Trabalho apresentado no XXXVIII CNMAC, Campinas - SP, 2018.

Proceeding Series of the Brazilian Society of Computational and Applied Mathematics

Estudo do desempenho de métodos de amostragem para minimização irrestrita não suave

David Ricardo Barreto Lima Silva[1]
Instituto de Matemática e Estatística, USP, São Paulo, SP
Sandra Augusta Santos[2]
Instituto de Matemática, Estatística e Computação Científica, UNICAMP, Campinas, SP
Lucas Eduardo Azevedo Simões[3]
Instituto de Matemática, Estatística e Computação Científica, UNICAMP, Campinas, SP

Resumo. A otimizacão não suave é um ramo da otimização que trabalha com funções objetivo não diferenciáveis em um subconjunto do domínio. O algoritmo *Gradient Sampling* (GS) foi proposto recentemente e minimiza a função objetivo com base no gradiente calculado em amostras de pontos gerados uniformemente em uma vizinhança do ponto corrente. Variações deste método envolvendo diferentes tamanhos de direções e diferentes valores de parâmetros foram exploradas no presente trabalho. Problemas conhecidos da literatura foram utilizados para analisar comparativamente o comportamento de algumas variantes do método e sua dependência com relação ao número de pontos amostrados. O número de iterações e o valor ótimo obtido foram as medidas de eficiência utilizadas.

Palavras-chave. Otimização não diferenciável, Otimização irrestrita, Algoritmos, Amostragem (Estatística), Experimentos numéricos

1 Introdução

Problemas de otimização envolvendo funções não suaves possuem grande interesse prático, uma vez que estes estão presentes em diversos problemas reais [8–10]. Tais problemas podem ser agrupados em duas classes distintas: aqueles que lidam com funções convexas e os demais problemas de otimização não suave, que apresentam funções não convexas como um complicador adicional.

Com o objetivo de apresentarem um método robusto para a segunda classe de problemas, os autores em [3] sugerem um algoritmo chamado *Gradient Sampling* (GS). O algoritmo tem suas raízes nos trabalhos apresentados em [1, 2], e tem como principal ideia generalizar o algoritmo de máxima descida para funções não suaves. Para tanto, o método GS faz uso de uma amostra de pontos em volta do ponto corrente para obter informações do comportamento local da função em questão. Assim, o algoritmo é capaz de calcular uma direção de busca satisfatória e obter um decréscimo suficiente da função objetivo a

[1]davidr@ime.usp.br
[2]sandra@ime.unicamp.br
[3]simoes.lea@gmail.com

DOI: 10.5540/03.2018.006.02.0277

cada iteração. Uma das vantagens deste método é lidar apenas com pontos onde a função objetivo é diferenciável. Em outras palavras, o método GS nunca faz uso de subgradientes. Neste trabalho apresentamos o algoritmo GS, discutindo brevemente algumas de suas variantes. Como contribuição do nosso estudo, análises de desempenho destas variantes são feitas a partir da solução de uma coleção de problemas de otimização não suave presentes na literatura [5, 12].

2 O Método GS e Variantes

O método *Gradient Sampling* tem como objetivo resolver o seguinte problema de otimização

$$\min_{x\in\mathbb{R}^n} f(x), \tag{1}$$

sendo $f : \mathbb{R}^n \to \mathbb{R}$ uma função localmente Lipschitz contínua e continuamente diferenciável em um conjunto aberto $\mathcal{D} \subset \mathbb{R}^n$. Ademais, $\mathcal{D}$ deve ser de tal forma que o conjunto $\mathbb{R}^n \setminus \mathcal{D}$ tenha medida de Lebesgue zero [4].

Este algoritmo pode ser visto como uma generalização do método de máxima descida. A cada iteração k, o algoritmo GS encontra uma direção de busca d^k e, através de uma busca linear, calcula um tamanho de passo t_k para, ao final da iteração, definir o próximo ponto x^{k+1} (veja Figura 1). A grande diferença entre o método de Cauchy e o algoritmo GS está na forma como estes métodos encontram uma direção de descida para a função f em x^k. Enquanto o método de máxima descida define $d^k = -\nabla f(x^k)$, o algoritmo GS necessita de uma coleção de gradientes calculados ao redor do ponto corrente x^k para garantir que d^k será uma boa direção de descida para a função objetivo.

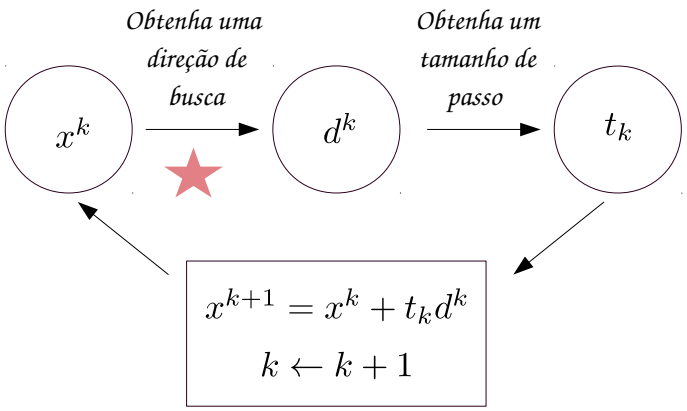

Figura 1: Esboço simplificado do algoritmo GS

Para facilitar a compreensão do passo ★ que aparece na Figura 1, recorremos ao auxílio

DOI: 10.5540/03.2018.006.02.0277

da Figura 2. Para o cálculo da direção d^k, o algoritmo sorteia uniformemente pontos ao redor do iterando x^k, em uma bola com raio amostral ϵ_k. Como, por hipótese, o conjunto $\mathbb{R}^n \setminus \mathcal{D}$ tem medida zero, isto significa que, com probabilidade 1, a função f é diferenciável nos pontos sorteados. Desta forma, podemos, com probabilidade 1, calcular os gradientes de f nestes pontos e obter g^k como o vetor de menor norma no conjunto formado pela combinação convexa destes gradientes. Assim, fazemos $d^k = -g^k$. Caso $\|d^k\|$ seja menor que uma dada tolerância ν_k, então reduzimos o raio amostral e a tolerância, e fazemos um novo sorteio. O algoritmo termina quando tanto o raio amostral ϵ_k quanto a tolerência ν_k estão pequenos o suficiente.

É possível demonstrar que, para toda iteração k, a direção calculada d^k é uma direção de descida para f em x^k. Assim, com o auxílio desta informação, Kiwiel [7] mostrou que, se f é limitada inferiormente, então qualquer ponto de acumulação da sequência $\{x^k\}$ é um ponto estacionário para f.

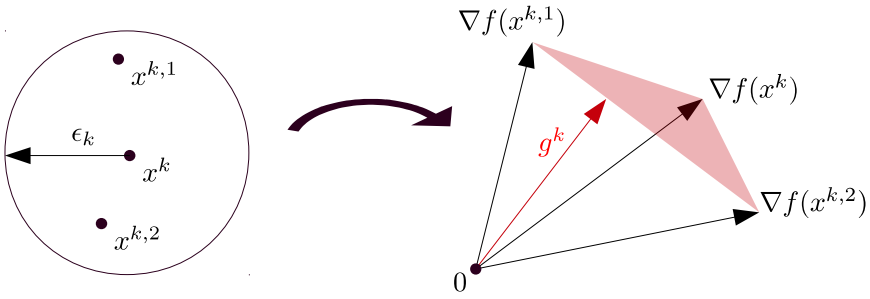

Figura 2: Esboço do cálculo da direção d^k. Os pontos $x^{k,1}$ e $x^{k,2}$ representam os pontos sorteados ao redor do iterando x^k. O valor ϵ_k é chamado de raio amostral. A direção g^k é o vetor de menor norma no fecho do casco convexo das direções $\nabla f(x^k)$, $\nabla f(x^{k,1})$ e $\nabla f(x^{k,2})$.

Inicialmente, em [3], o método GS foi proposto com $d^k = -g^k/\|g^k\|$. Porém, no ano de 2007, Kiwiel apresenta uma variante deste método que, a cada iteração, define $d^k = -g^k$. Além desta forma não normalizada da direção de busca ser mais coerente com o método de máxima descida, este novo d^k tem a vantagem de satisfazer, com mais facilidade, o decréscimo suficiente da busca linear quando x^k está próximo de um ponto estacionário. Contudo, mesmo com esta mudança, existe a possibilidade da busca linear falhar na prática. Isto ocorre porque, quando os pontos sorteados não apresentam satisfatoriamente a informação local da função f, a norma do vetor g^k pode ter um valor muito alto, mesmo quando x^k está próximo de um ponto estacionário. Assim, para que o decréscimo suficiente do valor de função seja satisfeito, devemos ter um tamanho de passo t_k excessivamente pequeno, o que, na prática, significa dizer que t_k deve ser menor do que a precisão da máquina. Com o objetivo de evitar esta falha, na prática acaba-se exigindo apenas o decréscimo simples do valor de função a cada iteração, mesmo sabendo que esta variante não possui convergência teórica garantida [3].

DOI: 10.5540/03.2018.006.02.0277

Desta forma, quatro variantes do método GS podem ser naturalmente consideradas:

GS-NsD: **G**radient **S**ampling **N**ormalizado **s**em **D**ecréscimo suficiente;

GS-nNsD: **G**radient **S**ampling **n**ão **N**ormalizado **s**em **D**ecréscimo suficiente;

GS-NcD: **G**radient **S**ampling **N**ormalizado **c**om **D**ecréscimo suficiente;

GS-nNcD: **G**radient **S**ampling **n**ão **N**ormalizado **c**om **D**ecréscimo suficiente.

Além destas quatro variantes, condideraremos, nos nosso experimentos numéricos, uma quinta versão do algoritmo GS, a qual iremos denotar por GS-LOC. Esta variante surgiu em [6] e tem como principal intuito mostrar a estreita relação entre o método GS e o algoritmo de máxima descida. Manipulando a velocidade em que os valores ϵ_k e ν_k decrescem ao longo das iterações, o método GS-LOC consegue garantir, para funções que possuem um comportamento suave em uma região contendo o ponto ótimo do problema, uma taxa linear de decréscimo do valor da função objetivo. Essa região precisa ser, pelo menos uma curva, mas pode ser uma superfície, ou ter maior dimensão. Em [6] foi mostrado que a convergência linear do método de Cauchy pode ser recuperada quando os valores de parâmetros do método GS são apropriadamente ajustados.

3 Resultados Computacionais

Com o objetivo de analisar a eficiência numérica das cinco versões do método GS descritas na seção anterior, escolhemos dezoito funções bem conhecidas na literatura de problemas de otimização não suave [5, 12] e fizemos uso de duas medidas: número de iterações, a qual representa a nossa medida de esforço; e o menor valor de função atingido, o qual representa a nossa medida de qualidade.

Os dados foram gerados a partir da minimização de cada uma das dezoito funções, com trinta pontos iniciais escolhidos randomicamente no domínio $[-1, 1]^n$, para as cinco primeiras funções, e $[0, 2]^n$, para as funções restantes.

De forma geral, pudemos notar que os desempenhos dos métodos GS-NsD e GS-NcD não apresentam grandes diferenças, uma vez que a única característica que os diferencia é o decréscimo exigido na busca linear. Similarmente, podemos ver que os métodos que normalizam a direção de busca e aqueles que não o fazem também apresentam pouca variação no desempenho quando comparados entre si. Em contrapartida, o método GS-LOC claramente se distingue dos demais. Por exemplo, podemos ver que este algoritmo possui um excelente desempenho para as funções F5 e F9 (veja as Figuras 3 e 4). Isto pode ser explicado pelo fato que estas funções satisfazem as hipóteses exigidas em [6] para que o método GS tenha uma convergência local linear. Por outro lado, quando consideramos funções que não satisfazem tais hipóteses como, por exemplo, as funções F1 e F6, as outras variantes tem um desempenho muito superior.

Como explicado na seção anterior, o método GS trabalha com pontos amostrados a cada iteração. Para que o resultado de convergência global do método seja válido, é necessário que o algoritmo faça um sorteio de, ao menos, $n + 1$ pontos. Desta forma,

DOI: 10.5540/03.2018.006.02.0277

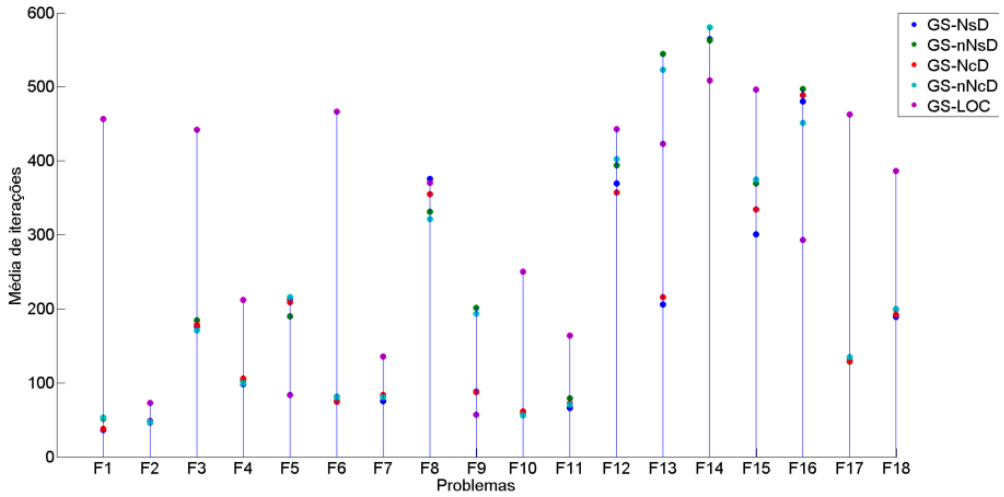

Figura 3: Média do número de iterações para $n = 20$.

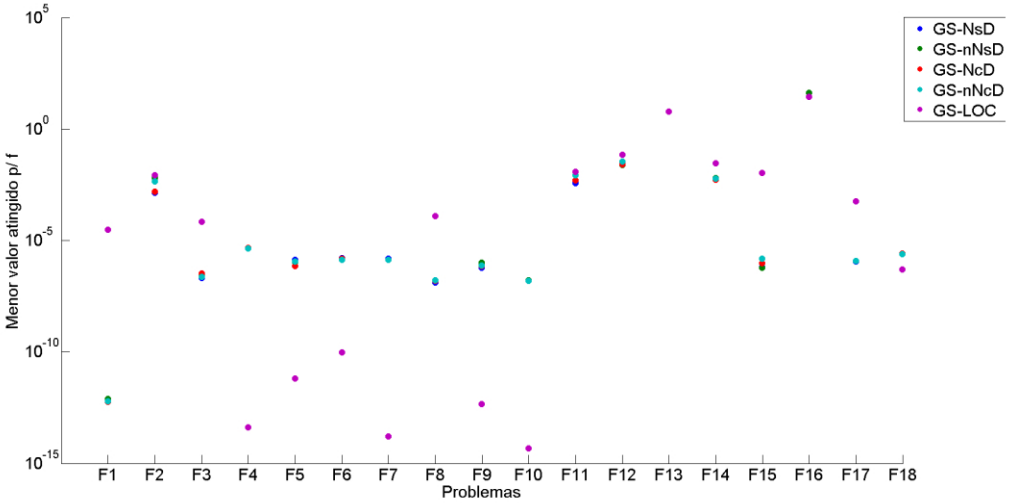

Figura 4: Mediana dos valores de função atingidos, relativa ao menor valor atingido dentre todos os experimentos, para $n = 20$.

podemos nos perguntar como cada método se comporta quando este número de pontos amostrados m é alterado. Com este objetivo, fizemos testes computacionais considerando $m \in \{\lceil n/2 \rceil, n+1, 2n, 3n\}$.

Após a análise dos experimentos, pudemos observar que o método GS não apresenta diferença significativa nos resultados quando $m = 2n$ ou $m = 3n$. Porém, como para $m = 3n$ o custo computacional é maior, concluímos que $m = 2n$ é um valor mais adequado. Por fim, quando consideramos os valores $m = \lceil n/2 \rceil$ e $m = n + 1$, percebemos que todos os métodos perdem em robustez para o valor $m = 2n$. Com base nisso, concluímos

DOI: 10.5540/03.2018.006.02.0277

que a escolha de $2n$ amostras, apresentada na versão padrão do algoritmo GS, é a mais satisfatória.

Nesta seção exibimos uma análise concisa dos experimentos que foram realizados. Para um estudo mais detalhado, encaminhamos o leitor ao trabalho apresentado em [11].

4 Conclusões

Com base em nossos experimentos numéricos, pudemos notar que o algoritmo GS é pouco sensível ao uso da condição de decréscimo suficiente na busca linear. Ademais, também pudemos observar pouca influência da normalização da direção de busca nos resultados.

Fica evidente que a variante que mais se distingue das demais é a versão que denotamos por GS-LOC. Como esperado, esta versão do método GS apresenta bons resultados quando a função objetivo tem um comportamento suave em uma região contendo o ponto ótimo do problema. Por fim, o número de pontos amostrais sugerido pelos autores originais do método [3] se mostrou satisfatório em nossos experimentos.

Agradecimentos

Agradecemos aos apoios CAPES, CNPq (302915/2016-8) e FAPESP (2013/05475-7, 2013/07375-0, 2016/22989-2).

Referências

[1] J. V. Burke, A. S. Lewis, and M. L. Overton, Approximating subdifferentials by random sampling of gradients, *Mathematics of Operations Research*, 27(3):567–584, 2002.

[2] J. V. Burke, A. S. Lewis, and M. L. Overton, Two numerical methods for optimizing matrix stability, *Linear Algebra and its Applications*, 351–352:117–145, 2002.

[3] J. V. Burke, A. S. Lewis, and M. L. Overton, A robust gradient sampling algorithm for nonsmooth, nonconvex optimization, *SIAM Journal on Optimization*, 15(3):751–779, 2005.

[4] A. A. Castro Jr. *Curso de Teoria da Medida*. 3 ed. IMPA, Rio de Janeiro, 2015.

[5] M. Haarala, K. Miettinen, and M. M. Mäkelä, New limited memory bundle method for large-scale nonsmooth optimization, *Optimization Methods and Software*, 19(6):673–692, 2004.

[6] E. S. Helou, S. A. Santos, and L. E. A. Simões, On the Local Convergence Analysis of the Gradient Sampling Method for Finite Max-Functions, *Journal of Optimization Theory and Applications*, 175(1):137–157, 2017.

[7] K. C. Kiwiel, Convergence of the gradient sampling algorithm for nonsmooth nonconvex optimization, *SIAM Journal on Optimization*, 18(2):379–388, 2007.

[8] P. Maréchal and J. J. Ye, Optimizing condition numbers, *SIAM Journal on Optimization*, 20(2):935–947, 2009.

[9] J. J. Moreau and P. D. Panagiotopoulos, *Nonsmooth mechanics and applications*, Springer, Vienna, 2014.

[10] C. Peng, X. Jin, and M. Shi, Epidemic threshold and immunization on generalized networks, *Physica A: Statistical Mechanics and its Applications*, 389(3):549–560, 2010.

[11] D. R. B. L. Silva, Estudo do desempenho de métodos de amostragem para minimização irrestrita não suave, Dissertação de Mestrado, IMECC, Unicamp, 2018.

[12] A. Skajaa, Limited memory BFGS for nonsmooth optimization, Dissertação de Mestrado, Instituo Courant de Ciências Matemáticas, Universidade de Nova Iorque, 2010.

DOI: 10.5540/03.2018.006.02.0277