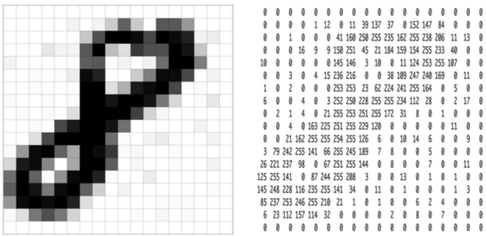
Foundational Concepts in Numerical Linear Algebra

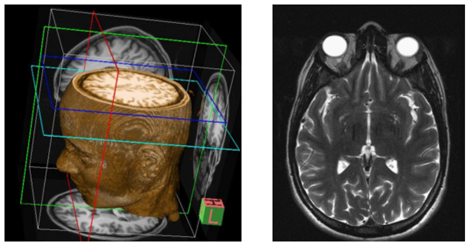
王偉仲 (台大應數所, 數學系與資料科學學程)
資料科學計算, Spring 2018

Matrix Representations

Image Data as a Matrix



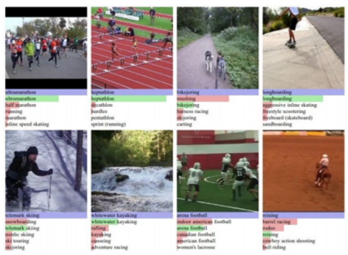
MRI Images as a 3D Matrix or Multiple 2D Matrices



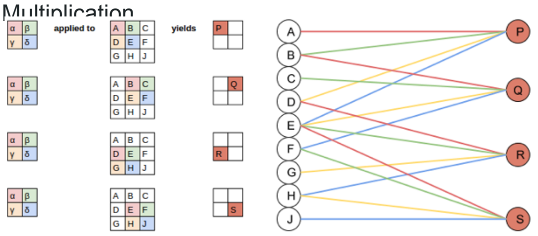
The Key Concepts of Linear Algebra

- From "Numbers"
  to "Vectors (can be matrics or functions too)"
  to "Spaces"
- The four subspaces

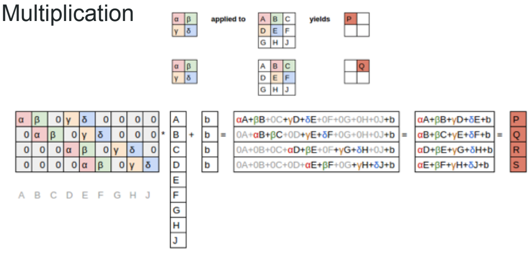
Convolution in CNN



Convolution as a Filter, Network, Matrix Multiplication

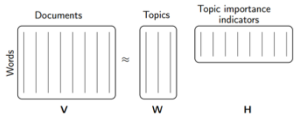


Convolution as a Filter, Network, Matrix Multiplication



Topic Modelling

	Anthony and Cleopatra	Julius Caesar	The Tempest	Hamlet	Othello	Macbeth ...
ANTHONY	157	73	0	0	0	1
BRUTUS	4	157	0	2	0	0
CAESAR	232	227	0	2	1	0
CALPURNIA	0	10	0	0	0	0
CLEOPATRA	57	0	0	0	0	0
MERCY	2	0	3	8	5	8
WORSER	2	0	1	1	1	5



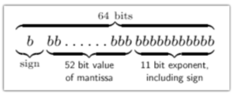
Floating Point Numbers

Floating Point Numbers

k	2^{-k}	b_k	$r_k = r_{k-1} - b_k 2^{-k}$
0	NA	NA	0.1
1	0.5	0	0.1
2	0.25	0	0.1
3	0.125	0	0.1
4	0.0625	1	0.1 - 0.0625 = 0.0375
5	0.03125	1	0.0375 - 0.03125 = 0.00625
6	0.015625	0	0.00625
7	0.0078125	0	0.00625
8	0.00390625	1	0.00625 - 0.00390625 = 0.00234375
9	0.001953125	1	0.0234375 - 0.001953125 = 0.000390625
10	0.0009765625	0	0.000390625
⋮	⋮		

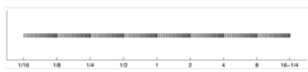
Therefore, the binary mantissa for 0.1 is $(00011\,0011\ldots)_2$.

Floating Point Numbers

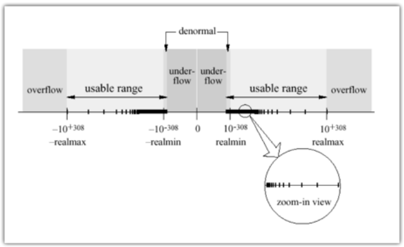


- (1.xxx)*2^e, e=-4, -3, ..., 3

- (1.xxxxx)*2^e, e=-4, -3, ..., 3



Floating Point Numbers



Floating Point Numbers

$$f(x) = \frac{1 - \cos x}{x^2}, \qquad x \neq 0$$

10 decimal digits, and it used rounded arithmetic.

x	Computed $f(x)$	True $f(x)$
0.1	0.4995834700	0.4995834722
0.01	0.4999960000	0.4999958333
0.001	0.5000000000	0.4999999583
0.0001	0.5000000000	0.4999999996
0.00001	0.0	0.5000000000

Floating Point Numbers

- 兩相近數相減，導致有效位數減損誤差 (loss of significance error)
- 數學上相等，計算效益不同

$$\begin{aligned} f(x) &= \frac{1 - \cos x}{x^2} = \frac{2\sin^2(x/2)}{x^2} \\ &= \frac{1}{2}\left[\frac{\sin(x/2)}{x/2}\right]^2 \end{aligned}$$

Ariane 5 Explosion

Ariane 5 Explosion

- 慣性參考系統軟體出錯
- 紀錄側向速度的64位元浮點數，在慣性參考系統軟體中，被轉換成16位元的整數
- 但卻超過其所能表示的最大數 (32,767)
- 慣性參考系統因此被認定故障，向主電腦送出錯誤信號，並自動關機
- 控制火箭主電腦，錯把錯誤訊息當成火箭 當時的火箭狀況參數，做出不必要的方向修正與旋轉
- 推進器與火箭因而被空氣動力裂解終於導致安全系統啟動，自動引爆

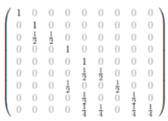
Quiz

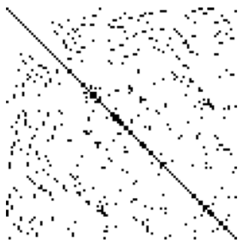
Estimate the size of a 3D MRI head image.

Memory Usage

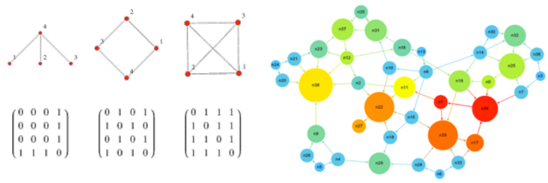
Dense and Sparse Matrix

Dense and Sparse Matrix



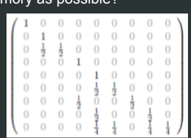


Graphic Matrix



Quiz

How do you store the following sparse matrix with as less of memory as possible?

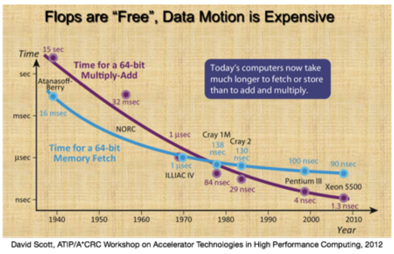


Performance

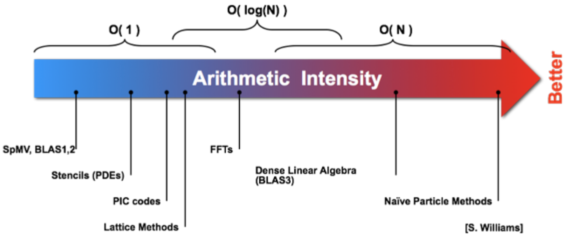
Complexity

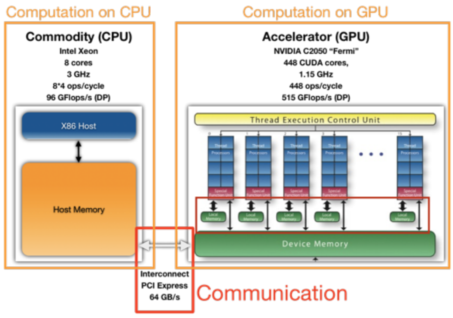
- Computational complexity
  - Big O notation
  - Usually in terms of number of rows (m) and number of columns (n)
  - Basic Linear Algebra Subroutines: BLAS 1, BLAS 2, BLAS 3
  - What about **sparse** matrix?
- Communication complexity
  - Getting more and more important

Computation and Communication



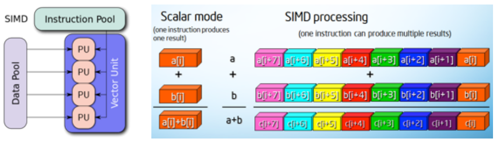
Arithmetic Intensity





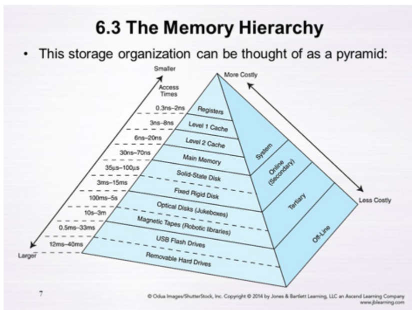
Vectorization

- SIMD: Single instruction, multiple data



- Will use vectorized operations in libraries (e.g. numpy, which in turn rely on specially tuned vectorized low level linear algebra APIs in particular, BLAS, and LAPACK)

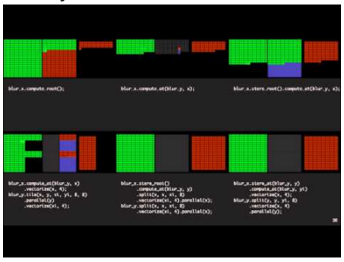
Locality



Locality

- To do computation required at that time, rather than having to load it multiple times each time we need it. Try to get higher arithmetics intensity (i.e. comp/access) 每拿一次資料，盡量多算一些.
- To access data items that are stored next to each other, so try to always use any data stored nearby that we know we'll need soon.
近期需要的資料，儘量放在附近記憶體
- Latency Numbers Every Programmer Should Know
  - https://people.eecs.berkeley.edu/~rcs/research/interactive_latency.html

Locality



- Locality is hard.
Potential trade-offs:
  - redundant computation to save memory bandwidth
  - sacrificing parallelism to get better reuse

Temporaries

- Temporary variables in RAM can significantly slow down the computation, comparing with in cache
- Numpy generally creates temporaries for every single operation or function it does.
  - E.g. a=b·c^2+ln(d) will create four temporaries (since there are four operations and functions)

Team Time

Complexity in Big O Notations
(Using not-boring math to measure code's efficiency)

https://goo.gl/yrnKkE