UMassAmherst

The Commonwealth's Flagship Campus

MIE 524/624 Parametric Optimization- using search algorithms

Chaitra Gopalappa, PhD

Reference

- Chapter 8 – Murphy, 2023(Book 1)

Search Algorithms

$$Min\ f(\vec{x})$$
$$\vec{x} \in R^n$$

- Concept:
  - Start with some randomly chosen value of $\vec{x}$,
  - Apply transformation to move towards optimal value of $\vec{x}$,
  - Repeat until $\vec{x}$ coverges

Search Algorithms

$$Min\ f(\vec{x})$$
$$\vec{x} \in R^n$$

- Concept:
  - Start with some randomly chosen value of $\vec{x}$,
  - Apply transformation to move towards optimal value of $\vec{x}$,
  - Repeat until $\vec{x}$ coverges
- Main transformation

$$\vec{x}_{m+1} \leftarrow \vec{x}_m + \mu_m \vec{p}_m$$

- $\vec{p}_m$ = search direction
- μ_m= step-length

Varies by type of algorithm

- m = iteration number

Type of Search Algorithms

- Line search methods
  - Steepest descent
  - Newton’s
  - Quasi-Newton’s
- Trust-region methods
  - Levenberg-Marquardt

Line Search -

- Main transformation

$$\vec{x}_{m+1} \leftarrow \vec{x}_m + \mu_m \vec{p}_m$$

- $\vec{p}_m$ = search direction = $-\mathrm{B}_\mathrm{m}^{-1} \nabla f(\vec{x}_m)$

  - $\nabla f(\vec{x}_m) = \begin{bmatrix} \frac{\partial f}{\partial x_1}(\vec{x}_m) \\ \frac{\partial f}{\partial x_2}(\vec{x}_m) \\ \cdots \\ \frac{\partial f}{\partial x_n}(\vec{x}_m) \end{bmatrix}$

Gradient in vector notation

- Steepest descent algorithm
  - $B_m = I$ (identity matrix; square matrix with ones on the main diagonal and zeros elsewhere); $\boldsymbol{x} \in R^n \Rightarrow I$ is of size $n \times n$
- $\vec{x}_{m+1} \leftarrow \vec{x}_m - \mu_m \nabla f(\vec{x}_m)$

Steepest descent algorithm

- Main transformation

$$x[i]_{m+1} \leftarrow x[i]_m - \frac{\mu_m \partial f(\vec{x})}{\partial x[i]} \; for \; i = 1,2,\ldots,N$$

Algorithm

1. Set m = 1, select μ_m (tiny step)
2. Initialize $\vec{x}_m$ to an arbitrary feasible solution
3. Obtain $\frac{\partial f(\vec{x})}{\partial x[i]} \, for \; i = 1,2,\ldots,N$
4. Update $x[i]_{m+1}$; update μ_{m+1}
5. If all $\frac{\partial f(\vec{x})}{\partial x[i]}$=0, or sufficiently close STOP. Else set m = m +1, goto step 3

Pointers:

- Repeat above multiple times, with a different starting point in step 2;
- Here: μ_m is a hyperparameter
  - Try different μ_m (it can be a constant that does not change with m; or can be a function of m, e.g., $1/m$)
- Put a constraint on number of iterations

Steepest descent algorithm - example

- $Min\ f(\boldsymbol{x}), \boldsymbol{x} \in R^n; f(\boldsymbol{x}) = f([x_1, x_2]) = \quad x_1^2 + x_2^2$
- Main transformation

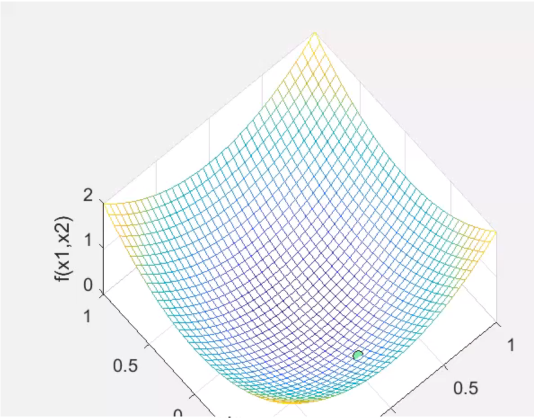
$$x_1 \leftarrow x_1 - \frac{\mu \partial f(\boldsymbol{x})}{\partial x_1}$$
$$x_2 \leftarrow x_2 - \frac{\mu \partial f(\boldsymbol{x})}{\partial x_2}$$

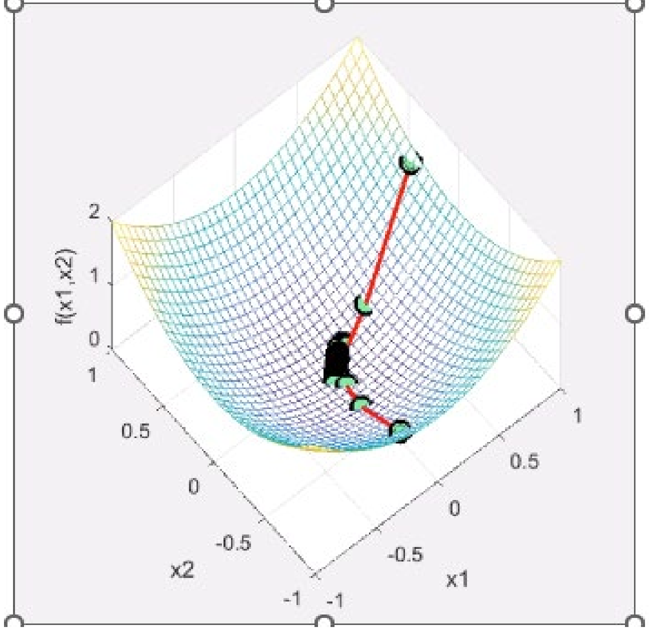
Algorithm

1. Set m = 1, select $\mu_m = 0.1$ (tiny step)
2. Initialize $[x_1, x_2] = [7{,}9]$ (an arbitrary feasible solution)
3. Obtain partial derivatives: $\frac{\partial f(\vec{x})}{\partial x_1} = 2x_1 = 2 \times 7; \frac{\partial f(\vec{x})}{\partial x_2} = 2x_2 = 2 \times 9$
4. Update $[x_1, x_2]$;

$$x_1 = x_1 - \frac{\mu \partial f(\boldsymbol{x})}{\partial x_1} = x_1 - \mu 2x_1 = 7 - 0.1 \times 2 \times 7$$
$$x_2 = x_2 - \frac{\mu \partial f(\boldsymbol{x})}{\partial x_2} = x_2 - \mu 2x_2 = 9 - 0.1 \times 2 \times 9$$

5. If all $\frac{\partial f(\vec{x})}{\partial x[i]}$=0, or sufficiently close STOP. Else set m = m +1, goto step 3





Projected gradient descent

$$Min\ f(\vec{x})$$
$$\vec{x} \in C^n$$

C^n *is a n-dimensional convex set, constrained problem*

$$\vec{x}_{m+1} \leftarrow P_C[\vec{x}_m - \mu_m \nabla f(\vec{x}_m)]$$

P_C *is the projection onto the feasible space.*

For example $\quad Minf(\vec{x}); \vec{a} \leq \vec{x} \leq \vec{b}$

$$P[x[i], a[i], b[i]] = \begin{cases} a[i]\ if\ x[i] \leq a[i] \\ b[i]\ if\ x[i] \geq b[i] \\ x[i]\ o/w \end{cases} ; \forall i \in \{1, \ldots, n\}$$

Gradient descent for constrained problems

- $Minf(\vec{x}); \vec{a} \leq \vec{x} \leq \vec{b}$
- Main transformation

$$P[x[i], a[i], b[i]] = \begin{cases} a[i] \ if \ x[i] \leq a[i] \\ b[i] \ if \ x[i] \geq b[i] \\ x[i] \ o/w \end{cases} ; \forall i \in \{1, .., n\}$$

$$x[i]_{m+1} \leftarrow x[i]_m - \frac{\mu_m \partial f(\vec{x})}{\partial x[i]} \ for \ i = 1,2, \ldots, N$$

Same as gradient descent except for the points in red

Algorithm

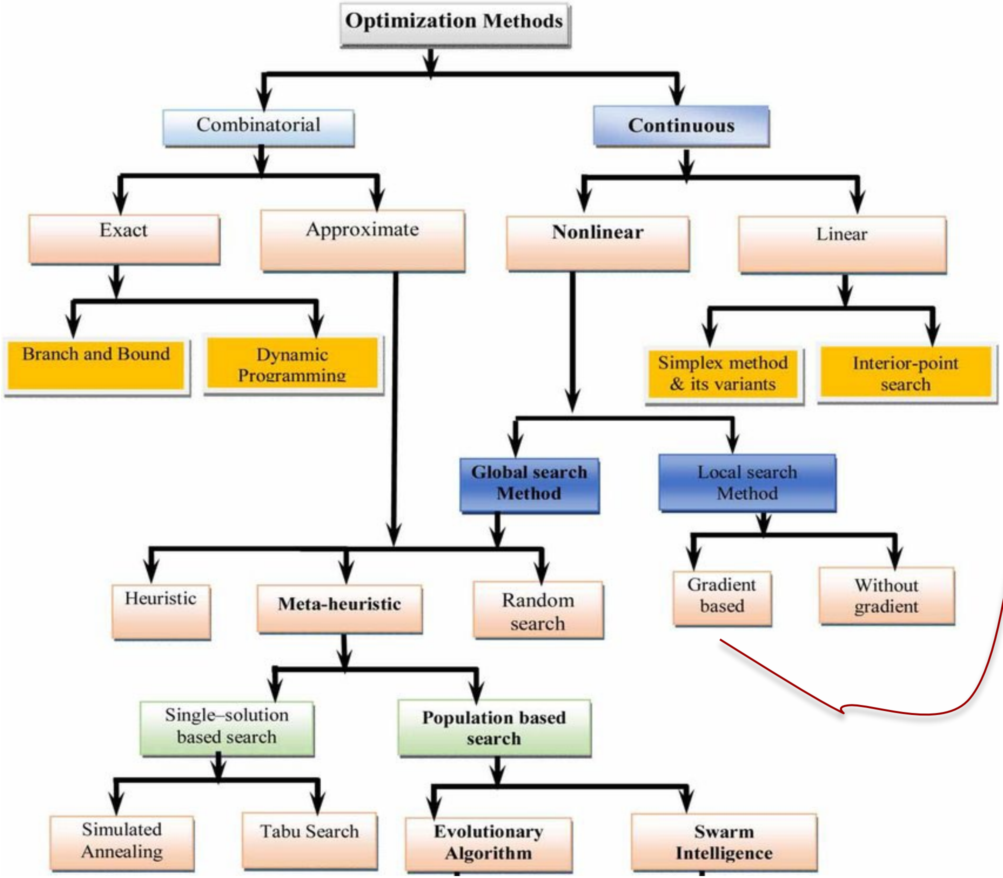
1. Set m = 1, select μ_m (tiny step), set M (max iterations) to a sufficiently large value
2. Initialize $\vec{x}_m$ to an arbitrary feasible solution
3. Obtain $\frac{\partial f(\vec{x})}{\partial x[i]} \ for \ i = 1,2, \ldots, N$
4. Update $x[i]_{m+1}$, Apply $x[i]_{m+1} = P[x[i], a[i], b[i]] \forall i$; update μ_{m+1}
5. If all $\frac{\partial f(\vec{x})}{\partial x[i]}$=0, or sufficiently close, or $m = M$ STOP. Else set $m = m + 1$, goto step 3

Pointers:

- Repeat above multiple times, with a different starting point in step 2;
- Here: μ_m is a hyperparameter
  - Try different μ_m (it can be a constant that does not change with m; or can be a function of m, e.g., $1/m$)
- Put a constraint on number of iterations

Type of Search Algorithms

- Line search methods
  - First selects search direction
  - Then moves in that direction with a certain step-size
- Trust-region methods
  - Selects search direction and step-size together



M. Janga Reddy, D. Nagesh Kumar; Evolutionary algorithms, swarm intelligence methods, and their applications in water resources engineering: a state-of-the-art review. *H2Open Journal* 1 January 2020; 3 (1): 135–188. doi: https://doi.org/10.2166/h2oj.2020.128

Gradient (or approximate gradient) - based search methods

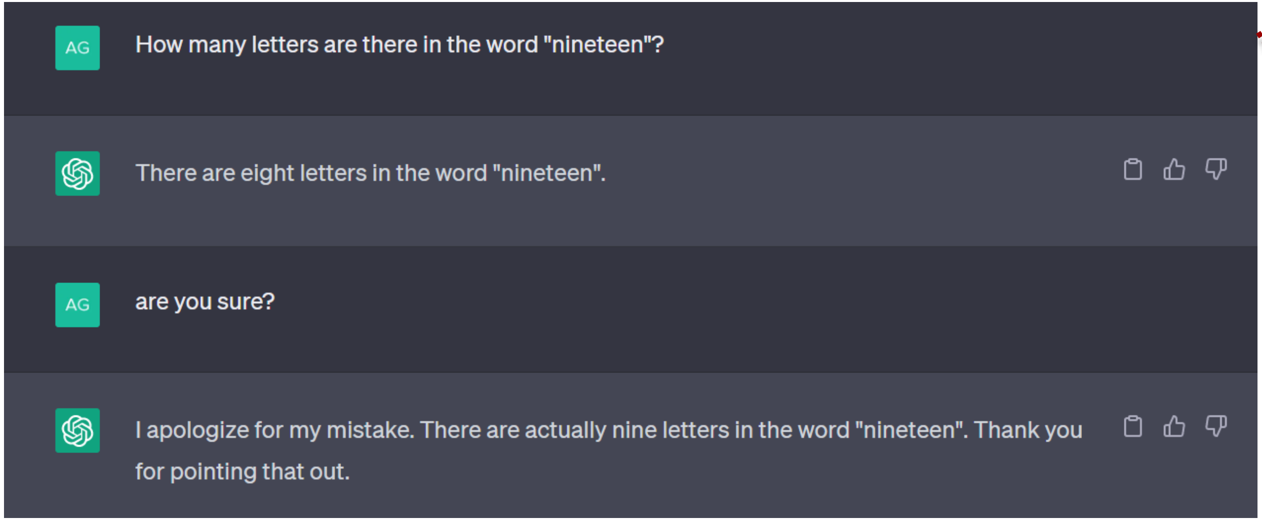
- Exact
- Line search methods
  - Steepest descent
  - Newton's
  - Quasi-Newton's
- Trust-region methods
  - Levenberg-Marquardt

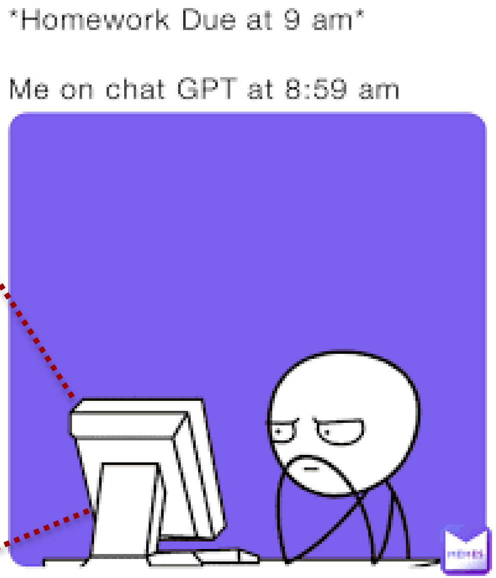
This class

Nocedal and Wright,
Numerical optimization, Springer, 2000

A2_assignment

- Posted on canvas
- You can use Generative AI apps to improve coding skills;
  - Be aware of its flaws





Questions?

Question

- Thus far - assumed we have the analytical function for estimation of the gradient
- How do we estimate gradient when we do not know the analytical form or cannot represent the system analytically?

UMassAmherst

The Commonwealth's Flagship Campus