Numerical Optimization

Instructor : Sung Chan Jun

Week #3 : September 13, 2023 (Wednesday Class)

Course Syllabus (Tentative)

Calendar	Description	Remarks
1st week	*Introduction of optimization*	
2nd week	*Univariate Optimization*	
3rd week	Univariate Optimization	
4th week	Unconstrained Optimization	
5th week	Unconstrained Optimization	
6th week	Constrained Optimization, No Class	Oct. 2 (Temporary National Holiday)
7th week	Constrained Optimization, No Class	Oct. 9 (National Holiday)
8th week	Constrained Optimization, Midterm	Oct. 18 (Midterm)

Announcements

- Teaching Assistant (TA)
  - Dr. Cheolki Im (AI Graduate School)
    - Post-doc at Biocomputing Lab
    - E-mail: chim@gm.gist.ac.kr
    - Phone: 2266 (internal)
    - Office: DASAN Bldg. Room 505

Recall – This Monday

- Univariate Optimization

  Minimize f(x) on $x \in R$

  - When f(x) is differentiable
    - Univariate optimization comes to finding root problem : $f'(x) = 0$.
  - When f(x) is not differentiable
    - How can we solve the optimization problem?
      - Consider methods using function evaluations only
  - Unimodality
    - f(x) is unimodal in [a, b] if there exists a unique $x^* \in [a, b]$ such that for any $x_1, x_2 \in [a, b]$ and $x_1 < x_2$,
      - If $x_2 < x^*$ then $f(x_1) > f(x_2)$. If $x_1 > x^*$ then $f(x_1) < f(x_2)$
    - If f is unimodal in the given interval, it exists a strong local minimum in it.

Recall – This Monday

- Univariate Optimization: Unimodality
  - When unimodal $f(x)$ is evaluated at two interior points x_1 and x_2 ($x_1 < x_2$) for given interval $[a, b]$, then
    - if $f(x_1) > f(x_2)$, then a minimum is in $[x_1, b]$
    - Otherwise (if $f(x_1) \leq f(x_2)$), a minimum is in $[a, x_2]$

"Elimination Step"

  - Let f be unimodal and $x^* \in [a, b]$ be minimum.
    - By elimination step, (letting $[a_0, b_0] = [a, b]$), we got the following bracket method:

$$[a_0, b_0] \supset [a_1, b_1] \supset [a_2, b_2] \supset [a_3, b_3] \supset \dots \supset [a_n, b_n] \supset \dots$$

sufficiently reduced

sufficiently reduced

sufficiently reduced

Whether or not this bracket method successfully works depends on how to choose interior points.

Recall – This Monday

- Univariate Optimization: Unimodality
  - Assume f(x) is unimodal. To efficiently reduce the interval of uncertainty by elimination step, we should choose two interior points every iteration as reasonably as possible.
  - How to find two interior points?
    - Two efficient ways to consider
      - Fibonacci search
      - Golden section search

Recall – This Monday

- Univariate Optimization : Fibonacci search

S1. Assume N function evaluations are possible.

S2. Generate Fibonacci numbers $\{F_0, F_1, F_2, \ldots, F_N\}$ such that $F_0 = F_1 = 1$, $F_k = F_{k-1} + F_{k-2}$.

S3. Choose two interior points x_1 and x_2 (let $L = b - a$) :

$$x_1 = a + F_{N-2}/F_N * L = a\, F_{N-1}/F_N + b\, F_{N-2}/F_N$$

$$x_2 = b - F_{N-2}/F_N * L = a\, F_{N-2}/F_N + b\, F_{N-1}/F_N$$

Internally dividing points of [a, b]
x_1 = ratio $F_{N-2} : F_{N-1}$
x_2 = ratio $F_{N-1} : F_{N-2}$

S4. Compute $f(x_1)$ & $f(x_2)$. A new reduced interval $[a_{new}, b_{new}]$ is generated by

elimination step.

S5. Set $N := N - 1$, $a := a_{new}$, $b := b_{new}$.

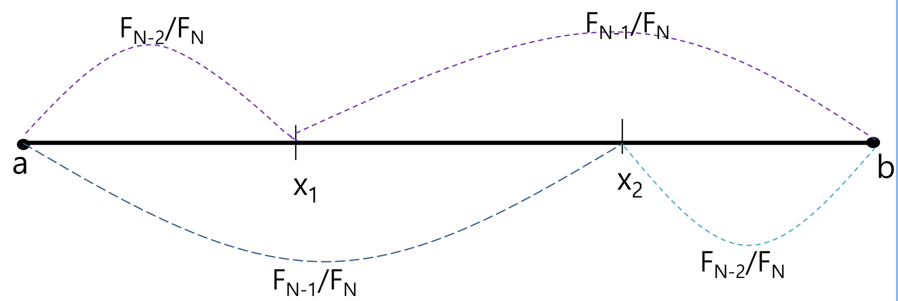
S6. Go to **S1** and repeat this until $N = 1$.

Recall – This Monday

- Univariate Optimization : Fibonacci search
  - Two interior points in Fibonacci search

    $x_1 = a + F_{N-2}/F_N * L = a\, F_{N-1}/F_N + b\, F_{N-2}/F_N$

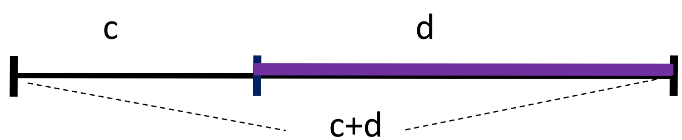
    $x_2 = b - F_{N-2}/F_N * L = a\, F_{N-2}/F_N + b\, F_{N-1}/F_N$



  - Due to Fibonacci sequences, every step requires just one more function evaluation except for the first step.
  - Final interval of uncertainty (N evaluations) : $1/F_N*(b-a)$
  - Cons
    - Require to store the Fibonacci numbers
    - Is not easy to apply for the case when termination criterion requires.

Recall – This Monday

- Univariate Optimization: Golden section search
  - Two interior points on [0, 1] are chosen as τ and $1-\tau$ such that $\tau > 1-\tau$.
  - Golden section ratio (τ)
    - $\tau = d/(c + d) = c/d$



$$\boxed{\tau^2 + \tau - 1 = 0 \qquad \tau = \frac{-1+\sqrt{5}}{2} \approx 0.6180}$$

  - Golden section search is a limiting case of Fibonacci search : $\lim_{k\to\infty} \frac{F_{k-1}}{F_k} = \tau$
  - It keeps good property of Fibonacci search
    - it requires just one additional function evaluation every step after 1st step.
  - Final interval of uncertainty (length of interval)
    - $\tau^{N-1}*(b-a)$, for N function evaluations
  - It is easy to answer how many function evaluations are needed to yield the given accuracy.

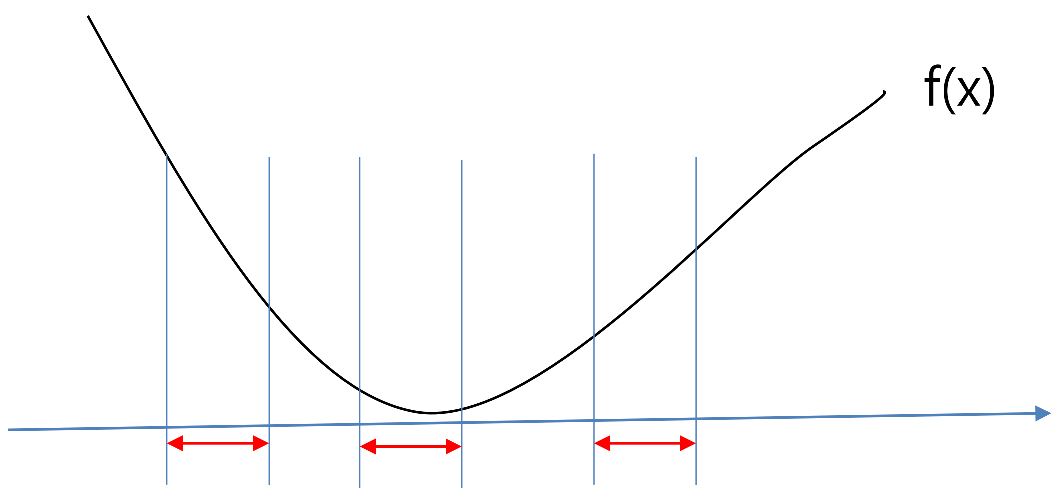
Recall – This Monday

- Univariate Optimization: Comparison of Search Algorithms

Fibonacci Search	Golden Section Search
▪ Use Fibonacci Sequences. ▪ Pros Every step requires one function evaluation only. ▪ Cons ✓ Require to store the Fibonacci numbers. ✓ not easy to apply for the case when termination criterion requires. ▪ Final length of interval $1/F_N$*(b–a) (after N function evaluations)	▪ Use Golden Section Ratio. ▪ Pros ✓ Every step requires one function evaluation only. ✓ Easily estimate how many iterations are needed to get the given accuracy. ▪ Final length of interval τ^{N-1}*(b–a) (after N function evaluations) ▪ This is a limiting case of Fibonacci search.

Univariate Optimization:
Seeking bound

- How to find initial interval [a, b] for a unimodal function f(x)?
    - If you choose randomly any interval, then there are three cases below:



Univariate Optimization:
Seeking bound

- How to find initial interval [a, b] for a unimodal function f(x)?
  - One of possible ideas

S1. Set randomly initial point x_0, step size $d_0 > 0$

S2. Evaluate $f_- := f(x_0 - d_0)$, $f_0 := f(x_0)$, $f_+ := f(x_0 + d_0)$

S3. If $f_- \geq f_0 \geq f_+$, then set $d := d_0$, $x_{-1} := x_0 - d_0$, $x_1 := x_0 + d_0$

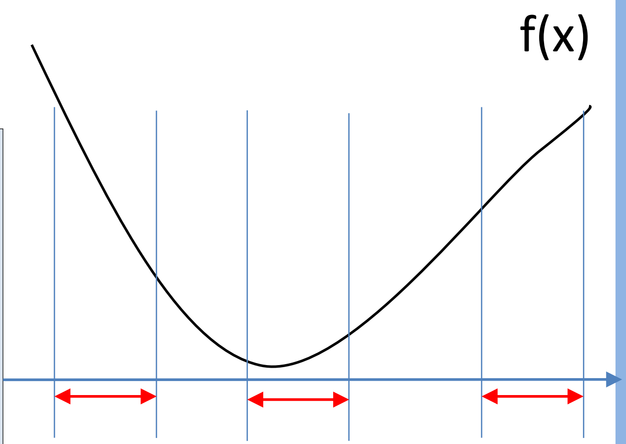
If $f_- \leq f_0 \leq f_+$, then set $d := -d_0$, $x_{-1} := x_0 + d_0$, $x_1 := x_0 - d_0$

If $f_- \geq f_0 \leq f_+$, then set $[a, b] := [x_0 - d_0, x_0 + d_0]$ and stop.

S4. For k = 1, 2, … $x_{k+1} = x_k + 2^k d$.

If $f(x_{k+1}) \geq f(x_k)$ & $d > 0$, then set $[a, b] := [x_{k-1}, x_{k+1}]$ and stop.

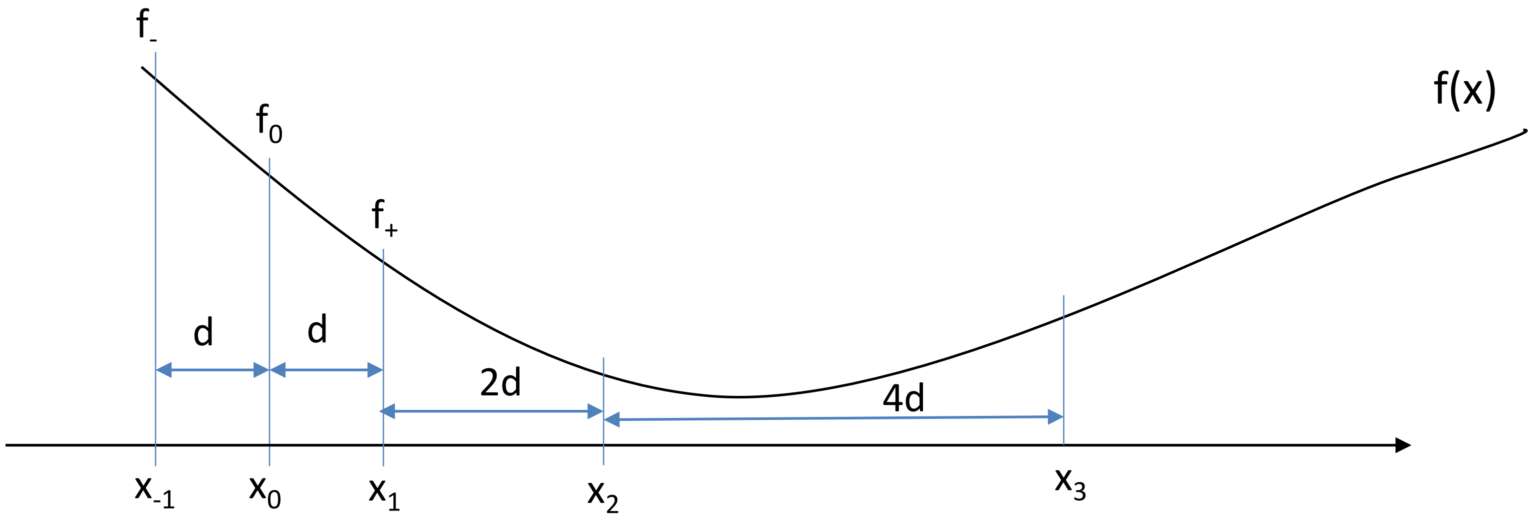
If $f(x_{k+1}) \geq f(x_k)$ & $d < 0$, then set $[a, b] := [x_{k+1}, x_{k-1}]$ and stop.



Univariate Optimization:
Seeking bound

S1. Set randomly initial point x_0, step size $d_0 > 0$

S2. Evaluate $f_-:=f(x_0-d_0)$, $f_0:=f(x_0)$, $f_+:=f(x_0+d_0)$



S3. If $f_- \geq f_0 \geq f_+$, then set $d:=d_0,\ x_{-1}:=x_0-d_0,\ x_1:=x_0+d_0$

S4. $k = 1$, $x_2 = x_1 + 2^1 d$. However, $f(x_2) < f(x_1)$ & $d > 0$. Move to $k = 2$

$k = 2$, $x_3 = x_2 + 2^2 d$. Then, $f(x_3) \geq f(x_2)$ & $d > 0$. Stop. $[x_1, x_3]$ is the desired interval.

Univariate Optimization:
Seeking bound

- Seeking bound

S1. Set randomly initial point x_0, step size $d_0 > 0$

S2. Evaluate $f_- := f(x_0 - d_0)$, $f_0 := f(x_0)$, $f_+ := f(x_0 + d_0)$

S3. If $f_- \geq f_0 \geq f_+$, then set $d := d_0,\ x_{-1} := x_0 - d_0,\ x_1 := x_0 + d_0$

If $f_- \leq f_0 \leq f_+$, then set $d := -d_0,\ x_{-1} := x_0 + d_0,\ x_1 := x_0 - d_0$

If $f_- \geq f_0 \leq f_+$, then set $[a, b] := [x_0 - d_0,\ x_0 + d_0]$ and stop.

S4. For k = 1, 2, … $x_{k+1} = x_k + 2^k d$.

- If $f(x_{k+1}) \geq f(x_k)$ & $d > 0$, then set $[a, b] := [x_{k-1}, x_{k+1}]$ and stop.
- If $f(x_{k+1}) \geq f(x_k)$ & $d < 0$, then set $[a, b] := [x_{k+1}, x_{k-1}]$ and stop.

Possible Ideas

- Any strictly increasing functions on k are acceptable.

Univariate Optimization

Minimize f(x) on $x \in R$

- When f(x) is not differentiable
  - Consider methods using function evaluations only
    - Fibonacci Search, Golden Section Search
  - What other methods?
- When f(x) is differentiable
  - Univariate optimization comes to finding root problem : $f'(x) = 0$.
    - Method of Bisection, Newton's, Secant, Regular falsi
  - What other methods?

Univariate Optimization:
Interpolation methods

- Assume f(x) is unimodal and twice continuously differentiable on [a, b].
  - Newton's method
    - Let f be twice continuously differentiable.
    - f ≈ quadratic interpolation function f^
    - By Taylor's expansion, with $f(x_k)$, $f'(x_k)$ and $f''(x_k)$

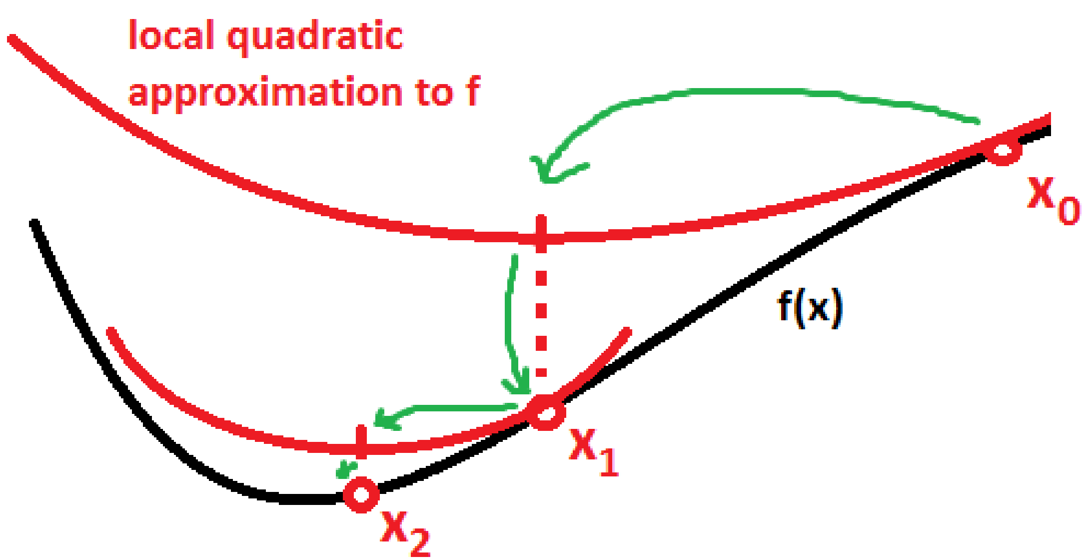
$$f^{\wedge}(x) = f(x_k) + f'(x_k)(x - x_k) + \frac{1}{2} f''(x_k)(x - x_k)^2$$

    - Find its minimum and call it x_{k+1}, then

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

Univariate Optimization:
Interpolation methods

- Newton's Method in Optimization



1-dimensional problem

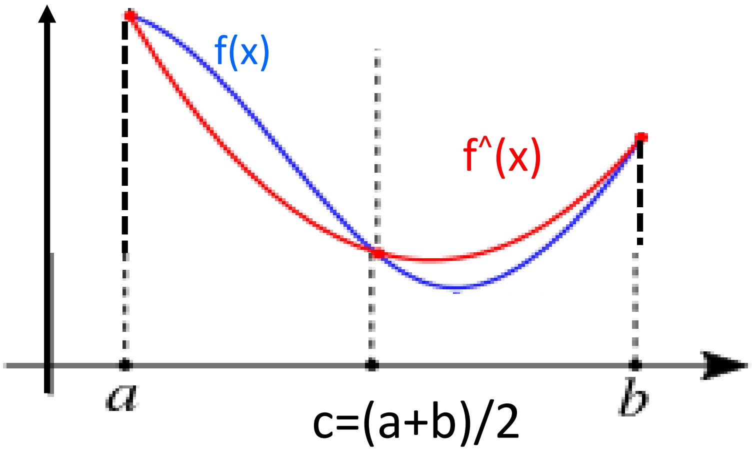
Univariate Optimization:
Interpolation Methods

- Assume $f(x)$ is unimodal and continuous on $[a, b]$.
  - Quadratic Interpolation without derivatives
    - Set interval to $[a, b]$ and midpoint $c := (a + b)/2$.
    - Evaluate f at three points : $(a, f(a))$, $(b, f(b))$, $(c, f(c))$.
    - $f \approx$ quadratic function passing through three points, find its minimum x.
    - Update the interval and do the same way again.

Univariate Optimization:
Interpolation Methods

- Lagrange polynomial interpolation
  - Polynomial passing three points (a, f(a)), (b, f(b)), and (c, f(c))

$$f\hat{}(x) = f(a)\frac{(x - c)(x - b)}{(a - c)(a - b)} + f(c)\frac{(x - a)(x - b)}{(c - a)(c - b)} + f(b)\frac{(x - a)(x - c)}{(b - c)(b - c)}$$



  - Minimum point of f^(x)

$$x = \frac{f(a)(b^2 - c^2) + f(b)(c^2 - a^2) + f(c)(a^2 - b^2)}{2[f(a)(b - c) + f(b)(c - a) + f(c)(a - b)]}$$

Univariate Optimization:
Interpolation Methods

- Assume f(x) is continuously differentiable and unimodal on [a, b]
  - Cubic interpolation with first derivatives
    - The minimum is in [a, b] such that $f'(a)f'(b)<0$.
    - Define $f^{\wedge}(x)$ cubic interpolation with following conditions:
      - (a, f(a)), (b, f(b))
      - (a, f'(a)), (b, f'(b))
    - Find a minimum of $f^{\wedge}(x)$.
    - Update the interval accordingly and do the same way again.

Univariate Optimization:
Safeguarded methods

- Assume f(x) is unimodal on [a, b]
  - Mixed method (reliable + rapid)
    - Reliable and guaranteed method
      - Fibonacci search
      - Golden Section search
    - Rapidly convergent method
      - Quadratic interpolation, and etc.

Multivariate Optimization

- Multivariate Optimization

$$\boxed{\text{Minimize } f(\mathbf{x}) \text{ on } \mathbf{x} = (x_1, x_2, \ldots, x_n)^T \in R^n}$$

  - Direct methods not using derivatives
    - Methods using function evaluations only
  - Derivative-based methods
    - Gradient-based methods
    - Second derivate methods

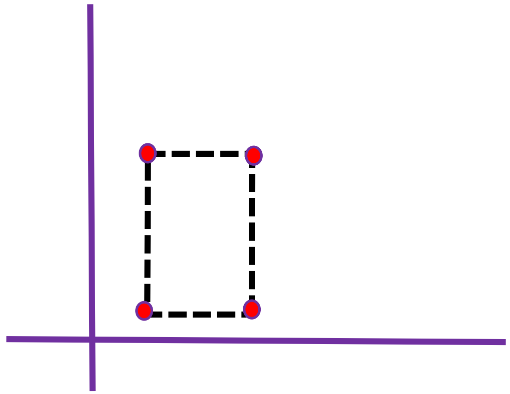
Multivariate Optimization:
Methods for Non-smooth functions

- Methods are based on function value comparisons.
  - How about considering the extension of search algorithms to multivariate optimization problem?
    - For two or higher dimensional problems, using elimination step may be less efficient due to the curse of dimensionality.
    - Caution! : This method would be used only when alternative method is not available.
  - In two dimensional problem, how about considering triangular shape in place of rectangular shape induced by interior points?

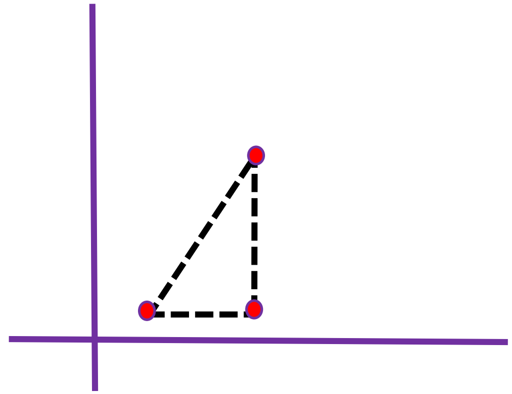
Multivariate Optimization:
Methods for Non-smooth functions

- Could you consider the elimination step in 2-dimensional space?

1-dimensional case



2-dimensional cases

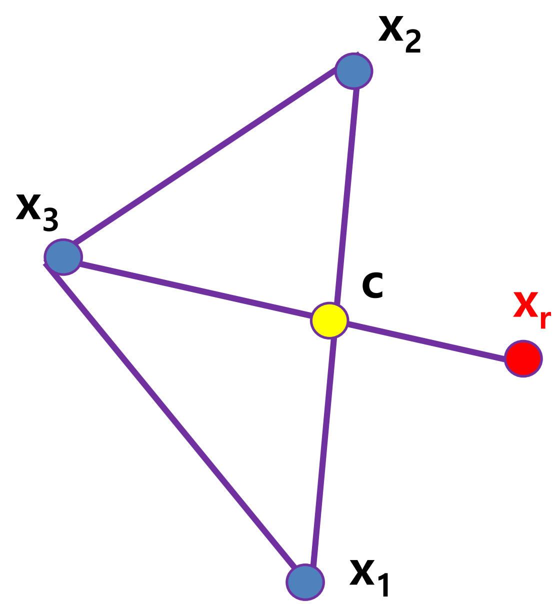


Multivariate Optimization:
Methods for Non-smooth functions

- Nelder and Mead Method (Downhill Simplex Method)
  - Consider a polygon (simplex) with N+1 vertices in N-dimensional space such as $\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_{N+1}$ and their corresponding function values $f_1, f_2, \ldots, f_{N+1}$.
  - Main Idea
    - Remove the vertex with the worst function value.
    - Replace it with a better value by reflecting, expanding, or contracting the polygon along the line joining the worst vertex with the centroid of the remaining vertices.
      - Basic three steps : reflection, expansion and contraction

Multivariate Optimization:
Methods for Non-smooth functions

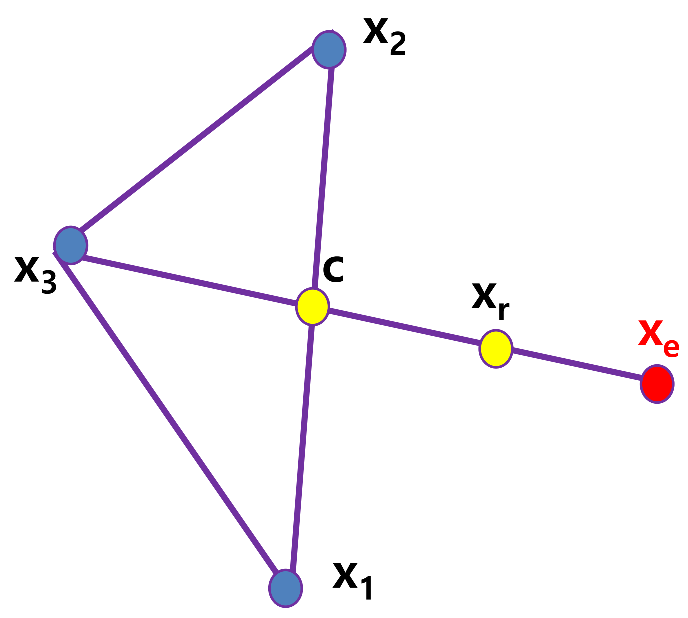
- Nelder and Mead Method
  - Reflection (Step 1)
    - Sort function values like $f_1 \le f_2 \le , \dots , \le f_{N+1}$ and corresponding vertices $\mathbf{x}_1, \mathbf{x}_2, \dots , \mathbf{x}_{N+1}$
    - Set $\mathbf{x}_r := \mathbf{c} + \alpha(\mathbf{c} - \mathbf{x}_{N+1}), \alpha > 0,\ \mathbf{c} = (\Sigma^N_{i=1} \mathbf{x}_i)/N$.
    - Evaluate f_r at $\mathbf{x}_r$.
    - If $f_1 \le f_r \le f_N$, replace $\mathbf{x}_{N+1}$ into $\mathbf{x}_r$ and sort function values $f_1 \le f_2 \le \dots \le f_{N+1}$, repeat this step.
    - If $f_r \ge f_N$, then go to Step 3 (contraction).
    - If $f_r \le f_1$, then go to Step 2 (expansion).



Multivariate Optimization:
Methods for Non-smooth functions

- Nelder and Mead Method
  - Expansion (Step 2)
    - Set $\mathbf{x}_e := \mathbf{c} + \beta(\mathbf{x}_r - \mathbf{c})$, $\beta > 1$
    - Evaluate f_e at $\mathbf{x}_e$.
    - If $f_e \leq f_r$, then $\mathbf{x}_e$ replaces $\mathbf{x}_{N+1}$.

      Otherwise $\mathbf{x}_r$ replaces $\mathbf{x}_{N+1}$.
    - Go to Step 1 (reflection)

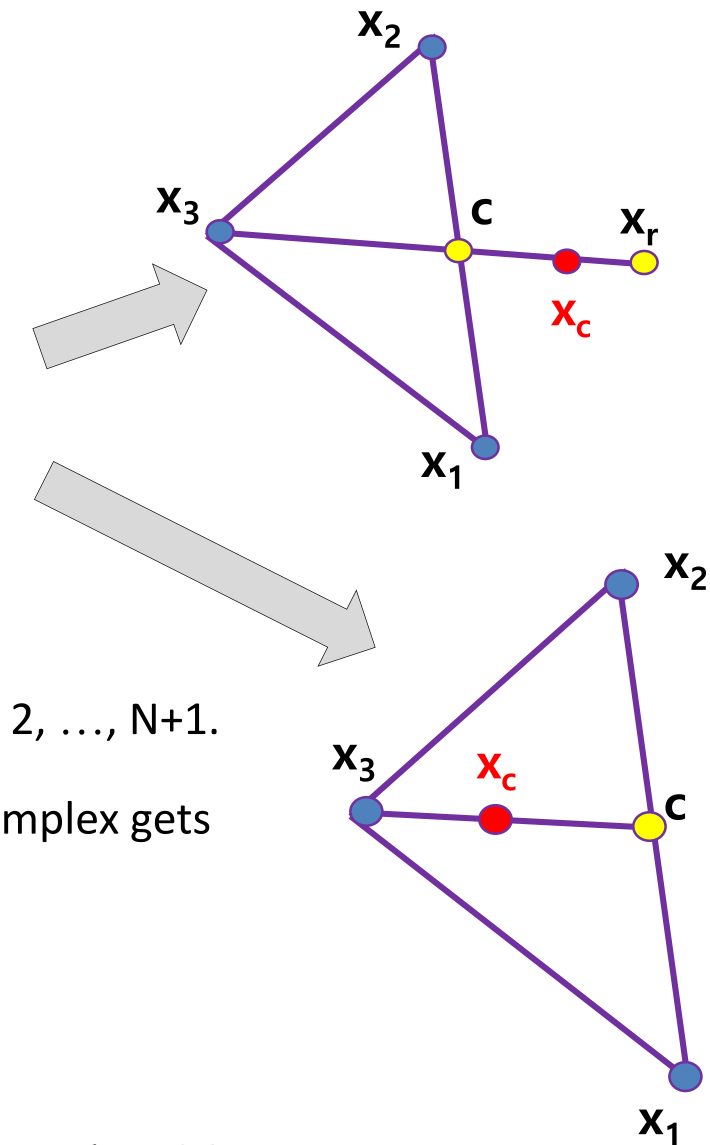


Multivariate Optimization: Methods for Non-smooth functions

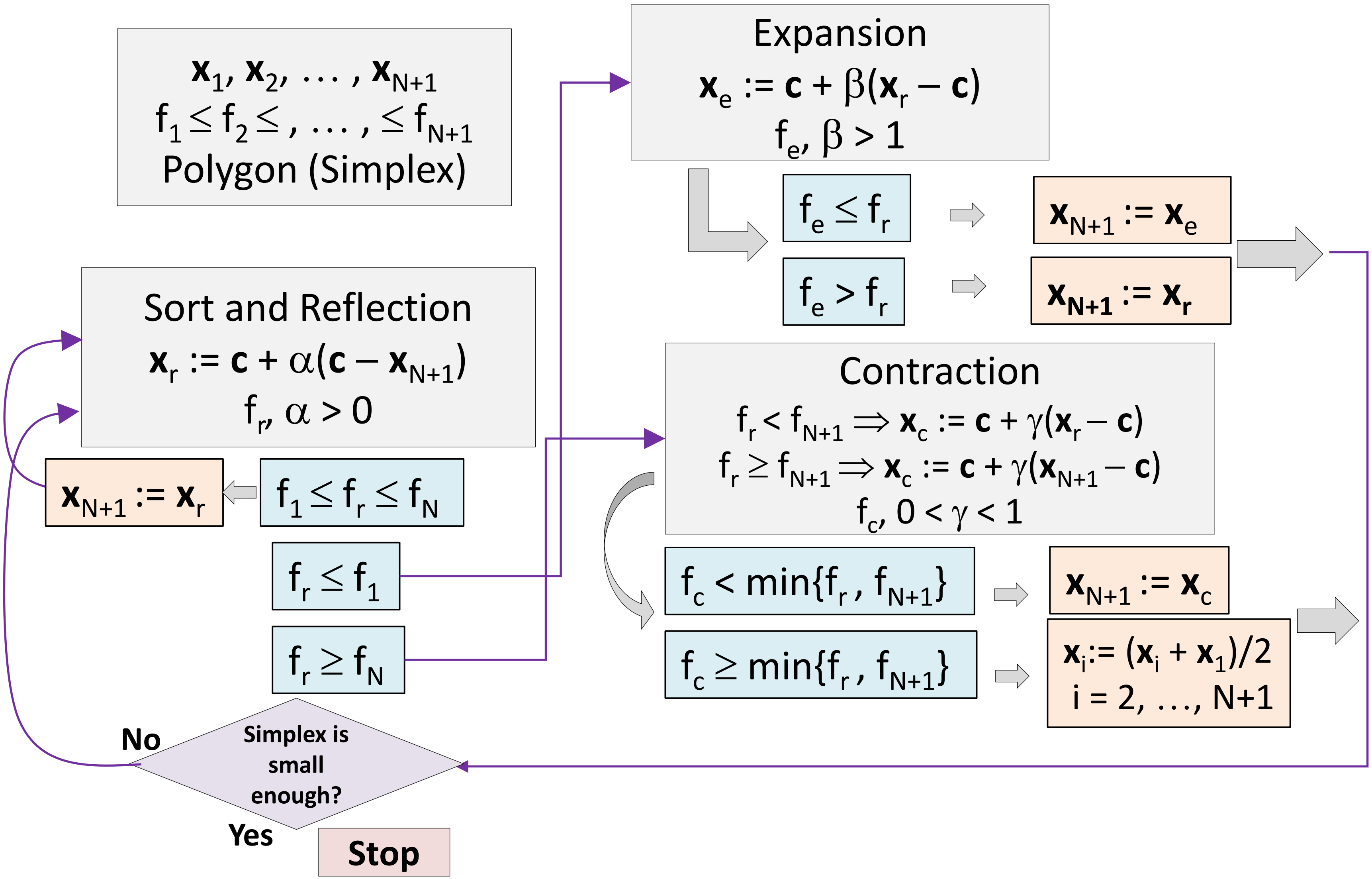
- Nelder and Mead Method
  - Contraction (Step 3, $0 < \gamma < 1$)
    - If $f_r < f_{N+1}$, then set $\mathbf{x}_c := \mathbf{c} + \gamma(\mathbf{x}_r - \mathbf{c})$.

      If $f_r \geq f_{N+1}$, then set $\mathbf{x}_c := \mathbf{c} + \gamma(\mathbf{x}_{N+1} - \mathbf{c})$.
    - Evaluate f_c at $\mathbf{x}_c$.
    - If $f_c < \min\{f_r, f_{N+1}\}$, then $\mathbf{x}_c$ replaces $\mathbf{x}_{N+1}$.

      If $f_c \geq \min\{f_r, f_{N+1}\}$, then $\mathbf{x}_i := (\mathbf{x}_i + \mathbf{x}_1)/2$, $i = 2, \ldots, N+1$.
    - Go to Step 1 (reflection) until the size of simplex gets below desired limit.



Multivariate Optimization:
Nelder and Mead Method

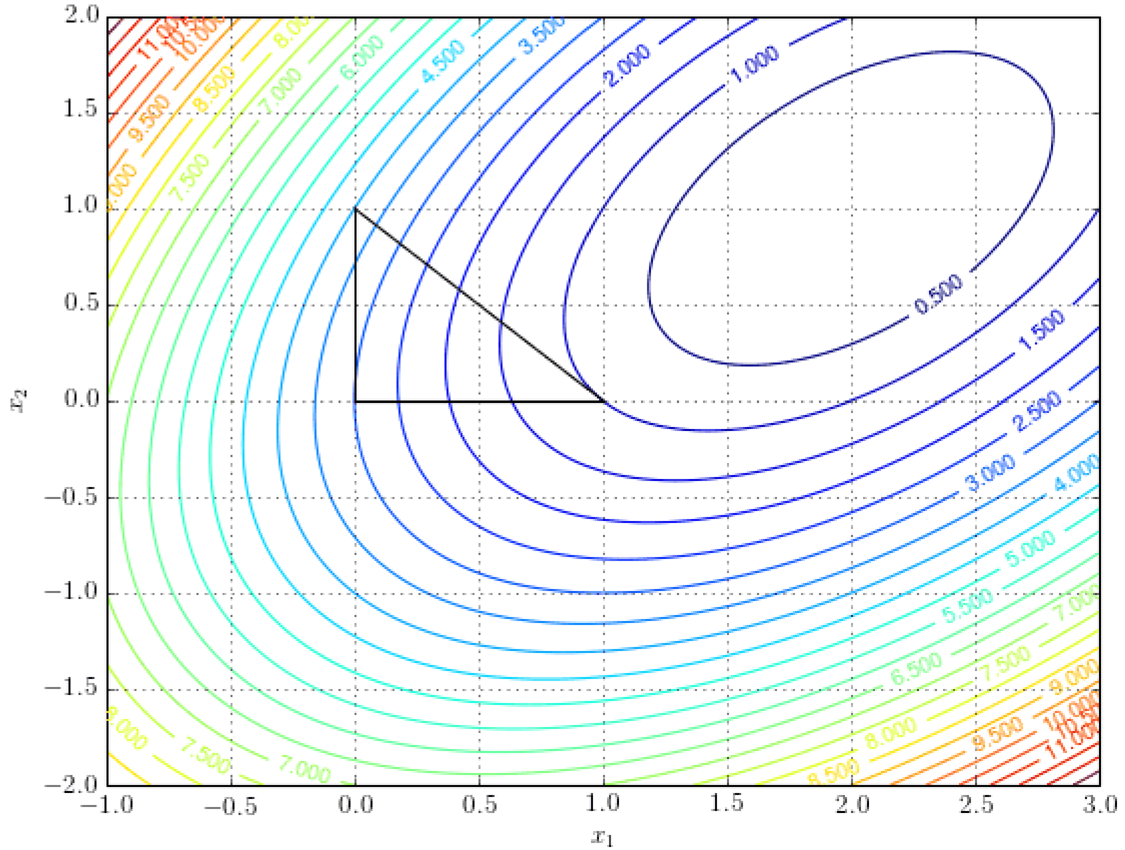


Multivariate Optimization:
Nelder and Mead Method

- Pros
  - Easy to implement.
  - Small memory to store.
- Cons
  - Convergence is slow.
  - Restart with a new polygon when the stagnation is detected.
  - Determine three control parameters reasonably

    for example, $\alpha = 1$, $\beta = 2$, $\gamma = 0.5$

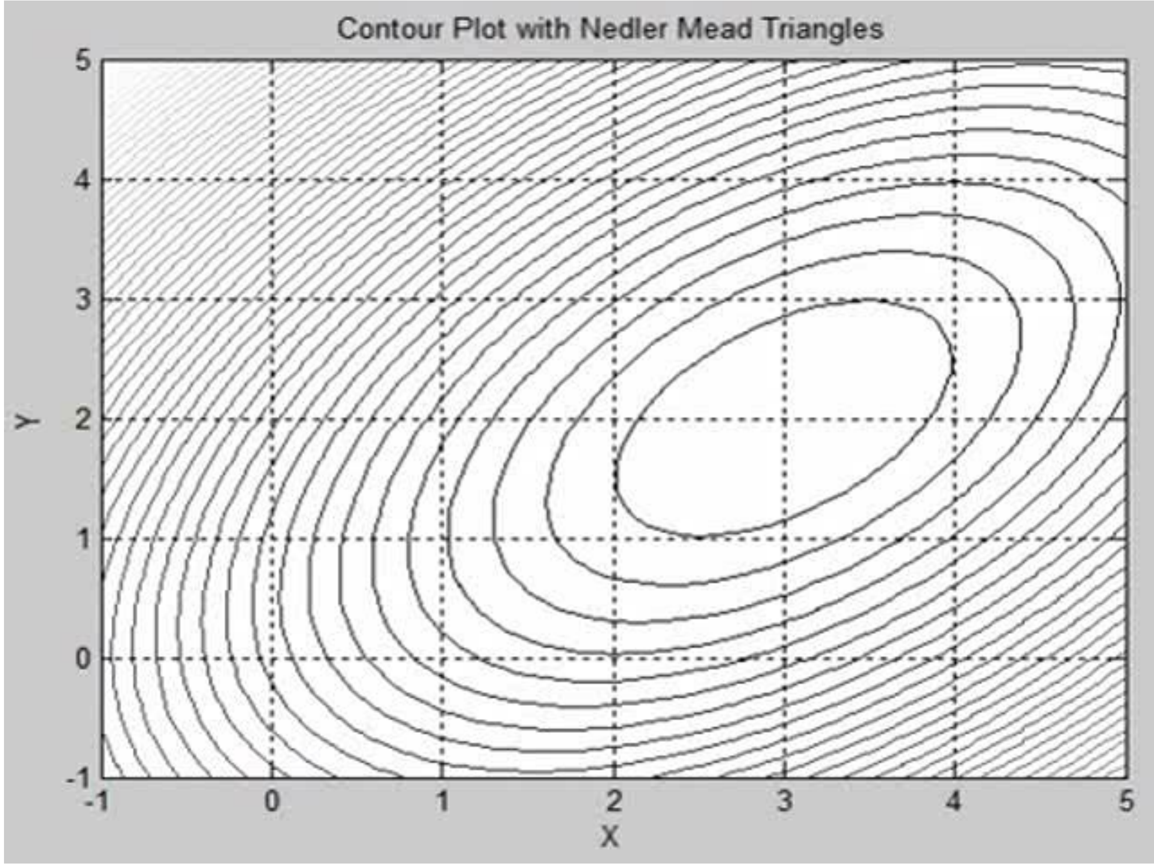
Nelder and Mead Method
(Downhill Simplex Method)



https://youtu.be/HUqLxHfxWqU

Nelder and Mead Method
(Downhill Simplex Method)

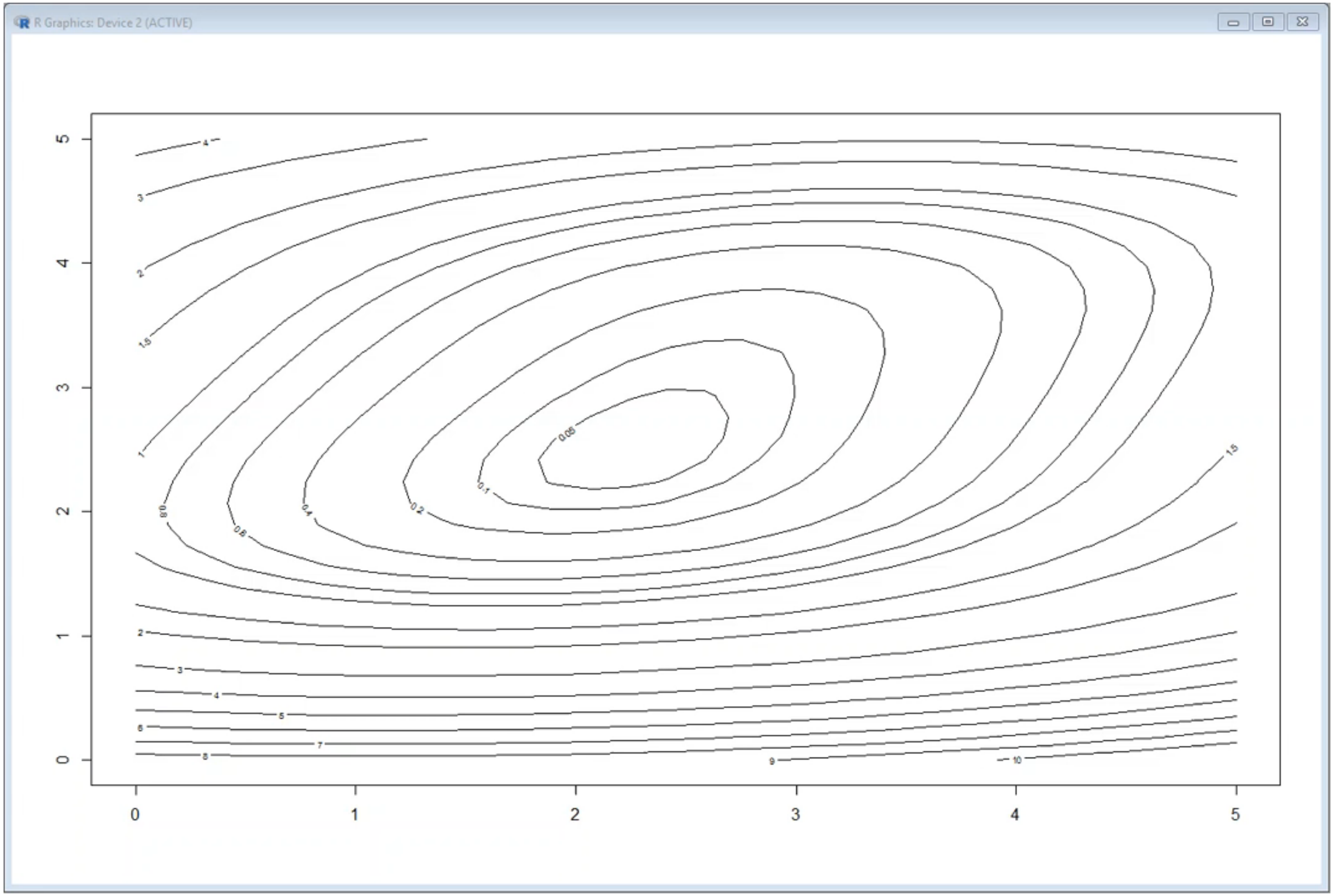
$$f(x, y) = x^2 - 4x + y^2 - y - xy$$



https://youtu.be/HUqLxHfxWqU

Nelder and Mead Method
(Downhill Simplex Method)

$$f(x, y) = ((x - y)^2 + (x - 2)^2 + (y - 3)^4) / 10$$



https://www.youtube.com/watch?v=j2gcuRVbwR0