UNIVERSIDADE FEDERAL DO RIO GRANDE DO SUL
ESCOLA DE ENGENHARIA
DEPARTAMENTO DE ENGENHARIA QUÍMICA
PROGRAMA DE PÓS-GRADUAÇÃO EM ENGENHARIA QUÍMICA

Novas Metodologias para Compressão de Dados de Processos e para o Ajuste do Sistema PI

DISSERTAÇÃO DE MESTRADO

Rodrigo Paulo Silveira

Porto Alegre
2012

UNIVERSIDADE FEDERAL DO RIO GRANDE DO SUL
ESCOLA DE ENGENHARIA
DEPARTAMENTO DE ENGENHARIA QUÍMICA
PROGRAMA DE PÓS-GRADUAÇÃO EM ENGENHARIA QUÍMICA

Novas Metodologias para Compressão de Dados de Processos e para o Ajuste do Sistema PI

Rodrigo Paulo Silveira

Dissertação de Mestrado apresentada como requisito parcial para obtenção do título de Mestre em Engenharia

Área de concentração: Pesquisa e Desenvolvimento de Processos.

Linha de Pesquisa: Engenharia de Sistemas – Projeto, Simulação, Modelagem, Controle e Otimização de Processos.

Orientadores:

Prof. Dr. Jorge Otávio Trierweiler

Prof. Dr. Marcelo Farenzena

Porto Alegre

2012

UNIVERSIDADE FEDERAL DO RIO GRANDE DO SUL

ESCOLA DE ENGENHARIA

DEPARTAMENTO DE ENGENHARIA QUÍMICA

PROGRAMA DE PÓS-GRADUAÇÃO EM ENGENHARIA QUÍMICA

A Comissão Examinadora, abaixo assinada, aprova a Dissertação *Novas Metodologias para Compressão de Dados de Processos e para o Ajuste do Sistema PI*, elaborada por Rodrigo Paulo Silveira, como requisito parcial para obtenção do Grau de Mestre em Engenharia.

Comissão Examinadora:

Dr. Mario Cesar Mello Massa de Campos

Prof. Dr. Celso José Munaro

Dr. Marcelo Escobar Aragão

Dedico este trabalho ao meu pai Edu Silveira (*in memorian*).

"*A vida só tem sentido na luta. O triunfo ou a derrota está nas mãos dos deuses...*
Então, celebremos a luta!"

Canto de guerra Swahili.

Agradecimentos

Em primeiro lugar, quero agradecer aos meus orientadores, professores Jorge Trierweiler e Marcelo Farenzena pela oportunidade, pelo auxílio nos momentos de dúvidas e discussões, pelo apoio técnico e moral e pelos conselhos sempre pertinentes. Também, à UFRGS e ao Departamento de Engenharia Química (DEQUI) pela infra-estrutura e à PETROBRAS S/A pelo suporte oferecido.

Quero dedicar sinceros agradecimentos aos meus colegas e amigos do DEQUI e aos amigos de longa data pelas vezes em que conversamos, em que me ajudaram e quando nos divertimos, pois esses momentos foram muito importantes nesta caminhada. Um muito obrigado também àqueles que, mesmo em passagem breve pela minha vida, contribuíram para o meu crescimento e para a conclusão deste objetivo.

Por fim quero agradecer com todo o meu amor às pessoas mais importantes: minha família. Às minhas irmãs, minha sobrinha e minha mãe pelo apoio e amor incondicionais, e em especial ao meu pai, que me guarda onde quer que eu esteja, meu muito obrigado.

Resumo

A consolidação da automação na indústria química trouxe um desafio: transformar o grande volume de dados provenientes dos processos em informação útil. Com o advento dos sistemas digitais, a expansão no desenvolvimento de sensores e métodos de amostragem possibilitou a aquisição de uma elevada quantidade de dados advindos do campo industrial, de forma que, atualmente, engenheiros e operadores dificilmente carecem de informações sobre as diferentes variáveis de um processo.

Os algoritmos para compressão de dados de planta surgiram como alternativa para reduzir o espaço de armazenamento demandado por essas informações. Comprimir dados significa gravar apenas uma porção da informação original, devendo-se preservar, no entanto, as características importantes que elas carregam. Ultimamente, não é apenas o espaço em disco que deve ser priorizado quando se fala em compressão. É necessário que os dados sejam fieis às reais informações do processo e, além disso, devem ser recuperados e transmitidos em velocidade razoável.

Neste trabalho são apresentadas duas metodologias para a compressão de dados de processos químicos. Para tanto, são utilizadas técnicas para o cálculo da derivada de sinais ruidosos e curvas polinomiais (*splines* cúbicas) que preservam as características dos dados. Também são propostas sistemáticas para o ajuste automático dos parâmetros nas rotinas de compressão do sistema PI da OSIsoft®.

A fim de avaliar essas propostas, foram utilizados alguns estudos de caso, compreendendo sinais reais de uma planta laboratorial e outros artificiais gerados por simulação computacional, abrangendo dinâmicas distintas e características peculiares. Os resultados mostram que os algoritmos são promissores para gravar dados de processos sem que se perca a essência da informação, a qual armazenada pode ser transformada em conhecimento sem prejuízo da qualidade em termos dinâmicos e estáticos.

Abstract

The consolidation of the automation in the chemical industry has brought a challenge: translating the large volume of data in useful information. With the advent of digital systems, the expansion in the development of sensors and sampling methods allowed the acquisition of a large amount of data from the process field. Thus, nowadays, engineers and operators rarely suffer from lack of information arising from the several process variables.

The algorithms for data compression appeared as an alternative to reducing the storage space demanded by these information. Data compressing means to record only a portion of the original information, preserving, however, the relevant features that they hold. Recently, is not only the disk space that must be prioritized when one talks about compression. It is necessary data to be reliable to the actual process information, and, moreover, must be retrived and transmitted at an aceptable speed.

In this work, two methodologies for data compression of chemical processes are presented. Therefore, techniques to estimate derivatives of noisy signals and polynomial curves (cubic splines), which preserve the data features, are used. Also, systematic approaches to automate the tuning of the parameters in the OSIsoft® PI System® compression routines are proposed.

In order to evaluate these proposals, some case studies are taken, composed by real signals, from a laboratorial plant, and artificial ones generated by computacional simulation, embracing diversified dynamics and peculiar features. The results show that the proposed algorithms are promising to store processes data without significant loss of information, which can be converted in knowledge without jeopardizing the quality in static and dynamic aspects.

Sumário

Lista de figuras

Lista de tabelas

Capítulo 1

Introdução

1.1 Gestão das informações de processos

O avanço tecnológico dos sistemas digitais e o recente progresso no desenvolvimento de sensores e de métodos para amostragem de dados cada vez mais rápidos provocou um aumento considerável dos dados coletados nas plantas de processos químicos (Misra *et al.*, 2000; Singhal e Seborg, 2005). No início deste século, estimou-se que, no mundo inteiro, era produzida uma quantidade de 1,5 bilhões de GB de dados por ano, armazenados em documentos impressos, dispositivos filmográficos, óticos e magnéticos (Lyman *et al.*, 2000). Uma parcela dessa produção provém dos processos nas indústrias de transformação, acompanhando o crescimento da tecnologia.

De uma forma genérica pode-se representar o fluxo das informações de um processo automatizado hierarquizando-o em uma estrutura conforme mostra a Figura 1.1. Os dados provenientes do "chão-de-fábrica", através dos equipamentos e dos sensores, aquisitados e processados por ferramentas em nível supervisório (CLP, SCADA), em geral necessitam de sistemas que centralizem e disponibilizem a informação para os níveis corporativos da pirâmide de controle. É nessa parte da estrutura que se encontram os Sistemas de Gestão da Informação de Processos, ou PIMS (*Process Information Management Systems*, do termo em inglês).

Os PIMS são grandes sistemas cuja função é coletar e organizar dados do "chão-de-fábrica" (através da aquisição em nível supervisório), além de armazená-los e disponibilizá-los para aplicações em nível de planejamento. Esses sistemas surgiram na indústria de processos contínuos, mais especificamente nas indústrias química e petroquímica, porém têm conquistado outros nichos de mercado, como papel e celulose, siderurgia, mineração, etc. (Seixas Filho, 1993).

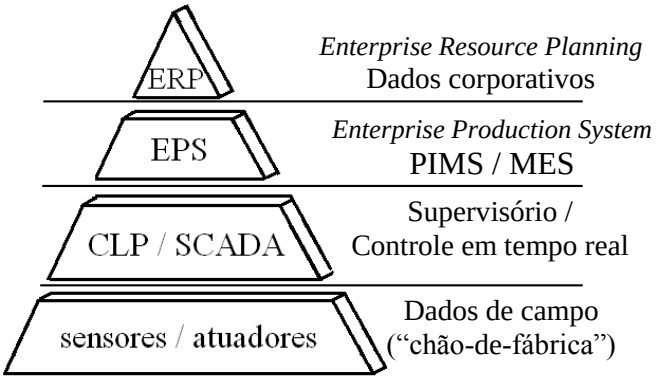


Figura 1.1: Pirâmide hierárquica de automação: os PIMS realizam a interface entre os dados coletados em campo e as decisões corporativas.

A boa gerência dos dados oriundos de um processo desempenha um papel importante na indústria. Deles depende a eficiência de suas operações, que é sustentada pela prática de diversos elementos, entre os quais estão (Bakshi e Stephanopoulos, 1996):

- o acompanhamento estatístico e o planejamento da produção;
- processos de melhoria contínuos;
- controle de qualidade;
- monitoramento, detecção e diagnóstico de falhas;
- modelagem e caracterização de sistemas;
- validação e identificação de modelos;
- auditoria de desempenho de malhas de controle.

Dos PIMS existentes no mercado, atualmente, os mais consolidados são o Plant Information™ (PI) da OSIsoft® e o InfoPlus® da AspenTech®. Para citar outros sistemas, o Enterprise Historian (ABB), o Uniformance/PHD Process History Database (Honeywell) e o Exaquantum (Yokogawa) também possuem fatias do mercado com essas aplicações. A Figura 1.2 mostra de forma simplificada a estrutura do sistema PI em típica implementação industrial, desde a aquisição dos dados em campo até a sua utilização para análise ou outras aplicações finais.

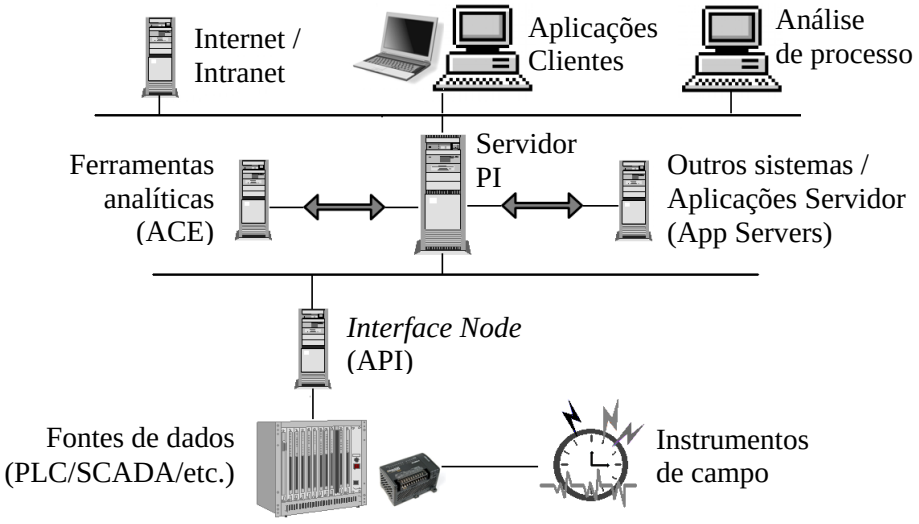


Figura 1.2: Caminho da informação no sistema PI da OSIsoft®.

Os constituintes principais de um PIMS são o **historiador** de processos, a **interface gráfica** para recuperação e visualização dos dados armazenados e as **aplicações clientes** complementares (Seixas Filho, 1993). Em um histórico, por exemplo, a base de dados é registrada, geralmente, como uma lista contendo os seguintes atributos:

- *Time stamp*: armazena data e tempo do correspondente valor;
- Identificação do dado (*tag*);
- Valor;
- Qualidade do dado: informa se o dado não é considerado confiável.

Um sistema de gerenciamento eficaz é capaz de concentrar a massa de dados e permitir transformá-los em informação útil, e essa informação em conhecimento. A Figura 1.3 ilustra essa ideia. Dados com elevado valor agregado são aqueles que, em pouca quantidade, permitem que deles se extraia conhecimento. No outro extremo, os dados brutos estão em grande quantidade e possuem pouco valor, pois a informação relevante não pôde ser isolada, e o conhecimento torna-se mais difícil de ser obtido. Assim, entre outros benefícios, um bom sistema de gerenciamento possibilita ao engenheiro de processos entender as situações operacionais que se apresentam comparando-as com padrões previamente arquivados.

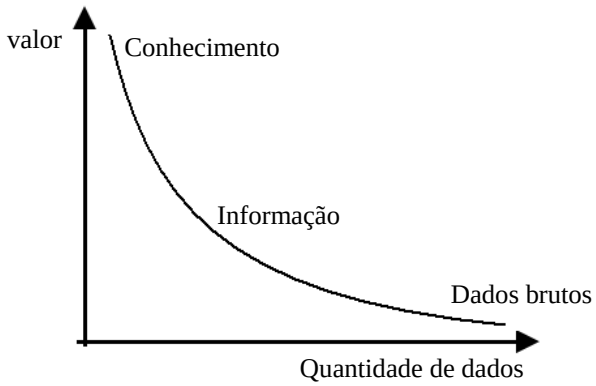


Figura 1.3: Transformando dados brutos em conhecimento (adaptado de: Seixas Filho, 1993).

1.2 Compressão de dados na indústria de processos

A compressão de dados tem se tornado parte essencial na integração dos computadores digitais com os processos da indústria química (Misra *et al.*, 2000), devido à quantidade de dados coletados, propiciado pelo progresso tecnológico, e à necessidade de geri-los para convertê-los em conhecimento. Comprimir dados significa eliminar informações redundantes de forma a guardar apenas a informação relevante que estes dados carregam. A Figura 1.4 mostra um exemplo de um típico sinal de processo submetido a um processo de compressão.

Os algoritmos de compressão surgiram, inicialmente, com o intuito de atingir requisitos mínimos de espaço em disco, pois o crescimento na escala dos processos e na produção de insumos não foi acompanhado pela evolução da tecnologia dos mecanismos para armazenamento em massa. Atualmente, grandes quantidades de memória estão disponíveis para guardar dados a um custo relativamente baixo. No entanto o armazenamento eficiente da

informação é de crucial importância quando deles se exige fidelidade ao processo real, ou quando estes dados necessitam ser transmitidos via rede entre diferentes setores da companhia ou entre sítios de produção distantes. Além disso, ele facilita o processo de busca da informação relevante.

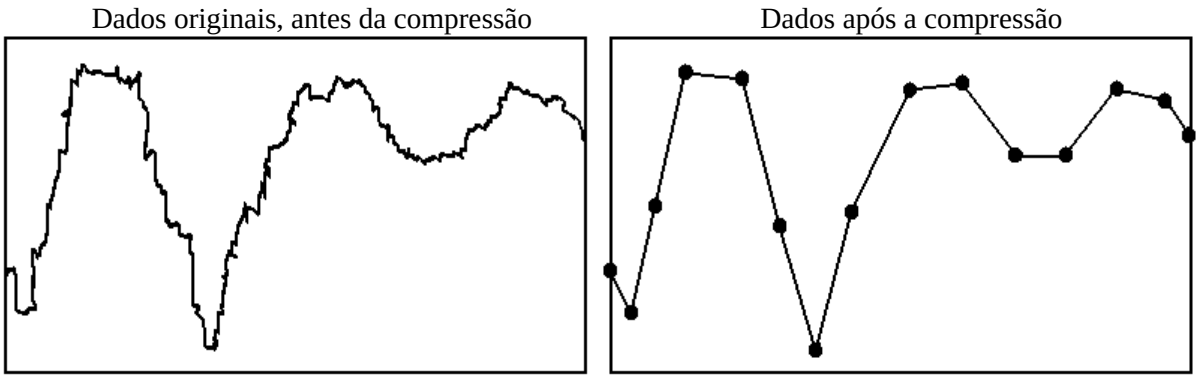


Figura 1.4: Exemplo de compressão de um conjunto de dados típico da indústria de processos. Os pontos escuros representam a informação que foi efetivamente gravada.

Os algoritmos para compressão de dados podem ser divididos em classes fundamentais, a saber (Souza *et al.*, 2005):

- sem perda de informação: a compressão é baseada na identificação de redundâncias, sendo que, na descompressão, os dados são integralmente reconstruídos. Muito utilizados na compressão de arquivos de texto e de programas, no formato *zip*, por exemplo;
- com perda de informação: o processo de compressão identifica redundâncias que são descartadas, e na reconstrução elas não são recuperadas. São, por sua vez, subdivididos em algoritmos de amostragem a intervalos fixos e de amostragem variável. Nesta classe encontram-se os algoritmos para compressão de sinais de processos. Destacam-se os *lineares por partes* (*piecewise*) e os que utilizam transformadas *wavelets*.

São os algoritmos com perda de informação, portanto, a classe de rotinas de compressão encontradas nos PIMS comerciais. Apesar do relativo sucesso na indústria, estes algoritmos são bastante limitados no que diz respeito à qualidade dos dados comprimidos. Após o surgimento das primeiras técnicas, poucas contribuições inovaram em direção à compressão de dados na indústria de processos. Alguns dos algoritmos existentes podem atingir elevadas taxas de compressão, porém comprometem a fidelidade da informação original. Também, a reconstrução dos dados é realizada por uma simples interpolação, de ordem zero (útil apenas para variáveis que permanecem em longos períodos em estado estacionário) ou linear (ordem 1), que provê uma reconstrução mais acurada, porém ainda modesta para aplicações mais exigentes. Além disso, os métodos de compressão implementados necessitam do ajuste de determinados parâmetros, que são, muitas vezes, deixados em seus valores padrão, principalmente porque não há sistemáticas de ajuste ou porque este processo é dispendioso (Alsmeyer, 2006).

1.3 Objetivos

Este trabalho visa propor novas metodologias para a compressão de dados, procurando atingir um bom compromisso entre espaço em disco (quantidade de informação armazenada) mantendo as características relevantes dos sinais típicos provenientes da operação de plantas de processos químicos. Além disso, os dados comprimidos devem ser capazes de gerar conhecimento, através da sua utilização para diversas finalidades como identificação de modelos, reconciliação de dados, auditoria de desempenho de malhas de controle, entre outras aplicações importantes para a engenharia de processos. Também é abordada aqui a compressão de dados do sistema comercial PI, da OSIsoft®, em que são propostas algumas metodologias para sintonizar seus parâmetros, de forma que possam ser aproveitadas nos processos reais da indústria, onde este sistema está implementado.

1.4 Estrutura da Dissertação

A presente Dissertação está organizada da seguinte forma: no capítulo 2 encontra-se uma revisão dos métodos já desenvolvidos para a compressão de dados de processos; o capítulo 3 apresenta as metodologias e as ferramentas utilizadas para realizar a compressão de dados nas propostas do trabalho; no capítulo 4 são sugeridas novas contribuições para o ajuste dos parâmetros da compressão do PIMS PI; a seguir, no capítulo 5, são apresentados estudos de caso para avaliar as metodologias propostas nos capítulos anteriores, algumas aplicações e os critérios utilizados; o capítulo 6 apresenta e discute os resultados obtidos e, por fim, no capítulo 7, as conclusões são tomadas juntamente com a sugestão de trabalhos futuros.

1.5 Principais definições e conceitos usados nesta Dissertação

Alguns termos serão usados ao longo deste texto e serão apresentados nesta seção como um guia para consultas breves. São eles:

Ajuste ou ***sintonia***: é a especificação dos parâmetros que afetam a qualidade da compressão nos algoritmos.

CompDev: desvio de compressão das rotinas do sistema PI.

Descompressão ou ***reconstrução***: é a interpolação dos pontos gravados na compressão, por métodos lineares ou polinomiais.

Erro de Reconstrução (***ER***): erro médio quadrático entre os dados originais e os dados reconstruídos após a compressão.

ExcDev: desvio de exceção das rotinas do sistema PI.

G: modelo de processo utilizado nas simulações para gerar os sinais artificiais.

idG: modelo de processo identificado utilizando os dados brutos das simulações.

idRefin: modelo de processo identificado utilizando os dados das simulações comprimidos pelo método Refinado.

idEvol: modelo de processo identificado utilizando os dados das simulações comprimidos pelo método Evolutivo

IEA: integral do erro absoluto.

IEQ: integral do erro quadrático.

Janela: conjunto de dados de uma série temporal.

PI: abreviação para o nome da ferramenta comercial *Plant Information*, da empresa OSIsoft®.

Piecewise Cubic Hermite Interpolating Polynomial (***PCHIP***): *spline* cúbica de Hermite, usada na reconstrução dos dados nos métodos propostos.

Razão de Compressão (***RC***): relação entre o número de dados originais e o número de dados gravados na compressão.

Referências

Alsmeyer, F. (2006) Automatic adjustment of data compression in process information management systems. *16th European Symposium on Computer Aided Process Engineering and 9th International Symposium on Process Systems Engineering*, 21**,** 1533-1538.

Bakshi, B. R. & G. Stephanopoulos (1996) Compression of chemical process data by functional approximation and feature extraction. *Aiche Journal*, 42**,** 477-492.

Souza, A. J. de, Feijó, R. H., Leitão, G. B. P., Medeiros, A. A. D., Bezerra, C. G., De Andrade, W. L., Guedes, L. A.& Maitelli, A. L.. GERÊNCIA DE INFORMAÇÕES DE PROCESSOS INDUSTRIAIS: UM ESTUDO DE CASO NA PRODUÇÃO DE PETRÓLEO E GÁS. In: CONGRESSO BRASILEIRO EM P&D EM PETRÓLEO E GÁS, 3. Salvador: 2-5 de out, 2005.

Lyman, P., Varian, H. R., Dunn, J., Strygin, A. & Swearingen, K. (2000). "How Much Information?" Project. School of Information Management and Systems, Universidade da California. Berkeley, CA. Disponível em < http://www.attitudeweb.be/doc/ resources/studies/how_much_information_produced_world_year_en.pdf >.

Misra, M., S. J. Qin, S. Kumar & D. Seemann (2000) On-line data compression and error analysis using wavelet technology. *Aiche Journal*, 46**,** 119-132.

Seixas Filho, C. (1993). Notas de Aula – Capitulo 6 PIMS – Process Information Management System. Disponível em: < http://www.cpdee.ufmg.br/~seixas/PaginaII /Download/DownloadFiles>.

Singhal, A. & D. E. Seborg (2005) Effect of data compression on pattern matching in historical data. *Industrial & Engineering Chemistry Research*, 44**,** 3203-3212.

Capítulo 2

Revisão Bibliográfica

No presente capítulo encontram-se descritos os principais algoritmos para compressão aplicados a dados de processos químicos descritos na literatura. As duas primeiras seções abordam as metodologias mais tradicionais: os métodos diretos, também conhecidos como lineares por partes, e os métodos baseados em transformadas. A terceira seção revisa brevemente a compressão de dados utilizando quantização vetorial, e a quarta seção apresenta alguns métodos alternativos que não se inserem nas classificações habituais. Na seção seguinte, são revisadas as metodologias existentes para sintonizar os parâmetros de compressão das diferentes técnicas. Alguns trabalhos comparativos e que apresentam critérios para avaliar o impacto dos algoritmos de compressão na análise dos dados de processo são encontrados na sexta seção. Por fim, é apresentada uma avaliação geral, segundo alguns critérios considerados, de todos os métodos aqui abordados.

2.1 Métodos de compressão diretos

Os métodos diretos são aqueles usados atualmente na indústria, principalmente porque podem ser aplicados em tempo real, no momento da aquisição dos dados. Também são conhecidos como "lineares por partes" (ou *piecewise linear*, no termo em inglês), porque é assumido que o sinal segue uma linha reta enquanto pontos amostrados estiverem dentro de uma tolerância especificada. Comercialmente, podem-se destacar os algoritmos *Boxcar* e *Backslope* (Hale e Sellars, 1981), presentes no PIMS da Aspentech®, o InfoPlus®, e o *Swinging Door* (Bristol, 1990) no PI da OSIsoft®. Além desses, há também o método conhecido por SLIM (Kortman, 1967), acrônimo para *Straight-Line-Interpolative-Method*, com suas variações − SLIM1, SLIM2 e SLIM3 (James, 1995). Outros métodos também foram propostos na literatura em abordagens alternativas, como o algoritmo PLOT (Mah *et al.*, 1995).

2.1.1 Boxcar e Backslope

O algoritmo Boxcar (BC), proposto por Hale e Sellars (1981), grava um valor apenas quando o valor atual ou corrente diferir do último valor gravado por uma quantidade que

extrapolar o limite de gravação especificado pelo usuário. O funcionamento deste método pode ser representado pela Figura 2.1a, onde a largura da "caixa" horizontal corresponde a duas vezes o valor do limite de gravação. Esse algoritmo é bastante simples e mostra-se eficiente para variáveis que operam em longos períodos de estado estacionário.

Por outro lado, o BC não fornece bons resultados quando há mudanças na tendência do processo, como *drifts* ou transições bruscas de pontos operacionais. A fim de contornar esse problema, os mesmos autores desenvolveram o algoritmo chamado *Backward Slope* (ou somente "Backslope", BS). Nesse algoritmo, o valor atual da variável é predito através de uma extrapolação linear entre os dois últimos valores gravados. Se a amostragem atual diferir do valor extrapolado por uma quantidade maior que um limite de gravação pré-determinado, então o ponto anterior ao atualmente amostrado é gravado, como mostra a Figura 2.1b. Os algoritmos Boxcar e Backslope podem ser utilizados em conjunto (BC/BS) de tal forma que um dado só é gravado quando violar os dois critérios simultaneamente (Figura 2.1c). Essa abordagem é utilizada no PIMS InfoPlus® (Aspentech® Technology Inc.).

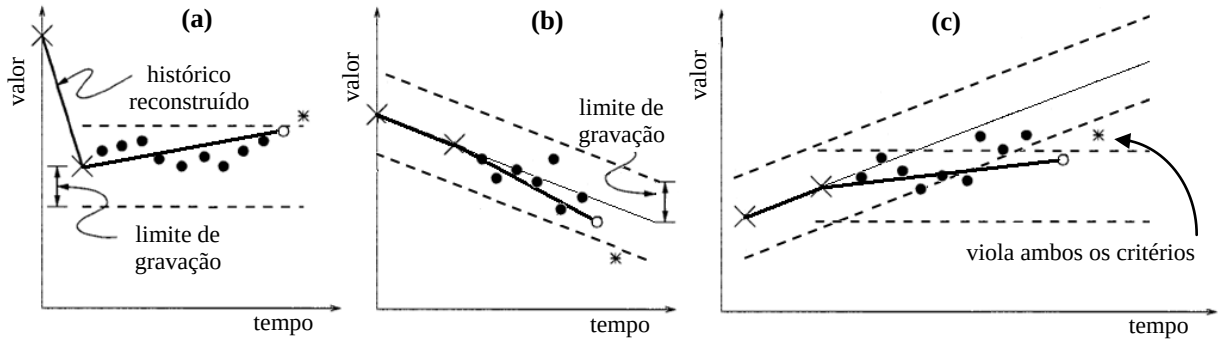


Figura 2.1: Esquema de funcionamento na compressão dos métodos (a) Boxcar, (b) Backslope, e (c) do sistema InfoPlus®, com ambos os métodos aplicados simultaneamente; (X) valores gravados; (o) último ponto gravado; (●) valores não gravados e (*) valor que causa uma gravação.

2.1.2 Swinging Door

O Swinging Door Trending (SDT) foi originalmente apresentado por Bristol (1990), e seu funcionamento está esquematizado na Figura 2.2. Assumindo que o algoritmo inicia no ponto *i*, constroem-se duas "portas" cujos suportes estão a uma distância *D* desse ponto. As portas estendem-se dos suportes até o ponto atual e abrem-se conforme novos dados são adquiridos. Uma vez abertas, as portas não mais fecham até que a soma dos seus ângulos interiores seja igual ou maior que 180°. Quando isso ocorrer, o processo é interrompido, e o ponto anterior ao atualmente amostrado é armazenado, iniciando, a partir da sua posição, o procedimento descrito até esta etapa.

A concepção original do algoritmo SDT foi adaptada nas implementações comerciais, como é o caso do sistema PI. A Figura 2.3 mostra esquematicamente o funcionamento desse método no sistema da OSIsoft® (OSIsoft® Inc.). O SDT está presente no *teste de compressão*, realizado no servidor do sistema, utilizando os valores de uma etapa prévia chamada *teste de exceção*, feita na interface. Nessa última, chama-se "exceção" o ponto que causa uma gravação. O teste de exceção tem seu funcionamento semelhante ao algoritmo BC, com dois parâmetros de ajuste: o *desvio de exceção*, representado pelo limite de gravação da Figura

2.1a, e o *tempo máximo para exceção*, que pode ser entendido como o comprimento máximo do retângulo horizontal que se permite não ocorrer uma exceção, enquanto as amostragens não violarem o desvio de exceção.

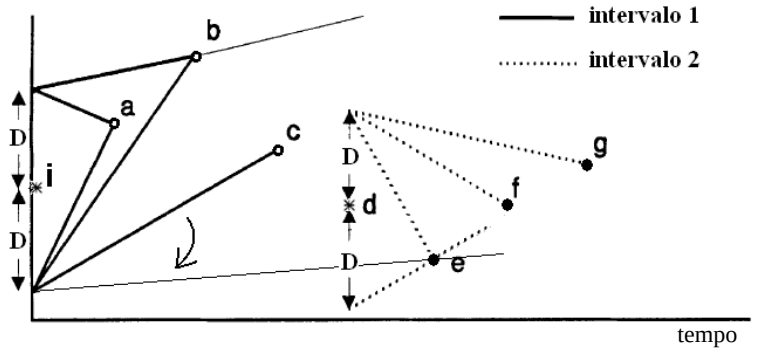


Figura 2.2: Dois intervalos no processo de compressão do algoritmo SDT proposto por Bristol (1990). No primeiro intervalo, o ponto *e* faz com que a porta inferior viole o critério; (o) pontos não gravados, (*) pontos gravados e (●) próximos a serem amostrados. (Adaptado de Mah *et al.*, 1995.)

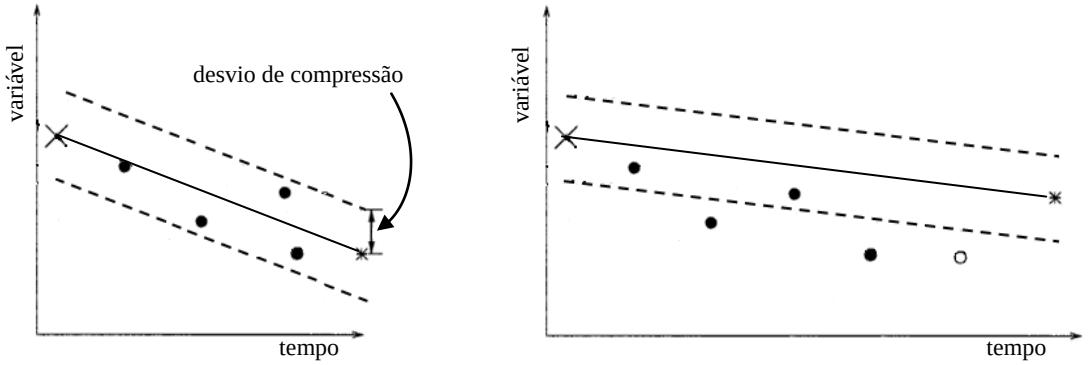


Figura 2.3: Sequência de duas amostragens no teste de compressão e funcionamento do algoritmo Swinging Door no PI: (X) valor gravado, (*) último amostrado, (o) último gravado e (●) valores não gravados.

A partir dos pontos enviados pelo teste de exceção, o teste de compressão forma um paralelogramo que se estende do último gravado nesta etapa ao último dado recebido. Para cada novo ponto que ingressa, o paralelogramo toma uma nova forma, e pontos intermediários vão se acumulando. Um evento é disparado quando, no mínimo, um dos pontos intermediários estiver fora da área de cobertura do paralelogramo. Nesse caso, grava-se o valor anterior ao atualmente amostrado, e a partir desse dado o ciclo recomeça. A largura do paralelogramo corresponde ao dobro do *desvio de compressão* (similar ao parâmetro *D*, da Figura 2.2), sendo este um valor a ser ajustado no algoritmo. É comum que outros dois parâmetros apareçam como critérios no teste de compressão: o *tempo mínimo a transcorrer para que possa haver uma compressão* e o *tempo máximo sem que haja uma compressão*. No primeiro caso, não haverá qualquer gravação enquanto não for transcorrido o tempo mínimo, enquanto que o tempo máximo é o maior intervalo que pode transcorrer sem que haja uma gravação.

2.1.3 Straight-Line-Interpolative-Method (SLIM)

Os métodos SLIM também se valem, a exemplo dos métodos anteriores, de uma banda de tolerância, diferindo, no entanto, na forma como os limites são construídos. A Figura 2.4 exibe algumas etapas do algoritmo SLIM1 (James, 1995) até que ocorra a gravação de um valor.

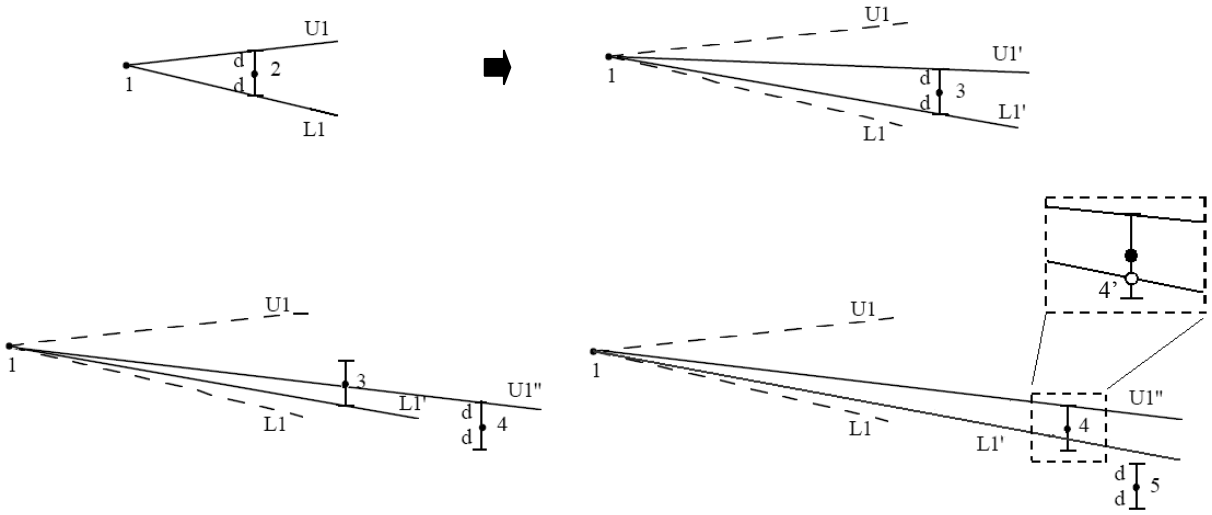


Figura 2.4: Etapas do algoritmo SLIM1: uma gravação ocorre quando a banda de tolerância do valor atual não for coberta pelos limites superior e inferior. (Adaptado de James, 1995.)

Dados os pontos 1 e 2, isto é (t_1, y_1) e (t_2, y_2), onde y é valor da variável medida e t o tempo a que esse valor corresponde, constroi-se um par de linhas com inclinações L1 e U1, na forma de um cone, onde d é o desvio especificado pelo usuário. Se o ponto amostrado a seguir (ponto 3) estiver incluído na banda de tolerância formada por L1 e U1, o ponto 2 é descartado, e as inclinações das linhas mudam para L1'e U1' formando um novo cone. Quando o ponto 4 é amostrado, apenas a inclinação da linha superior muda de U1' para U1", uma vez que só é permitido que as inclinações diminuam (em valor absoluto). Nesta etapa, o ponto 3 é descartado. Porém, se, a seguir, um valor na posição do ponto 5 for amostrado, um evento é disparado: as bandas de tolerância deste ponto não se sobrepõem à região compreendida pelas duas últimas linhas, L1' e U1". Neste caso, grava-se um pseudo-ponto (t_4, y_4'), que é a interseção entre a banda de tolerância do ponto 4 com a linha de inclinação L1'. A partir de y_4', o processo recomeça.

O algoritmo SLIM2 funciona similarmente ao SLIM1, porém armazenando o ponto anterior ao atualmente amostrado (ponto 4, da Figura 2.4) quando a banda de tolerância deste estiver fora da região cônica. Recomenda-se utilizar metade do valor do desvio d usado no SLIM1 (James, 1995). O SLIM3, por sua vez, grava o ponto anterior ao atualmente amostrado se este estiver fora da região de cobertura cônica (em vez de considerar sua banda de tolerância).

2.1.4 Piecewise Linear Online Trending (PLOT)

O algoritmo PLOT, proposto por Mah *et al.* (1995), armazena a informação utilizando critérios relacionados à mudança de tendência do sinal através de um ajuste linear dos dados

originais. Este algoritmo atua eficientemente na presença de *outliers*, adapta-se à variabilidade do processo e pode ser aplicado tanto no modo *online* como em dados em batelada. Os autores aplicaram o método a dados industriais e gerados computacionalmente, comparando seu desempenho com o do algoritmo SDT. Embora essa proposta tenha apresentado melhores resultados podendo adaptar-se à variabilidade e ao ruído do processo, ela exige uma carga computacional maior do que o método SDT (Singhal e Seborg, 2005).

No método PLOT assume-se que as medidas são discretas, em intervalos iguais de tempo, de acordo com o modelo

$$y_t = \mu_t + \varepsilon_t, \quad t = 1, 2, ... \tag{2.1}$$

onde μ_t é a média da variável na amostragem t e ε_t é o erro de medição no mesmo instante. Os erros de medição são admitidos independentes, seguindo uma distribuição normal. Considere que a tendência do sinal muda no tempo t, sendo $0 = t_0 < t_1 < t_2 < ...$. Durante o j-ésimo intervalo, ou seja, $t_{j-1} < t < t_j$, a tendência é aproximada por uma função linear dada por

$$\mu_t = \mu_{t_{j-1}} + \delta_{j-1} + \beta_j (t - t_{j-1}), \quad j = 1, 2, ... \tag{2.2}$$

onde $t_0 = 0$, $\delta_0 = 0$, δ_j representa uma "salto" ou uma mudança tipo degrau na média do processo no tempo t, e β_j é a inclinação da reta no intervalo j.

No algoritmo PLOT o objetivo é estimar uma função linear, conforme a Equação (2.2), que melhor representa o intervalo j, detectando mudanças de tendência, isto é, determinando o ponto t_j que marca o começo de um novo intervalo $j+1$. Também é desejável detectar *outliers* na medição da variável não permitindo que eles afetem a tendência real. O algoritmo utiliza a extrapolação da função linear estimada e a variância do sinal, com um determinado conjunto de pontos, para verificar se os dados posteriores seguirão a mesma tendência. Se a sequência de dados amostrados extrapolar critérios definidos como "intervalos de predição", isso significa uma mudança de tendência ou um *outlier*. Na ocorrência desses eventos procede-se a realização de testes a fim de fazer a distinção entre os dois elementos. A variância do processo também pode ser reestimada durante as etapas do algoritmo.

2.1.5 Critérios para avaliação da compressão

Em geral, dois critérios são utilizados para determinar a qualidade da compressão de um conjunto de dados: a *razão de compressão* (RC) e o *erro de reconstrução* (ER) (Singhal e Seborg, 2005). Nos métodos diretos ou lineares por partes, a razão de compressão é dada pela relação entre o número de pontos originais (sinal sem compressão) e o número de pontos gravados. Assumindo que o *tag* de tempo-data da medida requer a mesma quantidade de memória que o próprio valor da medida, então a razão de compressão é dada por (Watson *et al.*, 1998; Feng *et al.*, 2002):

$$RC = \frac{n^{o}\ medidas\ originais}{n^{o}\ de\ medidas\ gravadas \times 2} \tag{2.3}$$

Outros autores consideram apenas a razão entre o número de medidas originais e o número de medidas gravadas (James, 1995; Singhal e Seborg, 2005), desconsiderando o fator 2 no denominador, associado ao fato de se armazenar, além do valor da medida, o tempo em que ela ocorre (uma vez que o processo de compressão não gera dados igualmente espaçados). O erro de reconstrução geralmente é quantificado por uma média do erro quadrático entre o sinal original e o sinal reconstruído:

$$ER = \frac{\sum (y_i - \hat{y}_i)^2}{N} \tag{2.4}$$

onde y_i é o *i*-ésimo elemento do sinal original, que contém N medidas, e $\hat{y}_i$ é o *i*-ésimo elemento do sinal reconstruído. Outros critérios importantes têm sido propostos na literatura para avaliar os métodos de compressão, porém são revisados com maiores detalhes na seção 2.6.

2.2 Transformadas

Os métodos para compressão de dados baseados em transformadas fundamentam-se na representação do sinal no domínio da frequência, como nas transformadas de cosseno, de Laplace e Fourier, ou simultaneamente nos domínios da frequência e tempo, como nos desenvolvimentos mais recentes por transformadas *wavelets*. Essa última tem recebido atenção especial de vários grupos de pesquisa (Bakshi e Stephanopoulos, 1996; Watson *et al.*, 1995, Misra *et al.*, 2000 e 2001; Karim e Ismail, 2009) para aplicação a sinais de processos químicos.

Transformadas como as de Fourier e de cosseno fornecem um espectro de frequências, porém não informam onde essas frequências ocorrem (Watson *et al.*, 1998). Uma transformada *wavelet* pode ser representada simultaneamente em escala (ou frequência) e tempo – a chamada *análise multirresolução* (Mallat, 1989; Strang e Nguyen, 1996; Chun-Lin, 2010). A Figura 2.5 mostra um exemplo desse tipo de análise, fornecendo informações sobre a magnitude das frequências em diferentes intervalos de tempo. A transformada *wavelet* na forma discreta é dada pela aproximação:

$$f(t) = \sum_{j,k=-\infty}^{\infty} c_{j,k} \psi_{j,k}(t) \tag{2.5}$$

onde $\psi_{j,k}(t)$ são as *funções base*, ortonormais, provenientes da dilatação e da translação de uma função conhecida como *wavelet mãe*, $\psi(t)$. Os coeficientes $c_{j,k}$ são calculados através da equação:

$$c_{j,k} = 2^{j/2} \sum_{l=-\infty}^{\infty} f(l)\psi(2^j l - k) \tag{2.6}$$

em que $f(l)$ é o *l*-ésimo elemento da sequência de dados de entrada discretos, j é o fator de escala ou dilatação, e k o parâmetro de translação. A primeira e mais simples *wavelet* mãe é a *wavelet* de Haar (Strang e Nguyen, 1996), exibida na Figura 2.6a. Na mesma figura, em (b), é possível visualizar a *wavelet* mais comumente utilizada, a de Daubechies (Daubechies, 1988), entre outras existentes.

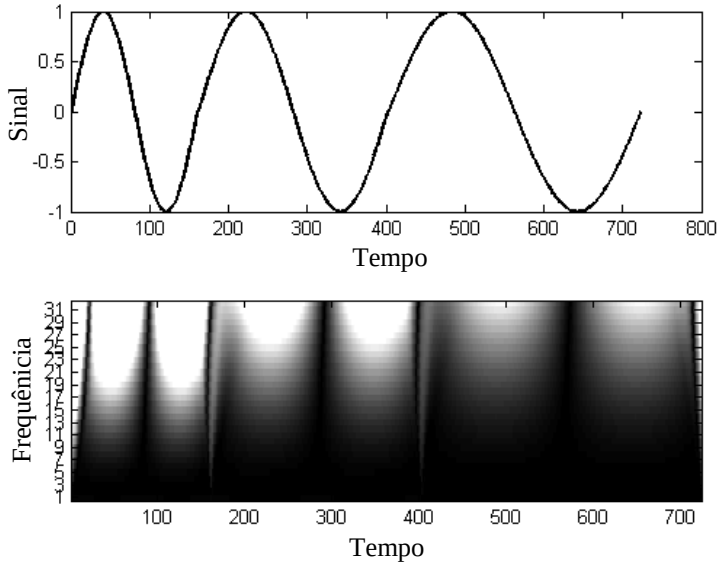


Figura 2.5: Análise multirresolução de uma transformada *wavelet*.

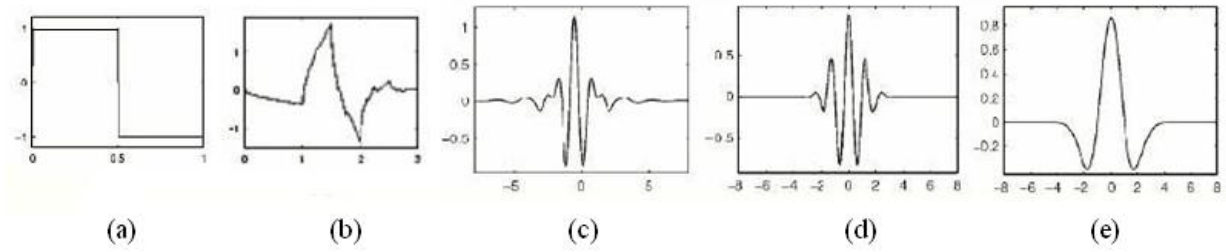


Figura 2.6: Exemplos de *wavelets* mãe: (a) Haar; (b) Daubechies; (c) Meyer; (d) Morlet e (e) Mexican Hat (Fonte: Panda and Nayak 2007).

A ideia por trás da análise multirresolução consiste na divisão do espaço de resolução J em subespaços representados por uma função de escala, $\phi_{J-1}(t)$, e a função *wavelet*, $\psi_{J-1}(t)$ (diz-se, por isso, que um subespaço $J-1$ está contido no subespaço J). Estas funções são ortogonais. Assim, um sinal em determinado nível de resolução é projetado em um subespaço de *aproximação* e outro de *detalhe*. A compressão de dados pela análise multirresolução é obtida através de restrições aos coeficientes dessas transformações a diferentes níveis de resolução. Componentes relativos a porções de alta frequência do sinal (coeficientes de detalhe) podem ser distinguidos daqueles relativos a tendências mais lentas (coeficientes de aproximação). Dessa forma, desconsiderando os coeficientes de menor magnitude, isto é, de detalhe, é possível gravar um número menor de informação ao mesmo

tempo em que se preserva a informação essencial, relativa às tendências mais lentas. Nesse caso, a razão de compressão pode ser dada por:

$$RC = \frac{n^o\ dados\ originais}{n^o\ coeficientes\ armazenados} \tag{2.7}$$

Inicialmente, as transformadas *wavelets* foram concebidas para sequências de dados em batelada (Watson *et al.*, 1995), baseando-se nas aplicações já existentes a outros tipos de sinais. Entretanto, algumas abordagens para aplicações *online* desse método foram propostas por Bakshi e Stephanopoulos (1996) e Misra *et al.* (2000 e 2001). Os primeiros autores propuseram um algoritmo baseado em pacotes de *wavelets*, realizando a seleção das melhores funções base juntamente com a eliminação dos coeficientes de menor valor para a compressão. A técnica é baseada na construção de "árvores duplas" de decomposição através de pacotes de *wavelets* variantes no tempo, utilizando critérios como a razão de compressão desejada e o erro médio quadrático aceitável da aproximação. Já as técnicas descritas por Misra *et al.* (2000 e 2001) também utilizam uma abordagem de construção sequencial das transformadas em forma de "árvores", porém neste caso, chamadas "árvores multirresolução". Conforme os dados são amostrados, a árvore é gerada, os coeficientes de detalhe são avaliados e comparados a um *threshold* (ou limite) especificado. Quando o valor desse coeficiente é menor que o *threshold*, ele é ignorado, enquanto o coeficiente de aproximação corresponde ao mesmo nível de resolução é adicionado à árvore. Caso o coeficiente de detalhe ultrapassar o valor do limite estabelecido, o crescimento da árvore é interrompido, seus parâmetros são reiniciados e ela é enviada ao histórico. Os autores ainda reportam que a RC da abordagem *online* é superior àquela no modo de dados em batelada, sendo que esta diferença aumenta quanto maior o valor do *threshold* especificado.

2.3 Quantização

A quantização é largamente utilizada em processamento de sinais digitais, encontrando ampla aplicação na compressão de dados, a partir da generalização da quantização escalar para uma abordagem vetorial (Gersho e Gray, 1992). O funcionamento do algoritmo consiste basicamente no arredondamento dos valores (não exatamente ao valor inteiro mais próximo), em um determinado nível de acuracidade. Um quantizador possui tamanho N e dimensão k, e mapeia um vetor de entrada de mesma dimensão em um espaço euclidiano R^k dentro de um conjunto finitos de N saídas (ou pontos de reprodução).

A Figura 2.7 ilustra os casos uni e bidimensional. Os números, na reta, ou os pares de números, no espaço de duas dimensões, são aproximados pelos valores representados pelo símbolo $*$, que serão os pontos a serem reproduzidos na descompressão. Os valores na parte superior da reta real são os códigos de saída. Os valores a que se refere o símbolo $*$ são denominados *codevectors*, e um conjunto deles é chamado de *codebook*. As regiões (ou células) formadas pela divisão do espaço são chamadas *regiões de codificação* (ou, mantendo a terminologia em inglês, *enconding regions*) e o conjunto delas é denominado *partição* do espaço (Gersho e Gray, 1992).

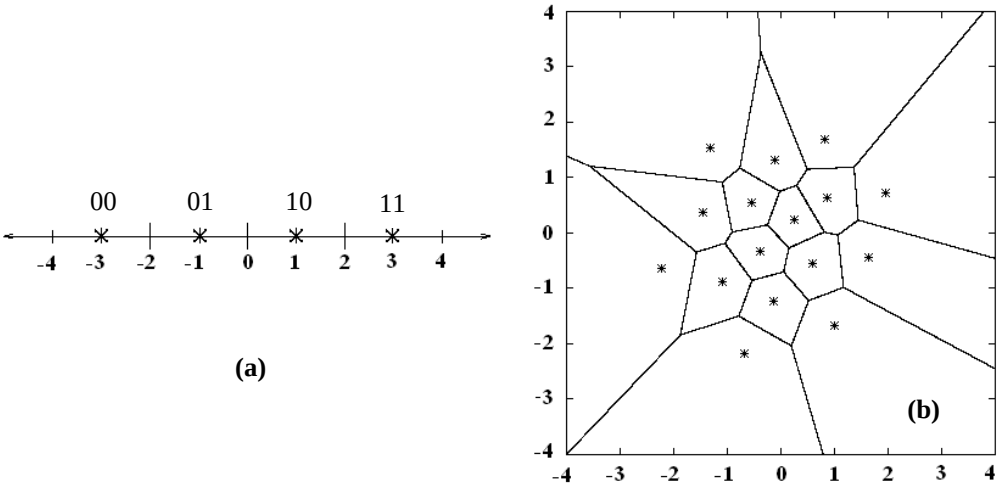


Figura 2.7: Representação dos quantizadores no espaço R^k para (a) $k = 1$ e (b) $k = 2$. Os símbolos * representam os *codevectors*. (Adapatado de: <http://www.data-compression.com/vq.html>)

A *resolução* ou *taxa* de um vetor quantizador é obtida por $r = (\log_2 N)/k$ (para codificação binária), e mede o número de *bits* por componente do vetor usado para representar o sinal de entrada. Esta variável fornece uma indicação da acuracidade ou precisão alcançável com um quantizador se o *codebook* for bem projetado. Diferentemente dos métodos de compressão diretos, a razão de compressão na quantização usualmente é dada em unidades de memória. Por exemplo, um computador de 64 *bits* requer $k \times 64$ *bits* para representar cada grupo de dados a serem quantizados em um *codebook* de dimensão k (Watson *et al.*, 1998). Então, a razão de compressão é:

$$RC = \frac{64 \times k}{\log_2 N} \tag{2.8}$$

O desempenho da compressão depende fortemente do projeto do quantizador. Esse projeto consiste em encontrar um *codebook* e uma partição do espaço que resultem na menor distorção média. Assume-se, primeiramente, que há uma sequência de *dados de treinamento*, consistindo de M vetores-fonte (ou números-fonte, no caso da quantização escalar, onde $k = 1$), e uma sequência de treinamento $T = \{x_1, x_2, ..., x_M\}$, que pode ser obtida de um extenso banco de dados. O número de *codevectors* é dado por N, e o conjunto $C = \{c_1, c_2, ..., c_N\}$ representa o *codebook*. S_n é a região de codificação associada ao *codevector* c_n, sendo $P = \{S_1, S_2, ..., S_N\}$ a partição do espaço. A distorção média (DM), então, é dada por:

$$DM = \frac{1}{M} \sum_{m=1}^{M} (x_m - Q(x_m))^2 \tag{2.9}$$

onde $Q(x_M)$ denota a aproximação do dado fonte x_m que está na região S_n pelo *codevector* c_n. De forma sucinta, o projeto do quantizador é dado pelo seguinte problema: dados T e N, encontrar C e P tal que DM seja mínimo. Algumas metodologias propostas na literatura são apresentadas de forma mais detalhada na seção 2.5.

2.4 Outras técnicas para compressão de dados

Além dos tradicionais métodos classificados nas seções anteriores, a literatura apresenta ainda outros algoritmos para a compressão de dados de processos. É o caso dos métodos propostos por Vedam *et al.* (1998), utilizando *B-splines* para compressão, com aplicações para monitoramento e diagnóstico de processos; e por Mo *et al.* (1998), baseado em algoritmo de clusterização (*k-means clustering*).

Assim como nos métodos por transformadas, a compressão de dados na metodologia proposta por Vedam *et. al.* (1998) é feita em uma abordagem multinível. Neste caso, utiliza-se uma aproximação L_2 de *B-splines diádicas*, dada por:

$$N(x) = \begin{cases} 1+x, & -1 < x < 0 \\ 1-x, & 0 < x < 1 \\ 0, & caso\,contrário \end{cases} \tag{2.10}$$

Semelhante às aproximações por transformadas *wavelets*, as *B-splines* são transladadas e dilatadas a fim de se obter bases fora do intervalo [-1,1]:

$$N_{j,k} = N(2^k x - j), \quad j \in Z, \quad k = 0,1,... \tag{2.11}$$

Dada uma janela de dados com $2^m + 1$ amostras sobre um intervalo finito [0,1], a aproximação por *B-splines* diádicas gera projeções dos dados em polinômios lineares por partes (P_k) a diferentes níveis, sendo que o nível k é obtido a partir do nível $k+1$.

A aproximação dos dados na janela usando as *B-splines*, então, é dada por:

$$P_m = \sum_{k=0}^{m} \sum_{j=0}^{2^k} d_{j,k} N_{j,k} \tag{2.12}$$

Essa equação é chamada *equação de decomposição*, uma vez que os dados são decompostos em coeficientes de N_k. O objetivo, assim, é obter P_m a partir dos coeficientes $d_{j,k}$ negligenciando aqueles que forem inferiores a um determinado *threshold*, o qual é escolhido baseado no desvio padrão do ruído de medição ou como uma porcentagem do *span* do sensor.

Através de técnicas para interpretação qualitativa de dados de processo, Mo *et al.* (1998) propuseram um método para compressão, baseado em uma versão adaptativa em tempo real do algoritmo *k-means clustering*. Esse algoritmo tem como principal objetivo detectar mudanças de tendência nos dados, tendo como critério a distância euclidiana entre os dados reais e o chamado centro do *cluster*. Na versão proposta os centros dos *clusters* são atualizados usando o conceito de *taxa de aprendizagem decrescente*, sendo que o valor do *cluster* inicial corresponde ao primeiro valor amostrado pelo sensor. Quando um novo dado é amostrado, o centro do *cluster* pode ser atualizado ou readaptado. Para isso, utiliza-se a distância euclidiana entre o centro do *cluster* $c_{i,t-1}$ e o novo valor obtido $x_{i,t}$. Para o caso

multivariável, essas distâncias são padronizadas pelos seus respectivos desvios padrão σ_i, e são dadas por $d = \sqrt{\sum((x_{i,t} - c_{i,t-1})/\sigma_i)^2}$. Quando d ultrapassar um determinado limite de gravação, então o valor anterior ou o valor central do *cluster* é armazenado. Esta técnica foi comparada, no mesmo trabalho, com outros algoritmos de compressão diretos (BC, BS, BC/BS e SDT) e mostrou desempenho superior em termos da RC e do ER.

2.5 Metodologias para ajuste dos algoritmos de compressão

2.5.1 Métodos Diretos ou Lineares por Partes

Apesar de os métodos diretos estarem bastante difundidos na indústria, principalmente por sua simplicidade e facilidade de implementação, os parâmetros que envolvem o seu bom funcionamento carecem de sistemáticas de ajuste. Embora a literatura tenha apresentado, nas últimas décadas, algum desenvolvimento no que diz respeito a novas técnicas para a compressão de dados de processos industriais, propostas para a sintonia de parâmetros dos algoritmos já existentes ainda são escassas.

Pettersson e Gutman (2004) propuseram algoritmos para a sintonia automática do limite de gravação do algoritmo BC/BS em abordagens *offline* e *online*. Foi proposta a minimização de um critério para a escolha deste limite com base na relação entre o número de dados gravados e o erro de reconstrução dos pontos gravados interpolados linearmente, cuja expressão é:

$$V_N(h) = (1-\alpha) r_N + \alpha \frac{R_N(e)}{\sigma_N^2(y)} \tag{2.13}$$

em que h é o valor do limite de gravação, $r_N = N_r / N$ onde N_r denota o número de pares $[t, y]$ gravados, R_N o erro médio quadrático (Equação 2.4) e σ_N^2 a variância dos dados. $\alpha \in [0,1]$ está associado ao compromisso entre a redução de dados gravados e a fidelidade na reprodução do sinal original. Os autores, utilizando propriedades estatísticas, provaram analiticamente que, para um determinado conjunto de pontos e valores não muito altos de h, V_N possui um mínimo global. No modo *online*, o algoritmo calcula h proporcionalmente ao valor de σ, sendo que esse valor e o fator de proporcionalidade são atualizados de forma recursiva.

O algoritmo SDT proposto por Bristol (1990) foi aperfeiçoado por Feng *et al.* (2002) de forma que se detecte e elimine *outliers*, além de adaptar o valor de D (ver Figura 2.2) ao longo do processo de compressão. A exemplo do método PLOT, a metodologia proposta investiga a ocorrência de *outliers* ou mudanças de tendência, porém, neste caso, avaliando a soma dos ângulos internos das portas superior e inferior do SDT. O valor de D é ajustado com base em um parâmetro F_{adj} $(0 < F_{adj} < 1)$, substituindo-o por $D * F_{adj}$ ou D / F_{adj}, diminuindo ou aumentando a razão de compressão, respectivamente. No entanto, para evitar extrapolações, D fica confinado em uma determinada faixa, que é dependente de resultados empíricos. A avaliação do desempenho desse algoritmo foi testada em alguns sinais artificiais,

e os autores obtiveram resultados superiores, tanto na RC quanto no ER, comparado ao SDT tradicional.

Em 2006, Alsmeyer propôs uma solução envolvendo as vantagens das transformadas *wavelets* e as possibilidades técnicas dos métodos diretos dos atuais PIMS para sintonizar, automaticamente, os parâmetros de compressão. Esta proposta é baseada na análise do sinal original no domínio das *wavelets* para ajustar os parâmetros de alguns métodos lineares (SDT e BC/BS), em um algoritmo externo chamado ALANDA. Primeiramente o sinal original é classificado como "resolução-dominante" ou "ruído-dominante", de acordo com critérios determinados. O autor propõe algumas regras heurísticas para ajustar os limites de gravação dos métodos lineares. Nos dados resolução-dominantes, esses limites são valores proporcionais a δ, em que δ é a menor resolução visível, e nos casos ruído-dominantes, proporcionais ao desvio padrão do ruído, cuja estimativa é obtida pelas transformadas *wavelets*.

2.5.2 Quantização

Visto o problema de projeto de um quantizador na seção 2.3, pode-se estabelecer critérios de otimalidade. Se C e P são uma solução do problema de otimização exposto na seção 2.3, então ela satisfará os seguintes critérios:

- Condição do vizinho mais próximo:

$$S_n = \{x : \|x - c_n\|^2 \leq \|x - c_{n'}\|^2 \ \forall n' = 1,2,..., N\} \tag{2.14}$$

Esta condição diz que a região de codificação S_n deve consistir de todos os vetores que estão mais próximos de $\boldsymbol{c_n}$ do que de qualquer dos demais *codevectors*. Para aqueles vetores que caírem na fronteira (linhas divisórias da Figura 2.7b), a escolha arbitrária por qualquer das regiões vizinhas é aceita.

- Condição do centroide:

$$c_n = \frac{\sum_{x_m \in S_n} x_m}{\sum_{x_m \in S_n} 1} \tag{2.15}$$

Por esta condição conclui-se que o *codevector* c_n deve ser a média de todos os vetores de treinamento que estão na região de codificação S_n. Deve-se garantir, contudo, que pelo menos 1 vetor de treinamento pertença a cada região de codificação, de forma que o denominador da equação acima nunca seja nulo.

A técnica mais utilizada para projeto de quantizadores é o algoritmo LBG (Linde-Buzo-Gray) (Linde *et al.*, 1980). Trata-se de um algoritmo iterativo que resolve ambos os critérios de otimalidade apresentados. O algoritmo LBG requer um *codebook* inicial $C^{(0)}$, que é obtido por um método de divisão (*splitting*). Nesse método, um *codevector* inicial é definido como a média do conjunto completo da sequência de treinamento. Esse *codevector* é,

então, dividido em dois vetores, que formam o *codebook* inicial do algoritmo iterativo. Os dois *codevectors*, por sua vez, são divididos em quatro, e o procedimento é repetido até obter-se o número desejado desses vetores.

Uma série de outros algoritmos é proposta na literatura (Equitz, 1989; Yair *et al.*, 1992; Zeger *et al.*, 1992; Lee *et al.*, 1997; Zheng *et al.*, 1997) como alternativa para o algoritmo LBG, porém não serão detalhados, pois visam aplicações além daquelas envolvendo sinais de processos da indústria química.

2.6 Avaliações e comparações entre métodos

Recentemente, alguns autores revisaram os principais métodos de compressão de dados para processos químicos a fim de comparar seus desempenhos e avaliar o impacto dessas técnicas em aspectos importantes para caracterizar e analisar dados de planta.

Watson *et al.* (1998) avaliaram os métodos lineares, de quantização vetorial e transformadas (Fourier, cosseno e *wavelets*) variando os parâmetros de cada algoritmo, e também compararam as técnicas entre si, em dois conjuntos de dados reais de planta. Para os métodos lineares (BC, BS, BC/BS e SDT), foram variados os valores dos limites de gravação, comparando a relação RC *vs* EGR, em que EGR é o *Erro Global Relativo* (%), dado por:

$$\%\,EGR = 100 \times \frac{\sum (f_i - \hat{f}_i)^2}{\sum (f_i)^2} \tag{2.16}$$

onde f_i é o *i*-ésimo elemento do sinal original e $\hat{f}_i$ é o *i*-ésimo elemento do sinal reconstruído. Para o primeiro conjunto de dados, os métodos apresentaram desempenho comparável, enquanto que, para o segundo conjunto, o SDT mostrou desempenho superior aos demais.

Os dados de processos também foram usados para avaliar a técnica de quantização, variando-se a dimensão e o tamanho dos *codebooks*. Para o primeiro conjunto, todos os dados foram usados para o projeto dos quantizadores, enquanto para o segundo, alguns subconjuntos foram selecionados. Os resultados mostraram que o desempenho da compressão é maior quanto maior o número de elementos (tamanho) e maior a dimensão do *codebook*, a um custo maior de tempo computacional. Para as técnicas envolvendo transformadas, foi variado o número de coeficientes descartados. As *wavelets* foram obtidas usando funções ortonormais de Daubechies de quinta ordem, sendo que este tipo de transformada e as transformadas de cosseno forneceram resultados similares. As melhores técnicas de cada família de algoritmos (SDT, dos métodos lineares, as *wavelets* e as transformadas de cosseno) foram também avaliadas em análises comparativas entre si. As *wavelets* apresentaram os melhores resultados quando se considera a relação RC *vs* EGR, sendo em alguns casos, inferiores ao método de quantização vetorial. No entanto, para estes casos, o projeto dos quantizadores foi feito utilizando o conjunto completo dos dados de teste. Na prática, este conjunto de dados costuma ser maior e usualmente não são usados por inteiro no treinamento do *codebook*. Os autores ainda ressaltam que *codebooks* projetados para uma determinada variável nem sempre irá

comprimir dados de outra variável eficientemente, além de enfatizarem a elevada demanda computacional exigida pelo método. Apesar de as transformadas *wavelets* e de cosseno terem apresentado resultados semelhantes nos testes anteriores, algumas diferenças entre elas evidenciam-se quando o sinal possui mudanças tipo degrau ou variações tipo impulso. Para estes casos, as *wavelets* apresentaram desempenho superior. Estas transformadas também apresentaram melhores resultados, comparadas aos demais algoritmos, quando avalia-se o *Erro Máximo Relativo* (EMR), em função da RC, dado por:

$$\%EMR = 100 \times \frac{\max_i \left(\left| f_i - \hat{f}_i \right| \right)}{\max_i \left(\left| f_i \right| \right)} \tag{2.17}$$

Um estudo sobre o impacto da compressão em análise de processos baseada em dados operacionais foi publicado por Thornhill *et al.* (2004). Aplicações como *benchmarking* de controladores de variância mínima, detecção de falhas, reconciliação de dados e desenvolvimento de sensores por inferência foram visadas, com foco em dados de processes industriais e, portanto, na compressão pelos métodos diretos ou lineares. Para atingir os objetivos do estudo, foi implementado o SDT, conforme descrito por Bristol (1990). Três exemplos distintos de sinais reais foram utilizados: um conjunto de dados que apresenta oscilação permanente; outro que possui variações bruscas de tendência, proveniente de uma malha com agarramento de válvula; e o último apenas corrompido por ruído randômico. O desvio de compressão foi ajustado de forma a se obter, para todos os casos, uma RC igual a 10. Entre as medidas de desempenho consideradas estão algumas propriedades estatísticas – diferença percentual entre médias, razão entre variâncias do sinal original e reconstruído, e entre erros de reconstrução –, medidas de não-linearidades e índice de Harris. Verificou-se que todas as propriedades avaliadas foram alteradas com a compressão dos dados, sendo que algumas aplicações como reconciliação de dados, balanços de massa e auditoria de controladores são mais prejudicadas com a baixa fidelidade dos dados comprimidos. Os autores avaliaram também as métricas estudadas variando o valor da RC. Destacam que o sinal com predominância de ruído é mais influenciado pela compressão com valores baixos de RC que os demais sinais, e alertam para o cuidado com o uso de RC's menores ou iguais a 3, para análise de processos baseada em dados.

Outro estudo analítico foi realizado por Singhal e Seborg (2005) a fim de verificar o efeito da compressão de dados na correspondência de padrões de dados históricos. Foram utilizados para esta análise o método BC, um algoritmo de compressão por média dos dados, o algoritmo usado no PI System™ e transformadas *wavelets*. As técnicas para correspondência de padrões envolveram métodos de análise de componentes principais (PCA), com o chamado *Fator de Similaridade por PCA*, e um segundo critério denominado *Fator de Similaridade por Distância*. Algumas métricas para quantificar o desempenho da correspondência de padrões foram definidas, e as análises foram feitas em simulações de um processo consistindo de um reator CSTR não-isotérmico. O algoritmo de compressão do PI foi o que apresentou o menor valor no ER, porém não se obteve bons resultados quando da utilização dos dados comprimidos na correspondência de padrões. As *wavelets*, por sua vez, forneceram os melhores resultados nessa última aplicação, além de apresentaram baixos valores no ER (ainda que maiores que os do PI).

2.7 Avaliação geral dos métodos de compressão

Nesta seção pretende-se abordar de forma geral todas as técnicas para compressão de dados de processos revisadas neste capítulo, através de uma avaliação subjetiva quanto a alguns critérios considerados. , A Tabela 2.1 exibe esta análise, onde os métodos são classificados em uma escala de 1 a 5, sendo que quanto maior o valor melhor o seu desempenho. Por exemplo, para o erro de reconstrução, notas altas significam um valor baixo deste atributo; para a demanda computacional, valores altos representam baixa exigência em termos de processamento. Por razões práticas, os métodos Boxcar e Backslope não foram avaliados independentemente, mas sim a combinação de ambos (BC/BS).

Tabela 2.1: Avaliação geral dos métodos de compressão.

	Diretos ou Lineares por partes				**Quantização**	**Transformadas**
	BC/BS	SDT	SLIM	PLOT		
Razão de compressão	3	3	4	3	5	4
Erro de reconstrução	2	3	3	4	4	5
Demanda computacional	5	5	5	5	1	4
Facilidade de ajuste	3	3	2	3	2	3
Facilidade de implementação	5	5	5	4	3	4
Média	3,6	3,8	3,8	3,8	3,0	4,0

De um modo geral, pode-se dizer que as técnicas que abordam as transformadas, principalmente as *wavelets*, como método de compressão são as que possuem melhor desempenho. No entanto, exigem um maior conhecimento das teorias sobre tais ferramentas, e seu ajuste não é suficientemente intuitivo. Em geral são aplicadas a dados em batelada (janelas), mas há a possibilidade de serem implementadas de forma *online*.

Os métodos lineares são simples e fáceis de implementar, podem ser aplicados *online*, porém são técnica menos sofisticadas, que apresentam maiores erros de reconstrução. A quantização, por sua vez, não apresenta muitas aplicações práticas direcionadas a dados de processos, pois não é vantajosa o ponto de vista de implementação e ajuste.

Um critério a ser considerado, mas que não é apreciado nos trabalhos da literatura é o erro na derivada do sinal. Estimar a derivada de sinais é importante quando da utilização dos dados de processos, principalmente no que diz respeito à modelagem de sistemas, impactando

em aplicações como identificação, procedimentos de auditoria e inferência de variáveis não medidas. Outro aspecto relevante também na avaliação dos métodos de compressão é o que diz respeito ao seu grau de propriedade intelectual. O método SDT teve registro em patente (Bristol, 1985), no entanto é considerado de domínio público, pois seu prazo de proteção já expirou. Ainda, a técnica de compressão via *wavelets online* foi registrada por Kumar *et al.* (2001), tendo, pelo seu período de vigência, proteção intelectual garantida. Há outros registros de patentes envolvendo métodos de compressão por transformadas *wavelets*, porém aplicados a tipos diferentes de sinais, como imagem e acústica.

Nos capítulos seguintes, serão apresentadas as técnicas propostas para a compressão de dados aplicadas a processos da indústria química, com ideias que se aproximam em simplicidade aos métodos lineares, porém com ajuste mais intuitivo, e resultados na reconstrução com maior grau de fidelidade, próximos às características das *wavelets*, com uma exigência computacional satisfatória.

Referências

Alsmeyer, F. (2006) Automatic adjustment of data compression in process information management systems. *16th European Symposium on Computer Aided Process Engineering and 9th International Symposium on Process Systems Engineering*, 21**,** 1533-1538.

Aspen® Technology Inc. Analysis of Data Storage Technologies for the Management of Real-Time Process Manufacturing Data. White Paper for InfoPlus® 2.1. Campbell, CA. Disponível em <http://www.cpdee.ufmg.br/~seixas/Especializacao/Download/DownloadFiles/White_Paper_for_IP_21.pdf>

Bakshi, B. R. & G. Stephanopoulos (1996) Compression of chemical process data by functional approximation and feature extraction. *Aiche Journal*, 42**,** 477-492.

Bristol EH. The Foxboro Company, MA. Data compression for display and storage. Intern'l Class H03K 005/00. U.S. 789531. 21 out. 1985. United States Patent.

Bristol, E. H. (1990). Swinging Door Trending: Adaptative Trend Recording? In *Advances in Instrumentation and Control*, 749-754. Instrument Society of America: Research Triangle Park, NC.

Chun-Lin, L. (2010) A Tutorial of the Wavelet Transform. Disponível em: <http://disp.ee.ntu.edu.tw/tutorial/>

Daubechies, I. (1988) ORTHONORMAL BASES OF COMPACTLY SUPPORTED WAVELETS. *Communications on Pure and Applied Mathematics*, 41**,** 909-996.

Equitz, W. H. (1989) A NEW VECTOR QUANTIZATION CLUSTERING-ALGORITHM. *Ieee Transactions on Acoustics Speech and Signal Processing*, 37**,** 1568-1575.

Feng, X. D., C. L. Cheng, C. L. Liu, H. E. Shao & T. U. Tu (2002) An improved process data compression algorithm. *Proceedings of the 4th World Congress on Intelligent Control and Automation, Vols 1-4*, 2190-2193.

Gersho, A. & R. M. Gray. (1992). *Vector Quantization and Signal Compression*. Norwell, MA: Springer.

Hale, J. C. & H. L. Sellars (1981) HISTORICAL DATA RECORDING FOR PROCESS COMPUTERS. *Chemical Engineering Progress,* 77**,** 38-43.

James, P. A. (1995). DATA COMPRESSION FOR PROCESS HISTORIANS. Richmond, CA: Chevron Research and Technology Company.

Karim, S. S. A. & M. T. Ismail (2009) Compression Of Chemical Signal Using Wavelet Transform. *European Journal of Scientific Research,* 36**,** 513-520.

Kortman, C. M. (1967) REDUNDANCY REDUCTION-A PRACTICAL METHOD OF DATA COMPRESSION. *Proceedings of the Institute of Electrical and Electronics Engineers,* 55**,** 253-&.

Kumar S, Misra M, Qin JS, Blevins TL, Seeman RC. Fisher Rosemont Systems Inc., TX. Recursive on-line wavelet data compression technique for use in data storage ans communications. Int. Cl.[7] G06K 9/36. U.S. 09/105,950. 26 jun. 1998, 10 abr. 2001.

Lee, D., S. Baek & K. Sung (1997) Modified K-means algorithm for vector quantizer design. *Ieee Signal Processing Letters,* 4**,** 2-4.

Linde, Y., A. Buzo & R. M. Gray (1980) ALGORITHM FOR VECTOR QUANTIZER DESIGN. *Ieee Transactions on Communications,* 28**,** 84-95.

Mah, R. S. H., A. C. Tamhane, S. H. Tung & A. N. Patel (1995) PROCESS TRENDING WITH PIECEWISE-LINEAR SMOOTHING. *Computers & Chemical Engineering,* 19**,** 129-137.

Mallat, S. G. (1989) A THEORY FOR MULTIRESOLUTION SIGNAL DECOMPOSITION - THE WAVELET REPRESENTATION. *Ieee Transactions on Pattern Analysis and Machine Intelligence,* 11**,** 674-693.

Misra, M., S. Kumar, S. J. Qin & D. Seemann (2001) Error based criterion for on-line wavelet data compression. *Journal of Process Control,* 11**,** 717-731.

Misra, M., S. J. Qin, S. Kumar & D. Seemann (2000) On-line data compression and error analysis using wavelet technology. *Aiche Journal,* 46**,** 119-132.

Mo, K. J., S. Eo, D. Shin & E. S. Yoon (1998) Qualitative interpretation and compression of process data using clustering method. *Computers & Chemical Engineering,* 22**,** S555-S562.

OSIsoft® Inc. (2003). PI Server Reference Guide.

Panda, D. R. & C. Nayak. (2007). EYE DETECTION USING WAVELETS AND ANN. Department of Electronics & Instrumentation Engineering - National Institute of Technology, Rourkela.

Pettersson, J. & P. O. Gutman (2004) Automatic tuning of the window size in the Box Car Back Slope data compression algorithm. *Journal of Process Control,* 14**,** 431-439.

Singhal, A. & D. E. Seborg (2005) Effect of data compression on pattern matching in historical data. *Industrial & Engineering Chemistry Research,* 44**,** 3203-3212.

Strang, G. & T. Nguyen. 1996. *Wavelets and Filter Banks*. Wellesley, MA: Wellesley-Cambridge Press.

Thornhill, N. F., M. Choudhury & S. L. Shah (2004) The impact of compression on data-driven process analyses. *Journal of Process Control,* 14**,** 389-398.

Vedam, H., V. Venkatasubramanian & M. Bhalodia (1998) A B-spline based method for data compression, process monitoring and diagnosis. *Computers & Chemical Engineering,* 22**,** S827-S830.

Watson, M. J., A. Liakipoulos, D. Brzakovic, C. Georgakis & C. Amer Automat Control (1995) Wavelet techniques in the compression of process data. *Proceedings of the 1995 American Control Conference, Vols 1-6***,** 1265-1269.

Watson, M. J., A. Liakopoulos, D. Brzakovic & C. Georgakis (1998) A practical assessment of process data compression techniques. *Industrial & Engineering Chemistry Research,* 37**,** 267-274.

Yair, E., K. Zeger & A. Gersho (1992) COMPETITIVE LEARNING AND SOFT COMPETITION FOR VECTOR QUANTIZER DESIGN. *Ieee Transactions on Signal Processing,* 40**,** 294-309.

Zeger, K., J. Vaisey & A. Gersho (1992) GLOBALLY OPTIMAL VECTOR QUANTIZER DESIGN BY STOCHASTIC RELAXATION. *Ieee Transactions on Signal Processing,* 40**,** 310-322.

Zheng, X. W., B. A. Julstrom, W. D. Cheng & Ieee (1997) Design of vector quantization codebooks using a genetic algorithm. *Proceedings of 1997 Ieee International Conference on Evolutionary Computation (Icec '97)***,** 525-529.

Capítulo 3

Metodologia

As técnicas propostas neste trabalho consistem de dois métodos, denominados *Refinado* e *Evolutivo*, que serão apresentados neste capítulo. Para ambos os métodos há uma premissa e duas etapas em comum. A premissa é de que a compressão dos dados é realizada a partir de um conjunto das medidas originais, isto é, deve haver uma *janela* prévia contendo um determinado número de pontos de dados. Não há um critério específico para o tamanho da janela, mas ela pode ser, por exemplo, composta por turnos de operação de uma determinada variável. Essa premissa é oportuna, pois as janelas de dados também são usadas em auditoria de desempenho de malhas.

As duas etapas comuns aos métodos são a prévia *suavização* ou *regularização* do sinal, e a reconstrução dos dados gravados utilizando *splines*. As características dessas ferramentas são descritas na seção 3.2, e os detalhes de cada um dos métodos são apresentados nas seções 3.3 e 3.4. Todas as implementações computacionais foram realizadas em ambiente MATLAB®.

3.1 Etapas gerais dos métodos propostos

Os procedimentos dos algoritmos propostos podem ser representados pelo diagrama da Figura 3.1. Após a aquisição dos dados em janelas de tamanhos especificados, o sinal ruidoso é submetido à etapa de suavização. Os dados suavizados passam aos algoritmos de compressão, nos quais é feita seleção dos pontos a serem gravados, de acordo com a metodologia desejada. O bloco que indica a reconstrução do sinal tanto integra os algoritmos de compressão (conforme descrito nas seções posteriores) como possui a função de descompressão para apresentação final dos dados.

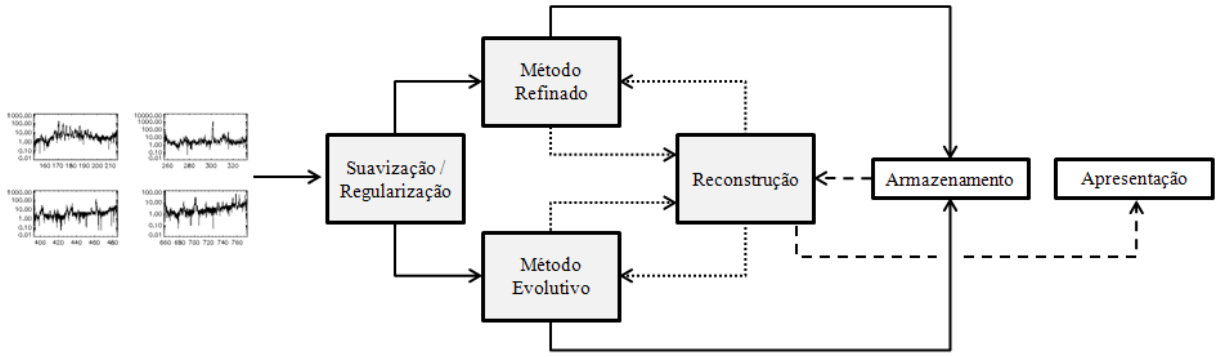


Figura 3.1: Representação das etapas dos algoritmos para compressão propostos.

3.2 Ferramentas utilizadas

3.2.1 Suavização/Regularização

A suavização ou regularização encontra larga aplicação em processamento de sinais, nas áreas de tratamento de imagens, acústica, e engenharia (Guan *et al.*, 1997, Yoon e Nelson 2000; Stickel, 2010). Mejia *et al.* (2010) mostraram que é possível obter uma boa estimativa das derivadas de sinais de processo corrompidos por ruído com diferentes métodos de suavização. Foram utilizados diversos tipos de sinais com características distintas como retas, rampas e oscilações, e testaram-se diferentes técnicas de suavização com parâmetros fixos e autoajustados: filtros *Butterworth*, *B-splines*, polinômios de Savistky-Golay e *Wavelet Shrinkage Denoising*. Os autores concluíram que os métodos com autoajuste se mostraram mais eficazes na estimativa das derivadas do que aqueles com parâmetros fixos. Também relataram que para sinais com dinâmicas lentas ou mudanças suaves os métodos polinomiais (*B-splines* e polinômios de Savitsky-Golay) são mais apropriados, enquanto que para sinais oscilatórios e não estacionários os melhores resultados são obtidos com suavização por filtros *Butterworth* e *Wavelet Shrinkage*.

As técnicas apresentadas por Mejia *et al.* (2010) foram satisfatoriamente aplicadas a alguns sinais provenientes de uma refinaria de petróleo nacional, e duas delas foram selecionadas para serem abordadas nos algoritmos propostos: filtros *Butterworth* e *B-splines*, ambos com parâmetros autoajustados. Essa escolha foi baseada considerando-se o esforço computacional de cada método e os erros obtidos entre o sinal original e o sinal suavizado, bem como os erros entre suas derivadas de primeira ordem. Também, objetiva-se avaliar se há impacto significativo na escolha dos métodos de regularização para a compressão de dados nos algoritmos propostos.

Os filtros passa-baixa *Butterworth* são filtros do tipo IIR (*Infinite Impulse Response*), e foram aplicados em ambas as direções, direta e reversa (fase zero). No algoritmo adaptativo, definem-se primeiramente a ordem dos filtros e as frequências de corte que serão testadas. O filtro escolhido será aquele que apresentar o menor valor do critério GCV (*Generalized Crossed Validation*), proposto por Wahba (1976), que é dado por

$$GCV = \frac{RSS}{(1-w_c)^2} \tag{3.1}$$

onde RSS é a soma quadrática dos resíduos (entre sinais original e regularizado), e w_c é a frequência de corte considerada.

A suavização adaptativa utilizando *B-splines* aproxima o sinal ruidoso por uma curva suave no sentido de mínimos quadrados, variando-se o número de nós em intervalos constantes, selecionando a aproximação que possui o mínimo valor do critério de Akaike. Inicialmente, o vetor de dados é expandido espelhando-se a janela até o ponto, no tempo, que contém o maior número de divisores das últimas *n*/5 medias, onde *n* é o número de dados originais na janela. Os divisores inteiros deste ponto são o número de nós na aproximação por *B-splines* a serem testados. O critério da informação de Akaike (Akaike, 1974) foi concebido originalmente para modelos paramétricos como uma medida estatística no ajuste de séries temporais. Para o caso em que o erro no sinal segue uma distribuição gaussiana, com variância desconhecida, o critério é dado por

$$AIC = 2K + n\ln\left(\frac{RSS}{n}\right) \tag{3.2}$$

onde K é o número de parâmetros (ordem da *B-spline* + número de nós) e n é o tamanho do vetor de dados.

3.2.2 Spline cúbica de Hermite

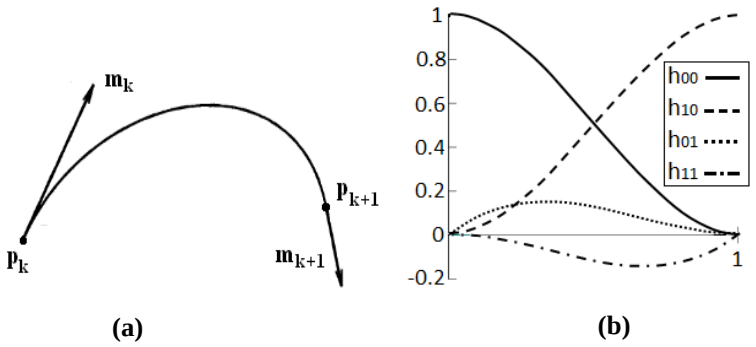
Splines são funções polinomiais que podem ser divididas em duas classes principais: *de aproximação* e *de interpolação*. As s*plines* cúbicas interpolantes são polinômios $P(t_i)$ de grau 3 que satisfazem a condição $P(t_i) = y_i$, onde y_i é o conjunto de pontos a serem interpolados, para $i = 0,1,...,n$.

A *spline cúbica de Hermite* é uma curva interpoladora definida como

$$P(t) = h_{00}(x)\,p_k + h_{10}(x)m_k + h_{01}(x)\,p_{k+1} + h_{11}(x)m_{k+1} \tag{3.3}$$

com $\boldsymbol{x} = [0,1]$, onde p_k e p_{k+1} são chamados pontos (ou nós) de controle, e m_k e m_{k+1} são as tangentes de controle, conforme a Figura 3.2a. As funções $h_{ij}(x)$ são as funções base da *spline*, dadas pelo conjunto de equações 3.4 e mostradas na Figura 3.2b. Com um ajuste correto das tangentes de controle, a *spline* de Hermite possui a propriedade de preservar a monotonicidade dos dados interpolados (Fritsch e Carlson, 1980). Em outras abordagens na literatura, é classificada também como *shape-preserving* (Moler, 2008), isto é, uma curva interpoladora que mantém a forma. Essas características a tornam conveniente para representar as principais características de sinais de processos.

$$h_{ij}(x) \begin{cases} h_{00}(x) = 2x^3 - 3x^2 + 1 \\ h_{10}(x) = x^3 - 2x^2 + x \\ h_{01}(x) = -2x^3 + 3x^2 \\ h_{11}(x) = x^3 - x^2 \end{cases} \tag{3.4}$$



(a) **(b)**

Figura 3.2: Elementos de controle da *spline* de Hermite (a), e as funções base que a compõem (b).

A *spline* cúbica de Hermite garante a continuidade da curva e da sua primeira derivada, porém não garante continuidade na segunda derivada. As tangentes m_i no conjunto de nós internos $p_i\,(i = 1,...,n)$ podem ser determinadas unicamente pelos nós do vetor p e pelas tangentes m_0 e m_n nos pontos extremos p_0 e p_n (Shikin e Plis, 1995). Essa última propriedade permite que apenas os pontos a serem gravados no processo de compressão $[t_i, y_i]$ sirvam como informação disponível para a reconstrução dos dados, mesmo que sejam necessários $2n$ elementos para compor a curva completa. Neste trabalho, uma implementação do polinômio cúbico interpolador de Hermite (PCHIP, acrônimo de *Piecewise Cubic Hermite Interpolating Polynomial*) em MATLAB® foi utilizada. Na Figura 3.3, é possível comparar as características da PCHIP com a *spline* cúbica padrão, para um conjunto qualquer de nós, evidenciando sua característica *shape-preserving*.

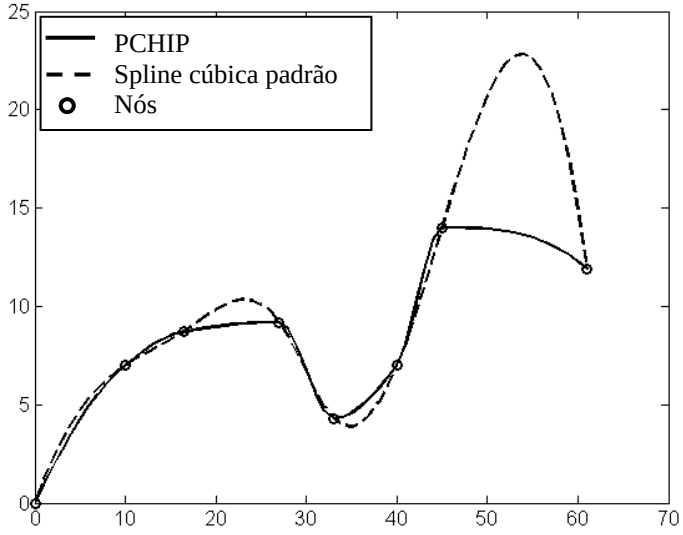


Figura 3.3: Comparação entre as implementações PCHIP e da *spline* cúbica padrão.

As tangentes na implementação PCHIP são calculadas da seguinte forma:

i) se o produto das inclinações de duas retas traçadas entre três nós consecutivos for maior que zero, isto é, as inclinações têm o mesmo sinal, a tangente no ponto central ($\boldsymbol{m_k}$) é determinada por uma média harmônica ponderada entre essas inclinações, dada pela Equação 3.5. A Figura 3.4a ilustra este caso.

$$m_k = \frac{d_{\min}}{\left(w_1 \frac{d_{k-1}}{d_{\max}} + w_2 \frac{d_k}{d_{\max}} \right)} \tag{3.5}$$

onde

$$\begin{aligned} d_k &= (y_{k+1} - y_k)/h_k \\ d_{\min} &= \min(|d_{k-1}|, |d_k|) \\ d_{\max} &= \max(|d_{k-1}|, |d_k|) \\ w_1 &= (h_{k-1} + h_s)/(3h_s) \\ w_2 &= (h_s + h_k)/(3h_s) \end{aligned} \tag{3.6}$$

$$\begin{aligned} h_s &= h_{k-1} + h_k \\ h_k &= (t_{k+1} - t_k) \end{aligned} \tag{3.7}$$

ii) se o produto das inclinações de duas retas traçadas entre três nós consecutivos for menor ou igual a zero, então a tangente no ponto central também é igual a zero (Figura 3.4b);

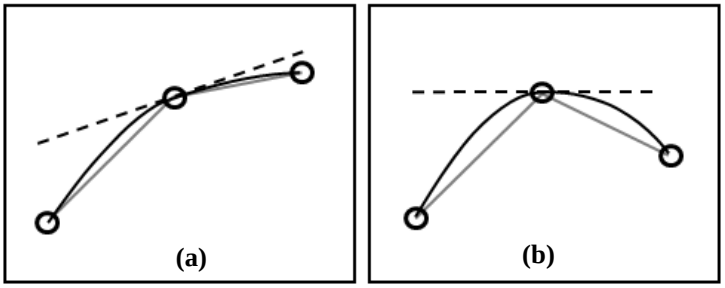


Figura 3.4: Determinação das tangentes de controle na implementação PCHIP: (a) duas inclinações consecutivas têm o mesmo sinal e (b) duas inclinações consecutivas possuem sinais distintos. (Fonte: Moler, 2008).

iii) as tangentes nos pontos extremos, m_1 e m_n, são determinadas pela *formulação forma-preservadora de três pontos não centralizada* (Fritsch e Carlson, 1980). As expressões resultantes, neste caso, são:

$$\begin{aligned} &m_1 = (2h_1 + h_2)d_1 - h_1 d_2 / (h_1 + h_2) \\ &\text{Se } \operatorname{sign}(m_1) \neq \operatorname{sign}(d_1), \quad \text{então } m_1 = 0. \\ &\text{Se } \operatorname{sign}(d_1) \neq \operatorname{sign}(d_2) \text{ e } |m_1| > 3|d_1|, \quad \text{então } m_1 = 3d_1. \end{aligned} \tag{3.8}$$

$$
\begin{aligned}
&m_n = (2h_{n-1} + h_{n-2})d_{n-1} - h_{n-1}d_{n-2}/(h_{n-1} + h_{n-2}) \\
&\text{Se } \operatorname{sign}(m_n) \neq \operatorname{sign}(d_{n-1}), \quad \text{então } m_n = 0. \\
&\text{Se } \operatorname{sign}(d_{n-2}) \neq \operatorname{sign}(d_2) \text{ e } |m_n| > 3|d_{n-1}|, \quad \text{então } m_n = 3d_{n-1}.
\end{aligned}
\tag{3.9}
$$

em que $\operatorname{sign}(j)$ denota o sinal de j.

3.3 Método Refinado

3.3.1 Descrição da estratégia

A compressão de dados nas abordagens propostas é realizada a partir de uma janela contendo os dados originais, que se desloca ao longo do sinal, em modo *offline* ou conforme eles vão sendo aquisitados. Neste método, após a etapa de regularização, a janela de dados suavizados é escalonada de modo que a faixa de valores da variável medida seja igual ao tamanho do intervalo de tempo considerado ($\delta y_s = \delta t$), conforme a Figura 3.5a, onde y_s representa o sinal suavizado. Essa normalização é feita para que os valores de derivada considerados como critério para disparar uma gravação sejam válidos em qualquer escala de dados a serem comprimidos, tornando-se mais simples identificá-los como uma mudança de tendência.

Gravam-se inicialmente o primeiro e o último pares ordenados deste intervalo, ou seja, (t_1, y_1) e (t_n, y_n). Percorrendo o sinal suavizado e escalonado na janela, na ordem crescente do tempo, avalia-se o valor da primeira derivada, cujo cálculo é feito através de diferenças finitas centrais (função `gradient` do MATLAB®). Assumem-se dois limites: $|dy_s/dt| = 1$ e $|dy_s/dt| = 0$. Quando o valor da inclinação cruzar esses limites, conforme a Figura 3.5c, da região clara para a região sombreada, o valor correspondente é gravado. Vale ressaltar que o ponto gravado não corresponde aos valores das medidas originais, mas àqueles obtidos da suavização. O valor nulo para o segundo limite é teórico. Para que ele faça sentido nesta estratégia, dever-se-ia obter exatamente $|dy_s/dt| = 0$. Uma vez que os valores da derivada são discretos, na prática, é assumido que $|dy_s/dt|$ é menor que um valor limítrofe, isto é $|dy_s/dt| < E$. Aqui, foi considerado $E = 0{,}3$ no limite inferior da Figura 3.5c.

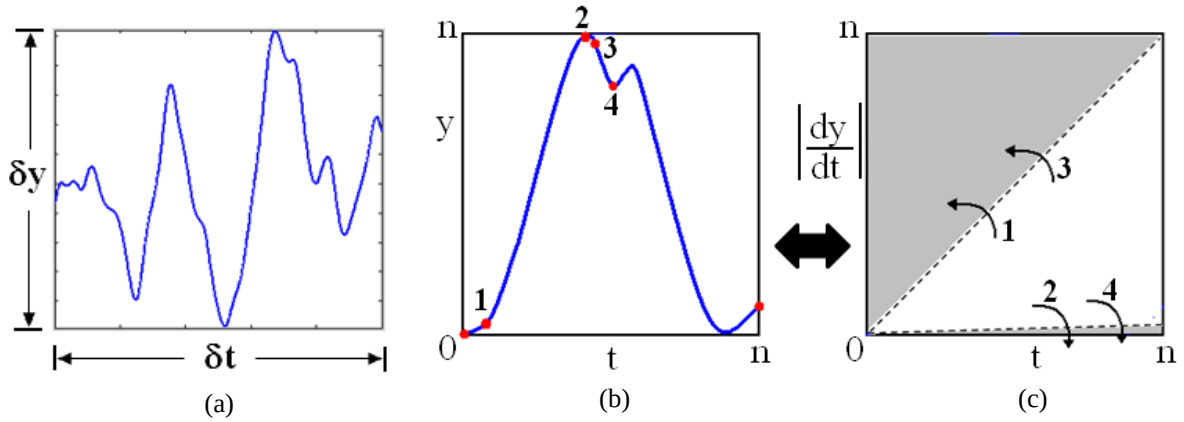


Figura 3.5: Etapas do método Refinado: (a) escalonamento da janela de dados suavizados; (b) e (c) o critério de gravação é o valor da primeira derivada, considerando-se a transição entre dois limites (linhas tracejadas). Os pontos vermelhos são os dados gravados até o local percorrido na janela (ponto 4).

Os testes realizados neste trabalho utilizaram algumas janelas finitas de dados, por isso, em todos os casos, o primeiro e o último pontos são gravados. Em uma implementação industrial, por exemplo, uma janela móvel dispensaria a gravação do último ponto, uma vez que este coincidiria com o primeiro ponto da sequência de dados seguinte. Desse modo, outro procedimento poderia ser adotado. Uma opção é não armazenar o último ponto da janela, fazendo com que a nova sequência inicie no ponto gravado anterior a ele.

Algumas porções do sinal, como estacionários ou pseudo-estacionários, podem ter, em consequência do método, pontos gravados em excesso, em casos que elas poderiam ser representadas com um número menor de dados, como exemplifica a Figura 3.6. Dessa forma, define-se um parâmetro α que representa uma tolerância para que pontos não sejam gravados nessas regiões. Este parâmetro é ajustado pelo usuário de acordo com o tipo de variável que está sendo comprimida. É possível notar a semelhança deste desvio com o parâmetro de ajuste do método Boxcar, descrito no capítulo 2. O ajuste de α, portanto, é relativamente simples e intuitivo (por exemplo, se comparado ao ajuste do algoritmo SDT), podendo ser baseado na magnitude do ruído de medição, ou em uma porcentagem do *span* do sensor ou da faixa de operação típica da variável. A cada novo dado gravado, o critério da derivada só será válido, a partir de então, caso o ponto correspondente violar a tolerância especificada.

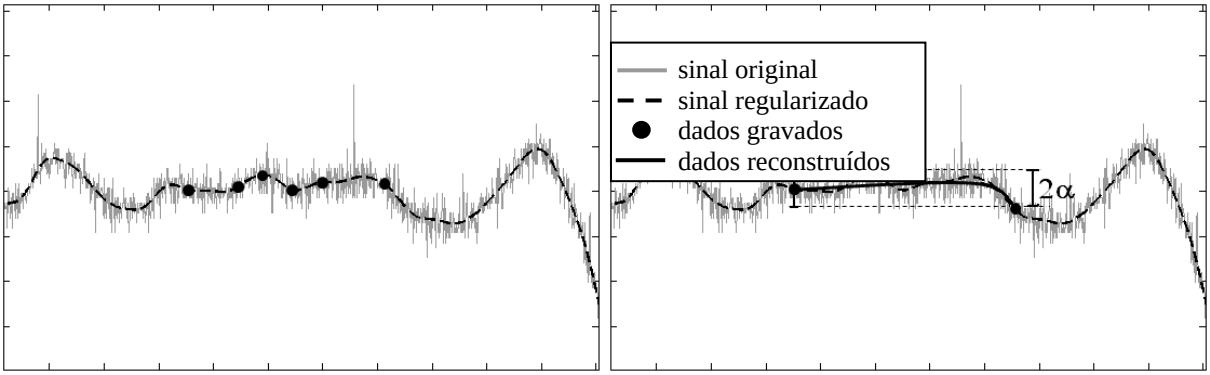


Figura 3.6: Uma porção do sinal pode ser reconstruída com poucos pontos, desde que o erro na interpolação não exceda a tolerância (parâmetro α).

3.3.2 Reconstrução e Refinamento

A reconstrução dos dados gravados é realizada através da implementação PCHIP, descrita na seção 3.2, onde os pontos armazenados representam os nós da curva polinomial. Apesar de a estratégia inicial resultar em uma reconstrução visualmente satisfatória do sinal, como mostra a Figura 3.7a, a interpolação final dos dados gravados pode apresentar algumas discrepâncias entre as tendências da curva interpolada e do sinal real. A Figura 3.7b ilustra o caso em que a região de transição mostrada possui uma considerável defasagem em sua dinâmica, comparando-se o sinal suavizado com o sinal reconstruído. Para aplicações como identificação de modelos, por exemplo, a utilização dos dados comprimidos, neste caso, geraria resultados equivocados. O algoritmo de compressão proposto é, então, submetido a um "refinamento", no qual são adicionados (gravados) novos valores nas regiões onde o erro entre a interpolação dos dados anteriormente gravados e os dados suavizados ultrapassar o valor de α. Tal procedimento é ilustrado na Figura 3.8.

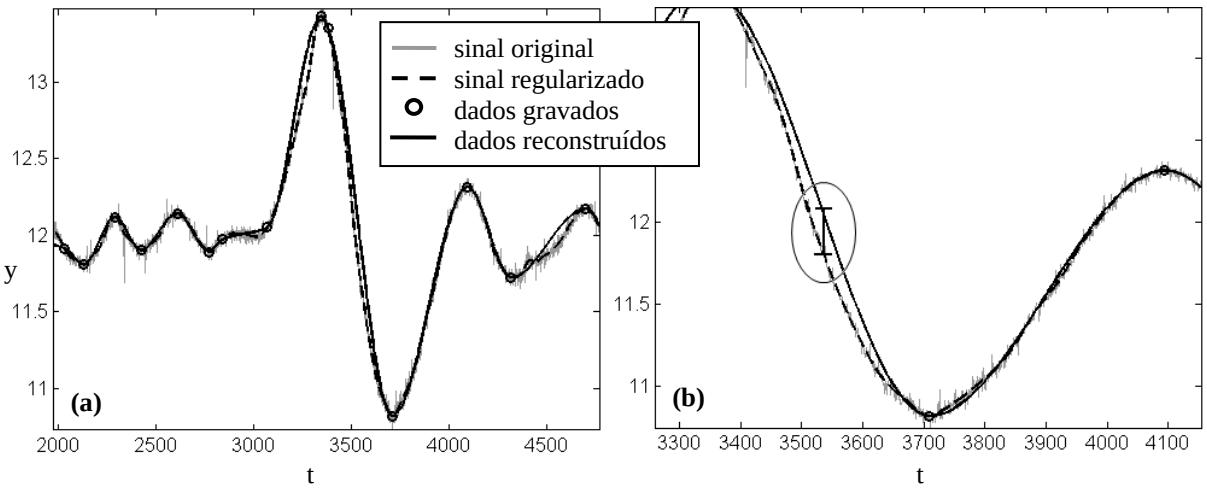


Figura 3.7: Reconstrução dos dados comprimidos: (a) a interpolação representa visualmente bem o sinal; (b) porém há discrepâncias significativas em algumas regiões.

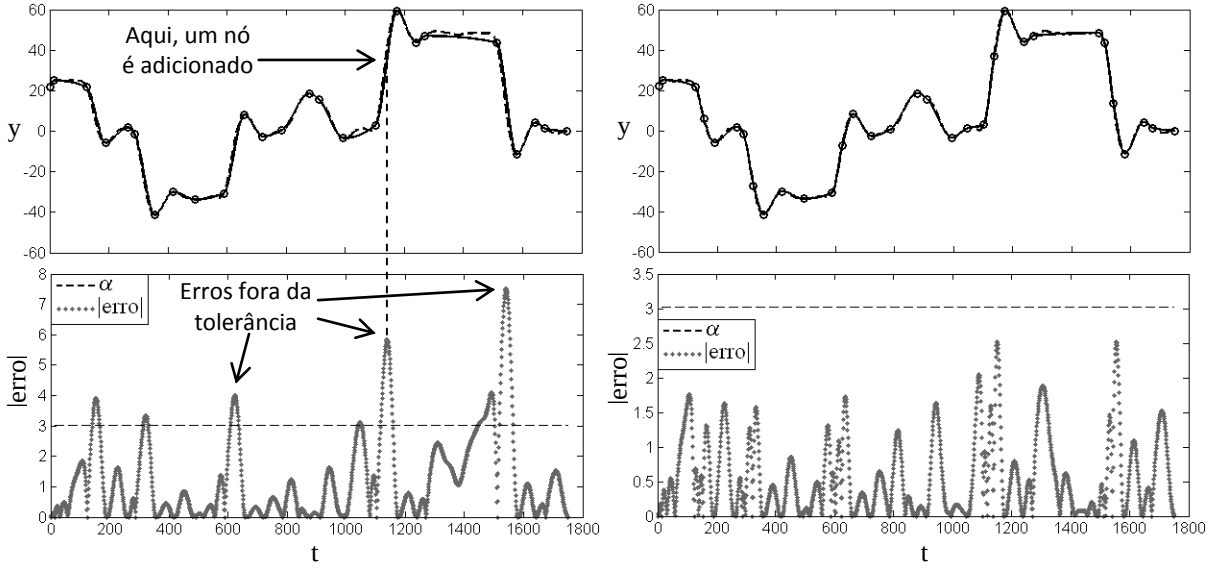


Figura 3.8: Refinamento do método de compressão que utiliza as derivadas: novos pontos são gravados nos locais de maior valor absoluto dos erros. Nos gráficos superiores, a curva tracejada é o sinal suavizado, os círculos são dados gravados e a curva preta contínua, os dados reconstruídos.

É necessário, além disso, verificar se os erros máximos (os picos do |*erro*| nos gráficos inferiores da Figura 3.8) que ultrapassam a tolerância estão dentro de uma mesma peça da *spline* composta. Denomina-se "peça" um conjunto interpolado entre dois nós consecutivos. Nesses casos, é inserido apenas 1 nó onde o erro tem maior valor absoluto dentro da mesma peça. Isso é feito, pois a adição de um único ponto, com uma nova interpolação, pode ser suficiente para eliminar todos os demais resíduos que excedem a tolerância.

Nesta etapa de refinamento, também foi considerado um critério que contempla, além do erro entre os valores dos dados suavizados e reconstruídos, o erro na sua tendência, isto é, a discrepância no valor da primeira derivada. Essa consideração, porém, foi abandonada, pois os resultados poderiam gerar interpretações equivocadas. A Figura 3.9 apresenta uma janela onde uma porção dos dados possui erros com valor baixo entre os sinais suavizado e

reconstruído, mas valores elevados no que se refere à primeira derivada (entre t = 4050 até 4400). Para o caso em questão, essa diferença não é significativa, uma vez que a curva reconstruída pode representar bem os dados da janela quando considerados no contexto geral das medidas desta variável. Em outras palavras, as oscilações que aparecem no sinal desta janela não representam real mudança de tendência, e a curva que o reconstroi é satisfatória. De outro modo, conforme mostra a região indicada na Figura 3.9, em $t \approx 3980$ (ou mesmo na Figura 3.7b), os valores da primeira derivada dos sinais suavizado e reconstruído não apresentam discrepância, porém o erro local entre os valores da medida são bastante significativos. Se for considerado, então, uma contribuição simultânea de erros locais e erros na derivada, novamente ter-se-ia um resultado equivocado, que indicaria a não necessidade de gravação de um valor naquela região. Ainda, se o erro relativo à derivada fosse apreciado nessa análise, seria necessário especificar um valor de tolerância para o critério que, além de gerar mais parâmetros de ajuste, não é tão intuitivo quanto o parâmetro α, já incluído no refinamento.

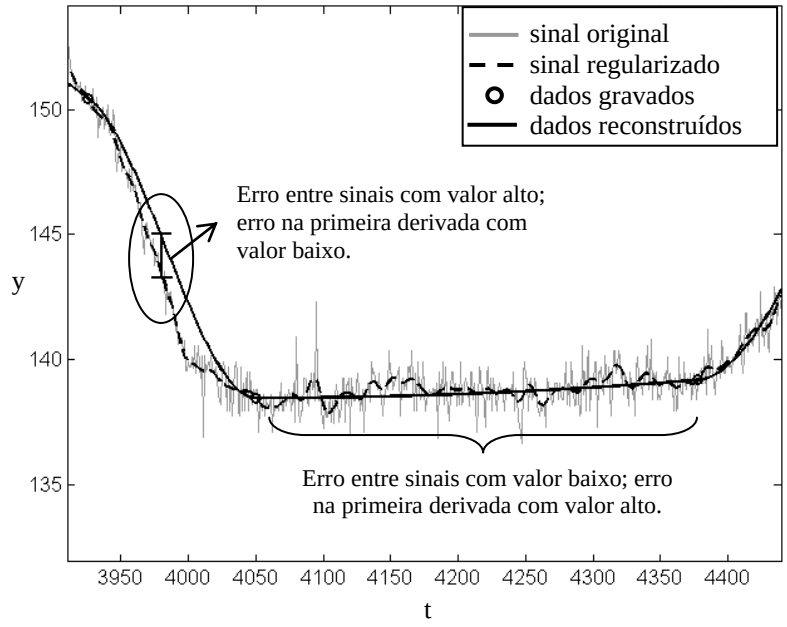


Figura 3.9: Critérios de refinamento considerando o valor da primeira derivada podem gerar resultados equivocados.

3.4 Método Evolutivo

A etapa de refinamento da metodologia apresentada na seção anterior pode ser individualmente aplicada como método de compressão. Para esse método, não é necessário realizar o escalonamento da janela de dados. Inicialmente, gravam-se o primeiro e o último valores contidos na janela. A estratégia consiste em interpolar, com a PCHIP, esses dois pontos e, a partir de então alternar entre a colocação de novos nós nos locais de maior erro e a interpolação do novo conjunto de pontos (etapas a, b e c da Figura 3.10), evoluindo para uma interpolação definitiva que reconstroi os dados sem violar a tolerância especificada (Figura, 3.10d).

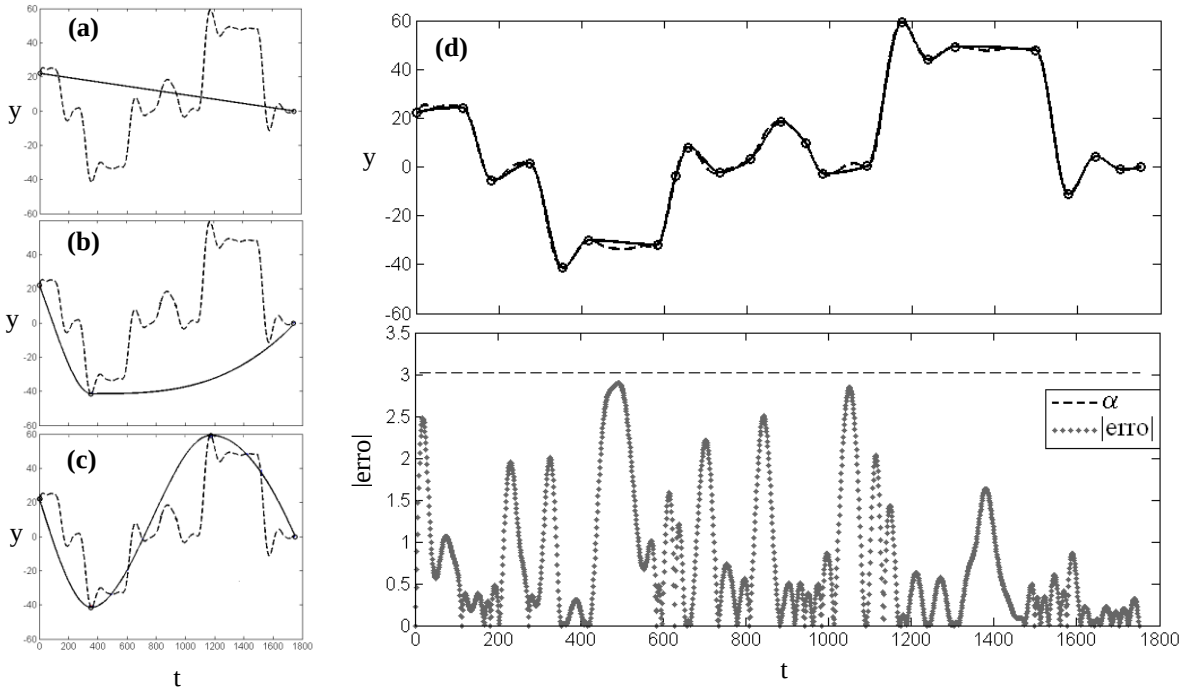


Figura 3.10: Funcionamento do algoritmo no método Evolutivo, exemplificado pelas etapas (a), (b) e (c), e o resultado final da compressão em (d), com os erros absolutos. Nos gráficos que representam a janela de dados, a curva tracejada é o sinal suavizado, os círculos são dados gravados e a curva preta contínua, os dados reconstruídos.

Referências

Akaike, H. (1974) NEW LOOK AT STATISTICAL-MODEL IDENTIFICATION. *Ieee Transactions on Automatic Control*, AC19**,** 716-723.

Fritsch, F. N. & R. E. Carlson (1980) MONOTONE PIECEWISE CUBIC INTERPOLATION. *Siam Journal on Numerical Analysis*, 17**,** 238-246.

Guan, L., J. A. Anderson & J. P. Sutton (1997) A network of networks processing model for image regularization. *Ieee Transactions on Neural Networks*, 8**,** 169-174.

Mejia, R. I. G., R. C. Freitas & J. O. Trierweiler. (2010). Derivatives Estimation Based on Smoothing Techniques. In *Bolivian Engineering and Technology Congress*. La Paz.

Moler, C. (2008). *Numerical Computing with MATLAB*. The Mathworks Inc. http://www.mathworks.com/moler/chapters.html (last accessed 2/24/2011).

Shikin, E. V. & A. I. Plis. 1995. *Handbook on Splines for the User*. Florida: CRC-Press.

Stickel, J. J. (2010) Data smoothing and numerical differentiation by a regularization method. *Computers & Chemical Engineering*, 34**,** 467-475.

Wahba, G. 1976. A survey of some smoothing problems and the method of generalized cross-validation for solving them. In *Conference on the Applications of Statistics*. Dayton, Ohio.

Yoon, S. H. & P. A. Nelson (2000) Estimation of acoustic source strength by inverse methods: Part II, experimental investigation of methods for choosing regularization parameters. *Journal of Sound and Vibration*, 233**,** 669-705.

Capítulo 4

Ajuste da Compressão no Sistema PI

Apesar do relativo sucesso na indústria, na prática a qualidade dos dados comprimidos no sistema PI da OSIsoft® frequentemente é insatisfatória devido a configurações inadequadas dos algoritmos de compressão. Algumas ferramentas comerciais foram desenvolvidas para realizar a sintonia dos parâmetros do sistema PI. O *Exele's PI Tunning Tools* (EXELE Information System Inc.) é uma dessas ferramentas. Ela permite ao usuário identificar *tags* que possuem muitos eventos de arquivamento, coleta dados brutos destes *tags*, plotando e comparando os dados originais com os dados comprimidos usando diferentes configurações para o desvio de compressão. Outra ferramenta existente no mercado foi desenvolvida pela Pattern Discovery Technologies Inc., chamada *CompressionInsight*™, que recomenda valores para os parâmetros de compressão através de técnicas estatísticas e algoritmos de otimização. A estrutura do sistema PI e o funcionamento dos algoritmos de exceção e compressão, responsáveis pelo armazenamento dos dados, foram revisados nos capítulos 1 e 2, respectivamente. Neste capítulo, serão apresentadas duas abordagens sistemáticas para a sintonia dos parâmetros envolvidos na compressão de dados deste sistema. As técnicas propostas, bem como os algoritmos de exceção e compressão do sistema PI foram implementados em ambiente MATLAB®.

A exemplo das técnicas descritas no capítulo 3, as abordagens propostas neste capítulo também se valem de um conjunto prévio dos dados originais. Deste conjunto devem constar alguns pontos julgados essenciais para representar o sinal, ou seja, esses pontos (ou outros, nas mesmas regiões) devem permanecer após o processo de compressão do sistema PI. Aqui, duas formas para obter estes pontos são utilizadas: i) pelo resultado da compressão dos dados com uma das técnicas descritas no capítulo 3; ii) localização de pontos convenientemente escolhidos pelo usuário. Na seção 4.1, é descrito o primeiro método, no qual o desvio de compressão é ajustado a partir desses pontos e daqueles obtidos com o teste de exceção, valendo-se do próprio funcionamento do algoritmo do PI. No segundo método (seção 4.2), é formulado um problema de otimização, segundo alguns critérios considerados, em que os parâmetros a serem ajustados (desvios de exceção e compressão) são as variáveis de decisão.

É importante que a janela selecionada para o procedimento de ajuste dos parâmetros contenha pontos na região de operação típica da variável ou da maior faixa possível do *span* do sensor. Dessa forma, os parâmetros são sintonizados apenas uma vez, e seus valores serão sempre adequados para a operação da variável em longo prazo.

4.1 Primeiro método: ajuste do desvio de compressão

A técnica proposta visa sintonizar o principal parâmetro do teste de compressão: o desvio de compressão ($CompDev$). Para o desvio de exceção ($ExcDev$), neste caso, é feito manualmente e, para os estudos de caso abordados, foram assumidos valores entre 2% e 3% da faixa de valores contida nas janelas de dados. Conforme discutido no capítulo 2, na prática este valor pode ser baseado no desvio padrão do ruído, em uma porcentagem da faixa típica de operação da variável ou do *span* do sensor. Para o parâmetro "tempo máximo para exceção" assumiu-se valor infinito. Nos testes realizados, para o caso em que se faz a seleção manual dos pontos, conforme a opção (ii) descrita anteriormente, foi utilizada a função `ginput()` do MATLAB®. Essa função captura as coordenadas de pontos selecionados graficamente através do *mouse*.

Para elucidar o procedimento da determinação do $CompDev$ será utilizado como exemplo o sinal da Figura 4.1. Consideram-se os pontos representados pelos círculos em vermelho (**1**, **2** e **3**): eles representam alguns dados do conjunto gravado por um dos métodos propostos no capítulo 3, ou daquele selecionado manualmente pelo usuário. Consideram-se também as retas traçadas entre os pares destes pontos (no detalhe da Figura 4.1) escalonando previamente os dados da janela (análogo à primeira etapa do método Refinado, descrito no capítulo anterior). Assume-se que há uma mudança na tendência dos dados se um, e apenas um, dos valores absolutos das tangentes de duas retas consecutivas estiver no intervalo $[0,1)$. Isto significa que o ponto intermediário (no exemplo, o ponto **2**), ou algum ponto nas suas proximidades, deve ser gravado. É possível acompanhar pela Figura 4.1, as etapas seguintes do procedimento:

I) Aplica-se o teste de exceção aos dados da janela;

II) Com os pontos obtidos no teste de exceção (círculos pretos), encontra-se aquele que estiver à menor distância euclidiana do ponto **2** (isto é, o ponto **P**);

III) O ponto **S** é o ponto do teste de exceção subsequente àquele encontrado no passo (II) (ponto **P**). Forma-se um paralelogramo entre os pontos **1** e **S** cuja largura terá valor igual à maior distância entre o centro do paralelogramo e os pontos do teste de exceção neste intervalo. Este será o $CompDev_1$;

IV) Repete-se o procedimento considerando os demais pares (outros "pontos vermelhos" que estiverem na janela). O valor final do desvio de compressão é obtido através de uma média aritmética entre as larguras de todos os $CompDev_i$ $(i = 1,2,...,n)$.

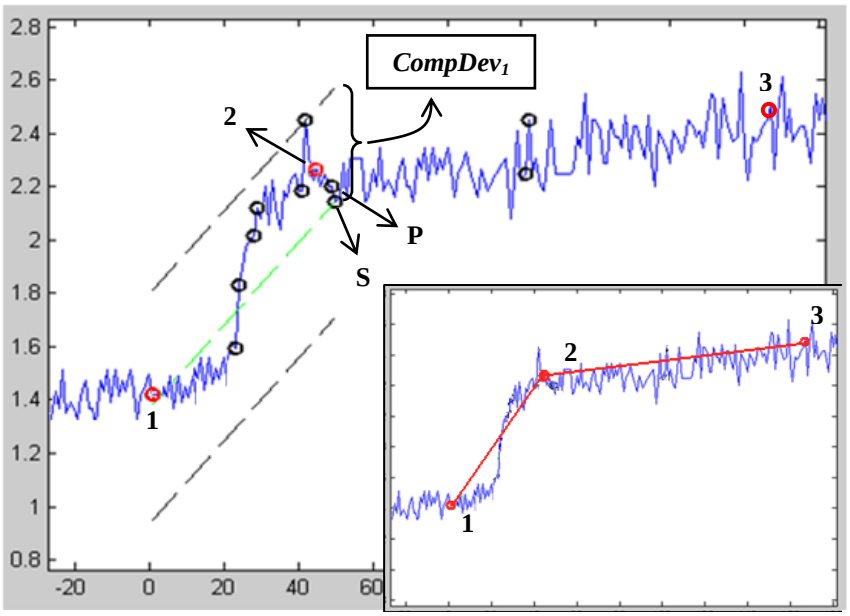


Figura 4.1: Exemplo de aplicação da primeira abordagem (apenas uma porção da janela) em uma implementação em MATLAB®.

4.2 Segundo método: ajuste por otimização

Nesta técnica é formulado um problema de otimização em que se busca minimizar a seguinte função objetivo:

$$J = \gamma_1 \left| \frac{N_S - N_{PI}}{N_S} \right| + \gamma_2 \,\mathrm{mse}(y_{iS} - y_{iPI}) \tag{4.1}$$

$$\mathrm{mse}(y_{iS} - y_{iPI}) = \frac{\sum_{j=1}^{T} (y_{iS_j} - y_{iPI_j})^2}{\sum_{j=1}^{T} 1} \tag{4.2}$$

onde:

$\boldsymbol{T}$: número de pontos na janela de dados;
$\boldsymbol{N_S}$: número de pontos pré-selecionados ou obtidos da compressão com os métodos propostos;
$\boldsymbol{N_{PI}}$: número de gravados pelo algoritmo do PI;
$\boldsymbol{y_{iS}}$: dados linearmente interpolados com os pontos pré-selecionados;
$\boldsymbol{y_{iPI}}$: dados linearmente interpolados com os pontos gravados pelo algoritmo do PI;
γ_1: peso para a contribuição da diferença relativa entre o número de pontos gravados;
γ_2 : peso para a contribuição do erro médio quadrático (*mse*) entre as interpolações.

Os pesos na função objetivo podem ser ajustados a fim de se atingir um bom compromisso entre o número de pontos gravados e a qualidade da representação dos dados originais, ou para priorizar um desses atributos. Neste trabalho, para todos os casos testados, γ_1 e γ_2 são iguais a 1. O vetor X das variáveis de decisão contempla os parâmetros de sintonia dos algoritmos na compressão do sistema PI (*ExcDev* e *CompDev*). O problema de otimização está sujeito às seguintes restrições:

$$\begin{aligned} ExcDev &> 0 \\ CompDev &> 0 \end{aligned} \tag{4.3}$$

Pode-se observar, pela Figura 4.2b, o comportamento de J em função das variáveis de decisão para a janela de dados Figura 4.2a juntamente com um determinado conjunto de pontos pré-selecionados. Devido à alta irregularidade da função objetivo, é necessário um método de busca que evite a utilização da informação sobre a derivada (método de Newton e suas variantes) no processo de solução. Ainda, como J é não convexa também é necessário um algoritmo de busca global. Para a implementação em MATLAB®, foi utilizado o algoritmo DIRECT (Finkel, 2003).

O processo de otimização depende fortemente da faixa de valores contemplada na região de busca, como a mostrada na Figura 4.2b. Este pode ser um fator importante para facilitar a busca pela solução ou reduzir o tempo computacional do problema. A escolha dos valores dessa faixa vai depender da variável para a qual estão se ajustando os parâmetros de compressão, ou ainda recorrendo-se a uma normalização dos dados da janela, de forma que esses valores possam permanecer fixos. Uma regra heurística razoável pode ser utilizada considerando o ajuste padrão do sistema PI, no qual o *CompDev* corresponde à metade do *ExcDev*. Assim, determinam-se somente os valores mínimo e máximo deste parâmetro, e a faixa de valores daquele será fixada automaticamente.

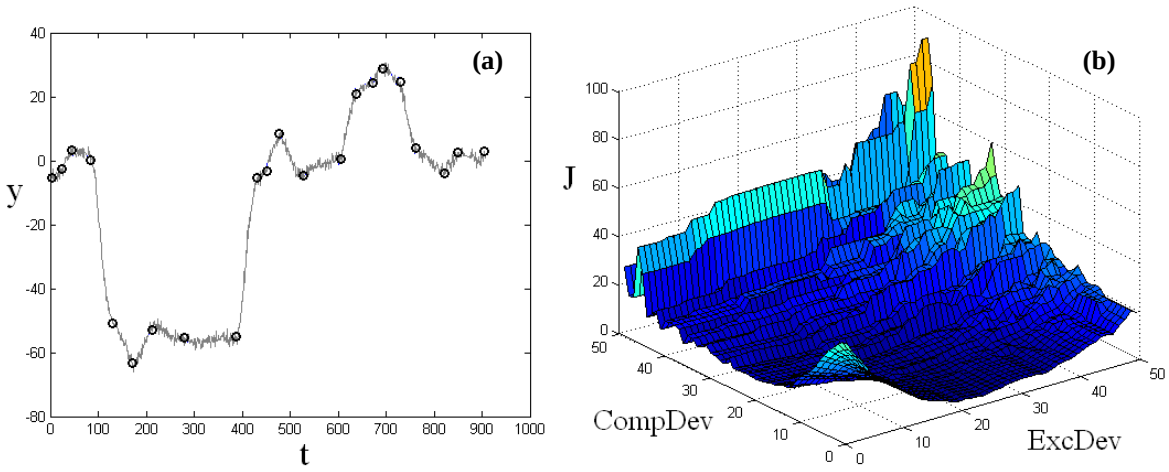


Figura 4.2: Função objetivo (b) do problema de otimização formulado para o sinal em (a), variando-se os desvios de exceção e compressão.

No capítulo 5 serão apresentados tanto os resultados das novas metodologias para compressão, abordadas no capítulo 3, quanto das propostas para ajuste do PI, descritas no presente capítulo.

Referências

EXELE Information Systems Inc. PI Tuning Tools, v. 4.0.11. Disponível em: <www.exele.com/products/PI/PI Tuning Tools/pitune.htm>. Acessado em: 19 de dezembro de 2011.

Finkel, D. E. (2003) DIRECT Optimization Algorithm User Guide. Technical Report CRSC-TR03-11, Center for Research and Scientific Computation, North Carolina State University, Raleigh.

PATTERN DISCOVERY TECHNOLOGIES INC. Compression*Insight*™. Disponível em: <www.patterndiscovery.com/products/compressioninsight.htm>. Acessado em: 19 de dezembro de 2011.

Capítulo 5

Estudos de Caso

5.1 Apresentação geral e critérios utilizados

Para avaliar as propostas deste trabalho, alguns conjuntos de dados de processos foram utilizados como estudo de caso. Quatro tipos de sinais reais – oriundos da planta laboratorial de seis tanques esféricos (Paim, 2009), localizada no laboratório LACIP II do Departamento de Engenharia Química, e mostrados na Figura 5.1 – visaram a avaliar as metodologias propostas no capítulo 3, segundo critérios pré-estabelecidos (discutidos subsequentemente). Além desses, uma extensa série de sinais foi gerada por simulação para complementar a comparação entre os métodos de compressão propostos.

Outro conjunto específico com 3 tipos de sinais artificiais, mostrado na Figura 5.2, foi gerado com o objetivo de verificar a utilização dos dados comprimidos para identificação de modelos, comparando estes resultados com aqueles que seriam obtidos pela identificação direta, isto é, com os dados sem compressão. Para avaliar as técnicas propostas para a sintonia da compressão do PI, 2 sinais reais, da Figura 5.1, e 2 artificiais, da Figura 5.2, foram utilizados.

A partir daqui os sinais reais serão identificados como SINALRa, SINALRb, SINALRc e SINALRd, de acordo com as mesmas letras que os identificam na Figura 5.1. Da mesma forma, aos sinais artificiais gerados para os procedimentos de identificação serão atribuídos os rótulos SINALAa, SINALAb e SINALAc, de acordo com a Figura 5.2. Percebe-se que boa parte desses sinais possui características dinâmicas diversificadas, bem como apresentam alguns *outliers* (valores atípicos, muito afastados dos demais valores próximos). Os dados artificiais foram obtidos por simulações de algumas funções de transferência com características distintas. Buscou-se gerar, nesses casos, peculiaridades como fase não mínima e subamortecimento. As unidades de y e u não são mostradas nas figuras, porque não são relevantes para as análises.

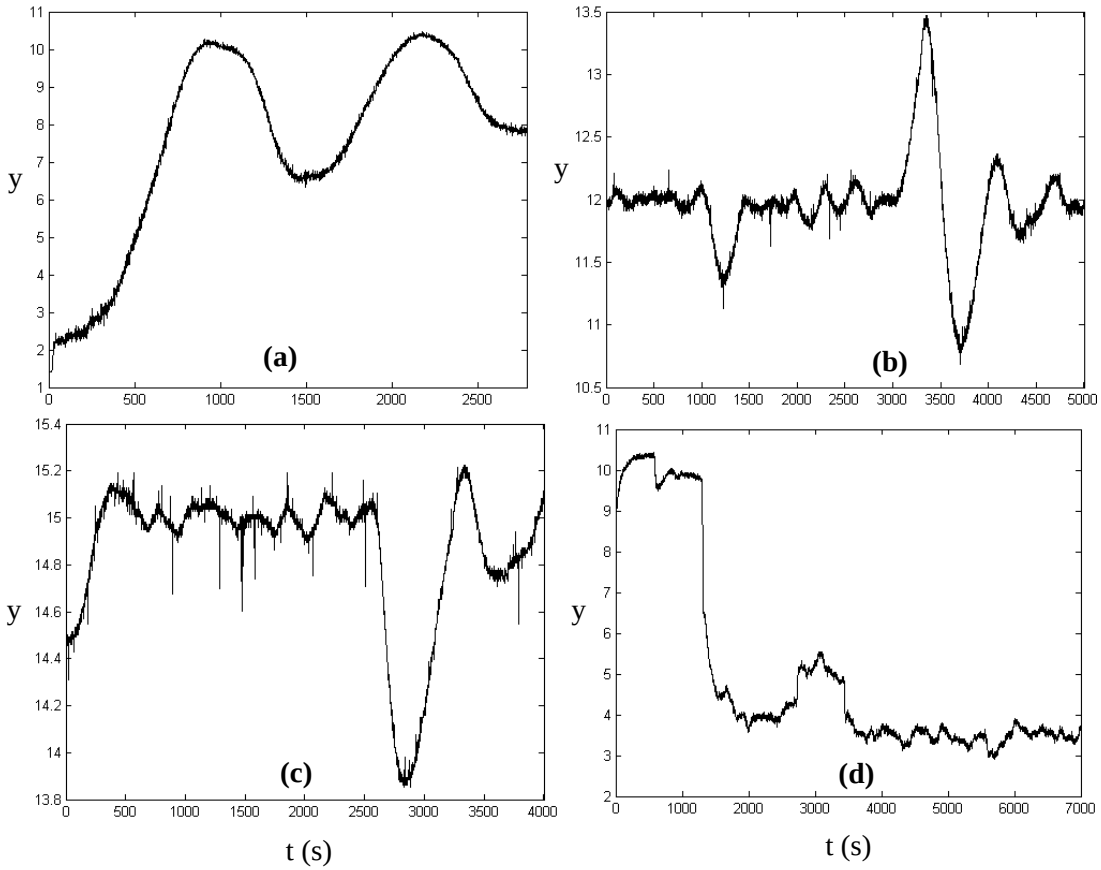


Figura 5.1: Sinais reais da planta de seis tanques esféricos.

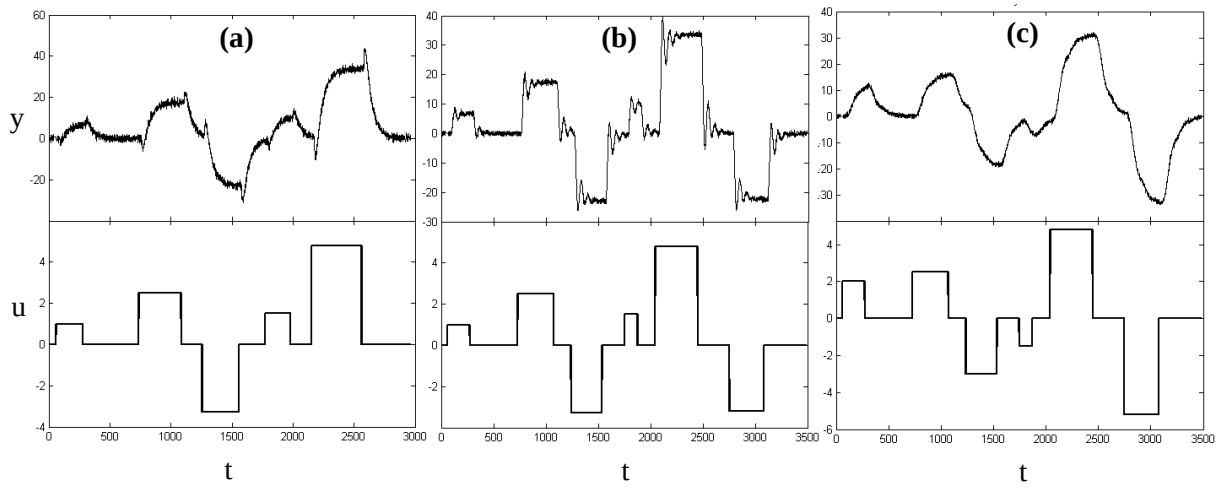


Figura 5.2: Sinais artificiais gerados com perturbações em alguns sistemas, por simulação.

Todos os testes serão avaliados de acordo com o número de pontos gravados e o erro de reconstrução (ER) definido no capítulo 2 através da equação 2.4. A quantidade de dados armazenados será traduzida pelo complementar do inverso da razão de compressão (CRC), que foi definida pela equação 2.3, dado, em percentual, por

$$CRC\ (\%) = \left(1 - \frac{1}{RC}\right) \times 100 \qquad (5.1)$$

Dessa forma, é possível avaliar este critério de forma relativa e independente do tamanho da janela de dados considerada. Bons algoritmos de compressão são aqueles em que se obtêm altas CRC's com menores valores possíveis no ER e no tempo computacional exigido.

Ainda, para a compressão dos dados reais com as metodologias do capítulo 3, será avaliado o erro médio quadrático na primeira derivada do sinal reconstruído frente ao valor do mesmo critério para o sinal suavizado, a fim de quantificar se a forma de escolha dos pontos a serem gravados reteve, após a interpolação, a maior parte da tendência das variáveis de processo.

Para os testes de identificação, com os sinais da Figura 5.2, foram comparados os valores das integrais do erro absoluto e quadrático entre os modelos identificados com os dados antes e após a compressão. As identificações foram feitas usando a ferramenta *System Identification Toolbox*™ do MATLAB®. Todos os modelos identificados foram do tipo *Output Error* (OE) de ordem 2, considerando os desvios do sinal em relação à média dos dados na janela. As rotinas para os procedimentos de identificação implementadas permitem variar o valor do tempo morto, e a seleção do melhor modelo é feita levando-se em conta os valores do erro na saída e da derivada deste erro.

5.2 Resultados e discussão

5.2.1 Métodos de suavização ou regularização

Primeiramente, foram analisadas as técnicas de suavização/regularização utilizadas neste trabalho e descritas no capítulo 3 – filtros *Butterworth* (GCV-But) e *B-splines* (Bsp-AIC) adaptativos – a fim de verificar a influência na escolha do método no algoritmo de compressão Refinado. A Figura 5.3 exibe o resultado da compressão/descompressão por este método para os sinais **b** e **d** da Figura 5.1 quando se aplicam estes métodos de suavização. Nesses casos, o parâmetro de ajuste α é igual a 3% da faixa total da variável medida na janela (δy^{jan}). Os resultados para o esforço computacional, RC e ER estão apresentados na Tabela 5.1.

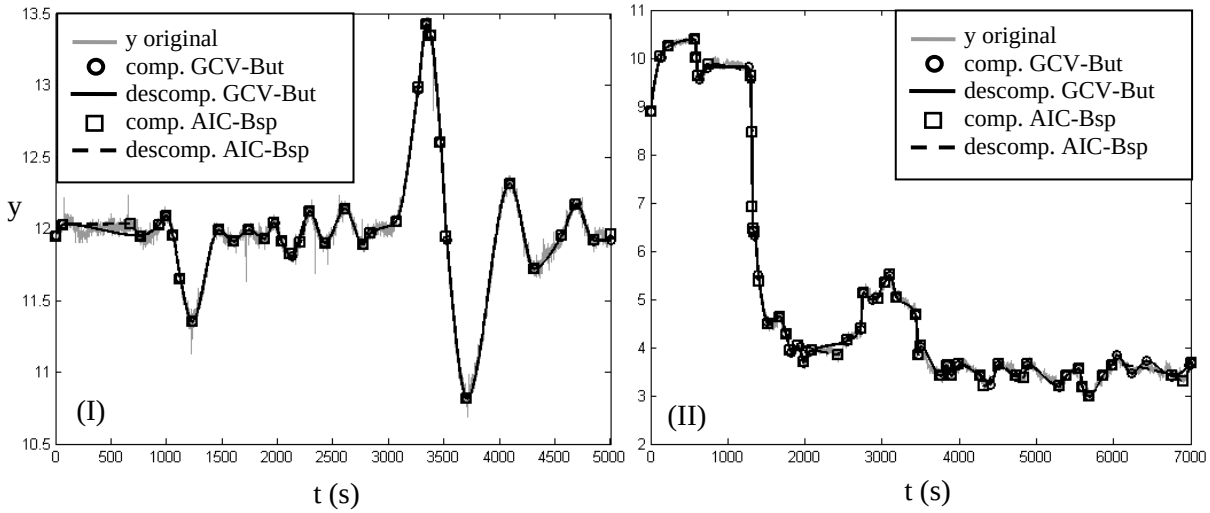


Figura 5.3: Resultados da compressão do SINALRb (I) e SINALRd (II) pelo método Refinado com suavização por GCV-But e AIC-Bsp.

Tabela 5.1: Resultados da compressão pelo método Refinado utilizando duas técnicas de suavização.

	Método	Tempo computacional (s)*	CRC (%)	ER
SINALRb	GCV-But	0,43	98,68	0,0014
	AIC-Bsp	6,64	98,60	0,0012
SINALRd	GCV-But	0,51	98,54	0,0076
	AIC-Bsp	13,66	98,54	0,0061

*Testes realizados em um computador com processador Intel® Core™i5 e 3 GB de memória RAM.

Visualmente, os resultados da compressão se apresentaram bastante semelhantes, e apesar de armazenarem praticamente o mesmo número de pontos, a utilização do método GCV-But apresentou um ER ligeiramente superior àquele obtido quando se utiliza o AIC-Bsp, porém exigiu uma carga computacional consideravelmente menor. A regularização com AIC-Bsp tende a gerar resultados levemente mais oscilatórios que o método GCV-But, como pode ser visto na Figura 5.4, que é a aproximação em uma porção do sinal da Figura 5.3b (SINALRd). Por isso, a localização dos pontos antes do refinamento do método de compressão proposto difere um pouco entre as técnicas de suavização, devido ao fato de se utilizar o valor da primeira derivada, o que pode explicar as diferenças observadas na Figura 5.3, obtidas ao término da aplicação do método Refinado.

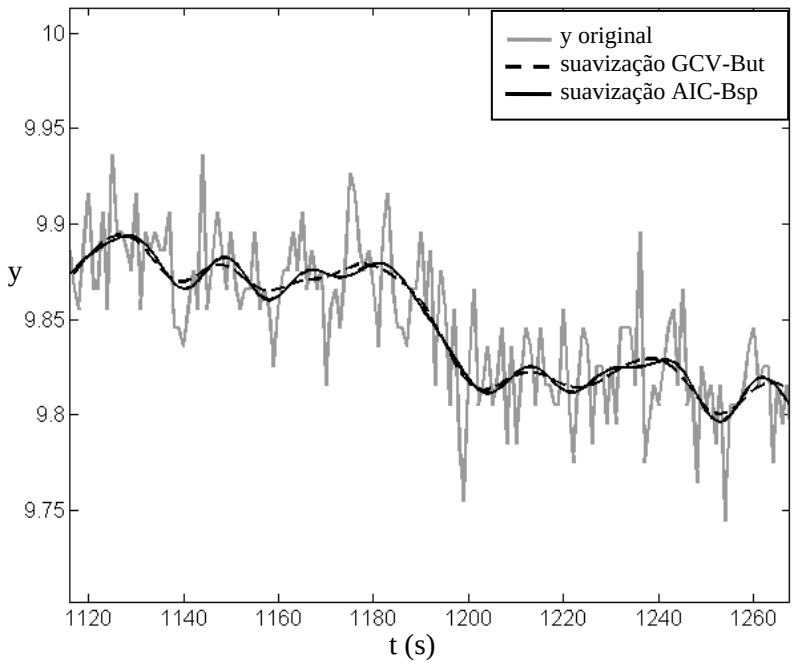


Figura 5.4: Uma porção do SINALRd em *zoom*. As técnicas de suavização utilizadas apresentam pequenas diferenças que podem influenciar no resultado da compressão.

5.2.2 Metodologias de compressão propostas

A Figura 5.5 exibe os resultados gráficos da compressão/descompressão dos sinais reais com os métodos Refinado e Evolutivo, em que α é igual a 3% de δy^{jan}. Os critérios para avaliação são apresentados na Tabela 5.2. Nesses casos e naqueles apresentados daqui em diante, foi utilizado o GCV-But como método de regularização, pois este apresentou menor esforço computacional, possuindo diferenças pouco significativas nos demais aspectos em relação ao método AIC-Bsp.

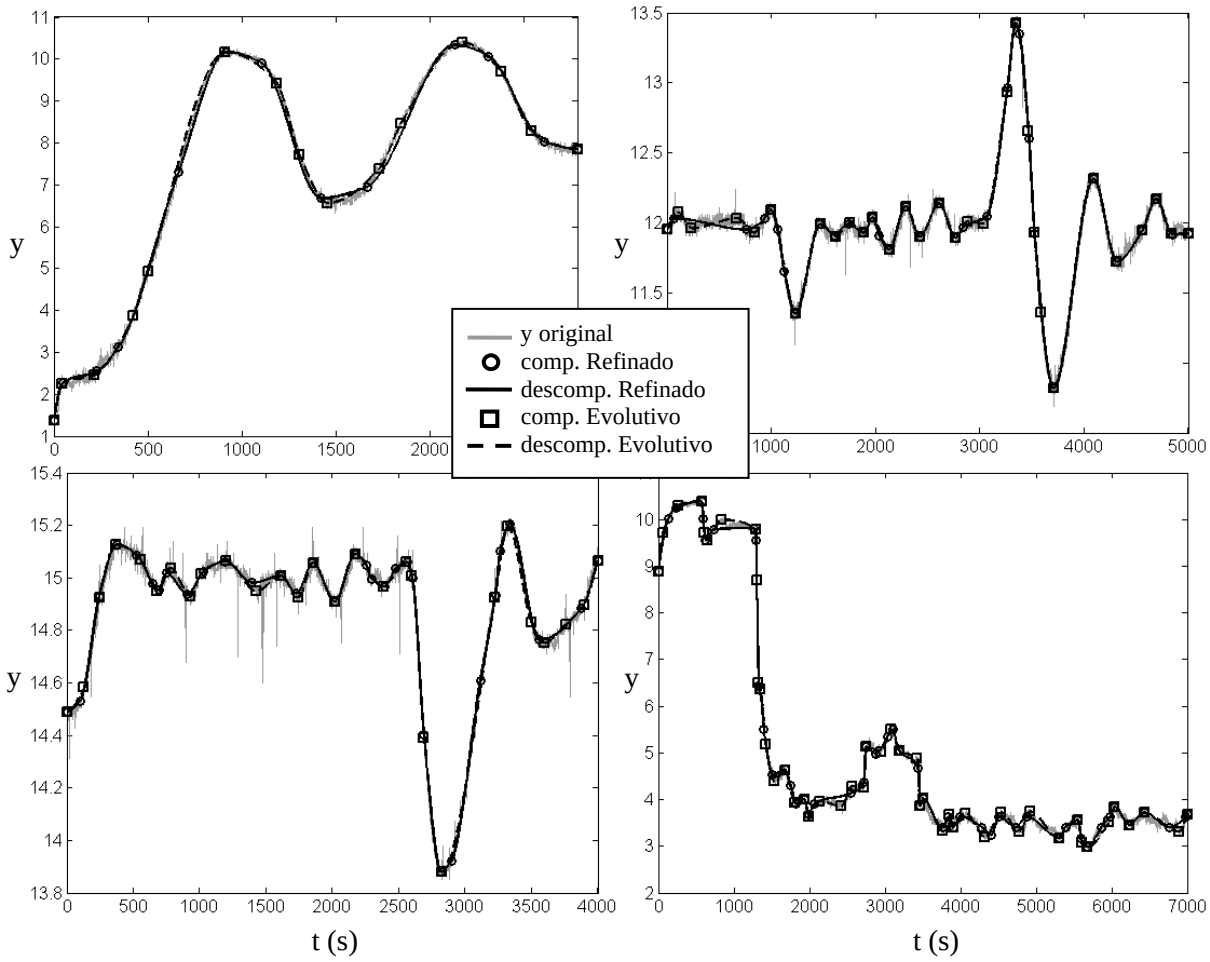


Figura 5.5: Resultados da compressão pelos métodos Refinado e Evolutivo, para os sinais reais.

Em termos visuais, os resultados não apresentam grandes diferenças quando da comparação entre os métodos Refinado e Evolutivo. Pela Tabela 5.2 é possível perceber que em alguns casos (SINALRb e SINALRd) a RC do método Evolutivo possui maior valor que o a RC do Refinado, porém o ER e o erro na derivada (E_{der}) são levemente inferiores. Para o SINALRc e o SINALRa, esta tendência não é verificada. Normalmente, a RC e o ER são grandezas diretamente proporcionais, isto é, quanto menor o número de pontos gravados (maior RC), maior é o erro na interpolação final (maior ER). Conclui-se então, para aqueles casos, que o método Evolutivo foi capaz de gravar um menor número de pontos, porém em locais mais apropriados que o Refinado, o que lhe rendeu menores ER's.

Tabela 5.2: Resultados da compressão dos sinais reais pelos métodos Refinado e Evolutivo.

	Método	CRC (%)	ER	E_{der}
SINALRa	Refinado	99,07	0,0155	$7{,}0 \cdot 10^{-6}$
	Evolutivo	98,92	0,0110	$7{,}0 \cdot 10^{-6}$
SINALRb	Refinado	98,68	0,0014	$4{,}5 \cdot 10^{-7}$
	Evolutivo	98,76	0,0011	$3{,}8 \cdot 10^{-7}$
SINALRc	Refinado	98,35	$6{,}60 \cdot 10^{-4}$	$3{,}3 \cdot 10^{-7}$
	Evolutivo	98,60	$6{,}75 \cdot 10^{-4}$	$3{,}6 \cdot 10^{-7}$
SINALRd	Refinado	98,54	0,0076	$2{,}35 \cdot 10^{-5}$
	Evolutivo	98,69	0,0071	$1{,}81 \cdot 10^{-5}$

A comparação, nesses casos, não é conclusiva, uma vez que foi utilizado um conjunto limitado de janelas de dados, e os resultados são dependentes do tipo de sinal. Para complementar essa avaliação, então, foi gerado um conjunto de mil janelas de dados com características distintas combinadas aleatoriamente, adicionando a estes diferentes níveis de ruído. Cada janela contém 1000 pontos e contempla um sinal-base de um dos três tipos a seguir: sinal constante, rampa e oscilação senoidal. A cada intervalo de 200 pontos, um tipo de distúrbio foi adicionado ao sinal-base aleatoriamente. As características desses distúrbios são: degrau, oscilação senoidal, resposta de primeira ordem, resposta de segunda ordem com subamortecimento e segunda ordem com resposta inversa. Por fim, para cada janela gerada com uma base e cinco distúrbios, foi adicionado ruído branco com seis magnitudes distintas: 0 (sem ruído), 0,01, 0,05, 0,1, 0,3 e 0,5, totalizando seis mil janelas de dados para comparar os métodos propostos. Um exemplo de sinal gerado de acordo com os procedimentos descritos pode ser visto na Figura 5.6.

Para cada uma das janelas de dados aplicaram-se os métodos Refinado e Evolutivo, e determinaram-se, para cada caso, a RC e o ER obtidos. O parâmetro α foi variado conforme o nível de ruído aplicado. De fato, para sinais com baixa relação sinal-ruído (SNR), $\alpha = 3\%$ de δy^{jan} mostrou-se adequado para comprimir os dados com boa relação RC x ER, conforme notado nos resultados anteriores. Para valores mais elevados da SNR, o sinal suavizado passa a apresentar algumas oscilações nas regiões onde estão presentes apenas características estacionárias, como se verifica na Figura 5.7. Dessa forma α deve ter seu valor aumentado para que pontos em excesso não sejam armazenados. Como os dados gerados não provêm de processos reais, não é possível usar, por exemplo, informações como o *span* do sensor ou a faixa típica de operação da variável de processo para determinar α.

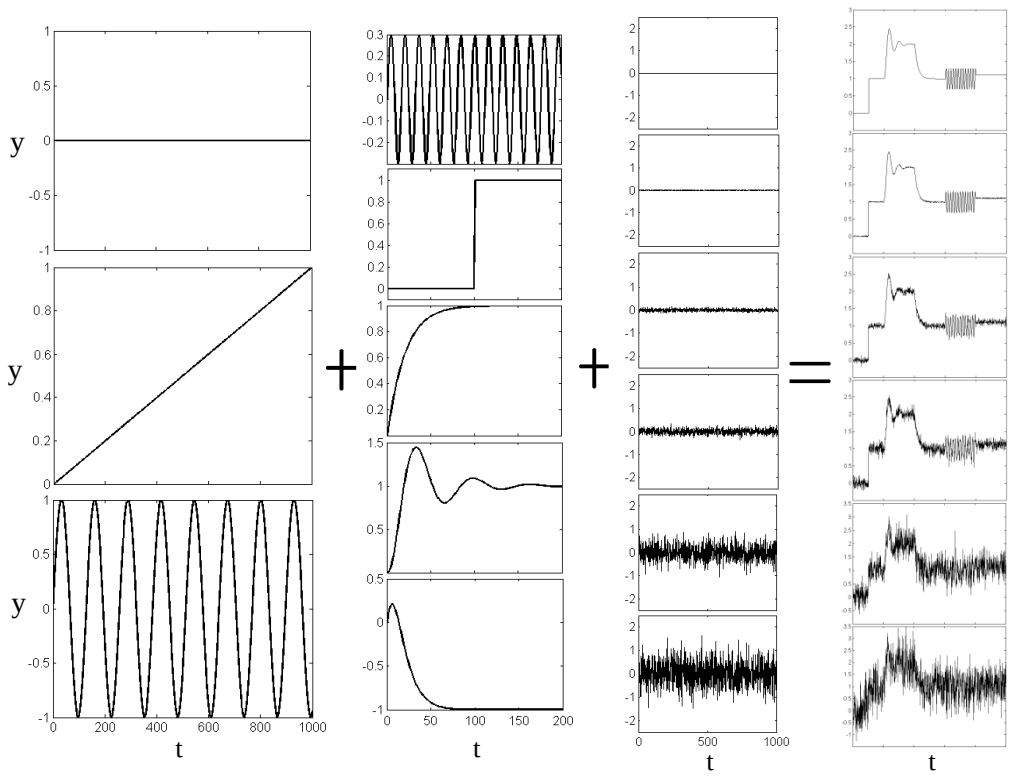


Figura 5.6: Representação esquemática da construção dos sinais com diferentes bases, distúrbios e níveis de ruído branco.

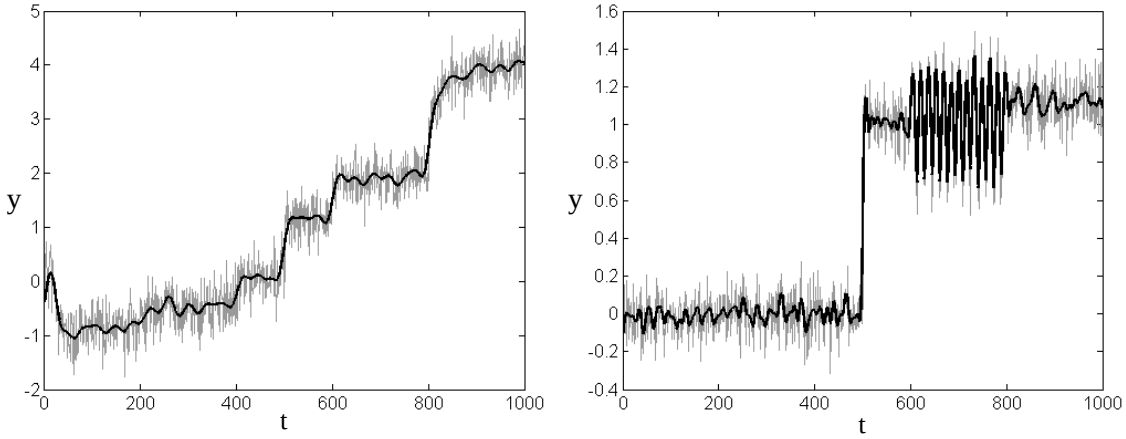


Figura 5.7: Valores elevados na SNR podem gerar oscilações indesejadas. Para compensar, α deve aumentar.

Nesse caso, então, foram utilizadas algumas regras heurísticas para escolher o valor de α com base na relação entre sinal e ruído. Aqui, esta relação será denominada *delta sinal-ruído* (DSR), definida como:

$$DSR = \frac{\max(y_{suav}) - \min(y_{suav})}{\max(ruído) - \min(ruído)} \tag{5.2}$$

onde $\max(k)$ e $\min(k)$ são os valores máximo e mínimo de k na janela, y_{suav} é o sinal suavizado e *ruído* são apenas os valores do ruído branco adicionado ao sinal na janela.

As heurísticas para estabelecer o valor de α foram:

- $\alpha = \sigma_r$, quando DSR ≤ 1, onde σ_r é o desvio padrão do ruído;
- $\alpha = 1{,}5 \times \sigma_r$, quando 1 < DSR ≤ 4;
- $\alpha = 3\%$ de δy^{jan} nos demais casos.

Alternativamente, para o caso intermediário (1 < DSR ≤ 4), podem-se utilizar outros procedimentos quando da etapa de suavização. Por exemplo, há métodos substitutos ao filtro *Butterworth* que resultam em sinais com menor quantidade de oscilações indesejadas. Maiores detalhes sobre outras técnicas de suavização/regularização podem ser obtidas no trabalho de Mejia *et al.* (2010). Outra forma de se proceder nessas circunstâncias é dividir a janela de dados, segregando as diferentes características dinâmicas presentes, uma vez que elas se influenciam mutuamente durante processo de suavização adaptativo.

A Tabela 5.3 resume os resultados obtidos para análises descritas acima. Para cada um dos critérios, RC e ER, foi calculada a média dos valores obtidos para cada nível de ruído, bem como a média contemplando todos os níveis juntos (células sombreadas da tabela). Percebe-se que o método Evolutivo apresentou desempenho superior em ambos os critérios, com maiores diferenças observadas no ER, sendo mais evidente para o ruído de magnitude 0,3.

Tabela 5.3: RC e ER para 6 mil sinais gerados e comprimidos com os métodos Refinado e Evolutivo.

Método	Magnitude do ruído	CRC (%) (média de 1000 sinais)	ER (média de 1000 sinais)
Refinado	0	95,41	685,03
	0,01	95,16	696,06
	0,05	94,97	687,88
	0,1	95,56	688,44
	0,3	97,16	794,85
	0,5	97,76	774,78
	média	96,00	721,17
Evolutivo	0	96,31	685,45
	0,01	96,21	530,43
	0,05	95,95	530,39
	0,1	96,44	530,37
	0,3	97,57	531,02
	0,5	98,06	531,23
	média	97.76	530,65

Ainda, o Evolutivo apresentou pouca variação no ER com o aumento do ruído, e isso se deve ao próprio funcionamento do método. Sabe-se que a relação da metodologia com a magnitude do ruído dá-se somente pelo valor da tolerância especificada (parâmetro α). Como o algoritmo Evolutivo inicia com a interpolação dos valores extremos da janela, a localização dos pontos gravados na sequência varia pouco, mesmo que α mude com o ruído, pois o erro avaliado é sempre em relação ao sinal suavizado. Tal comportamento não se verifica no método Refinado, pois ele opera de forma sequencial, percorrendo a janela de dados. As alterações na magnitude do ruído provocam mudanças no sinal suavizado, e isto, juntamente com o parâmetro α , tem maior impacto na gravação dos dados por este método.

A análise dos métodos de compressão propostos em aplicações de identificação de modelos pode ser feita através dos resultados da Figura 5.8 a 5.10 e das Tabelas 5.4 e 5.5. Nos gráficos, é possível visualizar o resultado da compressão/descompressão dos sinais gerados por simulação (a) e das respostas ao degrau unitário dos modelos (b). Neste último, apresentam-se as respostas do modelo original (*G*), usado para gerar as saídas, e dos modelos identificados antes e após a compressão pelos métodos Refinado (*idRefin*) e Evolutivo (*idEvol*),os quais foram obtidos por procedimentos idênticos. Percebe-se que, visualmente, os modelos identificados apresentaram respostas ao degrau bastante semelhantes entre si. A Tabela 5.4 exibe os valores das integrais do erro absoluto (IEA) e do erro quadrático (IEQ) entre as respostas dos modelos identificados por ambos os métodos de compressão propostos, e daqueles obtidos pela identificação direta, a partir dos dados brutos (*idG*).

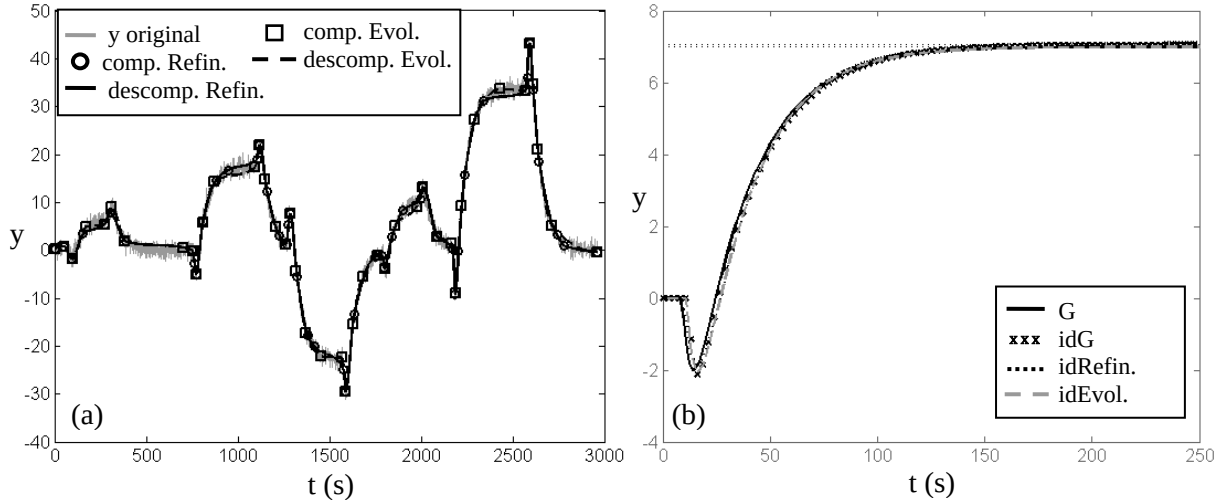


Figura 5.8: Compressão do SINALAa (a), e resposta a degrau unitário dos modelos identificados (b) comparando os métodos Refinado e Evolutivo (α = 3% de δy^{jan}); G é a planta original, e *id* representa os modelos identificados. A cor preta se refere ao método Refinado, e a cinza ao método Evolutivo.

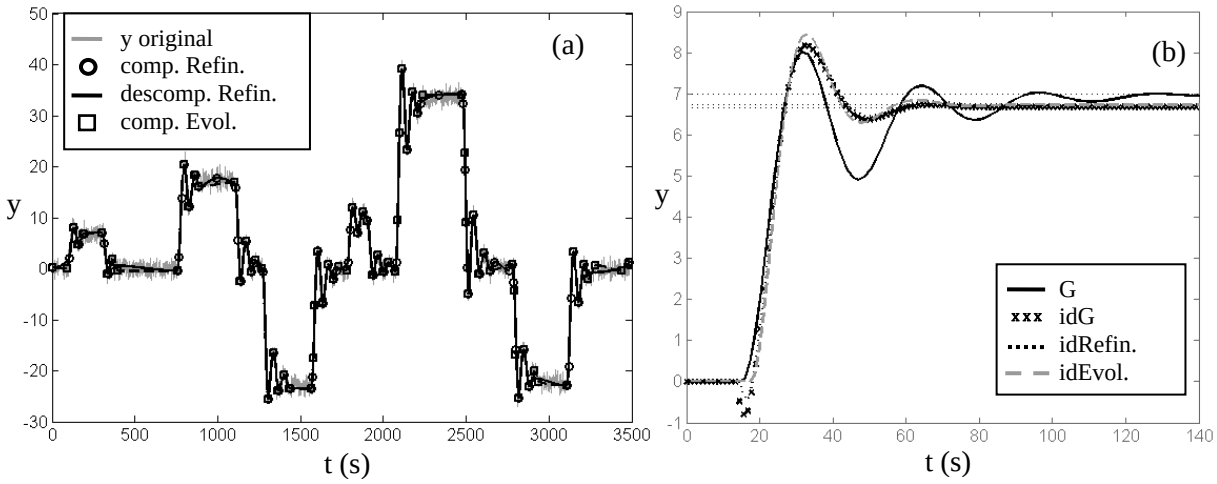


Figura 5.9: Compressão do SINALAb (a), e resposta a degrau unitário dos modelos identificados (b) comparando os métodos Refinado e Evolutivo (α = 2,5% de δy^{jan}); G é a planta original, e *id* representa os modelos identificados. A cor preta se refere ao método Refinado, e a cinza ao método Evolutivo.

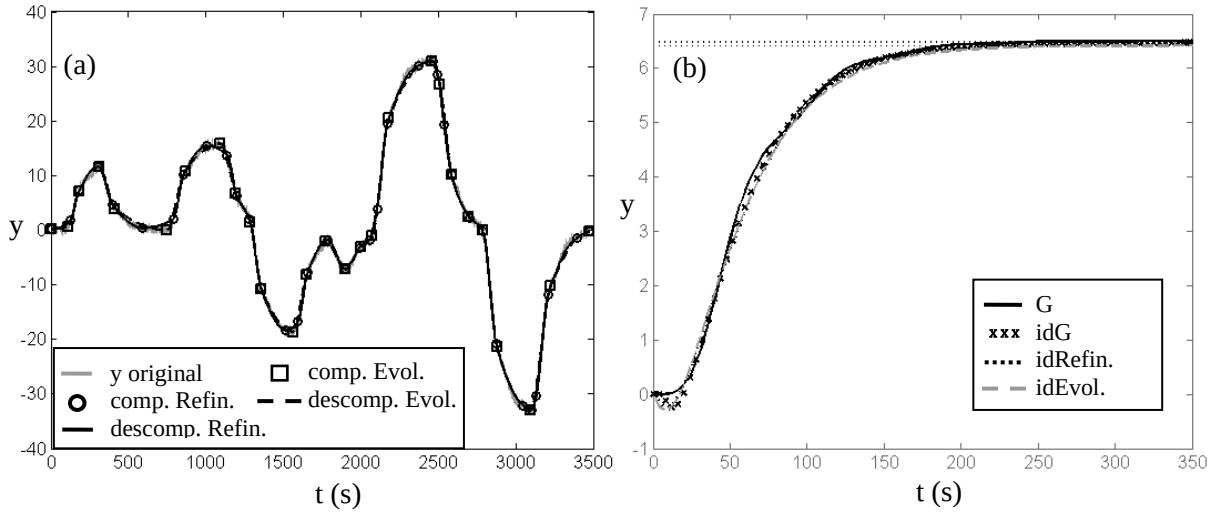


Figura 5.10: Compressão do SINALAc (a), e resposta a degrau unitário dos modelos identificados (b) comparando os métodos Refinado e Evolutivo (α = 2,5% de δy^{jan}); G é a planta original, e *id* representa os modelos identificados. A cor preta se refere ao método Refinado, e a cinza ao método Evolutivo.

Para o SINALAa, e o SINALAc observaram-se menores erros para a resposta do modelo identificado *idRefin*, enquanto para o SINALAb o modelo *idEvol* mostrou-se superior neste critério. Nos dois primeiros casos, o resultado pode ser devido à utilização da derivada como critério para gravar a informação no método Refinado, assim como o fato de a RC ser inferior a do Evolutivo, isto é, aquele método armazena mais pontos e, portanto, é mais fiel aos dados originais. No entanto, observando a Tabela 5.6, o método Evolutivo apresenta maior RC para o SINALAb, e forneceu um modelo identificado com menor valor tanto na IEA quanto na IEQ, ainda que esse modelo possua um sobressalto levemente superior ao modelo *idRefin*. Uma explicação possível é o menor erro obtido entre os ganhos do modelo *idG* e *idEvol* (0,15%, contra um erro de 1,12% para o *idRefin*).

Vale salientar ainda que, para o SINALAa, os modelos identificados com os dados descomprimidos pelo método Refinado apresentam-se mais próximos ao sistema original do que aqueles identificados sem comprimir os dados. A IEA entre *G* e *idG* foi de 2,73 (a IEQ foi de 7,47), enquanto que entre *G* e *idRefin* foi de 0,0715 (0,0051 para a IEQ).

O mais importante a se observar, contudo, é que, independente do método de compressão, é possível armazenar dados que, posteriormente, podem ser submetidos a procedimentos identificação de modelos e gerar resultados satisfatórios. Outros tipos de sistemas foram testados (porém não apresentados nessa Dissertação) e apresentaram resultados semelhantes aos discutidos aqui.

O método Refinado se sobressaiu em dois dos três casos estudados, porém as diferenças são pouco significativas em relação ao método Evolutivo. Esses erros podem ser diminuídos se outros procedimentos de identificação forem utilizados. É importante ressaltar que a comparação entre os métodos também deve ser feita considerando as finalidades na utilização dos modelos identificados. Em ajuste de controladores, por exemplo, a robustez e o desempenho atingidos com sintonias realizadas através desses modelos são possíveis critérios a serem avaliados.

Tabela 5.4: Integral do erro absoluto entre as respostas ao degrau unitário dos modelos identificados.

Sinal	$\int e\,dt$	
	$e = \lvert$ **idG - idRefin** $\rvert$	$e = \lvert$ **idG - idEvol** $\rvert$
SINALAa	2,31	6,59
SINALAb	3,47	1,66
SINALAc	12,63	17,14
	$\int e^2\,dt$	
	$e =$ **idG - idRefin**	$e =$ **idG - idEvol**
SINALAa	5,34	43,47
SINALAb	12,04	2,75
SINALAc	159,45	293,62

Tabela 5.5: Modelos das plantas originais simuladas e modelos obtidos na identificação.

SINALAa	G	$e^{-8s}\,\frac{-1{,}05s+0{,}07}{s^3+1{,}3s^2+0{,}31s+0{,}01}$
	idG	$e^{-12{,}3s}\,\frac{-1{,}436s+0{,}1039}{s^2+0{,}4346s+0{,}01472}$
	idRefin	$e^{-9{,}75s}\,\frac{-0{,}7509s+0{,}04935}{s^2+0{,}2119s+0{,}006999}$
	idEvol	$e^{-10{,}8s}\,\frac{-0{,}9899s+0{,}06498}{s^2+0{,}2725s+0{,}009296}$
SINALAb	G	$e^{-15s}\,\frac{-0{,}1867s+0{,}093}{s^3+0{,}1133s^2+0{,}0427s+0{,}0013}$
	idG	$e^{-13{,}8s}\,\frac{-0{,}8206s+0{,}2912}{s^2+0{,}1903s+0{,}04365}$
	idRefin	$e^{-14{,}8s}\,\frac{-0{,}5894s+0{,}2997}{s^2+0{,}1914s+0{,}04445}$
	idEvol	$e^{-17{,}8s}\,\frac{-0{,}02359s+0{,}3491}{s^2+0{,}1821s+0{,}05185}$
SINALAc	G	$e^{-s}\,\frac{-1{,}083\cdot10^{-4}s+7{,}2\cdot10^{-5}}{s^4+s^3+0{,}1067s^2+7{,}11\cdot10^{-4}s+1{,}11\cdot10^{-5}}$
	idG	$e^{-6{,}25s}\,\frac{-0{,}08981s+0{,}01103}{s^2+0{,}09048s+0{,}001703}$
	idRefin	$e^{-1{,}75s}\,\frac{-0{,}0753s+0{,}008104}{s^2+0{,}07108s+0{,}001262}$
	idEvol	$e^{-0{,}25s}\,\frac{-0{,}09108s+0{,}008629}{s^2+0{,}07701s+0{,}001344}$

Tabela 5.6: Resultados da compressão pelos métodos Refinado e Evolutivo dos sinais artificiais gerados para identificação.

	Método	CRC (%)	ER
SINALAa	Refinado	96,83	1,341
	Evolutivo	97,16	1,370
SINALAb	Refinado	95,18	1,126
	Evolutivo	95,75	1,081
SINALAc	Refinado	97,93	0,3505
	Evolutivo	98,45	0,4041

5.2.3 Sistemáticas para o ajuste do sistema PI

Os pontos utilizados como referência para as metodologias de ajuste automático dos parâmetros do sistema PI estão mostrados na Figura 5.11. Para o SINALAa e o SINALRd esses pontos são aqueles obtidos da compressão pelos métodos Evolutivo e Refinado, respectivamente; para o SINALAb e o SINALRa os pontos foram selecionados manualmente através da função `ginput` do MATLAB®. Essa função permite que, a partir da janela de dados, o usuário selecione alguns pontos em regiões do sinal que ele julga serem importantes para se armazenar.

Na Figura 5.12 visualizam-s os resultados gráficos da compressão pelos algoritmos do sistema PI, com os parâmetros sintonizados pelas metodologias propostas, para os sinais reais **a** e **d**, e na Figura 5.13, para os artificiais **a** e **b**. É possível ver que a maioria das tendências significativas dos sinais foram capturadas, apesar de não terem sido gravados exatamente os pontos de referência da Figura 5.11. Nos sinais simulados, algumas características particulares foram inseridas, como fase não-mínima e subamortecimento, e a compressão com os parâmetros ajustados foi capaz de manter essa informação.

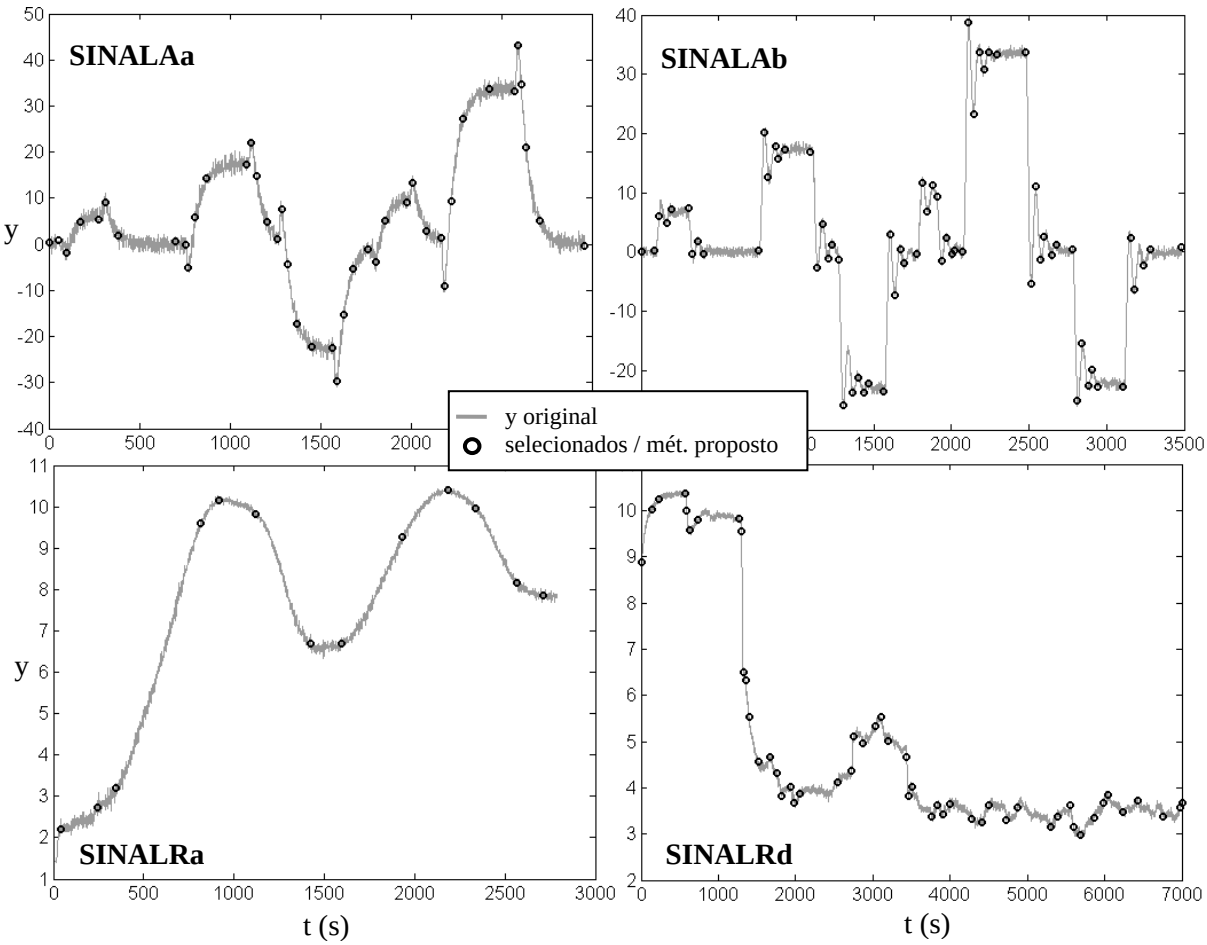


Figura 5.11: Pontos que servem como base para as metodologias de ajuste do PI.

A Tabela 5.7 mostra, para estes resultados, os valores dos parâmetros obtidos, assim como a RCR e o ER. Apesar de as metodologias serem distintas (por exemplo, no primeiro, a determinação do *ExcDev* é manual), alguns parâmetros apresentam valores próximos entre elas. Percebe-se também, pela tabela, que na maioria dos casos o método por otimização resultou na gravação de um maior número de pontos, isto é, tem menor RCR, porém apresenta, como consequência, menor ER. Ressalta-se, ainda, que este método apresenta um maior esforço computacional: por exemplo, para o SINALAa, com os pontos de referência da Figura 5.9, a solução foi obtida em 21,3 s, e para o SINALRd, em 70,2 s, com um processador Intel *Core2Duo* e 4 GB de memória RAM. Quanto maior a janela de dados, e maior o conjunto de pontos de referência, maior o tempo computacional requerido.

Ambos os métodos, portanto, puderam ajustar de forma satisfatória os parâmetros da compressão do PI e capturar as principais características dos sinais. O método por otimização mostrou-se levemente superior, apesar do maior tempo computacional exigido (lembrando que é necessário aplicar a técnica apenas uma vez para obter os parâmetros), além de sintonizar *ExcDev* e *CompDev* automaticamente, ao contrário do método anterior, que exige a sintonia manual de um dos parâmetros.

Outra análise interessante a ser feita é a comparação dos resultados obtidos através das propostas apresentadas com aqueles colhidos sem as sistemáticas de sintonia. No entanto,

como os estudos de caso não provêm de processos reais e não estavam previamente submetidos a uma compressão sem ajuste, tal comparação não se mostrava proveitosa. Essa contribuição, entretanto, será apresentada em outros trabalhos, tendo em vista que alguns dados de variáveis de processos serão cedidos por uma refinaria nacional.

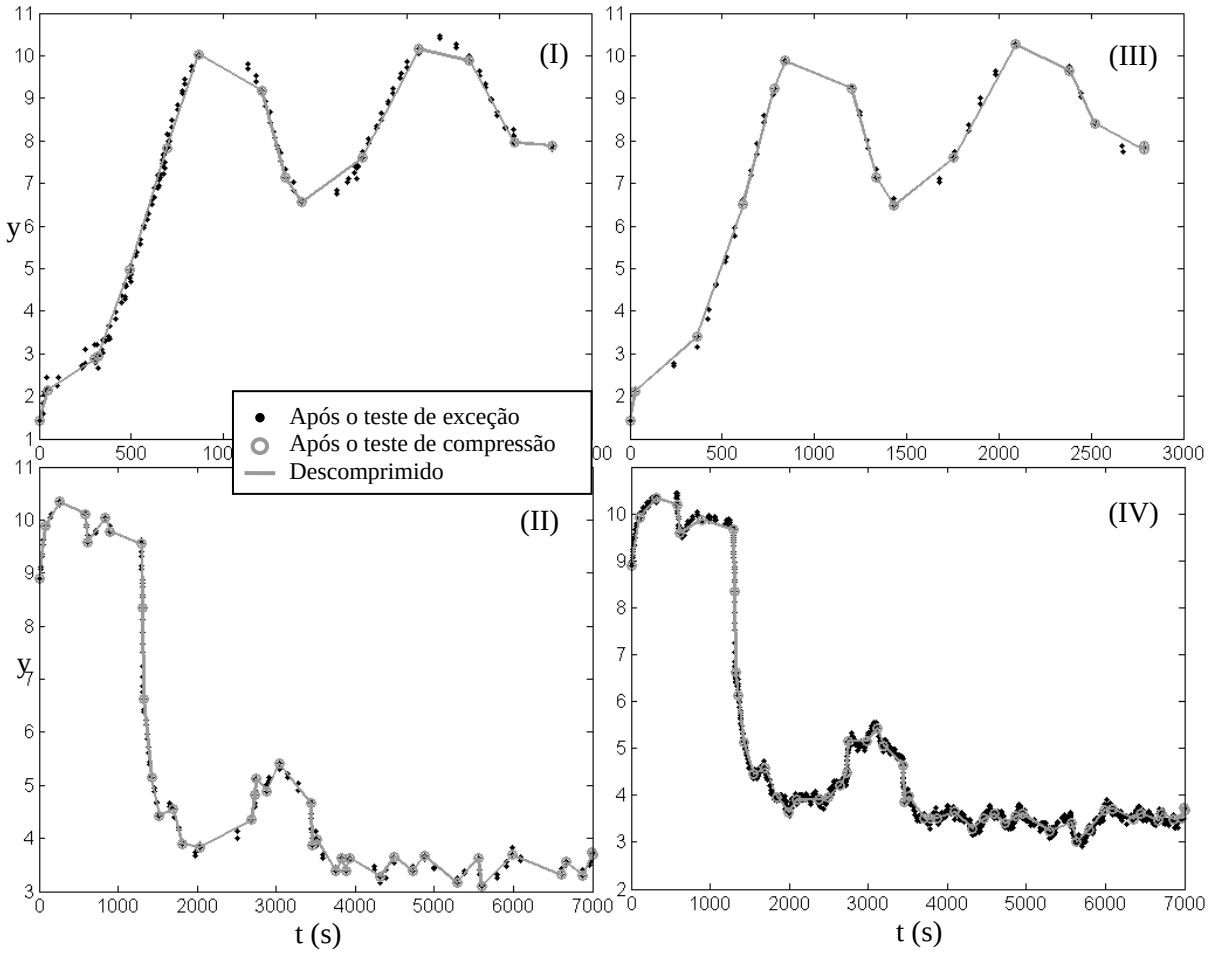


Figura 5.12: Resultado da compressão dos sinais reais. (I) e (II): com o *CompDev* do PI ajustado pela primeira abordagem; (III) e (IV): pelo método por otimização.

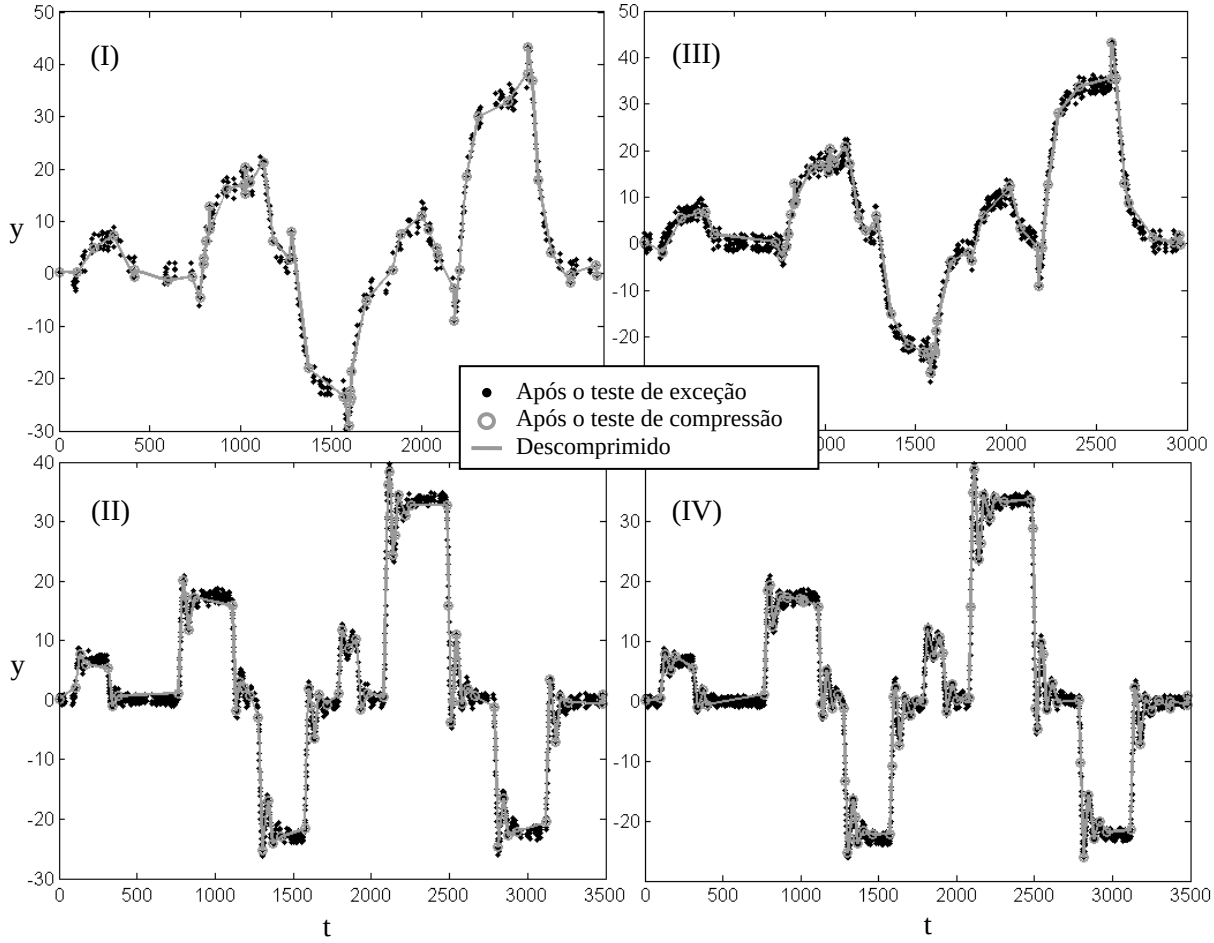


Figura 5.13: Resultado da compressão dos sinais artificiais. (I) e (II): com o *CompDev* do PI ajustado pela primeira abordagem e (III) e (IV): pelo método por otimização.

Tabela 5.7: Análise comparativa entre os métodos propostos (Mét) para ajuste da compressão do PI.

	Método	**ExcDev / CompDev**	**CRC (%)**	**ER**
SINALAa	Mét1: *CompDev*	2,3871 / 3,7097	96,42	2.1410
	Mét2: Otimiz.	1,7593 / 3,8580	96,02	2,1376
SINALAb	Mét1: *CompDev*	0,8220 / 2,7308	96,38	1,3318
	Mét2: Otimiz.	0,0926 / 2,0473	94,95	0,6723
SINALRa	Mét1: *CompDev*	0,2722 / 0,4422	98,92	0,0462
	Mét2: Otimiz.	0,6138 / 0,3464	98,92	0,0570
SINALRd	Mét1: *CompDev*	0,2228 / 0,1928	98,86	0,0143
	Mét2: Otimiz.	0,0858 / 0,2359	98,54	0,0087

Referências

Paim, A. d. C. *Controle Preditivo Retroalimentado por Estados Estimados*. Dissertação de Mestrado - Departamento de Engenharia Química, Porto Alegre, Rio Grande do Sul, 2009.

Mejia, R. I. G., R. C. Freitas & J. O. Trierweiler (2010). Derivatives Estimation Based on Smoothing Techniques. In *Bolivian Engineering and Technology Congress*. La Paz.

Capítulo 6

Conclusão e Trabalhos Futuros

Desde a implantação dos sistemas digitais para coletar, monitorar e controlar os processos na indústria houve um aumento considerável na quantidade de informação a ser gerenciada. Os primeiros algoritmos empregados para lidar com o elevado volume de dados proveniente do campo industrial visavam a comprimir dados para reduzir custos com espaço em disco, mas atualmente este tipo de recurso não é tão dispendioso. Ainda é importante economizar na aquisição e expansão de memória computacional, porém a exigência da compressão de dados voltou-se para a boa gerência e manipulação da informação, velocidade de recuperação e transferência, bem como a manutenção da essência e do conhecimento que dela possa se extrair. Nesse contexto também se pode destacar o crescimento das novas tecnologias descentralizadas em processos de produção, incluindo os sistemas *fieldbus* e as transmissões sem fio. Essas tecnologias também passam a ser importantes com as descobertas de novas bacias de petróleo no Brasil, e pela necessidade que as plataformas instaladas nesses locais têm no que diz respeito à comunicação e ao controle das operações *offshore*. Dessa forma, integrar a compressão de dados a esses sistemas acarretaria em maior eficiência operacional, minimizando a quantidade de informação transmitida.

No capítulo 2 ficaram evidentes os aspectos relativos compressão de dados no cenário da indústria de processos, as variedades de algoritmos existentes, bem como as carências que eles apresentam. Os métodos propostos nesta Dissertação (Refinado e Evolutivo) baseiam-se na compressão dos dados por janelas, a partir da suavização ou regularização de sinais ruidosos. Nesta etapa os filtros *Butterworth* com parâmetros adaptativos mostraram-se bastante adequados e computacionalmente mais eficientes. Em ambas as técnicas propostas, a descompressão é feita com a interpolação dos pontos gravados por *splines* cúbicas, que possuem características apropriadas para representar dados de processos. De modo geral, o método Evolutivo apresentou melhores resultados no compromisso entre espaço em disco exigido e erro na reconstrução, mas pode-se considerar que ambos os métodos tiveram desempenho comparável quando os dados comprimidos foram usados para identificar modelos.

Uma parte deste trabalho foi dedicada a propor sistemáticas para o ajuste dos parâmetros de compressão do sistema PI da OSIsoft®. O PI é um PIMS comercial implementado em muitas indústrias dedicado a coletar e organizar dados provenientes do "chão-de-fábrica". A compressão de dados nas rotinas do PI muitas vezes não apresenta resultados satisfatórios, pois não há um correto ajuste dos seus parâmetros. As técnicas propostas se valem de um conjunto prévio representativo das variáveis de processo, e, com base neles, ajustam-se os desvios de exceção e compressão. O método baseado em otimização mostrou desempenho levemente superior, porque considera explicitamente o compromisso entre o espaço em disco que os dados gravados ocupam e o erro na sua reconstrução. Sua rotina de resolução é mais dispendiosa computacionalmente, porém não é necessário aplicá-la mais de uma vez.

Há possibilidades para trabalhos futuros envolvendo as propostas desta Dissertação, como a utilização das ideias descritas visando também dados de entrada (variáveis manipuladas), abordando formulações alternativas para controladores preditivos (MPC) e novas técnicas para auditoria de desempenho. Também, existe a possibilidade de integrar os métodos de compressão de dados nos próprios sensores em aplicações com controle remoto. Outra sugestão a ser explorada é a proposição de técnicas de identificação de modelos que contemplem diretamente os dados comprimidos sem a necessidade de reconstruí-los.