Table of Contents

[INTRODUCTION 2](#_Toc446412309)

[Gauss Function 3](#_Toc446412310)

[BGauss Function 4](#_Toc446412311)

[Poisson’s Equation in 1D 5](#_Toc446412312)

[Poisson’s Equation in 2D 6](#_Toc446412313)

[Mastery Section: Poisson’s Equation in 3D 7](#_Toc446412314)

[Appendix 8](#_Toc446412315)

[Tables 8](#_Toc446412316)

[Graphs 8](#_Toc446412317)

[Representative Code 8](#_Toc446412318)

All work done in this project is my own unless stated otherwise

INTRODUCTION

This project deals with solving linear systems of equations, .

In the first section I implement a function Guass() that uses Gaussian Elimination to turn an inputted matrix, **,** into a upper diagonal matrix through which I can then use backwards substitution to get the answer vector, .

In the second section I implement a BGauss() function which solves a banded linear system of equations. That is a system where the matrix, , has non-zero terms only on the diagonal and a certain number of diagonal bands removed from the central diagonal.

In section 3, I solve a certain case of Poisson’s equation for the 1D case using my Gauss() and BGauss() functions and compare the two methods.

In section 4, I solve a similar form of Poisson’s 1D equation from section 3 but for the 2D case instead and again compare the Gauss() function to the BGauss() function.

Finally in the Mastery Section I solve Poisson’s Equation for the 3D case.

Gauss Function

The Gauss() function, with prototype

**double** **\***Gauss**(double** **\*\***A**,** **double** **\***y**,** **int** N**)**

solves a linear system of equations of the form , where is an matrix and is a vector of size . It does so using the Gaussian Elimination method. Here I treat and as one matrix of size and make it upper diagonal by performing row operations. Note for the purposes of this project, no pivoting has been implemented and so row operations are limited to arithmetic operations of addition, subtraction, multiplication and division of rows.

## Number of Additions/Subtractions, Multiplication and Divisions

Total number of operations for the forward elimination will be

and for the backward substitution will be

so I get a total number of floating point operations of

BGauss Function

The BGauss() function, with prototype

**double** **\***BGauss**(double** **\*\***A**,** **double** **\***y**,** **int** N**,** **int** B**)**

also solves a linear system of equations, form , where is an matrix and is a vector of size . Here only has non-zero values on a diagonal band of width .

## Number of Additions/Subtractions, Multiplication and Divisions

The total number of iterations required for BGuass are:

The number of values requires for forward elimination, backward substitution and the extra values that are not looped over as j reaches the end bounds of the matrix.

This gives a total in terms of N and B of

Poisson’s Equation in 1D

In this Section we consider a 1D Poisson Equation of the form

We split up our 1D grid/line into N+1 points from 0..1 in increments of . So for . . This gives us the approximation for as

So this gives us a set of N-1 equations that can be solved using our Gauss() and BGauss() functions.

# Gauss()

For the Gauss function I use the A matrix that looks like

where the size of A is .

# BGauss()

In BGauss I use a matrix that has -2 down the middle and band width of 1. It then looks like

Its size is which is considerably smaller than the matrix created for the Gauss function for large N.

# The Table that was asked for

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| N | max rough | pos rough | max rough | pos rough | Wall Time Gauss | Speed | Wall Time BGauss | Speed |
| 8 | 5.781250 | 0.37500 | 4.062500 | 0.37500 | 0.005598 | 0.00002054 | 0.000001 | 0.0540000 |
| 16 | 4.960937 | 0.43750 | 4.062500 | 0.43750 | 0.000905 | 0.00054254 | 0.000002 | 0.0630000 |
| 32 | 4.541016 | 0.40625 | 4.101562 | 0.40625 | 0.001573 | 0.00127845 | 0.000003 | 0.0900000 |
| 64 | 4.321289 | 0.40625 | 4.101562 | 0.40625 | 0.003449 | 0.00235518 | 0.000007 | 0.0797143 |
| 128 | 4.211426 | 0.40625 | 4.101562 | 0.40625 | 0.013840 | 0.00235802 | 0.000006 | 0.1890000 |
| 256 | 4.156494 | 0.40625 | 4.101563 | 0.40625 | 0.012215 | 0.01070905 | 0.000012 | 0.1905000 |
| 512 | 4.129028 | 0.40625 | 4.101563 | 0.40625 | 0.026790 | 0.01955099 | 0.000022 | 0.2086364 |
| 1024 | 4.115295 | 0.40625 | 4.101562 | 0.40625 | 0.050985 | 0.04111254 | 0.000049 | 0.1877143 |
| 2048 | 4.108429 | 0.40625 | 4.101563 | 0.40625 | 0.101163 | 0.08290141 | 0.000096 | 0.1918125 |
| 4096 | 4.104996 | 0.40625 | 4.101563 | 0.40625 | 0.213360 | 0.15724752 | 0.002430 | 0.0151630 |
| 8192 | 4.103279 | 0.40625 | 4.101563 | 0.40625 | 0.451171 | 0.29746932 | 0.000286 | 0.2577273 |
| 16384 | 4.102421 | 0.40625 | 4.101563 | 0.40625 | 1.693254 | 0.31705493 | 0.000513 | 0.2874035 |
| 32768 | 4.101992 | 0.40625 | 4.101562 | 0.40625 | 7.529262 | 0.28521399 | 0.000902 | 0.3269335 |
| 65536 | 4.101777 | 0.40625 | 4.101562 | 0.40625 | 58.923778 | 0.14577933 | 0.001679 | 0.3512841 |
| 131072 | 4.101670 | 0.40625 | 4.101562 | 0.40625 |  |  | 0.003043 | 0.3876536 |
| 262144 | 4.101616 | 0.40625 | 4.101562 | 0.40625 |  |  | 0.006723 | 0.3509264 |
| 524288 | 4.101590 | 0.40625 | 4.101563 | 0.40625 |  |  | 0.017839 | 0.2645089 |
| 1048576 | 4.101579 | 0.40625 | 4.101565 | 0.40625 |  |  | 0.024390 | 0.3869277 |
| 2097152 | 4.101572 | 0.40625 | 4.101565 | 0.40625 |  |  | 0.053287 | 0.3542018 |
| 4194304 | 4.101568 | 0.40625 | 4.101564 | 0.40625 |  |  | 0.105457 | 0.3579537 |
| 8388608 | 4.101563 | 0.40625 | 4.101561 | 0.40625 |  |  | 0.204318 | 0.3695096 |
| 16777216 | 4.101517 | 0.40625 | 4.101516 | 0.40625 |  |  | 0.459274 | 0.3287687 |
| 33554432 | 4.100841 | 0.40625 | 4.100841 | 0.40625 |  |  | 0.823110 | 0.3668888 |
| 67108864 | 4.090507 | 0.40625 | 4.090507 | 0.40625 |  |  | 1.879811 | 0.3212981 |
| 134217728 | 3.923872 | 0.40181 | 3.923872 | 0.40181 |  |  | 3.314395 | 0.3644585 |
| 268435456 | 2.303370 | 0.37245 | 2.303370 | 0.37245 |  |  | 16.960100 | 0.1424472 |
| 536870912 | 0.965351 | 0.37494 | 0.965351 | 0.37494 |  |  | 88.737658 | 0.0544508 |

Poisson’s Equation in 2D

In this section I consider a 2D Poisson Equation which can be described by splitting out domain into a grid of N points in the x direction and N points in the y direction.   
  
I then get the following equations that relate the points

In order to solve this system of equations I need be able to write it in the form , and so I arrange in the y lines. That is for N=4 I can write

My general matrix A then can be written as a composition of 2 matrices

A is then

For BGuass I then get a matrix A\_banded with a band of size , and total size of and looks like

For large N this is considerably smaller and thus allows for a lot more iterations before running out of memory.

# Table that was asked for

# Contour Plot

Mastery Section: Poisson’s Equation in 3D

Appendix

# 

# Tables

Here I have recreated some tables that I produced throughout this project.

## BGauss Poisson 1D

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | ROUGH | | | | | | SMOOTH | | | | |
| 2 | N | maxval | maxval pos | cpu time | wall time | speed | maxval | maxval pos | cpu time | wall time | speed |
| 3 | 8 | 5.781250 | 3 | 0.0000060 | 0.0000020 | 0.0270000 | 4.062500 | 3 | 0.0000010 | 0.0000010 | 0.0540000 |
| 4 | 16 | 4.960937 | 7 | 0.0000010 | 0.0000020 | 0.0630000 | 4.062500 | 6 | 0.0000010 | 0.0000020 | 0.0630000 |
| 5 | 32 | 4.541016 | 13 | 0.0000020 | 0.0000020 | 0.1350000 | 4.101562 | 13 | 0.0000020 | 0.0000030 | 0.0900000 |
| 6 | 64 | 4.321289 | 26 | 0.0000030 | 0.0000040 | 0.1395000 | 4.101562 | 26 | 0.0000060 | 0.0000070 | 0.0797143 |
| 7 | 128 | 4.211426 | 52 | 0.0000060 | 0.0000060 | 0.1890000 | 4.101562 | 52 | 0.0000060 | 0.0000060 | 0.1890000 |
| 8 | 256 | 4.156494 | 104 | 0.0000140 | 0.0000140 | 0.1632857 | 4.101563 | 104 | 0.0000110 | 0.0000120 | 0.1905000 |
| 9 | 512 | 4.129028 | 208 | 0.0000250 | 0.0000240 | 0.1912500 | 4.101563 | 208 | 0.0000210 | 0.0000220 | 0.2086364 |
| 10 | 1024 | 4.115295 | 416 | 0.0000480 | 0.0000490 | 0.1877143 | 4.101562 | 416 | 0.0000490 | 0.0000490 | 0.1877143 |
| 11 | 2048 | 4.108429 | 832 | 0.0000960 | 0.0000950 | 0.1938316 | 4.101563 | 832 | 0.0000950 | 0.0000960 | 0.1918125 |
| 12 | 4096 | 4.104996 | 1664 | 0.0001920 | 0.0001930 | 0.1909119 | 4.101563 | 1664 | 0.0011030 | 0.0024300 | 0.0151630 |
| 13 | 8192 | 4.103279 | 3328 | 0.0003510 | 0.0003500 | 0.2106000 | 4.101563 | 3328 | 0.0002870 | 0.0002860 | 0.2577273 |
| 14 | 16384 | 4.102421 | 6656 | 0.0005050 | 0.0005060 | 0.2913795 | 4.101563 | 6656 | 0.0005130 | 0.0005130 | 0.2874035 |
| 15 | 32768 | 4.101992 | 13312 | 0.0010540 | 0.0010530 | 0.2800513 | 4.101562 | 13312 | 0.0009020 | 0.0009020 | 0.3269335 |
| 16 | 65536 | 4.101777 | 26624 | 0.0016730 | 0.0016720 | 0.3527548 | 4.101562 | 26624 | 0.0016790 | 0.0016790 | 0.3512841 |
| 17 | 131072 | 4.101670 | 53248 | 0.0032930 | 0.0038610 | 0.3055245 | 4.101562 | 53248 | 0.0030440 | 0.0030430 | 0.3876536 |
| 18 | 262144 | 4.101616 | 106496 | 0.0074800 | 0.0101530 | 0.2323725 | 4.101562 | 106496 | 0.0067240 | 0.0067230 | 0.3509264 |
| 19 | 524288 | 4.101590 | 212992 | 0.0161720 | 0.0161760 | 0.2917022 | 4.101563 | 212992 | 0.0145270 | 0.0178390 | 0.2645089 |
| 20 | 1048576 | 4.101579 | 425984 | 0.0264130 | 0.0297330 | 0.3173970 | 4.101565 | 425984 | 0.0243900 | 0.0243900 | 0.3869277 |
| 21 | 2097152 | 4.101572 | 851968 | 0.0488250 | 0.0488250 | 0.3865714 | 4.101565 | 851968 | 0.0532680 | 0.0532870 | 0.3542018 |
| 22 | 4194304 | 4.101568 | 1703936 | 0.1016840 | 0.1017160 | 0.3711188 | 4.101564 | 1703936 | 0.1054220 | 0.1054570 | 0.3579537 |
| 23 | 8388608 | 4.101563 | 3407872 | 0.2005660 | 0.2005640 | 0.3764258 | 4.101561 | 3407872 | 0.2043200 | 0.2043180 | 0.3695096 |
| 24 | 16777216 | 4.101517 | 6815714 | 0.4773880 | 0.4778520 | 0.3159868 | 4.101516 | 6815714 | 0.4589440 | 0.4592740 | 0.3287687 |
| 25 | 33554432 | 4.100841 | 13631372 | 0.9202720 | 0.9229430 | 0.3272032 | 4.100841 | 13631372 | 0.8230980 | 0.8231100 | 0.3668888 |
| 26 | 67108864 | 4.090507 | 27263101 | 1.6467740 | 1.6467700 | 0.3667663 | 4.090507 | 27263101 | 1.8798350 | 1.8798110 | 0.3212981 |
| 27 | 134217728 | 3.923872 | 53929750 | 3.2325200 | 3.2327580 | 0.3736622 | 3.923872 | 53929750 | 3.3144030 | 3.3143950 | 0.3644585 |
| 28 | 268435456 | 2.303370 | 99977615 | 23.9114370 | 28.8705990 | 0.0836810 | 2.303370 | 99977615 | 15.6768500 | 16.9601000 | 0.1424472 |
| 29 | 536870912 | 0.965351 | 201295359 | 63.9122660 | 85.7657210 | 0.0563376 | 0.965351 | 201295359 | 67.8584030 | 88.7376580 | 0.0544508 |

# 

# Graphs

# Representative Code