Universidade Federal do ABC

MÉTODOS COMPUTACIONAIS PARA AGRUPAMENTO DE *SPIKES* EM SINAIS INTRACRANIANOS EXTRACELULARES

Renato Aguiar Castellani

Trabalho de Graduação apresentado a Universidade Federal do ABC, como requisito parcial para obtenção do diploma no curso de graduação em Engenharia de Informação.

UFABC
Santo André
2018

Universidade Federal do ABC

MÉTODOS COMPUTACIONAIS PARA AGRUPAMENTO DE *SPIKES* EM SINAIS INTRACRANIANOS EXTRACELULARES

Renato Aguiar Castellani

Trabalho de Graduação apresentado a Universidade Federal do ABC, como requisito parcial para obtenção do diploma no curso de graduação em Engenharia de Informação.

UFABC
Santo André
2018

Castellani, Renato A.

Métodos computacionais para agrupamento de *spikes* em sinais intracranianos extracelulares / Renato Aguiar Castellani. – Santo André : UFABC, 2018.

xviii + 51 p. ; ()

Trabalho de Graduação (Graduação em Engenharia de Informação) – Universidade Federal do ABC, Santo André, 2018.

Orientador : Ricardo Suyama.

1. *Spike sorting*. 2. Sinais intracranianos extracelulares. 3. Deteção de *spikes*. 4. Disparos neuronais. 5. Agrupamento de *spikes*. (ABNT, 2002b)

Graduação em Engenharia de Informação

"As convicções são inimigas mais perigosas da verdade do que as mentiras".

Friedrich Nietzsche

RESUMO

O *spike sorting* é o processo de separação e agrupamento dos *spikes* de um sinal por neurônio de origem. Uma série de algoritmos para este propósito foram publicados ao longo dos anos, mas ainda não existe uma solução universalmente aceita. A gravação de sinal intracraniano extracelular, a técnica de inserção de um eletrodo no tecido extracelular do cérebro para registrar a atividade de neurônios individuais é um método usado por neurocientistas para estudar como o cérebro funciona. Nos últimos anos, os pesquisadores também demonstraram seu potencial uso em tecnologias médicas para o tratamento de distúrbios, como paralisia, epilepsia e perda de memória. Embora a maioria dessas aplicações exija atividade de unidade única de neurônio, esses eletrodos gravam a atividade de vários neurônios que cercam o eletrodo. Será apresentada uma visão geral do problema da agrupamento de *spikes*, será implementada uma proposta de solução para o problema e comparada com outras soluções entre as mais utilizadas.

Palavras-chave: Spike sorting, Agrupamento de spikes em sinais intracranianos e extracelulares. Spike sorting.

LISTA DE FIGURAS

Pág.

LISTA DE TABELAS

Pág.

LISTA DE ALGORÍTIMOS

LISTA DE ABREVIATURAS E SIGLAS

DPJ	–	Desvio Padrão Janelado
DOA	–	*Direction Of Arrival*
MEAs	–	*Multi-Electrode-Arrays*
MNEO	–	Operador Não-linear de Energia Multiresolução
NEO	–	Operador N]ao-linear de Energia
PCA	–	*Principal Component Analysis*
SNEO	–	Operador Não-linear de Energia Suavizado
SPC	–	*Superparamagnetic Clustering*
SS	–	*Spike Sorting*
VA	–	Valor Absoluto
VC	–	*Volume Conductor*
WT	–	*Wavelet Transform*

SUMÁRIO

Pág.

1 Introdução

A aquisição e o processamento eficiente das atividades neuronais são pré-requisitos para vários tipos de estudo de funções cerebrais e constituem um desafio teórico e metodológico atual. Atuando-se de forma baseada nas atividades neuronais, podem-se realizar terapias para perdas de memória, cognição, paralisia e epilepsia, estudos sobre percepção, cognição, movimento, memória e interação homem-computador e cérebro-máquina.

As gravações eletrofisiológicas podem ser feitas a partir do interior de células (intracelulares) ou do seu exterior (extracelulares). Em estudos do sistema nervoso central, os eletrodos de pequeno diâmetro podem ser posicionados no espaço extracelular para registrar sinais elétricos dos neurônios ao seu redor. Esses eletrodos são capazes de detectar potenciais de ação de neurônios. A capacidade de gravação extracelular pode fornecer aos pesquisadores gravações de atividades de neurônios individuais com nível de dificuldade relativamente baixo para se realizar ao se comparar com a gravações intracelulares. Isso levou a gravação extracelular a se tornar uma técnica dominante em muitos estudos. Por exemplo, tem havido um movimento na pesquisa de neurociência para estudar não apenas neurônios individuais, mas redes de neurônios para entender como a atividade de neurônios interconectados resulta em funções de ordem superior como percepção, compreensão, movimento e memória. Tais estudos exigem a gravação extracelular de conjuntos de neurônios usando arrays de eletrodos multicanal. Em *Methods for Neural Ensemble Recordings* (Métodos para Gravações do Conjunto Neural), os autores Sameshima e Baccalá chegam a afirmar que "as gravações extracelulares são a única escolha prática em experimentos que pretendem estabelecer correlações entre respostas de conjunto neural e comportamentos envolvendo animais acordados" (NICOLELIS, 2007).

Avanços tecnológicos recentes possibilitam a gravação simultânea da atividade de um grande número de neurônios em animais acordados e com comportamento natural, utilizando-se eletrodos intracranianos no meio extracelular. A técnica de gravação extracelular já foi utilizada em diversas tecnologias médicas para o tratamento de distúrbios, como paralisia (NICOLELIS, 2001), (HOCHBERG et al., 2006), epilepsia (NICOLELIS, 2001) e mesmo perda cognitiva e de memória (BERGER et al., 2005). Assim, o estudo de um conjunto de neurônios utilizando-se desses eletrodos tem sido uma área de pesquisa relevante nos últimos anos (LEWICKI, 1998; QUIROGA; PANZERI, 2009; GIBSON et al., 2012; GIBSON et al., 2008; DAYAN; ABBOTT, 2005).

Uma forma de estudar os sinais neuronais é por meio da análise dos disparos de

neurônios obtidos a partir de uma série temporal adquirida por redes de eletrodos alocadas em determinadas posições do cérebro (DAYAN; ABBOTT, 2005). Essa série é formada por trechos curtos de sinais com formato de espículas (*spikes*), que representam disparos ou potenciais de ação dos neurônios próximos aos eletrodos. Além disso, ela é corrompida por outros sinais de origem biológica e de medição, usualmente modelados como ruído aditivo. A análise de dados provenientes de um único neurônio não é simples pelo fato de mais de um neurônio ser captado no registro de um mesmo eletrodo. Tipicamente, cada eletrodo pode capturar o sinal gerado por dois a dez neurônios simultaneamente (PEDREIRA et al., 2012). Nesse trabalho será implementada uma proposta de solução para o agrupamento de *spikes* e essa será comparada com outras soluções entre as mais utilizadas.

1.1 A detecção e o agrupamento de *spikes*

O spike sorting é o processo de separação e agrupamento dos spikes de um sinal por neurônio de origem. Uma série de algoritmos para este propósito foram publicados ao longo dos anos, mas ainda não existe uma solução universalmente aceita. O problema de detectar os *spikes* e agrupá-los de acordo com o neurônio que o emitiu é chamado de *spike sorting* (QUIROGA; PANZERI, 2009; DAYAN; ABBOTT, 2005; RUTISHAUSER et al., 2006; GIBSON et al., 2012). O *spike sorting* é realizado a partir do sinal proveniente do eletrodo e, em geral, segue três etapas: detecção, parametrização e agrupamento. Representações do sinal em cada uma dessas etapas são ilustradas na Figura 1.1.

O sinal tem sua medição registrada pelo eletrodo e é filtrado usualmente por um passa faixa entre 300 e 3000 Hz (REY et al., 2015), um exemplo de sinal é representado no canto superior direito da Figura 1.1. O objetivo do filtro é de separar o sinal de interesse que é o de disparos individuais de um neurônio dos sinais de baixa frequência associados a um comportamento e grupo de neurônios no cérebro e também de ruídos de altas frequências. A etapa da detecção faz inicialmente um pré-processamento nesse sinal para realçar a presença dos *spikes* em relação ao ruído. Posteriormente, aplica-se um limiar a fim de encontrar os instantes em que ocorrem os *spikes* em meio ao sinal ruidoso. Na Figura 1.1, abaixo do sinal bruto filtrado, pode-se observar um exemplo de sinal pré-processado com limiar e a indicação dos *spikes*.

A *parametrização* ou *extração de características* é a etapa em que os *spikes* detectados são transformados em um conjunto de parâmetros que enfatizam as diferenças entre eles. Em geral, busca-se um número de parâmetros significativamente menor do que o número de amostras usadas na representação temporal do *spike* detectado.

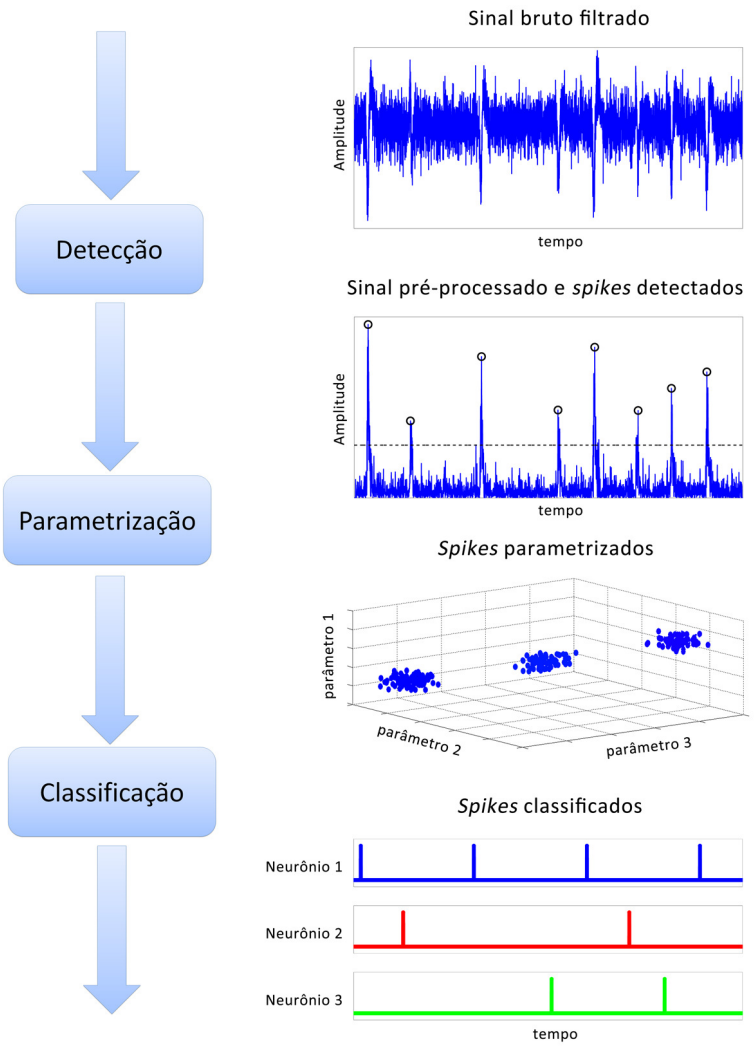

Figura 1.1 - Fluxograma das etapas do *spike sorting* com sinais ilustrativos de cada passo (CASTELLANI, 2014).

Com a parametrização, é possível realizar uma redução de dimensionalidade, isto é, representar as informações do *spike* de N amostras, por meio de K_C coeficientes, sendo $K_C \ll N$ (GIBSON et al., 2012). No caso do exemplo da Figura 1.1 cada *spike* é representado com três parâmetros, $K_C = 3$. Observe-se que esses parâmetros formam agrupamentos distintos no espaço dos parâmetros. É essencial nesta etapa guardar o instante de tempo associado a cada conjunto de parâmetros para a etapa seguinte.

O *agrupamento* ou *agrupamento* é a etapa em que os *spikes* são separados em grupos de acordo com o neurônio que o emitiu. É realizada utilizando-se diferenças estatísticas entre os parâmetros do conjunto de características extraídas. No caso do exemplo da Figura 1.1, como os parâmetros formaram três agrupamentos distintos,

considera-se que os *spikes* observados no sinal bruto filtrado são provenientes de três neurônios distintos. Pode-se usar então a informação dos instantes de disparo juntamente com o agrupamento a que pertence um dado disparo para gerar sinais como os do canto inferior direito da Figura 1.1, que representam os disparos de um dado neurônio. No caso, mostram-se os instantes de disparos dos três neurônios distintos que originaram o sinal bruto filtrado.

1.2 A diferença entre potencial de ação e o *spike*

Os neurônios são células longas que podem propagar informações de uma região a outra do sistema nervoso (GUYTON, 1984; DAYAN; ABBOTT, 2005). Eles são formados por um corpo celular e alguns prolongamentos. Os prolongamentos do corpo celular podem ser divididos em dois grupos: os *dendritos* e o *axônio*. Os dendritos são os responsáveis pelo recebimento de informações pelos neurônios. Cada neurônio faz em torno de 100 000 ligações por meio de dendritos recebendo em média duas entradas por micrometro. O axônio de um neurônio transmite o sinal gerado no corpo celular para outros neurônios. Eles são longos e podem alcançar neurônios distantes no cérebro e até no restante do corpo. Em média axônios fazem 180 ligações por milimetro de extensão (DAYAN; ABBOTT, 2005). Os neurônios são células cuja função é gerar sinais elétricos como respostas a estímulos químicos e elétricos e de propagá-los para outras células rapidamente por longas distâncias. Esse processo é realizado gerando um pulso elétrico característico, que viaja pelas fibras nervosas. As informações são transmitidas pelos neurônios por sequências de disparos elétricos em padrões temporais e intensidades diferentes (DAYAN; ABBOTT, 2005; GUYTON, 1984).

A membrana celular funciona como uma porta entre os meios intra e extra celulares e regula a permeabilidade aos íons, controlando assim uma diferença de potencial. Em repouso, o potencial de membrana é aproximadamente -70 mV, isto é, negativa em relação ao meio exterior que define-se com potencial 0 mV. O *disparo neuronal* é o processo no qual, em curto espaço de tempo, o potencial de membrana passa a positivo, atinge um pico e depois volta a ser negativo. Ele pode ser dividido em quatro estágios chamados de repouso, despolarização, repolarização e hiperpolarização, conforme esquematizado na Figura 1.2.

A variação de potencial de um neurônio pode ser gravada de maneira *intracelular* ou *extracelular* dependendo da posição do eletrodo. O mesmo evento de disparo neuronal tem características elétricas diferentes dependendo da metodologia de aquisição do sinal. O sinal é chamado de *potencial de ação* quando o eletrodo está no interior

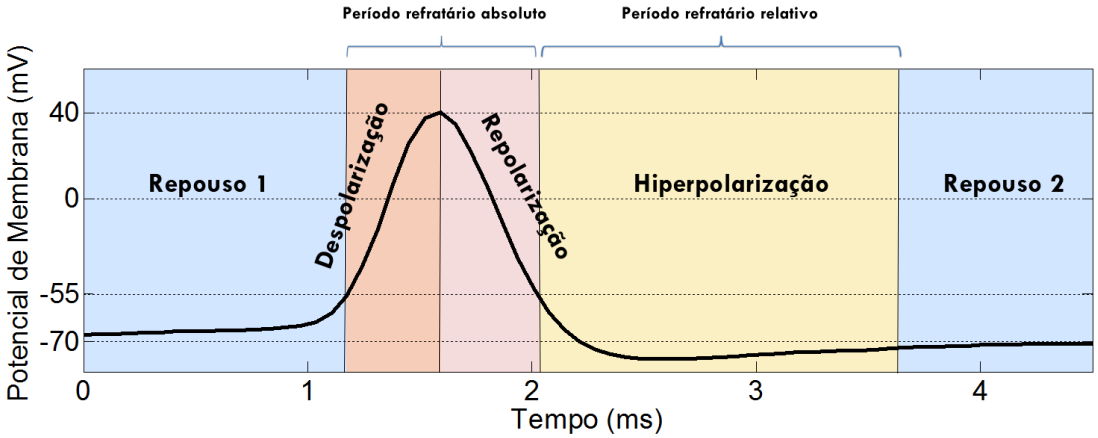

Figura 1.2 - Fases de um potencial de ação típico: repouso, despolarização, repolarização, hiperpolarização e novamente repouso (CASTELLANI, 2014).

do neurônio e é chamado de *spike* no caso em que o sensor está no exterior da célula (GIBSON et al., 2012).

Na Figura 1.3 é apresentado um trecho de sinal (a) medido no meio intracelular e (b) o mesmo disparo medido no meio extracelular. Essas aquisições foram obtidas da base de dados CRCNS (HENZE et al., 2000; CRCNS, 2000) de um trecho de gravação da região do hipocampo de um rato anestesiado. O procedimento de aquisição é decrito no trabalho de Henze et al. (HENZE et al., 2000). Pode-se observar que, de fato, o sinal extracelular V_{ex} é aproximadamente igual à derivada do sinal intracelular V_{in} com o sinal trocado. Estes sinais foram normalizados por amplitude nesse caso para se realizar uma comparação.

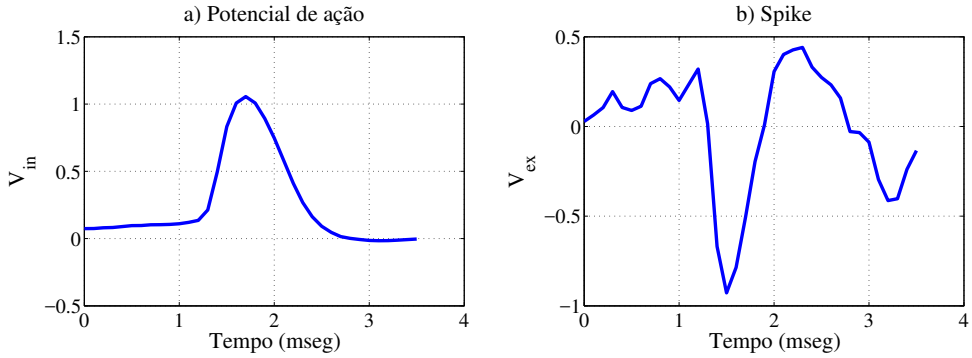

Figura 1.3 - Sinal de interesse com amplitude normalizada: (a) sinal intracelular; (b) sinal extracelular (CRCNS, 2000) (CASTELLANI, 2014).

O formato do sinal de disparo intracelular depende de sua geometria, da distribuição iônica, do tipo de célula e do ruído de medição. Já o formato do sinal extracelular depende da posição do eletrodo em relação ao neurônio, da quantidade de neurônios vizinhos e da região específica do cérebro em que foi posicionado o eletrodo, além dos mesmos fatores que influenciam o sinal intracelular (GIBSON et al., 2012). Outra grande diferença entre os tipos de medição é que os sinais intracelulares têm amplitude com ordem de magnitude três vezes maior que os extracelulares, isto é, sinais extracelulares têm amplitude da ordem de dezenas a centenas de microvolts, enquanto os intracelulares têm amplitude da ordem de centenas de milivolts (GIBSON et al., 2012). E isso acontece devido a posição do eletrodo, internamente a célula é menos propenso a ruídos, já que a membrana da célula atua como isolante. Gravações intracelulares apresentam maior confiabilidade e menor ruído nos resultados obtidos, já que contam com uma isolação provida pela membrana celular em relação a interferências externas ao neurônio. Porém, elas raramente são feitas em animais vivos por serem difíceis de se realizar. A implantação de um eletrodo dentro de um neurônio de um animal vivo sem danificar esta célula é uma limitação técnica. Sinais intracelulares são adquiridos de maneira mais comum em preparações *in vitro*, com pedaços de tecidos neuronais (DAYAN; ABBOTT, 2005).

Podem ser observados na Figura 1.4 outros exemplos de *spikes*. Esses trechos de sinais extracelulares são correspondentes a *spikes* detectados, alinhados e sobreamostrados a uma taxa de 100 kHz disponibilizados por Rutishauser et al. (RUTISHAUSER et al., 2006). Cada *spike* é composto por 256 amostras, ou 2,56 ms. No caso da Figura 1.4 pode-se observar que os *spikes* tem formatos consideravelmente diferentes entre si, apesar dos potenciais de ação de origem apresentarem características muito próximas. O motivo desta diferença é explicada pelos diferentes percursos que os sinais percorrem do interior dos neurônios até o eletrodo de medição, devido às diferentes posições relativas dos neurônios até o eletrodo, ou seja, uma diferença no canal.

1.3 Objetivos

Os objetivos deste trabalho são desenvolver e implementar um método de agrupamento de *spikes* baseado em técnicas comuns de detecção utilizadas na literatura (CASTELLANI, 2014) com extração de características por PCA e agrupamento por k-means. A técnica será comparada com os outros métodos tradicionais propostos por Quiroga (QUIROGA et al., 2004) e Rutishauser (RUTISHAUSER et al., 2006). Será utilizado um gerador de sinal controlado que simula de maneira tridimencional sinais neuronais extracelulares chamado NeuroCube (CAMUÑAS-MESA; QUIROGA, 2013) de

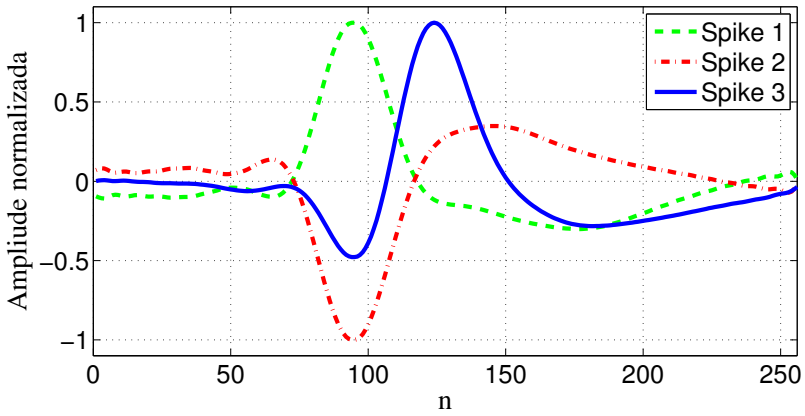

Figura 1.4 - *Spikes* adquiridos com eletrodos extracelulares (RUTISHAUSER et al., 2006; CASTELLANI, 2014).

modo que possa realizar as comparações.

A estrutura do trabalho foi dividida de forma que Capítulo 2 apresenta a revisão teórica fundamental para o problema de agrupamento de disparos neuronais *Spike Sorting*. Na seção 2.1 é descrito o gerador de sinais bioinspirado (CAMUÑAS-MESA; QUIROGA, 2013). Na seção 2.2 é descrita a técnica WaveClus (QUIROGA et al., 2004). Na seção 2.3 é descrita a técnica OSort (RUTISHAUSER et al., 2006). No capítulo 3 são apresentados cada um dos métodos utilizados no trabalho de (CASTELLANI, 2014) que foram implementados e comparados com as técnicas anteriores e também as técnicas de redução de dimensionalidade e agrupamento utilizadas. No Capítulo 4 são apresentadas as metodologias e formas de avaliação dos testes realizados e seus resultados e discussões. No Capítulo 5 é apresentadas as conclusões e as propostas para trabalhos futuros.

2 Técnicas de *Spike sorting*

Neste capítulo são descritos o funcionamentos do gerador de sinais e dos métodos de agrupamento de *spikes* utilizados. Na seção 2.1 é apresentado o gerador de sinais intracraniamos extracelulares bioinspirado. Na seção 2.2 é apresentada a técnica WaveClus (QUIROGA et al., 2004) que utiliza o valor absoluto do sinal para fazer a detecção, trasformada *wavelet* e algoritmo para agrupamento Superparamagnético. Na seção 2.3 é apresentada a técnica OSort (RUTISHAUSER et al., 2006), que utiliza o desvio padrão janelado para realizar a detecção e um algoritmo próprio para agrupamento.

2.1 Gerador de sinais *NeuroCube*

O *NeuroCube* é um gerador de sinais de gravações neuronais intracranianas extracelulares simulados desenvolvido por Quiroga et al. (CAMUÑAS-MESA; QUIROGA, 2013). O sinal resultante é obtido a partir do somatório de sinais gerados por um modelo biofísico detalhado de neurônios localizados aleatoriamente em uma abstração de um cubo de tecido cerebral ao redor de um eletrodo no centro desse cubo. Os neurônios mais próximos foram gerados por um modelo detalhado da atividade única de um neurônio e os mais distantes por sinais reais previamente gravados para reduzir o custo computacional e que contribuem como ruído de fundo.

O modelo de sinal neuronal extracelular é definido por duas zonas em relação a distância do eletrodo: a zona A, com neurônios próximos, distância menor que 150 μm, os quais foram simulados usando um modelo comportamental de sinais neuronais; e a zona B, para neurônios mais distantes que 150 μm, aos quais foram atribuídos aleatoriamente uma forma de *spike* obtida de um banco de dados de 594 neurônios gravados a priori. A distância de 150 μm foi definida a partir de estudos realizados e descritos com mais detalhes em (CAMUÑAS-MESA; QUIROGA, 2013). O sinal também é composto por ruído térmico de modo a considerar outras possíveis fontes de atividade elétrica que podem contribuir para o espectro de potência de ruído, como o ruído eletrônico, axônios, dendritos e correntes sinápticas (MARTINEZ et al., 2009). Para simular esse efeito, foi adicionado um ruído gaussiano com média zero e um desvio padrão de 1 μV (CAMUÑAS-MESA; QUIROGA, 2013).

2.1.1 Modelo comportamental

O modelo comportamental simula com precisão o comportamento biofísico das formas de *spike* de um neurônio em diferentes locais. O modelo comportamental de

sinais neuronais descreve a morfologia celular (soma, axônios e árvore dendrítica) e inclui a variedade de canais iônicos e densidades de condutância correspondentes (CAMUÑAS-MESA; QUIROGA, 2013). O modelo comportamental de neurônios utilizado para gerar potenciais de ação extracelular neste trabalho foi baseado na Aproximação da Fonte de Linha (LSA - Line Source Approximation) que é um método de cálculo do potencial elétrico extracelular resultante das correntes distribuídas em toda a superfície 3-D da membrana de um neurônio. O eletrodo foi modelado considerando os efeitos de tamanho finito, calculando a média em sua superfície e considerando os efeitos de filtragem de capacitância.

O disparo de cada neurônio seguiu uma distribuição de Poisson, com uma taxa de disparo média atribuída de acordo com a distribuição das taxas médias de disparo observadas em gravações reais do lóbulo temporal medial (distribuição exponencial com média de 1,25 Hz para células piramidais e cinco vezes maior para interneurônios) (QUIROGA et al., 2007).

Quando se leva em conta os muitos neurônios que contribuem para um sinal extracelular, simular o comportamento detalhado de todos os neurônios com o modelo comportamental gera uma carga computacional muito grande. No caso de uma simulação de um cubo de 1 mm^3 de hipocampo seriam 24 mil neurônios a serem simulados, considerando 8% de neurônios ativos e uma densidade neuronal de 300 mil neurônios por mm^3. Em um computador normal, o tempo médio de processamento para gerar cada forma de *spike* é de cerca de 3 segundos, o que significa que cerca de 20 horas seriam necessárias para calcular as formas de *spike* para todos os neurônios individuais dentro do cubo e cerca de 57 dias para simular a gravação com um eletrodo de tamanho finito (CAMUÑAS-MESA; QUIROGA, 2013). Para reduzir o tempo de processamento, considerou-se o fato de que as formas exatas dos picos de neurônios distantes não precisam ser modeladas em detalhes, já que contribuem apenas para o ruído de fundo. (CAMUÑAS-MESA; QUIROGA, 2013). Portanto, há um compromisso entre modelagem detalhada e carga computacional a ser considerada no simulador.

Para os dados reais foram utilizadas 624 gravações de cerca de meia hora, realizadas em 13 sessões experimentais com 48 canais cada, em dois pacientes implantados com eletrodos permitindo gravações intracranianas por razões clínicas.

2.2 Técnica WaveClus

O WaveClus é um algoritmo de detecção e ordenamento de *spikes* sem supervisão que combina a transformada *wavelet* com agrupamento superparamagnético (SPC - *Superparamagnetic Clustering*), que é um método usado na mecânica estatística (QUIROGA et al., 2004). Permite agrupar os dados sem pressupostos, como baixa variação ou sem distribuições gaussianas. No primeiro passo, os dados são filtrados por um filtro passa-altas e têm os *spikes* detectados, dado um limite de amplitude automático calculado. No segundo passo, um pequeno conjunto de coeficientes de *wavelet* de cada pico é escolhido como entrada para o algoritmo de agrupamento. Finalmente, o SPC agrupa os picos de acordo com o conjunto selecionado de coeficientes de *wavelet* (QUIROGA et al., 2004). O algoritmo foi implementado em MATLAB e pode-se observar a interface gráfica da aplicação na Figura 2.1.

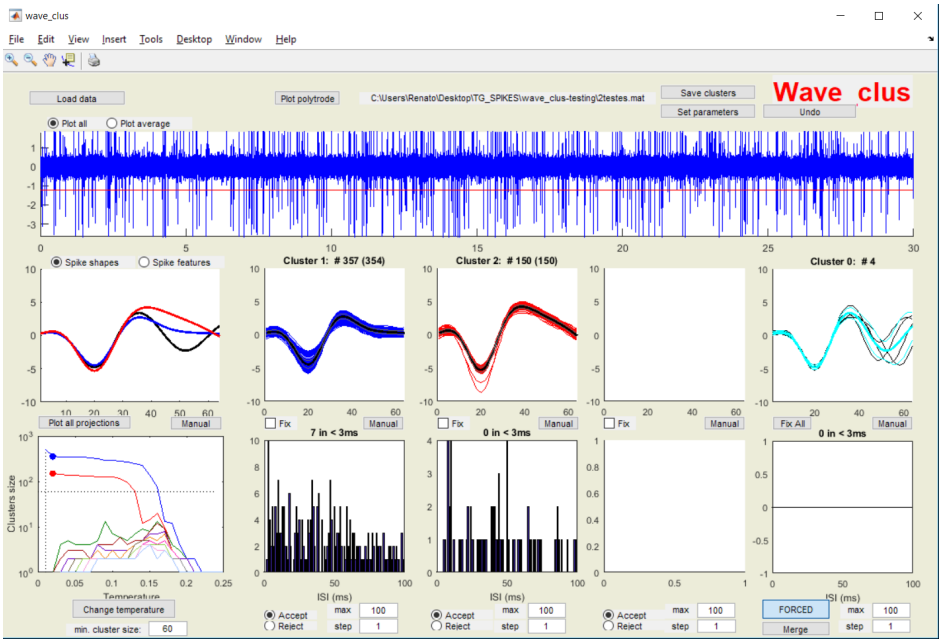

Figura 2.1 - Interface gráfica do WaveClus (QUIROGA et al., 2004).

Os três estágios principais do algoritmo são (1) detecção automática de *spikes* por limiar de amplitude, (2) a transformada *wavelet* é calculada para cada um dos *spikes* e os coeficientes para separar as classes de *spikes* são selecionados automaticamente e (3) os coeficientes de *wavelet* selecionados servem então como entrada para o algoritmo SPC, e o agrupamento é realizado após a seleção automática da temperatura correspondente à fase superparamagnética. A Tabela descreve os principais parâmetros do algoritmo WaveClus 2.1.

Tabela 2.1 - Principais parâmetros do algoritmo WaveClus

force auto	Forçar automaticamente a associação de spikes atribuídos ao cluster de ruído usando correspondência de modelo
inputs (entradas)	Número de coeficientes wavelet a serem usados como recursos para armazenamento em cluster
KNearNeighb (K vizinhos próximos)	Número de pontos de dados usados para as interações dos vizinhos mais próximos no SPC
min clus stop (parada de mínimo agrupamento)	Tamanho mínimo de um cluster (o cluster será excluído se o número de spikes contidos for menor que esse valor)
Mintemp (temperatura mínima)	Temperatura mínima do SPC - um valor de temperatura mais baixo agrupa todos os dados em um único cluster, enquanto valores mais altos permitem que os dados sejam divididos em mais clusters
scales (escalas)	Número de níveis de decomposição de wavelets usados
SWCycles (ciclos SW)	Número de iterações de Monte Carlo usadas pelo SPC
template type (tipo de modelo)	Tipo de método de correspondência de modelo usado - a correspondência de modelo é usada para acelerar a classificação de pico no caso de um grande número de picos ou para atribuir picos no cluster de ruído aos clusters existentes (se a força automática estiver definida)

2.2.1 Transformação Wavelet

A transformada *wavelet* (WT - *wavelet transform*) é uma representação tempo-frequência do sinal que tem duas vantagens principais em relação aos métodos convencionais: proporciona uma resolução nos domínios de tempo e frequência e elimina o requisito do sinal ser estacionário . É definida como a convolução entre o sinal x(t) e as funções *wavelet* $\psi_{a,b}(t)$,

$$W_\psi X(a,b) = \langle x(t) | \psi_{a,b} \rangle \tag{2.1}$$

onde $\psi_{a,b}(t)$ é dilatado ou contraído e deslocado em várias versões da wavelet

$$\psi_{a,b}(t) = |a|^{-\frac{1}{2}} \psi \left(\frac{t-b}{a} \right) \tag{2.2}$$

onde a e b são a escala e os parâmetros de translação, respectivamente (MALLAT, 1989; QUIROGA et al., 2004). A equação 2.1 pode ser invertida, proporcionando assim a reconstrução do sinal x(t). O WT mapeia o sinal que é representado por uma variável independente t em uma função de duas variáveis independentes a, b. Este procedimento é redundante e ineficiente para implementações algorítmicas; portanto, o WT geralmente é definido em escalas discretas a e discretas vezes, escolhendo o conjunto de parâmetros $\{a_j = 2^{-j}; b_{j,k} = 2^{-j}k\}$ com j e k inteiros. As versões contraídas da função *wavelet* combinam os componentes de alta freqüência, enquanto as versões dilatadas correspondem aos componentes de baixa freqüência. Então, ao correlacionar o sinal original com funções *wavelet* de diferentes tamanhos, podemos obter detalhes do sinal em várias escalas. Essas correlações com as diferentes funções *wavelet* podem ser organizadas em um esquema hierárquico denominado decomposição multi-resolução (MALLAT, 1989). O algoritmo de decomposição multi-resolução separa o sinal em detalhes em diferentes escalas. Neste estudo, implementamos uma decomposição de quatro níveis usando ondas de Haar, que são funções quadradas redimensionadas. As wavelets de Haar foram escolhidas devido ao seu suporte compacto e ortogonalidade, o que permite que as características discriminativas dos *spikes* sejam expressas com alguns coeficientes de *wavelet* e sem pressupostos a priori nas formas de spike.

2.2.2 Agrupamento Superparamagnético

O método de agrupamento superparamagnético do modelo de Potts é uma técnica para agrupamento de dados baseada nas propriedades físicas de um sistema que exibe comportamento magnético em função de um parâmetro conhecido como "temperatura". Esta técnica foi proposta por Blatt, Wiseman Domany (BLATT et al., 1996; QUIROGA et al., 2004).

O método calcula de maneira iterativa simulações entre cada *spike* e seus vizinhos mais próximos, neste trabalho foram utilizados os 11 vizinhos mais próximos (QUIROGA et al., 2004). O método é implementado como uma iteração de Monte Carlo de um modelo de Potts. O modelo de Potts é uma generalização do modelo de Ising em que, em vez de utilizar apenas dois estados, existem q estados diferentes por partícula.

Os resultados de agrupamento dependem principalmente da temperatura e são robustos para pequenas mudanças nos outros parâmetros (BLATT et al., 1996). As altas temperaturas correspondem a uma baixa probabilidade de mudar o estado dos pontos vizinhos.

O comportamento do sistema é uma analogia física a alguns materiais como de um vidro de spin. Em uma temperatura relativamente alta, todas os spins estão alternando aleatoriamente, independentemente de suas interações (fase paramagnética). Em uma temperatura baixa, todo o vidro de spin muda seu estado em conjunto (fase ferromagnética). Em um certo intervalo médio de temperaturas, o sistema atinge uma fase "superparamagnética"na qual somente as rotações que estão agrupadas mudam seu estado simultaneamente. Em relação ao nosso problema de agrupamento, a baixas temperaturas, todos os pontos mudarão seu estado em conjunto e, portanto, serão considerados como um único cluster; Em altas temperaturas, muitos pontos mudarão seu estado independentemente um do outro, dividindo os dados em vários clusters com apenas alguns pontos em cada um; e para as temperaturas correspondentes a fase superparamagnética, apenas os pontos agrupados irão mudar seu estado simultaneamente.

2.3 Técnica OSort

O OSort é um método de agrupamento on-line baseado na distância euclidiana entre os centros de agrupamentos estimados e os picos pré-branqueados (RUTISHAUSER et al., 2006). A técnica é uma implementação de um algoritmo de agrupamento de *spike* on-line baseado em modelo não supervisionado. A sua implementação foi realizada em MATLAB (RUTISHAUSER et al., 2006) e pode-se observar na Figura 2.2 a interface gráfica para configurações dos parâmetros. Na Tabela 2.2 pode-se observar os principais parâmetros de entrada da técnica.

Tabela 2.2 - Principais parâmetros do algoritmo OSort

minNrSpikes	Tamanho mínimo de um cluster (o cluster será excluído se o número de picos contidos for menor que esse valor)
correctionFactorThreshold	Valor que corrige uma estimativa de ruído de sinal usada como um limite de cluster

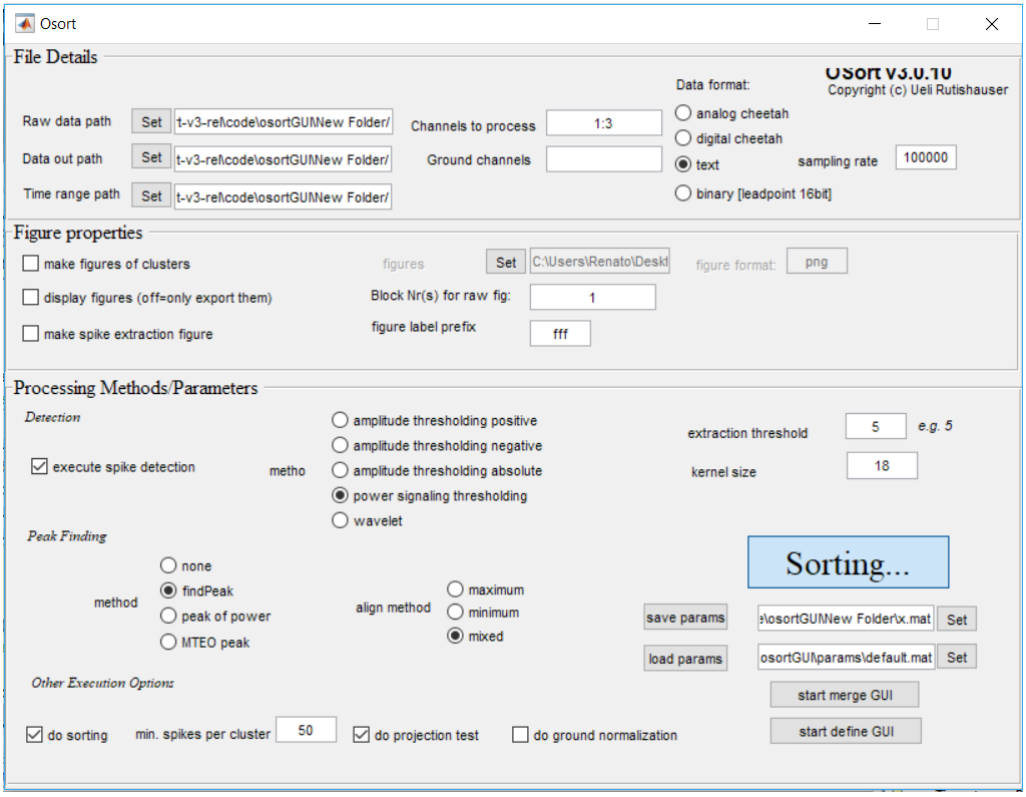

Figura 2.2 - Interface gráfica do OSort (RUTISHAUSER et al., 2006).

As subseções a seguir irão descrever as etapas de detecção e de agrupamento dessa técnica.

2.3.1 Detecção de *spikes*

A primeira etapa da detecção é o cálculo de um limiar de deteção, onde o sinal pré-processado $y_{DP}(n)$ utiliza o DPJ (Desvio padrão janelado) do sinal, sendo obtido como

$$y_{DP}(n) = \left\{ \frac{1}{M} \sum_{i=1}^{M} (x(n-i) - x_m(n))^2 \right\}^{1/2} \tag{2.3}$$

em que

$$x_m(n) = \frac{1}{M} \sum_{i=1}^{M} x(n-i), \tag{2.4}$$

é uma média-móvel de comprimento $M = 80$ para o caso que estamos considerando de taxa de amostragem de 100 kHz. O limiar de detecção define a separação de um *spike* do ruído e é calculado como cinco vezes o desvio padrão do sinal $y_{DP}(n)$ (CSICSVARI et al., 1998; RUTISHAUSER et al., 2006). Esse limiar é aplicado ao sinal e são consideradas amostras com *spike* aquelas que tiverem valores superiores ao

limiar.

2.3.2 Algoritmo de agrupamento OSort

A estimativa do número de neurônios presentes, bem como a atribuição de cada pico a um neurônio, é baseada em uma métrica de distância entre dois picos (RUTISHAUSER et al., 2006). Com base nesta distância, é utilizado um limiar para decidir quantos neurônios estão presentes e atribuir cada *spike* exclusivamente a um agrupamento de neurônios ou a um agrupamento de ruído se não é rotulado. O limite é calculado a partir das propriedades de ruído do sinal e é igual ao quadrado do desvio padrão médio do sinal, calculado com uma janela deslizante. A principal vantagem do OSort em relação aos seus concorrentes é que ele pode ser usado on-line, permitindo a agrupamento de *spikes* em tempo real durante um experimento (RUTISHAUSER et al., 2006).

No início, atribui-se o primeiro spike ao seu próprio agrupamento. Então, calcula-se a distância euclidiana entre o próximo spike e cada centróide do agrupamento. Se a menor distância for menor do que o limite, atribui-se o spike ao agrupamento mais próximo e é recalculada a média do agrupamento usando os últimos spikes mais recentes. Caso contrário, se inicia um novo agrupamento. Verificam-se as distâncias entre cada agrupamento e qualquer outro agrupamento. Se alguma distância estiver abaixo do limite de agrupamento, os dois agrupamentos são fundidos e se recalcula sua média. Pode-se observar na Figura 2.3 a sequência de atividades realizadas pela técnica OSort.

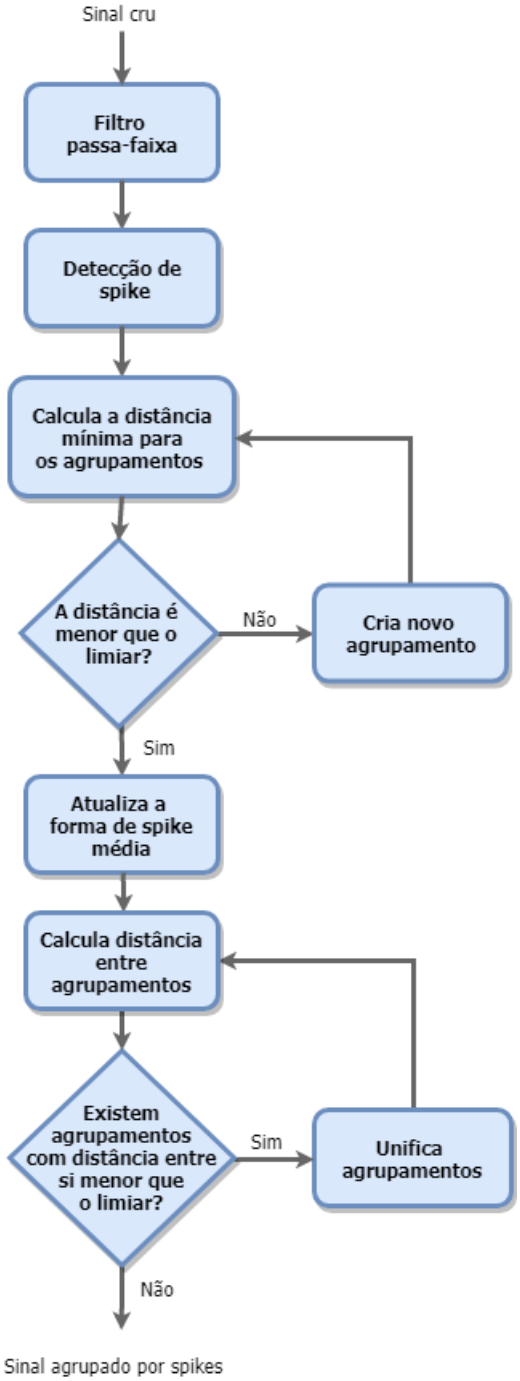

Figura 2.3 - Fluxograma do algoritmo do OSort (RUTISHAUSER et al., 2006).

No capítulo a seguir é apresentada a técnica proposta (CASTELLANI, 2014) para avaliação neste trabalho que utiliza os estudos de detecção de sinal, análise de componentes principais (PCA) e o algoritmo de agrupamento *k-means*.

3 Técnica proposta

A técnica proposta neste trabalho utiliza em sua primeira etapa os métodos de detecção estudados em Castellani (CASTELLANI, 2014), faz uma redução da dimensionalidade por meio de Análise de Componentes Principais (PCA - *Principal Component Analysis*) (ADAMOS et al., 2008; GIBSON et al., 2012; REY et al., 2015) e agrupamento com *k-means* (LEWICKI, 1998; GIBSON et al., 2012; REY et al., 2015).

3.1 Detecção de *spikes*

Muitas implementações de detecção encontradas na literatura utilizam discriminadores de amplitude, veja por exemplo (QUIROGA; PANZERI, 2009; RUTISHAUSER et al., 2006; SEMMAOUI et al., 2012; GIBSON et al., 2012; MUKHOPADHYAY; RAY, 1998). Estes discriminadores são baseados em limiares de detecção obtidos a partir de alguma propriedade estatística da série. Estipula-se um limiar de amplitude e todos os pontos da série que ultrapassem este limiar são candidatos a *spike*.

Os métodos de detecção abordados neste trabalho são baseados na aplicação de um limiar e as etapas estão resumidas no diagrama de blocos da Figura 3.1. Primeiramente, aplica-se um pré-processamento à série temporal $x(n)$ de forma a realçar os *spikes* em relação ao ruído, o que resulta no sinal $y(n)$. A seguir, utiliza-se um limiar L para se decidir sobre a presença ou não de um *spike* a cada amostra do sinal. Usualmente o valor de L é obtido a partir de um trecho da série temporal e depois aplicado a esse trecho. O valor de L é muitas vezes calculado utilizando-se múltiplos da média, do desvio padrão ou da mediana da série temporal pré-processada.

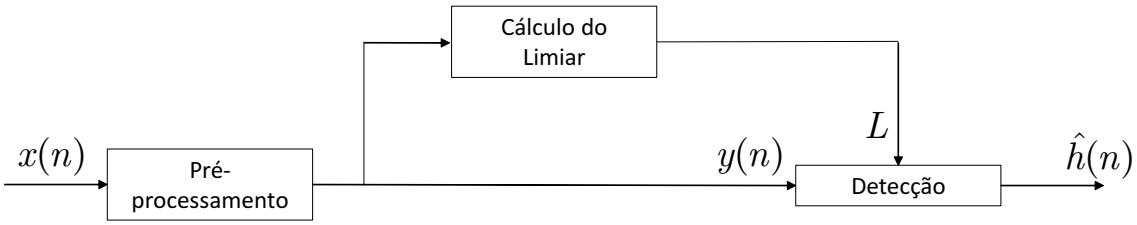

Figura 3.1 - Fluxograma das etapas dos métodos de detecção baseados na aplicação de um limiar.

A estimativa do instante de disparo pode ser feita considerando-se janelas de W_d amostras, sendo W_d aproximadamente a duração média do lóbulo principal de um *spike*. Neste trabalho, que tem a taxa de amostragem de 100 kHz, o valor estipulado foi de $W_d = 100$. Caso existam amostras que ultrapassem o limiar L, escolhe-se a

amostra de maior amplitude em cada janela W_d como instante de disparo de um só *spike*. Em outras palavras, considera-se como *spike* diferente, aquele com intervalo de ocorrência mínima de picos de, ao menos, W_d amostras. Esses picos definem então o instante de ocorrência de um *spike*. Esse procedimento resulta em um trem de impulsos $\hat{h}(n)$. Os impulsos desse trem $\hat{h}(n)$ localizam-se nos instantes em que os *spikes* foram detectados.

O que distingue uma técnica de detecção da outra é como o sinal pré-processado é obtido e como o limiar é calculado. De forma geral, o limiar aplicado é dado por (KAY, 1998)

$$L = \kappa\varepsilon, \tag{3.1}$$

sendo κ um fator escolhido de forma conveniente e ε uma estatística obtida a partir da sequência pré-processada $y(n)$. Algumas das metodologias para se obter $y(n)$ e L mais comumente empregadas são descritas a seguir.

Os métodos aqui apresentados para realçar o sinal em relação ao ruído são baseados no valor absoluto do sinal (VA) (QUIROGA et al., 2004), no Desvio Padrão Janelado (DPJ) (RUTISHAUSER et al., 2006) e no operador não-linear de energia (NEO - *Non-linear Energy Operator*) (MUKHOPADHYAY; RAY, 1998) e suas variações SNEO (NEO Suavizado) e MNEO (NEO Multiresolução).

3.1.1 Valor absoluto do sinal

Neste método, utiliza-se como pré-processamento simplesmente o valor absoluto do sinal (QUIROGA et al., 2004), ou seja,

$$y_{ABS}(n) = |x(n)|\,. \tag{3.2}$$

O método do valor absoluto do sinal foi amplamente utilizado por Quiroga et al. (QUIROGA et al., 2004). Possui um baixo custo computacional por ser muito simples, mas quando a SNR é baixa pode não detectar adequadamente os *spikes* (SHALCHYAN et al., 2012). Apesar disso, existe uma grande quantidade de artigos posteriores a (QUIROGA et al., 2004) que tratam do *spike sorting* e usam este método para realizar a detecção (RUTISHAUSER et al., 2006; GIBSON et al., 2010).

3.1.2 Desvio Padrão Janelado (DPJ)

Uma forma alternativa de pré-processamento foi proposto por Rutishauser et al. em (RUTISHAUSER et al., 2006). O sinal pré-processado, aqui definido como $y_{DP}(n)$, usa

o DPJ do sinal, sendo obtido como

$$y_{DP}(n) = \left\{ \frac{1}{M} \sum_{i=1}^{M} (x(n-i) - x_m(n))^2 \right\}^{1/2} \tag{3.3}$$

em que

$$x_m(n) = \frac{1}{M} \sum_{i=1}^{M} x(n-i), \tag{3.4}$$

é uma média-móvel de comprimento M.

Essa técnica tem o objetivo de estimar a energia localizada para encontrar um *spike*.

3.1.3 Operador Não-linear de Energia (NEO)

O Operador Não-linear de Energia (NEO - *Non-linear Energy Operator*), originalmente descrito em (KAISER, 1990), mensura não apenas o aumento da potência instantânea do sinal em relação à potência média do ruído, mas também, o quão localizado no tempo está esse acréscimo. É baseado no cálculo da energia de um movimento harmônico simples de uma massa, que não depende somente da amplitude, mas sim da amplitude e da frequência de oscilação. O NEO é uma espécie de analisador de tempo-frequência e tem sido usado em muitas aplicações, tais como processamento de voz, processamento de imagem, e demodulação de amplitude e frequência modulada (MARAGOS et al., 1993; CHOI et al., 2006; MUKHOPADHYAY; RAY, 1998). O método foi originalmente proposto no contexto de detecção de *spikes* em (MUKHOPADHYAY; RAY, 1998; KIM; KIM, 2003). O pré-processamento em tempo discreto, aqui definido como $y_{NEO}(n)$, é feito usando a seguinte expressão

$$y_{NEO}(n) = x^2(n) - x(n+\Delta)x(n-\Delta), \tag{3.5}$$

sendo Δ um atraso constante (GIBSON et al., 2012). Nota-se que o valor de $y_{NEO}(n)$ é tanto maior, quanto maior é $x^2(n)$ em relação a $x(n+\Delta)$ e $x(n-\Delta)$. Em outras palavras, o valor de $y_{NEO}(n)$ é tanto maior quando maior é a potência instantânea e a frequência do sinal $x(n)$. Como um *spike* é caracterizado por altas frequências localizadas e um aumento na potência instantânea, este método tem uma vantagem clara sobre métodos que preocupam-se apenas com um aumento na potência instantânea do sinal sem relacioná-lo com a frequência.

3.1.4 NEO Suavizado (SNEO)

Ao calcular o NEO, utiliza-se a diferença da amostra $x(n)$ com o produto $x(n-\Delta)$ e $x(n+\Delta)$, consequentemente pode-se amplificar o ruído, principalmente em baixas SNRs. Para resolver esse problema foi proposto em (SEMMAOUI et al., 2012) utilizar-se um filtro passa-baixas para suavizar o sinal do NEO. O sinal de pré-ênfase com o NEO suavizado, aqui denotado como $y_{SNEO}(n)$, é calculado como

$$y_{SNEO}(n) = w(n) * y_{NEO}(n), \tag{3.6}$$

sendo $w(n)$ a resposta ao impulso de um filtro Barlett ou de Hamming passa-baixas de ordem M (SEMMAOUI et al., 2012; MUKHOPADHYAY; RAY, 1998; CHOI; KIM, 2002; CHOI et al., 2006) e $y_{NEO}(n)$ dado por (3.5).

3.1.5 NEO com Multiresolução (MNEO)

Como uma forma de tornar o SNEO menos sensível à escolha do atraso Δ, foi proposto em (CHOI; KIM, 2002; CHOI et al., 2006) um pré-processamento baseado em uma composição de SNEOs com k diferentes atrasos Δ_k, $1 \leq k \leq N_k$. Em outras palavras, o pré-processamento do NEO com Multiresolução, aqui denotado como $y_{\text{MNEO}}(n)$, é calculado como

$$y_{\text{MNEO}}(n) = \max_k y_{\text{SNEO}_k}(n). \tag{3.7}$$

Cada $y_{\text{SNEO}_k}(n)$ é calculado com um atraso Δ_k e um filtro de ordem M_k diferente.

O método MNEO tende a apresentar resultados melhores que os outros métodos baseados em NEO se os atrasos forem escolhidos corretamente, já que é formado por uma composição de SNEOs.

3.1.6 Estatística e Limiar

A partir do sinal pré-processado, obtido com um dos métodos descritos nas subseções anteriores, utiliza-se uma estatística ε e (3.1) para se obter o limiar a ser aplicado a ele. As estatísticas mais utilizadas na literatura são:

- Mediana: a estatística ε é tomada proporcional à mediana temporal de $y(n)$. Especificamente,

$$\varepsilon = \text{mediana}\left(y(n)\right). \tag{3.8}$$

- Desvio-padrão: a estatística ε é o desvio-padrão temporal de $y(n)$, ou seja,

$$\varepsilon = \text{desvio-padrão}(y(n)). \tag{3.9}$$

- Média: a estatística ε é a média temporal de $y(n)$, ou seja,

$$\varepsilon = \text{média}(y(n)). \tag{3.10}$$

A ideia da detecção por limiar é que quando um *spike* está presente, ele altera significativamente o valor instantâneo de $y(n)$ em relação à estatística ε do ruído de fundo. Dessa forma, o *spike* é detectado quando $y(n)$ ultrapassa um limiar L.

Na Tabela 2.1 está descrito quais as técnicas estatísticas ε utilizadas para estimação do limiar de detecção e o fator de limiar κ na aplicação dessas técnicas de detecção que foram avaliadas neste trabalho pelas simulações de sinais provenientes do gerador de sinais intracranianos extracelulares. Os valores utilizadas foram definidos a partir de estudos realizados por Castellani em (CASTELLANI, 2014).

Tabela 3.1 - Estatísticas e fatores utilizados para o cálculo do limiar.

Técnica	ε	κ
Valor absoluto	desvio-padrão	5.7
DPJ	média	1.6
NEO	desvio-padrão	5.8
SNEO	desvio-padrão	3.6
MNEO	desvio-padrão	3.4

Após realizada a detecção, a próxima etapa é a redução de dimensionalidade e extração de características. Na técnica proposta é utilizado o algoritmo Análise de Componentes Principais (PCA) e que será descrito a seguir.

3.2 Análise de Componentes Principais

A PCA é baseada em técnicas de fatoração desenvolvidas em álgebra linear. Fatoração é comumente usada para diagonalizar uma matriz, de modo que sua inversa seja calculada facilmente. O objetivo do PCA é realizar uma transformação para evidenciar características estatísticas dos dados. O PCA resulta em novo um espaço ortogonal no qual os maiores autovalores correspondem respectivamente aos autovetores com maior dispersão. Encontrar novos 'eixos' que evidenciam as diferenças

entre os dados pode ser interessante para algumas aplicações como compressão de dados com baixa perda de informação e evidenciar diferenças para uma melhor agrupamento desses dados.

O objetivo do PCA é encontrar um subespaço contendo como bases ortonormais, os vetores que definem as principais direções de distribuição dos dados no espaço original otimizando o critério de correlação entre os dados, ou seja, o subespaço das principais componentes minimiza a correlação cruzada entre as amostras de dados (PEARSON, 1901).

De modo a organizar as informações para aplicação da técnica, para cada *spike* I_i, deve-se montar um vetor v em que cada posição desse vetor contenha dados que sejam correspondentes as mesmas tipos de informações para todos *spike* I_i por isso é aplicada um alinhamento de acordo com o valor absoluto máximo χ_m. Neste trabalho são $n = 200$ amostras por *spike*. Por exemplo, ao se montar o vetor que corresponde a um *spike*, dada a amostra central alinhada de pico χ_m, são utilizadas as $n/2$ amostras anteriores até as $n/2$ amostras posteriores ao pico. De maneira geral para formato, o vetor v pode ser expresso por:

$$v = [\chi_{m-n}, \chi_{m-n+1}, \chi_{m-n+2}, ..., \chi_m, ..., \chi_{m+n-1}, \chi_{m+n}], \tag{3.11}$$

onde n é o número de amostras do *spike* e χ_i corresponde a essas amostras..

O próximo passo é a organização dos vetores em uma matriz A em que cada vetor v seja uma linha. Pode-se observar a formação da matriz

$$A = \begin{bmatrix} v_1 \\ v_2 \\ ... \\ v_4 \\ v_N \end{bmatrix} \tag{3.12}$$

Obtém-se o vetor de *spike* médio μ, em que cada posição é o resultado da média da coluna correspondente da matriz A.

A covariância mede a dependência linear entre duas variáveis. Calculando a covariância pode-se determinar se há relação entre os dois conjuntos de dados. A covariância

de A é calculada por

$$\Sigma_A = \frac{(A_i - \mu)(A_i - \mu)^T}{N - 1}.$$

Calcula-se os autovalores e autovetores de Σ_A com a finalidade de encontrar um espaço linear em que dispersão entre os elementos de A seja maximalizada. A matriz de autovalores D e a matriz de autovetores Φ deve ser reordenada de maneira que os autovalores sejam colocados em ordem crescente e seus respectivos autovetores devem ser reordenados correspondentemente aos autovalores. Os maiores autovalores correspondem aos espaços com maiores dispersões entre os elementos e portanto com mais informações relevantes, já que o objetivo é diferenciar e agrupar esses elementos. Portanto, os autovetores com autovalores correspondentes de menores valores contém pouca informação relevante por ser portador de uma redundância elevada nos dados no subespaço que provê. Uma prática comum é desconsiderar esses subespaços com redundância para diminuir o custo computacional, sem perda significativa de informação.

Se considerarmos todos os autovetores compondo a matriz de autovetores Φ, podemos expressar qualquer vetor v por

$$v = \mu + \Phi b \tag{3.13}$$

onde b é um vetor de parâmetros dado por

$$b = \Phi^T(v - \mu) \tag{3.14}$$

Também é possível reconstruir um vetor v utilizando os autovetores correspondentes aos maiores autovalores, desta maneira recuperando tanta informação quanto se queira e tendo a possibilidade de eliminar redundância de informações, o que leva a uma redução no custo computacional.

$$v_r \approx \mu + \Phi_r b_r \tag{3.15}$$

onde b é um vetor de parâmetros dado por

$$b_r = \Phi_r^T(v - \mu) \tag{3.16}$$

Neste trabalho são utilizados os 20 autovetores com os maiores autovalores correspondentes.

3.3 Agrupamento k-means

As principais etapas do algoritmo são a definição de k (número de grupos *spikes* que representam um neurônio). Definição aleatoriamente os centroides do agrupamento k. Posteriormente atribui-se a cada *spike* ao agrupamento com o centro mais próximo (medida de distância euclidiana). Já com o novo *spike* é recalculado o centróide do agrupamento como centro desse agrupamento. O procedimento é realizado para todos os *spikes* e é repetido até que um critério de convergência seja atingido, como os agrupamentos pararem de mudar ou que o número máximo de iterações tenha sido atingido. Os passos do procedimento são detalhados no Algoritmo 3.3. O algoritmo *k-means* é baseado em uma métrica de distância. Um benefício de usar *k-means* é que é um algoritmo muito simples e rápido. No entanto, uma grande desvantagem para este algoritmo é que ele não é supervisionado, pois exige que o usuário defina o número de agrupamentos k.

Algoritmo 1: k-MEANS
Entrada: k

```
início
    Escolha k spikes aleatoriamente para serem centróides dos agrupamentos
    para todos os spikes restantes faça
        Calcule a distância entre o spike e os centróides
        Adicione o spike ao agrupamento que possuir a menor distância
        Recalcule o centróide do agrupamento
    fim
    para todos os k agrupamentos faça
        Calcule a soma de quadrados residual
    fim
    Enquanto Número de interações = i_max ou Não ocorra mudança de grupos
     entre spikes faca
        para todos os n spikes faça
            Mova o spike para os outros agrupamentos
            Recalcule a soma de quadrados residual
            se soma dos quadrados residual diminuiu então
                O objeto passa a fazer parte do grupo que produzir maior ganho
                Recalcule a soma de quadrados residual dos grupos alterados
            fim
        fim
    fim-enquanto
fim
```

4 Resultados e discussão

O *spike sorting* tem uma série de algoritmos propostos que foram publicados ao longo dos anos, mas ainda não existe uma solução universalmente aceita. Esse trabalho pretende realizar uma comparação entre as implementações do processo de separação e agrupamento dos *spikes* de um sinal por neurônio de origem. O gerador de sinais *NeuroCube* descrito em 2.1 foi utilizado para que a comparação seja realizada de maneira que exista um gabarito de referência para ser utilizado como avaliador, já que nos sinais reais esse processo é dificultado.

As técnicas de Rutishauser et al. (RUTISHAUSER et al., 2006) apresentada na Seção 2.3, Quiroga et al. (QUIROGA et al., 2004) apresentada na Seção 2.2 e a proposta por esse trabalho com PCA e *k-means* apresentada na Seção 3 com diferentes métodos de cálculo de detecção propostos por Castellani (CASTELLANI, 2014) são comparadas utilizando o gerador de sinais bioinspirados NeuroCube (CAMUÑAS-MESA; QUIROGA, 2013).

4.1 Avaliação

Utilizando o gerador de sinais neuronais simulados NeuroCube (CAMUÑAS-MESA; QUIROGA, 2013) foram gerados 600 sinais intracranianos extracelulares de 30 segundos cada com uma taxa de amostragem de 100 kHz. Todos os parâmetros propostos no trabalho original (CAMUÑAS-MESA; QUIROGA, 2013) são seguidos com exceção da taxa de disparo e a frequência de amostragem. A frequência de amostragem segue o trabalho de Castellani (CASTELLANI, 2014) de detecção de disparos que fez o levantamento de parâmetros ideais de detecção nas técnicas utilizadas. A taxa de disparo pode variar de acordo com a região nervosa e foram simulados dois cenários de modo que se possa compará-los.

Os sinais foram gerados com dois valores diferentes de taxa de neurônios ativos (*rate of active neurons*) e resultaram em dois grupos distintos de 300 sinais cada com quantidade de neurônios próximos ativos diferentes. O Grupo 1 teve disparos gerados por 8 neurônios em média próximos ao eletrodo, já o Grupo 2 teve disparos neuronais gerados por 30 neurônios em média. O número de neurônios ativos por grupo foi escolhido baseado nos resultados da literatura que mostram que os agrupadores de *spikes* conseguem ter boas taxas de acerto com até 7 ou 8 neurônios ativos no máximo (PEDREIRA et al., 2012; CAMUÑAS-MESA; QUIROGA, 2013). Uma representação gráfica do NeuroCube pode ser vista na Figura 4.1. Foram utilizados para gerar os sinais um cubo de 1 mm^3 com densidade de 300 mil de células de

neurônios por mm^3.

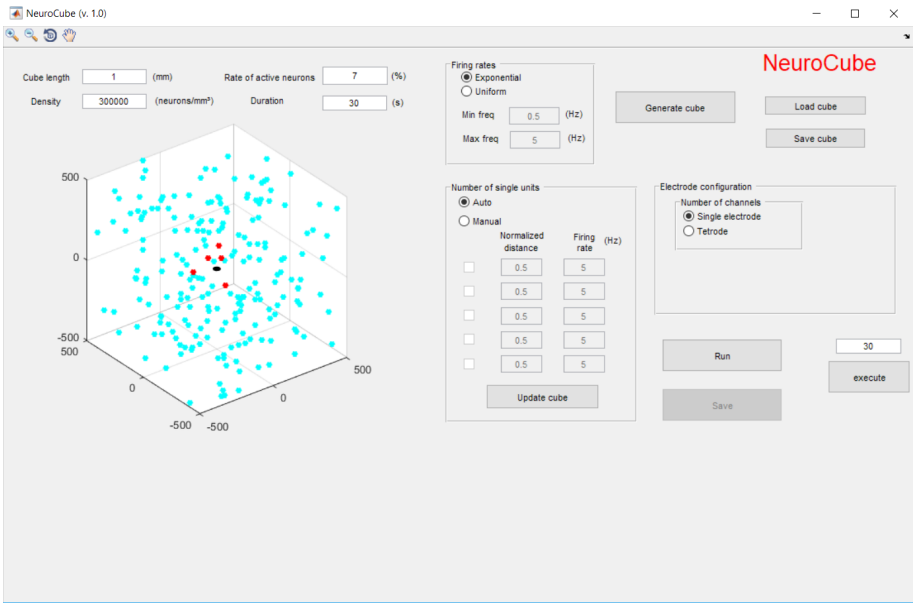

Figura 4.1 - Interface gráfica do NeuroCube (CAMUÑAS-MESA; QUIROGA, 2013).

A taxa de detecção foi avaliada de acordo com o gabarito do sinal obtido pelo próprio gerador de sinais Neurocube respeitando o parâmetro de detecção $W_d = 200$ amostras em que se fosse calculado pela técnica um disparo e o gabarito tivesse um *spike* a menos de W_d seria considerado como uma detecção correta. Caso não fosse encontrado um *spike* nesse intervalo, seria considerado uma detecção incorreta ou falha de detecção.

Os resultados são apresentados em termos da probabilidade de detecção P_D e a probabilidade de acerto de agrupamentos P_{Ag}. Essas probabilidades são definidas em termos de

$N_{Detectados}$ = **Número de spikes detectados dentre os spikes existentes no gabarito,**
$N_{Agrupados}$ = **Número de agrupados corretamente dentre os detectados e existentes,**
$N_{Existentes}$ = **Número de spikes existentes no gabarito.**

A probabilidade de detecção P_D é a relação entre o número de *spikes* detectados e que realmente existiam no gabarito pelo número de *spikes* que existiam no gabarito.

Matematicamente

$$P_D = \frac{N_{Detectados}}{N_{Existentes}}. \tag{4.1}$$

A probabilidade de acerto de agrupamentos P_{Ag} é a relação entre o número de agrupamentos corretos dado que a detecção foi correta de um *spikes* existente. Já que um disparo só é considerado para etapa de agrupamento caso tenha sido detectado na etapa de detecção. Matematicamente

$$P_{Ag} = \frac{N_{Agrupados}}{N_{Detectados}}. \tag{4.2}$$

Portanto a probabilidade de acerto global (P_G) calculada por

$$P_G = P_D \cdot P_{Ag} = \frac{N_{Agrupados}}{N_{Existentes}}. \tag{4.3}$$

Cada técnica tem características específicas e um modo de agrupar diferentes. Pode criar mais ou menos grupos do que o número de grupos que foi gerado pelo simulador de sinais. Para contornar esse problema e avaliar as técnicas de agrupamento de maneira igual, os resultados de agrupamentos foram pós processados. Caso os agrupamentos calculados tivessem mais da metade dos seus *spikes* pertencentes a um agrupamento de disparos de um neurônio ao se consultar o gabarito, esse agrupamento seria considerado um *spike* desse agrupamento, ou seja, seria considerado um disparo proveniente desse neurônio.

4.2 Resultados

Para obter-se o resultado do Grupo 1 e 2, respectivamente na Tabela 3.1 e 3.2, foram utilizados os parâmetros descritos no Capítulo 2. Para as técnicas Valor Absoluto (VA), desvio padrão janelado (DPJ), NEO, SNEO e MNEO os parâmetros utilizados para detecção são os da Tabela 2.1. Os algoritmos OSort (RUTISHAUSER et al., 2006) e WaveClus (QUIROGA et al., 2004) respeitam os parâmetros e as técnicas de detecção originais apresentadas em seus artigos de apresentação. As simulações foram realizadas utilizando MATLAB.

Os resultados do Grupo 1 na Tabela 3.1 com 8 neurônios em média disparando foram relativamente bons e de acordo com a literatura onde há boas taxas de acerto quanto há até 7 ou 8 neurônios ativos (PEDREIRA et al., 2012; CAMUÑAS-MESA; QUIROGA, 2013). Resultados melhores que esses na literatura foram obtidos, mas em situações

em que eram agrupados até 4 neurônios ativos em um mesmo sinal (PEDREIRA et al., 2012).

Tabela 4.1 - Grupo 1 - Resultados de taxa de detecção e taxa de acerto de agrupamentos para as técnicas avaliadas.

Técnica	Probabilidade de detecção (P_D)	Probabilidade de acerto de agrupamentos (P_{Ag})	Probabilidade de acerto (P_G)
VA, PCA e k-means	75.9%	**91.6**%	69.5%
DPJ, PCA e k-means	78.2%	84.4%	66.0%
NEO, PCA e k-means	90.9%	87.1%	71.9%
SNEO, PCA e k-means	93.4%	81.3%	75.9%
MNEO, PCA e k-means	**94.1**%	81.3%	**76.5**%
OSort (RUTISHAUSER et al., 2006)	70.1%	70.1%	49.1%
WaveClus (QUIROGA et al., 2004)	62.3%	74.1%	46.1%

A técnica que obteve a maior probabilidade de acerto geral foi da técnica que utilizou técnica de detecção MNEO com redução de dimensionalidade realizada por PCA e agrupamento por k-means com Probabilidade de acerto de 76.5%. Uma característica interessante desse resultado é que a técnica que obteve melhor resultado geral foi a que obteve melhor resultado na etapa de detecção com 94.1%, fato que havia sido sugerido pelo estudo de Castellani (CASTELLANI, 2014).

A técnica que obteve melhor desempenho na etapa de agrupamento foi a que realizou a detecção com a técnica de Valor Absoluto (VA) com redução de dimensionalidade realizada por PCA e agrupamento por k-means com com 94.6%. Pode-se observar que essa técnica não obteve um bom desempenho na detecção, ou seja, só foram agrupados disparos com menor ruído o que pode ter influenciado no bom desempenho no agrupamento. Conjectura-se que existe uma relação dos resultados etapa de detecção com o resultado da etapa de agrupamento, e que quanto maior a probabilidade de detecções corretas, menor a probabilidade de agrupamentos corretos. Novamente nota-se a importância de uma boa detecção.

Os resultados das técnicas com OSort (RUTISHAUSER et al., 2006) e WaveClus (QUIROGA et al., 2004) foram inferiores aos propostos por Castellani (CASTELLANI, 2014). Um fato a ser considerado é que os parâmetros das técnicas OSort e WaveClus originalmente não funcionam na frequência de amostragem de 100 kHz como foi realizado

nesse trabalho e o ajuste dos parâmetros pode ter tido influência nos resultados obtidos. As técnicas WaveClus e OSort tem como entrada o valor da frequência de amostragem e foram modificados para 100 kHz, entretanto parecem não ser robustos e adaptáveis na mudança desse parâmetro. Em Castellani todos os parâmetros foram calculados nessa frequência de amostragem.

Os resultados do Grupo 2 na Tabela 3.2 com 30 neurônios em média disparando foram ruins, mas de acordo com a literatura já que o número de médio de neurônios ativos 30 é mais que três vezes o valor em que se obtêm boas taxas de acerto que é até 7 ou 8 neurônios ativos (PEDREIRA et al., 2012; CAMUÑAS-MESA; QUIROGA, 2013).

Tabela 4.2 - Grupo 2 - Resultados de taxa de detecção e taxa de acerto de agrupamentos para as técnicas avaliadas. Com simulações com 30 neurônios próximos ativos.

Técnica	Probabilidade de detecção (P_D)	Probabilidade de acerto de agrupamentos (P_{Ag})	Probabilidade de acerto (P_G)
VA, PCA e k-means	38.5%	**66.4**%	25.5%
DPJ, PCA e k-means	58.7	41.0%	24.0%
NEO, PCA e k-means	65.0%	49.5%	**32.1**%
SNEO, PCA e k-means	68.0%	39.5%	26.8%
MNEO, PCA e k-means	**68.4**%	38.0%	25.9%
OSort (RUTISHAUSER et al., 2006)	38.0%	49.7%	18.8%
WaveClus (QUIROGA et al., 2004)	62.0%	33.7%	20.8%

A técnica que obteve a maior probabilidade de acerto geral foi da técnica que utilizou técnica de detecção NEO com redução de dimensionalidade realizada por PCA e agrupamento por k-means com 32.1%. Essa técnica não obteve melhor resultado nem em Probabilidade de detecção nem em Probabilidade de acerto de agrupamento. O método foi melhor devido ao mau desempenho dos outros que obtiveram melhores resultados em uma das etapas e não apresentarem bom desempenho na outra etapa. O que reforça a hipótese novamente do compromisso entre a Probabilidade de detecção e a Probabilidade de acerto de agrupamento.

A técnica que obteve melhor desempenho na etapa de detecção foi a que realizou a detecção com a técnica de MNEO com redução de dimensionalidade realizada por

PCA e agrupamento por k-means com 68.4%. Essa técnica obteve uma Probabilidade muito baixa de acerto na etapa de agrupamentos, apenas com 38.0%, conjectura-se que havia muito ruído no sinal dos detectados. A técnica que obteve melhor desempenho na etapa de agrupamento foi a que realizou a detecção com a técnica de Valor Absoluto (VA) com redução de dimensionalidade realizada por PCA e agrupamento por k-means com 66.4%. Essa técnica obteve uma Probabilidade muito baixa de acerto na etapa de detecção, apenas com 38.5%, conjectura-se que por ter realizado uma detecção com taxas tão baixas eliminou muitas detecções de ruído e assim pôde obter uma maior taxa de acerto de agrupamentos.

Como a comparação com WaveClus e OSort foi realizada somente para comparação com as técnicas propostas, essas poderiam ter resultados melhores se melhor parametrizadas, mas não era o objetivo do trabalho.

Em todos os casos analisados pode-se perceber a importância da etapa de detecção para o sucesso das demais etapas. Portanto métodos mais eficientes de registro do eletrodo que permitam um tratamento do ruído, permitiriam melhores detecções e consequentemente melhores resultados para o sistema.

5 Conclusão

O problema de detectar os *spikes* em um sinal e agrupá-los de acordo com o neurônio que o emitiu é chamado de *spike sorting* (QUIROGA; PANZERI, 2009; DAYAN; ABBOTT, 2005; RUTISHAUSER et al., 2006; GIBSON et al., 2012). Ainda não existe uma solução universalmente aceita, apesar de existirem diversos algoritmos para este processo (QUIROGA et al., 2004; RUTISHAUSER et al., 2006; GIBSON et al., 2012; REY et al., 2015). O *spike sorting* é realizado a partir do sinal proveniente de um eletrodo e tem como etapas principais: a detecção, a parametrização e a agrupamento.

No presente trabalho foram estudados métodos para o agrupamento de *spikes* em sinais intracranianos e extracelulares. Foram avaliadas as técnicas com detecção otimizada conforme o estudo aprofundado na parte da detecção dos *spikes* de Castellani (CASTELLANI, 2014) e o resultado foi comparado com os mais comuns métodos utilizadas no momento de Quiroga et al. (QUIROGA et al., 2004) e de Rutishauser et al. (RUTISHAUSER et al., 2006).

Cada uma das etapas do *spike sorting* apresenta o seu desafio. Porém, se a etapa da detecção não é feita de forma adequada, todo o processo de *spike sorting* pode resultar em altas taxas de erros. O que torna o problema da detecção difícil de ser tratado é que não se conhece exatamente a forma de onda do sinal a ser detectado e os instantes de disparo não são conhecidos. Cada neurônio possui uma constituição biofísica e, portanto, apresenta formato de disparos diferentes, que também podem variar no tempo. Além disso, em medições de sinais intracranianos, *a priori* tem-se apenas uma estimativa do número de neurônios que estão sendo captados pelos eletrodos e nenhuma informação sobre a posição dos neurônios em relação ao eletrodo. Esses fatos não permitem uma proposta de solução com um banco de filtros casados fixos, como é usual em Telecomunicações (OPPENHEIM, 1999). Também como não há controle sobre a emissão de disparos, não há, em princípio, como se treinar o sistema de detecção para uma correção adaptativa.

Os resultados da literatura para separação de disparos por neurônio tem sido muito inferiores aos valores teóricos baseados em considerações biofísicas e anatômicas (REY et al., 2015). Sabe-se que pode-se registrar atividades a uma distância de 50 micrômetros e que nesse raio deveriam ser registrados disparos de cerca de 100 neurônios, mas o número de neurônios relatados com canais únicos de gravação é geralmente inferior a dez. Há uma série de hipóteses sugeridas para explicar essa discrepância como uma lesão ao tecido pelo eletrodo, isolamento elétrico causado pelo substrato da sonda e baixa taxa de disparos de um número grande de neurônios.

Outro fator que contribuem pro número baixo de neurônios registrados é a limitação dos atuais algoritmos de *spike sorting* (REY et al., 2015).

Foi relatado na literatura por Pedreira (PEDREIRA et al., 2012) uma avaliação da capacidade desses algoritmos e o resultado foi quase perfeito para 4 neurônios, foram encontrados bons resultados até 8 neurônios. Os neurônios com menores taxas de disparos foram mais frequentemente perdidos em comparação aos com taxas de disparo maiores.

Nesse trabalho os melhores resultados foram obtidos quando o sinal gerado pelo NeuroCube (CAMUÑAS-MESA; QUIROGA, 2013) tinha em média 8 neurônios emissores com probabilidade de acerto de 76.5% com a melhor técnica que foi a combinação do uso de MNEO com PCA e k-*means*. Quando foi realizado um teste com 30 neurônios emissores em média os resultados foram de no máximo 32.1% que não é um resultado satisfatório, entretanto é coerente com resultados na literatura.

Buscando-se obter melhores resultados detecção de *spikes* em meio de um sinal intracraniano extracelular são propostos na literatura uma melhoria nos eletrodos de gravação desses sinais. Uma melhor detecção e agrupamento pode ser obtida utilizando tetrodos e politrodos (REY et al., 2015). As matrizes multi-eletrodos (MEAs - *multi-electrode-arrays*) já estão começando a ser usados com até milhares locais sendo registrados por eletrodos para o registro em retinas, culturas de células e fatias de cérebro (LITKE et al., 2004; LAMBACHER et al., 2011; FREY et al., 2009).

O registro de um evento de disparo sob diferentes eletrodos realizando gravações permite o processamento digital do sinal considerando características tridimensionais do meio observando a direção de chegada (DOA - *Direction of arrival*) do disparo e assim permitindo um tratamento de ruído e agrupamento mais eficientes. A avalização de técnicas de agrupamento de *spikes* considerando registros de MEAs e seu processamento deve ser assunto de trabalhos futuros.

REFERÊNCIAS BIBLIOGRÁFICAS

ADAMOS, D. A.; KOSMIDIS, E. K.; THEOPHILIDIS, G. Performance evaluation of pca-based spike sorting algorithms. **Computer methods and programs in biomedicine**, Elsevier, v. 91, n. 3, p. 232–244, 2008. 19

ASSOCIAÇÃO BRASILEIRA DE NORMAS TÉCNICAS (ABNT). **NBR 6023**: informação e documentação: referências: elaboração. Rio de Janeiro, aug. 2002b. 24 p. ii

BERGER, T. W.; AHUJA, A.; COURELLIS, S. H.; DEADWYLER, S. A.; ERINJIPPURATH, G.; GERHARDT, G. A.; GHOLMIEH, G.; GRANACKI, J. J.; HAMPSON, R.; HSAIO, M. C. et al. Restoring lost cognitive function. **IEEE Engineering in Medicine and Biology Magazine**, IEEE, v. 24, n. 5, p. 30–44, 2005. 1

BLATT, M.; WISEMAN, S.; DOMANY, E. Superparamagnetic clustering of data. **Physical review letters**, APS, v. 76, n. 18, p. 3251, 1996. 13

CAMUÑAS-MESA, L. A.; QUIROGA, R. Q. A detailed and fast model of extracellular recordings. **Neural computation**, MIT Press, v. 25, n. 5, p. 1191–1212, 2013. ix, 6, 7, 9, 10, 27, 28, 29, 31, 34

CASTELLANI, R. A. **Um estudo sobre a detecção de spikes em sinais neuronais**. Master Thesis (Mestrado) — UFABC - Universidade Federal do ABC, 2014. ix, 3, 5, 6, 7, 17, 19, 23, 27, 30, 33

CHOI, J. H.; JUNG, H. K.; KIM, T. A new action potential detector using the mteo and its effects on spike sorting systems at low signal-to-noise ratios. **Biomedical Engineering, IEEE Transactions on**, v. 53, n. 4, p. 738 –746, april 2006. ISSN 0018-9294. 21, 22

CHOI, J. H.; KIM, T. Neural action potential detector using multi-resolution teo. **Electronics Letters**, v. 38, n. 12, p. 541 –543, jun 2002. ISSN 0013-5194. 22

CRCNS. **Collaborative Research in Computational Neuroscience - Data sharing**. 2000. Available from: <http://crcns.org/>. ix, 5

CSICSVARI, J.; HIRASE, H.; CZURKO, A.; BUZSÁKI, G. Reliability and state dependence of pyramidal cell–interneuron synapses in the hippocampus: an ensemble approach in the behaving rat. **Neuron**, Elsevier, v. 21, n. 1, p. 179–189, 1998. 15

DAYAN, P.; ABBOTT, L. F. **Theoretical neuroscience: computational and mathematical modeling of neural systems**. [S.l.]: MIT Press, 2005. ISBN 978-0-262-54185-5. 1, 2, 4, 6, 33

FREY, U.; EGERT, U.; HEER, F.; HAFIZOVIC, S.; HIERLEMANN, A. Microelectronic system for high-resolution mapping of extracellular electric fields applied to brain slices. **Biosensors and Bioelectronics**, Elsevier, v. 24, n. 7, p. 2191–2198, 2009. 34

GIBSON, S.; JUDY, J.; MARKOVIC, D. Comparison of spike-sorting algorithms for future hardware implementation. In: **Conference proceedings: Annual International Conference of the IEEE Engineering in Medicine and Biology Society. IEEE Engineering in Medicine and Biology Society**. [S.l.: s.n.], 2008. p. 5015–5020. 1

GIBSON, S.; JUDY, J.; MARKOVIC, D. Technology-aware algorithm design for neural spike detection, feature extraction, and dimensionality reduction. **Neural Systems and Rehabilitation Engineering, IEEE Transactions on**, v. 18, n. 5, p. 469 –478, oct. 2010. ISSN 1534-4320. 20

GIBSON, S.; JUDY, J. W.; MARKOVI, D. Spike sorting the first step in decoding the brain. **IEEE Signal Processing Magazine**, v. 29, n. 1, p. 124–143, JAN 2012. ISSN 1053-5888. 1, 2, 3, 5, 6, 19, 21, 33

GUYTON, A. **Tratado de fisiología médica**. [S.l.]: Interamericana, 1984. ISBN 9789682508738. 4

HENZE, D. A.; BORHEGYI, Z.; CSICSVARI, J.; MAMIYA, A.; HARRIS, K. D.; BUZSáKI, G. Intracellular features predicted by extracellular recordings in the hippocampus in vivo. **Journal of Neurophysiology**, Center for Molecular and Behavioral Neuroscience, Rutgers, The State University of New Jersey, Newark, New Jersey 07102, USA. buzsaki@axon.rutgers.edu, v. 84, n. 1, p. 390–400, jul. 2000. ISSN 0022-3077. Available from: <http://jn.physiology.org/content/84/1/390.abstract>. 5

HOCHBERG, L. R.; SERRUYA, M. D.; FRIEHS, G. M.; MUKAND, J. A.; SALEH, M.; CAPLAN, A. H.; BRANNER, A.; CHEN, D.; PENN, R. D.; DONOGHUE, J. P. Neuronal ensemble control of prosthetic devices by a human with tetraplegia. **Nature**, Nature Publishing Group, v. 442, n. 7099, p. 164, 2006. 1

KAISER, J. On a simple algorithm to calculate the 'energy' of a signal. In: **Acoustics, Speech, and Signal Processing, 1990. ICASSP-90., 1990 International Conference on**. [S.l.: s.n.], 1990. p. 381 –384 vol.1. ISSN 1520-6149. 21

KAY, S. **Fundamentals of Statistical signal processing: Detection theory**. [S.l.]: Prentice Hall PTR, 1998. (Prentice Hall Signal Processing Series, v. 2). 20

KIM, K. H.; KIM, S. J. A wavelet-based method for action potential detection from extracellular neural signal recording with low signal-to-noise ratio. **Biomedical Engineering, IEEE Transactions on**, v. 50, n. 8, p. 999 –1011, aug. 2003. ISSN 0018-9294. 21

LAMBACHER, A.; VITZTHUM, V.; ZEITLER, R.; EICKENSCHEIDT, M.; EVERSMANN, B.; THEWES, R.; FROMHERZ, P. Identifying firing mammalian neurons in networks witháhigh-resolution multi-transistor array (mta). **Applied Physics A**, Springer, v. 102, n. 1, p. 1–11, 2011. 34

LEWICKI, M. S. A review of methods for spike sorting: the detection and classification of neural action potentials. **Network: Computation in Neural Systems**, Informa UK Ltd UK, v. 9, n. 4, p. R53–R78, 1998. 1, 19

LITKE, A.; BEZAYIFF, N.; CHICHILNISKY, E.; CUNNINGHAM, W.; DABROWSKI, W.; GRILLO, A.; GRIVICH, M.; GRYBOS, P.; HOTTOWY, P.; KACHIGUINE, S. et al. What does the eye tell the brain?: Development of a system for the large-scale recording of retinal output activity. **IEEE Transactions on Nuclear Science**, IEEE, v. 51, n. 4, p. 1434–1440, 2004. 34

MALLAT, S. G. A theory for multiresolution signal decomposition: the wavelet representation. **IEEE transactions on pattern analysis and machine intelligence**, Ieee, v. 11, n. 7, p. 674–693, 1989. 13

MARAGOS, P.; KAISER, J.; QUATIERI, T. On amplitude and frequency demodulation using energy operators. **Signal Processing, IEEE Transactions on**, v. 41, n. 4, p. 1532 –1550, apr 1993. ISSN 1053-587X. 21

MARTINEZ, J.; PEDREIRA, C.; ISON, M. J.; QUIROGA, R. Q. Realistic simulation of extracellular recordings. **Journal of neuroscience methods**, Elsevier, v. 184, n. 2, p. 285–293, 2009. 9

MUKHOPADHYAY, S.; RAY, G. A new interpretation of nonlinear energy operator and its efficacy in spike detection. **Biomedical Engineering, IEEE**

Transactions on, v. 45, n. 2, p. 180 –187, feb. 1998. ISSN 0018-9294. 19, 20, 21, 22

NICOLELIS, M. A. Actions from thoughts. **Nature**, Nature Publishing Group, v. 409, n. 6818, p. 403, 2001. 1

NICOLELIS, M. A. **Methods for neural ensemble recordings**. [S.l.]: CRC press, 2007. 1

OPPENHEIM, A. V. **Discrete-time signal processing**. [S.l.]: Pearson Education India, 1999. 33

PEARSON, K. On lines and planes of closest fit to systems of points in space. **The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science**, v. 2, n. 1, p. 559–572, 1901. ISSN 1941-5982. Available from: <http://dx.doi.org/10.1080/14786440109462720>. 24

PEDREIRA, C.; MARTINEZ, J.; ISON, M. J.; QUIROGA, R. Q. How many neurons can we see with current spike sorting algorithms? **Journal of Neuroscience Methods**, v. 211, n. 1, p. 58 – 65, 2012. ISSN 0165-0270. 2, 27, 29, 30, 31, 34

QUIROGA, Q.; PANZERI, S. Extracting information from neuronal populations: information theory and decoding approaches. **Nature Reviews Neuroscience**, Nature Publishing Group, v. 10, n. 3, p. 173–185, mar. 2009. ISSN 1471-0048. Available from: <http://dx.doi.org/10.1038/nrn2578>. 1, 2, 19, 33

QUIROGA, R. Q.; NADASDY, Z.; BEN-SHAUL, Y. Unsupervised spike detection and sorting with wavelets and superparamagnetic clustering. **Neural computation**, MIT Press, v. 16, n. 8, p. 1661–1687, 2004. ix, 6, 7, 9, 11, 13, 20, 27, 29, 30, 31, 33

QUIROGA, R. Q.; REDDY, L.; KOCH, C.; FRIED, I. Decoding visual inputs from multiple neurons in the human temporal lobe. **Journal of neurophysiology**, Am Physiological Soc, v. 98, n. 4, p. 1997–2007, 2007. 10

REY, H. G.; PEDREIRA, C.; QUIROGA, R. Q. Past, present and future of spike sorting techniques. **Brain Research Bulletin**, Elsevier, v. 119, n. Pt B, p. 106, 2015. 2, 19, 33, 34

RUTISHAUSER, U.; SCHUMAN, E. M.; MAMELAK, A. N. Online detection and sorting of extracellularly recorded action potentials in human medial temporal lobe

recordings, in vivo. **Journal of neuroscience methods**, Elsevier, v. 154, n. 1-2, p. 204–224, 2006. ix, 2, 6, 7, 9, 14, 15, 16, 17, 19, 20, 27, 29, 30, 31, 33

SEMMAOUI, H.; DROLET, J.; LAKHSSASSI, A.; SAWAN, M. Setting adaptive spike detection threshold for smoothed teo based on robust statistics theory. **Biomedical Engineering, IEEE Transactions on**, v. 59, n. 2, p. 474 –482, feb. 2012. ISSN 0018-9294. 19, 22

SHALCHYAN, V.; JENSEN, W.; FARINA, D. Spike detection and clustering with unsupervised wavelet optimization in extracellular neural recordings. **Biomedical Engineering, IEEE Transactions on**, v. 59, n. 9, p. 2576 –2585, sept. 2012. ISSN 0018-9294. 20

APÊNDICE A - Código MATLAB

A.1 Código MATLAB

Código MATLAB

Algorítimo A.1 - Rotina Principal Spike Sorting

```
clear all; clc; close all
save 'temp'
resultados = cell(1,300);
for iloop=1:300
disp(['simula  o ' num2str(iloop)])
load(['\sinal_do_simulador\sim_' num2str(iloop) '.mat'])
t= 0:1/1e5:(length(data)-1)/1e5;
valores_maximos_linhas = max(abs(data'));
for c=1:length(valores_maximos_linhas)
    sim.data(c,:)=data(c,:)/valores_maximos_linhas(c);
end
clear c
clear valores_maximos_linhas
sim.len = length(data);
sim.Close_neurons       = Close_neurons;
sim.Coordinates         = Coordinates;
sim.Coordinates_close   = Coordinates_close;
sim.Neurons_id          = Neurons_id;
sim.Spiketimes          = Spiketimes;
sim.Close_Spikeshapes   = Close_Spikeshapes;
clear data
clear Close_neurons
clear Coordinates
clear Coordinates_close
clear Neurons_id
clear Spiketimes
clear Close_Spikeshapes

%% calcula m ximo local dos spikes para gabarito
par.interv = 200;
calc.temSpike = zeros(1,length(sim.data));

spikes=zeros(size(sim.Close_neurons,1),size(sim.data,1)*par.interv);
%calc.CentroAlinhado = zeros(size(sim.Close_neurons,1),1);
for linha = 1:size(sim.Close_neurons,1)
    calc.AmostraComSpike(linha) = int32(round(100*sim.Close_neurons((linha),2)));
    if(calc.AmostraComSpike(linha) <par.interv/2)
        calc.temSpike(1:calc.AmostraComSpike(linha) )=1;
    elseif(calc.AmostraComSpike(linha) +(par.interv/2)>length(calc.temSpike))
        calc.temSpike(calc.AmostraComSpike(linha) :end)=1;
    else
        inicio_spike_calc = calc.AmostraComSpike(linha) - (par.interv/2) +1;
        intervalo= inicio_spike_calc : calc.AmostraComSpike(linha) +(par.interv/2);
        [~, valorPosicaoMaximo]= max(abs(sim.data(:,intervalo))');
        calc.CentroAlinhado(linha) = inicio_spike_calc + round(mean(
            valorPosicaoMaximo));
```

```
        calc.labelOrig(linha,:) = sim.Close_neurons(linha,1);
        intervaloCentroMedio = calc.CentroAlinhado(linha) - (par.interv/2) +1:
            calc.CentroAlinhado(linha) +(par.interv/2);
        if (min(intervaloCentroMedio)>0 && max(intervaloCentroMedio)<length(
            sim.data))
            if(size(sim.data,1)>1)
                spikes(linha,:)=[ sim.data(1,intervaloCentroMedio) sim.data(2,
                    intervaloCentroMedio) sim.data(3,intervaloCentroMedio) sim.data
                    (4,intervaloCentroMedio)];
            else
                spikes(linha,:)=sim.data(1,intervaloCentroMedio);
            end
            calc.temSpike(intervaloCentroMedio)=1;
        end
    end
end
clear linha
spikeSemCalculoPossivel = find(calc.CentroAlinhado==0);
spikes(spikeSemCalculoPossivel,:)=[];
calc.CentroAlinhado(spikeSemCalculoPossivel)=[];
calc.labelOrig(spikeSemCalculoPossivel)=[];
calc.porcentagemSinalComSpike=sum(calc.temSpike)/length(calc.temSpike)*100;
disp(['porcentagem do sinal com spikes= ' num2str(calc.porcentagemSinalComSpike) '%
    '])
calc.AmostraComSpike = calc.CentroAlinhado;
%% Parametros
par.ParametroDeteccao = 150;
par.ParametroAvaliacao = 150;

%%  Valor Absoluto
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
disp('Valor Absoluto')
kappa=5.7;
[VA]=metodoVA(sim.data,calc,par,kappa);

%%  Desvio Padr o Janelado
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
disp('Desvio Padr o Janelado')
% tic
kappa=1.6;
janela=80;
[DPJ]=metodoDPJ(sim.data,calc,par,kappa,janela);
%

%%  NEO
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
disp('NEO')
atraso=25;
kappa=5.8;
[NEO]=metodoNEO(sim.data,calc,par,kappa,atraso);

%%  SNEO
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
disp('SNEO')
atraso=25;
kappa=1.17;
kappa=3.6;
```

```
[SNEO]=metodoSNEO(sim.data,calc,par,kappa,atraso);


%%  MNEO
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
disp('MNEO')
atraso=[20 25 30];
kappa=3.4;
[MNEO]=metodoMNEO(sim.data,calc,par,kappa,atraso);

%%
close all
numeroDeNeuronios{iloop} = length(unique(sim.Close_neurons(:,1)))

 ['PorcNaoDetectados PorcDeteccoesCorretas  PorcAcertoAgrupamento']
  resultados{iloop}=[  VA.PorcNaoDetectados VA.PorcDeteccoesCorretas
      VA.PorcAcertoAgrupamento;
     DPJ.PorcNaoDetectados DPJ.PorcDeteccoesCorretas  DPJ.PorcAcertoAgrupamento
     NEO.PorcNaoDetectados NEO.PorcDeteccoesCorretas  NEO.PorcAcertoAgrupamento
     SNEO.PorcNaoDetectados SNEO.PorcDeteccoesCorretas  SNEO.PorcAcertoAgrupamento
     MNEO.PorcNaoDetectados MNEO.PorcDeteccoesCorretas  MNEO.PorcAcertoAgrupamento
   ]
save ('temp','numeroDeNeuronios','resultados','iloop')
clear all; clc; close all
load('temp')
end
```

Algorítimo A.2 - Rotina técnica de valor absoluto

```
function[VA]=metodoVA(sinal,calc,par,kappa)
tic
VA.kappa = kappa;
VA.sinal=abs(sinal);
VA.len = length(VA.sinal);
VA.size = size(VA.sinal);
% Gabarito
VA.gabaritoPos = calc.AmostraComSpike;
if(VA.size(1)==1)
    % C lculo do limiar
    VA.Limiar=VA.kappa*std(VA.sinal');
    % Detec  o
    [VA.DeteccaoPos, VA.DeteccaoValor]=deteccao(VA.sinal, VA.Limiar,
        par.ParametroDeteccao);
    % Avalia  o
    [VA.F_N, VA.F_P] = avaliacao(VA.DeteccaoPos, VA.gabaritoPos ,
        par.ParametroAvaliacao);
    VA.pN=(VA.F_N) / (length(VA.gabaritoPos)) *100;
    VA.pP=(VA.F_P) / (length(VA.gabaritoPos)) *100;
end

    % Alinhamento
spikes=zeros(length(VA.DeteccaoPos),size(VA.sinal,1)*par.interv);
VA.CentroAlinhado = zeros(length(VA.DeteccaoPos),1);
for linha = 1:length(VA.DeteccaoPos)
```

```
    inicio_spike_calc = VA.DeteccaoPos(linha) - (par.interv/2) +1;
    intervalo= inicio_spike_calc : VA.DeteccaoPos(linha) +(par.interv/2);
    if (min(intervalo)>0 && max(intervalo)<length(VA.sinal))
        [~, valorPosicaoMaximo]= max(abs(VA.sinal(:,intervalo))');
        VA.CentroAlinhado(linha) = inicio_spike_calc + round(mean(
            valorPosicaoMaximo));
        intervaloCentroMedio = VA.CentroAlinhado(linha) - (par.interv/2) +1:
            VA.CentroAlinhado(linha) +(par.interv/2);
        if (min(intervaloCentroMedio)>0 && max(intervaloCentroMedio)<length(
            VA.sinal))
            if(size(VA.sinal,1)>1)
                spikes(linha,:)=[ VA.sinal(1,intervaloCentroMedio) VA.sinal(2,
                    intervaloCentroMedio) VA.sinal(3,intervaloCentroMedio) VA.sinal
                    (4,intervaloCentroMedio)];
            else
                 spikes(linha,:)=VA.sinal(1,intervaloCentroMedio);
            end
        end
    end
end
clear linha
spikeSemCalculoPossivel = find(VA.CentroAlinhado==0);
spikes(spikeSemCalculoPossivel,:)=[];
VA.DeteccaoPos(spikeSemCalculoPossivel)=[];
VA.CentroAlinhado(find(VA.CentroAlinhado==0))=[];
r=VA.CentroAlinhado;

[PCA.coeff,PCA.score,PCA.latent,PCA.tsquared,PCA.explained,PCA.mu] = pca(spikes,'
    NumComponents',20);
n_clusters = 50;
idx=kmeans(PCA.score,n_clusters,'Options',statset('UseParallel',1),'MaxIter',10000)
    ;

for c=1:length(VA.gabaritoPos)
  [diferenca(c) posicao(c)] = min ( abs( VA.DeteccaoPos- double(VA.gabaritoPos(c))
      ) );
end
indDiffInvalidas=find(diferenca>par.ParametroDeteccao);

labelOrig=calc.labelOrig;
posicao(indDiffInvalidas)=[];
labelOrig(indDiffInvalidas)=[];
labelDetc=idx(posicao);

agrupamentoOriginal  = labelOrig;
agrupamentoCalculado = labelDetc;
[saida]=comparaAgrupamentos(agrupamentoOriginal, agrupamentoCalculado);

 VA.DeteccoesCorretas = length(VA.DeteccaoPos) - VA.F_P;
 VA.NaoDetectados = VA.F_N;

 VA.PorcAcertoAgrupamento=sum(agrupamentoOriginal == saida(agrupamentoCalculado)' )
     /length(agrupamentoOriginal)*100;

 VA.PorcDeteccoesCorretas =  VA.DeteccoesCorretas/length(VA.gabaritoPos)*100;
 VA.PorcNaoDetectados =  VA.NaoDetectados/length(VA.gabaritoPos)*100;
```

```
toc
```

Algorítimo A.3 - Rotina técnica de desvio padrão janelado

```
function[DPJ]=metodoDPJ(sinal,calc,par,kappa,janela)
%%
tic
DPJ.janela=janela;
DPJ.kappa=kappa;

DPJ.BDPJ = ones(1,DPJ.janela)/DPJ.janela;

% Pr -processamento
DPJ.sinal=PreProcDPJ(sinal, DPJ.BDPJ);
DPJ.len = length(DPJ.sinal);
% Gabarito
DPJ.gabaritoPos = calc.AmostraComSpike;
% C lculo do limiar
DPJ.Limiar=DPJ.kappa*mean(DPJ.sinal);
% Detec o
[DPJ.DeteccaoPos, DPJ.DeteccaoValor]=deteccao(DPJ.sinal, DPJ.Limiar,
    par.ParametroDeteccao);
DPJ.DeteccaoPos1 = DPJ.DeteccaoPos+(2*DPJ.janela);
% Avalia o
DPJ.gabaritoPos = DPJ.gabaritoPos +(DPJ.janela)-20;
[DPJ.F_N, DPJ.F_P d g] = avaliacao(DPJ.DeteccaoPos, DPJ.gabaritoPos ,
    par.ParametroAvaliacao);

DPJ.pN=(DPJ.F_N) / (length(DPJ.gabaritoPos)) *100;
DPJ.pP=(DPJ.F_P) / (length(DPJ.gabaritoPos)) *100;

  % Alinhamento

spikes=zeros(length(DPJ.DeteccaoPos),size(DPJ.sinal,1)*par.interv);
DPJ.CentroAlinhado = zeros(length(DPJ.DeteccaoPos),1);
for linha = 1:length(DPJ.DeteccaoPos)
    inicio_spike_calc = DPJ.DeteccaoPos(linha) - (par.interv/2) +1;
    intervalo= inicio_spike_calc : DPJ.DeteccaoPos(linha) +(par.interv/2);
    if (min(intervalo)>0 && max(intervalo)<length(DPJ.sinal))
        [~, valorPosicaoMaximo]= max(abs(DPJ.sinal(:,intervalo))');
        DPJ.CentroAlinhado(linha) = inicio_spike_calc + round(mean(
            valorPosicaoMaximo));
        intervaloCentroMedio = DPJ.CentroAlinhado(linha) - (par.interv/2) +1:
            DPJ.CentroAlinhado(linha) +(par.interv/2);
        if (min(intervaloCentroMedio)>0 && max(intervaloCentroMedio)<length(
            DPJ.sinal))
            if(size(DPJ.sinal,1)>1)
                spikes(linha,:)=[ DPJ.sinal(1,intervaloCentroMedio) DPJ.sinal(2,
                    intervaloCentroMedio) DPJ.sinal(3,intervaloCentroMedio)
                    DPJ.sinal(4,intervaloCentroMedio)];
            else
                 spikes(linha,:)=DPJ.sinal(1,intervaloCentroMedio);
            end
        end
```

```
    end
end
clear linha
spikeSemCalculoPossivel = find(DPJ.CentroAlinhado==0);
spikes(spikeSemCalculoPossivel,:)=[];
DPJ.DeteccaoPos(spikeSemCalculoPossivel)=[];
DPJ.CentroAlinhado(find(DPJ.CentroAlinhado==0))=[];
r=DPJ.CentroAlinhado;
length(DPJ.CentroAlinhado);

[PCA.coeff,PCA.score,PCA.latent,PCA.tsquared,PCA.explained,PCA.mu] = pca(spikes,'
    NumComponents',20);

n_clusters = 50;

idx=kmeans(PCA.score,n_clusters,'Options',statset('UseParallel',1),'MaxIter',10000)
    ;

for c=1:length(DPJ.gabaritoPos)
  [diferenca(c) posicao(c)] = min ( abs( DPJ.DeteccaoPos-double(DPJ.gabaritoPos(c))
      ) );
end
indDiffInvalidas=find(diferenca>par.ParametroDeteccao);
posicao(indDiffInvalidas)=[];
labelOrig=calc.labelOrig;
labelOrig(indDiffInvalidas)=[];
labelDetc=idx((posicao));

agrupamentoOriginal  = labelOrig;
agrupamentoCalculado = labelDetc;
[saida]=comparaAgrupamentos(agrupamentoOriginal, agrupamentoCalculado);

 DPJ.DeteccoesCorretas = length(DPJ.DeteccaoPos) - DPJ.F_P;
 DPJ.NaoDetectados = DPJ.F_N;

 DPJ.PorcAcertoAgrupamento=sum(agrupamentoOriginal == saida(agrupamentoCalculado)'
     )/length(agrupamentoOriginal)*100;

 DPJ.PorcDeteccoesCorretas =  DPJ.DeteccoesCorretas/length(DPJ.gabaritoPos)*100;
 DPJ.PorcNaoDetectados =  DPJ.NaoDetectados/length(DPJ.gabaritoPos)*100;

toc

function[SinalPreProc]=PreProcDPJ(sinal,B)

xmedio = filter(B,1,sinal);
sinalsubquad = (sinal - xmedio).^2;
SinalPreProcQuad = filter(B,1,sinalsubquad);
SinalPreProc = sqrt(SinalPreProcQuad);
```

Algorítimo A.4 - Rotina técnica NEO

```
function[NEO]=metodoNEO(sinal,calc,par,kappa,atraso)

tic
```

```
% M todo NEO
NEO.atraso=atraso;
NEO.kappa=kappa;

% Pr -processamento
NEO.sinal=PreProcNEO(sinal,NEO.atraso);
% Gabarito
NEO.gabaritoPos = calc.AmostraComSpike;
% C lculo do limiar
NEO.Limiar=NEO.kappa*std(NEO.sinal);
% Detec  o
[NEO.DeteccaoPos, NEO.DeteccaoValor]=deteccao(NEO.sinal, NEO.Limiar,
    par.ParametroDeteccao);
% Avalia  o
[NEO.F_N, NEO.F_P d g] = avaliacao(NEO.DeteccaoPos, NEO.gabaritoPos ,
    par.ParametroAvaliacao);

NEO.pN=(NEO.F_N) / (length(NEO.gabaritoPos)) *100;
NEO.pP=(NEO.F_P) / (length(NEO.gabaritoPos)) *100;
  % Alinhamento

spikes=zeros(length(NEO.DeteccaoPos),size(NEO.sinal,1)*par.interv);
NEO.CentroAlinhado = zeros(length(NEO.DeteccaoPos),1);
for linha = 1:length(NEO.DeteccaoPos)
    inicio_spike_calc = NEO.DeteccaoPos(linha) - (par.interv/2) +1;
    intervalo= inicio_spike_calc : NEO.DeteccaoPos(linha) +(par.interv/2);
    if (min(intervalo)>0 && max(intervalo)<length(NEO.sinal))
        [~, valorPosicaoMaximo]= max(abs(NEO.sinal(:,intervalo))');
        NEO.CentroAlinhado(linha) = inicio_spike_calc + round(mean(
            valorPosicaoMaximo));
        intervaloCentroMedio = NEO.CentroAlinhado(linha) - (par.interv/2) +1:
            NEO.CentroAlinhado(linha) +(par.interv/2);
        if (min(intervaloCentroMedio)>0 && max(intervaloCentroMedio)<length(
            NEO.sinal))
            if(size(NEO.sinal,1)>1)
                spikes(linha,:)=[ NEO.sinal(1,intervaloCentroMedio) NEO.sinal(2,
                    intervaloCentroMedio) NEO.sinal(3,intervaloCentroMedio)
                    NEO.sinal(4,intervaloCentroMedio)];
            else
                 spikes(linha,:)=NEO.sinal(1,intervaloCentroMedio);
            end
        end
    end
end
clear linha
spikeSemCalculoPossivel = find(NEO.CentroAlinhado==0);
spikes(spikeSemCalculoPossivel,:)=[];
NEO.DeteccaoPos(spikeSemCalculoPossivel)=[];
NEO.CentroAlinhado(find(NEO.CentroAlinhado==0))=[];
r=NEO.CentroAlinhado;
length(NEO.CentroAlinhado);
[PCA.coeff,PCA.score,PCA.latent,PCA.tsquared,PCA.explained,PCA.mu] = pca(spikes,'
    NumComponents',20);
n_clusters = 50;
idx=kmeans(PCA.score,n_clusters,'Options',statset('UseParallel',1),'MaxIter',10000)
    ;
for c=1:length(NEO.gabaritoPos)
```

```
  [diferenca(c) posicao(c)] = min ( abs( NEO.DeteccaoPos-double(NEO.gabaritoPos(c))
      ) );
end
indDiffInvalidas=find(diferenca>par.ParametroDeteccao);
posicao(indDiffInvalidas)=[];
labelOrig=calc.labelOrig;
labelOrig(indDiffInvalidas)=[];
labelDetc=idx((posicao));

agrupamentoOriginal  = labelOrig;
agrupamentoCalculado = labelDetc;
[saida]=comparaAgrupamentos(agrupamentoOriginal, agrupamentoCalculado);

 NEO.DeteccoesCorretas = length(NEO.DeteccaoPos) - NEO.F_P;
 NEO.NaoDetectados = NEO.F_N;

 NEO.PorcAcertoAgrupamento=sum(agrupamentoOriginal == saida(agrupamentoCalculado)'
     )/length(agrupamentoOriginal)*100;

 NEO.PorcDeteccoesCorretas =  NEO.DeteccoesCorretas/length(NEO.gabaritoPos)*100;
 NEO.PorcNaoDetectados =  NEO.NaoDetectados/length(NEO.gabaritoPos)*100;
 toc

 function[SinalPreProc]=PreProcNEO(sinal,Atraso);

sinalquad = sinal.^2;
sinaladianta = [sinal(Atraso+1:end) zeros(1,Atraso)];
sinalatrasa  = [zeros(1,Atraso) sinal(1:end-Atraso)];
SinalPreProc = sinalquad - sinaladianta.*sinalatrasa;
```

Algorítimo A.5 - Rotina técnica SNEO

```
function[SNEO]=metodoSNEO(sinal,calc,par,kappa,atraso)
%% M todo SNEO
tic

SNEO.atraso=atraso;
SNEO.kappa=kappa;

% Pr -processamento com ru do
SNEO.sinal=PreProcSNEO(sinal,SNEO.atraso);
% Gabarito
SNEO.gabaritoPos = calc.AmostraComSpike;
% C lculo do limiar
SNEO.Limiar=SNEO.kappa*std(SNEO.sinal);
% Detec  o
[SNEO.DeteccaoPos, SNEO.DeteccaoValor]=deteccao(SNEO.sinal, SNEO.Limiar,
    par.ParametroDeteccao);
% Avalia  o
[SNEO.F_N, SNEO.F_P  d g] = avaliacao(SNEO.DeteccaoPos, SNEO.gabaritoPos ,
    par.ParametroAvaliacao);

SNEO.pN=(SNEO.F_N) / (length(SNEO.gabaritoPos)) *100;
SNEO.pP=(SNEO.F_P) / (length(SNEO.gabaritoPos)) *100;
  % Alinhamento
```

```
spikes=zeros(length(SNEO.DeteccaoPos),size(SNEO.sinal,1)*par.interv);
SNEO.CentroAlinhado = zeros(length(SNEO.DeteccaoPos),1);
for linha = 1:length(SNEO.DeteccaoPos)
    inicio_spike_calc = SNEO.DeteccaoPos(linha) - (par.interv/2) +1;
    intervalo= inicio_spike_calc : SNEO.DeteccaoPos(linha) +(par.interv/2);
    if (min(intervalo)>0 && max(intervalo)<length(SNEO.sinal))
        [~, valorPosicaoMaximo]= max(abs(SNEO.sinal(:,intervalo))');
        SNEO.CentroAlinhado(linha) = inicio_spike_calc + round(mean(
            valorPosicaoMaximo));
        intervaloCentroMedio = SNEO.CentroAlinhado(linha) - (par.interv/2) +1:
            SNEO.CentroAlinhado(linha) +(par.interv/2);
        if (min(intervaloCentroMedio)>0 && max(intervaloCentroMedio)<length(
            SNEO.sinal))
            if(size(SNEO.sinal,1)>1)
                spikes(linha,:)=[ SNEO.sinal(1,intervaloCentroMedio) SNEO.sinal(2,
                    intervaloCentroMedio) SNEO.sinal(3,intervaloCentroMedio)
                    SNEO.sinal(4,intervaloCentroMedio)];
            else
                 spikes(linha,:)=SNEO.sinal(1,intervaloCentroMedio);
            end
        end
    end
end
clear linha
spikeSemCalculoPossivel = find(SNEO.CentroAlinhado==0);
spikes(spikeSemCalculoPossivel,:)=[];
SNEO.DeteccaoPos(spikeSemCalculoPossivel)=[];
SNEO.CentroAlinhado(find(SNEO.CentroAlinhado==0))=[];
r=SNEO.CentroAlinhado;
length(SNEO.CentroAlinhado);
[PCA.coeff,PCA.score,PCA.latent,PCA.tsquared,PCA.explained,PCA.mu] = pca(spikes,'
    NumComponents',20);
n_clusters = 50;
idx=kmeans(PCA.score,n_clusters,'Options',statset('UseParallel',1),'MaxIter',10000)
    ;
for c=1:length(SNEO.gabaritoPos)
  [diferenca(c) posicao(c)] = min ( abs( SNEO.DeteccaoPos-double(SNEO.gabaritoPos(c
      )) ) );
end
indDiffInvalidas=find(diferenca>par.ParametroDeteccao);
posicao(indDiffInvalidas)=[];
labelOrig=calc.labelOrig;
labelOrig(indDiffInvalidas)=[];
labelDetc=idx((posicao));

agrupamentoOriginal  = labelOrig;
agrupamentoCalculado = labelDetc;
[saida]=comparaAgrupamentos(agrupamentoOriginal, agrupamentoCalculado);

 SNEO.DeteccoesCorretas = length(SNEO.DeteccaoPos) - SNEO.F_P;
 SNEO.NaoDetectados = SNEO.F_N;

 SNEO.PorcAcertoAgrupamento=sum(agrupamentoOriginal == saida(agrupamentoCalculado)'
     )/length(agrupamentoOriginal)*100;

 SNEO.PorcDeteccoesCorretas =  SNEO.DeteccoesCorretas/length(SNEO.gabaritoPos)*100;
```

```
 SNEO.PorcNaoDetectados =  SNEO.NaoDetectados/length(SNEO.gabaritoPos)*100;

 toc

function[SinalPreProc]=PreProcSNEO(sinal,Atraso);

sinalquad = sinal.^2;
sinaladianta = [sinal(Atraso+1:end) zeros(1,Atraso)];
sinalatrasa  = [zeros(1,Atraso) sinal(1:end-Atraso)];
SinalPreProcNEO = sinalquad - sinaladianta.*sinalatrasa;
ordem=4*Atraso+1;
B=hamming(ordem);
SinalPreProc = filter(B,1,SinalPreProcNEO);
```

Algorítimo A.6 - Rotina técnica MNEO

```
function[MNEO]=metodoMNEO(sinal,calc,par,kappa,atraso)
%% M todo MNEO

tic
MNEO.atraso=atraso;
MNEO.kappa=kappa;

% Pr -processamento com ru do
MNEO.sinal=PreProcMNEO(sinal,MNEO.atraso)';
% Gabarito
MNEO.gabaritoPos = calc.AmostraComSpike;
% C lculo do limiar
MNEO.Limiar=MNEO.kappa*std(MNEO.sinal);
% Detec  o
[MNEO.DeteccaoPos, MNEO.DeteccaoValor]=deteccao(MNEO.sinal, MNEO.Limiar,
    par.ParametroDeteccao);
% Avalia  o
[MNEO.F_N, MNEO.F_P d g] = avaliacao(MNEO.DeteccaoPos, MNEO.gabaritoPos ,
    par.ParametroAvaliacao);
MNEO.pN=(MNEO.F_N) / (length(MNEO.gabaritoPos)) *100;
MNEO.pP=(MNEO.F_P) / (length(MNEO.gabaritoPos)) *100;

  % Alinhamento
spikes=zeros(length(MNEO.DeteccaoPos),size(MNEO.sinal,1)*par.interv);
MNEO.CentroAlinhado = zeros(length(MNEO.DeteccaoPos),1);
for linha = 1:length(MNEO.DeteccaoPos)
    inicio_spike_calc = MNEO.DeteccaoPos(linha) - (par.interv/2) +1;
    intervalo= inicio_spike_calc : MNEO.DeteccaoPos(linha) +(par.interv/2);
    if (min(intervalo)>0 && max(intervalo)<length(MNEO.sinal))
        [~, valorPosicaoMaximo]= max(abs(MNEO.sinal(:,intervalo))');
        MNEO.CentroAlinhado(linha) = inicio_spike_calc + round(mean(
            valorPosicaoMaximo));
        intervaloCentroMedio = MNEO.CentroAlinhado(linha) - (par.interv/2) +1:
            MNEO.CentroAlinhado(linha) +(par.interv/2);
        if (min(intervaloCentroMedio)>0 && max(intervaloCentroMedio)<length(
            MNEO.sinal))
            if(size(MNEO.sinal,1)>1)
                spikes(linha,:)=[ MNEO.sinal(1,intervaloCentroMedio) MNEO.sinal(2,
                    intervaloCentroMedio) MNEO.sinal(3,intervaloCentroMedio)
```

```
                    MNEO.sinal(4,intervaloCentroMedio)];
            else
                spikes(linha,:)=MNEO.sinal(1,intervaloCentroMedio);
            end
        end
    end
end
clear linha
spikeSemCalculoPossivel = find(MNEO.CentroAlinhado==0);
spikes(spikeSemCalculoPossivel,:)=[];
MNEO.DeteccaoPos(spikeSemCalculoPossivel)=[];
MNEO.CentroAlinhado(find(MNEO.CentroAlinhado==0))=[];
r=MNEO.CentroAlinhado;
length(MNEO.CentroAlinhado);
[PCA.coeff,PCA.score,PCA.latent,PCA.tsquared,PCA.explained,PCA.mu] = pca(spikes,'
    NumComponents',20);
n_clusters = 50;
idx=kmeans(PCA.score,n_clusters,'Options',statset('UseParallel',1),'MaxIter',10000)
    ;
for c=1:length(MNEO.gabaritoPos)
  [diferenca(c) posicao(c)] = min ( abs( MNEO.DeteccaoPos-double(MNEO.gabaritoPos(c
      )) ) );
end
indDiffInvalidas=find(diferenca>par.ParametroDeteccao);
posicao(indDiffInvalidas)=[];
labelOrig=calc.labelOrig;
labelOrig(indDiffInvalidas)=[];
labelDetc=idx((posicao));

agrupamentoOriginal  = labelOrig;
agrupamentoCalculado = labelDetc;
[saida]=comparaAgrupamentos(agrupamentoOriginal, agrupamentoCalculado);
 MNEO.DeteccoesCorretas = length(MNEO.DeteccaoPos) - MNEO.F_P;
 MNEO.NaoDetectados = MNEO.F_N;

 MNEO.PorcAcertoAgrupamento=sum(agrupamentoOriginal == saida(agrupamentoCalculado)'
      )/length(agrupamentoOriginal)*100;

 MNEO.PorcDeteccoesCorretas =  MNEO.DeteccoesCorretas/length(MNEO.gabaritoPos)*100;
 MNEO.PorcNaoDetectados =  MNEO.NaoDetectados/length(MNEO.gabaritoPos)*100;
 toc


function[SinalPreProc]=PreProcMNEO(sinal,Atrasos);
SinalPreProcOrdem=zeros(length(Atrasos),length(sinal));
for i=1:length(Atrasos);
    sinalquad = sinal.^2;
    sinaladianta = [sinal(Atrasos(i)+1:end) zeros(1,Atrasos(i))];
    sinalatrasa  = [zeros(1,Atrasos(i)) sinal(1:end-Atrasos(i))];
    SinalPreProcNEO = sinalquad - sinaladianta.*sinalatrasa;
    ordem=4*Atrasos(i)+1;
    B=hamming(ordem);
    SinalPreProcOrdem(i,:) = filter(B,1,SinalPreProcNEO);
end
SinalPreProc=max(SinalPreProcOrdem',[],2);
end
```