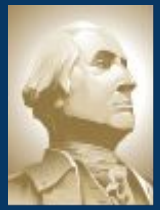
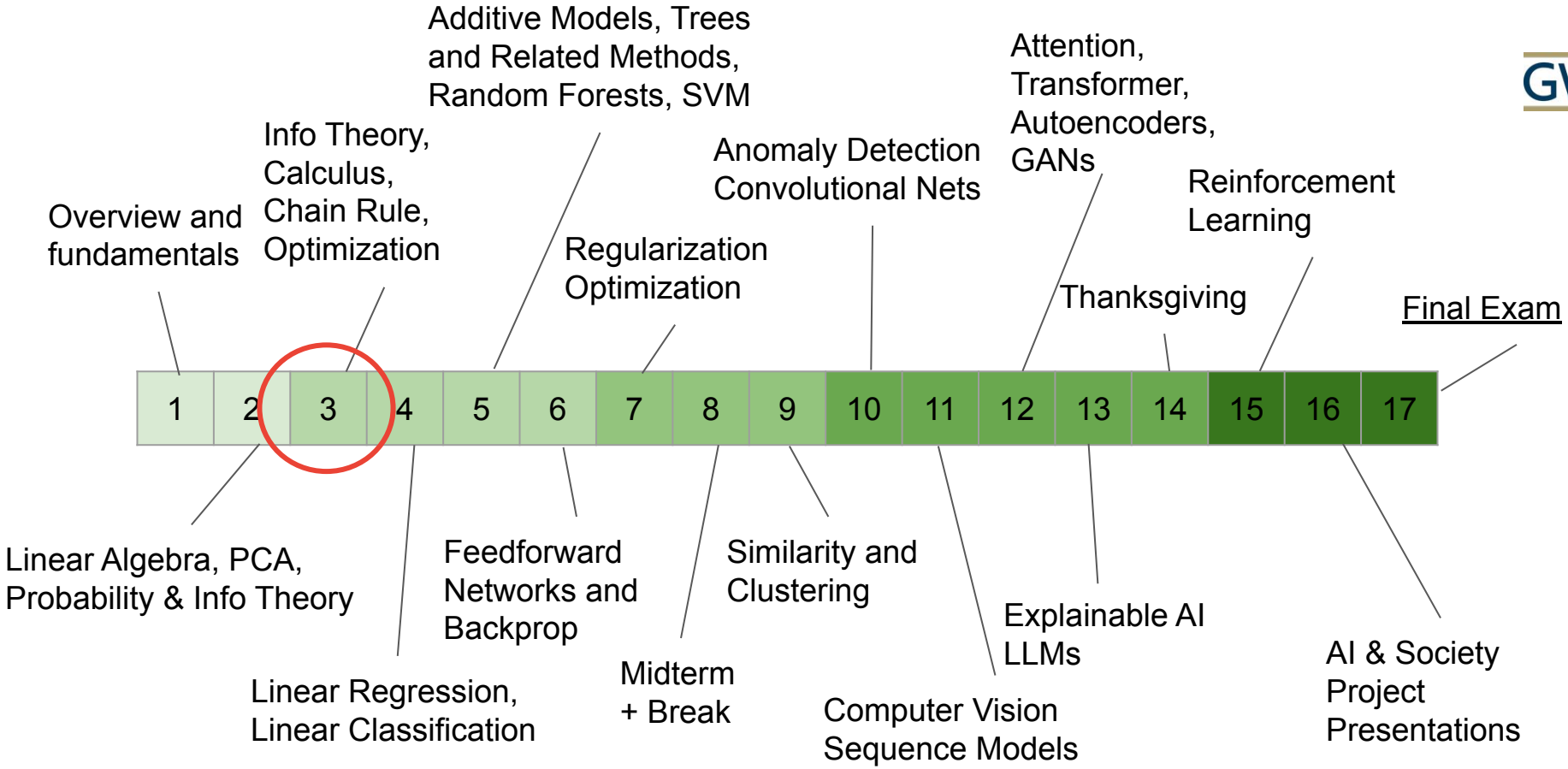


THE GEORGE WASHINGTON UNIVERSITY
WASHINGTON, DC

CS 4364/6364
Machine Learning

Fall Semester 9/7/2023
Lecture 5.
Numerical Computation and Gradient Optimization

John Sipple
jsipple@gwu.edu


Additive Models, Trees and Related Methods, Random Forests, SVM
Attention, Transformer, Autoencoders, GANs
Info Theory, Calculus, Chain Rule, Optimization
Anomaly Detection Convolutional Nets
Reinforcement Learning
Overview and fundamentals
Regularization Optimization
Thanksgiving
Final Exam
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
Linear Algebra, PCA, Probability & Info Theory
Feedforward Networks and Backprop
Similarity and Clustering
Explainable AI LLMs
AI & Society Project Presentations
Linear Regression, Linear Classification
Midterm + Break
Computer Vision Sequence Models

Overflow & Underflow

Overflow and Underflow

Optimization often runs up to the finite precision limits of the computer

Underflow:

- Very small number rounds down to zero
- Can cause problems with division or logarithm

Overflow:

- Large magnitudes approximated as +/- infinity
- Next operation will usually fail with Not A Number…

Gradient-Based Optimization

Optimization

Task of minimizing or maximizing some function $f(\boldsymbol{x})$ by altering x

$f(\boldsymbol{x})$ is called a **cost function**, **loss function**, or **error function**

Optimum: $\boldsymbol{x}^* = \arg\min f(\boldsymbol{x})$ (the point where $f(x)$ is least)

The Derivative – a tool for optimization

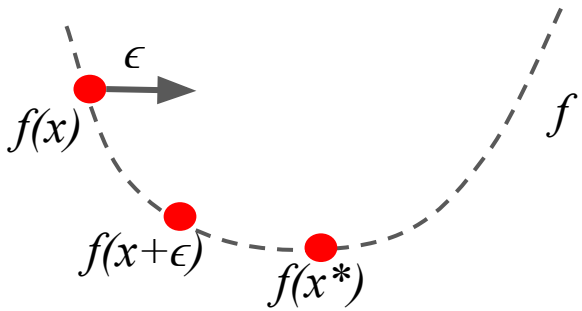
For a real-valued function $y = f(x)$, the derivative is

$$f'(x) \text{ or } \frac{dy}{dx}$$

Approximates the value of f some small displacement ϵ from x:

$$f(x + \epsilon) \approx f(x) + \epsilon f'(x)$$

Gradient descent - follow the derivative to x^*



Critical Points

Points where $f'(x) = 0$

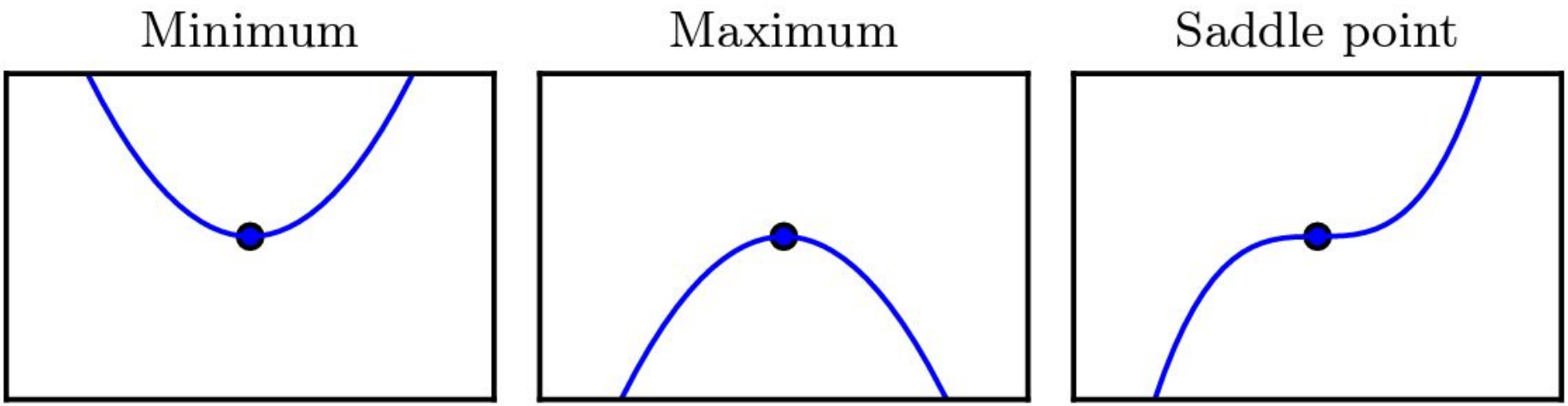


Figure 4.2

Approximate Optimization

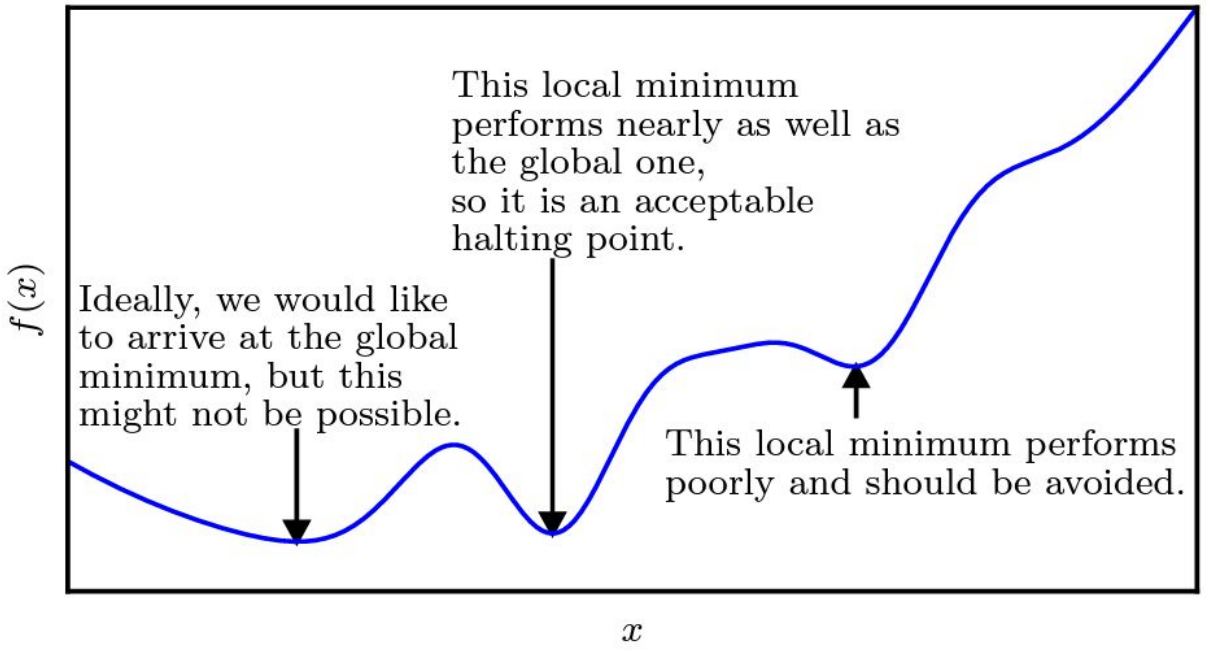


Figure 4.3

The gradient

ML Loss functions have multiple inputs:

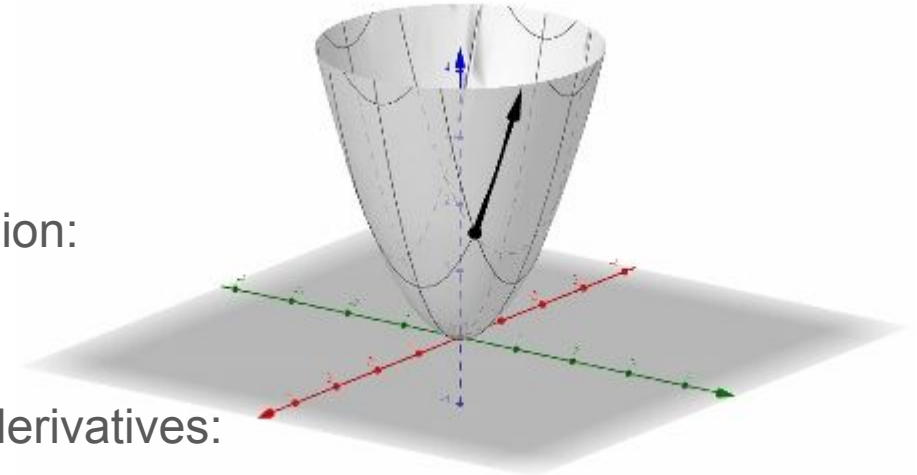
$$f : \mathbb{R}^n \to \mathbb{R}$$

Partial derivative with respect to i^{th} dimension:

$$\frac{\partial}{\partial x_i} f(\boldsymbol{x})$$

Gradient: n-dimensional vector of partial derivatives:

$$\nabla_{\boldsymbol{x}} f(\boldsymbol{x}) = \begin{bmatrix} \frac{\partial}{\partial x_1} f(\boldsymbol{x}) \\ \frac{\partial}{\partial x_2} f(\boldsymbol{x}) \\ \vdots \\ \frac{\partial}{\partial x_n} f(\boldsymbol{x}) \end{bmatrix}$$



Source: https://study.com/academy/lesson/directional-derivatives-gradient-of-f-and-the-min-max.html

Tracking to the minimum with the Gradient

Define a unit vector $\boldsymbol{u}$ in the direction of the gradient:

$$f(\boldsymbol{x} + \epsilon \boldsymbol{u})$$

$$\frac{\partial}{\partial \epsilon} f(\boldsymbol{x} + \epsilon \boldsymbol{u})$$

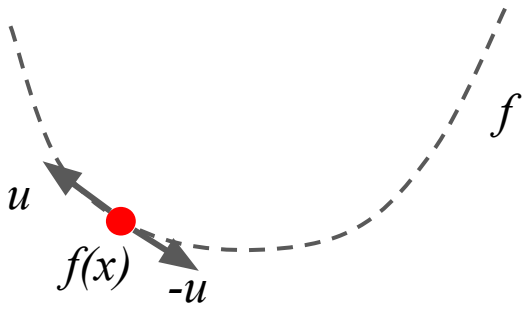
When $\epsilon = 0$, evaluates to $\boldsymbol{u}^\top \nabla_{\boldsymbol{x}} f(\boldsymbol{x})$

Since we want to minimize f:

$$\min_{\boldsymbol{u}, \boldsymbol{u}^\top \boldsymbol{u} = 1} \boldsymbol{u}^\top \nabla_{\boldsymbol{x}} f(\boldsymbol{x}) = \min_{\boldsymbol{u}, \boldsymbol{u}^\top \boldsymbol{u} = 1} ||\boldsymbol{u}||_2 ||\nabla_{\boldsymbol{x}} f(\boldsymbol{x})||_2 \cos\theta$$

Min achieved when following the opposite direction:

$$\boldsymbol{x}' = \boldsymbol{x} - \epsilon \nabla_{\boldsymbol{x}} f(\boldsymbol{x})$$



The Jacobian

If the function has m inputs and n outputs:

$$\boldsymbol{f} : \mathbb{R}^m \to \mathbb{R}^n$$

The derivative m-by-n matrix is called the Jacobian $\boldsymbol{J} \in \mathbb{R}^{n \times m}$ of $\boldsymbol{f}$:

$$\mathbf{J} = \begin{bmatrix} \frac{\partial}{\partial x_1} f(\boldsymbol{x})_1 & \frac{\partial}{\partial x_2} f(\boldsymbol{x})_1 & \cdots & \frac{\partial}{\partial x_m} f(\boldsymbol{x})_1 \\ \frac{\partial}{\partial x_1} f(\boldsymbol{x})_2 & \frac{\partial}{\partial x_2} f(\boldsymbol{x})_2 & \cdots & \frac{\partial}{\partial x_m} f(\boldsymbol{x})_2 \\ & \cdots & & \\ \frac{\partial}{\partial x_1} f(\boldsymbol{x})_n & \frac{\partial}{\partial x_2} f(\boldsymbol{x})_n & \cdots & \frac{\partial}{\partial x_m} f(\boldsymbol{x})_n \end{bmatrix}$$

The Hessian

Derivative of a derivative, **Curvature**, is the rate of change of slope.

For a loss function:

$$f : \mathbb{R}^n \to \mathbb{R}$$

We have an n-by-n matrix:

$$\mathbf{H}(f)(\boldsymbol{x})_{i,j} = \begin{bmatrix} \frac{\partial^2}{\partial x_1^2} f(\boldsymbol{x}) & \frac{\partial^2}{\partial x_1 \partial x_2} f(\boldsymbol{x}) & \cdots & \frac{\partial^2}{\partial x_1 \partial x_m} f(\boldsymbol{x}) \\ \frac{\partial^2}{\partial x_2 \partial x_1} f(\boldsymbol{x}) & \frac{\partial^2}{\partial x_2^2} f(\boldsymbol{x}) & \cdots & \frac{\partial^2}{\partial x_2 \partial x_n} f(\boldsymbol{x}) \\ \cdots & & & \\ \frac{\partial^2}{\partial x_n \partial x_1} f(\boldsymbol{x}) & \frac{\partial^2}{\partial x_n \partial x_2} f(\boldsymbol{x}) & \cdots & \frac{\partial^2}{\partial x_n^2} f(\boldsymbol{x}) \end{bmatrix}$$

The Hessian is the *Jacobian of the Gradient*

Curvature

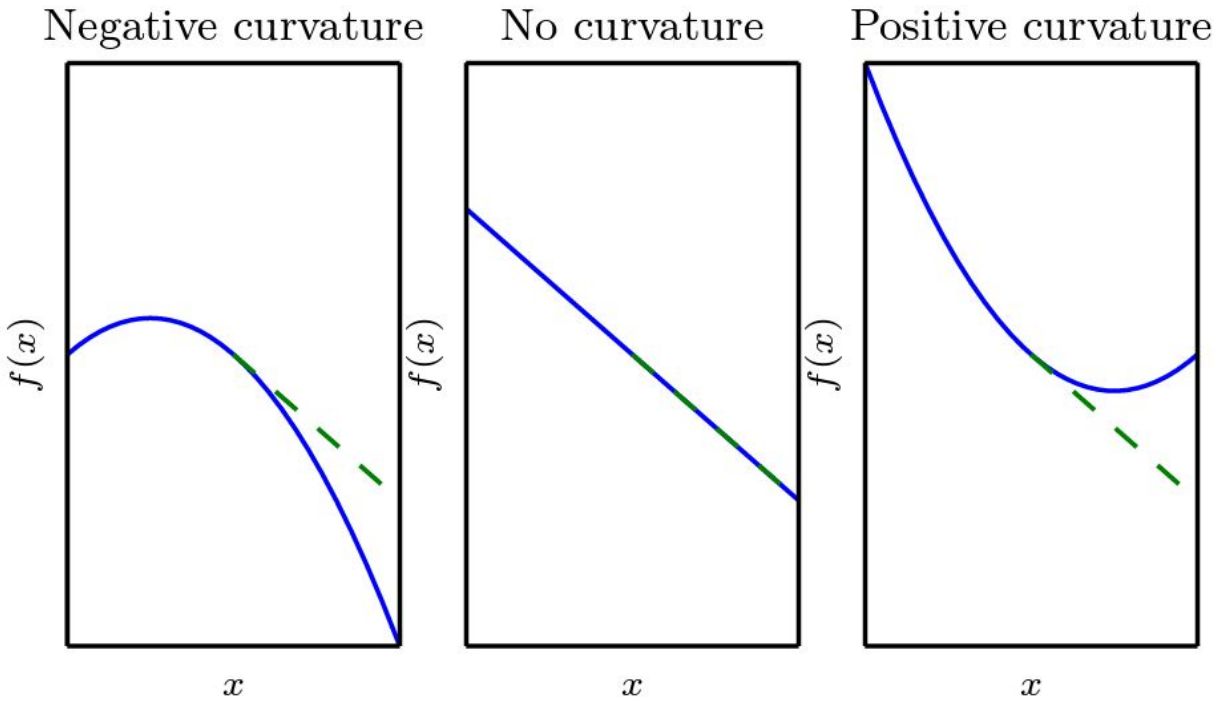


Figure 4.4

Properties of the Hessian

Since differential operators are commutative:

$$\frac{\partial^2}{\partial x_i \partial x_j} f(\boldsymbol{x}) = \frac{\partial^2}{\partial x_j \partial x_i} f(\boldsymbol{x})$$

The Hessian is real & symmetric

$$H_{i,j} = H_{j,i}$$

We can get real eigenvalues and real eigenvectors

2nd Order Taylor Series Approximation

In the neighborhopod of $\boldsymbol{x}^{(0)}$, $f(\boldsymbol{x})$ can be approximated as:

$$f(\boldsymbol{x}) \approx f(\boldsymbol{x}^{(0)}) + (\boldsymbol{x} - \boldsymbol{x}^{(0)})^\intercal \boldsymbol{g} + \frac{1}{2}(\boldsymbol{x} - \boldsymbol{x}^{(0)})^\intercal \boldsymbol{H}(\boldsymbol{x} - \boldsymbol{x}^{(0)})$$

With a learning rate of ϵ, the new point is at $\boldsymbol{x}^{(0)} - \epsilon \boldsymbol{g}$, and substituting:

$$f(\boldsymbol{x}^{(0)} - \epsilon \boldsymbol{g}) \approx f(\boldsymbol{x}^{(0)}) - \epsilon \boldsymbol{g}^\intercal \boldsymbol{g} + \frac{1}{2}\epsilon^2 \boldsymbol{g}^\intercal \boldsymbol{H} \boldsymbol{g}$$

Original point

Expected Improvement

Curvature Correction

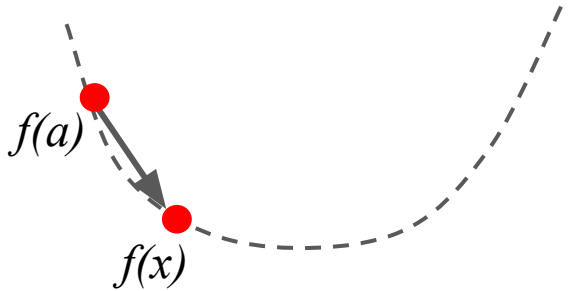
Taylor Series Approximation

Taylor series of a function is an infinite sum of terms that are expressed in terms of the function's derivatives at a single point.

$$f(x) = f(a) + \frac{f'(a)}{1!}(x-a) + \frac{f''(a)}{2!}(x-a)^2 + \frac{f'''(a)}{3!}(x-a)^3 + \dots$$

In practice, rarely use more than 2nd derivative.

$$f(x) \approx f(a) + \frac{f'(a)}{1!}(x-a) + \frac{f''(a)}{2!}(x-a)^2$$



2nd Derivative Test

Helps to identify **minimum**, **maximum** or **saddle points**.

Minimum: if the Hessian is positive definite (all positive eigenvalues)

Maximum: if the Hessian is negative definite (all negative eigenvalues)

Saddle point: if some positive and negative eigenvalues in Hessian

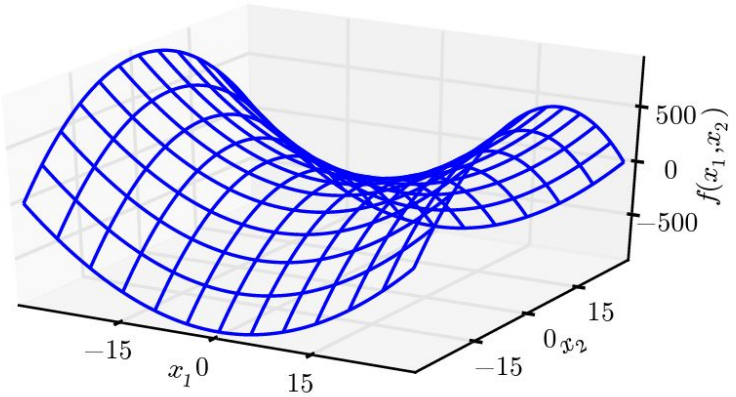


Figure 4.5

Condition Number of a Matrix

For n-by-n matrix $\boldsymbol{A} \in \mathbb{R}^{n \times n}$ the **condition number** is the ratio of its largest to smallest eigenvalue:

$$\max_{i,j} \left| \frac{\lambda_i}{\lambda_j} \right|$$

Large condition numbers lead to numeric instabilty.

If the condition number of the Hessian is large, gradient descent performs poorly.

Newton's Method

Use the curvature to adjust the step size:

$$f(\boldsymbol{x}) \approx f(\boldsymbol{x}^{(0)}) + (\boldsymbol{x} - \boldsymbol{x}^{(0)})^\intercal \nabla_{\boldsymbol{x}} f(\boldsymbol{x}^{(0)}) + (\frac{1}{2}\boldsymbol{x} - \boldsymbol{x}^{(0)})^\intercal \boldsymbol{H}(f)(\boldsymbol{x}^{(0)})(\boldsymbol{x} - \boldsymbol{x}^{(0)})$$

If f is positive definite and quadratic, Newton's Method jumps to the minimum:

$$\boldsymbol{x}* = \boldsymbol{x}^{(0)} - \boldsymbol{H}(f)(\boldsymbol{x}^{(0)})^{-1} \nabla_{\boldsymbol{x}} f(\boldsymbol{x}^{(\boldsymbol{0})})$$

Otherwise, multiple iterations are required to converge of $x*$

Constrained Optimization

Constrained Optimization

Constrained Optimization: rather than optimize over all possible points, apply constraints to some subset of the solution space $\mathbb{S}^n \subset \mathbb{R}^n$

feasible point: $\boldsymbol{x} \in \mathbb{S}$

- Guide gradient descent to stay inside S
- Transform into a different unconstrained optimization
- Karush-Kuhn-Tucker (KKT), generalized Lagrangian

Constrained Optimization with KKT

"Generalized Lagrangian"

Constraint $\mathbb{S}$ in terms of **equations** $g^{(i)}(\boldsymbol{x})$ and **inequalities** $h^{(i)}(\boldsymbol{x})$

$$\mathbb{S} = \{\boldsymbol{x} | \forall i, g^{(i)}(\boldsymbol{x}) = 0 \text{ and } \forall j, h^{(j)}(\boldsymbol{x}) \leq 0\}$$

Add in new variables, λ_i, and α_i, for each constraint:

$$L(\boldsymbol{x}, \boldsymbol{\lambda}, \boldsymbol{\alpha}) = f(\boldsymbol{x}) + \sum_i \lambda_i g^{(i)}(\boldsymbol{x}) + \sum_j \alpha_j h^{(j)}(\boldsymbol{x})$$

Original minimization function

Constraints

Constrained Optimization with KKT

If $f(\boldsymbol{x})$ has at least one feasible point and $f(\boldsymbol{x})$ cannot be ∞ we can write the optimization problem:

$$\min_{x} \max_{\lambda} \max_{\boldsymbol{\alpha},\boldsymbol{\alpha}\geq 0} L(x, \boldsymbol{\lambda}, \boldsymbol{\alpha})$$

- Has the same objective function value and set of optimal point $\boldsymbol{x}$ as

$$\min_{\boldsymbol{x}\in\mathbb{S}} f(\boldsymbol{x})$$

- When constraints are satisfied:

$$\max_{\lambda} \max_{\boldsymbol{\alpha},\boldsymbol{\alpha}\geq 0} L(x) = f(x)$$

- When constraints are not satisfied:

$$\max_{\lambda} \max_{\boldsymbol{\alpha},\boldsymbol{\alpha}\geq 0} L(x) = \infty$$

Guarantee: no infeasible point can be optimal

Constrained Optimization with KKT

Can reformulate the problem as a maximization:

$$\min_{x} \max_{\lambda} \max_{\alpha,\alpha\geq 0} -f(x) + \sum_{i} \lambda_i g^{(i)}(x) + \sum_{j} \alpha_j h^{(j)}(x)$$

or

$$\max_{x} \min_{\lambda} \min_{\alpha,\alpha\geq 0} f(x) + \sum_{i} \lambda_i g^{(i)}(x) - \sum_{j} \alpha_j h^{(j)}(x)$$

Constrained Optimization with KKT

KKT conditions for a point to be optimal:

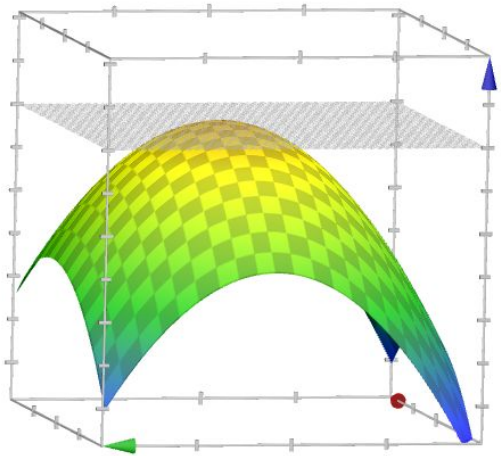
- Gradient is zero at $\boldsymbol{x}^*$
- All constraints on both $\boldsymbol{x}$ and the KKT multipliers ($\boldsymbol{\lambda}$, $\boldsymbol{\alpha}$) are satisfied
- Inequality constraints exhibit “complementary slackness”:

$$\boldsymbol{\alpha} \odot \boldsymbol{h}(\boldsymbol{x}) = \mathbf{0}$$

Constrained Optimization Example

The Problem

$$\begin{aligned} \text{maximize} \quad & -(x-2)^2 - 2(y-1)^2 \\ & \text{subject to:} \\ & x + 4y \leq 3 \\ & -x + y \leq 0 \end{aligned}$$



↘	From	To
x	-10.0000	10.0000
y	-10.0000	10.0000
z	-343.390	106.462

Constrained Optimization Example

The Lagrangian is:

$$L(x, y, \alpha_1, \alpha_2) = -(x-2)^2 - 2(y-1)^2 + \alpha_1(3 - x - 4y) + \alpha_2(x - y)$$

This gives the following KKT Conditions:

$$\frac{\partial L}{\partial x} = -2(x-2) - \alpha_1 + \alpha_2 = 0$$

$$\frac{\partial L}{\partial y} = -4(y-1) - 4\alpha_1 - \alpha_2 = 0$$

$$\alpha_1(3 - x - 4y) = 0$$

$$\alpha_2(x - y) = 0$$

$$\alpha_1, \alpha_2 \geq 0$$

Constrained Optimization Example

We have two complementary conditions, i.e., Complementary Slackness ($\boldsymbol{\alpha} \odot \hbar(x) = \mathbf{0}$) with four possible cases:

1. $\alpha_1 = \alpha_2 = 0 \rightarrow x = 2, y = 1$
2. $\alpha_1 = 0, x - y = 0 \rightarrow x = \frac{4}{3}, \alpha_2 = -\frac{4}{3}$
3. $3 - x - 4y = 0, \alpha_2 = 0 \rightarrow x = \frac{5}{3}, y = \frac{1}{3}, \alpha_1 = \frac{2}{3}$
4. $3 - x - 4y = 0, x - y = 0 \rightarrow x = \frac{3}{5}, y = \frac{3}{5}, \alpha_1 = \frac{22}{25}, \alpha_2 = -\frac{48}{25}$

Optimal Solution: $x^* = \frac{5}{3}, y^* = \frac{1}{3}, f(x^*, y^*) = -\frac{4}{9}$