Machine Learning

Linear Regression

Dariush Hosseini

dariush.hosseini@ucl.ac.uk
Department of Computer Science
University College London

Lecture Overview

Learning Outcomes for Today's Lecture

By the end of this lecture you should:

1. Understand the purpose of **Linear Regression**
2. Understand motivations for some approaches to linear regression, in particular **Ordinary Least Squares** (**OLS**)
3. Know how the **normal equations** and **gradient descent** can be used to solve the **OLS** approach to linear regression

Lecture Overview

Setting

- Recall that in regression problems we seek to **learn** a mapping between input features and a continuous output label
- We can then use this mapping to make output **predictions** given novel input data
- In **linear regression** we seek mappings which are linear functions of the input features

Notation

- **Inputs**
  $\mathbf{x} = [1, x_1, \ldots, x_m]^T \in \mathbb{R}^{m+1}$
- **Outputs**
  $y \in \mathbb{R}$
- **Training Data**
  $\mathcal{S} = \{(\mathbf{x}^{(i)}, y^{(i)})\}_{i=1}^{n}$

Probabilistic Environment

- We assume:
  - $\mathbf{x}$ is the outcome of a random variable $\mathcal{X}$
  - y is the outcome of a random variable $\mathcal{Y}$
  - $(\mathbf{x}, y)$ are drawn i.i.d. from some data generating distribution, $\mathcal{D}$, i.e.:

$$(\mathbf{x}, y) \sim \mathcal{D}$$

and:

$$\mathcal{S} \sim \mathcal{D}^n$$

Representation

- We seek to learn a linear mapping, $f_{\mathbf{w}}$, characterised by a weight vector $\mathbf{w} \in \mathbb{R}^{m+1}$, and drawn from a function class $\mathcal{F}$:

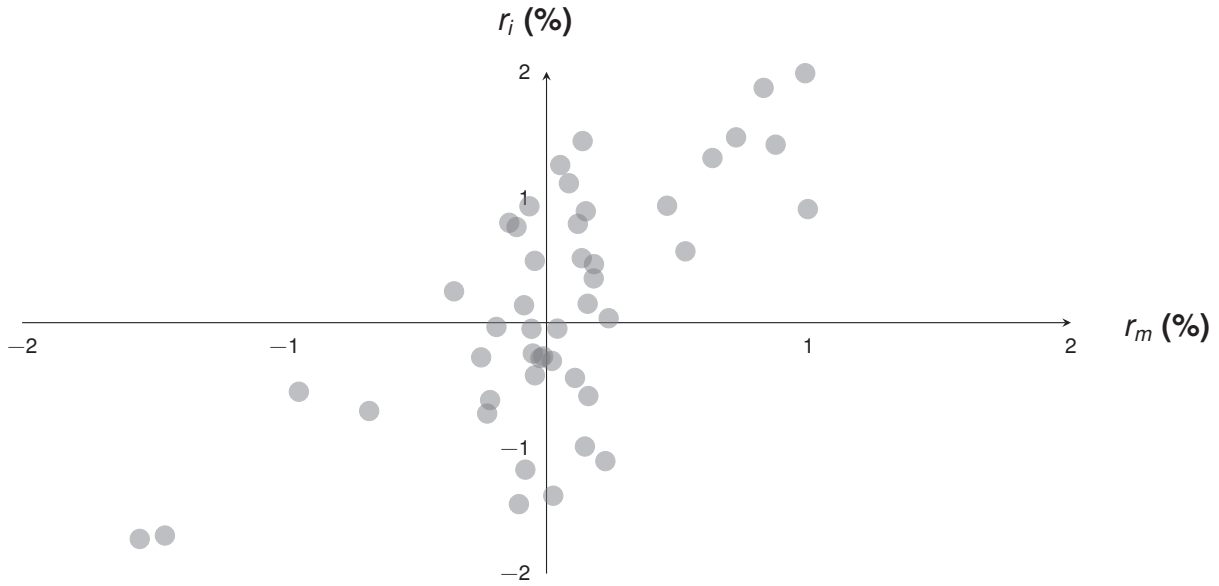
$$\mathcal{F} = \{f_{\mathbf{w}}(\mathbf{x}) = \mathbf{w} \cdot \mathbf{x} | \mathbf{w} = [w_0, w_1, ..., w_m]^T \in \mathbb{R}^{m+1}\}$$

- The mapping for a particular $\mathbf{w}$ is:

$$f_{\mathbf{w}}(\mathbf{x}) = w_0 + w_1 x_1 + ... + w_m x_m$$

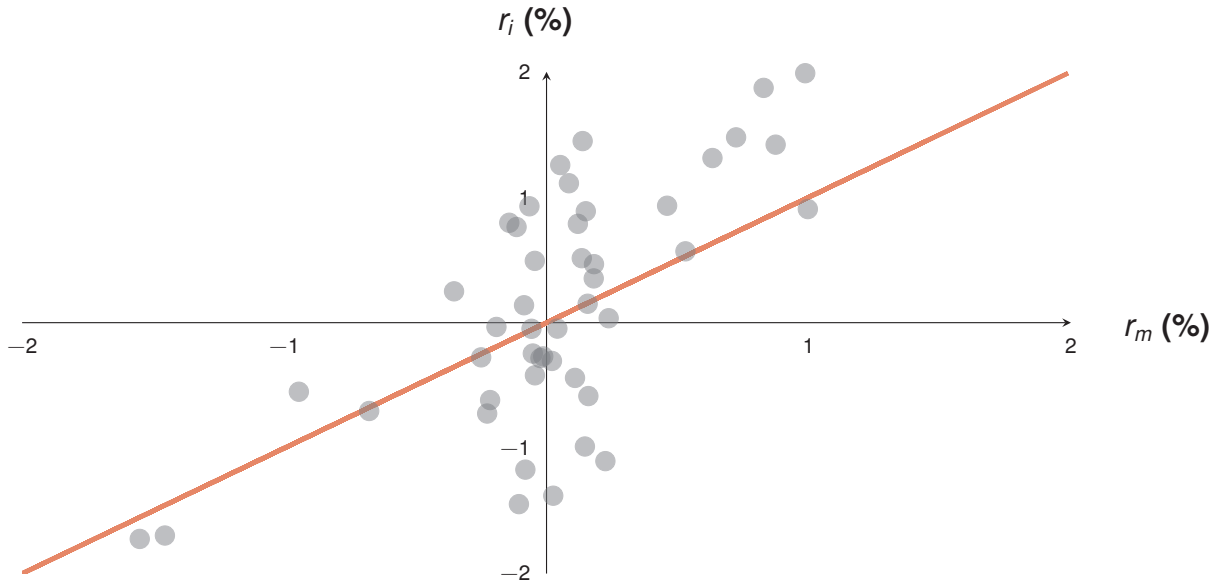
- Where w_0 is the **bias** term

Example: CAPM



- r_m is the **Market Return** over the **Risk-Free Return**
- r_i is the **Individual Stock Return** over the **Risk-Free Return**
- Equilibrium model of relationship between the returns of individual stocks and the broad market return

Example: CAPM



- r_m is the **Market Return** over the **Risk-Free Return**
- r_i is the **Individual Stock Return** over the **Risk-Free Return**
- Equilibrium model of relationship between the returns of individual stocks and the broad market return

Example: CAPM

- For our purposes we can treat it as a 1-dimensional illustration of linear regression, in which:
  - $\mathbf{x} = [1, r_m]^T$
  - $y = r_i$
- And we seek $\mathbf{w}$ such that:
  - $f_{\mathbf{w}}(r_m) = w_1 \times r_m$
  - $w_0 = 0$
- Empirically, the relationship is not strong, and often the model is extended to encompass further input attributes:
  - For example the **Fama-French Three Factor Model** includes inputs related to **Market Capitalisation** and **Book-to-Market Value**

Rephrasing The Problem

- We can represent our input training data more compactly as the **design matrix**, **X**:

$$\mathbf{X} = \begin{bmatrix} \mathbf{x}^{(1)T} \\ \mathbf{x}^{(2)T} \\ \cdot \\ \cdot \\ \mathbf{x}^{(n)T} \end{bmatrix} = \begin{bmatrix} 1 & x_1^{(1)} & \cdot & \cdot & x_m^{(1)} \\ 1 & x_1^{(2)} & \cdot & \cdot & x_m^{(2)} \\ \cdot & \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot & \cdot \\ 1 & x_1^{(n)} & \cdot & \cdot & x_m^{(n)} \end{bmatrix}$$

Rephrasing The Problem

- And our output training data as y:

$$\mathbf{y} = \begin{bmatrix} y^{(1)} \\ y^{(2)} \\ . \\ . \\ y^{(n)} \end{bmatrix}$$

- And we seek $\mathbf{w}$ such that:

$$\mathbf{y} = \mathbf{X}\mathbf{w}$$

Solution?

- Using our knowledge of matrix algebra can't we just solve this system of equations as follows:

$$\mathbf{w} = \mathbf{X}^{-1}\mathbf{y}$$

- No! (at least not in general)
- In general **X** is **singular** and therefore **non-invertible**
- **X** is only invertible if it is square, $n = (m + 1)$, and it is of full rank, $\text{rank}(\mathbf{X}) = m + 1$

Solution?

- Recall from our work on linear algebra that:
  - If $\text{rank}(\mathbf{X}) = \text{rank}(\mathbf{X}|\mathbf{y}) = (m+1) \leqslant n$
    - The equations are **consistent**
    - We have at least the same number of such equations as unknowns
    - We have a **unique** solution
  - If $\text{rank}(\mathbf{X}) = \text{rank}(\mathbf{X}|\mathbf{y}) < m+1$
    - The equations are **consistent**...
    - ...But $n < (m+1)$ (assuming the n equations are distinct), so our problem is **underdetermined** with no unique solutions
    - We have too few equations to specify the unknowns uniquely
  - If $\text{rank}(\mathbf{X}) < \text{rank}(\mathbf{X}|\mathbf{y})$
    - The equations are **inconsistent** and no solutions exists
    - We need a further condition to make the problem **well-posed**...

Evaluation

- In general we cannot find a $\mathbf{w}$ such that $y^{(i)} = f_{\mathbf{w}}(\mathbf{x}^{(i)})$ for all i
- We need an extra condition to guide us in choosing a particular $\mathbf{w}$ given $\mathcal{S}$
- Let's try using the **empirical squared error** evaluation function[1]:

$$\begin{aligned} \mathrm{L}(\mathcal{E}, \mathcal{S}, f_{\mathbf{w}}) &= \frac{1}{2} \sum_{i=1}^{n} (y^{(i)} - f_{\mathbf{w}}(\mathbf{x}^{(i)}))^2 \\ &= \frac{1}{2} \sum_{i=1}^{n} (y^{(i)} - \mathbf{w} \cdot \mathbf{x}^{(i)})^2 \end{aligned}$$

[1] N.B. - here and throughout this lecture, we drop the usual $\frac{1}{n}$ multiplicative constant for convenience. We can do this because the resulting optimisation problem will be equivalent.

Ordinary Least Squares

- We can use this evaluation function to select the optimal weight vector:

$$\underset{\mathbf{w}}{\operatorname{argmin}} \left(\frac{1}{2} \sum_{i=1}^{n} (y^{(i)} - \mathbf{w} \cdot \mathbf{x}^{(i)})^2 \right)$$

- In this case our optimal **w** is known as the **Ordinary Least Squares** (**OLS**) estimator ($\mathbf{w}_{\text{OLS}}$)

- And this approach to linear regression is known as OLS regression
  - Note that other evaluation functions lead to other approaches to linear regression

Lecture Overview

Motivation

- Why should we pick the least squares estimator?
- We can motivate this choice in a number of different ways:

Motivation 1:
Additive Noise Model with i.i.d. Noise

- Assume $(y^{(i)}, \mathbf{x}^{(i)})$ are related by an additive noise model:

$$y^{(i)} = \mathbf{w} \cdot \mathbf{x}^{(i)} + \varepsilon^{(i)}$$

  Where $\varepsilon^{(i)}$ are the outcomes of an i.i.d. random variable ϵ

- If we make these assumptions then the **Gauss Markov Theorem** tells us that the OLS estimator, $\mathbf{w}_{\text{OLS}}$, is the **Best Linear Unbiased Estimator** (**BLUE**)

Motivation 2:
Additive Noise Model with i.i.d. Gaussian Noise

- Assume further that the noise is normally distributed:

$$\varepsilon \sim \mathcal{N}(0, \sigma^2) \qquad \Longrightarrow \qquad y|\mathbf{x} \sim \mathcal{N}(\mathbf{w} \cdot \mathbf{x}, \sigma^2)$$

- Given this further assumption, then the Maximum Likelihood Estimator, $\mathbf{w}_{\text{MLE}} = \mathbf{w}_{\text{OLS}}$:

$$\begin{aligned}
\mathbb{P}\left(\{y^{(i)}\}_{i=1}^n | \{\mathbf{x}^{(i)}\}_{i=1}^n; \mathbf{w}, \sigma\right) &= \prod_{i=1}^n p_y(y^{(i)}|\mathbf{x}^{(i)}; \mathbf{w}, \sigma) \\
&= \prod_{i=1}^n \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{(y^{(i)} - \mathbf{w} \cdot \mathbf{x}^{(i)})^2}{2\sigma^2}\right)
\end{aligned}$$

Motivation 2:
Additive Noise Model with i.i.d. Gaussian Noise

$$
\begin{aligned}
\mathbf{w}_{\text{MLE}} &= \underset{\mathbf{w}}{\text{argmax}} \quad \ln \left(\prod_{i=1}^{n} \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{(y^{(i)} - \mathbf{w} \cdot \mathbf{x}^{(i)})^2}{2\sigma^2} \right) \right) \\
&= \underset{\mathbf{w}}{\text{argmax}} \sum_{i=1}^{n} \ln \left(\frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{(y^{(i)} - \mathbf{w} \cdot \mathbf{x}^{(i)})^2}{2\sigma^2} \right) \right) \\
&= \underset{\mathbf{w}}{\text{argmax}} \left(-n \ln \sqrt{2\pi\sigma^2} - \sum_{i=1}^{n} \left(\frac{(y^{(i)} - \mathbf{w} \cdot \mathbf{x}^{(i)})^2}{2\sigma^2} \right) \right) \\
&= \underset{\mathbf{w}}{\text{argmin}} \frac{1}{2} \sum_{i=1}^{n} (y^{(i)} - \mathbf{w} \cdot \mathbf{x}^{(i)})^2 \\
&= \mathbf{w}_{\text{OLS}}
\end{aligned}
$$

Aside:
Additive Noise Model with i.i.d. Laplacian Noise

- Assume instead that the noise is Laplace distributed:

$$\varepsilon \sim \text{Laplace}(\mu, b) \qquad \text{where:} \qquad \mu \in \mathbb{R}, \quad b \in \mathbb{R}^+$$

- This has a characteristic pdf, p_ϵ:

$$p_\epsilon(\varepsilon; \mu, b) = \frac{1}{2b} \exp\left(-\frac{|\varepsilon - \mu|}{b}\right)$$

$$\mathbb{E}_{\mathcal{D}}[\epsilon] = \mu$$

$$\text{Var}_{\mathcal{D}}[\epsilon] = 2b^2$$

Aside:
Additive Noise Model with i.i.d. Laplacian Noise

- For $\mu = 0$, the likelihood of our data is given by:

$$\mathbb{P}\left(\{y^{(i)}\}_{i=1}^{n} | \{\mathbf{x}^{(i)}\}_{i=1}^{n}; \mathbf{w}, \sigma\right) = \prod_{i=1}^{n} p_{\mathcal{Y}}(y^{(i)} | \mathbf{x}^{(i)}; \mathbf{w}, \sigma)$$

$$= \prod_{i=1}^{n} \frac{1}{2b} \exp\left(-\frac{|y^{(i)} - \mathbf{w} \cdot \mathbf{x}^{(i)}|}{b}\right)$$

Aside:
Additive Noise Model with i.i.d. Laplacian Noise

$$
\begin{aligned}
\mathbf{w}_{\text{MLE}} &= \underset{\mathbf{w}}{\text{argmax}} \quad \ln\left(\prod_{i=1}^{n} \frac{1}{2b} \exp\left(-\frac{|y^{(i)} - \mathbf{w}\cdot\mathbf{x}^{(i)}|}{b}\right)\right) \\
&= \underset{\mathbf{w}}{\text{argmax}} \sum_{i=1}^{n} \ln\left(\frac{1}{2b} \exp\left(-\frac{|y^{(i)} - \mathbf{w}\cdot\mathbf{x}^{(i)}|}{b}\right)\right) \\
&= \underset{\mathbf{w}}{\text{argmax}} \left(-n\ln 2b - \sum_{i=1}^{n} \frac{|y^{(i)} - \mathbf{w}\cdot\mathbf{x}^{(i)}|}{b}\right) \\
&= \underset{\mathbf{w}}{\text{argmin}} \sum_{i=1}^{n} |y^{(i)} - \mathbf{w}\cdot\mathbf{x}^{(i)}|
\end{aligned}
$$

- This is the **absolute deviation** evaluation function
  - Estimator is more **robust** to **outliers** than that of the **squared error**

Motivation 3

- The squared error function itself has directly attractive properties, it is:
  - **Intuitive**
  - **Smooth**
  - **Convex**
- Recall the **PAC** approach:
  - Does not attempt to characterise a data generating distribution (in this case the additive noise distribution)
  - Instead characterises a well-behaved evaluation function, appropriate to the task at hand

Lecture Overview

Recap

- **Representation:**

$$\mathcal{F} = \{\mathbf{w} \cdot \mathbf{x} | \mathbf{w} \in \mathbb{R}^{m+1}\}$$

- **Evaluation:**

$$\frac{1}{2}\sum_{i=1}^{n}(y^{(i)} - \mathbf{w} \cdot \mathbf{x}^{(i)})^2$$

- **Optimisation:**

$$\mathbf{w}_{\text{OLS}} = \underset{\mathbf{w}}{\text{argmin}} \frac{1}{2}\sum_{i=1}^{n}(y^{(i)} - \mathbf{w} \cdot \mathbf{x}^{(i)})^2$$

- But how do we perform this optimisation?

Analytic Optimisation

- Let us re-write our evaluation function:

$$\begin{aligned} \mathsf{L}(\mathcal{E}, \mathcal{S}, f) &= \frac{1}{2} \sum_{i=1}^{n} (y^{(i)} - \mathbf{w} \cdot \mathbf{x}^{(i)})^2 \\ &= \frac{1}{2} \|\mathbf{y} - \mathbf{X}\mathbf{w}\|_2^2 \end{aligned}$$

- This function is **continuous** and **differentiable** in **w**

Analytic Optimisation

- For functions of this type we can solve by finding the **stationary point**, i.e. the **w** for which $\nabla_{\mathbf{w}} \mathsf{L} = 0$
- Here:

$$\nabla_{\mathbf{w}} \mathsf{L} = \left[\frac{\partial \mathsf{L}}{\partial w_0}, \frac{\partial \mathsf{L}}{\partial w_1}, \cdots, \frac{\partial \mathsf{L}}{\partial w_m} \right]^T$$

Analytic Optimisation

- ...And by checking for **convexity** i.e. $\nabla^2_{\mathbf{w}} \mathrm{L} \succeq 0$:
- This ensures that the stationary point is **globally optimal**
- Here $\nabla^2_{\mathbf{w}} \mathrm{L}$ is the **Hessian Matrix**, $\mathcal{H}$:

$$\mathcal{H} = \begin{bmatrix} \frac{\partial^2 \mathrm{L}}{\partial w_0^2} & \frac{\partial^2 \mathrm{L}}{\partial w_0 \, \partial w_1} & \cdots & \frac{\partial^2 \mathrm{L}}{\partial w_0 \, \partial w_m} \\ \frac{\partial^2 \mathrm{L}}{\partial w_1 \, \partial w_0} & \frac{\partial^2 \mathrm{L}}{\partial w_1^2} & \cdots & \frac{\partial^2 \mathrm{L}}{\partial w_1 \, \partial w_m} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial^2 \mathrm{L}}{\partial w_m \, \partial w_0} & \frac{\partial^2 \mathrm{L}}{\partial w_m \, \partial w_1} & \cdots & \frac{\partial^2 \mathrm{L}}{\partial w_m^2} \end{bmatrix}$$

Analytic Optimisation

- Compute the derivative of $\mathsf{L}(\mathcal{E}, \mathcal{S}, f)$:

$$\begin{aligned}
\nabla_{\mathbf{w}}\mathsf{L} &= \frac{1}{2}\nabla_{\mathbf{w}}\left((\mathbf{y}-\mathbf{X}\mathbf{w})^T(\mathbf{y}-\mathbf{X}\mathbf{w})\right) \\
&= \frac{1}{2}\nabla_{\mathbf{w}}(\mathbf{y}^T\mathbf{y}-\mathbf{y}^T\mathbf{X}\mathbf{w}-\mathbf{w}^T\mathbf{X}^T\mathbf{y}+\mathbf{w}^T\mathbf{X}^T\mathbf{X}\mathbf{w}) \\
&= \frac{1}{2}\nabla_{\mathbf{w}}(\mathbf{w}^T\mathbf{X}^T\mathbf{X}\mathbf{w}-2\mathbf{y}^T\mathbf{X}\mathbf{w}) \\
&= \mathbf{X}^T\mathbf{X}\mathbf{w}-\mathbf{X}^T\mathbf{y}
\end{aligned}$$

Analytic Optimisation

- And set it equal to zero:

$$\mathbf{X}^T\mathbf{X}\mathbf{w}_{\text{OLS}} - \mathbf{X}^T\mathbf{y} = 0$$
$$\mathbf{X}^T\mathbf{X}\mathbf{w}_{\text{OLS}} = \mathbf{X}^T\mathbf{y}$$

- This is a system of linear equations (again)...how should we proceed?

Analytic Optimisation

- First note the following important fact:

$$\text{rank}(\mathbf{X}^T\mathbf{X}) = \text{rank}(\mathbf{X}^T\mathbf{X}|\mathbf{X}^T\mathbf{y})$$

- So the system is **consistent** and we have one or infinitely many solutions...

- Let's consider two cases:

Case 1: $\mathbf{X}^T\mathbf{X}$ is Invertible

- If $(\mathbf{X}^T\mathbf{X})^{-1}$ then we may solve the system to achieve the following, **unique** solution:

$$\mathbf{w}_{\text{OLS}} = (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{y}$$

- These are the **Normal Equations** which give an **analytic solution** to the OLS approach to linear regression

Case 2: $\mathbf{X}^T\mathbf{X}$ is Not Invertible

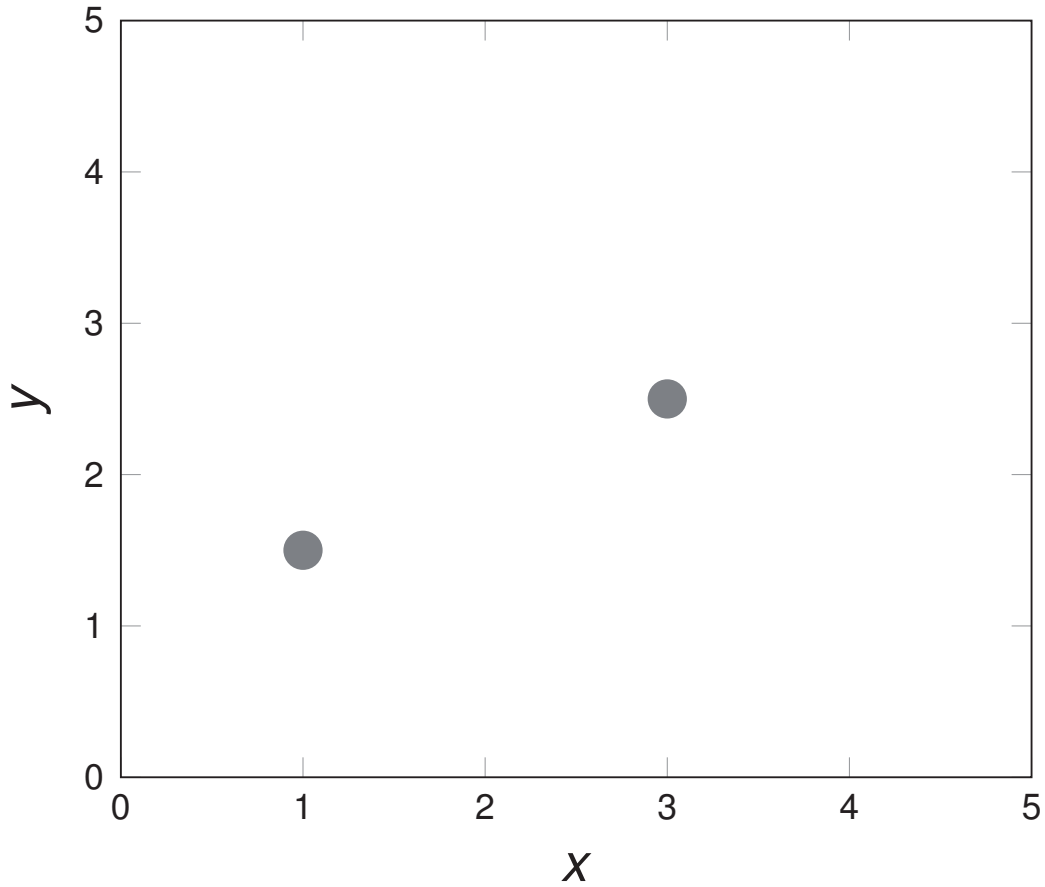
- If $(\mathbf{X}^T\mathbf{X})^{-1}$ does not exist then $\text{rank}(\mathbf{X}^T\mathbf{X}) < (m+1)$
- So there are too few equations to fully specify the number of unknowns and our problem is underdetermined
- And since we know that our system is consistent (and so solutions exist) then this must mean that there is no unique solution, instead there are an infinite number

Case 2: Overfitting

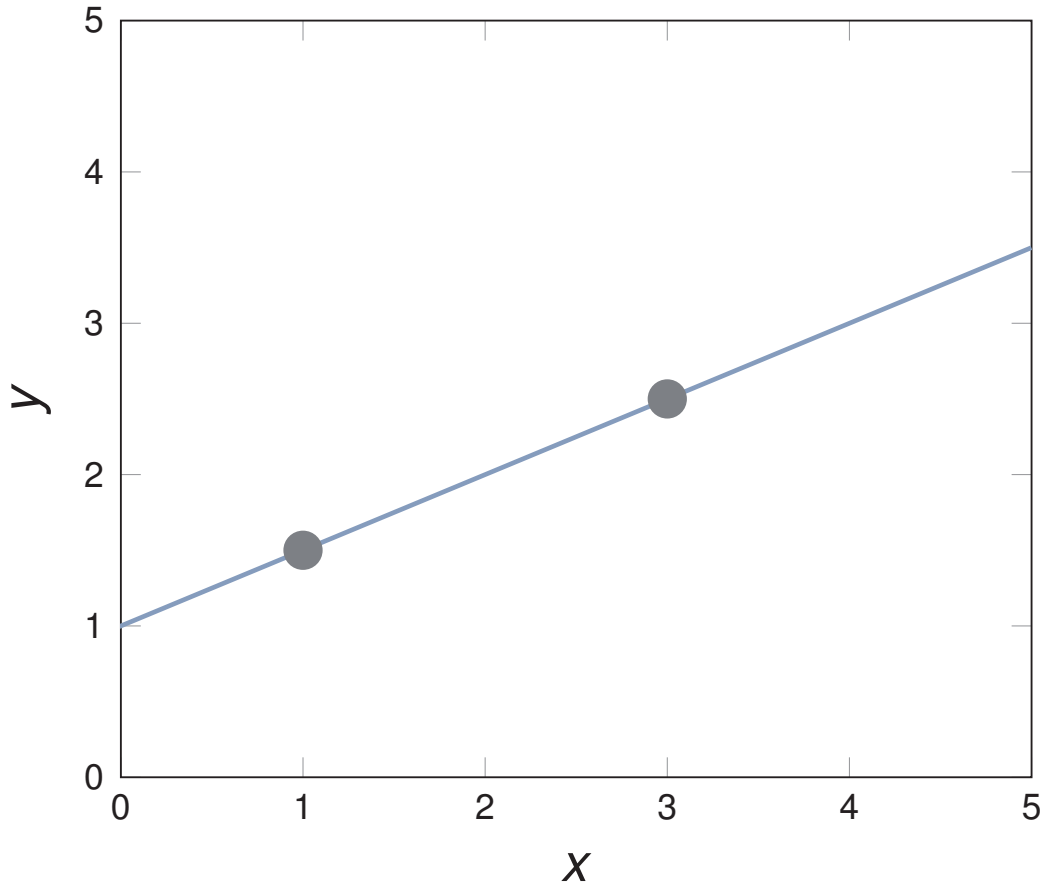
- When does this occur?
- For example, consider fitting quadratic functions to just 2 data points:

$$\begin{bmatrix} y^{(1)} \\ y^{(2)} \end{bmatrix} = \begin{bmatrix} 1 & x^{(1)} & x^{(1)2} \\ 1 & x^{(2)} & x^{(2)2} \end{bmatrix} \begin{bmatrix} w_0 \\ w_1 \\ w_2 \end{bmatrix}$$

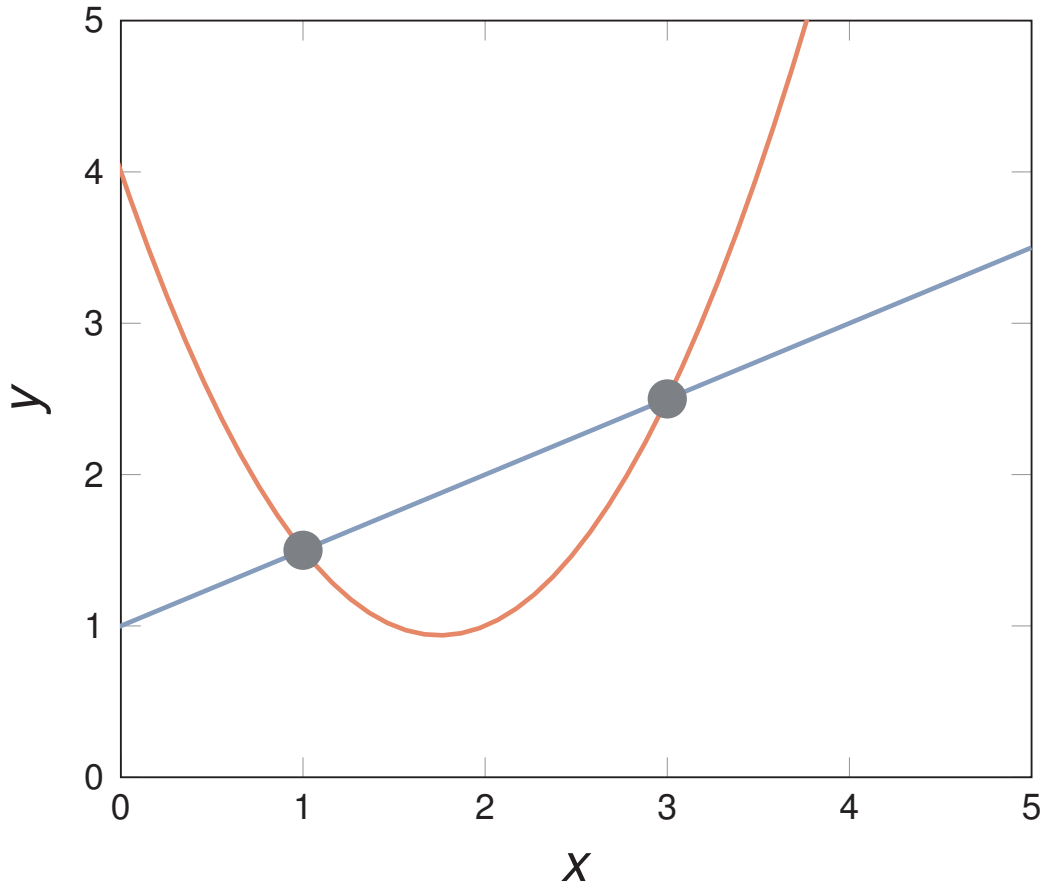
Case 2: Overfitting



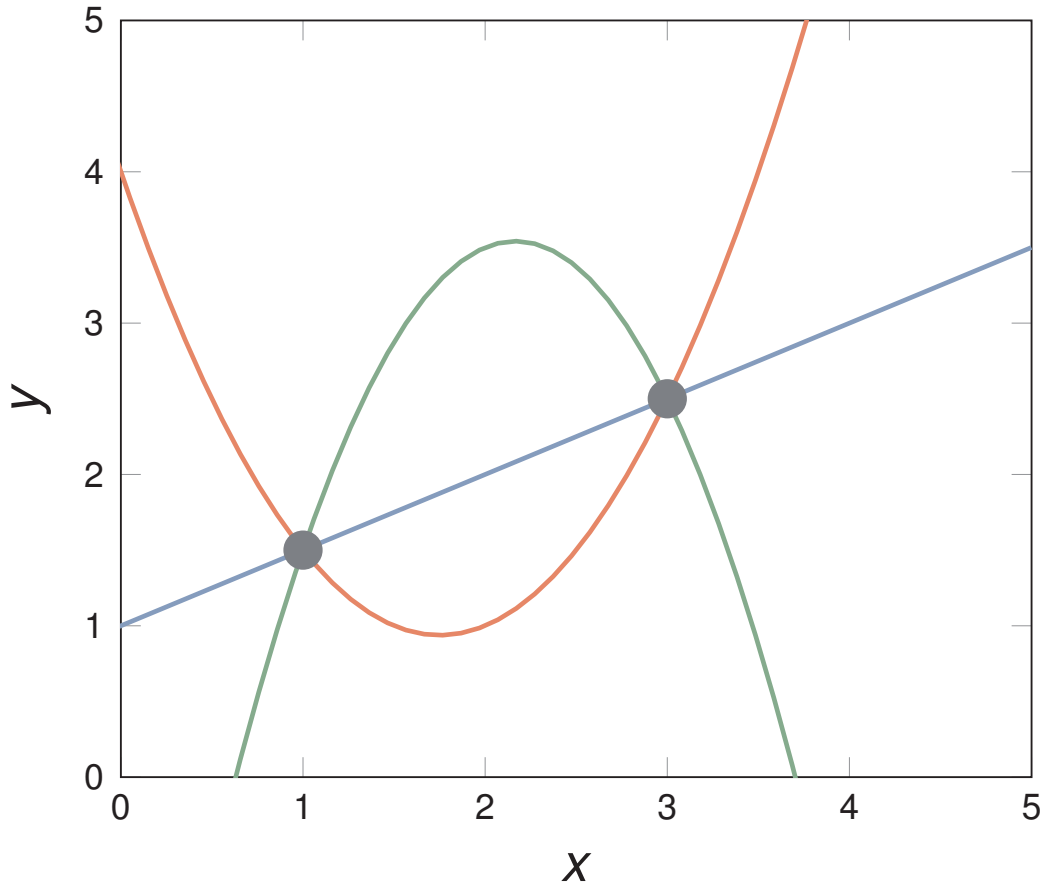
Case 2: Overfitting



Case 2: Overfitting



Case 2: Overfitting



Case 2: Overfitting

- There are infinite solutions...
- ...We have encountered a case of **overfitting**
- We will return to consider this problem and its remedy in later lectures...

Convexity

- Recall that in our optimisation we needed to check for convexity, let's do that now and check the Hessian:
- Note:

$$\frac{\partial^2 \mathsf{L}}{\partial w_i \, \partial w_j} = (\mathbf{X}^T \mathbf{X})_{ij}$$

- So:

$$\mathcal{H} = \nabla^2_{\mathbf{w}} \mathsf{L} = \mathbf{X}^T \mathbf{X}$$

Convexity

- Now, for any $\mathbf{a} \in \mathbb{R}^{m+1}$

$$\mathbf{a}^T \mathbf{X}^T \mathbf{X} \mathbf{a} = \|\mathbf{X}\mathbf{a}\|_2^2 \geqslant 0$$

- Therefore $\nabla_{\mathbf{w}}{}^2 \mathrm{L} \succeq 0$
- Therefore L is **convex** [*Theorem A.1*]
- Therefore our solution is **globally optimal** [*Theorem A.2*]
- ...But not necessarily unique
- Let's consider two cases again:

Case 1: $\mathbf{X}^T\mathbf{X}$ is Invertible

- Recall from our work on linear algebra that:

$$\exists(\mathbf{X}^T\mathbf{X})^{-1} \iff \mathbf{X}^T\mathbf{X} \succ 0$$

- But if $\mathbf{X}^T\mathbf{X} \succ 0$ then our objective is **strictly convex**

- And if our objective is strictly convex then our solution must be **unique** [*Theorem A.3*]

Case 2: $\mathbf{X}^T\mathbf{X}$ is Not Invertible

- Recall from our work on linear algebra that:

$$\exists(\mathbf{X}^T\mathbf{X})^{-1} \iff \mathbf{X}^T\mathbf{X} \succ 0$$

- So in this case $\mathbf{X}^T\mathbf{X} \not\succ 0$
- Now, recall our work on the optimisation of quadratic functions in the *Calculus* lecture. In particular, if $\mathbf{X}^T\mathbf{X} \not\succ 0$ then:
  - Our objective is not **strictly convex** (although it is **bounded** since $\mathbf{X}^T\mathbf{y} \in \text{range}(\mathbf{X}^T\mathbf{X})$)
  - So we will have an infinite number of solutions
- We will return to these considerations later in the module, but for now we note the symmetry between **invertibility** and **strict convexity**

Normal Equations: Drawbacks

- The normal equations are elegant, but for large m they have some drawbacks:
- **Time Constraints**:
  - Matrix inversion takes $\mathcal{O}(m^3)$ operations - for large $\mathbf{X}^T\mathbf{X}$ this is expensive
- **Space Constraints**:
  - For very high m, $\mathbf{X}^T\mathbf{X}$ will not fit into memory
- Is there an alternative?

Numerical Optimisation

- There are several different approaches to **numerical optimisation**. We will investigate one of the most common:

- **Gradient Descent**, a **first order**, **iterative**, optimisation algorithm

Gradient Descent

- Takes steps proportional to the negative of the gradient at each step point...
- ...Because the gradient of a function is in the direction of steepest descent at that point
  - Magnitude of descent in a general direction, $\widehat{\mathbf{u}}$, is:

$$\nabla_{\mathbf{w}} L \cdot \widehat{\mathbf{u}} = \|\nabla_{\mathbf{w}} L\|_2 \, \|\widehat{\mathbf{u}}\|_2 \cos\theta$$

  - This is maximal when $\theta = 0$ and $\nabla_{\mathbf{w}} L$ and $\widehat{\mathbf{u}}$ lie in the same direction
- Makes monotonic progress (subject to step size)
- Stops when the gradient is zero

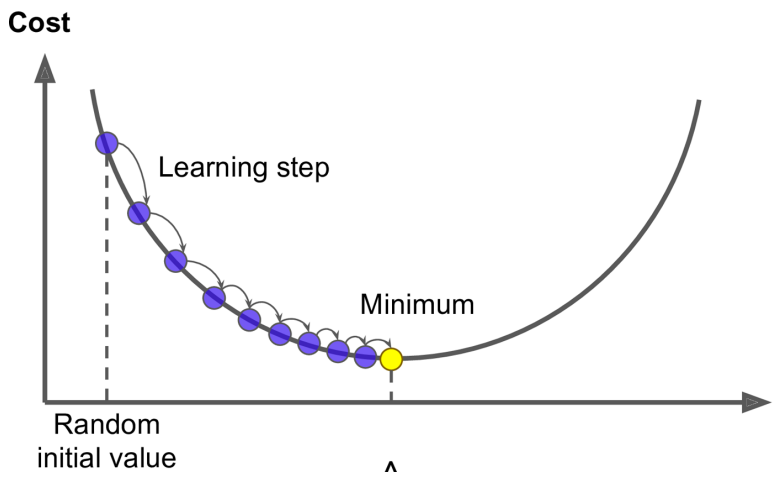
Batch Gradient Descent

Algorithm 1 Batch Gradient Descent

Set $\alpha > 0$
Initialise $t \leftarrow 0$
Initialise $\mathbf{w}_{(t=0)}$ randomly
repeat
    $\mathbf{w}_{(t+1)} \leftarrow \mathbf{w}_{(t)} - \alpha \nabla_{\mathbf{w}} L|_{\mathbf{w}=\mathbf{w}_{(t)}}$
    $t \leftarrow t + 1$
until convergence

- Here $\alpha > 0$ is the **learning rate** which determines the step size

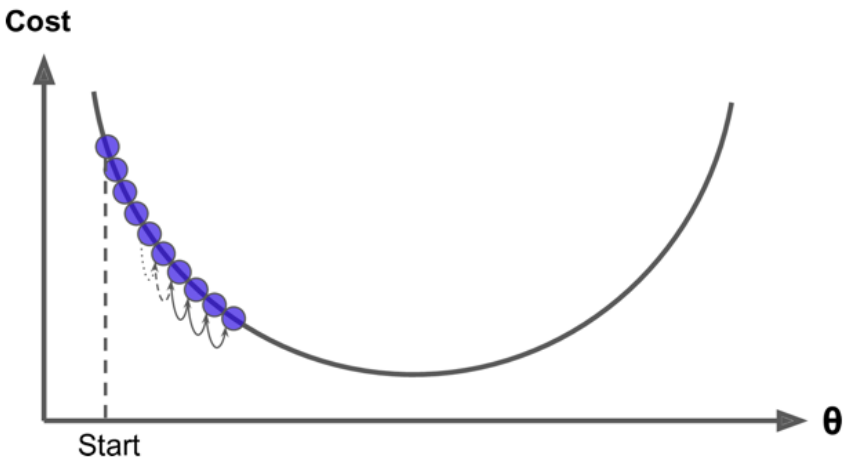
Batch Gradient Descent in 1 Dimension[2]

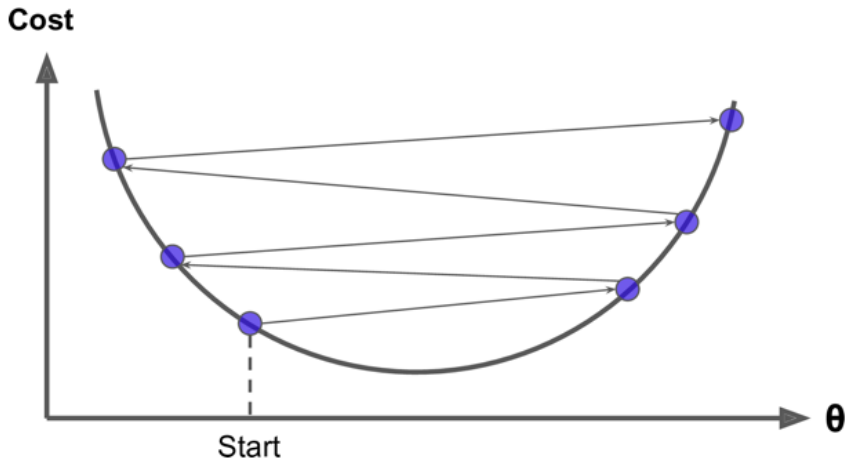


- Update is large when the gradient is large
- As the optimum is approached steps get smaller

[2] Geron, 'Hands-on Machine Learning With Scikit-Learn & Tensorflow' [2017]

Batch Gradient Descent: Learning Rate Size[3]





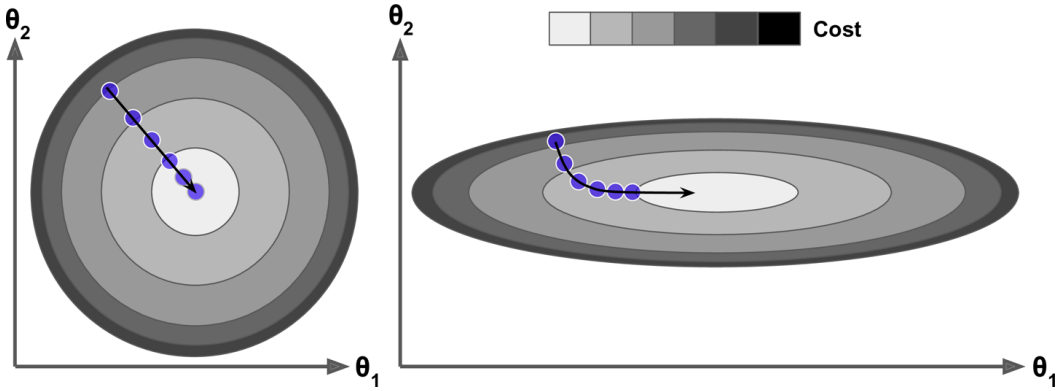
- For α too small, convergence is **slow**
- For α too large, we get **divergence**

[3] Geron, 'Hands-on Machine Learning With Scikit-Learn & Tensorflow' [2017]

Batch Gradient Descent: Convexity

- Gradient Descent is guaranteed to converge to a **local** optimum should one exist (for suitable α)...
- ...But not necessarily a **global** one
- If the objective is **non-convex** then GD may not find the global optimum
- But for OLS recall that $\nabla^2_{\mathbf{w}} \mathrm{L} \succeq 0$, which implies convexity [*Theorem A.1*], (and recall that $\mathbf{X}^T\mathbf{y} \in \text{range}(\mathbf{X}^T\mathbf{X})$, which implies that the objective is bounded below)...
- ...So GD will find a global optimum... [*Theorem A.2*]
- ...Although, for OLS, recall that unless $\nabla^2_{\mathbf{w}} \mathrm{L} \succ 0$ then such an optimum will not be **unique**

Batch Gradient Descent: Scaling[4]



- For **unscaled** attributes the gradient of the largest parameter will dominate the update leading to slow convergence
- For **scaled** attributes all parameters are updated in similar proportions

[4] Geron, 'Hands-on Machine Learning With Scikit-Learn & Tensorflow' [2017]

Batch Gradient Descent: Iteration Speed

Algorithm 2 Batch Gradient Descent for OLS

Set $\alpha > 0$
Initialise $t \leftarrow 0$
Initialise $\mathbf{w}_{(t=0)}$ randomly
repeat
    for $j = 0$ to m **do**
        $[\mathbf{w}_{(t+1)}]_j \leftarrow [\mathbf{w}_{(t)}]_j - \alpha \sum_{i=1}^{n} (\mathbf{w}_{(t)} \cdot \mathbf{x}^{(i)} - y^{(i)}) [\mathbf{x}^{(i)}]_j$
    end for
    $t \leftarrow t + 1$
until convergence

- For each update we need to scan through the entire training set
- For large n this can be costly

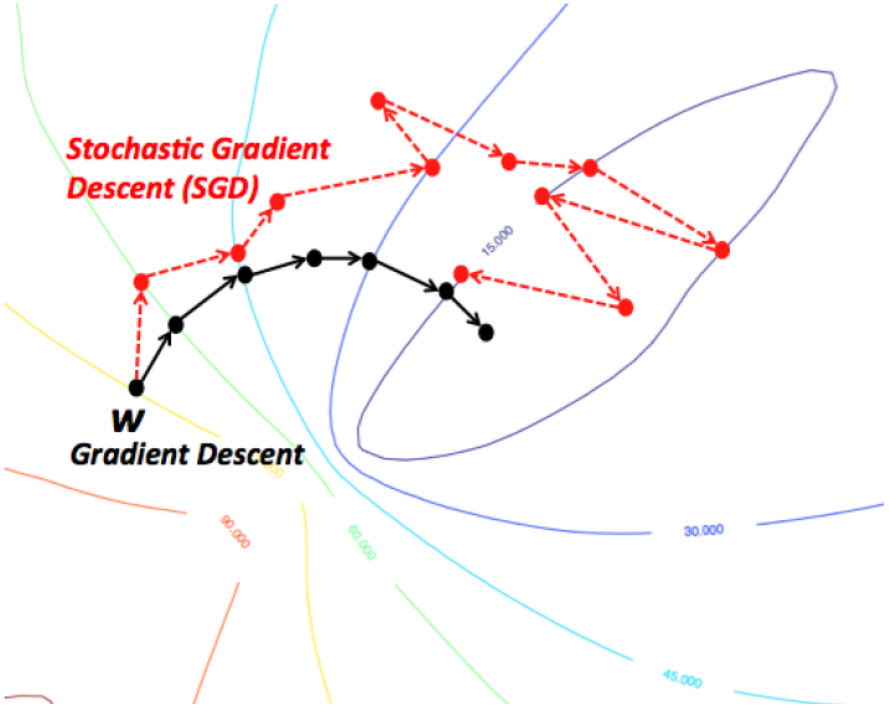
Stochastic Gradient Descent

Algorithm 3 Stochastic Gradient Descent for OLS

Set $\alpha > 0$
Initialise $t \leftarrow 0$
Initialise $\mathbf{w}_{(t=0)}$ randomly
repeat
    Randomly shuffle dataset:
    Initialise $i \leftarrow 1$
    repeat
        for $j = 0$ to m **do**
            $[\mathbf{w}_{(t+1)}]_j \leftarrow [\mathbf{w}_{(t)}]_j - \alpha(\mathbf{w}_{(t)} \cdot \mathbf{x}^{(i)} - y^{(i)})[\mathbf{x}^{(i)}]_j$
        end for
        $t \leftarrow t + 1$
        $i \leftarrow i + 1$
    until $i = n + 1$
until convergence

- Here an update is made immediately as each example is examined

Stochastic Gradient Descent



- Each individual SGD update is **less optimal** than a BGD update...
- ...And the algorithm will not converge **monotonically**...
- ...But overall **convergence** (for convex objectives) is generally **faster**, and scales better for large n

Analytic & Numerical Solutions: Symmetry

1. In general, for OLS:
   - $\text{rank}(\mathbf{X}^T\mathbf{X}) = \text{rank}(\mathbf{X}^T\mathbf{X}|\mathbf{X}^T\mathbf{y})$, so the system is **consistent** and we have a solution of some sort
   - $\mathbf{X}^T\mathbf{X} \succeq 0$, so the system is **convex**, and $\mathbf{X}^T\mathbf{y} \in \text{range}(\mathbf{X}^T\mathbf{X})$, so the system is bounded, thus gradient descent will converge to a global solution of some sort

2. If $(\mathbf{X}^T\mathbf{X})^{-1}$ exists:
   - The normal equations will work, $\text{rank}(\mathbf{X}^T\mathbf{X}) = (m+1)$, and we have a **unique** solution
   - $\mathbf{X}^T\mathbf{X} \succ 0$ so our objective is **strictly convex** and gradient descent will converge to a **unique** solution

3. If $(\mathbf{X}^T\mathbf{X})^{-1}$ does not exist:
   - The normal equations will not work, $\text{rank}(\mathbf{X}^T\mathbf{X}) < (m+1)$, and we have an **infinite** number of solutions
   - $\mathbf{X}^T\mathbf{X} \not\succ 0$, our objective is **not strictly convex** and gradient descent will converge to one of the **infinite** number of solutions

Lecture Overview

Summary

1. **Linear Regression** can be used to estimate a linear relationship that might exist between our input and our output data
2. **Ordinary Least Squares** is a well-motivated approach to linear regression which leads to a smooth, convex, optimisation problem
3. We can use the **normal equations** or various **gradient descent** procedures to solve the OLS optimisation problem and to learn the parameters of the linear hypothesis

In the next lecture we will consider extending our representation to capture richer relationships via: **polynomial regression**.

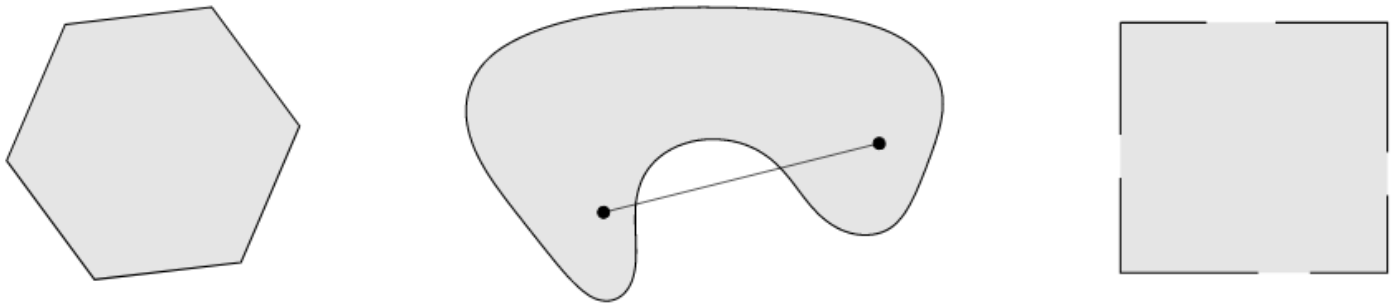
Lecture Overview

Convex Sets[5]

- **Definition:**

  A set Ω is **convex** if, for any $\mathbf{x}, \mathbf{y} \in \Omega$ and $\theta \in [0, 1]$, then $\theta\mathbf{x} + (1 - \theta)\mathbf{y} \in \Omega$.

  In other words, if we take any two elements in Ω, and draw a line segment between these two elements, then every point on that line segment also belongs to Ω.



[5] Boyd & Vandenberghe, 'Convex Optimisation' [2004]

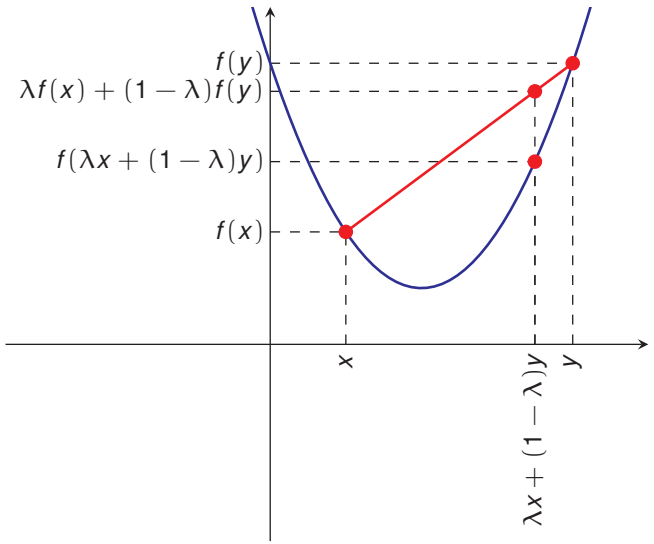
Convex Functions

- **Definition:**

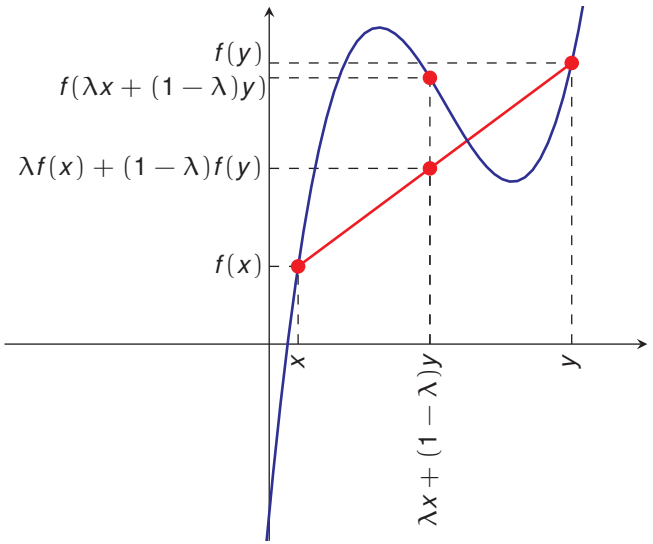
  A function $f : \mathbb{R}^n \to \mathbb{R}$ is **convex** if its domain is a **convex set** and if, for all **x**, **y** in its domain, and all $\lambda \in [0, 1]$, we have:

$$f(\lambda \mathbf{x} + (1 - \lambda)\mathbf{y}) \leqslant \lambda f(\mathbf{x}) + (1 - \lambda) f(\mathbf{y})$$

Convex



Non-Convex



1st & 2nd Order Characterisations of Convex, Differentiable Functions

- **Theorem A.1:**

  Suppose f is twice differentiable over an open domain. Then, the following are equivalent:

  - f is convex
  - $f(\mathbf{y}) \geqslant f(\mathbf{x}) + \nabla_{\mathbf{x}} f(\mathbf{x}) \cdot (\mathbf{y} - \mathbf{x}) \qquad \forall \quad \mathbf{x}, \mathbf{y} \in \mathrm{dom}(f)$
  - $\nabla^2_{\mathbf{x}} f(\mathbf{x}) \succeq 0 \qquad \forall \quad \mathbf{x} \in \mathrm{dom}(f)$

Global Optimality

- **Theorem A.2:**

  Consider an unconstrained optimisation problem:

$$\begin{aligned} \min_{\mathbf{x}} \quad & f(\mathbf{x}) \\ \text{subject to:} \quad & \mathbf{x} \in \mathbb{R}^n \end{aligned}$$

  Where $f : \mathbb{R}^n \to \mathbb{R}$ is **convex** and **differentiable**.

  Then any point $\mathbf{x}$ that satisfies $\nabla_{\mathbf{x}} f(\mathbf{x}) = 0$ is a **globally optimal** solution

Strict Convexity

- **Definition:**

  A function $f : \mathbb{R}^n \to \mathbb{R}$ is **strictly convex** if its domain is a **convex set** and if, for all $\mathbf{x}, \mathbf{y}, \mathbf{x} \neq \mathbf{y}$ in its domain, and all $\lambda \in (0, 1)$, we have:

  $$f(\lambda\mathbf{x} + (1-\lambda)\mathbf{y}) < \lambda f(\mathbf{x}) + (1-\lambda) f(\mathbf{y})$$

- **First Order Characterisation:**

  A function f is **strictly convex** on $\Omega \subseteq \mathbb{R}^n$, if and only if:

  $$f(\mathbf{y}) > f(\mathbf{x}) + \nabla_{\mathbf{x}} f(\mathbf{x}) \cdot (\mathbf{y} - \mathbf{x}) \qquad \forall \mathbf{x}, \mathbf{y} \in \Omega, \qquad \mathbf{x} \neq \mathbf{y}$$

- **Second Order Sufficient Condition:**

  A function f is **strictly convex** on $\Omega \subseteq \mathbb{R}^n$, if:

  $$\nabla^2_{\mathbf{x}} f(\mathbf{x}) \succ 0 \qquad \forall \mathbf{x} \in \Omega$$

Strict Convexity and Uniqueness of Optimal Solutions

- **Theorem A.3:**

  Consider an optimisation problem:

$$\begin{aligned} \min_{\mathbf{x}} \quad & f(\mathbf{x}) \\ \text{subject to:} \quad & \mathbf{x} \in \Omega \end{aligned}$$

  Where $f : \mathbb{R}^n \to \mathbb{R}$ is **strictly convex** on Ω and Ω is a **convex set**.

  Then the **optimal solution** must be **unique**