MAU11601:Introduction To Programming - Tutorial 6: Gaussian Elimination

Liam Burke

School Of Mathematics, Trinity College Dublin

Michaelmas Term 2022/2023

Goal of Tutorial...

- To understand Gaussian elimination and backward substitution for an arbitrary matrix $\mathbf{A}$ and vector $\mathbf{b}$, and understand what both algorithms are doing at some arbitrary step of the process.
- Understanding what the algorithm is doing at some arbitrary step is all we need in order to implement it on a computer.
- We will write MATLAB functions to perform Gaussian elimination and backward substitution and use both functions to solve a linear system.

Introduction

Gaussian Elimination

Gaussian elimination is used to solve a linear system $\mathbf{Ax} = \mathbf{b}$ by performing elementary row operations on the rows of $\mathbf{A}$ (and $\mathbf{b}$) to annihilate the elements below the main diagonal. The resulting upper triangular system can then be solved with backward substitution.

$$
\begin{bmatrix}
a_{1,1} & a_{1,2} & a_{1,3} & a_{1,4} & & & \cdots & a_{1,n} \\
{\color{red}a_{1,1}} & a_{2,2} & a_{2,3} & a_{2,4} & & & \cdots & a_{2,n} \\
{\color{red}a_{2,1}} & {\color{red}a_{2,1}} & a_{3,3} & a_{3,4} & & & \cdots & a_{3,n} \\
{\color{red}\vdots} & & & \ddots & & & & \vdots \\
 & & & & a_{i,i} & a_{i,i+1} & \cdots & a_{i,n} \\
 & & & & & \ddots & & \vdots \\
{\color{red}a_{n,1}} & {\color{red}a_{n,2}} & {\color{red}\cdots} & & & & & a_{n,n}
\end{bmatrix}
\begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ \vdots \\ x_i \\ \vdots \\ x_n \end{bmatrix}
=
\begin{bmatrix} b_1 \\ b_2 \\ b_3 \\ \vdots \\ b_i \\ \vdots \\ b_n \end{bmatrix}
$$

Gaussian Elimination

Algorithm 1 Gaussian Elimination

```
Input : Matrix A ∈ ℝ^{n×n}, vector b ∈ ℝ^n
for k = 1, 2, ..., n − 1 do
    for i = k + 1, k + 2, ..., n do
        m ← a_{i,k} / a_{k,k}
        for j = k, k + 1, k + 2, ..., n do
            a_{i,j} ← a_{i,j} − m a_{k,j}
        end for
        b_i ← b_i − m b_k
    end for
end for
```

Backward Substitution

Corollary

Gaussian Elimination reduces the problem of solving the linear system $\mathbf{Ax} = \mathbf{b}$ to solving an *upper triangular* linear system $\mathbf{Ux} = \mathbf{v}$.

$$\begin{bmatrix} u_{1,1} & u_{1,2} & u_{1,3} & u_{1,4} & & & \cdots & u_{1,n} \\ & u_{2,2} & u_{2,3} & u_{2,4} & & & \cdots & u_{2,n} \\ & & u_{3,3} & u_{3,4} & & & \cdots & u_{3,n} \\ & & & \ddots & & & & \vdots \\ & & & & u_{i,i} & u_{i,i+1} & \cdots & u_{i,n} \\ & & & & & \ddots & & \vdots \\ & & & & & & & u_{n,n} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ \vdots \\ x_i \\ \vdots \\ x_n \end{bmatrix} = \begin{bmatrix} v_1 \\ v_2 \\ v_3 \\ \vdots \\ v_i \\ \vdots \\ v_n \end{bmatrix}$$

Linear systems in this form are easily solved using *backward substitution.*

Backward Substitution cont...

- At step 1 we can easily solve for x_n via

$$x_n = \frac{v_n}{u_{n,n}}$$

- At step 2 we can use our computed x_n to solve for x_{n-1} via

$$x_{n-1} = \frac{1}{u_{n-1,n-1}}\left(v_{n-1} - u_{n-1,n}x_n\right)$$

$$\vdots$$

- At step i we can use all previously computed $x_{i+1}, x_{i+2}, \dots x_n$ to compute x_i via

$$x_i = \frac{1}{u_{i,i}}\left(v_i - \sum_{j=i+1}^{n} u_{i,j}x_j\right)$$

Backward Substitution cont...

Algorithm 2 Backward Substitution

Input : Upper triangular matrix $\mathbf{U} \in \mathbb{R}^{n \times n}$, vector $\mathbf{v} \in \mathbb{R}^n$
for $i = n, n-1, \ldots, 1$ **do**
    Set $s := 0$
    for $j = i+1, i+2, \ldots, n$ **do**
        $s \leftarrow s + u_{i,j} x_j$
    end for
    $x_i \leftarrow \frac{1}{u_{i,i}} \left(v_i - s \right)$
end for