MSc on Intelligent Critical Infrastructure Systems

Machine Learning

Lecture 1

Marios Polycarpou

Director, KIOS Research and Innovation Center of Excellence

Professor, Electrical and Computer Engineering

University of Cyprus

funded by:

Fourth Industrial Revolution

- What is a technological revolution?
- 1st Industrial Revolution (1760)
- 2nd Industrial Revolution (1900)
- 3rd Industrial Revolution (1960)
- 4th Industrial Revolution (2000) – term coined in 2015 by Klaus Schwab, executive director of the World Economic Forum.

→ Combines hardware, software, and biology (cyber-physical systems), and emphasizes advances in communication and connectivity. It is expected to be marked by breakthroughs in fields such as machine learning, robotics, nanotechnology, biotechnology, the internet of things (IoT), wireless technologies (5G), 3D printing and fully autonomous vehicles.

Motivation for Machine Learning

- Sensor technology
  - wealth of sensors
  - new generation sensors

Pebble
Smart Watch

Google
Smart Glasses

Sonitus
in the Mouth
Hearing Aid

- Information & Communication Technology (ICT)
  - store, process and transmit data **collected by sensors**

Motivation for Machine Learning

- Internet of Things (IoT)
  - **sensor enabled devices** connected to the internet and able to communicate with each other

- Big Data
  - **sensor technology** and ICT enabled the collection of extremely large data sets
  - contribute to the better perception of complex systems

Motivation for Machine Learning

- Image processing
- Speech recognition
- Prediction in financial services
- Finding patterns in marketing and sales
- Improving healthcare
- Automation – no human intervention needed
- Continuous improvement
- Wide applications

How is Machine Learning related to Intelligent Critical Infrastructure Systems?

Critical Infrastructure Systems (CIS)

Heterogeneous - Interdependent - Interconnected

Risks, Faults, Attacks

Information Communication Technologies (ICT)

Sensors, Actuators

Big Data, Internet of Things

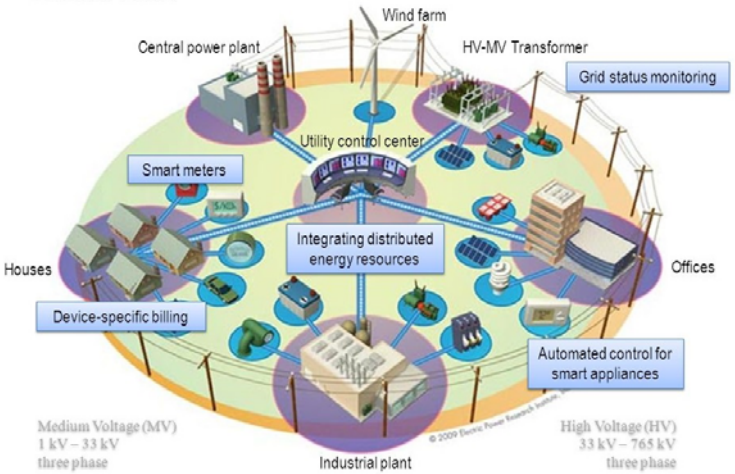
Intelligent Systems and Networks

Monitoring, Control

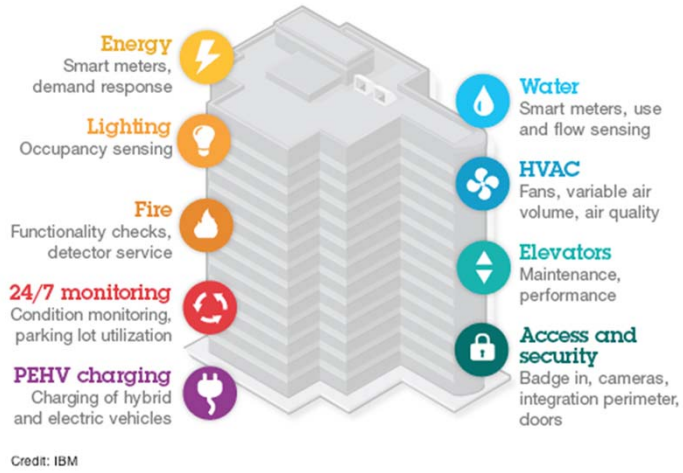
Management, Security

Big Data → Smart Decisions

How is Machine Learning related to Intelligent Critical Infrastructure Systems?



Smart Grids



Smart Buildings

How is Machine Learning related to Intelligent Critical Infrastructure Systems?

Water Systems

Revenue & Water Losses → Water Quality → Energy Consumption → Safety & Security

Water loss: seven things you need to know about an invisible global problem

A staggering 46bn litres of drinking water are lost globally every day. What can consumers, business and governments do?

▲ Iraqis fill drinking water and wash clothes at a broken water pipeline in a Shia district of Sadr City, Baghdad. Photograph: Karim Kadim/AP

China water contamination affects 2.4m after oil leak

12 April 2014 Share

Queuing up to buy bottled water on Friday - shops are now reported to have sold out

Guardian sustainable business

Solutions to the water energy nexus remain elusive

Volkswagen and Coca-Cola among businesses acting on interconnected issues of water and energy, but the voice of power companies and energy producers is largely absent

Giulio Boccaletti

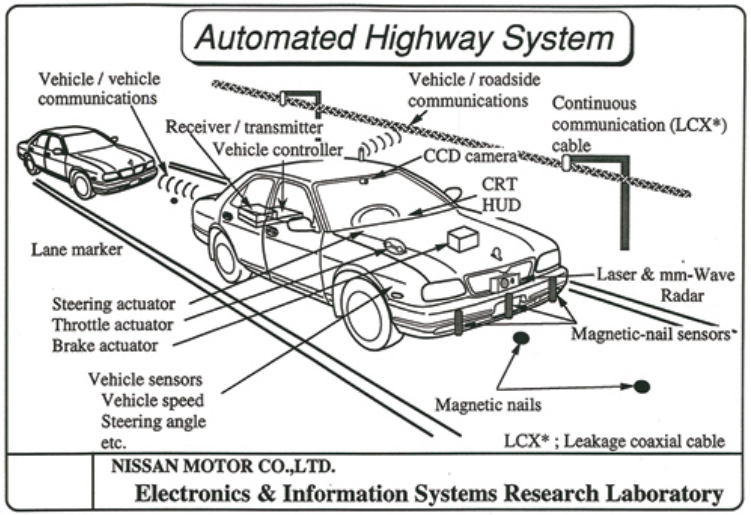
US news

Cyber-attack claims at US water facility

FBI and Homeland Security to investigate shutdown of a water pump suspected to be work of foreign hackers

▲ The US Department of Homeland Security and the FBI are to investigate claims that hackers retrieved control of a software at water utility shutting down its pump. Photograph: Jeff Gentner/AP

How is Machine Learning related to Intelligent Critical Infrastructure Systems?



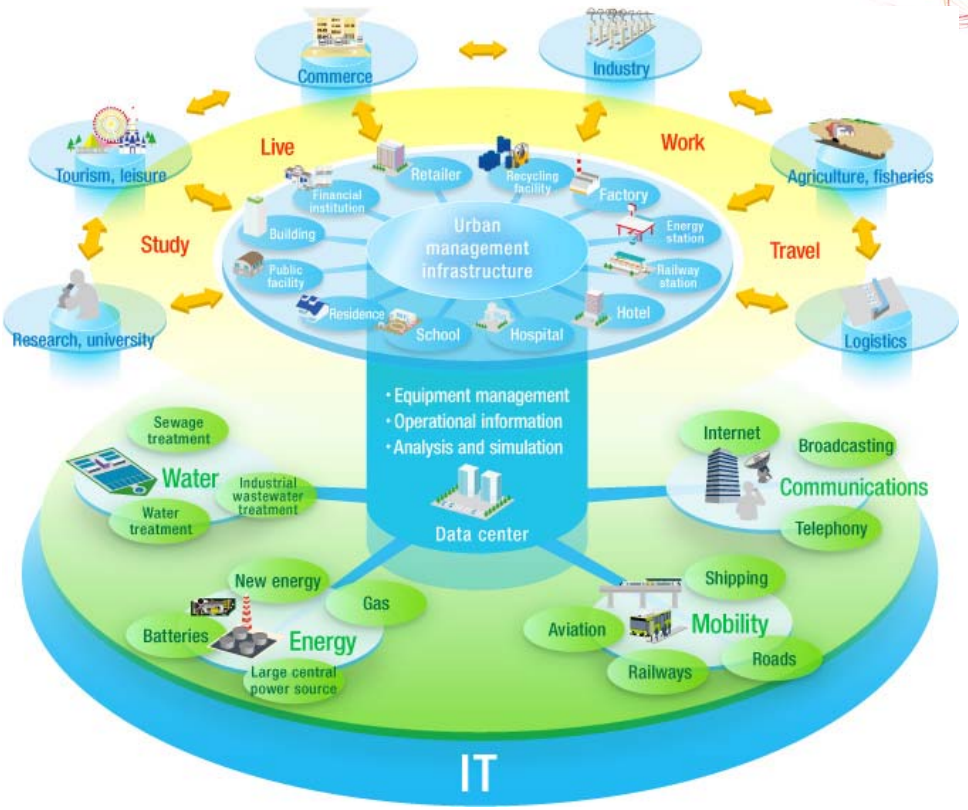
Intelligent Transportation

Multi-Agent Formation

How is Machine Learning related to Intelligent Critical Infrastructure Systems?

Smart Cities

- Physical Interconnections
- Cyber Interconnections
- Interdependencies



Defining Machine Learning

A machine learning algorithm is an algorithm that is able to learn from data. Other related terms: *data-driven methods*, *computational intelligence*, *artificial intelligence* (AI).

In general, Artificial Intelligence and Computational Intelligence are bigger concepts aimed at creating intelligent machines that can simulate human thinking capability and behaviour. Machine Learning is a subset of AI that allows machines to learn from data without being programmed explicitly.

Defining Machine Learning

A computer program is said to learn from experience ***E*** with respect to some class of tasks ***T*** and performance measure ***P***, if its performance at tasks in ***T***, as measured by ***P***, improves with experience ***E*** (*Mitchell, 1997*).

T.M. Mitchell, *Machine Learning*, McGraw-Hill, New York, 1997.

The Task *T*

- Classification
- Regression
- Monitoring and anomaly detection
- Transcription
- Speech recognition
- Machine translation
- Navigation and Control

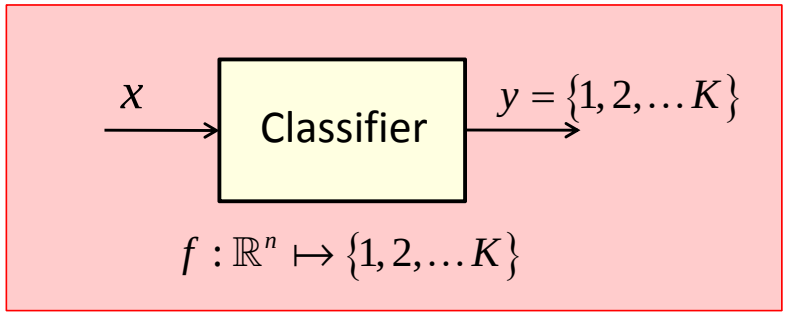
The Task *T* -- Classification

$$x = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ \vdots \\ x_n \end{bmatrix}$$

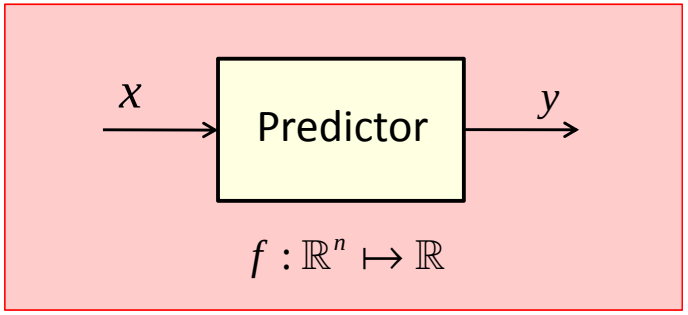
Features

Classes

$$y = \{1, 2, \ldots K\}$$



The Task T -- Regression



The Performance Measure *P*

- Error rate; error properties
- **Training** set; **Test** set
- Different performance measures may be used
- What are norms?

<u>Examples of norms</u> $(\varepsilon \in \mathbb{R}^n)$

$$\|\varepsilon\|_1 := \sum_{i=1}^{n} |\varepsilon_i|$$

(1-norm)

$$\|\varepsilon\|_2 := \left[\sum_{i=1}^{n} |\varepsilon_i|^2 \right]^{1/2}$$

(2-norm)
Euclidean norm

$$\|\varepsilon\|_\infty := \max_{1 \le i \le n} |\varepsilon_i|$$

(∞-norm)

The Experience *E*

What type of dataset is available?

Types of Learning

- Supervised Learning (there exist labelled examples)
- Unsupervised Learning (clustering, dimensionality reduction)
- Semi-Supervised Learning
- Reinforcement Learning

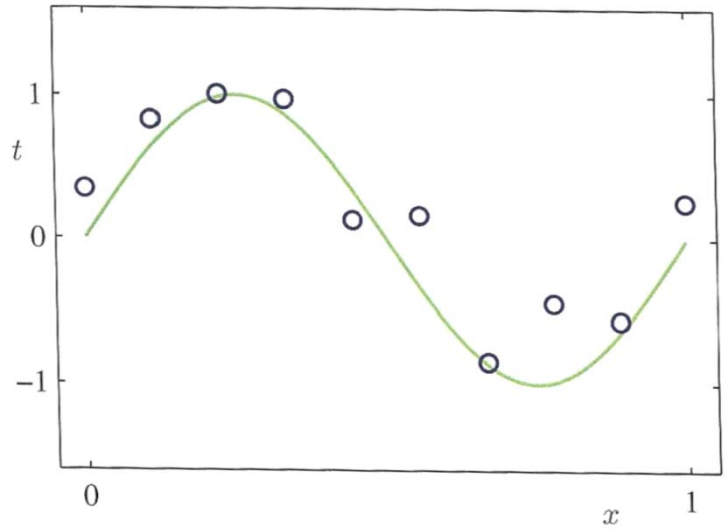
Machine Learning – Simple Example

Dataset:

$$f(x) = \sin(2\pi x)$$

$$y = f(x) + \varepsilon$$

$$x = (x_1, x_2, \ldots x_N) \qquad N = 10$$



Machine Learning – Simple Example

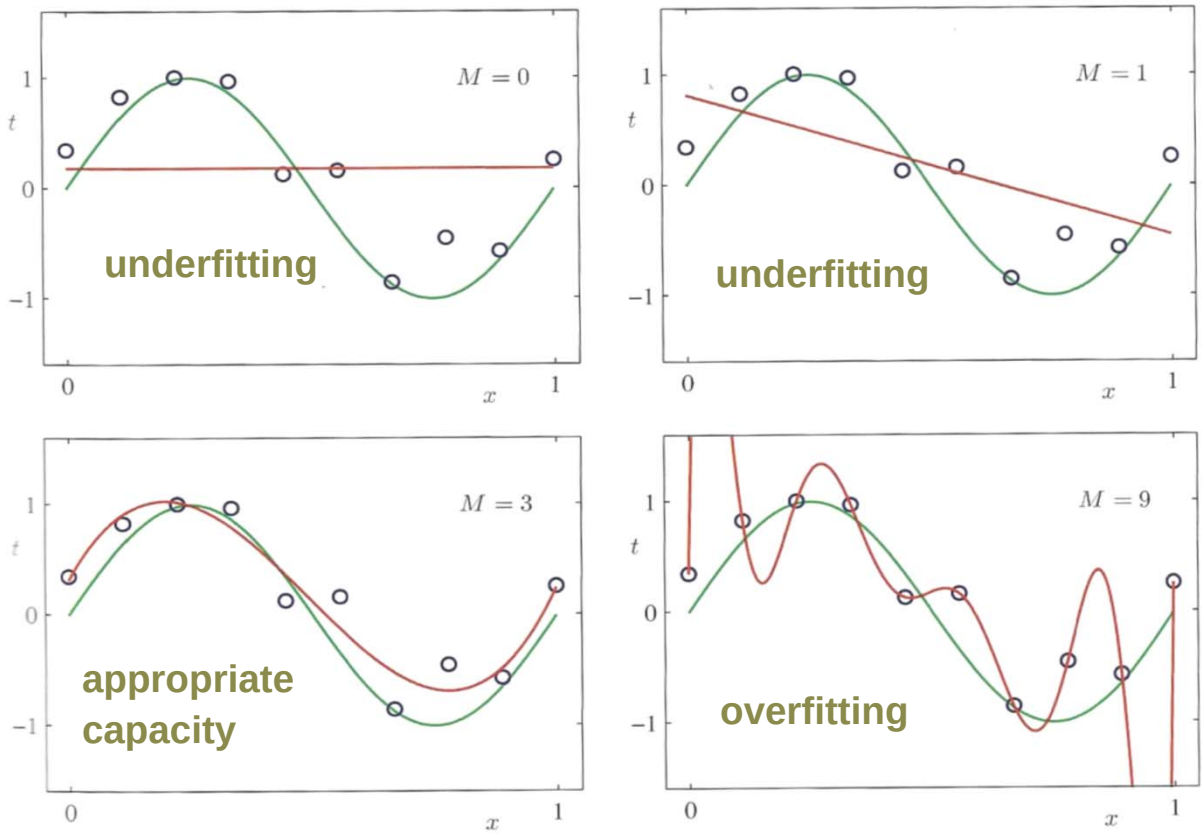
Adaptive Approximation Model: polynomial

$$\hat{f}(x;\theta) = \theta_0 + \theta_1 x + \theta_2 x^2 + \cdots + \theta_M x^M = \sum_{j=0}^{M} \theta_j x^j$$

Learning: minimizing a cost function (or error function)

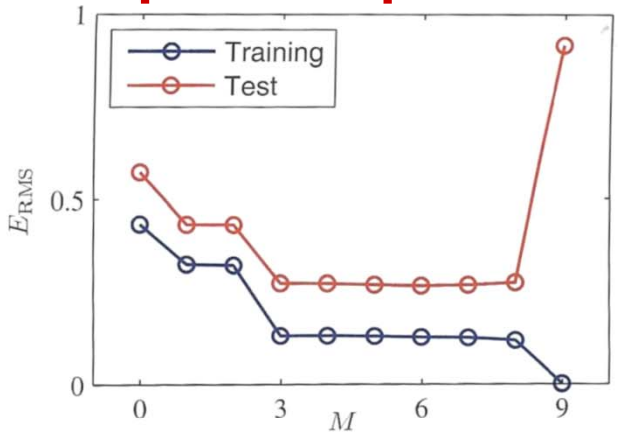
$$J(\theta) = \frac{1}{2}\sum_{n=1}^{N}\left\{\hat{f}(x_n;\theta) - y_n\right\}^2$$

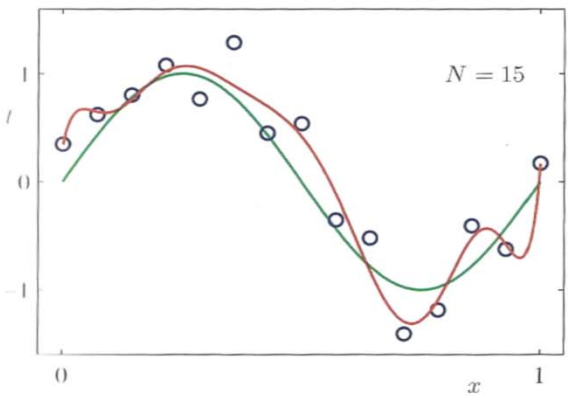
Machine Learning – Simple Example

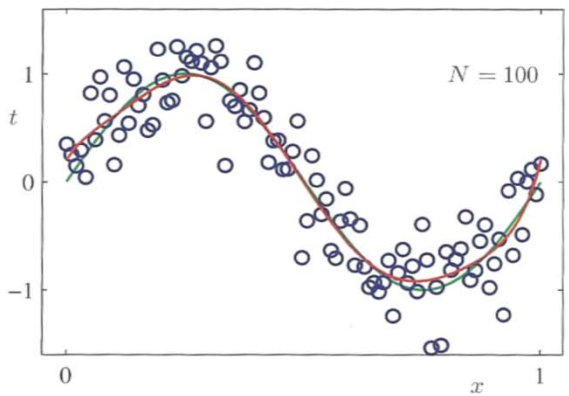


Machine Learning – Simple Example

underfitting and overfitting depends on both M and N







Capacity, Overfitting and Underfitting

- The ability to perform well on previously unobserved inputs is called ***generalization***. This is a key challenge in machine learning.
- Training set vs. test set. Typically, the generalization error (test error) is measuring the performance on a test set.
- The expected test error is greater than or equal to the expected training error

The performance of a machine learning algorithm is based on both:

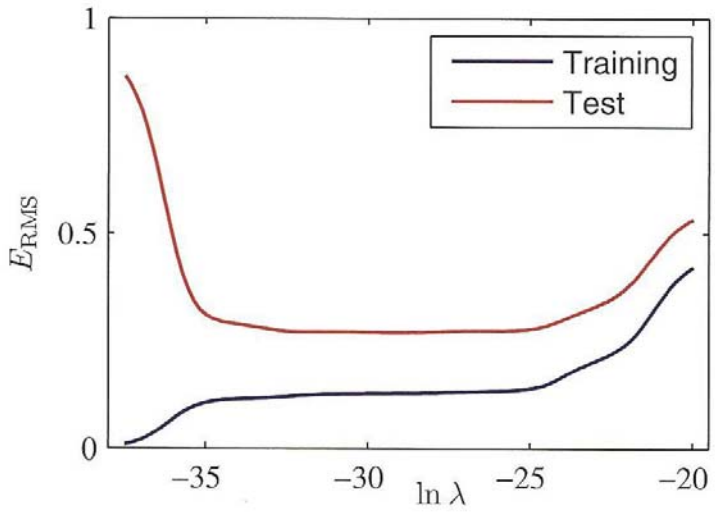
1. Making the training error small
2. Making the gap between the training and test error small

Capacity, Overfitting and Underfitting

- **Occam's Razor** – is a principle that states that among competing hypothesis that explain known observations equally well, we should choose the *simplest* one. Named after William of Ockham (1287-1347).
- **Statistical Learning Theory** provides various means of quantifying model capacity. The Vapnik-Chervonenkis dimension (VC dimension) is the most well known. However, it is not very practical with advanced machine learning algorithms

Regularizarion

$$J_R(\theta) = \frac{1}{2}\sum_{n=1}^{N}\left\{\hat{f}(x_n;\theta) - y_n\right\}^2 + \frac{\lambda}{2}\|\theta\|^2$$



Hyperparameters and Validation Sets

- **Hyperparameters** – parameters that affect the performance of the algorithm but they are not adapted by the learning algorithm itself; e.g., the degree of the polynomial is a capacity hyperparameter; the value of λ in regularization is another hyperparameter.
- **Validation Set** – is used to select the hyperparameters. Split the training set into two disjoint subsets; one subset of data is used to learn the parameters, the other subset (validation set) is used to estimate the generalization error in order to update the hyperparameters; e.g.; 80% of data is used for training and 20% for validation.
- Three types of data: **Training set, Validation set, Test set.**

Batch Learning vs. Online Learning

- **Batch Learning** – machine learning methods that are based on learning on the entire training data set.
- **Online Learning** – machine learning methods that are based on data becoming available in sequential order and is sued to update the parameters at each step. Data does not need to be stored after it is used to update the adjustable parameters
- Something in between batch learning and online learning is to use a window of data (not the entire data) to update the parameters.

Mathematical Foundations for Machine Learning

- Linear Algebra
- Optimization
- Approximation Theory
- Probability Theory
- Random Signals and Systems

Optimization

- Optimization is fundamental in multiple science and engineering fields
- Main idea is to take the derivative and set it to zero (but it soon becomes more complicating than that……)

Optimization

Optimization problem:

$F(x)$: Objective function

$\{c_i\}$: constraint functions

$$\begin{aligned} &\underset{x\in\mathbb{R}^n}{\text{minimize}}\ F(x) \\ &\text{subject to } c_i(x)=0 \quad i=1,2,\ldots,m' \\ &\qquad\qquad\quad c_i(x)\geq 0 \quad i=m',m'+1,\ldots,n \end{aligned}$$

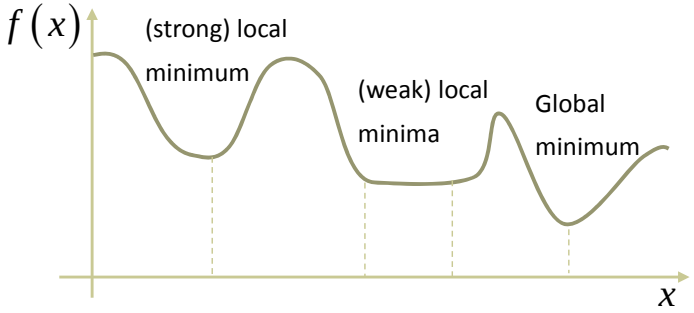
Some notation: $F:\mathbb{R}^n \mapsto \mathbb{R}$

Gradient of F : $\nabla F = \left[\frac{\partial F}{\partial x_1}, \frac{\partial F}{\partial x_2}, \ldots, \frac{\partial F}{\partial x_n}\right]^\top$

Hessian matrix:
($n\times n$ symmetric matrix)

$$\nabla^2 F = \frac{\partial^2 F}{\partial x_i \partial x_j} = \begin{bmatrix} \frac{\partial^2 F}{\partial x_1^2} & \frac{\partial^2 F}{\partial x_1 \partial x_2} & \cdots & \frac{\partial^2 F}{\partial x_1 \partial xn} \\ \vdots & & & \vdots \\ \frac{\partial^2 F}{\partial x_n \partial x_1} & \cdots & \cdots & \frac{\partial^2 F}{\partial x_n^2} \end{bmatrix}^\top$$

Optimization



<u>Necessary and sufficient conditions for unconstrained optimization</u>

$$\underset{x \in \mathbb{R}^1}{\text{minimize}} \; f(x)$$ (univariate case)

$$\begin{cases} f'(x^\star) = 0 \\ f''(x^\star) \geq 0 \end{cases}$$

<u>Necessary</u> conditions for $x^\star$ to be a minimum

$$\begin{cases} f'(x^\star) = 0 \\ f''(x^\star) > 0 \end{cases}$$

<u>Sufficient</u> conditions for $x^\star$ to be a minimum

Optimization

$$\underset{x \in \mathbb{R}^n}{\text{minmize}}\ F(x)$$

$$F(x) = F(x_1, x_2, \ldots, x_n) \quad \text{(multivariate case)}$$

<u>Necessary Conditions</u>

$$\begin{cases} \nabla F(x^\star) = 0 \\ \nabla^2 F(x^\star) \text{ is positive semi-definite} \end{cases}$$

<u>Sufficient Conditions</u>

$$\begin{cases} \nabla F(x^\star) = 0 \\ \nabla^2 F(x^\star) \text{ is positive definite} \end{cases}$$

Optimization

When is a matrix A positive (semi)-definite?

A is **positive-definite** $(A > 0)$ if $x^\top A x > 0$ for any non-zero x .

A is **positive-semidefinite** $(A \geq 0)$ if $x^\top A x \geq 0 \;\; \forall x.$

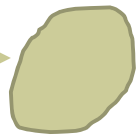
<u>Lemma</u>: $A > 0 \quad \Leftrightarrow \quad \lambda_i(A) > 0 \quad \forall i = 1, \ldots, n$

all the eigenvalues are positive

Optimization

Convexity:

The set S is convex if $\lambda x + (1-\lambda) y \in S \quad \forall x, y \in S, \ \forall \lambda \in [0,1]$.

convex 

Not convex 

A function $F : S \mapsto \mathbb{R}$ <u>is convex over S</u> if

$$F\left(\lambda x + (1-\lambda) y\right) \le \lambda F\left(x\right) + (1-\lambda) F\left(y\right) \qquad \forall x, y \in S, \ \forall \lambda \in [0,1]$$

<u>Lemma</u>: If $F \in C^1(-\infty, \infty)$ and F is convex over $\mathbb{R}^n$ then $x^\star$ is a **global minimum** of F iff $\nabla F\left(x^\star\right) = 0$.

Optimization

Example: Quadratic function $\underset{x\in\mathbb{R}^n}{\text{minimize}}\ c^\top x + \frac{1}{2}x^\top G x$

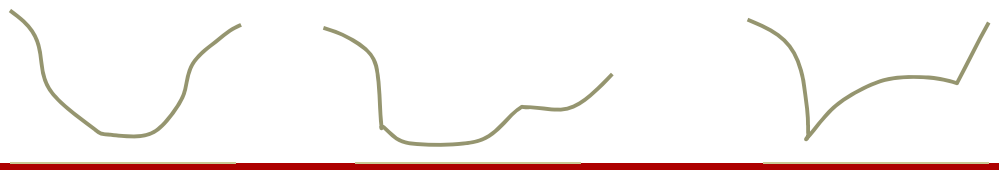
$c \in \mathbb{R}^n \quad G \in \mathbb{R}^{n\times n}$ symmetric

$$\nabla F(x) = c + Gx \ \Rightarrow\ \nabla F(x^\star) = 0 \ \Rightarrow\ \boxed{Gx^\star = -c}$$

$\nabla^2 F(x) = G$ $\quad x^\star$ is a minimum of $F(x)$ $\quad \Rightarrow \begin{cases} Gx^\star = -c \\ G \geq 0 \end{cases}$

Note: $F(x)$ is convex $\Rightarrow G \geq 0$

Warning: not all functions that we want to minimize are smooth

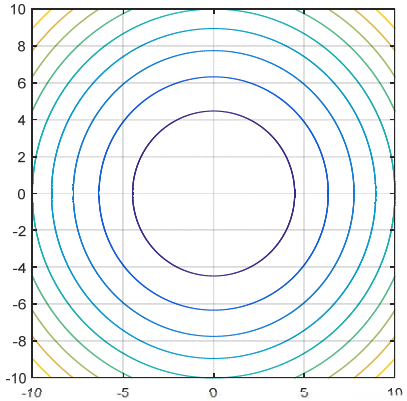


Optimization

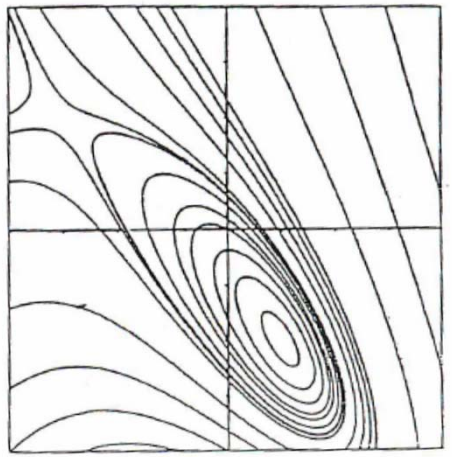
Contour plots:

$F(x) = \kappa$ for different values of κ .

- $F(x_1, x_2) = x_1^2 + x_2^2 = \kappa$



- $F(x_1, x_2) = e^{x_1}\left(4x_1^2 + 2x_2^2 + 4x_1x_2 + 2x_2 + 1\right)$



Optimization

- **<u>Linear Programming</u>:** (Simplex method)

Standard Form:
$$\begin{cases} \underset{x\in\mathbb{R}^n}{\text{minimize}} & c^\top x \\ \text{subject to} & Ax = b \\ & x \geq 0 \end{cases}$$
A, b, c are given.

- **<u>Unconstrained Minimization</u>:** (Steepest descent; Newton method)

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

- **<u>Constrained Minimization</u>**: (Gradient Projection; Penalty methods ; Lagrange methods)

$$\begin{cases} \underset{x\in\mathbb{R}^n}{\text{minimize}} & F(x) \\ \text{subject to} & c(x) = 0 \\ & d(x) \geq 0 \end{cases}$$

Optimization

Optimization Algorithms

What are the performance criteria?

- Convergence
- Rate of convergence
- Complexity

Optimization

Algorithms for Unconstrained Minimization $\min_{x \in \mathbb{R}^n} F(x)$

(A) Steepest Descent Method (Gradient Method)

Start $x_0 \to x_1 \to x_2 \to \ldots x_k \to \ldots$ vectors

$$\boxed{x_{k+1} = x_k - \lambda_k^\star \nabla F(x_k)} \qquad \lambda_k^\star > 0$$

$\lambda_k^\star$ is determined be a linear search so that x_{k+1} minimizes $F(x)$ in the direction $d_k = -\nabla F(x_k)$ from x_k.

Optimization

Trade-offs in choosing λ:

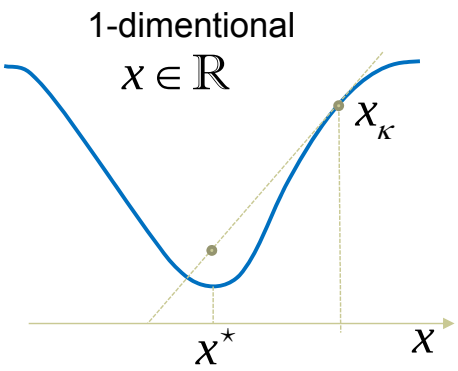
- speed
- convergence
- complexity

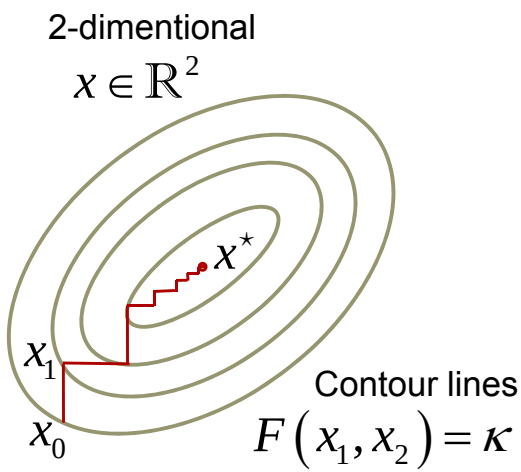
Simplest gradient descent:

$$x_{k+1} = x_k - \lambda \nabla F\left(x_k\right) \qquad \lambda = \text{const.}$$

Optimization

Intuition for Steepest Descent





Optimization

Example: $F(x) = \frac{1}{2}x^2$

$$\nabla F(x_k) = x_k$$

$$\boxed{x_{k+1} = x_k - \lambda x_k}$$

$$x_k = (1-\lambda)^k x_0$$

Example: $F(x_1, x_2) = e^{x_1}\left(4x_1^2 + 2x_2^2 + 4x_1x_2 + 2x_2 + 1\right)$

$$\nabla F(x_1, x_2) = \begin{bmatrix} e^{x_1}\left(4x_1^2 + 2x_2^2 + 4x_1x_2 + 8x_1 + 6x_2 + 1\right) \\ e^{x_1}\left(4x_1 + 4x_2 + 2\right) \end{bmatrix}$$

$$\boxed{\begin{bmatrix} x_{k+1} \end{bmatrix} = \begin{bmatrix} x_k \end{bmatrix} - \lambda_k^{\star} \begin{bmatrix} \nabla F(x_k) \end{bmatrix}}$$

Optimization

<u>In continuous-time:</u> $x_{k+1} = x_k - \lambda \nabla F(x_k) \Rightarrow \dfrac{x_{k+1} - x_k}{\lambda} = -\nabla F(x_k)$

as $\lambda \to 0$ $\quad \boxed{\dot{x} = -\nabla F(x)} \quad x(0) = x^0$

<u>After rescaling:</u> $\boxed{\dot{x} = -\Gamma \nabla F(x)} \quad \Gamma = \Gamma^\top > 0$ $\quad$ Γ is symmetric & positive definite

<u>Example:</u> $\min\limits_{x \in \mathbb{R}^n} x^\top G x = F(x)$

$$\nabla F(x) = Gx \Rightarrow \dot{x} = -Gx \; ; \; x(0) = x^0$$

$$\boxed{x(t) = e^{-Gt} x^0}$$

Optimization

(B) <u>Newton's Method</u>: $\min_{x \in \mathbb{R}^n} F(x)$ (F is convex)

$\nabla F(x^\star) = 0$ (necessary condition)

$$\nabla F(x) \approx \nabla F(x_k) + \underbrace{\frac{d}{dx} \nabla F(x)\Big|_{x=x_k}}_{\nabla^2 F(x_k)} (x - x_k)$$

Hessian

Let $H(x) := \nabla^2 F(x)$

$$\nabla F(x) \approx \nabla F(x_k) + H(x_k)(x - x_k) = 0 \text{ (we want)}$$

$$0 = \nabla F(x_{k+1}) = \nabla F(x_k) + H(x_k)(x_{k+1} - x_k)$$

$$\boxed{x_{k+1} = x_k - H(x_k)^{-1} \nabla F(x_k)}$$ (assuming $H(x_k)^{-1}$ exists)

Optimization

<u>In continuous-time:</u>

$$\boxed{\dot{x} = -\beta H(x)^{-1} \nabla F(x) \quad ; \quad \beta > 0}$$

<u>Example:</u> $F(x) = \frac{1}{2} x^2$

$$\nabla F(x) = x \qquad H(x) = 1 \quad H(x)^{-1} = 1$$

$$x_{k+1} = x_k - H(x_k)^{-1} \nabla F(x_k)$$

$$x_{k+1} = x_k - x_k = 0. \qquad x_0 \to x_1 = 0$$

Convergence in one step!

Optimization

Example: $F(x_1, x_2) = e^{x_1}\left(4x_1^2 + 2x_2^2 + 4x_1x_2 + 2x_2 + 1\right)$

$$\nabla F(x_1, x_2) = \begin{bmatrix} e^{x_1}\left(4x_1^2 + 2x_2^2 + 4x_1x_2 + 8x_1 + 6x_2 + 1\right) \\ e^{x_1}\left(4x_1 + 4x_2 + 2\right) \end{bmatrix}$$

$$H(x_1, x_2) = e^{x_1}\begin{bmatrix} 4x_1^2 + 2x_2^2 + 4x_1x_2 + 16x_1 + 10x_2 + 9 & 4x_1 + 4x_2 + 6 \\ 4x_1 + 4x_2 + 6 & 4 \end{bmatrix}$$

$$H(x_1, x_2)^{-1} = \frac{-e^{x_1}}{8(x_2^2 + 2x_1x_2 + 2x_1 - x_2)}\begin{bmatrix} 4 & -(4x_1 + 4x_2 + 6) \\ -(4x_1 + 4x_2 + 6) & \begin{matrix} 4x_1^2 + 2x_2^2 + 4x_1x_2 \\ +16x_1 + 10x_2 + 9 \end{matrix} \end{bmatrix}$$

$$\boxed{x_{k+1} = x_k - H(x_k)^{-1}\nabla F(x_k)}$$