

Lecture 14 - Optimization for Machine Learning

Dinesh Krishnamoorthy

Department of Engineering Cybernetics, NTNU

Spring 2026

Sources

Presentation based on material from:

- ▶ Nocedal & Wright (2006): Numerical optimization (Chapter 10)
- ▶ Boyd & Vandenberghe (2004): Convex Optimization (Chapter 6)

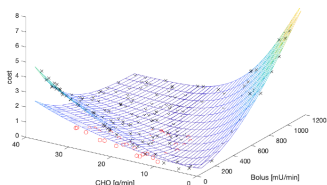
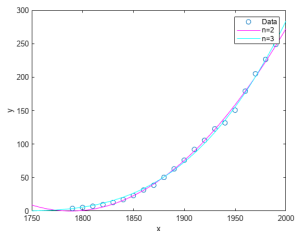
Data fitting / Machine learning

We have a data set $\mathcal{D} := \{(u_i, y_i)\}_{i=1}^N$

- ▶ $u \in \mathbb{R}^m$ - independent variables/input/features
- ▶ $y \in \mathbb{R}^n$ - dependent variables/outcome/response/label
- ▶

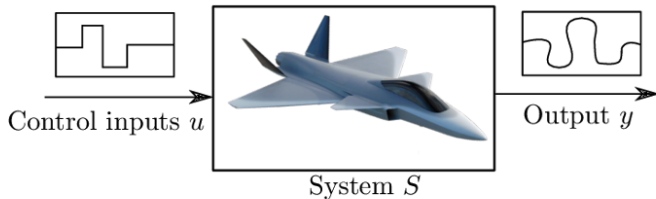
Goal: We want to be able to predict y , given u

Task: Build an inferential model $y = g_x(u)$ (supervised learning)



Examples

By fitting input-output data from a dynamic system we can identify transfer functions that describe the system.



Data fitting (Empirical Risk Minimization)

- ▶ Given the data set $\mathcal{D} := \{(u_i, y_i)\}_{i=1}^N$
- ▶ Choose a parametric model structure $y = g(u, x)$
- ▶ Find the parameter $x \in \mathbb{R}^p$ that gives the best fit

$$x^* = \arg \min_x \ell(x; \mathcal{D})$$

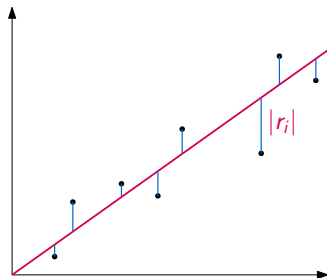
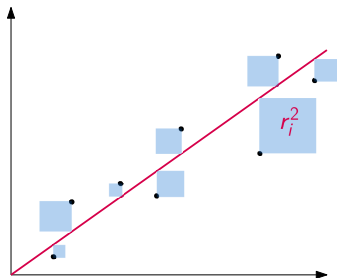
This is an unconstrained optimization problem.

Data fitting (Empirical Risk Minimization)

Loss function ℓ quantifies how good/bad a fit is.

- ▶ Mean squared error (MSE) $\ell(\mathbf{x}; \mathcal{D}) := \frac{1}{N} \sum_{i=1}^N \|r_i\|_2^2$
- ▶ Mean absolute error (MAE) $\ell(\mathbf{x}; \mathcal{D}) := \frac{1}{N} \sum_{i=1}^N |r_i|$

Residual: $r_i := y_i - g(\mathbf{x}, u_i)$



Least Squares Estimate: Analytical Solution

Analytical solution can also be conveniently written in matrix form:

$$Y := \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_N \end{bmatrix} \in \mathbb{R}^N \quad \Phi := \begin{bmatrix} u_1 & 1 \\ u_2 & 1 \\ \vdots & \vdots \\ u_N & 1 \end{bmatrix} \in \mathbb{R}^{N \times p}$$

$$\min_{x \in \mathbb{R}^p} \|Y - \Phi x\|^2$$

► First order necessary condition:

$$\Phi^T Y - \Phi^T \Phi x = 0$$

$$\Rightarrow \hat{x} = [\Phi^T \Phi]^{-1} \Phi^T Y$$

Least Square Curve Fit

$$\hat{x} = \arg \min_x \frac{1}{N} \sum_{i=1}^N \|y_i - g(x, u_i)\|_2^2$$

Data fitting using a linear combination of polynomials

Polynomial Basis

- ▶ $g(x, u_i) := a_0 + a_1 u_i + a_2 u_i^2 + \dots + a_p u_i^p = \phi_i^T x$
- ▶ $x := [a_0, a_1, a_2, \dots, a_p]^T$
- ▶ $\phi_i := [1, u_i, u_i^2, \dots, u_i^p]^T$

Question: Is there an analytical solution for fitting polynomials?

Least Square Curve Fit

$$\hat{x} = \arg \min_x \frac{1}{N} \sum_{i=1}^N \|y_i - g(x, u_i)\|_2^2 + R(x)$$

Data fitting using a linear combination of basis functions

$$\beta_j(u) = e^{-(u-\mu_j)^T(u-\mu_j)}$$

Radial Basis function Network

- ▶ $g(x, u_i) := a_0 + a_1\beta_1(u_i) + a_2\beta_2(u_i) + \dots + a_\eta\beta_\eta(u_i) = \phi_i^T x$
- ▶ $x := [a_0, a_1, a_2, \dots, a_\eta]^T$
- ▶ $\phi_i := [1, \beta_1(u_i), \beta_2(u_i), \dots, \beta_\eta(u_i)]^T$

Question: Is there an analytical solution for RBFN?

Chebyshev or minimax approximation

$$\min_x \|Y - \Phi x\|_\infty$$

- ▶ ℓ_∞ norm implies that we want to minimize the maximum residual

$$\max\{|r_1| + |r_2| + \cdots + |r_N|\}$$

- ▶ Not directly solvable $\rightarrow$ Re-cast as an LP

$$\begin{aligned} \min_{t,x} \quad & t \\ \text{s.t.} \quad & -t\mathbf{1} \leq Y - \Phi x \leq t\mathbf{1} \end{aligned}$$

where $x \in \mathbb{R}^p$ and $t \in \mathbb{R}$.

Robust estimator - Mean absolute error (MAE)

$$\min_x \|Y - \Phi x\|_1$$

- ▶ ℓ_1 norm implies that we want to minimize the sum of the absolute residuals

$$|r_1| + |r_2| + \cdots + |r_N|$$

- ▶ Not directly solvable $\rightarrow$ Re-cast as an LP

$$\min_{t,x} \mathbf{1}^T t$$

$$\text{s.t. } -t \leq Y - \Phi x \leq t$$

where $x \in \mathbb{R}^p$ and $t \in \mathbb{R}^N$.

Regularization

$$\hat{x} = \arg \min_x \frac{1}{N} \sum_{i=1}^N \|y_i - g(x, u_i)\|_2^2 + R(x)$$

- ▶ Tychonov/ ℓ_2 regularization: $R(x) := \omega \|x\|_2^2$ (Ridge regression)
- ▶ Sparse/ ℓ_1 regularization: $R(x) := \omega \|x\|_1$ (Lasso regression)

Weighted Least square

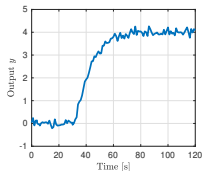
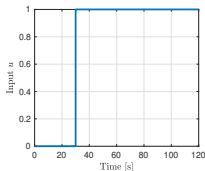
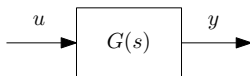
Weighted Least squares fitting can be used to give more importance to some data points over others.

$$\hat{x} = \arg \min_x \sum_{i=1}^N q_i \|y_i - g(x, u_i)\|_2^2$$

- ▶ $q_i > q_j$ implies that the data point (u_i, y_i) is more important than (u_j, y_j) for some $i \neq j$.

System Identification

Many engineering systems are dynamic.



System Identification uses input-output data to identify the system.

Question: Can this be framed as an optimization problem?

System Identification

Three basic entities

- ▶ Perturb the system to collect input-output data
- ▶ Choose a candidate model structure with unknowns x
- ▶ A rule by which the candidate models can be assessed $\rightarrow$ optimization

Artificial Neural Networks

$$\hat{x} = \arg \min_x \frac{1}{2} \sum_{i=1}^N \|y_i - g(x, u_i)\|_2^2 + R(x)$$

- ▶ No analytical solution
- ▶ Solve using Gradient Descent (see previous lecture)

Recap: Gradient Descent

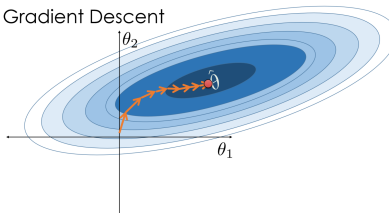
$$\hat{x} = \arg \min_x f(x) := \frac{1}{2} \sum_{i=1}^N \ell_i(x)$$

Search direction $s_k = -\nabla f(x_k)$

$$x_{k+1} = x_k - \alpha \sum_{i=1}^N \nabla \ell_i(x_k)$$

- ▶ Need to evaluate N gradients for ALL data points
- ▶ Repeat at every iteration
- ▶ Challenging if N very large!

Gradient Descent



Stochastic Gradient Descent

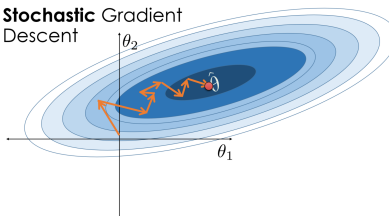
$$\hat{x} = \arg \min_x f(x) := \sum_{i=1}^N \ell_i(x)$$

Given a subset of samples $\mathcal{N}_k \subset \{1, 2, \dots, N\}$, and $b = \text{card}(\mathcal{N}_k)$

$$x_{k+1} = x_k - \alpha_k \sum_{i \in \mathcal{N}_k} \nabla \ell_i(x_k)$$

- Evaluate gradients for a small random subset of the data at each step.
- Repeat with different random subsets/batches at each iteration

Stochastic Gradient
Descent



Stochastic Gradient Descent

In the simplest case, we take a single data point, i.e., $b = 1$

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

- In expectation, we are moving in the direction of the gradient

$$\mathbb{E} [\nabla \ell_i(\mathbf{x}_k)] = \sum_{i=1}^N \nabla \ell_i(\mathbf{x}_k)$$

Stochastic Gradient Descent

In mini-batch case, we take a subset of samples, i.e., $1 < b < N$

$$\mathbf{x}_{k+1} = \mathbf{x}_k - \alpha_k \sum_{i \in \mathcal{N}_k} \nabla \ell_i(\mathbf{x}_k)$$

- Because we have $\mathbb{E} \left[\sum_{i \in \mathcal{N}_k} \nabla \ell_i(\mathbf{x}_k) \right] = \sum_{i=1}^N \nabla \ell_i(\mathbf{x}_k)$, the estimate is unbiased.

Stochastic Gradient Descent

For a problem with N data points, mini-batch size b and feature dimension m , we obtain the following computational costs

- ▶ Gradient Descent: $\mathcal{O}(Nm)$
- ▶ Stochastic gradient descent: $\mathcal{O}(m)$
- ▶ mini-batch SGD: $\mathcal{O}(bm)$

Nonlinear least squares problem

Recall the least squares problem

$$\min_{\mathbf{x}} \ell(\mathbf{x}) = \frac{1}{2} \sum_{i=1}^N r_i^2(\mathbf{x})$$

Define:

$$\mathbf{r}(\mathbf{x}) := [r_1(\mathbf{x}), r_2(\mathbf{x}), \dots, r_N(\mathbf{x})]^T$$

$$J(\mathbf{x}) := \begin{bmatrix} \nabla r_1(\mathbf{x})^T \\ \nabla r_2(\mathbf{x})^T \\ \vdots \\ \nabla r_N(\mathbf{x})^T \end{bmatrix}$$

$$\begin{aligned} \nabla \ell(\mathbf{x}) &= \sum_{i=1}^N \nabla r_i(\mathbf{x})^T r_i(\mathbf{x}) \\ &= J(\mathbf{x})^T \mathbf{r}(\mathbf{x}) \end{aligned}$$

$$\begin{aligned} \nabla^2 \ell(\mathbf{x}) &= \sum_{i=1}^N \nabla r_i(\mathbf{x}) \nabla r_i(\mathbf{x})^T + \sum_{i=1}^N r_i(\mathbf{x}) \nabla^2 r_i(\mathbf{x}) \\ &= J(\mathbf{x})^T J(\mathbf{x}) + \sum_{i=1}^N r_i(\mathbf{x}) \nabla^2 r_i(\mathbf{x}) \end{aligned}$$

Nonlinear least squares problem

- Recollect Newton iterate:

$$H_k s_k = -J_k$$

where $H_k := \nabla^2 \ell(x_k)$ and $J_k := \nabla \ell(x_k)$.

- Computing Hessian $H_k = J_k^T J_k + \sum_{i=1}^N r_i(x_k) \nabla^2 r_i(x_k)$ is very expensive.
- So, let's approximate

$$H_k \approx J_k^T J_k$$

- & compute the descent step

$$J_k^T J_k s_k = -J_k^T r_k$$

Nonlinear least squares problem

$$J_k^T J_k s_k = -J_k^T r_k$$

Gauss-Newton Algorithm

- Update x_{k+1} in the direction of s_k using line search method.

Levenberg-Marquardt Algorithm

- Update x_{k+1} in the direction of s_k using trust region method.

What did you learn?

Data Fitting

- Dataset $\mathcal{D} = \{(u, y)\}$
- Candidate model $g(x, u)$
- A rule to find best x

Unconstrained Optimization

$$\ell(x; \mathcal{D}) := \frac{1}{N} \sum_{i=1}^N \|y_i - g(x, u_i)\|^2$$

Iterative methods

- Gradient descent
- Stochastic gradient descent (SGD)

Analytical solution $x^* = [\Phi^T \Phi]^{-1} \Phi^T Y$

- Linear regression

$$\Phi = [U, 1] \quad X = [m, c]^T$$

- Linear combination of basis functions

$$\Phi = [\beta_n(u), \dots, \beta_2(u), \beta_1(u), 1]$$

$$X = [a_n, \dots, a_2, a_1, a_0]^T$$

(e.g., polynomials, radial basis, etc.)

$$x_{k+1} = x_k - \alpha_k \frac{1}{b} \sum_{i \in \mathcal{N}_k} \nabla \ell_i(x)$$

