Proceeding Series of the Brazilian Society of Applied and Computational Mathematics, Vol. 3, N. 1, 2015.
Trabalho apresentado no XXXV CNMAC, Natal-RN, 2014.

Otimização de parâmetros de algoritmos

Bruno H. Cervelin, **Maria Aparecida Diniz-Ehrhardt**,
Depto de Matemâtica Aplicada, IMECC, UNICAMP,
endereço
E-mail: bcervelin@gmail.com, cheti@ime.unicamp.br,

Resumo: *Métodos numéricos dependem de parâmetros que podem influenciar (muito) sua eficiência. Os autores dos métodos costumam apresentar sugestões para esses parâmetros, algumas vezes arbitrários. Neste trabalho, seguindo as ideias apresentadas em [2], desenvolvemos um problema de otimização que busca encontrar os parâmetros ótimos que maximizam a eficiência de métodos numéricos. Como a função objetivo desse problema é cara, optamos por atacá-lo sob a óptica de otimização sem derivadas. Apresentamos resultados obtidos utilizando o método SID-PSM [5] para otimizar parâmetros do método Nelder-Mead [8].*

Palavras-chave: *Otimização sem derivadas, Otimização matemática, Algoritmos.*

Introdução

Neste trabalho focamos nossa atenção ao problema de otimização de parâmetros de algoritmos. Esse problema, apresentado por Audet em [2], busca encontrar os parâmetros que minimizam alguma medida de desempenho para métodos numéricos.

Como esse problema é muito caro, técnicas tradicionais de otimização não são recomendadas. Propõe-se, então, uma abordagem por técnicas de otimização sem derivadas. Um método que se destaca nessa área, quando temos poucas variáveis, mas função objetivo muito cara, é o SID-PSM [5]. É um método de busca padrão [9], que faz uso de técnicas de região de confiança sem derivadas, para otimizar funções irrestritas. Para tentar avaliar menos vezes a função objetivo, o método faz uso de derivadas simplex [4], que são também utilizadas na construção dos modelos utilizados na região de confiança.

Na seção 1 deste trabalho apresentamos as derivadas simplex e algumas de suas propriedades que foram consideradas na construção do método SID-PSM. Na seção 2 fazemos uma breve descrição do método SID-PSM. Na seção 3 apresentamos a construção do problema de otimização de parâmetros, justificando as escolhas feitas. Na seção 4 apresentamos testes numéricos ao aplicar o SID-PSM para otimizar os parâmetros do método Nelder-Mead [8] em relação ao número de avaliações de função. Por fim, na seção 5, apresentamos algumas conclusões.

1 Derivadas Simplex

Alguns métodos de otimização sem derivadas baseiam-se em modelos de interpolação para a função objetivo. Muitos tipos de modelo podem ser utilizados, mas, em geral, usam-se modelos de interpolação polinomial.

O gradiente simplex avaliado no ponto y^0 foi inicialmente definido como os coeficientes das interpolações de f em torno de y^0 utilizando os pontos do simplex $S = \{y^0, y^1, \ldots, y^n\}$, onde S é posicionado para interpolação linear, ou seja, a matriz que define o sistema da interpolação possui posto completo.

Esta definição foi estendida de modo que o conjunto $S = \{y^0, y^1, \ldots, y^q\}$ possa ter qualquer

DOI: 10.5540/03.2015.003.01.0420

quantidade de pontos, assim o gradiente simplex avaliado em y^0 é definido como a solução de

$$\min_{x \in \mathbb{R}^n} \|L(S)^\top x - \delta f(S)\|, \tag{1}$$

onde

$$L(S) = [y^1 - y^0, y^2 - y^0, \ldots, y^q - y^0]^\top$$

e

$$\delta f(S) = \left[f(y^1) - f(y^0), f(y^2) - f(y^0), \ldots, f(y^q) - f(y^0)\right]^\top.$$

Se $q < n + 1$, o gradiente simplex é a solução de norma mínima do problema (1).

Seguindo a ideia da primeira definição do gradiente simplex, construímos um modelo de interpolação quadrática,

$$m(x) = f(y^0) + (x - y^0)^\top \nabla_S f(y^0) + \frac{1}{2}(x - y^0)^\top \nabla_S^2 f(y^0)(x - y^0), \tag{2}$$

tal que para todo $j = 0, 1, \ldots, q$, temos $m(y^j) = f(y^j)$.

A matriz $\nabla_S^2 f(y^0)$ é chamada de Hessiana simplex de f avaliada em y^0 [4]. Se f for duas vezes diferenciável e $\nabla^2 f$ for Lipschitz contínua, então esta é uma matriz simétrica, logo podemos impor que $\nabla_S^2 f(y^0)$ também seja, daí a Hessiana simplex estará unicamente determinada quando nossa amostra S possuir $(n + 1)(n + 2)/2$ pontos.

Em [4] demonstrou-se que nas funções com gradiente Lipschitz contínuo, o gradiente simplex, calculado usando (1), pode ser visto como uma aproximação para o gradiente da função objetivo. E, em [1], é demonstrado que, sob certas hipóteses, ele é uma ϵ-aproximação para as componentes grandes do gradiente.

Quando utilizamos (2), se temos $(n + 1)(n + 2)/2$ pontos, $\nabla^2 f$ Lipschitz contínua em um conjunto aberto contendo S e algumas condições de posicionamento, podemos demonstrar que a Hessiana simplex pode ser vista como uma aproximação para a Hessiana da função, e o gradiente simplex ser visto como uma aproximação para o gradiente da função [4].

Quando temos menos de $(n + 1)(n + 2)/2$ pontos, nosso modelo fica indeterminado. Uma escolha para $\nabla_S^2 f(y^0)$, sugerida em [3], é considerar a indeterminação apenas na matriz $\nabla_S^2 f(x_k)$, e escolhê-la de forma a minimizar a sua norma de Frobenius, ou seja, resolver o problema:

$$\begin{array}{ll} \min\limits_{H \in \mathbb{R}^{n \times n}} & \frac{1}{4}\|H\|_F \\ \text{s.a :} & H = H^\top \\ & m_H(y) = f(y) \quad \forall \quad y \in Y, \end{array} \tag{3}$$

onde m_H é a mesma função m da equação (2) com $\nabla_S^2 f(y^0) = H$.

Em [4], os autores demonstram que, se temos menos de $(n + 1)(n + 2)/2$, ∇f for Lipschitz contínuo com constante γ e algumas condições de posicionamento, o limitante do gradiente simplex, quando considerado como uma aproximação para o gradiente da função, é dado por

$$\|\nabla f(y^0) - \nabla_S f(y^0)\| \leq C_q \gamma K \Delta,$$

onde K é uma constante que depende do posicionamento, C_q depende do número de pontos na amostra e $\Delta = \max\limits_{i=1\ldots q} \|y^0 - y^i\|$.

2 SID-PSM

O método SID-PSM (Simplex in Derivative Pattern Search Method) [5] é um método de otimização sem derivadas que une a ideia de busca padrão com região de confiança [4].

Para simplificar o estudo dividimos o método em três etapas: passo de busca, passo de pesquisa e atualização do tamanho do passo.

São dados $x_k \in \mathbb{R}^n$, aproximação para a solução na iteração k, e $\alpha_k > 0$, o tamanho do passo.

No primeiro passo, o de busca, o método procura minimizar um modelo da função objetivo dentro de uma bola. Este modelo é construído usando derivadas simplex, que são calculadas com os pontos onde a função objetivo já foi avaliada nas iterações anteriores.

Se o minimizador do modelo oferece decréscimo para o valor da função objetivo, o ponto é aceito, declaramos sucesso e pulamos o passo de pesquisa. Caso contrário, passamos para o passo de pesquisa.

No passo de pesquisa é realizada a busca padrão, ou seja, tomamos um conjunto ordenado D^k positivamente gerador para o $\mathbb{R}^n$ e testamos os pontos da forma $x_k + \alpha_k d_k$, com $d_k \in D^k$. Se encontramos uma direção que nos ofereça decréscimo, aceitamos o ponto e declaramos sucesso. Caso contrário declaramos fracasso da iteração.

Podemos tentar acelerar este passo utilizando a estratégia de poda, neste caso avaliamos apenas o vetor em D^k que mais se aproxima, em relação ao ângulo, com $-\nabla_S f(x_k)$.

Caso a iteração tenha sido um fracasso, devemos diminuir o tamanho do passo α_k, caso contrário, mantemos ou aumentamos seu tamanho.

Este método pode ser descrito como um método de busca padrão, logo todas as propriedades de convergência demonstradas em [9] são válidas.

3 Otimização de Parâmetros

Nesta seção, baseados nas ideias apresentadas em [2], desenvolvemos um problema de otimização com intuito de encontrar parâmetros ótimos de algoritmos.

Formalmente o problema pode ser definido como: dado um método s, que depende dos parâmetros x_i, $i = 1, \ldots, n$, queremos encontrar os parâmetros x_i^*, $i = 1, \ldots, n$, que minimizam uma medida m de desempenho do método (por exemplo o tempo de processamento).

Talvez a abordagem mais intuitiva para a função objetivo seja: dado um conjunto de problemas P a serem resolvidos, somar o custo de se resolver cada um deles. Uma desvantagem bastante óbvia desta abordagem é que podemos ter um problema em P muito caro de ser resolvido com os parâmetros iniciais, e muitos problemas baratos. Como estamos otimizando a soma dos custos, com os parâmetros ótimos o problema que era caro pode se tornar barato, e grande parte dos problemas baratos podem se tornar mais caros. Assim, na prática, teremos um método pior.

Esse problema pode ser facilmente contornado se aplicarmos uma normalização para cada um dos problemas em P. Criamos um vetor auxiliar $f^0 \in \mathbb{R}^{|P|}$, onde f_i^0 representa o custo de se avaliar o problema i utilizando os parâmetros iniciais. Tomando $f : \mathbb{R}^{|P|} \to \mathbb{R}^{|P|}$, onde $f_i(x)$ representa o custo de se avaliar o problema i utilizando os parâmetros x, definimos nosso problema como

$$\hat{f}(x) = \sum_{i \in P} \frac{f_i(x)}{f_i^0}.$$

Dessa forma estamos otimizando eficiência do método, porém não estamos levando em consideração a robustez, nem o fato de que podem existir parâmetros restritos a alguma região.

Definimos o conjunto Ω como o conjunto com os possíveis parâmetros, e criamos a função modificada

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

note que se usarmos um método de descida para resolver $\min_x \hat{f}_\Omega(x)$ e $x_0 \in \Omega$, a sequência $\{x_k\}$ gerada pelo método satisfaz $\hat{f}_\Omega(x_k) \leq |P|$, para todo k. Agora nos resta apenas considerar a robustez do método.

DOI: 10.5540/03.2015.003.01.0420

Para essa última etapa definimos o conjunto $\bar{\Omega} \subset \Omega$, esse conjunto possui os parâmetros que ao aplicarmos o método s a todos os problemas de P é 25% pior que a solução obtida com os parâmetros originais (no sentido de valor de função objevo) em no máximo N problemas (N pequeno). Assim nosso problema é

$$\hat{f}_{\bar{\Omega}}(x) = \begin{cases} \sum\limits_{i \in P} \frac{f_i(x)}{f_i^0} & \text{se} \quad x \in \bar{\Omega} \\ |P| + 1 & \text{c.c.} \end{cases} . \tag{4}$$

Para avaliar cada elemento $f_i(x)$ do vetor $f(x)$ devemos resolver um problema de otimização, assim, a cada avaliação da função objetivo resolvemos $|P|$ problemas de otimização, logo métodos de otimização clássica são inviáveis para resolver esse problema, a alternativa é utilizar métodos de otimização sem derivadas.

4 Resultados Obtidos

Nesta seção apresentamos os resultados obtidos ao aplicarmos a estratégias acima ao método Nelder-Mead [8]. Os parâmetros que iremos otimizar serão ρ (reflexão), χ (expanção), γ (contração) e σ (redução), assim a variável do nosso problema é representada como

$$x = [\rho, \chi, \gamma, \sigma]^\top .$$

Pelas definições de cada um dos parâmetros temos o conjunto Ω pode ser descrito como

$$\Omega = \left\{ [\rho, \chi, \gamma, \sigma]^\top : \rho \in [0, \chi],\, \gamma \in (0,1),\, \sigma \in (0,1),\, \chi \geq 1 \right\}.$$

Para definir o conjunto $\bar{\Omega}$ utilizamos $N = 5$.

Utilizamos como medida de desempenho o número de avaliações de função. Definimos "resolver um problema" como:

Definição 1. *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 $\hat{\theta}_p = \min\limits_{\hat{s} \in S} \theta_p(\hat{s})$. Então se o método atinge um passo com tamanho menor que uma tolerância ϵ_{tol} em menos de 10^6 avaliações de função, e*

$$\frac{\theta_p(s) - \hat{\theta}_p}{\hat{\theta}_p} \leq 0,25 \quad \text{se} \quad \hat{\theta}_p \neq 0,$$
$$\theta_p(s) \leq \texttt{eps}^2 \quad \text{se} \quad \hat{\theta}_p = 0,$$

onde `eps` é o épsilon da máquina, diz-se que o método s resolve o problema p, em relação ao conjunto S.

Em outras palavras, dizemos que o método s resolve o problema p se a solução obtida por este é no máximo 25% pior que a melhor solução obtida.

Em [7] Moré e Wild sugeriram um conjunto de problemas testes para métodos de minimização irrestrita sem derivadas. Esse conjunto possui 22 problemas e é um subconjunto dos problemas do pacote CUTEr, cujas funções são somas de quadrados. Os autores sugerem diversas dimensões para cada problema, escolhemos sempre a menor, de modo que nossos problemas possuem dimensão entre 2 e 11.

Para minimizar a função (4) utilizamos o método SID-PSM, que se destaca na resolução de problemas pequenos e com função objetivo cara [5]. Utilizamos o SID-PSM com dois conjuntos distintos de parâmetros, em ambos

- aumentamos o passo fazendo $\alpha_{k+1} = 2\alpha_k$, e reduzimos fazendo $\alpha_{k+1} = \frac{1}{2}\alpha_k$,

- o passo de busca é feito construindo um modelo quadrático da função objetivo. O ponto testado no passo de busca é calculado encontrando o minimizador do modelo na bola $B(x_k, \Delta_k)$, onde $\Delta_k = 2\alpha_k \max_{d \in D_k} \|d\|$,

- Tentamos calcular um gradiente simplex a partir de um conjunto de pontos S com $n+1$ pontos onde a função objetivo já foi avaliada, tal que, $s^0 = x_k$ e dada a decomposição SVD reduzida $UDV^\top = L(S)^T/\Delta$, onde $\Delta = \max_{k=1,\ldots,n} \|s^0 - s^k\|$, temos $\|D^{-1}\| \leq 100$.

- se temos um gradiente simplex, ordenamos o conjunto de pesquisa usando o ângulo entre seus vetores e o gradiente simplex, caso contrário, continuamos a pesquisa a partir do último vetor que ofereceu decréscimo,

- interrompemos a execução do método se a função objetivo é calculada mais de 10^6 vezes, ou se o tamanho do passo é menor que ϵ_{tol} (testamos diferentes valores para este parâmetro).

Chamamos os métodos de SID-PSM1 e SID-PSM2. Suas diferenças são: o conjunto de pesquisa (pss); forma de atualização do passo (ssa); e uso de poda (pru). O método que chamamos de SID-PSM1 usa os seguintes parâmetros:

- pss: D^k é permutação do conjunto $\{e, -e, e_1, \ldots, e_n, -e_1, \ldots, -e_n\}$,

- ssa: usa estratégia de decréscimo esperado, ou seja, define

$$\rho_k = \frac{f(x_k) - f(x_{k+1})}{m_k(x_k) - m_k(x_{k+1})},$$

  e

  - se $\rho_k > \gamma_2$, aumentar o passo;
  - se $\gamma_1 < \rho_k \leq \gamma_2$, mantê-lo;
  - se $\rho_k \leq \gamma_1$, diminuí-lo;

  com $\gamma_1 = \frac{1}{4}$ e $\gamma_2 = \frac{3}{4}$,

- pru: testa somente a direção mais promissora no passo de pesquisa, ou seja, direção com o menor ângulo com $-\nabla_S f(x_k)$.

O SID-PSM2 possui os seguintes parâmetros:

- pss: usa D como uma base geradora positiva com ângulos uniformes entre seus vetores, D^k é permutação de D,

- ssa: aumenta o tamanho do passo somente quando a mesma direção oferece decrésimo em duas iterações consecutivas,

- pru: testa todos os pontos no conjunto de pesquisa até encontrar decréscimo, ou declarar fracasso.

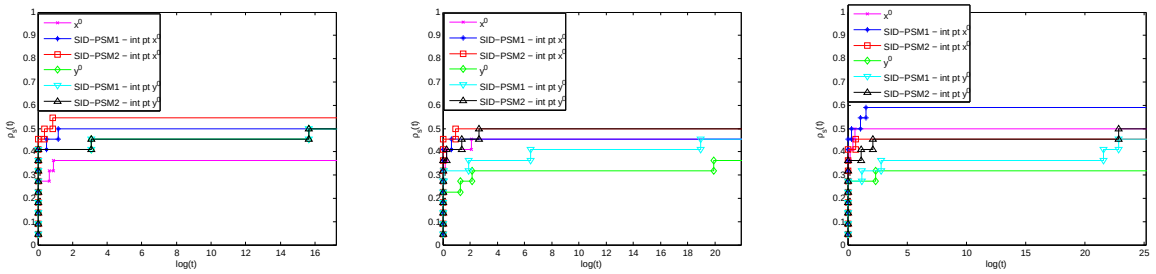
Executamos as duas versões do SID-PSM testando dois parâmetros iniciais,

$$x^0 = [1, 2, 0.5, 0.5]^\top, \qquad y^0 = [1, 4, 0.25, 0.25]^\top,$$

e utilizamos como tolerância para o SID-PSM $\epsilon_{tol} = 10^{-5}$, 10^{-6} e 10^{-7} e para o Nelder-Mead utilizamos $\frac{\epsilon_{tol}}{10}$.

Na figura 1 apresentamos os perfis de desempenho comparando os parâmetros obtidos por cada método, para cada tolerância. Os problemas utilizados para construir esses perfis são os mesmos utilizados para otimizar a função (4).

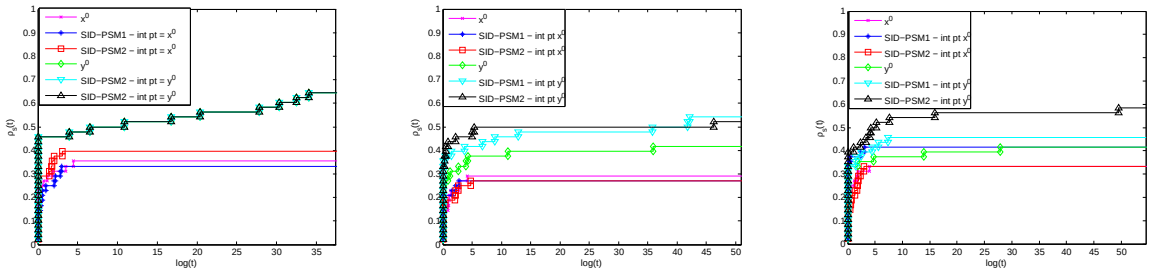
Figura 1: Perfil de desempenho do método Nelder-Mead com parâmetros x^0, y^0 e obtidos como solução do problema (4), resolvendo problemas em [7].



(a) $\epsilon_{tol} = 10^{-5}$ (b) $\epsilon_{tol} = 10^{-6}$ (c) $\epsilon_{tol} = 10^{-7}$

Esses perfis não são justos, pois testamos os parâmetros otimizados nos mesmos problemas que utilizamos para otimizá-los. Escolhemos um novo conjunto de problemas testes. Esse conjunto, apresentado em [6], possui 35 problemas de minimização irrestrita cuja função é a soma de quadrados. Os primeiros 20 problemas possuem dimensão entre 2 e 12 e são fixas. Os 15 restantes possuem dimensão livre e foram resolvidos com dimensão 12 e 20. Com esse conjunto de problemas construímos um perfil de desempenho comparando os parâmetros obtidos com cada uma das tolerância utilizadas. Esses perfis estão disponíveis na figura 2.

Figura 2: Perfil de desempenho do método Nelder-Mead com parâmetros x^0, y^0 e obtidos como solução do problema (4), resolvendo problemas em [6].



(a) $\epsilon_{tol} = 10^{-5}$ (b) $\epsilon_{tol} = 10^{-6}$ (c) $\epsilon_{tol} = 10^{-7}$

Pela figura 1, vemos que, no geral, os parâmetros otimizados nos forneceram uma melhora na eficiência, e pouca, ou nenhuma perda na robustez (em alguns casos houve até uma melhora).

Analisando a figura 2 (a) percebemos que, quando utilizamos y^0 como parâmetro inicial, não obtemos melhora no desempenho nem na robustez do método. Isso por que obtivemos como resultado do problema de otimização o próprio y^0. Já quando utilizamos x^0, temos uma pequena melhora tanto na robustez quanto na eficiência quando utilizamos o SID-PSM2, e uma pequena piora quando utilizamos o SID-PSM1.

Agora analisando a figura (b) percebemos que ao utilizarmos x^0 como parâmetro inicial perdemos robustez e eficiência, porém quando usamos y^0 melhoramos os dois quesitos uma quantia considerável.

Por fim na figura (c) novamente temos um melhor desempenho do método quando usamos os parâmetros resultantes da resolução do problema (2) com o parâmetro inicial y^0, os parâmetros obtidos através de x^0 também nos oferecem melhora para o método Nelder-Mead quando utilizamos o SID-PSM1 para obtê-los.

DOI: 10.5540/03.2015.003.01.0420

5 Conclusão

A partir dos resultados obtidos pela otimização do método Nelder-Mead vemos que, apesar de não possuirmos uma garantia, podemos obter parâmetros para os métodos que utilizem o máximo de seu potencial.

Apesar de em alguns casos não termos obtido nenhuma melhora, e em outros, o resultado obtido ter sido pior que o original, nos que obtivemos sucesso, a melhora foi significativamente maior do que quando falhamos.

Um dos motivos de não obtermos melhora foi encontrarmos como resultado o ponto inicial (por exemplo, $\epsilon_{tol} = 10^{-5}$ e como ponto inicial y^0). Uma explicação para esse fato está na descontinuidade da função $\bar{f}_\Omega$. Essa descontinuidade invalida muitos dos teoremas de convergência para métodos de busca padrão. Um desses teoremas é o que garante que o ponto de acumulação gerado pelo método de busca padrão é um ponto estacionário.

Os casos em que temos decréscimo na eficiência podem ser justificados pela escolha dos problemas testes. Como otimizamos a função (4) utilizando um conjunto de problemas, estamos otimizando o método para este conjunto específico. Se aplicamos o método com o parâmetro otimizado a um problema que não pertença a este conjunto, não teremos garantia que o parâmetro seja ótimo para este problema também.

A escolha de métodos de otimização sem derivadas foi adequada, pois mesmo que o problema fosse contínuo, não teríamos acesso à função, nem às derivadas, logo continuarímos sem poder utilizar métodos com derivadas.

Verificamos que no geral obtivemos resultados satisfatórios, o que demonstra o grande potencial desta técnica. Pretendemos, em trabalhos futuros, continuar com a manipulação da função (4) para obter melhores resultados na resolução do problema de otimização dos parâmetros.

Referências

[1] M.A. Abramson, C. Audet e J.E. Dennis, Generalized pattern searches with derivative information, *Math. Program.*,100, pp. 3-25, 2004.

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

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

[4] A.L. Custódio, K. Scheinberg e L.N. Vicente, Introduction to Derivative-Free Optimization, *MPS-SIAM Series on Optimization* 8, 2009.

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

[6] J.J. Moré, B.S. Garbow and K.E. Hillstrom, Testing unconstrained optimization software, *ACM Transactions on Mathematical Software*, 7, pp. 17-41, 1981.

[7] J.J. Moré and S. Wild, Benchmarking Derivative-Free Optimization Algorithms, *SIAM Journal on Optimization*, 20, pp. 172-191, 2009

[8] J.A. Nelder e R. Mead, A simplex method for function minimization, *The Computer Journal*, 7, pp. 308-313, 1965.

[9] V. Torczon, On the convergence of pattern search algorithm, *SIAM Journal on Optimization*, 7, pp. 1-25, 1997.

