ACSE-5 ADVANCED PROGRAMMING

1. Introduction

programming team name: linabell
members: Zhiyu Zhang, Zonghui Liu, Xinyan Kong
GitHub repo: https://github.com/acse-2020/group-assignment-linabell
Date: January 30, 2022

2. Generate matrix

The input to the dense matrix solver is a randomly generated diagonal dominant square matrix, as it is more generous in its matrix requirements. The input to a sparse matrix solver can only be a real symmetric positive definite matrix. Because there is an open root operation in the formula, it must be a positive real number and there must be values on the diagonal.

3. Linear solver structure

Our linear solver system is divided into four main parts: matrix class (constructs matrices, both dense and sparse, with sparse inheriting from dense) , solver (linear matrix solver, both dense and sparse) , test (test file) comparison (compares the time taken to solve various solutions) (fig. 1).

To facilitate the choice of solver, an interface has been designed. The user can choose the type of matrix he wants to use and the methods he wants to use.

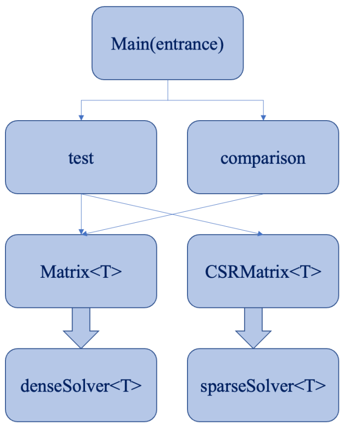


Figure 1. Structure of the code

4. Dense matrix solver class

The methods for solving linear system (Ax=b, where A is a dense matrix) can be roughly divided into three parts: the direct dense solvers, the stationary iterative methods and the non-stationary iterative methods. The difference between the stationary methods and the non-stationary one is that only the iterative solution x(k) is converging in the iteration.

4.1 Gaussian Solver (Direct)

The Gaussian Solver is a basic but important algorithm around which the other direct solvers are developed. The Gaussian Solver has two steps: **Gauss Elimination** and **Backward Substitution**. The residual of it can be very small but the algorithm needs more memory, so it is not suitable for solving large linear systems.

4.2 LU decomposition (Direct)

The coefficient matrix A can be divided **two triangular matrices**, one upper(U) which is a result of applying the Gauss Elimination on A and one lower(L). The algorithm stores the common factors into the lower triangular matrix(L) resulting in the **Doolittle's** algorithm. From L and U, it is simple to apply a forward substitution and backward substitution in order to solve the resulting system. Note that the partial pivoting is to make the solver more stable.

4.3 Jacobi and Gauss-Seidel methods (Stationary Iterative)

The iterative methods will always need more time than the direct solvers. But they are **self-correcting**, which means that round-off errors will be corrected in the subsequent cycles. The difference between the two methods is that the G-S solver updates guess(x(k)) dynamically with a relaxation factor w(fig.2). The purpose of adding relaxation factor is to **accelerate convergence**.

$$x_i \leftarrow \frac{1}{A_{ii}}(b_i - \sum_{j=1,j=/=i}^{n} A_{ij}x_j) + (1 - \omega)x_i$$

Figure 2. G-S formula

4.4 Conjugate Gradient Method (Non-stationary Iterative)

CG is simply the method of Conjugate Directions where the search directions are constructed by conjugation of the residuals and the search direction is a set of orthogonal vectors. The advantages are this method can use less memory and spend less time than the stationary methods, but it can be affected by the boundary conditions easily. Unfortunately, due to time limitation, we can not get a correct solution of CG.

4.5 Comparison of performance

The comparison graph shows that the direct solutions are very sensitive to the size of the matrix, so they are not suitable for solving huge linear systems. The graph also shows that the efficiency of Jacobi and G-S is similar and before size=800, the efficiency of G-S is higher than that of Jacobi. We are not sure about the reason for the special case when size=800, but we speculate that it may be due to the selection of relaxation factor.

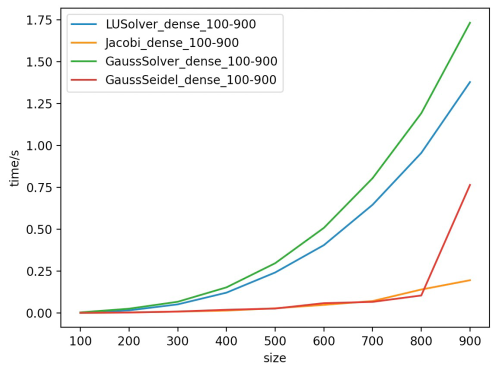


Fig 3. Comparison of performance

5. Sparse matrix solver class

5.1 Conjugate Gradient Method

The conjugate gradient method is great for solving large sparse matrices. It is an iterative algorithm that theoretically yields an answer in n steps, with 3n to 5n iterations considering rounding errors. The speed of convergence of the conjugate gradient method depends on the spectral distribution of the coefficient matrix. When the eigenvalues are relatively concentrated and the condition number of the coefficient matrix is small, the conjugate gradient method converges quickly.

5.2 Cholesky method

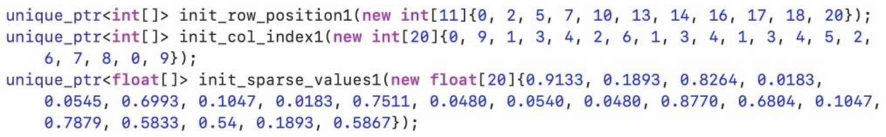
5.2.1 Cholesky decomposition

When A is an SPD (real Symmetric positive definite matrix), it can be decomposed into a lower triangle matrix L and its transpose, the upper triangle, which can be proved by induction that this decomposition must exist and is unique. In practice, if the matrix is positive definite, the Cholesky decomposition is more efficient and numerically stable than the LU decomposition.

5.2.2 Cholesky solver

After decomposition, we get an upper triangular matrix and a lower triangular matrix and solving the system of equations requires the use of Forward & Backward Substitution. Forward： Consider a set of equations in a matrix form L*y=b , where L is a lower triangular matrix with non-zero diagonal elements. Backward： Consider a set of equations in a matrix form L.T*x=y , where L.T is a upper triangular matrix with non-zero diagonal elements.

Test cases for sparseMatrix 10*10:

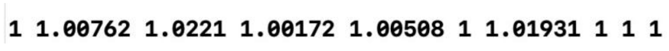


```
unique_ptr<int[]> init_row_position1(new int[11]{0, 2, 5, 7, 10, 13, 14, 16, 17, 18, 20});
unique_ptr<int[]> init_col_index1(new int[20]{0, 9, 1, 3, 4, 2, 6, 1, 3, 4, 1, 3, 4, 5, 2,
    6, 7, 8, 0, 9});
unique_ptr<float[]> init_sparse_values1(new float[20]{0.9133, 0.1893, 0.8264, 0.0183,
    0.0545, 0.6993, 0.1047, 0.0183, 0.7511, 0.0480, 0.0540, 0.0480, 0.8770, 0.6804, 0.1047,
    0.7879, 0.5833, 0.54, 0.1893, 0.5867});
```

Figure 3. test case

Vector b []: [1.1026, 0.8987, 0.804, 0.8174, 0.979, 0.6804, 0.8926, 0.5833, 0.54, 0.776]
Expected output: [1, 1, 1, 1, 1, 1, 1, 1, 1, 1]

Actual output:



```
1 1.00762 1.0221 1.00172 1.00508 1 1.01931 1 1 1
```

6. Output of solver

In dense solver, we can get the running time of each solver (stored in txt files) and the results of the performance comparison can be shown in plot. And in sparse solver, we can get the matrix and the result in terminal.

7. Division of work

Zonghui Liu: dense Gaussian, dense LU, dense Jacobi and Gauss-Seidel methods, dense Conjugate Gradient Method, test and performance comparison

Zhiyu Zhang: sparse Cholesky method, sparse Conjugate Gradient Method, user interface, test and performance comparison

Xinyan Kong: sparse Cholesky method, test, report, README

Reference

Vardit Cohen, Brian Hillman, and Kyle Suarez, 2012. Linear Algebra: Gaussian Elimination. [Online]
Available at: https://people.cs.rutgers.edu/~venugopa/parallel_summer2012/ge.html
[Accessed 24 January 2022].

GreeksforGreeks, 2021. Mathematics | L U Decomposition of a System of Linear Equations. [Online]
Available at: https://www.geeksforgeeks.org/l-u-decomposition-system-linear-equations/
[Accessed 23 January 2022].

GreeksforGreeks, 2021. Doolittle Algorithm : LU Decomposition. [Online]
Available at: https://www.geeksforgeeks.org/doolittle-algorithm-lu-decomposition/
[Accessed 24 January 2022].

Terry D. Johnson , 2000. Gaussian Elimination with Partial Pivoting . [Online]
Available at: https://web.mit.edu/10.001/Web/Course_Notes/GaussElimPivoting.html
[Accessed 23 January 2022].

Hailiang Zhao, 2021. Understanding the conjugate gradient method - top. [Online]
Available
at: https://hliangzhao.cn/articles/000001629787822242cbdf095acf4981bbdf6303f8e5f9cd000
[Accessed 26 January 2022].

Hailiang Zhao, 2021. Understanding the conjugate gradient method - below. [Online]
Available
at: https://hliangzhao.cn/articles/000001629787864466c3bf89af064543ad5f21272b027bf0000
[Accessed 26 January 2022].

Jonathan Richard Shewchuk, 1994. An Introduction to the Conjugate Gradient Method Without the Agonizing Pain. Technical Report

Mathuranathan, 2013. GaussianWaves. [Online]
Available at: https://www.gaussianwaves.com/2013/05/solving-a-triangular-matrix-using-forward-backward-substitution/
[Accessed 25 January 2022].

The MathWorks, 2021. sprandsym. [Online]
Available at: https://ch.mathworks.com/help/matlab/ref/sprandsym.html
[Accessed 27 January 2022].

Albers Uzila, 2021. Complete Step-by-step Conjugate Gradient Algorithm from Scratch. [Online]

Available at: https://towardsdatascience.com/complete-step-by-step-conjugate-gradient-algorithm-from-scratch-202c07fb52a8
[Accessed 27 January 2022].