Optimization Techniques

Programming for Data Science

Model Estimation

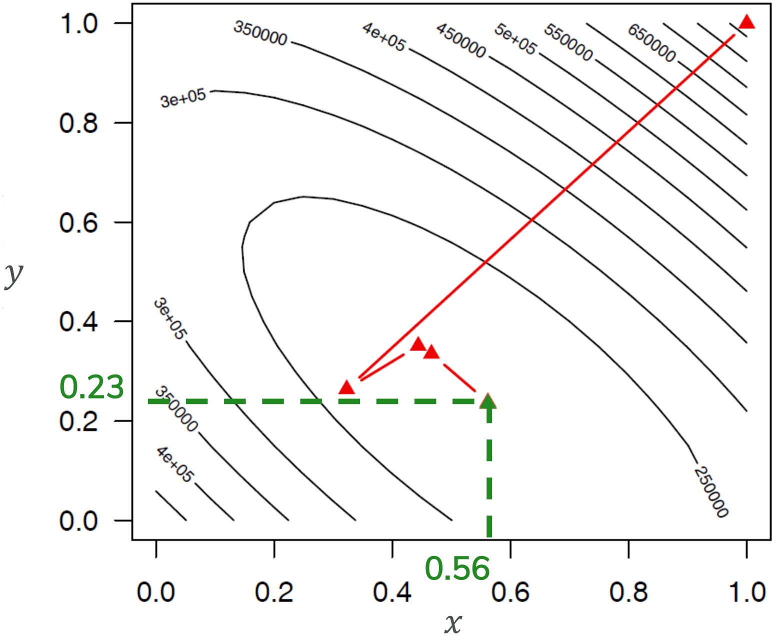
Dresden Database Systems Group

Problem

- Find the parameters that minimize the costs
- For all data points

Example

- Cost function:
  - $f(x,y) = x^2 + y^2$
- Optimization problem:
  - $\underset{x,y}{\operatorname{argmin}} f(x,y) = \underset{x,y}{\operatorname{argmin}} x^2 + y^2$
- Parameter Estimator
  - L-BFGS-B

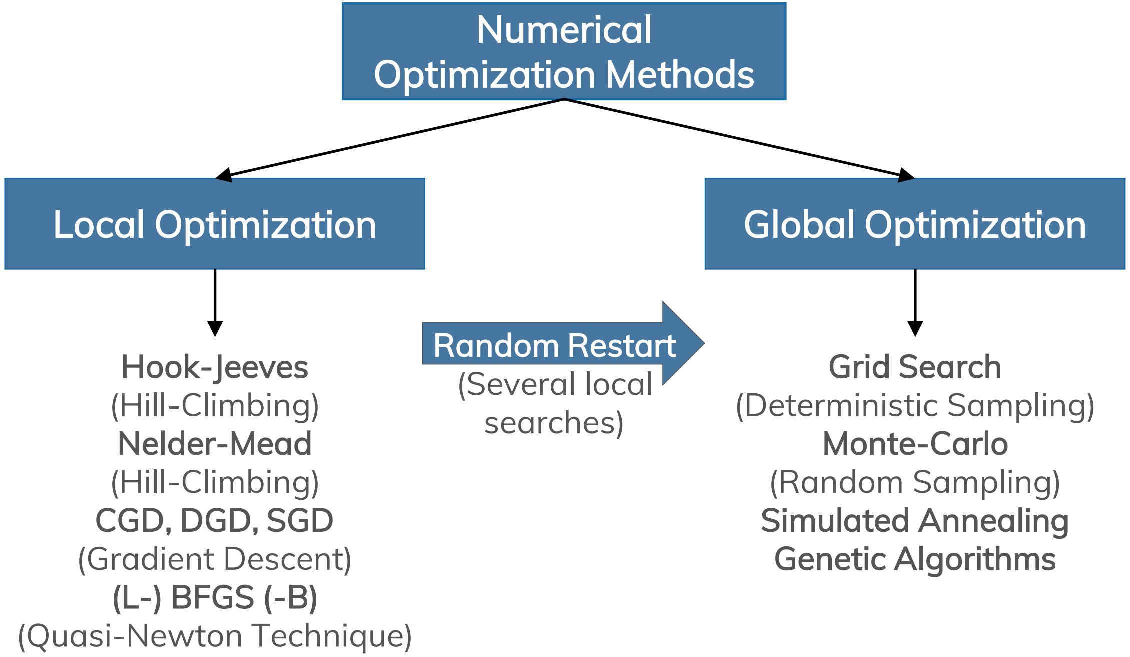


Optimization Methods

Dresden Database Systems Group

Overview on Optimization Methods (Not Complete!)

- Numerical general purpose optimization methods
- No analytical solutions (system of linear equations)
- No differentiable optimization target



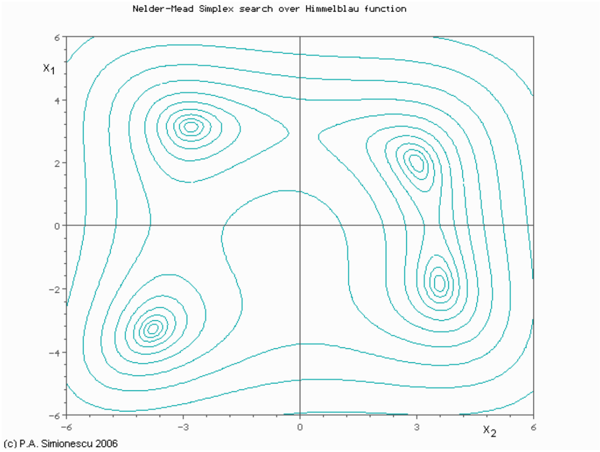
Nelder-Mead

Downhill-Simplex-Method

- Based on a simplex, most simple polygon with n+1 vertices in an n-dimensional space
  - One-dimensional: segment
  - Two-dimensional: triangle
  - Three-dimensional: rectangle
- Choose the worst point from the n+1 points
- In every Iteration the worst point is replaced by a new (hopefully better) point

Example

- Two dimensional parameter space with contour lines
- Triangle is approaching the optimum

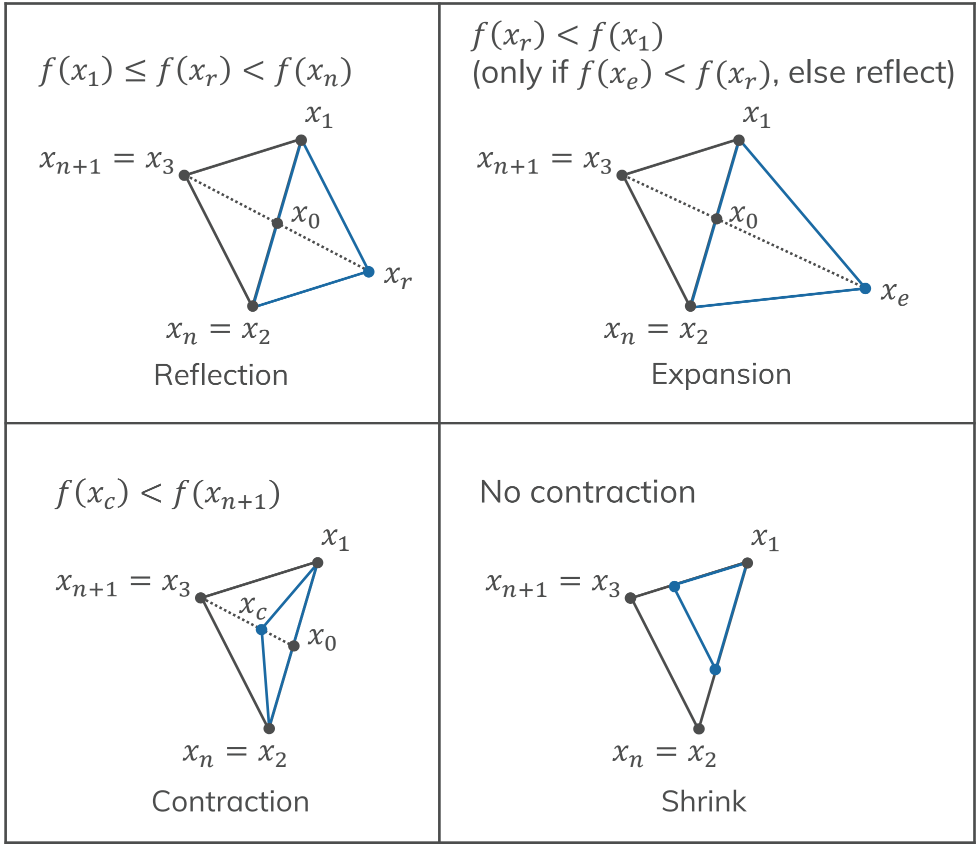


Nelder-Mead

4 strategies to calculate the new simplex

1. Sort the points in ascending order according to their costs f: $x_1, x_2, \ldots, x_n, x_{n+1}$
2. Calculate the centroid x_0 with x_1 to x_n

3. While $std(f(x_1), \ldots, f(x_{n+1})) > t$:
   According to the given criteria,
   1. **Reflect** the worst point at the centroid
      $x_r = \mathrm{x}_0 + \alpha(x_0 - x_{n+1})$
   2. **Reflect & Expand** the worst point at the centroid
      $x_e = \mathrm{x}_0 + \gamma(x_r - x_0)$
   3. **Contract** the worst point towards the centroid
      $x_c = \mathrm{x}_0 + \rho(x_{n+1} - x_0)$
   4. **Shrink** all points (except x_1) towards x_1
      $x_i = \mathrm{x}_1 + \sigma(x_i - x_1)$

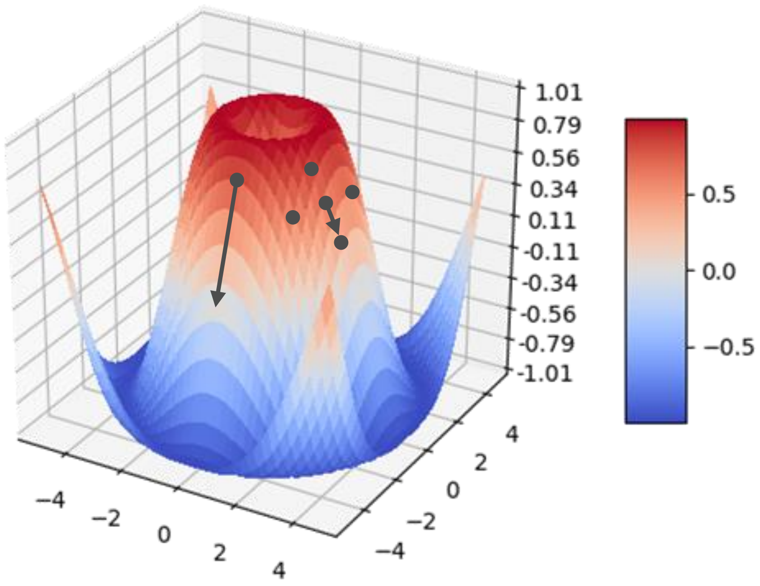


Internal optimizers

Dresden Database Systems Group

optim (R)/scipy.optimize.minimize (python)

- Takes a cost function and a starting configuration for all parameters.
- Supports several methods (Nelder-Mead, BFGS, L-BFGS-B, …).
- Some methods support pre-calculated gradient methods or Hessian matrices.



$$f'(x_i) = \frac{df}{dx_i}, \forall i$$

Task

Dresden Database Systems Group

Step 0

- You will get a csv file from us. Load it in your language/environment.
- Explore the data in it.
- Write a cost function $f(x, y) \rightarrow z$.

Step 1

- Implement the Nelder-Mead simplex algorithm.* Use $\alpha = 1, \gamma = 2, \rho = 0.5, \sigma = 0.5$, and $t = 2$.
- Find the minimum in the data.

Step 2

- Use optim (minimize) to find the minimum with the conjugate gradient method:
$$f(x, y) = (1.5 - x + xy)^2 + (2.25 - x + xy^2)^2 + (2.625 - x + xy^3)^2$$
- Use the pre-calculated gradient $f'(x, y)$ to boost the calculation.
- Inspect the differences in run time.

*use your own implementation

Package suggestions

R

- -

python3

- numpy
- scipy
- (matplotlib.pyplot)

Exercise Appointment

Dresden Database Systems Group

We compare and discuss the results

- Tuesday, 05.11.2019,
- Consultation: n/a (Mail),
- Please prepare your solutions! Send us your code!

If you have questions, please mail us:

claudio.hartmann@tu-dresden.de Orga + Code
lucas.woltmann@tu-dresden.de Tasks + Python
lars.kegel@tu-dresden.de Tasks + R