Unconstraint Optimization (one variable functions)

Prof. Cesar de Prada
Dpt. Systems Engineering and Automatic Control
UVA
prada@autom.uva.es

Unconstraint Optimization

$$\min_{x} J(x)$$

$$x \in R^n$$

Unconstraint optimization methods are important because:

✓Some problems do not have constraints in their variables

✓Allow an easy introduction of concepts that will be used in NLP problems

✓Many optimization methods use unconstraint methods in a phase of its implementation

✓Some NLP problems can be reformulated as unconstraint ones

Outline

- Examples
- Theoretical solution
- Optimizing a function of one variable
  - Newton type methods
  - Bracketing methods
  - Polynomial approximation methods
- Multivariate methods
  - Gradient based algorithms
  - Newton type algorithms
  - Gradient free algorithms
- Software

There exist many methods. Only some of them will be considered in the course

Redlich-Kwong's equation

Empirical relation among:

Pressure P

Temperature T

Molar volume v

of a real gas

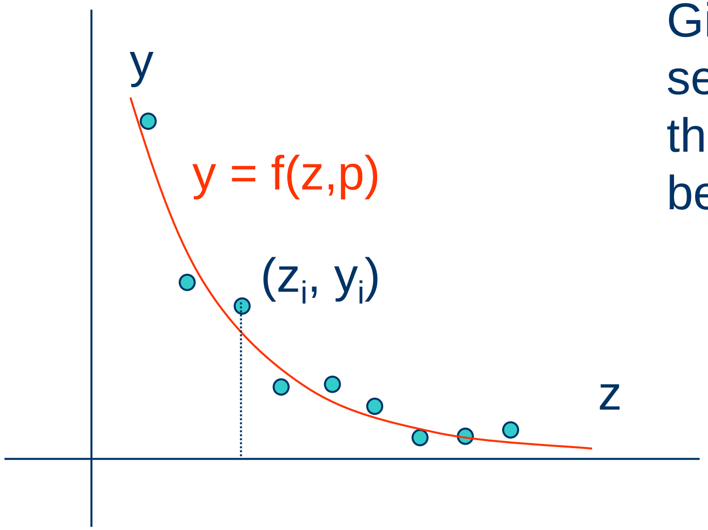
$$P = \frac{RT}{v-b} - \frac{a}{v(v+b)\sqrt{T}}$$

a and b are unknown coefficients that must be estimated using experimental data

Example: CO_2 data

volumen molar v	Temperatura T	Presión P
500	273	33
500	323	43
600	373	45
700	273	26
600	323	37
700	373	39
400	272	38
400	373	63,6

Data fit as an optimization problem



Given a function y = f(z, p) that should fit a set of N couples of data (z_i , y_i), estimate the unknown parameters p that provides the best fit

$$\min_{p} \sum_{i=1}^{N} (y_i - f(z_i, p))^2$$

The problem can be formulated as the minimization of the sum of squares of the residuals $y_i - f(z_i,p)$ with respect to the function parameters p

An unconstraint optimization problem

volumen molar v	Temperatura T	Presión P
500	273	33
500	323	43
600	373	45
700	273	26
600	323	37
700	373	39
400	272	38
400	373	63,6

$$P = \frac{RT}{v-b} - \frac{a}{v(v+b)\sqrt{T}}$$

$$\min_{a,b} \sum_{i=1}^{8} \left(P_i - \left(\frac{RT_i}{v_i - b} - \frac{a}{v_i(v_i+b)\sqrt{T_i}} \right) \right)^2$$

$$x_1 = a \quad x_2 = b$$

$$\min_{x} J(x)$$

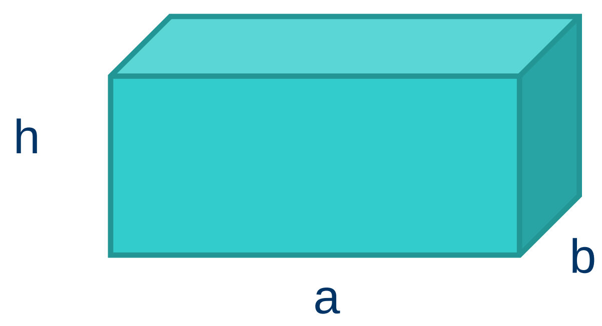
$$x = [x_1, x_2]' \in R^n$$

$$\min_{x_1,x_2} \sum_{i=1}^{8} \left(P_i - \left(\frac{RT_i}{v_i - x_2} - \frac{x_1}{v_i(v_i+x_2)\sqrt{T_i}} \right) \right)^2$$

Building a open tank

We want to construct a tank like the one in the figure, whose base length is 3 times the base width. The material used to build the bottom cost 10€/m^2 and the material used to build the sides is cheaper: 5€/m^2

If the tank must have a volume of 60 m^3 determine the dimensions that will minimize the cost of building the tank.



An unconstraint optimization problem

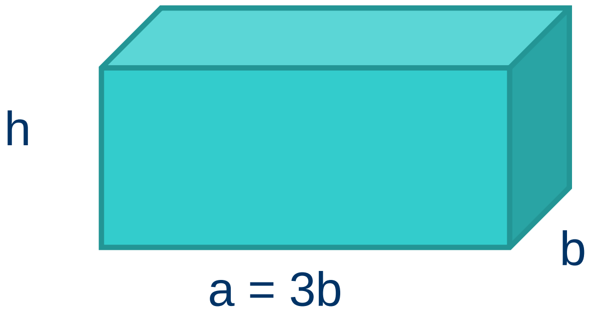
Cost = 10(3b . b) + 5(2 (3bh + bh)) = $30b^2$ + 40 bh

Volume = 3b . b .h = 60 h = $20/b^2$

Min $30b^2$ + 800/b

$$\min_{x} J(x) = 30x^2 + \frac{800}{x}$$

$$x = b \in R^n$$



Extremum analytical conditions

$$\min_x J(x)$$

$$x \in R^n$$

In unconstraint optimization problems there exist a set of analytical conditions for a point being the solution

Necessary condition

$$\left.\frac{\partial J(x)}{\partial x}\right|_{x^*} = 0$$

The hessian H determines the character of the possible optimum

$$H = \left.\frac{\partial^2 J(x)}{\partial x^2}\right|_{x^*}$$

Extremum analytical conditions

$$J(x) = J(x^*) + \left.\frac{\partial J}{\partial x}\right|_{x^*}' (x - x^*) + \frac{1}{2}(x - x^*)' \left.\frac{\partial^2 J(x)}{\partial x^2}\right|_{x^*} (x - x^*) + \ldots.$$

$$\left.\frac{\partial J}{\partial x}\right|_{x^*} = 0$$

x^* that satisfy the equation is called a stationary point

$$J(x) - J(x^*) \approx \frac{1}{2}(x - x^*)' \left.\frac{\partial^2 J(x)}{\partial x^2}\right|_{x^*} (x - x^*)$$

2nd order approximation

If H(x*) is PD or PSD, J(x) presents a minimum in x^*

If H(x*) is ND o NSD, J(x) presents a maximum in x*

If H(x*) is non definite there is no extremum, J(x) presents a saddle point in x^*

Extremum analytical conditions

$$\min_{x} J(x)$$

$$x \in R^{n}$$

$$\left.\frac{\partial J(x)}{\partial x}\right|_{x^{*}} = 0$$

The analytical solution of the problem usually is a non-linear equation difficult to solve, hence, very often is better to use direct numerical methods to solve the optimization problem

Single variable Optimization

$$\min_{x} J(x)$$

$$x \in R$$

There are important problems because:

✓They are used in an intermediate step of other algorithms

✓Many problems are single variable ones

There exist several methods based on different criteria:

a) Solving the analytical solution

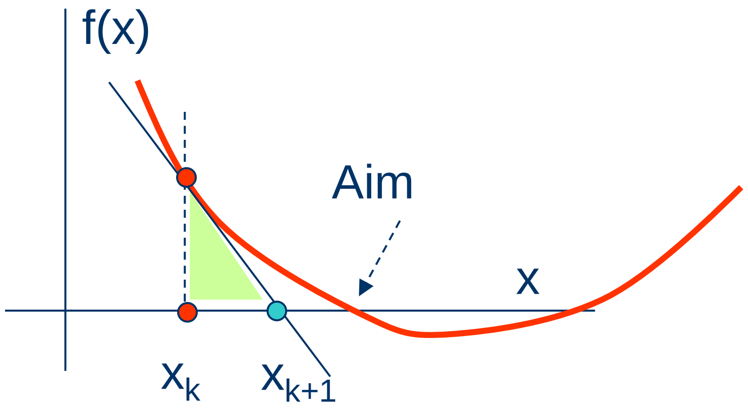
b) Minimizing the size of an interval containing the solution

c) Approximating the function by a polynomial

d) Other

Assumption: J(x) is unimodal and only have a local minimum

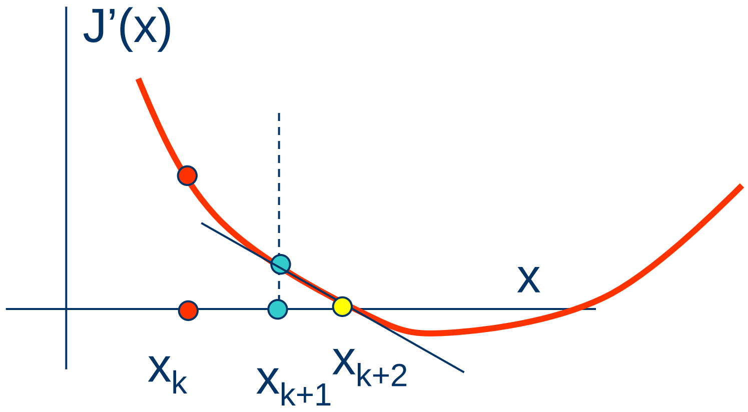
Newton-Raphson's method for solving non-linear equations f(x)=0



$$f'(x_k) = \frac{f(x_k)}{x_{k+1} - x_k}$$

$$x_{k+1} = x_k - \frac{f(x_k)}{f'(x_k)}$$

Each iteration corresponds to a linear approximation



Newton’s method (Newton-Raphson)

$$\min_{x} J(x)$$

$$x \in R$$

The analytical solution is J’(x)=0

How to solve this equation?

We can apply the Newton’s method for solving non-linear equations f(z) = 0:

$$z_{k+1} = z_k - \frac{f(z_k)}{f'(z_k)}$$

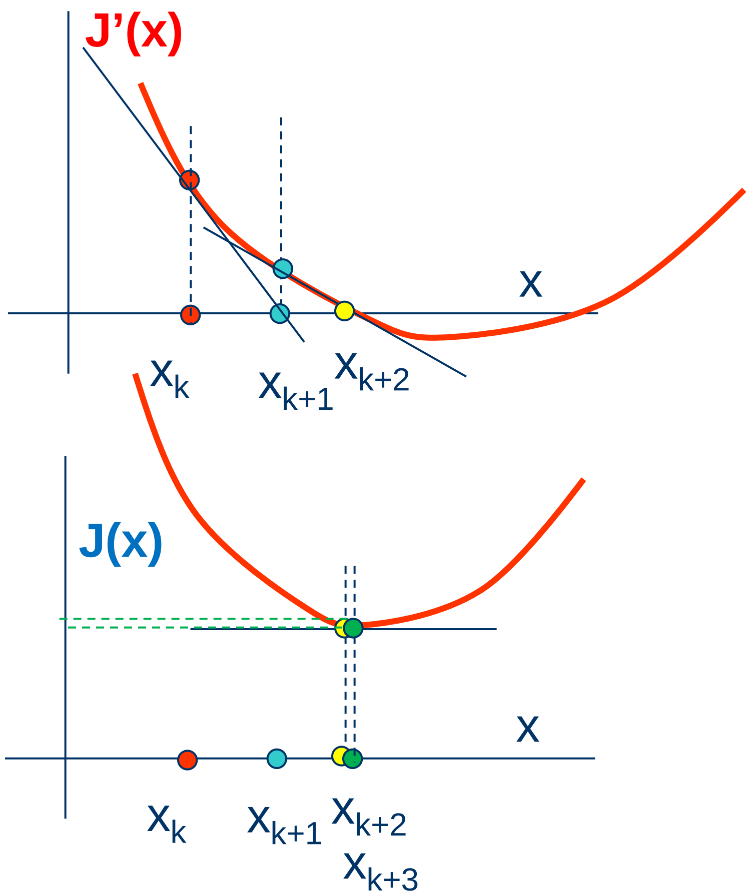
Which leads to

$$x_{k+1} = x_k - \frac{J'(x_k)}{J''(x_k)}$$

Starting from an initial value x_0 one can generate a sequence x_1, x_2, ... of values that improve the solution until a stopping criterion is satisfied

How many iterations?

$$x_{k+1} = x_k - \frac{J'(x_k)}{J''(x_k)}$$



Ending criteria:

$| J'(x_k) | < \varepsilon$

$| x_{k+1} - x_k | < \varepsilon$

$| J(x_{k+1}) – J(x_k)| < \varepsilon$

$k > N$

The main difficulty is associated to the computation of the derivatives, which may be also a time consuming task.

Derivatives

- If it is not possible to compute the derivatives analytically, they can be approximated by:

$$J'(x_k) = \frac{J(x_k + \sigma) - J(x_k - \sigma)}{2\sigma}$$

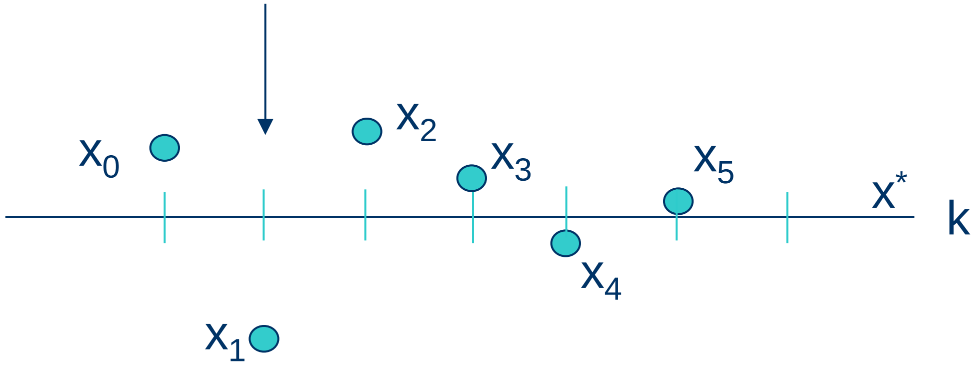
σ >0 small

$$J''(x_k) = \frac{J(x_k + \sigma) - 2J(x_k) + J(x_k - \sigma)}{\sigma^2}$$

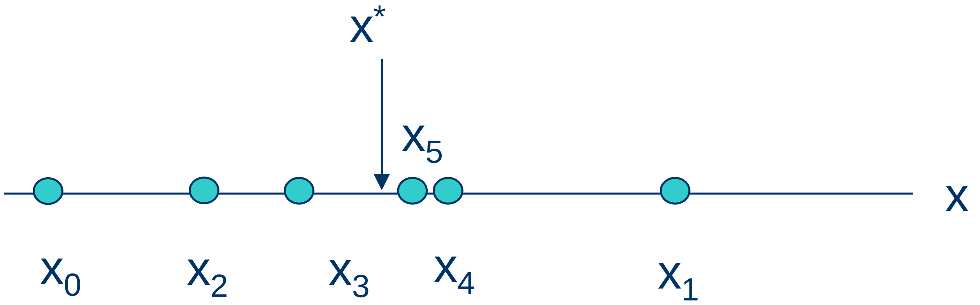
Hence:

$$x_{k+1} = x_k - \frac{J(x_k + \sigma) - J(x_k - \sigma)}{J(x_k + \sigma) - 2J(x_k) + J(x_k - \sigma)} \frac{\sigma^2}{2\sigma}$$

Convergence



$$x_{k+1} = x_k - \frac{J'(x_k)}{J''(x_k)}$$



We would say that the solution converges to x^* when the sequence of values x_k generated by the algorithm verifies:

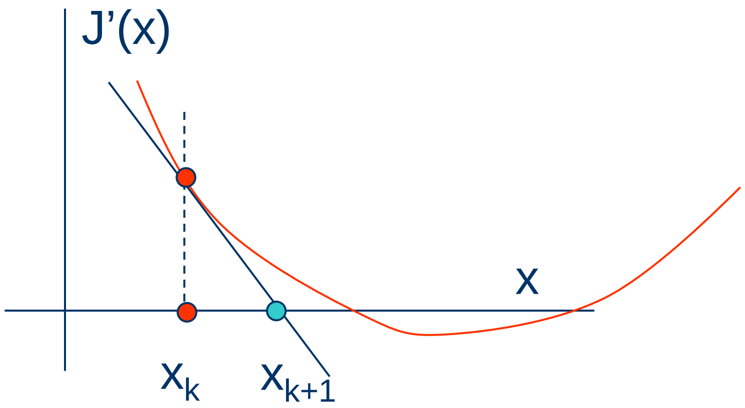
$$\left\| x_{k+1} - x^* \right\| \leq c \left\| x_k - x^* \right\|^p$$

$$0 < c < 1$$

From a certain k, so that the points x_k are closer and closer to x^*
c rate of convergence, p order of convergence

Newton's method

$$x_{k+1} = x_k - \frac{J'(x_k)}{J''(x_k)}$$



Advantages:

Quadratic local convergence

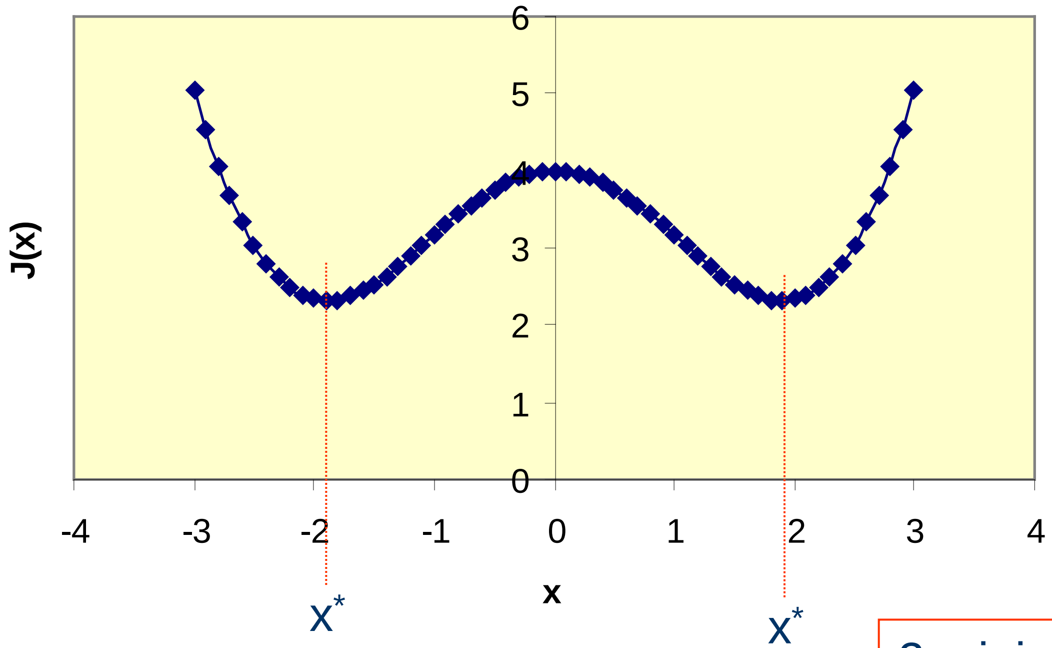
Inconveniences:

First and second derivatives must be computed or estimated

If $J''(x) \rightarrow 0$ it converges slowly

If x_0 is far away from x^* the method may not converge

Example $J(x) = x^2 + 4\cos(x)$



Minimize J(x)

$J'(x^*) =$

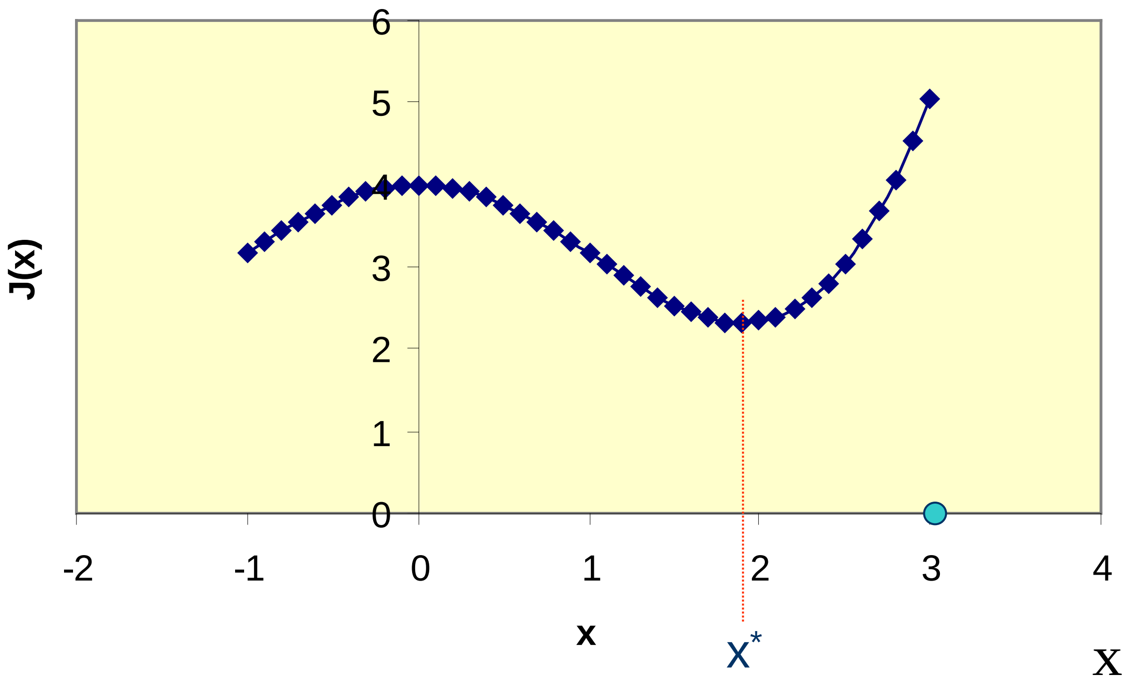
$= 2x^* - 4\text{sen}(x^*) = 0$

Using a numerical method:

$x^* = \pm\ 1.895..\ \ ,\ \ 0$

2 minimums (local, global) and one maximum

Minimize $J(x) = x^2 + 4\cos(x)$



Minimize J(x) using Newton

$J'(x) = 2x - 4\text{sen}(x)$

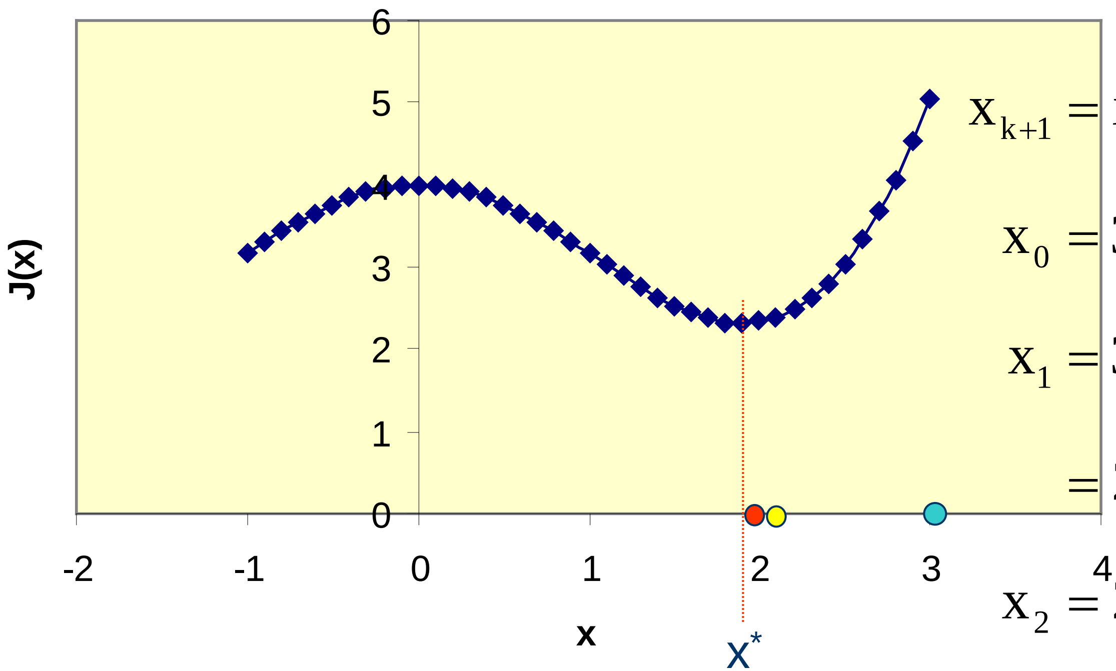
$J''(x) = 2 - 4\cos(x)$

$$x_{k+1} = x_k - \frac{J'(x_k)}{J''(x_k)}$$

$$x_{k+1} = x_k - \frac{x_k - 2\text{sen}(x_k)}{1 - 2\cos(x_k)}$$

$$x_0 = 3$$

Minimize $J(x) = x^2 + 4\cos(x)$



$$x_{k+1} = x_k - \frac{x_k - 2\operatorname{sen}(x_k)}{1 - 2\cos(x_k)}$$

$$x_0 = 3$$

$$x_1 = 3 - \frac{3 - 2\operatorname{sen}(3)}{1 - 2\cos(3)} =$$

$$= 2.088$$

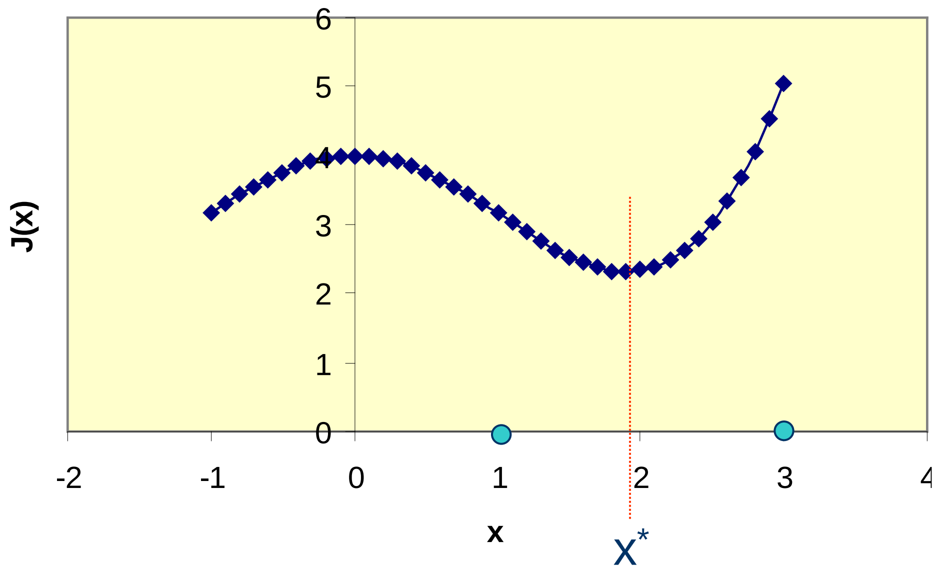
$$x_2 = 2.088 - \frac{2.088 - 2\operatorname{sen}(2.088)}{1 - 2\cos(2.088)}$$

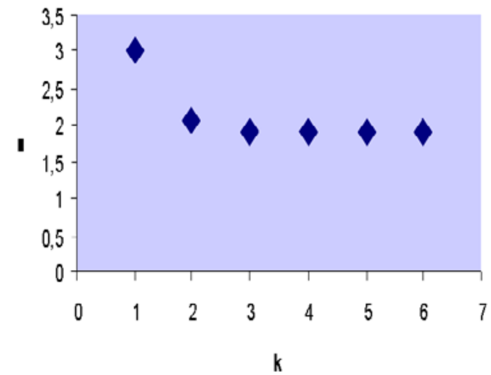
$$= 1.91$$

$$x_3 = \ldots\ldots$$

$$x_{k+1} = x_k - \frac{J'(x_k)}{J''(x_k)}$$

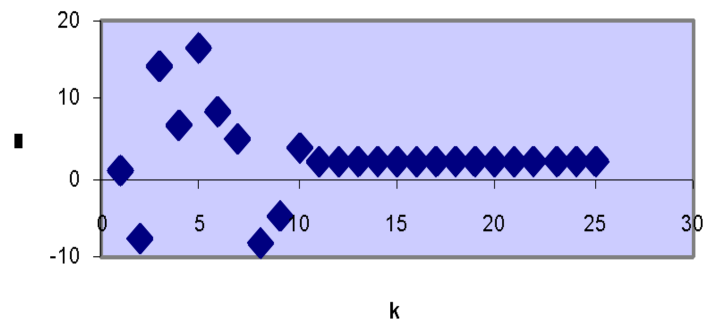
Example $J(x) = x^2 + 4\cos(x)$





$x_0 = 3$

$x_0 = 1$

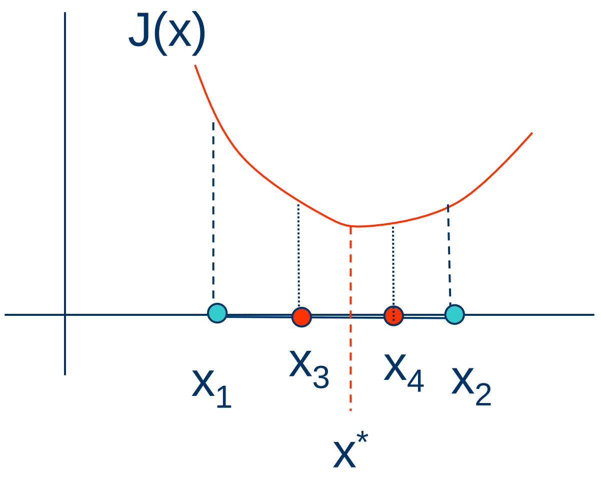


It converges at different speeds and to different points depending on the initial guess

Excel

Bracketing methods

They generate a series of intervals of decreasing sizes $[x_1 , x_2]$, $[x_3, x_2]$,.... containing the optimum, until the required precision is met



2 steps:

1. Find an initial interval containing x^*
2. Narrow the initial bracket until the required precision is met

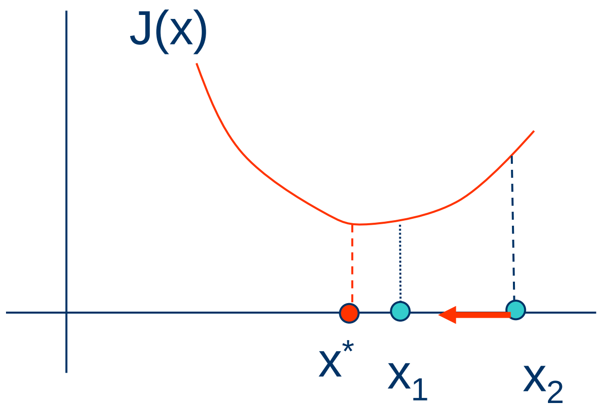
1 Initial (Semi)Bracket

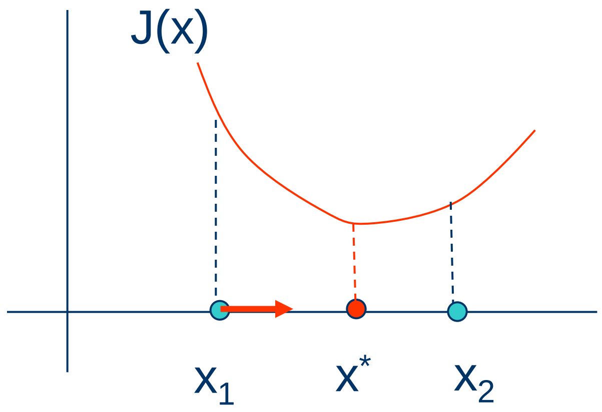
Choose any two points $x_1 < x_2$

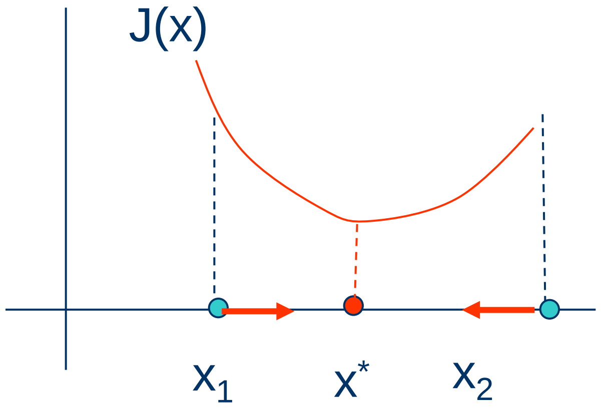
If $J(x_1) < J(x_2) \rightarrow x^* < x_2$

If $J(x_1) > J(x_2) \rightarrow x^* > x_1$

If $J(x_1) = J(x_2) \rightarrow x^* \in [\, x_1, x_2]$







1 Initial Bracket

Knowing an initial semi-bracket p.e. $[x_0, \infty)$ containing x*, in order to find an initial bracket, one can generate the following sequence of points:

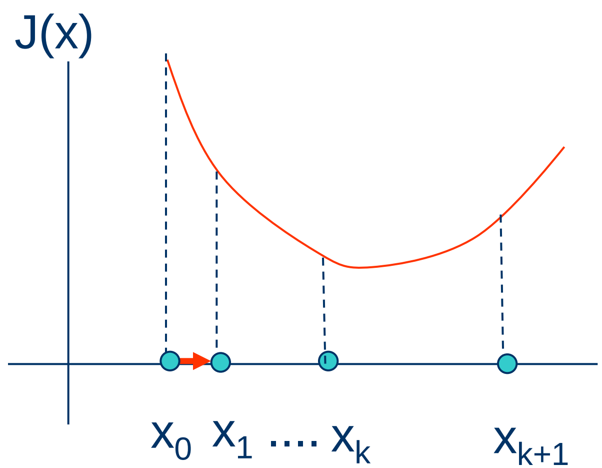
$$x_1 = x_0 + \delta$$

$$x_2 = x_0 + 2\delta$$

$$x_3 = x_0 + 2^2\delta$$

....

$$x_k = x_0 + 2^{k-1}\delta$$



Comparing $J(x_k)$ until:

$J(x_{k-1}) > J(x_k) \leq J(x_{k+1})$

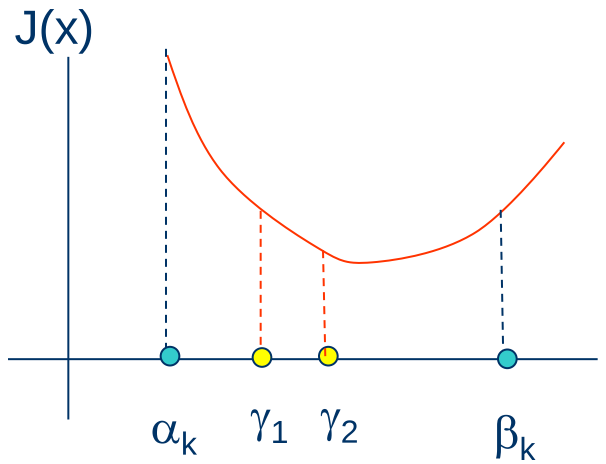
The initial bracket is: $[x_{k-1} , x_{k+1}]$

δ Positive or negative according to the semi bracket
Good compromise precision/number of iterations

2 Narrowing the bracket

If in iteration number k the bracket is:

$[\alpha_k , \beta_k]$, One can narrow its lenght $L_k = \beta_k - \alpha_k$ evaluating J(x) in two points $\gamma_1 < \gamma_2$ inside the bracket



$$J(\gamma_1) > J(\gamma_2) \Rightarrow [\alpha_{k+1}, \beta_{k+1}] = [\gamma_1, \beta_k]$$

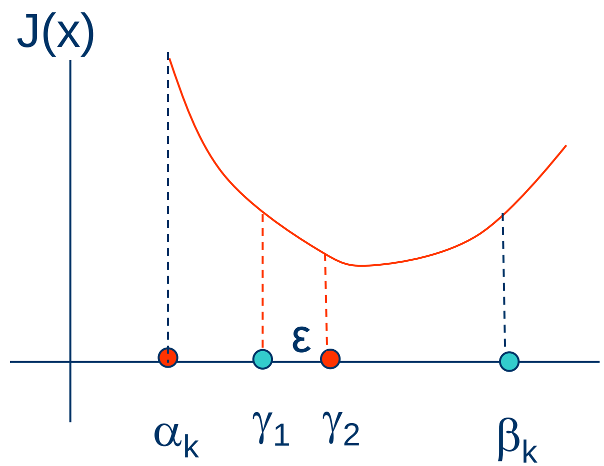
$$J(\gamma_1) < J(\gamma_2) \Rightarrow [\alpha_{k+1}, \beta_{k+1}] = [\alpha_k, \gamma_2]$$

$$J(\gamma_1) = J(\gamma_2) \Rightarrow [\alpha_{k+1}, \beta_{k+1}] = [\gamma_1, \gamma_2]$$

How to choose the two internal points?

Many methods: Fibonacci, ε-minimax, Golden section,….

2 Narrowing the bracket: ε-minimax



$$\gamma_1 + \varepsilon - \alpha_k = \beta_k - \gamma_1$$

$$\boxed{\begin{aligned} \gamma_1 &= \frac{\beta_k + \alpha_k}{2} - \frac{\varepsilon}{2} \\ \gamma_2 &= \frac{\beta_k + \alpha_k}{2} + \frac{\varepsilon}{2} \end{aligned}}$$

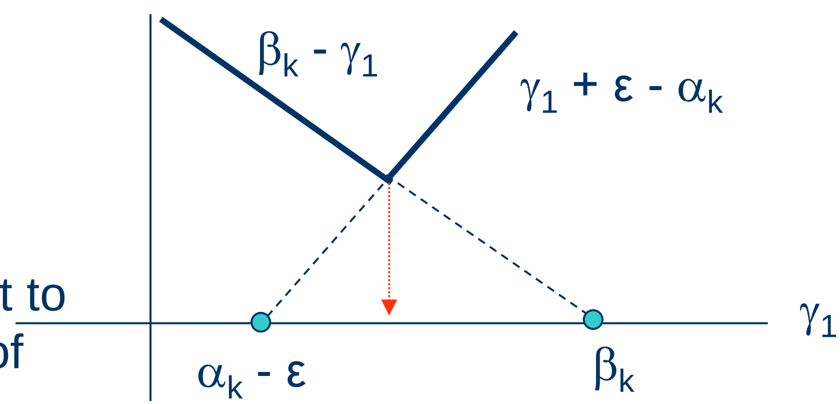
Symetrical with respect to the centre of the interval

Criterion: minimize the length of the biggest of the two possible intervals resulting in the next step

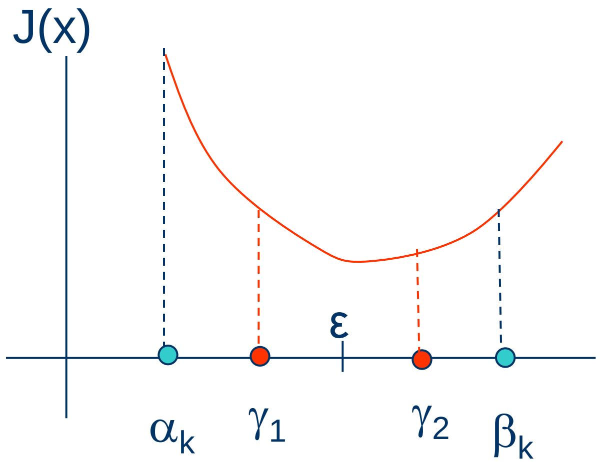
Min max { $\gamma_2 - \alpha_k$, $\beta_k - \gamma_1$, $\gamma_2 - \gamma_1$}

If $\quad \gamma_2 = \gamma_1 + \varepsilon$

Min max{$\gamma_1 + \varepsilon - \alpha_k$, $\beta_k - \gamma_1$}



2 Narrowing the bracket: ε-minimax



$\gamma_2 - \alpha_k = \beta_k - \gamma_1$

$$\gamma_1 = \frac{\beta_k + \alpha_k}{2} - \frac{\varepsilon}{2}$$

$$\gamma_2 = \frac{\beta_k + \alpha_k}{2} + \frac{\varepsilon}{2}$$

$$L_{k+1} = \gamma_2 - \alpha_k = \frac{\beta_k + \alpha_k}{2} + \frac{\varepsilon}{2} - \alpha_k =$$

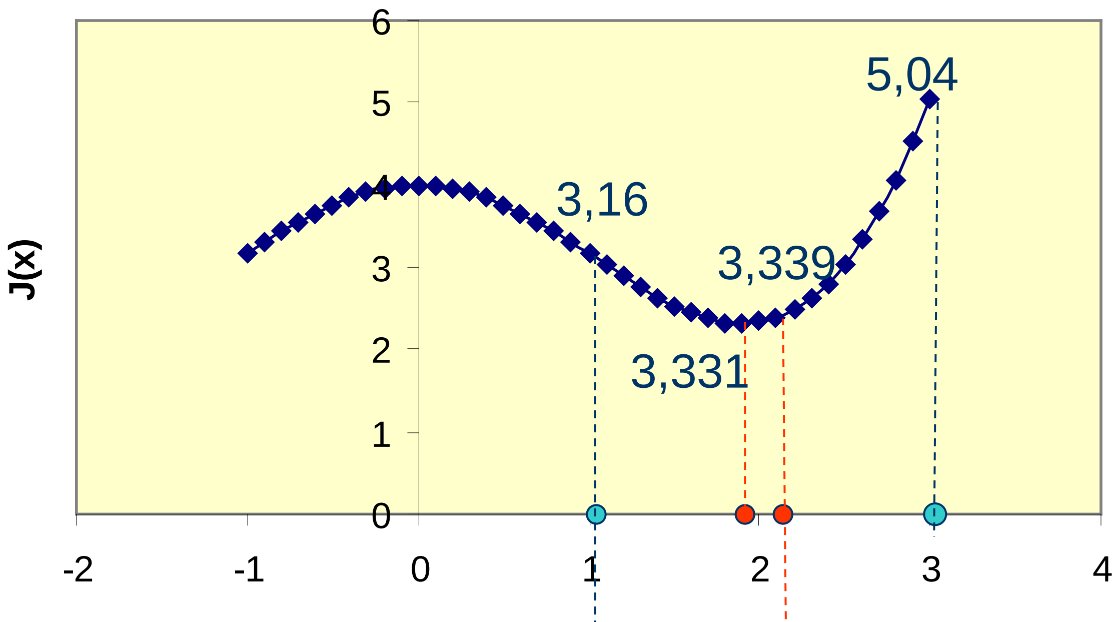
$$= \frac{\beta_k - \alpha_k}{2} + \frac{\varepsilon}{2} = \frac{L_k + \varepsilon}{2}$$

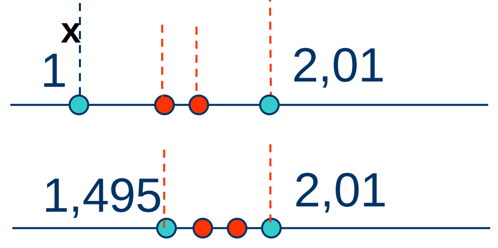
ε should be chosen as small as possible in order to reduce the length of the next bracket

In N steps:

$$\boxed{L_N = \frac{L_0 + (2^N - 1)\varepsilon}{2^N} \approx \frac{L_0}{2^N}}$$

Example $J(x) = x^2 + 4\cos(x)$





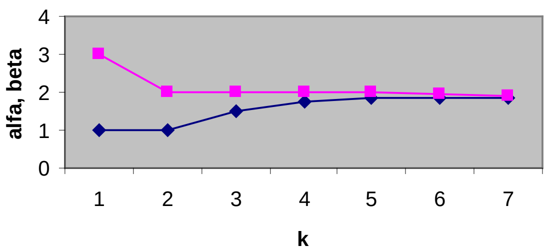
Minimize J(x)

initial bracket [1,3]

$\varepsilon = 0.02$

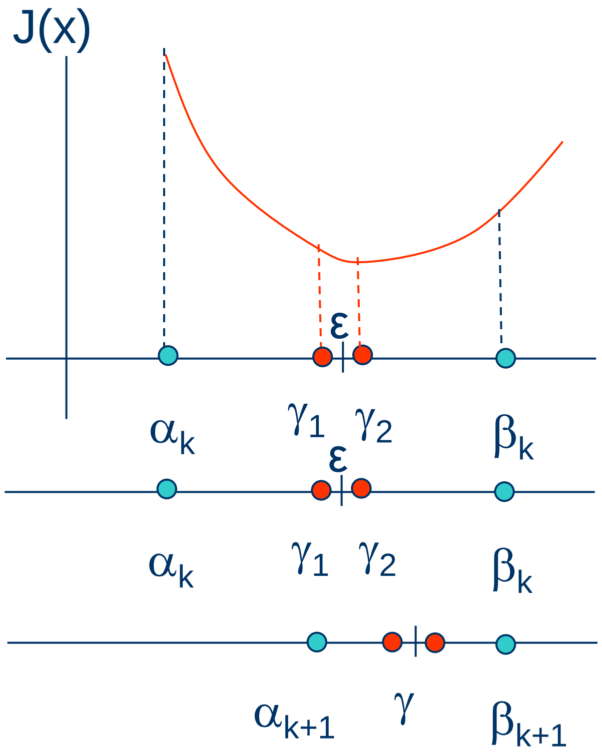
First two intermediate points: $(3+1)/2 \pm \varepsilon/2$

1,99, 2.01



Excel

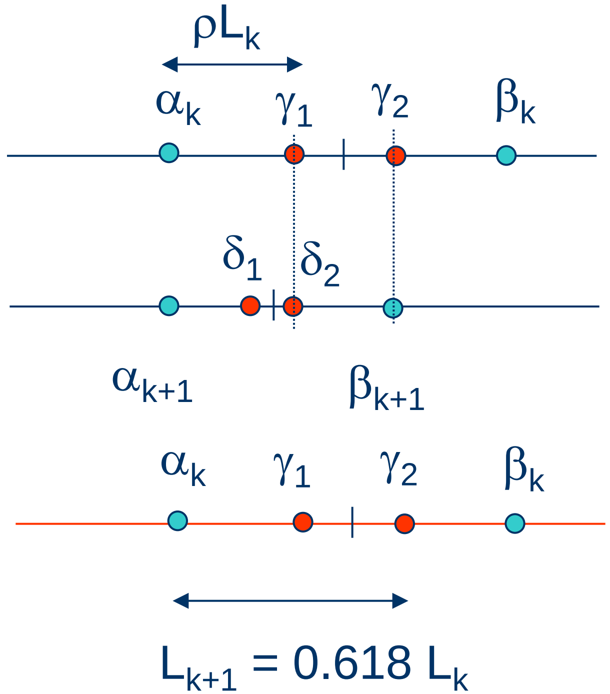
Golden section



The ε – minimax method computes the value of J(x) at two internal points but it does not uses this information in the following iteration

The same problem can be stated adding the condition that one of the internal points, and hence its associated cost J(x), can be used in the following iteration: This is the idea behind the Golden section method

Golden section method



$$\boxed{L_N \approx 0.618^N L_0}$$

$$\gamma_1 - \alpha_k = \rho(\beta_k - \alpha_k) = \rho L_k = \beta_k - \gamma_2$$

$$\gamma_2 - \gamma_1 = (1-2\rho)L_k$$

$$\gamma_2 - \alpha_k = (1-\rho)L_k$$

$$\gamma_1 - \alpha_k = \delta_2 - \alpha_{k+1} = (1-\rho)L_{k+1} =$$

$$= (1-\rho)(\gamma_2 - \alpha_k)$$

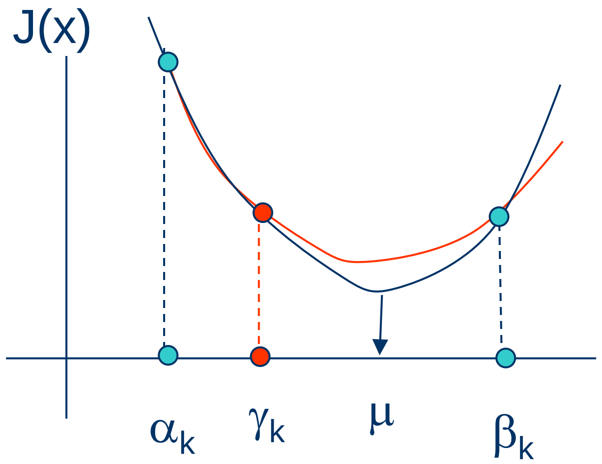
$$\rho L_k = (1-\rho)^2 L_k$$

$$\rho^2 - 3\rho + 1 = 0$$

$$\rho = \frac{3 \pm \sqrt{5}}{2} = \begin{cases} \text{is greater than } 0.5 \\ 0.382 \end{cases}$$

$$1 - \rho = 0.618$$

Polynomial approximation methods (2^{nd} order)



$P(x) = a + bx + cx^2$

$P(\alpha_k) = a + b\,\alpha_k + c\,\alpha_k^{\,2} = J(\alpha_k)$

$P(\gamma_k) = a + b\,\gamma_k + c\,\gamma_k^{\,2} = J(\gamma_k)$

$P(\beta_k) = a + b\,\beta_k + c\,\beta_k^{\,2} = J(\beta_k)$

If the value of J(x) is known at some points (for instance, 3 points) in the interval $[\alpha_k , \beta_k]$ it is possible to compute a polynomial P(x) (of second order) passing trough these points and approximating J(x) in the interval.

Then the analytical minimum of the approximating polynomial will provide a new point that will be used in reducing the length of the interval, but evaluating J(x) only once every iteration.

Polynomial approximation methods (2^{nd} order)

$P(x) = a + bx + cx^2$

$P(\alpha_k) = a + b\,\alpha_k + c\,\alpha_k{}^2 = J(\alpha_k)$

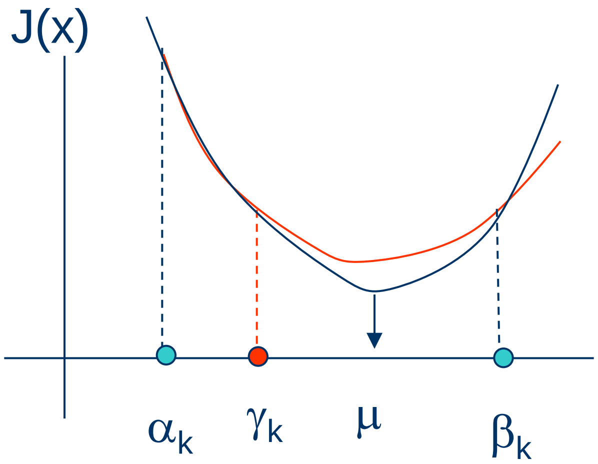
$P(\gamma_k) = a + b\,\gamma_k + c\,\gamma_k{}^2 = J(\gamma_k)$

$P(\beta_k) = a + b\,\beta_k + c\,\beta_k{}^2 = J(\beta_k)$

Set of 3 linear equations in a, b, c that can be solved:

$$c = \frac{J(\beta_k) - J(\gamma_k)}{(\beta_k - \alpha_k)(\beta_k - \gamma_k)} + \frac{J(\alpha_k) - J(\gamma_k)}{(\beta_k - \alpha_k)(\gamma_k - \alpha_k)}$$

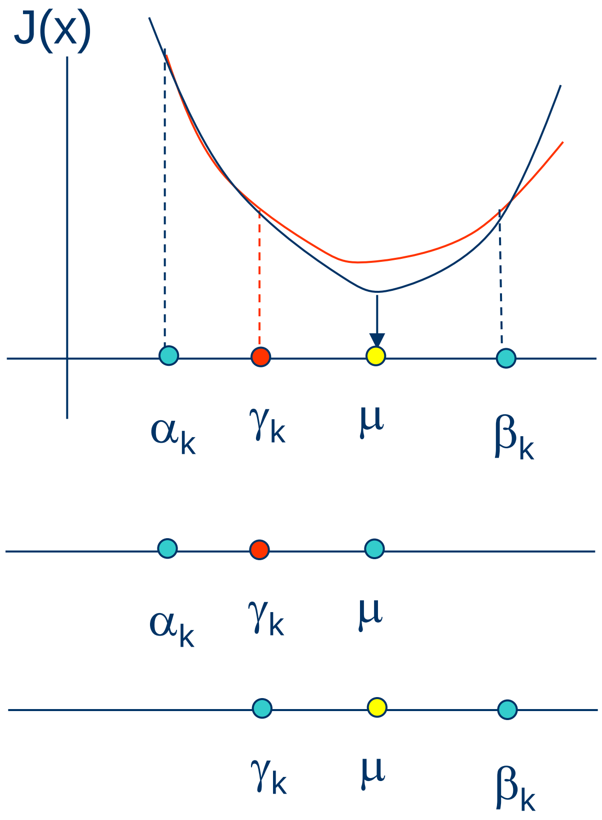
$$b = \frac{J(\gamma_k) - J(\alpha_k)}{\gamma_k - \alpha_k} - c(\gamma_k + \alpha_k)$$



Mínimum of $P(x) = a + bx + cx^2$

$\mu = - b / (2c)$, compute $J(\mu)$ and narrow the interval

Polynomial approximation methods (2nd order)



Mínimum of P(x) = a + bx + cx2
$\mu = - b / (2c)$, compute $J(\mu)$ and narrow the interval

Any of the two possible intervals for the next step contains an interior point were J(x) is known and can be used in the new iteration.

It is possible to use interpolating polynomials of different orders. For instance, a cubic interpolation using four points.