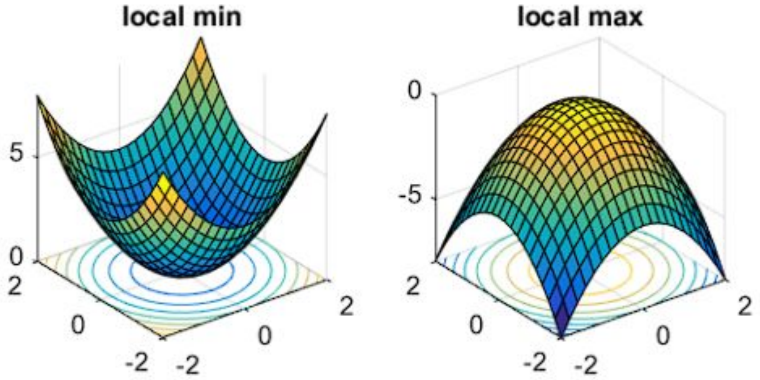
Topic 8: Applied Math (C) Optimization Methods

Group Number 9
Ryan Hammonds, Benjamin Pham, Gabriel Riegner
08 November 2022

Sec 1: Introduction

Overview of Optimization

- Optimization involves minimizing (or maximizing) an objective or loss function.
  $$\min_{\mathbf{x}\in\mathbb{R}^n} f(\mathbf{x})$$
- Includes optimizing unknowns in regression and classification problems to minimize error.
- Solved using:
  - Linear algebra:
    - $\mathbf{A}\mathbf{x} = \mathbf{b}$
  - Iterative methods
    - Gradient Descent
    - Newton's Method

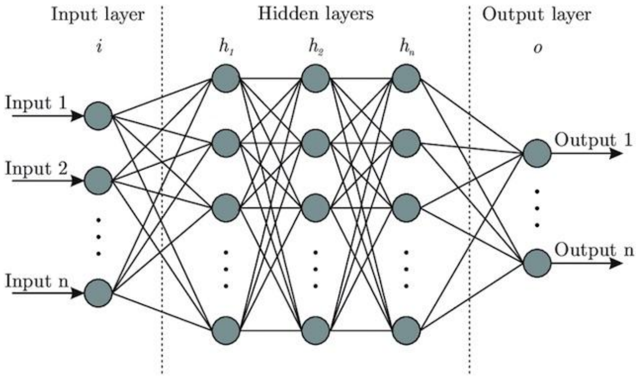


$$\nabla f(\mathbf{x}) = 0$$

Importance of the topic

- Optimization is required to solve regression and classification problems.
  - Parameter tuning
- Applies to simple and complex models.
  - Linear Regression
  - Neural networks
- Applications
  - Medicine
  - Economics / Finance
  - Computer vision
  - Speech recognition

$$
\begin{aligned}
Y_1 &= \beta_0 + \beta_1 X_1 + \epsilon_1 \\
Y_2 &= \beta_0 + \beta_1 X_2 + \epsilon_2 \\
\vdots \quad & \vdots \quad \vdots \\
Y_n &= \beta_0 + \beta_1 X_n + \epsilon_n
\end{aligned}
$$



Sec 2: Problem Formulation

#1 Problem formulation

1. Let $\mathbf{x}$ be a vector of unknown parameters.
2. Let $\mathbf{y}$ be a vector of the known targets.
3. Given an arbitrary loss function (e.g. L0, L1, or L2), iteratively minimize:

$$f(\mathbf{x}) = loss(\hat{\mathbf{y}}, \mathbf{y})$$

$$\min_{\mathbf{x} \in \mathbb{R}^n} f(\mathbf{x})$$

4. At each iterative step k, update $\mathbf{x}$ by subtracting either:
   a. the gradient $\nabla \cdot$ scaled by step size η

   $$\mathbf{x}_{k+1} = \mathbf{x}_k - \eta_k \nabla f(\mathbf{x}_k)$$

   b. the gradient $\nabla \cdot$ scaled by the inverse Hessian $\mathbf{H}^{-1}$

   $$\mathbf{x}_{k+1} = \mathbf{x}_k - \mathbf{H}(\mathbf{x}_k)^{-1} \nabla f(\mathbf{x}_k)$$

#2 Relation to Numerical Linear Algebra

In general, an optimization problem involves solving a linear system with multiple parameters to minimize the loss function.

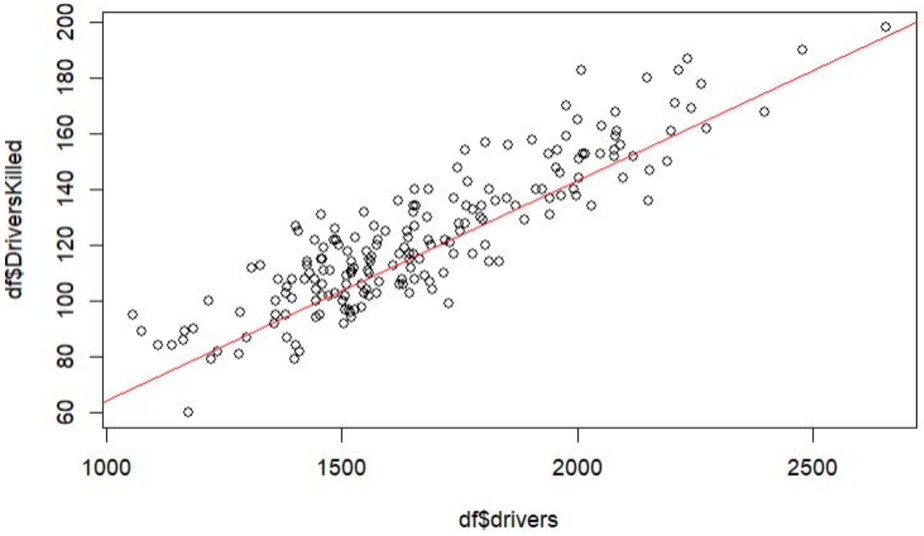
Solved via two main types of methods:

- Closed Form (Exact)

- Numerical (Estimation)

The solution of this linear system results in the set of the most optimal parameters

Simple Example

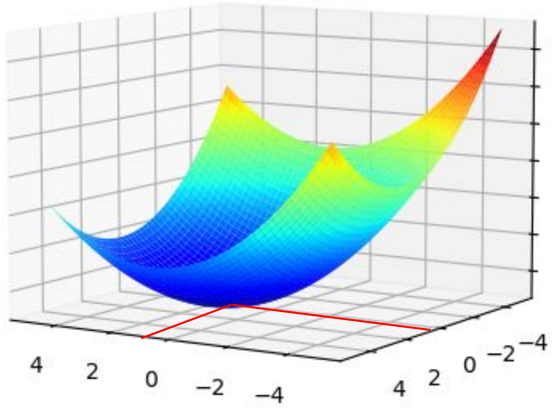


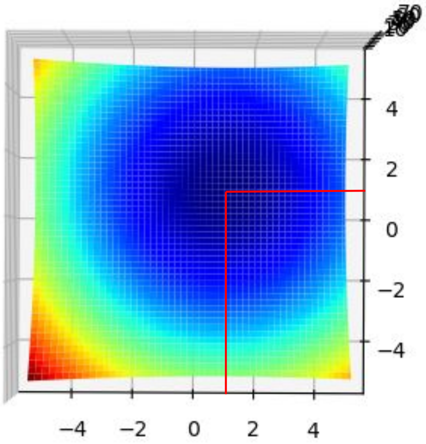
$$Y = \beta X + \epsilon$$

$$\underbrace{\begin{bmatrix} Y_1 \\ Y_2 \\ Y_3 \\ \vdots \\ Y_n \end{bmatrix} = \begin{bmatrix} 1 & x_{11} & x_{12} \\ 1 & x_{21} & x_{22} \\ 1 & x_{31} & x_{32} \\ \vdots & \vdots & \vdots \\ 1 & x_{n1} & x_{n2} \end{bmatrix}}_{\text{Known}} \underbrace{\begin{bmatrix} \beta_0 \\ \beta_1 \\ \beta_2 \end{bmatrix} + \begin{bmatrix} \epsilon_1 \\ \epsilon_2 \\ \epsilon_3 \\ \vdots \\ \epsilon_n \end{bmatrix}}_{\text{Unknown}}$$

Optimization Problem: Get β that creates the best fit line while minimizing the error term (distance between points and the best fit line)

Slightly More Complex Example





$$\begin{bmatrix} 1 & x_1 & x_2 \end{bmatrix} \begin{bmatrix} 6 & 0 & 0 \\ -2 & 1 & 0 \\ -2 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 \\ x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} 6-2x_1-2x_2 & x_1 & x_2 \end{bmatrix} \begin{bmatrix} 1 \\ x_1 \\ x_2 \end{bmatrix} = 6 - 2x_1 - 2x_2 + x_1^2 + x_2^2 = f(x_1, x_2)$$

Optimization helps find the pair that minimizes this function

#3 Approach of Numerical Linear Algebra (NLA)

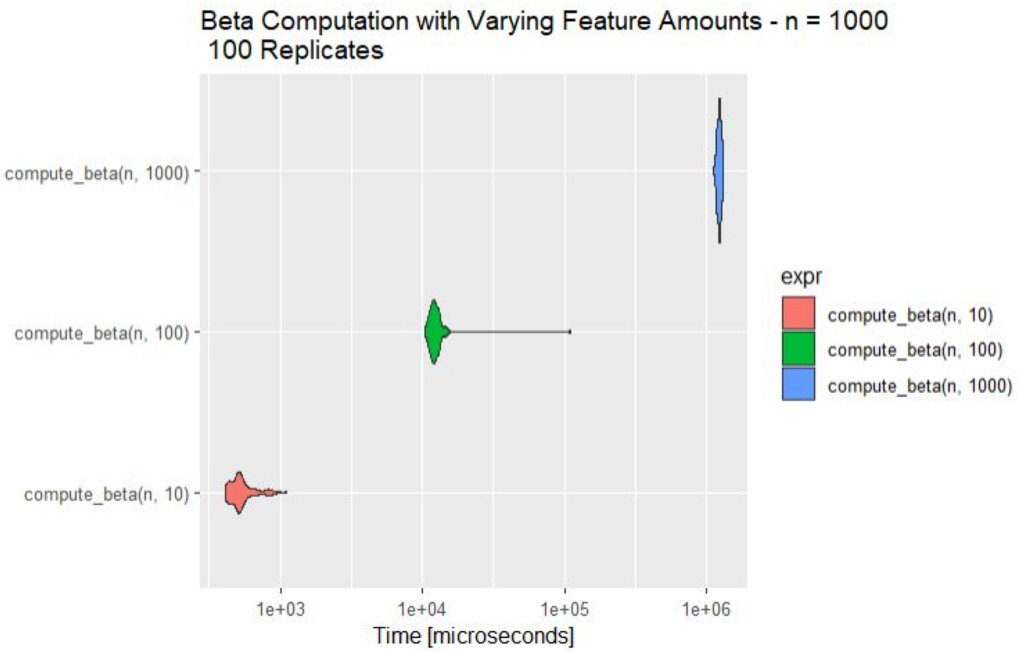
- Sometimes, the closed form solution is not available/too difficult to compute

  For example: in linear regression, solving for β analytically can be computationally expensive in high dimensional design matrices

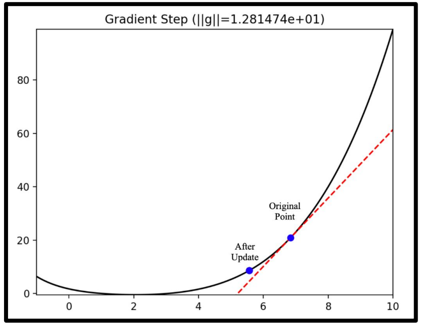
- Iterative methods such as **Stochastic Gradient Descent** and **Newton's Method** are used instead.

$$\hat{\beta} = \boxed{(X^T X)^{-1}} X^T y$$

Difficult to compute at high dimensions!



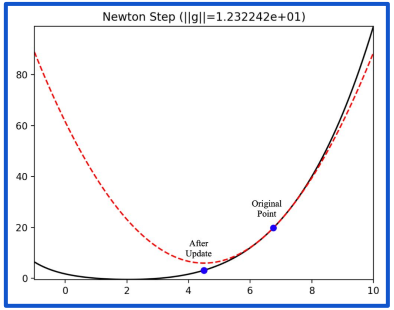
Iterative Optimization Algorithms



(Stochastic) Gradient Descent

- first order optimization
- direction of most rapid decrease, but no information about curvature
- slow and imprecise (error decreases linearly)

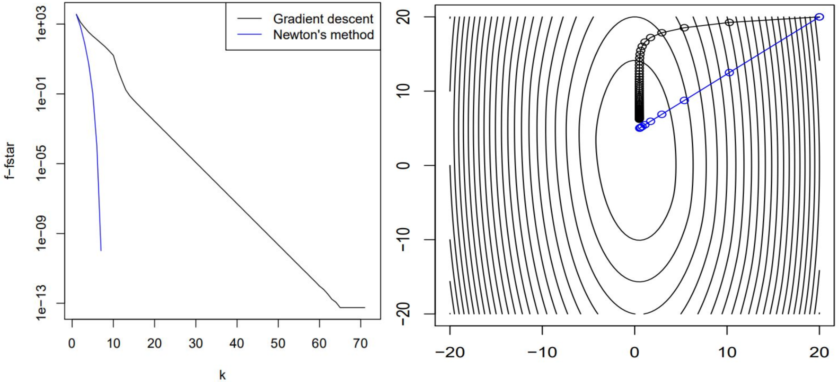
$$H = \begin{bmatrix} \frac{\partial^2 f}{\partial x_1^2} & \frac{\partial^2 f}{\partial x_1 \partial x_2} & \cdots & \frac{\partial^2 f}{\partial x_1 \partial x_d} \\ \frac{\partial^2 f}{\partial x_2 \partial x_1} & \frac{\partial^2 f}{\partial x_2^2} & \cdots & \frac{\partial^2 f}{\partial x_2 \partial x_d} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial^2 f}{\partial x_d \partial x_1} & \frac{\partial^2 f}{\partial x_d \partial x_2} & \cdots & \frac{\partial^2 f}{\partial x_d^2} \end{bmatrix}$$



Newton's Method

- second order optimization
- fast and precise convergence (error decreases quadratically)
- fewer steps, but each step requires n^2 to compute Hessian and n^3 to invert it
- Hessian matrix might not be invertible (positive definite)

comparison: k steps to convergence



Sec 3: State of the Art (SOTA)

SOTA: Quasi Newton Methods

- Computing the inverse Hessian is costly
- Replace Hessian with an approximation using Taylor expansion

$$\mathbf{B}_k^{-1} \approx \mathbf{H}(\mathbf{x}_k)^{-1}$$

- Hessian approximations must satisfy:

$$\mathbf{B}_{k+1}\Delta\mathbf{x} = \nabla f(\mathbf{x}_{k+1}) - \nabla f(\mathbf{x}_k) \qquad \Rightarrow \mathbf{A}\mathbf{x} = \mathbf{b}$$

- Methods to solve $\mathbf{B}_{k+1}$
  - BFGS
  - Broyden
  - SR1

Sec 4: Experiment

Experimental setup

libraries and tools:

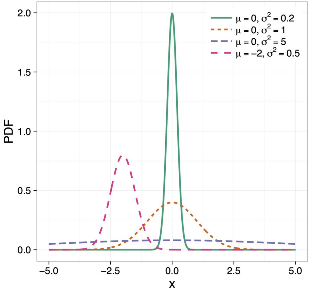
NumPy matplotlib SciPy

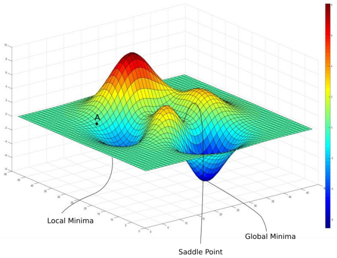
```python
def newtons_method(g, max_its, w, **kwargs):
    gradient = gradient(g)
    hessian = hessian(g)
    ...
```

final report:

datasets/benchmarks:

simulations





sklearn.datasets

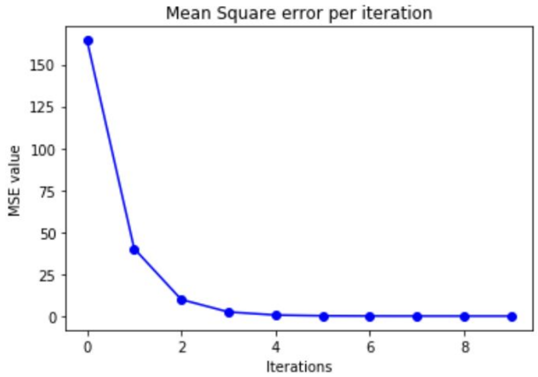
	sepal_length	sepal_width	petal_length	petal_width	species
0	5.1	3.5	1.4	0.2	setosa
1	4.9	3.0	1.4	0.2	setosa
2	4.7	3.2	1.3	0.2	setosa
3	4.6				
4	5.0				
...	...				

	subject	timepoint	event	region	signal
0	s13	18	stim	parietal	-0.017552
1	s5	14	stim	parietal	-0.080883
2	s12				
3	s11				
4	s10				
...	...				

	species	island	bill_length_mm	bill_depth_mm	flipper_length_mm	body_mass_g
0	Adelie	Torgersen	39.1	18.7	181.0	3750.0
1	Adelie	Torgersen	39.5	17.4	186.0	3800.0
2	Adelie	Torgersen	40.3	18.0	195.0	3250.0
3	Adelie	Torgersen	NaN	NaN	NaN	NaN
4	Adelie	Torgersen	36.7	19.3	193.0	3450.0
...	...	...	...	...	...	...

Sec 5: Concluding remarks

Conclusion



Next steps

- **simulation-based optimization** to benchmark linear algebra, gradient descent, and (quasi)-Newton methods

- **evaluation metrics**: **total step number**, **runtime**, **accuracy** with an increasing number of samples and parameters

- Aim to compare performance of each optimization algorithm as number of parameters increase from 1 to 1000

- apply same benchmarks under real data conditions, including regression and classification problems

- Smallest dataset: Linnerud **(20 observations, 3 features)**

- Largest dataset: Breast Cancer Wisconsin (Diagnostic) dataset **(569 observations, 30 features)**

Expected outcomes

- As the number of features/parameters increases, the step-to-convergence for all algorithms will increase resulting in a longer runtime

- gradient descent will be efficient for small problems, and state-of-the-art methods will be more efficient for high-dimensional, large data examples

- The size of the problem will dictate the optimization algorithm

References

https://course.ece.cmu.edu/~ece739/lectures/18739-2020-spring-lecture-08-second-order.pdf

https://www.psychologie.uni-heidelberg.de/ae/meth/team/mertens/blog/hessian.nb.html

https://www.inf.ed.ac.uk/teaching/courses/irds/miniproject-datasets.html

Hardt, M., & Recht, B. (2022). Patterns, predictions, and actions: A story about machine learning. Princeton University Press.

Boyd, S., & Vandenberghe, L. (2004). Convex optimization. Cambridge university press.