

# Lab 11: Otimização sem restrição

## INF1608 – Análise Numérica

Leonardo Quatrin Campagnolo

lquatrin@tecgraf.puc-rio.br

Departamento de Informática, PUC-Rio

Neste trabalho, iremos implementar alguns métodos de otimização sem restrição, vistos na última aula.

1. O **Método da Seção Áurea** (msa) consiste em determinar o mínimo de uma função em um intervalo  $[a, b]$ , considerando que a função objetiva é unimodal. Em cada iteração, duas estimativas  $x_1$  e  $x_2$  são definidas para determinar em qual intervalo o mínimo da função deve estar. Como vimos na aula passada, se considerarmos o intervalo  $[a, b] = [0, 1]$ ,  $x_2$  estará posicionado em  $g = \frac{\sqrt{5}-1}{2}$ , e  $x_1$  em  $1 - g$ . Para um intervalo  $[a, b]$  qualquer,  $x_1$  e  $x_2$  são calculados da seguinte forma:

$$\begin{aligned}x_1 &= a + (1-g)(b-a) \\ x_2 &= a + g(b-a)\end{aligned}$$

A partir dos pontos  $x_1$  e  $x_2$ , um novo intervalo é definido da seguinte forma: Se  $f(x_1) \leq f(x_2)$ , ajusta intervalo para  $[a, b \leftarrow x_2]$ . Senão, ajusta intervalo para  $[a \leftarrow x_1, b]$ . Podemos implementar este método apenas atualizando os limites  $a$  e  $b$  do intervalo em cada iteração (veja o algoritmo no slide da aula passada). No entanto, é importante armazenar os valores de  $f(x_1)$  e  $f(x_2)$  para *minimizar o número de avaliações da função objetiva*. Ao final, deve-se retornar o ponto médio do intervalo  $\frac{a+b}{2}$ .

Implemente o **Método da Seção Áurea**. Sua função deve receber como parâmetros o intervalo  $[a, b]$ , a *função objetiva*,  $f(x)$ , e um ponteiro  $x_{min}$  que deve ser preenchido com o valor de  $x$  onde a função tem valor mínimo dado pelo método. A função deve retornar o número de iterações executadas para se alcançar o resultado, seguindo o protótipo abaixo. O método deve retornar após 50 iterações ou após o erro chegar em  $|x - x_{sol}| \leq 10^{-5}$  (lembre-se que o erro progressivo em cada iteração pode ser estimado em  $\frac{b-a}{2}$ ).

```
int msa (double a, double b, double (*f) (double x),
        double tol, double* xmin);
```

2. O **Método da Interpolação Parabólica Sucessiva** (mips) parte de 3 estimativas iniciais  $r$ ,  $s$  e  $t$ , ajusta uma parábola interpolante e estima que o mínimo da parábola,  $x$ , é um próximo candidato a ser o ponto de mínimo da função:

$$x = \frac{r+s}{2} - \frac{(f(s) - f(r))(t-r)(t-s)}{2[(s-r)(f(t) - f(s)) - (f(s) - f(r))(t-s)]}$$

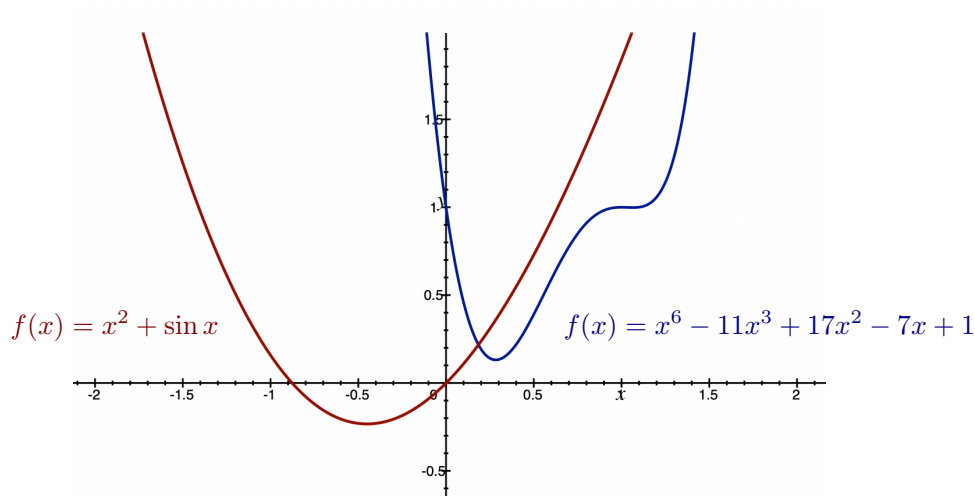
Se o denominador da expressão acima for zero (na prática se  $< 10^{-10}$ ), faz-se  $x = (r + s + t)/3$ . Essa estimativa substitui a menos recente, isto é,  $r$  passa a ter o valor de  $s$ ,  $s$  de  $t$  e  $t$

de  $x$ . O método continua iterando até que o critério de convergência seja atendido ou que um número máximo de iterações seja alcançado. *Após 3 iterações*, pode-se adotar como critério de convergência a diferença entre o valor da função nas duas últimas estimativas:  $|f(s) - f(t)| \leq \epsilon$ , onde  $\epsilon = 10^{-6}$  representa a tolerância requerida. O valor de  $x$  onde a função tem valor mínimo é então expresso pela média das estimativas:  $x_{min} = \frac{s+t}{2}$ .

Implemente uma função que codifica o **Método da Interpolação Parabólica Sucessiva** como descrito acima. Sua função deve receber como parâmetros a primeira estimativa inicial,  $r$ , e um valor de incremento,  $\delta$ . As duas outras estimativas iniciais devem ser dadas por:  $s = r - \delta$  e  $t = r + \delta$ . A função também recebe como parâmetros a *função objetivo*,  $f(x)$ , e um ponteiro  $x_{min}$  que deve ser preenchido com o valor de  $x$  onde a função tem valor mínimo dado pelo método. A função deve ainda retornar o número de iterações executadas para se alcançar o resultado, seguindo o protótipo abaixo. Se o método não convergir em até 50 iterações, a função deve retornar zero. *A implementação deve minimizar o número de avaliações da função  $f(x)$ .*

```
int mips (double r, double delta, double (*f) (double x),
         double tol, double* xmin);
```

3. Para testar seu código, determine os valores mínimos das funções ilustradas abaixo. Compare o uso de diferentes estimativas e o número de iterações necessário para o método convergir.



Agrupe os protótipos das funções pedidas em um módulo “otimizacao.h” e as implementações em um módulo “otimizacao.c”. Escreva o teste em outro módulo “main.c”.

**Entrega:** O código fonte deste trabalho (isto é, os arquivos “otimizacao.c”, “otimizacao.h” e “main.c”) deve ser enviado via página da disciplina no EAD até o final da aula. O sistema receberá trabalhos com atraso (com perda de 1 ponto na avaliação) até 2 horas após o final da aula.