

More on Systems of Linear Equations

Iñaki Rañó

The Mærsk Mc Kinney Møller Institute

2021

- Review of Gaussian elimination
- Gaussian elimination algorithm details
- Geometric interpretation of Gaussian elimination rules
 - Row scaling
 - Linear combination of rows
- Gauss-Jordan for inverse calculation

Solving Systems of Linear Equations (Review)

Systems of linear equations of the form: $A\mathbf{v} = \mathbf{b}$

- Can be seen as the intersection of planes/hyperplanes
- Cramer's rule
- Back substitution (for A upper triangular)
- Gaussian Elimination + back substitution
 - Augmented matrix $\tilde{A} = [A|\mathbf{b}]$
 - Solution or not depends on ranks of A , \tilde{A} and number of unknowns
- If A has an inverse (i.e. A is square and $|A| \neq 0$) A^{-1} then:

$$A^{-1}A\mathbf{x} = A^{-1}\mathbf{b} \quad \rightarrow \quad \mathbf{x} = A^{-1}\mathbf{b}$$

- Gaussian elimination is more efficient than calculating A^{-1}
- Calculating A^{-1} is useful for other problems

Gaussian Elimination Review

Goal: To turn matrix \tilde{A} into upper triangular (row echelon form)

Allowed operations:

- Swap two rows
- Multiply one row by a number different than zero
- Substitute row ' i ' by row ' i ' plus another multiplied by a number

The second rule is not really necessary to make the matrix upper triangular (it is just convenient sometimes to have 1 as first non-zero entry in the row)

Gaussian Elimination Procedure Review

While there are non-zero columns and rows:

- 1 Make sure entry row 1, column 1 is not zero (swap rows 1 and $i > 1$ if necessary)
- 2 Take entry row 1, column 1 and use last rule to make all zeros under it (from row 2 to last row)
- 3 Make sure entry row 2, column 2 is not zero (swap rows 2 and $i > 2$ if necessary)
- 4 Take entry row 2, column 2 and use last rule to make all zeros under it (from row 3 to last row)
- \vdots \vdots
- k_0 Make sure entry row ' k ', column ' k ' is not zero (swap rows ' k ' and $i > k$ if necessary)
- k_1 Move to entry row i , column i and use last rule to make all zeros under it

Gaussian Elimination as an Algorithm

For a matrix A of size $m \times n$

- The steps to do Gaussian elimination can be easily automated in a computer
- The current entry (row ' k ', column ' k ') is called **pivot**
- Take as pivot row ' k ', column ' k ' (if not zero, swap otherwise if possible) for $k = 1, 2, \dots, \min(m, n)$ and turn all entries under it to zero (row $k + 1, k + 2, \dots, m$, column ' k ')

Gaussian Elimination Algorithm

Input: Matrix A of size $m \times n$

Output: Matrix A in upper triangular (row echelon) form

$pivot_row \leftarrow 1$

▷ Initial pivot

$pivot_col \leftarrow 1$

while $pivot_row \leq m$ **and** $pivot_col \leq n$ **do**

$idx_max \leftarrow index_of_max_abs(A, pivot_row, pivot_col)$

▷ See later

if $A(idx_max, pivot_col) = 0$ **then**

▷ 'Column' of zeros

$pivot_col \leftarrow pivot_col + 1$

▷ Skip column

else

$A \leftarrow swap_rows(A, pivot_row, idx_max)$

▷ See later

for $i = pivot_row + 1, \dots, m$ **do**

▷ Zero entries under pivot

$\alpha \leftarrow \frac{A(i, pivot_col)}{A(pivot_row, pivot_col)}$

$A(i, pivot_col) \leftarrow 0$

▷ Avoid numerical errors

for $j = pivot_col + 1, \dots, n$ **do**

▷ Operate whole row

$A(i, j) \leftarrow A(i, j) - \alpha A(pivot_row, j)$

end for

end for

$pivot_row \leftarrow pivot_row + 1$

▷ Next pivot

$pivot_col \leftarrow pivot_col + 1$

end if

end while

Gaussian Elimination Algorithm (ii)

The Function *index_of_max_abs()* finds the index of the largest absolute value of *A* from *pivot_row* to the end of the *pivot_col* column.

Input: Matrix *A* of size $m \times n$, *pivot_row*, *pivot_col*

Output: Row number: *idx_max*

idx_max \leftarrow *pivot_row*

for $i = \text{pivot_row} + 1, \dots, m$ **do**

if $\text{abs}(A(i, \text{pivot_col})) > \text{abs}(A(\text{idx_max}, \text{pivot_col}))$ **then**

idx_max \leftarrow *i*

end if

end for

Notes:

- Check only under the pivot
- Any non-zero entry should work (doesn't need to be the absolute maximum)
- Function might exist in some programming languages

Gaussian Elimination Algorithm (iii)

The Function *swap_rows()* swaps two rows in the matrix A .

Input: Matrix A of size $m \times n$, $row1$, $row2$

Output: New matrix A of size $m \times n$,

for $i = 1, \dots, n$ **do**

$temp \leftarrow A(row1, i)$

$A(row1, i) \leftarrow A(row2, i)$

$A(row2, i) \leftarrow temp$

end for

- Need to swap the whole row? (not really, but simpler)
- Temporary copy of one entry
- Programming language arrays start on '0' or '1'? (applies to Gaussian algorithm too)

Interpretation of Linear Equations

In a linear system of equations $A\mathbf{x} = \mathbf{b}$ each equation (matrix row)

$$a_{i1}x_1 + a_{i2}x_2 + \cdots + a_{in}x_n = b_i$$

corresponds to a hyperplane in \mathbb{R}^n (line/plane in 2D/3D).

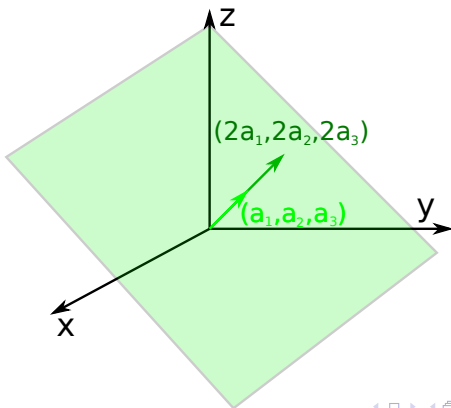
We define the vector $\mathbf{a}_i = [a_{i1}, a_{i2}, \dots, a_{in}]$ in \mathbb{R}^n and the equation becomes $\mathbf{a}_i \cdot \mathbf{x} = b_i$ (dot product of \mathbf{a}_i and \mathbf{x})

- If $b_i = 0$ the equation is: $\mathbf{a}_i \cdot \mathbf{x} = 0$.
 - Vectors \mathbf{a}_i and \mathbf{x} are perpendicular
 - Points ' \mathbf{x} ' in the plane when 'perpendicular' to \mathbf{a}_i
 - \mathbf{a}_i is called **normal vector**
- If $b_i \neq 0$ the equation is: $\mathbf{a}_i \cdot \mathbf{x} = b_i$.
 - Dot product is the projection of vector \mathbf{x} along direction \mathbf{a}_i
 - If $|\mathbf{a}_i| = 1$ then b_i is the distance from the plane to the origin $\mathbf{O} = [0, 0, \dots, 0]$ (perpendicular projection)
 - In general $b_i = |\mathbf{a}_i|d$ with ' d ' distance from the plane to the origin (if $d = 0$ previous case)

Geometrical Interpretation of the Scaling Rule (R^3)

Assuming $b_i = 0$, in Gaussian elimination multiplying a row by $\alpha \neq 0$ means scaling \mathbf{a}_i .

- Normal vector \mathbf{a}_i perpendicular to the plane
- Vector $\alpha \mathbf{a}_i$ has the same direction (remains perpendicular)
- Still 'true' for $b_i \neq 0$ and \mathbb{R}^n



Interpretation of Linear Equations (cont)

In a linear system of equations $A\mathbf{x} = \mathbf{b}$ considering two equation (matrix rows) simultaneously

$$a_{i1}x_1 + a_{i2}x_2 + \cdots + a_{in}x_n = b_i$$

$$a_{j1}x_1 + a_{j2}x_2 + \cdots + a_{jn}x_n = b_j$$

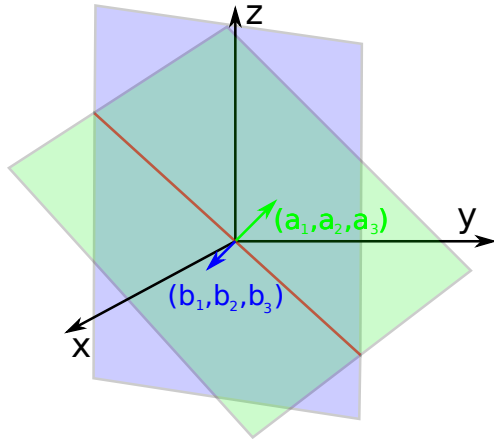
corresponds to the intersection of hyperplanes in \mathbb{R}^n (point/line in 2D/3D).

Defining the vectors as before \mathbf{a}_i and \mathbf{a}_j equation $\mathbf{a}_i \cdot \mathbf{x} = b_i$ can be replaced by $\mathbf{a}_i \cdot \mathbf{x} + \alpha \mathbf{a}_j \cdot \mathbf{x} = (\mathbf{a}_i + \alpha \mathbf{a}_j) \cdot \mathbf{x} = b_i + \alpha b_j$

- If $b_i = b_j = 0$ substitute $\mathbf{a}_i \cdot \mathbf{x} = 0$ by a hyperplane with normal $\mathbf{a}_i + \alpha \mathbf{a}_j$ (linear combination of \mathbf{a}_i and \mathbf{a}_j)
 - That hyperplane has the same intersection with $\mathbf{a}_j \cdot \mathbf{x} = 0$ as $\mathbf{a}_i \cdot \mathbf{x} = 0$
 - 'Rotation' of \mathbf{a}_i in 3D
- If $b_i \neq 0$ and $b_j \neq 0$ the hyperplane has the same intersection

Geometrical Interpretation of the Combination Rule (R^3)

- Intersection of two planes is a line (except when $\mathbf{a} = k\mathbf{b}$)
- Plane $(\mathbf{a} + \alpha\mathbf{b}) \cdot \mathbf{v} = 0$ is a plane 'rotated' along the intersection line



Gauss-Jordan Elimination

An extension of the Gauss Elimination algorithm that allows to obtain the inverse (A^{-1}) of a square matrix A (if $|A| \neq 0$).

Augmented matrix:

$$\tilde{A} = \left[\begin{array}{cccc|cccc} a_{11} & a_{12} & \cdots & a_{1n} & 1 & 0 & \cdots & 0 \\ a_{21} & a_{22} & \cdots & a_{2n} & 0 & 1 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots \\ a_{n1} & a_{n2} & \cdots & a_{nn} & 0 & 0 & \cdots & 1 \end{array} \right]$$

Augment the matrix A with the $n \times n$ identity matrix

- Make the submatrix A of \tilde{A} upper triangular
- Make the submatrix A of \tilde{A} diagonal
- Make the submatrix A of \tilde{A} the identity matrix
- The submatrix on the right is the inverse of A : A^{-1}

Gauss-Jordan Elimination Example

Using Gauss-Jordan elimination obtain the inverse of the matrix:

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

- Same rules apply:
 - Row swap
 - Row scaling
 - Row combination
- Steps:
 1. Build augmented matrix \tilde{A} adding the identity
 2. Convert to upper triangular
 3. Convert to diagonal (no row swapping)
 4. Convert to identity (only scaling)
- Maybe check if the inverse exists (product of diagonal entries of the upper triangular matrix)

Gauss-Jordan Elimination Example (Upper Triangular A)

Row 2 $+3 \times$ Row 1

$$\tilde{A} = \left[\begin{array}{ccc|ccc} -1 & 1 & 2 & 1 & 0 & 0 \\ 3 & -1 & 1 & 0 & 1 & 0 \\ -1 & 3 & 4 & 0 & 0 & 1 \end{array} \right] \rightarrow \left[\begin{array}{ccc|ccc} -1 & 1 & 2 & 1 & 0 & 0 \\ 0 & 2 & 7 & 3 & 1 & 0 \\ -1 & 3 & 4 & 0 & 0 & 1 \end{array} \right]$$

Gauss-Jordan Elimination Example (Upper Triangular A)

Row 3 $-$ Row 1

$$\begin{aligned}\tilde{A} &= \left[\begin{array}{ccc|ccc} -1 & 1 & 2 & 1 & 0 & 0 \\ 3 & -1 & 1 & 0 & 1 & 0 \\ -1 & 3 & 4 & 0 & 0 & 1 \end{array} \right] \rightarrow \left[\begin{array}{ccc|ccc} -1 & 1 & 2 & 1 & 0 & 0 \\ 0 & 2 & 7 & 3 & 1 & 0 \\ -1 & 3 & 4 & 0 & 0 & 1 \end{array} \right] \rightarrow \\ &\rightarrow \left[\begin{array}{ccc|ccc} -1 & 1 & 2 & 1 & 0 & 0 \\ 0 & 2 & 7 & 3 & 1 & 0 \\ 0 & 2 & 2 & -1 & 0 & 1 \end{array} \right]\end{aligned}$$

Gauss-Jordan Elimination Example (Upper Triangular A)

Row 3 $-$ Row 2

$$\begin{aligned}\tilde{A} = \left[\begin{array}{ccc|ccc} -1 & 1 & 2 & 1 & 0 & 0 \\ 3 & -1 & 1 & 0 & 1 & 0 \\ -1 & 3 & 4 & 0 & 0 & 1 \end{array} \right] &\rightarrow \left[\begin{array}{ccc|ccc} -1 & 1 & 2 & 1 & 0 & 0 \\ 0 & 2 & 7 & 3 & 1 & 0 \\ 0 & 2 & 2 & -1 & 0 & 1 \end{array} \right] \rightarrow \\ &\rightarrow \left[\begin{array}{ccc|ccc} -1 & 1 & 2 & 1 & 0 & 0 \\ 0 & 2 & 7 & 3 & 1 & 0 \\ 0 & 0 & -5 & -4 & -1 & 1 \end{array} \right]\end{aligned}$$

We reached the point where the submatrix A is upper triangular, but we have to continue to make it diagonal.

Gauss-Jordan Elimination Example (Diagonal A)

Row 2 $+\frac{7}{5}\times$ Row 3

$$\begin{aligned}\tilde{A} &= \left[\begin{array}{ccc|ccc} -1 & 1 & 2 & 1 & 0 & 0 \\ 3 & -1 & 1 & 0 & 1 & 0 \\ -1 & 3 & 4 & 0 & 0 & 1 \end{array} \right] \rightarrow \left[\begin{array}{ccc|ccc} -1 & 1 & 2 & 1 & 0 & 0 \\ 0 & 2 & 7 & 3 & 1 & 0 \\ 0 & 0 & -5 & -4 & -1 & 1 \end{array} \right] \rightarrow \\ &\rightarrow \left[\begin{array}{ccc|ccc} -1 & 1 & 2 & 1 & 0 & 0 \\ 0 & 2 & 0 & -\frac{13}{5} & -\frac{2}{5} & \frac{7}{5} \\ 0 & 0 & -5 & -4 & -1 & 1 \end{array} \right]\end{aligned}$$

Gauss-Jordan Elimination Example (Diagonal A)

Row 1 $-\frac{1}{2}$ Row 2

$$\begin{aligned}\tilde{A} = \left[\begin{array}{ccc|ccc} -1 & 1 & 2 & 1 & 0 & 0 \\ 3 & -1 & 1 & 0 & 1 & 0 \\ -1 & 3 & 4 & 0 & 0 & 1 \end{array} \right] &\rightarrow \left[\begin{array}{ccc|ccc} -1 & 1 & 2 & 1 & 0 & 0 \\ 0 & 2 & 0 & -\frac{13}{5} & -\frac{2}{5} & \frac{7}{5} \\ 0 & 0 & -5 & -4 & -1 & 1 \end{array} \right] \rightarrow \\ &\rightarrow \left[\begin{array}{ccc|ccc} -1 & 0 & 2 & \frac{23}{10} & \frac{2}{10} & -\frac{7}{10} \\ 0 & 2 & 0 & -\frac{13}{5} & -\frac{2}{5} & \frac{7}{5} \\ 0 & 0 & -5 & -4 & -1 & 1 \end{array} \right]\end{aligned}$$

Gauss-Jordan Elimination Example (Diagonal A)

Row 1 + $\frac{2}{5}$ Row 3

$$\tilde{A} = \left[\begin{array}{ccc|ccc} -1 & 1 & 2 & 1 & 0 & 0 \\ 3 & -1 & 1 & 0 & 1 & 0 \\ -1 & 3 & 4 & 0 & 0 & 1 \end{array} \right] \rightarrow \left[\begin{array}{ccc|ccc} -1 & 0 & 2 & \frac{23}{10} & \frac{1}{5} & -\frac{7}{10} \\ 0 & 2 & 0 & -\frac{13}{5} & -\frac{2}{5} & \frac{7}{5} \\ 0 & 0 & -5 & -4 & -1 & 1 \end{array} \right]$$
$$\rightarrow \left[\begin{array}{ccc|ccc} -1 & 0 & 0 & \frac{7}{10} & -\frac{1}{5} & -\frac{3}{10} \\ 0 & 2 & 0 & -\frac{13}{5} & -\frac{2}{5} & \frac{7}{5} \\ 0 & 0 & -5 & -4 & -1 & 1 \end{array} \right]$$

Now the submatrix A is diagonal, but it should be the identity matrix.

Gauss-Jordan Elimination Example (Identity A)

Row 1 $\rightarrow -1 \times$ Row 1

$$\begin{aligned}\tilde{A} = \left[\begin{array}{ccc|ccc} -1 & 1 & 2 & 1 & 0 & 0 \\ 3 & -1 & 1 & 0 & 1 & 0 \\ -1 & 3 & 4 & 0 & 0 & 1 \end{array} \right] &\rightarrow \left[\begin{array}{ccc|ccc} -1 & 0 & 0 & \frac{7}{10} & -\frac{1}{5} & -\frac{3}{10} \\ 0 & 2 & 0 & -\frac{13}{5} & -\frac{2}{5} & \frac{7}{5} \\ 0 & 0 & -5 & -4 & -1 & 1 \end{array} \right] \\ &\rightarrow \left[\begin{array}{ccc|ccc} 1 & 0 & 0 & -\frac{7}{10} & \frac{1}{5} & \frac{3}{10} \\ 0 & 2 & 0 & -\frac{13}{5} & -\frac{2}{5} & \frac{7}{5} \\ 0 & 0 & -5 & -4 & -1 & 1 \end{array} \right]\end{aligned}$$

Gauss-Jordan Elimination Example (Identity A)

Row 2 $\rightarrow \frac{1}{2} \times$ Row 2

$$\begin{aligned}\tilde{A} &= \left[\begin{array}{ccc|ccc} -1 & 1 & 2 & 1 & 0 & 0 \\ 3 & -1 & 1 & 0 & 1 & 0 \\ -1 & 3 & 4 & 0 & 0 & 1 \end{array} \right] \rightarrow \left[\begin{array}{ccc|ccc} 1 & 0 & 0 & -\frac{7}{10} & \frac{1}{5} & \frac{3}{10} \\ 0 & 2 & 0 & -\frac{13}{5} & -\frac{2}{5} & \frac{7}{5} \\ 0 & 0 & -5 & -4 & -1 & 1 \end{array} \right] \\ &\rightarrow \left[\begin{array}{ccc|ccc} 1 & 0 & 0 & -\frac{7}{10} & \frac{1}{5} & \frac{3}{10} \\ 0 & 1 & 0 & -\frac{13}{10} & -\frac{2}{10} & \frac{7}{10} \\ 0 & 0 & -5 & -4 & -1 & 1 \end{array} \right]\end{aligned}$$

Gauss-Jordan Elimination Example (Identity A)

Row 3 $\rightarrow -\frac{1}{5} \times \text{Row 3}$

$$\begin{aligned}\tilde{A} &= \left[\begin{array}{ccc|ccc} -1 & 1 & 2 & 1 & 0 & 0 \\ 3 & -1 & 1 & 0 & 1 & 0 \\ -1 & 3 & 4 & 0 & 0 & 1 \end{array} \right] \rightarrow \left[\begin{array}{ccc|ccc} 1 & 0 & 0 & -\frac{7}{10} & \frac{1}{5} & \frac{3}{10} \\ 0 & 1 & 0 & -\frac{13}{10} & -\frac{1}{5} & \frac{7}{10} \\ 0 & 0 & -5 & -4 & -1 & 1 \end{array} \right] \\ &\rightarrow \left[\begin{array}{ccc|ccc} 1 & 0 & 0 & -\frac{7}{10} & \frac{1}{5} & \frac{3}{10} \\ 0 & 1 & 0 & -\frac{13}{10} & -\frac{1}{5} & \frac{7}{10} \\ 0 & 0 & 1 & \frac{4}{5} & \frac{1}{5} & -\frac{1}{5} \end{array} \right]\end{aligned}$$

Gauss-Jordan Elimination Example: A^{-1}

$$\tilde{A} = \left[\begin{array}{ccc|ccc} -1 & 1 & 2 & 1 & 0 & 0 \\ 3 & -1 & 1 & 0 & 1 & 0 \\ -1 & 3 & 4 & 0 & 0 & 1 \end{array} \right] \rightarrow \left[\begin{array}{ccc|ccc} 1 & 0 & 0 & -\frac{7}{10} & \frac{1}{5} & \frac{3}{10} \\ 0 & 1 & 0 & -\frac{13}{10} & -\frac{1}{5} & \frac{7}{10} \\ 0 & 0 & 1 & \frac{4}{5} & \frac{1}{5} & -\frac{1}{5} \end{array} \right]$$

Therefore the inverse of A is:

$$A^{-1} = \left[\begin{array}{ccc} -\frac{7}{10} & \frac{1}{5} & \frac{3}{10} \\ -\frac{13}{10} & -\frac{1}{5} & \frac{7}{10} \\ \frac{4}{5} & \frac{1}{5} & -\frac{1}{5} \end{array} \right]$$

Gauss-Jordan Elimination Algorithm

Apply Gauss elimination part to \tilde{A} (size $n \times (2n)$) and then:

```
pivot  $\leftarrow n$                                 ▷ Initial pivot
while pivot  $\geq 1$  do
  if  $A(\textit{pivot}, \textit{pivot}) = 0$  then             ▷ Pivot is zero
    Error(Singular Matrix)
  else
    for  $i = \textit{pivot} - 1, \dots, 1$  do           ▷ Zero entries over pivot
       $\alpha \leftarrow \frac{A(i, \textit{pivot})}{A(\textit{pivot}, \textit{pivot})}$ 
       $A(i, \textit{pivot}) \leftarrow 0$                 ▷ Avoid numerical errors
      for  $j = \textit{pivot} + 1, \dots, 2n$  do         ▷ Operate whole row
         $A(i, j) \leftarrow A(i, j) - \alpha A(\textit{pivot}, j)$ 
      end for
    end for
     $\alpha = \frac{1}{A(\textit{pivot}, \textit{pivot})}$ 
    for  $j = 1, \dots, 2n$  do                   ▷ Scale row for identity
       $A(\textit{pivot}, j) \leftarrow \alpha A(\textit{pivot}, j)$ 
    end for
    pivot  $\leftarrow \textit{pivot} - 1$                 ▷ Next pivot
  end if
end while
```

Notes on Gauss-Jordan Elimination Algorithm

- **First apply Gauss elimination** to \tilde{A} (see corresponding algorithm)
- Last slide **only** second part of the algorithm
- Inverse A^{-1} must be taken from \tilde{A} (submatrix)
- If one diagonal entry in the upper triangular matrix is zero the original matrix A has no inverse (singular)

Back to Linear Systems of Equations

A linear system of ' n ' equations in ' n ' variables $A\mathbf{x} = \mathbf{b}$ can be solved using Gauss-Jordan elimination to obtain A^{-1} and the solution is then $\mathbf{x} = A^{-1}\mathbf{b}$ (matrix vector product).

Typically if the system $A\mathbf{x} = \mathbf{b}$ has ' m ' equations in ' n ' variables one can do:

- If $m > n$ (e.g. linear regression) multiplying both sides by A^T

$$A^T A \mathbf{x} = A^T \mathbf{b}$$

where $A^T A$ is an $n \times n$ matrix (hopefully not singular), and multiplying both sides by $(A^T A)^{-1}$ we get:

$$\mathbf{x} = (A^T A)^{-1} A^T \mathbf{b}$$

The matrix $A^\dagger = (A^T A)^{-1} A^T$ is called **pseudo-inverse** of A .

- If $m < n$ there is a similar "trick".

Summary

- Gaussian elimination algorithm
- Geometrical interpretation of Gaussian elimination operations in 3D
- Gauss-Jordan elimination for inverse calculation
- Inverse calculation algorithm