GMRES Project

Applied Linear Algebra

Duc Phan

March 12, 2019

Abstract

This project objective is to create a set of tools to that mainly help with operating and manipulating on sparse matrices in compressed sparse row (CSR) format. The library I chose to implement cover most of the basic operations, which will be discussed later in this paper, for *Vectors*, *Dense Matrices* and *CSR Matrices*. The focus of this project and paper is the Generalized Minimal Residual (GMRES) method/algorithm. This paper is an overview of the tool set mentioned above, the implementation of GMRES and some statistics produced by GMRES.

1. Introduction

This project is implemented mostly in Java for the *Vector, Dense/CSR Matrix* operations and NodeJS for data and stats graphing.

Some libraries I used in my program are:

- Java:

```
//For parsing String into json
import com.google.gson.Gson;
//IO handler
import java.io.File;
import java.io.FileNotFoundException;
import java.io.PrintStream;
import java.util.Scanner;
//Number handler - This will support number with a lot of decimals
import java.math.BigDecimal;
import java.math.MathContext;
import java.math.RoundingMode;
//Database
import java.util.LinkedList;
```

- NodeJS:

```
require('fs');          //For file reading/writing
require('path');        //For file searching
require('plotly');      //For graphing
```

The code and some sample results can be found at my GitHub repository:

https://github.com/ptmdmusique/Vector_Dense-CSR-Matrix_Operations

2. Acceptable Inputs

The program will be able to take and parse input of string type in multiple format:

- A string of numbers separated by single space for different columns and newline character (‘\n’) for different rows:

  A vector or matrix can be constructed directly using:

  1. the object’s constructor *Vector(String input)* or *Matrix(String input)* or *CSRMatrix(String input)*.
  2. the *TakeInput(String input)* method.

Example:

```
//This will create a vector with entries 1, 2, 3, 4, 5
Vector myVector = new Vector("1 2 3 4 5");

/* This will create a 2x4 matrix:
   1 2 3 4
   5 6 7 8
    (extra spaces after the last and before the first numbers of each row
can lead to bugs!)
*/
Matrix myMatrix = new Matrix("1 2 3 4\n5 6 7 8");
CSRMatrix myCSRMatrix = new CSRMatrix("1 2 3 4\n5 6 7 8");
```

- Matrix Market Exchange Format:

```
%%MatrixMarket matrix coordinate real general
%=================================================================================
%
% This ASCII file represents a sparse MxN matrix with L
% nonzeros in the following Matrix Market format:
%
% +----------------------------------------------+
% |%%MatrixMarket matrix coordinate real general | <--- header line
% |%                                             | <--+
% |% comments                                    |    |-- 0 or more comment lines
% |%                                             | <--+
% |    M  N  L                                   | <--- rows, columns, entries
% |    I1  J1  A(I1, J1)                         | <--+
% |    I2  J2  A(I2, J2)                         |    |
% |    I3  J3  A(I3, J3)                         |    |-- L lines
% |        . . .                                 |    |
% |    IL JL  A(IL, JL)                          | <--+
% +----------------------------------------------+
%
% Indices are 1-based, i.e. A(1,1) is the first element.
%
%=================================================================================
  5  5  8
    1     1   1.000e+00
    2     2   1.050e+01
    3     3   1.500e-02
    1     4   6.000e+00
    4     2   2.505e+02
    4     4  -2.800e+02
    4     5   3.332e+01
    5     5   1.200e+01
```

3.