① Pre-train word and phrases vectors (Google News word embedding dataset)

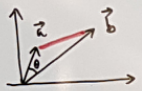
100 Billion words → Train → 3 Million words and phrases with 300 dimensional vectors.

Ex.

word	1	2	...	300
word 1	0.01	-0.02		0.7
word 2				
⋮				
word M				

← 300 dimensional vector

- **Cosine Similarity** of $\vec{a}$, $\vec{b}$



$$\cos\theta = \frac{\vec{a}\cdot\vec{b}}{\|\vec{a}\|\|\vec{b}\|} = \frac{\sum_{i=1}^{N} a_i \cdot b_i}{\sqrt{\sum_{i=1}^{N} a_i^2}\sqrt{\sum_{i=1}^{N} b_i^2}} = \frac{np.dot(\vec{a},\vec{b})}{np.linalg.norm(\vec{a}) \times np.linalg.norm(\vec{b})}$$

where $\vec{a}, \vec{b}$ are embedding (word) vectors.

$\theta \in [0° \div 180°] \Rightarrow \cos\theta \in [-1, 1] = [-1 \quad 0 \quad 1]$

dissimilarity similarity

- **Euclidean Distance**

$$d(\vec{a},\vec{b}) = d(\vec{b},\vec{a}) = \sqrt{\sum_{i=1}^{N}(a_i - b_i)^2}$$. Ex in 2D : $d(\vec{a},\vec{b}) = \sqrt{(a_1-b_1)^2 + (a_2-b_2)^2}$

$= np.sqrt(np.sum((\vec{a} - \vec{b})**2))$ ← Note: numpy broadcasting

- **Predict** : Ex. Canberra (city 1) is to Australia (country 1) as Hanoi (city 2) is to ? (country 2)

```
def get_country(city1, country1, city2, embeddings):
    embedding = { "Canberra" : [ . . . . . ]
                  "Australia" : [ . . . . . ]
                  ⋮
                  "US"        : [ . . . . . ]
```

country 1 − city 1 = country 2 − city 2

⇒ vec = country 1 − city 1 + city 2

⇒ Just chose the country that has maximum of cosine_similarity(vec, country)

- Test accuracy with given (known) set of (country, city)

PCA

Eigen Vector : Uncorrelated features of the data (orthogonal)

Eigen Value : The amount of information retained by each feature (variance)

Mean Normalized data (z-score) : $\overline{x_i} = \frac{x_i - \mu_i}{\sigma_i}$

↓

Covariance Matrix $\sum$ where $\Sigma_{ij} = cov_{ij} = \frac{1}{N-1}\sum_{i=1}^{N}\frac{(x_i - \overline{x_i})(x_j - \overline{x_j})}{N}$

↓

Perform SVD (Singular Value Decomposition) : $SVD(\Sigma)$

↓

Eigen Vector Eigen Value