

Universidade de São Paulo  
Instituto de Matemática e Estatística  
Matemática Aplicada - Bacharelado - Habilitação em Métodos Matemáticos

Rafael Pereira Lima

# **Rastreamento de Olhar Usando Compressive Sensing**

São Paulo  
Dezembro de 2016

# Rastreamento de Olhar Usando Compressive Sensing

Monografia final da disciplina  
MAP2080 – Trabalho de Formatura.

Supervisor: Prof. Dr. Carlos Hitoshi Morimoto

São Paulo

Dezembro de 2016

# Resumo

Informações do olhar podem ser usadas para melhorar a experiência do usuário ou possibilitar a interação com o computador quando, por limitações físicas, a pessoa é incapaz de interagir com dispositivos de entrada usuais. Rastreamento de olhar consiste na identificação da posição do olhar durante determinado período. Desenvolvemos um programa de rastreamento de olhar usando *Compressive Sensing*, uma técnica usada para reconstruir sinais a partir de poucas amostras. Os resultados obtidos foram semelhantes aos presentes na literatura.

**Palavras-chave:** *Compressive Sensing*, rastreamento de olhar, processamento de imagens.

# Abstract

Gaze information can be used to improve user experience or enable human-computer interaction when the person is unable to use common input devices due to physical disabilities. Gaze tracking is the gaze position estimation in some period of time. We developed a gaze tracking algorithm using Compressive Sensing, a technique used to recover signals from few samples. The results are similar to those in the literature.

**Keywords:** Compressive Sensing, gaze tracking, image processing.

# Sumário

<b>1</b>	<b>Introdução</b>	<b>6</b>
<b>2</b>	<b>Rastreamento de olhar</b>	<b>8</b>
2.1	Características em rastreamento de olhar . . . . .	10
<b>3</b>	<b><i>Compressive Sensing</i></b>	<b>12</b>
3.1	Unicidade de $(P_0)$ . . . . .	16
3.2	Unicidade de $(P_1)$ . . . . .	19
3.3	Equivalência entre $(P_0)$ e $(P_1)$ . . . . .	22
3.4	Matrizes que satisfazem as condições de CS . . . . .	27
<b>4</b>	<b>O algoritmo de homotopia</b>	<b>29</b>
4.1	Exemplo . . . . .	33
<b>5</b>	<b>O modelo <i>cross-and-bouquet</i></b>	<b>35</b>
<b>6</b>	<b>Desenvolvimento de um rastreador de olhar usando CS</b>	<b>37</b>
6.1	Coleta de amostras . . . . .	38
6.2	Rastreamento de olhar . . . . .	40
<b>7</b>	<b>Conclusão</b>	<b>44</b>

# Capítulo 1

## Introdução

Um dos problemas enfrentados por pessoas com deficiência motora é a falta de acesso à informação devido à impossibilidade ou dificuldade para usar dispositivos eletrônicos. Uma possível solução para esse problema seria usar o movimento dos olhos para interagir com o mundo exterior. O uso de técnicas para identificar onde determinada pessoa está olhando é chamado de rastreamento de olhar.

Para rastrear o olhar geralmente uma câmera é posicionada na frente de um dos olhos. A partir da imagem obtida pela câmera, técnicas de processamento de imagens são usadas para identificar a posição da íris e, a partir disso, identificar a direção em que o usuário está olhando. No entanto, o rastreamento de olhar é dificultado por vários fatores, como mudanças na iluminação do lugar e oclusão parcial dos olhos causada por alguma expressão facial. Portanto, não é uma tarefa trivial mas, apesar dessas dificuldades, deve ser executada em tempo real.

Uma possível solução seria processar não a imagem original, mas uma imagem menor e com as mesmas características desejadas encontradas na primeira.

Uma imagem pode ser representada como uma combinação linear de vetores que representam senoides de frequências diferentes. Empiricamente é assumido que frequências mais altas influenciam pouco no resultado final e, por esse motivo, em muitas aplicações essas

frequências são descartadas para compressão de imagens, ou seja, reduzir o 'espaço físico' usado para armazenar o arquivo.

A abordagem sugerida pelo Compressive Sensing (CS) é um pouco diferente: em vez de assumir quais vetores podem ser descartados, é assumido que uma determinada quantidade de vetores exerce pouca ou nenhuma influência no sinal. Chamamos de esparso um sinal que depende apenas de uma pequena quantidade de vetores da base. Quando o sinal depende de vários vetores, porém apenas uma pequena quantidade influencia significativamente no resultado final, esse sinal é conhecido como 'compressive'. Em muitos casos CS permite reconstruir, sem perda de qualidade, uma imagem usando menos amostras do que as necessárias para reconstruir imagens usando técnicas tradicionais de compressão de imagens.

# Capítulo 2

## Rastreamento de olhar

Rastreamento de olhar consiste em determinar a direção do olhar, ou seja, a partir da posição do olho identificar para onde uma pessoa está olhando em determinado instante e também mudanças na direção do olhar em determinado período [12].

Podemos usar o rastreamento de olhar para construir interfaces para ajudar pessoas com dificuldades motoras a interagir com computadores [10]. Informações de olhar também podem ser usadas em pesquisas com o objetivo de entender como o olho se move quando uma pessoa realiza diferentes atividades [12].

Através da análise da posição do olhar é possível avaliar a influência de um anúncio sobre a atenção dos consumidores e, assim, ajudar na criação de propagandas mais eficientes [4].

O olhar também pode ser usado como um indicador de usabilidade de interfaces de *software*, sendo usado em estudos de interação humano-computador (IHC) [4].

Algumas partes do olho humano, que estão representadas na Figura 2.1, são as seguintes [10]:

- **pupila:** a abertura que permite a entrada de luz no olho.
- **íris:** o músculo colorido que controla o diâmetro da pupila.
- **esclera:** tecido branco protetor que envolve o restante do olho.



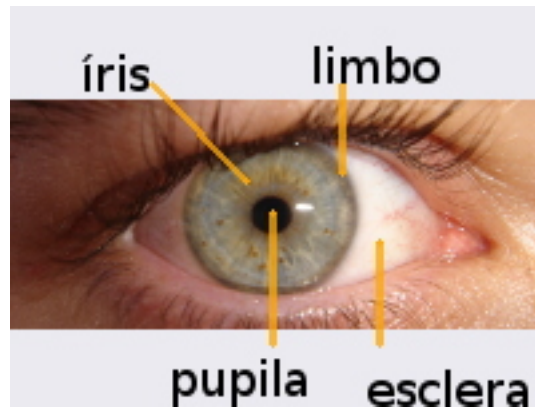


Figura 2.1: Regiões do olho. Adaptado de <sup>1</sup>

- **limbo**: o contorno entre a íris e a esclera.

Técnicas de rastreamento de olhar podem ser divididas em três modalidades [13]:

1. **Eletro-oculografia**, que registra diferenças de potencial elétrico na pele ao redor da cavidade ocular.
2. **Lente de contato**, uma bobina é instalada no olho sobre uma lente de contato e a posição do olho é estimada ao medir o campo eletromagnético. [3]. Este método é o mais preciso para medir movimentos do olho, porém é o mais invasivo [3].
3. **Rastreamento baseado em vídeo**, que usa técnicas de processamento de imagens para identificar a posição da pupila nas imagens do olho.

O rastreamento baseado em vídeo é a modalidade de rastreamento de olhar menos invasiva.

Dois tipos de imagem são comumente usados no rastreamento baseado em vídeo: imagens no espectro visual ou imagens em infravermelho.

---

<sup>1</sup>[http://commons.wikimedia.org/wiki/File:My\\_eye.jpg](http://commons.wikimedia.org/wiki/File:My_eye.jpg)

No rastreamento pelo espectro visual, a luz refletida pelos olhos é registrada. Neste caso, a eficiência do rastreamento depende das condições de iluminação do ambiente, o que torna o processo complicado [10].

No caso de imagens em infravermelho, não temos esse problema, pois o olho é constantemente iluminado por uma fonte de luz infravermelha, que não é percebida pelo usuário. Uma vantagem desta abordagem é que a pupila reflete parte da luz infravermelha recebida, sendo a região mais brilhante do olho na imagem [10]. Devido ao seu tamanho e posição, a pupila tem menor chance de ser parcialmente ocultada pelos cílios do que a esclera, outra região que reflete o infravermelho.

A principal desvantagem é que não é possível usar esse tipo de rastreador ao ar livre durante o dia devido à luz infravermelha do ambiente [10].

As técnicas de rastreamento de olhar também variam de acordo com a localização da câmera, que pode ser instalada junto à cabeça do usuário (*head-mounted*) ou remotamente. No caso do rastreamento *head-mounted*, o usuário deve usar um acessório equipado com uma câmera que registra as imagens do olho. A figura 2.2 mostra um rastreador desse tipo

Uma vantagem do rastreamento *head-mounted* é que a câmera se move com a cabeça, então mudanças de pose não causam mudanças da posição da pupila na imagem. A desvantagem é a necessidade de usar um equipamento acoplado à cabeça. Por estar muito próxima do olho, a câmera do olho pode ocultar parte do campo de visão, dificultando a interação com o usuário.

## 2.1 Características em rastreamento de olhar

Em algumas aplicações de rastreamento de olhar não estamos interessados apenas na direção do olhar em determinado instante, mas também podemos querer observar outras características relacionadas ao movimento dos olhos. As características mais específicas são [12]:



Figura 2.2: Um rastreador de olhar acoplado à cabeça. Reproduzido de <sup>2</sup>

- **Fixação e sacada:** Em 1989, Emile Java (oftalmologista francês) observou que os movimentos do olho não ocorrem de forma contínua, e sim de movimentos rápidos, chamados de sacadas, seguidos por breves pausas, conhecidas como fixações;
- **Área de interesse:** Região no ambiente que está presente no campo visual e que é de interesse em determinada pesquisa;
- **Duração do olhar:** Duração de um intervalo de tempo em que uma série de fixações está dentro de uma área de interesse;
- **Caminho de varredura:** localização espacial de uma sequência de fixações.

---

<sup>2</sup><https://pupil-labs.com/blog/2014-01/new-pupil-pro-headset-capture-software-0-3-7/>

# Capítulo 3

## *Compressive Sensing*

*Compressive Sensing* (CS), também conhecido como *Compressive Sampling* ou *Compressed Sensing*, estuda formas de reconstruir um vetor a partir de poucas amostras. De agora em diante, será assumido que as amostras  $(y_i)_{i=1}^m$ , representadas pelo vetor  $y \in \mathbb{R}^m$  de um sinal  $x \in \mathbb{R}^n$  são obtidas a partir de uma transformação linear  $y = Ax$ , onde  $A$  é uma matriz  $n \times m$ , com  $n < m$ .

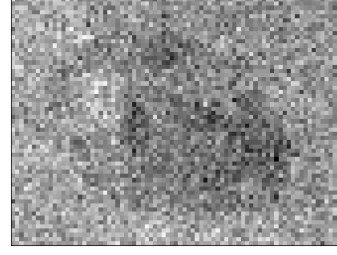
**Observação:** Durante o texto, sempre que referirmos a um vetor como uma imagem, assumimos que o vetor é calculado ao empilhar todas as linhas da imagem em escala de cinza.

Como  $n < m$ , a equação não possui uma única solução  $x$ , se existir. Poderíamos calcular  $x$  como  $x = A^\dagger y$ , resolvendo o problema de mínimos quadrados, ou seja, encontrando  $x$  que minimiza  $\|Ax - y\|_2$ . Porém, as imagens recuperadas dessa forma não são parecidas com a imagem original. A Figura 3.1 mostra uma imagem recuperada pelo método de mínimos quadrados para uma matriz  $A$   $m \times n$ , onde  $m = 80\% \cdot n$  (compressão de 20%) e as entradas de  $A$  são iid  $\mathcal{N}(0, 1)$ .

Sabemos que, geralmente, imagens são esparsas no espectro da frequência. Uma imagem pode ser decomposta como uma combinação linear de senoidais e, na prática, os coeficientes correspondentes às frequências mais altas são muito próximos de zero. Na Figura 3.2 a



(a) imagem original



(b) compressão de 20%

Figura 3.1: Recuperando imagem usando o método de mínimos quadrados para  $y = Ax$ , onde o vetor  $x$  representa a imagem original e  $A$  é uma matriz  $m \times n$  com  $m = 80\% \cdot n$  cujas entradas são iid e seguem uma distribuição  $\mathcal{N}(0, 1)$ . Adaptado de <sup>1</sup>

imagem é reconstruída apenas com as frequências cujos coeficientes estão entre os 60% maior valor absoluto (compressão de 40%). Note que as imagens **(a)** e **(b)** são bastante parecidas, indicando que a imagem é esparsa, ou aproximadamente esparsa (com muitos coeficientes muito próximos de zero) no espectro da frequência.

A ideia é então escolher, entre todas as soluções possíveis, o vetor  $x$  com o maior número de coeficientes nulos. Para entender melhor o problema, as seguintes definições, retiradas de [1], serão enunciadas:

**Def:** Seja  $x \in \mathbb{R}^n$ .  $x$  é  $k$ -esparso se possui, no mínimo,  $k$  coeficientes nulos.

**Def:** Dado  $x \in \mathbb{R}^n$ , a ‘norma’ 0 de  $x$  é definida como  $\|x\|_0 = \#\{i : |x_i| > 0\}$

Observe que a ‘norma’ 0 não é uma norma, pois, se  $x = (1, 0)$ ,  $\|2x\| = 1 \neq 2\|x\|$ . Porém,  $\|\cdot\|_0$  satisfaz as propriedades demonstradas na Proposição 1, retirada de [1].

**Proposição 1.** Fixados  $x, y \in \mathbb{R}^n, \lambda \in \mathbb{R}$ , a ‘norma’ 0 satisfaz as seguintes propriedades:

(i)  $\|x\|_0 \geq 0$

(ii)  $\|x\|_0 = 0 \Leftrightarrow x = 0$

(iii) Se  $\lambda \neq 0$ ,  $\|\lambda x\|_0 = |\lambda| \|x\|_0$

---

<sup>1</sup>[https://commons.wikimedia.org/wiki/File:5-stip\\_LHB\\_09.jpg](https://commons.wikimedia.org/wiki/File:5-stip_LHB_09.jpg)

$$(iv) \|x + y\|_0 \leq \|x\|_0 + \|y\|_0$$

(v) 'norma' 0 não é uma norma.

*Demonstração.* Fixados  $x, y \in \mathbb{R}^n, \lambda \in \mathbb{R}$

$$(i) \|x\|_0 = \#\{i \in \{1, \dots, n\} : |x_i| > 0\} \geq 0$$

(ii)

$$\begin{aligned} \|x\|_0 = 0 &\Rightarrow \#\{i \in \{1, \dots, n\} : |x_i| > 0\} = 0 \\ &\Rightarrow \forall i \in \{1, \dots, n\}, |x_i| = 0 \\ &\Rightarrow x = 0 \end{aligned}$$

(iii) Se  $\lambda \neq 0$ ,

$$\begin{aligned} \|\lambda x\|_0 &= \#\{i \in \{1, \dots, n\} : |\lambda x_i| > 0\} \\ &= \#\{i \in \{1, \dots, n\} : |x_i| > 0\} \\ &= \|x\|_0 \end{aligned}$$

(iv)

$$\begin{aligned} \|x\|_0 + \|y\|_0 &= \#\{i : |x_i| > 0\} + \#\{i : |y_i| > 0\} \\ &\geq \#(\{i : |x_i| > 0\} \cup \{i : |y_i| > 0\}) \\ &= \#\{i : |x_i| > 0 \text{ ou } |y_i| > 0\} \\ &= \#\{i : |x_i| + |y_i| > 0\} \\ &\geq \#\{i : |x_i + y_i| > 0\} \\ &= \|x + y\|_0 \end{aligned}$$

---

<sup>2</sup>[https://commons.wikimedia.org/wiki/File:Along\\_The\\_Main\\_Ride\\_-\\_Beale\\_Arboretum\\_-\\_West\\_Lodge\\_Park\\_-\\_Hadley\\_Wood\\_-\\_Enfield\\_London\\_2.jpg](https://commons.wikimedia.org/wiki/File:Along_The_Main_Ride_-_Beale_Arboretum_-_West_Lodge_Park_-_Hadley_Wood_-_Enfield_London_2.jpg)



(a) imagem original



(b) compressão de 60%

Figura 3.2: A imagem em **(b)** é calculada ao descartar 40% das senoidais que compõem a imagem cujos coeficientes têm menor módulo. Adaptado de <sup>2</sup>.

(v) Sejam  $\lambda = 2, x = e_1$ , então

$$\|\lambda x\|_0 = \|x\|_0 = 1 \neq |\lambda| \|x\|_0$$

Portanto ‘norma’ 0 não é uma norma.

□

O problema que resolveremos para encontrar o vetor  $x$  mais esparsos amostrado como  $x = Ay$  será formulado do seguinte modo:

$$\min_{x \in \mathbb{R}^n} \|x\|_0 \text{ sujeito a } Ax = y \quad (P_0)$$

No entanto,  $(P_0)$  é NP-difícil [6]. Então tentamos resolver o seguinte problema:

$$\min_{x \in \mathbb{R}^n} \|x\|_1 \text{ sujeito a } Ax = y \quad (P_1)$$

A Figura 3.3 nos ajuda a entender a semelhança entre  $(P_0)$  e  $(P_1)$ . A figura mostra o problema de minimização de  $\|x\|_p$  sujeito a  $Ax = y$  para  $x \in \mathbb{R}^2$  e  $y \in \mathbb{R}$ . O ponto  $x$  minimiza a norma em um único ponto se a fronteira de uma bola fechada centrada em 0 toca a reta  $Ax = y$  apenas em  $x$ . Neste caso, as soluções calculadas para  $p = 1$  e  $p = 1/2$  são esparsas. Note que  $\|\cdot\|_{1/2}$  não é norma.

CS tem como objetivo estudar condições para garantir a equivalência entre os dois problemas, e condições para a unicidade de soluções de  $(P_0)$  e  $(P_1)$ . Descrevemos estas condições nas seções 3.2, 3.3 e 3.4. Todos resultados mostrados nessas seções podem ser encontrados em [1], exceto quando indicados explicitamente.

### 3.1 Unicidade de $(P_0)$

Se a matriz  $A$  fosse  $n \times n$  de posto máximo, a unicidade estaria garantida. Como  $m < n$ , definimos o conceito de *spark*, que verbalmente é uma fusão dos conceitos ‘esparsos’ e



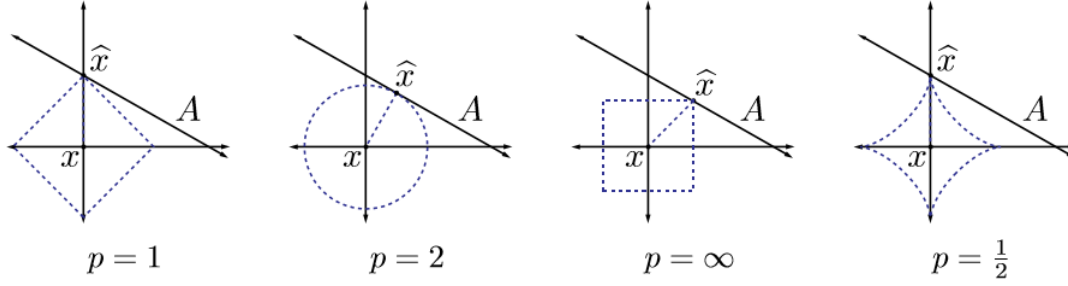


Figura 3.3: Pontos na reta  $Ax = y$  que minimizam a norma  $\|\cdot\|_p$  para diferentes valores de  $p$ . As soluções encontradas para  $p = 1$  e  $p = 1/2$  são esparsas. Note que  $\|\cdot\|_{\frac{1}{2}}$  não é norma. Reproduzido de [1].

‘posto’(rank) [9].

**Definição 1.** *Seja  $A$  uma matriz  $m \times n$ . O spark de  $A$ , denotado por  $\text{spark}(A)$ , é o menor número de colunas linearmente dependentes de  $A$ .*

O Teorema a seguir nos indica uma condição que  $A$  deve satisfazer para garantir a unicidade de  $(P_0)$

**Teorema 1.** *Seja  $A$  uma matriz  $m \times n$ . São equivalentes:*

(i)  $\text{spark}(A) > 2k$

(ii) *Para todo  $y \in \mathbb{R}^m$ , existe no máximo um  $x \in \Sigma_k$  tal que  $y = Ax$ .*

Portanto, é desejável que  $\text{spark}(A)$  seja alto pois, nesse caso, seria possível recuperar vetores  $x$  não tão esparsos.

Antes de demonstrar o teorema, enunciaremos uma definição e demonstraremos um lema.

**Definição 2.** *Dado  $k \in \{1, \dots, n\}$ ,  $\Sigma_k = \{x : \|x\|_0 \leq k\} = \{x : x \text{ é } k\text{-esparso}\}$*

**Lema 1.** *Dados  $u, v \in \Sigma_k, u - v \in \Sigma_{2k}$*

*Demonstração.* Sejam  $u, v \in \Sigma_k$ . Então

$$\begin{aligned}\|u - v\|_0 &\leq \|u\|_0 + \|v\|_0 \\ &= \|u\|_0 + \|v\|_0 \\ &\leq k + k = 2k \\ &\Rightarrow u - v \in \Sigma_{2k}\end{aligned}$$

□

Agora iremos demonstrar o Teorema 1.

*Demonstração.* Seja  $A$  uma matriz  $m \times n$ . Denote cada coluna  $j$  de  $A$  por  $A_j$ .

- $\neg(\text{i}) \Rightarrow \neg(\text{ii})$

Suponha que  $\text{spark}(A) \leq 2k$ . Então existe um conjunto  $J$  tal que  $\#J = 2k$  e existe uma bijeção  $\mu$  entre  $\{1, \dots, 2k\}$  tais que

$$\{A_{\mu(j)} : j \in \{1, \dots, 2k\}\} \text{ é LD}$$

Então existe  $c \in \mathbb{R}^{2k}, c \neq 0$  tal que

$$\sum_{j=1}^{2k} c_j A_{\mu(j)} = 0$$

Defina  $u, v \in \mathbb{R}^n$  tais que, para  $j \in \{1, \dots, n\}$

$$u_j = c_{\mu^{-1}(j)} \text{ se } j \in \mu^{-1}(\{1, \dots, k\}), 0 \text{ caso contrário}$$

$$v_j = -c_{\mu^{-1}(j)} \text{ se } j \in \mu^{-1}(\{k+1, \dots, 2k\}), 0 \text{ caso contrário}$$

Se  $j \in \mu^{-1}(\{1, \dots, 2k\}), u_j - v_j = c_j$ . Então  $u \neq v$

Então  $\|u\|_0, \|v\|_0 \leq k \Rightarrow u, v \in \Sigma_k$ .

Então

$$\begin{aligned}
\sum_{j=1}^{2k} c_j A_{\mu(j)} &= \sum_{j=1}^k c_j A_{\mu(j)} + \sum_{j=k+1}^{2k} c_j A_{\mu(j)} \\
&= \sum_{j=1}^k u_{\mu(j)} A_{\mu(j)} - \sum_{j=k+1}^{2k} v_{\mu(j)} A_{\mu(j)} \\
&= Au - Av = 0 \\
\Rightarrow Au &= Av
\end{aligned}$$

Então, definindo  $y = Au$ , existem  $u, v \in \mathbb{R}^n$  distintos tais que  $Au = Av$ .

- $\neg(\text{ii}) \Rightarrow \neg(\text{i})$

Suponha que existam  $y \in \mathbb{R}^m$ ,  $z, w \in \Sigma_k$  distintos tais que  $y = Az = Aw$ .

Seja  $x = z - w$ . Então  $Ax = 0$  e, pelo lema anterior[sic],  $x \in \Sigma_{2k}$ . Além disso,  $x \neq 0$ .

Defina  $K = \{i \in \{1, \dots, n\} : |x_i| > 0\}$  e  $\mu$  uma bijeção entre  $\{1, \dots, \|x\|_0\}$  e  $K$ . Então

$$Ax = \sum_{j=1}^{\|x\|_0} A_{\mu(j)} x_{\mu(j)} = 0$$

Como  $x \neq 0$ , existe algum  $j \in \{1, \dots, \|x\|_0\}$  tal que  $x_{\mu(j)} \neq 0$ . Logo,  $A$  possui  $\|x\|_0$  colunas linearmente dependentes. Portanto,  $\text{spark}(A) \leq \|x\|_0 \leq 2k$ .

□

## 3.2 Unicidade de $(P_1)$

Ao encontrar a solução única de  $(P_0)$ , encontramos o vetor desejado. No entanto,  $(P_0)$  é NP-difícil. Uma alternativa seria resolver o problema  $(P_1)$ .

Precisamos então estudar a unicidade de soluções para  $(P_1)$  e condições para a equivalência entre  $(P_0)$  e  $(P_1)$ , ou seja, em quais situações um vetor  $x$  é solução única de ambos  $(P_0)$  e  $(P_1)$ .

Definimos o conceito de *Null space property* para estudar a unicidade de  $(P_1)$ .

**Definição 3.** *Seja  $\Lambda \subset \{1, 2, \dots, n\}$ . Defina  $\Lambda^c = \{1, 2, \dots, n\} \setminus \Lambda$ . Dado  $x \in \mathbb{R}^n, x_\Lambda \in \mathbb{R}^n$ , onde, para  $i \in [n] = \{1, \dots, n\}$*

$$(x_\Lambda)_i = \begin{cases} x_i, & \text{se } i \in \Lambda \\ 0, & \text{caso contrário} \end{cases}$$

**Definição 4.** *Uma matriz  $A$  de tamanho  $m \times n$  satisfaz a null space property (NSP) relativa ao conjunto  $\Lambda \subset [n]$  se para todo  $h \in \ker(A) \setminus \{0\}$*

$$\|h_\Lambda\|_1 < \|h_{\Lambda^c}\|_1$$

**Definição 5.** *Dizemos que uma matriz de tamanho  $m \times n$  satisfaz a null space property (NSP) de ordem  $k$  se ela satisfaz a NSP para todo  $\Lambda \subset [n]$  com  $\#\Lambda \leq k$ .*

**Lema 2.** *Seja  $\Lambda \in [n]$ . Dado  $x \in \mathbb{R}^n$ ,*

$$\|x\|_1 = \|x_\Lambda\|_1 + \|x_{\Lambda^c}\|_1$$

*Demonstração.*

$$\|x\|_1 = \sum_{j=1}^n |x_j| = \sum_{j \in \Lambda} |x_j| + \sum_{j \in \Lambda^c} |x_j| = \|x_\Lambda\|_1 + \|x_{\Lambda^c}\|_1$$

□

**Teorema 2.** *Seja  $A$  uma matrix  $m \times n$ . Todo vetor  $x$  de suporte  $\Lambda$  tal que  $Ax = y$  é solução única de  $(P_1)$  se e somente se  $A$  satisfaz a NSP para  $\Lambda$ .*

*Demonstração.* Seja  $A$  uma matrix  $m \times n$ .

•  $(\Rightarrow)$

Seja  $h \in \ker(A) \setminus \{0\}$ . Então

$$0 = Ah = Ah_{\Lambda} + Ah_{\Lambda^c} \Rightarrow Ah_{\Lambda} = -Ah_{\Lambda^c} = A(-h_{\Lambda^c})$$

$h_{\Lambda}$  é solução única de  $(P_1)$ , pois tem suporte  $\Lambda$ . Como  $Ah_{\Lambda} = A(-h_{\Lambda^c})$ , temos que

$$\|h_{\Lambda}\|_1 < \| - h_{\Lambda^c}\|_1 = \|h_{\Lambda^c}\|_1$$

Portanto  $A$  satisfaz NSP para  $\Lambda$ .

•  $(\Leftarrow)$

Seja  $x$  uma solução de  $(P_1)$  com suporte  $\Lambda$  e  $z \in \mathbb{R}^n$  tal que  $Ax = Az$  e  $z \neq x$ . , Então  $x - z \in \ker(A) \setminus \{0\}$ . Definindo  $w = \frac{z}{3}$ , temos que

$$\begin{aligned} \|x\|_1 &\leq \|x - w\|_1 + \|w\|_1 \\ &= \|x_{\Lambda} - w_{\Lambda}\|_1 + \|x_{\Lambda^c} - w_{\Lambda^c}\|_1 + \|w\|_1 \\ &= \|x_{\Lambda} - w_{\Lambda}\|_1 + \|w_{\Lambda^c}\|_1 + \|w\|_1 \\ &< \|x_{\Lambda^c} - w_{\Lambda^c}\|_1 + \|w_{\Lambda^c}\|_1 + \|w\|_1 \\ &= \|w_{\Lambda^c}\|_1 + \|w_{\Lambda^c}\|_1 + \|w\|_1 \\ &= 2\|w_{\Lambda^c}\|_1 + \|w\|_1 \\ &\leq 2\|w\|_1 + \|w\|_1 \\ &= 3\|w\|_1 = \|z\|_1 \end{aligned}$$

Portanto  $x$  é solução única de  $(P_1)$ .

□

A noção de NSP nos diz que a norma 1 dos vetores do núcleo de  $A$  não estão concentradas num número pequeno de índices. O Corolário 1 estabelece uma relação entre NSP e a unicidade de soluções de  $(P_1)$ .

**Colorário 1.** *Seja  $A$  uma matrix  $m \times n$ . Todo vetor  $x$   $k$ -sparso tal que  $Ax = y$  é solução única de  $(P_1)$  se e somente se  $A$  satisfaz a NSP de ordem  $k$ .*

*Demonstração.* Seja  $A$  uma matrix  $m \times n$ .

•  $(\Rightarrow)$

Seja  $\Lambda \subset [n]$  tal que  $\#\Lambda = k$ . Seja  $x$  um vetor esparso de suporte em  $\Lambda$  tal que  $x$  é solução única de  $(P_1)$ . Então, [pelo teorema anterior],  $A$  satisfaz NSP para  $\Lambda$ . Como  $\Lambda$  é arbitrário,  $A$  satisfaz NSP de ordem  $k$ .

•  $(\Leftarrow)$

Seja  $x$  um vetor  $k$ -sparso tal que  $Ax = y$ . Então existe  $\Lambda \subset [n]$  tal que  $\#\Lambda = k$  e  $x_\Lambda = x$ . Como  $A$  satisfaz NSP para  $\Lambda$ ,  $x$  é solução única.

□

### 3.3 Equivalência entre $(P_0)$ e $(P_1)$

Para estudar a equivalência entre  $(P_0)$  e  $(P_1)$ , por enquanto, assumimos que as colunas  $a_i$  de  $A$  possuem norma  $\|a_i\|_2 = 1$ . Definimos também o conceito de coerência.

**Definição 6.** *Seja  $A$  uma matriz  $m \times n$  cujas colunas  $a_i$  possuem  $\|a_i\|_2 = 1$ . definimos a coerência (ou mutual coherence) de  $A$ , denotada por  $\mu(A)$ , como*

$$\mu(A) = \max_{i \neq j} |\langle a_i, a_j \rangle|$$

O seguinte teorema, que garante a equivalência entre  $(P_0)$  e  $(P_1)$  foi então estudado.

**Teorema 3.** *Seja  $x$  uma solução de  $(P_0)$  tal que*

$$\|x\|_0 < \frac{1}{2} \left( 1 + \frac{1}{\mu(A)} \right)$$

*Então  $x$  é solução única de  $(P_0)$  e  $(P_1)$ .*

Alguns resultados serão apresentados antes de demonstrar o Teorema 3.

**Definição 7.** *Seja  $x \in \mathbb{R}^n$ . Definimos  $|x|$  como um vetor  $|x| \in \mathbb{R}^n$  tal que para cada  $i \in [n]$ ,  $|x|_i = |x_i|$ .*

**Definição 8.** *Dados  $x, y \in \mathbb{R}^n$ ,  $x \leq y$  se  $\forall j \in [n], x_j \leq y_j$ .*

**Lema 3.** *Fixado  $x \in \mathbb{R}^n$ ,  $x \leq |x|$ .*

*Demonstração.*  $\forall i \in [n], x_i \leq |x_i| = |x|_i$  □

**Lema 4.** *Dado  $x \in \mathbb{R}^n$ ,  $\|x\|_1 = \||x|\|_1$*

*Demonstração.* Seja  $x \in \mathbb{R}^n$ ,

$$\|x\|_1 = \sum_{i=1}^n |x_i| = \sum_{i=1}^n |(|x|_i)| = \sum_{i=1}^n |(|x|)_i| = \||x|\|_1$$

□

**Definição 9.** *Seja  $A$  uma matriz  $m \times n$ .  $|A|$  é uma matriz  $m \times n$  cujas entradas  $|A|_{ij} = |A_{ij}|$ , para todo  $i \in [m], j \in [n]$ .*

**Lema 5.** *Sejam  $A$  uma matriz  $m \times n$  e  $x \in \mathbb{R}^n$ . Então  $|Ax| \leq |A||x|$ .*

*Demonstração.* Dado  $i \in [m]$ ,

$$|Ax|_i = |(Ax)_i| = \left| \sum_{k=1}^n A_{ik}x_k \right| \leq \sum_{k=1}^n |A_{ik}x_k| = \sum_{k=1}^n |A_{ik}||x_k| = (|A||x|)_i$$

□

**Teorema 4.** (Geršgorin) [8] *Seja  $A$  uma matriz  $n \times n$ , de elementos  $a_{ij}, 1 \leq i, j \leq n$ . Então, para cada autovalor  $\lambda$  de  $A$ ,*

$$\lambda \in \bigcup_{i=1}^n \overline{B}(a_{ii}, r_i)$$

onde

$$r_i = \sum_{j \neq i} |a_{ij}|$$

*Demonstração.* Sejam  $\lambda \in \mathbb{C}$  um autovalor de  $A$ ,  $v \in \mathbb{C}^n$  tal que  $Av = \lambda v$ . Escolha  $i \in [n]$  de forma que  $|v_j| \leq |v_i|$ , para todo  $j \in [n]$ . Então  $v_i \neq 0$  e

$$\begin{aligned} \lambda v_i &= (Av)_i = \sum_{j=1}^n a_{ij} v_j = a_{ii} v_i + \sum_{j \neq i} a_{ij} v_j \\ \Rightarrow (\lambda - a_{ii}) v_i &= \sum_{j \neq i} a_{ij} v_j \\ \Rightarrow |\lambda - a_{ii}| |v_i| &\leq \sum_{j \neq i} |a_{ij}| |v_j| \\ \Rightarrow |\lambda - a_{ii}| &\leq \sum_{j \neq i} |a_{ij}| \frac{|v_j|}{|v_i|} \leq \sum_{j \neq i} |a_{ij}| = r_i \\ \Rightarrow \lambda &\in \overline{B}(a_{ii}, r_i) \end{aligned}$$

□

**Lema 6.** Para toda matriz real  $A$  de tamanho  $m \times n$ , com colunas  $a_i$  tais que  $\|a_i\|_2 = 1$ ,

$$\text{spark}(A) \geq 1 + \frac{1}{\mu(A)}$$

*Demonstração.* Seja  $G = A^T A$ . Para todo  $i \in [m]$ ,  $G_{ii} = 1$ . Para  $i \neq j$ ,  $G_{ij} = \langle a_i, a_j \rangle \leq \|a_i\|_2 \|a_j\|_2 = 1$ . Logo,  $\mu(A) \leq 1$ . Então, para todo  $\lambda$  autovalor de  $G$ ,

$$|\lambda - 1| \leq \sum_{j \neq i} |G_{ij}| \leq (m - 1)$$

□

Agora demonstraremos o Teorema 3.

*Demonstração.* Assumindo que  $x$  é solução única de  $(P_0)$



Seja  $z \in \mathbb{R}^n$  tal que  $z \neq x$ . Então  $z = x + h$  para algum  $h \in \ker(A) \setminus 0$ . Seja  $\Lambda$  o suporte de  $x$ . Temos que

$$\begin{aligned}
\|z\|_1 - \|x\|_1 &= \sum_{k=1}^n |x_k + h_k| - \sum_{k=1}^n |x_k| \\
&= \sum_{k \in \Lambda} |x_k + h_k| + \sum_{k \in \Lambda^c} |x_k + h_k| - \sum_{k \in \Lambda} |x_k| \\
&= \sum_{k \in \Lambda} |x_k + h_k| + \sum_{k \in \Lambda^c} |h_k| - \sum_{k \in \Lambda} |x_k| \\
&= \sum_{k \in \Lambda^c} |h_k| + \sum_{k \in \Lambda} (|x_k + h_k| - |x_k|) \\
&\geq \sum_{k \in \Lambda^c} |h_k| - \sum_{k \in \Lambda} |h_k| \\
&= \|h_{\Lambda^c}\|_1 - \|h_{\Lambda}\|_1 \\
&= \|h_{\Lambda^c}\|_1 - \|h_{\Lambda}\|_1 + (\|h_{\Lambda}\|_1 - \|h_{\Lambda}\|_1) \\
&= \|h\|_1 - 2\|h_{\Lambda^c}\|_1
\end{aligned}$$

Como  $\forall h, Ah = 0 \Rightarrow A^T Ah = 0$ , temos que

$$\inf_{\substack{Ah=0 \\ \|h\|_1=1}} 1 - 2\|h_{\Lambda}\|_1 \geq \inf_{\substack{A^T Ah=0 \\ \|h\|_1=1}} 1 - 2\|h_{\Lambda}\|_1$$

Como  $\|h\|_1 = \| |h| \|_1$  e  $|h_{\Lambda}| = |h|_{\Lambda}$ , temos que

$$\inf_{\substack{A^T Ah=0 \\ \|h\|_1=1}} 1 - 2\|h_{\Lambda}\|_1 \geq \inf_{\substack{A^T Az=0 \\ \|z\|_1=1 \\ z \geq 0}} 1 - 2\|z_{\Lambda}\|_1$$

Definindo  $G = A^T A$ , cada entrada  $G_{ij} = \langle a_i, a_j \rangle$ .

$$\inf_{\substack{A^T Az=0 \\ \|z\|_1=1 \\ z \geq 0}} 1 - 2\|z_{\Lambda}\|_1 = \inf_{\substack{Gz=0 \\ \|z\|_1=1 \\ z \geq 0}} 1 - 2\|z_{\Lambda}\|_1$$

Como  $z \geq 0$  e  $Gz = 0$ ,

$$z = |-z| \Rightarrow z = |Gz - z| = |(G - I)z| \leq |G - I|z$$

Todos as entradas de  $G - I$  maiores os iguais a zero e os elementos da diagonal são nulos, então

$$|G - I|_{ij} \leq \max_{i \neq j} G_{ij} = \max_{i \neq j} \langle a_i, a_j \rangle = \mu(A)$$

Então

$$|G - I| \leq \mu(A)(\mathbb{1} - I)$$

Logo

$$z \leq \mu(A)(\mathbb{1} - I)z$$

Então, dado  $i \in \Lambda$ ,

$$\begin{aligned} z_i &\leq \mu(A) \left( \sum_{j=1}^n z_j - z_i \right) \\ &\leq \mu(A)(\|z\|_1 - z_i) \\ &= \mu(A)(1 - z_i) \\ \Rightarrow z_i + z_i \mu &\leq \mu(A) \\ \Rightarrow z_i &\leq \frac{\mu(A)}{1 + \mu(A)} \end{aligned}$$

Então

$$\|z_\Lambda\|_1 = \sum_{i \in \Lambda} z_i \leq \sum_{i \in \Lambda} \frac{\mu(A)}{1 + \mu(A)} = \|x\|_0 \frac{\mu(A)}{1 + \mu(A)}$$

Então

$$\begin{aligned}
\inf_{\substack{Gz=0 \\ \|z\|_1=1 \\ z \geq 0}} 1 - 2\|z_\Lambda\|_1 &\geq \inf_{\substack{Gz=0 \\ \|z\|_1=1 \\ z \geq 0}} 1 - 2\|x\|_0 \frac{\mu(A)}{1 + \mu(A)} \\
&> \inf_{\substack{Gz=0 \\ \|z\|_1=1 \\ z \geq 0}} 1 - 2\frac{1}{2} \left(1 + \frac{1}{\mu(A)}\right) \frac{\mu(A)}{1 + \mu(A)} \\
&= 1 - 2\frac{1}{2} \left(1 + \frac{1}{\mu(A)}\right) \frac{\mu(A)}{1 + \mu(A)} \\
&= 1 - \frac{\mu(A) + 1}{\mu(A)} \frac{\mu(A)}{1 + \mu(A)} \\
&= 1 - 1 = 0
\end{aligned}$$

□

### 3.4 Matrizes que satisfazem as condições de CS

Como vimos anteriormente, a matriz  $A$  deve satisfazer algumas condições para garantir a equivalência entre  $(P_0)$  e  $(P_1)$ , que chamaremos de condições de CS. Em vez de construir uma matriz  $A$  de forma determinística, o que talvez não seja fácil, podemos construir uma matriz aleatória que satisfaz as condições de CS.

Uma matriz  $A$  cujas entradas são amostradas através de uma distribuição normal satisfazem as condições de CS com alta probabilidade, como mostra o Teorema 5, que foi extraído de [16].

**Teorema 5.** [16] *Sejam  $m < n$ ,  $\Omega \subset \mathbb{R}^n$  e  $A \in \mathbb{R}^{m \times n}$  tais que apenas uma das afirmações é verdadeira:*

1. *os elementos  $a_{ij}$  de  $A$  são iid e  $a_{ij} \sim \mathcal{N}(0, 1)$*
2.  *$A$  é uma matriz de posto  $m$  com  $BA^T = 0$  para uma matriz  $B \in \mathbb{R}^{(n-m) \times n}$  cujas entradas são iid  $\mathcal{N}(0, 1)$ .*

Então com probabilidade maior que  $1 - e^{-c_0(n-m)}$ ,  $\bar{x}$  é solução única de  $(P_0)$  e  $(P_1)$  se  $\bar{x}$  satisfaz

$$\|\bar{x}\|_0 < \frac{c_1^2}{4} \frac{m}{1 + \log(n/m)}$$

onde  $c_0, c_1 > 0$  são constantes independentes das dimensões  $m$  e  $n$ .

# Capítulo 4

## O algoritmo de homotopia

O algoritmo de Homotopia [2] é um algoritmo com o objetivo de resolver  $(P_1)$  minimizando funções do tipo

$$J_{\lambda_n}(x) = \frac{1}{2}\|Ax - y\|_2^2 + \lambda_n\|x\|_1$$

para uma sequência decrescente  $(\lambda_n)$  de números positivos.

Podemos encontrar o mínimo de uma função diferencial  $f : \mathbb{R}^n \rightarrow \mathbb{R}$  procurando os pontos  $x$  no domínio em que  $\nabla f(x) = 0$ . No nosso caso, a norma 1 não é diferenciável. Então será definida uma generalização para a diferencial, que será usada no algoritmo, chamado de Homotopia.

**Definição 10.** *Seja  $f : \mathbb{R}^n \rightarrow \mathbb{R}$ . A subdiferencial de  $f$  em  $x \in \mathbb{R}$  é dada por*

$$\partial f(x) = \{v \in \mathbb{R}^N : f(y) - f(x) \geq \langle v, y - x \rangle, \forall y \in \mathbb{R}^N\}$$

Não é difícil ver que o operador  $\partial$  é linear. A subdiferencial coincide com a diferencial de uma função, quando ela for convexa e diferenciável, como mostra a seguinte proposição:

**Proposição 2.** *Se  $f : \mathbb{R}^n \rightarrow \mathbb{R}$  for convexa e diferenciável, então para todo  $x \in \mathbb{R}^n$ ,  $\partial f(x) = \{\nabla f(x)\}$ .*

Um ponto  $x_0$  é ponto de mínimo de uma função  $f$  se, e somente se,  $0 \in \partial f$ , pois

$$\forall x \in \mathbb{R}^n, f(x) - f(x_0) \geq 0 = \langle 0, x - x_0 \rangle$$

Como a norma 1 é convexa, sua subdiferencial em um ponto  $x$  é um vetor  $v$  tal que para cada  $i = 1, \dots, n$ ,

$$v_i = \begin{cases} \text{sgn}(v_i) & \text{se } v_i \neq 0 \\ \{1, -1\} & \text{se } v_i = 0 \end{cases}$$

O algoritmo de Homotopia [5] [2] minimiza a função em um número finito de passos. Fixado um  $\lambda$  positivo, considere a seguinte função :

$$J_\lambda(x) = \frac{1}{2} \|Ax - y\|_2^2 + \lambda \|x\|_1$$

e seu respectivo minimizador  $x_\lambda$ . Quando  $\lambda$  é grande,  $x_\lambda = 0$ . O ponto  $\tilde{x}$ , solução de  $(P_1)$  é solução de algum  $J_{\tilde{\lambda}}$ . A curva  $\lambda \mapsto x_\lambda$  é linear por partes, então é possível encontrar a solução de  $(P_1)$  com um número finito de passos.

A subdiferencial de  $J_\lambda$  é

$$\partial J_\lambda(x) = A^T(Ax - y) + \lambda \partial \|x\|_1$$

Se  $x = x_\lambda$  então  $0 \in \partial J_\lambda(x)$  é equivalente a

$$\begin{cases} (A^*(Ax - y))_i = \lambda \text{sgn}(x_i), & \text{se } x_i \neq 0 \\ |A^*(Ax - y)|_i \leq \lambda, & \text{se } x_i = 0 \end{cases} \quad (4.1)$$

para  $i \in \{1, \dots, n\}$ , onde  $\text{sgn}(x_i) = 1$  se  $x_i \geq 0$  e 0 caso contrário.

Escrevendo  $c = A^T(Ax - y)$  e denotando  $I$  como o suporte de  $x$ , então (4.1) equivale a

$$\begin{cases} c(I) = \lambda \text{sgn}(x_\lambda(I)) \\ |c(I^c)| \leq \lambda \end{cases} \quad (4.2)$$

O algoritmo de Homotopia procura os vértices da curva  $\lambda \mapsto x_\lambda$ . Começando com  $x_0 = 0$ , em uma iteração  $l$ ,  $\lambda_l = \|c(I)\|_\infty$ , com  $I$  denotando o suporte de  $x_l$ . Depois, o algoritmo calcula uma direção  $d_l$ , onde

$$\begin{cases} A_I^T A_I d_l(I) = \text{sgn}(c_l(I)) \\ d_l(I^c) = 0 \end{cases} \quad (4.3)$$

$A_I$  denota a matriz  $[A_{i_1}, \dots, A_{i_r}]$ , onde cada  $i_r \in I$  e cada  $A_{i_r}$  é uma  $i_r$ -ésima coluna de  $A$ . Da mesma maneira,  $d_l(I)$  é o vetor calculado após remover todos os elementos de  $d_l(i)$  tais que  $i \notin I$ .

A magnitude  $\gamma_l$  do passo  $d_l$  é calculada como o menor valor em que a equação (4.2) não seja mais válida, ou seja, para  $x_{l+1} = x_l + \gamma_l d_l$ , teremos que pelo menos uma das seguintes condições será satisfeita:

(i) para algum  $i \in I$ ,  $|(c_l)_i| > \lambda$ . Isso ocorre para  $\gamma_l = \gamma_l^+$ , onde

$$\gamma_l^+ = \min_{i \in I^c} \left\{ \frac{\lambda_l - c_l(i)}{1 - a_i^T v_l}, \frac{\lambda_l + c_l(i)}{1 + a_i^T v_l} \right\}$$

Onde  $v_l = A_I d_l(I)$ .

(ii) para algum  $i \in I$ ,  $(x_{l+1})_i = 0$ , o que equivale a  $\gamma = \gamma^-$ , onde

$$\gamma_l^- = \min_{i \in I} \left\{ \frac{-x_l(i)}{d_l(i)} \right\}$$

Então calculamos  $\gamma = \min\{\gamma_l^-, \gamma_l^+\}$ . O algoritmo termina quando  $c_l = 0$  ou quando  $\lambda_l = \|c_l\|_\infty \leq \lambda_{l-1} = \|c_{l-1}\|_\infty$

O Algoritmo 22 mostra o funcionamento passo a passo.

---

**Algoritmo 1:** Homotopia

---

**Entrada:**  $A$  matriz  $m \times n$ ,  $y \in \mathbb{R}^m$

**Saída:**  $x \in \mathbb{R}^n$  esparso com  $Ax = y$

```
1 início
2    $\lambda_0 = -1$ 
3    $l = 1$ 
4    $x_l = 0, c_l = -A^T y, \lambda_l = \|c_l\|_\infty$ 
5    $I = \text{supp}(c_l)$ 
6   enquanto  $\lambda_l \geq 0$  e  $\lambda_l > \lambda_{l-1}$  faça
7        $d_l = (A_I^T A_I)^\dagger \text{sgn}(c_l(I)) = A_I^\dagger (A_I^\dagger)^T \text{sgn}(c_l(I))$ 
8        $v_l = A_I d_l(I)$ 
9       Calcular  $\gamma_l^+$  como o valor mínimo da família  $\{\frac{\lambda_l - c_l(i)}{1 - a_i^T v_l}\}_{i \in I^c}$  e  $i^+$  o índice onde
       esse valor é atingido.
10      Calcular  $\gamma_l^-$  como o valor mínimo da família  $\{\frac{-x_l(i)}{d_l(i)}\}_{i \in I}$  e  $i^-$  o índice onde esse
       valor é atingido.
11       $\lambda_l = \min\{\lambda_l^-, \lambda_l^+\}$ 
12       $i = \min\{i^-, i^+\}$ 
13      se  $i \in I$  então
14          | Adicionar  $i$  a  $I$ 
15      senão
16          | Remover  $i$  de  $I$ 
17      fim
18       $x_{l+1} = x_l + \lambda_l d_l$ 
19       $l = l + 1$ 
20       $c_l = A^T(Ax_l - y)$ 
21 fim
22 fim
```

---

Como o programa calcula a pseudoinversa de  $A_I$  em cada iteração e entre uma iteração e a



seguinte  $A_I$  tem uma coluna a mais ou a menos que a  $A_I$  anterior, fizemos uma adaptação do algoritmo recursivo de pseudoinversa [7] para calcular de maneira mais eficiente  $A_I^\dagger$  quando adicionamos um elemento a  $I$ , a pseudoinversa anterior.

Quando removemos um elemento de  $I$ , calculamos  $A_I^\dagger$  da maneira usual.

## 4.1 Exemplo

Usamos o algoritmo de homotopia para recuperar uma imagem usando diferentes taxas de compressão.

Fixada uma taxa de compressão  $r$ , calculamos  $m = (1 - r) \cdot n$ . Dada uma imagem, calculamos um frame com a transformada do cosseno da imagem e depois obtemos um vetor  $x$  empilhando todas as linhas do frame. Em seguida, calculamos  $y = Ax$ , onde os elementos de  $A$  são iid  $\mathcal{N}(0, 1)$ . Depois encontramos uma aproximação  $\tilde{x}$  para  $x$  usando o algoritmo de homotopia.

A Figure 4.1 mostra uma imagem recuperada pelo algoritmo de homotopia usando uma taxa de compressão de 20%. Note que o esse algoritmo apresenta melhor resultado que o MMQ com a mesma taxa de compressão.

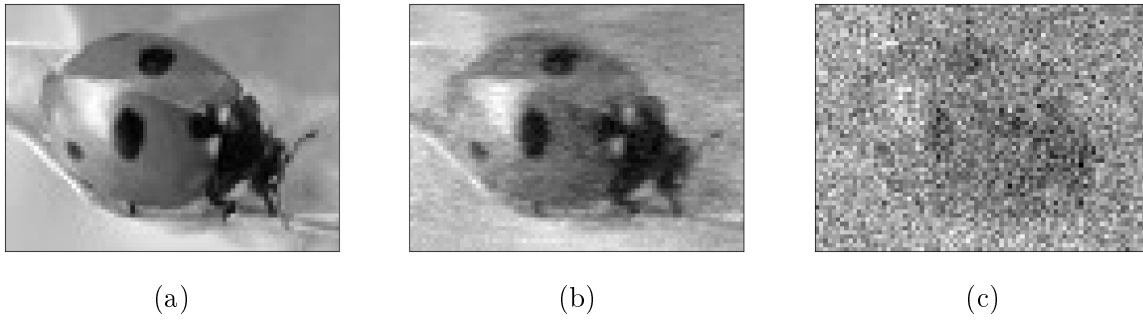


Figura 4.1: **(a)** Imagem original; **(b)** Imagem recuperada com o algoritmo de homotopia usando taxa de compressão de  $r = 20\%$ ; **(c)** Imagem recuperada usando o método de mínimos quadrados a  $r = 20\%$ . Adaptado de <sup>1</sup>

---

<sup>1</sup>[https://commons.wikimedia.org/wiki/File:5-stip\\_LHB\\_09.jpg](https://commons.wikimedia.org/wiki/File:5-stip_LHB_09.jpg)

# Capítulo 5

## O modelo *cross-and-bouquet*

O modelo *cross-and-bouquet* usa CS para classificar imagens, ou seja, dada uma amostra  $y$  e uma família de imagens  $(\Phi_i)$ , encontrar qual  $\Phi_i$  é mais próxima de  $y$ .

Podemos interpretar o problema  $(P_0)$  da seguinte forma: dado um vetor  $y \in \mathbb{R}^m$ , quais colunas  $a_i \in \mathbb{R}^m$  de  $A$  melhor representam  $y$ ? Se  $y$  puder ser escrito como  $y = Ax$  com  $x$  esparso, podemos assumir que a coluna  $a_i$  mais próxima de  $y$  é aquela onde  $x_i$  possui maior valor absoluto, ou seja,

$$i = \operatorname{argmax}_{j=1,\dots,n} |x_j|$$

Como vimos, CS garante que conseguimos encontrar  $x$  esparso resolvendo  $(P_1)$  apenas quando  $A$  é incoerente. Em aplicações, como em processamento de imagens, os vetores  $a_i$  não são “muito diferentes” entre si, por isso não podemos assumir que  $A$  é incoerente. Então formularemos o problema de uma forma um pouco diferente:

Dado  $y \in \mathbb{R}^m$  e  $A$  uma matriz  $m \gg n$  (ou seja, o número de amostras é pequeno se comparado à dimensão de cada amostra),

$$\min_{x \in \mathbb{R}^n} \|x\|_1 + \|e\|_1 \text{ sujeito a } y = Ax + e \quad (\tilde{P}_1)$$

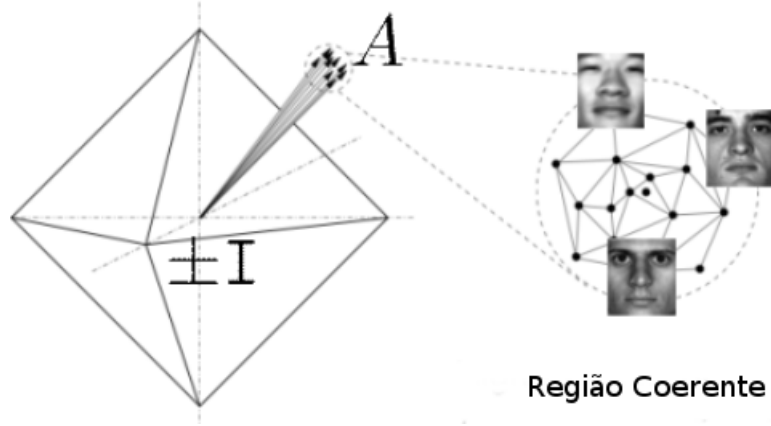


Figura 5.1: Modelo *cross-and-bouquet*. Imagem representando as colunas de  $[A|I]$  onde cada coluna de  $A$  pode representar uma imagem, por exemplo. Adaptado de [15].

o que é equivalente a encontrar um vetor  $c = \begin{bmatrix} x \\ e \end{bmatrix}$ , com  $x \in \mathbb{R}^n$  e  $e \in \mathbb{R}^m$ , onde  $c$  resolve o problema

$$\min_{c \in \mathbb{R}^{n+m}} \|c\|_1 \text{ sujeito a } y = \begin{bmatrix} A & I \end{bmatrix} c \quad (\tilde{P}_1)$$

onde  $I$  é a matriz identidade  $m \times m$ .

Como  $I$  é incoerente, pois é ortonormal, e possui muito mais colunas que  $A$ , é razoável supor que  $\begin{bmatrix} A & I \end{bmatrix}$  também seja incoerente e, neste caso, a solução  $c$  de  $(\tilde{P}_1)$  seria esparsa.

Identificamos a amostra  $y$  com o vetor  $a_i$  onde

$$i = \operatorname{argmax}_{i=1, \dots, n} |c_i|$$

Essa é a ideia do modelo *cross-and-bouquet* [14]. O modelo tem esse nome porque as colunas de  $I$  são ortonormais, lembrando uma cruz e as colunas de  $A$  estão próximas, lembrando um buquê, como mostra a Figura 5.1.

# Capítulo 6

## Desenvolvimento de um rastreador de olhar usando CS

Desenvolvemos um programa para estimar o olhar do usuário, ou seja, estimar qual coordenada  $(x, y)$  na tela o usuário está olhando. Para isso, usamos câmera da Pupil Labs para coletar imagens do olho direito. Essa câmera fica acoplada à cabeça e registra imagens em infravermelho <sup>1</sup>. O programa é dividido em duas etapas principais:

- **Calibração:** exibimos alvos em posições específicas na tela e coletamos uma imagem do olho por alvo, assumindo que o usuário está olhando para o alvo.
- **Rastreamento:** para cada frame  $f$ , encontramos as amostras mais parecidas com  $f$ . Estimamos a posição do olhar como a média das posições da na tela correspondentes às amostras selecionadas.

Essas etapas são descritas mais detalhadamente abaixo.

---

<sup>1</sup><https://github.com/pupil-labs/pupil/wiki/Environment>

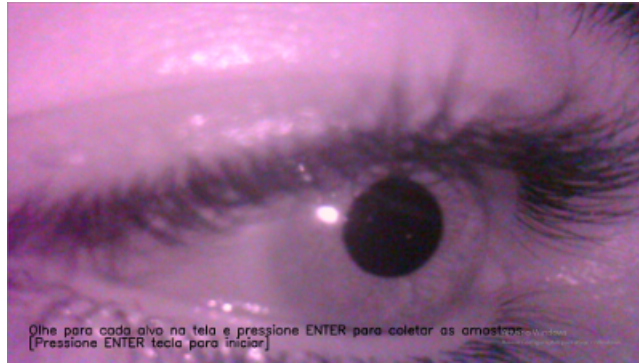


Figura 6.1: O programa exibe uma imagem do olho antes da coleta, para o usuário ajustar a câmera de forma que o olho inteiro seja registrado.

## 6.1 Coleta de amostras

Fizemos o experimento com seis pessoas para avaliar o desempenho do rastreador de olhar. Durante o experimento o participante ficou sentado a  $56,5\text{cm}$  de distância do monitor, com o olho direito alinhado com o centro da tela. Para evitar movimentos da cabeça durante a coleta das imagens, a pessoa ficou com o queixo apoiado sobre o punho e cotovelo correspondente apoiado na mesa. Registramos imagens do olho direito usando uma câmera do Pupil Labs, acoplada à cabeça, que grava imagens em infravermelho.

A câmera do Pupil Labs estava ligada a um suporte de formato parecido com uma armação de óculos. Por esse motivo, quem usa óculos teve que retirar os óculos para participar do experimento.

Antes de iniciar a coleta, mostramos a imagem da câmera do olho para a pessoa ajustar a câmera de forma que o olho inteiro esteja visível na imagem, como mostra a Figura 6.1.

Foram exibidos aleatoriamente 49 pontos dispostos numa grade  $7 \times 7$  de pontos igualmente espaçados, onde a distância entre cada lado da grade e o canto correspondente da tela corresponde a um grau do campo visual.

Mostramos cada alvo individualmente durante dois segundos e o usuário deveria olhar para o ponto, como mostra a Figura 6.2. Mostramos os pontos aleatoriamente para evitar

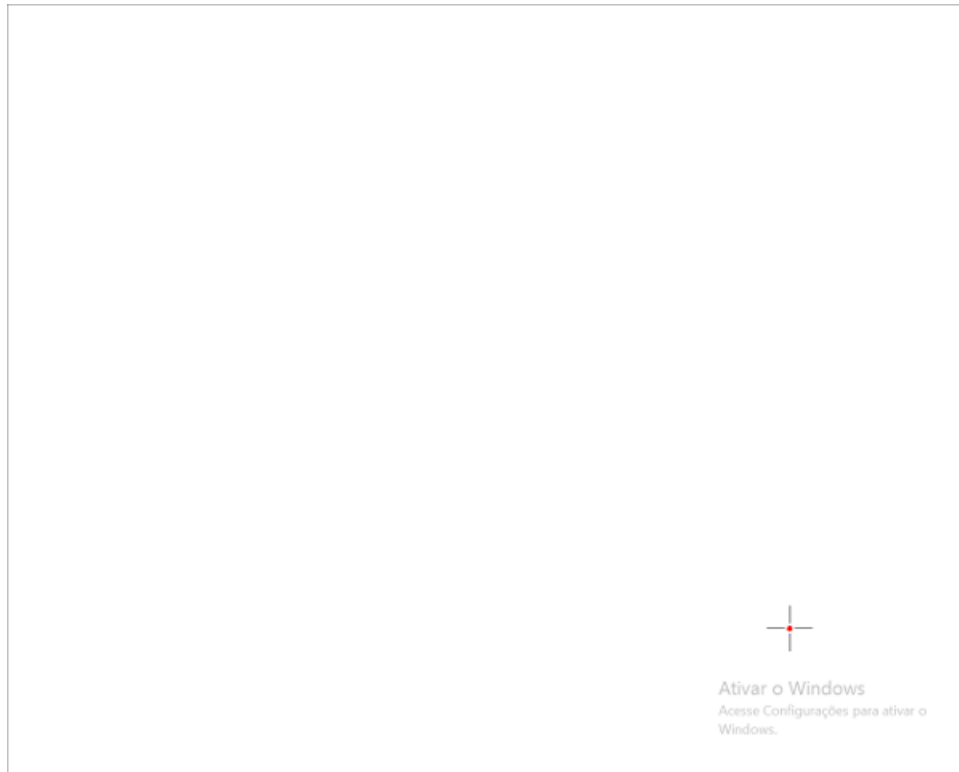


Figura 6.2: Durante uma coleta, a pessoa deve olhar para os alvos que são exibidos em posições diferentes em uma grade  $7 \times 7$ .

aprendizado do usuário, ou seja, evitar que o usuário olhe para o ponto correspondendo à amostra seguinte antes de terminar a coleta da amostra atual. Apesar de mostrar cada ponto durante dois segundos, coletamos imagens apenas após o primeiro segundo porque durante os primeiros instantes ocorre a sacada, ou seja, o movimento do olhar até a região de interesse, e queremos registrar apenas imagens correspondendo à fixação, ou seja, quando a pessoa está realmente olhando para o ponto.

Usamos um monitor de largura  $37,3cm$  e altura de  $32cm$ , com resolução  $1280 \times 1024$ . A coleta durou menos de 10 minutos por participante.

## 6.2 Rastreamento de olhar

Após terminar a coleta das imagens, estimamos a posição do olhar para imagens correspondendo a todos os pontos da grade  $7 \times 7$ . Como todas imagens registradas correspondem aos pontos da grade, já sabemos a posição do olhar em cada um deles, então usaremos a posição exata e a posição estimada para calcular o erro como a distância euclidiana entre elas. Usaremos a posição de cada alvo em graus horizontal e vertical, e não em *pixels* na tela.

Para cada posição na grade, consideramos a primeira imagem coletada como a amostra correspondente àquela posição na tela, e estimaremos a posição do olhar para cinco das outras imagens, selecionadas aleatoriamente, comparando esta com todas as amostras.

Pelo modelo *cross-and-bouquet*, para cada vetor correspondendo à imagem  $y$ , calculamos um vetor  $x = (c, e)$  tal que  $y = Ac + e$ . Podemos assumir que as amostras mais parecidas com  $y$ , são as colunas  $a_i$  de  $A$  onde  $c_i$  possuem maior valor em módulo. Assumindo isso, estimamos a posição do olhar para a imagem  $y$  como a média ponderada das posições do olhar das três imagens mais parecidas com  $y$ , onde  $|c_i|$  são os pesos.

Antes de comparar as imagens usando o modelo *cross-and-bouquet* reduzimos as imagens aplicando a pirâmide gaussiana 3, 4 ou 5 vezes, resultando em imagens  $60 \times 80$ ,  $40 \times 30$  e  $20 \times 15$ , respectivamente. Fazemos isso para evitar erros de memória durante a execução do programa.

A Tabela 6.1 mostra a média dos erros (conhecida como acurácia) e o desvio padrão (a precisão), considerando diferentes tamanhos de imagens. Podemos observar que o erro é menor se usamos imagens maiores para estimar o olhar. Para imagens  $60 \times 80$ , obtemos uma acurácia de 1,053 e precisão de 2,279, semelhantes aos resultados obtidos pelo rastreador Tobii EyeX, que tem acurácia de 1,42 e precisão de 1,70, segundo [11].

A Figura 6.3 mostra a distribuição dos erros nos pontos da grade para diferentes tamanhos de imagem, após reduzir as imagens originais aplicando a pirâmide gaussiana 3 vezes (resultando em uma imagem  $60 \times 80$ , 4 vezes (resultando em imagens  $30 \times 40$  e 5 vezes



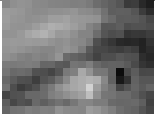
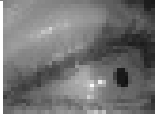
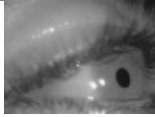
	dimensão das imagens		
dimensões das imagens	$15 \times 20$	$30 \times 40$	$60 \times 80$
Erro	$1,471 \pm 2,398$	$1,113 \pm 2,291$	$1,053 \pm 2,279$
Imagem			

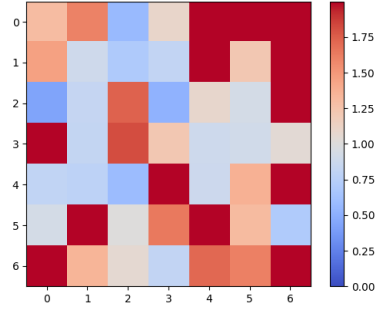
Tabela 6.1: Erros na estimação do olhar para diferentes tamanhos de amostras. Cada coluna representa o tamanho das imagens usadas, e cada célula representa o erro médio  $\pm$  o desvio padrão em graus. A última linha mostra exemplos de imagens usadas nas respectivas dimensões.

(resultando em imagens  $15 \times 20$ ). Podemos notar na última imagem que os erros são maiores nos cantos da grade.

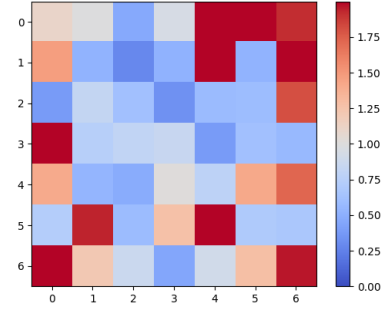
A Figura 6.4 mostra a acurácia para cada participante. Observando os dados e as imagens registradas durante o experimento, notamos que se o participante piscar durante a coleta das amostras, o resultado será pior. O participante 2, por exemplo, piscou algumas vezes durante a coleta de algumas amostras (que eram usadas para estimar o olhar nas outras imagens), acreditamos que por isso o resultado foi pior que os dos outros participantes.

Uma limitação do experimento foi que todas as imagens correspondentes a um ponto na grade foram coletadas durante um intervalo de um segundo, então elas talvez sejam mais parecidas entre si do que seriam se fossem registradas em momentos diferentes.

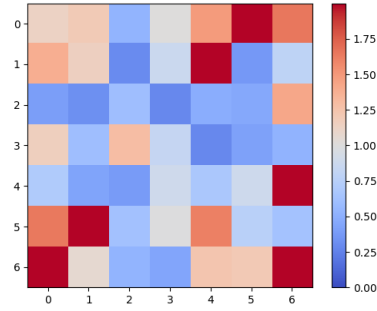
Apesar das limitações do experimento, o rastreador apresentou um desempenho semelhante ao de um rastreador comercial.



(a)



(b)



(c)

Figura 6.3: Erros médios para cada posição na grade, usando amostras correspondentes a uma grade  $7 \times 7$ . Em **(a)** As imagens usadas têm dimensão  $20 \times 15$ , em **(b)** as imagens são  $40 \times 30$  e em **(c)** as imagens são  $60 \times 80$ .

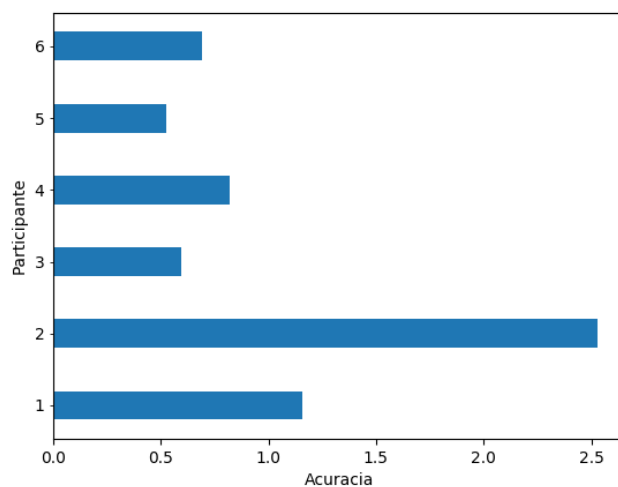


Figura 6.4: A figura mostra a acurácia para diferentes participantes usando o algoritmo para estimar olhar com imagens  $60 \times 80$ . A acurácia para o participante 2 foi pior porque ele piscou durante a coleta das amostras.

# Capítulo 7

## Conclusão

Neste trabalho estudamos conceitos de rastreamento de olhar, a teoria de *Compressive Sensing*(CS) e desenvolvemos um programa de rastreamento de olhar.

CS é útil para recuperar sinais esparsos, resolvendo com alta probabilidade um problema NP. Uma variação do CS pode ser usada para comparar imagens, usamos esta variação para construir um rastreador de olhar.

Elaboramos um experimento para testar a precisão e acurácia do rastreador e notamos que o programa apresenta algumas limitações, por exemplo, o programa não consegue estimar o olhar para determinada região se o participante piscar durante a coleta da amostra correspondente. Apesar das limitações, o experimento apresentou um resultado semelhante ao de um rastreador comercial.

Este trabalho contribuiu para a formação do aluno em Matemática Aplicada pois, durante o ano o aluno estudou a técnica de *Compressive Sensing*, que apresenta resultados não triviais, e a aplicou em um problema de processamento de imagens.

# Referências Bibliográficas

- [1] Mark A Davenport, Marco F Duarte, Yonina C Eldar, and Gitta Kutyniok. Introduction to compressed sensing. *Preprint*, 93(1):2, 2011.
- [2] D Donoho and Y Tsaig. Fast solution of l1-norm minimization problems when the solution may be sparse, 2006. *Preprint*, 1(2).
- [3] Andrew Duchowski. *Eye tracking methodology: Theory and practice*, volume 373. Springer Science & Business Media, 2007.
- [4] Andrew T Duchowski. A breadth-first survey of eye-tracking applications. *Behavior Research Methods, Instruments, & Computers*, 34(4):455–470, 2002.
- [5] Massimo Fornasier. Numerical methods for sparse recovery. *Theoretical foundations and numerical methods for sparse recovery*, 14:93–200, 2010.
- [6] Simon Foucart and Holger Rauhut. *A mathematical introduction to compressive sensing*, volume 1. Springer, 2013.
- [7] TNE Greville. The pseudoinverse of a rectangular matrix and its statistical applications. *Amer. Statist. Assoc., Proc. Soc. Statist. Sect*, pages 116–121, 1958.
- [8] Roger A Horn and Charles R Johnson. *Matrix analysis*. Cambridge university press, 2012.
- [9] Gitta Kutyniok. Theory and applications of compressed sensing. *GAMM-Mitteilungen*, 36(1):79–101, 2013.

- [10] Dongheng Li, David Winfield, and Derrick J Parkhurst. Starburst: A hybrid algorithm for video-based eye tracking combining feature-based and model-based approaches. In *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)-Workshops*, pages 79–79. IEEE, 2005.
- [11] R Lima, A Borges, and A Kurauchi. A comparison between intel realsense and tobii eyex for gaze estimation. 2016.
- [12] Robert Gabriel Lupu and Florina Ungureanu. A survey of eye tracking methods and applications. *Bul Inst Polit Iasi*, pages 71–86, 2013.
- [13] Roberto Valenti, Jacopo Staiano, Nicu Sebe, and Theo Gevers. Webcam-based visual gaze estimation. In *International Conference on Image Analysis and Processing*, pages 662–671. Springer, 2009.
- [14] J Wright and Y Ma. Dense error correction via l1-minimization (2008)(preprint).
- [15] Allen Y Yang, Arvind Genesh, Zihan Zhou, S Shankar Sastry, and Yi Ma. A review of fast l (1)-minimization algorithms for robust face recognition. Technical report, DTIC Document, 2010.
- [16] Y Zhang. Theory of compressive sensing via l1-minimization: a non-rip analysis and extensions, rice university, houston. Technical report, TX, Tech. Rep, 2008.