UNIVERSIDADE DE SÃO PAULO
Instituto de Ciências Matemáticas e de Computação

Relatório

Problemas de Mínimos Quadrados de Otimização Não Linear Aplicado a Reconstrução de Imagens

Prof. Dr. Elias Salomão Helou Neto

Anderson B. S. Lavinscky, 9045195
Emanuel Victor da Silva Favorato, 12558151
João Marcelo Rodrigues Júnior, 8531118

SÃO CARLOS
2024

1 Introdução

A otimização é uma área central da matemática aplicada que busca resolver questões práticas e teóricas relacionadas à escolha da melhor solução em um conjunto de possibilidades. Seja para minimizar custos em um processo industrial, maximizar lucros em um modelo econômico ou ajustar parâmetros em algoritmos de inteligência artificial, o objetivo é sempre encontrar a configuração mais eficiente ou desejada, de acordo com critérios estabelecidos. Essa busca por soluções ótimas abrange problemas extremamente diversos e está presente em campos como engenharia, ciência de dados, logística, finanças e muitas outras áreas.

Em geral, os problemas de otimização podem ser classificados em duas categorias principais: problemas contínuos e problemas discretos. Os problemas discretos envolvem conjuntos finitos de soluções, muitas vezes relacionados a questões como planejamento de rotas, alocação de recursos ou programação de tarefas. Por outro lado, nos problemas contínuos, como os encontrados na programação não linear, o conjunto viável de soluções é infinito e apresenta uma natureza contínua, permitindo a aplicação de ferramentas como o cálculo diferencial e conceitos de convexidade para encontrar soluções ótimas.

A complexidade surge quando a função objetivo ou as restrições possuem comportamento não linear. Esses problemas, ao mesmo tempo desafiadores e amplamente aplicáveis, requerem métodos sofisticados para serem resolvidos. Entre os métodos mais utilizados estão aqueles baseados no cálculo diferencial, como o método de Newton, que aproveita informações sobre a curvatura da função, e o método do gradiente conjugado, que é especialmente eficiente para problemas de alta dimensão. Essas ferramentas matemáticas e computacionais permitem abordar desde problemas de otimização irrestrita, onde as variáveis podem assumir qualquer valor dentro do espaço real, até problemas com restrições complexas, exigindo análises mais elaboradas para garantir a convergência a uma solução ótima. Para isso, é essencial concentrar-se nos fundamentos e nas condições que asseguram a otimalidade dessas soluções.

2 Problemas de Otimização Não Linear

Um problema clássico de otimização não linear pode ser formulado da seguinte maneira:

$$\begin{aligned} &\text{minimizar } f(x) \\ &\text{sujeito a } x \in S, \end{aligned}$$

onde $f : R^3 \longmapsto R$ é a função objetivo e $S \subset R^n$ é o conjunto de restrições, chamado de conjunto viável ou factível. O objetivo é encontrar um ponto $x^* \in S$ que minimize $f(x)$, ou seja, onde o valor de $f(x)$ seja menor ou igual ao de qualquer outro ponto viável. Neste tipo de problema há dois tipos de soluções a serem consideradas:

- Mínimo local: x^* é um mínimo local se existe um raio $\epsilon > 0$ tal que $f(x) \geq f(x^*)$ para todo $x \in S$ que satisfaça $\|x - x^*\| < \epsilon$.
- Mínimo global: x^* é um mínimo global se $f(x) \geq f(x^*)$ para todo $x \in S$.

Analogamente, a formulação de um problema que exige maximizar uma função $f(x)$, sujeita a restrições, busca determinar um ponto $x^* \in S$ tal que o valor de $f(x)$ seja maior ou igual ao de qualquer outro ponto viável. Assim como no caso da minimização, definem-se os máximos locais e globais com conceitos análogos. Vale notar que maximizar uma função f é equivalente a minimizar $-f$, o que pode tornar as análises mais simples.

2.1 Condições de Otimalidade

A otimização com ou sem restrições trata do problema de minimizar ou maximizar uma função de uma variável ou de várias variáveis. A maioria dos métodos busca uma direção e, em seguida, minimiza (ou maximiza) ao longo dessa direção. Esse processo de busca linear é equivalente à minimização de uma função de uma variável, com ou sem restrições, como limites inferiores e superiores nas variáveis.

Identificar as soluções ideais é utilizar as condições de otimalidade que analisam as propriedades das funções e seus gradientes. No caso de problemas sem restrições, onde $S = R^n$, temos as seguintes condições importantes:

- Primeira Ordem: Se x^* é um mínimo local, então $\nabla f(x^*) = 0$. Isso significa que o gradiente da função f se anula no ponto x^*, indicando uma possível extremidade (mínimo, máximo ou ponto de sela).
- Segunda Ordem: Para $f \in C^2$ (função duas vezes diferenciável), se x^* é um mínimo local:
  - $\nabla f(x^*) = 0$;
  - A matriz Hessiana $\nabla^2 f(x^*)$ deve ser semidefinida positiva ($\lambda_i > 0$ para todos os autovalores λ_i).

As condições necessárias para problemas sujeitos a restrições são análogas, onde a função em questão assume a forma $g(x^*) = f(x^*) + d^T h(x^*)$. Essas condições são a base de muitos algoritmos utilizados em otimização. E, sendo assim, a escolha da técnica de otimização depende da natureza da função $f(x)$ em estudo. Não há uma técnica de otimização que seja universal, mas existem informações que guiam essa escolha.

3 O Problema

A reconstrução de imagens é um dos problemas clássicos na área de otimização, frequentemente abordado em aplicações de processamento de sinais e análise de dados visuais. Nesse contexto, é comum que uma imagem degradada, como borrada ou distorcida, seja apresentada a um sistema computacional com o objetivo de extrair informações relevantes e reconstruir a figura original com o maior grau possível de fidelidade.

Entretanto, a resolução desse problema frequentemente demanda um esforço computacional significativo, especialmente devido ao tamanho elevado das imagens e à complexidade inerente dos modelos utilizados para descrever a degradação. Em tais casos, torna-se essencial o emprego de métodos avançados, como algoritmos iterativos, que permitem otimizar o processo de reconstrução ao balancear precisão e eficiência computacional.

Neste projeto, foram investigados três algoritmos iterativos amplamente utilizados na reconstrução de imagens: (i) Método de Máxima Descida com Busca Direcional Exata; (ii) Método de Nesterov com Tamanho de Passo Fixo; e (iii) Método de Gradientes Conjugados. A análise contemplará suas definições, propriedades matemáticas, e uma avaliação detalhada de suas vantagens e limitações. A comparação dos métodos será realizada com base em métricas como o tempo total de execução, o número de iterações necessárias para convergência, e o comportamento da função objetivo ao longo do processo iterativo, destacando a eficiência e a precisão de cada abordagem.

4 Métodos

Para a reconstrução da imagem, foram utilizados três métodos, sendo eles: método de máxima descida com busca direcional exata, método de Nesterov com tamanho de passo fixo e método de gradientes conjugados. Após a aplicação de vários cenários e a análise das variáveis de interesse, como tempo e qualidade da imagem reconstruída, decidimos que 100 iterações seriam idealmente suficientes para reduzir o custo computacional e melhorar o desempenho dos algoritmos no processo de reconstrução.

A descrição dos métodos empregados, bem como de suas propriedades e fórmulas matemáticas serão discorridas a seguir:

4.1 Método de máxima descida com busca direcional exata

4.1.1 Condições de Otimalidade

O algoritmo converge para um ponto x^* que satisfaz as condições de otimalidade:

- Primeira ordem: $\nabla f(x^*) = 0$ (estacionariedade).
- Segunda ordem: Se $\nabla^2 f(x^*) > 0$, então x^* é um mínimo local.

4.1.2 Direção de Descida

A direção do gradiente para minimizar $\phi(x)$ é:

$$d_k = -\nabla\phi(x_k),$$

onde $\nabla\phi(x_k)$ é o gradiente de $\phi(x)$, calculado como:

$$\nabla\phi(x) = J_F(x)^T F(x),$$

com $J_F(x)$ sendo a matriz Jacobiana de $F(x)$:

$$J_F(x) = \begin{bmatrix} \frac{\partial f_1}{\partial x_1} & \cdots & \frac{\partial f_1}{\partial x_n} \\ \vdots & \ddots & \vdots \\ \frac{\partial f_n}{\partial x_1} & \cdots & \frac{\partial f_n}{\partial x_n} \end{bmatrix}.$$

Assim, a direção de descida fica:

$$d_k = -J_F(x_k)^T F(x_k).$$

4.1.3 Procedimento

Este algoritmo segue os passos descritos a seguir.

1. Inicialização
   Escolha um ponto inicial x_0 e defina uma tolerância $\epsilon > 0$.
2. Iteração
   (a) Cálculo da direção de descida - Em cada iteração k, a direção de descida d_k é escolhida como o negativo do gradiente da função no ponto atual: $d_k = -\nabla f(x_k)$.
   Esta é a direção de maior declive, ou seja, a direção na qual $f(x)$ decresce mais rapidamente.
   (b) Busca direcional exata - Encontra-se o passo $\alpha_k > 0$ que minimiza $f(x_k + \alpha d_k)$ ao longo da direção d_k: $\alpha_k = \arg\min_{\alpha>0} f(x_k + \alpha d_k)$.
   (c) Incremento do ponto - O próximo ponto é calculado como $x_{k+1} = x_k + \alpha_k d_k$, onde α_k é o tamanho do passo.
   (d) Critério de parada - O algoritmo para quando $\|\nabla f(x_k)\| < \epsilon$, onde ϵ é um limite predefinido para a norma do gradiente, indicando que x_k está próximo de um ponto estacionário.

4.1.4 Busca Direcional Exata

No caso específico em que $F(x)$ é derivado de uma função quadrática, $\phi(x)$ também é quadrática:

$$\phi(x) = \frac{1}{2}x^T Ax + b^T x + c,$$

onde A é uma matriz simétrica definida positiva, b é um vetor e c é um escalar.

Para a busca direcional exata, minimizamos $\phi(x_k + \alpha d_k)$ com relação a α:

$$\phi(x_k + \alpha d_k) = \frac{1}{2}(x_k + \alpha d_k)^T A(x_k + \alpha d_k) + b^T(x_k + \alpha d_k) + c.$$

Expandindo:

$$\phi(x_k + \alpha d_k) = \frac{1}{2}\alpha^2 d_k^T A d_k + \alpha d_k^T(Ax_k + b) + \phi(x_k).$$

O valor ótimo de α é encontrado derivando em relação a α e igualando a zero:

$$\frac{d}{d\alpha}\phi(x_k + \alpha d_k) = \alpha d_k^T A d_k + d_k^T(Ax_k + b) = 0.$$

Resolvendo para α:

$$\alpha_k = -\frac{d_k^T(Ax_k + b)}{d_k^T A d_k}.$$

4.1.5 Propriedades e indicação

O método tem vantagens como ser simples de implementar, Utilização eficiente do gradiente para escolher a direção de descida, pode ser usado como uma etapa inicial em métodos mais avançados, como o método de Newton.

- Busca direcional exata - A minimização exata ao longo da direção d_k garante que a função objetivo decresça de maneira eficiente em cada iteração.
- Direção de maior declive - O uso do gradiente garante uma escolha natural da direção de descida.

Este algoritmo tem algumas limitações que precisam ser consideradas. A busca exata pode ser computacionalmente onerosa, especialmente em problemas de alta dimensão ou quando a função objetivo apresenta elevada complexidade. Além disso, em casos de problemas mal condicionados ou funções com comportamento não quadrático, o algoritmo pode demandar um número elevado de iterações para convergir. O custo associado à busca exata em cada iteração torna-se significativo. Outra limitação é que o método não considera a curvatura da função, o que pode levar a passos excessivamente curtos ou longos, dependendo da geometria do problema.

O algoritmo é mais adequado para funções suaves, onde a f(x) seja diferenciável com gradiente contínuo. E também empregado em problemas de baixa ou moderada dimensão e problemas bem condicionados, visto que o desempenho do método depende fortemente da forma e dimensão da função a ser estudada.

4.2 Método de Nesterov com tamanho de passo fixo

O algoritmo de Nesterov, também chamado de Método de Gradiente Acelerado de Nesterov (NAG), é uma técnica de otimização projetada para resolver problemas de minimização convexa com maior eficiência do que o método de gradiente clássico. É amplamente utilizado em aprendizado de máquina e em problemas de otimização convexa devido à sua capacidade de acelerar a convergência.

O método acelerado de Nesterov resolve problemas da forma:

$$\min_{x \in R^n} f(x),$$

onde $f(x)$ é uma função convexa e diferenciável com gradiente Lipschitz-contínuo, ou seja, existe uma constante $L > 0$ tal que:

$$\|\nabla f(x) - \nabla f(y)\| \leq L\|x - y\|, \quad \forall\ x, y \in R^n.$$

A versão com tamanho de passo fixo simplifica o método ao manter o tamanho do passo constante ao longo das iterações, em vez de ajustá-lo dinamicamente.

4.2.1 Condições de Otimalidade

1. Condição de Primeira Ordem - O ponto x^* é solução se satisfizer $\nabla f(x^*) = 0$.

2. Condição de Convexidade - A função $f(x)$ deve ser convexa e suavemente diferenciável, com constante de Lipschitz L.

3. Convergência Acelerada - Para funções convexas, a taxa de convergência do método de Nesterov é $O(1/k^2)$, onde k é o número de iterações. Essa taxa é superior ao método de gradiente padrão, que possui convergência de ordem $O(1/k)$.

4.2.2 Procedimento

1. Inicialização - Escolhem-se dois vetores iniciais, x_0 e y_0, e define-se o tamanho de passo η, que deve satisfazer $\eta < \frac{1}{L}$.

   - $x_0 = y_0$;
   - $t_0 = 1$,

   onde y_k é o ponto antecipado e t_k é o parâmetro de aceleração.

2. Iteração - Em cada transição k, realizamos os seguintes passos:

   (a) Atualização do Gradiente - Calculamos o gradiente da função objetivo no ponto y_k: $g_k = \nabla f(y_k)$.

   (b) Atualização do Ponto - Movemos o ponto na direção do gradiente: $x_{k+1} = y_k - \frac{1}{L} g_k$.

   (c) Atualização do Parâmetro de Aceleração - Atualizamos o parâmetro t_k: $t_{k+1} = \frac{1+\sqrt{1+4t_k^2}}{2}$.

   (d) Atualização do Ponto Antecipado - Calculamos y_{k+1} usando uma combinação do novo ponto x_{k+1} e do anterior x_k: $y_{k+1} = x_{k+1} + \frac{t_k - 1}{t_{k+1}}(x_{k+1} - x_k)$.

   (e) Critério de Parada - O algoritmo para quando $\|\nabla f(y_k)\| < \epsilon$, onde ϵ é a tolerância predefinida.

4.2.3 Busca Direcional Exata

No caso de funções quadráticas:

$$f(x) = \frac{1}{2} x^T A x - b^T x + c,$$

onde A é uma matriz simétrica definida positiva, a busca direcional exata encontra α_k que minimiza $f(y_k - \alpha g_k)$.

A função $f(y_k - \alpha g_k)$ é dada por:

$$f(y_k - \alpha g_k) = \frac{1}{2}(y_k - \alpha g_k)^T A (y_k - \alpha g_k) - b^T (y_k - \alpha g_k) + c.$$

Expandindo:

$$f(y_k - \alpha g_k) = \frac{1}{2}\alpha^2 g_k^T A g_k - \alpha g_k^T (A y_k - b) + f(y_k).$$

Derivando em relação a α e igualando a zero:

$$\frac{d}{d\alpha} f(y_k - \alpha g_k) = \alpha g_k^T A g_k - g_k^T (A y_k - b) = 0.$$

Resolvendo para α_k:

$$\alpha_k = \frac{g_k^T (A y_k - b)}{g_k^T A g_k}.$$

4.2.4 Propriedades e indicação

Este algoritmo oferece uma melhor taxa de convergência em comparação ao método de gradiente clássico. É mais simples de implementar e não exige cálculos adicionais para ajustar o passo dinamicamente. Além disso, é um método que se adequado eficientemente para problemas de otimização convexa em espaços de alta dimensão.

Por outro lado, o método requer conhecimento prévio da constante de Lipschitz L para definir o passo adequadamente. O algoritmo pode apresentar comportamento instável ou dificuldade em convergir para soluções globais em funções não convexas. Além disso, uma má escolha do tamanho do passo pode levar a oscilações ou convergência lenta.

Portanto, este algoritmo é recomendado para funções com boa regularidade e constante L conhecida, bem como para problemas de otimização convexa suave, como aqueles encontrados em regressão linear e modelos de aprendizado supervisionado.

4.3 Método de gradientes conjugados

Este algoritmo é um método iterativo amplamente utilizado para resolver problemas de otimização, especialmente em contextos de minimização de funções quadráticas e sistemas lineares simétricos e definidos positivos. Ele combina eficiência computacional com uma abordagem que não exige a armazenagem completa de matrizes, o que o torna adequado para problemas de alta dimensão.

O algoritmo busca minimizar uma função objetivo $f(x)$, que geralmente tem a forma quadrática:

$$f(x) = \frac{1}{2} x^T Q x - b^T x + c,$$

onde Q é uma matriz simétrica e definida positiva ($Q > 0$), b é um vetor de coeficientes e c é um escalar constante.

O método do gradiente conjugado não linear é usado para minimizar uma função $f(x)$, onde $f : R^n \to R$ é uma função não linear. O problema pode ser formulado como:

$$\min_{x \in R^n} f(x),$$

onde $f(x)$ é diferenciável e a busca é conduzida iterativamente até encontrar um ponto x tal que $\nabla f(x) = 0$. Utiliza direções conjugadas para encontrar o ponto de mínimo, evitando redundâncias que podem surgir com métodos baseados exclusivamente em gradientes.

4.3.1 Condições de Otimalidade

Para que um ponto x^* seja o mínimo de $f(x)$, ele deve satisfazer:

1. A condição de estacionariedade: $\nabla f(x^*) = Qx^* - b = 0$.
2. A matriz Hessiana Q deve ser definida positiva ($Q > 0$), garantindo que o ponto crítico é um mínimo global.

4.3.2 Procedimento

O método constrói uma sequência de aproximações $x_0, x_1, \ldots, x_k$ que convergem para o mínimo x^* ao longo de direções conjugadas. O funcionamento básico pode ser descrito como:

1. Inicialização
   - Escolha de um ponto inicial x_0;
   - Cálculo do gradiente inicial $r_0 = \nabla f(x_0) = Qx_0 - b$;
   - Definição da primeira direção $d_0 = -r_0$ (direção oposta ao gradiente).
2. Iteração
   As iterações do algoritmo são realizadas de acordo com os seguintes passos:
   (a) Busca Direcional Exata ou Aproximada - Determinamos o tamanho do passo α_k que minimiza $f(x_k + \alpha d_k)$: $\alpha_k = \arg\min_\alpha f(x_k + \alpha d_k)$.
   (b) Atualização da Solução - Atualizamos o ponto atual usando $x_{k+1} = x_k + \alpha_k d_k$.
   (c) Atualização do Gradiente - Calculamos o novo gradiente $g_{k+1} = \nabla f(x_{k+1})$.
   (d) Cálculo do Coeficiente β_k - Calculamos β_k, que controla a conjugação das direções. Existem diferentes fórmulas para β_k, incluindo:
   - Fórmula de Fletcher-Reeves: $\beta_k^{FR} = \frac{g_{k+1}^T g_{k+1}}{g_k^T g_k}$.
   - Fórmula de Polak-Ribiere: $\beta_k^{PR} = \frac{g_{k+1}^T (g_{k+1} - g_k)}{g_k^T g_k}$.
   - Fórmula de Hestenes-Stiefel: $\beta_k^{HS} = \frac{g_{k+1}^T (g_{k+1} - g_k)}{d_k^T (g_{k+1} - g_k)}$.

   Escolhemos uma fórmula adequada para o problema específico.
   (e) Atualização da Direção - Atualizamos a direção conjugada $d_{k+1} = -g_{k+1} + \beta_k d_k$.
   (f) Critério de Parada - O algoritmo termina quando $\|g_{k+1}\| < \epsilon$, onde ϵ é a tolerância predefinida.

Os critérios mais comuns para interromper o algoritmo incluem: (i) O gradiente ser menor que um limite pré-definido ($\|r_k\| < \epsilon$); (ii) Número máximo de iterações atingido.

4.3.3 Busca Direcional Exata

No caso de funções quadráticas:

$$f(x) = \frac{1}{2}x^T Ax - b^T x + c,$$

onde A é simétrica e definida positiva, a busca direcional exata é embutida no cálculo de α_k.

A função $f(x_k + \alpha d_k)$ é dada por:

$$f(x_k + \alpha d_k) = \frac{1}{2}(x_k + \alpha d_k)^T A(x_k + \alpha d_k) - b^T(x_k + \alpha d_k) + c.$$

Expandindo:

$$f(x_k + \alpha d_k) = \frac{1}{2}\alpha^2 d_k^T A d_k + \alpha d_k^T(Ax_k - b) + f(x_k).$$

Derivando em relação a α e igualando a zero:

$$\frac{d}{d\alpha}f(x_k + \alpha d_k) = \alpha d_k^T A d_k + d_k^T g_k = 0,$$

onde $g_k = \nabla f(x_k) = Ax_k - b$.

Resolvendo para α_k:

$$\alpha_k = -\frac{d_k^T g_k}{d_k^T A d_k}.$$

4.3.4 Propriedades e indicação

A convergência desse método para funções quadráticas com matriz Q de dimensão n ocorre em no máximo n passos, ou seja, a solução exata é encontrada com n iterações.

O método não exige o armazenamento completo de Q, o que torna computacionalmente eficiente e ideal para grandes sistemas esparsos. Além disso, em problemas não quadráticos, pode ser usado com variações, como o método de gradiente conjugado não linear.

O algoritmo de gradiente conjugado é especialmente indicado para: problemas de grandes dimensões em que Q é esparsa e definida positiva; Minimização de funções quadráticas: Problemas de ajuste de curva, resolução de sistemas lineares e simulações físicas; Como parte de algoritmos para otimização não linear, combinando-o com estratégias de linearização ou como subproblema em métodos de barreira ou penalidade.

A eficiência do método depende de Q ser definida positiva e simétrica. E em problemas não quadráticos, a convergência não é garantida, sendo necessário adaptar o método (e.g., reinicializações). Além disso, o método é sensível a erros de arredondamento em problemas mal condicionados (matriz Q com alto número de condicionamento).

Para problemas não quadráticos, o método é adaptado para lidar com funções mais gerais, usando, por exemplo: (i) Gradiente conjugado não linear: Considera atualizações da direção conjugada baseadas no gradiente atual; (ii) Métodos híbridos: Combina o gradiente conjugado com estratégias como o método de Newton ou quasi-Newton.

5 Metodologia

O estudo de reconstrução de imagens utilizou três algoritmos principais: Método de Máxima Descida com Busca Direcional Exata Método de Nesterov com Tamanho Fixo e Método de Gradientes Conjugados À imagem de estudo foi normalizada para o intervalo $[0, 1]$ e convertida para tons de cinza. Os dados de entrada incluíram valores de erro $\epsilon = 10^{-12}$, de número fixo de iterações máxima $n = 0$, tamanho de passo fixo e parâmetros de desfoque (σ), que tiveram valores $\sigma = 3,\ 5\ e\ 7$. Dentre as funções utilizadas no código vale a pena destacar as mais importantes como a função de perda ($loss_{func}$) usada no cálculo da função objetivo, a $grad_{func}$ para obtenção do gradiente da f, e algumas específicas como $gradient^{exact}_{descent}$, $nesterov^{fixed}_{step}$, e $conjugate_{grad}$ para a reconstrução da imagem. O método $sparse_{cg}$ é auxiliar da $conjugate_{grad}$ que é implantado com um pré-condicionador diagonal para obter soluções de gradiente conjugado para matrizes esparsas, resolvendo um sistema linear da forma $A \cdot x = b$.

Os resultados foram coletados em termos do parâmetro de desfoque (σ), tempo de execução e iterações necessárias para cada método. O processo em si consistiu em ler a imagem (Figura 1) a ser estudada, que originalmente é colorida, converter em tons de cinza e submeter ao processo de desfoque para, em seguida, realizar o processo de reconstrução da imagem borrada. Ao final, verificar se os algoritmos foram ou não eficientes em seu emprego de acordo com os parâmetros usados.

Figura 1: Imagem empregada para as análises.

6 Resultados e Discussões

A análise da eficácia dos algoritmos selecionados para reconstrução de imagens foi realizada por meio da aplicação de um filtro Gaussiano à imagem original em escala de cinza. Esse procedimento introduz um desfoque controlado, replicando a degradação frequentemente observada em problemas de processamento de sinais e visão computacional, como na restauração de imagens.

A Figura 2 ilustra os resultados obtidos ao aplicar o Método de Descida com busca direcional exata em uma imagem degradada por desfoque, utilizando o parâmetro $\sigma = 3$. Observou-se que a imagem reconstruída apresenta maior similaridade com a original do que com a borrada. O método demonstrou desempenho

satisfatório no processo de recuperação, contudo, ofereceu uma solução com tempo de execução elevado. Isto é condizente com o comportamento teórico esperado.

Figura 2: Comparação entre as imagens original, desfocada ($\sigma = 3$) e recuperada pelo emprego do método de Máxima Descida.

Figura 3: Comparação entre as imagens original, desfocada ($\sigma = 5$) e recuperada pelo emprego do método de Máxima Descida.

As imagens nas Figuras 3 e 4 exibiram comportamento semelhante ao observado na Figura 2, com as imagens recuperadas apresentando maior proximidade com a versão original do que com a desfocada. Contudo, à medida que o parâmetro de desfoque σ foi aumentado, o tempo de execução também se elevou significativamente, e o houve uma redução no desempenho do algoritmo em relação ao processo de recuperação da imagem.

As Figuras 5, 6 e 7 ilustram os resultados obtidos ao aplicar o Método de Nesterov com tamanho fixo à mesma imagem usada como entrada no algoritmo de Máxima Descida. Após ser degradada por desfoque, utilizando o parâmetro $\sigma = 3$, 5 e 7, respectivamente. Observou-se que a imagem reconstruída apresenta maior similaridade com a original do que com a borrada. Além disso, o método demonstrou uma baixa no desempenho no processo de recuperação com o aumento do parâmetro de desfoque. Contudo, o resultado

Figura 4: Comparação entre as imagens original, desfocada ($\sigma = 7$) e recuperada pelo emprego do método de Máxima Descida.

foi satisfatório mesmo com o valor mais alto de σ. O tempo de execução também foi similar ao método de Máxima Descida, ou seja, aumentou com o aumento de σ.

Figura 5: Comparação entre as imagens original, desfocada ($\sigma = 3$) e recuperada pelo emprego do método de Nesterov.

O Método de Gradiente Conjugado foi aplicado às imagens desfocadas com diferentes valores de σ (= 3, 5 e 7), seguindo a mesma abordagem utilizada para os algoritmos anteriores. As Figuras 8, 9 e 10 apresentam os resultados obtidos. Um aspecto notavelmente distinto em relação aos demais algoritmos foi o tempo de execução, que permaneceu reduzido mesmo para o maior valor do parâmetro de desfoque. Em relação ao processo de recuperação, o desempenho seguiu um comportamento similar ao dos outros métodos, com redução na qualidade à medida que σ aumentava. As imagens reconstruídas mostraram maior similaridade com as borradas do que com as originais, sendo o desempenho menos satisfatório para σ mais elevado. O tempo de execução, entretanto, destacou-se pela eficiência, permanecendo na ordem de décimos de segundo, independentemente da variação de σ.

O número de iterações foi mantido constante em todos os casos. Entretanto, devido ao rápido tempo

Figura 6: Comparação entre as imagens original, desfocada ($\sigma = 5$) e recuperada pelo emprego do método de Nesterov.

Figura 7: Comparação entre as imagens original, desfocada ($\sigma = 7$) e recuperada pelo emprego do método de Nesterov.

de execução do método de gradiente conjugado, testes adicionais foram realizados com valores mais elevados de iterações, alcançando $n_{\text{it}} = 10^9$, enquanto o parâmetro de desfoque permaneceu fixo ($\sigma = 7$). Os resultados obtidos mostraram-se consistentes, sem alterações significativas em relação ao teste anterior com 100 iterações e $\sigma = 7$. Esses resultados não foram armazenados e, portanto, não foram incluídos neste projeto.

Com base nas visualizações anteriores, foi possível visualizar que, considerando n = 100 iterações, o método de máxima descida com busca direcional exata foi o que melhor desempenho obteve na reconstrução da imagem, o método de Nesterov com tamanho de passo fixo igual a 0.005, o de melhor convergência para o problema apresentado, também apresentou resultados consistentes, retornando uma figura com boa resolução e com detalhes, como letras e funções, bastante visíveis. Por último, temos o método dos gradientes conjugados, o qual nos retornou uma fotografia de baixa resolução, difícil visualização e apresentando pouca melhora em relação à figura borrada.

Ao analisar visualmente as Figuras 8, 9 e 10 foi possível concluir que, após $n = 100$ iterações, o

Figura 8: Comparação entre as imagens original, desfocada ($\sigma = 3$) e recuperada pelo emprego do método de Gradiente Conjugado.

Figura 9: Comparação entre as imagens original, desfocada ($\sigma = 5$) e recuperada pelo emprego do método de Gradiente Conjugado.

método de Máxima Descida com busca direcional exata apresentou o melhor desempenho na reconstrução da imagem em todos os casos de desfoque da imagem. O método de Nesterov, utilizando tamanho de passo fixo igual a 0,005, destacou-se pela excelente convergência no problema em questão e também forneceu resultados consistentes, gerando uma figura com boa resolução e detalhes nítidos, como letras e funções visíveis. Em contraste, o método dos gradientes conjugados retornou uma imagem de baixa resolução, com visualização comprometida e pouca melhoria em relação à versão borrada que só piorou com o aumento de σ.

A análise dos tempos de execução dos algoritmos apresentou diferenças significativas entre as abordagens. O Método de Máxima Descida destacou-se pelo tempo de execução total e por iteração consideravelmente superior, chegando a ser quase cinco vezes maior que o do Método de Nesterov, o que confere a este último uma vantagem computacional, ainda que com perdas moderadas no desempenho de reconstrução. O algoritmo de Gradiente Conjugado obteve a melhor performance em termos de eficiência temporal, com tempos de execução de 0,11 segundos para $\sigma = 3$ e 0,10 segundos para $\sigma = 5$. No entanto, sua qualidade de

Figura 10: Comparação entre as imagens original, desfocada ($\sigma = 7$) e recuperada pelo emprego do método de Gradiente Conjugado.

Figura 11: Comparação entre os resultados das imagens recuperadas pelos três métodos para $\sigma = 3$.

Figura 12: Comparação entre os resultados das imagens recuperadas pelos três métodos para $\sigma = 5$.

Figura 13: Comparação entre os resultados das imagens recuperadas pelos três métodos para $\sigma = 7$.

reconstrução não atingiu o mesmo nível observado nos demais métodos. A Tabela 1 apresenta uma síntese dos tempos de execução para diferentes valores de σ.

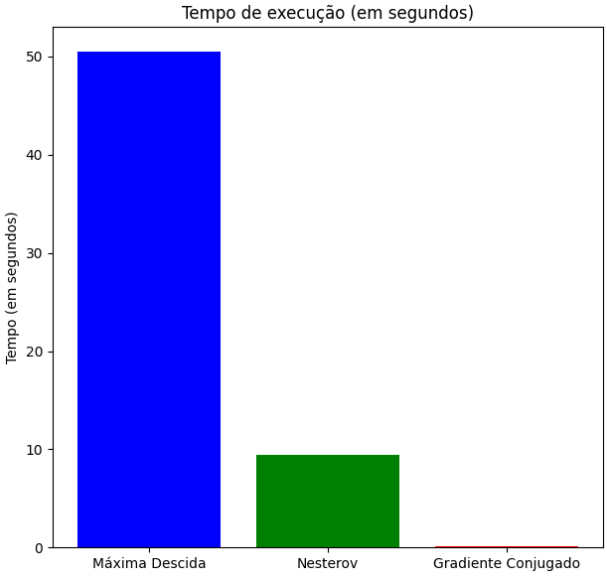

Figura 14: Tempo total de execução para $\sigma = 3$.

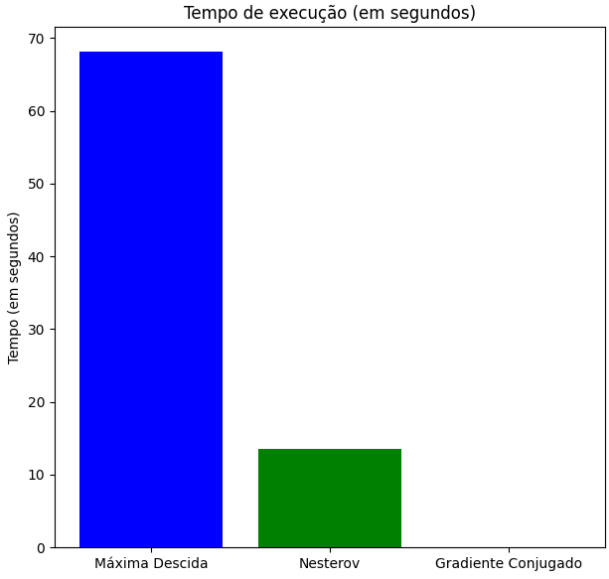

Figura 15: Tempo total de execução para $\sigma = 5$.

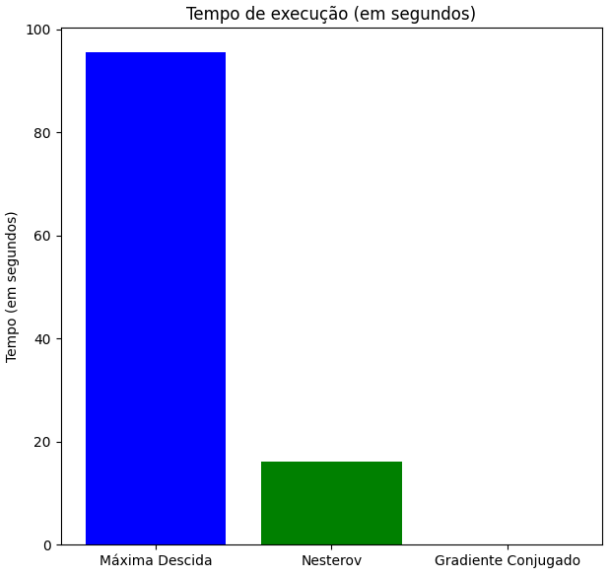

Figura 16: Tempo total de execução para $\sigma = 7$.

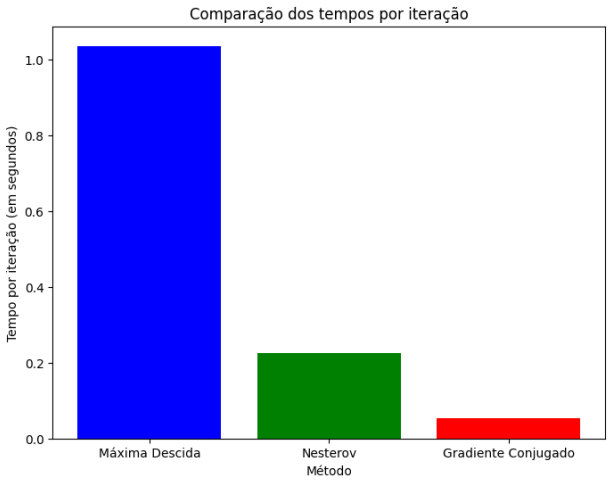

Figura 17: Tempo de execução por iteração para $\sigma = 7$.

σ	$t_{MD}(s)$	$t_{NT}(s)$	$t_{GC}(s)$
3	52,22	10,24	0,08
5	69,16	13,59	0,07
7	97,48	16,39	0,07

Tabela 1: Resumo dos testes realizados com os algoritmos de estudo.

Uma métrica de grande relevância na análise é a perda (ou função objetivo) por iteração, dado que o objetivo é minimizá-la ao máximo. A Figura 18 apresenta um comparativo do comportamento dessa função para os três algoritmos avaliados.

Os resultados mostram um valor relativamente baixo da função de perda para o método de gradiente conjugado, que convergiu em apenas 5 iterações. Em contrapartida, o método de Nesterov estabilizou por volta de 20 iterações, porém em níveis significativamente superiores aos demais algoritmos. Já o método de máxima descida exibiu uma trajetória consistentemente decrescente para a função objetivo ao longo das iterações.

Esse padrão observado é incomum, dado que seria esperado um comportamento uniforme de redução da função objetivo para os três algoritmos, com o método de Nesterov superando o de máxima descida em termos de valores finais, já que o primeiro é amplamente considerado uma abordagem otimizada para minimizar perdas. Contudo, deve-se ressaltar que as interpretações baseadas na Figura 18 devem ser feitas com cautela, pois menores perdas e convergência mais rápida não implicaram, necessariamente, em melhorias no desempenho da reconstrução imagética.

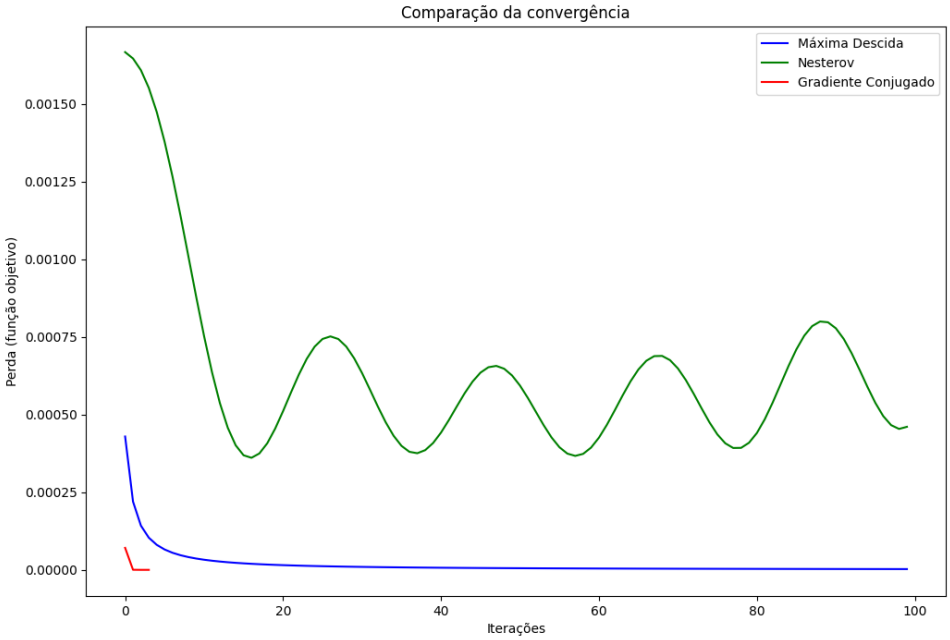

Figura 18: Avaliação da função objetivo em todos os algoritmos usados no estudo com n = 100 iterações.

7 Conclusão

O estudo realizado evidenciou a importância de algoritmos de otimização no problema de reconstrução de imagens degradadas. A análise das métricas, como o comportamento da função objetivo por iteração e a qualidade das imagens recuperadas, destacou as forças e limitações de cada método. O algoritmo de máxima descida com busca direcional exata apresentou boa redução da função objetivo, mas a um alto custo computacional. O método de Nesterov, apesar de sua rápida convergência, mostrou limitações na qualidade da reconstrução, enquanto o gradiente conjugado se destacou pelo equilíbrio entre eficiência e desempenho na minimização da função objetivo.

Os resultados mostram que a escolha do algoritmo ideal depende das características do problema, incluindo a natureza da degradação e os recursos computacionais disponíveis. Uma abordagem híbrida que combine as vantagens de diferentes métodos pode ser uma alternativa promissora. Além disso, a análise demonstrou que métricas como a função objetivo, embora úteis, não são suficientes para garantir melhorias qualitativas na reconstrução, sendo necessário um cuidado maior na avaliação dos resultados.

Em síntese, o projeto reforçou a relevância da otimização para o processamento de imagens, destacando a necessidade de soluções adaptativas para problemas complexos. O equilíbrio entre custo computacional e qualidade do resultado permanece como um desafio, abrindo caminho para futuras pesquisas e desenvolvimentos no campo.

8 Referências bibliográficas

1. OLIVEIRA, M. C., FERREIRA, L. V. B., and BARREIROS, M. O. Classificação de doenças cardiovasculares utilizando aprendizado de máquina, Multidebates, v.7, n. 1, pg.38-4, 2023, Disponível em: `https://revista.faculdadeitop.edu.br/index.php/revista/article/view/56`. Acesso em: 16 jun. 2024.

2. da SILVA FILHO, F. R., and COUTINHO, E. F. Aprendizado de máquina para predição de diagnósticos de doenças cardiovasculares, Anais do XXII Simpósio Brasileiro de Computação Aplicada à Saúde, p.358-369, org.SBC, 2022, Disponível em: `https://sol.sbc.org.br/index.php/sbcas/article/view/21646`. Acesso em: 16 jun. 2024.

3. BERTSEKAS, D. P. Non Linear Programming, Athena Scientific, 2003.

4. FRIEDLANDER, A., Elementos de programação não linear, Editora da Unicamp, 1994.

5. BAZARAA, M. S., and SHERALI, H. D., and SHETTY, C. M. Nonlinear programming: Theory and algorithms, Wiley, 2006.

6. NESTEROV, Yu. E. A method of solving a convex programming problem with convergence rate $O(1/k^2)$, Dokl. Akad. Nauk SSSR, v. 269, n. 3, 543–547, 1983.

7. Nonlinear conjugate gradient method. In: WIKIPÉDIA: a enciclopédia livre. Disponível em: `https://en.wikipedia.org/wiki/Nonlinear_conjugate_gradient_method`. Acesso em: 11 dez. 2024.

8. Gradient descent. In: WIKIPÉDIA: an enciclopédia livre. Disponível em: `https://en.wikipedia.org/wiki/Gradient_descent#Solution_of_a_non-linear_system`. Acesso em: 12 dez. 2024

9. Conjugate gradient method. In: WIKIPÉDIA: an enciclopédia livre. Disponível em: `https://en.wikipedia.org/wiki/Conjugate_gradient_method`. Acesso em: 12 dez. 2024.