Introduction to Optimisation

Dr Jamie A Ward

Lecture 4

Lecture 4

Univariate Linear Regression

- The loss function

Introduction to Optimisation

- Derivatives and critical points
- Gradient descent with one variable

Multidimensional Gradient Descent

- Gradient descent with N variables
- Gradient descent and Linear Regression

CO2 Prediction

Question:

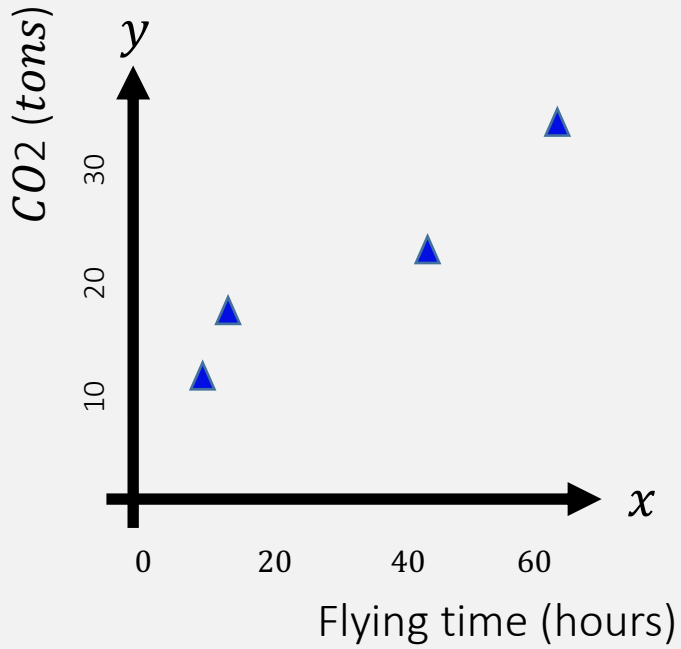
How can we predict CO2 for other flying times?

Answer:

Supervised Machine Learning

- Model as a regression (real valued output)

Fly (h)	CO2 (tons)
16	17.8
8	10.3
45	23.0
65	30.8
x	y



CO2 Prediction

Question:

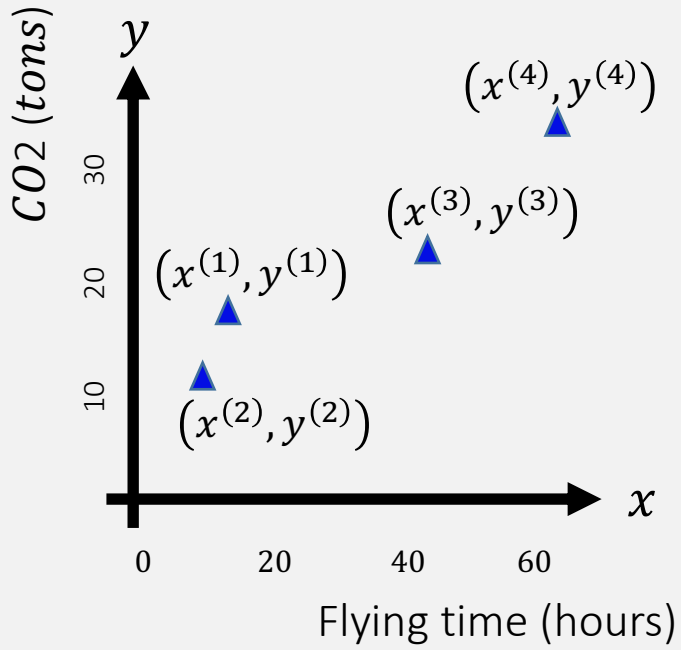
How can we predict CO2 for other flying times?

Answer:

Supervised Machine Learning

- Model as a regression (real valued output)

	Fly (h) x	CO2 (tons) y
(1)	16	17.8
(2)	8	10.3
(3)	45	23.0
(4)	65	30.8



Notation

x : input features, y: output / target values

m: number of training examples

$(x^{(i)}, y^{(i)})$: i^{th} training example

$\{(x^{(1)}, y^{(1)}), \ldots, (x^{(m)}, y^{(m)})\}$: training set

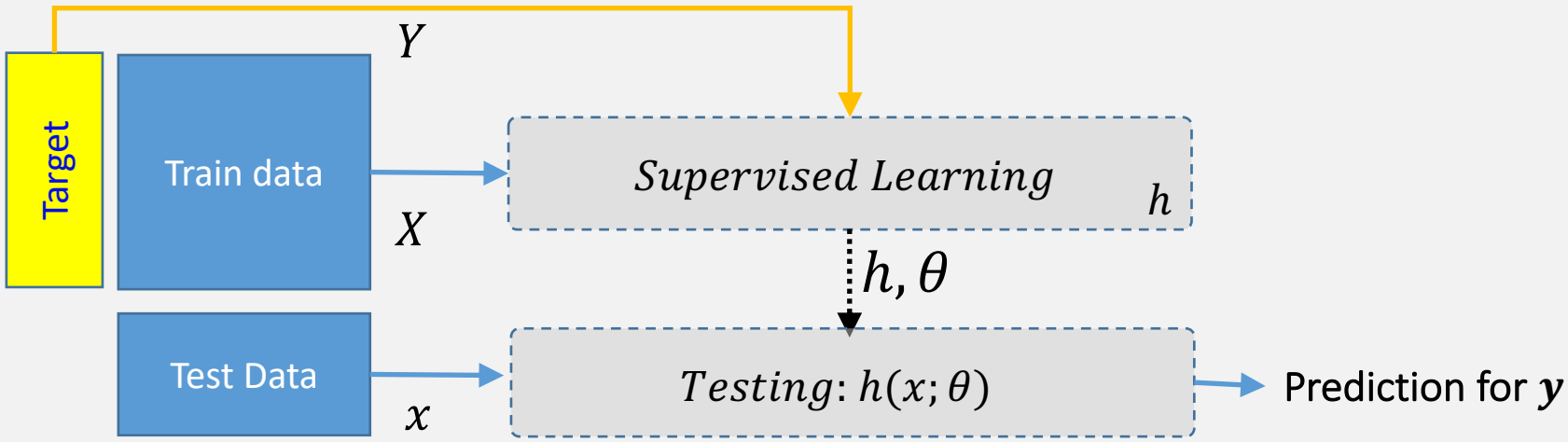
Supervised learning revisited

Goal: given a training set, learn a function h such that h is a good predictor for the corresponding value of (target) y

- $h(x) \rightarrow y$

What do we mean by learn a function?

- Learn the parameters θ that define a function
- So actually, $y = h(x;\theta) = h_\theta(x)$
- Where $h(x;\theta)$ reads "function h applied on x given parameters θ"



Regression analysis

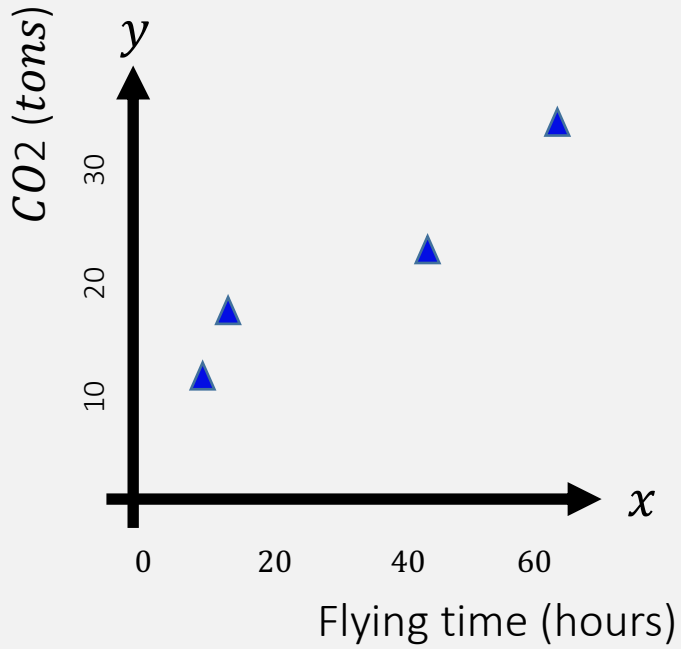
Find relationships between a dependent variable and *one or more* independent variables.

Questions:

Which variable is dependent?

Which is the independent?

Fly (h)	CO2 (tons)
16	17.8
8	10.3
45	23.0
65	30.8
x	y



Regression analysis

Find relationships between a dependent variable and *one or more* independent variables.

Questions:

Which variable is dependent?

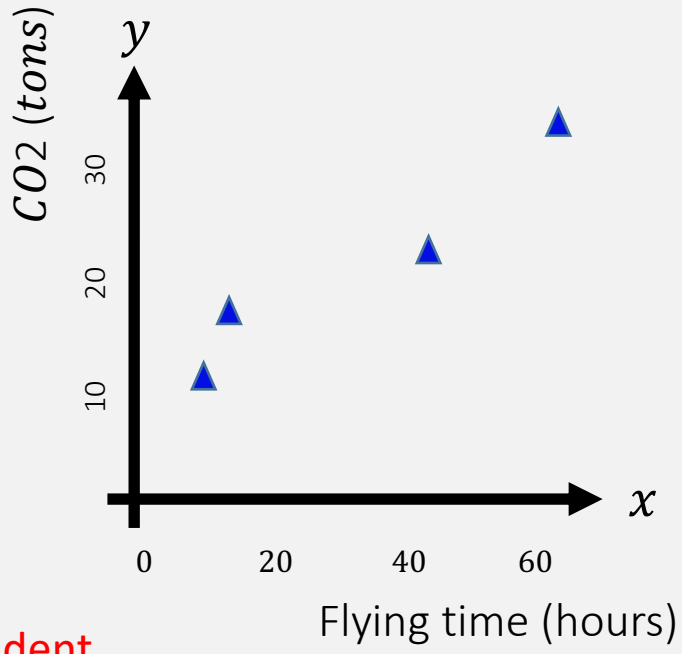
Which is the independent?

Fly (h)	CO2 (tons)
16	17.8
8	10.3
45	23.0
65	30.8

x y

independent (explanatory variable)

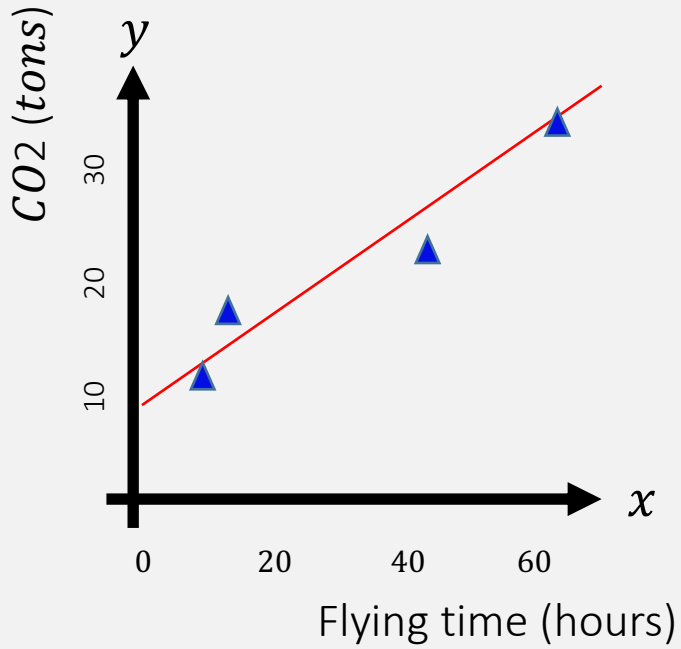
dependent (it depends on x)



Linear regression

Goal: given a training set, learn a function h such that h is a good predictor for the corresponding value of (target) y

- $h(x) \rightarrow y$



How would h look for linear regression?

It would look like a linear function of x (i.e. a line!):

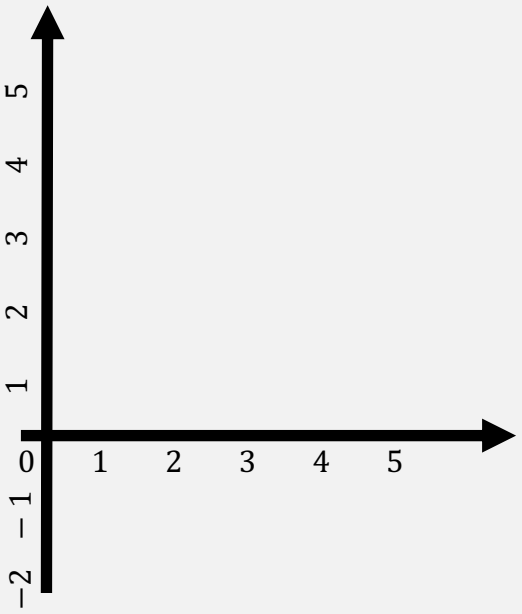
$$h(x) = \theta_0 + \theta_1 x$$

Again, $h(x)$ is an abbreviation of

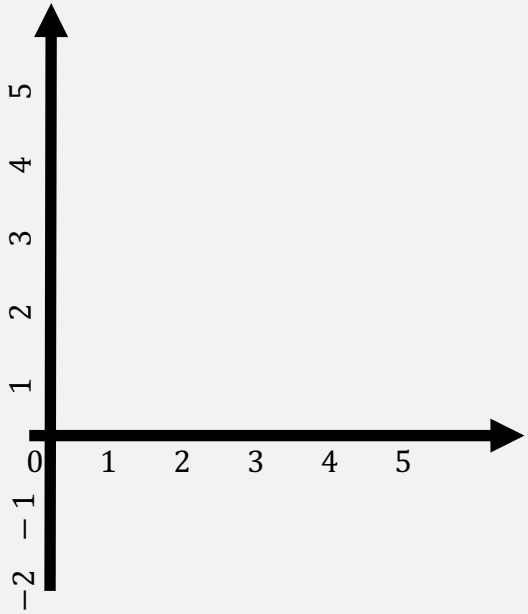
$$h(x;\theta) = h_\theta(x) = \theta_0 + \theta_1 x$$
$$where\ \theta = \{\theta_0, \theta_1\}$$

Draw the hypothesis function

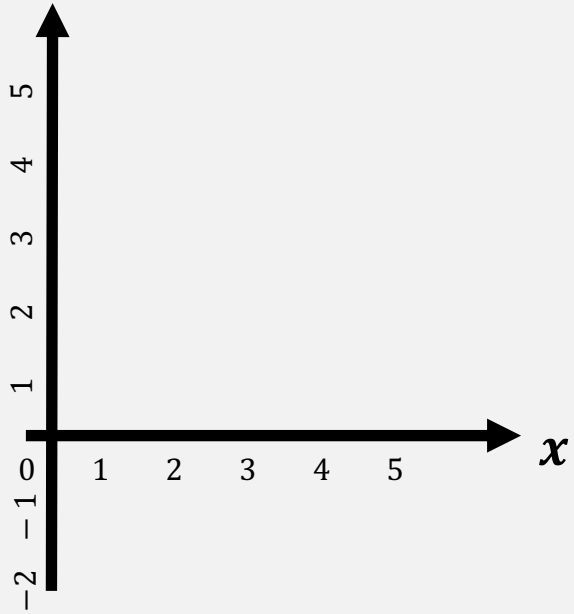
$$h(x;\theta) = \theta_0 + \theta_1 x$$



$$\begin{bmatrix}\theta_0\\ \theta_1\end{bmatrix} = \begin{bmatrix}3\\ 0\end{bmatrix}$$



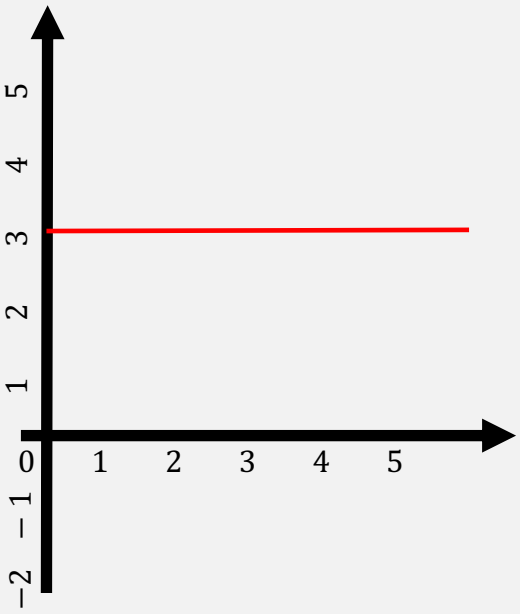
$$\begin{bmatrix}\theta_0\\ \theta_1\end{bmatrix} = \begin{bmatrix}0\\ 2\end{bmatrix}$$



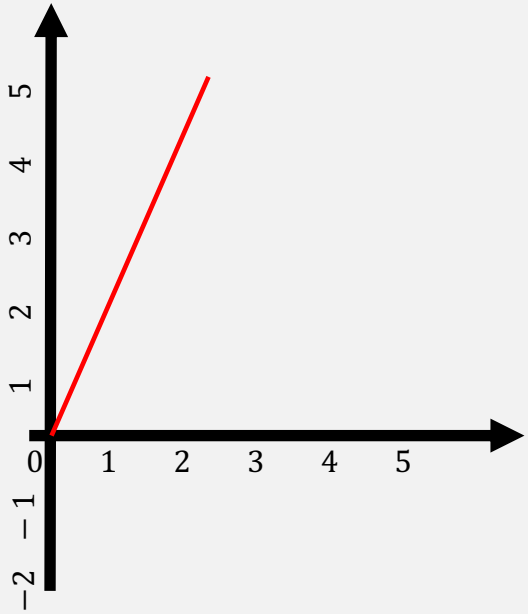
$$\begin{bmatrix}\theta_0\\ \theta_1\end{bmatrix} = \begin{bmatrix}-1\\ 0.5\end{bmatrix}$$

Draw the hypothesis function

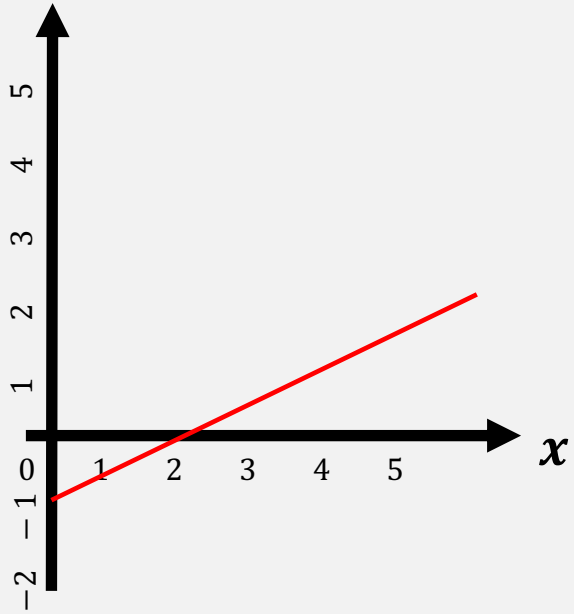
$$h(x;\theta) = \theta_0 + \theta_1 x$$



$$\begin{bmatrix}\theta_0\\ \theta_1\end{bmatrix} = \begin{bmatrix}3\\ 0\end{bmatrix}$$



$$\begin{bmatrix}\theta_0\\ \theta_1\end{bmatrix} = \begin{bmatrix}0\\ 2\end{bmatrix}$$

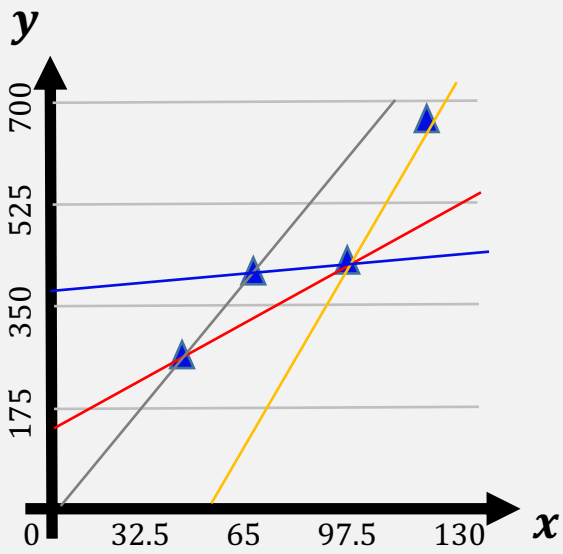


$$\begin{bmatrix}\theta_0\\ \theta_1\end{bmatrix} = \begin{bmatrix}-1\\ 0.5\end{bmatrix}$$

Find the best line

Hypothesis function
$h(x;\theta) = \theta_0 + \theta_1 x$

Challenge: use the training data to find the parameters (weights) $\theta = \{\theta_0, \theta_1\}$ of h.
(i.e. what's the best line?)



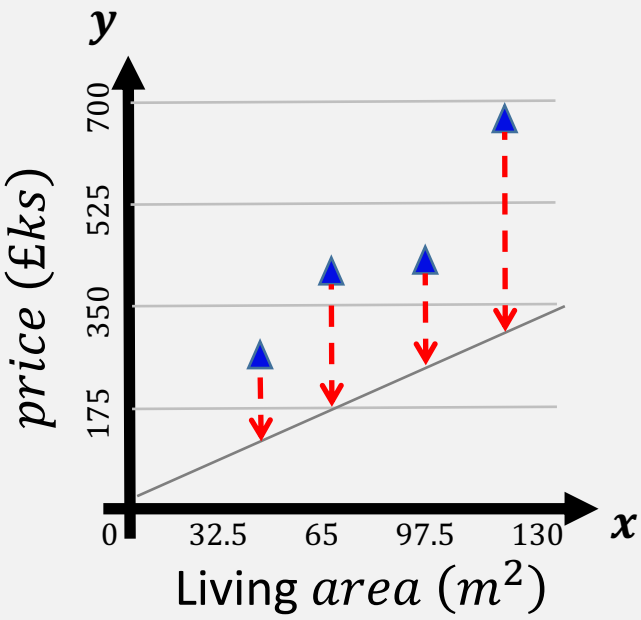
Think about how we found the line using 2 data points (from previous lectures)

- Data is noisy (not all on a perfect line)!
- What is the next best solution?

Find the best line

Iterative Solution:

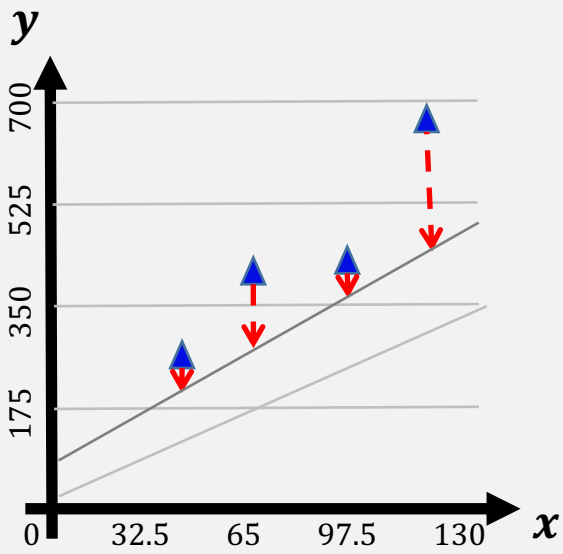
- Start with a (random) line
- Measure error or loss between line and data



Find the best line

Iterative Solution:

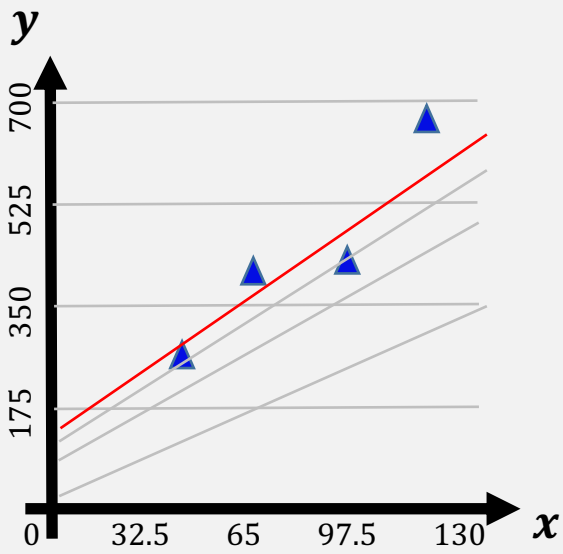
- Start with a (random) line
- Measure error or loss between line and data
- *Move line to try and lower the loss*



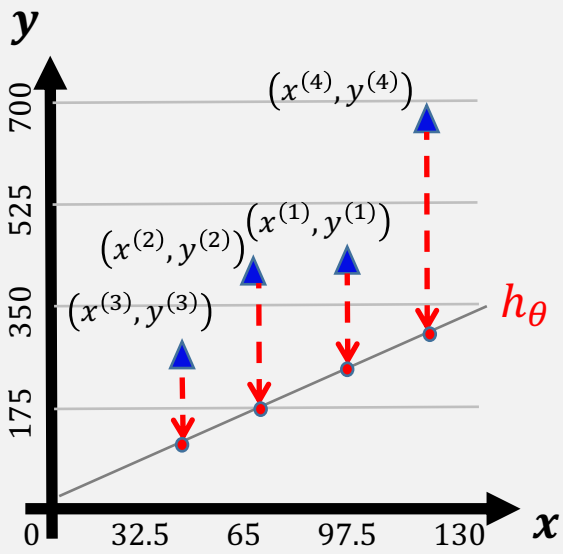
Find the best line

Iterative Solution:

- Start with a (random) line
- Measure error or loss between line and data
- *Move line to try and lower the loss*
- Loop until best line found (loss minimised)



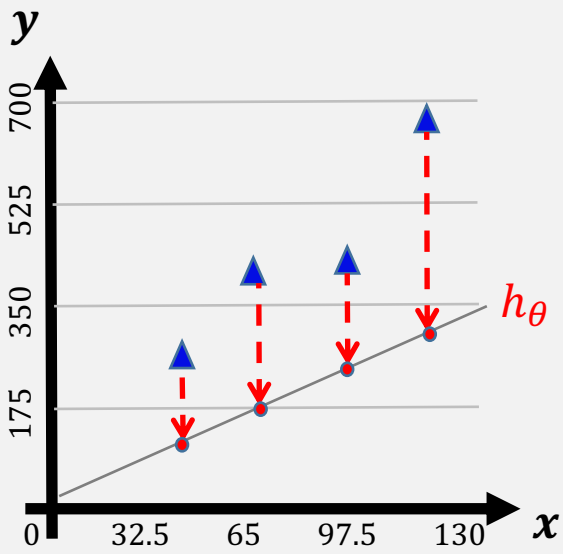
Mean Squared Error (the Loss function)



Given training data, $\{(x^{(1)}, y^{(1)}), \ldots, (x^{(m)}, y^{(m)})\}$, and regression model outputs, $h(x^{(1)}), \ldots, h(x^{(m)})$, the loss, or error, can be measured as:

$$J(\theta) = \frac{1}{2m} \sum_{i=1}^{m} \left(h(x^{(i)}; \theta) - y^{(i)} \right)^2$$

Mean Squared Error (the Loss function)



Given training data, $\{(x^{(1)}, y^{(1)}), \ldots, (x^{(m)}, y^{(m)})\}$, and regression model outputs, $h(x^{(1)}), \ldots, h(x^{(m)})$, the loss, or error, can be measured as:

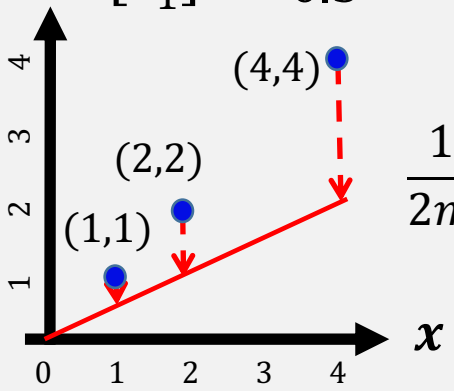
$$J(\theta) = \frac{1}{2m} \sum_{i=1}^{m} \left(h(x^{(i)}; \theta) - y^{(i)} \right)^2$$

Averaging term (also 1/2 makes calculations easier later on)

L2 Loss function

$$J(\theta) = \frac{1}{2m}\sum_{i=1}^{m}\left(h\left(x^{(i)};\theta\right) - y^{(i)}\right)^2$$

$$\begin{bmatrix}\theta_0\\ \theta_1\end{bmatrix} = \begin{bmatrix}0\\ 0.5\end{bmatrix} \qquad m=3$$



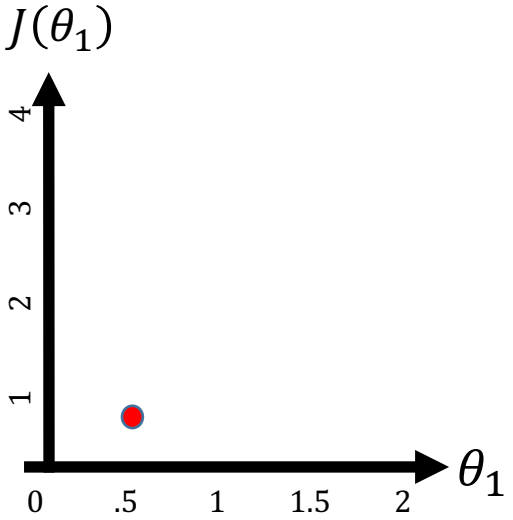
$$J\left(\theta = \begin{bmatrix}0\\ 0.5\end{bmatrix}\right) =$$

$$\frac{1}{2m}\left((0.5-1)^2 + (1-2)^2 + (2-4)^2\right)$$

$$= \frac{1}{6}\left((-0.5)^2 + (-1)^2 + (2)^2\right)$$

$$= \frac{1}{6}(0.25 + 1 + 4)$$

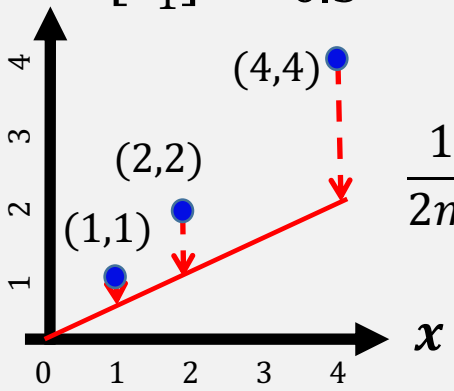
$$= 0.875$$



L2 Loss function

$$J(\theta) = \frac{1}{2m}\sum_{i=1}^{m}\left(h\left(x^{(i)};\theta\right) - y^{(i)}\right)^2$$

$$\begin{bmatrix}\theta_0\\ \theta_1\end{bmatrix} = \begin{bmatrix}0\\ 0.5\end{bmatrix} \qquad m=3$$



$$J\left(\theta = \begin{bmatrix}0\\ 0.5\end{bmatrix}\right) =$$

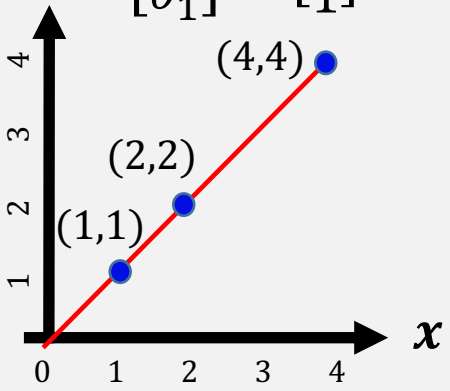
$$\frac{1}{2m}\left((0.5-1)^2 + (1-2)^2 + (2-4)^2\right)$$

$$= \frac{1}{6}\left((-0.5)^2 + (-1)^2 + (2)^2\right)$$

$$= \frac{1}{6}(0.25 + 1 + 4)$$

$$= 0.875$$

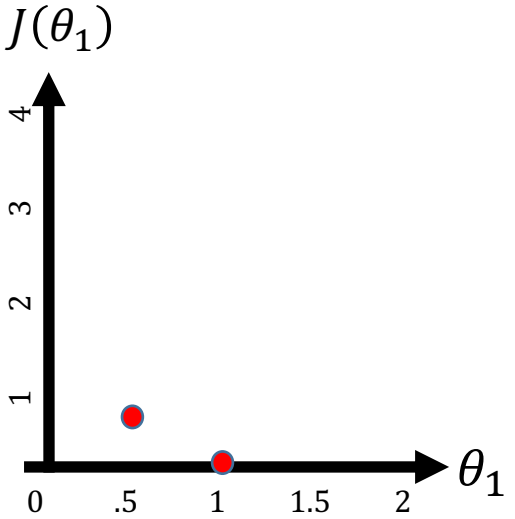
$$\begin{bmatrix}\theta_0\\ \theta_1\end{bmatrix} = \begin{bmatrix}0\\ 1\end{bmatrix}$$



$$J\left(\theta = \begin{bmatrix}0\\ 1\end{bmatrix}\right) =$$

$$\frac{1}{6}\left((1-1)^2 + (2-2)^2 + (4-4)^2\right)$$

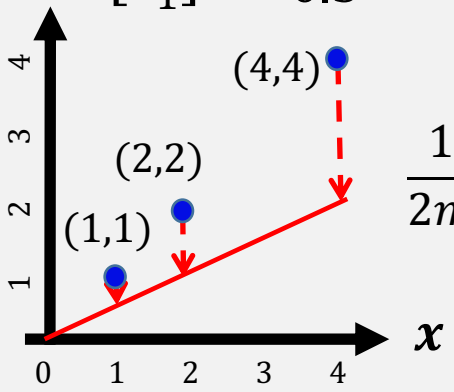
$$= 0$$



L2 Loss function

$$J(\theta) = \frac{1}{2m}\sum_{i=1}^{m}\left(h\left(x^{(i)};\theta\right) - y^{(i)}\right)^2$$

$$\begin{bmatrix}\theta_0\\ \theta_1\end{bmatrix} = \begin{bmatrix}0\\ 0.5\end{bmatrix} \qquad m=3$$



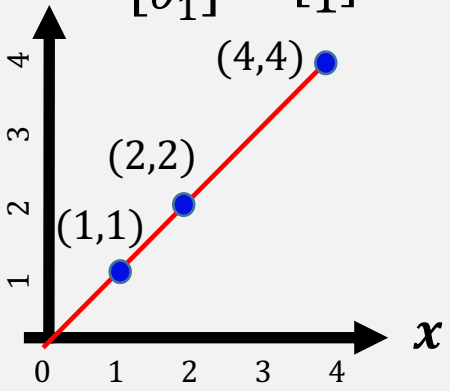
$$J\left(\theta = \begin{bmatrix}0\\ 0.5\end{bmatrix}\right) = \frac{1}{2m}\left((0.5-1)^2 + (1-2)^2 + (2-4)^2\right)$$

$$= \frac{1}{6}\left((-0.5)^2 + (-1)^2 + (2)^2\right)$$

$$= \frac{1}{6}(0.25 + 1 + 4)$$

$$= 0.875$$

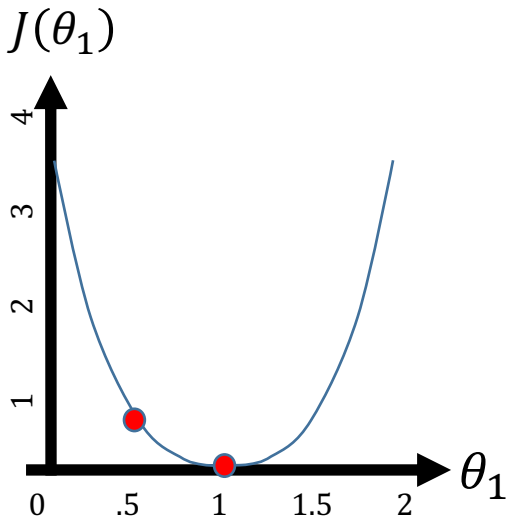
$$\begin{bmatrix}\theta_0\\ \theta_1\end{bmatrix} = \begin{bmatrix}0\\ 1\end{bmatrix}$$



$$J\left(\theta = \begin{bmatrix}0\\ 1\end{bmatrix}\right) = \frac{1}{6}\left((1-1)^2 + (2-2)^2 + (4-4)^2\right)$$

$$= 0$$

The loss function, $J(\theta_1)$, is quadratic w.r.t. θ_1



⇒Convex

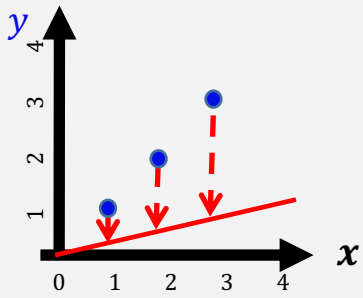
⇒Global minimum

Another example

L2 loss, or “Mean Squared Error”

$$J(\theta) = \frac{1}{2m}\sum_{i=1}^{m}\left(h_\theta\left(x^{(i)}\right) - y^{(i)}\right)^2$$

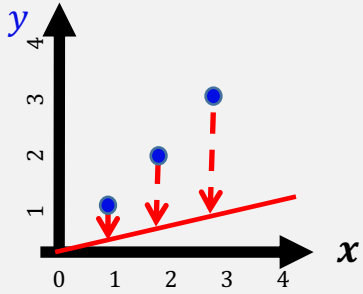
$J(\theta_1 = 0.25) = 7.4$

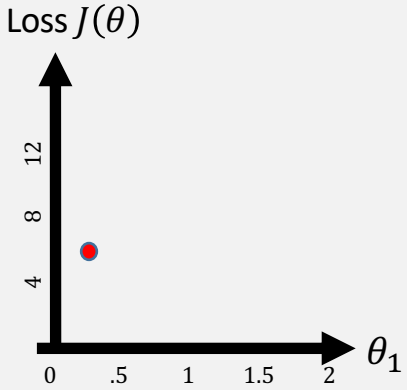


L2 loss, or “Mean Squared Error”

$$J(\theta) = \frac{1}{2m}\sum_{i=1}^{m}\left(h_\theta\left(x^{(i)}\right) - y^{(i)}\right)^2$$

$J(\theta_1 = 0.25) = 7.4$

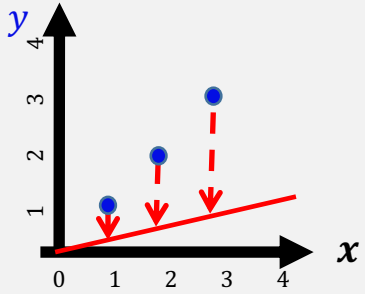




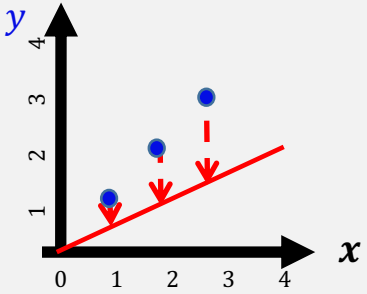
L2 loss, or “Mean Squared Error”

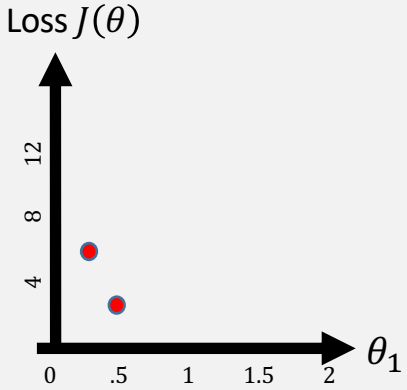
$$J(\theta) = \frac{1}{2m}\sum_{i=1}^{m}\left(h_\theta\left(x^{(i)}\right) - y^{(i)}\right)^2$$

$J(\theta_1 = 0.25) = 7.4$



$J(\theta_1 = 0.5) = 3.3$

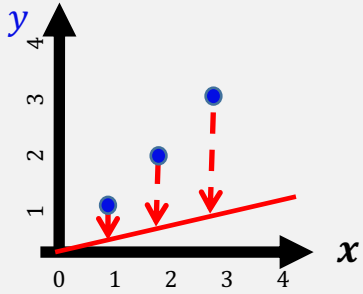




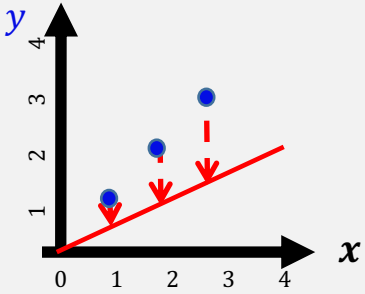
L2 loss, or “Mean Squared Error”

$$J(\theta) = \frac{1}{2m}\sum_{i=1}^{m}\left(h_\theta\left(x^{(i)}\right) - y^{(i)}\right)^2$$

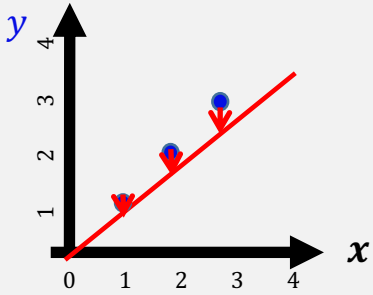
$J(\theta_1 = 0.25) = 7.4$



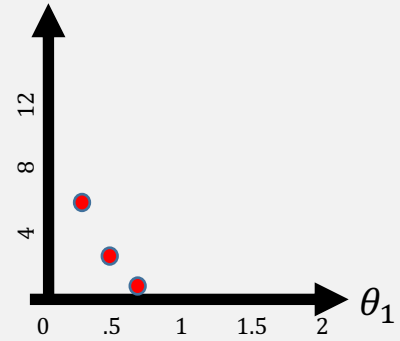
$J(\theta_1 = 0.5) = 3.3$



$J(\theta_1 = 0.75) = 0.82$



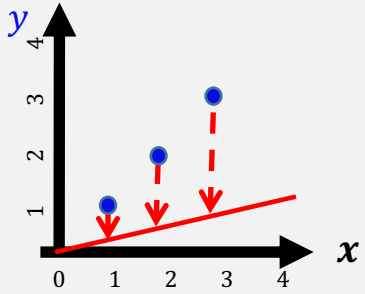
Loss $J(\theta)$



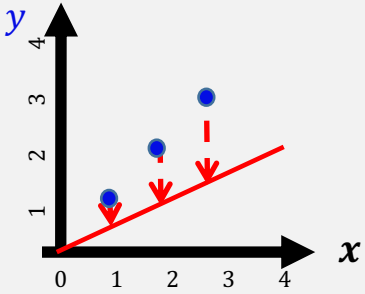
L2 loss, or “Mean Squared Error”

$$J(\theta) = \frac{1}{2m}\sum_{i=1}^{m}\left(h_\theta\left(x^{(i)}\right) - y^{(i)}\right)^2$$

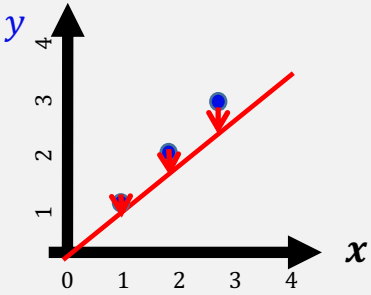
$J(\theta_1 = 0.25) = 7.4$



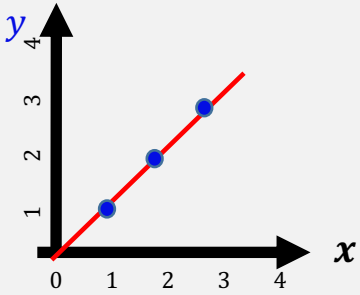
$J(\theta_1 = 0.5) = 3.3$



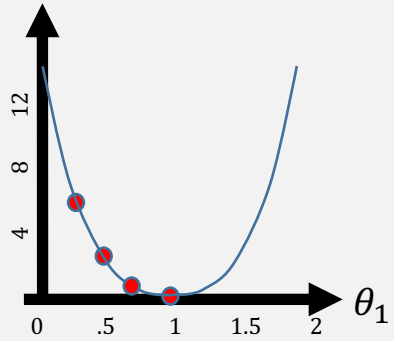
$J(\theta_1 = 0.75) = 0.82$



$J(\theta_1 = 1) = 0$



Loss $J(\theta)$



$\Rightarrow$Convex

$\Rightarrow$Global minimum

Lecture 4

Univariate Linear Regression

- The loss function

Introduction to Optimisation

- Derivatives and critical points
- Gradient descent with one variable

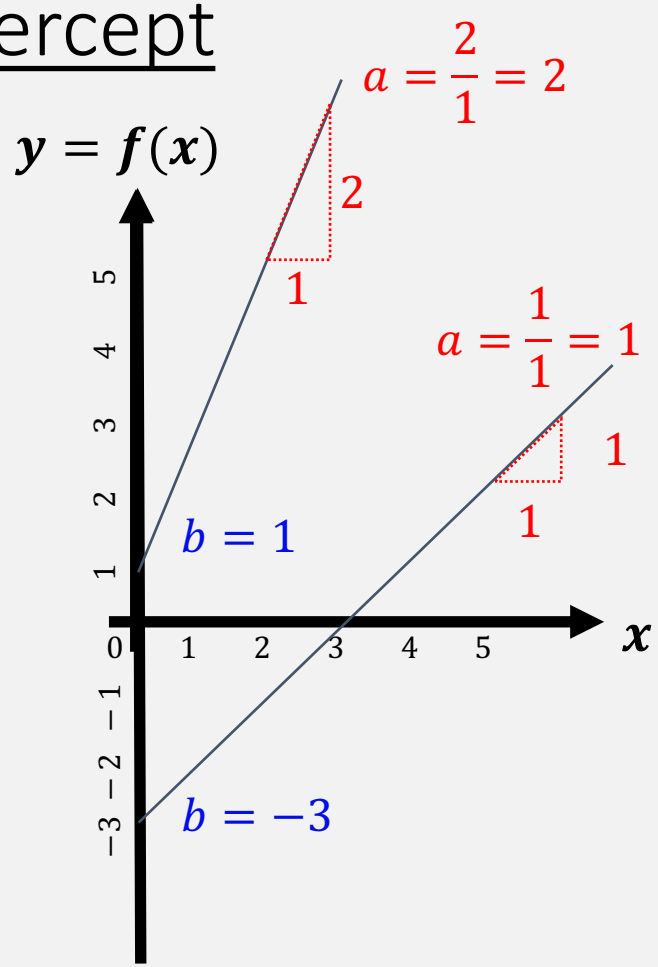
Multidimensional Gradient Descent

- Gradient descent with N variables
- Gradient descent and Linear Regression

Describing a line: slope and intercept

- a (the slope, or gradient)
- b (the intercept)

$$y = f(x) = b + ax$$

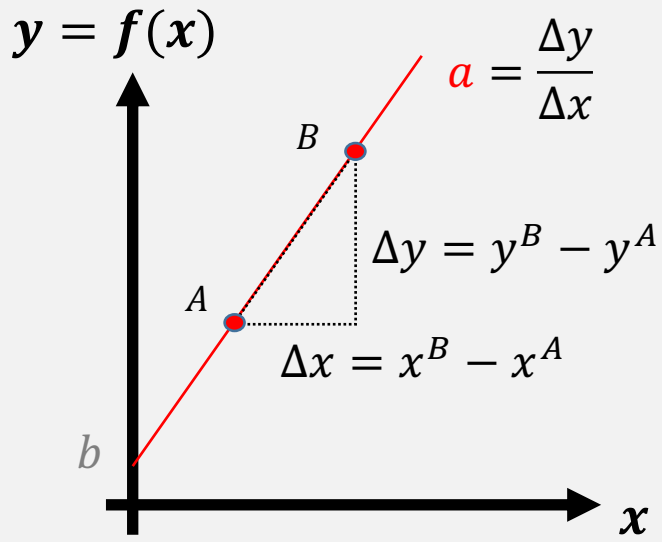


Describing a line: slope and intercept

- a (the slope, or gradient)
- b (the intercept)

$$y = f(x) = b + ax$$

$$f'(x) = a = \frac{\Delta y}{\Delta x}$$

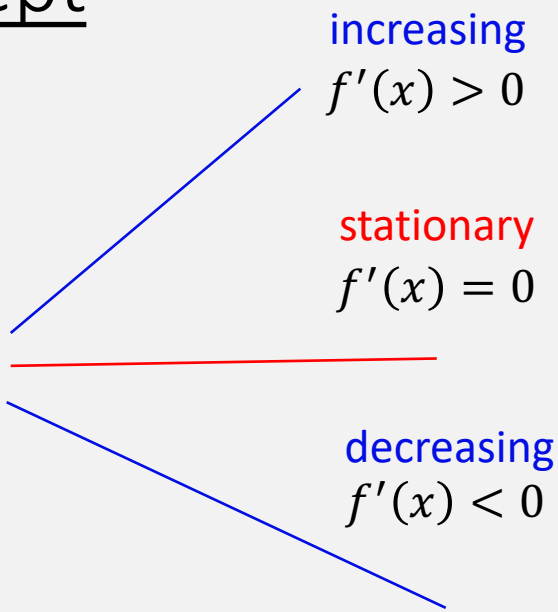


Describing a line: slope and intercept

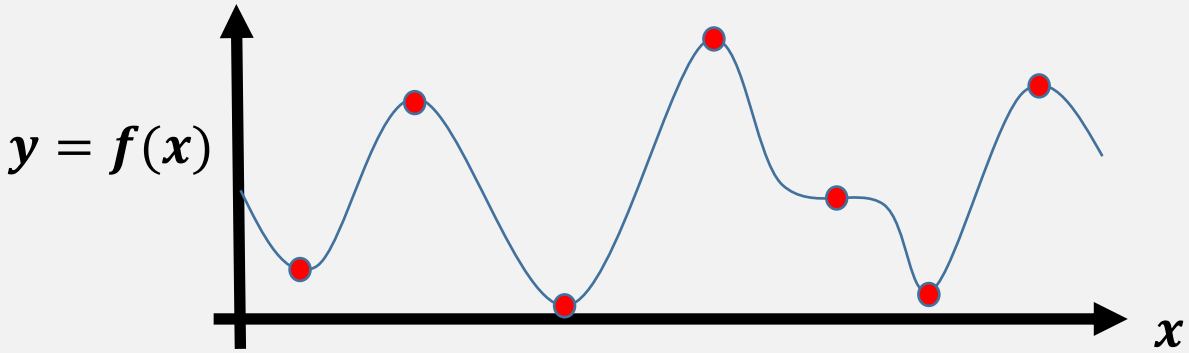
- a (the slope, or gradient)
- b (the intercept)

$$y = f(x) = b + ax$$

$$f'(x) = a = \frac{\Delta y}{\Delta x}$$



Critical or stationary points



Math primer: derivatives of a function

A derivative of a function f is the slope or gradient of that function with respect to some variable

e.g. the derivative of f w.r.t. x: $f'(x) = \frac{df(x)}{dx} = \frac{\Delta f(x)}{\Delta x}$

General rule: $\frac{d(x^n)}{dx} = n * x^{n-1}$

e.g. $f(x) = x^3 \Rightarrow f'(x) = 3x^2$

Constants disappear, e.g.

$$f(x) = x^2 + 10 \Rightarrow f'(x) = 2x$$

As do other variables, e.g.

Partial differential equation

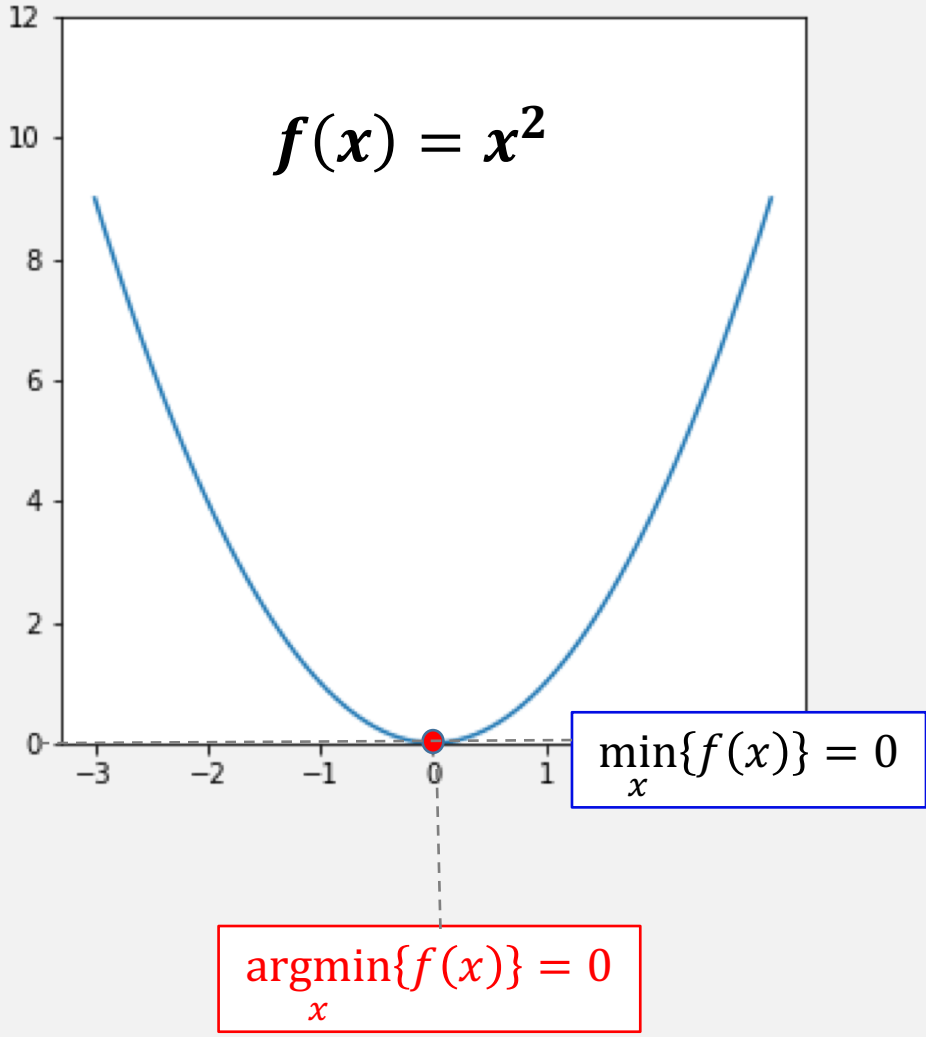
$$f(x,y) = x^2 + y^2 \Rightarrow \frac{\partial f(x,y)}{\partial x} = 2x$$

Optimisation

$f(x) = x^2$

What is the minimum value of $f(x)$?

What value of x minimises $f(x)$?

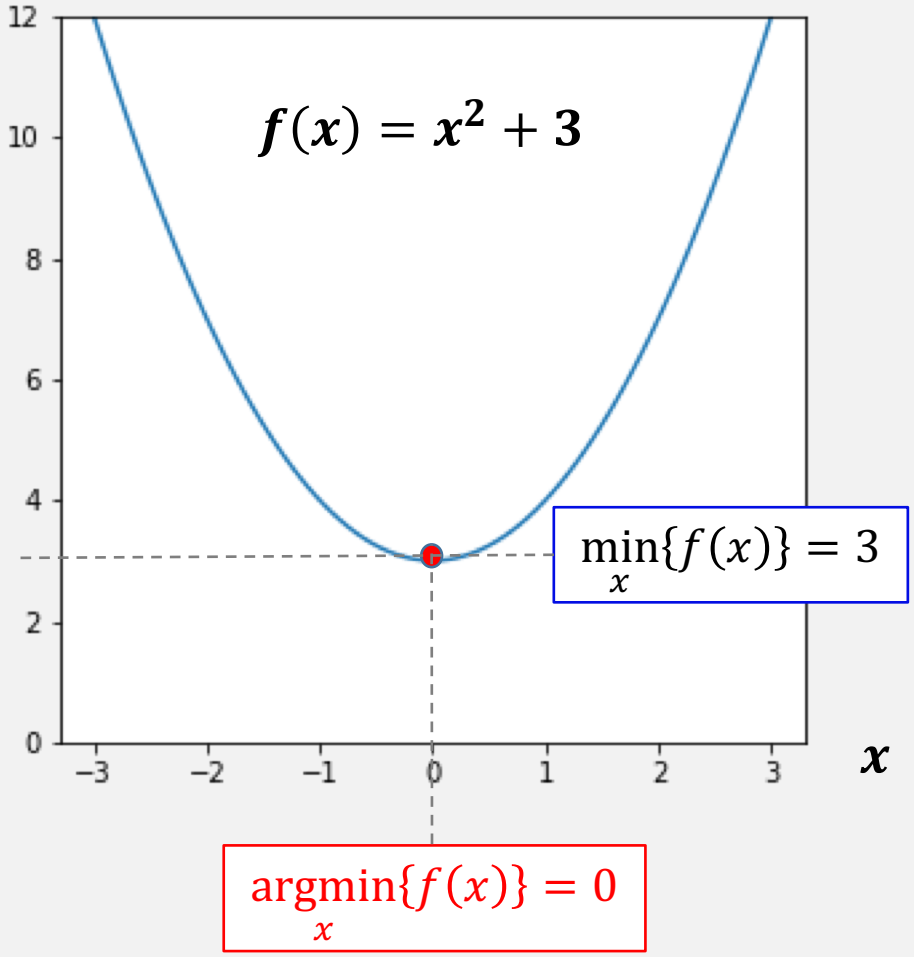


Optimisation

$f(x) = x^2 + 3$

What is the minimum value of $f(x)$?

What value of x minimises $f(x)$?



Optimisation

$f(x) = x^2 + 3$

What is the minimum value of $f(x)$?

What value of x minimises $f(x)$?

How can we calculate the optimal value of x (e.g. $\underset{x}{\text{argmin}}\, f(x)$)?

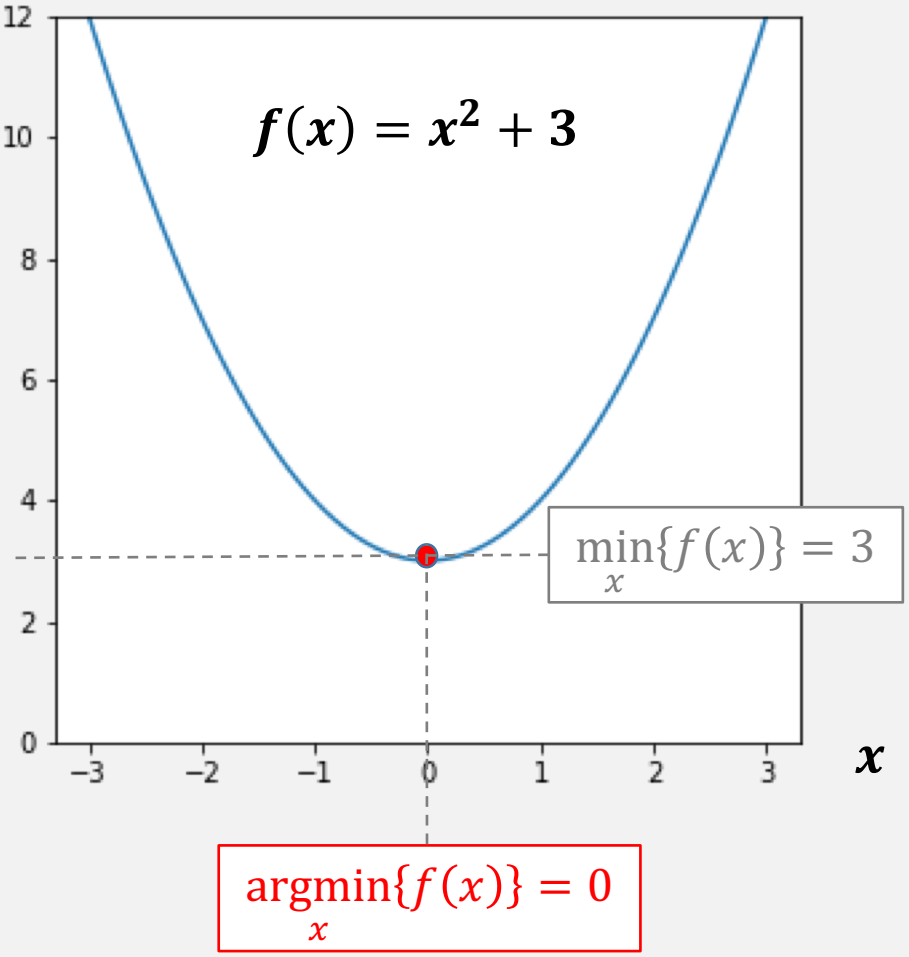
➔Use the derivative of f

$$(f'(x) = \text{slope} = \frac{\Delta f(x)}{\Delta x})$$

$$f'(x) = 2x,$$

Critical point: $f'(x) = 2x = 0$

$$\Rightarrow x = 0$$



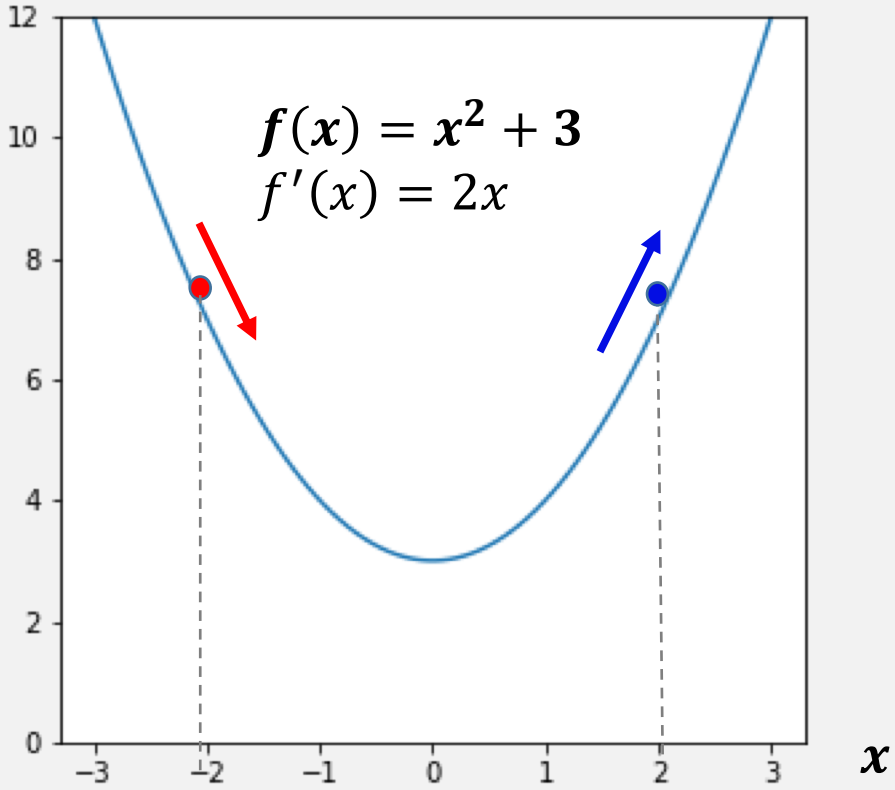
Optimisation

What is $f'(x)$ at $x = 2$?

$$f'(2) = 4$$
$$> 0$$
$$\Rightarrow f(x) \text{ increasing}$$

What is $f'(x)$ at $x = -2$?

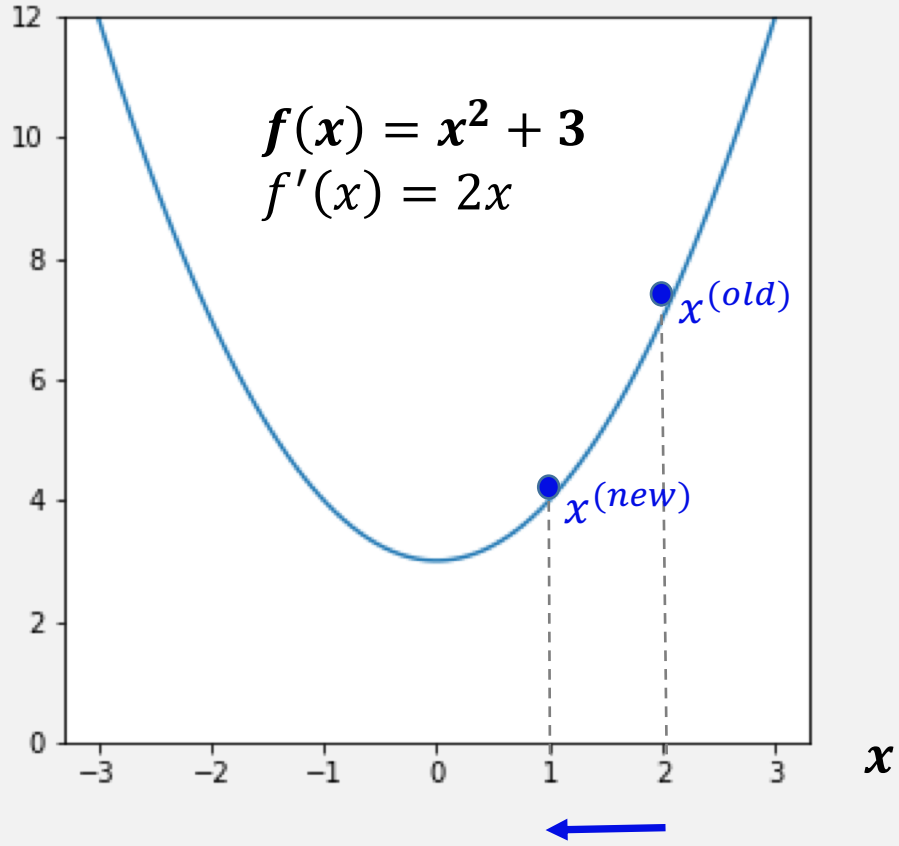
$$f'(-2) = -4$$
$$< 0$$
$$\Rightarrow f(x) \text{ decreasing}$$



Optimisation

Goal: find x that minimises $f(x)$

(from randomly chosen start $x^{(old)}$)



$f'(2) > 0 \Rightarrow f(x)$ increasing

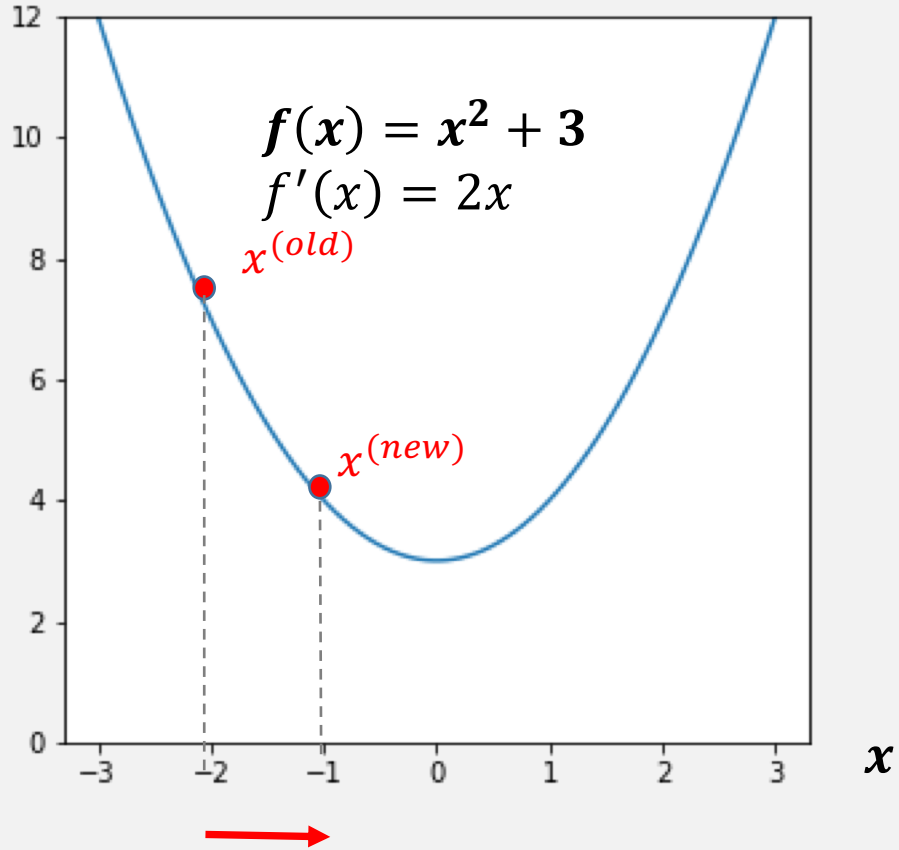
$\Rightarrow$ decrease $f(x)$ by **decreasing** x:

$$x^{(new)} = x^{(old)} - \alpha f'\left(x^{(old)}\right), \quad \alpha > 0$$

Optimisation

Goal: find x that minimises $f(x)$

(from randomly chosen start $x^{(old)}$)



$f'(-2) < 0 \Rightarrow f(x)$ decreasing

$\Rightarrow$ decrease $f(x)$ by **increasing** x:

$$x^{(new)} = x^{(old)} - \alpha f'\left(x^{(old)}\right), \quad \alpha > 0$$

$f'(2) > 0 \Rightarrow f(x)$ increasing

$\Rightarrow$ decrease $f(x)$ by **decreasing** x:

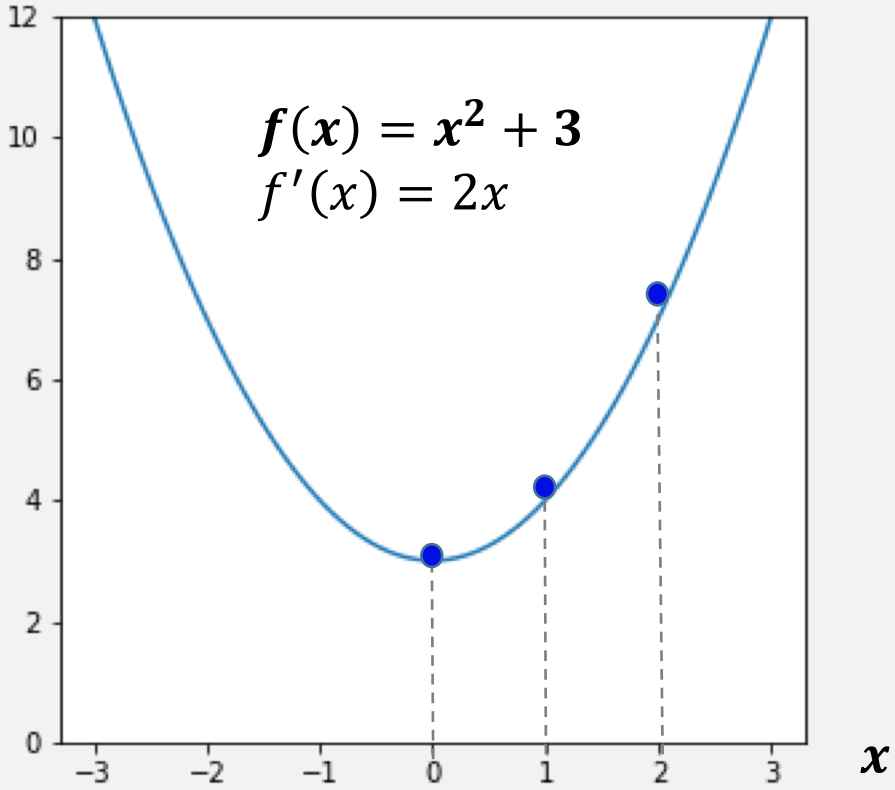
$$x^{(new)} = x^{(old)} - \alpha f'\left(x^{(old)}\right), \quad \alpha > 0$$

Gradient Descent

Goal: find x that minimises $f(x)$

1. Start at random x, say $x = 2$
2. Iterate until minimum $f(x)$ found:

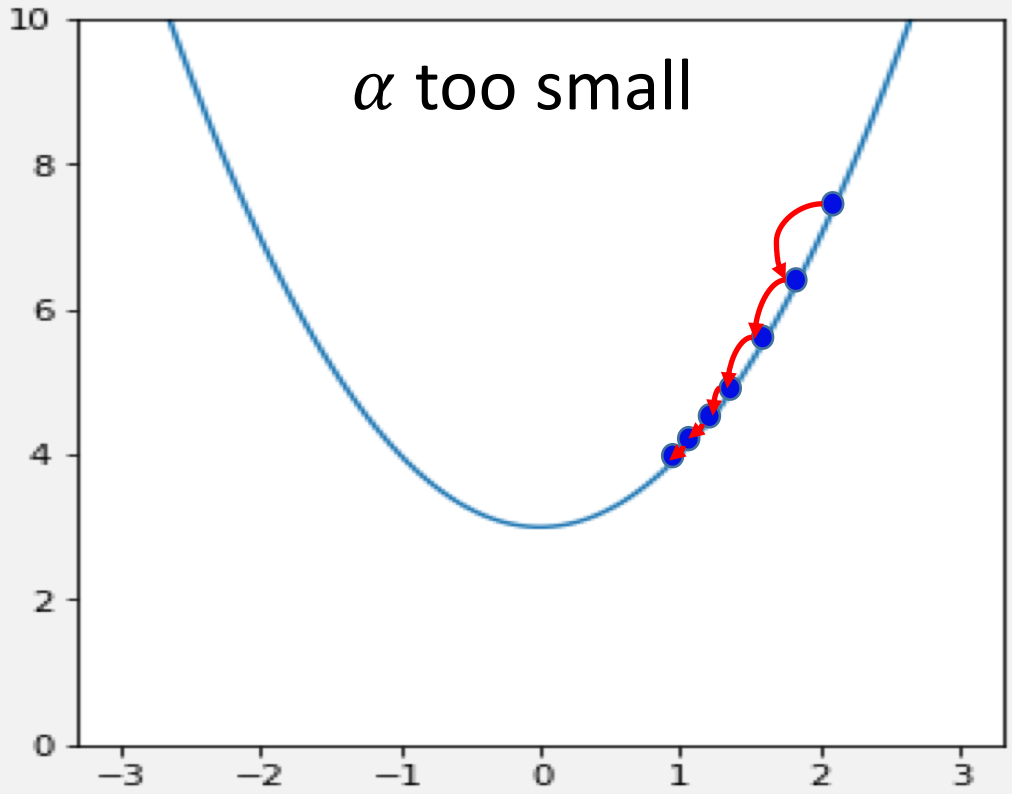
$$x^{(new)} = x^{(old)} - \alpha f'\left(x^{(old)}\right), \quad \alpha > 0$$



What does α do?

Choosing your learning rate (α) wisely

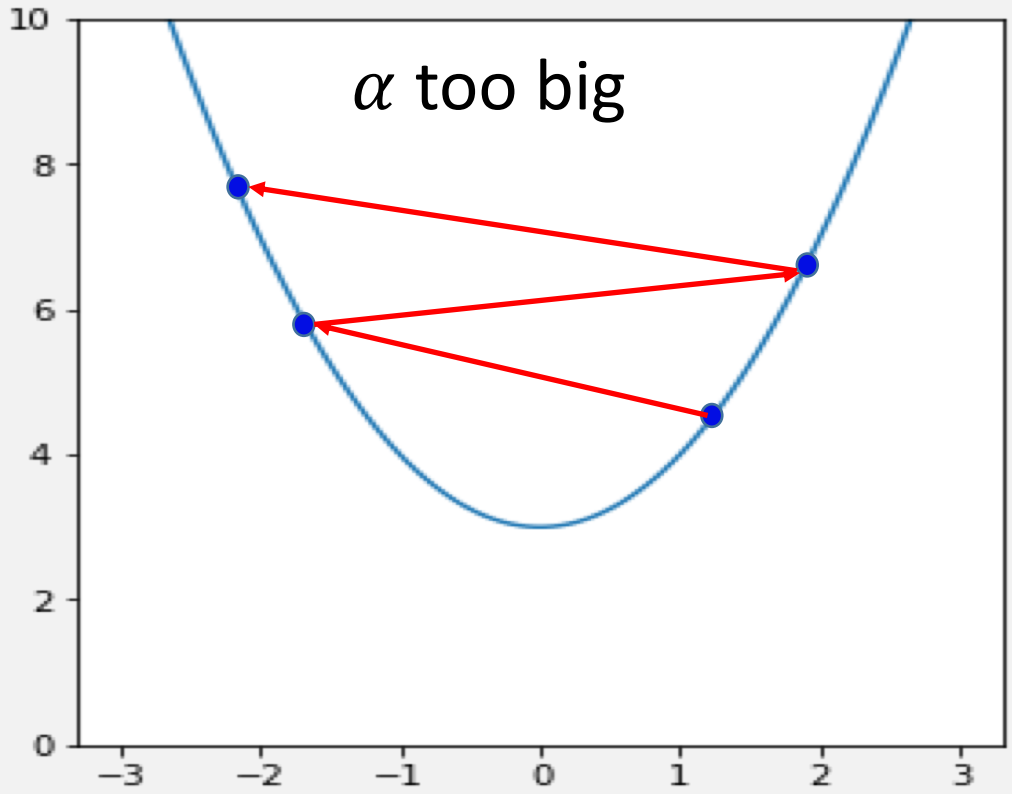
$$x^{(new)} = x^{(old)} - \alpha f'\left(x^{(old)}\right)$$



Takes a long time to converge (reach the minimum)

Choosing your learning rate (α) wisely

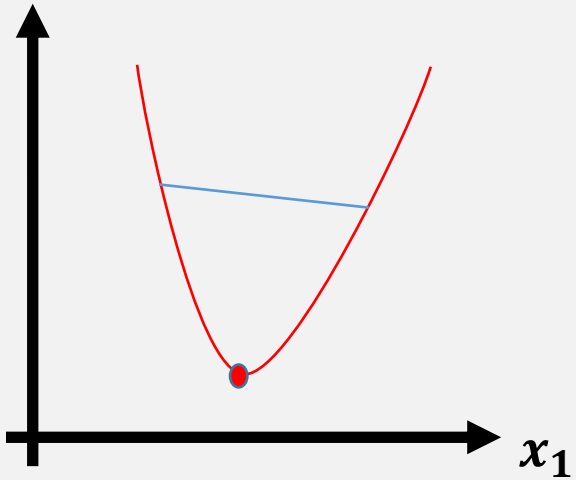
$$x^{(new)} = x^{(old)} - \alpha f'\left(x^{(old)}\right)$$



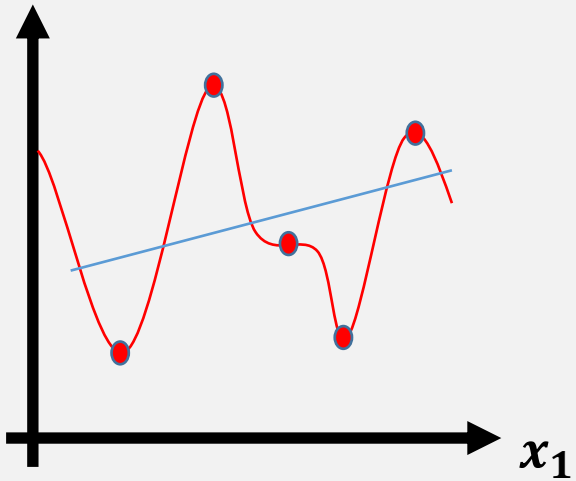
Overshoots – and may never converge

Convexity

Critical point, i.e. $f'(x_1) = 0$



(a)



(c)

Convex function: (a)

- ▸Single local optimum
- ▸Also global optimum

Non-convex function: (c)

- ▸Multiple local optimum

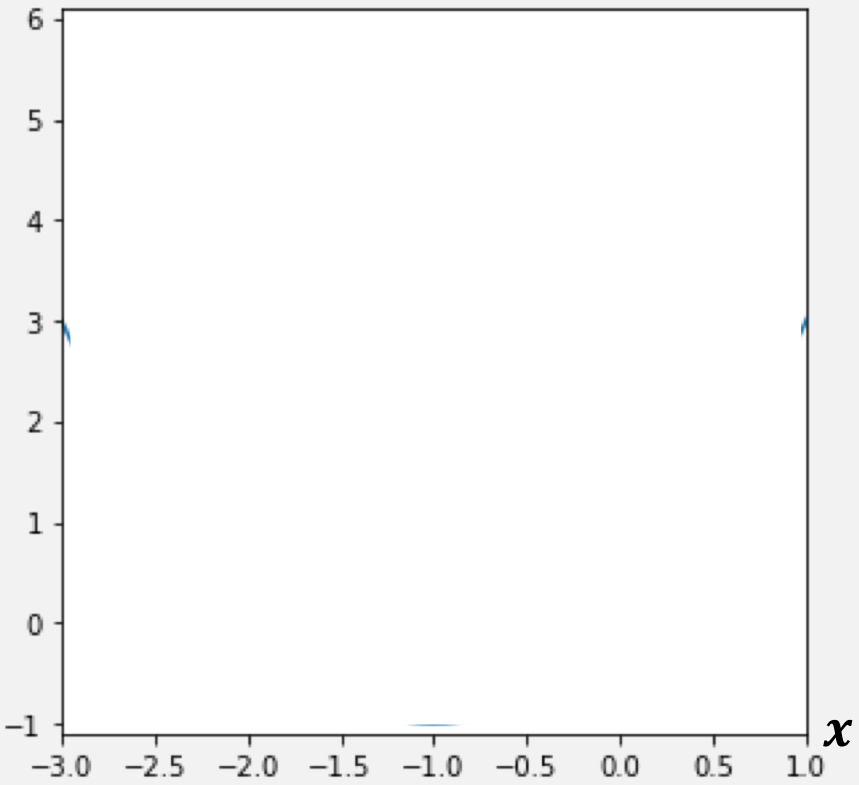
A function is convex if every line that connects any 2 points on the function graph lies on or above the graph.

Optimisation: example

Use the derivative to calculate the optimal value of x for

$$f(x) = 2x + x^2$$

(i.e. the x that minimises $f(x)$)



Optimisation: example

$$\frac{d(x^n)}{\mathrm{dx}} = n * x^{n-1}$$

Use the derivative to calculate the optimal value of x for

$$f(x) = 2x + x^2$$

(i.e. the x that minimises $f(x)$)

$$f(x) = 2x^1 + x^2$$

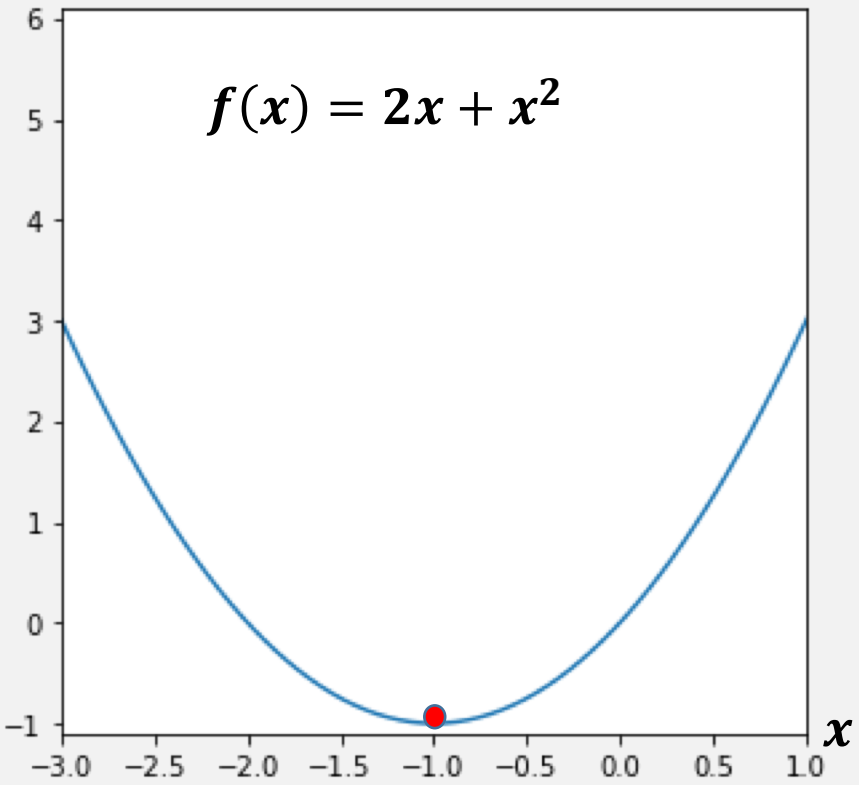
Derivative: $f'(x) = 1 * 2x^0 + 2 * x^1$

$$= 2 + 2x$$

Minimum at zero: $f'(x) = 2 + 2x = 0$

$$2x = -2$$

$$\Rightarrow x = -1$$



Lecture 4

Univariate Linear Regression

- The loss function

Introduction to Optimisation

- Derivatives and critical points
- Gradient descent with one variable

Multidimensional Gradient Descent

- Gradient descent with N variables
- Gradient descent and Linear Regression

Gradient Descent: N-Dimensional

In 2D we use the derivative to guide Gradient Descent

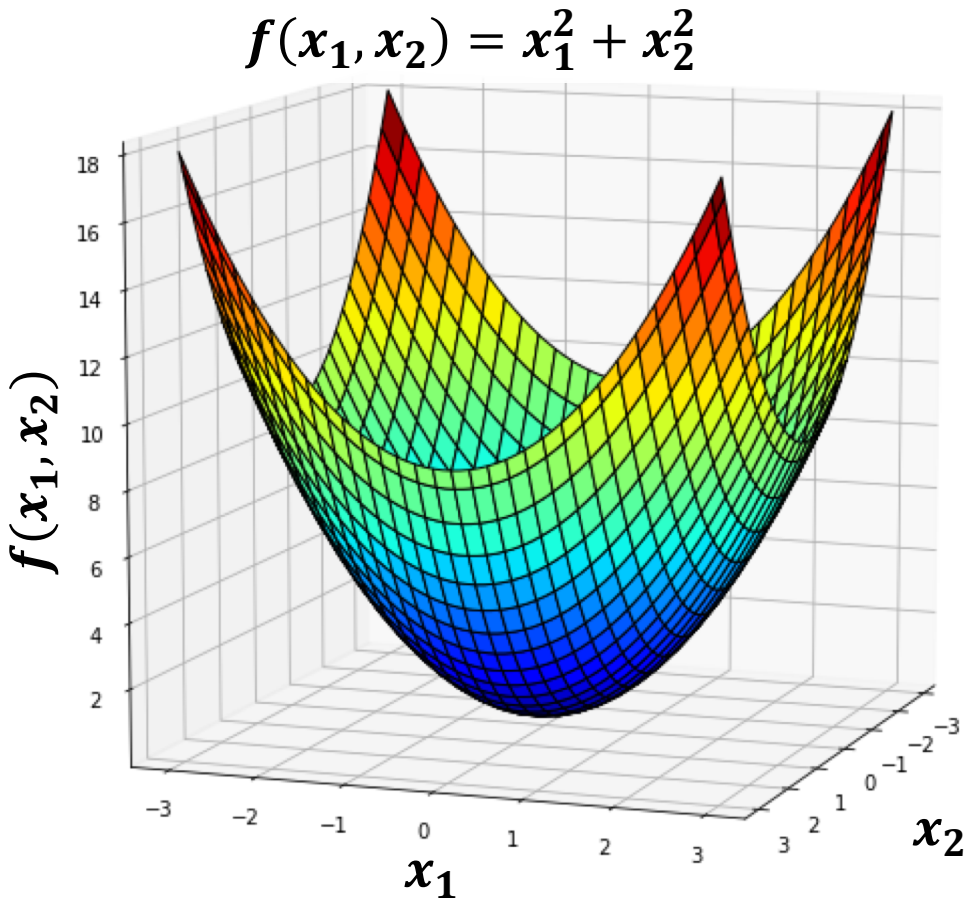
➔Update x towards steepest descent to minimise $f(x)$

➔(1D search)

What should we do in 3D?

➔Update x_1, x_2 towards steepest descent to minimise $f(x_1, x_2)$

➔(2D search)



Gradient Descent: N-Dimensional

Idea: use partial derivatives of f with respect to each of x_1, x_2:

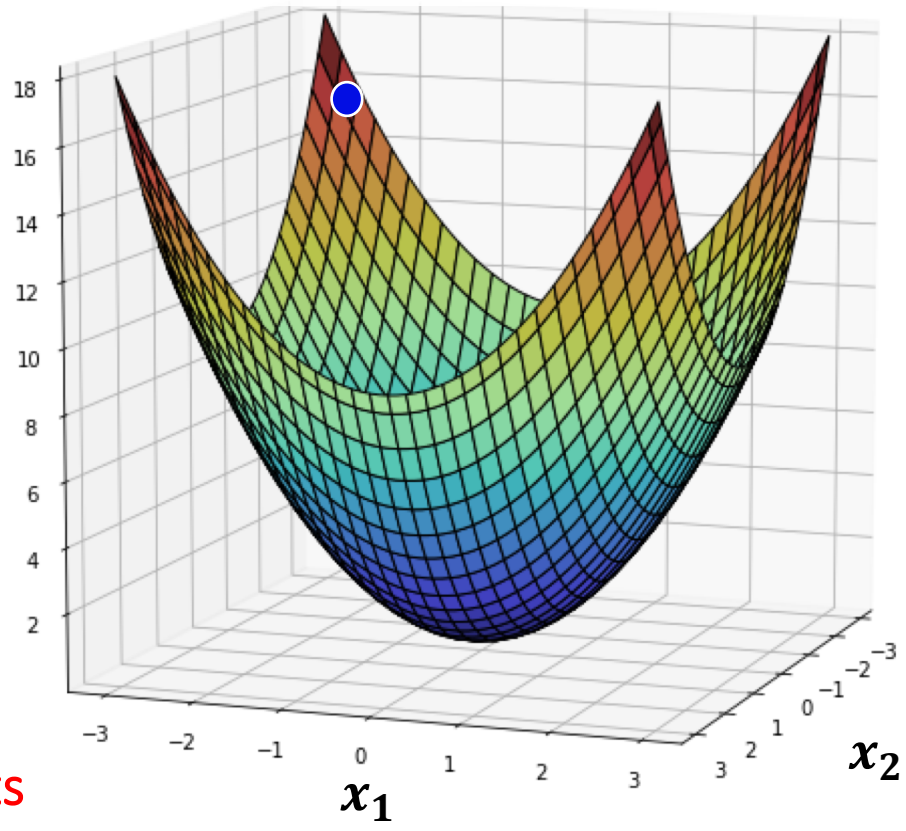
$$\frac{\partial f(x_1,x_2)}{\partial x_1} \text{ and } \frac{\partial f(x_1,x_2)}{\partial x_2}$$

Treat partial derivatives as vectors:

$$\nabla f(x_1, x_2)^T = \begin{bmatrix} \frac{\partial f(x_1,x_2)}{\partial x_1} \\ \frac{\partial f(x_1,x_2)}{\partial x_2} \end{bmatrix}$$

These vectors are known as gradients

$$f(x_1, x_2) = x_1^2 + x_2^2$$



e.g. for $f(x_1, x_2) = x_1^2 + x_2^2$:

$$\nabla f(x_1, x_2)^T = \begin{bmatrix} 2x_1 \\ 2x_2 \end{bmatrix}$$

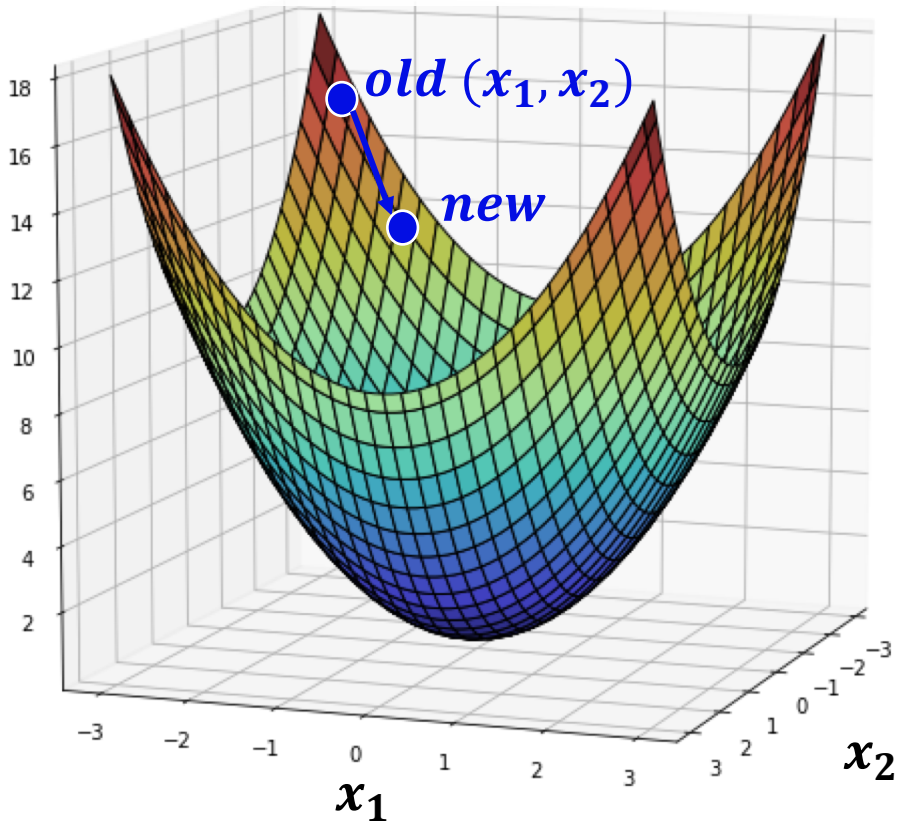
Gradient Descent: N-Dimensional

Gradient descent update rule:

$$\begin{bmatrix} x_1^{new} \\ x_2^{new} \end{bmatrix} = \begin{bmatrix} x_1^{old} \\ x_2^{old} \end{bmatrix} - \alpha \nabla f\left(x_1^{old}, x_2^{old}\right)^T$$

$$= \begin{bmatrix} x_1^{old} \\ x_2^{old} \end{bmatrix} - \alpha \begin{bmatrix} \frac{\partial f\left(x_1^{old}, x_2^{old}\right)}{\partial x_1^{old}} \\ \frac{\partial f\left(x_1^{old}, x_2^{old}\right)}{\partial x_2^{old}} \end{bmatrix}$$

$$f(x_1, x_2) = x_1^2 + x_2^2$$



Separately:

$$x_1^{new} = x_1^{old} - \alpha \frac{\partial f\left(x_1^{old}, x_2^{old}\right)}{\partial x_1^{old}}$$

$$x_2^{new} = x_2^{old} - \alpha \frac{\partial f\left(x_1^{old}, x_2^{old}\right)}{\partial x_2^{old}}$$

Gradient Descent: N-Dimensional

Gradient descent update rule:

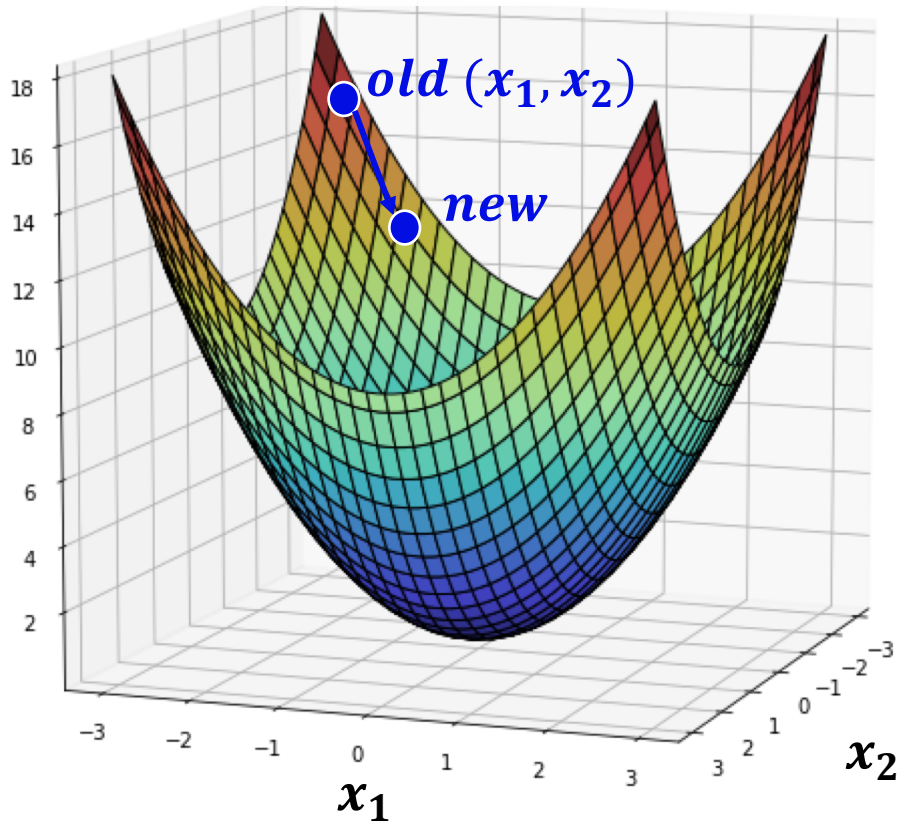
$$\begin{bmatrix} x_1^{new} \\ x_2^{new} \end{bmatrix} = \begin{bmatrix} x_1^{old} \\ x_2^{old} \end{bmatrix} - \alpha \nabla f\left(x_1^{old}, x_2^{old}\right)^T$$

So for $f(x_1, x_2) = x_1^2 + x_2^2$:

$$\begin{bmatrix} x_1^{new} \\ x_2^{new} \end{bmatrix} = \begin{bmatrix} x_1^{new} \\ x_2^{new} \end{bmatrix} - 2\alpha \begin{bmatrix} x_1^{old} \\ x_2^{old} \end{bmatrix}$$

Or:
$$x_1^{new} = x_1^{old} - \alpha(2x_1^{old})$$
$$x_2^{new} = x_2^{old} - \alpha(2x_2^{old})$$

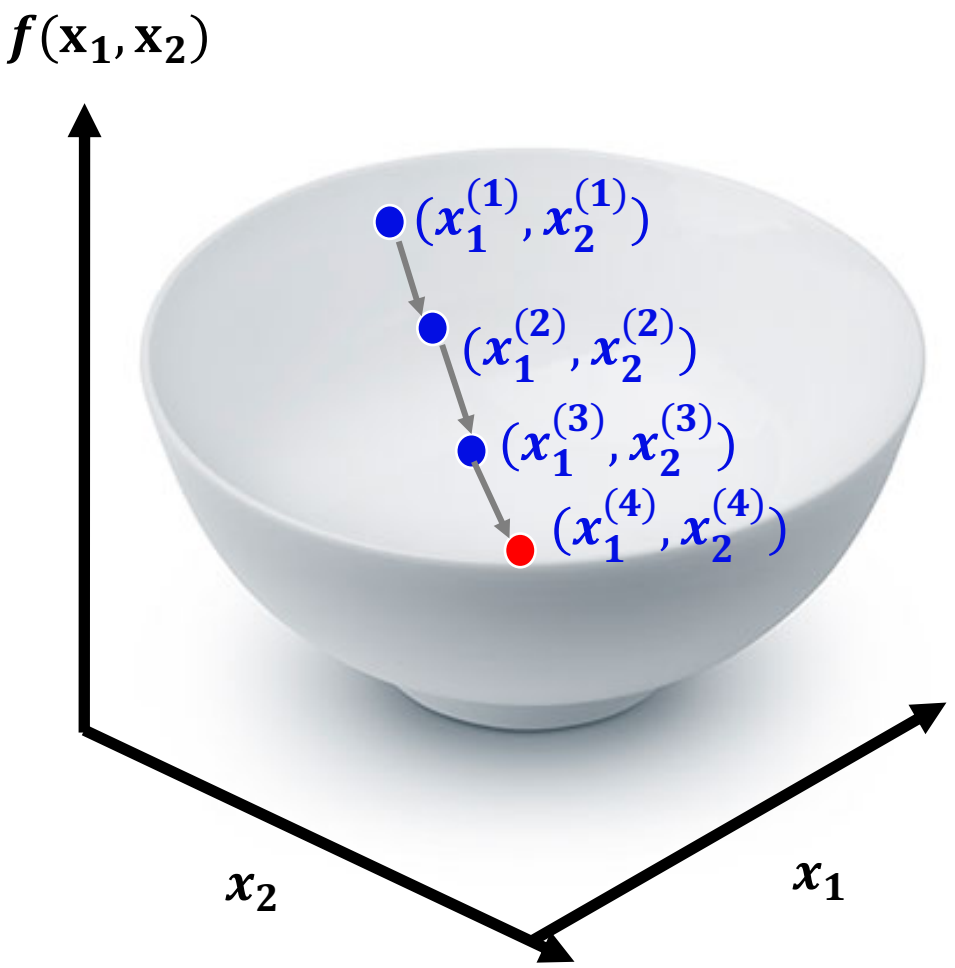
$f(x_1, x_2) = x_1^2 + x_2^2$



Separately:

$$x_1^{new} = x_1^{old} - \alpha \frac{\partial f\left(x_1^{old}, x_2^{old}\right)}{\partial x_1^{old}} \qquad x_2^{new} = x_2^{old} - \alpha \frac{\partial f\left(x_1^{old}, x_2^{old}\right)}{\partial x_2^{old}}$$

Optimising a convex function



Gradient Descent

Initialise at random point

Repeat:

$$\begin{bmatrix} x_1^{(i+1)} \\ x_2^{(i+1)} \end{bmatrix} = \begin{bmatrix} x_1^{(i)} \\ x_2^{(i)} \end{bmatrix} - \alpha \nabla f\left(\begin{bmatrix} x_1^{(i)} \\ x_2^{(i)} \end{bmatrix}\right)$$

Until convergence:

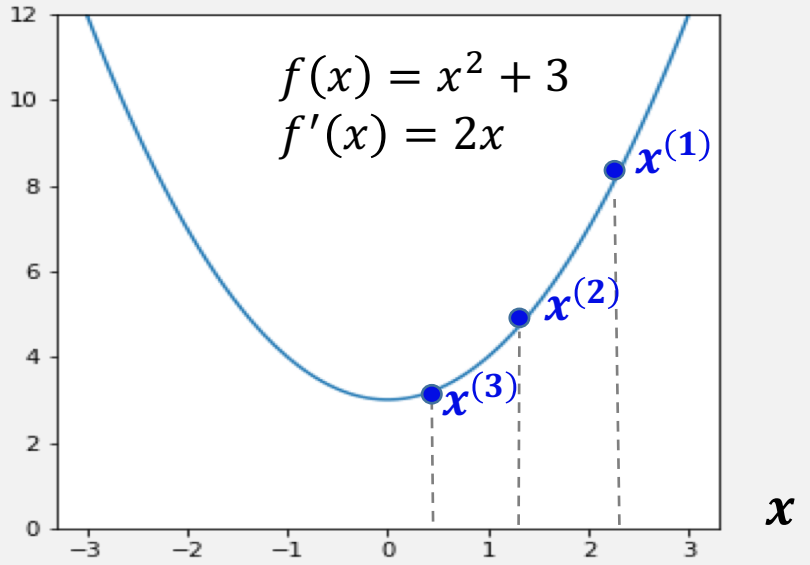
$$x^{(i+1)} \approx x^{(i)}$$

1D Gradient Descent

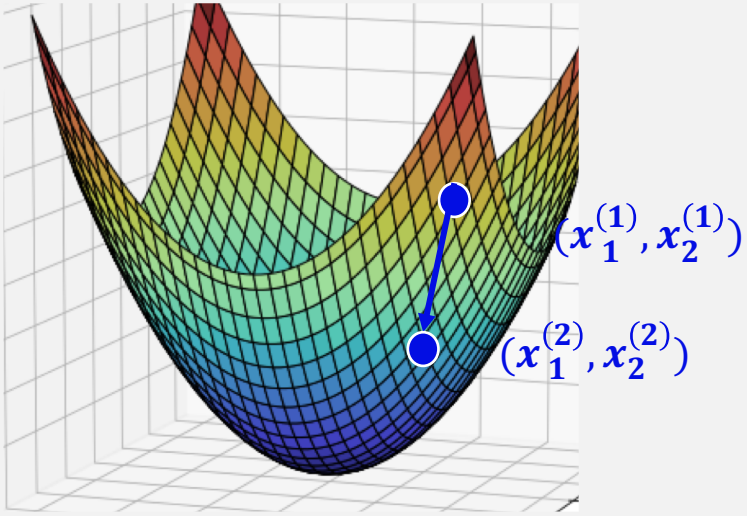
Goal: find x that minimises f

1. Start at random x
2. Iterate until minimum $f(x)$ found:

$$x^{(i+1)} = x^{(i)} - \alpha f'\left(x^{(i)}\right)$$



$$f(x_1, x_2) = x_1^2 + x_2^2$$
$$\nabla f = [2x_1, 2x_2]^T$$



2D Gradient Descent

Goal: find (x_1, x_2) that minimises f

1. Start at random (x_1, x_2)
2. Iterate until minimum $f(x)$ found:

$$\begin{bmatrix} x_1^{(i+1)} \\ x_2^{(i+1)} \end{bmatrix} = \begin{bmatrix} x_1^{(i)} \\ x_2^{(i)} \end{bmatrix} - \alpha \nabla f\left(\begin{bmatrix} x_1^{(i)} \\ x_2^{(i)} \end{bmatrix}\right)$$

Aside: Derivatives of a function

A derivative of a function f is the slope or gradient of that function with respect to some variable

e.g. the derivative of f w.r.t. x: $f'(x) = \frac{df(x)}{dx} = \frac{\Delta f(x)}{\Delta x}$

General rule: $\frac{d(x^n)}{dx} = n * x^{n-1}$

The chain rule: if $z = f(y)$ and $y = g(x)$

$$\frac{dz}{dx} = \frac{dz}{dy} \cdot \frac{dy}{dx} = f'(y)g'(x)$$

$$= f'\big(g(x)\big)g'(x)$$

e.g. if $f(x) = (3x-1)^2$, then $f'(x) = \frac{d(3x-1)^2}{d(3x-1)} \cdot \frac{d(3x-1)}{dx} = 2 * (3x-1) * 3$

General Gradient Descent for N-Dimensions

Update all parameters x in the opposite direction to steepest ascent to minimise a function, f:

$$x_j^{new} = x_j^{old} - \alpha \frac{\partial f\left(x_j^{old}\right)}{\partial x_j^{old}}$$

$\alpha > 0$, the "learning rate" is a scalar value

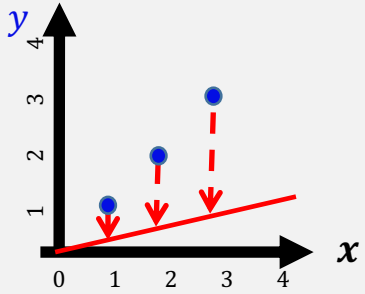
Partial derivative

Just do usual derivative in terms of a single parameter of interest (in this case x_j) – all other terms in f can be treated like a constant.

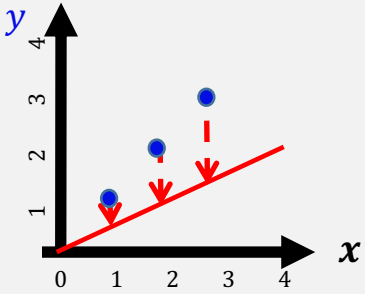
Gradient Descent (1D) on Linear Regression

$$J(\theta) = \frac{1}{2m}\sum_{i=1}^{m}\left(\theta_0 + \theta_1 x^{(i)} - y^{(i)}\right)^2$$

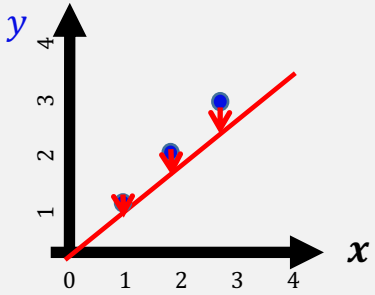
$J(\theta_1 = 0.25) = 7.4$



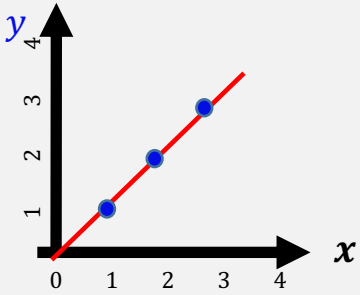
$J(\theta_1 = 0.5) = 3.3$



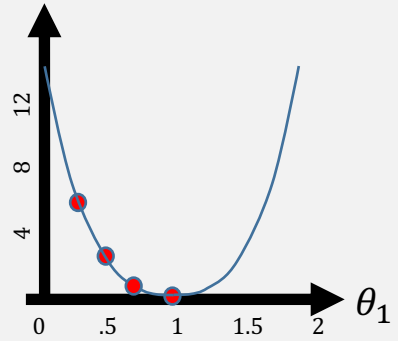
$J(\theta_1 = 0.75) = 0.82$



$J(\theta_1 = 1) = 0$



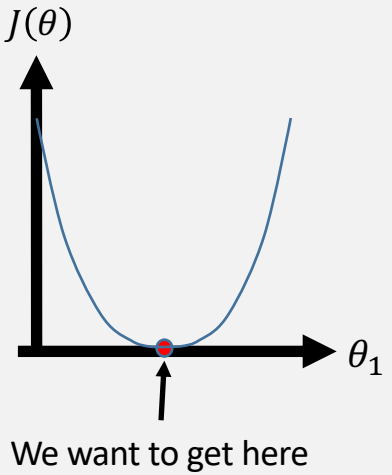
Loss $J(\theta)$



Gradient Descent (1D) on Linear Regression

$$J(\theta) = \frac{1}{2m}\sum_{i=1}^{m}\left(\theta_0 + \theta_1 x^{(i)} - y^{(i)}\right)^2$$

Goal: Find the parameter, θ_1, that minimises loss, $J(\theta)$



Gradient Descent (1D)

Goal: Find

Differentiating the Loss

L2 loss

$$J(\theta) = \frac{1}{2m}\sum_{i=1}^{m}\left(\theta_0 + \theta_1 x^{(i)} - y^{(i)}\right)^2$$

o get here

Gradient Descent (1D)

Differentiating the Loss

L2 loss

$$J(\theta) = \frac{1}{2m}\sum_{i=1}^{m}\left(\theta_0 + \theta_1 x^{(i)} - y^{(i)}\right)^2$$

Gradient of the loss

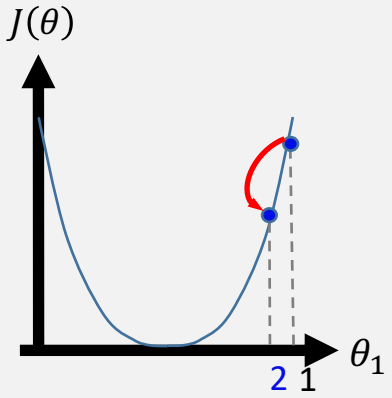
$$J_1'(\theta) = \frac{\partial J(\theta)}{\partial \theta_1} = \frac{1}{m}\sum_{i=1}^{m}\left(\theta_0 + \theta_1 x_1^{(i)} - y^{(i)}\right) * x_1^{(i)}$$

Gradient Descent (1D)

Goal: Find the parameter, θ_1, that minimises loss, $J(\theta)$

Gradient descent update rule

$$\theta_1^{(2)} = \theta_1^{(1)} - \alpha J_1'\left(\theta^{(1)}\right)$$

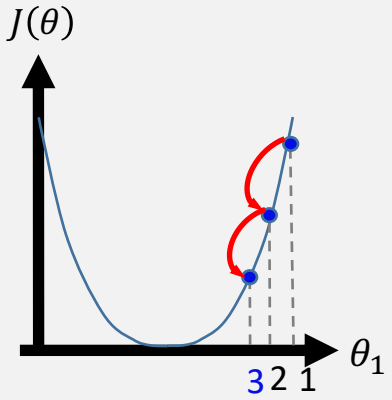


Gradient Descent (1D)

Goal: Find the parameter, θ_1, that minimises loss, $J(\theta)$

Gradient descent update rule

$$\theta_1^{(3)} = \theta_1^{(2)} - \alpha J_1'\left(\theta^{(2)}\right)$$

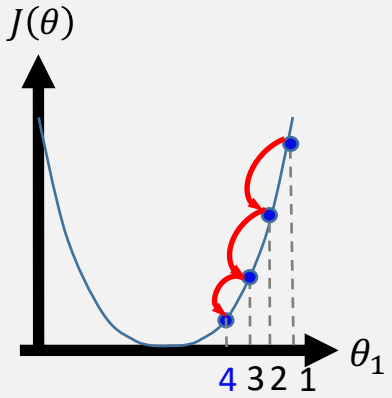


Gradient Descent (1D)

Goal: Find the parameter, θ_1, that minimises loss, $J(\theta)$

Gradient descent update rule

$$\theta_1^{(4)} = \theta_1^{(3)} - \alpha J_1'\left(\theta^{(3)}\right)$$

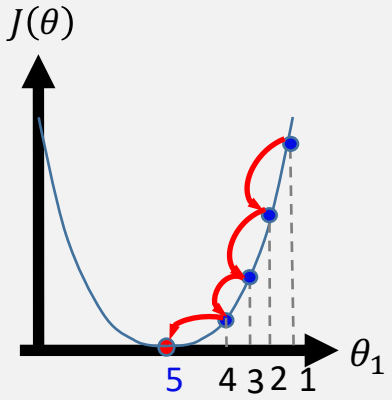


Gradient Descent (1D)

Goal: Find the parameter, θ_1, that minimises loss, $J(\theta)$

Gradient descent update rule

$$\theta_1^{(5)} = \theta_1^{(4)} - \alpha J_1'\left(\theta^{(4)}\right)$$

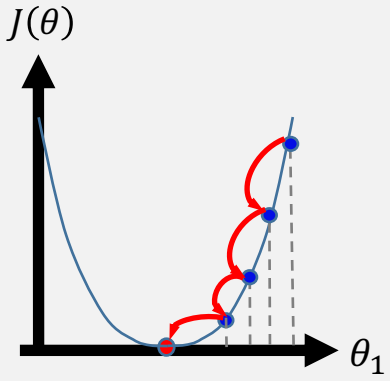


Gradient Descent (1D)

Goal: Find the parameter, θ_1, that minimises loss, $J(\theta)$

Gradient descent update rule

$$\theta_1^{(new)} = \theta_1^{(old)} - \alpha J_1'\left(\theta^{(old)}\right)$$



REMINDER

General Gradient Descent for N-Dimensions

Update all parameters x in the opposite direction to steepest ascent to minimise a function, f:

$$x_j^{new} = x_j^{old} - \alpha \frac{\partial f\left(x_j^{old}\right)}{\partial x_j^{old}}$$

$\alpha > 0$, the "learning rate" is a scalar value

Partial derivative

Just do usual derivative in terms of a single parameter of interest (in this case x_j) – all other terms in f can be treated like a constant.

Gradient Descent on Linear Regression

Update all parameters θ in the opposite direction to steepest ascent to minimise a loss function, J:

$$\theta_j^{new} = \theta_j^{old} - \alpha \frac{\partial J(\theta_j^{old})}{\partial \theta_j^{old}}$$

$\alpha > 0$, the "learning rate" is a scalar value

Partial derivative

Just do usual derivative in terms of a single parameter of interest (in this case θ_j) – all other terms in J can be treated like a constant.

Example: Partial derivatives in practice

Using the (mean squared error) loss function:

$$J(\theta) = \frac{1}{2m}\sum_{i=1}^{m}\left(h\left(x^{(i)};\theta\right) - y^{(i)}\right)^2$$

With only 1 data point (m=1):

$$J(\theta) = \frac{1}{2}(h(x;\theta) - y)^2$$

Given that $h(x;\theta) = \theta_0 + \theta_1 x$, has two parameters θ_0 and θ_1, what do we need to be able to apply the general gradient descent rule?

$$\theta_j^{new} = \theta_j^{old} - \alpha \frac{\partial J(\theta)}{\partial \theta_j^{old}}$$

Hint: use the chain rule

$$\frac{d\{f(g(x))\}}{\mathrm{dx}} = f'(g(x))g'(x)$$

Linear regression and gradient descent

Given a loss function with m=1 :

$$J(\theta) = \frac{1}{2}(h_\theta(x) - y)^2$$

$$\frac{d\{f(g(x))\}}{\mathrm{dx}} = f'(g(x))g'(x)$$

The partial derivatives:

$$\frac{\boldsymbol{\partial} J(\theta)}{\boldsymbol{\partial}\theta_j} = \frac{1}{2}\frac{\boldsymbol{\partial}}{\boldsymbol{\partial}\theta_j}(h_\theta(x) - y)^2$$

$$= \frac{1}{2}2(h_\theta(x) - y)\frac{\boldsymbol{\partial}}{\boldsymbol{\partial}\theta_j}(h_\theta(x) - y)$$

$$= (h_\theta(x) - y)\frac{\boldsymbol{\partial}}{\boldsymbol{\partial}\theta_j}(h_\theta(x) - y)$$

Therefore, we just need to find the partial derivative of $\big((h(x;\theta) - y\big)$ w.r.t. parameters $\theta_0$and θ_1.

Linear regression and gradient descent

Find the partial derivative of $\left((h(x;\theta) - y\right)$ w.r.t. parameters θ_0 and θ_1

Given hypothesis, $h_\theta(x) = h(x;\theta) = \theta_0 + \theta_1 x$:

$$\frac{\partial}{\partial \theta_0}(h_\theta(x) - y)$$
$$= \frac{\partial}{\partial \theta_0}(\theta_0 + \theta_1 x - y)$$
$$= 1$$

$$\frac{\partial}{\partial \theta_1}(h_\theta(x) - y)$$
$$= \frac{\partial}{\partial \theta_1}(\theta_0 + \theta_1 x - y)$$
$$= x$$

Now use these to calculate the partial derivatives of our loss function:

(with m=1)

$$\frac{\partial J(\theta)}{\partial \theta_j} = (h_\theta(x) - y)\frac{\partial}{\partial \theta_j}(h_\theta(x) - y)$$

$$\frac{\partial J(\theta)}{\partial \theta_0} = (h_\theta(x) - y) * 1$$

$$\frac{\partial J(\theta)}{\partial \theta_1} = (h_\theta(x) - y) * x$$

Linear regression and gradient descent

Let's scale this up to handle more data, i.e. $m > 1$:

$$J(\theta) = \frac{1}{2m}\sum_{i=1}^{m}\left(h\left(x^{(i)};\theta\right) - y^{(i)}\right)^2$$

$m = 1$

$$\frac{\partial J(\theta)}{\partial \theta_0} = (h(x;\theta) - y)$$

$$\frac{\partial J(\theta)}{\partial \theta_1} = (h(x;\theta) - y)x$$

$m > 1$

$$\frac{\partial J(\theta)}{\partial \theta_0} = \frac{1}{m}\sum_{i=1}^{m}\left(h\left(x^{(i)};\theta\right) - y^{(i)}\right)$$

$$\frac{\partial J(\theta)}{\partial \theta_1} = \frac{1}{m}\sum_{i=1}^{m}\left(h\left(x^{(i)};\theta\right) - y^{(i)}\right)x^{(i)}$$

Linear regression and gradient descent

$$\frac{\partial J(\theta)}{\partial \theta_0} = \frac{1}{m}\sum_{i=1}^{m}\left(h\left(x^{(i)};\theta\right) - y^{(i)}\right)$$

$$\frac{\partial J(\theta)}{\partial \theta_1} = \frac{1}{m}\sum_{i=1}^{m}\left(h\left(x^{(i)};\theta\right) - y^{(i)}\right)x^{(i)}$$

Use these partial derivatives with the Gradient Descent Update rule:

$$\theta_j^{new} = \theta_j^{old} - \alpha \frac{\partial f\left(\theta_j^{old}\right)}{\partial \theta_j^{old}}$$

$$\theta_0^{new} = \theta_0^{old} - \frac{\alpha}{m}\sum_{i=1}^{m}\left(h\left(x^{(i)};\theta_0^{old}\right) - y^{(i)}\right)$$

$$\theta_1^{new} = \theta_1^{old} - \frac{\alpha}{m}\sum_{i=1}^{m}\left(h\left(x^{(i)};\theta_1^{old}\right) - y^{(i)}\right)x_1^{(i)}$$

Linear regression and gradient descent

Quiz: which of these is the correct ordering (if implemented in code)?

$$v_0 = \theta_0 - \alpha \frac{\partial J(\theta_0,\theta_1)}{\partial \theta_0}$$
$$\theta_0 = v_0$$
$$v_1 = \theta_1 - \alpha \frac{\partial J(\theta_0,\theta_1)}{\partial \theta_1}$$
$$\theta_1 = v_1$$

(a)

$$v_0 = \theta_0 - \alpha \frac{\partial J(\theta_0,\theta_1)}{\partial \theta_0}$$
$$v_1 = \theta_1 - \alpha \frac{\partial J(\theta_0,\theta_1)}{\partial \theta_1}$$
$$\theta_0 = v_0$$
$$\theta_1 = v_1$$

(b)

(v_0 and v_1 are temporary variables)

Linear regression and gradient descent

Quiz: which of these is the correct ordering (if implemented in code)?

$$v_0 = \theta_0 - \alpha \frac{\partial J(\theta_0,\theta_1)}{\partial \theta_0}$$
$$\theta_0 = v_0$$
$$v_1 = \theta_1 - \alpha \frac{\partial J(\theta_0,\theta_1)}{\partial \theta_1}$$
$$\theta_1 = v_1$$

(a)

$$v_0 = \theta_0 - \alpha \frac{\partial J(\theta_0,\theta_1)}{\partial \theta_0}$$
$$v_1 = \theta_1 - \alpha \frac{\partial J(\theta_0,\theta_1)}{\partial \theta_1}$$
$$\theta_0 = v_0$$
$$\theta_1 = v_1$$

(b)

(v_0 and v_1 are temporary variables)

Linear regression and gradient descent

Given a function $f(x_1, x_2, \dots, \theta_1, \dots \theta_N)$ where x_i are *known*, gradient descent aims to optimise (find the minimum) of f by iteratively updating the N unknown parameters $\theta_1 \dots \theta_N$

1. Take the partial derivative of f with respect to $\theta_1 \dots \theta_N$
2. Use the general gradient descent rule to update each θ_i (for $i = 1 \dots N$):

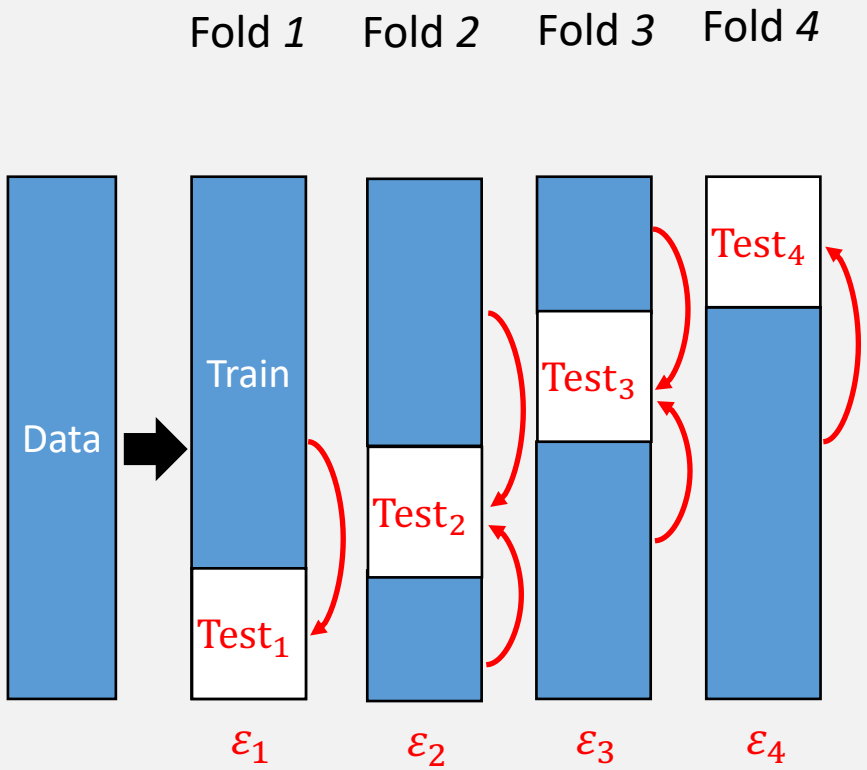
$$\theta_i^{new} = \theta_i^{old} - \alpha \frac{\partial f}{\partial \theta_i}$$

Alternatively update them all together as vector values:

$$\begin{bmatrix} \theta_1^{new} \\ \vdots \\ \theta_N^{new} \end{bmatrix} = \begin{bmatrix} \theta_1^{old} \\ \vdots \\ \theta_N^{old} \end{bmatrix} - \alpha \nabla f$$

If f is convex, gradient descent should always find the optimal solution.

Revision: (n-fold) Nested Cross Validation



Revision: (n-fold) Nested Cross Validation

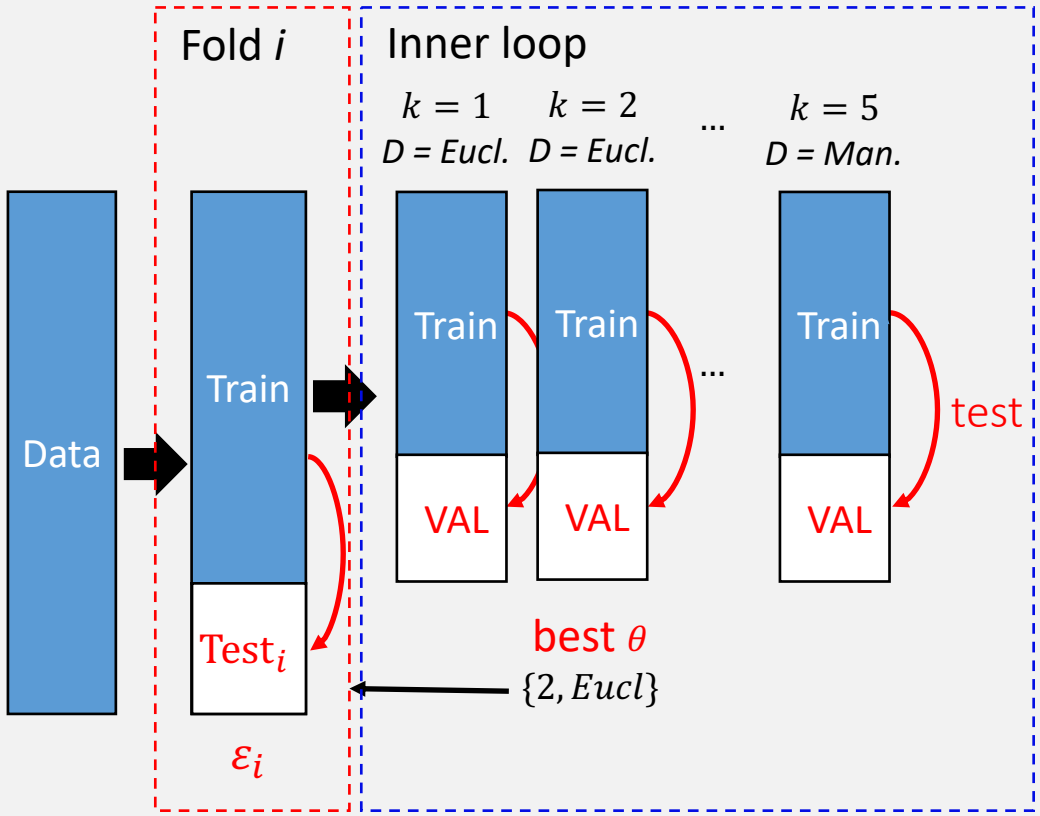
How to pick the best from a set of parameters?

- e.g., θ={ k={1,2,3,4,5}, distance={Euclidean, Manhattan} }

For every fold, *i*:
Create Train and Validation
Loop through params θ:
- Train on Train
- Test on VAL
  - Save best θ

Evaluate Test_i with best θ

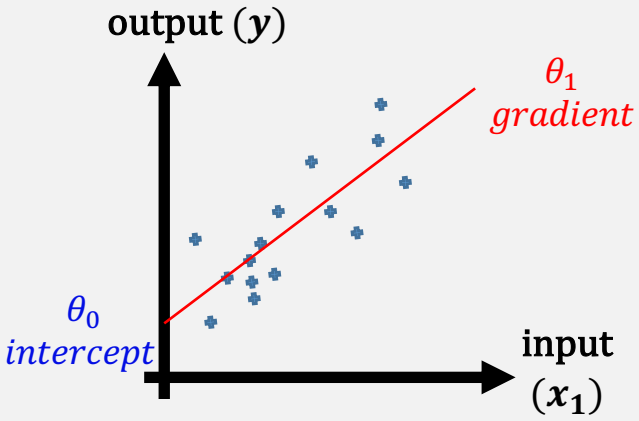
Within each CV fold, split training data into training and validation sub-sets to tune hyperparameters.



Revision: Linear regression

$$h_\theta(x) = \theta_0 + \theta_1 x_1$$

$$= \sum_{j=0}^{1} \theta_j x_j \quad \text{(with } x_0 = 1\text{)}$$

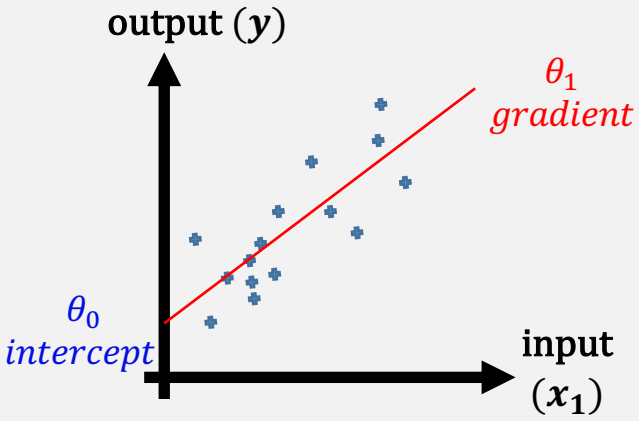


Revision: Linear regression

$$h_\theta(x) = \theta_0 + \theta_1 x_1$$

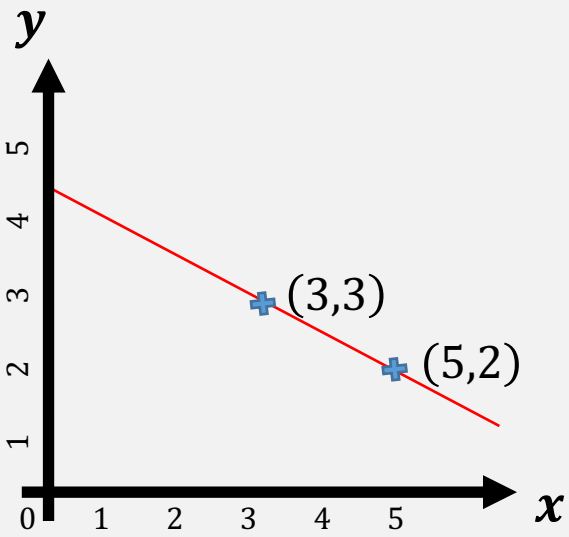
$$= \sum_{j=0}^{1} \theta_j x_j \quad \text{(with } x_0 = 1\text{)}$$

$$= [1 \quad x] \begin{bmatrix} \theta_0 \\ \theta_1 \end{bmatrix}$$



Revision Quiz: Linear systems

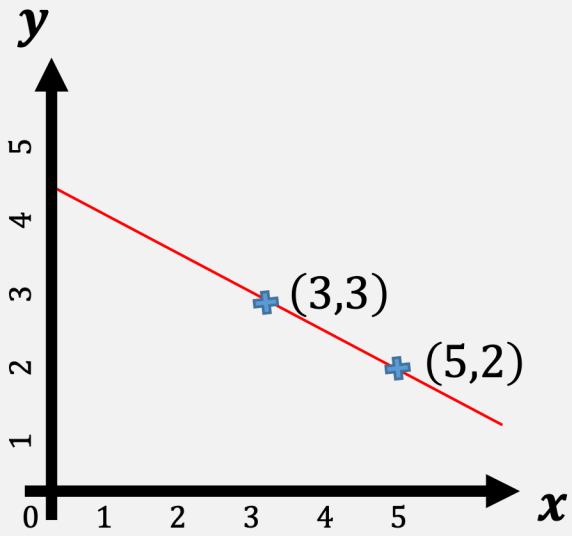
Calculate the parameters for the line passing through [3,3] and [5,2]?



Revision Quiz: Linear systems

$$\begin{bmatrix} 1 & x^{(1)} \\ 1 & x^{(2)} \end{bmatrix} \begin{bmatrix} \theta_0 \\ \theta_1 \end{bmatrix} = \begin{bmatrix} y^{(1)} \\ y^{(2)} \end{bmatrix}$$

Calculate the parameters for the line passing through [3,3] and [5,2]?



$$\left.\begin{aligned} \theta_0 + \theta_1 x &= y \\ \theta_0 + 3\theta_1 &= 3 \\ \theta_0 + 5\theta_1 &= 2 \end{aligned}\right\} \leftrightarrow \underbrace{\begin{bmatrix} 1 & 3 \\ 1 & 5 \end{bmatrix} \begin{bmatrix} \theta_0 \\ \theta_1 \end{bmatrix} = \begin{bmatrix} 3 \\ 2 \end{bmatrix}}_{X \quad \theta \quad y}$$

$$\theta_0 = 3 - 3\theta_1 = 2 - 5\theta_1$$

$$2\theta_1 = -1$$

$$\Rightarrow \theta_1 = -\frac{1}{2}$$

$$\Rightarrow \theta_0 = 2 + \frac{5}{2} = 4.5$$

$$\theta = X^{-1}y$$

```
import numpy as np
X=np.array([[1, 3],[1, 5]])
y=np.array([3, 2])
answer2 = np.linalg.inv(X).dot(y)
```