Max Kramer

I affirm that I have adhered to the honor code on this assignment.

Hello again, scientist! I'll write in italics, and problems for you will always be in **bold***. As a general rule, I expect you to do at least as much writing as I do. Code should be part of your solution, but I expect variables to be clear and explanation to involve complete sentences. Cite your sources; if you work with someone in the class on a problem, that's an extremely important source.*

Problem 5.3.

First, ***let Q be a 3x3 matrix whose entries are randomly drawn from the one-digit numbers 0-9. Show me Q and det(Q).*** *Note: your output should be different every time you publish your m file.*

```
Q = randi(10,3)-1
detQ = det(Q)
```

```
Q =

     6     1     6
     6     0     3
     3     9     9


detQ =

  117.0000
```

The randi(r,n) command creates a square matrix of size n made up of random integers between 1 and r. By using 10 as the seed and then subtracting 1, we get a matrix of random integers betwen 0 and 9. The det() command is then used to calculate the determinant.

Uncomment the following code and explain what property of determinants is being demonstrated with each example.

```
det(2*Q)

det(Q^2)

det(Q^(-1))

det(Q.')

R = [2 0 0; 0 1 0; 0 0 3]; det(R*Q)

S = Q; S(1,:) = S(1,:)+S(2,:); det(S)
```

```
ans =
```

```
  936.0000


ans =

   1.3689e+04


ans =

    0.0085


ans =

   117


ans =

   702


ans =

   117
```

The first line demonstrates that if one row of A is multiplied by k to produce B, then det(B) = k *det(A). The second line demonstrates that for two matrices A and B, det(AB) = det(A) * det(B). The third line demonstrates that det(A) = (-1)^r * product of pivots in U for a square matrix A, its reduced echelon form U, and the number of interchanges r it takes to go from A to U. The fourth line demonstrates that for a square matrix A, det(A) is equal to det(A'). The fifth line demonstrates the multiplicative property of determinants (det(A)*det(B)=det(AB)). The last line demonstrates that If a multiple of one row of A is added to another row to produce a matrix B, then det(B) = det(A).

There's actually a small chance that one of the above will throw an error when you uncomment it. ***Which one, and why?***

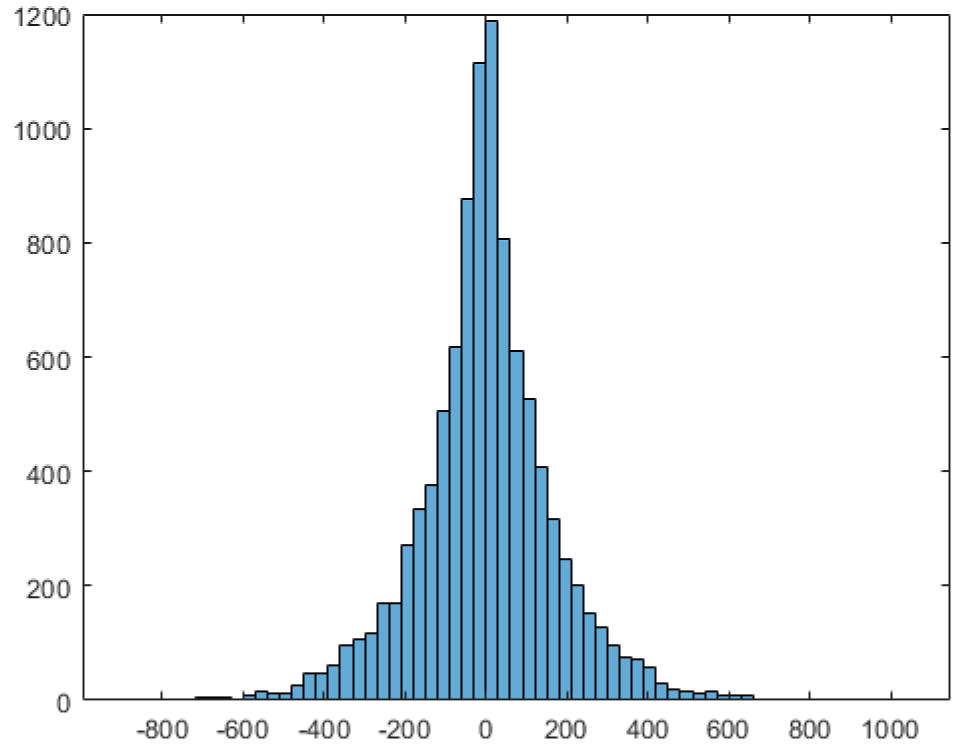
det(Q^(-1)) may throw an error as it is entirely possible that the randomly generated matrix Q will be a singular matrix and thus not be invertible.

Well yeah okay, but what are the chances of that? ***Put abunchofdets.m in the same path as this m file and use it to generate the determinants of 10^4 random 3x3 matrices.*** *Use a dang semicolon.*

```
B = abunchofdets(3,10^4);
% help abunchofdets % this'll show you how to use abunchofdets()
```

Show me a histogram of the output.

```
histogram(B)
```



How many of these 10^4 matrices are not invertible? What's that as a percentage of the total?

```
zerodet = length(B)-nnz(B)
zerodetpercent = zerodet / length(B)
```

```
zerodet =

   187


zerodetpercent =

    0.0187
```

The nnz command calculates the number of nonzero entries in a vector. By subtracting this number from the total determinants calculated, we are left with the number of 0 entries which is the number of non-invertible matrices.

Of the matrices that you found, what's the largest determinant?

```
biggestdet = max(B)
```

```
biggestdet =
```

```
1026
```

The max() command returns the largest entry in a vector.

Bonus! CS majors take note, but this isn't part of the problem. rng(seed), where seed is a number between 1 and 2^32, makes the random number generator return a repeatable output. Uncomment the following code, then change n around until you find a value that makes L as large as you can get it. Because there's absolutely no way to shortcut this other than brute force, this is a good example of what's called a 'proof of work' task. You're basically mining cryptocurrency, except that you can't sell it. But I'll bring candy or balloons or something if anyone finds an n that gives L = 1458.

```
%bigL = [];
%tic;
%for n = 1:1000000
%    rng(n);
%    L = abunchofdets(3,1);
%    bigL(1,n) = L;
%end
%toc;
%max(bigL)
```

By checking every seed between 1 and 1,000,000 I found a maximum determinant of 1080. This computation took 207.67 seconds.

Published with MATLAB® R2019b