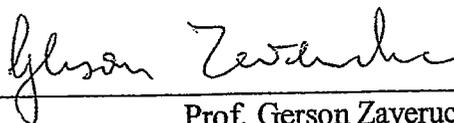


APRENDIZADO NÃO-PARAMÉTRICO DE FILTROS DE PARTÍCULAS EM
PREVISÃO DE SÉRIES TEMPORAIS

Aloísio Carlos de Pina

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 DE SISTEMAS E COMPUTAÇÃO.

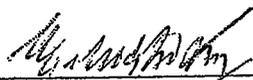
Aprovada por:



Prof. Gerson Zaverucha, Ph.D.



Prof. Alexandre Pinto Alves da Silva, Ph.D.



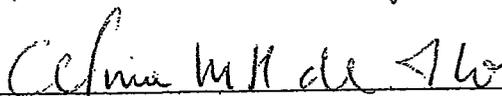
Prof. Hejio dos Santos Migon, Ph.D.



Prof.ª Marley Maria Bernardes Rebuzzi Vellasco, Ph.D.



Prof. Reinaldo Castro Souza, Ph.D.



Prof.ª Celina Miraglia Herrera de Figueiredo, Ph.D.

RIO DE JANEIRO, RJ - BRASIL

JUNHO DE 2008

PINA, ALOÍSIO CARLOS DE

Aprendizado Não-Paramétrico de Filtros de
Partículas em Previsão de Séries Temporais
[Rio de Janeiro] 2008

IX, 166 p. 29,7 cm (COPPE/UFRJ, D.Sc.,
Engenharia de Sistemas e Computação, 2008)

Tese - Universidade Federal do Rio de
Janeiro, COPPE

1. Filtros de Partículas
2. Estimação com Kernels
3. Séries Temporais

I. COPPE/UFRJ II. Título (série)

À minha Família.

A vitória e o sucesso de cada um de nós sempre representará a vitória e o sucesso para todos nós. Amo vocês.

AGRADECIMENTOS

Agradeço a todos aqueles que direta ou indiretamente tenham colaborado de alguma forma para a realização deste trabalho.

Agradeço ao apoio financeiro fornecido pelo CNPq durante todo o decorrer do trabalho. Enquanto existirem pessoas interessadas em financiar pesquisas para o avanço da tecnologia e conseqüente melhoria da sociedade, o homem continuará evoluindo.

Agradeço aos professores Alexandre Pinto Alves da Silva e Reinaldo Castro Souza por seus valiosos comentários e sugestões no exame de qualificação deste trabalho. Agradeço também ao professor Helio dos Santos Migon por suas sugestões nos estágios finais de desenvolvimento desta Tese.

Agradeço ao professor Gerson Zaverucha pela fundamental ajuda no desenvolvimento deste trabalho. Graças à sua experiência e inteligência, seus conselhos e orientações foram muito importantes para conclusão do trabalho. Além disso, durante todo o período que convivemos ele mostrou ser mais do que um simples orientador. Obrigado, meu amigo.

Agradeço à minha Família pelo apoio incondicional demonstrado em todos os momentos. É muito bom saber que existe alguém sempre torcendo e vibrando com nossas conquistas, rezando para que tudo transcorra sem problemas. Essa força foi e sempre será essencial para o término deste e de outros futuros trabalhos.

Acima de tudo e de todos, agradeço a DEUS pela chance de poder desenvolver este trabalho, completando mais um ciclo de minha vida com sucesso, tornando real mais um de meus sonhos. Muito obrigado.

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

APRENDIZADO NÃO-PARAMÉTRICO DE FILTROS DE PARTÍCULAS EM PREVISÃO DE SÉRIES TEMPORAIS

Aloísio Carlos de Pina

Junho/2008

Orientador: Gerson Zaverucha

Programa: Engenharia de Sistemas e Computação

A maioria dos algoritmos para a previsão de séries temporais faz suposições sobre os dados de forma a facilitar a previsão ou necessita da especificação de um modelo paramétrico a ser aprendido. Muitos problemas reais exibem alto grau de não-linearidade e em geral é difícil definir um modelo adequado. O objetivo deste trabalho é desenvolver um sistema automático que seja capaz de fazer a previsão de séries temporais sem a necessidade de qualquer outra informação e que não faça nenhuma suposição sobre os dados. O sistema proposto é baseado em filtros de partículas, utilizando estimação não-paramétrica de densidades com *kernels*, realizando o aprendizado dos *kernels* com um algoritmo similar ao EM. Outra proposta é o uso de curvas características de erro de regressão na avaliação comparativa de previsores de séries temporais e na seleção de componentes em comitês, de forma a maximizar a performance do conjunto. Uma extensa avaliação experimental é reportada e seus resultados mostram que a nova abordagem pode alcançar boa performance em domínios altamente não-lineares, com a vantagem de não necessitar de conhecimento especializado, permitindo seu uso por não-experts.

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

NONPARAMETRIC LEARNING OF PARTICLE FILTERS IN TIME SERIES
FORECASTING

Aloísio Carlos de Pina

June/2008

Advisor: Gerson Zaverucha

Department: Systems Engineering and Computer Science

Most of the algorithms for time series forecasting makes assumptions about the data, in order to facilitate prediction, or needs specifying a parametric model to be learned. Many real-world problems exhibit a high degree of non-linearity and usually it is difficult to define a proper model to be used. The objective of this work is to present a totally automated system for time series forecasting which does not need any other information but the series, and which does not make assumptions about the data. The proposed system is based on particle filters, utilizing nonparametric density estimation with kernels, performing the learning of kernels by using an EM-like algorithm. Another contribution is the use of regression error characteristic curves in the comparative evaluation of time series predictors and in the selection of components for ensembles, so that the global performance is maximized. An extensive experimental evaluation is reported and its results show that the new approach can achieve good performance in highly non-linear domains, with the advantage of no need of specialized knowledge, allowing its use for non-experts.

SUMÁRIO

1. INTRODUÇÃO	1
1.1. Objetivo	1
1.2. Organização da Tese	3
2. PREVISÃO DE SÉRIES TEMPORAIS	5
2.1. Modelos de Espaço de Estados	6
2.2. Filtros de Kalman	9
2.3. Filtro de Kalman Estendido	10
2.4. Filtros de Kalman com Pontos Sigma	11
2.4.1. Filtro de Kalman Unscented	13
2.4.2. Filtros de Kalman com Diferença Central	15
2.4.3. Formas Raiz-Quadrada do UKF e do CDKF	16
2.5. Outras Abordagens	16
3. FILTROS DE PARTÍCULAS	23
3.1. Amostragem por Importância	24
3.2. O Problema da Degeneração	26
3.3. Escolha da Densidade de Importância	28
3.4. Reamostragem	29
3.5. Regularização	31
3.6. Filtros Auxiliares	33
3.7. Rao-Blackwellização	35
3.8. Suavização	36
3.9. Filtro de Partículas com Somas de Gaussianas	39
3.10. Filtro de Partículas com Pontos Sigma	40

4. ESTIMAÇÃO DE DENSIDADES	43
4.1. O Algoritmo EM	45
4.2. Estimação Não-Paramétrica com Kernels	48
4.3. Seleção de Largura de Kernels	49
4.4. Kernels vs. Modelos de Misturas	53
5. COMITÊS DE ESTIMADORES	56
5.1. Redução de Bias e Variância	57
5.2. Métodos para a Construção de Comitês	58
5.3. Métodos para a Combinação de Estimadores	61
5.4. Métodos para a Seleção de Estimadores	62
5.5. Curvas Características de Erro de Regressão	63
6. FILTRO DE PARTÍCULAS NÃO-PARAMÉTRICO	66
6.1. Algoritmo Básico	77
6.2. Densidade de Importância	78
6.3. Passo de Reamostragem	80
6.4. Suavização	83
6.5. Função Kernel	85
6.6. Seletor de Largura de Kernel	88
6.7. Avaliação Experimental do Sistema Final.....	93
7. COMITÊS DE PREVISORES DE SÉRIES TEMPORAIS	108
7.1. Curvas REC para Seleção de Modelos em Comitês	108
7.1.1. Boosting para Regressão Usando Curvas REC	109
7.1.2. Avaliação Experimental	112
7.2. Comitês de Filtros de Kalman com Pontos Sigma	118
7.2.1. Comparando Filtros de Kalman através de Curvas REC	118
7.2.2. Comparando Comitês de Filtros de Kalman	120

7.3. Comitês de Filtros de Partículas	123
7.3.1. Comparando Filtros de Partículas através de Curvas REC	124
7.3.2. Comparando Comitês de Filtros de Partículas	127
7.3.3. Comparando Comitês de Previsores de Séries Temporais	128
7.4. Comitês de Filtros de Partículas Não-Paramétricos	130
8. MELHORANDO A PERFORMANCE ATRAVÉS DA COMBINAÇÃO DE ATRIBUTOS	133
6.1. Combinando Atributos	133
6.2. Avaliação Experimental	136
9. CONCLUSÕES	144
REFERÊNCIAS BIBLIOGRÁFICAS	147
APÊNDICE	163

1. INTRODUÇÃO

A previsão de séries temporais tem aplicações práticas em muitas áreas do conhecimento. Muitas vezes não se tem informação alguma a respeito da série temporal a ser analisada. Nesse caso existe a necessidade de razoáveis conhecimentos em estatística para a definição de um modelo paramétrico que seja adequado aos dados. Mesmo assim, pode ainda não existir um modelo paramétrico adequado ou a definição de um pode ser muito complexa. Existem vários métodos automáticos para a previsão de séries temporais. Entretanto, a maioria deles faz suposições sobre os dados de forma a facilitar a previsão. Muitos problemas reais exibem alto grau de não-linearidade e em geral tais suposições não condizem com a realidade. Portanto, seria de grande importância o desenvolvimento de um sistema automático que fosse capaz de fazer a previsão de séries temporais sem a necessidade de qualquer outra informação e que não fizesse nenhuma suposição sobre os dados.

1.1. Objetivo

O objetivo deste trabalho é desenvolver um sistema não-paramétrico para previsão de séries temporais que alcance uma performance comparável a de abordagens que se utilizam de modelos paramétricos pré-definidos. Tal sistema deve ser capaz de aprender todas as informações necessárias para fazer as predições tendo como entrada apenas a série temporal.

O sistema proposto é baseado em filtros de partículas (DOUCET *et al.*, 2001) para a previsão de séries temporais, utilizando estimação não-paramétrica de densidades com *kernels* (PARZEN, 1962) para representar as funções de transição de estados e observações, realizando o aprendizado dos *kernels* com um algoritmo similar ao EM.

Os filtros de partículas aproximam a densidade posterior dos estados sem que se faça qualquer suposição explícita sobre sua forma, eles portanto são a escolha mais adequada para serem usados em problemas não-lineares e não-Gaussianos. A estimação não-paramétrica das densidades com *kernels* elimina o *bias* introduzido pela especificação de uma densidade incorreta. Os *kernels* podem ser vistos como “parâmetros” cuja quantidade aumenta com o número de pontos de dados. Dessa forma, o algoritmo EM, que será discutido na Seção 4.1, pode ser utilizado para aprendê-los.

Outra contribuição desta Tese é introduzir o uso de curvas características de erro de regressão (BI e BENNETT, 2003) na avaliação comparativa de previsores de séries temporais e na seleção de componentes em comitês de estimadores (DIETTERICH, 1998), de forma a maximizar a performance do conjunto.

A análise de curvas características de erro de regressão (REC, *Regression Error Characteristic*) é uma técnica para a avaliação e comparação de modelos de regressão que permite a visualização da performance de muitas funções de regressão simultaneamente em um único gráfico. O uso de comitês aumenta a acurácia preditiva do sistema, diminuindo o *bias* e a variância dos resultados, podendo ser construídos de forma a otimizar os recursos computacionais disponíveis.

1.2. Organização da Tese

Os Capítulos do trabalho foram organizados da seguinte forma: no Capítulo 2 serão apresentados os modelos de espaço de estados, os algoritmos mais utilizados atualmente para a previsão de séries temporais e novas abordagens para o problema. No Capítulo 3 serão apresentados os filtros de partículas, métodos que podem ser aplicados a qualquer modelo de espaço de estados e que generalizam os tradicionais métodos de filtros de Kalman (KALMAN, 1960). No Capítulo 4 será abordada a estimação de densidades, com ênfase na estimação não-paramétrica de densidades com o uso de *kernels*, suas vantagens em relação a métodos paramétricos e aspectos práticos de sua utilização. Será apresentado também o algoritmo EM, amplamente usado no aprendizado de parâmetros em modelos paramétricos. No Capítulo 5 são apresentados os comitês de estimadores, que apresentam *bias* e variância menores que seus componentes funcionando individualmente. Também serão apresentadas técnicas de construção de comitês de estimadores, as formas mais comuns de combinação de resultados e formas de selecionar componentes para maximizar sua performance. Além disso será apresentada a técnica da análise de curvas características de erro de regressão, usada para a comparação de estimadores de valores contínuos. O Capítulo 6 é devotado à introdução do Filtro de Partículas Não-Paramétrico, novo algoritmo desenvolvido para a previsão de séries temporais. Primeiro é apresentado o algoritmo básico e as decisões de projeto, depois é reportada a avaliação experimental, comparando o algoritmo desenvolvido com vários outros descritos nos Capítulos 2 e 3. O Capítulo 7 é devotado à análise de comitês de previsores de séries temporais, utilizando curvas características de erro de regressão na comparação de modelos e seleção de componentes. No início do

capítulo é apresentado o algoritmo de *boosting* para regressão usando curvas REC, primeira utilização de análise REC na seleção de componentes em comitês de estimadores. Depois são analisados, através de curvas REC, a seleção de modelos de comitês de filtros de Kalman tradicionais e filtros de partículas, concluindo com a avaliação do algoritmo proposto como base de um comitê. Como última contribuição da Tese, no Capítulo 8 é apresentada uma abordagem para melhorar a performance de um algoritmo de regressão, o NBR (FRANK *et al.*, 2000), através da combinação de atributos, técnica que pode ser aplicada no contexto de previsão de séries temporais. Por fim, no Capítulo 9, são apresentadas as conclusões do trabalho, bem como as propostas para trabalhos futuros.

2. PREVISÃO DE SÉRIES TEMPORAIS

Uma série temporal é um conjunto de observações y_0, \dots, y_T de uma variável aleatória contínua indexada y_t , onde cada uma pertence a um tempo específico t . Nos últimos anos, representações de espaço de estados e as recursões de Kalman (KALMAN, 1960) associadas tiveram um profundo impacto na análise de séries temporais.

Uma classe extremamente rica de modelos para séries temporais, incluindo e indo bem além dos modelos clássicos utilizados, podem ser formulados como casos especiais do modelo geral de espaço de estados (BROCKWELL e DAVIS, 2002).

Modelos lineares formam uma extensa classe de modelos com parâmetros que variam com o tempo, úteis na modelagem de séries temporais (HARRISON e STEVENS, 1976). Entretanto, modelos lineares têm um número limitado de aplicações e não podem modelar a dinâmica complexa da maioria dos sistemas reais.

Modelos de espaço de estados não-lineares e não-Gaussianos são métodos mais poderosos para a modelagem probabilística de séries temporais e generalizam os métodos de filtragem de Kalman tradicionais.

Na próxima seção serão apresentados os modelos de espaço de estados e suas principais características. Na Seção 2.2 será feita uma introdução ao Filtro de Kalman, ou modelo de espaço de estados Gaussiano linear. Em seguida, na Seção 2.3 será apresentada a extensão do Filtro de Kalman para modelos não-lineares: o Filtro de Kalman Estendido. A Seção 2.4 aborda os Filtros de Kalman com Pontos Sigma, uma família de filtros que consegue propagar aproximações Gaussianas de forma mais eficiente que o Filtro de Kalman Estendido. Por fim, na Seção 4.5 serão apresentadas

algumas novas abordagens usadas na previsão de séries temporais com modelos de espaço de estados não-lineares e não-Gaussianos.

2.1. Modelos de Espaço de Estados

Um modelo de espaço de estados (BROCKWELL e DAVIS, 2002) para uma série temporal consiste de duas funções. A primeira, conhecida como função das observações, expressa a observação y_t como uma função de uma variável de estado x_t e de ruído w_t :

$$y_t = h(x_t, w_t) \quad t = 1, 2, \dots \quad (1)$$

A segunda função, chamada função de transição de estados, determina o estado x_{t+1} no tempo $t + 1$ em termos do estado anterior x_t e de um termo de ruído v_t :

$$x_{t+1} = f(x_t, v_t) \quad t = 1, 2, \dots \quad (2)$$

Uma série temporal tem uma representação de espaço de estados se existe um modelo de espaço de estados para ela especificado pelas funções de transição de estados e de observações.

É possível encontrar uma representação de espaço de estados para um grande número de modelos de séries temporais. Tais modelos tipicamente exibem complexas não-linearidades e distribuições não-Gaussianas (KANTZ e SCHREIBER, 1999).

Em modelos de espaço de estados não-lineares o problema de aprendizado está intimamente ligado ao problema de inferir a probabilidade dos espaços dadas as observações. Para realizar o aprendizado, é preciso antes fazer inferência no espaço de estados.

A densidade de transição dos estados $p(x_{t+1} | x_t)$ é totalmente especificada por f e pela distribuição do ruído no processo $p(v_t)$, enquanto que h e a distribuição de ruído nas observações $p(w_t)$ especificam completamente a verossimilhança das observações $p(y_t | x_t)$.

No *framework* Bayesiano, a densidade posterior filtrada $p(x_t | y_{1:t})$ dos estados dadas as observações $y_{1:t} = \{y_1, y_2, \dots, y_t\}$ constitui a solução para o problema de inferência probabilística e permite o cálculo de qualquer estimativa dos estados. O método ótimo para atualizar recursivamente a densidade posterior é dado pelo algoritmo de estimação Bayesiana recursiva. Essa abordagem primeiro projeta para frente no tempo a densidade posterior no tempo $t - 1$, $p(x_{t-1} | y_{1:t-1})$, usando o modelo de processo probabilístico, ou seja:

$$p(x_t | y_{1:t-1}) = \int p(x_t | x_{t-1})p(x_{t-1} | y_{1:t-1})dx_{t-1} \quad (3)$$

e então incorpora as observações atuais usando a verossimilhança para gerar a densidade posterior atualizada:

$$p(x_t | y_{1:t}) \propto p(y_t | x_t)p(x_t | y_{1:t-1}) \quad (4)$$

A estimativa filtrada $p(x_t | y_{1:t})$ pode ser vista como uma “mensagem” que é propagada ao longo do tempo, sendo modificada a cada transição de estados e atualizada a cada nova observação. Por isso ela é chamada *mensagem forward* (“para frente”).

Para obter melhores estimativas dos estados é comum usar *suavização*, que consiste em computar a distribuição posterior de um estado passado dadas todas as observações até o presente, $p(x_k | y_{1:t})$ para $1 \leq k \leq t$. Para isso, primeiro computa-se

$p(y_{k+1:t} | x_k)$, probabilidade conhecida como *mensagem backward* (“para trás”), através de um processo recursivo que volta no tempo:

$$p(y_{k+1:t} | x_k) = \int p(y_{k+1} | x_{k+1})p(y_{k+2:t} | x_{k+1})p(x_{k+1} | x_k)dx_{k+1} \quad (5)$$

A estimativa suavizada é então dada por:

$$\begin{aligned} p(x_k | y_{1:t}) &= p(x_k | y_{1:k}, y_{k+1:t}) \\ &\propto p(x_k | y_{1:k})p(y_{k+1:t} | x_k, y_{1:k}) \\ &= p(x_k | y_{1:k})p(y_{k+1:t} | x_k) \end{aligned} \quad (6)$$

A Equação (6) é portanto calculada pela multiplicação da mensagem *forward* pela mensagem *backward* no tempo k . O algoritmo que registra os resultados da filtragem de toda a seqüência de estados e depois volta computando as estimativas suavizadas é chamado *forward-backward* (BAUM e EGON, 1967).

Embora essa seja a solução recursiva ótima, usualmente só é tratável para sistemas Gaussianos lineares no caso em que a solução recursiva em forma-fechada é o conhecido filtro de Kalman. Entretanto, para a maioria dos sistemas reais, as integrais multidimensionais são intratáveis e soluções aproximadas são necessárias, tais como o filtro de Kalman estendido (JAZWINSKY, 1970), filtros de Kalman com pontos sigma (van der MERWE e WAN, 2003a) e filtros de partículas (DOUCET *et al.*, 2001).

2.2. Filtros de Kalman

O *filtro de Kalman* (KALMAN, 1960), também conhecido como *modelo de espaço de estados linear Gaussiano* (WEST e HARRISON, 1997), assume que a densidade posterior a cada instante de tempo é Gaussiana linear e portanto parametrizada por uma média e covariância.

Se a distribuição $p(x_{t-1} | y_{1:t-1})$ é Gaussiana e o modelo e transição $p(x_t | x_{t-1})$ é Gaussiano linear, então a distribuição da previsão um passo a frente dada por:

$$p(x_t | y_{1:t-1}) = \int p(x_t | x_{t-1}) p(x_{t-1} | y_{1:t-1}) dx_{t-1} \quad (7)$$

também é Gaussiana linear.

Se a distribuição da previsão um passo a frente $p(x_t | y_{1:t-1})$ é Gaussiana e a verossimilhança $p(y_t | x_t)$ é Gaussiana linear, então a densidade posterior:

$$p(x_t | y_{1:t}) \propto p(y_t | x_t) p(x_t | y_{1:t-1}) \quad (8)$$

é também Gaussiana.

O filtro de Kalman calcula de forma exata todos os termos dessas equações no caso linear e pode ser visto como um método eficiente para propagar analiticamente uma variável aleatória Gaussiana através da dinâmica de um sistema linear. Entretanto, para modelos não-lineares e não-Gaussianos sua performance em geral é bem fraca.

2.3. Filtro de Kalman Estendido

Sabe-se que para a maioria dos problemas reais, a recursão Bayesiana ótima é intratável. O *filtro de Kalman estendido* (EKF, *Extended Kalman Filter*) (JAZWINSKY, 1970) é uma solução aproximada que se tornou um dos algoritmos mais usados em muitas aplicações.

Se as funções de transição de estados e das observações são funções não-lineares, então uma linearização local pode fornecer uma descrição suficiente da não-linearidade. O EKF aproxima a distribuição dos estados com uma variável aleatória Gaussiana, que é então propagada através da linearização de “primeira-ordem” do sistema não-linear.

A linearização é feita através de uma expansão em série de Taylor ao redor da média da variável aleatória Gaussiana, ou seja:

$$y = g(x) = g(\bar{x} + \delta_x) \quad (9)$$

$$y = g(\bar{x}) + \nabla g \delta_x + \frac{1}{2} \nabla^2 g \delta_x^2 + \frac{1}{3!} \nabla^3 g \delta_x^3 + \dots \quad (10)$$

onde a variável aleatória de média zero δ_x tem a mesma covariância P_x de x . A média e a covariância usadas no EKF são dadas por:

$$\bar{y} \approx g(\bar{x}) \quad (11)$$

$$P_y^{LIN} = \nabla g P_x (\nabla g)^T \quad (12)$$

O maior problema do filtro de Kalman estendido é sua limitação na acurácia das médias e covariâncias propagadas resultante do método de linearização que é acurado apenas até a primeira ordem na expansão em série de Taylor. Essa linearização pode introduzir grandes erros nas verdadeiras média e covariância posteriores da

variável aleatória transformada, que podem comprometer a acurácia ou mesmo levar à divergência de um sistema de inferência baseado no EKF ou que use o EKF como componente, em especial quando os modelos são altamente não-lineares e os efeitos dos termos de ordem mais alta da expansão em série de Taylor se tornam significativos.

Além disso, o EKF sempre aproxima a densidade posterior por uma Gaussiana. Se a verdadeira densidade é não-Gaussiana, então pode ser que uma Gaussiana não a descreva bem. Em tais casos, métodos aproximados, como os filtros de partículas, apresentam melhor performance que o EKF.

2.4. Filtros de Kalman com Pontos Sigma

Os *filtros de Kalman com pontos sigma* (SPKF, *Sigma-Point Kalman Filters*) (van der MERWE e WAN, 2003a), uma família de filtros baseados em linearização estatística sem derivada, alcançam performances mais altas que o EKF em muitos problemas e são aplicáveis em áreas onde o EKF não pode ser usado.

Ao invés de linearizar a função não-linear através de uma expansão em séries de Taylor em um único ponto (usualmente a média da variável aleatória), os SPKF linearizam a função através de regressão linear entre s pontos, chamados *pontos sigma*, tirados a partir da distribuição à priori da variável aleatória, e fazem a avaliação da função não-linear desses pontos. Uma vez que essa técnica de aproximação estatística leva em consideração as propriedades estatísticas da variável aleatória a priori, o erro esperado resultante da linearização tende a ser menor que o de uma linearização com série de Taylor truncada.

A *transformada unscented* (JULIER e UHLMANN, 1996) é um método para calcular as estatísticas de uma variável aleatória que sofre uma transformação não-linear que se baseia no princípio de que é mais fácil aproximar uma distribuição de probabilidade que uma função não-linear arbitrária. Considere propagar uma variável aleatória n_x -dimensional x através de uma função não-linear arbitrária $g : R^{n_x} \mapsto R^{n_y}$ para gerar y :

$$y = g(x) \quad (13)$$

Assuma que x tem média \bar{x} e covariância P_x . Para calcular as estatísticas dos dois primeiros momentos de y usando a transformada *unscented*, primeiro, um conjunto de $(2n_x + 1)$ amostras ponderadas, ou *pontos sigma*, $S_i = \{W_i, \chi_i\}$ são escolhidos deterministicamente de forma que capturem completamente as verdadeiras médias e covariâncias da variável aleatória x . Um esquema de seleção que satisfaz esse requerimento é:

$$\begin{aligned} \chi_0 &= \bar{x} & W_0 &= k/(n_x + k) & i &= 0 \\ \chi_i &= \bar{x} + \left(\sqrt{(n_x + k)P_x} \right)_i & W_i &= 1/\{2(n_x + k)\} & i &= 1, \dots, n_x \\ \chi_i &= \bar{x} - \left(\sqrt{(n_x + k)P_x} \right)_i & W_i &= 1/\{2(n_x + k)\} & i &= n_x + 1, \dots, 2n_x \end{aligned} \quad (14)$$

onde k é um parâmetro de escala, $\left(\sqrt{(n_x + k)P_x} \right)_i$ é a i -ésima linha ou coluna da matriz raiz quadrada de $(n_x + k)P_x$ e W_i é o peso associado ao i -ésimo ponto, tal que:

$$\sum_{i=0}^{2n_x} W_i = 1 \quad (15)$$

Cada ponto sigma é então propagado através da função não-linear:

$$v_i = g(\chi_i) \quad i = 0, \dots, 2n_x \quad (16)$$

A média e covariância estimadas de y são computadas por:

$$\bar{y} = \sum_{i=0}^{2n_x} W_i \nu_i \quad (17)$$

$$P_y = \sum_{i=0}^{2n_x} W_i (\nu_i - \bar{y})(\nu_i - \bar{y})^T \quad (18)$$

Essas estimativas da média e covariância são acuradas até a segunda ordem da expansão de $g(x)$ em série de Taylor para qualquer função não-linear. Erros são introduzidos no terceiro momento e superiores, mas são escalados pela escolha do parâmetro k . Em comparação, o EKF somente calcula acuradamente a média e a covariância posterior até a primeira ordem e todos os momentos de ordem superior são truncados.

As formas com que o número e específicas localizações dos pontos sigma são escolhidos, bem como seus pesos de regressão correspondentes, diferenciam as variantes dos SPKF. A família SPKF é composta por quatro algoritmos: *filtro de Kalman unscented* (UKF), *filtro de Kalman com diferença central* (CDKF), a *forma raiz quadrada do filtro de Kalman unscented* (SR-UKF) e a *forma raiz quadrada do filtro de Kalman com diferença central* (SR-CDKF).

2.4.1. Filtro de Kalman Unscented

Ao contrário do EKF, o *filtro de Kalman unscented* (UKF) (JULIER *et al.*, 1995) não aproxima as funções de transição de estados e das observações não-lineares, ele usa os modelos não-lineares e aproxima apenas a distribuição posterior. No UKF, a densidade posterior é representada por uma Gaussiana, mas ela é especificada usando-se um conjunto mínimo de pontos escolhidos deterministicamente. Esses pontos

descrevem completamente as verdadeiras média e covariância da Gaussiana, e quando propagados através do modelo não-linear, capturam acuradamente a média e covariância posteriores até a segunda ordem para qualquer não-linearidade, com erros introduzidos na terceira ordem e superiores.

O UKF deriva a localização dos pontos sigma bem como seus pesos correspondentes de forma que os pontos sigma capturem as propriedades estatísticas mais importantes da variável aleatória a priori x . Isso é conseguido pela escolha de pontos de acordo com uma equação de restrições que é satisfeita pela minimização de uma função de custo, cujo propósito é incorporar características estatísticas de x que são desejáveis, mas que não precisam necessariamente ser satisfeitas. As informações estatísticas necessárias capturadas pelo UKF são o primeiro e o segundo momentos de $p(x)$. O número de pontos sigma necessários para fazer isso é $s = 2L + 1$, onde L é a dimensão de x . O conjunto resultante de pontos sigma e pesos utilizados pelo UKF é dado pela *transformada unscented escalada* (JULIER, 2000):

$$\begin{aligned}
 \chi_0 &= \bar{x} & W_0^{(m)} &= \frac{\lambda}{L + \lambda} & W_0^{(c)} &= \frac{\lambda}{L + \lambda} + (1 - \alpha^2 + \beta) \\
 \chi_i &= \bar{x} + \left(\sqrt{(L + \lambda)P_x} \right)_i & W_i^{(m)} &= W_i^{(c)} = \frac{1}{2(L + \lambda)} & i &= 1, \dots, L \\
 \chi_i &= \bar{x} - \left(\sqrt{(L + \lambda)P_x} \right)_i & W_i^{(m)} &= W_i^{(c)} = \frac{1}{2(L + \lambda)} & i &= L + 1, \dots, 2L
 \end{aligned} \tag{19}$$

onde $\lambda = \alpha^2(L + k) - L$ é um parâmetro de escala, α determina a dispersão dos pontos sigma ao redor da média \bar{x} , k é um parâmetro de escala secundário, β é um parâmetro escalar de grau de liberdade extra usado para incorporar qualquer conhecimento extra da distribuição a priori de x , $\left(\sqrt{(L + \lambda)P_x} \right)_i$ é a i -ésima linha ou coluna da matriz raiz quadrada ponderada da covariância P_x , $W_i^{(m)}$ e $W_i^{(c)}$ são, respectivamente, os pesos usados no cálculo da média e da covariância da densidade posterior.

2.4.2. Filtro de Kalman com Diferença Central

O *filtro de Kalman com diferença central* (CDKF) (ITO e XIONG, 2000) é outra implementação de SPKF, cuja formulação foi obtida pela troca das derivadas de primeira e segunda ordem calculadas analiticamente na expansão em séries de Taylor pelas diferenças divididas centrais avaliadas numericamente:

$$\nabla g \approx \frac{g(x + h\delta_x) - g(x - h\delta_x)}{2h} \quad (20)$$

$$\nabla^2 g \approx \frac{g(x + h\delta_x) + g(x - h\delta_x) - 2g(x)}{h^2} \quad (21)$$

O conjunto resultante de pontos sigma para o CDKF é também um conjunto de $(2L + 1)$ pontos deterministicamente tirados das estatísticas a priori de x :

$$\begin{aligned} \chi_i &= \bar{x} & W_i &= \frac{h^2 - L}{h^2} & i &= 0 \\ \chi_i &= \bar{x} + \left(\sqrt{h^2 P_x}\right)_i & W_i &= \frac{1}{2h^2} & i &= 1, \dots, L \\ \chi_i &= \bar{x} - \left(\sqrt{h^2 P_x}\right)_i & W_i &= \frac{1}{2h^2} & i &= L + 1, \dots, 2L \end{aligned} \quad (22)$$

Pesquisas (ITO e XIONG, 2000) mostraram que o CDKF teoricamente apresenta acurácia ligeiramente maior que a do UKF, mas na prática a diferença entre a performance de ambos é insignificante. Entretanto, tal como acontece com o UKF, o CDKF gera estimativas que são claramente superiores às calculadas pelo EKF. A vantagem do CDKF sobre o UKF é que ele usa apenas um parâmetro de escala, o tamanho do meio-passo da diferença central, em comparação com os três que o UKF utiliza (α, k, β) .

2.4.3. Formas Raiz-Quadrada do UKF e do CDKF

SR-UKF e SR-CDKF (van der MERWE e WAN, 2001) são formas raiz-quadrada numericamente eficientes do UKF e CDKF respectivamente. Ao invés de calcular a matriz raiz-quadrada da covariância dos estados a cada passo de tempo (uma operação muito custosa) para construir o conjunto de pontos sigma, essas formas propagam e atualizam a raiz-quadrada da covariância dos estados diretamente na forma fatorada de Cholesky, usando técnicas de álgebra linear. Isso também proporciona mais estabilidade numérica.

As formas raiz-quadrada dos SPKFs (SR-UKF e SR-CDKF) alcançam acurácia igual ou ligeiramente melhor quando comparados aos SPKF padrões. Além disso, elas apresentam custo computacional menor e uma estabilidade numérica consistentemente maior.

2.5. Outras Abordagens

Nos últimos anos, várias abordagens foram desenvolvidas para aprendizado de modelos de espaço de estados não-lineares e não-Gaussianos para a previsão de séries temporais.

Ghahramani e Roweis (1999) combinaram dois algoritmos clássicos, um da Estatística e o outro de Engenharia de Sistemas, para fazer o aprendizado de sistemas dinâmicos não-lineares estocásticos. Eles mostraram que usando o algoritmo do Suavizador de Kalman Estendido (EKS, *Extended Kalman Smoothing*) para a estimação

dos estados no passo E, com um modelo de aprendizado com função de base radial que permite a solução analítica do passo M, o algoritmo EM é capaz de aprender um modelo dinâmico não-linear a partir dos dados. O Suavizador de Kalman Estendido, algoritmo baseado no filtro de Kalman estendido com suavização das estimativas, foi usado para realizar inferência aproximada no passo E do EM. Para sistemas dinâmicos lineares com ruídos Gaussianos na evolução dos estados e nas observações, a densidade condicional $p(x_t | y_{1:T})$ é Gaussiana e o algoritmo recursivo para computar sua média e covariância é conhecido como Suavizador de Kalman. O Suavizador de Kalman é diretamente análogo ao algoritmo *forward-backward* para a computação da distribuição condicional dos estados escondidos em Modelos de Markov Escondidos (HMM, *Hidden Markov Models*) - a parte *forward* do Suavizador de Kalman é o Filtro de Kalman. O Suavizador de Kalman Estendido simplesmente aplica o Suavizador de Kalman a uma linearização local do sistema não-linear. Embora o EKS não possua as garantias teóricas de outros métodos, ele é largamente utilizado como método de inferência em sistemas dinâmicos não-lineares devido a sua simplicidade. O passo M do algoritmo EM reestima os parâmetros dadas as observações e as distribuições condicionais dos estados escondidos. Para o modelo proposto, os parâmetros definem as não-linearidades f e g . Duas complicações surgem no passo M: pode não ser computacionalmente viável reestimar completamente f e g ; e f e g têm que ser treinadas usando as estimativas incertas dos estados fornecidas pelo algoritmo EKS. Os autores mostram como contornar essas complicações escolhendo-se funções de base radial Gaussianas para modelar f e g , de forma que as integrais possam ser tratadas, permitindo que o passo M possa ser resolvido por intermédio de sistemas de equações lineares. Além do fato de que o algoritmo foi testado apenas com dados sintéticos, as desvantagens dessa abordagem

são que o EKS usa linearização local do sistema não-linear e a suposição de que a densidade posterior é uma Gaussiana multidimensional.

Andrews (2005) propôs um método que se assemelha ligeiramente ao algoritmo EM, envolvendo a iteração de um passo de inferência no espaço de estados, dados valores fixos de parâmetros, e um passo de estimação de parâmetros, dados os estados inferidos. De acordo com os autores, a abordagem proposta para aprender em modelos de espaço de estados não-lineares (NSSM, *Nonlinear State-Space Models*) difere de outras abordagens em diversos aspectos. Primeiro, uma vez que a inferência em NSSM é um pré-requisito essencial para o aprendizado, eles usam filtros de partículas para inferência, que obtêm resultados melhores que algumas das aproximações que têm sido usadas em outros trabalhos. Segundo, ao contrário da maioria das pesquisas com filtros de partículas para a inferência em sistemas não-lineares e não-Gaussianos, eles lidam primariamente com o problema de aprendizado, ou estimação de parâmetros, um problema não muito considerado na literatura. Isso é feito usando-se filtragem por partículas para inferência no passo E do algoritmo EM para estimação de máxima verossimilhança, analogamente ao método usado por Ghahramani e Roweis (1999) descrito anteriormente, sem a deficiência das suposições Gaussianas. Em outras abordagens onde o aprendizado é considerado, os parâmetros de sistemas não-lineares são algumas vezes tratados como variáveis latentes, e a distribuição conjunta dos estados e parâmetros, $p(x_t, \theta | y_{1:t})$, é estimada estendendo-se os métodos de inferência com filtros de partículas tradicionais. Uma vez que os valores dos parâmetros não variam com o tempo, esses filtros de partículas podem levar a estimativas altamente degeneradas das distribuições a posteriori dos parâmetros. Em outros casos, métodos baseados em gradiente para a estimação de máxima

verossimilhança foram utilizados em conjunção com estimativas baseadas em amostras da verossimilhança. Finalmente, os NSSM têm observações discretas e contínuas, e isso torna o NSSM diretamente comparável a um HMM (*Hidden Markov Model*) com um número infinito de estados. O autor fornece um sistema dinâmico não-linear em particular como caso de estudo, entretanto ele afirma que a forma é geral o suficiente de maneira que os métodos aplicados a ele podem ser facilmente estendidos para outros sistemas. O método de aprendizado Bayesiano proposto é um método de amostragem com cadeia de Markov que irá amostrar a partir da posterior conjunta das trajetórias do espaço de estados e parâmetros dos mapeamentos dinâmico e observacional. Ele se baseia em um método de amostragem com filtro de partículas para as trajetórias do espaço de estados. Uma desvantagem do método proposto é que a inferência é realizada com filtros de partículas convencionais e portanto a função de transição de estados e a função das observações devem ser definidas.

Thrun *et al.* (1999) propuseram um algoritmo de aprendizado para modelos de Markov escondidos não-paramétricos com espaços de estados e observações contínuos, chamado modelos escondidos de Markov Monte-Carlo (*MCHMM, Monte-Carlo Hidden Markov Models*). Todas as densidades de probabilidades necessárias são aproximadas usando amostras, junto com árvores de densidades geradas a partir de tais amostras. Árvores de densidades (OMOHUNDRO, 1991) são usadas porque a estimativa suavizada da distribuição dos estados é o produto normalizado de duas densidades, as mensagens *forward* e *backward*, cada uma representada por seu próprio conjunto de amostras, e não é trivial obter uma aproximação desse produto usando conjuntos de amostras. MCHMMs transformam conjuntos de amostras em árvores de densidades que efetivamente “generalizam” distribuições discretas em distribuições contínuas. Cada nó

em uma árvore de densidade é associado a um subespaço híper-retangular de $\text{dom}(f)$, denotado por V_i para o i -ésimo nó. Inicialmente, todas as amostras são associadas ao nó raiz, que cobre o domínio inteiro de f . A árvore cresce dividindo-se cada nó i sempre que duas condições são satisfeitas: pelo menos \sqrt{N} amostras caem em V_i ; e sua profundidade, isto é, sua distância ao nó raiz, não excede $\lfloor 0,25 \log_2 N \rfloor$. Se um nó é dividido, seu híper-retângulo é dividido ao longo de sua maior dimensão, em dois híper-retângulos de igual tamanho. Uma versão Monte Carlo do algoritmo EM é empregada para aprender modelos a partir dos dados. No passo E, *likelihood-weighting* (KANAZAWA *et al.*, 1995) é aplicado para as projeções *forward* e *backward*, usando as aproximações em árvore da priori dos estados, da função de transição de estados e da função das observações. As densidades suavizadas são obtidas multiplicando-se os fatores de importância de um dos conjuntos de amostras com seus valores de densidade na árvore induzidos a partir do outro conjunto de amostras. No passo M, o novo modelo é obtido gerando-se amostras a partir da distribuição inicial dos estados, distribuições de pares de estados consecutivos (para a função de transição) e distribuições dos estados com suas observações correspondentes (para a função das observações). Árvores são então induzidas a partir desses conjuntos de amostras. As árvores para a função de transição de estados e para a função das observações representam distribuições conjuntas, e não distribuições condicionais. Logo, ao usar essas árvores, elas devem ser condicionadas a um estado. As desvantagens do método são o uso do algoritmo *likelihood-weighting*, que sofre degradação na performance com o aumento no número de observações e o uso de árvores de densidades, que pode não ser o mais adequado. Os autores apresentam resultados empíricos obtidos em um domínio de reconhecimento de

sinais, a fim de prover uma primeira evidência da utilidade de MCHMMs na prática, mas desde então o sistema não foi avaliado em outros domínios.

Uma outra abordagem muito popular para problemas de previsão de séries temporais é o uso de redes neurais, sendo que os modelos mais amplamente utilizados são os baseados em redes neurais *feedforward* com algoritmo de aprendizado de retropropagação. Eles consistem em aproximar a função f por uma rede neural *feedforward* multicamadas. Considerando o vetor (x_k, \dots, x_{k-d}) como o k -ésimo padrão de entrada da rede, o valor previsto por um modelo neural pode ser escrito como $x_{k+1} = f(x_k, \dots, x_{k-d}, W_1)$, onde W_1 é o conjunto de pesos do modelo neural, que é obtido usando o algoritmo de retropropagação (RUMELHART *et al.*, 1986). A atualização do conjunto de parâmetros é baseada no erro local entre os valores medidos e previstos. Além das redes neurais *feedforward*, as redes neurais parcialmente recorrentes também são muito utilizadas na prática, devido aos seus estados internos, que as tornam apropriadas à previsão de séries temporais. Tais estados funcionam como memória de curto prazo e são capazes de representar informação sobre as entradas anteriores (STAGGE e SENHO, 1997). Essa informação tem um importante papel quando a meta é a previsão a longo prazo. Redes parcialmente recorrentes são redes de perceptrons multicamadas nas quais algumas conexões recorrentes são introduzidas. Geralmente, elas contém um grupo especial de neurônios na camada de entrada, chamados de neurônios de contexto ou neurônios de estado. Assim, na camada de entradas de redes parcialmente recorrentes existem dois tipos distintos de neurônios, aqueles que atuam como entrada em si, recebendo os sinais externos, e os neurônios de contexto, que recebem valores de saída de uma das camadas atrasadas em um passo. Os modelos de redes neurais recorrentes mais conhecidos são a rede de Jordan e a rede de Elman.

A rede neural proposta por Jordan (1986) tem como característica que os neurônios de contexto recebem uma cópia dos neurônios de saída e deles mesmos, isto é, a rede de Jordan contém tantos neurônios de contexto quanto neurônios de saída. As conexões recorrentes da camada de saída para os neurônios de contexto têm um parâmetro associado, μ , que em geral recebe um valor constante positivo e menor que um. Para a previsão de séries temporais, a rede terá um neurônio de saída que representa o valor previsto da série em tempos futuros. Portanto, a rede terá somente um neurônio de contexto e sua ativação no instante t é dada por $c_t = \mu c_{t-1} + x_{t-1}$, onde x_{t-1} é a saída da rede no instante $t-1$. O resto das ativações da rede são calculadas como em um perceptron multicamadas, basta considerar como vetor de entrada a concatenação das ativações de entrada externas e as ativações do neurônio de contexto. O neurônio de contexto acumula a saída da rede em todos os instantes anteriores e o valor do parâmetro μ determina a sensibilidade do neurônio de contexto para reter tal informação.

A rede de Elman (1990) tem como característica que os neurônios de contexto recebem uma cópia dos neurônios escondidos da rede e essas conexões não são associadas a qualquer parâmetro. Portanto existem tantos neurônios de contexto quanto neurônios escondidos na rede. Como acontece com a rede de Jordan, o resto das ativações são calculadas como em um perceptron multicamadas, considerando a concatenação de entradas externas e neurônios de contexto como vetor de entrada da rede.

3. FILTROS DE PARTÍCULAS

Filtros de partículas (DOUCET *et al.*, 2001) são métodos de Monte Carlo seqüenciais baseados na representação das densidades de probabilidade através de pontos de massa (ou “partículas”), que podem ser aplicados a qualquer modelo de espaço de estados e que generalizam os tradicionais métodos de filtros de Kalman. Essa abordagem é conhecida também como *filtro bootstrap*, *algoritmo de condensação*, *interação de aproximações por partículas*, e *sobrevivência do mais adequado*.

A idéia chave é representar a função de densidade posterior requerida por um conjunto de amostras aleatórias com pesos associados (chamadas *partículas*) e computar estimativas baseado nessas amostras e pesos. Quando o número de amostras é bem grande, essa caracterização é uma representação equivalente à descrição funcional usual da função de densidade posterior.

Na próxima seção, o processo de amostragem por importância será descrito em detalhes. Logo após, será abordado o problema da degeneração, inerente aos filtros de partículas, e em seguida serão discutidos os dois métodos mais comuns usados para reduzir seus efeitos: a escolha de uma densidade de importância apropriada e o uso de reamostragem. Depois, será apresentado o método da regularização, desenvolvido para evitar o empobrecimento das amostras, problema introduzido pela utilização de reamostragem. O uso de filtros auxiliares será abordado na Seção 3.6 e a técnica de Rao-Blackwellização será abordada na Seção 3.7. Por fim serão discutidos os filtros de partículas mais avançados em uso atualmente: o filtro de partículas com misturas de Gaussianas e os filtros de partículas com pontos sigma, estes últimos pertencentes à família de filtros de Kalman com pontos sigma.

3.1. Amostragem por Importância

Em um filtro de partículas, as partículas formam um conjunto $\{x_{0:t}^i, w_t^i\}_{i=1}^N$ que caracteriza a *pdf* posterior $p(x_{0:t} | y_{1:t})$, onde $\{x_{0:t}^i, i = 0, \dots, N\}$ é um conjunto de pontos de suporte com pesos associados $\{w_t^i, i = 1, \dots, N\}$, $x_{0:t} = \{x_j, j = 0, \dots, t\}$ é o conjunto de todos os estados até o tempo t e $y_{1:t} = \{y_j, j = 1, \dots, t\}$ é o conjunto de observações até o tempo t . Os pesos são normalizados de forma que:

$$\sum_{i=1}^N w_t^i = 1 \quad (23)$$

Então, a densidade posterior no tempo t pode ser aproximada por:

$$p(x_{0:t} | y_{1:t}) \approx \sum_{i=1}^N w_t^i \delta(x_{0:t} - x_{0:t}^i) \quad (24)$$

Essa é uma aproximação discreta ponderada para a verdadeira densidade posterior $p(x_{0:t} | y_{1:t})$. Os pesos são escolhidos usando o princípio da *amostragem por importância* (DOUCET *et al.*, 2001): suponha que $p(x) \propto \pi(x)$ é uma densidade de probabilidade a partir da qual é difícil amostrar, mas $\pi(x)$ pode ser avaliada. Sejam $x^i \sim q(x)$, $i = 1, \dots, N$ amostras que podem ser facilmente obtidas a partir da densidade $q(\cdot)$, chamada *densidade de importância*. Então, uma aproximação ponderada para a densidade $p(\cdot)$ é dada por:

$$p(x) \approx \sum_{i=1}^N w^i \delta(x - x^i) \quad (25)$$

onde:

$$w^i \propto \frac{\pi(x^i)}{q(x^i)} \quad (26)$$

é o peso normalizado da i -ésima partícula.

Portanto, se as amostras $x_{0:t}^i$ forem tiradas de uma densidade de importância $q(x_{0:t}^i | y_{1:t})$, os pesos serão dados por:

$$w_t^i \propto \frac{p(x_{0:t}^i | y_{1:t})}{q(x_{0:t}^i | y_{1:t})} \quad (27)$$

Se a densidade de importância for fatorada de forma que:

$$q(x_{0:t} | y_{1:t}) = q(x_t | x_{0:t-1}, y_{1:t})q(x_{0:t-1} | y_{1:t-1}) \quad (28)$$

e $p(x_{0:t}^i | y_{1:t})$ for fatorado da seguinte forma:

$$\begin{aligned} p(x_{0:t} | y_{1:t}) &= \frac{p(y_t | x_{0:t} | y_{1:t-1})p(x_{0:t} | y_{1:t-1})}{p(y_t | y_{1:t-1})} \\ &= \frac{p(y_t | x_{0:t} | y_{1:t-1})p(x_t | x_{0:t-1} | y_{1:t-1})}{p(y_t | y_{1:t-1})} \times p(x_{0:t-1} | y_{1:t-1}) \\ &= \frac{p(y_t | x_t)p(x_t | x_{t-1})}{p(y_t | y_{1:t-1})} \times p(x_{0:t-1} | y_{1:t-1}) \\ &\propto p(y_t | x_t)p(x_t | x_{t-1})p(x_{0:t-1} | y_{1:t-1}) \end{aligned} \quad (29)$$

Substituindo as Equações (28) e (29) na Equação (27), a equação de atualização de pesos é então dada por:

$$\begin{aligned} w_t^i &\propto \frac{p(y_t | x_t^i)p(x_t^i | x_{t-1}^i)p(x_{0:t-1}^i | y_{1:t-1})}{q(x_t^i | x_{0:t-1}^i, y_{1:t})q(x_{0:t-1}^i | y_{1:t-1})} \\ &= w_{t-1}^i \frac{p(y_t | x_t^i)p(x_t^i | x_{t-1}^i)}{q(x_t^i | x_{0:t-1}^i, y_{1:t})} \end{aligned} \quad (30)$$

Normalmente, apenas a estimativa filtrada de $p(x_t | y_{1:t})$ é requerida a cada instante de tempo. Logo, a densidade posterior filtrada $p(x_t | y_{1:t})$ pode ser aproximada por:

$$p(x_t | y_{1:t}) \approx \sum_{i=1}^N w_t^i \delta(x_t - x_t^i) \quad (31)$$

e a atualização dos pesos é dada por:

$$w_t^i \propto w_{t-1}^i \frac{p(y_t | x_t^i) p(x_t^i | x_{t-1}^i)}{q(x_t^i | x_{t-1}^i, y_t)} \quad (32)$$

Pode ser mostrado que quando $N \rightarrow \infty$, essa aproximação tende ao valor verdadeiro da densidade posterior.

As várias versões de filtros de partículas propostas na literatura podem ser vistas como casos especiais do algoritmo padrão. Esses casos especiais podem ser derivados a partir do algoritmo padrão através da escolha apropriada da densidade de importância, inclusão de um passo de reamostragem e outras técnicas, algumas das quais serão mostradas nas próximas seções.

3.2. O Problema da Degeneração

Um problema inerente ao filtro de partículas é o fenômeno da degeneração, onde após poucas iterações todas as partículas têm peso insignificante, exceto uma. Já foi mostrado que a variância dos pesos de importância pode apenas aumentar com o

passar do tempo e assim é impossível evitar o fenômeno da degeneração. Essa degeneração implica que um grande esforço computacional é devotado na atualização de partículas cujas contribuições para a aproximação de $p(x_t | y_{1:t})$ é quase zero. Uma medida adequada da degeneração do algoritmo é o tamanho efetivo da amostra N_{ef} definido por:

$$N_{ef} = \frac{N}{1 + Var(w_t^{*i})} \quad (33)$$

onde:

$$w_t^{*i} = \frac{p(x_t^i | y_{1:t})}{q(x_t^i | x_{t-1}^i, y_t)} \quad (34)$$

é referido como o “peso verdadeiro”. Esse valor não pode ser calculado de forma exata, mas uma estimativa do N_{ef} pode ser obtida por (LIU e CHEN, 1998):

$$\hat{N}_{ef} = \frac{1}{\sum_{i=1}^N (w_t^i)^2} \quad (35)$$

onde w_t^i é o peso normalizado obtido usando a Equação (32). Deve-se notar que $N_{ef} \leq N$, e um N_{ef} pequeno indica severa degeneração. Claramente, o problema da degeneração é um efeito indesejável em filtros de partículas. A abordagem de força bruta para reduzir seus efeitos é usar um N muito grande. Isso é freqüentemente impraticável, portanto conta-se com dois outros métodos: uma boa escolha da densidade de importância e o uso de reamostragem. Ambos serão vistos a seguir.

3.3. Escolha da Densidade de Importância

Um dos métodos para reduzir os efeitos da degeneração é a escolha de uma boa densidade de importância, minimizando $Var(w_t^{*i})$ de forma que N_{ef} seja maximizado. A função de densidade de importância ótima que minimiza a variância dos pesos reais condicionada a x_{t-1}^i e y_t foi mostrada ser (DOUCET, 1998):

$$\begin{aligned} q(x_t | x_{t-1}^i, y_t) &= p(x_t | x_{t-1}^i, y_t) \\ &= \frac{p(y_t | x_t | x_{t-1}^i) p(x_t | x_{t-1}^i)}{p(y_t | x_{t-1}^i)} \end{aligned} \quad (36)$$

Substituindo a Equação (36) na Equação (32), tem-se:

$$\begin{aligned} w_t^i &\propto w_{t-1}^i p(y_t | x_{t-1}^i) \\ &= w_{t-1}^i \int p(y_t | x_t') p(x_t' | x_{t-1}^i) dx_t' \end{aligned} \quad (37)$$

Essa densidade de importância é ótima uma vez que para um dado x_{t-1}^i , w_t^i tem o mesmo valor, qualquer que seja a amostra retirada a partir de $q(x_t | x_{t-1}^i, y_t)$, e portanto, condicionado em x_{t-1}^i , $Var(w_t^{*i}) = 0$.

O uso da densidade de importância ótima apresenta duas grandes dificuldades: requer que se possa amostrar a partir de $p(x_t | x_{t-1}^i, y_t)$ e avaliar a integral mostrada na Equação (37). Entretanto, é possível construir aproximações subótimas, dentre as quais a priori de transição de estados é a mais usada, pois é intuitiva e simples de implementar:

$$q(x_t | x_{t-1}^i, y_t) = p(x_t | x_{t-1}^i) \quad (38)$$

e portanto os pesos são fornecidos por:

$$w_t^i \propto w_{t-1}^i p(y_t | x_t^i) \quad (39)$$

3.4. Reamostragem

Uma forma de reduzir os efeitos da degeneração é usar reamostragem sempre que uma degeneração significativa for constatada, ou seja, quando o número efetivo de amostras, dado pela Equação (35), cai abaixo de algum valor limiar. A idéia básica da reamostragem é eliminar partículas que tem pesos muito pequenos e se concentrar nas partículas com grande peso. O passo de reamostragem consiste em gerar um novo conjunto de partículas pela reamostragem com reposição, N vezes, a partir da representação aproximada discreta de $p(x_t | y_{1:t})$, dada por:

$$p(x_t | y_{1:t}) \approx \sum_{i=1}^N w_t^i \delta(x_t - x_t^i) \quad (40)$$

A amostra resultante é de fato uma amostra *i.i.d.* da densidade discreta (40), e portanto os pesos são resetados para $w_t^i = 1/N$.

O filtro de *amostragem por importância com reamostragem* (*SIR, Sampling Importance Resampling*) proposto em (GORDON *et al.*, 1993) é o exemplo mais simples de filtro de partículas com passo de reamostragem.

O algoritmo *SIR* pode ser facilmente derivado a partir do algoritmo padrão através da escolha apropriada da densidade de importância, que é escolhida como a densidade a priori $p(x_t | x_{t-1}^i)$, e da inclusão do passo de reamostragem, que é aplicado a cada instante de tempo.

A escolha da priori como densidade de importância implica que são necessárias amostras a partir de $p(x_t | x_{t-1}^i)$. Uma amostra $x_t^i \sim p(x_t | x_{t-1}^i)$ pode ser conseguida primeiro gerando-se uma amostra do ruído no processo v_{t-1}^i e setando $x_t^i = f_t(x_{t-1}^i, v_{t-1}^i)$. Para essa escolha de densidade de importância em particular, é evidente que os pesos são dados por:

$$w_t^i \propto w_{t-1}^i p(y_t | x_t^i) \quad (41)$$

Entretanto, considerando que a reamostragem é aplicada a cada instante de tempo, tem-se que $w_{t-1}^i = 1/N \forall i$, portanto:

$$w_t^i \propto p(y_t | x_t^i) \quad (42)$$

Os pesos dados por essa proporcionalidade são normalizados antes do estágio de reamostragem.

Como a densidade de amostragem por importância para o filtro *SIR* é independente da medida y_t , o espaço de estados é explorado sem qualquer conhecimento das observações atuais. Portanto esse filtro pode ser ineficiente e é sensível a pontos espúrios. Além disso, como a reamostragem é aplicada a cada iteração, isso pode resultar em rápida perda de diversidade das partículas. Entretanto, o método *SIR* tem a vantagem de que os pesos são facilmente calculados e a densidade de importância pode ser facilmente amostrada.

Embora o passo de reamostragem reduza os efeitos do problema da degeneração, seu uso introduz outro problema, que é a perda de diversidade entre as partículas, uma vez que as partículas com pesos grandes são escolhidas muitas vezes. Conhecido como *empobrecimento das amostras*, esse problema é grave no caso de pouco ruído no processo, onde em poucas iterações todas as partículas passam a ter o

mesmo valor. Além disso, a perda de diversidade faz com que estimativas suavizadas degenerem. Uma abordagem muito utilizada para evitar o empobrecimento das amostras é o uso de regularização (MUSSO *et al.*, 2001).

3.5. Regularização

O passo de reamostragem é um método para reduzir o problema da degeneração, que é inerente aos filtros de partículas. Entretanto, a reamostragem em troca introduz o problema da perda de diversidade entre as partículas, devido ao fato de que as amostras são geradas de uma distribuição discreta ao invés de uma contínua. Se esse problema não for tratado apropriadamente, pode levar ao “colapso” das partículas, que é um caso severo de empobrecimento das amostras onde todas as partículas ocupam o mesmo ponto no espaço de estados, fornecendo uma representação pobre da densidade posterior. Um filtro de partículas modificado, conhecido como filtro de *partículas regularizado* (*RPF*, *Regularized Particle Filter*) (MUSSO *et al.*, 2001), foi proposto como uma solução em potencial para esse problema.

O *RPF* é idêntico ao filtro *SIR*, exceto pelo estágio de reamostragem. A regularização consiste em reamostrar a partir de uma aproximação contínua da densidade posterior $p(x_t | y_{1:t})$, ao invés de reamostrar a partir da aproximação discreta dada pela Equação (40). Mais especificamente, no *RPF* as amostras são tiradas a partir de uma aproximação com *kernels*:

$$p(x_t | y_{1:t}) \approx \frac{1}{h^{n_x}} \sum_{i=1}^N w_t^i K\left(\frac{x_t - x_t^i}{h}\right) \quad (43)$$

onde $K(\cdot)$ é a função *kernel*, $h > 0$ é a largura do *kernel*, n_x é a dimensão do vetor de estados x , e w_t^i , $i = 1, \dots, N$ são os pesos normalizados. A estimação de densidades com *kernels* será discutida em mais detalhes no Capítulo 4.

A função *kernel* $K(\cdot)$ e a largura h são escolhidos de forma a minimizar o erro médio quadrático integrado (*MISE*, *Mean Integrated Squared Error*) entre a verdadeira densidade posterior e a correspondente representação empírica regularizada dada pela Equação (43), que é definido como:

$$MISE(\hat{p}) = E \left[\int [\hat{p}(x_t | y_{1:t}) - p(x_t | y_{1:t})]^2 dx_t \right] \quad (44)$$

onde $\hat{p}(\cdot | \cdot)$ denota a aproximação para $p(x_t | y_{1:t})$ dada pelo lado direito da Equação (43). No caso especial em que todas as amostras têm o mesmo peso, a escolha ótima do *kernel* é o *kernel Epanechnikov*:

$$K = \begin{cases} \frac{n_x + 2}{2c_{n_x}} (1 - \|x\|^2), & \text{se } \|x\| < 1 \\ 0, & \text{caso contrário} \end{cases} \quad (45)$$

onde c_{n_x} é o volume da hipersfera unitária em R^{n_x} . Além disso, quando a densidade é Gaussiana com matriz de covariância unitária, a escolha ótima para largura do *kernel* é:

$$h = AN^{\frac{1}{(n_x+4)}} \quad (46)$$

onde:

$$A = [8c_{n_x}^{-1} (n_x + 4) (2\sqrt{\pi})^{n_x}]^{\frac{1}{(n_x+4)}} \quad (47)$$

Embora os resultados das Equações (45) e (46) sejam ótimos somente no caso especial de partículas com pesos iguais e densidade Gaussiana, esses resultados podem ainda ser usados no caso geral para obter um filtro subótimo.

Em termos de complexidade, o *RPF* é comparável ao *SIR* uma vez que só requer N gerações adicionais a partir do *kernel* $K(\cdot)$ a cada passo de tempo. O *RPF* tem a desvantagem teórica de que não é mais garantido que as amostras aproximem assintoticamente a posteriori. Em casos práticos, a performance do *RPF* é melhor que a do *SIR* quando o empobrecimento das amostras é severo, por exemplo, quando o ruído no processo é pequeno.

3.6. Filtros Auxiliares

O filtro de partículas auxiliar foi desenvolvido por Pitt e Shephard (1999) como uma variante do filtro *SIR* padrão. Esse filtro pode ser derivado a partir do filtro de partículas padrão escolhendo-se uma densidade de importância $q(x_t, i | y_{1:t})$, a partir da qual é amostrado o par $\{x_t^j, i^j\}_{j=1}^M$, onde i^j é uma variável latente auxiliar que se refere ao índice da partícula em $t - 1$.

Aplicando a regra de Bayes, uma proporcionalidade pode ser derivada para $p(x_t, i | y_{1:t})$ da seguinte forma:

$$\begin{aligned}
 p(x_t, i | y_{1:t}) &\propto p(y_t | x_t) p(x_t, i | y_{1:t-1}) \\
 &= p(y_t | x_t) p(x_t | i, y_{1:t-1}) p(i | y_{1:t-1}) \\
 &= p(y_t | x_t) p(x_t | x_{t-1}^i) w_{t-1}^i
 \end{aligned} \tag{48}$$

O filtro opera obtendo uma amostra a partir da densidade conjunta $p(x_t, i | y_{1:t})$ e então omitindo os índices i no par (x_t, i) para produzir uma amostra $\{x_t^j\}_{j=1}^N$ a partir da

densidade marginal $p(x_t | y_{1:t})$. A densidade de importância usada para amostrar

$\{x_t^j, i^j\}_{j=1}^M$ é definida para satisfazer a proporcionalidade:

$$q(x_t, i | y_{1:t}) \propto p(y_t | \mu_t^i) p(x_t | x_{t-1}^i) w_{t-1}^i \quad (49)$$

onde μ_t^i é alguma caracterização de x_t , dado x_{t-1}^i . Essa caracterização poderia ser a média, nesse caso $\mu_t^i = E[x_t | x_{t-1}^i]$, ou uma amostra, $\mu_t^i \sim p(x_t | x_{t-1}^i)$. Escrevendo:

$$q(x_t, i | y_{1:t}) = q(i | y_{1:t}) q(x_t | i, y_{1:t}) \quad (50)$$

e escolhendo:

$$q(x_t | i, y_{1:t}) = p(x_t | x_{t-1}^i) \quad (51)$$

segue da Equação (49) que:

$$q(i | y_{1:t}) \propto p(y_t | \mu_t^i) w_{t-1}^i \quad (52)$$

Então é atribuído à amostra $\{x_t^j, i^j\}_{j=1}^M$ um peso proporcional à razão dos lados direitos das Equações (48) e (49):

$$w_t^j \propto w_{t-1}^{i^j} \frac{p(y_t | x_t^j) p(x_t^j | x_{t-1}^{i^j})}{q(x_t^j, i^j | y_{1:t})} = \frac{p(y_t | x_t^j)}{p(y_t | \mu_t^{i^j})} \quad (53)$$

Embora desnecessário, o filtro de partículas auxiliar original proposto em (PITT e SHEPHARD, 1999) consistia de um passo a mais, a saber, um estágio de reamostragem para produzir uma amostra *i.i.d.* $\{x_t^j, i^j\}_{j=1}^M$ com pesos iguais.

Comparado com o filtro *SIR*, a vantagem do filtro de partículas auxiliar é que ele naturalmente gera pontos a partir da amostra em $t - 1$ que, condicionados às medidas atuais, são provavelmente mais próximos do verdadeiro estado. A idéia do filtro de partículas auxiliar pode ser vista como reamostrar no instante de tempo anterior, baseado em alguma estimativa pontual μ_t^i que caracteriza $p(x_t | x_{t-1}^i)$. Se o ruído no

processo é pequeno de forma que $p(x_t | x_{t-1}^i)$ seja bem caracterizado por μ_t^i , então ele não será tão sensível a pontos espúrios como o *SIR*, e os pesos serão mais uniformes. Entretanto, se o ruído no processo é grande, um único ponto não caracteriza bem $p(x_t | x_{t-1}^i)$, e o filtro reamostra baseado numa aproximação pobre de $p(x_t | x_{t-1}^i)$. Nesses casos, o uso do filtro de partículas auxiliar degrada a performance.

3.7. Rao-Blackwellização

A idéia básica da Rao-Blackwellização (DOUCET *et al.*, 2000) consiste em amostrar algumas das variáveis e marginalizar as demais de forma exata usando um filtro de Kalman ou qualquer outro filtro ótimo. O nome da técnica está relacionado à fórmula de Rao-Blackwell (CASELLA e ROBERT, 1996).

Um dos maiores problemas dos filtros de partículas é que amostrar em espaços de grandes dimensões pode ser ineficiente. Entretanto, em alguns casos o modelo tem uma “subestrutura tratável”, que pode ser analiticamente marginalizada, condicionada a atribuição de valores em outros nós. A marginalização analítica pode ser realizada usando um algoritmo padrão, tal como o filtro de Kalman, discutido no Capítulo 2. A vantagem dessa estratégia é que ela pode reduzir drasticamente o espaço a partir do qual é preciso amostrar.

Suponha que as variáveis de estado x_t possam ser divididas em dois grupos, r_t e z_t , tais que:

$$p(x_t | x_{t-1}) = p(z_t | r_{t-1:t}, z_{t-1})p(r_t | r_{t-1}) \quad (54)$$

e, condicionada em $r_{0:t}$, a distribuição posterior condicional $p(z_{0:t} | y_{1:t}, r_{0:t})$ é analiticamente tratável. Então pode-se facilmente marginalizar $z_{0:t}$ a partir da densidade posterior e estimar $p(r_{0:t} | y_{1:t})$, em um espaço dimensional reduzido. Formalmente, usa-se a seguinte decomposição da posterior, que segue da regra da cadeia:

$$p(r_{0:t}, z_{0:t} | y_{1:t}) = p(z_{0:t} | y_{1:t}, r_{0:t})p(r_{0:t} | y_{1:t}) \quad (55)$$

Uma vez que a dimensão de $p(r_{0:t} | y_{1:t})$ é menor que a dimensão de $p(r_{0:t}, z_{0:t} | y_{1:t})$, espera-se obter melhores resultados.

Filtros de partículas Rao-Blackwellizados (RBPF, Rao-Blackwellised Particle Filters) já foram aplicados em contextos específicos, tais como misturas de Gaussianas (DOUCET *et al.*, 2000). O grande problema da Rao-Blackwellização está em automaticamente identificar quais variáveis devem ser amostradas e quais devem ser tratadas analiticamente, ponto que permanece em aberto.

3.8. Suavização

Como descrito na Seção 2.1, a *suavização* consiste em computar a distribuição posterior de um estado passado dadas todas as observações até o presente, ou seja, computar $p(x_t | y_{1:T})$ para $1 \leq t \leq T$, a fim de obter melhores estimativas dos estados. Existem dois métodos de suavização mais usados em filtros de partículas: o *Suavizador Forward-Backward* e *Suavizador com Dois Filtros* (KITAGAWA, 1996).

O Suavizador *Forward-Backward* se baseia no fato de que a densidade suavizada $p(x_t | y_{1:T})$ pode ser fatorada como segue:

$$\begin{aligned}
p(x_t | y_{1:T}) &= \int p(x_t, x_{t+1} | y_{1:T}) dx_{t+1} \\
p(x_t | y_{1:T}) &= \int p(x_{t+1} | y_{1:T}) p(x_t | x_{t+1}, y_{1:t}) dx_{t+1} \\
p(x_t | y_{1:T}) &= p(x_t | y_{1:t}) \int \frac{p(x_{t+1} | y_{1:T}) p(x_{t+1} | x_t)}{\int p(x_{t+1} | x_t) p(x_t | y_{1:t}) dx_t} dx_{t+1} \tag{56}
\end{aligned}$$

O primeiro termo do lado direito da Equação (56), fora da integral, corresponde à estimativa filtrada no tempo t . O primeiro termo do numerador é a estimativa suavizada para $t + 1$, enquanto o segundo termo é a probabilidade de transição de estados. O denominador corresponde a predição para o tempo $t + 1$.

Portanto, essa é uma fórmula recursiva para calcular a densidade suavizada $p(x_t | y_{1:T})$ em termos da densidade de filtragem $p(x_t | y_{1:t})$, da densidade de predição $p(x_{t+1} | y_{1:t})$, e da densidade suavizada no tempo $t + 1$ $p(x_{t+1} | y_{1:T})$. Então, após a filtragem, feita para a frente no tempo, a suavização é feita voltando no tempo.

A densidade suavizada pode ser aproximada por:

$$p(x_t | y_{1:T}) = \sum_{i=1}^N w_{t|T}^i \delta_{x_t^i}(x_t), \tag{57}$$

onde os pesos de importância $w_{t|T}^i$ são obtidos através da recursão retrocedendo no tempo:

$$w_{t|T}^i = w_t^i \left[\sum_{j=1}^N w_{t+1|T}^j \frac{p(x_{t+1}^j | x_t^i)}{\sum_{k=1}^N w_t^k p(x_{t+1}^k | x_t^k)} \right], \tag{58}$$

com $w_{T|T}^i = w_T^i$. Deve-se notar que o Suavizador *Forward-Backward* mantém as localizações originais das partículas e atualiza seus pesos para obter uma aproximação da densidade suavizada. Assim, seu sucesso depende de que a densidade filtrada tenha suporte onde a densidade suavizada é significativa.

Considerando que o cálculo do denominador da fração na Equação (58) é independente da partícula cujo peso está sendo atualizado, a suavização pode ser feita em $O(N^2)$.

Uma distribuição marginal suavizada pode também ser obtida pela combinação do resultado de dois procedimentos de filtragem independentes, um deles fazendo inferência para frente no tempo e o outro para trás. Essa é a idéia básica do Suavizador com Dois Filtros. A seguinte fatorização é utilizada:

$$\begin{aligned}
 p(x_t | y_{1:T}) &= p(x_t | y_{1:t-1}, y_{t:T}) \\
 p(x_t | y_{1:T}) &= \frac{p(x_t | y_{1:t-1})p(y_{t:T} | y_{1:t-1}, x_t)}{p(y_{t:T} | y_{1:t-1})} \\
 p(x_t | y_{1:T}) &\propto p(x_t | y_{1:t})p(y_{t+1:T} | x_t)
 \end{aligned} \tag{59}$$

O Filtro 1 calcula $p(x_t | y_{1:t})$ através do procedimento de filtragem já conhecido. O Filtro 2 computa $p(y_{t+1:T} | x_t)$ seqüencialmente, usando o *Filtro de Informação Backward* (MAYNE, 1966), notando que:

$$p(y_{t:T} | x_t) = \int p(y_{t+1:T} | x_{t+1})p(x_{t+1} | x_t)p(y_t | x_t)dx_{t+1} \tag{60}$$

O problema com essa recursão é que $p(y_{t:T} | x_t)$ não é uma função de densidade de probabilidade no argumento x_t e assim sua integral sobre x_t pode não ser finita (a recursão explode). Portanto, os filtros de partículas existentes não podem ser usados para aproximar $p(y_{t:T} | x_t)$, a menos que sejam feitas suposições irreais. Kitagawa (1996) implicitamente assume que $\int p(y_{t:T} | x_t)dx_t < \infty$ para fazer suavização em filtros de partículas, entretanto se essa suposição é violada os métodos não se aplicam.

3.9. Filtro de Partículas com Somas de Gaussianas

O *filtro de partículas com somas de Gaussianas* (KOTECHA e DJURIC, 2003) aproxima as distribuições de filtragem e preditiva por misturas de Gaussianas ponderadas e são basicamente “bancos” de filtros de partículas Gaussianos.

O ruído no processo é aproximado por uma mistura de Gaussianas, enquanto o ruído nas observações é aproximado por uma Gaussiana, embora a extensão para o caso não-Gaussiano possa ser facilmente deduzida por uma aproximação com misturas de Gaussianas. A saída de um banco de filtros de partículas Gaussianos é usada para calcular aproximações com misturas de Gaussianas da densidade preditiva.

Em outras palavras, o filtro de partículas com somas de Gaussianas pode ser visto como um filtro de partículas cujas partículas são componentes de uma mistura de Gaussianas. Desse modo, a abordagem sofre de problemas semelhantes aos dos filtros de partículas comuns: a covariância das componentes cresce com o tempo, podendo acontecer o colapso das componentes, de forma que após muitas iterações a posterior é aproximada por uma única Gaussiana, que pode ser inadequada. Além disso a escolha do número G de componentes da mistura não é automática e depende do problema. Extensões do filtro de partículas com somas de Gaussianas são apresentadas em (KOTECHA e DJURIC, 2003).

3.10. Filtro de Partículas com Pontos Sigma

A família dos filtros de Kalman com pontos sigma contém dois algoritmos que baseiam seu funcionamento na aproximação de densidades por partículas: o *filtro de partículas unscented* (van der MERWE *et al.*, 2000) e o *filtro de partículas com pontos sigma e misturas de gaussianas* (van der MERWE e WAN, 2003b).

Como foi mencionado anteriormente, o uso da priori da transição de estados como densidade de importância pode ser ineficiente, pois não se consideram as observações mais atuais. É proposto em (van der MERWE *et al.*, 2000) o uso do *filtro de Kalman unscented* (UKF) (discutido no Capítulo 2) para gerar a densidade de importância, de forma que as partículas sejam “movidas” para as regiões de alta verossimilhança. Apenas é requerido que se propaguem estatísticas suficientes do UKF para cada partícula. O filtro de partículas resultante da aplicação do *UKF* para gerar a densidade de importância é chamado *filtro de partículas unscented*, ou simplesmente *filtro de partículas com pontos sigma*.

O *filtro de partículas com pontos sigma e misturas de gaussianas* (van DER MERWE e WAN, 2003b) não é estritamente um filtro de partículas. Ele combina um banco de filtros de Kalman com pontos sigma, para o passo de atualização do tempo e geração da densidade de importância, com um filtro de partículas, para o passo de atualização das observações. A densidade posterior é representada por uma mistura de Gaussianas construída a partir do conjunto de partículas do passo de atualização das observações por intermédio de um algoritmo EM. Esse passo substitui o estágio de reamostragem necessário pela maioria dos filtros de partículas e atenua o problema de empobrecimento das amostras.

No tempo $t - 1$, a densidade posterior é aproximada pela mistura de Gaussianas com G componentes:

$$p(x_{t-1} | y_{1:t-1}) = \sum_{g=1}^G \alpha_{t-1}^g N(x_{t-1}, \mu_{t-1}^g, \Sigma_{t-1}^g) \quad (61)$$

onde α_{t-1}^g são as proporções da mistura, μ_{t-1}^g são as médias e Σ_{t-1}^g são as covariâncias das componentes.

Os ruídos no processo e nas observações também são aproximados por misturas de Gaussianas. A saída de um banco de filtros de Kalman com pontos sigma é usada para calcular aproximações com misturas de Gaussianas da densidade preditiva $p(x_t | y_{1:t-1})$ e da densidade posterior $p(x_t | y_{1:t})$, que é usada apenas como densidade de importância no passo de atualização das observações.

No passo de atualização das observações, são tiradas N amostras $\{\chi_t^i, i = 1, \dots, N\}$ a partir da aproximação de $p(x_t | y_{1:t-1})$ e os pesos correspondentes são calculados:

$$w_t^i = \frac{p(y_t | \chi_t^i) \hat{p}(\chi_t^i | y_{1:t-1})}{\hat{p}(\chi_t^i | y_{1:t})} \quad (62)$$

Os pesos são então normalizados e um algoritmo EM é usado para adequar uma mistura de Gaussianas com G componentes ao conjunto de partículas ponderadas $\{\chi_t^i, w_t^i, i = 1, \dots, N\}$ representando a aproximação da distribuição posterior atualizada com as observações do tempo t :

$$p(x_t | y_{1:t}) = \sum_{g=1}^G \alpha_t^g N(x_t, \mu_t^g, \Sigma_t^g) \quad (63)$$

O algoritmo EM é inicializado com as G médias, covariâncias e proporções de misturas de $p(x_{t-1} | y_{1:t-1})$.

Uma grande limitação do filtro de partículas com pontos sigma e misturas de gaussianas, como no filtro de partículas com somas de Gaussianas, é que o número de componentes das misturas não é escolhido automaticamente e depende do problema abordado.

4. ESTIMAÇÃO DE DENSIDADES

Suponha que se tem um conjunto de observações de uma variável aleatória contínua e deseja-se estimar a densidade a partir da qual provêm. Esse problema é conhecido como estimação de densidades.

A estimação de densidade é a adequação de uma função de densidade de probabilidade a dados (SCOTT, 1997). Pode-se fazer uma estimação paramétrica ou não-paramétrica. Usualmente a expressão “estimação de densidade” se refere à metodologia não-paramétrica.

A abordagem paramétrica para estimação de uma densidade envolve assumir que ela pertence a uma família paramétrica de distribuições, tal como a família gama, e então estimar os parâmetros desconhecidos usando, por exemplo, o algoritmo EM. Como exemplo, o filtro de Kalman, abordado no Capítulo 2, considera que as densidades são todas Gaussianas lineares.

A estimação paramétrica de densidades requer uma especificação adequada da forma da densidade amostrada e a estimação do vetor de parâmetros. Portanto, a modelagem paramétrica implica em dois riscos de *bias*: na estimação dos parâmetros e na especificação incorreta da densidade de amostragem.

A estimação de densidades com *kernels* (PARZEN, 1962) é uma técnica usada para a estimação não-paramétrica de funções de densidade de probabilidade, especialmente efetiva quando modelos paramétricos não são apropriados.

Para entender a diferença entre modelos paramétricos e não-paramétricos, é necessário considerar como um dado modelo mudaria se o número de pontos de dados N fosse aumentado. Para modelos paramétricos, a estrutura básica do modelo

permanece fixa com o aumento de N . Por outro lado, no caso não-paramétrico, a classe das densidades aumenta com o aumento de N . Com $N + 1$ pontos de dados é possível se obter densidades com $N + 1$ modas, o que não é possível com N pontos de dados.

Existe um *tradeoff* entre flexibilidade e eficiência estatística que é relevante na escolha entre modelos paramétricos e não-paramétricos na estimação de densidades (JORDAN, 2002). Se os dados realmente vêm uma família paramétrica de densidades, então provavelmente seria melhor estimá-la usando um modelo paramétrico, embora a estimação com *kernels* também possa ser usada. O uso de *kernels* eventualmente converge para a verdadeira densidade, mas pode requerer muitos pontos de dados. Um estimador paramétrico convergirá mais rápido. Entretanto, se a verdadeira densidade não condiz com o modelo paramétrico escolhido, então a estimativa paramétrica convergirá para a densidade errada, enquanto que, mesmo assim, a estimativa não-paramétrica eventualmente convergirá para a verdadeira densidade.

Resumindo se forem feitas mais suposições então a convergência é mais rápida, mas a performance pode ser pobre se a realidade não condiz com as suposições. Estimadores não-paramétricos precisam de menos suposições mas requerem mais pontos de dados para níveis comparáveis de performance.

Na próxima seção será apresentado o algoritmo EM, método iterativo largamente utilizado no aprendizado de parâmetros em modelos paramétricos. Depois será abordada a estimação não-paramétrica de densidades com *kernels*, e em seguida serão mostrados alguns métodos usados para calcular a largura dos *kernels*, conceito que será introduzido mais adiante. Por fim, será feita uma comparação entre a estimação com *kernels* e a estimação paramétrica com modelos de misturas.

4.1. O Algoritmo EM

O algoritmo EM (BILMES, 1998) é um procedimento iterativo eficiente para computar a estimativa de máxima verossimilhança no caso de dados desconhecidos ou faltantes.

Na estimação da máxima verossimilhança, desejamos estimar os parâmetros do modelo para os quais os dados observados são mais prováveis.

Cada iteração do algoritmo EM consiste de duas etapas: o passo E e o passo M.

No passo E (*Expectation*), os dados faltando são estimados dadas as observações e as atuais estimativas dos parâmetros do modelo. Isso é realizado usando-se a esperança condicional, o que explica o nome do passo.

No passo M (*Maximization*), a função de verossimilhança é maximizada supondo-se que os dados faltando são conhecidos. São usadas as estimativas dos dados faltando conseguidas no passo E.

A convergência é garantida uma vez que é garantido que o algoritmo aumenta a verossimilhança a cada iteração.

Seja Y um vetor aleatório de observações que resulta de uma família parametrizada. Deseja-se estimar o vetor de parâmetros θ tal que a verossimilhança dos dados $p(Y | \theta)$ seja máxima. Essa estimativa é conhecida como estimativa de máxima verossimilhança para θ . Para estimá-la, é comum considerar a função de log-verossimilhança definida como:

$$L(\theta) = \ln p(Y | \theta) \tag{64}$$

Uma vez que $\ln(\cdot)$ é uma função estritamente crescente, o valor de θ que maximiza $p(Y | \theta)$ também maximiza $L(\theta)$.

O algoritmo EM pode ser visto então como um procedimento iterativo para maximizar $L(\theta)$. Assuma que após a n -ésima iteração a atual estimativa para θ é dada por θ_n . Uma vez que o objetivo é maximizar $L(\theta)$, deseja-se computar uma estimativa atualizada para θ tal que:

$$L(\theta) > L(\theta_n) \quad (65)$$

Equivalentemente, deseja-se maximizar a diferença:

$$L(\theta) - L(\theta_n) = \ln p(Y | \theta) - \ln p(Y | \theta_n) \quad (66)$$

Em problemas onde existem variáveis não-observadas ou faltando, o algoritmo EM constitui uma forma natural para sua inclusão.

Considere um vetor de variáveis aleatórias não-observadas x . A probabilidade total $p(Y | \theta)$ pode ser escrita em termos das variáveis desconhecidas x da seguinte forma:

$$p(Y | \theta) = \sum_x p(Y | x, \theta) p(x | \theta) \quad (67)$$

A Equação (66) pode então ser reescrita da seguinte forma:

$$\begin{aligned} L(\theta) - L(\theta_n) &= \ln \sum_x p(Y | x, \theta) p(x | \theta) - \ln p(Y | \theta_n) \\ &= \ln \sum_x p(Y | x, \theta) p(x | \theta) \frac{p(x | Y, \theta)}{p(x | Y, \theta_n)} - \ln p(Y | \theta_n) \\ &= \ln \sum_x p(x | Y, \theta_n) \frac{p(Y | x, \theta) p(x | \theta)}{p(x | Y, \theta_n)} - \ln p(Y | \theta_n) \end{aligned} \quad (68)$$

Sabe-se pela inequação de Jensen (BORMAN, 2004) que:

$$\ln \sum_{i=1}^n \lambda_i x_i \geq \sum_{i=1}^n \lambda_i \ln(x_i) \quad (69)$$

onde:

$$\sum_{i=1}^n \lambda_i = 1 \quad \lambda_i \geq 0 \quad (70)$$

Usando a inequação de Jensen com $\lambda_i = P(x|Y, \theta_n)$, uma vez que é uma medida de probabilidade e portanto $\lambda_i \geq 0$ e $\sum_{i=1}^n \lambda_i = 1$ como requerido, a partir da Equação (68), tem-se que:

$$L(\theta) - L(\theta_n) \geq \sum_x p(x|Y, \theta_n) \ln \left(\frac{p(Y|x, \theta) p(x|\theta)}{p(x|Y, \theta_n)} \right) - \ln p(Y|\theta_n) \quad (71)$$

$$= \sum_x p(x|Y, \theta_n) \ln \left(\frac{p(Y|x, \theta) p(x|\theta)}{p(x|Y, \theta_n) p(Y|\theta_n)} \right) \quad (72)$$

Formalmente tem-se:

$$\theta_{n+1} = \arg \max_{\theta} \{L(\theta) - L(\theta_n)\} \quad (73)$$

$$\theta_{n+1} = \arg \max_{\theta} \left\{ \sum_x p(x|Y, \theta_n) \ln \left(\frac{p(Y|x, \theta) p(x|\theta)}{p(x|Y, \theta_n) p(Y|\theta_n)} \right) \right\} \quad (74)$$

Eliminando os termos que são constantes em relação a θ :

$$\theta_{n+1} = \arg \max_{\theta} \left\{ \sum_x p(x|Y, \theta_n) \ln (p(Y|x, \theta) p(x|\theta)) \right\} \quad (75)$$

$$= \arg \max_{\theta} \left\{ \sum_x p(x|Y, \theta_n) \ln \left(\frac{p(Y, x, \theta)}{p(x, \theta)} \cdot \frac{p(x, \theta)}{p(\theta)} \right) \right\} \quad (76)$$

$$= \arg \max_{\theta} \left\{ \sum_x p(x|Y, \theta_n) \ln p(Y, x|\theta) \right\} \quad (77)$$

Nesta equação pode-se identificar os passos E e M:

$$\theta_{n+1} = \arg \max_{\theta} \{E[\ln p(Y, x|\theta) | Y, \theta_n]\} \quad (78)$$

Em resumo, o algoritmo EM assim consiste em iterar os seguintes passos:

1. Passo E: Determinar a esperança condicional $E[\ln P(Y,x|\theta) | Y,\theta_n]$
2. Passo M: Maximizar essa expressão em relação a θ .

4.2. Estimação Não-Paramétrica com Kernels

A estimação de densidade não-paramétrica constitui uma classe geral de métodos para lidar com casos onde há escassez de conhecimento. Muitas vezes os dados podem vir de um mecanismo complexo sobre o qual se tem pouco ou nenhum conhecimento a priori e a densidade dos dados pode não se adequar a nenhuma das formas padrões. A estimação de densidade não-paramétrica é consistente para quase qualquer densidade contínua e evita o passo de especificação de um modelo.

A idéia básica da estimação de densidades com *kernels* é que cada ponto de dado provê evidência para densidade de probabilidade não-zero naquele ponto (JORDAN, 2002). Uma forma simples de entender essa idéia é colocar um “átomo” de massa naquele ponto. Além disso, fazendo a suposição de que a densidade de probabilidade é suave, os átomos devem ter uma “largura” não-zero. A superposição de N átomos desses, um para cada ponto de dado, fornece uma estimativa da densidade.

Formalmente, seja $k(x, x_n, \lambda)$ uma função *kernel* - função não-negativa cuja integral com relação a x é igual a 1. O argumento x_n determina a localização da função *kernel* e λ é um parâmetro geral de suavização que determina a largura das funções *kernel* e conseqüentemente a suavidade da estimativa da densidade resultante. Superpondo N dessas funções *kernel* e dividindo por N , obtemos uma densidade de probabilidade:

$$\hat{p}(x) = \frac{1}{N} \sum_{n=1}^N k(x, x_n, \lambda) \quad (79)$$

Essa densidade é conhecida como estimativa de densidade por *kernels* (*Kernel Density Estimator*) da densidade $p(x)$.

Uma variedade de diferentes funções *kernel* são usadas na prática. As mais simples são freqüentemente preferidas, em parte por questões computacionais, pois calcular a densidade num dado ponto x requer N avaliações da função. No caso de *kernels* Gaussianos, a média da Gaussiana é o dado e o desvio padrão é a largura do *kernel*.

A extensão da estimação de densidades por *kernels* para o caso multivariado é de grande importância, uma vez que a modelagem paramétrica é mais difícil que no caso univariado. Entretanto a extensão da metodologia apresenta dificuldades. Um deles é que quanto maior a dimensão, maior será o número de larguras de *kernels* que precisarão ser estimados. Outro é a chamada *maldição da dimensionalidade*, que significa que, com tamanhos de amostras praticáveis, estimação não-paramétrica de densidades razoável é muito difícil em mais que cinco dimensões. Entretanto, a estimação por *kernels* tem sido uma ferramenta efetiva no caso bivariado.

4.3. Seleção de Largura de Kernels

A implementação prática de um estimador por *kernels* requer a especificação da largura do *kernel*. Na maioria dos casos não há conhecimento sobre a estrutura dos dados que possa indicar qual largura fornece a estimação mais próxima da verdadeira

densidade. Um método que usa os dados para produzir uma largura para os *kernels* é chamado seletor de largura (WAND e JONES, 1995).

Os seletores de largura de *kernels* disponíveis atualmente podem ser divididos em duas classes. A primeira classe consiste de fórmulas simples facilmente computáveis que objetivam encontrar uma largura razoável para uma larga faixa de situações, mas sem qualquer garantia matemática de estar próxima ao valor ótimo. Tais seletores de largura são motivados pela necessidade de ter estimativas por *kernels* automaticamente geradas para algoritmos que requerem muitos passos de estimação. O segundo tipo de seletor de largura baseia-se em argumentos matemáticos mais desenvolvidos e requerem consideravelmente mais esforço computacional, mas objetivam prover uma boa largura para uma classe bem geral de funções. Os dois seletores abordados a seguir pertencem à primeira classe descrita.

Uma largura h é *EQIMA*-ótima se ela minimiza o erro quadrático integrado médio assintótico, dado por:

$$EQIMA\{\hat{f}(\cdot, h)\} = (nh)^{-1} R(K) + \frac{1}{4} h^4 \mu_2(K)^2 R(f'') \quad (80)$$

onde n é o número de pontos de dados, K é a função *kernel* e:

$$R(K) = \int K(x)^2 dx \quad (81)$$

$$\mu_2(K) = \int x^2 K(x) dx \quad (82)$$

O *seletor de largura de escala normal* consiste em usar a largura que é *EQIMA*-ótima para a densidade normal tendo a mesma escala que a estimada para a densidade dos dados, e é dada pela fórmula:

$$\hat{h}_{EN} = \left[\frac{8\pi^{\frac{1}{2}} R(K)}{3\mu_2(K)^2 n} \right]^{\frac{1}{5}} \hat{\sigma} \quad (83)$$

onde $\hat{\sigma}$ é alguma estimativa do desvio padrão, normalmente o desvio padrão amostral.

O seletor de largura de escala normal fornece respostas razoáveis quando a densidade verdadeira se aproxima de uma normal. Entretanto, para multimodalidade esse seletor tende a suavizar demais as estimativas, mascarando características importantes dos dados.

O princípio da *suavização máxima* se baseia no fato de que existe um limite superior simples para a largura *EQIMA*-ótima para a estimação de densidades com um valor fixo de uma medida de escala em particular (por exemplo, o desvio padrão). Tal limite serve como base para o *seletor de largura supersuavisada*:

$$\hat{h}_{SS} = \left[\frac{243R(K)}{35\mu_2(K)^2 n} \right]^{\frac{1}{5}} \hat{\sigma} \quad (84)$$

onde $\hat{\sigma}$ é o desvio padrão amostral.

Embora o seletor de largura supersuavisada forneça uma largura grande demais para a estimação ótima de uma densidade qualquer, ele provê um excelente ponto de partida para a escolha subjetiva da largura, que se segue considerando-se frações convenientes de seu valor.

Os métodos de seleção de largura que serão abordados a seguir fazem parte da segunda classe de seletores descrita no início desta seção.

A *validação cruzada dos mínimos quadrados* é o nome dado ao seletor de largura que consiste em escolher a largura que minimiza a seguinte equação:

$$VCMQ(h) = \int \hat{f}(x, h)^2 dx - 2n^{-1} \sum_{i=1}^n \hat{f}_{-i}(X_i, h) \quad (85)$$

onde:

$$\hat{f}_{-i}(x, h) = (n-1)^{-1} \sum_{j \neq i}^n K_h(x - X_j) \quad (86)$$

é a estimativa da densidade baseada na amostra com X_i deletado, freqüentemente chamado estimador de densidade “exclui-um”. Esse é o motivo do termo validação cruzada, que se refere ao uso de parte de uma amostra para obter informação sobre outra parte.

É possível que $VCMQ(h)$ tenha mais de um mínimo local. Portanto é preciso muito cuidado ao utilizar na prática a validação cruzada dos mínimos quadrados para selecionar a largura dos *kernels*. Pesquisas mostraram que a performance teórica e prática desse seletor de largura são desapontadoras (WAND e JONES, 1995), entretanto a idéia serviu de base para o desenvolvimento de outros seletores.

O *seletor de largura por validação cruzada viesada* consiste na minimização da equação:

$$VCE(h) = (nh)^{-1} R(K) + \frac{1}{4} h^4 \mu_2(K)^2 \tilde{R}(f'') \quad (87)$$

onde:

$$\tilde{R}(f'') = n^{-2} \sum_{i \neq j} (K_h'' * K_h'')(X_i - X_j) \quad (88)$$

A principal vantagem desse seletor é que ele é mais estável que a validação cruzada dos mínimos quadrados, no sentido que sua variância assintótica é consideravelmente menor, entretanto, seu *bias* é maior. Como no caso da função $VCMQ(h)$, a função $VCE(h)$ pode ocasionalmente ter mais de um mínimo local.

Além dos seletores vistos anteriormente, existem vários outros mais elaborados, como por exemplo os seletores de largura *plug-in*, que são baseados na idéia simples de “plugar” em fórmulas para largura assintoticamente ótima, estimativas por *kernels* das quantidades desconhecidas que aparecem.

4.4. Kernels vs. Modelos de Misturas

Uma densidade de mistura para uma variável aleatória x é dada pela soma convexa das densidades componentes $f_k(x | \theta_k)$:

$$p(x | \theta) = \sum_{k=1}^K \alpha_k f_k(x | \theta_k) \quad (89)$$

onde θ_k é um vetor de parâmetros e α_k é uma constante não-negativa tal que:

$$\sum_{k=1}^K \alpha_k = 1 \quad (90)$$

As densidades $f_k(x | \theta_k)$ são chamadas *componentes da mistura* e os parâmetros α_k são as *proporções de mistura*. O vetor de parâmetros θ é o conjunto de todos os parâmetros, incluindo as proporções de mistura:

$$\theta \stackrel{\Delta}{=} (\alpha_1, \dots, \alpha_K, \theta_1, \dots, \theta_K) \quad (91)$$

Uma vez que a soma das proporções de mistura é igual a 1 (Equação (90)), a função definida na Equação (89) é de fato uma densidade.

Como exemplo, a densidade de uma mistura de Gaussianas com K componentes seria dada por:

$$p(x | \theta) = \sum_{k=1}^K \alpha_k N(x | \mu_k, \sigma_k^2) \quad (92)$$

onde cada componente da mistura é uma distribuição Gaussiana com média μ_k e variância σ_k^2 . As misturas de Gaussianas são as formas mais populares de modelos de misturas.

Comparando a Equação (89), da densidade de um modelo de mistura, com a Equação (79), de estimação por *kernels*, é impossível não notar uma certa semelhança e é inevitável se perguntar qual das duas formas de estimar densidades seria a melhor e por quê.

A diferença chave entre as duas abordagens é revelada quando o número de pontos de dados N aumenta. O modelo de mistura geralmente é visto como um modelo paramétrico, no qual o número K de componentes da mistura não aumenta com o aumento no número de pontos de dados. Na abordagem de estimação não-paramétrica com *kernels*, por outro lado, quando o número de pontos de dados aumenta, o número de *kernels* também aumenta, e geralmente espera-se que a largura dos *kernels* λ diminua para permitir uma adequação melhor aos detalhes da verdadeira densidade (JORDAN, 2002).

Considere o caso particular em que tem-se uma mistura de Gaussianas e funções *kernel* Gaussianas. Nesse caso, o estimador com *kernels* Gaussianos pode ser visto como uma mistura de Gaussianas em que as médias são fixas nas localizações dos pontos de dados, as variâncias são iguais a λ^2 e as proporções de mistura são setadas para $1/N$.

Foi provado que qualquer densidade de probabilidade pode ser aproximada o tanto quanto for desejado por uma mistura de Gaussianas, *para algum número de*

componentes (ANDERSON e MOORE, 1979). Entretanto, é possível que não se saiba de antemão o número K de componentes que deve ser usado. Filtros baseados em misturas de Gaussianas, tais como o filtro de partículas com somas de Gaussianas e o filtro de partículas com pontos sigma e misturas de Gaussianas, discutidos no Capítulo 3, falhariam caso o número de componentes escolhido não fosse suficiente para representar a densidade real dos dados.

Ao contrário de modelos de misturas, que podem convergir para uma densidade errada, o estimador de densidades por *kernels* eventualmente converge para a densidade verdadeira, desde que sejam fornecidos pontos de dados o suficiente.

5. COMITÊS DE ESTIMADORES

Um comitê de estimadores (*ensemble*) (DIETTERICH, 1998) é um conjunto de estimadores cujas decisões individuais são combinadas de alguma forma (tipicamente por voto ponderado ou não ponderado). Comitês são freqüentemente muito mais acurados que os estimadores que os compõem. Pesquisas (SUEN *et al.*, 2005) mostram que o uso de comitês de algoritmos de aprendizado pode reduzir *bias* e *variância* em conjunto, de forma a minimizar o erro de teste.

Nesta seção será apresentada relação entre *bias* e *variância* em um modelo gerado por um algoritmo de aprendizado. Logo após serão descritas as classes de métodos às quais pertencem a grande maioria dos algoritmos para a construção de comitês já desenvolvidos. Serão apresentadas também as formas mais comuns usadas para combinar os resultados dos estimadores a fim de obter a predição final de um comitê. Em seguida será abordado o problema de se escolher quais estimadores devem fazer parte do comitê e quais devem ser descartados, pois degradam a performance ou não produzem nenhuma diferença nos resultados. Por fim, será descrita a análise de *curvas características de erro de regressão* (BI e BENNETT, 2003), técnica para a avaliação e comparação de modelos de regressão, utilizada com sucesso na seleção de modelos de comitês de filtros de Kalman com pontos sigma para a previsão de séries temporais (PINA e ZAVERUCHA, 2006b).

5.1. Redução de Bias e Variância

A performance de generalização de um método de aprendizado está relacionada a sua capacidade de predição em dados de teste independentes. Quantificar essa performance é extremamente importante na prática, uma vez que ela guia a escolha do método ou modelo de aprendizado e fornece uma medida da qualidade do modelo escolhido.

Seja Y a variável objetivo, X um vetor de entradas, e $\hat{f}(X)$ um modelo de predição que foi estimado a partir de um conjunto de treinamento. O erro de teste ou generalização de um algoritmo de aprendizado pode ser decomposto em três termos (GEMAN *et al.*, 1992):

$$\begin{aligned} \text{Erro}(x) &= E[(Y - \hat{f}(x))^2 \mid X = x]^2 \\ &= \sigma_\varepsilon^2 + [E\hat{f}(x) - f(x)]^2 + E[\hat{f}(x) - E\hat{f}(x)]^2 \\ &= \sigma_\varepsilon^2 + \text{Bias}^2(\hat{f}(x)) + \text{Var}(\hat{f}(x)) \\ &= \text{Erro Irreduzível} + \text{Bias}^2 + \text{Variância} \end{aligned} \tag{93}$$

O primeiro termo é o *erro irreduzível*, variância do objetivo em torno de sua média real e não pode ser evitado. O segundo termo é o *bias* quadrático, quantidade pela qual a média das estimativas difere da média real. O último termo é a *variância* da estimativa em torno de sua média.

O *bias* e a variância das estimativas podem ser controlados e constituem o erro médio quadrático ao se estimar $f(X)$.

Tipicamente a complexidade do modelo deve ser escolhida para lidar com *bias* e variância de forma a minimizar o erro de teste (HASTIE *et al.*, 2001).

Se o modelo adaptar-se demais aos dados de treinamento (*overfitting*) ele não generalizará bem, ou seja, haverá um grande erro de teste. Nesse caso haverá grande variância nas predições.

Em contraste, se o modelo não é complexo o suficiente, haverá subadequação (*underfitting*) e poderá ter um grande *bias*, novamente resultando em pobre generalização.

Entre esses dois extremos existe uma complexidade ótima de modelo que fornece o erro de teste mínimo.

5.2. Métodos para a Construção de Comitês

Muitos métodos para a construção de comitês foram desenvolvidos. Alguns métodos são gerais e podem ser aplicados a qualquer algoritmo de aprendizado, outros só podem ser aplicados a determinados algoritmos.

Os métodos mais populares para a construção de comitês funcionam baseados na subamostragem dos exemplos de treinamento, isto é, manipulando-se os exemplos de treinamento para a geração de múltiplas hipóteses. O algoritmo de aprendizado é executado várias vezes, cada uma delas com um diferente subconjunto dos exemplos de treinamento. Essa técnica funciona especialmente bem para algoritmos de aprendizado instáveis, para os quais pequenas alterações nos dados de treinamento geram grandes mudanças nas hipóteses aprendidas. Dentre os métodos de construção de comitês baseados na subamostragem dos exemplos de treinamento estão: *bagging* (BREIMAN,

1996), comitês com validação cruzada (PARMANTO *et al.*, 1996) e *boosting* (SCHAPIRE, 1990).

Uma outra classe de métodos para a construção de comitês de estimadores consiste na manipulação dos atributos de entrada disponíveis para o algoritmo, como por exemplo em (TUMER e GHOSH, 1996). Cada estimador do comitê é treinado usando-se apenas um subconjunto dos atributos de entrada. Tais métodos apenas funcionam quando os atributos de entrada são altamente redundantes, e por isso são pouco usados.

Outra forma de construir um comitê de estimadores é manipulando o valor da função objetivo nos exemplos de treinamento. A *codificação de saída com correção de erro* (DIETTERICH e BAKIRI, 1995) é um método que se baseia nessa idéia. A cada iteração o domínio de classes é dividido em duas superclasses e faz-se a renomeação dos exemplos que são fornecidos ao algoritmo, gerando um estimador do comitê. A predição é feita baseada na contagem da pertinência das classes dentre as superclasses preditas por cada estimador.

Uma das formas mais simples de se obter estimadores para um comitê é alterando-se os valores dos parâmetros do algoritmo a cada execução. Esse procedimento inevitavelmente gera estimadores bons e ruins. Já foi mostrado (BREIMAN, 1996) que bons estimadores tendem a ser bons em uma grande parcela dos exemplos, enquanto que estimadores ruins podem ser bons em um conjunto diferente de exemplos. Essa diversidade é uma das causas da boa performance de um comitê quando comparado a um estimador funcionando individualmente, principalmente em problemas de regressão (BROWN *et al.*, 2005).

Um comitê de estimadores pode também ser construído alterando-se o código do algoritmo usado de forma que decisões que eram tomadas deterministicamente possam ser aleatórias, gerando diferentes estimadores a cada execução. Ali e Pazzani (1996) usaram esse método com o algoritmo FOIL (*First Order Inductive Learner*) (QUINLAN, 1990) para o aprendizado em lógica de primeira-ordem (LAVRAC e DZEROSKI, 1994). Algoritmos estocásticos podem naturalmente ser usados para se construir comitês dessa forma.

Além de todos os métodos descritos anteriormente, é possível combinar métodos para se alcançar uma performance ainda maior. Dentre tais métodos estão *bagging gradient boosting* e *bagging stochastic gradient boosting* (SUEN *et al.*, 2005), que são formados pela combinação de *bagging* com *gradient boosting* (FRIEDMAN, 1999) e com *stochastic gradient boosting* (FRIEDMAN *et al.*, 2000), respectivamente.

Deve-se notar que os estimadores componentes de um comitê não precisam necessariamente ser gerados por um único algoritmo. Vários algoritmos de aprendizado podem ser utilizados para gerar os estimadores, se beneficiando das diferentes capacidades de generalização. De fato, muitas pesquisas já foram realizadas combinando com sucesso vários algoritmos de aprendizado (ver, por exemplo, (CARUANA e NICULESCU-MIZIL, 2004)).

5.3. Métodos para a Combinação de Estimadores

Dado que um comitê de estimadores foi treinado, é preciso escolher um método para combinar as decisões individuais de cada estimador.

A abordagem mais simples usada em classificação é o *voto não-ponderado*, onde a classe será aquela predita pela maioria dos estimadores. Para problemas de regressão, o modo mais simples de se combinar os resultados individuais dos componentes de um comitê é fazer a média da saída de cada um deles.

O *voto ponderado* em classificação é um método que se baseia na atribuição de um peso a cada estimador, proporcional a sua acurácia num conjunto de validação. Existem vários métodos de combinação por voto que se baseiam no *ranking* das classes feito por cada estimador (KELVIN *et al.*, 2003), atribuindo pesos às classes em função das posições no *ranking*. Em regressão, os pesos podem, por exemplo, ser inversamente proporcionais à variância das estimativas (PERRONE e COOPER, 1993).

O *empilhamento (stacking)* (WOLPERT, 1992) é um método mais elaborado para a combinação de estimadores que funciona da seguinte forma: suponha que se tem um conjunto de algoritmos de aprendizado e um conjunto de exemplos de treinamento. Cada um desses algoritmos é treinado para produzir um conjunto de estimadores-base. Os resultados computados por esse conjunto de estimadores são combinados para formar novas instâncias, chamadas meta-instâncias. Cada “atributo” da meta-instância é a saída de um dos estimadores e o valor objetivo é o mesmo da instância original. Outro algoritmo é treinado com as meta-instâncias e produz um meta-estimador, que é responsável pela predição final do comitê.

5.4. Métodos para a Seleção de Estimadores

Dado um conjunto de estimadores treinados, é possível construir um comitê de forma a maximizar sua performance pela seleção de seus componentes, evitando a inclusão de estimadores que possam degradar a performance do comitê, ou que constituam apenas *overhead* computacional, não influenciando os resultados de forma significativa.

Tal seleção pode ser exaustiva, testando-se todas as combinações de estimadores possíveis. Esse método é praticável apenas para poucos estimadores, sendo pouco usado.

Uma outra forma de selecionar os estimadores componentes de um comitê consiste numa busca gulosa em que, começando com um comitê vazio, a cada iteração é adicionado o estimador que maximiza a performance do conjunto. Em classificação a performance do comitê é medida pela acurácia, ou seja, porcentagem de acerto em um conjunto de validação. Para problemas de regressão, em geral usa-se o erro médio quadrático do comitê. Entretanto, pesquisas recentes (PINA e ZAVERUCHA, 2007b) apontam a análise de curvas características de erro de regressão como uma técnica mais adequada para a seleção de estimadores em um comitê.

Outras técnicas podem ser agregadas à busca, tais como uma boa inicialização do comitê, com estimadores que apresentam alta performance individual, ou o uso de seleção com reposição, em que um mesmo estimador pode ser escolhido mais de uma vez, reforçando sua contribuição no resultado final (CARUANA e NICULESCU-MIZIL, 2004).

5.5. Curvas Características de Erro de Regressão

Resultados alcançados por Provost, Fawcett e Kohavi (1998) indicam a análise ROC (PROVOST e FAWCETT, 1997) como uma metodologia superior à comparação de acurácias na avaliação de algoritmos de aprendizado classificadores. Mas as curvas ROC são limitadas a problemas de classificação. Curvas Características de Erro de Regressão (REC, *Regression Error Characteristic*) (BI e BENNETT, 2003) generalizam curvas ROC para regressão com benefícios similares. Como nas curvas ROC, o gráfico caracteriza a qualidade do modelo de regressão para diferentes níveis de tolerância de erro.

A análise de curvas REC é uma técnica para a avaliação e comparação de modelos de regressão que facilita a visualização da performance de muitas funções de regressão simultaneamente em um único gráfico. Um gráfico REC contém uma ou mais curvas monotonicamente crescentes (curvas REC), cada uma correspondendo a um modelo de regressão.

Pode-se facilmente comparar muitas funções de regressão examinando-se a posição relativa de suas curvas REC. O formato da curva revela informações adicionais que podem ser usadas para guiar a modelagem.

Curvas REC são traçadas com a tolerância de erro no eixo x e a acurácia de uma função de regressão no eixo y . A acurácia é definida como a porcentagem de pontos preditos dentro da tolerância.

Em regressão, o residual é o conceito análogo ao erro de classificação num problema de classificação. O residual é definido como a diferença entre o valor predito $f(x)$ e o valor real de resposta y para cada ponto (x, y) . Pode ser o erro quadrático

$(y - f(x))^2$ ou o desvio absoluto $|y - f(x)|$, dependendo da métrica de erro empregada. Residuais devem ser maiores que uma tolerância e para que sejam considerados como erros.

A área acima da curva REC (AOC, *Area Over the Curve*) é uma estimativa viesada do erro esperado para o modelo de regressão. É uma estimativa viesada porque ela sempre subestima a esperança real. Se e é calculado usando desvio absoluto (DA), então o valor da AOC é próximo ao desvio absoluto médio (DAM). Se e é baseado no erro quadrático (EQ), o valor da AOC aproxima o erro médio quadrático (EMQ). A avaliação de modelos de regressão usando curvas REC é qualitativamente invariante à escolha da métrica de erro e à escala de representação do residual. Quanto menor for a AOC, melhor será a função de regressão.

A fim de ajustar as curvas REC no gráfico REC, um modelo nulo é usado para fazer a escala do gráfico. Abordagens de regressão razoáveis produzem modelos de regressão que são melhores que o modelo nulo. O modelo nulo pode ser, por exemplo, o *mean model* (modelo da média): uma função constante igual à média da resposta dos dados de treinamento. Um exemplo de gráfico REC pode ser visto na Figura 1. O número entre parênteses na figura é o valor da AOC para cada curva.

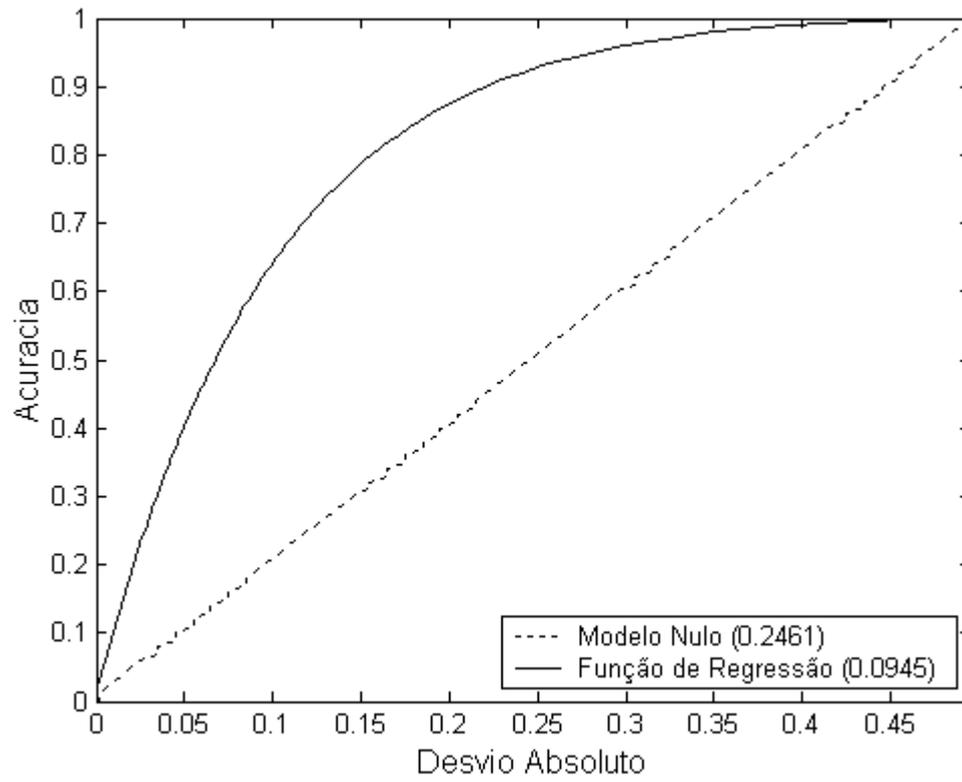


Figura 1. Exemplo de curva REC.

6. FILTRO DE PARTÍCULAS NÃO-PARAMÉTRICO

A principal dificuldade na utilização de filtros de partículas é que o pesquisador necessita ter conhecimentos sobre estatística para analisar a série temporal e escolher o modelo de espaço de estados a ser usado, definindo a função de transição de estados e a função das observações:

$$x_{t+1} = f(x_t, v_t) \quad t = 1, 2, \dots \quad (94)$$

$$y_t = h(x_t, w_t) \quad t = 1, 2, \dots \quad (95)$$

Para eliminar a necessidade de conhecimento especializado, o algoritmo desenvolvido estima tais funções não-parametricamente com o uso de *kernels*.

Para isso é necessário que exista um conjunto de *kernels* que represente cada uma dessas densidades. Tais conjuntos de *kernels* são inicializados aleatoriamente a partir dos dados e então, numa abordagem semelhante ao algoritmo EM, no passo E usa-se um filtro de partículas para inferir um conjunto de amostras que represente as estimativas dos estados. Tal conjunto é utilizado no passo M para atualizar os *kernels* que representam as funções de transição de estados e observações, de forma a convergirem para as densidades reais. Essa é a idéia do *Filtro de Partículas Não-Paramétrico* (FPNP). Apesar da semelhança que o nome pode sugerir, a abordagem proposta é totalmente diferente de *Kernel Dynamical Modeling* (RALAIVOLA e d'ALCHÉ-BUC, 2003), uma extensão com kernels de modelos dinâmicos lineares.

Como é um método aproximado e todas as informações são obtidas a partir dos dados, o filtro de partículas não-paramétrico pode se beneficiar de uma significativa redução em *bias* e variância se for utilizado como componente de um comitê.

Na próxima seção será apresentado o algoritmo básico do filtro de partículas não-paramétrico. O algoritmo básico pode ser melhorado em suas diversas partes componentes. Nas seções seguintes serão mostradas as decisões de projeto tomadas, seja com embasamento teórico ou através da avaliação empírica das opções adequadas. Após analisadas todas as decisões de projeto, o algoritmo final foi desenvolvido e, por fim, uma extensa avaliação experimental foi conduzida, comparando o sistema proposto com os sistemas de predição de séries temporais mais usados atualmente. A seguir serão apresentadas as séries temporais usadas tanto nas tomadas de decisões de projeto empíricas quanto na avaliação do algoritmo final.

Nos experimentos, 10 séries temporais com dados de problemas reais foram usadas com o intuito de tentar estabelecer um ranking geral entre os modelos testados. Os nomes e tamanhos das séries temporais usadas são mostrados na Tabela 1.

Tabela 1. Séries temporais usadas na avaliação experimental.

Séries Temporais	Tamanho das Séries
A	1000
Burstin	2001
Darwin	1400
Earthquake	2048
Leuven	2000
Series 1	96
Series 2	96
Series 3	96
Soiltemp	2304
Speech	1020

Os últimos 100 pontos de cada série (exceto Series 1, 2 e 3) foram usados para avaliar as performances dos algoritmos, os 100 pontos anteriores foram usados para validação e os demais foram usados para treinamento. Para Series 1, 2 e 3, a divisão usada foi: 36 pontos para treinamento, 24 pontos para validação e 36 pontos para teste (TEIXEIRA, 2005).

A série temporal A contém 1000 pontos de dados medidos em um experimento de laboratório de física. A série mostra a transição de periódico para caótico das pulsações de intensidade de um laser. Foi usada como conjunto de dados na competição de previsões de Santa Fe. Uma descrição meticulosa de suas características e o processo de geração podem ser encontrados em (WEIGEND e GERSHENFELD, 1994). A série é mostrada na Figura 2 e pode ser obtida em:

<http://www-psych.stanford.edu/~andreas/Time-Series/SantaFe.html>

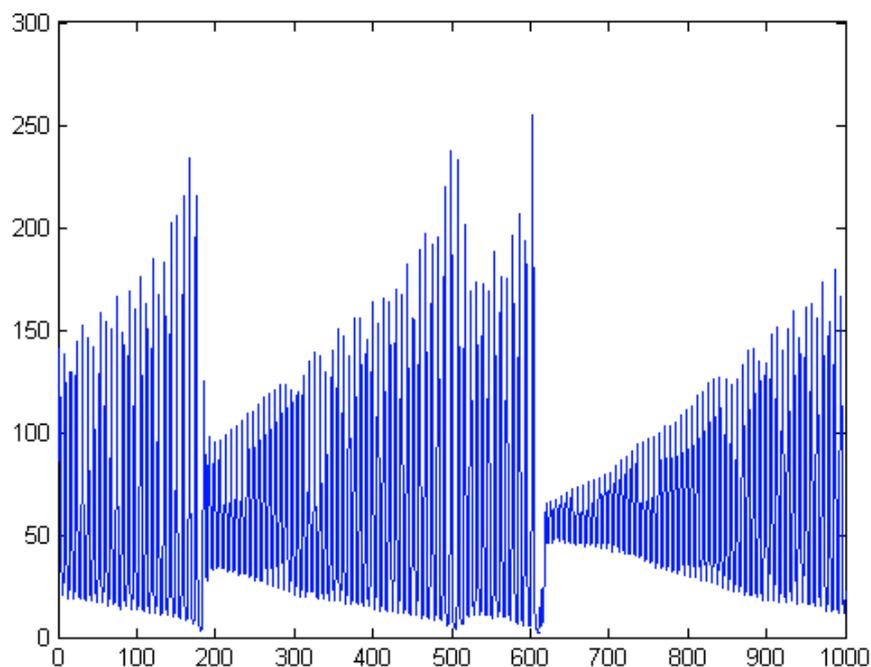


Figura 2. Série temporal A.

A série temporal Burstin (ZHU e SHASHA, 2003) é composta por uma seqüência de 2001 pontos de dados usados no estudo da detecção de ruptura. Detecção de ruptura é a atividade de encontrar agregados anormais em seqüências de dados. A série é mostrada na Figura 3 e pode ser obtida em:

<http://cs.nyu.edu/cs/faculty/shasha/papers/burst.d/burst.html>

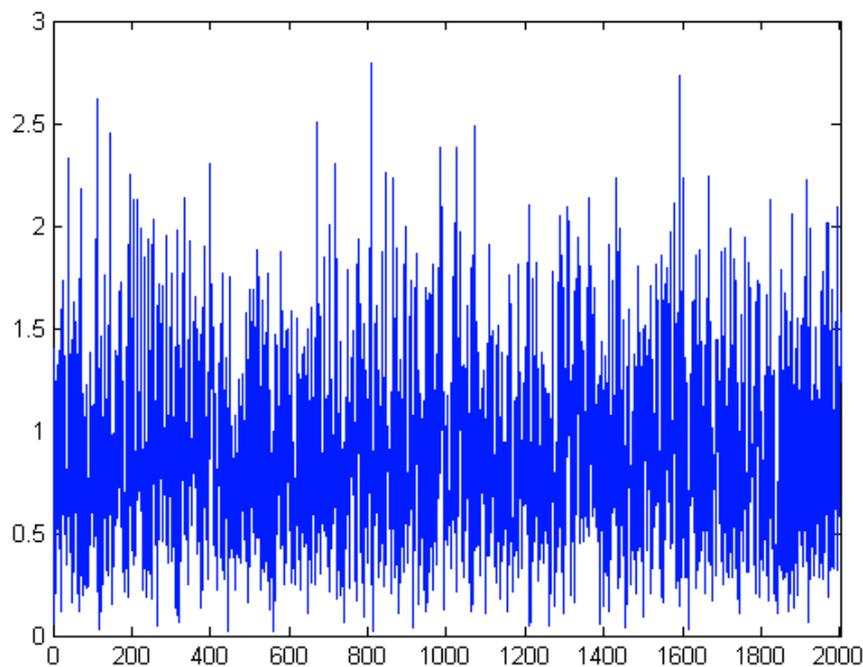


Figura 3. Série temporal Burstin.

A série temporal Darwin contém 1400 valores mensais de pressão do ar ao nível do mar em Darwin, Austrália, medidos entre 1882 e 1998. Essa série é um indicador chave de padrões climatológicos e foi utilizada em vários estudos relacionados ao El Niño e ao SOI (*Southern Oscillation Index*), que mede mudanças na pressão do ar relacionadas às temperaturas na superfície do mar. A série é mostrada na Figura 4 e pode ser obtida em:

http://www.stat.duke.edu/~mw/ts_data_sets.html

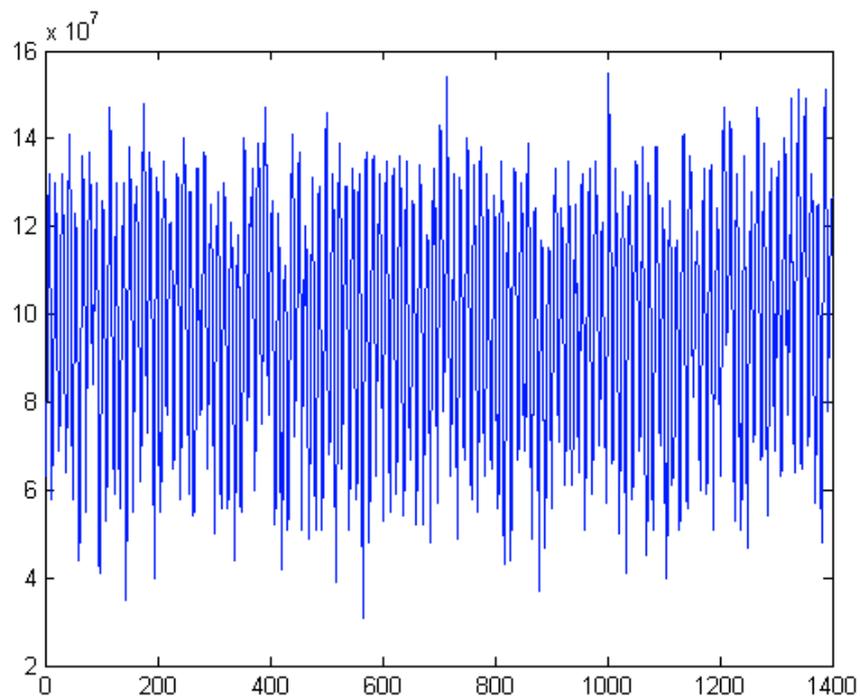


Figura 4. Série temporal Darwin.

A série temporal Earthquake (SHUMWAY e STOFFER, 2006) contém 2048 pontos de dados de um terremoto, gravados em uma estação de gravação sísmica. Os instrumentos de gravação na Escandinávia observam terremotos e explosões em minas. O problema geral de interesse está em distinguir ou discriminar entre as formas de ondas geradas por terremotos e aquelas geradas por explosões. A série é mostrada na Figura 5 e pode ser obtida em:

<http://lib.stat.cmu.edu/general/tsa/tsa.html>

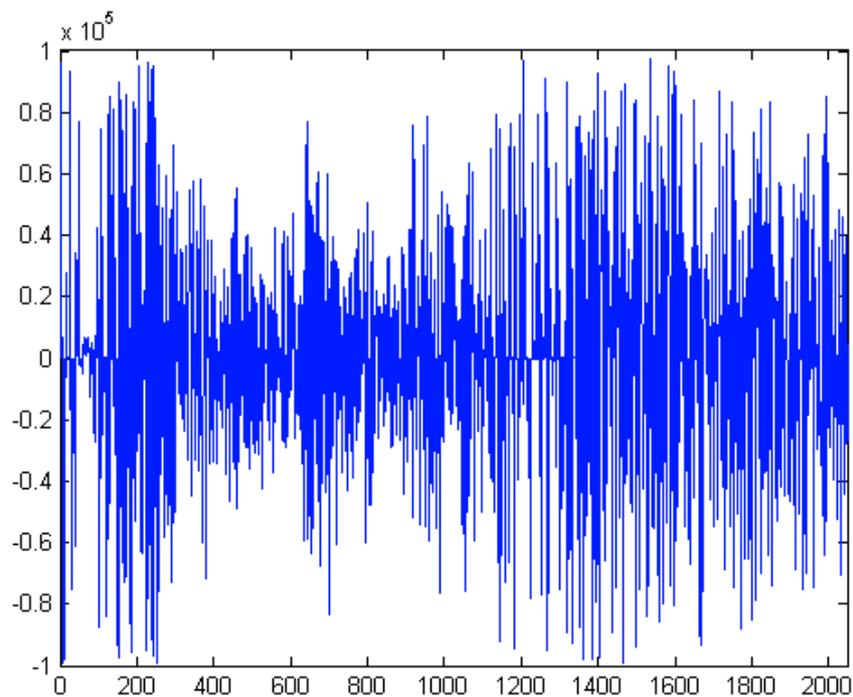


Figura 5. Série temporal Earthquake.

A série temporal Leuven contém 2000 pontos de dados gerados por um circuito de Chua generalizado, o circuito eletrônico mais simples que exibe caos, como verificado em muitos experimentos de laboratório, simulações computacionais e análise matemática rigorosa. A série foi usada na competição de previsões de K.U. Leuven (SUYKENS e VANDEWALLE, 2000). A série é não-linear com muitos pontos de quebra ocorrendo aleatoriamente, similar à série A, mas as oscilações não são tão regulares e tem diferentes deslocamentos determinados por cinco atratores. A série é mostrada na Figura 6 e pode ser obtida em:

<ftp://ftp.esat.kuleuven.ac.be/pub/sista/suykens/workshop/>

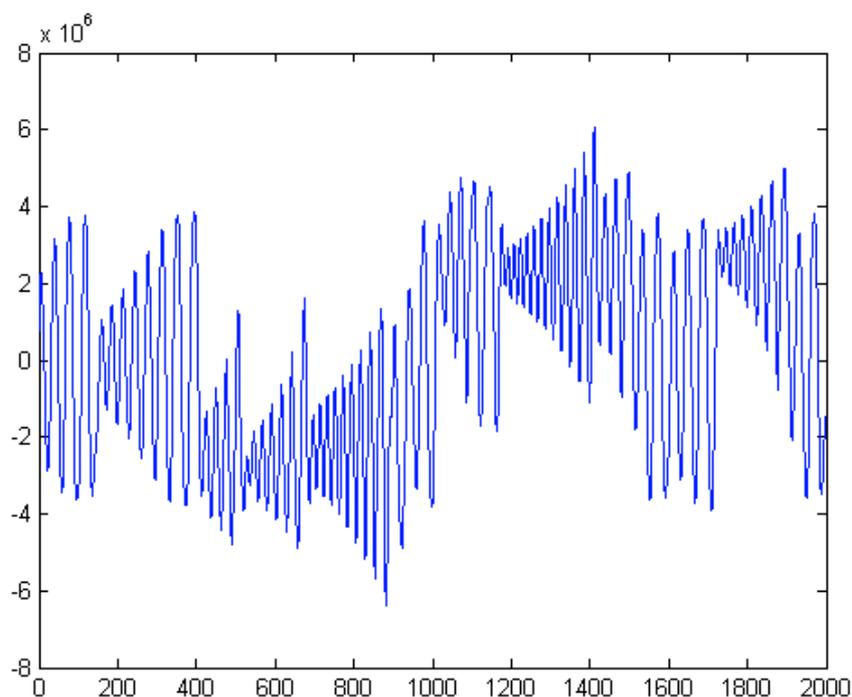


Figura 6. Série temporal Leuven.

Series 1, Series 2 e Series 3 (TEIXEIRA, 2005) são séries de carga elétrica mensal obtidas de empresas brasileiras de energia elétrica e apresentam um comportamento de mudança abrupta e significativa em seus últimos anos, quando ocorre um racionamento de energia. Cada uma das séries contém dados de 8 anos (96 pontos) que são mostrados nas Figuras 7, 8 e 9.

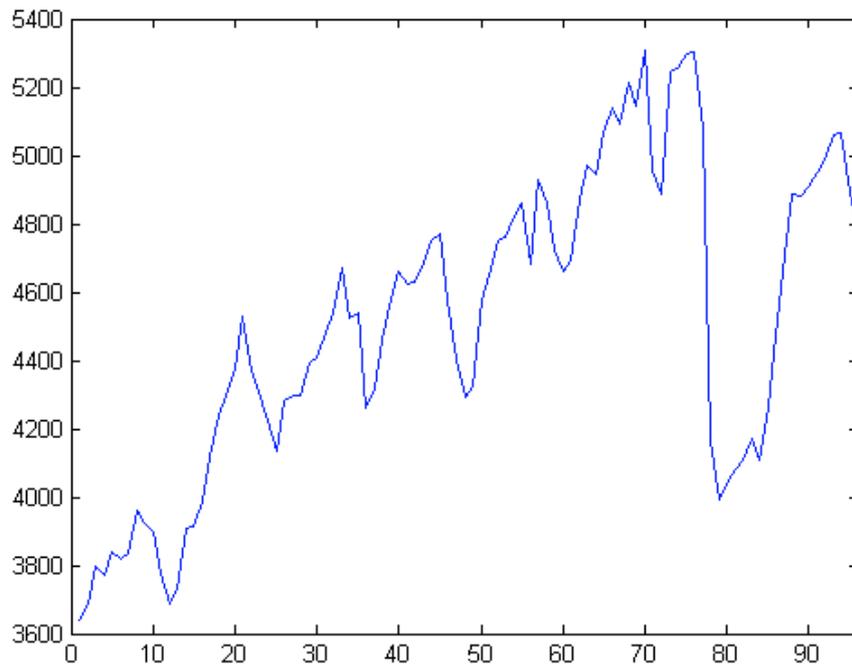


Figura 7. Série temporal Series 1.

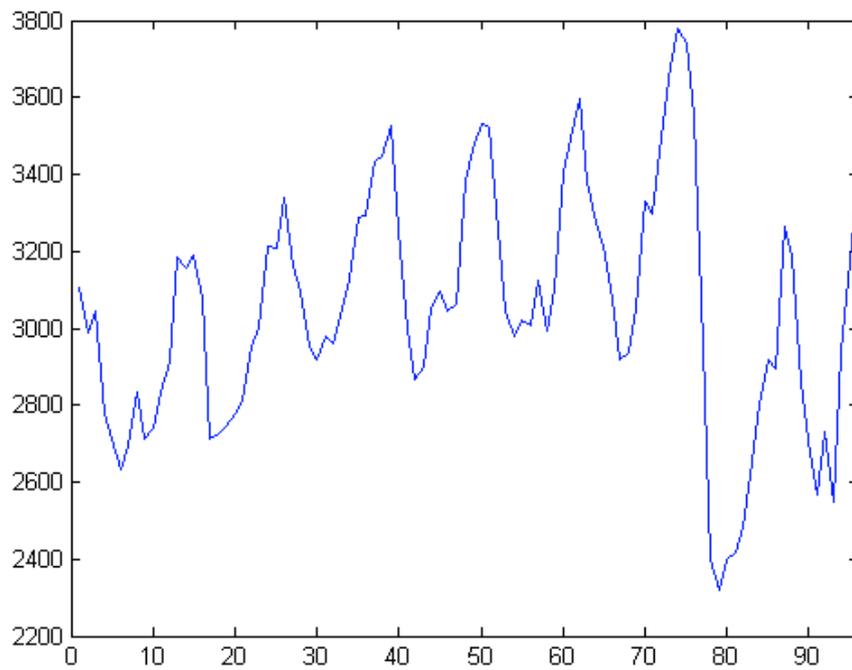


Figura 8. Série temporal Series 2.

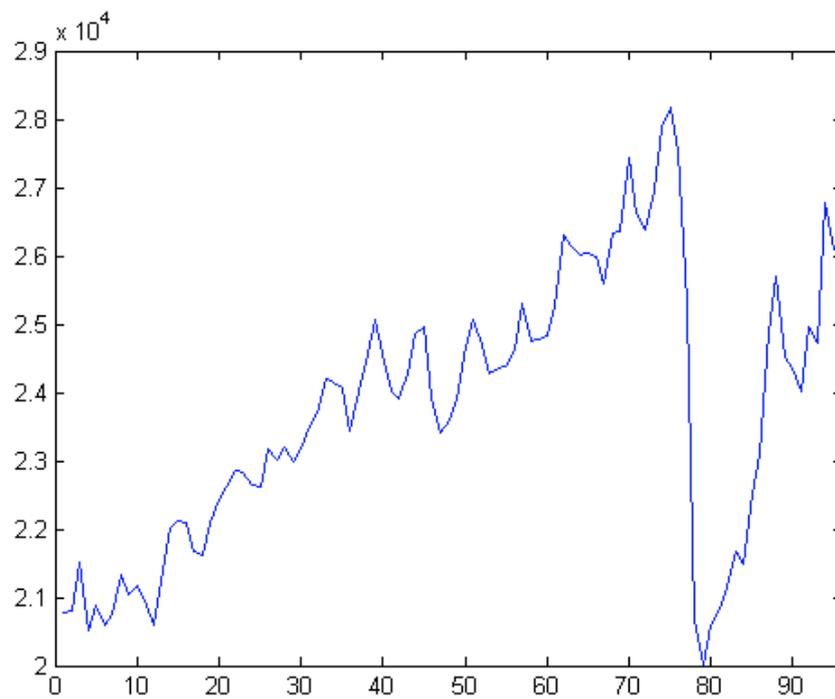


Figura 9. Série temporal Series 3.

A série temporal Soiltemp contém 2034 pontos de dados que representam a temperatura do solo em um campo agrícola. A série é mostrada na Figura 10 e pode ser obtida em:

<http://lib.stat.cmu.edu/general/tsa/tsa.html>

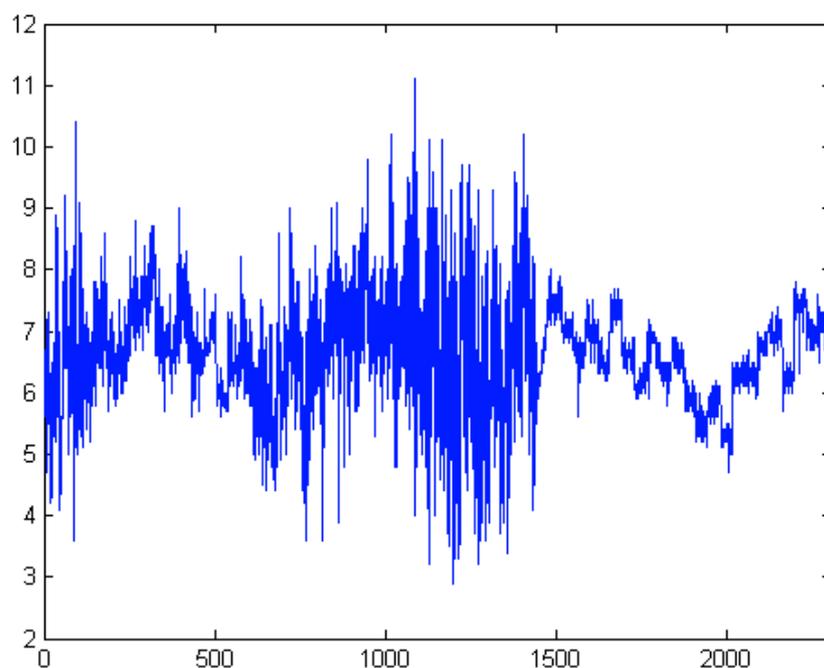


Figura 10. Série temporal Soiltemp.

A série temporal Speech (SHUMWAY e STOFFER, 2006) contém 1000 pontos de dados que representam uma pequena amostra de 0.1 segundo de voz gravada dizendo *aaa...hhh*, contendo uma complicada mistura de frequências relacionadas à abertura e fechamento da glote. O reconhecimento de fala é um problema de grande interesse, que requereria converter esse sinal em particular na palavra *aaa...hhh*. A série é mostrada na Figura 11 e pode ser obtida em:

<http://lib.stat.cmu.edu/general/tsa/tsa.html>

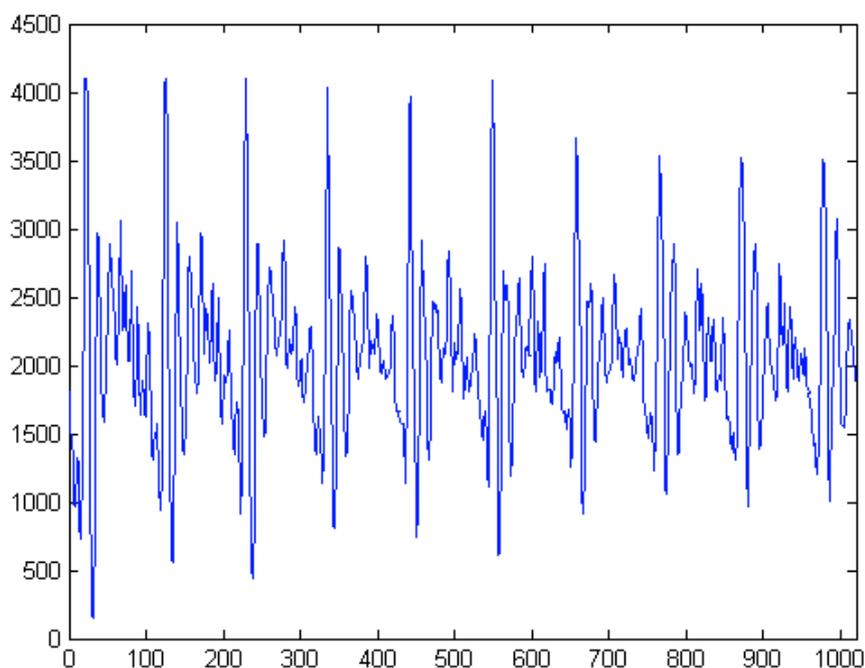


Figura 11. Série temporal Speech.

Os resultados dos testes empíricos realizados na tomada de decisões de projeto e na avaliação do sistema final foram comparados através da análise de curvas REC e sua significância foi medida seguindo a metodologia recomendada por Demsar (2006), utilizando o teste de Friedman (FRIEDMAN, 1937) para detectar se houve diferença significativa entre os resultados, seguido pelo teste de Nemenyi (NEMENYI, 1963), para tomar decisões de projeto, ou pelo teste de Holm (HOLM, 1979), para comparar o sistema proposto com a estado-da-arte em filtragem de Kalman e filtragem por partículas. Nos experimentos onde apenas dois modelos foram comparados um com o outro, foi usado o teste de Wilcoxon (WILCOXON, 1945). A hipótese nula é de que os algoritmos sendo comparados alcançam performances estatisticamente iguais. Se a hipótese nula é rejeitada, houve diferença significativa entre os algoritmos. Os testes de Wilcoxon, Friedman, Nemenyi e Holm são apresentados no Apêndice.

6.1. Algoritmo Básico

O algoritmo básico do filtro de partículas não-paramétrico é apresentado em Algoritmo 1 (PINA, 2006d). A inferência no sistema é realizada por um filtro de partículas. Logo, a densidade posterior filtrada $p(x_t | y_{1:t})$ pode ser aproximada por:

$$p(x_t | y_{1:t}) \approx \sum_{i=1}^N w_t^i \delta(x_t - x_t^i) \quad (96)$$

e a atualização dos pesos é dada por:

$$w_t^i \propto w_{t-1}^i \frac{p(y_t | x_t^i) p(x_t^i | x_{t-1}^i)}{q(x_t^i | x_{t-1}^i, y_t)} \quad (97)$$

O método de inferência escolhido foi o de filtragem por partículas pois são mais gerais que os filtros de Kalman e podem ser aplicados a uma gama maior de problemas. No entanto, o uso de filtros de partículas introduz uma necessidade de se definir a função de transição de estados e a função das observações, que determinam a dinâmica do sistema. Essa necessidade foi contornada estimando-se todas as densidades necessárias não-parametricamente com kernels.

O algoritmo EM funciona da seguinte forma: no passo E são inferidas as estimativas dos estados dadas as observações, ou seja, as probabilidades de filtragem. A suavização das estimativas é um passo opcional.

Essas estimativas são então usadas no passo M para atualizar os conjunto de *kernels* que representam as funções de transição de estados e das observações (μ e ν).

Algoritmo 1. Algoritmo básico do filtro de partículas não-paramétrico.

Inicialização do Modelo: Inicialize $\lambda = \{\mu, \nu\}$ com conjuntos de amostras de dimensões apropriadas. Escolha o tamanho do conjunto de amostras $N > 0$.

Passo E:

1. Para $t = 1$ até T faça inferência com um filtro de partículas, usando como função de transição de estados e função das observações suas representações com kernels, dadas respectivamente pelos conjuntos μ e ν .

2. Para $t = T - 1$ até 1 faça a suavização das estimativas.

Passo M:

1. Atualização de μ : Escolha N tempos aleatórios $t \in \{1, \dots, T - 1\}$, gere amostras x^t e $x^{t'}$ a partir das estimativas [suavizadas] usando estimação com *kernels* e as adicione no conjunto de amostras que representa μ .

2. Atualização de ν : Escolha N tempos aleatórios $t \in \{1, \dots, T\}$, gere amostras a partir das estimativas [suavizadas] usando estimação com *kernels* e as adicione no conjunto de amostras que representa ν .

6.2. Densidade de Importância

A densidade de importância deve ser tal que seja possível amostrar a partir dela, no passo de atualização do tempo, e que seja possível avaliá-la, no passo de atualização das observações, mais especificamente, no cálculo dos pesos das novas partículas, como foi visto na Seção 3.1.

Usar a densidade de importância ótima não é fácil, por causa dos motivos descritos na Seção 3.3. Portanto foi escolhida uma densidade de importância subótima, a priori de transição de estados $p(x_t | x_{t-1})$. Usando a regra de Bayes, tem-se que:

$$p(x_t | x_{t-1}) = \frac{p(x_t, x_{t-1})}{\int p(x_t, x_{t-1}) dx_t} \quad (98)$$

Foram usados estimadores com kernels bivariados para aproximar $p(x_t, x_{t-1})$:

$$\hat{p}(x_t, x_{t-1}^i) = \frac{1}{Nh_{x_t} h_{x_{t-1}}} \sum_{j=1}^N K\left(\frac{x_t - x_t^j}{h_{x_t}}\right) K\left(\frac{x_{t-1}^i - x_{t-1}^j}{h_{x_{t-1}}}\right) \quad (99)$$

Para aproximar $\int p(x_t, x_{t-1}) dx_t$ foram usados estimadores com kernels univariados:

$$\int \hat{p}(x_t, x_{t-1}^i) dx = \frac{1}{Nh_{x_{t-1}}} \sum_{j=1}^N K\left(\frac{x_{t-1}^i - x_{t-1}^j}{h_{x_{t-1}}}\right) \quad (100)$$

Portanto, para amostrar a partir da priori de transição de estados $p(x_t | x_{t-1})$ a seguinte representação com kernels foi utilizada:

$$p(x_t | x_{t-1}) = \frac{\frac{1}{Nh_{x_t} h_{x_{t-1}}} \sum_{j=1}^N K\left(\frac{x_t - x_t^j}{h_{x_t}}\right) K\left(\frac{x_{t-1}^i - x_{t-1}^j}{h_{x_{t-1}}}\right)}{\frac{1}{Nh_{x_{t-1}}} \sum_{j=1}^N K\left(\frac{x_{t-1}^i - x_{t-1}^j}{h_{x_{t-1}}}\right)} \quad (101)$$

Simplificando e rearrumando a fórmula, tem-se:

$$p(x_t | x_{t-1}) = \frac{\frac{1}{h_{x_t}} \sum_{j=1}^N K\left(\frac{x_t - x_t^j}{h_{x_t}}\right) K\left(\frac{x_{t-1}^i - x_{t-1}^j}{h_{x_{t-1}}}\right)}{\sum_{j=1}^N K\left(\frac{x_{t-1}^i - x_{t-1}^j}{h_{x_{t-1}}}\right)} \quad (102)$$

De forma análoga, os pesos das partículas são escolhidos usando-se a representação com kernels bivariada de $p(y_t | x_t^i)$.

6.3. Passo de Reamostragem

A segunda decisão de projeto tomada foi referente à necessidade de um passo de reamostragem na filtragem por partículas. Para esse propósito, foram testadas três versões do sistema, uma sem reamostragem, uma com reamostragem a cada instante de tempo, e uma em que é feita reamostragem quando o número efetivo de partículas, como descrito na Seção 3.2, e dado por:

$$\hat{N}_{ef} = \frac{1}{\sum_{i=1}^N (w_i^i)^2} \quad (103)$$

cai abaixo de um valor limiar, selecionado como 50% do número total de partículas.

As Figuras 12, 13 e 14 mostram três exemplos de gráficos REC obtidos. O valor de limiar igual a 0 corresponde à versão do algoritmo sem passo de reamostragem e o valor de limiar igual a 1 corresponde à versão do algoritmo em que o passo de reamostragem é aplicado a cada instante de tempo. Foi possível notar que não existe um ranking de performance claro entre as três versões, que valha para a maioria das séries testadas.

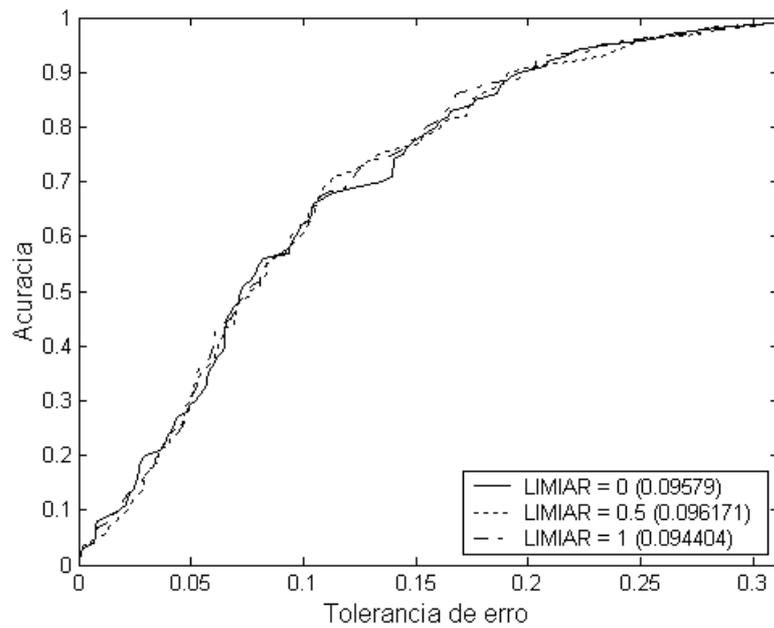


Figura 12. Gráfico REC para a série temporal Burstin. Limiar = 0 significa que não foi usado passo de reamostragem e limiar = 1 significa que é aplicada reamostragem a cada instante de tempo.

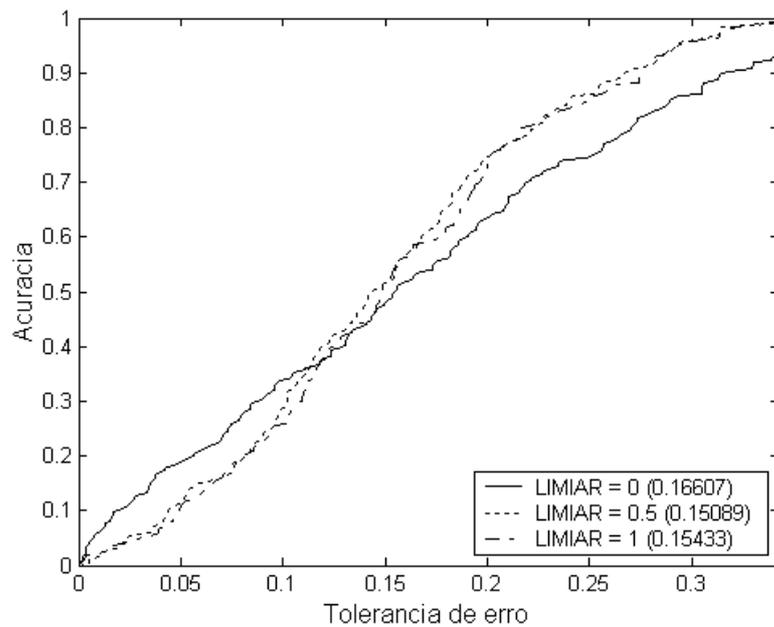


Figura 13. Gráfico REC para a série temporal Leuven (limiar = 0, sem reamostragem, e limiar = 1, reamostragem a cada instante de tempo).

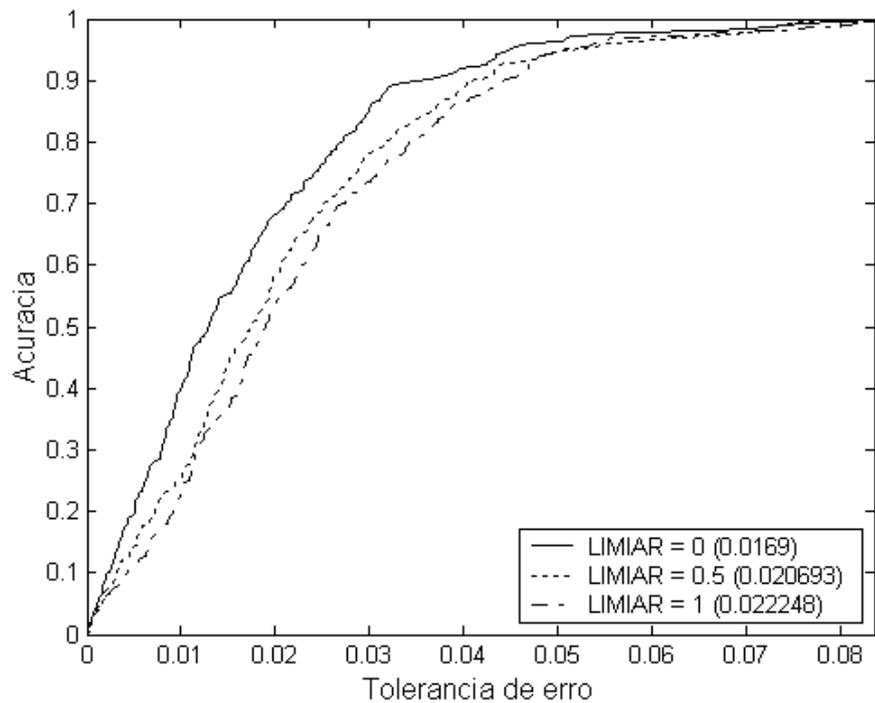


Figura 14. Gráfico REC para a série temporal Soiltemp (limiar = 0, sem reamostragem, e limiar = 1, reamostragem a cada instante de tempo).

Embora a análise das curvas REC não mostre claramente a superioridade no uso de um dos valores de limiar e o teste de Friedman não tenha detectado significância estatística na diferença entre eles, foi decidido utilizar reamostragem com limiar de 0,5, por motivos teóricos, como forma de amenizar os efeitos do fenômeno da degeneração, sem no entanto causar a rápida perda de diversidade das partículas.

6.4. Suavização

A suavização em filtros de partículas raramente recebeu qualquer atenção, por duas razões. A primeira é que os algoritmos de suavização, tais como o Suavizador Forward-Backward e o Suavizador com Dois Filtros, descritos na Seção 3.8, têm complexidade computacional da ordem de N^2 , onde N é o número de partículas, geralmente grande. Deve-se notar que, em contraste, o filtro de partículas tem complexidade $O(N)$. A segunda é que os Suavizadores com Dois Filtros existentes fazem suposições que são difíceis de se verificar ou satisfazer na prática.

A despeito disso, foram realizados testes com os dois suavizadores mencionados a fim de verificar se sua inclusão resultaria em benefícios significativos para o sistema. No Suavizador Forward-Backward, os pesos de importância são atualizados através da recursão para trás no tempo dada pela equação:

$$w_t^i = w_t^i \left[\frac{\sum_{j=1}^N w_{t+1}^j \frac{p(x_{t+1}^j | x_t^i)}{\sum_{k=1}^N w_t^k p(x_{t+1}^k | x_t^k)}}{\sum_{k=1}^N w_t^k p(x_{t+1}^k | x_t^k)} \right] \quad (104)$$

Já no Suavizador com Dois Filtros, um segundo filtro de partículas é usado para computar $p(y_{t+1:T} | x_t)$ seqüencialmente, notando que:

$$p(y_{t:T} | x_t) = \int p(y_{t+1:T} | x_{t+1}) p(x_{t+1} | x_t) p(y_t | x_t) dx_{t+1} \quad (105)$$

A simples análise das curvas REC sugere que o sistema sem suavização alcança resultados ligeiramente melhores que as versões que utilizam suavização. Isso pode ser visto nos gráficos REC apresentados nas Figuras 15 e 16. Entretanto, o teste de Friedman não detectou diferença estatisticamente significativa entre os resultados das três versões.

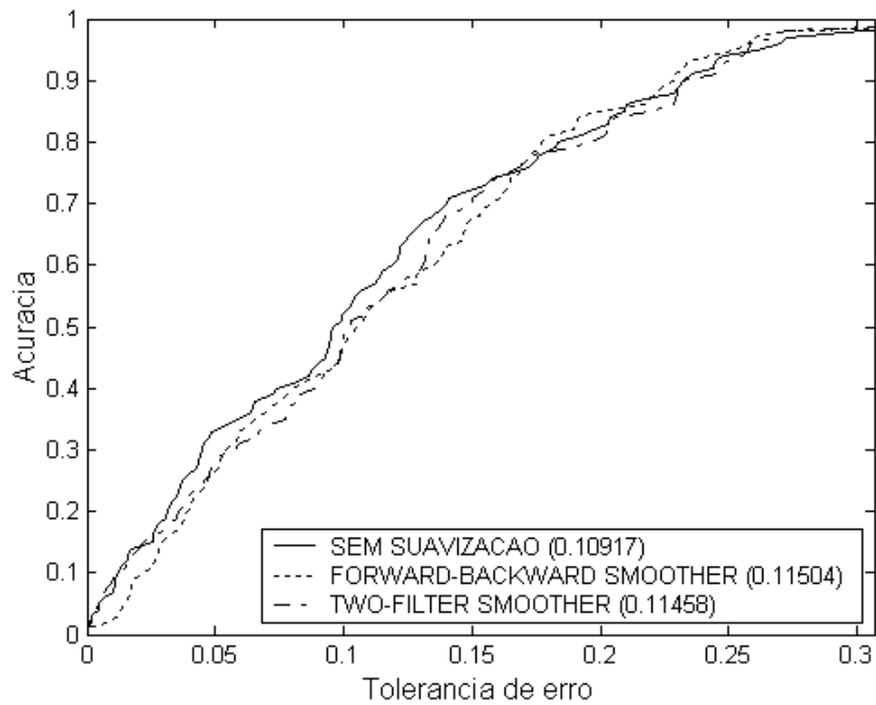


Figura 15. Gráfico REC para a série temporal Darwin.

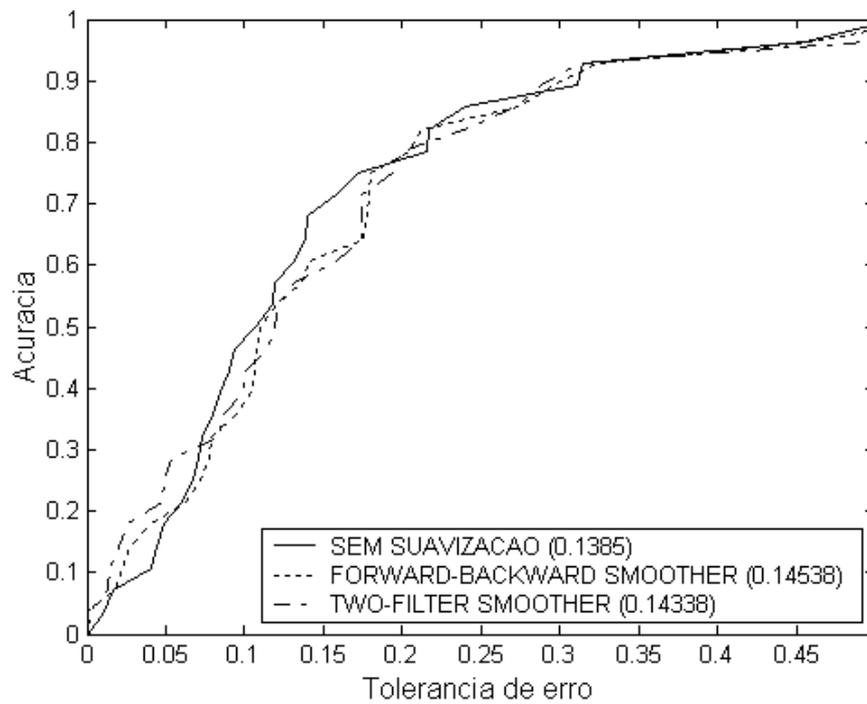


Figura 16. Gráfico REC para a série temporal Serie2.

Como não houve evidência estatística que comprovasse a necessidade de uma etapa de suavização, sua inclusão resultaria apenas em *overhead* computacional, já bastante elevado devido à natureza não-paramétrica do sistema, e portanto a idéia foi descartada.

6.5. Função Kernel

Uma vez que calcular a densidade em um dado ponto requer N avaliações da função kernel e essa operação é realizada muitas vezes no sistema proposto, foi necessário utilizar uma função kernel simples e de fácil computação. Foram feitos testes com duas funções kernel bastante populares na literatura: a função kernel Gaussiana:

$$K(t) = \frac{e^{\left(\frac{-t^2}{2}\right)}}{\sqrt{2\pi}} \quad (106)$$

e a função kernel de Epanechnikov:

$$K(t) = \begin{cases} \frac{3}{4} \cdot (1-t^2) & \text{se } |t| \leq 1 \\ 0 & \text{caso contrário} \end{cases} \quad (107)$$

A análise das curvas REC mostra clara vantagem do uso do kernel Gaussiano sobre o kernel de Epanechnikov (por exemplo, Figuras 17 e 18), exceto para as séries temporais A e Burstin, nas quais a função kernel de Epanechnikov obtém resultados ligeiramente melhores (Figuras 19 e 20).

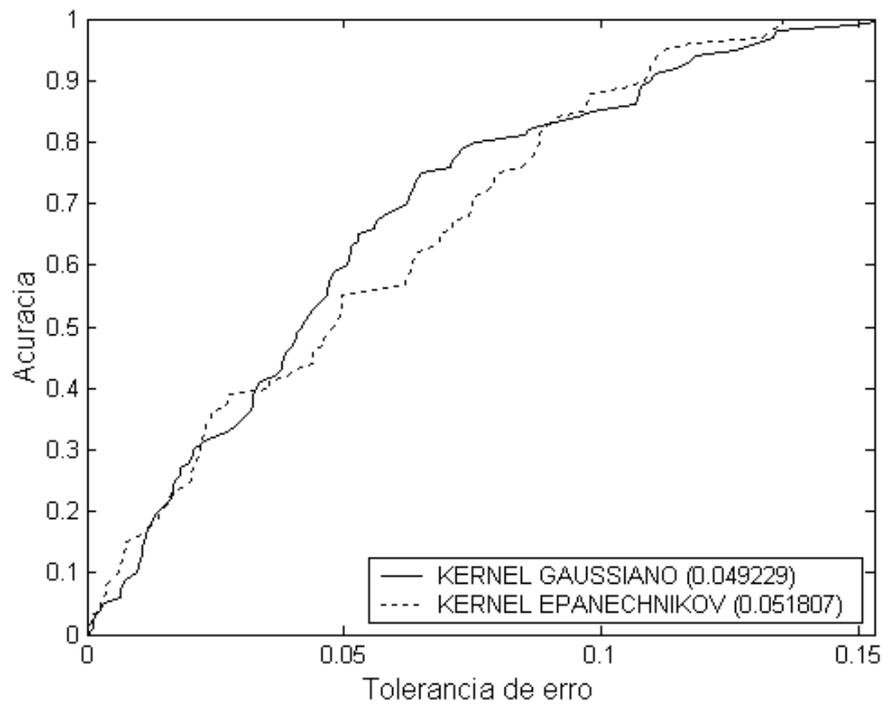


Figura 17. Gráfico REC para a série temporal Earthquake.

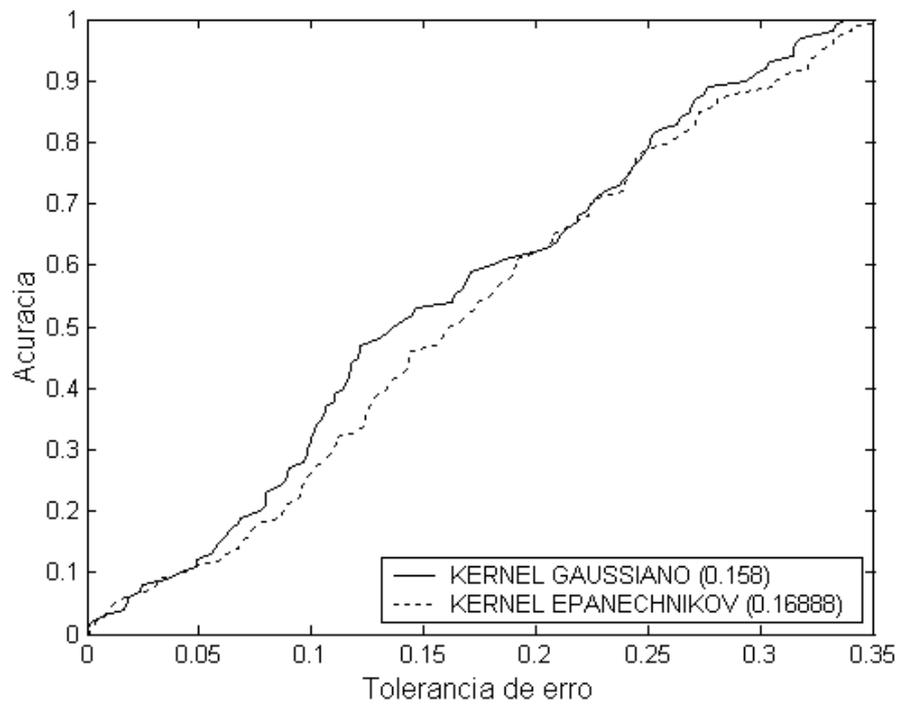


Figura 18. Gráfico REC para a série temporal Leuven.

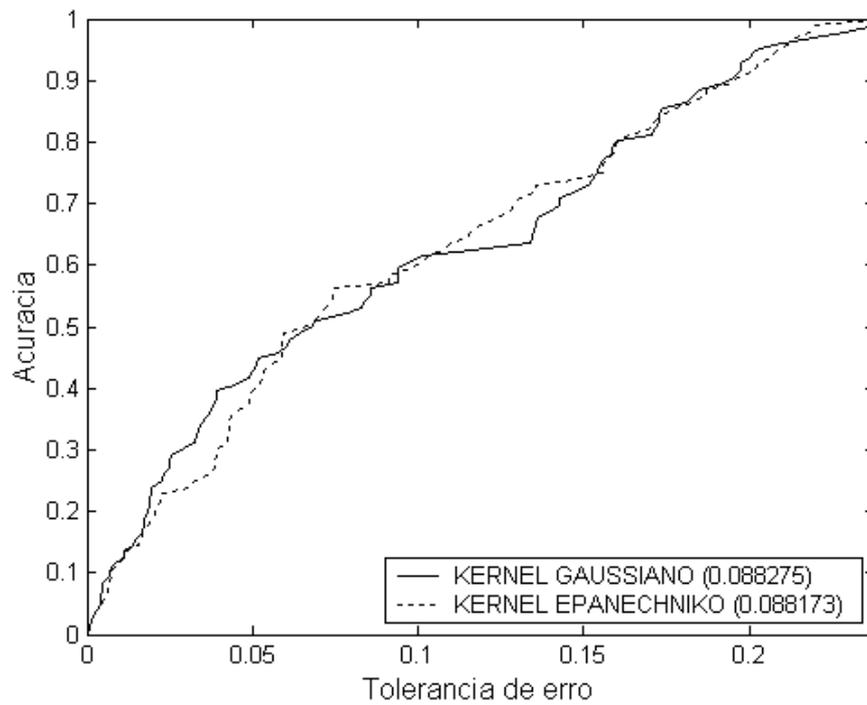


Figura 19. Gráfico REC para a série temporal A.

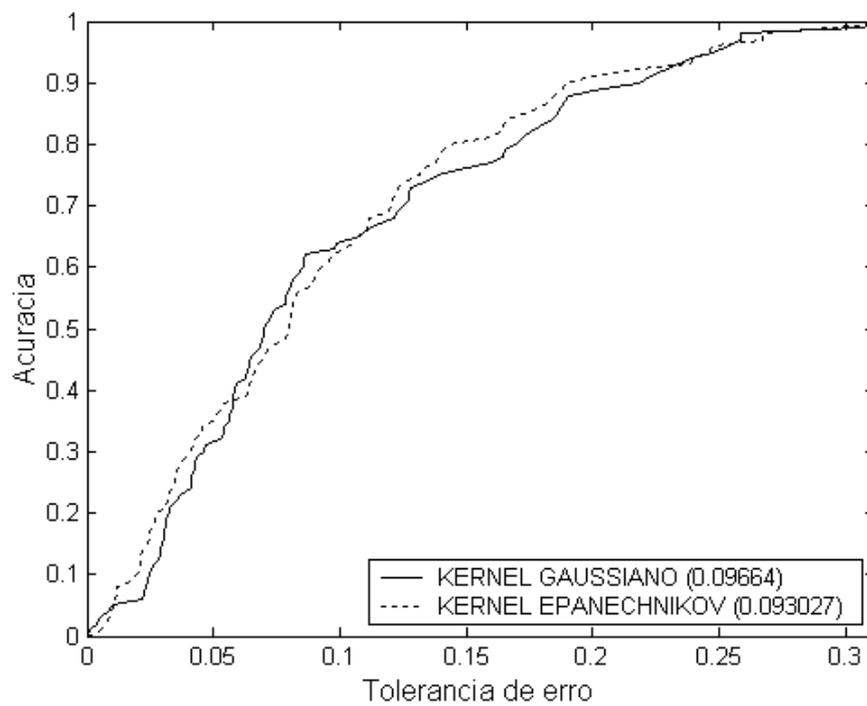


Figura 20. Gráfico REC para a série temporal Burstin.

Como apenas dois modelos foram testados para tomar esta decisão de projeto, foi usado o teste de Wilcoxon para determinar se houve diferença significativa entre os resultados de ambos. O teste mostrou que a diferença foi realmente estatisticamente significativa e portanto a função kernel Gaussiana foi selecionada como a melhor opção.

6.6. Seletor de Largura de Kernel

Uma vez que o algoritmo proposto realiza muitos passos de estimação com kernels, é desejável que as larguras também sejam facilmente computadas. Portanto, os seletores de largura de kernels mais adequados ao sistema proposto são os mais simples, mas que no entanto fornecem uma boa estimativa para uma larga faixa de aplicações. Dentre esses, três seletores de largura foram testados neste trabalho: o seletor de Escala Normal, o seletor usado no sistema WEKA (WITTEN e FRANK, 2005) e o seletor usado no sistema S-PLUS.

O seletor de escala normal é dado pela seguinte fórmula:

$$\hat{h}_{EN} = \left[\frac{8\pi^{\frac{1}{2}} R(K)}{3\mu_2(K)^2 n} \right]^{\frac{1}{5}} \hat{\sigma} \quad (108)$$

onde $\hat{\sigma}$ é uma estimativa de σ , geralmente o desvio padrão amostral, e:

$$R(K) = \int K(x)^2 dx \quad (109)$$

$$\mu_2(K) = \int x^2 K(x) dx \quad (110)$$

Uma vez que para *kernels* Gaussianos:

$$R(K) = \int K(x)^2 dx = \frac{1}{2\sqrt{\pi}} \quad (111)$$

$$\mu_2(K) = \int x^2 K(x) dx = 1 \quad (112)$$

e o desvio padrão amostral é dado por:

$$\hat{\sigma} = \left[\frac{1}{n} \sum_{i=1}^n (x_i - \bar{x}_n)^2 \right]^{1/2} \quad (113)$$

onde \bar{x}_n é a média amostral.

Então, para função *kernel* Gaussiana, selecionada na Seção 6.5, a largura de *kernel* dada pelo seletor de escala normal é dada por:

$$\hat{h}_{EN} = \left[\frac{4}{3n} \right]^{1/5} \cdot \left[\frac{1}{n} \sum_{i=1}^n (x_i - \bar{x}_n)^2 \right]^{1/2} \quad (114)$$

Os seletores de largura usados pelos sistemas WEKA e S-PLUS são baseados na faixa de valores de x , e apesar de não terem forte suporte teórico, alcançam performances comparáveis às de seletores mais complexos. O seletor de largura do WEKA é dado por:

$$\hat{h}_{WEKA} = \frac{\max(x) - \min(x)}{\sqrt{n}} \quad (115)$$

e o seletor de largura do S-PLUS é dado pela fórmula:

$$\hat{h}_{SPLUS} = \frac{\max(x) - \min(x)}{2 \cdot (1 + \log_2 n)} \quad (116)$$

A análise das curvas REC geradas não foi conclusiva, como pode ser notado nas Figuras 21 e 22. Não obstante, o teste de Friedman foi capaz de detectar diferença significativa entre eles.

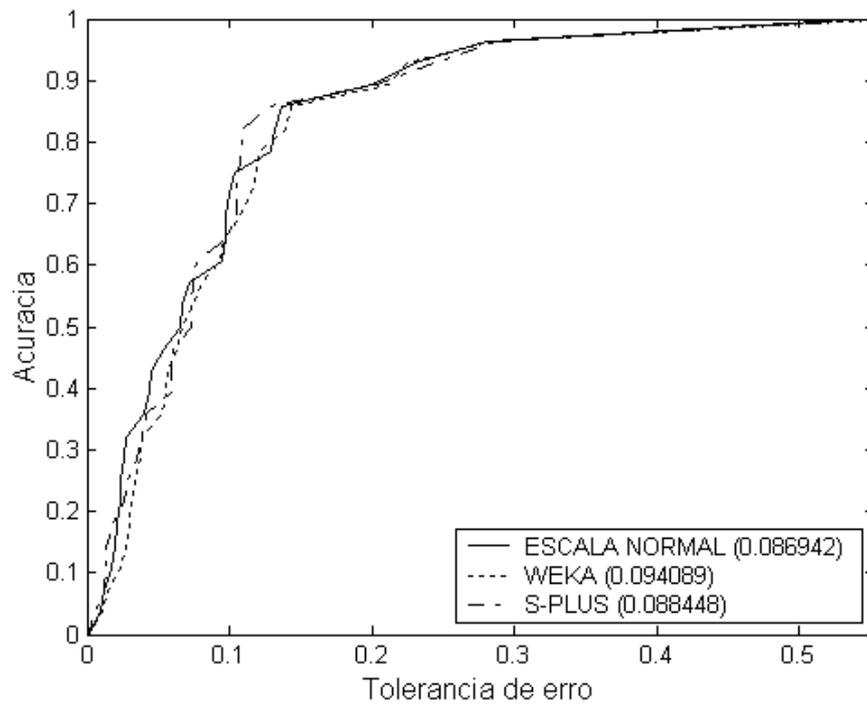


Figura 21. Gráfico REC para a série temporal Serie3.

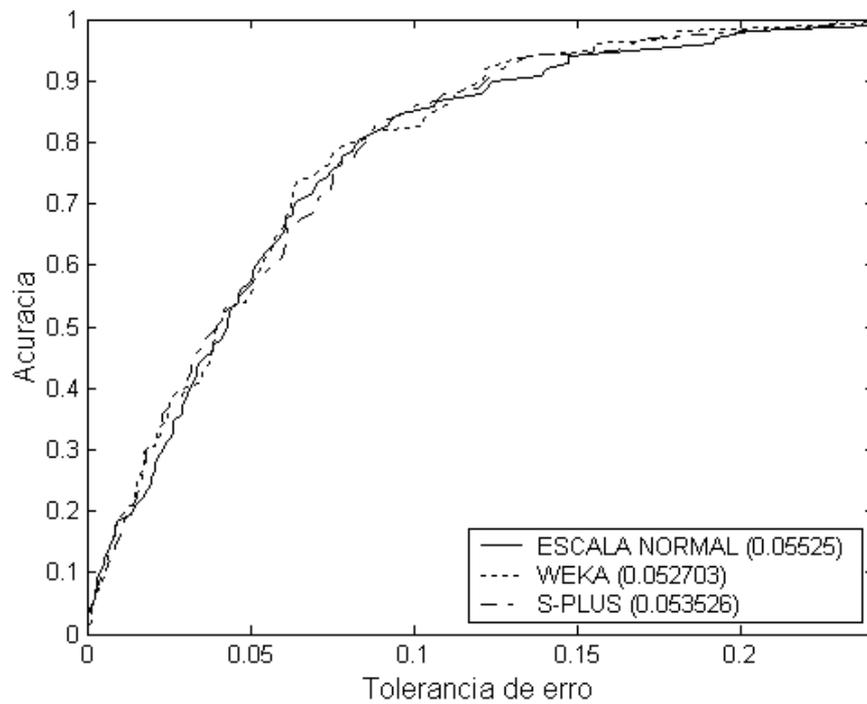


Figura 22. Gráfico REC para a série temporal Speech.

Embora o teste de Friedman tenha detectado que existe diferença estatisticamente significativa entre os resultados obtidos usando-se os três seletores de largura de kernel, o teste de Nemenyi não foi poderoso o suficiente para detectar a diferença. Foi consirado então o rank médio de cada algoritmo, mostrado na Tabela 2.

Tabela 2. Rank dos seletores de largura.

	Escala Normal	WEKA	S-PLUS
Rank médio	1,8	2,2	2,0

O rank médio sugere que o seletor que deve ser usado é o seletor de Escala Normal. Então, o teste de Holm foi aplicado usando o seletor de Escala Normal como base de comparação, na tentativa de verificar se a diferença significativa detectada pelo teste de Friedman está presente apenas entre o seletor de Escala Normal e o seletor do WEKA, ou se também está presente entre o seletor de Escala Normal e o seletor do S-PLUS. O teste de Holm também foi inconclusivo, portanto o seletor de Escala Normal foi escolhido como parte do sistema baseado no fato de que ele apresentou um rank médio melhor e, considerando o teste de Friedman, obteve uma performance significativamente maior que pelo menos um dos outros seletores testados.

Dada esta última decisão de projeto, o algoritmo final do Filtro de Partículas Não-Paramétrico é mostrado em Algoritmo 2.

Algoritmo 2. Algoritmo do filtro de partículas não-paramétrico.

Inicialização do Modelo: Inicialize $\lambda = \{\mu, \nu\}$ com conjuntos de amostras de dimensões apropriadas. Escolha o tamanho do conjunto de amostras $N > 0$.

Passo E:

1. Inicialize α_0 com amostras aleatórias a partir dos dados.
2. Para $t = 1$ até T faça:
 - Para cada amostra x_{t-1}^i de α_{t-1} , gere a densidade condicional $p(x_t | x_{t-1}^i)$ usando estimação com *kernels* (usando kernels Gaussianos e seletor de largura de escala normal). Amostre um único x_t^i a partir dessa densidade.
 - Atribua a w_t^i um valor proporcional a $p(y_t | x_t^i)$ usando sua representação com *kernels* (usando kernels Gaussianos e seletor de largura de escala normal).
 - Normalize os pesos: $w_t^i = \frac{\sum_{i=1}^N w_t^i}{N}$
 - Calcule $\hat{N}_{ef} = \frac{1}{\sum_{i=1}^N (w_t^i)^2}$
 - Se $\hat{N}_{ef} < 0,5N$ então
 - Reamostre N amostras a partir do novo conjunto representando α_t .

Passo M:

1. Atualização de μ : Escolha N tempos aleatórios $t \in \{1, \dots, T-1\}$, gere amostras x^t e $x^{t'}$ a partir das estimativas usando estimação com *kernels* e as adicione no conjunto de amostras que representa μ .
2. Atualização de ν : Escolha N tempos aleatórios $t \in \{1, \dots, T\}$, gere amostras a partir das estimativas usando estimação com *kernels* e as adicione no conjunto de amostras que representa ν .

6.7. Avaliação Experimental do Sistema Final

Foram utilizados na avaliação experimental: o Filtro de Partículas Não-Paramétrico (FNP), o Filtro de Kalman Estendido (EKF), os Filtros de Kalman com Pontos Sigma - Filtro de Kalman *Unscented* (UKF), Filtro de Kalman com Diferença Central (CDKF) e suas formas raiz quadrada (SRUKF e SRCDKF, versões otimizadas do UKF e do CDKF, respectivamente), o Filtro de Partículas com Reamostragem (SIR), o Filtro de Partículas com Somas de Gaussianas (GSPF), os Filtros de Partículas com Pontos Sigma (SPPF, usando SRUKF e SRCDKF para gerar a densidade de importância) e o Filtro de Partículas com Pontos Sigma e Misturas de Gaussianas (GMSPPF). As implementações dos filtros de Kalman com pontos sigma e dos filtros de partículas com pontos sigma foram obtidas a partir do pacote ReBEL, sob licença acadêmica de OGI School of Science and Engineering, Rudolph van der Merwe e Eric A. Wan. Todos os algoritmos foram implementados em Matlab e os testes foram realizados em um computador Intel Core 2 Duo 6300, 1.86GHz, 1GB de RAM.

Os valores dos parâmetros dos Filtros de Kalman com Pontos Sigma foram escolhidos de acordo com a recomendação de seus autores: para o Filtro de Kalman *unscented* e sua forma raiz quadrada, foram escolhidos $\alpha = 1$, $\beta = 2$, e $k = 0$; para o Filtro de Kalman com Diferença Central e sua forma raiz quadrada, foi escolhido $h = \sqrt{3}$.

Determinar o número de partículas necessárias é um problema muito difícil quando se trabalha com filtros de partículas. Os limites citados na literatura de filtros de partículas são muito vagos para serem de interesse prático. Existem vários artigos em que os autores utilizaram um baixo número de partículas e ainda assim conseguiram

bons resultados. Foram realizados testes aumentando-se gradativamente o número de partículas usadas até haver estabilização dos resultados. Como o esforço computacional requerido pelo sistema proposto é um fator importante que precisa ser considerado, o número de partículas escolhido foi razoavelmente baixo: $N = 200$. Para manter a comparação justa, todos os filtros de partículas utilizados nas avaliações experimentais usaram o mesmo número de partículas.

Para avaliar a performance da nova abordagem, o Filtro de Partículas Não-Paramétrico foi comparado aos Filtros de Kalman e aos Filtros de Partículas tradicionais utilizando também estimação não-paramétrica, com redes neurais, para modelar a dinâmica dos sistemas. Foram usadas redes neurais *feedforward* com função de ativação tangente hiperbólica na camada escondida e função de ativação linear no neurônio de saída, como mostrado na Figura 23 ($W1$ é a matriz de pesos sinápticos entre a entrada e a camada escondida e $W2$ é o vetor de pesos entre a camada escondida e o neurônio de saída). O treinamento foi realizado com o algoritmo de retropropagação do erro (RUMELHART *et al.*, 1986) e o valor de p foi determinado usando a abordagem *wrapper* (KOHAVI e JOHN, 1997), escolhendo o valor que proporcionou melhor resultado no conjunto de validação. No modelo de espaço de estados correspondente, a função das observações é dada por $Y_t = X_t$.

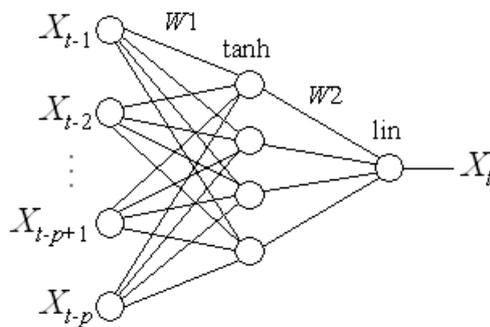


Figura 23. Rede neural usada para modelar a dinâmica dos sistemas.

A Tabela 3 contém os valores de AOC das curvas REC geradas para o Filtro de Partículas Não-Paramétrico (FNP) e para os filtros de Kalman tradicionais usados nesta pesquisa: o Extended Kalman Filter (EKF) e a família dos filtros de Kalman com Pontos Sigma (UKF, CDKF, SRUKF, SRCDKF). Repare que o CDKF e sua versão otimizada, o SRCDKF, obtiveram performances idênticas, ao contrário do UKF e do SRUKF. Isso provavelmente se deve à estabilidade numérica do CDKF, maior que do UKF e teoricamente uma de suas vantagens. Analisando os valores na tabela é possível concluir que o FNP alcançou uma performance melhor que os filtros de Kalman em oito das séries estudadas.

Tabela 3. AOC's das curvas REC para o FPNF e para os filtros de Kalman.

Série Temporal	FNP	EKF	UKF	CDKF	SRUKF	SRCDKF
A	0,086743	0,26615	0,14579	0,13216	0,14954	0,13216
Burstin	0,09336	1,6815	0,14242	0,1437	0,14242	0,1437
Darwin	0,11054	1,3019	0,11711	0,11101	0,11711	0,11101
Earthquake	0,049648	0,31558	0,088879	0,20975	0,08003	0,20975
Leuven	0,16498	0,44577	0,1548	0,18579	0,1407	0,18579
Series 1	0,077789	3,8396	0,11234	0,12468	0,12102	0,12468
Series 2	0,14105	1,6903	0,13977	0,17075	0,13991	0,17075
Series 3	0,094089	0,13904	0,1276	0,16197	0,12759	0,16197
Soiltemp	0,018666	0,023886	0,039946	0,066588	0,031505	0,066588
Speech	0,052703	0,61147	0,075437	0,20336	0,074109	0,20336

Um bom exemplo da performance superior alcançada pelo FPNP é mostrado nas Figuras 24 e 25, para a série A. Pode-se notar que o modelo gerado pelo FPNP domina todos os demais sem sombra de dúvida.

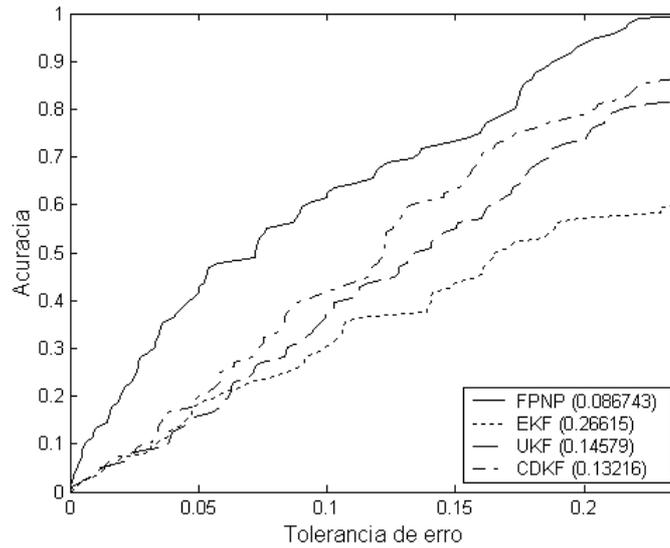


Figura 24. FPNP, EKF, UKF e CDKF aplicados à série temporal A.

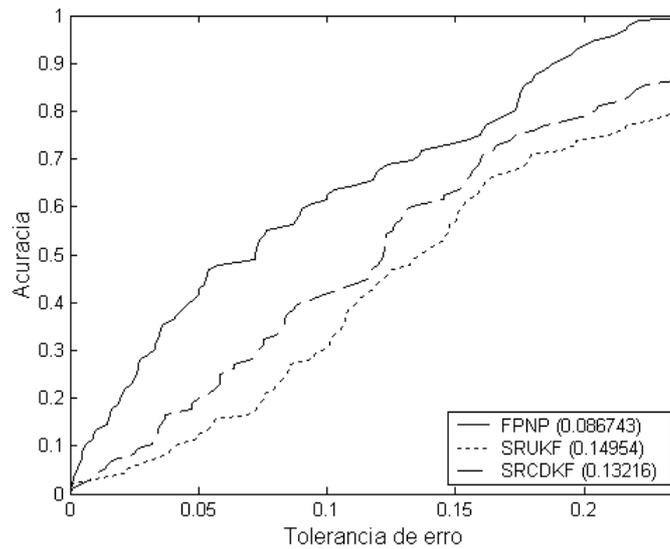


Figura 25. FPNP, SRUKF e SRCDFK aplicados à série temporal A.

A pior performance do FPNP foi verificada quando aplicado à série temporal Leuven, para a qual o SRUKF obteve o melhor resultado, seguido por sua versão sem otimização, o UKF. Deve-se notar porém, analisando as curvas REC mostradas nas Figuras 26 e 27, que a diferença entre as performances não foi tão grande, principalmente para baixas tolerâncias de erro.

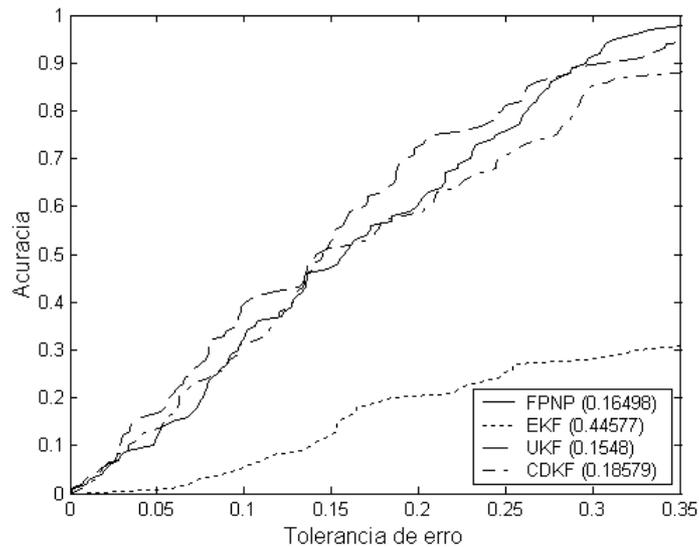


Figura 26. FPNP, EKF, UKF e CDKF aplicados à série temporal Leuven.

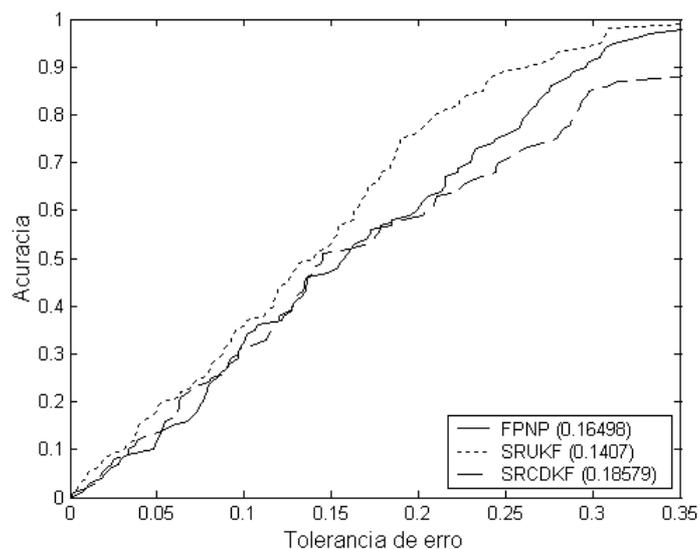


Figura 27. FPNP, SRUKF e SRCDKF aplicados à série temporal Leuven.

A outra série para a qual o FPNP não foi o melhor algoritmo foi a série temporal Series 2, novamente superado pelo SRUKF e pelo UKF, sendo que estes inverteram posições, desta vez com o UKF em primeiro e o SRUKF em segundo, com pouca diferença, como pode ser visto nas Figuras 28 e 29.

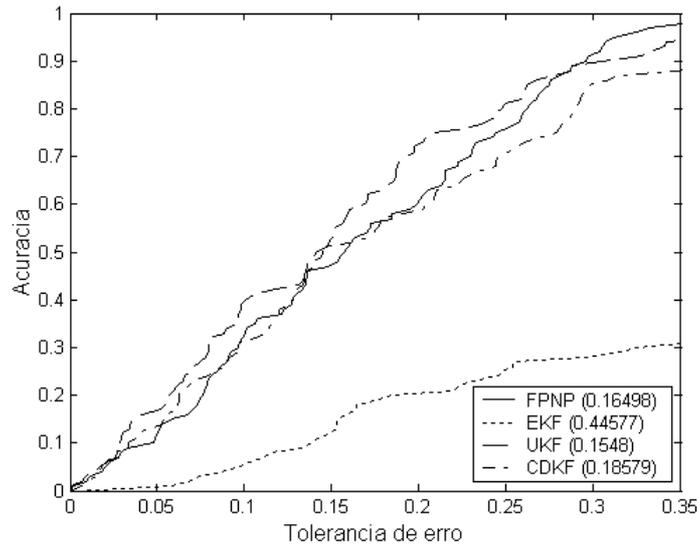


Figura 28. FPNP, EKF, UKF e CDKF aplicados à série temporal Series 2.

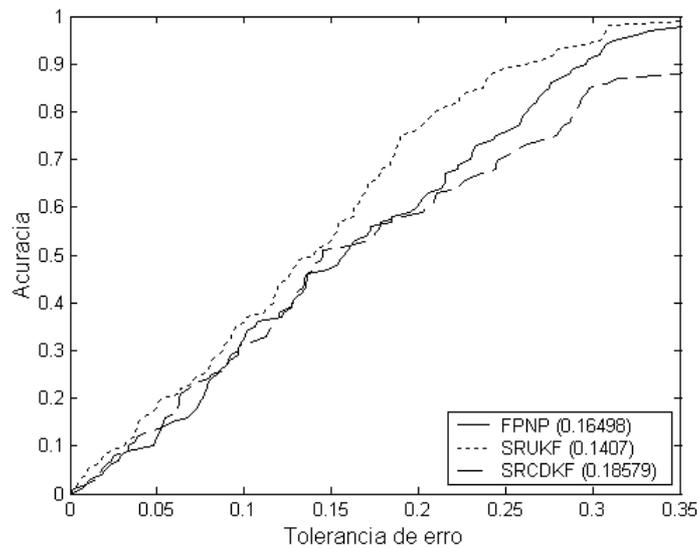


Figura 29. FPNP, SRUKF e SRCDKF aplicados à série temporal Series 2.

A pior performance de todos os algoritmos, foi claramente do EKF, sendo o último no ranking para oito das dez séries temporais testadas. Isso é explicado pela dificuldade do EKF em lidar com as não-linearidades das séries. Um caso quase extremo pode ser visto na Figura 30, para a série temporal Darwin. Não obstante, o EKF obteve uma ótima performance para a série Soiltemp, sendo superado apenas pelo FPNP (Figura 31).

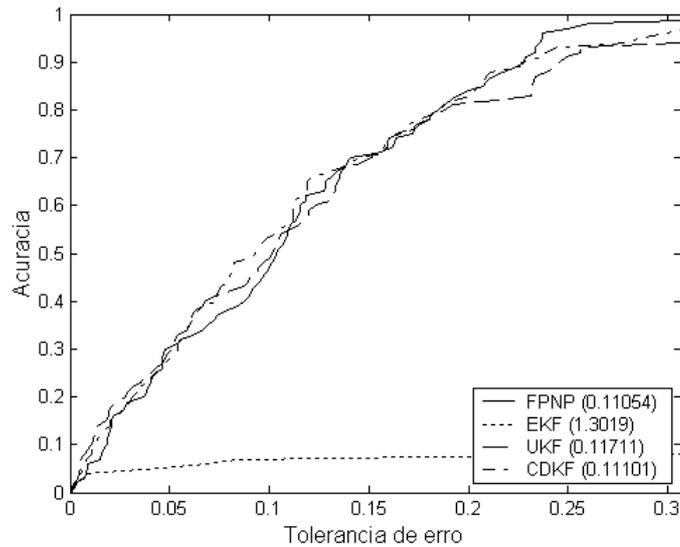


Figura 30. FPNP, EKF, UKF e CDKF aplicados à série temporal Darwin.

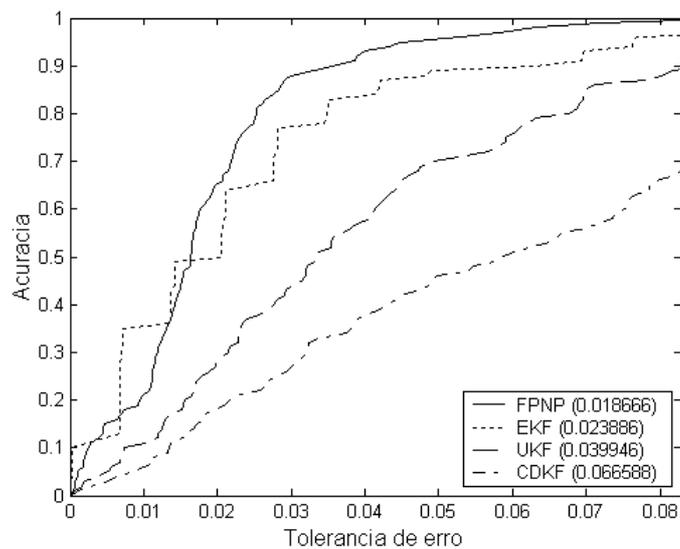


Figura 31. FPNP, EKF, UKF e CDKF aplicados à série temporal Soiltemp.

Prosseguindo a análise, o FPNP foi então comparado com outros métodos de filtragem por partículas, a saber: o filtro Sampling Importance Resampling (SIR), o Gaussian Sum Particle Filter (GSPF) e os Filtros de Partículas com Pontos Sigma, GMSPPF e SPPF (usando o SRUKF ou o SRCDKF para gerar a densidade de importância. As AOC's das REC curvas geradas para cada filtro de partículas são apresentadas na Tabela 4. Repare que comparado à outros filtros de partículas, a performance do FPNP não se mostra tão superior quanto ocorreu na comparação com filtros de Kalman tradicionais. No entanto, deve-se salientar que mesmo assim, o FPNP alcançou melhor performance em mais séries temporais que os demais (seis contra quatro, do segundo colocado).

Tabela 4. AOC's das curvas REC para o FPNP e para os demais filtros de partículas.

Série Temporal	FPNP	SIR	GSPF	GMSPPF	SPPF-SRUKF	SPPF-SRCDKF
A	0,086743	0,092569	0,091292	0,17247	0,27028	0,24854
Burstin	0,09336	1,2138	0,15682	0,15457	1,1425	1,2328
Darwin	0,11054	0,10916	0,11854	0,12537	0,47984	0,83357
Earthquake	0,049648	0,048115	0,055299	0,18906	0,574	0,15395
Leuven	0,16498	0,14315	0,16193	0,23336	0,40495	0,4513
Series 1	0,077789	0,085905	0,10719	0,12529	0,28606	0,62289
Series 2	0,14105	0,13493	0,15078	0,18032	0,4341	0,34265
Series 3	0,094089	0,11047	0,11061	0,21182	0,16549	0,12183
Soiltemp	0,018666	0,031723	0,21502	0,074003	0,21484	0,1916
Speech	0,052703	0,061352	0,0569	0,19801	0,33662	0,35516

Uma das séries em que a alta performance do FPNP pode ser verificada foi a série temporal Burstin, como pode ser visto nas Figuras 32 e 33.

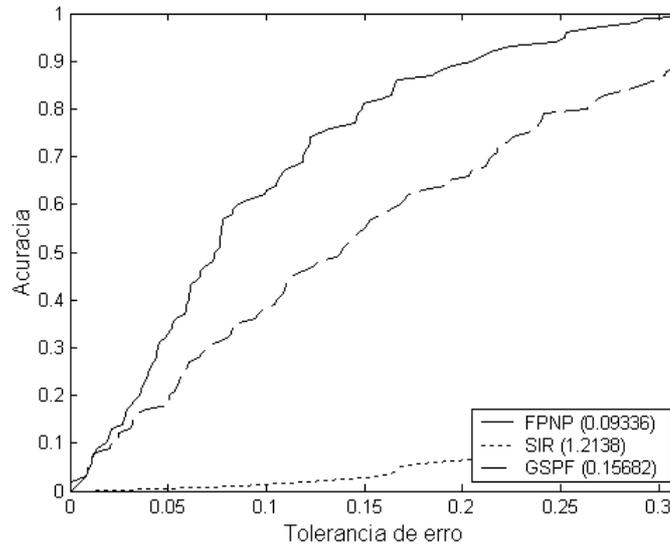


Figura 32. FPNP, SIR e GSPF aplicados à série temporal Burstin.

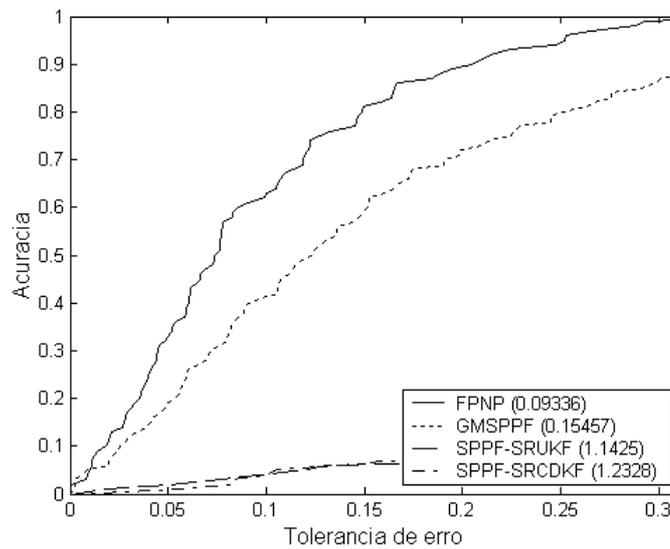


Figura 33. FPNP, GMSPPF e SPPF aplicados à série temporal Burstin.

Apesar de ser melhor que os filtros de Kalman tradicionais quando aplicado às séries temporais Darwin e Earthquake, comparado com outros filtros de partículas o FPNP fica em segundo lugar, perdendo para o filtro SIR por uma diferença quase imperceptível nos gráficos REC, como pode ser visto nas Figuras 34 e 35.

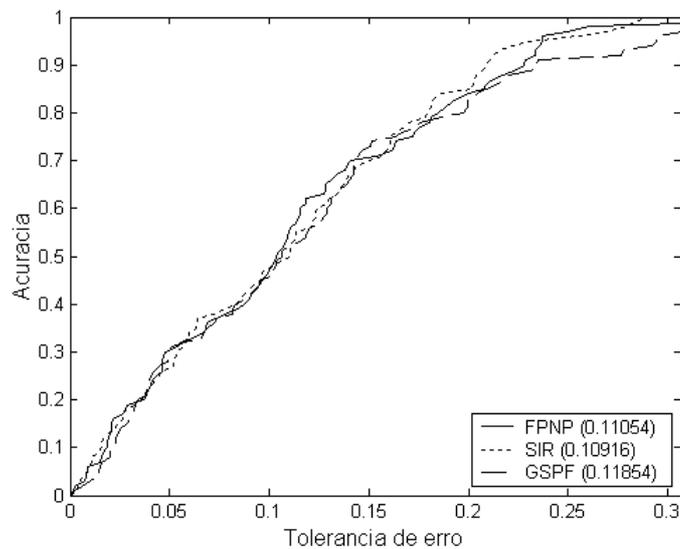


Figura 34. FPNP, SIR e GSPF aplicados à série temporal Darwin.

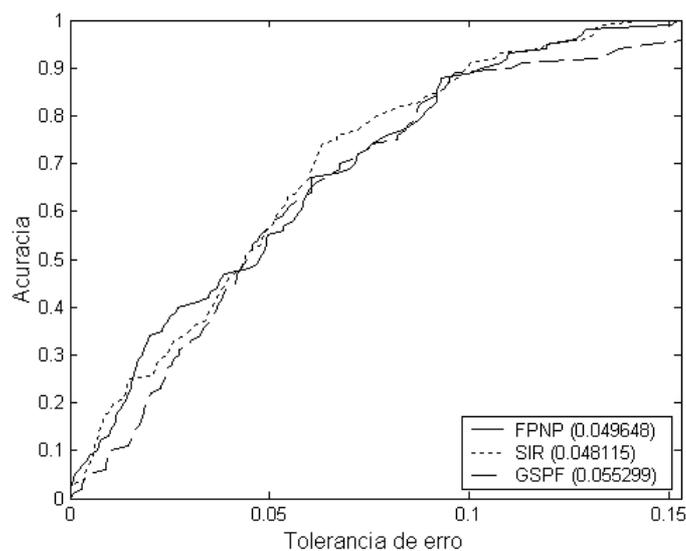


Figura 35. FPNP, SIR e GSPF aplicados à série temporal Earthquake.

É interessante observar que o segundo melhor algoritmo não foi o GSPF e nem um dos complexos filtros de partículas com pontos sigma, mas o filtro SIR, um dos filtros de partículas mais básicos. De fato, o filtro SIR pode ser visto como um “componente” do FPNP, devido principalmente à escolha da densidade de importância.

Pode-se notar que o SPPF, seja usando o SRUKF ou o SRCDKF para gerar a densidade de importância, obteve as piores performances para a maioria das séries testadas, sendo superior apenas ao EKF (isso pode ser visto, por exemplo, nas curvas REC geradas para a série temporal Series 1, nas Figuras 36 e 37).

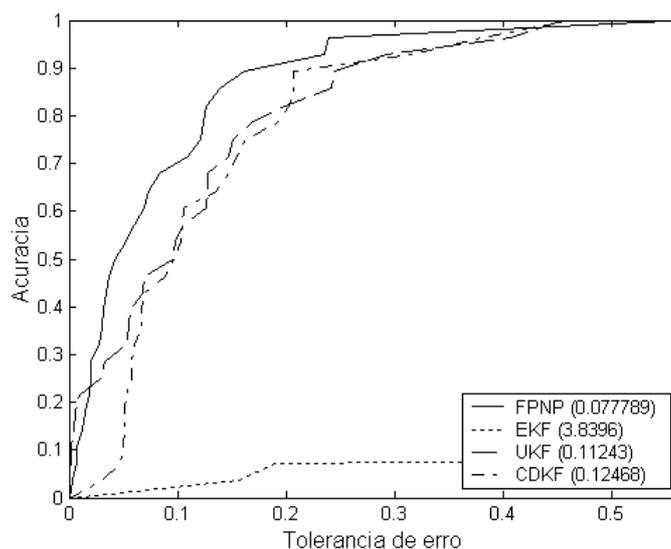


Figura 36. FPNP, EKF, UKF e CDKF aplicados à série temporal Series 1.

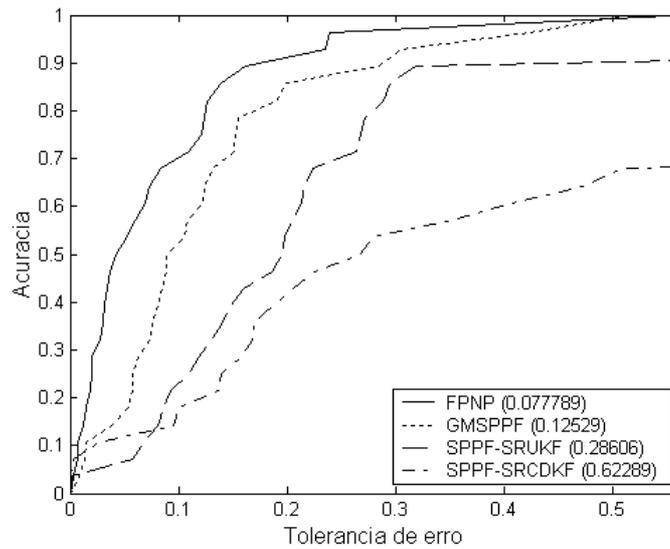


Figura 37. FPNP, GMSPPF e SPPF aplicados à série temporal Series 1.

Analisando-se todos os resultados dos filtros de Kalman e dos filtros de partículas simultaneamente, chegou-se ao ranking apresentado na Tabela 5, usado na determinação da significância com o teste de Friedman. O teste de Friedman acusou significância estatística na diferença entre resultados. Foi então realizado o teste de Holm para identificar as diferenças significativas. O resultado do teste de Holm mostrou que o FPNP alcançou um performance significativamente melhor que o EKF, CDKF, SRCDKF, GSPF, GMSPPF, SPPF-SRUKF e SPPF-SRCDKF, com nível de significância de 0,05. O teste não foi poderoso o suficiente para detectar diferença significativa entre o FPNP e os filtros UKF, SRUKF e SIR.

Tabela 5. Ranking dos algoritmos. Os empates são representados pela média dos ranks.

Série Temporal	FPNP	EKF	UKF	CDKF	SRUKF	SRCDKF	SIR	GSPF	GMSPPF	SPPF-SRUKF	SPPF-SRCDKF
A	1	10	6	4,5	7	4,5	3	2	8	11	9
Burstin	1	11	2,5	4,5	2,5	4,5	8	7	6	9	10
Darwin	2	11	5,5	3,5	5,5	3,5	1	7	8	9	10
Earthquake	2	10	5	8,5	4	8,5	1	3	7	11	6
Leuven	5	10	3	6,5	1	6,5	2	4	8	9	11
Series 1	1	11	4	6,5	5	6,5	2	3	8	9	10
Series 2	4	11	2	6,5	3	6,5	1	5	8	10	9
Series 3	1	7	6	8,5	5	8,5	2	3	11	10	4
Soiltemp	1	2	5	6,5	3	6,5	4	11	8	10	9
Speech	1	11	5	7,5	4	7,5	3	2	6	9	10
Rank médio	1,9	9,4	4,4	6,3	4,0	6,3	2,7	4,7	7,8	9,7	8,8

A Tabela 6 mostra o impacto, em cada algoritmo, da escolha de um modelo adequado (MIGON *et al.*, 2005) para representar a dinâmica do sistema para cada série temporal testada. Para a série A, as pulsações caóticas seguem mais ou menos o modelo teórico de Lorenz de um sistema de dois níveis (HUEBNER *et al.*, 1989). Para modelar a série temporal Burstin foi usado um modelo ARIMA (KEDEM e FOKIANOS, 2002). A função de autocorrelação sugere que um modelo ARIMA sazonal fornece uma boa descrição para a série Darwin (STORCH e ZWIERS, 1999). Para a série Earthquake, foi utilizado um modelo não linear para séries temporais de escala evolutiva, chamado modelo de volatilidade estocástica (KITAGAWA, 1987). O modelo usado para a série temporal Leuven é uma combinação não linear das três variáveis de estado do circuito de Chua generalizado (McNAMES *et al.*, 1999). Modelos para as séries temporais Series 1, 2 e 3 foram desenvolvidos usando o software BATS (POLE *et al.*, 1994). A

partir de um certo ponto as oscilações na série Soiltemp se tornam bem estáveis e periódicas, que podem ser modeladas com um modelo autoregressivo linear (SHUMWAY e STOFFER, 2006). A série temporal Speech apresenta um comportamento quasi-periódico, portanto também foi usado um modelo autoregressivo linear simples (SHUMWAY e STOFFER, 2006).

Tabela 6. AOC's das curvas REC para o FPNP e para os demais filtros.

Série Temporal	FPNP	EKF	SRUKF	SRCDKF	SIR	GSPF	GMSPPF	SPPF-SRUKF	SPPF-SRCDKF
A	0,0868	0,5861	0,0926	0,0926	0,0893	0,0845	0,0838	0,5465	0,4968
Burstin	0,0934	0,5187	0,0931	0,0931	0,0934	0,0934	0,0933	0,0970	0,0991
Darwin	0,1106	2,9703	0,1120	0,1120	0,1126	0,1105	0,1106	0,2162	0,1935
Earthquake	0,0497	1,5039	0,0482	0,0482	0,0486	0,0483	0,0484	0,0710	0,1057
Leuven	0,1650	0,5565	0,1675	0,1675	0,1676	0,1681	0,1680	0,1697	0,1720
Series 1	0,0778	0,2638	0,0671	0,0671	0,0672	0,0678	0,0679	0,0653	0,0724
Series 2	0,1411	0,3060	0,1231	0,1231	0,1234	0,1249	0,1250	0,1229	0,1977
Series 3	0,0941	0,2747	0,0711	0,0711	0,0714	0,0720	0,0721	0,0654	0,0752
Soiltemp	0,0187	0,7212	0,0152	0,0152	0,0155	0,0145	0,0146	0,0470	0,0448
Speech	0,0528	0,9161	0,0569	0,0569	0,0586	0,0544	0,0540	0,1680	0,1445

É possível constatar que mesmo não se beneficiando da escolha de um modelo feita por um especialista, o FPNP foi capaz de alcançar o melhor resultado em duas das dez séries testadas. O ranking médio mostrado na Tabela 7 sugere ainda que o FPNP obteve performance média maior que metade dos algoritmos testados, melhor inclusive que o SPPF-SRUKF, que obteve mais vitórias. O teste de Friedman detectou que existe diferença significativa nos resultados. Foi então realizado o teste de Nemenyi, que detectou diferença significativa entre a performance do EKF e de todos os outros

algoritmos, exceto os SPPF. Foi também detectada diferença significativa comparando SRUKF e SRCDKF com SPPF-SRUKF. Não foi detectado que o FPNP tenha obtido performance significativamente pior que qualquer um dos outros algoritmos testados.

Tabela 7. Ranking dos algoritmos. Os empates são representados pela média dos ranks.

Série Temporal	FPNP	EKF	SRUKF	SRCDKF	SIR	GSPF	GMSPPF	SPPF-SRUKF	SPPF-SRCDKF
A	3	9	5,5	5,5	4	2	1	8	7
Burstin	4	9	1,5	1,5	5,5	5,5	3	7	8
Darwin	2	9	4,5	4,5	6	1	3	8	7
Earthquake	6	9	1,5	1,5	5	3	4	7	8
Leuven	1	9	2,5	2,5	4	6	5	7	8
Series 1	8	9	2,5	2,5	4	5	6	1	7
Series 2	7	9	2,5	2,5	4	5	6	1	8
Series 3	8	9	2,5	2,5	4	5	6	1	7
Soiltemp	6	9	3,5	3,5	5	1	2	8	7
Speech	1	9	4,5	4,5	6	3	2	8	7
Rank médio	4,6	9,0	3,1	3,1	4,75	3,65	3,8	5,6	7,4

7. COMITÊS DE PREVISORES DE SÉRIES TEMPORAIS

Neste capítulo serão apresentadas as contribuições do trabalho na área de comitês de estimadores. Na próxima seção será apresentada a primeira aplicação de curvas REC na avaliação e seleção de componentes de um comitê de estimadores. A abordagem é utilizada no algoritmo de *boosting* para regressão. Em seguida, na Seção 7.2, são apresentados os resultados da comparação, através de curvas REC, de Filtros de Kalman com Pontos Sigma na previsão de séries temporais e comitês formados por tais filtros. Na Seção 7.3 a avaliação se estende, incluindo na comparação vários métodos de filtragem por partículas, funcionando individualmente e como estimadores base de comitês. Por fim, na Seção 7.4 é apresentada a avaliação de comitês de Filtros de Partículas Não-Paramétricos.

7.1. Curvas REC para Seleção de Modelos em Comitês

Boosting (SCHAPIRE, 1990) é um dos métodos mais populares para construção de comitês. Ele produz múltiplos modelos ao repetidamente alterar o conjunto de treinamento dado a um algoritmo de aprendizado. São dados pesos aos exemplos e a cada iteração uma nova hipótese é aprendida e os pesos dos exemplos são atualizados para que o sistema mantenha o foco em exemplos que a última hipótese não conseguiu prever corretamente. O estimador final é construído pelo voto ponderado dos modelos individuais. Cada estimador recebe um peso de acordo com sua acurácia. Boosting foi desenvolvido por pesquisadores de aprendizado computacional para

garantir a melhora na performance de algoritmos de aprendizado fracos que apenas precisam gerar uma hipótese com acurácia maior que 1/2 (FREUND e SCHAPIRE, 1996). Nos últimos anos, muitas pesquisas foram feitas sobre boosting (SCHAPIRE, 2002).

Em (AVNIMELECH e INTRATOR, 1999) é apresentado um algoritmo de boosting para regressão. O algoritmo, chamado de Regressor-Boosting, é baseado em uma observação fundamental de que freqüentemente o erro médio quadrático (EMQ) de um estimador é significativamente maior que a mediana quadrática do erro, devido a um pequeno número de erros grandes. Reduzindo o número de erros grandes, é possível reduzir o EMQ. O problema é definir o limiar a partir do qual um residual deve ser considerado um erro grande, uma vez que muitos fatores devem ser considerados. Em (PINA e ZAVERUCHA, 2005) é proposto o uso de Curvas Características de Erro de Regressão (BI e BENNETT, 2003), a fim de selecionar um bom valor limiar, de forma que apenas residuais maiores que tal valor sejam considerados erros.

7.1.1. Boosting para Regressão Usando Curvas REC

A essência do algoritmo Regressor-Boosting (AVNIMELECH e INTRATOR, 1999) é a construção de um comitê de três estimadores. Eles são treinados com diferentes distribuições de entrada e suas saídas são combinadas a fim de reduzir a taxa de erros grandes. Um erro é considerado grande se ele é maior que um valor limiar γ (ele é então chamado de erro grande com referência a γ).

O algoritmo funciona como segue. Os dados de treinamento são divididos em três conjuntos. O primeiro conjunto é usado para treinar um primeiro estimador. O segundo conjunto é usado para construir o conjunto de treinamento para o segundo estimador. Tal conjunto de treinamento contém as instâncias do segundo conjunto para as quais o primeiro estimador apresentou erro grande e uma quantidade similar de instâncias para as quais ele não apresentou erro grande. Os dois estimadores são testados com o terceiro conjunto, e o conjunto de treinamento do terceiro estimador contém somente as instâncias do terceiro conjunto para as quais exatamente um dos dois primeiros estimadores apresentou erro grande. A saída do comitê é a mediana das saídas dos três estimadores.

Em vez do voto majoritário do comitê usado para classificação, o modelo de combinação usado é a mediana dos três estimadores. Quando o comitê é usado para predição, a pior das três estimativas para qualquer instância é irrelevante, pois a mediana deve ser uma das outras estimativas. A mediana se mostrou teoricamente e empiricamente melhor que a média das saídas (AVNIMELECH e INTRATOR, 1999), e pode também ser aplicada a versões mais recentes do algoritmo de boosting.

Na prática existem muitos fatores que devem ser considerados na escolha do limiar γ usado pelo algoritmo Regressor-Boosting, tais como o tamanho do conjunto de dados. O valor ótimo para γ é aquele para o qual os erros grandes são responsáveis por uma parte significativa do EMQ, mas a taxa de erros grandes é baixa. Usualmente os conjuntos de dados com os quais o segundo e o terceiro estimadores são treinados são mais difíceis e tem uma taxa de erros grandes maior. Na maioria dos casos a escolha de um bom γ pode requerer tuning.

Em (PINA e ZAVERUCHA, 2006a) é proposto o uso de curvas REC a fim de encontrar um bom valor para γ . O valor para o parâmetro γ é selecionado como função da AOC da curva REC obtida para um estimador. Os erros grandes são aqueles residuais maiores que $f(\text{AOC})$, onde $f(\cdot)$ deve ser definido. Foram conseguidos bons resultados apenas multiplicando o valor da AOC por um escalar, ajustado por meio de conjuntos de validação em uma validação cruzada interna (MITCHELL, 1997).

Entretanto, a maior melhoria proporcionada por essa abordagem é que pode-se ter acesso a todos os benefícios das curvas REC descritos no Capítulo 5, tornando possível uma melhor avaliação de cada parte do comitê e facilitando o tuning. O algoritmo de boosting para regressão usando curvas REC é descrito em Algoritmo 3.

Algoritmo 3. Boosting para regressão usando curvas REC.

1. Divida o conjunto de dados em conjunto de treinamento e conjunto de teste.
2. Divida o conjunto de treinamento em três conjuntos: Set1, Set2 e Set3.
3. Treine Expert1 com TrainingSet1 = Set1.
4. Teste Expert1 com Set2. Construa a curva REC, calcule a AOC e determine o limiar γ . Adicione ao TrainingSet2 todas as instâncias nas quais Expert1 teve erro grande. Adicione ao TrainingSet2 uma quantidade similar de instâncias nas quais Expert1 não teve erro grande.
5. Treine Expert2 com TrainingSet2.
6. Teste Expert1 e Expert2 com Set3. Construa as curvas REC, calcule as AOCs e determine os limiares γ_1 e γ_2 . Adicione ao TrainingSet3 todas as instâncias nas quais exatamente um dos experts (Expert1 ou Expert2) teve erro grande (com referência a γ_1 para Expert1 e com referência a γ_2 para Expert2).
7. Treine Expert3 com TrainingSet3.
8. A saída do comitê para cada instância do conjunto de teste é a mediana das saídas dos três experts.

7.1.2. Avaliação Experimental

Uma avaliação experimental extensiva foi realizada com o objetivo de analisar o uso das Curvas Características de Erro de Regressão para a seleção de modelos em comitês de Redes Neurais. Esta seção descreve os aspectos dos experimentos e apresenta seus resultados.

Nos experimentos, 20 conjuntos de dados foram usados a fim de incluir muitos domínios e dificuldades. Os conjuntos de dados foram obtidos em: UCI Machine Learning Repository (BLAKE e MERZ, 2005), empresas brasileiras (TEIXEIRA e ZAVERUCHA, 2003), Delve repository [<http://www.cs.toronto.edu/~delve/>], MLnet Archive [<http://www.mlnet.org/>], StatLib data [<http://lib.stat.cmu.edu/>] e na home page de Luís Torgo [<http://www.niaad.liacc.up.pt/~ltorgo/>]. Na Tabela 8 são apresentados os conjuntos de dados utilizados e um resumo de suas principais características: o nome dos arquivos, o número de instâncias e o número total de atributos. O método de teste usado nesta pesquisa foi a validação cruzada com 10 partições (DIETTERICH, 1997).

Os parâmetros das Redes Neurais usadas nos testes (Multilayer Perceptrons) foram ajustados por intermédio de uma validação cruzada interna com conjuntos de validação. Foram feitos testes também usando Decision Stumps para regressão (baseada no erro médio quadrático). Embora seja mais rápidas, seus resultados foram piores que os das Redes Neurais em todos os testes realizados e por isso foram deixados de lado. As implementações das Redes Neurais e das Decision Stumps foram obtidas no sistema WEKA (WITTEN e FRANK, 2005). Todo o código fonte foi escrito em Java.

Tabela 8. Conjuntos de dados usados na avaliação experimental.

Conjunto de Dados	Arquivo	Instâncias	Atributos
Abalone	abalone	4177	9
Airplane Companies	stock	950	10
Auto Imports	autoPrice	159	16
Auto-Mpg	auto	398	8
Bank	bank8FM	4499	9
Basketball Points	baskball	96	5
Brazilian utilities	electricload	83	13
Breast Cancer	r_wpbc	194	33
California Housing	cal_housing	20640	9
Census	house_8L	22784	9
Computer Activity	cpu_small	8192	13
Elevators	elevators	8752	19
Housing	housing	506	14
Kinematics	kin8nm	8192	9
Machine-Cpu	machine	209	7
Pole Telecomm	pol	5000	49
Pollution	pollution	60	16
Pumadyn	puma8NH	4499	9
pwLinear	pwLinear	200	11
Triazines	triazines	186	61

Foram testados ambos os métodos de combinação de saídas das três redes: a mediana e a média. Alguns testes confirmaram os resultados alcançados por Avnimelech e Intrator (1999), nos quais a mediana mostrou ser a melhor abordagem. A Figura 38 mostra um desses casos. Entretanto, na maioria dos testes a diferença foi insignificante.

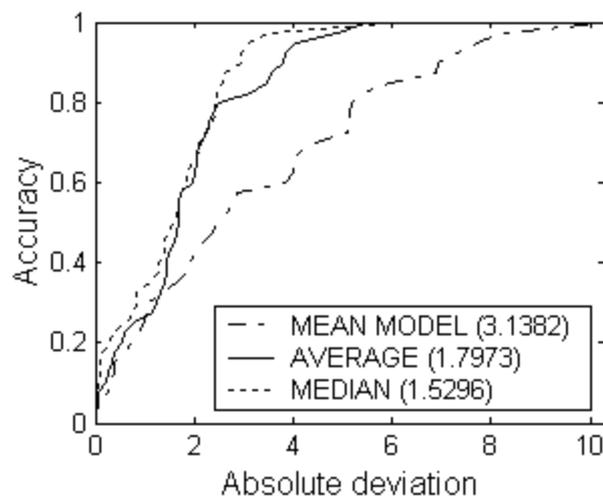


Figura 38. Comparação entre mediana e média como métodos para combinar as saídas das três redes para o conjunto de dados pwLinear.

Foi verificado que a performance do boosting é melhor ou tão boa quanto à da melhor rede. A Figura 39 ilustra essa conclusão.

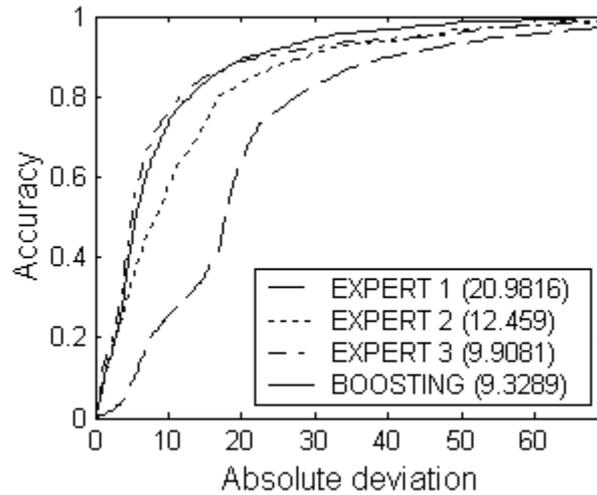


Figura 39. Comparação entre as performances individuais das redes e o boosting para o conjunto de dados Pole Telecomm.

As AOCs das curvas REC geradas por uma Rede Neural e pelo boosting para cada conjunto de dados são mostradas na Tabela 9. Na última coluna da Tabela 9 estão marcados os conjuntos de dados para os quais o boosting gerou aumento de performance. Como esperado, o boosting consegue melhores resultados quando é aplicado a grandes conjuntos de dados.

Com a maioria dos conjuntos de dados a melhora nos resultados com boosting é claramente visível, como pode ser visto, por exemplo, na Figura 40.

Tabela 9. AOCs das curvas REC geradas para cada conjunto de dados.

Conjunto de Dados	AOC Rede Neural	AOC Boosting	
Abalone	2,427	2,328	*
Airplane Companies	4,905	6,761	
Auto Imports	4500,893	2499,902	*
Auto-Mpg	3,956	2,907	*
Bank	0,027	0,023	*
Basketball Points	0,068	0,062	*
Brazilian utilities	0,066	0,022	*
Breast Cancer	45,227	41,710	*
California Housing	111899,660	38471,861	*
Census	24173,773	19091,456	*
Computer Activity	3,493	2,258	*
Elevators	0,0018	0,0016	*
Housing	2,105	2,142	
Kinematics	0,113	0,104	*
Machine-Cpu	39,221	46,020	
Pole Telecomm	10,465	9,328	*
Pollution	39,459	28,055	*
Pumadyn	3,234	2,683	*
pwLinear	2,085	1,529	*
Triazines	0,147	0,109	*

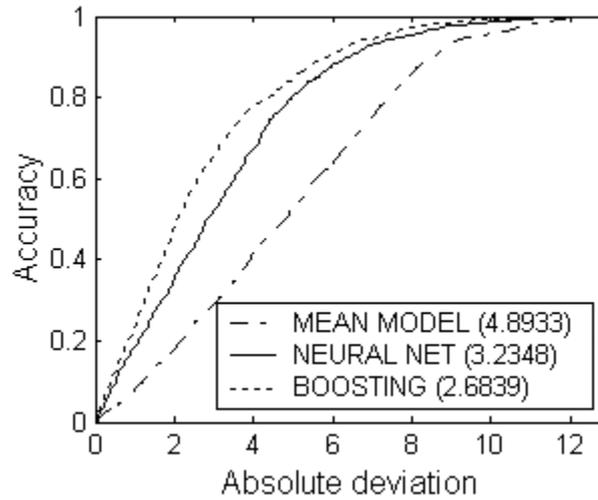


Figura 40. Comparação entre Rede Neural e Boosting para o conjunto de dados Pumadyn.

No entanto, com alguns conjuntos de dados a diferença é mais sutil, porém ainda assim significativa. Um exemplo desse caso é mostrado na Figura 41. Deve-se notar que, embora as curvas REC estejam próximas umas das outras, a curva correspondente ao boosting cobre quase completamente as demais, ou seja, a função de regressão gerada pelo boosting domina as demais e é portanto preferível.

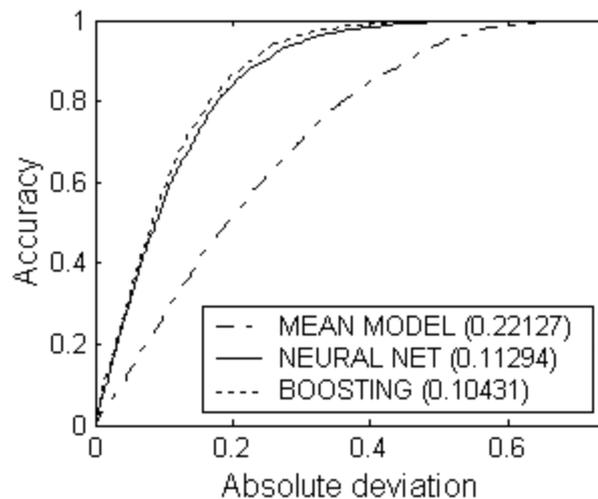


Figura 41. Comparação entre Rede Neural e Boosting para o conjunto de dados Kinematics.

7.2. Comitês de Filtros de Kalman com Pontos Sigma

São apresentados os resultados da comparação, usando curvas REC, da família de filtros SPKF e EKF aplicados a séries temporais e investigamos os uso de um método de combinação de componentes em comitês (empilhamento (WOLPERT,1992)) com os modelos testados, avaliando-os através de curvas REC, como sugerido por Bi e Bennett (2003). As mesmas séries temporais descritas no Capítulo 6 foram utilizadas nos experimentos e os resultados foram publicados em (PINA e ZAVERUCHA, 2006c).

7.2.1. Comparando Filtros de Kalman através de Curvas REC

Primeiro, foram comparados o UKF e o CDKF com suas formas raiz-quadrada, SRUKF e SRCDKF (versões otimizadas do UKF e do CDKF) respectivamente. Como esperado, as curvas REC para o UKF e para o SRUKF são muito similares. Isso significa que a diferença entre as performances dos modelos gerados pelo UKF e pelo SRUKF foi insignificante. O mesmo fato pôde ser verificado com as curvas REC para o CDKF e o SRCDKF. Portanto, considerando esses resultados e as outras vantagens mencionadas antes no Capítulo 2, os experimentos continuaram somente com as formas raiz-quadrada dos SPKF.

Analisando os gráficos REC gerados, pôde-se verificar que, para a maioria das séries temporais, o modelo fornecido pelo SRUKF domina os modelos fornecidos pelo SRCDKF e pelo EKF, isto é, a curva REC para o modelo do SRUKF está sempre acima

das curvas REC para o SRCDKF e o EKF. Portanto, o modelo gerado pelo SRUKF seria preferível. Um exemplo é mostrado na Figura 42.

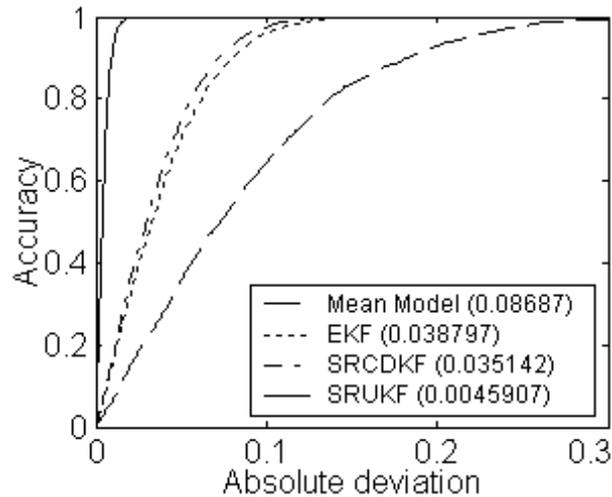


Figura 42. EKF e SPKF's aplicados à série temporal Burstin.

SRCDKF e EKF alcançaram performances similares para quase todas as séries temporais, como pode ser visto, por exemplo, na Figura 43. Entretanto, a análise das AOC's dá uma pequena vantagem para o SRCDKF. A baixa performance do EKF quando comparado aos outros é provavelmente causada pela não-linearidade das séries. Portanto, o SRUKF consistentemente se mostrou a melhor alternativa para ser usada com essas séries, seguido pelo SRCDKF e pelo EKF, nesta ordem. Uma Árvore Modelo (QUINLAN, 1992) e o algoritmo NBR (FRANK *et al.*, 2000) também foram testados para a previsão de séries temporais, mas ambos geraram modelos pobres.

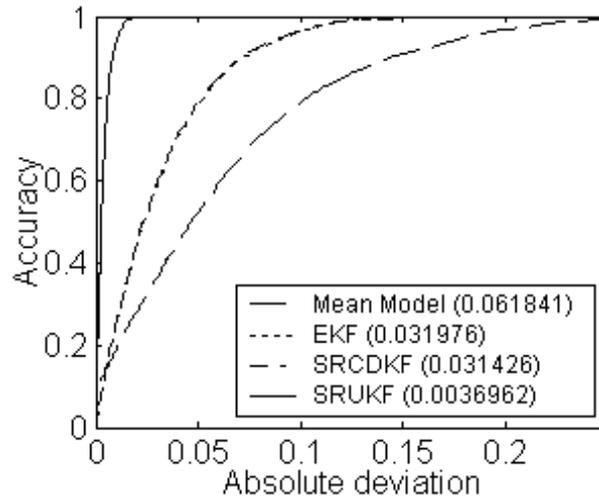


Figura 43. EKF e SPKF's aplicados à série temporal Earthquake.

7.2.2. Comparando Comitês de Filtros de Kalman

Foi utilizado empilhamento (descrito na Seção 5.3) para combinar componentes em comitês de SPKFs e EKF. Uma Árvore Modelo (QUINLAN, 1992) foi escolhida como meta-regressor, não somente porque ela alcançou bons resultados em experimentos iniciais, mas também porque constitui o estado-da-arte em métodos de regressão e já foi utilizada com sucesso como meta-classificador para empilhamento (DZEROSKI e ZENKO, 2004), superando todos os outros métodos de combinação testados. A implementação de árvore modelo foi obtida a partir do sistema WEKA (WITTEN e FRANK, 2005). Todo o código fonte foi escrito em Java.

A fim de determinar qual subconjunto de algoritmos pode gerar o melhor comitê, foram construídos quatro modelos com empilhamento: um contendo as formas raiz-quadrada dos SPKFs e o EKF, e os outros deixando um deles de fora. Se muitos algoritmos estivessem sendo testados, um método heurístico poderia ser usado para selecionar os componentes do comitê (CARUANA e NICULESCU-MIZIL, 2004). A

Tabela 10 mostra os empilhamentos construídos: o Empilhamento 1 é composto pelo EKF e pelo SRCDKF; o Empilhamento 2 é composto pelo EKF e pelo SRUKF; o Empilhamento 3 é composto pelo SRCDKF e pelo SRUKF; e o Empilhamento 4 é composto pelo EKF, pelo SRCDKF e pelo SRUKF. Todos os empilhamentos utilizaram uma Árvore Modelo como meta-regressor.

Tabela 10. Empilhamentos construídos.

Empilhamento	Estimadores Base
Empilhamento 1	EKF, SRCDKF
Empilhamento 2	EKF, SRUKF
Empilhamento 3	SRCDKF, SRUKF
Empilhamento 4	EKF, SRCDKF, SRUKF

As curvas REC mostram que todos os empilhamentos que continham o SRUKF como estimador base alcançaram performances altas similares. Isso pode ser visto, por exemplo, na Figura 44.

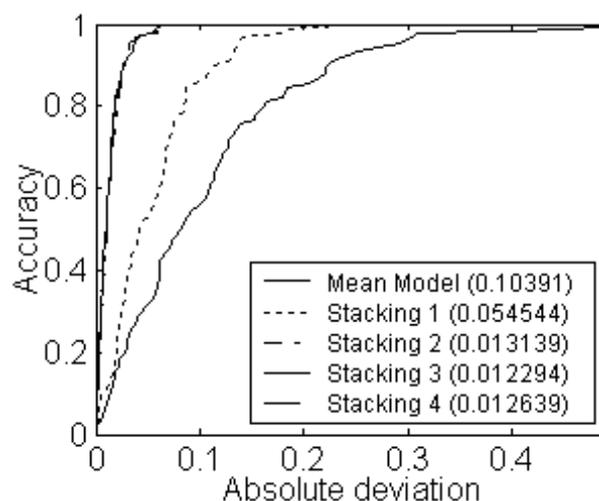


Figura 44. Empilhamentos aplicados à série temporal Series 2.

A Tabela 11 mostra os valores de AOC das curvas REC geradas pelos empilhamentos com o SRUKF como estimador base. Analisando os valores pode-se ver que dentre os três empilhamentos que contém o SRUKF, aqueles que têm o SRCDKF como estimador base alcançam performances ligeiramente superiores. Uma vez que o número de séries temporais em que o Empilhamento 3 alcançou a melhor performance é quase o mesmo número de séries temporais em que o Empilhamento 4 foi o melhor, foi considerado que a inclusão do EKF como estimador base não compensa o *overhead* em termos de custo computacional. Assim, o modelo escolhido como o melhor é aquele gerado pelo Empilhamento 3 (SRCDKF e SRUKF como estimadores base).

Tabela 11. AOC's das curvas REC geradas para os empilhamentos contendo SRUKF como estimador base.

Série Temporal	Empilhamento 2	Empilhamento 3	Empilhamento 4
A	0,001366	0,001497	0,001310
Burstin	0,001740	0,001613	0,001740
Darwin	0,013934	0,014069	0,014052
Earthquake	0,000946	0,000943	0,000946
Leuven	0,005172	0,005190	0,005142
Series 1	0,001167	0,001306	0,001111
Series 2	0,013139	0,012294	0,012639
Series 3	0,000800	0,000717	0,000767
Soiltemp	0,000884	0,000780	0,000782
Speech	0,000714	0,000713	0,000706

Comparando o melhor modelo de empilhamento (SRCDKF e SRUKF como estimadores base e uma Árvore Modelo como meta-regressor) com o melhor algoritmo individual (SRUKF), pôde-se verificar que o empilhamento alcançou uma performance significativamente mais alta para todas as séries testadas. Isso pode ser claramente notado na Figura 45.

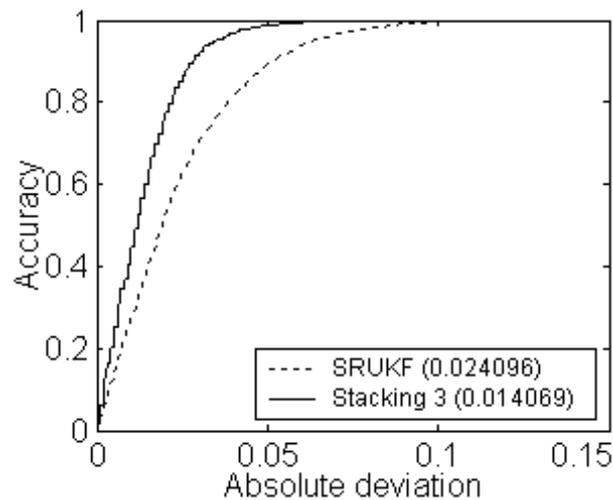


Figura 45. SRUKF e Empilhamento 3 aplicados à série temporal Darwin.

7.3. Comitês de Filtros de Partículas

São apresentados os resultados da comparação, através de curvas REC, de vários métodos de filtragem por partículas aplicados a problemas difíceis de previsão de séries temporais não-lineares (descritos no Capítulo 6). Então, o melhor modelo de filtro de partículas foi comparado com o melhor modelo de filtro de Kalman. Depois, foi analisado, através de curvas REC, o uso de comitês de filtros de partículas. Os

experimentos foram concluídos com a análise do uso de todos os algoritmos testados na construção de um melhor modelo de comitê (PINA e ZAVERUCHA, 2007c).

Os algoritmos usados nos experimentos foram os mesmos usados nos experimentos descritos no Capítulo 6: EKF, UKF, CDKF, SRUKF, SRCDKF, SIR, GSPF, GMSPPF, SPPF-SRUKF (SPPF usando o SRUKF para gerar a densidade de importância), e SPPF-SRCDKF (SPPF usando SRCDKF para gerar a densidade de importância). As séries temporais utilizadas também foram as mesmas descritas no Capítulo 6.

7.3.1. Comparando Filtros de Partículas através de Curvas REC

Primeiro, foram comparadas as performances individuais de todos os métodos de filtragem por partículas. Analisando os gráficos REC gerados, pôde-se verificar que os modelos gerados pelo GSPF e pelo GMSPPF alcançaram performances similarmente boas para todas as séries testadas. O gráfico REC na Figura 46 mostra um caso onde o GSPF gerou um modelo ligeiramente melhor que o GMSPPF. Para essa série temporal, o SPPF-SRUKF alcançou uma boa performance também. Está claro que o filtro SIR teve uma performance mais baixa, especialmente para tolerâncias de erros mais altas. Deve-se notar que o modelo gerado pelo SPPF-SRCDKF é muito pobre e é dominado por todos os outros, isto é, as curvas REC dos demais filtros estão sempre acima da curva REC do SPPF-SRCDKF. Portanto, os modelos gerados pelo SPPF-SRCDKF não seriam recomendáveis.

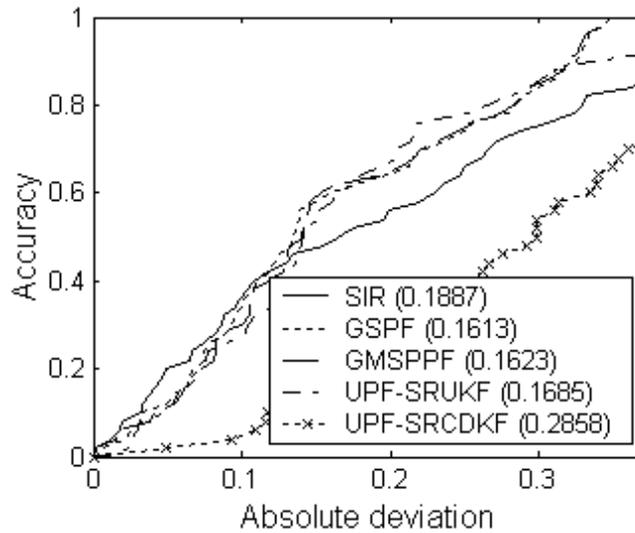


Figura 46. Filtros de partículas aplicados à série temporal Leuven.

Na Figura 47, o comportamento das curvas REC geradas pelo GSPF e pelo GMSPPF pode ser visto em mais detalhes. As curvas se cruzam, em outras palavras, cada modelo é melhor para diferentes faixas de tolerância de erro. A fim de selecionar um dos modelos como o melhor, uma análise das AOC's foi conduzida.

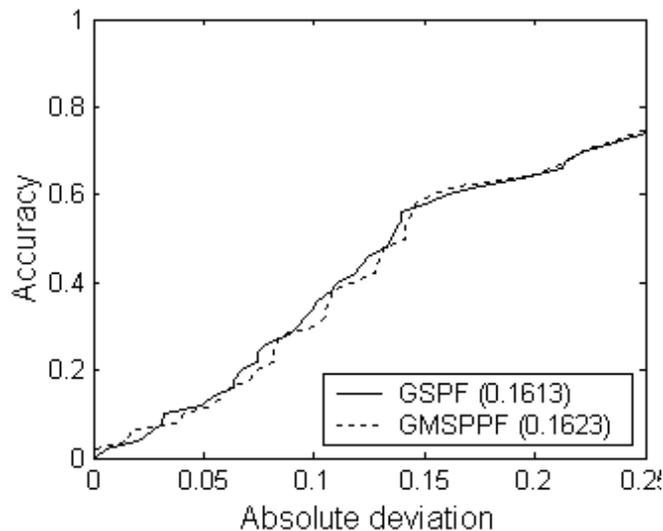


Figura 47. Melhores filtros de partículas testados aplicados à série temporal Leuven.

Os valores das AOC's das curvas REC geradas por cada filtro de partículas são mostrados na Tabela 12. Considerando todas as séries temporais, a análise das AOC's dá uma pequena vantagem ao GSPF.

Tabela 12. AOC's das curvas REC geradas por cada filtro de partículas.

Série Temporal	GSPF	GMSPPF	SIR	SPPF-SRUKF	SPPF-SRCDKF
A	0,1039	0,1048	0,1051	0,1166	0,1119
Burstin	0,0933	0,0933	0,0931	0,0992	0,0934
Darwin	0,1100	0,1101	0,1102	0,1559	0,1471
Earthquake	0,0486	0,0486	0,0487	0,0541	0,0540
Leuven	0,1613	0,1623	0,1887	0,1685	0,2858
Series 1	0,0669	0,0705	0,0701	0,0705	0,0755
Series 2	0,1246	0,1248	0,1236	0,1331	0,1511
Series 3	0,0716	0,0716	0,0720	0,0700	0,0721
Soiltemp	0,0153	0,0153	0,0157	0,0318	0,0265
Speech	0,0571	0,0573	0,0590	0,0723	0,0728

A performance do GSPF, considerado o melhor filtro de partículas testado, foi comparada com a do SRUKF (versão otimizada do UKF), considerado um dos melhores filtros de Kalman. Pode-se notar por exemplo no gráfico REC na Figura 48 que o filtro de partículas gerou um modelo consistentemente melhor.

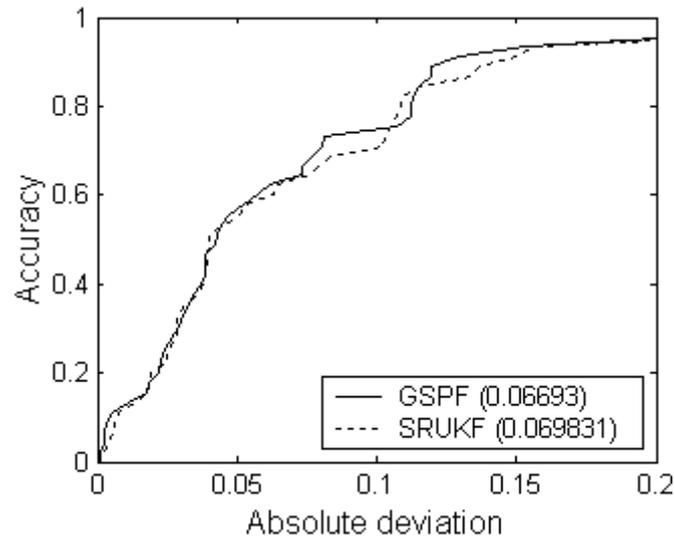


Figura 48. GSPF e SRUKF aplicados à série temporal Series 1.

7.3.2. Comparando Comitês de Filtros de Partículas

Foi utilizado empilhamento para construir comitês de filtros de partículas. Uma *Árvore Modelo* foi escolhida como meta-regressor, pelos motivos já citados anteriormente. Mais recentemente, foi mostrado que *Árvores Modelo* alcançam alta performance como meta-regressores em empilhamentos de filtros de Kalman para a previsão de séries temporais (PINA e ZAVERUCHA, 2006c).

Para determinar que subconjunto dos algoritmos pode gerar o melhor comitê, foram construídos modelos de empilhamento através de uma abordagem *wrapper* (KOHAVI e JOHN, 1997): conjuntos de validação são usados para estimar a AOC da curva REC do comitê para um conjunto de componentes. O melhor subconjunto de componentes é mantido. Uma vez que o conjunto de algoritmos usados não foi grande, pôde-se realizar uma busca exaustiva em todo o espaço de subconjuntos de algoritmos possíveis a fim de se encontrar o melhor modelo de comitê.

Os resultados mostraram que o melhor comitê de filtros de partículas encontrado é composto por SIR, GSPF, e SPPF-SRUKF. Comparando o melhor modelo de empilhamento com o melhor algoritmo individual de filtragem por partículas (GSPF) verificou-se que o comitê alcançou uma performance mais alta, especialmente em uma determinada faixa de tolerância de erro. Isso pode ser notado no gráfico REC apresentado na Figura 49.

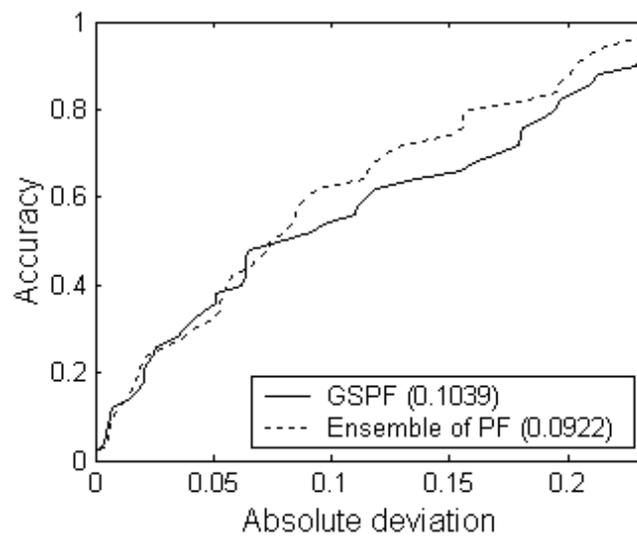


Figura 49. GSPF e o melhor comitê de filtros de partículas aplicados à série temporal A.

7.3.3. Comparando Comitês de Previsores de Séries Temporais

A fim de encontrar um melhor modelo de previsão de séries temporais, foram incluídos no conjunto de possíveis componentes para o comitê o EKF e a família de filtros SPKF. Novamente, as previsões individuais foram combinadas com empilhamento usando uma Árvore Modelo como meta-regressor.

A busca pelo melhor comitê foi também realizada através da abordagem *wrapper*. Embora o número de algoritmos seja maior que nos experimentos anteriores, ainda foi possível realizar uma busca exaustiva em tempo razoável. O melhor comitê encontrado é composto por: CDKF, GMSPPF e SPPF-SRUKF.

O melhor comitê de previsores de séries temporais foi comparado com o melhor comitê de filtros de partículas. Pode-se notar no gráfico REC da Figura 50 que embora ambos os comitês encontrados tenham componentes diferentes, os resultados finais são bem similares. Deve-se notar também que apesar de o GSPF, que foi considerado o melhor modelo individual, não estar presente neste último comitê, o GMSPPF está, e como foi verificado antes, suas performances são bem parecidas. O SPPF-SRUKF não alcançou uma das melhores performances individuais, entretanto ele foi incluído em ambos os comitês. Isso sugere que o SPPF-SRUKF tem uma boa performance em um conjunto de pontos menor, mas diferente dos demais previsores.

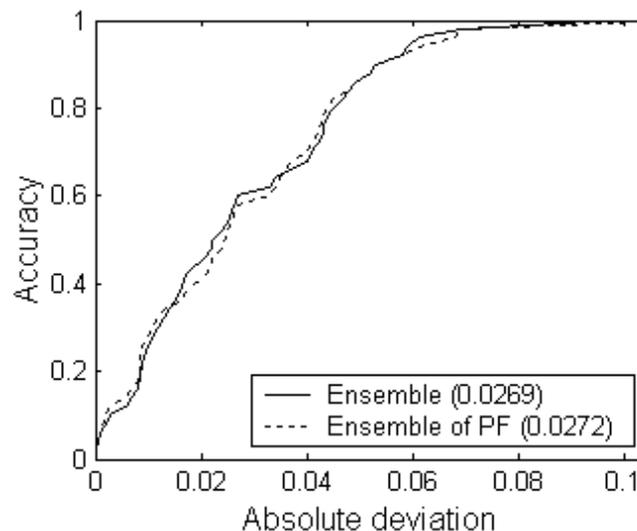


Figura 50. Melhores comitês aplicados à série temporal Earthquake.

7.4. Comitês de Filtros de Partículas Não-Paramétricos

Para gerar comitês de Filtros de Partículas Não-Paramétricos, o algoritmo foi executado 50 vezes e em seguida foi feita uma busca gulosa com reposição, incluindo a cada passo o estimador que resultasse na melhor performance combinada, medida num conjunto de validação através do uso de curvas características de erro de regressão, parando assim que a inclusão de nenhum estimador resultasse em melhor performance combinada. O método de combinação de resultados escolhido foi a média dos resultados individuais dos componentes. Também foram construídos comitês de tamanho fixo, a fim de comparar os resultados.

Primeiro foram analisados os comitês de tamanho fixo igual a 10, 20, 30, 40 e 50 componentes, escolhidos aleatoriamente. A análise das curvas REC apontou o maior comitê como o melhor, obtendo a performance mais alta em 5 das 10 séries temporais testadas. O comitê formado por 10 componentes foi o segundo colocado, sendo o melhor para 3 séries temporais. Os comitês com 20 e 40 componentes venceram em apenas uma série, e o comitê com 30 componentes não obteve nenhuma vitória.

O melhor comitê de tamanho fixo para cada série foi então comparado com o comitê construído através de uma busca gulosa. O comitê cuja seleção de componentes foi feita usando uma busca heurística, obteve a melhor performance para 8 das 10 séries temporais testadas. Comparando o melhor comitê formado com um FPNP funcionando individualmente, o comitê alcançou resultados melhores para todas as series temporais.

Ao realizar a comparação do comitê de FPNP com os filtros de Kalman e filtros de partículas, pôde-se verificar que o comitê consegue superar os filtros de Kalman em 9 das 10 séries testadas, e consegue superar os filtros de partículas em 8 das

10 séries testadas. Considerando todos os algoritmos, o comitê de FPNP alcançou a melhor performance em 8 das 10 séries temporais estudadas (lembrando que, individualmente, o FPNP obteve a melhor performance em apenas 6 das 10 séries que fizeram parte desta pesquisa). Os valores das AOC's das curvas REC geradas para cada filtro são mostrados nas Tabelas 13 e 14.

Tabela 13. AOC's do melhor comitê de FPNP e filtros de Kalman.

Série Temporal	FPNP	EKF	UKF	CDKF	SRUKF	SRCDKF
A	0,083709	0,26615	0,14579	0,13216	0,14954	0,13216
Burstin	0,092154	1,6815	0,14242	0,1437	0,14242	0,1437
Darwin	0,10748	1,3019	0,11711	0,11101	0,11711	0,11101
Earthquake	0,048569	0,31558	0,088879	0,20975	0,08003	0,20975
Leuven	0,16064	0,44577	0,1548	0,18579	0,1407	0,18579
Series 1	0,074493	3,8396	0,11234	0,12468	0,12102	0,12468
Series 2	0,13425	1,6903	0,13977	0,17075	0,13991	0,17075
Series 3	0,089988	0,13904	0,1276	0,16197	0,12759	0,16197
Soiltemp	0,014593	0,023886	0,039946	0,066588	0,031505	0,066588
Speech	0,051614	0,61147	0,075437	0,20336	0,074109	0,20336

Tabela 14. AOC's do melhor comitê de FPNP e demais filtros de partículas.

Série Temporal	FPNP	SIR	GSPF	GMSPPF	SPPF-SRUKF	SPPF-SRCDKF
A	0,083709	0,092569	0,091292	0,17247	0,27028	0,24854
Burstin	0,092154	1,2138	0,15682	0,15457	1,1425	1,2328
Darwin	0,10748	0,10916	0,11854	0,12537	0,47984	0,83357
Earthquake	0,048569	0,048115	0,055299	0,18906	0,574	0,15395
Leuven	0,16064	0,14315	0,16193	0,23336	0,40495	0,4513
Series 1	0,074493	0,085905	0,10719	0,12529	0,28606	0,62289
Series 2	0,13425	0,13493	0,15078	0,18032	0,4341	0,34265
Series 3	0,089988	0,11047	0,11061	0,21182	0,16549	0,12183
Soiltemp	0,014593	0,031723	0,21502	0,074003	0,21484	0,1916
Speech	0,051614	0,061352	0,0569	0,19801	0,33662	0,35516

8. MELHORANDO A PERFORMANCE ATRAVÉS DA COMBINAÇÃO DE ATRIBUTOS

O algoritmo Naive Bayes para Regressão (NBR) (FRANK *et al.*, 2000) usa a metodologia do Naive Bayes para tarefas de predição numéricas, modelando a distribuição de probabilidade do valor objetivo com estimadores de densidades por kernels. Entretanto, a incrível performance do Naive Bayes para problemas de classificação não é observada para problemas de regressão. Baseados em resultados experimentais que isolam a suposição de independência como a principal razão da pobre performance do NBR, Frank *et al.* (2000) concluíram que o algoritmo só deve ser aplicado a problemas de regressão quando a suposição de independência é verdadeira. Infelizmente, alguns dos métodos mencionados anteriormente não podem ser aplicados ao NBR e outros necessitariam do uso de discretização. Em (PINA e ZAVERUCHA, 2007a) é apresentada a análise e avaliação experimental de uma nova abordagem para melhorar os resultados do algoritmo NBR. Isso é realizado através da combinação de atributos por meio de algoritmos de regressão auxiliares.

8.1. Combinando Atributos

Uma vez que a principal razão da baixa performance do NBR é a suposição de independência, tentou-se superar essa situação combinando atributos relacionados. A abordagem proposta é similar a descrita em (PAZZANI, 1996), na qual atributos nominais são combinados se isso gera aumento da acurácia preditiva. A principal

diferença é que em (PAZZANI, 1996) os atributos originais são substituídos por um novo, cujos valores possíveis são as combinações de todos os valores possíveis para cada um deles (atributos numéricos são discretizados), enquanto que na abordagem proposta os atributos (nominais ou numéricos) são combinados por meio de um algoritmo de regressão auxiliar. A princípio, esse algoritmo auxiliar, chamado *combinador*, pode ser qualquer método de regressão. O valor da combinação é a predição do atributo objetivo fornecida por um combinador baseado apenas nos atributos sendo combinados, isto é, o combinador usa somente aqueles atributos para fazer uma predição. A Figura 51 mostra o algoritmo NBR e um modelo construído para combinar dois atributos: o combinador faz uma predição C' do atributo objetivo C baseado apenas nos atributos A_3 e A_4 .

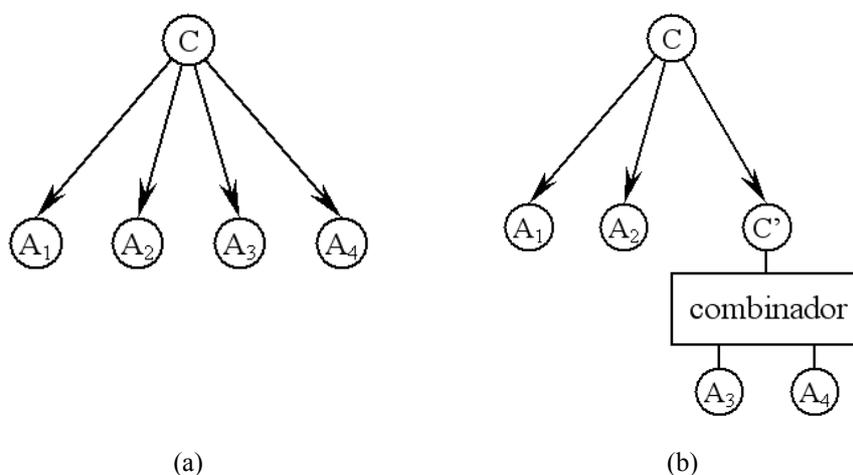


Figura 51. Naive Bayes para Regressão (a) e o modelo conseguido pela combinação dos atributos A_3 e A_4 (b).

O procedimento para treinar o modelo funciona como segue. Primeiro, o algoritmo NBR é treinado tratando todos os atributos como condicionalmente independentes. Então ele considera a substituição de cada par de atributos por um novo

atributo combinado. Para isso, usando o conjunto de treinamento, é gerado um conjunto de instâncias, chamadas *instâncias parciais*, formado apenas pelo par de atributos e pelo atributo objetivo. Essas instâncias parciais são usadas para treinar o combinador, como é mostrado na Figura 52: as geradas pelo combinador por meio de validação cruzada leave-one-out serão os valores para o novo atributo combinado, substituindo os valores dos atributos sendo combinados no conjunto de treinamento original, assim gerando o conjunto de treinamento para o NBR. Usando esse conjunto gerado, o NBR é treinado, considerando a estrutura modificada, como vista na Figura 53.

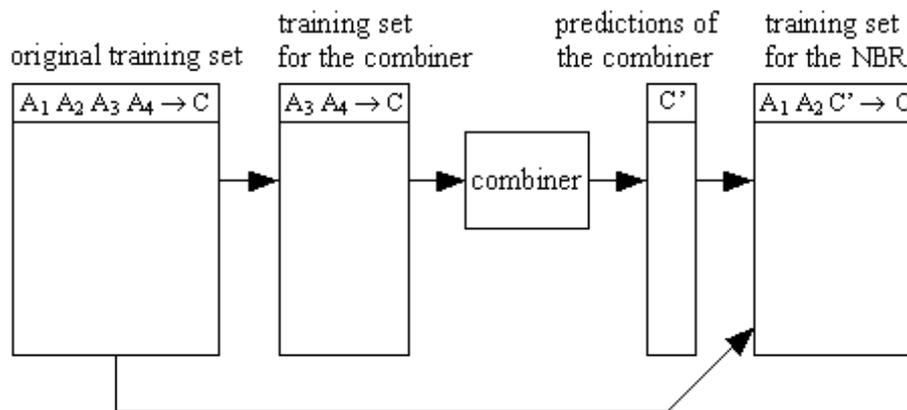


Figura 52. Gerando o conjunto de treinamento para o NBR.

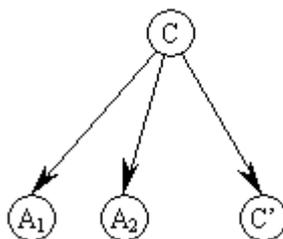


Figura 53. Estrutura do NBR structure depois da combinação dos atributos A_3 e A_4 .

A performance do modelo gerado, como na Figura 54, é avaliada com um conjunto de validação. Se a combinação resulta em melhoria, o modelo atual é selecionado e o procedimento continua, considerando combinações dois a dois no modelo modificado. Esse procedimento e a complexidade do algoritmo usado como combinador determinam o custo computacional da abordagem.

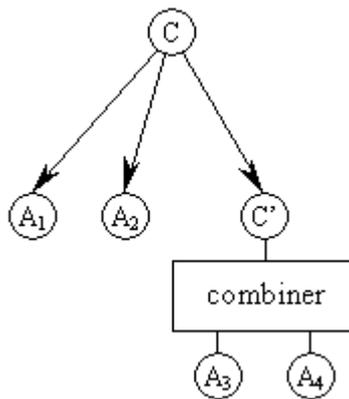


Figura 54. Modelo final gerado pela combinação dos atributos A_3 e A_4 .

8.2. Avaliação Experimental

Uma extensa avaliação experimental foi realizada com o objetivo de analisar a performance da nova abordagem. Esta seção descreve os experimentos e apresenta seus resultados.

Uma vez que o NBR é um algoritmo lento (devido a suas características inerentes), Decision Stumps (IBA e LANGLEY, 1992) para regressão (baseadas no erro médio quadrático) foram escolhidas como combinadores, porque, embora sejam algoritmos de aprendizado fracos, elas são rápidas, permitindo a obtenção de resultados em um tempo de execução viável. Algoritmos de regressão melhores serão usados como

combinadores em experimentos futuros. A função de perda usada nos experimentos com o NBR foi o erro quadrático. A implementação do algoritmo NBR foi fornecida por Eibe Frank, um de seus autores. Foi usada a implementação da Decision Stump do sistema WEKA (WITTEN e FRANK, 2005). Todo o código fonte foi escrito em Java.

Tabela 15. Conjuntos de dados.

Conjunto de Dados	Inst.	Ats.	Ats. comb.
autoMpg ¹	398	7	4,8
bank8FM ²	8193	8	4,2
basketball ¹	96	4	3,1
bolts ¹	40	7	3,9
breastTumor ¹	286	9	4,2
cloud ¹	108	6	4,1
cpu_small ²	8192	12	7,7
cpu ¹	209	7	4,8
delta_ailerons ²	7129	5	3,4
delta_elevators ²	9517	6	3,8
detroit ¹	13	13	7,0
diabetes_numeric ²	43	2	2,0
echoMonths ¹	130	9	3,6
elusage ¹	55	2	2,0
fishcatch ¹	158	7	4,3
fruitfly ¹	125	4	1,4
gascons ¹	27	4	2,4
hungarian ¹	294	13	9,2
longley ¹	16	6	3,2
lowbwt ¹	189	9	4,6
mbagrade ¹	61	2	2,0
pharynx ¹	195	11	5,5
puma8NH ²	8192	8	4,0
pwLinear ¹	200	10	6,5
quake ¹	2178	3	1,8
sensory ¹	576	11	5,8
servo ¹	167	4	3,0
strike ¹	625	6	3,5
veteran ¹	137	7	5,4
vineyard ¹	52	3	2,3

¹ Repositório de Aprendizado de Máquinas da UCI.

² Página pessoal de Luís Torgo.

Nos experimentos, 30 conjuntos de dados foram utilizados a fim de incluir muitos domínios e dificuldades. Os conjuntos de dados foram obtidos a partir do repositório de Aprendizado de Máquinas da Universidade da Califórnia (BLAKE e MERZ, 2005) e da página pessoal de Luís Torgo (<http://www.niaad.liacc.up.pt/~ltorgo/>). Nas primeiras três colunas da Tabela 15, apresentamos cada um dos conjuntos de dados usados, seu número de instâncias e seu número de atributos (excluindo o atributo objetivo). O método de teste usado nesta pesquisa foi o teste *t* emparelhado com validação cruzada de 10 partições (DIETTERICH, 1997; MITCHELL, 1997).

A quarta coluna da Tabela 15 mostra o número médio de atributos após o passo de combinação. Analisando os valores pode-se verificar que 2,7 combinações são aceitas em média.

A Tabela 16 mostra para cada conjunto de dados o erro médio quadrático (emq) alcançado pelo algoritmo NBR e pelo modelo com combinadores. Onde está marcado com “***”, a performance foi significativamente mais alta com nível de significância de 0,05, e onde está marcado com “*”, a performance foi significativamente mais alta com nível de significância de 0,1.

A Tabela 17 apresenta um resumo dos resultados. O modelo construído com combinadores alcançou EMQ menor em mais da metade dos conjuntos de dados testados. Se forem considerados apenas os resultados estatisticamente significativos nos níveis citados acima, a nova abordagem apresentou melhor performance que o NBR em sete conjuntos de dados (cinco com nível de significância de 0,05 e dois com nível de significância de 0,1) enquanto que o algoritmo NBR venceu significativamente (com nível de significância de 0,1) em apenas um conjunto de dados. Em cinco conjuntos de

dados houve empate. Três deles são conjuntos de dados com apenas dois atributos. Para estes conjuntos de dados não houve melhoria com o passo de combinação. Os outros dois empates foram verificados em conjuntos de dados para os quais ambos os modelos alcançaram erro praticamente zero.

Tabela 16. Resultados experimentais.

Conjunto de Dados	NBR	NBR + Combinadores
autoMpg	15,9308	16,6334
bank8FM	0,0049	0,0042**
basketball	0,0090*	0,0103
bolts	87,2708	62,7886
breastTumor	104,2892	101,8824
cloud	0,3373	0,3873
cpu_small	18,9957	18,4744
cpu	11353,9600	5669,3247
delta_ailerons	0,0000	0,0000
delta_elevators	0,0000	0,0000
detroit	3023,5271	874,9849
diabetes_numeric	0,3564	0,3564
echoMonths	152,6883	123,2886*
elusage	155,0093	155,0093
fishcatch	58501,5113	51005,7864
fruitfly	297,4220	269,5326*
gascons	235,5067	210,1115
hungarian	0,2067	0,2528
longley	795914,3021	1034660,3298
lowbwt	305216,4736	262223,0827
mbagrade	0,0960	0,0960
pharynx	105828,4724	118369,0570
puma8NH	17,4688	16,1270**
pwLinear	533,2886	493,7218**
quake	0,0381	0,0356**
sensory	0,7730	0,7368
servo	1,4082	0,7372**
strike	301324,7891	317903,4610
veteran	19574,5786	20325,9729
vineyard	9,3684	14,7656

Tabela 17. Resumo dos resultados

Medida	NBR	NBR + Combinadores
Número de vitórias	9	16
Número de vitórias significativas	1	7

Nos experimentos reportados em (FRANK *et al.*, 2000), regressão linear superou significativamente o algoritmo NBR em 18 conjuntos de dados, enquanto que o inverso ocorreu em apenas 8, em um total de 32 conjuntos de dados testados. Portanto, o NBR teve uma performance menor que regressão linear em mais de 56% dos testes, alcançando melhores resultados em apenas 25% deles.

A abordagem proposta foi comparada com regressão linear a fim de verificar se o uso de combinadores pode melhorar a performance do NBR para ser pelo menos comparável à regressão linear (PINA e ZAVERUCHA, 2008).

A Tabela 18 mostra os resultados da comparação. Onde está marcado com “(++)” (“(−)”), a performance foi significativamente maior (menor) que regressão linear no nível de significância de 0,05, e onde está marcado com “(+)” (“(−)”), a performance foi significativamente maior (menor) que regressão linear no nível de significância de 0,1.

Comparando NBR e regressão linear, pode ser visto que regressão linear alcança resultados significativamente melhores em 10 dos 30 conjuntos de dados, isto é, 33% deles, enquanto que o NBR alcança resultados significativamente melhores em 6 conjuntos de dados, ou seja, 20% deles. Deve-se notar que somente 20 dos 30 conjuntos de dados usados nesta pesquisa foram usados nos experimentos descritos em (FRANK *et al.*, 2000), daí as diferentes porcentagens.

Tabela 18. Comparação com regressão linear.

Data Set	Linear Regression	NBR	NBR + Combiners
autoMpg	14,6252	15,9308	16,6334
bank8FM	0,0015	0,0049 ⁽⁻⁾	0,0042 ⁽⁻⁾
basketball	0,0073	0,0090 ⁽⁻⁾	0,0103 ⁽⁻⁾
bolts	139,2701	87,2708 ⁽⁺⁺⁾	62,7886 ⁽⁺⁺⁾
breastTumor	98,3335	104,2892	101,8824
cloud	0,1692	0,3373 ⁽⁻⁾	0,3873 ⁽⁻⁾
cpu_small	97,7955	18,9957 ⁽⁺⁺⁾	18,4744 ⁽⁺⁺⁾
cpu	7077,5757	11353,9600	5669,3247
delta_ailerons	0,0000	0,0000	0,0000
delta_elevators	0,0000	0,0000	0,0000
detroit	3670,5263	3023,5271	874,9849 ⁽⁺⁾
diabetes_n.	0,4072	0,3564	0,3564
echoMonths	117,1016	152,6883 ⁽⁻⁾	123,2886
elusage	132,2234	155,0093	155,0093
fishcatch	16875,3082	58501,5113 ⁽⁻⁾	51005,7864
fruitfly	261,1713	297,4220 ⁽⁻⁾	269,5326
gascons	1757,5933	235,5067 ⁽⁺⁺⁾	210,1115 ⁽⁺⁺⁾
hungarian	0,1347	0,2067 ⁽⁻⁾	0,2528
longley	626534,0651	795914,3021	1034660,3298
lowbwt	290406,6596	305216,4736	262223,0827 ⁽⁺⁾
mbagrade	0,0848	0,0960 ⁽⁻⁾	0,0960 ⁽⁻⁾
pharynx	185388,5343	105828,4724 ⁽⁺⁺⁾	118369,0570 ⁽⁺⁾
puma8NH	19,9074	17,4688 ⁽⁺⁾	16,1270 ⁽⁺⁺⁾
pwLinear	494,49719	533,2886	493,7218
quake	0,0358	0,0381 ⁽⁻⁾	0,0356
sensory	0,8835	0,7730	0,7368
servo	0,7180	1,4082 ⁽⁻⁾	0,7372
strike	293905,0630	301324,7891	317903,4610
veteran	21724,2110	19574,5786	20325,9729
vineyard	14,2825	9,3684 ⁽⁺⁾	14,7656

Para 6 conjuntos de dados, nos quais NBR perdia para regressão linear significativamente, o uso de combinadores pôde melhorar a performance o bastante para tornar o NBR estatisticamente igual a regressão linear.

Para 2 conjuntos de dados, nos quais a diferença entre as performances do NBR e de regressão linear não era significativa, o uso de combinadores melhorou a performance do NBR de forma que este se tornou significativamente melhor que regressão linear.

Apenas para 3 conjuntos de dados a performance do NBR foi prejudicada significativamente pelos combinadores em comparação com regressão linear.

Como resumo desses resultados, o modelo construído com combinadores superou regressão linear em 7 conjuntos de dados (23% deles), ao passo que regressão linear só foi significativamente melhor em apenas 4 (13% deles).

A fim de verificar os resultados alcançados na comparação de EMQ, foi realizada a análise REC. Dois dos gráficos REC gerados podem ser vistos nas Figuras 55 e 56. Analisando os valores de AOC e a posição relativa das curvas REC, pôde-se concluir que o modelo construído com combinadores alcançou performance melhor. Pode-se notar que a curva para NBR + combinadores cobre as outras e portanto tal modelo é preferível.

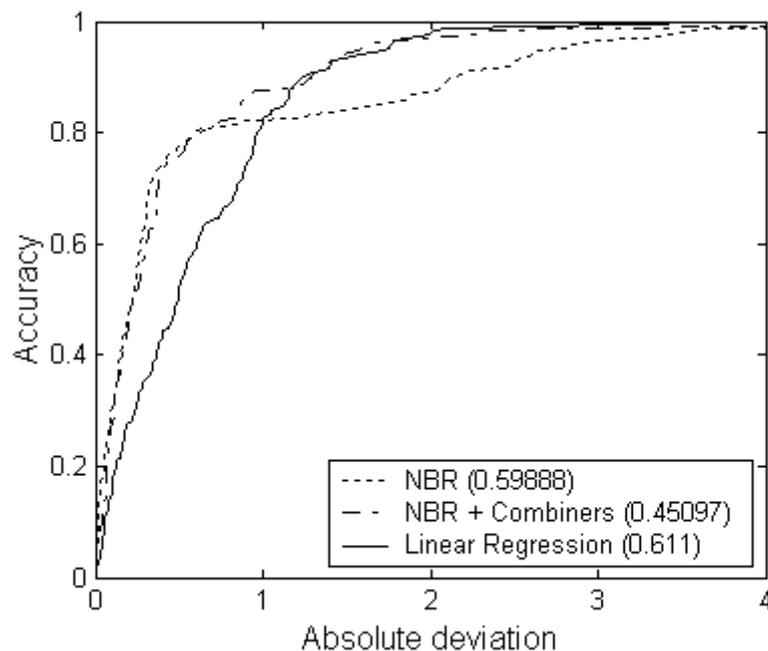


Figura 55. Gráfico REC para o conjunto de dados servo.

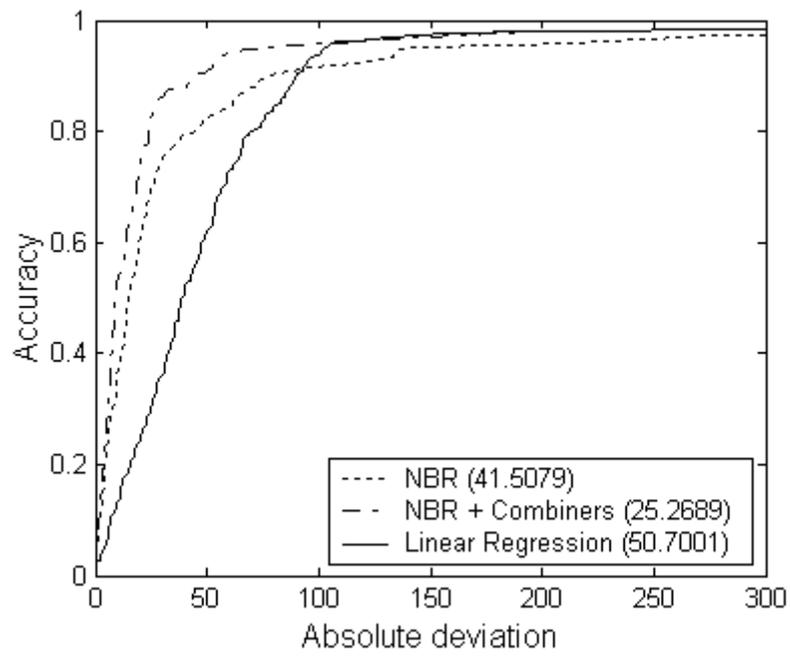


Figura 56. Gráfico REC para o conjunto de dados cpu.

9. CONCLUSÕES

As técnicas mais conhecidas e utilizadas para a previsão de séries temporais são baseadas em modelagem paramétrica. Os filtros de partículas que compõem o estado-da-arte para o tratamento de sistemas não-lineares e não-Gaussianos requerem a definição das funções que representam os modelos de espaço de estados escolhidos. Esta pesquisa apresenta uma abordagem para o uso de filtros de partículas que elimina a necessidade de definição de um modelo, aprendendo a dinâmica do sistema a partir de estimação não-paramétrica com *kernels*. As partes componentes do novo algoritmo, chamado filtro de partículas não-paramétrico, foram estudadas e decisões de projeto foram tomadas com a finalidade de maximizar a performance do sistema. A avaliação experimental mostrou que o filtro de partículas não-paramétrico apresenta um bom desempenho quando comparado com filtros de Kalman tradicionais e filtros de partículas utilizando redes neurais para modelar a função de transição de estados e a função observacional. Mesmo quando comparado a outros algoritmos que se beneficiaram de uma boa escolha de modelo de espaço de estados, o FPNP conseguiu resultados comparáveis. Deve-se notar que a escolha do modelo é crucial para a boa performance dos algoritmos que dependem de sua definição. Como pôde ser visto nos experimentos, algoritmos que apresentaram performance melhor que os demais para uma determinada escolha de modelo, se mostraram bastante ineficientes quando um modelo mais adequado foi definido para todos os algoritmos. Esse foi o caso, por exemplo, do GSPF. É muito importante salientar também que o filtro de partículas não-paramétrico não usou qualquer tipo de informação externa para fazer suas previsões e nenhum pré-processamento foi realizado nas entradas. Como trabalho futuro pretende-

se alterar a abordagem para que seja feito o pré-processamento da série de forma automática, de modo a determinar características que possam auxiliar as previsões e melhorar a performance do sistema. Também será feita uma análise para tentar identificar as características das séries temporais mais adequadas à abordagem proposta.

O preço que é pago pela alta performance de um sistema baseado em filtro de partículas é o elevado custo computacional. Pode ser que para alguns problemas a performance melhor de um filtro de partículas compense tal esforço computacional, enquanto que para outros não. Esse *trade-off* depende do problema tratado e deve ser investigado de forma individual. Entretanto, com a evolução da tecnologia e o aparecimento de computadores cada vez mais rápidos, o custo computacional de um sistema baseado em partículas como o proposto torna-se cada vez mais acessível.

Além do filtro de partículas não-paramétrico, nesta Tese foi introduzido o uso de curvas características de erro de regressão não só na avaliação comparativa de previsores de séries temporais e comitês de previsores, como também na seleção de componentes em comitês. Foi desenvolvido o algoritmo de boosting para regressão usando curvas REC para a seleção de modelos de comitês de redes neurais, e então a análise foi estendida à comitês de filtros de filtros de Kalman tradicionais e comitês de filtros de partículas. Uma extensa avaliação experimental foi realizada a fim de selecionar os melhores modelos de comitês de previsores de séries temporais. Foi também analisado o uso do filtro de partículas não-paramétrico como base de comitês, como forma de reduzir *bias* e variância do sistema. Os resultados foram melhores que os do algoritmo funcionando individualmente, como esperado.

Como contribuição final, foi desenvolvido um método de combinação de atributos para melhorar a performance do algoritmo NBR, que considerando o atributo

objetivo como *hidden*, pode ser usado para estimação de probabilidades, da mesma forma que o Naive Bayes já foi usado antes (LOWD e DOMINGOS, 2005), permitindo sua aplicação em previsão de séries temporais.

REFERÊNCIAS BIBLIOGRÁFICAS

- ALI, K. M., PAZZANI, M. J., 1996, “Error reduction through learning multiple descriptions”, *Machine Learning*, 24(3), pp. 173–202.
- ANDERSON, B. D. O., MOORE, J. B., 1979, *Optimal Filtering*, Prentice Hall Inc., Englewood Cliffs, New Jersey 07632.
- ANDREWS, M. W., 2005, *Bayesian Learning in Nonlinear State-Space Models*, Technical Report.
- AVNIMELECH, R., INTRATOR, N., 1999, “Boosting Regression Estimators”, *Neural Computation*, 11, pp. 491–513.
- BAUM, L. E., EGON, J.A., 1967, “An Inequality with Applications to Statistical Estimation for Probabilistic Functions of a Markov Process and to a Model for Ecology”, *Bull. Amer. Meteorol. Soc.*, 73, pp 360–363.
- BI, J., BENNETT, K. P., 2003, “Regression Error Characteristic Curves”, In: *Proceedings of the 20th International Conference on Machine Learning (ICML 2003)*, Washington, DC, pp. 43–50.

BILMES, J. A., 1998, *A gentle tutorial of the EM algorithm and its applications to parameter estimation for gaussian mixture and hidden Markov models*, Technical Report TR-97-021, International Computer Science Institute, Berkeley, California.

BLAKE, C. L., MERZ, C. J., 2005, *UCI Repository of Machine Learning Database, Machine-readable data repository*, Department of Information & Computer Science, University of California, Irvine.

[<http://www.ics.uci.edu/~mlearn/MLRepository.html>]

BORMAN, S., 2004, *The Expectation Maximization Algorithm*, Technical Report.

BREIMAN, L., 1996, “Bagging predictors”, *Machine Learning*, 24(2), 123-140.

BROCKWELL, P. J., DAVIS, R. A., 2002, *Introduction to Time Series and Forecasting*, Springer-Verlag, New York.

BROWN, G., WYATT, J. L., TIÑO, P., 2005, “Managing Diversity in Regression Ensembles”, *Journal of Machine Learning Research*, 6, pp. 1621–1650.

CARUANA, R., NICULESCU-MIZIL, A., 2004, “An Empirical Evaluation of Supervised Learning for ROC Area”, In: *Proceedings of the First Workshop on ROC Analysis (ROCAI 2004)*, pp. 1–8.

- CASELLA, G., ROBERT, C. P., 1996, “Rao-Blackwellisation of sampling schemes”, *Biometrika*, 83, pp. 81-94.
- DEMSAR, J., 2006, “Statistical Comparisons of Classifiers over Multiple Data Sets”, *Journal of Machine Learning Research*, 7, pp. 1–30.
- DIETTERICH, T. G., 1997, “Approximate Statistical Tests for Comparing Supervised Classification Learning Algorithms”, *Neural Computation*, 10, pp. 1895–1924.
- DIETTERICH, T. G., 1998, “Machine Learning Research: Four Current Directions”, *The AI Magazine*, 18, pp. 97–136.
- DIETTERICH, T. G., BARIKI, G., 1995, “Solving multiclass learning problems via error-correcting output codes”, *Journal of Artificial Intelligence Research*, 2, 263-286.
- DOUCET, A., 1998, *On sequential Monte Carlo methods for Bayesian filtering*, Technical Report, Department of Engineering, University of Cambridge, UK.
- DOUCET, A., de FREITAS, N., MURPHY, K., RUSSELL, S., 2000, “Rao-Blackwellised particle filtering for dynamic Bayesian networks”. In: *Proceedings of the Sixteenth Conference on Uncertainty in Artificial Intelligence*, pages 176-183, Stanford.

- DOUCET, A., de FREITAS, J. F. G., GORDON, N. J., 2001, *Sequential Monte Carlo Methods in Practice*, Springer-Verlag.
- DZEROSKI, S., ZENKO, B., 2004, “Is combining classifiers with stacking better than selecting the best one?”, *Machine Learning*, vol. 54, pp. 255–273.
- ELMAN, J. L., 1990, “Finding structure in time”, *Cognitive Science*, 14, pp. 179-211.
- FRANK, E., TRIGG, L., HOLMES, G., WITTEN, I. H., 2000, “Naive Bayes for regression,” *Machine Learning*, vol. 41, pp. 5–25.
- FREUND, Y., SCHAPIRE, R. E., 1996, “Experiments with a new boosting algorithm”, In: *Proceedings of the Thirteenth International Conference on Machine Learning*, Bari, Italy, pp. 148–156.
- FRIEDMAN, M., 1937, “The Use of Ranks to Avoid the Assumption of Normality Implicit in the Analysis of Variance”, *Journal of the American Statistical Association*, 32, pp. 675–701.
- FRIEDMAN, J., 1999, *Greedy function approximation: a gradient boosting machine*, Technical report, Stanford University Statistics Department.

FRIEDMAN, J., HASTIE, T., TIBSHIRANI, R., 2000, *Additive logistic regression: a statistical view of boosting*, Technical report, Stanford University Statistics Department.

GEMAN, S., BIENENSTOCK, E., DOURSAT, R., 1992, “Neural networks and the bias/variance dilemma”, *Neural Computation*, 4(1), pp. 1–58.

GHAHRAMANI, Z., ROWEIS, T., 1999, “Learning Nonlinear Dynamical Systems using an EM Algorithm”, *Advances in Neural Information Processing Systems*, 11, Cambridge, MA, MIT Press.

GORDON, N., SALMOND, D., SMITH, A., 1993, “Novel approach to nonlinear and non-Gaussian Bayesian state estimation”, *Proc. Int. Elect. Eng.*, 104, pp. 107-113.

HARRISON, P. J., STEVENS, C., 1976, “Bayesian forecasting (with discussion)”, *J. Roy. Statist. Soc.*, 38, 205-247.

HASTIE, T., TIBSHIRANI, R., FRIEDMAN, J., 2001, *The Elements of Statistical Learning – Data Mining, Inference, and Prediction*, Springer Series in Statistics.

HOLM, S., 1979, “A Simple Sequentially Rejective Multiple Test Procedure”, *Scandinavian Journal of Statistics*, 6, pp. 65–70.

- HUEBNER, U., KLISCHE, W., ABRAHAM, N. B., WEISS, C. O., 1989, “Comparison of Lorenz-like Laser Behavior with the Lorenz Model”, *Coherence and Quantum Optics VI*, Plenum Press, New York, p. 517.
- IBA, W., LANGLEY, P., 1992, “Induction of one-level decision trees,” In: *Proceedings of the 9th International Conference on Machine Learning*, Morgan Kaufmann, Aberdeen, Scotland, pp. 233–240.
- IMAN, R. L., DAVENPORT, J. M., 1980, “Approximations of the Critical Region of the Friedman Statistic”, *Communications in Statistics*, pp. 571–595.
- ITO, K., XIONG, K., 2000, “Gaussian Filters for Nonlinear Filtering Problems”, *IEEE Transactions on Automatic Control*, 45, pp. 910–927.
- JAZWINSKY, A., 1970, *Stochastic Processes and Filtering Theory*, Academic Press, New York.
- JORDAN, M. I., 1986, “Attractor dynamics and parallelism in a connectionist sequential machine”, In: *Proc. of the Eighth Annual Conference of the Cognitive Science Society*, NJ: Erlbaum, pp. 531-546.
- JORDAN, M. I., 2002, *A Introduction to Probabilistic Graphical Models*, University of California, Berkeley.

- JULIER, S., 2000, “The Scaled Unscent Transformation”, *Automatica*.
- JULIER, S., UHLMANN, J., 1996, *A General Method for Aproximating Nonlinear Transformations of Probability Distributions*, Technical Report, Department of Engineering Science, Oxford Univerty.
- JULIER, S., UHLMANN, J., DURRANT-WHYTE, H., 1995, “A New Approach for Filtering Nonlinear Systems”, In: *Proceedings of the American Control Conference*, pp. 1628–1632.
- KALMAN, R. E., 1960, “A New Approach to Linear Filtering and Prediction Problems, Journal of Basic Engineering”, *Transactions ASME*, Series D 82, pp. 35–45.
- KANAZAWA, K., KOLLER, D., RUSSELL, S., 1995, “Stochastic Simulation Algorithms for Dynamic Probabilistic Networks”, In: *Proceedings of the Eleventh Conference on Uncertainty in Artificial Intelligence*, Montreal, pp. 346-351.
- KANTZ, H., SCHREIBER, T., 1999, *Nonlinear Time Series Analysis*, Cambridge University Press.
- KEDEM, B., FOKIANOS, K., 2002, *Regression Models for Time Series Analysis*, John Wiley & Sons, Inc., New Jersey.

- KELVIN, T., LEUNG, D., PARKER, S., 2003, “Empirical Comparisons of Various Voting Methods in Bagging”, In: *Proceedings of KDD 2003*.
- KITAGAWA, G., 1987, “Non-Gaussian state space modelling of non-stationary time series”, *Journal of American Statistical Association*, 82, pp. 503–514.
- KITAGAWA, G., 1996, “Monte Carlo Filter and Smoother for Non-Gaussian Nonlinear State Space Models”, *JCGS*, 5, pp. 1-25.
- KOHAVI, R., JOHN, G. H., 1997, “Wrappers for Feature Subset Selection”, *Artificial Intelligence*, vol. 97, pp. 273–324.
- KOTECHA, J. H., DJURIC, P. M., 2003, “Gaussian Sum Particle Filtering”, *IEEE Transactions on Signal Processing*, vol. 51, no. 10, pp. 2602-2612.
- LAVRAC, N., DZEROSKI, S., 1994, *Inductive Logic Programming: Techniques and Applications*, Ellis Horwood, New York.
- LIU, J. S., CHEN, R., 1998, “Sequential Monte Carlo Methods for Dynamical Systems”, *J. Amer. Statist. Assoc.*, vol. 93, pp. 1032–1044.
- LOWD, D., DOMINGOS, P., 2005, “Naive Bayes Models for Probability Estimation”, In: *Proceedings of the 22nd ICML*, Alemanha, pp. 529-536.

MAYNE, D. Q., 1966, “A Solution of the Smoothing Problem for Linear Dynamic Systems”, *Automatica*, 4, pp. 73-92.

McNAMES, J., SUYKENS, J., VANDEWALLE, J., 1999, “Winning Entry of the K.U. Leuven time-series prediction competition”, *International Journal of Bifurcation and Chaos*, Vol.9, No.8, pp.1485-1500.

van der MERWE, R., de FREITAS, N., DOUCET, A., WAN, E., 2000, *The Unscented Particle Filter*, num. CUED/F-INFENG/TR 380, Cambridge University Engineering Department, Cambridge, England.

van der MERWE, R., WAN, E., 2001, “Efficient Derivative-Free Kalman Filters for Online Learning”, In: *Proceedings of the 9th European Symposium on Artificial Neural Networks (ESANN 2001)*, Bruges, Bélgica.

van der MERWE, R., WAN, E., 2003a, “Gaussian Mixture Sigma-Point Particle Filters for Sequential Probabilistic Inference in Dynamic State-Space Models”, In: *Proceedings of IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, Hong Kong.

van der MERWE, R., WAN, E., 2003b, “Sigma-Point Kalman Filters for Probabilistic Inference in Dynamic State-Space Models”, In: *Proceedings of the Workshop on Advances in Machine Learning*, Montreal, Canadá.

- MIGON, H. S., GAMERMAN, D., LOPES, H. F., FERREIRA, M. A. R., 2005, “Bayesian Dynamic Models”, In: *Handbook of Statistics*, 25, pp. 553-588.
- MITCHELL, T. M., 1997, *Machine Learning*, New York, McGraw-Hill.
- MUSSO, C., OUDJANE, N., LeGLAND, 2001, *Improving Regularized Particle Filters*, In *Sequential Monte Carlo Methods in Practice*, Springer, New York.
- NEMENYI, P. B., 1963, *Distribution-Free Multiple Comparisons*, Ph.D. Thesis, Princeton University.
- OMOHUNDRO, S. M., 1991, “Bumtrees for Efficient Function, Constraint, and Classification Learning”, *NIPS91*.
- PARMANTO, B., MUNRO, P. W., DOYLE, H. R., 1996, “Improving committee diagnosis with resampling techniques”, *Advances in Neural Information Processing Systems*, vol. 8, pp. 882-888 Cambridge, MA. MIT Press.
- PARZEN, E., 1962, “On Estimation of a Probability Density Function and Mode”, *Ann. Math. Stat.* 33, pp. 1065-1076.
- PAZZANI, M., 1996, “Searching for dependencies in Bayesian classifiers,” In: *Learning from Data: Artificial Intelligence and Statistics V*, D. Fisher and H. J. Lenz, Eds. New York, NY: Springer-Verlag, pp. 239–248.

- PERRONE, M. P., COOPER, L. N., 1993, “When networks disagree: Ensemble methods for hybrid neural networks”, *Neural networks for speech and image processing*, Chapman and Hall.
- PINA, A. C. de, ZAVERUCHA, G., 2005, “Boosting for Regression Using Regression Error Characteristic Curves”. In: *II Workshop on Roc Analysis in Machine Learning (ROCML)*, Bonn, Alemanha.
- PINA, A. C., ZAVERUCHA, G., 2006a, “Using Regression Error Characteristic Curves for Model Selection in Ensembles of Neural Networks”, *ESANN'2006 proceedings - European Symposium on Artificial Neural Networks*, Bruges, Bélgica, ISBN 2-930307-06-4, pp. 425-430.
- PINA, A. C. de, ZAVERUCHA, G., 2006b, “Applying REC Analysis to Ensembles of Sigma-Point Kalman Filters”. In: *III Workshop on Roc Analysis in Machine Learning (ROCML)*, Pittsburgh, USA,.
- PINA, A. C. de, ZAVERUCHA, G., 2006c, “Applying REC Analysis to Ensembles of Sigma-Point Kalman Filters”, In: *Proceedings of the 16th International Conference on Artificial Neural Networks (ICANN)*, Athens, Greece, Lecture Notes in Computer Science, Springer-Verlag, vol. 4132, pp. 151–160.
- PINA, A. C. de, 2006d, *Aprendizado Não-Paramétrico de Filtros de Partículas em Previsão de Séries Temporais*. Exame de Qualificação, COPPE/UFRJ.

- PINA, A. C. de, ZAVERUCHA, G., 2007a, “Melhorando a Performance do Algoritmo Naive Bayes para Regressão Através da Combinação de Atributos”. In: *VI Encontro Nacional de Inteligência Artificial (ENIA)*, Rio de Janeiro, RJ.
- PINA, A. C. de, ZAVERUCHA, G., 2007b, “Applying REC Analysis to Ensembles of Particle Filters”. In: *XX International Joint Conference On Neural Networks (IJCNN)*, Orlando, Flórida, USA.
- PINA, A. C. de, ZAVERUCHA, G., 2007c, *Neural Computing and Applications*, Springer London, aceito para publicação.
- PINA, A. C. de, ZAVERUCHA, G., 2008, “Combining Attributes to Improve the Performance of Naive Bayes for Regression”. In: *XXI International Joint Conference on Neural Networks (IJCNN)*, Hong Kong, China.
- PITT, M., SHEPHARD, N., 1999, “Filtering via simulation: Auxiliary Particle Filters”, *J. Amer. Statist. Assoc.*, vol 94, no 446, pp. 590-599.
- POLE, A., WEST, M., HARRISON, J., 1994, *Applied Bayesian Forecasting and Time Series Analysis*, Chapman-Hall, New York.

PROVOST, F., FAWCETT, T., 1997, “Analysis and Visualization of Classifier Performance: Comparison Under Imprecise Class and Cost Distributions”, In: *Proceedings of the International Conference on Knowledge Discovery and Data Mining (KDD'97)*, AAAI Press, pp. 43–48.

PROVOST, F., FAWCETT, T., KOHAVI, R., 1998, “The Case Against Accuracy Estimation for Comparing Classifiers”, In: *Proceedings of the 15th International Conference on Machine Learning (ICML 1998)*, Morgan Kaufmann, pp. 445–453.

QUINLAN, J. R., 1990, “Learning Logical Definitions from Relations”, *Machine Learning*, 5, pp. 239-266.

QUINLAN, J. R., 1992, “Learning with continuous classes”, In: *Proceedings of the 5th Australian Joint Conference on Artificial Intelligence*, Hobart, Tasmania, pp. 343–348.

RALAIVOLA, L., d'ALCHÉ-BUC, F., 2003, “Dynamical Modeling with Kernels for Nonlinear Time Series Prediction”, *Proceedings of NIPS'03*.

RUMELHART, D., HINTON, G., WILLIAMS, R. J., 1986, “Parallel Distributed Processing”, In: *Learning internal representations by error propagation*, MIT Press, Cambridge.

- SCHAPIRE, R. E., 1990, “The strength of weak learnability”, *Machine Learning*, 5, pp. 197–227.
- SCHAPIRE, R. E., 2002, “The boosting approach to machine learning: An overview”, *MSRI Workshop on Nonlinear Estimation and Classification*.
- SCOTT, D. W., 1997, *Density Estimation*, Technical Report, Rice University.
- SHUMWAY, R. H., STOFFER, D. S., 2006, *Time Series Analysis and Its Applications With R Examples*, 2nd Edition, Springer Science and Business Media, New York, NY.
- STAGGE, P., SENHO, B., 1997, “An extended elman net for modelling time series”, In: *International Conference on Artificial Neural Networks*.
- STORCH, H., ZWIERS, F.W., 1999, *Statistical analysis in climate research*, Cambridge University Press.
- SUEN, Y. L., MELVILLE, P., MOONEY, R. J., 2005, “Combining Bias and Variance Reduction Techniques for Regression Trees”, In: *Proceedings of the 16th European Conference on Machine Learning (ECML 2005)*, Porto, Portugal, pp. 741–749.

SUYKENS, J. A. K., VANDEWALLE, J., 2000, “The K.U.Leuven competition data: a challenge for advanced neural network techniques”, In: *Proceedings of the European Symposium on Artificial Neural Networks (ESANN’2000)*, Bruges, Bélgica, pp. 299-304.

TEIXEIRA, M., 2005, *Redes Bayesianas Dinâmicas para Previsão de Séries Temporais: Aplicação ao Setor Elétrico*, Tese de Doutorado, COPPE/UFRJ.

TEIXEIRA, M., ZAVERUCHA, G., 2003, “Fuzzy Bayes and Fuzzy Markov Predictors”, *Journal of Intelligent and Fuzzy Systems*, 13, pp. 155–165.

THRUN, S., LANGFORD, J. C., FOX, D., 1999, “Monte Carlo hidden Markov models: Learning nonparametric models of partially observable stochastic processes”, In: *Proc. 16th International Conference on Machine Learning*, Morgan Kaufmann, San Francisco, CA, pp. 415-424.

TUMER, K., GHOSH, J., 1996, “Error correlation and error reduction in ensemble classifiers”, *Connection Science*, 8(3-4), pp. 385–404.

WAND, M. P., JONES, M. C., 1995, *Kernel Smoothing*, Chapman & Hall, UK.

WEIGEND, A., GERSHENFELD, N., 1994, *Time Series Prediction - Forecasting the Future and Understanding the Past*, Addison-Wesley Publishing Company.

WEST, M., HARRISON, J., 1997, *Bayesian Forecasting and Dynamic Models*, Springer-Verlag, New York.

WILCOXON, F., 1945, “Individual Comparisons by Ranking Methods”, *Biometrics*, 1, pp. 80-83.

WITTEN, I. H., FRANK, E., 2005, *Data Mining: Practical Machine Learning Tools and Techniques*, 2nd ed., San Francisco, CA: Morgan Kaufmann.

WOLPERT, D., 1992, “Stacked generalization”, *Neural Networks*, 5, pp. 241–260.

ZHU, Y, SHASHA, D., 2003, “Efficient Elastic Burst Detection in Data Streams”,
In: *Proceedings of the 9th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, Washington, DC, pp. 336-345.

APÊNDICE

O teste de Wilcoxon (WILCOXON, 1945) é uma alternativa não-paramétrica ao teste t emparelhado, que faz o ranking das diferenças nas performances de dois algoritmos para cada conjunto de dados, ignorando os sinais, e compara os ranks para as diferenças positivas e negativas.

Seja d_i a diferença entre as performances de dois classificadores para o i -ésimo de N conjuntos de dados. Faz-se o ranking das diferenças de acordo com seus valores absolutos. No caso de empates, usa-se o rank médio. Seja R^+ a soma dos ranks para os conjuntos de dados nos quais o segundo algoritmo superou o primeiro, e R^- a soma dos ranks onde ocorre o contrário. Ranks de $d_i = 0$ são divididos igualmente entre os somatórios, se houver um número ímpar deles, um é ignorado:

$$R^+ = \sum_{d_i > 0} \text{rank}(d_i) + \frac{1}{2} \sum_{d_i = 0} \text{rank}(d_i) \quad (117)$$

$$R^- = \sum_{d_i < 0} \text{rank}(d_i) + \frac{1}{2} \sum_{d_i = 0} \text{rank}(d_i) \quad (118)$$

Seja T a menor das somas, $T = \min(R^+, R^-)$. Para valores de N até 25, em geral usa-se uma tabela de valores críticos exatos para T . Para um número maior de conjuntos de dados, a estatística:

$$z = \frac{T - \frac{1}{4}N(N+1)}{\sqrt{\frac{1}{24}N(N+1)(2N+1)}} \quad (119)$$

é aproximadamente normal. Com significância de 0,05, a hipótese nula pode ser rejeitada se z é menor que $-1,96$.

O teste de Wilcoxon assume comensurabilidade das diferenças, mas apenas qualitativamente: diferenças maiores contam mais, o que é provavelmente desejável, mas as magnitudes absolutas são ignoradas.

O teste de Wilcoxon assume diferenças contínuas d_i , portanto elas não devem ser arredondadas para uma ou duas casas decimais, pois isso diminuiria o poder do teste devido ao maior número de empates.

O teste de Friedman (FRIEDMAN, 1937) é um teste não-paramétrico que faz o ranking dos algoritmos para cada conjunto de dados separadamente, sendo que o melhor recebe o rank 1, o segundo melhor recebe o rank 2 etc. No caso de empates, usa-se o rank médio.

Seja r_i^j o rank do j -ésimo dos k algoritmos para o i -ésimo dos N conjuntos de dados. O teste de Friedman compara os ranks médios dos algoritmos:

$$R_j = \frac{1}{N} \sum_i r_i^j \quad (120)$$

Sob a hipótese nula, que afirma que todos os algoritmos são equivalentes e assim seus ranks R_j deveriam ser iguais, a estatística de Friedman:

$$\chi_F^2 = \frac{12N}{k(k+1)} \left[\sum_j R_j^2 - \frac{k(k+1)^2}{4} \right] \quad (121)$$

é distribuída de acordo com χ^2 com $k - 1$ graus de liberdade, quando N e k são grades o suficiente ($N > 10$ e $k > 5$). Para um número menor de algoritmos e conjuntos de dados, valores críticos exatos podem ser encontrados em tabelas.

Iman e Davenport (1980) mostraram que a estatística χ_F^2 é indesejavelmente conservadora e derivaram uma estatística melhor

$$F_F = \frac{(N-1)\chi_F^2}{N(k-1) - \chi_F^2} \quad (122)$$

que é distribuída de acordo com a distribuição F com $k - 1$ e $(k - 1)(N - 1)$ graus de liberdade. A tabela de valores críticos pode ser encontrada em qualquer livro de estatística.

Se a hipótese nula é rejeitada, pode-se prosseguir com um teste post-hoc.

O teste de Nemenyi (NEMENYI, 1963) é usado quando todos os classificadores são comparados dois a dois. A performance de dois classificadores é significativamente diferente se os ranks médios correspondentes diferem por pelo menos a diferença crítica:

$$DC = q_\alpha \sqrt{\frac{k(k+1)}{6N}} \quad (123)$$

onde os valores críticos q_α são baseados na estatística de limites Studentizada dividida por $\sqrt{2}$.

O teste de Holm (1979) é usado quando todos os classificadores são comparados com um classificador de controle. Embora seja mais conservador e tenha menor poder, nesse caso específico o teste de Holm é mais poderoso que o teste de Nemenyi.

A estatística para comparar o i -ésimo e o j -ésimo classificador usando esse método é:

$$z = \frac{(R_i - R_j)}{\sqrt{\frac{k(k+1)}{6N}}} \quad (124)$$

O valor de z é usado para descobrir a probabilidade correspondente a partir de uma tabela da distribuição normal, que é então comparada com uma significância apropriada α (o valor é ajustado para compensar as múltiplas comparações).

O teste de Holm sequencialmente testa as hipóteses ordenadas por sua significância. Sejam p_1, p_2, \dots, p_{k-1} os p valores ordenados tais que $p_1 \leq p_2 \leq \dots \leq p_{k-1}$. O teste compara cada p_i com $\alpha/(k - i)$, começando com o p valor mais significativo. Se p_1 é menor que $\alpha/(k - 1)$, a hipótese correspondente é rejeitada e então o teste prossegue comparando p_2 é menor que $\alpha/(k - 2)$, e assim por diante. Tão logo uma hipótese nula não possa ser rejeitada, todas as outras hipóteses também são mantidas.

Algumas vezes o teste de Friedman reporta uma diferença significativa, mas o teste post-hoc falha em detectá-la. Isso é devido ao menor poder do teste post-hoc. Nenhuma outra conclusão além de que alguns algoritmos diferem pode ser tirada nesse caso.