

Metaheuristic Optimization: Nelder-Mead Simplex Algorithm

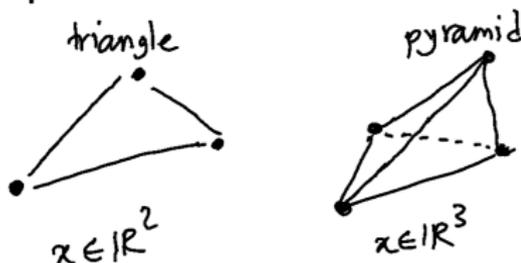
Adaptive and Cooperative Algorithms (ECE 457A)

ECE, MME, and MSCI Departments,
University of Waterloo, ON, Canada

Course Instructor: Benjamin Ghojogh
Fall 2023

Nelder-Mead Simplex Algorithm

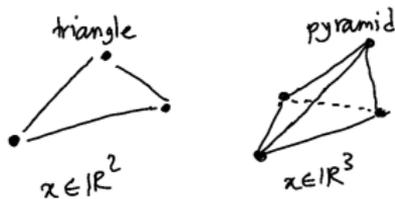
- The **Nelder-Mead simplex algorithm**, also called the **Nelder-Mead method**, was proposed by John A. Nelder and Roger Mead in 1965 [1].
- Used in “**fminsearch**” function of **MATLAB**:
<https://www.mathworks.com/help/matlab/ref/fminsearch.html>
- Its idea:
 - ▶ If the dimensionality of optimization variable is d , choose $d + 1$ random points in the feasible set to make a **simplex**.



- ▶ Update this simplex iteratively until it converges to the optimal solution (it gradually moves toward the solution and shrinks to the solution.)

Nelder-Mead Method: initial simplex

- If the dimensionality of optimization variable is d , choose $d + 1$ random points in the feasible set to make a **simplex**.



- This initial simplex is important. A too small simplex may get stuck in a local optimum (cannot do enough exploration).
- It is suggested in [1] to select the initial simplex as the following:
 - ▶ a random point for $\mathbf{x}_1 = [x_{11}, x_{12}, \dots, x_{1d}]^T$
 - ▶ each of $\{\mathbf{x}_2, \dots, \mathbf{x}_{d+1}\}$ is a fixed step along each dimension in turn:

$$\mathbf{x}_2 = [x_{11} + \delta, x_{12}, \dots, x_{1d}]^T,$$

$$\mathbf{x}_3 = [x_{11}, x_{12} + \delta, \dots, x_{1d}]^T,$$

\vdots

$$\mathbf{x}_{d+1} = [x_{11}, x_{12}, \dots, x_{1d} + \delta]^T,$$

where $\delta > 0$ is a not-too-small number.

Nelder-Mead Method: order

- We want to **minimize** the function $f(\cdot)$. In the feasibility set, we make some simplex and change it in the feasibility set iteratively to converge to the solution.
- **order**: at the start of every iteration, order (sort) the corners of simplex:

$$f(\mathbf{x}_1) \leq f(\mathbf{x}_2) \leq \dots \leq f(\mathbf{x}_{d+1}),$$

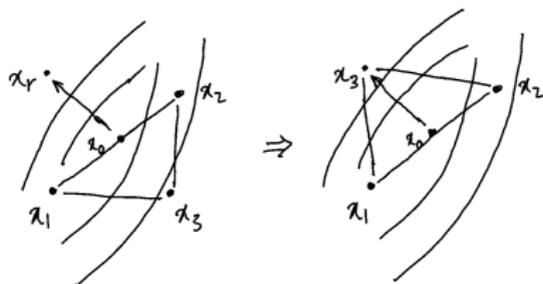
where we denote the best and worst corners by \mathbf{x}_1 and \mathbf{x}_{d+1} , respectively.

Nelder-Mead Method: order and reflection

- **centroid**: the centroid of all points except \mathbf{x}_{d+1} as: $\mathbf{x}_o = \frac{1}{d} \sum_{i=1}^d \mathbf{x}_i$.
- **reflection**:
 - ▶ the reflected point:

$$\mathbf{x}_r = \mathbf{x}_o + \alpha(\mathbf{x}_o - \mathbf{x}_{d+1}).$$

- ▶ $\alpha > 0$, usually $\alpha = 1$.
- ▶ if $f(\mathbf{x}_1) \leq f(\mathbf{x}_r) < f(\mathbf{x}_d)$:
 - ★ replace the worst point with the reflected point, $\mathbf{x}_{d+1} := \mathbf{x}_r$.
 - ★ go to the next iteration and order the points again.



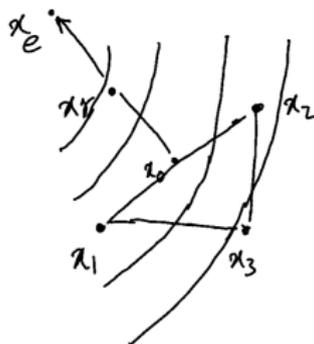
- It has some connection with **opposition learning** proposed in 2005 [2] and used in metaheuristic optimization in 2008 [3].

Nelder-Mead Method: expansion

- **expansion:** if $f(\mathbf{x}_r) \leq f(\mathbf{x}_1)$:
 - ▶ the expanded point:

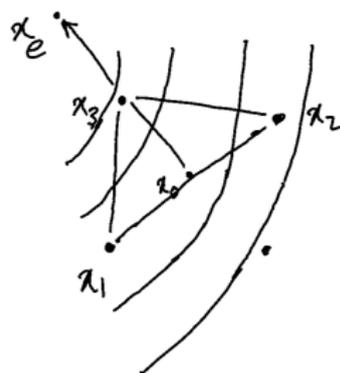
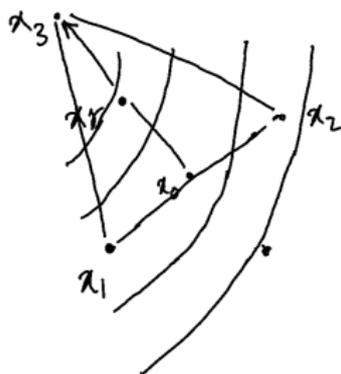
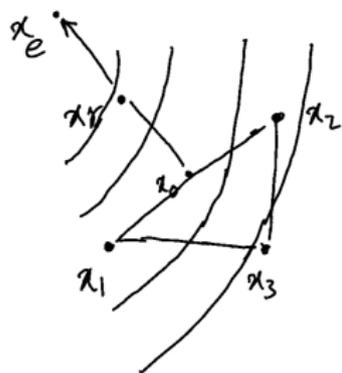
$$\mathbf{x}_e = \mathbf{x}_o + \gamma(\mathbf{x}_r - \mathbf{x}_o).$$

- ▶ $\gamma > 1$, usually $\gamma = 2$.



Nelder-Mead Method: expansion

- if the expanded point is better than the reflected point, $f(\mathbf{x}_e) < f(\mathbf{x}_r)$:
 - ▶ replace the worst point with the expanded point, $\mathbf{x}_{d+1} := \mathbf{x}_e$.
- else, $f(\mathbf{x}_e) \geq f(\mathbf{x}_r)$:
 - ▶ replace the worst point with the reflected point, $\mathbf{x}_{d+1} := \mathbf{x}_r$.
- go to the next iteration and order the points again.



Nelder-Mead Method: contraction & shrinking

● **contraction:** if $f(\mathbf{x}_r) \geq f(\mathbf{x}_d)$:

▶ if $f(\mathbf{x}_r) < f(\mathbf{x}_{d+1})$:

★ the contracted point **outside**:

$$\mathbf{x}_c = \mathbf{x}_o + \rho(\mathbf{x}_r - \mathbf{x}_o).$$

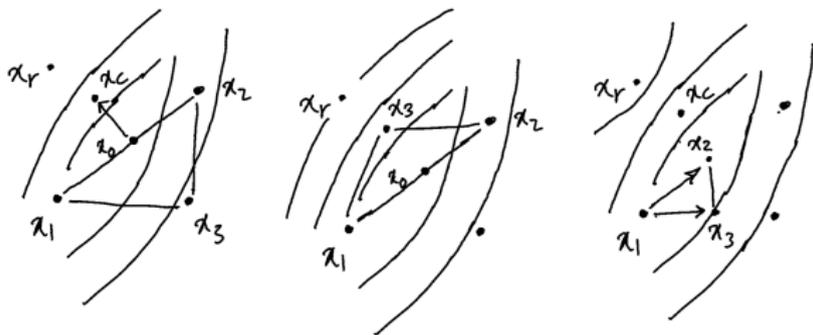
★ $0 < \rho \leq 0.5$, usually $\rho = 0.5$.

★ if the contracted point is better than the reflected point, $f(\mathbf{x}_c) < f(\mathbf{x}_r)$:
replace the worst point with the contracted point, $\mathbf{x}_{d+1} := \mathbf{x}_c$.

★ else, $f(\mathbf{x}_c) \geq f(\mathbf{x}_r)$:

shrinking: replace all points (except the best point \mathbf{x}_1) with the reflected point, $\mathbf{x}_i := \mathbf{x}_1 + \sigma(\mathbf{x}_i - \mathbf{x}_1)$, where $\sigma = 0.5$.

★ go to the next iteration and order the points again.

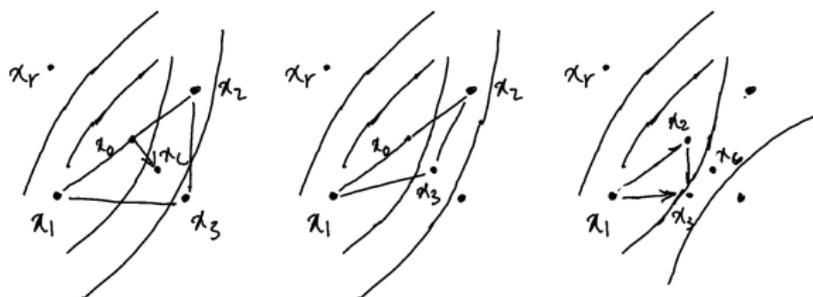


Nelder-Mead Method: contraction & shrinking

- **contraction:** if $f(\mathbf{x}_r) \geq f(\mathbf{x}_d)$:
 - ▶ if $f(\mathbf{x}_r) \geq f(\mathbf{x}_{d+1})$:
 - ★ the contracted point **inside**:

$$\mathbf{x}_c = \mathbf{x}_o + \rho(\mathbf{x}_{d+1} - \mathbf{x}_o).$$

- ★ $0 < \rho \leq 0.5$, usually $\rho = 0.5$.
- ★ if the contracted point is better than the worst point, $f(\mathbf{x}_c) < f(\mathbf{x}_{d+1})$:
replace the worst point with the contracted point, $\mathbf{x}_{d+1} := \mathbf{x}_c$.
- ★ else, $f(\mathbf{x}_c) \geq f(\mathbf{x}_{d+1})$:
shrinking: reflect all points (except the best point \mathbf{x}_1) with the reflected point, $\mathbf{x}_i := \mathbf{x}_1 + \sigma(\mathbf{x}_i - \mathbf{x}_1)$, where $\sigma = 0.5$.
- ★ go to the next iteration and order the points again.



Nelder-Mead Method: summary

initialize simplex with $(d+1)$ points in feasibility set

loop until convergence:

calculate centroid: x_0

calculate reflected point: x_r

if $f(x_1) \leq f(x_r) \leq f(x_d)$:

| $x_{d+1} \leftarrow x_r$
| continue

elseif $f(x_r) \leq f(x_1)$:

| $x_e = x_0 + \gamma(x_r - x_0)$

| if $f(x_e) < f(x_r)$:

| | $x_{d+1} \leftarrow x_e$

| else:

| | $x_{d+1} \leftarrow x_r$

| continue

Nelder-Mead Method: summary

loop until convergence:

```

|
elseif  $f(x_r) \geq f(x_d)$ :
  if  $f(x_r) < f(x_{d+1})$ :
     $x_c = x_0 + \rho(x_r - x_0)$ 
    if  $f(x_c) < f(x_r)$ :
      |
       $x_{d+1} \leftarrow x_c$ 
    else:
       $x_i \leftarrow x_1 + \sigma(x_i - x_1) \quad \forall i$ 
    continue
  else:
     $x_c = x_0 + \rho(x_{d+1} - x_0)$ 
    if  $f(x_c) < f(x_{d+1})$ :
      |
       $x_{d+1} \leftarrow x_c$ 
    else:
       $x_i \leftarrow x_1 + \sigma(x_i - x_1) \quad \forall i$ 
    continue
```

Acknowledgment

- Another YouTube video of mine about this algorithm:
<https://www.youtube.com/watch?v=s-6rV1nMw0w>

References

- [1] J. A. Nelder and R. Mead, "A simplex method for function minimization," *The computer journal*, vol. 7, no. 4, pp. 308–313, 1965.
- [2] H. R. Tizhoosh, "Opposition-based learning: a new scheme for machine intelligence," in *International conference on computational intelligence for modelling, control and automation and international conference on intelligent agents, web technologies and internet commerce (CIMCA-IAWTIC'06)*, vol. 1, pp. 695–701, IEEE, 2005.
- [3] S. Rahnamayan, H. R. Tizhoosh, and M. M. Salama, "Opposition-based differential evolution," *IEEE Transactions on Evolutionary computation*, vol. 12, no. 1, pp. 64–79, 2008.