CR2

May 10, 2023

0.0.1 Problem Description

In this report we will examine the LU decomposition and Gaussian elimination without pivoting methods for solving a system Ax - b. Specifically we will examine the time to compute and the residual error and matrices of different sizes using each method and record these values. We will create a plot of the time to compute and and table of the residual errors.

0.0.2 Results

See 'Wallclock Time and Errors' section below.

0.0.3 Guasian Elimination Without Pivoting

```
rref(A) =  [[9, 11, 3], [0, -2.5555555555555554, 2.6666666666666665], [0, 0,
5.956521739130435]]
```

0.0.4 Back Substitution

```
x =  [1.0197820797630384, -0.6074262139003492, 0.16788321167883213]
Residual Error:  4.440892098500626e-16
Time to Compute:  0.0
```

0.0.5 LU Factorization

```
L =  [[1.0, 0.0, 0.0], [3.5, 1.0, 0.0], [4.5, 0.875, 1.0]]
U =  [[2, 4, 5], [0.0, -8.0, -12.5], [0.0, 0.0, -8.5625]]
```

0.0.6 Solve LUx = b via Forward and Backwards Substitution

```
x =  [-0.25547445255474455, 0.13868613138686126, 0.5912408759124088]
Residual Error:  4.965068306494546e-16
Time to Compute:  0.0
```

0.0.7 Define a $(n \times n)$ matrix $A = (5\sqrt{n})I + R$

0.0.8 Solve Using n = 50

```
Time to Compute Using n =  50 with LU:  0.016570568084716797
Residual error:  0.6312783040038418
Time to Compute Using n =  50 with GE:  0.010084390640258789
Residual error:  6.42099832184827
```

0.0.9 Solving Using Size n = 100

```
Time to Compute Using n =  100 with LU:  0.041735172271728516
Residual error:  1.5630335525243462
Time to Compute Using n =  100 with GE:  0.011214017868041992
Residual error:  10.653541733998619
```

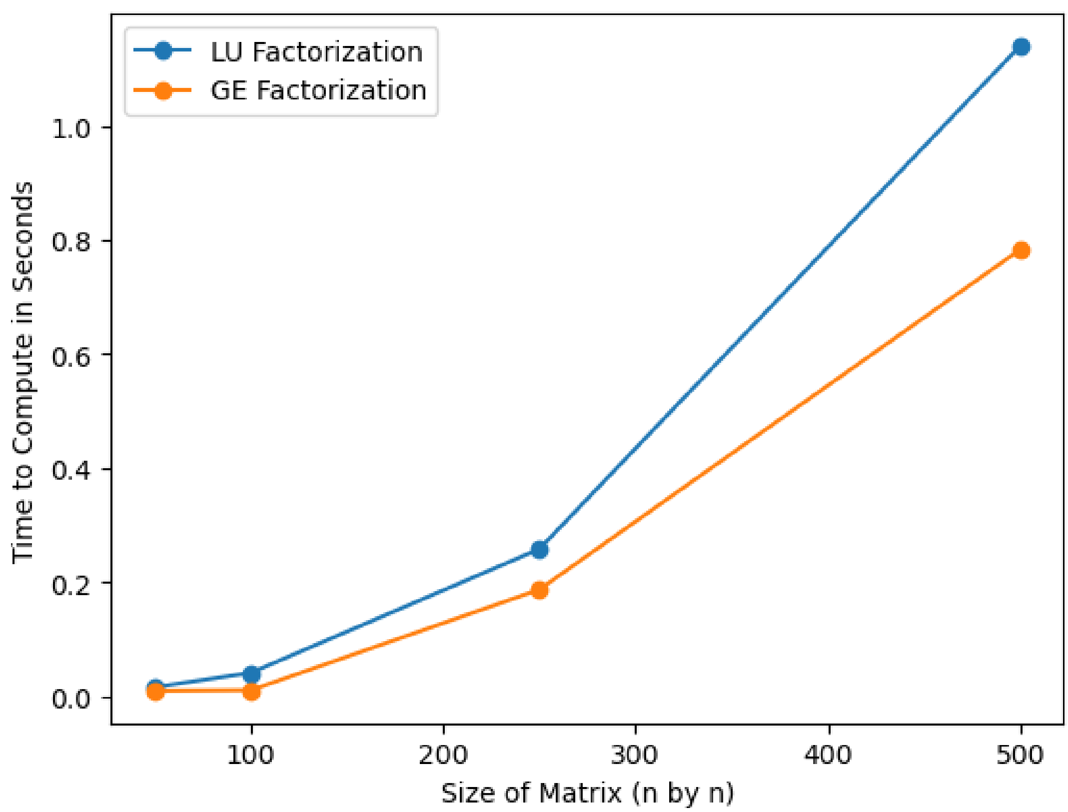
0.0.10 Solving Using n = 250

```
Time to Compute Using n =  250 with LU:  0.2585735321044922
Residual error:  2.4037620129700445
Time to Compute Using n =  250 with GE:  0.1876082420349121
Residual error:  16.314090731320803
```

0.0.11 Solving Using n = 500

```
Time to Compute Using n =  500 with LU:  1.1399741172790527
Residual error:  3.313586902166185
Time to Compute Using n =  500 with GE:  0.7832252979278564
Residual error:  22.559363605237124
```

0.0.12 Plot of Time to Compute by Method, by n



0.0.13 Table of Errors by Method, by n

n	GE	LU
50	6.42099832184827	0.6312783040038418
100	10.653541733998619	1.5630335525243462
250	16.314090731320803	2.4037620129700445
500	22.559363605237124	3.313586902166185

0.0.14 Errors and Wallclock Time

The errors were consistently smaller using the LU decomposition instead of Gaussian elimination. The time to compute was similar for smaller sized matrices (up to size 100), after n grew large enough LU decomposition began to take significantly more time and was outpacing Gauss for how quickly the time to compute increased. The wallclock time differnce between the mehtod was somewhat expected, due to the nature of LU decomposition requiring the composition of two matrices. More generally:

For Gauss, once the matrix is in this form, back substitution is performed to obtain the solution. This process has a computational complexity of O(n^3), where n is the size of the matrix. For LU, Once A is decomposed, solving systems of equations with different right-hand sides becomes computationally efficient, typically requiring O(n^2) operations for each system.

Therefore, if you have to solve the system Ax - b = 0 multiple times with different matrices A but the same LU decomposition, the LU decomposition approach can potentially be faster than performing Gaussian elimination for each system independently. By reusing the LU decomposition, you can save computational time by avoiding the need to perform the factorization repeatedly.

However, if the matrix A changes with every system to be solved, (as in this case) then re-factoring the matrix using LU decomposition for each system may offset the benefits.

0.0.15 Family A^ = R

```
Time to Compute Using n =  50 with LU:  0.01657247543334961
Residual error:  67.31365959855555
Time to Compute Using n =  50 with GE:  0.0036454200744628906
Residual error:  nan
```

```
C:\Users\gg\AppData\Local\Temp\ipykernel_6292\1616214449.py:11: RuntimeWarning:
divide by zero encountered in divide
  x[i] /= U[i][i]
C:\Users\gg\AppData\Local\Temp\ipykernel_6292\1616214449.py:10: RuntimeWarning:
invalid value encountered in multiply
  x[i] -= U[i][j] * x[j]
```

What I notice is that the residual error for this family A^ = R is much larger than the residual error for the same sized matrices of the family A^ = (5√n)I + R. The time to compute was similar for both cases.