Backward Substitution 1

October 9, 2012

The backward substitution algorithm necessary to solve a upper-triangular linear system can be represented as follows:

$$\begin{aligned} x_n &= \frac{b_n}{a_{nn}} \\ x_i &= \frac{d_i - \sum_{j=i+1}^{n} a_{ij} x_j}{a_{ii}} \qquad for \quad i = n-1, n-2, \ldots, 1 \end{aligned}$$

Write a function that takes as input a square upper-triangular matrix A and vector b. It uses these inputs to solve the upper triangular linear system $Ax = b$ by implementing the backward substitution algorithm above, and returns the vector x as output. The function should first check of the matrix A is square and upper-triangular and if not, set x to infinity. You might try implementing the summation above as a vectorized operation rather than a `for` loop.

Round the estimate to 10 decimal digits of precision.

Example:

```
>> A
A =
     2     3     7
     0     5     8
     0     0     9
>> b=A*ones(3,1)
b =
    12
    13
     9
>> x=my_backsub(A,b)
x =
     1
     1
     1
```