Otimização Não Linear

CM106/CMM204/CMI043

Tópico 05 - Detalhes Computacionais

Abel Soares Siqueira - UFPR

2020/s1

Detalhes Computacionais

- Critérios de parada.
- Definição de problemas computacionalmente.
- Resolução de sistemas lineares.
- Comparação de algoritmos.

Critérios de parada

Critérios de parada

- Problemas irrestritos pedem $\nabla f(x) = 0$.
- Usaremos $\|\nabla f(x_k)\| \leq \epsilon_a + \epsilon_r \|\nabla f(x_0)\|$.
- Também paramos com máximo de tempo, iterações e número de avaliações de função.

Definição de problemas

Definição de problemas computacionalmente

- Você inventa um método e quer mostrar que ele funciona. Como você faz isso?
- Outra pessoa faz a mesma coisa. Vocês recriam os problemas?
- Em programação linear, é possível mandar um arquivo com as matrizes e vetores.
- Em PNL não é tão simples. Queremos a função e suas derivadas, e criar funções novas.
- AMPL é uma das linguagens de modelagem famosas. É proprietária, no entanto.
- CUTE/CUTEr/CUTEst é um dos repositórios mas conhecidos.
- Linguagens de modelagens também são opções, mas são mais restritas no uso.
- Um dos problemas é encontrar as derivadas.

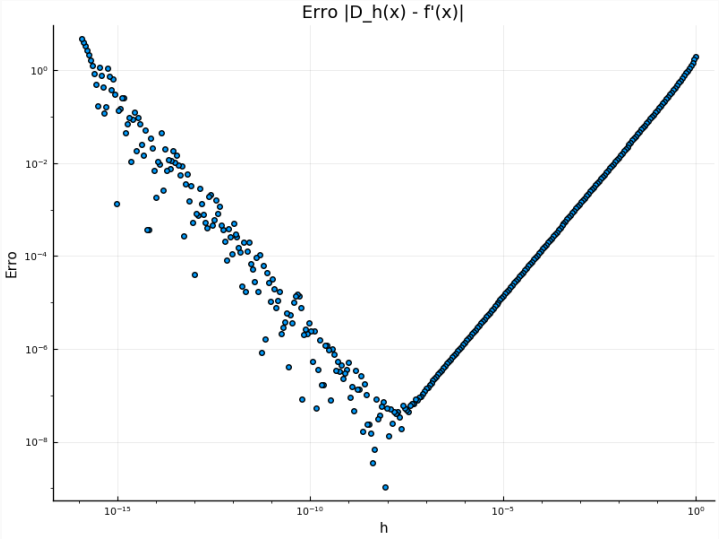
Derivadas - Diferenças finitas

- A primeira alternativa é aproximar a derivada por diferenças finitas, por exemplo:

$$f'(x) \approx D_h(x) = \frac{f(x+h) - f(x)}{h}.$$

- Teoricamente $D_h(x) \to f'(x)$ quando $h \to 0$. Porém, o erro proveniente da aproximação numérica tende a ∞ quanto $h \to 0$.

Derivadas - Diferenças finitas



Derivadas - Valores duais

- Uma maneira bastante sofisticada, mas de difícil implementação é usar números duais: $a + b\epsilon$, onde $\epsilon^2 = 0$.
- Vale $f(a + b\epsilon) = f(a) + \epsilon b f'(a)$ por Taylor, então basta saber calcular o valor de função para bases diferentes de f.
- Para isso definimos computacionalmente estruturas para um número dual (e.g. $D(a, b)$), e definimos o que acontece quando aplicamos funções nesses números.
- Por exemplo: $D(a, b)^\alpha = D(a^\alpha, \alpha a^{\alpha-1} b)$, $e^{D(a,b)} = D(e^a, be^a)$, $\ln(D(a, b)) = D(\ln a, b/a)$.

Derivadas - Valores duais

- Com as definições acima, se quisermos a derivada de $f(x) = \sqrt{1 + e^x}$, por exemplo, basta perguntar qual o argumento acompanhando ϵ de $f(x + \epsilon)$:

$$\begin{aligned} f(x+\epsilon) &= \sqrt{1 + e^{x+\epsilon}} = (D(1,0) + e^{D(x,1)})^{1/2} \\ &= (D(1,0) + D(e^x, e^x))^{1/2} = D(1 + e^x, e^x)^{1/2} \\ &= D(\sqrt{1+e^x}, \tfrac{1}{2}(1+e^x)^{-1/2}e^x) \end{aligned}$$

- Logo, a derivada é $\frac{1}{2}(1 + e^x)^{-1/2}e^x$.
- Note que a estratégia é numérica, isto é, cada passo é feito com um valor, não com símbolos.

Derivadas - Valores duais

```
 1 struct Dual
 2   a :: Float64
 3   b :: Float64
 4 end
 5
 6 Dual(x) = Dual(x, 0.0)
 7
 8 function Base.print(io :: IO, d :: Dual)
 9   print(io, "$(d.a) + $(d.b)eps")
10 end
11
12 import Base.+, Base.-, Base.*, Base.^
13 function +(d1 :: Dual, d2 :: Dual)
14   Dual(d1.a + d2.a, d1.b + d2.b)
15 end
```

Derivadas - Valores duais

```
17 +(d :: Dual, x) = d + Dual(x)
18 +(x, d :: Dual) = Dual(x) + d
19
20 function -(d1 :: Dual, d2 :: Dual)
21   Dual(d1.a - d2.a, d1.b - d2.b)
22 end
23
24 -(d :: Dual, x) = d - Dual(x)
25 -(x, d :: Dual) = Dual(x) - d
26
27 function -(d :: Dual)
28   Dual(-d.a, -d.b)
29 end
```

Derivadas - Valores duais

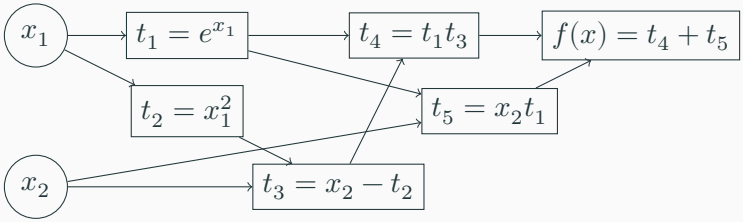
```
31 function *(d1 :: Dual, d2 :: Dual)
32   Dual(d1.a * d2.a, d1.a * d2.b + d2.a * d1.b)
33 end
34
35 function ^(d :: Dual, p :: Real)
36   if p == 0
37     return Dual(1, 0)
38   else
39     return Dual(d.a^p, p * d.a^(p - 1) * d.b)
40   end
41 end
42
43 function Base.inv(d :: Dual)
44   Dual(inv(d.a), -inv(d.a)^2 * d.b)
45 end
```

Derivadas - Valores duais

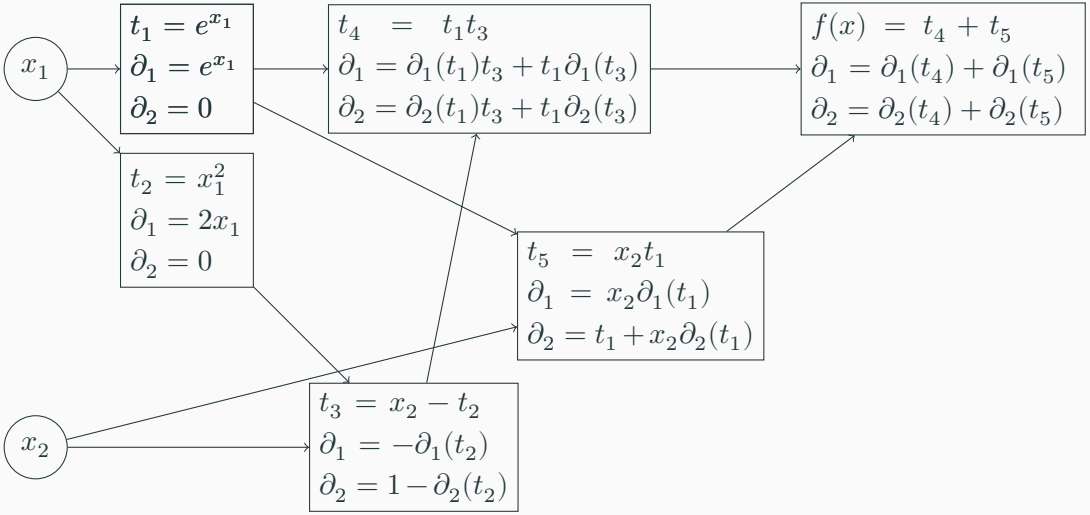
```
47 function Base.sqrt(d :: Dual)
48   Dual(sqrt(d.a), 0.5 / sqrt(d.a) * d.b)
49 end
50
51 function Base.exp(d :: Dual)
52   Dual(exp(d.a), d.b * exp(d.a))
53 end
```

Derivadas - Grafo de operações

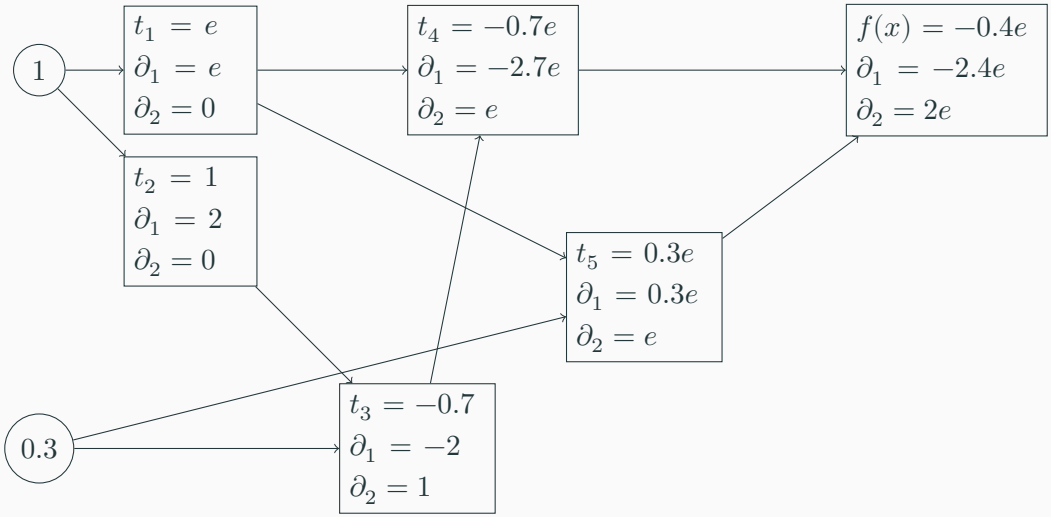
- Uma outra alternativa é montar um grafo de operações.
- Diferenciação simbólica e diferenciação automática fazem isso.
- Por exemplo $f(x) = e^{x_1}(x_2 - x_1^2) + x_2 e^{x_1}$.



Derivadas - Grafo de operações



Derivadas - Grafo de operações



Sistemas Lineares

Sistemas Lineares

- Existem diversos sistemas para serem resolvidos, é importante lidar com eles de acordo.
- No método de Newton, se a matriz é definida positiva, usamos Cholesky.
- Mas como saber se a matriz é definida positiva? Cholesky.
- Alternativamente, poderíamos resolver o sistema de Newton com gradientes conjugados, e também descobriríamos no caminho se a matriz é definida positiva.
- Note também que se a matriz for bastante esparsa, devemos implementar Cholesky com uma estratégia diferente.

Alternativas a Newton

- No processo de resolução de Newton, podemos perceber que a matriz não é definida positiva. O que fazer?
- Podemos desistir e chamar a tentativa de fracasso. Alternativamente podemos mudar para um método de gradiente.
- Temos algumas alternativas que podem ser um pouco mais viáveis.

Alternativas a Newton

- **Newton modificado:** Usar $B_k = \nabla^2 f(x_k) + \rho_k I$. Começamos com $\rho_k = 0$, e se a matriz não for definida positiva, aumentamos ρ_k. Fazemos busca linear na solução de $B_k d_k = -\nabla f(x_k)$.
- **Newton truncado com busca linear:** Tentar resolver $\nabla^2 f(x_k) d_k = -\nabla f(x_k)$ com gradientes conjugados, mas se a iteração falhar, usar a última aproximação para d_k. Fazemos busca linear em d_k. No pior caso, $d_k = -\nabla f(x_k)$, que é o primeiro passo tentado por GC.
- **Newton truncado com região de confiança / Steihaug-Toint**: Usamos gradientes conjugados para minimizar a quadrática, mas se algum passo ultrapassar Δ, ele é truncado. No caso de direção de curvatura negativa, seguimos ela até a borda.

Alternativas a Newton - Quase-Newton

- Uma classe de métodos são os de quase-Newton, definidos pela equação secante $B_{k+1}(x_{k+1} - x_k) = \nabla f(x_{k+1}) - \nabla f(x_k)$.
- As matrizes B_k são calculadas a partir de $B_0 = I$ e de uma atualização seguindo a equação secante.
- A aproximação mais famosa é a BFGS (Broyden-Fletcher-Goldfarb-Shanno):

$$B_{k+1} = B_k + \frac{y_k y_k^T}{y_k^T s_k} - \frac{B_k s_k s_k^T B_k}{s_k^T B_k s_k},$$

  onde $s_k = x_{k+1} - x_k$ e $y_k = \nabla f(x_{k+1}) - \nabla f(x_k)$.
- Essa aproximação é definida positiva se $s_k^T y_k > 0$.
- Não queremos, em geral, a matriz B_k explicitamente, e sim valores do tipo $B_k v$, que podem ser calculados iterativamente usando os valores s_j e y_j para $j = 0, 1, \ldots, k-1$.

Alternativas a Newton - Quase-Newton

- Note que nesse caso não usamos Cholesky, já que ele exige criar a matriz. Mas podemos usar gradiente conjugados.
- Uma variante comum é usar uma quantidade finita de vetores s_k e y_k. Se o número de vetores ultrapassar um valor pré-definido, nós jogamos fora os primeiros.
- Outra variante é usar a inversa de B_k, dada por $H_k = B_k^{-1}$ e pela fórmula

$$H_{k+1} = \left(I - \frac{s_k y_k^T}{y_k^T s_k}\right) H_k \left(I - \frac{y_k s_k^T}{y_k^T s_k}\right) + \frac{s_k s_k^T}{y_k^T s_k}.$$

- Nesse caso, a direção de busca é simplesmente $d_k = -H_k \nabla f(x_k)$, ou seja, muito mais fácil de se calcular.
- O método LBFGS com busca linear é um dos mais utilizados.

Alternativas a Newton - Gradiente Conjugados Não-linear

- Outro método é o de gradientes conjugados não-lineares.
- ...

Comparação de Algoritmos

Comparação de Algoritmos

- Uma questão importante é a comparação de dois métodos.
- Não é uma tarefa simples, e não existe uma resposta 100% aceita, mas existe o tradicional.
- Primeiro, as coisas mais importantes de se comparar são tempo e “trabalho”.
- No quesito de trabalho, comumente se compara número de avaliações de função, ou de fatorações, ou de multiplicações matriz-vetor.
- Uma essência do problema de comparação é que nem sempre um método funciona, e às vezes ele funciona até certo ponto.
- Uma questão às vezes trazida à tona é comparar também o valor de função, já que os métodos podem parar em mínimos locais.

Comparação de Algoritmos - Perfil de desempenho

- A maneira mais tradicional de se comparar métodos gerais de otimização é o perfil de desempenho.
- Esse método de comparação usa um único valor de custo, por exemplo, tempo.
- Caso o método não funcione para um problema, definimos que o custo é ∞.
- O método consiste em relativizar o custo de acordo com o mais rápido, e plotar a distribuição acumulada de problemas resolvidos.
- Notação: solvers $s \in S$ e problemas $p \in P$.

Comparação de Algoritmos - Perfil de desempenho

- Tempo de dois métodos para para resolver 5 problemas (t_{sp}):

t_{sp} **(s)**	Método 1	Método 2
Prob 1	3.4	3.7
Prob 2	1.8	3.5
Prob 3	10.4	∞
Prob 4	0.8	0.5
Prob 5	0.9	0.1

Comparação de Algoritmos - Perfil de desempenho

- Tempo de dois métodos para para resolver 5 problemas (t_{sp}):
- A razão é a linha pelo mínimo $\tau_{sp} = \dfrac{t_{sp}}{\min_s t_{sp}}$.

t_{sp} **(s)**	Método 1	Método 2
Prob 1	3.4	3.7
Prob 2	1.8	3.5
Prob 3	10.4	∞
Prob 4	0.8	0.5
Prob 5	0.9	0.1

$\rightarrow$

τ_{sp}	Método 1	Método 2
Prob 1	1.0	1.09
Prob 2	1.0	1.94
Prob 3	1.0	∞
Prob 4	1.6	1.0
Prob 5	9	1.0

Comparação de Algoritmos - Perfil de desempenho

- Tempo de dois métodos para para resolver 5 problemas (t_{sp}):
- A razão é a linha pelo mínimo $\tau_{sp} = \dfrac{t_{sp}}{\min_s t_{sp}}$.
- Porcetagem acumulada de problemas $\rho_s(\tau) = \dfrac{|\{p : \ \tau_{sp} \leq \tau\}|}{|P|}$

τ_{sp}	Método 1	Método 2
Prob 1	1.0	1.09
Prob 2	1.0	1.94
Prob 3	1.0	∞
Prob 4	1.6	1.0
Prob 5	9	1.0

$\rightarrow$

ρ_s	Método 1	Método 2
1.0	3 / 5	2 / 5
1.09	3 / 5	3 / 5
1.6	4 / 5	3 / 5
1.94	4 / 5	4 / 5
9	5 / 5	4 / 5

Comparação de Algoritmos - Perfil de desempenho

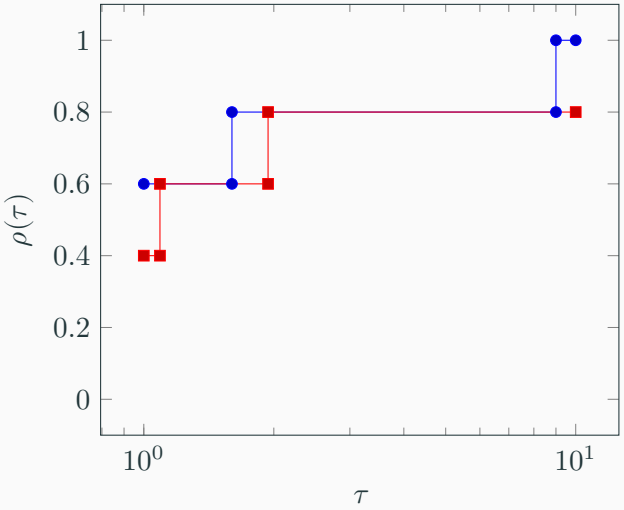
- Tempo de dois métodos para para resolver 5 problemas (t_{sp}):
- A razão é a linha pelo mínimo $\tau_{sp} = \dfrac{t_{sp}}{\min_s t_{sp}}$.
- Porcetagem acumulada de problemas $\rho_s(\tau) = \dfrac{|\{p : \ \tau_{sp} \leq \tau\}|}{|P|}$

τ_{sp}	Método 1	Método 2
Prob 1	1.0	1.09
Prob 2	1.0	1.94
Prob 3	1.0	∞
Prob 4	1.6	1.0
Prob 5	9	1.0

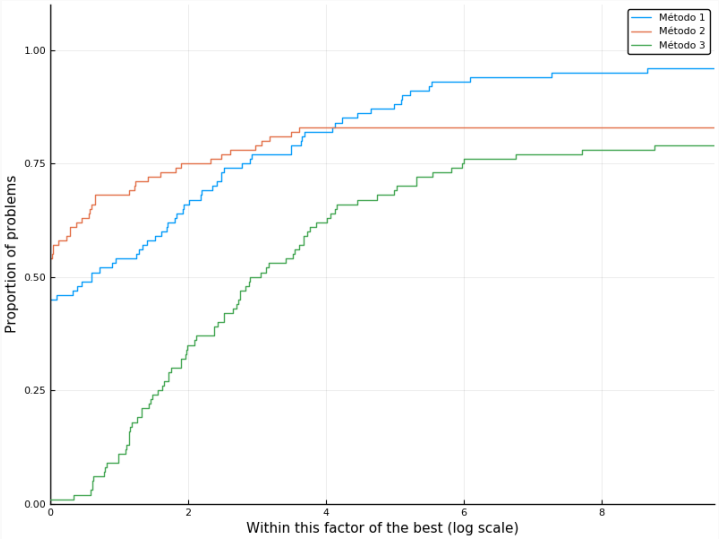
$\rightarrow$

	Método 1	Método 2
1.0	0.6	0.4
1.09	0.6	0.6
1.6	0.8	0.6
1.94	0.8	0.8
9	1.0	0.8

Comparação de Algoritmos - Perfil de desempenho



Comparação de Algoritmos - Perfil de desempenho



FIM