

Sobre um Método de Minimização Irrestrita Baseado em Derivadas Simplex

Bruno Henrique Cervelin

Orientadora: Profa. Dra. Maria Aparecida Diniz Ehrhardt

Mestrado em Matemática Aplicada
IMECC - Unicamp

7 de abril de 2013

- 1 Introdução
- 2 Métodos clássicos sem derivadas
- 3 Derivadas simplex
- 4 SID-PSM
- 5 Comparação entre os métodos
- 6 Problema de Parâmetros ótimos de algoritmos
- 7 Conclusão

Problema a ser tratado

- Problemas de minimização irrestrita :

$$\min f(x) \quad x \in \mathbf{R}^n.$$

- ∇f contínuo, porém não está disponível.

Com derivadas vs. sem derivadas

- Os métodos com derivadas possuem maior eficiência e robustez, e ainda, possuem ampla teoria de convergência na literatura.

Com derivadas vs. sem derivadas

- Os métodos com derivadas possuem maior eficiência e robustez, e ainda, possuem ampla teoria de convergência na literatura.

Por que utilizar métodos sem derivadas?

Com derivadas vs. sem derivadas

- Os métodos com derivadas possuem maior eficiência e robustez, e ainda, possuem ampla teoria de convergência na literatura.

Por que utilizar métodos sem derivadas?

Usamos quando não conseguimos (ou não queremos) calcular as derivadas!

Métodos clássicos - Busca Direta

- Principal característica: Valor da função objetivo é utilizado apenas de forma comparativa.

Métodos clássicos - Busca Direta

- Principal característica: Valor da função objetivo é utilizado apenas de forma comparativa.
- Exemplos:
 - ▶ Nelder-Mead;
 - ▶ Busca Padrão.

Método Nelder-Mead

- Método de Busca Simplex.

Método Nelder-Mead

- Método de Busca Simplex.
- Simplex é um conjunto de $n + 1$ pontos no \mathbb{R}^n ,

$$S = \{x^0, x^1, \dots, x^n\}$$

Método Nelder-Mead

- Método de Busca Simplex.
- Simplex é um conjunto de $n + 1$ pontos no \mathbb{R}^n ,

$$S = \{x^0, x^1, \dots, x^n\}$$

- Simplex não degenerados $\Leftrightarrow \{x^1 - x^0, \dots, x^n - x^0\}$ é L.I.

Método Nelder-Mead

- Método de Busca Simplex.
- Simplex é um conjunto de $n + 1$ pontos no \mathbb{R}^n ,

$$S = \{x^0, x^1, \dots, x^n\}$$

- Simplex não degenerados $\Leftrightarrow \{x^1 - x^0, \dots, x^n - x^0\}$ é L.I.
- Não possui resultados teóricos de convergência, possui exemplos em que falha.

Método Nelder-Mead

- Método de Busca Simplex.
- Simplex é um conjunto de $n + 1$ pontos no \mathbb{R}^n ,

$$S = \{x^0, x^1, \dots, x^n\}$$

- Simplex não degenerados $\Leftrightarrow \{x^1 - x^0, \dots, x^n - x^0\}$ é L.I.
- Não possui resultados teóricos de convergência, possui exemplos em que falha.
- Porém, possui bons resultados na prática e facilidade de implementação.

Método Nelder-Mead

- O método começa com um simplex não degenerado

$$S_0 = \{x_0^0, x_0^1, \dots, x_0^n\}.$$

Método Nelder-Mead

- O método começa com um simplex não degenerado

$$S_0 = \{x_0^0, x_0^1, \dots, x_0^n\}.$$

- A cada iteração k , o simplex S_k é ordenado de modo que

$$f(x_k^0) \leq f(x_k^1) \leq \dots \leq f(x_k^n).$$

Método Nelder-Mead

- O método começa com um simplex não degenerado

$$S_0 = \{x_0^0, x_0^1, \dots, x_0^n\}.$$

- A cada iteração k , o simplex S_k é ordenado de modo que

$$f(x_k^0) \leq f(x_k^1) \leq \dots \leq f(x_k^n).$$

- Tentamos substituir o pior ponto (x_k^n) através de uma reflexão, expansão ou contração.

Método Nelder-Mead

- O método começa com um simplex não degenerado

$$S_0 = \{x_0^0, x_0^1, \dots, x_0^n\}.$$

- A cada iteração k , o simplex S_k é ordenado de modo que

$$f(x_k^0) \leq f(x_k^1) \leq \dots \leq f(x_k^n).$$

- Tentamos substituir o pior ponto (x_k^n) através de uma reflexão, expansão ou contração.
- Se todas as formas acima falham, reduzimos o simplex mantendo apenas o melhor ponto (x_k^0)

Método Nelder-Mead

- O método começa com um simplex não degenerado

$$S_0 = \{x_0^0, x_0^1, \dots, x_0^n\}.$$

- A cada iteração k , o simplex S_k é ordenado de modo que

$$f(x_k^0) \leq f(x_k^1) \leq \dots \leq f(x_k^n).$$

- Tentamos substituir o pior ponto (x_k^n) através de uma reflexão, expansão ou contração.
- Se todas as formas acima falham, reduzimos o simplex mantendo apenas o melhor ponto (x_k^0)
- O critério de parada pode ser o diâmetro Δ do simplex

$$\Delta = \max_{i=1, \dots, n} \|x^i - x^0\|.$$

Ilustração Nelder-Mead

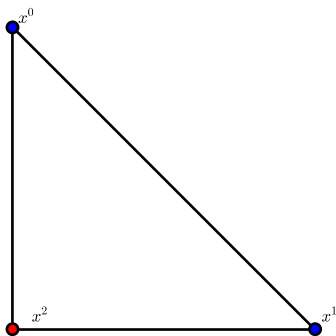


Ilustração Nelder-Mead

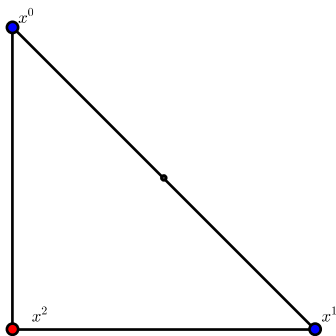


Ilustração Nelder-Mead

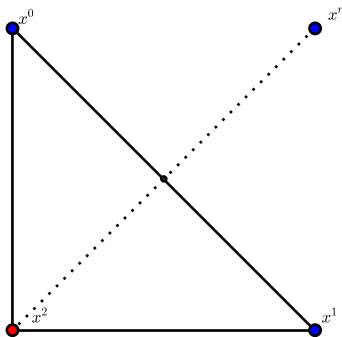


Ilustração Nelder-Mead

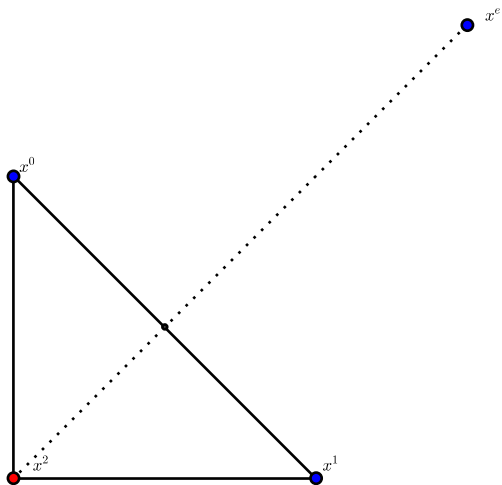


Ilustração Nelder-Mead

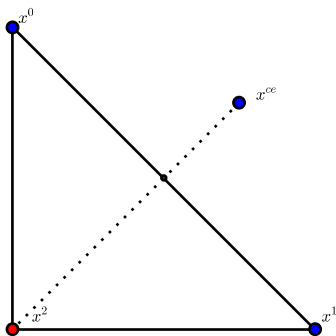


Ilustração Nelder-Mead

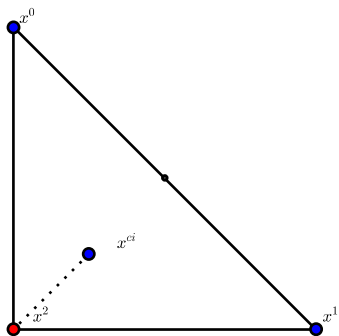
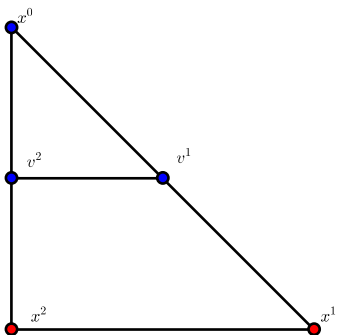


Ilustração Nelder-Mead



Método de Busca Padrão

- Método intuitivo e de fácil implementação.
- Possui teoria de convergência de primeira ordem.

Método de Busca Padrão

- Método intuitivo e de fácil implementação.
- Possui teoria de convergência de primeira ordem.

Pode ser dividido em três passos:

- 1) Passo de Busca;
- 2) Passo de Pesquisa;
- 3) Atualização de parâmetros.

Busca Padrão: Passo de Busca

- Esse passo é opcional, não é necessário para a teoria de convergência.

Busca Padrão: Passo de Busca

- Esse passo é opcional, não é necessário para a teoria de convergência.
- Se for executado, avaliamos finitos pontos da malha

$$M_k = \{x_k + \alpha_k Dz : z \in \mathbb{Z}^{|D|}\},$$

onde $D \subset \mathbb{R}^{n \times |D|}$ gera positivamente o \mathbb{R}^n .

Busca Padrão: Passo de Busca

- Esse passo é opcional, não é necessário para a teoria de convergência.
- Se for executado, avaliamos finitos pontos da malha

$$M_k = \{x_k + \alpha_k Dz : z \in \mathbb{Z}^{|D|}\},$$

onde $D \subset \mathbb{R}^{n \times |D|}$ gera positivamente o \mathbb{R}^n .

- Se encontrarmos algum ponto que diminua o valor de f , declaramos sucesso e pulamos o passo de pesquisa.

Busca Padrão: Passo de Pesquisa

- $D_k \subseteq D$, gerador positivo do \mathbf{R}^n e ordenado seguindo um padrão.

Busca Padrão: Passo de Pesquisa

- $D_k \subseteq D$, gerador positivo do \mathbf{R}^n e ordenado seguindo um padrão.
- Analisamos a função objetivo nos pontos do conjunto

$$P_k = \{x_k + \alpha_k d : d \in D_k\}.$$

Busca Padrão: Passo de Pesquisa

- $D_k \subseteq D$, gerador positivo do \mathbf{R}^n e ordenado seguindo um padrão.
- Analisamos a função objetivo nos pontos do conjunto

$$P_k = \{x_k + \alpha_k d : d \in D_k\}.$$

- Se encontrarmos algum ponto que diminua o valor de f , declaramos sucesso e terminamos o passo.

Busca Padrão: Passo de Pesquisa

- $D_k \subseteq D$, gerador positivo do \mathbf{R}^n e ordenado seguindo um padrão.
- Analisamos a função objetivo nos pontos do conjunto

$$P_k = \{x_k + \alpha_k d : d \in D_k\}.$$

- Se encontrarmos algum ponto que diminua o valor de f , declaramos sucesso e terminamos o passo.
- Se não, declaramos fracasso.

Busca Padrão: Atualização do passo

Fracasso:

Devemos diminuir o tamanho do passo fazendo:

$$\alpha_{k+1} = \theta \alpha_k,$$

onde $\theta \in (0, 1)$.

Busca Padrão: Atualização do passo

Sucesso:

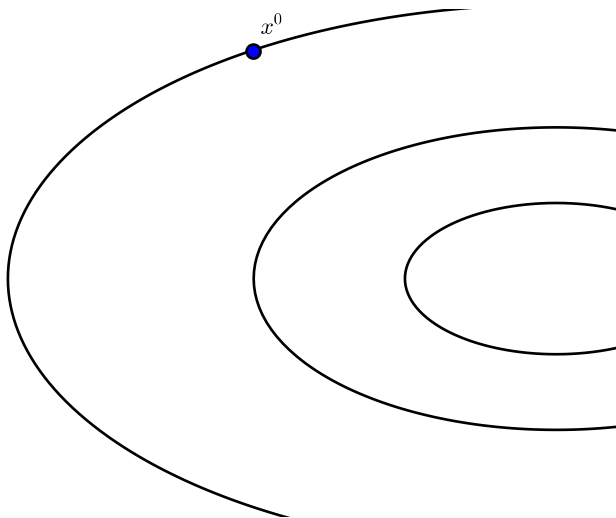
Ou aumentamos fazendo

$$\alpha_{k+1} = \phi \alpha_k,$$

onde $\phi > 1$, ou mantemos o tamanho do passo, fazendo

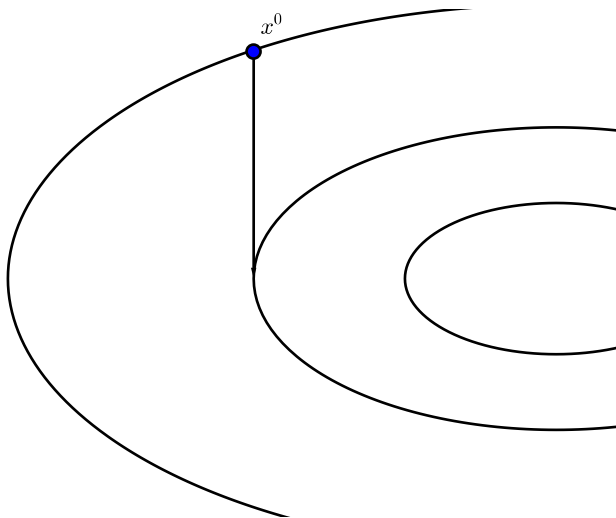
$$\alpha_{k+1} = \alpha_k.$$

Exemplo de execução do método de busca padrão



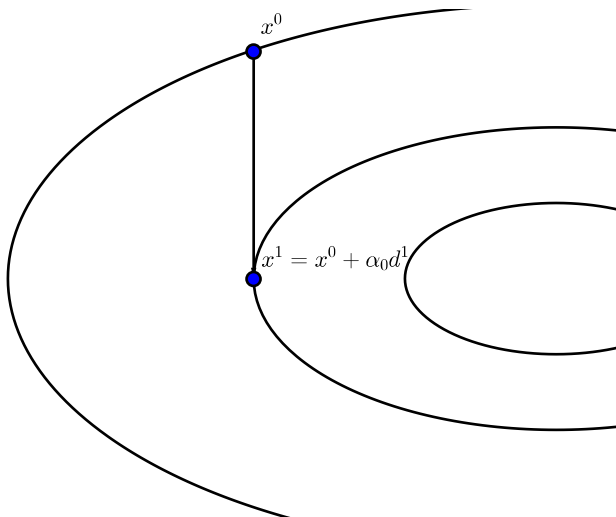
$$d^1 = (0, -1)^T; d^2 = (-1, 0)^T; d^3 = (1, 1)^T; \alpha_0 = 1$$

Exemplo de execução do método de busca padrão



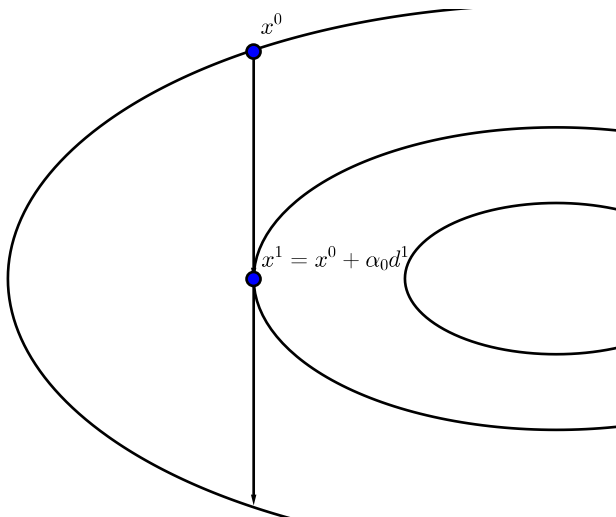
$$d^1 = (0, -1)^T; d^2 = (-1, 0)^T; d^3 = (1, 1)^T; \alpha_0 = 1$$

Exemplo de execução do método de busca padrão



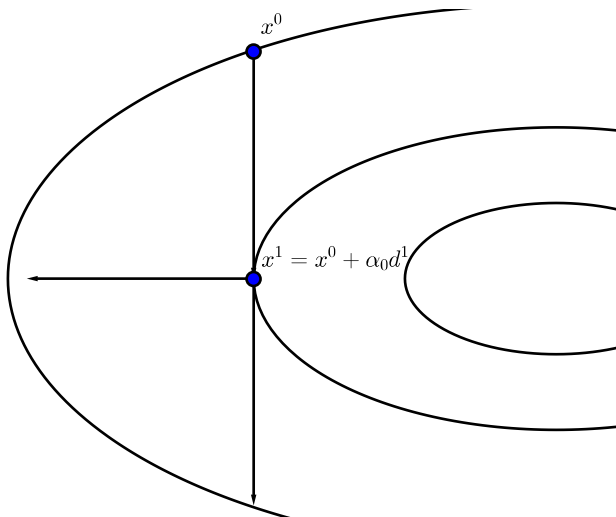
$$d^1 = (0, -1)^T; d^2 = (-1, 0)^T; d^3 = (1, 1)^T; \alpha_1 = 1$$

Exemplo de execução do método de busca padrão



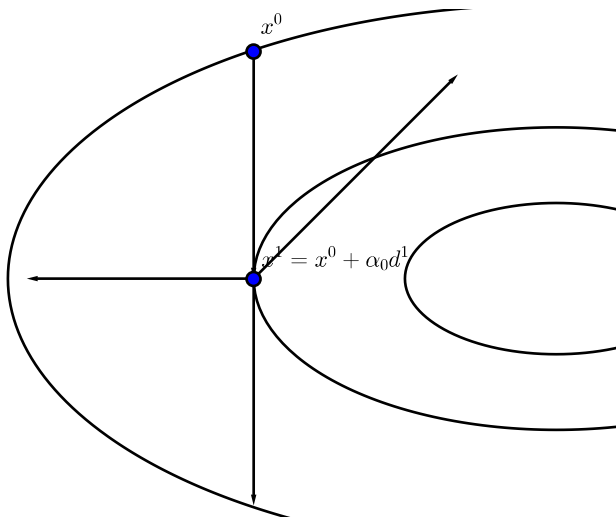
$$d^1 = (0, -1)^T; d^2 = (-1, 0)^T; d^3 = (1, 1)^T; \alpha_1 = 1$$

Exemplo de execução do método de busca padrão



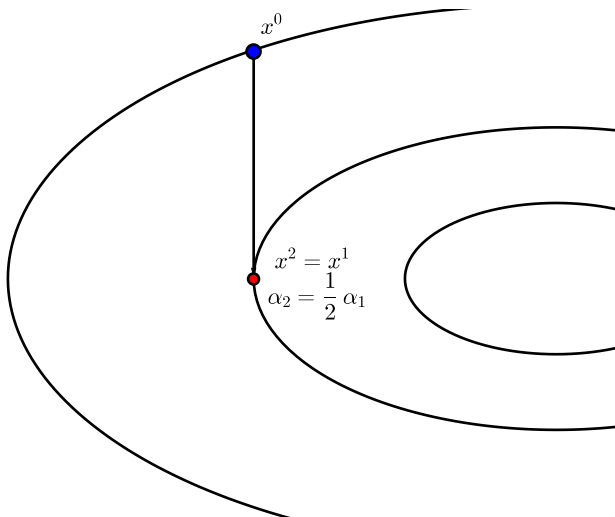
$$d^1 = (0, -1)^T; d^2 = (-1, 0)^T; d^3 = (1, 1)^T; \alpha_1 = 1$$

Exemplo de execução do método de busca padrão



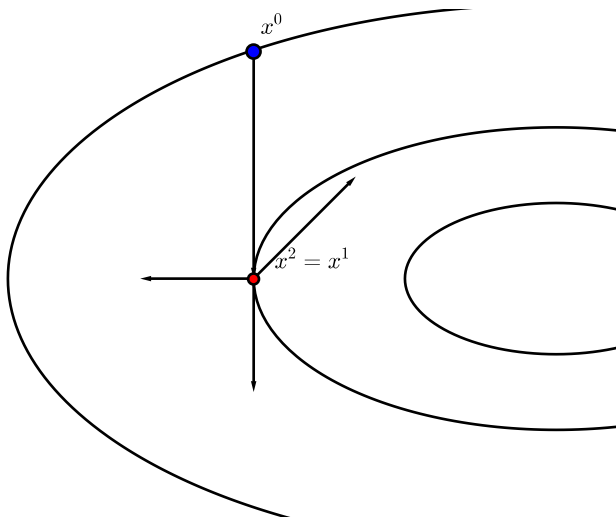
$$d^1 = (0, -1)^T; d^2 = (-1, 0)^T; d^3 = (1, 1)^T; \alpha_1 = 1$$

Exemplo de execução do método de busca padrão



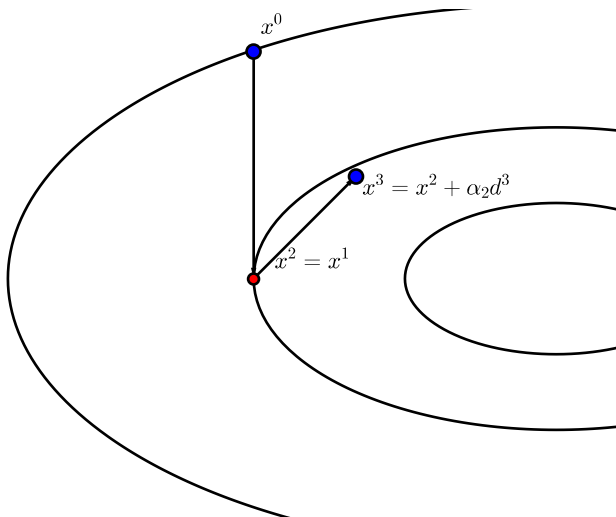
$$d^1 = (0, -1)^T; d^2 = (-1, 0)^T; d^3 = (1, 1)^T; \alpha_2 = 0.5$$

Exemplo de execução do método de busca padrão



$$d^1 = (0, -1)^T; d^2 = (-1, 0)^T; d^3 = (1, 1)^T; \alpha_2 = 0.5$$

Exemplo de execução do método de busca padrão



$$d^1 = (0, -1)^T; d^2 = (-1, 0)^T; d^3 = (1, 1)^T; \alpha_3 = 0.5$$

Gradiente Simplex

- $S = \{y^0, y^1, \dots, y^n\}$ um simplex não degenerado.

Gradiente Simplex

- $S = \{y^0, y^1, \dots, y^n\}$ um simplex não degenerado.

$$L = [y^1 - y^0, \dots, y^n - y^0]$$

Gradiente Simplex

- $S = \{y^0, y^1, \dots, y^n\}$ um simplex não degenerado.

$$L = [y^1 - y^0, \dots, y^n - y^0]$$

$$\delta f(S) = [f(y^1) - f(y^0), f(y^2) - f(y^0), \dots, f(y^n) - f(y^0)]^T$$

Gradiente Simplex

- $S = \{y^0, y^1, \dots, y^n\}$ um simplex não degenerado.

$$L = [y^1 - y^0, \dots, y^n - y^0]$$

$$\delta f(S) = [f(y^1) - f(y^0), f(y^2) - f(y^0), \dots, f(y^n) - f(y^0)]^T$$

Gradiente Simplex

O Gradiente Simplex é definido como a solução do sistema

$$L^T \nabla_S f(y^0) = \delta f(S),$$

esta definição é estendida caso $|S| \neq n + 1$.

Interpretação do Gradiente Simplex

- Da equação do gradiente simplex temos, $\forall i$

Interpretação do Gradiente Simplex

- Da equação do gradiente simplex temos, $\forall i$

$$(y^i - y^0)^\top \nabla_S f(y^0) = f(y^i) - f(y^0),$$

Interpretação do Gradiente Simplex

- Da equação do gradiente simplex temos, $\forall i$

$$(y^i - y^0)^\top \nabla_S f(y^0) = f(y^i) - f(y^0),$$

- daí

$$f(y^i) = f(y^0) + (y^i - y^0)^\top \nabla_S f(y^0),$$

Interpretação do Gradiente Simplex

- Da equação do gradiente simplex temos, $\forall i$

$$(y^i - y^0)^\top \nabla_S f(y^0) = f(y^i) - f(y^0),$$

- daí

$$f(y^i) = f(y^0) + (y^i - y^0)^\top \nabla_S f(y^0),$$

O gradiente simplex de f em y^0 nos dá os coeficientes do polinômio da interpolação linear de f nos pontos de Y .

Conjunto Posicionado

Definição: conjunto posicionado

Seja $\phi = \{\phi_0, \dots, \phi_p\}$ uma base para \mathcal{P}_n^d (polinômios do \mathbf{R}^n com grau menor ou igual a d). Um conjunto $S = \{y^0, y^1, \dots, y^q\}$ é dito posicionado, para o cálculo da interpolação de ordem d , se a matriz

$$M(\phi, Y) = \begin{bmatrix} \phi_0(y^0) & \phi_1(y^0) & \phi_2(y^0) & \dots & \phi_p(y^0) \\ \phi_0(y^1) & \phi_1(y^1) & \phi_2(y^1) & \dots & \phi_p(y^1) \\ \vdots & \vdots & \vdots & & \vdots \\ \phi_0(y^q) & \phi_1(y^q) & \phi_2(y^q) & \dots & \phi_p(y^q) \end{bmatrix},$$

que define a interpolação, tem posto completo.

Λ -posicionamento

Λ -posicionamento para interpolação

Seja $\Lambda > 0$ dado. Seja $\phi = \{\phi_0(x), \phi_1(x), \dots, \phi_p(x)\}$ uma base para \mathcal{P}_n^d .

Um conjunto $Y = \{y^0, y^1, \dots, y^p\}$ é dito Λ -posicionado, para interpolação, em um conjunto B dado se, e somente se, para qualquer $x \in B$ existe $\lambda(x) \in \mathbf{R}^p$ tal que

$$\sum_{i=0}^p \lambda_i(x) \phi(y^i) = \phi(x) \quad \text{com} \quad \|\lambda(x)\|_\infty \leq \Lambda.$$

Propriedades do Gradiente Simplex

$\nabla_S f(y^0)$ como aproximação para $\nabla f(y^0)$

Seja S um simplex Λ -posicionado. Considere a bola fechada $B(y^0, \Delta)$, onde

$$\Delta = \max_{j=1, \dots, n} \|y^j - y^0\|.$$

Se ∇f for Lipschitz contínuo em um conjunto aberto $\Omega \supset B(y^0, \Delta)$, com constante $\gamma > 0$, então

$$\|\nabla f(y^0) - \nabla_S f(y^0)\| \leq \sqrt{n} \frac{\gamma}{2} \Lambda \Delta.$$

Propriedades do Gradiente Simplex

Definição: ϵ -aproximação

Seja $g \in \mathbf{R}^n$ não nulo e $\epsilon \geq 0$. Defina $J^\epsilon(g) = \{i \in \{1, \dots, n\} : |g_i| + \epsilon \geq \|g\|_\infty\}$, e para todo $i \in \{1, \dots, n\}$ defina

$$d_i^\epsilon(g) = \begin{cases} \text{sign}(g_i) & \text{se } i \in J^\epsilon(g) \\ 0 & \text{c.c.} \end{cases}$$

O vetor g é uma ϵ -aproximação para as componentes grandes de um vetor não nulo $v \in \mathbf{R}^n$ se, e somente se, $i \in J^\epsilon(g)$ sempre que $|v_i| = \|v\|_\infty$ e se $\text{sign}(g_i) = \text{sign}(v_i)$ para todo $i \in J^\epsilon(g)$.

Propriedades do Gradiente Simplex

$\nabla_S f(y^0)$ como ϵ -aproximação para $\nabla f(y^0)$

Tomemos $B(y^0, \Delta)$, onde $\Delta = \sigma \alpha \max_{b \in \bar{B}} \|b\|$, $\sigma > 0$ e \bar{B} é uma base positiva para o \mathbb{R}^n . Tomemos $S \subset B(x, \Delta)$ um simplex Λ -posicionado. Se f for continuamente diferenciável em $\Omega \supset B(x, \Delta)$ aberto, ∇f for Lipschitz contínua em Ω com constante $\gamma > 0$, e ainda, se

$$\Delta \leq \frac{\|\nabla f(x)\|_\infty}{\Lambda \sqrt{n} \gamma},$$

então $-\nabla_S f(x)$ é uma ϵ -aproximação para as componentes grandes de $-\nabla f(x)$, onde

$$\epsilon = \Lambda \Delta \sqrt{n} \gamma.$$

Hessiana Simplex

- Tomaremos modelos de interpolação quadrática;

Hessiana Simplex

- Tomaremos modelos de interpolação quadrática;

$$f(y^i) = f(y^0) + (y^i - y^0)^\top \nabla_S f(y^0) + \frac{1}{2} (y^i - y^0)^\top \nabla_S^2 f(y^0) (y^i - y^0)$$

Hessiana Simplex

- Tomaremos modelos de interpolação quadrática;

$$f(y^i) = f(y^0) + (y^i - y^0)^\top \nabla_S f(y^0) + \frac{1}{2} (y^i - y^0)^\top \nabla_S^2 f(y^0) (y^i - y^0)$$

- $\nabla_S^2 f(y^0)$ é chamado Hessiana Simplex;

Hessiana Simplex

- Tomaremos modelos de interpolação quadrática;

$$f(y^i) = f(y^0) + (y^i - y^0)^\top \nabla_S f(y^0) + \frac{1}{2}(y^i - y^0)^\top \nabla_S^2 f(y^0)(y^i - y^0)$$

- $\nabla_S^2 f(y^0)$ é chamado Hessiana Simplex;
- Está unicamente determinada quando o conjunto interpolador possui $(n + 1)(n + 2)/2$ pontos (simétrica).

Propriedades da Hessiana Simplex

Propriedades do modelo determinado

Seja $Y = \{y^0, \dots, y^q\}$, com $q + 1 = (n + 1)(n + 2)/2$, um conjunto posicionado contido na bola $B(y^0, \Delta)$, onde $\Delta = \max_{i=1, \dots, q} \|y^0 - y^i\|$.

Seja f duas vezes diferenciável tal que $\nabla^2 f$ seja Lipschitz contínuo em um conjunto aberto $\Omega \supset B(y^0, \Delta)$, com constante $\gamma > 0$. Então:

- o erro entre a Hessiana simplex e a Hessiana da função satisfaz

$$\|\nabla^2 f(y^0) - \nabla_S^2 f(y^0)\|_F \leq \kappa_H \Delta;$$

- o erro entre o gradiente simplex e o gradiente da função satisfaz

$$\|\nabla f(y^0) - \nabla_S f(y^0)\| \leq \kappa_g \Delta^2,$$

onde κ_H e κ_g dependem de γ e da geometria de Y .

Hessiana Simplex indeterminada

- Se temos menos de $(n + 1)(n + 2)/2$ pontos nosso modelo de interpolação está indeterminado,

Hessiana Simplex indeterminada

- Se temos menos de $(n + 1)(n + 2)/2$ pontos nosso modelo de interpolação está indeterminado,
- Podemos ter infinitas soluções,

Hessiana Simplex indeterminada

- Se temos menos de $(n + 1)(n + 2)/2$ pontos nosso modelo de interpolação está indeterminado,
- Podemos ter infinitas soluções,
- Apresentaremos a sugerida por Custódio, Rocha e Vicente¹

¹A.L. Custódio, H. Rocha e L.N. Vicente, Incorporating minimum Frobenius norm models in direct search, *Computational Optimization and Applications*, pp 265-278, 2010.

Hessiana Simplex indeterminada

- Se temos menos de $(n + 1)(n + 2)/2$ pontos nosso modelo de interpolação está indeterminado,
- Podemos ter infinitas soluções,
- Apresentaremos a sugerida por Custódio, Rocha e Vicente¹

$$\begin{aligned} \min \quad & \frac{1}{4} \|H\|_F \\ & H = H^\top \\ f(y^i) = & f(y^0) + (y^i - y^0)^\top \nabla_S f(y^0) + \\ & + \frac{1}{2} (y^i - y^0)^\top H (y^i - y^0) \end{aligned}$$

¹A.L. Custódio, H. Rocha e L.N. Vicente, Incorporating minimum Frobenius norm models in direct search, *Computational Optimization and Applications*, pp 265-278, 2010.

Propriedades do modelo indeterminado

Erro do gradiente do modelo

Sejam $Y = \{y^0, \dots, y^q\}$, com $q + 1 < (n + 1)(n + 2)/2$ e f uma função continuamente diferenciável em um conjunto aberto na bola $B(y^0, \Delta)$, onde $\Delta = \max_{i=1 \dots q} \|y^0 - y^i\|$, com o gradiente Lipschitz contínuo na bola $B(y^0, \Delta)$. Se Y for Λ -posicionado para interpolação linear então

$$\|\nabla f(y^0) - \nabla_S f(y^0)\| \leq C_q \Lambda [\gamma + \|\nabla_S^2 f(y^0)\|_{\mathbf{F}}] \Delta,$$

onde $C_q > 0$ depende de q e γ é a constante de Lipschitz do gradiente da função.

Propriedades do modelo indeterminado

Λ -posicionamento

Sejam $\Lambda > 0$, um conjunto $B \in \mathbb{R}^n$ e Φ uma base para \mathcal{P}_n^2 . Tomemos o problema

$$\begin{aligned} \min \quad & \|M(\Phi_Q, Y)^\top \lambda(x) - \Phi_Q(x)\|^2 \\ \text{s.a} \quad & M(\Phi_L, Y)^\top \lambda(x) = \Phi_L(x), \end{aligned} \quad (1)$$

onde Φ_L e Φ_Q são, respectivamente, as partes linear e quadrática de Φ e $M(\Phi, Y)$ é a matriz que define o sistema linear da interpolação nos pontos em Y .

Um conjunto Y é dito Λ -posicionado em B , no sentido de norma mínima de Frobenius, se, e somente se, para qualquer $x \in B$ a solução $\lambda(x) \in \mathbb{R}^p$ do problema (1) é tal que

$$\|\lambda(x)\|_\infty \leq \Lambda.$$

Propriedades do modelo indeterminado

Limitante para a Hessiana Simplex

Sejam $Y = \{y^0, \dots, y^q\}$, com $q + 1 < (n + 1)(n + 2)/2$ e f uma função continuamente diferenciável em um conjunto aberto na bola $B(y^0, \Delta)$, onde $\Delta = \min_{i=1, \dots, q} \|y^i - y^0\|$, com o gradiente Lipschitz contínuo na bola $B(y^0, \Delta)$. Se Y for Λ_F -posicionado no sentido da norma mínima de Frobenius então

$$\|\nabla_S^2 f(y^0)\|_F \leq C_{p,q} \gamma \Lambda_F,$$

onde $C_{p,q}$ é uma constante que depende de p e q , e γ é a constante de Lipschitz do gradiente da função.


Propriedades do modelo indeterminado

- Utilizando os dois teoremas anteriores juntos concluímos, que sob, as hipóteses apresentadas, o gradiente do modelo satisfaz

$$\|\nabla f(y^0) - \nabla_S f(y^0)\| \leq C_q \Lambda \gamma [1 + C_{p,q} \Lambda_F] \Delta,$$


Simplex Derivative in Pattern Search Method

- Método desenvolvido por Custódio e Vicente²

²A.L. Custódio e L.N. Vicente, Using sampling and simplex derivatives in pattern search methods, SIAM Journal on Optimization, 18, pp. 537-555, 2007. 


Simplex Derivative in Pattern Search Method

- Método desenvolvido por Custódio e Vicente²
- Baseado no método de Busca Padrão;

²A.L. Custódio e L.N. Vicente, Using sampling and simplex derivatives in pattern search methods, SIAM Journal on Optimization, 18,pp. 537-555, 2007. 


Simplex Derivative in Pattern Search Method

- Método desenvolvido por Custódio e Vicente²
- Baseado no método de Busca Padrão;
- Utiliza derivadas simplex como forma de tentar aumentar a velocidade da convergência;

²A.L. Custódio e L.N. Vicente, Using sampling and simplex derivatives in pattern search methods, SIAM Journal on Optimization, 18, pp. 537-555, 2007. 

Simplex Derivative in Pattern Search Method

- Método desenvolvido por Custódio e Vicente²
- Baseado no método de Busca Padrão;
- Utiliza derivadas simplex como forma de tentar aumentar a velocidade da convergência;
- Informação sobre os pontos armazenada em uma lista \mathcal{L} (com no máximo l_{max} pontos).

²A.L. Custódio e L.N. Vicente, Using sampling and simplex derivatives in pattern search methods, SIAM Journal on Optimization, 18, pp. 537-555, 2007. 

Simplex Derivative in Pattern Search Method

- Método desenvolvido por Custódio e Vicente²
- Baseado no método de Busca Padrão;
- Utiliza derivadas simplex como forma de tentar aumentar a velocidade da convergência;
- Informação sobre os pontos armazenada em uma lista \mathcal{L} (com no máximo l_{max} pontos).
- A derivada simplex é calculada com os pontos da lista.

²A.L. Custódio e L.N. Vicente, Using sampling and simplex derivatives in pattern search methods, SIAM Journal on Optimization, 18,pp. 537-555, 2007.

Cálculo da derivada simplex

- Verificamos se $|\mathcal{L}| \geq n_{min}$.

Cálculo da derivada simplex

- Verificamos se $|\mathcal{L}| \geq n_{min}$.
- Procuramos $S \subseteq \mathcal{L} \cap B(x_k, \Delta_k)$;
 - ▶ $\Delta_k = \sigma \alpha_k \max_{d \in D_k} \|d\|$, $\sigma > 0$.

Cálculo da derivada simplex

- Verificamos se $|\mathcal{L}| \geq n_{min}$.
- Procuramos $S \subseteq \mathcal{L} \cap B(x_k, \Delta_k)$;
 - ▶ $\Delta_k = \sigma \alpha_k \max_{d \in D_k} \|d\|$, $\sigma > 0$.
- Maior possível (com no máximo n_{max} pontos);
- Contenha o iterando atual x_k .
- Λ -posicionado ($\|\Sigma^{-1}\| \leq \Lambda$).
 - ▶ SVD-reduzida $(\frac{L}{\Delta}) = U\Sigma V^T$, $\Lambda > 0$.

Uso das derivadas simplex

- Passo de busca: calcular direção de descida em potencial d_p .
Exemplo: $d_p = -\nabla_S f(x_k)$.

Uso das derivadas simplex

- Passo de busca: calcular direção de descida em potencial d_p .
Exemplo: $d_p = -\nabla_S f(x_k)$.
- Padrão de ordenamento de D_k .

$$\cos(d_p, d_1) \geq \cos(d_p, d_2) \geq \dots \geq \cos(d_p, d_{|D_k|})$$

Uso das derivadas simplex

- Passo de busca: calcular direção de descida em potencial d_p .
Exemplo: $d_p = -\nabla_S f(x_k)$.
- Padrão de ordenamento de D_k .

$$\cos(d_p, d_1) \geq \cos(d_p, d_2) \geq \dots \geq \cos(d_p, d_{|D_k|})$$

- Podar o conjunto D_k .

Uso das derivadas simplex

- Passo de busca: calcular direção de descida em potencial d_p .
Exemplo: $d_p = -\nabla_S f(x_k)$.
- Padrão de ordenamento de D_k .

$$\cos(d_p, d_1) \geq \cos(d_p, d_2) \geq \dots \geq \cos(d_p, d_{|D_k|})$$

- Podar o conjunto D_k .
- Atualização do passo*.

Uso das derivadas simplex

- Passo de busca: calcular direção de descida em potencial d_p .
Exemplo: $d_p = -\nabla_S f(x_k)$.
- Padrão de ordenamento de D_k .

$$\cos(d_p, d_1) \geq \cos(d_p, d_2) \geq \dots \geq \cos(d_p, d_{|D_k|})$$

- Podar o conjunto D_k .
- Atualização do passo*.
- Critério de parada.

Atualização do passo

- Modelo $m_k : \mathbf{R}^n \rightarrow \mathbf{R}$ baseado nas derivadas simplex.
Exemplo: $m_k(x) = f(x_k) + \nabla_S f(x_k)^\top (x - x_k)$

Atualização do passo

- Modelo $m_k : \mathbf{R}^n \rightarrow \mathbf{R}$ baseado nas derivadas simplex.
Exemplo: $m_k(x) = f(x_k) + \nabla_S f(x_k)^\top (x - x_k)$
- Estratégia de descréscimo esperado:

Atualização do passo

- Modelo $m_k : \mathbf{R}^n \rightarrow \mathbf{R}$ baseado nas derivadas simplex.
Exemplo: $m_k(x) = f(x_k) + \nabla_S f(x_k)^\top (x - x_k)$
- Estratégia de descréscimo esperado:
 - ▶ $\rho_k = \frac{f(x_k) - f(x_{k+1})}{m_k(x_k) - m_k(x_{k+1})}$.

Atualização do passo

- Modelo $m_k : \mathbf{R}^n \rightarrow \mathbf{R}$ baseado nas derivadas simplex.
Exemplo: $m_k(x) = f(x_k) + \nabla_S f(x_k)^\top (x - x_k)$
- Estratégia de descréscimo esperado:
 - ▶ $\rho_k = \frac{f(x_k) - f(x_{k+1})}{m_k(x_k) - m_k(x_{k+1})}$.
 - ▶ Se $\rho_k > \gamma_2$, aumentamos o passo;

Atualização do passo

- Modelo $m_k : \mathbf{R}^n \rightarrow \mathbf{R}$ baseado nas derivadas simplex.
Exemplo: $m_k(x) = f(x_k) + \nabla_S f(x_k)^\top (x - x_k)$
- Estratégia de descréscimo esperado:
 - ▶ $\rho_k = \frac{f(x_k) - f(x_{k+1})}{m_k(x_k) - m_k(x_{k+1})}$.
 - ▶ Se $\rho_k > \gamma_2$, aumentamos o passo;
 - ▶ Se $\gamma_1 < \rho_k \leq \gamma_2$, mantemos o passo;

Atualização do passo

- Modelo $m_k : \mathbf{R}^n \rightarrow \mathbf{R}$ baseado nas derivadas simplex.
Exemplo: $m_k(x) = f(x_k) + \nabla_S f(x_k)^\top (x - x_k)$
- Estratégia de descréscimo esperado:
 - ▶ $\rho_k = \frac{f(x_k) - f(x_{k+1})}{m_k(x_k) - m_k(x_{k+1})}$.
 - ▶ Se $\rho_k > \gamma_2$, aumentamos o passo;
 - ▶ Se $\gamma_1 < \rho_k \leq \gamma_2$, mantemos o passo;
 - ▶ Se $\rho_k \leq \gamma_1$, diminuimos o passo.

Atualização do passo

- Modelo $m_k : \mathbf{R}^n \rightarrow \mathbf{R}$ baseado nas derivadas simplex.
Exemplo: $m_k(x) = f(x_k) + \nabla_S f(x_k)^\top (x - x_k)$
- Estratégia de descréscimo esperado:
 - ▶ $\rho_k = \frac{f(x_k) - f(x_{k+1})}{m_k(x_k) - m_k(x_{k+1})}$.
 - ▶ Se $\rho_k > \gamma_2$, aumentamos o passo;
 - ▶ Se $\gamma_1 < \rho_k \leq \gamma_2$, mantemos o passo;
 - ▶ Se $\rho_k \leq \gamma_1$, diminuimos o passo.
 - ▶ $\gamma_2 > \gamma_1 \geq 0$.

Parâmetros usados

- Lista limitada em $l_{max} = (n + 1)(n + 2)$ pontos.
- Gradiente simplex com $n_{min} = n_{max} = n + 1$;

Parâmetros usados

- Lista limitada em $l_{max} = (n + 1)(n + 2)$ pontos.
- Gradiente simplex com $n_{min} = n_{max} = n + 1$;
- $\Lambda = 100$ e $\sigma = 2$;

Parâmetros usados

- Lista limitada em $l_{max} = (n + 1)(n + 2)$ pontos.
- Gradiente simplex com $n_{min} = n_{max} = n + 1$;
- $\Lambda = 100$ e $\sigma = 2$;
- $\phi = 2$ e $\theta = \frac{1}{2}$.

Parâmetros usados

- Lista limitada em $l_{max} = (n + 1)(n + 2)$ pontos.
- Gradiente simplex com $n_{min} = n_{max} = n + 1$;
- $\Lambda = 100$ e $\sigma = 2$;
- $\phi = 2$ e $\theta = \frac{1}{2}$.
- Os critérios de parada foram:
 - ▶ $\alpha_k \leq 10^{-6}$.
 - ▶ $\#f_{eval} \geq 10^6$.

Parâmetros usados

- Busca: sempre que possível criamos um modelo quadrático com os pontos em \mathcal{L} , se não for possível, atualizamos a parte linear do modelo. Resolvemos problema

$$\begin{array}{ll} \min & m_k(x) \\ \text{s.a} & x \in B(x_k, \Delta_k) \end{array}$$

onde $\Delta_k = \sigma \alpha_k \min_{d \in D_k} \|d\|$.

Parâmetros usados

- Busca: sempre que possível criamos um modelo quadrático com os pontos em \mathcal{L} , se não for possível, atualizamos a parte linear do modelo. Resolvemos problema

$$\begin{aligned} \min \quad & m_k(x) \\ \text{s.a} \quad & x \in B(x_k, \Delta_k) \end{aligned}$$

onde $\Delta_k = \sigma \alpha_k \min_{d \in D_k} \|d\|$.

Se \mathcal{L} possuir mais de $(n+1)(n+2)/2$ elementos, então fazemos uma regressão ao invés de uma interpolação.

Parâmetros usados

- Busca: sempre que possível criamos um modelo quadrático com os pontos em \mathcal{L} , se não for possível, atualizamos a parte linear do modelo. Resolvemos problema

$$\begin{aligned} \min \quad & m_k(x) \\ \text{s.a} \quad & x \in B(x_k, \Delta_k) \end{aligned}$$

onde $\Delta_k = \sigma \alpha_k \min_{d \in D_k} \|d\|$.

Se \mathcal{L} possuir mais de $(n+1)(n+2)/2$ elementos, então fazemos uma regressão ao invés de uma interpolação.

- D_k ordenado pelo ângulo com $\nabla_S f(x_k)$.
Caso este não esteja disponível, começa pelo último vetor que ofereceu decréscimo.

Perfil de desempenho

- P : conjunto de problemas, m : medida de desempenho.

Perfil de desempenho

- P : conjunto de problemas, m : medida de desempenho.
- $\rho_s : \mathcal{R} \rightarrow [0, 1]$, taxa de desempenho do método s .
Probabilidade do método s resolver algum problema.

Perfil de desempenho

- P : conjunto de problemas, m : medida de desempenho.
- $\rho_s : \mathcal{R} \rightarrow [0, 1]$, taxa de desempenho do método s .
Probabilidade do método s resolver algum problema.
- $\rho_s(1)$: eficiência do método s em relação a m .

Perfil de desempenho

- P : conjunto de problemas, m : medida de desempenho.
- $\rho_s : \mathcal{R} \rightarrow [0, 1]$, taxa de desempenho do método s .
Probabilidade do método s resolver algum problema.
- $\rho_s(1)$: eficiência do método s em relação a m .
- $\bar{t} = \min_{s \in S} \{t_s : \rho_s(t_s) = 1\}$: nos mostra qual o método mais robusto.

Perfil de desempenho

- P : conjunto de problemas, m : medida de desempenho.
- $\rho_s : \mathcal{R} \rightarrow [0, 1]$, taxa de desempenho do método s .
Probabilidade do método s resolver algum problema.
- $\rho_s(1)$: eficiência do método s em relação a m .
- $\bar{t} = \min_{s \in S} \{t_s : \rho_s(t_s) = 1\}$: nos mostra qual o método mais robusto.
- No nosso caso $m = \#f_eval$.

Conjunto de problemas

- Todos os problemas de Moré, Garbow e Hillstrom³.

³J.J. Moré, B.S. Garbow e K.E. Hillstrom, Testing unconstrained optimization software, ACM Transactions on Mathematical Software, 7, pp. 17-41, 1981

Conjunto de problemas

- Todos os problemas de Moré, Garbow e Hillstrom³.
- Problemas de minimização irrestrita, soma de quadrados de funções.
- Os problemas de tamanho variável foram resolvidos com dimensão 12 e 20.

³J.J. Moré, B.S. Garbow e K.E. Hillstrom, Testing unconstrained optimization software, ACM Transactions on Mathematical Software, 7, pp. 17-41, 1981

Definição: Resolver problemas

Sejam $\theta_p(s)$ o menor valor de função obtido pelo método s , em um conjunto de métodos S , ao resolver o problema p e $\hat{\theta}_p = \min_{\hat{s} \in S} \theta_p(\hat{s})$. Então se o método atinge um passo com tamanho menor que 10^{-6} em menos de 10^6 avaliações de função, e

$$\theta_p(s) \leq 1,25\hat{\theta}_p \quad \text{se} \quad \hat{\theta}_p > 0;$$

diz-se que o método s resolve o problema p , em relação ao conjunto S .

Métodos comparados

NM

BP1

BP2

SID-PSM1

SID-PSM2

Métodos comparados

NM Nelder-Mead, implementado na função `fminsearch`.

BP1

BP2

SID-PSM1

SID-PSM2

Métodos comparados

NM Nelder-Mead, implementado na função `fminsearch`.

BP1 Busca Padrão com base pos. minimal com ângulos de tam. uniforme entre seus vetores.

BP2

SID-PSM1

SID-PSM2

Métodos comparados

NM Nelder-Mead, implementado na função `fminsearch`.

BP1 Busca Padrão com base pos. minimal com ângulos de tam. uniforme entre seus vetores.

BP2 Busca Padrão, $D = \{e, -e, I_{n \times n}, -I_{n \times n}\}$.

SID-PSM1

SID-PSM2

Métodos comparados

NM Nelder-Mead, implementado na função `fminsearch`.

BP1 Busca Padrão com base pos. minimal com ângulos de tam. uniforme entre seus vetores.

BP2 Busca Padrão, $D = \{e, -e, I_{n \times n}, -I_{n \times n}\}$.

SID-PSM1 SID-PSM, direção mais promissora, $D = \{e, -e, I_{n \times n}, -I_{n \times n}\}$, decréscimo esperado com $\gamma_1 = 0,25$ e $\gamma_2 = 0,75$.

SID-PSM2

Métodos comparados

NM Nelder-Mead, implementado na função `fminsearch`.

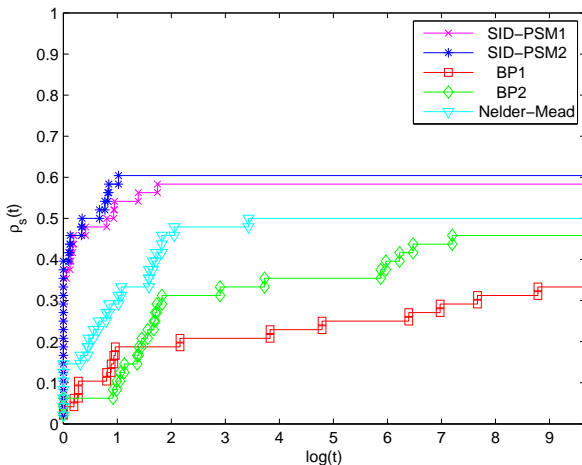
BP1 Busca Padrão com base pos. minimal com ângulos de tam. uniforme entre seus vetores.

BP2 Busca Padrão, $D = \{e, -e, I_{n \times n}, -I_{n \times n}\}$.

SID-PSM1 SID-PSM, direção mais promissora, $D = \{e, -e, I_{n \times n}, -I_{n \times n}\}$, decréscimo esperado com $\gamma_1 = 0,25$ e $\gamma_2 = 0,75$.

SID-PSM2 SID-PSM, sem poda, com base pos. minimal com ângulos de tam. uniforme entre seus vetores, aumenta o passo quando a mesma direção oferece decréscimo duas iterações seguidas.

Perfil de desempenho entre os métodos



Perfil de desempenho entre os métodos

- É clara a superioridade do SID-PSM tanto em robustez quanto em eficiência sobre os outros métodos.

Perfil de desempenho entre os métodos

- É clara a superioridade do SID-PSM tanto em robustez quanto em eficiência sobre os outros métodos.
- Tanto em termos de robustez quanto de eficiência o método SID-PSM2 se demonstrou um pouco superior ao método SID-PSM1.


Problema de parâmetros ótimos de algoritmos

- Problema apresentado por Audet e Orban⁴

⁴C. Audet e D. Orban, Finding optimal algorithmic parameters using derivative-free optimization, *SIAM Journal on Optimization*, 3, pp. 642-664, 2006.

Problema de parâmetros ótimos de algoritmos

- Problema apresentado por Audet e Orban⁴
- Métodos numéricos dependem de parâmetros que podem influenciar muito em sua eficiência.

⁴C. Audet e D. Orban, Finding optimal algorithmic parameters using derivative-free optimization, *SIAM Journal on Optimization*, 3, pp. 642-664, 2006. 

Problema de parâmetros ótimos de algoritmos

- Problema apresentado por Audet e Orban⁴
- Métodos numéricos dependem de parâmetros que podem influenciar muito em sua eficiência.
- Esses parâmetros nem sempre são bem estudados antes de serem definidos.

⁴C. Audet e D. Orban, Finding optimal algorithmic parameters using derivative-free optimization, *SIAM Journal on Optimization*, 3, pp. 642-664, 2006.


Problema de parâmetros ótimos de algoritmos

- Problema apresentado por Audet e Orban⁴
- Métodos numéricos dependem de parâmetros que podem influenciar muito em sua eficiência.
- Esses parâmetros nem sempre são bem estudados antes de serem definidos.
- Se definirmos uma medida de desempenho podemos otimizar o método em relação a estes parâmetros.

⁴C. Audet e D. Orban, Finding optimal algorithmic parameters using derivative-free optimization, *SIAM Journal on Optimization*, 3, pp. 642-664, 2006.

Problema de parâmetros ótimos de algoritmos

- Problema apresentado por Audet e Orban⁴
- Métodos numéricos dependem de parâmetros que podem influenciar muito em sua eficiência.
- Esses parâmetros nem sempre são bem estudados antes de serem definidos.
- Se definirmos uma medida de desempenho podemos otimizar o método em relação a estes parâmetros.
- Esse é um problema ideal para métodos sem derivadas, pois não temos acesso à função objetivo.

⁴C. Audet e D. Orban, Finding optimal algorithmic parameters using derivative-free optimization, SIAM Journal on Optimization, 3, pp. 642-664, 2006. 

Definindo o problema

- s : um método matemático.

Definindo o problema

- s : um método matemático.
- Depende dos parâmetros x_i , $i = 1, \dots, n$.

Definindo o problema

- s : um método matemático.
- Depende dos parâmetros x_i , $i = 1, \dots, n$.
- Ω : conjunto dos valores permitidos para os x_i 's.

Definindo o problema

- s : um método matemático.
- Depende dos parâmetros $x_i, i = 1, \dots, n$.
- Ω : conjunto dos valores permitidos para os x_i 's.
- m : medida de desempenho.

Definindo o problema

- s : um método matemático.
- Depende dos parâmetros $x_i, i = 1, \dots, n$.
- Ω : conjunto dos valores permitidos para os x_i 's.
- m : medida de desempenho.
- P : conjunto de problemas-teste.

Definindo o problema

- s : um método matemático.
- Depende dos parâmetros $x_i, i = 1, \dots, n$.
- Ω : conjunto dos valores permitidos para os x_i 's.
- m : medida de desempenho.
- P : conjunto de problemas-teste.
- Definimos $f : \mathbb{R}^n \rightarrow \mathbb{R}$.

Definindo o problema

- s : um método matemático.
- Depende dos parâmetros x_i , $i = 1, \dots, n$.
- Ω : conjunto dos valores permitidos para os x_i 's.
- m : medida de desempenho.
- P : conjunto de problemas-teste.
- Definimos $f : \mathbb{R}^n \rightarrow \mathbb{R}$.

Se $x \in \Omega$, $f(x)$ = desempenho de s para resolver P .

$$\text{Caso contrário, } f(x) = M \geq \sup_{x \in \Omega} f(x).$$

SID-PSM1

- Decidimos otimizar o método SID-PSM1.

SID-PSM1

- Decidimos otimizar o método SID-PSM1.
- P : subconjunto com 22 problemas do pacote CUTEr⁵.

⁵<http://www.mcs.anl.gov/~more/dfo/> (Moré e Wild)

SID-PSM1

- Decidimos otimizar o método SID-PSM1.
- P : subconjunto com 22 problemas do pacote CUTER⁵.
- Os parâmetros variáveis:
 - ▶ γ_1 (medida de qualidade do modelo),
 - ▶ γ_2 (medida de qualidade do modelo),
 - ▶ ϕ (parâmetro de crescimento do passo),
 - ▶ θ (parâmetro de decrescimento do passo),
 - ▶ $\bar{\Lambda}$, onde $\bar{\Lambda} = \frac{\Lambda}{100}$
 - ▶ σ .

⁵<http://www.mcs.anl.gov/~more/dfo/> (Moré e Wild)

SID-PSM1

- Decidimos otimizar o método SID-PSM1.
- P : subconjunto com 22 problemas do pacote CUTEr⁵.
- Os parâmetros variáveis:
 - ▶ γ_1 (medida de qualidade do modelo),
 - ▶ γ_2 (medida de qualidade do modelo),
 - ▶ ϕ (parâmetro de crescimento do passo),
 - ▶ θ (parâmetro de decrescimento do passo),
 - ▶ $\bar{\Lambda}$, onde $\bar{\Lambda} = \frac{\Lambda}{100}$
 - ▶ σ .

$$\Omega = \left\{ (\gamma_1, \gamma_2, \phi, \theta, \bar{\Lambda}, \sigma) : \begin{array}{l} \gamma_1 < \gamma_2 \\ 0 \leq \theta < 1, \\ \phi \geq 1, \\ \gamma_1, \gamma_2, \sigma, \bar{\Lambda} \geq 0 \end{array} \right\}.$$

⁵<http://www.mcs.anl.gov/~more/dfo/> (Moré e Wild)

Função objetivo

- Definimos $f : \mathbb{R}^n \rightarrow \mathbb{R}$, tal que

$$f(x) = \begin{cases} \sum_{i \in P} f_i(x) & \text{se } x \in \Omega \\ M & \text{c. c.} \end{cases},$$

$f_i(x) = \#_{f_eval}$ realizadas pelo SID-PSM1 ao resolver o problema i usando os parâmetros x .

Função objetivo

- Definimos $f : \mathbb{R}^n \rightarrow \mathbb{R}$, tal que

$$f(x) = \begin{cases} \sum_{i \in P} f_i(x) & \text{se } x \in \Omega \\ M & \text{c. c.} \end{cases},$$

$f_i(x) = \#_{f_eval}$ realizadas pelo SID-PSM1 ao resolver o problema i usando os parâmetros x .

- $\sup_{x \in \Omega} f_i(x) = 10^6 \Rightarrow \sup_{x \in \Omega} f(x) = |P|10^6$.

Função objetivo

- Definimos $f : \mathbb{R}^n \rightarrow \mathbb{R}$, tal que

$$f(x) = \begin{cases} \sum_{i \in P} f_i(x) & \text{se } x \in \Omega \\ M & \text{c. c.} \end{cases},$$

$f_i(x) = \#_{f_eval}$ realizadas pelo SID-PSM1 ao resolver o problema i usando os parâmetros x .

- $\sup_{x \in \Omega} f_i(x) = 10^6 \Rightarrow \sup_{x \in \Omega} f(x) = |P|10^6$.
- Escolhemos $M = (|P| + 1)10^6$.

Qualidade da solução

- g_i^0 : menor valor de função obtido pelo SID-PSM1, aplicado ao problema i com os parâmetros originais.

Qualidade da solução

- g_i^0 : menor valor de função obtido pelo SID-PSM1, aplicado ao problema i com os parâmetros originais.
- $g_i(x)$: menor valor de função obtido pelo SID-PSM1, aplicado ao problema i com os parâmetros x .

Qualidade da solução

- g_i^0 : menor valor de função obtido pelo SID-PSM1, aplicado ao problema i com os parâmetros originais.
- $g_i(x)$: menor valor de função obtido pelo SID-PSM1, aplicado ao problema i com os parâmetros x .
- Esperamos $g_i(x) \leq 1,25g_i^0, \forall i \in P$

Qualidade da solução

- g_i^0 : menor valor de função obtido pelo SID-PSM1, aplicado ao problema i com os parâmetros originais.
- $g_i(x)$: menor valor de função obtido pelo SID-PSM1, aplicado ao problema i com os parâmetros x .
- Esperamos $g_i(x) \leq 1,25g_i^0, \forall i \in P$
- $g_i^+(x) = \max\{g_i(x) - 1,25g_i^0, 0\}$

Modificando f Função \bar{f}

$$\bar{f}(x) = \begin{cases} \sum_{i \in P} f_i(x) + \frac{\|g^+(x)\|}{\|g^0\|} M & \text{se } x \in \Omega \\ M & \text{c. c.} \end{cases} .$$

Ponto inicial

- Aplicamos SID-PSM1 e SID-PSM2 à função \bar{f} ;

Ponto inicial

- Aplicamos SID-PSM1 e SID-PSM2 à função \bar{f} ;
-

$$x^0 = \begin{bmatrix} \gamma_1 \\ \gamma_2 \\ \phi \\ \theta \\ \bar{\lambda} \\ \sigma \end{bmatrix} = \begin{bmatrix} 0,25 \\ 0,75 \\ 2 \\ 0,5 \\ 1 \\ 2 \end{bmatrix},$$

Ponto inicial

- Aplicamos SID-PSM1 e SID-PSM2 à função \bar{f} ;



$$x^0 = \begin{bmatrix} \gamma_1 \\ \gamma_2 \\ \phi \\ \theta \\ \bar{\lambda} \\ \sigma \end{bmatrix} = \begin{bmatrix} 0,25 \\ 0,75 \\ 2 \\ 0,5 \\ 1 \\ 2 \end{bmatrix},$$

- $\bar{f}(x^0) = 213551$

Resultados

- Saídas:

$$x_1^* = \begin{bmatrix} 1,25047302246094 \\ 4,00047302246094 \\ 4,00047302246094 \\ 0,500473022460938 \\ 1,00041198730469 \\ 6,00047302246094 \end{bmatrix}$$

$$\text{e } x_2^* = \begin{bmatrix} 0,356119791666667 \\ 0,727660636235692 \\ 2,24152294576982 \\ 0,227170815547733 \\ 0,448801810194877 \\ 2 \end{bmatrix},$$

Resultados

- Saídas:

$$x_1^* = \begin{bmatrix} 1,25047302246094 \\ 4,00047302246094 \\ 4,00047302246094 \\ 0,500473022460938 \\ 1,00041198730469 \\ 6,00047302246094 \end{bmatrix}$$

$$\text{e } x_2^* = \begin{bmatrix} 0,356119791666667 \\ 0,727660636235692 \\ 2,24152294576982 \\ 0,227170815547733 \\ 0,448801810194877 \\ 2 \end{bmatrix},$$

- Custo:

SID-PSM1 : 175 avaliações;

SID-PSM2 : 199 avaliações.

Resultados

- Saídas:

$$x_1^* = \begin{bmatrix} 1,25047302246094 \\ 4,00047302246094 \\ 4,00047302246094 \\ 0,500473022460938 \\ 1,00041198730469 \\ 6,00047302246094 \end{bmatrix} \quad \text{e} \quad x_2^* = \begin{bmatrix} 0,356119791666667 \\ 0,727660636235692 \\ 2,24152294576982 \\ 0,227170815547733 \\ 0,448801810194877 \\ 2 \end{bmatrix},$$

- Custo:

SID-PSM1 : 175 avaliações;

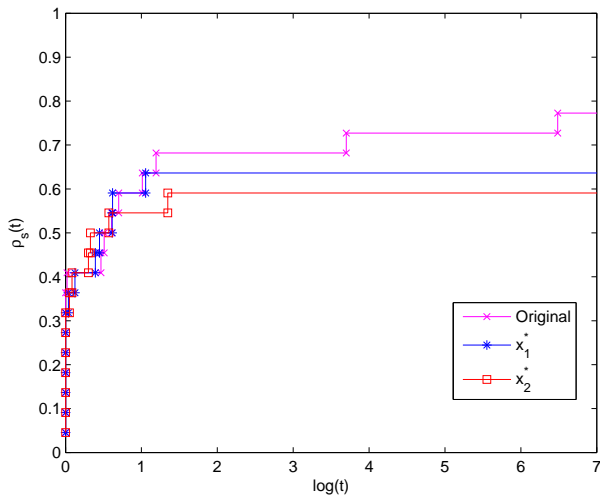
SID-PSM2 : 199 avaliações.

- Valor da função:

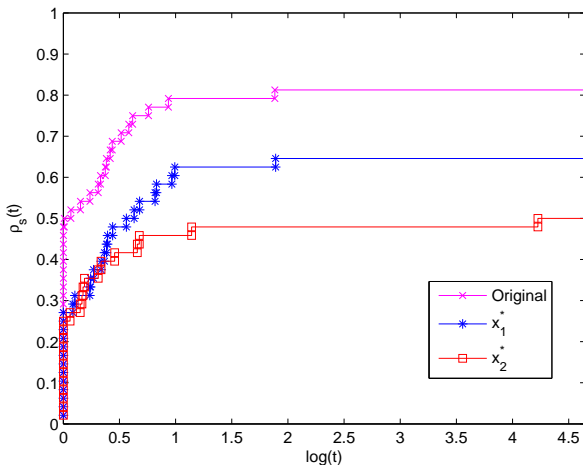
SID-PSM1 : $\bar{f}(x_1^*) = 20782,8$;

SID-PSM2 : $\bar{f}(x_2^*) = 24324,3$;

Perfil de desempenho usando problemas em P



Perfil de desempenho usando problemas de MGH



Análises

- Diminuimos o valor da função objetivo, mas não melhoramos eficiência em P ;

Análises

- Diminuimos o valor da função objetivo, mas não melhoramos eficiência em P ;
- Perda de eficiência e de robustez em ambos os casos;

Análises

- Diminuimos o valor da função objetivo, mas não melhoramos eficiência em P ;
- Perda de eficiência e de robustez em ambos os casos;
- Valor da função objetivo não inteiro indica que pelo menos um problema não está sendo resolvido com a precisão desejada;

Análises

- Diminuimos o valor da função objetivo, mas não melhoramos eficiência em P ;
- Perda de eficiência e de robustez em ambos os casos;
- Valor da função objetivo não inteiro indica que pelo menos um problema não está sendo resolvido com a precisão desejada;
- Repensar na função objetivo.

Repensando função objetivo

- Fazemos a otimização utilizando o conjunto de problemas, a comparação é feita problema-a-problema;

Repensando função objetivo

- Fazemos a otimização utilizando o conjunto de problemas, a comparação é feita problema-a-problema;
- Se um problema gasta 10^4 avaliações de função e outro gasta 10^2 , o segundo pode aumentar o número de avaliações que realiza que não irá interferir tanto no valor final da função objetivo;

Repensando função objetivo

- Fazemos a otimização utilizando o conjunto de problemas, a comparação é feita problema-a-problema;
- Se um problema gasta 10^4 avaliações de função e outro gasta 10^2 , o segundo pode aumentar o número de avaliações que realiza que não irá interferir tanto no valor final da função objetivo;
- Para resolver este problema podemos impor pesos a cada um dos problemas;

Repensando função objetivo

- Sejam f_i^0 o número de avaliações de função necessárias para o método SID-PSM1, utilizando os parâmetros originais, resolver o problema i ;

Repensando função objetivo

- Sejam f_i^0 o número de avaliações de função necessárias para o método SID-PSM1, utilizando os parâmetros originais, resolver o problema i ;
- Assim nossa nova função passa a ser

$$\hat{f}_{\bar{\Omega}}(x) = \begin{cases} \sum_{i \in P} \frac{f_i(x)}{f_i^0} & \text{se } x \in \bar{\Omega}, \\ (|P| + 1) & \text{c. c.} \end{cases},$$

Repensando Conjunto de Parâmetros válidos

- Ω original foi limitado apenas pelos pontos onde podemos rodar o método;

Repensando Conjunto de Parâmetros válidos

- Ω original foi limitado apenas pelos pontos onde podemos rodar o método;
- iremos impor que os pontos de $\bar{\Omega}$ devem satisfazer algumas condições sobre a robustez;

Repensando Conjunto de Parâmetros válidos

- Ω original foi limitado apenas pelos pontos onde podemos rodar o método;
- iremos impor que os pontos de $\bar{\Omega}$ devem satisfazer algumas condições sobre a robustez;
- $x \in \bar{\Omega}$ se $x \in \Omega$ e se ao rodar todos os 22 problemas em P no máximo 1 não satisfaça a condição $g_i(x) \leq 1,25g_i^0$.

Novos resultados

- Aplicamos SID-PSM1 e SID-PSM2 à função \hat{f}_{Ω} ;

Novos resultados

- Aplicamos SID-PSM1 e SID-PSM2 à função $\hat{f}_{\bar{\Omega}}$;
-

$$x^0 = \begin{bmatrix} \gamma_1 \\ \gamma_2 \\ \phi \\ \theta \\ \bar{\lambda} \\ \sigma \end{bmatrix} = \begin{bmatrix} 0,25 \\ 0,75 \\ 2 \\ 0,5 \\ 1 \\ 2 \end{bmatrix},$$

Novos resultados

- Aplicamos SID-PSM1 e SID-PSM2 à função $\hat{f}_{\bar{\Omega}}$;



$$x^0 = \begin{bmatrix} \gamma_1 \\ \gamma_2 \\ \phi \\ \theta \\ \bar{\lambda} \\ \sigma \end{bmatrix} = \begin{bmatrix} 0,25 \\ 0,75 \\ 2 \\ 0,5 \\ 1 \\ 2 \end{bmatrix},$$

- $\bar{f}(x^0) = 22$

Resultados

- Saídas:

$$x_3^* = \begin{bmatrix} 0,1015625000 \\ 0,7509765625 \\ 2 \\ 0,5 \\ 1 \\ 2 \end{bmatrix}$$

$$\text{e } x_4^* = \begin{bmatrix} 0,005208333333 \\ 0,71918708446 \\ 2 \\ 0,5 \\ 1 \\ 2 \end{bmatrix},$$

Resultados

- Saídas:

$$x_3^* = \begin{bmatrix} 0,1015625000 \\ 0,7509765625 \\ 2 \\ 0,5 \\ 1 \\ 2 \end{bmatrix}$$

$$\text{e } x_4^* = \begin{bmatrix} 0,00520833333 \\ 0,71918708446 \\ 2 \\ 0,5 \\ 1 \\ 2 \end{bmatrix},$$

- Custo:

SID-PSM1 : 165 avaliações;

SID-PSM2 : 172 avaliações.

Resultados

- Saídas:

$$x_3^* = \begin{bmatrix} 0,1015625000 \\ 0,7509765625 \\ 2 \\ 0,5 \\ 1 \\ 2 \end{bmatrix}$$

$$\text{e } x_4^* = \begin{bmatrix} 0,00520833333 \\ 0,71918708446 \\ 2 \\ 0,5 \\ 1 \\ 2 \end{bmatrix},$$

- Custo:

SID-PSM1 : 165 avaliações;

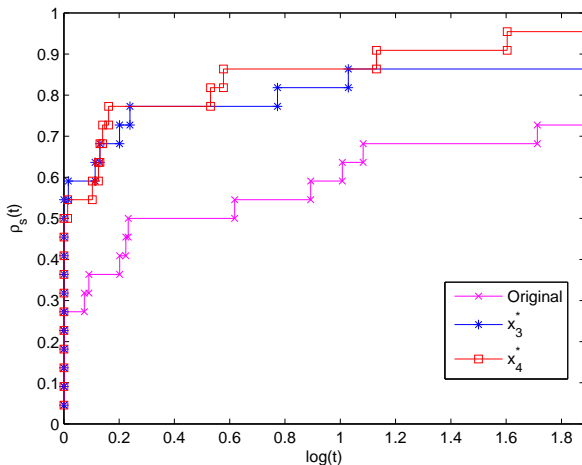
SID-PSM2 : 172 avaliações.

- Valor da função:

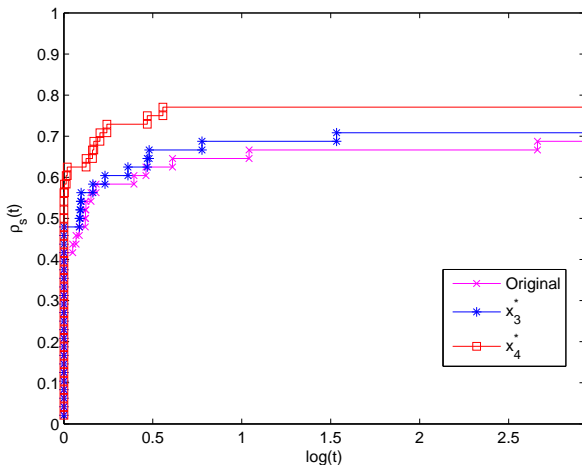
SID-PSM1 : $\hat{f}_{\Omega}(x_1^*) = 17,029$;

SID-PSM2 : $\hat{f}_{\Omega}(x_2^*) = 18,601$;

Perfil de desempenho usando problemas em P



Perfil de desempenho usando problemas de MGH



Análises

- Soluções foram bastante satisfatórias

Análises

- Soluções foram bastante satisfatórias
- Novos métodos mais eficientes e mais robustos que o original.

Análises

- Soluções foram bastante satisfatórias
- Novos métodos mais eficientes e mais robustos que o original.
- Isso mostra o grande potencial desta técnica para obter “novos métodos” melhores e mais rápidos.

Conclusões

- SID-PSM superior aos métodos Nelder-Mead e Busca Padrão;

Conclusões

- SID-PSM superior aos métodos Nelder-Mead e Busca Padrão;
- Dimensão pequena: iteração muito cara:

Conclusões

- SID-PSM superior aos métodos Nelder-Mead e Busca Padrão;
- Dimensão pequena: iteração muito cara;
- Ordem de n^2 verificações de Λ -posicionamento.

Conclusões - Otimização de parâmetros de algoritmos

- Utilizamos o método SID-PSM pois não temos acesso à função objetivo, logo não temos acesso também às derivadas da função;

Conclusões - Otimização de parâmetros de algoritmos

- Utilizamos o método SID-PSM pois não temos acesso à função objetivo, logo não temos acesso também às derivadas da função;
- Nossa função é muito cara, porém temos poucas variáveis (6), logo o custo da iteração do SID-PSM passa a ser irrelevante;

Conclusões - Otimização de parâmetros de algoritmos

- Utilizamos o método SID-PSM pois não temos acesso à função objetivo, logo não temos acesso também às derivadas da função;
- Nossa função é muito cara, porém temos poucas variáveis (6), logo o custo da iteração do SID-PSM passa a ser irrelevante;
 - 1- Resultados insatisfatórios;
 - 1- Possíveis motivos: simplicidade da função e forma de penalização.

Conclusões - Otimização de parâmetros de algoritmos

- Utilizamos o método SID-PSM pois não temos acesso à função objetivo, logo não temos acesso também às derivadas da função;
- Nossa função é muito cara, porém temos poucas variáveis (6), logo o custo da iteração do SID-PSM passa a ser irrelevante;
 - 1- Resultados insatisfatórios;
 - 1- Possíveis motivos: simplicidade da função e forma de penalização.
 - 2- Resultados satisfatórios;
 - 2- Tanto a eficiência quanto a robustez melhoraram.

Principais Referências

- C. Audet e D. Orban, Finding optimal algorithmic parameters using derivative-free optimization, SIAM Journal on Optimization, 3, pp. 642-664, 2006.
- A.L. Custódio e L.N. Vicente, Using sampling and simplex derivatives in pattern search methods, SIAM Journal on Optimization, 18, pp. 537-555, 2007.
- J.C. Lagarias, J.A. Reeds, M.H. Wright e P.E. Wright, Convergence properties of the Nelder- Mead simplex algorithm in low dimensions, SIAM Journal on Optimization 9, pp. 112-147, 1998
- J.J. Moré, B.S. Garbow e K.E. Hillstrom, Testing unconstrained optimization software, ACM Transactions on Mathematical Software, 7, pp. 17-41, 1981
- V. Torczon, On the convergence of pattern search algorithm, SIAM Journal on Optimization, 7, pp. 1-25, 1997.

OBRIGADO!

Base Pos. Min. com ângulos de tam. uniformes

- Queremos gerar $D \in \mathbf{R}^{n \times n+1}$, tal que

$$d_i^\top d_j = \alpha \quad \forall i \neq j$$

e

$$\|d_i\| = 1 \quad \forall i = 1, \dots, n+1.$$

Base Pos. Min. com ângulos de tam. uniformes

- Queremos gerar $D \in \mathbf{R}^{n \times n+1}$, tal que

$$d_i^\top d_j = \alpha \quad \forall i \neq j$$

e

$$\|d_i\| = 1 \quad \forall i = 1, \dots, n+1.$$

- Neste caso é possível mostrar que o único valor possível para α , de modo que D não possua todas as colunas iguais é

$$\alpha = -\frac{1}{n}$$

Base Pos. Min. com ângulos de tam. uniformes

- Primeiramente encontraremos um conjunto V , tal que

$$V^T V = \begin{bmatrix} 1 & -1/n & -1/n & \dots & -1/n \\ -1/n & 1 & -1/n & \dots & -1/n \\ \vdots & \vdots & \ddots & \vdots & -1/n \\ -1/n & -1/n & -1/n & \dots & 1 \end{bmatrix} = A \in \mathbf{R}^{n \times n}$$

Base Pos. Min. com ângulos de tam. uniformes

- Primeiramente encontraremos um conjunto V , tal que

$$V^T V = \begin{bmatrix} 1 & -1/n & -1/n & \dots & -1/n \\ -1/n & 1 & -1/n & \dots & -1/n \\ \vdots & \vdots & \ddots & \vdots & -1/n \\ -1/n & -1/n & -1/n & \dots & 1 \end{bmatrix} = A \in \mathbf{R}^{n \times n}$$

- Tomando G a decomposição de Cholesky de A temos

$$GG^T = A,$$

daí se tomarmos $V = G^T$ temos as condições exigidas no slide anterior garantidas.

Base Pos. Min. com ângulos de tam. uniformes

- V possui n vetores, o último vetor é dado por

$$d_{n+1} = - \sum_{i=1}^n v_i$$

Base Pos. Min. com ângulos de tam. uniformes

- V possui n vetores, o último vetor é dado por

$$d_{n+1} = - \sum_{i=1}^n v_i$$

- Prova-se que d_{n+1} junto de V satisfaz as condições exigidas. Assim podemos definir nossa base positiva com vetores de ângulos de tamanho uniforme como

$$D = [V \quad d_{n+1}],$$

e como V é base para \mathbb{R}^n claramente D é base positiva para o \mathbb{R}^n .

Perfil de desempenho do método SID-PSM1 otimizado sem problemas repetidos

