# Maximum Likelihood Estimation

Prof. C. F. Jeff Wu

ISyE 8813

Section 1

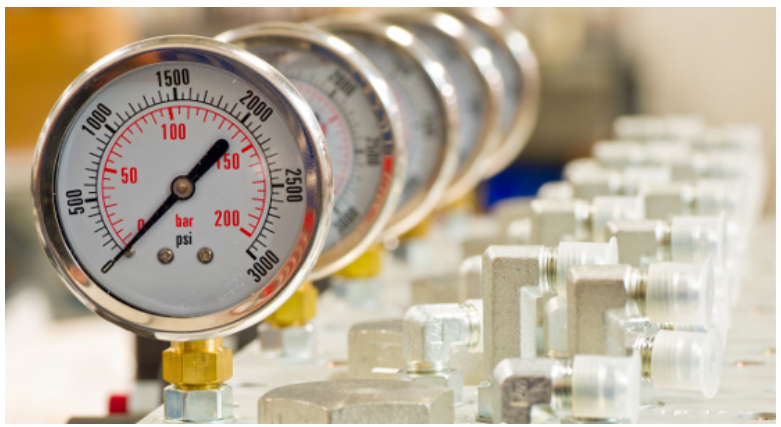
# Motivation

# What is parameter estimation?

- A modeler proposes a **model** $\mathcal{M}(\theta)$ for explaining some **observed** phenomenon
- $\theta$ are the **parameters** which dictate properties of such a model
- **Parameter estimation** is the process by which the modeler determines the best parameter choice $\theta$ given a set of training observations

# Why parameter estimation?

- $\theta$ often encodes important and interpretable properties of the model $\mathcal{M}$
  - e.g., calibration parameters in computer experiment models
- An estimate of $\theta$ (call this $\widehat{\theta}$) allows for:
  - **Model validation**: Checking whether the proposed model fits the observed data
  - **Prediction**: Forecasting future observations at untested settings



# Simple example

- Suppose we **observe** a random sample $X_1, X_2, \cdots, X_n$:
  - $X_i = 1$ if student $i$ own a sports car,
  - $X_i = 0$ if student $i$ does not own a sports car
- We then postulate the following **model**: $X_i \overset{\text{i.i.d.}}{\sim} \text{Bernoulli}(\theta)$, i.e., $X_i$'s are i.i.d. Bernoulli random variables with the same (unknown) parameter $\theta$
- We would like to estimate $\theta$ based on the observations $X_1, X_2, \cdots, X_n$

# Popular estimation techniques


- Maximum-likelihood estimation (MLE)
- Mnimax estimation
- Methods-of-moments (MOM)
- (Non-linear) Least-squares estimation

# Popular estimation techniques


- **Maximum-likelihood estimation (MLE)**
- Mnimax estimation
- Methods-of-moments (MOM)
- **(Non-linear) Least-squares estimation**

We will focus on these two techniques in this lecture.

Section 2

# Maximum likelihood estimation (MLE)

# Likelihood function

- In words, MLE chooses the parameter setting which maximizes the **likelihood** of the observed sample
- But how do we define this likelihood as a function of $\theta$?

**Definition**

Let $f(\underline{x}|\theta)$ be the probability of observating the sample $\underline{x} = (x_1, \cdots, x_n)$ from $M(\theta)$. The **likelihood function** is defined as:

$$L(\theta|\underline{x}) = f(\underline{x}|\theta)$$

- MLE tries to find the parameter setting which maximizes the probability of observing the data

# Log-likelihood Function

- If $X_1, X_2, \cdots, X_n \overset{i.i.d.}{\sim} f(\underline{x}|\theta)$, the likelihood function simplifies to $L(\theta|\underline{x}) = \prod_{i=1}^{n} f(x_i|\theta)$
- In this case, the likelihood can become very small, because we are multiplying many terms
- To fix this computational problem, the **log-likelihood** $l(\theta|\underline{x}) = \log(L(\theta|\underline{x}))$ is often used instead
- For i.i.d observations, this becomes:

$$l(\theta|\underline{x}) = \log(L(\theta|\underline{x})) = \sum_{i=1}^{n} \log[f(x_i|\theta|\underline{x})]$$

# Maximum likehood estimator

We can now formally define the estimator for MLE:

**Definition**

Given observed data $\underline{x}$, the **maximum likelihood estimator** (MLE) of $\theta$ is defined as:

$$\widehat{\theta} \in \underset{\theta}{\text{Argmax}}[L(\theta|\underline{x})]$$

Equivalently, because the log-function is monotonic, we can instead solve for:

$$\widehat{\theta} \in \underset{\theta}{\text{Argmax}}[l(\theta|\underline{x})]$$

The latter is more numerically stable for optimization.

# MLE: a simple example

- Suppose some count data is observed $X_i \in \mathbb{Z}_+$, and the following Poisson model is assumed $X_i \overset{i.i.d.}{\sim} Pois(\lambda)$
- The **likelihood function** can be shown to be

$$L(\lambda|\underline{x}) = \prod_{i=1}^{n} \left( \frac{\lambda^{x_i}}{x_i!} e^{-\lambda} \right)$$

with **log-likelihood function**:

$$l(\lambda|\underline{x}) = \log\left[ \left( \prod_{i=1}^{n} \frac{\lambda^{x_i}}{x_i!} e^{-\lambda} \right) \right] = \log(\lambda) \left( \sum_{i=1}^{n} x_i \right) - \sum_{i=1}^{n} \log(x_i!) - n\lambda$$

# MLE: a simple example

Since $l(\lambda|\underline{x})$ is differentiable, we can solve for its minimizer by:

- Differentiating the log-likelihood:

$$\frac{d[l(\lambda|\underline{x})]}{d\lambda} = \frac{1}{\lambda}\sum_{i=1}^{n} x_i - n$$

- Setting it to 0, and solving for $\lambda$:

$$\widehat{\lambda} = \frac{\sum_{i=1}^{n} x_i}{n} = \bar{X}$$

- Checking the Hessian matrix is positive-definite:

$$\nabla^2_{\lambda} l(\lambda|\underline{x}) \succeq 0$$

# What if a closed-form solution does not exist ☹?


In most practical models, there are two computational difficulties:

- No closed-form solution exists for the MLE,
- Multiple locally optimal solutions or stationary points.

Standard **non-linear optimization** methods (Nocedal and Wright, 2006) are often used as a black-box technique for obtaining locally optimal solutions.

# What if a closed-form solution does not exist ☹?

We can do better if the model exhibits specific structure:

- If optimization program is **convex**, one can use some form of accelerated gradient descent (Nesterov, 1983)
- If **non-convex** but **twice-differentiable**, the limited-memory Broyden-Fletcher-Goldfard-Shanno (L-BFGS) method works quite well for local optimization
- If **missing data** is present, the EM algorithm (Dempster et al, 1977; Wu, 1983) can be employed

Section 3

# Non-linear least-squares estimation

(Adapted from
en.wikipedia.org/wiki/Non-linear_least_squares)

# Problem statement

Consider a set of $m$ data points, $(x_1, y_1), (x_2, y_2), \ldots, (x_m, y_m)$, and a **curve** (model function) $y = f(x; \boldsymbol{\beta})$:

- Curve model depends on $\boldsymbol{\beta} = (\beta_1, \beta_2, \ldots, \beta_n)$, with $m \geq n$
- Want to choose the parameters $\boldsymbol{\beta}$ so that the curve **best fits** the given data in the **least squares** sense, i.e., such that

$$S(\boldsymbol{\beta}) = \sum_{i=1}^{m} r_i(\boldsymbol{\beta})^2$$

is minimized, where the residuals $r_i(\boldsymbol{\beta})$ are defined as $r_i(\boldsymbol{\beta}) = y_i - f(x_i; \boldsymbol{\beta})$

# Problem statement

The minimum value of S occurs when the gradient equals zero:

- Since the model contains n parameters, there are $n$ gradient equations to solve:

$$\frac{\partial S}{\partial \beta_j} = 2 \sum_i r_i \frac{\partial r_i}{\partial \beta_j} = 0, \quad j = 1, \ldots, n.$$

- In a nonlinear system, this system of equations typically do not have a closed-form solution ☹

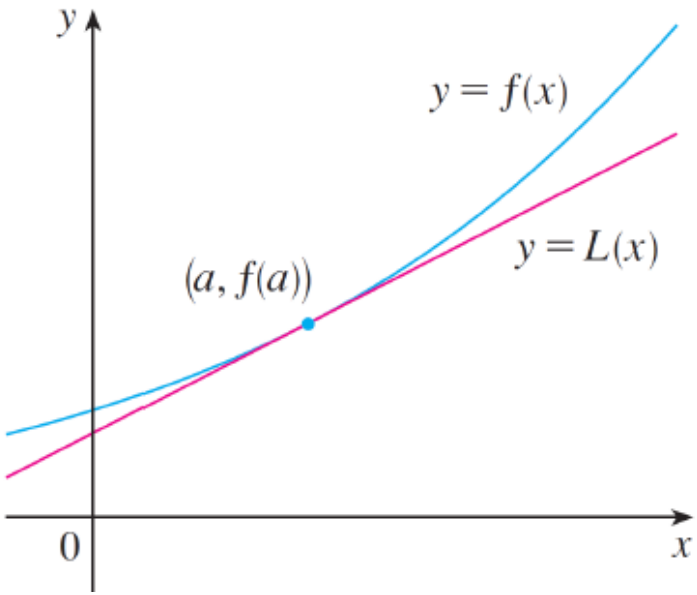
One way to solve for this system is:

- Set initial values for parameters
- Update parameters using the successive iterations:

$$\beta_j^{(k+1)} = \beta_j^{(k)} + \Delta\beta_j.$$

Here, $k$ is the increment count, and $\Delta\boldsymbol{\beta}$ is known as the **shift vector**

# Deriving the normal equations




From first-order Taylor series expansion about $\boldsymbol{\beta}^{k}$, we get:

$$f(x_i; \boldsymbol{\beta}) \approx f(x_i; \boldsymbol{\beta}^{(k)}) + \sum_j \frac{\partial f(x_i; \boldsymbol{\beta}^{(k)})}{\partial \beta_j} \left( \beta_j - \beta_j^{(k)} \right) \approx f(x_i; \boldsymbol{\beta}^{(k)}) + \sum_j J_{ij} \, \Delta \beta_j,$$

where $\mathbf{J}$ is the **Jacobian** matrix.

# Deriving the normal equations

In terms of the linearized model:

$$J_{ij} = -\frac{\partial r_i}{\partial \beta_j},$$

with residuals given by:

$$r_i = \left(y_i - f(x_i; \boldsymbol{\beta}^{(k)})\right) + \left(f(x_i; \boldsymbol{\beta}^{(k)}) - f(x_i; \boldsymbol{\beta})\right) = \Delta y_i - \sum_{s=1}^{n} J_{is} \Delta \beta_s$$

where $\Delta y_i = y_i - f(x_i; \boldsymbol{\beta}^{(k)})$. Substituting these into the gradient equations, we get:

$$-2 \sum_{i=1}^{m} J_{ij} \left( \Delta y_i - \sum_{s=1}^{n} J_{is} \, \Delta \beta_s \right) = 0$$

# Deriving the normal equations

This system of equations is better known as the set of **normal equations**:

$$\sum_{i=1}^{m} \sum_{s=1}^{n} J_{ij} J_{is} \, \Delta\beta_s = \sum_{i=1}^{m} J_{ij} \, \Delta y_i \qquad j = 1, \ldots, n.$$

In matrix form, this becomes:

$$\left(\mathbf{J}^{\mathsf{T}} \mathbf{J}\right) \Delta\boldsymbol{\beta} = \mathbf{J}^{\mathsf{T}} \, \Delta\mathbf{y}$$

where $\Delta\mathbf{y} = (\Delta y_1, \cdots, \Delta y_n)^{\mathsf{T}}$.

# Solving the normal equations

Several methods have been proposed to solve the set of normal equations:

- Gauss-Newton method
- Levenberg-Marquardt algorithm
- Gradient methods
  - Davidson-Fletcher-Powell
  - Steepest descent

# Solving the normal equations

Several methods have been proposed to solve the set of normal equations:

- **Gauss-Newton method**
- **Levenberg-Marquardt-Fletcher algorithm**
- Gradient method
  - Davidson-Fletcher-Powell
  - Steepest descent

We will focus on two in this lecture.

# Gauss-Newton method

Starting with an initial guess $\boldsymbol{\beta}^{(0)}$ for the minimum, the **Gauss-Newton method** iteratively updates $\boldsymbol{\beta}^{(k)}$ by solving for the shift vector $\Delta\boldsymbol{\beta}$ in the normal equations:

$$\boldsymbol{\beta}^{(k+1)} = \boldsymbol{\beta}^{(k)} + \left(\mathbf{J}^{\mathsf{T}}\mathbf{J}\right)^{-1}\mathbf{J}^{\mathsf{T}}\mathbf{r}(\boldsymbol{\beta}^{(k)}).$$

This approach can encounter several problems:

- $(\mathbf{J}^{\mathsf{T}}\mathbf{J})^{-1}$ is often **ill-conditioned**,
- When far from fixed-point solution, such an iterative map may not be contractive; the sequence $(\boldsymbol{\beta}^{(k)})_{k=1}^{\infty}$ may diverge without reaching a limiting solution.

# Levenberg-Marquardt-Fletcher algorithm


Historical perspective of the **Levenberg-Marquardt-Fletcher (LMF)** algorithm:

- The first form of LMF was first published in 1944 by Kenneth Levenberg, while working at the Frankford Army Arsenal,
- It was then rediscovered and improved upon by Donald Marquardt in 1963, who worked as a statistician at DuPont,
- A further modification was suggested by Roger Fletcher in 1971, which greatly improved the robustness of the algorithm.

# Levenberg's contribution

To improve the numerical stability of the algorithm, Levenberg (1944) proposed the modified iterative updates:

$$\boldsymbol{\beta}^{(k+1)} = \boldsymbol{\beta}^{(k)} + \left(\mathbf{J}^{\mathsf{T}}\mathbf{J} + \lambda\mathbf{I}\right)^{-1}\mathbf{J}^{\mathsf{T}}\mathbf{r}(\boldsymbol{\beta}^{(k)}),$$

where $\lambda$ is a **damping factor** and $\mathbf{I}$ is the identity matrix.

- $\lambda \to 0^{+}$ gives the previous **Gauss-Newton** updates, which converges quickly but may diverge,
- $\lambda \to \infty$ gives the **steepest-descent** updates, which converge slowly but is more stable.

# Marquardt's contribution

How does one choose $\lambda$ to control this **trade-off** between quick convergence and numerical stability? Marquardt (1963) proposes the improved iterative updates:

$$\boldsymbol{\beta}^{(k+1)} = \boldsymbol{\beta}^{(k)} + \left(\mathbf{J}^{\mathsf{T}}\mathbf{J} + \lambda^{(k)}\mathbf{I}\right)^{-1} \mathbf{J}^{\mathsf{T}}\mathbf{r}(\boldsymbol{\beta}^{(k)}),$$

where the damping factor $\lambda^{(k)}$ can vary between iterations.

- When convergence is stable, the damping factor is iteratively decreased by $\lambda^{(k+1)} \leftarrow \lambda^{(k)}/\nu$ to exploit the accelerated Gauss-Newton rate,
- When divergence is observed, the damping factor is increased by $\lambda^{(k+1)} \leftarrow \lambda^{(k)}\nu$ to restore stability.

# Fletcher's contribution

Fletcher (1971) proposes a further improvement of this algorithm through the following update scheme:

$$\boldsymbol{\beta}^{(k+1)} = \boldsymbol{\beta}^{(k)} + \left(\mathbf{J}^{\mathsf{T}}\mathbf{J} + \lambda\, \mathrm{diag}\{\mathbf{J}^{\mathsf{T}}\mathbf{J}\}\right)^{-1} \mathbf{J}^{\mathsf{T}}\mathbf{r}(\boldsymbol{\beta}^{(k)}),$$

where $\mathrm{diag}\{\mathbf{J}^{\mathsf{T}}\mathbf{J}\}$ is a diagonal matrix of the diagonal entries in $\mathbf{J}^{\mathsf{T}}\mathbf{J}$.

- **Intuition:** By scaling each gradient component by the curvature, greater movement is encouraged in directions where the gradient is smaller,
- Can be viewed as a pre-conditioning step for solving ill-conditioned problems,
- Similar approach is used in **Tikhonov regularization** for linear least-squares.

# Summary

- **Parameter estimation** is an unavoidable step in model validation and prediction,
- **MLE** is a popular approach for parameter estimation,
- For the non-linear least-squares problem, the **Levenberg-Marquardt-Fletcher algorithm** provides a stable and efficient method for estimating coefficients.