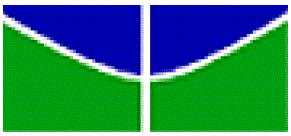

Universidade de Brasília

Instituto de Ciências Exatas

Departamento de Estatística

Dissertação de Mestrado

Selecionador de Características para classificação de sinais de EEG e construção de Interfaces Cérebro-Máquina

por

Murilo Coutinho Silva

Orientador: Prof. George F. von Borries, Ph.D.

Dezembro de 2012

Murilo Coutinho Silva

Selecionador de Características para classificação de sinais de EEG e construção de Interfaces Cérebro-Máquina

Dissertação apresentada ao Departamento de Estatística do Instituto de Ciências Exatas da Universidadede de Brasília como requisito parcial à obtenção do título de Mestre em Estatística.

Universidade de Brasília

Brasília, Dezembro de 2012

Termo de Aprovação

Murilo Coutinho Silva

Selecionador de Características para classificação de sinais de EEG e construção de Interfaces Cérebro-Máquina

Dissertação apresentada ao Departamento de Estatística do Instituto de Ciências Exatas da Universidadede de Brasília como requisito parcial à obtenção do título de Mestre em Estatística.

Data da defesa: 12 de dezembro de 2012

Orientador:

Prof. George Freitas von Borries - EST/UnB.

Comissão Examinadora:

Prof. Antônio Pereira - Instituto do Cérebro - UFRN.

Prof. André Luiz Fernandes Cançado - EST/UnB.

Prof. Pushpa Narayan Rathie

Colaborador EST/UnB. (Suplente)

Brasília, Maio de 2013

Ficha Catalográfica

COUTINHO, MURILO
Selecionador de Características para classificação de sinais
de EEG e construção de Interfaces Cérebro-Máquina,
(UnB - IE, Mestre em Estatística, 2010).

Dissertação de Mestrado - Universidade de Brasília.
Departamento de Estatística - Instituto de Ciências Exatas.

1. eletroencefalografia 2. EEG 3. support vector machine 4. SVM
5. interface cérebro-máquina 6. cérebro 7. neurônios 8. classificação
9. aprendizado estatístico 10. selecionador de características
11. análise de variância 12. teste FDR 13. reconhecimento neural
14. imagética motora 15. artefatos

Agradecimentos

Vivemos em um mundo caótico. Como profetiza o efeito borboleta pequenas alterações presentes podem trazer grandes consequências futuras e, por isso, não podemos deixar de nos questionar como chegamos nesse exato instante de nossas vidas. Que atos e decisões tomadas no passado poderiam nos levar a realidades completamente distintas se tivéssemos agido de maneira diferente?

Esse tipo de pensamento sugere que esta dissertação de mestrado não é, por algum ponto de vista, um trabalho realizado apenas por mim com o auxílio do meu orientador, mas um trabalho que aparece como mera consequência das minhas escolhas durante a vida e das relações com todas as pessoas que fazem parte dela. Portanto, sinto-me obrigado a agradecer a estas pessoas que, tenho certeza, sem elas nada disso seria possível.

Primeiramente gostaria de agradecer à minha mãe que me criou com muito esforço, cuidou de mim com muito carinho em todos os momentos que mais precisei dela, me ensinou amor, moral e ética. Sou imagem da pessoa maravilhosa que ela é pois é nela em quem me espelho.

Em seguida agradeço ao resto da minha família. Agradeço minha irmã, minha avó, a minha tia, aos meus tios, meu padrasto e meus primos que são grandes amigos acima de tudo. Agradeço a quase todos os momentos de minha vida em que estive na ilustre presença dessas pessoas que foram sempre momentos de extrema felicidade.

Agradeço, também, a outro pilar da minha vida: a minha companheira Lana, uma pessoa maravilhosa que me completa de maneira perfeita. Ela me apoia e me incentiva em tudo que faço e constitui parte importante deste trabalho.

A personalidade de cada pessoa é moldada, em grande parte, pelos amigos que são feitos durante a vida. O amigos constituem, portanto, a composição social mais

caótica da vida. Alguns têm péssimas influências, já eu tive ótimas. Agradeço a esse grandes amigos que estiveram comigo desde a infância. Nos transformamos juntos no que somos, e a ausência de qualquer um desse grupo mudaria enormemente a vida que levei.

Agradeço, agora, os professores que me ergueram ao patamar em que me encontro. O primeiro professor a quem devo muito é o Luís Antônio, meu professor do ensino médio com quem aprendi cálculo e que me incentivou enormemente nos estudos. O segundo é o professor Rathie com quem fiz dois anos de iniciação científica e que me ensinou a fazer e gostar da pesquisa, me proporcionando, inclusive, alguns artigos publicados. O terceiro é o meu orientador (e amigo) George von Borries, que é, obviamente, de fundamental importância para este trabalho.

Para finalizar devo citar mais uma pessoa: meu falecido avô, Mário Coutinho, que de forma estranha e sobrenatural me proporciona uma enorme força com cada simples lembrança, como se estivesse sempre ao meu lado segurando minha mão para me levar aonde tenho que ir, como me levava há muito tempo atrás.

Sumário

Lista de Figuras

Lista de Tabelas

Resumo

A classificação de sinais de eletroencefalografia (EEG) vem sendo muito estudada recentemente para proporcionar aplicações como as Interfaces-Cérebro Máquina. Parte fundamental do processo de classificação é a chamada *extração de características* dos sinais de EEG. Na literatura, diversas técnicas de extração de características foram apresentadas e, entretanto, não existe uma técnica que supere as demais em todas as situações. Para solucionar este problema, este trabalho apresenta um novo algoritmo que seleciona automaticamente as melhores *características* selecionadas por várias técnicas de extração simultaneamente, produzindo um conjunto ótimo e reduzido de características que não é necessariamente o mesmo para cada aplicação prática. Pelo uso do novo algoritmo, todas as técnicas já apresentadas na literatura e as técnicas futuras podem ser combinadas para produzir o melhor e mais poderoso conjunto de características gerando taxas de classificação excelentes. Neste trabalho, o selecionador de características é testado utilizando vários conjuntos de dados reais obtendo as melhores taxas de classificação quando comparado a outras técnicas de classificação de dados de EEG.

Palavras Chave: *eletroencefalografia, EEG, extração de características, feature extraction, support vector machine, SVM, interface cérebro-máquina, cérebro, neurônios, classificação, aprendizado estatístico, selecionador de características, análise de variância, teste FDR, reconhecimento neural, imagética motora, artefatos.*

Abstract

The classification of electroencephalography signals (EGG) has been extensively studied recently to provide applications such as Brain-Machine Interfaces. An important part of the classification of EEG signals is called *feature extraction*. In the literature, several techniques for feature extraction were presented, however, there is not one technique that overcomes the others in all situations. To solve this problem, this paper presents a new algorithm that selects the best set of *features*, extracted by several techniques simultaneously, producing an optimal and reduced set of features which is not necessarily the same for every practical application. By using the new algorithm, all the techniques presented in the literature can be combined to produce the best and most powerful set of features generating excellent classification rates. In this work, the new algorithm for feature extraction is tested using several real data sets obtaining the best classification rates when compared to other classification techniques of EEG data.

key words: *eletroencefalography, EEG, feature extraction, support vector machine, SVM, brain-machine interface, brain, neurons, classification, statistical learning, feature selector, variance analysis, FDR test, neural recognition, motor imagery, artifacts.*

Capítulo 1

Introdução

Considerando que os pensamentos que temos quando acordados nos podem ocorrer também quando dormimos, sem que neste caso nenhum seja verdadeiro, resolvi supor que tudo o que até então encontrara acolhimento no meu espírito não era mais verdadeiro que as ilusões dos meus sonhos. Mas, logo em seguida, notei que, enquanto assim queria pensar que tudo era falso, eu, que assim o pensava, necessariamente era alguma coisa. E notando esta verdade: eu penso, logo existo, era tão firme e tão certa que todas as extravagantes suposições dos céticos seriam impotentes para abalar, julguei que a podia aceitar, sem escrúpulo, para primeiro princípio da filosofia que procurava.

- René Descartes, em 'Discurso do Método'

Ao tentar compreender o mundo e a realidade, o filósofo e fisiologista René Descartes chegou a maior e mais incontestável verdade concebível pela mente humana: "penso, logo existo". Enquanto o mundo à nossa volta pode, pela mente, ser posto em xeque como um mero sonho, esta não pode duvidar de si e da existência de algo. Porém, como pode a mente se convencer da sua existência, do mundo ao seu redor e da existência de outras mentes, que, por serem mentes, devem por definição existir?

Afinal como pensamos? Nossa mente é, na verdade, um subproduto do funcionamento de um órgão extremamente importante e misterioso: o *cérebro* que, conhecido de todos, é o responsável por todos os nossos pensamentos, sensações, sentimentos, memórias e raciocínios. O cérebro é um órgão composto por uma rede de 86 bilhões

de neurônios que se comunicam via impulsos elétricos, um número assustadoramente grande, próximo do número de estrelas contidas na via láctea, formando uma rede 12 vezes maior do que a rede que seria formada se todas as pessoas do planeta estivessem conectadas à internet ao mesmo tempo!

O mistério de como essa rede de neurônios possibilita a um ser vivo ser capaz de perceber, entender e interagir com o mundo ao seu redor foi o combustível do trabalho de vários cientistas que focaram suas atenções no estudo do funcionamento do cérebro. Desde Herophilus em 300 a.C., o primeiro a reconhecer o cérebro como o centro do sistema nervoso e, mais precisamente a partir 1894, quando Santiago Ramón y Cajal, o pai da neurociência, publicou o trabalho "Les nouvelles idées sur la fine anatomie des centres nerveux", a neurociência evoluiu e, nos dias atuais, é a estrela principal dos horizontes da evolução humana.

Hoje, existem equipamentos dos mais diversos tipos nos quais é possível registrar a atividade cerebral em bancos de dados que, em outras palavras, registram pensamentos em arquivos em um computador. Talvez Descartes ficasse surpreso ao descobrir que a realidade, assim como o eu de uma mente, também existe como princípio básico, já que o pensamento de uma mente pode ser registrado ou mesmo visualizado em tempo real por equipamentos de eletroencefalografia ou de ressonância magnética, por exemplo.

E não é só isso, pesquisadores e cientistas das mais diversas áreas de formação e de toda parte do mundo passaram a tentar compreender os padrões que permeiam esses dados registrados do cérebro com o intuito de, literalmente, ler a mente do ser humano submetido ao uso desses aparelhos de captação da atividade cerebral. Na verdade, ainda um feito difícil de ser alcançado.

No entanto, a interpretação dos dados cerebrais fez surgir a possibilidade única almejada durante muitos anos pelos seres humanos de controlar o mundo pelo pensamento via construção das *Interfaces Cérebro-Máquina* (ICM). Uma ICM (ver Figura 1.1) nada mais é do que um sistema conectando uma pessoa à um computador de forma que, ao pensar, os dados captados do cérebro sejam enviados ao computador que, por sua vez, os interpreta usando algum sistema de inteligência artificial e produz uma resposta que torna-se um comando para um computador ou máquina.

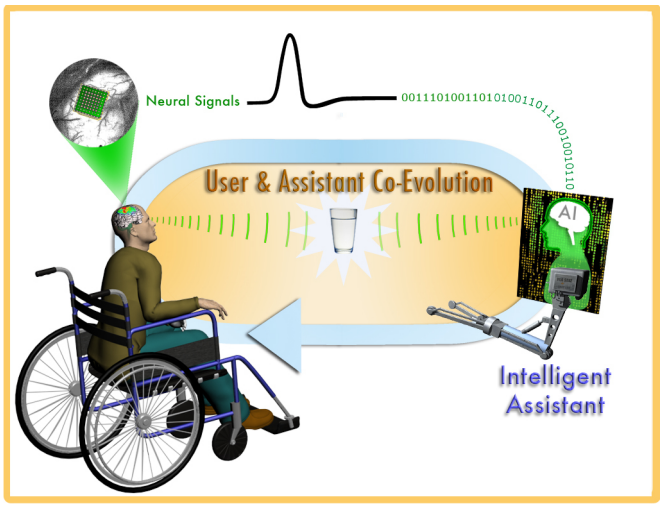

Figura 1.1: *Organização de uma Interface Cérebro-Máquina (retirado do site do Neuroprosthetics Research Group, Universidade de Miami).*

Embora seja uma área de estudo nova e que ainda engatinha, a concretização de uma Interface Cérebro-Máquina promete revolucionar o mundo que conhecemos e proporcionar um salto tecnológico gigantesco tendo em vista o universo amplo de aplicações que são possíveis. Imagine um mundo em que pessoas com deficiências físicas possam recuperar seus movimentos com o uso de próteses mecânicas controladas pelo cérebro, imagine um computador que possa escrever seus pensamentos eliminando o teclado e possa falar por pessoas que perderam a fala, imagine uma casa inteiramente conectada ao dono onde tudo possa ser controlado pelo pensamento! Essas são apenas algumas das aplicações possíveis pela concretização de uma Interface Cérebro-Máquina.

É então nesse contexto de desenvolvimento diário ao redor do mundo das Interfaces Cérebro-Máquina que se estrutura este trabalho que tenta promover uma contribuição à pesquisa apresentando técnicas estatísticas de aprendizado de máquina para classificação e interpretação de dados de eletroencefalografia (EEG).

O objetivo principal deste trabalho é apresentar ferramentas e algoritmos capazes de extrair as características ótimas, do ponto de vista de classificação, de bancos de dados provenientes da coleta de sinais de EEG. A organização do trabalho foi feita da seguinte forma:

O segundo capítulo busca contextualizar o leitor a respeito do que é o cérebro e o que já sabemos sobre ele. Para tanto são apresentados os aspectos históricos da neurociência e também uma breve explanação do que é conhecido sobre o funcionamento do cérebro e dos neurônios.

O terceiro capítulo faz um breve estudo sobre a história do eletroencefalograma (EEG) para situar o leitor sobre o básico desta técnica de coleta de dados cerebrais.

O quarto capítulo apresenta uma introdução ao conceito de interfaces cérebro-máquina, mostrando o que são, como funcionam e também alguns dos grupos que trabalham ao redor do mundo nesta área de pesquisa.

O quinto capítulo apresenta técnicas da área de aprendizado estatístico (extração de características e classificadores) que serão utilizadas na construção de um novo algoritmo para a seleção de características e classificação de sinais de EEG.

O sexto capítulo apresenta uma comparação de técnicas antigas para a classificação de sinais de EEG por meio de alguns exemplos com dados reais. Esses exemplos foram escolhido para mostrar que, atualmente, não existe uma técnica de classificação que se sobreponha as demais e que não há um consenso sobre qual a melhor forma de extração de características.

Esses fatos servem de motivação para o objetivo do trabalho que é concretizado no sétimo capítulo: estabelecer um algoritmo capaz de fazer a seleção automática de característica para a classificação de dados de eletroencefalografia.

Finalmente, o oitavo e último capítulo apresenta alguns experimentos realizados com dados reais que comprovam a eficácia do algoritmo proposto e sua superioridade quando comparado a outros métodos de classificação.

Capítulo 2

O cérebro

2.1 História da Neurociência

"Rifa-se um coração que na realidade está um pouco usado, meio calejado, muito machucado e que teima em alimentar sonhos, e cultivar ilusões. Um pouco inconsequente que nunca desiste de acreditar nas pessoas."

- Clarice Lispector (Rifa)

O coração dos poemas lembra, pensa, sente e sonha. O coração, então, toma, como metáfora, o lugar de direito do cérebro que, convenhamos, de poético não tem nada. Mas quem sabe esse coração inteligente dos poetas não é uma herança dos nossos antepassados que por volta de 350 AC acreditavam no coração como, de fato, centro do sistema nervoso?

Foi Praxagoras [1], inspirado nos estudos de Aristóteles sobre a anatomia do corpo humano, que criou e introduziu o cardiocentrismo insistindo que o coração era o centro de todas as funções psíquicas dos seres humanos. Porém, Herophilus de Chalcedon (330-250 AC), um discípulo de Praxagoras em Alexandria, desenvolveu o método de estudo da anatomia de Aristóteles com o uso da dissecção e vivissecção (técnicas que não eram permitidas em nenhum outro lugar do mundo) e provou que o cérebro, e não o coração, era o responsável pelas funções nervosas e psíquicas dos seres humanos.

Com a comprovação do cérebro como centro do sistema nervoso tornou-se questão de tempo para que os estudos sobre o funcionamento deste órgão começassem. Erasis-

tratus (Grego, 304-250 AC), contemporâneo de Herophilus, também estudou o cérebro via dissecção e foi o primeiro a notar diferenças entre certas regiões do cérebro, e dividiu cérebro e cerebelo identificando este como o responsável pela função motora.

A partir de Erasistratus vários estudos sobre as funções das regiões cerebrais passaram a ser desenvolvidos: Nemesius (390 DC) identificou o que acreditava ser as regiões cerebrais responsáveis pelas percepções sensoriais, imaginação, memória e julgamento. Em 1025, Avicenna escreveu sobre a visão e o olho humano. Em 1681, Thomas Willis escreveu a primeira monografia sobre a anatomia do cérebro e fisiologia descrevendo as divisões do cérebro em hemisférios e cunhando termos como "neurologia". Em 1704, Antonio Valsalva publicou um trabalho sobre a audição. Em 1760, Arne-Charles demonstrou que danos ao cerebelo afetavam a coordenação motora. Em 1789, John Dalton produziu uma explicação científica do daltonismo. Também em 1836, Marc Dax apresentou um trabalho mostrando que danos no hemisfério esquerdo do cérebro afetavam a fala.

Porém, nem todos os avanços da neurociência ocorreram seguindo o método científico, algumas descobertas aconteceram de forma ocasional e por vezes trágica. Um exemplo é o incrível caso de Phineas Gage, um jovem supervisor de construção de ferrovias em Vermont que, em 1848, precisava explodir uma pedra, trabalho que era feito abrindo-se um buraco na pedra e colocando uma mistura de pólvora e areia que então era comprimida utilizando um bastão de ferro. Ao tentar comprimir a pólvora, Gage, que provavelmente não utilizou areia na mistura, sofreu um grave acidente: a pólvora explodiu e o bastão de ferro foi arremessado na direção do seu rosto atingindo-o na bochecha e atravessando o crânio de baixo para cima (ver Figura 2.1).

Gage sofreu convulsões e foi levado imediatamente a um médico local e incrivelmente, apesar de ter perdido muito sangue e sofrido com graves infecções, sobreviveu e se recuperou bem. Porém Gage sofreu lesões nos lobos frontais, mais especificamente no córtex pré-frontal que é a região relacionada com a personalidade, e o rapaz que era descrito como um profissional responsável, uma pessoa tranquila e equilibrada e com boas relações sociais, simplesmente se transformou em outra pessoa, um homem desrespeitoso, irritado e incapaz de adequar-se as normas sociais. O caso de Phineas Gage foi, para a neurociência, importante e bastante estudado, uma forma triste e trágica de entrar para a história da ciência.

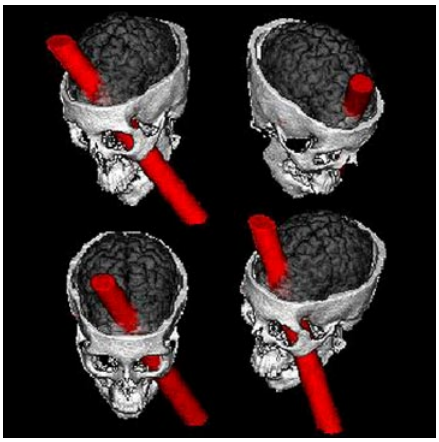

Figura 2.1: *Acidente de Phineas Gage.*

Todos os cientistas citados tiveram sua importância, mas o primeiro grande neurocientista, vencedor do prêmio nobel de 1906 em fisiologia e medicina e considerado o "pai da neurociência" foi Santiago Ramón y Cajal (1852-1934) [2]. Foi em 1852 que Cajal, nascido no vilarejo de Petilla na Espanha, foi exposto ao evento chave que daria início a sua revolução: Luis Simarro Lacabra mostrou a Cajal um tecido cerebral impregnado com um composto de nitrado de prata chamado de método de coloração de Golgi. Impressionado Cajal passou a estudar incansavelmente os tecidos nervosos como ele próprio descreve em sua autobiografia: "Ideias ferviam e borbulhavam em minha mente. Uma febre por publicações me devorava..." [3].

Utilizando o nitrato de prata Cajal produziu diversas imagens extremamente nítidas do tecido nervoso (ver Figura 2.2) e contrariando a ideia de Gerlach (1820-1896) de que o sistema nervoso era uma rede de elementos contínuos, propôs que o sistema nervoso era composto por bilhões de células nervosas, unidades elementares que seriam chamadas de "neurônios" por Waldeyer, conclusão que levou a concretização do modelo moderno de organização do sistema nervoso. Além disso definiu a *lei de polarização dinâmica* que diz que as células nervosas são polarizadas recebendo informação pelo corpo celular e pelos dendritos conduzindo informação para regiões distantes através dos axônios, o que se tornou o princípio básico de funcionamento das conexões neurais.

A partir de Cajal vários outros cientistas produziram trabalhos importantes para a neurociência, como por exemplo, Edgar Adrian e Charles Sherrington que venceram

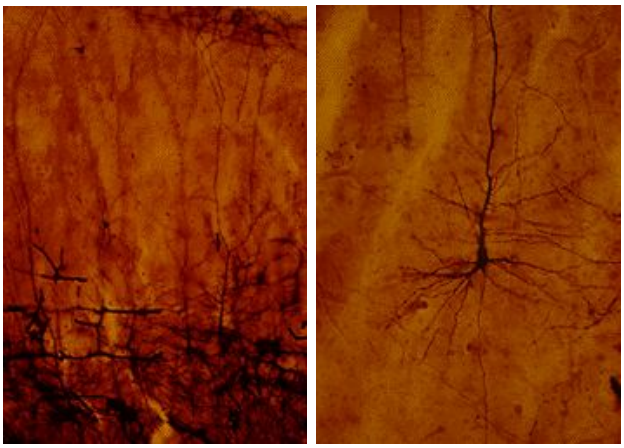

Figura 2.2: *Imagens do tecido nervoso produzidas por Cajal (guardadas no museu Cajal, no instituto Cajal em Madri, na Espanha).*

o nobel de medicina em 1932 por sua representação detalhada do mecanismo pelo qual os nervos transmitem mensagens. Em 1944, Joseph Erlanger e Habert Spencer Gasser venceram o nobel pelas descobertas das funções diferenciadas das fibras nervosas. Em 1963, Eccles, Hodgkin e Huxley venceram o nobel por descobrirem o meio químico pelo qual os impulsos nervosos ocorrem. Em 1981, Wiesel e Hubel venceram o nobel por seu trabalho sobre o modo pelo qual informações visuais são transmitidos da retina para o cérebro. Mais recentemente no ano 2000, Carlsson, Greengard e Kandel venceram o prêmio nobel pelas suas descobertas relativas a transdução de sinais no sistema nervoso.

Existem ainda diversos outros cientistas e trabalhos que mereceriam ser citados e que desenvolvem a neurociência na atualidade. Embora os cientistas tenham descoberto muito sobre a estrutura e funcionamento do sistema nervoso ainda há muito o que descobrir. Inumeráveis são os mistérios escondidos nesse complexo órgão dos quais vários, provavelmente, jamais tenhamos uma resposta. Como nas palavras de Steven Rose, professor de Neurobiologia na Universidade de Londres e autor de mais de 300 trabalhos sobre neurobiologia em grande parte sobre a memória: "O cérebro é o maior desafio da biologia. Talvez, de certo modo, o maior desafio da ciência como um todo, maior que as viagens à lua, que as menores partículas da física quântica e o infinito do espaço..." [4].

2.2 Neurônios

"Estima-se que o cérebro possui cem bilhões de células nervosas,
dois milhões de milhas de axônios e um milhão de bilhões
de sinapses, transformando-o na estrutura mais
complexa, natural ou artificial, na Terra."

- Tim Green, Stephen F. Heinemann e Jim F. Gusella [1]

O cérebro é constituído por bilhões de células chamadas neurônios que se comunicam entre si através de impulsos elétricos. Essa comunicação é a responsável por todos nossos pensamentos, memórias, raciocínios, sentimentos, e tudo que depende da atividade cerebral.

O neurônio é uma célula especial composta por algumas partes principais nomeadas corpo celular, axônio e dendritos, como mostrado na Figura 2.3. O corpo celular é a zona estrutural do neurônio onde estão localizados o citoplasma, as organelas e o citoesqueleto. Além disso é o corpo celular a parte capaz de produzir os impulsos elétricos para serem transmitidos a outros neurônios.

O axônio é o responsável pela transmissão dos impulsos elétricos produzidos no corpo celular para outros neurônios e é a parte mais comprida do neurônio. Se juntássemos todos os axônios de uma pessoa em uma mesma linha, produziríamos uma "corda" com mais de 3 milhões de quilômetros, comprimento suficiente para ir à Lua e voltar mais de três vezes ou mesmo dar mais de 75 voltas em torno da Terra.

Finalmente, os dendritos, do grego *déndron* (árvore), são vários filamentos que, espalhados, atuam na recepção de estímulos nervosos do ambiente ou de outros neurônios, transmitindo esses estímulos para o corpo celular.

Os impulsos elétricos emitidos pelos neurônios são causados por uma diferença de potencial entre o interior e o exterior da célula causados pela chamada bomba de sódio e potássio: a membrana plasmática do neurônio troca íons de sódio Na^+ do interior da célula por íons de potássio K^+ do exterior da célula, porém faz isso de maneira não balanceada de forma que para cada três íons de sódio bombeados para

[1] em um artigo em Neuron, vol. 420, pag 427, 1998.

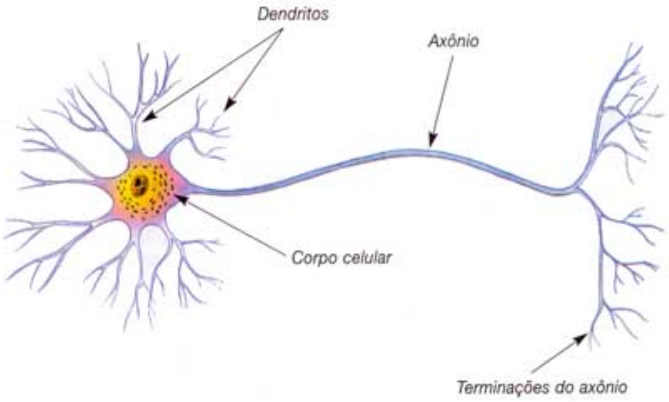

Figura 2.3: *O neurônio e suas partes principais (retirado do site http://www.sogab.com.br/).*

o líquido extracelular, somente dois íons de potássio são bombeados para o líquido intracelular.

Esta troca de sódio e potássio ocorre até que o interior da célula estabilize quando atinge um potencial de -70mV em relação ao meio externo. Porém as correntes elétricas que chegam pelos dendritos via sinapses deixam o potencial da membrana menos negativo de modo que ao atingir -55mV ocorre uma abertura da membrana para os íons de sódio que entram na célula deixando o potencial positivo por um instante ocorrendo o chamado *potencial de ação*. Em seguida os íons de potássio voltam a entrar enquanto que os íons de sódio voltam a sair da célula deixando o potencial negativo novamente. É o potencial de ação que é transmitido como informação para os neurônios subsequentes.

Capítulo 3

Eletroencefalografia

3.1 O EEG e sua coleta

O eletroencefalograma (EEG) é um tipo de coleta dos potenciais elétricos emitidos pelo cérebro. O EEG é uma técnica não invasiva, ou seja, a coleta é feita de forma externa ao crânio não havendo o contato direto com o cérebro. Isso acarreta algumas dificuldades pois o potencial elétrico emitido por um único neurônio é muito fraco para atravessar o crânio e ser captado de forma não invasiva. Porém, o sinal produzido pelo conjunto de vários neurônios é forte o suficiente para atravessar os tecidos e o crânio e pode ser coletado por eletrodos posicionados sobre o couro cabeludo.

Desta forma, os aparelhos de coleta do EEG são formados por um conjunto de eletrodos posicionados sobre o couro cabeludo que coletam os potenciais elétricos emitidos por diversas partes do cérebro, como pode ser visto na Figura 3.1. Existem diversos tipos de aparelhos no mercado com número de eletrodos variando de 1 à 256, e apresentando as mais diversas frequências de amostragem. Além disso, vários tipos de eletrodos são utilizados como, por exemplo, os de discos reutilizáveis que podem ser feitos de prata ou ouro, e também os do tipo agulha, que são compostos por diversas agulhas colocadas entre o crânio e o couro cabeludo.

Portanto, o EEG consiste em uma forma simples, prática e barata de se coletar os potenciais elétricos emitidos pelo cérebro, o que explica sua escolha para uso neste trabalho.

Figura 3.1: *Exemplo de aparelho de coleta de EEG.*

3.2 História do EEG

"É com imenso prazer que lhe digo que vosso resultado é considerado uma descoberta original, e que seu experimento foi replicado com sucesso se comprovando extremamente exato[...]. Em resumo, agora tudo é eletricidade animal, e seu nome é famoso em Pavia."

- Mariano Fontana, em nota se referindo a Luigi Galvani.

Nascido na Itália, Luigi Galvani (1737-1798) [5] foi um físico e fisiologista importante na história dos sinais biomédicos por ser o primeiro a demonstrar a existência de eletricidade nos seres vivos. Estudioso do que ele chamava de eletricidade animal, Galvani produziu um experimento crucial no qual fazia um sapo morto "pular" apenas tocando os seus músculos e nervos com um instrumento metálico conectado a uma máquina eletroestática (ver Figura 3.2). Tal experimento, considerado revolucionário pelos cientistas de seu tempo, fez com que Galvani estudasse a relação entre a eletricidade e a vida, motivo pelo qual é considerado o "pai da eletricidade animal".

Galvani publicou suas descobertas em 1792, em um trabalho de 53 páginas que produziu um grande alvoroço nos círculos científicos, tanto que seu experimento foi

replicado por todas as partes e logo o Galvanismo tornou-se uma febre, como se pode notar na nota destinada a Galvani escrita por Mariano Fontana que introduz essa seção. O Galvanismo também causou uma repercussão enorme entre os leigos e curiosos que o transformaram em sinônimo de "essência da vida" o que inspirou, inclusive, histórias de ficção sobre mortos voltando à vida, como foi o caso da obra, *Frankenstein*, de Mary Shelley.

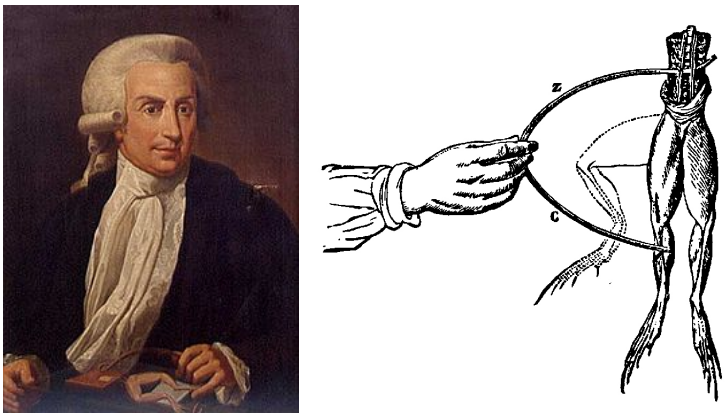

Figura 3.2: *Galvani e seu experimento com o sapo (trabalho de David Ames Wells, 1859).*

O sucesso de Galvani foi eventualmente abafado devido às críticas de outro cientista famoso de sua época: o físico Alessandro Volta (1745-1827), inventor da pilha elétrica, que acreditava que as contrações que ocorriam nas pernas do sapo dependiam dos metais utilizados por Galvani para conectar nervos e músculos. Apesar disso o nome de Galvani se perpetua na história por suas descobertas e também nos termos cunhados em sua homenagem como por exemplo o Galvanômetro (aparelho capaz de medir correntes elétricas de baixa intensidade) que foi utilizado por Richard Caton para fazer o primeiro registro da atividade elétrica do cérebro de animais.

Richard Caton (1842-1926) [6] foi um físico pioneiro que estudou a fundo os fenômenos eletrofisiológicos e, utilizando um galvanômetro, fez o primeiro registro da atividade elétrica do cérebro de animais. Em seus estudos, Caton notou as correntes elétricas que permeavam o cérebro e, embora seus registros não possam ser considerados um EEG propriamente dito, cunhou a frase que é considerada o marco do nascimento da eletroencefalografia: "correntes fracas de diferentes direções pas-

sam pelo multiplicador quando os eletrodos são colocados em dois pontos distintos da superfície externa, ou um eletrodo é colocado na matéria cinzenta e o outro na superfície do crânio".

Foi Hans Berger (1873-1941) [6] que, em 1924, registrou o primeiro EEG de atividade do cérebro humano (ver Figura 3.3) utilizando um galvanômetro muito sensível. Criador do termo eletroencefalograma, Berger estudou as sucessivas posições do elemento móvel do galvanômetro, registradas numa folha de papel. Dessa forma, foi capaz de perceber os padrões resultantes dessas ondas cerebrais variando com o tempo. Berger notou que essas ondas não eram totalmente aleatórias e demonstravam certos padrões de periodicidade e regularidade e que, além disso, as ondas cerebrais dos indivíduos tinham alta amplitude e baixa frequência durante o sono e baixa amplitude e alta frequência quando acordados. Outra descoberta de Berger, e uma das mais interessantes, foi que o EEG era modificado durante um ataque epilético que causava um grande aumento da energia do sinal.

Apesar de sua importante contribuição à neurociência, o trabalho de Hans Berger demorou a ficar conhecido sendo aceito pela comunidade científica apenas uma década depois quando, em 1934, os ingleses Adrian e Matthews publicaram um artigo verificando suas descobertas.

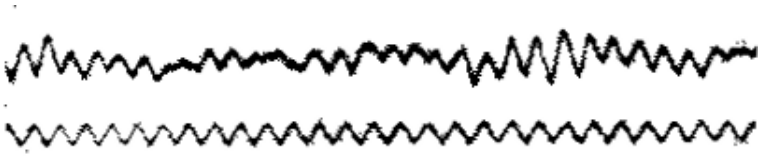

Figura 3.3: *O primeiro registro de um eletroencefalograma de um ser humano. A linha de baixo representa uma senoide utilizada como referência temporal. A linha de cima é o registro do EEG do filho mais novo de Hans Berger. De Berger, H. 1929. Arch. Psychiat.*

3.3 Artefatos

Durante uma coleta de EEG, os eletrodos posicionados sobre o couro cabeludo captam os potenciais elétricos emitidos pelo cérebro. Porém, não são apenas os sinais

produzidos pelo cérebro que são captados pelos eletrodos, outros sinais que podem ter origem da própria pessoa ou do ambiente externo podem ser captados causando uma interferência ou ruído nos potenciais registrados pelos eletrodos. Esses sinais que interferem no EEG são chamados de *artefatos* [7, 8] e são divididos em *artefatos biológicos* que são produzidos pelo próprio sujeito sob análise e *artefatos técnicos* que são produzidos pelo ambiente ou pelo equipamento em torno do experimento. Tais artefatos, que podem atrapalhar em demasia a análise e classificação dos dados de EEG, são divididos em tipos diversos com especificações distintas para cada um deles.

3.3.1 Artefatos Biológicos

EOG

Os artefatos eletro-oculográficos (EOG) surgem devido aos movimentos ou piscar dos olhos que causam uma mudança no campo elétrico ocular que existe devido a diferença de potencial entre a córnea e a parte de trás do olho. Os artefatos de EOG tem, em média, amplitudes 5 vezes maiores que os sinais de EEG causando, devido à proximidade aos eletrodos, uma interferência particularmente grande.

ECG

Os artefatos eletrocardiográficos (ECG) consequência da atividade elétrica do coração causada pela depolarização e repolarização que resulta de cada batida do coração. O ECG pode afetar o registro dos sinais de EEG e varia de acordo com a largura e comprimento do pescoço do sujeito sob observação, mas são mais fáceis de serem eliminados dos dados devido a sua periodicidade característica. As amplitudes dos sinais de ECG são 50 vezes maiores que os sinais de EEG, porém em geral causam pouca alteração graças a distância entre o cérebro e o coração.

EMG

Os artefatos eletromiográficos (EMG) são causados pelos potenciais elétricos gerados pelas células musculares quando estas estão eletricamente ou neurologicamente ativadas o que geralmente ocorre devido às contrações musculares. Os artefatos de EMG devem ser considerados durante o planejamento do experimento de forma a

evitar movimentos, principalmente da cabeça, por parte do sujeito sob observação. Na região da pele próxima ao músculo a amplitude do sinal de EMG é em média de $50\mu V$ (25 vezes maior que a amplitude do sinal de EEG), portanto os movimentos dos músculos próximos aos eletrodos podem alterar significativamente os sinais captados pelos eletrodos.

Outros Artefatos Biológicos

Também existem outros artefatos biológicos que podem interferir na qualidade do sinal de EEG, são eles: respiração, movimento da língua, movimento do maxilar e a deglutição.

3.3.2 Artefatos técnicos

Os artefatos técnicos são causados pelo ambiente externo e equipamentos com tecnologia insatisfatória. Eletrodos com impedância elevada podem fazer com que o equipamento funcione como um tipo de antena captando os sinais eletromagnéticos do ambiente. Em geral esses artefatos influenciam as bandas de frequência entre 50Hz e 60Hz e podem ser causados por diversos fatores como por exemplo as lâmpadas próximas ao equipamento, computadores, telefones celulares e a própria bateria do equipamento quando existir.

Capítulo 4

Interface Cérebro-Máquina

"Algum dia, para o bem ou para o mal[...] um ser humano poderá ser ligado a um computador muito avançado e, através deste, se conectar com um ou vários outros seres humanos. Pensamentos e sentimentos serão compartilhados na totalidade, sem a seletividade e dissimulação que a linguagem permite."

- Murray Gell-Mann, em seu livro "O quark e o jaguar", 1994.

4.1 O que é e como funciona?

Uma *Interface Cérebro-Máquina* (ICM) nada mais é do que um programa de computador capaz de interpretar dados provenientes da atividade cerebral e com isso produzir um comando para o computador ou máquina. Portanto, a ICM é composta de três partes fundamentais que cientistas de várias áreas buscam estudar e desenvolver:

- Monitorar e registrar a atividade cerebral: como já foi mencionado esse registro pode ser feito através dos dados de EEG, mas existem também outras tecnologias tais como o eletrocorticograma (ECoG) ou *functional magnetic resonance imaging* (fMRI), por exemplo.

- Interpretar os dados da atividade cerebral automaticamente em um computador: isso pode ser atingido utilizando-se de algoritmos de aprendizado estatístico ou inteligência artificial, que buscam encontrar e interpretar padrões e

características intrínsecos aos dados cerebrais, identificando assim, o provável pensamento do sujeito sob análise.

- Utilizar a resposta dada pelo algoritmo de aprendizado de máquina como entrada em um computador ou máquina com aplicabilidade relevante: esta parte é extremamente ampla e depende apenas da imaginação do projetista, alguns exemplos são jogos de computador e controle de próteses mecânicas.

4.2 Estado da Arte

Em 1929, Hans Berger, o primeiro cientista a registrar sinais de EEG de seres humanos, especulou sobre a possibilidade de ler pensamentos pelos padrões do eletroencefalograma com o uso de sofisticadas ferramentas matemáticas. Apesar da sugestão deixada por Berger, foi apenas recentemente que os primeiros cientistas passaram a investigar de fato os padrões do EEG e de outras técnicas de registro da atividade cerebral com vistas à construção de uma ICM.

Atualmente existem diversos grupos espalhados pelo mundo que buscam implementar uma ICM de forma eficiente utilizando diversos tipos de técnicas de coleta da atividade cerebral. Especificamente, existem dois tipos principais de ICM: as que utilizam técnicas não-invasivas e as que utilizam técnicas invasivas de registro da atividade cerebral. Particularizando esses dois tipos apresentam-se alguns exemplos de grupos ao redor do mundo trabalhando com ICM e que representam uma pequena amostra dos muitos grupos que existem atualmente trabalhando nesta área.

4.2.1 ICM não-invasivas

As ICM não-invasivas utilizam equipamentos de coleta que funcionam de forma externa ao corpo humano. A coleta de dados por EEG tem sido a abordagem mais utilizada para aplicação em ICM não-invasivas devido a sua praticidade, preço acessível, facilidade de uso e excelente resolução temporal, porém o EEG como ICM é muito suscetível à presença de ruídos. Além do EEG, magnetoencefalografia (MEG) e ressonância magnética (fMRI) tem sido utilizados com sucesso.

Na Universidade de Graz, na Alemanha, está localizado um dos grupos mais importantes na pesquisa sobre ICM via EEG. Criadores de uma das primeiras ICM com funcionamento em tempo real, a chamada *"Graz BCI"*, os pesquisadores estudam métodos de coleta e análise de sinais de EEG visando devolver a liberdade motora a pacientes tetraplégicos, criação de aparelhos para soletração e jogos de computador. Outro trabalho do grupo [9], consiste em uma ICM que permite ao usuário se locomover por uma cidade virtual pelo uso de imagética motora (imaginação de movimentos das mãos, dedos ou pés).

O grupo de Taipei [10], no Taiwan, utilizou uma técnica baseada em imagética motora que extrai dois tipos de características neurais chamadas de mapa contralateral e ipsilateral pelo uso de *análise de componentes independentes* (ICA) para implementar uma ICM. Seus resultados mostraram que o uso de ICA melhorava as taxas de classificação significativamente. O grupo também apresentou estudos relacionados à extração de características pelo uso de Wavelets contínuas e dimensão fractal para imagética motora.

Outro grupo que trabalha com ICM é o Berlin Brain-Computer Interface (BBCI) [11], na Alemanha. Uma das linhas de pesquisa mais importantes do grupo BBCI é a que apresenta técnicas para construção de uma ICM que não necessite de treinamento prévio por parte do sistema e do usuário e que não dependa da concentração exclusiva deste. Ou seja, o sujeito que utiliza a ICM não precisa se concentrar na tarefa de, por exemplo, mover um braço mecânico, podendo agir naturalmente pensando em outros assuntos.

4.2.2 ICM invasiva

Neste tipo de ICM os sinais produzidos pelo cérebro são registrados por eletrodos implantados diretamente no cérebro por meio de cirurgias. Devido a esta dificuldade, os experimentos que tratam sobre o assunto são realizados predominantemente em animais como ratos e macacos. Nesta linha de pesquisa não apenas os métodos de processamento de sinais são estudados, mas também, em grande parte, o desenvolvimento de eletrodos melhores e adequados ao contato com o tecido cerebral.

O grupo mais importante na área de ICM invasivas é o liderado por Miguel Nicole-

lis [12, 13, 14], cientista brasileiro considerado um dos 20 maiores cientistas do mundo no começo da década passada pela revista *Scientific American*. O grupo trabalha na Universidade de Duke, nos Estados Unidos, e estuda a atividade cerebral de animais como ratos e macacos. O principal objetivo dos pesquisadores é construir uma ICM capaz de devolver os movimentos a pessoas paraplégicas e tetraplégicas.

Em seu experimento mais famoso, Nicolelis e seus colegas demonstraram que primatas podem aprender a manipular objetos virtuais controlando um braço robótico, uma espécie de jogo virtual. Enquanto o primata joga o jogo sua atividade cerebral é monitorada pelos eletrodos implantados no cérebro e os padrões dos sinais característicos do movimento são encontrados. Em seguida, o braço robótico gradualmente deixa de funcionar de modo que o macaco percebe que consegue jogar apenas pelo pensamento, concretizando a ICM.

No início deste capítulo dividimos a construção de uma ICM em três partes fundamentais. O foco principal deste trabalho é justamente a segunda parte: o uso de algoritmos de *aprendizado estatístico* para identificar os padrões escondidos nos sinais de EEG. Desta forma, o próximo capítulo introduz e define o que se chama de aprendizado estatístico e também a técnica, chamada de *máquina de suporte vetorial* (*Support Vector Machine* (SVM)), aqui utilizada.

Capítulo 5

Aprendizado Estatístico

"Os silenciosos estatísticos mudaram nosso mundo;
não descobrindo novos fatos ou avanços técnicos,
mas transformando as formas pelas quais
pensamos, realizamos experimentos
e formamos nossas opiniões..."

- Ian Hacking.

A estatística é a arte de interpretar, probabilisticamente, números e dados. Com o advento da internet e dos computadores modernos esses dados, que antigamente eram apenas pequenas tabelas de números, se transformaram em arquivos enormes que dificultaram o uso das técnicas atuais. Surgiu então a necessidade de se criar algoritmos e ferramentas computacionais estatísticas capazes de lidar com essas imensas bases de dados encontrando padrões e características escondidos em meio aos números.

Dada esta necessidade, surgiram, desenvolvidas por estatísticos, engenheiros e programadores, as técnicas conhecidas como *aprendizado estatístico* (AE) ou *aprendizado de máquina*. Pode-se dizer que as técnicas de AE revolucionaram e continuam revolucionando o mundo pois as mais diversas áreas de conhecimento utilizam o AE para desenvolver novas aplicações e tecnologias, como por exemplo:

- Reconhecimento de digitais para sistemas de segurança.
- Reconhecimento facial e de voz nos telefones modernos.

- Sistemas de busca na internet, como o Google.
- Identificação de tendências na economia.
- Identificação de *spams* pelos servidores de e-mails.
- Diagnósticos médicos.
- Detecção de fraude de cartão de crédito.
- Jogos de computador.
- E claro, interfaces cérebro máquina.

Percebe-se, portanto, a importância e aplicabilidade do AE. Basicamente, os problemas de aprendizado se dividem em *supervisionados* e *não-supervisionados*. No aprendizado supervisionado utiliza-se um banco de dados para *treinar* um algoritmo de classificação (treinar é o mesmo que estimar os parâmetros ótimos do modelo) com o objetivo de predizer a categoria ou valor de alguma medida para um novo conjunto de dados de entrada. Já no aprendizado não-supervisionado, este banco de dados inicial de treinamento não existe, de modo que o objetivo é descrever as associações e padrões presentes em um conjunto de dados.

Neste estudo utilizamos técnicas de aprendizado *supervisionado* com foco na classificação. Dentre essas técnicas estão os chamados *classificadores* que são algoritmos que buscam, no banco de dados, *características* que possam diferenciar elementos do banco de dados entre duas ou mais *classes*. Por exemplo, imagine um banco de dados com medidas de comprimento e largura de diversas notas de cem reais, e um sistema com a finalidade de identificar notas falsas. Neste caso, o sistema é um *classificador* que utiliza as *características* (comprimento e largura) para classificar notas em uma de duas *classes* (notas legítimas e notas falsas) [15].

Em diversas situações, como no caso dos sinais de EEG, os bancos de dados não possuem características adequadas para fazer uma classificação satisfatória utilizando-se os algoritmos conhecidos. Neste caso, é necessário utilizar técnicas auxiliares que servem para fazer a chamada *extração de características*, ou seja, técnicas que produzem um novo banco de dados com características adequadas ao bom funcionamento do classificador. Portanto, define-se:

Definição: Denota-se por *característica* quaisquer valores, estatísticas, padrões ou variáveis utilizadas para discriminar duas ou mais classes.

Definição: Denota-se por *extração de características* quaisquer técnicas ou transformações que, aplicadas a um determinado conjunto de dados, produzem um outro conjunto formado por características.

Assim, é importante exibir uma revisão sobre como tem sido feita a extração de características para classificação de dados de eletroencefalografia e também quais os classificadores que tem sido utilizados. Além disso, encontra-se neste capítulo a descrição do classificador que foi utilizado neste trabalho: a máquina de suporte vetorial. É imprescindível lembrar que o foco principal deste trabalho é justamente expor um novo algoritmo para fazer a seleção automática de características em sinais de EEG e, portanto, um capítulo a parte é reservado para sua apresentação.

5.1 Extração de Características para classificação de sinais de EEG

A extração de características é, provavelmente, a parte mais importante no processo de classificação de dados de EEG. Escolher as características mais adequadas ao problema é de suma importância para o bom desempenho do classificador. Para uma aplicação em ICM é fundamental que as técnicas utilizadas sejam capazes de lidar com os ruídos presentes no sinal e com a dimensionalidade elevada do banco de dados. Além disso, seria interessante que o algoritmo utilizado para extração de características fosse rápido o suficiente para permitir sua aplicação em tempo real.

Não existe até agora um método padronizado para extração de características dos sinais de EEG. De fato, são muitas as técnicas apresentadas na literatura e a ausência de trabalhos que as comparem inviabiliza a definição de qual técnica se sobressai as demais. Para exemplificar, foram utilizadas na literatura as técnicas de Discrete Wavelet Transform (DWT) [16, 17, 18], amplitude dos sinais [19], técnicas de agrupamento [20], autoregressive (AR) e adaptive autoregressive (AAR) parameters [21, 22], power spectral density (PSD) [23], entre outros.

Fica clara, portanto, a dificuldade de se escolher as características a serem uti-

lizadas. Por isso, justifica-se a proposta deste trabalho de apresentar um algoritmo que seja capaz de testar vários tipos de características e selecionar automaticamente as melhores e mais adequadas para a classificação. A seguir, são apresentadas as técnicas de extração de características que serão utilizadas neste trabalho.

5.1.1 Periodograma

A análise de frequências de Fourier [24] é uma ferramenta muito importante no processamento de sinais e o periodograma e a densidade espectral são dois dos seus subprodutos. A *densidade espectral* de uma série $\{X_t\}$ é a função $f(.)$ definida por

$$f(\lambda) = \frac{1}{2\pi} \sum_{h=-\infty}^{\infty} e^{-ih\lambda}\gamma(h), \qquad -\infty < \lambda < \infty, \tag{5.1}$$

onde $e^{i\lambda} = \cos(\lambda) + i\,\mathrm{sin}(\lambda)$ e $i = \sqrt{-1}$, além disso

$$\gamma(k) = \int_{-\pi}^{\pi} e^{ik\lambda} f(\lambda) d\lambda = \int_{-\pi}^{\pi} \cos(k\lambda) f(\lambda) d\lambda. \tag{5.2}$$

O *periodograma* mostra como a variância de uma série temporal é distribuída pela frequência, e para introduzi-lo considere o vetor complexo:

$$\mathbf{x} = (x_1, x_2, ..., x_n)' \in \mathbb{C}^n \tag{5.3}$$

onde $\mathbb{C}^n$ denota o conjunto de todos os vetores coluna com componentes complexos. Seja também

$$\omega_k = \frac{2\pi k}{n}, \qquad k = -\left\lfloor \frac{n-1}{2} \right\rfloor, ..., \left\lfloor \frac{n}{2} \right\rfloor, \tag{5.4}$$

onde $\lfloor y \rfloor$ representa o maior inteiro menor ou igual a y. Denota-se o conjunto F_n dos valores ω_k por *Frequências de Fourier* associadas ao tamanho n. Introduz-se também os vetores

$$\mathbf{e}_k = \frac{1}{\sqrt{n}}(e^{i\omega_k}, e^{2i\omega_k}, ..., e^{ni\omega_k})', \qquad k = -\left\lfloor \frac{n-1}{2} \right\rfloor, ..., \left\lfloor \frac{n}{2} \right\rfloor. \tag{5.5}$$

Note que $\mathbf{e_1}, ..., \mathbf{e_n}$ são ortonormais pois

$$\mathbf{e_j}^* \mathbf{e_k} = \begin{cases} 1, & \text{se } j = k \\ 0, & \text{se } j \neq k \end{cases} \tag{5.6}$$

onde $\mathbf{e_j}^*$ denota o vetor de conjugados complexos do vetor $\mathbf{e_j}$. Isso implica que $\{\mathbf{e_1}, ..., \mathbf{e_n}\}$ forma uma base de $\mathbb{C}^n$, ou seja, qualquer vetor $\mathbf{x} \in \mathbb{C}^n$ pode ser expresso por

$$\mathbf{x} = \sum_{k=-\lfloor (n-1)/2 \rfloor}^{\lfloor n/2 \rfloor} a_k \mathbf{e}_k. \tag{5.7}$$

Os coeficientes a_k podem ser encontrados utilizando 5.6 e 5.7:

$$a_k = \mathbf{e_k}^* \mathbf{x} = \frac{1}{\sqrt{n}} \sum_{t=1}^{n} x_t e^{-it\omega_k} \tag{5.8}$$

A sequência $\{a_k\}$ é chamada *transformada de Fourier discreta* da sequência $\{x_1, ..., x_n\}$.

Desta forma pode-se reescrever

$$x_t = \sum_{k=-\lfloor (n-1)/2 \rfloor}^{\lfloor n/2 \rfloor} a_k(\cos(\omega_k t) + i \sin(\omega_k t)), \qquad t = 1, ..., n, \tag{5.9}$$

portanto, pode-se representar uma série temporal em termos da frequência em componentes de senos e cossenos.

Finalmente, define-se o periodograma:

$$I_n(\lambda) = \frac{1}{n} \left| \sum_{t=1}^{n} x_t e^{-it\lambda} \right|^2. \tag{5.10}$$

Observe que no conjunto de frequências de Fourier temos $I_n(\omega_k) = |a_k|^2$ e portanto

$$\sum_{t=1}^{n} |x_t|^2 = \mathbf{x}^* \mathbf{x} = \sum_{k=-\lfloor (n-1)/2 \rfloor}^{\lfloor n/2 \rfloor} |a_k|^2 = \sum_{k=-\lfloor (n-1)/2 \rfloor}^{\lfloor n/2 \rfloor} I_n(\omega_k) \tag{5.11}$$

logo o valor do periodograma na frequência ω_k é sua contribuição para a soma de quadrados ou energia da série $\{x_t\}$.

Como para n grande os valores do periodograma para as frequências de Fourier são aproximadamente independentes com variâncias mudando apenas suavemente entre os intervalos de frequência, pode-se construir um estimador consistente da densidade espectral $f(\lambda)$ suavizando-se o periodograma com alguma técnica de suavização qualquer como médias móveis por exemplo.

5.1.2 Transformada Wavelet Contínua (CWT)

Wavelets tipicamente utilizam bases ortonormais para representar uma função. São utilizadas principalmente por conseguirem representar um sinal em termos do tempo

e da frequência, enquanto que bases similares como as de Fourier só são capazes de representar as frequências.

A CWT $W(s,t)$ de um sinal $f(t)$ é definida por

$$W(s,t) = \frac{1}{\sqrt{s}} \int_{-\infty}^{\infty} f(\tau)\psi\left(\frac{\tau - t}{s}\right) dt \tag{5.12}$$

onde t denota um deslocamento de tempo, s denota a escala e ψ é a função wavelet, que deve integrar para zero.

Deste modo a CWT é um tipo de covariância entre a função original e uma função wavelet. A escolha de uma função wavelet $\psi(t)$ na Eq.(5.12) depende do tipo de característica que se deseja extrair. Neste trabalho, assim como em [33], utilizaremos a wavelet "chapéu mexicano" pela segunda derivada da densidade gaussiana (ver Figura 5.1), ou

$$\psi(t) = (1 - t^2/\sigma^2)\exp(-t^2/(2\sigma^2)) \tag{5.13}$$

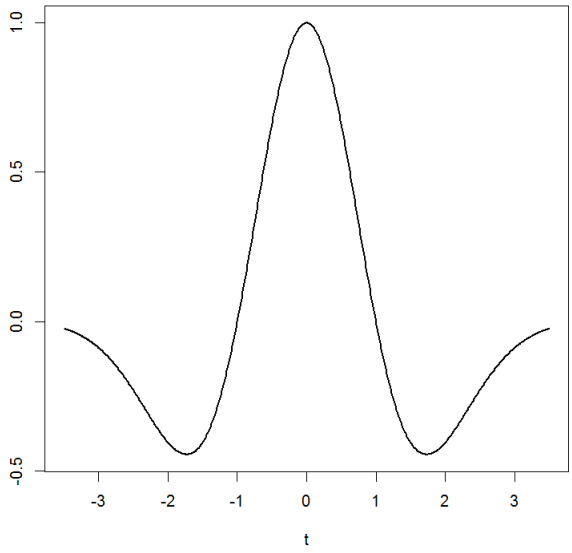

Figura 5.1: *Wavelet "chapéu mexicano".*

5.1.3 Janelamento

Sejam $\{x_t\}$, $t = 1, 2, ..., n$ uma sequência ou série temporal, o tamanho da janela $w > 0 \in \mathbb{Z}$, e a função $g : \mathbb{R}^w \to \mathbb{R}$ então define-se o *janelamento* como:

$$S_i = g(x_i, x_{i+1}, ..., x_{i+w-1}), \qquad i = 1, 2, ..., n - w + 1. \tag{5.14}$$

Portanto a sequência $\{S_t\}$, $t = 1, 2, ..., n - w + 1$, é formada pelas características extraídas da série original $\{x_t\}$, $t = 1, ..., n$. A função g pode, por exemplo, ser alguma estatística ou característica específica como a média, variância ou energia da série $\{x_t\}$.

Vale notar que se $g(x_i, x_{i+1}, ..., x_{i+w-1}) = (x_i + x_{i+1} + ... + x_{i+w-1})/w$, ou seja a média amostral, então o janelamento produz a mesma sequência que a técnica de médias móveis que pode ser utilizada para suavizar o periodograma, apresentado na seção 5.1.1.

5.1.4 Componentes Principais como extração de características

Uma análise de componentes principais é utilizada para explicar a estrutura de variância-covariância de um conjunto de variáveis por um conjunto menor de variáveis formadas por combinações lineares das variáveis originais. Em geral, em bancos de dados cujas variáveis são fortemente correlacionadas - como no caso de dados de EEG - a técnica de componentes principais é muito útil na redução da dimensão do problema. Esse tipo de análise, em muitos casos, também é capaz de revelar relações inesperadas ou difíceis de se identificar entre as variáveis.

Considere um sistema com p variáveis aleatórias $X_1, X_2, ...X_p$. Seja Σ a matriz de variâncias-covariâncias do vetor aleatório $\mathbf{X}' = [X_1, X_2, ..., X_p]$. Considere também as combinações lineares

$$\begin{aligned}
Y_1 &= \mathbf{a_1'X} &&= a_{11}X_1 + a_{12}X_2 + ... + a_{1p}X_p \\
Y_2 &= \mathbf{a_2'X} &&= a_{21}X_1 + a_{22}X_2 + ... + a_{2p}X_p \\
... & && ... \\
Y_p &= \mathbf{a_p'X} &&= a_{p1}X_1 + a_{p2}X_2 + ... + a_{pp}X_p.
\end{aligned}$$

Neste contexto tem-se que:

$$\mathrm{Var}(Y_i) = \mathbf{a_i'}\Sigma\mathbf{a_i} \tag{5.15}$$

$$\mathrm{Cov}(Y_i, Y_j) = \mathbf{a_i'}\Sigma\mathbf{a_j}. \tag{5.16}$$

Chamam-se componentes principais as p combinações lineares $Y_1, Y_2, ...Y_p$, não correlacionadas, com variâncias (dadas em 5.15) decrescentes e máximas.

Portanto, a primeira componente principal é a combinação linear com maior variância possível, ou seja, precisamos maximizar $\mathrm{Var}(Y_1) = \mathbf{a_1'}\Sigma\mathbf{a_1}$. É claro que $\mathrm{Var}(Y_1)$ cresce apenas aumentando a norma de $\mathbf{a_1}$, desse modo adiciona-se a restrição de que $\mathbf{a_1'a_1} = 1$ ao problema. A segunda componente principal é a combinação linear não correlacionada com a primeira com maior variância possível, e assim por diante.

Prova-se facilmente [25] que a solução é $\mathbf{a_1} = \mathbf{e_1}$, $\mathbf{a_2} = \mathbf{e_2}$, ... , $\mathbf{a_p} = \mathbf{e_p}$, onde $(\lambda_1, \mathbf{e_1})$, $(\lambda_2, \mathbf{e_2})$, ... , $(\lambda_p, \mathbf{e_p})$ são os pares de autovalores-autovetores da matriz de variâncias-covariâncias Σ onde $\lambda_1 \geq \lambda_2 \geq ... \geq \lambda_p \geq 0$. Então a i-ésima componente principal é dada por

$$Y_i = \mathbf{e_i'X}. \tag{5.17}$$

Prova-se também que $\sum_{i=1}^{p} \mathrm{Var}(X_i) = \sum_{i=1}^{p} \mathrm{Var}(Y_i)$, ou seja, as componentes principais representam toda a variabilidade do sistema, porém podemos diminuir a dimensão do problema escolhendo as k primeiras componentes principais que representem a maior parte da variabilidade.

Para utilização como extração de características pode-se utilizar o janelamento nas componentes principais. Além disso pode-se utilizar os *loadings*, que são os valores de a_{ij} como características.

5.2 Classificadores para sinais de EEG

Após extrair as características é hora de treinar um *classificador*. Da mesma forma que na fase de extração de características temos aqui o problema da falta de padronização, ou seja, não há um consenso sobre qual é o melhor classificador para dados de EEG. Vários classificadores já foram utilizados na literatura, tais como artificial neural network (ANN) [16, 17, 26, 27], mixture of expert (ME) model [17], linear discriminant

analysis [18], support vector machine (SVM) [18, 28], decision trees [29], least squares support vector machine (LS-SVM) [20, 30], hidden markov models (HMM) [23], entre outros. Para uma revisão mais completa das técnicas utilizadas ver [31].

5.3 Máquina de Suporte Vetorial (SVM)

A técnica de Máquina de Suporte Vetorial, do inglês *Support Vector Machine* (SVM), surgiu em 1992 e é uma técnica relativamente nova mas que vem sendo muito utilizada recentemente. A técnica de SVM é utilizada tanto em problemas de regressão quanto em problemas de classificação e reconhecimento de padrões.

Os dados de EEG produzem, em geral, um conjunto de características de dimensionalidade elevada em comparação com o número de repetições disponíveis de cada classe. Este fato torna a técnica de SVM especialmente adequada para uso na classificação de dados de EEG já que é robusta à dimensionalidade elevada dos dados.

Nesta seção serão apresentadas algumas definições e conceitos matemáticos importantes para a definição da técnica de SVM, além da apresentação da mesma. O estudo do SVM será dividido no caso linear separável, linear e não-linear.

5.3.1 Caso de dados binários e linearmente separáveis

Para iniciar a discussão sobre Máquinas de Suporte Vetorial (SVM), abordaremos o caso particular em que se deseja separar duas classes que são separáveis por um hiperplano

$$X^T\beta + \beta_0 = 0. \tag{5.18}$$

Logicamente, como se pode visualizar na Figura 5.2, esse hiperplano não é único e, portanto, é necessário determinar um critério que nos permita otimizar essa separação. Desse modo, escolhe-se o hiperplano que deixe a maior margem (M) de distância possível entre as classes, como pode-se visualizar na Figura 5.3.

Definições e conceitos matemáticos

Seja o hiperplano L dado pela equação

$$g(x) = X^T\beta + \beta_0 = 0, \qquad X \in \mathsf{R}^p \tag{5.19}$$

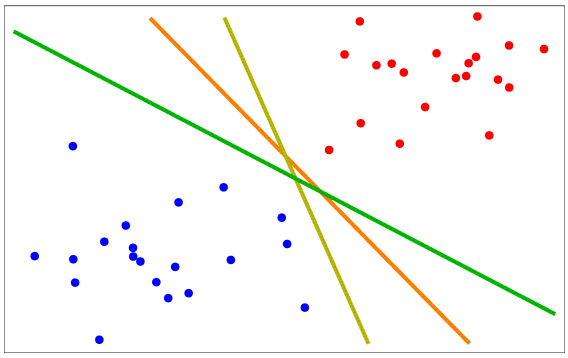

Figura 5.2: *Um exemplo de duas classes separáveis. Três exemplos de hiperplanos (retas no caso bidimensional) são mostrados. É claro que existem infinitos hiperplanos que podem separar esses dados e portanto é necessário um critério para escolher o hiperplano ótimo.*

onde $\beta \in \mathbb{R}^p$ and $\beta_0 \in \mathbb{R}$ então a distância de qualquer ponto $x_i \in \mathbb{R}^p$ ao hiperplano L é dada por

$$D = \frac{1}{\|\beta\|}\left(x_i^T\beta + \beta_0\right) = \frac{g(x_i)}{\|g'(x_i)\|}. \tag{5.20}$$

Sejam $x_i \in \mathbb{R}^p$, $i = 1, 2, ..., N$, os vetores (*feature vectors*) de um conjunto de treinamento (*training set*) X, classificados em uma de duas classes W_1 ou W_{-1} de modo que $y_i g(x_i) > 0$ se x_i for corretamente classificado em sua classe pelo hiperplano L, sendo y_i uma variável indicadora definida por:

$$y_i = \begin{cases} 1 & \text{se} \quad x_i \in W_1 \\ -1 & \text{se} \quad x_i \in W_{-1} \end{cases} \tag{5.21}$$

Hiperplano Ótimo

A técnica de SVM busca o hiperplano que separe as classes e maximize a margem de distância entre as classes e os hiperplanos. Para definir matematicamente a margem M, considere os pares, vetor de característica e variável indicadora (x_i, y_i), onde $i \in I$, sendo $I = \{1, 2, ..., N\}$. Seja $j = \{-1, 1\}$ e considere, também, os conjuntos $I_j = \{i \in I : y_i = j\}$. Deste modo defina

$$D_j = \{d_i : i \in I_j\}, \tag{5.22}$$

onde d_i é a distância, definida na Eq.(5.20), entre o vetor x_i e o hiperplano L. Seja $M_j = min(D_j)$ a menor distância entre os vetores da classe W_j e o hiperplano L.

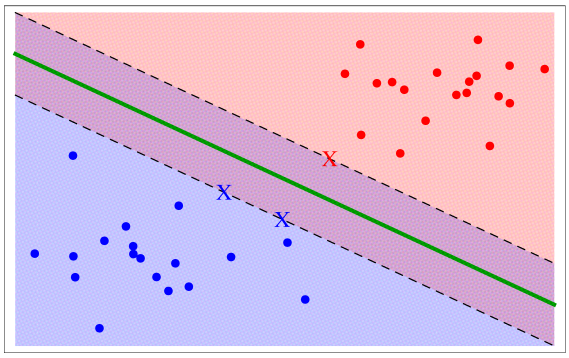

Figura 5.3: *O mesmo exemplo de duas classes separáveis. O hiperplano em linha contínua (reta no caso bidimensional) é o hiperplano ótimo. As retas pontilhadas definem a margem que é máxima neste caso. Os pontos marcados com um "X" são os chamados vetores suporte.*

Define-se, finalmente, a margem M como

$$M = \frac{M_1 + M_{-1}}{2}. \tag{5.23}$$

O objetivo é escolher o hiperplano que maximiza a margem M de modo que $M = M_1 = M_{-1}$, isto é, escolher o hiperplano de maior margem de distância possível entre as classes. É claro que nenhum ponto x_i tem distância menor do que M em relação ao hiperplano, portanto em termos matemáticos tem-se, usando a equação (5.20),

$$\left| \frac{1}{\|\beta\|} \left(x_i^T \beta + \beta_0 \right) \right| \geq M \tag{5.24}$$

e como as duas classes são separáveis, temos equivalentemente

$$\frac{1}{\|\beta\|} y_i \left(x_i^T \beta + \beta_0 \right) \geq M \Rightarrow$$

$$\Rightarrow y_i \left(x_i^T \beta + \beta_0 \right) \geq \|\beta\| M.$$

Para qualquer β e β_0 que satisfaçam essa equação, qualquer múltiplo positivo destes também a satisfaz. Logo, é possível escolher arbitrariamente $\|\beta\| = \frac{1}{M}$ de modo que

$$y_i \left(x_i^T \beta + \beta_0 \right) \geq 1 \tag{5.25}$$

Desta forma, o problema se resume ao problema de otimização

$$\begin{aligned} &\underset{\beta,\beta_0}{\text{Max}} \quad M \\ &\text{sujeito a} \quad \frac{1}{\|\beta\|} y_i \left(x_i^T \beta + \beta_0\right) \geq M, \qquad i \in I \end{aligned}$$

ou de mesmo modo

$$\begin{aligned} &\underset{\beta,\beta_0}{\text{Min}} \quad \frac{1}{2}\|\beta\|^2 \\ &\text{sujeito a} \quad y_i \left(x_i^T \beta + \beta_0\right) \geq 1, \qquad i \in I \end{aligned} \tag{5.26}$$

Este é um problema de otimização convexa, então, usando multiplicadores de Lagrange, obtém-se a função primal

$$L_p = \frac{1}{2}\|\beta\|^2 - \sum_{i=1}^{N} \alpha_i \left[y_i \left(x_i^T \beta + \beta_0\right) - 1\right], \tag{5.27}$$

onde α_i são os multiplicadores de Lagrange. Derivando (5.27) em relação a β e β_0 obtém-se

$$\beta = \sum_{i=1}^{N} \alpha_i y_i x_i, \tag{5.28}$$

$$\sum_{i=1}^{N} \alpha_i y_i = 0. \tag{5.29}$$

Substituindo (5.28) e (5.29) em (5.27), encontra-se a função dual de Wolfe:

$$L_D = \sum_{i=1}^{N} \alpha_i - \frac{1}{2} \sum_{i=1}^{N} \sum_{k=1}^{N} \alpha_i \alpha_k y_i y_k x_i^T x_k, \qquad \alpha_i \geq 0 \tag{5.30}$$

A solução é obtida maximizando L_D, um problema simples de otimização convexa que deve satisfazer (5.28), (5.29) e (5.30) além de satisfazer as condições de Karush-Kuhn-Tucker (KKT):

$$\alpha_i \left[y_i \left(x_i^T \beta + \beta_0\right) - 1\right] = 0, \qquad i \in I \tag{5.31}$$

Encontra-se, desse modo, o hiperplano ótimo para separar as duas classes. É importante notar que a equação (5.31) implica que $\alpha_i = 0$ ou que $y_i \left(x_i^T \beta + \beta_0\right) = 1$, ou seja, α_i só é diferente de zero (em geral) se e somente se o ponto x_i estiver sobre

uma das margens e portanto a uma exata distância M do hiperplano L respeitando a restrição $y_i\left(x_i^T\beta+\beta_0\right)=1$. Os pontos que respeitam essa restrição (ver Figura 5.3) são chamados de Vetores Suporte (Support Vectors) e são pontos muito importantes já que o parâmetro β é determinado exclusivamente devido a eles.

Para finalizar define-se o classificador

$$\hat{G}(x)=sinal(\hat{f}(x)), \tag{5.32}$$

$$\text{onde } \hat{f}(x)=x^T\hat{\beta}+\hat{\beta}_0 \tag{5.33}$$

para a classificação de observações futuras.

5.3.2 Caso de dados binários não separáveis

Na seção 5.3.1 foi definido o hiperplano ótimo na separação de duas classes linearmente separáveis. Na prática, entretanto, muitas vezes os dados não são linearmente separáveis e a matemática descrita até aqui passa a não ser mais válida.

A solução para este problema é simplesmente permitir observações entre as margens e observações incorretamente classificadas pelo hiperplano. Desse modo encontrar-se-á três tipos de pontos:

i) Pontos corretamente classificados fora ou em cima das margens, ou seja:

$$\frac{1}{\|\beta\|}y_i\left(x_i^T\beta+\beta_0\right)\geq M$$

ii) Pontos corretamente classificados residindo entre as margens:

$$0<\frac{1}{\|\beta\|}y_i\left(x_i^T\beta+\beta_0\right)<M$$

iii) Pontos incorretamente classificados:

$$\frac{1}{\|\beta\|}y_i\left(x_i^T\beta+\beta_0\right)\leq 0$$

Esses três casos podem ser resumidos em um único caso ao se introduzir variáveis de folga ξ_i da seguinte maneira:

$$\frac{1}{\|\beta\|}y_i\left(x_i^T\beta+\beta_0\right)\geq M(1-\xi_i). \tag{5.34}$$

Desse modo, para $\xi_i = 0$ tem-se pontos do primeiro tipo, para $0 < \xi_i < 1$ tem-se pontos do segundo tipo e para $\xi_i \geq 1$ tem-se pontos do terceiro tipo. Além disso, escolhe-se β e β_0 em (5.34) convenientemente de modo que $\|\beta\| = \frac{1}{M}$ e portanto obtém-se

$$y_i \left(x_i^T \beta + \beta_0\right) \geq 1 - \xi_i, \qquad i \in I. \tag{5.35}$$

As variáveis ξ_i são conhecidas como variáveis de folga e sua representação gráfica pode ser vista na Figura 5.4. Logicamente, o intuito da técnica é ter o mínimo de pontos do tipo (ii) e do tipo (iii) e para isso minimiza-se a soma $\sum_{i=1}^{N} \xi_i$ que é a quantidade total pela qual pontos estão proporcionalmente do lado errado da margem. Levando essas situações em conta, pode-se generalizar o problema dado em (5.26) por

$$\underset{\beta,\beta_0,\xi_i}{\text{Min}} \quad \frac{1}{2}\|\beta\|^2 + C\sum_{i=1}^{N} \xi_i \tag{5.36}$$

sujeito a

$$y_i \left(x_i^T \beta + \beta_0\right) \geq 1 - \xi_i, \qquad i \in I$$

$$\xi_i \geq 0, \qquad i \in I.$$

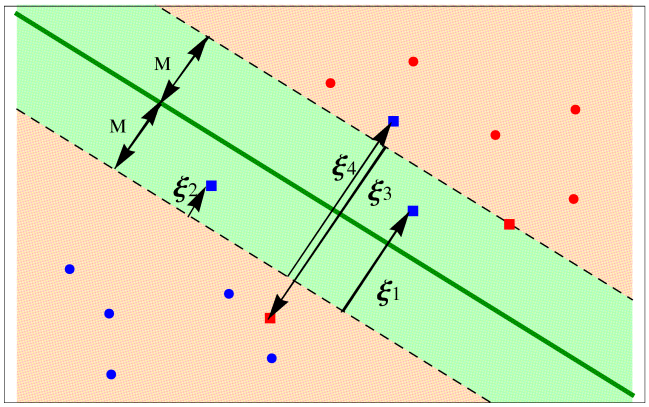

Figura 5.4: *Um exemplo de duas classes não separáveis. O hiperplano em linha contínua (reta no caso bidimensional) é o hiperplano ótimo. As linhas pontilhadas delimitam as margens de largura M. Os quadrados são os vetores suporte (Support Vectors). Os vetores representados por ξ_i são vetores cuja norma é proporcional (M vezes) à variável de folga correspondente ao seu respectivo ponto x_i.*

A constante C é previamente escolhida e determina a influência dos dois termos no problema de minimização. Note que para C grande a influência das variáveis ξ_i

se torna maior resultando assim em uma margem menor. Se existirem duas classes linearmente separáveis pode-se utilizar as equações dadas em (5.36) fazendo $C \to \infty$.

Cálculo do hiperplano ótimo

Para resolver este problema será novamente utilizada a técnica dos multiplicadores de Lagrange. Do problema definido em (5.36) obtém-se a função primal de Lagrange

$$L_p = \frac{1}{2}\|\beta\|^2 + C\sum_{i=1}^{N}\xi_i - \sum_{i=1}^{N}\alpha_i\left[y_i\left(x_i^T\beta + \beta_0\right) - (1-\xi_i)\right] - \sum_{i=1}^{N}\lambda_i\xi_i \tag{5.37}$$

onde $\alpha_i \geq 0$ e $\lambda_i \geq 0$ são os multiplicadores de Lagrange. Derivando em relação a β, β_0 e ξ_i, e igualando a zero, encontra-se

$$\beta = \sum_{i=1}^{N}\alpha_i y_i x_i, \tag{5.38}$$

$$\sum_{i=1}^{N}\alpha_i y_i = 0, \tag{5.39}$$

$$\alpha_i = C - \lambda_i, \tag{5.40}$$

com $i \in I$. Substituindo (5.38), (5.39) e (5.40) em (5.37) obtém-se a função dual de Wolfe

$$L_D = \sum_{i=1}^{N}\alpha_i - \frac{1}{2}\sum_{i=1}^{N}\sum_{k=1}^{N}\alpha_i\alpha_k y_i y_k x_i^T x_k. \tag{5.41}$$

A solução é obtida maximizando L_D, um problema simples de otimização convexa que deve satisfazer as condições (5.38), (5.39), (5.40) e $0 \leq \alpha_i \leq C$ além de satisfazer as condições de Karush-Kuhn-Tucker (KKT):

$$\alpha_i\left[y_i\left(x_i^T\beta + \beta_0\right) - (1-\xi_i)\right] = 0, \tag{5.42}$$

$$\lambda_i\xi_i = 0, \tag{5.43}$$

$$y_i\left(x_i^T\beta + \beta_0\right) \geq (1-\xi_i), \tag{5.44}$$

para $i \in I$. Chega-se então ao estimador de β que é dado por

$$\hat{\beta} = \sum_{i=1}^{N}\hat{\alpha}_i y_i x_i. \tag{5.45}$$

Desse modo, determinou-se o hiperplano ótimo para separar as duas classes. É importante notar novamente que a equação (5.42) implica que $\alpha_i = 0$ ou que $y_i\left(x_i^T\beta + \beta_0\right) = 1 - \xi_i$ ou seja, α_i é diferente de zero (em geral) se e somente se $y_i\left(x_i^T\beta + \beta_0\right) = 1 - \xi_i$, e portanto como $\xi_i \geq 0$, α_i só é diferente de zero se seu ponto x_i correspondente estiver em cima de uma das margens, entre as margens, ou se tiver sido incorretamente classificado (ver Figura 5.4).

Os pontos cujos respectivos α_is são diferentes de zero são chamados de vetores suporte (Support Vectors) e são de suma importância já que o parâmetro β é determinado exclusivamente devido a esses pontos como pode-se notar na equação (5.45). Essa é uma característica muito interessante desse classificador pois ele foca atenção nos pontos que realmente importam.

Outra motivação para o uso de SVM é que nenhuma suposição sobre a distribuição dos dados é necessária, já que o que é levado em conta é apenas a distância entre as classes.

5.3.3 Classificador não linear

Considere duas classes que não são linearmente separáveis. Suponha que exista um mapeamento

$$x \in \mathsf{R}^p \rightarrow y \in \mathsf{R}^k \tag{5.46}$$

que transforme o espaço dos dados em um espaço k-dimensional onde as classes possam ser satisfatoriamente separáveis por um hiperplano. Se tal mapeamento existir pode-se utilizar as técnicas vistas na seção anterior para encontrar o hiperplano ótimo que separe essas classes no espaço k-dimensional, resultando em uma função não linear no espaço original.

À primeira vista essa ideia não parece ser muito viável já que aumentando as dimensões dos dados, o processo computacional de estimação se tornaria, muitas vezes, inviável ou mesmo impossível. Porém, existe uma bela propriedade da técnica de SVM que permite aumentar muito o número de dimensões (tornando-as infinitas se necessário) sem adicionais demandas computacionais.

SVM e Kernels

Nas seções anteriores foi definido nosso problema de minimização para o cálculo do hiperplano ótimo. Note então na equação (5.41) que todos os cálculos envolvidos sobre os dados estão na forma do produto interno $x_i^T x_i$. Da mesma forma, para o novo espaço k-dimensional, os cálculos envolvidos estarão na forma de produto interno. O fato interessante é que para alguns mapeamentos esses produtos internos podem ser calculados como função do produto interno original $x_i^T x_i$.

Teorema. Teorema de Mercer. Sejam $\mathbf{x} \in \mathsf{R}^p$ e um mapeamento ϕ,

$$\mathbf{x} \to \phi(\mathbf{x}) \in H \tag{5.47}$$

onde H é um espaço Euclideano. A operação de produto interno tem a representação equivalente

$$K(\mathbf{x}, \mathbf{z}) = \sum_i \phi_i(\mathbf{x})\phi_i(\mathbf{z}) \tag{5.48}$$

onde $\phi_i(\mathbf{x})$ é o i-ésimo componente do mapeamento $\phi(\mathbf{x})$ e $K(\mathbf{x}, \mathbf{z})$ é uma função simétrica que satisfaz

$$\int K(\mathbf{x}, \mathbf{z})g(\mathbf{x})g(\mathbf{z})d\mathbf{x}d\mathbf{z} \geq 0 \tag{5.49}$$

para qualquer $g(\mathbf{x})$, $\mathbf{x} \in \mathsf{R}^p$ tais que

$$\int [g(\mathbf{x})]^2 d\mathbf{x} < +\infty \tag{5.50}$$

O inverso também é verdadeiro, ou seja, para qualquer função $K(\mathbf{x}, \mathbf{z})$ que satisfaça (5.49) e (5.50) existe um espaço onde $K(\mathbf{x}, \mathbf{z})$ define um produto interno. Tais funções são conhecidas como *Kernels* e o espaço H como "*Reproducing Kernel Hilbert Space*" (RKHS).

Alguns dos Kernels mais utilizados são os:

i) Polinomiais

$$K(\mathbf{x}, \mathbf{z}) = (\mathbf{x^T z} + 1)^q, \qquad q > 0 \tag{5.51}$$

ii) Radiais ou Gaussianos

$$K(\mathbf{x}, \mathbf{z}) = \exp\left(-\frac{||\mathbf{x} - \mathbf{z}||^2}{\sigma^2}\right) \tag{5.52}$$

iii) Sigmoide

$$K(\mathbf{x}, \mathbf{z}) = \tanh(\gamma \mathbf{x^T z} + \lambda) \tag{5.53}$$

para valores de γ e λ que satisfaçam as condições do teorema de Mercer.

Portanto, adotado um Kernel conveniente, o problema de otimização dado em (5.41) se torna

$$\begin{aligned} &\underset{\alpha}{\text{Max}} \quad \sum_{i=1}^{N} \alpha_i - \frac{1}{2} \sum_{i=1}^{N} \sum_{k=1}^{N} \alpha_i \alpha_k y_i y_k K(\mathbf{x_i}, \mathbf{x_k}) \\ &\text{sujeito a} \quad 0 \leq \alpha_i \leq C, \qquad i \in I \\ &\qquad\qquad \sum_i \alpha_i y_i = 0 \end{aligned} \tag{5.54}$$

A característica que deve ser ressaltada é que a complexidade computacional envolvida na técnica de SVM independe do número de dimensões extras adicionadas pelo Kernel, ou seja, pode-se trabalhar em dimensões elevadas sem ter que estimar um grande número de parâmetros.

5.3.4 Classificação de múltiplas classes

O classificador SVM é por definição um discriminante binário, ou seja, utilizado para classificar duas classes. Para fazer a classificação quando da existência de mais de duas classes, métodos auxiliares são utilizados, como por exemplo a técnica *um-contra-um* (utilizada neste trabalho).

Nesta técnica um discriminante SVM é calculado para cada par de classes, o que totaliza $C(C-1)/2$ classificadores diferentes, onde C é o número total de classes. Para classificar um novo dado é utilizado um sistema de *votação* no qual é feita a classificação por todos os discriminantes e cada decisão consiste em um *voto*. A classe com maioria de votos é escolhida como decisão final.

Em caso de empate é utilizado o critério de *magnitude total das funções discriminante* (MTFD) que consiste escolher a classe cuja MTFD é maior:

$$MTFD = \sum |\hat{f}_{c_1,c_2}(x)| \tag{5.55}$$

onde $\hat{f}_{c_1,c_2}$ é dado na Eq. 5.33, sendo $(c_1, c_2) \in \{1, 2, ..., C\}$ o par de classe em questão.

Capítulo 6

Comparação de técnicas de classificação de dados de EEG

Nosso objetivos só podem ser alcançados por meio de um plano, no qual devemos acreditar intensamente, e pelo qual devemos agir vigorosamente. Não existe outra rota para o sucesso.

- Pablo Picasso

Por ser uma nova área de pesquisa, a classificação de dados de EEG sofre de um problema de padronização e síntese. Várias técnicas foram apresentadas pela literatura especializada e, além disso, inexistem trabalhos que as comparem de forma completa, resultando em uma dificuldade imensa para qualquer pesquisador que deseje escolher a técnica mais adequada para uma aplicação em Interface Cérebro Máquina (ICM).

O leitor melhor informado poderia discordar da afirmação de que não existem comparações entre as técnicas de classificação de dados de EEG já que vários trabalhos utilizam bancos de dados em comum e também pela existência das *BCI competitions*, que são competições nas quais várias técnicas são comparadas em termos da melhor taxa de classificação em determinado banco de dados.

Porém, uma comparação *completa* deveria testar simultaneamente várias técnicas em diversos bancos de dados. Note que não é este o caso nas comparações existentes já que se confrontam algumas técnicas com o uso de bancos de dados específicos. De

fato, uma técnica de classificação só poderia ser considerada superior as demais se apresentasse classificações melhores em diversas situações e bancos de dados distintos.

Para comprovar esta afirmação serão utilizadas duas técnicas e dois bancos de dados que mostrarão que o fato de dada técnica ser melhor sucedida em um experimento não implica que a mesma será melhor sucedida em um segundo. Serão utilizadas as técnicas *Weighted Fourier frequencies and SVM* [32] e *Continuous Wavelet Transform and the t-value Scalogram* [33] (vencedora da *BCI competition 2003*). Como essas técnicas não são o foco principal deste trabalho, apresentam-se, a seguir, apenas uma breve descrição destas sendo sugerido ao leitor o estudo dos trabalhos [32, 33]. Antes das técnicas propriamente ditas, apresenta-se também algumas definições básicas para facilitar o entendimento a respeito de alguns termos que serão repetidos diversas vezes ao longo do trabalho.

6.1 Noções e Definições básicas

Um banco de dados de EEG advém de um processo de coleta de dados no qual um *sujeito* utiliza um equipamento, que possui um total de L *eletrodos*, de registro de sinais elétricos do cérebro.

Notação: O l-ésimo eletrodo será representado por $l = \{1, 2, ..., L\}$.

Cada eletrodo l registra uma série temporal dos impulsos elétricos emitidos pelo córtex cerebral. Nos experimentos realizados para coleta dos sinais utilizados nesse trabalho o *sujeito* foi submetido várias vezes a um total de C *estímulos* distintos que serão referidos como *classes*.

Notação: Existem C classes denotadas por W_c, onde $c = \{1, ..., C\}$.

O *sujeito* é submetido a cada um dos *estímulos* diversas vezes em ordem aleatória o que resulta em diversas *repetições* de cada *classe*.

Notação: A n-ésima *repetição* da *classe* W_c será denotada por $n_c = \{1, 2, ..., N_c\}$.

O processo de *classificação* consiste em três fases: a fase de *extração de características* na qual algumas técnicas (como o periodograma por exemplo) são aplicadas nos sinais de forma a extrair padrões característicos de cada sinal; a fase de *treinamento* na qual as N_c repetições de cada classe W_c são utilizadas para treinar o *classificador*; e, finalmente, a fase de *teste* na qual um novo conjunto de sinais de classe desconhecida será classificado em alguma das classes W_c.

6.1.1 Continuous Wavelet Transform and the t-value Scalogram

A técnica continuous wavelet transform and the t-value Scalogram (t-CWT) foi apresentada em [33] e utiliza transformada de wavelet contínua CWT (ver seção 5.1.2) e a estatística do teste t para extração de características de duas classes. Isto é feito em 5 passos simples:

1- A CWT $W^{ln_g}(s,t)$ é computada para cada eletrodo $l = 1, ..., L$, para cada classe $g = \{1, 2\}$ e para cada repetição $n_g = 1, 2, ..., N_g$ dos dados de treinamento, onde s representa a escala e t o deslocamento no tempo (ver Eq. 5.12).

2- Em seguida calcula-se a média $\overline{W_g^l(s,t)}$ e a variância $\sigma_g^l(s,t)$ para cada uma das classes ($g = 1, 2$), onde

$$\overline{W_g^l(s,t)} = \frac{1}{N_g} \sum_{n_g=1}^{N_g} W^{ln_g}(s,t) \tag{6.1}$$

$$\sigma_g^l(s,t) = \frac{1}{N_g - 1} \sum_{n_g=1}^{N_g} (W^{ln_g}(s,t) - \overline{W_g^l(s,t)})^2 \tag{6.2}$$

onde N_g denota o número de repetições na classe g ($g = \{1, 2\}$).

3- A estatística t, $t^l(s,t)$ é computada

$$t^l(s,t) = \sqrt{\frac{N_1 N_2}{N_1 + N_2}} \frac{\overline{W_2^l(s,t)} - \overline{W_1^l(s,t)}}{\sigma_{pl}^l(s,t)} \tag{6.3}$$

onde $\sigma_{pl}^l(s,t)$ é a variância composta (pooled variance) das duas amostras.

4- Os pontos (s_i^l, t_i^l) nos quais a função $t^l(s,t)$ tem um extremo local são encontrados e constituem o conjunto $\mathbb{U}$. Esses são os pontos de máxima diferença entre os dois grupos na escala de tempo e frequência e formam o vetor de características.

5- A CWT $W_i^{ln_g}$ é computada para cada ponto $(s_i^l, t_i^l) \in \mathbb{U}$ e para cada repetição n_g para os dados de treinamento e de teste de acordo com

$$W_i^{ln_g} = W^{ln_g}(s_i^l, t_i^l) \tag{6.4}$$

então é utilizado algum algoritmo de classificação nesses vetores. Em [33] foi utilizado o LDA, porém utilizaremos SVM.

6.1.2 Weighted Fourier frequencies and SVM

O método de classificação apresentado em [32] chamado de *Weighted Fourier frequencies and SVM* (WFF-SVM) basicamente utiliza o periodograma (ver seção 5.1.1) para extração de características e em seguida calcula um discriminante SVM diferente para cada frequência do periodograma. Finalmente, utiliza um sistema de pesos para obter uma decisão final.

Assim, para cada eletrodo $l = 1, 2, ..., L$ e para cada frequência ω_k (ver Eq. 5.4) defina $\text{SVM}_{l,k}[.]$ como sendo a função discriminante produzida pela técnica de SVM que classifica um novo valor $I_l^{Test}(\omega_k)$ (ver Eq. 5.10) do periodograma para um novo sinal em uma das duas classes W_1 ou W_2 de acordo com

$$\text{SVM}_{l,k}[I_l^{Test}(\omega_k)] = \begin{cases} 1, & \text{se } I_l^{Test}(\omega_k) \text{ é classificado em } W_1. \\ -1, & \text{se } I_l^{Test}(\omega_k) \text{ é classificado em } W_2. \end{cases}, \tag{6.5}$$

Deste modo, cada discriminante irá classificar um novo sinal em uma das duas classes dependendo se o periodograma possui valores altos ou baixos para um determinado eletrodo e frequência. Porém cada discriminante $\text{SVM}_{l,k}[.]$ pode apresentar uma decisão diferente. Assim, para unificar essas decisões é utilizado um sistema de pesos que gera uma decisão única ao problema. Esses pesos são definidos por

$$\Psi_{l,k} = [1 - 2 \times (\text{Min}\{\text{Taxa de erro}, 0.5\})]^{\rho}, \tag{6.6}$$

onde *Taxa de erro* $\in [0, 1]$, ρ é um parâmetro escolhido a priori, $l = 1, ..., L$ representa o eletrodo e k representa a frequência ω_k do periodograma ou o k-ésimo ponto do periodograma suavizado, ver seção 5.1.1.

Em seguida, dado um novo sinal, classifica-se utilizando os discriminantes da equação 6.5 e os pesos da equação 6.6 de acordo com a seguinte regra de classificação:

$$D = \text{sign}\left\{\sum_{l=1}^{L}\sum_{k=-[(n-1)/2]}^{[n/2]} \Psi_{l,k} \times \text{SVM}_{l,k}[I_l^{Test}(\omega_k)]\right\}. \tag{6.7}$$

$$\text{Decisão} = \begin{cases} W_1, & \text{se } D = 1 \\ W_2, & \text{if } D = -1 \\ None, & \text{se } D = 0 \end{cases}, \tag{6.8}$$

6.2 Aplicações com dados reais

Esta seção apresenta duas aplicações com dados reais de EEG. A primeira aplicação utiliza dados coletados em um experimento conduzido pelo laboratório Multi-Sensing-Processing and Learning Lab (MSPL) na universidade do Texas em El Paso, UTEP. A segunda aplicação utiliza dados coletados na UnB no laboratório LAICO utilizando o aparelho Emotiv.

6.2.1 Classificação de estímulos visuais

Os dados utilizados nesta seção foram coletados por um voluntário que utilizou um equipamento de 128 eletrodos para o registro do EEG. Dez estímulos visuais foram apresentados ao sujeito, dentre estes 2 contas matemáticas e 8 imagens abstratas (ver Figura 6.1). Cada uma dessas imagens foram apresentadas em ordem aleatória 4 vezes cada uma. Para mais detalhes sobre o banco de dados ver [32].

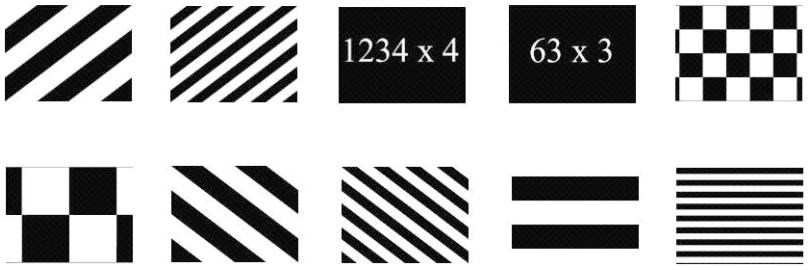

Figura 6.1: *Estímulos apresentados no experimento.*

WFF-SVM	t-CWT
95%	53.9%

Tabela 6.1: Taxas utilizando as técnicas WFF-SVM e t-CWT.

Em Coutinho et. al (2012) [32] os autores utilizaram validação cruzada para estimar a taxa de classificação correta de acordo com o seguinte procedimento:

i- Dados os sinais referentes as 4 repetições de uma figura matemática e as 4 repetições de um figura abstrata retiram-se do conjunto os sinais referentes à primeira repetição de cada figura.

ii- Os sinais restantes são utilizados para treinar o classificador.

iii- Os sinais retirados são classificados contabilizando-se o acerto ou não da classificação.

iv- Repõem-se os sinais retirados. Retiram-se, agora, os sinais referentes à segunda repetição de cada classe. Realizam-se os passos ii e iii novamente.

v- Repõem-se os sinais retirados. Retiram-se, agora, os sinais referentes à terceira repetição de cada classe. Realizam-se os passos ii e iii novamente.

vi- Repõem-se os sinais retirados. Retiram-se, agora, os sinais referentes à quarta repetição de cada classe. Realizam-se os passos ii e iii novamente.

vii- Repetem-se os passos i à vi para todos os pares possíveis de figuras matemáticas e figuras abstratas.

Os autores utilizaram componentes principais para selecionar os eletrodos referentes às regiões do córtex cerebral ativas durante o experimento e a técnica WFF-SVM (apresentada na seção 6.1.2) para classificar os dados em questão, obtendo uma taxa de classificação correta de 95%. Utilizando os mesmos eletrodos selecionados no artigo e classificando os dados empregando a técnica t-CWT obteve-se taxa de acerto de apenas 53.9%.

WFF-SVM	t-CWT
56%	65%

Tabela 6.2: Taxas utilizando as técnicas WFF-SVM e t-CWT.

Nota-se que a extração de características valendo-se do periodograma possui uma clara vantagem sobre a extração utilizando-se wavelets. Porém essa vantagem nem sempre ocorre, como observa-se na aplicação subsequente.

6.2.2 Classificação de imagética motora

Nesta seção apresentamos uma segunda aplicação relativa a dados coletados no laboratório LAICO do departamento de computação da Universidade de Brasília. Os dados foram coletados de um voluntário que utilizou o *headset* da Emotiv que possui 14 eletrodos para o registro do EEG.

O experimento ocorreu da seguinte forma: o sujeito sentou-se à frente do computador que apresentou aleatoriamente setas apontando para esquerda ou para direita. Dependendo da direção apontada pela seta o voluntário tinha que imaginar os movimentos da mão esquerda ou direita. No total, obtiveram-se 20 repetições para cada classe.

Para estimar a taxa de classificação correta dos métodos de classificação utilizou-se uma validação cruzada em que 15 das 20 repetições de cada classe foram selecionadas aleatoriamente para o treinamento e 5 foram deixadas para fase de teste. Esse processo foi repetido 30 vezes aleatoriamente e, portanto, testaram-se 300 sinais.

Para utilizar a técnica WFF-SVM suavizou-se o periodograma empregando a técnica de médias móveis (ver seção 5.1.3) com janela de tamanho 20 e os parâmetros $\rho = 2$ (Eq. 6.6) e $C = 1$ (Eq. 5.36), obtendo-se uma taxa de apenas 56% (o que pode ser devido à ausência de um ajuste fino nos parâmetros). Já utilizando-se a técnica t-CWT, foi obtida uma taxa de 65% de classificação correta.

Essas duas aplicações comprovam o que foi mencionado anteriormente sobre a dificuldade de se definir sobre a melhor técnica para classificação de dados de EEG.

Afirma-se, contudo, que a maior dificuldade é, especificamente, escolher a técnica de extração de características mais adequada a cada situação. De fato, na primeira aplicação o periodograma se sobressaiu à transformada de wavelet contínua e na segunda o inverso ocorreu.

Construídos todos esses elementos, fortificados em teoria e na prática, reafirma-se e solidifica-se a necessidade fundamental de se construir um método de seleção de características que seja robusto às diversas situações práticas e que possa fazer automaticamente, sem necessidade de estimações excessivas de parâmetros e computações em demasia, a seleção das características ótimas para o treinamento de um classificador. Uma proposta para tal método de seleção de características é apresentada no capítulo subsequente.

Capítulo 7

Selecionador de Características

"O verdadeiro sinal de inteligência não está no conhecimento mas na imaginação."

- Albert Einstein.

Este capítulo apresenta o algoritmo que seleciona, dado um conjunto de treinamento, as características mais adequadas para a classificação de sinais de EEG. A ideia do procedimento é permitir ao usuário testar *várias* técnicas de extração de características *simultaneamente* deixando o trabalho de escolher as *melhores* características para o algoritmo.

Além disso, o algoritmo busca *minimizar* a redundância das características selecionadas, escolhendo apenas aquelas que possam, de fato, contribuir com informações novas. Minimizar a quantidade de características é de extrema importância para que, mais a frente, a classificação de novos dados possa ser feita em *tempo real*.

Para atingir tais feitos o procedimento executa uma sequência de 3 testes estatísticos: análise de variância (ver Anexo 1), teste para o coeficiente de correlação (ver Anexo 1) e o teste FDR, apresentado abaixo.

7.1 False Discovery Rate (FDR)

Apresentado por Benjamini e Hochberg [35], o teste FDR (False Discovery Rate) é uma forma diferente de se considerar e controlar os erros em testes estatísticos múltiplos. Inicialmente, considere o problema de se testar simultaneamente m hipóteses

das quais m_0 são verdadeiras e $\mathbf{R}$ delas são rejeitadas, conforme resumido na Tabela 7.1.

Desta forma, as m hipóteses são consideradas conhecidas à priori e $\mathbf{R}$ é uma variável aleatória observável. Além disso, $\mathbf{U}$, $\mathbf{V}$, $\mathbf{S}$, e $\mathbf{T}$ são variáveis aleatórias não-observáveis. Portanto, se cada hipótese é testada individualmente com nível de significância α então $\mathbf{R} = \mathbf{R}(\alpha)$ cresce em α.

	Declarado não-significante	Declarado significante	Total
Hipótese nula é verdadeira	$\mathbf{U}$	$\mathbf{V}$	m_0
Hipótese nula é falsa	$\mathbf{T}$	$\mathbf{S}$	$m - m_0$
	$m - \mathbf{R}$	$\mathbf{R}$	m

Tabela 7.1: Número de erros cometidos ao testar m hipóteses.

Para definir a FDR, note que a proporção dos erros cometidos por rejeitar a hipótese nula, equivocadamente, pode ser vista através da variável aleatória $\mathbf{Q} = \mathbf{V}/\mathbf{R}$. Naturalmente, define-se $\mathbf{Q} = 0$ quando $\mathbf{R} = 0$ e nenhuma rejeição equivocada pode ser feita. Assim como $\mathbf{V}$, $\mathbf{Q}$ é uma variável aleatória não observável mesmo após o experimento e análise dos dados. Finalmente, define-se a FDR Q_e pelo valor esperado de $\mathbf{Q}$,

$$Q_e = E(\mathbf{Q}) = E\left(\frac{\mathbf{V}}{\mathbf{R}}\right). \tag{7.1}$$

O objetivo do teste FDR é, portanto, controlar o valor Q_e. Para obter tal efeito, considere testar as hipóteses $\mathbf{H}_1$, $\mathbf{H}_2$,..., $\mathbf{H}_m$ baseado nos *p-valores* $\mathbf{P}_1$, $\mathbf{P}_2$,..., $\mathbf{P}_m$. Sejam $\mathbf{P}_{(1)} \leq \mathbf{P}_{(2)} \leq ... \leq \mathbf{P}_{(m)}$ os *p-valores* ordenados e denote por $\mathbf{H}_{(i)}$ a hipótese correspondente a $\mathbf{P}_{(i)}$. Defina o seguinte teste múltiplo do tipo Bonferroni:

$$\text{Seja } k \text{ o maior valor de } i \text{ para o qual } \mathbf{P}_{(i)} \leq \tfrac{i}{m}\alpha$$

$$\text{então rejeite todas } \mathbf{H}_{(\mathbf{i})} \text{ tais que } i \leq k. \tag{7.2}$$

Teorema 1. Para estatísticas independentes e qualquer configuração da hipótese nula o procedimento apresentado em 7.2 controla a FDR ao nível α.

Prova. A prova do Teorema 1 segue do seguinte lema, cuja prova pode ser encontrada em [35].

Lema. Para quaisquer $0 \leq m_0 \leq m$ p-valores independentes correspondendo às verdadeiras hipóteses, e quaisquer $m_1 = m - m_0$ p-valores correspondendo às falsas hipóteses, o teste múltiplo definido em 7.2 satisfaz

$$E(\mathbf{Q}|\mathbf{P}_{m_0+1} = p_1, ..., \mathbf{P}_m = p_{m_1}) \leq \frac{m_0}{m}\alpha. \tag{7.3}$$

Portanto, suponha que $m_1 = m - m_0$ das hipóteses sejam falsas. Então, qualquer que seja a distribuição conjunta dos p-valores correspondentes a essas falsas hipóteses $\mathbf{P'}_1, ..., \mathbf{P'}_{m_1}$, integrando a inequação acima obtemos

$$E(\mathbf{Q}) \leq \frac{m_0}{m}\alpha \leq \alpha, \tag{7.4}$$

e o FDR está controlado.

Em geral, em aplicações práticas, as estatísticas dos diversos testes não são independentes, o que impossibilita o uso do teste apresentado. Porém, Benjamini e Yekutieli [36] apresentaram um teorema que relaxa a hipótese de independência:

Teorema 2. Se a distribuição conjunta das estatísticas do teste possuem dependência positiva de regressão no subconjunto de estatísticas correspondentes às verdadeiras hipóteses, então o procedimento de Benjamini e Hochberg controla o FDR a um nível menor ou igual a $\frac{m_0}{m}\alpha$.

7.2 Algoritmo de seleção

Como mencionado, o propósito do selecionador de características é proporcionar ao usuário a possibilidade de aplicar, simultaneamente, diversas técnicas de extração de características nos dados de EEG e, em seguida, deixar para o procedimento a tarefa de escolher as características dos sinais que possuam, em conjunto, o máximo de informação com a menor dimensionalidade possível. Precede-se à exposição do algoritmo algumas notações que serão utilizadas. Sugere-se, ainda, a releitura da Seção 6.1 para maior esclarecimento sobre tais notações.

Dado um conjunto de treinamento $\mathbf{X}$ composto dos potenciais elétricos emitidos pelo cérebro e captados por cada eletrodo $l = \{1, 2, ..., L\}$ de um sujeito S submetido a um experimento E no qual S foi submetido a C estímulos distintos, que constituem as classes W_c, onde $c = \{1, 2, ..., C\}$, repetidos N_c vezes cada um, em ordem aleatória

e com durações idênticas. Denota-se x_{i,l,c,n_c} pelo $i-$ésimo potencial elétrico registrado pelo eletrodo l quando S foi submetido ao estímulo da classe W_c pela n_c-ésima vez, onde $n_c = \{1, 2, ..., N_c\}$.

Seja $\mathbf{X}_{c,n_c} \in \mathbf{X}$ o conjunto referente a todos os potenciais elétricos coletados por todos os eletrodos quando S foi submetido ao estímulo de classe W_c, $c = \{1, 2, ..., C\}$, pela n_c-ésima vez, $n_c = \{1, 2, ..., N_c\}$. Aplica-se em cada conjunto $\mathbf{X}_{c,n_c}$ um total de T técnicas de extração de características (ver a seção 5.1 para alguns exemplos) de forma que, para cada técnica $t = \{1, 2, ..., T\}$, obtém-se os vetores de características $\mathbf{F}_{t,c,n_c}$ compostos de M características $m = \{1, 2, ..., M\}$ com um total de N_c amostras f_{m,t,c,n_c} para cada classe c.

A seleção de características ocorre em duas partes distintas. Para isso os vetores de características são divididos em dois grupos. Portanto, sejam U_c e V_c dois conjuntos tais que $U_c \cup V_c = \{1, 2, ..., N_c\}$, $U_c \cap V_c = \emptyset$, $|U_c| \geq 2$ e $|V_c| \geq 1$, $\forall c = \{1, 2, ..., C\}$. Desta forma, formam-se os conjuntos:

$$\mathbb{T}_1 = \{\mathbf{F}_{t,c,n_c} : t = \{1, 2, ..., T\}, c = \{1, 2, ..., C\}, n_c \in U_c\} \tag{7.5}$$

$$\mathbb{T}_2 = \{\mathbf{F}_{t,c,n_c} : t = \{1, 2, ..., T\}, c = \{1, 2, ..., C\}, n_c \in V_c\} \tag{7.6}$$

Com essas definições e notações, prossegue-se para a apresentação do selecionador de características. Para facilitar o entendimento, o algoritmo apresentado será dividido em 4 partes menores. Cabe ressaltar também que o algoritmo foi simplificado em prol da didática e, portanto, não apresenta uma estrutura computacionalmente otimizada e protegida de erros causados pelo usuário ou por dados atípicos.

7.3 Rotina principal

A rotina principal é apresentada no Algoritmo 1. Basicamente o programa necessita como entrada os vetores de características $\mathbf{F}_{t,c,n_c}$, os conjuntos U_c e V_c, o total de técnicas de extração de características utilizadas T, o total de classes C, os níveis de significância α e "alphacorr" utilizados no teste FDR e no teste de correlação, respectivamente, o parâmetro ρ utilizado no teste de correlação e finalmente o parâmetro "minacc" que será explicado nas seções que seguem.

O algoritmo inicia o vetor vazio "Features" (linha 5) que guardará, ao final, todas as características selecionadas. Em seguida para cada técnica de extração de características t o programa utiliza o vetor $\mathbf{F}_{t,c,n_c}$, que contém M características, para iniciar a execução de 3 procedimentos: "AV.FDR", "T.SVM" e "T.Corr".

Esse 3 procedimentos funcionam de forma a eliminar características através de uma sequência de testes estatísticos que serão mais adequadamente explicados nas seções seguintes. As características que não são eliminadas (representadas pelo vetor "which$_t$" na linha 11) são, para cada técnica t, incorporadas ao vetor "Features" pela função "*append*".

Algorithm 1 Rotina principal

1: **Procedure:** FeaSelect
2: **Input:** $\mathbf{F}_{t,c,n_c}$, U_c, V_c, T, C, α, alphacor, minacc, ρ
3: **Output:** Features
4: **External:** AV.FDR, T.SVM, T.Corr, *append*
5: Features $\leftarrow \{.\}$
6: **for** $t = 1$ to T **do**
7: $M \leftarrow |\mathbf{F}_{t,c,n_c}|$
8: AV.FDR(α)
9: T.SVM(minacc)
10: T.Corr(alphaCor)
11: *append* which$_t$ to Features
12: **end for**

7.4 Procedimento AV.FDR: Análise de Variância e FDR

O Algoritmo 2 recebe todas as variáveis criadas no Algoritmo 1, em particular todas as amostras $f_{m,t,c,n_c} \in \mathbf{F}_{t,c,n_c}$, $c = \{1, 2, ..., C\}$, $n_c = \{1, 2, ..., N_c\}$, $m = \{1, 2, ..., M\}$. Observe que, restringido à técnica t', à característica m' e $u_c = \{1, 2, ..., |U_c|\}$, pode-se escrever $y_{c,u_c} = f_{m',t',c,u_c}$. Fica claro que podemos, neste caso, definir y_{c,u_c} como sendo a u_c-ésima observação do $c-$ésimo tratamento onde $u_c = \{1, 2, ..., |U_c|\}$ e

$c = \{1, 2, ..., C\}$, sendo imediata a associação com a técnica de análise de variância (ver Anexo 1).

Adicionalmente, voltando ao objetivo principal, deseja-se selecionar as características que melhor discriminem as classes. Ora, é claro que para que uma característica possa diferenciar classes ela deve se comportar de maneira diferente para cada classe. Intuitivamente, se uma características se comporta de maneira diferente para cada classe, então, provavelmente, o mesmo deve ocorrer para o valor médio de amostras dessa característica (isso não acontece necessariamente).

Pelo processo inverso, conclui-se que, se identificarmos diferenças nas médias da característica μ_c de cada grupo $c = \{1, 2, ..., C\}$ então identificamos características que provavelmente contêm informação importante para a classificação de um novo dado e que devem ser selecionadas. Neste caso queremos testar a hipótese

$$H_0 : \ \mu_i = \mu_j \ \forall i, j$$

contra a hipótese

$$H_1 : \ \mu_i \neq \mu_j \text{ para pelo menos um par } i, j. \tag{7.7}$$

o que remete novamente à análise de variância e justifica seu uso no algoritmo de seleção de características.

Porém, ainda restrito à técnica t', um total de M testes de hipótese serão realizados (um para cada característica $m = \{1, 2, ..., M\}$). Portanto, deve-se atentar para o problema de comparações múltiplas que será controlado, neste caso, pelo teste FDR apresentado na seção 7.1.

Os testes mencionados compõem, dessa forma, o Algoritmo 2. O procedimento, que utiliza todas as variáveis existentes no Algoritmo 1, inicia criando o vetor vazio "AV" (linha 4) que guardará os p-valores dos testes F das análises de variância que serão realizadas. Em seguida, para cada característica do tipo m e classe W_c, o vetor vazio "ν_{cm}" (linha 7) é criado e recebe todas as amostras f_{m,t,c,n_c} onde $n_c \in U_c$ (linha 9).

Nas linhas 12 e 13, o algoritmo utiliza a função externa "av.pvalue" que calcula o p-valor do teste F da análise de variância dado os vetores $\nu_{1m}, \nu_{2m}, ...\nu_{Cm}$ que representam as amostras de cada tratamento ou, no caso, classe W_c. O p-valor calculado é, em seguida, incrementado ao vetor "AV" pela função "*append*".

Algorithm 2 Análise de variância e FDR

1: **Procedure:** AV.FDR
2: **Input:** α
3: **External:** *append*, av.pvalue, FDR
4: $AV \leftarrow \{.\}$
5: **for** $m = 1$ to M **do**
6: **for** $c = 1$ to C **do**
7: $\nu_{cm} \leftarrow \{.\}$
8: **for all** $n_c \in U_c$ **do**
9: *append* f_{m,t,c,n_c} to ν_{cm}
10: **end for**
11: **end for**
12: $p \leftarrow$ av.pvalue$(\nu_{1m}, \nu_{2m}, ...\nu_{Cm})$
13: *append* p to AV
14: **end for**
15: which$_t \leftarrow$ FDR(AV,α)

Finalmente, na linha 15, o algoritmo utiliza a função externa "FDR" que recebe o vetor "AV" e o nível de significância α e retorna um vetor contendo as características m tais que o teste de hipótese correspondente tenha sido rejeitado pelo teste FDR dado pela Eq. 7.2. O vetor retornado, "which$_t$", e todas as demais variáveis criadas neste algoritmo são globais e serão utilizadas nos algoritmos subsequentes.

7.5 Seleção com SVM

O fato das médias das amostras das características selecionadas pelo Algoritmo 2 terem comprovado a existência de diferenças entre essas médias não é suficiente, embora importante, para comprovar que essas características sejam interessantes do ponto de vista de classificação.

Para ter essa comprovação utiliza-se, no Algoritmo 3, um teste de classificação usando-se a técnica de SVM no qual uma taxa mínima de classificação correta (parâmetro de entrada "minacc") deve ser obtida.

Desta forma, o Algoritmo 3, que utiliza todas as variáveis criadas nos algoritmos anteriores, calcula uma função discriminante "classifier" (linha 5) para cada característica m pertencente ao vetor "which$_t$" (ver Algoritmo 2) utilizando a função "SVM" que, basicamente, recebe as amostras de cada classe W_c e característica m ($\nu_{1m}, \nu_{2m}, ...\nu_{Cm}$) e constrói classificadores SVM que serão utilizados para discriminar novas amostras baseado na técnica *um-contra-um* definida na Seção 5.3.4.

Em seguida, na linha 11, as amostras f_{m,t,c,n_c} são classificadas para cada classe W_c e repetição $n_c \in V_c$, incrementando o contador "acc" caso a amostra seja classificada corretamente. Note que o treinamento é feito com amostras pertencentes ao conjunto $\mathbb{T}_1$ e o teste é realizado com amostras pertencentes ao conjunto $\mathbb{T}_2$.

Finalmente, na linha 18, a característica m é excluída (pela função externa "*exclude*") do vetor "which$_t$" caso a taxa de acerto obtida "acc/tot" seja inferior a taxa de acerto mínima estipulada "minacc", concluindo o algoritmo. Cabe ressaltar ainda que, na linha 10, a amostra f_{m,t,c,n_c} foi incluída ao vetor ν_{cm} para uso no Algoritmo 4.

7.6 Teste para o coeficiente de correlação

Com os testes apresentados nas seções anteriores se pôde obter um conjunto de características com grandes chances de serem importantes para uma classificação futura. Porém, ainda é possível eliminar características redundantes que não trazem informações novas relativas ao conjunto total de características.

Eliminar tais características é de extrema importância para diminuir a dimensionalidade dos vetores de características e principalmente para que, numa aplicação de Interfaces Cérebro-Máquina, a classificação possa ser feita o mais rápido possível e em tempo real.

Para eliminar as características redundantes, o Algoritmo 4 utiliza o teste para o coeficiente de correlação apresentado no Anexo 1. Deste modo, para cada conjunto de características considerados "correlacionados" pelo teste, apenas aquela com menor p-valor obtido no teste F da análise de variância é mantida e as demais são descartadas. É importante enfatizar que a aproximação descrita é feita com o intuito apenas de diminuir a dimensionalidade do conjunto final de características e não representa

Algorithm 3 Teste SVM

Procedure: T.SVM
Input: minacc
External: *append*, *exclude*, SVM
for all $m \in \text{which}_t$ **do**
    classifier $\leftarrow$ SVM($\nu_{1m}, \nu_{2m}, ...\nu_{Cm}$)
    acc$\leftarrow$ 0
    tot$\leftarrow$ 0
    for $c = 1$ to C **do**
        for all $n_c \in V_c$ **do**
            append f_{m,t,c,n_c} to ν_{cm}
            if classifier(f_{m,t,c,n_c}) == c **then**
                acc $\leftarrow$ acc +1
            end if
            tot $\leftarrow$ tot +1
        end for
    end for
    if $\frac{\text{acc}}{\text{tot}} <$ minacc **then**
        exclude m from which_t
    end if
end for

Algorithm 4 Teste para coeficiente de correlação

Procedure: T.Corr
Input: alphaCor, ρ
External: *append*, *exclude*, rc.cor, correlation, unique
for $c = 1$ to C **do**
    exc$\leftarrow$ {.}
    maxcor$\leftarrow$rc.cor(alphaCor, ρ)
    for all $m_1 \in$ which$_t$ **do**
        for all $m_2 \in$ which$_t$ and $m_2 > m_1$ **do**
            cor $\leftarrow$ |correlation(ν_{cm_1}, ν_{cm_2})|
            if cor>maxcor **then**
                if AV[m_1] <AV[m_2] **then**
                    append m_2 to exc
                else
                    append m_1 to exc
                end if
            end if
        end for
    end for
end for
exc$\leftarrow$unique(exc)
for all $m \in$ exc **do**
    exclude m from which$_t$
end for

qualquer negação sobre a significância das características descartadas.

Dessa forma, o Algoritmo 4 inicia criando o vetor vazio "exc" que será utilizado para anotar as características m que deverão ser excluídas do vetor "which$_t$". Além disso, o algoritmo cria a constante "maxcor" obtida pelo uso da função externa "rc.cor" que recebe o nível de significância "alphaCor" e o parâmetro "ρ", que define ρ_0 no teste correlação (ver Anexo 1), e retorna o valor limiar para a região crítica do teste, ou seja, o teste é rejeitado para qualquer estatística obtida cujo valor absoluto seja maior que "maxcor".

Em seguida, para cada par de características m_1 e m_2 pertencentes ao vetor "which$_t$" e cada classe W_c, o algoritmo calcula a correlação "cor" entre ν_{cm_1} e ν_{cm_2} utilizando a função externa "correlation" (linha 9). Na sequência, se a correlação calculada "cor", em valor absoluto, for maior que o limiar da região crítica "maxcor" (linha 10), então o algoritmo inclui m_1 ou m_2 no vetor "exc" dependendo de qual p-valor do teste F da análise de variância seja maior (linha 11).

Em seguida, na linha 20, a função externa "unique" é utilizada para eliminar as características repetidas no vetor "exc". Finalmente, cada característica presente no vetor "exc" é excluída do vetor "which$_t$". Portanto, o vetor "which$_t$" contém todas as características selecionas pelo algoritmo de seleção de características para a técnica de extração t.

7.7 Treinamento e teste

Após a seleção das características pelo Algoritmo 1 em conjuntos com os demais algoritmos apresentados, procede-se para o treinamento do classificador. Neste trabalho, utiliza-se os vetores de características obtidos para treinar um classificador SVM. Note, entretanto, que a seleção de características é independente do treinamento do classificador e poderia ser utilizada qualquer outra técnica de classificação.

Para a fase de testes ou aplicação prática apenas as características selecionadas são calculadas, trazendo um ganho de tempo computacional importante. Portanto, o vetor de características extraído dos novos dados é classificado em uma das C classes. Resta agora comprovar a aplicabilidade e funcionalidade do algoritmo de seleção de característica e, por isso, o capítulo seguinte apresenta algumas aplicações com dados

reais para verificar tais resultados.

7.8 Pacote no R

Foi implementado um pacote no software R, chamado *eegclass*, que contém o selecionador de características apresentado neste capítulo. Além disso, o pacote apresenta uma série de outras funções como os classificadores apresentados nas Seções 6.1.1 e 6.1.2, funções gráficas capazes de plotar gráficos de EEG por canal, suas transformadas Wavelet contínua, o periodograma e o t-value scalogram da Seção 6.1.1.

No pacote do R, o selecionador de características seleciona, definido pelo usuário, características de janelamento dos sinais originais ou do seu periodograma, valores da transformada Wavelet contínua, e *loadings* das componentes principais aplicadas nos sinais originais ou no seu espectro. O *help* do pacote pode ser visto no Anexo 2.

7.8.1 Performance

O tempo computacional de um algoritmo é sempre uma preocupação relevante e, nesse aspecto, o novo algoritmo se sai muito bem. É evidente que na fase de treinamento o novo algoritmo exige um tempo computacional muito mais elevado que outras técnicas de extração de características pois calcula um conjunto de características muito maior e em seguida passa a selecionar nesse grande conjunto as melhores e mais relevantes características para manter no modelo.

Entretanto, a fase de treinamento não é, em geral, uma preocupação pois é feita *offline*. Em uma aplicação prática o mais importante é a fase *online*, como por exemplo controlar um braço mecânico via pensamento em tempo real.

Neste caso o novo algoritmo é extremamente relevante pois produz um conjunto de características pequeno que reflete em uma classificação extremamente rápida. No R, por exemplo, a classificação de um novo conjunto de sinais é feita em décimos de segundo.

7.8.2 Instalação

O pacote, nomeado *eegclass*, está disponível em formato *source*. A instalação no sistema operacional *Linux* é igual a instalação de qualquer outro pacote do R. Já no *Windows* alguns passos devem ser seguidos pois o pacote não está em formato binário.

O primeiro passo é instalar os outros pacotes necessários ao funcionamento do pacote, são eles: *e1071, class, wmtsa, splus2R, ifultools, MASS, fields e spam.*

Em seguida, no próprio R, acessar o diretório onde está a pasta *eegclass.tar.gz*:

```
setwd("C:/ ...caminho do diretório... ")
```

Finalmente, utilize a seguinte função para finalizar o processo de instalação:

```
install.packages("eegclass_1.0.tar.gz",type="source",repos=NULL)
```

7.8.3 Funções

Simulação de dados

O pacote *eegclass* possui a função *randeeg()* que "simula" dados de EEG. Na verdade, a função simula modelos de séries temporais do tipo ARIMA, o que não corresponde à verdadeira natureza dos dados de EEG. Sendo assim, a simulação é útil apenas para testar as funções do pacote.

Funções gráficas

O pacote *eegclass* possui duas funções gráficas: A função *ploteeg()*, que faz gráficos dos sinais, periodogramas, wavelets e também do *t-value scalogram* apresentado na seção 6.1.1. A função *plotwindows()* faz gráficos da técnica de janelamento utilizando uma estatística definida pelo usuário.

Selecionador de características

O pacote implementa o algoritmo proposto neste trabalho nas funções *FeatureEEG()* e *Feature3EEG()*. A primeira faz a seleção de características de duas classes, já a segunda faz a seleção otimizada para 3 classes. Para um número maior de classes, deve-se utilizar a função *FeatureEEG()* em todas as combinações possíveis de classes.

Funções de treinamento

O pacote disponibiliza 3 funções de treinamento. A primeira é a função *eeg.discr.cwt()* que calcula um discriminante utilizando *wavelets* como extração de características (dependo dos parâmetros escolhidos a função calcula o discriminante da Seção 6.1.1).

A segunda é a função *eeg.discr.spec()* que calcula um discriminante utilizando o periodograma como extração de características (dependendo dos parâmetros escolhidos a função calcula o discriminante da Seção 6.1.2).

A terceira é a função *svmEEG()* que calcula um discriminante *dado* um objeto criado pela função *FeatureEEG()*, ou seja, dadas as características selecionadas pelo algoritmo apresentado neste trabalho (ver a Tabela 7.2 para analisar as relações de dependência entre as funções de classificação).

Funções de classificação

As funções de classificação utilizam um objeto criado por uma das funções de treinamento para classificar um novo sinal de EEG (ver a Tabela 7.2 para analisar as relações de dependência entre as funções de classificação). Existem 3 funções de classificação:

A primeira é a função *eeg.class.cwt()* que utiliza o classificador criado pela função *eeg.discr.cwt()*. A segunda é a função *eeg.class.spec()* que utiliza o classificador criado pela função *eeg.discr.spec()*. Finalmente, a terceira é a função *classifyEEG()* que utiliza o classificador criado pela função *svmEEG()*.

Funções de validação cruzada

Existem duas funções que utilizam a técnica de validação cruzada para estimar a taxa de classificação correta de um algoritmo dado um conjunto de dados. A função *svmcwt.cv()* faz a validação cruzada utilizando um discriminante do tipo *eeg.discr.cwt()*. E a função *svmspec.cv()* faz a validação cruzada utilizando um discriminante do tipo *eeg.discr.spec()*.

Técnica de Classificação	Seleção de Características	Cálculo do Discriminante	Classificação de um novo sinal
t-CWT	...	*eeg.discr.cwt()*	*eeg.class.cwt()*
WFF-SVM	...	*eeg.discr.spec()*	*eeg.class.spec()*
Selecionador de Características	*FeatureEEG()*	*svmEEG()*	*classifyEEG()*

Tabela 7.2: Tabela que mostra a ordem das funções que devem ser aplicadas para cada técnica de classificação.

7.8.4 Formato do banco de dados

As funções do pacote *eegclass* necessitam que os dados de EEG estejam organizados em um formato específico. Mais precisamente, são necessários dois vetores e uma matriz.

A matriz, identificada nas funções pelo parâmetro *data*, é o banco de dados de EEG. As entradas dessa matriz denotam o potencial elétrico captado por um eletrodo em um instante de tempo específico. Assim, cada coluna dessa matriz deve representar um eletrodo e cada linha deve representar um instante do tempo (em geral, em ordem cronológica). Veja na Tabela 7.3, um exemplo de banco de dados que possui 4 eletrodos que captaram 6 valores cada um (no R a matriz deve conter apenas os números do exemplo).

	EL1	EL2	EL3	EL4
t1	0.023	0.001	0.031	0.067
t2	0.043	0.021	0.035	0.012
t3	0.045	0.060	0.024	0.084
t4	0.033	0.023	0.051	0.007
t5	0.057	0.026	0.021	0.056
t6	0.018	0.050	0.009	0.033

Tabela 7.3: Exemplo de matriz de dados de EEG.

O primeiro vetor necessário é identificado nas funções pelo parâmetro *classes*. Esse vetor tem tamanho igual ao número de linhas da matriz de dados. Cada entrada do

vetor deve indicar a classe do estímulo captado pelos eletrodos naquele instante de tempo. No exemplo da Tabela 7.3, considere $classes = (1, 1, 1, 2, 2, 2)$, isso significa que as primeira 3 linhas da matriz (instantes t1, t2 e t3) representam observações da primeira classe e as últimas três linhas (instantes t4, t5 e t6) representam observações da segunda classe.

O segundo vetor é identificado nas funções pelo parâmetro *reps*. Esse vetor também tem tamanho igual ao número de linhas da matriz de dados. Cada entrada do vetor deve indicar a repetição de cada classe do estímulo captado pelos eletrodos naquele instante de tempo. No exemplo da Tabela 7.3, seja $reps = (1, 1, 1, 1, 1, 1)$, isso significa que as primeira 3 linhas da matriz (instantes t1, t2 e t3) representam observações da primeira repetição da primeira classe e as últimas três linhas (instantes t4, t5 e t6) representam a primeira repetição da segunda classe.

7.8.5 Exemplos de uso do pacote

Funções gráficas

O pacote *eegclass* disponibiliza ao usuário uma série de funções interessantes. Por exemplo é possível "simular" um banco de dados de EEG através da função *randeeg()* (a simulação, na verdade, é feita por modelos de séries temporais do tipo ARIMA, o que não necessariamente reproduz dados reais de EEG. Contudo, a simulação é útil para testar o funcionamento de outras funções do pacote):

```
Sim <- randeeg(nclass=2,nrep=4,nobs=150,nel = 5, ar=
  matrix(c(0.3,0.1,0,0), 2,2,byrow=TRUE),
  ma=matrix(c(0.7,0),2,1,byrow=TRUE),
  order=matrix(c(2,0,1,0,0,0),2,3,byrow=TRUE),
  vars = c(1,1))
```

Neste exemplo, *nclass* denota o número de classes, *nrep* o número de repetições de cada classe, *nobs* o número de observações de cada sinal, *nel* o número de eletrodos e o restante são parâmetros relativos aos modelos ARIMA, detalhados no *help* do pacote.

Portanto *Sim* é um objeto que guarda o banco de dados e pode ser utilizado por outras funções do pacote. Por exemplo, pode-se traçar os gráficos dos sinais (ver Figura 7.1) pela função *ploteeg()* da seguinte forma:

```
ploteeg(Sim$data, Sim$reps , Sim$class , which.reps="ALL",
  which.classes = "ALL", which.el=1, type = 'original')
```

Ou o *t-value scalogram* apresentado na seção 6.1.1 (ver Figura 7.2):

```
ploteeg(Sim$data, Sim$reps , Sim$class , which.reps="ALL",
  which.classes = "ALL", which.el="ALL", type = 'T.pvalue',
  wavelet="gaussian2",abs=TRUE)
```

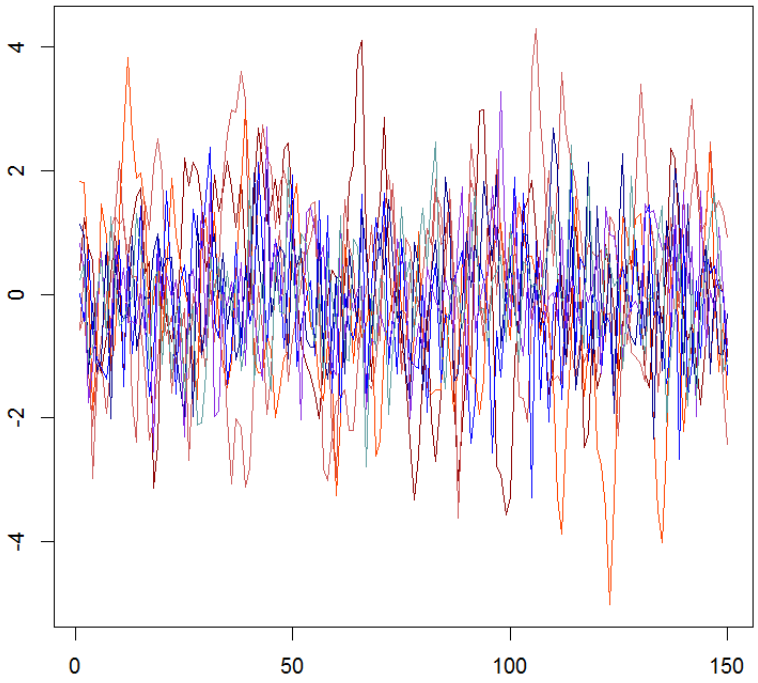

Figura 7.1: *Gráfico dos sinais, as cores avermelhadas representam uma classe e as azuladas a outra classe.*

Também é possível plotar gráficos utilizando a técnica de janelamento apresentada na seção 5.1.3 por meio da função *plotwindows()*. Veja o exemplo a seguir no qual é utilizado a soma da energia em janelas de tamanho 10 (ver Figura 7.3):

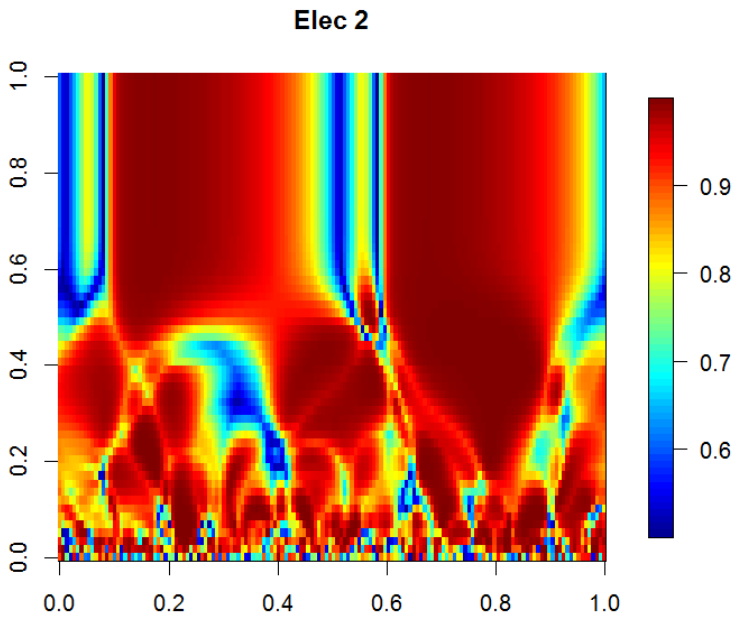

Figura 7.2: *O t-value scalogram proposto por Bostanov.*

```
plotwindows(data,reps,classes,which.reps="ALL", which.classes="ALL",
  which.el= "ALL",win=10,power=2,abs=FALSE,
  stat="sum", complete=TRUE,mintomax=FALSE)
```

Classificação de dados reais

Esta seção apresenta um exemplo completo de como ler um banco de dados, organizá-lo no formato necessário, treinar os classificadores apresentados neste trabalho e finalmente classificar novos sinais.

O primeiro passo é ler o banco de dados e o pacote. Neste exemplo existem 2 bancos de dados: um refere-se aos sinais propriamente ditos (denotados por *data*) e o outro refere-se as classes de cada linha do sinal (denotado por *Cl*). Procede-se, portanto, à leitura dos dados e do pacote pelos seguintes comandos:

```
library(eegclass)
```

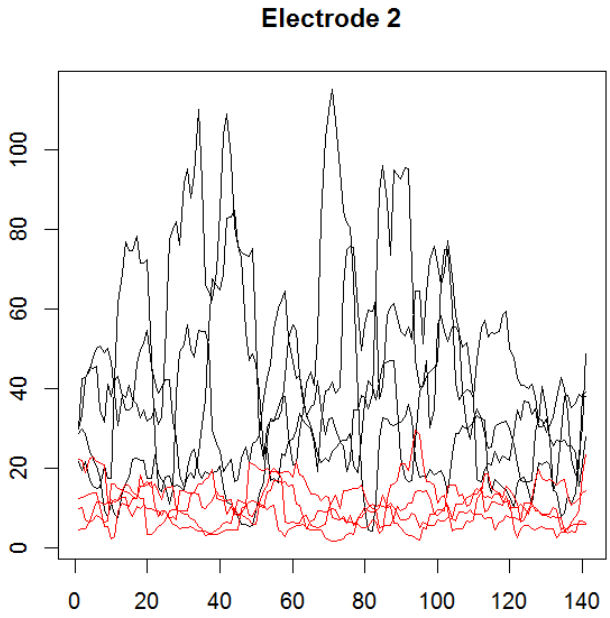

Figura 7.3: *Gráfico do janelamento utilizando a soma da energia dos sinais.*

```
setwd("C:/Users/Avell/Dropbox/BMI-UnB/datasets/29_11_2011")
 Cl<-read.csv2("motor-imagery-1-[2011.11.29-12.0]-
 ESTIMULOS.csv",header=T,dec=".")
data<-read.csv2("motor-imagery-1-[2011.11.29-12.0]-
 sinais.csv",header=T,dec=".")
```

Para utilizar o pacote do R, os dados devem estar no formato matricial mostrado anteriormente. Portanto, deve-se colocar o banco de dados lido no formato especificado. Note, entretanto, que esse procedimento será diferente para cada banco de dados em particular e, portanto, os detalhes do código apresentado abaixo foram omitidos.

```
Janela=1.25  #Duracao do estimulo.
Est1=770  #estimulo 1 (OVTK_GDF_Right)
Est2=769  #estimulo 2 (OVTK_GDF_Left)
...
data<-Data
```

```
...
classes<-c(normal[,colLabels],epi[,colLabels])
rep<-numeric(161*20)
for (i in 1:20){rep[(161*(i-1)+1):(161*i)]<-rep(i,161)}
reps<-c(rep,rep)
...
redureps<-c()
for (i in 1:15) {redureps<-c(redureps,rep(i,161))}
redureps<-c(redureps,redureps)
```

Após a leitura dos dados, procede-se à análise propriamente dita. Existem, no banco de dados em questão, 2 classes e 20 repetições de cada classe que serão divididas, aleatoriamente, em 15 repetições para a fase de treinamento e 5 para a fase de testes. Os comandos a seguir sorteiam as repetições que irão compor a fase de treinamento para as duas classes (*S1 e S2*), além de inicializar 4 contadores.

```
contBostanov<-0
contSpec<-0
contFea<-0
tot<-0

S1<-sample(1:20,15)
S2<-sample(1:20,15)
```

Em seguida, utiliza-se *S1* e *S2* para separar o banco de dados de treinamento (*usedata, useclasses*), como se pode ver abaixo:

```
prov1<-data[which(reps%in%S1 & classes==1),]
prov2<-data[which(reps%in%S2 & classes==2),]
usedata<-rbind(prov1,prov2)

prov1<-classes[which(reps%in%S1 & classes==1)]
prov2<-classes[which(reps%in%S2 & classes==2)]
```

```
useclasses<-c(prov1,prov2)
```

Neste ponto foram utilizadas as três técnicas de classificação apresentadas neste trabalho: t-CWT (seção 6.1.1), WFF-SVM (seção 6.1.2) e o selecionador de características (seção 7.2). As funções necessárias para classificação são diferentes para cada técnica como foi apresentado na Tabela 7.2.

Primeiramente, fez-se o cálculo do discriminante com o uso das técnicas WFF-SVM (função *eeg.discr.spec*) e t-CWT (função *eeg.discr.cwt*). É importante notar, no código abaixo, a diferença no valor do parâmetro *byfreq*: o valor *TRUE* significa calcular um discriminante para cada frequência e utilizar um sistema de pesos adequado e o valor *FALSE* significa calcular um único discriminante multidimensional para todas as frequências.

```
CLspec<-eeg.discr.spec(usedata, redureps , useclasses,ma=2,
   ma.s=20,alpha=0.995,test="wilcox",byfreq=TRUE)


CLbostanov<-eeg.discr.cwt(usedata, redureps, useclasses,
   byfreq = FALSE, alpha = 0.9, wavelet = "gaussian2", variance = 1,
   abs = TRUE, type = "local.max", kernel = "linear")
```

Em seguida calculou-se o discriminante utilizando-se o selecionador de características apresentado neste trabalho. Note que, neste caso, duas funções são utilizadas: *FeatureEEG* é o selecionador de características propriamente dito e *svmEEG* calcula o discriminante dadas as características selecionadas. Os parâmetros utilizados na função *FeatureEEG* estão presentes no Anexo 3.

```
x<-FeatureEEG(data=usedata,reps=redureps,classes=useclasses,
   nselec=nselec,featype=featype,ncomps=ncomps,win=win,stat=stat,
   power=power,abs=abs,log=log,mintomax=mintomax,Alpha=Alpha,
   AlphaCorr=AlphaCorr,minacc=minacc,Nfea=Nfea,wavelet=wavelet,
   variance=variance)


CLFeaSelec<-svmEEG(x)
```

As três funções discriminantes foram calculadas com sucesso e procedeu-se para a classificação de novos sinais. Para isso, utilizaram-se 5 repetições de cada classe excluídas da fase de treinamento:

```
test1<-c(1:20)[-S1]
test2<-c(1:20)[-S2]
```

O código abaixo faz a classificação desejada. Existem dois *loops* que percorrem os sinais de teste das classes 1 e 2, respectivamente. Dentro de cada *loop* selecionam-se os dados necessários armazenando-os em *testdata*, *testreps* e *testclasses*.

Em seguida, o sinal utilizado é classificado utilizando cada um dos discriminantes calculados anteriormente. Para isso, utilizaram-se as funções: *eeg.class.cwt*, *eeg.class.spec* e *classifyEEG* (ver Tabela 7.2). Finalmente, caso o sinal fosse classificado corretamente, o contador correspondente era incrementado. A taxa de classificação correta obtida neste exemplo pode ser vista na Tabela 7.4.

```
for (i in test1){
  testdata<-data[which(reps==i & classes==1),]
  testreps<-reps[which(reps==i & classes==1)]
  testclasses<-classes[which(reps==i & classes==1)]

  if (eeg.class.cwt(testdata, CLbostanov)==1)
      contBostanov<-contBostanov+1
  if (eeg.class.spec(testdata, CLspec)==1)
      contSpec<-contSpec+1
  if (classifyEEG(CLFeaSelec, as.vector(as.matrix(testdata)))[2]==1)
      contFea<-contFea+1

  tot<-tot+1
}

for (i in test2){
  testdata<-data[which(reps==i & classes==2),]
```

Técnica de Classificação	Taxa de Acerto
t-CWT	60%
WFF-SVM	60%
Selecionador de Características	80%

Tabela 7.4: Taxas de classificação correta obtida por cada técnica de classificação no exemplo apresentado.

```
  testreps<-reps[which(reps==i & classes==2)]
  testclasses<-classes[which(reps==i & classes==2)]

  if (eeg.class.cwt(testdata, CLbostanov)==2)
      contBostanov<-contBostanov+1
  if (eeg.class.spec(testdata, CLspec)==2)
      contSpec<-contSpec+1
  if (classifyEEG(CLFeaSelec, as.vector(as.matrix(testdata)))[2]==2)
      contFea<-contFea+1

  tot<-tot+1
}
```

Capítulo 8

Aplicações com dados reais

"Não importa o quão bela é sua teoria, nem
importa o quão inteligente você é.
Se ela não concordar com o
experimento, está errada.

- Richard P. Feynman.

Para comprovar a eficácia do algoritmo de seleção de características proposto, algumas aplicações com dados reais são apresentadas neste capítulo. Além disso, a técnica é comparada com outras técnicas de classificação de sinais de EEG que foram apresentadas em outros trabalhos e também com as técnicas das seções 6.1.1 e 6.1.2.

8.1 Características testadas

Como mencionado no capítulo anterior, várias técnicas de extração de características podem ser utilizadas e o algoritmo de seleção escolhe as características que melhor se adaptem ao problema.

Para as aplicações que serão apresentadas na sequência, diversas técnicas de extração de características foram utilizadas, tais como:

- Técnica de janelamento (ver Seção 5.1.3) aplicada nos sinais originais, utilizando estatísticas como máximo, energia média, energia máxima, mediana, mediana da energia, produtório das energias, para os tamanhos de janela 4, 5, 10, 15, 20, 30.

- Técnica de janelamento aplicada nos periodogramas dos dados (ver Seção 5.1.1), utilizando as diversas estatísticas e tamanhos de janelas citados no item anterior.
- Técnica de janelamento aplicada nas componentes principais dos dados (ver Seção 5.1.4), utilizando as diversas estatísticas e tamanhos de janelas citados no primeiro item.
- As estatísticas calculadas utilizando o janelamento dos itens anteriores foram selecionadas tanto em termos do tempo quanto dos quantis.
- Loadings das componentes principais (ver Seção 5.1.4).
- Wavelet (ver Seção 5.1.2) utilizando a wavelet mãe definida na Eq. 5.13.

8.2 Classificação de imagética motora

A aplicação desta seção utiliza os dados classificados na seção 6.2.2. Ressalta-se, porém, que os dados da seção 6.2.1 não foram classificados pois o uso de apenas 3 repetições no treinamento acarretaria uma performance ruim dos testes de análise de variância e SVM. Como mencionado na seção 6.2.2 os dados utilizados foram coletados no laboratório LAICO do departamento de computação da Universidade de Brasília de um voluntário que utilizou o *headset* da Emotiv que possui 14 eletrodos para o registro do EEG. Recordando:

O experimento ocorreu da seguinte forma: o sujeito sentou-se à frente do computador que apresentou, aleatoriamente, setas apontando para esquerda ou para direita. Dependendo da direção apontada pela seta, o voluntário devia imaginar os movimentos da mão esquerda ou direita. No total, obtiveram-se 20 repetições para cada classe.

Para estimar a taxa de classificação correta dos métodos de classificação utilizou-se uma validação cruzada na qual 15 das 20 repetições de cada classe foram selecionadas aleatoriamente para o treinamento e 5 foram deixadas para fase de teste. Esse processo foi repetido 30 vezes aleatoriamente e, portanto, testou-se 300 sinais.

A taxa de classificação correta utilizando o selecionador de características *FeaSelect* (Algoritmo 1) foi de 81%, bastante superior as taxas de 53% e 65% obtidas pelas

WFF-SVM	t-CWT	*FeaSelect*
53%	65%	81%

Tabela 8.1: Taxas utilizando as técnicas WFF-SVM, t-CWT e o algoritmo *FeaSelect*.

técnicas de classificação WFF-SVM e t-CWT, como mostra a Tabela 8.1. Foram utilizadas, além das técnicas de seleção apresentadas na Seção 8.1, nível de significância de 10% para o teste F da análise de variância, 20% para o teste de correlação (escolhido para eliminar uma quantidade menor de características), acurácia mínima de 65% para o teste SVM (ou seja, parâmetro "minacc"= 0.65 no Algoritmo 3). Além disso as 15 repetições foram divididas em 10 para a análise de variância e 5 para o teste SVM.

8.3 Aproveitamento de artefatos

A classificação de dados de EEG é uma tarefa bastante complicada, principalmente quando múltiplas classes estão envolvidas. Uma ideia que pode ser utilizada para melhorar as taxas de classificação e permitir a inclusão de mais classes é o aproveitamento de artefatos.

O procedimento é bem simples. Considere, por exemplo, o artefato provocado pelo piscar dos olhos (EOG), neste caso, como demonstrado na seção seguinte, um aumento do número de piscadas é facilmente identificado pelo algoritmo de classificação *FeaSelect*. Ou seja, pode-se utilizar o EOG como uma classe por si só e discernir facilmente entre as demais. Caso o novo sinal seja classificado nas classes restantes, procede-se ao processo de filtragem do EOG e classifica-se o sinal filtrado utilizando padrões cerebrais propriamente ditos como, por exemplo, a imaginação do movimento das mãos apresentados na seção anterior.

Imagine por exemplo, uma aplicação em que se deseja controlar um carro de controle remoto via sinais de EEG. O carro acelera constantemente sozinho e inclui-se as 3 classes seguintes:

- Classe 1: Frear.

- Classe 2: Virar para a esquerda.
- Classe 3: Virar para a direita.

Neste caso seria possível um classificador aproveitar o artefato do piscar dos olhos classificando os dados de acordo com o seguinte algoritmo:

1- Classifique na classe 1 caso identifique uma frequência excessiva do piscar dos olhos.

2- Caso contrário filtre os sinais coletados.

3- Classifique o sinal restante entre a classe 2 ou 3.

Desta forma, o usuário pisca com uma frequência maior para frear, imagina o movimento da mão esquerda para mover o carro para a esquerda e imagina o movimento da mão direita para mover o carro para a esquerda. Deste modo se pode aumentar o número de classes em uma aplicação de Interface Cérebro-Máquina. Seguem nas seções subsequentes duas aplicações: classificação dos artefatos do piscar dos olhos e de movimentos de mastigação.

8.3.1 Classificação do EOG

Os dados foram coletados no laboratório LAICO do departamento de computação da Universidade de Brasília de um voluntário que utilizou o *headset* da Emotiv que possui 14 eletrodos para o registro do EEG. O experimento de coleta indicava ao sujeito para piscar em ritmo normal ou acelerado. No total foram coletadas 20 repetições de cada classe, em ordem aleatória, com duração de 1.25 segundos de coleta.

Para estimar a taxa de classificação correta utilizou-se validação cruzada: das 20 repetições de cada classe 15 eram selecionadas aleatoriamente para o treinamento e 5 restantes eram deixadas para fase de teste. Este processo foi repetido 30 vezes aleatoriamente e, portanto, testaram-se 300 sinais.

Utilizando a técnica t-CWT de Bostanov, obteve-se uma taxa de classificação correta de 95% e utilizando a técnica WFF-SVM, com janelamento de tamanho 10 no periodograma, obteve-se uma taxa de 97% (ver a Tabela 8.3). Já a classificação

utilizando o algoritmo *FeaSelect* atingiu a taxa de 100% de classificação correta, ou seja, 300 sinais foram classificados corretamente.

Foram utilizadas, além das técnicas de seleção apresentadas na Seção 8.1, nível de significância de 1% para o teste F da análise de variância, 10% para o teste de correlação, acurácia mínima de 90% para o teste SVM (ou seja, parâmetro "minacc"= 0.65 no Algoritmo 3). Além disso as 15 repetições foram divididas em 10 para a análise de variância e 5 para o teste SVM.

WFF-SVM	t-CWT	*FeaSelect*
97%	95%	100%

Tabela 8.2: Taxas de classificação correta utilizando as técnicas WFF-SVM, t-CWT e o algoritmo *FeaSelect*.

8.3.2 Classificação do movimento de mastigação

Os dados foram coletados no laboratório LAICO do departamento de computação da Universidade de Brasília de um voluntário que utilizou o *headset* da Emotiv que possui 14 eletrodos para o registro do EEG. O experimento de coleta indicava ao sujeito para ficar em estado normal ou simular o movimento de mastigação. No total foram coletadas 40 repetições de cada classe, em ordem aleatória, com duração aleatória entre 1 e 2.5 segundos.

Os sinais coletados foram divididos em sinais de durações iguais de 0.5 segundos com saltos de 0.2 segundos entre o início de cada um deles. No total obtiveram-se 505 sinais relativos ao estado normal e 536 sinais relativos ao movimento de mastigar. Observe que sinais com duração de 0.5 segundos são mais difíceis de serem classificados, devido à quantidade reduzida de informação, do que os sinais com duração de 1.25 das seções anteriores.

Deste total, as 100 primeiras repetições de cada classe foram utilizadas no treinamento e as demais foram usadas para estimar a taxa de classificação correta, portanto, 841 sinais foram testados. Empregando a técnica de t-CWT de Bostanov, obteve-se taxa de 84.9% e utilizando WFF-SVM, com janelamento de tamanho 10, uma taxa

de 94%.

Usando o algoritmo de seleção *FeaSelect* obteve-se uma taxa de classificação correta de 95.4%. Foram utilizadas, além das técnicas de seleção da Seção 8.1, nível de significância de 5% para o teste F da análise de variância, 20% para o teste de correlação (para eliminar menos características), acurácia mínima de 70% para o teste SVM (parâmetro "minacc" do Algoritmo 3), e as 100 repetições de cada classe foram divididas em 60 para a análise de variância e 40 para o teste SVM.

WFF-SVM	t-CWT	*FeaSelect*
94%	84.9%	95.4%

Tabela 8.3: Taxas de classificação correta utilizando as técnicas WFF-SVM, t-CWT e o algoritmo *FeaSelect*.

8.4 Reconhecimento Neural

Outra aplicação bem interessante, na qual o algoritmo *FeaSelect* se saiu muito bem é no *Reconhecimento Neural*. Esta técnica, assim como a identificação de digitais e o reconhecimento de voz, busca identificar e diferenciar sujeitos podendo ser utilizada em sistemas de segurança.

O EEG de três sujeitos diferentes foram coletados no laboratório LAICO do departamento de computação da Universidade de Brasília utilizando o *headset* da Emotiv que possui 14 eletrodos para aquisição do EEG. Durante a coleta, os sujeitos se mantiveram relaxados por cerca de 3.5 minutos, esse processo foi repetido 2 vezes.

Os sinais coletados foram filtrados mantendo apenas as ondas alfa e divididos em 174 sinais menores com duração de 1.1 segundos para cada banco de dados. Utilizando-se o primeiro banco de dados coletados de cada sujeito, sortearam-se 50 repetições de cada banco para fazer o treinamento. Observe que esta aplicação considera 3 classes distintas simultaneamente, o que é mais difícil de classificar do que as 2 classes de cada aplicação anterior.

Treinou-se um classificador SVM nas características selecionadas pelo algoritmo de seleção *FeaSelect* com nível de significância de 5% para o teste F da análise

de variância, 20% para o teste de correlação (para eliminar menos características), acurácia mínima de 70% para o teste SVM (parâmetro "minacc" do Algoritmo 3), e as 50 repetições de cada classe foram divididas em 30 para a análise de variância e 20 para o teste SVM.

Duas classificações foram realizadas, a primeira utilizou as 124 repetições restantes de cada classe e as classificou obtendo 100% de classificação correta. A segunda classificação utilizou o segundo banco de dados de cada sujeito com 174 sinais de cada classe e os classificou com 100% de taxa de acerto. Portanto, nota-se que o Reconhecimento Neural foi extremamente eficiente utilizando o algoritmo selecionador de características.

8.5 Classificação de dados de Epilepsia

Esta aplicação utiliza um banco de dados disponível publicamente [38, 37] que consiste em 5 conjuntos de dados distintos contendo segmentos de EEG captados por 100 canais diferentes. Dois destes bancos (denotados A e B) foram obtidos captando-se os sinais do cérebro de 5 pacientes saudáveis com olhos abertos e fechados respectivamente. Os conjuntos C,D e E foram originados de um arquivo de EEG de diagnósticos pré-cirúrgicos de pacientes que sofrem de epilepsia. Os segmentos do banco D foram coletados da região epileptogênica enquanto que os dados do conjuntos C foram coletados da região oposta, o hipocampo. O banco E contém registros de um ataque epilético. Como em trabalhos anteriores [32, 27, 17, 18], utilizamos apenas os bancos A e E para testar o novo discriminante.

Cada banco (A e E) possui 100 sinais, um para cada eletrodo, e cada um desses sinais tem 4096 pontos. Para fazer a classificação descartou-se o início e o final dos sinais que, em seguida, foram divididos em 20 sinais de 200 pontos. Então a validação cruzada foi feita da seguinte forma:

i- A primeira repetição de cada grupo foi excluída do treinamento e foi utilizada na fase de testes. Acertos e erros de classificação foram computados.

ii- A primeira repetição do primeiro grupo e a segunda repetição do segundo grupo foram excluídas da fase de treinamento e foram utilizadas no teste.

iii- Esse procedimento foi repetido até que todas as combinações possíveis (400 neste caso) fossem utilizadas na fase de testes.

Note que os sinais utilizados na fase de teste não foram usados para treinar o classificador, sendo portanto uma aproximação confiável para a taxa de classificação correta. Vários autores fizeram uso deste banco de dados. Segue, abaixo, os resultados que foram obtidos:

- Subasi (2007) [17] usou *Mixture of Expert* (ME) model e Redes neurais artificiais (ANN) nos conjuntos A e E. Sua taxa de classificação correta foi de 94.5% usando ME e 93.2% usando ANN.

- Nigam e Graupe (2004) [27] também utilizaram ANN porém com outro método para a extração de características nos mesmos bancos A e E. Seu método obteve um taxa de 97.2%.

- Subasi e Gursoy (2010) [18] usaram transformada wavelet discreta (DWT), análise de discriminante linear (LDA), análise de componentes principais (PCA), análise de componentes independentes (ICA) e *Support Vector Machine* (SVM) nos conjuntos A e E obtendo uma taxa de 98.75% usando DWT, PCA e SVM, 99.5% usando DWT, ICA e SVM e 100% usando DWT, LDA e SVM.

- Guo et al. 2009 [26] empregaram energia wavelet relativa e redes neurais artificiais nos conjuntos A e E obtendo uma taxa de 95%.

- Jahankhani et al. (2006) [16] utilizaram wavelet para extração de características e redes neurais nos conjuntos A e E obtendo uma taxa de 98%.

- Polat et al. (2007) [29] usaram um sistema híbrido baseado numa árvore de decisão e a transformada de Fourier nos conjuntos A e E obtendo uma taxa de 98.72%.

- Siuly, et al. (2010) [20] aplicaram uma técnica de agrupamento e *least square support vector machine* nos conjuntos A e E obtendo uma taxa de 99.90%.

- Chandaka et al. (2009) [28] usaram correlação cruzada e SVM nos conjuntos A e E obtendo uma taxa de 95.96%.

- Ubeyli 2010 [30] usou LS-SVM nos conjuntos A e E obtendo uma taxa de 99.56%.

Treinou-se um classificador SVM nas características selecionadas pelo algoritmo de seleção *FeaSelect* com nível de significância de 5% para o teste F da análise de variância, 10% para o teste de correlação, acurácia mínima de 95% para o teste SVM (parâmetro "minacc" do Algoritmo 3), e as 19 repetições de cada classe foram divididas em 10 para a análise de variância e 9 para o teste SVM.

Obteve-se uma taxa de classificação correta de 100% para a classificação utilizando o algoritmo de seleção *FeaSelect*, superior a todas as taxas expostas nos artigos citados. Conclui-se, portanto, com esta e as demais aplicações a eficiência elevada do algoritmo proposto para a classificação de dados de EEG, em comparação com várias outras técnicas apresentadas.

Capítulo 9

Conclusões e Trabalhos Futuros

O estudo da classificação dos sinais de EEG é ainda muito recente e ainda tem muito a evoluir. Existem diversas técnicas que foram apresentadas na literatura sobre este assunto de forma que se torna difícil a escolha da melhor. Além disso, este trabalho mostrou que, pela extrema variedade dos bancos de dados de EEG, a melhor técnica em uma situação não é, necessariamente, a melhor em uma outra situação.

Por esta razão foi proposto um novo algoritmo selecionador de características. Este algoritmo é capaz de juntar diversas técnicas distintas de extração simultaneamente produzindo um grande conjunto de características. Em seguida, encontra, por meio de vários testes estatísticos, o conjunto ótimo de características para posterior treinamento de um classificador.

Desta forma, o algoritmo elimina do conjunto as características redundantes ou que não trazem um ganho na taxa de classificação produzindo um conjunto pequeno e coerente. Por sua vez, este conjunto reduzido permite que a classificação de novos sinais seja feita em décimos de segundo.

Outro atributo importante do algoritmo é sua flexibilidade em se adaptar a diferentes situações produzindo um conjunto de características distintas e ótimas para cada ocasião.

Finalmente, o selecionador de características foi testado em várias aplicações práticas obtendo, sempre, taxas maiores ou iguais as apresentadas por outras técnicas de extração e classificação.

Apesar dos bons resultados obtidos, ainda há muito a ser feito. Neste traba-

lho utilizamos algumas técnicas de extração de características (como periodograma e transformada Wavelet contínua), porém, existem diversas outras técnicas que poderiam ser incluídas e testadas no modelo.

Além disso a classificação das características selecionadas foi feita apenas utilizando-se SVM pois o foco do trabalho era a apresentação e uso do novo algoritmo de seleção de características. Portanto, outros classificadores como, por exemplo, Redes Neurais ou Redes Bayesianas poderiam ser testados.

Finalmente, ainda resta testar o novo algoritmo em uma aplicação em tempo real de Interface Cérebro-Máquina que é, afinal, o verdadeiro objetivo de todos os esforços de todos os pesquisadores que escolheram trabalhar nessa bela e desafiadora área de pesquisa.

Referências Bibliográficas

[1] Imai, M., 1954. "Herophilus of Chalcedon and the Hippocratic Tradition in Early Alexandrian Medicine." *Fac. of Humanities, University of Hirosaki, Japan.*

[2] "Life and Discoveries of Santiago Ramón y Cajal". Nobelprize.org. 1 Sep 2012 http://www.nobelprize.org/nobel-prizes/medicine/laureates/1906/cajal-article.html

[3] Cajal, R. y, 1937. "Recuerdos de mi Vida". *Cambridge: MIT Press. ISBN 84-206-2290-7.*

[4] Rose, S., 1973. "The Conscious Brain 1973", *ISBN 0-394-46066-9.*

[5] Piccolino, M., 1998. "Animal electricity and the birth of electrophysiology: The legacy of Luigi Galvani" *Brain Research Bulletin* **46**, No. 5, 381-407. Marco Piccolino*

[6] Niedermeyer, E., da Silva, F.L., 2005. "Electroencephalography: Basic Principles, Clinical Applications, and Related Fields" *Fifth ed., Lippincott Williams and Wilkins*

[7] Hoffmann, A., 2010. "EEG Signal Processing and Emotiv's Neuro Headset" *Tese, Universidade Técnica de Darmstadt.*

[8] Sörnmo, L., Laguna, P., 2005. "Bioelectrical Signal Processing in cardiac and neurological applications" *Elsevier* **1st ed.**

[9] Pfurtscheller, G., Leeb, R., 2004. "Walking through a virtual city by thought" *Proceedings of the 26th Annual Conference of the IEEE EMBS, San Francisco, USA.* 4503-4506.

[10] Hung, C. I., et al. 2005. "Recognition of motor imagery EEG using independent component analysis and machine classifiers" *Annals of Biomedical Engineering* **33**, No. 8, 1053-1070.

[11] Blankertz, B., et al. 2006. "The Berlin Brain-Computer Interface: EEG-based communication without subject training" *English IEEE transactions on neural systems and rehabilitation engineering* **14**, No. 2, 147-152.

[12] Nicolelis, M.A.L., et al., 2000. "Real-time prediction of hand trajectory by ensembles of cortical neurons in primates" *Nature* **408**, 361-365.

[13] Nicolelis, M.A.L., et al., 2003. "Learning to control a brain machine interface for reaching and grasping by primates" *PlosBiology* **1**, No. 2, 193-208.

[14] Nicolelis, M.A.L., 2011. "Muito além do nosso eu" *Companhia das Letras.*

[15] Flury, B., Riedwyl, H., 1988. "Multivariate Statistics. A Practical Approach" *Chapman and Hall, London.*

[16] Jahankhani, P., Kodogiannis, V., Revett, K., 2006. "EEG signal classification using wavelet feature extraction and neural networks" *IEEE John Vincent Atanasoff 2006 International Symposium on Modern Computing* , 52-57.

[17] Subasi, A., 2007. "EEG signal classification using wavelet feature extraction and a mixture of expert model" *Expert Systems with Applications* **32**, 1084-1093.

[18] Subasi, A., Gursoy, M. I., 2010. "EEG signal classification using PCA, ICA, LDA and support vector machines" *Expert Systems with Applications* **37**, 8659-8666.

[19] Kaper, M., Meinicke, P., Grossekathoefer, U., Lingner, T. and Ritter, H., 2003. "BCI competition 2003-data set iib: support vector machines for the p300 speller paradigm" *IEEE Trans. Biomed. Eng.* **51** , 1073-6.

[20] Siuly, et al., 2010. "Clustering technique-based least square support vector machine for EEG signal classification" *Comput. Methods Programs Biomed.*

[21] Penny, W.D., Roberts, S.J., Curran, E.A., and Stokes, M.J., 2000. "EEG-based communication: a pattern recognition approach". *IEEE Trans. Rehabil. Eng.***8**, 214-5.

[22] Pfurtscheller, G., Neuper, C., Schlogl, A. and Lugger, K., 1998. "Separability of EEG signals recorded during right and left motor imagery using adaptive autoregressive parameters." *IEEE Trans. Rehabil. Eng.* **6** , 316-25.

[23] Chiappa, S. and Bengio, S., 2004. "HMM and IOHMM modeling of EEG rhythms for asynchronous BCI systems". *European Symposium on Artificial Neural Networks ESANN* .

[24] Brockwell, P.J., Davis, R.A. 2002. "Introduction to Time Series and Forecasting". *2nd ed. Colorado: Springer.* **Cap. 4**.

[25] Johnson, R.A., Wichern, D.W. 2007. "Applied Multivariate Statistical Analysis". *New Jersey: Pearson Prentice Hall* **6th ed**.

[26] Guo, L., Rivero, D., Seoane, J.A., Pazos, A., 2009. "Classification of EEG signals using relative wavelet energy and artificial neural networks". *GCE*, 12-14.

[27] Nigam, V.P., and Graupe, D., 2004. "A neural-network-based detection of epilepsy". *Neurological Reasarch* **26(1)**, 55-60.

[28] Chandaka, S., Chatterjee, A., Munshi, S., 2009. "Cross-correlation aided support vector machine classifier for classification of EEG signals" *Expert Syst. Appl.* **36**, 1329-1336.

[29] Polat, K., Gunes, S., 2007. "Classification of epileptiform EEG using a hybrid system based on decision tree classifier and fast Fourier transform". *Appl. Math. Comput.* **187**.

[30] Ubeyli, E.D., 2010. "Least square support vector machine employing model-based methods coefficients for analysis of EEG signals". *Expert Syst. Appl.* **37** .

[31] Lotte, F., Congedo, M., Lécuyer, A., Lamarche, F. and Arnaldi, B., 2007. "A review of classification algorithms for EEG-based brain-computer interfaces." *Journal of Neural Engineering* **4** .

[32] Coutinho, M., Borries, G.F., Borries, R.F., 2012. "EEG data classification using Support Vector Machine and Fourier data analysis, a new approach".

[33] Bostanov, V., 2004. "BCI Competition 2003 - Data Sets Ib and IIb: Feature Extraction From Event-Related Brain Potentials With the Continuous Wavelet Transform and the t-Value Scalogram." *IEEE transactions on biomedical engineering* **V. 51, no. 6**.

[34] Montgomery, D.,C. 2001. "Design and Analysis of Experiments." *John Wiley and Sons.* **5th ed.**, 60-119.

[35] Benjamini, Y., Hochberg, Y. 1995. "Controlling the False Discovery Rate: A Practical an Powerful Approach to Multiple Testing." *Journal of the Royal Statistical Society, Series B.* **V. 57, no. 1**, 289-300.

[36] Benjamini, Y., Yekutieli, D., 2001. "The Control of the False Discovery Rate in Multiple Testing under Dependency". *The Annals of Statistics.* **V. 29, no. 4**, 1165-1188.

[37] EEG time series are available under http://www.meb.unibonn.de/epileptologie/science/physik/eegdata.html.

[38] Andrzejak, R. G., Lehnertz, K., Mormann, F., Rieke, C., David, P., Elger, C. E. 2001. "Indications of nonlinear deterministic and finite-dimensional structures in time series of brain electrical activity: Dependence on recording region and brain state." *Physical Review E.* **V. 64, 061907**.

Anexo 1

Análise de Variância

A análise de variância é utilizada quando se deseja comparar as médias de *a tratamentos* (μ_i, $i = \{1, 2, ..., a\}$) ou grupos, para testar a hipótese

$$H_0 : \ \mu_i = \mu_j, \ \forall i, j$$

contra a hipótese

$$H_1 : \ \mu_i \neq \mu_j, \ \text{para pelo menos um par } i, j. \tag{9.1}$$

Para isso, defina y_{ij} como sendo $j-$ésima observação do $i-$ésimo grupo onde $i = \{1, 2, ..., a\}$, $j = \{1, 2, ..., n_i\}$ e n_i denota o total de observações referentes ao tratamento i, e sejam também

$$y_{i.} = \sum_{j=1}^{n_i} y_{ij} \tag{9.2}$$

$$y_{..} = \sum_{i=1}^{a} \sum_{j=1}^{n_i} y_{ij} \tag{9.3}$$

onde $N = an$ é o total de observações.

O nome *análise de variância* surge devido à partição da variabilidade total dos dados em suas partes componentes. Dessa forma a soma de quadrados total corrigida

$$SS_T = \sum_{i=1}^{a} \sum_{j=1}^{n_i} y_{ij}^2 - \frac{y_{..}^2}{N} \tag{9.4}$$

é utilizada como medida da variabilidade total dos dados. A soma de quadrados total pode ser dividida em soma de quadrados dos tratamentos

$$SS_{Trat} = \sum_{i=1}^{a} \frac{y_{i.}^2}{n_i} - \frac{y_{..}^2}{N} \tag{9.5}$$

e dos erros

$$SS_e = SS_T - SS_{Trat} \tag{9.6}$$

Pode-se mostrar [34] que a estatística

$$F_0 = \frac{N-a}{a-1}\frac{SS_{Trat}}{SS_e} \tag{9.7}$$

tem distribuição F de Snedecor com $a-1$ e $N-a$ graus de liberdade. A Eq. 9.7 representa a estatística para testar a hipótese apresentada em 9.1. Intuitivamente, se a variabilidade devido aos tratamentos é grande então devemos rejeitar a hipótese 9.1, desta forma rejeitamos H_0 a um nível de significância α se

$$F_0 > F_{\alpha,a-1,N-a}. \tag{9.8}$$

Teste para o coeficiente de correlação

Dadas duas amostras $\{x_1, x_2, ..., x_n\}$ e $\{y_1, y_2, ..., y_n\}$ estima-se o coeficiente de correlação populacional ρ pelo coeficiente de correlação amostral

$$r = \frac{\sum_{i=1}^{n}(x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum_{i=1}^{n}(x_i - \bar{x})^2 \sum_{i=1}^{n}(y_i - \bar{y})^2}}. \tag{9.9}$$

Para testar a hipótese

$$H_0 : \rho = \rho_0, \tag{9.10}$$

Fisher sugeriu a seguinte transformação para a estatística r:

$$\xi = \frac{1}{2}\ln\frac{1+r}{1-r}, \tag{9.11}$$

que segue, sob a hipótese nula, uma distribuição aproximadamente normal com média

$$\mu_\xi = \frac{1}{2}\ln\frac{1+\rho_0}{1-\rho_0} \tag{9.12}$$

e variância

$$\sigma_\xi^2 = \frac{1}{n-3}. \tag{9.13}$$

No caso da hipótese

$$H_0 : \rho = 0, \tag{9.14}$$

existe um teste exato advindo da estatística

$$T = r\sqrt{\frac{n-2}{1-r^2}} \tag{9.15}$$

que segue uma distribuição t de Student com $n-2$ graus de liberdade.

Anexo 2

Package 'eegclass'

April 6, 2013

Type Package

Title Methods for exploration and classification of EEG data

Version 1.1

Date 2012-10-07

Author Murilo Coutinho (Universidade de Brasilia, Brazil)

Maintainer Murilo Coutinho <murilo9988@gmail.com>

Depends R (>= 2.14.0), e1071 (>= 1.6), wmtsa (>= 1.1-1) , fields (>= 6.6.3)

Description Software for analysis of EEG (electroencephalogram) data including methods for classification and plotting of patterns.

License GPL-2

Collate eeg.class.cwt.R eeg.class.spec.R eeg.discr.cwt.R
eeg.discr.spec.R ploteeg.R randeeg.R svmcwt.cv.R svmspec.cv.R
plotwindows.R FeatureEEG.R svmEEG.R classifyEEG.R Feature3EEG.R TrainEEG.R

R topics documented:

`eegclass-package` *Methods for exploration and classification of EEG data*

Description

Software for analysis of EEG (electroencephalogram) data including graphics and pattern classification methods.

Details

Package:	eegclass
Type:	Package
Version:	1.0
Date:	2012-03-11
License:	GPL-2

This package is a collection of tools to explore and classify EEG (electroencephalogram) data. EEG is the recording of electrical activity of the brain. The classification of EEG patterns is very important and it could be used to identify brain disorders and also to implement brain-machine interfaces (BMI) implementations.

To explore the data visually, graphical methods are available with functions `ploteeg` and `plotwindows`. The user is able to plot, in a very easy way, graphics of the EEG signals, the spectrum, wavelets and the t-value scalogram, for example.

To classify the data in one of two classes several methods were implemented: `eeg.discr.spec` creates a classifier using the spectrum as feature extraction and `eeg.class.spec` classifies a new input. The function `eeg.discr.cwt` creates a classifier using the continuous wavelet transform as feature extraction and `eeg.class.cwt` classifies a new input. The functions `svmcwt.cv` and `svmspec.cv` perform a cross validation to estimate the correct classification rate using the wavelet and the spectrum as feature extraction respectively. The function `FeatureEEG` can be used to extract the best features to train a SVM classifier using `svmEEG`, Function `classifyEEG` can be used to classify a new dataset.

Author(s)

Murilo Coutinho (Universidade de Brasilia)

Maintainer: Murilo Coutinho <murilo9988@gmail.com>

References

Bostanov, V. (2004) *BCI Competition 2003 - Data Sets Ib and IIb: Feature Extraction From Event-Related Brain Potentials With the Continuous Wavelet Transform and the t-Value Scalogram.* IEEE transactions on biomedical engineering, V. 51, no. 6.

Hastie, T., Tibshirani, R., Friedman, J. (2009) *The Elements of Statistical Learning: Data Mining, Inference, and Prediction.* 2nd ed. Stanford: Springer.

Coutinho, M., Borries, G.F., Borries, R.F. (2012) *EEG data classification using Support Vector Machine and Fourier data analysis, a new approach.*

See Also

`eeg.discr.cwt`, `eeg.discr.spec`, `eeg.class.cwt`, `eeg.class.spec`, `ploteeg`, `randeeg`, `svmcwt.cv`, `svmspec.cv`.

Examples

```
library(eegclass)

#Simulating a dataset with randeeg.
Sim <- randeeg(nclass=2,nrep=4,nobs=150,nel = 5,
ar=matrix(c(0.3,0.1,0,0),2,2,
byrow=TRUE),ma=matrix(c(0.7,0),2,1,byrow=TRUE),
order=matrix(c(2,0,1,0,0,0),2,3,
byrow=TRUE), vars = c(1,1))

#Plotting the data.
ploteeg(Sim$data, Sim$reps , Sim$class , which.reps="ALL",
which.classes = "ALL", which.el=1, type = 'original')

#Plotting the spectrum of the data.
ploteeg(Sim$data, Sim$reps , Sim$class , which.reps="ALL",
which.classes = "ALL", which.el=1, type = 'spectrum', ma.s = 10)

#Plotting the wavelet of the data.
ploteeg(Sim$data, Sim$reps , Sim$class , which.reps="ALL",
which.classes = "ALL", which.el=1, type = 'wavelet',
wavelet="gaussian2",abs=TRUE)

#Plotting the T value scalogram (based on the T statistic).
ploteeg(Sim$data, Sim$reps , Sim$class , which.reps="ALL",
which.classes = "ALL", which.el="ALL", type ='T.pvalue',
wavelet="gaussian2",abs=TRUE)

#Making a cross validation to estimate the
#correct classification rate,
#using the spectrum as feature extraction.
svmspec.cv(Sim$data, Sim$reps , Sim$class , all = TRUE , ma.s=10,
class.labels = c(1,2), print=TRUE,byfreq=TRUE, alpha=0.99,
test="t.test",type = "p.value")


#Making a cross validation to estimate the
#correct classification rate,
#using the continuous wavelet transform as feature extraction.
svmcwt.cv(Sim$data, Sim$reps , Sim$class , all = TRUE , abs=TRUE,mmr=10,mmc=10)


#Creating a new discriminant using eeg.discr.spec
```

```
cl<-eeg.discr.spec(Sim$data, Sim$reps , Sim$class, ma.s=10)
cl

#Simulating a test dataset with randeeg.
test <- randeeg(nclass=1,nrep=1,nobs=150,nel = 5, ar=c(0.3,0.1),
ma=c(0.7), order=matrix(c(2,0,1),1,3,byrow=TRUE), vars = c(1))

#Classifying the test signal
eeg.class.spec(test$data, cl)


#Creating a new discriminant using eeg.discr.cwt
cl<-eeg.discr.cwt(Sim$data, Sim$reps , Sim$class, mmc=10, mmr=10,abs=TRUE)
cl

#Simulating a test dataset with randeeg.
test <- randeeg(nclass=1,nrep=1,nobs=150,nel = 5, ar=c(0.3,0.1),
ma=c(0.7), order=matrix(c(2,0,1),1,3,byrow=TRUE), vars = c(1))

#Classifying the test signal
eeg.class.cwt(test$data, cl)
```

`classifyEEG` *Classifies a new sample of EEG data.*

Description

Classifies a new sample of EEG data based on an object produced by svmEEG.

Usage

```
classifyEEG(y,data)
```

Arguments

`y`	an object produced by svmEEG.
`data`	a new data set. Each column must represent each electrode.

Details

See `FeatureEEG` for more details.

Value

`pred`	a vector with the predicted class and an associated probability.

Author(s)

Murilo Coutinho

References

Bostanov, V. (2004) *BCI Competition 2003 - Data Sets Ib and IIb: Feature Extraction From Event-Related Brain Potentials With the Continuous Wavelet Transform and the t-Value Scalogram.* IEEE transactions on biomedical engineering, V. 51, no. 6.

Hastie, T., Tibshirani, R., Friedman, J. (2009) *The Elements of Statistical Learning: Data Mining, Inference, and Prediction.* 2nd ed. Stanford: Springer.

Coutinho, M., Borries, G.F., Borries, R.F. (2011) *EEG data classification using Support Vector Machine and Fourier data analysis, a new approach.*

See Also

`FeatureEEG`, `Feature3EEG`, `svmEEG`

Examples

```
library(eegclass)

###Producing the data set.
Sim <- randeeg(nclass=2,nrep=20,nobs=150,nel = 5,
ar=matrix(c(0.3,0.1,0,0),2,2, byrow=TRUE),
ma=matrix(c(0.7,0),2,1,byrow=TRUE), order=matrix(c(2,0,1,0,0,0),
2,3, byrow=TRUE), vars = c(1,1))

###Selecting the features
x<-FeatureEEG(Sim$data,Sim$reps,Sim$class,nselec=c(12,12),featype=
c('f1','f3'), win=c(20,20),stat=c('sum','sum'),power=c(2,1),abs=
c(FALSE,FALSE),log=c(FALSE,FALSE), mintomax=c(TRUE,TRUE),Alpha=
c(0.05,0.05), AlphaCorr=0.9,minacc=rep(0.7,2))

###Producing the classifier
y<-svmEEG(x)


###Generating a test data set of the first class
test <- randeeg(nclass=1,nrep=1,nobs=150,nel = 5, ar=
matrix(c(0.3,0.1),1,2,byrow=TRUE), ma=0.7, order=
matrix(c(2,0,1),1,3, byrow=TRUE), vars = 1)

###Classifying the test data
classifyEEG(y,data=test$data)

###Generating a test data set of the second class
test <- randeeg(nclass=1,nrep=1,nobs=150,nel = 5, ar=
matrix(c(0,0),1,2,byrow=TRUE), ma=0, order=matrix(
c(0,0,0),1,3, byrow=TRUE), vars = 1)

###Classifying the test data
classifyEEG(y,data=test$data)
```

`eeg.class.cwt` *Classifies a new signal in one of two classes.*

Description

Classifies a new signal in one of two classes given an object of "EEGCWTdiscr.All" class or "EEGCWTdiscr.Byfreq" class produced by `eeg.discr.cwt`.

Usage

```
eeg.class.cwt(newdata, disc)
```

Arguments

`newdata`	is a new data frame that contains the EEG signals to be classified. The data frame must be organized as follows: each column represents a different electrode so that each electrical potential collected by each electrode are represented in each row.
`disc`	an object of "EEGdiscr.All" class or "EEGdiscr.Byfreq" class produced by `eeg.discr.spec`.

Author(s)

Murilo Coutinho

References

Bostanov, V. (2004) *BCI Competition 2003 - Data Sets Ib and IIb: Feature Extraction From Event-Related Brain Potentials With the Continuous Wavelet Transform and the t-Value Scalogram.* IEEE transactions on biomedical engineering, V. 51, no. 6.

Hastie, T., Tibshirani, R., Friedman, J. (2009) *The Elements of Statistical Learning: Data Mining, Inference, and Prediction.* 2nd ed. Stanford: Springer.

Coutinho, M., Borries, G.F., Borries, R.F. (2011) *EEG data classification using Support Vector Machine and Fourier data analysis, a new approach.*

See Also

`svm`, `eeg.discr.spec`, `wavCWT`

Examples

```
library(eegclass)

#Simulating a dataset with randeeg.
Sim <- randeeg(nclass=2,nrep=4,nobs=150,nel = 5, ar=
matrix(c(0.3,0.1,0,0),2,2,byrow=TRUE), ma=matrix(c(0.7,0),
2,1,byrow=TRUE), order=matrix(c(2,0,1,0,0,0),2,3,
byrow=TRUE), vars = c(2,1))
```

```
#Creating a new discriminant using eeg.discr.cwt
cl<-eeg.discr.cwt(Sim$data, Sim$reps , Sim$class, mmc=10, mmr=10,abs=TRUE)
cl

#Simulating a test dataset with randeeg.
test <- randeeg(nclass=1,nrep=1,nobs=150,nel = 5, ar=c(0.3,0.1),
ma=c(0.7), order=matrix(c(2,0,1),1,3,byrow=TRUE), vars = c(1))

#Classifying the test signal
eeg.class.cwt(test$data, cl)
```

`eeg.class.spec` *Classifies a new signal in one of two classes.*

Description

Classifies a new signal in one of two classes given an object of "EEGdiscr.All" class or "EEGdiscr.Byfreq" class produced by `eeg.discr.spec`.

Usage

```
eeg.class.spec(newdata, disc)
```

Arguments

`newdata`	is a new data frame that contains the EEG signals to be classified. The data frame must be organized as follows: each column represents a different electrode so that each electrical potential collected by each electrode are represented in each row.
`disc`	an object of "EEGdiscr.All" class or "EEGdiscr.Byfreq" class produced by `eeg.discr.spec`.

Author(s)

Murilo Coutinho

References

Bostanov, V. (2004) *BCI Competition 2003 - Data Sets Ib and IIb: Feature Extraction From Event-Related Brain Potentials With the Continuous Wavelet Transform and the t-Value Scalogram.* IEEE transactions on biomedical engineering, V. 51, no. 6.

Hastie, T., Tibshirani, R., Friedman, J. (2009) *The Elements of Statistical Learning: Data Mining, Inference, and Prediction.* 2nd ed. Stanford: Springer.

Coutinho, M., Borries, G.F., Borries, R.F. (2011) *EEG data classification using Support Vector Machine and Fourier data analysis, a new approach.*

See Also

svm, eeg.discr.spec, spectrum

Examples

```
#Simulating a dataset with randeeg.
Sim <- randeeg(nclass=2,nrep=4,nobs=150,nel = 5, ar=
matrix(c(0.3,0.1,0,0),2,2, byrow=TRUE), ma=matrix(c(0.7,0),
2,1,byrow=TRUE), order=matrix(c(2,0,1,0,0,0),2,3, byrow=TRUE),
vars = c(1,1))

#Creating a new discriminant using eeg.discr.spec
cl<-eeg.discr.spec(Sim$data, Sim$reps , Sim$class, ma.s=10)
cl

#Simulating a test dataset with randeeg.
test <- randeeg(nclass=1,nrep=1,nobs=150,nel = 5, ar=c(0.3,0.1),
ma=c(0.7), order=matrix(c(2,0,1),1,3,byrow=TRUE), vars = c(1))

#Classifying the test signal
eeg.class.spec(test$data, cl)
```

---

eeg.discr.cwt *Creates a classification object.*

Description

Creates an object which contains all the information needed to classify a new signal in one of two classes based in a classification function that uses the continuous wavelet transform as feature extraction.

Usage

```
eeg.discr.cwt(data, reps, classes, mmc = 1, mmr = 1,
class.labels = c(1, 2), cost = 1, Rho = 2, byfreq = T,
alpha = 0.99, wavelet = "gaussian2", variance = 1,
abs = F, test = "t.test", type = "p.value", kernel = "linear")
```

Arguments

data	data is the data frame containing the EEG signals. The data frame must be organized as follows: each column represents a different electrode so that each electrical potential collected by each electrode is represented in each row. To identify which class and which replicate each line represents, the vectors classes and reps are given and must have length equal to the number of rows in the data frame.

classes	is a vector with length equal to the number of rows in data. Thus each value in the array identifies the class of each stimulus in each row of the database. For example, let classes <- c (1,1,1,1,1,2,2,2,2,2), this means that the first 5 rows of data represents the first class and the lines 6 to 10 represent the second class.
reps	is a vector with length equal to the number of rows in data. Thus each value in the array identifies the replicate of each stimulus in each row of the database. For example, let reps <- c (1,1,1,1,1,2,2,2,2,2), this means that the first 5 rows of data represents the first replicate of some stimulus and the lines 6 to 10 represent the second replicate of the stimulus. reps must be numeric and numerated from 1 to the total number of replicates for each class.
mmr	is the moving average parameter for the rows of the continuous wavelet transform matrix.
mmc	is the moving average parameter for the columns of the continuous wavelet transform matrix.
class.labels	a vector containing the representation of each class in the vector classes.
byfreq	if TRUE uses a different svm function for each entry of the continuous wavelet transform matrix and then uses a weight system to define the final decision. If FALSE uses only one svm function for all entries of the continuous wavelet transform matrix.
Rho	parameter used in the weight system. Used only if byfreq=TRUE.
wavelet	which wavelet is used. See wavCWT for more details.
abs	if TRUE then the absolute value of the continuous wavelet transform matrix is used.
variance	The variance parameter for the continuous wavelet transform. See wavCWT for more details.
test	defines the statistic used to extract the features. If test="ALL" then all entries of the continuous wavelet transform matrix are used. If test="t.test" then the t statistic is used. If test="f.test" then the F statistic is used. If test="wilcox" then the wilcox statistic is used.
type	is the type of feature extraction. If type="p.value" then the features are selected based on the p-value obtained using the statistic selected with the parameter test. If type="max" then the features are selected based on the maximum value obtained using the statistic selected with the parameter test. If type="max.el" then the features are selected based on the maximum value obtained, for each electrode, using the statistic selected with the parameter test. If type="local.max" then the program finds the all the local maximum of the scalogram obtained using the statistic selected with the parameter test.
alpha	if type="p.value" then the significance level for the selected features is 1-alpha.
kernel	which kernel is used for the support vector machine classifier. If byfreq=T then only kernel="linear" can be used. See svm for more information.
cost	the cost parameter used for the support vector machine classifier. See svm for more information.

Value

`model`	if `byfreq=F` then `model` is the SVM object produced by `svm`.
`pars`	if `byfreq=T` then `pars` is a data frame containing all the necessary information to classify a new signal.
`w`	Is a vector indicating which frequencies were selected by the program based on the parameters `test` and `type`.

Author(s)

Murilo Coutinho

References

Bostanov, V. (2004) *BCI Competition 2003 - Data Sets Ib and IIb: Feature Extraction From Event-Related Brain Potentials With the Continuous Wavelet Transform and the t-Value Scalogram.* IEEE transactions on biomedical engineering, V. 51, no. 6.

Hastie, T., Tibshirani, R., Friedman, J. (2009) *The Elements of Statistical Learning: Data Mining, Inference, and Prediction.* 2nd ed. Stanford: Springer.

Coutinho, M., Borries, G.F., Borries, R.F. (2011) *EEG data classification using Support Vector Machine and Fourier data analysis, a new approach.*

See Also

`svm`, `wavCWT`, `svmcwt.cv`, `eeg.class.cwt`

Examples

```
#Simulating a dataset with randeeg.
Sim <- randeeg(nclass=2,nrep=4,nobs=150,nel = 5, ar=
matrix(c(0.3,0.1,0,0),2,2, byrow=TRUE), ma=matrix(c(0.7,0),2,
1,byrow=TRUE), order=matrix(c(2,0,1,0,0,0),2,3, byrow=TRUE),
vars = c(2,1))

#Creating a new discriminant using eeg.discr.cwt
cl<-eeg.discr.cwt(Sim$data, Sim$reps , Sim$class, mmc=10, mmr=10,abs=TRUE)
cl

#Simulating a test dataset with randeeg.
test <- randeeg(nclass=1,nrep=1,nobs=150,nel = 5, ar=c(0.3,0.1),
ma=c(0.7), order=matrix(c(2,0,1),1,3,byrow=TRUE), vars = c(1))

#Classifying the test signal
eeg.class.cwt(test$data, cl)
```

`eeg.discr.spec` *Build a classification function*

Description

Creates an object which contains all the information needed to classify a new signal in one of two classes based in a classification function which use the spectrum as feature extraction.

Usage

```
eeg.discr.spec(data, reps, classes, ma = 1, ma.s = 1,
class.labels = c(1, 2), cost = 1, Rho = 2, byfreq = T,
alpha = 0.99, test = "t.test", type = "p.value",
kernel = "linear")
```

Arguments

`data`	data is the data frame containing the EEG signals. The data frame must be organized as follows: each column represents a different electrode so that each electrical potential collected by each electrode is represented in each row. To identify which class and which replicate each line represents, the vectors `classes` and `reps` are given and must have length equal to the number of rows in the data frame.
`classes`	is a vector with length equal to the number of rows in `data`. Thus each value in the array identifies the class of each stimulus in each row of the database. For example, let `classes <- c (1,1,1,1,1,2,2,2,2,2)`, this means that the first 5 rows of `data` represents the first class and the lines 6 to 10 represent the second class.
`reps`	is a vector with length equal to the number of rows in `data`. Thus each value in the array identifies the replicate of each stimulus in each row of the database. For example, let `reps <- c (1,1,1,1,1,2,2,2,2,2)`, this means that the first 5 rows of `data` represents the first replicate of some stimulus and the lines 6 to 10 represent the second replicate of the stimulus. `reps` must be numeric and numerated from 1 to the total number of replicates for each class.
`ma`	is the moving average parameter for the signals.
`ma.s`	is the moving average parameter for the spectrum.
`class.labels`	a vector containing the representation of each class in the vector `classes`.
`byfreq`	if TRUE uses a different svm function for each frequency of the spectrum and then uses a weight system to define the final decision. If FALSE uses only one svm function for all frequencies of the spectrum.
`Rho`	parameter used in the weight system. Used only if `byfreq=TRUE`.
`test`	defines the statistic used to extract the features. If `test="ALL"` then all the frequencies of the spectrum are used. If `test="t.test"` then the t statistic is used. If `test="f.test"` then the F statistic is used. If `test="wilcox"` then the wilcox statistic is used.

type	is the type of feature extraction. If type="p.value" then the features are selected based on the p-value obtained using the statistic selected with the parameter test. If type="max" then the features are selected based on the maximum value obtained using the statistic selected with the parameter test. If type="max.el" then the features are selected based on the maximum value obtained, for each electrode, using the statistic selected with the parameter test.
alpha	if type="p.value" then the significance level for the selected features is 1-alpha.
kernel	which kernel is used for the support vector machine classifier. If byfreq=T then only kernel="linear" can be used. See svm for more information.
cost	the cost parameter used for the support vector machine classifier. See svm for more information.

Value

model	if byfreq=F then model is the SVM object produced by svm.
pars	if byfreq=T then pars is a data frame containing all the necessary information to classify a new signal.
w	Is a vector indicating which frequencies were selected by the program based on the parameters test and type.

Author(s)

Murilo Coutinho

References

Bostanov, V. (2004) *BCI Competition 2003 - Data Sets Ib and IIb: Feature Extraction From Event-Related Brain Potentials With the Continuous Wavelet Transform and the t-Value Scalogram.* IEEE transactions on biomedical engineering, V. 51, no. 6.

Hastie, T., Tibshirani, R., Friedman, J. (2009) *The Elements of Statistical Learning: Data Mining, Inference, and Prediction.* 2nd ed. Stanford: Springer.

Coutinho, M., Borries, G.F., Borries, R.F. (2011) *EEG data classification using Support Vector Machine and Fourier data analysis, a new approach.*

See Also

svm, svmspec.cv, spectrum, eeg.class.spec

Examples

```
#Simulating a dataset with randeeg.
Sim <- randeeg(nclass=2,nrep=4,nobs=150,nel = 5, ar=
matrix(c(0.3,0.1,0,0),2,2, byrow=TRUE), ma=matrix(c(0.7,0),2,
1,byrow=TRUE), order=matrix(c(2,0,1,0,0,0),2,3, byrow=TRUE),
 vars = c(1,1))
```

```
#Creating a new discriminant using eeg.discr.spec
cl<-eeg.discr.spec(Sim$data, Sim$reps , Sim$class, ma.s=10)
cl

#Simulating a test dataset with randeeg.
test <- randeeg(nclass=1,nrep=1,nobs=150,nel = 5, ar=c(0.3,0.1),
ma=c(0.7), order=matrix(c(2,0,1),1,3,byrow=TRUE), vars = c(1))

#Classifying the test signal
eeg.class.spec(test$data, cl)
```

`Feature3EEG` *Select features to classify EEG data having 3 classes.*

Description

Using several statistical tests, the algorithm selects the features which have the greatest chance of construct a classification algorithm with good classification rates.

Usage

```
Feature3EEG(data,reps,classes,nselec,featype,ncomps=1,
win,stat,power,abs,log,mintomax,Alpha, AlphaCorr=0.95,minacc,
 wavelet="gaussian2", variance=1,Nfea=10)
```

Arguments

`data`	data is the data frame containing the EEG signals. The data frame must be organized as follows: each column represents a different electrode so that each electrical potential collected by each electrode is represented in each row. To identify which class and which replicate each line represents, the vectors `classes` and `reps` are given and must have length equal to the number of rows in the data frame.
`classes`	is a vector with length equal to the number of rows in `data`. Thus each value in the array identifies the class of each stimulus in each row of the database. For example, let `classes <- c (1,1,1,1,1,2,2,2,2,2,3,3,3,3,3)`, this means that the first 5 rows of `data` represents the first class and the lines 6 to 10 represent the second class and so on.
`reps`	is a vector with length equal to the number of rows in `data`. Thus each value in the array identifies the replicate of each stimulus in each row of the database. For example, let `reps <- c (1,1,1,1,1,2,2,2,2,2)`, this means that the first 5 rows of `data` represents the first replicate of some stimulus and the lines 6 to 10 represent the second replicate of the stimulus. `reps` must be numeric and numerated from 1 to the total number of replicates for each class.
`nselec`	is a vector of size 2 representing the number of signals to be randomly choosen to be used in train phase I (see details for more information).

`featype`	is a string vector indicating the type of features to be tested. Using along with the parameters win, stat, power, abs, log and mintomax the user can define all the features to be tested by the feature selector. Several values are permitted: f1 - Denote the usage of the original signal by dividing the signal in windows (parameter win) and calculating statistics in each window. f2 - Denote the usage of the loadings of the principal component analysis. f3 - Denote the usage of the spectrum. f4 - Denote the usage of the loadings of the principal component analysis applied over the spectrum. f5 - Denote the usage of the series produced by the principal component analysis. f6 - Denote the usage of a double windowing of the original signal. f7 - Denote the usage of the continuous wavelet transform.
`ncomps`	denote the number of principal components to be calculated if the features chosen by the user is f2 or f4.
`win`	is a vector indicating the size of the window if featype is f1, f3,f5, f6. This vector has not necessarily the same size of vector featype. For example, we can have featype=c('f1','f1','f2') and win=c(10,15).
`stat`	is a vector indicating the statistic to be calculated in each window if featype is f1, f3,f5, f6. This vector has the same size of the vector win.
`power`	is a vector indicating the power to be calculated in each window if featype is f1, f3,f5, f6 according with the formula log(abs(data^power)+1). This vector has the same size of the vector win.
`abs`	is a boolean vector indicating whether or not the absolute value should be calculated each window if featype is f1, f3,f5, f6 according with the formula log(abs(data^power)+1). This vector has the same size of the vector win.
`log`	is a boolean vector indicating whether or not the log should be calculated each window if featype is f1, f3,f5, f6 according with the formula log(abs(data^power)+1). This vector has the same size of the vector win.
`mintomax`	is a boolean vector indicating whether or not the vector of statistics calculated from each window should be sorted to the comparison be performed in terms of quantiles.
`Alpha`	significance level for the FDR test. See details.
`AlphaCorr`	significance level for the correlation test. See details.
`minacc`	minimum accuracy to be achieved by the SVM classifier in training phase I. See details.
`wavelet`	the wavelet type used.
`variance`	variance parameter for the wavelet used.
`Nfea`	the maximum number of features types selected by the feature selector for each pair of classes.

Details

The feature selector is a statistical algorithm which performs several tests to select the best features to classify a new data set in the future. The feature selection and training are performed in two phases:

Phase I:

1- The training set is divided in two sets. The number of signals of each class in each set is defined by the parameter nselec. 2- The T statistic is calculated for each feature type and a false discovery rate (FDR) test, with significance level given by the parameter Alpha, is performed to select the features with the greatest differences between classes. 3- For each feature selected a SVM classifier is trained individually. 4- The features are extracted from the second set and classified. The features which obtain a correct classification rate of at least minacc (parameter) are selected. 5- A correlation test are performed to eliminate redundant features. 6- This procedure is used for each pair of classes.

Phase II:

Using the selected features from FeratureEEG, the function svmEEG() trains a SVM classifier.

See ?svmEEG.

Value

x	An object to be used in svmEEG.

Author(s)

Murilo Coutinho

References

Bostanov, V. (2004) *BCI Competition 2003 - Data Sets Ib and IIb: Feature Extraction From Event-Related Brain Potentials With the Continuous Wavelet Transform and the t-Value Scalogram.* IEEE transactions on biomedical engineering, V. 51, no. 6.

Hastie, T., Tibshirani, R., Friedman, J. (2009) *The Elements of Statistical Learning: Data Mining, Inference, and Prediction.* 2nd ed. Stanford: Springer.

Coutinho, M., Borries, G.F., Borries, R.F. (2011) *EEG data classification using Support Vector Machine and Fourier data analysis, a new approach.*

See Also

svmEEG, classifyEEG

Examples

```
library(eegclass)

###Producing the data set.
Sim <- randeeg(nclass=3,nrep=20,nobs=150,nel = 5, ar=
matrix(c(0.3,0.1,0,0,0.8,-0.3),3,2, byrow=TRUE), ma=
matrix(c(0.7,0,-0.3),3,1,byrow=TRUE), order=matrix(c(2,0,1,0,
0,0,2,0,1),3,3, byrow=TRUE), vars = c(1,1,2))

###Selecting the features
x<-Feature3EEG(Sim$data, Sim$reps, Sim$class, nselec=c(12,12),
featype= c('f1','f3'), win=c(20,20), stat=c('sum','sum'),power=c(2,1),
abs=c(FALSE,FALSE), log=c(FALSE,FALSE), mintomax=c(TRUE,TRUE),
Alpha=c(0.05,0.05), AlphaCorr=0.9,minacc=rep(0.7,2))
```

```
###Producing the classifier
y<-svmEEG(x)


###Generating a test data set of the first class
test <- randeeg(nclass=1,nrep=1,nobs=150,nel = 5, ar=
matrix(c(0.3,0.1),1,2, byrow=TRUE), ma=0.7, order=matrix(c(2,0,
1),1,3,byrow=TRUE), vars = 1)

###Classifying the test data
classifyEEG(y,data=test$data)

###Generating a test data set of the second class
test <- randeeg(nclass=1,nrep=1,nobs=150,nel = 5, ar=
matrix(c(0,0),1,2,  byrow=TRUE), ma=0, order=matrix(c(0,0,0),
1,3, byrow=TRUE), vars = 1)

###Classifying the test data
classifyEEG(y,data=test$data)


###Generating a test data set of the third class
test <- randeeg(nclass=1,nrep=1,nobs=150,nel = 5, ar=
matrix(c(0.8,-0.3),1,2, byrow=TRUE), ma=-0.3, order=matrix(c(2,
0,1),1,3, byrow=TRUE), vars = 2)

###Classifying the test data
classifyEEG(y,data=test$data)
```

`FeatureEEG` *Select features to classify EEG data.*

Description

Using several statistical tests, the algorithm selects the features which have the greatest chance of construct a classification algorithm with good classification rates.

Usage

```
FeatureEEG(data,reps,classes,nselec,featype,ncomps=1,
win,stat,power,abs,log,mintomax,Alpha, AlphaCorr=0.95,minacc,
 wavelet="gaussian2", variance=1,Nfea=10)
```

Arguments

`data`	is the data frame that contains the EEG signals. The data frame must be organized as follows: each column represents a different electrode so that each electrical potential collected by each electrode are represented in each row. To identify which class and which replicate each line represents, the vectors `classes`

and `reps` must be given and must have length equal to the number of rows in the data frame.

`classes` is a vector with length equal to the number of rows in `data`. Thus each value in the array identifies the class of each stimulus in each row of the database. For example, let `classes <- c (1,1,1,1,1,2,2,2,2,2)`, this means that the first 5 rows of `data` represents the first class and the lines 6 to 10 represent the second class.

`reps` is a vector with length equal to the number of rows in `data`. Thus each value in the array identifies the replicate of each stimulus in each row of the database. For example, let `reps <- c (1,1,1,1,1,2,2,2,2,2)`, this means that the first 5 rows of `data` represents the first replicate of some stimulus and the lines 6 to 10 represent the second replicate of the stimulus. `reps` must be numeric and numerated from 1 to the total number of replicates for each class.

`nselec` is a vector of size 2 representing the number of signals to be randomly choosen to be used in train phase I (see details for more information).

`featype` is a string vector indicating the type of features to be tested. Using along with the parameters win, stat, power, abs, log and mintomax the user can define all the features to be tested by the feature selector. Several values are permitted: f1 - Denote the usage of the original signal by dividing the signal in windows (parameter wind) and calculating statistics in each window. f2 - Denote the usage of the loadings of the principal component analysis. f3 - Denote the usage of the spectrum. f4 - Denote the usage of the loadings of the principal component analysis applied over the spectrum. f5 - Denote the usage of the series produced by the principal component analysis. f6 - Denote the usage of a double windowing of the original signal. f7 - Denote the usage of the continuous wavelet transform.

`ncomps` denote the number of principal components to be calculated if the features chosen by the user is f2 or f4.

`win` is a vector indicating the size of the window if featype is f1, f3,f5, f6. This vector has not necessarily the same size of vector featype. For example, we can have featype=c('f1','f1','f2') and win=c(10,15).

`stat` is a vector indicating the statistic (sum, mean, var, prod, min, max, sd, geometric, harmonic) to be calculated in each window if featype is f1, f3,f5, f6. This vector has the same size of the vector win.

`power` is a vector indicating the power to be calculated in each window if featype is f1, f3,f5, f6 according with the formula log(abs(data^power)+1). This vector has the same size of the vector win.

`abs` is a boolean vector indicating whether or not the absolute value should be calculated each window if featype is f1, f3,f5, f6 according with the formula log(abs(data^power)+1). This vector has the same size of the vector win.

`log` is a boolean vector indicating whether or not the log should be calculated each window if featype is f1, f3,f5, f6 according with the formula log(abs(data^power)+1). This vector has the same size of the vector win.

`mintomax` is a boolean vector indicating whether or not the vector of statistics calculated from each window should be sorted to the comparison be performed in terms of quantiles.

Alpha	a vector with the significance level for each FDR test to be performed. See details.
AlphaCorr	significance level for the correlation test. See details.
minacc	minimum accuracy to be achieved by the SVM classifier in training phase I. See details.
wavelet	the wavelet type used.
variance	variance parameter for the wavelet used.
Nfea	the maximum number of features types selected by the feature selector.

Details

The feature selector is a statistical algorithm which performs several tests to select the best features to classify a new data set in the future. The feature selection and training are performed in two phases:

Phase I:

1- The training set is divided in two sets. The number of signals of each class in each set is defined by the parameter nselec. 2- The T statistic is calculated for each feature type and a false discovery rate (FDR) test, with significance level given by the parameter Alpha, is performed to select the features with the greatest differences between classes. 3- For each feature selected a SVM classifier is trained individually. 4- The features are extracted from the second set and classified. The features which obtain a correct classification rate of at least minacc (parameter) are selected. 5- A correlation test are performed to eliminate redundant features. Phase II:

Using the selected features from FeratureEEG, the function svmEEG() trains a SVM classifier.

See ?svmEEG.

Value

x	An object to be used in svmEEG.

Author(s)

Murilo Coutinho

References

Bostanov, V. (2004) *BCI Competition 2003 - Data Sets Ib and IIb: Feature Extraction From Event-Related Brain Potentials With the Continuous Wavelet Transform and the t-Value Scalogram.* IEEE transactions on biomedical engineering, V. 51, no. 6.

Hastie, T., Tibshirani, R., Friedman, J. (2009) *The Elements of Statistical Learning: Data Mining, Inference, and Prediction.* 2nd ed. Stanford: Springer.

Coutinho, M., Borries, G.F., Borries, R.F. (2011) *EEG data classification using Support Vector Machine and Fourier data analysis, a new approach.*

See Also

svmEEG, classifyEEG

Examples

```
library(eegclass)

###Producing the data set.
Sim <- randeeg(nclass=2,nrep=20,nobs=150,nel = 5, ar=
matrix(c(0.3,0.1,0,0),2,2, byrow=TRUE), ma=matrix(c(0.7,0),
2,1,byrow=TRUE), order=matrix(c(2,0,1,0,0,0),2,3, byrow=TRUE),
vars = c(1,1))

###Selecting the features
x<-FeatureEEG(Sim$data,Sim$reps,Sim$class,nselec=c(12,12),featype=
c('f1','f3'), win=c(20,20),stat=c('sum','sum'),power=c(2,1),abs=
c(FALSE,FALSE),log=c(FALSE,FALSE), mintomax=c(TRUE,TRUE),Alpha=
c(0.05,0.05), AlphaCorr=0.9,minacc=rep(0.7,2))

###Producing the classifier
y<-svmEEG(x)


###Generating a test data set of the first class
test <- randeeg(nclass=1,nrep=1,nobs=150,nel = 5, ar=
matrix(c(0.3,0.1),1,2, byrow=TRUE), ma=0.7, order=matrix(c(2,0,1),
1,3,byrow=TRUE), vars = 1)

###Classifying the test data
classifyEEG(y,data=test$data)

###Generating a test data set of the second class
test <- randeeg(nclass=1,nrep=1,nobs=150,nel = 5, ar= matrix(c(0,0),
1,2,byrow=TRUE), ma=0, order=matrix(c(0,0,0),1,3, byrow=TRUE), vars = 1)

###Classifying the test data
classifyEEG(y,data=test$data)
```

`ploteeg` *Plot EEG data*

Description

This function was designed to do different types of charts of EEG data. Graphs of the original data as well as the spectrum, continuous wavelet transform and t-value scalogram of the signals can be plotted.

Usage

```
ploteeg(data, reps, classes, which.reps = "ALL", which.classes =
"ALL", which.el = "ALL", type = "original", ma = 1, ma.s = 1,
```

```
 n.col = 200, wavelet = "gaussian2", mmr = 1, mmc = 1, abs = F,
Real = T, variance = 1)
```

Arguments

data	data is the data frame containing the EEG signals. The data frame must be organized as follows: each column represents a different electrode so that each electrical potential collected by each electrode is represented in each row. To identify which class and which replicate each line represents, the vectors `classes` and `reps` are given and must have length equal to the number of rows in the data frame.
classes	is a vector with length equal to the number of rows in `data`. Thus each value in the array identifies the class of each stimulus in each row of the database. For example, let `reps <- c (1,1,1,1,1,2,2,2,2,2)`, this means that the first 5 rows of `data` represents the first class and the lines 6 to 10 represent the second class.
reps	is a vector with length equal to the number of rows in `data`. Thus each value in the array identifies the replicate of each stimulus in each row of the database. For example, let `reps <- c (1,1,1,1,1,2,2,2,2,2)`, this means that the first 5 rows of `data` represents the first replicate of some stimulus and the lines 6 to 10 represent the second replicate of the stimulus. `reps` must be numeric and numerated from 1 to the total number of replicates for each class.
which.reps	A list representing which replicates will be plotted. For example, if `which.reps= list(c(1,3,4),c(1,2,4))` then the replicates indicated in the vector `reps` as 1,3 and 4 will be plotted for the first class and the replicates indicated in the vector `reps` as 1,2 and 4 will be plotted for the second class. If `which.reps="ALL"` then all replicates will be plotted.
which.classes	A vector representing which classes will be plotted. For example, if `which.classes= c(1,3)` then the classes indicated in the vector `classes` as 1 and 3 will be plotted. If `which.classes= "ALL"` then all classes will be plotted. Obs: if `type = "T.pvalue"` then only two classes are allowed.
which.el	A vector representing which electrodes will be plotted. For example, if `which.el= c(1,3)` then the electrodes 1 and 3 will be plotted. If `which.el="ALL"` then all electrodes will be plotted.
type	if `type="original"` then the original signals will be plotted, if `type= "spectrum"` then the spectrum of each signal will be plotted, if `type="wavelet"` then the continuous wavelet transform matrix will be plotted, if `type="T.pvalue"` then the t-value scalogram of the signals will be plotted.
ma	is the moving average parameter for the signals. Used if `type="spectrum"` or if `type="original"`.
ma.s	is the moving average parameter for the spectrum. Used if `type="spectrum"`.
n.col	is the number of colors for in the contour plot, used if `type="wavelet"` or `type="T.pvalue"`.
wavelet	which wavelet is used. See `wavCWT` for more details.

mmr	is the moving average parameter for the rows of the continuous wavelet transform matrix. Used if type="wavelet" or type="T.pvalue".
mmc	is the moving average parameter for the columns of the continuous wavelet transform matrix. Used if type="wavelet" or type="T.pvalue".
abs	if TRUE then the absolute value of the continuous wavelet transform matrix is used. Used if type="wavelet" or type="T.pvalue".
Real	If TRUE takes the real part of the continuous wavelet transform matrix, if FALSE takes the imaginary part. Used if wavelet="morlet".
variance	The variance parameter for the continuous wavelet transform. See wavCWT for more details.

Author(s)

Murilo Coutinho

References

Bostanov, V. (2004) *BCI Competition 2003 - Data Sets Ib and IIb: Feature Extraction From Event-Related Brain Potentials With the Continuous Wavelet Transform and the t-Value Scalogram.* IEEE transactions on biomedical engineering, V. 51, no. 6.

Hastie, T., Tibshirani, R., Friedman, J. (2009) *The Elements of Statistical Learning: Data Mining, Inference, and Prediction.* 2nd ed. Stanford: Springer.

Coutinho, M., Borries, G.F., Borries, R.F. (2011) *EEG data classification using Support Vector Machine and Fourier data analysis, a new approach.*

See Also

wavCWT, spectrum

Examples

```
#Simulating a dataset with randeeg.
Sim <- randeeg(nclass=2,nrep=4,nobs=150,nel = 5, ar=
matrix(c(0.3,0.1,0,0),2,2, byrow=TRUE), ma=matrix(c(0.7,0),2,1,
byrow=TRUE), order=matrix(c(2,0,1,0,0,0),2,3, byrow=TRUE),
vars = c(1,1))

#Plotting the data.
ploteeg(Sim$data, Sim$reps , Sim$class , which.reps="ALL",
which.classes = "ALL", which.el=1, type = 'original')

#Plotting the spectrum of the data.
ploteeg(Sim$data, Sim$reps , Sim$class , which.reps="ALL",
which.classes = "ALL", which.el=1, type = 'spectrum', ma.s = 10)

#Plotting the wavelet of the data.
ploteeg(Sim$data, Sim$reps , Sim$class , which.reps="ALL",
which.classes = "ALL", which.el=1, type = 'wavelet',
wavelet="gaussian2",abs=TRUE)
```

```
#Plotting the T value scalogram (based on the T statistic).
ploteeg(Sim$data, Sim$reps , Sim$class , which.reps="ALL",
which.classes = "ALL", which.el="ALL", type ='T.pvalue',
wavelet="gaussian2",abs=TRUE)
```

`plotwindows` *Plot statistics of the EEG data calculated by windows*

Description

This function was designed to do different types of charts of EEG data. Basically, the original signal (for each electrode, class and replicate) is divided by windows and some statistical function (as the mean or variance) is used for all samples in each window. Then, the sequence of values obtained for each window is plotted.

Usage

```
plotwindows(data, reps , classes , which.reps="ALL", which.classes =
 "ALL", which.el="ALL", win=10, stat="sum", power = 2,
 abs=F,log=F,complete = F, mintomax=F)
```

Arguments

`data`	data is the data frame containing the EEG signals. The data frame must be organized as follows: each column represents a different electrode so that each electrical potential collected by each electrode is represented in each row. To identify which class and which replicate each line represents, the vectors `classes` and `reps` are given and must have length equal to the number of rows in the data frame.
`classes`	is a vector with length equal to the number of rows in `data`. Thus each value in the array identifies the class of each stimulus in each row of the database. For example, let `classes <- c(1,1,1,1,1,2,2,2,2,2)`, this means that the first 5 rows of `data` represents the first class and the lines 6 to 10 represent the second class.
`reps`	is a vector with length equal to the number of rows in `data`. Thus each value in the array identifies the replicate of each stimulus in each row of the database. For example, let `reps <- c(1,1,1,1,1,2,2,2,2,2)`, this means that the first 5 rows of `data` represents the first replicate of some stimulus and the lines 6 to 10 represent the second replicate of the stimulus. `reps` must be numeric and numerated from 1 to the total number of replicates for each class.
`which.reps`	A list representing which replicates will be plotted. For example, if `which.reps= list(c(1,3,4),c(1,2,4))` then the replicates indicated in the vector `reps` as 1,3 and 4 will be plotted for the first class and the replicates indicated in the vector `reps` as 1,2 and 4 will be plotted for the second class. If `which.reps= "ALL"` then all replicates will be plotted.

`which.classes`
: A vector representing which classes will be plotted. For example, if `which.classes= c(1,3)` then the classes indicated in the vector `classes` as 1 and 3 will be plotted. If `which.classes= "ALL"` then all classes will be plotted. Obs: if `type = "T.pvalue"` then only two classes are allowed.

`which.el`
: A vector representing which electrodes will be plotted. For example, if `which.el= c(1,3)` then the electrodes 1 and 3 will be plotted. If `which.el= "ALL"` then all electrodes will be plotted.

`win`
: The window size.

`stat`
: Is the statistic used for all observations (or samples) in each window. If `stat= "sum"` then the sum is used. If `stat="mean"` then the mean is used. If `stat="var"` then the variance is used. If `stat="sd"` then the standard deviation is used. If `stat="max"` then the maximum value is used. If `stat="min"` then the minimum value is used. If `stat="prod"` then the product is used. If `stat= "median"` then the median is used. If `stat="geometric"` then the geometric mean is used. If `stat="harmonic"` then the harmonic mean is used.

`power`
: A transformation of the data can be used before using the choosen statistic, in this case the transformation will be `data^power`. OBS: watch out for negative values if `power` is not an integer.

`abs`
: A transformation of the data can be used before using the choosen statistic, in this case, if `abs=TRUE`, then the trasformation will be `abs(data)^power`

`log`
: A transformation of the data can be used before using the choosen statistic, in this case, if `log=TRUE` and `abs=TRUE`, then the transformation will be `log(abs(data))^power`. OBS: watch out for negative values if `abs= FALSE`

`complete`
: If `complete=FALSE`, then the windows will be completely separate. For example, if `win=10` then the first window will consist of observations 1 to 10, the second window will consist of observations 11 to 20, and so on. If `complete=TRUE`, then some observations of each window will coincide. For example, if `win=10` then the first window will consist of observations 1 to 10, the second window will consist of observations 2 to 11, the third will consist of observations 3 to 12, and so on.

`mintomax`
: If `mintomax=FALSE` then the function will plot each value obtained for each window in order. If `mintomax=TRUE` then the function will sort the values obtained for each window and then plot these values from the minimum value to the maximum value.

Value

`data`
: Is a data frame with all the values computed by the function for each window. Arranged in the same way as the input parameter `data`.

`classes`
: Is a vector with the same meaning as the input parameter `classes` but related to the output parameter `data`.

`reps`
: Is a vector with the same meaning as the input parameter `reps` but related to the output parameter `data`.

Author(s)

Murilo Coutinho

References

Bostanov, V. (2004) *BCI Competition 2003 - Data Sets Ib and IIb: Feature Extraction From Event-Related Brain Potentials With the Continuous Wavelet Transform and the t-Value Scalogram.* IEEE transactions on biomedical engineering, V. 51, no. 6.

Hastie, T., Tibshirani, R., Friedman, J. (2009) *The Elements of Statistical Learning: Data Mining, Inference, and Prediction.* 2nd ed. Stanford: Springer.

Coutinho, M., Borries, G.F., Borries, R.F. (2011) *EEG data classification using Support Vector Machine and Fourier data analysis, a new approach.*

See Also

`ploteeg`

Examples

```
#Simulating a dataset with randeeg.
Sim <- randeeg(nclass=2,nrep=4,nobs=150,nel = 5, ar=
matrix(c(0.3,0.1,0,0),2,2,  byrow=TRUE), ma=matrix(c(0.7,0),2,1,
byrow=TRUE), order=matrix(c(2,0,1,0,0,0),2,3,byrow=TRUE), vars = c(2,1))

  data<-Sim$data
  reps<-Sim$reps
  classes<-Sim$class
  which.el<-c(1,5)
  which.reps=list(c(1,3,4),c(1,4))
  which.classes="ALL"

x<-plotwindows(data,reps,classes,which.reps=which.reps,which.classes=
which.classes ,  which.el= which.el,win=10,power=1,abs=TRUE,stat="sum",
complete=TRUE, mintomax=FALSE)

  print(x)
  summary(x)
  plot(x)
```

---

randeeg *Simulates data in the format require by other functions.*

Description

Create an object simulating ARIMA random variables. The created object cotains data in the format required to use other methods of this package. Makes a simulation similar to EEG data to test the capabilities those methods.

Usage

```
randeeg(nclass = 2, nrep = 10, nobs = 250, nel = 20, ar =
mat.or.vec(2, 1), ma = mat.or.vec(2, 1), order = mat.or.vec(2, 3),
vars = c(1, 2))
```

Arguments

nclass	number of different classes.
nrep	number of signals or replicates of each class.
nobs	number of observations or samples for each signal.
nel	number of electrodes in the simulation.
ar	is a matrix with AR coefficients of the ARIMA model. Each row contains the AR parameters for each class and the electrode. One can make the number of rows in ar equal to the number of classes and, in this case, the function uses the same coefficients for all electrodes of each class. Another option is to choose the number of rows equal to nclass*nel and, in this case, the function uses different ARIMA models for each class and each electrode. For example if we have 2 classes and ar=matrix(c(0.1,0.5,-0.2,0.3,0,0),2,3,byrow=T) then the first class will be simulated from an AR(3) and the second class will be simulated from an AR(1) with parameter 0.3.
ma	is a matrix with MA coefficients, defined the same way as in ar. See the ar parameter for more information.
order	is a matrix with the same number of rows of ar and ma but with 3 columns. Thus, each line forms a vector which is used in the function arima.sim. For more information see the parameter order in the function arima.sim.
vars	a vector with length equal to the number of rows in ar, ma and order. Denote the variance for each ARIMA model.

Value

data	The simulated data frame. See ploteeg for more details.
class	The vector indicating the classes for each row of data. See ploteeg for more details.
reps	The vector indicating the replicates for each row of data. See ploteeg for more details.
nrep	number of signals or replicates of each class.
nclass	number of different classes.
nobs	number of observations or samples for each signal.
nel	number of electrodes in the simulation.
vars	a vector with length equal to the number of rows in ar, ma and order. Denote the variance for each ARIMA model.

Author(s)

Murilo Coutinho

References

Brockwell, P.J., Davis, R.A. (2002) *Introduction to Time Series and Forecasting*. 2nd ed. Colorado: Springer. Cap. 4.

See Also

`ploteeg`, `arima.sim`

Examples

```
#Simulating a dataset with randeeg.
Sim <- randeeg(nclass=2,nrep=4,nobs=150,nel = 5, ar=
matrix(c(0.3,0.1,0,0),2,2, byrow=TRUE), ma=matrix(c(0.7,0),2,1,
byrow=TRUE), order=matrix(c(2,0,1,0,0,0),2,3, byrow=TRUE),
vars = c(1,1))

#Plotting the spectrum of the data.
ploteeg(Sim$data, Sim$reps , Sim$class , which.reps="ALL",
which.classes = "ALL", which.el=1, type = 'spectrum', ma.s = 10)
```

---

svmcwt.cv *Cross validation to estimate the correct classification rate.*

Description

Performs a cross validation to estimate the correct classification rate using the continuous wavelet transform as feature extraction.

Usage

```
svmcwt.cv(data, reps, classes, all = FALSE, numTest = c(1, 1),
n.iter = 100, mmc = 1, mmr = 1, class.labels = c(1, 2),
kernel = "linear", cost = 1, wavelet = "gaussian2", alpha = 0.99,
abs = F, Rho = 2, type = "p.value", test = "t.test", print = T,
byfreq = T, variance = 1)
```

Arguments

`data`	data is the data frame containing the EEG signals. The data frame must be organized as follows: each column represents a different electrode so that each electrical potential collected by each electrode is represented in each row. To identify which class and which replicate each line represents, the vectors `classes` and `reps` are given and must have length equal to the number of rows in the data frame.
`classes`	is a vector with length equal to the number of rows in `data`. Thus each value in the array identifies the class of each stimulus in each row of the database. For example, let `classes <- c (1,1,1,1,1,2,2,2,2,2)`, this means that the first 5 rows of `data` represents the first class and the lines 6 to 10 represent the second class.

`reps` is a vector with length equal to the number of rows in `data`. Thus each value in the array identifies the replicate of each stimulus in each row of the database. For example, let `reps <- c (1,1,1,1,1,2,2,2,2,2)`, this means that the first 5 rows of `data` represents the first replicate of some stimulus and the lines 6 to 10 represent the second replicate of the stimulus. `reps` must be numeric and numerated from 1 to the total number of replicates for each class.

`all` if TRUE svmcwt.cv performs the cross validation using one replicate of each class for the test phase and repeats this process for every possible combination of test signals. If FALSE the cross validation is performed specifying the parameters `numTest` and `n.iter`. See `details` for more information.

`numTest` is a vector representing the number of replicates of each class used in the test phase. For example, if `numTest=c(3,2)` then 3 replicates of the first class and 2 replicates of the second class are removed from the training phase and used in the testing phase. Only used if `all=FALSE`. See `details` for more information.

`n.iter` numeric, determines the maximum number of iterations performed by the cross validation. Only used if `all=FALSE`. See `details` for more information.

`print` if TRUE prints a message at the end of each iteration.

`mmr` is the moving average parameter for the rows of the continuous wavelet transform matrix.

`mmc` is the moving average parameter for the columns of the continuous wavelet transform matrix.

`class.labels` a vector containing the representation of each class in the vector `classes`.

`wavelet` which wavelet is used. See `wavCWT` for more details.

`abs` if TRUE then the absolute value of the continuous wavelet transform matrix is used.

`variance` The variance parameter for the continuous wavelet transform. See `wavCWT` for more details.

`byfreq` if TRUE uses a different svm function for each entry of the continuous wavelet transform matrix and then uses a weight system to defines the final decision. If FALSE uses only one svm function for all entries of the same matrix.

`Rho` parameter used in the weight system. Only used if `byfreq=TRUE`.

`test` defines the statistic used to extract the features. If `test="ALL"` then all the entries of the continuous wavelet transform matrix are used. If `test="t.test"` then the t statistic is used. If `test="f.test"` then the F statistic is used. If `test="wilcox"` then the wilcox statistic is used.

`type` is the type of feature extraction. If `type="p.value"` then the features are selected based on the p-value obtained using the statistic selected with the parameter `test`. If `type="max"` then the features are selected based on the maximum value obtained using the statistic selected with the parameter `test`. If `type="max.el"` then the features are selected based on the maximum value obtained, for each electrode, using the statistic selected with the parameter `test`. If `type="local.max"` then the program finds the all the local maximum of the scalogram obtained using the statistic selected with the parameter `test`.

alpha	if type="p.value" then the significance level for the selected features is 1-alpha.
kernel	which kernel is used for the support vector machine classifier. See svm for more information.
cost	the cost parameter used for the support vector machine classifier. See svm for more information.

Details

If all=FALSE the cross validation is performed as follows: 1- some of the replicates of all classes are randomly removed from the database (based on the parameter numTest) and the discriminant function is then calculated using the remaining replicates. 2- the removed replicates are used to test the discriminant. 3- this process is repeated n.iter times and the classification rate is estimated.

If all=TRUE the cross validation is performed as follows: 1- one replicate of each class are removed from the database and the discriminant function is then calculated using the remaining replicates. 2- the removed replicates are used to test the discriminant. 3- this process is repeated until all possible pairs of replicates are removed from the training phase and used in the test phase. 4- the correct classification rate is estimated.

Value

c.matrix	Confusion matrix.
acc	accuracy, the correct classification rate.

Author(s)

Murilo Coutinho

References

Bostanov, V. (2004) *BCI Competition 2003 - Data Sets Ib and IIb: Feature Extraction From Event-Related Brain Potentials With the Continuous Wavelet Transform and the t-Value Scalogram.* IEEE transactions on biomedical engineering, V. 51, no. 6.

Hastie, T., Tibshirani, R., Friedman, J. (2009) *The Elements of Statistical Learning: Data Mining, Inference, and Prediction.* 2nd ed. Stanford: Springer.

Coutinho, M., Borries, G.F., Borries, R.F. (2011) *EEG data classification using Support Vector Machine and Fourier data analysis, a new approach.*

See Also

svm, wavCWT, eeg.discr.cwt

Examples

```
#Simulating a dataset with randeeg.
Sim <- randeeg(nclass=2,nrep=4,nobs=150,nel = 5, ar=
matrix(c(0.3,0.1,0,0),2,2, byrow=TRUE), ma=matrix(c(0.7,0),2,1,
byrow=TRUE), order=matrix(c(2,0,1,0,0,0),2,3, byrow=TRUE),
```

```
vars = c(2,1))

#Making a cross validation to estimate the correct classification
#rate, using the continuous wavelet transform as feature extraction.
svmcwt.cv(Sim$data, Sim$reps , Sim$class , all = TRUE ,
abs=TRUE,mmr=10,mmc=10)
```

`svmEEG` *Produces an object to classify new EEG signals.*

Description

Given the features selected by FeatureEEG or Feature3EEG, svmEEG produces an object to classify new EEG signals.

Usage

```
svmEEG(x,kernel="radial",degree=3,gamma="Default",coef0=0,cost=1)
```

Arguments

`x`	an object produced by FeatureEEG or Feature3EEG.
`kernel`	See `svm`. the kernel used in training and predicting. You might consider changing some of the following parameters, depending on the kernel type. linear: u'*v polynomial: (gamma*u'*v + coef0)^degree radial basis: exp(-gamma*|u-v|^2) sigmoid: tanh(gamma*u'*v + coef0)
`degree`	See `svm`. parameter needed for kernel of type polynomial (default: 3)
`gamma`	See `svm`. parameter needed for all kernels except linear (default: 1/(data dimension))
`coef0`	See `svm`. parameter needed for kernels of type polynomial and sigmoid (default: 0)
`cost`	See `svm`. cost of constraints violation (default: 1)—it is the 'C'-constant of the regularization term in the Lagrange formulation.

Details

The feature selector is a statistical algorithm which performs several tests to select the best features to classify in the future. The feature selection and training are performed in two phases:

Phase I:

1- The training set is divided in two sets. The number of signals of each class in each set is defined by the parameter nselec. 2- The T statistic is calculated for each feature type and a false discovery

rate (FDR) test, with significance level given by the parameter Alpha, is performed to select the features with the greatest differences between classes. 3- For each feature selected a SVM classifier is trained individually. 4- The features are extracted from the second set and classified. The features which obtain a correct classification rate of at least minacc (parameter) are selected. 5- A correlation test are performed to eliminate redundant features. 6- This procedure is used for each pair of classes.

See ?FeatureEEG.

Phase II:

Using the selected features from FeratureEEG, the function svmEEG() trains a SVM classifier.

Value

y	An object to be used in classifyEEG.

Author(s)

Murilo Coutinho

References

Bostanov, V. (2004) *BCI Competition 2003 - Data Sets Ib and IIb: Feature Extraction From Event-Related Brain Potentials With the Continuous Wavelet Transform and the t-Value Scalogram.* IEEE transactions on biomedical engineering, V. 51, no. 6.

Hastie, T., Tibshirani, R., Friedman, J. (2009) *The Elements of Statistical Learning: Data Mining, Inference, and Prediction.* 2nd ed. Stanford: Springer.

Coutinho, M., Borries, G.F., Borries, R.F. (2011) *EEG data classification using Support Vector Machine and Fourier data analysis, a new approach.*

See Also

`FeatureEEG`, `Feature3EEG`, `classifyEEG`

Examples

```
library(eegclass)

###Producing the data set.
Sim <- randeeg(nclass=2,nrep=20,nobs=150,nel = 5, ar=
matrix(c(0.3,0.1,0,0),2,2, byrow=TRUE), ma=matrix(c(0.7,0),2,1,
byrow=TRUE), order=matrix(c(2,0,1,0,0,0),2,3, byrow=TRUE),
vars = c(1,1))

###Selecting the features
x<-FeatureEEG(Sim$data,Sim$reps, Sim$class, nselec=c(12,12),
featype=c('f1','f3'), win=c(20,20),stat=c('sum','sum'),
power=c(2,1),abs=c(FALSE,FALSE),log=c(FALSE,FALSE), mintomax=
c(TRUE,TRUE),Alpha=c(0.05,0.05), AlphaCorr=0.9,minacc=rep(0.7,2))

###Producing the classifier
```

```
y<-svmEEG(x)


###Generating a test data set of the first class
test <- randeeg(nclass=1,nrep=1,nobs=150,nel = 5, ar=matrix(c(0.3,0.1),1,2,
    byrow=TRUE), ma=0.7, order=matrix(c(2,0,1),1,3,
    byrow=TRUE), vars = 1)

###Classifying the test data
classifyEEG(y,data=test$data)

###Generating a test data set of the second class
test <- randeeg(nclass=1,nrep=1,nobs=150,nel = 5, ar=matrix(c(0,0),1,2,
byrow=TRUE), ma=0, order=matrix(c(0,0,0),1,3,
byrow=TRUE), vars = 1)

###Classifying the test data
classifyEEG(y,data=test$data)
```

`svmspec.cv` *Performs a cross validation to estimate the correct classification rate*

Description

Performs a cross validation to estimate the correct classification rate using the spectrum as feature extraction.

Usage

```
svmspec.cv(data, reps, classes, all = FALSE, numTest = c(1, 1),
n.iter = 100, ma = 1, ma.s = 1, class.labels = c(1, 2), kernel =
"linear", cost = 1, Rho = 2, print = T, byfreq = T, alpha = 0.99,
test = "t.test", type = "p.value")
```

Arguments

`data`	data is the data frame containing the EEG signals. The data frame must be organized as follows: each column represents a different electrode so that each electrical potential collected by each electrode is represented in each row. To identify which class and which replicate each line represents, the vectors `classes` and `reps` are given and must have length equal to the number of rows in the data frame.
`classes`	is a vector with length equal to the number of rows in `data`. Thus each value in the array identifies the class of each stimulus in each row of the database. For example, let `classes <- c (1,1,1,1,1,2,2,2,2,2)`, this means that the first 5 rows of `data` represents the first class and the lines 6 to 10 represent the second class.

reps	is a vector with length equal to the number of rows in data. Thus each value in the array identifies the replicate of each stimulus in each row of the database. For example, let reps <- c (1,1,1,1,1,2,2,2,2,2), this means that the first 5 rows of data represents the first replicate of some stimulus and the lines 6 to 10 represent the second replicate of the stimulus. reps must be numeric and numerated from 1 to the total number of replicates for each class.
all	if TRUE svmspec.cv performs the cross validation using one replicate of each class for the test phase and repeats this process for every possible combination of test signals. If FALSE the cross validation is performed specifying the parameters numTest and n.iter. See details for more information.
numTest	is a vector representing the number of replicates of each class used in the test phase. For example, if numTest=c(3,2) then 3 replicates of the first class and 2 replicates of the second class are removed from the training phase and used in the testing phase. Only used if all=FALSE. See details for more information.
n.iter	numeric, determines the number of iterations performed by the cross validation. Only used if all=FALSE. See details for more information.
print	if TRUE prints a message at the end of each iteration.
ma	is the moving average parameter for the signals.
ma.s	is the moving average parameter for the spectrum.
class.labels	a vector containing the representation of each class in the vector classes.
byfreq	if TRUE uses a different svm function for each frequency of the spectrum and then uses a weight system to defines the final decision. If FALSE uses only one svm function for all frequencies of the spectrum.
Rho	parameter used in the weight system. Only used if byfreq=TRUE.
test	defines the statistic used to extract the features. If test="ALL" then all the frequencies of the spectrum are used. If test="t.test" then the t statistic is used. If test="f.test" then the F statistic is used. If test="wilcox" then the wilcox statistic is used.
type	is the type of feature extraction. If type="p.value" then the features are selected based on the p-value obtained using the statistic selected with the parameter test. If type="max" then the features are selected based on the maximum value obtained using the statistic selected with the parameter test. If type="max.el" then the features are selected based on the maximum value obtained, for each electrode, using the statistic selected with the parameter test.
alpha	if type="p.value" then the significance level for the selected features is 1-alpha.
kernel	which kernel is used for the support vector machine classifier. See svm for more information.
cost	the cost parameter used for the support vector machine classifier. See svm for more information.

Details

If `all=FALSE` the cross validation is performed as follows: 1- some of the replicates of all classes are randomly removed from the database (based on the parameter `numTest`) and the discriminant function is then calculated using the remaining replicates. 2- the removed replicates are used to test the discriminant. 3- this process is repeated `n.iter` times and the classification rate is estimated.

If `all=TRUE` the cross validation is performed as follows: 1- one replicate of each class are removed from the database and the discriminant function is then calculated using the remaining replicates. 2- the removed replicates are used to test the discriminant. 3- this process is repeated until all possible pairs of replicates are removed from the training phase and used in the test phase. 4- the correct classification rate is estimated.

Value

`c.matrix`	Confusion matrix.
`acc`	accuracy, the correct classification rate.

Author(s)

Murilo Coutinho

References

Bostanov, V. (2004) *BCI Competition 2003 - Data Sets Ib and IIb: Feature Extraction From Event-Related Brain Potentials With the Continuous Wavelet Transform and the t-Value Scalogram.* IEEE transactions on biomedical engineering, V. 51, no. 6.

Hastie, T., Tibshirani, R., Friedman, J. (2009) *The Elements of Statistical Learning: Data Mining, Inference, and Prediction.* 2nd ed. Stanford: Springer.

Coutinho, M., Borries, G.F., Borries, R.F. (2011) *EEG data classification using Support Vector Machine and Fourier data analysis, a new approach.*

See Also

`svm`, `eeg.discr.spec`, `spectrum`

Examples

```
#Simulating a dataset with randeeg.
Sim <- randeeg(nclass=2,nrep=4,nobs=150,nel = 5, ar=
matrix(c(0.3,0.1,0,0),2,2,byrow=TRUE), ma=matrix(c(0.7,0),2,1,
byrow=TRUE), order=matrix(c(2,0,1,0,0,0),2,3,
byrow=TRUE), vars = c(1,1))

#Making a cross validation to estimate the correct classification
#rate, using the spectrum as feature extraction.
svmspec.cv(Sim$data, Sim$reps , Sim$class , all = TRUE , ma.s=10,
class.labels = c(1,2), print=TRUE,byfreq=TRUE, alpha=0.99,
test="t.test",type = "p.value")
```

`TrainEEG` *Trains a classifier.*

Description

A function which uses svmEEG and FeatureEEG internally to train a classifier receiving data with the format produced by the Open Vibe software.

Usage

```
TrainEEG(stimuli,signals,classes_label,epoch_duration,
classifier_name, epoch_offset)
```

Arguments

`stimuli`	a data set containig the classes identifiers in Open Vibe like format.
`signals`	a data set containig the signals identifiers in Open Vibe like format.
`classes_label`	a vector identifying the classes labels of stimuli.
`epoch_duration`	the duration in seconds of each signal.
`classifier_name`	a string giving the name of the file to be saved containing the classifier object.
`epoch_offset`	The 'jump' in seconds to be considered to produce the data set to be trained.

Details

This function is used to implement a brain machine interface on the open vibe.

Value

`y`	An object to be used in classifyEEG.

Author(s)

Murilo Coutinho

References

Bostanov, V. (2004) *BCI Competition 2003 - Data Sets Ib and IIb: Feature Extraction From Event-Related Brain Potentials With the Continuous Wavelet Transform and the t-Value Scalogram.* IEEE transactions on biomedical engineering, V. 51, no. 6.

Hastie, T., Tibshirani, R., Friedman, J. (2009) *The Elements of Statistical Learning: Data Mining, Inference, and Prediction.* 2nd ed. Stanford: Springer.

Coutinho, M., Borries, G.F., Borries, R.F. (2011) *EEG data classification using Support Vector Machine and Fourier data analysis, a new approach.*

Anexo 3

```
Alpha=rep(0.1,600)
Prob=0.9
AlphaCorr=0.1
minacc=rep(0.8,600)
Nfea=5

featype<-c("f1","f1","f1","f1","f1","f1","f1","f1","f1","f1",
"f1","f1","f1","f1","f1","f1","f1","f1","f1","f1","f1","f1",
"f1","f1","f1","f1","f1","f1","f1","f1","f1","f1","f3","f3",
"f3","f3","f3","f3","f3","f3","f3","f3","f3","f3","f3","f3",
"f3","f3","f3","f3","f3","f3","f3","f3","f3","f3","f3","f3",
"f3","f3","f3","f3","f3","f3","f5","f5","f5","f5","f5","f5",
"f5","f5","f5","f5","f5","f5","f5","f5","f5","f5","f5","f5",
"f5","f5","f5","f5","f5","f5","f5","f5","f5","f5","f5","f5",
"f5","f5","f2","f4","f7")

stat<-c("sum","sum","sum","sum","sum","sum","sum","sum","sum","sum",
"sum","sum","sum","sum","sum","sum","sum","max","max","max",
"max","max","max","max","max","max","max","max","max","max",
"max","max","max","sum","sum","sum","sum","sum","sum","sum",
"sum","sum","sum","sum","sum","sum","sum","sum","sum","max",
"max","max","max","max","max","max","max","max","max","max",
"max","max","max","max","max","sum","sum","sum","sum","sum",
"sum","sum","sum","sum","sum","sum","sum","sum","sum","sum",
```

```
"sum","max","max","max","max","max","max","max","max","max",
"max","max","max","max","max","max","max")

power<-c(2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,1,1,
1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,
2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2
,2,2,1,1,1,1,1,1,1,1)

mintomax<-c(TRUE,TRUE,TRUE,TRUE,TRUE,TRUE,TRUE,TRUE,FALSE,
FALSE,FALSE,FALSE,FALSE,FALSE,FALSE,FALSE,TRUE,TRUE,TRUE,TRUE,
FALSE,FALSE,FALSE,FALSE,TRUE,TRUE,TRUE,TRUE,FALSE,FALSE,FALSE,
FALSE,TRUE,TRUE,TRUE,TRUE,TRUE,TRUE,TRUE,TRUE,FALSE,FALSE,
FALSE,FALSE,FALSE,FALSE,FALSE,FALSE,TRUE,TRUE,TRUE,TRUE,FALSE,
FALSE,FALSE,FALSE,TRUE,TRUE,TRUE,TRUE,FALSE,FALSE,FALSE,FALSE,
TRUE,TRUE,TRUE,TRUE,TRUE,TRUE,TRUE,TRUE,FALSE,FALSE,FALSE,FALSE,
FALSE,FALSE,FALSE,FALSE,TRUE,TRUE,TRUE,TRUE,FALSE,FALSE,FALSE,
FALSE,TRUE,TRUE,TRUE,TRUE,FALSE,FALSE,FALSE,FALSE)

win<-c(5,10,15,20,5,10,15,20,5,10,15,20,5,10,15,20,15,30,
15,30,15,30,15,30,15,30,15,30,15,30,15,30,1,4,10,15,1,4,10,
15,1,4,10,15,1,4,10,15,1,4,10,15,10,15,10,15,10,15,10,15,10,
15,10,15,5,10,15,20,5,10,15,20,5,10,15,20,5,10,15,20,15,30,
15,30,15,30,15,30,15,30,15,30,15,30,15,30)

abs<-c(FALSE,FALSE,FALSE,FALSE,FALSE,FALSE,FALSE,FALSE,FALSE,
FALSE,FALSE,FALSE,FALSE,FALSE,FALSE,FALSE,FALSE,FALSE,FALSE,
FALSE,FALSE,FALSE,FALSE,FALSE,TRUE,TRUE,TRUE,TRUE,TRUE,TRUE,
TRUE,TRUE,FALSE,FALSE,FALSE,FALSE,FALSE,FALSE,FALSE,FALSE,
FALSE,FALSE,FALSE,FALSE,FALSE,FALSE,FALSE,FALSE,FALSE,FALSE,
FALSE,FALSE,FALSE,FALSE,FALSE,FALSE,FALSE,FALSE,FALSE,FALSE,
FALSE,FALSE,FALSE,FALSE,FALSE,FALSE,FALSE,FALSE,FALSE,FALSE,
FALSE,FALSE,FALSE,FALSE,FALSE,FALSE,FALSE,FALSE,FALSE,FALSE,
```

```
FALSE,FALSE,FALSE,FALSE,FALSE,FALSE,FALSE,FALSE,TRUE,TRUE,
TRUE,TRUE,TRUE,TRUE,TRUE,TRUE)

log<-c(FALSE,FALSE,FALSE,FALSE,TRUE,TRUE,TRUE,TRUE,FALSE,
FALSE,FALSE,FALSE,TRUE,TRUE,TRUE,TRUE,FALSE,FALSE,TRUE,TRUE,
FALSE,FALSE,TRUE,TRUE,FALSE,FALSE,TRUE,TRUE,FALSE,FALSE,TRUE,
TRUE,FALSE,FALSE,FALSE,FALSE,TRUE,TRUE,TRUE,TRUE,FALSE,FALSE,
FALSE,FALSE,TRUE,TRUE,TRUE,TRUE,FALSE,FALSE,TRUE,TRUE,FALSE,
FALSE,TRUE,TRUE,FALSE,FALSE,TRUE,TRUE,FALSE,FALSE,TRUE,TRUE,
FALSE,FALSE,FALSE,FALSE,TRUE,TRUE,TRUE,TRUE,FALSE,FALSE,
FALSE,FALSE,TRUE,TRUE,TRUE,TRUE,FALSE,FALSE,TRUE,TRUE,FALSE,
FALSE,TRUE,TRUE,FALSE,FALSE,TRUE,TRUE,FALSE,FALSE,TRUE,TRUE)
wavelet="gaussian2"

 variance=1
nselec=c(10,10)
ncomps=3
```