IE 510 Applied Nonlinear Programming

Lecture 0: Introduction

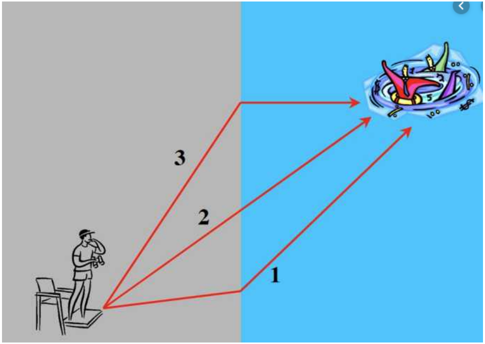
Ruoyu Sun

Jan 21, 2020

Outline

History of Optimization: Fermat's Principle

Fermat's principle: light travel in shortest path

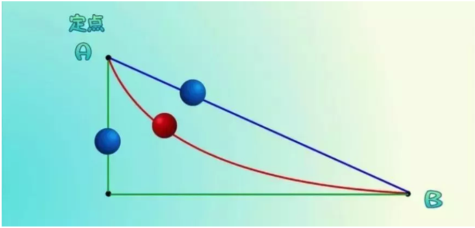


Nature is searching for "optimum"!

History of Optimization: Brachistochrone

Bernolli's chllenge 1696: Brachistochrone (shortest time curve).

Solved by John Bernolli, Netwon, Leibniez, etc.



Euler invented calculus of variations.

History of Optimization: Early Methods

Pierre De Fermat and Joseph-Louis Lagrange first found calculus-based formulae for identifying optima.

Isaac Newton (17xx) and Johann C.F. Gauss (1824) first proposed iterative methods to search for an optimum.

- Newton method
- Gauss-Seidel method

Steepest descent method (rooted in unpublished notes of Riemann in 1863.

We will learn all the above in this class.

References: https://empowerops.com/en/blogs/2018/12/6/brief-history-of-optimization.

History of Optimization: since 1900

Linear programming: Kantorovich (1939, Nobel prize); George Dantzig (1947, Stanford); John von Neumann.

1940s-1970s: classic optimization approaches were developed rapidly and peaked in the 1970s.

1980-2000: interior point methods.

After 2010: ?large-scale optimization?

What is Optimization (in general)

What is optimization?

Example 1: physics. Light travels in shortest path

Example 2: AI. Is our brain doing optimization?

Example 3: Operations of companies.
McKincy efficiency manual: to improve the efficiency, you need to optimize the process/allocation/etc.

Japanese companies are relatively bad at this.

What is Optimization (in general)

What is optimization?

Example 1: physics. Light travels in shortest path

Example 2: AI. Is our brain doing optimization?

Example 3: Operations of companies.
McKincy efficiency manual: to improve the efficiency, you need to optimize the process/allocation/etc.
Japanese companies are relatively bad at this.

What is Optimization (mathematically)

$$\begin{array}{ll} \text{minimize} & f(x) \\ \text{subject to} & x \in X \end{array} \tag{1}$$

- x: the decision variable (discrete, continuous)
- f : the objective function (differentiable, convex, linear...)
- X: the feasible region (convex, nonempty,...)

Key questions:

- When is the problem feasible?
- Does the optimal solution exist?
- How to determine if a feasible x is an optimal solution?
- How to find an optimal solution? (not by exhaustive search)

How To Use Optimization

Questions

1. How to model the problem by optimization?

   This is not just a math problem.

   It's about understanding the core aspects, extract the key elements, figure out the logic.

   This is the most challenging part, if you work for a company.
2. How to solve it?
   Focus of traditional optimization course.

It is hard to teach modeling, but we will provide examples to help a bit.

We will teach the methods. Knowing the methods is a great help for modeling.

How To Use Optimization

Regular Questions

1. Is my problem easy to solve? (if no, don't spend time on it!)
2. Which algorithm should I use?
3. How fast should I get my results?
4. How do I know that the answer from my computer run is global minimum?

Other Questions

1. My cost function is nondifferentiable, what should I do?
2. Why does my algorithm become very very slow?

Each question may require a sub-area. Many of them unknown! But knowing the basics helps a lot.

Answer (partially): This course

The Main Topics to be Covered

- Unconstrained Optimization
  1. Optimality Conditions
  2. Gradient/Newton's Methods
  3. Conjugate Gradient Method
- Constrained Optimization
  1. Optimality Conditions
  2. Descent Methods
  3. Conditional Gradient Method
  4. Block Coordinate Descent Method
- Lagrangian Multiplier Theory
  1. Equality Constrained Problems
  2. Inequality Constrained Problems
  3. Linearly Constrained Problems

The Main Topics to be Covered (continued)

- Lagrangian Multiplier Method
  1. Penalty Method
  2. Method of Multipliers
- Other Topics (if time permits)
  1. Proximal gradient method
  2. Linear systems
  3. ADMM
  4. Accelerated methods
- How to apply the knowledge? Applications in Machine Learning, Signal Processing, Communication and others will be discussed together as the course continues
- For more advanced topics such as mirror descent, smoothing techniques, stochastic optimization, online algorithms, see IE 598 - BIG DATA OPTIMIZATION, Niao He

Difference With Other Courses

Difference with other courses:

Linear programming course: focus on linear programming. Won't talk about general gradient methods in detail

Convex optimization: focus on convex problems, especially linear programming and conic programming. Not so much on nonlinear problems (most machine learning problems are nonlinear)

Machine learning course: won't talk about convergence issue, and constrained problems.

My comment: if you want to understand machine learning algorithms, this course is the best fit (as introductory)

Outline

1. Introduction: What is Optimization
2. Course Introduction
3. Examples in Machine Learning
4. Mathematical Review

Organization

Instructor:
Ruoyu Sun (`ruoyus at illinois.edu`)
Assistant professor, ISE, CSL (affiliated) and ECE (affiliated)
209D Transportation Building

Administrative Details:
Lecture time/location: Tu and Th: 2:00pm - 3:20PM, Enigneering Hall 410C1
Office hour: Tu: 3:30 - 4:30pm, Transportation Bld. 209D, or by appointment

TA: Dawei Li (`dawei2@illinois.edu`)
Office hour: TBD
Location: TBD

Organization

Textbook
D. Bertsekas, *Nonlinear Programming*, Main Textbook
Luenberger and Y. Ye. *Linear and nonlinear programming*. , Reference
Nesterov, *Introductory Lectures on Convex Optimization: A Basic Course* Reference
Nocedal and S. Wright. *Numerical optimization*. Springer Science & Business Media, 2006.

Other relevant materials will be distributed and discussed throughout the course

Syllabus

1. Course website: will be available this week (announced via email; and also found in my homepage, teaching, 2020 Spring IE510)
2. Compass 2g should also be available this week
3. Syllabus will be available on course website
4. Some of the key points here

Components of the course

1. Numerical grade = homework (35 %) + 1 in-class exam (30 %) + class project (30%) + attendance (5%) + bonus points (up to 10%)
2. Homework will be assigned regularly, some of them are mathematical, some of them requires programming
3. Course slides will be distributed on the course website regularly, but not all results/materials will be on the slides
4. One in-class mid-term (Mar 24, Tuesday after Spring break)
5. Class project report
6. No final

Homework and Project Policy

- **Homework and project submission:** electronically via Compass2g (no email)
- You are given 3 "grace days" (self-granted extensions) which you can use to give yourself extra time without penalty.
- Instructor-granted extensions are only considered after all grace days are used and only given in exceptional situations.
- Late work handed in when you have run out of grace days leads to reduction of 20% of the total points of that assignment
- Hard deadline of 3 days past the original due date. Late submissions after the hard deadline (penalty or not) lead to ZERO point.
- Bonus points = bonus problems in homework/exam and/or excellent project
- No late project

Attendance

- Attendence measured by participation.
- In-class exercises are often.
- Final summary of the class may be assigned sometimes.

More on the course

1. Lectures will be delivered using both slides and in-class notes
2. **Pre-requisite**: The course is mostly self-contained, no explicit requirements on previous courses on optimization
3. **Pre-requisite**: basic knowledge about linear algebra and calculus is required
4. Helpful knowledge: probability, numerical linear algebra, complexity theory, machine learning
5. **Theoretical or practical**? Mixture. Lots of theoretical analysis, but will try to provide insights/discussion/applications whenever possible

Project

1. There is a class project (30% of the numerical grade)
2. One-person, or multiple-people projects (indicate who did what)
3. **Time line**: 1-page proposal by (roughly) Mar 1;
   full report due on early May
4. **Option 1**: Apply optimization to practical problems
   Examples: recommendation system, object detection, beamforming design, reinforcement learning, GAN.
   I may suggest some ideas.
5. **Option 2**: study of optimization algorithms. It doesn't need to involve rigorous proofs, but also not pure applications.
   - How is AdaGrad compared to gradient descent?
   - When is cyclic coordinate descent slower than randomized coordinate descent?
   - When does Lagrangian multiplier method diverge?
6. **Option 3**: work on a theoretical question

Academic Misconduct

1. There is zero tolerance on academic misconduct. Individuals suspected of committing academic dishonesty will be reported to the university. Penalty for academic misconduct (up to 100%).
2. Collaboration is encouraged, but not copying each others' homeworks

Outline

1. Introduction: What is Optimization
2. Course Introduction
3. Examples in Machine Learning
4. Mathematical Review

Key questions to keep in mind

- How to set up an optimization problem?
  - What are the optimization variables here?
  - What is the optimization objective?
- How to analyze the formulation? (optimal solutions, etc.)
- How to solve the resulting problem?

Toy Example

Solve equation $aw = 3$.

Optimization problem: $\min_{w \in \mathbb{R}} (aw - 3)^2$.

Solve a system of equations $Aw = b$.

Optimization problem: $\min_{w \in \mathbb{R}^n} \|Aw - b\|^2$.

Example 1: Regression Problem

1. **Training data sets** (n data points, $\mathbf{a}_i$ is the feature, and b_i is the label)

$$\{\mathbf{a}_i, b_i\}_{i=1,\cdots,n}, \ \mathbf{a}_i \in \mathbb{R}^d, \ b_i \in \mathbb{R}$$

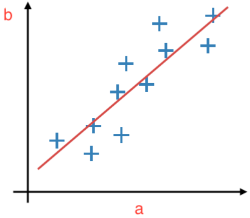
2. **Objective**: Learn a predictive function, characterized by $\mathbf{x} \in \mathbb{R}^d, x_0 \in \mathbb{R}$,

$$f(\mathbf{a}; \mathbf{x}) = \mathbf{x}^T \mathbf{a} + x_0 = \tilde{\mathbf{x}}^T \tilde{\mathbf{a}}, \quad \text{with} \quad \tilde{\mathbf{x}} := [\mathbf{x}, x_0], \tilde{\mathbf{a}} := [\mathbf{a}, 1]$$

3. We have absorbed x_0 in $\mathbf{x}$ and augmenting $\mathbf{a}_i$'s with extra 1
4. Use the cost function

$$\text{Loss}(\mathbf{x}) = \underbrace{\frac{1}{2} \sum_{i=1}^{n} (\tilde{\mathbf{x}}^T \tilde{\mathbf{a}}_i - b_i)^2}_{\text{squared loss}} = \|\mathbf{A}^T \tilde{\mathbf{x}} - \mathbf{b}\|^2$$

Minimize w.r.t. $\tilde{\mathbf{x}}$: $\tilde{\mathbf{x}}^* = (\mathbf{A}\mathbf{A}^T)^{-1} \mathbf{A}\mathbf{b}$ (closed-form solution!)



Example 1: Regression Problem

- It is the basic regression model; the predicted quantities are numerical
- A lot of aspects (including modeling and algorithm) can be improved when the data matrix $\mathbf{A}$ becomes "large"

Example 1: Regression Problem

- What if we also want to select a few key features that are most important?
  LASSO.

- What if the data sets are arriving sequentially?
  online optimization/learning.

- What if the data sets are distributed at different locations?
  distributed optimization.

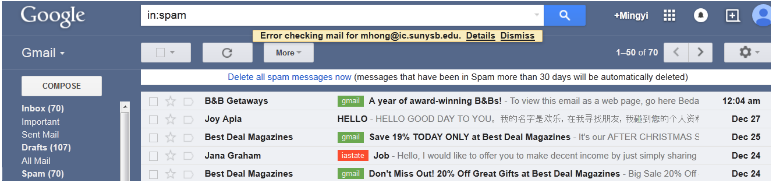
- What if the dimension of the variable is huge ($\mathbf{x}$ very long, lots of features), while the data is scarce (n is small)?
  Large-scale optimization.

- Why we are using the ℓ_2 loss? any other loss function?

- What if the relationship is nonlinear?
  neural networks, nonlinear least squares

Example 2: Classification

- Every email services have automatic spam detection mechanisms
- The basic task is the following
  1. Given an excerpt of an email, represented by a vector $\mathbf{a}_i$, where the elements can be words from the email
  2. Ask the question whether it is a spam or not ($b_i = +1, -1$)

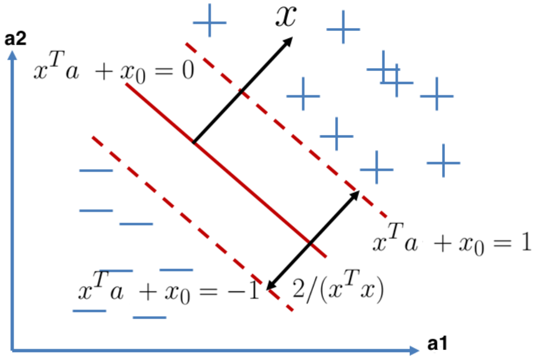


Example 2: Classification (cont.)

- **Training Stage**: data set $\{\mathbf{a}_i, b_i\}_{i=1}^n$, $\mathbf{a}_i \in \mathbb{R}^d$, $b_i \in \{-1, +1\}$
- $b_i = +1$ (spam); $b_i = -1$ (regular)
- **Objective**: learn the classification function
  $f(\mathbf{a}; \mathbf{x}) = \text{sgn}(\mathbf{x}^T\mathbf{a} + x_0)$, where sgn is the "sign function"
- **Testing stage**
  1. Given a feature vector $\widehat{\mathbf{a}}$
  2. Plug in $f(\mathbf{a}; \mathbf{x})$
  3. Classify based on the sign (i.e., "+" as spam, "-" as regular)
- Suppose the training data are linearly separable (?)
- **Task 1**: Select two hyperplanes to separate the data points
- **Task 2**: Try to maximize their distance
- What's a good formulation?

Example 2: Classification (cont.)

- **Intuition**: Find the separating plane that is far away from both classes



Example 2: Classification (cont.)

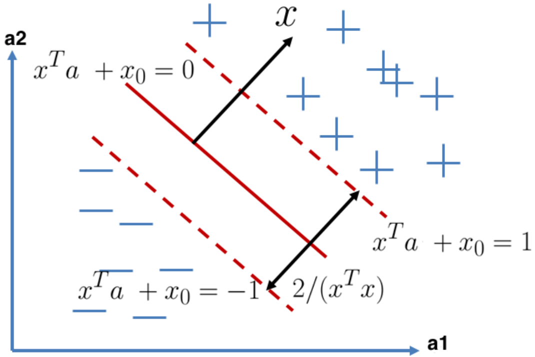
- **Formulation**: Consider the following problem

$$\begin{aligned} \min_{\mathbf{x}} \quad & \mathbf{x}^T\mathbf{x} \\ \text{subject to} \quad & b_i(\mathbf{x}^T\mathbf{a}_i + x_0) \geq 1, \ \forall\, i \end{aligned}$$

- One data point, one constraint
- The blue part says there is no classification error
- If $b_i = 1$ (is a spam), then $(\mathbf{x}^T\mathbf{a}_i + x_0) \geq 1$
- If $b_i = -1$ (regular), then $(\mathbf{x}^T\mathbf{a}_i + x_0) \leq 1$

Example 2: Classification (cont.)

- **Intuition**: Find the separating plane that is far away from both classes

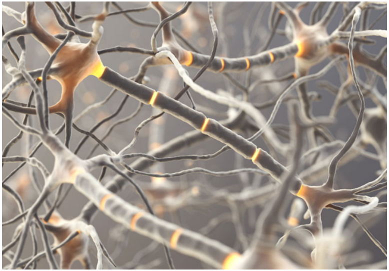


Example 2: Classification (cont.)

- This is one instance of the support vector machine (SVM) problem
- How about non-separable cases?
- How to solve the underlying optimization problem?
- Directly or need some re-formulation?
- What if the decision boundary is nonlinear?
- Data comes in sequentially?
- ....

Example 3: Training Neural Networks

- Neural Networks, especially Deep Neural Networks (DNN) become increasingly popular for various machine learning tasks
- Lots of high-profile applications: "Google Brain" by Google, 2012
- $16,000$ computer processors and 100 million images, $20,000$ distinct items, look for cats; research paper can be found here
- A nice article about the history of DL and NN on here



Example 3: Training Neural Network

Finding nonlinear classification boundary?

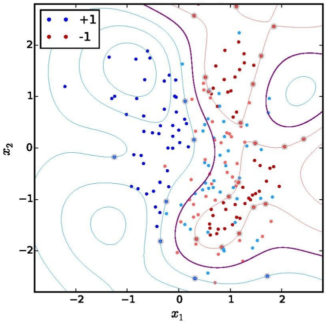
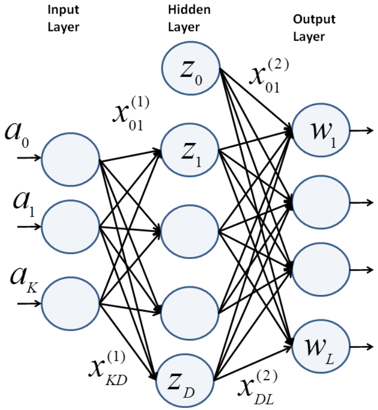


Figure: Nonlinear Classification [Wikipedia]

Example 3: Training Neural Network

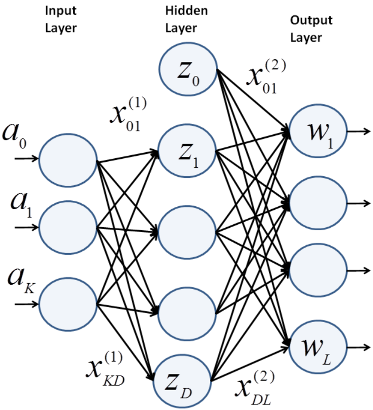
- **Input layer**: Each input one data point, K-dimensional: $a_0, \cdots, a_k$
- Note: Here we have a single data point, a_k denotes its kth element



Example 3: Training Neural Network

- **Hidden layer**: Each node, a nonlinear function g (tanh, sigmoid)
- Nonlinearly transform the inputs to outputs

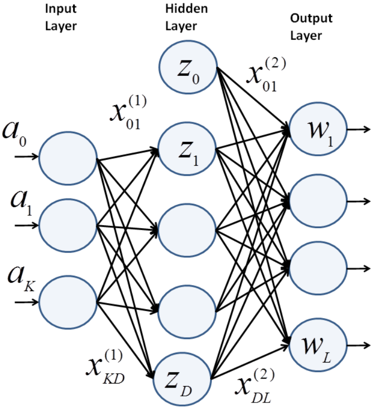
$$z_d = g\left(\sum_{k=1}^{K} a_k x_{k,d}^{(1)}\right)$$



Example 3: Training Neural Network

- **Output layer**: Each output node, a nonlinear function (sigmoid)

$$w_\ell = h\left(\sum_{d=0}^{D} z_d x_{d,\ell}^{(2)}\right), \quad \text{if binary classification, } L = 1$$



Example 3: Training Neural Network

- What are the optimization variables here?
- How to setup the optimization problem?
- How to solve the resulting problem?

Example 3: Training Neural Network

- What are the optimization variables here?
- How to setup the optimization problem?
- How to solve the resulting problem?

Example 3: Training Neural Network

- What are the optimization variables here?
- How to setup the optimization problem?
- How to solve the resulting problem?

Outline

1. Introduction: What is Optimization
2. Course Introduction
3. Examples in Machine Learning
4. Mathematical Review

Overview

1. Notations: Sets, functions, derivatives, gradients
2. Vectors, matrices
3. Norms, sequences, limits, continuity
4. Mean value theorems
5. Implicit function theorem
6. Contraction mappings
7. Reference Appendix A, B of the textbook
8. Get yourself familiar with them

Notations

1. **Sets**: $X,\ x \in X,\ X_1 \cup X_2,\ X_1 \cap X_2$
2. **Inf and Sup**:
The supremum of a nonempty set $X \subset \mathbb{R}$ is the smallest scalar y:
$$y \geq x,\ \forall\, x \in X$$
The infimum of a nonempty set $X \subset \mathbb{R}$ is the largest scalar y:
$$y \leq x,\ \forall\, x \in X$$
If $\sup X \in X$ (or, $\inf \in X$), then we say $\sup X = \max X$ (or, $\inf X = \min X$).
$$\sup\{1/n \mid n \geq 1\} =?, \quad \inf\{x \in \mathbb{R} \mid 0 < x < 1\} =?$$
3. **Function**:
$$f : X \to Y,\ X \text{ is called the ____} \qquad Y \text{ is called the ____}$$
Monotonicity $\longrightarrow$ Inverse function f^{-1} exists

Vectors

1. **Vector**: a vector $\mathbf{x} = [x_1, \cdots, x_n]$ is a column of scalars
2. **Linear combination**: if $\mathbf{x} = [x_1, \cdots, x_n]$ and $\mathbf{y} = [y_1, \cdots, y_n]$, then the linear combination is given by

$$\alpha\mathbf{x} + \beta\mathbf{y} = (\alpha x_1 + \beta y_1, \alpha x_2 + \beta y_2, \cdots, \alpha x_n + \beta y_n)$$

3. **Inner product**: $\langle \mathbf{x}, \mathbf{y} \rangle = \mathbf{x}'\mathbf{y} = \sum_{i=1}^{n} x_i y_i$

   Question: when is inner product positive, negative, zero?

   Orthogonality: $\mathbf{x} \perp \mathbf{y}$ iff (if and only if) ____

4. **Linearly Independent**: A set of vectors $\{\mathbf{x}^1, \cdots, \mathbf{x}^r\}$ are linearly independent if there does not exist a $(\alpha_1, \cdots, \alpha_r) \neq 0$ s.t.

____

Vectors

1. Basis and dimension of a subspace:
2. Orthogonal complement of a subspace S:

$$S^{\perp} := \{\mathbf{x} \mid \langle \mathbf{x}, \mathbf{y} \rangle = 0, \ \forall \ \mathbf{y} \in S\}$$

3. **Vector norms**: A norm $\|\mathbf{x}\|$ on $\mathbb{R}^n$ that assigns a scalar $\|\mathbf{x}\|$ to every $\mathbf{x} \in \mathbb{R}^n$ that satisfying
   1. $\|\mathbf{x}\| \geq 0$ for all $\mathbf{x}$
   2. $\|c\mathbf{x}\| = |c|\|\mathbf{x}\|$ for all $c \in \mathbb{R}$ and all $\mathbf{x}$
   3. $\|\mathbf{x}\| = 0$ iff $\mathbf{x} = 0$
   4. $\|\mathbf{x} + \mathbf{y}\| \leq \|\mathbf{x}\| + \|\mathbf{y}\|$ for all $\mathbf{x}, \mathbf{y}$
4. Measure some kind of “distance”

Vectors

1. **Common norms**

   Euclidean Norm : $\|\mathbf{x}\|_2 = (\mathbf{x}'\mathbf{x})^{1/2} =$

   ℓ_p Norm : $\|\mathbf{x}\|_p =$ for some $p \geq 1$

   ℓ_1 Norm : $\|\mathbf{x}\|_1 =$

   ℓ_∞ Norm : $\|\mathbf{x}\|_\infty =$

2. **Cauchy-Schwartz inequality**: $\langle \mathbf{x}, \mathbf{y} \rangle \leq \|\mathbf{x}\|_2 \|\mathbf{y}\|_2$
   related: $\langle \mathbf{x}, \mathbf{y} \rangle \leq \|\mathbf{x}\|_1 \|\mathbf{y}\|_\infty$

Cauchy-Schwartz inequality

An important inequality about the inner product of two vectors is the Cauchy-Swartz inequality

1. Characterizes the inner product of two vectors with their norms
2. Given two vectors $\mathbf{x}$ and $\mathbf{y}$ of the same size, we have

$$\langle \mathbf{x}, \mathbf{y} \rangle \leq \|\mathbf{x}\|_2 \|\mathbf{y}\|_2$$

3. Why this is true? Geometrically?
4. Useful fact about inner product

$$\langle \mathbf{x}, \mathbf{y} \rangle = \cos(\theta) \|\mathbf{x}\|_2 \|\mathbf{y}\|_2$$

   where θ is the angle between $\mathbf{x}$ and $\mathbf{y}$

Matrices

1. For any matrix $\mathbf{A}$, we use a_{ij} (or A_{ij}) to denote its (i,j)th entry.
2. Matrix addition, multiplication, transpose, symmetirx matrices $\mathbf{A} = \mathbf{A}'$.
$$[\mathbf{AB}]' = \mathbf{B}'\mathbf{A}', \ \mathbf{AB} \neq \mathbf{BA}$$
3. Let $\mathbf{A}$ be a $m \times n$ matrix.
   - Range of $\mathbf{A}$: $R(\mathbf{A}) = \{\mathbf{Ax}, \mid \mathbf{x} \in \mathbb{R}^n\}$;
   - Null space of $\mathbf{A}$: $N(\mathbf{A}) = \{\mathbf{x} \mid \mathbf{Ax} = 0\}$
   - Rank of $\mathbf{A}$ $\mathrm{Rank}(\mathbf{A})$. Full rank matrix $\mathbf{A}$: $\mathrm{Rank}(\mathbf{A}) = \min\{m, n\}$.
4. **Inner product**:
$$\langle \mathbf{A}, \mathbf{B} \rangle = \mathrm{Tr}(\mathbf{AB}') = \sum_{i,j} A_{ij} B_{ij}$$
where the trace operate is given by
$$\mathrm{Tr}[\mathbf{A}] = \sum_{i=1}^{n} A_{ii}$$

Square Matrices

1. Square matrix $(m = n)$; Identity matrix $\mathbf{I}$
2. Determinant $\det(\mathbf{A})$, inverse $\mathbf{A}^{-1}$. $\mathbf{A}^{-1}$ exists iff $\det(\mathbf{A}) \neq 0$
3. Useful identities: $\det(\mathbf{A}) = \det(\mathbf{A}')$
4. Orthogonal matrices: $\mathbf{A}\mathbf{A}' = \mathbf{I}$
5. (Complex) Eigenvalue λ: $\mathbf{A}\mathbf{x} = \lambda\mathbf{x}$ for some $\mathbf{x} \neq 0$
6. Spectral radius: $\rho(\mathbf{A}) = \max_i\{|\lambda_i|\}$, λ_i is an eigenvalue of $\mathbf{A}$.

Square Matrices

1. Eigen-decomposition of a symmetric matrix:

$$\mathbf{A} = \mathbf{P}'\mathbf{\Lambda}\mathbf{P}$$

   where $\mathbf{P}$ is orthonormal ($\mathbf{P}'\mathbf{P} = \mathbf{I}$), $\mathbf{\Lambda}$ is diagonal and real.

2. Positive (semi-) definite matrix: $\mathbf{A} \succeq 0$

$$\mathbf{x}^T\mathbf{A}\mathbf{x} \geq 0, \ \forall\ \mathbf{x} \qquad (2)$$

3. Property: $\mathbf{A} \succeq 0,\ \mathbf{B} \succeq 0 \rightarrow \mathbf{A} + \mathbf{B} \succeq 0$; $\mathbf{A} \succeq \mathbf{B} \rightarrow \mathbf{A} - \mathbf{B} \succeq 0$
   All eigenvalue of $\mathbf{A}$ are non-negative
4. Square root $\mathbf{A}^{1/2}$: $\mathbf{A}^{1/2} := \mathbf{P}\sqrt{\mathbf{\Lambda}}\mathbf{P}$, where $\mathbf{A} = \mathbf{P}\mathbf{\Lambda}\mathbf{P}'$ is the eigen-decomposition of $\mathbf{A} \succeq 0$.

   A "generalization" of positive number for the scalars

Single Value Decomposition

1. Single value decomposition (SVD):

$$\mathbf{A} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}' \in \mathbb{R}^{m\times n}, \tag{3}$$

   - $\mathbf{\Sigma} \in \mathbb{R}^{m\times n}$ rectangular diagonal matrix with diagonals $\sigma_1 \geq \sigma_2 \geq \cdots, \sigma_n \geq 0$;
   - $\mathbf{U} \in \mathbb{R}^{m\times m}$, $\mathbf{V} \in \mathbb{R}^{n\times n}$ orthonormal

2. Relationship of SVD and ED: σ_i^2 is an eigenvalue of $\mathbf{A}\mathbf{A}'$ (Why?)

$$\mathbf{A}\mathbf{A}' = (\mathbf{U}\mathbf{\Sigma}\mathbf{V}')(\mathbf{V}\mathbf{\Sigma}\mathbf{U}') = \mathbf{U}\mathbf{\Sigma}^2\mathbf{U}'$$

3. Condition number: $\kappa(\mathbf{A}) = \sigma_1/\sigma_n$
   very important for optimization!!

Matrices and Norms

1. Norms:

$$\text{Frobenious Norm}: \|\mathbf{A}\|_F = \left(\sum_{i,j} |A_{ij}|^2\right)^{1/2} = \left(\sum_i \sigma_i^2\right)^{1/2} \tag{4}$$

$$\text{Nuclear Norm}: \|\mathbf{A}\|_* = \sum_i \sigma_i \tag{5}$$

$$\text{Matrix 2-norm (spectral norm)}: \|\mathbf{A}\|_2 = \sup_{\mathbf{x}\neq 0} \frac{\|\mathbf{A}\mathbf{x}\|_2}{\|\mathbf{x}\|_2} = \max_i \sigma_i \tag{6}$$

2. Useful inequalities:

$$\|\mathbf{A}\mathbf{x}\|_2 \leq \|\mathbf{A}\|_2 \|\mathbf{x}\|_2, \ \|\mathbf{A}\|_* \geq \|\mathbf{A}\|_F \geq \|\mathbf{A}\|_2$$

Derivatives

Suppose $f : \mathbb{R}^n \to \mathbb{R}$ is a continuously twice differentiable function.

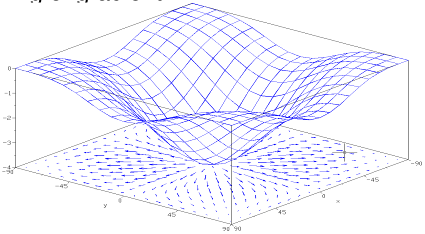
1. Partial derivative (where $\mathbf{e}_i$ is the ith unit vector of $\mathbb{R}^n$)

$$\frac{\partial f(\mathbf{x})}{\partial x_i} := \lim_{t \to 0} \frac{f(\mathbf{x} + t\mathbf{e}_i) - f(\mathbf{x})}{t}$$

2. Gradient vector (a column vector) [example: $f(\mathbf{x}) = \mathbf{a}^T\mathbf{x} - \mathbf{b}$]:

$$\nabla f(\mathbf{x}) = \left(\frac{\partial f(\mathbf{x})}{\partial x_1}, \cdots, \frac{\partial f(\mathbf{x})}{\partial x_n} \right)$$

Physical meaning of gradient?



Wikipedia: "the gradient points in the direction of the greatest rate of increase of the function, and its magnitude is the slope of the graph in that direction"

Derivatives

Suppose $f : \mathbb{R}^n \to \mathbb{R}$ is a continuously twice differentiable function.

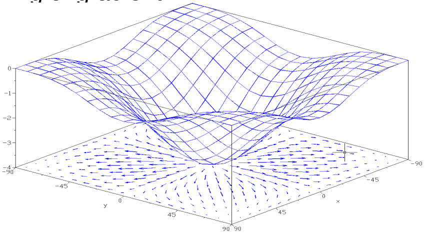
1. Partial derivative (where $\mathbf{e}_i$ is the ith unit vector of $\mathbb{R}^n$)

$$\frac{\partial f(\mathbf{x})}{\partial x_i} := \lim_{t \to 0} \frac{f(\mathbf{x} + t\mathbf{e}_i) - f(\mathbf{x})}{t}$$

2. Gradient vector (a column vector) [example: $f(\mathbf{x}) = \mathbf{a}^T\mathbf{x} - \mathbf{b}$]:

$$\nabla f(\mathbf{x}) = \left(\frac{\partial f(\mathbf{x})}{\partial x_1}, \cdots, \frac{\partial f(\mathbf{x})}{\partial x_n} \right)$$

Physical meaning of gradient?



Wikipedia: “the gradient points in the direction of the greatest rate of increase of the function, and its magnitude is the slope of the graph in that direction”

Derivatives

Suppose $f : \mathbb{R}^n \to \mathbb{R}$ is a continuously twice differentiable function.

1. Hessian matrix:

$$\nabla^2 f = \left[\frac{\partial f(\mathbf{x})}{\partial x_i \partial x_j} \right] \in \mathbb{R}^{n \times n}$$

2. Taylor expansion:

$$f(\mathbf{y}) - f(\mathbf{x}) = \nabla f(\mathbf{x})'(\mathbf{y} - \mathbf{x}) + \frac{1}{2}(\mathbf{x} - \mathbf{y})' \nabla^2 f(\mathbf{x})(\mathbf{x} - \mathbf{y}) + o(\|\mathbf{x} - \mathbf{y}\|^2)$$

Practice:
Q1: How to compute the Hessian of $f(x) = x^T A x$?

Q2: How to compute the Hessian of $f(x,y) = \|M - xy^T\|_F^2$, where $x, y \in \mathbb{R}^n$?

Derivatives: Chain Rule

1. Scalar case: suppose $f, g : \mathbb{R} \to \mathbb{R}$ are functions, and $f'(x)$ and $g'(f(x))$ exist, then $h(x) \triangleq g(f(x))$ satisfies

$$h'(x) = g'(f(x))f'(x).$$

Example: $f(x) = \sin x$, $g(y) = y^2$, $h(x) = (\sin x)^2$, then

2. Vector case: $f : \mathbb{R}^k \to \mathbb{R}^m$ and g: $\mathbb{R}^m \to \mathbb{R}^n$, and $f'(x)$ and $g'(f(x))$ exist. Let $h(\mathbf{x}) = g(f(\mathbf{x}))$ (function composition), then

$$\nabla h(\mathbf{x}) = \nabla f(\mathbf{x}) \nabla g(f(\mathbf{x}))$$

Example: $g(z) = z^2/2$, $f(\mathbf{x}) = \mathbf{A}\mathbf{x} - \mathbf{b}$, then

$$\nabla \left(\frac{1}{2} \|\mathbf{A}\mathbf{x} - \mathbf{b}\|^2 \right) = \mathbf{A}'(\mathbf{A}\mathbf{x} - \mathbf{b}), \ \nabla^2 f(\mathbf{A}\mathbf{x}) = \mathbf{A}'\mathbf{A}$$

Contraction Mapping

1. **Lipschitzian Property**: f: $\mathbb{R}^n \to \mathbb{R}^m$ satisfies

   $$\|f(\mathbf{x}) - f(\mathbf{y})\| \leq \gamma \|\mathbf{x} - \mathbf{y}\|, \ \forall \ \mathbf{x}, \mathbf{y}$$

   γ is called the Lipschitz constant; the function is called γ-Lipschitz continuous

   Example 1: $f(x) = 2x$ is 2-Lipschitz continuous;

   Example 2: What about $f(x) = Ax$?

   Example 3: What about $f(x) = x^2$?

2. If $\gamma \leq 1$, then f is called a non-expansive mapping

3. If $\gamma < 1$, then f is called a contraction mapping

   Example 1: $f(x) = 2x$ is not a contraction mapping; $f(x) = 0.5x$ is.

   Example 2: $f(x) = Ax$ is a contraction mapping iff ?

Fixed Point Theorem

1. **Fixed point theorem**: If f is a contraction mapping, then the iterated function sequence

$$\mathbf{x}, f(\mathbf{x}), f(f(\mathbf{x})), \cdots,$$

converges to a unique fixed point $\mathbf{x}^*$ (independent of $\mathbf{x}$) satisfying

$$\mathbf{x}^* = f(\mathbf{x}^*).$$

Homework

1. Getting yourself familiar with mathematical backgrounds
2. Reading: Appendix A of the textbook.