Simulation & Logical Arrays

Lecture 13 (Mar 7)
CS100M – Spring 2006

Announcements

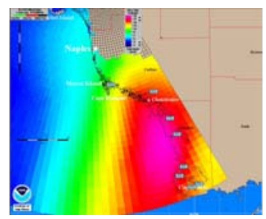
- Project 3
  - Due: Thursday, March 9
  - Demo
- Prelim II
  - 7:30pm
  - Thursday, March 16
  - Includes material through this week

Topics

- Reading: CFile 9, Section 9.3
- Recall
  - Matlab vectors (1D arrays) & matrices (2D arrays)
  - Characters & Strings
  - Vectorized code
- Plans for today
  - Simulation
  - More on Logical arrays

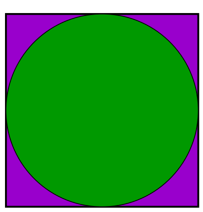
Simulation

- The application of mathematical and computer models to imitate the behavior of a system
  - Usually a real-world system (but not always)
  - Useful for design, training, & games
- Matlab provides many tools useful for simulation
  - We'll examine some very simple simulations



Example: Simulation of Darts

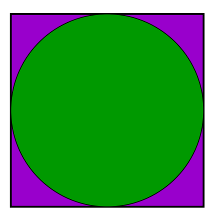
- Goal: Simulate darts thrown at a simple target to derive an estimate of π
- We did this example earlier using iteration
- Assume hits are distributed uniformly over this 2-by-2 square
  - $N_{in}/N = A_{circle}/A_{square} = \pi/4$



Original Code (for Just One Throw)

```
close all
hold on
axis('equal');
axis([-1 1 -1 1]);

px = 2*rand - 1;
py = 2*rand - 1;
if (px^2 + py^2 <= 1)
    plot(px, py, 'og');
else
    plot(px, py, 'or');
end
```



Throwing Darts using Vectorized Code

- How can we compute all throws at once by using a nDarts-by-2 matrix?
- How can we determine each throw's distance from origin?
- How can we count how many of the throws are within the circle?

```
function estimate = approxPi(nDarts)

throws = -1 + 2*rand(nDarts, 2);
x = throws(:, 1);
y = throws(:, 2);

dist = sqrt(x.^2 + y.^2);
in = sum(dist <= 1);
estimate = 4 * in/nDarts;
```

Example: Rolling a Fair Die

- Goal: Simulate the rolling of a fair die and create a histogram of the outcome
- How can we compute all the die rolls at once?
- How can we count how many of each roll occurred?

```
function count = rollDie (nRolls)

count= zeros(1,6);
rolls = ceil(6 * rand(1, nRolls));

for k= 1:6
    count(k) = sum(rolls == k);
end
```

More about Logical Arrays

- Logical arrays
  - Occur when you use vectorized relational operators
  - Consist of 0's (for false) and 1's (for true)
- In examples up to now, we've mostly used function sum( ) to count the number of true items in a logical array
  - Example: Count the number of s's in a sentence:
    sum('s' == 'This is a sentence.')
- The Workspace viewer (in the Desktop menu) shows the "class" of each of your variables

Logical Arrays Can Be Subscripts!

- When used in this way, the logical array "picks out" just some of the items
  - Example: v = [7 0 5 2 4 6 3 8 1]

```
logical = v > 4;            % [ 1 0 1 0 0 1 0 1 0 ]
selection = v(logical);     % [ 7 5 6 8 ]
selection = v(~logical);    % [ 0 2 4 3 1 ]
```

- This works on 2D arrays (matrices), too
  - But the matrix and the logical array must have same shape
  - The result is always a column vector
  - Example: v = [7 0 5; 2 4 6; 3 8 1]

```
logical = v > 4;            % [ 1 0 1; 0 0 1; 0 1 0 ]
selection = v(logical);     % [ 7; 5; 6; 8 ]
```

You Can Use Logical Subscripts to Assign to Part of an Array

- Example: To "zero out" all the negative numbers in a matrix

```
m = 20*rand(5,5) – 10;      % Random #s between –10 and 10
logical = m < 0;            % 5-by-5 logical array
m(logical) = 0;             % Sets all negative #s to 0
```

- Example: To replace all occurrences of a letter in a string

```
s = 'assign to part of an array';
s(s == 'a') = 'x';          % 'xssign to pxrt of xn xrrxy'
```

Can Find Indices using Find( )

- Example: v = [7 0 5 2 4 6 3 8 1]

```
logical = v < 3;            % [ 0 1 0 1 0 0 0 0 1 ]
indices = find(v < 3);      % [ 2 4 9 ]
```

- Example: v = [7 0 5; 2 4 6; 3 8 1]

```
logical = v < 3;            % [ 0 1 0; 1 0 0; 0 0 1]
[r c] = find(v < 3);        % r = [ 2; 1; 3 ]
                            % c = [ 1; 2; 3 ]
                            % I.e., (2,1), (1,2), (3,3)
```