UNIVERSIDADE ESTADUAL DE CAMPINAS

Instituto de Matemática, Estatística e Computação Científica

ANA CAMILA RODRIGUES ALONSO

Algoritmo *branch-and-prune* para escalonamento multidimensional

Campinas

2016

Ana Camila Rodrigues Alonso

Algoritmo *branch-and-prune* para escalonamento multidimensional

Tese apresentada ao Instituto de Matemática, Estatística e Computação Científica da Universidade Estadual de Campinas como parte dos requisitos exigidos para a obtenção do título de Doutora em Matemática Aplicada.

Orientador: Aurelio Ribeiro Leite de Oliveira

Coorientador: Carlile Campos Lavor

Este exemplar corresponde à versão final da Tese defendida pela aluna Ana Camila Rodrigues Alonso e orientada pelo Prof. Dr. Aurelio Ribeiro Leite de Oliveira.

Campinas

2016

Agência(s) de fomento e nº(s) de processo(s): CNPq, 140239/2009-0

Ficha catalográfica
Universidade Estadual de Campinas
Biblioteca do Instituto de Matemática, Estatística e Computação Científica
Maria Fabiana Bezerra Muller - CRB 8/6162

AL72a

Alonso, Ana Camila Rodrigues, 1981-
Algoritmo *branch-and-prune* para escalonamento multidimensional / Ana Camila Rodrigues Alonso. – Campinas, SP : [s.n.], 2016.

Orientador: Aurelio Ribeiro Leite de Oliveira.
Coorientador: Carlile Campos Lavor.
Tese (doutorado) – Universidade Estadual de Campinas, Instituto de Matemática, Estatística e Computação Científica.

1. Escalonamento multidimensional. 2. Algoritmos branch-and-prune. 3. Análise multivariada. I. Oliveira, Aurelio Ribeiro Leite de,1962-. II. Lavor, Carlile Campos,1968-. III. Universidade Estadual de Campinas. Instituto de Matemática, Estatística e Computação Científica. IV. Título.

Informações para Biblioteca Digital

Título em outro idioma: Branch-and-prune algorithm for multidimensional scaling
Palavras-chave em inglês:
Multidimensional scaling
Branch-and-prune algorithms
Multivariate analysis
Área de concentração: Matemática Aplicada
Titulação: Doutora em Matemática Aplicada
Banca examinadora:
Aurelio Ribeiro Leite de Oliveira [Orientador]
Douglas Soares Gonçalves
Luiz Satoru Ochi
Cristiano Torezzan
João Eloir Strapasson
Data de defesa: 11-03-2016
Programa de Pós-Graduação: Matemática Aplicada

Tese de Doutorado defendida em 11 de março de 2016 e aprovada

Pela Banca Examinadora composta pelos Profs. Drs.

Prof(a). Dr(a). AURELIO RIBEIRO LEITE DE OLIVEIRA

Prof(a). Dr(a). DOUGLAS SOARES GONÇALVES

Prof(a). Dr(a). LUIZ SATORU OCHI

Prof(a). Dr(a). CRISTIANO TOREZZAN

Prof(a). Dr(a). JOÃO ELOIR STRAPASSON

A Ata da defesa com as respectivas assinaturas dos membros
encontra-se no processo de vida acadêmica do aluno.

Ao meu filho Lucas Vinícius, com todo carinho!

Agradecimentos

Ao concluir este trabalho, agradeço:

Primeiramente a Deus por mais uma etapa concluída.

Ao Prof. Dr. Aurelio Ribeiro Leite de Oliveira, pela oportunidade e honra de trabalharmos juntos. Sem a sua dedicação e competência, este trabalho não estaria concluído.

Ao Prof. Dr. Carlile Lavor Campos, por ter assumido a co-orientação desta tese, tendo-me brindado com importantes colaborações na discussão do trabalho, dosando as críticas com comentários de incentivo.

Aos membros da banca examinadora, pela disponibilidade, atenção despendida ao trabalho, correções, críticas e elogios.

Aos professores do IMECC pelos conhecimentos passados durante o curso de doutorado.

Aos meus pais Jose Carlos e Sueli, meus exemplos de vida, pelo amor, estímulo, carinho e compreensão.

Ao meu esposo Alex, pelo estímulo ao desenvolvimento desta tese, sempre paciente e generoso em meus momentos de desânimo e por participar junto nos meus sonhos.

Ao meu filho Lucas Vinícius, que ainda nem nasceu, mas já me faz querer ser melhor e dar o melhor de mim.

A minha irmã Juliana, pela inarravél oportunidade de ter convivido junto durante parte do meu doutorado, compartilhando momentos alegres e sempre me apoiando nos momentos difíceis.

Aos meus amigos queridos que de uma forma ou de outra contribuíram com sua força e estímulo para que eu conseguisse completar este percurso.

Ao CNPq que tornou possível esta pesquisa através do apoio financeiro e científico.

Finalmente, meu agradecimento a todos que direta ou indiretamente contribuíram para a realização deste trabalho: professores, funcionários, amigos e familiares.

Resumo

Neste trabalho, propomos uma nova abordagem para resolver uma classe de problemas de escalonamento multidimensional, representando medidas de proximidade entre pares de objetos como distâncias entre pontos em um espaço geométrico, de modo que as distâncias estejam o máximo possível relacionadas com as proximidades entre os objetos. Nesta classe de problemas, a prioridade foi manter a estrutura geométrica original, com o objetivo de visualizar os dados em $\mathbb{R}^3$. A proposta tem como base o algoritmo *branch-and-prune*, inicialmente utilizado para obter estruturas moleculares, a partir de algumas distâncias conhecidas. Várias adaptações, principalmente na busca e na poda, foram realizadas, destacando-se a redução da dimensão de $\mathbb{R}^m$ para $\mathbb{R}^3$. Os bons resultados computacionais obtidos, em problemas de pequeno porte, indicam um novo caminho para tratar o problema.

Palavras-chave: escalonamento multidimensional, algoritmo *branch-and-prune*, análise de coordenadas principais.

Abstract

In this work, we propose a new approach for solving a class of multidimensional scaling problems, representing proximity measures between pairs of objects as distances between points in a geometric space, such that the distances are maximally related to the proximity between the objects. In this class of problems, the priority was to maintain the original geometric structure, in order to view the data in $\mathbb{R}^3$. The proposal is based on the branch-and-prune algorithm, initially used for obtaining molecular structures, from some known distances. Several adjustments, especially in search and pruning, were made, highlighting the reduction of dimension from $\mathbb{R}^m$ for $\mathbb{R}^3$. Good computational results on small problems, indicate a new way to treat the problem.

Keywords: multidimensional scaling, branch-and-prune algorithm, principal coordinates analysis.

Lista de ilustrações

Lista de tabelas

Sumário

Introdução

O escalonamento multidimensional (*Multidimensional Scaling - MDS*), também conhecido como mapeamento percentual, é constituído por métodos que estudam conjunto de dados representado por pontos em um espaço em que as relações geométricas entre estes dados, isto é, entre estes pontos, correspondam o mais próximo possível às relações empíricas associadas aos dados (CARROLL; ARABIE, 1980; COXON, 1982).

Em síntese, o MDS é caracterizado por métodos cujo objetivo é procurar pontos em um espaço Euclidiano onde as distâncias euclidianas entre suas coordenadas sejam o mais fidedignas possíveis em relação às dissimilaridades dadas. Nesta área, o método tradicional é o Análise de Coordenadas Principais, conhecido na literatura inglesa como *Principal Coordinate Analysis, Classical Scaling, Torgerson Scaling* ou *Torgerson-Gower Scaling* (BORG; GROENEN, 2010), que tem como principal objetivo obter a visualização das dissimilaridades entre os dados.

A geometria de distâncias considera problemas que caracterizam-se por determinar as coordenadas no espaço Euclidiano k-dimensional de um conjunto de pontos, dadas algumas de suas distâncias (LAVOR et al., 2014). Estes pontos podem ser átomos, sensores em redes de telecomunicações ou vértices abstratos de um grafo, por exemplo. Em (LAVOR et al., 2012; LAVOR et al., 2008), temos o problema de encontrar as posições dos átomos, de uma dada molécula de proteína, conhecidas algumas distâncias. Para solucionar este problema foi proposto o algoritmo *branch-and-prune* (BP), com base em uma abordagem combinatória.

Branch-and-prune são algoritmos muito utilizados, por exemplo, para resolver problemas de satisfação de restrição (*Constraint Satisfaction Problem* - CSP) (KUMAR, 1992). Aplicações *branch-and-prune* utilizam técnicas de busca exaustiva baseadas em ramificação (*branch*) e poda (*prune*) de ramos que determinam soluções inviáveis.

Neste trabalho, serão apresentadas modificações no algoritmo *branch-and-prune*. Uma das modificações realizadas, neste algoritmo, foi na busca das soluções. O principal objetivo da modificação consiste em manter a estrutura original de um conjunto de pontos em $\mathbb{R}^m$ após uma redução dimensional para $\mathbb{R}^k$. Esta redução dimensional tem como foco essencial vizualizar a estrutura de um conjunto de pontos. Outra modificação no algoritmo está relacionada a poda de ramos que determinam soluções inviáveis.

O Capítulo 1 apresenta uma revisão dos conceitos básicos do escalonamento multidimensional, necessários para a compreensão do método análise de coordenadas principais apresentado no Capítulo 2. O Capítulo 3 e 4 descrevem a formulação discreta para o problema de geometria de distâncias molecular e suas modificações para adaptá-

lo ao problema proposto. Antes das conclusões, o Capítulo 5 apresenta os resultados computacionais.

1 Escalonamento Multidimensional

A técnica de escalonamento multidimensional provém de uma família de técnicas de análise de proximidade de dados e tem-se mostrado um importante instrumento matemático de mensuração. Este capítulo apresenta conceitos básicos do escalonamento multidimensional necessários para o desenvolvimento deste trabalho.

1.1 Introdução

Entre pesquisadores de diversas disciplinas é comum surgirem obstáculos para mensurar a estrutura de seu objeto de estudo, dificultando o desenvolvimento de sua pesquisa. Esta situação ocorre principalmente quando a estrutura de um objeto encontra-se implícita, tendo mais de um fator que não se manifesta claramente, mas é essencial para a interpretação dos dados (PASQUALI, 1999). Estas características geralmente estão presentes em objetos com uma estrutura multidimensional, isto é, quando mais de uma dimensão subjacente é apropriada para a interpretação dos dados. Desta forma, corre-se o risco de não se ter uma interpretação fidedigna dos dados por não alcançar todas as suas características.

Visando aumentar a precisão na interpretação dos dados que apresentam multidimensionalidade de parâmetros, a técnica de escalonamento multidimensional tem-se mostrado uma importante ferramenta de mensuração (SILVA; FILHO, 2006). Esta técnica apresenta um caráter quantitativo e computacional e é utilizada por diversos programas computacionais: *Indscal, Alscal, Minissa, Multiscale, Mdscal, Clascal, Exscal* (SCHIFFMAN; REYNOLDS; YOUNG, 1981).

1.2 Escalonamento Multidimensional

Escalonamento multidimensional é uma técnica estatística que originou-se dentro da psicologia, mais especificamente na psicometria, que é uma área que liga a matemática aplicada à psicologia. Psicometria consiste em um conjunto de técnicas utilizadas para mensurar, de forma adequada e comprovada experimentalmente, um conjunto de comportamentos que se deseja conhecer melhor. Atualmente, o escalonamento multidimensional é aplicável em diversas áreas , tais como sociologia, economia, biologia, química e arqueologia (SILVA; FILHO, 2006; SUáREZ et al., 2016).

Os métodos do escalonamento multidimensional representam medidas de pro-

ximidade (dissimilaridade ou similaridade) entre pares de objetos como distâncias entre pontos em um espaço geométrico, de maneira que as distâncias estejam o máximo possível relacionadas com as proximidades entre os objetos (BLASIUS J., 2009). Objetos similares são representados por pontos que estão próximos e objetos dissimilares são representados por pontos que estão distantes um do outro. A finalidade destes métodos consiste em representar graficamente a estrutura dos dados, visando ampliar a análise e entendimento de uma matriz de dados.

Existem diversas abordagens de escalonamento multidimensional, que distinguem-se entre si pelo tipo de geometria em que se pretende mapear os dados, pela função de mapeamento, pelos algoritmos usados para encontrar uma representação dos dados originais, pelo tratamento do erro estatístico no modelo ou a possibilidade de representar não somente uma, mas várias matrizes de similaridade ao mesmo tempo.

O escalonamento multidimensional pode apresentar quatro finalidades principais:

1. escalonamento multidimensional como um método que representa proximidades como distâncias em um espaço com baixa dimensão a fim de tornar acessível a análise visual dos dados considerados.

2. escalonamento multidimensional como uma técnica que permite testar se determinados critérios, pelos quais pode-se distinguir diferentes objetos de interesse, estão refletidos nas correspondentes diferenças empíricas desses objetos.

3. escalonamento multidimensional como uma aproximação dado-analítico que leva-nos a descobrir as dimensões subjacentes a julgamentos de dissimilaridade (similaridade).

4. escalonamento multidimensional como um modelo psicológico que explica julgamentos de dissimilaridades em termos de regras que imitam um tipo particular de função distância.

A dificuldade de definir escalonamento multidimensional está relacionada às diferentes definições apresentadas pela literatura. Em algumas, utiliza-se o termo para representar algumas técnicas específicas e outras apresentam o termo de uma forma generalizada. Desta forma, tem-se o escalonamento multidimensional no sentido amplo e no sentido restrito. Escalonamento multidimensional no sentido amplo abrange várias formas de análise de *clusters* e análise multidimensional multivariada. No sentido restrito, representa dissimilaridades entre dados em um espaço de dimensão menor (LEEUW; HEISER, 1982). Neste trabalho, adotaremos a definição menos abrangente, apresentada por Carroll e Arabie (1983) (CARROLL; ARABIE, 1980; DAVISON, 1983). Esta limita o termo escalonamento multidimensional *à uma família de modelos de distância espacial como forma de representação de dados de proximidade.*

1.3 Proximidade entre os dados e função *Stress*

As proximidades entre dados investigados reflete o grau de dissimilaridade (similaridade) entre estes, os quais serão apresentados graficamente por meio de pontos em um espaço Euclidiano. O método de escalonamento multidimensional procura uma configuração espacial dos dados, de modo que as distâncias entre estes dados estejam relacionadas com as proximidades, tanto quanto possível. Na prática, o espaço euclidiano é requerido devido à conveniência matemática nos procedimentos de escalonamento multidimensional (STEYVERS, 2002).

Assumiremos aqui que medidas de similaridades ou dissimilaridades, denotadas pelo termo geral proximidade, p_{ij}, são atribuídas a pares (i, j) de n dados. Normalmente, as proximidades entre os n dados são representadas por uma matriz de proximidade simétrica, tendo as seguintes definições:

1. uma matriz de **dissimilaridade** $(P = (p_{ij}))$ é uma matriz simétrica, em que $p_{ij} \geqslant 0$ e $p_{ii} = 0$, $\forall\ i$. A interpretação de dissimilaridade segue da propriedade de monotonicidade: (i, j) é mais dissimilar ao par (k, l) se, e somente se, $p_{ij} < p_{kl}$.

2. uma matriz de **similaridade** $(P = (p_{ij}))$ é uma matriz simétrica, em que $p_{ij} \geqslant 0$ e satisfaz $p_{ii} \geqslant p_{ij}$, $\forall\ (i, j)$. A interpretação de similaridade segue da propriedade de monotonicidade: (i, j) é mais similar ao par (k, l) se, e somente se, $p_{ij} > p_{kl}$.

A matriz de proximidade serve como dado de entrada para um método de escalonamento multidimensional.

O escalonamento multidimensional é frequentemente usado para reconstruir pontos usando somente pares de distâncias. De uma forma mais geral, o escalonamento multidimensional tenta representar proximidades p_{ij} como distâncias em um espaço m-dimensional com configuração $\mathbf{X}$. De uma maneira mais específica, é escolhida uma função $f(p_{ij})$ que especifica como a proximidade está relacionada à distância $d_{ij}(\mathbf{X})$. A função distância mais utilizada é a distância euclidiana.

A distância euclidiana entre dois pontos x_i e x_j em um espaço m-dimensional com configuração $\mathbf{X}$ é calculada pela seguinte fórmula:

$$d_{ij}(\mathbf{X}) = \| x_i - x_j \|_2 = \left(\sum_{k=1}^{m} |x_{ik} - x_{jk}|^2 \right)^{\frac{1}{2}}. \tag{1.1}$$

Portanto, o escalonamento multidimensional aplica proximidades, p_{ij}, em correspondentes distâncias $d_{ij}(\mathbf{X})$ de um espaço m-dimensional com configuração $\mathbf{X}$. Isto é,

temos uma função representada por:

$$f : p_{ij} \rightarrow d_{ij}(\mathbf{X}), \tag{1.2}$$

onde a escolha particular de f especifica o modelo de escalonamento multidimensional.

O objetivo do escalonamento multidimensional é determinar o conjunto $d_{ij}(\mathbf{X})$ a partir de p_{ij}.

1.3.1 Aproximações

Através do escalonamento multidimensional os dados de proximidade são representados numa configuração espacial de pontos de forma que as distâncias entre os pontos correpondam às proximidades tão perto quanto possível (SCHOLTEN; CALDEIRA, 1997). Logo, não se deve exigir $f(p_{ij}) = d_{ij}(\mathbf{X})$, mas sim que $f(p_{ij}) \approx d_{ij}(\mathbf{X})$.

1.3.2 *Loss Function*

A condição que determina o quão próximo $f(p_{ij})$ está de $d_{ij}(\mathbf{X})$ é conhecida como *loss function*. *Loss function* é uma expressão matemática que representa erro,

$$e_{ij} = f(p_{ij}) - d_{ij}(\mathbf{X}).$$

A norma-2 deste erro representa a função *stress*, dada por

$$e_{ij}^2 = [f(p_{ij}) - d_{ij}(\mathbf{X})]^2.$$

Somando e_{ij}^2 para todo par (i, j), tal que $i < j$, tem-se uma medida de ajuste de ineficiência, conhecida por *raw Stress* (KRUSKAL, 1964),

$$\sigma_r = \sigma_r(\mathbf{X}) = \sum_{i<j} [f(p_{ij}) - d_{ij}(\mathbf{X})]^2 . \tag{1.3}$$

Uma das formas de tentar evitar problemas de escala nos ajustes é fazendo a seguinte normalização:

$$\sigma_1^2 = \sigma_1^2(\mathbf{X}) = \frac{\sigma_r(\mathbf{X})}{\sum_{i<j} d_{ij}^2(\mathbf{X})} = \frac{\sum_{i<j} [f(p_{ij}) - d_{ij}(\mathbf{X})]^2}{\sum_{i<j} d_{ij}^2(\mathbf{X})}.$$

Tomando a raiz quadrada de σ_1^2 teremos um valor conhecido como função *Stress*-1 (KRUSKAL, 1964). A razão de usar σ_1 no lugar de σ_1^2 é que esses são quase sempre muito pequenos na prática, assim σ_1 são mais fáceis de discriminar. Mais especificamente temos,

$$\text{Stress-1} = \sigma_1 = \sqrt{\frac{\sum_{i<j}[f(p_{ij}) - d_{ij}(\mathbf{X})]^2}{\sum_{i<j} d_{ij}^2(\mathbf{X})}}.$$

Encontrar um escalonamento ótimo de $\mathbf{X}$ em um espaço m-dimensional requer minimizar a função *Stress*-1.

1.4 Propriedades gerais de representações de distâncias

Em matemática, a distância entre dois pontos é formalizada pela teoria de distâncias ou pelo conceito de métrica. Métrica é um conceito que generaliza a ideia geométrica de distância. Um espaço métrico é um conjunto X munido de uma métrica (ou distância), isto é, uma função $d : X \times X \to \mathbb{R}$ tal que para quaisquer $x, y, z \in X$,

1. $d(x, y) \geqslant 0$: não negatividade;
2. $d(x, y) = 0 \Leftrightarrow x = y$;
3. $d(x, y) = d(y, x)$: simetria;
4. $d(x, z) \leqslant d(x, y) + d(y, z)$: desigualdade triangular.

Estas propriedades são fundamentais na teoria de escalonamento multidimensional, pois as proximidades poderão ser aplicadas como distâncias se estas satisfazerem certas propriedades. Em alguns casos, temos proximidades onde $d_{ij} \neq d_{ji}$, neste caso a matriz de proximidades é não simétrica. Se $d_{ij} \neq d_{ji}$, não for consequência de erros aderentes aos dados, não poderemos representar diretamente estas proximidades em um espaço métrico. Portanto, simetria é uma pré-condição para escalonamento multidimensional. As outras propriedades de distâncias podem ou não ser condições necessárias para EMD. Se as proximidades violarem a restrição da desigualdade triangular, elas podem ou não ser representadas como distâncias. No escalonamento multidimensional ordinal isto não é um problema pois é possível adicionar uma constante a d_{ij}, eliminando toda violação. Por outro lado, no escalonamento multidimensional raio, as proximidades assumem ter uma origem fixa, logo qualquer violação da desigualdade triangular acarreta sérios problemas (BORG; GROENEN, 2010).

Além da métrica euclidiana, uma outra métrica muito utilizada dentro do escalonamento multidimensional é a métrica retangular que pode ser definida da seguinte maneira. Dados dois pontos $x_i, x_j \in \mathbb{R}^n$, temos

$$d(x_i, x_j) = \sum_{k=1}^{m} |x_{ik} - x_{jk}| = \| x_i - x_j \|_1.$$

A métrica retangular é conhecida informalmente como métrica de Manhattan ou métrica metropolitana, devido ao delineamento da rede urbana de transportes da Ilha de Manhattan na cidade de New York.

Na próxima seção, apresentaremos um método do escalonamento multidimensional que utiliza fortemente o conceito de distância, tendo como objetivo principal preservar distâncias em mudança de espaços.

1.5 Escalonamento Multidimensional Clássico

O escalonamento multidimensional clássico, também conhecido por *Torgerson scaling* e *Torgerson-Gower scaling*, foi o primeiro método prático disponível para escalonamento multidimensional. Desenvolvido por Torgerson (TORGERSON, 1952; TORGERSON, 1958) e Grower (GOWER, 1966), este método é baseado nos trabalhos de Eckart-Young (ECKART; YOUNG, 1936) e Young-Householder (YOUNG; HOUSEHOLDER, 1938).

A ideia do escalonamento multidimensional clássico consiste em assumir que as proximidades são distâncias, sendo apenas necessário encontrar as coordenadas que preservam estas distâncias. Este fato é aceito para dados derivados de matrizes de correlação, mas raramente para dissimilaridades diretas. Este método tornou-se popular, pois fornece uma solução analítica, não sendo necessário utilizar métodos iterativos (BORG; GROENEN, 2010).

Se analisarmos o problema de encontrar as distâncias entre algumas cidades, dadas as coordenadas, podemos solucioná-lo através da distância euclidiana. Podemos solucionar o problema inverso utilizando o escalonamento multidimensional clássico.

A seguir, apresentaremos os passos para solucionar este problema através do escalonamento multidimensional clássico.

Etapas de um algoritmo de EMD clássico

O método clássico (BORG; GROENEN, 2010; TORGERSON, 1952) baseia-se no fato que a matriz de coordenadas X pode ser obtida da decomposição espectral da matriz $B = XX^T$. O problema de construir B a partir de uma matriz de proximidade P é

solucionado multiplicando $P \circ P = [p_{ij}^2]$ tanto pelo lado direito quanto pelo lado esquerdo pela matriz $J = I - n^{-1}ee^T$, onde e é um vetor de uns, com $e \in \mathbb{R}^n$ e n é o número de dados (BORG; GROENEN, 2010). Esta operação é conhecida por *double centering*. As etapas a seguir resumem o algoritmo que determina a matriz de coordenadas X (BORG; GROENEN, 2010):

1. Define-se a matriz de proximidades ao quadrado: $P \circ P = [p_{ij}]^2$, onde o símbolo $\circ$ representa o produto de Hadamard entre duas matrizes, definido como $C \circ D \equiv [c_{ij}d_{ij}]$, isto é, como o produto de elementos correspondentes das duas matrizes (necessariamente de dimensões iguais).

2. Aplique o *double centering* a esta matriz:

$$B = -\frac{1}{2}J(P \circ P)J.$$

3. Calcule a decomposição espectral de $B = V\Lambda V^T$, onde V é a matriz dos autovetores e Λ é a matriz diagonal dos autovalores (GOLUB; LOAN, 2013) .

4. Seja Λ_+ a matriz diagonal dos m autovalores maiores que zero em ordem decrescente, e V_+ os m autovetores de B relacionados aos autovalores de Λ_+. Então, a matriz das coordenadas do escalonamento multidimensional clássico é dada por $X = V_+\Lambda_+^{\frac{1}{2}}$.

Este método será descrito detalhadamente no Capítulo 2, pois apresenta características que melhor se aproximam ao algoritmo apresentado neste trabalho.

Na próxima seção, apresentaremos uma aplicação do escalonamento multidimensional.

1.6 Exemplo de Aplicação

A análise exploratória de dados é usada para estudar teoricamente dados amorfos, ou seja, dados que não estão ligados a uma teoria explícita que prevê suas grandezas ou padrões (BORG; GROENEN, 2010).

No livro *Modern Multidimensional Scaling: Theory and Applications* (BORG; GROENEN, 2010), encontramos o seguinte exemplo: um resumo estatístico de 1970, emitido pelo *Bureau of the Census*, que fornece dados sobre a taxa de diferentes crimes em 50 estados dos E.U.A. (Wilkinson, 1990). Uma pergunta que foi feita sobre estes dados é até que ponto se pode prever uma elevada taxa de assassinato sabendo que a taxa de roubo é alta. Uma resposta parcial a esta pergunta é fornecida calculando as correlações das taxas de crimes em 50 estados dos E.U.A., descritas pela Tabela 1.1.

Crime	No.	1	2	3	4	5	6	7
Assassinato	1	1,00	0,52	0,34	0,81	0,28	0,06	0,11
Estupro	2	0,52	1,00	0,55	0,70	0,68	0,60	0,44
Latrocínio	3	0,34	0,55	1,00	0,56	0,62	0,44	0,62
Assalto	4	0,81	0,70	0,56	1,00	0,52	0,32	0,33
Roubo	5	0,28	0,68	0,62	0,52	1,00	0,80	0,70
Furto	6	0,06	0,60	0,44	0,32	0,80	1,00	0,55
Roubo de Carros	7	0,11	0,44	0,62	0,33	0,70	0,55	1,00

Tabela 1 – Correlações entre índices de criminalidade em 50 estados dos EUA.

Em geral, não é uma tarefa simples entender a estrutura dos coeficientes apenas com a matriz de correlação. Isto se torna mais simples representando as correlações na forma de figura, como podemos ver na Figura 1, a qual foi obtida através do método de escalonamento multidimensional.

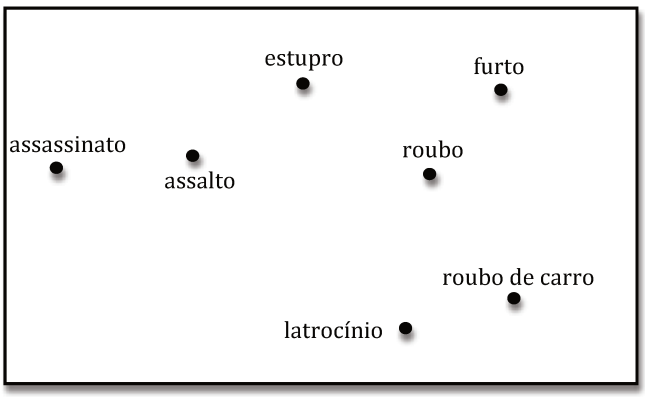

Figura 1 – Representação em um espaço com duas dimensões das correlações apresentadas na tabela anterior.

A Figura 1 é uma representação em um plano das correlações, onde cada crime é mostrado como um ponto. Os pontos são arranjados de tal modo que as suas distâncias correspondem às correlações. Isto é, dois pontos permanecem próximos (tal como assassinato e assalto) se suas taxas relacionadas são altas. Da mesma forma, dois pontos estão distantes se suas taxas apresentam uma baixa relação (tal como assalto e furto).

No próximo capítulo apresentaremos, o método escalonamento multidimensional clássico, conhecido como análise de coordenadas principais que será utilizada como método de comparação para a nova abordagem a ser apresentada.

2 Análise de Coordenadas Principais

Neste capítulo, a técnica clássica análise de coordenadas principais, ou escalonamento multidimensional clássico, é apresentada em detalhe.

2.1 Introdução

Um método tradicional de escalonamento multidimensional (EMD) é a Análise de Coordenadas Principais ou, na literatura inglesa, *Principal Coordinate Analysis*, *Torgerson Classical Scaling* ou *Torgerson Gower Scaling*. Como mencionado, o desenvolvimento da técnica foi baseada nos trabalhos de Eckart-Young (ECKART; YOUNG, 1936) e de Young-Householder (YOUNG; HOUSEHOLDER, 1938). A seguir apresentaremos os resultados destes trabalhos os quais serão explorados no decorrer do capítulo.

Teorema 2.1. *Para cada matriz $A \in \mathbb{R}^{m\times n}$ de posto r, existem matrizes ortogonais $U_{m\times m}$, $V_{n\times n}$ e uma matriz diagonal $D_{r\times r} = diag(\sigma_1, \sigma_2, \cdots, \sigma_r)$ tal que*

$$A = U \begin{pmatrix} D & 0 \\ 0 & 0 \end{pmatrix} V^T \text{ com } \sigma_1 \geqslant \sigma_2 \geqslant \cdots \geqslant \sigma_r > 0.$$

Os σ_i's são chamados valores singulares não nulos de A. Esta fatoração é chamada decomposição em valores singulares de A, e as colunas de U e V são chamadas vetores singulares à direita e à esquerda de A, respectivamente.

Teorema 2.2. *Seja a decomposição em valores singulares (SVD) de $A \in \mathbb{R}^{m\times n}$, dada por $A = U\Sigma V^T$, em que $U \in \mathbb{R}^{m\times m}$ e $V \in \mathbb{R}^{n\times n}$ são matrizes ortogonais e $\Sigma \in \mathbb{R}^{m\times n}$ é uma matriz diagonal com $posto(A) = r$. Se $k < r$ e $A_k = \sum_{i=1}^{k} \sigma_i u_i v_i^T$, então*

$$\min_{posto(B)=k} \| A - B \|_2 = \| A - A_k \|_2 = \sigma_{k+1}. \tag{2.1}$$

A análise de coordenadas principais (PCoA) é uma generalização da análise de componentes principais (PCA), sendo sua principal vantagem não necessitar dos dados originais, mas apenas da matriz de dissimilaridade (MANLY, 1994; JONGMAN; BRAAK; TONGEREN, 1995).

Análise de Coordenadas Principais é uma técnica padrão para análise exploratória e de visualização das dissimilaridades ou similaridades dos dados. Sua importância está relacionada ao fato de ser um método simples e não-paramétrico que extrai informações relevantes de conjunto de dados extensos e nebulosos. A extração de informações relevantes ou compressão de dados é realizada através da redução dimensional.

Dada uma matriz de dissimilaridade $D = [\delta_{ij}]$ entre n indivíduos, procura-se uma representação dos n indivíduos em um espaço Euclidiano de tal forma que as distâncias euclidianas d_{ij} entre cada par dos n pontos nessa representação sejam iguais, ou o mais fidedignas possíveis, em relação às dissimilaridades δ_{ij} da matriz inicial.

A ideia do método consiste em assumir que as dissimilaridades iniciais foram calculadas com base nas distâncias euclidianas. A matriz de dissimilaridades, neste caso, será representada por pontos em um espaço de dimensão menor definido pelas primeiras componentes principais. No entanto, nem sempre será possível obter uma representação euclidiana dos indivíduos, estando esta possibilidade dependente da natureza e propriedades das medidas de dissimilaridades utilizadas. A utilidade deste método reside na sua ampla aplicabilidade, por exemplo, no caso de apenas a matriz de distâncias estar definida, não conhecendo-se a matriz de dados originais.

2.2 Matriz de Distâncias Euclidianas

Um espaço Euclidiano $\mathbb{R}^m$ é um espaço vetorial real de dimensão finita munido de um produto interno, induzindo uma métrica (KREYSZIG, 1989). Uma matriz de distâncias euclidianas é uma matriz $n \times n$ de distâncias d_{ij} entre n pontos $x_1, x_2, ..., x_n$ em $\mathbb{R}^m$ em que

$$d_{ij}^2 = \| x_i - x_j \|_2^2 = (x_i - x_j)^t (x_i - x_j),\ \forall i, j.$$

Definição 2.1. *Seja D uma matriz $n \times n$ de dissimilaridade entre n indivíduos. A matriz D diz-se uma matriz Euclidiana se existirem n pontos $\{x_i\}_{i=1}^n \in \mathbb{R}^m$ tais que*

$$\delta_{ij}^2 = \| x_i - x_j \|_2^2 .$$

Uma matriz de dissimilaridade $D = [\delta_{ij}]$, $i, j \in \{1, ..., n\}$, é uma matriz Euclidiana se for possível determinar n pontos em $\mathbb{R}^m$, com $m \in \mathbb{N}$, onde as distâncias entre qualquer par desses n pontos seja precisamente a dissimilaridade correspondente na matriz D.

Considerando, por conveniência, que os n pontos $\{x_i\}_{i=1}^n \in \mathbb{R}^m$ formam a matriz $X_{n \times m}$ tal que $x_i^t = e_i^t X$. Assim, a diferença entre o i-ésimo ponto e o j-ésimo ponto é dada por:

$$(x_i - x_j)^t = (e_i - e_j)^t X \tag{2.2}$$

e a dissimilaridade ao quadrado pode ser escrita da seguinte forma

$$d_{ij}^2 = (x_i - x_j)^t (x_i - x_j) = (e_i - e_j)^t X X^t (e_i - e_j). \tag{2.3}$$

Quando uma matriz é euclidiana, existe um número infinito de maneiras de representar os n pontos em um espaço Euclidiano de dimensão k, ou seja, a matriz X não é única. Todas as representações estão relacionadas a transformações isométricas: rotação, reflexão e translação (DATTORRO, 2005).

Em particular, será sempre possível considerar uma solução cujo centróide dos pontos desta solução seja a origem do referencial em $\mathbb{R}^k$. Isso corresponde a dizer que é sempre possível determinar uma matriz X que tem as colunas centradas, sendo por isso da forma

$$X = (I_n - \frac{1}{n}ee^t)X.$$

A matriz $n \times n$, definida por

$$I_n - \frac{1}{n}ee^t,$$

em que e é um vetor de n uns, é denominada matriz de centragem. A matriz de centragem é uma matriz simétrica e idempotente, que, quando multiplicada por um vetor tem o mesmo efeito que a subtração da média dos componentes do vetor de cada componente (MARDEN, 1995).

Definição 2.2. *A matriz de centragem de tamanho n é definida como uma matriz $n \times n$:*

$$C_n = I_n - \frac{1}{n}ee^t.$$

Outra forma de apresentar a mesma ideia corresponde a notar que os vetores $e_i - e_j$ introduzidos na equação (2.2) pertencem ao complemento ortogonal do subespaço de $\mathbb{R}^m$ gerado pelo vetor e de n uns, uma vez que a soma das suas coordenadas é nula. Ou seja, $e_i - e_j \in span\{e\}^{\perp}$, $\forall\ i, j = 1, ..., n$. Logo, esses vetores permanecem invariantes quando projetados sobre esse subespaço,

$$(I_n - \frac{1}{n}ee^t)(e_i - e_j) = e_i - e_j.$$

Assim, é possível re-escrever a equação (2.3) da seguinte forma:

$$d_{ij}^2 = (e_i - e_j)^T(I_n - \frac{1}{n}ee^t)XX^T(I_n - \frac{1}{n}ee^t)(e_i - e_j), \tag{2.4}$$

sendo $(I_n - \frac{1}{n}ee^t)X$ a matriz X com colunas centradas cujas linhas contêm as coordenadas dos n pontos com centro geométrico na origem.

Na próxima seção, abordaremos duas questões interligadas: (i) saber quando uma dada matriz A de dissimilaridade é Euclidiana; e (ii) saber como determinar uma matriz X de coordenadas para os n pontos, com as características acima descritas.

2.3 Coordenadas de uma Matriz de Dissimilaridade Euclidiana

O método Análise de Coordenadas Principais assume que as dissimilaridades iniciais são calculadas com base nas distâncias euclidianas.

Denominando a matriz dos produtos internos entre os vetores (centrados) x_i por Q, tem-se:

$$Q = (I_n - \frac{1}{n}ee^t)XX^t(I_n - \frac{1}{n}ee^t). \tag{2.5}$$

A partir da equação (2.4), denominando o elemento da linha i e coluna j de Q por q_{ij}, temos,

$$d_{ij}^2 = (e_i - e_j)^T Q(e_i - e_j),$$

$$d_{ij}^2 = e_i^t Q e_i - e_i^t Q e_j - e_j^t Q e_i + e_j^t Q e_j,$$

$$d_{ij}^2 = q_{ii} - 2q_{ij} + q_{jj} \tag{2.6}$$

pois Q é uma matriz simétrica.

Com base na equação (2.6) concluímos que, se apenas conhecermos a matriz de dissimilaridade (distância), é possível determinar a matriz Q.

Pelo fato de as colunas de X estarem centradas em torno da sua média, terão que ser nulas:

1. todas as somas das colunas de Q;
2. todas as somas das linhas de Q;
3. a soma de todos os elementos de Q.

Ou seja,

$$\sum_{i=1}^{n} q_{ij} = 0, \quad \forall \ j,$$
$$\sum_{j=1}^{n} q_{ij} = 0, \quad \forall \ i,$$
$$\sum_{i=1}^{n}\sum_{j=1}^{n} q_{ij} = 0.$$

A partir da equação (2.6), temos:

1. $\sum_{i=1}^{n} d_{ij}^2 = \sum_{i=1}^{n} q_{ii} + nq_{jj}$,
2. $\sum_{j=1}^{n} d_{ij}^2 = nq_{ii} + \sum_{j=1}^{n} q_{jj}$,
3. $\sum_{i=1}^{n}\sum_{j=1}^{n} d_{ij}^2 = 2n\sum_{k=1}^{n} q_{kk}$.

Equivalente à:

1. $tr(Q) = \frac{1}{2n} \parallel Q \parallel_F^2$,
2. $q_{jj} = \frac{1}{n}\sum_{i=1}^{n} d_{ij}^2 - \frac{1}{2n^2}\sum_{i=1}^{n}\sum_{j=1}^{n} d_{ij}^2$,
3. $q_{ii} = \frac{1}{n}\sum_{j=1}^{n} d_{ij}^2 - \frac{1}{2n^2}\sum_{i=1}^{n}\sum_{j=1}^{n} d_{ij}^2$.

Novamente, a partir da equação (2.6) tem-se:

$$q_{ij} = -\frac{1}{2}(d_{ij}^2 - q_{ii} - q_{jj}),$$

e substituindo q_{jj} por

$$\frac{1}{n}\sum_{i=1}^{n} d_{ij}^2 - \frac{1}{2n^2}\sum_{i=1}^{n}\sum_{j=1}^{n} d_{ij}^2,$$

e q_{ii} por

$$\frac{1}{n}\sum_{j=1}^{n} d_{ij}^2 - \frac{1}{2n^2}\sum_{i=1}^{n}\sum_{j=1}^{n} d_{ij}^2,$$

tem-se:

$$q_{ij} = -\frac{1}{2}\left(d_{ij}^2 - \sum_{j=1}^{n} d_{ij}^2 - \sum_{i=1}^{n} d_{ij}^2 + \frac{1}{2n}\sum_{i=1}^{n}\sum_{j=1}^{n} d_{ij}^2\right).$$

Logo, conhecendo-se apenas a matriz de dissimilaridade D, é possível recuperar a matriz Q dos produtos internos. Assim, é possível encontrar as coordenadas que correspondem à solução do problema através da matriz Q. De fato, considere a Decomposição Espectral (GOLUB; LOAN, 2013) da matriz simétrica Q:

$$Q_{n\times n} = (I_n - \frac{1}{n}ee^t)XX^T(I_n - \frac{1}{n}ee^t) = W\Lambda_{n\times n}W^t.$$

Desde que a matriz Q seja semi-definida positiva, os elementos da diagonal de Λ são não negativos e é possível definir a matriz $\Lambda^{\frac{1}{2}}$. A matriz $W_+\Lambda_+^{\frac{1}{2}}$ é uma matriz cujas n linhas podem representar os n indivíduos em $\mathbb{R}^k$, de forma a serem respeitados os produtos internos, e daí as distâncias euclidianas d_{ij} entre eles (KSHIRSAGAR, 1972).

Portanto, o método encontra uma solução se Q for uma matriz semi-definida positiva. Se a matriz de dissimilaridades D é euclidiana, então Q é semi-definida positiva. De fato,

$$Qe = (I_n - \frac{1}{n}ee^t)XX^t(I_n - \frac{1}{n}ee^t)e = (I_n - \frac{1}{n}ee^t)XX^t(e - e) = 0,$$

isto é,

$$Qe = 0 \cdot e, \tag{2.7}$$

isto garante que Q não pode ser uma matriz definida positiva, pois como podemos ver na equação (2.7), a matriz Q admite um autovalor nulo.

A solução encontrada $W\Lambda^{\frac{1}{2}}$ não necessariamente corresponde às coordenadas da matriz de dados iniciais, ou seja, estas soluções tem em comum a matriz de distâncias euclidianas entre os n indivíduos observados em $\mathbb{R}^k$. Os dados iniciais podem ser determinados através de rotações da nova solução $W\Lambda^{\frac{1}{2}}$. Ou seja, para qualquer matriz de colunas ortogonais R, tem-se:

$$(W\Lambda^{\frac{1}{2}}R^t)(W\Lambda^{\frac{1}{2}}R^t)^t = W\Lambda^{\frac{1}{2}}(R^tR)\Lambda^{\frac{1}{2}}W^t = W\Lambda W^t = Q. \tag{2.8}$$

Esta indeterminação na reconstituição da matriz de dados originais X, a partir da matriz de distâncias, corresponde a dizer que as distâncias fixam a configuração dos pontos, a menos de rotação e/ou reflexões em torno da origem X.

A ideia do método de análise de coordenadas principais é rotacionar os eixos transformando-os em eixos principais. Eixos principais são as dimensões de um sistema particular de coordenadas ortogonais (BORG; GROENEN, 2010). O primeiro eixo está próximo de todos os pontos da configuração X; o segundo eixo mantém a proximidade dos pontos com a restrição de ser ortogonal ao primeiro e, assim, sucessivamente.

Uma matriz de rotação é uma matriz quadrada que, quando multiplicada por um vetor tem o efeito de mudar a direção do vetor, mas não a sua magnitude. Este vetor desloca-se em torno de um eixo de rotação definido pelos elementos desta matriz. O resultado é um segundo vetor resultante da rotação. A situação física, ver Figura 2,

associada pode ser compreendida como uma mudança de referencial estabelecida entre dois sistemas de coordenadas com origens comuns, mas que tenham seus eixos coordenados não coincidentes (via diferenças providas por uma relação em torno do mesmo eixo de rotação) e pelo mesmo valor angular, antes associados à rotação do vetor (GOLDSTEIN, 1922).

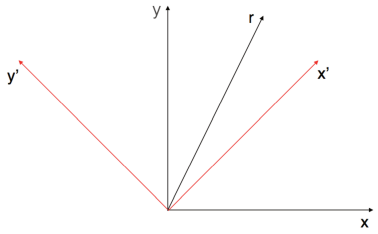

Figura 2 – Interpretação da matriz de rotação como determinante de uma mudança de referencial.

Pelo Teorema 2.2 de Eckart-Young (ECKART; YOUNG, 1936), sabemos que a melhor representação dos n pontos será dada pelas primeiras colunas da matriz $W\Lambda^{\frac{1}{2}}$. A qualidade desta representação pode ser medida pelo quociente

$$\frac{\sum_{i=1}^{k} |\lambda_i|}{\sum_{j=1}^{n} |\lambda_j|},$$

onde λ_i são os autovalores da matriz Λ.

Uma das expectativas ao se fazer uma análise de coordenadas principais é de que as variâncias da maioria dos índices sejam tão baixas a ponto de serem desprezíveis, permitindo que as representações em dimensões menores sejam o mais fidedignas possíveis aos dados iniciais. Neste caso, a maior parte da variação no conjunto de dados iniciais pode ser descrita adequadamente pelas k primeiras variáveis.

2.4 Coordenadas de uma Matriz de Dissimilaridade

A matriz de dissimilaridade resulta dos dados inicias depois de aplicar o método análise de coordenadas principais. Em alguns casos, não se sabe se a matriz de dissimilaridade é ou não uma matriz euclidiana. Abordaremos este problema nesta seção.

Vamos admitir inicialmente que a matriz de dissimilaridade D é simétrica, com elementos não negativos e diagonal nula.

Dada uma matriz $D = [\delta_{ij}]$ de dissimilaridade simétrica, as etapas para determinar a matriz Q serão as mesmas apresentadas na seção anterior:

1. Cria-se, a partir da matriz de dissimilaridade D, uma nova matriz que se designará $B = [b_{ij}]$:

$$b_{ij} = -\frac{1}{2}\delta_{ij}^2.$$

Em termos matriciais,

$$B = -\frac{1}{2}(D \circ D).$$

2. Constrói-se a matriz Q, obtida a partir da centragem das linhas e das colunas da matriz B, isto é, a matriz Q terá os seguintes elementos:

$$q_{ij} = b_{ij} - \sum_{j=1}^{n} b_{ij} - \sum_{i=1}^{n} b_{ij} + \sum_{i=1}^{n}\sum_{j=1}^{n} b_{ij},$$

onde $\sum_{j=1}^{n} b_{ij}$, $\sum_{i=1}^{n} b_{ij}$ e $\sum_{i=1}^{n}\sum_{j=1}^{n} b_{ij}$ são, respectivamente, as médias dos elementos da linha i, dos elementos da coluna j, e da totalidade dos elementos da matriz B. Em termos matriciais, tem-se:

$$Q = -\frac{1}{2}(I_n - \frac{1}{n}ee^t)B(I_n - \frac{1}{n}ee^t). \tag{2.9}$$

3. Se a matriz D for simétrica então as matrizes B e Q também serão simétricas e, assim, Q admite uma deomposição espectral

$$Q = W\Lambda W^t.$$

4. Caso a matriz Q seja semi-definida positiva, procede-se como descrito anteriormente, determinando se a matriz

$$Y = W\Lambda^{\frac{1}{2}},$$

e as n linhas desta matriz correspondem às coordenadas que representam cada ponto em um espaço Euclidiano n-dimensional.

5. Novamente pelo Teorema 2.2, a nuvem de pontos obtida sobre $\mathbb{R}^k$ (em geral $k = 2$ ou 3) é construída retendo-se apenas as k primeiras colunas da matriz Y, isto é, através das linhas da matriz:

$$Y_k = W_k\Lambda_k^{\frac{1}{2}},$$

onde W_k é a matriz $n \times k$ obtida retendo apenas as k primeiras colunas da matriz de autovetores de Q, e $\Lambda_k^{\frac{1}{2}}$ é a matriz $k \times k$ obtida retendo apenas as raízes quadradas dos k primeiros autovalores de Λ , isto é, retendo apenas as k primeiras linhas e colunas da matriz $\Lambda^{\frac{1}{2}}$.

6. Se Q não for semi-definida positiva, alguns dos seus autovalores serão negativos. Essa situação corresponde a dizer que não existe representação exata em um espaço Euclidiano real (ver Teorema 2.4), isto é, que não é possível garantir a representação dos dados em um espaço $\mathbb{R}^m$ de forma a respeitar as igualdades $d_{ij} = \delta_{ij}$ entre dissimilaridades iniciais e distâncias euclidianas no espaço $\mathbb{R}^m$. Ou ainda, se Q não for semi-definida positiva, então nenhuma fatoração da forma $Q = R^t R$ é possível, não havendo, assim, linhas de R que possam representar Euclidianamente os dados.

Seja $D_{n\times n}$ uma matriz simétrica, então:

i) D é definida positiva se, e somente se, existir uma matriz $X \in \mathbb{R}^{m\times n}$ de $posto(X) = n$ tal que $D = X^T X$.

ii) D é semi-definida positiva com $posto(D) = k < n$ se, e somente se, existir uma matriz $X \in \mathbb{R}^{m\times n}$ de $posto(X) = k$ tal que $D = X^T X$.

Uma demonstração deste teorema pode ser encontrada em (GOLUB; LOAN, 2013).

Portanto, uma matriz de dissimilaridade D é Euclidiana no espaço $\mathbb{R}^k$ se, e somente se, a matriz $Q = (I_n - \frac{1}{n}ee^t)B(I_n - \frac{1}{n}ee^t)$, com $B = -\frac{1}{2}(D \circ D)$, é semi-definida positiva com posto menor ou igual a k.

2.5 Matriz de Dissimilaridade não Euclidianas

Algumas matrizes de dissimilaridade não são Euclidiana. Neste caso, ao trabalhar com o método análise de coordenadas principais cuja matriz Q apresenta autovalores negativos, duas alternativas podem ser consideradas:

1. Caso os autovalores negativos de Q sejam pequenos em magnitude, ou seja, são pequenos em relação à soma dos autovalores positivos, pode-se ignorá-los. Neste caso, trabalha-se com uma configuração Euclidiana aproximada, resultante de considerar apenas os autovetores associados aos autovalores positivos de Q. Em particular, pode-se escolher o número máximo de eixos coordenados principais, de acordo com as seguintes regras:

Critério de traço: Reter os eixos cuja soma de autovalores associados seja aproximadamente igual ao traço da matriz Q.

Critério do valor absoluto: Reter eixos cujo autovalores associados sejam maiores do que o módulo do menor autovalor negativo.

Como na prática o objetivo maior é obter uma visualização gráfica da representação Euclidiana, opta-se por escolher $k = 2$ ou $k = 3$, ou seja, os 2 ou 3 primeiros eixos principais. Uma interpretação utilizada é que os autovalores de cada eixo correspondem proporcionalmente a variabilidade dos dados apresentados. Dois critérios específicos, propostos na literatura para medir a qualidade da representação obtida, utilizam os k eixos coordenados principais cujos autovalores associados se admite serem todos positivos, ou seja,

$$P_1 = \frac{\sum_{i=1}^{k} \lambda_i}{\sum_{i=1}^{n} |\lambda_i|},$$

$$P_2 = \frac{\sum_{i=1}^{k} \lambda_i^2}{\sum_{i=1}^{n} \lambda_i^2},$$

onde λ_i são os autovalores da matriz Λ, ver (2.8).

2. O **problema da constante aditiva** (CAILLIEZ, 1983) consiste em somar uma constante, c^*, a todos elementos de uma matriz (exceto a diagonal) de dissimilaridade, com o objetivo de tornar a matriz Q uma matriz semi-definida positiva possibilitando assim uma representação Euclidiana dos dados. A solução do problema (CAILLIEZ, 1983) consiste em tomar c^* igual ao maior autovalor da matriz

$$\begin{bmatrix} 0_n & 2Q \\ -I_n & -4Q(d_{ij}) \end{bmatrix},$$

onde $Q(d_{ij})$ corresponde à matriz obtida pela dupla centragem de uma matriz análoga à matriz B, mas com os elementos dados por $-\frac{1}{2}\delta_{ij}$, em vez de $-\frac{1}{2}\delta_{ij}^2$.

2.6 Outras Técnicas Visando Representações Euclidianas

Dentro do princípio de obter uma representação euclidiana a partir de uma matriz de dissimilaridade D entre n indivíduos, pode-se formular o problema de uma forma

diferente. Nesta formulação alternativa, procuram-se pontos em um espaço euclidiano de uma dada dimensão onde as dissimilaridades δ_{ij} não correspondem às distâncias euclidianas, mas a algum critério de qualidade no ajuste.

Estas técnicas de *Multidimensional Scaling (MDS)* podem ser sintetizadas nos seguintes passos:

1. Fixa-se a dimensão k do espaço Euclidiano onde se deseja a representação.

2. Parte-se de uma configuração inicial de n indivíduos nesse espaço, isto é, determina-se uma matriz $n \times k$ cujas linhas são as coordenadas de cada indivíduo. Uma escolha possível de configuração inicial pode ser a solução k-dimensional produzida pela análise de coordenadas principais.

3. Calculam-se as distâncias euclidianas usuais entre os n pontos da configuração proposta.

4. Calcula-se o valor (para esta configuração) de algum critério de ajuste que se deseja otimizar.

5. Efetuam-se operações à configuração (de acordo com algum conjunto de regras pré-especificadas) e calcula-se o novo valor do critério.

6. Repete-se o passo anterior até alguma condição de parada (usualmente ligada à ideia de que alterações na configuração não produzem melhorias no valor do critério).

   O critério de qualidade mais frequente e a função **Stress**, apresentada na equação (1.3). Este ajuste proposto por Kruskal (KRUSKAL, 1964) e Shepard (SHEPARD, 1962) apresenta algumas variações :

$$Stress = \sqrt{\frac{\sum_i \sum_{j<i} (e_{ij} - f(\delta_{ij}))^2}{\sum_i \sum_{j<i} e_{ij}^2}},$$

   onde δ_{ij} representa a dissimilaridade inicial entre os indivíduos i e j, e_{ij} representa a distância euclidiana habitual entre os representantes desses indivíduos na configuração que está sendo proposta e f indica alguma função crescente. Escolhendo f como sendo a função identidade, vemos um critério cujo valor mínimo (zero) corresponderia a um conjunto de n pontos ideais, em que as distâncias euclidianas entre os pontos i e j são sempre iguais às dissimilaridades δ_{ij} dadas na matriz D. Permitir que f seja uma outra função crescente é uma opção indicada para casos em que os valores dessas dissimilaridades são subjetivos, sendo mais importante o posto da matriz do que os valores destas dissimilaridades.

2.7 Exemplo Numérico

O exemplo abaixo ilustra uma aplicação do escalonamento multidimensional para medidas de dissimilaridade, e mostra como construir uma configuração que permite visualizar suas dissimiliaridades. Este exemplo foi tirado do site *www.mathworks.com* (MATHWORKS,) que mostra uma aplicação do método análise de coordenadas principais utilizando como ferramenta o *Matlab*.

Distância genética é uma medida de dissimilaridade de material genético entre diferentes espécies ou indivíduos, (NEI, 1978). Seja

$$D = \begin{bmatrix} 0 & 4,69 & 6,79 & 3,50 & 3,11 & 4,46 & 5,57 & 3,00 \\ 4,69 & 0 & 2,10 & 2,27 & 2,65 & 2,36 & 1,99 & 1,74 \\ 6,79 & 2,10 & 0 & 3,78 & 4,53 & 2,83 & 2,44 & 3,79 \\ 3,50 & 2,27 & 3,78 & 0 & 1,98 & 4,35 & 2,07 & 0,53 \\ 3,11 & 2,65 & 4,53 & 1,98 & 0 & 3,80 & 3,31 & 1,47 \\ 4,46 & 2,36 & 2,83 & 4,35 & 3,80 & 0 & 4,35 & 3,82 \\ 5,57 & 1,99 & 2,44 & 2,07 & 3,31 & 4,35 & 0 & 2,57 \\ 3,00 & 1,74 & 3,79 & 0,53 & 1,47 & 3,82 & 2,57 & 0 \end{bmatrix}, \tag{2.10}$$

uma matriz com as distâncias genéticas, ou dissimilaridades, entre uma série de subpopulações locais de uma única espécie animal e seja

$$X = \begin{bmatrix} 39,1 & 18,7 \\ 40,7 & 21,2 \\ 41,5 & 21,5 \\ 39,2 & 21,8 \\ 38,7 & 20,6 \\ 41,7 & 20,1 \\ 40,1 & 22,1 \\ 39,2 & 21,6 \end{bmatrix}, \tag{2.11}$$

a matriz das localizações geográficas.

Gostaríamos de saber a proximidade entre as distâncias genéticas e espaciais. A proximidade entre estas duas distâncias implica que o cruzamento entre as subpopulações é afetado por suas localizações geométricas.

O comando do *Matlab* que efetua uma análise de coordenadas principais é o comando `cmdscale`,

```
Y = cmdscale(D).
```

Dada uma matriz de distâncias D o comando `cmdscale` retorna uma matriz $n \times k$, onde k ($k < n$) é a dimensão do menor espaço no qual os n pontos estarão configurados. A ideia principal é que esses pontos estejam em duas ou três dimensões e as distâncias euclidianas entre eles reproduza a matriz D. Assim, um gráfico de dispersão de pontos criado pelo comando `cmdscale` fornece uma representação visual das distâncias originais. Como a matriz X é formada por elementos em $\mathbb{R}^2$, será procurada uma representação bi-dimensional e serão consideradas as coordenadas dos pontos nos dois primeiros eixos coordenados principais. Como estão sendo consideradas 8 espécies, os pontos (linhas de `Y`) podem ter até oito dimensões (colunas de `Y`). Esta dimensionalidade pode ser alterada para uma k dimensão, desde que $k < n$. Os autovalores de $\mathtt{YY}^t$ podem ser determinados pelo comando

```
[Y,E] = cmdscale(D),
```

cuja magnitude relativa pode indicar quantas dimensões podem ser usadas com segurança. Se mais de três autovalores forem relativamente grandes em sua magnitude, a solução pode não permitir uma boa configuração em um espaço de baixa dimensão, ou seja, em um espaço que permite a visualização das distâncias:

```
[Y,eigvals] = cmdscale(D),
```

$$\mathtt{Y} = \begin{bmatrix} 3,6513 & -0,7919 & 0,4620 & 0,1537 & -0,0424 \\ -0,9787 & -0,2062 & -0,0675 & -0,4166 & -0,3435 \\ -3,1073 & -0,3814 & 0,0901 & -0,0502 & 0,2658 \\ 0,5731 & 1,4088 & 0,4371 & 0,0230 & 0,2742 \\ 1,2325 & 0,3946 & -1,2765 & 0,0954 & 0,0834 \\ -0,7021 & -2,7500 & 0,0371 & 0,1786 & 0,0080 \\ -1,4586 & 1,5821 & 0,1669 & 0,5082 & -0,2597 \\ 0,7900 & 0,7439 & 0,1508 & -0,4922 & 0,0141 \end{bmatrix}, \mathtt{eigvals} = \begin{bmatrix} 29,0371 \\ 13,5746 \\ 2,0987 \\ 0,7418 \\ 0,3403 \\ 0,0000 \\ -0,4542 \\ -3,1755 \end{bmatrix}. \tag{2.12}$$

Observe que há apenas dois autovalores positivos com magnitude elevada permitindo uma configuração em 2 dimensões. Os dois autovalores negativos refletem, como se viu anteriormente, a impossibilidade de os 8 pontos respeitarem, em um espaço de dimensão 8, as distâncias genéticas indicadas na matriz de dissimilaridade inicial. Como

os autovalores negativos são pequenos em relação ao autovalores positivos, a redução para as duas primeiras colunas de `Y` devem ser bastante precisa. Pode-se verificar isto através do erro relativo das distâncias genéticas em 2 dimensões e das distâncias originais:

`maxrelerr = max(abs(D - squareform(pdist(Y(:,1:2)))))/max(D),`

`maxrelerr = 0.1078.`

A configuração retornada por `cmdscale` é única, ou seja, o método é apenas capaz de recuperar posições relativas, a menos de rotações rígidas, reflexões em torno dos eixos e translações da origem. É provável que a localização genética não coincida com a localização geográfica e não esteja na mesma escala. Para solucionar este tipo de problema utiliza-se o comando `D = procrustes(X, Y)`.

A função procrustes analisa a distribuição de um conjunto de configurações através da análise de procrustes, que consiste de uma análise estatística utilizada para analisar a distribuição de um conjunto de configurações (GOWER; DIJKSTERHUIS, 2004). O nome procrustes refere-se a um bandido da mitologia grega que fez suas vítimas caberem em suas camas ou esticando seus membros ou cortando-os. Na matemática, o problema procrustes ortogonal é um problema de aproximação de matriz que procura encontrar rotações e/ou reflexões ótimas para a sobreposição de um conjunto de dados em relação a outro. O solução do problema de procrustes transforma uma dada matriz A em uma matriz B através de uma matriz de transformação ortogonal T tal que a soma dos quadrados dos erros $E = AT - B$ seja mínimo. Este problema é equivalente a encontrar a matriz ortogonal mais próxima dada pela matriz $S = A^T B$. Para encontrar a matriz E utiliza-se decomposição em valores singulares,

$$S = U\Sigma V^T,$$

que leva a $E = UV^T$. Um problema de procrustes restrito, sujeito à $det(R) = 1$, em que R é uma matriz de rotação, é um método que pode ser utilizado para determinar a rotação ótima (GOWER; DIJKSTERHUIS, 2004).

O comando `D = procrustes(X, Y)`, determina uma transformação linear (translação, rotação ortogonal, reflexão e escalar) dos pontos na matriz `Y` para melhor adequar aos pontos da matriz X. O critério de ajuste é a soma dos quadrados dos erros. O comando `D = procrustes(X, Y)` retorna o valor minimizado destas medidas de dissimilaridades em D:

`[D,Z] = procrustes(X,Y(:,1:2)).`

A Figura 3 mostra os pontos após aplicar o método procrustes. Aparentemente, as distâncias genéticas têm uma estreita ligação com as distâncias espaciais entre as subpopulações.

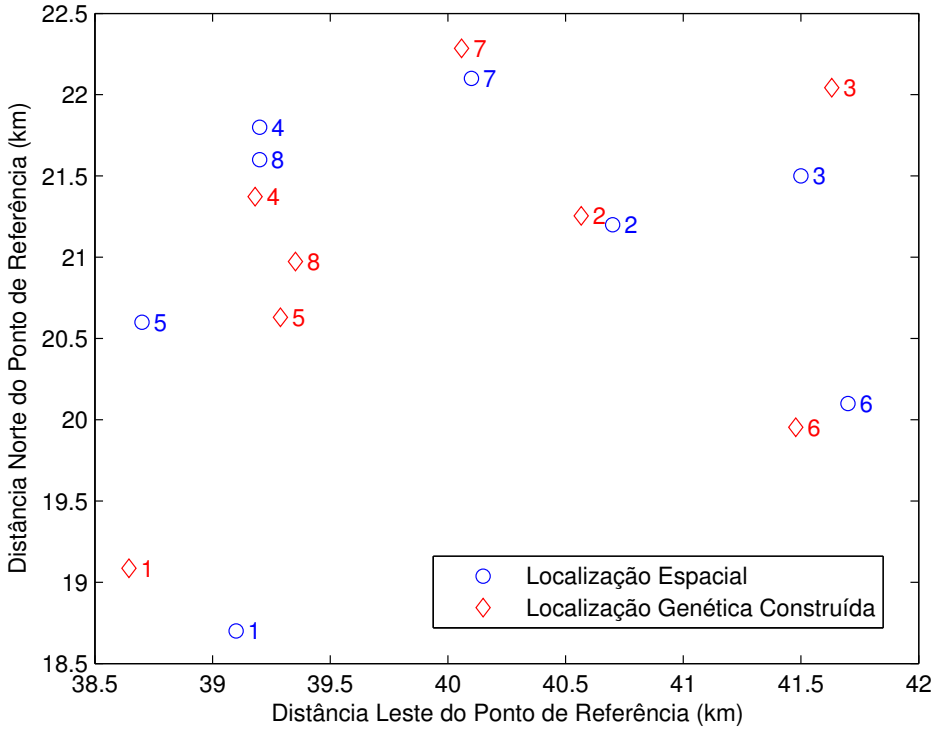

Figura 3 – Comparação entre as localizações genética através do `cmdscale` e geográfica.

No próximo capítulo, abordaremos a metodologia aplicada ao problema de determinar a estrutura tridimensional de proteínas. Neste mesmo capítulo, apresentaremos as semelhanças entre o problema discreto de distâncias moleculares e o problema de manter uma configuração após reduzir a dimensão do espaço. Apresentaremos também modificações realizadas na resolução do problema discreto de distâncias moleculares para esta ser aplicada ao problema de escalonamento multidimensional. Estas modificações têm como principal finalidade manter o máximo possível as distâncias de uma dada matriz de dissimilaridade ao reduzir a dimensão do espaço original da estrutura.

3 Problema Discreto de Escalonamento Multidimensional

Neste capítulo veremos o problema discreto de distâncias moleculares e apresentaremos uma adaptação deste problema para o contexto de redução dimensional. A seguir, apresentaremos algumas definições e notações úteis ao decorrer do capítulo.

3.1 Definições e notações

Um grafo orientado é um par $G = (V, E)$ onde V é um conjunto finito de vértices e E é um conjunto de pares ordenados de vértices, chamados de arestas. Uma aresta de G é denotada por (u, v) e diz-se que u é o início da aresta e v o final da aresta (AHUJA; MAGNANTI; ORLIN, 1993).

Um grafo é dito ponderado se está associado a ele uma função peso $\omega : E \to \mathbb{R}$ que associa arestas a valores reais. Em um grafo G, uma sequência de vértices $P = \langle v_0, v_1, ..., v_k \rangle$ é um caminho de v_0 a v_k. O peso de um caminho $P = \langle v_0, v_1, ..., v_k \rangle$ é a soma dos pesos das suas arestas e é denotado por $\omega(P)$. A distância de u a v em G, denotado por $dist(u, v)$, é o peso mínimo de um caminho de u a v em G. Se o caminho não existir, define-se $dist(u, v) = \infty$.

3.2 Problemas de Geometria de Distâncias

Problemas de Geometria de Distâncias (PGD) caracterizam-se por determinar as coordenadas de n pontos no espaço Euclidiano k-dimensional dadas algumas distâncias entre pares de pontos. Estes pontos podem ser átomos, sensores em redes de telecomunicações ou vértices abstratos de um grafo entre outros. Em problemas de geometria molecular, as distâncias são conhecidas por técnicas experimentais como, por exemplo, ressonância magnética nuclear (LAVOR et al., 2011).

Em (LAVOR et al., 2012; LAVOR et al., 2008), estuda-se o problema de encontrar as posições dos átomos de uma dada molécula de proteína, conhecidas algumas distâncias. Este problema apresenta a seguinte formulação:

PROBLEMA DE DISTÂNCIAS MOLECULARES (PDM): dado um grafo não-direcionado simples $G = (V, E, d)$ com $d : E \to \mathbb{R}_+$, existe uma aplicação $x : V \to \mathbb{R}^3$ tal que $\|x_u - x_v\|_2 = d_{uv}$, para cada $\{u, v\} \in E$?

O conjunto V representa os átomos em $\mathbb{R}^3$ e o conjunto E são os pares de átomos $\{u, v\}$ cujas distâncias d_{uv} são conhecidas. Este problema é resolvido em tempo linear quando as distâncias entre os átomos forem conhecidas (DONG Q., 2002). O problema de distâncias moleculares é formulado como um problema de otimização não convexa contínuo:

$$ming(x),$$

que que

$$g(x) = \sum_{\{u,v\} \in E} (||x_u - x_v||^2 - d_{uv}^2)^2.$$

Claramente, x é solução se, e somente se $g(x) = 0$.

Devido a estrutura particular de algumas cadeias de átomos, é possível reduzir o espaço de busca a um conjunto discreto e finito de pontos. Esta particularidade possibilitou formular o problema discreto de distâncias moleculares (PDDM), o qual consiste de um subconjunto de instâncias do problema de distâncias moleculares (PDM). O problema de busca discreta causou impacto na velocidade e na precisão da solução, devido aos cálculos em aritmética de ponto flutuante serem menores do que os métodos de pesquisa contínua (LAVOR et al., 2011; LAVOR et al., 2012).

O problema discreto de distâncias moleculares, apresentado em (LAVOR et al., 2011), contém a seguinte formulação:

PROBLEMA DISCRETO DE DISTÂNCIAS MOLECULARES (PDDM): dado um grafo não-direcionado simples $G = (V, E, d)$, tal que existe uma ordem em V satisfazendo:

1. E contém todos os cliques de quatro vértices consecutivos:
   $\forall\ i \in \{4, ..., n\}\ \forall\ j, k \in \{i - 3, ..., i\}\ (\{j, k\} \in E)$;
2. A seguinte desigualdade triangular estrita vale:
   $\forall\ i \in \{2, ..., n - 1\},\ d_{i-1,i+1} < d_{i-1,i} + d_{i,i+1}$,

existe uma aplicação $x : V \to \mathbb{R}^3$ tal que $\|x_u - x_v\|_2 = d_{uv}$ para cada $\{u, v\} \in E$?

O conjunto V representa os átomos em $\mathbb{R}^3$ e o conjunto E são os pares de átomos $\{u, v\}$ cujas distâncias d_{uv} são conhecidas. No problema discreto de distâncias

moleculares, $d_{i-1,i}$ é a distância euclidianas entre os pontos $i-1$ e i, para todo $i = 2, ..., n$, o ângulo $\theta_{i-2,i} \in [0, \pi]$ representa o ângulo formado entre os segmentos definidos pelos pontos $i-2$, $i-1$ e i, para todo $i = 3, ..., n$ e o ângulo de torção $\omega_i \in [0, 2\pi]$ representa o ângulo entre as retas normais dos planos definidos pelos pontos $i-3, i-2, i-1$ e $i-2, i-1, i$, para todo $i = 4, ..., n$ [ver Figura 4]. Neste problema, a ordem em V é definida por uma cadeia linear de átomos conectados entre si por ligações covalentes.

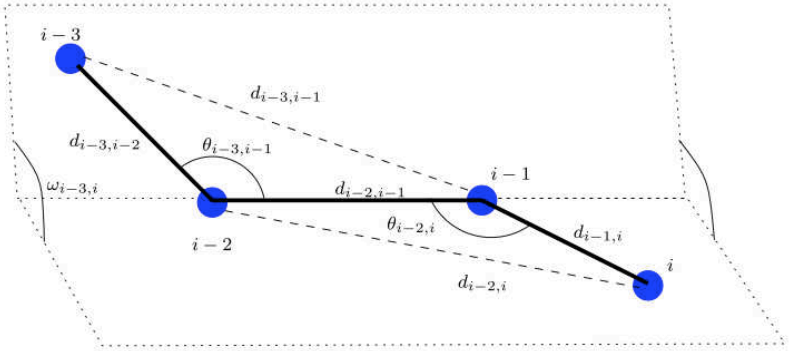

Figura 4 – Definindo distâncias, ângulos e ângulos de torção.

O problema de distâncias moleculares pode ser solucionado como um problema discreto de distâncias moleculares, caso seja possível encontrar uma ordem entre os átomos que satisfaça as suposições 1 e 2. Com base nas ideias do problema discreto de distâncias moleculares, formulamos o problema denominado problema discreto de distâncias (PDD) ou problema discreto de escalonamento multidimensional. Neste problema, temos como dados de entrada as distâncias entre n pontos em $\mathbb{R}^m$, $m \in \mathbb{N}$. A seguir, apresentaremos os conceitos básicos do problema discreto de distâncias moleculares, e em seguida mostraremos as adaptações para o problema de escalonamento multidimensional.

3.3 Problema Discreto de Distâncias Moleculares

A intuição geométrica da formulação discreta é que o i-ésimo átomo está na intersecção de três esferas centradas nos átomos $i-3$, $i-2$ e $i-1$ e com os respectivos raios $d_{i-3,i}$, $d_{i-2,i}$ e $d_{i-1,i}$. Pelas suposições 1 e 2 do PDDM, a intersecção das três esferas definem no máximo 2 posições possíveis para o átomo (denominadas i e i' na Figura 5).

No caso em que uma subsequência de três átomos consecutivos i, $i+1$ e $i+2$, onde o ângulo θ_{i+2} é $k\pi$ para $k \in \mathbb{Z}$, a formulação discreta não é possível. Em outras palavras, se o ângulo θ_{i+2} é múltiplo de π, temos a situação apresentada na Figura 6, onde $d_{i,i+3}$ é viável para todas as posições de $i+3$ no círculo mostrado no desenho. Uma vez que o conjunto $\{\pi\}$ tem medida 0 em $[0, 2\pi]$, a probabilidade de qualquer instância ser discretizável é 1. Este caso costuma ocorrer frequentemente na prática (LAVOR et al., 2008).

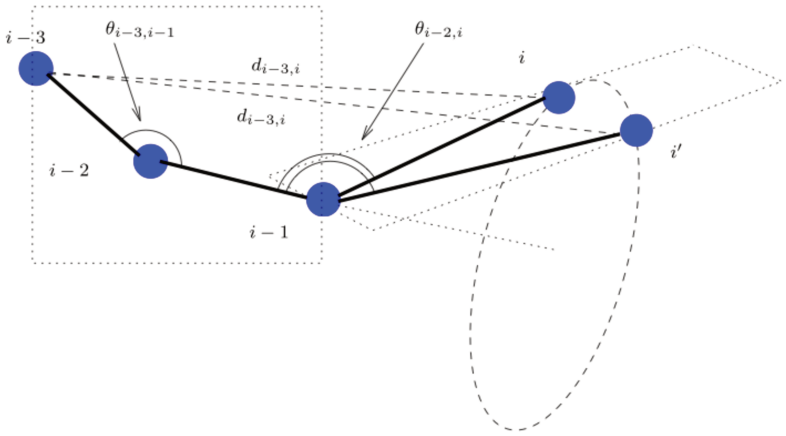

Figura 5 – Problema Discreto: possíveis posições para o i-ésimo ponto.

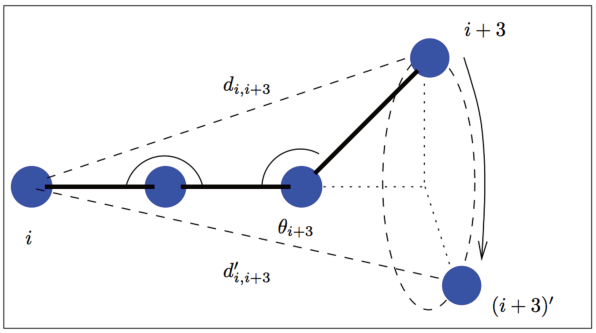

Figura 6 – Uma instância que não pode ser discretizada.

Dadas as distâncias $d_{1,2}, ..., d_{n-1,n}$, os ângulos $\theta_{1,3}, ..., \theta_{n-2,n}$ e os ângulos de torção $\omega_{1,4}, ..., \omega_{n-3,n}$, as coordenadas Cartesianas $x_i = (x_{i1}, x_{i2}, x_{i3})$, para cada ponto i, podem ser obtidas. Os ângulos de torção são fornecidos pelas retas normais através dos planos definidos pelos átomos i, $i+1$, $i+2$ e $i+1$, $i+2$, $i+3$, para $i = 1, ..., n-3$. Descreveremos a seguir como as coordenadas são calculadas.

3.3.1 Cálculo das Coordenadas

O cálculo das coordenadas é proveniente de um procedimento clássico desenvolvido por Eyring (EYRING, 1932). As coordenadas Cartesianas (x_{i1}, x_{i2}, x_{i3}) para cada átomo i na molécula podem ser obtidas por:

$$\begin{bmatrix} x_{i1} \\ x_{i2} \\ x_{i3} \\ 1 \end{bmatrix} = B_1 B_2 ... B_i \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \end{bmatrix}, \forall\, i \in \{1, \ldots, n\},$$

onde B_1 é uma matriz identidade de dimensão 4,

$$B_2 = \begin{bmatrix} -1 & 0 & 0 & -d_{1,2} \\ 0 & 1 & 0 & 0 \\ 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}, \tag{3.1}$$

$$B_3 = \begin{bmatrix} -cos\theta_{1,3} & -sen\theta_{1,3} & 0 & -d_{2,3}cos\theta_{1,3} \\ sen\theta_{1,3} & -cos\theta_{1,3} & 0 & d_{2,3}sen\theta_{1,3} \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}, \tag{3.2}$$

$$B_i = \begin{bmatrix} -cos\theta_{i-2,i} & -sen\theta_{i-2,i} & 0 & -d_{i-1,i}cos\theta_{i-2,i} \\ sen\theta_{i-2,i}cos\omega_{i-3,i} & -cos\theta_{i-2,i}cos\omega_{i-3,i} & -sen\omega_{i-3,i} & d_{i-1,i}sen\theta_{i-2,i}cos\omega_{i-3,i} \\ sen\theta_{i-2,i}sen\omega_{i-3,i} & -cos\theta_{i-2,i}sen\omega_{i-3,i} & cos\omega_{i-3,i} & d_{i-1,i}sen\theta_{i-2,i}sen\omega_{i-3,i} \\ 0 & 0 & 0 & 1 \end{bmatrix}, \tag{3.3}$$

para $i \in \{4, ..., n\}$. Então, as coordenadas Cartesianas de todos os átomos na molécula são determinados por $cos\omega_{i-3,i}$ e $sen\omega_{i-3,i}$ para $i \in \{4, ..., n\}$.

A cada três átomos consecutivos $x_i, x_{i+1}, x_{i+2}, x_{i+3}$, podemos obter o cosseno do ângulo de torção ω_{i+3} em termos das distâncias $d_{i,i+1}, d_{i+1,i+3}, d_{i,i+3}$ e ângulos θ_{i+2}, θ_{i+3}. De acordo com Pogorelov (POGORELOV, 1987), a lei do cosseno para um triedro indica uma relação entre o cosseno e o seno dos ângulos formados por seus vetores e o cosseno do ângulo de torção [ver Figura 7].

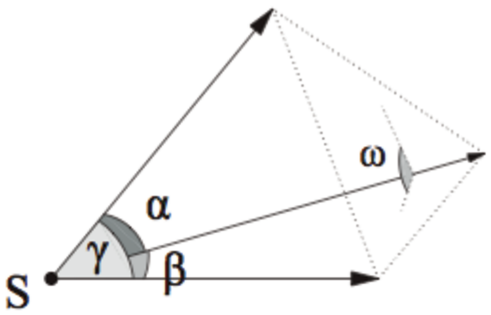

Figura 7 – Ângulos e ângulo diedro de um triedro.

Sejam α, β e γ ângulos dos planos e ω o ângulo diedral oposto a γ. A lei do cosseno para um triedral é dada por

$$cos\gamma = cos\alpha cos\beta + sen\alpha sen\beta cos\omega.$$

Visto que α, $\beta \in [0, \pi]$, podemos tomar $sen\alpha = \sqrt{1 - cos^2\alpha}$ e $sen\beta = \sqrt{1 - cos^2\beta}$. Além disto, se α e β são diferentes de 0 e π, podemos obter a seguinte expressão:

$$cos\gamma = \frac{cos\alpha - cos\alpha cos\beta}{\sqrt{1 - cos^2\alpha}\sqrt{1 - cos^2\beta}}.$$

Considere quatro átomos consecutivos com posições dadas por x_i, x_{i+1}, x_{i+2}, $x_{i+3} \in \mathbb{R}^3$. A fim de estudar o ângulo diedral dado pelas retas normais através dos planos definidos pelas posições x_i, x_{i+1}, x_{i+2} e x_{i+1}, x_{i+2}, x_{i+3}, podemos escolher o átomo x_{i+1} sendo o vértice comum de S. Um triedro é definido pelos vetores $a = x_i - x_{i+1}$, $b = x_{i+2} - x_{i+1}$, $c = x_{i+3} - x_{i+1}$, respectivamente opostos aos ângulos α, β e γ. Usando distâncias entre as coordenadas x_i, x_{i+1}, x_{i+2}, x_{i+3} obtemos:

$$cos\alpha = \frac{d^2_{i,i+1} + d^2_{i+1,i+2} - d^2_{i,i+2}}{2d_{i,i+1}d_{i+1,i+2}},$$

$$cos\beta = \frac{d^2_{i+1,i+3} + d^2_{i+1,i+2} - d^2_{i+2,i+3}}{2d_{i+1,i+3}d_{i+1,i+2}},$$

$$cos\gamma = \frac{d^2_{i,i+1} + d^2_{i+1,i+3} - d^2_{i,i+3}}{2d_{i,i+1}d_{i+1,i+3}}.$$

Usando estes valores, podemos obter o cosseno de um ângulo de torção em termos de distâncias entre átomos,

$$\begin{aligned} \cos\omega &= \frac{cos\gamma - cos\alpha cos\beta}{sen\alpha sen\beta} \\ &= \frac{\frac{d^2_{i,i+1}+d^2_{i+1,i+3}-d^2_{i,i+3}}{2d_{i,i+1}d_{i+1,i+3}} - cos\alpha cos\beta}{sen\alpha sen\beta} \\ &= \frac{d^2_{i,i+1} + d^2_{i+1,i+3} - 2d_{i,i+1}d_{i+1,i+3}cos\alpha cos\beta - d^2_{i,i+3}}{2d_{i,i+1}d_{i+1,i+3}sen\alpha sen\beta}. \end{aligned} \tag{3.4}$$

Se substituirmos α por θ_{i+2}, β por θ_{i+3} e ω por ω_{i+3} teremos, ver Figura 8,

$$\cos\omega_{i,i+3} = \frac{d^2_{i,i+1} + d^2_{i+1,i+3} - 2d_{i,i+1}d_{i+1,i+3}cos\theta_{i,i+2}cos\theta_{i+1,i+3} - d^2_{i,i+3}}{2d_{i,i+1}d_{i+1,i+3}sen\theta_{i,i+2}sen\theta_{i+1,i+3}}. \tag{3.5}$$

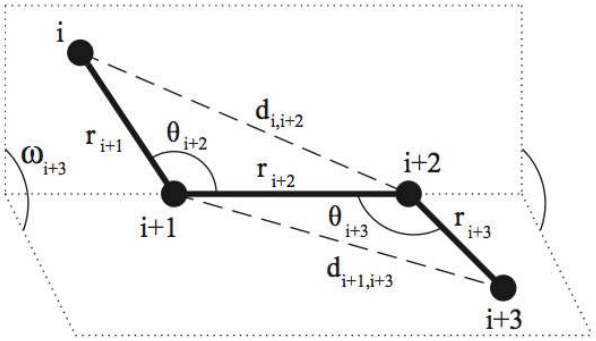

Figura 8 – Distâncias, ângulos e ângulos de torção.

Fixando o primeiro átomo na origem, é possível determinar as posições dos três primeiros átomos:

$$x_1 = \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix}, x_2 = \begin{pmatrix} -d_{1,2} \\ 0 \\ 0 \end{pmatrix},$$

e

$$x_3 = \begin{pmatrix} d_{2,3}cos\theta_{1,3} - d_{1,2} \\ d_{2,3}sen\theta_{1,3} \\ 0 \end{pmatrix}.$$

O *seno* do ângulo de torção $\omega_{1,4}$ pode assumir apenas dois valores:

$$\text{sen}\omega_{1,4} = \pm\sqrt{1 - (\cos\omega_{1,4})^2},$$

uma vez que a distância entre x_1 e x_4 também é conhecida (LAVOR et al., 2008). Consequentemente, existem duas posições possíveis, x_4 e $x^{'}{}_4$, para o quarto átomo.

$$x_4 = \begin{bmatrix} -d_{1,2} + d_{2,3}cos\theta_{1,3} - d_{3,4}cos\theta_{1,3}cos\theta_{2,4} + d_{3,4}sen\theta_{1,3}sen\theta_{2,4}cos\omega_{1,4} \\ d_{2,3}sen\theta_{1,3} - d_{3,4}sen\theta_{1,3}cos\theta_{2,4} - d_{3,4}cos\theta_{1,3}sen\theta_{2,4}cos\omega_{1,4} \\ -d_{3,4}sen\theta_{2,4}(\sqrt{1 - (cos\omega_{1,4})^2}) \end{bmatrix}$$

e

$$x^{'}_4 = \begin{bmatrix} -d_{1,2} + d_{2,3}cos\theta_{1,3} - d_{3,4}cos\theta_{1,3}cos\theta_{2,4} + d_{3,4}sen\theta_{1,3}sen\theta_{2,4}cos\omega_{1,4} \\ d_{2,3}sen\theta_{1,3} - d_{3,4}sen\theta_{1,3}cos\theta_{2,4} - d_{3,4}cos\theta_{1,3}sen\theta_{2,4}cos\omega_{1,4} \\ -d_{3,4}sen\theta_{2,4}(-\sqrt{1 - (cos\omega_{1,4})^2}) \end{bmatrix}.$$

Na próxima seção será apresentada uma adaptação para o problema de escalonamento multidimensional mantendo as ideias principais do problema discreto de distâncias moleculares.

3.4 Formulação do Problema de Escalonamento Multidimensional

No problema de escalonamento multidimensional, vamos considerar um conjunto de n pontos em $\mathbb{R}^m$, $m \in \mathbb{N}$. Definiremos em $\mathbb{R}^3$ uma representação desses n pontos que mantém as mesmas distâncias dadas em $\mathbb{R}^m$ entre os pontos $i-1$ e i, para $i \in \{2, ..., n\}$, $i-2$ e i, para $i \in \{3, ..., n\}$ e também entre os pontos $i-3$ e i, para $i \in \{4, ..., n\}$. Neste caso, todas as distâncias são conhecidas *a priori* e satisfazem as suposições 1 e 2 do PDDM. Com essa representação em $\mathbb{R}^m$, e utilizando as distâncias conhecidas entre os pontos, aplicamos o algoritmo branch-and-prune modificado, que será apresentado no próximo capítulo. O algoritmo tem como objetivo obter uma representação em $\mathbb{R}^3$ dos n pontos tentando preservar as distâncias dadas.

O primeiro passo do algoritmo branch-and-prune modificado é verificar se a matriz de dissimilaridades entre os pontos é uma matriz de distâncias euclidianas.

3.4.1 Matrizes de Distâncias Euclidianas

De acordo com as propriedades de norma, todas as entradas de uma matriz de distâncias euclidianas tem que estar de acordo com as propriedades da métrica euclidiana, ou seja, a matriz D apresenta as seguintes propriedades:

- $d_{ij} \geqslant 0$ para todo i, j;
- $d_{ii} = 0$ para todo i;
- $d_{ij} = d_{ji}$ para todo i, j (matriz simétrica);
- $d_{ij} \leqslant d_{ik} + d_{kj}$ (desigualdade triangular).

A matriz D tem n^2 entradas, mas somente $n(n-1)/2$ informações e qualquer rotação, translação ou reflexão dos n pontos $\{x_k,\ k \in \{1, ..., n\}\}$ em $\mathbb{R}^m$, produz a mesma matriz D (DATTORRO, 2005).

Após garantir que a matriz D é uma matriz de distâncias Euclidiana, o próximo passo é determinar as coordenadas dos n pontos em $\mathbb{R}^k$. Na próxima seção, apresentaremos a abordagem utilizada para os cálculos das coordenadas através de intersecção das esferas.

3.4.2 Pontos de Intersecção de n esferas em $\mathbb{R}^n$

O problema de determinar os pontos de intersecção de n esferas em $\mathbb{R}^n$ tem muitas aplicações. Um exemplo é o problema de determinar as posições em $\mathbb{R}^3$ dos átomos de uma molécula, dadas algumas distâncias entre estes (LAVOR et al., 2008). Neste problema, o método utilizado para encontrar os pontos de intersecção é restrito a pontos em $\mathbb{R}^3$.

De acordo com Coope (COOPE, 2000), se as coordenadas de n pontos em $\mathbb{R}^n$ são conhecidas e queremos determinar as coordenadas de pontos cujas distâncias aos pontos dados são conhecidas, este problema é equivalente a encontrar a intersecção de n esferas em $\mathbb{R}^n$. Um caso simples é tomarmos 3 pontos em $\mathbb{R}^3$ cujas distâncias a um quarto ponto são conhecidas, este quarto ponto é determinado encontrando a intersecção das 3 esferas. Para n pontos em $\mathbb{R}^n$, o problema é formulado como um sistema de n equações não lineares.

Seja $\mathbf{a}_j \in \mathbb{R}^n$, $j \in \{1, ..., n\}$, o centro das n esferas e d_j, $j \in \{1, ..., n\}$ os correspondentes raios. O problema de intersecção de n esferas consiste em encontrar $\mathbf{x} \in \mathbb{R}^n$ que satisfaz as n equações não-lineares:

$$\| \mathbf{x} - \mathbf{a}_j \|_2^2 = d_j^2, \quad j \in \{1, ..., n\}, \tag{3.6}$$

ou equivalentemente,

$$\mathbf{x}^T\mathbf{x} - 2\mathbf{x}^T\mathbf{a}_j + \mathbf{a}_j^T\mathbf{a}_j = d_j^2, \quad j \in \{1, ..., n\}. \tag{3.7}$$

Em algumas casos, como apresentado aqui, os raios das esferas correspondem às distâncias. As suposições 1 e 2 do PDDM sendo satisfeitas, permite que o problema seja solucionado via eliminação Gaussiana e via decomposição ortogonal.

3.4.2.1 Solução via Eliminação Gaussiana

Seja $\mathbf{A} \in \mathbb{R}^{n \times n}$ uma matriz onde as colunas são os vetores $\mathbf{a}_j$, $j \in \{1, ..., n\}$. Se os vetores $\{\mathbf{a}_j\}_{j=1}^n$ são linearmente independentes, então a matriz $\mathbf{A}$ é não singular e a abordagem a seguir fornece uma técnica de solução simples. Temos,

$$\mathbf{a}_j^T\mathbf{x} = (\mathbf{x}^T\mathbf{x} + \mathbf{a}_j^T\mathbf{a}_j - d_j^2)/2, \tag{3.8}$$

para $j \in \{1, 2, ..., n\}$. Primeiro, reescreveremos as equações não-lineares (3.8) como

$$\mathbf{a}_j^T\mathbf{x} = (r + b_j)/2, \tag{3.9}$$

para $j \in \{1, 2, ..., n\}$, onde $r = \mathbf{x}^T\mathbf{x}$ e $b_j = \mathbf{a}_j^T\mathbf{a}_j - d_j^2$, $j \in \{1, 2, ..., n\}$. Na forma matricial, a equação (3.9) torna-se $\mathbf{A}^T x = (re + b)/2$ ou

$$\mathbf{x} = (r\mathbf{u} + \mathbf{v})/2, \tag{3.10}$$

onde

$$\mathbf{u} = \mathbf{A}^{-T}e, \ \mathbf{v} = \mathbf{A}^{-T}\mathbf{b}. \tag{3.11}$$

Daí segue que, $r = \mathbf{x}^T\mathbf{x} = \frac{1}{4}(r\mathbf{u} + \mathbf{v})^T(r\mathbf{u} + \mathbf{v})$ ou ainda,

$$(\mathbf{u}^T\mathbf{u})r^2 + (2\mathbf{u}^T\mathbf{v} - 4)r + \mathbf{v}^T\mathbf{v} = 0, \tag{3.12}$$

que é uma equação quadrática escalar em r. Resolvendo temos

$$r = \frac{2 - \mathbf{u}^T\mathbf{v} \pm \sqrt{(2 - \mathbf{u}^T\mathbf{v})^2 - (\mathbf{u}^T\mathbf{u})(\mathbf{v}^T\mathbf{v})}}{\mathbf{u}^T\mathbf{u}}, \tag{3.13}$$

e as duas soluções para x podem então ser recuperadas usando a equação (3.10).

A abordagem acima é eficiente, pois exige a solução de apenas dois sistemas lineares (3.11) de ordem n com a mesma matriz sendo possível uma decomposição LU. Infelizmente, esta abordagem está restrita a matrizes cujas colunas são linearmente independentes.

Em alguns casos, a origem poderá ser um dos n pontos, resultando em uma coluna nula da matriz $\mathbf{A}$. Esta é uma das situações que $\mathbf{A}$ será singular. De acordo com (COOPE, 2000), na maioria das aplicações o $posto(\mathbf{A}) \geqslant n - 1$ pois, os pontos $\mathbf{a}_j$, $j \in \{1, ..., n\}$, serão afimente independentes. Deste modo, uma possível solução quando $posto(\mathbf{A}) = n - 1$ é transladar a origem tornando $\mathbf{A}$ não singular. A seguir, apresentamos uma abordagem para solucionar este caso.

3.4.2.2 Solução via Decomposição Ortogonal

Uma alternativa para solucionar o problema é aplicar uma translação e rotação/reflexão aos eixos e então trabalhar no espaço transformado. Primeiro, desloca-se a origem para um dos centros. Por conveniência, escolhemos o primeiro ponto $\mathbf{a}_1$.

Seja $\mathbf{A}^*$ uma matriz $n \times (n-1)$ de pontos deslocados:

$$\mathbf{A}^* = [\mathbf{a}_2 - \mathbf{a}_1, \mathbf{a}_3 - \mathbf{a}_1, ..., \mathbf{a}_n - \mathbf{a}_1]. \tag{3.14}$$

Vamos aplicar uma transformação ortogonal de modo a anular as entradas da última linha de $\mathbf{A}^*$. Calcularemos a fatoração $\mathbf{QR}$ de $\mathbf{A}^*$:

$$\mathbf{A}^* = \mathbf{Q}_{n\times n} \begin{bmatrix} \mathbf{R} \\ 0^T \end{bmatrix}_{n\times(n-1)}, \tag{3.15}$$

em que $\mathbf{Q}$ é uma matriz ortogonal $n\times n$ e $\mathbf{R}$ é uma matriz triangular superior $(n-1)\times(n-1)$. Como $\mathbf{Q}$ é uma base de $\mathbb{R}^n$ podemos escrever,

$$\mathbf{x} - \mathbf{a}_1 = \mathbf{Q} \begin{bmatrix} \mathbf{y} \\ z \end{bmatrix}, \tag{3.16}$$

onde $\mathbf{y} \in \mathbb{R}^{n-1}$ e $z \in \mathbb{R}$.

Um das vantagens desta transformação é que as operações de translação e rotação/reflexão preservam a distância euclidiana, assim as equações (3.6) podem ser escritas

$$\| \mathbf{y} - \mathbf{r}_j \|_2^2 + z^2 \;=\; d_j^2, \quad j = \{1, ..., n-1\}, \tag{3.17}$$

e

$$\| \mathbf{y} \|_2^2 + z^2 \;=\; d_1^2, \tag{3.18}$$

onde $\mathbf{r}_j$ denota a j-ésima coluna de $\mathbf{R}$. Usando a equação (3.16) para substituir os termos não lineares em (3.17) por d_n^2, temos o sistema de equações lineares

$$\mathbf{R}^T\mathbf{y} = \mathbf{c}, \tag{3.19}$$

onde $\mathbf{c} \in \mathbb{R}^{n-1}$ tem componentes

$$c_j = \frac{1}{2}(d_1^2 - d_j^2 + \| r_j \|_2^2).$$

O sistema linear (3.19) é facilmente resolvido por substituição e (3.18) fornece

$$z_1 = +\sqrt{d_1^2 - \| \mathbf{y} \|_2^2}, \tag{3.20}$$

$$z_2 = -\sqrt{d_1^2 - \| \mathbf{y} \|_2^2} \tag{3.21}$$

e os pontos de intersecção são determinados aplicando a transformação (3.16).

Utilizando um dos dois métodos, determina-se duas posições possíveis para o ponto. Em seguida, é realizada uma análise para determinar se esta é uma solução

viável ou não. Caso a solução seja inviável, esta é podada de acordo com algum critério estabelecido.

No próximo capítulo apresentaremos critérios de poda para as possíveis posições e as principais modificações no algoritmo *branch-and-prune*, tanto na sua estrutura de busca quanto na de poda, permitindo generalizar como dados de entrada, pontos em $\mathbb{R}^m$.

4 Algoritmo *Branch-and-Prune*

Neste capítulo, apresentaremos uma breve descrição do algoritmo *branch-and-prune* projetado para determinar as coordenadas cartesianas de cada átomo de uma molécula, a partir de algumas distâncias inter-atômicas conhecidas. Descreveremos também as principais modificações realizadas no algoritmo, principalmente na estrutura de busca e poda, que permitem generalizar os dados de entrada para distâncias entre pontos em $\mathbb{R}^m$.

4.1 Motivação

Um algoritmo é qualquer procedimento computacional bem definido que toma algum valor ou conjunto de valores como entrada e produz algum valor ou conjunto de valores como saída (CORMEN; LEISERSON; RIVEST, 1990). Um algoritmo de busca em um grafo é um algoritmo que percorre seus vértices e arcos. Há muitas maneiras de fazer tal busca. Cada estratégia de busca é caracterizada pela ordem em que os vértices são visitados.

Uma estratégia de busca conhecida é a busca em profundidade. Esta busca progride através da expansão do primeiro vértice filho da árvore de busca e se aprofunda cada vez mais, até que o alvo da busca seja encontrado ou até que ele se depare com um vértice que não possui filhos (vértice folha). Então, a busca retrocede (*backtrack*) e começa no próximo vértice (EVEN, 2011). Numa implementação não-recursiva, todos os vértices expandidos recentemente são adicionados a uma pilha para realizar a exploração.

Um exemplo de aplicação, que realiza busca em profundidade em uma árvore é o de determinar a estrutura tridimensional de uma proteína. O problema de distâncias moleculares, apresentado no Capítulo 3, consiste em encontrar as coordenadas cartesianas de átomos de uma molécula, conhecidas algumas de suas distâncias.

No Capítulo 3 comentamos que a estrutura particular de algumas cadeias de átomos permite que o espaço de busca seja reduzida a um conjunto discreto de pontos. Esta particularidade, possibilita formular o problema discreto de distâncias moleculares (PDDM), o qual consiste em um subconjunto de instâncias do problema de distâncias moleculares (PDM).

O problema discreto de distâncias moleculares, apresentado em (LAVOR et al., 2011), contém a seguinte formulação:

PROBLEMA DISCRETO DE DISTÂNCIAS MOLECULARES (PDDM): dado um grafo não-direcionado simples $G = (V, E, d)$, tal que existe uma ordem em V,

satisfazendo:

1. E contém todos os cliques de quatro vértices consecutivos:
   $\forall i \in \{4, ..., n\} \ \forall \ j, k \in \{i-3, ..., i\} \ (\{j, k\} \in E)$;
2. A desigualdade triangular estrita é satisfeita:
   $\forall i \in \{2, ..., n-1\}$, $d_{i-1,i+1} < d_{i-1,i} + d_{i,i+1}$,

então existe uma aplicação $x : V \rightarrow \mathbb{R}^3$ tal que $\|x_u - x_v\|_2 = d_{uv}$ para cada $\{u, v\} \in E$?

Para solucionar este problema foi proposto o algoritmo *branch-and-prune* (BP), com base em uma abordagem combinatória.

Intuitivamente, o problema discreto de distâncias moleculares é descrito da seguinte forma:

- conhecidas algumas distâncias entre os átomos, em $\mathbb{R}^3$, é possível determinar as posições destes átomos em $\mathbb{R}^3$?

O problema de escalonamento multidimensional métrico é descrito da seguinte forma:

- conhecidas todas as distâncias entre n pontos em $\mathbb{R}^m$, deseja-se encontrar n pontos em $\mathbb{R}^3$ ou $\mathbb{R}^2$ com a "mesma" estrutura dos pontos originais permitindo uma visualização destes pontos.

Uma vez que o problema de escalonamento multidimensional métrico é um tipo de generalização do problema discreto de distâncias moleculares, pensamos em realizar modificações no algoritmo *branch-and-prune* para solucionar problemas de escalonamento multidimensional. As modificações no algoritmo *branch-and-prune* tem como princípio permitir a imersão de pontos de $\mathbb{R}^m$ em um espaço de menor dimensão.

Na próxima seção descreveremos a estratégia de poda do algoritmo *branch-and-prune*.

4.2 Algoritmo *Branch-and-Prune*

O algoritmo tipo *branch-and-prune* é muito utilizado, por exemplo, para resolver problemas de satisfação de restrição (*Constraint Satisfaction Problem* - CSP) (KUMAR, 1992). Aplicações *branch-and-prune* utilizam técnicas de busca exaustiva baseadas em ramificação (*branch*) e poda (*prune*) de ramos associados a soluções inviáveis.

Busca exaustiva é uma estratégia que percorre todo o espaço de possíveis soluções em busca da solução ótima do problema. Tipicamente, uma solução por busca exaustiva é composta de duas funções: uma que gera todas as possíveis soluções e outra que verifica se a solução gerada é a solução que atende ao problema.

No contexto apresentado, árvore é uma estrutura de dados formada durante a busca, onde cada vértice indica uma solução parcial do problema e cada ramo da árvore representa um conjunto de possíveis soluções viáveis a partir de um determinado vértice.

Os principais termos utilizados quando trabalhamos com árvore de busca binária são:

- Vértices - são todos os itens guardados na árvore,
- Raiz - é o vértice do topo da árvore,
- Filhos - são os vértices que vem depois dos outros vértices,
- Pais - são os vértices que vem antes dos outros vértices.

Uma árvore binária é uma estrutura de dados caracterizada por não ter elementos (árvore vazia) ou ter um elemento distinto, denominado raiz, com dois ponteiros para duas estruturas diferentes, denominadas sub-árvore esquerda e sub-árvore direita, ver Figura 9. Uma árvore de busca binária, pode ser representada por uma estrutura de dados encadeados em que cada vértice é um objeto.

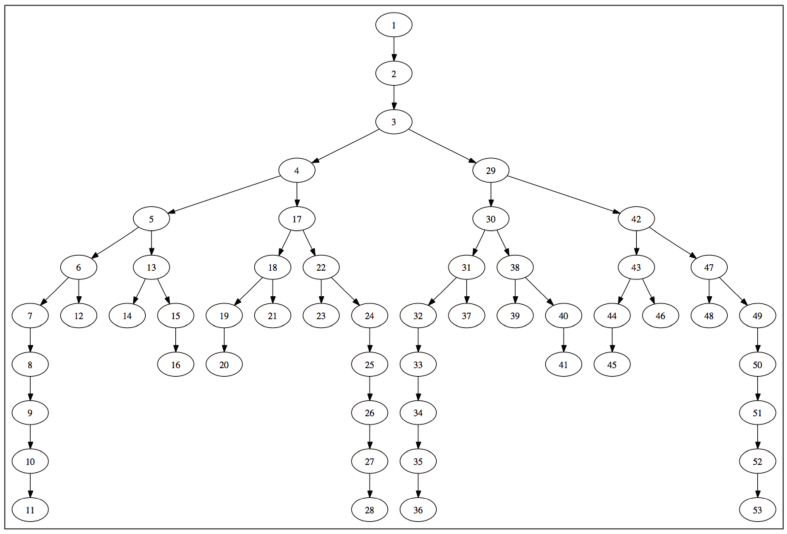

Figura 9 – Árvore Binária.

Além de um campo chave e de dados satélites, cada vértice contém campo esquerdo, campo direito e p, que apontam para os vértices correspondentes a seu filho

da esquerda, seu filho da direita e a seu pai, respectivamente (CORMEN; LEISERSON; RIVEST, 1990).

Operações em uma árvore binária requerem comparações entre vértices, em que a busca por um valor específico pode ser um processo recursivo ou iterativo.

O algoritmo *branch-and-prune* adaptado ao problema de proteínas utiliza a busca em árvore binária. Este algoritmo é caracterizado por fixar os 4 primeiros átomos e a cada passo, encontrar duas possíveis posições, x_i e $x_i^{'}$, para o átomo i que se deseja representar em $\mathbb{R}^3$, podendo estas posições serem consideradas infactíveis, uma vez não respeitada a tolerância predeterminada.

Existem três possíveis resultados:

1. ambas posições, x_i e $x_i^{'}$, são factíveis: neste caso serão armazenadas e exploradas,
2. apenas uma das posições é factível: somente a posição factível é armazenada e a infactível é descartada, ou seja, a árvore de busca é podada,
3. nenhuma posição é factível: poda-se ambas e a busca retrocede na árvore.

A factibilidade é verificada por testes de poda. No problema discreto de distâncias moleculares, tem-se o seguinte teste de poda: para todos os pares de distâncias $(j, i) \in E$ com $j < i$ (i é o átomo atual e j é anterior a i), verifica-se

$$|\| x_j - x_i \| - d_{ji}| < \epsilon, \tag{4.1}$$

em que $\epsilon > 0$ representa uma tolerância predeterminada. Ou seja, a poda elimina posições que acumulam erros que extrapolem uma tolerância predeterminada. O procedimento continua até encontrar todas as soluções do problema ou para-se a busca após encontrar o último átomo em uma posição factível.

Seja T uma representação gráfico da árvore de busca. Inicialmente, T é inicializada para $1 \rightarrow 2 \rightarrow 3$ porque os primeiros três átomos pode ser fixado em posições factíveis x_1, x_2, x_3, como explicado anteriormente. Em cada nó de busca da árvore de posição i armazenamos:

- a posição $x_i \in \mathbb{R}^3$ do i-ésimo átomo;
- o produto $C_i = \Pi_{j=1}^{i} B_j$ das matrizes de torção;
- um apontador para o nó pai $P(i)$;
- ponteiros para subnós $L(i)$ e $R(i)$.

Algoritmo 1: Algoritmo Branch-and-Prune

início
BranchAndPrune(T, v, i, n)
se $(i \leqslant n-1)$ **então**
Calcule as possíveis posições para o i-ésimo átomo:
Calcule as matrizes de torção B_i e $B_i^{'}$;
Recupera a matriz de torção C_{i-1} do vértice $P(v)$;
Calcula $C_i = C_{i-1}B_i$, $C_i^{'} = C_{i-1}B_i^{'}$ e x_i, $x_i^{'}$ de C_iy, $C_i^{'}y$;
Seja $\lambda = 1$, $\rho = 1$;
Verificar Factibilidade:
para *todo* $(j, i) \in E$ **faça**
Seja $\delta_{ji} = (\| x_j - x_i \|^2 - d_{ji}^2)^2$ e $\delta_{ji}^{'} = (\| x_j - x_i^{'} \|^2 - d_{ji}^2)^2$;
se $(\delta_{ji} > \epsilon)$ **então**
$\lambda = 0$
fim
se $(\delta_{ji}^{'} > \epsilon)$ **então**
$\rho = 0$
fim
fim
Criar subvértices conforme exigido:
se $(\lambda = 1)$ **então**
Criar um vértice z, armazenar C_i e x_i em z, seja $P(z) = v$ e $L(v) = z$;
$T \leftarrow T \cup \{z\}$;
BranchAndPrune$(T, z, i+1, n)$; **senão**
$L(v) = PODA$;
fim
se $(\rho = 1)$ **então**
Criar um vértice $z^{'}$, armazenar C_i e x_i em $z^{'}$, seja $P(z) = v$ e $R(v) = z^{'}$;
$T \leftarrow T \cup \{z^{'}\}$;
BranchAndPrune$(T, z^{'}, i+1, n)$; **senão**
$R(v) = PODA$;
fim
fim
fim

No Algoritmo 1, tem-se o pseudo-código do algoritmo *branch-and-prune*.

No problema de escalonamento multidimensional, os erros possuem outras características que exigem uma estratégia de poda mais complexa, devido à transformação de $\mathbb{R}^m$ para $\mathbb{R}^3$.

A seguir, comentaremos sobre o algoritmo de Dijkstra que servirá de base para formular a estratégia de poda para solucionar problemas de escalonamento multidimensional.

4.3 Problema do Caminho Mínimo

O problema do caminho mínimo, considerado clássico em otimização combinatória tem sido amplamente estudado (DENARDO; FOX, 1979; DEO; PANG, 1979; DIAL et al., 1979; DIJKSTRA, 1959), em vista de suas aplicações em inúmeras situações práticas que ocorrem em transportes, logística, redes de computadores e de telecomunicações, etc. Estas situações são conhecidas como o problema do caminho mínimo pois visam minimizar o custo entre dois vértices; custo este dado pela soma dos pesos de cada aresta percorrida. Há um grande número de situações possíveis ao se efetuar uma busca dos caminhos mínimos, podendo-se considerar ou não diversas premissas, como por exemplo (NETTO, 1979):

1. obtenção dos caminhos mínimos de um vértice dado, aos vértices restantes quando o peso das arestas são não negativos;
2. obtenção dos caminhos mínimos de um vértice dado, aos vértices restantes quando os pesos das arestas são arbitrários;
3. obtenção dos caminhos mínimos entre todos os pares de vértices;
4. generalizações do problema dos caminhos mínimos.

O problema dos caminhos mínimos com uma única fonte [1] consiste em: dado $G = (V, E)$ um grafo ponderado, obter um caminho mais curto de um determinado vértice de origem $s \in V$ até todo vértice $v \in V$.

Um algoritmo para resolver o problema dos caminhos mínimos com única fonte foi proposto por E. W. Dijkstra (DIJKSTRA, 1959). A motivação surgiu quando lhe foi solicitado minimizar a quantidade de cabos no painel de um computador que estava sendo projetado.

Apresentaremos aqui uma breve descrição formal do algoritmo.

4.3.1 Algoritmo de Dijkstra

O algoritmo de Dijkstra é o mais conhecido dos algoritmos para problemas dos caminhos de custo mínimo entre vértices de um grafo e, na prática, o mais empregado. Foi criado por um cientista da computação chamado Edsger W. Dijkstra. Escolhido um vértice como raiz de busca, o algoritmo calcula o peso (custo) mínimo deste vértice aos demais vértices de um grafo orientado ou não, com arestas de peso não negativo, em tempo computacional $O([m + n]logn)$ onde m representa o número de arestas e n representa o número de vértices (CORMEN; LEISERSON; RIVEST, 1990). O algoritmo não foi

[1] *Single source shortest paths problem*

desenvolvido para aplicações onde as arestas apresentam pesos negativos. No contexto apresentado, a restrição de todas as arestas possuírem pesos não negativos não cria dificuldades, pois os pesos no problema apresentado representam erros.

O algoritmo de Dijkstra resolve o seguinte problema. Dado um grafo ponderado $G = (V, E, \omega)$ e um vértice como raiz de busca s, ele devolve:

1. para cada vértice $v \in V$, o peso de um caminho mínimo de s a v e
2. uma árvore dos caminhos mínimos com raiz em s. Um caminho de s a v nesta árvore é um caminho mínimo de s a v em G.

Para cada vértice $v \in V$, o algoritmo mantém dois atributos $dist[v]$ e $cam[v]$:

- $dist[v]$ é uma estimativa do caminho mais curto até v. Inicialmente, as estimativas do caminho mais curto de todos os vértices além da fonte são tomados como ∞.
- $cam[v]$ armazena o antecessor de v em um caminho de s a v com peso $dist[v]$. É utilizado para representar a árvore dos caminhos mínimos que o algoritmo devolve.

Os passos são apresentados no Algoritmo 2. Se $v \neq s$ e $cam[v] \longleftarrow Nulo$, isto significa que v não pertence à árvore determinada até o momento. Após a inicialização, o algoritmo atualiza a distância estimada de um vértice ao longo de uma aresta.

No Algoritmo 2, T representa a árvore dos caminhos mais curtos à fonte s, ou seja, T é a informação de saída. O algoritmo começa com uma árvore contendo apenas s e, a cada iteração, um novo vértice é acrescentado à T.

Exemplo:

Seja $G(V, E, \omega)$ um grafo orientado [ver figura 10]. Seja S o vértice de origem (raiz) do grafo. Atribui-se zero a sua $dist[S]$ pois o peso de ir de S a S é zero. Todos os outros vértices i tem suas distâncias $dist[i]$ inicializadas com infinito (inf).

A partir de S consulta-se os vértices adjacentes a ele [ver figura 11], que no grafo G são U e X. Seja Z todos os vértices adjacentes, então

1: Se $dist[Z] > dist[S] + \omega(S, Z)$ então
2: $dist[Z] = dist[S] + \omega(S, Z)$,
3: $cam[Z] \longleftarrow S$.

Algoritmo 2: ALGORITMO DIJKSTRA

Entrada: `Grafo ponderado direcionado` $G = (V, E, \omega)$, `um vértice inicial` s.
Saída: `Vetor de distâncias dist[] do vértice` s `a todos os outros e cam[].`

início
  $dist[s] \longleftarrow 0$
  $cam[s] \longleftarrow indefinido$
  para *cada* $v \in V$ **faça**
    se $v \neq s$ **então**
      $dist[v] \longleftarrow \infty$
      $cam[v] \longleftarrow indefinido$
    fim
    adiciona v em W
  fim
  se $W \neq \varnothing$ **então**
    $u \longleftarrow$ vértice em W com mínima $dist[u]$
    remover u de W
  fim
  para *cada* v *adjacente a* u **faça**
    se $dist[v] > dist[u] + \omega(u, v)$ **então**
      $dist[v] = dist[u] + \omega(u, v)$
      $cam[v] \longleftarrow u$
    fim
  fim
fim

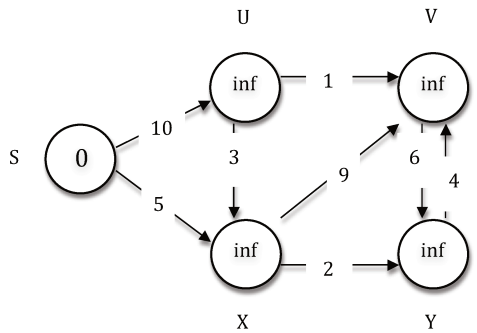

Figura 10 – Funcionamento do Algoritmo de Dijkstra.

Dentre todos os vértices adjacentes, escolhe-se aquele com menor distância. Neste caso, vértice X, pois $dist[X] = 5$ [ver figura 12].

A partir de X consulta-se os vértices adjacentes a X que não fazem parte do caminho [ver figura 13], que no grafo G são V e Y. Para os vértices adjacentes Z, calcula-se:

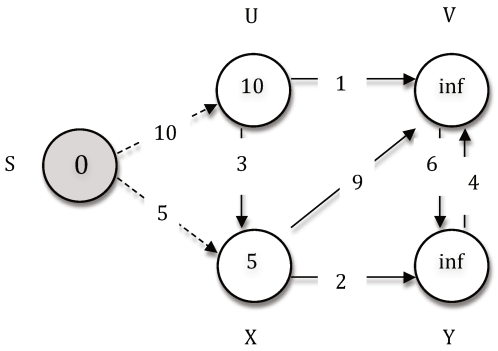

Figura 11 – Funcionamento do Algoritmo de Dijkstra.

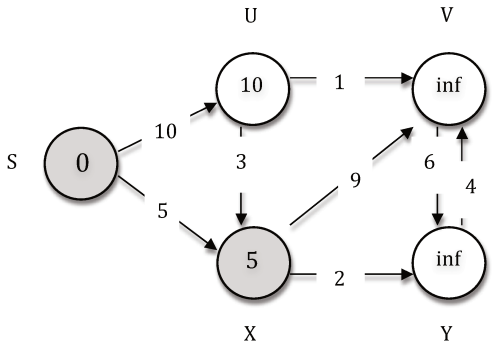

Figura 12 – Funcionamento do Algoritmo de Dijkstra.

1: Se $dist[Z] > dist[X] + \omega(X, Z)$ então
2: $dist[Z] = dist[X] + \omega(X, Z)$,
3: $cam[Z] \longleftarrow X$.

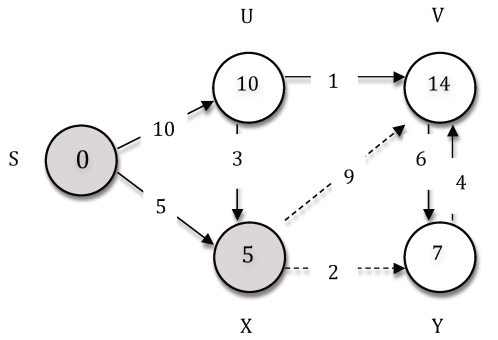

Figura 13 – Funcionamento do Algoritmo de Dijkstra.

Dentre todos os vértices adjacentes, escolhe-se aquele com menor distância. Neste caso, vértice Y, pois $dist[Y] = 7$ [ver figura 14].

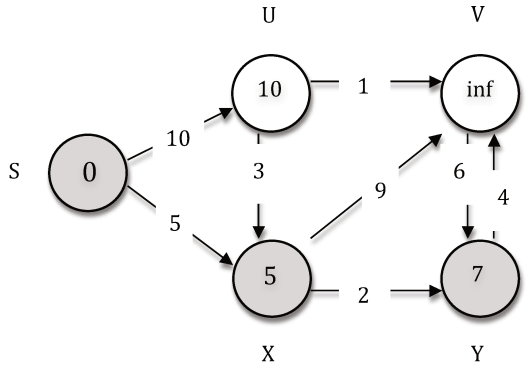

Figura 14 – Funcionamento do Algoritmo de Dijkstra.

A partir de Y consulta-se os vértices adjacentes a Y que não fazem parte do caminho [ver figura 15], que no grafo G é apenas o vértice V. Para os vértices adjacentes Z, calcula-se:

1: Se $dist[V] > dist[Y] + \omega(Y, V)$ então
2: $dist[V] = dist[Y] + \omega(Y, V)$,
3: $cam[V] \longleftarrow Y$.

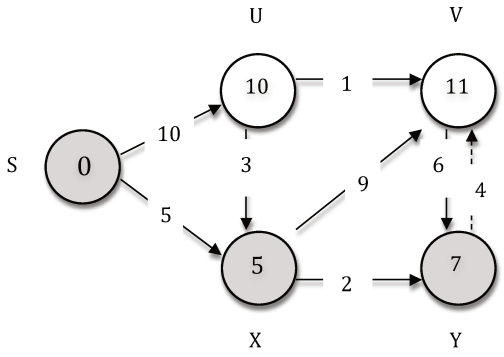

Figura 15 – Funcionamento do Algoritmo de Dijkstra.

Dentre os vértices não pertencentes ao caminho, escolhe-se aquele com menor distância. Neste caso é o vértice U, pois $dist[U] = 10$ [ver figura 16].

Inclui-se então U e consulta-se os vértices adjacentes a ele que não pertencem ao caminho, que é apenas o vértice V [ver figura 17].

1: Se $dist[V] > dist[U] + \omega(U, V)$ então
2: $dist[V] = dist[U] + \omega(U, V)$,
3: $cam[V] \longleftarrow U$.

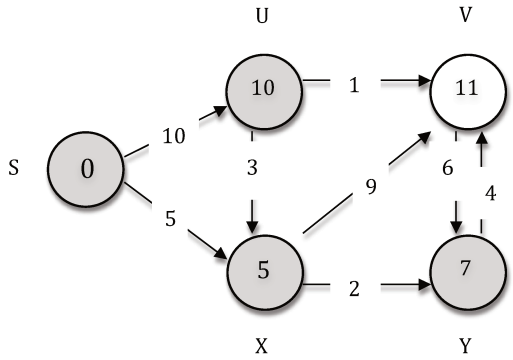

Figura 16 – Funcionamento do Algoritmo de Dijkstra.

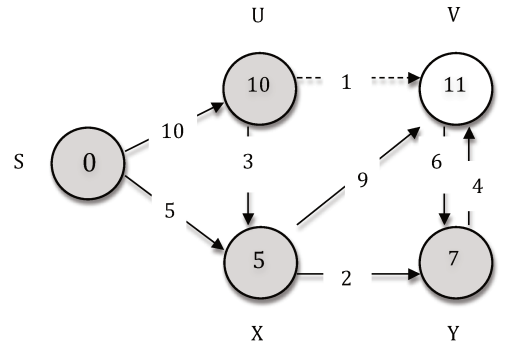

Figura 17 – Funcionamento do Algoritmo de Dijkstra.

Dentre os vértices não pertencentes ao caminho, escolhe-se aquele com menor distância. Neste caso é o único vértice V, pois $dist[V] = 11$. Faz-se V pertencer ao caminho e a busca é finalizada [ver figura 18]. Na figura 18 podemos observar, por exemplo, que o peso do caminho mínimo de S a V é 11 e o caminho é $S \rightarrow U \rightarrow V$.

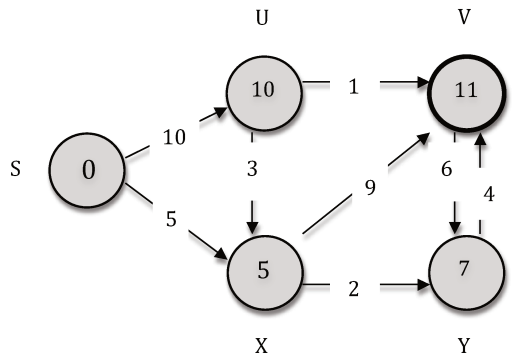

Figura 18 – Funcionamento do Algoritmo de Dijkstra.

Existem várias aplicações do algoritmo de Dijkstra, entre estas estão: saber a trajetória de menor distância entre 2 cidades tendo conhecimento de diversas estra-

das que passam por várias cidades; cálculo de rotas para GPS; otimização de redes de telecomunicações e processamento de imagens (XIAO-YAN L., 2010; SINGAL P., 2014).

Apresentaremos nas próximas seções como é realizada a busca no algoritmo *branch-and-prune* modificado e a estratégia de poda com base no algoritmo de Dijkstra adotada para solucionar problemas de escalonamento multidimensional.

4.4 *Branch-and-Prune* Modificado

Nesta seção, descreveremos os principais passos do algoritmo *branch-and-prune* modificado. Os dados iniciais ou dados de entrada do algoritmo são:

- uma matriz $n \times n$ cujo os elementos são dissimilaridades,
- dimensão de imersão k, principal caso, $k = 3$.

O primeiro passo do algoritmo *branch-and-prune* modificado é elevar ao quadrado cada componente da matriz de entrada e analisar se esta matriz é considerada uma matriz de distâncias euclidianas. Inicialmente, verifica se

1. $d(x, y) \geqslant 0$: não negatividade;
2. $d(x, y) = 0 \Leftrightarrow x = y$;
3. $d(x, y) = d(y, x)$: simetria;
4. $d(x, z) \leqslant d(x, y) + d(y, z)$: desigualdade triangular.

Caso a matriz atenda às condições iniciais, o próximo passo é calcular as posições dos $k + 1$ primeiros pontos: fixamos o primeiro ponto na origem, determinamos as posições dos $k + 1$ primeiros pontos através da resolução de sistemas não lineares apresentados na seção 3.3.2. Para $k = 3$, fixa-se os 4 primeiros pontos. Os dois primeiros pontos são determinados por:

$$\mathbf{x}_1 = \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix} \quad , \quad \mathbf{x}_2 = \begin{pmatrix} d_{12} \\ 0 \\ 0 \end{pmatrix}.$$

O terceiro ponto é determinado pela solução das 2 equações não-lineares:

$$\| \mathbf{x}_3 - \mathbf{x}_1 \|_2^2 = d_{13}^2, \tag{4.2}$$

$$\| \mathbf{x}_3 - \mathbf{x}_2 \|_2^2 = d_{23}^2, \tag{4.3}$$

ou equivalentemente,

$$\mathbf{x}_3 = \begin{pmatrix} (d_{12}^2 + d_{31}^2 - d_{32}^2)/2d_{12} \\ \sqrt{d_{31}^2 - ((d_{12}^2 + d_{31}^2 - d_{3,2}^2)/2d_{12})^2} \\ 0 \end{pmatrix}.$$

O quarto ponto é determinado pela solução do sistema:

$$\| \mathbf{x}_4 - \mathbf{x}_1 \|_2^2 = d_{14}^2, \tag{4.4}$$

$$\| \mathbf{x}_4 - \mathbf{x}_2 \|_2^2 = d_{24}^2, \tag{4.5}$$

$$\| \mathbf{x}_4 - \mathbf{x}_3 \|_2^2 = d_{34}^2. \tag{4.6}$$

As demais posições também podem ser calculados via decomposição ortogonal ou eliminação Gaussiana, seção 3.3.2. Neste processo, em cada passo, podem ocorrer três situações:

- nenhuma solução é encontrada: neste caso, o problema não atende à desigualdade triangular estrita, ou seja, $\forall i \in \{2, ..., n-1\}$, $d_{i-1,i+1} < d_{i-1,i} + d_{i,i+1}$;
- apenas uma solução é encontrada: neste caso, armazena-se o erro e busca-se o próximo ponto;
- duas soluções são encontradas.

As duas possíveis soluções encontradas são oriundas da eliminação de Gauss, como visto na equação (3.13):

$$r_1 = \frac{2 - \mathbf{u}^T\mathbf{v} + \sqrt{(2 - \mathbf{u}^T\mathbf{v})^2 - (\mathbf{u}^T\mathbf{u})(\mathbf{v}^T\mathbf{v})}}{\mathbf{u}^T\mathbf{u}},$$

$$r_2 = \frac{2 - \mathbf{u}^T\mathbf{v} - \sqrt{(2 - \mathbf{u}^T\mathbf{v})^2 - (\mathbf{u}^T\mathbf{u})(\mathbf{v}^T\mathbf{v})}}{\mathbf{u}^T\mathbf{u}},$$

onde $\mathbf{u}$ e $\mathbf{v}$ são dados em (3.11).

Enquanto, por decomposição ortogonal, as duas possíveis soluções são encontradas por:

$$z_1 = \sqrt{d_1^2 - \| \mathbf{y} \|_2^2} \quad , \quad z_2 = -\sqrt{d_1^2 - \| \mathbf{y} \|_2^2},$$

onde d_1 e $\mathbf{y}$ são dados em (3.18) e (3.16), respectivamente.

No caso de serem encontradas duas soluções, o próximo passo é analisar a factibilidade destas soluções parciais. A busca acha todas as soluções com erro mínimo, ou seja, todas soluções com mesmo erro final aparecem no final do processo. Na próxima seção, serão apresentadas as principais alterações na estratégia de poda com base no algoritmo de Dijkstra.

4.5 Estratégia de Poda do Algoritmo *Branch-and-Prune* Modificado

A poda do algoritmo *branch-and-prune* aplicado a proteínas ocorre quando o erro entre as distâncias é maior que a tolerância ϵ, como podemos ver no teste de poda em (4.1). Neste caso, as distâncias são oriundas do próprio espaço de imersão, ou seja, estes erros são mínimos sendo na maioria dos casos erros computacionais. Generalizando este processo para pontos em $\mathbb{R}^m$, ao tentar representar estes pontos em $\mathbb{R}^3$, geralmente o erro é muito maior que ϵ. Assim, para tolerâncias muito pequenas, o algoritmo interrompe o processo não chegando a uma solução. Para evitar este tipo de situação é necessário um aprimoramento no teste de poda.

O algoritmo *branch-and-prune* determina pontos em $\mathbb{R}^3$ preservando as distâncias da matriz 7-diagonal: a diagonal nula e as três primeiras diagonais acima e abaixo da diagonal da matriz de distâncias original D. Uma matriz $D_{n\times n}$ é definida $(2k+1)$-diagonal se $d_{ij} = 0$, sempre que $\mid i - j \mid > k$. Por exemplo, seja $D_{6\times 6}$ uma matriz de distâncias euclidianas:

$$D = \begin{bmatrix} 0 & d_{12} & d_{13} & d_{14} & d_{15} & d_{16} \\ d_{12} & 0 & d_{23} & d_{24} & d_{25} & d_{26} \\ d_{13} & d_{23} & 0 & d_{34} & d_{35} & d_{36} \\ d_{14} & d_{24} & d_{34} & 0 & d_{45} & d_{46} \\ d_{15} & d_{25} & d_{35} & d_{45} & 0 & d_{56} \\ d_{16} & d_{26} & d_{36} & d_{46} & d_{56} & 0 \end{bmatrix}.$$

O algoritmo *branch-and-prune* preserva as seguintes distâncias:

$$D^* = \begin{bmatrix} 0 & d_{12} & d_{13} & d_{14} & * & * \\ d_{21} & 0 & d_{23} & d_{24} & d_{25} & * \\ d_{31} & d_{32} & 0 & d_{34} & d_{35} & d_{36} \\ d_{41} & d_{42} & d_{43} & 0 & d_{45} & d_{46} \\ * & d_{52} & d_{53} & d_{54} & 0 & d_{56} \\ * & * & d_{63} & d_{64} & d_{65} & 0 \end{bmatrix}.$$

A poda neste caso tem como objetivo encontrar soluções onde as distâncias que não são preservadas ($d_{ij} = *$) estejam próximas, o máximo possível, das distâncias originais, ou seja, minimiza-se o erro nestas posições. Sendo o objetivo encontrar soluções com erro mínimo, a estratégia de busca do algoritmo *branch-and-prune* modificado teve como base o algoritmo de Dijkstra. Busca-se encontrar soluções com um erro mínimo dentre todas as soluções.

Seja $D_{n\times n}$ uma matriz de distâncias euclidianas e T a representação gráfica da árvore de busca que é inicializada pelas posições $1 \rightarrow 2 \rightarrow 3 \rightarrow 4$, sem presença de erro, logo sem ocorrência de poda. As posições x_1, x_2, x_3, e a quarta posição x_4 podem ser fixadas em qualquer uma das duas possíveis posições sem gerar erro (LAVOR et al., 2012).

Fixando as posições x_1, x_2, x_3 e x_4, encontra-se as posições x_5 e x_5' solucionando sistemas não lineares. Calcula-se o erro das duas soluções parciais:

$$(x_1, x_2, x_3, x_4, x_5) \text{ e } (x_1, x_2, x_3, x_4, x_5').$$

Uma das possíveis maneiras de definir o erro é por uma p-norma matricial da diferença entre a matriz parcial de distâncias originais DP e DP_{BP} a matriz parcial de distâncias encontradas através do Algoritmo 3:

$$Erro = \| DP - DP_{BP} \|_p.$$

No próximo capítulo abordaremos algumas análises de erro juntamente com algumas simulações.

Após o cálculo do erro, a posição que apresentar erro mínimo, por exemplo x_5', é utilizada para encontrar as posições dos próximos pontos, enquanto que a solução com maior erro, x_5, é armazenada. Após encontrar a solução parcial com erro mínimo, determina-se as posições x_6 e x_6'. Novamente, calcula-se o erro das duas soluções parciais e analisamos a posição que apresenta erro mínimo. Supondo que x_5 apresente erro mínimo entre as soluções parciais, esta é utilizada para encontrar as posições dos próximos pontos, x_6'' e x_6''', enquanto que a solução com maior erro é armazenada.

A busca prossegue até que todas as soluções com erro mínimo sejam encontradas. Logo, o algoritmo realizou uma busca até deparar com um vértice que não possui filhos, ou seja, realizou uma busca em profundidade. Tanto o algoritmo *branch-and-prune* quanto o *branch-and-prune* modificado, encontram todas as soluções com erro mínimo. Esta é uma característica da escolha do algoritmo Dijkstra como referência na busca por soluções. O maior problema deste algoritmo esta no fato de não existir uma poda definitiva de possíveis soluções antes das soluções ótimas serem encontradas, sendo necessário armazenar uma grande quantidade de dados. Este problema não impede de encontrar as soluções ótimas, mas pode ocasionar falhas pelo excesso de memória ocupado. Uma solução alternativa é apresentanda na próxima seção.

4.6 Nova Poda

Na ausência de poda, o algoritmo encontra todas as possíveis soluções antes de obter uma solução ótima, sendo necessário armazenar uma quantidade excessiva de dados calculados em cada iteração. Desta forma, temos que a maior dificuldade deste algoritmo está no esforço computacional exigido. Para solucionar este transtorno, foi implementada um novo tipo de poda.

Na busca, mesmo obtendo uma solução parcial com erro mínimo, não podemos descartar quaisquer outras soluções parciais com erros maiores, uma vez que faltam vértices para completar a solução, ou seja, é possível que a melhor solução até o momento deixe de ser nos passos seguintes. Uma forma encontrada para solucionar esta dificuldade consiste em localizar uma solução gulosa, que é determinada pelas posições com erro mínimo entre as duas possíveis posições, ou seja, a busca prossegue escolhendo a melhor posição com base nas informações locais, estratégia conhecida como estratégia gulosa (CORMEN; LEISERSON; RIVEST, 1990; BANG-JENSEN J., 2004). A partir deste erro poda-se qualquer solução parcial que possua um erro até então maior. Assim, as soluções podadas até encontrar a solução ótima não poderiam ser ótimas, pois seu erro já é maior que o da solução gulosa encontrada mesmo não estando ainda completas. Estes passos são repetidos até se obter o ponto n, ou seja, até encontrar possíveis soluções ótimas.

O próximo passo após encontrar uma possível solução ótima é podar todas as soluções parciais com erro maior que o erro apresentando pela possível solução ótima. Ou seja, esta busca retrocede (*backtrack*) e avalia, se necessário, o próximo vértice. Após a poda, dentre as soluções parciais restantes, escolhe-se aquela com erro mínimo. A partir da solucão parcial escolhida, x_i, determina-se as posições x_{i+1} e x'_{i+1}. Novamente, calcula-se o erro das duas soluções parciais armazenando a posição que apresentar maior erro. A solução parcial com erro mínimo será comparada com a possível solução ótima, podendo apresentar três situações, como será mostrado abaixo.

Erro próximo ao da possível solução ótima: a proximidade é analisada, neste trabalho, pela diferença entre os erros com precisão de 10^{-8}. O próximo passo é verificar se a solução parcial apresenta n pontos, número de pontos do conjunto de entrada. Neste caso, a solução parcial é acrescentada ao conjunto de possíveis soluções ótimas. Caso contrário, permanece no conjunto de soluções parciais.

Erro menor ao da possível solução ótima: Se a solução parcial apresentar n pontos, então, as possíveis soluções ótimas serão podadas e a solução parcial é acrescentada ao conjunto de possíveis soluções ótimas.

Erro maior ao da possível solução ótima: A solução parcial será podada.

Estes passos são realizados enquanto existirem soluções não podadas. Neste caso, a poda faz com que a busca pare de avaliar soluções que apresentem erros piores do que soluções previamente examinadas, retornando no final do processo um conjunto de soluções ótimas para o problema.

A Figura 19 mostra um exemplo da nova poda, onde tem-se 8 pontos em $\mathbb{R}^m$ e deseja-se obter a mesma estrutura em $\mathbb{R}^3$.

Como pode-se observar na Figura 19, foram feitas várias escolhas que pareciam ser a melhor no momento, com objetivo de ter uma solução ótima no final. Inicialmente, fixamos as posições

$$1 \to 2 \to 3 \to 4,$$

e determinamos as duas possíveis posições para o quinto ponto, 5 e 6. Entre estas duas, o ponto 5 apresenta menor erro, assim tem-se como solução parcial

$$1 \to 2 \to 3 \to 4 \to 5.$$

Fazemos isso sucessivamente, até encontrar a solução

$$1 \to 2 \to 3 \to 4 \to 5 \to 8 \to 9 \to 12. \tag{4.7}$$

Esta solução apresenta um erro maior do que o erro da solução parcial,

$$1 \to 2 \to 3 \to 4 \to 5 \to 8 \to 10.$$

Neste caso, armazena-se a solução encontrada até o momento e analisa-se a solução parcial, obtendo como solução

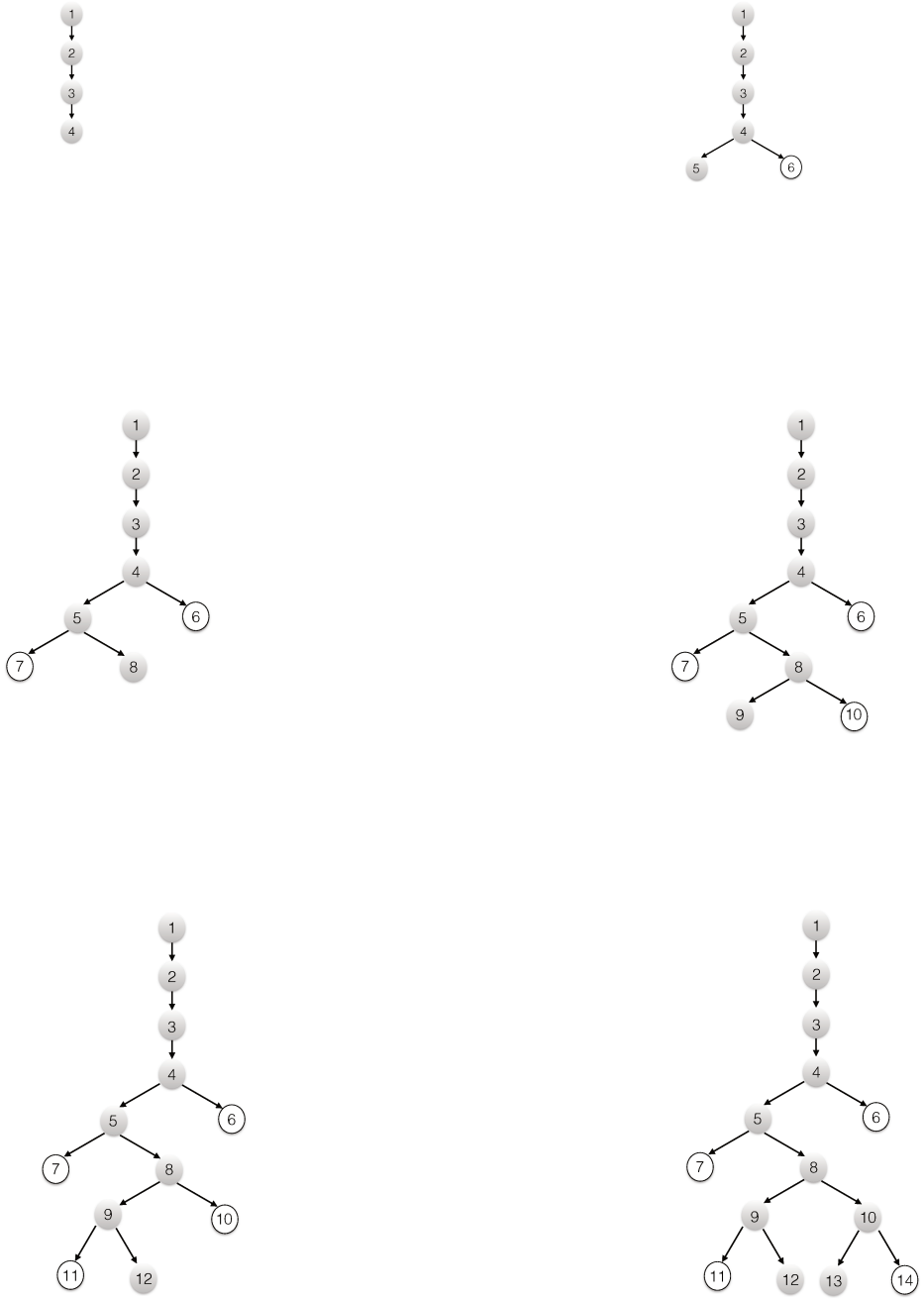

Figura 19 – Estratégia de Poda.

$$1 \to 2 \to 3 \to 4 \to 5 \to 8 \to 10 \to 13.$$

Ou seja, ao encontrar uma solução com menor erro, poda-se a solução encontrada até o momento. Tendo esta solução um erro menor do que a solução encontrada, tem-se esta como solução ótima, caso contrário, (4.10) é a solução ótima.

Um dos objetivos da nova poda no algoritmo *branch-and-prune* modificado é reduzir o número de vértices a serem analisados na árvore de busca, reduzindo assim a

quantidade de memória utilizada.

O algoritmo com ausência de poda e com uma poda definitiva são equivalentes, no sentido de que ambos encontram apenas as soluções mínimas. Entretanto, do ponto de vista de eficiência computacional, o procedimento com nova poda é menos eficiente em relação ao tempo de processamento, quando comparado ao mesmo procedimento sem poda, contudo, ganha muito no que se refere a espaço de memória utilizado, uma vez que a poda pode ser feita desde o início, eliminando conjuntos de soluções parciais não ótimos.

No próximo capítulo, serão apresentados alguns resultados computacionais obtidos com o algoritmo *branch-and-prune* modificado (nova poda), onde também apresentamos comparações com os resultados do método análise de coordenadas principais, apresentado no Capítulo 2.

5 Resultados Computacionais

Neste capítulo apresentaremos alguns resultados obtidos a partir do algoritmo *branch-and-prune* modificado com a nova poda e faremos uma comparação com o método análise de coordenadas principais apresentado no Capítulo 2.

5.1 Avaliação dos Métodos

A análise dos resultados obtidos pelos dois métodos tem como base as seguintes variáveis: número de pontos no espaço de origem, dimensão do espaço de origem, dimensão de imersão, erro utilizado na poda e erro cometido ao tentar manter a matriz de distâncias originais.

Os testes realizados variam com o número de pontos e dimensão do espaço em que estes pontos se encontram, sendo mantida a análise para dimensão de imersão com $k = 3$ que caracteriza a principal aplicação, ou seja, obter a visualização da estrutura destes pontos e norma-2 para cálculo do erro na poda. Foram gerados pontos em $\mathbb{R}^N$ e calculadas as matrizes de distâncias euclidianas entre os pontos, logo são conhecidas todas as distâncias. A comparação da estrutura original e a estrutura de imersão foi realizada com base nos erros dados por uma p-norma matricial da diferença entre a matriz de distâncias originais D e D_{BP} a matriz de distâncias dos pontos obtidos pelo algoritmo BP modificado,

$$Erro = \| D - D_{BP} \|_p,$$

referentes à norma-2, norma-4 e à norma infinito ou norma do máximo.

5.2 Resultados para N dimensões de origem

O algoritmo *branch-and-prune* modificado foi implementado em *Matlab2010* e para análise de coordenadas principais foi utilizada a função `cmdscale` no *Statistics Toolbox*^{TM}. As instâncias nesta seção foram geradas pelo comando `rand` do *Matlab2010* que fornece números aleatórios distribuídos uniformemente no intervalo $(0, 1)$. Todos os resultados descritos nesta seção foram testados em um computador com processador Intel Core 2 Duo 2.66GHz e sistema operacional MAC OSX.

Observando as tabelas 5.1 a 5.7 referentes às dimensões de origem, pode-se verificar que os erros para o algoritmo *branch-and-prune* modificado são menores,

- **Dimensão de Origem** $N = 4$ e **Dimensão de Imersão** $k = 3$

Algoritmos	Normas	Teste 1	Teste 2	Teste 3	Teste 4	Teste 5
Análise de Coordenadas Principais	Norma-2	0,4778	0,8745	0,3462	0,3829	0,8245
	Norma-4	0,3458	0,9712	0,2919	0,2931	0,4489
	Máximo	0,2204	0,8128	0,1948	0,2083	0,2433
Branch-and-Prune Modificado	Norma-2	0,0159	0,0388	0,0944	0,0285	0,1781
	Norma-4	0,0189	0,0462	0,1122	0,0339	0,2118
	Máximo	0,0159	0,0388	0,0944	0,0285	0,1781

Tabela 2 – Comparação entre os métodos. 5 pontos gerados em $\mathbb{R}^4$

- **Dimensão de Origem** $N = 4$ e **Dimensão de Imersão** $k = 3$

Algoritmos	Normas	Teste 1	Teste 2	Teste 3	Teste 4	Teste 5
Análise de Coordenadas Principais	Norma-2	12,1288	8,7966	11,3265	8,0702	10,7964
	Norma-4	4,8843	3,4553	4,1671	3,6504	4,1463
	Máximo	2,9314	2,1207	2,6724	2,2118	2,5915
Branch-and-Prune Modificado	Norma-2	4,6153	3,7754	4,3844	3,7142	4,6084
	Norma-4	2,7484	2,1704	2,5165	2,2383	2,6381
	Máximo	1,2694	1,0066	1,1307	1,0917	1,2205

Tabela 3 – Comparação entre os métodos. 30 pontos gerados em $\mathbb{R}^4$

- **Dimensão de Origem** $N = 10$ e **Dimensão de Imersão** $k = 3$

Algoritmos	Normas	Teste 1	Teste 2	Teste 3	Teste 4	Teste 5
Análise de Coordenadas Principais	Norma-2	11,0401	16,0654	14,5836	23,0543	22,4684
	Norma-4	7,4061	9,7017	8,2007	13,0513	14,3134
	Máximo	5,4054	7,0586	5,4576	9,2882	11,2151
Branch-and-Prune Modificado	Norma-2	3,8687	2,6868	3,1115	3,5264	3,3147
	Norma-4	3,4057	2,9670	2,9706	3,6138	2,6461
	Máximo	2,0907	2,3031	2,0033	2,7891	1,5707

Tabela 4 – Comparação entre os métodos. 10 pontos gerados em $\mathbb{R}^{10}$

- **Dimensão de Origem** $N = 50$ e **Dimensão de Imersão** $k = 3$

Algoritmos	Normas	Teste 1	Teste 2	Teste 3	Teste 4	Teste 5
Análise de Coordenadas Principais	Norma-2	392,0143	351,6853	380,5167	353,4899	424,2512
	Norma-4	182,3993	176,6635	184,3586	168,5463	201,6522
	Máximo	98,1856	120,4037	114,6035	90,5813	113,7591
Branch-and-Prune Modificado	Norma-2	17,8805	16,3047	18,9781	17,6342	16,9658
	Norma-4	16,6094	15,4366	17,3233	15,9579	16,0181
	Máximo	10,3596	9,1038	10,7033	9,6082	10,9581

Tabela 5 – Comparação entre os métodos. 10 pontos gerados em $\mathbb{R}^{50}$

independente da quantidade de pontos e da dimensão de origem. Por exemplo, na tabela (5.3), foram gerados 10 pontos em $\mathbb{R}^{10}$ e na coluna do Teste 1 pode-se observar que os pontos gerados pelo método análise de coordenadas principais apresentam erro de $11,0401$ em relação às distâncias originais para norma-2, $7,4061$ em relação às distâncias originais para norma-4 e $5,4054$ para norma do máximo. Para os pontos gerados pelo algoritmo *branch-and-prune* modificado (BPM), suas distâncias possuem erro $3,8687$ para

- **Dimensão de Origem** $N = 50$ e **Dimensão de Imersão** $k = 3$

Algoritmos	Normas	Teste 1	Teste 2	Teste 3	Teste 4	Teste 5
Análise de Coordenadas Principais	Norma-2	871,4361	783,4989	707,7136	754,3872	833,0048
	Norma-4	273,5995	253,1707	217,6220	235,2813	263,2131
	Máximo	129,6438	121,5897	100,9276	124,7464	124,6632
Branch-and-Prune Modificado	Norma-2	63,3054	61,7057	60,5103	63,0439	69,4805
	Norma-4	31,7384	30,8361	30,8000	31,0484	37,9116
	Máximo	11,7142	12,1531	12,0348	15,3323	17,1210

Tabela 6 – Comparação entre os métodos. 30 pontos gerados em $\mathbb{R}^{50}$

- **Dimensão de Origem** $N = 500$ e **Dimensão de Imersão** $k = 3$

Algoritmos	Normas	Teste 1	Teste 2	Teste 3	Teste 4	Teste 5
Análise de Coordenadas Principais	Norma-2	3,3917e+04	3,7085e+04	3,4026e+04	3,4577e+04	3,5723e+04
	Norma-4	1,1607e+04	1,2836e+04	1,1157e+04	1,1709e+04	1,1837e+04
	Máximo	0,6174e+04	0,7014e+04	0,6105e+04	0,5622e+04	0,5626e+04
Branch-and-Prune Modificado	Norma-2	0,0349e+04	0,0354e+04	0,0364e+04	0,0351e+04	0,0348e+04
	Norma-4	0,0247e+04	0,0247e+04	0,0249e+04	0,0246e+04	0,0248e+04
	Máximo	0,0092e+04	0,0093e+04	0,0093e+04	0,0090e+04	0,0090e+04

Tabela 7 – Comparação entre os métodos. 20 pontos gerados em $\mathbb{R}^{500}$

- **Dimensão de Origem** $N = 1000$ e **Dimensão de Imersão** $k = 3$

Algoritmos	Normas	Teste 1	Teste 2	Teste 3	Teste 4	Teste 5
Análise de Coordenadas Principais	Norma-2	1,0991e+05	1,0033e+05	1,0386e+05	1,0660e+05	1,0216e+05
	Norma-4	4,7893e+04	4,3779e+04	4,5476e+04	4,6649e+04	4,3813e+04
	Máximo	2,6382e+04	2,4508e+04	2,5950e+04	2,4297e+04	2,4630e+04
Branch-and-Prune Modificado	Norma-2	0,0344e+04	0,0344e+04	0,0338e+04	0,0346e+04	0,0338e+04
	Norma-4	0,0334e+04	0,0376e+04	0,0378e+04	0,0332e+04	0,0334e+04
	Máximo	0,0173e+04	0,0266e+04	0,0267e+04	0,0170e+04	0,0175e+04

Tabela 8 – Comparação entre os métodos. 10 pontos gerados em $\mathbb{R}^{1000}$

norma- 2, $3,4057$ para norma-4 e $2,0907$ para norma do máximo. Podemos notar que há uma diferença significativa entre os métodos, ou seja, os resultados mostram que o algoritmo *branch-and-prune* modificado conserva melhor as distâncias originais permitindo uma visualização mais precisa de sua estrutura. Mais especificamente, com relação ao aumento da dimensão de origem e aumento do número de pontos, houve uma redução significativa do erro quando comparado com o método análise de coordenadas principais (ACP).

Na Figura 20, tem-se um gráfico contendo os perfis de desempenho (DOLAN E. D.; MORE, 2002) do BPM e ACP. Os perfis de desempenho são uma técnica poderosa na comparação do desempenho de *softwares*. Para a criação deste gráfico, um mesmo conjunto de problemas foi resolvido pelo BPM e ACP, medindo-se erros na solução. A partir destes dados, calculou-se as razões de desempenho e, então, os perfis de desempenho. A razão de desempenho mostra o comportamento de um *software* na resolução de um determinado problema.

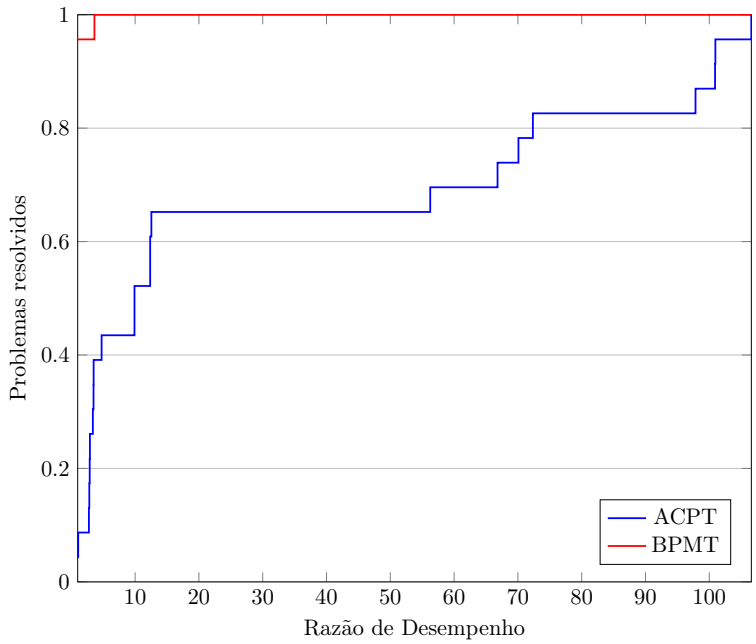

Figura 20 – Perfis de desempenho para ACP e BPM.

Observando a curva do perfil desempenho BPM, vê-se que ele obteve melhor desempenho que o ACP, em quase 100% dos problemas.

Com estes resultados, pode-se concluir que quando o objetivo principal da redução dimensional é preservar as distâncias, permitindo sua vizualização, para problemas de pequeno e médio porte, o método implementado (BPM) comparado como método utilizado atualmente (ACP) apresenta bons resultados.

5.3 Exemplo

O escalonamento multidimensional é uma forma de construir pontos utilizando apenas distâncias. De uma forma generalizada, estas distâncias podem ser dissimilaridades ou similaridades. Neste caso, o objetivo é visualizar informações que não são pontos no

sentido usual. As variáveis descritas no livro *Modern Multidimensional Scaling: Theory and Applications* (BORG; GROENEN, 2010) são um exemplo. Neste livro, encontramos o exemplo apresentado na seção 1.6: um resumo estatístico de 1970, emitido pelo *Bureau of the Census*, que fornece dados sobre a taxa de diferentes crimes em 50 estados dos E.U.A. (Wilkinson, 1990), ver Figura 1.1.

Crime	No.	1	2	3	4	5	6	7
Assassinato	1	1,00	0,52	0,34	0,81	0,28	0,06	0,11
Estupro	2	0,52	1,00	0,55	0,70	0,68	0,60	0,44
Latrocínio	3	0,34	0,55	1,00	0,56	0,62	0,44	0,62
Assalto	4	0,81	0,70	0,56	1,00	0,52	0,32	0,33
Roubo	5	0,28	0,68	0,62	0,52	1,00	0,80	0,70
Furto	6	0,06	0,60	0,44	0,32	0,80	1,00	0,55
Roubo de Carros	7	0,11	0,44	0,62	0,33	0,70	0,55	1,00

Tabela 9 – Correlações entre índices de criminalidade em 50 estados dos EUA.

Neste caso, temos correlações entre índices de criminalidade em 50 estados dos EUA. Estas medidas de similaridades podem ser transformadas em medidas de dissimilaridades, pois todas as correlações são positivas (BORG; GROENEN, 2010). Neste caso, usaremos, $D = 1 - S$, onde

$$D = \begin{bmatrix} 0 & 0.48 & 0.66 & 0.19 & 0.72 & 0.94 & 0.89 \\ 0.48 & 0 & 0.45 & 0.30 & 0.32 & 0.40 & 0.56 \\ 0.66 & 0.45 & 0 & 0.44 & 0.38 & 0.56 & 0.38 \\ 0.19 & 0.30 & 0.44 & 0 & 0.48 & 0.68 & 0.67 \\ 0.72 & 0.32 & 0.38 & 0.48 & 0 & 0.20 & 0.30 \\ 0.94 & 0.40 & 0.56 & 0.68 & 0.20 & 0 & 0.45 \\ 0.89 & 0.56 & 0.38 & 0.67 & 0.30 & 0.45 & 0 \end{bmatrix}.$$

Aqui utilizamos o comando do *Matlab* que efetua uma análise de coordenadas principais (comando `cmdscale`), para obter visualização. A saída do `cmdscale` é

```
[Y,eigvals] = cmdscale(D),
```

$$\mathtt{Y} = \begin{bmatrix} 0.5458 & 0.0048 & -0.0491 & 0.0088 \\ 0.0726 & -0.1775 & 0.0206 & 0.0717 \\ -0.0463 & 0.2435 & 0.1433 & 0.0019 \\ 0.3002 & -0.0043 & 0.0058 & -0.0457 \\ -0.1858 & -0.0534 & -0.0510 & -0.0614 \\ -0.3629 & -0.2201 & 0.0452 & -0.0085 \\ -0.3237 & 0.2069 & -0.1147 & 0.0332 \end{bmatrix}$$

e

$$\texttt{eigvals} = \begin{bmatrix} 0.6664 \\ 0.1850 \\ 0.0412 \\ 0.0122 \\ 0.0000 \\ -0.0076 \\ -0.0351 \end{bmatrix}.$$

A primeira saída do `cmdscale`, `Y`, é uma matriz de pontos que pode ter até oito dimensões (colunas de Y). A visualização das distâncias depende da utilização dos pontos em apenas duas ou três dimensões. A segunda saída do `cmdscale`, `eigvals`, é um conjunto de autovalores ordenados cuja magnitudes relativa indica quantas dimensões pode-se usar com segurança. Se apenas os primeiros dois ou três autovalores são grandes em relação aos demais, então apenas aquelas coordenadas dos pontos em `Y` são necessárias para reproduzir com precisão D.

Na Figura 21, comparamos as correlações apresentadas pelo comando `cmdscale` e pelo algoritmo BP modificado, em um espaço com duas dimensões.

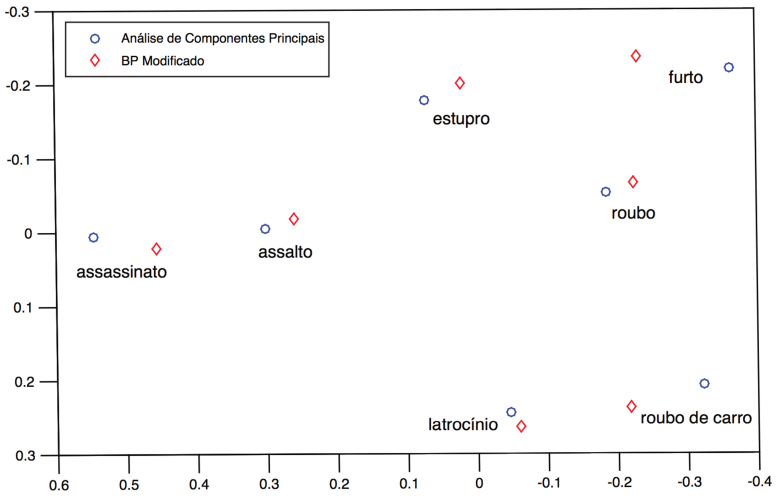

Figura 21 – Comparação entre análise de coordenadas principais e BP modificado.

Como podemos observar, os dados apresentados por ambos os métodos apresentam resultados satisfatórios, ou seja, erros pequenos.

No próximo capítulo, comentaremos as principais conclusões deste trabalho.

6 Conclusões e Trabalhos Futuros

Neste trabalho, foi proposta uma nova abordagem para resolver uma classe de problemas de escalonamento multidimensional, representando medidas de proximidade entre pares de objetos como distâncias entre pontos em um espaço geométrico, de modo que as distâncias estejam o máximo possível relacionadas com as proximidades entre os objetos. Nesta classe de problemas, a prioridade foi manter a estrutura objetivando visualização de dados em $\mathbb{R}^3$.

Esta nova proposta teve como base o algoritmo *Branch-and-Prune* inicialmente utilizado para obter estruturas moleculares de proteínas a partir das distâncias conhecidas, determinando as posições destes átomos em $\mathbb{R}^3$. Para a utilização deste algoritmo em $\mathbb{R}^m$, originalmente proposto em $\mathbb{R}^3$, foram feitas adaptações, dando origem a um algoritmo *Branch-and-Prune* modificado.

O algoritmo *Branch-and-Prune* modificado determina a melhor solução dentre as possíveis soluções. Para evitar uma busca exaustiva, a poda do algoritmo *Branch-and-Prune* modificado pôde ser aprimorada, evitando o cálculo de uma grande quantidade de pontos infactíveis.

Os resultados computacionais obtidos ao comparar a composição estrutural oriundas tanto do algoritmo *Branch-and-Prune* modificado quanto da técnica *análise de coordenadas principais* foram satisfatórios em problemas de pequeno e médio porte, sendo possível observar uma redução significativa dos erros ao trabalhar com o algoritmo modificado. Isto ocorre devido às expectativas da técnica análise de coordenadas principais. A expectativa é que os primeiros autovalores tenham magnitude suficientemente alta de modo que quando comparados com os demais estes tornam-se supérfulos.

Além disto, estes resultados nos motivam a continuar com esta linha de pesquisa estendendo esta abordagem a outras instâncias do problema de escalonamento multidimensional. Um trabalho futuro seria priorizar a ordenação dos pontos com base nos autovalores da matriz de distâncias. Uma implementação mais refinada pode classificar os objetos de estudo em subgrupos com as mesmas características além de permitir aplicações em problemas de médio porte. Uma implementação em linguagem C pode permitir uma comparação com relação ao tempo computacional. Um trabalho futuro também seria obter resultados para comparação com o método análise de componentes principais.

Referências

AHUJA, A.; MAGNANTI, T.; ORLIN, J. *Network Flows: Theory, Algorithms and Applications*. [S.l.]: Prentice-Hall, 1993.

BANG-JENSEN J., G. G. Y. A. When the greedy algorithm fails. *Discrete Optimization*, v. 1, p. 121–127, 2004.

BLASIUS J., G. M. G. P. . V. M. v. d. Special issue on correspondence analysis and related methods. *Computational Statistics and Data Analysis*, v. 53, p. 3103–3106, 2009.

BORG, I.; GROENEN, P. *Modern Multidimensional Scaling: Theory and Applications*. [S.l.]: Springer, 2010.

CAILLIEZ, F. The analytical solution of the additive constant problem. *Psychometrika*, v. 48, p. 305–308, 1983.

CARROLL, J. D.; ARABIE, P. Multidimensional scaling. *Annual Review of Psychology*, v. 31, p. 607–649, 1980.

COOPE, I. D. Reliable computation of the points of intersection of n spheres in $\mathbb{R}^n$. *Australian and New Zealand Industrial and Applied Mathematics Journal (ANZIAM)*, v. 42, p. 461–477, 2000.

CORMEN, T.; LEISERSON, C.; RIVEST, R. *Introduction to Algorithms*. [S.l.]: MIT Press, 1990.

COXON, A. P. M. *The user's guide to multidimensional scaling*. London: Heinemann Education Books, 1982.

DATTORRO, J. *Convex Optimization & Euclidean Distance Geometry*. [S.l.]: Meboo, 2005.

DAVISON, M. L. *Multidimensional scaling*. New York: John Wiley, 1983.

DENARDO, E.; FOX, B. Shortest route methods: reaching, pruning and buckects. *Operations Research*, v. 27, p. 161–186, 1979.

DEO, N.; PANG, C. Shortest paths algorithms: taxonomy and annotation. *Networks*, 1979.

DIAL, R.; GLOVER, G.; KARNEY, D.; KLINGMAN, D. A computational analysis of alternative algorithms and labeling techniques for finding shortest path trees. *Networks*, v. 9, p. 215–248, 1979.

DIJKSTRA, E. W. A note on two problems in connexion with graphs. *Numerische Mathematik*, v. 1, n. 1, p. 269–271, 1959.

DOLAN E. D.; MORE, J. J. Benchmarking optimization software with performance profiles. *Mathematical programming*, v. 91, p. 201–213, 2002.

DONG Q., W. Z. A linear-time algorithm for solving the molecular distance geometry problem with exact inter-atomic distances. *J. Glob. Optim.*, v. 22, p. 365–375, 2002.

ECKART, C.; YOUNG, G. Approximation of one matrix by another of lower rank. *Psychometrika*, v. 1, p. 211–218, 1936.

EVEN, S. *Graph Algorithms*. [S.l.]: Cambridge University Press, 2011.

EYRING, H. The resultant electric moment of complex molecules. *Phys. Rev.*, American Physical Society, v. 39, p. 746–748, Feb 1932. Disponível em: ⟨http://link.aps.org/doi/10.1103/PhysRev.39.746⟩.

GOLDSTEIN, H. *Classical Mechanics*. [S.l.]: Addison-Wesley Publishing Company, 1922.

GOLUB, G. H.; LOAN, C. F. V. *Matrix Computations 2nd Edition*. Baltimore, Maryland: The Johns Hopkins University Press, 2013.

GOWER, J. C. Some distance properties of latent root and vector methods used in multivariate analysis. *Biometrika*, v. 53, p. 325–338, 1966.

GOWER, J. C.; DIJKSTERHUIS, G. B. *Procrustes Problems*. Oxford: Oxford University Press, 2004.

JONGMAN, R. H. G.; BRAAK, C. J. F. T.; TONGEREN, O. F. R. V. *Data analysis in community and landscape ecology*. Cambridge: Combridge Univ. Press, 1995.

KREYSZIG, E. *Introductory Functional Analysis with Applications*. [S.l.]: Wiley, 1989.

KRUSKAL, J. B. Multidimensional scaling by optimizing goodness of fit to a nonmetric hypothesis. *Psychometrika*, v. 29, p. 1–27, 1964.

KSHIRSAGAR, A. M. *Multivariate Analysis*. New York: Marcel Dekker, 1972.

KUMAR, V. Algorithms for constraint satisfaction problems: A survey. *AI Magazine*, v. 13, n. 1, p. 32–44, 1992.

LAVOR, C.; LIBERTI, L.; MACULAN, N.; MUCHERINO, A. A branch-and-prune algorithm for the molecular distance geometry problem. *International Transactions in Operational Research*, v. 15, p. 1–17, 2008.

_____. Molecular distance geometry methods: From continuous to discrete. *International Transactions in Operational Research*, v. 18, p. 33–51, 2011.

_____. The discretizable molecular distance geometry problem. *Comp. Opt. and Appl.*, v. 52, n. 1, p. 115–146, 2012. Disponível em: ⟨http://dblp.uni-trier.de/db/journals/coap/coap52.html#LavorLMM12⟩.

_____. Euclidean distance geometry and applications. *SIAM Review*, v. 56, p. 3–69, 2014.

LEEUW, J.; HEISER, W. Theory of multidimensional scaling. *Handbook of statistics*, v. 2, p. 285–316, 1982.

MANLY, B. F. J. *Multivariate statistical methods*. London: Chapman Hall, 1994.

MARDEN, J. I. *Analyzing and Modeling Rank Data*. London: Chapman Hall, 1995.

MATHWORKS. Http://www.mathworks.com. Accessed: 24-09-2014.

NEI, M. The theory of genetic distance and evolution of human races. *Japanese Journal of Human Genetics*, v. 23, p. 341–369, 1978.

NETTO, P. O. B. *Teoria e Modelos de Grafos*. [S.l.]: E. Blucher, 1979.

PASQUALI, L. *Instrumentos psicológicos: Manual prático de elaboração*. [S.l.]: LabPAM., 1999.

POGORELOV, A. *Geometry*. [S.l.]: Mir Publishers, 1987.

SCHIFFMAN, S. S.; REYNOLDS, M. L.; YOUNG, F. W. *Introduction to Multidimensional Scaling: Theory, methods and applications*. [S.l.]: Academic Press, 1981.

SCHOLTEN, M.; CALDEIRA, P. Z. O senso do escalonamento multidimensional. *Analise Psicologica*, p. 63–85, 1997.

SHEPARD, R. N. The analysis of proximities: multidimensional scaling with an unknown distance function. *Psychometrika*, v. 27, p. 125–139, 1962.

SILVA, J. A. D.; FILHO, N. P. R. *Avaliação e mensuração de dor: pesquisa, teoria e prática*. [S.l.]: FUNPEC, 2006.

SINGAL P., C. R. S. Dijkstra shortest path algorithm using global positioning system. *International Journal of Computer Applications*, v. 101, p. 012–018, 2014.

STEYVERS, M. *Multidimensional scaling*. Stanford: Encyclopedia of cognitive science, 2002.

SUáREZ, M. H.; HERNáNDEZ, A. I. M.; GALDóN, B. R.; RODRíGUEZ, L. H. Application of multidimensional scaling technique to differentiate sweet potato (ipomoea batatas (l.) lam) cultivars according to their chemical composition. *Journal of Food Composition and Analysis*, v. 46, p. 43 – 49, 2016.

TORGERSON, W. S. Multidimensional scaling: I. theory and method. *Psychometrika*, v. 17, p. 401–419, 1952.

_____. *Theory and methods of scaling*. New York: Wiley, 1958.

XIAO-YAN L., Y.-L. C. Application of dijkstra algorithm in logistics distribution lines. *Proceedings of the Third International Symposium on Computer Science and Computational Technology*, p. 048–050, 2010.

YOUNG, G.; HOUSEHOLDER, A. S. Discussion of a set of point in terms of their mutual distances. *Psychometrika*, v. 3, p. 19–22, 1938.