Writeup CME 211 Project Part 1

Thomas Brink (06446502)

November 16, 2021

1 Algorithm

In Algorithm 1, we provide pseudo-code for the conjugate gradient algorithm we use to solve the linear system. Note that, in this algorithm, we use **bold** notation for vectors and matrices.

Algorithm 1: Conjugate gradient pseudo-code

Data: Sparse CSR matrix $\mathbf{A}_{n\times n}$; linear system outcome $\mathbf{b}$; tolerance level *tol*

Result: Number of iterations i^* required to reach convergence; solution vector $\mathbf{u}^*$

Initialize $\mathbf{u}_0 = \mathbf{1}_n$; $i = 0$; $i_{max} = n$; $\mathbf{r}_0 = \mathbf{b} - \mathbf{A}\mathbf{u}_0$; $\mathbf{p}_0 = \mathbf{r}_0$

while $i < i_{max}$ **do**

- $i := i + 1$
- $\alpha := \dfrac{\mathbf{r}_{i-1}^{\mathrm{T}}\mathbf{r}_{i-1}}{\mathbf{p}_{i-1}^{\mathrm{T}}\mathbf{A}\mathbf{p}_{i-1}}$
- $\mathbf{u}_i = \mathbf{u}_{i-1} + \alpha\mathbf{p}_{i-1}$
- $\mathbf{r}_i = \mathbf{r}_{i-1} - \alpha\mathbf{A}\mathbf{p}_{i-1}$
- **if** $\dfrac{||\mathbf{r}_i||_2}{||\mathbf{r}_0||_2} < tol$ **then**
  - $i^* := i$
  - $\mathbf{u}^* = \mathbf{u}_i$
  - break
- $\beta := \dfrac{\mathbf{r}_i^{\mathrm{T}}\mathbf{r}_i}{\mathbf{r}_{i-1}^{\mathrm{T}}\mathbf{r}_{i-1}}$
- $\mathbf{p}_i = \mathbf{r}_i + \beta\mathbf{p}_{i-1}$

2 Code Breakdown

In this project, we provide code for solving a linear sparse system using conjugate gradient. The code is broken down in several parts, which we will briefly discuss sequentially below. The majority of code for implementing the conjugate gradient solver is captured by the *CGSolver.cpp* and *matvecops.cpp* files, in addition to which several other files implement further functionality.

- **CGSolver.cpp**. In the *CGSolver.cpp* file, we perform the conjugate gradient algorithm provided in Algorithm 1. This algorithm finds a solution $\mathbf{x}^*$ to the system $\mathbf{Ax} = \mathbf{b}$, where $\mathbf{A}$ is sparse (CSR) and b is given beforehand. Our code returns the number of iterations until convergence. For this algorithm, we need to define several matrix and vector operations. Instead of doing so within the algorithm for each separate line, we make use of a separate file with functions for specific operations, called *matvecops.cpp*. This avoids redundancy in the code as we avoid excess code for similar operations (e.g., lines 5 and 12 in Algorithm 1).

- **matvecops.cpp**. As previously stated, in the *matvecops.cpp* file, we define useful matrix-vector operations for the conjugate gradient algorithm. We break these operations down into several small functions that all represent a single operation, which should make the code modular, less redundant, and easy to understand. The functions include:

  1. *vecSum*: this function performs vector addition given two input vectors.
  2. *vecScalarProd*: this function multiplies an input vector by a scalar.
  3. *dotProduct*: this function computes the dot product between two input vectors.
  4. *L2Norm*: this function computes the l_2-norm of a given vector.
  5. *matVecProd*: this function computes the matrix-vector product of a CSR-type sparse matrix and a given vector.

- **COO2CSR.cpp**. In this function, which was provided by the instructor, a COO matrix is transformed into a CSR matrix. We do so as the previous matrix-vector operations work well with the CSR format.

- **main.cpp**. In the *main.cpp* file, we provide the main functionality for running our code. That is, the main file i) takes a matrix input file and a solution file name, ii) performs the conjugate gradient algorithm (and the required prior computations, such as the COO to CSR transform), iii) writes the solution $\mathbf{x}^*$ to the solution file, and iv) prints a 'success' or 'error' message.

- **Header files**. Besides providing the code definitions in the .cpp files, we provide function declarations in separate header files. *matvecops.hpp*, *COO2CSR.hpp*, and *CGSolver.hpp*.

- **makefile**. In addition to the above files, we provide a makefile to compile our program.