Lecture #4

Psych 254

Russ Poldrack, instructor

Outline

- System-level interaction
- using eval()
- Random numbers

Interacting with the system

- MATLAB has functions that allow interaction with operating system
  - Very useful on unix systems
  - Relatively useless on windows systems

System interaction methods

```
>> ls                                   % built-in ls function

ans =
expinvcdf.m mynormpdf.m quincunx.m

>> !ls exp*.m                           % exclamation pt
expinvcdf.m                             % runs subsequent command

>> [status, output]=system('ls exp*.m') % system() runs
                                        % a string in the
status =                                % OS, returns
     0                                  % both the status
                                        % and the output
output =
expinvcdf.m
```

Using MATLAB to create scripts

- MATLAB can be used to create arbitrary text files
  - shell scripts in UNIX
  - other MATLAB scripts
- This can be useful for creating complex programs where there is less chance of error than creating the file by hand
  - requires mastery of the subtleties of fprintf

Automating MATLAB functions

- MATLAB programs can be run from a shell script in UNIX
  - `echo “x=[0:0.1:pi*8];plot(x,sin(x))”|matlab`
  - `matlab < viz.m`
    - where viz.m is a matlab script

the eval() function

- eval(s) will evaluate the string s as a MATLAB command
- This can be useful in cases where a command needs to be created on the fly

using eval()

```
>> cmd=sprintf('a=1; j=2; fprintf(''a=%%d, j=%%d\\n'', a,j);');
>> cmd

cmd =
a=1; j=2; fprintf('a=%d, j=%d\n', a,j);

>> eval(cmd)
a=1, j=2


>> stat='median';
>> j=rand(1,100);
>> cmd=sprintf('central_tendency=%s(j)',stat)
cmd =
central_tendency=median(j)

>> eval(cmd)
central_tendency =
    0.4666
```

Random numbers

- Because computers are deterministic systems, they cannot produce truly random numbers
  - e.g., the time between ticks of a Geiger counter due to radioactive decay
- Most computers have “random number generators” that create “pseudo-random” numbers
- RNG’s generally start with a “seed” that allows a particular sequence of random numbers to be repeated
  - the seed is generally set to some function of the current time, in order to provide new random numbers each time

Why do we need random numbers?

- Random numbers are used extensively in scientific programming
  - Data analysis
    - methods using random sampling (e.g., bootstrap)
  - Simulation
    - need to simulate a process that involves noise
  - Modeling
    - need to find a random starting point for the model
    - need to introduce variability at different points in the model
  - Program testing/validation
    - need to create lots of variable data to test a program

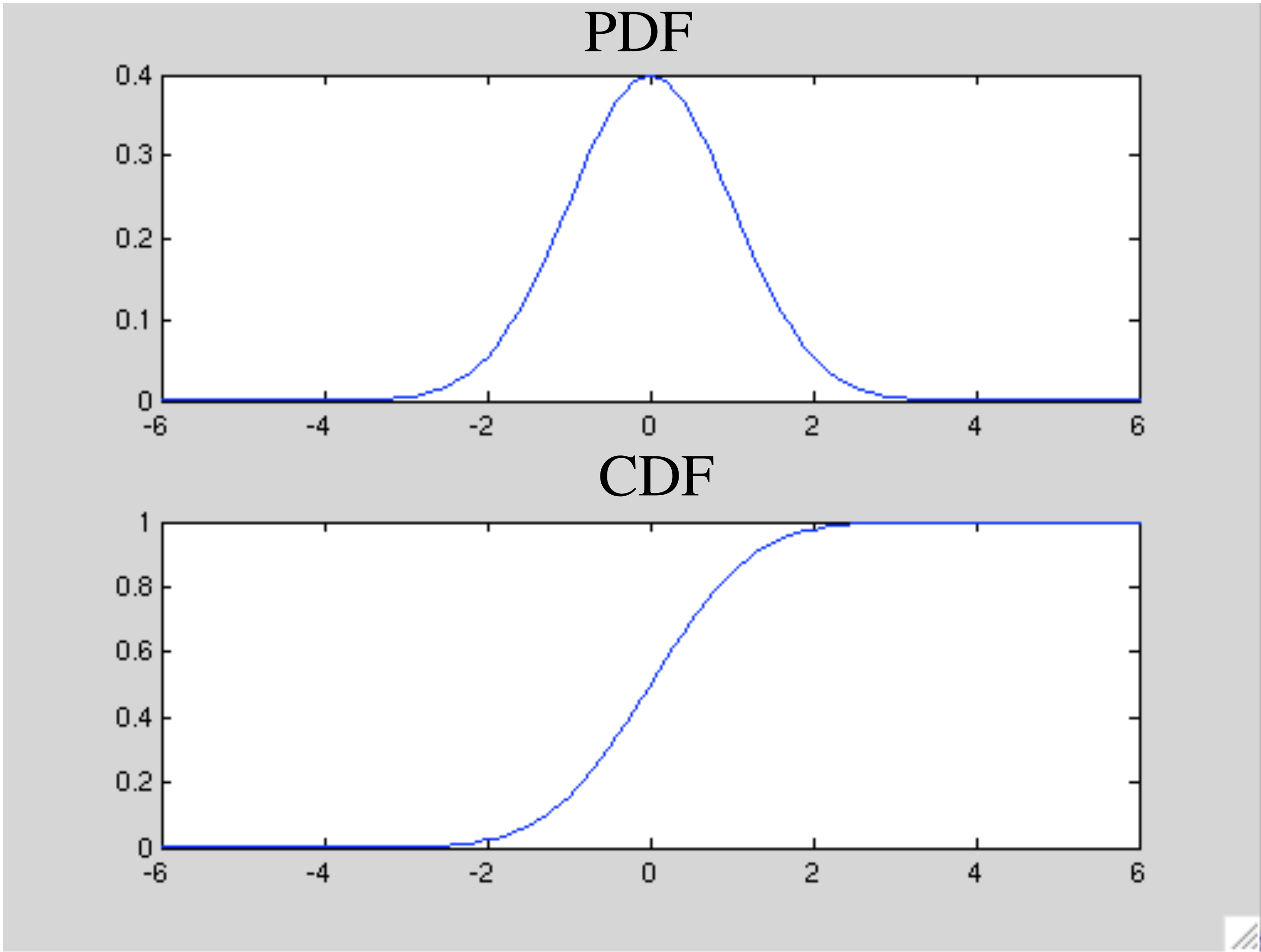
Types of random numbers

- The basic form of random numbers is the uniform random variate
  - range: [0,1]
- A uniform random variate has equal probability of occurring anywhere in its range

Probability distributions

- A probability distribution has two characteristic functions that describe it:
  - probability density function (PDF)
    - For discrete distributions, describes the probability of a particular value occuring
    - For continuous distributions, describes the probability of values falling in a particular interval
    - The range of a PDF differs between distributions
  - cumulative distribution function (CDF)
    - the probability of a value falling below some specified value

Normal PDF vs CDF

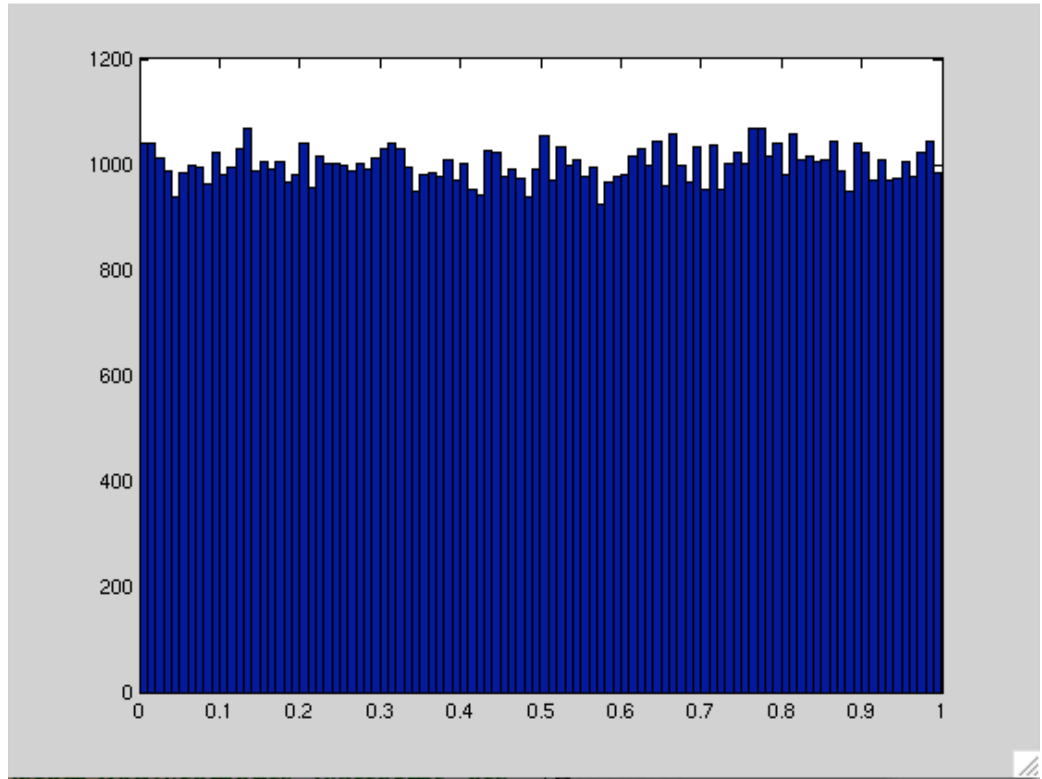


Uniform random variates

Uniform variates are sampled from a distribution where the PDF is flat across its range

Sampling from this distribution is equally likely to pull values from anywhere in the range

this is what rand() returns



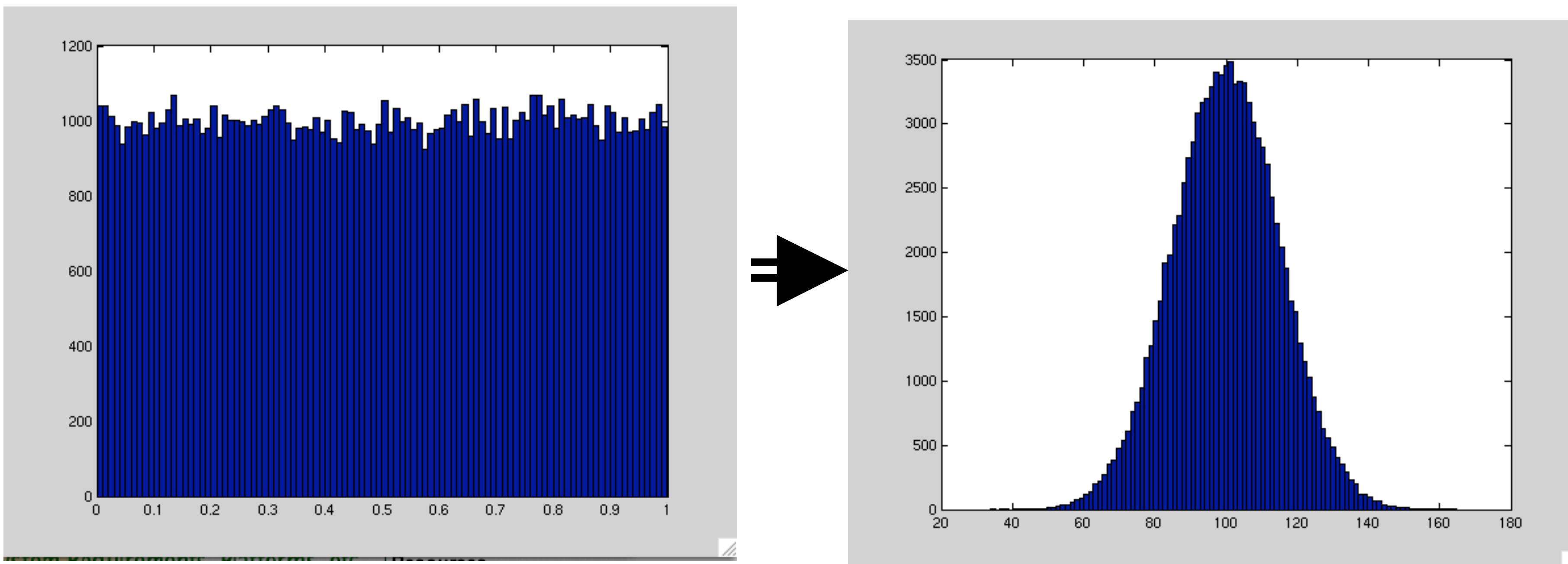

```
r=rand(1,100000);
h=hist(r,100);
```

Non-uniform random variates

- Using uniform random variates, we can create random variates from any other distribution if we have its inverse CDF
  - the CDF maps the range of the distribution to cumulative probability
  - the inverse CDF maps backwards from probabilities to values on the distribution
    - the domain of the inverse CDF is [0,1]
- There are other ways to create random variates if the inverse CDF is not known
  - see “Numerical Recipes”

Normal random variates

```
r=rand(1,100000);
nr=spm_invNcdf(r,100,15);
hist(nr);
```



Other distributions

- The standard MATLAB distribution does not provide anything beyond uniform random variates
- The SPM package has functions to provide inverse CDF’s for many different functions:
  - beta, F, gamma, normal, Poisson, t, chi-squared
- The MATLAB statistics toolbox has many more

Monte Carlo simulation

- A method of creating synthetic simulated data sets by sampling from a particular theoretical distribution
- Allows the solution of problems that do not have an analytic solution
  - e.g., estimation of error for complex distributions
- “Offered the choice between mastery of a five-foot shelf of analytical statistics books and middling ability at performing statistical Monte Carlo simulations, we would surely choose the latter skill” - Press et al., “Numerical Methods in C”

Quick and dirty Monte Carlo: The Bootstrap

- The bootstrap is a way of finding confidence intervals for a dataset when the underlying distribution is unknown
- Involves repeatedly resampling the data with replacement and computing the statistics of interest
- Then, use the empirical distribution of the sample statistics to characterize its sampling distribution

The Bootstrap