W1005
Intro to CS and Programming in MATLAB

Math

Fall 2014
Instructor: Ilia Vovsha

http://www.cs.columbia.edu/~vovsha/w1005

Outline

- Random numbers
- Linear algebra
  - Linear equations
  - Least squares
- Numeric errors

Random Numbers (purpose)

- Sampling and simulation (statistics)
- Cryptography (security)
- Algorithms
- Games

Random Numbers (generation)

- Two methods:
  - Measure physical phenomena
  - Computational algorithms
- Pseudo-Random Number Generator: algorithm which generates a sequence of predictable values if the initial *seed/key* is known
- Default generator: Mersenne Twister
- Issues:
  - Eventually the sequence repeats
  - The sequence converges to some number
  - Not (cryptographically) secure

Middle-Square Method

- Simple (but poor quality) PRNG:
  - Goal: generate sequence of n-digit numbers
  - Seed: n-digit number
  - Algorithm:
    1. Initial value = seed
    2. Square initial value (V) producing a 2*n-digit number (P)
    3. If result is less than 2*n digits, add leading zeros
    4. The middle n digits of P would be the next number in the sequence.
    5. Set V = P and go to step (2)
- Example: 0540 -> 2916 -> 5030 -> 3009

Random Numbers (commands)

- Relevant built-in functions:
  - rng(<seed>, '<generator>'): set generator
  - rand(m,n): uniformly distr. random numbers (0,1)
  - randi([imin,imax],m,n): uniformly distr. Integers from the interval
  - randn(m,n): normally distr. random numbers (mean = 0, var = 1)
  - randperm(n): random permutation of integers from 1 to n

Box-Muller Transform

- Method for generating standard, normally distr. random numbers given uniformly distr. numbers

$$f(x) = \frac{1}{\sigma\sqrt{2\pi}} \exp^{\frac{-(x-\mu)^2}{2\sigma^2}} \Rightarrow f(x) = \frac{1}{\sqrt{2\pi}} \exp^{-x^2/2}$$

- Given U_1,U_2, get Z_1,Z_2:

$$Z_1 = \sqrt{-2\ln U_1}\cos(2\pi U_2)$$

$$Z_2 = \sqrt{-2\ln U_1}\sin(2\pi U_2)$$

Linear Equations (notation)

- Common linear algebra problem: find solution for a set of linear equations
- Notation:

$$\begin{bmatrix} ax_1 + by_1 = c \\ ax_2 + by_2 = c \end{bmatrix} \equiv \begin{bmatrix} x_1 & y_1 \\ x_2 & y_2 \end{bmatrix} \begin{bmatrix} \mathrm{a/c} \\ \mathrm{b/c} \end{bmatrix} = \begin{bmatrix} 1 \\ 1 \end{bmatrix}$$

$$\equiv \begin{bmatrix} e_{1,1} & e_{1,2} \\ e_{2,1} & e_{2,2} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} 1 \\ 1 \end{bmatrix}$$

$$\equiv A\vec{x} = \vec{b}$$

Linear Equations (concepts)

- Fundamental questions:
  1. Does the solution exist?
  2. Is the solution unique?
  3. Approach for finding the solution?
- General considerations:
  1. Rank of matrix (linear independence)
  2. Number of equations vs. number of unknowns
  3. Gaussian elimination, LU factorization, computing inverse

$$A\vec{x} = \vec{b}$$

Linear Equations (solution)

- Rank of matrix: # of linearly independent columns
- Fewer eqtns than vars (under-determined) ➔ inifintely many solutions
- Fewer vars than eqtns (over-determined) ➔ no solution
- Does 'A' have an inverse?

$$A\vec{x} = \vec{b}$$

$$\vec{x} = A^{-1}\vec{b}$$

Linear Equations (commands)

- Rank of matrix: rank(A) == rank([A,b])
- If augmented matrix has same rank, then A has an inverse, and the solution is unique
- Find inverse:
  - inv(A) ➔ x = inv(A)*b;
  - Better approach (LU factorization): x = A\b;
- Over-determined case: *least squares* solution
- Under-determined case:
  - Maximally sparse solution ➔ x1 = A\b;
  - Smallest norm solution ➔ x2 = pinv(A)*b;
  - Compare norms: norm(x1)

$$A\vec{x} = \vec{b}$$

$$\vec{x} = A^{-1}\vec{b}$$

Least Squares (commands)

- Over-determined case: *least squares* solution
- Minimize norm of the squared *residual error*
- Least squares problems:
  - Linear least squares
  - Weighted least squares (weight for each example)
  - Nonlinear least squares (fit nonlinear model to data)
  - Commands: x = A\b, lscov(), lsqnonneg()

$$\vec{e} = A\vec{x} - \vec{b}$$

Numeric Errors

- Sources of numerical inaccuracies in computation
- *Representational error:*
  - Some numbers can't be represented exactly
  - Errors depend on storage bits
- *Cancellation error:*
  - Large numbers can cancel out smaller numbers (addition)
- *Arithmetic underflow/overflow error:*
  - Multiplication of two very small numbers is zero (underflow)
  - Multiplication of two very large numbers is infinity (overflow)

Numeric Computation (examples)

- Representational error:
  - a = 0.42 – 0.5 + 0.08
  - b = 0.08 – 0.5 + 0.42
  - a == b ?
  - sin(0) == sin(pi) ?