Visualização, kernels e subespaços: um estudo prático

Adriano Oliveira Barbosa

Adriano Oliveira Barbosa

Visualização, kernels e subespaços: um estudo prático

Tese apresentada ao Instituto de Ciências Matemáticas e de Computação – ICMC-USP, como parte dos requisitos para obtenção do título de Doutor em Ciências – Ciências de Computação e Matemática Computacional. VERSÃO REVISADA

Área de Concentração: Ciências de Computação e Matemática Computacional

Orientador: Prof. Dr. Luis Gustavo Nonato

USP – São Carlos
Fevereiro de 2017

Ficha catalográfica elaborada pela Biblioteca Prof. Achille Bassi
e Seção Técnica de Informática, ICMC/USP,
com os dados fornecidos pelo(a) autor(a)

Oliveira Barbosa, Adriano
O238v Visualização, kernels e subespaços: um estudo prático / Adriano Oliveira Barbosa; orientador Luis Gustavo Nonato. -- São Carlos, 2016.
89 p.

Tese (Doutorado - Programa de Pós-Graduação em Ciências de Computação e Matemática Computacional) -- Instituto de Ciências Matemáticas e de Computação, Universidade de São Paulo, 2016.

1. kernel. 2. subspace clustering. 3. projeção multidimensional. 4. visualização. I. Gustavo Nonato, Luis, orient. II. Título.

Adriano Oliveira Barbosa

Visualization, kernels and subspaces: a practical study

Doctoral dissertation submitted to the Instituto de Ciências Matemáticas e de Computação – ICMC-USP, in partial fulfillment of the requirements for the degree of the Doctorate Program in Computer Science and Computational Mathematics. FINAL VERSION

Concentration Area: Computer Science and Computational Mathematics

Advisor: Prof. Dr. Luis Gustavo Nonato

USP – São Carlos
February 2017

Este trabalho é dedicado a minha esposa e meus pais.

AGRADECIMENTOS

Agradeço aos meus pais, Cicero e Cicera, que sempre investiram em minha educação e acreditaram em meus sonhos. À minha esposa, Karla, pelo incondicional apoio, dedicação, paciência, cumplicidade e companherismo em todos os momentos.

Agradeço a todos os amigos, professores e funcionários da Universidade de São Paulo pelo apoio durante essa jornada, em especial aos grandes amigos que fiz no LCAD/LMACC e aos professores que ministraram as disciplinas durante o doutorado. Pelas ótimas lembranças dos presentes e dos que se foram.

Agradeço ao grande amigo Douglas pelas infinitas discussões e ajuda.

Agradeço aos colaboradores acadêmicos Afonso, Fabiano, Filip, Fernando e Siome pela imensa ajuda.

Agradeço imensamente ao meu orientador Gustavo pela ajuda, tempo dedicado e pelo exemplo de profissional.

Agradeço à CAPES e à Universidade de São Paulo pelo apoio financeiro.

“ Se A é o sucesso, então A é igual a X mais Y mais Z.
O trabalho é X; Y é o lazer; e Z é manter a boca fechada.”
(Albert Einstein)

RESUMO

BARBOSA, A. O. Visualização, kernels e subespaços: um estudo prático. 2017. 89 p. Tese (Doutorado em Ciências – Ciências de Computação e Matemática Computacional) – Instituto de Ciências Matemáticas e de Computação, Universidade de São Paulo, São Carlos – SP, 2017.

Dados de alta dimensão são tipicamente tratados como pertencentes a um único subespaço do espaço onde estão imersos. Entretanto, dados utilizados em aplicações reais estão usualmente distribuídos entre subespaços independentes e com dimensões distintas. Um objeto de estudo surge a partir dessa afirmação: como essa distribuição em subespaços independentes pode auxiliar tarefas de visualização?

Por outro lado, se o dado parece estar embaralhado nesse espaço de alta dimensão, como visualizar seus padrões e realizar tarefas como classificação? Podemos, por exemplo, mapear esse dado num outro espaço utilizando uma função capaz de o desembaralhar, de modo que os padrões intrínsecos fiquem mais claros e, assim, facilitando nossa tarefa de visualização ou classificação.

Essa Tese apresenta dois estudos que abordam ambos os problemas. Para o primeiro, utilizamos técnicas de subspace clustering para definir, quando existente, a estrutura de subespaços do dado e estudamos como essa informação pode auxiliar em visualizações utilizando projeções multidimensionais. Para o segundo problema, métodos de kernel, bastante conhecidos na literatura, são as ferramentas a nos auxiliar. Utilizamos a medida de similaridade do kernel para desenvolver uma nova técnica de projeção multidimensional capaz de lidar com dados imersos no espaço de características induzido implicitamente pelo kernel.

Palavras-chave: kernel, subspace clustering, projeção multidimensional, visualização.

ABSTRACT

BARBOSA, A. O. Visualization, kernels and subspaces: a practical study. 2017. 89 p. Tese (Doutorado em Ciências – Ciências de Computação e Matemática Computacional) – Instituto de Ciências Matemáticas e de Computação, Universidade de São Paulo, São Carlos – SP, 2017.

High-dimensional data are typically handled as laying in a single subspace of the original space. However, data involved in real applications are usually spread around in distinct subspaces which may have different dimensions. We would like to study how the subspace structure information can be used to improve visualization tasks.

On the other hand, what if the data is tangled in this high-dimensional space, how to visualize it's patterns or how to accomplish classification tasks? One could, for example, map the data in another high-dimensional space using a mapping capable of untangle the data making the patterns clear, rendering the visualization or classification an easy task.

This dissertation presents an study for both problems pointed out above. For the former, we use subspace clustering techniques to define, when it exists, a subspace structure, studying how this information can be used to support visualization tasks based on multidimensional projections. For the latter problem we employ kernel methods, well known in the literature, as a tool to assist visualization tasks. We use a similarity measure given by the kernel to develop a completely new multidimensional projection technique capable of dealing with data embedded in the implicit feature space defined by the kernel.

Keywords: kernel methods, subspace clustering, multidimensional projection, visualization.

LISTA DE ILUSTRAÇÕES

LISTA DE ALGORITMOS

LISTA DE TABELAS

LISTA DE ABREVIATURAS E SIGLAS

ALM	Augmented Lagrange Multiplier (LIN et al., 2009)
CSMP	Class-Specific Multidimensional Projection (JOIA et al., 2012)
Kelp	Kernel-based Linear Projection (BARBOSA et al., 2016)
KPCA	Kernel PCA (SCHÖLKOPF; SMOLA; MÜLLER, 1998)
LAMP	Local Affine Multidimensional Projection (JOIA et al., 2011)
LDA	Linear Discriminant Analysis (BISHOP, 2006)
LLE	Locally Linear Embedding (HAM et al., 2004)
LRR	Low-Rank Representation (ZHANG et al., 2014)
LSP	Least Square Projection (PAULOVICH et al., 2008)
MDS	Multidimensional Scaling (JOURDAN; MELANCON, 2004)
PCA	Principal Component Analysis (BISHOP, 2006)
PLMP	Part-Linear Multidimensional Projection (PAULOVICH; SILVA; NONATO, 2010)
PLP	Piece-wise Laplacian-based Projection (PAULOVICH et al., 2011)
SVD	Decomposição em Valores Singulares (BISHOP, 2006)
SVM	Support Vector Machine (SCHÖLKOPF; SMOLA, 2001)
t-SNE	t-Distributed Stochastic Neighbor Embedding (MAATEN; HINTON, 2008)

LISTA DE SÍMBOLOS

R^n — Espaço cartesiano de dimensão n

H — Espaço de características (Hilbert) definido implicitamente pelo kernel

$k(x, y)$ — Função kernel calculada entre as instâncias x e y

$x^T y$ — Produto interno entre os vetores x e y

$\varphi(x)$ — Mapeamento implícito definido pela função kernel

M_{ij} — Entrada i j da matriz M

$M(i, :)$ — i-ésima linha da matriz M

$M(:, j)$ — j-ésima coluna da matriz M

$M\ (m \times n)$ — Espaço das matrizes $m \times n$

$\mathbb{1}_m$ — Matriz $m \times m$ cujas entradas são todas iguais a $\frac{1}{m}$

1_n — Vetor de dimensão n cujas entradas são todas iguais a $\frac{1}{n}$

$\langle x, y \rangle_H$ — Produto interno entre as instâncias x e y no espaço H

$[M, N]$ — Concatenação das matrizes M e N horizontalmente

$[M; N]$ — Concatenação das matrizes M e N verticalmente

SUMÁRIO

CAPÍTULO

1

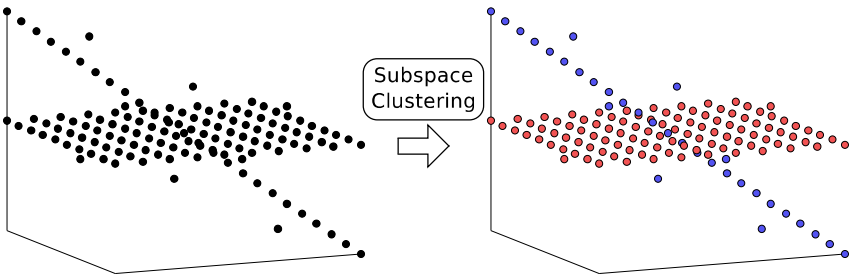
INTRODUÇÃO

Dados de alta dimensão podem ser coletados de várias fontes, incluindo pesquisa de campo, experimentos físicos e coleções de imagens. Esse tipo de dado é usualmente descrito em termos de coordenadas em um espaço cartesiano de alta dimensão (R^n com n grande). Como não conseguimos visualizar esse espaço diretamente, utilizamos ferramentas matemáticas e computacionais, como projeção multidimensional, que tem a capacidade de processar e apresentar esse tipo de dado de uma forma compreensível.

Algumas técnicas de projeção multidimensional assumem que os dados estão contidos ou próximo a um único subespaço ou variedade de dimensão menor que o espaço cuja dimensão é dada pelo número de atributos dos dados. Entretanto, não há razão para que essa suposição seja sempre verdade. Suponha que as instâncias de dado estão imersas em subespaços independentes. Essa suposição nos leva diretamente para um problema de segmentação de subespaços (subspace clustering), cujo objetivo é segmentar as instâncias em grupos, cada um pertencendo a um subespaço. Técnicas de subspace clustering são capazes de detectar estrutura de subespaços, mesmo que a quantidade e dimensão dos subespaços seja desconhecida. A Figura 1 ilustra o funcionamento de uma técnica de subspace clustering onde o dado é composto por instâncias pertencentes a um pedaço de plano e a um segmento de reta não paralelos, ambos com ruído e imersos no espaço tridimensional. As instâncias são identificadas como pertencentes a um dos subespaços. Tais técnicas mostram sua importância em várias áreas como visão computacional (COSTEIRA; KANADE, 1998; KANATANI, 2001; ZELNIK-MANOR; IRANI, 2003), processamento de imagem (YANG et al., 2008; HONG et al., 2006) e sistemas de identificação (ZHANG; BITMEAD, 2005)

Técnicas de projeção multidimensional são ferramentas importantes por conseguir lidar com um grande número de atributos e instâncias (PAULOVICH et al., 2011), tornando-as ideais para visualização de dados de alta dimensão. Uma vez que detecção de subespaço tem se mostrado útil em diversos contextos, seria possível empregá-las em conjunto com projeções

Figura 1 – Ilustração da aplicação de uma técnica de subspace clustering: dados contidos em subespaços de dimensão 1 e 2 imersos em R^3 são classificados.



multidimensional a fim de gerar visualizações ainda mais informativas?

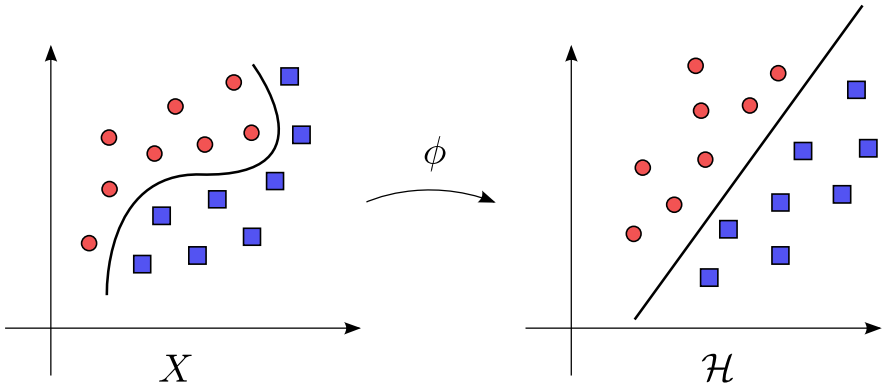
No Capítulo 3 apresentaremos um estudo (BARBOSA; SADLO; NONATO, 2015) sobre efeito da aplicação de técnicas de subspace clustering, como a Low-Rank Representation (ZHANG et al., 2014) (LRR), no auxílio a projeções multidimensionais. Utilizamos a segmentação em subespaços gerada pela LRR como rótulo para as instâncias de dado e efetuamos redução de dimensionalidade com Linear Discriminant Analysis (BISHOP, 2006) (LDA). Comparamos a qualidade dessas projeções com técnicas bastante conhecidas na literatura como a Local Affine Multidimensional Projection (JOIA et al., 2011) (LAMP) e t-Distributed Stochastic Neighbor Embedding (MAATEN; HINTON, 2008) (t-SNE). Esse é um primeiro estudo no sentido de aplicar técnicas de subspace clustering combinadas com técnicas de visualização e redução de dimensionalidade. Além disso, propomos uma mudança na LAMP para que possa fazer uso da informação de subespaços do dado durante o processo de projeção.

Outra grande ferramenta que surge para auxiliar no estudo de projeções multidimensionais envolve técnicas de kernel. O crescente interesse nesses métodos é motivado pelos resultados positivos obtidos em aplicações de agrupamento e classificação (SCHÖLKOPF; SMOLA, 2001). Funções kernel podem ser vistas como medidas de similaridade entre instâncias de dados e fornecem uma maneira de transformar dados para um novo espaço de características (espaço de Hilbert H), permitindo que estruturas e padrões presentes no dado sejam melhor caracterizados neste espaço. Dessa forma, tarefas de classificação e agrupamento podem ser efetuadas com mais facilidade e eficiência.

Associado a uma função kernel existe uma mapeamento implícito φ responsável por levar o dado do seu espaço original para o espaço de características H (dado kernelizado). Esse mapeamento é geralmente difícil de ser encontrado, tornando complexo o entendimento de estruturas intrínsecas dos dados, como vizinhança. A Figura 2 ilustra o mapeamento φ associado a uma função kernel transformando uma estrutura não linear no espaço original numa estrutura linear no espaço de características.

Entender como um kernel realiza o mapeamento dos dados no espaço de características é muito importante, principalmente para auxiliar a confecção de kernels com propriedades específicas e entender como as relações de vizinhança são afetadas.

Figura 2 – Imersão dos dados X no espaço de alta dimensão H transformando um padrão não linear numa estrutura linear.



No Capítulo 4 propomos uma técnica de visualização de informação chamada Kernel-based Linear Projection (BARBOSA et al., 2016) (Kelp) capaz de lidar com dados kernelizados, mesmo sem conhecer suas coordenadas. A Kelp permite visualizar como a imersão realizada por um kernel no espaço de características afeta a estrutura de vizinhança dos dados. Nossa técnica é composta por dois passos principais. Primeiramente, um subconjunto pequeno de instâncias – pontos de controle – é projetado no espaço visual através de uma estratégia de forças que busca minimizar uma função de energia. Em seguida, todo o conjunto de dados é projetado tomando como referência o posicionamento dos pontos de controle. Fazemos também uma comparação entre os resultados obtidos pela Kelp e técnicas de projeção multidimensional que utilizam distância como dado de entrada.

A ideia do método é assumir que conhecemos a imersão φ dos dados no espaço de características associada a função kernel, ou seja, sua imagem nesse espaço. Com base na posição dos pontos de controle no espaço visual, definimos uma transformação linear que interpola tal posicionamento e projeta o conjunto de dados utilizando essa transformação linear. As contribuições desse trabalho incluem uma nova técnica de projeção multidimensional com base matemática sólida capaz de visualizar dados kernelizados. Além disso, combinamos a Kelp com uma versão kernelizada das coordenadas diferenciais (LIPMAN et al., 2004) a fim de entender como a estrutura de vizinhança dos dados é afetada durante o mapeamento da função φ associada ao kernel.

As principais contribuições desta tese são:

- utilizamos técnicas de subspace clustering combinadas com técnicas de visualização para redução de dimensionalidade;
- um estudo da eficácia da segmentação em subespaços nas tarefas de visualização comparando técnicas que necessitam da informação de classe;
- uma modificação na LAMP para que possa utilizar a informação de classe;

- uma nova técnica de projeção multidimensional chamda Kelp capaz de lidar com dados kernelizados;
- o uso da Kelp, como ferramenta de visualização, para auxiliar em aplicações baseadas em kernel; por exemplo, classificação e segmentação de imagem;
- uma versão kernelizada das coordenadas diferenciais combinada com a Kelp a fim de entender como o kernel afeta as estruturas de vizinhança do dado durante o mapeamento para o espaço de características.

Esta tese está estruturada da seguinte forma:

Capítulo 2: fazemos uma revisão bibliográfica dos principais trabalhos das áreas de subspace clustering e projeção multidimensional, incluindo alguns trabalhos que utilizam técnicas de kernel;

Capítulo 3: descrevemos nosso estudo sobre o efeito do uso das técnicas de subspace clustering no auxílio de projeções multidimensionais (BARBOSA; SADLO; NONATO, 2015);

Capítulo 4: descrevemos a técnica Kelp (BARBOSA et al., 2016), principal resultado do doutorado;

Capítulo 5: conclusões sobre os trabalhos e sobre suas contribuições;

Apêndice: contém as demonstrações dos resultados mais relevantes utilizados na fundamentação teórica dos problemas tratados na tese.

CAPÍTULO

2

TRABALHOS RELACIONADOS

Apresentaremos trabalhos relevantes na área de subspace clustering e visualização, tanto no que diz respeito a projeções multidimensionais quanto na utilização da teoria de kernel para fins de visualização. Os trabalhos discutidos neste capítulo foram escolhidos por se tratarem de estudos clássicos da área ou por representarem o estado da arte no contexto da tese. As técnicas discutidas possuem uma característica em comum: possibilidade do usuário intervir na projeção, ou seja, são técnicas assistidas pelo usuário.

2.1 Subspace clustering

Em vários problemas, o dado é representado como instâncias imersas em um espaço de alta dimensão, frequentemente na vizinhança de subespaços independentes desse espaço. Motivado por esse fato, uma série de técnicas vêm sendo desenvolvidas para detectar tal estrutura de subespaço. Por exemplo, Principal Component Analysis (BISHOP, 2006) (PCA) assume que o dado pertence a um único subespaço e busca as direções de maior variância dos dados para formar a base do subespaço procurado. Entretanto, supor que o dado pode estar contido num único subespaço é bem restritivo. Técnicas de subspace clustering assumem que o dado pode estar distribuido próximo a vários subespaços independentes, mesmo sem saber a quantidade desses subespaços ou a qual subespaço cada instância de dado pertence. Quando o número de subespaços é igual a 1, o problema se reduz ao PCA. O objetivo das técnicas de subspace clustering é estimar, caso exista, a estrutura de subespaços, o número e a dimensão de cada subespaço para, em seguida, determinar a qual subespaço cada instância pertence.

Alguns problemas principais surgem quando tentamos estimar a estrutura de subespaços presente nos dados:

- A distribuição dos dados nos subespaços é geralmente desconhecida. Se os dados em cada subespaço estão distribuidos em grupos e esses grupos são distantes entre si, o problema

se torna mais simples. Por outro lado, se a distribuição dos dados se dá de modo que as instâncias estão próximas da interseção dos subespaços, então o problema acaba se tornando mais difícil;

- Os dados podem estar corrompidos, conter ruído e outliers, por exemplo. Esse problema pode fazer com que a estimativa dos subespaços seja completamente errada;

- Escolha do modelo de estimativa: no PCA, basta saber a dimensão do subespaço (que pode ser encontrada buscando o subespaço de menor dimensão que comporta os dados). Por outro lado, no caso de múltiplos subespaços, podemos encaixar os dados desde n subespaços de dimensão 1 a um único subespaço de dimensão d. A dificuldade está em definir a combinação do número de subespaços e suas dimensões.

Técnicas de subspace clustering podem ser divididas em quatro tipos: algébricas (GEAR, 1998; COSTEIRA; KANADE, 1998; BOULT; BROWN, 1991), as quais estão interessadas em obter a separação do dado a partir da fatoração da matriz do mesmo em matrizes de posto baixo; iterativas (AGARWAL; MUSTAFA, 2004; BRADLEY; MANGASARIAN; PARDALOS, 2000; TSENG, 2000), concebidas para melhorar os resultados obtidos pelos algoritmos algébricos em dados com ruído — utilizam a segmentação obtida pelos algoritmos algébricos e aplicam PCA para cada subespaço, iterando entre esse dois passos e refinando o resultado de segmentação do dado em subespaços. Enquanto as técnicas acima utilizam propriedades algébricas e geométricas, não estão preocupadas com a distribuição do dado (ou do ruído) nos subespaços. Para preencher essa lacuna, surgem as técnicas denominadas estatísticas (TIPPING; BISHOP, 1999; DERKSEN et al., 2007). A classe de métodos espectrais (ELHAMIFAR; VIDAL, 2009; ZHANG et al., 2014) gera uma matriz de afinidade cuja entrada ij mede a similaridade entre as instâncias i e j. Idealmente, entradas com valor 1 significam que as instâncias i e j pertencem ao mesmo subespaço. De posse de uma matriz desse tipo, a segmentação do dado nos seus respectivos subespaços pode ser obtida aplicando o algoritmo k-means aos autovetores da matriz Laplaciana associada a matriz de afinidade.

Um bom tutorial sobre subspace clustering pode ser encontrado em (VIDAL, 2010). A técnica Low-Rank Representation (LRR) (Capítulo 3) busca, como o nome diz, uma representação (decomposição) em função de uma matriz de posto baixo através de um processo de otimização com restrição. A LRR é capaz de resolver os três principais problemas descritos acima, estimando o número de subespaços e gerando uma correspondência entre as instâncias de dado e os subespaços. Esta foi a técnica utilizada no estudo descrito no Capítulo 3.

Um trabalho envolvendo subspace clustering e visualização pode ser encontrado em (LIU et al., 2015), onde a base e a dimensão dos subespaços são estimados utilizando a distância Grassmanniana, permitindo uma exploração visual interativa do dado através de projeções dinâmicas.

2.2 Projeções Multidimensionais

Sammon (1969) desenvolveu uma técnica cujo objetivo é criar uma projeção dos dados multidimensionais de forma que as distâncias no espaço original sejam preservadas da melhor forma possível, baseado na medida de stress

$$E = \frac{1}{\sum_{i<j} \delta_{ij}} \sum_{i<j} \frac{(\delta_{ij} - d_{ij})^2}{\delta_{ij}}$$

onde d_{ij} representa a distância entre a instância i e a instância j no espaço original e δ_{ij} a distância entre as mesmas instâncias no espaço visual. Inicialmente as instâncias são mapeadas no espaço visual (podendo ser posicionados aleatoriamente) e em seguida o stress dessa configuração é calculado. A partir daí, o posicionamento dos pontos no espaço visual é alterado em pequenos passos de modo a melhorar o resultado da função de erro dado pelo stress. Esse procedimento é repetido até que seja encontrado um mínimo da função de erro.

Dois problemas são inerentes à técnica de Sammon: a complexidade do algoritmo — pois o cálculo da função de stress tem complexidade $O(n^2)$, onde n indica a quantidade de pontos a serem projetados — e o fato de ser necessário refazer todo o processo caso um novo ponto precise ser projetado.

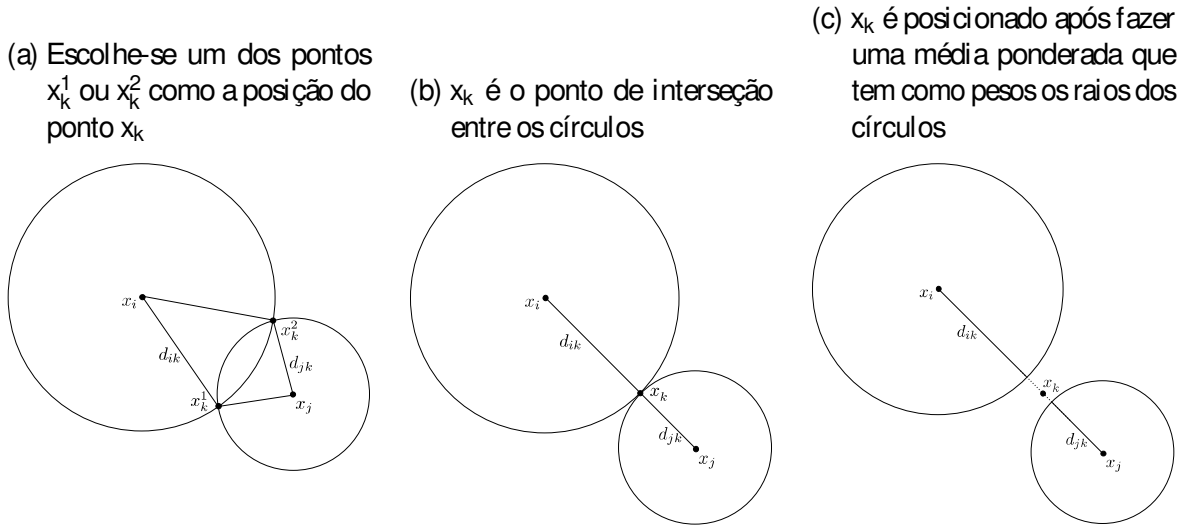
Para contornar as dificuldades inerentes ao Sammon Mapping, Pekalska et al. (1999) apresentou uma generalização da técnica de Sammon. Primeiramente, foi realizado um estudo sobre os algoritmos que poderiam ser utilizados para efetuar a minimização do stress. Em seguida foi proposta uma forma de efetuar a projeção em função de um subconjunto dos dados. Na proposta de Pekalska, um subconjunto dos dados seria projetado seguindo o algoritmo original e a partir do posicionamento deles o resto dos dados seria posicionado de modo a evitar distorções no stress.

Três maneiras de posicionar as instâncias restantes foram estudadas: triangulação, redes neurais e mapeamento de distância. A primeira utiliza dois pontos já projetados como guia para a projeção de cada novo ponto como ilustrado na Figura 3.

A abordagem por redes neurais utiliza como conjunto de treinamento o dado original e a saída produzida é a projeção no espaço visual. Desejamos que a saída produzida possa ser ilimitada, entretanto nesse tipo de abordagem é comum o uso de funções de ativação como a sigmoide $f(x) = \frac{1}{1+e^{-x}}$ que retorna valores no intervalo $[0,1]$. Para contornar essa dificuldade, pode-se utilizar outra função de ativação, como uma função linear, por exemplo. Por outro lado, funções de ativação lineares tornam o treino mais complexo, uma vez que é mais difícil encontrar um mínimo para a função de erro.

A última proposta é definir uma transformação linear T que atua na matriz de distâncias de uma amostra dos dados no espaço original, D_S, resultando no posicionamento dos pontos no espaço visual Y_S. Para um conjunto de dados n dimensional com N instâncias, toma-se uma

Figura 3 – A posição do novo ponto x_k é dada pela interseção dos círculos definidos pelos pontos já projetados, x_i e x_j, e com raio igual a distância entre o ponto projetado e o novo ponto no espaço original.



amostra com k ($k < N$) instâncias e define-se a transformação linear:

$$T_{p\times k} D_{S_{k\times k}} = Y_{S_{p\times k}}$$

e utiliza-se essa transformação linear para mapear as instâncias restantes

$$T_{p\times k} D'_{k\times (N-k)} = Y'_{p\times (N-k)}$$

resultando na projeção dos $N - k$ pontos restantes.

Paulovich et al. (2008) apresenta a técnica chamada Least Square Projection (PAULOVICH et al., 2008) (LSP), a qual é baseada em aproximações feitas por mínimos quadrados. A partir da projeção de pontos de controle ($\{p_1 \ldots p_h\}$) por um método de Multidimensional Scaling (JOURDAN; MELANCON, 2004) (MDS), são calculadas vizinhanças ($\{V_1 \ldots V_h\}$) de cada ponto e a partir de cada uma é definida uma matriz que dá origem a um conjunto de sistemas lineares:

$$Lx_1 = 0 \ldots Lx_2 = 0 \ldots Lx_d = 0$$

onde L é uma matriz $n \times n$ com entradas

$$l_{ij} = \begin{cases} 1 & i = j \\ -\alpha_{ij} & p_j \in V_i \\ 0 & \text{caso contrário.} \end{cases}$$

Os parâmetros α_{ij} acima são tais que

$$\tilde{p}_i - \sum_{p_j \in V_i} \alpha_{ij} \tilde{p}_j = 0 \qquad (2.1)$$

$$0 \leq \alpha_{ij} \leq 1; \sum \alpha_{ij} = 1$$

e $\tilde{p}_i$ são as coordenadas de p_i no espaço visual.

Problemas com o posto das matrizes levaram Paulovich et al. (2008) a acoplar linhas que representam pontos de controle ao sistema linear, tornando o sistema sobredeterminado e resolvendo-o utilizando o método dos mínimos quadrados. Dessa forma, o sistema a ser resolvido é

$$Ax = b$$

onde A tem dimensões $(n + nc) \times n$ e é da forma

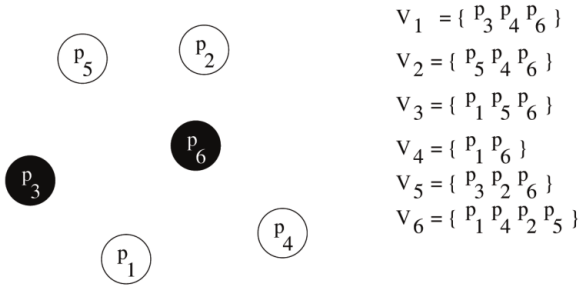
$$A = \begin{matrix} L \\ C \end{matrix} \quad \text{com } c_{ij} = \begin{cases} 1 & \text{se } p_j \text{ é um ponto de controle} \\ 0 & \text{caso contrário} \end{cases}$$

e

$$b_i = \begin{cases} 0 & i \leq n \\ x_{p_{c_i}} & n < i \leq n + nc \end{cases}$$

onde $x_{p_{c_i}}$ é uma das coordenadas do ponto de controle p_{c_i}. A Figura 4 ilustra a construção do sistema para um conjunto de seis pontos. As soluções dos sistemas, dadas pelo método dos mínimos quadrados, retornam as projeções.

Figura 4 – Matriz A com pontos de controle p_3 e p_6. Figura retirada de (PAULOVICH et al., 2008)



$$A = \begin{matrix} L \\ \left[\begin{array}{cccccc} 1 & 0 & -1/3 & -1/3 & 0 & -1/3 \\ 0 & 1 & 0 & -1/3 & -1/3 & -1/3 \\ -1/3 & 0 & 1 & 0 & -1/3 & -1/3 \\ -1/2 & 0 & 0 & 1 & 0 & -1/2 \\ 0 & -1/3 & -1/3 & 0 & 1 & -1/3 \\ -1/4 & -1/4 & 0 & -1/4 & -1/4 & 1 \end{array}\right] \\ \left[\begin{array}{cccccc} 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{array}\right] \\ C \end{matrix}$$

O algoritmo possui complexidade computacional $O(k^2 + n^2)$, onde k é o número de pontos de controle e n a cardinalidade do conjunto de dados.

Outra técnica que faz projeção multidimensional é a Part-Linear Multidimensional Projection (PAULOVICH; SILVA; NONATO, 2010) (PLMP). O trabalho apresenta uma técnica que tenta preservar a distância entre instâncias de dado no espaço original: matematicamente falando, a projeção Φ deve satisfazer

$$\Phi = \underset{\hat{\Phi} \in L\,(\mathbb{R}^m \to \mathbb{R}^n)}{\arg\min} \left(\frac{1}{D} \sum_{i \neq j} d(x_i, x_j) - d(\hat{\Phi}(x_i), \hat{\Phi}(x_j)) \right)^2$$

onde $D = \sum_{i,j} d(x_i, x_j)^2$ e $L \in (R^m, R^n)$ é o espaço das funções lineares de R^m em R^n.

Para isso, o método utiliza algumas amostras dos dados $X' = \{x'_1, \ldots, x'_k\}$, $k < n$, já projetadas de modo a preservar distância da melhor forma possível. Daí, a projeção Φ deve satisfazer

$$\Phi(x'_i) = \bar{x}'_i \tag{2.2}$$

para todo $i = 1, \ldots, k$, onde $\bar{x}'_i$ é a projeção de x'_i no espaço visual.

Para cada i, a equação (2.2) nos diz que

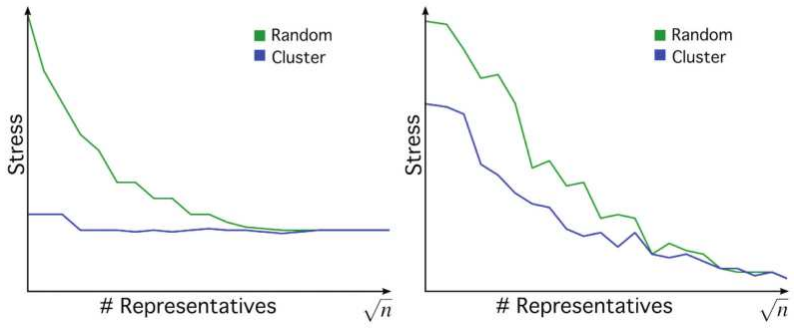
$$\begin{aligned} \varphi_1 x'_{1i} + \cdots + \varphi_m x'_{1m} &= \bar{x}_{i1} \\ \varphi_1 x'_{2i} + \cdots + \varphi_m x'_{2m} &= \bar{x}_{i2} \\ \vdots \qquad\qquad\qquad &\quad \vdots \\ \varphi_1 x'_{ki} + \cdots + \varphi_m x'_{km} &= \bar{x}_{ik} \end{aligned} \tag{2.3}$$

onde φ_{ij} é a entrada da matriz Φ na linha i e coluna j. Reescrevendo o sistema (2.3) como $L\varphi_i = \bar{x}_i$, com $L_{k \times m}$, e assumindo que k é maior que m, Paulovich, Silva e Nonato (2010) resolvem o sistema através da solução da equação normal

$$L^T L \varphi_i = L^T \bar{x}_i \tag{2.4}$$

Visto que a projeção final depende da escolha dos pontos de controle, Paulovich, Silva e Nonato (2010) fazem uma comparação entre duas formas de escolhê-los, agrupando ou escolhendo aleatoriamente. A Figura 5 mostra que a medida de stress estabiliza tanto quando se utiliza uma escolha aleatória quanto utilizando agrupamento quando a quantidade de amostras é próxima de $\sqrt{n}$.

Figura 5 – Comparação entre escolha por agrupamento e aleatória testada em dois conjuntos de dados diferentes. Figura retirada de (PAULOVICH; SILVA; NONATO, 2010)



Paulovich, Silva e Nonato (2010) fazem uma análise de custo computacional e o número de amostras $k = \sqrt{n}$ se apresenta uma boa escolha, pois gera projeções de qualidade adequada e fazem o custo computacional do algoritmo ser linear (após o cálculo da transformação L) com relação a quantidade de instâncias de dado.

A PLMP apresenta as mesmas limitações que a LSP, pois se trata de uma projeção de natureza global.

Uma técnica cujo ponto forte é a interatividade é a Piece-wise Laplacian-based Projection (PAULOVICH et al., 2011) (PLP). Ela é desenvolvida em três passos: amostragem, construção do grafo de vizinhança e solução do sistema linear Laplaciano. A escolha dos pontos de controle pode ser feita de acordo com a aplicação. Essa escolha funciona de forma semelhante à feita na PLMP (aleatoriamente ou agrupamento, por exemplo). Essas amostras são utilizadas para dividir o conjunto de dados em subconjuntos que irão gerar o grafo de vizinhança que dá origem ao sistema Laplaciano. Sejam p_i uma instância de dado que pertence ao subconjunto D_i e $Viz(p_i) = \{p_{i_1}, \ldots, p_{i_k}\}$ o conjunto de nós conectados a p_i no grafo de vizinhança de D_i. Analogamente a LSP, a PLP assume a hipótese da combinação convexa (equação (2.1)), dessa forma:

$$\tilde{p}_i = (x_{p_i}, y_{p_i}) = \sum_{p_{i_j} \in Viz(p_i)} \alpha_{ij} (x_{p_{i_j}}, y_{p_{i_j}})$$

onde $\alpha_{ij} > 0$, $\sum \alpha_{ij} = 1$ e $(x_{p_{i_j}}, y_{p_{i_j}})$ são as coordenadas de p_{i_j} no espaço visual.

Daí são derivados dois sistemas lineares:

$$Lx = 0 \quad e \quad Ly = 0$$

onde

$$l_{ij} = \begin{cases} 1 & \text{se } i = j \\ -\alpha_{ij}/\alpha_i^* & \text{se } i \neq j \text{ e } p_{i_j} \in Viz(p_i) \\ 0 & \text{caso contrário} \end{cases}$$

e $\alpha_i^* = \sum_{p_{i_j} \in Viz(p_i)} \alpha_{ij}$. Resolver os sistemas resulta em encontrar a projeção.

Outra contribuição do trabalho foi o desenvolvimento de um mecanismo para definir vizinhança no espaço de alta dimensão a partir da vizinhança no espaço visual. A interatividade é utilizada no ajuste dos pontos projetados a fim de melhorar a projeção. Entretanto, esses ajustes fazem o custo computacional da técnica ser elevado. A Figura 6 apresenta um resultado alcançado pela PLP.

A técnica Local Affine Multidimensional Projection (LAMP) (JOIA et al., 2011) foca na flexibilidade e interatividade com o usuário. O método calcula, para cada ponto, uma aplicação afim

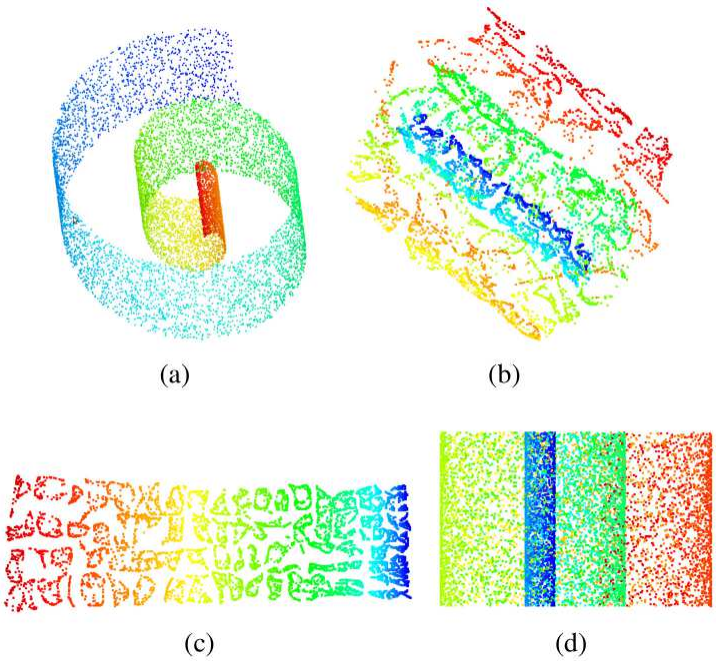
$$\begin{aligned} f : R^n &\longrightarrow R^2 \\ p &\mapsto f(p) = Mp + t \end{aligned}$$

que mapeia os pontos no espaço visual obedecendo a restrição $MM^T = I$. A translação t é escrita em função da matriz M e, com isso, para calcular a transformação f é suficiente calcular a matriz M. A restrição garante projeções com melhor qualidade. O cálculo da matriz M é feito através de um processo de otimização idêntico ao Problema de Procrustes (GOWER; DIJKSTERHUIS, 2004), cuja solução é dada por

$$M = UV^T, A^T B = UDV$$

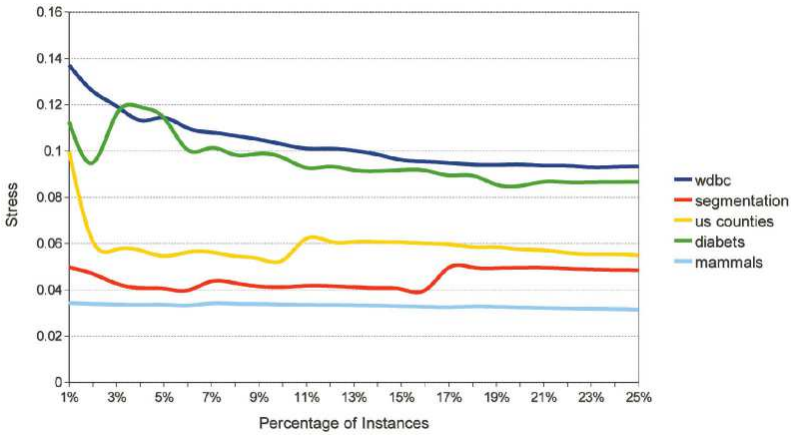
onde UDV é a decomposição em valores singulares de $A^T B$. As matrizes A e B são matrizes construídas durante o processo de minimização. A grande vantagem da técnica se deve ao fato de

Figura 6 – (a) Swiss Roll. (b) PLP usando Force Scheme e distância euclidiana para posicionar os pontos de controle. (c) PLP usando Isomap para posicionar os pontos de controle. (d) PLMP usando Isomap para posicionar os pontos de controle. Figura retirada de (PAULOVICH et al., 2011)



utilizar pouquíssimos pontos de controle quando comparado as demais e ainda assim conseguir apresentar resultados muito bons com a métrica de stress, como ilutrado na Figura 7.

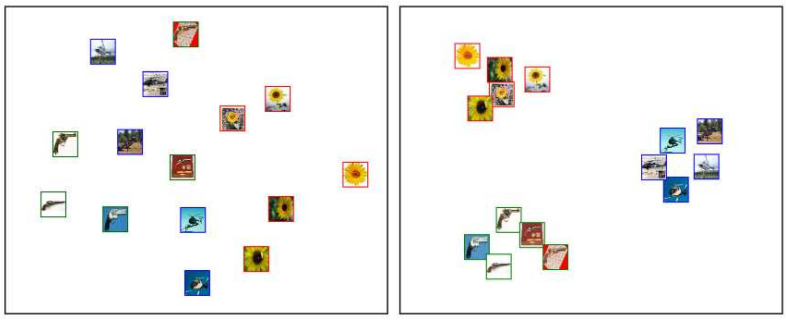
Figura 7 – Medida de stress da LAMP com relação a quantidade de pontos de controle tomados para gerar a projeção. Figura retirada de (JOIA et al., 2011)



Em consequência da grande possibilidade de aplicações, Joia et al. (2012) desenvolveu a Class-Specific Multidimensional Projection (JOIA et al., 2012) (CSMP), uma técnica interativa e que tem a capacidade de efetuar buscas multiobjetivas (multi-objective searches) e é aplicada no contexto de Content-Based Image Retrieval (CBIR). O método usa métricas específicas e projeção multidimensional na classificação e visualização das imagens. Primeiramente é feita extração de características das imagens e em seguida, com base nos objetivos da pesquisa, é criada uma métrica específica para cada um deles. Uma grande vantagem da metodologia é poder

escolher visualmente a qual classe pertence cada imagem dada como entrada na pesquisa como ilutrado na Figura 8.

Figura 8 – Manipulação do usuário para escolher as classes de cada imagem. Figura retirada de (JOIA et al., 2012)



Sejam Q_i um subconjunto das imagens de pesquisa que pertencem a mesma classe e $I_{Q_i} = \{i_1, \ldots, i_k\}$ os índices das características que melhor representam as imagens em Q_i. A métrica da classe é definida por

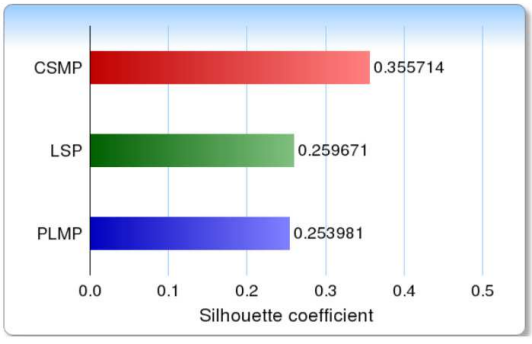
$$d_{Q_i}(\alpha, \beta) = \sum_{j \in I_{Q_i}} (\alpha_j - \beta_j)^2$$

onde α_j e β_j são as j-ésimas coordenadas das imagens α e β, respectivamente.

Essa métrica tem como base que imagens de um mesmo grupo tenham distância menor entre elas do que quando comparadas com imagens de outro grupo. Com base na dissimilaridade calculada pelas métricas de cada grupo é feita a projeção no espaço visual. A técnica apresenta bons resultados tanto no contexto de projeção multidimensional quanto no contexto de CBIR. As Figuras 9a e 9b mostram um comparativo da CSMP com outras técnicas no contexto de projeção multidimensional e CBIR, respectivamente.

Figura 9 – Resultados da técnica CSMP. Figura retirada de (JOIA et al., 2012)

(a) Comparação da CSMP com outras técnicas de projeção através da silhueta



(b) Comparação da CSMP com a técnica FIRE no contexto de CBIR

Um trabalho que tem despertado cada vez mais interesse da comunidade de visualização é descrito por Maaten e Hinton (2008) devido a sua capacidade de manter a estrutura local dos

dados e revelar importantes estruturas globais como agrupamento. A t-Distributed Stochastic Neighbor Embedding (t-SNE) é uma técnica não linear de redução de dimensionalidade composta por dois estágios principais. Inicialmente uma medida de similaridade é construída baseada em uma distribuição de probabilidades de modo que a probabilidade de que uma instância x_j seja escolhida como vizinha de uma instância x_i seja alta se essas instâncias são similares. Matematicamente, a t-SNE calcula as probabilidades p_{ij} entre as instâncias x_i e x_j por

$$p_{ij} = \frac{p_{j|i} + p_{i|j}}{2n}$$

onde n é o número de instâncias e

$$p_{j|i} = \frac{\exp(-\|x_i - x_j\|^2 / 2\sigma_i^2)}{\sum_{k \neq i} \exp(-\|x_i - x_k\|^2 / 2\sigma_i^2)}$$

com σ_i sendo a variância da gaussiana centrada em x_i.

Em seguida, outra medida de similaridade baseada em uma distribuição de probabilidades é construída, mas dessa vez utilizando as instâncias de dado mapeadas no espaço visual. Como o objetivo é encontrar um mapeamendo no espaço visual que preserve a medida p_{ij} da melhor forma possível, define-se a medida de similaridade q_{ij} entre y_i e y_j (projeções de x_i e x_j) da seguinte forma

$$q_{ij} = \frac{(1 + \|y_i - y_j\|^2)^{-1}}{\sum_{k \neq l} (1 + \|y_k - y_l\|^2)^{-1}}$$

A posição das instâncias no espaço visual é determinada minimizando a divergência de Kullback-Leibler entre as distribuições P e Q acima, isto é

$$KL(P||Q) = \sum_{i \neq j} p_{ij} \log \frac{p_{ij}}{q_{ij}}$$

Para efetuar a minimização, utiliza-se o gradiente

$$\frac{\partial C}{\partial y_i} = 4 \sum_j (p_{ij} - q_{ij})(y_i - y_j)(1 + \|y_i - y_j\|^2)^{-1}$$

2.3 Projeções utilizando dados kernelizados

A extensão natural do PCA (BISHOP, 2006) para dados kernelizados foi feita por Schölkopf, Smola e Müller (1998), chamada de Kernel PCA (SCHÖLKOPF; SMOLA; MÜLLER, 1998) (KPCA). Os autores utilizam a mesma metodologia do PCA, mas aplicada aos dados imersos no espaço de características. Dado um conjunto $X = \{x_1, x_2, \ldots, x_N\}$, a função kernel

$$\begin{aligned} k : X \times X &\longrightarrow R \\ (x_i, x_j) &\mapsto k(x_i, x_j) = \varphi(x_i)^T \varphi(x_j) \end{aligned} \tag{2.5}$$

define um mapeamento não linear $\varphi : X \to H$ do conjunto X num espaço de Hilbert H. A aplicação φ é definida implicitamente pela função kernel.

Uma vez que os dados foram imersos no espaço de características H, digamos $\varphi(X) = \{\varphi(x_1), \varphi(x_2), \ldots, \varphi(x_N)\}$, queremos encontrar uma direção $w \in H$ tal que a variância dos dados projetados nessa direção seja a máxima possível. Matematicamente,

$$\arg\max_w \{ var(w^T \varphi(X)) \} = \arg\max_w \{ w^T C w \}$$

onde $C = \frac{1}{N} \sum_{i=1}^{N} (\varphi(x_i) - \mu)(\varphi(x_i) - \mu)^T$ e $\mu = \sum_{i=1}^{N} \varphi(x_i)$.

Isso equivale a resolver o problema de autovetores e autovalores

$$Cw = \lambda w$$

e o autovetor associado ao maior autovalor é a direção que maximiza a variância. SchÖlkopf, Smola e MÜller (1998) mostram ainda que resolver o problema acima é equivalente a resolver o seguinte problema

$$Kv = \gamma v \quad \text{com } K = \begin{bmatrix} k(x_i, x_j) \end{bmatrix}$$

onde os autovetores e autovalores de C (w, γ) e K (v, λ) obedecem a seguinte relação

$$\gamma = N\lambda \quad e \quad w = \sum_{i=1}^{N} v_i \varphi(x_i)$$

Por fim, a projeção de uma instância de dado $\varphi(x_j)$ sobre a i-ésima direção principal w_i é calculada como

$$w_i^T \varphi(x_j) = v_i^T K_j \quad \text{com } K_j = [\, k(x_1, x_j) \ldots k(x_N, x_j) \,]^T$$

Ham et al. (2004) utiliza a teoria de kernel learning para fazer a interpretação de três algoritmos de redução de dimensionalidade, Isomap, Laplacian Eigenmap, Locally Linear Embedding (LLE), utilizando uma abordagem por kernel. A abordagem constrói um mapeamento implícito do conjunto de treinamento no espaço de características que preserva as propriedades importantes do dado. Esse mapeamento é descrito pela matriz do kernel que representa o produto interno do espaço de características.

Para alcançar resultados semelhantes aos da Isomap, Ham et al. (2004) utilizam o kernel que retorna a matriz

$$K_{Isomap} = -\frac{1}{2}(I - ee^T) S (I - ee^T)$$

onde S é a matriz das distâncias ao quadrado e $e = (1, \ldots, 1)^T$.

A técnica Laplacian eigenmap utiliza uma matriz de adjacências W, onde

$$W_{ij} = \begin{cases} > 0 & \text{se } i \text{ é vizinho de } j \ (i \sim j) \\ = 0 & \text{caso contrário.} \end{cases}$$

Além disso, o grafo Laplaciano L definido em termos da matriz de adjacências W é tal que

$$L_{ij} = \begin{cases} \sum_{j \sim i} W_{ij} & \text{se } i = j \\ -W_{ij} & \text{se } i \text{ é vizinho de } j \\ 0 & \text{caso contrário} \end{cases}$$

onde a matriz do kernel é dada pela matriz dos autovetores associados aos maiores autovalores da matriz pseudo-inversa de L.

Por fim, como o algoritmo Locally Linear Embedding (HAM et al., 2004) (LLE) constrói uma matriz de pesos W que leva em consideração os vizinhos de cada instância de dado, Ham et al. (2004) definem a matriz

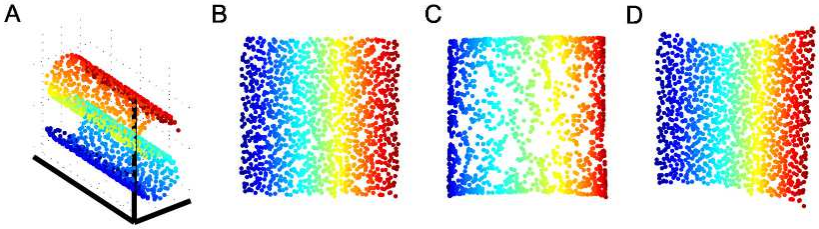
$$M = (I - W^T)(I - W)$$

cujo maior autovalor é chamado λ_{max}. Daí, a matriz do kernel utilizada é dada por

$$K = \lambda_{max} I - M$$

A Figura 10 apresenta o resultado das técnicas de redução de dimensionalidade aplicadas à variedade S.

Figura 10 – Técnicas Isomap (B), Laplacian Eigenmap (C) e LLE (D) aplicadas a variedade S (A). Figura retirada de (HAM et al., 2004)



Uma proposta para interpolação para novas instâncias baseado em combinações lineares de instâncias já mapeadas é feita por Inaba, Salles e Rauber (2011). Os autores utilizam a teoria de kernel para mapear os dados num espaço de Hilbert e a partir desses dados aplicar uma metodologia similar à feita por Sammon (1969). A distância utilizada no espaço de Hilbert é dada por

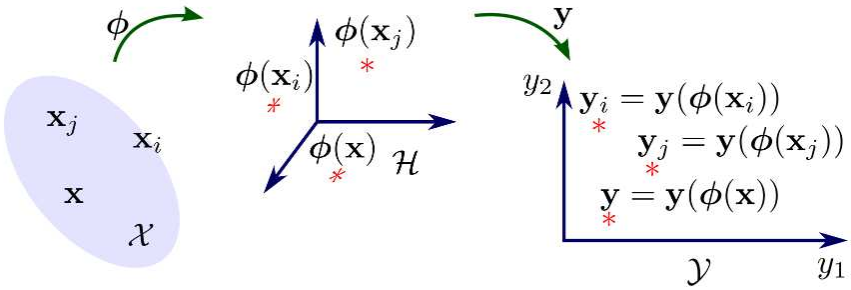
$$\begin{aligned} D_{ij}^2 &= \|\varphi(x_i) - \varphi(x_j)\|^2 \\ &= \varphi(x_i)^T \varphi(x_i) - 2\varphi(x_i)^T \varphi(x_j) + \varphi(x_j)^T \varphi(x_j) \\ &= k(x_i, x_i) - 2k(x_i, x_j) + k(x_j, x_j) \end{aligned}$$

e é escrita em função do kernel. A Figura 11 ilustra o pipeline da técnica.

Uma contribuição de Inaba, Salles e Rauber (2011) é a formulação de uma medida de stress definida em função de distância baseada em kernel no espaço de características

$$E_n = E(\{y_1, \ldots, y_n\}) = \sum_{i<j}^{n} \frac{(D_{ij} - d_{ij})^2}{D_{ij}}$$

Figura 11 – Pipeline da técnica Kernel Sammon Map. Figura retirada de (INABA; SALLES; RAUBER, 2011)



Além disso, Inaba, Salles e Rauber (2011) apresentam uma metodologia para mapear novas instâncias de dados a projeção sem a necessidade de recalcular todo mapeamento e de modo a não piorar o erro, pois para calcular o erro

$$E_{n+1} = E_n + \sum_{i=1}^{n} \frac{(D_{i,n+1} - d_{i,n+1})^2}{D_{i,n+1}}$$

só é necessário estimar a segunda parcela.

Em (ALZATE; SUYKENS, 2008) é feita uma extensão da formulação least squares support vector machine para o Kernel PCA (KPCA) utilizando uma generalização da função de perda, $L : R \rightarrow R$

$$\min_{w,e,b} J_p(w,e,b) = \frac{\gamma}{2}\sum_{i=1}^{N} L(e_i) - \frac{1}{2}w^T w$$

onde, $e_i = w^T \varphi(x_i) + b$.

Outro trabalho que apresenta uma revisão do algoritmo Isomap utilizando a teoria de kernel é devido a Choi e Choi (2004). Dada a matriz de distâncias geodésicas (aproximadas) D^2, é construída a matriz

$$K(D^2) = -\frac{1}{2}HD^2H$$

onde H é a matriz de centralização, dada por

$$H = I - \frac{1}{N}ee^T \quad e = [1 \; \cdots \; 1]^T$$

A partir daí, é calculado o maior autovalor c da matriz

$$\begin{bmatrix} 0 & 2K(D^2) \\ -I & -4K(D) \end{bmatrix}$$

e por fim, é calculada a matriz do kernel

$$K = K(D^2) + 2cK(D) + \frac{1}{2}c^2H$$

As técnicas apresentadas nesse capítulo são o estado da arte no tocante a projeções de dados multidimensionais (dados euclidianos). Entretanto, são limitadas ao lidar com dados que

não estão imersos num espaço com coordenadas, seja pela falta de interatividade com o usuário ou pela incapacidade de lidar com dados kernelizados. Uma maneira de contornar essa limitação é utilizar técnicas capazes de explorar propriamente esse novo paradigma de dados e que ainda permita a interação do usuário no processo de visualização como a Kelp (Capítulo 4).

CAPÍTULO

3

UM ESTUDO SOBRE SUBSPACE CLUSTERING E PROJEÇÕES MULTIDIMENSIONAIS

Neste capítulo, apresentaremos um estudo sobre o efeito de técnicas de subspace clustering no auxílio a técnicas de projeção multidimensional. Faremos inicialmente uma introdução às técnicas de subspace clustering e em seguida os resultados obtidos no estudo. Esse trabalho foi primeiramente descrito em (BARBOSA; SADLO; NONATO, 2015).

3.1 Low-Rank Representation

Nesta seção apresentaremos brevemente alguns detalhes da técnica de subspace clustering denominada Low-Rank Representation (LRR) (ZHANG et al., 2014). Sejam um conjunto de dados $\{x_1, x_2, \ldots, x_n\}$, $x_i \in R^d$, onde n é o número de instâncias de dado no espaço cartesiano d-dimensional e X sua representação matricial, onde cada instância x_i é disposta como coluna de X. Suponha que o dado possa ser decomposto da forma $X = D + E$, onde D é o dado obtido de subespaços independentes e E é o "erro" no dado devido a outliers, ruído ou falhas na coleta, por exemplo. O objetivo da técnica LRR é encontrar uma matriz D de posto baixo a partir da matriz X com erro E.

Podemos escrever a questão acima como o seguinte problema de minimização com regularização:

$$\min_{D,E} \text{posto}(D) + \lambda \|E\|_\ell \text{ tal que } X = D + E \tag{3.1}$$

onde $\lambda > 0$ é um parâmetro e $\|\cdot\|_\ell$ é a estratégia de regularização a ser escolhida. Entretanto, para conseguir uma flexibilidade maior, reescrevemos (3.1) como:

$$\min_{Z,E} \text{posto}(Z) + \lambda \|E\|_\ell \text{ tal que } X = AZ + E \tag{3.2}$$

com A sendo um "dicionário" que gera o espaço dos dados. De posse da solução (Z^*, E^*) do problema (3.2), temos a matriz $D = AZ^*$, que pode ser interpretada como o dado limpo, isto é, sem a interferência do erro originalmente presente em X. Note que, como $posto(AZ^*) \leq \min(posto(A), posto(Z^*)) = posto(Z^*)$, o produto AZ^* também é uma matriz de posto baixo, satisfazendo as condições necessárias para D.

Para resolver o problema (3.2), Zhang et al. iniciaram o estudo por problemas mais simples e construíram a solução geral a partir deles. Suponha então que o dado não possui influência de erro algum, ou seja, a matriz $E = 0$. Dessa forma, o problema (3.2) se torna

$$\min_Z posto(Z) \text{ tal que } X = AZ \tag{3.3}$$

cuja solução pode não ser única. Assim, como de costume para problemas de otimização envolvendo o posto de matrizes, o problema (3.3) foi reescrito como

$$\min_Z \|Z\|_* \text{ tal que } X = AZ \tag{3.4}$$

onde $\|\cdot\|_*$ é a norma nuclear (soma dos valores singulares da matriz). O teorema abaixo apresenta uma forma fechada para a solução do problema (3.4):

Teorema 1. Se $A \neq 0$ e $X = AZ$ possui solução para X e A dados, então $Z^* = A^\dagger X$ é a única solução do problema (3.4), onde $A^\dagger$ é a pseudo inversa da matriz A.

Vale também o resultado abaixo que faz a ponte entre as soluções de (3.3) e (3.4)

Corolário 1. Suponha $A \neq 0$ e que $X = AZ$ possui solução para X e A dados. Seja Z^* a solução do problema (3.4), então $posto(Z^*) = posto(X)$ e Z^* é também solução do problema (3.3).

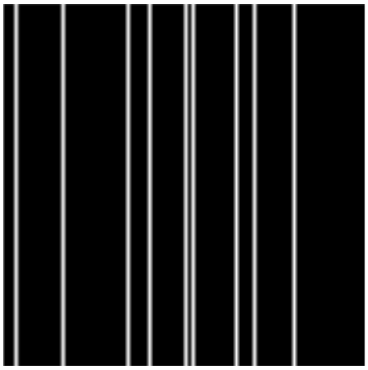
A prova desses resultados pode ser encontrada no Apêndice.

Figura 12 – Exemplos de erros comuns presentes no dado: (a) o dado possui perturbações; (b) o dado contém entradas com erros, falta de informação, por exemplo; (c) instâncias de dado (colunas) estão distantes. (ZHANG et al., 2014)

(a) Dado com ruído

(b) Dado com erros aleatórios

(c) Erros de amostragem (e outliers)



Sabendo que a norma nuclear é adequada para substituir a função posto, Zhang et al. seguiram o estudo do problema analisando o efeito do termo de regularização

$$\min_{Z,E} \|Z\|_* + \lambda \|E\|_{2,1} \text{ tal que } X = AZ + E \tag{3.5}$$

onde $\|E\|_{2,1} = \sum_i \|E(:,i)\|_2$ é uma norma capaz de capturar outliers e erros de amostragem (sample-specific corruptions), Figura 12c. Para ruído Gaussiano pequeno, Figura 12a, a norma $\|E\|_F^2$ (Frobenius ao quadrado) deve ser a estratégia de regularização escolhida e para erros aleatórios, Figura 12b, a norma $\|E\|_1$ é a escolha apropriada.

O problema (3.5) pode ser resolvido por vários métodos e, por questão de eficiência, Zhang et al. optaram por resolvê-lo utilizando Augmented Lagrange Multiplier (LIN et al., 2009) (ALM). Primeiramente, o problema (3.5) é convertido no seguinte problema equivalente

$$\min_{Z,E,J} \|J\|_* + \lambda \|E\|_{2,1} \text{ tal que } X = AZ + E, Z = J \tag{3.6}$$

o qual é resolvido pelo ALM através da seguinte função de Lagrange

$$\begin{aligned} L = {} & \|J\|_* + \lambda \|E\|_{2,1} + \operatorname{tr}\left(Y_1^T (X - AZ - E)\right) + \operatorname{tr}\left(Y_2^T (Z - J)\right) \\ & + \frac{\mu}{2} \|X - AZ - E\|_F^2 + \|Z - J\|_F^2 \end{aligned} \tag{3.7}$$

O problema acima não possui restrição, logo pode ser minimizado com respeito a uma das variáveis por vez — fixando as demais — e então atualizando os multiplicadores Y_1 e Y_2. O Algoritmo 1 descreve o procedimento utilizado para resolver o problema.

O passo 1 do Algoritmo 1 pode ser resolvido através do operador Singular Value Thresholding (SVT) (CAI; CANDÈS; SHEN, 2010), enquanto que o passo 3 é resolvido pelo seguinte Lema:

Lema 1. (YANG et al., 2009) Dada uma matriz Q, se W^* é a solução ótima de

$$\min_W \alpha \|W\|_{2,1} + \frac{1}{2} \|W - Q\|_F^2$$

então a i-ésima coluna de W^* é

$$W^*(:,i) = \begin{cases} \frac{\|Q(:,i)\|_2 - \alpha}{\|Q(:,i)\|_2} Q(:,i) & \text{se } \|Q(:,i)\|_2 > \alpha \\ 0 & \text{caso contrário} \end{cases}$$

Uma implementação do Algoritmo 1 pode ser encontrada online (LIU, 2016).

De posse da solução (Z^*, E^*) do problema (3.5), Zhang et al. descrevem, através do Algoritmo 2, um método para segmentar o dado $D = AZ$, inspirado pelo método de Shape Interaction Matrix (SIM) (COSTEIRA; KANADE, 1998). O processo consiste em determinar a qual subespaço cada instância do dado pertence baseado no espaço linha da matriz do dado limpo, sem a influência de erro.

Algoritmo 1 – Solução do problema (3.6) (ZHANG et al., 2014)

entrada : Matriz do dado X, parâmetro λ
inicialize: $Z = J = 0, E = 0, Y_1 = 0, Y_2 = 0, \mu = 10^{-6}, \mu_{max} = 10^{6}, \rho = 1.1$ e $\varepsilon = 10^{-8}$
enquanto não convergir faça

1 fixe as demais variáveis e atualize J pela equação

$$J = \arg\min \frac{1}{\mu}\|J\|_* + \frac{1}{2}\left\| J - \left(Z + \frac{Y_2}{\mu} \right) \right\|_F^2$$

2 fixe as demais variáveis e atualize Z pela equação

$$Z = (I + A^T A)^{-1} \left(A^T (X - E) + J + \frac{A^T Y_1 - Y_2}{\mu} \right)$$

3 fixe as demais variáveis e atualize E pela equação

$$E = \arg\min \frac{\lambda}{\mu}\|E\|_{2,1} + \frac{1}{2}\left\| E - \left(X - AZ + \frac{Y_1}{\mu} \right) \right\|_F^2$$

4 atualize os multiplicadores

$$Y_1 = Y_1 + \mu(X - AZ - E)$$

$$Y_2 = Y_2 + \mu(Z - J)$$

5 atualize o parâmetro μ pela equação

$$\mu = \min(\rho\mu, \mu_{max})$$

6 verifique a convergência

$$\|X - AZ - E\|_\infty < \varepsilon \text{ e } \|Z - J\|_\infty < \varepsilon$$

Algoritmo 2 – Segmentação do dado em subespaços

entrada: Matriz do dado X, número de subespaços k, solução Z^* do problema (3.5)

1 calcule a decomposição SVD $Z^* = U\Sigma V^T$

2 construa a matriz de afinidade $W = [w_{ij}]$, com $w_{ij} = \tilde{u}_{ij}^2$, onde $\tilde{u}_{ij}$ são as entradas da matriz $\tilde{U} = (U\Sigma^{\frac{1}{2}})(U\Sigma^{\frac{1}{2}})^T$

3 aplique Normalized Cuts (SHI; MALIK, 2000) e segmente o dado em k grupos

Além disso, um estimador para o número de subespaços é apresentado no Algoritmo 3 e tem como base o fato da matriz de afinidade ter a estrutura de bloco diagonal. A estimativa é feita contando o número de valores singulares não nulos da matriz Laplaciana L da matriz de afinidade. Na prática, a matriz de afinidade é aproximadamente bloco diagonal. Dessa forma, o

estimador utiliza um threshold para retornar o número de subespaços:

$$k = n - \text{int}\left(\sum_{i=1}^{n} f_\tau(\sigma_i)\right) \tag{3.8}$$

onde, n é o número de instâncias do dado, σ_i são os valores singulares da matriz L, $\text{int}(\cdot)$ retorna o inteiro mais próximo, $f_\tau(\cdot)$ é dado por

$$f_\tau(\sigma) = \begin{cases} 1 & \text{se } \sigma \geq \tau \\ \log_2\left(1 + \frac{\sigma^2}{\tau^2}\right) & \text{caso contrário} \end{cases}$$

e $0 < \tau < 1$ é um parâmetro.

Algoritmo 3 – Estimativa para o número de subespaços

entrada: Matriz do dado X, matriz de afinidade W do algoritmo 2

1 calcule a matriz Laplaciana $L = I - D^{\frac{1}{2}}WD^{\frac{1}{2}}$, onde $D = \text{diag}\left(\sum_j W(1,j), \sum_j W(2,j), \ldots, \sum_j W(n,j)\right)$

2 estime o número de subespaços k pela equação (3.8)

3.2 Linear Discriminant Analysis

Apresentamos agora uma explicação do funcionamento do método Linear Discriminant Analysis (LDA), que consiste em encontrar uma combinação linear dos atributos capaz de separar o dado em classes. A LDA é comumente utilizada em estatística, reconhecimento de padrões e aprendizado de máquina funcionando como um classificador linear ou para redução de dimensionalidade.

Estamos interessados em encontrar um subespaço onde as classes do dado fiquem separadas da melhor forma possível. Sejam $\{(x_1, l_1), (x_2, l_2), \ldots, (x_n, l_n)\}$ o conjunto de dados e suas respectivas classes, onde $x_i \in R^D$ e $l_i \in \{1, 2, \ldots, k\}$. Cada instância de dado x_i foi classificada em uma das k classes e essa classificação recebeu o rótulo l_i. Denotando o centroide (média) de cada classe por $\bar{x}_i$, $i = 1, 2, \ldots, k$, e por $\bar{x}$ o centroide de todo o dado, temos $\bar{x}_i = \frac{1}{n_i}\sum_{x_j \in C_i} x_j$, onde n_i é o número de instâncias de dado pertencentes a classe C_i, e $\bar{x} = \frac{1}{n}\sum_{i=1}^{n} x_i$.

Denotemos

$$S_w = \frac{1}{n}\left[\sum_{x_j \in C_1}(x_j - \bar{x}_1)(x_j - \bar{x}_1)^T + \cdots + \sum_{x_j \in C_k}(x_j - \bar{x}_k)(x_j - \bar{x}_k)^T\right]$$

$$S_b = \frac{1}{n}\sum_{j=1}^{k} n_j(\bar{x}_j - \bar{x})(\bar{x}_j - \bar{x})^T$$

Assim $S_w, S_b \in M(D \times D)$ (matrizes $D \times D$). Então,

$$\text{tr}(S_w) = \frac{1}{n}\sum_{j=1}^{k}\sum_{i=1}^{n_j} \|x_i - \bar{x}_j\|^2 \text{ (variância das instâncias em cada classe)}$$

$$\text{tr}(S_b) = \frac{1}{n}\sum_{j=1}^{k} n_j\|\bar{x}_j - \bar{x}\|^2 \text{ (variância dos centroides)}$$

Note que, se a projeção das instâncias de dado num subespaço resulta em pouca variância dos elementos em cada classe e muita variância dos centroides, então existe grande probabilidade de que as classes estejam bem separadas. Se P é a matriz cujas colunas formam uma base do subespaço onde queremos projetar o dado, então

$$\begin{aligned}\tilde{S}_w &= \frac{1}{n}\sum_{i=1}^{k}\sum_{x_j \in C_i} P^T(x_j - \bar{x}_i)\left[P^T(x_j - \bar{x}_i)\right]^T \\ &= \sum_{i=1}^{k}\sum_{j \in C_i}\left[P^T(x_j - \bar{x}_i)(x_j - \bar{x}_i)^T P\right] \\ &= P^T S_w P\end{aligned}$$

e, analogamente, $\tilde{S}_b = P^T S_b P$. Portanto, queremos encontrar P tal que $tr(\tilde{S}_w)$ seja mínimo e $tr(\tilde{S}_b)$ seja máximo. P deve então satisfazer:

$$P_{opt} = \arg\max_P \frac{tr(\tilde{S}_b)}{tr(\tilde{S}_w)} = \arg\max_P \frac{tr(P^T S_b P)}{tr(P^T S_w P)} \tag{3.9}$$

Mas como o problema (3.9) é de difícil resolução, resolveremos um problema semelhante:

$$\arg\max_P tr\left[(P^T S_b P)(P^T S_w P)^{-1}\right] \tag{3.10}$$

cuja solução é dada pelo seguinte teorema:

Teorema 2. (BISHOP, 2006) Sejam A e B matrizes simétricas $n \times n$ onde B é definida positiva. Sejam ainda $\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_n$ e $u_1, u_2, \ldots, u_n$ os autovalores e autovetores do problema de autovalor generalizado

$$Ax = \lambda Bx$$

Então

$$\max_{X \in M\,(n \times p)} tr\left[(X^T AX)(X^T BX)^{-1}\right] = \sum_{i=1}^{p} \lambda_i$$

Dessa forma, a solução do problema (3.10) é dada pelos autovetores associados aos maiores autovalores do problema de autovalores generalizado $S_b P = S_w P\Lambda$, onde as colunas de P são os autovetores associados aos autovalores formando a matriz diagonal Λ. Para visualização, utilizaremos $p = 2$.

3.3 Estudando o efeito de subspace clustering em projeções multidimensionais

Apresentaremos agora um estudo do efeito da aplicação de técnicas de subspace clustering, como a LRR, no auxílio a projeções multidimensionais. Utilizamos a divisão em subespaços

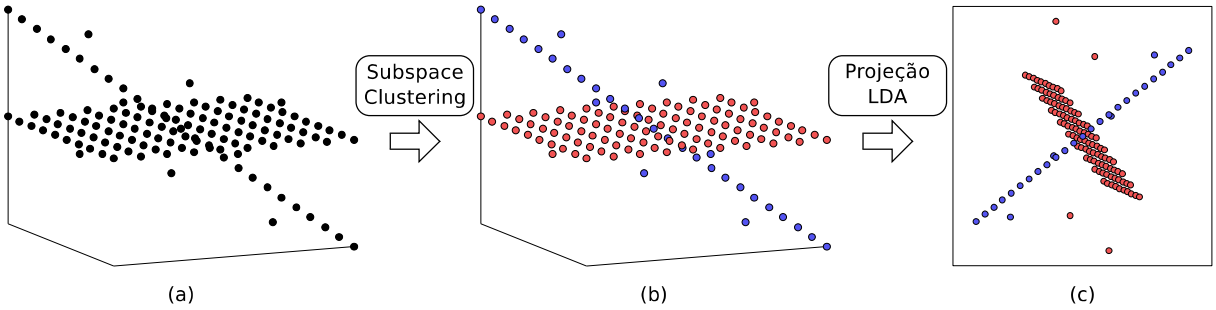
gerado pela LRR como entrada para as classes na LDA e comparamos a qualidade dessas projeções com técnicas bastante conhecidas na literatura como a LAMP (JOIA et al., 2011) e t-SNE (MAATEN; HINTON, 2008). Além disso, propomos uma mudança simples na LAMP para que possa fazer uso da informação de classes do dado.

As principais contribuições desse trabalho são listadas abaixo:

- Utilizamos técnicas de subspace clustering combinadas com técnicas de visualização para redução de dimensionalidade;
- Estudamos a eficácia da classe dos dados nas visualizações comparando com técnicas que necessitem da informação de classe;
- Uma modificação simples na LAMP para que possa utilizar a informação de classe do dado.

Considere um conjunto de dados de alta dimensão, queremos visualizar padrões e estruturas intrínsecas pertencentes a esse dado. Supondo que as instâncias do dado foram obtidas de subespaços independentes, utilizamos a LRR para estimar a quantidade de subespaços e a qual subespaço cada instância pertence. Tomamos cada subespaço definido pela LRR como uma classe e as instâncias pertencentes a cada subespaço como pertencentes àquela classe. Por fim, utilizamos essa informação de classes na LDA para efetuar a projeção no espaço visual ($p = 2$). A Figura 13 ilustra o pipeline do nosso método.

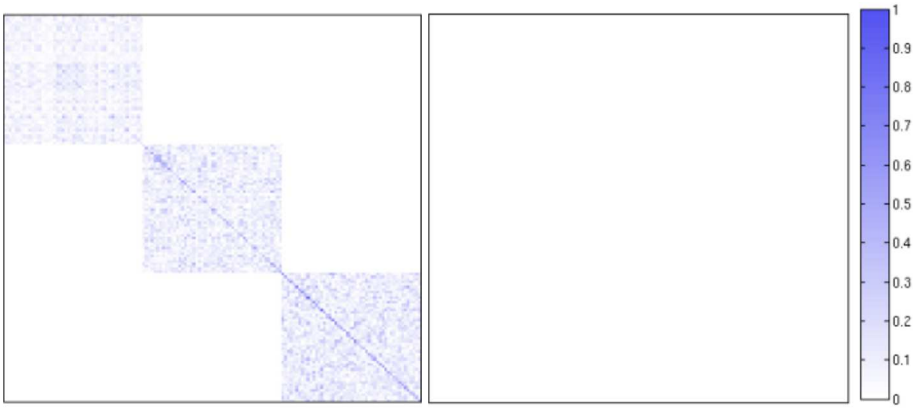
Figura 13 – Pipeline do método: (a) dado sem informação de classes; (b) dado com a informação de classes gerada pelo subspace clustering; (c) visualização do dado utilizando LDA.



3.3.1 Experimentos

Para validar o método, geramos um dado artificial composto por 50 instâncias pertencentes a R^3, 50 pertencentes a R^7, 50 pertencentes a R^{10} e mergulhamos todas as 150 instâncias desse dado em R^{30}. O resultado da LRR com parâmetro λ igual a 0.5 é mostrado na Figura 14. Observe que a representação da matriz E está em branco denotando que os valores da mesma são todos zero; ou seja, a LRR, como esperávamos, não detectou nenhuma instância contendo erro (ruído nem outlier).

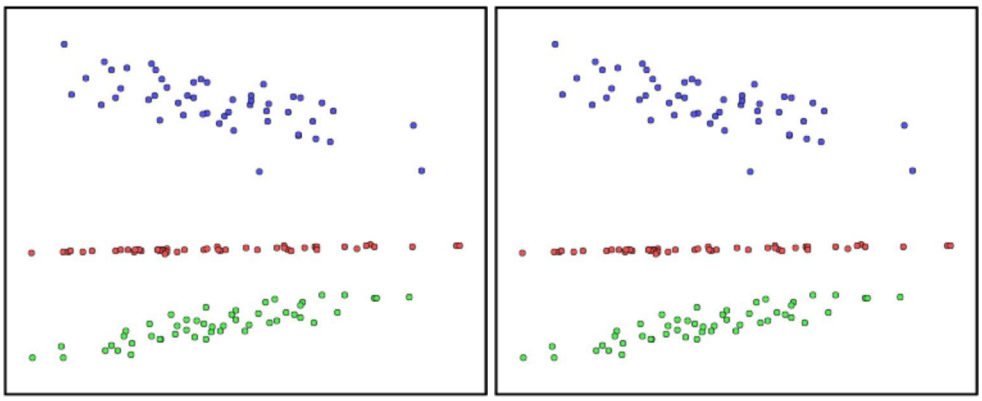
Figura 14 – Resultado da LRR no dado artificial.



(a) Matriz Z (b) Matriz E

A Figura 15 mostra o resultado da projeção do dado artificial usando a LDA com classes geradas pela LRR. Como a LRR classificou o dado perfeitamente, a projeção colorida com as classes da LRR e com as classes reais são idênticas.

Figura 15 – Projeção com LDA do dado artificial.

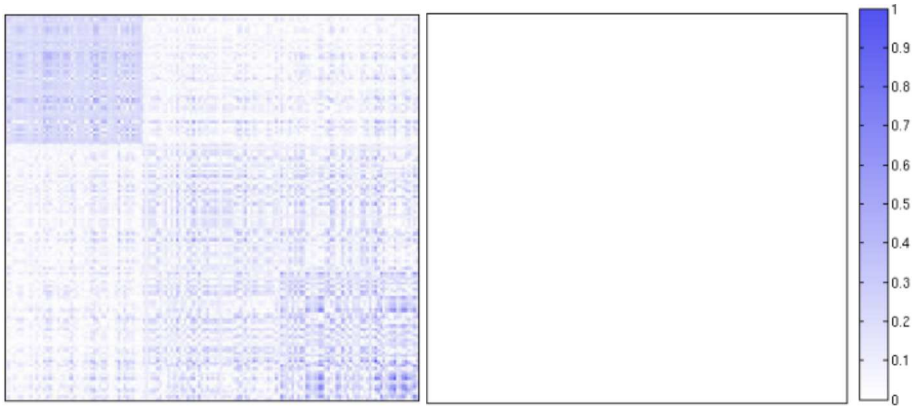


(a) Coloração dada pela LRR (b) Cor dada pelas classes reais do dado

Como segundo teste, utilizamos o conjunto de dados da flor iris (FRANK; ASUNCION, 2010). O dado é composto por 150 instâncias de dimensão 4 divididas em 3 classes, as espécies da flor iris. O resultado da LRR com parâmetro λ igual a 0.5 é exibido na Figura 16. Observe que uma das classes pode ser identificada facilmente, enquanto que as outras duas são mais difíceis de distinguir. A matriz E é nula devido à natureza do dado e ao fato de termos utilizado a norma $\|\cdot\|_{2,1}$ em nosso teste, a qual é sensível a outliers.

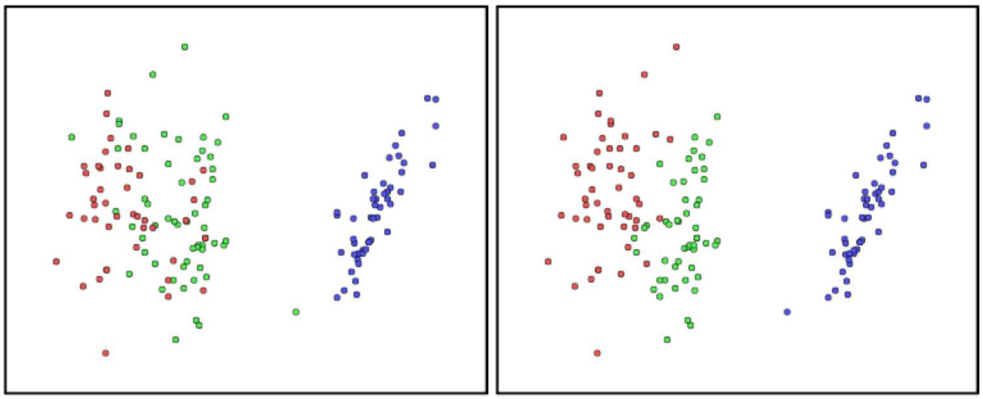
Da mesma forma que no experimento anterior, projetamos o conjunto de dados utilizando a LDA e colorimos com a informação de classes gerada pela LRR e com as classes reais do dado (Figura 17). Aqui podemos ver onde a LRR conseguiu classificar melhor o dado em cada subespaço.

Figura 16 – Resultado da LRR aplicado ao Iris.



(a) Matriz Z (b) Matriz E

Figura 17 – Projeção com LDA do Iris.



(a) Coloração dada pela LRR (b) Cor dada pelas classes reais do dado

3.3.2 Resultados e discussão

Avaliamos nosso método comparando as projeções obtidas pela LDA com classes geradas pela LRR com outras três técnicas: LAMP, t-SNE e LAMP modificada. Primeiramente explicaremos a modificação feita na LAMP para que possa utilizar informação das classes do dado no processo de projeção. Originalmente, utilizamos pesos dados por $\alpha_i = \frac{1}{\|x - x_i\|^2}$, onde x é a instância a ser projetada e x_i é um ponto de controle. Nossa modificação consiste em calcular os pesos da seguinte forma:

$$\alpha_i = \begin{cases} \frac{1}{\|x - x_i\|^2} & \text{se } x \text{ e } x_i \text{ têm a mesma classe} \\ 0 & \text{caso contrário.} \end{cases}$$

Avaliamos também a qualidade das projeções geradas por nosso método segundo quatro métricas: stress, preservação de vizinhança e duas métricas de silhueta. A função de stress utilizada é dada por

$$\text{stress} = \frac{1}{\sum_{ij} d_{ij}} \sum_{ij} \frac{(d_{ij} - \bar{d}_{ij})^2}{d_{ij}^2}$$

onde d_{ij} é a distância medida no espaço original e $\bar{d}_{ij}$ é a distância no espaço visual. A métrica de preservação de vizinhança conta, para cada instância de dado, quantos dos seus k vizinhos mais próximos no espaço original estão entre os k vizinhos mais próximos no espaço visual. A silhueta mede coesão e separação das classes e é dada por

$$\text{silhueta} = \frac{1}{n}\sum_{i} \frac{b_i - a_i}{\max a_i, b_i}$$

onde a_i (coesão) é a média das distâncias entre a projeção y_i (projeção de x_i) e todas as outras instâncias projetadas que estão na mesma classe que y_i e b_i (separação) é a menor distância entre y_i e todas as outras projeções pertencentes as outras classes. Como podemos ver, para calcular a silhueta, precisamos da informação das classes do dado. Dessa forma, utilizamos ambas informações de classe para avaliar as projeções: classe real do dado (silh1) e classes geradas pela LRR (silh2).

Tabela 1 – Conjuntos de dado utilizados. Da esquerda para direita, as colunas correspondem a: nome do conjunto de dados, número de instâncias, dimensão (número de atributos) e fonte.

Nome	Tamanho	Dim	Fonte
Iris	150	4	(FRANK; ASUNCION, 2010)
Sintético	150	4	(GUYON, 2003)
Artificial	150	30	*
Wine	178	13	(FRANK; ASUNCION, 2010)
Mammals	1000	72	(FRANK; ASUNCION, 2010)

A Tabela 1 contém os conjuntos de dados utilizados e a Tabela 2 resume os resultados obtidos. Comparada com a LAMP, a versão modificada da LAMP obtém resultado melhor para a silh2 (classes dadas pela LRR) e com pequena diferença em termos de stress. Enquanto o stress da LDA é maior que o da LAMP e o da LAMP modificada (o que é esperado, pois o objetivo da LDA é encontrar o subespaço com melhor separabilidade entre as classes), ela acaba obtendo resultados melhores quanto a silh2. Os resultados da LDA indicam que a combinação com a LRR pode ser uma escolha válida para redução de dimensionalidade ou problemas de classificação sem supervisão onde a classe real das instâncias de dado são desconhecidas.

3.3.3 Limitações

Técnicas de subspace clustering assumem que o dado está distribuído em subespaços independentes. Nos casos onde não for possível recuperar uma estrutura de subespaços nos dados, mesmo variando todos os parâmetros extensivamente, é razoável assumir que tal estrutura é inexistente e, portanto, não faz sentido aplicar o método.

Tabela 2 – Resultados obtidos. Da esquerda para direita as colunas correspondem a conjunto de dado, técnica e métricas: stress, preservação de vizinhança e silhuetas. Valores em negrito são os melhores para cada conjunto de dado e métrica.

Dado	Técnica	stress	PV (%)	silh1	silh2
Iris	LAMP	0.0418	81.8	0.6371	0.3437
	LAMP (M)	0.0791	77.8	0.6032	0.4221
	LDA	0.3095	63.6	0.6889	0.6758
	t-SNE	1.71e+6	86.9	0.7633	0.3392
Synthetic	LAMP	0.0597	80.9	0.8584	0.8584
	LAMP (M)	0.0521	82.0	0.9045	0.9045
	LDA	0.0862	85.7	0.9299	0.9299
	t-SNE	6.2266	89.4	0.9956	0.9956
Artificial	LAMP	0.0539	85.4	0.6770	0.6770
	LAMP (M)	0.0749	86.0	0.7787	0.7787
	LDA	0.3749	81.2	0.9492	0.9492
	t-SNE	0.2962	90.9	0.8961	0.8961
Wine	LAMP	0.0383	90.7	0.2174	0.3629
	LAMP (M)	0.1371	86.9	0.2269	0.4491
	LDA	0.9802	53.3	0.2694	0.5314
	t-SNE	0.9312	94.3	0.3139	0.4262
Mammals	LAMP	0.0112	87.9	0.9825	0.9825
	LAMP (M)	0.0172	85.5	0.9924	0.9924
	LDA	1.0000	81.4	0.9311	0.9311
	t-SNE	0.3829	87.7	0.9653	0.9653

CAPÍTULO

4

KERNEL BASED LINEAR PROJECTION

Neste Capítulo, apresentamos uma nova técnica de projeção multidimensional baseada em kernel capaz de lidar com dados imersos no espaço de características implicitamente induzido por um kernel. Este trabalho foi descrito primeiramente em (BARBOSA et al., 2016). As contribuições do mesmo são listadas a baixo:

- Uma nova técnica de projeção multidimensional chamada Kelp capaz de lidar com dados kernelizados;
- O uso da Kelp como ferramenta de visualização para auxiliar em aplicações baseadas em kernel como classificação e segmentação de imagem;
- Uma versão kernelizada das coordenadas diferenciais (LIPMAN et al., 2004) combinada com a Kelp a fim de entender como o kernel afeta as estruturas de vizinhança do dado durante o mapeamento para o espaço de características.

4.1 Fundamentação matemática

Seja $X = \{x_1, x_2, \ldots, x_m\}$ o conjunto de dado e $k : X \times X \rightarrow R$ uma função real que, para cada par $x_i, x_j \in X$, associa o valor de similaridade $k(x_i, x_j)$ entre as instâncias x_i e x_j. A função k é uma função kernel (ou apenas kernel) se a matriz $K = [k_{ij}]$ (matriz do kernel), onde $k_{ij} = k(x_i, x_j)$, é simétrica e positiva definida.

Associada a cada kernel existe uma aplicação $\varphi : X \rightarrow H$ tal que

$$k(x_i, x_j) = \varphi(x_i)^T \varphi(x_j)$$

onde H é um espaço de características com alta dimensão (possivelmente de dimensão infinita) onde os dados são mapeados pela aplicação φ. Dessa forma, o kernel é um produto interno

no espaço de características H e a matriz K é uma matriz de Gram (SCHÖLKOPF; SMOLA, 2001).

Observe o seguinte exemplo: se $X \subset R^2$, temos que $k(x_i, x_j) = (x_i^T x_j)^2$ é um kernel e o mapeamento φ associado a ele é dado por $\varphi(x_i) = \varphi(x, y) = (x^2, y^2, xy, yx)$, que leva cada instância de X para o espaço de características, que nesse caso é o R^4. De fato,

$$\begin{aligned}
\varphi(x_i)^T \varphi(x_j) &= \varphi(x, y)^T \varphi(x', y') \\
&= (x^2, y^2, xy, yx)^T (x'^2, y'^2, x'y', y'x') \\
&= x^2x'^2 + y^2y'^2 + xyx'y' + yxy'x' \\
&= (xx')^2 + 2(xx')(yy') + (yy')^2 \\
&= (xx' + yy')^2 \\
&= ((x, y)^T (x', y'))^2 \\
&= (x_i^T x_j)^2 \\
&= k(x_i, x_j)
\end{aligned}$$

Os kernels mais comuns na literatura são:

Gaussiano: $k(x_i, x_j) = e^{-\frac{\|x_i - x_j\|^2}{2\sigma^2}}$

Polinomial: $k(x_i, x_j) = (x_i^T x_j)^d$

Sigmoide: $k(x_i, x_j) = \tanh(\gamma(x_i^T x_j) + \theta)$

onde σ, γ, d e θ são parâmetros.

Supondo que a imagem de X por φ esteja centrada no espaço de características (a Seção 4.3 tem os detalhes sobre como centralizar o dado quando necessário), ou seja, $\frac{1}{m}\sum_{i=1}^{m} \varphi(x_i) = 0$, sua matriz de covariância é dada por

$$C = \frac{1}{m}\sum_{i=1}^{m} \varphi(x_i)\varphi(x_i)^T \tag{4.1}$$

onde $\varphi(x_i)^T$ é o transposto de $\varphi(x_i)$. Como φ é uma aplicação definida implicitamente pelo kernel, não podemos, em geral, calcular a matriz C diretamente. Para superar essa dificuldade, utilizamos o kernel trick, que consiste em utilizar a informação do produto interno definido pelo kernel no espaço de características, evitando o uso explícito das coordenadas do dado mapeado nesse espaço.

Denotando por u_i um autovetor de C e λ_i seu autovalor correspondente, da equação (4.1) e de $Cu_i = \lambda_i u_i$, segue-se que

$$\begin{aligned}
&\frac{1}{m}\sum_{j=1}^{m}\varphi(x_j)\varphi(x_j)^T u_i = \lambda_i u_i \\
\Rightarrow\ & u_i = \frac{1}{m\lambda_i}\sum_{j=1}^{m}\varphi(x_j)\ \varphi(x_j)^T u_i \\
\Rightarrow\ & u_i = \sum_{j=1}^{m}\frac{1}{m\lambda_i}\ \varphi(x_j)^T u_i\ \ \varphi(x_j) \\
\Rightarrow\ & u_i = \sum_{j=1}^{m} a_{ij}\varphi(x_j) \quad \text{com} \quad a_{ij} = \frac{1}{m\lambda_i}\varphi(x_j)^T u_i
\end{aligned} \tag{4.2}$$

Chamando de $a_i = (a_{i1} \ldots a_{im})^T$ o vetor cujas coordenadas são dadas pelos coeficientes a_{ij}, mostraremos que a_i é um autovetor da matriz K do kernel. De fato, substituindo u_i na primeira igualdade de (4.2) e multiplicando por $\varphi(x_l)^T$ temos:

$$\begin{aligned}
&\frac{1}{m}\varphi(x_l)^T\sum_{p=1}^{m}\varphi(x_p)\varphi(x_p)^T\sum_{j=1}^{m}a_{ij}\varphi(x_j) = \lambda_i\varphi(x_l)^T\sum_{j=1}^{m}a_{ij}\varphi(x_j) \\
\Rightarrow\ & \frac{1}{m}\sum_{p=1}^{m}\varphi(x_l)^T\varphi(x_p)\sum_{j=1}^{m}a_{ij}\varphi(x_p)^T\varphi(x_j) = \lambda_i\sum_{j=1}^{m}a_{ij}\varphi(x_l)^T\varphi(x_j) \\
\Rightarrow\ & \frac{1}{m}\sum_{p=1}^{m}k(x_l, x_p)\sum_{j=1}^{m}a_{ij}k(x_p, x_j) = \lambda_i\sum_{j=1}^{m}a_{ij}k(x_l, x_j)\ \forall l = 1 \ldots m \\
\Rightarrow\ & K^2 a_i = m\lambda_i K a_i \\
\Rightarrow\ & K a_i = \gamma_i a_i
\end{aligned} \tag{4.3}$$

mostrando que a_i é autovetor de K com autovalor associado γ_i. Fica então demonstrada uma importante relação entre os autovetores de u_i de C e os autovetores a_i de K

$$u_i = \sum_{j=1}^{m} a_{ij}\varphi(x_j) \tag{4.4}$$

Além disso, a equação (4.3) também nos dá a relação $\gamma_i = m\lambda_i$ entre os autovalores γ_i de K e os autovalores λ_i de C.

4.2 A Kelp

Nossa técnica utiliza um subconjunto dos dados (pontos de controle) como guia para o posicionamento das demais instâncias. Seja $X_s = \{x_{s_1}, x_{s_2}, \ldots, x_{s_n}\} \subset X$ o conjunto desses pontos de controle (note que m denota o número de instâncias de X e n o número de pontos de controle) e $Y_s = \{y_{s_1}, y_{s_2}, \ldots, y_{s_n}\}$ a imagem de X_s no espaço visual (Y_s é resultado da aplicação da técnica Force Scheme a X_s). Vamos denotar por K_s a matriz do kernel construída apenas com os pontos de controle, isto é, $K_s = \left[k(x_{s_i}, x_{s_j})\right]$.

Supondo que conhecemos a função φ associada ao kernel k, queremos encontrar uma transformação linear $M : H \rightarrow R^2$ (R^2 será nosso espaço visual) tal que

$$M\varphi(x_{s_i}) = y_{s_i} \quad i = 1, \ldots, n \tag{4.5}$$

Ou seja, queremos uma transformação linear que leve a imagem de cada ponto de controle $\varphi(x_{s_i})$ do espaço de características para sua posição y_{s_i} no espaço visual. Esperamos com isso que a estrutura de vizinhança de $\varphi(x_{s_1})$ seja preservada por M, por conta da linearidade.

Podemos escrever a equação (4.5) em forma matricial como

$$M\Phi = Y \tag{4.6}$$

onde

$$\Phi = \begin{bmatrix} \vdots & & \vdots \\ \varphi(x_{s_1}) & \cdots & \varphi(x_{s_n}) \\ \vdots & & \vdots \end{bmatrix} \quad Y = \begin{bmatrix} \vdots & & \vdots \\ y_{s_1} & \cdots & y_{s_n} \\ \vdots & & \vdots \end{bmatrix}$$

têm dimensões $h \times n$ e $2 \times n$ respectivamente, e h é a dimensão de $\text{span}\{\varphi(x_{s_1}), \ldots, \varphi(x_{s_n})\}$.

Multiplicando a equação (4.6) por Φ^T, temos

$$M\Phi\Phi^T = Y\Phi^T \Rightarrow nMC_s = Y\Phi^T \tag{4.7}$$

onde C_s é a matriz de covariância definida na equação (4.1), mas calculada apenas nos pontos de controle. Como C_s é simétrica, pode ser decomposta da forma $C_s = UDU^T$, onde as colunas de U são ortonormais compostas pelos autovetores u_i de C_s e D é matriz diagonal contendo os autovalores λ_i associados aos autovalores u_i. Então, a pseudo-inversa de C_s é dada por $C_s^\dagger = U\tilde{D}^{-1}U^T$, sendo $\tilde{D}^{-1}$ a inversa de D considerando apenas os elementos não nulos da diagonal. Multiplicando ambos os lados de (4.7) por $C_s^\dagger$

$$M = \frac{1}{n}Y\Phi^T C_s^\dagger = \frac{1}{n}Y\Phi^T U\tilde{D}^{-1}U^T$$

Assim, a projeção da imagem de uma instância qualquer $\varphi(x)$ é dada por

$$M\varphi(x) = \frac{1}{n}Y\Phi^T U\tilde{D}^{-1}U^T\varphi(x) \tag{4.8}$$

Seja A a matriz cujas colunas são dadas pelos autovetores a_i de K_s (equação (4.4)) e, abusando da notação, U será a partir de agora a matriz contendo apenas os autovetores de C_s cujo autovalor associado seja não nulo. Da equação (4.4), tem-se

$$U = \Phi A \Rightarrow \Phi^T U = \Phi^T\Phi A \Rightarrow \Phi^T U = K_s A \tag{4.9}$$

e

$$U^T\varphi(x) = (\Phi A)^T\varphi(x) = A^T\Phi^T\varphi(x) = A^T k_x \tag{4.10}$$

onde $k_x = (k(x, x_{s_1}), k(x, x_{s_2}), \ldots, k(x, x_{s_n}))^T$.

Utilizando a relação entre os autovalores γ_i de K_s e os autovalores λ_i de C_s, dada por $\gamma_i = n\lambda_i$ (equação (4.2)), escrevemos

$$\tilde{D}^{-1} = \frac{1}{\lambda_i} = n\frac{1}{\gamma_i} = n\Gamma^{-1} \tag{4.11}$$

onde Γ é a matriz diagonal com entradas γ_i. A equação (4.11) juntamente com as equações (4.9) e (4.10) nos dão que

$$M\varphi(x) = YK_sA\Gamma^{-1}A^Tk_x \tag{4.12}$$

Dessa forma, escrevemos a projeção $M\varphi(x)$ da imagem de uma instância x qualquer no espaço de características em função apenas de quantidades conhecidas: Y é a matriz com a posição dos pontos de controle no espaço visual, K_s é a matriz do kernel calculada nos pontos de controle, as colunas de A são os autovetores de K_s, a diagonal de Γ são os autovalores de K_s e o vetor k_x é a medida de similaridade dada pelo kernel entre a instância x a ser projetada e os pontos de controle x_{s_i}. Mais ainda, Y, K_s, A e Γ precisam ser calculados apenas uma vez, pois não dependem da instância a ser projetada, apenas do kernel e dos pontos de controle.

4.3 Centralização no espaço de características

Na seção 4.1 supomos que o dado estaria centrado no espaço de características. Nessa seção explicaremos como centrar o dado quando necessário. Como não temos acesso as coordenadas do dado no espaço de características, faremos a centralização utilizando a matriz K_s e o vetor k_x.

O procedimento para centralizar uma matriz de kernel K já é conhecido da literatura de aprendizado de máquina (SCHÖLKOPF; SMOLA; MÜLLER, 1998). Suponha φ não necessariamente centrada no espaço de características, logo

$$\tilde{\varphi}(x_i) = \varphi(x_i) - \frac{1}{m}\sum_{j=1}^{m}\varphi(x_j)$$

estará centrada no espaço de características. Dessa forma, toda teoria desenvolvida na Seção 4.1 vale para $\tilde{\varphi}$. Seja $\tilde{K}$ a matriz do kernel associado à função $\tilde{\varphi}$, temos que $\tilde{K} = \tilde{\varphi}(X)^T\tilde{\varphi}(X)$. Logo, a entrada $\tilde{k}_{ij}$ de $\tilde{K}$ é escrita como

$$\begin{aligned}
\tilde{k}_{ij} &= \tilde{\varphi}(x_i)^T\tilde{\varphi}(x_j) \\
&= \left(\varphi(x_i) - \frac{1}{m}\sum_{p=1}^{m}\varphi(x_p)\right)^T\left(\varphi(x_j) - \frac{1}{m}\sum_{q=1}^{m}\varphi(x_q)\right) \\
&= \varphi(x_i)^T\varphi(x_j) - \frac{1}{m}\sum_{p=1}^{m}\varphi(x_p)^T\varphi(x_j) - \frac{1}{m}\sum_{q=1}^{m}\varphi(x_i)^T\varphi(x_q) + \frac{1}{m^2}\sum_{p,q=1}^{m}\varphi(x_p)^T\varphi(x_q) \\
&= k_{ij} - \frac{1}{m}\sum_{p=1}^{m}k_{pj} - \frac{1}{m}\sum_{q=1}^{m}k_{iq} + \frac{1}{m^2}\sum_{p,q=1}^{m}k_{pq}
\end{aligned}$$

Se chamarmos de $\mathbb{1}_m$ a matriz $m \times m$ cujas entradas são todas iguais a $\frac{1}{m}$, conseguimos escrever

$$\tilde{K} = K - \mathbb{1}_m K - K\mathbb{1}_m + \mathbb{1}_m K \mathbb{1}_m \tag{4.13}$$

obtendo a centralização $\tilde{K}$ da matriz K.

Idealizamos a centralização do vetor k_x inspirados na centralização da matriz K. Com a notação da Seção 4.2, temos que

$$\tilde{\varphi}(x_{s_i}) = \varphi(x_{s_i}) - \frac{1}{n}\sum_{j=1}^{n}\varphi(x_{s_j})$$

será centrada entre o pontos de controle, ou seja, $\frac{1}{n}\sum_{i=1}^{n}\tilde{\varphi}(x_{s_i}) = 0$. Denotando por $\tilde{k}_x = \begin{bmatrix}\tilde{k}(x,x_{s_1}) & \tilde{k}(x,x_{s_2}) & \cdots & \tilde{k}(x,x_{s_n})\end{bmatrix}^T$, onde $\tilde{k}$ é o kernel associado a função $\tilde{\varphi}$, podemos escrever as coordenadas de $\tilde{k}_x$ como

$$\begin{aligned}
\tilde{k}_x(x_{s_i}) &= \tilde{\varphi}(x_x)^T\tilde{\varphi}(x_{s_i}) \\
&= \left[\varphi(x) - \frac{1}{n}\sum_{j=1}^{n}\varphi(x_{s_j})\right]^T\left[\varphi(x_{s_i}) - \frac{1}{n}\sum_{p=1}^{n}\varphi(x_{s_p})\right] \\
&= \varphi(x)^T\varphi(x_{s_i}) - \frac{1}{n}\sum_{p=1}^{n}\varphi(x)^T\varphi(x_{s_p}) - \frac{1}{n}\sum_{j=1}^{n}\varphi(x_{s_j})^T\varphi(x_{s_i}) + \frac{1}{n^2}\sum_{j,p=1}^{n}\varphi(x_{s_j})^T\varphi(x_{s_p}) \\
&= k(x,x_{s_i}) - \frac{1}{n}\sum_{p=1}^{n}k(x,x_{s_p}) - \frac{1}{n}\sum_{j=1}^{n}k(x_{s_j},x_{s_i}) + \frac{1}{n^2}\sum_{j,p=1}^{n}k(x_{s_j},x_{s_p})
\end{aligned}$$

para todo $i = 1,2,\ldots,n$. Escrevendo na forma matricial, temos

$$\tilde{k}_x = k_x - K_s 1_n - \mathbb{1}_n k_x + \mathbb{1}_n K_s 1_n \tag{4.14}$$

onde 1_n é o vetor com de tamanho n com todas as coordenadas iguais a $\frac{1}{n}$.

4.4 Projeção dos pontos de controle

Nossa técnica, efetua no primeiro passo, a projeção dos pontos de controle utilizando o Force Scheme (TEJADA; MINGHIM; NONATO, 2003). Como o Force Scheme utiliza a informação de distância, precisamos das distâncias no espaço de características para poder efetuar essa projeção inicial. Entretanto, toda informação que temos sobre o dado no espaço de características é a medida de similaridade dada pelo kernel. Dado um produto interno no espaço de características, temos automaticamente uma medida de distância proveniente desse produto interno: $d(X_i,X_j) = \sqrt{\langle X_i - X_j, X_i - X_j\rangle_H}$, $\forall X_i, X_j \in H$, onde $\langle\cdot,\cdot\rangle_H$ denota um produto interno no espaço de características H. Como o kernel define um produto interno no espaço de características, temos uma medida de distância definida pelo kernel dada por

$$\begin{aligned}
d(\varphi(x_i),\varphi(x_j)) &= \sqrt{\left[\varphi(x_i) - \varphi(x_j)\right]^T\left[\varphi(x_i) - \varphi(x_j)\right]} \\
&= \sqrt{\varphi(x_i)^T\varphi(x_i) - 2\varphi(x_i)^T\varphi(x_j) + \varphi(x_j)^T\varphi(x_j)} \\
&= \sqrt{k(x_i,x_i) - 2k(x_i,x_j) + k(x_j,x_j)}
\end{aligned} \tag{4.15}$$

A distância dada pela equação (4.15) é utilizada no cálculo da posição Y_s dos pontos de controle no espaço visual pelo Force Scheme. O Force Scheme calcula, para cada y_i no espaço visual, o vetor $v = y_j - y_i$ e move y_j na direção em que v aponta por uma fração $\Delta = \frac{D(x_i, x_j) - D_{min}}{D_{max} - D_{min}} - d(y_i, y_j)$ de seu comprimento, com D e d sendo as medidas de distância no espaço original e no espaço visual, respectivamente. Como M é linear, tem que levar a origem do espaço de características na origem do espaço visual. Portanto, para preservar essa propriedade, após calcular o posicionamento dos pontos de controle pelo Force Scheme, centralizamos Y_s de modo que seu centroide coincida com a origem do espaço visual.

4.5 Aspectos computacionais

Na construção da Kelp (Seção 4.2), assumimos que a matriz de covariância C possui colunas u_i ortonormais. Então, pela equação (4.4), temos

$$1 = u_i{}^T u_i = \sum_{p,q=1}^{n} a_{ip} a_{iq} \varphi(x_{s_p})^T \varphi(x_{s_q}) = a_i{}^T K_s a_i = \gamma_i a_i{}^T a_i$$

mostrando que os autovetores de K_s definidos na equação (4.4) são ortogonais, mas não são unitários. Porém, como as bibliotecas numéricas usualmente retornam autovetores unitários, para garantir que $\|u_i\|_2 = 1$, temos que multiplicar os autovetores unitários de K_s (dados pelas bibliotecas numéricas) por $\frac{1}{\sqrt{\gamma_i}}$.

A parte mais custosa computacionalmente da nossa técnica é o cálculo dos autovetores e autovalores da matriz K_s, cuja complexidade computacional é $O(n^3)$, onde n é o número de pontos de controle. Entretanto, esse cálculo é feito apenas uma vez e geralmente para um número pequeno de pontos de controle. O Algoritmo 4 exibe os passos para projeção utilizando a Kelp.

Algoritmo 4 – O algoritmo da Kelp

---

entrada: Conjunto de dados X e pontos de controle X_s

1 Projete X_s usando o Force Scheme com a medida de distância definida na equação (4.15). Calcule o centroide $\bar{y} = \frac{1}{n}\sum_{j=1}^{n} y_{s_j}$ e centralize Y por $Y = (y_{s_1} - \bar{y}, \ldots, y_{s_n} - \bar{y})$

2 Calcule a matriz K_s para X_s e centralize usando a equação (4.13)

3 Calcule os autovetores $a_1, \ldots, a_n$ de K_s e seus autovalores associados $\gamma_1, \ldots, \gamma_n$

4 Crie a matriz $A_{n \times p}$ com colunas $\frac{a_1}{\sqrt{\gamma_1}}, \ldots, \frac{a_p}{\sqrt{\gamma_p}}$, onde $\gamma_i \neq 0, \forall i = 1, \ldots, p$

5 Cria a matriz diagonal $\Gamma_{p \times p}{}^{-1}$ com entradas $\frac{1}{\gamma_i}$, $i = 1, \ldots, p$

6 Calcule a matriz $P = YK_sA\Gamma^{-1}A^T$

para cada $x \in X$ faça

7 Calcule k_x e centralize utilizando a equação (4.14)

8 Calcule a projeção $y = Pk_x$

4.6 Resultados e comparações

Os resultados apresentados nessa seção foram gerados utilizando pontos de controle escolhidos aleatoriamente e em quantidade $n = \sqrt{m}$, onde m é o tamanho de todo o conjunto de dado (uma discussão sobre o tamanho do conjunto de pontos de controle pode ser encontrada em (PAULOVICH; SILVA; NONATO, 2010)). Alguns experimentos utilizam uma estratégia diferente para a escolha dos pontos de controle, deixaremos esses casos claros no decorrer do texto.

Começaremos avaliando a Kelp quanto à sua precisão e tempo de execução. Comparamos nossa técnica com outras cinco e em oito conjuntos de dados diferentes com variabilidade na quantidade de instâncias e dimensão do dado (Tabela 3). As técnicas utilizadas na comparação foram escolhidas por conta de suas similaridades com a Kelp; mais especificamente: precisam de um conjunto de pontos de controle e são capazes de lidar com a informação de um kernel. Além disso, essas técnicas são bem conhecidas por seu desempenho em termos de precisão e/ou tempo computacional, garantindo que as comparações são justas e se equiparam aos métodos que caracterizam o estado da arte. As técnicas escolhidas foram: Fastmap (FALOUTSOS; LIN, 1995), Hybrid (JOURDAN; MELANCON, 2004), Landmark MDS (SILVA; TENENBAUM, 2004), Pekalska (PEKALSKA et al., 1999) e PLP (PAULOVICH et al., 2011). Todas apresentam bom desempenho quanto ao stress e tempo. Quanto a implementação, Pekalska e PLP resolvem um sistema linear e precisam de bibliotecas numéricas para tal. Landmark MDS requer uma implementação eficiente da decomposição SVD, como o algoritmo LAS2 (BERRY, 1992). Fastmap e Hybrid podem ser implementadas diretamente. Assim como a Kelp, PLP e Pekalska permitem que o usuário manipule interativamente os pontos de controle no espaço visual.

Tabela 3 – Conjuntos de dados utilizados nas comparações, da esquerda para direita as colunas correspondem ao nome do conjunto de dados, tamanho, dimensão e fonte.

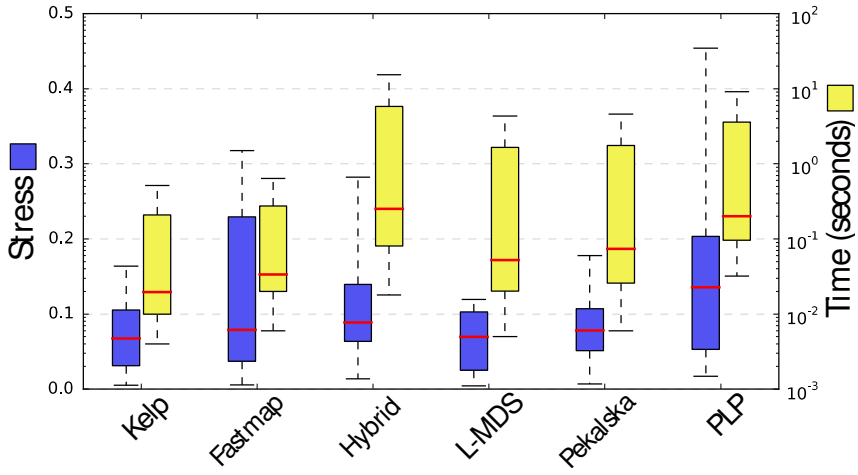
Nome	Tamanho	Dim	Fonte
wdbc	569	30	(FRANK; ASUNCION, 2010)
diabetes	768	8	(FRANK; ASUNCION, 2010)
segmentation	2.100	19	(FRANK; ASUNCION, 2010)
us-countries	3.028	14	(SHNEIDERMAN; SEO, 2008)
wine	4.898	11	(FRANK; ASUNCION, 2010)
letter rcn	20.000	16	(FRANK; ASUNCION, 2010)
mammals	50.000	72	(FRANK; ASUNCION, 2010)
viscontest	200.000	10	(WHALEN; NORMAN, 2008)

Para medir a precisão, utilizamos a função de stress dada por $\frac{1}{\sum_{ij} d_{ij}} \sum_{ij} \frac{(d_{ij} - \bar{d}_{ij})^2}{d_{ij}^2}$, onde $d_{ij} = d(\varphi(x_i), \varphi(x_j))$ é a distância dada pela equação (4.15) e $\bar{d}_{ij}$ é a distância Euclidiana entre as

imagens de x_i e x_j no espaço visual. As projeções da Kelp foram produzidas utilizando o kernel Gaussiano com parâmetro σ igual a média da variância dos dados e a distância foi calculada usando a equação (4.15).

O box plot azul na Figura 18 mostra os valores de stress obtidos nas projeções da Kelp e das demais técnicas para cada conjunto de dados da Tabela 3. Podemos facilmente dizer que a Kelp é uma das técnicas mais precisas, sendo comparável às técnicas altamente precisas como Landmark MDS e Pekalska. O box plot amarelo mostra que a Kelp também obtém bons resultados em termos de tempo de execução, sendo comparável à Fastmap, a qual é conhecida por sua eficiência computacional. Observe que a Kelp é quase uma ordem de magnitude mais rápida que Landmark MDS e Pekalska, ambas técnicas semelhantes à Kelp em termos de precisão.

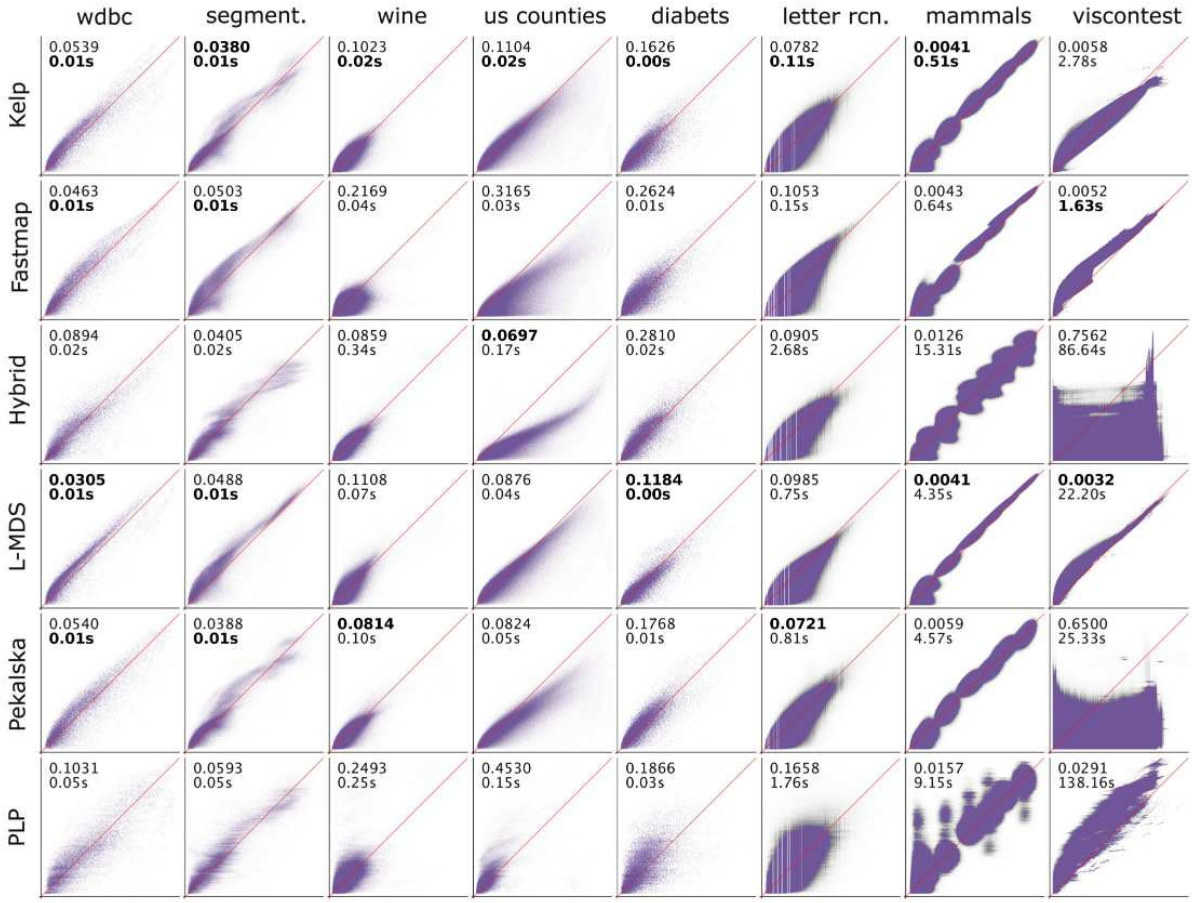
Figura 18 – Box plot do stress e tempo em cada conjunto de dado.



Os gráficos de distância original × distância projetada apresentados na Figura 19 atestam a precisão visual da Kelp. Observe que o formato dos gráficos da Kelp estão próximos da linha de 45° em quase todos os testes, atestando que a vizinhança dos pontos é preservada no espaço visual. O mesmo não é verdadeiro para outras técnicas como Hybrid e PLP, cujo resultado está espalhado na direção da diagonal.

Os conjuntos de dados utilizados nas comparações são compostos por instâncias contidas em espaços vetoriais reais, permitindo assim o uso de técnicas altamente precisas como a LAMP (JOIA et al., 2011). Entretanto, a Figura 20 diz que nossa técnica não é desnecessária para esse tipo de dado, pois as projeções da Kelp têm grupos melhor definidos que os gerados por projeções que tomam os dados no seu espaço original. As Figuras 20a e 20b mostram que as projeções resultantes da aplicação da Kelp e da LAMP no conjunto de dados Segmentation e as Figuras 20c e 20d mostram os resultados num conjunto de dados com 574 artigos científicos composto por três tópicos diferentes (PAULOVICH et al., 2008). As projeções da Kelp na Figura 20 foram produzidas kernelizando o dado utilizando o kernel Gaussiano com parâmetro σ igual a média da variância dos dados (bag-of-words do conjunto de dados de artigos científicos). O layout produzido pela Kelp é mais bem definido, evidenciando os grupos e instâncias similares.

Figura 19 – Gráfico de distância original × distância projetada. Os números no canto superior significam stress e tempo de execução (em segundos). Valores em negrito são os melhores para cada conjunto de dado.

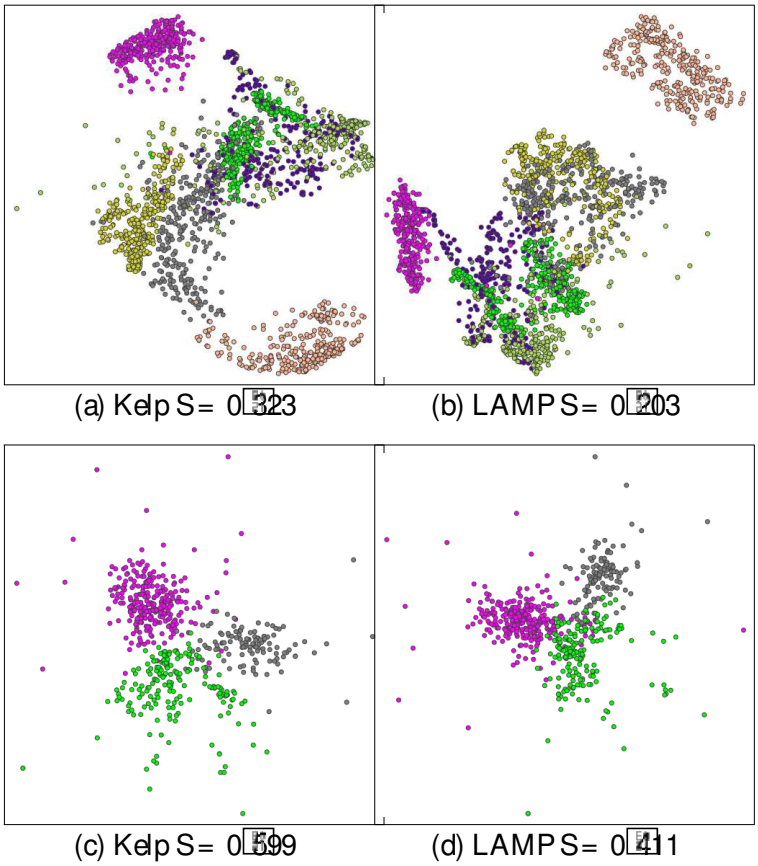


Os resultados da Kelp quanto à silhueta corroboram essa afirmação. A silhueta foi calculada segundo $S = \frac{1}{m}\sum_i \frac{(b_i - a_i)}{\max\{a_i, b_i\}}$ como na Seção 3.3.2. A silhueta varia no intervalo $[-1, 1]$ e, quanto maior o valor de S, melhor é a coesão e a separação dos grupos.

A sensibilidade da Kelp quanto a intervenção do usuário é analisada na Figura 21. A figura no canto superior direito mostra a posição dos pontos de controle, enquanto que a imagem maior mostra a projeção resultante; a Figura 21a mostra a projeção produzida pela Kelp com o posicionamento dos pontos de controle dados pelo Force Scheme. As Figuras 21b, 21c e 21d mostram as projeções da Kelp, PLMP e LAMP após a manipulação do usuário nos pontos de controle, que agrupou o dado segundo suas classes. Observe que a Kelp obtém novamente o melhor resultado quanto à silhueta, sendo superior inclusive à LAMP, que é conhecida por ser bem sensível à intervenção do usuário.

A Figura 22 apresenta o comportamento da Kelp quanto à variação no número de pontos de controle. O serrilhado no gráfico é devido a seleção aleatória dos pontos de controle, fenômeno já conhecido na literatura (PAULOVICH; SILVA; NONATO, 2010). Observe que a amplitude da oscilação é pequena, perto de 0.05. Na prática, observamos que a Kelp produz bons resultados mesmo quando utilizamos uma quantidade reduzida de pontos de controle, o que nos permitiu

Figura 20 – Melhorando a coesão e separabilidade dos grupos usando a Kelp. A silhueta (S) é maior para as projeções da Kelp comparada com técnicas baseadas em distância Euclidiana.



(a) Kelp S= 0.323 (b) LAMP S= 0.203

(c) Kelp S= 0.599 (d) LAMP S= 0.411

tomar $\sqrt{m}$ nos nossos experimentos.

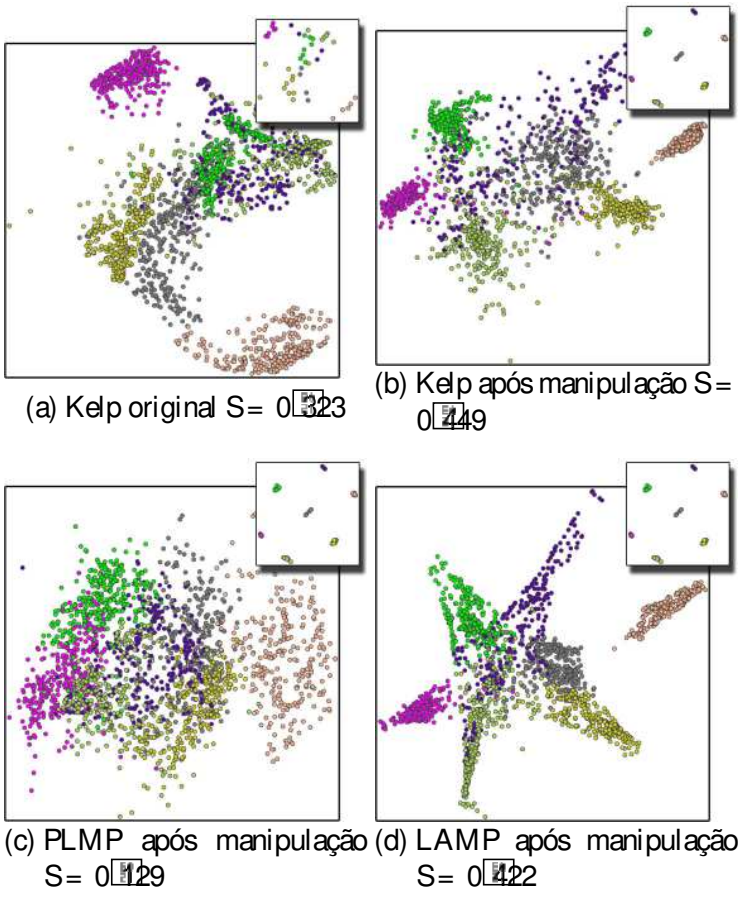
Analisamos também a Kelp segundo alguns artefatos que podem aparecer em projeções multidimensionais, denominados tears e falsos vizinhos, seguindo a metodologia CheckViz desenvolvida em (LESPINATS; AUPETIT, 2011). Tears aparecem quando instâncias vizinhas no espaço original são mapeadas distantes no espaço visual, enquanto que falsos vizinhos, como o nome induz, são instâncias que foram mapeadas próximas no espaço visual mas que são distantes no espaço original. Esses artefatos podem ser detectados através de duas medidas de stress:

$$\sum_{i<j} |d_{ij} - \bar{d}_{ij}|^2 F(d_{ij}) \tag{4.16}$$

$$\sum_{i<j} |d_{ij} - \bar{d}_{ij}|^2 F(\bar{d}_{ij}) \tag{4.17}$$

onde d_{ij} e $\bar{d}_{ij}$ representam as medidas de distância no espaço original e no espaço visual e F é uma função de peso. Supondo d_{ij} grande e $\bar{d}_{ij}$ pequeno, as instâncias i e j são falsos vizinhos. Como na equação (4.16) F depende apenas de d_{ij}, podemos escolher F de modo que o resultado da soma seja pequeno, fazendo assim com que a equação (4.16) aponte tears quando seu valor for alto. Analogamente, se d_{ij} é pequeno e $\bar{d}_{ij}$ é grande, temos que as instâncias i e j são tears e podemos utilizar a equação (4.17) para apontar falsos vizinhos quando seu valor for

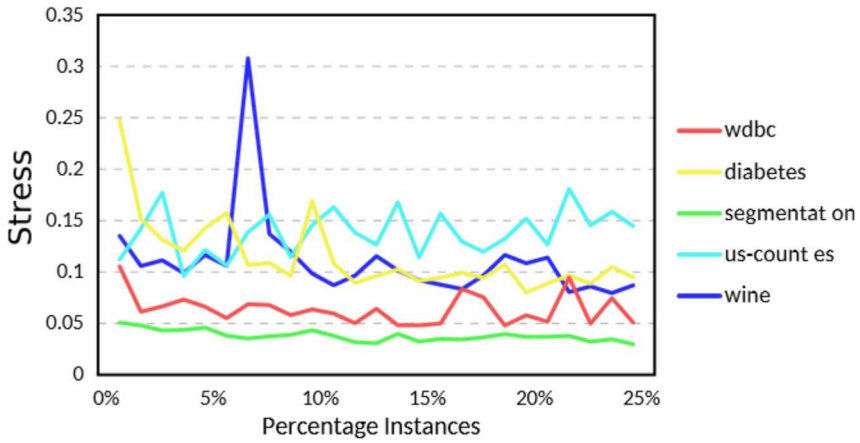
Figura 21 – Comparando a sensibilidade da Kelp quanto a intervenção do usuário. Canto superior direito mostra a posição dos pontos de controle.



(a) Kelp original S= 0.323 (b) Kelp após manipulação S= 0.449

(c) PLMP após manipulação S= 0.129 (d) LAMP após manipulação S= 0.422

Figura 22 – Variando o número de pontos de controle: o stress estabiliza próximo dos 5% do número de instâncias.



alto. Lespinats e Aupetit (2011) recomendam a escolha de F como a função de peso Heaviside:

$$F_\sigma(x) = \begin{cases} 1 & \text{se } x < \sigma \\ 0 & \text{caso contrário} \end{cases}$$

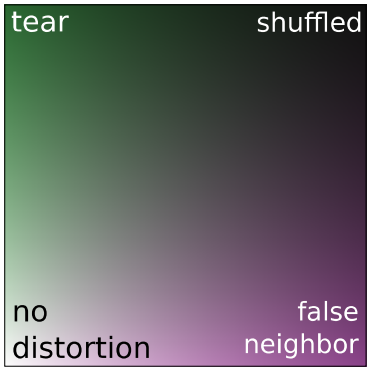
Utilizamos o seguinte código de cores para representar essas distorções: roxo para indicar os falsos vizinhos, verde indicando tears, preto indica que a instância sofre de ambas distorções

simultaneamente e branco representando as instâncias sem distorções. Definindo, para cada instância x_i, o par ordenado (f_i, t_i), onde

$$f_i = \sum_j |d_{ij} - \bar{d}_{ij}|^2 F(d_{ij})$$
$$t_i = \sum_j |d_{ij} - \bar{d}_{ij}|^2 F(\bar{d}_{ij})$$

podemos visualizar o código de cor que descreve os tipos de distorção no plano e, mais ainda, quantificar a intensidade dessas cores (Figura 23).

Figura 23 – Espaço de cores utilizado na análise de tears e falsos vizinhos.



Lespinats e Aupetit (2011) se baseiam em duas regras para analisar os resultados:

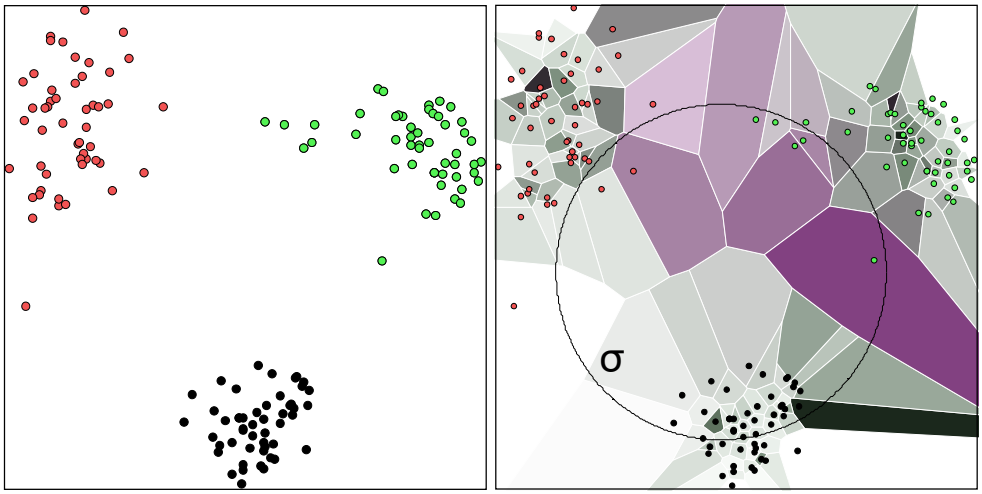
- Regra da separação: se várias instâncias roxas são disjuntas, então o espaço entre elas é verdadeiro no espaço original. Pois, caso contrário, essas instâncias corresponderiam a tears e deveriam assumir a cor verde ou preto.
- Regra da sobreposição: se várias instâncias verdes estão próximas umas das outras ou se sobrepõe, então estão localizadas próximas umas das outras no espaço original. Caso contrário, deveriam assumir a cor roxa ou preta.

A Figura 24 mostra o resultado obtido pela Kelp na metodologia CheckViz. As instâncias interiores aos grupos indicam a presença de tears, mas falsos vizinhos são observados apenas para pontos posicionados na fronteira dos grupos (entre os grupos). Pelas regras acima, temos a melhor configuração possível, pois dentro dos grupos a metodologia indica uma verdadeira sobreposição e na fronteira dos grupos as instâncias são apontadas como falsos vizinhos indicando uma verdadeira separação no espaço visual.

4.7 Aplicações

Nesta seção ilustramos a utilidade de Kelp em três aplicações distintas. A primeira aplicação é motivada pela discussão feita no início deste capítulo: queremos entender o comportamento

Figura 24 – Verificando a qualidade da projeção segundo a metodologia CheckViz: (a) projeção de um conjunto de dados artificial com 150 instâncias e quatro dimensões (GUYON, 2003); (b) regiões roxas indicam falsos vizinhos e regiões indicam tears.



(a) Projeção da Kelp (b) Análise CheckViz

da função φ definida implicitamente pelo kernel. Mais especificamente, utilizamos a Kelp como ferramenta para analisar o efeito do kernel nas relações de vizinhança do dado. A segunda aplicação explora a interatividade da Kelp para auxiliar tarefas de classificação utilizando Support Vector Machine (SCHÖLKOPF; SMOLA, 2001) (SVM). A última aplicação mostra como o processo de visualização pode tirar proveito de técnicas baseadas em kernel como a Kelp.

4.7.1 Mudança de vizinhança induzidas pelo kernel

Como motivado no início do Capítulo, entender o efeito de um kernel nas estruturas de vizinhança do dado é de fundamental importância para uma escolha apropriada, construção e aprimoramento de um kernel para uma tarefa específica.

Nossa abordagem utiliza uma métrica para comparar as estruturas de vizinhança definidas no espaço original (usualmente Cartesiano), onde o dado se encontra, e sua imagem no espaço de características induzido pelo kernel. A métrica é definida da seguinte forma: seja $\delta_i = x_i - \frac{1}{\#N_i}\sum_{j\in N_i} x_j$ a coordenada diferencial de x_i, onde N_i denota os índices dos k vizinhos mais próximos de x_i e $\#N_i$ é a cardinalidade do conjunto N_i. Assim, a norma $k\delta_i k$ mede o quão longe x_i está do centroide dos seus vizinhos. A norma da coordenada diferencial δ_{φ_i} de $\varphi_i = \varphi(x_i)$ no

espaço de características é dada por $\|\delta_{\varphi_i}\| = \sqrt{\delta_{\varphi_i}{}^T \delta_{\varphi_i}}$. Logo

$$\begin{aligned}\|\delta_{\varphi_i}\|^2 &= \left(\varphi_i - \frac{1}{\#N_i}\sum_{j\in N_i}\varphi_j\right)^T \left(\varphi_i - \frac{1}{\#N_i}\sum_{j\in N_i}\varphi_j\right) \\ &= \varphi_i^T\varphi_i - \frac{1}{\#N_i}\sum_{j\in N_i}\varphi_i^T\varphi_j - \frac{1}{\#N_i}\sum_{j\in N_i}\varphi_j^T\varphi_i + \frac{1}{(\#N_i)^2}\sum_{j,s\in N_i}\varphi_j^T\varphi_s \\ &= k(x_i,x_i) - \frac{2}{\#N_i}\sum_{j\in N_i}k(x_i,x_j) + \frac{1}{(\#N_i)^2}\sum_{j,s\in N_i}k(x_j,x_s)\end{aligned} \tag{4.18}$$

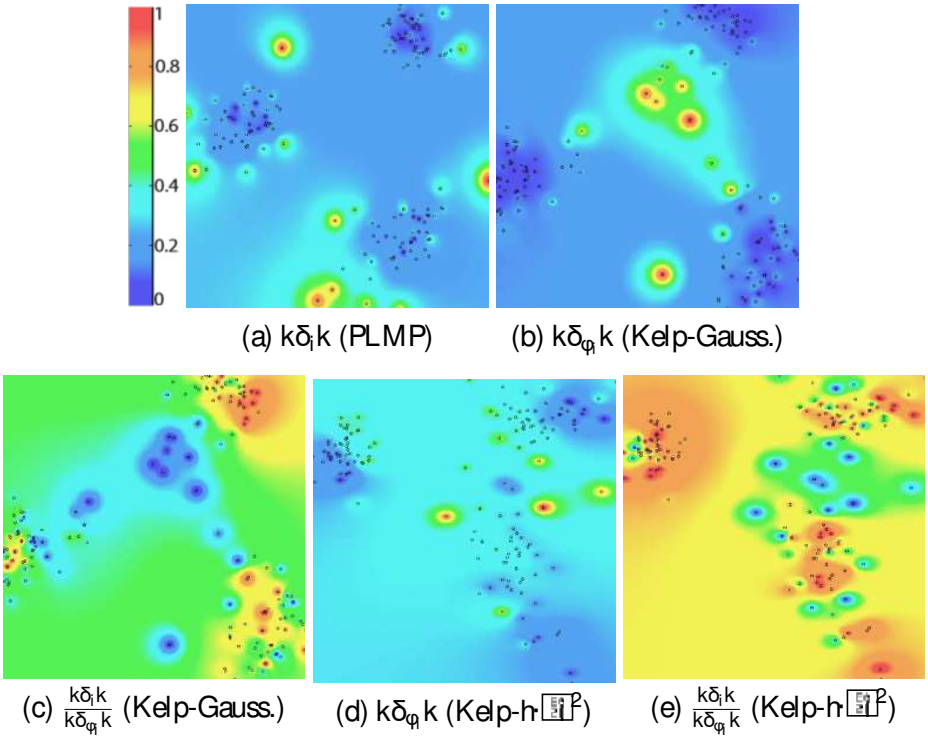
A equação (4.18) mostra que a norma da coordenada diferencial no espaço de características pode ser obtida a partir do kernel, tornando possível medir quão longe cada instância está do centroide de seus vizinhos nesse espaço, mesmo sem saber as coordenadas da instância nem dos vizinhos.

As cores nas Figuras 25a e 25b representam os valores correspondentes de $\|\delta_i\|$ e $\|\delta_{\varphi_i}\|$ calculados em cada instância (no conjunto de dados artificial (GUYON, 2003)). O layout dos pontos foi gerado pela PLMP e pela Kelp, respectivamente. Regiões em vermelho correspondem a valores altos de $\|\delta_i\|$ e $\|\delta_{\varphi_i}\|$, enquanto que regiões em azul correspondem a valores baixos e regiões em verde são valores intermediários. Escolhemos PLMP para projetar o dado a partir do espaço original, porque, como a Kelp, a PLMP faz uso de uma transformação linear, tornando a comparação mais justa. Observe que, mais uma vez, a projeção baseada em kernel fez os grupos ficarem melhor definidos.

A razão $\|\delta_i\| / \|\delta_{\varphi_i}\|$ mede a mudança na vizinhança quando o dado é levado no espaço de características pela φ definida implicitamente pelo kernel. Valores próximos de 1 indicam que a estrutura da vizinhança não mudou, valores próximos de 0 indicam que as instâncias ficaram mais distantes dos seus vizinhos e valores maiores que 1 significa que as instâncias ficaram centralizadas com relação a seus vizinhos quando não eram centradas no espaço original. Utilizando uma função de transferência como da Figura 26, podemos visualizar as regiões onde a vizinhança é mais afetada pelo kernel. O fundo da Figura 25 foi colorido interpolando o valor da coordenada diferencial (ou de sua razão) de cada instância para uma grade regular no fundo. Figura 25b nos diz que o kernel Gaussiano posiciona melhor os pontos em termos dos vizinhos que estão intragrupo; ou seja, o kernel Gaussiano tende a posicionar instâncias mais próximas do centroide do seu grupo. Entretanto, a Figura 25c claramente mostra que, analisando a razão entre as normas das coordenadas diferenciais, as regiões em vermelho (valores próximos de zero e maiores que um) aparecem no interior dos grupos. Como $\|\delta_{\varphi_i}\|$ é pequeno no interior de grupos bem definidos (Figura 25b) e os grupos não se espalharam pela ação do kernel, concluímos que os altos valores da razão são devidos a um agrupamento mais apertado feito pelo kernel Gaussiano. Portanto, como esperado, o kernel Gaussiano tende a criar grupos melhor definidos.

A mesma análise pode ser feita com outros kernels além do Gaussiano, como ilustrado nas Figuras 25d e 25e. A Figura 25d descreve $\|\delta_{\varphi_i}\|$ quando o kernel polinomial dado por

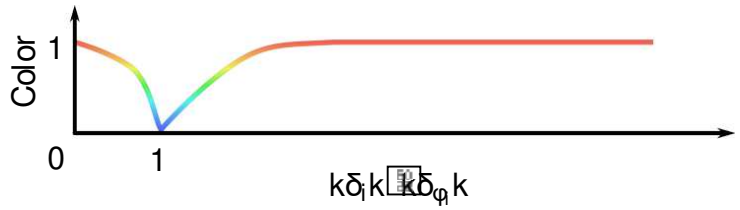
Figura 25 – Visualizando como o kernel afeta as estruturas de vizinhança: magnitude das coordenadas diferenciais com layout gerado pela PLMP com distância Euclidiana em (a), pela Kelp com kernel Gaussiano em (b) e kernel polinomial em (d); magnitude da razão com Kelp-Gaussiano em (c) e Kelp-Polinomial em (e).



(a) $k\delta_i k$ (PLMP) (b) $k\delta_{\varphi_i} k$ (Kelp-Gauss.)

(c) $\frac{k\delta_i k}{k\delta_{\varphi_i} k}$ (Kelp-Gauss.) (d) $k\delta_{\varphi_i} k$ (Kelp-h·□²) (e) $\frac{k\delta_i k}{k\delta_{\varphi_i} k}$ (Kelp-h·□²)

Figura 26 – Função de transferência.



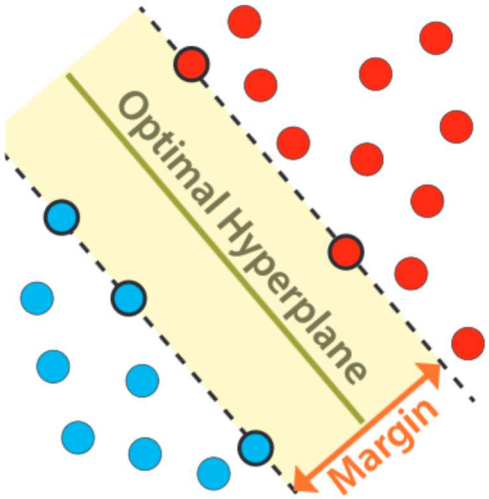
$k(x_i \square x_j) = (x_i^T x_j)^2$ é usado. Kernels polinomiais são menos intuitivos que o kernel Gaussiano, dificultando seu uso em aplicações práticas. Usando nossa ferramenta visual, podemos ver que o kernel polinomial acima se comporta quase que da mesma forma que o kernel Gaussiano, evitando a criação de "outliers" próximos dos grupos enquanto que aperta as instâncias dentro dos mesmos.

Como podemos ver claramente, coordenadas diferenciais são bastante efetivas para visualização das mudanças na estrutura de vizinhança induzidas por um kernel. Vale mencionar que, até onde sabemos, essa é a primeira vez que coordenadas diferenciais foram utilizadas para medir a variação da vizinhança no contexto de dados envolvendo kernel, sendo essa mais uma contribuição do nosso trabalho.

4.7.2 Visualização de SVM

Support Vector Machine (SVM) é um classificador linear que opera no espaço de características onde o hiperplano de separação tem margem máxima. Intuitivamente, instâncias distantes da margem no espaço de características são classificadas com um grau de confiabilidade maior do que instâncias mais próximas das margens. A Figura 27 ilustra o que acontece no espaço de características durante o processo de classificação do SVM.

Figura 27 – Modelo intuitivo da separação feita no espaço de características quando aplicamos SVM.

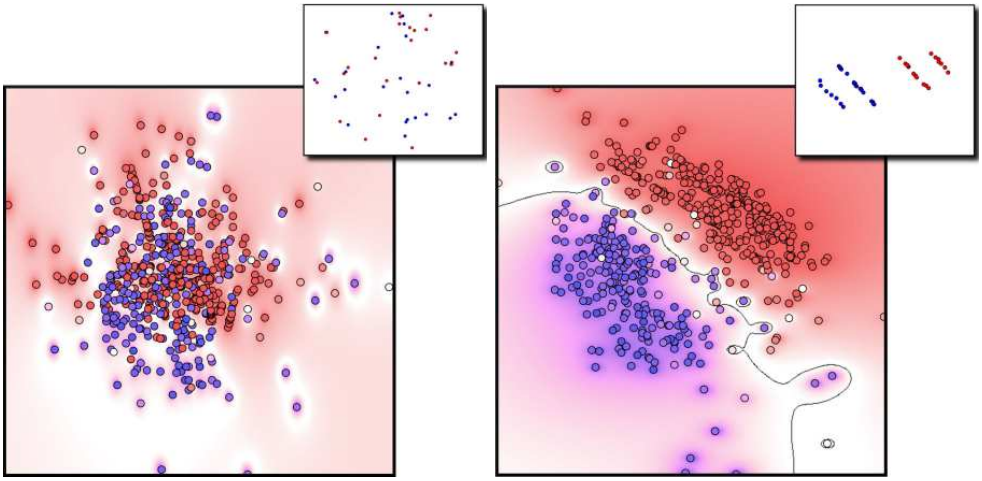


Nós tiramos proveito da interatividade da Kelp para mudar a posição dos pontos de controle no espaço visual de modo a separar os dados como no modelo intuitivo que temos do SVM. A Figura 28 apresenta a projeção do conjunto de dados wdbc (Tabela 3) usando kernel Gaussiano. O mesmo kernel Gaussiano foi usado no SVM. Utilizando a biblioteca LIBSVM (CHANG; LIN, 2011) para executar a classificação com SVM, nós obtemos, para cada instância, a probabilidade dela pertencer a cada uma das classes. Cores mais escuras na Figura 28 indicam maior probabilidade da instância realmente pertencer aquela classe, enquanto que cores mais claras indicam instâncias cuja classificação não é tão confiante. A linha preta corresponde a região onde a probabilidade interpolada é igual a 50%. Observe na Figura 28a que as regiões de confiabilidade não podem ser determinadas quando os pontos de controle não estão separados no espaço visual. Como o conjunto de dados utilizado possui um número pequeno de instâncias, nós utilizamos 10% das instâncias para compor o conjunto dos pontos de controle.

Entretanto, quando organizamos os pontos de controle interativamente separando as classes, como na Figura 28b, o layout resultante claramente tem a forma do modelo intuitivo do SVM, tornando fácil interpretar o comportamento do classificador. Observe que inclusive a margem onde o classificador fica em dúvida é exibida corretamente. O fundo da Figura 28 é colorido interpolando a probabilidade (dada pelo SVM) de cada instância pertencer a uma das classes.

É importante salientar que outros métodos capazes de visualizar o resultado do SVM (HA-

Figura 28 – Realizando o modelo intuitivo do SVM: (a) projeção inicial onde a linha de separação não é clara e (b) projeção após a manipulação dos pontos de controle. A figura no canto superior apresenta o posicionamento dos pontos de controle.



(a) (b)

MEL, 2006; JAKULIN et al., 2005) assumem que o dado pode ser imerso num espaço cartesiano e o kernel é utilizado apenas no classificador, enquanto que a Kelp utiliza exclusivamente a informação do kernel.

4.7.3 Segmentação de imagem baseada em kernel

Interação com usuário é uma característica que define uma classe importante dos métodos de segmentação. Nesse contexto, técnicas que permitem que o usuário interaja no espaço da imagem guiando o processo de segmentação são maioria (CASACA et al., 2013). Entretanto, técnicas que operam no espaço de características não são comuns, apesar de trabalhos recentes mostrarem vantagem em tarefas como coloração e retrieval (CASACA et al., 2012; MAMANI et al., 2013).

A seguir, mostraremos como a Kelp pode ser utilizada para auxiliar num processo interativo de segmentação que opera diretamente no espaço de características. Mais especificamente, construímos um kernel baseado na filtragem bilateral (TOMASI; MANDUCHI, 1998) que permite utilizar a Kelp como ferramenta para que o usuário interaja diretamente no espaço de características.

Sejam I a imagem original (dado de entrada) e I' a imagem resultante da filtragem bilateral:

$$I'_p = \frac{1}{W} \sum_{q \in N_p} I_p G_{\sigma_1}(\|I_p - I_q\|) G_{\sigma_2}(\|p - q\|)$$

$$\text{com } W = \sum_{q \in N_p} G_{\sigma_1}(\|I_p - I_q\|) G_{\sigma_2}(\|p - q\|)$$

onde $G_\sigma(x) = e^{-\frac{x^2}{2\sigma^2}}$. Os valores I_p e I_p são a cor no sistema CIE-Lab do pixel p da imagem I e I, respectivamente, σ^2 é a variância tipicamente utilizada em filtros Gaussianos e N_p é a vizinhança quadrada ao redor do pixel p.

Definimos o kernel k como

$$k(p, q) = G_\sigma \left(\| I_p - I_q \| \right) \tag{4.19}$$

A Figura 29b mostra a projeção resultante da Kelp utilizando o kernel definido na equação (4.19), onde cada pixel da imagem na Figura 29a é uma instância de dado. A cor de cada ponto na projeção é a cor do pixel correspondente. Observe na projeção gerada pela Kelp que o amarelo da banana é claramente separado do amarelo do fundo da imagem (Figuras 29c e 29d). Além disso, o usuário pode definir os grupos da segmentação interativamente na projeção, o que é equivalente a definir essas regiões no espaço de características e, consequentemente, no espaço original, como mostrado nas Figuras 29e e 29f.

4.8 Discussão e limitações

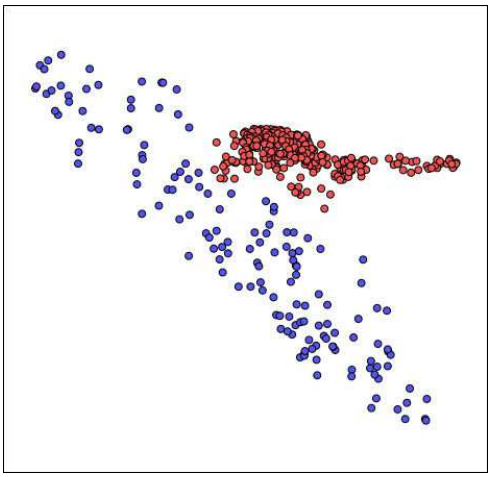
Os resultados e comparações apresentados na Seção 4.6 claramente mostram a eficiência da Kelp, que apresenta um bom balanço entre precisão e tempo de execução. A formulação matemática sólida e a facilidade de implementação (requer essencialmente o cálculo de autovetores e autovalores) são aspectos que mostram a força da Kelp. Um outro ponto interessante da técnica se deve ao fato de ser facilmente paralelizável: após o cálculo da matriz $YK_sA\Gamma^{-1}A^T$, a projeção das instâncias de dado podem ser feitas de forma independente.

Uma limitação que não está propriamente na formulação da Kelp, mas que afeta seu funcionamento é o kernel utilizado na aplicação e seus parâmetros. O kernel Gaussiano, por exemplo, depende do parâmetro σ e esse valor afeta o resultado da classificação do SVM, a redução de dimensionalidade do KPCA e da Kelp, pois todas essas técnicas dependem dos valores do kernel. Encontrar o valor correto para cada parâmetro do kernel é uma tarefa difícil. Em nossos testes utilizamos a média da variância do dado, mas não há garantia que isso irá funcionar para todo conjunto de dados. Acreditamos que, devido às suas capacidades, a Kelp pode ser uma ferramenta bastante útil para auxiliar no entendimento do efeito de um kernel permitindo uma melhor escolha desses parâmetros, bem como a criação de um novo kernel através de kernels já conhecidos.

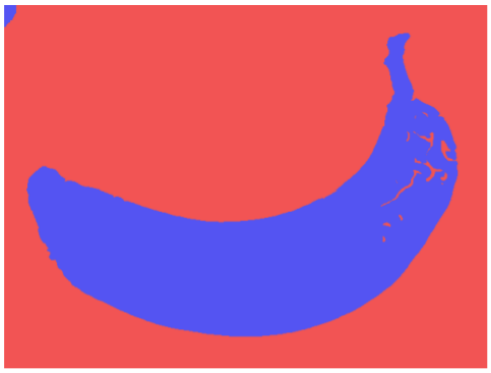
Figura 29 – Nosso pipeline de segmentação: a imagem de entrada é projetada, onde cada pixel é uma instância de dado, em seguida os grupos são definidos na projeção e o resultado desse agrupamento é exibido como uma imagem segmentada.

(a) Imagem original

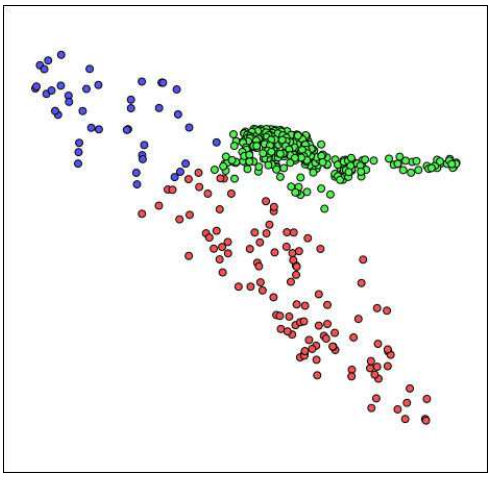
(b) Projeção da Kelp



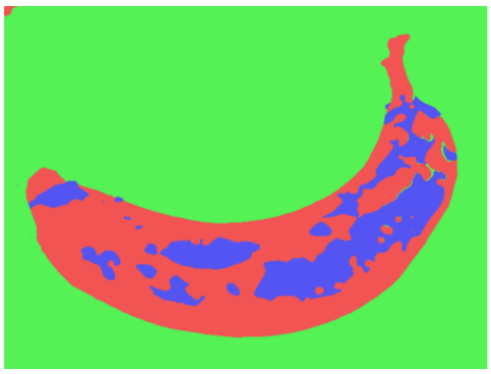
(c) Segmentação em 2 grupos



(d) Imagem segmentada



(e) Segmentação em 3 grupos



(f) Imagem segmentada

CAPÍTULO

5

CONCLUSÃO

Esta tese apresentou estudos práticos sobre a utilização de técnicas de subspace clustering e métodos de kernel para auxiliar projeções multidimensionais e a criação de uma técnica completamente nova baseada em kernel que possui capacidades até então não exploradas.

Técnicas de subspace clustering se mostram ferramentas promissoras para a análise da estrutura de subespaço intrínseca do dado. Por outro lado, quando o dado não possui tal estrutura, as técnicas não podem ser aplicadas. A informação de classe gerada pela LRR nos permite utilizar técnicas como a LDA em tarefas como classificação e redução de dimensionalidade com uma boa qualidade em termos das métricas que testamos.

Estimar a dimensão e uma base para esses subespaços são tarefas que pretendemos executar em trabalhos futuros, além de estudar o efeito e a eficiência dessas novas informações em tarefas de redução de dimensionalidade.

Do lado dos métodos de kernel, propomos uma nova técnica de projeção multidimensional, chamada Kelp, que possui uma base matemática sólida e com desempenho, em termos de precisão e tempo de execução, comparável e até melhores do que o das técnicas que são o estado da arte. A Kelp apresentou ótimos resultados em tarefas baseadas em kernel e, por conta disso, possui um grande potencial em tarefas gerais, bastando que um kernel próprio para tal aplicação seja utilizado. Além disso, sua flexibilidade, eficiência e facilidade de implementação a torna uma técnica de projeção multidimensional muito atrativa para aplicações em geral, particularmente, para aplicações que possam tirar proveito de métodos de kernel. Um trabalho sobre a construção de novos kernels através da combinação de kernels conhecidos já está em andamento.

REFERÊNCIAS

AGARWAL, P. K.; MUSTAFA, N. H. K-means projective clustering. In: Proceedings of the Twenty-third ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems. New York, NY, USA: ACM, 2004. (PODS '04), p. 155–165. ISBN 158113858X. Citado na página 32.

ALZATE, C.; SUYKENS, J. Kernel component analysis using an epsilon-insensitive robust loss function. IEEE Transactions on Neural Networks, IEEE Press, Piscataway, NJ, USA, v. 19, p. 1583–1598, 2008. ISSN 1045-9227. Citado na página 43.

BARBOSA, A.; PAULOVICH, F. V.; PAIVA, A.; GOLDENSTEIN, S.; PETRONETTO, F.; NONATO, L. G. Visualizing and interacting with kernelized data. IEEE Transactions on Visualization & Computer Graphics, IEEE Computer Society, Los Alamitos, CA, USA, v. 22, n. 3, p. 1314–1325, 2016. ISSN 1077-2626. Citado nas páginas 21, 29, 30 e 57.

BARBOSA, A.; SADLO, F.; NONATO, L. G. An initial study on high-dimensional data visualization through subspace clustering. In: . [S.l.]: Sociedade Brasileira de Computação, 2015. Citado nas páginas 28, 30 e 45.

BERRY, M. W. Large-scale sparse singular value computations. International Journal of Supercomputer Applications, v. 6, n. 1, p. 13–49, 1992. Citado na página 64.

BISHOP, C. M. Pattern Recognition and Machine Learning. Secaucus, NJ, USA: Springer-Verlag New York, Inc., 2006. ISBN 0387310738. Citado nas páginas 21, 28, 31, 40, 50 e 87.

BOULT, T. E.; BROWN, L. G. Factorization-based segmentation of motions. In: Proceedings of the IEEE Workshop on Visual Motion. [S.l.: s.n.], 1991. p. 179–186. Citado na página 32.

BRADLEY, P. S.; MANGASARIAN, O. L.; PARDALOS, P. k-plane clustering. Journal of Global Optimization, v. 16, n. 1, p. 249–252, 2000. Citado na página 32.

CAI, J.; CANDÈS, E.; SHEN, Z. A singular valor thresholding algorithm for matrix completion. SIAM J. on Optimization, v. 20, n. 4, p. 1956–1982, 2010. Citado na página 47.

CASACA, W.; GOMEZ-NIETO, E.; FERREIRA, C. O.; TAVARES, G.; PAGLIOSA, P.; PAULOVICH, F.; NONATO, L. G.; PAIVA, A. Colorization by multidimensional projection. In: IEEE. Conference on Graphics, Patterns and Images (SIBGRAPI). [S.l.], 2012. p. 32–38. Citado na página 74.

CASACA, W.; PAIVA, A.; GOMEZ-NIETO, E.; JOIA, P.; NONATO, L. G. Spectral image segmentation using image decomposition and inner product-based metric. Journal of mathematical imaging and vision, Springer, v. 45, n. 3, p. 227–238, 2013. Citado na página 74.

CHANG, C.-C.; LIN, C.-J. LIBSVM: A library for support vector machines. ACM Transactions on Intelligent Systems and Technology, v. 2, p. 27:1–27:27, 2011. Software available at <http://www.csie.ntu.edu.tw/~cjlin/libsvm>. Citado na página 73.

CHOI, H.; CHOI, S. Kernel isomap. Electronics Letters, v. 40, n. 25, p. 1612–1613, 2004. ISSN 0013-5194. Citado na página 43.

COSTEIRA, J.; KANADE, T. A multibody factorization method for independetly moving objects. International Journal of Computer Vision, v. 29, n. 3, p. 159–179, 1998. Citado nas páginas 27, 32 e 47.

DERKSEN, H.; MA, Y.; HONG, W.; WRIGHT, J. Segmentation of multivariate mixed data via lossy coding and compression. IEEE Transactions on Pattern Analysis and Machine Intelligence, v. 3, p. 1546–1562, 2007. Citado na página 32.

ELHAMIFAR, E.; VIDAL, R. Sparse subspace clustering. In: Proceedings of the Conference on Computer Vision and Pattern Recognition. [S.l.: s.n.], 2009. p. 2790–2797. Citado na página 32.

FALOUTSOS, C.; LIN, K. FastMap: A fast algorithm for indexing, datamining and visualization of traditional and multimedia databases. In: ACM SIGMOD. [S.l.: s.n.], 1995. p. 163–174. Citado na página 64.

FRANK, A.; ASUNCION, A. UCI machine learning repository. 2010. Disponível em: <http://archive.ics.uci.edu/ml>. Citado nas páginas 52, 54 e 64.

GEAR, C. W. Multibody grouping from motion images. International Journal of Computer Vision, Springer Netherlands, v. 29, n. 2, p. 133–150, ago. 1998. Citado na página 32.

GOWER, J.; DIJKSTERHUIS, G. Procrustes Problems. [S.l.]: Oxford University Press, 2004. (Oxford Statistical Science Series). ISBN 9780198510581. Citado na página 37.

GUYON, I. Design of experiments of the NIPS 2003 variable selection benchmark. In: NIPS 2003 workshop on feature extraction and feature selection. [s.n.], 2003. Disponível em: <http://www.nipsfsc.ecs.soton.ac.uk/datasets/>. Citado nas páginas 54, 70 e 71.

HAM, J.; LEE, D. D.; MIKA, S.; SCHÖLKOPF, B. A kernel view of the dimensionality reduction of manifolds. New York, NY, USA: ACM, 2004. 47-54 p. (ICML '04). ISBN 1-58113-838-5. Citado nas páginas 21, 41 e 42.

HAMEL, L. Visualization of support vector machines with unsupervised learning. In: IEEE Symposium on Computational Intelligence and Bioinformatics and Computational Biology. [S.l.: s.n.], 2006. p. 1–8. Citado na página 74.

HONG, W.; WRIGHT, J.; HUANG, K.; MA, Y. Multi-scale hybrid linear models for lossy image representation. IEEE Trans. on Image Processing, v. 15, n. 12, p. 3655–3671, 2006. Citado na página 27.

INABA, F.; SALLES, E.; RAUBER, T. W. Kernel sammon map. In: SIBGRAPI. [S.l.: s.n.], 2011. p. 329–336. Citado nas páginas 42 e 43.

JAKULIN, A.; MOZINA, M.; DEMSAR, J.; BRATKO, I.; ZUPAN, B. Nomograms for visualizing support vector machines. In: ACM SIGKDD. [S.l.: s.n.], 2005. p. 108–117. Citado na página 74.

JOIA, P.; COIMBRA, D.; CUMINATO, J. A.; PAULOVICH, F. V.; NONATO, L. G. Local affine multidimensional projection. IEEE Trans. on Visualization and Computer Graphics, v. 17, n. 12, p. 2563–2571, 2011. Citado nas páginas 21, 28, 37, 38, 51 e 65.

JOIA, P.; GOMEZ-NIETO, E.; NETO, J. B.; CASACA, W.; BOTELHO, G.; PAIVA, A.; NONATO, L. G. Class-specific metrics for multidimensional data projection applied to CBIR. The Visual Computer, Springer-Verlag, v. 28, p. 1027–1037, 2012. ISSN 0178-2789. Citado nas páginas 21, 38 e 39.

JOURDAN, F.; MELANCON, G. Multiscale hybrid mds. In: Information Visualization. [S.l.: s.n.], 2004. Citado nas páginas 21, 34 e 64.

KANATANI, K. Motion segmentation by subspace separation and model selection. IEEE Int. Conf. on Computer Vision, v. 2, p. 586–591, 2001. Citado na página 27.

LESPINATS, S.; AUPETIT, M. Checkviz: Sanity check and topological clues for linear and non-linear mappings. Computer Graphics Forum, Wiley Online Library, v. 30, n. 1, p. 113–125, 2011. Citado nas páginas 67, 68 e 69.

LIN, Z.; CHEN, M.; WU, L.; MA, Y. The augmented Lagrange multiplier method for exact recovery of corrupted low-rank matrices. [S.l.], 2009. Citado nas páginas 21 e 47.

LIPMAN, Y.; SORKINE, O.; COHEN-OR, D.; LEVIN, D.; ROSSI, C.; SEIDEL, H.-P. Differential coordinates for interactive mesh editing. In: IEEE. Shape Modeling Applications, 2004. Proceedings. [S.l.], 2004. p. 181–190. Citado nas páginas 29 e 57.

LIU, G. Implementação da técnica Low-Rank Representation. 2016. Disponível em: <https://sites.google.com/site/guangcanliu/>. Citado na página 47.

LIU, S.; WANG, B.; THIAGARAJAN, J. J.; BREMER, P.-T.; PASCUCCI, V. Visual Exploration of High-Dimensional Data through Subspace Analysis and Dynamic Projections. Computer Graphics Forum, The Eurographics Association and John Wiley & Sons Ltd., p. 271–280, 2015. Citado na página 32.

MAATEN, L. van der; HINTON, G. Visualizing data using t-SNE. Journal of Machine Learning Research, v. 9, p. 2579–2605, 2008. Citado nas páginas 21, 28, 39 e 51.

MAMANI, G. M.; FATORE, F. M.; NONATO, L. G.; PAULOVICH, F. V. User-driven feature space transformation. In: WILEY ONLINE LIBRARY. Computer Graphics Forum. [S.l.], 2013. v. 32, n. 3, p. 291–299. Citado na página 74.

PAULOVICH, F. V.; ELER, D. M.; POCO, J.; BOTHA, C. P.; MINGHIM, R.; NONATO, L. G. Piecewise Laplacian-based projection for interactive data exploration and organization. Computer Graphics Forum, v. 30, n. 3, p. 1091–1100, 2011. Citado nas páginas 21, 27, 37, 38 e 64.

PAULOVICH, F. V.; NONATO, L. G.; MINGHIM, R.; LEVKOWITZ, H. Least square projection: A fast high-precision multidimensional projection technique and its application to document mapping. IEEE Transactions on Visualization and Computer Graphics, v. 14, n. 3, p. 564–575, 2008. Citado nas páginas 21, 34, 35 e 65.

PAULOVICH, F. V.; SILVA, C. T.; NONATO, L. G. Two-phase mapping for projecting massive data sets. IEEE Trans. on Visualization and Computer Graphics, v. 16, n. 6, p. 1281–1290, 2010. Citado nas páginas 21, 35, 36, 64 e 66.

PEKALSKA, E.; RIDDER, D. de; DUIN, R. P. W.; KRAAIJVELD, M. A. A new method of generalizing Sammon mapping with application to algorithm speed-up. In: Annual Conf. Advanced School for Comput. Imag. [S.l.: s.n.], 1999. p. 221–228. Citado nas páginas 33 e 64.

SAMMON, J. W. A nonlinear mapping for data structure analysis. IEEE Transactions on Computers, IEEE Computer Society, Washington, DC, USA, v. 18, p. 401–409, 1969. ISSN 0018-9340. Citado nas páginas 33 e 42.

SCHÖLKOPF, B.; SMOLA, A. Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond. Cambridge, MA, USA: MIT Press, 2001. ISBN 0262194759. Citado nas páginas 21, 28, 58 e 70.

SCHÖLKOPF, B.; SMOLA, A.; MÜLLER, K.-R. Nonlinear component analysis as a kernel eigenvalue problem. Neural Computation, MIT Press, Cambridge, MA, USA, v. 10, p. 1299–1319, 1998. ISSN 0899-7667. Citado nas páginas 21, 40, 41 e 61.

SHI, J.; MALIK, J. Normalized cuts and image segmentation. IEEE Trans. Pattern Analysis and Machine Intelligence, p. 888–905, 2000. Citado na página 48.

SHNEIDERMAN, B.; SEO, J. Hierarchical Clustering Explorer for Interactive Exploration of Multidimensional Data. 2008. Disponível em: <http://www.cs.umd.edu/hcil/hce/examples/application_examples.html>. Citado na página 64.

SILVA, V. de; TENENBAUM, J. B. Sparse multidimensional scaling using landmark points. [S.l.], 2004. Citado na página 64.

TEJADA, E.; MINGHIM, R.; NONATO, L. G. On improved projection techniques to support visual exploration of multidimensional data sets. Information Visualization, v. 2, n. 4, p. 218–231, 2003. Citado na página 62.

TIPPING, M. E.; BISHOP, C. M. Mixtures of probabilistic principal component analysers. Neural Computation, v. 11, n. 2, p. 443–482, 1999. Citado na página 32.

TOMASI, C.; MANDUCHI, R. Bilateral Filtering for Gray and Color Images. In: Proceedings of the Sixth International Conference on Computer Vision. Washington, DC, USA: IEEE Computer Society, 1998. (ICCV '98), p. 839+. ISBN 81-7319-221-9. Citado na página 74.

TSENG, P. Nearest q-flat to m points. Journal of Optimization Theory and Applications, v. 105, n. 1, p. 249–252, 2000. Citado na página 32.

VIDAL, R. A tutorial on subspace clustering. IEEE Signal Processing Magazine, v. 28, n. 2, p. 52–68, 2010. Citado na página 32.

WHALEN, D.; NORMAN, M. L. Competition data set and description. In: 2008 IEEE Visualization Design Contest. [s.n.], 2008. Disponível em: <http://vis.computer.org/VisWeek2008/vis/contests.html>. Citado na página 64.

YANG, A. Y.; WRIGHT, J.; MA, Y.; SASTRY, S. S. Unsupervised segmentation of natural images via lossy data compression. Computer Vision and Image Understanding, v. 110, n. 2, p. 212–225, 2008. Citado na página 27.

YANG, J.; YIN, W.; ZHANG, Y.; WANG, Y. A fast algorithm for edge-preserving variational multichannel image restoration. SIAM J. Imaging Sciences, v. 2, n. 2, p. 569–592, 2009. Citado na página 47.

ZELNIK-MANOR, L.; IRANI, M. Degeneracies, dependencies and their implications in multi-body and multi-sequence factorization. IEEE Conf. on Computer Vision and Pattern Recognition, v. 2, p. 287–293, 2003. Citado na página 27.

ZHANG, C.; BITMEAD, R. Subspace system identification for training based MIMO channel estimation. Automatica, v. 41, n. 9, p. 1623–1632, 2005. Citado na página 27.

ZHANG, H.; LIN, Z.; ZHANG, C.; GAO, J. Robust latent low rank representation for subspace clustering. Neurocomputing, v. 145, p. 369–373, 2014. ISSN 0925-2312. Citado nas páginas 21, 28, 32, 45, 46, 48 e 87.

Apêndices

APÊNDICE

A

DEMONSTRAÇÕES

Mostraremos aqui algumas das demonstrações referentes ao trabalho de Zhang et al. (2014). Para provar o Teorema 1 precisamos dos seguintes resultados:

Lema 2. Sejam U, V e M matrizes com dimensões compatíveis. Suponha U e V ortogonais, então

$$\|M\|_* = \|UMV^T\|_*$$

Prova: Tome a Decomposição em Valores Singulares (BISHOP, 2006) (SVD) de $M = U_M \Sigma_M V_M^T$, logo $UMV^T = U\,U_M \Sigma_M V_M^T\,V^T = (UU_M)\,\Sigma_M (VV_M)^T$. Como $(UU_M)^T(UU_M) = U_M^T U^T U U_M = U_M^T U_M = I$ e $(VV_M)^T(VV_M) = V_M^T V^T V V_M = V_M^T V_M = I$, assim $(UU_M)\Sigma_M(V_M V)^T$ é a decomposição SVD de UMV^T. E pela definição da norma nuclear, temos $\|M\|_* = \mathrm{tr}(\Sigma_M) = \|UMV^T\|_*$. ∎

Lema 3. Se B, C, D e F são matrizes com dimensões compatíveis, então

$$\left\| \begin{bmatrix} B & C \\ D & F \end{bmatrix} \right\|_* \geq \|B\|_*$$

onde a igualdade acontece se, e somente se, $C = 0$, $D = 0$ e $F = 0$.

Prova: Como $\|[M\ N]\|_* \geq \|M\|_*$ e $\|[M;N]\|_* \geq \|M\|_*$, o resultado segue diretamente, onde a igualdade acontece quando $N = 0$. ∎

Lema 4. Sejam U, V e M matrizes com dimensões compatíveis e suponha U e V com colunas ortogonais, então

$$\min_Z \|Z\|_* \text{ tal que } U^T Z V = M$$

tem uma solução única, $Z^* = UMV^T$.

Prova: Mostraremos inicialmente que $k\mathbf{M}k_*$ é o valor mínimo da função objetivo e $\mathbf{Z}^* = \mathbf{U}\mathbf{M}\mathbf{V}^T$ é uma solução. Para uma solução $\mathbf{Z}$, tome sua decomposição SVD $\mathbf{Z} = \mathbf{U}_Z\mathbf{\Sigma}_Z\mathbf{V}_Z^T$. Sejam $\mathbf{B} = \mathbf{U}^T\mathbf{U}_Z$ e $\mathbf{C} = \mathbf{V}_Z^T\mathbf{V}$. Assim, a restrição do problema pode ser reescrita como

$$\mathbf{U}^T\mathbf{Z}\mathbf{V} = \mathbf{U}^T\mathbf{U}_Z\mathbf{\Sigma}_Z\mathbf{V}_Z^T\mathbf{V} = \mathbf{B}\mathbf{\Sigma}_Z\mathbf{C} = \mathbf{M}$$

Além disso,

$$\mathbf{B}\mathbf{B}^T = \mathbf{U}^T\mathbf{U}_Z\mathbf{U}^T\mathbf{U}_Z^T = \mathbf{U}^T\mathbf{U}_Z\mathbf{U}_Z^T\mathbf{U} = \mathbf{U}^T\mathbf{U} = \mathbf{I}$$
$$\mathbf{C}^T\mathbf{C} = \mathbf{V}_Z^T\mathbf{V}\mathbf{V}_Z^T\mathbf{V} = \mathbf{V}^T\mathbf{V}_Z\mathbf{V}_Z^T\mathbf{V} = \mathbf{V}^T\mathbf{V} = \mathbf{I}$$

Tomando os complementos ortogonais $\mathbf{B}^\perp$ e $\mathbf{C}^\perp$, construímos as matrizes ortogonais $[\mathbf{B};\mathbf{B}^\perp]$ e $[\mathbf{C}\mathbf{C}^\perp]$ (caso $\mathbf{B}^\perp = \emptyset$ ou $\mathbf{C}^\perp = \emptyset$, a o argumento ainda é válido). Assim,

$$k\mathbf{Z}k_* = k\mathbf{\Sigma}_Z k_* = k[\mathbf{B};\mathbf{B}^\perp]\mathbf{\Sigma}_Z[\mathbf{C}\mathbf{C}^\perp]k_* = \begin{bmatrix} \mathbf{B}\mathbf{\Sigma}_Z\mathbf{C} & \mathbf{B}\mathbf{\Sigma}_Z\mathbf{C}^\perp \\ \mathbf{B}^\perp\mathbf{\Sigma}_Z\mathbf{C} & \mathbf{B}^\perp\mathbf{\Sigma}_Z\mathbf{C}^\perp \end{bmatrix}^{\#}_* \geq k\mathbf{B}\mathbf{\Sigma}_Z\mathbf{C}k_* = k\mathbf{M}k_*$$

Portanto, $k\mathbf{M}k_*$ é o valor mínimo da função objetivo. Pelo Lema 2, temos $k\mathbf{Z}^*k_* = k\mathbf{U}\mathbf{M}\mathbf{V}^Tk_* = k\mathbf{M}k_*$. Então, $\mathbf{Z}^* = \mathbf{U}\mathbf{M}\mathbf{V}^T$ é uma solução do problema.

Para mostrar que a solução $\mathbf{Z}^* = \mathbf{U}\mathbf{M}\mathbf{V}^T$ é única, suponha que $\mathbf{Z}_1 = \mathbf{U}\mathbf{M}\mathbf{V}^T + \mathbf{H}$ é outra solução. Assim, $\mathbf{U}^T\mathbf{Z}_1\mathbf{V} = \mathbf{M}$ e $\mathbf{U}^T\mathbf{H}\mathbf{V} = 0$. De modo análogo a primeira parte da demonstração, podemos construir matrizes ortogonais $[\mathbf{U};\mathbf{U}^\perp]$ e $[\mathbf{V}\mathbf{V}^\perp]$, logo

$$k\mathbf{M}k_* = k\mathbf{Z}_1k_* = k\mathbf{U}\mathbf{M}\mathbf{V}^T + \mathbf{H}k_* = k[\mathbf{U};\mathbf{U}^\perp](\mathbf{U}\mathbf{M}\mathbf{V}^T + \mathbf{H})[\mathbf{V}\mathbf{V}^\perp]k$$
$$= \begin{bmatrix} \mathbf{M} & \mathbf{U}^T\mathbf{H}\mathbf{V}^\perp \\ \mathbf{U}^{\perp T}\mathbf{H}\mathbf{V} & \mathbf{U}^{\perp T}\mathbf{H}\mathbf{V}^\perp \end{bmatrix}^{\#}_* \geq k\mathbf{M}k_*$$

Pelo Lema 3, a igualdade acima é válida se, e somente se,

$$\mathbf{U}^T\mathbf{H}\mathbf{V}^\perp = \mathbf{U}^{\perp T}\mathbf{H}\mathbf{V} = \mathbf{U}^{\perp T}\mathbf{H}\mathbf{V}^\perp = 0$$

e portanto, $\mathbf{H} = 0$, provando a unicidade da solução. ∎

Demonstremos então o Teorema 1:

Prova: Como $\mathbf{X} \in \text{span}(\mathbf{A})$, temos $\text{posto}([\mathbf{X}\mathbf{A}]) = \text{posto}(\mathbf{A})$. Calculando a decomposição SVD da matriz $[\mathbf{X}\mathbf{A}] = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^T$, chame $\mathbf{V} = [\mathbf{V}_X;\mathbf{V}_A]$, onde $\mathbf{X} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}_X^T$ e $\mathbf{A} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}_A^T$. Dessa forma, temos que $\mathbf{V}_A^T$ tem posto de linhas completo, ou seja, se denotarmos a decomposição SVD de $\mathbf{V}_A^T = \mathbf{U}_1\mathbf{\Sigma}_1\mathbf{V}_1^T$, $\mathbf{U}_1$ será uma matriz ortogonal. Dessa forma,

$$\mathbf{V}_A\mathbf{V}_A^T\mathbf{V}_A^{-1} = \mathbf{V}_1\mathbf{\Sigma}_1^T\mathbf{U}_1^T\mathbf{U}_1\mathbf{\Sigma}_1\mathbf{V}_1^T\mathbf{V}_1\mathbf{\Sigma}_1^T\mathbf{U}_1^{T-1} = \mathbf{V}_1\mathbf{\Sigma}_1^T\mathbf{U}_1^T\mathbf{U}_1\mathbf{\Sigma}_1\mathbf{\Sigma}_1^T\mathbf{U}_1^{T-1}$$
$$= \mathbf{V}_1\mathbf{\Sigma}_1^T\mathbf{U}_1^T(\mathbf{U}_1^T)^{-1}(\mathbf{\Sigma}_1^T)^{-1}\mathbf{\Sigma}_1^{-1}\mathbf{U}_1^{-1} = \mathbf{V}_1\mathbf{\Sigma}_1^{-1}\mathbf{U}_1^{-1} = \mathbf{V}_1\mathbf{\Sigma}_1^{-1}\mathbf{U}_1^T$$

Além disso,

$$X = AZ \Rightarrow U\Sigma V_X^T = U\Sigma V_A^T Z \Rightarrow V_X^T = V_A^T Z \Rightarrow V_X^T = U_1\Sigma_1 V_1^T Z$$
$$\Rightarrow \Sigma_1^{-1} U_1^T V_X^T = V_1^T Z \quad (20)$$

Assim, o problema (3.4) pode ser escrito como

$$\min_Z \|Z\|_* \quad (21) \quad \text{tal que } V_1^T Z = \Sigma_1^{-1} U_1^T V_X^T \quad (20)$$

Pelo Lema 4, o problema acima tem solução única $Z^* = V_1\Sigma_1^{-1}U_1^T V_X^T = V_A(V_A^T V_A)^{-1} V_X^T$. Como $V_A^T = \Sigma^{-1}U^T A$ e $V_X^T = \Sigma^{-1}U^T x$, temos então

$$\begin{aligned} Z^* &= A^T U\Sigma^{-1}(\Sigma^{-1}U^T AA^T U\Sigma^{-1})^{-1}\Sigma^{-1}U^T X \\ &= A^T U(U^T AA^T U)^{-1}U^T X \\ &= (U^T A)^\dagger U^T X \\ &= (\Sigma_A V_A^T)^\dagger U^T X \\ &= (U\Sigma_A V_A^T)^\dagger X \\ &= A^\dagger X \quad (8) \end{aligned}$$

Partimos agora para a demonstração do Corolário:

Prova: Como $X \in \text{span}(A)$, temos $\text{posto}(A^\dagger X) = \text{posto}(X)$. Dessa forma, $\text{posto}(Z^*) = \text{posto}(X)$. Além disso, para qualquer solução Z de (3.4), temos $\text{posto}(Z) \geq \text{posto}(AZ) = \text{posto}(X)$. Portanto, Z^* também é solução de (3.3). ∎