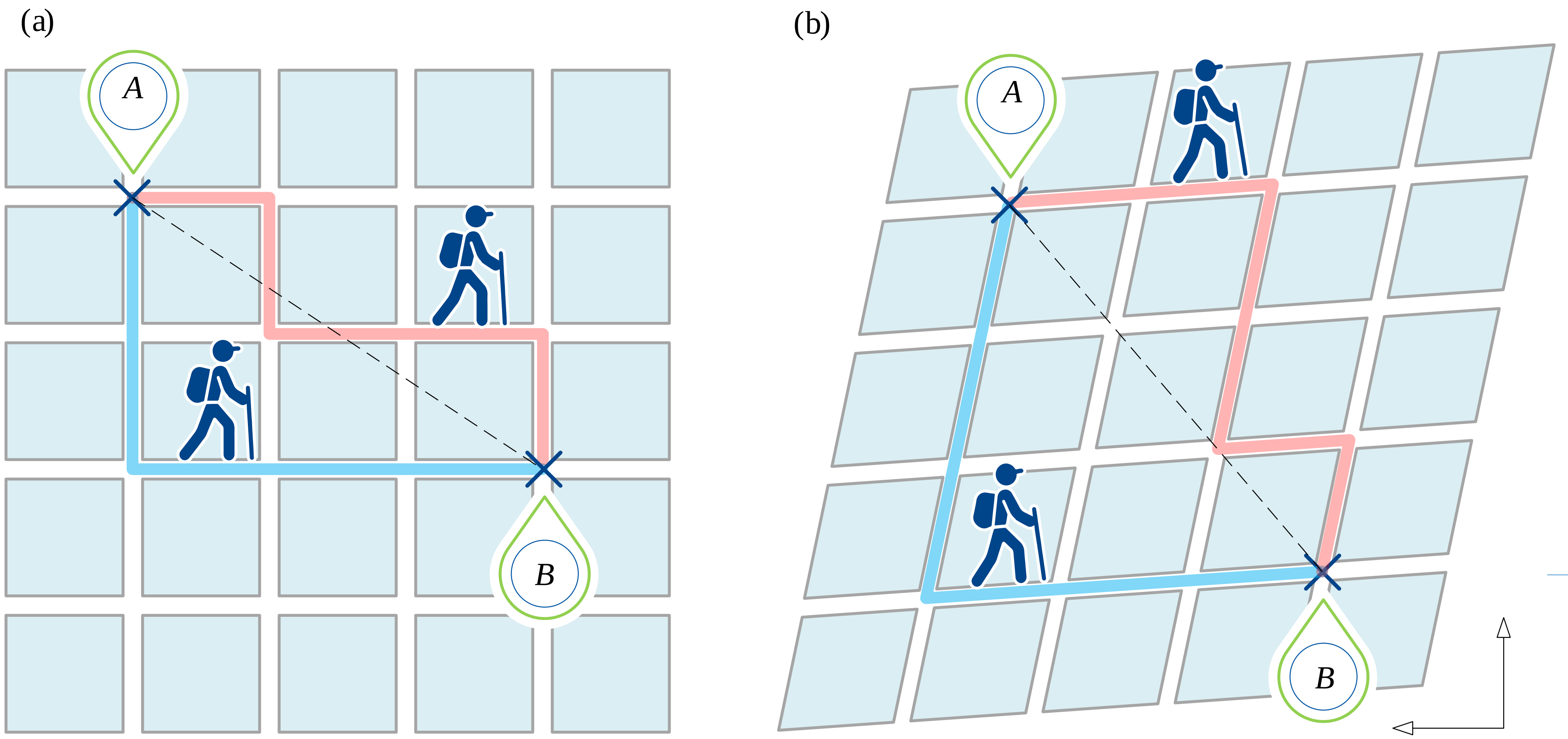
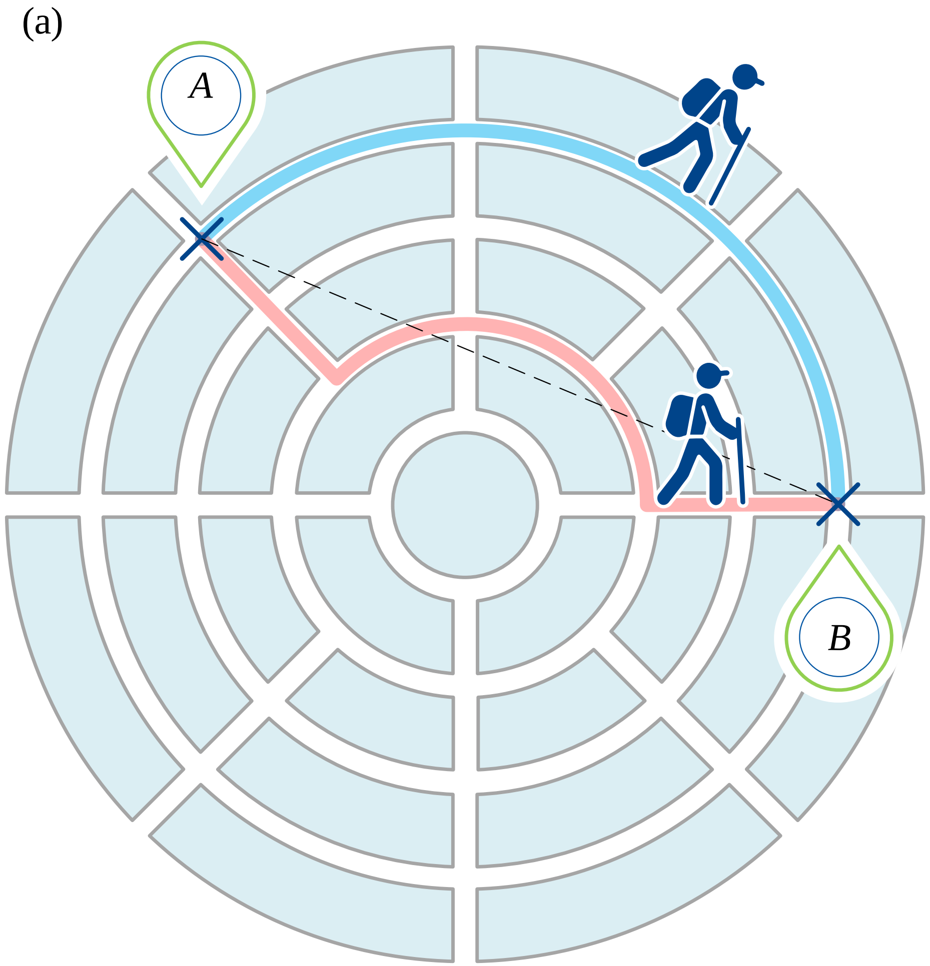
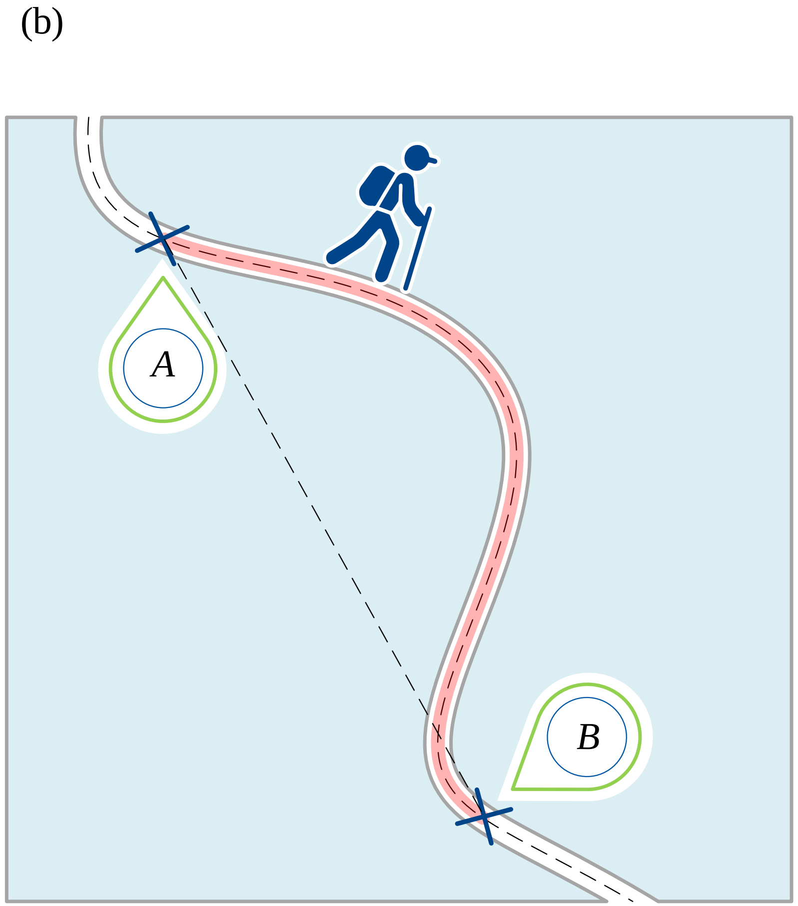
距离度量

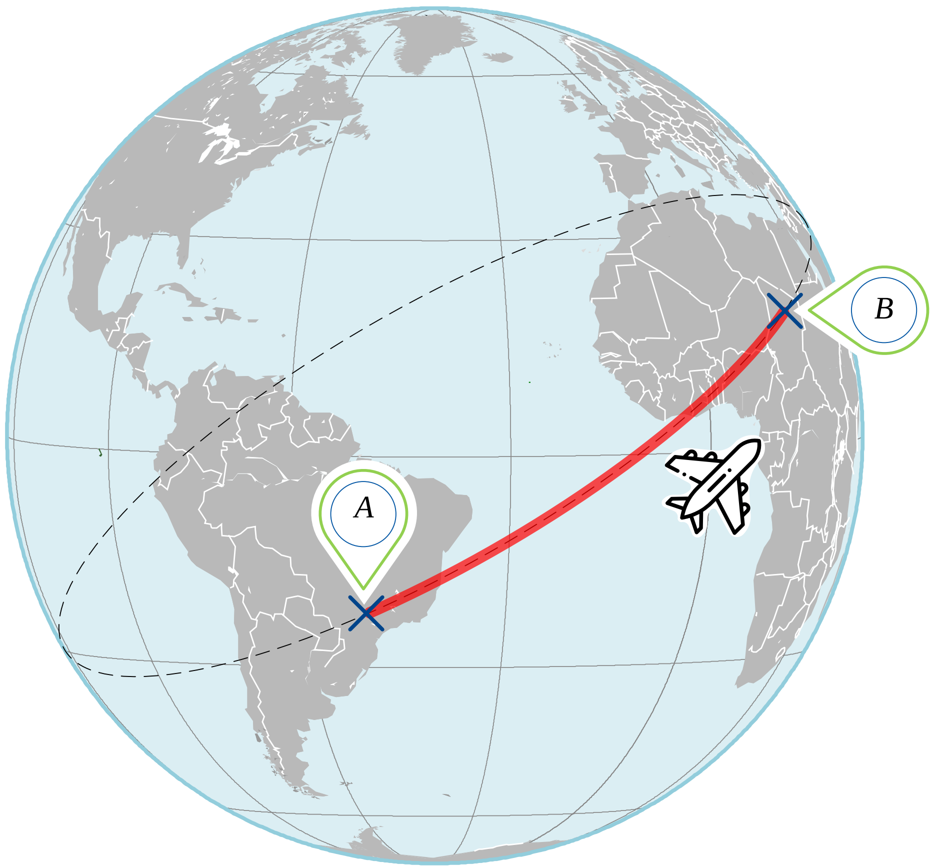


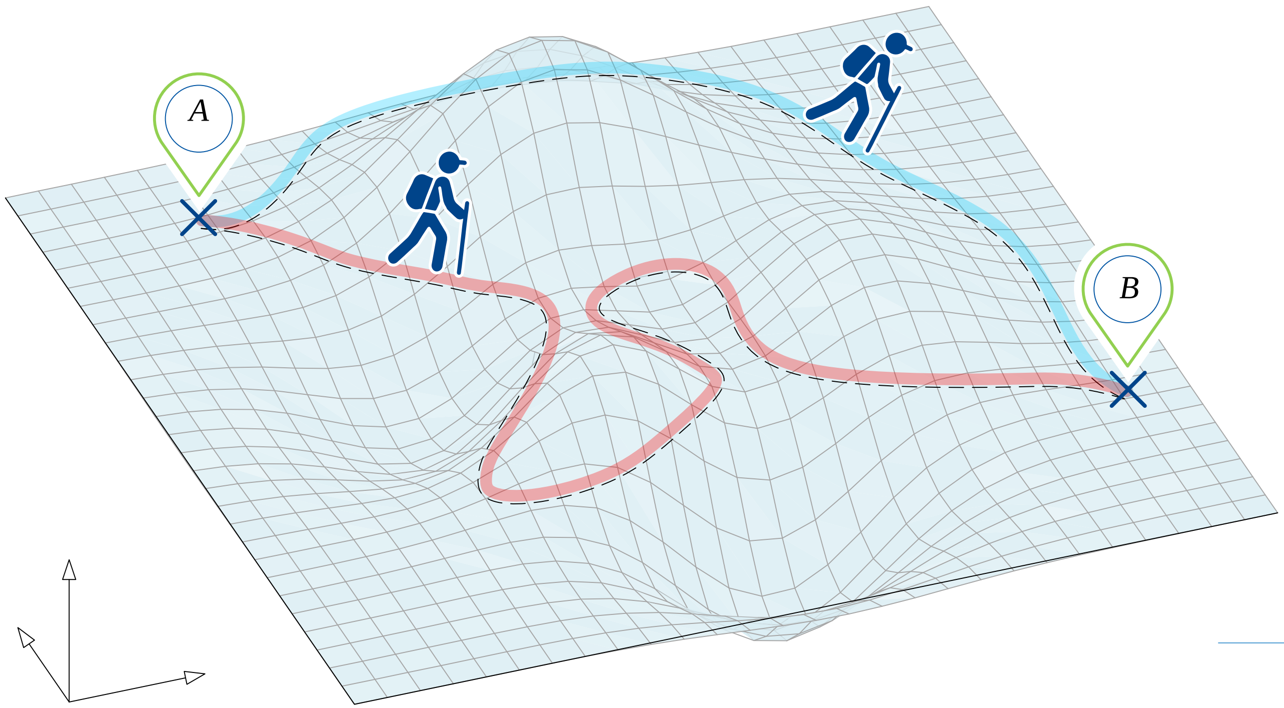
距离度量





距离度量

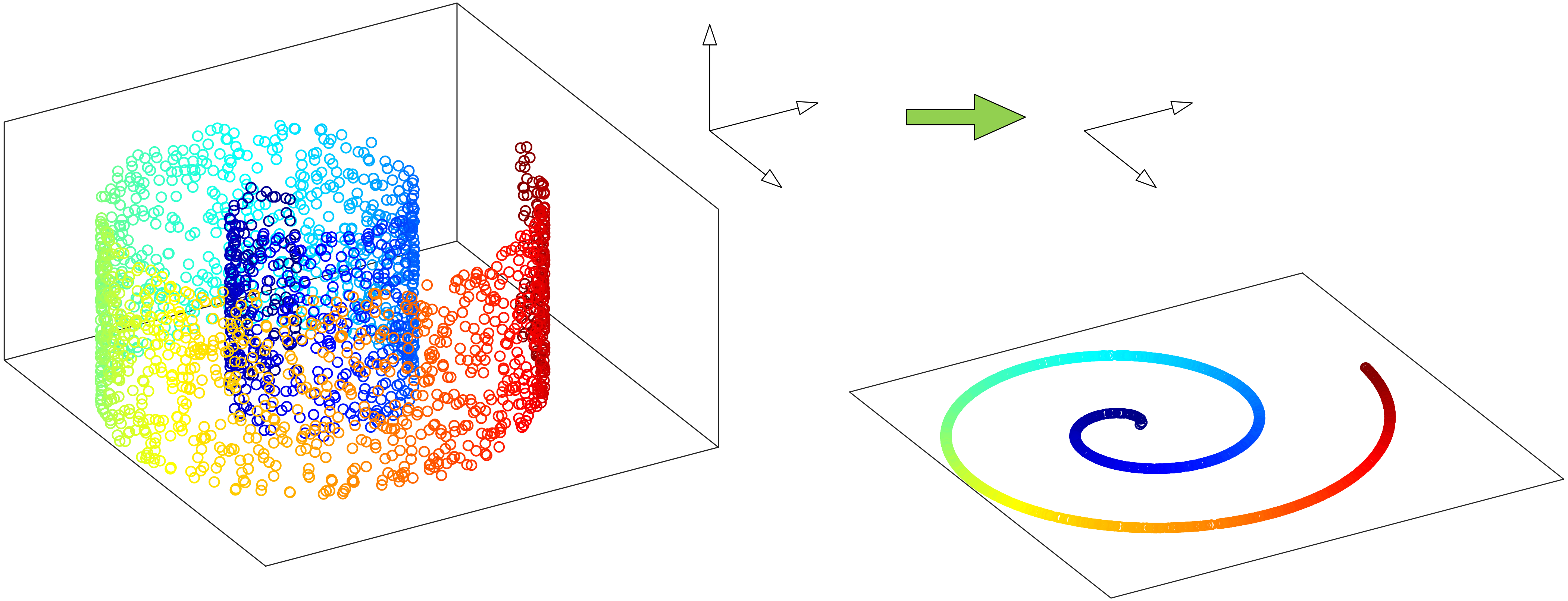




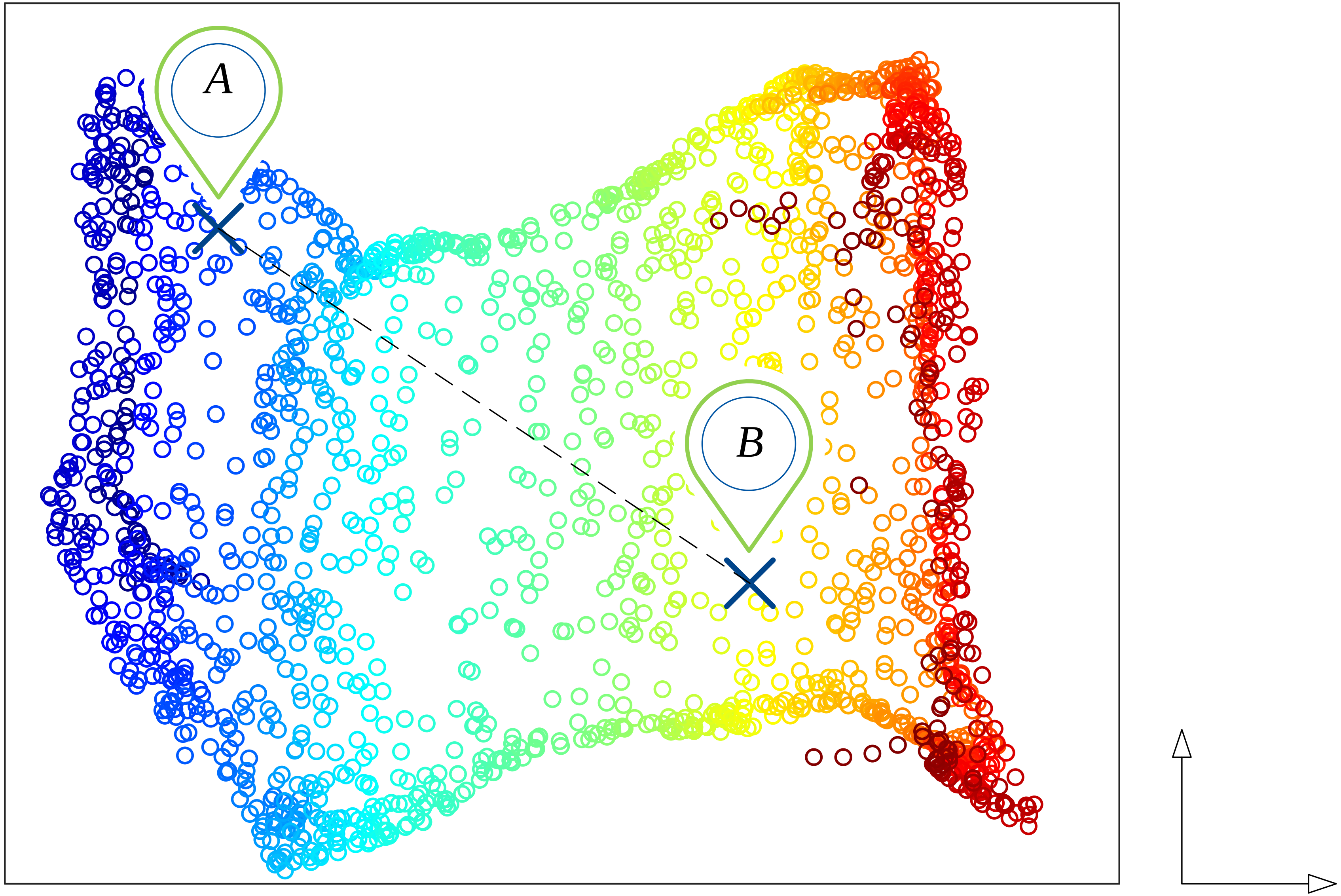
bilibili

知乎

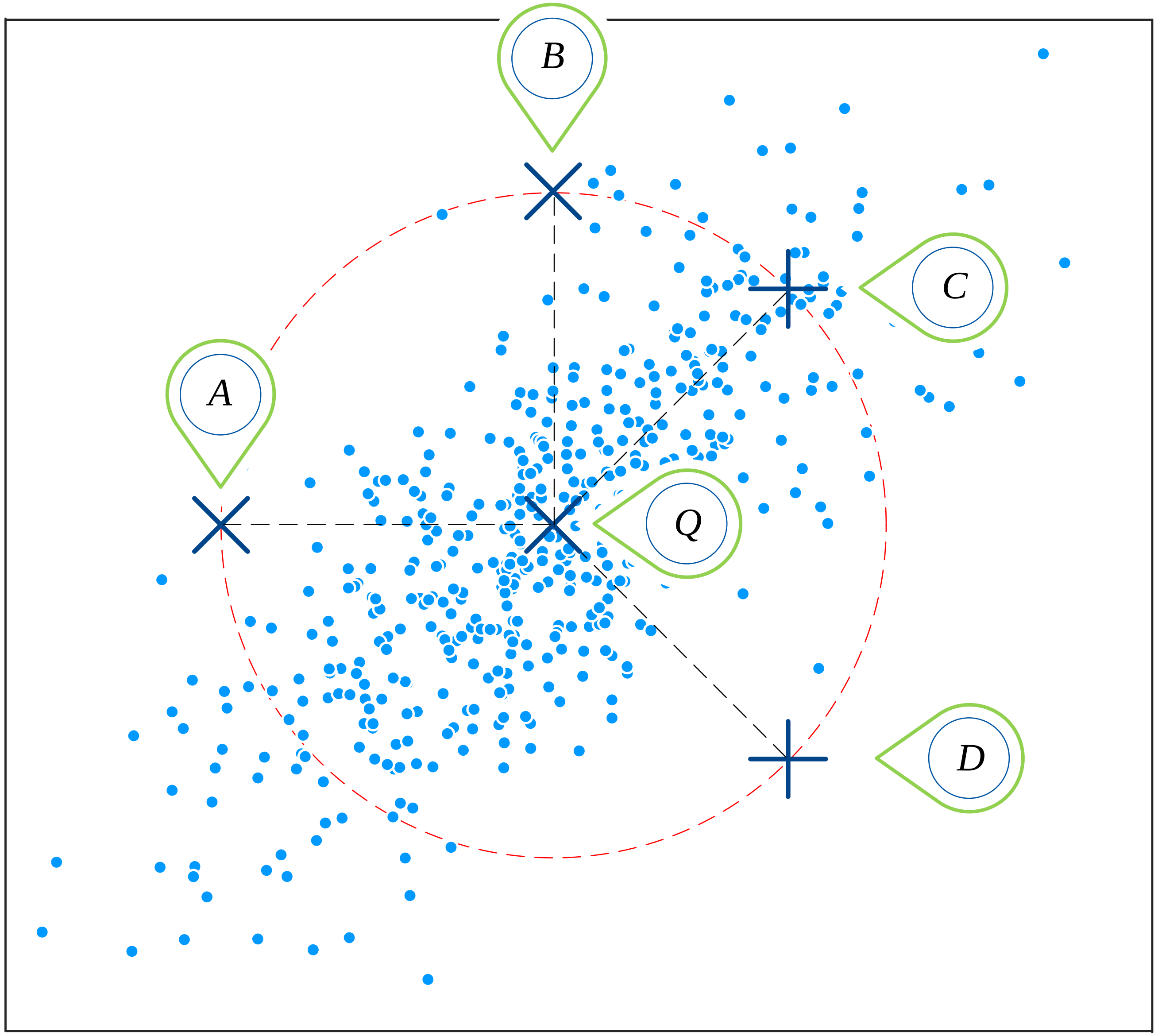
距离度量



距离度量



距离度量



欧氏距离

$$d\left(\boldsymbol{x},\boldsymbol{q}\right)=\mathrm{dist}\left(\boldsymbol{x},\boldsymbol{q}\right)=\sqrt{\left(\boldsymbol{x}-\boldsymbol{q}\right)^{\mathrm{T}}\left(\boldsymbol{x}-\boldsymbol{q}\right)}$$

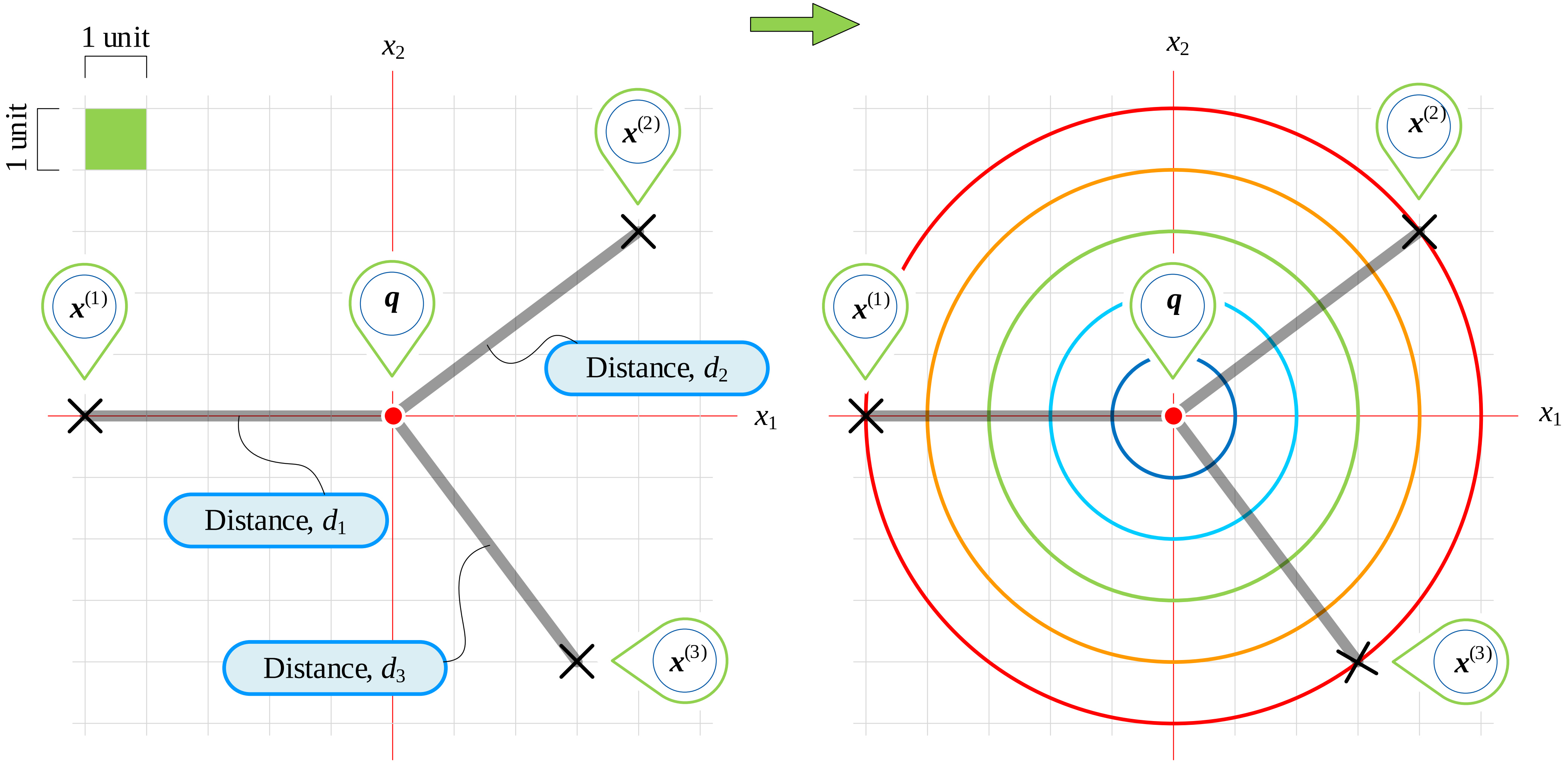
$$\begin{aligned}d\left(\boldsymbol{x},\boldsymbol{q}\right)&=\sqrt{\left(\begin{bmatrix}x_1 & x_2 & \cdots & x_D\end{bmatrix}-\begin{bmatrix}q_1 & q_2 & \cdots & q_D\end{bmatrix}\right)\left(\begin{bmatrix}x_1 & x_2 & \cdots & x_D\end{bmatrix}-\begin{bmatrix}q_1 & q_2 & \cdots & q_D\end{bmatrix}\right)^{\mathrm{T}}}\\&=\sqrt{\begin{bmatrix}x_1-q_1 & x_2-q_2 & \cdots & x_D-q_D\end{bmatrix}\begin{bmatrix}x_1-q_1 & x_2-q_2 & \cdots & x_D-q_D\end{bmatrix}^{\mathrm{T}}}\\&=\sqrt{\left(x_1-q_1\right)^2+\left(x_2-q_2\right)^2+\ldots+\left(x_D-q_D\right)^2}\end{aligned}$$

$$d\left(\boldsymbol{x},\boldsymbol{q}\right)=\sqrt{\left(x_1-q_1\right)^2+\left(x_2-q_2\right)^2}$$

bilibili

知乎

欧氏距离


1 unit
1 unit
x_2
x_1
$\boldsymbol{x}^{(1)}$
$\boldsymbol{x}^{(2)}$
$\boldsymbol{x}^{(3)}$
$\boldsymbol{q}$
Distance, d_1
Distance, d_2
Distance, d_3

欧氏距离


```python
from scipy.spatial import distance
import numpy as np

x_i = (0, 0, 0) # data point
q   = (4, 8, 6) # query point

# calculate Euclidean distance
dst_1 = distance.euclidean(x_i, q)

dst_2 = np.linalg.norm(np.array(x_i) - np.array(q))
```

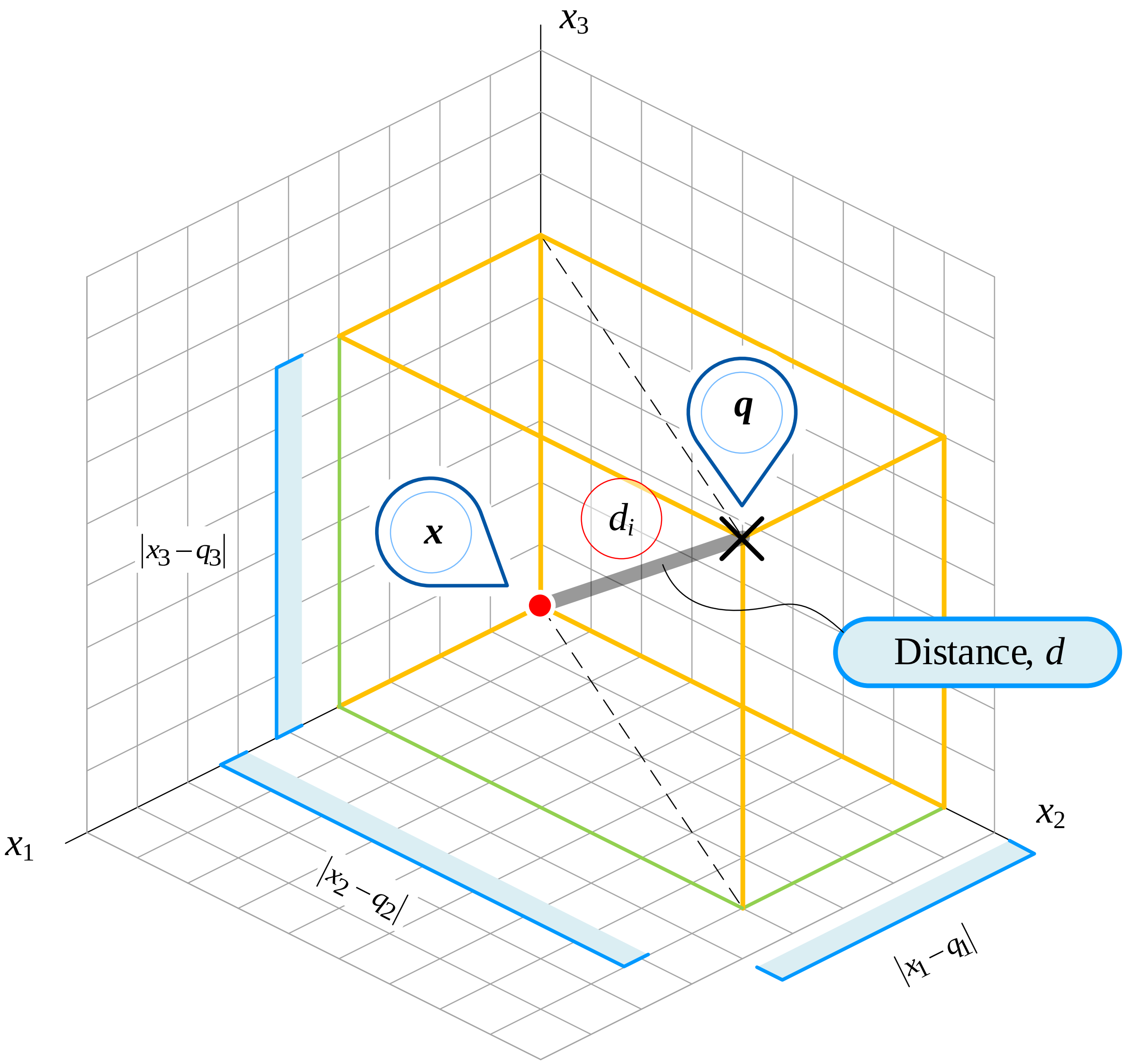
bilibili

生姜

知乎

生姜

欧氏距离



x_3
$|x_3 - q_3|$
$\boldsymbol{q}$
$\boldsymbol{x}$
d_i
Distance, d
x_1
x_2
$|x_2 - q_2|$
$|x_1 - q_1|$

欧氏距离


```python
from sklearn.metrics.pairwise import euclidean_distances

# Sample data points
X = [[-5, 0], [4, 3], [3, -4]]

# Query point
q = [[0, 0]]

# pairwise distances between rows of X and q
dst_pairwise_X_q = euclidean_distances(X, q)
print('Pairwise distances between X and q')
print(dst_pairwise_X_q)

# pairwise distances between rows of X and itself
dst_pairwise_X_X = euclidean_distances(X, X)
print('Pairwise distances between X and X')
print(dst_pairwise_X_X)
```

生姜

标准化欧氏距离

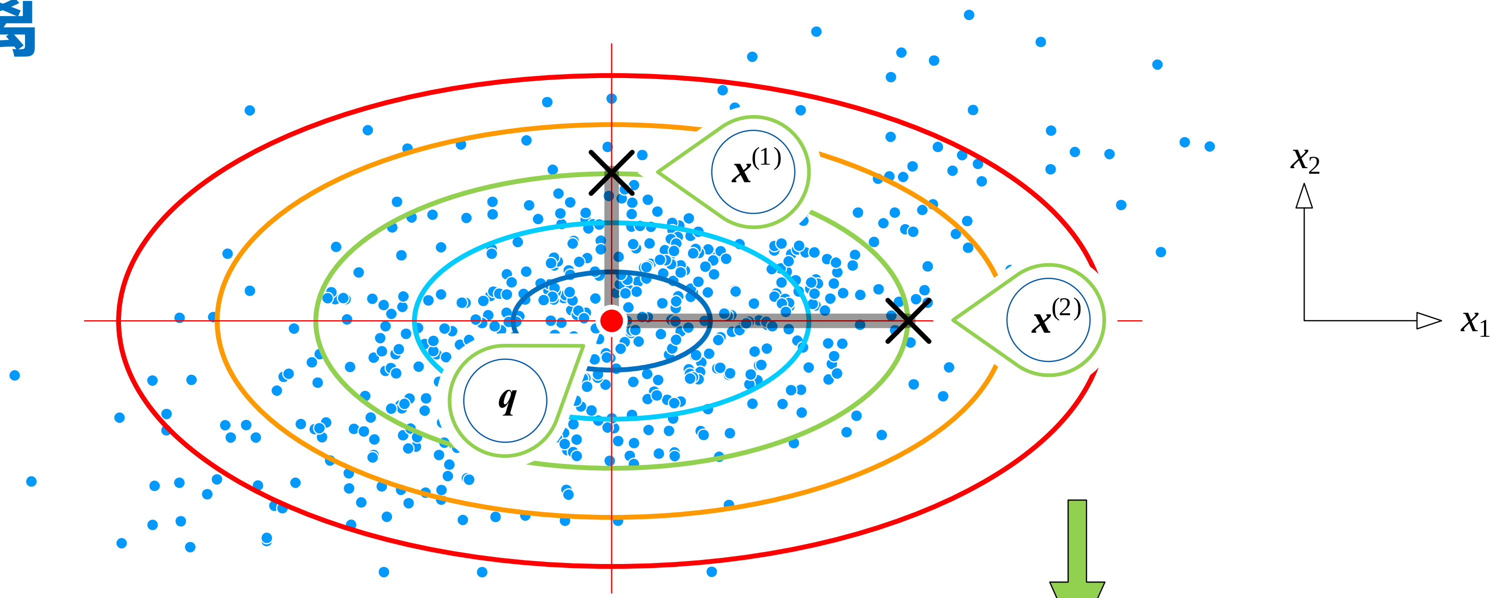
$$d(\boldsymbol{x},\boldsymbol{q}) = \sqrt{(\boldsymbol{x}-\boldsymbol{q})^{\mathrm{T}} \boldsymbol{V}^{-1} (\boldsymbol{x}-\boldsymbol{q})}$$

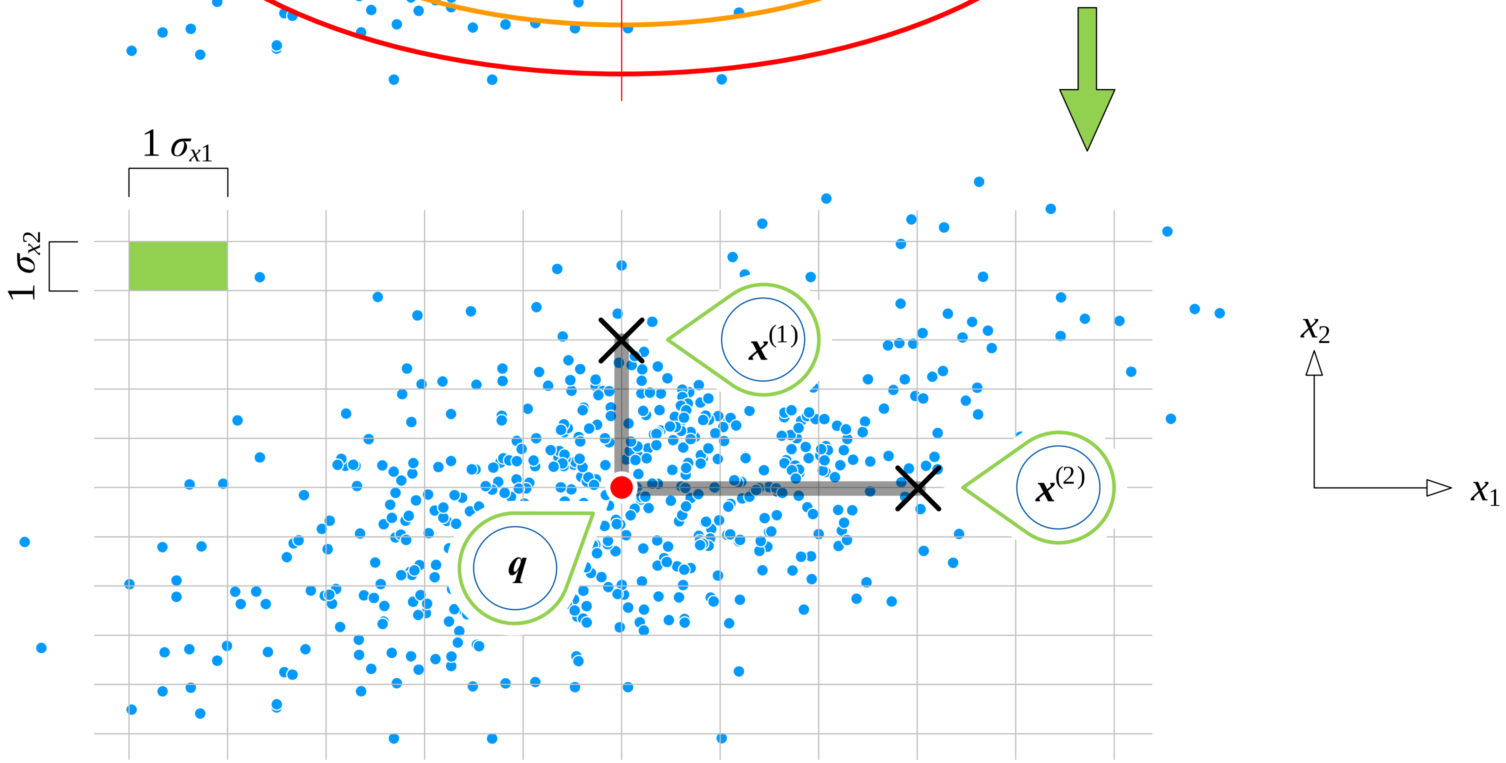
$$\boldsymbol{V} = \operatorname{diag}\left(\operatorname{diag}\left(\boldsymbol{\Sigma}\right)\right) = \operatorname{diag}\left(\operatorname{diag}\begin{bmatrix} \sigma_1^2 & \rho_{1,2}\sigma_1\sigma_2 & \cdots & \rho_{1,D}\sigma_1\sigma_D \\ \rho_{1,2}\sigma_1\sigma_2 & \sigma_2^2 & \cdots & \rho_{2,D}\sigma_2\sigma_D \\ \vdots & \vdots & \ddots & \vdots \\ \rho_{1,D}\sigma_1\sigma_D & \rho_{2,D}\sigma_2\sigma_D & \cdots & \sigma_D^2 \end{bmatrix}\right) = \begin{bmatrix} \sigma_1^2 & & & \\ & \sigma_2^2 & & \\ & & \ddots & \\ & & & \sigma_D^2 \end{bmatrix}$$

标准化欧氏距离

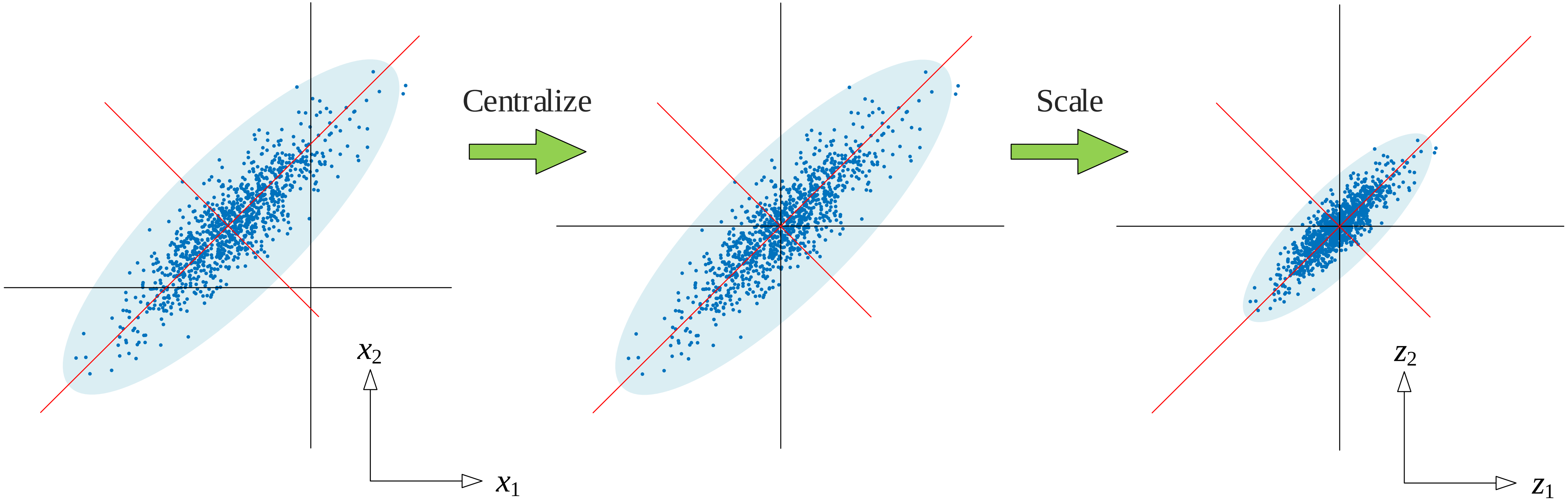
$$
\begin{aligned}
d(\boldsymbol{x},\boldsymbol{q}) &= \sqrt{\begin{bmatrix} x_1-q_1 & x_2-q_2 & \cdots & x_D-q_D \end{bmatrix}\begin{bmatrix} \sigma_1^2 & & & \\ & \sigma_2^2 & & \\ & & \ddots & \\ & & & \sigma_D^2 \end{bmatrix}^{-1}\begin{bmatrix} x_1-q_1 & x_2-q_2 & \cdots & x_D-q_D \end{bmatrix}^{\mathrm{T}}} \\
&= \sqrt{\frac{(x_1-q_1)^2}{\sigma_1^2}+\frac{(x_2-q_2)^2}{\sigma_2^2}+\dots+\frac{(x_D-q_D)^2}{\sigma_D^2}} \\
&= \sqrt{\sum_{j=1}^{D}\frac{(x_j-q_j)^2}{\sigma_j^2}}
\end{aligned}
$$

标准化欧氏距离





标准化欧氏距离



标准化欧氏距离


```python
from scipy.spatial import distance
import numpy as np

# Variance-covariance matrix
SIGMA = np.array([[2, 1], [1, 2]])

q   = [0, 0];          # query point
x_1 = [-3.5, -4];    # data point 1
x_2 = [2.75, -1.5]; # data point 1

# Calculate standardized Euclidean distances

d_1 = distance.seuclidean(q, x_1, np.diag(SIGMA))
d_2 = distance.seuclidean(q, x_2, np.diag(SIGMA))

# Note1: V is an 1-D array of component variances
```

马氏距离

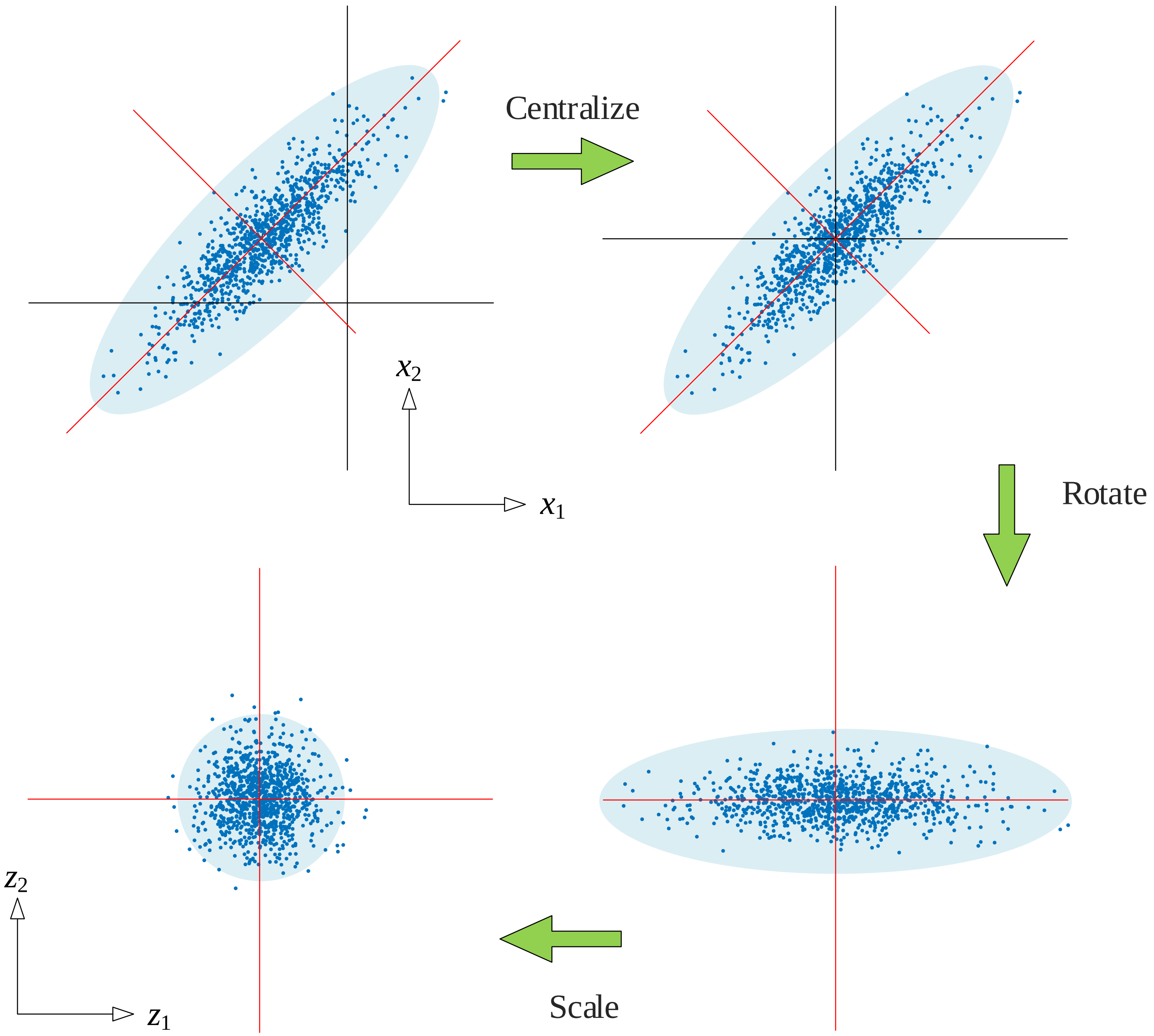
$$d\left(\boldsymbol{x},\boldsymbol{q}\right)=\sqrt{\left(\boldsymbol{x}-\boldsymbol{q}\right)^{\mathrm{T}}\boldsymbol{\Sigma}^{-1}\left(\boldsymbol{x}-\boldsymbol{q}\right)}$$

$$\begin{aligned}
d\left(\boldsymbol{x},\boldsymbol{q}\right)^{2} &= \left(\boldsymbol{x}-\boldsymbol{q}\right)^{\mathrm{T}}\boldsymbol{\Sigma}^{-1}\left(\boldsymbol{x}-\boldsymbol{q}\right)=\left(\boldsymbol{x}-\boldsymbol{q}\right)^{\mathrm{T}}\boldsymbol{\Sigma}^{\frac{-1}{2}}\boldsymbol{\Sigma}^{\frac{-1}{2}}\left(\boldsymbol{x}-\boldsymbol{q}\right)\\
&=\left[\boldsymbol{\Sigma}^{\frac{-1}{2}}\left(\boldsymbol{x}-\boldsymbol{q}\right)\right]^{\mathrm{T}}\left[\boldsymbol{\Sigma}^{\frac{-1}{2}}\left(\boldsymbol{x}-\boldsymbol{q}\right)\right]\\
&=\left[\underset{\text{Scale}}{\boldsymbol{\Lambda}^{\frac{-1}{2}}}\ \underset{\text{Rotate}}{\boldsymbol{V}}\underset{\text{Centralize}}{\left(\boldsymbol{x}-\boldsymbol{q}\right)}\right]^{\mathrm{T}}\left[\boldsymbol{\Lambda}^{\frac{-1}{2}}\boldsymbol{V}\left(\boldsymbol{x}-\boldsymbol{q}\right)\right]
\end{aligned}$$

bilibili 生姜

知乎 生姜

马氏距离



马氏距离

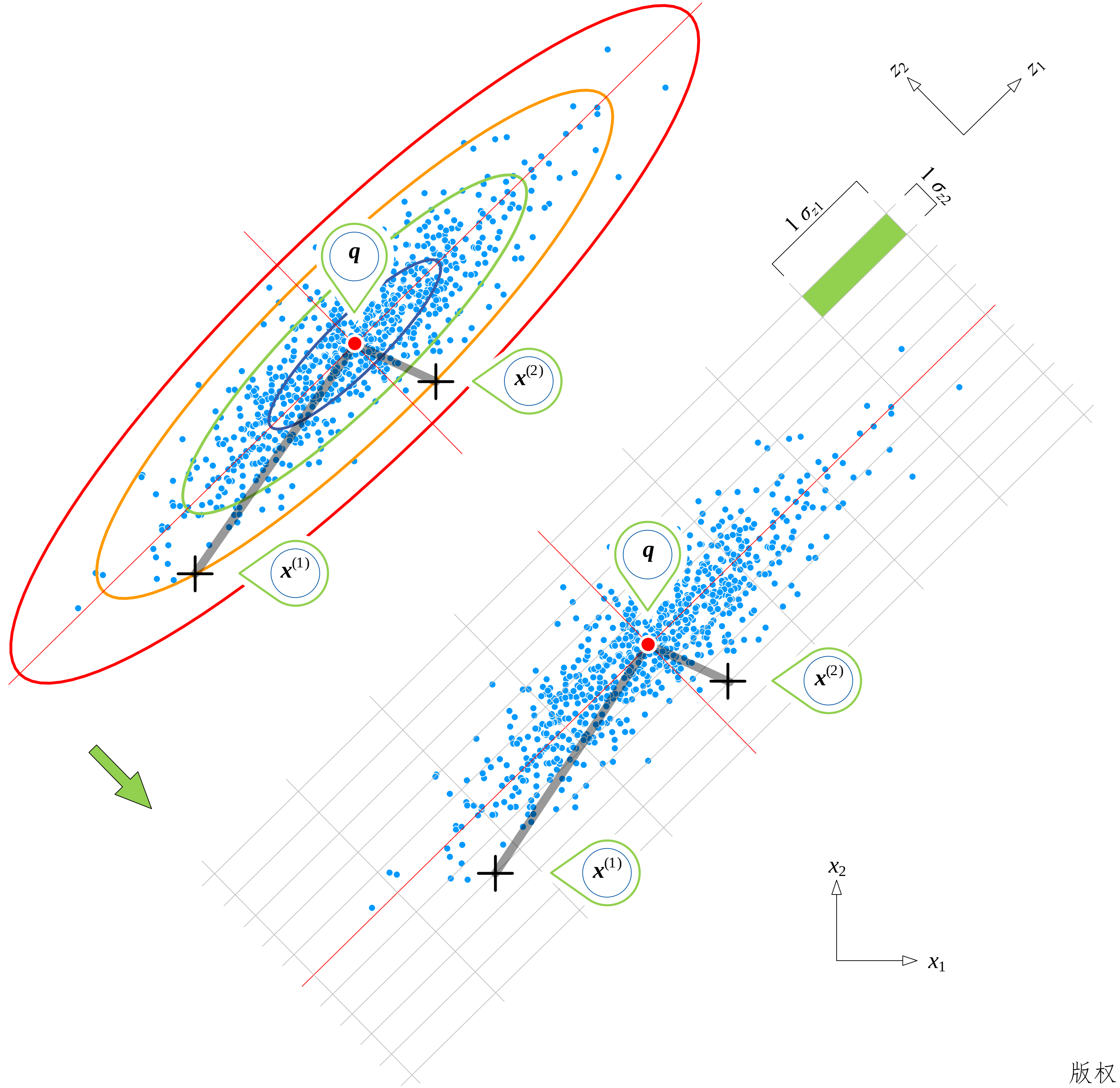
$$\boldsymbol{x}^{(1)} = \begin{bmatrix} -3.5 & -4 \end{bmatrix}^{\mathrm{T}}, \quad \boldsymbol{x}^{(2)} = \begin{bmatrix} 2.75 & -1.5 \end{bmatrix}^{\mathrm{T}}$$

$$\boldsymbol{\Sigma} = \begin{bmatrix} 2 & 1 \\ 1 & 2 \end{bmatrix}$$

$$\begin{aligned} d_1 &= \sqrt{\left(\begin{bmatrix} -3.5 & -4 \end{bmatrix} - \begin{bmatrix} 0 & 0 \end{bmatrix}\right)\begin{bmatrix} 2 & 1 \\ 1 & 2 \end{bmatrix}^{-1}\left(\begin{bmatrix} -3.5 & -4 \end{bmatrix} - \begin{bmatrix} 0 & 0 \end{bmatrix}\right)^{\mathrm{T}}} \\ &= \sqrt{\begin{bmatrix} -3.5 & -4 \end{bmatrix} \cdot \frac{1}{3} \cdot \begin{bmatrix} 2 & -1 \\ -1 & 2 \end{bmatrix}\begin{bmatrix} -3.5 & -4 \end{bmatrix}^{\mathrm{T}}} = 3.08 \end{aligned}$$

$$\begin{aligned} d_2 &= \sqrt{\left(\begin{bmatrix} 2.75 & -1.5 \end{bmatrix} - \begin{bmatrix} 0 & 0 \end{bmatrix}\right)\begin{bmatrix} 2 & 1 \\ 1 & 2 \end{bmatrix}^{-1}\left(\begin{bmatrix} 2.75 & -1.5 \end{bmatrix} - \begin{bmatrix} 0 & 0 \end{bmatrix}\right)^{\mathrm{T}}} \\ &= \sqrt{\begin{bmatrix} 2.75 & -1.5 \end{bmatrix} \cdot \frac{1}{3} \cdot \begin{bmatrix} 2 & -1 \\ -1 & 2 \end{bmatrix}\begin{bmatrix} 2.75 & -1.5 \end{bmatrix}^{\mathrm{T}}} = 3.05 \end{aligned}$$

马氏距离



马氏距离


```python
from scipy.spatial import distance
import numpy as np
from numpy.linalg import inv

# Variance-covariance matrix
SIGMA = np.array([[2, 1], [1, 2]])

q   = [0, 0];         # query point
x_1 = [-3.5, -4];     # data point 1
x_2 = [2.75, -1.5]; # data point 1

# Calculate Mahal distances

d_1 = distance.mahalanobis(q, x_1, inv(SIGMA))
d_2 = distance.mahalanobis(q, x_2, inv(SIGMA))

# Note1: the output of the function is Mahal distance, not Mahal distance squared
# Note2: input is the inverse of the covariance matrix
```

城市街区距离

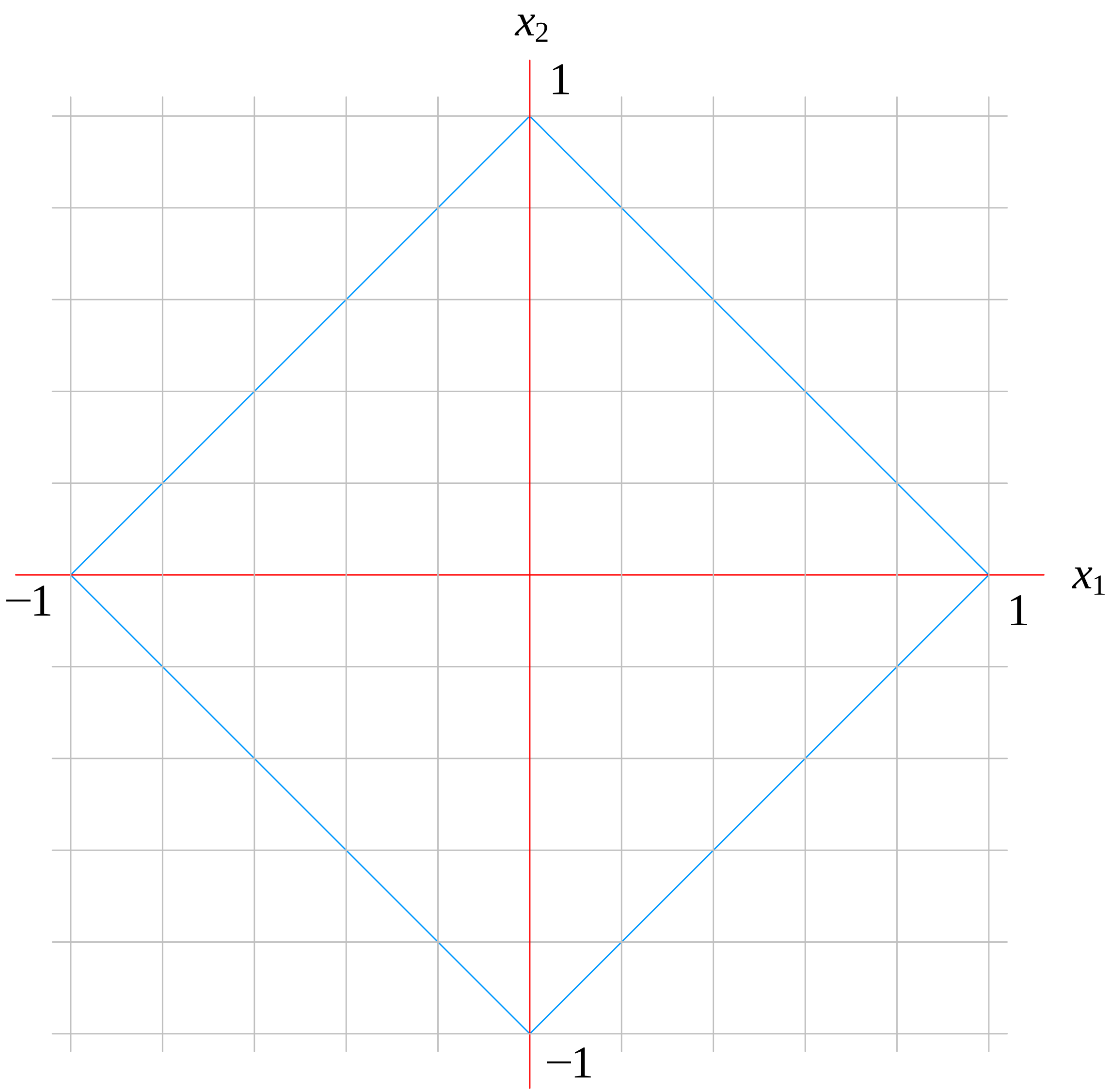
$$d(\boldsymbol{x},\boldsymbol{q})=\sum_{j=1}^{D}\left|x_j-q_j\right|$$

$$d(\boldsymbol{x},\boldsymbol{q})=\left|x_1-q_1\right|+\left|x_2-q_2\right|+\ldots+\left|x_D-q_D\right|$$

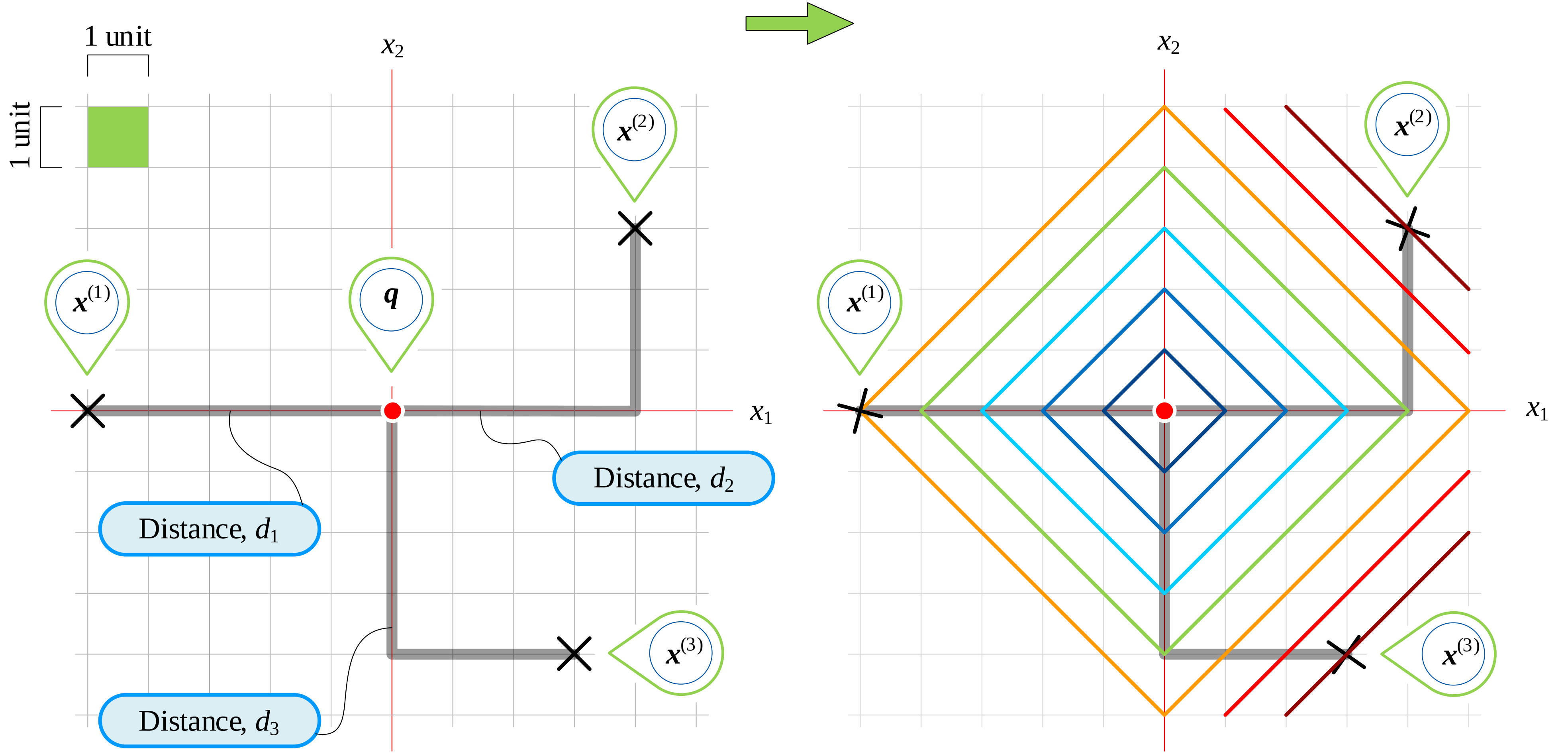
$$d(\boldsymbol{x},\boldsymbol{q})=\left|x_1-q_1\right|+\left|x_2-q_2\right|$$

城市街区距离

$$\left|x_1\right|+\left|x_2\right|=1$$



城市街区距离


1 unit
1 unit
x_2
x_1
$x^{(1)}$
$x^{(2)}$
$x^{(3)}$
q
Distance, d_1
Distance, d_2
Distance, d_3
x_2
x_1
$x^{(1)}$
$x^{(2)}$
$x^{(3)}$

城市街区距离


```python
from scipy.spatial import distance
from sklearn.metrics import pairwise_distances

# Sample data points
X = [[-5, 0], [4, 3], [3, -4]]

# Query point
q = [[0, 0]]

# Compute the City Block (Manhattan) distance.
d_1 = distance.cityblock(q, X[0])
d_2 = distance.cityblock(q, X[1])
d_3 = distance.cityblock(q, X[2])

# pairwise distances between rows of X and q
dst_pairwise_X_q = pairwise_distances(X, q, metric='cityblock')
print('Pairwise City Block distances between X and q')
print(dst_pairwise_X_q)
```

切比雪夫距离

$$d(\boldsymbol{x},\boldsymbol{q}) = \max_j \left\{ \left| x_j - q_j \right| \right\}$$

$$d(\boldsymbol{x},\boldsymbol{q}) = \max \left\{ \left| x_1 - q_1 \right|,\ \ \left| x_2 - q_2 \right|,\ \ \ldots,\ \ \left| x_D - q_D \right| \right\}$$

$$d(\boldsymbol{x},\boldsymbol{q}) = \max \left\{ \left| x_1 - q_1 \right|,\ \ \left| x_2 - q_2 \right| \right\}$$

bilibili

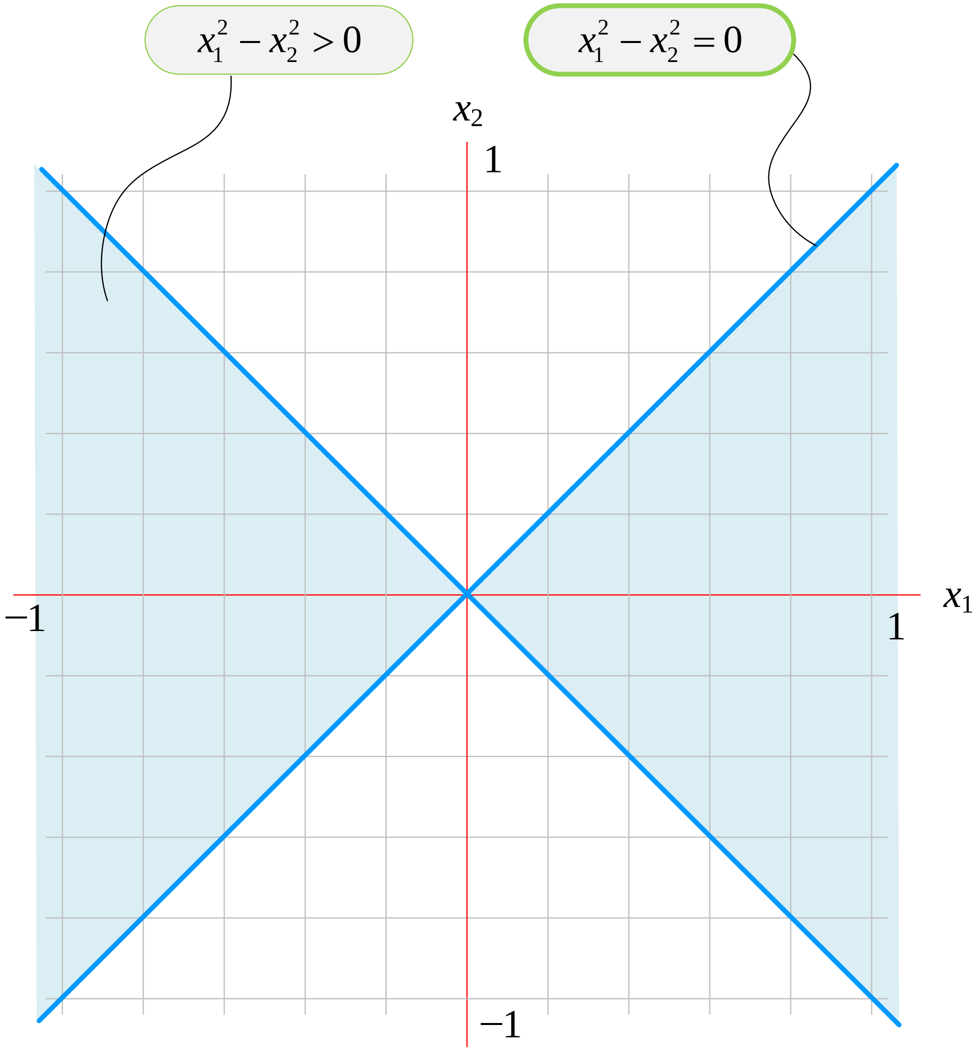
知乎

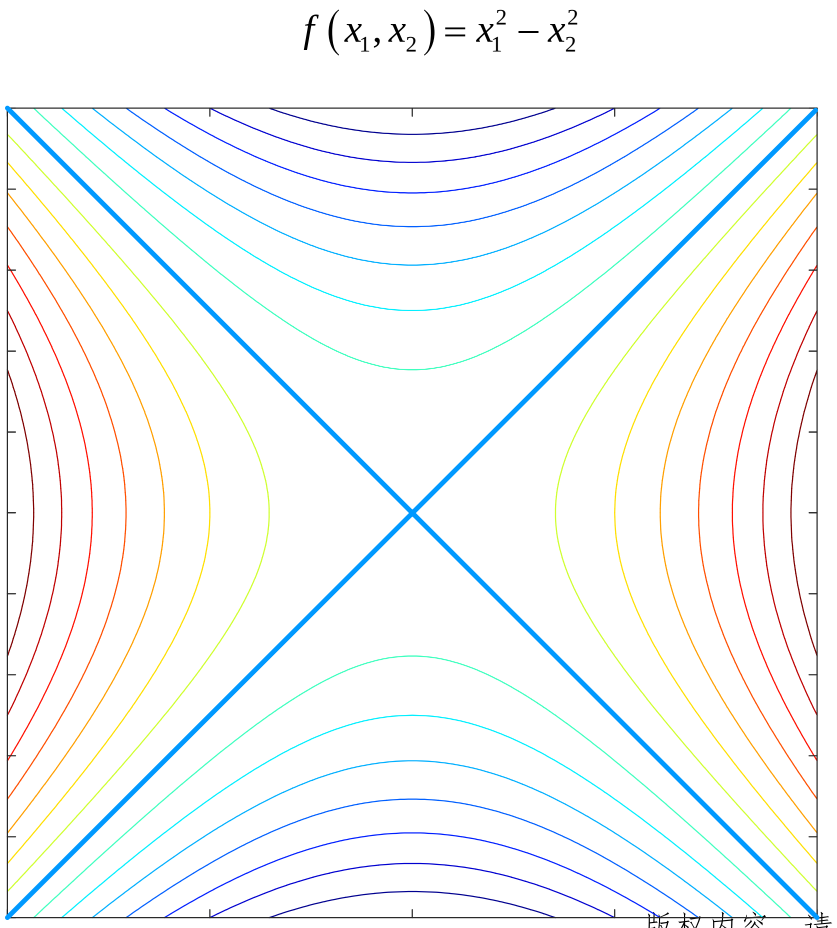
切比雪夫距离

$$\max\{|x_1|,\ |x_2|\}=1$$

$$\begin{cases} |x_1|=1 & |x_1|>|x_2| \\ |x_2|=1 & |x_2|>|x_1| \end{cases}$$

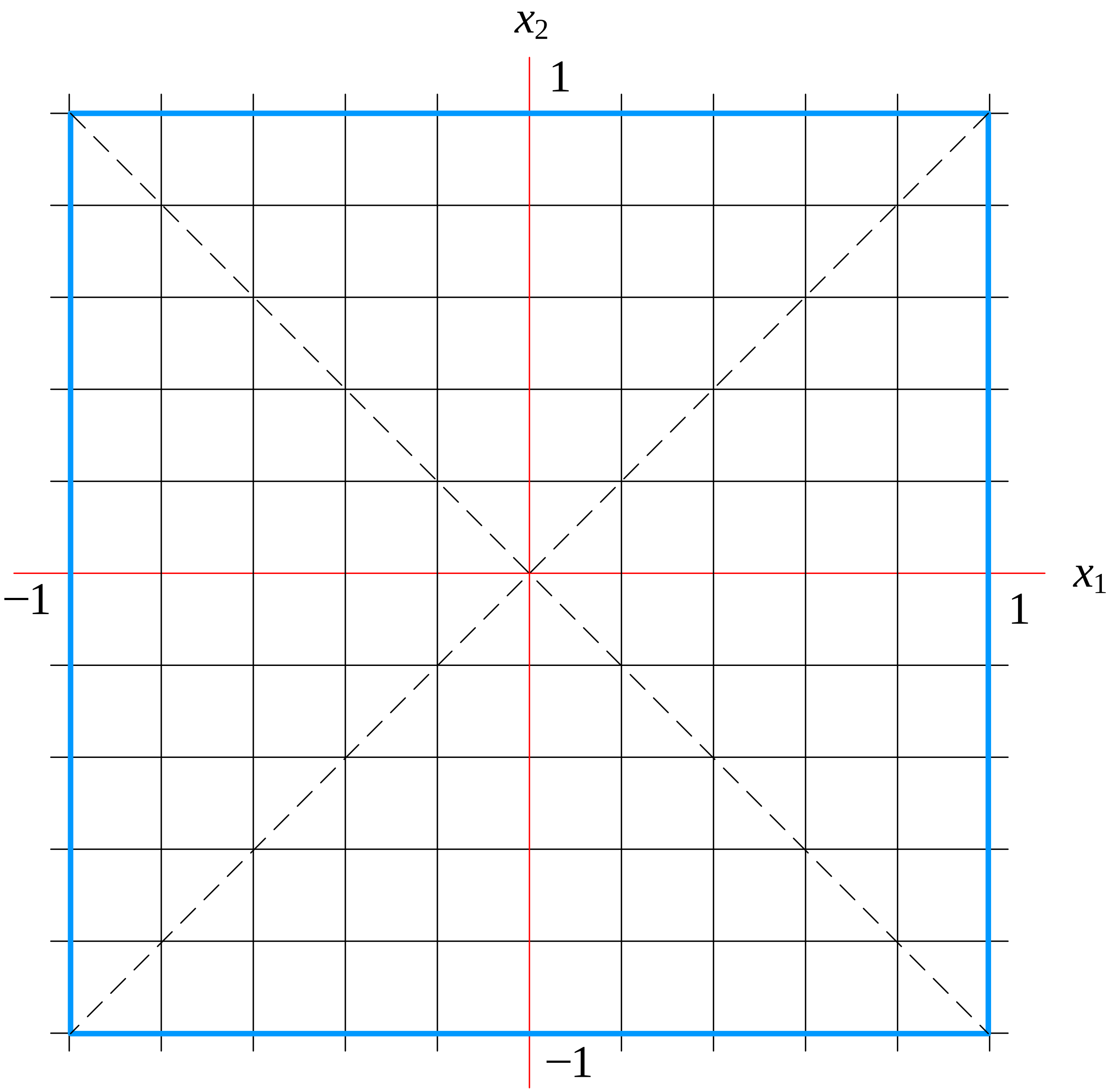
$$x_1^2-x_2^2>0$$



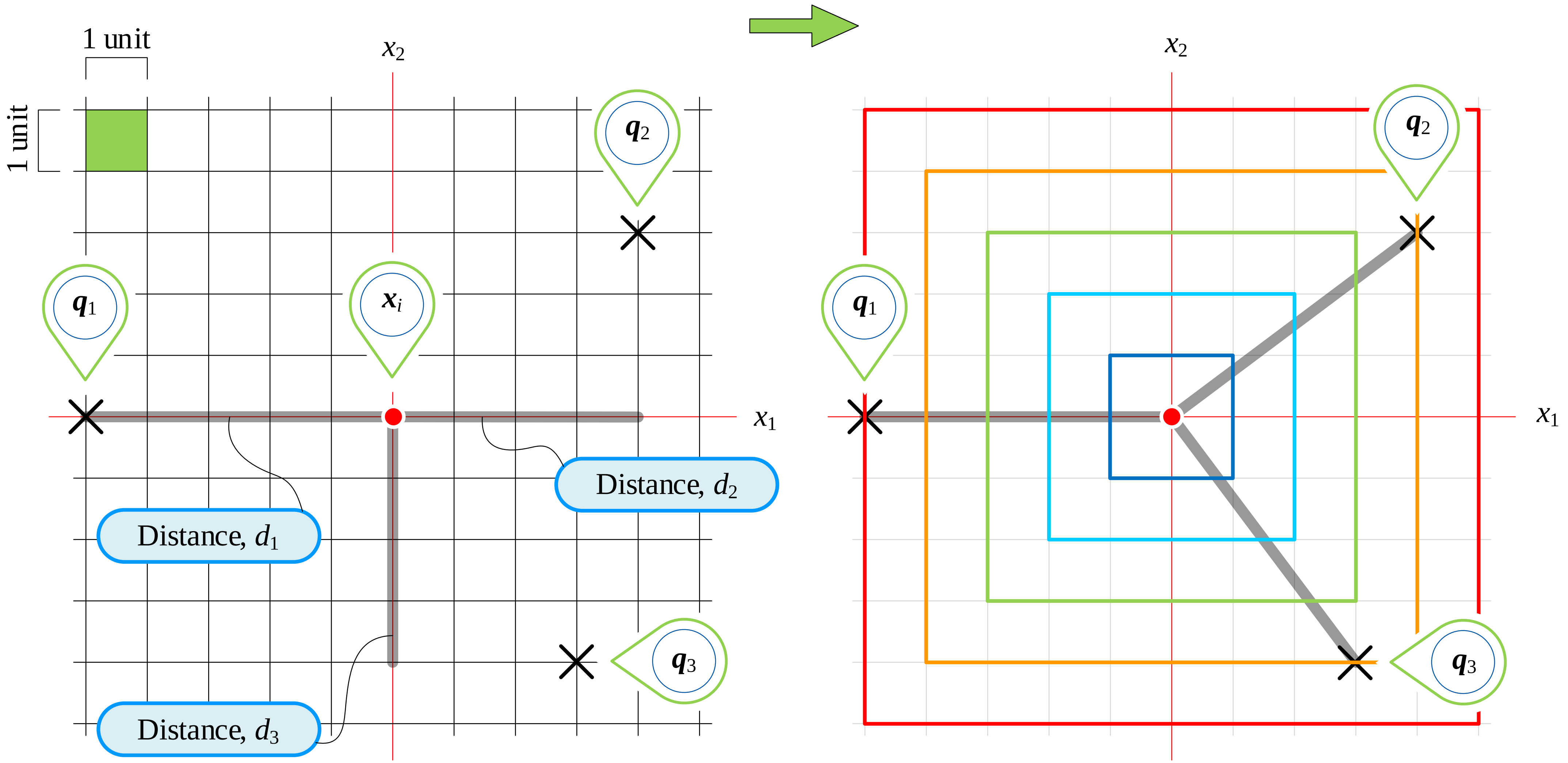


切比雪夫距离

$$\begin{cases} x_1 = \pm 1 & x_1^2 - x_2^2 > 0 \\ x_2 = \pm 1 & x_1^2 - x_2^2 < 0 \end{cases}$$



切比雪夫距离


1 unit
1 unit
x_2
x_1
q_1
q_2
q_3
x_i
Distance, d_1
Distance, d_2
Distance, d_3
x_2
x_1
q_1
q_2
q_3

切比雪夫距离


```
from scipy.spatial import distance
from sklearn.metrics import pairwise_distances

# Sample data points
X = [[-5, 0], [4, 3], [3, -4]]

# Query point
q = [[0, 0]]

# Compute the Chebyshev distance.
d_1 = distance.chebyshev(q, X[0])
d_2 = distance.chebyshev(q, X[1])
d_3 = distance.chebyshev(q, X[2])

# pairwise distances between rows of X and q
dst_pairwise_X_q = pairwise_distances(X, q, metric='chebyshev')
print('Pairwise Chebyshev distances between X and q')
print(dst_pairwise_X_q)
```

闵氏距离

$$d\left(\boldsymbol{x},\boldsymbol{q}\right)=\left(\sum_{j=1}^{D}\left|x_j-q_j\right|^p\right)^{1/p}$$

bilibili

知乎

闵氏距离