CITY UNIVERSITY LONDON

Computer Vision
INM460 / IN3060

Drs Greg Slabaugh and Sepehr Jalali

Mathematics Primer

with examples in Matlab

Overview of this session

- Mathematics of computer vision
  - Linear algebra
    - Vectors
    - Matrices
    - Homogenous coordinates
  - Calculus
    - Derivatives
    - Integrals

Scalars

- A scalar is simply a number
- Mathematically, it is denoted with a non-bold font, for example
  - $x = 6.0$
  - $y = 5.0$
- Scalars can be added, subtracted, multiplied, and divided (except divide by 0):
  - $x + y = 11.0$
  - $x - y = 1.0$
  - $x * y = 30.0$
  - $x / y = 1.2$

In Matlab

```
x = 6
y = 5
x + y
x - y
x * y
x / y
```

Vectors

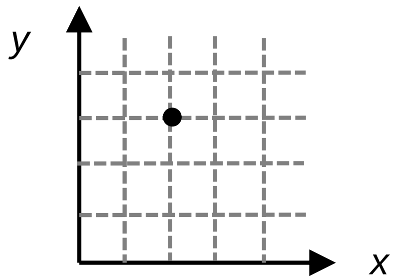
- Notation: using a bold font
- Used to represent, in *N* (typically, 2, 3, or 4) dimensions:
  - Position of a point in space (from the origin)
  - Direction
  - Colour: E.g., red, green, blue
- An *N* dimensional vector can be written as

$$\mathbf{v} = \begin{bmatrix} v_1 \\ v_2 \\ \vdots \\ v_N \end{bmatrix}$$

components (or elements)

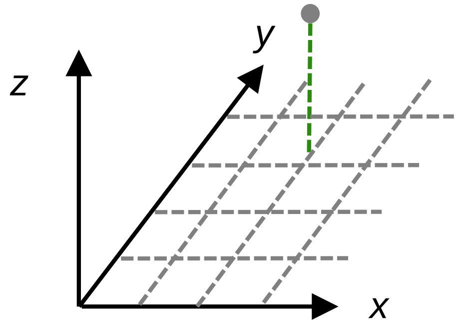
Vector examples

- Represent the point (2, 3) in 2D using a vector



$$\mathbf{p}_1 = \begin{bmatrix} 2 \\ 3 \end{bmatrix}$$

- Represent the point (2, 3, 3) in 3D using a vector



$$\mathbf{p}_2 = \begin{bmatrix} 2 \\ 3 \\ 3 \end{bmatrix}$$

In Matlab

```
p1 = [2; 3]
p2 = [2; 3; 3]
```

Column vectors and row vectors

- A *column vector* is a vector that has all components in a vertical column

$$\mathbf{v} = \begin{bmatrix} v_1 \\ v_2 \\ \vdots \\ v_N \end{bmatrix}$$

- A *row vector* has components in a horizontal row.
- A *transpose* changes column vector to a row vector, and vice-versa.

$$\mathbf{v}^T = [v_1, v_2, \cdots v_N]$$

In Matlab

```
v1 = 1
v2 = 2
v3 = 3
v = [v1; v2; v3]
vt = v'
```

CITY UNIVERSITY LONDON

Vector arithmetic

- Vectors can be added and subtracted, to form a new vector

$$\mathbf{p}+\mathbf{q}=\begin{bmatrix} p_1+q_1 \\ p_2+q_2 \\ \vdots \\ p_N+q_N \end{bmatrix} \qquad \mathbf{p}-\mathbf{q}=\begin{bmatrix} p_1-q_1 \\ p_2-q_2 \\ \vdots \\ p_N-q_N \end{bmatrix}$$

- For example, if **p** = [2, 2, 2]^T and **q** = [1, 0, 1]^T, determine **p** – **q**.

$$\mathbf{p}-\mathbf{q}=\begin{bmatrix} 2 \\ 2 \\ 2 \end{bmatrix}-\begin{bmatrix} 1 \\ 0 \\ 1 \end{bmatrix}=$$

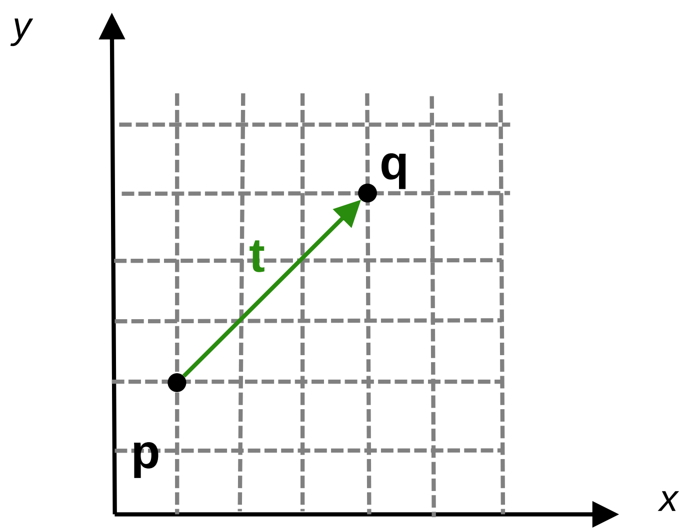
In Matlab

```
p = [2, 2, 2]';
q = [1, 0, 1]';
p - q
```

Vector addition and subtraction

- Often used for translation:
  - Example: move the point **p** = $[1, 2]^T$ by adding a displacement **t =** $[3, 3]^T$ to form the point **q.**

$$\mathbf{q} = \mathbf{p} + \mathbf{t} = \begin{bmatrix} 1 \\ 2 \end{bmatrix} + \begin{bmatrix} 3 \\ 3 \end{bmatrix} = \begin{bmatrix} 4 \\ 5 \end{bmatrix}$$



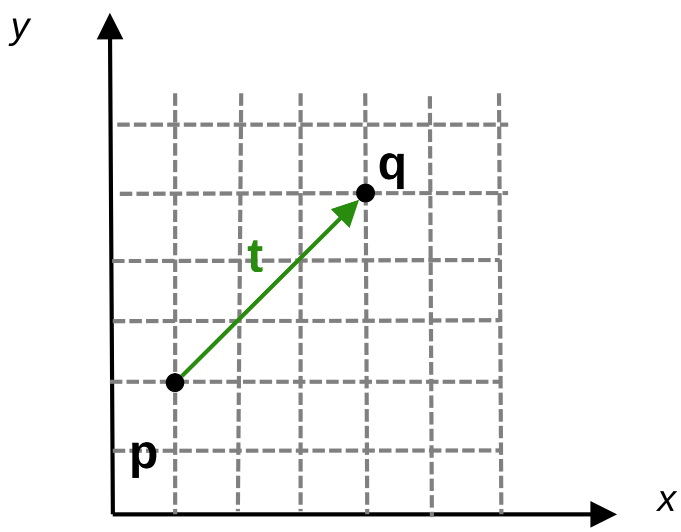
In Matlab

```
p = [1, 2]';
t = [3, 3]';
q = p + t
```

Vector addition and subtraction

- Also commonly used to find a vector between points:
  - Example: find the vector **t** between points **p** and **q** and originating from **p**.

$$\mathbf{t} = \mathbf{q} - \mathbf{p} = \begin{bmatrix} 4 \\ 5 \end{bmatrix} - \begin{bmatrix} 1 \\ 2 \end{bmatrix} = \begin{bmatrix} 3 \\ 3 \end{bmatrix}$$



In Matlab

```
p = [1, 2]';
q = [4, 5]';
t = q - p
```

Scalar multiplication

- Scalar multiplication: multiplies each component of the vector by a scalar.

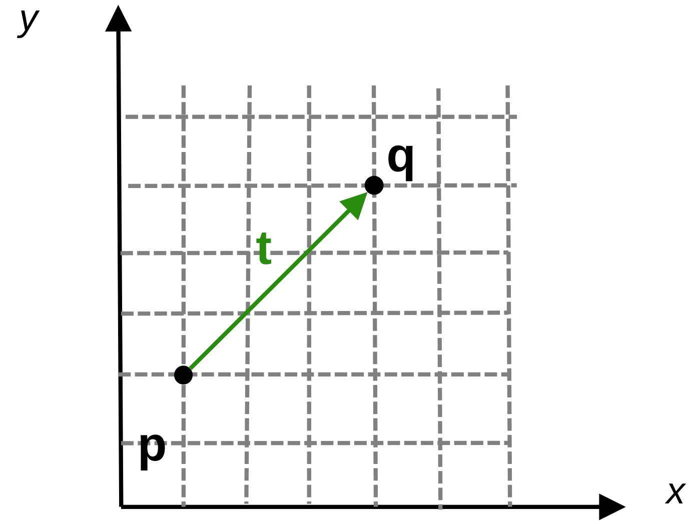
$$a\mathbf{v} = \mathbf{v}a = \begin{bmatrix} av_1 \\ av_2 \\ \vdots \\ av_N \end{bmatrix}$$

In Matlab

```
v = [1, 2, 3]';
a = 4;
a*v
```

Vector length

- A vector has a length (or *magnitude*) given by the expression: $||\mathbf{v}|| = \sqrt{\sum_{i=1}^{N} v_i^2}$
- Note that the vector length is a *scalar*.
- What is the length of **t?**



$$\mathbf{t} = \begin{bmatrix} 3 \\ 3 \end{bmatrix}$$

$$||\mathbf{t}|| = \sqrt{3^2 + 3^2} = \sqrt{18} = 3\sqrt{2}$$

In Matlab
(Two ways)

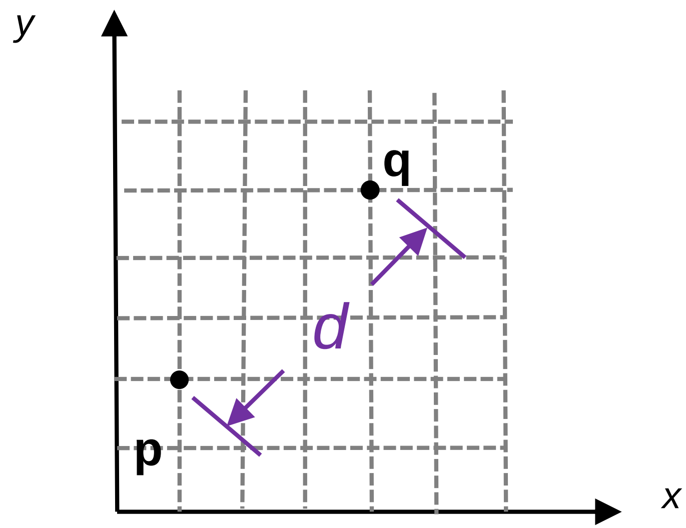
```
p = [1, 2]';
q = [4, 5]';
t = q - p
lengthT = sqrt(sum(t.^2))
lengthT = norm(t)
```

CITY UNIVERSITY LONDON

Distance between two points

- The distance between two points is simply the length of the vector between the two points, as described on the previous slide!
- It can be expressed as

$$d = \sqrt{\sum_i (p_i - q_i)^2}$$



In Matlab
(Two ways)

```
p = [1, 2]';
q = [4, 5]';
d = norm(p-q)
d = pdist2(p', q')
```

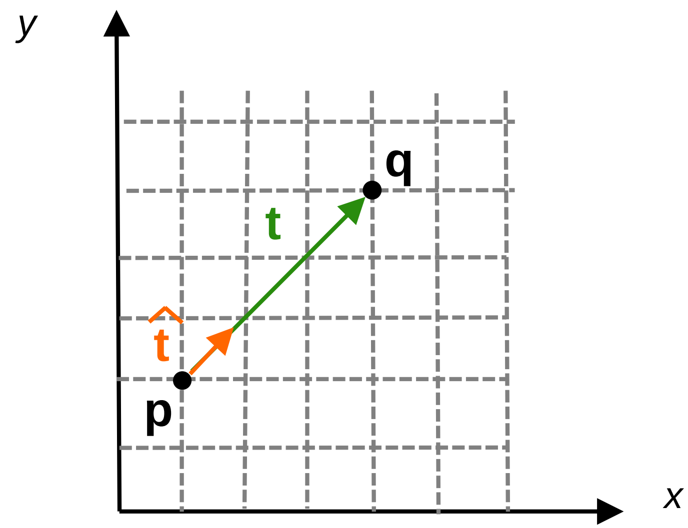
Vector normalisation

- Normalisation scales a vector so that it has unit length. That is, the length of the vector is one after normalisation.
- Any vector with at least one non-zero component can be normalised.

$$\frac{\mathbf{v}}{||\mathbf{v}||}$$

- Determine $\hat{\mathbf{t}}$, by normalising $\mathbf{t}$

$$\hat{\mathbf{t}} = \frac{\mathbf{t}}{||\mathbf{t}||} = \frac{1}{3\sqrt{2}} \left[\begin{array}{c} 3 \\ 3 \end{array} \right] = \left[\begin{array}{c} \frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} \end{array} \right]$$



In Matlab

```
p = [1, 2]';
q = [4, 5]';
t = q - p;
that = t / norm(t)
```

Element-wise product

- The element-wise product of two vectors is the product of the matching elements of each vector. The result is a vector.
- In this module, we will use Matlab syntax of .* to represent the element-wise product.

  - Example: What is the element-wise product of vectors **p** = $[1, 2, 3]^T$ and **q** = $[4, 5, 6]^T$?

$$\begin{bmatrix} 1 \\ 2 \\ 3 \end{bmatrix} .* \begin{bmatrix} 4 \\ 5 \\ 6 \end{bmatrix} = \begin{bmatrix} 4 \\ 10 \\ 18 \end{bmatrix}$$

In Matlab

```
p = [1, 2, 3]';
q = [4, 5, 6]';
p .* q
```

Dot product

- Dot product
  - Definition $$\mathbf{p} \cdot \mathbf{q} = \sum_{i=1}^{N} p_i q_i$$
  - Multiplies the *i*th component of each vector together, then takes the sum.
  - Note that the dot product is a *scalar*.
  - Example: What is the dot product of vectors **p** = $[3, 2, 1]^T$ and **q** = $[1, 0, -1]^T$?

Answer: (3)(1) + (2)(0) + (1)(-1) = 2

In Matlab

```
p = [3, 2, 1]';
q = [1, 0, -1]';
dot(p, q)
```

Homogenous coordinates

- In computer vision (and computer graphics), often homogeneous coordinates are used to represent points. This is useful when dealing with transformations (like rotation or translation).
- Homogeneous coordinates add another dimension (or element) to the vector representing a point. For example, a 2D point $[x, y]^T$ is represented in homogeneous coordinates as

$$\begin{bmatrix} x \\ y \end{bmatrix} \Rightarrow \begin{bmatrix} x \\ y \\ w \end{bmatrix}$$

- Two sets of homogenous coordinates represent the same point if one is a multiple of another. For example, $[1, 2, 1]^T = [2, 4, 2]^T$
- Dividing by w (when non-zero) puts the point in Cartesian coordinates. [x, y, w]T = [x/w, y/w, 1]T

Matrices

- Of fundamental importance in computer vision
- A matrix **F** is a rectangular array of numbers (elements) that has N rows and M columns (matrix with size N x M). For example, here is a 3 x 4 matrix:

$$\mathbf{F} = \begin{bmatrix} F_{11} & F_{12} & F_{13} & F_{14} \\ F_{21} & F_{22} & F_{23} & F_{24} \\ F_{31} & F_{32} & F_{33} & F_{34} \end{bmatrix}$$

index

element

- Matrices for which N = M are called *square*.
- Matrices are typically used to represent transformations (e.g., between coordinate systems, rotation, scale, translation, etc.)
- In computer vision, matrices are often size 3 x 3 or 3 x 4.

Matrices

- A vector is simply a matrix with N or M equal to 1. For example, this is column vector is a 3 x 1 matrix:

$$\mathbf{p} = \begin{bmatrix} p_x \\ p_y \\ p_z \end{bmatrix}$$

- Two matrices are equal if and only if all elements are equal. Note this requires the matrices have the same size. For example,

$$\mathbf{A} = \begin{bmatrix} 1 & 3 \\ 3 & 8 \end{bmatrix} \qquad \mathbf{B} = \begin{bmatrix} 8 & 3 \\ 3 & 8 \end{bmatrix} \qquad \mathbf{A} \neq \mathbf{B}$$

CITY UNIVERSITY LONDON

Matrix diagonal

- The matrix diagonal is the set of matrix elements along the diagonal of the (typically square) matrix.
- A diagonal matrix has
  - non-zero elements along the diagonal
  - zero off diagonal
- An example of a diagonal matrix:

$$\mathbf{A} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 2 & 0 & 0 \\ 0 & 0 & 3 & 0 \\ 0 & 0 & 0 & 4 \end{bmatrix}$$

In Matlab
(Two ways)

```
A = diag([1, 2, 3, 4])
A = [1, 0, 0, 0; 0 2 0 0; 0 0 3 0; 0 0 0 4]
```

Matrix transpose

- Transpose: switches rows with columns $F_{ij}^T = F_{ji}$

$$\mathbf{F} = \begin{bmatrix} F_{11} & F_{12} & F_{13} & F_{14} \\ F_{21} & F_{22} & F_{23} & F_{24} \\ F_{31} & F_{32} & F_{33} & F_{34} \end{bmatrix}$$

$$\mathbf{F}^T = \begin{bmatrix} F_{11} & F_{21} & F_{31} \\ F_{12} & F_{22} & F_{32} \\ F_{13} & F_{23} & F_{33} \\ F_{14} & F_{24} & F_{34} \end{bmatrix}$$

Addition, subtraction: performed element-wise. Requires matrices to have the same size.

$$\mathbf{F} + \mathbf{G} = \begin{bmatrix} F_{11} + G_{11} & F_{12} + G_{12} & \cdots & F_{1M} + G_{1M} \\ F_{21} + G_{21} & F_{22} + G_{22} & \cdots & F_{2M} + G_{2M} \\ \vdots & \vdots & \ddots & \vdots \\ F_{N1} + G_{N1} & F_{N2} + G_{N2} & \cdots & F_{NM} + G_{NM} \end{bmatrix}$$

In Matlab

```
F = [1, 2; 3, 4];
F'
G = [1, 1; 1, 1];
F + G
```

Scalar / matrix multiplication

- Scalar / matrix multiplication multiplies each element with scalar

$$a\mathbf{F} = \mathbf{F}a = \begin{bmatrix} aF_{11} & aF_{12} & \cdots & aF_{1M} \\ aF_{21} & aF_{22} & \cdots & aF_{2M} \\ \vdots & \vdots & \ddots & \vdots \\ aF_{N1} & aF_{N2} & \cdots & aF_{NM} \end{bmatrix}$$

In Matlab	`F = [1, 2; 3, 4]`
	`a = 2;`
	`a*F`

Element-wise product

- There is also an element-wise product for matrices, which simply multiplies the matching elements of each matrix.
  - Example: What is **A** .* **B**, for $\mathbf{A} = \begin{bmatrix} 1 & 2 \\ 3 & 4 \end{bmatrix}$ $\mathbf{B} = \begin{bmatrix} 5 & 6 \\ 7 & 8 \end{bmatrix}$?

$$\mathbf{A}. * \mathbf{B} = \begin{bmatrix} 5 & 12 \\ 21 & 32 \end{bmatrix}$$

In Matlab

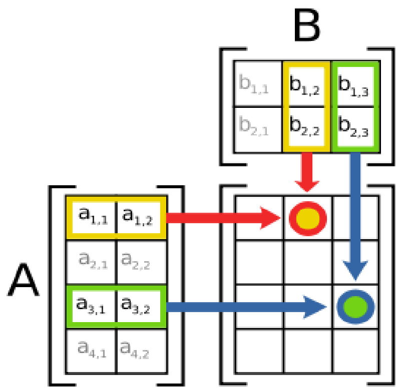
```
A = [1, 2; 3, 4];
q = [5, 6; 7, 8]';
A .* B
```

Matrix multiplication

- Two matrices can be multiplied together **only** when the number of columns in the matrix on the left equals the number of rows in the matrix on the right.
- For example, suppose
  - matrix A is N x M
  - matrix B is L x R

  ⇨ The matrix product **AB** is only defined in M = L. The resulting matrix will be of size N x R.
- It is performed by multiplying each row of A by each column of B

$$[\mathbf{AB}]_{ij} = \sum_{k=1}^{M} A_{ik} B_{kj}$$



Matrix multiplication example

- Determine **AB**

$$\mathbf{A} = \begin{bmatrix} 1 & 2 \\ 3 & 4 \end{bmatrix} \qquad \mathbf{B} = \begin{bmatrix} 5 & 6 \\ 7 & 8 \end{bmatrix}$$

$$\begin{array}{cc} & \begin{bmatrix} 5 & 6 \\ 7 & 8 \end{bmatrix} \leftarrow \mathbf{B} \\ \begin{bmatrix} 1 & 2 \\ 3 & 4 \end{bmatrix} & \begin{bmatrix} & \\ & \end{bmatrix} \end{array}$$

$\mathbf{A}$

Matrix multiplication example

- Multiplication example

$$\mathbf{A} = \begin{bmatrix} 1 & 2 \\ 3 & 4 \end{bmatrix} \quad \mathbf{B} = \begin{bmatrix} 5 & 6 \\ 7 & 8 \end{bmatrix}$$

$$\begin{bmatrix} 5 & 6 \\ 7 & 8 \end{bmatrix}$$

$$\begin{bmatrix} 1 & 2 \\ 3 & 4 \end{bmatrix} \begin{bmatrix} (1)(5)+(2)(7) & \\ & \end{bmatrix}$$

Matrix multiplication example

- Multiplication example $\mathbf{A} = \begin{bmatrix} 1 & 2 \\ 3 & 4 \end{bmatrix}$ $\mathbf{B} = \begin{bmatrix} 5 & 6 \\ 7 & 8 \end{bmatrix}$

$$\begin{array}{cc} & \begin{bmatrix} 5 & 6 \\ 7 & 8 \end{bmatrix} \\ \begin{bmatrix} 1 & 2 \\ 3 & 4 \end{bmatrix} & \begin{bmatrix} (1)(5)+(2)(7) & (1)(6)+(2)(8) \\ & \end{bmatrix} \end{array}$$

Matrix multiplication example

- Multiplication example

$$\mathbf{A} = \begin{bmatrix} 1 & 2 \\ 3 & 4 \end{bmatrix} \qquad \mathbf{B} = \begin{bmatrix} 5 & 6 \\ 7 & 8 \end{bmatrix}$$

$$\begin{bmatrix} 5 & 6 \\ 7 & 8 \end{bmatrix}$$

$$\begin{bmatrix} 1 & 2 \\ 3 & 4 \end{bmatrix} \begin{bmatrix} (1)(5)+(2)(7) & (1)(6)+(2)(8) \\ (3)(5)+(4)(7) & \end{bmatrix}$$

Matrix multiplication example

- Multiplication example

$$\mathbf{A} = \begin{bmatrix} 1 & 2 \\ 3 & 4 \end{bmatrix} \qquad \mathbf{B} = \begin{bmatrix} 5 & 6 \\ 7 & 8 \end{bmatrix}$$

$$\begin{array}{cc} & \begin{bmatrix} 5 & 6 \\ 7 & 8 \end{bmatrix} \\ \begin{bmatrix} 1 & 2 \\ 3 & 4 \end{bmatrix} & \begin{bmatrix} (1)(5)+(2)(7) & (1)(6)+(2)(8) \\ (3)(5)+(4)(7) & (3)(6)+(4)(8) \end{bmatrix} \end{array}$$

Matrix multiplication example

- Multiplication example

$$\mathbf{A} = \begin{bmatrix} 1 & 2 \\ 3 & 4 \end{bmatrix} \quad \mathbf{B} = \begin{bmatrix} 5 & 6 \\ 7 & 8 \end{bmatrix}$$

$$\begin{array}{cc} & \begin{bmatrix} 5 & 6 \\ 7 & 8 \end{bmatrix} \\ \begin{bmatrix} 1 & 2 \\ 3 & 4 \end{bmatrix} & \begin{bmatrix} (1)(5)+(2)(7) & (1)(6)+(2)(8) \\ (3)(5)+(4)(7) & (3)(6)+(4)(8) \end{bmatrix} = \begin{bmatrix} 19 & 22 \\ 43 & 50 \end{bmatrix} \end{array}$$

In Matlab

```
A = [1, 2; 3, 4]
B = [5, 6; 7, 8];
A*B
```

(Double) Exercise

- Let $\mathbf{A} = \begin{bmatrix} 1 & 2 \\ 3 & 4 \end{bmatrix}$ and $\mathbf{B} = \begin{bmatrix} 0 & 1 \\ 1 & 1 \end{bmatrix}$

Left side of room:
Determine **AB**

Right side of room:
Determine **BA**

```
ans =
     2     3
     4     7
```

```
ans =
     3     4
     4     6
```

In Matlab

```
A = [1, 2; 3, 4]
B = [0, 1; 1, 1];
A*B
```

In Matlab

```
A = [1, 2; 3, 4]
B = [0, 1; 1, 1];
B*A
```

Matrix multiplication not commutative

- One point to be aware of is that matrix multiplication generally does not commute. That is,

$$\mathbf{AB} \neq \mathbf{BA}$$

- For example,

$$\begin{bmatrix} a & b \\ c & d \end{bmatrix}\begin{bmatrix} e & f \\ g & h \end{bmatrix} = \begin{bmatrix} ae+bg & \ldots \\ \ldots & \ldots \end{bmatrix} \quad \begin{bmatrix} e & f \\ g & h \end{bmatrix}\begin{bmatrix} a & b \\ c & d \end{bmatrix} = \begin{bmatrix} ea+fc & \ldots \\ \ldots & \ldots \end{bmatrix}$$

Pre- and post-multiplication

- There are terms for describing the order of matrix multiplication
  - When a matrix appears on the left, it is *pre-multiplied.*
  - When a matrix appears on the right, it is *post-multiplied.*
- For example, in the matrix product **AB**,
  - **A** is pre-multiplied to **B**
  - **B** is post-multiplied to **A**

Identity matrix

- In scalar multiplication, any number multiplied with 1 results in the original number.
- Similarly, there is a special square matrix, called the *identity matrix*, which when multiplied by another matrix **A**, results in **A**.
- The identity matrix has the form

$$\mathbf{I} = \begin{bmatrix} 1 & 0 & \cdots & 0 \\ 0 & 1 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 1 \end{bmatrix}$$

- For a square matrix **A**, **AI** = **IA** = **A**

In Matlab

```
A = [1, 2; 3, 4]
I = eye(2)
A*I
I*A
```

Inverse matrix

- Also in scalar multiplication, multiplication (1/a)*a = 1; that is, multiplication of a number by its inverse gives identity. Similarly, multiplication a matrix by its inverse gives the identity matrix.
- The inverse of a matrix **A** is denoted $\mathbf{A}^{-1}$. So $\mathbf{AA}^{-1}=\mathbf{I}$, and $\mathbf{A}^{-1}\mathbf{A}=\mathbf{I}$
- If the inverse exists, the matrix is said to be *invertible*. Note: not all matrices are invertible!
- The inverse matrix can be computed for square matrices (number of rows equals number of columns) only.

$$\mathbf{A}=\begin{bmatrix} 9 & -2 \\ -3 & -4 \end{bmatrix} \quad \mathbf{A}^{-1}=\begin{bmatrix} \frac{2}{21} & \frac{-1}{21} \\ \frac{-1}{14} & \frac{-3}{14} \end{bmatrix} \quad \mathbf{AA}^{-1}=\begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$$

In Matlab

```
A = [9, -2; -3, -4];
inv(A)
A*inv(A)
```

System of linear equations

- In a practical context, matrices are used all the time to represent a system of linear equations. In computer vision, this comes up frequently, for example, in
  - Transformations (e.g., warping, rotation, projection)
  - Estimation (least squares, optical flow)
  - (more)
- Example:

$$\begin{aligned} 2x + 3y &= 6 \\ 4x + 9y &= 15 \end{aligned}$$

Can be written as

$$\begin{bmatrix} 2 & 3 \\ 4 & 9 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} 6 \\ 15 \end{bmatrix}$$

CITY UNIVERSITY LONDON

Solving a system of linear equations

- This has the form of **A x** = **b**, where **A** is a 2x2 matrix, and **x** and **b** are 2x1 matrices.

$$\underbrace{\begin{bmatrix} 2 & 3 \\ 4 & 9 \end{bmatrix}}_{\uparrow\ \mathbf{A}} \underbrace{\begin{bmatrix} x \\ y \end{bmatrix}}_{\uparrow\ \mathbf{x}} = \underbrace{\begin{bmatrix} 6 \\ 15 \end{bmatrix}}_{\uparrow\ \mathbf{b}}$$

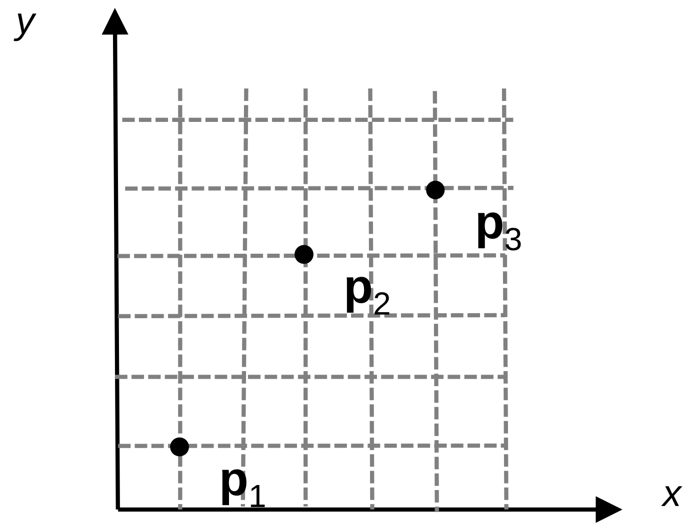
- Using the inverse of **A**, there is a simple solution, namely **x** = **A**$^{-1}$ **b**

In Matlab

```
A = [2, 3; 4, 9];
b = [6, 15]';
x = inv(A)*b
```

Overdetermined system

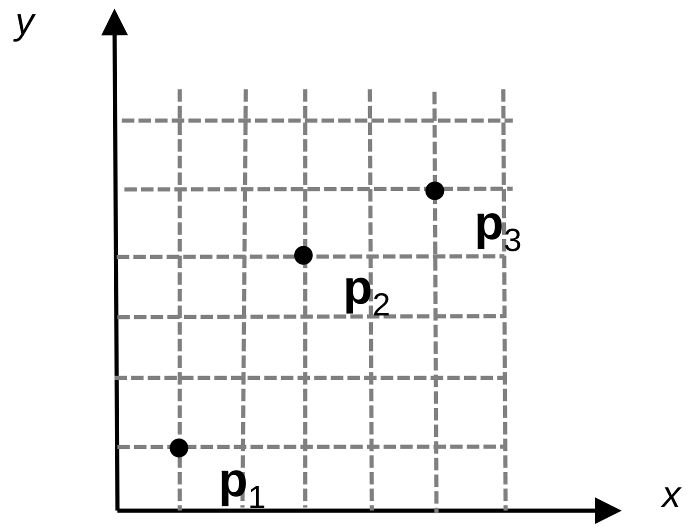
- In the previous example, there were 2 equations and 2 unknowns (x and y). When the number of equations (N) is equal to the number of unknowns (M), we say the system is *determined*.
- If N < M, the system is *undetermined*; there is no unique solution.
- If N > M, the system is *overdetermined*; and we normally will look for the best fitting solution (for example, minimising the error in the least squares sense).
- Example: Find the best fitting line through the points $\mathbf{p}_1 = [1, 1]^T$, $\mathbf{p}_2 = [3, 4]^T$ and $\mathbf{p}_3 = [5, 5]^T$.



CITY UNIVERSITY LONDON

Overdetermined system

- Equation of line: y = mx + b, which is the same as xm + b = y. In our problem, we have two unknowns (m, b) and three equations (as we have three points).



$$\begin{aligned} 1m + b &= 1 \\ 3m + b &= 4 \\ 5m + b &= 5 \end{aligned}$$

$$\underbrace{\begin{bmatrix} 1 & 1 \\ 3 & 1 \\ 5 & 1 \end{bmatrix}}_{\mathbf{A}} \underbrace{\begin{bmatrix} m \\ b \end{bmatrix}}_{\mathbf{c}} = \underbrace{\begin{bmatrix} 1 \\ 4 \\ 5 \end{bmatrix}}_{\mathbf{d}}$$

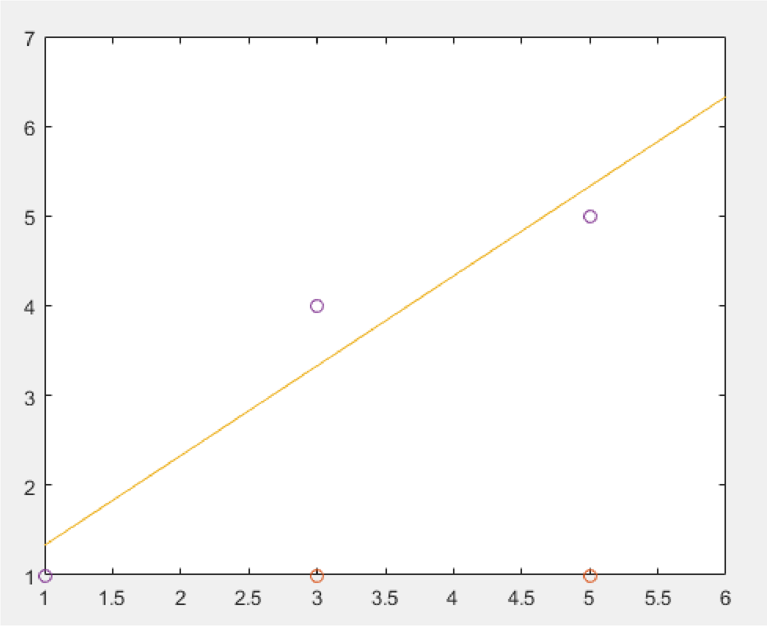
Overdetermined system

- Note: we cannot invert **A** directly, since it is not square (i.e., it does not have an equal number of rows and columns).
- However, we can use the *pseudo-inverse*, $(\mathbf{A}^T\mathbf{A})^{-1}\mathbf{A}^T$
- That is, $\mathbf{c} = (\mathbf{A}^T\mathbf{A})^{-1}\mathbf{A}^T\,\mathbf{d}$. In Matlab the `pinv` function implements this. This provides a least squares solution.

In Matlab

```
A = [1, 1; 3, 1; 5, 1];
d = [1; 4; 5];
c = pinv(A)*d

% visualise the result
m = c(1);
b = c(2);
x = [1:6];
y = m*x+b;
plot(x,y);
hold on;
scatter(A(:, 1), d);
```



Transformations

- Transformations are frequently used in computer vision to transform points, that is, taking points and translating, rotating, scaling, shearing, and/or projecting them.
- Transformations are often represented using matrices, and to transform a point, it is simply a matter of matrix and vector multiplication.
- This takes the form **p**' = **T p**, where **T** is the transformation matrix, **p** is the original point, and **p**' is the transformed result.

CITY UNIVERSITY LONDON

Example (rigid body transformation)

- One type of transformation is a *rigid body transformation*. This applies rotation, and translation.
- Since neither rotation or translation cause distortions to the shape, the object moves rigidly to its new location.

$$\begin{bmatrix} x' \\ y' \\ w' \end{bmatrix} = \begin{bmatrix} \cos\theta & -\sin\theta & t_x \\ \sin\theta & \cos\theta & t_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

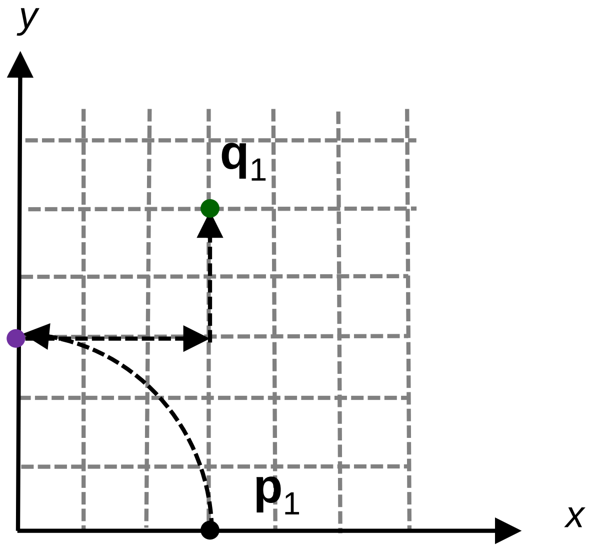
- Performing the matrix multiplication, you can see

$$\begin{aligned} x' &= x\cos\theta - y\sin\theta + t_x \\ y' &= x\sin\theta + y\cos\theta + t_y \\ w' &= 1 \end{aligned}$$

CITY UNIVERSITY LONDON

Example (rigid body transformation)

- Let's use this transformation to rotate by 90 degrees and translate by 3 in x and 2 in y. If we apply this transformation to a point $\mathbf{p}_1 = [3, 0]^T$, we would expect it to move to $\mathbf{q}_1 = [3, 5]^T$.
- Note: Matlab trigonometric functions like `sin` and `cos` expect angles in radians.
- Applying a transformation to all the points making up a shape will transform the shape.

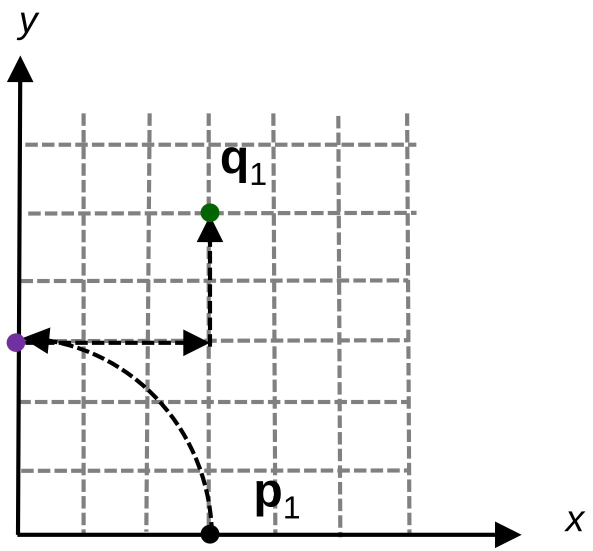


In Matlab

```
p1 = [3, 0, 1]';
tx = 3;
ty = 2;
theta = deg2rad(90);
T = [cos(theta), -sin(theta), tx;...
sin(theta), cos(theta), ty; 0, 0 1];
q1 = T*p1
```

Using affine2d

- Matlab has an `affine2d` class useful for many transformations, including the rigid body transformation.
- You can create an `affine2d` object that includes a transformation using the `affine2d` function. Note this function expects the *transpose* of the matrix described earlier.
- Then you can transform points using the `transformPointsForward` function.



In Matlab

```
tx = 3;
ty = 2;
theta = deg2rad(90);
T = affine2d(...
[cos(theta), -sin(theta), tx;...
sin(theta), cos(theta), ty; ...
0, 0 1]');
[x, y] =transformPointsForward(T,3,0)
```

CITY UNIVERSITY LONDON

Watch out!

- When multiplying matrices **A** and **B** together, multiplication is only possible if the number columns of **A** equals the number of rows of **B**.
  - Example: Multiplying a 3 x 3 matrix with a 3 x 1 matrix (vector)

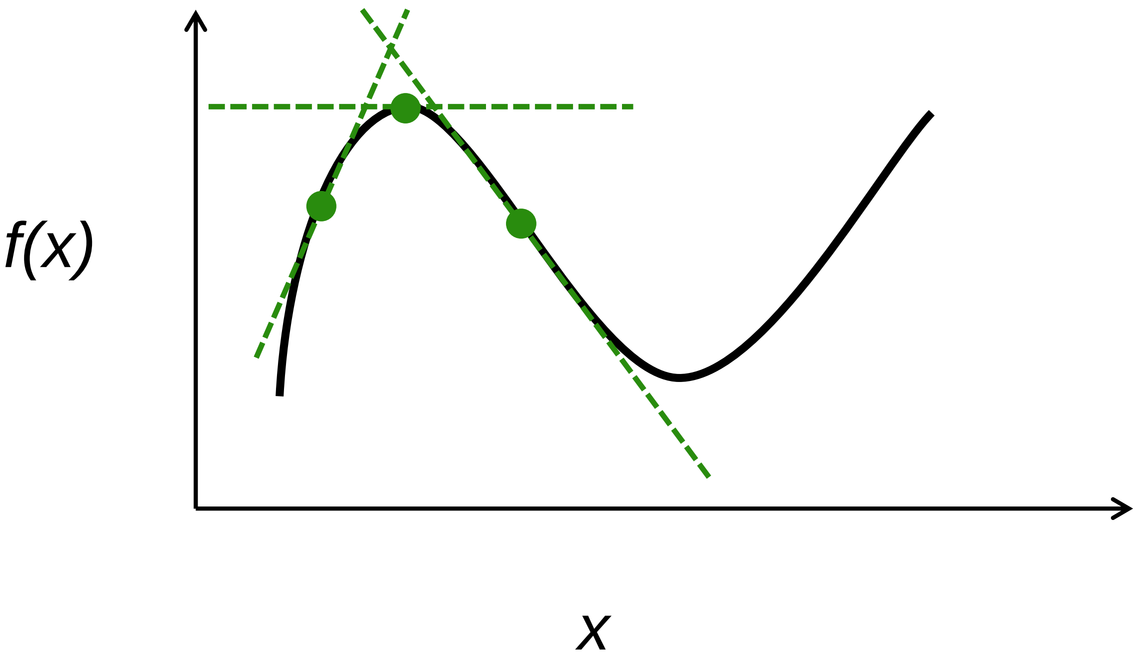
```
A = [1, 2, 3; 4, 5, 6; 7, 8, 9];
B = [3, 3, 3]';
A*B
ans =
    18
    45
    72
```

  - Example: Multiplying a 3 x 3 matrix with a 1 x 3 matrix (vector)

```
A = [1, 2, 3; 4, 5, 6; 7, 8, 9];
B = [3, 3, 3];
A*B
Error using  *
Inner matrix dimensions must agree.
```

Derivatives

- A derivative measures how much a function changes as its input changes.
- For a function *f*(*x*), the derivative (with respect to x) measures how much *f* changes when *x* changes.
- The (first) derivative provides the slope of a curve at a given point.



Notation for derivatives

- There are several ways one might denote a derivative.
- A standard notation is

$$\frac{d}{dx}f(x)$$

this denotes the change in *f*(*x*) as *x* changes.

- This is equivalent to

$$\frac{df(x)}{dx}$$

here the *f*(*x*) simply appears in the numerator.

- For notational convenience, sometimes the *x* in *f*(*x*) is dropped. Since the derivative is with respect to *x*, it is inferred that *f* is a function of *x*.

$$\frac{df}{dx}$$

- Finally, you may see a shorthand notation for a derivative, f_x

CITY UNIVERSITY LONDON

Derivatives

- The process of finding a derivative is called *differentiation*
- There are some basic rules:
  - The derivative of a constant (denoted with *c* below) is 0. This is intuitive because a constant value does not change.

$$\frac{d}{dx}c = 0$$

  - The *power rule* provides derivatives for integer powers of *x*

$$\frac{d}{dx}x^n = nx^{n-1}$$

What is the derivative of f(x) = 5?

$$\frac{d}{dx}5 = 0$$

What is the derivative of f(x) = x^2?

$$\frac{d}{dx}x^2 = 2x$$

Derivatives

- *Constant multiple rule*: If *f* is a differentiable function of *x*, and *c* is a constant, then

$$\frac{d}{dx}(cf) = c\frac{df}{dx}$$

- *Sum rule:* The derivative of a sum is the sum of derivatives

$$\frac{d}{dx}(f_1 + f_2) = \frac{df_1}{dx} + \frac{df_2}{dx}$$

What is the derivative of *f*(x) = $5x^2$ + 5x + 5?

$$\begin{aligned} \frac{d}{dx}(5x^2 + 5x + 5) &= \frac{5x^2}{dx} + \frac{5x}{dx} + \frac{d}{dx}5 \\ &= \frac{d}{dx}(5x^2) + \frac{d}{dx}(5x) + \frac{d}{dx}5 \\ &= 5\frac{d}{dx}(x^2) + 5\frac{d}{dx}(x) + \frac{d}{dx}5 \\ &= 5(2x) + 5(1) + 0 \\ &= 10x + 5 \end{aligned}$$

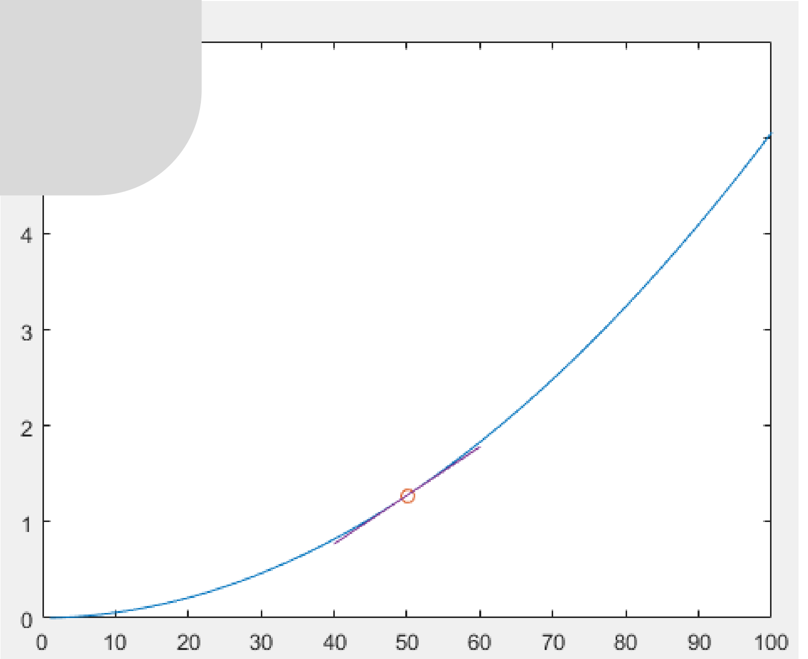
CITY UNIVERSITY LONDON

In Matlab

In Matlab

```
x = [1:100];
f = 5*x.^2+5*x+5;
plot(x,f);
a = 50; % A point to analyse
hold on;
scatter(x(a), f(a));

% Plot tangent at a
f_x = 10*x+5;
m = f_x(a);
xx = [x(a)-10:x(a)+10];
y = f(x(a))+m*(xx-x(a));
plot(xx, y);
```



Second derivative

- You can differentiate a function multiple times. The second derivative is denoted as:

$$\frac{d^2}{dx^2}f\,, \quad \frac{df^2}{dx^2}\text{, or} \quad f_{xx}$$

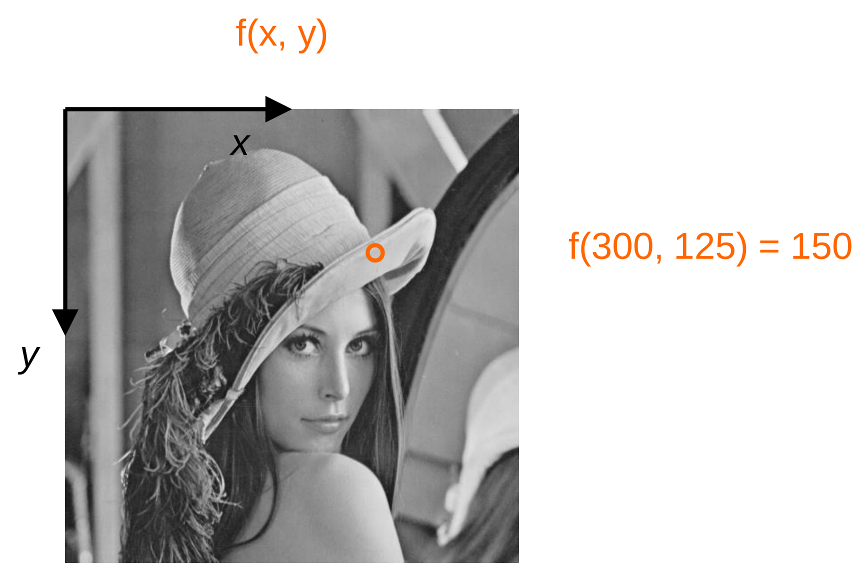
- This is simply achieved by taking the derivative of the derivative.

What is the second derivative of $f(x) = 5x^2 + 5x + 5$?

$$\begin{aligned}
\frac{d^2}{dx^2}(5x^2 + 5x + 5) &= \frac{d}{dx}\left(\frac{d}{dx}(5x^2 + 5x + 5)\right) \\
&= \frac{d}{dx}(10x + 5) \\
&= 10
\end{aligned}$$

Multi-variate case

- The previous examples of derivatives were single variate, since *f*(*x*) describes *f* as function of a single variable *x*.
- In computer vision (and many other fields), often data has multiple dimensions.
- A good example is an image, which describes brightness (or colour) as function of two variables, *x* and *y*.



Partial derivative

- Derivatives can be taken in the multi-variate case as well. The *partial* derivative is typically notated with a curly symbol; or the shorthand notation.

$$\frac{\partial}{\partial x} f, \quad \frac{\partial f}{\partial x}, \text{ or } \quad f_x$$

- When taking a partial derivative, the derivative is taken with respect to a specific variable; the others are held constant.

What is the partial derivatives $\frac{\partial f}{\partial x}, \frac{\partial f}{\partial y}$ of f(x, y) = $x^2 + xy + y^2$?

$$\begin{aligned}
\frac{\partial f}{\partial x} &= \frac{\partial}{\partial x}x^2 + \frac{\partial}{\partial x}xy + \frac{\partial}{\partial x}y^2 \\
&= 2x + (1)y + (0) \\
&= 2x + y \\
\frac{\partial f}{\partial y} &= \frac{\partial}{\partial y}x^2 + \frac{\partial}{\partial y}xy + \frac{\partial}{\partial y}y^2 \\
&= (0) + x(1) + 2y \\
&= x + 2y
\end{aligned}$$

Gradient

- The *gradient* of a multi-variate function *f* is a vector, that has a derivative with respect to each of its arguments. Consider a 2D function f(x, y). The gradient is

$$\nabla f = \left[\frac{\partial f}{\partial x}, \frac{\partial f}{\partial y}\right]^T$$

- The gradient generalises the concept of a derivative to several dimensions. It is a vector that points in the direction of maximal increase of *f*.
- Example: Imagine you're on a walk, you're at a point (x, y), and f(x, y) measures height of the terrain. The gradient will point in the direction of greatest change in height at (x, y)

http://cycloclimbing.com/tour2007/

Laplacian

- The Laplacian is the dot product of the gradient with itself. It is denoted as

$$\Delta f = \nabla^2 f = \nabla f \cdot \nabla f$$

- In 2D, the Laplacian of a function f(x, y) is

$$\begin{aligned}\Delta f &= \left[\frac{\partial f}{\partial x}, \frac{\partial f}{\partial y}\right]^T \cdot \left[\frac{\partial f}{\partial x}, \frac{\partial f}{\partial y}\right]^T \\ &= \frac{\partial^2 f}{\partial x^2} + \frac{\partial^2 f}{\partial y^2}\end{aligned}$$

Mixed partial derivative

- You can take a mixed derivative, that is a derivative using multiple variables. For example, $\frac{\partial^2 f}{\partial x \partial y}$, or f_{xy}

  denotes the derivative of *f* with respect to both *x* and *y*. The order in which you take the derivatives does not matter (e.g, *x* first then, *y*; or *y* first, then *x*).

What is the mixed partial derivative f_{xy} of $f(x, y) = x^2 + xy + y^2$?

$$
\begin{aligned}
\frac{\partial^2 f}{\partial x \partial y}(x^2 + xy + y^2) &= \frac{\partial}{\partial x}\left(\frac{\partial}{\partial y}(x^2 + xy + y^2)\right) \\
&= \frac{\partial}{\partial x}(x + 2y) \\
&= 1
\end{aligned}
$$

Taylor series expansion

- A function $f(x)$ can be approximated at a point a using a Taylor series expansion.
- This expresses the function $f(x)$ at a using derivatives.

$$f(x) = f(a) + f_x(a)(x-a) + \frac{f_{xx}(a)}{2}(x-a)^2 + \ldots$$

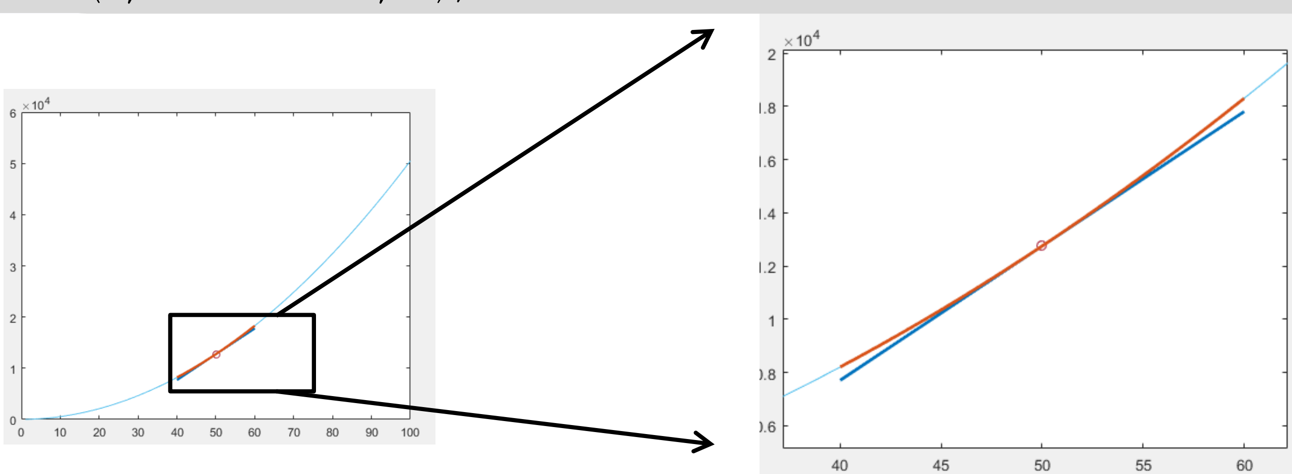
- The … represents higher order terms, which for simplicity are often ignored as we're interested in an approximation.

Taylor series expansion

In Matlab

```
x = [1:100];
f = 5*x.^2+5*x+5;
f_x = 10*x + 5;
f_xx = 10*ones(size(x));
plot(x,f); hold on;
a = 50; % A point to analyse
scatter(x(a), f(a));

x = [x(a)-10:x(a)+10];
% First order approximation at a
f_first = f(a) + f_x(a)*(x-a);
h = plot(x, f_first);
set(h, 'LineWidth', 2);
% Second order approximation at a
f_second = f(a) + f_x(a)*(x-a)+0.5*f_xx(a)*(x-a).^2;
h = plot(x, f_second);
set(h, 'LineWidth', 2);
```



CITY UNIVERSITY LONDON

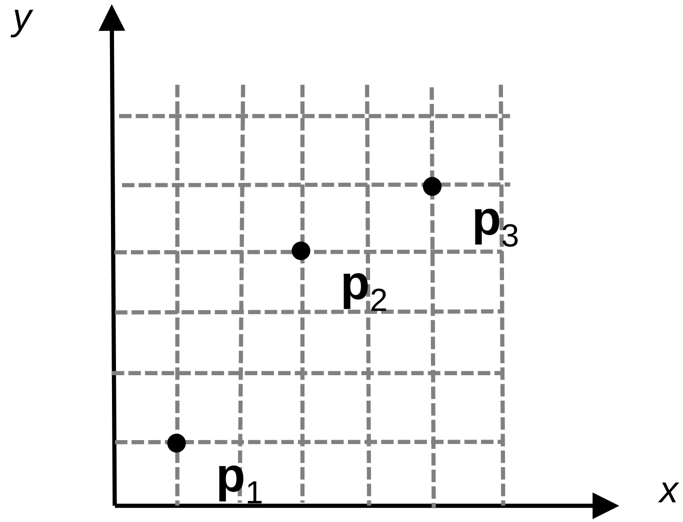
Optimisation

- We've already seen an example of optimisation – using least squares to solve an overdetermined system of linear equations. This trick is incredibly useful.
- Matlab has an Optimisation Toolbox that can be used to solve harder optimisation problems. For example, suppose you wanted to solve a least squares problem, but it was constrained: solutions only within a specific range are valid. For this, you can use `lsqlin`, which is part of Matlab.
- Or, perhaps you have a non-linear optimisation problem, where the variables you're trying to find are combined in non-linear ways. For this, you can use the `lsqnonlin` function in Matlab.
- These functions try to find a solution that minimises an error (that you can define).

CITY UNIVERSITY LONDON

Nonlinear optimisation example

- Find the equation of a curve $f(x) = a + bx + abx^2$ that best passes through the points $\mathbf{p}_1 = [1, 1]^T$, $\mathbf{p}_2 = [3, 4]^T$ and $\mathbf{p}_3 = [5, 5]^T$.
- Here, we can't use a linear technique, because the variables we'd like to determine, a and b, are combined in a non-linear way on the last term.



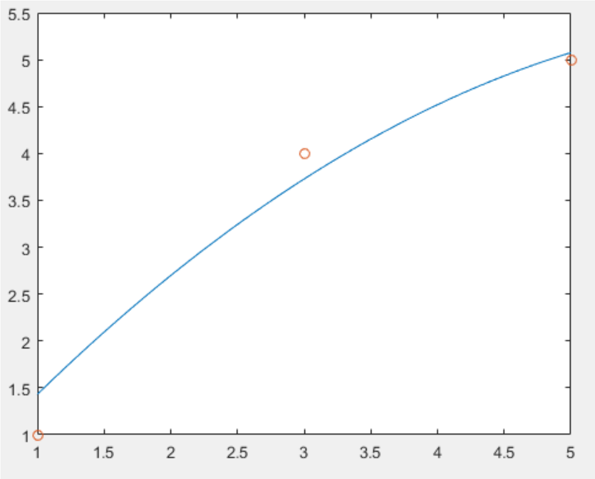
Nonlinear optimisation example

In Matlab

```
function error = CurveError(t, p)
a = t(1);
b = t(2);
L = size(p, 1);
for i = 1:L
    x = p(i, 1);
    y = p(i, 2);
    error(i) = a + b*x +a*b*x^2 - y;
end
```

In Matlab

```
p = [1, 1; 3, 4; 5, 5];
t0 = [0, 0];
topt = lsqnonlin(@(t)CurveError(t, p), t0);
a = topt(1);
b = topt(2);
x = [1:.1:5];
f = a+b*x+a*b*x.^2;
plot(x, f);
hold on;
scatter(p(:, 1), p(:, 2));
```



Finds the t that minimises the CurveError function, starting with an initial guess t0 = [a, b]

Integration

- Integration is the opposite of differentiation. For example, we know

$$\begin{aligned} f(x) &= x^2 \\ \frac{df}{dx} &= 2x \end{aligned}$$

- That is, the derivative of x^2 is $2x$. Going the other way, the integral of $2x$ is

$$\int (2x)dx = x^2 + C$$

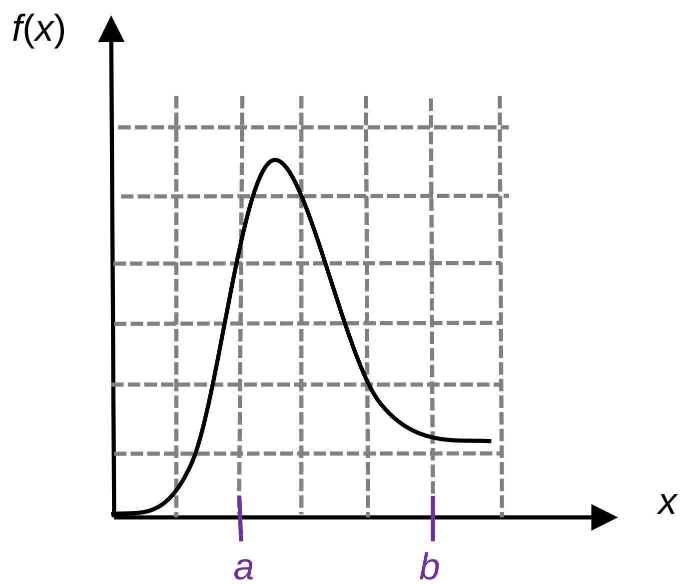
- This gives us x^2 again. A constant C has been added, since any function of the form $x^2 + C$ has a derivative $2x$, since the derivative of a constant is 0.

Indefinite vs definite integral

- An equation of the form $\int f(x)dx$ is an *indefinite* integral.

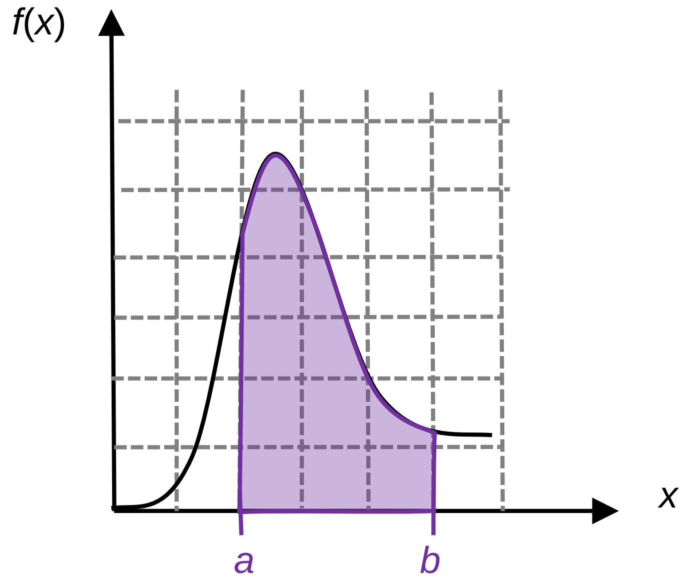
- An equation of the form $\int_a^b f(x)dx$ is a *definite* integral. The difference

is that in the case of a definite integral, we are only interested in evaluating the integral in a defined region of *x* ∊ [*a*, *b*].



Area under the curve

- For a signal $f(x)$, an integral measures the area under the curve. A definite integral allows the area to be limited to a specific region.

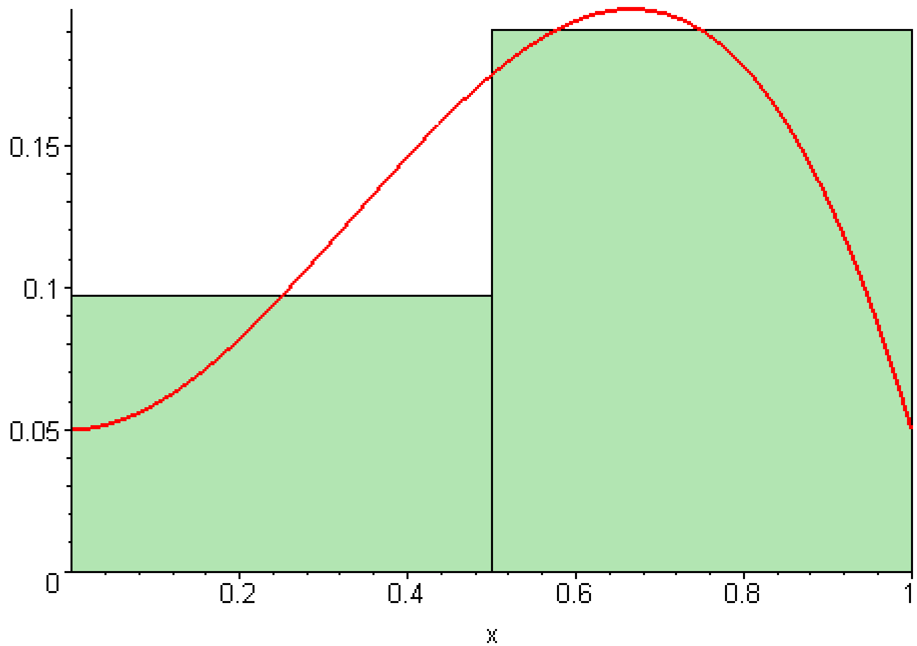


$$\int_a^b f(x)dx$$

The integral as a sum

- Often on a computer, the integral is approximated using a sum. One can sample the signal and use rectangles to approximate the area.
- The integral is approximated by summing the areas of the rectangles.
- As the size of the rectangles gets smaller, we get a better approximation.

⇨ *When you see an integral symbol, it may be helpful to think of it as a sum*



$$\int_a^b f(x)dx \approx \sum_{x=a}^{b} f(x)\delta x$$

δx is the rectangle width

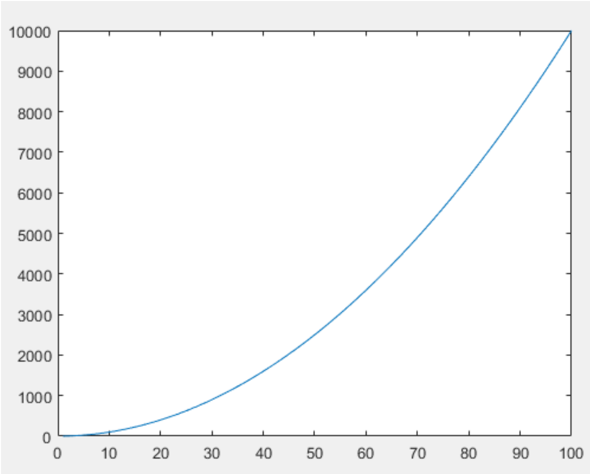
The integral as a sum

- Example: determine $\int_{20}^{80} x^2 dx$

In Matlab

```
x = [1:100];
f = x.^2;
plot(x, f);

% Between 20 and 80, sample f(x) every dx
% units and multiply by box width (dx)
a = 20;
b = 80;
dx = 5;
intfx = 0;
for xx = a:dx:b-dx
    intfx = intfx + f(xx) * dx;
end
```



- When
  - dx = 5, intfx = 153250
  - dx = 1, intfx = 165010
  - Actual area: 168000

Double integrals

- Integrals can be done over multiple variables. This is quite common in computer vision, as images are defined a two dimensional domain (*x* and *y*). An indefinite integral over a 2D domain may look like:

$$\int\int f(x,y)dxdy$$

- Sometimes definite integrals are defined using a symbol like *R* to mean a particular region (of an image, for example):

$$\iint_R f(x,y)dxdy$$