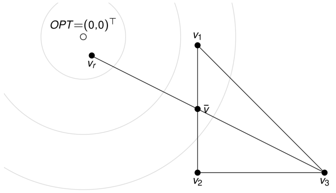
Optimization in Machine Learning

Nelder-Mead method



Learning goals

- General idea
- Reflection, expansion, contraction
- Advantages & disadvantages
- Examples

NELDER-MEAD METHOD

- Derivative-free method $\Rightarrow$ heuristic
- Generalization of bisection in d-dimensional space
- Based on d-simplex, defined by $d + 1$ points:
  - $d = 1$ interval
  - $d = 2$ triangle
  - $d = 3$ tetrahedron
  - $\cdots$

NELDER-MEAD METHOD

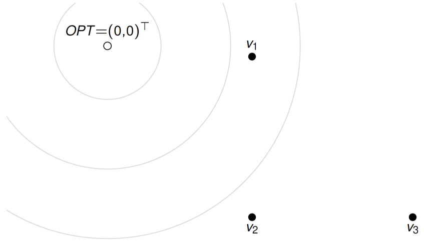
A version of the Nelder-Mead method:

Initialization: choose $d + 1$ random, affinely independent points $\mathbf{v}_i$
($\mathbf{v}_i$ are vertices: corner points of the simplex/polytope)

1. Order points according to ascending function values

$$f(\mathbf{v}_1) \leq f(\mathbf{v}_2) \leq \ldots \leq f(\mathbf{v}_d) \leq f(\mathbf{v}_{d+1})$$

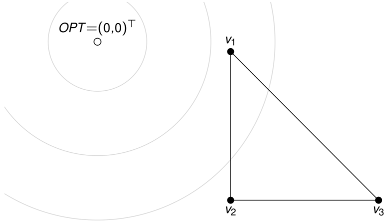
   with $\mathbf{v}_1$ best point, $\mathbf{v}_{d+1}$ worst point

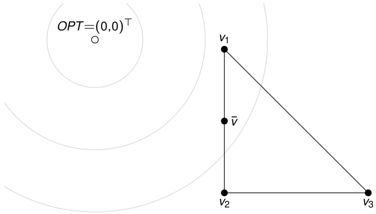


NELDER-MEAD METHOD

2. Compute centroid without worst point

$$\bar{\mathbf{v}} = \frac{1}{d}\sum_{i=1}^{d}\mathbf{v}_i.$$



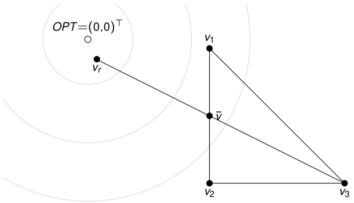


NELDER-MEAD METHOD

3. Reflection: compute reflection point

$$\mathbf{v}_r = \bar{\mathbf{v}} + \rho(\bar{\mathbf{v}} - \mathbf{v}_{d+1}),$$

with $\rho > 0$. Compute $f(\mathbf{v}_r)$.



Note: Default value for reflection coefficient: $\rho = 1$

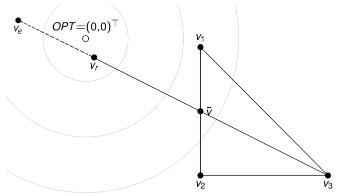
NELDER-MEAD METHOD

Distinguish three cases:

- Case 1: $f(\mathbf{v}_1) \leq f(\mathbf{v}_r) < f(\mathbf{v}_d) \Rightarrow$ Accept $\mathbf{v}_r$ and discard $\mathbf{v}_{d+1}$
- Case 2: $f(\mathbf{v}_r) < f(\mathbf{v}_1) \Rightarrow$
  Expansion:

  $$\mathbf{v}_e = \bar{\mathbf{v}} + \chi(\mathbf{v}_r - \bar{\mathbf{v}}), \quad \chi > 1$$

  We discard $\mathbf{v}_{d+1}$ and accept the better of $\mathbf{v}_r$ and $\mathbf{v}_e$



Note: default value for expansion coefficient: $\chi = 2$

NELDER-MEAD METHOD

- Case 3: $f(\mathbf{v}_r) \geq f(\mathbf{v}_d) \Rightarrow$ Contraction:

$$\mathbf{v}_c = \bar{\mathbf{v}} + \gamma(\mathbf{v}_{d+1} - \bar{\mathbf{v}})$$

  with $0 < \gamma \leq 1/2$
  - If $f(\mathbf{v}_c) < f(\mathbf{v}_{d+1})$, accept $\mathbf{v}_c$
  - Otherwise, shrink entire simplex (Shrinking):

$$\mathbf{v}_i = \mathbf{v}_1 + \sigma(\mathbf{v}_i - \mathbf{v}_1) \quad \forall i$$

  Note: default values for contraction and shrinking coefficients $\gamma = \sigma = 1/2$

4. Repeat all steps until stopping criterion met

NELDER-MEAD SUMMARY

Advantages:

- No gradients needed
- Robust, often works well for non-differentiable functions

Drawbacks:

- Relatively slow (not applicable in high dimensions)
- Not each step improves, only mean of corner values is reduced
- No guarantee for convergence to local optimum / stationary point

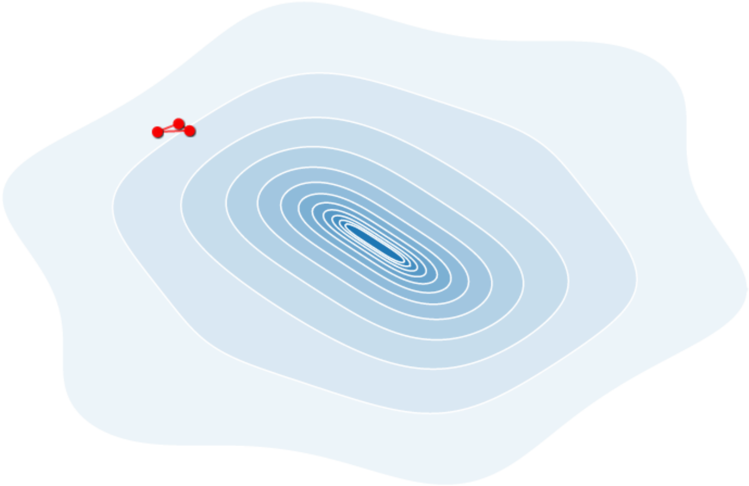
Visualization:

`http://www.benfrederickson.com/numerical-optimization/`

Note: Nelder-Mead is default method of `R` function `optim()`
If gradient is available and cheap, L-BFGS is preferred

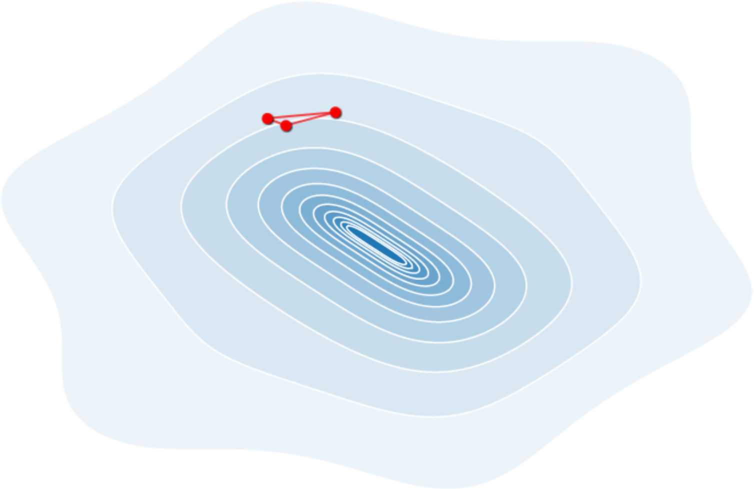
NELDER-MEAD VISUALIZATION IN 2D

$$\min_{\mathbf{x}} f(x_1, x_2) = x_1^2 + x_2^2 + x_1 \cdot \sin x_2 + x_2 \cdot \sin x_1$$



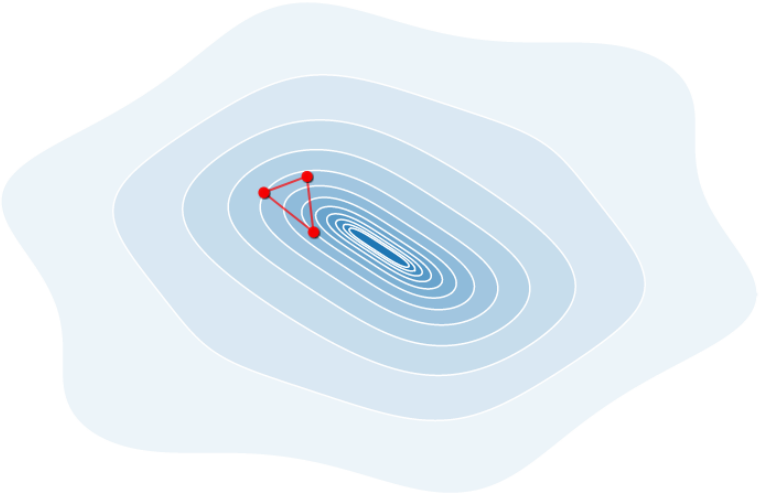
NELDER-MEAD VISUALIZATION IN 2D

$$\min_{\mathbf{x}} f(x_1, x_2) = x_1^2 + x_2^2 + x_1 \cdot \sin x_2 + x_2 \cdot \sin x_1$$

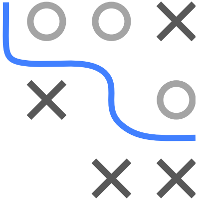


NELDER-MEAD VISUALIZATION IN 2D

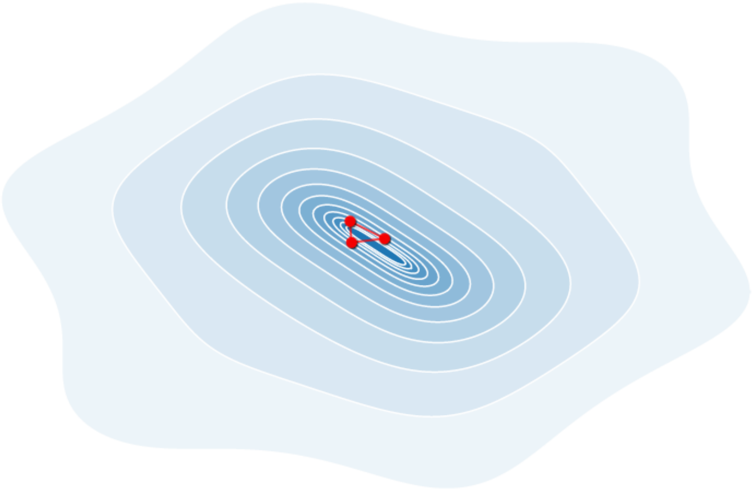
$$\min_{\mathbf{x}} f(x_1, x_2) = x_1^2 + x_2^2 + x_1 \cdot \sin x_2 + x_2 \cdot \sin x_1$$



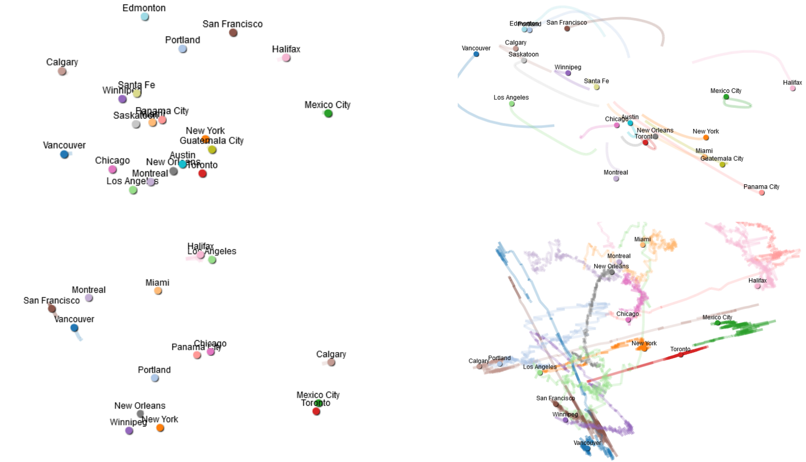
NELDER-MEAD VISUALIZATION IN 2D



$$\min_{\mathbf{x}} f(x_1, x_2) = x_1^2 + x_2^2 + x_1 \cdot \sin x_2 + x_2 \cdot \sin x_1$$

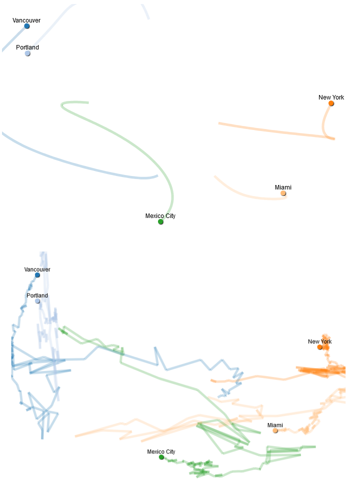


NELDER-MEAD VS. GD



Nelder-Mead in multiple dimensions: Organize points (US cities) to keep predefined mutual distances. For 10 cities, gradient descent (top) converges well for a suitable learning rate. Nelder-Mead (bottom) fails to converge, even after many iterations.

NELDER-MEAD VS. GD



Even for only 5 cities, Nelder-Mead (bottom) performs poorly

However, gradient descent (top) still works