CLASSIFICADOR DE MODULAÇÕES BASEADO
EM MÁQUINAS DE VETORES DE SUPORTE

Felipe Aurélio Caetano de Bastos

TESE SUBMETIDA AO CORPO DOCENTE DA COORDENAÇÃO DOS PROGRAMAS DE PÓS-GRADUAÇÃO DE ENGENHARIA DA UNIVERSIDADE FEDERAL DO RIO DE JANEIRO COMO PARTE DOS REQUISITOS NECESSÁRIOS PARA A OBTENÇÃO DO GRAU DE DOUTOR EM CIÊNCIAS EM ENGENHARIA ELÉTRICA.

Aprovada por:

_____
Prof. Marcello Luiz Rodrigues de Campos, Ph.D.

_____
Dr. Bruno Cosenza de Carvalho, D.Sc.

_____
Prof. Carlos Eduardo Pedreira, Ph.D.

_____
Prof. Jacques Szczupak, Ph.D.

_____
Prof. Juraci Ferreira Galdino, D.Sc.

_____
Prof. Luiz Pereira Calôba, Dr.Ing.

RIO DE JANEIRO, RJ - BRASIL
ABRIL DE 2007

BASTOS, FELIPE AURÉLIO CAETANO DE

Classificador de Modulações Baseado em Máquinas de Vetores de suporte [Rio de Janeiro] 1989

XIII, 98 p. 29,7 cm (COPPE/UFRJ, D.Sc., Engenharia Elétrica, 2007)

Tese - Universidade Federal do Rio de Janeiro, COPPE

1. Classificação de Modulações

2. Máquinas de Vetores de Suporte

I. COPPE/UFRJ II. Título (série)

DEDICATÓRIA

Dedico este trabalho

À minha esposa Katia, Amor da minha Vida;

Aos meus filhos Rafael e Gabriel, Razão da minha Vida;

e aos meus pais Armelim e Sara, Exemplos de Vida.

AGRADECIMENTOS

Gostaria de expressar minha sincera gratidão às seguintes pessoas:

- Ao professor e amigo Marcello Campos, por sua dedicação na orientação deste trabalho e por seus valiosos conselhos e críticas;

- Aos meus pais pelo amor incondicional e apoio irrestrito em todos os momentos de minha vida;

- À minha esposa Katia, que nos momentos de minha ausência respondia com compreensão e nos momentos de dificuldade respondia com incentivo, que compartilha os meus sonhos e que é sinônimo de amor e carinho;

- Aos meus filhos que são a razão da minha vida e fonte de eterna alegria;

- Aos muitos amigos que nessa jornada estiveram próximos e que me ajudaram a seguir em frente;

- A todos os funcionários do Programa de Engenharia Elétrica da COPPE/UFRJ que de alguma forma ajudaram na realização deste trabalho.

Resumo da Tese apresentada à COPPE/UFRJ como parte dos requisitos necessários para a obtenção do grau de Doutor em Ciências (D.Sc.)

CLASSIFICADOR DE MODULAÇÕES BASEADO EM MÁQUINAS DE VETORES DE SUPORTE

Felipe Aurélio Caetano de Bastos

Abril/2007

Orientador: Marcello Luiz Rodrigues de Campos

Programa: Engenharia Elétrica

Este trabalho inicialmente apresenta um novo algoritmo para treinamento de uma Máquina de Vetores de Suporte (SVM) utilizada em Reconhecimento de Padrões. Uma análise teórica do classificador proposto, denominado UPSVM, foi realizada, obtendo-se sua complexidade computacional e uma estimativa para a sua probabilidade de acerto após a fase de treinamento. O UPSVM foi comparado com outros classificadores SVM, e mostrou-se mais rápido durante o treinamento e mais robusto na generalização do que os demais métodos analisados. Em seguida, a referida técnica foi aplicada na classificação de modulações digitais, dando origem ao classificador CMSVM, proposto neste trabalho. Simulações computacionais foram realizadas com o objetivo de determinar o seu desempenho em relação à escolha do parâmetro de regularização, definido pelo usuário, e em relação à relação sinal-ruído (RSR). Os resultados obtidos mostram que o mesmo pode ser empregado com êxito na classificação de sinais com RSR acima de 5 dB. Além disso, o classificador CMSVM possui baixa complexidade computacional, permite fácil adição ou exclusão de modulações, e obteve melhor desempenho do que o classificador CMADB, encontrado na literatura e que originalmente propôs o espaço de características também utilizado pelo classificador CMSVM.

Abstract of Thesis presented to COPPE/UFRJ as a partial fulfillment of the requirements for the degree of Doctor of Science (D.Sc.)

MODULATION CLASSIFIER BASED ON SUPPORT VECTOR MACHINES

Felipe Aurélio Caetano de Bastos

April/2007

Advisor: Marcello Luiz Rodrigues de Campos

Department: Electrical Engineering

This work initially presents a new training algorithm for Support Vector Machines (SVM) used in pattern recognition applications. The computational complexity and a estimative for the probability of correct classification was obtained for the proposed classifier, named UPSVM. It was compared to other SVM classifiers, and it proved to be the fastest for training and the most robust in the classification of unknown data. In addition, this method was applied to the classification of digital modulations obtaining a new classifier, named CMSVM. Computational simulations were done to determine how the CMSVM classifier would perform due to the choice of the regularization parameter, defined by the user, and due to the Signal-to-Noise ratio (SNR). The obtained results show that the proposed CMSVM classifier can be used to classify communication signals with SNR greater or equal to 5 dB. Besides, the CMSVM classifier has low computational complexity, allows easy addition or exclusion of modulations, and achieved better performance than the CMADB classifier, found in the literature, that originally proposed the feature space also used in CMSVM classifier.

ÍNDICE

ÍNDICE DE FIGURAS

ÍNDICE DE TABELAS

LISTA DE ABREVIAÇÕES E SÍMBOLOS

ASK2	Modulação por chaveamento em amplitude em dois níveis
ASK4	Modulação por chaveamento em amplitude em quatro níveis
ASVM	Máquina de vetores de suporte do tipo proximal e ativo
AWGN	*Additive White Gaussian Noise* – Ruído branco, gaussiano e aditivo
BFD	*Backward – Foward Diference* – Algoritmo para obtenção da freqüência instantânea a partir da fase
CMADB	Classificador de Modulação baseado em Árvore de Decisão Binária
DAGSVM	*Directed Acyclic Graph – Support Vector Machine* – Algoritmo de generalização SVM
DSP	Processamento Digital de Sinais ou Processador de Sinais Digitais
GE	Guerra Eletrônica
FLOP	Operação em Ponto flutuante
FPGA	*Field Programable Gate Array* – conjunto de blocos lógicos programáveis em um chip
FSK2	Modulação por chaveamento em freqüência em dois níveis
FSK4	Modulação por chaveamento em freqüência em quatro níveis
HSVM	Máquina de Vetor de Support do tipo *Hard*
ISI	*Inter Symbol interference* - Interferência Intersimbólica
L1SVM	Máquina de Vetor de Suporte do tipo *Soft* – L1
L2SVM	Máquina de Vetor de Suporte do tipo *Soft* – L2
ML	Máxima Verossimilhança
MLP	Rede de Múltiplos Perceptrons
PDF	Probability Density Function – Função Densidade de Probabilidade
POLY	*Kernel* polinomial
PSK2	Modulação por chaveamento em freqüência em dois níveis
PSK4	Modulação por chaveamento em freqüência em quatro níveis
PSVM	Máquina de Vetor de Suporte do tipo Proximal
QP	Programação Quadrática
RBF	Função de Base Radial
RDS	Rádio Definido por Software
	Regressão Linear Múltipla

RLM	Reconhecimento de Padrões
RP	Relação Sinal-Ruído
RSR	*Sequential Minimal Optimization* – Algoritmo para treinamento de uma
SMO	máquina de vetores de suporte do tipo L1SVM
	Vetores de Suporte
SV	Máquina de Vetores de Suporte
SVM	Teorema Central do Limite
TCL	Máquina de Vetores de Suporte do tipo Proximal e não Polarizada
UPSVM	

1. INTRODUÇÃO

1.1 Motivação, Objetivo e Contribuições do Trabalho

O reconhecimento automático do tipo de modulação que foi empregada para a criação de um sinal de comunicações é um tópico de grande importância para as atividades de controle e vigilância do espectro eletromagnético em aplicações civis ou militares. Em particular, sistemas de Guerra Eletrônica (GE) necessitam conhecer o tipo de modulação, entre outras informações, para configurar corretamente o(s) receptor(es) utilizado(s) para o registro de uma determinada comunicação sob monitoramento. É importante ressaltar que os fabricantes de sistemas de GE tratam o tema como segredo industrial e comercializam o classificador de modulações como uma caixa-preta de alto valor agregado.

Outra aplicação de grande importância para um classificador de modulação é a implementação de rádios cognitivos. Neste contexto, o processo de classificação prove a inteligência necessária para que um rádio definido por *software* (RDS) possa se configurar ou reprogramar automaticamente.

No que se refere à sua implementação, o classificador de modulação pode utilizar diversas técnicas, incluindo técnicas de Reconhecimento de Padrões (RP). Dentro do contexto de RP, Máquinas baseadas em Vetores de Suporte, ou *Support Vector Machines* (SVM), vem sendo empregadas com êxito em diversas aplicações. Um ponto favorável à união da técnica SVM à classificação de modulação é o fato de que o classificador SVM obtido após uma fase de treinamento é rápido e pode ser implementado por um processador de sinais digitais (DSP), ou por um conjunto de blocos lógicos e programáveis (FPGA) para aplicações em tempo real.

Além disso, sua utilização na classificação de sinais no que se refere à modulação é promissora e inovadora. Os resultados oriundos do estudo, análise e utilização desta técnica podem resultar em várias contribuições importantes que podem ser aplicadas diretamente em diversas áreas, não sendo, portanto, específicas para o referido classificador.

Motivado pelo descrito acima, estabeleceu-se como objetivo do presente trabalho a criação de um classificador para sinais de comunicação baseado em Máquinas de Vetores de Suporte. A referida classificação de sinais refere-se à modulação empregada e deve funcionar para diversos valores de relação sinal-ruído.

Buscando alcançar este objetivo, várias contribuições foram obtidas e são citadas sucintamente a seguir:

- a análise sobre a Máquina de Vetores de Suporte do tipo Proximal com relação à polarização do hiperplano ótimo, à generalização não ortodoxa para o caso de classes não separáveis linearmente e à sua comparação com o estimador de Ridge (cap 2, seções 2.6 a 2.7);
- a proposta e a análise de um novo algoritmo para treinamento SVM baseado em Máquina de Vetores de Suporte do tipo Proximal não polarizado (UPSVM) (cap 3);
- a comparação entre os diversos classificadores SVM e a análise sobre o desempenho de cada classificador com relação ao parâmetro de regularização (cap 4);
- a análise de um classificador de modulação encontrado na literatura baseado em uma árvore de decisão binária e de pequena complexidade computacional (cap 5);
- o uso de um estimador de máxima verossimilhança para o cálculo da freqüência instantânea a ser utilizada no classificador de modulação, com o intuito de melhorar o desempenho na classificação de modulações em freqüência (caps 5 e 6); e
- a aplicação da técnica SVM à classificação de modulação e a comparação entre o classificador de modulação proposto neste trabalho e o classificador de modulação anteriormente analisado (cap 6).

As várias contribuições foram distribuídas ao longo do texto da tese e são abordadas com mais detalhes nas seções e capítulos supracitados.

1.2 Descrição do Trabalho

O presente trabalho está dividido em sete capítulos como se segue. O segundo capítulo, ainda introdutório, aborda a teoria por trás da técnica SVM utilizada em reconhecimento de padrões para separação de duas classes. Neste capítulo, são apresentados três diferentes tipos de treinamento para a obtenção de classificadores SVM tradicionais (seções 2.2 e 2.3) e uma arquitetura para implementação de um classificador SVM não linear (seção 2.4). São apresentados também métodos existentes para a criação de classificadores para K classes a partir dos classificadores SVM para 2 classes (seção 2.5). Na seção 2.6, uma alternativa aos métodos SVM tradicionais, o classificador PSVM apresentado por Mangasarian e Fung em [1], é abordada. A seção 2.7 discute a analogia entre o Regressor de Ridge (solução para problemas de Regressão Linear Múltipla) e o classificador PSVM, concluindo que esta analogia não justifica a introdução de polarização na determinação do hiperplano ótimo que define o classificador PSVM.

O terceiro capítulo apresenta um novo algoritmo de treinamento SVM denominado UPSVM, que é uma das contribuições deste trabalho. As versões linear e não linear desse classificador são descritas com detalhes nas seções 3.2 e 3.3, enquanto que uma análise teórica do UPSVM linear e a complexidade computacional das duas versões foram apresentadas na seção 3.4. Em particular, é importante destacar que a referida análise teórica estabelece uma estimativa para a probabilidade de acerto do classificador UPSVM linear após o treinamento e que a sua complexidade computacional no treinamento é O(N).

O capítulo 4 apresenta uma comparação entre três algoritmos rápidos para o treinamento SVM, dois deles já citados anteriormente (PSVM e UPSVM) e um terceiro denominado SMO, abreviação do seu nome em inglês *Sequential Minimal Optimization*, utilizado para treinamento de uma SVM tradicional. A comparação foi realizada em função da probabilidade de acerto, tempo e complexidade computacional da fase de treinamento, variância dos hiperplanos ótimos estimados e número de vetores de suporte (SV). Os resultados obtidos a partir dessa comparação mostram que o classificador UPSVM proposto nesse trabalho se mostrou o mais rápido na fase de treinamento em todas as aplicações estudadas, tanto na versão linear como na versão

não linear, obtendo taxas de acerto similares ou superiores aos demais métodos. Além disso, mostrou ser o único método robusto quanto à escolha do parâmetro de regularização da técnica SVM, parâmetro que deve ser escolhido pelo usuário.

O quinto capítulo apresenta uma visão geral sobre os diversos tipos de classificadores de sinais de comunicação no que se refere à modulação empregada (seção 5.1). Dos diversos métodos existentes na literatura, destaca-se o método proposto por Assouz e Nandi em [2] (seção 5.3), devido a sua facilidade de implementação e rapidez de processamento, podendo ser empregado em tempo real. A análise deste método também é uma contribuição do trabalho de tese, sendo apresentada na seção 5.4.

O sexto capítulo descreve a aplicação do método UPSVM na classificação de modulações digitais (ASK2, ASK4, FSK2, FSK4, PSK2 e PSK4) que resultou na criação do classificador denominado CMSVM, outra contribuição deste trabalho. Neste capítulo foram analisadas as influências do parâmetro de regularização SVM e da relação sinal-ruído (RSR) no desempenho do classificador de modulação CMSVM.

O sétimo capítulo contém as conclusões deste trabalho e as propostas para trabalhos futuros. O apêndice apresenta rotinas em Matlab para a implementação do algoritmo SMO.

2. MÁQUINAS BASEADAS EM VETORES DE SUPORTE

2.1 Introdução

Uma máquina de vetores de suporte, doravante denominada SVM (a partir do termo em inglês), pode ser utilizada para o reconhecimento de padrões M-dimensionais entre duas classes distintas. O classificador SVM utiliza um hiperplano, obtido a partir dos chamados vetores de suporte, para separar o universo M-dimensional em duas regiões, cada uma associada a uma classe. O hiperplano é definido pela equação:

$$\mathbf{x}^T\boldsymbol{\omega} + b = 0 \qquad \text{Eq. 2.1}$$

Os parâmetros $\boldsymbol{\omega} \in \Re^M$ e $b \in \Re$ definem o hiperplano e são denominados vetor de pesos e polarização, respectivamente. Uma observação $\mathbf{x_i}$ é dita pertencer à classe 1 se $\mathbf{x_i}^T\boldsymbol{\omega}+b \geq 0$, caso contrário pertence à classe 2.

Para implementar o classificador SVM é necessário determinar os parâmetros $\boldsymbol{\omega}$ e b ótimos, o que é feito durante uma fase de treinamento supervisionado onde se procura maximizar a margem de separação entre as duas classes. A conceituação do termo margem de separação e a definição do classificador do tipo *hard* [3] (HSVM), utilizado para o caso de classes linearmente separáveis, serão abordadas na seção 2.2. Em seguida, na seção 2.3, serão abordados os classificadores tipo *soft* [3,4] (L1SVM e L2SVM), utilizados para o caso de classes que não são linearmente separáveis. Neste último caso, a determinação do classificador SVM leva em conta também a minimização de uma função de risco associada ao erro de treinamento.

2.2 Hiperplano Ótimo para a Separação de Classes Linearmente Separáveis

Suponha a existência de duas classes linearmente separáveis, C1 e C2. Seja $\mathbf{X} \in \Re^{MXN}$ uma matriz composta por N observações dispostas na forma de vetores-coluna com M elementos cada, e cujas classificações são conhecidas. Seja $\mathbf{x_i}$,$1<i<N$, a i-ésima coluna de $\mathbf{X}$. A distância, r_i, do ponto $\mathbf{x_i}$ ao hiperplano ótimo é dada pela eq. 2.2, onde $\boldsymbol{\omega_0}$ e b_0 são, respectivamente, os valores ótimos do vetor de pesos e da polarização.

$$r_i = \frac{\mathbf{x_i}^T \boldsymbol{\omega_0} + b_0}{\|\boldsymbol{\omega_0}\|} \qquad \text{Eq. 2.2}$$

Os vetores de suporte são definidos como sendo as observações mais próximas da superfície de decisão e, portanto, correspondem às observações que definiram a superfície separadora. Seja $\mathbf{x_S}$ um vetor de suporte, sem perda de generalidade pode-se afirmar que:

$$\mathbf{x_S}^T\boldsymbol{\omega_0} + b_0 = 1, \text{ se } \mathbf{x_S} \in C1 \qquad \text{Eq. 2.3}$$

$$\mathbf{x_S}^T\boldsymbol{\omega_0} + b_0 = -1, \text{ se } \mathbf{x_S} \in C2. \qquad \text{Eq. 2.4}$$

Esta normalização é possível, já que a multiplicação de $\boldsymbol{\omega_0}$ e b_0 por uma constante não altera o hiperplano ótimo dado pela eq. 2.1, assim como não altera a distância de $\mathbf{x_S}$ ao hiperplano, calculada pela eq. 2.2. A partir da definição de vetores de suporte e das eqs. 2.3 e 2.4, chega-se a conclusão que:

$$\mathbf{x_i}^T\boldsymbol{\omega_0} + b_0 \geq 1, \text{ se } \mathbf{x_i} \in C1 \qquad \text{Eq. 2.5}$$

$$\mathbf{x_i}^T\boldsymbol{\omega_0} + b_0 \leq -1, \text{ se } \mathbf{x_i} \in C2. \qquad \text{Eq. 2.6}$$

A figura 2.1 ilustra o uso de um hiperplano ótimo para separar duas classes linearmente separáveis em um espaço bidimensional. Nela também estão mostrados os vetores de suporte para cada classe e os hiperplanos que os contêm, de agora em diante chamados hiperplanos de suporte. É importante destacar que $\boldsymbol{\omega_0}$ é um vetor ortogonal ao hiperplano ótimo, e que b_0 define a distância do hiperplano ótimo à origem. Neste tipo de classificador, conhecido como do tipo *hard* ou HSVM, todos os vetores de suporte estarão contidos nos hiperplanos de suporte.

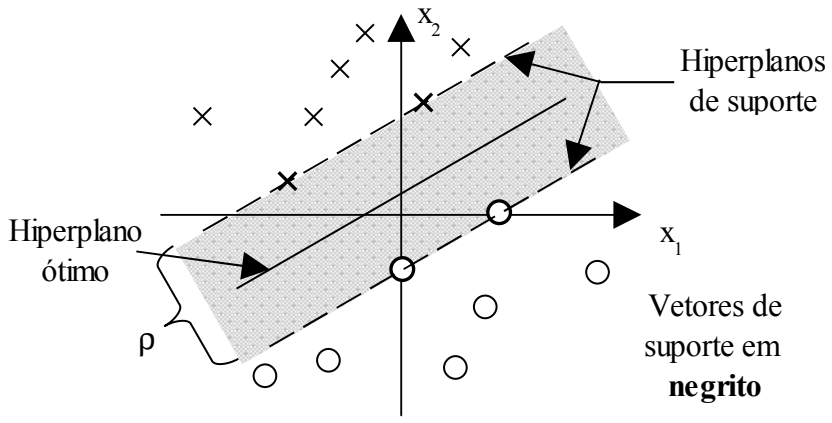

Figura 2.1: Exemplo do uso de um hiperplano para separação de classes (x ∈ Classe1 e o ∈ Classe2).

A margem de separação, definida como sendo a distância entre os dois hiperplanos que contêm os vetores de suporte, é dada por:

$$\rho = 2|r_S| = \frac{2}{\|\boldsymbol{\omega}_0\|} \qquad \text{Eq. 2.7}$$

Para tornar o classificador SVM mais robusto, os parâmetros $\boldsymbol{\omega}_0$ e b_0 devem ser escolhidos de modo a obter a maior margem de separação possível entre os hiperplanos de suporte de cada classe. Sendo assim, levando em consideração que $\|\boldsymbol{\omega}\|^2 = \boldsymbol{\omega}^T\boldsymbol{\omega}$, o problema pode ser descrito matematicamente como a minimização de uma função-objetivo convexa e quadrática sujeita a restrições lineares, ou seja:

$$J = \min_{\boldsymbol{\omega}}\left(\frac{1}{2}\boldsymbol{\omega}^T\boldsymbol{\omega}\right), \text{ sujeita à restrição } \mathbf{D}\left(\mathbf{X}^T\boldsymbol{\omega} + b\mathbf{e}\right) \geq \mathbf{e} \qquad \text{Eq. 2.8}$$

onde $\mathbf{e}$ é um vetor de dimensão apropriada e cujos elementos são iguais a 1; $\mathbf{D}$ é uma matriz diagonal cujo i-ésimo elemento não nulo (d_i) é igual a 1, se $\mathbf{x_i} \in$ C1, ou é igual a -1, se $\mathbf{x_i} \in$ C2; e $\mathbf{X}$ é a matriz de observações que tem $\mathbf{x_i}$ como sua i-ésima coluna.

A função-objetivo que deve ser minimizada na eq. 2.8 é diretamente proporcional ao quadrado do módulo do vetor de pesos, já que se deseja maximizar a margem de separação entre as duas classes. A escolha de $\boldsymbol{\omega}_0$ e b_0 deve também obedecer às restrições impostas pelas eqs. 2.5 e 2.6, que foram concatenadas para formar a restrição apresentada na eq. 2.8.

A solução para o problema apresentado pode ser obtida com a utilização do método de multiplicadores de Lagrange. Nesta técnica, um novo termo composto pelo somatório de todas as restrições é subtraído da função-objetivo, obtendo-se uma nova função denominada função Lagrangeana. Nesse somatório, cada restrição é ponderada por um número não negativo denominado multiplicador de Lagrange. A solução para o problema de minimização com restrições é obtida no ponto de sela da função Lagrangeana [3].

Portanto, a função Lagrangeana definida a partir da função-objetivo apresentada na eq. 2.8 é dada a seguir

$$J(\boldsymbol{\omega}, b, \boldsymbol{\alpha}) = \frac{1}{2}\boldsymbol{\omega}^T\boldsymbol{\omega} - \boldsymbol{\alpha}^T\left[\mathbf{D}\left(\mathbf{X}^T\boldsymbol{\omega} + b\mathbf{e}\right) - \mathbf{e}\right] \qquad \text{Eq. 2.9}$$

onde $\boldsymbol{\alpha}$ é o vetor composto pelos multiplicadores de Lagrange que nunca são negativos.

O ponto de sela da função Lagrangeana é obtido minimizando-a com relação aos parâmetros $\boldsymbol{\omega}$ e b e maximizando-a com relação ao parâmetro $\boldsymbol{\alpha}$. Calculando as derivadas parciais da função Lagrangeana em função dos parâmetros de interesse, $\boldsymbol{\omega}$ e b, e igualando-as a zero chega-se, respectivamente, a :

$$\boldsymbol{\omega} = \mathbf{XD}\boldsymbol{\alpha} \qquad \text{Eq. 2.10}$$

$$\boldsymbol{\alpha}^{\mathrm{T}}\mathbf{De} = 0 \qquad \text{Eq. 2.11}$$

Substituindo a eq 2.10 na eq. 2.9, usando a eq. 2.11 convenientemente, e multiplicando o resultado final por -1, obtém-se uma nova função-objetivo que deve ser minimizada em função de $\boldsymbol{\alpha}$:

$$J(\boldsymbol{\alpha}) = \frac{1}{2}\boldsymbol{\alpha}^{\mathrm{T}}\mathbf{DX}^{\mathrm{T}}\mathbf{XD}\boldsymbol{\alpha} - \boldsymbol{\alpha}^{\mathrm{T}}\mathbf{e} \qquad \text{Eq. 2.12}$$

É importante destacar que existe uma solução analítica caso a matriz inversa de $\mathbf{X}^{\mathrm{T}}\mathbf{X}$ exista, o que só pode ocorrer se M≥N, onde M corresponde ao número de parâmetros de uma observação e ao número de linhas da matriz de observações $\mathbf{X}$, e N corresponde ao número de observações e ao número de colunas contidas na matriz $\mathbf{X}$. No entanto este caso não é útil na prática, uma vez que o número de observações é geralmente muito maior do que o número de parâmetros de cada observação.

Conclui-se que o classificador SVM do tipo HSVM é obtido por meio de uma fase de treinamento que consiste na solução de um problema dual, definido como sendo:

" Minimizar a função-objetivo dada por

$$J(\boldsymbol{\alpha}) = \frac{1}{2}\boldsymbol{\alpha}^{\mathrm{T}}\mathbf{DX}^{\mathrm{T}}\mathbf{XD}\boldsymbol{\alpha} - \boldsymbol{\alpha}^{\mathrm{T}}\mathbf{e}$$

em função do vetor $\boldsymbol{\alpha}$ e sujeita às seguintes restrições:

- $\boldsymbol{\alpha}^{\mathrm{T}}\mathbf{De} = 0$
- $\alpha_i \geq 0$, para $i = 1,2, \ldots , N$

onde $\mathbf{X}$ e $\mathbf{D}$ são parâmetros para o treinamento. "

onde a primeira restrição é uma conseqüência da minimização da função Lagrangeana em função de b, enquanto que a segunda restrição se faz necessária devido ao tipo de restrição imposta pelo problema analisado, ou seja, N inequações positivas. Após a solução do problema dual, a solução do problema principal, definido pela eq. 2.8, é obtida através das equações:

$$\boldsymbol{\omega_0} = \mathbf{XD}\boldsymbol{\alpha_0} \qquad \text{Eq. 2.13}$$

$$b_0 = d_S - \boldsymbol{\omega_0}^{\mathrm{T}}\mathbf{x_S} \qquad \text{Eq. 2.14}$$

onde $\boldsymbol{\alpha_0}$ é o vetor ótimo de multiplicadores de Lagrange, $\mathbf{x_S}$ é um vetor de suporte e d_S é igual a 1 se $\mathbf{x_S} \in$ C1, ou igual a –1, caso contrário.

É importante destacar também que, no ponto de sela [3], tem-se:

$$\alpha_i\left[d_i\left(\boldsymbol{\omega}^{\mathrm{T}}\mathbf{x_i} + b\right) - 1\right] = 0 \qquad \text{Eq. 2.15}$$

Sendo assim, os vetores de suporte podem ser obtidos a partir dos multiplicadores de Lagrange, ou seja, $\mathbf{x_i}$ será um vetor de suporte se $\alpha_i \neq 0$. Deste modo, o parâmetro b_0 pode ser obtido pela eq. 2.14 depois da obtenção de pelo menos um vetor de suporte pela análise da eq. 2.15, também conhecida como condição de Karush – Kuhn – Tucker para otimalidade [3]. Do exposto acima e pela eq. 2.13, pode-se afirmar que apenas os vetores de suporte são usados para a determinação do vetor de pesos.

2.3 Classificação de Classes Não Separáveis Linearmente

2.3.1 Introdução

O método apresentado na seção anterior é aplicável apenas se as classes forem separáveis linearmente. Caso não sejam, a utilização de um hiperplano como superfície de separação entre as classes implica uma probabilidade de erro na classificação diferente de zero.

Seja $\xi_i \geq 0$ uma medida do quanto a observação x_i se afastou da condição de separação ideal, ou seja: $\xi_i = 0$, se $d_i(\mathbf{x_i}^{\mathrm{T}}\boldsymbol{\omega}_0 + b_0) \geq 1$, ou $\xi_i = 1 - d_i(\mathbf{x_i}^{\mathrm{T}}\boldsymbol{\omega_0} + b_0)$, caso contrário. Deseja-se maximizar a margem de separação, dada pela eq. 2.7, e conseqüentemente minimizar o módulo de $\boldsymbol{\omega}$. Entretanto, uma maior separação entre as classes pode implicar em um número maior de observações com $\xi_i \neq 0$ (vide fig. 2.2), Tal fato pode não ser desejável, pois, embora ξ_i não deva ser literalmente interpretado como o erro de treinamento associado a i-ésima observação, ξ_i está intimamente relacionado a ele, uma vez que $\xi_i > 1$ implica na classificação incorreta da i-ésima observação, e conseqüentemente em erro de treinamento.

Na tentativa de introduzir o erro de treinamento de forma matematicamente tratável na formulação do problema, Vapnik [5] propõe a modificação da eq. 2.8 de acordo com o conceito de minimização do risco estrutural, obtendo-se:

$$J(\boldsymbol{\omega},\boldsymbol{\xi}) = \min_{\boldsymbol{\omega},\boldsymbol{\xi}}\left(\frac{1}{2}\boldsymbol{\omega}^T\boldsymbol{\omega} + \frac{C}{p}\sum_{i=1}^{N}(\xi_i)^p\right), \text{ sujeita à restrição } \mathbf{D}\left(\mathbf{X}^T\boldsymbol{\omega} + b\mathbf{e}\right) \geq \mathbf{e} - \boldsymbol{\xi} \quad \text{Eq. 2.16}$$

onde o parâmetro de regularização, $C \geq 0$, é uma constante de livre escolha do usuário que define o quanto se está disposto a aceitar que a margem de separação seja violada, ou indiretamente, o quanto de ênfase se deseja dar ao erro de treinamento.

Por sua vez, a variável p na eq. 2.16 define o tipo de classificador SVM, ou seja, p=1 dá origem ao classificador L1SVM [3,4] e p=2, ao L2SVM [4]. Em ambos os casos, os hiperplanos de suporte não confinam mais as observações de cada classe. Por isso os referidos classificadores são considerados do tipo *soft*, ao contrário daquele que foi visto na seção anterior (*hard*).

A figura 2.2 ilustra o uso de um hiperplano ótimo para separar duas classes não separáveis linearmente. Nela também estão mostrados os vetores de suporte para cada classe e os hiperplanos que os contêm. A figura 2.2 também ilustra os vetores que não estão confinados pelos hiperplanos de suporte. Algumas destas observações não acarretam erro de classificação ($0 < \xi_i < 1$), pois ainda estão no lado correto da superfície de decisão, enquanto outras acarretam erro de classificação ($\xi_i \geq 1$).

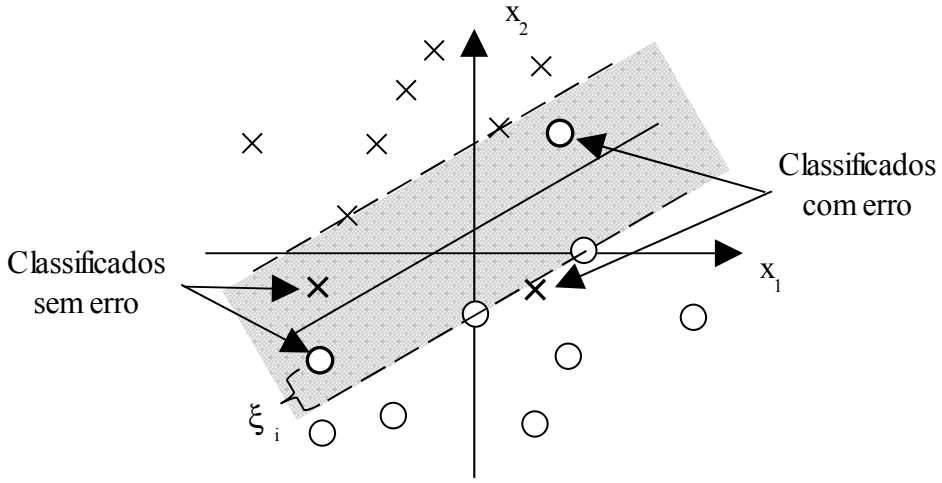

Figura 2.2: Exemplo do uso de um hiperplano para separação de classes não linearmente separáveis (x∈C1 e o∈C2).

2.3.2 O Classificador L1SVM

Analogamente ao que foi visto na seção anterior, a solução do problema principal, definido pela eq. 2.16, é obtida utilizando o método de Lagrange. Para o classificador **L1SVM**, a nova função Lagrangeana a ser minimizada é:

$$J(\boldsymbol{\omega}, b, \boldsymbol{\xi}, \boldsymbol{\alpha}, \boldsymbol{\mu}) = \frac{1}{2}\boldsymbol{\omega}^{\mathrm{T}}\boldsymbol{\omega} + C\boldsymbol{\xi}^{\mathrm{T}}\mathbf{e} - \boldsymbol{\alpha}^{T}\left[\mathbf{D}\left(\mathbf{X}^{\mathrm{T}}\boldsymbol{\omega} + b\mathbf{e}\right) - \mathbf{e} + \boldsymbol{\xi}\right] - \boldsymbol{\mu}^{\mathrm{T}}\boldsymbol{\xi} \qquad \text{Eq. 2.17}$$

onde $\boldsymbol{\mu}$ é o vetor de multiplicadores de Lagrange cujo elemento não negativo μ_i está associado a restrição $\xi_i \geq 0$, para i variando de 1 a N. Procedendo de maneira similar ao que foi feito para a determinação do problema dual para o classificador HSVM, obtém-se o problema dual, dado por:

" Minimizar a função-objetivo dada por

$$J(\boldsymbol{\alpha}) = \frac{1}{2}\boldsymbol{\alpha}^{\mathrm{T}}\mathbf{D}\mathbf{X}^{\mathrm{T}}\mathbf{X}\mathbf{D}\boldsymbol{\alpha} - \boldsymbol{\alpha}^{\mathrm{T}}\mathbf{e}$$

em função do vetor $\boldsymbol{\alpha}$ e sujeita às seguintes restrições:

- $\boldsymbol{\alpha}^{\mathrm{T}}\mathbf{D}\mathbf{e} = 0$
- $0 \leq \alpha_i \leq \mathrm{C}$, para $i = 1,2, \ldots, \mathrm{N}$

onde **X** e **D** são parâmetros para o treinamento. "

A única diferença entre os classificadores L1SVM e HSVM é que os multiplicadores de Lagrange têm um limite superior dado pelo parâmetro C. Este parâmetro, usualmente escolhido de forma empírica, define o quanto de ênfase é dada ao erro de classificação na fase de treinamento em relação à margem de separação.

Além disto, no ponto de sela da função Lagrangeana dada pela eq.2.17, têm-se:

$$\alpha_i\left[d_i\left(\boldsymbol{\omega}^{\mathrm{T}}\mathbf{x}_i + b\right) - 1 + \xi_i\right] = 0 \qquad \text{Eq. 2.18}$$

$$\mu_i\xi_i = 0 \qquad \text{Eq. 2.19}$$

Por sua vez, a derivada parcial da função Lagrangeana em função da variável ξ_i resulta na condição $\mu_i + \alpha_i = \mathrm{C}$, ou seja:

- Se $\alpha_i = 0$, então $\mu_i = \mathrm{C}$ e $\xi_i = 0$, o que implica que $\mathbf{x}_i$ é um vetor confinado pelo hiperplano de suporte e não é um vetor de suporte.
- Se $0 < \alpha_i < \mathrm{C}$, então $\mu_i \neq 0$ e $\xi_i = 0$, o que implica que $\mathbf{x}_i$ é um vetor de suporte contido em um hiperplano de suporte.

- Se α_i = C, então então μ_i = 0 e $\xi_i \neq 0$, o que implica que $\mathbf{x_i}$ é um vetor de suporte e não está confinado pelo hiperplano de suporte.

2.3.3 O Classificador L2SVM

Para o classificador **L2SVM**, a nova função Lagrangeana a ser minimizada é:

$$J(\boldsymbol{\omega}, b, \boldsymbol{\xi}, \boldsymbol{\alpha}) = \frac{1}{2}\boldsymbol{\omega}^T\boldsymbol{\omega} + C\frac{\boldsymbol{\xi}^T\boldsymbol{\xi}}{2} - \boldsymbol{\alpha}^T\left[\mathbf{D}\left(\mathbf{X}^T\boldsymbol{\omega} + b\mathbf{e}\right) - \mathbf{e} + \boldsymbol{\xi}\right] \qquad \text{Eq. 2.20}$$

Procedendo de maneira similar ao que foi feito para a determinação do problema dual dos classificadores HSVM e L1SVM, obtém-se o problema dual, dado por:

" Minimizar a função-objetivo dada por

$$J(\boldsymbol{\alpha}) = \frac{1}{2}\boldsymbol{\alpha}^T\mathbf{D}\mathbf{X}^T\mathbf{X}\mathbf{D}\boldsymbol{\alpha} + \frac{\boldsymbol{\alpha}^T\boldsymbol{\alpha}}{2C} - \boldsymbol{\alpha}^T\mathbf{e}$$

em função do vetor $\boldsymbol{\alpha}$ e sujeita às seguintes restrições:

- $\boldsymbol{\alpha}^T\mathbf{D}\mathbf{e} = 0$
- $\alpha_i \geq 0$, para i = 1,2, … , N

onde **X** e **D** são parâmetros para o treinamento. "

Neste caso, a única diferença entre os classificadores L2SVM e HSVM está na função-objetivo a ser minimizada (acréscimo do termo $\boldsymbol{\alpha}^T\boldsymbol{\alpha}/(2C)$). A determinação dos vetores de suporte é feita da mesma maneira que foi feita para o classificador HSVM, ou seja, $\mathbf{x_i}$ será um vetor de suporte se $\alpha_i \neq 0$, o que sempre implica em $\xi_i \neq 0$, pois igualar a zero a derivada parcial da função Lagrangeana, dada pela eq. 2.20, em função de ξ_i, resulta em

$$\boldsymbol{\xi} = \frac{\boldsymbol{\alpha}}{C} \qquad \text{Eq. 2.21}$$

Logo, os vetores de suporte correspondem aos vetores que não foram confinados pelos hiperplanos de suporte e não pertencem aos mesmos, enquanto que se $\alpha_i = 0$ então $\xi_i = 0$, o que equivale a dizer que a observação $\mathbf{x_i}$ é confinada por um hiperplanos de suporte, ou está contida no mesmo.

Após a solução do problema dual, a solução do problema principal, definido pela eq. 2.16, tanto para o classificador L1SVM como para o classificador L2SVM, é

obtida através da eq. 2.13, repetida em seguida por conveniência, e da eq.2.14 modificada, ou seja,

$$\boldsymbol{\omega}_0 = \mathbf{X}\mathbf{D}\boldsymbol{\alpha}_0$$

$$b_0 = d_S(1 - \xi_S) - \boldsymbol{\omega}_0^T \mathbf{x}_S$$

onde ξ_S é a medida do quanto o vetor de suporte escolhido ($\mathbf{x}_S$) se afastou da condição de separação ideal e d_S é igual a 1 se $\mathbf{x}_S \in$ C1, ou igual a –1, caso contrário. Para o classificador L1SVM, ξ_S é igual a zero, enquanto que para o classificador L2SVM, ξ_S é dada pela eq. 2.21, para α_S diferente de zero.

2.4 Implementação de Classificadores SVM não Lineares

O teorema de Cover [3] afirma que é mais fácil obter padrões linearmente separáveis em um espaço de alta dimensão, obtido a partir de um espaço de menor dimensão por meio de transformações não lineares. Sendo assim, o desempenho dos classificadores SVM pode, em teoria, ser melhorado se o espaço de entrada, $\Re^M$, for mapeado em um espaço de maior dimensão, usualmente chamado de espaço de características, por meio de um conjunto de transformações não lineares.

Seja o vetor $\boldsymbol{\varphi}(\mathbf{x}) = [\ \varphi_1(\mathbf{x}),\ \varphi_2(\mathbf{x}),\ \ldots,\ \varphi_Q(\mathbf{x})\ ]^T$, onde $\varphi_i(\mathbf{x})$, para $1 \le i \le Q$ (Q>M), é uma transformação não linear supostamente conhecida do vetor $\mathbf{x} \in \Re^{\mathbf{M}}$ em $\Re$. Suponha que um classificador SVM foi desenvolvido para reconhecer padrões no espaço de características, então o hiperplano ótimo é dado por:

$$\boldsymbol{\varphi}(\mathbf{x})^T\boldsymbol{\omega}_0 + b_0 = 0 \qquad \text{Eq. 2.22}$$

Por sua vez, uma observação $\mathbf{x_i}$, pertencente ao espaço de entrada, é dita pertencer à classe 1 se sua imagem no espaço de características estiver "acima" do hiperplano ótimo, ou seja, $\boldsymbol{\varphi}(\mathbf{x_i})^T\boldsymbol{\omega}_0 + b_0 \ge 0$. Caso contrário, a observação $\mathbf{x_i}$ pertence à classe 2.

Independentemente da escolha do tipo de classificador SVM, a determinação de $\boldsymbol{\omega}_0$ e b_0 é feita a partir da solução do problema dual correspondente, substituindo $\mathbf{X}^T\mathbf{X}$ pela matriz de *kernels* $\mathbf{K} \in \Re^{NxN}$, onde o *kernel* $K_{ij} = \boldsymbol{\varphi}(\mathbf{x_i})^T\boldsymbol{\varphi}(\mathbf{x_j}) = K_{ji}$. Após a obtenção de $\boldsymbol{\alpha}_0$, os parâmetros ótimos de interesse são calculados por:

$$\boldsymbol{\omega_0} = \sum_{i=1}^{N} \alpha_{0,i} d_i \boldsymbol{\varphi}(\mathbf{x_i}) \qquad \text{Eq. 2.23}$$

$$b_0 = d_S(1 - \xi_S) - \boldsymbol{\omega_0}^T \boldsymbol{\varphi}(\mathbf{x_S}) \qquad \text{Eq. 2.24}$$

onde $\mathbf{x_S}$ é um vetor de suporte qualquer, definido pelas regras estipuladas para cada tipo de classificador SVM abordado nas seções anteriores, ξ_S é igual a zero para o classificador L1SVM, ou é dado pela eq. 2.21 para o classificador L2SVM ($\alpha_S \neq 0$), e d_S é igual a 1 se $\mathbf{x_S} \in$ C1, ou igual a –1, caso contrário.

Na prática, as transformações não lineares, $\varphi_i(\mathbf{x})$, para $1 \leq i \leq Q$, não são conhecidas a priori, e $\boldsymbol{\omega_0}$ e b_0 não podem ser obtidos diretamente a partir das eqs. 2.23 e 2.24. Porém, a "distância" de um ponto $\mathbf{x}$ ao hiperplano ótimo ($dist(\mathbf{x})$) é dada por:

$$dist(\mathbf{x}) = \boldsymbol{\alpha}_0^T \mathbf{D}\mathbf{k}(\mathbf{x}) + b_0 \qquad \text{Eq. 2.25}$$

$$b_0 = d_S - \boldsymbol{\alpha}_0^T \mathbf{D}\mathbf{k}(\mathbf{x_S}) \qquad \text{Eq. 2.26}$$

onde $\mathbf{k(x_S)}^T = [\boldsymbol{\varphi}(\mathbf{x_S})^T\boldsymbol{\varphi}(\mathbf{x_1}), \boldsymbol{\varphi}(\mathbf{x_S})^T\boldsymbol{\varphi}(\mathbf{x_2}), \ldots, \boldsymbol{\varphi}(\mathbf{x_S})^T\boldsymbol{\varphi}(\mathbf{x_N})]$, $\mathbf{x}_S$ é um vetor de suporte qualquer e $\mathbf{k(x)}^T = [\boldsymbol{\varphi}(\mathbf{x})^T\boldsymbol{\varphi}(\mathbf{x_1}), \boldsymbol{\varphi}(\mathbf{x})^T\boldsymbol{\varphi}(\mathbf{x_2}), \ldots, \boldsymbol{\varphi}(\mathbf{x})^T\boldsymbol{\varphi}(\mathbf{x_N})]$. Portanto a classificação de um padrão $\mathbf{x}$, não utilizado na fase de treinamento, ainda é possível se o *kernel* $\boldsymbol{\varphi}(\mathbf{x})^T\boldsymbol{\varphi}(\mathbf{y})$ for bem definido para quaisquer duas observações $\mathbf{x}$ e y. Neste caso, o vetor $\mathbf{x}$ será classificado como pertencendo a C1 se $dist(\mathbf{x})$ for maior que zero, e à classe C2, caso contrário.

Para não tornar a estimativa do parâmetro b_0 sensível à escolha do vetor de suporte, usualmente ele é obtido por meio da média aritmética de todos os valores calculados pela eq. 2.26 para cada um dos vetores de suporte obtidos na fase de treinamento. Além disso, as eqs. 2.25 e 2.26 podem ser simplificadas, levando-se em consideração apenas os termos em que $\alpha_{0,i}$, $1 \leq i \leq N$, é diferente de zero. A figura 2.3 mostra a arquitetura de um classificador SVM conforme descrito pelas eqs 2.25 e 2.26.

Em resumo, o vetor de entrada $\mathbf{x} \in \Re^M$ é um vetor de observação qualquer. Os vetores $\mathbf{x_i} \in \Re^M$, $i = 1, 2, \ldots, L$, $L \leq N$, são observações utilizadas na fase de treinamento associadas a um multiplicador de Lagrange $\alpha_i \neq 0$ e cuja classificação é dada por d_i ($d_i = \pm 1$). Em outras palavras, os vetores $\mathbf{x_1}$, $\mathbf{x_2}$, … e $\mathbf{x_L}$ são vetores de suporte que podem estar localizados em um dos dois hiperplanos de suporte ou do lado errado do hiperplano de suporte associado a sua classe. O vetores $\mathbf{k(x)}$ é obtido a partir da avaliação da função *kernel* entre os vetores de suporte e o vetor $\mathbf{x}$. Algumas opções

para o *kernel* são dadas na tabela 2.1 [3]. Por sua vez, b_0 é a polarização ótima, enquanto que *dist*(**x**) é a saída numérica que representa o resultado da classificação. Se *dist*(**x**) for positivo, então o vetor de entrada **x** pertence à classe 1, caso contrário, pertence à classe 2.

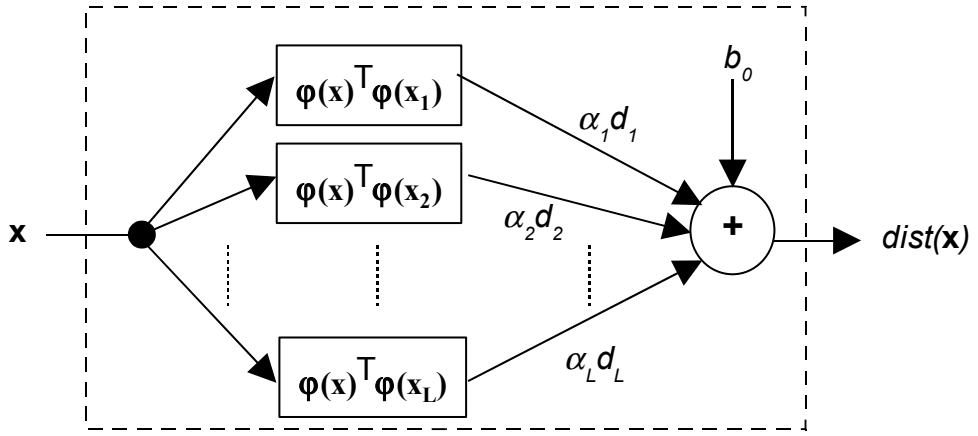

Figura 2.3: Implementação de um classificador SVM usando kernel (K) não linear.

Tabela 2.1: Exemplos de Kernel não linear

KERNEL	DEFINIÇÃO *	COMENTÁRIOS
Polinomial (**POLY**)	$(\mathbf{x_i}^T\mathbf{x_j} + 1)^P$	p é um parâmetro escolhido pelo usuário
Função de Base Radial (**RBF**)	$\text{Exp}\left(\frac{-1}{2\sigma^2}\left\|\mathbf{x_i} - \mathbf{x_j}\right\|^2\right)$	σ é um parâmetro escolhido pelo usuário
Duas camadas de Perceptron (**MLP**)	$\text{Tanh}(\beta_0\,\mathbf{x_i}^T\mathbf{x_j} + \beta_1)$	β_0 e β_1 são parâmetros escolhidos pelo usuário

- Exp é a função exponencial, Tanh é a tangente hiperbólica, e $\mathbf{x_i}$ e $\mathbf{x_j}$ são duas observações quaisquer.

2.5 Expansão do Classificador SVM para Z Classes

A construção de um classificador para mais do que duas classes utilizando a técnica SVM é feita por meio da combinação de vários classificadores SVM típicos. Três métodos diferentes são encontrados na literatura [6,7,8]. O primeiro deles, denominado **1vsR** (1 contra o Resto), é apresentado na figura 2.4 (para Z=4) e consiste na criação de Z classificadores SVM cujo i-ésimo classificador é responsável por separar a i-ésima classe das demais. As saídas destes classificadores são encaminhadas a um bloco decisor.

Idealmente, deve existir apenas um classificador cuja saída é positiva, porém pode existir mais do que um classificador nesta condição no caso de classes que não são linearmente separáveis. Da mesma forma, pode ocorrer o caso em que nenhum classificador tenha apresentado uma saída positiva. Logo, a regra de decisão consiste na escolha do classificador SVM que obteve a maior saída, positiva ou não. No caso em que nenhum classificador apresentou uma saída positiva, uma alternativa à regra usualmente empregada é a de indicar na saída do bloco decisor que a observação em questão não foi classificada ou não pertence a nenhuma das classes.

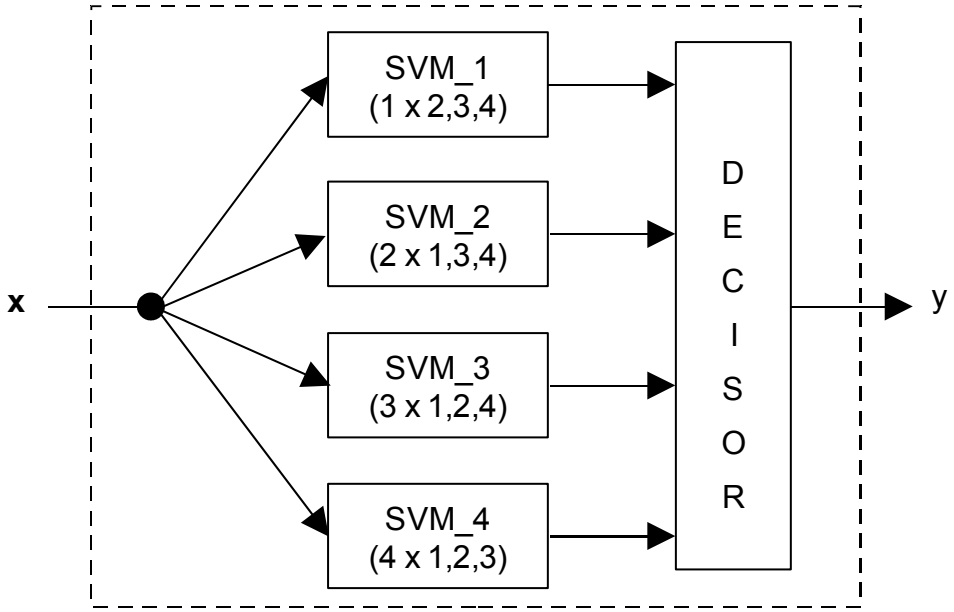

Figura 2.4: Classificador 1vsR para 4 classes.

O segundo método, denominado **1vs1** (1 contra 1), é apresentado na figura 2.5 (para Z = 4) e consiste na criação de Z(Z-1)/2 classificadores SVM. Cada classificador é responsável em separar a classe i da classe j, para i < j; i = 1, 2,…, Z-1; e j = 2, …, Z. Novamente, a saída de cada classificador é encaminhada a um bloco decisor que pode empregar uma entre as duas regras definidas a seguir:

1ª regra: A saída de cada classificador conta como um voto para a classe escolhida por ele e a saída do bloco decisor é igual a classe com o maior numero de votos.

2ª regra: A saída do bloco decisor é obtida a partir de uma função "AND" entre as saídas de cada classificador.

A primeira regra de decisão foi apresentada por Friedman [9] e não esclarece o que fazer em caso de empate. Porém, parece bastante óbvio que se apenas duas classes

empatarem em número de votos, vence a que tiver ganho o confronto direto. Já a segunda regra foi apresentada por Knerr e outros em [10] e, a princípio, se não houver a determinação de uma classe, então assume-se que a observação analisada não foi classificada, ou não pertence a nenhuma das classes.

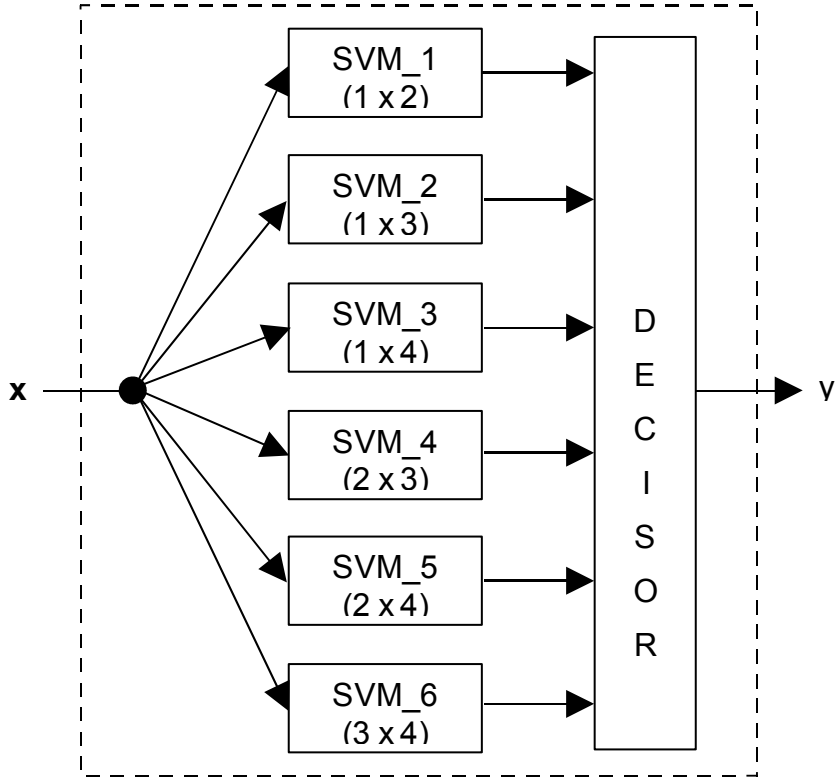

Figura 2.5: Classificador 1vs1 para 4 classes.

O terceiro método, apresentado por J. Platt e outros [6], consiste em usar os mesmos Z(Z-1)/2 classificadores SVM utilizados no método **1vs1,** porém arrumados na forma de um grafo de decisão binária, muito parecido com uma árvore de decisão binária, onde cada classificador é representado por um nó do grafo. As regras para a construção do grafo são as seguintes.

- A raiz da árvore corresponde ao classificador que separa as classes 1 e Z.
- O i-ésimo classificador da j-ésima camada da árvore, $1 \leq j < Z-1$, está ligado aos classificadores (filhos) de índice i (a esquerda) e i+1 (a direita) na camada de índice j+1.
- Se um classificador na j-ésima camada, $1 \leq j < Z-1$, é utilizado para separar as classes r e s, $r < s$, $r \in \{1, 2, \ldots, Z-1\}$ e $s \in \{2, 3, \ldots, Z\}$, então

o seu filho à direita separa as classes r+1 e s, enquanto que o seu filho à esquerda separa as classes r e s-1.

- Como cada classificador decide por apenas um dos dois caminhos (filhos) possíveis, então apenas um classificador entre todos os classificadores da j-ésima camada é utilizado.
- A classificação é dada pela saída do classificador utilizado na camada de índice Z-1.

Este método, denominado **DAGSVM** (Directed Acyclic Graph – Support Vector Machine) é ilustrado na figura 2.6 para Z = 4. Repare que para classificar uma observação, apenas Z-1 classificadores do grafo são necessários, um número menor do que os Z classificadores utilizados no método **1vsR** e do que os Z(Z-1)/2 classificadores utilizados no método **1vs1**.

Os três métodos de generalização abordados nesta seção foram comparados em [6,7], onde cada classificador SVM utilizado é do tipo L1SVM. Hsu e Lin [7] também compararam dois classificadores SVM multiclasses obtidos a partir da formulação de um único problema de otimização, mas eles se mostraram pouco competitivos na prática, devido principalmente ao grande tempo de treinamento necessário e à complexidade do processo de otimização, por isso não são abordados neste trabalho. As simulações computacionais realizadas nos dois trabalhos citados basearam-se em problemas práticos, cujos dados estão disponíveis em repositórios na Internet [11].

A comparação foi realizada com base no tempo gasto durante a fase de treinamento; tempo gasto durante a fase de classificação usando os classificadores já treinados; e taxa de acerto geral na classificação. Os três métodos obtiveram desempenho similar no que se refere à taxa de erro de classificação. Os métodos **1vs1** e **DAGSVM** obtiveram tempos de treinamento similares e sempre menores do que o tempo gasto na fase de treinamento para o método **1vsR**, variando de acordo com o problema analisado entre 2,2 a 11,5 vezes em [6] ou 1,5 a 6,5 vezes menores em [7]. Este resultado já era esperado, pois apesar de os métodos **1vs1** e **DAGSVM** terem mais classificadores para serem treinados, o treinamento de cada classificador SVM leva em conta um número menor de observações do que o utilizado no método **1vsR**.

É importante destacar também que os métodos **1vs1** e **DAGSVM** têm a mesma fase de treinamento se os parâmetros utilizados para treinar cada classificador SVM forem iguais. Isto nem sempre ocorreu nos trabalhos citados, pois a maior taxa de acerto nem sempre ocorreu para os mesmos valores de C e σ_{RBF}, respectivamente, parâmetro de regularização do classificador SVM e largura do *kernel* RBF utilizado.

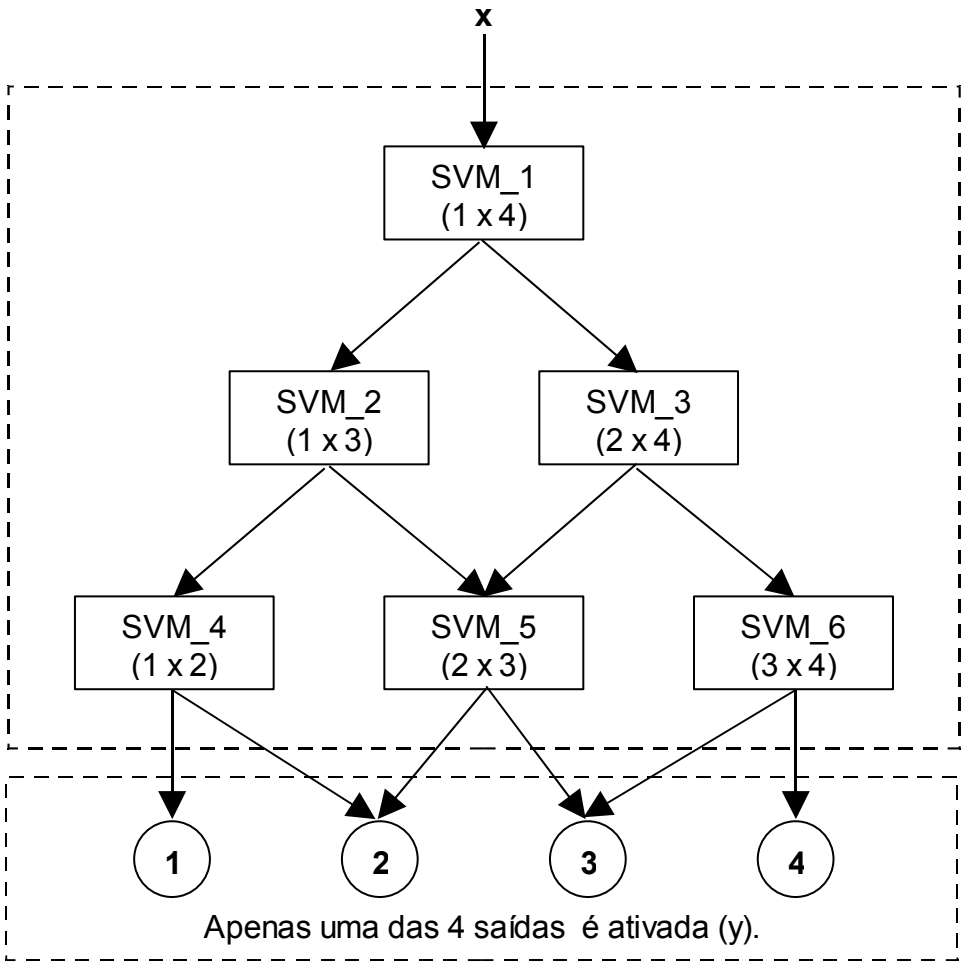

Figura 2.6: Classificador DAGSVM para 4 classes.

Por último, o método **DAGSVM** se mostrou o mais rápido método na fase de testes, sendo sempre um pouco melhor que o segundo colocado, o método **1vs1**. Pelo exposto nesta seção, poder-se-ia acreditar que o método **DAGSVM** seria um pouco mais rápido do que o método **1vsR** e ambos muito mais rápidos do que o método **1vs1**, uma vez que a classificação final de uma observação, $\mathbf{x} \in \Re^M$, é obtida após Z-1, Z e Z(Z-1)/2 classificações, respectivamente, onde Z é o número de classes presentes. No entanto, o tempo de teste não é dominado pelo número de classificadores envolvidos no processo de classificação, mas sim pelo número de vezes que um *kernel* $K(\mathbf{x},\mathbf{x_S})$ é

avaliado, o que está intimamente relacionado ao número de diferentes vetores de suporte, $\mathbf{x}_S$, obtidos durante a fase de treinamento.

Do exposto acima, pode-se concluir que os métodos **DAGSVM** e **1vs1** são os mais indicados para expandir classificadores SVM de duas para Z classes, pois oferecem os menores tempos de treinamento e teste, mantendo um desempenho competitivo com o método **1vsR** no que se refere à taxa de acerto. No entanto, deve-se destacar que o método **DAGSVM** pode vir a privilegiar determinadas classes em detrimento de outras, independentemente do tipo de classificador que é utilizado em cada nó.

A figura 2.7 ilustra um determinado nó do grafo que representa o classificador responsável por separar as classes i e j . O caminho à direita é tomado se o classificador decidir-se pela classe j, enquanto que o caminho à esquerda é tomado se o classificador decidir-se pela classe i. Seja $\mathbf{x} \in \Re^M$ uma observação pertencente à classe L . Define-se como q_L a probabilidade do caminho à esquerda ser tomado caso $L \neq i, j$. Define-se como p_L a probabilidade do caminho certo ser tomado caso $L = i$ ou $L = j$.

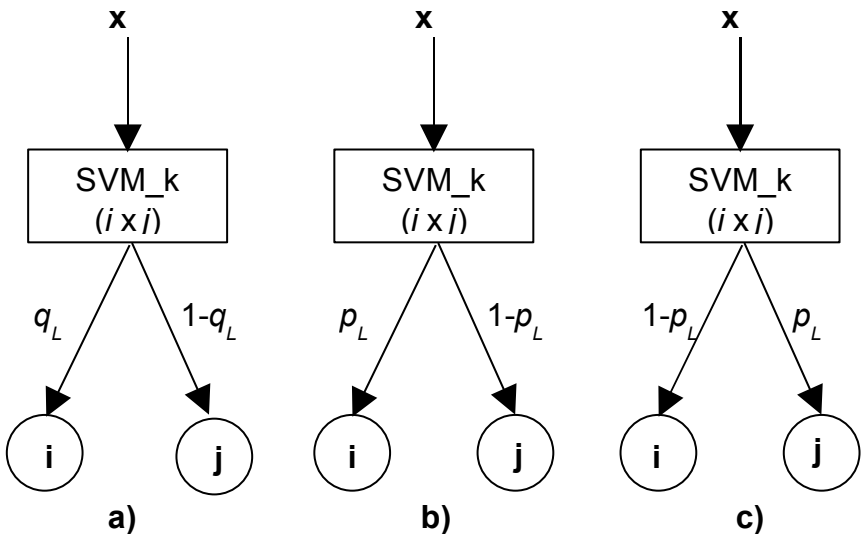

Figura 2.7: Classificação da observação x ∈ à classe L usando um classificador SVM (*i*x*j*). a) L≠*i*,*j*. b) L=*i* e L≠*j*. c) L=*j* e L≠*i*.

Analisando o exemplo ilustrado na figura 2.6 é possível calcular a probabilidade de acerto (P_A) para cada classe, ou seja:

$$P_A(\mathrm{L}) = \begin{cases} p_L^{\,3}, \text{ para } \mathrm{L} = 1,4 \\ q_L p_L + (1 - q_L) p_L^{\,2}, \text{ para } \mathrm{L} = 2 \\ (1 - q_L) p_L + q_L p_L^{\,2}, \text{ para } \mathrm{L} = 3 \end{cases}$$

Mais especificamente, se q_L=1/2 e p_L=p para L=1,2,3,4, então:

$$P_A(\mathrm{L}) = \begin{cases} p^3, \text{ para } \mathrm{L} = 1,4 \\ \frac{1}{2}\left(p + p^2\right), \text{ para } \mathrm{L} = 2,3 \end{cases}$$

Nas condições acima, o método **1vs1** apresenta ao decisor saídas com probabilidades de acerto iguais, o que não ocorre para o método **DAGSVM**, onde as classes 1 e 4 têm uma probabilidade de acerto menor do que as classes 2 e 3. Por este motivo, entre outros, o método **1vs1** foi escolhido para ser empregado no classificador de modulação proposto neste trabalho e apresentado no capítulo 5.

2.6. Máquinas Baseadas em Vetores de Suporte do Tipo Proximal

2.6.1 Formulação matemática para o caso linear

Fung e Mangasarian [1] descrevem em seu artigo uma modificação do SVM tradicional, chamado "Proximal Support Vector Machine" (PSVM). Esta técnica também foi usada por Li e outros em [8] associada ao algoritmo DAGSVM para generalização deste classificador SVM para Z classes. No classificador PSVM linear, tenta-se minimizar o módulo do vetor erro de classificação $\boldsymbol{\xi}$; ao mesmo tempo em que se tenta maximizar a margem de separação das classes, através da minimização do módulo do vetor de pesos $\boldsymbol{\omega}$; e minimizar a proximidade do hiperplano ótimo à origem, por meio da minimização da polarização b ao quadrado.

O parâmetro C, fator de regularização usado no SVM tradicional, continua a ser usado para ponderar a parcela relativa ao erro de classificação dentro da função-objetivo, porém, diferentemente do SVM tradicional, a inequação de restrição é substituída por uma igualdade, fato que muda inteiramente a natureza dos hiperplanos $\mathbf{x}^T\boldsymbol{\omega}+b=\pm 1$, que agora passam a se chamar hiperplanos proximais, pois são aqueles em

torno dos quais as observações de cada classe se concentram. Ou seja, o algoritmo PSVM procura solucionar o seguinte problema:

$$J = \min_{\boldsymbol{\omega},b,\boldsymbol{\xi}}\left\{\frac{1}{2}C\boldsymbol{\xi}^T\boldsymbol{\xi} + \frac{1}{2}\left(\boldsymbol{\omega}^T\boldsymbol{\omega} + b^2\right)\right\} \text{ sujeita à restrição } \boldsymbol{\xi} = \mathbf{e} - \mathbf{D}\left(\mathbf{X}^T\boldsymbol{\omega} + \mathbf{e}b\right) \qquad \text{Eq. 2.27}$$

Repare que o vetor de erro $\boldsymbol{\xi}$ pode ter elementos positivos ou negativos, ao contrário do que ocorre nos classificadores abordados nas seções anteriores. Isto simplifica a determinação do problema dual, uma vez que não é necessário atribuir um novo multiplicador de Lagrange para cada elemento do vetor de erros. Enfim, a solução encontrada para o problema apresentado pela eq. 2.27, após a aplicação dos multiplicadores de Lagrange ($\boldsymbol{\alpha}$) e das condições de Karush–Kuhn–Tucker [1], é dada pelas eqs. 2.28 a 2.31 a seguir

$$\boldsymbol{\alpha}_0 = \left(\mathbf{I}/C + \mathbf{H}\mathbf{H}^T\right)^{-1}\mathbf{e} \qquad \text{Eq. 2.28}$$

$$\boldsymbol{\omega}_0 = \mathbf{X}\mathbf{D}\boldsymbol{\alpha}_0 \qquad \text{Eq. 2.29}$$

$$b_0 = \mathbf{e}^T\mathbf{D}\boldsymbol{\alpha}_0 \qquad \text{Eq. 2.30}$$

$$\boldsymbol{\xi}_0 = \boldsymbol{\alpha}_0/C \qquad \text{Eq. 2.31}$$

onde $\mathbf{H}=\mathbf{D}[\mathbf{X}^T \ -\mathbf{e}]$. Nota-se que o vetor de erro é uma versão proporcional do vetor que contém os multiplicadores de Lagrange. Logo, serão considerados como vetores de suporte toda as observações presentes na matriz $\mathbf{X}$ associadas a um erro não nulo ($\xi_i \neq 0 \leftrightarrow \alpha_i \neq 0,\ 0 \leq i \leq \mathrm{L}$).

A principal vantagem desta técnica em comparação com o SVM tradicional é a rapidez para a obtenção da solução ótima, obtida de forma analítica por meio da solução de um pequeno sistema de equações lineares. A eq. 2.28 mostra que a solução para este problema consiste em inverter uma matriz NxN, onde N é o número de observações presentes. A solução encontrada pode ser simplificada utilizando o lema de inversão de matrizes para diminuir a complexidade da inversão da matriz ($\mathbf{I}/C+\mathbf{H}\mathbf{H}^T$). Deste modo, a complexidade do problema diminui expressivamente, pois será necessário apenas inverter uma matriz (M+1)x(M+1), onde M (M<N) é o número de parâmetros que compõem uma observação. Ou seja, não há necessidade de usar algoritmos de otimização lentos e/ou complexos, pois o vetor de multiplicadores de Lagrange utilizados para a solução do problema pode ser obtido diretamente da equação a seguir:

$$\boldsymbol{\alpha}_0 = C\left[\mathbf{I} - \mathbf{H}\left(\mathbf{I}/C + \mathbf{H}^T\mathbf{H}\right)^{-1}\mathbf{H}^T\right]\mathbf{e} \qquad \text{Eq. 2.32}$$

O artigo de Fung e Mangasarian [1] não deixa clara a contribuição obtida com a minimização da distância do hiperplano ótimo à origem, embora tente comprovar através de experimentos computacionais que a modificação feita ao SVM tradicional não prejudica seu desempenho na classificação de padrões. A inclusão do parâmetro *b* à função-objetivo já havia sido explorada por Mangasarian e Musicant na construção do SVM ativo (ASVM) [12], precursor do PSVM, que trabalha com a mesma função-objetivo descrita pela eq. 2.27, porém mantém como restrição a mesma inequação presente no SVM tradicional.

Neste último artigo fica claro que o parâmetro *b* foi introduzido na função-objetivo para eliminar a restrição ($\boldsymbol{\alpha}^T\mathbf{De}=0$) presente no problema dual que define o SVM tradicional (eq. 2.12) e obter, ao mesmo tempo, uma solução analítica para *b* em função dos multiplicadores de Lagrange. Ou seja, o uso do módulo do vetor de erros e principalmente a inclusão do parâmetro *b* na função-objetivo são artifícios para simplificar o problema e permitir a obtenção de uma solução analítica cuja implementação leva a um algoritmo com menor complexidade computacional do que os algoritmos utilizados para o treinamento de classificadores SVM tradicionais.

No próximo capítulo, são apresentadas modificações ao PSVM com a finalidade de eliminar a dependência da função-objetivo ao parâmetro *b*, tendo em vista que a minimização da distância do hiperplano ótimo à origem pode piorar o desempenho do classificador, como é mostrado na seção 3.5.

2.6.2 Formulação matemática para o caso não linear

As eqs 2.30 e 2.32 descrevem o algoritmo PSVM linear, porém, conforme discutido na seção 2.4, o desempenho dos classificadores SVM empregados na separação de classes não linearmente separáveis pode ser melhorado se o espaço de entrada, $\Re^M$, for mapeado em um espaço de maior dimensão, usualmente chamado de espaço de características, por meio de um conjunto de transformações não lineares $\boldsymbol{\varphi}(\mathbf{x})$.

Portanto, o algoritmo PSVM deve ser generalizado para permitir a utilização de um *kernel* não linear. Esta generalização deve ser feita pela substituição da observação

$\mathbf{x}_i$ por $\boldsymbol{\varphi}(\mathbf{x}_i)$, e, conseqüentemente, $\boldsymbol{\omega}^T\boldsymbol{\omega}$ por $\boldsymbol{\alpha}^T\mathbf{DKD}\boldsymbol{\alpha}$ e $\mathbf{X}^T\boldsymbol{\omega}$ por $\mathbf{KD}\boldsymbol{\alpha}$, respectivamente na função-objetivo e na restrição que definem a formulação matemática do problema. No SVM tradicional, isto implicou na substituição da matriz $\mathbf{X}^T\mathbf{X}$ pela matriz de *kernel* **K** no problema dual definido a partir dos multiplicadores de Lagrange.

No entanto, a generalização do algoritmo PSVM para o caso não linear foi realizada por Fung e Mangasarian [1] substituindo apenas $\mathbf{X}^T\boldsymbol{\omega}$ por $\mathbf{KD}\boldsymbol{\alpha}$ e substituindo $\boldsymbol{\omega}^T\boldsymbol{\omega}$ por $\boldsymbol{\alpha}^T\boldsymbol{\alpha}$. Na prática, isto implicou na substituição da matriz **X** (ou $\mathbf{X}^T$) pela matriz de *kernel* **K**, e na substituição da matriz $\mathbf{X}^T\mathbf{X}$ pela matriz $\mathbf{K}^T\mathbf{K}$ na obtenção do vetor de multiplicadores de Lagrange. Logo, o vetor de multiplicadores de Lagrange para o PSVM não linear continua sendo calculado pela eq. 2.28, porém fazendo $\mathbf{H}=\mathbf{D}[\mathbf{K}^T \ -\mathbf{e}]$. Não há nenhuma vantagem em se usar a eq. 2.32 para o cálculo de $\boldsymbol{\alpha}_0$ porque a matriz **K** é quadrada. Finalmente, a distância de uma observação **x** é dada por:

$$dist(\mathbf{x}) = \mathbf{k}^T\mathbf{KD}\boldsymbol{\alpha}_0 + b_0 \qquad \text{Eq. 2.33}$$

onde *b* continua sendo calculado pela eq. 2.30, **K** é a matriz de *kernel* utilizada no treinamento, e $\mathbf{k}^T = [\boldsymbol{\varphi}(\mathbf{x})^T\boldsymbol{\varphi}(\mathbf{x}_1), \ldots, \boldsymbol{\varphi}(\mathbf{x})^T\boldsymbol{\varphi}(\mathbf{x}_N)]$.

Como será visto em um estudo de caso apresentado no capítulo 3, esta generalização pouco ortodoxa implicará na degradação de desempenho do algoritmo PSVM não linear utilizado para separar classes que não são linearmente separáveis.

2.7. Comparação entre o PSVM e o Estimador de Ridge

Agarwal em [13] questiona a novidade do algoritmo PSVM argumentando que ele é a solução de um problema matemático idêntico, fazendo-se algumas considerações importantes que serão detalhadas em breve, ao que originou a formulação do Regressor de Ridge, conhecido pela comunidade científica há muitos anos. Além disso, tenta justificar a inclusão proposital de polarização argumentado que o erro médio quadrático é função da variância e da polarização do estimador, e que ao introduzi-la, pode-se diminuir a variância de tal forma que o erro médio quadrático também diminua.

Esta seção tem por objetivo apresentar uma avaliação correta da relação existente entre o algoritmo PSVM e o Regressor de Ridge com o intuito de provar que os dois métodos são muito diferentes, apesar da formulação matemática similar. Esta seção também tem por objetivo concluir sobre a necessidade ou não de se incluir polarização no processo de estimação dos coeficientes do hiperplano ótimo utilizado na separação de classes em aplicações de reconhecimento de padrões.

2.7.1. O Regressor de Ridge

Em um problema de Regressão Linear Múltipla (RLM), supõem-se que os pontos $\{(\mathbf{x}_i, y_i),\ i=1,2,...,N\}$ obtidos em N experiências fazem parte de um hiperplano, porém foram deslocados de seus pontos originais devido a erros de medição aleatórios. O dado y_i corresponde ao valor medido na i-ésima experiência utilizando o conhecido vetor de parâmetros $\mathbf{x}_i$. Sendo assim, o problema em questão consiste na estimação dos coeficientes de um hiperplano que melhor se aproxime dos dados observados. Usualmente a métrica utilizada para isso é a minimização do erro médio quadrático, o que leva a seguinte definição matemática para o problema:

"Minimizar a função-objetivo dada por

$$J(\mathbf{r}) = \frac{\mathbf{r}^T\mathbf{r}}{2}$$

onde $\mathbf{r} = \mathbf{y} - \mathbf{X}^T\boldsymbol{\omega} - b\mathbf{e}$

e $\mathbf{X}$ e $\mathbf{y}$ são conhecidos."

Nesta definição, a matriz $\mathbf{X}$ é composta pelos vetores coluna $\mathbf{x}_i$ (i=1,...,N), enquanto que o vetor $\mathbf{y}$ é a coletânea de observações individuais y_i de cada experiência. Por sua vez, $\mathbf{e}$ é um vetor coluna, cujos elementos são todos iguais a um.

A função-objetivo apresentada nesta formulação matemática pode ser reescrita conforme a equação abaixo

$$J(\boldsymbol{\beta}) = \frac{(\mathbf{y} - \mathbf{H}\boldsymbol{\beta})^T(\mathbf{y} - \mathbf{H}\boldsymbol{\beta})}{2} \qquad \text{Eq. 2.34}$$

onde $\mathbf{H} = [\mathbf{e}\ \ \mathbf{X}^T]$ é a matriz de parâmetros e $\boldsymbol{\beta}^T = [b\ \ \boldsymbol{\omega}^T]$ é o vetor de coeficientes do hiperplano**.** A solução para a minimização desta função-objetivo é dada por:

$$\boldsymbol{\beta}_0 = \left(\mathbf{H}^T\mathbf{H}\right)^{-1}\mathbf{H}^T\mathbf{y} \qquad \text{Eq. 2.35}$$

O Regressor de Ridge foi inicialmente proposto por Arthur e Robert [14] ao perceber que os coeficientes dos hiperplanos obtidos pela eq. 2.35 poderiam ser instáveis, extremamente grandes em módulo e com alguns sinais trocados em determinadas circunstâncias. A partir daí, eles sugeriram a reformulação do problema, acrescentado a minimização da norma do vetor de coeficientes ponderada pelo fator de regularização C. Em outras palavras, o novo problema passa a ser a minimização da função-objetivo apresentada pela eq.2.36. A solução deste problema é dada pela eq. 2.37, onde **I** é a matriz de identidade de dimensão apropriada.

$$J(\boldsymbol{\beta}) = \mathrm{C}\frac{\left(\mathbf{y} - \mathbf{H}\boldsymbol{\beta}\right)^T\left(\mathbf{y} - \mathbf{H}\boldsymbol{\beta}\right)}{2} + \frac{\boldsymbol{\beta}^T\boldsymbol{\beta}}{2} \qquad \text{Eq. 2.36}$$

$$\boldsymbol{\beta}_0 = \left(\frac{\mathbf{I}}{\mathrm{C}} + \mathbf{H}^T\mathbf{H}\right)^{-1}\mathbf{H}^T\mathbf{y} \qquad \text{Eq. 2.37}$$

Pode-se concluir que a minimização do vetor de coeficientes, incluindo aí o parâmetro *b*, não está relacionada à inclusão de polarização para minimizar o erro médio quadrático a partir da diminuição conjunta da variância dos coeficientes estimados, mas sim, à necessidade de se obter uma solução estável e factível para problemas mal-condicionados. A mesma solução foi proposta por Tikhonov no contexto da álgebra linear para a solução de sistemas lineares mal-condicionados [15].

2.7.2. Analogia entre o algoritmo PSVM e o Regressor de Ridge

Para transformar o problema de minimização da função-objetivo apresentada na eq. 2.36, que deu origem ao Regressor de Ridge, na formulação matemática que deu origem ao algoritmo PSVM (eq. 2.27), Agarwal fez as seguintes substituições

- $\mathbf{r} = \mathbf{D}\xi$ (o que implica em $\mathbf{r}^T\mathbf{r} = \xi^T\xi$)
- $\mathbf{y} = \mathbf{De}$

onde **D** é a matriz de classificação e ξ é o vetor de erro no contexto de uma máquina baseada em vetores de suporte aplicada a classificação binária.

A partir destas substituições, verifica-se que o algoritmo PSVM poderia ser visto como um problema de regressão linear múltipla onde a observação da i-ésima experiência hipotética, y_i, é igual a 1 se $\mathbf{x_i}$ pertence a classe C1, ou igual a -1 caso contrário. É importante destacar neste momento que não faz nenhum sentido assumir que os pontos $\{\mathbf{x_i}, y_i = \pm 1\}$ façam parte de um hiperplano, logo o algoritmo PSVM não pode ser considerado igual ao Regressor de Ridge. Além disso, também é importante ressaltar que o algoritmo PSVM linear não pode ser interpretado estatisticamente como a moda posterior de $\boldsymbol{\beta}$ supondo que o vetor de observações $\mathbf{y}$ tem distribuição gaussiana com média $\mathbf{X\beta}$ e variância $\sigma^2\mathbf{I}$, mesmo porque, como já foi visto na analogia entre os dois métodos, $\mathbf{y}$ corresponde ao vetor de classificação e portanto é um vetor conhecido e determinístico.

Para melhor ilustrar de forma prática a diferença entre o Regressor de Ridge e o algoritmo PSVM, suponha que N valores são observados em uma experiência hipotética. O modelo matemático que representa a saída da i-ésima experiência é $z_i = at_i - b + n_i$, onde t_i é o í-ésimo instante de tempo e n_i é uma variável aleatória gaussiana, de média zero e variância σ^2, que modela o erro de medição e que é independente de n_j para $j \neq i$. A aplicação do Regressor de Ridge (no contexto RLM) implica na solução dada pela eq. 2.37, onde:

$$\mathbf{H} = \begin{bmatrix} 1 & t_1 \\ 1 & t_2 \\ \vdots & \vdots \\ 1 & t_N \end{bmatrix} \qquad \text{Eq. 2.38}$$

$$\mathbf{y}^T = \begin{bmatrix} z_1 & z_2 & \cdots & z_N \end{bmatrix} \qquad \text{Eq. 2.39}$$

Por sua vez para aplicar a técnica PSVM ao problema, suponha que os pontos obtidos foram divididos em duas classes: a classe C1 que contém os pontos acima da reta $f(t) = at - b$, onde a e b são constantes conhecidas; e a classe C2 que contém os demais pontos. Empregando a técnica PSVM, sabe-se que o vetor de coeficientes é dado pela eq. 2.37, sendo:

$$\mathbf{H} = \begin{bmatrix} 1 & t_1 & z_1 \\ 1 & t_2 & z_2 \\ \vdots & \vdots & \vdots \\ 1 & t_N & z_N \end{bmatrix} \qquad \text{Eq. 2.40}$$

$$\mathbf{y}^T = (\mathbf{De})^T = \begin{bmatrix} d_1 & d_2 & \cdots & d_N \end{bmatrix} \qquad \text{Eq. 2.41}$$

onde d_i é igual a 1 ou -1 dependendo da classe a que pertence o ponto $\{t_i, y_i\}$.

Embora as duas soluções não sejam explicitamente calculadas aqui, elas são diferentes e, portanto, a técnica PSVM não pode ser considerada igual ao Regressor de Ridge. Além disso como já mencionado anteriormente os pontos $\{t_i, y_i, d_i, i=1,2,..., N\}$ não formam um plano, o que não possibilita a interpretação da técnica PSVM como um problema de RLM, que tem o Regressor de Ridge como uma solução possível.

Enfim, o algoritmo PSVM e o Regressor de Ridge têm formulações matemáticas similares, porém não podem ser considerados iguais. Devido aos diferentes contextos, a analogia entre eles não traz nenhuma informação útil e, principalmente, não traz nenhuma indicação que a inclusão do parâmetro *b* na função-objetivo a ser minimizada no contexto da classificação binária esteja associada diretamente a melhora de desempenho do classificador (hiperplano ótimo) estimado.

Pelo contrário, cabe aqui ressaltar, conforme mostrado pela eq. 2.7, que a margem de separação entre classes é inversamente proporcional apenas ao módulo do vetor de pesos $\boldsymbol{\omega}$, e levar em conta a polarização *b* na função-objetivo a ser minimizada pode acarretar em um hiperplano polarizado e na degradação do desempenho do classificador PSVM, como será visto na seção 3.5. Um novo algoritmo de treinamento é proposto no próximo capítulo, com o intuito de corrigir este problema e o problema associado à forma como o PSVM não linear foi derivado.

3 ALGORITMO PROPOSTO – UPSVM

3.1 Introdução

Neste capítulo o classificador Unbiased PSVM é proposto para o caso linear (seção 3.2) e para o caso não linear (seção 3.3). A motivação para sua derivação está baseada em dois problemas encontrados no classificador PSVM:

- a probabilidade de acerto do classificador PSVM decresce de forma significativa à medida que o parâmetro C diminui abaixo de um determinado valor, devido a polarização do hiperplano ótimo; e
- a generalização do classificador PSVM linear para o caso não linear não é realizada de forma apropriada [1], o que pode acarretar em degradação do seu desempenho.

A solução desses dois problemas pelo algoritmo proposto não compromete a sua complexidade computacional e rapidez no treinamento. Na verdade, os resultados de simulações computacionais apresentados na seção 3.5 mostram que o algoritmo proposto é ainda mais rápido que o classificador PSVM.

3.2 Formulação Matemática do UPSVM Linear

Para solucionar o primeiro problema descrito acima, deve-se retirar o parâmetro de polarização *b* da função-objetivo dada pela eq. 2.27, garantindo assim a obtenção de um hiperplano classificador não polarizado, independentemente da escolha do parâmetro de regularização *C*.

Enfim, suponha a seguinte função-objetivo:

$$J = \min_{\boldsymbol{\omega},\boldsymbol{\xi}} \left\{ \frac{1}{2} C\boldsymbol{\xi}^T\boldsymbol{\xi} + \frac{1}{2}\boldsymbol{\omega}^T\boldsymbol{\omega} \right\} \text{ sujeita à restrição } \mathbf{D}\left(\mathbf{X}^T\boldsymbol{\omega} + \mathbf{e}b\right) + \boldsymbol{\xi} = \mathbf{e} \qquad \text{Eq. 3.1}$$

O problema em questão continua a ser do tipo proximal, pois, devido à restrição presente na eq. 3.1, os hiperplanos $\mathbf{x}^T\boldsymbol{\omega} + b = \pm 1$ obtidos e definidos em [1] como hiperplanos proximais são aqueles em torno dos quais as observações de cada classe se concentram. Portanto, diferentemente do que ocorre no SVM tradicional com os

chamados hiperplanos de suporte, os hiperplanos proximais não confinam necessariamente a maior parte das observações correspondentes a cada classe.

Com o intuito de obter uma função-objetivo que depende apenas dos parâmetros do hiperplano ótimo, deve-se substituir o vetor $\boldsymbol{\xi}$ da eq. 3.1 pelo obtido a partir da restrição presente na mesma equação, o que leva à minimização da função-objetivo abaixo.

$$J = \min_{\boldsymbol{\omega},b}\left\{ \frac{\mathrm{C}}{2}\left(\boldsymbol{\omega}^T\mathbf{X}\mathbf{X}^T\boldsymbol{\omega} + 2b\boldsymbol{\omega}^T\mathbf{X}\mathbf{e} + b^2\mathrm{N} + \mathrm{N} - 2\boldsymbol{\omega}^T\mathbf{X}\mathbf{D}\mathbf{e} - 2b\mathbf{e}^T\mathbf{D}\mathbf{e}\right) + \frac{1}{2}\boldsymbol{\omega}^T\boldsymbol{\omega}\right\} \qquad \text{Eq. 3.2}$$

Para a minimização desta função-objetivo, o gradiente da eq. 3.2 é calculado e igualado a zero, obtendo-se:

$$\frac{\partial J}{\partial \boldsymbol{\omega}} = \mathrm{C}\left(\mathbf{X}\mathbf{X}^T\boldsymbol{\omega} + b\mathbf{X}\mathbf{e} - \mathbf{X}\mathbf{D}\mathbf{e}\right) + \boldsymbol{\omega} = \mathbf{0} \qquad \text{Eq. 3.3}$$

$$\frac{\partial J}{\partial b} = \mathrm{C}\left(\boldsymbol{\omega}^T\mathbf{X}\mathbf{e} + b\mathrm{N} - \mathbf{e}^T\mathbf{D}\mathbf{e}\right) = 0 \qquad \text{Eq. 3.4}$$

onde **0** é um vetor de dimensão apropriada, cujos elementos são todos iguais a zero.

Resolvendo este sistema de equações lineares, chega-se a:

$$\boldsymbol{\omega}_0 = \left[\frac{\mathbf{I}}{\mathrm{C}} + \mathbf{X}\left(\mathbf{I} - \frac{\mathbf{e}\mathbf{e}^T}{N}\right)\mathbf{X}^T \right]^{-1} \mathbf{X}\mathbf{D}\mathbf{e} - \mathbf{X}\left(\frac{\mathbf{e}\mathbf{e}^T}{N}\right)\mathbf{D}\mathbf{e} \qquad \text{Eq. 3.5}$$

$$b_0 = \frac{\mathbf{e}^T\mathbf{D}\mathbf{e} - \mathbf{e}^T\mathbf{X}^T\boldsymbol{\omega}_0}{N} \qquad \text{Eq. 3.6}$$

As eqs 3.5 e 3.6 podem ser simplificadas caso o número de observações da classe 1 seja igual ao número de observações da classe 2. Neste caso,

$$\boldsymbol{\omega}_\mathbf{0} = \left(\mathbf{I}/\mathrm{C} + \mathbf{X}\left(\mathbf{I} - \mathbf{e}\mathbf{e}^T/\mathrm{N}\right)\mathbf{X}^T\right)^{-1}\mathbf{X}\mathbf{D}\mathbf{e} \qquad \text{Eq. 3.7}$$

$$b_0 = -\frac{\mathbf{e}^T\mathbf{X}^T\boldsymbol{\omega}_\mathbf{0}}{\mathrm{N}} \qquad \text{Eq. 3.8}$$

pois,

$$\mathbf{e}^T\mathbf{D}\mathbf{e} = \sum_{i=1}^{N} D_{i,i} = 0 \qquad \text{Eq. 3.9}$$

Deve-se salientar que tal simplificação não implica em perda de generalidade, uma vez que, mesmo que o banco de dados seja assimétrico, sempre é possível balanceá-lo se algumas observações da classe que possui o maior número de padrões forem desprezadas.

Uma solução analítica foi obtida para o classificador UPSVM linear (eqs. 3.5 e 3.6 ou eqs 3.7 e 3.8). Nota-se que a obtenção dos parâmetros do hiperplano ótimo envolve apenas a inversão de uma matriz MxM, o que é realmente rápido se comparado ao tempo gasto na fase de treinamento empregada no SVM tradicional, e um pouco mais rápido se comparado ao classificador PSVM original. Tendo em vista que o tempo de processamento de um algoritmo depende da máquina em que é implementado, costuma-se comparar a "velocidade" de algoritmos em função de sua complexidade computacional ou número de adições e multiplicações em ponto flutuante. A seção 3.4 apresenta ambos para o algoritmo UPSVM linear, enquanto que o estudo de alguns exemplos é realizado no capítulo 4 para ilustrar as vantagens do método proposto.

3.3 Formulação Matemática do UPSVM Não Linear

Conforme discutido na seção 2.4, o desempenho dos classificadores SVM empregados na separação de classes não linearmente separáveis pode ser melhorado se o espaço de entrada, $\Re^M$, for mapeado em um espaço de maior dimensão, usualmente chamado de espaço de características, por meio de um conjunto de transformações não lineares $\boldsymbol{\varphi}(\mathbf{x})$.

Neste trabalho, a generalização do UPSVM linear para a obtenção de uma versão não linear é realizada de maneira análoga ao que foi feito para a generalização do problema SVM original, ou seja, substituindo em todas as equações o *kernel* linear $\mathbf{X}^T\mathbf{X}$ pela matriz $\mathbf{K}$ (*kernel* não linear). Para tanto, deve-se inicialmente aplicar o lema de inversão de matrizes à eq. 3.5, obtendo

$$\boldsymbol{\omega}_0 = C\left[\mathbf{I} - \mathbf{X}\left(\frac{\mathbf{I}}{C} + \mathbf{A}\mathbf{X}^T\mathbf{X}\right)^{-1}\mathbf{A}\mathbf{X}^T\right]\mathbf{X}\mathbf{D}\mathbf{e} - \mathbf{X}\left(\frac{\mathbf{e}\mathbf{e}^T}{N}\right)\mathbf{D}\mathbf{e} \qquad \text{Eq. 3.10}$$

onde

$$\mathbf{A} = \left(\mathbf{I} - \frac{\mathbf{e}\mathbf{e}^T}{N}\right) \qquad \text{Eq. 3.11}$$

Sabendo que b_0 é dado em função de $\boldsymbol{\omega}_0$ pela eq. 3.6 e supondo $\boldsymbol{\omega}_0 = \mathbf{X}\mathbf{D}\boldsymbol{\alpha}_0$, sendo $\boldsymbol{\alpha}_0$ igual a

$$\boldsymbol{\alpha}_0 = \mathbf{CD}\left[\mathbf{I} - \left(\frac{\mathbf{I}}{\mathrm{C}} + \mathbf{A}\mathbf{X}^T\mathbf{X}\right)^{-1}\mathbf{A}\mathbf{X}^T\mathbf{X} - \frac{\mathbf{e}\mathbf{e}^T}{CN}\right]\mathbf{De} \qquad \text{Eq. 3.12}$$

então o classificador UPSVM não linear é finalmente definido, a partir da substituição na eq. 3.12 da matriz $\mathbf{X}^T\mathbf{X}$ por **K,** como sendo:

$$\boldsymbol{\alpha}_0 = \mathbf{CD}\left[\mathbf{I} - \left(\frac{\mathbf{I}}{\mathrm{C}} + \mathbf{AK}\right)^{-1}\mathbf{AK} - \frac{\mathbf{e}\mathbf{e}^T}{CN}\right]\mathbf{De} \qquad \text{Eq. 3.13}$$

$$b_0 = \frac{\mathbf{e}^T\mathbf{De} - \mathbf{e}^T\mathbf{KD}\boldsymbol{\alpha}_0}{N} \qquad \text{Eq. 3.14}$$

De forma análoga, as eqs. 3.13 e 3.14 podem ser simplificadas se eDe = 0, resultando em:

$$\boldsymbol{\alpha}_0 = \mathbf{CD}\left[\mathbf{I} - \left(\frac{\mathbf{I}}{\mathrm{C}} + \mathbf{AK}\right)^{-1}\mathbf{AK}\right]\mathbf{De} \qquad \text{Eq. 3.15}$$

$$b_0 = -\frac{\mathbf{e}^T\mathbf{KD}\boldsymbol{\alpha}_0}{N} \qquad \text{Eq. 3.16}$$

Por último, a distância de um vetor de teste ao hiperplano ótimo é dada pela eq. 2.25. Repare que não há necessidade de se conhecer o conjunto de transformações não lineares, mas apenas a matriz de *kernels* **K**, assim como os parâmetros ótimos $\boldsymbol{\alpha}_0$ e b_0. Também é importante destacar que a generalização do classificador UPSVM para o caso não linear foi realizada de forma correta, diferente da forma como o PSVM foi generalizado, onde a matriz **X** ou $\mathbf{X}^T$ foi substituída por **K** e, conseqüentemente, a matriz $\mathbf{X}^T\mathbf{X}$ foi substituída por $\mathbf{K}^T\mathbf{K}$, o que pode implicar na diminuição de desempenho.

3.4 Análise do Classificador UPSVM

3.4.1 Complexidade Computacional do UPSVM

O número de operações (multiplicações e adições) em ponto flutuante (flops) calculado para o UPSVM linear, definido pelas equações 3.7 e 3.8, é mostrado na tabela 3.1, onde foram feitas as seguintes considerações:

- a matriz $\mathbf{X}\mathbf{X}^T$ é simétrica;
- multiplicações envolvendo o vetor **e** podem ser expressas por somas; e

- a matriz resultante das operações mostradas na quinta linha da tabela é positiva definida, e portanto pode-se aplicar o algoritmo de Choleski para a obtenção de um sistema triangular equivalente ao original.

No caso do UPSVM não linear descrito pelas eqs. 3.15 e 3.16, o número de flops é mostrado na tabela 3.2. Os valores mostrados não levam em consideração o número de flops necessários para a determinação do *kernel* K, tendo em vista que este será o mesmo para qualquer método SVM, não sendo útil para a comparação entre eles.

Tabela 3.1: Número de flops para o UPSVM linear.

OPERAÇÃO	Nr. de MULT	Nr. de ADIÇÕES
XDe	--	$M(N-1)$
$\mathbf{X}\mathbf{X}^T$	$M(M+1)N/2$	$M(M+1)(N-1)/2$
$\mathbf{X}\mathbf{e}/N$	M	$M(N-1)$
$\mathbf{X}\mathbf{e}(\mathbf{X}\mathbf{e})^T/N$	$(M+1)M/2$	--
$\mathbf{I}/C + \mathbf{X}\mathbf{X}^T + \mathbf{X}\mathbf{e}(\mathbf{X}\mathbf{e})^T/N$	1	$M+M(M+1)/2$
Choleski [16]	$M^3/6$	$M^3/6$
Subst. "Backward/foward" [16]	$M^2 + M + 2$	$M^2 + M$
Cálculo de b_0	M	$M-1$
TOTAL	$N(M^2+M)/2 + M^3/6 + 3M^2/2 + 7M/2 + 3$	$N(M^2+5M)/2 + M^3/6 + 3M^2/2 + M - 1$

Tabela 3.2: Número de flops para o UPSVM não linear.

OPERAÇÃO	Nr. de MULT	Nr. de ADIÇÕES
$\mathbf{e}^T\mathbf{K}$	--	$N(N-1)$
$\mathbf{e}\mathbf{e}^T\mathbf{K}/N$	N	--
AK	--	N^2
AKDe	--	$N(N-1)$
$\mathbf{I}/C + \mathbf{AK}$	1	N
Gauss [16]	N^3	N^3
Subst. "Backward/foward" [16]	$N^2 + N + 2$	$N(N+1)$
C(**De** – resultado anterior)	N	N
Cálculo de b_0	N	$N-1$
TOTAL	$N^3 + N^2 + 4N + 3$	$N^3 + 4N^2 + 2N - 1$

Percebe-se que para valores de N muito maiores do que M, o número de operações em ponto flutuante é aproximadamente igual a $N(M^2+3M)$, no caso do classificador UPSVM linear, de onde se conclui que a sua complexidade computacional é O(N). Para as mesmas condições, o número de flops associado ao UPSVM não linear é aproximadamente igual a $2N^3$, e, portanto, a complexidade computacional do classificador UPSVM não linear é $O(N^3)$.

3.4.2 Estimativa para a Probabilidade de Acerto do UPSVM Linear

Um vetor **y** será corretamente classificado por um classificador SVM ou PSVM se *dist*(**y**) for maior do que zero para **y** pertencente à classe 1, ou menor do que zero caso contrário. A variável *dist*(**y**) é dada pela eq. 2.25, onde $\boldsymbol{\alpha}_0$ e b_0 correspondem aos valores ótimos para o vetor de multiplicadores de Lagrange e para a polarização, respectivamente. Embora a eq. 2.25 possa ser utilizada também para o caso linear, costuma-se reescrevê-la em função do vetor de pesos $\boldsymbol{\omega}_0$, obtendo-se:

$$dist(\mathbf{y}) = \boldsymbol{\omega}_0^T \mathbf{y} + b_0 \qquad \text{Eq. 3.17}$$

Multiplicando a eq.3.17 por d_y, que corresponde à etiqueta de classificação da observação **y**, e definindo a variável aleatória *z* como sendo igual a $d_y dist(\mathbf{y})$, pode-se afirmar que a probabilidade de acerto na classificação do vetor **y** é igual a probabilidade da variável *z* ser maior ou igual a zero, independentemente da classe a que **y** pertença.

No caso do UPSVM linear, a distância de uma observação ao hiperplano ótima pode ser obtida apenas em função do vetor $\boldsymbol{\omega}_0$, substituindo b_0, dado pela eq.3.8, na eq.3.17, obtendo-se:

$$dist(\mathbf{y}) = \boldsymbol{\omega}_0^T \mathbf{y} - \frac{\boldsymbol{\omega}_0^T \mathbf{X}\mathbf{e}}{N} \qquad \text{Eq. 3.18}$$

onde supõe-se que foi utilizado um número igual de observações de cada classe para a composição da matriz **X** usada no treinamento.

Em uma determinado aplicação, se um número grande de parâmetros, supostamente considerados variáveis aleatórias contínuas e independentes entre si, estiver sendo utilizado, pode-se assumir pelo Teorema Central do Limite (TCL) [17]

que a variável aleatória z condicionada a $\boldsymbol{\omega}_0$ e a classe do vetor de teste ($\mathbf{y}$) tem distribuição gaussiana. Por sua vez a probabilidade de acerto do classificador recém treinado independentemente da classe a que $\mathbf{y}$ pertence é dada por

$$P\left(z \geq 0 \middle/ \boldsymbol{\omega}_0 \right) = P(\mathbf{y} \in C_1) \frac{e^{-\left(z-\mu_{Z/\boldsymbol{\omega}_0,\mathbf{y}\in C_1}\right)^2 \Big/ 2\sigma^2_{Z/\boldsymbol{\omega}_0,\mathbf{y}\in C_1}}}{\sqrt{2\pi}\,\sigma_{Z/\boldsymbol{\omega}_0,\mathbf{y}\in C_1}} + P(\mathbf{y} \in C_2) \frac{e^{-\left(z-\mu_{Z/\boldsymbol{\omega}_0,\mathbf{y}\in C_2}\right)^2 \Big/ 2\sigma^2_{Z/\boldsymbol{\omega}_0,\mathbf{y}\in C_2}}}{\sqrt{2\pi}\,\sigma_{Z/\boldsymbol{\omega}_0,\mathbf{y}\in C_2}} \qquad \text{Eq. 3.19}$$

onde $\mu_{Z/\omega 0\,,\,\mathbf{y}\in \mathrm{Ci}}$ e $\sigma_{Z/\omega 0\,,\,\mathbf{y}\in \mathrm{Ci}}$ são a média e a variância da variável z condicionada a ω_0 e a classe (i=1 ou i =2) a que o vetor $\mathbf{y}$ pertence ($z/\boldsymbol{\omega}_0$, $\mathbf{y} \in C_i$).

Se as classes ocorrerem de forma equiprovável então a eq. 3.19 pode ser simplificada obtendo-se

$$P\left(z \geq 0 \middle/ \boldsymbol{\omega}_0 \right) = \frac{1}{2}\left(\frac{e^{-\left(z-\mu_{Z/\boldsymbol{\omega}_0,\mathbf{y}\in C_1}\right)^2 \Big/ 2\sigma^2_{Z/\boldsymbol{\omega}_0,\mathbf{y}\in C_1}}}{\sqrt{2\pi}\,\sigma_{Z/\boldsymbol{\omega}_0,\mathbf{y}\in C_1}} + \frac{e^{-\left(z-\mu_{Z/\boldsymbol{\omega}_0,\mathbf{y}\in C_2}\right)^2 \Big/ 2\sigma^2_{Z/\boldsymbol{\omega}_0,\mathbf{y}\in C_2}}}{\sqrt{2\pi}\,\sigma_{Z/\boldsymbol{\omega}_0,\mathbf{y}\in C_2}} \right) \qquad \text{Eq. 3.20}$$

Se as referidas variáveis aleatórias (parâmetros) forem também identicamente distribuídas, então Papoulis [17] afirma que o referido teorema pode ser aplicado para um número maior ou igual a 30 variáveis. Por último, se as funções densidade de probabilidade de cada variável aleatória também forem suaves, então a aproximação é válida para 5 ou mais variáveis.

Para o cálculo da média de $\mathbf{y}$ condicionada a $\boldsymbol{\omega}_0$ e à classe, supondo que $\mathbf{y} \in C_1$, tem-se:

$$\begin{aligned} \mu_{z/\boldsymbol{\omega}_0,\mathbf{y}\in C_1} &= E\left[\boldsymbol{\omega}_0{}^T \mathbf{y} - \frac{\boldsymbol{\omega}_0{}^T \mathbf{X}\mathbf{e}}{N} \right] = \\ &= \boldsymbol{\omega}_0{}^T \boldsymbol{\mu}_1 - \frac{1}{N}\boldsymbol{\omega}_0{}^T E\left[\sum_{i=1}^{N} \mathbf{x}_i \right] = \\ &= \boldsymbol{\omega}_0{}^T \boldsymbol{\mu}_1 - \frac{1}{N}\boldsymbol{\omega}_0{}^T \sum_{i=1}^{N} \mathbf{x}_i \end{aligned} \qquad \text{Eq. 3.21}$$

onde $\boldsymbol{\mu}_1$ é a média das observações pertencentes à classe C1. Analogamente para o caso em que y pertence à classe 2, tem-se:

$$\mu_{z/\boldsymbol{\omega}_0,\mathbf{y}\in C_2} = \frac{1}{N}\boldsymbol{\omega}_0{}^T \sum_{i=1}^{N} \mathbf{x}_i - \boldsymbol{\omega}_0{}^T \boldsymbol{\mu}_2 \qquad \text{Eq. 3.22}$$

onde $\boldsymbol{\mu}_2$ é a média das observações pertencentes à classe C2

Como geralmente a PDF de cada classe não é conhecida, e conseqüentemente as médias das classes também não são conhecidas então será necessário estimá-las pela média aritmética das observações utilizadas no treinamento. Neste caso, tem-se:

$$\hat{\mu}_{z/\boldsymbol{\omega}_0,\mathbf{y}\in C_1} = \frac{\boldsymbol{\omega}_0^T\left(\hat{\boldsymbol{\mu}}_1 - \hat{\boldsymbol{\mu}}_2\right)}{2} = \hat{\mu}_{z/\boldsymbol{\omega}_0,\mathbf{y}\in C_2} \qquad \text{Eq. 3.23}$$

onde $\boldsymbol{\mu}_1$ e são as médias aritméticas das observações utilizadas no treinamento para as classes C_1 e C_2.

Por sua vez, a variância de $\mathbf{y}$ condicionada a $\boldsymbol{\omega}_0$ e supondo que $\mathbf{y} \in C_1$ é dada por

$$\begin{aligned}
\sigma_{z/\boldsymbol{\omega}_0,\mathbf{y}\in C_1} &= E\left[\left(\boldsymbol{\omega}_0^T\mathbf{y} - \frac{\boldsymbol{\omega}_0^T\mathbf{X}\mathbf{e}}{N}\right)^2\right] - \mu^2_{z(\mathbf{y})/\boldsymbol{\omega}_0,\mathbf{y}\in C_1} = \\
&= \boldsymbol{\omega}_0^T\left(E\left[\mathbf{y}\mathbf{y}^T\right] - 2E\left[\frac{\mathbf{y}\mathbf{e}^T\mathbf{X}^T}{N}\right] + E\left[\frac{\mathbf{X}\mathbf{e}\mathbf{e}^T\mathbf{X}^T}{N^2}\right]\right)\boldsymbol{\omega}_0 - \mu^2_{z/\boldsymbol{\omega}_0,\mathbf{y}\in C_1} = \\
&= \boldsymbol{\omega}_0^T\left(\mathbf{R}_1 - \frac{2}{N}E\left[\mathbf{y}\sum_{i=1}^{N}\mathbf{x}_i^T\right] + \frac{1}{N^2}E\left[\sum_{i=1}^{N}\sum_{j=1}^{N}\mathbf{x}_i\mathbf{x}_j^T\right]\right)\boldsymbol{\omega}_0 - \mu^2_{z/\boldsymbol{\omega}_0,\mathbf{y}\in C_1} = \\
&= \boldsymbol{\omega}_0^T\left(\mathbf{R}_1 - \frac{2}{N}E\left[\mathbf{y}\sum_{i=1}^{N}\mathbf{x}_i^T\right] + \frac{1}{N^2}E\left[\sum_{i=1}^{N}\sum_{j=1}^{N}\mathbf{x}_i\mathbf{x}_j^T\right]\right)\boldsymbol{\omega}_0 - \mu^2_{z/\boldsymbol{\omega}_0,\mathbf{y}\in C_1} = \\
&= \boldsymbol{\omega}_0^T\left(\mathbf{R}_1 - \frac{2}{N}E[\mathbf{y}]\sum_{i=1}^{N}\mathbf{x}_i^T + \frac{1}{N^2}\sum_{i=1}^{N}\sum_{j=1}^{N}E\left[\mathbf{x}_i\mathbf{x}_j^T\right]\right)\boldsymbol{\omega}_0 - \mu^2_{z/\boldsymbol{\omega}_0,\mathbf{y}\in C_1} = \\
&= \boldsymbol{\omega}_0^T\left(\mathbf{R}_1 - \frac{2}{N}\mu_{z/\boldsymbol{\omega}_0,\mathbf{y}\in C_1}\sum_{i=1}^{N}\mathbf{x}_i^T + \frac{1}{N^2}\sum_{i=1}^{N}\sum_{j=1}^{N}\mathbf{x}_i\mathbf{x}_j^T\right)\boldsymbol{\omega}_0 - \mu^2_{z/\boldsymbol{\omega}_0,\mathbf{y}\in C_1} \qquad \text{Eq. 3.24}
\end{aligned}$$

onde $\mathbf{R}_1$ é a matriz de autocorrelação para a classe C1. Por sua vez, utilizando a eq. 3.21, obtém-se:

$$\mu^2_{z/\boldsymbol{\omega}_0,\mathbf{y}\in C_1} = \boldsymbol{\omega}_0^T\left(\mu_1\mu_1^T - \frac{2}{N}\mu_{z/\boldsymbol{\omega}_0,\mathbf{y}\in C_1}\sum_{i=1}^{N}\mathbf{x}_i^T + \frac{1}{N^2}\sum_{i=1}^{N}\sum_{j=1}^{N}\mathbf{x}_i\mathbf{x}_j^T\right)\boldsymbol{\omega}_0 \qquad \text{Eq. 3.25}$$

Aplicando a eq. 3.25 a eq. 3.24 obtém-se

$$\sigma_{z(\mathbf{y})/\boldsymbol{\omega}_0,\mathbf{y}\in C_1} = \boldsymbol{\omega}_0^T\mathbf{C}_1\boldsymbol{\omega}_0 \qquad \text{Eq. 3.26}$$

onde $\mathbf{C}_1$ é a matriz de covariância para a classe C1.

Procedendo de forma análoga, pode-se provar que

$$\sigma_{z(\mathbf{y})/\boldsymbol{\omega}_0,\mathbf{y}\in C_2} = \boldsymbol{\omega}_0^T\mathbf{C}_2\boldsymbol{\omega}_0 \qquad \text{Eq. 3.27}$$

onde $\mathbf{C_2}$ é a matriz de covariância para a classe C2. Novamente, se as PDFs de cada classe não forem conhecidas, deve-se estimar as matrizes de covariância de cada classe utilizando as observações disponíveis para treinamento.

Tendo em vista que as hipóteses necessárias para o uso do TCL não são sempre totalmente satisfeitas (grande número de parâmetros independentes), deve-se considerar que a eq. 3.19 (ou eq. 3.20) é apenas uma estimativa para a probabilidade de acerto do classificador UPSVM linear, onde a média e a variância da variável *z* condicionada a $\boldsymbol{\omega}_0$ e à classe a que **y** pertence são dadas respectivamente pelas eqs. 3.21, 3.22, 3.24 e 3.25. Resultados obtidos a partir de simulações computacionais são apresentados no próximo capítulo e corroboram a validade da estimativa obtida para a taxa de acerto do referido classificador, mesmo quando poucos parâmetros são utilizados.

4 ESTUDO DE CASOS

4.1 Introdução

Neste capítulo são apresentados alguns resultados obtidos a partir de dois casos de estudo hipotéticos e dois casos de estudo reais baseados em dados disponíveis pela Internet [11]. O primeiro caso de estudo refere-se a duas classes linearmente separáveis cujas observações pertencem a círculos. O segundo caso de estudo refere-se a duas classes com distribuições gaussianas e que não são linearmente separáveis. O terceiro caso refere-se à classificação de três diferentes tipos de plantas Íris. O último caso de estudo refere-se à classificação de tumor de mama, sendo composto por duas classes (benigno ou maligno). Estes dois últimos casos foram escolhidos porque são considerados casos clássicos da literatura sobre reconhecimento de padrões e são usualmente empregados na comparação entre algoritmos de classificação. Nos dois últimos casos, os dados estão disponíveis em [11].

Três classificadores são analisados: o classificador PSVM apresentado na seção 2.6, o classificador UPSVM proposto neste trabalho e apresentado nas seções 3.2 e 3.3, e o classificador proposto por Platt em [18] para treinamento de máquinas SVM do tipo L1SVM (ou HSVM sob algumas circunstâncias). Este último classificador denominado *Sequential Minimal Optimization* (SMO) tem sido muito empregado justamente por ser considerado um algoritmo de treinamento rápido para SVM. Sua principal característica é a minimização da função-objetivo utilizando apenas 2 observações de treinamento por iteração, sempre obedecendo às restrições impostas na formulação do problema dual correspondente ao classificador L1SVM. Essa otimização é feita analiticamente, o que lhe confere grande rapidez no processamento quando comparado a algoritmos tradicionais de otimização do tipo programação quadrática (QP).

A comparação entre os três classificadores leva em consideração a probabilidade de acerto estimada a partir da média da taxa de acerto obtida na classificação de observações de teste, o tempo gasto no treinamento, a complexidade computacional, o número de vetores de suporte e a variância dos parâmetros que determinam o hiperplano ótimo.

4.2 Classes linearmente separáveis

O primeiro caso refere-se à separação entre duas classes bidimensionais que são linearmente separáveis. A classe 1 é composta pelo conjunto de vetores $Z= \{z \in \Re^2 / z^T =[r\cos(\theta) + 2, r\text{sen}(\theta) + 2]\}$, onde r e θ são variáveis aleatórias, com funções densidade de probabilidade (PDFs) uniformes nos intervalos [0 0,5] e [0 2π], respectivamente. Já a classe 2 é composta pelo conjunto de vetores $z \in \Re^2$, tal que $z^T =[r\cos(\theta), r\text{sen}(\theta)]$, onde r e θ são variáveis aleatórias, com PDFs uniformes nos intervalos [0 1] e [0 2π], respectivamente. Ou seja, os elementos da classe 1 estão contidos em um círculo de raio 0.5 e centro $[2\ 2]^T$, enquanto que os elementos da classe 2 estão contidos em um círculo de raio unitário e centro na origem.

Para cada método analisado (UPSVM, PSVM e SMO) e para cada valor do parâmetro C, foram realizadas 100 iterações de treinamento, cada iteração utilizando 100 observações por classe. Inicialmente, foram determinados o tempo médio gasto no treinamento e a variância dos coeficientes dos hiperplanos ótimos obtidos que, neste caso de estudo, são retas. Posteriormente, para determinar a média da probabilidade de acerto, cada classificador foi avaliado usando um conjunto de teste composto por 500 observações por classe.

A tabela 4.1 mostra a média aritmética da taxa de acerto obtida por cada um dos classificadores para diferentes valores de C. É importante destacar que a estimativa da probabilidade de acerto calculada a partir das eq. 3.19 resultou em aproximadamente 1 para todos os valores de C, estando de acordo com os valores medidos e apresentados na tabela 4.1 para o UPSVM linear. Também foi observado que à medida que o valor de C diminui, a média e a variância da variável *z* condicionada a $\boldsymbol{\omega_0}$, calculadas respectivamente pelas eqs. 3.22 e 3.23, também diminuem, porém sem alterar o valor final obtido para a probabilidade de acerto estimada.

Tanto o classificador SMO como o classificador UPSVM conseguiram classificar corretamente todas as observações de teste, independentemente do parâmetro C. Já o classificador PSVM apresenta uma grande taxa de erro para valores pequenos de C porque o referido método de treinamento dá grande ênfase a minimização conjunta

do módulo de *b* e do módulo de **ω**, gerando um valor muito pequeno para *b*. Foi verificado que o problema persistiu mesmo quando o classificador PSVM foi treinado com um número maior de observações por classe (200 a 500). É importante destacar que, teoricamente, o hiperplano ótimo pode ser obtido para C=0, pois o caso de estudo em questão trata de classes linearmente separáveis e o método tradicional originalmente proposto para resolvê-lo é o HSVM.

Tabela 4.1: Média aritmética da taxa de acerto (%).

C / Método	10^{-4}	10^{-3}	10^{-2}	10^{-1}	1	10	100
UPSVM	100	100	100	100	100	100	100
PSVM	77,53	86,82	99,36	100	100	100	100
SMO	100	100	100	100	100	100	100

A tabela 4.2 mostra a média aritmética do tempo gasto, em milisegundos, no treinamento. O método UPSVM proposto neste trabalho obteve os menores tempos de treinamento, tempos significativamente menores do que os obtidos pelo método SVM tradicional treinado pelo SMO. Este resultado já era esperado porque a complexidade computacional da técnica SMO foi determinada empiricamente em [18] como sendo kN^{γ}, onde k é uma constante, γ é, no melhor caso, igual a dois e N é o número de observações envolvidas. Ou seja, a complexidade computacional da técnica SMO é maior do que a obtida pelo método proposto neste trabalho.

Tabela 4.2: Média dos tempos de treinamento em ms

C / Método	10^{-4}	10^{-3}	10^{-2}	10^{-1}	1	10	100
UPSVM	<1	<1	<1	<1	<1	<1	<1
PSVM	≅1	≅1	≅1	≅1	≅1	≅1	≅1
SMO	273,1	272,8	311,2	230,6	140,2	136,4	148,8

OBS: Resultados obtidos com um PC AMD SEMPRO 2600+ , 512 MB.

No que se refere ao treinamento SMO, também foi observado que o vetor de coeficientes de Lagrange saturou em seu valor máximo (C) para valores de C menores ou iguais a 0,1. Isto equivale a dizer que todas as observações de treinamento encontram-se dentro da margem de separação e por isso são vetores de suporte. No

entanto, uma vez que todas as observações de treinamento foram corretamente classificadas, estas encontram-se do lado correto da reta que caracteriza o classificador. Por sua vez, não houve saturação de nenhum multiplicador de Lagrange para valores de C maiores ou iguais a 10. Neste caso, cada classificador obtido pelo método SMO é do tipo HSVM, tendo em vista que nenhuma amostra do treinamento foi encontrada dentro da margem de separação. Por último, para C=1, 20% dos classificadores obtidos pelo método SMO são do tipo L1SVM e o restante, do tipo HSVM.

Do exposto anteriormente, conclui-se que na separação de classes linearmente separáveis, o número de vetores de suporte tende a diminuir até encontrar seu limite mínimo (que depende da aplicação) a medida que o parâmetro C aumenta. Isto pode ser visto na tabela 4.3 que apresenta o número de vetores de suporte para o classificador SMO.

Tabela 4.3: Número de vetores de suporte obtido pelo treinamento SMO

C	**10^{-4}**	**10^{-3}**	**10^{-2}**	**10^{-1}**	**1**	**10**	**100**
Número de VS	200	200	66 a 74	12 a 17	2 a 4	2 a 3	2 a 3

Ao analisar o laço principal do algoritmo SMO [18], constata-se que, na maioria das iterações, apenas os candidatos a vetores de suporte são levados em consideração para a minimização da função-objetivo. Portanto, espera-se que o tempo de treinamento SMO dependa proporcionalmente do número de vetores de suporte encontrados. Isto explica porque o tempo de treinamento para o classificador SMO, apresentado na tabela 4.2, é menor para C>1 e maior para C<0,01. Porém, não se deve esquecer que cada multiplicador de Lagrange está limitado pelo parâmetro de regularização C, isto implica em dizer que a busca pelo vetor de multiplicadores de Lagrange ótimo ocorrerá em um espaço cujo hipervolume aumenta à medida que C aumenta, e conseqüentemente, o tempo de treinamento também pode aumentar à medida que C aumenta.

Em resumo, há dois fatores conflitantes que influenciam o tempo de treinamento SMO: o número de vetores de suporte, que diminui quando C aumenta, e o hipervolume

do espaço a que o vetor de multiplicadores de Lagrange pertence, que aumenta à medida que C cresce.

Outro ponto importante que merece destaque é a variância dos parâmetros estimados do hiperplano obtido, pois uma menor variância desses parâmetros significa uma maior probabilidade de se obter um hiperplano próximo ao hiperplano ótimo teórico a partir de um único treinamento. Neste caso de estudo, o hiperplano obtido após cada treinamento é uma reta que pode ser representada pela equação $y=m_1x+m_2$, onde m_1 e m_2 são os coeficientes da reta. A correspondência entre os coeficientes m_1 e m_2 e o vetor de peso $\boldsymbol{\omega}$ e a polarização b é dada por:

$$m_1 = -\frac{\omega_2}{\omega_1} \qquad \text{Eq. 3.22}$$

$$m_2 = -\frac{b}{\omega_1} \qquad \text{Eq. 3.23}$$

A tabela 4.4 mostra a variância dos coeficientes m_1 (1ª linha) e m_2 (2ª linha). Foi observado que a variância dos parâmetros de interesse cresce à medida que C cresce. Portanto é interessante trabalhar com valores pequenos de C e neste caso o método UPSVM é mais vantajoso que o método PSVM, pois este último obtém uma versão polarizada do parâmetro b, o que implica, como já foi visto anteriormente, em uma baixa taxa de acertos. A polarização do parâmetro b explica a drástica redução de variância obtida pelo PSVM para o coeficiente m_2 para C≤0.001.

Tabela 4.4: Variância dos coeficiente m_1 e m_2 (x10^{-4}).

C / Método	10^{-4}	10^{-3}	10^{-2}	10^{-1}	1	10	100
UPSVM	8	12	18	149	222	384	231
	21	28	28	163	237	423	254
PSVM	12	25	129	131	277	194	255
	<1	1	58	133	273	207	265
SMO	9	9	30	76	285	309	359
	74	60	53	278	425	451	537

O classificador UPSVM também é mais vantajoso que o SMO para valores pequenos de C, porque este último "satura", ou seja, todos os dados de treinamento são

ponderados igualmente por C, o que equivale a dizer que nenhum treinamento era necessário. Tal fato pode vir a degradar o desempenho do classificador SMO.

4.3 Classes não separáveis linearmente

O caso de estudo apresentado nesta subseção também foi utilizado por Haykin em [3] para avaliação do desempenho do classificador SVM tradicional na separação de duas classes bidimensionais não linearmente separáveis. A primeira classe é um processo estocástico independente e igualmente distribuído (i.i.d.) [17] com distribuição gaussiana, vetor média $\boldsymbol{\mu}_1^T = [0,0]$ e matriz de covariância $\mathbf{C}_1 = \mathbf{I}$, onde $\mathbf{I}$ é a matriz de identidade. Por sua vez, a segunda classe também é um processo estocástico i.i.d. com distribuição gaussiana, vetor média $\boldsymbol{\mu}_2^T = [2,0]$ e matriz de covariância $\mathbf{C}_2 = 4\mathbf{I}$.

Sabe-se que neste caso o estimador de Bayes [3,17] é ótimo e consiste em comparar a observação analisada com a circunferência de raio $r \cong 2{,}34$ e centro $\mathbf{x}_C^T=[-2/3 \ , \ 0]$. Se a observação estiver dentro da circunferência então é classificado como pertencendo a classe 2, senão é classificado como pertencendo a classe 1. A probabilidade de classificação correta é $p_C \cong 81{,}51\%$. A aplicação de um *kernel* linear à técnica SVM resulta em uma reta como função de decisão, obviamente diferente da curva de decisão ótima obtida pelo estimador de Bayes. Portanto, um *kernel* não linear deve ser empregado. Haykin [3] empregou um *kernel* do tipo RBF (c.f. seção 2.4), com variância $\sigma_{RBF}=2$, tanto na fase de treinamento, como na fase de testes, configuração que também foi adotada neste trabalho.

Cinco classificadores, por método e para cada valor de C, foram treinados utilizando 250 observações por classe em cada fase de treinamento. Já na fase de testes, cada classificador foi treinado utilizando 32000 observações. Essa configuração para as fases de treinamento e teste também foi usada em [3], porém, apenas para avaliar o classificador SVM tradicional no que se refere à probabilidade de acerto para C= 0,1. A tabela 4.5 mostra a média da taxa de acerto obtida por cada método e para cada valor de C empregado no treinamento.

Os resultados obtidos encontram-se muito próximos da probabilidade de acerto do classificador ótimo, exceto para o classificador PSVM quando treinados com valores de $C \leq 0{,}001$. É importante destacar que geralmente as PDFs associadas a cada classe não são conhecidas e por isso não se pode usar o classificador de Bayes. Também é importante destacar que valores maiores do que a probabilidade de acerto máxima teórica podem ser encontrados devido a erros de estimação. Analisando os resultados apresentados na tabela 4.5, pode-se perceber uma característica interessante do método UPSVM: a sua estabilidade de desempenho com relação ao parâmetro C. Isso se deve aos fatores listados abaixo:

- Para valores pequenos de C (neste caso, para $C \leq 0.01$), o vetor de multiplicadores de Lagrange obtido pelo treinamento SMO satura, degradando o seu desempenho. Para valores elevados de C, um número pequeno de observações (vetores de suporte) são utilizadas para a determinação do hiperplano ótimo. No entanto, os métodos baseados em hiperplanos proximais (PSVM e UPSVM) usam todas as observações disponíveis no treinamento para a determinação do hiperplano ótimo. Isto acaba por ter impacto positivo na estimação dos seus parâmetros, e conseqüentemente, na média da taxa de acerto.
- Para valores pequenos de C, o treinamento PSVM dá grande ênfase na minimização conjunta do módulo de $\boldsymbol{\omega}$ e de b. Como a margem de separação entre as classes só depende do módulo do primeiro, o classificador obtido é polarizado, tendo em vista que b é menor do que o desejável.

Tabela 4.5: Média aritmética da taxa de acerto (%).

C / Método	10^{-4}	10^{-3}	10^{-2}	10^{-1}	1	10	100
BAYES	81,51						
UPSVM	81,23	81,25	81,42	81,56	81,50	81,34	81,14
PSVM	77,84	80,00	81,37	81,50	81,49	81,45	81,33
SMO	80,57	80,57	80,58	81,52	81,42	81,27	80,96

Com relação ao tempo de treinamento, a tabela 4.6 mostra a média aritmética dos tempos obtidos em segundos, durante a fase de treinamento. Para a medida deste

parâmetro, não se levou em consideração o tempo gasto para a avaliação do *kernel*, já que esse tempo independe do método usado.

Constata-se novamente que o algoritmo de treinamento UPSVM é mais rápido do que os demais métodos, sendo 7,5 vezes mais rápido do que o treinamento PSVM e de 22 vezes a muitas ordens de grandeza mais rápido do que o treinamento SMO, dependendo do valor de C. Constata-se também um aumento exponencial do tempo de treinamento SMO para C≥1, diferente do que ocorreu no caso de estudo anterior (tabela 4.2). Esse aumento significativo deve-se exclusivamente ao maior hipervolume do espaço de busca para os multiplicadores de Lagrange, tendo em vista que o número de vetores de suporte, apresentados na tabela 4.7, não sofreu alterações significativas para esses valores de C.

Tabela 4.6: Média dos tempos de treinamento em s.

C / Método	10^{-4}	10^{-3}	10^{-2}	10^{-1}	1	10	100
UPSVM	0,4	0,4	0,4	0,4	0,4	0,4	0,4
PSVM	3,0	3,0	3,0	3,0	3,0	3,0	3,0
SMO	12,1	9,8	12,2	9	13,2	202,3	3.571,1

OBS: Resultados obtidos em um PC AMD SEMPRO 2600+ , 512 MB.

Tabela 4.7: Número de vetores de suporte obtido pelo treinamento SMO

C	10^{-4}	10^{-3}	10^{-2}	10^{-1}	1	10	100
Número de VS	500	500	500	278 a 302	215 a 246	200 a 227	191 a 223

A única desvantagem observada para o método UPSVM não linear (também compartilhada pelo PSVM não linear) está no fato de que serão necessários todos as observações de treinamento para construir o classificador. Isto implica em um maior número de avaliações de *kernel* e, portanto, em uma maior complexidade computacional do classificador na fase de teste (ou generalização) se comparado à obtida pelo SMO. Vale lembrar que na técnica SVM tradicional, o classificador é definido por um subconjunto das observações de treinamento, os chamados vetores de suporte. Na aplicação em questão, isto ocorre para C≥0,1, conforme ilustrado pela tabela 4.7 que mostra o número de vetores de suporte obtidos pelo SMO.

A tabela 4.8 apresenta o número de operações em ponto flutuante (flops) normalizado pelo menor valor obtido entre todos os métodos usados e entre todos os valores de C empregados. O treinamento SMO obteve menor complexidade computacional que os demais métodos para C≤1 , no entanto o tempo de treinamento não é uma função do número de flops unicamente, ele também leva em conta o número de acessos à memória. Isto explica porque o UPSVM é mais rápido do que o SMO, mesmo apresentando, em alguns casos, maior complexidade computacional.

Além disso, observa-se que o número de flops cresceu exponencialmente com C devido ao aumento do hipervolume do espaço de busca. Isto sugere trabalhar com pequenos valores de C, porém o classificador SMO teve seu desempenho diminuído devido à saturação dos multiplicadores de Lagrange para C≤0,01. O classificador UPSVM foi novamente o único que manteve seu bom desempenho para pequenos valores de C.

Tabela 4.8: Número de flops normalizado

C / Método	10^{-4}	10^{-3}	10^{-2}	10^{-1}	1	10	100
UPSVM	5,8	5,8	5,8	5,8	5,8	5,8	5,8
PSVM	82,6	82,6	82,6	82,6	82,6	82,6	82,6
SMO	1,0	1	1,0	1,7	5,4	138,3	1753,8

4.4 Dados reais – Planta Íris

O conjunto de dados analisado neste caso de estudo clássico da literatura de Reconhecimento de Padrões refere-se a 3 classes, cada uma correspondendo a um diferente tipo de planta Íris (Setosa, Versicolor e Virginica). O referido conjunto de dados é composto por 50 observações por classe e 4 atributos por observação. A classe Setosa é linearmente separável das outras duas, enquanto que as classes Versicolor e Virginica não são linearmente separáveis. Neste estudo de caso dois tipos de problemas são analisados. Inicialmente, deseja-se separar as classes Setosa e Versicolor usando um *kernel* linear. O segundo problema consiste na separação das classes Virginica e Versicolor usando uma Função de Base Radial (RBF) como *kernel*.

Neste ponto, cabe ressaltar que apenas os dois problemas citados acima foram analisados porque o objetivo deste capítulo é a comparação entre o algoritmo proposto e os algoritmos SVM estudados, que são por natureza classificadores binários. Além disso os dois problemas citados englobam o uso de classificadores lineares e não lineares, sendo redundante e desnecessário separar as classes Setosa e Virginica. Também é importante ressaltar que não há interesse em obter um único classificador capaz de separar as três classes e por isso nenhum método de generalização de classes foi empregado. A comparação entre diferentes técnicas de generalização para mais de duas classes foi objeto de outros estudos [6,7,8] e não faz parte do escopo deste trabalho.

Nos dois problemas, foram executados 100 iterações de treinamento. Em cada treinamento, metade das observações do banco de dados foi escolhida aleatoriamente e utilizada, enquanto que a outra metade foi usada em testes visando a avaliação de desempenho de cada algoritmo de treinamento.

a) Setosa x Versicolor

A tabela 4.9 mostra a média aritmética da taxa de acerto obtida por cada um dos classificadores para diferentes valores de C. Novamente, a estimativa da probabilidade de acerto calculada a partir da eq. 3.19 resultou em aproximadamente 1 para todos os valores de C, estando de acordo com os valores medidos. Novamente, pôde-se notar que à medida que o valor de C diminuiu, a média e a variância da variável *z* condicionada a ω_0, calculadas respectivamente pelas eqs. 3.22 e 3.23, também diminuíram, porém sem alterar o valor final obtido para a probabilidade de acerto estimada.

Tabela 4.9: Média aritmética da taxa de acerto (%).

C / Método	10^{-4}	10^{-3}	10^{-2}	10^{-1}	1	10	100
UPSVM	100	100	100	100	100	100	100
PSVM	50	94,76	99,88	100	100	100	100
SMO	99,64	99,32	100	100	100	100	100

A tabela 4.10 mostra o número de operações em ponto flutuante normalizado. Como se pode observar, a complexidade computacional do UPSVM foi 5 a 9 vezes menor que a obtida pelo SMO, enquanto que a probabilidade de acerto foi igual ou ligeiramente melhor. Cabe ressaltar também que o tempo gasto pelo UPSVM para treinamento foi em média igual a 0,01s, por sua vez o tempo médio gasto pelo SMO foi de 2s (utilizando um PC AMD K6III 400MHZ, 64MB RAM). Em outras palavras, a transferência de dados no SMO mostrou-se novamente um fator mais crítico do que o número de operações em ponto flutuante neste quesito de desempenho.

Tabela 4.10: Número de flops normalizado

Método \ C	10^{-4}	10^{-3}	10^{-2}	10^{-1}	1	10	100
UPSVM	1	1	1	1	1	1	1
PSVM	1,13	1,13	1,13	1,13	1,13	1,13	1,13
SMO	5,68	5,85	8,97	6,58	5,51	5,43	4,94

Também pode-se observar que o número de flops para o SMO é variável, fato já esperado tendo em vista que a complexidade computacional do SMO não é determinística e pode variar ainda que as mesmas observações sejam utilizadas em dois treinamentos. Por sua vez, o número de flops necessários para o treinamento UPSVM (ou PSVM) é determinístico e independente de C.

Com relação à comparação entre UPSVM e PSVM, pode-se afirmar que o primeiro tem menor complexidade computacional (neste caso, aproximadamente 12% inferior) e que o método PSVM não foi capaz de separar as duas classes para valores de C menores do que 0,1, devido a polarização do hiperplano ótimo estimado. Em particular, para $C = 10^{-4}$, o método UPSVM foi capaz de corretamente separar as duas classes, enquanto que o SMO obteve pouquíssimos erros e o PSVM classificou todas as observações como pertencendo a classe Setosa.

As variâncias dos parâmetros estimados ($\boldsymbol{\omega}_0$ e b_0), mostradas na tabela 4.11, consistem em outro importante fator de comparação, pois uma grande variância significa uma menor probabilidade de se obter um hiperplano ótimo próximo ao hiperplano ótimo teórico, supondo que um único treinamento foi realizado. As três

primeiras linhas da tabela 4.11, para cada método de treinamento, mostram a variância dos três últimos elementos do vetor $\boldsymbol{\omega_0}$ normalizados pelo primeiro elemento, e a quarta linha mostra a variância do parâmetro b_0 também normalizado pelo primeiro elemento de $\boldsymbol{\omega_0}$.

Observa-se claramente que a variância dos parâmetros estimados cresce à medida que C aumenta, independentemente do método empregado. Este resultado sugere que é interessante trabalhar com o menor valor de C possível, ou seja, se a probabilidade de acerto para diferentes valores de C não varia muito para uma determinada aplicação, então use o menor valor de C. Neste caso o método UPSVM é o único que conseguiu 100% de acerto.

Tabela 4.11: Variância dos parâmetros estimados.(x 10^{-4})

Método \ C	10^{-4}	10^{-3}	10^{-2}	10^{-1}	1	10	100
UPSVM	48	48	46	89	176	175	168
	57	58	62	130	535	618	603
	27	27	37	269	1920	2707	2740
	4226	4227	4492	8637	8955	2740	5279
PSVM	36	9	6	24	154	171	168
	51	37	35	104	460	541	590
	27	31	37	158	1256	2260	2668
	<1	1	4	18	382	2752	4808
SMO	49	49	41	201	1054	1052	1052
	57	57	65	330	1045	1042	1041
	26	26	41	573	3228	3219	3214
	3984	5301	3419	12271	44422	44524	44518

Além disso, também é importante destacar que os resultados apresentados na tabela 4.11 mostram que o método SMO apresentou variâncias mais elevadas que os outros dois métodos para $C \geq 0{,}1$, e que os métodos UPSVM e PSVM apresentaram valores similares de variância, com exceção do parâmetro b_0 Tal fato pode ser explicado pelo número de observações de treinamento que contribuem efetivamente para a estimação do hiperplano ótimo. No caso dos métodos baseados em hiperplanos

proximais, todas as observações são utilizadas, enquanto que para os métodos baseados em hiperplanos de suporte, apenas os vetores de suporte são usados.

No caso em questão, o número de vetores de suporte encontrados após o treinamento SMO, e efetivamente utilizados para a determinação de $\boldsymbol{\omega_0}$ e b_0, é mostrado na tabela 4.12. É interessante ressaltar também que o vetor de multiplicadores de Lagrange saturou para C≤0.001 e que o treinamento SMO gerou classificadores do tipo HSVM para C≥1.

Tabela 4.12: Número de vetores de suporte obtido pelo treinamento SMO

C	**10^{-4}**	**10^{-3}**	**10^{-2}**	**10^{-1}**	**1**	**10**	**100**
Número de VS	50	50	36 a 41	4 a 8	2 a 4	2 a 5	2 a 5

b) Virginica x Versicolor

Para a separação das classes Virginica e Versicolor, um *kernel* RBF foi usado. A variância do *kernel* RBF foi variada entre diversos valores, obtendo-se a maior probabilidade de acerto quando esta foi igual a 10, valor usado neste trabalho. A taxa de acerto na classificação de observações de teste e a taxa de acerto na classificação de observações de treinamento são apresentadas nas tabelas 4.13 e 4.14, respectivamente. O tempo gasto para o cálculo do *kernel* não foi calculado, pois é o mesmo para todos os métodos analisados.

Os resultados obtidos mostram que o UPSVM foi melhor do que os outros dois métodos em todos os quesitos comparados e para qualquer valor de C escolhido. Em particular, o PSVM apresentou novamente uma significativa redução na probabilidade de acerto quando valores pequenos de C foram utilizados no treinamento. Além disso, pode-se afirmar que para o SMO e o PSVM treinados com valores elevados de C ocorreu fato similar ao efeito de *overtraining* em redes neurais treinadas tradicionalmente.

Ou seja, comparando as probabilidades de acerto na classificação das observações de teste, apresentadas na tabela 4.13, com as probabilidades de acerto na classificação das observações utilizadas no treinamento, apresentadas na tabela 4.14,

percebe-se que a melhora de desempenho na classificação de observações no treinamento para valores elevados de C implicou na diminuição do desempenho na classificação de observações de teste para a mesma faixa de valores.

Tabela 4.13: Média aritmética da taxa de acerto obtida no teste.

Método \ C	10^{-4}	10^{-3}	10^{-2}	10^{-1}	1	10	100
UPSVM	88,16	88,20	88,36	89,29	93,08	96,96	96,92
PSVM	61,40	79,28	87,12	86,96	87,92	87,36	87,44
SMO	84,96	86,36	85,56	86,68	87,88	86,38	85,68

Tabela 4.14: Média aritmética da taxa de acerto obtida no treinamento.

Método \ C	10^{-4}	10^{-3}	10^{-2}	10^{-1}	1	10	100
UPSVM	90,16%	90,20%	90,24%	91,16%	95,08%	97,52%	97,88%
PSVM	67,52%	81,44%	85,08%	86,80%	90,76%	96,48%	97,52%
SMO	85,80%	87,12%	87,08%	87,36%	94,32%	97,44%	97,36%

A degradação de desempenho na utilização do classificador SMO ocorreu porque foi dada uma ênfase muito grande à minimização do erro de treinamento em detrimento a maximização da margem de separação entre as classes (C grande), e porque o hiperplano ótimo é obtido em função de apenas algumas poucas observações de treinamento, logo a estimação dos parâmetros ótimos é severamente prejudicada quando um (ou mais) *outlier* ocorre dentro desse conjunto de vetores. Por *outlier* entende-se uma observação que não é típica da classe em questão e que pode ocorrer porém com pequena probabilidade. Já o UPSVM utiliza todas as observações de treinamento disponíveis para a determinação do hiperplano ótimo, sendo portanto menos suscetível à presença de *outliers*.

Por sua vez, a degradação no desempenho do classificador PSVM para valores elevados de C ocorreu porque a generalização deste classificador para o caso não linear foi realizada de forma equivocada pela substituição da matriz **X** por **K**, e conseqüentemente, $\mathbf{X}^T\mathbf{X}$ por $\mathbf{K}^T\mathbf{K}$ (seção 2.6), ao invés de simplesmente **K** como realizado nos classificadores SMO e UPSVM.

Em resumo, o método UPSVM foi o único que não apresentou restrições de uso para nenhuma faixa específica de valores para C, embora fique claro neste caso que o classificador UPSVM treinado com C= 10 é o mais indicado para uso, pois foi o que obteve a maior probabilidade de acerto na classificação de observações de teste.

No que se refere ao tempo de treinamento e ao número de flops normalizado, mostrados respectivamente nas tabelas 4.15 e 4.16, o UPSVM continuou sendo o mais rápido dos algoritmos analisados, embora o número de flops tenha sido quase sete vezes maior que o calculado para o SMO. Novamente se conclui que o tempo gasto em transferência de dados é um fator crítico de desempenho. O tempo gasto para o cálculo de um *kernel* não foi levado em consideração para o cálculo do tempo de treinamento pois é o mesmo para todos os métodos analisados.

Como no caso de estudo anterior, a única desvantagem observada para o método UPSVM não linear (também compartilhada pelo PSVM não linear) está no fato de que serão necessárias todas as observações de treinamento para construir o classificador. Isto implica em um maior número de avaliações de *kernel* e, portanto, em uma maior complexidade computacional do classificador na fase de teste se comparado à obtida pelo SMO. Na aplicação em questão, isto ocorre para C≥1, conforme ilustrado pela tabela 4.17 que mostra o número de vetores de suporte obtidos pelo SMO.

Tabela 4.15: Média dos tempos de treinamento em ms

C / Método	10^{-4}	10^{-3}	10^{-2}	10^{-1}	1	10	100
UPSVM	5,2	6,8	6,2	4,4	6,2	6,8	6,6
PSVM	6,2	5,6	6,6	6,6	6,4	6,8	6,4
SMO	266,2	230,7	270,2	267,2	409,1	319,8	470,7

OBS: Resultados obtidos com um PC AMD K6 III 400MHz, 64MB RAM

Tabela 4.16: Número de flops normalizado

C / Método	10^{-4}	10^{-3}	10^{-2}	10^{-1}	1	10	100
UPSVM	6,73	6,73	6,73	6,73	6,73	6,73	6,73
PSVM	21,26	21,26	21,26	21,26	21,26	21,26	21,26
SMO	1,02	1	1,12	1,11	1,65	2,03	4,03

Tabela 4.17: Número de vetores de suporte obtido pelo treinamento SMO

C	10^{-4}	10^{-3}	10^{-2}	10^{-1}	1	10	100
Número de VS	50	50	50	50	32 a 38	14 a 21	5 a 12

4.5 Dados reais – Tumor de Mama

Neste estudo de caso, criado pelo Dr. William H. Wolberg (University of Wisconsin Hospitals) e disponível em [11], existem duas classes: a primeira associada ao tumor benigno e a segunda associada ao tumor maligno. O referido banco de dados contém 699 observações (458 da classe benigna e 241 da classe maligna), onde cada observação é composta por 10 atributos. Foram realizados 100 treinamentos. Em cada treinamento um subconjunto de 400 amostras (200 observações por classe) foi aleatoriamente escolhido e o *kernel* linear foi usado. As demais observações foram utilizadas no teste de cada classificador obtido.

A taxa de acerto na classificação das observações de teste é mostrada na tabela 4.18. A estimativa obtida para a probabilidade de acerto do UPSVM linear para este caso de estudo é apresentada na tabela 4.19. Os valores estimados são ligeiramente maiores do que os obtidos e quase não variam em relação ao parâmetro C. Portanto, pode-se concluir que a estimativa teórica para a probabilidade de acerto dada pela eq.3.19 é válida e corrobora a idéia de que o UPSVM é um classificador robusto com relação à escolha do parâmetro C.

Tabela 4.18: Média aritmética da taxa de acerto

C / Método	10^{-4}	10^{-3}	10^{-2}	10^{-1}	1	10	100
UPSVM	94,92	96,40	96,42	96,40	96,40	96,40	96,40
PSVM	14,05	81,56	95,29	96,42	96,39	96,40	96,40
SMO	94,47	95,95	96,51	96,12	95,53	94,18	90,88

Da tabela 4.18, percebe-se também que a probabilidade de acerto do UPSVM é semelhante à obtida pelo SMO para os mesmo valores de C, exceto para C≥1, quando a

probabilidade de acerto obtida pelo SMO começa a cair, enquanto que para o UPSVM, ela se mantém constante e igual ao seu valor máximo. A explicação para a degradação de desempenho do SMO é a presença de "outliers" entre os vetores de suporte, comprometendo o desempenho quando se é dada ênfase à minimização do erro de treinamento.

Tabela 4.19: Estimativa para a probabilidade de acerto

C / Método	10^{-4}	10^{-3}	10^{-2}	10^{-1}	1	10	100
UPSVM	99,42	99,76	99,79	99,80	99,80	99,79	99,79

O UPSVM também foi melhor do que o PSVM no que se refere à probabilidade de acerto, pois obteve bons resultados mesmo para valores pequenos de C ($C<0.01$). Em particular, para $C=10^{-4}$, o PSVM classificou corretamente, em média, apenas 1 observação da classe tumor benigno, sendo as demais 257 observações classificadas erroneamente como sendo da classe tumor maligno. Ainda para este valor de C, todas as observações da classe tumor benigno foram classificadas como tumor maligno em 73 dos 100 testes realizados. Ao se calcular a média dos parâmetros por classe, percebe-se que a média dos parâmetros associados a observações do tumor maligno está "acima" da tumor benigno, portanto, a polarização do hiperplano ótimo, característica típica do classificador PSVM, foi o principal motivo para a diminuição da taxa de acerto, que neste caso de estudo, ocorreu de forma abrupta.

O tempo gasto no treinamento e o número de flops normalizado foram calculados e são mostrados nas tabelas 4.20 e 4.21. Nota-se que o UPSVM foi 3 a 6 ordens de magnitude mais rápido que o SMO e um pouco mais rápido que o PSVM. É importante destacar que para o SMO tanto o tempo de treinamento como a complexidade computacional cresceram exponencialmente à medida que o valor de C aumentou. O mesmo já havia ocorrido com o SMO no primeiro caso de estudo a utilizar um *kernel* não linear.

No que se refere à variância dos parâmetros estimados, ela cresce à proporção que C aumenta, confirmando resultados anteriores e novamente sugerindo que deve-se dar preferência à utilização do menor valor de C possível.

Tabela 4.20: Média dos tempos de treinamento em ms

C / Método	10^{-4}	10^{-3}	10^{-2}	10^{-1}	1	10	100
UPSVM	2,2	2,1	2,7	3,1	2,9	2,3	2,9
PSVM	4,8	4,2	3,8	3,5	3,6	3,4	3,8
SMO	6820	3801	8817	62513	$>4x10^5$	$>10^6$	$>2x10^6$

OBS: Resultados obtidos com um PC AMD K6 III 400MHz, 64MB RAM

Tabela 4.21: Número de flops normalizado

C / Método	10^{-4}	10^{-3}	10^{-2}	10^{-1}	1	10	100
UPSVM	1	1	1	1	1	1	1
PSVM	1,06	1,06	1,06	1,06	1,06	1,06	1,06
SMO	12	13	22	101	722	2499	4250

5. O CLASSIFICADOR DE MODULAÇÃO CMADB

5.1. Introdução

Um classificador de modulação é um equipamento que reconhece automaticamente o tipo de modulação que foi empregada para atrelar o sinal de mensagem m(t) à portadora $s_C(t)$ e dar origem ao sinal modulado s(t). A principal motivação para o estudo deste tipo de classificador é o fato de ele ser muito empregado nas áreas de controle e vigilância do espectro eletromagnético, seja para aplicações civis ou militares. Outra aplicação de interesse para um classificador de modulação é a sua implementação em rádios definidos por *software* (RDS). Muitos métodos foram propostos na literatura [2,19–32] para a implementação deste classificador. Basicamente existem duas abordagens principais: técnicas baseadas em Reconhecimento de Padrões (RP) [2,19–28] e técnicas baseadas no uso de detectores de Máxima Verossimilhança (ML – do termo em Inglês *Maximum Likelihood*) [29-32].

Na abordagem RP, a tarefa de classificação é precedida pela extração de características relevantes (para o processo de classificação) do sinal de comunicações. As características relevantes são geralmente obtidas a partir da amplitude (A_{INST}), freqüência (F_{INST}) e fase (φ_{INST}) instantâneas do sinal. Em [19–21] foram utilizados histogramas que representam estimativas para a função densidade de probabilidade (PDF) de um ou mais parâmetros entre os que se seguem: A_{INST} [19,20], F_{INST} [19–21], taxa de cruzamento de zeros [19], fase zero [21] e vetor de dados [20]. Já em [2,22–23] a grande maioria das características relevantes utilizadas foram as estimativas para média e/ou variância de A_{INST} [2,22,23], F_{INST} [2,22–25], e φ_{INST} [2,23]. Por último, momentos de ordem elevada foram usados em [26,27], e autovalores e autovetores obtidos a partir da decomposição de uma matriz de dados que representa o sinal foram usados em [28].

Após a fase de extração das características relevantes, a distância entre elas e cada padrão que representa uma modulação conhecida é calculada. O padrão mais próximo determina a modulação escolhida como resultado do processo de classificação. A definição de distância depende de cada implementação.

Na abordagem ML, um ou mais testes de hipótese são elaborados de acordo com a razão de verossimilhança [33] entre as modulações envolvidas. Na verdade, uma aproximação da razão de verossimilhança é geralmente utilizada, implicando na criação de classificadores sub-ótimos. Polydoros e Huang [29] estabeleceram uma ligação entre classificadores ML e classificadores criados por Liedtke [20] e por Soliman e Hsue [26]. Boiteau e Le Martret [30] generalizaram o trabalho desenvolvido por Polydoros e outros, para o caso em que a duração do pulso não é necessariamente igual ao tempo de duração de um símbolo.

Além disso, outra contribuição importante de Boiteau e Le Martret é a constatação de que a estatística suficiente [33] dos classificadores ML resulta em momentos de ordem elevada. Apesar de existir uma análise teórica bem fundamentada sobre estes métodos, é importante destacar que a grande ênfase dada aos classificadores ML estudados está centrada apenas na classificação de sinais modulados digitalmente em fase (MPSK). Além disso, os classificadores ML são computacionalmente complexos e precisam de um grande número de amostras de sinal, o que também ocorre com os métodos baseados em momentos de ordem elevada e em histogramas.

Neste trabalho, o classificador apresentado em [2], doravante denominado CMADB (Classificador de Modulação baseado em Árvore de Decisão Binária), foi analisado e utilizado como referência para comparação com o classificador de modulações baseado na técnica SVM, proposto no próximo capítulo. Os dois classificadores seguem a abordagem RP e compartilham os mesmos parâmetros extraídos, que são apresentados na próxima seção. Tanto o CMADB como o classificador de modulações proposto são empregados para a classificação de modulações digitais chaveadas em amplitude (ASK2, ASK4), em freqüência (FSK2, FSK4) e em fase (PSK2, PSK4). O número associado a cada tipo de modulação corresponde ao número de bits transmitidos por vez.

Os sinais de comunicações para cada modulação de interesse foram gerados conforme descrito em [2], o que implica na suposição de um canal contaminado por ruído branco, gaussiano e aditivo (canal AWGN). O sinal de comunicações é limitado em banda no transmissor e no receptor pelo uso de filtros com largura de banda apropriada para cada tipo de modulação [2]. A freqüência de amostragem utilizada é

igual a 1,2 Mamostras/s, a freqüência de portadora é igual a 150 KHz e a taxa de símbolos é igual a 12,5 bauds.

O classificador CMADB segue a a abordagem RP, portanto a classificação propriamente dita é precedida de uma fase de extração de características do sinal de comunicação analisado. Os parâmetros extraídos são descritos na seção 5.2. A apresentação do classificador CMADB é feita na seção 5.3, e sua análise, na seção 5.4. O classificador CMADB foi escolhido por apresentar as seguintes vantagens em relação aos demais classificadores para modulações digitais:

- Útil para a implementação em sistemas de classificação em tempo real, pois sua implementação é baseada em uma pequena árvore de decisão binária (estrutura muito semelhante àquela utilizada no classificador DAGSVM apresentado na seção 2.5) em que cada nó apresenta um classificador com pequena complexidade computacional; e
- possibilidade de expansão, podendo incluir até modulações analógicas, como visto em [23].

5.2 Extração de Parâmetros

O classificador CMADB segue a abordagem RP e seus parâmetros, determinados empiricamente, são definidos a seguir.

- γ: Representa o valor máximo da densidade espectral de energia da amplitude instantânea, centralizada e normalizada ($\mathbf{a}_{CN}$). Ou seja:

$$\gamma = max(|DFT(\boldsymbol{a}_{CN})|^2) \qquad \text{Eq. 5.1}$$

onde $\mathbf{a}_{CN}$ é definida, a partir do vetor de amplitude instantânea ($\mathbf{a}_{INST}$), como sendo

$$\mathbf{a}_{CN}(i) = \frac{\mathbf{a}_{INST}(i) - m_A}{m_A}, \qquad m_A = \frac{1}{N}\sum_{i=1}^{N} \mathbf{a}_{INST}(i) \qquad \text{Eq. 5.2}$$

onde N o número de amostras do sinal.

- σ_{AA}: Representa uma estimativa do desvio-padrão do módulo da amplitude instantânea, centralizada e normalizada. Ou seja

$$\sigma_{AA} = \sqrt{\frac{1}{N}\left(\sum_{i=1}^{N} \mathbf{a}_{CN}^2(i)\right) - \left(\frac{1}{N}\sum_{i=1}^{N} |\mathbf{a}_{CN}(i)|\right)^2} \qquad \text{Eq. 5.3}$$

onde N é o número de amostras do sinal.

- σ_{FA}: Representa uma estimativa do desvio-padrão do módulo da freqüência instantânea, centralizada e normalizada. Ou seja

$$\sigma_{FA} = \sqrt{\frac{1}{G}\left(\sum_{G} \mathbf{f}_N^2(i)\right) - \left(\frac{1}{G}\sum_{G} |\mathbf{f}_N(i)|\right)^2} \qquad \text{Eq. 5.4}$$

onde G é o número de amostras em que a amplitude instantânea do sinal é maior do que um limiar a_T, definido pelo usuário. Por sua vez, a freqüência instantânea centralizada e normalizada, $\mathbf{f}_N$, é dada por

$$\mathbf{f}_N(i) = \frac{\mathbf{f}_{INST}(i) - m_F}{r_B}, \qquad m_F = \frac{1}{N}\sum_{i=1}^{N} \mathbf{f}_{INST}(i) \qquad \text{Eq. 5.5}$$

onde r_B é a taxa de bits e $\mathbf{f}_{INST}(i)$ é a i-ésima amostra do vetor que contém as amostras da freqüência instantânea do sinal.

É importante destacar que não há referência sobre o cálculo de $\mathbf{f}_{INST}$ em [2]. A freqüência instantânea poderia ser obtida a partir da aproximação da derivada da fase instantânea pela sua diferença entre dois instantes de tempo consecutivos. Esta estimação é utilizada em outros classificadores de modulação [20,21,23,26]. Ou seja

$$\mathbf{f}_{INST}(i) = \frac{1}{2\pi}[\boldsymbol{\varphi}_{INST}(i) - \boldsymbol{\varphi}_{INST}(i-1)] \qquad \text{Eq. 5.6}$$

onde $\boldsymbol{\varphi}_{INST}(i)$ é a fase instantânea do sinal. No entanto, sabe-se que a freqüência instantânea calculada pela eq. 5.6 é muito sensível ao ruído [34]. Uma melhor estimação da freqüência instantânea é obtida utilizando-se o estimador de Kay, ou a modificação deste último, ambos analisados em [34]. Portanto, o estimador de Kay foi escolhido para o cálculo de $\mathbf{f}_{INST}$, uma vez que a freqüência instantânea é constante (estacionária) na maioria dos casos.

- σ_{AP}: Representa uma estimativa do desvio-padrão do módulo da fase não linear instantânea e centralizada ($\boldsymbol{\varphi}_{NL}$), definida nos pontos em que a amplitude do sinal está acima do limiar previamente estabelecido. Ou seja

$$\sigma_{AP} = \sqrt{\frac{1}{G}\left(\sum_{G} \boldsymbol{\varphi}_{NL}^2(i)\right) - \left(\frac{1}{G}\sum_{G} |\boldsymbol{\varphi}_{NL}(i)|\right)^2} \qquad \text{Eq. 5.7}$$

onde G é o número de amostras em que a amplitude instantânea do sinal é maior do que o limiar a_T. A componente não linear da fase é obtida diminuindo da fase

instantânea, a sua componente linear. A componente linear da fase pode ser obtida fazendo-se

$$\varphi_{\mathbf{L}}(i) = \frac{2\pi \hat{f}_C\, i}{f_S}, \quad \hat{f}_C = m_F \qquad \text{Eq. 5.8}$$

onde f_S é a freqüência de amostragem do sinal e $\hat{f}_C$ é uma estimativa da freqüência da portadora, dada pela média aritmética da freqüência instantânea, m_F, dada pela equação 5.5.

- σ_{DP} : Representa uma estimativa do desvio-padrão da fase não linear instantânea e centralizada, definida nos pontos em que a amplitude do sinal não é pequena. Ou seja

$$\sigma_{DP} = \sqrt{\frac{1}{\mathrm{G}}\left(\sum_{\mathrm{G}} \boldsymbol{\varphi}_{\mathbf{NL}}^2(i)\right) - \left(\frac{1}{\mathrm{G}}\sum_{\mathrm{G}}^{\mathrm{N}} \boldsymbol{\varphi}_{\mathbf{NL}}(i)\right)^2} \qquad \text{Eq. 5.9}$$

onde G é o número de amostras considerado.

Os parâmetros apresentados conferem ao classificador CMADB algumas características interessantes. Por exemplo, o classificador CMADB é imune a variações lentas do ganho do canal, ou seja, se o ganho do canal variar lentamente de tal maneira que ele possa ser considerado constante no intervalo de tempo de um segmento, porém varie de segmento a segmento, ainda assim a forma como a variável $\mathbf{a_{CN}}$ é calculada permite que o parâmetro σ_{AA} seja imune a essa variação.

Além disso, deve-se destacar que não há necessidade de se recuperar a portadora, pois esta é estimada por m_{F} para o cálculo dos parâmetros de interesse que precisam dessa informação (σ_{FA}, σ_{AP}, e σ_{DP}). Além disso, também não é necessário recuperar o sincronismo do relógio devido à superamostragem do sinal, ou seja, cada segmento analisado é composto por 22 símbolos, cada símbolo contendo 96 amostras, e o instante inicial da amostragem não cria nenhum problema para a estimação dos parâmetros de interesse.

Em outras palavras, os parâmetros extraídos permitem a simplificação do receptor que pode ser implementado por um simples circuito de conversão (não coerente) de freqüência. Outro ponto de grande importância prática é o comportamento do classificador na presença de interferência intersimbólica (ISI). Neste ponto deve-se

destacar que os filtros utilizados na recepção e na geração dos sinais para cada modulação são do tipo elíptico, o que gera flutuações na amplitude instantânea dentro da duração de um bit. Os filtros utilizados tem como principal função a limitação de banda do sinal transmitido, assim como a modelagem apropriada de um canal de comunicações. Além disso, os filtros escolhidos geram ISI que como será visto neste e no próximo capítulos não impede a classificação de modulações.

5.3 Algoritmo de Classificação CMADB

Uma vez que todos os parâmetros de interesse foram obtidos, pode-se dar início a etapa de classificação propriamente dita. O algoritmo de classificação consiste em dividir um sinal de N amostras em M segmentos, e classificar cada segmento isoladamente. A classificação do sinal será feita optando-se pela modulação que ocorrer em maior número de segmentos.

A classificação de um segmento consiste na obtenção dos cinco parâmetros descritos acima e na posterior comparação com seus respectivos limiares pré-estabelecidos, segundo a árvore binária de decisão apresentada na figura 5.1. Como se pode perceber, a primeira etapa da classificação de um segmento consiste em separar as modulações que tem informação presente na amplitude instantânea, das que não têm. Esta separação é feita através do parâmetro γ, pois se o sinal tiver amplitude instantânea constante, então $\mathbf{a}_{CN}$ será igual ao vetor nulo, e sua transformada de Fourier também.

Embora teoricamente as modulações PSK2 e PSK4 tenham amplitude instantânea constante, elas não apresentaram esta característica na prática devido à limitação de largura de banda imposta ao sinal oriundo do modulador digital, de modo a tornar a simulação computacional mais realista [2].

Sendo assim, quando o parâmetro γ for maior do que determinado limiar, então a modulação em questão pertence ao grupo {ASK2, ASK4, PSK2, PSK4} . Por sua vez, se o parâmetro gama for menor do que determinado limiar, então a modulação em questão pertence ao grupo {FSK2, FSK4}.

O parâmetro σ_{FA} é usado para definir o tipo de modulação FSK utilizada. O desvio-padrão do módulo da freqüência instantânea deve ser igual a zero para a modulação FSK2, uma vez que esta só pode assumir dois valores, e, conseqüentemente $\mathbf{f_N}$ deve assumir os valores ±1. Em outras palavras, o módulo de $\mathbf{f_N}$ é constante para uma relação sinal-ruído (RSR) infinita. O mesmo não ocorre para a modulação FSK4.

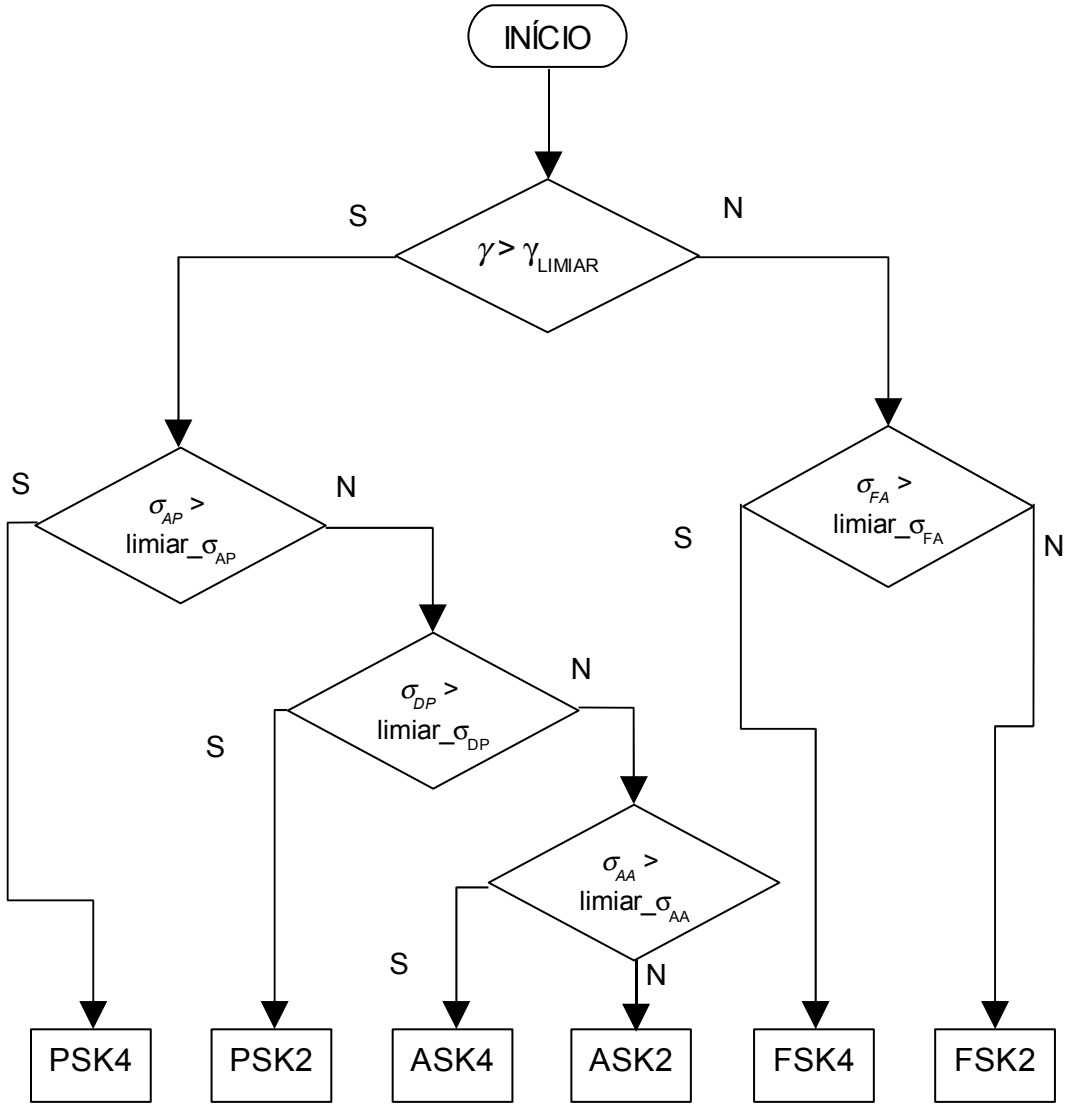

Figura 5.1: Procedimento para a classificação de um segmento.

As modulações em amplitude são separadas das modulações em fase através dos parâmetros σ_{AP} e σ_{DP}, uma vez que as modulações ASK2 e ASK4 apresentam teoricamente fase linear para RSR infinita. Para distinguir entre as modulações PSK2 e PSK4, utiliza-se o parâmetro σ_{AP} uma vez que $\boldsymbol{\varphi}_{NL}$ deve assumir os valores $\pm\pi/2$ para a modulação PSK2, tendo, portanto, módulo constante com desvio padrão nulo para RSR

infinita. O mesmo não ocorre para a modulação PSK4. Logo, se o parâmetro σ_{AP} for maior que um limiar pré-estabelecido, então a modulação em questão é PSK4, caso contrário, pertencerá ao grupo {PSK2, ASK2, ASK4}. Este último grupo pode ser dividido em PSK2 e {ASK2 e ASK4} utilizando o parâmetro σ_{DP} .

Por último, a distinção entre as modulações ASK2 e ASK4 se dá através do parâmetro σ_{AA} , uma vez que para a modulação ASK2, $\mathbf{a_{CN}}$ só pode assumir os valores ±1, tendo, portanto, módulo constante e desvio padrão nulo para RSR infinita. O mesmo não ocorre para a modulação ASK4.

Os limiares utilizados para cada teste foram determinados experimentalmente em [2], através da simulação computacional de vários sinais modulados com a modulação desejada, e posteriormente corrompidos por ruído aditivo branco e gaussiano. Os limiares foram escolhidos de maneira a permitir a separação de cada modulação para RSR (E_b/N_0) maior do que 7dB . Os limiares utilizados pelo algoritmo estudado estão apresentados na tabela 5.1.

Tabela 5.1: Parâmetros e limiares utilizados para o algoritmo estudado.

PARÂMETROS	LIMIARES
γ	10000
σ_{AA}	0,25
σ_{AP}	$\pi/6$
σ_{DP}	$\pi/3$
σ_{FA}	0,35
a_T	0,1

5.4. Análise do Classificador CMADB

Como já foi discutido na seção anterior, a correta distinção entre as modulações ASK2 e ASK4 se dá pelo parâmetro σ_{AA}. Este parâmetro deve ser nulo para a

modulação ASK2, e diferente de zero para a modulação ASK4, para uma RSR infinita. A figura 5.2 apresenta a amplitude instantânea de um sinal ASK2 (gerado conforme determinado em [2]) para a seqüência de bits B={ 0 1 0 1 0 } e para uma RSR infinita. No entanto, se o algoritmo apresentado na figura 5.1 for empregado e a média aritmética da amplitude instantânea, m_A, for calculada de acordo com a eq. 5.2, obtém-se a amplitude $\mathbf{a}_{CN}$ cujo módulo é apresentado na figura 5.3.

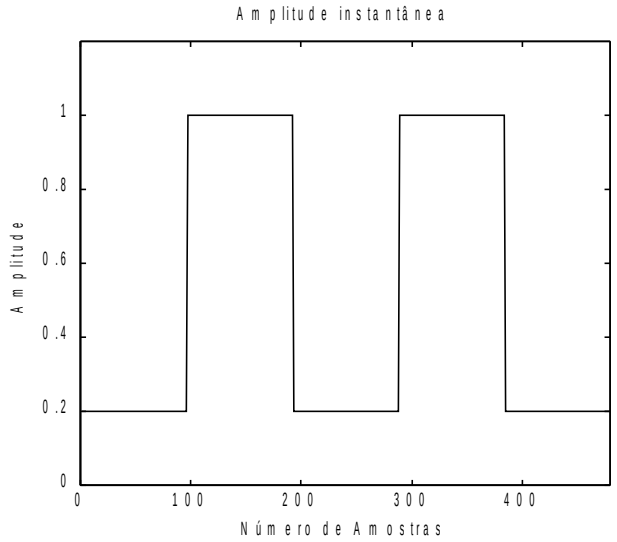

Figura 5.2: Amplitude instantânea (A_{INST}) para RSR infinita.

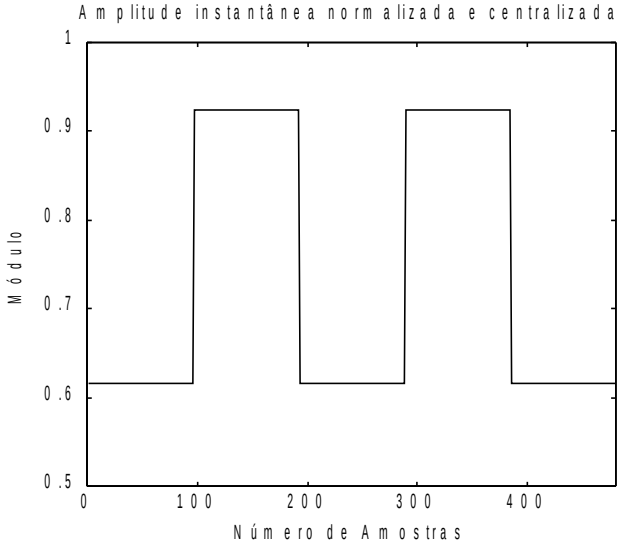

Figura 5.3: Módulo da amplitude instantânea, centralizada e normalizada (a_{CN}) para RSR infinita e m_A=0,52 .

Pode-se perceber que o módulo de $\mathbf{a}_{CN}$ não é constante e portando σ_{AA} não é nulo, mesmo para uma RSR infinita. Isto se deve ao fato de que a média m_A, calculada

pela eq. 5.2, não é igual à média aritmética das amplitudes utilizadas para representar cada um dos símbolos, uma vez que o número de bits '1' e o número de bits '0' gerados pela fonte não são necessariamente iguais. É importante destacar que fato análogo ocorre na distinção entre as modulações FSK2 e FSK4 através do parâmetro σ_{FA} , usando a média m_F calculada pela eq. 5.5, assim como, na distinção entre as modulações PSK2 e PSK4 através do parâmetro σ_{AP} , usando a média da fase não linear.

Do exposto, fica claro que o desempenho do classificador CMADB depende da seqüência de bits que gerou o sinal analisado e não apenas da modulação empregada ou da RSR. Mais especificamente, o desempenho do método depende da diferença entre o número de bits '1' e o número de bits '0'. Se esta diferença não for igual a zero, então o algoritmo pode classificar erroneamente um ou mais segmentos do sinal, mesmo para RSR infinita.

Para avaliar o desempenho do classificador CMADB, foram realizados 50 testes. Em cada teste, foram gerados 120 segmentos para cada modulação. Um segmento contém 22 bits, ou seja, 2112 amostras, o que equivale a 1,76ms. A tabela 5.2 mostra duas matrizes (matrizes de confusão) que sintetizam os resultados obtidos pelo classificador CMADB (apresentado na figura 5.1 e utilizando os limiares apresentados na tabela 5.1) para a classificação de modulação utilizando fontes equiprováveis ideal e real, e para uma RSR de 20dB. Nessa tabela, as linhas determinam o tipo de modulação empregada no sinal de teste, enquanto que as colunas indicam a classificação obtida. Cada célula da tabela apresenta o número de segmentos classificados como sendo da modulação indicada pela respectiva coluna, dividido pelo número total de segmentos gerados para a modulação indicada pela respectiva linha.

Sendo assim, os elementos na diagonal representam as probabilidades de acerto de classificação para cada modulação , enquanto que o somatório dos elementos de uma coluna, excetuando o elemento da diagonal principal, é igual a probabilidade de falso alarme para a modulação correspondente à referida coluna. Por sua vez, a soma de todos os elementos de uma linha deve ser igual a 1. Cabe ainda lembrar que a parte da tabela correspondente à fonte equiprovável ideal foi extraída de [2] e reproduzida neste trabalho por conveniência.

Como se pode ver, para RSR igual a 20 dB, o desempenho real do classificador CMADB é pior do que o apresentado em [2], principalmente no que se refere à classificação correta da modulação ASK2. Neste caso específico, uma pequena melhora pode ser obtida se o limiar para σ_{AA} for reajustado, por exemplo, para 0,30, conforme mostrado na tabela 5.3. No entanto, é importante destacar que à medida que o limiar para σ_{AA} é aumentado, a taxa de acertos para a modulação ASK2 aumenta, porém a taxa de acertos para a modulação ASK4 diminui, implicando no aumento significativo da taxa de falso alarme para a modulação ASK2. Se o limiar para σ_{AA} for diminuído, então ocorrerá o inverso.

Tabela 5.2: Resultados obtidos para o CLASSIFICADOR CMADB (RSR=20dB e σ_{AA} =0,25).

	Fonte Equiprovável Ideal						Fonte Equiprovável Real					
	ASK2	ASK4	FSK2	FSK4	PSK2	PSK4	ASK2	ASK4	FSK2	FSK4	PSK2	PSK4
ASK2	100	0	0	0	0	0	58,75	0,4125	0	0	0	0
ASK4	0	100	0	0	0	0	9,17	90,83	0	0	0	0
FSK2	0	0	100	0	0	0	0	0	99,25	0,75	0	0
FSK4	0	0	0	100	0	0	0	0	10,44	88,48	0	11,08
PSK2	0	0	3,5	0	96,25	0,25	0	0	10,58	0	85,42	4
PSK4	0	0	0	0	1	99	0,08	0	7.67	0	0,17	92,08

Tabela 5.3: Resultados obtidos para o CLASSIFICADOR CMADB (RSR = 20dB e σ_{AA} =0,30)

	Fonte Equiprovável Real					
	ASK2	ASK4	FSK2	FSK4	PSK2	PSK4
ASK2	82,17	17,83	0	0	0	0
ASK4	81,75	18,25	0	0	0	0
FSK2	0	0	99,25	0,75	0	0
FSK4	0	0	10,44	88,48	0	11,08
PSK2	0	0	10,58	0	85,42	4
PSK4	0,08	0	7.67	0	0,17	92,08

Conclui-se que não é possível com este único parâmetro separar as modulações ASK2 e ASK4, pois há uma determinada faixa de valores de σ_{AA} que é comum às duas modulações. Pode-se dizer que o mesmo fato ocorre com as demais modulações e parâmetros. Isto pode ser melhor observado na figura 5.4 que mostra a evolução de cada parâmetro para as seis diferentes modulações (ASK2, ASK4, FSK2, FSK4, PSK2, PSK4) presentes em um sinal de teste que contém 720 segmentos (6x120) e foi gerado

por uma fonte real utilizando uma RSR de 20dB. A linha horizontal mostrada em cada gráfico corresponde ao limiar (figura 5.1 e tabela 5.1) para cada parâmetro.

Em particular, analisando os resultados presentes na figura 5.4, pode-se afirmar que um limiar um pouco maior para γ melhoraria a taxa de acerto de modulações FSK4, porém diminuiria a taxa de acertos para as modulações em fase, que provavelmente seriam classificadas como FSK2. Por sua vez, uma redução no limiar para σ_{FA} também aumentaria a taxa de acerto de FSK4, porém diminuiria a taxa de acerto da modulação FSK2. Tal problema, a difícil sintonia dos diversos limiares, não ocorre no classificador de modulação utilizando a técnica SVM proposto neste trabalho e abordado no próximo capítulo.

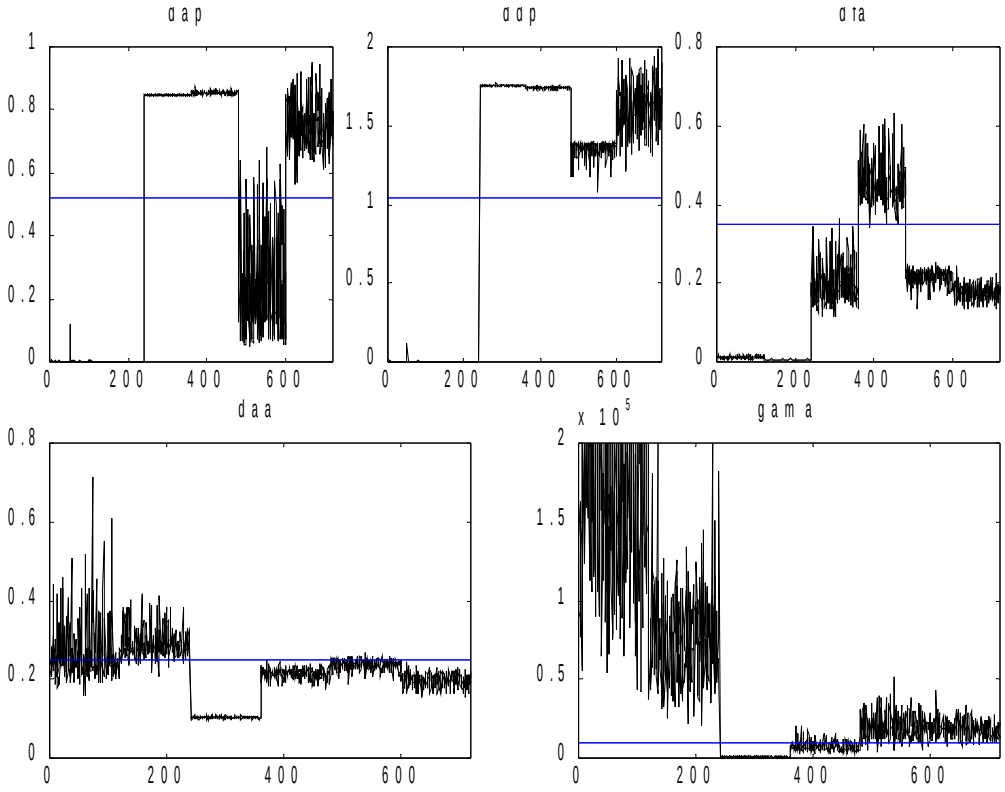

Figura 5.4: Parâmetros Extraídos para as modulações ASK2, ASK4, FSK2, FSK4 PSK2 e PSK4, com RSR de 20dB e fonte real.

A tabela 5.4 mostra os resultados obtidos para uma RSR de 10dB, onde novamente a parte da tabela correspondente à fonte equiprovável ideal foi extraída de [2] e reproduzida neste trabalho por conveniência. Ao comparar os resultados obtidos para cada tipo de fonte, percebe-se que o desempenho do classificador também piorou quando a fonte equiprovável real foi utilizada, exceto para as modulações FSK2 e

PSK4. Porém, não se deve pensar que um pequeno aumento na correta classificação da modulação PSK4 corresponde a uma melhora de desempenho para a classificação desta modulação, pois veio acompanhada de um aumento significativo das respectivas taxas de falso alarme, principalmente em se tratando de sinais FSK4 (devido a valores mais altos de γ, ocasionados pelo ruído).

Com relação à modulação FSK2, percebe-se uma pequena melhora com relação à probabilidade de falso alarme associada às modulações em fase (também devido a valores mais altos de γ) e o aparecimento de um pequeno número de segmentos FSK4 que foram erroneamente classificados como FSK2. Comparando os resultados obtidos para a modulação FSK2 nas tabelas 5.2 e 5.4 para fontes reais percebe-se que não há variação com relação à RSR. Sendo assim, a melhora de desempenho se deve à utilização do estimador de Kay no lugar do estimador BFD para a obtenção da freqüência instantânea, o que leva, conseqüentemente a uma melhor estimativa para σ_{FA}.

Tabela 5.4: Resultados obtidos para o CLASSIFICADOR CMADB (RSR=10dB).

	Fonte Equiprovável Ideal						Fonte Equiprovável Real					
	ASK2	**ASK4**	**FSK2**	**FSK4**	**PSK2**	**PSK4**	**ASK2**	**ASK4**	**FSK2**	**FSK4**	**PSK2**	**PSK4**
ASK2	98,25	1,75	0	0	0	0	53,42	44,16	0	0	0	2,42
ASK4	0	100	0	0	0	0	6,75	93,17	0	0	0	0,08
FSK2	0	0	91,25	8,75	0	0	0	0	99,25	0,75	0	0
FSK4	0	0	0	100	0	0	0	0	0,75	86,58	0	12,67
PSK2	0	0	9	0	90,75	0,25	0	0	8,67	0	88,33	3
PSK4	0	0	10,75	0	0	89,25	0	0	8,25	0	0,08	91,67

Tal fato pode ser comprovado comparando as figuras 5.4 e 5.5, onde esta última mostra a evolução de cada parâmetro para as seis diferentes modulações (ASK2, ASK4, FSK2, FSK4, PSK2, PSK4) presentes em um sinal de teste que contém 720 segmentos (6x120) e foi gerado por uma fonte real utilizando uma RSR de 10dB.

Comparando as duas figuras percebe-se que o parâmetro σ_{FA} varia muito pouco com relação à RSR (para cada um das modulações analisadas), o que explica a pequena variação nas taxas de acerto para as modulações FSK2 e FSK4 (tabelas 5.2 e 5.4). Destaca-se também que as modulações em freqüência obtiveram a menor variância para todos os parâmetros avaliados exceto para σ_{FA}.

Em resumo, o classificador CMADB tem um desempenho muito pior do que o publicado em [2] devido à difícil sintonia dos seus diversos limiares, uma vez que seus parâmetros são muito sensíveis à distribuição de zeros e uns do sinal digital e à relação sinal-ruído, quando uma fonte equiprovável real é utilizada. Novamente, vale a pena destacar que tal problema não ocorre no classificador de modulação utilizando a técnica SVM proposto neste trabalho e abordado no próximo capítulo. O classificador CMSVM proposto só utiliza um parâmetro cuja escolha é simples.

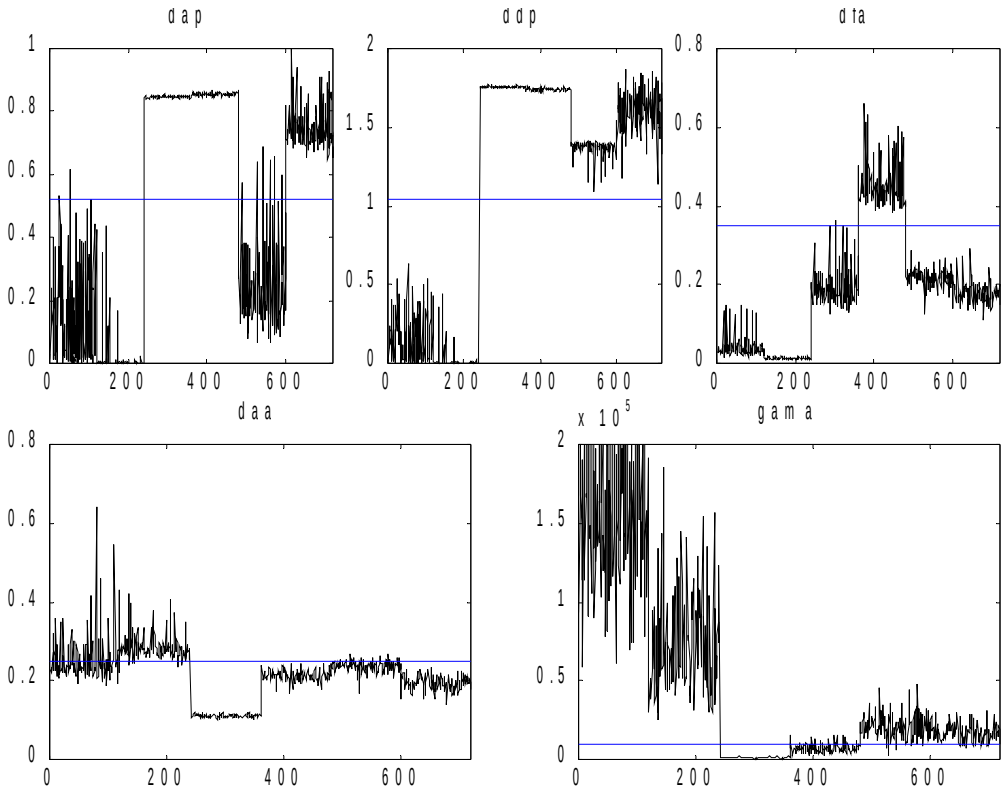

Figura 5.5: Parâmetros Extraídos para as modulações ASK2, ASK4, FSK2, FSK4 PSK2 e PSK4, com RSR de 10dB e fonte real.

6. O CLASSIFICADOR DE MODULAÇÃO CMSVM

6.1 Introdução

Neste capítulo, a aplicação da técnica SVM para a classificação de sinais de comunicação no que se refere à sua modulação é abordada. Tendo em vista que o problema em questão envolve 6 classes, é necessário utilizar um dos métodos de generalização descritos na seção 2.5. Optou-se por utilizar o esquema de generalização do tipo 1vs1 (um contra um). Esta escolha se deu porque o treinamento do classificador em questão é mais rápido se comparado ao treinamento utilizando o método 1vsR (um contra todos), além disso, permite fácil inclusão ou exclusão de classes, e, por último, não possui os problemas apresentados pelo método DAGSVM, no que se refere à preferência dada a algumas classes de acordo com a posição que ocupam na árvore.

A figura 6.1 mostra o diagrama de blocos do classificador de modulação proposto neste trabalho, doravante chamado CMSVM. Ele é baseado em subclassificadores UPSVM e utiliza o esquema de generalização 1x1. A fase de extração de características foi descrita no capítulo anterior, sendo responsável por transformar um segmento do sinal de comunicações, **x**, em um vetor de características . Ao analisar este último, o subclassificador UPSVM de índice K será responsável por separar a classe i da classe j, onde i varia de 1 a 5, e j, sempre maior que i, varia de 2 a 6. A saída desse subclassificador será igual a i ou j de acordo com a classe vitoriosa.

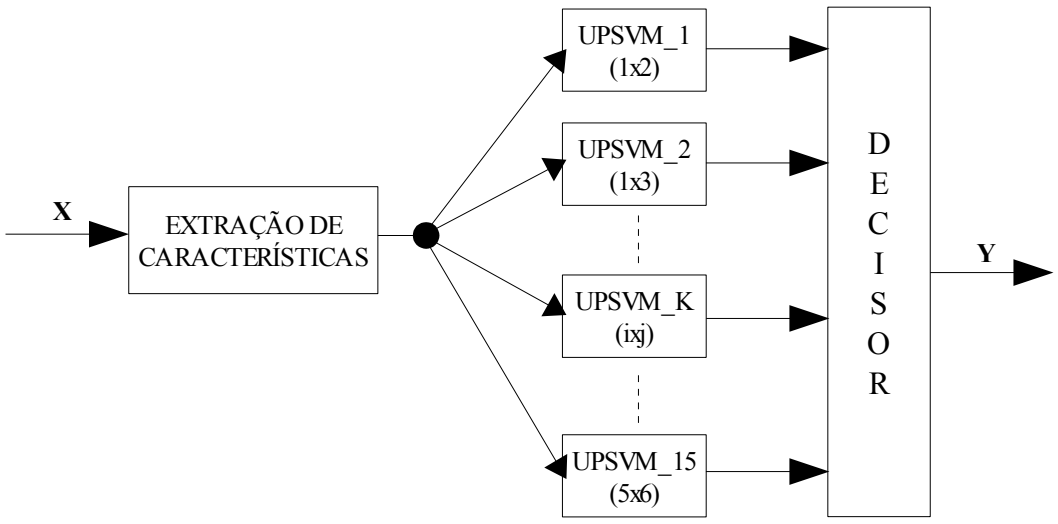

Figura 6.1 – Diagrama de blocos do classificador CMSVM.

Por sua vez, o bloco decisor foi construído a partir da regra apresentada por Friedman [9], ou seja, a saída de cada classificador conta como um voto para a respectiva classe e a classificação final será dada pela classe que obteve o maior número de votos. Em caso de empate entre duas classes, optou-se por escolher a classe que venceu o confronto direto. No caso de empate com um número maior de modulações, a classificação se dará pela modulação para a qual o vetor de características analisado obteve a maior distância,em módulo, em relação ao hiperplano ótimo.

6.2 Treinamento e Teste do Classificador CMSVM

Antes de treinar um classificador CMSVM, é necessário criar um sinal de treinamento e extrair dele as características relevantes citadas no capítulo anterior. Logo, inicialmente, um sinal de treinamento composto por 120 segmentos de 22bits (120x22x96 amostras por sinal) foi gerado para cada modulação que se deseja classificar. Os sinais de treinamento foram gerados conforme descrito em [2], ou seja, considerando uma freqüência de amostragem de 1,2 Mamostras/s, uma freqüência de portadora de 150 KHz e uma taxa de símbolos de 12,5 bauds. Posteriormente, todos os sinais de treinamento passaram por uma fase de extração de características, resultando, para cada modulação, em uma matriz de parâmetros, $\mathbf{X} \in \Re^{5x120}$, onde cada linha corresponde a evolução de um determinado parâmetro entre aqueles definidos na tabela 6.1. O procedimento descrito acima foi realizado para RSRs de 10 e 20 dB.

Para o treinamento propriamente dito do subclassificador responsável por separar as classes i e j, onde i e j correspondem aos índices da modulação dentro do conjunto de classes possíveis (Classes={ASK2, ASK4, FSK2, FSK4, PSK2, PSK4}), utilizou-se a rotina UPSVM e *kernel* linear. A matriz de parâmetros de treinamento foi obtida através da concatenação das matrizes $\mathbf{X_i}$ e $\mathbf{X_j}$, correspondentes, respectivamente, às matrizes de parâmetros para as modulações i e j e obtidas na fase anterior. Foi realizado um treinamento para cada valor do parâmetro de regularização C entre os seguintes valores: 0.01, 0.1, 1, 10, e 100. Sendo assim, após a fase de treinamento, foram criados 10 diferentes classificadores CMSVM de acordo com os valores escolhidos para RSR e C. A listagem 6.1 mostra o pseudocódigo do procedimento de treinamento realizado.

Para avaliar o desempenho do classificador CMSVM, foram criados 50 sinais de teste para cada modulação e RSR de interesse. Cada sinal de teste contém 120 segmentos, totalizando 6000 segmentos (experimentos) por modulação e RSR. Foram escolhidos valores iguais a 5, 10, 15 e 20dB como RSR dos sinais de teste. Cada um dos 10 classificadores CMSVM obtidos após a fase de treinamento foi utilizado para classificar os diversos sinais de teste gerados, fornecendo como resultado final uma matriz com 50 linhas, uma para cada sinal de teste; e 720 colunas (6 mod x 120 seg), uma para cada segmento. A listagem 6.2 mostra o pseudocódigo para o teste realizado.

Listagem 6.1: Pseudocódigo para Treinamento CMSVM

Algoritmo TREINAR_CMSVM
INÍCIO
Para RSR=10 dB e 20 dB faça
    Para i = 1 até 6 faça
        - *Gerar sinal de treinamento (120 x 22 bits) para a modulação de índice i*
        - $\boldsymbol{X_i}$ *= parâmetros de interesse extraídos para a modulação de índice i*
    Fim_Para
    Armazenar as matrizes $\boldsymbol{X_1}$, $\boldsymbol{X_2}$, $\boldsymbol{X_3}$, $\boldsymbol{X_4}$, $\boldsymbol{X_5}$ *e* $\boldsymbol{X_6}$ *em arquivo*
    Para j = 1 até 5
        - *Treinar os 15 subclassificadores UPSVM para o valor de C de índice j (C = 0.01, 0.1, 1, 10 e 100), gerando a matriz de pesos,* $\boldsymbol{P_j} \in \Re^{6x15}$
        - *Armazenar a matriz de pesos (*$\boldsymbol{P_j} \in \Re^{6x15}$*) em arquivo .*
    Fim_Para
Fim_Para
FIM

Listagem 6.2: Pseudocódigo para Avaliação do CMSVM

Algoritmo TESTAR_CMSVM
INÍCIO
Para RSR = 5, 10, 15 e 20dB faça
    Para i = 1 até 50 (Nº total de sinais de teste) faça
        Para j = 1 até 6 faça
            - $\boldsymbol{X_j}$ *= parâmetros de interesse extraídos para a modulação de índice j*
        Fim_Para
        - *Armazenar o sinal de teste* $\boldsymbol{X} = [\boldsymbol{X_1}\ \boldsymbol{X_2}\ \boldsymbol{X_3}\ \boldsymbol{X_4}\ \boldsymbol{X_5}\ \boldsymbol{X_6}]$ *em arquivo*
        Para k =1 até 10 (Nº total de classificadores) faça
            - *Classificar o* $i^{º}$ *sinal de teste utilizando o* $k^{º}$ *classificador CMSVM*
            - *Armazenar vetor de classificação com seus 720 elementos na i-ésima linha da matriz* ***ÍNDICE***
        Fim Para
    - *Armazenar a matriz* ***ÍNDICE*** *em um arquivo específico*
    Fim_Para
Fim_Para
FIM

6.3 Análise do classificador CMSVM

6.3.1 O classificador CMSVM e o Parâmetro C

Com o objetivo de estudar a relevância do parâmetro C no treinamento e classificação de um sinal de comunicações, foram analisados os resultados obtidos pela classificação de sinais de teste com RSR igual a 20dB, utilizando classificadores SVM treinados a partir de um sinal de treinamento também com RSR igual a 20 dB. As tabelas 6.1 a 6.3 mostram a matriz de confusão para diferentes valores do parâmetro de regularização C.

Tabela 6.1:Resultados obtidos para o CLASSIFICADOR CMSVM para RSR igual a 20 dB (C = 0,01 e 0,1)

	RSR_TESTE=20dB e C=0,01						RSR_TESTE=20dB e C=0,1					
	ASK2	ASK4	FSK2	FSK4	PSK2	PSK4	ASK2	ASK4	FSK2	FSK4	PSK2	PSK4
ASK2	71,07	28,93	0	0	0	0	73,63	26,37	0	0	0	0
ASK4	5,28	93,84	0	0	0,88	0	5,17	94,83	0	0	0	0
FSK2	0	0	100	0	0	0	0	0	100	0	0	0
FSK4	0	0	0	98,33	0	1,67	0	0	0	99,38	0	0,62
PSK2	0	0	1,67	5,63	79,15	13,55	0	0	0	0	99,58	0,42
PSK4	0	0	0,84	2,48	11,78	84,9	0	0	0,9	0	0	99,1

Tabela 6.2:Resultados obtidos para o CLASSIFICADOR CMSVM para RSR igual a 20 dB (C = 1 e 10)

	RSR_TESTE=20dB e C=1						RSR_TESTE=20dB e C=10					
	ASK2	ASK4	FSK2	FSK4	PSK2	PSK4	ASK2	ASK4	FSK2	FSK4	PSK2	PSK4
ASK2	75,57	24,43	0	0	0	0	81,95	18,05	0	0	0	0
ASK4	4,18	95,82	0	0	0	0	2,5	97,5	0	0	0	0
FSK2	0	0	100	0	0	0	0	0	100	0	0	0
FSK4	0	0	0	100	0	0	0	0	0	100	0	0
PSK2	0	0	0	0	100	0	0	0	0	0	100	0
PSK4	0	0	0,87	0	0	99,13	0	0	0,53	0	0	99,47

Tabela 6.3:Resultados obtidos para o CLASSIFICADOR CMSVM para RSR igual a 20 dB (C = 100)

	RSR_TESTE=20dB e C=100					
	ASK2	ASK4	FSK2	FSK4	PSK2	PSK4
ASK2	94,05	5,95	0	0	0	0
ASK4	0,83	99,17	0	0	0	0
FSK2	0	0	100	0	0	0
FSK4	0	0	0	100	0	0
PSK2	0	0	0	0	100	0
PSK4	0	0	0	0	0,02	99,98

Da análise desses resultados, pode-se concluir que:

- A média da taxa de acerto para todas as modulações, caracterizada pela diagonal principal de cada matriz de confusão, aumenta a medida que o valor de C aumenta.
- Independentemente do valor de C utilizado, as modulações em amplitude foram facilmente separadas das modulações exponenciais. Este fato fica evidente ao observar que todas as matrizes de confusão apresentam valores nulos para as primeiras duas colunas, da 3ª linha em diante, assim como para as duas primeiras linhas, da 3ª coluna em diante, exceto para C=1 e modulação ASK4 que em um número muito pequeno de casos foi confundida com a modulação PSK2.
- Exceto para c=0,01, as modulações mais difíceis de serem separadas são ASK2 e ASK4. Este resultado já era esperado tendo em vista que o ruído empregado é aditivo (canal AWGN). A melhoria no desempenho do classificador CMSVM com relação ao parâmetro C foi mais significativa justamente para estas duas modulações. Em particular, para C=100, o desempenho do classificador proposto no que se refere às modulações em amplitude se assemelha ao desempenho obtido para as demais modulações, supostamente mais fáceis de serem reconhecidas.
- A modulação FSK2 foi corretamente classificada para todos os valores de C utilizados. Por sua vez, a modulação FSK4 foi facilmente detectada para qualquer valor de C empregado. Este fato se deve principalmente ao uso do estimador freqüência instantânea [34] utilizado no lugar do algoritmo BFD.
- Os erros mais comuns, para qualquer valor de C, foram a confusão entre modulações ASK2 e ASK4. Este resultado sugere que o uso de um *kernel* não linear para o subclassificador SVM associados pode melhorar ainda mais os resultados obtidos.
- Por fim, o classificador CMSVM obteve resultados muito melhores do que os obtidos pelo classificador CMADB (tabela 4.2. - fonte equiprovável real), mesmo para valores baixos de C, exceto para a classificação de modulações em fase quando C=0,01. Além disso, ao alterar o valor do parâmetro C, o desempenho do classificador CMSVM melhora para todas as modulações, enquanto que ao modificarmos um determinado limiar do classificador CMADB, a melhoria de desempenho no reconhecimento de uma determinada

modulação geralmente vem acompanhada de uma queda de desempenho no reconhecimento de outra(s) modulação(ões).

Resultados semelhantes foram obtidos para a classificação de sinais de teste com RSR igual a 10dB, utilizando classificadores SVM treinados a partir de um sinal de treinamento também com RSR igual a 10 dB. As tabelas 6.4 a 6.6 mostram esses resultados.

Tabela 6.4:Resultados obtidos para o CLASSIFICADOR CMSVM para RSR igual a 10 dB (C = 0,01 e 0,1)

	RSR_TESTE=10dB e C=0,01						RSR_TESTE=10dB e C=0,1					
	ASK2	ASK4	FSK2	FSK4	PSK2	PSK4	ASK2	ASK4	FSK2	FSK4	PSK2	PSK4
ASK2	75,97	24,03	0	0	0	0	84,17	15,83	0	0	0	0
ASK4	7,55	92,45	0	0	0	0	3,58	96,42	0	0	0	0
FSK2	0	0	100	0	0	0	0	0	100	0	0	0
FSK4	0	0	0	98,34	0,83	0,83	0	0	0	99,17	0	0,83
PSK2	0	0	0,75	4,3	76,65	18,3	0	0	0	0	98,72	1,28
PSK4	0		1,67	3,85	11,57	82,92	0	0	2,8	0	0	97,2

Tabela 6.5:Resultados obtidos para o CLASSIFICADOR CMSVM para RSR igual a 10 dB (C = 1 e 10)

	RSR_TESTE=10dB e C=1						RSR_TESTE=10dB e C=10					
	ASK2	ASK4	FSK2	FSK4	PSK2	PSK4	ASK2	ASK4	FSK2	FSK4	PSK2	PSK4
ASK2	85,7	14,3	0	0	0	0	90,18	9,82	0	0	0	0
ASK4	0,15	99,85	0	0	0	0	0,02	99,98	0	0	0	0
FSK2	0	0	100	0	0	0	0	0	100	0	0	0
FSK4	0	0	0	99,17	0	0,83	0	0	0	99,17	0	0,83
PSK2	0	0	0	0	100	0	0	0	0	0	100	0
PSK4	0	0	1,82	0	0	98,18	0	0	2,03	0	0	97,97

Tabela 6.6:Resultados obtidos para o CLASSIFICADOR CMSVM para RSR igual a 10 dB (C = 100)

	RSR_TESTE=10dB e C=100					
	ASK2	ASK4	FSK2	FSK4	PSK2	PSK4
ASK2	90,97	9,03	0	0	0	0
ASK4	0,03	99,97	0	0	0	0
FSK2	0	0	100	0	0	0
FSK4	0	0	0	99,17	0	0,83
PSK2	0	0	0	0	100	0
PSK4	0	0	0,03	0	0	99,97

Além do melhor desempenho, cabe também destacar que o classificador CMSVM apresenta uma outra vantagem se comparado com o classificador CMADB. A

expansão deste último para outras modulações implica na criação de novos parâmetros e limiares [23], uma vez que cada parâmetro e respectivo limiar é responsável por separar uma única modulação. Isto torna a fase de pré-processamento do sinal mais lenta e a classificação propriamente dita mais complexa, porque existirá um número maior de limiares para serem sintonizados. Por sua vez, o classificador CMSVM proposto pode ser expandido para reconhecer novas modulações sem a necessidade de criação de novos parâmetros, bastando apenas treinar os novos subclassificadores responsáveis por distinguir a classe que está sendo incluída das classes existentes. Os demais subclassificadores permanecem inalterados. No caso de exclusão de uma modulação, basta excluir todos os subclassificadores relacionados com à modulação excluída.

6.3.2 O classificador CMSVM e a relação RSR

A avaliação realizada até o momento presume que os sinais de teste têm a mesma RSR do sinal de comunicações utilizado para treinamento de cada subclassificador SVM. Na prática, isto não ocorre. Sendo assim, com o objetivo de avaliar o desempenho do classificador CMSVM com relação à diferentes RSR, também foram gerados sinais de teste com RSR iguais a 5 e 15 dB. As tabelas 6.7 a 6.15 mostram os resultados obtidos utilizando diferentes valores de C e de RSR para sinais de testes, levando em consideração que cada subclassificador foi treinado utilizando um sinal de treinamento com RSR de 20dB. As figuras 6.2 a 6.4 sintetizam os resultados obtidos no que se refere à média da taxa de acerto (diagonal principal para diferentes modulações e RSR.

Tabela 6.7:Resultados obtidos para o CLASSIFICADOR CMSVM para RSR de 20dB no treinamento e RSR de 15 dB para teste (C = 0,01 e 0,1)

	RSR_TESTE=15dB e C=0,01						RSR_TESTE=15dB e C=0,1					
	ASK2	ASK4	FSK2	FSK4	PSK2	PSK4	ASK2	ASK4	FSK2	FSK4	PSK2	PSK4
ASK2	72,55	27,45	0	0	0	0	73,3	26,7	0	0	0	0
ASK4	10,43	88,87	0	0	0,7	0	10,37	89,63	0	0	0	0
FSK2	0	0	100	0	0	0	0	0	100	0	0	0
FSK4	0	0	0	98,08	0	1,92	0	0	0	100	0	0
PSK2	0	0	3,33	5,03	79,56	12,08	0	0	0	0	99,98	0,02
PSK4	0	0	0	3,77	9,27	86,96	0	0,92	0	0	0,83	98,25

Tabela 6.8:Resultados obtidos para o CLASSIFICADOR CMSVM para RSR de 20dB no treinamento e RSR de 15 dB para teste (C = 1 e 10)

	RSR_TESTE=15dB e C=1						RSR_TESTE=15dB e C=10					
	ASK2	ASK4	FSK2	FSK4	PSK2	PSK4	ASK2	ASK4	FSK2	FSK4	PSK2	PSK4
ASK2	80,52	19,48	0	0	0	0	95,05	4,95	0	0	0	0
ASK4	8,47	91,53	0	0	0	0	8,18	91,82	0	0	0	0
FSK2	0	0	100	0	0	0	0	0	100	0	0	0
FSK4	0	0	0	100	0	0	0	0	0	100	0	0
PSK2	0	0	0	0	100	0	0	0	0	0	100	0
PSK4	0	0	0,83	0	0,83	98,34	0	0	0,83	0	0,83	98,34

Tabela 6.9:Resultados obtidos para o CLASSIFICADOR CMSVM para RSR de 20dB no treinamento e RSR de 15 dB para teste (C = 100)

	RSR_TESTE=15dB e C=100					
	ASK2	ASK4	FSK2	FSK4	PSK2	PSK4
ASK2	99,1	0,9	0	0	0	0
ASK4	10,87	89,13	0	0	0	0
FSK2	0	0	100	0	0	0
FSK4	0	0	0	100	0	0
PSK2	0	0	0	0	100	0
PSK4	0	0	0	0	0,83	99,17

Tabela 6.10:Resultados obtidos para o CLASSIFICADOR CMSVM para RSR de 20dB no treinamento e RSR de 10 dB para teste (C = 0,01 e 0,1)

	RSR_TESTE=10dB e C=0,01						RSR_TESTE=10dB e C=0,1					
	ASK2	ASK4	FSK2	FSK4	PSK2	PSK4	ASK2	ASK4	FSK2	FSK4	PSK2	PSK4
ASK2	73,73	26,27	0	0	0	0	75,57	24,43	0	0	0	0
ASK4	8,46	90,47	0	0	1,07	0	8,72	91,28	0	0	0	0
FSK2	0	0	100	0	0	0	0	0	100	0	0	0
FSK4	0	0	0	98,27	0,83	0,9	0	0	0	99,17	0	0,83
PSK2	0	0	0	5,03	79,55	15,42	0	0	0	0	98,93	1,07
PSK4	0	0	0,83	4,85	10,32	84	0	0	2,65	0	0	97,35

Tabela 6.11:Resultados obtidos para o CLASSIFICADOR CMSVM para RSR de 20dB no treinamento e RSR de 10 dB para teste (C = 1 e 10)

	RSR_TESTE=10dB e C=1						RSR_TESTE=10dB e C=10					
	ASK2	ASK4	FSK2	FSK4	PSK2	PSK4	ASK2	ASK4	FSK2	FSK4	PSK2	PSK4
ASK2	87,87	11,3	0	0	0,83	0	98,33	0	0	0	1,2	0,47
ASK4	7,27	92,73	0	0	0	0	15,62	84,38	0	0	0	0
FSK2	0	0	100	0	0	0	0	0	100	0	0	0
FSK4	0	0	0	100	0	0	0	0	0	100	0	0
PSK2	0	0	0	0	100	0	0	0	0	0	100	0
PSK4	0	0	1,67	0	0,02	98,31	0	0	1,53	0	0	98,47

Tabela 6.12:Resultados obtidos para o CLASSIFICADOR CMSVM para RSR de 20dB no treinamento e RSR de 10 dB para teste (C = 100)

	RSR_TESTE=10dB e C=100					
	ASK2	ASK4	FSK2	FSK4	PSK2	PSK4
ASK2	98,33	0	0	0	1,25	0,42
ASK4	68,3	31,7	0	0	0	0
FSK2	0	0	100	0	0	0
FSK4	0	0	0	100	0	0
PSK2	0	0	0	0	100	0
PSK4	0	0	0	0	0	100

Tabela 6.13:Resultados obtidos para o CLASSIFICADOR CMSVM para RSR de 20dB no treinamento e RSR de 5 dB para teste (C = 0,01 e 0,1)

	RSR_TESTE=5dB e C=0,01						RSR_TESTE=5dB e C=0,1					
	ASK2	ASK4	FSK2	FSK4	PSK2	PSK4	ASK2	ASK4	FSK2	FSK4	PSK2	PSK4
ASK2	74,22	25,78	0	0	0	0	74,6	19,5	0	0	5,9	0
ASK4	4,5	93,23	0	0	2,27	0	4,6	95,4	0	0	0	0
FSK2	0	0	100	0	0	0	0	0	100	0	0	0
FSK4	0	0	0	94,85	0	5,15	0	0	0	98,95	0	1,05
PSK2	0	0	0	2,83	72,57	24,6	0	0	0	0	98,2	1,8
PSK4	0	0	0,83	2,67	9,45	87,05	0	0	0,85	0	0,83	98,32

Tabela 6.14:Resultados obtidos para o CLASSIFICADOR CMSVM para RSR de 20dB no treinamento e RSR de 5 dB para teste (C = 1 e 10)

	RSR_TESTE=5dB e C=1						RSR_TESTE=5dB e C=10					
	ASK2	ASK4	FSK2	FSK4	PSK2	PSK4	ASK2	ASK4	FSK2	FSK4	PSK2	PSK4
ASK2	71,4	3,28	0	0	18,97	6,35	66,63	0,12	0	0	3,83	29,42
ASK4	6,05	93,95	0	0	0	0	41,88	58,12	0	0	0	0
FSK2	0	0	100	0	0	0	0	0	100	0	0	0
FSK4	0	0	0	100	0	0	0	0	0	100	0	0
PSK2	0	0	0	0	100	0	0	0	0	0	100	0
PSK4	0	0	0,48	0	1,05	98,47	0	0	0	0	0,9	99,1

Tabela 6.15:Resultados obtidos para o CLASSIFICADOR CMSVM para RSR de 20dB no treinamento e RSR de 5 dB para teste (C = 100)

	RSR_TESTE=5dB e C=100					
	ASK2	ASK4	FSK2	FSK4	PSK2	PSK4
ASK2	65,45	0	0	0	1,68	32,87
ASK4	99,8	0,2	0	0	0	0
FSK2	0	0	100	0	0	0
FSK4	0	0	0	100	0	0
PSK2	0	0	0	0	100	0
PSK4	0	0	0	0	0,83	99,17

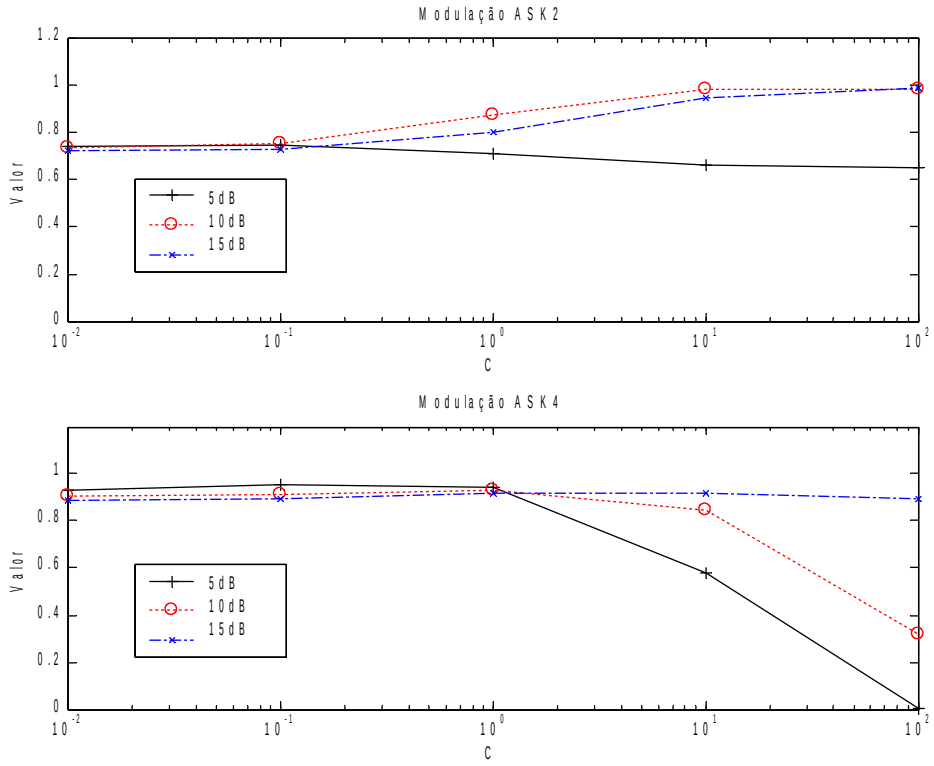

Figura 6.2: Média da taxa de acertos para o CLASSIFICADOR CMSVM para modulações em amplitude

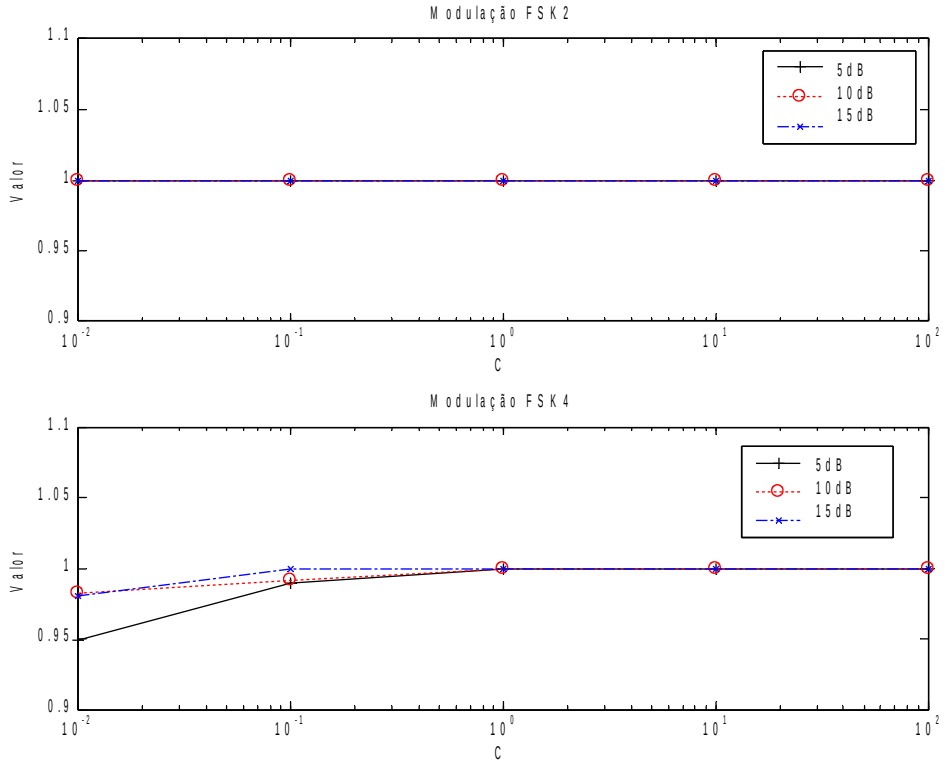

Figura 6.3: Média da taxa de acertos para o CLASSIFICADOR CMSVM para modulações em freqüência

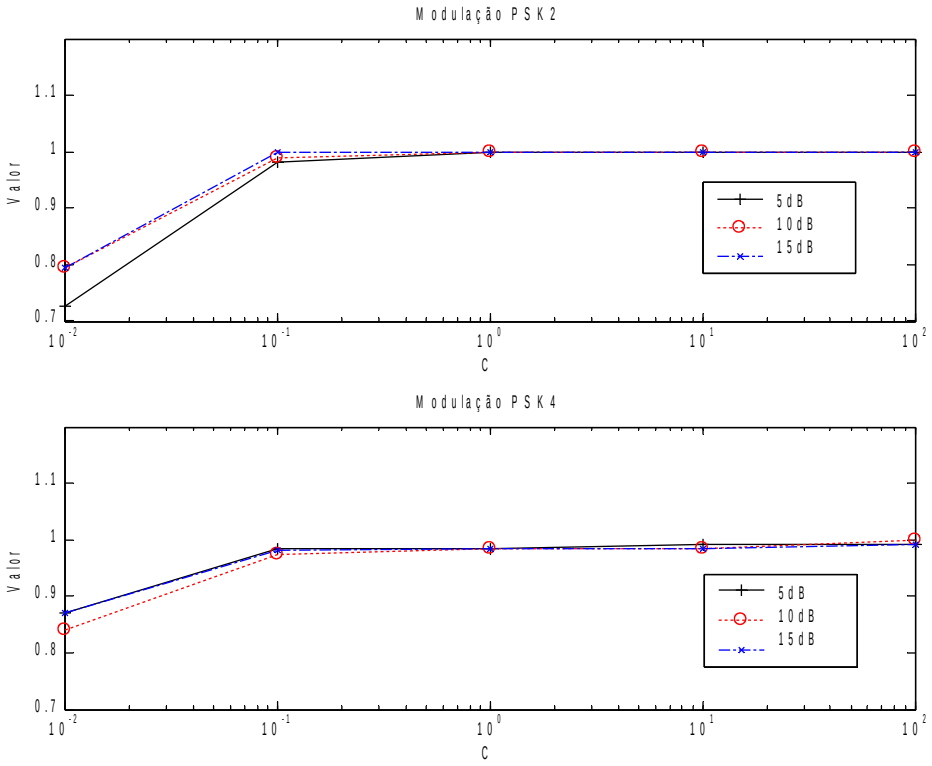

Figura 6.4: Média da taxa de acertos para o CLASSIFICADOR CMSVM para modulações em fase

Destes resultados pode-se concluir que:

- Pode-se observar nitidamente um comportamento muito diferente com relação às modulações em amplitude e exponenciais. No que se refere às modulações exponenciais, conclui-se que os resultados obtidos para qualquer RSR sofreram pequenas alterações em relação ao parâmetro C quando este variou de 0.1 a 100. Sendo que o desempenho sempre piorou ou manteve-se inalterado à medida que o parâmetro C diminuía.
- A modulação FSK2 sempre foi corretamente classificada para qualquer valor de C e para qualquer RSR Este fato se deve principalmente ao uso do estimador de freqüência instantânea, estudado em [34], no lugar do algoritmo BFD.
- Para as modulações em amplitude, pode-se observar, para uma RSR de 15dB, próximo da RSR utilizada para o treinamento, que o melhor resultado para a modulação ASK2 foi obtido quando C=100 e foi piorando a medida que C diminuía, enquanto que a modulação ASK4 teve uma pequena melhora na sua classificação utilizando C=10 ou C=1 para uma RSR de 10dB. Neste caso, o

melhor resultado para a modulação ASK4 foi obtido para C=1. Já para C=100, o resultado obtido foi muito ruim, o pior de todos.

- Para uma RSR de 10 dB, os resultados obtidos para a modulação ASK2 se mantiveram estáveis por mais tempo, ou seja, a classificação deste tipo de modulação não foi prejudicada para C=10 e caiu menos para C=1 se comparado com os resultados obtidos para RSR igual a 15 dB. Por último, para uma RSR de 5dB, tanto para as modulação ASK2 como para a modulação ASK4, os resultados melhoraram a medida que C diminuía até C=0.1, condição em que os melhores resultados foram observados para as duas modulações.

Em resumo, percebe-se que as modulações exponenciais são menos sensíveis à escolha do parâmetro C do que as modulações em amplitude, quando os sinais a serem classificados não têm a mesma RSR do sinal utilizado no treinamento. Ainda com relação a modulações exponenciais, um valor alto do parâmetro de regulação C garante um bom desempenho, independentemente da RSR do sinal de teste.

Por sua vez, a escolha do parâmetro C passa a ter um papel preponderante na classificação de modulações em amplitude. Percebe-se claramente que a taxa de acerto para a modulação ASK4 cai drasticamente à medida que a RSR também diminui, para C>1, enquanto a taxa de falso alarme e a taxa de acerto para a modulação ASK2 aumentam. A única exceção ocorre para uma RSR igual a 5dB, quando uma boa parte das observações em ASK2 são erroneamente classificadas como sendo do tipo PSK4.

Cabe ressaltar que a estimação da probabilidade de erro apresentada na subseção 3.4.2 não pode ser aplicada aqui, pois sua análise teórica parte do pressuposto que a observação usada no teste tem as mesmas média e matriz de covariância das observações utilizadas no treinamento, o que não ocorre quando a RSR na fase de treinamento é diferente na fase de teste.

Para valores mais baixos de C, percebe-se que há pouca variação da probabilidade de acerto na classificação de modulações ASK2 e ASK4, porém com valores abaixo de 75% para a modulação ASK2. Conclui-se que não há um valor ótimo para o parâmetro C quando a RSR não é conhecida a priori. Neste caso, como já

discutido na seção anterior, o uso de um *kernel* não linear para o subclassificador SVM associado à separação das classes ASK2 e ASK4 pode obter melhores resultados.

6.3.3 O uso de kernel não linear no classificador CMSVM

A escolha do parâmetro C não é difícil para a classificação de modulações exponenciais, no entanto é preponderante para a classificação de modulações em amplitude supondo que a RSR do sinal de teste pode variar dentro de uma grande faixa dinâmica. Sendo assim, decidiu-se estudar o emprego de um subclassificador utilizando o algoritmo UPSVM não linear para separar apenas as modulações ASK2 e ASK4. O *kernel* escolhido foi a função de base radial RBF, definida na seção 2.4, tabela 2.1, onde a variância do *kernel* é doravante representada por σ_{RBF}.

O treinamento do subclassificador responsável por separar as classes ASK2 e ASK4 utilizou como entrada uma matriz de parâmetros de treinamento contendo 120 observações de cada classe, com uma RSR de 20dB. Um subclassificador foi obtido para cada valor de C e de σ_{RBF}, que puderam assumir independentemente um dos seguintes valores: 0,01; 0,1; 1; 10 ou 100. Sendo assim, após a fase de treinamento foram obtidos 25 diferentes subclassificadores, procedendo-se de forma bastante parecida ao procedimento descrito pela listagem 6.1.

Por sua vez, a avaliação de cada subclassificador foi feita de forma análoga ao procedimento descrito na listagem 6.2, supondo apenas 2 modulações e 25 subclassificadores. As tabelas 6.16 a 6.19 mostram os resultados obtidos para cada valor de σ_{RBF}, C e RSR.

Se a RSR não é conhecida (não foi estimada anteriormente) apenas uma configuração (subclassificador) pode ser utilizada na prática. Neste caso, para diminuir o número de possíveis candidatos, optou-se por escolher as configurações que garantam uma probabilidade de acerto média em relação à RSR maior do que 90%. De posse desse resultado, buscou-se encontrar a configuração com a menor diferença média entre taxas de acerto. Valores de C e σ_{RBF} iguais a 100 correspondem a melhor configuração obtida, garantindo que uma taxa de acerto na classificação maior que 88% para a

modulação ASK2, e maior do que 90% para a modulação ASK4.

No entanto, se uma estimativa da RSR estiver disponível antes do processo de classificação então a melhor solução para cada RSR pode ser utilizada. A tabela 6.20 mostra as configurações utilizadas para gerar o melhor subclassificador de acordo com a faixa estimada da RSR.

Tabela 6.16: Taxa de acerto para a classificação de modulações ASK usando kernel não linear (RSR_{TESTE} = 20dB)

	RSR (Treinamento=20dB e Teste=20dB)									
	C=0,01		C=0,1		C=1		C=10		C=100	
σ_{RBF}	ASK2	ASK4	ASK2	ASK4	ASK2	ASK4	ASK2	ASK4	ASK2	ASK4
0,01	98,72	85,58	98,78	85,65	96,45	87,75	96,03	86,70	96,28	86,97
0,1	94,90	8570	94,62	86,95	93,38	89,25	90,95	90,65	93,70	91,15
1	95,00	85,98	95,00	87,45	94,68	88,87	95,00	89,60	98,83	88,82
10	93,72	89,52	92,73	90,78	92,90	90,78	94,92	89,68	98,30	89,57
100	87,65	94,75	90,08	92,70	91,82	91,68	92,30	91,65	94,87	91,62

Tabela 6.17: Taxa de acerto para a classificação de modulações ASK usando kernel não linear (RSR_{TESTE} = 15dB)

	RSR (Treinamento=20dB e Teste=15dB)									
	C=0,01		C=0,1		C=1		C=10		C=100	
σ_{RBF}	ASK2	ASK4	ASK2	ASK4	ASK2	ASK4	ASK2	ASK4	ASK2	ASK4
0,01	99,90	80,95	99,95	81,43	99,77	83,57	99,55	83,57	99,78	83,98
0,1	94,25	83,47	93,85	85,22	92,75	86,47	91,55	86,42	93,43	86,63
1	94,15	84,83	93,85	86,37	93,72	86,77	94,17	86,72	98,43	86,50
10	91,75	89,40	90,25	90,18	90,43	90,20	93,10	89,43	98,23	87,90
100	84,07	91,20	85,03	90,83	86,88	90,80	88,67	90,82	92,42	90,80

Tabela 6.18: Taxa de acerto para a classificação de modulações ASK usando kernel não linear (RSR_{TESTE} = 10dB)

	RSR (Treinamento=20dB e Teste=10dB)									
	C=0,01		C=0,1		C=1		C=10		C=100	
σ_{RBF}	ASK2	ASK4	ASK2	ASK4	ASK2	ASK4	ASK2	ASK4	ASK2	ASK4
0,01	100	6,32	100	6,12	100	6,95	100	8,77	100	8,37
0,1	100	80,22	100	84,92	100	90,25	100	92,33	100	92,60
1	95,40	92,80	94,37	95,02	93,98	96,03	94,78	96,75	99,63	98,33
10	86,12	95,85	84,67	96,08	85,15	96,20	91,25	96,00	99,08	97,60
100	75,67	97,43	76,93	96,82	80,47	96,63	83,00	96,63	89,52	96,63

Tabela 6.19: Taxa de acerto para a classificação de modulações ASK usando kernel não linear (RSR_{TESTE} = 5dB)

	RSR (Treinamento=20dB e Teste=5dB)									
	C=0,01		C=0,1		C=1		C=10		C=100	
σ_{RBF}	ASK2	ASK4	ASK2	ASK4	ASK2	ASK4	ASK2	ASK4	ASK2	ASK4
0,01	100	0	100	0	100	0	100	0	100	0
0,1	100	0	100	0	100	0	100	0	100	0
1	99,97	84,17	100	87,37	100	90,05	100	89,92	100	82,32
10	75,68	93,37	74,02	93,82	75,55	93,75	84,75	92,57	99,63	87,32
100	61,53	95,02	62,27	95,00	65,97	94,97	69,45	94,75	88,00	93,07

Tabela 6.20:Configuração para o subclassificador ASK

	C	σ_{RBF}
RSR < 7,5	1	1
7,5 ≤ RSR < 12,5	100	1
12,5 ≤ RSR < 17,5	10	LINEAR
RSR < 17,5	100	

7. CONCLUSÕES

Este trabalho propôs como tese de doutorado a aplicação da técnica SVM à classificação de modulações digitais, dando origem ao classificador CMSVM. A metodologia empregada para a sua obtenção também constitui uma contribuição da tese, e pode ser utilizada para expandir o referido classificador com o objetivo de abranger um número maior de modulações. Além disso, o classificador SVM binário, denominado UPSVM, utilizado como ponto de partida para a criação do classificador CMSVM também foi proposto e analisado neste trabalho.

O UPSVM foi comparado com outros dois classificadores SVM de treinamento rápido, conhecidos por PSVM e SMO, em quatro casos de estudo. A partir dessa comparação, descrita no capítulo 4, pôde-se concluir que o método UPSVM foi o mais robusto dos algoritmos SVM estudados, pois:

- em todos os casos de estudo, o UPSVM obteve taxas de acerto similares ou maiores do que os outros dois métodos independentemente do valor de C; e
- a variação da taxa de acerto com o parâmetro C para o UPSVM foi sempre menor que a obtida para os demais classificadores estudados, independentemente do *kernel* utilizado (linear ou não linear).

O UPSVM também foi o mais rápido algoritmo para treinamento SVM, tendo sido muito mais rápido que o SMO mesmo quando este último apresentou uma complexidade computacional inferior à complexidade computacional do primeiro, conhecida como sendo $O(N^3)$ quando um *kernel* não linear é utilizado, tendo em vista que o número de acessos a memória também influencia no tempo de treinamento. Também é importante destacar que o algoritmo de treinamento para o UPSVM linear tem complexidade O(N).

O UPSVM apresenta um número de vetores de suporte similar ao PSVM porém maior do que o número de vetores de suporte obtidos pelo SMO. Este fato tem duas conseqüências:

- a variância dos parâmetros estimados para o hiperplano ótimo foi menor para o UPSVM do que para o SMO; e

- o tempo para a classificação de uma observação não utilizada no treinamento será maior para o UPSVM não linear do que para o SMO não linear, pois um número maior de vetores de suporte implica em um número maior de cálculos de *kernel*.

A primeira conseqüência reforça a idéia de robustez do UPSVM, enquanto que a segunda conseqüência é uma desvantagem do UPSVM que se aplica apenas aos casos em que um *kernel* não linear é utilizado. Caso um *kernel* linear seja utilizado, a máquina SVM obtida após o treinamento terá a mesma estrutura para qualquer um dos algoritmos de treinamento estudados neste trabalho.

Neste trabalho também foi demonstrado que o classificador PSVM (linear ou não linear) tem um desempenho ruim quando treinado utilizando pequenos valores do parâmetro C. Isto ocorre porque a inclusão do parâmetro *b* na função-objetivo a ser minimizada não diminui a margem de separação entre classes, mas introduz uma forte polarização do hiperplano ótimo. Além disso, foi mostrado que o procedimento incorreto utilizado para obter o PSVM não linear a partir da sua versão linear implicou na diminuição da taxa de acerto deste classificador para valores mais altos de C nos casos de estudo apresentados neste trabalho.

Demonstrou-se também que a analogia entre o PSVM e o estimador de Ridge não pode ser usada para ajudar a entender o porquê da inclusão do parâmetro *b* à função-objetivo que define o problema. Na verdade, o PSVM é uma evolução do ASVM, onde o parâmetro *b* foi introduzido de modo a possibilitar a obtenção de uma solução recursiva e direta para ω_0 e para *b*, ao contrário do que geralmente ocorre, em que ω_0 é calculado diretamente e *b* é obtido indiretamente.

No que se refere ao SMO, pôde-se concluir que existe um valor para C abaixo do qual o treinamento SMO leva à saturação dos elementos do vetor de multiplicadores de Lagrange, o que equivale a dizer que nenhum treinamento era necessário. Para valores elevados de C pode ser necessário um grande tempo de treinamento ou, no caso em que um *kernel* não linear foi utilizado, o classificador SMO pode ter seu desempenho diminuído devido a possível presença de *outliers* na matriz de treinamento.

Em outras palavras, o classificador SMO deve ser treinado utilizando um valor não muito grande e não muito pequeno para o parâmetro C. Os limites inferior e superior para o parâmetro C dependem da aplicação em questão.

O classificador CMSVM (que usa o UPSVM) foi empregado na classificação de modulações digitais (ASK2, ASK4, FSK2, FSK4, PSK2 e PSK4) e foi capaz de garantir taxas de acerto maiores do que 90%, mesmo se a RSR chegar a 5dB. Os resultados obtidos são melhores do que os resultados obtidos para o classificador CMADB para a mesma fonte de dados equiprovável real, que obteve taxas de acerto superiores apenas 50%, para uma RSR de 20dB.

Ao contrário do classificador CMADB, o classificador CMSVM permite a fácil inclusão ou exclusão de modulações (classes), uma vez que não é necessário introduzir ou retirar parâmetros (característica extraída dos dados de treinamento), basta treinar novos subclassificadores, mantendo os demais inalterados, ou retirar subclassificadores que não interessam mais.

Por último, é importante salientar que, como já era esperado, a classificação de modulações exponenciais em sinais corrompidos por ruído branco aditivo é mais fácil do que a classificação de modulações em amplitude. No entanto, o bom desempenho na classificação de modulações em freqüência só foi possível porque o estimador de Kay foi usado para a determinação da freqüência instantânea, no lugar do método BFD geralmente empregado.

Como temas para trabalhos futuros, sugere-se:

- o estabelecimento de um critério para definir o tipo de *kernel* e o valor ótimo do(s) parâmetro(s) que o definem e que são de livre escolha do usuário, para a obtenção do classificador UPSVM não linear;
- o estabelecimento de um critério para definir o menor número de observações a serem usadas no treinamento UPSVM não linear, a fim de minimizar o tempo gasto na fase de generalização;
- a obtenção de uma estimativa para a probabilidade de acerto após o treinamento de uma máquina SVM do tipo UPSVM não linear;
- a obtenção de uma estimativa para a probabilidade de acerto após o treinamento

de uma máquina SVM treinada a partir do PSVM ou SMO;

- o uso da técnica UPSVM em outras aplicações, como por exemplo na equalização de canal;
- o aumento do número de modulações que o classificador CMSVM pode identificar; e
- a implementação do CMSVM na prática, utilizando para tanto microprocessadores dedicados para processamento de sinais (DSPs), ou conjuntos de blocos lógicos reprogramáveis (FPGAs), ou uma mistura dois dois.

8. BIBLIOGRAFIA

[1] Mangasarian, O.L., Fung, G., "Proximal Support Vector Machine Classifiers". In: *Proceedings of the International Conference on Knowledge Discovery and Data Mining*, pp. 64-70, San Francisco, USA, Aug. 2001.

[2] Azzouz, E.E., Nandi, A.K., "Automatic Identification of Digital Modulation Types", *Signal Processing*, v. 47, n. 1, pp. 55-69, 1995.

[3] Haykin, S., *Neural Networks – A Comprehensive Foundation*, 2ª Ed., New York, Prentice Hall, 1999.

[4] Abe, S., "Analysis of Support Vector machines". In: *Proceedings of the IEEE Workshop on Neural Networks for Signal Processing*, pp. 89-98, Martign, Swizterland, Set. 2002.

[5] Vapnik V.N., *Statistical Learning Theory*, 1ª Ed., New York, John Wiley & Sons, 1998.

[6] Platt, J.C., Cristianini, N. ,Taylor, J.S., "Large Margin DAGS for Multiclass Classification". In: *Advances in Neural Information Processing System*, v. 12, MIT Press, pp. 547-553, 2000.

[7] Hsu, C-W., Lin, C-J, "A Comparison of Methods for Multiclass Support Vector Machines", *IEEE Transactions on Neural Networks*, v. 13, n. 2, pp. 415-425, 2002.

[8] Li, K-L., Tian, Z-F., Huang H-H., "A Novel Multiclass SVM Classifier Based on DDAG". In: *Proceedings of the 1st International Conference on Machine Learning and Cybernetics*, v. 3, pp. 1203-1207, Beijing, China, Nov. 2002.

[9] Friedman, J., *Another Approach to Polychotomous Classification*, Dpt. of Statistics, Stanford University, Stanford, USA http://www-stat.stanford.edu/~jhf/, 1996.

[10] Kner, S., Personnaz, L., Dreyfus, G., "Single-Layer Learning Revisited: A Stepwise Procedure for Building and Training a Neural Network". In: *Neurocomputing: Algorithms, Architectures and Applications*, v. F68, *NATO Advanced Study Institutes Series*, Springer-Verlag, pp. 41 - 50, 1990.

[11] UCI Repository of Machine Learning Databases, http://www.ics.uci.edu/~mlearn/MLRepository.html.

[12] Mangasarian, O.L., Musicant, D.R., "Active Support Vector Machine Classification". In: *Advances in Neural Information Processing Systems Conference*, v.13, MIT Press, pp. 577-583, 2001.

[13] Agarwal, D. K., "Shrinkage Estimator Generalizations of Proximal Support Vector Machines". In: *Proceedings of the International Conference on Knowledge Discovery and Data Mining*, pp. 173-182, Edmonton, Canada, Jul. 2002.

[14] Arthur, H., Robert, K., "Biased Estimation for Nonorthogonal Problems". *Technometrics*, v. 12, pp. 55-67, 1970.

[15] Tikhonov, A.N., Arsenin, V.A., *Solution of Ill-posed Problems*, 1ª Ed., Washington, Winston & Sons, 1977

[16] Golub, G.H., Van Loan, C.F., *Matrix Computations*, 3ª Ed., Baltimore, John Hopkins University Press, 1996.

[17] Papoulis, A., *Probability, Random Variables, and Stochastic Processes,* 3ª Ed., New York, McGraw-Hill,1991.

[18] Platt, J.C., "Fast Training of Support Vector Machines Using Sequential Minimal Optimization", In: B. Schölkopf, C.J.C. Burges and A.J.Smola (eds), 1ª Ed., *Advances in Kernel Methods – Support Vector Learning*, chapter 12, Cambridge, MIT Press.

[19] Hsue, S.Z., Soliman, S.S., "Automatic Modulation Classification Using Zero Crossing", *IEE Proceedings F: Radar & Signal Processing*, v. 137, n. 6, pp. 459-464, 1990

[20] Liedtke, F.F., "Computer Simulation of an Automatic Classification Procedure for Digitally Modulation Communication Signals with Unknown Parameters", *Signal Processing*, v. 6, pp. 311-313, 1984.

[21] Jondral, F., "Automatic Classification of High Frequency Signals", *Signal Processing*, v. 9, pp. 177-190, 1985.

[22] Matic, V., Lestar, B., Tadic, V., "The Use of a Digital Signal Processing for a Modulation Classification". In: *Proceedings of IEEE Mediterranean Electrotechnical Conference*, pp. 126-130, Cairo, Egypt, Mai. 2002.

[23] Nandi, A.K., Azzou, E.E., "Algorithms for Automatic Modulation Recognition of Communications Signals", *IEEE Transactions on Communications*, v. 46, n. 4, pp. 431-436, 1998.

[24] Al-Jalili, Y.O., "Identification Algorithm for Upper Sideband and Lower Sideband SSB Signals", *Signal Processing*, v. 42, pp. 207-213, 1995.

[25] Farrell, K.R., Mammone, R.J., "Modulation Classification Using a Neural Tree Network", In: *Proceedings of the IEEE Military Communication Conference*, pp. 1028-1032, Boston, USA, Out. 1993.

[26] Hsue, S.Z., Soliman, S.S., "Signal Classification Using Statistical Moments", *IEEE Transactions on Communications*, v. 40, n. 5, pp. 908-916, 1992.

[27] Dai, W., Wang, Y., Wang, J., "A Joint Estimation and Modulation Classification Using Second And Higher Statistics". In: *Proceedings of the IEEE Wireless Communications and Networking Conference*, pp. 155-158, Orlando, USA, Mar. 2002.

[28] Szczupak, J., Carvalho, B. C., "Um Novo Método de Classificação de Sinais quanto ao Tipo de Modulação". In: *Anais do XX Simpósio Brasileiro de Telecomunicações*, pp. 378-383, Rio de Janeiro, RJ, 2003.

[29] Polydoros, A., Huang, C., "Likelihood Methods for MPSK Modulation Classification", *IEEE Transactions on Communications*, v. 43, n. 234, pp. 1493-1504, 1995.

[30] Boiteau, D., Le Martret, C., "A General Maximum Likelihood Framework for Modulation Classification". In: *Proceedings of the International Conference on Acoustics, Speech and Signal Processing*, pp. 2165-2168, Seattle, USA, Mai. 1998.

[31] Polydoros, A., Kim, K., "On The Detection and Classification of Quadrature Digital Modulations In Broad-Band Noise", *IEEE Transactions on Communications*, v. 38, n 8, pp. 1199-1211, 1990.

[32] Polydoros, A., Chugg, K.M. e Long, C-S., "Combined Likelihood Power Estimation and Multiple Hypothesis Modulation Classification", *IEEE Proceedings of the 29th Asilomar Conference on Signals, Systems and Computers*, pp. 1137-1141, Pacific Groove, USA, 1995.

[33] Van Trees, H.L., *Detection, Estimation and Modulation Theory – Part I*, 1ª Ed., New York, John Wiley and Sons, 1968.

[34] Bastos, F.A.C. e Campos, M.R.L, "A New Method for Estimating The Instantaneous Frequency Based on Maximum Likelihood", *Proceedings of the International Conference on Acoustics, Speech and Signal Processing*, pp. II677 – II680, Istambul, Turkey, 2000.

APÊNDICE - IMPLEMENTAÇÃO EM MATLAB DO SMO

A rotina SMO.m, que executa o treinamento de uma máquina SVM usando o Matlab, é descrita abaixo. Suas sub-rotinas ExamineExample e TakeStep.m são listadas em seguida. Essas sub-rotinas não devem ser usadas diretamente.

Para usar a rotina SMO.m, deve-se informar a matriz de *kernel* para treinamento (Kernel), o vetor de classificação de cada observação de treinamento (d1), o valor do parâmetro de regularização (C1) e o limite máximo de iterações(LIMITE). A matriz K pode ser composta por um *kernel* linear ou não. Os parâmetros de saída são: o vetor de multiplicadores de Lagrange (alfa), a polarização (b), o número de iterações de treinamento (N_iteracoes) e a taxa de acerto no treinamento (txacerto). Caso tenha-se usado um *kernel* linear, o vetor de coeficientes pode ser calculado posteriormente pela eq.2.13, ou no Matlab, pelo comando: `w=X*diag(d1)*alfa;` , onde se supõe que a matriz de dados (1 observação por coluna) foi armazenada na variável X.

Listagem A.1: Programa de Matlab – smo.m

```
function[alfa,b,N_iteracoes,txacerto]=smo(Kernel,d1,C1,LIMITE)
%     Este programa executa o treinamento SMO (kenel linear ou nao)
% para uma maquina SVM.
%
% Sintaxe: [alfa0,b0,N_iteracoes,t,nflops]=smo(K,d,C,LIMITE)
% K ->  Kernel linear ou nao
% d -> Vetor de classificacao
% C -> Parametro de regularizacao
% Limite -> Nr. maximo de iteracoes
% alfa0 -> Vetor de multiplicadores de Lagrange otimo
% b0 -> polarizacao otima
% N_iteracoes-> Nr. de iteracoes executadas
% t -> Tempo gasto no treinamento
% nflops -> Nr. de flops executadas no treinamento
%
clear global X K d C sv Erro alfa b  w
```

```
global X K d C sv Erro alfa b  w
K=Kernel;d=d1;C=C1;
[N,M]=size(K);
Erro=-d;
alfa=zeros(N,1);
b=0;
ExamineAll=1;
N_iteracoes=1;
NKKTV=0; % Número de violações KKT
while (ExamineAll | NKKTV>0)&(N_iteracoes<LIMITE)
   sv=find((alfa~=0)&(alfa~=C));
   NKKTV=0;
   if ExamineAll
      for i=1:N
         NKKTV=NKKTV+ExamineExample(i); % i2=i
      end
      ExamineAll=0;
   else
      for i=1:length(sv)
         NKKTV=NKKTV+ExamineExample(sv(i));  % i2=sv(i);
      end
      if NKKTV==0
         ExamineAll=1;
      end
   end
   N_iteracoes=N_iteracoes+1;
end
if (N~=M)    % Provavelmente kernel linear
   dist=K'*alfa-b;
else    % Provavelemente kernel nao-linear
   dist=K*diag(d)*alfa-b;
end;
erro=dist.*d<0;
txacerto=1-sum(erro)/length(erro);
```

Listagem A.2: Programa de Matlab – ExamineExample.m

```
function resultado=ExamineExample(i2)
global X K d C sv Erro alfa b  w
resultado=0; tol=10^(-3);
r2=Erro(i2)*d(i2);
if  ( (r2<-tol) & (alfa(i2)<C) ) | ( (r2>tol) & (alfa(i2)>0) )
   % Houve violocao das condicoes KKT - 1a heuristica valida

   % 2a Heuristica
   if length(sv)>0
      if Erro(i2) >0
         [valor,i]=min(Erro(sv));
      else
         [valor,i]=max(Erro(sv));
      end
      if TakeStep(sv(i),i2)  % 1o caso (i1=sv(i))
         resultado=1;
         return;
      else
         [valor,k]=sort(rand(1,length(sv)));
         for i=1:length(sv)
            if TakeStep(sv(k(i)),i2)  % 2o caso (i1=sv(k(i)), k(i) aleato'rio)
               resultado=1;
               return;
            end
         end
      end
   end
   i1=find( (alfa==0) | (alfa==C) );  % Busca fora dos candidatos a sv
   if length(i1)>0
      [valor,k]=sort(rand(1,length(i1)));
      for i=1:length(i1)
         if TakeStep(i1(k(i)),i2)  % 3o caso (k(i) e' aleato'rio)
```

```
            resultado=1;
            return;
          end
        end
    end
end
```

Listagem A.3: Programa de Matlab – TakeStep.m

```
function resultado=TakeStep(i1,i2)
global X K d C sv Erro alfa b  w
resultado=0; tol2=10^(-3);
if i1==i2
   return;
end
s=d(i1)*d(i2);
if s ~= 1
   L=max([0,alfa(i2)-alfa(i1)]);
   H=min([C,C+alfa(i2)-alfa(i1)]);
else
   L=max([0,alfa(i2)+alfa(i1)-C]);
   H=min([C,alfa(i2)+alfa(i1)]);
end
if L==H
   return;
end
eta=2*K(i1,i2)-K(i1,i1)-K(i2,i2);
if eta<0
   a2=alfa(i2)-d(i2)*(Erro(i1)-Erro(i2))/eta;
   if a2<L
      a2=L;
   elseif a2>H
      a2=H;
```

```
    end
else
    gama=alfa(i1)+s*alfa(i2);
    v1 = Erro(i1)+d(i1)-alfa(i1)*d(i1)*K(i1,i1)-d(i2)*alfa(i2)*K(i1,i2)+b;
    v2 = Erro(i2)+d(i2)-alfa(i1)*d(i1)*K(i1,i2)-d(i2)*alfa(i2)*K(i2,i2)+b;
    Lobj = L*(1-s) - 0.5*K(i1,i1)*(gama-s*L)^2 - 0.5*K(i2,i2)*L^2 - s*K(i1,i2)*(gama-
s*L)*L - d(i1)*(gama-s*L)*v1-d(i2)*L*v2;
    Hobj = H*(1-s) - 0.5*K(i1,i1)*(gama-s*H)^2 - 0.5*K(i2,i2)*H^2 - s*K(i1,i2)*(gama-
s*H)*H - d(i1)*(gama-s*H)*v1-d(i2)*H*v2;
    if  Lobj > Hobj+eps
        a2=L;
    elseif  Lobj < Hobj-eps
        a2=H;
    else
        a2=alfa(i2);
    end
end
if a2 < tol2
    a2=0;
elseif a2 > C - tol2
    a2=C;
end
if abs(a2-alfa(i2))<eps*(a2+alfa(i2)+eps)
    return
end
a1=alfa(i1)+s*(alfa(i2)-a2);
% Calculo do parametro b
if (a2>0)&(a2<C)
    b_New=Erro(i2)+d(i1)*(a1-alfa(i1))*K(i1,i2)+d(i2)*(a2-alfa(i2))*K(i2,i2)+b;
else
    if (a1>0)&(a1<C)
        b_New=Erro(i1)+d(i1)*(a1-alfa(i1))*K(i1,i1)+d(i2)*(a2-alfa(i2))*K(i1,i2)+b;
    else
        b1=Erro(i1)+d(i1)*(a1-alfa(i1))*K(i1,i1)+d(i2)*(a2-alfa(i2))*K(i1,i2)+b;
```

```
        b2=Erro(i2)+d(i1)*(a1-alfa(i1))*K(i1,i2)+d(i2)*(a2-alfa(i2))*K(i2,i2)+b;
        b_New=(b1+b2)/2;
    end
end
% Atualizacao dos parametros Erro, b e alfa
Erro=Erro+d(i1)*(a1-alfa(i1))*K(:,i1)+d(i2)*(a2-alfa(i2))*K(:,i2)+(b-b_New);
b=b_New;
alfa(i1)=a1;
alfa(i2)=a2;
resultado=1;
```