Notas em Matemática Aplicada

ISSN 2175-3385
E-ISSN 2236-5915

Volume 92, 2021

Corpo Editorial

Sandra Mara Cardoso Malta (Editor Chefe)
Laboratório Nacional de Computação Científica - LNCC
Petrópolis, RJ, Brasil

Eduardo V. O. Teixeira (Editor Executivo)
University of Central Florida - UCF
Orlando, FL, EUA

Lilian Markenzon
Universidade Federal do Rio de Janeiro - UFRJ
Rio de Janeiro, RJ, Brasil

Marcelo Sobottka
Universidade Federal de Santa Catarina - UFSC
Florianópplis, SC, Brasil

Paulo F. de Arruda Mancera
Universidade Estadual Paulista Júlio de Mesquita Filho- UNESP
Botucatu, SP, Brasil

Sandra Augusta Santos
Universidade Estadual de Campinas - UNICAMP
Campinas, SP, Brasil

Tânia Schmitt
Universidade de Brasília - UnB
Brasília, DF, Brasil

Sociedade Brasileira de Matemática Aplicada e Computacional

2021

A Sociedade Brasileira de Matemática Aplicada e Computacional - SBMAC publica, desde as primeiras edições do evento, monografias dos cursos que são ministrados nos CNMAC.

Para a comemoração dos 25 anos da SBMAC, que ocorreu durante o XXVI CNMAC em 2003, foi criada a série **Notas em Matemática Aplicada** para publicar as monografias dos minicursos ministrados nos CNMAC, o que permaneceu até o XXXIII CNMAC em 2010.

A partir de 2011, a série passa a publicar, também, livros nas áreas de interesse da SBMAC. Os autores que submeterem textos à série Notas em Matemática Aplicada devem estar cientes de que poderão ser convidados a ministrarem minicursos nos eventos patrocinados pela SBMAC, em especial nos CNMAC, sobre assunto a que se refere o texto.

O livro deve ser preparado em **Latex (compatível com o Miktex versão 2.9), as figuras em eps** e deve ter entre **80 e 150 páginas**. O texto deve ser redigido de forma clara, acompanhado de uma excelente revisão bibliográfica e de **exercícios de verificação de aprendizagem** ao final de cada capítulo.

Veja todos os títulos publicados nesta série na página

http://www.sbmac.org.br/p_notas.php

Sociedade Brasileira de Matemática Aplicada e Computacional

2021

PROCESSAMENTO DE SINAIS SOBRE GRAFOS: FUNDAMENTOS E APLICAÇÕES

Juliano Bandeira Lima
juliano.lima@ufpe.br

Guilherme Boaviagem Ribeiro
guilherme.boaviagem@gmail.com

Departamento de Eletrônica e Sistemas
Centro de Tecnologia e Geociências
Universidade Federal de Pernambuco

Wallace Alves Martins[1,2]
wallace.martins@smt.ufrj.br

Vitor Rosa Meireles Elias[1]
vtrmeireles@poli.ufrj.br

Gabriela Lewenfus[1]
gabriela.lewenfus@gmail.com

[1]Departamento de Engenharia Eletrônica e de Computação
Escola Politécnica
Programa de Engenharia Elétrica - Coppe
Universidade Federal do Rio de Janeiro

[2]Signal Processing and Satellite Communications group, SIGCOM
Interdisciplinary Centre for Security, Reliability and Trust, SnT
Universidade de Luxemburgo

Sociedade Brasileira de Matemática Aplicada e Computacional

São Carlos - SP, Brasil
2021

Coordenação Editorial: Mateus Bernardes

Coordenação Editorial da Série: Sandra M. C. Malta

Editora: SBMAC

Capa: Matheus Botossi Trindade

Patrocínio: SBMAC

Copyright ©2021 by Juliano Bandeira Lima, Guilherme Boaviagem Ribeiro, Wallace Alves Martins, Vitor Rosa Meireles Elias e Gabriela Lewenfus. Direitos reservados, 2021 pela SBMAC. A publicação nesta série não impede o autor de publicar parte ou a totalidade da obra por outra editora, em qualquer meio, desde que faça citação à edição original.

Catalogação elaborada pela Biblioteca do IBILCE/UNESP

Bibliotecária: Maria Luiza Fernandes Jardim Froner

Lima, Juliano B.
Processamento de Sinais sobre Grafos: Fundamentos e Aplicações - São Carlos, SP : SBMAC, 2021, 98 p., 21,5 cm - (Notas em Matemática Aplicada; v. 92)

ISBN 978-65-86388-05-3 e-ISBN 978-65-86388-04-6

1. Processamento de Sinais 2. Grafos 3. Processamento de Sinais sobre Grafos I. Lima, Juliano B. II. Ribeiro, Guilherme B. III. Martins, Wallace A. IV. Elias, Vitor R. M. V. Lewenfus, Gabriela . VI. Título. VII. Série

CDD - 51

Juliano dedica este trabalho a sua esposa, Ana Raphaela, e a seus filhos, José Bento e Joaquim.

Guilherme dedica este trabalho a sua esposa, Bárbara, e a seus filhos, Rafael e Gianna, que são força e doçura em seu caminho.

Wallace dedica este trabalho a sua esposa, Claudia, e a suas filhas, Anne e Rebecca.

Vitor dedica este trabalho a seus pais, Sandra e Roberto.

Gabriela dedica este trabalho a seus pais, Suely e Israel.

Agradecimentos

Este livro é fruto de uma colaboração entre membros da Universidade Federal de Pernambuco (UFPE) e da Universidade Federal do Rio de Janeiro (UFRJ). Essa colaboração acadêmica iniciou-se no XXXV Simpósio Brasileiro de Telecomunicações e Processamento de Sinais (SBrT-2017), São Pedro, SP, e evoluiu para uma amizade que transcende o ambiente acadêmico e dá mais vigor a projetos como este.

O apoio do Departamento de Eletrônica e Sistemas (DES) e do Programa de Pós-graduação em Engenharia Elétrica (PPGEE) da UFPE, do Departamento de Engenharia Eletrônica e de Computação (DEL) da Escola Politécnica (Poli) da UFRJ e do Programa de Engenharia Elétrica (PEE) da COPPE foram fundamentais para a conclusão deste trabalho.

Somos especialmente gratos aos membros do Grupo de Pesquisa em Processamento Digital de Sinais da UFPE e a todos os alunos, funcionários e colegas do Laboratório de Sinais Multimídia e Telecomunicações (SMT) da COPPE/Poli/UFRJ, especialmente aos professores Paulo Sergio Ramirez Diniz, Luiz Wagner Pereira Biscainho, Markus Vinicius Santos Lima e Tadeu Nagashima Ferreira (este último da Universidade Federal Fluminense, UFF).

Agradecemos especialmente aos nossos familiares e amigos que nos deram todo incentivo e suporte necessários para a conclusão deste trabalho.

Agradecemos também o constante incentivo por parte de nossa editora, Profa. Sandra Mara Cardoso Malta, e as revisões técnicas por parte dos revisores anônimos. Por fim, o apoio financeiro de CNPq, CAPES e FAPERJ foram imprescindíveis para o término do livro.

Conteúdo

Prefácio **xiii**

1 Teoria dos Grafos: Apresentando a Terminologia **1**

1.1 Definindo o sinal e seu domínio 2

1.2 Inferindo grafos 4

1.3 Problemas 4

2 $\mathbf{GSP_A}$: Grafos e o Processamento Algébrico de Sinais **7**

2.1 Filtros sobre grafos 8

2.2 A transformada de Fourier sobre grafos 8

2.3 O domínio da frequência 9

2.4 Problemas 12

3 $\mathbf{GSP_L}$: A Teoria Espectral de Grafos em GSP **15**

3.1 O operador diferença e a GFT 17

3.2 Filtragem 18

3.3 Problemas 20

4 Amostragem e Reconstrução de Sinais sobre Grafos **23**

4.1 Limitação de faixa e amostragem 23

4.2 Estimação adaptativa de sinais sobre grafos 25

4.3 Seleção do conjunto de amostragem 27

4.4 Recuperação prescindindo do suporte do sinal na frequência 28

4.5 Problemas 28

5 Análise Vértice-Frequência **31**

5.1 Transformada de Fourier sobre grafos com janelamento 31

5.1.1 Translação em grafos 32

5.1.2 Modulação em grafos 33

5.1.3 Átomos e espectrograma da WGFT 33

5.1.4 Exemplos numéricos 34
5.2 Transformada *wavelet* em grafos 37
5.2.1 Transformada *wavelet* espectral em grafos 39
5.2.2 Reconstrução 39
5.2.3 *Kernels* para *wavelets* 40
5.2.4 Aproximação por polinômios de Chebyshev 41
5.2.5 Computação Distribuída 42
5.3 SGWT adaptada ao espectro 42
5.3.1 Experimento numérico 45
5.4 Problemas 46

6 Exemplos e Aplicações 47
6.1 Remoção de ruído em rede de sensores 47
6.2 Estimação de sinais climáticos 48
6.3 Compressão de *light fields* 52
6.4 Outras áreas de aplicação 55
6.5 Problemas 56

7 Implementação em *Software* e Oportunidades de Estudo 59
7.1 As ferramentas típicas da comunidade 59
7.2 Implementando GSP em Python 60
7.2.1 Métodos básicos de PyGSP 61
7.2.2 Caso prático: encontrando um filtro passa-baixas LSI via mínimos quadrados 64
7.3 Algumas oportunidades de estudo 68
7.4 Problemas 69

Prefácio

Informações associadas a estruturas em rede são a todo instante geradas, armazenadas e processadas em sistemas relacionados às diversas áreas da engenharia e da tecnologia. Dados coletados por *smartphones*, medições em conjuntos de sensores ou de dispositivos de Internet das Coisas (IoT), números de citações em uma rede de colaboração científica ou relações em mídias sociais, atributos de objetos virtuais 3D, níveis de interação entre indivíduos em um ecossistema e sinais que se propagam ao longo de conexões cerebrais são exemplos dessas informações.

Em cenários como os mencionados, as estruturas em rede podem ser modeladas como grafos, a cujos vértices se encontrem associadas amostras de sinais que representam alguma informação. Num conjunto de sensores de temperatura espalhados por determinada região, por exemplo, os nós do grafo representariam os sensores (respeitando suas posições geográficas) e as amostras do sinal sobre esses nós assumiriam valores correspondentes às temperaturas medidas pelos sensores; cada par de sensores (nós) estaria conectado por uma aresta ponderada por um valor inversamente proporcional à distância geográfica entre eles.

Nesse escopo, a principal particularidade é que as redes em questão podem ter topologias arbitrárias; isso não acontece quando se considera um sinal que varia ao longo de um eixo de tempo discreto sobre o qual as respectivas amostras são dispostas de modo equiespaçado, ou uma imagem digital cujos *pixels* são posicionados, também equiespaçadamente, sobre uma grade retangular num plano. Então, como processar e analisar sinais e informações distribuídas sobre essas redes? A busca por uma resposta a essa pergunta impulsionou, na última década, o surgimento do processamento de sinais sobre grafos (GSP, da expressão em inglês *graph signal processing*). Com o GSP, visa-se à extensão de métodos clássicos de processamento de sinais para casos em que o domínio é *irregular*, o que, no presente contexto, significa dizer que as amostras dos sinais em questão podem ser dispostas sobre vértices de um grafo arbitrariamente posicionados.

A comunidade científica envolvida com GSP tem crescido vertiginosamente, o que se pode inferir pela grande quantidade e alta qualidade dos trabalhos que têm sido publicados sobre o assunto. Isso tem proporcionado vários avanços na área, a qual lança mão de conceitos relacionados à teoria espectral de grafos, álgebra linear, análise de Fourier, filtragem, otimização, aprendizado de máquina etc. Por outro lado, embora uma gama de ferramentas teóricas tenha sido introduzida e esteja sendo aplicada com sucesso à resolução de problemas práticos, o GSP como um todo ainda não alcançou a maturidade em seu desenvolvimento, possuindo várias ambiguidades, lacunas e restrições de ordem prática.

A diversidade de cenários enquadráveis no escopo descrito e o grande número

de problemas em aberto a respeito do GSP justificam a atenção que a temática tem recebido e ratificam a importância que possui a disseminação de seus conceitos e perspectivas. A presente obra tem o intuito de contribuir com tal divulgação, oferecendo aos estudantes e profissionais interessados uma visão ampla dos fundamentos do GSP e fornecendo exemplos de aplicações que ilustram seu potencial de uso prático. Este livro vem preencher uma lacuna na literatura em língua portuguesa a respeito do tema GSP, uma vez que, até onde os autores têm conhecimento, o assunto tem sido abordado em alguns livros disponíveis apenas na língua inglesa. Vale salientar, ainda, que um material preliminar sobre GSP foi produzido pelos quatro primeiros autores para dar suporte a um minicurso ministrado por eles no XXXVI Simpósio Brasileiro de Telecomunicações e Processamento de Sinais, ocorrido em 2018, na cidade de Campina Grande / PB. O referido material foi documentado na forma de um capítulo no livro de minicursos do evento, o qual possui alguma superposição com a presente obra. Mais especificamente, como ponto de partida para estruturação deste livro, converteu-se a estrutura de seções do mencionado capítulo numa estrutura de capítulos, com algumas fusões de mais de uma seção (do capítulo) num mesmo capítulo (do livro). Os capítulos resultantes foram, então, revistos e expandidos, de modo a privilegiar uma apresentação mais didática de cada ponto; mais detalhes sobre determinados assuntos foram contemplados, trechos do texto original foram ajustados, novos exemplos foram incluídos etc. Além disso, foi formulado um capítulo completamente novo, que trata do importante campo da análise vértice-frequência. Finalmente, foram elaborados problemas que são apresentados ao final de cada capítulo; alguns desses problemas envolvem simulações computacionais e outros conduzem o leitor a realizar demonstrações de resultados não tratados explicitamente no texto, mas que guardam relevância aos interessados em se aprofundar no processamento de sinais sobre grafos.

Pretende-se proporcionar aos leitores deste livro a possibilidade de (i) identificar o contexto em que o GSP se encontra inserido, distinguindo os conhecimentos relevantes para o desenvolvimento da temática e reconhecendo as interfaces com áreas de pesquisa correlatas, (ii) conhecer estratégias para modelamento de estruturas em rede por meio de grafos, considerando questões relacionadas, por exemplo, à inferência de topologias, à atribuição de pesos às arestas e à dinâmica na forma como os vértices se distribuem, (iii) compreender as especificidades das principais operações aplicáveis a sinais sobre grafos (transformada de Fourier, deslocamento, filtragem, amostragem, estimação, análise vértice-frequência etc.), (iv) reconhecer a aplicabilidade do GSP em cenários relacionados a estimação / amostragem adaptativa de sinais em grafos, compressão em *light-fields*, compressão de imagens, detecção de comunidades e de nós com atividade anômala em certos tipos de redes, e (v) utilizar ferramentas computacionais básicas para realização de experimentos em processamento de sinais sobre grafos. Este texto deve oferecer uma leitura mais fluida aos que possuem conhecimentos básicos do processamento de sinais clássico. De toda forma, por meio de explicações sucintas e de notas de rodapé, os autores procuraram auxiliar mesmo o leitor menos familiarizado com o tema, direcionando-o aos conceitos mais importantes para o entendimento do conteúdo apresentado.

Este livro se encontra organizado como descrito a seguir. No Capítulo 1, são apresentados de forma sucinta os principais fundamentos e termos relacionados à teoria dos grafos; o objetivo é que mesmo o leitor com pouco ou nenhum contato com o assunto possa ter facilitado o seu percurso ao longo das seções que tratam especificamente do GSP. Ainda no capítulo inicial, é exposto o conceito de sinal sobre grafo e são abordadas, de forma preliminar, questões relativas à inferência de

um grafo subjacente a um sinal.

Os Capítulos 2 e 3 contemplam os fundamentos das duas principais abordagens para o processamento de sinais sobre grafos. No Capítulo 2, é apresentado o que nesta obra é denotado por $\mathrm{GSP_A}$, que corresponde ao GSP desenvolvido a partir da teoria do processamento algébrico de sinais; são apresentados pontos relativos à filtragem e à transformada de Fourier sobre grafos (GFT, do termo em inglês *graph Fourier transform*), e discutidas questões a respeito do domínio da frequência nesse contexto. No Capítulo 3, é apresentado o $\mathrm{GSP_L}$, que se baseia na teoria espectral de grafos, e são discutidas, novamente, à luz dessa abordagem, questões ligadas à filtragem e à GFT.

Em seguida, são apresentados capítulos que abordam temas de maior complexidade em GSP e cuja compreensão sugere a necessidade do conhecimento, por parte do leitor, dos pontos contemplados nos capítulos iniciais do livro. Mais especificamente, no Capítulo 4, trata-se da amostragem e da reconstrução de sinais sobre grafos, temáticas que envolvem a extensão ao GSP de uma série de conceitos e operações consolidadas no processamento de sinais clássico: limitação de faixa, estimação adaptativa e seleção do conjunto de amostragem. No Capítulo 5, aborda-se a análise vértice-frequência, que seria análoga à bem conhecida análise tempo-frequência. Este capítulo é iniciado com a exposição da transformada de Fourier sobre grafos com janelamento, a qual inclui os conceitos de translação e modulação em grafos, bem como a representação de sinais sobre grafos por meio de espectrogramas. Em seguida, apresenta-se a transformada *wavelet* em grafos, contemplando pontos como reconstrução e *kernels* para *wavelets*, além de questões computacionais relativas ao tema. As última seção do capítulo aborda a transformada *wavelet* espectral em grafos (SGWT) adaptada ao espectro.

Os dois últimos capítulos deste livro são relacionados às áreas de aplicação do GSP e a questões relativas à sua implementação em *software*. No Capítulo 6, a teoria apresentada nos capítulos anteriores do livro é empregada à remoção de ruído em rede de sensores, à estimação de sinais climáticos e à compressão de *light fields*. Outras áreas de aplicação são mencionadas. No Capítulo 7, o foco se volta à implementação em Python das ferramentas do GSP; métodos básicos de PyGSP são apresentados e um caso prático é discutido. Por fim, algumas oportunidades de estudo e problemas de GSP abertos à investigação são sugeridos.

Recife, 20 de setembro de 2021.

Juliano B. Lima

Guilherme B. Ribeiro

Rio de Janeiro, 20 de setembro de 2021.

Wallace A. Martins

Vitor R. M. Elias

Gabriela Lewenfus

Capítulo 1

Teoria dos Grafos: Apresentando a Terminologia

Um *grafo* é definido em sua forma mais geral como sendo o par $(\mathcal{V}, \mathcal{E})$, onde $\mathcal{V}$ é um conjunto cujos elementos chamam-se *vértices* e $\mathcal{E}$ (conjunto das *arestas*) é um subconjunto de $\mathcal{V}^2$. Para os propósitos de processamento de sinais sobre grafos (GSP), convém definir um grafo como sendo a estrutura $\mathcal{G} = \{\mathcal{V}, \mathbf{A}\}$ cujas relações entre os $|\mathcal{V}| = N$ vértices são capturadas pelas entradas da matriz de adjacência ponderada $\mathbf{A}$: se $A_{i,j} \neq 0$, então o vértice $v_j \in \mathcal{V}$ influencia (ou seja, é *adjacente*) o vértice $v_i \in \mathcal{V}$ e se diz que há uma aresta de v_j para v_i com peso $A_{i,j}$. Para o escopo deste livro, é suficiente considerar apenas os casos em que $A_{i,j} \in \mathbb{R}$. Por sua própria definição, $\mathbf{A}$ é uma matriz quadrada de dimensões $N \times N$. A soma dos pesos das arestas incidentes em um vértice v_i é o *grau de entrada* de v_i, ou d_i^-; a soma dos pesos das arestas que partem de v_i é o *grau de saída* d_i^+.

O grafo é *direcionado* (ou *dirigido*, *orientado*, ou simplesmente um *digrafo*) se, e somente se, a matriz $\mathbf{A}$ não é simétrica. Caso contrário, o grafo é dito *não-direcionado*, e o *grau* do vértice v_i é definido como $d_i = d_i^+ = d_i^-$. A Fig. 1.1 mostra exemplos de grafos direcionado e não-direcionado.

A *matriz Laplaciana* de um grafo é dada por $\mathbf{L} = \mathbf{D} - \mathbf{A}$, com $\mathbf{D}$ sendo a *matriz de grau* do grafo, uma matriz diagonal com a i-ésima entrada igual ao grau do vértice v_i.

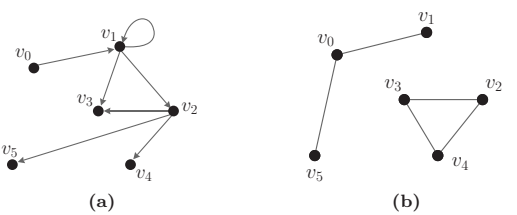


Figura 1.1: Exemplos de grafos (a) direcionado e (b) não-direcionado, definidos sobre o mesmo conjunto de vértices.

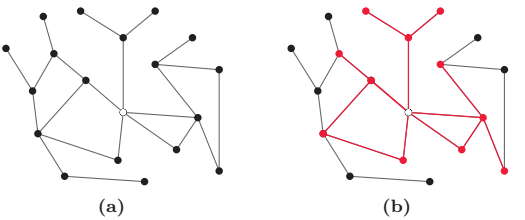


Figura 1.2: (a) Um grafo; (b) em vermelho, o conjunto $\mathcal{N}(i,2)$ em que v_i é o vértice branco.

Chama-se *laço* uma aresta que *parte de* e *chega a* um mesmo vértice. Fala-se em *múltiplas arestas* sempre que a um único par de vértices estão associadas duas ou mais arestas. Um grafo não-direcionado é dito *simples* se ele não possui laços nem múltiplas arestas.

Um grafo *completo* de N vértices é aquele em que quaisquer dois vértices tomados em $\mathcal{V}$ são adjacentes. Um grafo é *conexo* quando, para quaisquer dois vértices $v_i, v_j \in \mathcal{V}$, existe um *caminho* (conjunto de arestas distintas com vértices em comum) ligando v_i a v_j. Equivalentemente, diz-se que um grafo é conexo se e somente se, para todo par $(v_i, v_j) \in \mathcal{V}^2$, existe uma sequência de inteiros menores ou iguais a N, como $(i, k, \ell, m, \ldots, n, j)$, tal que as entradas $A_{i,k}, A_{k,\ell}, A_{\ell,m}, \ldots, A_{n,j}$ de $\mathbf{A}$ são todas não nulas.

Grafos *ponderados* são aqueles em que a cada aresta associa-se um valor numérico (um *peso*); em grafos *não-ponderados*, as entradas não nulas da matriz de adjacência são todas unitárias. Um grafo é dito cíclico se ele possuir algum caminho fechado e acíclico caso contrário.

Um *subgrafo* de $\mathcal{G}$ é um grafo $\mathcal{G}'$ cujos vértices e arestas formam subconjuntos dos conjuntos de vértices e arestas de $\mathcal{G}$. Uma *componente conexa* de um grafo $\mathcal{G}$ é um subgrafo conexo de $\mathcal{G}$ (vide, por exemplo, o grafo da Fig. 1.1b, que possui 2 componentes conexas).

Pode-se definir a vizinhança de um vértice v_i como $\mathcal{N}_i$, o conjunto de vértices adjacentes a v_i. Denota-se por $\mathcal{N}(i, K)$ o conjunto de vértices conectados a v_i por um caminho com K ou menos arestas, como uma vizinhança dentro de um raio K (vide Fig. 1.2).

Para uma introdução mais completa à Teoria dos Grafos, os autores recomendam as referências [2, 9, 19, 40].

1.1 Definindo o sinal e seu domínio

No processamento digital de sinais, um sinal é representado matematicamente por uma sequência de números. Nesse contexto, o i-ésimo número numa sequência x é denotado por x_i, $i \in \mathbb{Z}$. Um sinal s sobre um grafo $\mathcal{G} = \{\mathcal{V}, \mathbf{A}\}$, com $|\mathcal{V}| = N$, é definido como uma função discreta que mapeia $\mathcal{V}$ em um conjunto de grandezas escalares, usualmente os números complexos ou reais,

$$s: \ \mathcal{V} \to \mathbb{C} \mid s(v_i) = s_i, \tag{1.1.1}$$

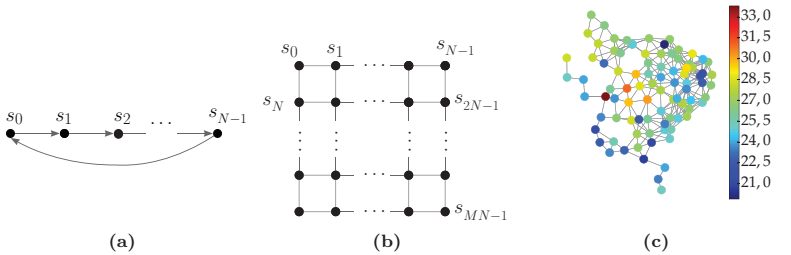


Figura 1.3: Representações de sinais sobre (a) um grafo em anel direcionado, (b) um grafo em grade retangular uniforme e (c) um grafo formado por cidades do Nordeste brasileiro.

de modo que o sinal como um todo pode ser representado por um vetor em $\mathbb{C}^N$ *indexado pelos vértices de* $\mathcal{G}$. Uma vez que se determina uma rotulação e ordem específica para os elementos de $\mathcal{V} = \{v_0, \ldots, v_{N-1}\}$, não há ambiguidade em representar o sinal como o vetor $\mathbf{s} = [s_0 \; s_1 \; \ldots \; s_{N-1}]^{\mathrm{T}} \in \mathbb{C}^N$. Representa-se por $\mathcal{S}$ o espaço de sinais definidos sobre um certo grafo $\mathcal{G} = \{\mathcal{V}, \mathbf{A}\}$.

A Fig. 1.3 traz exemplos de representações de sinais sobre grafos, nos quais a rotulação dos vértices é implícita (associa-se a amostra s_i ao vértice v_i). Os valores das amostras dos sinais podem ser indicados *numericamente*, junto aos vértices (Figs. 1.3a e 1.3b), ou por meio de uma escala de cores (Fig. 1.3c). Ao longo deste capítulo, será priorizado o uso desta última representação.

Um sinal de comprimento finito e de tempo discreto é modelado pelo grafo em anel direcionado (considera-se os pesos unitários), como mostrado na Fig. 1.3a: a noção de evolução temporal é capturada pelas arestas direcionadas; a periodicidade imposta pelas condições de fronteira da análise de Fourier de tempo discreto é modelada pela aresta realimentando a última amostra à primeira. O grafo na Fig. 1.3b é um modelo para imagens digitais [58] chamado *nearest-neighbor*, em que a dependência entre *pixels* é aproximada para existir apenas entre vizinhos, e o grafo na Fig. 1.3c é um exemplo de grafo de rede de sensores[1], com pesos das arestas dados pelo inverso da distância euclidiana, sobre o qual definiu-se o sinal da temperatura à meia-noite de 01 de fevereiro de 2012 em cidades do Nordeste brasileiro[2].

As características espectrais de um sinal dependem fortemente do domínio sobre o qual ele é definido. Em geral, diz-se que um sinal contém majoritariamente baixas frequências se amostras *adjacentes* têm valores *próximos*, e altas frequências se têm valores *díspares*. Considerando sinais definidos sobre grafos, fica evidente que a noção de amostras adjacentes depende da topologia do grafo em questão, e portanto *um mesmo sinal pode apresentar espectros distintos se definido sobre grafos diferentes*. A Fig. 1.4 confirma essa intuição, apresentando o espectro de um sinal segundo a base de autovetores da matriz Laplaciana do grafo, para dois grafos distintos; na Fig. 1.4c, as amostras de maior valor são adjacentes àquelas

[1] Quando não é especificado em contrário, chama-se *grafo de (rede de) sensores* um grafo conectado com vértices distribuídos uniformemente numa região do espaço euclidiano bidimensional.

[2] Fonte: Banco de Dados Meteorológicos para Ensino e Pesquisa (BDMEP) do Instituto Nacional de Meteorologia. Acesso gratuito, disponível em: `http://www.inmet.gov.br/portal/index.php?r=bdmep/bdmep`.

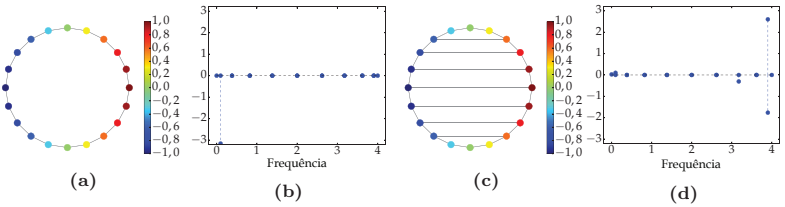


Figura 1.4: Um mesmo sinal definido sobre (a) um grafo não-direcionado em anel e (c) um grafo derivado do primeiro, mas com topologia distinta. (b) Espectro de Fourier, segundo $\mathrm{GSP_L}$ (segundo definido na Subseção 3.1), do sinal sobre o grafo em *(a)* e (d) em *(c)*.

de menor valor, o que corresponde a componentes de maior frequência do que se o grafo usado como domínio fosse um anel a pesos constantes (Fig. 1.4a).

1.2 Inferindo grafos

Em algumas situações em GSP, dispõe-se de um conjunto de dados, um ou mais sinais, mas não se tem o grafo subjacente. Informações sobre o contexto da aplicação e o fenômeno que gerou o sinal, no entanto, geralmente sugerem um ou outro método adequado para se estimar o grafo em questão.

Para grafos de sensores (não-direcionados), por exemplo, como na Fig. 1.3c, ponderar as arestas utilizando uma função gaussiana sobre a distância euclidiana $\mathrm{dist}(v_i, v_j)$ entre as posições correspondentes ao i-ésimo e ao j-ésimo vértices,

$$A_{i,j} = \exp\left(\frac{-\,\mathrm{dist}^2(v_i, v_j)}{2\theta^2}\right), \tag{1.2.2}$$

é muitas vezes adequado. Um limiar T é escolhido tal que, se $\mathrm{dist}\,(v_i, v_j) > T$, então faz-se $A_{i,j} = 0$, e os parâmetros T e θ são determinados de acordo com a aplicação.

No entanto, o uso do critério em (1.2.2) pode levar a um compromisso entre manter o grafo conexo e obter uma matriz de adjacência esparsa. Se a distância entre um vértice e seu vizinho mais próximo for muito maior do que a distância média entre vértices e seus vizinhos mais próximos, fazer com que o grafo seja conexo implica impor um limiar T grande, o que produziria muitas arestas. Para contornar esse problema e obter um grafo que capture razoavelmente as inter-relações entre os vértices, uma alternativa é ligar um vértice aos K mais próximos, ponderando adequadamente as arestas.

Essas técnicas supõem que há uma métrica de distância adequada para avaliar a similaridade entre as amostras de um par de vértices. Mas, dada a diversidade de possibilidades de sinais e aplicações, estimar a topologia do grafo subjacente a certo conjunto de dados já constitui um desafio em si [44, 45, 65].

1.3 Problemas

Problema 1.1. As matrizes Laplaciana e de adjacência de um grafo são exemplos de *operadores de deslocamento* (ou GSO, do inglês *graph shift operators*), um conceito fundamental em GSP. Dado um grafo $\mathcal{G}$ com conjunto de vértices $\mathcal{V} = \{v_0, \ldots, v_{N-1}\}$ e conjunto de arestas $\mathcal{E} \subseteq \mathcal{V} \times \mathcal{V}$, então, por definição, *um GSO é uma matriz* $\mathbf{S} \in \mathbb{R}^{N \times N}$ *tal que* $S_{i,j} \neq 0$ *se e somente se* $i = j$ *ou* $(v_i, v_j) \in \mathcal{E}$.

Suponha que um sinal $\mathbf{x}$ sobre o grafo $\mathcal{G}$ seja multiplicado à esquerda por um GSO $\mathbf{S}$,

$$\mathbf{y} = \mathbf{S}\mathbf{x}, \tag{1.3.3}$$

então de quais vértices de $\mathcal{G}$ virão os dados que comporão a amostra y_i, i.e. a amostra do sinal resultante, sobre o vértice v_i? (*Comentário*: a resposta deve ser concisa e independer do GSO em questão. Claro que, ao mudar a matriz $\mathbf{S}$, mudam os vértices que influenciam y_i, mas a intenção aqui é ressaltar a *ação local* do GSO).

Problema 1.2. Como será comentado no Capítulo 3, a matriz Laplaciana de um grafo não-direcionado simples é *positiva semi-definida*, o que significa que

$$\mathbf{x}^{\mathrm{T}}\mathbf{L}\mathbf{x} \geq 0 \tag{1.3.4}$$

para qualquer sinal $\mathbf{x} \in \mathbb{R}^N$ (não nulo) sobre o grafo.

(a) Partindo de (1.3.4), demonstre que todos os autovalores de $\mathbf{L}$ (para grafos não-direcionados simples) são maiores ou iguais a zero (i.e. não-negativos). Tal propriedade reflete diretamente na representação espectral de sinais sob $\mathrm{GSP}_{\mathrm{L}}$.

(b) Para toda matriz Laplaciana de um grafo não-direcionado, existe uma matriz $\mathbf{B}$ tal que $\mathbf{L} = \mathbf{B}^{\mathrm{T}}\mathbf{B}$. Utilizando esta relação, demonstre que (1.3.4) é verdadeira.

Problema 1.3. Considere um grafo direcionado em anel, com N vértices e arestas com pesos unitários.

(a) Obtenha os autovalores da matriz de adjacência $\mathbf{A}$ do grafo em questão e, então, mostre que a base usualmente empregada na definição da transformada discreta de Fourier (DFT, do inglês *discrete Fourier transform*) constitui uma possível base de autovetores da referida matriz (isto é, mostre que a matriz da DFT diagonaliza $\mathbf{A}$).

(b) A autobase mencionada no item anterior é única? Justifique sua resposta.

(c) Conside, agora, um grafo não-direcionado em caminho, também com N vértices e com arestas com pesos unitários. Obtenha os autovalores de sua matriz Laplaciana $\mathbf{L}$ e construa uma base formada por seus autovetores. Nesse caso, a base é única? A que transformada discreta (diferente da DFT) a base construída está relacionada? Justifique suas respostas.

Problema 1.4. Sejam $\mathcal{G}_1$ e $\mathcal{G}_2$ grafos com N_1 e N_2 vértices e cujas matrizes Laplacianas têm autovalores denotados por $\lambda_i(\mathcal{G}_1)$, $i \in \{0, 1, \ldots, N_1 - 1\}$, e $\lambda_j(\mathcal{G}_2)$, $j \in \{0, 1, \ldots, N_2 - 1\}$, respectivamente.

(a) Mostre que os autovalores da matriz Laplaciana do grafo produto $\mathcal{G} = \mathcal{G}_1 \times \mathcal{G}_2$ são todas as possíveis somas $\lambda_i(\mathcal{G}_1) + \lambda_j(\mathcal{G}_2)$, $i \in \{0, 1, \ldots, N_1 - 1\}$, $j \in \{0, 1, \ldots, N_2 - 1\}$.

(b) Além disso, mostre que, se $\mathbf{v}^{(i)}$ é um autovetor da matriz Laplaciana de $\mathcal{G}_1$, associado ao autovalor $\lambda_i(\mathcal{G}_1)$, e $\mathbf{v}^{(j)}$ é um autovetor da matriz Laplaciana de $\mathcal{G}_2$, associado ao autovalor $\lambda_j(\mathcal{G}_2)$, então $\mathbf{v}^{(i)} \otimes \mathbf{v}^{(j)}$ (em que $\otimes$ denota o produto de Kronecker) é um autovetor de $\mathcal{G}$ associado ao autovetor $\lambda_i(\mathcal{G}_1) + \lambda_j(\mathcal{G}_2)$.

Capítulo 2

$\mathrm{GSP_A}$: Grafos e o Processamento Algébrico de Sinais

Em 2006, Püschel e Moura publicaram sua teoria de processamento algébrico de sinais (ASP, do inglês *algebraic signal processing*) [53]. No trabalho em questão, os referidos autores apresentam descrições de sinais e de filtros que, partindo de um ponto de vista algébrico, são intencionalmente abstratas e genéricas; isso contrasta com a descrição usualmente adotada na literatura clássica do processamento de sinais, que caracteriza um filtro, por exemplo, como sendo um sistema capaz de modificar as frequências componentes de um sinal umas em relação às outras. No ASP, tanto os filtros como os sinais são simplesmente elementos de espaços vetoriais dotados de multiplicação e distributividade. Especificamente, o espaço de filtros é uma *álgebra*, sobre a qual toma-se um *módulo* para ser o espaço de sinais. Uma descrição detalhada sobre ASP foge ao escopo deste capítulo, mas o leitor interessado pode tirar bastante proveito dos trabalhos originais de Püschel e Moura [54, 55].

O aspecto de ASP que levou ao processamento sobre grafos, o que de fato nos interessa diretamente, é que o espaço de filtros é *gerado* pelo operador de atraso unitário de sinais. Um exemplo tornará a afirmação menos obscura: seja a álgebra polinomial $\mathbb{C}[x]/(x^N - 1)$, ou $\mathscr{A} = \{\sum_{\ell=0}^{N-1} a_\ell x^\ell | a_\ell \in \mathbb{C}\}$, que consiste em todos os polinômios de coeficientes complexos com grau menor do que N, dotados da adição e multiplicação polinomiais usuais, módulo $x^N - 1$. É comum representar sinais e filtros de tempo discreto e comprimento N como elementos de $\mathscr{A}$, caso em que a convolução cíclica[1] torna-se a simples multiplicação polinomial modular. Por exemplo, o sinal $\mathbf{s}_1 = (0, 2, 1, 0)$ é representado por $s_1(x) = x^2 + 2x$, e sua versão deslocada de uma unidade, $\mathbf{s}_2 = (0, 0, 2, 1)$, é dada por $s_2(x) = x^3 + 2x^2$. Fica claro que, neste caso, o filtro de atraso unitário é $d(x) = x$, cujas potências *geram* a base $(1, x, x^2, \ldots, x^{N-1})$ para o espaço de filtros. Esta relação se mantém para qualquer álgebra em ASP; portanto, Moura concluiu que, ao definir um operador de deslocamento unitário para sinais sobre grafos, isto daria início à construção da

[1] Da teoria clássica do processamento digital de sinais, o sinal discreto y resultante da convolução cíclica (ou circular) de comprimento N entre dois sinais discretos x e h de duração finita tem sua n-ésima componente calculada por $y_n = \sum_{m=0}^{N-1} x_m h_{(n-m) \pmod N}$.

teoria de GSP.

Partindo do grafo em anel direcionado com pesos unitários, modelo para o domínio de tempo discreto e periódico, observou-se que a matriz de adjacência do grafo,

$$\mathbf{C} = \begin{bmatrix} & & & 1 \\ 1 & & & \\ & \ddots & & \\ & & 1 & \end{bmatrix}, \tag{2.0.1}$$

realiza precisamente o papel de atraso num sinal de tempo discreto. Ou seja, se um sinal $\mathbf{s} = [s_0 \; s_1 \; \ldots \; s_{N-1}]^{\mathrm{T}}$ definido em um grafo em anel é multiplicado à esquerda por $\mathbf{C}$, tem-se

$$\mathbf{s}^{\langle 1 \rangle} = \mathbf{C}\mathbf{s}, \tag{2.0.2}$$

com $\mathbf{s}^{\langle 1 \rangle} = [s_{N-1} \; s_0 \; \ldots \; s_{N-2}]^{\mathrm{T}}$. Generalizando a ideia, Sandryhaila e Moura definiram o *operador de deslocamento unitário como sendo a matriz de adjacência do grafo sobre o qual o sinal está definido* [60]. É por isso que, neste capítulo, a vertente de GSP derivada da teoria de ASP é identificada por GSP$_{\mathrm{A}}$, indicando o papel fundamental exercido pela matriz de adjacência. Para um sinal $\mathbf{x}$ sobre um grafo qualquer $\mathcal{G} = \{\mathcal{V}, \mathbf{A}\}$, $\mathbf{A}$ age como um *filtro* de atraso (ou deslocamento) sobre $\mathbf{x}$, e a versão deslocada deste sinal é representada por $\mathbf{x}^{\langle 1 \rangle} = \mathbf{A}\mathbf{x}$.

2.1 Filtros sobre grafos

Observar a matriz de adjacência como um filtro levou à definição de um filtro de sinais sobre grafos como sendo qualquer matriz $\mathbf{H} \in \mathbb{C}^{N \times N}$ [61], visto que o produto matriz-vetor sempre resulta num vetor (ou *filtro* $\times$ *sinal* $=$ *sinal*). Isso implica que os filtros sobre grafos são sempre lineares, uma vez que a distributividade da multiplicação em relação à adição matricial garante que

$$\mathbf{H}(\alpha_1 \mathbf{x}_1 + \alpha_2 \mathbf{x}_2) = \alpha_1 \mathbf{H}\mathbf{x}_1 + \alpha_2 \mathbf{H}\mathbf{x}_2. \tag{2.1.3}$$

A propriedade de invariância no tempo (ou ao deslocamento) requer que $\mathbf{A}\mathbf{H}\mathbf{x} = \mathbf{H}\mathbf{A}\mathbf{x}$, $\forall \mathbf{x}$. Nesse contexto, filtros lineares e invariantes ao deslocamento, ou LSI (do inglês *linear and shift-invariant*), têm a forma de um polinômio $h(\cdot)$ avaliado em $\mathbf{A}$,

$$h(\mathbf{A}) = \sum_{\ell=0}^{L-1} h_\ell \mathbf{A}^\ell, \tag{2.1.4}$$

com L menor ou igual ao grau do polinômio mínimo $m_{\mathbf{A}}$ de $\mathbf{A}$ [60,63]. Assim, filtros LSI são uma série de potências finita no operador de deslocamento, exatamente como ocorre em processamento clássico de sinais de tempo discreto, em que os filtros lineares e invariantes no tempo (LTI, do inglês *linear and time-invariant*) têm representação em termos de polinômios em z^{-1}, pela transformada Z [17]. Além disso, os filtros LSI formam uma álgebra polinomial $\mathbb{C}[x]/m_{\mathbf{A}}(x)$.

2.2 A transformada de Fourier sobre grafos

Uma vez que a transformada de Fourier de um sinal é a sua projeção em uma base de funções invariantes à filtragem LTI [48], em GSP$_{\mathrm{A}}$ define-se a transformada de

Fourier sobre grafos (GFT, do inglês *graph Fourier transform*) como a decomposição de um sinal em termos de uma base de autovetores da filtragem LSI [62].

Seja $\mathbf{A}$ a matriz de adjacência de um grafo de N vértices. Se $\mathbf{A}$ for diagonalizável, tem-se[2]

$$\mathbf{A} = \mathbf{V}\boldsymbol{\Lambda}\mathbf{V}^{-1}, \tag{2.2.5}$$

em que $\mathbf{V}$ contém os N autovetores de $\mathbf{A}$ em suas colunas, isto é,

$$\mathbf{V} = [\mathbf{v}_0 \ \mathbf{v}_1 \ \ldots \ \mathbf{v}_{N-1}]. \tag{2.2.6}$$

Como filtros LSI são polinômios em $\mathbf{A}$, as colunas de $\mathbf{V}$ formam uma base de vetores invariantes à filtragem LSI para o espaço de sinais $\mathcal{S}$ sobre o grafo com matriz de adjacência $\mathbf{A}$. Desta forma, um sinal $\mathbf{x} \in \mathcal{S}$ pode ser decomposto em suas componentes na base $\mathbf{V}$ como

$$\begin{aligned} \mathbf{x} &= \widehat{x}_0\mathbf{v}_0 + \cdots + \widehat{x}_{N-1}\mathbf{v}_{N-1} = \mathbf{V}[\widehat{x}_0 \ \widehat{x}_1 \ \ldots \ \widehat{x}_{N-1}]^{\mathrm{T}} \\ &= \mathbf{V}\widehat{\mathbf{x}}, \end{aligned} \tag{2.2.7}$$

e esta é definida como a equação de síntese da *transformada de Fourier sobre grafos*. A equação de análise da GFT é, portanto,

$$\widehat{\mathbf{x}} = \mathbf{V}^{-1}\mathbf{x}. \tag{2.2.8}$$

Para sinais de tempo discreto, foi assinalado que seu domínio é modelado como um grafo com matriz de adjacência $\mathbf{C}$ em (2.0.1). Como $\mathbf{C}$ é circulante, ela é diagonalizada pela matriz da transformada discreta de Fourier (DFT, do inglês *discrete Fourier transform*), dada por $\mathbf{F}$, $F_{n,k} = \exp\left(-\jmath\frac{2\pi}{N}nk\right)$, em que $\jmath = \sqrt{-1}$[3]. Assim, pode-se escrever

$$\mathbf{C} = \mathbf{F}^{-1}\boldsymbol{\Lambda}_{\mathbf{C}}\mathbf{F}, \tag{2.2.9}$$

em que $\boldsymbol{\Lambda}_{\mathbf{C}}$ é uma matriz diagonal com os autovalores de $\mathbf{C}$. Vê-se que a matriz da GFT, para grafos em anel, é $\mathbf{V}^{-1} = \mathbf{F}$, o que resulta na desejável propriedade de que a GFT de sinais de tempo discreto coincide com a DFT, demonstrando consistência com a teoria clássica.

2.3 O domínio da frequência

A definição empregada para a GFT sugere interpretar os autovetores $\mathbf{v}_i$ da matriz de adjacência como as "componentes de frequência" associadas às *frequências de grafo* representadas pelos autovalores λ_i (como a componente de Fourier $\mathrm{e}^{-\jmath\Omega t}$, no domínio do tempo contínuo t, é associada à frequência Ω). A menos que os polinômios característico e mínimo de $\mathbf{A}$ sejam iguais, uma mesma frequência estará associada a duas ou mais componentes linearmente independentes, como ocorreu com o sinal na Fig. 2.1. Na mesma figura, nota-se que, embora o sinal seja visualmente suave, seu espectro possui componentes associadas a autovalores de grande magnitude, o que levanta a questão de definir um critério coerente para se falar em *altas* e *baixas* frequências de sinais sobre grafos.

[2] Se $\mathbf{A}$ não for diagonalizável, o raciocínio pode ser repetido utilizando-se a forma canônica de Jordan.

[3] Aqui, adotamos a notação usualmente empregada em textos de Engenharia Elétrica, em que a unidade imaginária é associada à letra $\jmath$; evita-se utilizar a letra $\imath$, pois esta é comumente escolhida para representar correntes elétricas.

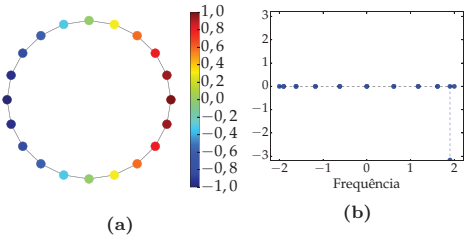


Figura 2.1: (a) Sinal sobre um grafo em anel não-direcionado e (b) seu espectro em GSP$_A$.

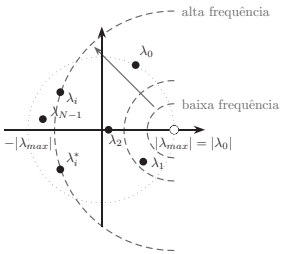


Figura 2.2: Ordenamento das frequências de sinais sobre grafos, de baixa para alta, no plano complexo. Fonte: adaptado de [64].

Para fundamentar matematicamente a noção de frequência no contexto de GSP, partiu-se de uma métrica usual para sinais de tempo discreto: a *variação total*, que calcula a soma das diferenças entre amostras adjacentes de um sinal e, portanto, assume valores maiores para sinais de maior frequência. Sua expressão matemática, para certo sinal de comprimento finito $\mathbf{x} = (x_0, x_1, \ldots, x_{N-1})$, é

$$\mathrm{TV}(\mathbf{x}) = \sum_{n=0}^{N-1} |x_n - x_{n-1 \bmod N}|. \tag{2.3.10}$$

De (2.0.1) e (2.0.2), vê-se que (2.3.10) pode ser reescrita em termos da norma ℓ_1[4] como $\mathrm{TV}(\mathbf{x}) = \|\mathbf{x} - \mathbf{C}\mathbf{x}\|_1$, utilizando a matriz de adjacência do grafo em anel para realizar o deslocamento cíclico do sinal. Assim, uma generalização da função $\mathrm{TV}(\cdot)$ para sinais definidos sobre grafos quaisquer, i. e., a *variação total sobre grafos* para um sinal $\mathbf{s}$ sobre o grafo $\mathcal{G} = \{\mathcal{V}, \mathbf{A}\}$, foi definida como

$$\mathrm{TV}_{\mathrm{G}}(\mathbf{s}) \triangleq \|\mathbf{s} - \mathbf{A}^{norm}\mathbf{s}\|_1, \tag{2.3.11}$$

com $\mathbf{A}^{norm} = |\lambda_{\max}|^{-1}\mathbf{A}$ e $\lambda_{\max}$ o autovalor de $\mathbf{A}$ com maior módulo. A normalização de $\mathbf{A}$ visa evitar a magnificação excessiva das amostras do sinal deslocado [64].

[4] A norma ℓ_1 é um caso particular da norma ℓ_p de um vetor $\mathbf{x} \in \mathbb{C}^N$, definida como $\|\mathbf{x}\|_p \triangleq \left(\sum_{k=0}^{N-1} |x_k|^p\right)^{1/p}$. Quando não for explicitamente indicado, a notação sem subscrito $\|\cdot\|$ indica a norma ℓ_2.

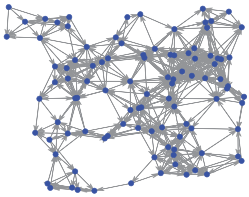


Figura 2.3: Grafo de sensores direcionado, com 100 vértices, sem laços ou múltiplas arestas.

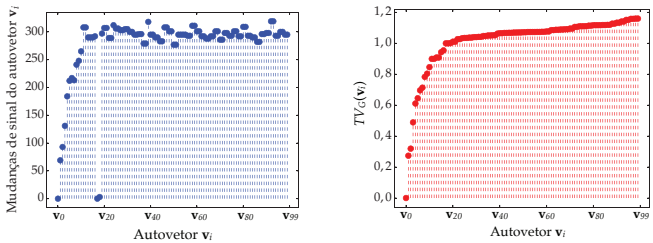


Figura 2.4: (a) Número de mudanças de sinal e (b) variação total dos autovetores $(\mathbf{v}_i)_{i\in\{0,\ldots,N-1\}}$ da matriz $\mathbf{A}$, ordenados de modo que os respectivos autovalores se disponham do mais próximo ao mais distante do ponto real $|\lambda_{\max}|$ no plano complexo.

Seja $\mathbf{A}$ diagonalizável e com autovalores (possivelmente complexos) ordenados tais que

$$|\lambda_0| \leq |\lambda_1| \leq \cdots \leq |\lambda_{N-1}| \triangleq |\lambda_{\max}|, \tag{2.3.12}$$

associados aos autovetores $(\mathbf{v}_i)_{i=0,\ldots,N-1}$ escolhidos de modo que $\|\mathbf{v}_k\|_1 = 1$. Tomando a variação total de um autovetor $\mathbf{v}_k$ associado a λ_k, tem-se

$$\begin{aligned}\mathrm{TV}_{\mathrm{G}}(\mathbf{v}_k) &= \|\mathbf{v}_k - \mathbf{A}\mathbf{v}_k\|_1 = \|\mathbf{v}_k - \frac{1}{|\lambda_{\max}|}\lambda_k\mathbf{v}_k\|_1 = \left|1 - \frac{\lambda_k}{|\lambda_{\max}|}\right| \|\mathbf{v}_k\|_1 \\ &= \Big|\lambda_k - |\lambda_{\max}|\Big| \frac{\|\mathbf{v}_k\|_1}{|\lambda_{\max}|}\end{aligned}$$

de forma que, como foi feito $\|\mathbf{v}_k\|_1 = 1$, tem-se a equivalência

$$\Big|\lambda_i - |\lambda_{\max}|\Big| \leq \Big|\lambda_j - |\lambda_{\max}|\Big| \iff \mathrm{TV}_{\mathrm{G}}(\mathbf{v}_i) \leq \mathrm{TV}_{\mathrm{G}}(\mathbf{v}_j), \tag{2.3.13}$$

ou seja, componentes de frequência associadas a autovalores mais próximos do ponto $|\lambda_{\max}|$ no plano complexo são mais suaves e são, portanto, ditas de *baixa frequência*. A Fig. 2.2 ilustra esse ordenamento para as frequências de grafos, o que esclarece a leitura do espectro do sinal na Fig. 2.1a, cuja matriz de adjacência tem autovalores reais.

Para verificar a consistência desta noção de frequência, tomemos o grafo da Fig. 2.3. O número de mudanças de sinal (número de arestas ligando vértices com

amostras de sinal distinto) e a variação total $\mathrm{TV_G}$ de cada autovetor da matriz de adjacência do grafo são mostrados na Fig. 2.4. Ambas as métricas visam a quantificar a taxa de variação de um sinal sobre o grafo, mas, uma vez que o número de mudanças de sinal ignora as variações que não cruzam o zero, a variação total sobre grafos representa de forma mais fiel a medida de frequência em um sinal. A Fig. 2.4b mostra como $\mathrm{TV_G}(\mathbf{v}_k)$ cresce monotonicamente à medida que o respectivo autovalor de $\mathbf{v}_k$, λ_k, se afasta do ponto $|\lambda_{\max}|$ no plano complexo.

Para concluir a análise do domínio da frequência, convém apresentar a resposta em frequência de um filtro LSI sobre grafos. A definição dada na Subseção 2.1 considera a ação de uma matriz sobre um sinal $\mathbf{x}$ no domínio dos vértices de um grafo $\mathcal{G} = \{\mathcal{V}, \mathbf{A}\}$. Para entender como o filtro age no domínio da GFT, doravante chamado domínio da frequência, será utilizada (2.2.5) e a representação polinomial de filtros LSI. Seja o filtro $\mathbf{H} = \sum_{\ell=0}^{L} h_\ell \mathbf{A}^\ell$ e sua resposta ao sinal $\mathbf{x}$ dada por

$$\mathbf{H}\mathbf{x} = \sum_{\ell=0}^{L} h_\ell \mathbf{A}^\ell \mathbf{x} = \sum_{\ell=0}^{L} h_\ell \left(\mathbf{V}\mathbf{\Lambda}\mathbf{V}^{-1}\right)^\ell \mathbf{x} = \mathbf{V}\left(\sum_{\ell=0}^{L} h_\ell \mathbf{\Lambda}^\ell\right)\mathbf{V}^{-1}\mathbf{x}. \tag{2.3.14}$$

Tomando a GFT em ambos os lados da última equação, tem-se

$$\mathbf{V}^{-1}\mathbf{H}\mathbf{x} = h(\mathbf{\Lambda})\widehat{\mathbf{x}}, \tag{2.3.15}$$

de forma que a ação no domínio da frequência a que corresponde a filtragem por $\mathbf{H}$ é a multiplicação pela matriz $h(\mathbf{\Lambda})$, i. e. $h(\mathbf{\Lambda})$ representa a resposta em frequência do filtro $\mathbf{H}$.

2.4 Problemas

Problema 2.1. No contexto de sinais sobre grafos, dadas duas matrizes de adjacência (ou operadores de deslocamento) $\mathbf{A}$ e $\widetilde{\mathbf{A}}$, os filtros $h(\mathbf{A})$ e $g(\widetilde{\mathbf{A}})$ são ditos equivalentes, se, para todos os sinais de entrada $\mathbf{s} \in \mathcal{S}$, eles produzem saídas iguais, isto é, $h(\mathbf{A})\mathbf{s} = g(\widetilde{\mathbf{A}})\mathbf{s}$. Considerando o que se acabou de definir, responda os seguintes itens:

(a) Mostre que, para qualquer matriz $\mathbf{A}$, existe uma matriz $\widetilde{\mathbf{A}}$ e um polinômio $r(x)$, tais que $\mathbf{A} = r(\widetilde{\mathbf{A}})$ e $p_{\widetilde{\mathbf{A}}}(x) = m_{\widetilde{\mathbf{A}}}(x)$, em que $p_{\widetilde{\mathbf{A}}}(x)$ e $m_{\widetilde{\mathbf{A}}}(x)$ denotam, respectivamente, o polinômio característico e o polinômio mínimo de $\widetilde{\mathbf{A}}$.

(b) Mostre que um filtro definido como em (2.1.4) possui um filtro equivalente único no mesmo grafo, com no máximo $\deg m_{\mathbf{A}} = N_{\mathbf{A}}$ derivações.

Problema 2.2. A equação (2.2.9) estabelece o paralelo entre a análise espectral de sinais de tempo discreto e sinais sobre grafos dirigidos em anel. Demonstre que o ordenamento de frequências proposto em $\mathrm{GSP_A}$, ilustrado pela Fig. 2.2, coincide exatamente com o ordenamento em DSP clássico. Ou seja, considerando que o i-ésimo autovalor da matriz de adjacência do grafo em anel é $\lambda_i = \exp\left(\jmath\frac{2\pi}{N}i\right)$, mostre que

(a) O autovetor mais suave (de menor frequência) é aquele associado ao autovalor mais próximo de $|\lambda_{\max}| = 1$, no plano complexo.

(b) O(s) autovetor(es) menos suave(s) (de maior frequência) é(são) aquele(s) associado(s) ao(s) autovalor(es) mais distantes de $|\lambda_{\max}|$, no plano complexo.

(c) Autovetores associados a autovalores complexos conjugados têm exatamente a mesma frequência.

Problema 2.3. Empregando um *software* à sua escolha, produza um grafo de sensores direcionado como aquele ilustrado na Fig. 2.3. Você pode consultar o Capítulo 7, que contém orientações preliminares sobre o uso de ferramentas computacionais que facilitam a implementação de operações e a produção de gráficos e imagens relacionadas ao processamento de sinais sobre grafos.

(a) Produza gráficos como aqueles apresentados na Fig. 2.4, de modo que se possa verificar a consistência da noção de frequência para sinais sobre grafos, quando se considera a variação total TV_{G}.

(b) Defina um sinal senoidal sobre o grafo em questão; considere o referido sinal com diferentes frequências e, para cada uma dessas frequências, plote-o sobre o grafo, obtendo uma imagem como a da Fig. 2.3, mas colorindo os vértices de modo que se possa ter uma noção visual de variação do sinal sobre o grafo.

(c) Calcule e plote a GFT do sinal em questão, para cada uma das frequências consideradas no item (b). Que reflexos dessas mudanças de frequência podem ser enxergados no domínio da transformada?

Capítulo 3

$\mathrm{GSP_L}$: A Teoria Espectral de Grafos em GSP

A teoria espectral dos grafos é um ramo de teoria dos grafos que estuda propriedades oriundas da autodecomposição das matrizes de adjacência e Laplaciana de grafos. Com forte base em álgebra linear, ela encontra aplicações em mecânica quântica, redes de comunicações, química etc. [9], e foi utilizada por Ortega, Shuman e outros para desenvolver um ferramental teórico de processamento de sinais sobre grafos distinto de $\mathrm{GSP_A}$ [68]; esse ferramental considera, em particular, o espectro da matriz Laplaciana, e por isso é referido aqui por $\mathrm{GSP_L}$. Neste ramo de GSP, os autores costumam restringir seu estudo apenas a grafos não-direcionados e com arestas de peso real não-negativo, como observaram Sandryhaila e Moura [63]. O arcabouço teórico apresentado a seguir é baseado na matriz Laplaciana

$$\mathbf{L} = \mathbf{D} - \mathbf{A}; \tag{3.0.1}$$

oportunamente, pode-se considerar também a sua versão normalizada, a qual é largamente utilizada na área de teoria espectral de grafos e é expressa por

$$\boldsymbol{\mathcal{L}} = \mathbf{D}^{-1/2}\mathbf{L}\mathbf{D}^{-1/2}. \tag{3.0.2}$$

Uma importante propriedade da matriz $\mathbf{L}$ para grafos não-direcionados e ponderados pode ser derivada a partir de sua decomposição em termos da *matriz de incidência orientada*, aqui denotada por $\mathbf{B}$ (são considerados grafos sem laços ou múltiplas arestas).

Definição 3.1 (Orientação de um grafo)**.** *Seja o grafo não-direcionado $\mathcal{G} = \{\mathcal{V}, \mathbf{A}\}$. Uma* orientação *de $\mathcal{G}$ é um grafo $\mathcal{G}' = \{\mathcal{V}, \mathbf{A}'\}$ onde a cada aresta de $\mathcal{G}$ foi imposta uma direção qualquer.*

Definição 3.2 (Matriz de incidência orientada)**.** *Seja o grafo não-direcionado $\mathcal{G} = \{\mathcal{V}, \mathbf{A}\}$, com $|\mathcal{V}| = N$ vértices e $|\mathcal{E}| = E$ arestas ponderadas segundo a função $\omega : \mathcal{E} \to \mathbb{R}^+$. Seja, ainda, $\mathcal{G}' = \{\mathcal{V}, \mathbf{A}'\}$ uma orientação de $\mathcal{G}$. Uma* matriz de incidência orientada *$\mathbf{B} \in \mathbb{R}^{N\times E}$ de $\mathcal{G}$ tem entradas dadas por*

$$B_{i,j} \triangleq \begin{cases} \sqrt{\omega(e'_j)}, & \textit{se } e'_j \text{ chega em } v_i \\ -\sqrt{\omega(e'_j)}, & \textit{se } e'_j \text{ parte de } v_i \\ 0, & \textit{c.c.} \end{cases} \tag{3.0.3}$$

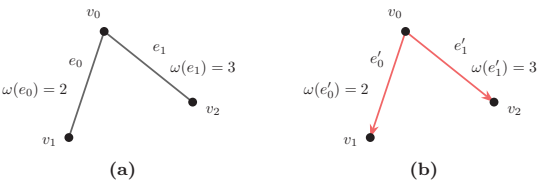


Figura 3.1: (a) Um grafo não-direcionado, ponderado e conexo, e (b) uma possível orientação sua. A função $\omega : \mathcal{V}^2 \to \mathbb{R}^+$ mapeia a aresta e no valor real não-negativo $\omega(e)$.

É importante notar que, no contexto de $\mathrm{GSP_L}$, no qual os grafos têm pesos reais não-negativos, a matriz de incidência orientada $\mathbf{B}$ tem entradas sempre reais. O Teorema 3.1 [22, Proposição 2.3] traz a decomposição da matriz Laplaciana em termos da matriz de incidência orientada.

Teorema 3.1. *Para um dado grafo não-direcionado, com matriz de incidência orientada* $\mathbf{B}$*, a matriz Laplaciana é dada por*

$$\mathbf{L} = \mathbf{B}\mathbf{B}^{\mathrm{T}}. \tag{3.0.4}$$

Exemplo 3.1. *Para ilustrar a decomposição da matriz Laplaciana para grafos simples ponderados, em termos da matriz de incidência de uma orientação do grafo, usemos o grafo da Fig. 3.1a, do qual uma possível orientação é mostrada na Fig. 3.1b. A matriz de incidência para esta orientação é*

$$\mathbf{B} = \begin{array}{c} \\ v_0 \\ v_1 \\ v_2 \end{array} \begin{array}{c} \begin{array}{cc} e_0 & e_1 \end{array} \\ \begin{bmatrix} -\sqrt{2} & -\sqrt{3} \\ \sqrt{2} & 0 \\ 0 & \sqrt{3} \end{bmatrix} \end{array}, \tag{3.0.5}$$

de modo que

$$\begin{aligned} \mathbf{B}\mathbf{B}^{\mathrm{T}} &= \begin{bmatrix} -\sqrt{2} & -\sqrt{3} \\ \sqrt{2} & 0 \\ 0 & \sqrt{3} \end{bmatrix} \begin{bmatrix} -\sqrt{2} & \sqrt{2} & 0 \\ -\sqrt{3} & 0 & \sqrt{3} \end{bmatrix} = \begin{bmatrix} 5 & -2 & -3 \\ -2 & 2 & 0 \\ -3 & 0 & 3 \end{bmatrix} = \begin{bmatrix} 5 & 0 & 0 \\ 0 & 2 & 0 \\ 0 & 0 & 3 \end{bmatrix} - \begin{bmatrix} 0 & 2 & 3 \\ 2 & 0 & 0 \\ 3 & 0 & 0 \end{bmatrix} \\ &= \mathbf{D} - \mathbf{A} = \mathbf{L}. \end{aligned}$$

O Teorema 3.1 permite demonstrar que a matriz Laplaciana de um grafo não-direcionado é positiva semi-definida ($\mathbf{x}^{\mathrm{T}}\mathbf{L}\mathbf{x} \geq 0, \forall \mathbf{x}$), pois para qualquer sinal real $\mathbf{x} = [x_0 \; x_1 \; \ldots \; x_{N-1}]^{\mathrm{T}}$,

$$\mathbf{x}^{\mathrm{T}}\mathbf{L}\mathbf{x} = \mathbf{x}\mathrm{T}\mathbf{B}\mathbf{B}\mathrm{T}\mathbf{x} = (\mathbf{B}\mathrm{T}\mathbf{x})\mathrm{T}(\mathbf{B}\mathrm{T}\mathbf{x}) = \|\mathbf{B}\mathrm{T}\mathbf{x}\|^2 \geq 0. \tag{3.0.6}$$

Assim, todos os autovalores de $\mathbf{L}$ são reais não-negativos, pois se $\mathbf{u}$ é um autovetor unitário com autovalor γ, então

$$0 \leq \mathbf{u}\mathrm{T}\mathbf{L}\mathbf{u} = \mathbf{u}\mathrm{T}\gamma\mathbf{u} = \gamma\mathbf{u}\mathrm{T}\mathbf{u} = \gamma. \tag{3.0.7}$$

Além disso, pode-se mostrar que zero é um autovalor com multiplicidade igual ao número de componentes conexas do grafo não-direcionado [68] e, portanto, grafos conexos têm apenas um autovalor de $\mathbf{L}$ nulo. Como conclusão, os autovalores de $\mathbf{L}$, denotados por γ, podem ser ordenados como $\gamma_0 = 0 < \gamma_1 \leq \cdots \leq \gamma_{N-1}$.

3.1 O operador diferença e a GFT

De (3.0.1), percebe-se que a matriz Laplaciana age como um operador diferença sobre um sinal $\mathbf{x}$ definido em um grafo $\mathcal{G} = \{\mathcal{V}, \mathbf{A}\}$, atualizando cada amostra pela diferença entre o valor em certo vértice e a soma dos valores em vértices adjacentes. Ou seja,

$$\mathbf{L}\mathbf{x} = \mathbf{D}\mathbf{x} - \mathbf{A}\mathbf{x}, \tag{3.1.8}$$

$$\begin{aligned} \Rightarrow (\mathbf{L}\mathbf{x})_i &= d_i x_i - \sum_{k \mid v_k \in \mathcal{N}_i} A_{ik} x_k = \sum_k A_{ik} x_i - \sum_{k \mid v_k \in \mathcal{N}_i} A_{ik} x_k \\ &= \sum_{k \mid v_k \in \mathcal{N}_i} A_{ik}[x_i - x_k]. \end{aligned} \tag{3.1.9}$$

Na concepção de sua teoria e formulação de sua transformada de Fourier sobre grafos (GFT), Shuman *et al.* basearam-se no fato de que a transformada de Fourier de tempo contínuo é uma expansão de um sinal numa base de exponenciais complexas, que são autofunções do operador Laplaciano unidimensional (segunda derivada) [68, 69]:

$$\Delta \mathrm{e}^{\jmath\Omega t} = \frac{\partial^2}{\partial t^2} \mathrm{e}^{\jmath\Omega t} = -\Omega^2 \mathrm{e}^{\jmath\Omega t}. \tag{3.1.10}$$

Com isto, a GFT de um sinal $\mathbf{x}$, sobre o grafo não-direcionado $\mathcal{G} = \{\mathcal{V}, \mathbf{A}\}$ com matriz Laplaciana $\mathbf{L} = \mathbf{U}\boldsymbol{\Gamma}\mathbf{U}^{-1}$, foi definida como a sua expansão sobre a base de autovetores de $\mathbf{L}$, ou seja,

$$\widetilde{\mathbf{x}} \triangleq \mathbf{U}^{-1}\mathbf{x} \qquad \text{(análise)} \; ; \qquad \mathbf{x} = \mathbf{U}\widetilde{\mathbf{x}}, \qquad \text{(síntese)} \tag{3.1.11}$$

e como a matriz $\mathbf{L}$ é real e simétrica, é sempre diagonalizável. Mais do que isso, existe uma matriz de autovetores *real* e *ortonormal* $\mathbf{U}$, i. e. $\mathbf{U}^{-1} = \mathbf{U}\mathrm{T}$ [31, Teorema 4.4].

Tal definição traz uma ideia de frequência para sinais sobre grafos coerente com a interpretação clássica. Em (3.1.10), a informação de frequência está contida nos autovalores $-\Omega^2$ associados a cada autofunção $\mathrm{e}^{\jmath\Omega t}$, e autovalores próximos de zero estão associados a harmônicos mais suaves. O mesmo observa-se nos autovalores e autovetores da Laplaciana de um grafo: autovetores associados a menores autovalores são mais suaves que aqueles associados a maiores autovalores. Como a Laplaciana age como um operador diferença, a suavidade de $\mathbf{x}$ pode ser medida pela norma de $\mathbf{L}\mathbf{x}$, uma métrica similar à variação total em $\mathrm{GSP_A}$ (cf. (2.3.10)), e, se $\mathbf{u}_i$ é o autovetor de $\mathbf{L}$ associado a γ_i,

$$\|\mathbf{L}\mathbf{u}_i\| = \|\gamma_i \mathbf{u}_i\| = \gamma_i \|\mathbf{u}_i\| \quad (\text{pois } \gamma_i \geq 0), \tag{3.1.12}$$

de forma que, normalizando os autovetores de $\mathbf{L}$, percebe-se que se $\gamma_i \leq \gamma_j$ então $\|\mathbf{L}\mathbf{u}_i\| \leq \|\mathbf{L}\mathbf{u}_j\|$ e $\mathbf{u}_i$ é mais suave que $\mathbf{u}_j$.

Além disso, observando o autovetor associado ao único autovalor nulo do Laplaciano para grafos não-direcionados conexos, tem-se que

$$\mathbf{L}\mathbf{u}_0 = \gamma_0 \mathbf{u}_0 = \mathbf{0}, \tag{3.1.13}$$

que se trata de um sistema de equações cujas soluções formam o espaço nulo de $\mathbf{L}$. Como a dimensão deste espaço é igual à multiplicidade geométrica de $\gamma_0 = 0$, que é

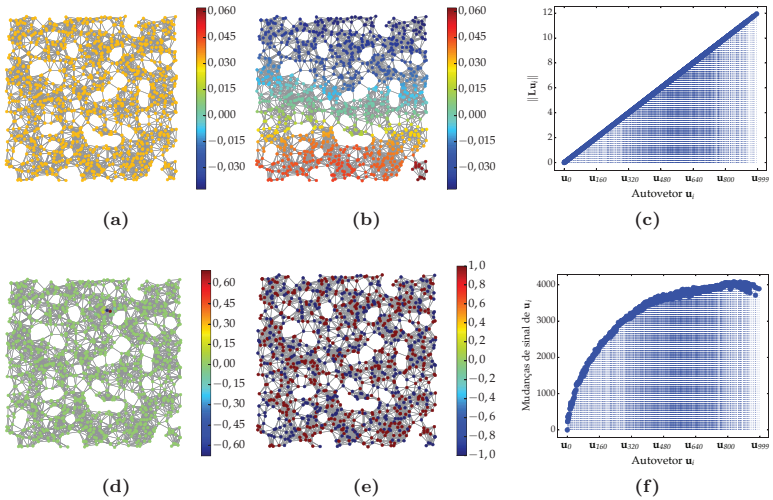


Figura 3.2: Alguns autovetores da matriz Laplaciana de um grafo de sensores com 1000 vértices: (a) $\mathbf{u}_0$, (b) $\mathbf{u}_1$ e (d) $\mathbf{u}_{999}$. Como $\mathbf{u}_{999}$ tem um pico difícil de discernir das demais amostras do sinal, em (e) é mostrada uma versão binarizada $\bar{\mathbf{u}}_{999}$, valendo 1 em amostras positivas de $\mathbf{u}_{999}$ e -1 nas negativas. Em (c) e (f) são mostradas medidas de suavidade de um sinal: a norma da diferença $\|\mathbf{L}\mathbf{u}_i\|$ e o número de mudanças de sinal dos autovetores $\mathbf{u}_i$.

igual a 1, basta encontrar uma solução não-trivial de (3.1.13) para obter uma base dos possíveis autovetores $\mathbf{u}_0$. De (3.1.9), vê-se que

$$(\mathbf{L}\mathbf{u}_0)_i = 0 \Rightarrow \sum_{k \mid v_k \in \mathcal{N}_i} A_{ik}[u_{0i} - u_{0k}] = 0, \tag{3.1.14}$$

o que mostra que qualquer vetor *constante* é solução para (3.1.13). Isto equivale a dizer que o autovetor associado ao autovalor (frequência) nulo é constante, como ocorre na teoria clássica de processamento de sinais. Se os autovetores forem normalizados, então cada entrada de $\mathbf{u}_0$ vale $1/\sqrt{N}$.

A Fig. 3.2 ilustra o conceito de frequência em $\mathrm{GSP_L}$. Foi utilizado um grafo de sensores com 1000 vértices, com arestas ponderadas segundo (1.2.2). A componente de frequência nula é o autovetor $\mathbf{u}_0$, e a suavidade das componentes diminui à medida que se aumenta a frequência (autovalor). A Fig. 3.2f mostra o número de mudanças de sinal para cada autovetor em função do respectivo autovalor, calculado como o número de arestas ligando vértices cujo produto das amostras é negativo.

3.2 Filtragem

O conceito de filtragem de sinais sobre grafos, no presente ferramental teórico, é definido primeiro no domínio da frequência: a resposta em frequência de um filtro

$\mathbf{H}$ é o vetor $\widetilde{\mathbf{h}} \in \mathbb{R}^N$ tal que, para certo sinal $\mathbf{x}$ com GFT $\widetilde{\mathbf{x}} = \mathbf{U}^{-1}\mathbf{x}$ sobre um certo grafo, sua saída nesse filtro é

$$\widetilde{\mathbf{y}} \triangleq \widetilde{\mathbf{h}} \odot \widetilde{\mathbf{x}} = \operatorname{diag}(\widetilde{\mathbf{h}})\widetilde{\mathbf{x}}, \tag{3.2.15}$$

em que $\odot$ representa o produto de Hadamard e $\operatorname{diag}(\cdot)$ é uma matriz diagonal com as entradas da diagonal dadas pelo argumento. Tomando a GFT inversa em ambos os membros da equação,

$$\mathbf{U}\widetilde{\mathbf{y}} = \mathbf{y} = \mathbf{U}\left[\operatorname{diag}(\widetilde{\mathbf{h}})\widetilde{\mathbf{x}}\right] = \mathbf{U}\operatorname{diag}(\widetilde{\mathbf{h}})\mathbf{U}^{-1}\mathbf{x}, \tag{3.2.16}$$

ou seja, a filtragem no domínio dos vértices do grafo é dada por

$$\mathbf{y} = \mathbf{H}\mathbf{x}, \tag{3.2.17}$$

com

$$\mathbf{H} \triangleq \mathbf{U}\operatorname{diag}(\widetilde{\mathbf{h}})\mathbf{U}^{-1}. \tag{3.2.18}$$

Pode-se fazer as componentes de $\widetilde{\mathbf{h}}$ serem polinômios de grau menor ou igual a K nos autovalores de $\mathbf{L}$ [68], de modo que

$$\widetilde{h}_\ell = \sum_{k=0}^{K} a_k \gamma_\ell^k, \tag{3.2.19}$$

e que a resposta do filtro à entrada $\mathbf{x}$, obtida pela aplicação em (3.2.15) da GFT inversa, tenha entradas

$$y_i = \sum_{\ell=0}^{N-1} u_{\ell,i}\widetilde{h}_\ell \widetilde{x}_\ell, \tag{3.2.20}$$

em que $u_{\ell,i}$ é a i-ésima componente do autovetor $\mathbf{u}_\ell$. Substituindo (3.2.19) em (3.2.20),

$$y_i = \sum_{\ell=0}^{N-1} u_{\ell,i}\left(\sum_{k=0}^{K} a_k\gamma_\ell^k\right)\left(\sum_{j=0}^{N-1} u_{\ell,j}x_j\right) = \sum_j x_j\left[\sum_k a_k\left(\sum_\ell \gamma_\ell^k u_{\ell,i}u_{\ell,j}\right)\right].$$

Mas como $L_{i,j} = \left(\mathbf{U}\mathbf{\Gamma}\mathbf{U}^T\right)_{i,j} = \sum_{\ell=0}^{N-1}\gamma_\ell u_{\ell,i}u_{\ell,j}$, então

$$\sum_{\ell=0}^{N-1}\gamma_\ell^k u_{\ell,i}u_{\ell,j} = \left(\mathbf{L}^k\right)_{i,j} \tag{3.2.21}$$

e, portanto,

$$y_i = \sum_{j=0}^{N-1} x_j \sum_{k=0}^{K} a_k\left(\mathbf{L}^k\right)_{i,j}. \tag{3.2.22}$$

Por fim, usa-se o fato de que, se o menor número de arestas ligando os vértices v_i a v_j é maior do que k, então $\left(\mathbf{L}^k\right)_{i,j} = 0$ [25, Lema 5.2], como citado em [68]). Com isso,

$$y_i = \sum_{j|v_j\in\mathcal{N}(i,K)} H_{i,j}x_j, \tag{3.2.23}$$

em que $H_{i,j} = \sum_{k=0}^{K} a_k\left(\mathbf{L}^k\right)_{i,j}$. É indicado por (3.2.23) que a filtragem no domínio dos vértices é uma operação linear localizada, combinando linearmente amostras na vizinhança de um vértice dentro de um raio de até K arestas.

Figura 3.3: Um grafo do tipo *path*.

3.3 Problemas

Problema 3.1. Considere um grafo completo não-direcionado de 4 vértices. Relembre os conceitos de *orientação* e *grafo completo* na Definição 3.1 e na introdução do Capítulo 1.

(a) Quantos diferentes grafos direcionados podem ser construídos como uma *orientação* deste grafo?

(b) Generalize o cálculo da letra anterior para o caso de um grafo completo de N vértices e encontre a expressão. Utilize uma calculadora ou software para obter a quantidade quando $N = 10$.

Problema 3.2. Partindo da definição da Laplaciana $\mathbf{L} = \mathbf{D} - \mathbf{A}$, *calcule* o resultado da multiplicação de $\mathbf{L}$ pelo vetor $(1, 1, \ldots, 1)^{\mathrm{T}}$. Perceba como o resultado corrobora com a noção de $\mathbf{L}$ como um operador diferença.

Dica: em (3.1.13), foi demonstrado o mesmo resultado, mas utilizando um raciocínio inverso — partindo do autovalor nulo, encontrou-se seu autovetor.

Problema 3.3. Seja o grafo "caminho" (*path*) de 4 vértices, mostrado na Fig. 3.3.

(a) Escreva a matriz Laplaciana deste grafo (considere os pesos unitários). Descreva agora o caso geral: consegue enxergar como é a matriz para o caso de um grafo caminho de N vértices?

(b) Mostre que a matriz Laplaciana de um grafo caminho de N vértices, $\mathbf{L}_{\mathrm{path}}$, com pesos unitários, é diagonalizada pela matriz da transformada discreta do cosseno, tipo-2 (DCT-2):

$$\mathbf{C}_{n,k} = \cos k\frac{\pi}{N}\left(n + \frac{1}{2}\right). \tag{3.3.24}$$

Ou seja, demonstre que a matriz $\mathbf{C}^{\mathrm{T}}\mathbf{L}_{\mathrm{path}}\mathbf{C}$ é diagonal.

Dica: o que se deseja provar é o mesmo que mostrar que as colunas de $\mathbf{C}$ formam um conjunto linearmente independente e são autovetores de $\mathbf{L}_{\mathrm{path}}$.

Discussão:

A DCT-2 é uma transformada amplamente utilizada em processamento de sinais, muito conhecida por sua aplicação em compressão de imagens. O resultado obtido neste exercício tem uma consequência relevante: **quando o grafo é do tipo *path*, a GFT coincide com a DCT-2**.

Para aprofundar-se na relação entre os tipos de DCT e as matrizes que elas diagonalizam, confira o material do prof. Gilbert Strang [73].

Experimento em software:

O leitor pode se basear no código em Python 3.1 para verificar numericamente o que foi demonstrado na letra (b).

```python
import numpy as np
from scipy.linalg import circulant

def dct2(N):
    """Matriz (nao-ortogonal) da DCT-2."""
    C = np.zeros((N, N))
    for n in range(N):
        for k in range(N):
            C[n, k] = np.cos(k * np.pi/N * (n + 0.5))
    return C

def path_laplacian(N):
    """Laplaciana do grafo caminho nao-dirigido de N vertices (N >
    2)."""
    assert N >= 2, "Numero de vertices menor que 2."
    D = np.diag([1] + (N-2)*[2] + [1])  # graus
    A = np.tri(N, k=1) - np.tri(N, k=-2) - np.eye(N)  # adjacencia
    return D - A

N = 4  # quantos vertices?
C = dct2(N)  # matriz da DCT2
L = path_laplacian(N)
out = np.round(C.T @ L @ C, decimals=4)

print("DCT2:\n", np.round(C, decimals=4))
print("Laplaciana:\n", L)
print("Produto C.T * L * C:\n", out)
print("Autovalores de L:", np.diag(out))
```

Código 3.1: Implementação da diagonalização da Laplaciana do grafo caminho pela matriz de transformação da DCT2.

Problema 3.4. O grafo em anel com 4 vértices, não-direcionado e com pesos unitários, tem como possível matriz *inversa* da GFT (segundo $\mathrm{GSP_L}$)

$$\mathbf{F}^{-1} = \begin{bmatrix} 0.5 & 0.7071 & -0.5 & 0 \\ 0.5 & 0 & 0.5 & -0.7071 \\ 0.5 & -0.7071 & -0.5 & 0 \\ 0.5 & 0 & 0.5 & 0.7071 \end{bmatrix} \tag{3.3.25}$$

(aproximando em 4 casas decimais). Antes de prosseguir, tente identificar quais passos foram precisos para calcular essa matriz.

Considere o seguinte sinal no domínio da frequência: $\widetilde{\mathbf{x}} = (0, 0, 2, 0)\mathrm{T}$.

(a) O que seria *mais apropriado* dizer: este sinal é de alta ou de baixa frequência?

(b) Calcule a sua GFT inversa, obtendo o sinal no domínio dos vértices. O resultado está de acordo com a resposta da letra (a)?

(c) Observe o número de amostras não-nulas no domínio dos vértices e compare com o domínio da frequência, onde há apenas uma amostra não-nula. Veja que, ainda que mudemos a posição da componente não-nula na frequência (deixando apenas a componente da frequência mais baixa, ou apenas a da mais alta, ou qualquer outra), o curioso fenômeno se repete. Por que você acha que essa diferença ocorre?

Extra: pesquise, se ainda não conhecer, sobre o Princípio da Incerteza de Processamento de Sinais (*uncertainty principle in signal processing*, para uma busca na literatura em inglês).

Problema 3.5. Empregando um *software* à sua escolha, produza um grafo de sensores como aquele ilustrado na Fig. 2.3, porém não-direcionado. Você pode consultar o Capítulo 7, que contém orientações preliminares sobre o uso de ferramentas computacionais que facilitam a implementação de operações e a produção de gráficos e imagens relacionadas ao processamento de sinais sobre grafos.

(a) Produza gráficos como aqueles apresentados na Fig. 3.2 (c) e na Fig. 3.2 (f), de modo que se possa avaliar os resultados providos por duas medidas de suavidade de um sinal sobre um grafo não-direcionado: a norma da diferença $\|\mathbf{L}\mathbf{u}_i\|$ e o número de mudanças de sinal dos autovetores $\mathbf{u}_i$. O que você observa?

(b) Defina um sinal senoidal sobre o grafo em questão; considere o referido sinal com diferentes frequências e, para cada uma dessas frequências, plote-o sobre o grafo, obtendo uma imagem como a da Figura 2.3, mas colorindo os vértices de modo que se possa ter uma noção visual de variação do sinal sobre o grafo.

(c) Calcule e plote a GFT do sinal em questão, para cada uma das frequências consideradas no item (b). Que reflexos dessas mudanças de frequência podem ser enxergados no domínio da transformada?

Capítulo 4

Amostragem e Reconstrução de Sinais sobre Grafos

Analogamente ao que ocorre em DSP, a amostragem[1] e a reconstrução de sinais definidos sobre grafos desempenham papéis importantes em GSP. Nesse contexto, o conceito de limitação de faixa precisa ser estendido convenientemente para o caso de sinais em grafos de forma a viabilizar suas reconstruções perfeitas, as quais se dão geralmente via um simples procedimento de interpolação por mínimos quadrados, desde que alguns requisitos técnicos sejam atendidos [6]. A amostragem de sinais em grafos permite que se trabalhe com os sinais em dimensão reduzida, diminuindo a complexidade das ferramentas de GSP. Além disso, assumindo que a limitação em faixa do sinal sobre o grafo é conhecida, o processo de amostragem e interpolação permite a redução de ruído do sinal original limitado em faixa. Quando os sinais amostrados estão sob influência de ruídos, abordagens adaptativas [37–39] são alternativas promissoras para estimar sinais sobre grafos. Esta estimação adaptativa baseia-se nos conhecidos algoritmos *least-mean-squares* (LMS) e *recursive least-squares* (RLS) [16] e viabiliza a reconstrução e o rastreamento em tempo real de sinais cujos modelos probabilísticos são não-estacionários. As principais ideias relacionadas a esses conceitos são descritas a seguir.

4.1 Limitação de faixa e amostragem

No contexto de GSP_L, a extensão do conceito de limitação de faixa dá-se assim:

Definição 4.1 (Limitação de faixa)**.** *Um sinal* $\mathbf{x}$ *é* limitado em faixa*, ou* espectralmente esparso (eesparso)*, quando sua representação no domínio da frequência* $\widetilde{\mathbf{x}}$*, dada pela GFT em* (3.1.11)*, é esparsa. Mais especificamente, tomando* $\mathcal{F}$ *como um subconjunto de índices de* $\{0, 1, \ldots, N-1\}$*, o sinal* $\mathbf{x}$ *é dito* $\mathcal{F}$-eesparso *se os elementos de* $\widetilde{\mathbf{x}}$ *com índices em* $\overline{\mathcal{F}} \triangleq \{0, 1, \ldots, N-1\} \setminus \mathcal{F}$ *são nulos, isto é,* $\widetilde{\mathbf{x}}_{\overline{\mathcal{F}}}$ *é um vetor nulo de dimensão* $N - |\mathcal{F}|$*.*

Cabe aqui ressaltar que a definição acima também se aplica ao contexto de

[1] Talvez uma definição mais precisa seja de *subamostragem*, em vez de *amostragem*, tendo em vista que o sinal em grafo é originalmente definido no domínio discreto dos vértices. Porém será mantido o uso mais comum do termo *amostragem*.

GSP_A, apenas mudando $\widetilde{\mathbf{x}}$ por $\widehat{\mathbf{x}}$ definido em (2.2.8). Optou-se por trabalhar com GSP_L apenas para fixar a notação.

Note que, se $\mathbf{x}$ é $\mathcal{F}$-eesparso, então a operação de síntese em (3.1.11) pode ser realizada por

$$\mathbf{x} = \mathbf{U}_{\mathcal{F}}\widetilde{\mathbf{x}}_{\mathcal{F}}\,, \tag{4.1.1}$$

onde $\mathbf{U}_{\mathcal{F}}$ é a matriz $N \times |\mathcal{F}|$ formada por autovetores da Laplaciana do grafo indexados por $\mathcal{F}$.

Em DSP, sinais temporais com faixa limitada podem ser amostrados e reconstruídos sem perda de informação, desde que o critério de Nyquist seja satisfeito. Descreve-se agora um resultado semelhante para sinais sobre grafos. Inicialmente, considere que as operações de amostragem e reconstrução de um sinal $\mathbf{x}$ podem ser vistas como resultado da pré-multiplicação do sinal original por uma matriz de amostragem $\mathbf{D}$ e por uma matriz de interpolação $\mathbf{\Phi}$. Para a reconstrução perfeita do sinal original, deseja-se encontrar matrizes $\mathbf{D}$ e $\mathbf{\Phi}$ tais que $\mathbf{x} = \mathbf{\Phi}\mathbf{D}\mathbf{x}$ para qualquer $\mathbf{x}$ descrito por (4.1.1).

A etapa de amostragem consiste na observação dos valores de um sinal sobre grafo num *conjunto de amostragem* $\mathcal{D} \subseteq \mathcal{V}$. Nesse contexto, pode-se especificar a matriz de amostragem como $\mathbf{D}_{\mathcal{D}}$, uma matriz diagonal com entradas $d_i = 1$, se $v_i \in \mathcal{D}$, e $d_i = 0$, caso contrário. Assim, obtém-se o vetor

$$\mathbf{x}_{\mathcal{D}} \triangleq \mathbf{D}_{\mathcal{D}}\mathbf{x}\,. \tag{4.1.2}$$

Para reconstruir um sinal $\mathcal{F}$-eesparso $\mathbf{x}$ a partir de uma versão amostrada $\mathbf{x}_{\mathcal{D}}$, utiliza-se a expressão (4.1.1) e, quando a matriz $\mathbf{U}_{\mathcal{F}}^{\mathrm{T}}\mathbf{D}_{\mathcal{D}}\mathbf{U}_{\mathcal{F}}$ tem posto completo, $\mathbf{\Phi}$ é escolhida como

$$\mathbf{\Phi} = \mathbf{U}_{\mathcal{F}}\big(\mathbf{U}_{\mathcal{F}}^{\mathrm{T}}\mathbf{D}_{\mathcal{D}}\mathbf{U}_{\mathcal{F}}\big)^{-1}\mathbf{U}_{\mathcal{F}}^{\mathrm{T}}\,, \tag{4.1.3}$$

de tal forma que o procedimento de amostragem e interpolação descrito por $\mathbf{\Phi}\mathbf{D}_{\mathcal{D}}\mathbf{x}$ resulta no sinal original $\mathbf{x} = \mathbf{U}_{\mathcal{F}}\widetilde{\mathbf{x}}_{\mathcal{F}}$. A condição de posto completo é satisfeita desde que o conjunto de amostragem escolhido $\mathcal{D}$ garanta a igualdade [5, 6]

$$\mathrm{rank}(\mathbf{D}_{\mathcal{D}}\mathbf{U}_{\mathcal{F}}) = |\mathcal{F}|\,. \tag{4.1.4}$$

Como $\mathrm{rank}(\mathbf{D}_{\mathcal{D}}) = |\mathcal{D}|$, a partir de (4.1.4) conclui-se que uma condição necessária para a recuperação perfeita de um sinal sobre grafo amostrado é $|\mathcal{D}| \geq |\mathcal{F}|$, isto é, o número de amostras retidas deve ser, pelo menos, a quantidade de componentes não-nulas de $\widetilde{\mathbf{x}}$.

O Algoritmo 1 mostra como empregar a estratégia de interpolação para lidar com sinais sobre grafos com faixa limitada e variantes no tempo. Este algoritmo basicamente calcula uma estimativa instantânea de mínimos quadrados $\mathbf{x}_{\mathrm{e}}[k]$ para o sinal original de faixa limitada $\mathbf{x}[k]$ a partir do sinal amostrado atual $\mathbf{D}_{\mathcal{D}}\mathbf{x}_w[k]$ no instante do tempo $k \in \mathbb{Z}$, onde $\mathbf{x}_w[k]$ é o vetor de sinal medido, possivelmente corrompido por ruído. Cabe ressaltar aqui que o termo algoritmo está sendo utilizado num sentindo amplo para englobar a situação comum no contexto em questão em que se assume uma entrada de duração infinita, exigindo, portanto, o cômputo de saídas para todo instante $k \in \mathbb{Z}$.

Algoritmo 1 Interpolação linear instantânea

1: $k \leftarrow 0$
2: $\mathbf{\Phi} = \mathbf{U}_{\mathcal{F}}\left(\mathbf{U}_{\mathcal{F}}^{\mathrm{T}}\mathbf{D}_{\mathcal{D}}\mathbf{U}_{\mathcal{F}}\right)^{-1}\mathbf{U}_{\mathcal{F}}^{\mathrm{T}}$
3: **while** (true) **do**
4: $\quad \mathbf{x}_{\mathrm{e}}[k] = \mathbf{\Phi}\mathbf{D}_{\mathcal{D}}\mathbf{x}_w[k]$
5: $\quad k \leftarrow k+1$
6: **end**

4.2 Estimação adaptativa de sinais sobre grafos

Uma tentativa inicial de mesclar a tradicional área de filtragem adaptativa [16] com o novo campo de GSP é feita em [38], onde os autores sugerem estratégias baseadas em LMS para lidar com o problema de reconstrução de um sinal definido sobre grafo. Além disso, estratégias baseadas em RLS são propostas em [39] para tarefas de reconstrução e aprendizado de sinais sobre grafo.

Uma estratégia adaptativa num contexto de estimação de sinais sobre grafos possibilita uma reconstrução robusta e o rastreamento em tempo real de sinais aleatórios com estatísticas variantes no tempo. A robustez da abordagem adaptativa é útil para trabalhar em cenários ruidosos, como (4.3.11), quando espera-se que uma estratégia adaptativa forneça um *mean-square deviation* (MSD) menor em comparação a uma interpolação instantânea do sinal, como no Algoritmo 1.

Estimação LMS

O algoritmo de filtragem adaptativa mais utilizado é o LMS [78, 79]. Entre muitas razões, como a convergência não polarizada para a solução de Wiener e o comportamento estável quando implementado com precisão finita, a popularidade do algoritmo LMS se deve principalmente à sua simplicidade e baixa complexidade computacional [16].

Num contexto de reconstrução de sinais sobre grafos [36, 38], considera-se o modelo para o sinal observado no instante k

$$\boldsymbol{x}_w[k] = \mathbf{x}[k] + \boldsymbol{w}[k], \tag{4.2.5}$$

onde o sinal inicial de faixa limitada $\mathbf{x}[k]$ é corrompido por um ruído aditivo de média zero, $\boldsymbol{w}[k]$, gerando uma sequência de observações $\boldsymbol{x}_w[k]$.[2] A tarefa de reconstrução consiste na recuperação de $\mathbf{x}[k] = \mathbf{x}_0$ a partir de $\boldsymbol{x}_w[k]$. Note que, no caso em que o sinal inicial varia com o tempo, a recuperação adaptativa de $\mathbf{x}[k]$ a partir de $\boldsymbol{x}_w[k]$ consiste numa tarefa de rastreamento.

Explorando a esparsidade do sinal de faixa limitada, o algoritmo LMS é projetado de forma a minimizar a média do erro quadrático sobre as amostras dos vértices em $\mathcal{D}$,

$$\min_{\widetilde{\mathbf{x}}_{\mathcal{F}}[k]} \mathbb{E}\left\{ ||\mathbf{D}_{\mathcal{D}}(\boldsymbol{x}_w[k] - \mathbf{U}_{\mathcal{F}}\widetilde{\mathbf{x}}_{\mathcal{F}}[k])\|_2^2 \right\} . \tag{4.2.6}$$

Usando a abordagem de minimização baseada em gradiente estocástico para o problema (4.2.6), encontra-se uma expressão de atualização para $\widetilde{\mathbf{x}}_{\mathcal{F}}[k+1]$, que por

[2] Aqui, $\boldsymbol{x}$ denota um vetor aleatório com realizações denotadas como $\mathbf{x}$.

sua vez gera uma estimativa $\mathbf{x}_{\mathrm{e}}[k]$ para o sinal de faixa limitada $\mathbf{x}[k]$ em (4.1.1), que corresponde à equação de atualização do LMS [36]

$$\mathbf{x}_{\mathrm{e}}[k+1] = \mathbf{x}_{\mathrm{e}}[k] + \mu \mathbf{U}_{\mathcal{F}} \mathbf{U}_{\mathcal{F}}^{\mathrm{T}} \mathbf{D}_{\mathcal{D}} (\mathbf{x}_w[k] - \mathbf{x}_{\mathrm{e}}[k]) \,, \tag{4.2.7}$$

em que $\mu \in \mathbb{R}_+$ é o chamado fator de convergência [16], parâmetro que controla o comportamento do algoritmo para melhorar sua velocidade de convergência ou reduzir o erro em estado estacionário. No caso em que o sinal de entrada é variante no tempo, a capacidade de rastreamento do algoritmo está associada à velocidade com que $\mathbf{x}[k]$ varia com k.

Dessa forma, o procedimento para encontrar uma estimativa em tempo real de $\mathbf{x}[k]$ com base na equação de atualização do LMS (4.2.7) é apresentado no Algoritmo 2. O valor inicial $\mathbf{x}_{\mathrm{e}}[0]$ deve ser um sinal limitado em faixa (como $\mathbf{x}_{\mathrm{e}}[0] = \mathbf{0}$) e o tamanho de passo μ deve ser suficientemente pequeno para que o algoritmo seja estável [38].

Algoritmo 2 Estimação LMS de sinais sobre grafos

1: $k \leftarrow 0$
2: **while** (true) **do**
3: $\quad \mathbf{x}_{\mathrm{e}}[k+1] = \mathbf{x}_{\mathrm{e}}[k] + \mu\, \mathbf{U}_{\mathcal{F}} \mathbf{U}_{\mathcal{F}}^{\mathrm{T}} \mathbf{D}_{\mathcal{D}} (\mathbf{x}_w[k] - \mathbf{x}_{\mathrm{e}}[k])$
4: $\quad k \leftarrow k+1$
5: **end**

Estimação RLS

Outra técnica bem conhecida na área de filtragem adaptativa consiste num procedimento iterativo que resolve o problema de mínimos quadrados para cada sinal de entrada de forma recursiva, sendo este chamado de algoritmo RLS [16]. O método RLS oferece uma abordagem mais rápida que o LMS, atingindo seu estado estacionário em menos tempo. No entanto, o compromisso encontrado ao utilizar um algoritmo de menor tempo de convergência, como o RLS, é o aumento considerável na complexidade computacional, o que pode ser um fator limitante, dependendo da aplicação.

De forma similar à ideia do LMS, o algoritmo RLS para estimação de sinais sobre grafos [37] pretende minimizar uma função do erro quadrático sobre as amostras nos vértices em $\mathcal{D}$, da forma

$$\min_{\widetilde{\mathbf{x}}_{\mathcal{F}}} \sum_{l=1}^{k} \beta^{k-l} \|\mathbf{D}_{\mathcal{D}} (\mathbf{x}_w[l] - \mathbf{U}_{\mathcal{F}} \widetilde{\mathbf{x}}_{\mathcal{F}})\|^2_{\mathbf{C}_w^{-1}} + \beta^k \|\widetilde{\mathbf{x}}_{\mathcal{F}}\|^2_{\mathbf{\Pi}}, \tag{4.2.8}$$

em que $0 \ll \beta \leq 1$ é o fator de esquecimento, e $\mathbf{\Pi}$ é uma matriz de regularização, geralmente escolhida como $\mathbf{\Pi} = \delta \mathbf{I}$, para um pequeno $\delta > 0$.

Ao resolver o problema convexo em (4.2.8), verifica-se que a estimativa para o sinal limitado em faixa $\mathbf{x}[k]$ é dada por

$$\mathbf{x}_{\mathrm{e}}[k] = \mathbf{U}_{\mathcal{F}} \mathbf{\Psi}^{-1}[k] \boldsymbol{\psi}[k], \tag{4.2.9}$$

onde $\mathbf{\Psi}[k]$ e $\boldsymbol{\psi}[k]$ são calculados por

$$\begin{aligned} \mathbf{\Psi}[k] &= \beta \mathbf{\Psi}[k-1] + \mathbf{U}_{\mathcal{F}}^{\mathrm{T}} \mathbf{D}_{\mathcal{D}} \mathbf{C}_w^{-1}[k]\, \mathbf{U}_{\mathcal{F}} \,, \\ \boldsymbol{\psi}[k] &= \beta \boldsymbol{\psi}[k-1] + \mathbf{U}_{\mathcal{F}}^{\mathrm{T}} \mathbf{D}_{\mathcal{D}} \mathbf{C}_w^{-1}[k]\, \mathbf{x}_w[k] \,. \end{aligned} \tag{4.2.10}$$

Assim, com base nas equações de atualização (4.2.9) e (4.2.10), a técnica RLS para estimação de sinais definidos sobre grafos [37] é apresentada no Algoritmo 3. Os valores iniciais deste algoritmo podem ser tomados como $\mathbf{\Psi}[0] = \mathbf{\Pi}$ e um vetor $\boldsymbol{\psi}[0]$ qualquer, como $\boldsymbol{\psi}[0] = \mathbf{0}$.

Algoritmo 3 Estimação RLS de sinais sobre grafos

1: $k \leftarrow 0$
2: **while** (true) **do**
3: $\quad \mathbf{\Psi}[k] = \beta\mathbf{\Psi}[k-1] + \mathbf{U}_{\mathcal{F}}^{\mathrm{T}}\mathbf{D}_{\mathcal{D}}\mathbf{C}_w^{-1}[k]\,\mathbf{U}_{\mathcal{F}}$
4: $\quad \boldsymbol{\psi}[k] = \beta\boldsymbol{\psi}[k-1] + \mathbf{U}_{\mathcal{F}}^{\mathrm{T}}\mathbf{D}_{\mathcal{D}}\mathbf{C}_w^{-1}[k]\,\mathbf{x}_w[k]$
5: $\quad \mathbf{x}_{\mathrm{e}}[k] = \mathbf{U}_{\mathcal{F}}\mathbf{\Psi}^{-1}[k]\boldsymbol{\psi}[k]$
6: $\quad k \leftarrow k+1$
7: **end**

4.3 Seleção do conjunto de amostragem

Até este ponto, qualquer escolha de *conjunto de amostragem* $\mathcal{D}$ respeitando a condição (4.1.4) resulta numa recuperação perfeita do sinal original ao aplicar o Algoritmo 1. No entanto, em uma situação prática é de se esperar que as medições obtidas sejam corrompidas por ruído.

Com base nesta suposição, considera-se uma modelagem em que o ruído é introduzido durante o proceso de amostragem, de forma que

$$\mathbf{D}_{\mathcal{D}}\boldsymbol{x}[k] = \mathbf{D}_{\mathcal{D}}\mathbf{x}[k] + \boldsymbol{w}[k] \tag{4.3.11}$$

Considerando o modelo em (4.3.11), pode-se calcular algumas figuras de mérito para um sinal estimado com a estratégia de interpolação no Algoritmo 1, $\mathbf{x}_{\mathrm{e}}[k]$, como o MSD dado por

$$\begin{aligned}\mathrm{MSD} &= \mathbb{E}\left\{\|\boldsymbol{x}_{\mathrm{e}}[k] - \mathbf{x}[k]\|_2^2\right\} \\ &= \mathbb{E}\left\{\|\mathbf{U}_{\mathcal{F}}\big(\mathbf{U}_{\mathcal{F}}^{\mathrm{T}}\mathbf{D}_{\mathcal{D}}\mathbf{U}_{\mathcal{F}}\big)^{-1}\mathbf{U}_{\mathcal{F}}^{\mathrm{T}}\boldsymbol{w}[k]\|_2^2\right\}\,,\end{aligned} \tag{4.3.12}$$

ou o *square deviation* (SD) dado por

$$\begin{aligned}\mathrm{SD} &= \|\mathbf{x}_{\mathrm{e}}[k] - \mathbf{x}[k]\|_2^2 \\ &= \|\mathbf{U}_{\mathcal{F}}(\mathbf{U}_{\mathcal{F}}^{\mathrm{T}}\mathbf{D}_{\mathcal{D}}\mathbf{U}_{\mathcal{F}})^{-1}\mathbf{U}_{\mathcal{F}}^{\mathrm{T}}\mathbf{w}[k]\|_2^2\,.\end{aligned} \tag{4.3.13}$$

As expressões em (4.3.12) e (4.3.13) apresentam uma dependência explícita de $\mathbf{D}_{\mathcal{D}}$, indicando que o conjunto de amostragem $\mathcal{D}$ pode ser apropriadamente escolhido para otimizar a figura de mérito desejada. No entanto, o problema de otimização resultante é de natureza combinatória.

A fim de obter um bom compromisso entre o erro de reconstrução obtido e o tempo necessário para o cálculo do conjunto de amostragem $\mathcal{D}$, uma abordagem comum é empregar um algoritmo guloso para a otimização. Um algoritmo guloso basicamente reduz a complexidade computacional geral, procurando por uma seleção ótima em cada estágio e esperando encontrar um valor final próximo do ótimo global. Informações detalhadas a respeito da capacidade de reconstrução de estratégias gulosas para GSP são apresentadas em [3].

Um exemplo de seleção do conjunto de amostragem é apresentado no Algoritmo 4, que emprega em cada iteração uma busca gulosa pelo índice $n \in \{0, 1, \ldots, N-1\}$ a ser adicionado ao conjunto atual $\mathcal{D}$ para maximizar o autovalor mínimo não negativo de $\mathbf{U}_{\mathcal{F}}^{\mathrm{T}}\mathbf{D}_{\mathcal{D}\cup\{n\}}\mathbf{U}_{\mathcal{F}}$. O número de índices no conjunto final $\mathcal{D}$ é denotado como D. Este método foi inicialmente sugerido em [6] e tem a mesma forma que uma das estratégias de amostragem descritas em [38]. De fato, seguindo uma ideia similar a [6], pode ser mostrado que o Algoritmo 4 usa um esquema guloso para minimizar o SD em (4.3.13).

Algoritmo 4 Seleção gulosa de $\mathcal{D}$.

1: $\mathcal{D} \leftarrow \emptyset$
2: **while** $|\mathcal{D}| < D$ **do**
3: $\quad d = \underset{n\in\{0,1,\ldots,N-1\}}{\operatorname{argmax}} \lambda_{\min}^{+}(\mathbf{U}_{\mathcal{F}}^{\mathrm{T}}\mathbf{D}_{\mathcal{D}\cup\{n\}}\mathbf{U}_{\mathcal{F}})$
4: $\quad \mathcal{D} \leftarrow \mathcal{D} + \{d\}$
5: **end**
6: **return** $\mathcal{D}$

4.4 Recuperação prescindindo do suporte do sinal na frequência

Até então, foi assumido o conhecimento do suporte (faixa) do sinal no domínio da frequência para sua reconstrução/estimação. De fato, todos os algoritmos até então apresentados utilizam a matriz $\mathbf{U}_{\mathcal{F}}$, que corresponde a uma seleção adequada dos autovetores da matriz Laplaciana de acordo com o pré-conhecimento do suporte do sinal no domínio da frequência. Uma abordagem alternativa seria, dado o conhecimento do conjunto de amostragem $\mathcal{D}$, estimar diretamente o valor do sinal nos demais vértices do grafo, de modo que o sinal estimado seja suave. No contexto de GSP_A, Sandryhaila e Moura em [64] quiseram estimar um sinal de valores discretos, de modo a aplicar o método a problemas de classificação, e para tanto aplicaram um algoritmo de otimização para obter o sinal que varia o mínimo sobre o grafo, minimizando a norma $\|\mathbf{s} - \mathbf{A}^{norm}\mathbf{s}\|_2$, sujeito, também, à condição de que o sinal estimado divergisse pouco do sinal original nos vértices com valores conhecidos *a priori*.

Uma abordagem semelhante para GSP_L seria utilizar a norma da matriz Laplaciana aplicada ao sinal como métrica da sua suavidade, como apresentado na Seção 3.1. Isto se traduz no problema de otimização

$$\mathbf{x}_{\mathrm{e}} = \underset{\mathbf{s}\in\mathbb{R}^N}{\arg\min} \|\mathbf{L}\mathbf{s}\|_2, \quad \text{sujeito a} \quad \mathbf{D}_{\mathcal{D}}\mathbf{s} = \mathbf{D}_{\mathcal{D}}\mathbf{x}, \tag{4.4.14}$$

em que $\mathbf{x}$ é o sinal original sobre o grafo, não necessariamente limitado em faixa. Algoritmos adaptativos poderiam ser deduzidos a partir dessa abordagem também.

4.5 Problemas

Problema 4.1. Considere um grafo $\mathcal{G}$, com autovalores e autovetores da matriz Laplaciana dados, respectivamente, por γ_i e $\mathbf{u}_i$, com $0 \leq i \leq N-1$. Defina o

conjunto $\mathcal{F}$ de forma que o sinal $\mathcal{F}$-eesparso em $\mathcal{G}$ seja equivalente à saída de um filtro:

(a) Passa-baixas com frequência de corte γ_l

(b) Passa-altas com frequência de corte γ_h

(c) Passa-faixa com frequências de corte γ_{b1} e γ_{b2}

Problema 4.2. Para o problema anterior, mostre que os valores dos índices em $\mathcal{F}$ não determinam o tipo do sinal (por exemplo, passa-baixas) em um grafo arbitrário.

Problema 4.3. Considere um grafo para o qual a matriz Laplaciana tem autovetores dada por:

$$\mathbf{U} = \begin{bmatrix} 0{,}4472 & 0{,}1659 & 0{,}3350 & -0{,}2066 & -0{,}7858 \\ 0{,}4472 & 0{,}3976 & -0{,}8009 & 0{,}0205 & -0{,}0084 \\ 0{,}4472 & 0{,}1731 & 0{,}3148 & -0{,}5805 & 0{,}5779 \\ 0{,}4472 & -0{,}8748 & -0{,}1850 & -0{,}0204 & -0{,}0037 \\ 0{,}4472 & 0{,}1382 & 0{,}3362 & 0{,}7871 & 0{,}2201 \end{bmatrix}.$$

Construa o sinal limitado em faixa $\mathbf{x}[k] = \mathbf{x}_0 = \mathbf{u}_0 + \mathbf{u}_2$ e o sinal observado $\mathbf{x}_w[k] = \mathbf{x}[k] + \boldsymbol{w}[k]$, para $1 \leq k \leq 500$, onde $\boldsymbol{w}[k]$ são observações de um ruído branco Gaussiano de média zero e matriz de covariância $0{,}1 \cdot \mathbf{I}_{5\times 5}$. Plote o erro quadrático médio do sinal observado e do sinal estimado utilizando o Algoritmo 1 com $\mathcal{D} = \{v_0, v_1\}$ e $\mathcal{D} = \{v_0, v_1, v_2, v_4\}$.

Problema 4.4. Repita o problema anterior utilizando o algoritmo LMS para $\mu \in \{0{,}01, 0{,}1, 0{,}5\}$ e $\mathcal{D} = \{v_0, v_1, v_2, v_4\}$. Compare os resultados do LMS com os resultados do Algoritmo 1.

Problema 4.5. Mostre como obter a equação de atualização do LMS a partir do problema (4.2.6).

Capítulo 5

Análise Vértice-Frequência

Em DSP, a análise tempo-frequência viabiliza o estudo dos sinais nos domínios do tempo e da frequência de forma conjunta, tornando mais evidentes informações que doutro modo poderiam ser mascaradas em representações em um único domínio (tempo ou frequência). Algumas técnicas bastante utilizadas para esse fim são a transformada de Fourier de tempo curto (STFT, sigla proveniente do inglês *short-time Fourier transform*) e as transformadas *wavelets*. Analogamente à análise tempo-frequência, é possível analisar sinais sobre grafos no domínio dos vértices e das frequências espectrais de forma conjunta por meio da análise vértice-frequência (VFA, da expressão *vertex-frequency analysis*). A Seção 5.1 deste capítulo abordará a transformada de Fourier sobre grafos com janelamento (WGFT, do inglês *windowed graph Fourier transform* [70]), uma generalização da STFT para sinais sobre grafos. Um método de *wavelets* [26] com base em funções de *kernel* no domínio espectral é apresentado na Seção 5.2 e a Seção 5.3 apresenta *wavelets* adaptadas ao espectro do grafo.

Considerando que a maior parte dos resultados apresentados neste capítulo foram originalmente derivados para $\mathrm{GSP_L}$ (Capítulo 3), consideraremos, por simplicidade, apenas essa abordagem.

5.1 Transformada de Fourier sobre grafos com janelamento

A STFT permite realizar uma análise espectral dependente do tempo e, portanto, é útil para lidar com sinais não-estacionários de forma a extrair/evidenciar o conteúdo de frequência para cada curto intervalo de tempo. Em DSP, a STFT divide o sinal temporal $x[m]$ em segmentos sobre os quais se aplica a transformada de Fourier:

$$\mathrm{STFT}\{x[m]\}[n,\omega] = \sum_{m\in\mathbb{Z}} x[m]w[m-n]e^{-\jmath\omega m}, \tag{5.1.1}$$

em que $\omega \in \mathbb{R}$ representa a frequência normalizada (rd/amostra), $w[m]$ denota uma função-janela (isto é, uma função que é não-nula apenas por um curto período de tempo e que, tipicamente, é do tipo passa-baixas—ou seja, seu conteúdo espectral concentra-se em frequências baixas; exemplos: Janela de Hann e janela gaussiana) e $w[m-n]$ é a janela transladada para o n-ésima instante de tempo discreto.

Note que, ao definir-se o átomo $w_{n,\omega}[m] = w[m-n]e^{j\omega m}$ como a *translação* da janela $w[m]$ seguida de uma *modulação* (isto é, multiplicação por uma exponencial complexa, base da transformada de Fourier, operação esta equivalente à uma translação espectral), a STFT em (5.1.1) pode ser reescrita como o produto interno

$$\mathrm{STFT}\{x[m]\}[n,\omega] = \langle x, w_{n,\omega} \rangle \,. \tag{5.1.2}$$

Sendo assim, para generalizar essa ferramenta, introduziremos, primeiramente, os conceitos de translação e de modulação em grafos.

5.1.1 Translação em grafos

Em DSP, transladar um sinal $x[m]$ de n amostras é o mesmo que convoluí-lo com a função delta de Kronecker

$$\delta_n[m] \triangleq \begin{cases} 1 & \text{quando } \; m = n \\ 0 & \text{caso contrário} \end{cases} , \tag{5.1.3}$$

tendo em vista que

$$x[m-n] = \sum_{k \in \mathbb{Z}} x[k]\delta_n[m-k] \,. \tag{5.1.4}$$

A operação de convolução temporal pode ser implementada por meio do produto ponto-a-ponto das respectivas transformadas de Fourier dos sinais envolvidos, seguido da transformada inversa de Fourier sobre o resultado do produto.

Em GSP, por sua vez, a GFT de $\boldsymbol{\delta}_n$ (n-ésimo vetor da base canônica de $\mathbb{R}^N$) é dada por $\mathbf{U}^{\mathrm{T}}\boldsymbol{\delta}_n = \bar{\mathbf{u}}_n$, em que $\bar{\mathbf{u}}_n^{\mathrm{T}}$ é a n-ésima linha de $\mathbf{U}$. A translação do sinal $\mathbf{x}$ para o n-ésimo vértice é o sinal no domínio do vértice definido como

$$\mathbf{t}_n^{\mathbf{x}} \triangleq \mathbf{U}\left(\widetilde{\mathbf{x}} \odot \bar{\mathbf{u}}_n\right). \tag{5.1.5}$$

Considere os sinais $\mathbf{x}_1$ e $\mathbf{x}_2$ sobre o grafo $\mathcal{G}$ e defina a *convolução no domínio do vértice* como

$$\mathbf{x}_1 \circledast \mathbf{x}_2 \triangleq \mathbf{U}(\widetilde{\mathbf{x}}_1 \odot \widetilde{\mathbf{x}}_2) \,. \tag{5.1.6}$$

Então, valem as seguintes propriedades:

(P1) $\mathbf{t}_n^{\mathbf{x}_1 \circledast \mathbf{x}_2} = \mathbf{t}_n^{\mathbf{x}_1} \circledast \mathbf{x}_2 = \mathbf{x}_1 \circledast \mathbf{t}_n^{\mathbf{x}_2}$;

(P2) $\mathbf{t}_n^{\mathbf{t}_m^{\mathbf{x}_1}} = \mathbf{t}_m^{\mathbf{t}_n^{\mathbf{x}_1}}$;

(P3) $\sum\limits_{m=0}^{N-1} t_n^{\mathbf{x}_1}[m] = \frac{1}{\sqrt{N}} \sum\limits_{m=0}^{N-1} x_1[m]$.

Note que a interpretação da propriedade (P2) neste caso é diferente de sua análoga em DSP. A translação de um sinal sobre um grafo para um vértice n significa concentrar esse sinal no respectivo vértice em vez de, necessariamente, transladar a função por n amostras; portanto, $\mathbf{t}_n^{\mathbf{t}_m^{\mathbf{x}}} \neq \mathbf{t}_{n+m}^{\mathbf{x}}$ para a maioria dos grafos. No caso de $\mathcal{G}$ ser um anel circular, por exemplo, $\mathbf{t}_n^{\mathbf{t}_m^{\mathbf{x}}} = \mathbf{t}_{(n-1)+(m-1)-1 \bmod N}^{\mathbf{x}}$. Essa definição

de translação também não permite, de forma geral, preservar a energia do sinal em questão.

O objetivo principal da translação definida em (5.1.5) é centralizar uma janela em cada vértice do grafo para então computar sua transformada de Fourier, assim como na STFT. Dessa forma, para que os átomos da WGFT, que serão apresentados posteriormente, sejam localizados no domínio do vértice, é importante que a translação seja capaz de concentrar a janela em cada vértice. O Teorema 5.1 mostra que o valor da janela transladada $t_n^{\widetilde{\mathbf{w}}}$ é relativamente pequeno em vértices distantes de n.

Teorema 5.1. *Considere os vértices n e m tais que o número mínimo de arestas que se deve percorrer para ir de m para n é d_{nm}. Seja $\widetilde{w} : [0, \lambda_{\max}] \to \mathbb{R}$ uma função janela d_{nm}-vezes continuamente diferenciável com $w(0) \neq 0$ e defina $\widetilde{\mathbf{w}} \triangleq [\widetilde{w}(\lambda_0) \ \cdots \ \widetilde{w}(\lambda_{N-1})]^{\mathrm{T}}$. Então:*

$$\frac{|t_n^{\mathbf{w}}[m]|}{\|\mathbf{t}_n^{\mathbf{w}}\|_2} \leq \left[\frac{2\sqrt{N}}{d_{nm}!|w(0)|} \left(\frac{\lambda_{\max}}{4} \right)^{d_{nm}} \right] \sup_{\lambda \in [0, \lambda_{\max}]} |\widetilde{w}^{(d_{nm})}(\lambda)|. \tag{5.1.7}$$

5.1.2 Modulação em grafos

Para se obter localização no domínio da frequência, o teorema da convolução é aplicado em sua forma dual: para concentrar o sinal $\mathbf{x}$ na frequência λ_k, $\mathbf{x}$ é multiplicado ponto-a-ponto com o autovetor $\mathbf{u}_k$. Essa operação é chamada modulação e é definida por:

$$\mathbf{m}_k(\mathbf{x}) \triangleq \mathbf{x} \odot \mathbf{u}_k. \tag{5.1.8}$$

Se o sinal $\mathbf{x}$ concentra-se em torno do autovalor 0 no domínio da frequência, então $\mathbf{m}_k(\mathbf{x})$ concentra-se em λ_k.

5.1.3 Átomos e espectrograma da WGFT

Diferente de (5.1.1), em que a janela transladada w está definida no domínio do tempo, a WGFT é construída com base em uma janela $\widetilde{w} : [0, \lambda_{\max}] \to \mathbb{R}$ definida no domínio da frequência. Definindo $\widetilde{\mathbf{w}} \triangleq [\widetilde{w}(\lambda_1) \ \cdots \ \widetilde{w}(\lambda_N)]^{\mathrm{T}}$, a translação da janela é dada por:

$$\mathbf{t}_n^{\mathbf{w}} = \mathbf{U} \left(\widetilde{\mathbf{w}} \odot \bar{\mathbf{u}}_n \right) = \mathbf{U} \widetilde{w}(\mathbf{\Gamma}) \bar{\mathbf{u}}_n. \tag{5.1.9}$$

Unindo as equações (5.1.8) e (5.1.9), os átomos da WGFT e os coeficientes correspondentes à transformação do sinal $\mathbf{x}$, assim como em (5.1.2), são, respectivamente, definidos como:

$$\mathbf{w}_{n,k} \triangleq \mathbf{m}_k \left(\mathbf{t}_n^{\mathbf{w}} \right) \quad \text{e} \quad x_{n,k}^w \triangleq \langle \mathbf{x}, \mathbf{w}_{n,k} \rangle. \tag{5.1.10}$$

A matriz contendo os N^2 coeficientes da WGFT, $\mathbf{X}^w$, pode ser calculada da seguinte forma: o sinal $\mathbf{x}$ é primeiramente filtrado por $\mathbf{T} = \mathbf{U} \widetilde{w}(\mathbf{\Gamma}) \mathbf{U}^{\mathrm{T}}$ e o resultado $\mathbf{T}\mathbf{x}$ é multiplicado ponto-a-ponto pela n-ésima coluna de $\mathbf{U}$ para gerar a n-ésima coluna de $\mathbf{X}^w$.

O *espectrograma* do sinal $\mathbf{x}$ é a matriz $N \times N$ $\mathbf{S}_x^w$, cujas colunas correspondem aos vértices e as linhas, às componentes espectrais, sendo definida por

$$[\mathbf{S}_x^w]_{k,n} \triangleq |x_{n,k}^w|^2, \quad \text{com } n, k \in \{0, 1, \ldots, N-1\}. \tag{5.1.11}$$

Se a janela $\widetilde{w}$ possui média não-nula, o sinal original $x[m]$ pode ser perfeitamente recuperado a partir dos coeficientes da WGFT pela seguinte fórmula:

$$x[m] = \frac{1}{\|\mathbf{t}_m^{\mathbf{w}}\|_2^2} \sum_{n=0}^{N-1} \sum_{k=0}^{N-1} x_{n,k}^w w_{n,k}[m]. \tag{5.1.12}$$

5.1.4 Exemplos numéricos

Nesta seção, apresentamos espectrogramas de funções contínuas por partes sobre diferentes grafos: grafo caminho (linear não direcionado), grafo formado por grupos de vértices, e o grafo de estações climáticas brasileiras.

Considere, primeiramente, um grafo caminho e um grafo com 3 grupos de vértices, ambos com $N = 60$ vértices. No grafo de grupos, os 20 primeiros vértices pertencem ao primeiro grupo, os 20 vértices intermediários pertencem ao segundo grupo e os últimos 20 vértices estão no terceiro grupo. Vértices pertencentes a um mesmo grupo estão conectados uns com os outros com probabilidade $p_1 = 0{,}5$ enquanto pares de vértices em grupos diferentes estão conectados com probabilidade $p_2 = 0{,}05$. Este grafo é ilustrado na Figura 5.1. Para ambos os grafos, o sinal contínuo por partes é a combinação do autovetor $\mathbf{u}_{15}$ nos primeiros 20 vértices, $\mathbf{u}_{30}$ restrito aos vértices $v_{21}, v_{22}, \cdots, v_{40}$ e $\mathbf{u}_{45}$ restrito aos demais vértices.

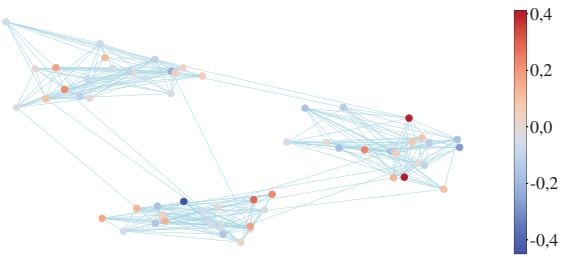
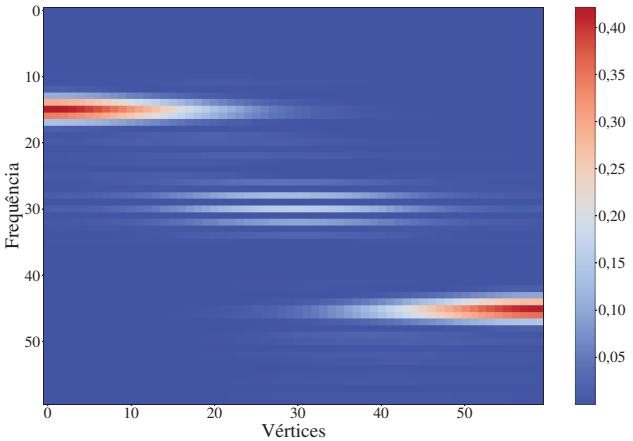


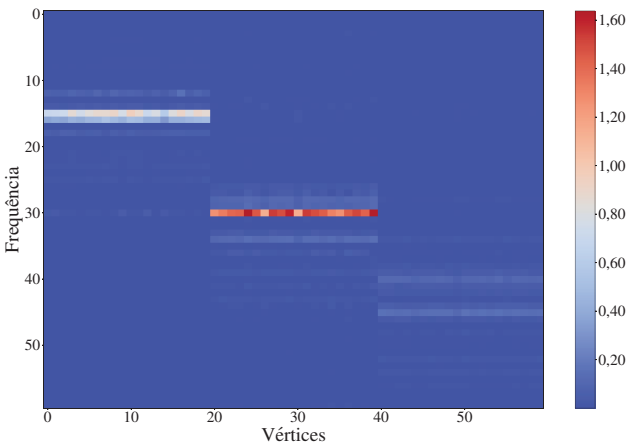
Figura 5.1: Grafo formado por grupos de vértices.

Para ambos os grafos, a janela utilizada para construir os átomos da WGFT é $w(\lambda) = Ce^{-\tau\lambda}$, onde a constante $C = (\sum_{k=1}^{N} e^{-2\tau\lambda_k})^{-1/2}$ com $\tau = 300$ para o grafo caminho e $\tau = 5$ para o grafo de grupos. A Figura 5.2 mostra os espectrogramas obtidos para o grafo caminho e para o grafo de grupos.

Em ambos os espectrogramas, podemos observar faixas de energia em torno das frequências $\lambda_{15}, \lambda_{30}$, e λ_{45}. O compromisso entre localização no domínio do vértice e da frequência pode ser controlado pelo parâmetro τ. Se τ aumenta, a localização no domínio da frequência aumenta enquanto os átomos da WGFT se



(a)



(b)

Figura 5.2: Espectrograma de função contínua por partes sobre (a) o grafo caminho e (b) o grafo de grupos.

tornam mais espalhados no domínio do vértice. No entanto, a propriedade de co-localização vértice-frequência depende da estrutura do grafo e pode variar para diferentes vértices no mesmo grafo. Dada a janela w, $\mathbf{t}_n^{\mathbf{w}}$ estará concentrado no vértice v_n. Porém, se $|U_{mk}|$ for pequeno para algum $m \in \mathcal{N}_1(n)$, $\mathbf{w}_{n,k} = \mathbf{t}_n^{\mathbf{w}} \odot \mathbf{u}_k$ pode não estar localizado ao redor do vértice v_n. Analogamente, $\mathbf{m}_k(\mathbf{w})$ será localizado na frequência λ_k, mas, se $\bar{\mathbf{u}}_{n\ell}$ é pequeno para ℓ próximo a k, então $\mathbf{w}_{n,k}$ não será localizado no vértice v_n no domínio do vértice e em λ_k no domínio da frequência simultaneamente.

A Figura 5.3 apresenta um sinal contínuo por partes definido sobre um grafo de sensores climáticos distribuídos pelo Brasil. Os coeficientes não-nulos da matriz de adjacência são:

$$A_{m,n} \triangleq \frac{\gamma_{mn}}{\sqrt{\left(\sum\limits_{i\in\mathcal{N}_n} \gamma_{in}\right)\left(\sum\limits_{j\in\mathcal{N}_m} \gamma_{jm}\right)}}, \tag{5.1.13}$$

onde $\gamma_{mn} = e^{-\frac{d_{mn}^2 + h_{mn}^2}{\sigma^2}}$, d_{mn} e h_{mn} representam, respectivamente, a distância geodésica e a diferença de altitude entre as estações climáticas m e n, e $\sigma > 0$ é um parâmetro livre. Para que a matriz tenha um melhor condicionamento, todas as suas entradas foram dividas pelo módulo do autovalor de máxima magnitude. O sinal contínuo por partes $\mathbf{x}$ foi artificialmente gerado como:

$$x[n] = \begin{cases} 2u_{10}[n] + \epsilon, & \text{se } n \in \mathcal{N}, \\ u_{100}[n] + \epsilon, & \text{se } n \in \mathcal{S}, \end{cases} \tag{5.1.14}$$

em que $\mathcal{N}$ (norte) e $\mathcal{S}$ (sul) são subconjuntos de vértice destacados em diferentes cores na Figura 5.3(a), e ϵ é gerado de uma distribuição gaussiana de média zero e variância 0,01. A janela gaussiana $\widetilde{w}(\lambda) = e^{-\lambda^2/10}$ foi usada no cálculo dos coeficientes da WGFT. A Figura 5.3(b) mostra o sinal $\mathbf{x}$ e as Figuras 5.3(c)-(d) apresentam os valores absolutos dos coeficientes da WGFT correspondendo aos dois autovalores da matriz Laplaciana cujos coeficientes possuem maior energia.

Como esperado, os vértices da região norte apresentam coeficientes de maior magnitude para frequências próximas ao décimo autovalor, enquanto os vértices da região sul apresentam maiores coeficientes para frequências no entorno de λ_{100}.

Muitos vértices na região norte do Brazil apresentam valores menores que o esperado na Figura 5.3(c). Isso ocorre porque o autovetor $\mathbf{u}_{11}$ apresenta suas entradas concentradas no nordeste do grafo e, consequentemente, assume valores de baixa magnitude no restante do grafo. Os átomos da WGFT $\mathbf{w}_{n,k}$ não são necessariamente localizados simultaneamente nos domínios do vértice e da frequência, pois alguns autovetores podem ser muito concentrados em poucos vértices. De fato, $\mathbf{t}_n^{\mathbf{w}}$ em (5.1.5) é bem localizado no entorno do vértice n, mas se $u_k[n]$ for próximo de zero, então, $w_{n,k}[n] = t_n^{\mathbf{w}}[n]u_k[n]$ em (5.1.10) terá magnitude pequena. De forma similar, $\mathbf{m}_k(\cdot)$ em (5.1.8) é bem localizado em λ_k no domínio da frequência, mas se $\bar{u}_n[k]$ for próximo de zero, $\tilde{w}_{n,k}[k]$ também terá magnitude pequena. Se um autovetor $\mathbf{u}_k$ se concentra fortemente em um único vértice n, assumirá baixa magnitude nos demais vértices (os autovetores devem ter norma unitária), levando ao problema mencionado. Dessa forma, a co-localização no domínio do vértice e da frequência é limitada pela coerência do grafo, definida como: $\mu \triangleq \max\limits_{n,k} |u_k[n]| \leq 1$. O grafo na Figura 5.3 possui $\mu = 0{,}94$ (um valor elevado) e então alguns átomos da WGFT não poderão ser localizados no domínio da frequência e dos vértices simultaneamente.

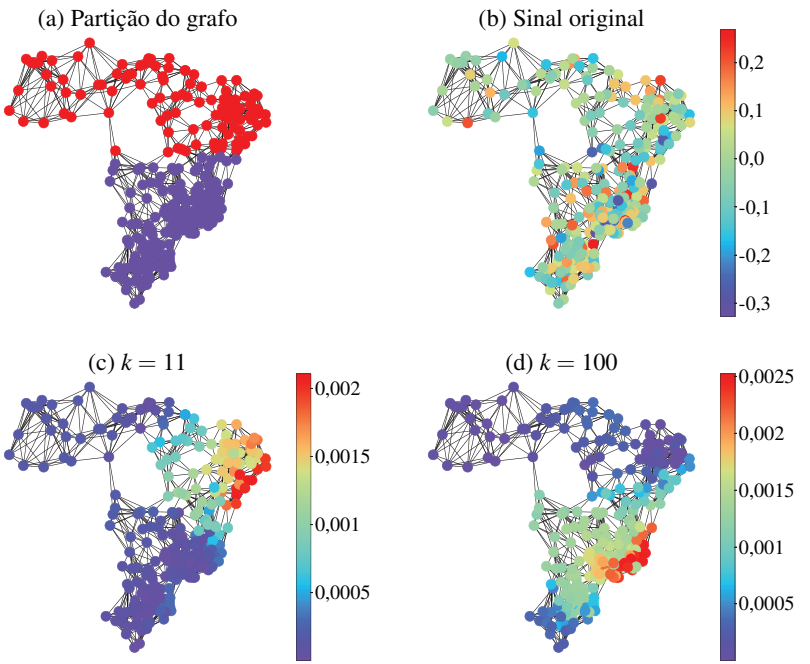


Figura 5.3: Grafo de estações climáticas brasileiras: (a) grafo dividido em dois subconjuntos de vértices ($\mathcal{N}$ e $\mathcal{S}$); (b) sinal original $\mathbf{x}$ em (5.1.14); e (c)-(d) coeficientes da WGFT $x^{w}_{n,k}$ em (5.1.10) nas frequências λ_{11} e λ_{100} , respectivamente.

5.2 Transformada *wavelet* em grafos

A WGFT possui algumas desvantagens, a saber:

- o número de coeficientes na representação cresce de N para N^2;
- é computacionalmente cara e apresenta baixa escalabilidade pois, para construir os átomos da WGFT, é necessário calcular todos os autovetores da matriz Laplaciana, sendo, então, inviável para grafos muito grandes. A translação pode pode ser aproximada por um polinômio de Chebyshev (a ser abordado mais à frente) sobre a matriz $\mathbf{L}$, porém, não se conhece ainda um método para aproximar a etapa de modulação em (5.1.8);
- os átomos da WGFT não favorecem uma representação robusta à presença de ruído;
- a resolução no domínio da frequência é uniforme assim como no caso da STFT.

As transformadas *wavelets*, por outro lado, podem

- apresentar uma representação mais compacta com $N \cdot S$ coeficientes, em que $S < N$;

- ser implementadas sem a necessidade de calcular os autovetores da matriz Laplaciana, a não ser o maior autovalor;
- gerar uma representação estável a depender das escolhas da *wavelet* mãe e da função-escala;
- ter resolução adaptativa de acordo com o princípio de Heisenberg [42].

Wavelets são extremamente úteis para extração de características de dados, sendo utilizadas em diversas áreas, tais como compressão de dados e redução de ruído. Para sinais em grafos, transformadas *wavelets* também têm sido aplicadas na extração de padrões, por exemplo. Os trabalhos pioneiros nessa área são as *wavelets* de difusão [10], [41], definidas no domínio espectral, e as *wavelets* para análise de rede de tráfego [12], em que uma transformada *wavelet* é construída sobre o grafo em questão com base no tamanho do mínimo caminho que conecta dois vértices. Além disso, a transformada *wavelet* em grafos (SGWT) tem sido aplicada no reconhecimento de padrões e em problemas de classificação em conjunto com ferramentas de aprendizado de máquina. Em [11], a SGWT é aplicada na classificação de eventos em redes temporalmente dinâmicas e em [43], os átomos da SGWT são utilizados na extração de atributos para classificação de formas 3D.

A transformada *wavelet* discreta (DWT, sigla do inglês *discrete wavelet transform*) clássica em $L^2(\mathbb{R})$ é gerada pela translação e dilatação de uma única função ψ, denominada *wavelet* mãe , como por exemplo:

$$\psi_{n,s}(t) = a^k \psi(a^s t - n)\,, \quad n, s \in \mathbb{Z}, \tag{5.2.15}$$

onde, geralmente, toma-se $a = 2$.

Para que uma função seja considerada uma *wavelet* mãe, ela deve atender as seguintes propriedades:

$$\text{(P1)} \quad \int_{-\infty}^{\infty} \psi(t)\mathrm{d}t = 0\,, \tag{5.2.16}$$

$$\text{(P2)} \quad \int_{-\infty}^{\infty} |\psi(t)|^2 \mathrm{d}t < \infty\,, \tag{5.2.17}$$

ou seja, deve possuir média zero (nível DC nulo) e energia finita. Como as *wavelets* funcionam como filtros passa-faixa que, para cada nível s, dividem a banda de passagem pela metade (no caso de $a = 2$), seriam necessários infinitos níveis de escala para representar as baixas frequências. Para que todo o espectro possa ser representado por um número finito de níveis, uma função passa-baixas chamada função-escala, ϕ, pode ser introduzida. Dado o sinal contínuo $x(t) \in L^2(\mathbb{R})$, a DWT consiste em identificar os coeficientes de aproximação $\rho[n]$ e detalhe $\varphi[n, s]$, $n, s \in \mathbb{Z}$:

$$x(t) = \sum_{n\in\mathbb{Z}} \rho[n]\phi_n(t) + \sum_{s\in\mathbb{Z}} \sum_{n\in\mathbb{Z}} \varphi[n, s]\psi_{n,s}(t), \tag{5.2.18}$$

em que $\phi_n(t) = a^k \psi(t - n)$. Se $\{\psi_{n,s}(t), \phi_n(t)\}_{n,s\in\mathbb{Z}}$ for uma base ortonormal, os coeficientes da DWT podem ser obtidos por

$$\rho[n] = \langle \phi_n(t), x(t)\rangle, \tag{5.2.19}$$

$$\varphi[n, s] = \langle \psi_{n,s}(t), x(t)\rangle. \tag{5.2.20}$$

A próxima seção apresenta uma generalização dos átomos da transformada *wavelet*, $\{\phi_n(t), \psi_{n,s}(t)\}_{n,s\in\mathbb{Z}}$ como sinais sobre o grafo $\mathcal{G}$ com base na definição de translação em (5.1.5).

5.2.1 Transformada *wavelet* espectral em grafos

Assim como na definição da WGFT, a SGWT é definida por uma função chamada *wavelet* mãe/*kernel*, $\psi : \mathbb{R}^+ \to \mathbb{R}^+$ no domínio da frequência. Essa função deve ser passa-faixa com $\psi(0) = 0$ e $\lim_{z\to\infty} \psi(z) = 0$. Analogamente à (5.2.15), os átomos da transformada *wavelet*, $\boldsymbol{\psi}_{n,s}$, com $(n, s) \in \{1, \ldots, N\} \times \{1, \ldots, S\}$, são sinais construídos a partir da dilatação da *wavelet* mãe, ψ, por um fator $\alpha_s \in \mathbb{R}^+$, e, então, translada para o vértice n, como a seguir:

$$\boldsymbol{\psi}_{n,s} \triangleq \mathbf{U}\psi(\alpha_s\boldsymbol{\Gamma})\bar{\mathbf{u}}_n = \left(\mathbf{U}\psi(\alpha_s\boldsymbol{\Gamma})\mathbf{U}^{\mathrm{T}}\right)\boldsymbol{\delta}_n. \tag{5.2.21}$$

Dado sinal $\mathbf{x}$, os coeficientes da SGWT são definidos como:

$$x^{\psi}_{n,s} \triangleq \langle \mathbf{x}, \boldsymbol{\psi}_{n,s}\rangle, \tag{5.2.22}$$

e, coletando os coeficientes em um vetor $\mathbf{x}^{\psi}$, temos $\mathbf{x}^{\psi} = \boldsymbol{\Psi}^{\mathrm{T}}\mathbf{x}$, onde $\boldsymbol{\Psi} \in \mathbb{R}^{N\times SN}$ é composta $\boldsymbol{\psi}_{n,s}$ nas sua colunas.

Os atomos de *Wavelet* são ortogonais ao autovetor $\mathbf{u}_0$ associado a $\lambda_0 = 0$,

$$\begin{aligned}
\langle \boldsymbol{\psi}_{n,s}, \mathbf{u}_0\rangle &= \sum_{m=1}^{N}\left(\sum_{k=0}^{N-1}\psi(\lambda_k)\overline{u}_k[n]u_k[m]\right)u_0[m] \\
&= \sum_{k=0}^{N-1}\psi(\lambda_k)\overline{u}_k[n]\sum_{m=1}^{N}u_k[m]u_0[m] \\
&= \sum_{k=0}^{N-1}\psi(\lambda_k)\overline{u}_k[n]\sum_{m=1}^{N}\delta[k] \\
&= \psi(\lambda_0)\overline{u}_0[n] = 0
\end{aligned}$$

em que a última igualdade decorre do fato de que $\psi(0) = 0$. Além disso, os átomos são quase ortogonais aos autovetores associados a autovalores pequenos. Dessa forma, para que as baixas frequências sejam representadas, é necessário introduzir uma função-escala como no caso do DSP clássico. A função-escala $\phi : \mathbb{R}^+ \to \mathbb{R}^+$ deve ser um filtro passa-baixas tal que $\phi(0) > 0$ e $\lim_{z\to\infty} \phi(z) = 0$.

Se a *wavelet* mãe ψ for concentrada no domínio espectral, então os átomos da SGWT serão concentrados em cada autovalor da matriz Laplaciana. A localização no domínio do vértice, por outro lado, requer uma análise mais cuidadosa. Para simplificar, se ψ é suficientemente suave na vizinhança do $\lambda_0 = 0$, então $\boldsymbol{\psi}_{n,s}$ será concentrado no vértice n (ex.: $\frac{\boldsymbol{\psi}_{n,s}}{\|\boldsymbol{\psi}_{n,s}\|_2}$ será pequeno nos vértices distantes de n).

5.2.2 Reconstrução

Em processamento de sinais, diversas aplicações requerem que o sinal original ou alguma transformação sua seja recuperada para o domínio original. Um sinal $\tilde{\mathbf{x}}$

pode ser reconstruído a partir dos seus coeficientes da SGWT $\mathbf{x}^{\psi}$ pelo seguinte problema de mínimos quadrados:

$$\tilde{\mathbf{x}} \triangleq \underset{\mathbf{y}}{\operatorname{argmin}} \|\mathbf{x}^{\psi} - \mathbf{\Psi}^{\mathrm{T}}\mathbf{y}\|_2. \tag{5.2.23}$$

5.2.3 *Kernels* para *wavelets*

As seguintes funções-escala ϕ e *wavelets* mães ψ têm sido utilizadas pela literatura de GSP :

(a) *Spline* cúbico [26];

$$\psi(\lambda; a, b, \lambda_1, \lambda_2) = \begin{cases} \lambda_1^{-a}\lambda^a, & \lambda < \lambda_1, \\ p(\lambda), & \lambda_{\leq}\lambda \leq \lambda_2, \\ \lambda_2^b\lambda^{-b}, & \lambda > \lambda_2, \end{cases} \tag{5.2.24}$$

em que $p(\lambda)$ é um polinômio cúbico tal que $p(\lambda_1) = p(\lambda_2) = 1$, $p'(\lambda_1) = a/\lambda_1$ e $p'(\lambda_2) = -b/\lambda_2$. Escolhendo $a = b = 2$, $\lambda_1 = 1$ e $\lambda_2 = 2$, temos $p(\lambda) = -5 + 11\lambda - 6\lambda^2 + \lambda^3$. As restrições que definem $p(\lambda)$ garantem a continuidade de ψ e ψ' em λ.

Os níveis das *wavelets* α_s são escolhidos tais que sejam logaritimicamente equi-espaçados entre $\alpha_S = \lambda_2/\lambda_{\min}$ e $\alpha_1 = \lambda_2/\lambda_{\max}$ com $\lambda_{\min} = \lambda_{\max}/K$ e K é um parâmetro de projeto que pode ser escolhido de forma livre. Nesse caso, $\psi(\alpha_S\lambda)$ se comporta como um polinômio mônico para $\lambda < \lambda_{\max}$ como desejado.

A função-escala é escolhida como

$$\phi(\lambda) = \gamma \exp\left\{-\left(\frac{\lambda}{0.6\lambda_{\max}}\right)^4\right\},$$

onde $\gamma = \max_{\lambda\in[0,\lambda_{\max}]} \psi(\lambda)$.

(b) Meyer [32]

$$\psi(\lambda) = \begin{cases} \operatorname{sen}\left(\frac{\pi}{2}p\left(\frac{\lambda}{a} - 1\right)\right), & a < \lambda \leq 2a, \\ \cos\left(\frac{\pi}{2}p\left(\frac{\lambda}{2a} - 1\right)\right), & 2a < \lambda \leq 4a, \\ 0, & \text{c.c.} \end{cases} \tag{5.2.25}$$

$$\phi(\lambda) = \begin{cases} 1 & 0 < \lambda \leq a, \\ \cos\left(\frac{\pi}{2}p\left(\frac{\lambda}{a} - 1\right)\right), & a < \lambda \leq 2a, \\ 0, & \text{c.c.}, \end{cases} \tag{5.2.26}$$

em que $p(\lambda) = \lambda^4(35 - 84\lambda + 70\lambda^2 - 20\lambda^3)$. Aqui, usaremos $(a, M) = (\frac{2}{3}, 2)$.

(c) Hann [72], com parâmetros $(K, a_0, a_1, R) = (1, \frac{1}{2}, \frac{1}{2}, 3)$;

$$\psi(\lambda) = \begin{cases} \sum_{k=0}^{K} a_k \cos\left(2\pi k\left(\frac{S+1-R}{R\lambda_{\max}}\right)\right), & -\frac{R\lambda_{\max}}{S+1-R} < \lambda < 0, \\ 0, & \text{c.c.}, \end{cases} \tag{5.2.27}$$

com $2 < R < S$ e $K < R/2$. Para cada índice de escala $s \in \{0, \ldots, S\}$, as dilatações das *wavelets* são dadas por

$$\psi_s(\lambda) = \psi\left(\lambda - s\frac{\lambda_{\max}}{S + 1 - R}\right). \tag{5.2.28}$$

(d) Filtro ideal [30]. Para $s \in \{0, ..., S-1\}$,

$$\psi_s(\lambda) = \begin{cases} 1, & \mu_s < \lambda < \mu_{s+1}, \\ 0, & \text{c.c.} \end{cases} \tag{5.2.29}$$

e

$$\phi(\lambda) = \begin{cases} 1, & 0 < \lambda < \mu_1 \\ 0, & \text{c.c.}, \end{cases} \tag{5.2.30}$$

em que $\mu_1, ..., \mu_S$ são as frequências de corte dos filtros ideais. Esses filtros ideais, assim como os filtros de Hann, não seguem exatamente a construção por dilatação da função mãe proposta na definição da SGWT, mas estão incluídos nesta seção por serem utilizados para as mesmas aplicações que os demais *kernels*[1].

5.2.4 Aproximação por polinômios de Chebyshev

A construção dos átomos de *wavelet* em (5.2.21) depende da decomposição espectral da matriz Laplaciana, que, como mencionado anteriormente nesta seção, se torna inviável para problemas com alta dimensão. Os filtros podem, alternativamente, ser aproximados por polinômios de Chebyshev sobre a matriz Laplaciana e a filtragem ser realizada diretamente no domínio dos vértices.

Os polinômios de Chebyshev são definidos no intervalo $[-1, 1]$ e são gerados segundo a recursão:

$$C_q(x) = C_{q-1}(x) - 2xC_{q-2}(x),\ x \in [-1, 1].$$

Dada a função $f \in L^2\left([-1,1], \frac{1}{\sqrt{1+x^2}}\right)$, o espaço das funções quadraticamente integráveis em $[-1, 1]$ com respeito à medida $\frac{1}{\sqrt{1+x^2}}$, sua aproximação de ordem Q é dada por $\tilde{f}(x) = \frac{1}{2}c_0 + \sum_{q=1}^{Q} c_q C_q(x)$ com

$$c_q = \int_{-1}^{1} \frac{C_q(x)f(x)}{\sqrt{1+x^2}} \mathrm{d}x = \frac{2}{\pi} \int_0^{\pi} \cos(k\theta) f(\cos(\theta)) \mathrm{d}\theta.$$

Fixando uma escala de *wavelet* s, a matriz $\boldsymbol{\Psi}_s$ com $\boldsymbol{\psi}_{n,s},\ n \in \{0, \ldots, N-1\}$ em suas colunas pode ser aproximada por polinômios de Chebyshev sobre a matriz Laplaciana. Primeiramente, para abranger o espectro inteiro da matriz, é necessário mapear o intervalo $[0, \lambda_{\max}]$ em $[-1, 1]$, e então a construção dos polinômios em $\mathbf{L}$ é definida como:

$$C_q(\mathbf{L}) = 2\overline{\mathbf{L}}C_{q-1}(\mathbf{L}) - C_{q-2}(\mathbf{L}), \tag{5.2.31}$$
$$\text{com } \overline{\mathbf{L}} \triangleq \frac{2}{\lambda_{\max}}\mathbf{L} - \mathbf{I},\ C_0(\mathbf{L}) \triangleq \mathbf{I},\ \text{e } C_1(\mathbf{L}) \triangleq \overline{\mathbf{L}},$$

e os coeficientes de aproximação $\psi_s(\mathbf{L}) \triangleq \mathbf{U}\psi_s(\boldsymbol{\Gamma})\mathbf{U}^{\mathrm{T}}$ em (5.2.21) via

$$\widetilde{\psi}_s(\mathbf{L}) \triangleq \frac{1}{2}c_0 + \sum_{q=1}^{Q} c_{s,q} C_q(\mathbf{L}) \tag{5.2.32}$$

[1] Outras opções para *wavelet kernels* são apresentadas em [32].

são

$$c_{s,q} \triangleq \frac{2}{\pi}\int_0^{\pi} \cos(q\theta)\psi_s\left(\frac{\lambda_{\max}}{2}(\cos\theta+1)\right)\mathrm{d}\theta\,. \tag{5.2.33}$$

Essa aproximação é válida para qualquer matriz real simétrica com $\lambda_{\min}$ não necessariamente nulo tal como a matriz de adjacência de um grafo não-direcionado. Nesse caso, $\overline{\mathbf{A}} \triangleq \frac{2}{\lambda_{\max}-\lambda_{\min}}(\mathbf{A}-\lambda_{\min}\mathbf{I})-\mathbf{I}$ e

$$c_{s,q} \triangleq \frac{2}{\pi}\int_0^{\pi} \cos(q\theta)\psi_s\left(\frac{\lambda_{\max}}{2}(\cos\theta+1)+\lambda_{\min}\right)\mathrm{d}\theta\,. \tag{5.2.34}$$

Os polinômios de Chebyshev em (5.2.31) permitem que os átomos da SGWT sejam calculados mais rapidamente e $C_q(\mathbf{L})$ é calculada apenas uma vez para todos os filtros. Apenas os coeficientes em (5.2.33) necessitam ser atualizados. Além disso, o cálculo de $\widetilde{\psi}_s(\mathbf{L})\mathbf{x}$ com a soma na equação (5.2.32) é dominado pela multiplicação matriz-vetor $\mathbf{Lx}$ e, então, o custo computacional é proporcional ao número de arestas $|\mathcal{E}|$ (pequeno para grafos esparsos).

5.2.5 Computação Distribuída

Considere a aproximação polinomial de Chebyshev de ordem Q, $\widetilde{\mathbf{H}}$, do filtro $\mathbf{H}$, de forma que $\widetilde{H}_{nm} = 0$ para todo vértice $m \notin \mathcal{N}(n,Q)$. Então, os coeficientes aproximados de SGWT podem ser calculados localmente em cada vértice do grafo. Cada vértice n conhece [71]:

(1) o sinal $\mathbf{x}$ restrito à vizinhança de comprimento Q;

(2) os pesos das arestas que conectam o vértice n com cada um de seus vizinhos $(A_{nm}, m \in \mathcal{N}(n,Q))$;

(3) os coeficientes do polinômio de Chebyshev de ordem Q, $c_{r,q}$, que podem ser pré-computados de forma central e enviados para cada vértice ou computados localmente em cada vértice;

(4) um limite superior para $\lambda_{\max}$.

Cada vértice calcula $(\widetilde{\mathbf{H}}\mathbf{x})_n$ trocando informação com seus vizinhos de forma iterativa.

5.3 SGWT adaptada ao espectro

Um ponto importante do projeto de filtros em grafos é a distribuição do espectro. As *wavelets* mães mencionadas anteriormente são adaptadas ao intervalo $[0, \lambda_{\max}]$. No entanto, como matrizes Laplacianas podem ter autovalores irregularmente distribuídos, é possível obter filtros sem autovalores dentro da banda de passagem. As Figuras 5.4 (a)-(c) mostram os *kernels* de SGWT para três diferentes grafos onde se pode observar que, para o grafo Cometa, a maior parte dos autovalores se concentra em valores pequenos e distantes do autovalor máximo. Considerando essas irregularidades no espectro, em [72], *wavelets* são construídas a partir de uma deformação do *kernel* utilizando uma aproximação da função espectral acumulada apresentada na Definição 5.1.

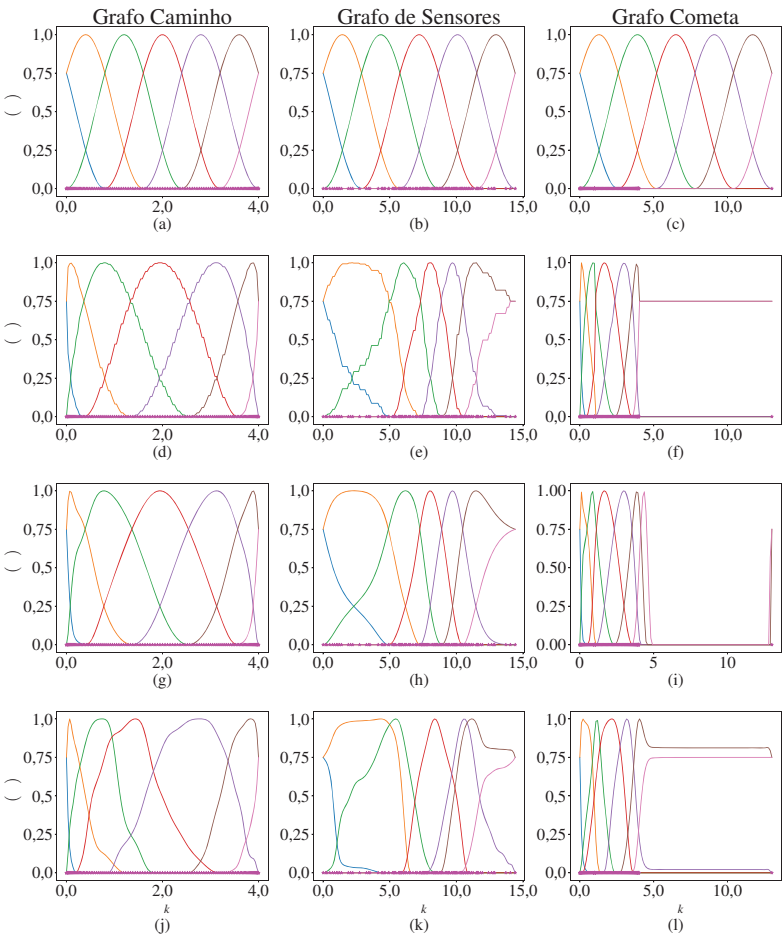


Figura 5.4: Hann *wavelet* adaptada ao espectro. Os "*" em rosa representam os autovalores distribuídod no intervalo $[0, \lambda_{\max}]$ no eixo horizontal. (a)-(c) *Kernels* não adaptados. (d)-(f) *Wavelets* adaptadas ao espectro pela FEA. (g)-(i) *Wavelets* Hann adaptadas ao espectro usando interpolação por *spline* cúbico para aproximação. A interpolação foi realizada com $K = 12$ pontos como em (5.3.36). (j)-(l) *Wavelets* de Hann adaptadas ao espectro pela aproximação em (5.3.38).

Definição 5.1. *A função espectral acumulada (FEA) de uma matriz (adjacência ou Laplaciana) é*

$$P_\lambda(z) \triangleq \frac{1}{N}\sum_{k=0}^{N-1} \mathbb{1}_{[\lambda_k \leq z]}, \tag{5.3.35}$$

em que $\mathbb{1}_{[\lambda_k \leq z]}$ *é a função indicadora do intervalo* $[\lambda_k \leq z]$.

Considere a matriz Laplaciana como bloco de construção da SGWT, então as *wavelets* adaptadas ao espectro e a função-escala são dadas por $\tilde{g}_r(\lambda) = g_r(P_\lambda(\lambda))$ e $\tilde{h}_r(\lambda) = h_r(P_\lambda(\lambda))$, respectivamente. As Figuras 5.4(d)-(f) mostram os *kernels* deformados. Como a FEA pode não ser contínua, é desejável utilizar uma aproximação contínua da mesma para que se possa construir uma *wavelet* suave e bem localizada no domínio do vértice. A FEA pode ser, por exemplo, aproximada por uma interpolação por *spline* cúbico [21] cujos pontos de interpolação são:

$$\left\{\lambda_{\theta_k}, \frac{k}{K-1}\right\},\ k \in \{0, ..., K-1\}, \quad \theta_k = \frac{Nk}{K-1}. \tag{5.3.36}$$

As Figuras 5.4(g)-(i) mostram a aproximação de *wavelets* de Hann *spline* cúbicos por partes com $K = 12$ para o grafo de caminho, grafo aleatório de sensores e grafo cometa, respectivamente. Os três grafos, não direcionados e sem pesos, são descritos na Tabela 5.1.

Tabela 5.1: Descrição dos grafos

Grafo	N	Descrição
Grafo de Caminho	120 vértices	Cada vértice está conectado a 2 vértices, exceto pelos vértices de extremidade, que se conectam a apenas 1 vértice.
Grafo de Sensores	120 vértices	Os vértices são conectados por arestas geradas de forma aleatória, mantendo a centralidade do grafo igual 8 (em média, cada vértice está conectado a outros 8 vértices).
Grafo cometa	120 vértices	Um único vértice é conectado a 30 vértices e os demais vértices organizados como um grafo de caminho.

A interpolação da FEA depende de calcular K autovalores da matriz Laplaciana. Caso esse cálculo seja computacionalmente custoso, a FEA pode ser aproximada por polinômios de Chebyshev na matriz Laplaciana do grafo (5.2.31). Definindo a função $\Theta_t(\lambda) = \mathbb{1}_{[\theta_t \leq \lambda]}$ onde $\theta_1, ..., \theta_T$, são T valores equi-espaçados em $[0, \lambda_{\max}]$, então o número de autovalores da matriz Laplaciana menores ou iguais a θ_t é

$$\sigma_t = \operatorname{tr}\{\Theta_t(\mathbf{L})\} = \mathbb{E}\{\mathbf{y}^{\mathrm{T}}\Theta_t(\mathbf{L})\mathbf{y}\} \approx \frac{1}{L}\sum_{\ell=1}^{L} \mathbf{y}_\ell^{\mathrm{T}}\Theta_t(\mathbf{L})\mathbf{y}_\ell \tag{5.3.37}$$

$$\approx \frac{1}{L}\sum_{\ell=1}^{L} \mathbf{y}_\ell^{\mathrm{T}}\tilde{\Theta}_t(\mathbf{L})\mathbf{y}_\ell. \tag{5.3.38}$$

onde (5.3.37) se deve ao estimador estocástico do traço de Hutchinson [29], $\tilde{\Theta}_t(\mathbf{L})$ em (5.3.38) é aproximação de Jackson-Chebyshev de $\Theta_t(\mathbf{L})$, como em (5.3.39), e $\mathbf{y}$ é um vetor de variáveis aleatórias Gaussianas independentes. Os vetores $\mathbf{y}_\ell$ são amostrados da mesma distribuição do vetor $\mathbf{y}$ e a FEA é, então, aproximada pela interpolação por polinômios cúbicos nos pontos $\left\{\theta_t, \frac{\sigma_t}{N}\right\}$.

5.3.1 Experimento numérico

A Figura 5.5 mostra a aproximação do *spline* cúbico, *kernels* de Meyer e Hann , e filtros ideais por polinômios de Chebyshev de ordem 50.

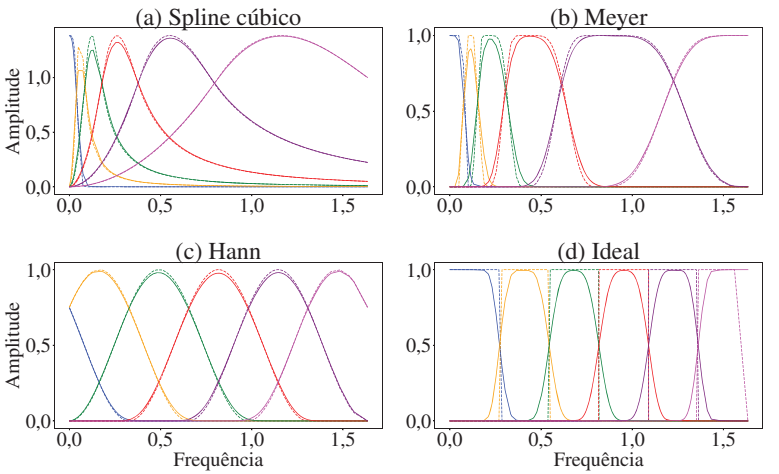


Figura 5.5: *Wavelets* e suas aproximações no domínio da frequência para $S = 5$. Um filtro passa-baixas (em azul) e 5 filtros passa-faixa (em amarelo, verde, vermelho, roxo, e magenta) são mostrados. As linhas tracejadas correspondem função *wavelet* verdadeira em $[0, \lambda_{\max}]$. As linhas sólidas mostram as correspondentes aproximações por polinômios de ordem 50 dos seguintes *kernels*: (a) *spline* cúbico; (b) Meyer; (c) Hann ; e (d) filtro ideal.

Os filtros ideais foram aproximados pelos polinômios de Jackson-Chebyshev $\sum_{q=0}^{Q} a_{s,q}\mathbf{L}^q$ com

$$a_{s,0} = \frac{1}{2}c_{s,0} \quad \text{e} \quad a_{s,q} = \gamma_{q,Q}c_{s,q}, \tag{5.3.39}$$

onde

$$\gamma_{q,Q} = \frac{\left(1 - \frac{q}{Q+2}\right)\operatorname{sen}\left(\frac{\pi}{Q+2}\right)\cos\left(\frac{q\pi}{Q+2}\right) + \frac{q}{Q+2}\cos\left(\frac{\pi}{Q+2}\right)\operatorname{sen}\left(\frac{q\pi}{Q+2}\right)}{\operatorname{sen}\left(\frac{\pi}{Q+2}\right)}.$$

Essa aproximação é capaz de cancelar os *ripples* ao custo de expandir a faixa de transição e é útil para garantir a atenuação fora da faixa de passagem.

Como mostrado na Figura 5.5, o *spline* cúbico fornece maior localização nas baixas frequências. A *wavelet* Meyer precisa de um polinômio de ordem alta para

garantir boa aproximação. A matriz Laplaciana é geralmente esparsa, mas quanto maior o grau do polinômio de Chebyshev, mais densa a aproximação pode ficar. Se o grafo for muito grande, uma aproximação de ordem alta pode ser inviável não só pelo número de multiplicações mas também para o armazenamento. Para obter uma melhor aproximação de baixa ordem, o *kernel* da *wavelet* pode ser construído com base em funções senoidais [72], [57]. A *wavelet* baseada no *kernel* de Hann, mostrada na Figura 5.5(c) apresenta faixas de tamanho regulares e, então, podem ser deformadas por $\omega(\lambda) = \log(\lambda)$ para obter maior resolução nas frequências baixas [72].

5.4 Problemas

1. Demonstre as propriedades P1, P2 e P3 do operador de translação da Seção 5.1.1.

2. Mostre que o operador de translação da Seção 5.1.1 é isométrico no grafo dado por um anel, isto é, $\|\mathbf{t}_n^{\mathbf{x}}\mathbf{x}\| = \|\mathbf{x}\|$. Mostre ainda que, no caso geral,

$$\tilde{x}^2[0] \leq \|\mathbf{t}_n^{\mathbf{x}}\|_2^2 \leq N\mu^2\|\mathbf{x}\|_2^2, \tag{5.4.40}$$

   onde $\mu = \max_{n,k} |u_{nk}|$.

3. Usando o Teorema 5.1, mostre que se $\widetilde{w}(\lambda) = e^{\tau\lambda}$, então,

$$\frac{|t_n^{\widetilde{\mathbf{w}}}[m]|}{\|\mathbf{t}_n^{\widetilde{\mathbf{w}}}\|_2} \leq \sqrt{\frac{2N}{d_{nm}\pi}} \left(\frac{\tau\lambda_{\max}e}{4d_{nm}}\right)^{d_{nm}} e^{-\frac{1}{12d_{nm}+1}}, \tag{5.4.41}$$

   onde d_{nm} é o número mínimo de arestas que se deve percorrer para ir de m para n.

   (dica: utilize a aproximação de Stirling $x! \geq \sqrt{2\pi x}\,(\frac{m}{2})^x\, e^{\frac{1}{12x+1}}$)

4. Mostre que os átomos da SGWT, $\boldsymbol{\psi}_{n,s}$, são quase ortogonais ao autovetor $\mathbf{u}_k$ associado a λ_k, com $\psi(\lambda_k) < \epsilon$.

5. Construa um grafo de sensores aleatório com $N = 64$ vértices (pode utilizar a função *Sensors* da biblioteca *PyGSP*) e compute os respectivos átomos da WGFT utilizando $\tilde{w}(\lambda) = Ce^{-\tau\lambda}$, onde C é tal que $\|\widetilde{\mathbf{w}}\|_2 = 1$.

   (a) Escolhendo $\tau = 3$, plote alguns átomos da WGFT nos domínios do vértice e da frequência. Observe se há casos em que o átomo não está localizado na região esperada, por exemplo, $w_{n,k}[n]$ ou $\widetilde{w}_{n,k}[k]$ são pequenos. Nesses casos, observe a entrada da matriz de autovetores $\mathbf{U}$, u_{nk}, o que se pode concluir?

   (b) Repita o experimento do item anterior com $\tau = 10$. Como variou a localização dos átomos nos domínios do vértice e da frequência?

   (c) Construa o sinal $\mathbf{x}$ tal que:

$$x[n] = \begin{cases} u_{16}[n], & |u_{1[n]}| > 0, \\ u_{48}[n], & \text{caso contrario.} \end{cases} \tag{5.4.42}$$

   Plote os espectrogramas utilizando os átomos da WGFT construídos nos itens (a) e (b). O que você observa?

Capítulo 6

Exemplos e Aplicações

Nesta seção, pretende-se ilustrar o uso dos conceitos de GSP a cenários com dados estruturados em rede, a fim de ressaltar como a consideração acerca do domínio subjacente ao sinal provê informação crucial para o melhor tratamento dos dados. Os exemplos envolverão a remoção de ruído em uma rede de sensores, o problema de estimação de dados climáticos sobre um grafo de cidades brasileiras (uma boa oportunidade para comentar as técnicas de reconstrução de sinal), e uma aplicação de GSP a compressão de imagens de *light-field*. Concluindo a seção, serão mencionados diversos outros contextos de aplicação encontrados na literatura.

6.1 Remoção de ruído em rede de sensores

Consideremos o sinal suave definido sobre o grafo de Minnesota, como na Fig. 6.1a, com componentes dadas por

$$s_i = \cos\big(\mathrm{coord}_x(v_i)\big) + \mathrm{sen}\left(\mathrm{coord}_y(v_i)\right), \tag{6.1.1}$$

em que coord_x e coord_y indicam as coordenadas geográficas dos vértices sobre o estado de Minnesota.

Ao sinal $\mathbf{s}$ foi adicionado uma realização $\mathbf{w}$ de um ruído gaussiano de média nula e desvio padrão igual a 20% da amplitude de $\mathbf{s}$, resultando no sinal mostrado na Fig. 6.1b. A análise espectral do sinal original e do ruído gaussiano mostrou que, embora o ruído seja aproximadamente branco, o sinal — por ser suave — concentra sua energia em componentes de baixa frequência, em autovalores de módulo aproximadamente $0 \leq \gamma_i \leq 0{,}5$ (vide Fig. 6.1c). Por isso, criou-se um filtro passa-baixas ideal com espectro $\widetilde{\mathbf{h}}_{LPF}$, de ganho unitário e frequência de corte $\gamma_{\mathrm{corte}} = 0{,}5$, a fim de reduzir a influência do ruído sobre o sinal.

O sinal obtido após a filtragem passa-baixas ideal é mostrado na Fig. 6.1d. A raiz quadrada do erro médio quadrático (RMSE, *root-mean-square error*) em relação ao sinal original caiu de

$$\|\mathbf{s} - (\mathbf{s} + \mathbf{w})\|_2 = \|\mathbf{w}\|_2 \approx 44{,}6\% \tag{6.1.2}$$

para

$$\|\mathbf{s} - \mathbf{U}\big(\widetilde{\mathbf{h}}_{LPF} \odot (\widetilde{\mathbf{s}} + \widetilde{\mathbf{w}})\big)\|_2 \approx 18{,}9\%, \tag{6.1.3}$$

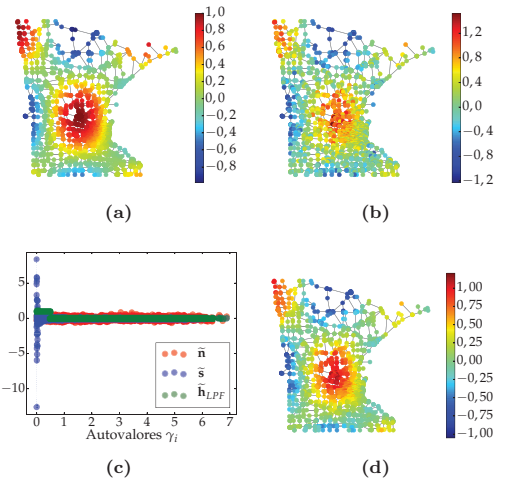


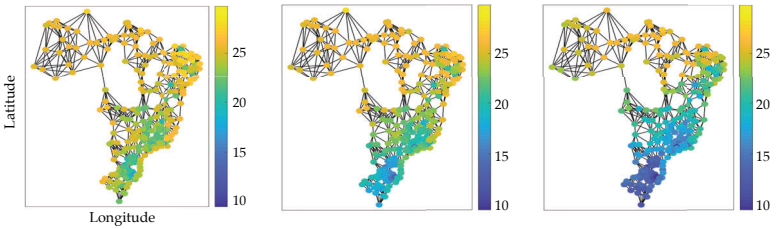
Figura 6.1: (a) Sinal suave sobre o grafo de Minnesota. (b) Resultado da adição de um ruído gaussiano com desvio padrão de 20% da amplitude do sinal original. (c) Espectro do sinal original, do ruído gaussiano e do filtro passa-baixas ideal. (d) Sinal resultante da filtragem passa-baixas.

uma redução pela metade. É importante observar que, por mais simples que seja o exemplo, ele ressalta a importância de se considerar a estrutura subjacente a sinais definidos em domínios irregulares: a alta compactação de energia do sinal $\mathbf{s}$, por exemplo, só foi possível graças à decomposição na base de autovetores do Laplaciano do grafo que lhe serve de domínio. Com estas considerações, técnicas simples podem obter resultados úteis e relevantes em contextos como este.

6.2 Estimação de sinais climáticos

Este exemplo se concentra na aplicação dos conceitos de amostragem e recuperação de sinais em grafos para a estimação de sinais climáticos. Para tanto, serão utilizados sinais de temperatura e índices pluviométricos de estações meteorológicas brasileiras. Inspirado pela aplicação de compressão de sinais definidos sobre grafos em [60], verifica-se que o sinal extraído de uma distribuição espacial de estações meteorológicas medindo sinais climáticos locais é geralmente suave, o que permite sua aproximação por um sinal sobre grafo de faixa limitada. A vantagem de representar um sinal sobre grafo de sinais climáticos por um número reduzido de suas amostras é notória, pois possibilita estimar índices climáticos em certas regiões sem ter um elemento sensor naquele local, reduzindo a quantidade de dispositivos sensores e de dados coletados/armazenados/transmitidos.

Buscando comparar o desempenho das estratégias adaptativas apresentadas no Capítulo 4 com o método tradicional de reconstrução de sinais sobre grafos segundo o Algoritmo 1 daquele capítulo, foram organizadas duas bases de dados, na forma



(a) **(b)** **(c)**

Figura 6.2: Sinais em grafos ilustrando as temperaturas médias mensais das estações meteorológicas brasileiras durante o período de 1961-1990. Meses: (a) Janeiro, (b) Abril e (c) Julho.

de tabelas, do site do Instituto Nacional de Meteorologia (INMET):[1] a primeira tabela contém as coordenadas de latitude e longitude de estações meteorológicas ativas, enquanto a segunda apresenta uma temperatura média mensal registrada em algumas dessas estações, durante o período de 1961-1990. Destes dados obtém-se um total de 299 nós para o grafo $\mathcal{G} = (\mathcal{V}, \mathbf{A})$ a ser construído, em que cada um desses vértices representa uma estação meteorológica. Assim, um nó v_n do grafo $\mathcal{G}$ e seu valor de sinal $x_n[k]$ correspondem, respectivamente, às coordenadas geográficas e à temperatura média da estação meteorológica associada a determinado mês.

Como não há conexões explícitas entre estações meteorológicas, há alguma liberdade no projeto do conjunto de arestas $\mathcal{E}$ e escolha dos pesos $A_{i,j}$. Para simplificar, as arestas do grafo foram construídas conectando um vértice v_n a seus $K = 8$ nós vizinhos mais próximos [68], considerando que a distância entre dois vértices do grafo é dada pela fórmula de Haversine, que avalia a distância entre pontos usando suas coordenadas de latitude e longitude, de forma que

$$A_{i,j} = \begin{cases} \exp\left(-\frac{d_{\mathrm{H}}(i,j)^2}{2\theta^2}\right), & \text{se } (v_i, v_j) \in \mathcal{E} \\ 0 & , \text{ caso contrário} \end{cases} \tag{6.2.4}$$

em que $d_{\mathrm{H}}(i,j)$ é a distância de Haversine entre os vértices v_i e v_j, θ é escolhido como $2 \cdot 10^3$ e a condição $(v_i, v_j) \in \mathcal{E}$ verifica se a aresta ligando os nós v_i e v_j faz parte do conjunto de arestas $\mathcal{E}$ do grafo. Assim, com base neste procedimento, foram obtidos os grafos ilustrados na Fig. 6.2.

Para decidir quantos componentes de frequência são necessários para representar o sinal com um erro de reconstrução razoável, avalia-se o *average reconstruction error* (ARE) $\|\mathbf{x} - \mathbf{x}^{(P)}\|_2 / \|\mathbf{x}\|_2$ para diferentes estimativas $\mathbf{x}^{(P)}$ usando apenas P componentes frequenciais do sinal original $\mathbf{x}$, correspondente aos dados do mês julho representados na Fig. 6.2(c). O sinal reconstruído $\mathbf{x}^{(P)}$ é obtido ordenando os valores absolutos de $\widetilde{\mathbf{x}}$ obtidos de (3.1.11) e selecionando os índices p dos P maiores componentes para formar o conjunto auxiliar $\mathcal{F}_P$ que definirá o suporte de tamanho P. Com base nestes índices $p \in \mathcal{F}_P$, escolhe-se o p-ésimo autovetor de $\mathbf{U}$ e a p-ésima componente frequencial de $\widetilde{\mathbf{x}}$ para definir $\mathbf{U}_{\mathcal{F}_P}$ e $\widetilde{\mathbf{x}}_{\mathcal{F}_P}$. Então, a estimativa $\mathbf{x}^{(P)}$ usando P componentes é dada por $\mathbf{U}_{\mathcal{F}_P}\widetilde{\mathbf{x}}_{\mathcal{F}_P}$. A Fig. 6.3 exibe o

[1] Fonte: Banco de Dados Meteorológicos para Ensino e Pesquisa (BDMEP) do INMET. Acesso gratuito, disponível em: `http://www.inmet.gov.br/portal/index.php?r=bdmep/bdmep`

percentual de ARE para diferentes valores P no intervalo $[50, 250]$. Considerando que um erro de reconstrução de 2,5% é aceitável na aplicação atual, aproxima-se o sinal original pelos seus $P = 200$ maiores componentes frequenciais. A partir desta suposição, pode-se definir o conjunto de faixa limitada aproximada $\mathcal{F} = \mathcal{F}_{\mathcal{P}}$, onde $|\mathcal{F}| = P = 200$.

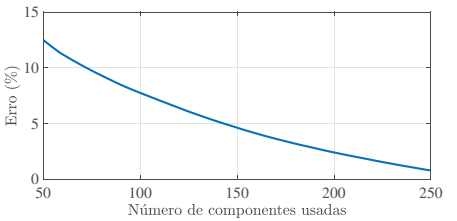


Figura 6.3: Erro de reconstrução percentual quando o sinal original é comprimido usando P componentes frequenciais.

Com base neste sinal $\mathcal{F}$-eesparso $\mathbf{x}^{(P)}$, é necessário escolher o valor $|\mathcal{D}|$ bem como quais vértices $v_n \in \mathcal{V}$ do sinal sobre grafo devem ser amostrados. Como mencionado em [3], aumentar o número de amostras em $\mathcal{D}$ sempre diminui o MSD em (4.3.12). No entanto, como também deseja-se reduzir a quantidade de nós a serem medidos, foi tomado $|\mathcal{D}| = 210$ por ser um compromisso razoável. O conjunto de amostragem $\mathcal{D}$ foi então obtido por meio do Algoritmo 4, com $M = |\mathcal{D}| = 210$. Então, neste ponto, obtém-se o sinal sobre grafo amostrado e de faixa limitada que será usado para testar as estratégias de interpolação linear instantânea e de reconstrução adaptativa.

A Fig. 6.4 ilustra os resultados da simulação de reconstrução, em que as leituras dos nós do grafo $\mathbf{x}_w[k]$ são representadas pelo cenário ruidoso em (4.3.11), com um sinal sobre grafo constante $\mathbf{x}[k] = \mathbf{x}^{(P)}$ até o instante de tempo $k = 2500$, quando o sinal de referência é escalado por um fator de 1,1. Os parâmetros dos algoritmos foram escolhidos como: estimação inicial $\mathbf{x}_{\mathrm{e}}[0] = \mathbf{0} \in \mathbb{R}^{299}$, matriz de covariância de ruído $\mathbf{C}_w = \sigma_w^2 \mathbf{I}$, com $\sigma_w^2 = 0{,}01$, fator de convergência $\mu = 0{,}25$ e fator de esquecimento $\beta = 0{,}95$. Como esperado, o Algoritmo 1 (interpolação *instantânea*) ignora os valores de entrada anteriores e, após um curto período de tempo, as estratégias adaptativas obtêm uma estimativa mais precisa do sinal original $\mathbf{x}[k]$. Entre as técnicas adaptativas, fica claro que o algoritmo RLS apresenta uma convergência bem mais rápida que o LMS.

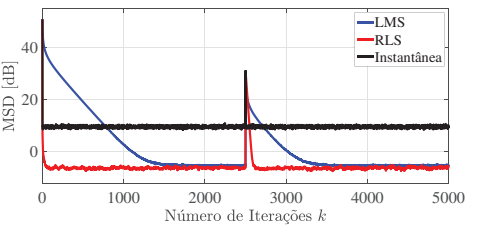


Figura 6.4: MSD para diferentes algoritmos de reconstrução usando um sinal constante $\mathbf{x}[k]$ com uma transição em degrau em $k = 2500$.

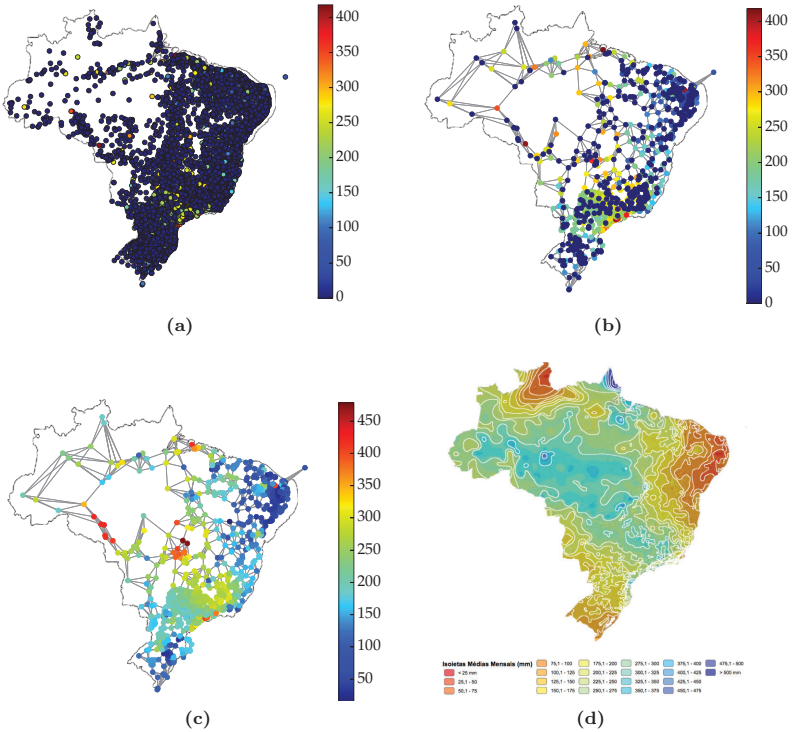


Figura 6.5: (a) Sinal sobre o grafo (sem arestas) dos 5570 municípios brasileiros, com amostras nulas exceto nos 609 vértices correspondendo ao banco de dados da Embrapa. (b) Sinal $\mathbf{x}_{\mathcal{D}}$ com 1000 amostras, 609 das quais provêm de base de dados pluviométricos e as demais são nulas. (c) Sinal $\mathbf{x}_{\mathrm{e}}$, estimando as amostras anteriormente nulas de modo a obter um sinal suave. (d) Isoieta média do mês de janeiro, correspondente ao período de 1977 a 2006. O contorno do mapa do Brasil sobre os grafos é inexato, estando aqui apenas para guiar o leitor, sem a pretensão de alinhá-lo com as coordenadas corretas de cada cidade. Foi desenhado por Felipe Micaroni Lalli e está disponível sob licença CC By-SA no endereço `https://commons.wikimedia.org/wiki/File:Contorno_do_mapa_do_Brasil.svg`.

Para ilustrar o caso em que não se conhece o suporte $\mathcal{F}$, mas ainda se tem a expectativa de que o sinal sobre o grafo seja suave, considere o seguinte problema proposto: dados os valores de pluviometria média do mês de janeiro de 609 cidades brasileiras, retirados do banco de dados da Embrapa[2], como se poderia estimar o valor deste índice em outras cidades do Brasil? A ideia é obter um sinal $\mathbf{x}_e$ com os índices de precipitação média em janeiro que seja suave, ao mesmo tempo em que preserve os dados nas 609 cidades conhecidos *a priori*. Isto se traduz no problema de otimização da função objetivo descrita em (4.4.14), em que o conjunto de amostragem $\mathcal{D}$ indica as cidades cujos dados estão no banco de dados.

Na Fig. 6.5a, as 5570 cidades brasileiras são representadas por vértices de um grafo sem arestas[3], das quais às 609 da base de dados da Embrapa são atribuídos os valores de pluviometria média. Para executar o algoritmo de otimização com mais rapidez, optou-se por reduzir o número de vértices para $N = 1000$, obtendo o sinal mostrado na Fig. 6.5b. O grafo foi criado ligando um vértice aos 5 vizinhos mais próximos, de modo a ter um grafo conexo ao mesmo tempo em que se evita uma matriz de adjacência demasiado cheia. Ponderou-se as arestas segundo (6.2.4) e, uma vez que se fez o grafo conexo e não foi utilizado um limite para julgar quais arestas seriam criadas, não há parâmetro que indique qual o melhor valor da variância θ a ser usado, razão pela qual se fez $2\theta^2 = 1$.

O sinal estimado deveria, idealmente, aproximar o mapa na Fig. 6.5d, com as isoietas (curvas de mesmo índice pluviométrico) médias do mês de janeiro, relativas ao período de 1977 a 2006[4]. É importante dirigir a atenção para a escala de cores do mapa, que é inversa ao padrão adotado ao longo deste capítulo: quanto mais quente a cor, menor o valor representado.

A restrição em (4.4.14) foi imposta dentro da declaração da função objetivo, a fim de reduzir seu número de variáveis, de modo que havia apenas $1000 - 609 = 391$ variáveis independentes. Dessa forma, trata-se de uma otimização sem restrições (*unconstrained optimization* [47]) com 391 variáveis, e para resolvê-la foi utilizado o algoritmo BFGS e a função `optimize. minimize()` do pacote Scipy, da linguagem Python. O resultado obtido após 2000 iterações do algoritmo de minimização é exibido na Fig. 6.5c e mostra que o algoritmo conseguiu aproximar de forma razoável o mapa de isoietas na Fig. 6.5d.

6.3 Compressão de *light fields*

Outra possível aplicação de GSP explora a propriedade de compactação de energia exercida pela GFT para a compressão de dados. Esta compactação permite maior eficiência da quantização de coeficientes de um sinal no domínio da frequência quando comparada com a quantização do sinal em seu domínio original. Neste exemplo, considera-se a compressão de imagens de *light fields* [34]. O imageamento de

[2] *Banco de Dados Climáticos do Brasil*, produzido pela Embrapa e pela ESALQ/USP, disponível em: `https://www.cnpm.embrapa.br/projetos/bdclima/index.html`

[3] As informações de latitude e longitude dos 5570 municípios brasileiros, utilizada para posicionar os vértices do grafo, foram retiradas da seguinte base: `https://github.com/kelvins/Municipios-Brasileiros`.

[4] Dados do Serviço Geológico do Brasil, também conhecido como CPRM, por conta de sua razão social Companhia de Pesquisa de Recursos Minerais, empresa pública vinculada ao Ministério de Minas e Energia. Este e outros atlas pluviométricos do Brasil estão disponíveis em: `http://www.cprm.gov.br/publique/Hidrologia/Mapas-e-Publicacoes/Atlas-Pluviometrico-do-Brasil-1351.html`

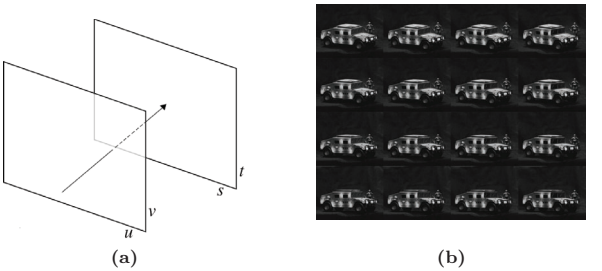


Figura 6.6: (a) Parametrização 4-D e (b) exemplo de resultados de captura de *light field*.

light field é uma tecnologia de captura de dados 4-D de uma cena, o que a torna uma tecnologia promissora para indústrias de entretenimento, como fotografia e cinema. Idealmente, esta tecnologia capturaria toda a informação sobre o campo de luz associado a uma cena sob a forma 5-D da função plenóptica $L(x, y, z, \theta, \phi)$, que indica a radiança (quantidade de luz) que passa por todos os pontos (x, y, z) do espaço em todas as direções (θ, ϕ). Esta informação permitiria manipulação de elementos de uma imagem após sua captura, como por exemplo ajuste do plano focal ou mudança da posição da vista da cena.

Na prática, a captura da função plenóptica não é realizável e uma parametrização é adotada para capturar informações da cena em 4-D. Esta parametrização é composta por um plano *uv* (plano da câmera) e um plano *st* (plano focal), como ilustrado na Fig. 6.6a. Implementações comuns para esta parametrização são realizadas das seguintes formas:

- Um arranjo de câmeras, localizado no plano *uv* com todas as cameras focadas na cena, localizada no plano *st*, criando uma versão discreta do plano *uv*;
- Uma única câmera se movendo por uma grade de posições no plano *uv*, capturando a cena do plano *st* a partir de cada posição na grade;
- Um arranjo de microlentes posicionado dentro de uma câmera digital convencional, de modo que cada microlente captura luz de uma direção, gerando diferentes vistas da cena.

Em todos os casos, o resultado de uma única captura de imagem em *light field* é um conjunto de vistas como ilustrado na Fig. 6.6b. Muitas vezes, a quantidade de imagens capturadas de uma única cena está na ordem de centenas ou milhares, o que gera uma enorme quantidade de dados. Algumas abordagens consideram o uso do *High-effiency video coding* (HEVC), que usa a transformada do cosseno discreto (DCT, do inglês *discrete-cosine transform*), para comprimir dados de *light field*. O HEVC processa *resíduos* de blocos de pixels que compõem uma imagem inteira. O resíduo é resultado do processo de *predição* e pode ser entendido como a diferença entre o bloco de pixels no *frame* que está sendo codificado e blocos de pixels em *frames* anteriores. Como tipicamente blocos em *frames* consecutivos são similares, os resíduos possuem menor entropia do que os blocos originais, facilitando a quantização. No caso de *light field*, diferentes vistas da cena fazem o papel dos

frames. O papel da transformada é mapear dados de resíduos para um domínio onde a quantização se torna mais eficiente. Como demonstrado em [27,67], a GFT é capaz de concentrar energia em uma quantidade menor de coeficientes quando comparada com a DCT. Assim, se a mesma quantidade de coeficientes da transformada for mantida para as duas transformadas e a transformada inversa for aplicada, a GFT alcança *distorção* menor para o bloco reconstruído do que aquela alcançada pela DCT.

Como já demonstrado, a GFT está diretamente ligada à matriz de adjacências. Em alguns casos, a inferência de $\mathbf{A}$ depende dos dados e, portanto, não existe uma matriz de GFT fixa. Com isso, um sistema de compressão de dados baseado na GFT precisa conhecer $\mathbf{A}$ ou $\mathbf{V}$ (ou $\mathbf{U}$ no caso de GSP_L) no codificador e no decodificador, o que faz com que a quantidade de dados a ser transmitida cresça consideravelmente além dos elementos comprimidos. Nesta aplicação, a inferência e aplicação de $\mathbf{A}$ são exploradas de forma a minimizar seu impacto.

Considera-se GSP_A e grafos do tipo apresentado na Fig. 1.3b para representar blocos de resíduos de 32×32 pixels, seguindo o modelo *nearest-neighbor* (NN) [58]. Com esta estrutura, cada pixel está associado a um vértice e se liga a, no máximo, quatro vizinhos mais próximos (três para pixels na borda e dois para pixels no canto), oferecendo uma matriz de adjacência esparsa. Além disso, considera-se um único peso para arestas horizontais ligando todos pares de pixels dadas duas colunas do bloco. O mesmo é considerado para arestas verticais ligando pares de pixels dadas duas linhas. Por exemplo, todas as arestas ligando pixels da coluna de s_0 com a coluna de s_1 na Fig. 1.3b possuem o mesmo peso. Outro peso é dado a todas as arestas ligando as colunas de s_1 e s_2 e outro às arestas ligando as linhas dos pixels s_0 e s_N. Para um bloco de $M_1 \times M_2$ pixels, estas considerações resultam em $\mathbf{A}$ com estrutura fixa e $(M_1 - 1) + (M_2 - 1)$ coeficientes não nulos. É importante notar que, ao se transmitir a matriz $\mathbf{A}$ ao invés da matriz de transformada, adiciona-se complexidade ao decodificador, que precisa calcular seus autovetores.

Considerando que blocos da mesma posição t_0 (Fig. 6.7) em resíduos diferentes são similares, é possível usar uma única matriz $\mathbf{A}$ para representar blocos em diferentes resíduos, mitigando ainda mais seu impacto. A partir do modelo NN, os valores das arestas são calculados, para um único bloco, de modo a reduzir a distorção ℓ_2 introduzida pelo operador de atraso, isto é, $\|\mathbf{A}\mathbf{s} - \mathbf{s}\|_2$, onde $\mathbf{s}$ é o vetor contendo os valores dos pixels do bloco. Este raciocínio pode ser estendido para calcular uma matriz representativa de múltiplos blocos ao se minimizar $\sum_{k=k_1}^{k_2} \|\mathbf{A}\mathbf{s}_k - \mathbf{s}_k\|_2$, considerando blocos k_1 até k_2 em um grupo de $K - 1$ blocos (por exemplo, um bloco para cada um dos $K-1$ resíduos de uma linha do *light field* com $K \times K$ vistas). Esta abordagem faz com que a eficiência da transformada seja mais uniforme quando a mesma matriz é aplicada para vários blocos. Se um único bloco for utilizado no cálculo de $\mathbf{A}$, a transformada será mais eficiente neste bloco [18].

Esta metodologia foi aplicada a uma base de dados composta por três *light fields* reais, capturados por uma câmera móvel, e a quatro *light fields* gerados em computador. Considerando que apenas uma matriz $\mathbf{A}$ é calculada para blocos numa mesma posição em cada linha do *light field*, foi analisada a eficiência da remoção de coeficientes e a distorção em erro quadrático médio (MSE, do inglês *mean squared error*), comparando com a DCT. A simulação considera 100 coeficientes mantidos da DCT (de um total de $32 \times 32 = 1024$) e remove coeficientes da GFT enquanto o MSE resultante é menor que o alcançado pela DCT. Para cada posição de bloco, a matriz $\mathbf{A}$ foi calculada de duas formas: em função apenas do bloco pertencente ao

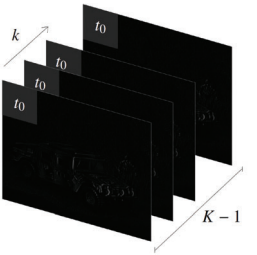


Figura 6.7: Representação de blocos em uma dada posição t_0 em um grupo de $K-1$ resíduos.

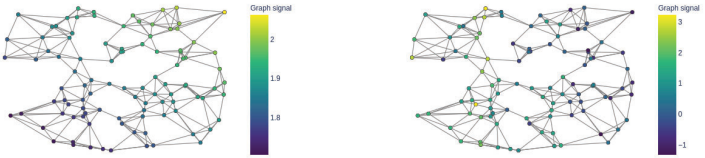
resíduo central da linha, minimizando $\|\mathbf{As}-\mathbf{s}\|_2$; e em função de todos os resíduos da linha do *light field*, minimizando $\sum_{k=k_1}^{k_2} \|\mathbf{As}_k - \mathbf{s}_k\|_2$.

Gerando cada matriz $\mathbf{A}$ (associada a cada posição de bloco) a partir de blocos pertencentes apenas ao resíduo central, observa-se que a GFT é capaz de fornecer o mesmo MSE com 6% menos coeficientes que a DCT, na média dos 7 *light fields*, considerando também todos os coeficientes necessários para a transmissão da matriz $\mathbf{A}$. Este valor chega a 21,22% dependendo do *light field*. Como apenas blocos de um resíduo foram utilizados, a eficiência em outros resíduos (não considerados no cálculo de $\mathbf{A}$) é menor, resultando em um desvio padrão médio de 5,3 coeficientes necessários por resíduo. Quando todos os resíduos são considerados na inferência da matriz de adjacências, a redução em número de coeficientes melhora para 8,4% na média dos *light fields* e o desvio padrão médio na quantidade de coeficientes por resíduo cai para 1,4. Embora o sistema de compressão adotado seja uma versão simplificada (com menos etapas e realizando eliminação de coeficientes ao invés de quantização) daqueles empregados no estado da arte, estes resultados mostram que a GFT é capaz de oferecer desempenho competitivo para compressão em relação a transformadas tradicionais.

6.4 Outras áreas de aplicação

Os exemplos anteriores ilustram o uso de GSP em processamento de redes de sensores, um contexto que naturalmente surge da definição do domínio do sinal como sendo um grafo. No entanto, outros cenários menos evidentes permitem a aplicação da teoria.

Primeiramente, o leitor deve lembrar do comentário de que uma imagem pode ser modelada como um grafo não-direcionado em forma de grade (Fig. 1.3b) [58], segundo o modelo NN, considerado na seção anterior. Com ajuste apropriado dos pesos das arestas, Sandryhaila e Moura mostraram que a GFT leva a uma compressão de imagens bastante eficiente, com erros de aproximação por truncamento do espectro menores do que aqueles utilizando-se a DFT, DCT ou DWT [62]. De fato, a análise espectral de imagens via DCT, conhecida por sua propriedade de redução de dimensionalidade, está diretamente relacionada à GFT segundo GSP_L, pois o Laplaciano do grafo da Fig. 1.3b, com pesos unitários, é diagonalizado pela matriz da DCT [20]. No contexto de imagens de *light field*, além do uso de GSP em compressão, a teoria também tem fornecido ferramentas para codificação [74] e esquemas de super-resolução [56]. Para uma revisão extensa sobre aplicação de



(a) **(b)**

Figura 6.8: Dois sinais sobre o mesmo grafo não-direcionado.

GSP em compressão, restauração, filtragem e segmentação de imagens, os autores recomendam [8].

Outra importante área de estudo para GSP envolve redes biológicas (*biological networks*), i.e. o tratamento de sistemas em rede – humanos ou da natureza – para inferência ou processamento de dados. Por exemplo, com respeito às redes genéticas regulatórias, o uso de métodos baseados em grafos levou à melhoria dos três esquemas do estado-da-arte em inferência de redes [51, 52]. Nguyen *et al.* representaram o corpo humano como uma grade dinâmica tridimensional e aplicaram bancos de filtros de *wavelets* sobre grafos para comprimir os dados de posição e cor, superando os métodos usuais [46]. O processamento de sinais cerebrais, no entanto, parece ser a mais intrigante e prolífica dentre estas aplicações, surgindo pela atribuição de sinais como os de imagem por ressonância magnética funcional (fMRI, do inglês *functional magnetic resonance imaging*) a grafos definidos por redes cerebrais funcionais [24, 33]. Tais trabalhos têm demonstrado, por exemplo, correlação entre as componentes de baixa e alta frequências destes sinais e o processo de aprendizado de uma tarefa motora [28].

Como mencionado na subseção 4.4, a regularização de um sinal sobre grafos com amostras discretas foi utilizada como método de classificação de dados em [64]. Este é um exemplo dos muitos usos de GSP em aprendizado de máquina, dos quais se pode também citar o aprendizado semi-supervisionado pelo uso de filtros adaptativos sobre grafos [4], sistemas de recomendação consistindo em filtragem colaborativa e baseada em conteúdo, utilizando regularização com variação total em grafos [1], e detecção de comunidades em redes via projeto de *wavelets* em grafos, o que permite mineração multiescala de comunidades [75].

6.5 Problemas

Problema 6.1. Considere os dois sinais distintos sobre o mesmo grafo, na Fig. 6.8. Se replicarmos o gráfico da Fig. 6.3 em ambos os sinais, calculando o ARE (*average reconstruction error*) para o sinal reconstruído com cada vez mais componentes de frequência (tomadas em ordem decrescente de magnitude), qual dos dois sinais terá a curva que converge mais rápido para zero? Justifique.

Problema 6.2. Considere a estimação de dados ausentes ilustrada na Fig. 6.5. Como vimos, é um cenário de tentativa de reconstrução do sinal quando ele é assumido suave, mas não se conhece seu suporte na frequência. A estratégia usada foi encontrar o sinal *o mais suave possível* que satisfaz na igualdade as amostras conhecidas – é exatamente o significado de (4.4.14). Com base no que foi apresentado até aqui, em especial sobre $\mathrm{GSP_A}$, apresente ao menos uma razão pela qual *não seria uma boa ideia* utilizar deslocamentos sucessivos do sinal, multiplicando pela matriz de adjacência, para encontrar as novas amostras sobre os vértices antes deconhecidos.

Problema 6.3. Construa uma imagem de dimensões 256×256 usando a seguinte expressão:

$$\mathrm{Im}(k,l) = 0{,}5 + 0{,}5 \cos\left(\frac{2\pi}{d}(k\,\cos\theta + l\,\mathrm{sen}\,\theta)\right), \tag{6.5.5}$$

com $d = 2$ e $\theta = \pi/4$. Construa um grafo de 256×256 vértices como mostrado na Fig. 1.3b, de modo que cada vértice corresponda a um pixel da imagem gerada. Considere o sinal do grafo como o empilhamento vertical de todas as colunas da imagem.

(a) Obtenha a representação em frequência de grafo da imagem usando a GFT definida pra $\mathrm{GSP_A}$.

(b) Obtenha a representação em frequência de grafo da imagem usando a GFT definida pra $\mathrm{GSP_L}$ e compare com os resultados do item anterior.

(c) Carregue uma imagem natural (por exemplo, um retrato de rosto ou uma paisagem) com dimensões 256×256 pixels e em escala de cinza, com seus pixels no intervalo $[0, 1]$. Repita os passos (a) e (b) e compare com os resultados anteriores.

(d) Aumentando o valor de d para valores inteiros menores ou iguais a 256, repita os passos (a) e (b) e observe os resultados.

(e) Para imagems naturais, espera-se que seus sinais sejam de baixa ou alta frequencia no grafo? Comente o motivo.

Capítulo 7

Implementação em *Software* e Oportunidades de Estudo

O rápido desenvolvimento da teoria de processamento de sinais sobre grafos, na última década, trouxe consigo o interesse pela criação de ferramentas computacionais que pudessem facilmente pô-la em prática. Embora os cálculos em GSP envolvam basicamente álgebra matricial, o que já é satisfatoriamente atendido por diversos *softwares* de computação científica, a implementação de ferramentas específicas justifica-se pela conveniência de se ter tipos de dados e operadores definidos diretamente para GSP, como classes particulares de grafos, transformadas e filtros, por exemplo.

7.1 As ferramentas típicas da comunidade

Certamente a primeira ferramenta com que o leitor terá contato, ao buscar seus primeiros códigos em GSP, será a GSPBOX[1]. Trata-se de uma *toolbox* em MATLAB criada pelo *Signal Processing Laboratory LTS2* da *Ecole Polytechnique Fédérale de Lausanne*, dentre cujos autores figuram nomes importantes para esta publicação, como David Shuman e Pierre Vandergheynst.

O GSPBOX foi construído sob a perspectiva de GSP_L e, portanto, representa os sinais sobre grafos segundo o autoespaço da matriz Laplaciana. O pacote traz uma grande quantidade de operadores, classes específicas de grafos e funções, todos com descrições disponíveis em sua documentação. Aqueles que não dispõem do MATLAB, podem executar os comandos básicos do GSPBOX no equivalente Octave, de código aberto. No entanto, algumas funções, como por exemplo a `gsp_nn_graph`, que cria um grafo conectado a partir de uma *pointcloud*, requerem outras *toolboxes* específicas, como a Statistics e a Machine Learning, o que pode ser um incômodo para o programador de Octave. A Fig. 7.1a traz um grafo toroidal e um dos autovetores de sua matriz Laplaciana, gerada via GSPBOX.

O segundo pacote para MATLAB que convém mencionar é o GraSP[2] [23], criado

[1] Sua documentação, *links* para *download* e demais informações estão disponíveis no endereço: `https://epfl-lts2.github.io/gspbox-html/` [50]. O código é distribuído gratuitamente, sob licença GNU GPLv3.

[2] O pacote GraSP, com sua versão mais recente de julho de 2016 (consultada em setembro

por Benjamin Girault para implementar algumas ferramentas da sua teoria de sinais estacionários sobre grafos (que não cobrimos neste capítulo), como a sua versão da operação de translação no domínio dos vértices. A motivação para a sua definição é criar um deslocamento que preserve a norma ℓ_2 do sinal, como ocorre na teoria clássica com sinais de tempo discreto. O resultado é um deslocamento que não preserva a norma ℓ_1 do sinal, e tem interpretação menos direta do que a translação pela matriz de adjacência de GSP_A. Veja-se, por exemplo, a Fig. 7.1b, em que a translação de Girault foi aplicada três vezes ao impulso unitário localizado no vértice v_{10}, e o sinal resultante não espalhou-se tanto a partir do vértice de origem, como ocorreria com o deslocamento usual pela matriz $\mathbf{A}$.

Embora a linguagem do programa MATLAB seja largamente utilizada pela comunidade de processamento de sinais e computação científica, a criação de uma API em Python certamente encontraria um número bem maior de usuários. Afinal de contas, em setembro de 2020, Python figurou como a linguagem de programação mais popular do mundo, pelo índice PYPL[3], além de ser uma das mais buscadas para ciência de dados, aprendizagem de máquina e áreas afins. Por essa razão, Nathanaël Perraudin, membro da equipe do GSPBOX, uniu-se a outros pesquisadores para implementar as funcionalidades desta *toolbox* como uma biblioteca em Python, a PyGSP [13]. Em dezembro de 2020, o repositório do pacote no Github possuía 1652 *commits* (atualizações), a mais recente das quais foi em 24 de agosto.

7.2 Implementando GSP em Python

O leitor é convidado a conferir a implementação de várias funções e métodos de GSP em Python, que criamos e disponibilizamos em um reporitório no Github intitulado GSPy[4]. A implementação não está otimizada para performance, mas, antes, para fácil leitura e para permitir ao leitor identificar oa elementos de GSP permeando o código.

Assim como ocorre no GSPBOX, a exibição de um grafo com GSPy requer apenas sua matriz de adjacência ponderada $\mathbf{A}$, de dimensões $N \times N$, e um *array* $N \times 2$ de números reais, para armazenar as coordenadas dos vértices do grafo. A criação de uma imagem contendo o grafo requer a mistura de um gráfico de dispersão (o *scatter plot* dos pacotes gráficos de computação científica), para representar os vértices, e gráficos de linha para desenhar as arestas. É importante notar que, em geral, gráficos de dispersão com *muitos* pontos geram objetos gráficos muito "pesados", de renderização lenta e ocupam grande espaço na memória[5].

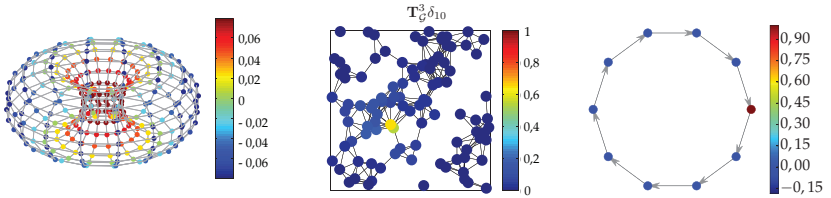
---

de 2020), com seus arquivos e documentação, está disponível em `https://gforge.inria.fr/projects/grasp/`.

[3]Índice baseado no número de buscas por tutoriais da linguagem de programação no Google. Disponível em `http://pypl.github.io/PYPL.html`.

[4]O código está disponível no endereço `https://github.com/gboaviagem/GSPy`.

[5]O problema de visualizar grafos massivos é bem conhecido e e vai além do mero custo computacional: como dispor os vértices — que, em geral, não têm informação de geolocalização associada — no espaço 2D, de modo a que a imagem *tenha sentido* para o analista? Há pacotes em Python próprios para a manipulação e visualização de grafos, como NetworkX, python-igraph e graph-tool, e o leitor pode encontrar um bom comparativo entre eles (e outros) num artigo de Timothy Lin [35], de maio de 2020. Todos oferecem métodos de visualização, com algoritmos diversos para cálculo da disposição dos vértices, mas quanto maior o grafo mais tempo os algoritmos tomam e mais raros são os cenários em que a visualização é útil. Se a intenção é apenas visualizar grafos massivos, com razoável flexibilidade quanto ao layout, há programas *standalone* que podem ser úteis, como Gephi e Cytoscape. A transferência de um grafo de um software para outro pode



(a) (b) (c)

Figura 7.1: (a) Autovetor sobre um grafo toroidal, gerado com o GSPBOX. (b) Operador de translação de Girault aplicado três vezes sobre um impulso num grafo de sensores, originalmente sobre o vértice amarelo, utilizando o GraSP. (c) Impulso unitário sobre o vértice v_0 de um grafo direcionado em anel, gerado pelo GSPy.

Uma operação fundamental que surge em aplicações de GSP é a GFT, que é rapidamente implementada utilizando as funções de álgebra linear do Numpy. O código 7.1 mostra a aplicação de um filtro passa-baixas a um sinal `s`, definido sobre um grafo de matriz de adjacência `A`. A função `stem` e as constantes `fsize` (tamanho de fonte) e `msize` (tamanho do marcador) foram definidas em outro trecho do *script*.

```
import numpy as np
[eigvals,V] = np.linalg.eig(A) # eigendecomposition of the adjacency matrix
Vinv = np.linalg.inv(V)
ss = np.dot(Vinv,s) # GFT according to GSP_A

hh_LPF = 1.0 -  1.0/(1 + np.exp(2*(eigvals - np.mean(eigvals))/np.mean(np.abs(eigvals)))) # filter spectrum

plt.figure()
stem(eigvals,ss,fsize,msize,'b',':',r'\widehat{\mathbf{s}}$') # input signal spectrum
stem(eigvals,ss*hh_LPF,fsize,msize-2,'r',':',r'$h(\Lambda) \widehat{\mathbf{s}}$') # output signal spectrum
```

Código 7.1: Exemplo de filtragem passa-baixas num certo sinal `s`.

7.2.1 Métodos básicos de PyGSP

Vários métodos e algoritmos de GSP, tanto de uso genérico como específico, encontram-se implementados no GSPBOX e PyGSP. Esta subseção pretende comentar sobre as possibilidades apresentadas pela versão em Python.

A classe mais importante do PyGSP é a `pygsp.graphs.Graph()`, que instancia o objeto responsável por representar um grafo, a partir de sua matriz de adjacência. É importante mencionar que a biblioteca usa a letra **W** para representar a matriz de adjacência. Um exemplo de como criar um grafo em anel, não-direcionado, com três vértices, é mostrado no código 7.2. Neste exemplo, o código é exibido tal como seria num *shell* IPython (diretamente no interpretador, e não num script isolado).

ser feita utilizando formatos de arquivos próprios para exportar dados estruturados como grafos, como GraphML e GEXF, ou formatos para serialização de objetos, como Pickle e YAML.

As linhas que começam com `In` ou `...:` mostram comandos de entrada, enquanto as demais exibem as saídas.

```
In [1]: from pygsp import graphs
...: import numpy as np
...:
...: A = np.array([[0, 0, 1],
...:                [1, 0, 0],
...:                [0, 1, 0]])
...:
...: G = graphs.Graph(W=A)
...: print(G.W)
(0, 2)  1
(1, 0)  1
(2, 1)  1
In [2]: print("Qtd de vertices:", G.N)
...: print("Qtd de arestas:", G.Ne)
Qtd de arestas: 3
Qtd de vertices: 3
```

Código 7.2: Criando um grafo em PyGSP.

Perceba que a matriz de adjacência é armazenada como uma lista de arestas (pares ordenados de vértices) e pesos. Internamente, o atributo faz uso de uma representação de matrizes esparsas do Scipy (pacote em Python para computação científica). Trata-se de uma forma mais eficiente de guardar as matrizes de grafos comuns, obtidos de fenômenos naturais, que são geralmente esparsas.

Com o exemplo anterior, o leitor pode observar que o objeto da classe `Graph()` guarda nos atributos `W`, `N` e `Ne`, respectivamente, a matriz de adjacência ponderada, o número de vértices e o número de arestas. Há ainda diversos outros métodos e atributos desta classe para uso básico em GSP, tais como:

Matriz Laplaciana: o método `compute_laplacian()` calcula a matriz Laplaciana e a armazena no atributo `L`. O tipo de Laplaciana a ser calculado pode ser *combinatorial* ($\mathbf{L} = \mathbf{D} - \mathbf{A}$) ou *normalizada* ($\mathbf{L} = \mathbf{I} - \mathbf{D}^{-1/2}\mathbf{A}\mathbf{D}^{1/2}$), escolhida através do parâmetro `lap_type`. Por padrão, calcula-se a Laplaciana combinatorial. Continuando o exemplo anterior, o código seria:

```
In [3]: G.compute_laplacian()
...: print(G.L)
(0, 0)  1.0
(1, 0)  -0.5
(2, 0)  -0.5
(0, 1)  -0.5
(1, 1)  1.0
(2, 1)  -0.5
(0, 2)  -0.5
(1, 2)  -0.5
(2, 2)  1.0
In [4]: print(G.L.todense())  # Converte da forma esparsa para a
    forma densa
[[ 1.   -0.5 -0.5]
 [-0.5  1.   -0.5]
 [-0.5 -0.5  1. ]]
```

GFT: dado que a matriz Laplaciana já foi calculada, o método `compute_fourier_basis()` a diagonaliza e armazena a base ortogonal de autovetores (nas colunas) e os respectivos autovalores nos atributos `U` e `e`. O conjugado transposto da matriz `U` é o operador de Fourier em $\mathrm{GSP}_{\mathrm{L}}$, $\mathbf{F} = \mathbf{U}^*$.

```
In [5]: G.compute_fourier_basis()
...: print(G.U)
[[ 0.57735027 -0.50766227 -0.63948861]
 [ 0.57735027  0.80764452 -0.11990412]
 [ 0.57735027 -0.29998225  0.75939273]]
In [6]: print(G.e)
[0.  1.5 1.5]
```

As GFTs direta e inversa são computadas pelos métodos `gft()` e `igft()`.

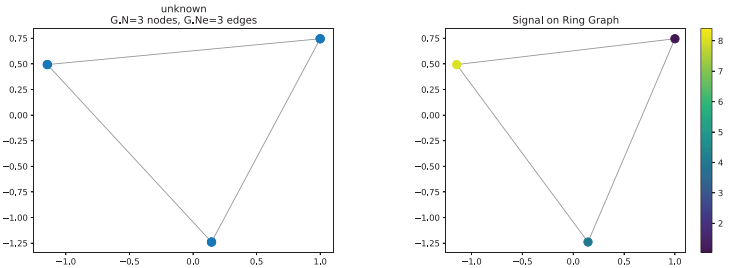
```
In [7]: s = np.array([1, 4, 8])  # Sinal sobre o grafo
...: s_hat = G.gft(s)  # GFT
...: print(s_hat)
[7.5055535 , 0.32305781, 4.95603676]
In [8]: print(np.conjugate(G.U).T @ s)  # GFT pelo produto
    matricial
[7.5055535 , 0.32305781, 4.95603676]
In [9]: print(np.sum(G.igft(s_hat) - s))  # Erro de reconstrucao
-4.884981308350689e-15
```

Visualização Os métodos de visualização do PyGSP requerem que as coordenadas dos vértices sejam informadas. Para isso, o usuário pode passar um *array* 2D ou 3D para o atributo `coords`, ao instanciar o objeto `Graph()`, ou utilizar o método `set_coordinates()`. Este método aceita tanto o *array* de coordenadas, como uma *string* indicando o algoritmo a ser usado para calculá-las automaticamente. As opções são: *community2D*, *random2D*, *random3D*, *ring2D*, *line1D* ou *spring* (que usa o algoritmo Fruchterman-Reingold de disposição dos vértices). Caso nenhum parâmetro seja passado, o padrão é calcular as posições dos vértices com o método *spring*.

Uma vez que as coordenadas estão armazenadas no objeto, o usuário pode gerar uma imagem do grafo com o método `plot()`, ou incluir a representação do sinal `s` sobre o grafo com o método `plot_signal(s)`. Vale mencionar que o usuário que estiver mais confortável com o paradigma procedural de programação, do que com a orientação a objeto, pode obter o mesmo resultado importando esses métodos e usando-os como funções. Há vários parâmetros opcionais, para configurar a imagem gerada, e podem ser acessadas pelo `help()` no *shell* IPython ou Jupyter Notebook, ou ainda investigando o código diretamente no Github.

```
In [10]: coords = np.array([[1.0, 0.75], # vertice 0
                            [0.2, -1.25], # vertice 1
                            [-1.2, 0.5]]) # vertice 2
...: G.set_coordinates(coords)
...: G.plot(save_as="ring_graph.pdf")
...: G.plot_signal(s, plot_name="Signal on Ring Graph", save_as="
    signal_ring_graph_pygsp.pdf")
In [11]: from pygsp.plotting import plot, plot_signal
...: # Mesmo resultado, de forma procedural:
...: plot(G, save_as="ring_graph.pdf")
...: plot_signal(G, s, plot_name="Signal on Ring Graph", save_as=
    "signal_ring_graph_pygsp.pdf")
In [12]: help(plot_signal)  # Exibe a documentacao
```

As imagens geradas com o código acima são exibidas na Fig. 7.2. O leitor é convidado a explorar a documentação do PyGSP e conhecer todo o potencial do pacote, que excede em muito a breve introdução aqui apresentada.



(a) **(b)**

Figura 7.2: (a) Visualização do (a) grafo e do (b) sinal sobre o grafo, utilizando os métodos de PyGSP com diferentes parâmetros.

7.2.2 Caso prático: encontrando um filtro passa-baixas LSI via mínimos quadrados

Um exemplo interessante que mistura matemática, processamento de sinais e programação é aquele feito por Moura *et al.* [59], ilustrando como se pode encontrar um filtro passa-baixas LSI simples através de uma minimização por mínimos quadrados.

Para determinar um filtro invariante ao deslocamento, basta encontrar os coeficientes de seu polinômio no operador de deslocamento,

$$h(\mathbf{S}) = \sum_{\ell=0}^{L-1} h_\ell \mathbf{S}^\ell. \tag{7.2.1}$$

Trata-se de (2.1.4), em que a matriz de adjacência fazia o papel de operador de deslocamento.

Na seção 6.1, foi feita uma filtragem passa-baixas definindo o filtro ideal diretamente no domínio da frequência. Ou seja, definiu-se o filtro pelos valores de $h(\lambda_i)$ (resposta do filtro em cada autovalor do operador de deslocamento) e não pelos valores de h_ℓ (coeficientes do filtro). No presente caso, podemos usar do fato de que conhecemos λ_i e sabemos o perfil da resposta $h(\lambda_i)$ que queremos impor (i.e. um filtro passa-baixas ideal), para encontrar h_ℓ.

Ao descrever a resposta em frequência do filtro para cada autovalor γ_i, o resultado será um sistema de equações lineares,

$$\begin{aligned} h(\gamma_0) &= \alpha_0, \\ h(\gamma_1) &= \alpha_1, \\ &\vdots \\ h(\gamma_{N-1}) &= \alpha_{N-1}, \end{aligned} \tag{7.2.2}$$

ou, sabendo que $h(\cdot)$ é um polinômio de grau L,

$$\begin{aligned} h_0 + h_1\gamma_0 + \cdots + h_L\gamma_0^L &= \alpha_0, \\ h_0 + h_1\gamma_1 + \cdots + h_L\gamma_1^L &= \alpha_1, \\ &\vdots \\ h_0 + h_1\gamma_{N-1} + \cdots + h_L\gamma_{N-1}^L &= \alpha_{N-1}, \end{aligned} \tag{7.2.3}$$

o que pode ser escrito matricialmente utilizando uma matriz de Vandermonde com os autovalores,

$$\begin{bmatrix} 1 & \gamma_0 & \gamma_0^2 & \cdots & \gamma_0^L \\ 1 & \gamma_1 & \gamma_1^2 & \cdots & \gamma_1^L \\ & \vdots & & \vdots & \\ 1 & \gamma_{N-1} & \gamma_{N-1}^2 & \cdots & \gamma_{N-1}^L \end{bmatrix} \begin{bmatrix} h_0 \\ h_1 \\ \vdots \\ h_L \end{bmatrix} = \begin{bmatrix} \alpha_0 \\ \alpha_1 \\ \vdots \\ \alpha_{N-1} \end{bmatrix}. \tag{7.2.4}$$

É importante ressaltar novamente que *conhecemos* os valores desejados para α_i, $i = 0, \ldots, N-1$. Por exemplo, se o filtro passa-baixas buscado tiver frequência de corte no autovalor $\gamma_{i_{\text{cut}}}$, seria satisfatório ter

$$\begin{cases} \alpha_j = 1 & \text{para } j \leq i_{\text{cut}} \\ \alpha_j = 0 & \text{c.c.} \end{cases} \tag{7.2.5}$$

Uma vez que, em geral, tem-se $N \geq L+1$, o sistema de equações é *sobredeterminado* e não possui solução. Uma estratégia possível é encontrar os coeficientes h_ℓ, $\ell = 0, \ldots, L-1$, que minimizam, no sentido dos mínimos quadrados, o desvio da resposta ideal do filtro. Ou seja, busca-se solucionar a otimização

$$\min_{\{h_\ell\}_{0,\ldots,L-1}} \left(\sum_{n=0}^{N-1} h(\gamma_n) - \alpha_n \right)^2. \tag{7.2.6}$$

Exemplo 7.1. Tomemos o já conhecido grafo da malha viária de Minnesota, mostrado na Fig. 6.1. O filtro passa-baixas que foi usado no exemplo da seção 6.1 foi definido diretamente no domínio da frequência; nada se sabe sobre sua forma polinomial no operador de deslocamento. Encontremos o filtro passa-baixas LSI de grau 12 que mais se aproxima do ideal, no sentido dos mínimos quadrados. Para tanto, utilizemos a matriz Laplaciana como operador de deslocamento, no contexto de GSP_L.

Tomando os dados do grafo de Minnesota diretamente da memória do PyGSP, os comandos para criação e diagonalização do grafo ocupam poucas linhas (código 7.3 – alguns detalhes de formatação e *layout* do gráfico foram omitidos, por concisão). Os autovalores da Laplaciana são exibidos na Figura 7.3.

```
1 import matplotlib.pyplot as plt
2 from pygsp import graphs
3
4 G = graphs.Minnesota()
5 G.compute_fourier_basis()
6
```

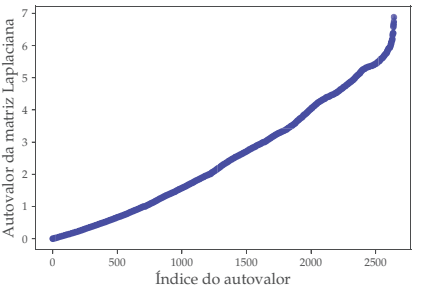


Figura 7.3: Autovalores da matriz Laplaciana do grafo de Minnesota.

```
plt.plot(G.e, marker='.',
         color=np.array([0.25, 0.25, 0.75, 0.3]),
         markersize=10, alpha=0.7)
```

Código 7.3: Criação do grafo de Minnesota e diagonalização do operador de deslocamento (matriz Laplaciana).

De posse dos autovalores e do grau do polinômio que desejamos obter, resta definir o perfil ideal da resposta desejada do filtro. Para seguir o exemplo da seção 6.1, façamos a resposta ideal ser 1 para autovalores $0 \leq \gamma \leq 0{,}5$ e zero fora desse intervalo. Ou seja, será criado um vetor $\alpha = [\alpha_0, \ldots, \alpha_{N-1}]$, com os valores que aparecem no sistema de equações em (7.2.2), em que $\alpha_i = 1$ se, e somente se, o autovalor γ_i satisfaz $0 \leq \gamma_i \leq 1$.

Para a otimização via mínimos quadrados, será utilizada a função `leastsq`, do módulo `scipy.optimize`. Para tanto, é preciso definir a função custo que será minimizada: trata-se do erro entre a resposta do filtro ideal e aquela do filtro LSI de grau 12, i.e. é a diferença entre os lados esquerdo e direito de (7.2.4)[6]. Uma vez que a diferença será *minimizada* é preciso tomar seu valor absoluto. O código 7.4 traz a definição desta função custo e do vetor α com a resposta ideal do filtro. O leitor é convidado a testar cada parte do código, para compreender os detalhes de cada comando, se assim desejar.

```
import numpy as np

deg = 12  # Order of the low-pass filter
eigvals = np.sort(G.e)
eig_vander = np.fliplr(np.vander(eigvals, N=deg+1))

def cost(h_coeff, x_vander, y):
    """
    Cost function for a set of linear equations.

    Parameters
    ----------
    h_coeff : np.ndarray, shape=(L+1,)
```

[6]Não é necessário definir a função custo como o erro médio quadrático, basta que ela retorne o vetor de erros por amostra. Internamente, dada a função custo `func`, a `leastsq` já tenta resolver o problema `arg min(sum(func(p)**2,axis=0)`, dado o vetor de parâmetros livres `p`.

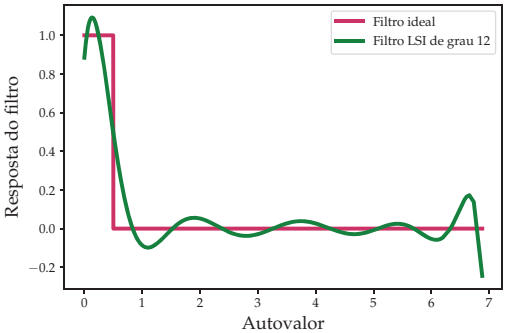


Figura 7.4: Resposta ideal do filtro passa-baixas (i.e. vetor α), em vermelho. Em verde, a resposta do filtro LSI de grau 12.

```
        Array of polynomial coefficients.
    x_vander : np.ndarray, shape=(M, L+1)
        Vandermonde matrix with the abscissa.
    y : np.ndarray, shape=(M,)
        Array of expected polynomial outputs.
    """
    assert isinstance(x_vander, np.ndarray), "x must be a np.ndarray."
    return np.abs((x_vander @ h_coeff) - y)

alpha = np.zeros(len(eigvals))
idx = eigvals <= 0.5
alpha[idx] = 1
```

Código 7.4: Definindo a função custo e a resposta ideal do filtro passa-baixas.

A execução do otimizador é feita com as linhas do código 7.5, em que os coeficientes que mais aproximam o filtro LSI daquele ideal são armazenados no vetor `h_coeff_`. As respostas do filtro obtido e do ideal são mostradas na Fig. 7.4.

```
from numpy.polynomial import Polynomial
from scipy.optimize import leastsq

h_coeff_, cov_x, infodict, msg, ier = leastsq(
    cost, x0=np.zeros(deg+1), args=(eig_vander, alpha),
    full_output=True)

# Checando se o otimizador convergiu
print(f"Optimization output message: {msg}")

H = Polynomial(h_coeff_)
resposta_LSI = H(eigvals)  # resposta do filtro LSI
```

Código 7.5: Calculando os coeficientes do filtro LSI.

□

7.3 Algumas oportunidades de estudo

Como comentado, a abordagem de GSP_L foi desenvolvida considerando grafos não-direcionados, com pesos reais não-negativos. Nesses casos, $\mathbf{L}$ é diagonalizável e positiva semi-definida, o que gera propriedades convenientes para a aplicação dos conceitos teóricos, como ter frequências sempre reais e uma componente DC associada à frequência nula. O arcabouço de GSP_A apresenta a vantagem de considerar grafos gerais, desde que sem laços e múltiplas arestas, mas também traz desvantagens. Como comentado em [66], o uso da matriz de adjacência como bloco elementar e a aplicação a grafos direcionados carrega uma série de empecilhos quando a matriz $\mathbf{A} = \mathbf{V}_G \mathbf{J} \mathbf{V}_G^{-1}$ é não-diagonalizável. Por exemplo, há os problemas de instabilidade numérica, que ocorrem na decomposição de Jordan para matrizes de ordem prática, e o fato de que $\mathbf{V}_G$ é uma base de autovetores (generalizados) que não são em geral ortogonais, e por isso a GFT não preserva o produto escalar.

Além disso, [15] mostra que ambas as abordagens ainda apresentam o problema da não-unicidade da definição da GFT, sempre que há frequências repetidas (λ_i ou γ_i). Pode-se ilustrar o problema com um exemplo numérico, utilizando um grafo completo de 3 vértices (um triângulo) não-ponderado (pesos unitários). Grafos completos de n vértices têm matriz Laplaciana *normalizada* com autovalores 0, com multiplicidade 1, e $n/(n-1)$, com multiplicidade $n-1$ [9, Exemplo 1.1]. Portanto, sua matriz Laplaciana não-normalizada pode ser expressa como

$$\begin{aligned}
\mathbf{L} &= \begin{bmatrix} 2 & -1 & -1 \\ -1 & 2 & -1 \\ -1 & -1 & 2 \end{bmatrix} = \mathbf{U}_1 \mathbf{\Gamma} \mathbf{U}_1^T \\
&= \begin{bmatrix} 1/\sqrt{3} & -1/\sqrt{2} & -1/\sqrt{2} \\ 1/\sqrt{3} & 0 & 1/\sqrt{2} \\ 1/\sqrt{3} & 1/\sqrt{2} & 0 \end{bmatrix} \begin{bmatrix} 0 & 0 & 0 \\ 0 & 3 & 0 \\ 0 & 0 & 3 \end{bmatrix} \begin{bmatrix} 1/\sqrt{3} & 1/\sqrt{3} & 1/\sqrt{3} \\ -1/\sqrt{2} & 0 & 1/\sqrt{2} \\ -1/\sqrt{2} & 1/\sqrt{2} & 0 \end{bmatrix},
\end{aligned}$$

mas também uma outra matriz de autovetores ortonormal é possível, substituindo os autovetores associados a $\gamma = 3$ pela soma e diferença (normalizadas) dos vetores $(-1/\sqrt{2}, 0, 1/\sqrt{2})^T$ e $(-1/\sqrt{2}, 1/\sqrt{2}, 0)^T$,

$$\mathbf{U}_2 = \begin{bmatrix} 1/\sqrt{3} & -2/\sqrt{6} & 0 \\ 1/\sqrt{3} & 1/\sqrt{6} & -1/\sqrt{2} \\ 1/\sqrt{3} & 1/\sqrt{6} & 1/\sqrt{2} \end{bmatrix}. \tag{7.3.7}$$

Isso leva a duas definições distintas da GFT, pois um mesmo sinal passa a ter um ou outro espectro, a depender se a matriz $\mathbf{U}_1$ ou $\mathbf{U}_2$ (ou qualquer uma das outras infinitas possibilidades) é considerada. Por exemplo, o sinal $\mathbf{x} = (-4 \;\; 0 \;\; 4)^T$ teria, neste caso, espectros dados por $\mathbf{U}_1^T \mathbf{x} \approx (0 \;\; 5{,}66 \;\; 0)^T$ e $\mathbf{U}_2^T \mathbf{x} \approx (0 \;\; 4{,}9 \;\; 2{,}83)^T$. Deri e Moura abordam este problema sugerindo o uso de projetores espectrais oblíquos (*oblique spectral projectors*) para obter uma representação única da GFT [15].

Embora GSP_L considere apenas os grafos não-direcionados, a teoria não impede *a priori* a aplicação a grafos direcionados, uma vez que a matriz Laplaciana continua sendo definida por $\mathbf{L} = \mathbf{D} - \mathbf{A}$, em que $\mathbf{D}$ é escolhida como a matriz de graus de entrada ou de saída. Poderíamos, por exemplo, manter a definição de frequência como sendo os autovalores de $\mathbf{L}$, pois teríamos que a norma $\|\mathbf{L}\mathbf{u}_i\|$ varia linearmente com *o módulo* de cada autovalor,

$$\|\mathbf{L}\mathbf{u}_i\| = \|\gamma_i \mathbf{u_i}\| = |\gamma_i|. \tag{7.3.8}$$

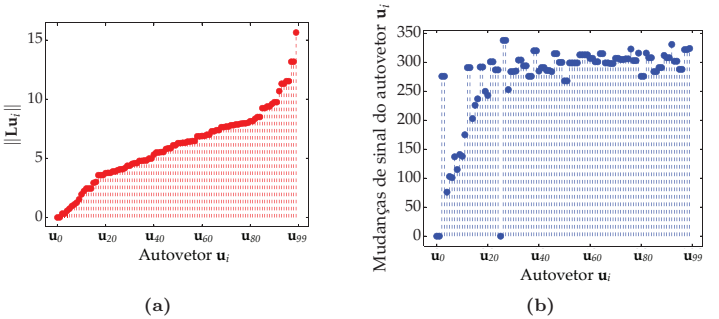


Figura 7.5: (a) Norma $\|\mathbf{Lu}\|$ e (a) número de mudanças de sinal dos autovetores do grafo direcionado da Fig. 2.3, dispostos segundo a ordem crescente dos módulos dos respectivos autovalores.

A Fig. 7.5 traz a norma $\|\mathbf{Lu}_i\|$ e o número de mudanças de sinal dos autovetores de $\mathbf{L}$ para o grafo *direcionado* da Fig. 2.3 (pág. 11), indicando que há relação entre o módulo dos autovalores e a suavidade do autovetor associado. No entanto, a matriz Laplaciana para grafos direcionados não é mais simétrica, por isso não é garantida ser positiva semi-definida ou diagonalizável.

Muitos outros problemas em GSP ainda se mantêm abertos à investigação, seja para desenvolver formas eficientes para se aplicar a teoria no contexto de redes e dados estruturados massivos (como o *método ágil inexato* de J. Deri [14, 15], que reduz o tempo de cálculo — de semanas para minutos — da GFT baseada em projetores espectrais), seja para melhor compreender e aplicar as técnicas utilizadas em cenários como os da seção 6. Por exemplo, os Teoremas da Amostragem (vide seção 4) e da Incerteza para grafos já foram extensivamente estudados [3, 7, 77], mas ainda há oportunidade para análise, como acerca da robustez de amostragem de sinais não-perfeitamente limitados em faixa [76]. Ortega *et al.* também ressaltam que muitas das ferramentas de GSP foram criadas para qualquer tipo de grafo, muito embora a investigação de grafos específicos possa gerar novas possibilidades em GSP, para certas aplicações [49]. Trata-se, enfim, de um campo de estudo recente, abrangente e transversal, com diversas opções de direções para tomar. Se é verdade o que disse David Hilbert, que "somente enquanto um ramo da ciência oferecer uma abundância de problemas ele estará vivo"[7], então GSP será uma área de pesquisa ativa ainda por muitos anos.

7.4 Problemas

Os dois problemas a seguir conduzem o leitor por dois cenários de aplicação de GSP. Os materiais e soluções estão disponíveis no repositório do Github `https://github.com/gboaviagem/gsp-ebook-exercises`.

[7] "*As long as a branch of science offers an abundance of problems, so long is it alive; a lack of problems foreshadows extinction or the cessation of independent development*". Discurso *Mathematical Problems*, Segundo Congresso Internacional de Matemáticos, Paris, 1900.

Problema 7.1. Nas Seções 6.3 e 6.4, vimos como a GFT é uma transformação que, por considerar o domínio do sinal, pode levar a representações compactas do sinal e, portanto, ser usada em compressão de dados. O mesmo fica evidente ao olhar o espectro na Fig. (6.1c). Neste exercício vamos realizar a compressão de uma imagem em escala de cinza. Antes de prosseguir, acesse o repositório *gsp-ebook-exercises* para fazer o *download* da imagem `cameraman.tif`.

(a) Utilizando a linguagem ou programa de sua escolha, leia o arquivo `cameraman.tif` e o armazene como uma matriz, ou *array*. Por exemplo, em Python, a leitura pode ser feita utilizando Matplotlib,

```
import matplotlib.pyplot as plt

im = plt.imread("cameraman.tif")
plt.imshow(im)  # Use para exibir a imagem
print(im.shape)
```

Código 7.6: Leitura da imagem.

O comando `print(im.shape)` exibe as dimensões do Numpy array criado. Verifique se o resultado obtido é (512, 512).

(b) Reduza a dimensão da imagem por um fator 8. Isto é, para cada linha e cada coluna, guarde um pixel e descarte os 7 seguintes, até ter percorrido toda a matriz. A intenção aqui é apenas reduzir o esforço computacional dos passos seguintes. O *shape* final deve ser $(64, 64)$ e a imagem deve ficar como na Fig. 7.6a.

(c) Antes de obter a matriz da GFT, é preciso ter a matriz de adjacência do grafo que é domínio da imagem (grafo da Fig. 1.3b). Felizmente, podemos usar o fato de que *o grafo em grade é o produto cartesiano entre dois grafos do tipo caminho*. A matriz de adjacência do produto cartesiano entre os grafos com matrizes $\mathbf{A}_1$, de dimensão $n_1 \times n_1$, e $\mathbf{A}_2$, de dimensão $n_2 \times n_2$, é dada por

$$\mathbf{A} = \mathbf{A}_1 \otimes \mathbb{I}_{n_2} + \mathbb{I}_{n_1} \otimes \mathbf{A}_2, \tag{7.4.9}$$

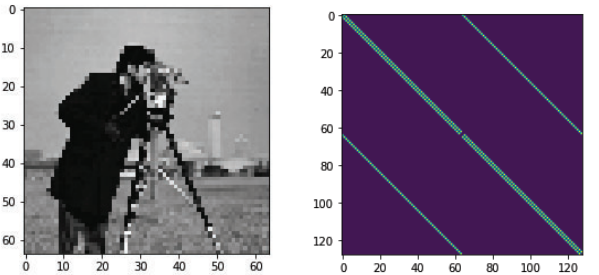
em que $\otimes$ indica o produto de Kronecker e $\mathbb{I}_n$ é a matriz identidade de ordem n.

Assuma que todos os pesos das arestas são unitários. Calcule a matriz $\mathbf{A}$.

Dica: em Python, experimente usar a função `numpy.kron()`. Em Matlab, seria a função `kron()`. Após calculada a matriz, ela terá dimensão $(64^2, 64^2)$. O seu canto superior esquerdo, de tamanho $(128, 128)$, foi exibido como uma imagem pseudo-colorida na Fig. 7.6b.

(d) De posse da matriz de adjacência $\mathbf{A}$, experimente visualizar a imagem reconstruída a partir das p componentes de frequência de maior magnitude, segundo $\mathrm{GSP}_{\mathrm{L}}$, para os valores $p = [5, 800, 1500, 2500, 3200]$.

Sugestão: construa uma função `reconstruct_from_p(x, U, Uinv, p)` que recebe o sinal sobre o grafo x, a base ortonormal $\mathbf{U}$ de autovetores de $\mathbf{L}$, sua inversa e o valor p. Lembre de, quando necessário, transformar o sinal em um vetor unidimensional ou em uma matriz 64×64. Em Python, supondo que `im` é o *array* lido no Código 7.6, a reconstrução e visualização para os valores pedidos de p poderiam ser feitos com o seguinte *loop*:



(a) (b)

Figura 7.6: (a) Imagem de teste `cameraman`, após amostragem para dimensão $(64, 64)$. (b) Visualização do canto superior esquerdo (primeiras 128 linhas e 128 colunas) da matriz de adjacência do grafo em grade, com pesos unitários, que serve de domínio para a Fig. 7.6a.

```
for p in [5, 800, 1500, 2500, 3200]:
    xr = reconstruct_from_p(im.ravel(), U, Uinv, p=p)
    imr = xr.reshape((64, 64))
    plt.figure()
    plt.imshow(np.abs(imr))
```

Código 7.7: Loop sobre os valores de p.

(e) Continuando o item anterior, repita o processo de utilizar apenas as p maiores componentes (valor absoluto) para reconstruir a imagem, mas agora comece com $p = 10$ e incremente em intervalos de 50. Ou seja, use os valores $[10, 60, 110, \ldots, 4010, 4060]$. Além disso, a cada iteração, calcule o ARE (vide a seção 6.2) e, ao fim, gere o gráfico ARE *vs.* p. O resultado deve ser como o da Fig. 7.7, onde se lê "Abordagem 1". Consulte seus dados calculados e responda: qual o menor valor de p (dentre esses utilizados no gráfico) que resulta numa ARE menor que 10%? Qual o percentual de compressão da imagem neste valor?

Observação: perceba que o percentual de compressão da imagem, ao utilizar apenas p componentes, pode ser definido como $(64^2 - p)/64^2 \times 100\%$.

Dica: usando o *slicing* de *arrays* Numpy, em Python, os valores de p são obtidos via `np.arange(10, len(U))[::50]`.

(f) Podemos melhorar o resultado do item anterior. Em vez de utilizar um grafo com pesos unitários, pode-se escolher pesos menores para arestas ligando amostras similares, de modo a minimizar a variação total do sinal no grafo. É um princípio semelhante ao utilizado na seção 6.3.

Em Python, o leitor pode aplicar o código 7.8, que define uma função a ser minimizada (exatamente $\| \mathbf{A}\mathbf{x} - \mathbf{x} \|_2$) e utiliza uma função auxiliar, `make_grid`, para obter a matriz de adjacência do grafo a partir de dois grafos do tipo caminho. Essa função contém simplesmente os cálculos da letra (c). Os pesos dos dois grafos-caminho formam, juntos, o *array* `weights`, que funciona como o conjunto de variáveis livres do problema de minimização. A convergência do

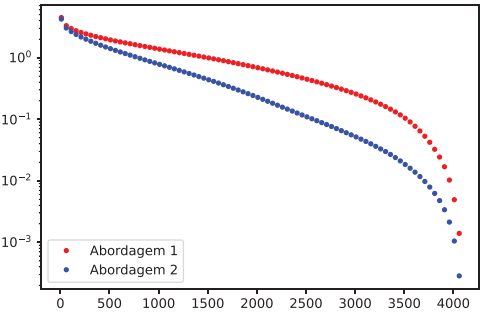


Figura 7.7: Valores de ARE *versus* quantidade p de componentes de frequência com maior magnitude. Na abordagem 1, o grafo em grade é contém apenas pesos unitários. Na abordagem 2, os pesos são obtidos a partir da minimização da variação total do sinal sobre o grafo, de modo que a GFT consegue uma compressão maior do sinal no domínio da frequência.

algoritmo pode ser verificada no atributo `res.success` e o vetor final de pesos fica armazenado em `res.x`. Com isso em mãos, após obter o novo grafo, calcule novamente: qual o menor valor de p (dentre aqueles utilizados no gráfico) que resulta numa ARE menor que 10%? E qual o percentual de compressão da imagem neste valor?

```
from scipy.optimize import minimize

def total_variation_from_weights(weights, im):
    """Total variation from image signal in a grid graph.

    The grid graph is built as a cartesian product between two
    path graphs, the weights of which are concatenated to
    form the `weights` input.

    Parameters
    ----------
    weights : array
        Concatenated weights of two path graphs. Its length equals
        N + M - 2, where N and M are taken from the shape of `im`.
    im : 2D array, shape=(N, M)
        Input image.

    """
    N, M = im.shape
    A, _ = make_grid(
        N, M, weights_r=weights[:(N-1)],
        weights_c=weights[(N-1):])
    x = im.ravel().reshape((N*M, 1))
    return np.linalg.norm(A @ x - x)

N, M = im.shape
rnd = np.random.RandomState(seed=2021)
# Initializing with gaussian random weights. If the initial
# weights are all equal, the optimizer has way more trouble
# finding an adjacency matrix on which the graph signal is smooth.
```

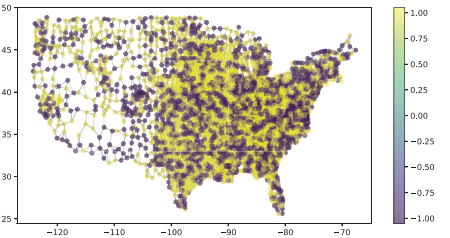


Figura 7.8: Grafo de condados americanos. As amostras do sinal sobre o grafo assumem o valor 1 (ou -1), se a taxa de obesidade entre adultos aumentou (diminuiu) de 2015 para 2016.

```
weights0 = 0.5 + 0.3 * rnd.randn(N-1 + M-1)
weights0[weights0 < 0] = 0

res = minimize(
    total_variation_from_weights, args=tuple([im]), x0=weights0,
    method='SLSQP',
    bounds=[(0, None) for i in range(len(weights0))],
    options=dict(disp=True, maxiter=5))

print("Successfull?", res.success)
weights = res.x

weights_naive = [1] * (N-1 + M-1)
print(
    "Total variation with unit weights:",
    total_variation_from_weights(weights_naive, im))

print(
    "Total variation with optimized weights:",
    total_variation_from_weights(weights, im))
```

Código 7.8: Encontrando o grafo que minimiza a variação do sinal.

Problema 7.2. Neste problema, iremos investigar a aplicação de GSP em um problema de classificação binária, implementando uma estratégia delineada por Sandryhaila e Moura [61], num dos artigos que lançaram as bases de GSP_A.

(a) Comece lendo a base de dados `us_obesity_by_county.csv`, que está no repositório do Github. Suas primeiras linhas são exibidas na Tabela 7.1. Os dados correspondem a registros de taxa de obesidade por condado americano nos anos de 2015 e 2016. Importe este arquivo *.csv* utilizando o método de sua preferência (em Python, a escolha mais comum seria o método `read_csv` do Pandas, ou o `loadtxt` do Numpy).

(b) Utilize as colunas *LONGITUDE* e *LATITUDE* (nesta ordem) como coordenadas e construa com elas um grafo de k vizinhos mais próximos. Utilize $k = 4$. Pondere as arestas segundo a eq. (1.2.2), com $2\theta^2 = 1$. Calcule e exiba a distribuição de grau do grafo gerado (grau *vs.* contagem de vértices com este grau).

Observação: ao construir um grafo dos k vizinhos mais próximos, é natural que alguns vértices tenham grau maior do que k.

	FIPS	LATITUDE	LONGITUDE	ADULT_ OBESITY_ 2015	ADULT_ OBESITY_ 2016	GRAPH_ SIGNAL
0	1001	32.536382	-86.644490	0.313	0.309	-1
1	1003	30.659218	-87.746067	0.250	0.267	1
2	1005	31.870670	-85.405456	0.384	0.408	1
3	1007	33.015893	-87.127148	0.373	0.401	1
4	1009	33.977448	-86.567246	0.309	0.324	1

Tabela 7.1: Primeiras cinco linhas dos dados para o exercício.

(c) Defina a coluna *GRAPH_SIGNAL* da base de dados como um sinal sobre o grafo. Os valores nessa coluna são iguais a 1 nos casos em que o respectivo condado observou um *aumento* na taxa de obesidade, entre 2015 e 2016. Caso contrário, atribui-se o valor -1. Visualize o sinal sobre o grafo (por exemplo, usando o `G.plot_signal()` do PyGSP). O resultado deve ser como na Figura 7.8.

(d) Divida o sinal sobre o grafo em uma partição com 3 conjuntos de amostras. Para isso, defina os conjuntos disjuntos de vértices $\mathcal{V}^{(train)}, \mathcal{V}^{(valid)}, \mathcal{V}^{(test)}$ (em que $\mathcal{V}$ é o conjunto de vértices do grafo), tais que

- $\mathcal{V}^{(train)} \cup \mathcal{V}^{(valid)} \cup \mathcal{V}^{(test)} = \mathcal{V}$,
- $|\mathcal{V}^{(train)}| = |\mathcal{V}^{(valid)}| = 0{,}1|\mathcal{V}|$,
- $|\mathcal{V}^{(test)}| = 0{,}8|\mathcal{V}|$.

Crie os sinais

- $\mathbf{s}^{(known)}$, cujas amostras coincidem com o sinal original nos vértices em $\mathcal{V}^{(train)} \cup \mathcal{V}^{(valid)}$, mas valem zero nas amostras em $\mathcal{V}^{(test)}$.
- $\mathbf{s}^{(train)}$, que corresponde ao sinal original apenas nos vértices $\mathcal{V}^{(train)}$, valendo zero nos demais.

(e) Vamos agora utilizar o fato de que um filtro LSI sobre o grafo é um operador local, i. e. cada amostra em um sinal filtrado é a soma ponderada (pelos coeficientes do filtro e pelos pesos das arestas) de amostras vizinhas. A ideia é calcular um filtro adaptativo $h(\mathbf{A})$ e assumir que, utilizando apenas as amostras em $\mathcal{V}^{(train)}$, ele classifica corretamente os vértices em $\mathcal{V}^{(train)} \cup \mathcal{V}^{(valid)}$ se as n-ésimas amostras dos sinais $h(\mathbf{A})\mathbf{s}^{(train)}$ e $\mathbf{s}^{(known)}$ coincidem. Desta forma, a filtragem LSI é usada como um classificador binário.

A hipótese enunciada há pouco pode ser traduzida[8] em

$$\mathbf{D}h(\mathbf{A})\mathbf{s}^{(train)} \geq 0, \tag{7.4.10}$$

em que $\mathbf{D} = \operatorname{diag}\left(\mathbf{s}^{(known)}\right)$.

Assumindo que h possui ordem L, e representando o vetor de coeficientes do filtro como $\mathbf{h} = (h_0 \dots h_L)^T$, encontrar h traduz-se em um problema de mini-

[8] O leitor é encorajado a complementar o entendimento desta explicação no artigo original [61]

mização por mínimos quadrados,

$$\begin{aligned}\mathbf{h}_{opt} &= \text{argmin} \parallel \mathbf{D}h(\mathbf{A})\mathbf{s}^{(train)} - \mathbf{1}_N \parallel_2 \\ &= \text{argmin} \parallel \underbrace{(\mathbf{D}\mathbf{A}^0\mathbf{s}^{(train)} \ldots \mathbf{D}\mathbf{A}^L\mathbf{s}^{(train)})}_{=\mathbf{M}} \mathbf{h} - \mathbf{1}_N \parallel_2 .\end{aligned} \tag{7.4.11}$$

Considere $L = 10$ e utilize algum pacote de otimização para calcular $\mathbf{h}_{opt}$.

Dica: calcule previamente a matriz $\mathbf{M}$, cuja i-ésima coluna contém o vetor $\mathbf{D}\mathbf{A}^i\mathbf{s}^{(train)}$. Em Python, o leitor pode procurar o módulo `scipy.optimize`.

(f) Gere o gráfico da resposta em frequência do filtro calculado no item anterior. Qual o perfil de resposta: passa-baixas, passa-altas, rejeita-faixa ou passa-banda? Comente se o resultado já era esperado.

(g) Execute a predição dos valores nas amostras desconhecidas (sobre o conjunto $\mathcal{V}^{(test)}$), a partir do sinal $h(\mathbf{A})\mathbf{s}^{(known)}$. Substitua cada amostra positiva por $+1$, e cada negativa por -1. Calcule a acurácia (quantidade de acertos sobre número de amostras no conjunto) da previsão em $\mathcal{V}^{(test)}$. Comente o resultado.

Bibliografia

[1] BENZI, K. et al. Song recommendation with non-negative matrix factorization and graph total variation. In: IEEE. *Acoustics, Speech and Signal Processing (ICASSP), 2016 IEEE International Conference on*. [S.l.], 2016. p. 2439–2443.

[2] BONDY, J. A.; MURTY, U. S. R. *Graph theory, volume 244 of Graduate Texts in Mathematics*. [S.l.]: Springer, New York, 2008.

[3] CHAMON, L. F. O.; RIBEIRO, A. Greedy sampling of graph signals. *IEEE Trans. Signal Process.*, Institute of Electrical and Electronics Engineers (IEEE), v. 66, n. 1, p. 34–47, jan 2018. Disponível em: <`https://doi.org/10.1109/tsp.2017.2755586`>.

[4] CHEN, S. et al. Semi-supervised multiresolution classification using adaptive graph filtering with application to indirect bridge structural health monitoring. *IEEE Trans. Signal Process.*, IEEE, v. 62, n. 11, p. 2879–2893, 2014.

[5] CHEN, S.; SANDRYHAILA, A.; KOVAČEVIĆ, J. Sampling theory for graph signals. In: *2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. [S.l.: s.n.], 2015. p. 3392–3396. ISSN 1520-6149.

[6] CHEN, S. et al. Discrete signal processing on graphs: Sampling theory. *IEEE Trans. Signal Process.*, Institute of Electrical and Electronics Engineers (IEEE), v. 63, n. 24, p. 6510–6523, dec 2015. Disponível em: <`https://doi.org/10.1109%2Ftsp.2015.2469645`>.

[7] CHEN, S. et al. Signal recovery on graphs: Fundamental limits of sampling strategies. *IEEE Trans. Signal Inf. Process. Netw.*, IEEE, v. 2, n. 4, p. 539–554, 2016.

[8] CHEUNG, G. et al. Graph spectral image processing. *Proc. IEEE*, Institute of Electrical and Electronics Engineers (IEEE), p. 1–24, 2018. Disponível em: <`https://doi.org/10.1109/jproc.2018.2799702`>.

[9] CHUNG, F. R. K. *Spectral graph theory*. [S.l.]: American Mathematical Soc., 1997.

[10] COIFMAN, R. R.; MAGGIONI, M. Diffusion wavelets. *Applied and Computational Harmonic Analysis*, Elsevier, v. 21, n. 1, p. 53–94, 2006.

[11] COL, A. D. et al. Wavelet-based visual analysis of dynamic networks. *IEEE transactions on visualization and computer graphics*, IEEE, v. 24, n. 8, p. 2456–2469, 2017.

[12] CROVELLA, M.; KOLACZYK, E. Graph wavelets for spatial traffic analysis. *IEEE INFOCOM 2003*, v. 3, p. 1848–1857, 2003.

[13] DEFFERRARD, M. et al. *PyGSP: Graph Signal Processing In Python*. Zenodo, 2017. Disponível em: <https://zenodo.org/record/1003158>.

[14] DERI, J. A. *Graph Signal Processing: Structure and Scalability to Massive Data Sets*. Tese (Doutorado) — Carnegie Mellon University, 2016. Disponível em: <http://repository.cmu.edu/dissertations/725>.

[15] DERI, J. A.; MOURA, J. M. Spectral projector-based graph Fourier transforms. *IEEE J. Sel. Topics Signal Process.*, IEEE, v. 11, n. 6, p. 785–795, 2017.

[16] DINIZ, P. S. R. *Adaptive Filtering: Algorithms and Practical Implementation*. [S.l.]: Springer, 2013.

[17] DINIZ, P. S. R.; SILVA, E. A. B. da; NETTO, S. L. *Digital Signal Processing: System Analysis and Design*. [S.l.]: Cambridge University Press, 2019.

[18] ELIAS, V.; MARTINS, W. On the use of graph fourier transform for light-field compression. *Journal of Communication and Information Systems*, Sociedad Brasileira de Telecomunicacoes, v. 33, n. 1, p. 92–103, 2018. Disponível em: <https://doi.org/10.14209/jcis.2018.10>.

[19] FEOFILOFF, P.; KOHAYAKAWA, Y.; WAKABAYASHI, Y. Uma Introdução Sucinta à Teoria dos Grafos. 2011. https://www.ime.usp.br/~pf/teoriadosgrafos/.

[20] FRACASTORO, G.; FOSSON, S. M.; MAGLI, E. Steerable discrete cosine transform. *IEEE Trans. Image Process.*, IEEE, v. 26, n. 1, p. 303–314, 2017.

[21] FRITSCH, F. N.; CARLSON, R. E. Monotone piecewise cubic interpolation. *SIAM Journal on Numerical Analysis*, SIAM, v. 17, n. 2, p. 238–246, 1980.

[22] GALLIER, J. *Spectral theory of unsigned and signed graphs – applications to graph clustering: a survey*. University of Pennsylvania (Course notes, December 2017. Disponível em: <http://www.cis.upenn.edu/ jean/spectral-graph-notes.pdf>.

[23] GIRAULT, B. et al. Grasp: A matlab toolbox for graph signal processing. In: *2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2017. Disponível em: <https://doi.org/10.1109/icassp.2017.8005300>.

[24] GOLDSBERRY, L. et al. Brain signal analytics from graph signal processing perspective. In: IEEE. *Acoustics, Speech and Signal Processing (ICASSP), 2017 IEEE International Conference on*. [S.l.], 2017. p. 851–855.

[25] HAMMOND, D. K.; VANDERGHEYNST, P.; GRIBONVAL, R. Wavelets on graphs via spectral graph theory. *Applied and Computational Harmonic Analysis*, Elsevier, v. 30, n. 2, p. 129–150, 2011.

[26] HAMMOND, D. K.; VANDERGHEYNST, P.; GRIBONVAL, R. Wavelets on graphs via spectral graph theory. *Applied and Computational Harmonic Analysis*, Elsevier, v. 30, n. 2, p. 129–150, 2011.

[27] HU, W. et al. Multiresolution graph fourier transform for compression of piecewise smooth images. *IEEE Transactions on Image Processing*, Institute of Electrical and Electronics Engineers (IEEE), v. 24, n. 1, p. 419–433, jan 2015. Disponível em: <https://doi.org/10.1109/tip.2014.2378055>.

[28] HUANG, W. et al. Graph frequency analysis of brain signals. *IEEE J. Sel. Topics Signal Process.*, IEEE, v. 10, n. 7, p. 1189–1203, 2016.

[29] HUTCHINSON, M. F. A stochastic estimator of the trace of the influence matrix for laplacian smoothing splines. *Communications in Statistics-Simulation and Computation*, Taylor & Francis, v. 19, n. 2, p. 433–450, 1990.

[30] JIN, Y.; SHUMAN, D. I. an M-channel critically sampled filter bank for graph signals. p. 3909–3913, 2017.

[31] LANG, S. *Álgebra Linear*. [S.l.]: Editora Ciência Moderna, 2003.

[32] LEONARDI, N.; VILLE, D. V. D. Tight wavelet frames on multislice graphs. *IEEE Transactions on Signal Processing*, IEEE, v. 61, n. 13, p. 3357–3367, 2013.

[33] LEONARDI, N.; VILLE, D. V. D. Tight wavelet frames on multislice graphs. *IEEE Trans. Signal Process.*, IEEE, v. 61, n. 13, p. 3357–3367, 2013.

[34] LEVOY, M.; HANRAHAN, P. Light field rendering. In: *Proceedings of the 23rd Annual Conference on Computer Graphics and Interactive Techniques*. New York, NY, USA: ACM, 1996. (SIGGRAPH '96), p. 31–42. ISBN 0-89791-746-4. Disponível em: <http://doi.acm.org/10.1145/237170.237199>.

[35] LIN, T. *Benchmark of popular graph/network packages*. Acessado em: 20 de outubro de 2020. Disponível em: <https://www.timlrx.com/2020/05/10/benchmark-of-popular-graph-network-packages-v2/>.

[36] LORENZO, P. D.; BANELLI, P.; BARBAROSSA, S. Optimal sampling strategies for adaptive learning of graph signals. In: *2017 25th European Signal Processing Conference (EUSIPCO)*. [S.l.: s.n.], 2017. p. 1684–1688.

[37] LORENZO, P. D. et al. Adaptive graph signal processing: Algorithms and optimal sampling strategies. *CoRR*, abs/1709.03726, 2017. Disponível em: <http://arxiv.org/abs/1709.03726>.

[38] LORENZO, P. D. et al. Adaptive least mean squares estimation of graph signals. *IEEE Transactions on Signal and Information Processing over Networks*, v. 2, n. 4, p. 555–568, Dec 2016. ISSN 2373-776X.

[39] LORENZO, P. D. et al. Distributed recursive least squares strategies for adaptive reconstruction of graph signals. In: *2017 25th European Signal Processing Conference (EUSIPCO)*. [S.l.: s.n.], 2017. p. 2289–2293.

[40] LUCCHESI, C. L. *Introdução à teoria dos grafos*. [S.l.]: IMPA, 1979.

[41] MAGGIONI, M. et al. Biorthogonal diffusion wavelets for multiscale representation on manifolds and graphs. In: INTERNATIONAL SOCIETY FOR OPTICS AND PHOTONICS. *Wavelets XI*. [S.l.], 2005. v. 5914, p. 59141M.

[42] MALLAT, S. *A wavelet tour of signal processing*. [S.l.]: Elsevier, 1999.

[43] MASOUMI, M.; REZAEI, M.; HAMZA, A. B. Shape analysis of carpal bones using spectral graph wavelets. In: *Vertex-Frequency Analysis of Graph Signals*. [S.l.]: Springer, 2019. p. 419–436.

[44] MEI, J.; MOURA, J. M. Signal processing on graphs: Performance of graph structure estimation. In: IEEE. *Acoustics, Speech and Signal Processing (ICASSP), 2016 IEEE International Conference on*. [S.l.], 2016. p. 6165–6169.

[45] MEI, J.; MOURA, J. M. F. Signal processing on graphs: Causal modeling of unstructured data. *IEEE Trans. Signal Process.*, IEEE, 2016.

[46] NGUYEN, H. Q.; CHOU, P. A.; CHEN, Y. Compression of human body sequences using graph wavelet filter banks. In: IEEE. *Acoustics, Speech and Signal Processing (ICASSP), 2014 IEEE International Conference on*. [S.l.], 2014. p. 6152–6156.

[47] NOCEDAL, J.; WRIGHT, S. J. *Numerical optimization 2nd*. [S.l.]: Springer, 2006.

[48] OPPENHEIM, A. V.; WILLSKY, A. S.; NAWAB, S. H. *Signals and Systems*. Prentice Hall, 1997. (Prentice-Hall signal processing series). ISBN 9780138147570. Disponível em: <https://books.google.com.br/books?id=LwQqAQAAMAAJ>.

[49] ORTEGA, A. et al. Graph signal processing: Overview, challenges, and applications. *Proc. IEEE*, Institute of Electrical and Electronics Engineers (IEEE), v. 106, n. 5, p. 808–828, may 2018. Disponível em: <`https://doi.org/10.1109/jproc.2018.2820126`>.

[50] Perraudin, N. et al. GSPBOX: A toolbox for signal processing on graphs. *ArXiv e-prints*, ago. 2014.

[51] PIRAYRE, A. et al. Brane cut: biologically-related a priori network enhancement with graph cuts for gene regulatory network inference. *BMC Bioinformatics*, BioMed Central, v. 16, n. 1, p. 368, 2015.

[52] PIRAYRE, A. et al. Brane clust: Cluster-assisted gene regulatory network inference refinement. *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, IEEE, 2017.

[53] PÜSCHEL, M.; MOURA, J. M. F. Algebraic Signal Processing Theory. *CoRR*, abs/cs/0612077, 2006. Disponível em: <`http://arxiv.org/abs/cs/0612077`>.

[54] PÜSCHEL, M.; MOURA, J. M. F. Algebraic signal processing theory: 1-D space. *IEEE Trans. Signal Process.*, v. 56, n. 8, p. 3586–3599, 2008.

[55] PÜSCHEL, M.; MOURA, J. M. F. Algebraic signal processing theory: Foundation and 1-D time. *IEEE Trans. Signal Process.*, v. 56, n. 8, p. 3572–3585, 2008.

[56] ROSSI, M.; FROSSARD, P. Graph-based light field super-resolution. In: IEEE. *Multimedia Signal Processing (MMSP), 2017 IEEE 19th International Workshop on*. [S.l.], 2017. p. 1–6.

[57] SAKIYAMA, A.; WATANABE, K.; TANAKA, Y. Spectral graph wavelets and filter banks with low approximation error. *IEEE Transactions on Signal and Information Processing over Networks*, IEEE, v. 2, n. 3, p. 230–245, 2016.

[58] SANDRYHAILA, A.; MOURA, J. M. Nearest-neighbor image model. In: IEEE. *Image Processing (ICIP), 2012 19th IEEE International Conference on*. [S.l.], 2012. p. 2521–2524.

[59] SANDRYHAILA, A.; MOURA, J. M. Discrete signal processing on graphs: Frequency analysis. *IEEE Transactions on Signal Processing*, IEEE, v. 62, n. 12, p. 3042–3054, 2014.

[60] SANDRYHAILA, A.; MOURA, J. M. F. Discrete signal processing on graphs. *IEEE Trans. Signal Process.*, IEEE, v. 61, n. 7, p. 1644–1656, 2013.

[61] SANDRYHAILA, A.; MOURA, J. M. F. Discrete signal processing on graphs: Graph filters. In: *2013 IEEE International Conference on Acoustics, Speech and Signal Processing*. Institute of Electrical and Electronics Engineers (IEEE), 2013. Disponível em: <`https://doi.org/10.1109%2Ficassp.2013.6638849`>.

[62] SANDRYHAILA, A.; MOURA, J. M. F. Discrete signal processing on graphs: Graph Fourier transform. In: *2013 IEEE International Conference on Acoustics, Speech and Signal Processing*. Institute of Electrical and Electronics Engineers (IEEE), 2013. Disponível em: <`https://doi.org/10.1109%2Ficassp.2013.6638850`>.

[63] SANDRYHAILA, A.; MOURA, J. M. F. Big data analysis with signal processing on graphs: Representation and processing of massive data sets with irregular structure. *IEEE Signal Process. Mag.*, IEEE, v. 31, n. 5, p. 80–90, 2014.

[64] SANDRYHAILA, A.; MOURA, J. M. F. Discrete signal processing on graphs: Frequency analysis. *IEEE Trans. Signal Process.*, Institute of Electrical and Electronics Engineers (IEEE), v. 62, n. 12, p. 3042–3054, jun 2014. Disponível em: <`https://doi.org/10.1109%2Ftsp.2014.2321121`>.

[65] SARDELLITTI, S.; BARBAROSSA, S.; LORENZO, P. D. Graph topology inference based on transform learning. In: *2016 IEEE Global Conference on Signal and Information Processing (GlobalSIP)*. IEEE, 2016. Disponível em: <https://doi.org/10.1109/globalsip.2016.7905863>.

[66] SARDELLITTI, S.; BARBAROSSA, S.; LORENZO, P. D. On the graph Fourier transform for directed graphs. *IEEE J. Sel. Topics Signal Process.*, IEEE, 2017.

[67] SHEN, G. et al. Edge-adaptive transforms for efficient depth map coding. In: *28th Picture Coding Symposium*. IEEE, 2010. Disponível em: <`https://doi.org/10.1109/pcs.2010.5702565`>.

[68] SHUMAN, D. I. et al. The emerging field of signal processing on graphs: Extending high-dimensional data analysis to networks and other irregular domains. *IEEE Signal Process. Mag.*, IEEE, v. 30, n. 3, p. 83–98, 2013.

[69] SHUMAN, D. I.; RICAUD, B.; VANDERGHEYNST, P. A windowed graph Fourier transform. In: IEEE. *Statistical Signal Processing Workshop (SSP), 2012 IEEE*. [S.l.], 2012. p. 133–136.

[70] SHUMAN, D. I.; RICAUD, B.; VANDERGHEYNST, P. A windowed graph fourier transform. In: IEEE. *2012 IEEE Statistical Signal Processing Workshop (SSP)*. [S.l.], 2012. p. 133–136.

[71] SHUMAN, D. I.; VANDERGHEYNST, P.; FROSSARD, P. Chebyshev polynomial approximation for distributed signal processing. In: IEEE. *2011 International Conference on Distributed Computing in Sensor Systems and Workshops (DCOSS)*. [S.l.], 2011. p. 1–8.

[72] SHUMAN, D. I. et al. Spectrum-adapted tight graph wavelet and vertex-frequency frames. *IEEE Transactions on Signal Processing*, IEEE, v. 63, n. 16, p. 4223–4235, 2015.

[73] STRANG, G. The discrete cosine transform. *SIAM review*, SIAM, v. 41, n. 1, p. 135–147, 1999.

[74] SU, X. et al. Graph-based light fields representation and coding using geometry information. In: *IEEE International Conference on Image Processing (ICIP)*. [S.l.: s.n.], 2017.

[75] TREMBLAY, N.; BORGNAT, P. Graph wavelets for multiscale community mining. *IEEE Trans. Signal Process.*, IEEE, v. 62, n. 20, p. 5227–5239, 2014.

[76] TSITSVERO, M.; BARBAROSSA, S.; LORENZO, P. D. Signals on graphs: Uncertainty principle and sampling. *IEEE Trans. Signal Process.*, IEEE, v. 64, n. 18, p. 4845–4860, 2016.

[77] WANG, X.; CHEN, J.; GU, Y. Generalized graph signal sampling and reconstruction. In: IEEE. *Signal and Information Processing (GlobalSIP), 2015 IEEE Global Conference on*. [S.l.], 2015. p. 567–571.

[78] WIDROW, B.; HOFF, M. E. Adaptive switching circuits. *WESCOM Conv. Rec.*, n. pt 4, p. 96–140, Aug 1960. ISSN 0018-9219.

[79] WIDROW, B. et al. Stationary and nonstationary learning characteristics of the LMS adaptive filter. *Proceedings of the IEEE*, v. 64, n. 8, p. 1151–1162, Aug 1976. ISSN 0018-9219.