12.010 Computational Methods of Scientific Programming
Fall 2008

12.010 Computational Methods of Scientific Programming

Lecturers

Thomas A Herring
Chris Hill

Review of last Lecture

- Looked a class projects
- Graphics formats and standards:
  – Vector graphics
  – Pixel based graphics (image formats)
  – Combined formats
- Characteristics as scales are changed

Class Projects

- Class evaluation today
- Order of presentations

Stefan Gimmillaro	The Sodoku Master:
Amanda Levy	Solar Subtense Program
Adrian Melia	Model of an accelerating universe
Eric Quintero	Encryption/decryption algorithm
Karen Sun and Javier Ordonez	Truss Collapse Mechanism
Melissa Tanner and Sean Wahl	Phase diagram generator for binary and ternary solid-state solutions
Celeste Wallace	Adventure Game

Advanced computing

- A new development is fast computing is to use the computers Graphics Processing Unit (GPU) (not Central Processing Unit CPU).
- Drivers and software are available for the NVIDIA 8000-series of graphics cards (popular card).
- Company http://www.accelereyes.com/ makes software available for Matlab that uses these features.
- CUDA (*Compute Unified Device Architecture*) software downloaded from http://www.nvidia.com/object/cuda_get.html

Example performance gains

- For image smoothing: Convolution run on GPU:
  - Mean CPU time = 7059.88 ms
  - Mean GPU time = 828.907 ms
  - Speedup (CPU time / GPU time) = 8.51709
- Matrix mutiply example by size
- In-class demo of raindrop example

FFT example

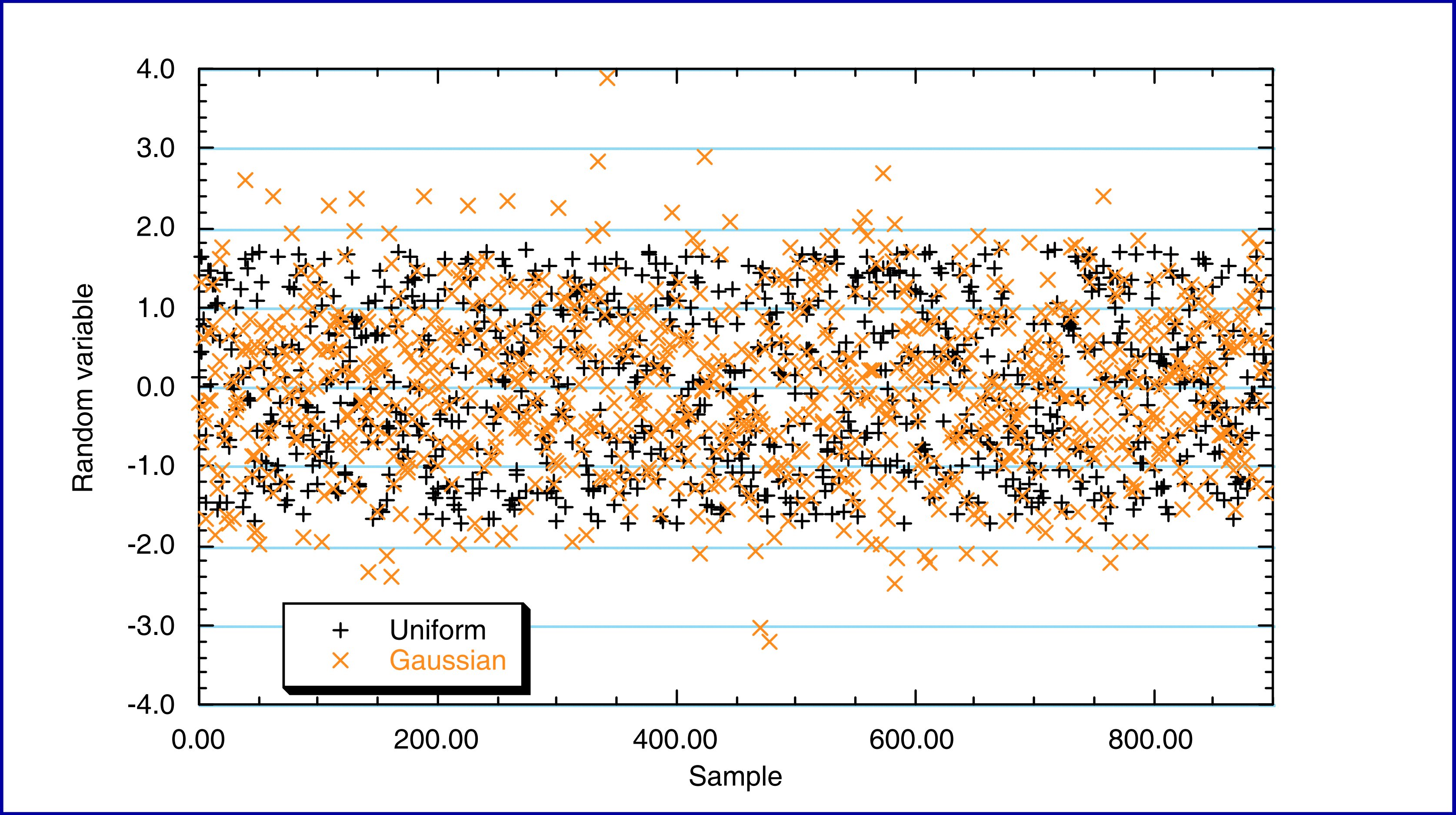
GPU processing

- Matlab interface is a convenient way to access the GPU processing power but the documentation is not quite complete yet and many crashes of Matlab (even when codes have run before).
- This type of processing will get more common in the future and the robustness should improve.
- NVIDIA GeForce chip set (plus other NVIDIA processors).
- Direct C programming is also possible with out the need for Matlab

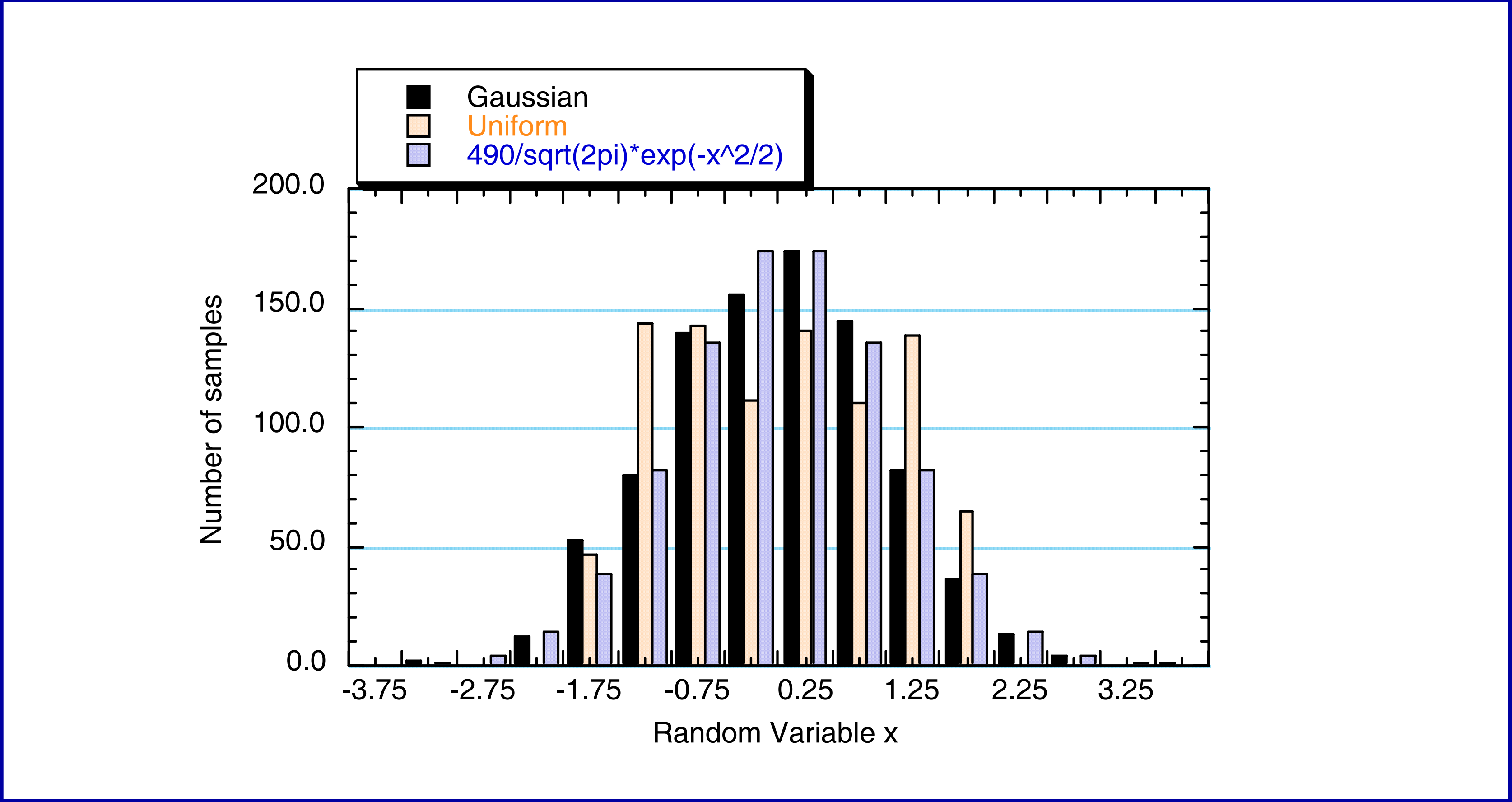
Review of statistics

- Random errors in measurements are expressed with probability density functions that give the probability of values falling between x and x+dx.
- Integrating the probability density function gives the probability of value falling within a finite interval
- Given a large enough sample of the random variable, the density function can be deduced from a histogram of residuals.

Example of random variables



Histograms of random variables



Characterization Random Variables

- When the probability distribution is known, the following statistical descriptions are used for random variable x with density function f(x):

Expected Value	$< h(x) >$	$\int h(x) f(x) dx$
Expectation	$< x >$	$\int x f(x) dx = \mu$
Variance	$< (x-\mu)^2 >$	$\int (x-\mu)^2 f(x) dx$

Square root of variance is called standard deviation

Theorems for expectations

- For linear operations, the following theorems are used:
  - For a constant <c> = c
  - Linear operator <cH(x)> = c<H(x)>
  - Summation <g+h> = <g>+<h>
- Covariance: The relationship between random variables $f_{xy}(x,y)$ is joint probability distribution:

$$\sigma_{xy} = <(x-\mu_x)(y-\mu_y)> = \int (x-\mu_x)(y-\mu_y) f_{xy}(x,y)\,dx\,dy$$

$$\text{Correlation}: \ \rho_{xy} = \sigma_{xy} / \sigma_x \sigma_y$$

Estimation on moments

- Expectation and variance are the first and second moments of a probability distribution

$$\hat{\mu}_x \approx \sum_{n=1}^{N} x_n / N \approx \frac{1}{T} \int x(t)dt$$

$$\hat{\sigma}_x^2 \approx \sum_{n=1}^{N} (x - \mu_x)^2 / N \approx \sum_{n=1}^{N} (x - \hat{\mu}_x)^2 / (N-1)$$

- As N goes to infinity these expressions approach their expectations. (Note the N-1 in form which uses mean)

Probability distributions

- While there are many probability distributions there are only a couple that are common used:
-

$$\text{Gaussian} \quad f(x) = \frac{1}{\sigma\sqrt{2\pi}} e^{-(x-\mu)^2/(2\sigma^2)}$$

$$\text{Multivariant} \quad f(\mathbf{x}) = \frac{1}{\sqrt{(2\pi)^n |V|}} e^{-\frac{1}{2}(\mathbf{x}-\mu)^T \mathbf{V}^{-1}(\mathbf{x}-\mu)}$$

$$\text{Chi-squared} \quad \chi_r^2(x) = \frac{x^{r/2-1} e^{-x/2}}{\Gamma(r/2) 2^{r/2}}$$

Probability distributions

- The chi-squared distribution is the sum of the squares of r Gaussian random variables with expectation 0 and variance 1.
- With the probability density function known, the probability of events occurring can be determined. For Gaussian distribution in 1-D; $P(|x|<1\sigma) = 0.68$; $P(|x|<2\sigma) = 0.955$; $P(|x|<3\sigma) = 0.9974$.
- Conceptually, people thing of standard deviations in terms of probability of events occurring (ie. 68% of values should be within 1-sigma).

Central Limit Theorem

- Why is Gaussian distribution so common?
- "*The distribution of the sum of a large number of independent, identically distributed random variables is approximately Gaussian*"
- When the random errors in measurements are made up of many small contributing random errors, their sum will be Gaussian.
- Any linear operation on Gaussian distribution will generate another Gaussian. Not the case for other distributions which are derived by convolving the two density functions.

Random Number Generators

- Linear Congruential Generators (LCG)
  - $x(n) = a * x(n-1) + b \bmod M$
- Probably the most common type but can have problems with rapid repeating and missing values in sequences
- The choice of a b and M set the characteristics of the generator. Many values of a b and M can lead to not-so-random numbers.
- One test is to see how many dimensions of k-th dimensional space is filled. (Values often end up lying on planes in the space.
- Famous case from IBM filled only 11-planes in a k-th dimensional space.
- High-order bits in these random numbers can be more random than the low order bits.

Example coefficents

- Poor IBM case: $a = 65539$, $b = 0$ and $m = 2^{31}$.
- MATLAB values: $a = 16807$ and $m = 2^{31} - 1 = 2147483647$.
- Knuth's Seminumerical Algorithms, 3rd Ed., pages 106--108: $a = 1812433253$ and $m = 2^{32}$
- Second order algorithms: From Knuth:
  $x_n = (a_1 x_{n-1} + a_2 x_{n-2}) \bmod m$
  $a_1 = 271828183$, $a_2 = 314159269$, and
  $m = 2^{31} - 1$.

Gaussian random numbers

- The most common method (Press et al.)
  Generated in pairs from two uniform random number x and y
  z1 = sqrt(-2ln(x)) cos(2*pi*y)
  z2 = sqrt(-2ln(x)) sin(2*pi*y)
- Other distributions can be generated directly (eg, gamma distribution), or they can be generated from the Gaussian values (chi^2 for example by squaring and summing Gaussian values)
- Adding 12-uniformly distributed values also generates close to a Gaussian (Central Limit Theorem)

Conclusion

- Examined random number generators:
- Tests should be carried out to test quality of generator or implement your (hopefully previously tested) generator
- Look for correlations in estimates and correct statistical properties (i.e., is uniform truly uniform)
- Test some algorithms with matlab: randtest.m