백터

- 숫자를 원소로 가지는 리스트(list) 또는 배열(array)

```
x = [1, 7, 2]
x = np.array([1,7,2])
```

- 공간에서 **한 점**을 나타냄
- 원점으로부터 상대적 위치 표현
- 숫자를 곱하면 길이만 변한다.
- α: 스칼라곱

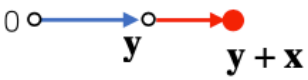
$$\alpha x = \begin{bmatrix} \alpha x_1 \\ \alpha x_2 \\ \alpha x_3 \end{bmatrix}$$

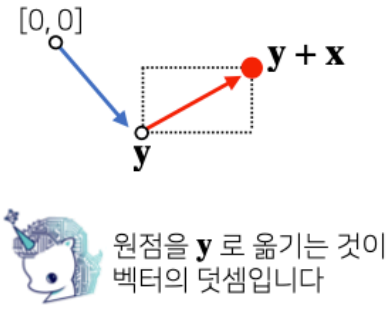
> $\alpha x_1 > 1$ => 길이 늘어남
>
> $\alpha x_1 < 1$ => 길이 줄어듬
>
> $\alpha x_1 < 0$ => 반대 방향

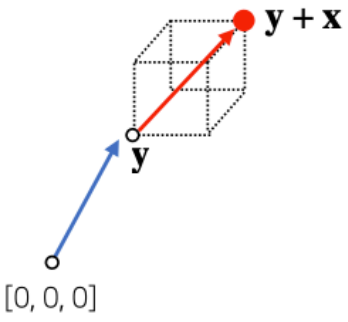
- 같은 모양을 가지면 덧셈, 뺄셈을 계산할 수 있다.
- 같은 모양을 가지면 성분곱(Hadamard product) 계산 가능

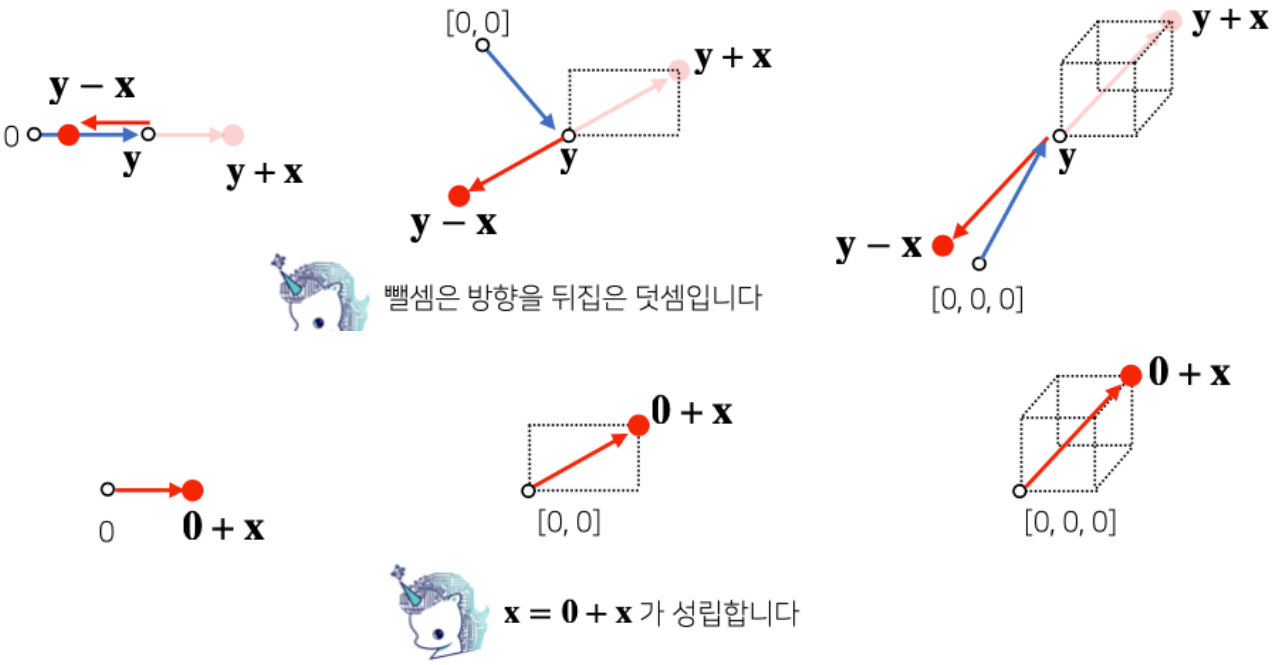
$$x = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_d \end{bmatrix}, y = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_d \end{bmatrix}, x \odot y = \begin{bmatrix} x_1 y_1 \\ x_2 y_2 \\ \vdots \\ x_d y_d \end{bmatrix}$$

- 두 벡터의 덧셈: 다른 백터로부터 상대적 위치이동 표현

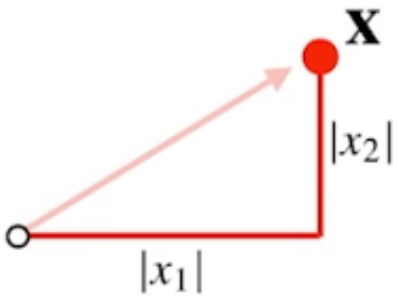




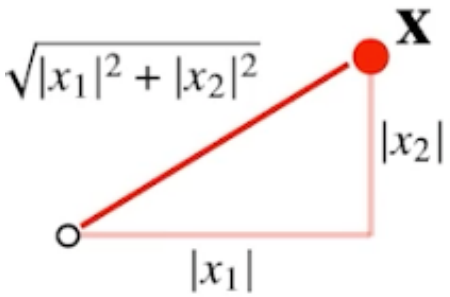




- 백터의 **노름(norm)**: 원점에서부터의 거리
  - L_1**-노름**: 각 성분의 변화량의 절대값을 모두 더함 => 맨하튼 노름

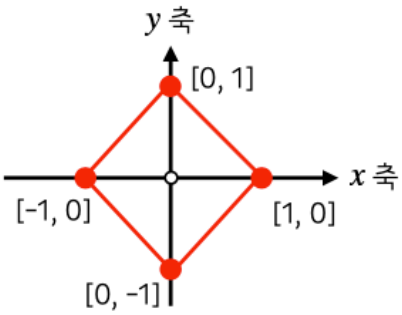


  - L_2**-노름**: 피타고라스 정리를 이용해 유클리드 거리 계산 => 유클리드 노름

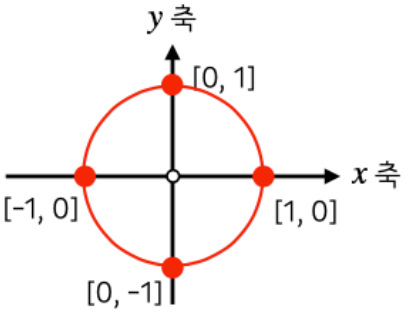


```python
def l1_norm(x):
  x_norm = np.abs(x)
  x_norm = np.sum(x_norm)
  return x_norm

def l2_norm(x):
  x_norm = x*x
  x_norm = np.sum(x_norm)
  x_norm = np.sqrt(x_norm)
  return x_norm

# L2 노름은 np.linalg.norm을 이용해도 구현 가능
```

- 노름의 종류에 따라 **기하학적 성질 달라진다.**



L_1-노름 상의 원: $\{\mathbf{x} : \|\mathbf{x}\|_1 = 1\}$

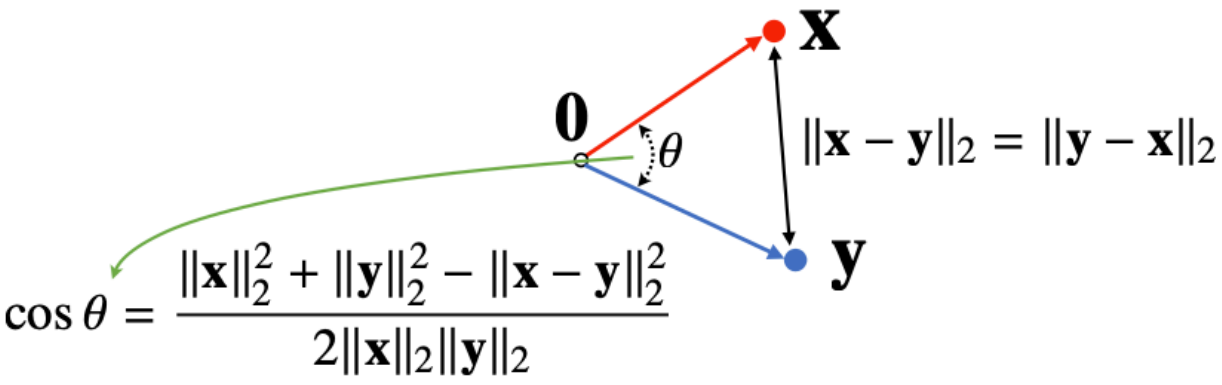


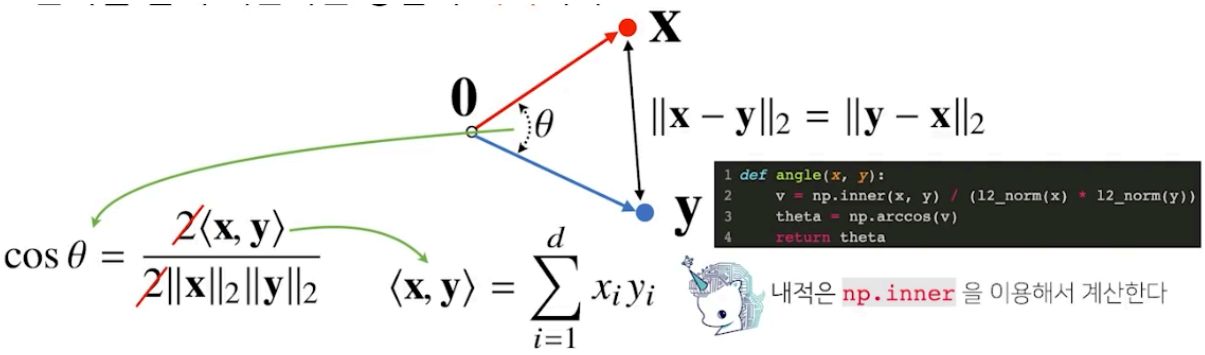
L_2-노름 상의 원: $\{\mathbf{x} : \|\mathbf{x}\|_2 = 1\}$

- 두 벡터 사이의 거리 계산 => 벡터 뺄셈 이용 $\|x - y\|_2 = \|y - x\|_2$
- 두 벡터 사이의 거리 이용하여 각도 계산 **(L_2-노름인 $\|\ldots\|_2$만 가능하다)**

  제2 코사인 법칙에 의해 두 벡터 사이의 각도 계산할 수 있다.

  분자를 쉽게 계산하는 방법이 **내적(inner product)**

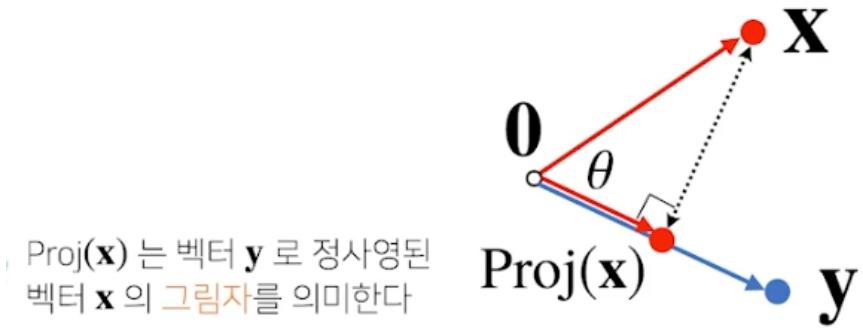




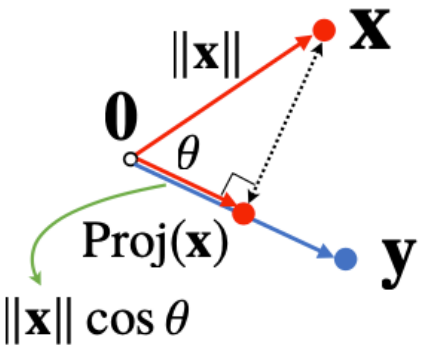
- **내적**

  정사영(orthogonal projection)된 벡터의 길이와 관련

  $< x, y >= \|x\|_2 \|y\|_2 \cos(\theta)$



- Proj(x)의 길이는 코사인법칙에 의해 $\|x\|cos\theta$ 가 된다.



- 내적은 정사영의 길이를 백터 y의 길이 $\|y\|$만큼 조정한 값