Felipe Eliasquevici

Estudo comparativo de métodos de Compressão de Dados e Detecção de Danos em Monitoramento de Integridade Estrutural

Brasil

2018

Felipe Eliasquevici

Estudo comparativo de métodos de Compressão de Dados e Detecção de Danos em Monitoramento de Integridade Estrutural

Trabalho de Conclusão de Curso apresentado para obtenção do grau de Bacharel em Ciência da Computação.

UNIVERSIDADE FEDERAL DO PARÁ

INSTITUTO DE CIÊNCIAS EXATAS E NATURAIS

FACULDADE DE COMPUTAÇÃO

CURSO DE BACHARELADO EM CIÊNCIA DA COMPUTAÇÃO

Orientador: Prof. Dr. Claudomiro de Souza de Sales Junior

Brasil

2018

Felipe Eliasquevici
Estudo comparativo de métodos de Compressão de Dados e Detecção de Danos em Monitoramento de Integridade Estrutural/ Felipe Eliasquevici. – Brasil, 2018-
56 p. : il. (algumas color.) ; 30 cm.

Orientador: Prof. Dr. Claudomiro de Souza de Sales Junior

Trabalho de Conclusão de Curso – UNIVERSIDADE FEDERAL DO PARÁ
INSTITUTO DE CIÊNCIAS EXATAS E NATURAIS
FACULDADE DE COMPUTAÇÃO
CURSO DE BACHARELADO EM CIÊNCIA DA COMPUTAÇÃO, 2018.

1. SHM. 2. Compressão. 2. Deteção de Danos. I. Prof. Dr. Caludomiro de Souza Sales Junior. II. Universidade Federal do Pará. III. Faculdade de computação. IV. Estudo comparativo de métodos de Compressão de Dados e Detecção de Danos em Monitoramento de Integridade Estrutural

Felipe Eliasquevici

Estudo comparativo de métodos de Compressão de Dados e Detecção de Danos em Monitoramento de Integridade Estrutural

Trabalho de Conclusão de Curso apresentado para obtenção do grau de Bacharel em Ciência da Computação.

Trabalho aprovado. Belém, 23 de fevereiro de 2018:

Prof. Dr. Claudomiro de Souza de Sales Junior
Orientador

Regiane Silva Kawasaki Frances
Convidado 1

Mauro Rodrigo Larrat Frota e Silva
Convidado 2

Brasil
2018

Agradecimentos

Agradeço à minha família por tudo que sempre me proporcionaram e por estarem sempre comigo, aos meus pais Roberto e Marianne, minhas irmãs Carolinne e Rafaela em especial. Agradeço também a todas as pessoas que me ajudaram a escrever esse trabalho, me tiraram dúvidas e contribuíram na menor coisa que seja, em especial meu orientador Claudomiro e o professor Mauro Larrat. Sou grato também a todos os meus amigos que fazem parte da minha vida e a tornam melhor. Por último agradeço a todos os professor que fizeram parte dessa minha trajetória até aqui, do ensino infantil até hoje, pela sua contribuição no meu conhecimento e sempre me impulsionarem pra frente.

Resumo

O Monitoramento de Integridade Estrutural (SHM) é um processo que objetiva detectar danos em estruturas através de sensores instalados na mesma que captam dados e fazem quantização de dano em tempo real. Um número grande de sensores coletando informações em um intervalos curtos de tempo podem gerar uma quantidade muito grande de dados, que precisam ser transmitidos e armazenados. A compressão de dados torna-se essencial então para esses sistemas. Durante o processo de compressão, entretanto, características importantes dos sinais gerados pelos sensores podem ser perdidas, e a detecção de danos pode ficar comprometida. Nesse contexto, este trabalho propõe um estudo comparativo entre combinações de métodos de compressão de dados e técnicas de detecção de danos. sobre os dados comprimidos. Para mensurar a eficiência dessas combinações utiliza-se uma métrica paramétrica, a qual quantifica o quão boa uma combinação é em uma única medida. Apos comparados os resultados foi observado que as combinações envolvendo o algoritmo *Chebshevy Approximation* geraram os melhores resultados, especialmente com os algoritmos *Fuzy-C-Means* e *K-Means*.

Palavras-chave: SHM. Compressão de Dados. Detecção de Danos. Método de Avaliação. Clusterização.

Abstract

Structural Health Monitoring (SHM) is a process that aims to detect damage in structures through sensors attached on this structure that capture the data and quantify the damage in real time. A big number of sensors capturing information in short time intervals can generate a huge volume of data, that needs to be transmitted and stored. Data compression then becomes essential to these kinds of system. In the compression process, however, essential characteristics of the signals generated by the sensors can be lost, and the damage detection could be compromised. Considering that, this study propose to compare several combinations of data compression methods and damage detection techniques. To measure the efficiency of those combinations it is utilized a parametric metric, which quantifies how good a combination is in a single measure. After comparing the results it was observed that the algorithm Chebshevy Approximation generate the best results, specially when combined with Fuzzy-C-Means and K-Means.

Keywords: SHM. Data Compression. Damage Detection. Evaluation Method. Clustering.

Lista de ilustrações

Figura 1 – Framework de testes em SHM que inclui compressão de dados. 20
Figura 2 – Exemplo das etapas da aplicação do PCA com janela de tamanho 3. . . 26
Figura 3 – Exemplo das etapas da aplicação do APCA. 27
Figura 4 – Exemplo das etapas de aplicação do PWLH. 29
Figura 5 – Mecanismo de filtragem do Slide Filter. 30
Figura 6 – Exemplo de aplicação do Slide Filter. 31
Figura 7 – Exemplo de aplicação do Chebyshev Approximation. 33
Figura 8 – Exemplo de aplicação do K-Means. 35
Figura 9 – Exemplo de aplicação do GMEANS. 37
Figura 10 – Exemplo de clusterização por densidade. 37
Figura 11 – Exemplo de ponto de núcleo, borda e ruído para o DBSCAN. 38
Figura 12 – Exemplo de função de pertinência em um cluster hard (B) e em um cluster soft (C) aplicados ao mesmo conjunto de dados (A). 39
Figura 13 – Ilustração da z24 43
Figura 14 – Frequencias naturais extraidas do sensor 5 nas 234 amostras no horario de 12h 45
Figura 15 – Taxas de compressão dos algoritmos 46
Figura 16 – Média de erros Tipo I e II utilizando PWLH para compressão e o algoritmo AP para detecção de danos. 47
Figura 17 – Média de erros Tipo I e II utilizando CHEB para compressão e o algoritmo GMM para detecção de danos. 48
Figura 18 – Média de erros Tipo I e II utilizando CHEB para compressão e o algoritmo AP para detecção de danos. 48
Figura 19 – Média de erros Tipo I e II utilizando CHEB para compressão e o algoritmo K-Means para detecção de danos. 49
Figura 20 – Média de erros Tipo I e II utilizando CHEB para compressão e o algoritmo FCM para detecção de danos. 49
Figura 21 – Média de erros Tipo I e II utilizando CHEB para compressão e o algoritmo DBSCAN para detecção de danos. 50
Figura 22 – Valores de $S(k) > 0,75$ utilizando CHEB para compressão. 51

Lista de tabelas

Tabela 1 – Dez maiores $S(K)$ entre as combinações testadas. 50
Tabela 2 – Quantidade de valores de k para qual cada combinação possui $S(k) > 0,75$. 51

Lista de abreviaturas e siglas

AANN	*Auto-associative Neural Network*
AP	*Affinity Propagation*
APCA	*Adaptive Piecewise Constant Approximation*
CHEB	*Chebyshev Approximation*
CHEB	*Density Based Spatial Clustering of Applications with Noise*
FA	*Factor Analysis*
FCM	*Fuzzy-C-Means*
FN	Falso Negativo
FP	Falso Positivo
GAMPS	*Grouping and Amplitude Scaling*
GMM	*Gaussian Mixture Model*
MSD	*Mahalanobis Distance*
NLPCA	*Nonlinear Principal Component Analysis*
P	Precisão
PCA	*Piecewise Constant Approximation*
PCA-D	*Principal Component Analysis*
PWLH	*Piecewise Linear Histogram*
R	Sensitividade – *Recall*
RMSD	*Root-mean Square-deviation*
SF	*Slide Filters*
SHM	Monitoramento de Integridade Estrutural - *Structural Health Monitoring*
SOM	*Self Organizing Map*
SVD	*Singular Value Decomposition*
VN	*Verdadeiro Negativo*
VP	*Verdadeiro Positivo*

Lista de símbolos

$F(k)$	*F-Measure*
$S(K)$	*S-Measure*
$T(K)$	*Taxa de compressão*
ω	Peso do método de avaliação
β	Peso da *F-Measure*
w	Janela de compressão
$\sigma(f)$	Desvio padrão da base de dados

Sumário

1 INTRODUÇÃO 13
1.1 Trabalhos Relacionados 14
1.2 Motivação 15
1.3 Justificativa 16
1.4 Objetivo 16
1.4.1 Objetivo Geral 16
1.4.2 Objetivos Específicos 16
1.5 Metodologia 17
1.6 Organização 17

2 MONITORAMENTO DE INTEGRIDADE ESTRUTURAL 19
2.1 Aquisição de dados 20
2.2 Definição de Parâmetros 21
2.3 Compressão e Validação 21
2.4 Extração de características 22
2.5 Detecção de danos 22
2.6 Análise de resultados 23

3 TÉCNICAS DE COMPRESSÃO DE DADOS 25
3.1 Piecewise Constant Approximation 25
3.2 Adaptive Piecewise Constant Approximation 27
3.3 Piecewise Linear Histogram 28
3.4 Slide Filter 29
3.5 Chebyshev Approximation 32

4 DETECÇÃO DE DANOS 34
4.1 K-Means 35
4.2 G-Means 36
4.3 DBSCAN 37
4.4 Fuzzy-C-Means 39
4.5 Gaussian Mixture Models 40
4.6 Affinity Propagation 41

5 METODOLOGIA DE TESTES 43
5.1 Aquisição de Dados 43
5.2 Definição de parâmetros 44

5.3 Extração de características 44

6 RESULTADOS 46
6.1 Compressão de Dados 46
6.2 Quantidade média de erros 47
6.3 Resultados da *S-Measure* 50
6.4 Análise dos resultados 52

7 CONSIDERAÇÕES FINAIS 53
7.1 Trabalhos Futuros 53

REFERÊNCIAS 54

1 Introdução

Nos últimos 20 anos foram construídas uma grande quantidade de estruturas civis. Manter suas operações seguras e confiáveis torna-se crucial para garantir e proteger os investimentos realizados, bem como assegurar o bem-estar das pessoas que são afetadas por essas construções (LI; YI; QIAO, 2017). Além disso, em termos de design e sistemas de engenharia, tem-se visto o uso de muitos materiais novos cujo o processo de degradação a longo prazo não é muito conhecido (FARRAR; WORDEN, 2012).

Nesse contexto entra o paradigma de Monitoramento de Integridade Estrutural (*Structural Health Monitoring* - SHM) como processo de monitoramento dessas estruturas. Esse processo permite identificar danos nas estruturas de forma autônoma (FIGUEIREDO; MOLDOVAN; MARQUES, 2006). O monitoramento é dado com sensores ligados na estrutura que coletam informações sobre o estado atual da mesma. Os dados coletados são transformados em características sensíveis a dano (SANTOS et al., 2016). Para realiar essa detecção de danos existem vários métodos e algoritmos de aprendizado de máquina que podem ser aplicados nos dados. Esses métodos podem analisar vários tipos de dados como vibração e pressão para criar modelos estatísticos de deteção de dano (FIGUEIREDO et al., 2011).

Visto que os sensores muitas vezes não tem capacidade de processar os dados, é necessário enviar eles pela rede para o local onde eles são processados. Um número grande de sensores pode sobrecarregar a rede pela qual os dados são enviados, assim como os bancos de dados que armazenam os mesmos (MADDEN, 2010). Por esse motivo é muito comum nos sistemas atuais de SHM incorporar métodos de compressão de dados (LIMA et al., 2016).

Na etapa de compressão, para poder diminuir a quantidade de informação armazenada, são aplicadas técnicas que transformam o sinal original em uma versão aproximada, perdendo parte de sua informação (SALOMON, 2006). Essa etapa tem como objetivo diminuir o máximo possível os dados sem perder características relevantes do sinal (ELMELEEGY et al., 2009).

Os sinais aproximados passam então por uma transformação para que dele sejam extraidas caracteristicas que representem melhor o estado da estrutura (FARRAR; WORDEN, 2012). Os algoritmos de detecção dano tem a função de, dado um conjunto de caractersticas do estado de uma estrutura, quantificar um indicador de dano. Se o indicador de dano for maior que um limiar a estrutura é classificada como com dano.

Quanto mais se comprime os dados, mais eficiente é a transferência e o armazenamento dele. Entretanto, é possível que muitas características significativas do sinal sejam

perdidas e a etapa de detecção de danos fique comprometida. Torna-se necessário então fazer um balanceamento entre ambos os aspectos (LIMA et al., 2016).

1.1 Trabalhos Relacionados

Existem vários trabalhos na área de SHM que combinam compressão e detecção de dano. O estudo de

(PARK et al., 2008) fez testes com PCA-D com a técnica de detecção de danos *Root-mean Square Deviation* (RMSD) e *K-Means*. Os estudos foram realizados com sensores de impedância para identificar parafusos soltos em uma estrutura de alumínio. Se chegou a conclusão que o PCA-D consegue eliminar ruídos indesejáveis dos sinais dos sensores e a classificação de dano com o PCA-D é mais eficiente do que sem. Não é um estudo comparativo robusto de um número grande de técnicas e os resultados não são comparados com outros estudos por usar dados de uma estrutura proprietária.

Outro estudo (TIBADUIZA; MUJICA; RODELLAR, 2013) utilizou o método de compressão *Principal Component Analysis* (PCA-D) e *Self Organizing Map* (SOM) para detecção de danos em uma estrutura artificial. O resultado do estudo concluiu que era possível utilizar essas técnicas para achar danos e suas localizações. Esse estudo testa somente um algoritmo de detecção de dano e um algoritmo de compressão de dados, não existe uma análise comparativa com outros resultados.

Dentro dos estudos comparativos, foi feito um estudo (FIGUEIREDO et al., 2014) comparando as técnicas de detecção de dano *Auto-associative Neural Network* (AANN), *Factor Analysis* (FA), *Mahalanobis Distance* (MSD) e *Singular Value Decomposition* (SVD). Foi utilizado para testes sinais coletados de uma estrutura artificial com danos simulados. AANN e MSD foram os melhores para identificar erros Tipo II enquanto os outros dois foram melhores para identificar erros Tipo I. Esse estudo faz uma comparação entre varias técnicas sofisticada sde deteção de dano, porém não possui uma etapa de compressão de dados.

Ainda dentro dos estudos comparativos, o estudo de (HUNG; JEUNG; ABERER, 2013) testou a eficiência de algoritmos de compressão em dados reais de sinais de sensores. Tais sensores coletavam dados diversos como temperatura, umidade, radiação e pressão. Foi utilizado no estudo *Piecewise Constant Approximation* (PCA), *Adaptive Piecewise Constant Approximation* (APCA), *Piecewise Linear Histogram* (PWLH), *Slide Filters* (SF), *Chebyshev Approximation* (CHEB) e *Grouping and Amplitude Scaling* (GAMPS). O resultado dos estudos foi que os algoritmos APCA e SF são os mais eficazes para comprimir os dados e para transmitir os dados, principalmente para sinais de baixa flutuação. Em relação a taxa de erro de aproximação, os algoritmos PCA e CHEB se mostraram mais eficientes em reconstruir o sinal sem perdas. Esse estudo apesar de fazer uma comparação

de vários métodos de compressão de dados não inclui uma etapa de detecção de danos, logo não estuda o efeito dos sinais aproximados nessa etapa.

Foi feito então por (LIMA et al., 2016), um estudo que utilizava um framework desenvolvido para avaliar de maneira conjunta detecção de dano e compressão de dados. Fazendo a compressão dos dados com os algoritmos PCA, APCA, PWLH, SF e CHEB e testando a deteção de danos com FA, SVD, Nonlinear Principal Component Analysis (NLPCA), PCA-D e MSD . O estudo chegou à conclusão que o PWLH com o NLPCA foi de forma geral o melhor para limiares de compressão baixos, enquanto para um limiar de compressão mais alto o APCA com o PCA-D obteve a melhor perfomance.

Esse estudo, entretanto ainda avaliava as duas etapas de com métricas diferentes. Para resolver isso foi feito um estudo (DANIN, 2016) que utilizava somente uma métrica para medir a eficiência da combinação de compressão com detecção de danos. No estudo foi utilizado PCA, APCA, PWLH, SF e CHEB para compressão e NLPCA para deteção de danos. Foi concluído que os algoritmos APCA e PWLH são melhores de forma geral, principalmente quando a compressão é mais importante. No estudo entretanto foi utilizada somente uma técnica de detecção de danos.

1.2 Motivação

Existe hoje uma maior demanda por sistemas SHM, visto que são sistemas que conseguem identificar quase em tempo real falhas em estruturas. Identificando essas falhas antes, é possível prevenir que ocorram desastres que acarretem em grandes prejuízos financeiros e principalmente perdas de vidas humanas (FARRAR; WORDEN, 2012).

O sistema SHM envolve uma rede de sensores que monitoram a estrutura em tempo real. Quanto mais sensores, mais precisos os resultados, ou seja, melhor a capacidade de identificar os danos na estrutura e possivelmente sua localização. Entretanto mais sensores também aumenta o volume de dados gerados pelo sistema (ZHOU; JIN; ZHENG, 2015).

Compressão se torna então importante para diminuir o volume de dados gerados que serão transmitidos, armazenados e processados. Por esse motivo alguns trabalhos recentes (LIMA et al., 2016) (DANIN, 2016) (HUNG; JEUNG; ABERER, 2013) já vem tratando a compressão de dados como uma etapa do processo de SHM.

Nesse contexto se torna importante fazer um método de avaliação de SHM com essa etapa inclusa, como feito em (DANIN, 2016), e testar uma quantidade maior de técnicas e algoritmos para validar esses resultados.

1.3 Justificativa

Esse trabalho surge nesse contexto de avaliar ambas as etapas de compressão de dados e detecção de danos em uma única métrica, testando outras técnicas de detecção de dano de clusterização em conjunto com algoritmos de compressão para tentar achar as combinações mais eficientes.

Foram utilizados os algoritmos PCA, APCA, PWLH, SF e CHEB para testar a compressão com o objetivo de ter um resultado mais facilmente comparado com o estudo original e ainda estarem entre os mais relevantes na literatura.

Para a etapa de detecção de danos a escolha de algoritmo se deu por serem métodos de clusterização simples e bastantes conhecidos e eficientes na literatura de aprendizado de máquina, e serem técnicas com resultados bons em testes de SHM. Não foi utilizado nenhuma técnica dessa categoria nos estudos da framework (DANIN, 2016). Essas ténicas variam de clusterização em centroides e por densidade, *soft* e *hard* e parametricas e não parametricas. Com toda essa variedade se espera que sejam encontradas mais combinações que possam ser usadas em SHM.

1.4 Objetivo

1.4.1 Objetivo Geral

O objetivo desse trabalho é estender a análise do método proposto em (DANIN, 2016) considerando outras técnicas de detecção de danos. Os resultados permitirão definir quais os melhores resultados dos algoritmos de clusterizição, com compressão de dados, utilizando uma métrica genérica.

1.4.2 Objetivos Específicos

- Revisão e estudo da literatura de SHM, tanto na área de compressão de dados quanto na área de detecção de danos;
- Implementar as técnicas de compressão de dados PCA, APCA, PWLH, SF e CHEB;
- Implementar técnicas de aprendizado de máquina *K-Means*, *G-Means*, *Fuzzy-C-Means* (FCM), *Gaussian Mixture Model* (GMM), *Density Based Spatial Clustering of Applications with Noise* (DBSCAN) e *Affinity Propagation* (AP) ;
- Avaliar as técnicas selecionadas quanto a compressão e sua consequência na detecção de dano.

1.5 Metodologia

Esse trabalho iniciou com um estudo e revisão da literatura atual de SHM, focando tanto em técnicas de compressão de dados utilizadas na área como técnicas de detecção de danos. Além disso foi pesquisado na literatura de compressão de dados novas técnicas utilizadas em compressão de sinais, bem como na literatura de aprendizado de máquina novas técnicas que pudessem ser utilizadas para detecção de danos em SHM. Com base nesse estudo foi possível entender o atual estado da arte e escolher as técnicas mais relevantes para esse trabalho.

Para realizar os testes foram utilizados os dados extraídos da ponte Z24 na Suíça. Foram coletados os dados do sensor número 5, extraídos as 12h da tarde, totalizando 234 amostras. O sensor número 5 é segundo (DANIN, 2016) o que melhor representa as frequências naturais da ponte. O horário é o que apresente a maior variabilidade da temperatura. Essas amostras foram comprimidas e descomprimidas por cada um dos métodos avaliados com diferentes limiares de compressão e foi salvo a taxa de compressão delas.

Com cada um dos sinais aproximados por cada técnica para cada limiar de compressão foi dividido para treino e testes. O treino foi realizado com as amostras coletadas nos primeiros 158, todas sem dano. Os testes foram realizados utilizando toda as 234 amostras, sendo as 197 primeiras sem dano e o restante com dano.

Foram aplicados os algoritmos de aprendizado de máquina de clusterização selecionados para os testes. Para cada teste foi calculado os centros de cada cluster com as amostras de treino e calculado a distância euclidiana de cada amostra para o centro para encontrar o indicador de dano (DI) da mesma. O DI da que ocupa a posição de 95% dos DIs de treino ordenados é escolhido como o limiar de dano. As amostras de teste que tiverem um valor de DI menor que o limiar de dano são consideradas como sem dano, enquanto as outras são consideradas com dano.

Para cada combinação de técnica de compressão, limiar de compressão e método de detecção de dano, foi calculado o número de falsos positivos e falsos negativos. Utilizando os dados das quantidade de erros foi calculado a *F-Measure*, métrica que avalia uma combinação em um valor de 0 a 1 com base na Precisão (P) e Sensitividade (R) da mesma. As métricas P e R estão relacionadas ao número de falsos positívos e falsos negativos, respectivamente. Com esses dados foi calculado as *S-Measure* de cada combinação, métrica que avalia uma combinação com base na *F-Measure* e taxa de compressão.

1.6 Organização

O presente trabalho está organizado da seguinte forma:

- Capítulo 2: Monitoramento de Integridade Estrutural

  Esse capítulo faz uma explicação breve sobre SHM e sobre as etapas relacionadas ao processo e suas importâncias.

- Capítulo 3: Técnicas de Compressão de Dados

  Aqui são feitas explicações gerais relevantes de compressão de dados, assim como é explicado o funcionamento de cada um dos algoritmos utilizados nesse trabalho.

- Capítulo 4: Detecção de Dano

  É discutido o processo de detecção de dano em SHM e explicado todas as técnicas utilizadas nesse trabalho.

- Capítulo 5: Metodologia de Testes

  É explicado como foram feitos os testes, a base de dados utilizada e os parâmetros escolhidos para cada uma das etapas.

- Capítulo 6: Resultados

  Apresenta os resultados obtidos nos testes e faz uma análise comparativa entre dados obtidos.

- Capítulo 7: Considerações finais

  São feitas as considerações finais sobre o trabalho e discutido os possíveis futuros trabalhos que podem surgir a partir desse.

2 Monitoramento de Integridade Estrutural

A nossa sociedade depende hoje cada vez mais de um número maior de estruturas, como pontes, barragens, edifícios e portos, para seu pleno funcionamento (FARRAR; WORDEN, 2012). Manter essas grandes estruturas funcionando é muito importante para o funcionamento econômico de várias operações, além de garantir a segurança pública de pessoas envolvidas com essas estruturas (LI; YI; QIAO, 2017).

Em um ambiente real, essas estruturas estão expostas a diversos fatores naturais, como fatores climáticos, umidade, chuvas e desastres. Além de fatores naturais, os fatores operacionais, como o uso humano, que causa, por exemplo, colisões e perfurações, também interferem na integridade da estrutura (GLISIC; INAUDI, 2008).

Dado esses fatores as estruturas estão sujeitas a desgastes que podem acarretar em danos para as estruturas. Esses danos podem ser causados por mudanças nas propriedades dos materiais, mudanças nas propriedades geométricas da estrutura, mudanças nas condições de contorno e mudanças nas condições de conectividade (FARRAR; WORDEN, 2012).

Nem todos os danos são facilmente identificados por vistorias manuais, causando que alguns danos fiquem anos sem serem detectados (FARRAR; WORDEN, 2012). Se os danos chegarem a um estado crítico é possível que o resultado seja um colapso da estrutura, gerando impactos financeiros e riscos a vida humanas e ambientais (PARK et al., 2008).

Atendendo a essas necessidades, surgiu um processo de monitoramento de estruturas que objetiva identificar em tempo real danos que afetem sua operabilidade. Esse processo é denominado monitoramento de integridade estrutural (SHM) (FIGUEIREDO; MOLDOVAN; MARQUES, 2006).

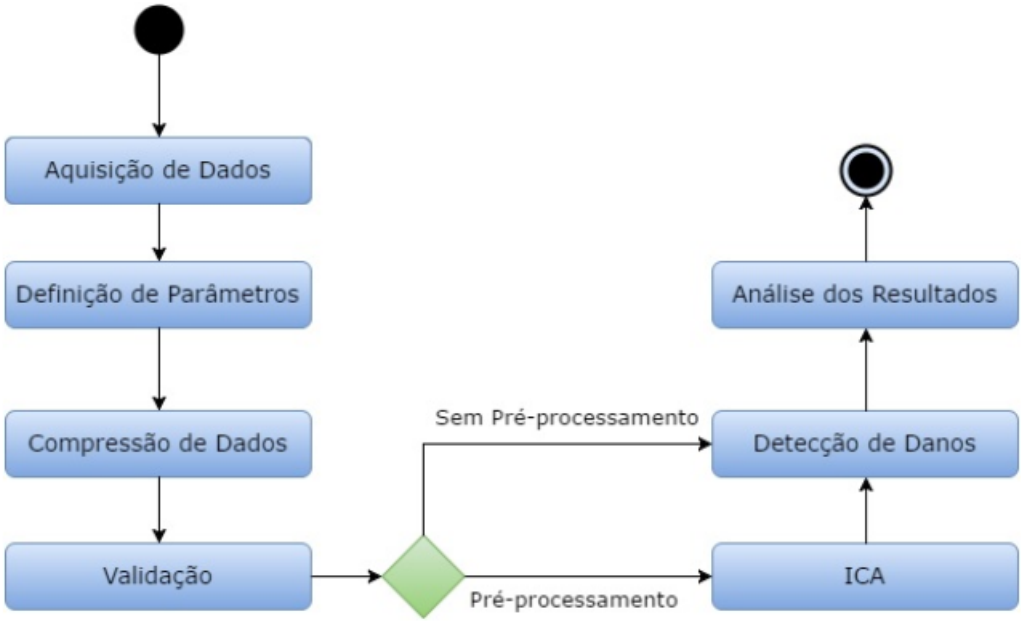
Um típico sistema de SHM consiste em uma rede de sensores instalados fisicamente na estrutura e ligados a um computador central que processa esses sinais. Os sensores podem medir diversas informações da estrutura, como tensão, deslocamento e acelerações. Esses dados são processados por algoritmos de detecção de danos para avaliar o estado atual da estrutura (FIGUEIREDO; MOLDOVAN; MARQUES, 2006).

A literatura SHM comumente aborda esse processo com um problema de reconhecimento de padrões estatísticos (FARRAR; DOEBLING; NIX, 2001). Podemos assim identificar quatro etapas para o processo: Avaliação operacional, aquisição de dados, extração de características e modelagem estatística.

Como visto anteriormente, em um sistema SHM geralmente é necessário fazer compressão de dados antes de fazer detecção de danos, por esse motivo foi criado um

framework que engloba ambos esses aspectos (LIMA et al., 2016).

Figura 1 – Framework de testes em SHM que inclui compressão de dados.



Fonte: (LIMA et al., 2016) Adaptado.

Na Figura 1 podemos ver as etapas do processo: aquisição de dados, definição de parâmetros, compressão de dados, validação, extração de características, detecção de danos e análise dos resultados. Essas etapas serão explicadas a seguir.

2.1 Aquisição de dados

Essa etapa corresponde em acoplar sensores a estruturas físicas para medir a resposta das mesmas as condições do sistema. A escolha dos sensores, quantos e onde eles serão posicionados na estrutura, assim como a frequência com a qual eles vão fazer as leituras e a transmissão desses dados também são tarefas feitas nessa fase (FIGUEIREDO; MOLDOVAN; MARQUES, 2006).

É necessário fazer uma análise prévia da estrutura para poder identificar que tipo de informação é relevante ser medida. Fatores como os materiais usados na construção, condições climáticas e uso da estrutura interferem nos dados coletados e devem ser levados em conta na hora de escolher os sensores.

O objetivo da aquisição de dados é fazer somente as medições, os sensores não tem responsabilidade pela realização da detecção de danos. Esses dados serão usados posteriormente para gerar informação necessária sobre a existência de danos ou não (FIGUEIREDO; MOLDOVAN; MARQUES, 2006).

2.2 Definição de Parâmetros

Nesta etapa todas as entradas de algoritmos e variáveis de funções que não estão relacionadas aos dados são definidas, tais como tamanhos de janelas de compressão, número de clusters, entre outros (LIMA et al., 2016).

O importante dessa etapa é ter entendimento das técnicas utilizadas e quais os parâmetros ideais para a situação em que elas estão sendo aplicadas. Essa etapa pode ser feita por meio de vários testes com parâmetros distintos para tentar encontrar o mais adequado.

2.3 Compressão e Validação

A etapa de compressão de dados é responsável por comprimir dados originais usando as técnicas selecionadas (LIMA et al., 2016). Essas técnicas geralmente funcionam dividindo os dados em várias janelas, em que para cada janela sãp usadas funções para aproximar os dados contidos nela gerando um sinal aproximado. Para manter a qualidade das informações e garantir que não exista muitas perdas no sinal aproximado, as aproximações seguem um limiar de erro, definido na etapa anterior, chamado de limiar de compressão e denominado por ε (HUNG; JEUNG; ABERER, 2013).

A escolha desse limiar de erro, nesse trabalho, é feita segundo a fórmula

$$\varepsilon = k * \sigma(f) \tag{2.1}$$

onde k é uma constante definida como parâmetro na etapa anterior e $\sigma(f)$ é o desvio padrão dos dados do sinal, sendo assim o ε tem seu valor relativo a base de dados que ele está tentando comprimir (DANIN, 2016).

Nessa etapa é necessário calcular alguma métrica de eficiência do algoritmo. Uma das mais comuns (SALOMON, 2006) é usar a taxa de compressão (C). C é calculado como a razão entre o tamanho dos dados comprimidos Z_c e o tamanho dos dados originais Z_i, como visto na fórmula

$$C = \frac{Z_c}{Z_i} \tag{2.2}$$

Sendo assim $C = 0.3$ quer dizer que os dados comprimidos têm 30% do tamanho dos dados originais. Porém, nesse trabalho é usada a métrica $T(k) = 1 - C$, ou seja:

$$T(k) = 1 - \frac{Z_c}{Z_i} \tag{2.3}$$

Que tem como significado a redução do tamanho em relação ao original para um dado valor da constante k. Por exemplo, $T(k) = 0.3$ significa dessa vez que o dado foi reduzido em 30% em relação tamanho original.

A etapa de validação dos dados consiste em descomprimir os dados comprimidos da etapa anterior e validar que eles reconstruam de maneira fiel o sinal original, obedecendo o limiar de erro ε.

2.4 Extração de características

A etapa de pré-processamento é responsável por transformar os dados brutos coletados em informações mais relevantes sobre a estrutura (FARRAR; WORDEN, 2012). Alguns algoritmos de detecção de danos são capazes de trabalhar com os dados direto dos sensores (DANIN, 2016), porém para a maior parte deles é melhor trabalhar com as características extraídas pois a ideia é que elas são alteradas de forma mais consistente à medida que o nível de dano aumenta (FIGUEIREDO; MOLDOVAN; MARQUES, 2006). Algumas dessas características mais comumente empregadas em SHM são a frequência natural, modos de vibração e taxa de amortecimento (YANG et al., 2004).

A ideia é que as medidas dessas características são atenuadas em casos sem dano e ressaltadas em casos com dano tornando a diferença entre os dois estados mais claras (FARRAR; SOHN; PARK, 2004), facilitando assim o trabalho de classificação dos algoritmos de aprendizado de máquina. Entretanto, vale ressaltar que assim como o dano é capaz de atenuar alguma das características, existem outros fatores ambientais e operacionais que também são igualmente capazes de influenciar essas características (FIGUEIREDO et al., 2011). Portanto essa etapa tem que ser feita com cautela e usando métodos inteligentes.

2.5 Detecção de danos

A etapa de detecção de danos ocorre quando são criados modelos estatísticos para classificar as características extraídas (em caso de pré-processamento) e quantificar o nível de dano de uma estrutura (FARRAR; WORDEN, 2012). Dado a natureza dos dados SHM e sua alta variabilidade é comum que esses modelos sejam criados usando aprendizado de máquina, visto que são algoritmos que têm geralmente mais capacidade de tirar os fatores ambientais e operacionais da sua análise.

Os resultados dos testes da detecção de danos podem classificar corretamente uma amostra dados tanto como sendo com dano, quanto sendo sem dano. Se for classificada com dano uma amostra que realmente tem dano é um verdadeiro positivo, se for classificado como sem dano uma amostra sem dano é um verdadeiro negativo. Além dos casos onde o

modelo acerta a classificação, existe os casos que ele classifica as amostras como sendo falsos positivos e falsos negativos, respectivamente erros do Tipo I e II (LIMA et al., 2016). Erros Tipo II causam riscos a vidas humanas, além de possíveis prejuízos financeiros, enquanto os erros Tipo I também causam algum prejuízo financeiro visto que é custoso mandar equipes para possíveis reparos e inatividade da estrutura. É fundamental que as técnicas escolhidas nessa etapa minimizem os dois tipos de erro, entretanto como os erros Tipo II oferecem mais riscos eles são mais importantes de serem minimizados (LIMA et al., 2016).

Ainda nessa etapa são calculadas duas métricas referentes à eficiência da técnica de detecção de dano em relação a presença de erros Tipo I e Tipo II, a taxa de precisão (P) e a taxa de sensibilidade (R), explicadas nas formulas:

$$P = \frac{VP}{(VP - FP)} \tag{2.4}$$

e

$$R = \frac{VP}{(VP - FN)} \tag{2.5}$$

Onde VP indica o número de verdadeiros positivos, FP falsos positivos e FN falsos negativos. A precisão, que indica a exatidão, está no intervalo de 0 a 1. Quanto mais falsos positivos menor o valor de P, logo quanto maior a precisão melhor. O mesmo é válido para sensibilidade, que por sua vez indica a completude da técnica, quanto maior e mais próxima de 1 melhor.

2.6 Análise de resultados

É na etapa de análise de resultados que o trabalho proposto mais se difere de outros estudos em SHM. Tendo por base o framework de Danin (2016), em vez de realizar a avaliação da compressão e da detecção de danos de forma separada tal qual é comumente encontrado nos estudos que fazem uso das duas etapas no processo, tais como (LIMA et al., 2016) aqui é realizado um cálculo de uma única métrica que leva os resultados dessas duas etapas em consideração.

O primeiro passo dessa análise é a definição de três parâmetros β, ω_1, ω_2. β tem a função de definir a proporção de importância da precisão (P) e sensibilidade (R), ou seja, da importância da redução de erros Tipo I e Tipo II respectivamente. ω_1 e ω_2 são responsáveis definir a proporção de importância entre a compressão e a detecção de danos (DANIN, 2016).

Depois é calculado a métrica relacionada a detecção de danos, chamada $F - Measure$. Essa métrica é dada pela fórmula:

$$F(k) = (1 + \beta^2)\frac{P.R}{(\beta^2.P) + R} \tag{2.6}$$

Sendo assim $F(k) \in [0, 1]$ para um dado valor da constante k. β, a constante de peso, é um valor real maior que zero e é usada para equilibrar a importância dada a P e R no valor final de $F(k)$. Um valor de $\beta = 1$ faz com que as duas contribuam igual para o valor de $F(k)$. Um valor de $\beta < 1$ representa um cenário onde P é mais importante que R, analogamente um valor de $\beta > 1$ representa um cenário onde R é mais importante que P (DANIN, 2016).

Finalmente é feito o cálculo da métrica unificada, que une a métrica extraída pelo algoritmo de compressão e a métrica extraída pela detecção de danos. Essa métrica é chamada de *S-Measure* e é dada pela fórmula

$$S(k) = \frac{(\omega_1.T(k)) + (\omega_2.F(k))}{(\omega_1 + \omega_2)} \tag{2.7}$$

onde $T(k)$ é dado pela equação 2.3 e $F(K)$ é dado pela equação 2.6. As constantes ω_1 e ω_2 são usadas para distribuir a contribuição de cada uma das métricas no valor final. Um valor de $\omega_1 > \omega_2$ significa que a taxa de compressão contribui mais com o valor final, enquanto $\omega_2 > \omega_1$ significa que a diminuição do erro na detecção de danos é mais importante.

3 Técnicas de Compressão de Dados

Em sistemas SHM, um número elevado de sensores geralmente está relacionado com mais precisão na hora de detectar o dano (ZHOU; JIN; ZHENG, 2015). Portanto a quantidade de dados para se ter uma precisão alta também aumenta, assim como a necessidade de armazenar e transmitir esses dados (MADDEN, 2010).

Por esse motivo, a quantidade de dados geradas por sistemas SHM está aumentando e com isso vem a necessidade de compressão de dados cada vez maior nesses sistemas (HUNG; JEUNG; ABERER, 2013). Por meio do uso de aproximações dos dados, eles reduzem o uso e a quantidade de informação armazenada eliminando as partes redundantes.

Existem dois tipos de compressão de dados, com perdas e sem perdas. As compressões com perda diminuem o tamanho dos dados e o reconstroem o dado exatamente igual ele era antes. Já as compressões sem perda tendem a ter taxas de compressão muito maiores, porém o sinal reconstruído tem perdas em relação ao sinal original (SALOMON, 2006). Em aplicações onde a perda de um bit de informação acarreta mudança significativa no conteúdo original, como por exemplo texto, não se deve usar compressão com perda, já para dados de sinais a perda é aceitável (SALOMON, 2006).

Com base nisso, esse trabalho vai utilizar com os algoritmos identificados por Lima et al. (2016) e Hung, Jeung e Aberer (2013) como os mais comumente usado na literatura SHM. Os algoritmos são: *Piecewise Constant Approximation* (PCA), *Adaptive Piecewise Constant Approximation* (APCA), *Piecewise Linear Histogram* (PWLH), *Slide Filter* (SF) e *Chebyshev Approximation* (CHEB).

3.1 Piecewise Constant Approximation

A técnica PCA, introduzida por Lazaridis e Mehrotra (2003), trabalha dividindo os dados em vários pedaços de tamanhos fixos, denominados janelas, e atribuindo um valor constante que representa todos os dados dessa janela. O sinal original vira então um conjunto de valores aproximados. Caso essa janela não possa ser representada por um valor constante sem que algum dos dados esteja a mais de um limiar de erro máximo (ε) desse valor a janela não é comprimida e são salvos os dados originais do sinal (LAZARIDIS; MEHROTRA, 2003).

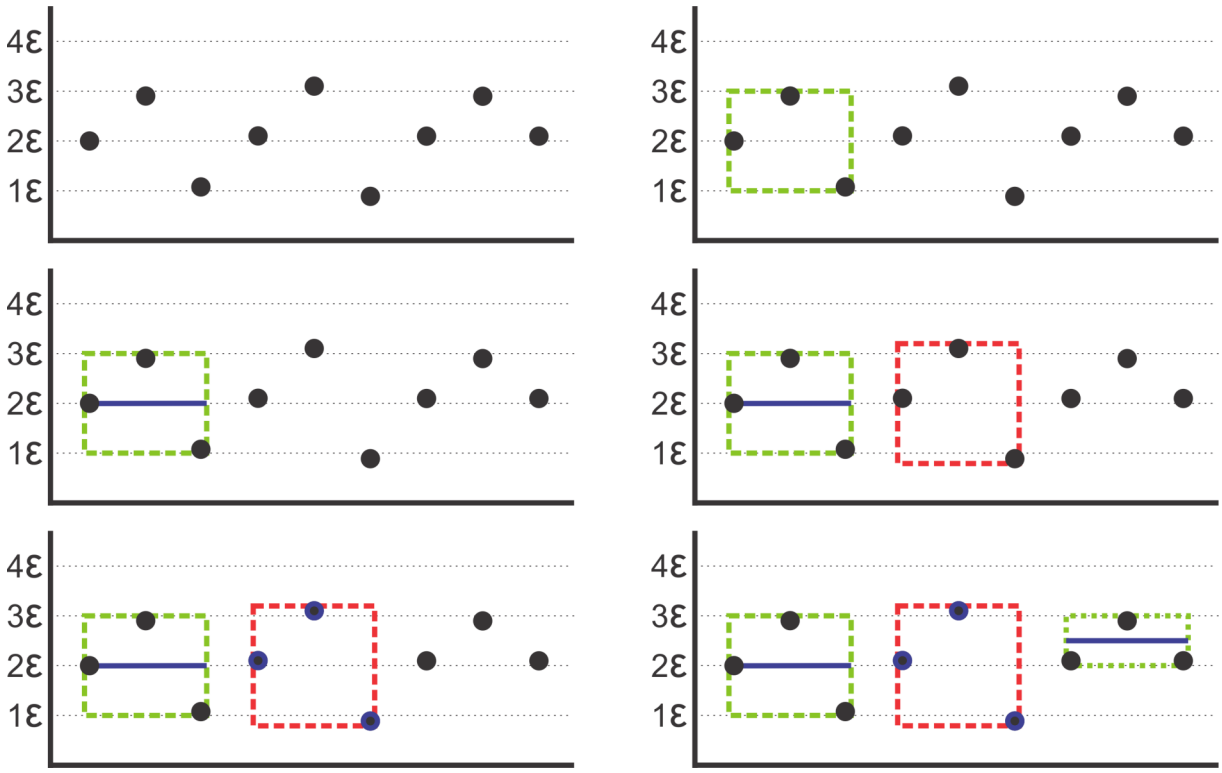
O processo de compressão é dado da seguinte maneira, considerando um conjunto de valores $V = \left\{v_1, v_2, v_3, ..., v_n\right\}$, um tamanho de janela $|w| \geqslant 2$ e um limiar de compressão ε, o algoritmo começa avaliando os primeiros $|w|$ valores consecutivos de V. Com esses valores se cria uma janela, dada por $w = \left\{v_1, v_2, v_2, ..., v_{|w|}\right\}$. Os valores máximo e mínimo

dessa janela são salvos. Se $v_{max} - v_{min} < 2\varepsilon$ então é calculado o valor $\hat{v}$ conforme a fórmula:

$$\hat{v} = \frac{v_{max} - v_{min}}{2} \tag{3.1}$$

É criado então uma entrada $(\hat{v}, f)$ no conjunto de dados aproximados, onde $f = 1$, representando que os dados dessa entrada estão comprimidos. Caso $v_{max} - v_{min} < 2\varepsilon$ não seja verdadeiro, para cada valor de W é criado uma entrada (w_j, f) para $j \in [1, |w|]$, onde $f = 0$, representando que os dados dessa entrada não estão comprimidos. O processo se repete avaliando a próxima janela de dados com os próximos valores consecutivos de c até que todos valores de V sejam aproximados (LAZARIDIS; MEHROTRA, 2003). Logo, o sinal comprimido é representado por: $P = \left\{(\hat{v}_1, f_1), (\hat{v}_2, f_2), (\hat{v}_3, f_3), ..., (\hat{v}_m, f_m)\right\}$

Figura 2 – Exemplo das etapas da aplicação do PCA com janela de tamanho 3.



Fonte: Produzido pelo autor.

Na Figura 2 podemos ver o exemplo da aplicação do PCA com $|w| = 3$. Na primeira janela os três pontos pertencentes a elas, conseguem ser representados por uma constante com distância de no máximo ε para todos os pontos, representada pela linha azul. Já na segunda, janela vemos que a condição $v_{max} - v_{min} < 2\varepsilon$ não é satisfeita, logo cada um dos pontos é salvo individualmente. A terceira janela também é representada por uma constante.

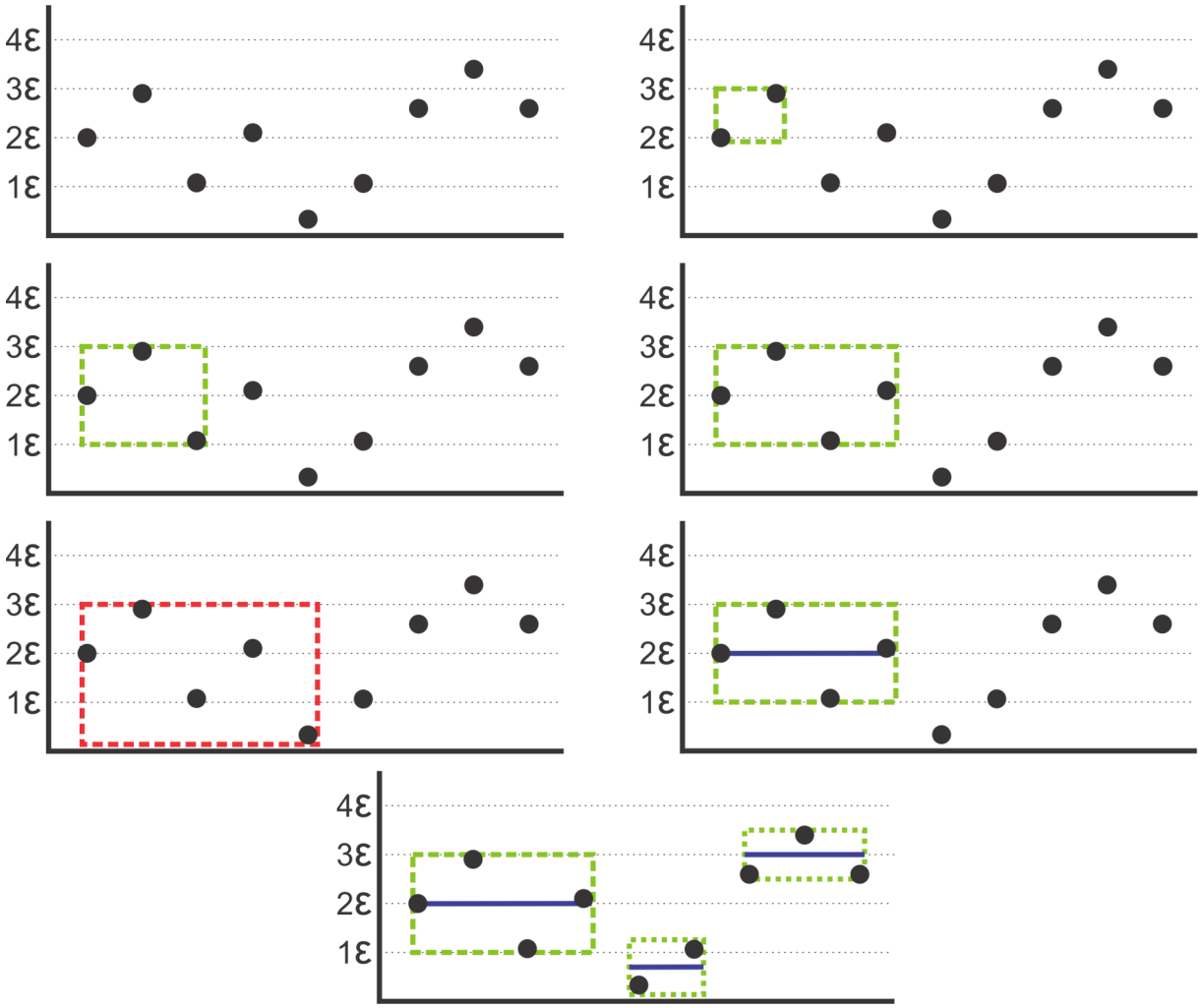
A partir da Figura 2, é possível observar também que na prática o algoritmo tenta envolver os pontos de uma janela dentro de um retângulo de altura máxima 2ε e salva o ponto da altura média do mesmo. Caso não consiga, salva os pontos de maneira separada.

3.2 Adaptive Piecewise Constant Approximation

Proposto por Keogh et al. (2001), o algoritmo APCA é semelhante ao PCA, porém o tamanho das janelas é variável. Com isso os valores constantes aproximados podem ser válidos para diferentes quantidades de dados, tornando o algoritmo mais fléxivel (CHAKRABARTI et al., 2002).

O seu funcionamento é semelhante, considerando os valores $V = \{v_1, v_2, v_3, ..., v_n\}$ e um limiar de compressão ε o algoritmo adiciona um a um os valores consecutivos de V em uma janela w, até que $v_{max} - v_{min} \geqslant 2\varepsilon$. No momento que essa condição é atingida o último valor inserido é removido da janela, e se cria uma nova entrada $(\hat{v}, t)$ no conjunto aproximado, onde $\hat{v}$ é dado pela equação 3.1 e t é o instante no tempo do último valor da janela w. Após isso se começa uma nova janela com o valor descartado da última janela e repete o processo até todos os valores de V terem sido aproximados (CHAKRABARTI et al., 2002). Logo o sinal aproximado é representado por $APCA = \{(\hat{v}_1, t_1), (\hat{v}_2, t_2), (\hat{v}_3, t_3), ..., (\hat{v}_m, t_m)\}$

Figura 3 – Exemplo das etapas da aplicação do APCA.



Fonte: Produzido pelo autor.

Na Figura 3 temos um exemplo de compressão utilizando o algoritmo APCA. Podemos ver que o tamanho das janelas é variável, sempre englobado o máximo de valores que ele consegue de forma que as janelas obedeçam a condição $v_{max} - v_{min} < 2\varepsilon$. No momento que a condição é quebrada, representado pelo retângulo vermelho, o algoritmo volta, salva a janela com um valor constante, representado pela linha azul, e continua de onde parou.

Assim como no PCA, ele também tenta criar um retângulo de altura máxima 2ε que contenha todos os pontos, entretanto ele tenta colocar o máximo de pontos possiveis na janela até que o retângulo perca essa propriedade.

3.3 Piecewise Linear Histogram

O PWLH é uma extensão do APCA, trabalhando de maneira semelhante. Entretanto, em vez de tentar aproximar os valores das janelas com um valor constante, é utilizada uma função linear(BURAGOHAIN; SHRIVASTAVA; SURI, 2007).

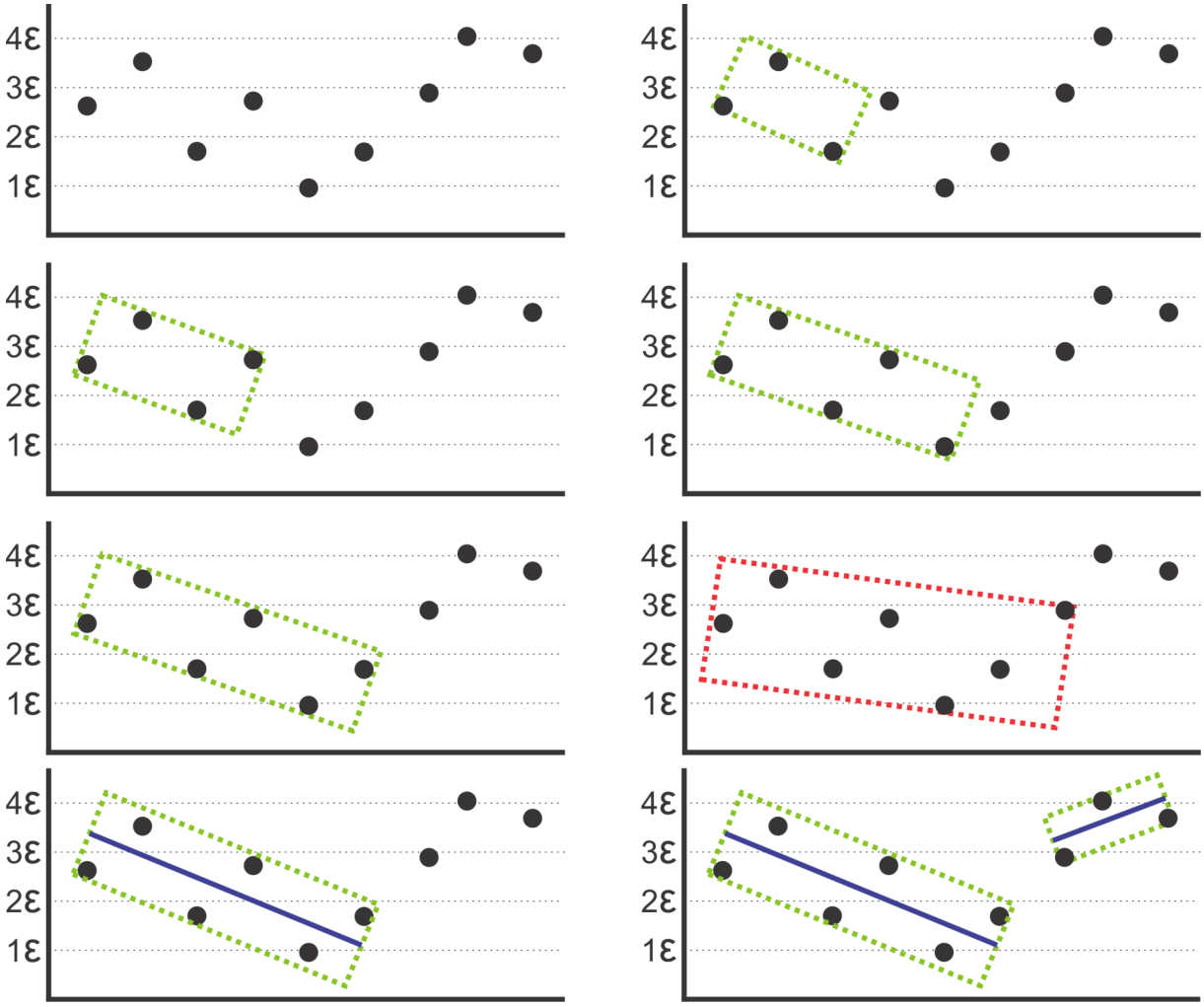
Essa utilização de funções lineares, diferentemente do PCA e do APCA que usam valores constantes, requer maior esforço computacional, sendo possível com isso aproximar melhor os dados com janelas maiores(BURAGOHAIN; SHRIVASTAVA; SURI, 2007).

Seu funcionamento é bastante similar ao APCA, dado um sinal $V = \left\{v_1, v_2, v_2, ..., v_n\right\}$ e um limiar de compressão ε o PWLH adiciona consecutivamente os valores de V em uma janela w e mantém uma reta que passa por todos os pontos sendo que a distância d entre cada ponto e a reta obedece $d \leqslant \varepsilon$ e essa reta seja a reta com os menores valores para d possíveis para essa janela w. No momento que um valor entra na janela e causa que algum $d > \varepsilon$ então esse valor é retirado da janela.

Os valore da janela final são então salvos como uma entrada $(\hat{v}^b, \hat{v}^e, t)$ onde $\hat{v}^b$ é o valor do primeiro ponto da linha, $\hat{v}^e$ é o valor do último ponto da linha e t é instante de tempo final. Esse processo se repete e o sinal aproximado pode ser representado pelo conjunto $PWLH = \left\{(\hat{v}^b_1, \hat{v}^e_1, t_1), (\hat{v}^b_2, \hat{v}^e_2, t_2), (\hat{v}^b_3, \hat{v}^e_3, t_3), ..., (\hat{v}^b_m, \hat{v}^e_m, t_m)\right\}$

A Figura 4 mostra um exemplo de dados comprimidos pelo algoritmo PWLH. Tal qual nos algoritmos anteriores, este também tenta englobar todos os pontos de uma janela em um retângulo com altura máxima de 2ε, entretanto diferente dos outros ele rotaciona esse retângulo para caber mais dados em uma janela e aproximar melhor os dados. O funcionamento interno do algoritmo inclusive calcula as funções lineares usando essa noção de retângulo com menor altura que contém todos os pontos da janela.

Figura 4 – Exemplo das etapas de aplicação do PWLH.



Fonte: Produzido pelo autor.

3.4 Slide Filter

O algoritmo SF, proposto por Elmeleegy et al. (2009), é semelhante ao PWHL, porém a técnica utilizada para gerar a reta tem como objetivo cobrir a maior quantidade de dados possíveis em uma janela (HUNG; JEUNG; ABERER, 2013).

O algoritmo tem dois mecanismos principais, o de filtragem e o de registro (ELMELEEGY et al., 2009). Começando com um sinal $V = \{v_1, v_2, v_3, ..., v_n\}$ e um ε o algoritmo adiciona os dados de V em uma janela w. No momento que a janela w tem dois itens começa a filtragem. Esse mecanismo é responsável por delimitar uma reta U^w como limite superior e uma L^w como limite inferior de todos os dados coletados.

Cada item no SF pode ser representado por (v_i, t_i) onde v_i é um valor e t_i é o instante que ele ocorre. Sendo assim os limites de uma janela w com n valores são calculados

para um novo ponto j dado as fórmulas:

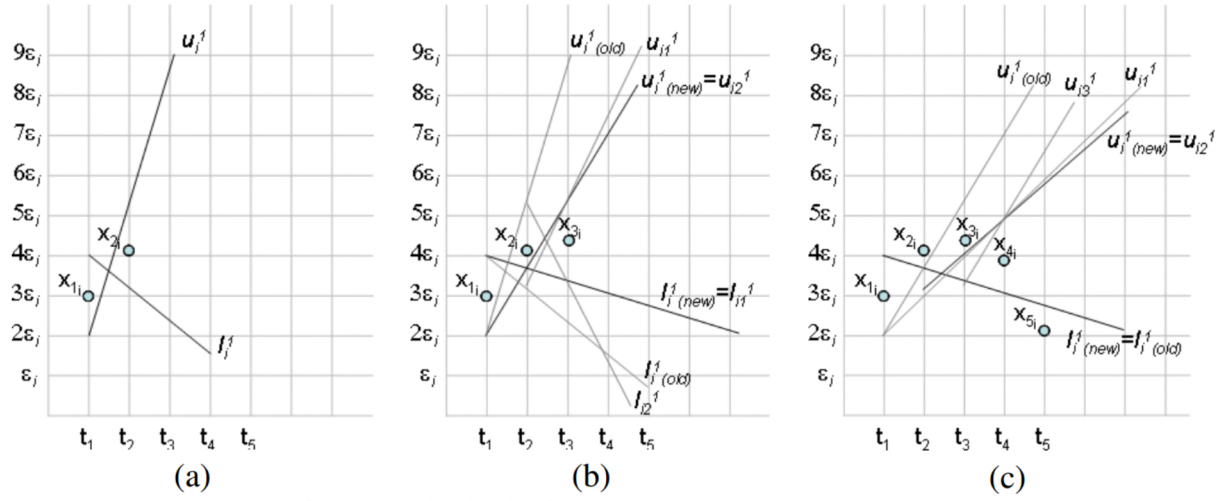
$$u_{ij}^{w} = (\hat{u}_i^b, \hat{u}_j^e) = ((v_i - \varepsilon, t_i), (v_j + \varepsilon, t_j)) \tag{3.2}$$

$$l_{ij}^{w} = (\hat{l}_i^b, \hat{l}_j^e) = ((v_i + \varepsilon, t_i), (v_j - \varepsilon, t_j)) \tag{3.3}$$

para todo i de 1 até n.

Sempre que o algoritmo tenta inserir um item novo no intervalo ele checa se o item está dentro dos limites U^w e L^w ou a no máximo ε de distância deles. Caso o item novo esteja dentro do intervalo é calculado então todas as retas limites superiores e inferiores com os outros pontos do intervalo pelas mesmas formulas 3.2 e 3.3, caso alguma dessas retas de limite superior tenha inclinação menor que U^w ela será a nova U^w e analogamente o mesmo é verdadeiro para o limite inferior.

Figura 5 – Mecanismo de filtragem do Slide Filter.



Fonte: (ELMELEEGY et al., 2009).

Na Figura 5 podemos ver em (a) os limites sendo criados para os dois primeiros pontos da janela, em (b) quando o terceiro ponto chega são calculados os U^1 e L^1 novos para acomodar o ponto novo. Em (c) quando chega o ponto 4 vemos que só U^1 é atualizado por que ele já estava a ε de distância de L^1 .

Caso o item novo esteja fora do limite das retas limites e a mais de ε de distância das mesmas, esse ponto vai para uma nova janela e se fecha a janela atual com o mecanismo de registro. Dado o conjunto G^w de todas as retas g que passam pela intercessão de U^w e L^w, o mecanismo de registro é encarregado de achar o segmento g^w de reta que melhor representa a janela. O mecanismo tem duas formas de achar essa melhor representação, a depender se o segmento for desconectado ou conectado (ELMELEEGY et al., 2009).

Segmentos conectados começam a partir do segmento anterior g^{w-1}, isso acontece caso exista pelo menos um segmento de reta em G^w que intercepte também g^{w-1} após a intercessão de U^{w-1} e L^{w-1} . Existindo esse segmento, g^w vai ser igual ao segmento de reta que obedeca essa condição e minimize o erro quadrado médio de todos os pontos do intervalo. Caso não exista tal segmento, então os pontos são desconectados e a única condição para escolher o segmento $g^w \in G^w$ é que ele minimize o erro quadrado médio de todos os pontos do intervalo (ELMELEEGY et al., 2009).

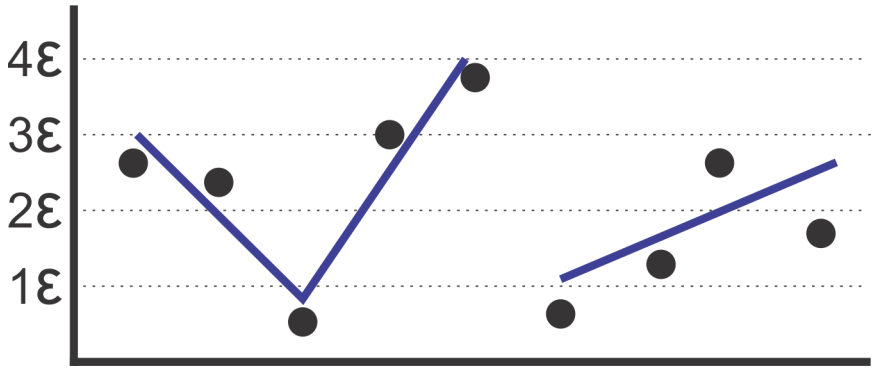
Os segmentos conectados servem para diminuir o número de dados no conjunto de dados comprimido, visto que para armazenar segmentos desconectados é preciso do ponto final e do inicial, enquanto para armazenar segmentos conectados só o final basta (ELMELEEGY et al., 2009).

Sendo assim o conjunto aproximado SV é representado pela formula

$$SV(V) = \left\{(\left\{\hat{v}_1^b\right\}, \hat{v}_1^e, t_1, f_1), (\left\{\hat{v}_2^b\right\}, \hat{v}_2^e, t_2, f_2), (\left\{\hat{v}_3^b\right\}, \hat{v}_3^e, t_3, f_3), ..., (\left\{\hat{v}_m^b\right\}, \hat{v}_m^e, t_m, f_m)\right\} \tag{3.4}$$

onde $\hat{v}_i^b$ é o ponto inicial do segmento de reta, $\hat{v}_i^e$ é o ponto final, t_i é o instante no tempo onde ocorre o ponto final e f_i é uma constante lógica que indica se esse segmento é conectado ou não. Caso o segmento seja conectado com o anterior, ou seja, o primeiro ponto do sinal, então $\left\{\hat{v}_i^b\right\} = \emptyset$.

Figura 6 – Exemplo de aplicação do Slide Filter.



Fonte: Produzido pelo autor.

A Figura 6 ilustra um exemplo da aproximação dos dados com o Slide Filter, em que podem ser vistos em azul os segmentos de reta que compõe o sinal aproximado. Vemos que o segundo segmento é conectado ao primeiro, enquanto o terceiro é desconectado do segundo.

3.5 Chebyshev Approximation

O algoritmo CHEB, proposto por Cheng et al (2007), é próximo do PCA no que consiste em ter uma janela de tamanho fixo para aproximar os dados, porém não usa um valor constante, nem uma função linear, para aproximar os dados. Este algoritmo emprega os polinômios de Chebysheb (HUNG; JEUNG; ABERER, 2013). Sendo assim o sinal original é aproximado por curvas, o que pode gerar uma alta taxa de compressão com pouca perda da qualidade do sinal (LIMA et al., 2016)

Dado um sinal $V = \left\{v_1, v_2, v_3, ..., v_n\right\}$, janela w, sendo $|w| \geq 2$, um grau p para o polinômio e um limiar de compressão ε o algoritmo começa igual o PCA. É crida uma primeira janela com os primeiros $|w|$ valores consecutivos de V. Logo então é feito o processo de aproximação dessa janela em uma combinação linear de polinômios de chebyshev segundo a fórmula:

$$\hat{v}_t = \sum_{j=1}^{p} \delta_j . H_j(t') \tag{3.5}$$

onde

$$t' = (t - \frac{(|w|+1)}{2}) . \frac{2}{(|w|-1)} \tag{3.6}$$

é o valor normal de t, e

$$H_j(x) = \cos(j . \cos^{-1}(x)) \tag{3.7}$$

é a polinomial de chebyshev de grau j e δ é o coeficiente da polinomial de grau j.

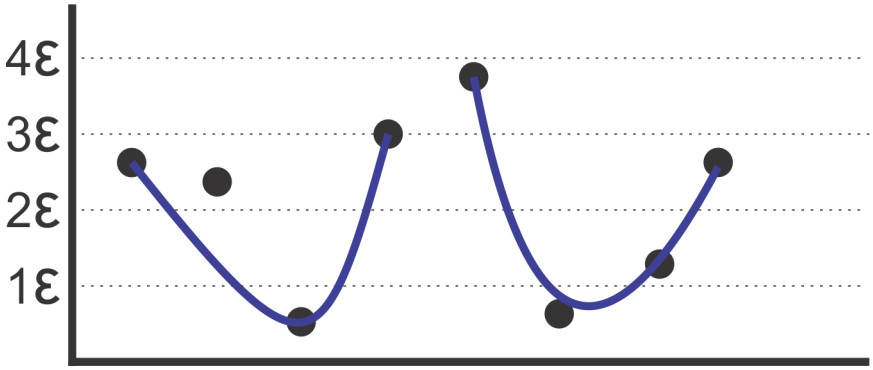
Essa aproximação gera um conjunto $\delta_m = \left\{\delta_m^1, \delta_m^2, \delta_m^3, ..., \delta_m^p\right\}$ de coeficientes para a janela. Tirando o primeiro coeficiente da janela, para todos os outros é checada a condição $|\delta_i| > \varepsilon$, caso seja verdadeira ele continua com seu valor. Caso seja falso, esse coeficiente é substituído por 0. Para cada janela é criada uma palavra de controle cw com tamanho p. Para todo $\delta_m^i \in \delta_m$, $cw_i = 1$ caso o coeficiente seja mantido e $cw_i = 1$ caso não seja mantido. O algoritmo faz então um processo de compressão para cw e transforma em cw'. O processo se repete para todas as janelas seguintes (CHENG et al., 2004).

Sendo assim os dados aproximados podem ser representados pelo conjunto

$$CHEB(V) = \left\{(\left\{\delta_1\right\}, cw_1), (\left\{\delta_2\right\}, cw_2), (\left\{\delta_3\right\}, cw_3), ..., (\left\{\delta_m\right\}, cw_m)\right\} \tag{3.8}$$

Podemos ver na Figura 7 que mostra um sinal aproximado pelo algoritmo de Chebyshev com $|w| = 4$, não é um polinômio que representa perfeitamente os dados

Figura 7 – Exemplo de aplicação do Chebyshev Approximation.



Fonte: Produzido pelo autor.

originais, e sim uma aproximação que tem como objetivo reduzir a quantidade de dados armazenados sem perder muita informação original.

4 Detecção de Danos

No processo de detecção de danos é preciso criar modelos estatísticos para quantificar e qualificar os danos de uma estrutura. Isso pode ser feito usando técnicas de aprendizado de máquina, como proposto neste trabalho.

Dentro dos algoritmos de aprendizado de máquina existem duas classes de algoritmo, supervisionados e não supervisionados (SHALEV-SHWARTZ; BEN-DAVID, 2014). Aprendizado supervisionado é quando o algoritmo tem acesso às respostas certas do que ele quer aprender. No contexto de SHM, por exemplo, seria quando o algoritmo é treinado com dados com dano e sem dano e sabe, de antemão, a diferença entre os dois. No caso do aprendizado não supervisionado, o algoritmo não tem acesso aos resultados corretos e trabalha para criar modelos para classificar os dados de entrada. Em SHM, seria o caso de receber somente dados sem dano e criar um modelo a partir disso(FARRAR; WORDEN, 2010).

Ao se usar modelos supervisionados em SHM, é possível obter resultados mais completos e com maior quantidade de informações sobre o dano (tipo de dano, extensão do dano, vida útil restante, entre outras), enquanto o aprendizado não supervisionado é geralmente limitado a identificar a presença de dano e, em alguns casos, a localização do mesmo (FARRAR; WORDEN, 2010)

Em cenários reais é muito custoso criar e manter as condições de danos em estruturas. Por esse motivo, a maior parte dos dados coletados são geralmente sem danos, tornando mais comum o uso de algoritmos não supervisionados na área de SHM (FIGUEIREDO et al., 2011)

Nesse trabalho foram testados algoritmos de aprendizado de máquina de clusterização não supervisionadas. Clusterização tem como objetivo agrupar os dados em clusters de dados similares. São técnicas geralmente interessantes para SHM pois consegue criar modelos que agrupem situações ambientais e operacionais separados das condições de dano.

Os algoritmos usados nesse trabalho são: *Density Based Spatial Clustering of Applications with Noise* (DBSCAN), *G-Means*, *Affinity Propagation* (AP), *Fuzzy-C-Means* (FCM), *Gaussian Mixture Model* (GMM) e *K-Means*. Os três primeiros são capazes de identificar o número de clusters por conta própria, enquanto nos três últimos é necessário usar métodos para definir o número de clusters. A escolha do número de clusters é importante pois poucos clusters não aprendem bem as nuances da base de dados e muitos clusters podem se especializar tanto que não é possível generalizar a informação adquirida para outras bases de dados e amostras novas.

Alguns dos algoritmos usam funções para medir distância entre pontos. Existem diversas funções que podem ser utilizadas para esse propósito. Nesse trabalho, as distancias são consideradas como a distância euclidiana dada pela fórmula:

$$d(p,q) = \sqrt{\sum_{i=1}^{n}(q_i - p_i)^2} \tag{4.1}$$

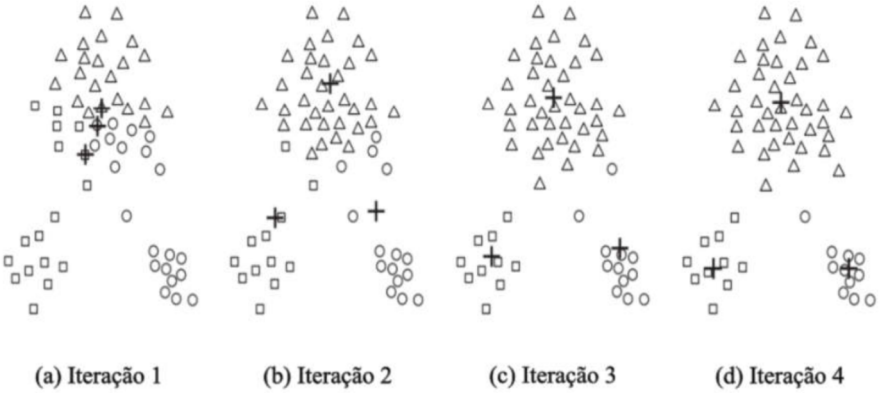
onde p e q são dois pontos de n dimensões, e p_i e q_i são os valores desse ponto na i ésima dimensão.

4.1 K-Means

O algoritmo *K-Means*, criado por Alsabti et al. (1988), é um algoritmo de clusterização que separa os dados em N grupos, ou clusters, representados por um centroide que é a média de todos os pontos pertencem a esses clusters. É um algoritmo que precisa ter definido antes da execução o número N de clusters.

Sua execução é dada em poucas etapas. Primeiro, são iniciados os N centroides com valores aleatórios, informados na entrada do algoritmo, ou usando alguma métrica pré-definida. Em seguida, cada ponto passa a pertencer ao cluster de centroide mais próximo. Cada cluster então calcula seu novo centroide como a média dos pontos que pertencem a ele. A partir daí, novamente cada ponto é atribuído ao cluster com o centroide mais próximo e se repete a etapa de calcular os novos centros. Essas etapas são repetidas até que a posição dos centroides não mude de uma etapa para a outra ou se chegue a um número máximo de iterações (KANUNGO et al., 2002).

Figura 8 – Exemplo de aplicação do K-Means.



Fonte: (TAN; STEINBACH; KUMAR, 2006).

Na Figura 8 podemos ver um exemplo de aplicação do algoritmo K-means com três clusters em uma base de dados bidimensional. É possível ver os centros dos clusters convergindo a cada iteração até que eles se estabilizam na iteração 4.

4.2 G-Means

Proposto por Hamerly e Elkan (2003), o algoritmo tem como objetivo aprimorar o K-Means fazendo com que o número de clusters seja descoberto durante a execução e não como parâmetro de entrada. Para fazer isso, assume que os dados que ele está tentando agrupar seguem uma distribuição gaussiana, ou normal.

O algoritmo funciona aplicando o *K-Means* com um cluster, ou um outro número pré-determinado de clusters iniciais, no conjunto X de dados e testando para cada cluster se ele parece seguir uma distribuição normal. Cada não seja bem representado por uma distribuição normal, esse cluster é dividido por dois usando k-means somente nos dados que pertenciam ao cluster original. Cada cluster resultante recebe os dados que pertencem a ele é o algoritmo se repte de maneira recursiva (HAMERLY; ELKAN, 2003)

Para fazer a verificação se os dados se comportam como uma gaussiana, é usado o teste de Anderson-Darling, o qual foi considerado, conforme Hamerly e Elkan (2003), como um dos mais poderosos para medir a normalidade de uma amostra. Dado valores $x_i \in X$, sendo X um conjunto com média 0 e desvio padrão 1. Sendo $x(i)$ o i ésimo elemento de X ordenado. Sendo $z_i = F(x(i))$, onde F igual a $N(0,1)$ função de distribuição cumulativa. A fórmula para a estatística Anderson-Darling é

$$A^2(Z) = -\frac{1}{n}\sum_{i=1}^{n}(2i-1)[\log(z_i) + \log(1 - z_{n+1-i})] - n \tag{4.2}$$

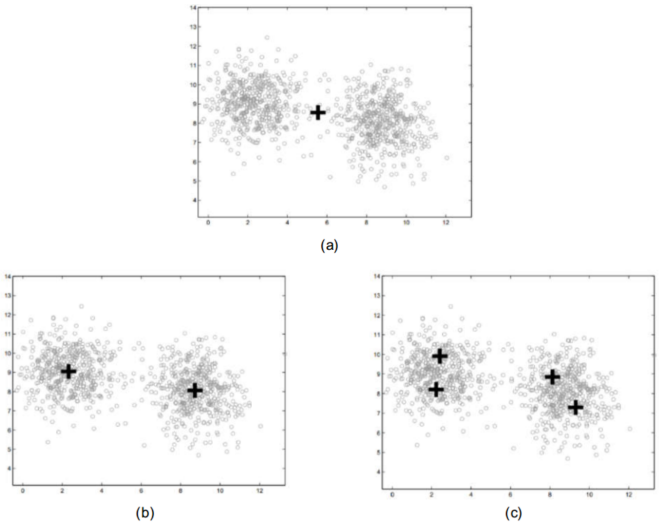
Em casos que a média e o desvio padrão tenham sido estimados pelo conjunto de dados, como é o caso da clusterização, é preciso corrigir a estatística para

$$A^2_*(Z) = A^2(Z)(1 + 4/n - 25/(n^2)) \tag{4.3}$$

Dado um cluster C, após dividi-lo em dois filhos c_1 e c_2, roda-se o k-means com dois clusters, com os dados do cluster C original. Quando terminada a execução, mede-se o vetor de distância entre c_1 e c_2, $v = c_1 - c_2$, pois essa direção foi encontrada pelo algoritmo como significativa para separar os dados. Projeta-se, então, todos os dados de C em v em um novo conjunto C', tal que $c'_i = \left\langle c_i, v \right\rangle / ||v||^2$ para todo $c_i \in C$. O conjunto C' resultante em seguida é ajustado para ter media 0 e desvio padrão 1. Calcula-se $A2(C')$, caso o valor resultante esteja dentro de um intervalo significativo α o cluster original C é aceito. Caso contrário, o processo se repete para c_1 e c_2.

Na Figura 9 vemos em (a) um cluster que não passou pelo teste Anderson-Darling, logo foi dividido em dois outros clusters vistos em (b) que também não passaram no teste e foram divididos em dois cada, vistos em (c) . Os quatro clusters resultantes seguem uma distribuição gaussiana logo o processo para.

Figura 9 – Exemplo de aplicação do GMEANS.

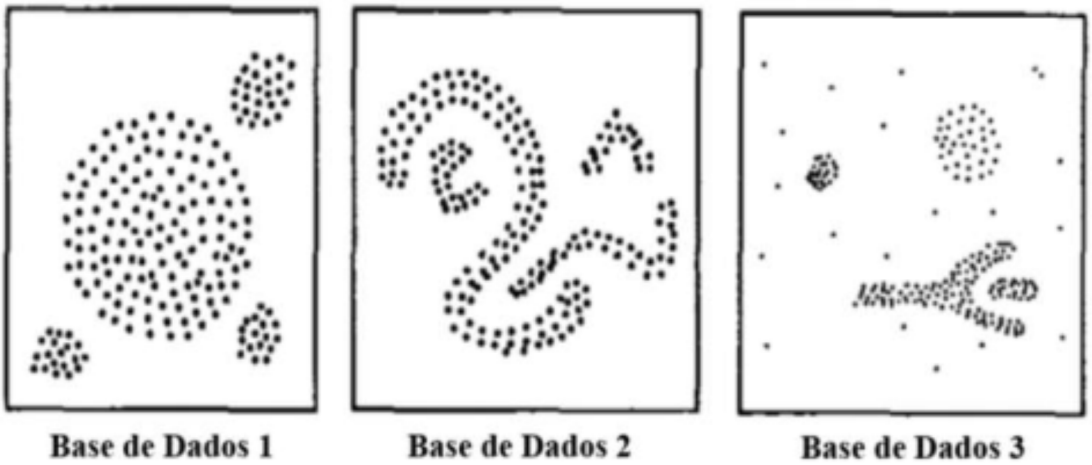


Fonte: (HAMERLY; ELKAN, 2003).

4.3 DBSCAN

O algoritmo DBSCAN, proposto por Ester et al. (1996), tem como ideia principal tentar encontrar regiões com alta densidade de pontos de dados. Uma das características do DBSCAN é não precisar entrar com o parâmetro de número de clusters, além de conseguir tanto agrupar os dados de várias formas como lidar bem com ruídos. Vemos em 10 um exemplo de clusterização por densidade.

Figura 10 – Exemplo de clusterização por densidade.



Fonte: (ESTER et al., 1996).

O DBSCAN trabalha calculando a densidade (d) de cada ponto, ou seja, a quantidade de pontos até no máximo uma distancia r dele. Caso a densidade de um ponto seja maior que um número mínimo de amostras n ele e seus vizinhos no raio r pertencerão ao mesmo cluster. O algoritmo então faz a mesma checagem em todos os esses pontos

vizinhos (ESTER et al., 1996). Ambos n e r são dados como parâmetro do DBSCAN.

De maneira mais detalhada, o DBSCAN separa os pontos em três tipos:

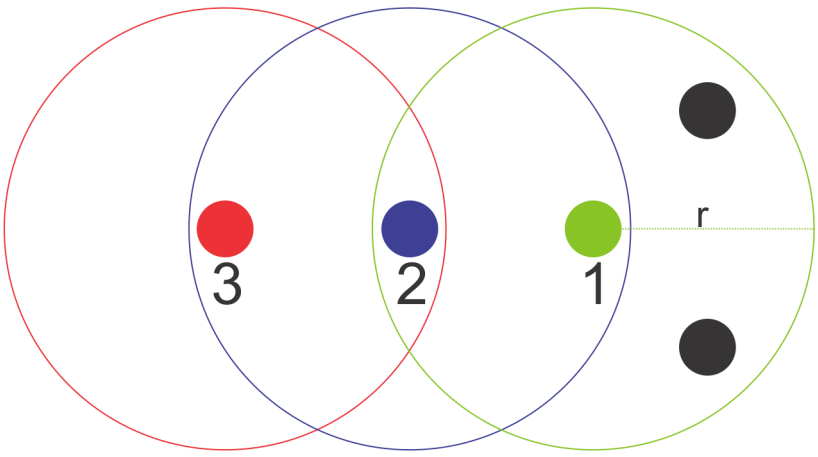
- Pontos de núcleo: são pontos onde $d \geq n$
- Pontos de borda: são pontos onde $d < n$, porém pelo menos um dos pontos na sua vizinhança é um ponto de núcleo.
- Pontos de ruído: são pontos onde $d < n$ e nenhum dos pontos na sua vizinhança é um ponto de núcleo.

Um ponto q é diretamente alcançável por densidade por p se eles tiverem uma distância menor que r e ambos forem pontos de núcleo. Um ponto q é alcançável por densidade por p se existir uma sequência $p_1, p_2, ...p_n$, sendo $p_1 = p$ e $p_n = q$, onde cada p_{i+1} é diretamente alcançável por densidade por p, sendo todos os pontos da sequência necessariamente pontos de núcleo, com a exceção de q que pode ser um ponto de borda. Dois pontos p e q tem conexão por densidade se existir um ponto o de onde ambos possam ser alcançados por densidade.

Dado todas essas definições, um cluster é um conjunto não vazio de pontos C que satisfaça duas condições:

- Para cada dois pontos p,q se $p \in C$ e q é alcançado por densidade por p, então $q \in C$
- Para cada dois pontos $p, q \in C$, p e q são conectados por densidade.

Figura 11 – Exemplo de ponto de núcleo, borda e ruído para o DBSCAN.



Fonte: Produzido pelo autor.

Vemos na Figura 11 um exemplo de DBSCAN com o número mínimo de de vizinhos $n = 3$. O ponto 1 tem $d = 3$, ou seja, tem outros três pontos a uma distância r, isso caracteriza ele como um ponto de núcleu. O ponto 2 tem $d = 2$ e como o ponto 1, que é

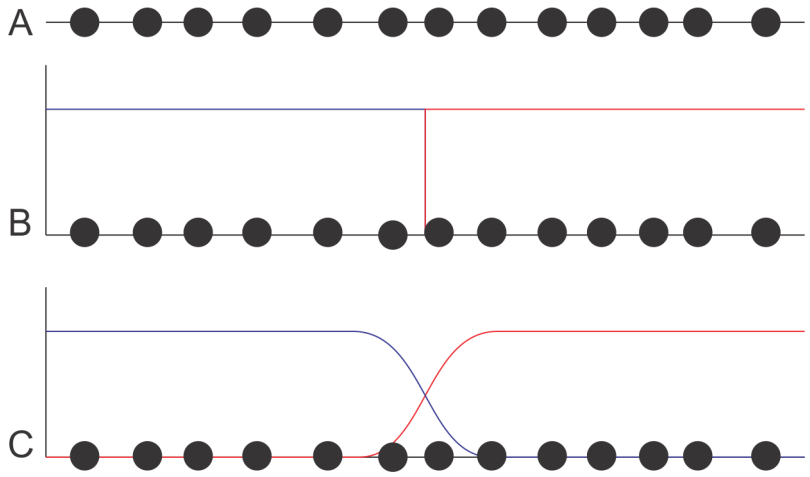
de núcleu, está na sua vizinhança então ele se torna um ponto de borda. O ponto 3 tem $d = 1$ e não tem nenhum ponto de núcleu na sua vizinhaça, caracterizando ele como ruído.

4.4 Fuzzy-C-Means

Proposto por Bezdek et al. (1984), o FCM é um algoritmo de clusterização categorizado como *soft*. Diferente dos anteriores, chamados *hard clustering*, em que uma amostra de dado pertencia a um cluster, em algoritmos de *soft clustering*, cada amostra de dado pertence a todos os clusters com um grau de pertinência (SHALEV-SHWARTZ; BEN-DAVID, 2014), ou existe uma probabilidade de ele pertencer a cada cluster.

A Figura 12 mostra um exemplo da diferença entre *soft* e *hard clustering*. Podemos ver que no primeiro caso, os elementos pertencem a um cluster ou ao outro, representados pelas linhas vermelhas e azul. Já no segundo, vemos que não existe essa rigidez e os dados próximos da fronteira pertencem aos dois clusters, também representado pelas funções vermelha e azul.

Figura 12 – Exemplo de função de pertinência em um cluster hard (B) e em um cluster soft (C) aplicados ao mesmo conjunto de dados (A).



Fonte: Produzido pelo autor.

O FCM, em específico, é bastante semelhante ao *K-Means* no sentido de procurar o centroide de cada cluster, recalcular as funções de pertinência, calcular os novos centroides e repetir o processo até que os centroides não se movam mais que um limiar α (BEZDEK; EHRLICH; FULL, 1984). Entretanto, o processo para calcular o quanto um ponto pertence ou não a cada cluster, bem como o cálculo das centroides são diferentes.

Considerando uma conjunto de dados $X = \{x_1, x_2, x_3, ..., x_n\}$ elementos, um conjunto $C = \{c_1, c_2, c_3, ..., c_c\}$ de clusters, a pertinência de um ponto x_i em um cluster c_j é

dada por:

$$w_{ij} = \Big(\sum_{k=1}^{c} \Big(\frac{d_{ij}}{d_{ik}} \Big)^{2/(m-1)} \Big)^{-1} \tag{4.4}$$

Onde d_{ij} é a distância entre o ponto x_i e o centroide c_j e m é uma constante de imprecisão. Quanto maior o valor, mais *soft* vai ser as pertinências em cada cluster, ou seja, valores menores de pertinências em um número maior de clusters aumentando a incerteza.

Após calculado as pertinências são calculados os centroides c_i pela fórmula:

$$c_i = \frac{\sum_{k=1}^{N} (w_{ik})^m . X_k}{\sum_{k=1}^{N} (w_{ik})^m} \tag{4.5}$$

E se repete o processo até que a diferença entre os centroides de duas iterações seja menor que um limiar α.

4.5 Gaussian Mixture Models

O GMM é um método que constrói um modelo probabilístico que tenta estimar gaussianas para para englobar uma base dados. Neste algoritmo uma gaussiana j é representada por uma tupla $(\mu, \Sigma_j \alpha_j)$, onde μ_j é a média da gaussiana, Σ_j é a covariância e α_j é um parâmetro de proporção. Uma gaussiana é considerada como se fosse um cluster. Um número N de gaussianas é dado como entrada do algoritmo. A função de probabilidade de uma amostra x_i pertencer a uma gaussiana $(\mu_j, \Sigma_j, \alpha_j)$ é dada pela equação:

$$f(x_i|\mu_j, \Sigma_j) = \frac{1}{\sqrt{(2\pi)^K |\Sigma_j|}} \exp(\frac{1}{2}(x_i - \mu_j,)^T \Sigma_j^{-1} (x_i - u_j)) \tag{4.6}$$

Assim como o FCM, o GMM também é um método de *soft clustering*, ou seja, uma amostra tem uma probabilidade de uma amostra x_i pertencer a um certo cluster c_j, calculada pela equação:

$$r_{ij} = \frac{\alpha_j f(x_i|\mu_j, \Sigma_j)}{\sum_{c=1}^{K} \alpha_c f(x_i|\mu_c, \Sigma_c)} \tag{4.7}$$

E para calcular os parâmetros de uma gaussiana k temos as seguintes equações:

$$\alpha_k = \frac{1}{|X|} \sum_{i=1}^{|X|} r_{ik}, \tag{4.8}$$

$$\mu_k = \frac{\sum_{i=1}^{|X|} r_{ik} x_i}{\sum_{i=1}^{|X|} r_{ik}}, \tag{4.9}$$

$$\Sigma_k = \frac{\sum_{i=1}^{|X|} r_{ik}(x_i - \mu_k)^2}{\sum_{i=1}^{|X|} r_{ik}}, \tag{4.10}$$

sendo $|X|$ o número de amostras no conjunto X.

Calcular de maneira analitica as gaussianas que melhor representam os dados é uma tarefa muito complexa, por tanto é usado um método numérico chamado Maximização de Expectativas (EM). O método é bastante similar ao método de convergência do *K-Means*, a iteração dele é dada em duas etapas. Calcular as probabilidades de todas as amostras pertencerem a cada cluster e calcular os novos parâmetros das gaussianas. Isso se repete até que haja convergencia.

A convergência é atingida quando o *log-likehood*, a mistura de todas as gaussianas, é menor que um limiar dado ao algoritmo. A equação do *log-likehood* é dada por:

$$L = \sum_{i=1}^{N} \log(\sum_{k=1}^{K} \alpha_k f(x_i|\mu_j, \Sigma_k)) \tag{4.11}$$

4.6 Affinity Propagation

O método AP, proposto por (FREY; DUECK, 2007), difere dos anteriores por ser baseado em troca de mensagens entre amostras. Além disso ele também não tenta achar um centro para os clusters, e sim amostras exemplares que definam um cluster. E como outros já vistos, não é preciso especificar um número de clusters anteriormente (FREY; DUECK, 2007).

Para executar o algoritmo é definido uma função $s(x_i, x_k)$ que define similaridade entre dois pontos. Essa função tem que ser uma na qual se x_i é mais similar de x_j do que de x_k então $s(x_i, x_,) > s(x_i, x_k)$. Nesse trabalho a função é igual a negativa da distância euclidiana, dada pela formula 4.1.

Para o caso especial $s(x_k, x_k)$ o algoritmo define um valor específico de inicialização que pode ser escolhido entre vários métodos, como a média de similaridade com os outros pontos ou a menor das similaridades com outros pontos. Esses métodos resultam em um número moderado e um número pequeno de clusters, respectivamente (FREY; DUECK, 2007). Nesse trabalho é usado a média das similaridades com outros pontos.

Existem dois tipos de mensagens trocadas entre os pontos, responsabilidades (r) e disponibilidade (a). A responsabilidade $r(i, k)$ é mandada do ponto i para o potencial exemplar k dizendo quão adequado k é de ser o ponto exemplar de i, levando em conta outros potenciais exemplares para i. Dado pela equação

$$r(i, k) = s(i, k) - max_{k's.t.k' \neq j}\Big\{(a(i, k') + s(i, k'))\Big\} \tag{4.12}$$

A disponibilidade $a(i,k)$ é mandada do potencial exemplar k para i dizendo quão adequado ele é de ser o exemplar de i, levando em conta o apoio que ele tem dos demais pontos para ser exemplar. No caso especial $a(k,k)$ essa mensagem quer dizer a evidencia acumulada de esse ponto ser um exemplar e é calculada de maneira diferente. As equações são:

$$a(i,k) = min\Big\{0, r(k,k) + \sum_{i' \ s.t. \ t \neq \{i,j\}} max\Big\{0 + r(i',k))\Big\}\Big\} \tag{4.13}$$

$$a(k,k) = \sum_{i' \ s.t. \ t \neq i} max\Big\{0 + r(i',k))\Big\} \tag{4.14}$$

O algoritmo funciona mandando todas as mensagens de responsabilidade e depois todas as mensagens de disponibilidade, e repete até convergência, ou seja, os clusters permaneçam os mesmos, ou ultrapasse um número máximo de iterações. Os clusters resultantes são dados pelos pontos onde $r(k,k) + a(k,k) > 0$.

5 Metodologia de Testes

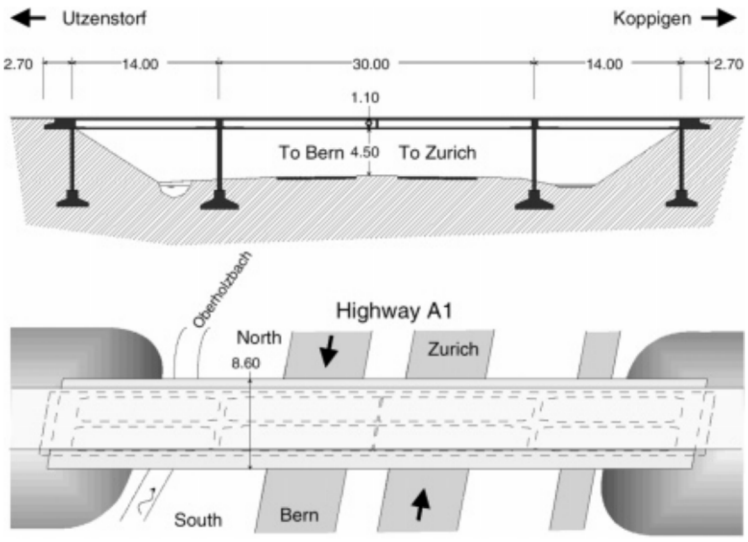
Para os testes foi utilizada a framework que compreende das etapas de aquisição de dados, definição de parâmetros, compressão de dados, validação, extração de características, detecção de danos e análise de resultados.

Para avaliar os algoritmos PCA, APCA, PWLH, SF e CHEB de compressão de dado e as técnicas *K-Means*, *G-Means*, DBSCAN, FCM, GMM e AP de detecção de dano foi escolhido utilizar a base de dados da ponte Z24 na Suíça.

5.1 Aquisição de Dados

A ponte Z24 foi construída em 1961 com a intenção de ligar as aldeias de Utzenstorf e Koppigen na Suíça. Essa ponte é uma estrutura pós-tensionada de vigas de caixa de concreto. A ponte possui um vão principal no meio de 30 metros e dois vãos laterais de 14 metros cada, como visto na Figura 13.

Figura 13 – Ilustração da z24



Fonte: (PEETERS; ROECK, 2001)

Por motivos que não tinham relação com a qualidade ou estrutura da ponte, ela precisou ser demolida para dar espaço a uma ponte maior. Entretanto antes de demolir ela, foram feitos testes para o projeto *System Identification to Monitor Civil Engineering* (SIMCES) para extrair dados que pudessem ser usados para testes de detecção de danos (PEETERS; ROECK, 2001)

Nesses testes foram implantados 49 sensores que coletaram dados de vibração e ambientais. De hora em hora durante aproximadamente 10 meses entre 1997 e 1998

foram coletados dados dos sensores, gerando um volume de 3932 medições (235 dias) (FIGUEIREDO et al., 2014).

Por volta do último mês, foram introduzidas condições de danos propositalmente para adquirir dados dos sensores em cenários com danos (PEETERS; ROECK, 2001). Sendo assim a base dados tem um longo período de medições sem dano e um período com danos, possibilitando fazer testes e validações com os dados gerados.

Foi identificado (DANIN, 2016) que dentre os 8 sensores que funcionam por todo o período dos estudos, o sensor de número 5 era o que melhor representa as características da ponte, portanto nesse trabalho será usado os dados referentes a esse sensor.

5.2 Definição de parâmetros

Para os parâmetros de compressão foi definido que $w = 16$ para todos os algoritmos que utilizam janelas no seu funcionamento. O valor de k foi escolhido como todos entre 0.1 e 5 variando de 0.1, totalizando 50 valores de k para cada método de compressão. Esses valores foram utilizados pois foi observado que para $k > 5$ os valores de $T(k)$ permanenciam quase constantes.

Para os algoritmos de detecção de dano cujo o número de cluster precisa ser previamente definido (GMM, *K-Means* e FCM), foi utilizado todos os valores entre 1 e 15 e escolhido o que tinha as melhores médias. Para o GMM foi 4, para o *K-Means* foi 2 e para o FCM foi 3. Para o DBSCAN foi escolhido que a distancia d fosse 0,9 e o número de vizinhos n minímo para categorizar um ponto de núcleo fosse 3.

Para as constantes relacionadas os valores foram definidos tais que $\omega_2/\omega_1 = 1,5$. Segundo (DANIN, 2016) esse valor é o que apresenta o melhor balanceamento entre compressão e detecção de danos. A outra constante de balanceamento de erros Tipo I e II foi escolhida como $\beta = 1$, dando a mesma importância para ambos.

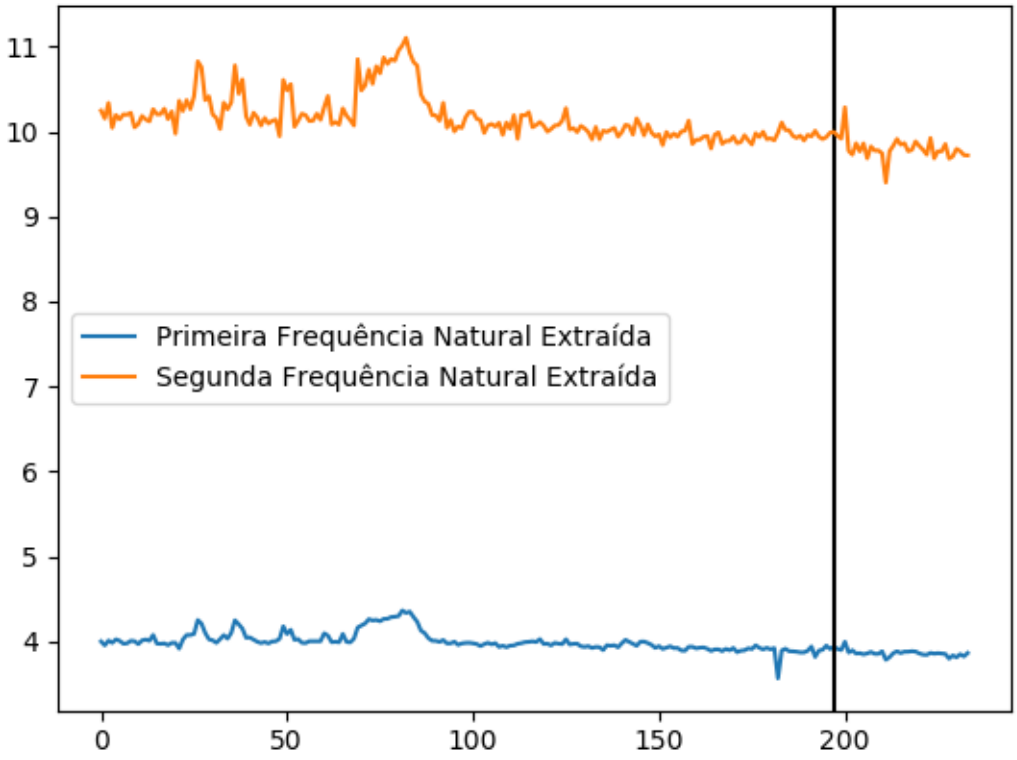
5.3 Extração de características

Nessa etapa foi extraído as características relevantes dos dados obtidos durante os 10 meses de testes e monitoramento da Z24. Conforme mencionado antes, foram utilizados somente os dados do sensor 5 para esse propósito. Outra decisão foi utilizar os dados obtidos durante as medições de 12h, pois estão mais sucintos a variabilidade de temperatura e portanto são um desafio maior de classificar.

Não foi, porém, utilizado os dados brutos dos sensores. Para extrair características relevantes desse sinal foi utilizado um processo de decomposição do sinal no domínio da frequência. Usando somente os dados do sensor 5, foram encontradas 2 frequências naturais

de vibração da ponte.

Figura 14 – Frequencias naturais extraidas do sensor 5 nas 234 amostras no horario de 12h



Fonte: Produzido pelo autor.

Na Figura 14 vemos os dados das duas frequências naturais durante os 234 dias com observação no horário de 12h. Podemos observar que da amostra 197 para frente, quando foram introduzidos os danos artificiais nas pontes, as frequências diminuem. Logo essa é uma característica significativa que tem relação com a presença de danos.

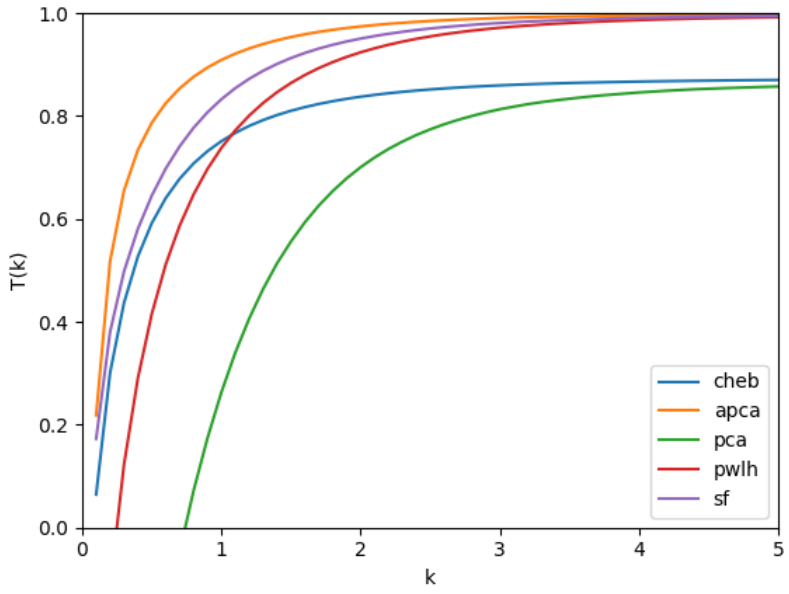
6 Resultados

Esse capítulo mostra os resultados obtidos através dos testes mencionados anteriormente e faz uma análise dos mesmos. Essa análise é feita sobre a compressão de dados, para visualizar os resultados da métrica $T(k)$. Para visualizar a F-Measure é feita uma análise dos erros Tipo I e Tipo II que foram obtidos com cada combinação. É feita a análise de $S(k)$ para os valores das constantes ω_1, ω_2 e β, utilizando $T(k)$ e $F(k)$.

6.1 Compressão de Dados

Na etapa de compressão de dados, os dados dos sensores foram comprimidos e descomprimidos e foi feito o cálculo da taxa de compressão para cada valor de k entre 0,1 e 5.

Figura 15 – Taxas de compressão dos algoritmos



Fonte: Produzido pelo autor.

A Figura 15 mostra a taxa de compressão, dada pela equação 2.3, para cada um dos algoritmos. Quanto maior k, maior $T(k)$, visto que ε está diretamente relacionado com k.

Os resultados obtidos mostram que o APCA, SF e PWLH são os que conseguem obter uma taxa de compressão maior. Esses três métodos se aproximaram de 100% de compressão a partir de aproximadamente $K = 3,5$. O maior pico foi de 99,7% para o algoritmo APCA.

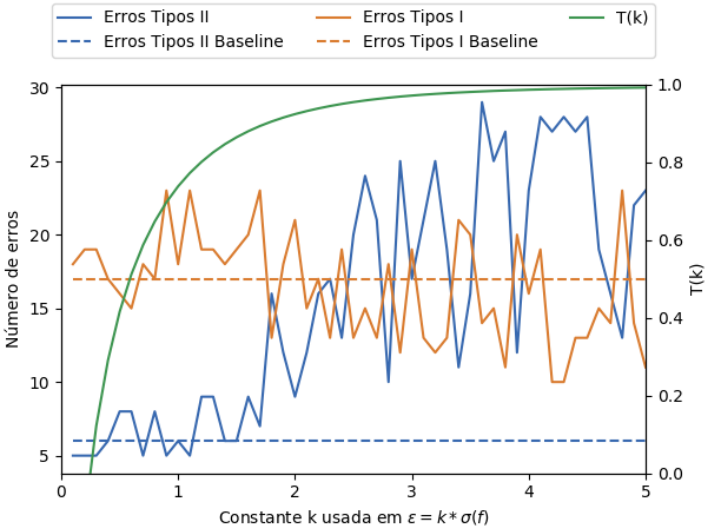
O CHEB e o PCA não mantêm uma taxa de compressão tão grande, se estabilizam em um valor por volta de $k = 1,5$ e $k = 3,5$, respectivamente. O maior valor de $T(k)$ para essas tecnicas foi 85,7% e 87%, para $k = 5$. Os resultados obtidos nessa etapa são condizentes com o estudo de Danin (2016).

6.2 Quantidade média de erros

Foi avaliado também, por algoritmo de compressão a quantidade média de erros Tipo I e Tipo II para cada algoritmo de detecção de dano. Somente as combinações que obtiveram as melhores médias de erro vão ser consideradas.

Na Figura 16 podemos ver o número de erros de cada tipo para cada valor de k, assim como o valor da taxa de compressão da combinação PWLH com AP. É possível visualizar que os números de erros não seguem um padrão constante, com muita flutuação. Apesar disso, existe uma tendência dos erros Tipo II crescerem junto com $T(k)$, enquanto os erros Tipo I diminuem em proporção um pouco menor.

Figura 16 – Média de erros Tipo I e II utilizando PWLH para compressão e o algoritmo AP para detecção de danos.

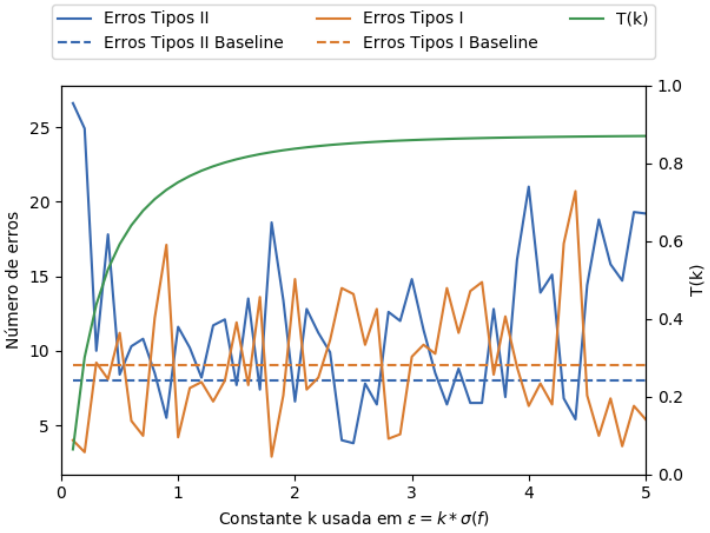


Fonte: Produzido pelo autor.

A Figura 17 mostra que o número de erros Tipo I e Tipo II da combinação de CHEB com GMM, ambos permanecem em média próximo dos resultados sem compressão, porém com uma grande variação. Assim como no gráfico da combinação anterior os gráficos para os erros Tipo I e II se relacionam de maneira inversamente proporcional, nesse caso quase simétrica.

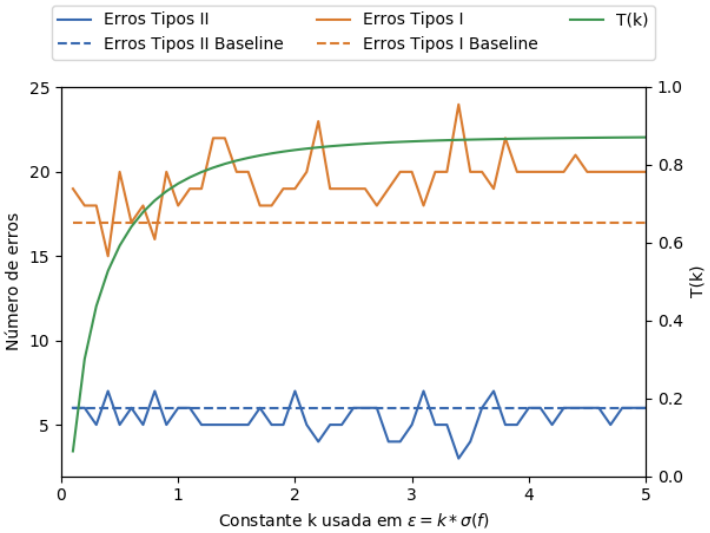
A Figura 18, que mostra combinação CHEB e AP, é semelhante a anterior em que os erros permanecem próximos do baseline, nesse caso entretanto a variação de erros é muito menor.

Figura 17 – Média de erros Tipo I e II utilizando CHEB para compressão e o algoritmo GMM para detecção de danos.



Fonte: Produzido pelo autor.

Figura 18 – Média de erros Tipo I e II utilizando CHEB para compressão e o algoritmo AP para detecção de danos.

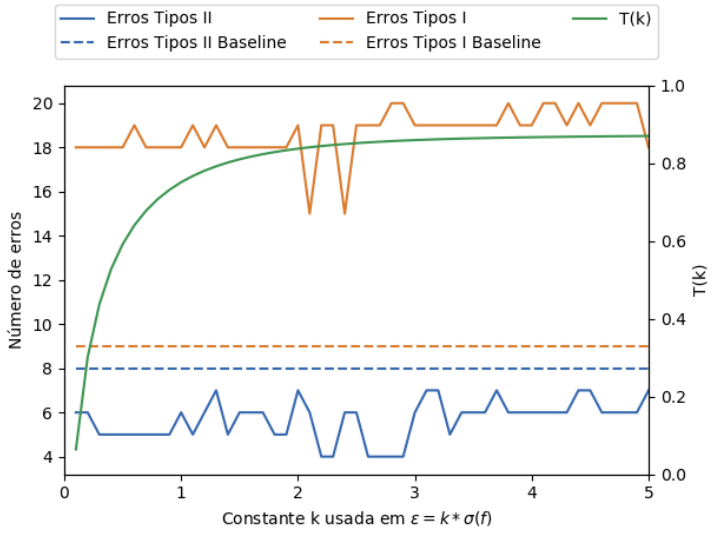


Fonte: Produzido pelo autor.

A Figura 19 mostra a Combinação CHEB e K-Means. É possível ver que as combinações envolvendo o método CHEB tendem a manter o número de erros constantes. No caso do K-Means o número de erros Tipo I se mantém constante bem acima da baseline, enquanto os erros Tipo II se mantém sempre abaixo da baseline.

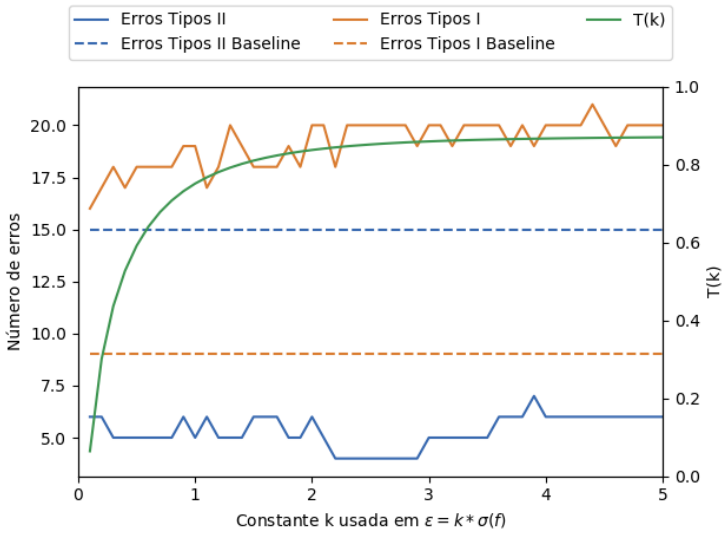
A Figura 20 mostra um comportamento menos constante que as anteriores relacionadas ao CHEB, quando combinado com o FCM. Os erros Tipo I aumentam um pouco enquanto os Tipo II permanecem quase constantes.

Figura 19 – Média de erros Tipo I e II utilizando CHEB para compressão e o algoritmo K-Means para detecção de danos.



Fonte: Produzido pelo autor.

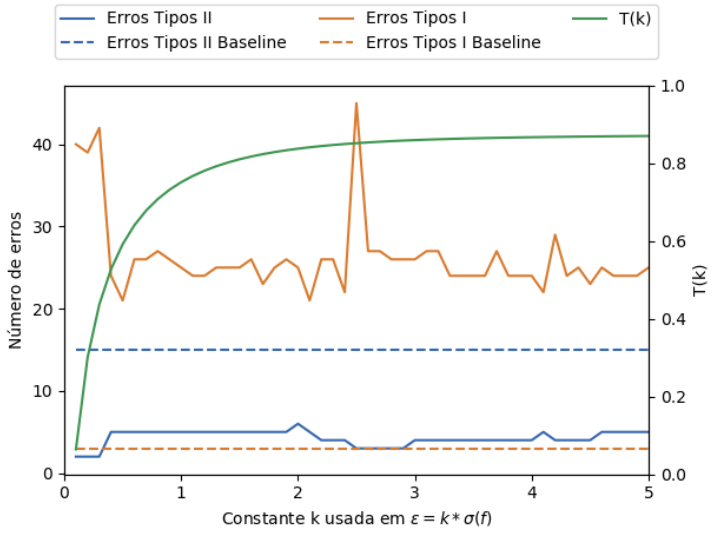
Figura 20 – Média de erros Tipo I e II utilizando CHEB para compressão e o algoritmo FCM para detecção de danos.



Fonte: Produzido pelo autor.

O gráfico na Figura 21 mostra que o DBSCAN, em conjunto com o CHEB, tem um comportamento atípico em relação ao erros Tipo I com picos grandes que não são relacionados a mudanças nos erros Tipo II.

Figura 21 – Média de erros Tipo I e II utilizando CHEB para compressão e o algoritmo DBSCAN para detecção de danos.



Fonte: Produzido pelo autor.

6.3 Resultados da *S-Measure*

Com o erro e a compressão calculado é possível fazer o cálculo de $S(k)$. A Tablea 1 mostra os dez melhores valores da métrica para os testes realizados com os parámetros $\omega_2/\omega_1 = 1,5$ e $\beta = 1$

Tabela 1 – Dez maiores $S(K)$ entre as combinações testadas.

Compressão	Clusterização	K	T(K)	Tipo I	Tipo II	$F(K)$	$S(K)$
CHEB	GMM	2.5	0.851679165	13.8	3.8	0.7907	0.8151
CHEB	GMM	2.4	0.849486469	14.2	4	0.7839	0.8101
CHEB	GMM	3.8	0.866000978	12.3	6.9	0.7584	0.8015
CHEB	GMM	2.6	0.853659928	10.4	7.8	0.7622	0.7988
CHEB	GMM	3.2	0.861697098	9.8	8.5	0.7567	0.7987
CHEB	GMM	2.7	0.855363547	12.8	6.4	0.7598	0.7980
CHEB	GMM	2.9	0.858330616	4.4	12	0.7530	0.7951
CHEB	GMM	3.5	0.864166938	14	6.5	0.7486	0.7948
CHEB	GMM	3.3	0.862618194	14.2	6.4	0.7486	0.7942
CHEB	GMM	3.6	0.864827193	14.6	6.5	0.7437	0.7922

Todas combinações que ficaram entre as dez melhores são de CHEB com GMM, para intervalos de $k = 2,4$ até $k = 2,9$ e $k = 3,2$ até $k = 3,9$. Esses resultados conseguiram média 18,9 erros no total, ou seja menos de 1% de erro ao todo, com taxas de compressão de aproximadamente 85,5% em média.

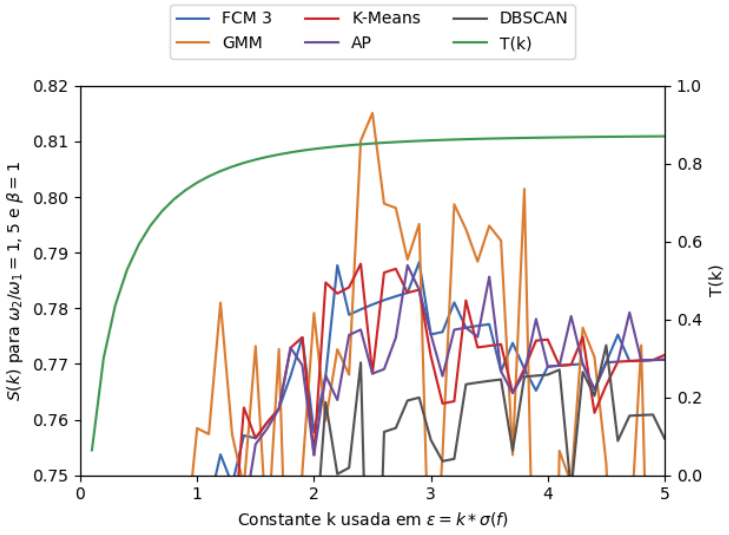
Em uma outra análise, podemos ver na Tablea 2 a quantidade de vezes que cada combinação alcança um $S(k) > 0,75$.

Tabela 2 – Quantidade de valores de k para qual cada combinação possui $S(k) > 0,75$.

Compressão	Clusterização	Número de Vezes
CHEB	FCM	38
CHEB	*K-Means*	37
CHEB	AP	36
CHEB	GMM	30
CHEB	DBSCAN	28
CHEB	*G-Means*	17
PWLH	AP	8
APCA	AP	3
PWLH	*G-Means*	2
PWLH	GMM	2
PCA	AP	1
PWLH	FCM	1
PWLH	*K-Means*	1
SF	*G-Means*	1

Vemos que o CHEB ocupa as seis primeiras posições com todos os algoritmos de clusterização, mostrando que os bons resultados na detecção de danos compensam a taxa menor de compressão. O GMM, que ocupa as dez primeiras colocações, não é mais o primeiro colocado, pois ele alcança picos altos, mas é menos constante. Nesse quesito o FCM se mostra o mais constante em relação a combinação com o CHEB, enquanto o AP aparece mais em todas as combinações.

Figura 22 – Valores de $S(k) > 0,75$ utilizando CHEB para compressão.



Fonte: Produzido pelo autor.

Os valores vistos em 22 confirmam o que foi discutido anteriormente, o GMM consegue alcançar valores altos de maneira menos constante que os outros. O FCM e o K-Means, além de serem os que mais aparecem tem as maiores médias de $S(k)$ entre essas

aparições com 0.771 e 0.772 respectivamente. Ambos atingem seu melhor desempenho no intervalo de k entre 2 e 3.

6.4 Análise dos resultados

Em relação aos algoritmos de compressão de dados, o CHEB foi bastante superior em todos os testes. Mesmo não tendo uma taxa de compressão tão alta, chegando somente próximo de 0.9 enquanto outros algoritmos conseguem chegar a quase 100%, ele consegue minimizar os erros tão bem que compensa.

Quanto a clusterização, combinados com o CHEB, os algoritmos FCM e *K-Means* possuem as melhores médias de $S(k)$, para $S(k) > 0,75$, além de serem os que mais alcançam valores acima de 0,75. O GMM por sua vez é o que alcança os valores de $S(k)$ mais altos em termos absolutos, mas não o faz de maneira consistente.

7 Considerações Finais

Esse trabalho apresentou um estudo comparativo para técnicas de compressão e detecção de danos em SHM utilizando uma métrica que junta ambos os aspectos em um só valor para facilitar a comparação entre elas. Essa métrica, *S-Measure*, calcula um valor de 0 a 1 para uma dada combinação.

Como técnica de deteção de dano foram utilizados algoritmos de aprendizado de máquina não supervisionados de clusterização. Esses algoritmos têm como objetivo agrupar os dados em conjuntos com base em semelhanças. As técnicas utilizadas para detecção de dano foram: K-Means, G-Means, Fuzy-C-Means (FCM), Density Based Spatial Clustering of Applications with Noise (DBSCAN), Gaussian Mixture Model (GMM) e Affinity Propagation.

Foi utilizado cinco algoritmos de compressão bastante mencionados na área de SHM. Todos eles são algoritmos de compressão com perda que garantem um erro máximo na aproximação. Esses algoritmos são: *Piecewise Constant Approximation* (PCA), *Adaptive Piecewise Constant Approximation* (APCA), *Piecewise Linear Histogram* (PWLH), *Slide Filter*s (SF) e *Chebyshev Approximation* (CHEB)

Foi concluído que para obter resultados bons de forma mais confiável é melhor utilizar a combinação de CHEB com FCM ou K-Means, de preferência com o valor de k entre 2 e 3. A combinação CHEB com GMM alcança resultados melhores em casos específicos, mas não é confiável.

7.1 Trabalhos Futuros

O primeiro passo para complementar esse estudo é validar esses resultados em outras bases de dados para descobrir o quanto eles podem ser extrapolados para novos sistemas. Pode se pensar também em outra validações nessa mesma base de dados utilizando dados de um conjunto maior de sensores e outros tipos de características extraídas.

Outro caminho para estender esse estudo é procurar novas técnicas de compressão e detecção de danos na literatura SHM e no estado da arte da área de compressão e de aprendizado de máquina. Para facilitar esse estudo comparativo, um outro projeto que pode surgir daqui é implementar esse framework de maneira genérica que seja facilmente estendido para outras técnicas e bases de dados, automatizando o processo de obter essas análises.

Referências

BEZDEK, J. C.; EHRLICH, R.; FULL, W. Fcm the fuzzy c-means clustering algorithm. *Computers & Geosciences*, v. 10, n. 2, p. 191–203, 1984. Citado na página 39.

BURAGOHAIN, C.; SHRIVASTAVA, N.; SURI, S. Space efficient streaming algorithms for the maximum error histogram. In: *2007 IEEE 23rd International Conference on Data Engineering*. [S.l.: s.n.], 2007. p. 1026–1035. ISSN 1063-6382. Citado na página 28.

CHAKRABARTI, K. et al. Locally adaptive dimensionality reduction for indexing large time series databases. *ACM Trans. Database Syst.*, ACM, New York, NY, USA, v. 27, n. 2, p. 188–228, jun. 2002. ISSN 0362-5915. Disponível em: <http://doi.acm.org/10.1145/568518.568520>. Citado na página 27.

CHENG, A. et al. *Data compression using Chebyshev transform*. Google Patents, 2004. US Patent App. 10/633,447. Disponível em: <https://www.google.tl/patents/US20040093364>. Citado na página 32.

DANIN, L. A. M. R. *Proposta de Método de Avaliação para Framawork de Compressão de Dados e Detecção de Danos em Monitoramento De Integridade Estrutural*. [S.l.]: Universidade Federal do Pará, 2016. (Trabalho de Conclusão de Curso (Bacharelado Ciência da Computação)). Citado 8 vezes nas páginas 15, 16, 17, 21, 22, 23, 24 e 44.

ELMELEEGY, H. et al. Online piece-wise linear approximation of numerical streams with precision guarantees. *Proc. VLDB Endow.*, VLDB Endowment, v. 2, n. 1, p. 145–156, ago. 2009. ISSN 2150-8097. Disponível em: <http://dx.doi.org/10.14778/1687627.1687645>. Citado 4 vezes nas páginas 13, 29, 30 e 31.

ESTER, M. et al. A density-based algorithm for discovering clusters in large spatial databases with noise. In: . [S.l.]: AAAI Press, 1996. p. 226–231. Citado 2 vezes nas páginas 37 e 38.

FARRAR, C.; SOHN, H.; PARK, G. A statistical pattern recognition paradigm for structural health monitoring. 01 2004. Citado na página 22.

FARRAR, C.; WORDEN, K. *Structural Health Monitoring: A Machine Learning Perspective*. Wiley, 2012. ISBN 9781118443217. Disponível em: <https://books.google.com.br/books?id=2w_sp6lersUC>. Citado 4 vezes nas páginas 13, 15, 19 e 22.

FARRAR, C. R.; DOEBLING, S. W.; NIX, D. A. Vibration–based structural damage identification. *Philosophical Transactions of the Royal Society of London A: Mathematical, Physical and Engineering Sciences*, The Royal Society, v. 359, n. 1778, p. 131–149, 2001. ISSN 1364-503X. Disponível em: <http://rsta.royalsocietypublishing.org/content/359/1778/131>. Citado na página 19.

FARRAR, C. R.; WORDEN, K. An introduction to structural health monitoring. In: _____. *New Trends in Vibration Based Structural Health Monitoring*. Vienna: Springer Vienna, 2010. p. 1–17. ISBN 978-3-7091-0399-9. Disponível em: <https://doi.org/10.1007/978-3-7091-0399-9_1>. Citado na página 34.

FIGUEIREDO, E.; MOLDOVAN, I.; MARQUES, M. J. B. *Data Compression: The Complete Reference*. Secaucus, NJ, USA: Springer-Verlag, 2006. Citado 4 vezes nas páginas 13, 19, 20 e 22.

FIGUEIREDO, E. et al. Machine learning algorithms for damage detection under operational and environmental variability. *Structural Health Monitoring*, v. 10, n. 6, p. 559–572, 2011. Disponível em: <https://doi.org/10.1177/1475921710388971>. Citado 3 vezes nas páginas 13, 22 e 34.

FIGUEIREDO, E. et al. A bayesian approach based on a markov-chain monte carlo method for damage detection under unknown sources of variability. v. 80, p. 1–10, 12 2014. Citado 2 vezes nas páginas 14 e 44.

FREY, B. J.; DUECK, D. Clustering by passing messages between data points. *Science*, v. 315, p. 972–976, 2007. Disponível em: <www.psi.toronto.edu/affinitypropagation>. Citado na página 41.

GLISIC, B.; INAUDI, D. *Fibre Optic Methods for Structural Health Monitoring*. Wiley, 2008. ISBN 9780470517802. Disponível em: <https://books.google.com.br/books?id=0dPYmueNAToC>. Citado na página 19.

HAMERLY, G.; ELKAN, C. Learning the k in k-means. In: *NIPS*. [S.l.: s.n.], 2003. Citado 2 vezes nas páginas 36 e 37.

HUNG, N. Q. V.; JEUNG, H.; ABERER, K. An evaluation of model-based approaches to sensor data compression. *IEEE Transactions on Knowledge and Data Engineering*, v. 25, p. 2434 – 2447, 2013. Citado 6 vezes nas páginas 14, 15, 21, 25, 29 e 32.

KANUNGO, T. et al. An efficient k-means clustering algorithm: analysis and implementation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, v. 24, n. 7, p. 881–892, Jul 2002. ISSN 0162-8828. Citado na página 35.

LAZARIDIS, I.; MEHROTRA, S. Capturing sensor-generated time series with quality guarantees. In: *Proceedings 19th International Conference on Data Engineering (Cat. No.03CH37405)*. [S.l.: s.n.], 2003. p. 429–440. Citado 2 vezes nas páginas 25 e 26.

LI, H.-N.; YI, T.-H.; QIAO, P. Special issue on health monitoring technologies for civil infrastructure. *Journal of Aerospace Engineering*, v. 30, n. 2, p. B2017001, 2017. Citado 2 vezes nas páginas 13 e 19.

LIMA, M. A. P. de et al. A framework for data compression and damage detection in structural health monitoring applied on a laboratory three-story structure. *Revista Brasileira de Computação Aplicada*, v. 8, n. 2, 2016. Citado 7 vezes nas páginas 13, 14, 15, 20, 21, 23 e 32.

MADDEN, S. Database abstractions for managing sensor network data. *Proceedings of the IEEE*, v. 98, p. 1879 – 1886, 2010. Citado 2 vezes nas páginas 13 e 25.

PARK, S. et al. Electro-mechanical impedance-based wireless structural health monitoring using pca-data compression and k-means clustering algorithms. *Journal of Intelligent Material Systems and Structures*, v. 19, n. 4, p. 509–520, 2008. Disponível em: <https://doi.org/10.1177/1045389X07077400>. Citado 2 vezes nas páginas 14 e 19.

PEETERS, B.; ROECK, G. D. One-year monitoring of the z24-bridge: environmental effects versus damage events. v. 30, p. 149–171, 02 2001. Citado 2 vezes nas páginas 43 e 44.

SALOMON, D. *Data Compression: The Complete Reference*. Secaucus, NJ, USA: Springer-Verlag, 2006. Citado 3 vezes nas páginas 13, 21 e 25.

SANTOS, A. et al. Output-only structural health monitoring based on mean shift clustering for vibration-based damage detection. 07 2016. Citado na página 13.

SHALEV-SHWARTZ, S.; BEN-DAVID, S. *Understanding Machine Learning: From Theory to Algorithms*. New York, NY, USA: Cambridge University Press, 2014. ISBN 1107057132, 9781107057135. Citado 2 vezes nas páginas 34 e 39.

TAN, P.; STEINBACH, M.; KUMAR, V. *Introduction to Data Mining*. Pearson Addison Wesley, 2006. (Always learning). ISBN 9780321321367. Disponível em: <https://books.google.com.br/books?id=YHsWngEACAAJ>. Citado na página 35.

TIBADUIZA, D. A.; MUJICA, L. E.; RODELLAR, J. Damage classification in structural health monitoring using principal component analysis and self-organizing maps. *Structural Control and Health Monitoring*, v. 20, n. 10, p. 1303–1316, 2013. ISSN 1545-2263. Disponível em: <http://dx.doi.org/10.1002/stc.1540>. Citado na página 14.

YANG, J. N. et al. Hilbert-huang based approach for structural damage detection. *Journal of Engineering Mechanics*, v. 130, n. 1, p. 85–95, 2004. Citado na página 22.

ZHOU, B.; JIN, H.; ZHENG, R. A parallel high speed lossless data compression algorithm in large-scale wireless sensor network. *International Journal of Distributed Sensor Networks*, v. 2015, p. 133, 2015. Citado 2 vezes nas páginas 15 e 25.