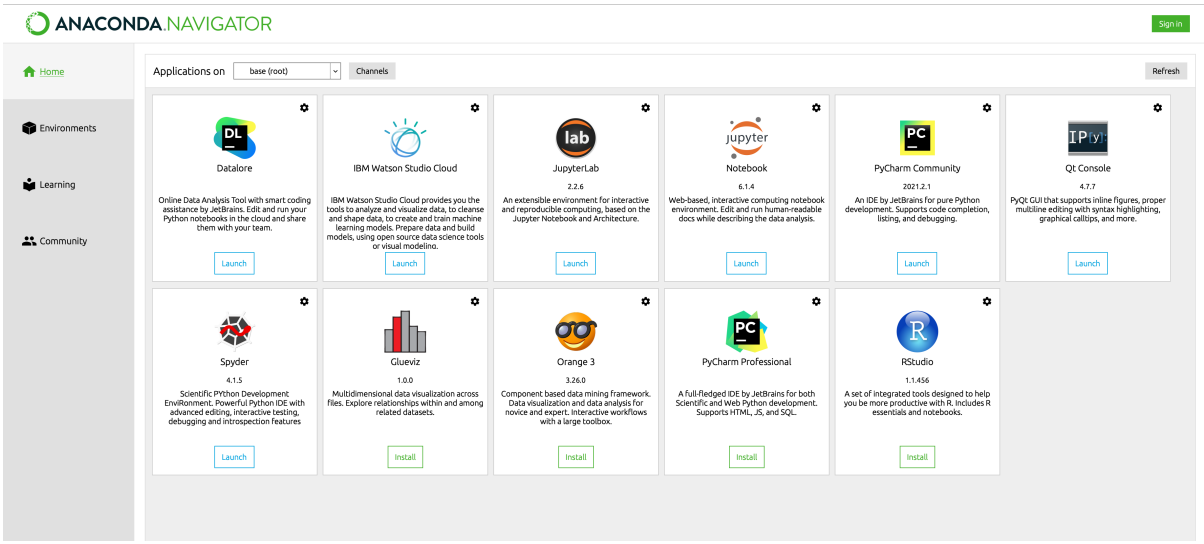
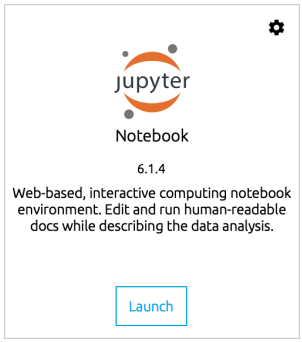
Basic matrix operations



Download Anaconda:

https://www.anaconda.com/products/individual



Basic matrix operations

jupyter Untitled24 Last Checkpoint: 2 minutes ago (autosaved)

Logout

File Edit View Insert Cell Kernel Widgets Help

Trusted | Python 3

Run Code

In [1]:
```
import sympy as sp
```

In [2]:
```
A = sp.Matrix([[1,1,-1]]);A
```

Out[2]: $\begin{bmatrix} 1 & 1 & -1 \end{bmatrix}$

In [3]:
```
B = sp.Matrix([[2],[7],[8]]);B
```

Out[3]: $\begin{bmatrix} 2 \\ 7 \\ 8 \end{bmatrix}$

In [4]:
```
C = sp.Matrix([[1,2],[3,4]]);C
```

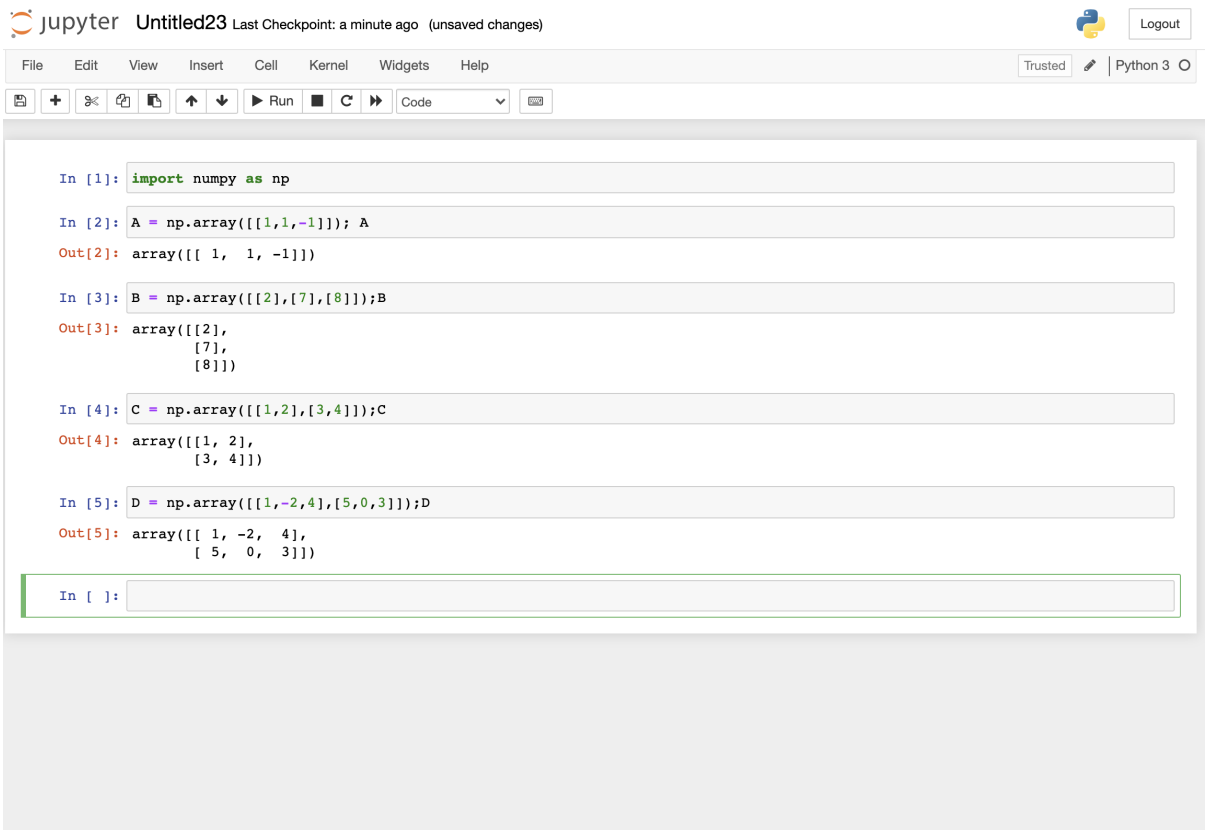
Out[4]: $\begin{bmatrix} 1 & 2 \\ 3 & 4 \end{bmatrix}$

In [5]:
```
D = sp.Matrix([[1,-2,4],[5,0,3]]);D
```

Out[5]: $\begin{bmatrix} 1 & -2 & 4 \\ 5 & 0 & 3 \end{bmatrix}$

In [ ]:

Basic matrix operations



Basic matrix operations

Why SymPy?

There are many computer algebra systems out there. This Wikipedia article lists many of them. What makes SymPy a better choice than the alternatives?

First off, SymPy is completely free. It is open source, and licensed under the liberal BSD license, so you can modify the source code and even sell it if you want to. This contrasts with popular commercial systems like Maple or Mathematica that cost hundreds of dollars in licenses.

Second, SymPy uses Python. Most computer algebra systems invent their own language. Not SymPy. SymPy is written entirely in Python, and is executed entirely in Python. This means that if you already know Python, it is much easier to get started with SymPy, because you already know the syntax (and if you don't know Python, it is really easy to learn). We already know that Python is a well-designed, battle-tested language. The SymPy developers are confident in their abilities in writing mathematical software, but programming language design is a completely different thing. By reusing an existing language, we are able to focus on those things that matter: the mathematics.

What is NumPy?

NumPy is the fundamental package for scientific computing in Python. It is a Python library that provides a multidimensional array object, various derived objects (such as masked arrays and matrices), and an assortment of routines for fast operations on arrays, including mathematical, logical, shape manipulation, sorting, selecting, I/O, discrete Fourier transforms, basic linear algebra, basic statistical operations, random simulation and much more.

Basic matrix operations

[1]:
```
# Import SymPy package
import sympy as sp
```

[2]:
```
# Define matrices (SymPy package)
A = sp.Matrix([[1,1,-1]]); A
```

[2]: $\begin{bmatrix}1 & 1 & -1\end{bmatrix}$

[3]:
```
B = sp.Matrix([[2],[7],[8]]); B
```

[3]: $\begin{bmatrix}2\\7\\8\end{bmatrix}$

[4]:
```
C = sp.Matrix([[1,2],[3,4]]); C
```

[4]: $\begin{bmatrix}1 & 2\\3 & 4\end{bmatrix}$

[5]:
```
D = sp.Matrix([[1,-2,4],[5,0,3]]); D
```

[5]: $\begin{bmatrix}1 & -2 & 4\\5 & 0 & 3\end{bmatrix}$

[6]:
```
# Size of matrices
A.shape,B.shape,C.shape,D.shape
```

[6]:
```
((1, 3), (3, 1), (2, 2), (2, 3))
```

[7]:
```
# Import NumPy package
import numpy as np
```

[8]:
```
# Define matrices (NumPy package)
A = np.array([[1,1,-1]]); A
```

[8]:
```
array([[ 1,  1, -1]])
```

[9]:
```
B = np.array([[2],[7],[8]]); B
```

[9]:
```
array([[2],
       [7],
       [8]])
```

[10]:
```
C = np.array([[1,2],[3,4]]); C
```

[10]:
```
array([[1, 2],
       [3, 4]])
```

[11]:
```
D = np.array([[1,-2,4],[5,0,3]]); D
```

[11]:
```
array([[ 1, -2,  4],
       [ 5,  0,  3]])
```

[12]:
```
# Size of matrices
A.shape,B.shape,C.shape,D.shape
```

[12]:
```
((1, 3), (3, 1), (2, 2), (2, 3))
```

Basic matrix operations

Addition, Subtraction and Scalar Multiplication

For any two $m \times n$ matrices $A = [a_{ij}]$ and $B = [b_{ij}]$ and any scalar c,

1. $A + B$ is the $m \times n$ matrix with $a_{ij} + b_{ij}$ as its (i, j)th entry.
2. $A - B$ is the $m \times n$ matrix with $a_{ij} - b_{ij}$ as its (i, j)th entry.
3. cA is the $m \times n$ matrix with ca_{ij} as its (i, j)th entry.

Properties for Addition and Scalar Multiplication

Let A, B and C be $m \times n$ matrices and let c and d be scalars.

1. $A + B = B + A$,
2. $(A + B) + C = A + (B + C)$,
3. $c(A + B) = cA + cB$,
4. $(c + d)A = cA + dA$,
5. $c(dA) = (cd)A$,
6. $A + \mathbf{0} = A$. Here, $\mathbf{0} = \begin{bmatrix}0 & \cdots & 0\\ \vdots & \ddots & \vdots\\ 0 & \cdots & 0\end{bmatrix}$ is the $m \times n$ **zero matrix**.

Basic matrix operations

Example 1 For $A = \begin{bmatrix} 4 & 0 & 5 \\ -1 & 3 & 2 \end{bmatrix}$ and $B = \begin{bmatrix} 1 & 1 & 1 \\ 3 & 5 & 7 \end{bmatrix}$,

[1]:
```
import sympy as sp
```

[2]:
```
A = sp.Matrix([[4,0,5],[-1,3,2]]); A
```

[2]: $\begin{bmatrix} 4 & 0 & 5 \\ -1 & 3 & 2 \end{bmatrix}$

[3]:
```
B = sp.Matrix([[1,1,1],[3,5,7]]); B
```

[3]: $\begin{bmatrix} 1 & 1 & 1 \\ 3 & 5 & 7 \end{bmatrix}$

[4]:
```
A+B
```

[4]: $\begin{bmatrix} 5 & 1 & 6 \\ 2 & 8 & 9 \end{bmatrix}$

[5]:
```
A-B
```

[5]: $\begin{bmatrix} 3 & -1 & 4 \\ -4 & -2 & -5 \end{bmatrix}$

[6]:
```
2*B
```

[6]: $\begin{bmatrix} 2 & 2 & 2 \\ 6 & 10 & 14 \end{bmatrix}$

$$
\begin{aligned}
A + B &= \begin{bmatrix} 4 & 0 & 5 \\ -1 & 3 & 2 \end{bmatrix} + \begin{bmatrix} 1 & 1 & 1 \\ 3 & 5 & 7 \end{bmatrix} = \begin{bmatrix} 5 & 1 & 6 \\ 2 & 8 & 9 \end{bmatrix} \\
A - B &= \begin{bmatrix} 4 & 0 & 5 \\ -1 & 3 & 2 \end{bmatrix} - \begin{bmatrix} 1 & 1 & 1 \\ 3 & 5 & 7 \end{bmatrix} = \begin{bmatrix} 3 & -1 & 4 \\ -4 & -2 & -5 \end{bmatrix} \\
2B &= 2 \cdot \begin{bmatrix} 1 & 1 & 1 \\ 3 & 5 & 7 \end{bmatrix} = \begin{bmatrix} 2 & 2 & 2 \\ 6 & 10 & 14 \end{bmatrix}
\end{aligned}
$$

Basic matrix operations

Matrix Multiplication

For any $m \times n$ matrix $A = [a_{ij}]$ and $n \times p$ matrix $B = [b_{ij}]$, the product of A and B, denoted by AB, is the $m \times p$ matrix with $\sum_{k=1}^{n} a_{ik} b_{kj}$ as the (i, j)th entry. Equivalently, if $C = [c_{ij}] = AB$, then

$$
c_{ij} = \sum_{k=1}^{n} a_{ik} b_{kj} = a_{i1} b_{1j} + a_{i2} b_{2j} + \cdots + a_{in} b_{nj}.
$$

Example 2 For $A = \begin{bmatrix} 2 & 3 \end{bmatrix}$, $B = \begin{bmatrix} 4 \\ -1 \end{bmatrix}$, $C = \begin{bmatrix} 2 & 3 & 4 \end{bmatrix}$, and $D = \begin{bmatrix} 4 \\ -1 \\ 1 \end{bmatrix}$,

$$
AB = \overbrace{\begin{bmatrix} 2 & 3 \end{bmatrix}}^{1\times 2} \overbrace{\begin{bmatrix} 4 \\ -1 \end{bmatrix}}^{2\times 1} = \overbrace{\begin{bmatrix} 2 \cdot 4 + 3 \cdot (-1) \end{bmatrix}}^{1\times 1} = \begin{bmatrix} 5 \end{bmatrix}
$$

$$
CD = \underbrace{\begin{bmatrix} 2 & 3 & 4 \end{bmatrix}}_{1\times 3} \underbrace{\begin{bmatrix} 4 \\ -1 \\ 1 \end{bmatrix}}_{3\times 1} = \underbrace{\begin{bmatrix} 2 \cdot 4 + 3 \cdot (-1) + 4 \cdot 1 \end{bmatrix}}_{1\times 1} = \begin{bmatrix} 9 \end{bmatrix}
$$

Basic matrix operations

Example 3 For $A = \begin{bmatrix} 2 & 3 \\ 1 & -5 \end{bmatrix}$, $B = \begin{bmatrix} 4 & 3 & 6 \\ 1 & -2 & 3 \end{bmatrix}$, and $C = \begin{bmatrix} 4 & 3 \\ 1 & -2 \end{bmatrix}$.

$$
\begin{aligned}
AB &= \underbrace{\begin{bmatrix} 2 & 3 \\ 1 & -5 \end{bmatrix}}_{2\times 2} \underbrace{\begin{bmatrix} 4 & 3 & 6 \\ 1 & -2 & 3 \end{bmatrix}}_{2\times 3} \\
&= \underbrace{\begin{bmatrix} 2\cdot 4 + 3\cdot 1 & 2\cdot 3 + 3\cdot(-2) & 2\cdot 6 + 3\cdot 3 \\ 1\cdot 4 + (-5)\cdot 1 & 1\cdot 3 + (-5)\cdot(-2) & 1\cdot 6 + (-5)\cdot 3 \end{bmatrix}}_{2\times 3} \\
&= \begin{bmatrix} 11 & 0 & 21 \\ -1 & 13 & -9 \end{bmatrix}.
\end{aligned}
$$

$BA = \underbrace{\begin{bmatrix} 4 & 3 & 6 \\ 1 & -2 & 3 \end{bmatrix}}_{2\times 3} \underbrace{\begin{bmatrix} 2 & 3 \\ 1 & -5 \end{bmatrix}}_{2\times 2}$ is NOT defined.

$$
\begin{aligned}
AC &= \begin{bmatrix} 2 & 3 \\ 1 & -5 \end{bmatrix} \begin{bmatrix} 4 & -3 \\ 1 & 2 \end{bmatrix} = \begin{bmatrix} 11 & 0 \\ -1 & -13 \end{bmatrix} \\
CA &= \begin{bmatrix} 4 & -3 \\ 1 & 2 \end{bmatrix} \begin{bmatrix} 2 & 3 \\ 1 & -5 \end{bmatrix} = \begin{bmatrix} 5 & 27 \\ 4 & -7 \end{bmatrix} \neq AC
\end{aligned}
$$

[1]:
```
import sympy as sp
A = sp.Matrix([[2,3],[1,-5]])
B = sp.Matrix([[4,3,6],[1,-2,3]])
C = sp.Matrix([[4,-3],[1,2]])
```

[2]:
```
A
```

[2]: $\begin{bmatrix} 2 & 3 \\ 1 & -5 \end{bmatrix}$

[3]:
```
B
```

[3]: $\begin{bmatrix} 4 & 3 & 6 \\ 1 & -2 & 3 \end{bmatrix}$

[4]:
```
C
```

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

[5]:
```
# Matrix Multiplication
A@B
```

[5]: $\begin{bmatrix} 11 & 0 & 21 \\ -1 & 13 & -9 \end{bmatrix}$

Basic matrix operations

[6]:
```
B@A
```

```
---------------------------------------------------------------------------
ShapeError                                Traceback (most recent call last)
/var/folders/wh/1v8jgf9d5gbfykybbcypgbt40000gn/T/ipykernel_25582/2247667524.py in <module>
----> 1 B@A

/opt/anaconda3/lib/python3.9/site-packages/sympy/core/decorators.py in binary_op_wrapper(sel
↪other)
    134                     if f is not None:
    135                         return f(self)
--> 136             return func(self, other)
    137         return binary_op_wrapper
    138     return priority_decorator

/opt/anaconda3/lib/python3.9/site-packages/sympy/matrices/common.py in __matmul__(self, othe
   2727             return NotImplemented
   2728
-> 2729         return self.__mul__(other)
   2730
   2731     def __mod__(self, other):

/opt/anaconda3/lib/python3.9/site-packages/sympy/core/decorators.py in binary_op_wrapper(sel
↪other)
    134                     if f is not None:
    135                         return f(self)
--> 136             return func(self, other)
    137         return binary_op_wrapper
    138     return priority_decorator
ShapeError: Matrix size mismatch: (2, 3) * (2, 2).
```

Basic matrix operations

Power of A

If A is an $n \times n$ matrix and k is a positive integer, then

$$A^k = \underbrace{A \cdot A \cdot A \cdots A}_{k \text{ copies}}.$$

Transpose of A

Given an $m \times n$ matrix $A = [a_{ij}]$, the **transpose** of A is the $n \times m$ matrix, denoted by A^T, whose columns are formed from the corresponding rows of A. That is, the (i, j)th entry of A^T is equal to a_{ji}.

Identity Matrix

The $n \times n$ **identity matrix** I_n has the (i, j)th entry $\delta_{ij} = \begin{cases} 1 & i = j \\ 0 & i \neq j \end{cases}$. That is, $I_n = \underbrace{\begin{bmatrix} 1 & 0 & \cdots & 0 \\ 0 & 1 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 1 \end{bmatrix}}_{n \times n}$.

```
[7]: # Power
     A**3
```

[7]: $\begin{bmatrix} 5 & 66 \\ 22 & -149 \end{bmatrix}$

```
[8]: A**10
```

[8]: $\begin{bmatrix} 1110172 & -8172999 \\ -2724333 & 20180503 \end{bmatrix}$

```
[9]: # Transpose
     A.T
```

[9]: $\begin{bmatrix} 2 & 1 \\ 3 & -5 \end{bmatrix}$

```
[10]: B.T
```

[10]: $\begin{bmatrix} 4 & 1 \\ 3 & -2 \\ 6 & 3 \end{bmatrix}$

```
[11]: # Identity matrix
      sp.eye(4)
```

[11]: $\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$

Basic matrix operations

Properties for Transpose

Let A and B denote matrices whose sizes are appropriate for the following sums and products and c be any scalar.

1. $(A^T)^T = A$
2. $(A + B)^T = A^T + B^T$
3. $(cA)^T = cA^T$
4. $(AB)^T = B^T A^T$

Example 4 For $A = \begin{bmatrix} 2 & 3 \\ 1 & -5 \end{bmatrix}$ and $C = \begin{bmatrix} 4 & 3 \\ 1 & -2 \end{bmatrix}$,

$$AC = \begin{bmatrix} 2 & 3 \\ 1 & -5 \end{bmatrix} \begin{bmatrix} 4 & -3 \\ 1 & 2 \end{bmatrix} = \begin{bmatrix} 11 & 0 \\ -1 & -13 \end{bmatrix}$$

$$A^T C^T = \begin{bmatrix} 2 & 1 \\ 3 & -5 \end{bmatrix} \begin{bmatrix} 4 & 1 \\ -3 & 2 \end{bmatrix} = \begin{bmatrix} 5 & 4 \\ 27 & -7 \end{bmatrix}$$

$$C^T A^T = \begin{bmatrix} 4 & 1 \\ -3 & 2 \end{bmatrix} \begin{bmatrix} 2 & 1 \\ 3 & -5 \end{bmatrix} = \begin{bmatrix} 11 & -1 \\ 0 & -13 \end{bmatrix}$$

```
[12]: A@C
```

[12]: $\begin{bmatrix} 11 & 0 \\ -1 & -13 \end{bmatrix}$

```
[13]: A.T@C.T
```

[13]: $\begin{bmatrix} 5 & 4 \\ 27 & -7 \end{bmatrix}$

```
[14]: C.T@A.T
```

[14]: $\begin{bmatrix} 11 & -1 \\ 0 & -13 \end{bmatrix}$

Determinant of 2×2 / 3×3 matrices

The **determinant** of a 2×2 matrix is defined and denoted by

$$\det A = |A| = \det\left(\begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{bmatrix}\right) = \begin{vmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{vmatrix} = a_{11}a_{22} - a_{12}a_{21}.$$

The **determinant** of a 3×3 matrix is defined and denoted by

$$\begin{aligned} \det(A) &= \begin{vmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{vmatrix} = a_{11}a_{22}a_{33} + a_{31}a_{12}a_{23} + a_{21}a_{32}a_{13} \\ &\quad - a_{31}a_{22}a_{13} - a_{21}a_{12}a_{33} - a_{11}a_{23}a_{32}. \end{aligned}$$

Cofactor

Let A be an $n \times n$ matrix.

1. Denote by A_{ij} the $(n-1) \times (n-1)$ matrix obtained from A by deleting its ith row and jth column.
2. The (i,j)-**th cofactor** C_{ij} is defined by

$$C_{ij} = (-1)^{i+j} \det A_{ij}.$$

$$A = \begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{bmatrix} \quad A_{11} = \begin{bmatrix} a_{22} & a_{23} \\ a_{32} & a_{33} \end{bmatrix} \quad C_{11} = (-1)^{1+1} \begin{vmatrix} a_{22} & a_{23} \\ a_{32} & a_{33} \end{vmatrix}$$

$$A = \begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{bmatrix} \quad A_{12} = \begin{bmatrix} a_{21} & a_{23} \\ a_{31} & a_{33} \end{bmatrix} \quad C_{12} = (-1)^{1+2} \begin{vmatrix} a_{21} & a_{23} \\ a_{31} & a_{33} \end{vmatrix}$$

$$A = \begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{bmatrix} \quad A_{13} = \begin{bmatrix} a_{21} & a_{22} \\ a_{31} & a_{32} \end{bmatrix} \quad C_{13} = (-1)^{1+3} \begin{vmatrix} a_{21} & a_{22} \\ a_{31} & a_{32} \end{vmatrix}$$

$$A = \begin{bmatrix} a_{11} & a_{12} & a_{13} & a_{14} \\ a_{21} & a_{22} & a_{23} & a_{24} \\ a_{31} & a_{32} & a_{33} & a_{34} \\ a_{41} & a_{42} & a_{43} & a_{44} \end{bmatrix} \quad A_{11} = \begin{bmatrix} a_{22} & a_{23} & a_{24} \\ a_{32} & a_{33} & a_{34} \\ a_{42} & a_{43} & a_{44} \end{bmatrix} \quad C_{11} = (-1)^{1+1} \begin{vmatrix} a_{22} & a_{23} & a_{24} \\ a_{32} & a_{33} & a_{34} \\ a_{42} & a_{43} & a_{44} \end{vmatrix}$$

$$A = \begin{bmatrix} a_{11} & a_{12} & a_{13} & a_{14} \\ a_{21} & a_{22} & a_{23} & a_{24} \\ a_{31} & a_{32} & a_{33} & a_{34} \\ a_{41} & a_{42} & a_{43} & a_{44} \end{bmatrix} \quad A_{23} = \begin{bmatrix} a_{11} & a_{12} & a_{14} \\ a_{31} & a_{32} & a_{34} \\ a_{41} & a_{42} & a_{44} \end{bmatrix} \quad C_{23} = (-1)^{2+3} \begin{vmatrix} a_{11} & a_{12} & a_{14} \\ a_{31} & a_{32} & a_{34} \\ a_{41} & a_{42} & a_{44} \end{vmatrix}$$

$$A = \begin{bmatrix} a_{11} & a_{12} & a_{13} & a_{14} \\ a_{21} & a_{22} & a_{23} & a_{24} \\ a_{31} & a_{32} & a_{33} & a_{34} \\ a_{41} & a_{42} & a_{43} & a_{44} \end{bmatrix} \quad A_{34} = \begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{41} & a_{42} & a_{43} \end{bmatrix} \quad C_{34} = (-1)^{3+4} \begin{vmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{41} & a_{42} & a_{43} \end{vmatrix}$$

Determinant

For any $n \geq 2$, the **determinant** of an $n \times n$ matrix $A = [a_{ij}]$ is defined by

$$\det A = a_{11}C_{11} + a_{12}C_{12} + a_{13}C_{13} + a_{14}C_{14} + \cdots + a_{1n}C_{1n} = \sum_{j=1}^{n} a_{1j}C_{1j}.$$

It can also be computed by a cofactor expansion across any row or down any column.

- The expansion across the ith row is

$$\det A = a_{i1}C_{i1} + a_{i2}C_{i2} + a_{i3}C_{i3} + a_{i4}C_{i4} + \cdots + a_{in}C_{in} = \sum_{j=1}^{n} a_{ij}C_{ij}.$$

- The expansion down the jth column is

$$\det A = a_{1j}C_{1j} + a_{2j}C_{2j} + a_{3j}C_{3j} + a_{4j}C_{4j} + \cdots + a_{nj}C_{nj} = \sum_{i=1}^{n} a_{ij}C_{ij}.$$

Properties of determinant

For any $n \times n$ matrices A and B and scalar c,

$$\det A^T = \det A, \quad \det(cA) = c^n \det A \quad \text{and} \quad \det(AB) = \det A \cdot \det B.$$

Example 5 Suppose $A = \begin{bmatrix} 1 & -2 & 5 & 0 \\ 2 & 0 & 4 & -1 \\ 3 & 1 & 0 & 7 \\ 0 & 4 & -2 & 0 \end{bmatrix}$.

$$\begin{aligned}
& \det A \\
= \; & a_{11}C_{11} + a_{12}C_{12} + a_{13}C_{13} + a_{14}C_{14} \\
= \; & 1 \cdot \begin{vmatrix} 0 & 4 & -1 \\ 1 & 0 & 7 \\ 4 & -2 & 0 \end{vmatrix} - (-2) \cdot \begin{vmatrix} 2 & 4 & -1 \\ 3 & 0 & 7 \\ 0 & -2 & 0 \end{vmatrix} + 5 \cdot \begin{vmatrix} 2 & 0 & -1 \\ 3 & 1 & 7 \\ 0 & 4 & 0 \end{vmatrix} - 0 \cdot \begin{vmatrix} 2 & 0 & 4 \\ 3 & 1 & 0 \\ 0 & 4 & -2 \end{vmatrix} \\
= \; & 1 \cdot (114) + 2 \cdot (34) + 5 \cdot (-68) - 0 = -158 \\
= \; & a_{41}C_{41} + a_{42}C_{42} + a_{43}C_{43} + a_{44}C_{44} \\
= \; & -0 \cdot \begin{vmatrix} -2 & 5 & 0 \\ 0 & 4 & -1 \\ 1 & 0 & 7 \end{vmatrix} + 4 \cdot \begin{vmatrix} 1 & 5 & 0 \\ 2 & 4 & -1 \\ 3 & 0 & 7 \end{vmatrix} - (-2) \cdot \begin{vmatrix} 1 & -2 & 0 \\ 2 & 0 & -1 \\ 3 & 1 & 7 \end{vmatrix} + 0 \cdot \begin{vmatrix} 1 & -2 & 5 \\ 2 & 0 & 4 \\ 3 & 1 & 0 \end{vmatrix} \\
= \; & 0 + 4 \cdot (-57) + 2 \cdot (35) + 0 = -158.
\end{aligned}$$

Adjoint

For any $n \times n$ matrix A, the matrix of cofactors

$$\begin{bmatrix} C_{11} & C_{21} & \cdots & C_{n1} \\ C_{12} & C_{22} & \cdots & C_{n2} \\ \vdots & \vdots & \ddots & \vdots \\ C_{1n} & C_{2n} & \cdots & C_{nn} \end{bmatrix}.$$

is called the **adjoint (adjugate)** of A, denoted by $\operatorname{adj} A$.

$$\operatorname{adj} A = \underbrace{\begin{bmatrix} C_{11} & C_{21} \\ C_{12} & C_{22} \end{bmatrix}}_{(2\times 2 \text{ case})} \quad \underbrace{\begin{bmatrix} C_{11} & C_{21} & C_{31} \\ C_{12} & C_{22} & C_{32} \\ C_{13} & C_{23} & C_{33} \end{bmatrix}}_{(3\times 3 \text{ case})} \quad \underbrace{\begin{bmatrix} C_{11} & C_{21} & C_{31} & C_{41} \\ C_{12} & C_{22} & C_{32} & C_{42} \\ C_{13} & C_{23} & C_{33} & C_{43} \\ C_{14} & C_{24} & C_{34} & C_{44} \end{bmatrix}}_{(4\times 4 \text{ case})}$$

An important property for adjoint

For any $n \times n$ matrix A,

$$A \cdot \operatorname{adj} A = (\det A)\, I_n.$$

[1]:
```
import sympy as sp;import numpy as np
```

[2]:
```
A = sp.Matrix([[1,-2,5,0],[2,0,4,-1],[3,1,0,7],[0,4,-2,0]]); A
```

[2]:
$$\begin{bmatrix} 1 & -2 & 5 & 0 \\ 2 & 0 & 4 & -1 \\ 3 & 1 & 0 & 7 \\ 0 & 4 & -2 & 0 \end{bmatrix}$$

[3]:
```
# Determinant @ SymPy
A.det()
```

[3]: -158

[4]:
```
# Adjugate @ SymPy
A.adjugate()
```

[4]:
$$\begin{bmatrix} 114 & -112 & -16 & 61 \\ -34 & 14 & 2 & -57 \\ -68 & 28 & 4 & -35 \\ -44 & 46 & -16 & -18 \end{bmatrix}$$

[5]:
```
A = np.array([[1,-2,5,0],[2,0,4,-1],[3,1,0,7],[0,4,-2,0]]); A
```

[5]:
```
array([[ 1, -2,  5,  0],
       [ 2,  0,  4, -1],
       [ 3,  1,  0,  7],
       [ 0,  4, -2,  0]])
```

[6]:
```
# Determinant @ NumPy
np.linalg.det(A)
```

[6]: -158.00000000000003

Determinant

[1]:
```
import sympy as sp; import numpy as np; import time
```

[2]:
```
# generate a 5x5 random matrix between 0 and 9 @ Sympy
A = sp.randMatrix(5,5,0,9); A
```

[2]:
$$\begin{bmatrix} 2 & 2 & 6 & 2 & 2 \\ 7 & 9 & 6 & 3 & 2 \\ 7 & 2 & 1 & 2 & 1 \\ 9 & 4 & 4 & 5 & 0 \\ 5 & 2 & 2 & 6 & 7 \end{bmatrix}$$

[3]:
```
start = time.time(); a = A.det(); end = time.time(); a
```

[3]: -4422

[4]:
```
print("The computation time is :", end-start)
```

```
The computation time is : 0.0014178752899169922
```

[5]:
```
# generate a 200x200 random matrix between 0 and 9 @ SymPy
A = sp.randMatrix(200,200,0,9); A.shape
```

[5]: (200, 200)

[6]:
```
start = time.time(); a = A.det(); end = time.time(); a
```

[6]: $-14683656320388240268110009984217319541944253029804836488164958844207160042447383727736453 1$

[7]:
```
print("The computation time is :", end-start)
```

```
The computation time is : 58.75287485122681
```


Determinant

[8]:
```
# generate a 5x5 random matrix between 0 and 9 @ NumPy
A = np.random.randint(10,size=(5,5)); A
```

[8]:
```
array([[3, 1, 7, 5, 7],
       [8, 8, 5, 4, 3],
       [1, 1, 0, 6, 2],
       [6, 3, 0, 5, 9],
       [0, 7, 1, 7, 7]])
```

[9]:
```
start = time.time(); a = np.linalg.det(A); end = time.time(); a
```

[9]: 20678.999999999978

[10]:
```
print("The computation time is :", end-start)
```

```
The computation time is : 0.0001728534698486328
```

[11]:
```
# generate a 200x200 random matrix between 0 and 9 @ NumPy
A = np.random.randint(10,size=(200,200)); A.shape
```

[11]: (200, 200)

[12]:
```
start = time.time(); a = np.linalg.det(A); end = time.time(); a
```

[12]: 6.309923345255805e+279

[13]:
```
print("The computation time is :", end-start)
```

```
The computation time is : 0.0021011829376220703
```

Nonsingular matrices

- An $n \times n$ matrix A is said to be **nonsingular (or invertible)** if there is the unique **inverse**, denoted by A^{-1}, such that

$$A^{-1}A = I_n \quad \text{and} \quad AA^{-1} = I_n.$$

- A matrix that is NOT nonsingular is called a **singular matrix**.

Properties for nonsingular matrices

Suppose A and B are $n \times n$ nonsingular matrices. Then A^{-1}, A^T and AB are nonsingular and

1. $(A^{-1})^{-1} = A$,
2. $(A^T)^{-1} = (A^{-1})^T$,
3. $(AB)^{-1} = B^{-1}A^{-1}$.

Inverse Formula

A matrix A is nonsingular if and only if $\det A \neq 0$ and the inverse, if exists, is equal to

$$A^{-1} = \frac{1}{\det A} \operatorname{adj} A.$$

Inverse formula for 2×2 and 3×3 nonsingular matrices

Given a 2×2 or 3×3 matrix $A = [a_{ij}]$ with $\det A \neq 0$. Then

$(2 \times 2$ case$)$:

$$A^{-1} = \frac{1}{\det A} \begin{bmatrix} C_{11} & C_{21} \\ C_{12} & C_{22} \end{bmatrix} = \frac{1}{\det A} \begin{bmatrix} a_{22} & -a_{12} \\ -a_{21} & a_{11} \end{bmatrix}$$

$(3 \times 3$ case$)$:

$$A^{-1} = \frac{1}{\det A} \begin{bmatrix} C_{11} & C_{21} & C_{31} \\ C_{12} & C_{22} & C_{32} \\ C_{13} & C_{23} & C_{33} \end{bmatrix}$$

$$= \frac{1}{\det A} \begin{bmatrix} \begin{vmatrix} a_{22} & a_{23} \\ a_{32} & a_{33} \end{vmatrix} & - \begin{vmatrix} a_{12} & a_{13} \\ a_{32} & a_{33} \end{vmatrix} & \begin{vmatrix} a_{12} & a_{13} \\ a_{22} & a_{23} \end{vmatrix} \\ - \begin{vmatrix} a_{21} & a_{23} \\ a_{31} & a_{33} \end{vmatrix} & \begin{vmatrix} a_{11} & a_{13} \\ a_{31} & a_{33} \end{vmatrix} & - \begin{vmatrix} a_{11} & a_{13} \\ a_{21} & a_{23} \end{vmatrix} \\ \begin{vmatrix} a_{21} & a_{22} \\ a_{31} & a_{32} \end{vmatrix} & - \begin{vmatrix} a_{11} & a_{12} \\ a_{31} & a_{32} \end{vmatrix} & \begin{vmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{vmatrix} \end{bmatrix}$$

Nonsingular matrices

[1]:
```
import sympy as sp; import numpy as np
```

[2]:
```
A = sp.Matrix([[1,-2,5,0],[2,0,4,-1],[3,1,0,7],[0,4,-2,0]]); A
```

[2]:
$$\begin{bmatrix} 1 & -2 & 5 & 0 \\ 2 & 0 & 4 & -1 \\ 3 & 1 & 0 & 7 \\ 0 & 4 & -2 & 0 \end{bmatrix}$$

[3]:
```
# Inverse @ SymPy
A.inv()
```

[3]:
$$\begin{bmatrix} -\frac{57}{79} & \frac{56}{79} & \frac{8}{79} & -\frac{61}{158} \\ \frac{17}{79} & -\frac{7}{79} & -\frac{1}{79} & \frac{57}{158} \\ \frac{34}{79} & -\frac{14}{79} & -\frac{2}{79} & \frac{35}{158} \\ \frac{22}{79} & -\frac{23}{79} & \frac{8}{79} & \frac{9}{79} \end{bmatrix}$$

[4]:
```
A = np.array([[1,-2,5,0],[2,0,4,-1],[3,1,0,7],[0,4,-2,0]]); A
```

[4]:
```
array([[ 1, -2,  5,  0],
       [ 2,  0,  4, -1],
       [ 3,  1,  0,  7],
       [ 0,  4, -2,  0]])
```

[5]:
```
# Inverse @ NumPy
np.linalg.inv(A)
```

[5]:
```
array([[-0.72151899,  0.70886076,  0.10126582, -0.38607595],
       [ 0.21518987, -0.08860759, -0.01265823,  0.36075949],
       [ 0.43037975, -0.17721519, -0.02531646,  0.22151899],
       [ 0.27848101, -0.29113924,  0.10126582,  0.11392405]])
```


Partitioned matrices

In general, given a matrix

$$A = \begin{bmatrix} 2 & -3 & 1 & 0 & -4 & 1 \\ 1 & 5 & -2 & 3 & -1 & 2 \\ 0 & -4 & -2 & 7 & -1 & 3 \end{bmatrix}.$$

We can partition A as

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

The matrix can be written as the 2×3 partitioned (block) matrix

$$\begin{bmatrix} A_{11} & A_{12} & A_{13} \\ A_{21} & A_{22} & A_{33} \end{bmatrix}$$

whose entries are the blocks (submatrices)

$$A_{11} = \begin{bmatrix} 2 & -3 & 1 \\ 1 & 5 & -2 \end{bmatrix} \quad A_{12} = \begin{bmatrix} 0 & -4 \\ 3 & -1 \end{bmatrix} \quad A_{13} = \begin{bmatrix} 1 \\ 2 \end{bmatrix}$$

$$A_{21} = \begin{bmatrix} 0 & -4 & -2 \end{bmatrix} \quad A_{22} = \begin{bmatrix} 7 & -1 \end{bmatrix} \quad A_{23} = \begin{bmatrix} 3 \end{bmatrix}$$

[1]:
```
A = sp.Matrix([[2,-3,1,0,-4,1],␣
↪[1,5,-2,3,-1,2],␣
↪[0,-4,-2,7,-1,3]]); A
```

[1]:
$$\begin{bmatrix} 2 & -3 & 1 & 0 & -4 & 1 \\ 1 & 5 & -2 & 3 & -1 & 2 \\ 0 & -4 & -2 & 7 & -1 & 3 \end{bmatrix}$$

[2]:
```
A11 = A[0:2,0:3]; A11
```

[2]:
$$\begin{bmatrix} 2 & -3 & 1 \\ 1 & 5 & -2 \end{bmatrix}$$

[3]:
```
A12 = A[0:2,3:5]; A12
```

[3]:
$$\begin{bmatrix} 0 & -4 \\ 3 & -1 \end{bmatrix}$$

[4]:
```
A13 = A[0:2,5]; A13
```

[4]:
$$\begin{bmatrix} 1 \\ 2 \end{bmatrix}$$

[5]:
```
A21 = A[2,0:3]; A21
```

[5]:
$$\begin{bmatrix} 0 & -4 & -2 \end{bmatrix}$$

$$\begin{bmatrix} A & C \\ 0 & B \end{bmatrix}^{-1} = \begin{bmatrix} X & Z \\ 0 & Y \end{bmatrix}.$$

$\Rightarrow$ X, Y, Z ?

$$\begin{bmatrix} A & C \\ 0 & B \end{bmatrix} \begin{bmatrix} X & Z \\ 0 & Y \end{bmatrix} = \begin{bmatrix} I & 0 \\ 0 & I \end{bmatrix}$$

$$\Rightarrow \begin{bmatrix} AX & AZ+CY \\ 0 & BY \end{bmatrix} = \begin{bmatrix} I & 0 \\ 0 & I \end{bmatrix}$$

$$\Rightarrow \begin{cases} AX = I \\ AZ+CY = 0 \\ 0 = 0 \\ BY = I \end{cases} \Rightarrow \begin{array}{l} X = A^{-1} \\ Y = B^{-1} \\ Z = -A^{-1}CB^{-1} \end{array}$$

Addition and Scalar Multiplication

Suppose

$$A = [A_{ij}] = \begin{bmatrix} A_{11} & \cdots & A_{1s} \\ \vdots & \ddots & \vdots \\ A_{r1} & \cdots & A_{rs} \end{bmatrix} \quad \text{and} \quad B = [B_{ij}] = \begin{bmatrix} B_{11} & \cdots & B_{1s} \\ \vdots & \ddots & \vdots \\ B_{r1} & \cdots & B_{rs} \end{bmatrix}$$

are $r \times s$ block matrices c is a scalar. Then

1. $cA = [cA_{ij}] = \begin{bmatrix} cA_{11} & \cdots & cA_{1s} \\ \vdots & \ddots & \vdots \\ cA_{r1} & \cdots & cA_{rs} \end{bmatrix}$,
2. $A + B = [A_{ij} + B_{ij}] = \begin{bmatrix} A_{11} + B_{11} & \cdots & A_{1s} + B_{1s} \\ \vdots & \ddots & \vdots \\ A_{r1} + B_{r1} & \cdots & A_{rs} + B_{rs} \end{bmatrix}$.

Here, A_{ij} and B_{ij} have sizes for which the indicated sums and products are well defined.

Block Multiplication

Suppose

$$A = [A_{ij}] = \begin{bmatrix} A_{11} & \cdots & A_{1s} \\ \vdots & \ddots & \vdots \\ A_{r1} & \cdots & A_{rs} \end{bmatrix} \quad \text{and} \quad B = [B_{ij}] = \begin{bmatrix} B_{11} & \cdots & B_{1t} \\ \vdots & \ddots & \vdots \\ B_{s1} & \cdots & B_{st} \end{bmatrix}$$

are $r \times s$ and $s \times t$ block matrices respectively. Then

$$AB = \left[\sum_{k=1}^{s} A_{ik} B_{kj} \right] = \begin{bmatrix} \sum_{k=1}^{s} A_{1k} B_{k1} & \cdots & \sum_{k=1}^{s} A_{1k} B_{kt} \\ \vdots & \ddots & \vdots \\ \sum_{k=1}^{s} A_{rk} B_{k1} & \cdots & \sum_{k=1}^{s} A_{1k} B_{kt} \end{bmatrix}$$

In particular, if

$$A = \left[\begin{array}{c|c|c|c} \mathbf{a}_1 & \mathbf{a}_2 & \cdots & \mathbf{a}_s \end{array} \right] \quad \text{and} \quad B = \left[\begin{array}{c} \mathbf{b}_1 \\ \hline \mathbf{b}_2 \\ \hline \vdots \\ \hline \mathbf{b}_s \end{array} \right].$$

Then

$$AB = \left[\mathbf{a}_1 \mathbf{b}_1 + \mathbf{a}_2 \mathbf{b}_2 + \cdots + \mathbf{a}_s \mathbf{b}_s \right].$$

$\begin{bmatrix} A & C \\ 0 & B \end{bmatrix} \cdot \begin{bmatrix} X & Z \\ 0 & Y \end{bmatrix} = \begin{bmatrix} I & 0 \\ 0 & I \end{bmatrix}$

Example 6 Assume that A, B, X, and Y are square matrices. If the matrix $\begin{bmatrix} A & C \\ \mathbf{0} & B \end{bmatrix}$ is invertible and its inverse is $\begin{bmatrix} X & Z \\ \mathbf{0} & Y \end{bmatrix}$. Find X, Y, Z in terms of A, B, and C.

Solution. Since $\begin{bmatrix} X & Z \\ \mathbf{0} & Y \end{bmatrix}$ is the inverse of $\begin{bmatrix} A & C \\ 0 & B \end{bmatrix}$,

$$\begin{bmatrix} I & \mathbf{0} \\ \mathbf{0} & I \end{bmatrix} = I = \begin{bmatrix} A & C \\ 0 & B \end{bmatrix} \begin{bmatrix} X & Z \\ \mathbf{0} & Y \end{bmatrix} = \begin{bmatrix} AX & AZ+CY \\ \mathbf{0} & BY \end{bmatrix}.$$

Then $AX = I$, $BY = I$, and $AZ + CY = \mathbf{0}$. The first equation $AX = I$ implies that A is invertible and $X = A^{-1}$. The second equation $BY = I$ implies that B is invertible and $Y = B^{-1}$. Finally,

$$AZ + CY = \mathbf{0} \implies AZ = -CY = -CB^{-1} \implies Z = -A^{-1}CB^{-1}$$

Therefore, the inverse

$$\begin{bmatrix} X & Z \\ \mathbf{0} & Y \end{bmatrix} = \begin{bmatrix} A^{-1} & -A^{-1}CB^{-1} \\ \mathbf{0} & B^{-1} \end{bmatrix}.$$

System of linear equations

System of linear equations

- A **system of linear equations (or a linear system)** is a collection of one or more linear equations

$$\begin{cases} a_{11}x_1 + a_{12}x_2 + \cdots + a_{1n}x_n = b_1 \\ a_{21}x_1 + a_{22}x_2 + \cdots + a_{2n}x_n = b_2 \\ \qquad\qquad \vdots \qquad\qquad\qquad\quad \vdots \\ a_{m1}x_1 + a_{m2}x_2 + \cdots + a_{mn}x_n = b_m \end{cases}$$

- The system of linear equations can be written in **matrix form** $A\mathbf{x} = \mathbf{b}$, where

$$A = \begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{11} & a_{22} & \cdots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{m1} & a_{m2} & \cdots & a_{mn} \end{bmatrix}, \quad \mathbf{x} = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_m \end{bmatrix} \quad \text{and} \quad \mathbf{b} = \begin{bmatrix} b_1 \\ b_2 \\ \vdots \\ b_m \end{bmatrix}.$$

The matrix A is called the **coefficient matrix (or matrix of coefficients)** of the system.

$$\begin{bmatrix} A & C \\ 0 & B \end{bmatrix} \begin{bmatrix} X & Z \\ 0 & Y \end{bmatrix} = \begin{bmatrix} I & 0 \\ 0 & I \end{bmatrix}$$

$$AX + C \cdot 0 = I \Rightarrow X = A^{-1}$$

$$AZ + CY = 0$$

$$0 \cdot X + B \cdot 0 = 0$$

$$0 \cdot Z + BY = I$$

$$\Rightarrow Y = B^{-1}$$

$$AZ + CB^{-1} = 0$$

$$\Rightarrow AZ = -CB^{-1}$$

$$\downarrow A^{-1}$$

$$Z = -A^{-1}CB^{-1}$$

System of linear equations

- The matrix

$$\begin{bmatrix} A & \mathbf{b} \end{bmatrix} = \begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1n} & b_1 \\ a_{11} & a_{22} & \cdots & a_{2n} & b_2 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ a_{m1} & a_{m2} & \cdots & a_{mn} & b_m \end{bmatrix}$$

is called the **augmented matrix** of the system.

$Ax=b$

- A system of linear equations $A\mathbf{x} = \mathbf{b}$ has three possibilities.
  - No solution
  - Exactly one solution
  - Infinitely many solutions

$\Rightarrow \det(A) \neq 0$

$\Rightarrow \det(A) = 0$

It is said to be **consistent** if it has either one or infinitely many solution(s); a system is **inconsistent** if it has no solution.

Example 7 Consider the system of linear equations

$$\begin{cases} x_1 - 2x_2 + x_3 = 0 \\ 2x_2 - 8x_3 = 8 \\ -4x_1 + 5x_2 + 9x_3 = -9 \end{cases}$$

[1]:
```
import sympy as sp
```

[2]:
```
A = sp.
↪Matrix([[1,-2,1],[0,2,-8],[-4,5,9]]); A
```

[2]:
$$\begin{bmatrix} 1 & -2 & 1 \\ 0 & 2 & -8 \\ -4 & 5 & 9 \end{bmatrix}$$

[3]:
```
b = sp.Matrix([[0],[8],[-9]]); b
```

[3]:
$$\begin{bmatrix} 0 \\ 8 \\ -9 \end{bmatrix}$$

[4]:
```
# define x1, x2, x3
x1, x2, x3 = sp.symbols('x1 x2 x3')
```

[5]:
```
# Solve Ax = b @ SymPy
sp.linsolve((A,b),x1,x2,x3)
```

[5]: $\{(29,\ 16,\ 3)\}$

The solution is

$$\begin{cases} x_1 = 29 \\ x_2 = 16 \\ x_3 = 3 \end{cases}$$

System of linear equations

Example 7 Consider the system of linear equations

$$\begin{cases} x_1 - 2x_2 + x_3 = 0 \\ 2x_2 - 8x_3 = 8 \\ -4x_1 + 5x_2 + 9x_3 = -9 \end{cases}$$

[1]:
```
import numpy as np
```

[2]:
```
A = np.
↪array([[1,-2,1],[0,2,-8],[-4,5,9]]); A
```

[2]:
```
array([[ 1, -2,  1],
       [ 0,  2, -8],
       [-4,  5,  9]])
```

[3]:
```
b = np.array([[0],[8],[-9]]); b
```

[3]:
```
array([[ 0],
       [ 8],
       [-9]])
```

[4]:
```
# Solve Ax = b @ NumPy
np.linalg.solve(A,b)
```

[4]:
```
array([[29.],
       [16.],
       [ 3.]])
```

The solution is

$$\begin{cases} x_1 = 29 \\ x_2 = 16 \\ x_3 = 3 \end{cases}$$

System of linear equations

Example 8 Solve the system $\begin{cases} x_2 - 4x_3 = 8 \\ 2x_1 - 3x_2 + 2x_3 = 1 \\ 5x_1 - 8x_2 + 7x_3 = 1 \end{cases}$

[1]:
```
import sympy as sp
```

[2]:
```
A = sp.
↪Matrix([[0,1,-4],[2,-3,2],[5,-8,7]]); A
```

[2]:
$$\begin{bmatrix} 0 & 1 & -4 \\ 2 & -3 & 2 \\ 5 & -8 & 7 \end{bmatrix}$$

[3]:
```
b = sp.Matrix([[8],[1],[1]]); b
```

[3]:
$$\begin{bmatrix} 8 \\ 1 \\ 1 \end{bmatrix}$$

[4]:
```
# Solve Ax = b @ SymPy
x1, x2, x3 = sp.symbols('x1 x2 x3')
sp.linsolve((A,b),x1,x2,x3)
```

[4]: $\emptyset$

The system is inconsistent (has no solution).

[1]:
```
import numpy as np
```

[2]:
```
A = np.
↪array([[0,1,-4],[2,-3,2],[5,-8,7]]); A
```

[2]:
```
array([[ 0,  1, -4],
       [ 2, -3,  2],
       [ 5, -8,  7]])
```

[3]:
```
b = np.array([[8],[1],[1]]); b
```

[3]:
```
array([[8],
       [1],
       [1]])
```

[4]:
```
# Solve Ax = b @ NumPy
np.linalg.solve(A,b)
```

[4]:
```
array([[-9.00719925e+15],
       [-7.20575940e+15],
       [-1.80143985e+15]])
```

System of linear equations

Example 9 Solve the system of linear equations

$$\begin{cases} & 3x_2 - 6x_3 + 6x_4 + 4x_5 = -5 \\ 3x_1 - & 7x_2 + 8x_3 - 5x_4 + 8x_5 = 9 \\ 3x_1 - & 9x_2 + 12x_3 - 9x_4 + 6x_5 = 15 \end{cases}$$

[1]:
```
import sympy as sp
```

[2]:
```
A = sp.Matrix([[0,3,-6,6,4],
↪[3,-7,8,-5,8], [3,-9,12,-9,6]]);A
```

[2]:
$$\begin{bmatrix} 0 & 3 & -6 & 6 & 4 \\ 3 & -7 & 8 & -5 & 8 \\ 3 & -9 & 12 & -9 & 6 \end{bmatrix}$$

[3]:
```
b = sp.Matrix([[-5],[9],[15]]); b
```

[3]:
$$\begin{bmatrix} -5 \\ 9 \\ 15 \end{bmatrix}$$

[5]:
```
x1, x2, x3, x4, x5 = sp.symbols('x1 x2 x3
↪x4 x5')
```

[6]:
```
sp.linsolve((A,b),x1,x2,x3,x4,x5)
```

[6]: $\{(2x_3 - 3x_4 - 24,\ 2x_3 - 2x_4 - 7,\ x_3,\ x_4,\ 4)\}$

So the **general solution** of the system is

$$\begin{cases} x_1 = -24 + 2x_3 - 3x_4 \\ x_2 = -7 + 2x_3 - 2x_4 \\ x_5 = 4 \\ x_3,\ x_4 \text{ are free.} \end{cases}$$

the solution can also be rewritten as

$$\mathbf{x} = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \end{bmatrix} = \begin{bmatrix} -24 + 2x_3 - 3x_4 \\ -7 + 2x_3 - 2x_4 \\ x_3 \\ x_4 \\ 4 \end{bmatrix}.$$

System of linear equations

[7]:
```
import numpy as np
A = np.array([[0,3,-6,6,4], [3,-7,8,-5,8], [3,-9,12,-9,6]]); A
```

[7]:
```
array([[ 0,  3, -6,  6,  4],
       [ 3, -7,  8, -5,  8],
       [ 3, -9, 12, -9,  6]])
```

[8]:
```
b = np.array([[-5],[9],[15]]); b
```

[8]:
```
array([[-5],
       [ 9],
       [15]])
```

[9]:
```
# np.linalg.solve cannot solve infinity many solution case
np.linalg.solve(A,b)
```

```
---------------------------------------------------------------------------
LinAlgError                               Traceback (most recent call last)
/var/folders/wh/1v8jgf9d5gbfykybbcypgbt40000gn/T/ipykernel_26654/351369431.py in <module>
      1 # np.linalg.solve cannot solve infinity many solution case
----> 2 np.linalg.solve(A,b)

<__array_function__ internals> in solve(*args, **kwargs)

/opt/anaconda3/lib/python3.9/site-packages/numpy/linalg/linalg.py in solve(a, b)
    378     a, _ = _makearray(a)
    379     _assert_stacked_2d(a)
--> 380     _assert_stacked_square(a)
    381     b, wrap = _makearray(b)
    382     t, result_t = _commonType(a, b)

LinAlgError: Last 2 dimensions of the array must be square
```

Elementary row operations

Row Echelon Form

The **leading entry** of a row refers to the leftmost **nonzero** entry.

$$\begin{bmatrix} 0 & 3 & -6 & 6 & 4 & -5 \\ 3 & -7 & 8 & -5 & 8 & 9 \\ 3 & -9 & 12 & -9 & 6 & 15 \end{bmatrix} \quad \begin{bmatrix} 3 & -9 & 12 & -9 & 6 & 15 \\ 3 & -7 & 8 & -5 & 8 & 9 \\ 0 & 3 & -6 & 6 & 4 & -5 \end{bmatrix} \quad \begin{bmatrix} 3 & -9 & 12 & -9 & 6 & 15 \\ 0 & 2 & -4 & 4 & 2 & -6 \\ 0 & 0 & 0 & 0 & 1 & 4 \end{bmatrix}$$

A rectangular matrix is in **row echelon form (REF)** if it has the following two properties:

1. If both the k-th and $(k+1)$-th rows have nonzero entries, the number of zeros before the leading entry in the k-th row is strictly smaller than the number of zeros before the leading entry in the $(k+1)$-th row.

$$\begin{bmatrix} * & * & * & * & * & * \\ 0 & 3 & * & * & * & * \\ 0 & 0 & 0 & 4 & * & * \\ 0 & 0 & 0 & 0 & * & * \end{bmatrix} \quad \begin{bmatrix} * & * & * & * & * & * \\ 0 & 0 & 0 & 4 & * & * \\ 0 & 3 & * & * & * & * \\ 0 & 0 & 0 & 0 & * & * \end{bmatrix} \quad \begin{bmatrix} * & * & * & * & * & * \\ 0 & 3 & * & * & * & * \\ 0 & 4 & * & * & * & * \\ 0 & 0 & 0 & * & * & * \end{bmatrix}$$

(Satisfied) (Not satisfied) (Not satisfied)

2. If there are rows whose entries are all zero, they are below the rows having nonzero entries.

$$\begin{bmatrix} * & * & * & * & * & * \\ 0 & 3 & * & * & * & * \\ 0 & 0 & 0 & 4 & * & * \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \quad \begin{bmatrix} * & * & * & * & * & * \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 3 & * & * & * & * \\ 0 & 0 & 0 & 4 & * & * \end{bmatrix} \quad \begin{bmatrix} * & * & * & * & * & * \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 4 & * & * & * & * \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

(Satisfied) (Not satisfied) (Not satisfied)

Elementary row operations

Reduced Row Echelon Form

If a matrix in **row echelon form** satisfies the following additional conditions, then it is in **reduced row echelon form (RREF)** :

3. The leading entry in each nonzero row is 1.
4. Each leading 1 is the only nonzero entry in its column.

$$\begin{bmatrix} 1 & 0 & * & 0 & * & * \\ 0 & 1 & * & 0 & * & * \\ 0 & 0 & 0 & 1 & * & * \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \quad \begin{bmatrix} 1 & 0 & * & * & * & * \\ 0 & 1 & * & * & * & * \\ 0 & 0 & 0 & 1 & * & * \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \quad \begin{bmatrix} 2 & 0 & * & 0 & * & * \\ 0 & 3 & * & 0 & * & * \\ 0 & 0 & 0 & 4 & * & * \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

(RREF) (Not RREF) (Not RREF)

Uniqueness of the Reduced Row Echelon Form

Each matrix is row equivalent to one and only one reduced row echelon form.

Remark A matrix can have many different row echelon forms.

Pivot position / column

- A **pivot position** in a matrix A is a location in A that corresponds to a leading 1 in the reduced row echelon form of the matrix A.
- A **pivot column** is a column of A that contains a pivot position.

Elementary row operations

1. (Replacement) Replace one row by the sum of itself and a multiple of another row.

$$\begin{bmatrix} a_1 & a_2 & a_3 & a_4 \\ b_1 & b_2 & b_3 & b_4 \\ c_1 & c_2 & c_3 & c_4 \end{bmatrix} \underset{\sim}{R_2 - \frac{b_1}{a_1} R_1 \to R_2} \begin{bmatrix} a_1 & a_2 & a_3 & a_4 \\ 0 & b_2 - \frac{b_1}{a_1} a_2 & b_3 - \frac{b_1}{a_1} a_3 & b_4 - \frac{b_1}{a_1} a_4 \\ c_1 & c_2 & c_3 & c_4 \end{bmatrix}$$

2. (Scaling) Multiply all entries in a row by a nonzero constant.

$$\begin{bmatrix} a_1 & a_2 & a_3 & a_4 \\ b_1 & b_2 & b_3 & b_4 \\ c_1 & c_2 & c_3 & c_4 \end{bmatrix} \underset{\sim}{k \times R_2 \to R_2} \begin{bmatrix} a_1 & a_2 & a_3 & a_4 \\ k\,b_1 & k\,b_2 & k\,b_3 & k\,b_4 \\ c_1 & c_2 & c_3 & c_4 \end{bmatrix}$$

3. (Interchange) Interchange two rows.

$$\begin{bmatrix} a_1 & a_2 & a_3 & a_4 \\ b_1 & b_2 & b_3 & b_4 \\ c_1 & c_2 & c_3 & c_4 \end{bmatrix} \underset{\sim}{R_1 \leftrightarrow R_2} \begin{bmatrix} b_1 & b_2 & b_3 & b_4 \\ a_1 & a_2 & a_3 & a_4 \\ c_1 & c_2 & c_3 & c_4 \end{bmatrix}$$

Example 10 [Re-visit] Consider the system of linear equations

$$\begin{cases} x_1 - 2x_2 + x_3 = 0 \\ 2x_2 - 8x_3 = 8 \\ -4x_1 + 5x_2 + 9x_3 = -9 \end{cases}$$

[1]:
```
A = sp.Matrix([[1,-2,1,0], [0,2,-8,8],␣
↪[-4,5,9,-9]]); A
```

[1]:
$$\begin{bmatrix} 1 & -2 & 1 & 0 \\ 0 & 2 & -8 & 8 \\ -4 & 5 & 9 & -9 \end{bmatrix}$$

[2]:
```
# RREF
A.rref()
```

[2]:
```
(Matrix([
 [1, 0, 0, 29],
 [0, 1, 0, 16],
 [0, 0, 1,  3]]),
 (0, 1, 2))
```

The RREF of the augmented matrix is

$$\begin{bmatrix} 1 & -2 & 1 & 0 \\ 0 & 2 & -8 & 8 \\ -4 & 5 & 9 & -9 \end{bmatrix} \sim \cdots \sim \begin{bmatrix} 1 & 0 & 0 & 29 \\ 0 & 1 & 0 & 16 \\ 0 & 0 & 1 & 3 \end{bmatrix}.$$

The solution is

$$\begin{cases} x_1 = 29 \\ x_2 = 16 \\ x_3 = 3 \end{cases}$$

Rank(A) + dim Null(A) = n.

Elementary row operations

Example 11 [Re-visit] Solve the system of linear equations

$$\begin{cases} & 3x_2 - 6x_3 + 6x_4 + 4x_5 = -5 \\ 3x_1 - 7x_2 + 8x_3 - 5x_4 + 8x_5 = 9 \\ 3x_1 - 9x_2 + 12x_3 - 9x_4 + 6x_5 = 15 \end{cases}$$

```
[1]: A = sp.
     ↪Matrix([[0,3,-6,6,4,-5],␣
     ↪[3,-7,8,-5,8,9],␣
     ↪[3,-9,12,-9,6,15]]);A
```

```
[1]: ⎡0   3   -6   6   4   -5⎤
     ⎢3   -7  8    -5  8   9 ⎥
     ⎣3   -9  12   -9  6   15⎦
```

Solution. Consider the augmented matrix $\begin{bmatrix} A & \mathbf{b} \end{bmatrix}$.

$$\begin{bmatrix} 0 & 3 & -6 & 6 & 4 & -5 \\ 3 & -7 & 8 & -5 & 8 & 9 \\ 3 & -9 & 12 & -9 & 6 & 15 \end{bmatrix} \sim \begin{bmatrix} 1 & 0 & -2 & 3 & 0 & -24 \\ 0 & 1 & -2 & 2 & 0 & -7 \\ 0 & 0 & 0 & 0 & 1 & 4 \end{bmatrix}.$$

```
[2]: # RREF
     A.rref()
```

```
[2]: (Matrix([
      [1, 0, -2, 3, 0, -24],
      [0, 1, -2, 2, 0,  -7],
      [0, 0,  0, 0, 1,   4]]),
      (0, 1, 4))
```

The associated system now is

$$\begin{cases} x_1 - 2x_3 + 3x_4 = -24 \\ x_2 - 2x_3 + 2x_4 = -7 \\ x_5 = 4 \end{cases}$$

So the **general solution** of the system is

$$\begin{cases} x_1 = -24 + 2x_3 - 3x_4 \\ x_2 = -7 + 2x_3 - 2x_4 \\ x_5 = 4 \\ x_3,\ x_4 \text{ are free.} \end{cases} \iff \mathbf{x} = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \end{bmatrix} = \begin{bmatrix} -24 + 2x_3 - 3x_4 \\ -7 + 2x_3 - 2x_4 \\ x_3 \\ x_4 \\ 4 \end{bmatrix}.$$

Elementary row operations

Example 12 [Re-visit] Solve the system of linear equations

$$\begin{cases} & 3x_2 - 6x_3 + 6x_4 + 4x_5 = 0 \\ 3x_1 - 7x_2 + 8x_3 - 5x_4 + 8x_5 = 0 \\ 3x_1 - 9x_2 + 12x_3 - 9x_4 + 6x_5 = 0 \end{cases}$$

Solution. Consider the augmented matrix $\begin{bmatrix} A & \mathbf{b} \end{bmatrix}$.

$$\begin{bmatrix} 0 & 3 & -6 & 6 & 4 & 0 \\ 3 & -7 & 8 & -5 & 8 & 0 \\ 3 & -9 & 12 & -9 & 6 & 0 \end{bmatrix} \sim \cdots \sim \begin{bmatrix} 1 & 0 & -2 & 3 & 0 & 0 \\ 0 & 1 & -2 & 2 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \end{bmatrix}.$$

So the **general solution** of the system is

$$\begin{cases} x_1 = 2x_3 - 3x_4 \\ x_2 = 2x_3 - 2x_4 \\ x_5 = 0 \\ x_3,\ x_4 \text{ are free.} \end{cases}$$

The above solution can be rewritten in the parametric vector form

$$\mathbf{x} = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \end{bmatrix} = \begin{bmatrix} 2x_3 - 3x_4 \\ 2x_3 - 2x_4 \\ x_3 \\ x_4 \\ 0 \end{bmatrix} = x_3 \underbrace{\begin{bmatrix} 2 \\ 2 \\ 1 \\ 0 \\ 0 \end{bmatrix}}_{\mathbf{v}_1} + x_4 \underbrace{\begin{bmatrix} -3 \\ -2 \\ 0 \\ 1 \\ 0 \end{bmatrix}}_{\mathbf{v}_2} = x_3\mathbf{v}_1 + x_4\mathbf{v}_2.$$

This shows that every solution is a linear combination of $\mathbf{v}_1$ and $\mathbf{v}_2$ (Explain later).

Elementary row operations

[1]:
```
A = sp.Matrix([[0,3,-6,6,4],
↪[3,-7,8,-5,8], [3,-9,12,-9,6]]); A
```

[1]:
$$\begin{bmatrix} 0 & 3 & -6 & 6 & 4 \\ 3 & -7 & 8 & -5 & 8 \\ 3 & -9 & 12 & -9 & 6 \end{bmatrix}$$

[2]:
```
# define 3x1 zero vector
b = sp.zeros(3,1); b
```

[2]:
$$\begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}$$

Method 1

[3]:
```
x1,x2,x3,x4,x5 = sp.symbols('x1 x2 x3 x4
↪x5')
sp.linsolve((A,b),x1,x2,x3,x4,x5)
```

[3]: $\{(2x_3 - 3x_4,\ 2x_3 - 2x_4,\ x_3,\ x_4,\ 0)\}$

Method 2

[4]:
```
A.rref()
```

[4]:
```
(Matrix([
 [1, 0, -2, 3, 0],
 [0, 1, -2, 2, 0],
 [0, 0,  0, 0, 1]]),
 (0, 1, 4))
```

Method 3

[5]:
```
A.nullspace()
```

[5]:
```
[Matrix([
 [2],
 [2],
 [1],
 [0],
 [0]]),
 Matrix([
 [-3],
 [-2],
 [ 0],
 [ 1],
 [ 0]])]
```


Homogeneous system

Homogeneous System

- A system of linear equation is said to be **homogeneous** if it can be written in the form
$$A\mathbf{x} = \mathbf{0},$$
where A is an $m \times n$ matrix and $\mathbf{0}$ is the zero vector in $\mathbb{R}^m$.
- A homogeneous system $A\mathbf{x} = \mathbf{0}$ **always** has at least one solution, namely, $\mathbf{x} = \mathbf{0}$. This zero solution is usually called the **trivial solution**.
- A nonzero vector $\mathbf{x}$ that satisfies $A\mathbf{x} = \mathbf{0}$ is called a **nontrivial solution**.

Properties of homogeneous system

- A homogeneous system $A\mathbf{x} = \mathbf{0}$ is always consistent.
- A homogeneous equation $A\mathbf{x} = \mathbf{0}$ has a nontrivial solution if and only if the equation has **at least one free variable**.
- Suppose A is a **square** matrix. Then $A\mathbf{x} = \mathbf{0}$ has a nontrivial solution if and only if A is singular.
- If A is an $m \times n$ matrix with $m < n$, then $A\mathbf{x} = \mathbf{0}$ has a nontrivial solution.