COL 100M - Lab Exam 2

March 17, 2018

Instructions

- This exam consists of five questions, each with sub parts.
- No notes, phones, local or internet resources are allowed.
- Submit the following files: **test1.ml, test2.ml, test3.ml, test4.ml, test5.ml**
- For any file you should directly call the functions declared in the previous files using **open**. For example in test3.ml you can use the functions declared in test2.ml by writing **open Test2**
- A password will be announced in class that you can use to submit code on Moodle. You will be allowed to submit / evaluate your code atmost 15 times.

1 Gaussian Elimination Algorithm

The Gaussian elimination algorithm is used to solve systems of linear equations using simple row transformations on a matrix. Given b, a vector of size m, A, an $m \times n$ matrix, the goal is to compute a solution vector x such that the matrix equation $Ax = b$ is satisfied. Consider the following system of equations.

$$\begin{aligned} x_1 + x_2 + x_3 &= 4 \\ x_1 + 2x_2 - x_3 &= 1 \\ 2x_1 - x_2 + x_3 &= 3 \end{aligned}$$

This system is represented by the following 2 matrices. The matrix A is the co-efficient matrix, where each row corresponds to the *co-efficients* of x_1, x_2 and x_3 respectively.

$$A = \begin{bmatrix} 1 & 1 & 1 \\ 1 & 2 & -1 \\ 2 & -1 & 1 \end{bmatrix}$$

$$b = \begin{bmatrix} 4 \\ 1 \\ 3 \end{bmatrix}$$

The solution to the matrix equation $Ax = b$ is

$$x = \begin{bmatrix} 1 \\ 1 \\ 2 \end{bmatrix}$$

The Gaussian elimination algorithm works by reducing the *augmented* matrix $[A \quad b]$ to its equivalent *Row Echelon* form using row transformations. A matrix is in row echelon form if it satisfies the following conditions.

- The first non-zero element in each row, called the *leading entry*, is in a column to the right of the leading entry in the previous row.

- Rows with all zero elements, if any, are below rows having at least one non-zero element.

In this example, the matrix $[A \quad b]$

$$\begin{bmatrix} 1 & 1 & 1 & 4 \\ 1 & 2 & -1 & 1 \\ 2 & -1 & 1 & 3 \end{bmatrix}$$

is reduced to its equivalent row echelon form

$$\begin{bmatrix} 1 & 1 & 1 & 4 \\ 0 & 1 & -2 & -3 \\ 0 & 0 & -7 & -14 \end{bmatrix}$$

which is equivalent to the reduced system of equations

$$\begin{aligned} x_1 + x_2 + x_3 &= 4 \\ x_2 - 2x_3 &= -3 \\ -7x_3 &= -14 \end{aligned}$$

From this system, Back-substitution is used to obtain the solution $x = [1, 1, 2]$.

2 Exam

Your goal in this exam is to implement the gaussian elimination algorithm. Please follow the steps below to ensure you get full credit for your solution. *Note:* Even though we provide inputs and expected outputs in a "list of lists" data structure, you are free to use your own representation. But, please ensure that your function signatures match those given below.

2.1 Input Validation

[**2 marks**] In the file **test1.ml** write a function `checkDimension : float list list -> float list -> bool`. `checkDimension A b` returns `true` if the number of rows in `A` is equal to the number of elements in `b`, and `A` is a valid matrix, `false` otherwise.

2.2 Row transformations

In this section, you will implement row transformations to be done on the augmented matrix $X = [A \quad b]$. In **test2.ml** file write the following functions.

(a) [**6 marks**] Write a function `swap : float list list -> int -> int -> float list list` such that `swap X i j` exchanges rows `i` and `j` of `X`. For example,

`swap [[1.0;2.0;3.0];[4.0;5.0;6.0];[7.0;8.0;9.0]] 1 2` returns

`[[1.0;2.0;3.0];[7.0;8.0;9.0];[4.0;5.0;6.0]]`

(b) [**6 marks**] Write a function `mult : float list list -> int -> float -> float list list`. `mult X i c` multiplies row `i` of `X` with a constant `c`. For example,

`mult [[1.0;2.0;3.0];[4.0;5.0;6.0];[7.0;8.0;9.0]] 1 2.0` returns

`[[1.0;2.0;3.0];[8.0;10.0;12.0];[7.0;8.0;9.0]]`

(c) [**6 marks**] Write a function `addRows : float list list -> int -> int -> float list list`. `addRows X i j` adds the rows `i` and `j` of `X` and puts in the result in row `i`.

`addRows [[1.0;2.0;3.0];[4.0;5.0;6.0];[7.0;8.0;9.0]] 0 1` returns

`[[5.0;7.0;9.0];[4.0;5.0;6.0];[7.0;8.0;9.0]]`

2.3 Row Echelon Form

The following algorithm converts the augmented matrix X to its row echelon form.

1. If X has only 1 row, return X.
2. Let c be the leftmost column of X with at least 1 non-zero entry.
3. Let i be any row such that $X[i, c] \neq 0$.
4. Swap rows i and 0 in X.
5. For all rows $j = 1 \ldots m$ of X replace row $X[j]$ with
$$X[j] = X[j] - \frac{X[j, c]}{X[0, c]} \times X[0]$$
6. Let the dimensions of X be $m \times (n + 1)$, and let X' be a matrix of size $(m - 1) \times (n + 1)$ which has all but the first row of X. Recursively compute Y', the row echelon form of X'. Return $X[0] :: Y'$

[**10 marks**] In the file **test3.ml** write a function `rowEchelon : float list list -> float list list` such that `rowEchelon X` returns the row echelon form of `X`. For example,
`rowEchelon [[1.0;1.0;1.0;4.0];[1.0;2.0;-1.0;1.0];[2.0;-1.0;1.0;3.0]]` returns
`[[1.0;1.0;1.0;4.0];[0.0;1.0;-2.0;-3.0];[0.0;0.0;-7.0;-14.0]]`

2.4 Infinite/Unique/No solutions

- In the row echelon form of the matrix X, if there is a row such that all entries are zero *except the entry in the last column*, then the system of equations has *no solution*.
- If no such row exists, and for each column c, there is a row i such that $X[i, c]$ is the first non-zero entry in row i,then the system has a *unique solution*.
- If there exists a column c for which there is no row i such that $X[i, c]$ is the leading entry of row i, then the system has an *infinite number of solutions*.

[**5 marks**] In **test4.ml** write a function `numSolutions : float list list -> int` such that `numSolutions X` returns `1` if the system of equations represented by `A` has a unique solution, `0` if it has no solution, and `max_int` otherwise.

2.5 Solve $Ax = b$

Given a matrix of size $m \times n$, and a vector b of size m, the Gaussian elimination algorithm is applied to the matrix $[A \quad b]$ of size $(m + 1) \times n$ to reduce it to its row echelon form. Assuming the system has a unique solution, back-substitution computes the vector x of size n such that $Ax = b$. For example the following system of equations is in row echelon form:

$$\begin{aligned} x_1 + x_2 + x_3 &= 4 \\ x_2 - 2x_3 &= -3 \\ -7x_3 &= -14 \end{aligned}$$

Starting from the last equation, we know that $x_3 = 2$. Substituting x_3 in the second equation, we obtain $x_2 = 1$. Substituting both x_2 and x_3 in the first equation, we arrive at $x_1 = 1$ and also the solution vector $[1, 1, 2]$.

In the **test5.ml** file write the following functions.h

- [**5 marks**] Write a function `solveRowEchelon : float list list -> float list` such that `solveRowEchelon X` assumes that `X` is in row echelon form and has a unique solution, and returns a vector with the solution to the system of equations represented by `X`.

- **[5 marks]**Write a function `solve : float list list -> float list -> float list`. `solve A b`
  - Raises exception `Dimension_mismatch` if `A` is not a valid matrix or if the number of rows in `A` is not equal to the number of elements in `b`.
  - Raises exception `No_solutions` if the system `[A b]` has no solutions.
  - Raises exception `Infinite_solutions` if the system `[A b]` has infinitely many solutions.
  - Returns vector `x` such that `Ax = b` if `[A b]` has a unique solution.