Say we want to encode messages up to $c = 10$ characters. For example, we take the following message: „I love KIU". Lets use Python:

```
message = 'I love KIU'
message_L = list(message)
print(message)
print(message_L)
```

```
I love KIU
['I', ' ', 'l', 'o', 'v', 'e', ' ', 'K', 'I', 'U']
```

First, we translate this message into corresponding unicode and calculate the length of the list of numbers

```
NumericMessage = [ord(s) for s in message_L] #ord() returns the number representing
                                              #the unicode code of a specified character.
print(NumericMessage, 'length of this vector is', len(NumericMessage))
```

```
[73, 32, 108, 111, 118, 101, 32, 75, 73, 85] length of this vector is 10
```

Now, lets transform this list of numbers into 10 dimentional column vector (adding empty space if necessary)

```
c = 10
for i in range(c-len(NumericMessage)):
        NumericMessage.append(ord(' '))
v = vector(NumericMessage).column()
show(v)
```

$$\vec{v} = \begin{pmatrix} 73 \\ 32 \\ 108 \\ 111 \\ 118 \\ 101 \\ 32 \\ 75 \\ 73 \\ 85 \end{pmatrix}$$

We would like to encode this vector $\vec{v}$. For this we generate random (NOT SMART, WE WILL SEE why) 100 numbers a_{ij}, $i,j = 1, \dots, 100$ and create a new vector w by $w_i = a_{i1} v_1 + a_{i2} v_2 + \dots + a_{i10} v_{10} = a_{i1} \cdot 73 + a_{i2} \cdot 32 + \dots + a_{i10} \cdot 85$. In other words, we generate 10×10 matrix A with random entries a_{ij}, and put $\vec{w} = A \cdot \vec{v}$

```
A = random_matrix(ZZ, 10,10)
show(A)
```

$$\begin{pmatrix} -1 & -2 & -8 & 0 & 2 & -1 & -2 & 4 & 0 & -2 \\ 2 & 1 & 4 & 1 & 0 & 3 & 1 & -2 & 5 & 6 \end{pmatrix}$$

For example, for w_1 we have

```
show(A)
```

$$\begin{pmatrix} -1 & -2 & -8 & 0 & 2 & -1 & -2 & 4 & 0 & -2 \\ 2 & 1 & 4 & 1 & 0 & 3 & 1 & -2 & 5 & 6 \\ -1 & 0 & 0 & -1 & 0 & -3 & 1 & -1 & -7 & -1 \\ 1 & -1 & 0 & 0 & -1 & 1 & 1 & 36 & -1 & 1 \\ 1 & -3 & 0 & 5 & 1 & 0 & 1 & -2 & 0 & -10 \\ -3 & 0 & 0 & 1 & -1 & 1 & -1 & 3 & -6 & -11 \\ -1 & 0 & -3 & 0 & 0 & 2 & 1 & 38 & -325 & 0 \\ 0 & 2 & 0 & -1 & 3 & 0 & 1 & 2 & 2 & 1 \\ 0 & 0 & 1 & 5 & 15 & -2 & -4 & -1 & 0 & 1 \\ -2 & 1 & 2 & -2 & -2 & -3 & 3 & 1 & 4 & 1 \end{pmatrix}$$

For example, from w_1 we have
$w_1 = -1\cdot 73 - 2\cdot 32 - 8\cdot 108 + 0\cdot 111 + 2\cdot 118 -$
$- 1\cdot 101 - 2\cdot 32 + 4\cdot 75 + 0\cdot 73 - 2\cdot 85 =$
$= -800.$
So, doing this for each w_i, we calculate $\vec{w} = A\cdot\vec{v}$ to get an encoded message $\vec{w}$.

```
Encoded_message = A*v
show(Encoded_message)
```

$$\begin{pmatrix} -800 \\ 1781 \\ -1126 \\ 2768 \\ -318 \\ -1305 \\ -21038 \\ 720 \\ 2113 \\ -111 \end{pmatrix} = \vec{w}.$$

How is recipient suppose to decode our message, if we agreed on random matrix A beforehand?
(otherwise it is not possible).
Recipient only recieves numbers w_i, that is vector $\vec{w}$, so she has to solve system of 10 equations $a_{i1}x_1 + a_{i2}x_2 + \dots + a_{i10}x_{10} = w_i$ for x_i, to recover $\vec{x} = \vec{v}$.

Equivalently, she solves a system $A\cdot\vec{x} = \vec{w}$.
This amounts to bringing $(A|\vec{w})$ to the echelon form.

```
w = A*v
A_aug = A.augment(w)
show(A_aug)
```

$$\left(\begin{array}{cccccccccc|c} -1 & -2 & -8 & 0 & 2 & -1 & -2 & 4 & 0 & -2 & -800 \\ 2 & 1 & 4 & 1 & 0 & 3 & 1 & -2 & 5 & 6 & 1781 \\ -1 & 0 & 0 & -1 & 0 & -3 & 1 & -1 & -7 & -1 & -1126 \\ 1 & -1 & 0 & 0 & -1 & 1 & 1 & 36 & -1 & 1 & 2768 \\ 1 & -3 & 0 & 5 & 1 & 0 & 1 & -2 & 0 & -10 & -318 \\ -3 & 0 & 0 & 1 & -1 & 1 & -1 & 3 & -6 & -11 & -1305 \\ -1 & 0 & -3 & 0 & 0 & 2 & 1 & 38 & -325 & 0 & -21038 \\ 0 & 2 & 0 & -1 & 3 & 0 & 1 & 2 & 2 & 1 & 720 \\ 0 & 0 & 1 & 5 & 15 & -2 & -4 & -1 & 0 & 1 & 2113 \\ -2 & 1 & 2 & -2 & -2 & -3 & 3 & 1 & 4 & 1 & -111 \end{array}\right)$$

Augmented matrix $(A|\vec{w})$ of the system.

```
show(A_aug.rref())
```

$$\begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 73 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 32 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 108 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 111 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 118 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 101 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 32 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 75 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 73 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 85 \end{pmatrix}$$

rref$(A|\vec{w})$ gives a unique solution $\vec{v}$.

$$\left(\begin{array}{cccccccccc|c}
0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 75 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 73 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 85
\end{array}\right)$$

But, the receiver has to do this solving procedure for every different message. This is not practical! If $\vec{y}$ is any random message, receiver is looking for the decoding transformation

$$\vec{y} \longrightarrow \vec{x}$$

which is the inverse of the encoding transformation

$$\vec{x} \xrightarrow{A\cdot} \vec{y}$$

given by multiplication by A. Method of finding such procedure is nothing new. We just have to perform same elimination now for some general vector $\vec{y}$ and unknown $\vec{x}$. That is, row reduce $(A|\vec{y})$ for some general $\vec{y}$. As a result, we get 10 equations

$$x_i = b_{i1} y_1 + b_{i2} y_2 + \cdots + b_{i\,10} y_{10} \qquad i = 1, .., 10.$$

Or, in matrix form

$$\vec{x} = B\vec{y}$$

Such matrix B is called an inverse matrix of A and is denoted by $A^{-1} := B$. Computing:

```
B = A.inverse()
show(B)
```

$$\begin{pmatrix}
-\frac{10991727}{18711521} & -\frac{23591104}{18711521} & -\frac{27433411}{18711521} & -\frac{592817}{18711521} & \frac{3536318}{18711521} & -\frac{10921568}{18711521} & \frac{502860}{18711521} & \frac{3877705}{18711521} & \frac{202426}{18711521} & \frac{3868377}{18711521} \\
-\frac{409406791}{149692168} & -\frac{1127932475}{149692168} & -\frac{1826265297}{149692168} & -\frac{42163165}{74846084} & \frac{127551111}{149692168} & -\frac{427532323}{149692168} & \frac{4789164}{18711521} & \frac{30870069}{74846084} & \frac{21024983}{37423042} & \frac{158414447}{37423042} \\
\frac{40960729}{74846084} & \frac{141212497}{74846084} & \frac{237527659}{74846084} & \frac{6075883}{37423042} & -\frac{19039033}{74846084} & \frac{57015273}{74846084} & -\frac{1283404}{18711521} & -\frac{4956235}{37423042} & -\frac{2214014}{18711521} & -\frac{21107858}{18711521} \\
-\frac{310805581}{149692168} & -\frac{894251041}{149692168} & -\frac{1516019563}{149692168} & -\frac{34534655}{74846084} & \frac{120003061}{149692168} & -\frac{350848289}{149692168} & \frac{4054871}{18711521} & \frac{2224599}{74846084} & \frac{19628427}{37423042} & \frac{138670339}{37423042} \\
18592288 & 51478684 & 83703490 & 3676763 & 5802074 & 19776400 & 1746031 & 749939 & 3367635 & 29523445
\end{pmatrix}$$

$$\begin{pmatrix}
\frac{40960729}{74846084} & \frac{141212497}{74846084} & \frac{237527659}{74846084} & \frac{6075883}{37423042} & -\frac{19039033}{74846084} & \frac{57015273}{74846084} & -\frac{1283404}{18711521} & -\frac{4956235}{37423042} & -\frac{2214014}{18711521} & -\frac{21107858}{18711521} \\
-\frac{310805581}{149692168} & -\frac{894251041}{149692168} & -\frac{1516019563}{149692168} & -\frac{34534655}{74846084} & \frac{120003061}{149692168} & -\frac{350848289}{149692168} & \frac{4054871}{18711521} & \frac{2224599}{74846084} & \frac{19628427}{37423042} & \frac{138670339}{37423042} \\
\frac{18592288}{18711521} & \frac{51478684}{18711521} & \frac{83703490}{18711521} & \frac{3676763}{18711521} & -\frac{5802074}{18711521} & \frac{19776400}{18711521} & -\frac{1746031}{18711521} & \frac{749939}{18711521} & -\frac{3367635}{18711521} & -\frac{29523445}{18711521} \\
\frac{44119743}{37423042} & \frac{119889635}{37423042} & \frac{179002043}{37423042} & \frac{3732438}{18711521} & -\frac{13596931}{37423042} & \frac{47391463}{37423042} & -\frac{1849662}{18711521} & -\frac{250312}{18711521} & -\frac{4874466}{18711521} & -\frac{31987409}{18711521} \\
\frac{50521757}{74846084} & \frac{126062089}{74846084} & \frac{180199979}{74846084} & \frac{3307595}{37423042} & -\frac{3108761}{74846084} & \frac{42048733}{74846084} & -\frac{818976}{18711521} & \frac{5312761}{37423042} & -\frac{2985147}{18711521} & -\frac{12245178}{18711521} \\
-\frac{1516650}{18711521} & -\frac{4190127}{18711521} & -\frac{6595209}{18711521} & \frac{216818}{18711521} & \frac{381929}{18711521} & -\frac{1482844}{18711521} & \frac{133488}{18711521} & \frac{98367}{18711521} & \frac{382376}{18711521} & \frac{2322698}{18711521} \\
-\frac{126839}{37423042} & -\frac{554695}{37423042} & -\frac{1090943}{37423042} & \frac{27190}{18711521} & \frac{66967}{37423042} & -\frac{186365}{37423042} & -\frac{45569}{18711521} & \frac{29078}{18711521} & \frac{25341}{18711521} & \frac{219993}{18711521} \\
-\frac{13969313}{149692168} & -\frac{54519093}{149692168} & -\frac{118521799}{149692168} & -\frac{2896735}{74846084} & \frac{3721537}{149692168} & -\frac{29298285}{149692168} & \frac{357774}{18711521} & -\frac{5313805}{74846084} & \frac{2123697}{37423042} & \frac{13301707}{37423042}
\end{pmatrix} = A^{-1}$$

So, given any message $\vec{v}$, we code by multiplieng by A and decode a vector by multiplieng by A^{-1}.

$$\vec{v} \underset{\text{decoding with } A^{-1}}{\overset{\text{coding with } A\cdot\_}{\rightleftarrows}} \vec{w}$$

Lets check: (recall $B = A^{-1}$)

```
Decoded_message = B*w
print([chr(i) for i in Decoded_message.list()]) #chr() inverso of ord
print(''.join([chr(i) for i in Decoded_message.list()]))
```

```
['I', ' ', 'l', 'o', 'v', 'e', ' ', 'K', 'I', 'U']
I love KIU
```

So, we indeed recovered our message.

But what if we chose A such that $A\vec{x} = \vec{w}$ does not have a unique solution for some $\vec{w}$. Then we will not be able to decode! (For example, if $\operatorname{rank}(A) \neq 10$)

Explanation: $n \times n$ matrix A defines a map

$$\mathbb{R}^n \xrightarrow{A} \mathbb{R}^n$$
$$x \longmapsto Ax$$

if $A\vec{x} = \vec{w}$ has more than one solution, say

$A\vec{x} = A\vec{y} = \vec{w} \iff A$ is not injective. ($\operatorname{rank}(A) \neq n$)

if now $A\vec{x} = \vec{w}$ has a unique solution for if now $A\vec{x}=\vec{w}$ has a unique solution for every $\vec{w}$ $\iff$ A is bijective ($\text{rank}(A)=n$)

But A is not just any function! We proved that $A(\vec{x}+\vec{y}) = A\vec{x} + A\vec{y}$ and $A(k\vec{x}) = k(A\vec{x})$ $k \in \mathbb{R}$.

These are very important in linear algebra, we generalize for any pair of dimensions and define:

Definition: a function $T: \mathbb{R}^m \to \mathbb{R}^n$ is called a linear transformation if

a) $T(\vec{x}+\vec{y}) = T\vec{x} + T\vec{y}$ and b) $T(k\vec{x}) = k(T\vec{x})$

for all $\vec{x}, \vec{y} \in \mathbb{R}^m$ and $k \in \mathbb{R}$.

Befor continuing, lets introduce some notation. Let $\vec{e}_i \in \mathbb{R}^m$ denote the vector

$$\vec{e}_i = \begin{pmatrix} 0 \\ 0 \\ \vdots \\ 1 \\ \vdots \\ 0 \end{pmatrix} \leftarrow 1 \text{ in the } i^{th} \text{ place.}$$

For example, for $e_1, e_2, e_3 \in \mathbb{R}^3$, we have

$$e_1 = \begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix}, \quad e_2 = \begin{pmatrix} 0 \\ 1 \\ 0 \end{pmatrix}, \quad e_3 = \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix}.$$

Note, that for any vector $\vec{x} \in \mathbb{R}^n$, we have the equality

$$\vec{x} = \begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{pmatrix} = \begin{pmatrix} x_1 \\ 0 \\ 0 \\ \vdots \\ 0 \end{pmatrix} + \begin{pmatrix} 0 \\ x_2 \\ 0 \\ \vdots \\ 0 \end{pmatrix} + \begin{pmatrix} 0 \\ 0 \\ \vdots \\ x_n \end{pmatrix} = x_1 \begin{pmatrix} 1 \\ 0 \\ \vdots \\ 0 \end{pmatrix} + \dots + x_n \begin{pmatrix} 0 \\ \vdots \\ 1 \end{pmatrix} =$$

$$= x_1 \cdot \vec{e}_1 + x_2 \vec{e}_2 + \dots + x_n \vec{e}_n$$

Theorem: a function $T: \mathbb{R}^m \to \mathbb{R}^n$ is a linear transformation if and only if there exists an $n \times m$ matrix A, such that

$$T(\vec{x}) = A \cdot \vec{x}$$

for all $\vec{x} \in \mathbb{R}^m$.

Proof: We know that for a matrix A, a linear transform. $A: \mathbb{R}^m \to \mathbb{R}^n$ satisfies a) and b). We need to prove converse. Say $T: \mathbb{R}^m \to \mathbb{R}^n$ satisfies a) and b). We want to show that there exists $n \times m$ matrix A, such that $T(\vec{x}) = A\vec{x}$ for all $\vec{x} \in \mathbb{R}^m$. But, for every $\vec{x} \in \mathbb{R}^m$

$$T(\vec{x}) = T\left(\begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_m \end{pmatrix}\right) = T(x_1 \vec{e}_1 + \dots + x_m \vec{e}_m) \overset{a)}{=} T(x_1 \vec{e}_1) + \dots$$

$$\dots + T(x_m \vec{e}_m) \overset{b)}{=} x_1 T(\vec{e}_1) + \dots + x_m T(\vec{e}_m) \overset{\text{def.}}{=}$$

$$= \begin{pmatrix} | & | & & | \\ T(\vec{e}_1) & T(\vec{e}_2) & \dots & T(\vec{e}_m) \\ | & | & & | \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_m \end{pmatrix} = A_T \vec{x}$$

$$\begin{pmatrix} | & | & & | \end{pmatrix} \begin{pmatrix} \vdots \\ x_m \end{pmatrix}$$

where $$A_T = \begin{pmatrix} | & | & & | \\ T(\vec{e}_1) & T(\vec{e}_2) & \cdots & T(\vec{e}_m) \\ | & | & & | \end{pmatrix}.$$ (columns are $T(\vec{e}_i)$.)

$\square$.

While proving the above Theorem, we have also proved

The columns of the matrix of a linear transformation

Consider a linear transformation T from $\mathbb{R}^m$ to $\mathbb{R}^n$. Then, the matrix of T is

$$A_T = \begin{bmatrix} | & | & & | \\ T(\vec{e}_1) & T(\vec{e}_2) & \cdots & T(\vec{e}_m) \\ | & | & & | \end{bmatrix}, \quad \text{where} \quad \vec{e}_i = \begin{bmatrix} 0 \\ 0 \\ \vdots \\ 1 \\ \vdots \\ 0 \end{bmatrix} \leftarrow i\text{th} \; .$$

If we denote $T(\vec{x}) = A_T\vec{x} = \vec{y}$, and write this in components, we have

$$\begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{bmatrix} = \begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1m} \\ a_{21} & a_{22} & \cdots & a_{2m} \\ \vdots & \vdots & & \vdots \\ a_{n1} & a_{n2} & \cdots & a_{nm} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_m \end{bmatrix} = \begin{bmatrix} a_{11}x_1 + a_{12}x_2 + \cdots + a_{1m}x_m \\ a_{21}x_1 + a_{22}x_2 + \cdots + a_{2m}x_m \\ \vdots \qquad \vdots \qquad \vdots \\ a_{n1}x_1 + a_{n2}x_2 + \cdots + a_{nm}x_m \end{bmatrix},$$

or

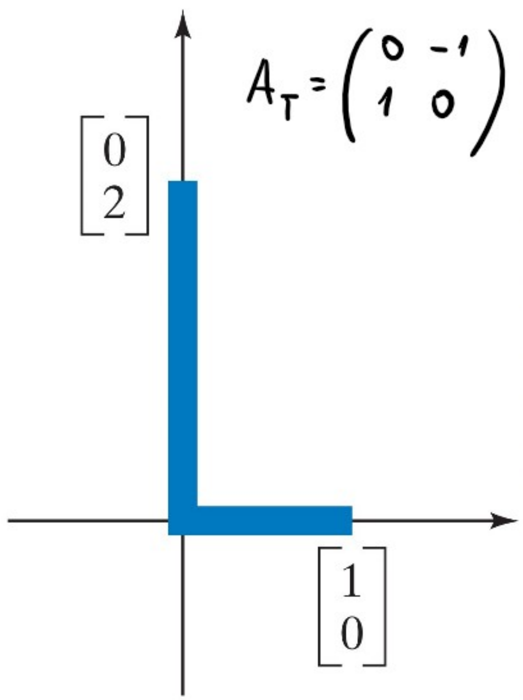
$$\begin{aligned} y_1 &= a_{11}x_1 + a_{12}x_2 + \cdots + a_{1m}x_m \\ y_2 &= a_{21}x_1 + a_{22}x_2 + \cdots + a_{2m}x_m \\ \vdots &= \vdots \qquad \vdots \qquad \vdots \\ y_n &= a_{n1}x_1 + a_{n2}x_2 + \cdots + a_{nm}x_m. \end{aligned}$$

So, the output variables y_i are linear functions

So, the output variables y_i are linear functions of input variables x_i. (Note that there is no constant term!). Hence the name „linear".

Geometry of linear transformations

$$A_T = \begin{pmatrix} 0 & -1 \\ 1 & 0 \end{pmatrix}$$

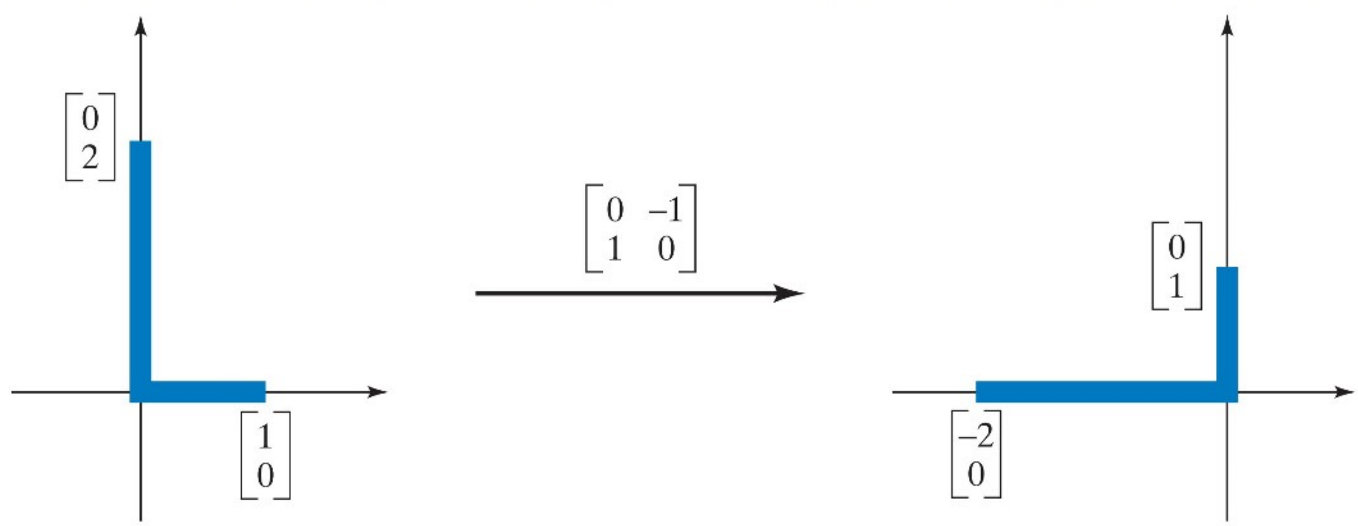


Consider letter „L" (for „linear") in $\mathbb{R}^2$ made up of vectors $\begin{pmatrix} 1 \\ 0 \end{pmatrix}$ and $\begin{pmatrix} 0 \\ 2 \end{pmatrix}$. Lets examine what happens under linear transformation $T(\vec{x}) = \begin{pmatrix} 0 & -1 \\ 1 & 0 \end{pmatrix}\vec{x}$. We have,

$$T\begin{pmatrix} 1 \\ 0 \end{pmatrix} = \begin{pmatrix} 0 & -1 \\ 1 & 0 \end{pmatrix}\begin{pmatrix} 1 \\ 0 \end{pmatrix} = \begin{pmatrix} 0 \\ 1 \end{pmatrix}$$

$$T\begin{pmatrix} 0 \\ 2 \end{pmatrix} = \begin{pmatrix} 0 & -1 \\ 1 & 0 \end{pmatrix}\begin{pmatrix} 0 \\ 2 \end{pmatrix} = \begin{pmatrix} -2 \\ 0 \end{pmatrix}.$$

So,



Now, for a general vector $\vec{x} = \begin{pmatrix} x_1 \\ x_2 \end{pmatrix}$, we have

$$T(\vec{x}) = \begin{pmatrix} 0 & -1 \\ 1 & 0 \end{pmatrix}\begin{pmatrix} x_1 \\ x_2 \end{pmatrix} = \begin{pmatrix} -x_2 \\ x_1 \end{pmatrix}$$

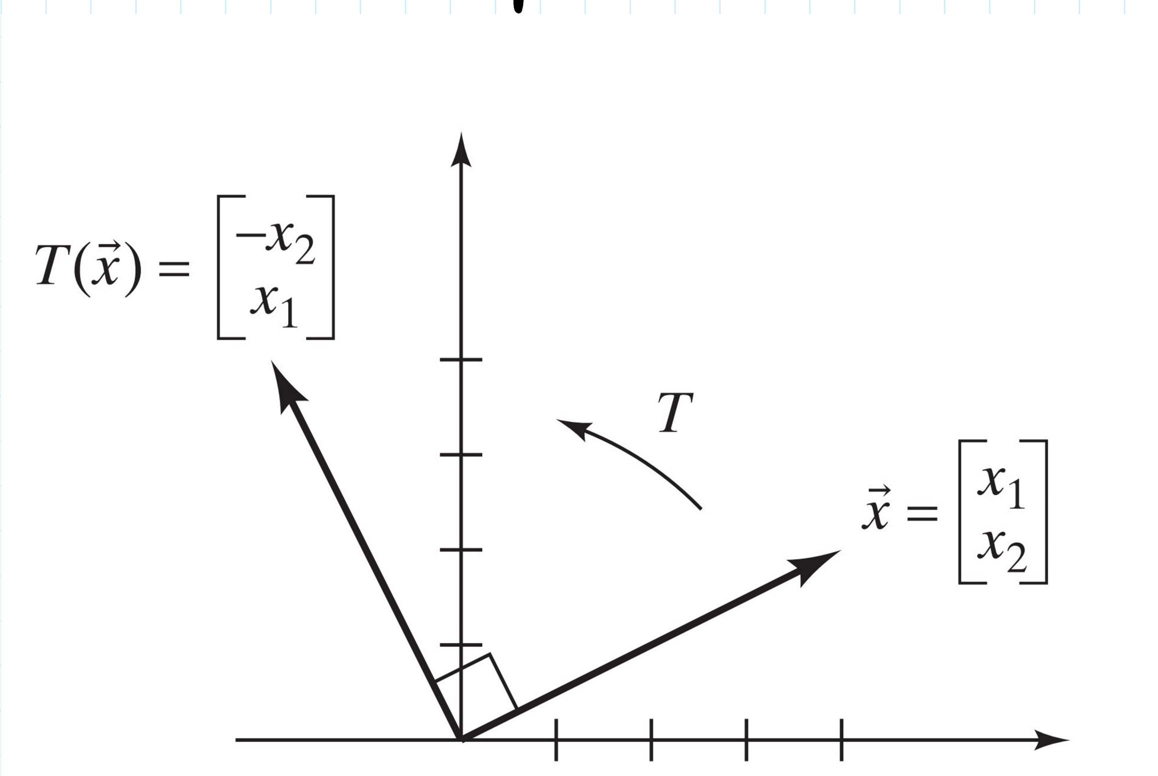
First, observe that length of $\vec{x}$ and $T(\vec{x})$ is the same:

$$\sqrt{x_1^2 + x_2^2} = \sqrt{(-x_2)^2 + x_1^2}$$

Also, for the dot product

$$\begin{pmatrix} x_1 \\ x_2 \end{pmatrix} \cdot \begin{pmatrix} -x_2 \\ x_1 \end{pmatrix} = -x_1 \cdot x_2 + x_1 \cdot x_2 = 0$$

which means $\vec{x}$ and $T(\vec{x})$ are perpendicular. Paying attention to the signs of the components, we see that T represents a $90°$ rotation.

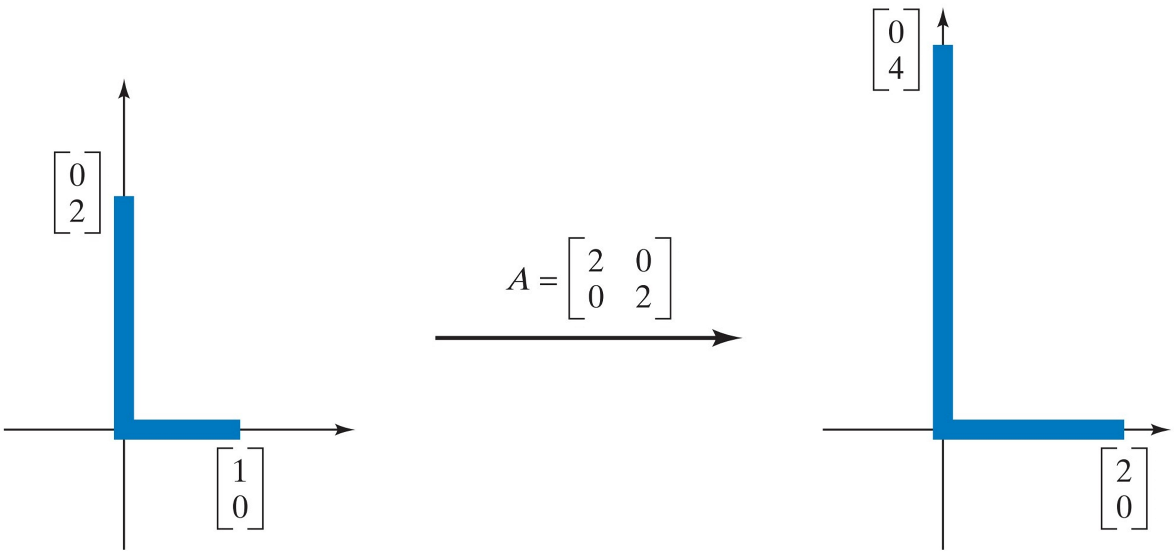


Every linear transformation $T: \mathbb{R}^n \to \mathbb{R}^n$ for $n = 2, 3$ corresponds

Every linear transformation $T: \mathbb{R}^n \to \mathbb{R}^n$ for $n=2,3$ corresponds to a transformation on a plane / in space respectively

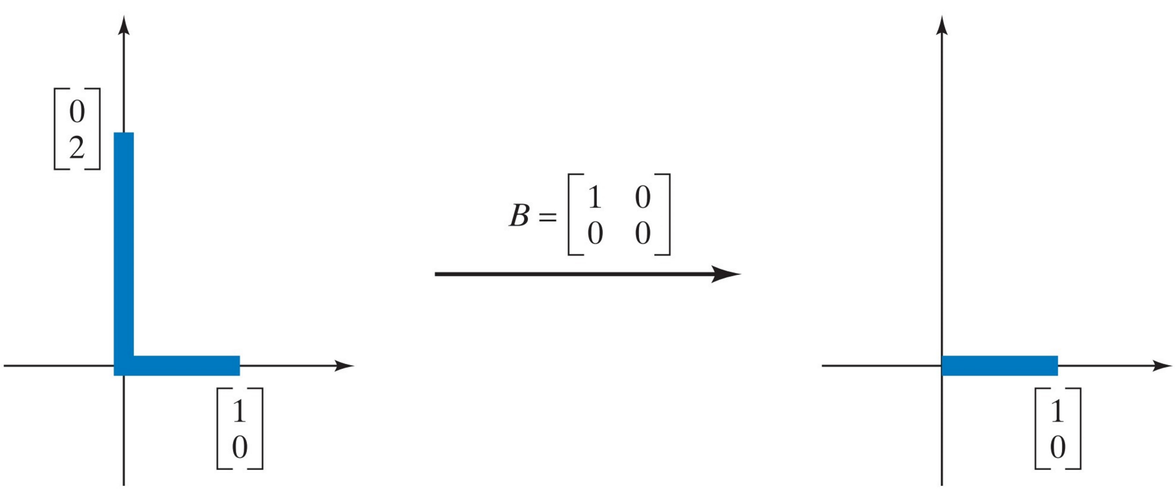
Lets look at some other examples:

a.



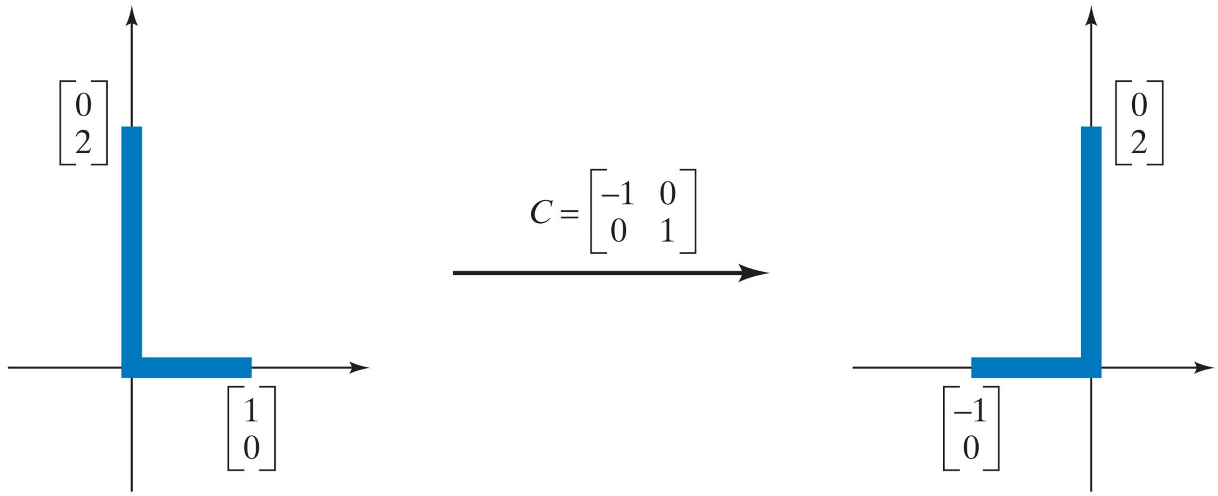
The L gets enlarged by a factor of 2; we will call this transformation a *scaling* by 2.

b.



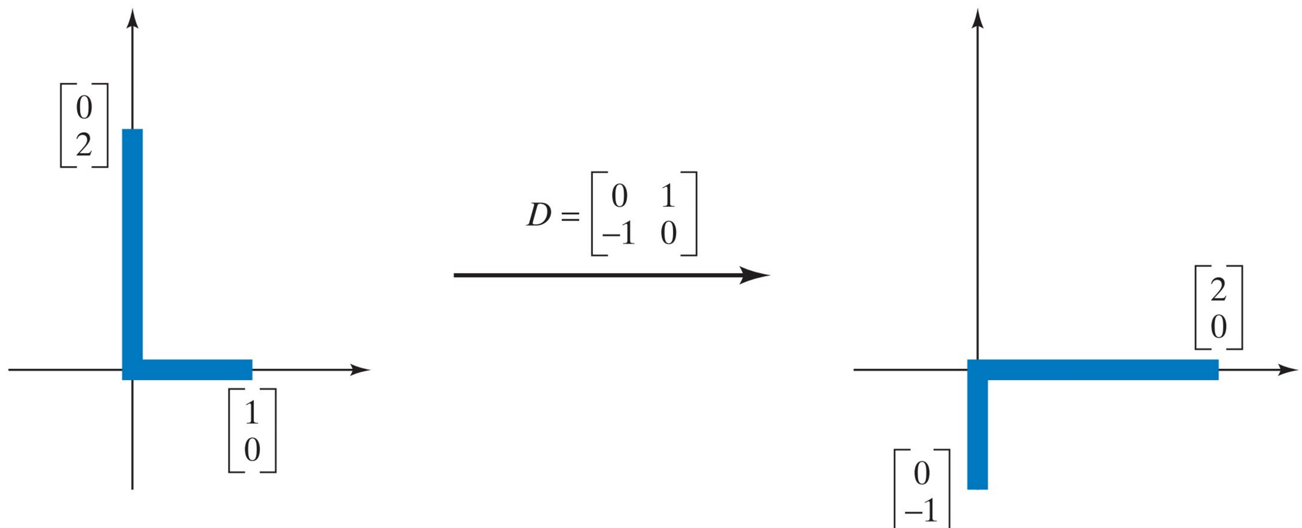
The L gets smashed into the horizontal axis. We will call this transformation the *orthogonal projection onto the horizontal axis*.

c.



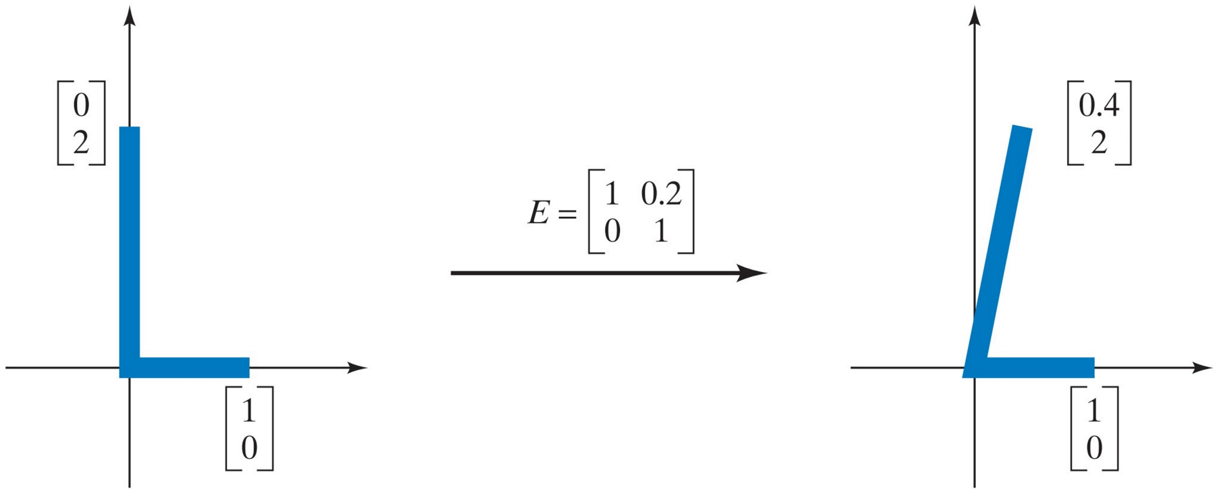
The L gets flipped over the vertical axis. We will call this the *reflection about the vertical axis*.

d.



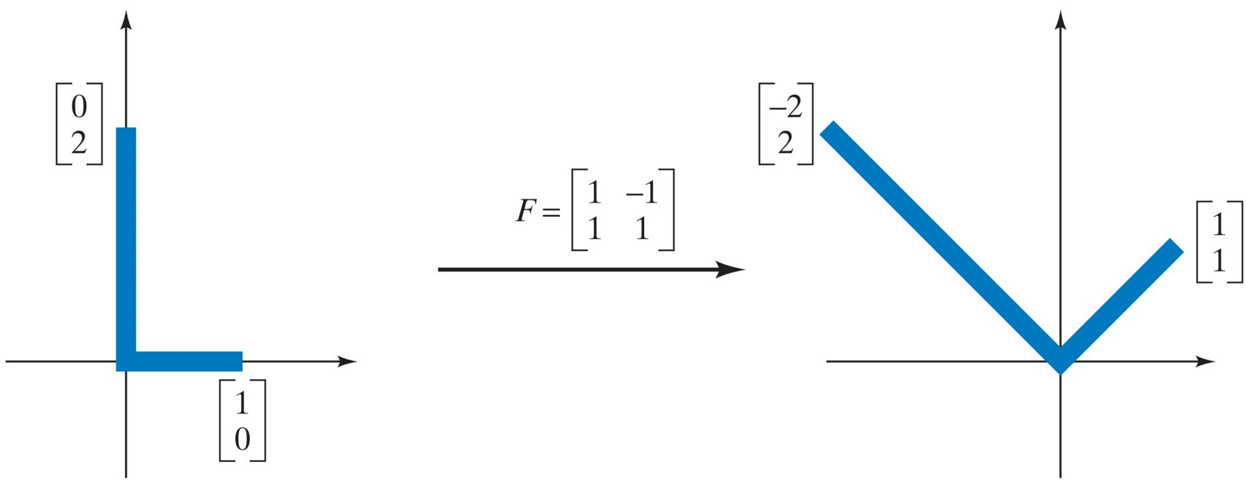
90° clockwise rotation. opposite of our exmple where transformation matrix was $\begin{pmatrix}0 & -1\\1 & 0\end{pmatrix}$.

e.



The foot of the L remains unchanged, while the back is shifted horizontally to the right; the L is italicized, becoming *L*. We will call this transformation a *horizontal shear*.

f.



There are two things going on here: The L is rotated through 45° and also enlarged (scaled) by a factor of $\sqrt{2}$. This is a *rotation combined with a scaling* (you may perform the two transformations in either order).

Before finishing, we discuss one more example:

Lets develop a simple model on how people surf the World Wide Web. For simplicity, consider a „mini-web" with 4 pages, labelled 1, 2, 3 and 4, linked as with 4 pages, labelled 1, 2, 3 and 4, linked as demonstrated by a directed graph:

$$\begin{tikzcd} 1 \arrow[r, shift left] \arrow[d] & 2 \arrow[l, shift left] \arrow[dl] \\ 3 \arrow[r] & 4 \arrow[u] \end{tikzcd}$$

Let x_1, x_2, x_3, x_4 be a proportion of surfers who find themselves on each of four pages initially. We collect this information in a 4-dimentional distribution vector $\left(\sum_{i=1}^{4} x_i = 1\right)$

$$\vec{x} = \begin{pmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{pmatrix}.$$ Example: $\vec{x} = \begin{pmatrix} 0.4 \\ 0.1 \\ 0.3 \\ 0.2 \end{pmatrix}$, meaning 40% of surfers are initially on page 1, 10% on page 2, 30% on page 3, and 20% on page 4. Componens of the distribution vector add up to 1, that is, 100%.

At a predetermined time, at the same exact time, each surfer will follow one of the links randomly, with equal proportion following each link if several links are available. Example: $\frac{x_1}{2}$ will go to page 2, $\frac{x_1}{2}$ will go to page 3, x_4 will go to page 2 and so on.

Let $\vec{y}$ be a distribution vector after transition. Then

$$y_1 = \frac{1}{2} x_2$$

$$
\begin{aligned}
y_1 &= \tfrac{1}{2}x_2 \\
y_2 &= \tfrac{1}{2}x_1 + x_4 \\
y_3 &= \tfrac{1}{2}x_1 + \tfrac{1}{2}x_2 \\
y_4 &= x_3
\end{aligned}
$$

Or, in vector form $\vec{y} = A\vec{x}$, where

$$
A = \begin{pmatrix} 0 & \frac{1}{2} & 0 & 0 \\ \frac{1}{2} & 0 & 0 & 1 \\ \frac{1}{2} & \frac{1}{2} & 0 & 0 \\ 0 & 0 & 1 & 0 \end{pmatrix}
$$

(So $T(x) = A\vec{x}$ is a linear transformation)

j^{th} column of A tells us where surfers go from page j. Generally, let c_j denote the number of links going out of page j. (in our case: $c_1 = 2$, $c_2 = 2$, $c_3 = 1$, $c_4 = 1$)

We have:

$$
a_{ij} = \begin{cases} \frac{1}{c_j} & \text{if there is a link } i \to j \\ 0 & \text{otherwize} \end{cases}
$$

Now, we ask if there is an equilibrium point. This is a vector $\vec{x}$, such that $A\vec{x} = \vec{x}$ (In other words, „nothing changes anymore".) To find out, we need to solve

$$
\begin{cases} \tfrac{1}{2}x_2 = x_1 \end{cases} \qquad \begin{cases} -x_1 + \tfrac{1}{2}x_2 = 0 \end{cases}
$$

$$\begin{cases} \frac{1}{2}x_2 = x_1 \\ \frac{1}{2}x_1 + x_4 = x_2 \\ \frac{1}{2}x_1 + \frac{1}{2}x_2 = x_3 \\ x_3 = x_4 \end{cases} \overset{\text{or}}{\Longrightarrow} \begin{cases} -x_1 + \frac{1}{2}x_2 = 0 \\ \frac{1}{2}x_1 - x_2 + x_4 = 0 \\ \frac{1}{2}x_1 + \frac{1}{2}x_2 - x_3 = 0 \\ x_3 - x_4 = 0 \end{cases}$$

$$\left(\begin{array}{cccc|c} -1 & \frac{1}{2} & 0 & 0 & 0 \\ \frac{1}{2} & -1 & 0 & 1 & 0 \\ \frac{1}{2} & \frac{1}{2} & -1 & 0 & 0 \\ 0 & 0 & 1 & -1 & 0 \end{array}\right) \overset{\text{rref}}{\Longrightarrow} \left(\begin{array}{cccc|c} 1 & 0 & 0 & -\frac{2}{3} & 0 \\ 0 & 1 & 0 & -\frac{4}{3} & 0 \\ 0 & 0 & 1 & -1 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{array}\right)$$

So, solution is

$$\vec{x} = \begin{pmatrix} \frac{2t}{3} \\ \frac{4t}{3} \\ t \\ t \end{pmatrix}, \quad \text{for all } t \in \mathbb{R}.$$

In addition, we want $1 = x_1 + x_2 + x_3 + x_4 = \frac{2t}{3} + \frac{4t}{3} + t + t = 4t$.

So, $t = \frac{1}{4}$. Thus,

$$\vec{x}_{equ} = \begin{pmatrix} \frac{1}{6} \\ \frac{1}{3} \\ \frac{1}{4} \\ \frac{1}{4} \end{pmatrix} \approx \begin{pmatrix} 16.7\ \% \\ 33.3\ \% \\ 25\ \% \\ 25\ \% \end{pmatrix}.$$

In this context, an interesting question arises: If we iterate our transition, letting the surfers move to a new page over and over again, following links at random, will the system eventually approach this equilibrium state $\vec{x}_{equ}$, *regardless of the initial distribution*? Perhaps surprisingly, the answer is affirmative for the mini-Web considered in this example, as well as for many others: *The equilibrium distribution represents the distribution of the surfers in the long run, for any initial distribution.*

$\vec{x}_{equ}$ is actually a simplified version of PageRank by Sergei Brin and Lawrence Page (1998), from seminal paper „The Anatomy of a large-scale Hypertextual Search Engine" (Here, they presented a prototyp of a search engine Google). In our example, Page 2 most popular, with PageRank of 1/3, while page 1 is half as popular, with PageRank 1/6.