DSC 140B

Representation Learning

Lecture 05 | Part 1

Change of Basis Matrices

Changing Basis

- Suppose $\vec{x} = \begin{pmatrix} a_1 \\ a_2 \end{pmatrix} = a_1 \hat{e}^{(1)} + a_2 \hat{e}^{(2)}$.
- $\hat{u}^{(1)}$ and $\hat{u}^{(2)}$ form a new, **orthonormal** basis $\mathcal{U}$.
- What is $[\vec{x}]_{\mathcal{U}}$?
- That is, what are b_1 and b_2 in $\vec{x} = b_1 \hat{u}^{(1)} + b_2 \hat{u}^{(2)}$.

Exercise

Find the coordinates of $\vec{x}$ in the new basis:

$$\hat{u}^{(1)} = (\sqrt{3}/2, 1/2)^T$$
$$\hat{u}^{(2)} = (-1/2, \sqrt{3}/2)^T$$
$$\vec{x} = (1/2, 1)^T$$

Change of Basis

- Suppose $\hat{u}^{(1)}$ and $\hat{u}^{(2)}$ are our new, **orthonormal** basis vectors.
- We know $\vec{x} = x_1 \hat{e}^{(1)} + x_2 \hat{e}^{(2)}$
- We want to write $\vec{x} = b_1 \hat{u}^{(1)} + b_2 \hat{u}^{(2)}$
- Solution

$$b_1 = \vec{x} \cdot \hat{u}^{(1)} \qquad b_2 = \vec{x} \cdot \hat{u}^{(2)}$$

Change of Basis Matrix

- Changing basis is a linear transformation

$$\vec{f}(\vec{x}) = (\vec{x} \cdot \hat{u}^{(1)})\hat{u}^{(1)} + (\vec{x} \cdot \hat{u}^{(2)})\hat{u}^{(2)} = \begin{pmatrix} \vec{x} \cdot \hat{u}^{(1)} \\ \vec{x} \cdot \hat{u}^{(2)} \end{pmatrix}_{\mathcal{U}}$$

- We can represent it with a matrix

$$\begin{pmatrix} \uparrow & \uparrow \\ f(\hat{e}^{(1)}) & f(\hat{e}^{(2)}) \\ \downarrow & \downarrow \end{pmatrix}$$

Example

$$\begin{aligned}\hat{u}^{(1)} &= (\sqrt{3}/2, 1/2)^T \\ \hat{u}^{(2)} &= (-1/2, \sqrt{3}/2)^T \\ f(\hat{e}^{(1)}) &= \\ f(\hat{e}^{(2)}) &= \\ A &= \end{aligned}$$

Observation

- The new basis vectors become the **rows** of the matrix.

Example

- Multiplying by this matrix gives the coordinate vector w.r.t. the new basis.

$$\hat{u}^{(1)} = (\sqrt{3}/2, 1/2)^T$$

$$\hat{u}^{(2)} = (-1/2, \sqrt{3}/2)^T$$

$$A = \begin{pmatrix} \sqrt{3}/2 & 1/2 \\ -1/2 & \sqrt{3}/2 \end{pmatrix}$$

$$\vec{x} = (1/2, 1)^T$$

Change of Basis Matrix

- Let $\hat{u}^{(1)}, \ldots, \hat{u}^{(d)}$ form an orthonormal basis $\mathcal{U}$.
- The matrix U whose **rows** are the new basis vectors is the **change of basis** matrix from the standard basis to $\mathcal{U}$:

$$U = \begin{pmatrix} \leftarrow & \hat{u}^{(1)} & \rightarrow \\ \leftarrow & \hat{u}^{(2)} & \rightarrow \\ & \vdots & \\ \leftarrow & \hat{u}^{(d)} & \rightarrow \end{pmatrix}$$

Change of Basis Matrix

- If U is the change of basis matrix, $[\vec{x}]_{\mathcal{U}} = U\vec{x}$
- To go *back* to the standard basis, use U^T:

$$\vec{x} = U^T[\vec{x}]_{\mathcal{U}}$$

Exercise

Let U be the change of basis matrix for $\mathcal{U}$.
What is $U^T U$?

Hint: What is $U^T(U\vec{x})$?

DSC 140B
Representation Learning

Lecture 05 | Part 2

Diagonalization

Matrices of a Transformation

- Let $\vec{f} : \mathbb{R}^d \to \mathbb{R}^d$ be a linear transformation
- The matrix representing $\vec{f}$ wrt the **standard basis** is:

$$A = \begin{pmatrix} \uparrow & \uparrow & \uparrow & \uparrow \\ \vec{f}(\hat{e}^{(1)}) & \vec{f}(\hat{e}^{(2)}) & \cdots & \vec{f}(\hat{e}^{(d)}) \\ \downarrow & \downarrow & \downarrow & \downarrow \end{pmatrix}$$

Matrices of a Transformation

- If we use a different basis $\mathcal{U} = \{\hat{u}^{(1)}, \ldots, \hat{u}^{(d)}\}$, the matrix representing $\vec{f}$ is:

$$A_{\mathcal{U}} = \begin{pmatrix} \uparrow & \uparrow & \uparrow & \uparrow \\ [\vec{f}(\hat{u}^{(1)})]_{\mathcal{U}} & [\vec{f}(\hat{u}^{(2)})]_{\mathcal{U}} & \cdots & [\vec{f}(\hat{u}^{(d)})]_{\mathcal{U}} \\ \downarrow & \downarrow & \downarrow & \downarrow \end{pmatrix}$$

- If $\vec{y} = A\vec{x}$, then $[\vec{y}]_{\mathcal{U}} = A_{\mathcal{U}}[\vec{x}]_{\mathcal{U}}$

Diagonal Matrices

- Diagonal matrices are very nice / easy to work with.
- Suppose A is a matrix. Is there a basis $\mathcal{U}$ where $A_{\mathcal{U}}$ is diagonal?
- Yes! *If* A is symmetric.

The Spectral Theorem[1]

- **Theorem**: Let A be an $n \times n$ *symmetric* matrix. Then there exist n eigenvectors of A which are all mutually orthogonal.

[1] for symmetric matrices

Eigendecomposition

- If A is a symmetric matrix, we can pick d of its eigenvectors $\hat{u}^{(1)}, \dots, \hat{u}^{(d)}$ to form an orthonormal basis.
- Any vector $\vec{x}$ can be written in terms of this **eigenbasis**.
- This is called its **eigendecomposition**:

$$\vec{x} = b_1 \hat{u}^{(1)} + b_2 \hat{u}^{(2)} + \dots + b_d \hat{u}^{(d)}$$

Matrix in the Eigenbasis

- **Claim**: the matrix of a linear transformation $\vec{f}$, written in a basis of its eigenvectors, is a **diagonal** matrix.

- The entries along the diagonal will be the **eigenvalues**.

Why?

$$A_{\mathcal{U}} = \begin{pmatrix} \uparrow & \uparrow & \uparrow & \uparrow \\ [\vec{f}(\hat{u}^{(1)})]_{\mathcal{U}} & [\vec{f}(\hat{u}^{(2)})]_{\mathcal{U}} & \cdots & [\vec{f}(\hat{u}^{(d)})]_{\mathcal{U}} \\ \downarrow & \downarrow & \downarrow & \downarrow \end{pmatrix}$$

- $\vec{f}(\hat{u}^{(1)}) = \lambda_1 \hat{u}^{(1)}$, so $[\vec{f}(\hat{u}^{(1)})]_{\mathcal{U}} = (\lambda_1, 0, \dots, 0)^T$.
- $\vec{f}(\hat{u}^{(2)}) = \lambda_2 \hat{u}^{(2)}$, so $[\vec{f}(\hat{u}^{(2)})]_{\mathcal{U}} = (0, \lambda_2, \dots, 0)^T$.
- ...

Matrix Multiplication

- We have seen that matrix multiplication evaluates a linear transformation.

- In the standard basis:

$$\vec{f}(\vec{x}) = A\vec{x}$$

- In another basis:

$$[\vec{f}(\vec{x})]_{\mathcal{U}} = A_{\mathcal{U}}[\vec{x}]_{\mathcal{U}}$$

Diagonalization

- Another way to compute $\vec{f}(x)$, starting with $\vec{x}$ in the standard basis:
    1. Change basis to the eigenbasis with U.
    2. Apply $\vec{f}$ in the eigenbasis with the diagonal $A_{\mathcal{U}}$.
    3. Go *back* to the standard basis with U^T.
- That is, $A\vec{x} = U^T A_{\mathcal{U}} U\vec{x}$. It follows that $A = U^T A_{\mathcal{U}} U$.

Spectral Theorem (Again)

- **Theorem**: Let A be an $n \times n$ *symmetric* matrix. Then there exists an orthogonal matrix U and a diagonal matrix Λ such that $A = U^T \Lambda U$.

- The *rows* of U are the eigenvectors of A, and the entries of Λ are its eigenvalues.

- U is said to **diagonalize** A.

DSC 140B
Representation Learning

Lecture 05 | Part 3

Dimensionality Reduction

High Dimensional Data

- Data is often high dimensional (many features)

- Example: Netflix user
  - Number of movies watched
  - Number of movies saved
  - Total time watched
  - Number of logins
  - Days since signup
  - Average rating for comedy
  - Average rating for drama
  - $\vdots$

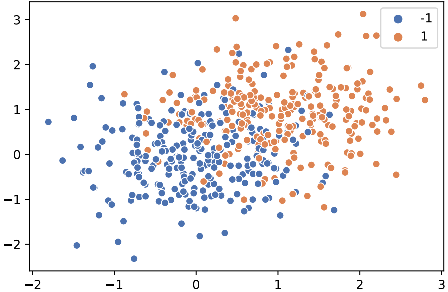
High Dimensional Data

- More features can give us more information
- But it can also cause problems
- **Today**: how do we reduce dimensionality without losing too much information?

More Features, More Problems

- Difficulties with high dimensional data:
  1. Requires more compute time / space
  2. Hard to visualize / explore
  3. The “curse of dimensionality”: it’s harder to learn

Experiment



- On this data, low 80% train/test accuracy
- Add 400 features of pure noise, re-train
- Now: 100% train accuracy, **58%** test accuracy
- **Overfitting!**

Task: Dimensionality Reduction

- We'd often like to **reduce** the dimensionality to improve performance, or to visualize.
- We will typically lose information
- Want to minimize the loss of useful information

Redundancy

- Two (or more) features may share the same information.
- Intuition: we may not need all of them.

Today

- Today we'll think about reducing dimensionality from $\mathbb{R}^d$ to $\mathbb{R}^1$
- Next time we'll go from $\mathbb{R}^d$ to $\mathbb{R}^{d'}$, with $d' \leq d$

Today's Example

- Let's say we represent a phone with two features:
  - x_1: screen width
  - x_2: phone weight

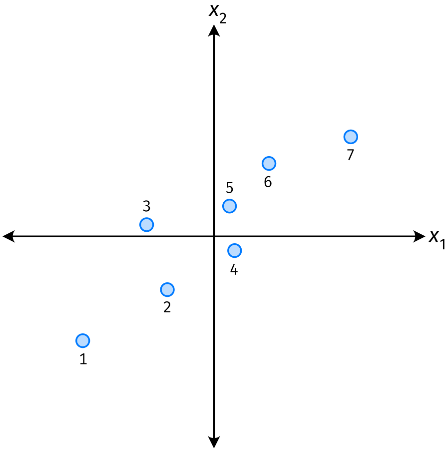
- Both measure a phone's "size".

- Instead of representing a phone with both x_1 and x_2, can we just use a single number, z?
  - Reduce dimensionality from 2 to 1.

First Approach: Remove Features

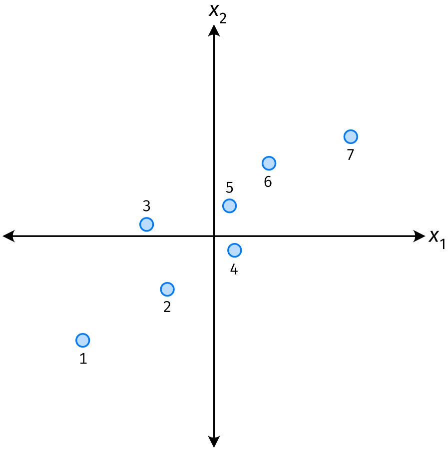
- Screen width and weight share information.
- **Idea:** keep one feature, remove the other.
- That is, set new feature $z = x_1$ (or $z = x_2$).

Removing Features



- Say we set $z^{(i)} = \vec{x}_1^{(i)}$ for each phone, i.
- Observe: $z^{(4)} > z^{(5)}$.
- Is phone 4 really "larger" than phone 5?

Removing Features



- Say we set $z^{(i)} = \vec{x}_2^{(i)}$ for each phone, i.
- Observe: $z^{(3)} > z^{(4)}$.
- Is phone 3 really "larger" than phone 4?

Better Approach: Mixtures of Features

- **Idea**: z should be a combination of x_1 and x_2.

- One approach: linear combination.

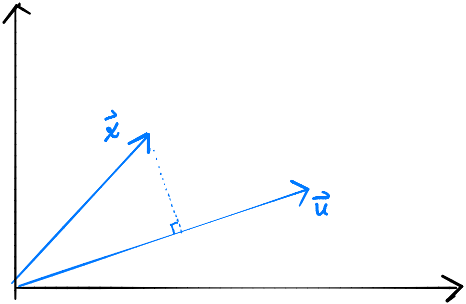
$$\begin{aligned} z &= u_1 x_1 + u_2 x_2 \\ &= \vec{u} \cdot \vec{x} \end{aligned}$$

- $u_1, \ldots, u_2$ are the mixture coefficients; we can choose them.

Normalization

- Mixture coefficients generalize proportions.
- We could assume, e.g., $|u_1| + |u_2| = 1$.
- But it makes the math easier if we assume $u_1^2 + u_2^2 = 1$.
- Equivalently, if $\vec{u} = (u_1, u_2)^T$, assume $\|\vec{u}\| = 1$

Geometric Interpretation

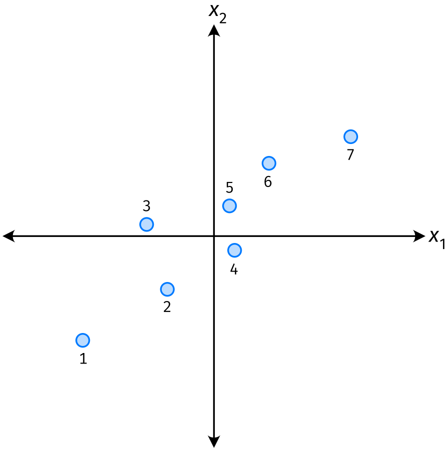


- z measures how much of $\vec{x}$ is in the direction of $\vec{u}$
- If $\vec{u} = (1, 0)^T$, then $z = x_1$
- If $\vec{u} = (0, 1)^T$, then $z = x_2$

Choosing $\vec{u}$

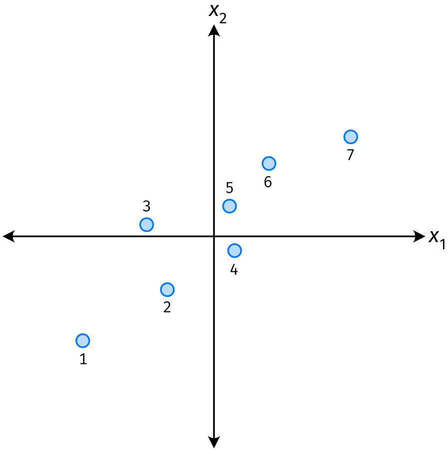
- Suppose we have only two features:
  - x_1: screen size
  - x_2: phone thickness
- We'll create single new feature, z, from x_1 and x_2.
  - Assume $z = u_1 x_1 + u_2 x_2 = \vec{x} \cdot \vec{u}$
  - Interpretation: z is a measure of a phone's size
- How should we choose $\vec{u} = (u_1, u_2)^T$?

Example



- $\vec{u}$ defines a direction
- $\vec{z}^{(i)} = \vec{x}^{(i)} \cdot \vec{u}$ measures position of $\vec{x}$ along this direction

Example



- Phone "size" varies most along a diagonal direction.
- Along direction of "max variance", phones are well-separated.
- **Idea:** $\vec{u}$ should point in direction of "max variance".

Our Algorithm (Informally)

- **Given**: data points $\vec{x}^{(1)}, \ldots, \vec{x}^{(n)} \in \mathbb{R}^d$
- Pick $\vec{u}$ to be the direction of "max variance"
- Create a new feature, z, for each point:

$$z^{(i)} = \vec{x}^{(i)} \cdot \vec{u}$$

PCA

- This algorithm is called **Principal Component Analysis**, or **PCA**.
- The direction of maximum variance is called the **principal component**.

Exercise

Suppose the direction of maximum variance in a data set is

$$\vec{u} = (1/\sqrt{2}, -1/\sqrt{2})^T$$

Let

- $\vec{x}^{(1)} = (3, -2)^T$
- $\vec{x}^{(2)} = (1, 4)^T$

What are $z^{(1)}$ and $z^{(2)}$?

Problem

- How do we compute the “direction of maximum variance”?

DSC 140B
Representation Learning

Lecture 05 | Part 4

Covariance Matrices

Variance

- We know how to compute the variance of a set of numbers $X = \{x^{(1)}, \dots, x^{(n)}\}$:

$$\mathrm{Var}(X) = \frac{1}{n}\sum_{i=1}^{n}(x^{(i)} - \mu)^2$$

- The variance measures the “spread” of the data

Generalizing Variance

- If we have two features, x_1 and x_2, we can compute the variance of each as usual:

$$\text{Var}(x_1) = \frac{1}{n}\sum_{i=1}^{n}(\vec{x}_1^{(i)} - \mu_1)^2$$

$$\text{Var}(x_2) = \frac{1}{n}\sum_{i=1}^{n}(\vec{x}_2^{(i)} - \mu_2)^2$$

- Can also measure how x_1 and x_2 vary together.

Measuring Similar Information

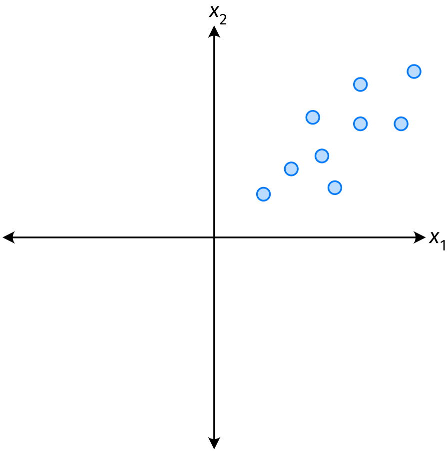
- Features which share information if they *vary together*.
  - A.k.a., they “co-vary”
- Positive association: when one is above average, so is the other
- Negative association: when one is above average, the other is below average

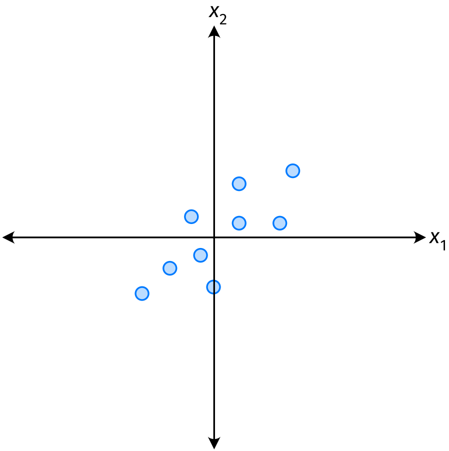
Examples

- Positive: temperature and ice cream cones sold.
- Positive: temperature and shark attacks.
- Negative: temperature and coats sold.

Centering

- First, it will be useful to **center** the data.





Centering

- Compute the mean of each feature:

$$\mu_j = \frac{1}{n} \sum_{1}^{n} \vec{x}_j^{(i)}$$

- Define new centered data:

$$\vec{z}^{(i)} = \begin{pmatrix} \vec{x}_1^{(i)} - \mu_1 \\ \vec{x}_2^{(i)} - \mu_2 \\ \vdots \\ \vec{x}_d^{(i)} - \mu_d \end{pmatrix}$$

Centering (Equivalently)

- Compute the mean of all data points:

$$\mu = \frac{1}{n} \sum_{1}^{n} \vec{x}^{(i)}$$

- Define new centered data:

$$\vec{z}^{(i)} = \vec{x}^{(i)} - \mu$$

Exercise

Center the data set:

$$\vec{x}^{(1)} = (1, 2, 3)^T$$
$$\vec{x}^{(2)} = (-1, -1, 0)^T$$
$$\vec{x}^{(3)} = (0, 2, 3)^T$$

Quantifying Co-Variance

- One approach is as follows[2].

$$\text{Cov}(x_i, x_j) = \frac{1}{n} \sum_{k=1}^{n} \vec{x}_i^{(k)} \vec{x}_j^{(k)}$$

  - For each data point, multiply the value of feature *i* and feature *j*, then average these products.

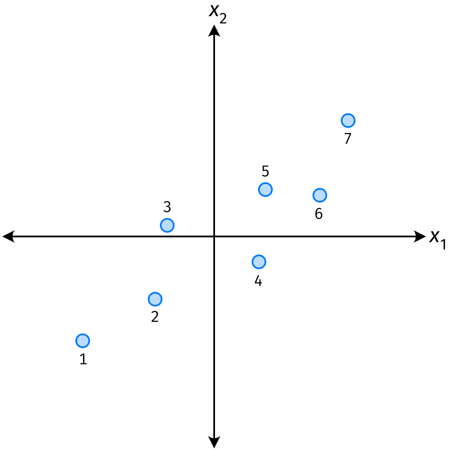
  - This is the **covariance** of features *i* and *j*.

[2] Assuming centered data

Quantifying Covariance

- Assume the data are **centered**.

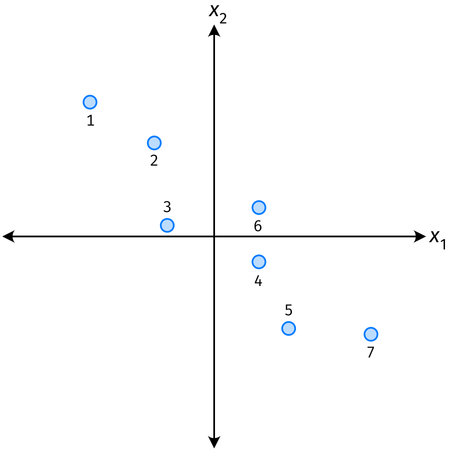
$$\text{Covariance} = \frac{1}{7}\sum_{i=1}^{7} \vec{x}_1^{(i)} \times \vec{x}_2^{(i)}$$



Quantifying Covariance

- Assume the data are **centered**.

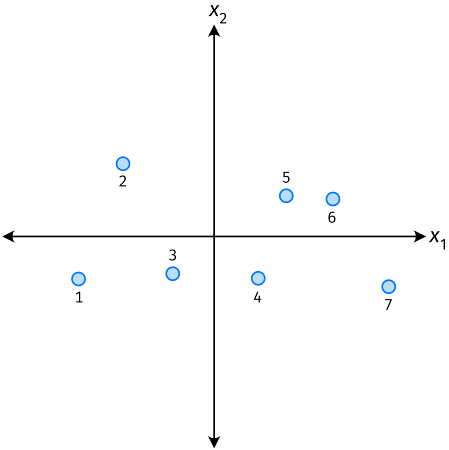
$$\text{Covariance} = \frac{1}{7}\sum_{i=1}^{7} \vec{x}_1^{(i)} \times \vec{x}_2^{(i)}$$



Quantifying Covariance

- Assume the data are **centered**.

$$\text{Covariance} = \frac{1}{7}\sum_{i=1}^{7} \vec{x}_1^{(i)} \times \vec{x}_2^{(i)}$$



Quantifying Covariance

- The **covariance** quantifies extent to which two variables vary together.

- Assume we have centered the data.

- The **sample covariance** of feature i and j is:
$$\sigma_{ij} = \frac{1}{n} \sum_{k=1}^{n} \vec{x}_i^{(k)} \vec{x}_j^{(k)}$$

Exercise

True or False: $\sigma_{ij} = \sigma_{ji}$?

$$\sigma_{ij} = \frac{1}{n} \sum_{k=1}^{n} \vec{x}_i^{(k)} \vec{x}_j^{(k)}$$

Covariance Matrices

- Given data $\vec{x}^{(1)}, \ldots, \vec{x}^{(n)} \in \mathbb{R}^d$.
- The **sample covariance matrix** C is the $d \times d$ matrix whose ij entry is defined to be σ_{ij}.

$$\sigma_{ij} = \frac{1}{n} \sum_{k=1}^{n} \vec{x}_i^{(k)} \vec{x}_j^{(k)}$$

Observations

- Diagonal entries of C are the variances.
- The matrix is **symmetric**!

Note

- Sometimes you'll see the sample covariance defined as:

$$\sigma_{ij} = \frac{1}{n-1} \sum_{k=1}^{n} \vec{x}_i^{(k)} \vec{x}_j^{(k)}$$

  Note the $1/(n-1)$
- This is an **unbiased** estimator of the population covariance.
- Our definition is the **maximum likelihood** estimator.
- In practice, it doesn't matter: $1/(n-1) \approx 1/n$.
- For consistency, in this class use $1/n$.

Computing Covariance

- There is a “trick” for computing sample covariance matrices.
- Step 1: make $n \times d$ data matrix, X
- Step 2: make Z by centering columns of X
- Step 3: $C = \frac{1}{n} Z^T Z$

Computing Covariance (in code)[3]

```
>>> mu = X.mean(axis=0)
>>> Z = X - mu
>>> C = 1 / len(X) * Z.T @ Z
```

[3] Or use `np.cov`