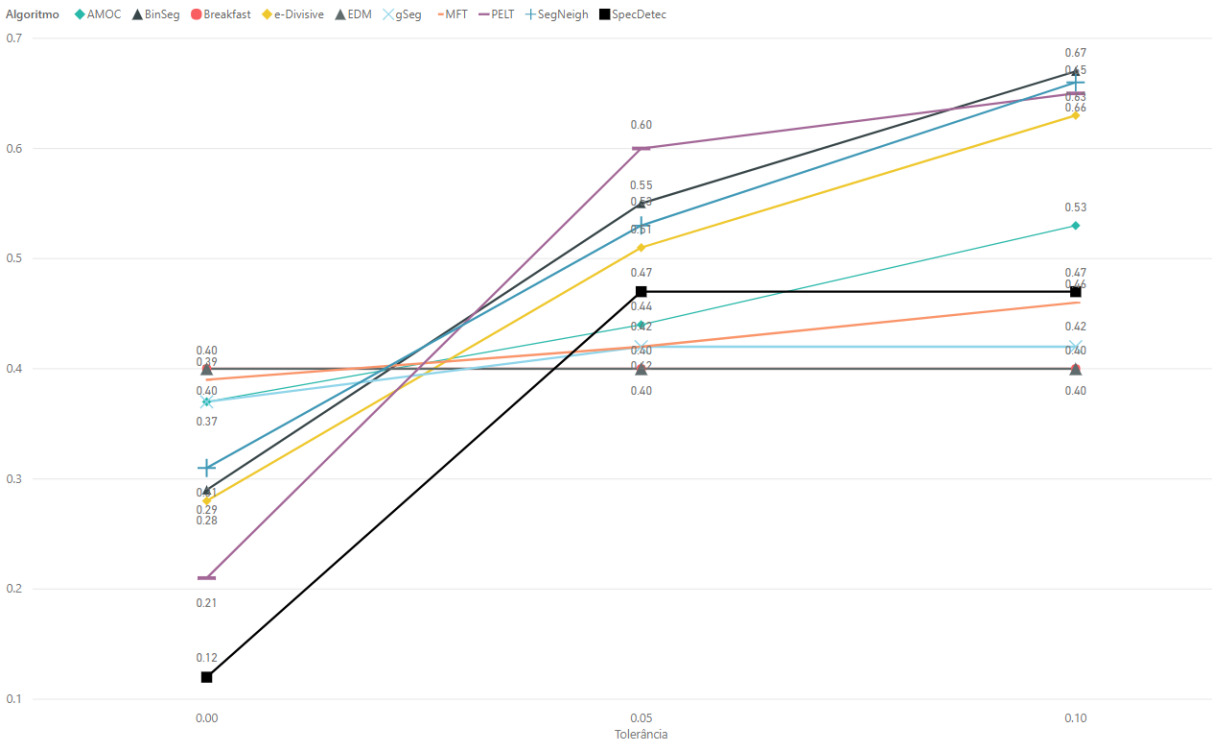


Figura 68: Comparação do Resultado do DataSet Phoneme Por Tolerância

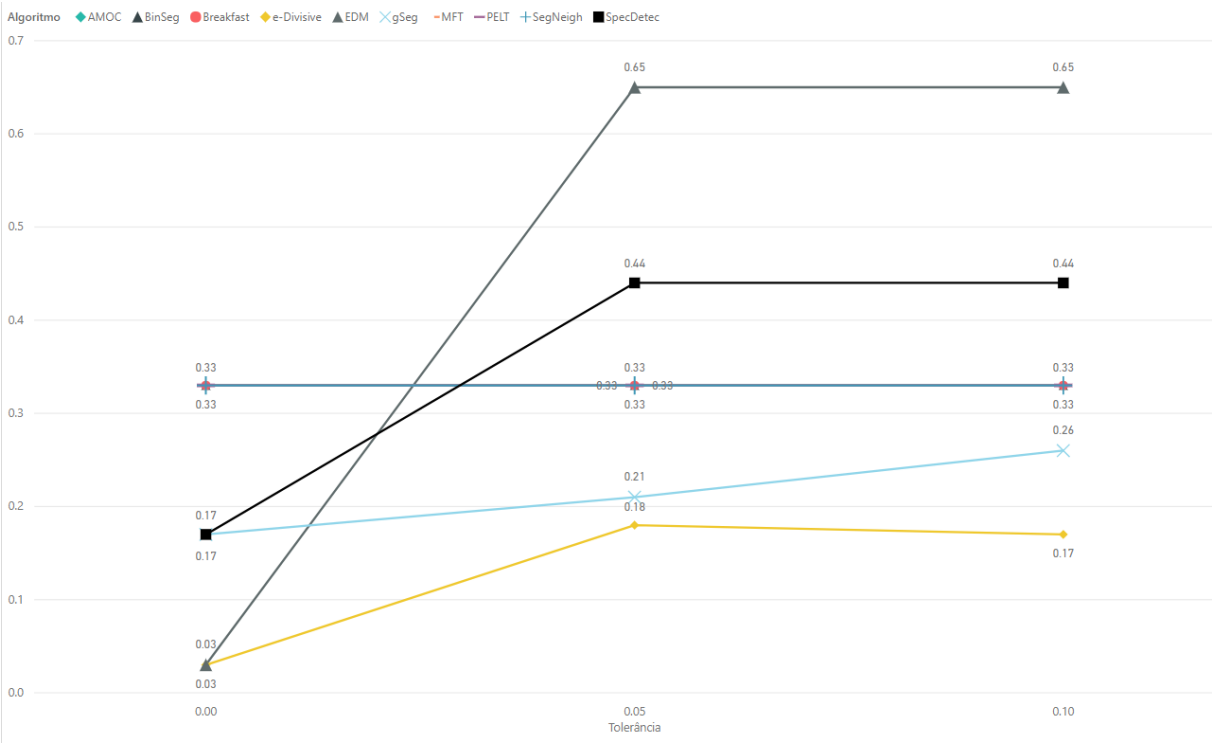


Figura 69: Comparação do Resultado do DataSet Yoga Por Tolerância