

# Gaussian Elimination

## *MATLAB Implementation*

---

Tamas Kis | [kis@stanford.edu](mailto:kis@stanford.edu)

TAMAS KIS  
<https://github.com/tamaskis>

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.*



# Contents

<b>gaussian_elimination</b>	<b>4</b>
Syntax . . . . .	4
Description . . . . .	4
Examples . . . . .	4
Links . . . . .	5
<b>Gaussian Elimination</b>	<b>6</b>
<b>References</b>	<b>8</b>

## gaussian\_elimination

---

Solves the linear system  $\mathbf{Ax} = \mathbf{b}$  for  $\mathbf{x}$  using Gaussian elimination with partial pivoting.

### Syntax

---

`x = gaussian_elimination(A,b)`

### Description

---

`x = gaussian_elimination(A,b)` solves the linear system  $\mathbf{Ax} = \mathbf{b}$  for the vector  $\mathbf{x} \in \mathbb{R}^n$ , where  $\mathbf{A} \in \mathbb{R}^{n \times n}$  and  $\mathbf{b} \in \mathbb{R}^n$ .

### Examples

---

#### Example 1

Solve the linear system  $\mathbf{Ax} = \mathbf{b}$  for  $\mathbf{x}$ , where

$$\mathbf{A} = \begin{bmatrix} 2 & -1 & 5 \\ 1 & 1 & -3 \\ 2 & 4 & 1 \end{bmatrix}, \quad \mathbf{b} = \begin{bmatrix} 10 \\ -2 \\ 1 \end{bmatrix}$$

#### ■ SOLUTION

Entering  $\mathbf{A}$  and  $\mathbf{b}$  into MATLAB,

```
% defines matrix A
A = [2,-1, 5;
     1, 1,-3;
     2, 4, 1];

% defines vector b
b = [10;
     -2;
     1];
```

To solve the linear system for  $\mathbf{x}$ ,

```
x = gaussian_elimination(A,b)
```

This yields the result

```
x =
     2
    -1
     1
```

## Links

---

MATLAB® Central's File Exchange:

[https://www.mathworks.com/matlabcentral/fileexchange/89306-gaussian-elimination-gaussian\\_elimination](https://www.mathworks.com/matlabcentral/fileexchange/89306-gaussian-elimination-gaussian_elimination)

GitHub®:

[https://github.com/tamaskis/gaussian\\_elimination-MATLAB](https://github.com/tamaskis/gaussian_elimination-MATLAB)

# 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  $\mathbf{x}, \mathbf{b} \in \mathbb{R}^{n \times 1}$ . In Algorithms 1 and 2 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:

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

Algorithms 1 and 2<sup>1</sup> below implement Gaussian elimination with partial pivoting and are adapted from Algorithm 6.2 in [1, pp. 374–375]. Note that  $\varepsilon$  is used to denote the machine epsilon.

---

<sup>1</sup> Note that together, these algorithms represent a single algorithm. However, this single algorithm is split into two due to its length and the limitations of typesetting algorithms in L<sup>A</sup>T<sub>E</sub>X.

---

**Algorithm 1:** Gaussian elimination with partial pivoting (part 1).

---

**1 Given:**  $\mathbf{A}$ ,  $\mathbf{x}$ ,  $\mathbf{b}$

*// determines  $n$  (where  $\mathbf{A} \in \mathbb{R}^{n \times n}$ )*

**2**  $n = \text{size}(\mathbf{A}, 1)$

*// redefines  $A$  by augmenting the original  $A$  with  $\mathbf{b}$*

**3**  $\mathbf{A} = [\mathbf{A} \ \mathbf{b}]$

*// initializes boolean variable to keep track if matrix is singular*

**4** singular = false

*// elimination process*

**5 for**  $i = 1$  **to**  $n - 1$  **do**

*// determines pivot row*

**6** Define the vector  $\mathbf{p} = (i, i + 1, \dots, n - 1, n)^T$ .

**7** Remove the elements  $p_k$  of  $\mathbf{p}$  where  $A_{k,i} = 0$ .

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

*// exit the loop if all possible pivots in the  $i^{\text{th}}$  column are zero (to machine precision) because that would result in a singular matrix*

**9 if**  $\max |\mathbf{a}_i| \leq \varepsilon$  **then**

**10** | singular = true

**11** | Exit loop – the matrix is singular and no unique solution exists.

**12 end**

*// switches pivot row with the  $i^{\text{th}}$  row if  $p \neq i$*

**13 if**  $p \neq i$  **then**

**14** | Store the  $i^{\text{th}}$  row of  $\mathbf{A}$  as  $\bar{a}_i$ .

**15** | Store the  $p^{\text{th}}$  row of  $\mathbf{A}$  as  $\bar{a}_p$ .

**16** | Set the  $i^{\text{th}}$  row of  $\mathbf{A}$  to  $\bar{a}_p$ .

**17** | Set the  $p^{\text{th}}$  row of  $\mathbf{A}$  to  $\bar{a}_i$ .

**18 end**

*// elementary row operation*

**19 for**  $j = i + 1$  **to**  $n$  **do**

**20** |  $\bar{a}_j = \bar{a}_j - \left( \frac{A_{j,i}}{A_{i,i}} \right) \bar{a}_i$

**21 end**

**22 end**

---

---

**Algorithm 2:** Gaussian elimination with partial pivoting (part 2).

---

```
// determines 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)
1 if  $|A_{n,n}| \leq \varepsilon$  then
2   | singular = true
3 end

4 Preallocate/initialize the solution vector  $\mathbf{x} \in \mathbb{R}^{n \times 1}$ .

// perform backward substitution to solve  $\mathbf{Ax} = \mathbf{b}$  if  $\mathbf{A}$  is nonsingular
5 if !(singular) then
6   |  $x_n = \frac{A_{n,n+1}}{A_{n,n}}$ 
7   | for  $i = n - 1$  to 1 by  $-1$  do
8     |  $S = 0$  for  $i + 1$  to  $n$  do
9       |  $S = S + A_{i,j}x_j$ 
10    | end
11    |  $x_i = \frac{A_{i,n+1} - S}{A_{i,i}}$ 
12  | end
13 end
```

---



## References

---

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