

Computing for Math and Stats

Lecture 5

Matrix Operations

- Most regular arithmetic operations work for matrices and vectors
- You can add two matrices (or vectors)
 - As long as they can be added
- You can multiply them, too
 - Matrix and vector multiplication rules apply
- Division is a bit trickier

Division

- Our notation for real number division (fractions) rely on commutativity of multiplication
- In Matlab we need
 - Matrix inverse
 - Left division
 - Right division
- We have notation for all

Inverse

- We can invoke the `inv()` function
- We can raise a matrix to -1 ($^{-1}$)
- If the matrix is invertible it is inverted
- In an ideal world a matrix is invertible if its determinant is non-zero
- Back to reality, it is trickier
 - Computer accuracy is finite (due to round-off error)
 - The properties of matrices can play tricks (condition number)
- See `invertedfun.m`,

Left Division

- Solve $AX = B$
 - X and B are (column) vectors
 - A is square (and invertible) matrix
- Solution is $X = A^{-1}B$
- If we want to avoid the full inversion we write
 - $X = A \backslash B$
- Matlab uses different numerical algorithms to calculate $A \backslash B$

Right Division

- Solve $XC = D$
 - X and D are row vectors
 - C is a square (and invertible) matrix
- Solution is $X = D C^{-1}$
- In Matlab we write
 - $X=D/C$

Element-wise Operations

- These appear sometimes in linear algebra or statistics
- Appear often in image and audio processing
- Sometimes are useful shortcuts
- They have a dot before the regular operator
- We have `.*`, `./`, `.^`

Built-in Functions

- Most functions that work on real numbers work on vectors/matrices as well
 - Sine, cosine, sqrt etc
- We also have
 - `mean(A)`, `median(A)`, `sum(A)`, `std(A)`
 - `sort(A)`
 - `m=max(A)`, `[d,n]=max(A)`, `min(A)`
 - `det(A)`, `inv(A)`
 - `dot(v1,v2)`, `cross(v1,v2)`

Random Matrices

- Random vectors and matrices are extremely useful
 - To try our algorithms to see if they “always” work
 - Some algorithms require a random number generator (Monte Carlo methods)
- We have the
 - `rand(m,n)`: uniformly distributed matrix. With one arg `n` produces `n x n` matrix and without it produces `1x1`.
 - `randi(imax, m, n)`: random integers `1...imax`
 - `randn(m,n)`: normally distributed