
Gaussian Elimination

Tamas Kis | tamas.a.kis@outlook.com | <https://tamaskis.github.io>

CONTENTS

1 Gaussian Elimination	2
References	4

Copyright © 2021 Tamas Kis

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.



1 GAUSSIAN ELIMINATION

Gaussian elimination can be used to solve the linear system

$$\mathbf{Ax} = \mathbf{b}$$

where $\mathbf{A} \in \mathbb{R}^{n \times n}$ and $\mathbf{x}, \mathbf{b} \in \mathbb{R}^n$. In Algorithm 1 below, we will be referring to the rows and columns of \mathbf{A} as well as to the elements of \mathbf{x} at different points. Here are the conventions we use:

$$\mathbf{A} = [\mathbf{a}_1 \quad \mathbf{a}_2 \quad \dots \quad \mathbf{a}_j \quad \dots \quad \mathbf{a}_{n-1} \quad \mathbf{a}_n] = \begin{bmatrix} \bar{a}_1 \\ \bar{a}_2 \\ \vdots \\ \bar{a}_i \\ \vdots \\ \bar{a}_{n-1} \\ \bar{a}_n \end{bmatrix}, \quad \mathbf{x} = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_i \\ \vdots \\ x_{n-1} \\ x_n \end{bmatrix}$$

Algorithm 1 below implements Gaussian elimination with partial pivoting and is adapted from Algorithm 6.2 in [1, pp. 374–375]. Note that ε is used to denote the machine epsilon.

Algorithm 1: gaussian_elimination

Gaussian elimination with partial pivoting.

Given:

- $\mathbf{A} \in \mathbb{R}^{n \times n}$ - matrix
- $\mathbf{b} \in \mathbb{R}^n$ - vector

Note:

- \mathbf{A} and \mathbf{b} define the linear system $\mathbf{Ax} = \mathbf{b}$.

Procedure:

1. Determine n (where $\mathbf{A} \in \mathbb{R}^{n \times n}$).
2. Redefine \mathbf{A} by augmenting the original \mathbf{A} with \mathbf{b} .

$$\mathbf{A} = [\mathbf{A} \quad \mathbf{b}]$$

3. Initialize a boolean variable to keep track of if the matrix is singular.

$$\text{singular} = \text{false}$$

4. Perform the elimination process.

$$\text{for } i = 1 \text{ to } n - 1$$

- (a) Determine the pivot row as follows:
- Define the vector $\mathbf{p} = (i, i + 1, \dots, n - 1, n)^T$.
 - Remove the elements p_k of \mathbf{p} where $A_{k,i} = 0$.
 - Find the index (p) of the pivot row.

$$p = \min(\mathbf{p})$$

- (b) Exit the loop if all possible pivots in the i^{th} column are zero (to within machine precision) because that would result in a singular matrix.

```

if  $\max |\mathbf{a}_i| \leq \varepsilon$ 
    singular = true
    Exit loop – the matrix is singular and no unique solution exists.
end

```

- (c) Switch pivot row with the i^{th} row if $p \neq i$.

```

if  $p \neq i$ 
    Store the  $i^{\text{th}}$  row of  $\mathbf{A}$  as  $\bar{a}_i$ .
    Store the  $p^{\text{th}}$  row of  $\mathbf{A}$  as  $\bar{a}_p$ .
    Set the  $i^{\text{th}}$  row of  $\mathbf{A}$  to  $\bar{a}_p$ .
    Set the  $p^{\text{th}}$  row of  $\mathbf{A}$  to  $\bar{a}_i$ .
end

```

- (d) Perform the elementary row operation.

```

for  $j = i + 1$  to  $n$ 
     $\bar{a}_j = \bar{a}_j - \left( \frac{A_{j,i}}{A_{i,i}} \right) \bar{a}_i$ 
end

```

end

5. Determine if \mathbf{A} is singular (if the bottom right element of \mathbf{A} is 0 (to within machine precision), then the entire bottom row is 0 and \mathbf{A} is singular).

```

if  $|A_{n,n}| \leq \varepsilon$ 
    singular = true
end

```

6. Preallocate $\mathbf{x} \in \mathbb{R}^n$ to store the solution.
 7. Perform backward substitution to solve $\mathbf{Ax} = \mathbf{b}$ if \mathbf{A} is nonsingular.

if !(singular)

```
      |  $x_n = \frac{A_{n,n+1}}{A_{n,n}}$   
      | for  $i = n - 1$  to 1 by -1  
      |   |  $S = 0$   
      |   | for  $j = i + 1$  to  $n$   
      |   |   |  $S = S + A_{i,j}x_j$   
      |   |   | end  
      |   |  $x_i = \frac{A_{i,n+1} - S}{A_{i,i}}$   
      |   | end  
      | end
```

Return:

- $\mathbf{x} \in \mathbb{R}^n$ - solution of the linear system $\mathbf{Ax} = \mathbf{b}$

REFERENCES

- [1] Richard L. Burden and J. Douglas Faires. *Numerical Analysis*. 9th. Boston, MA: Brooks/Cole, Cengage Learning, 2011.