

COMP6237 Data Mining

Covariance, EVD, PCA & SVD

Jonathon Hare
jsh2@ecs.soton.ac.uk

Variance and Covariance

Random Variables and Expected Values

- Mathematicians talk variance (and covariance) in terms of *random variables* and *expected values*
- A random variable is a variable that takes on different values due to chance.
 - The set of sample values from a single dimension of a featurespace can be considered to be a random variable
- The expected value (denoted $E[X]$) is the most likely value a random variable will take.
 - If we **assume** that the values an element of a feature can take are all equally likely then expected value is thus just the **mean value**

Variance

$$\sigma^2(x) = \frac{1}{n} \sum_{i=1}^n (x_i - \mu)^2$$

Variance (σ^2) is the mean squared difference from the mean (μ).

It's a measure of how spread-out the data is.

technically it's $E[(X - E[X])^2]$

Covariance

$$\sigma(x, y) = \frac{1}{n} \sum_{i=1}^n (x - \mu_x)(y - \mu_y)$$

Covariance ($\sigma(x,y)$) measures how two variables change together

technically it's $E[(x - E[x])(y - E[y])]$

Covariance

$$\sigma(x, y) = \frac{1}{n} \sum_{i=1}^n (x - \mu_x)(y - \mu_y)$$

The variance is the covariance when the two variables are the same ($\sigma(x,x)=\sigma^2(x)$)

Covariance

$$\sigma(x, y) = \frac{1}{n} \sum_{i=1}^n (x - \mu_x)(y - \mu_y)$$

A covariance of 0 means the variables are uncorrelated

(Covariance is related to Correlation... see notes)

Covariance Matrix

$$\Sigma = \begin{bmatrix} \sigma(X_1, X_1) & \sigma(X_1, X_2) & \dots & \sigma(X_1, X_n) \\ \sigma(X_2, X_1) & \sigma(X_2, X_2) & \dots & \sigma(X_2, X_n) \\ \vdots & \vdots & \ddots & \vdots \\ \sigma(X_n, X_1) & \sigma(X_n, X_2) & \dots & \sigma(X_n, X_n) \end{bmatrix}$$

A covariance matrix encodes how all possible pairs of dimensions in an n -dimensional dataset vary together

Covariance Matrix

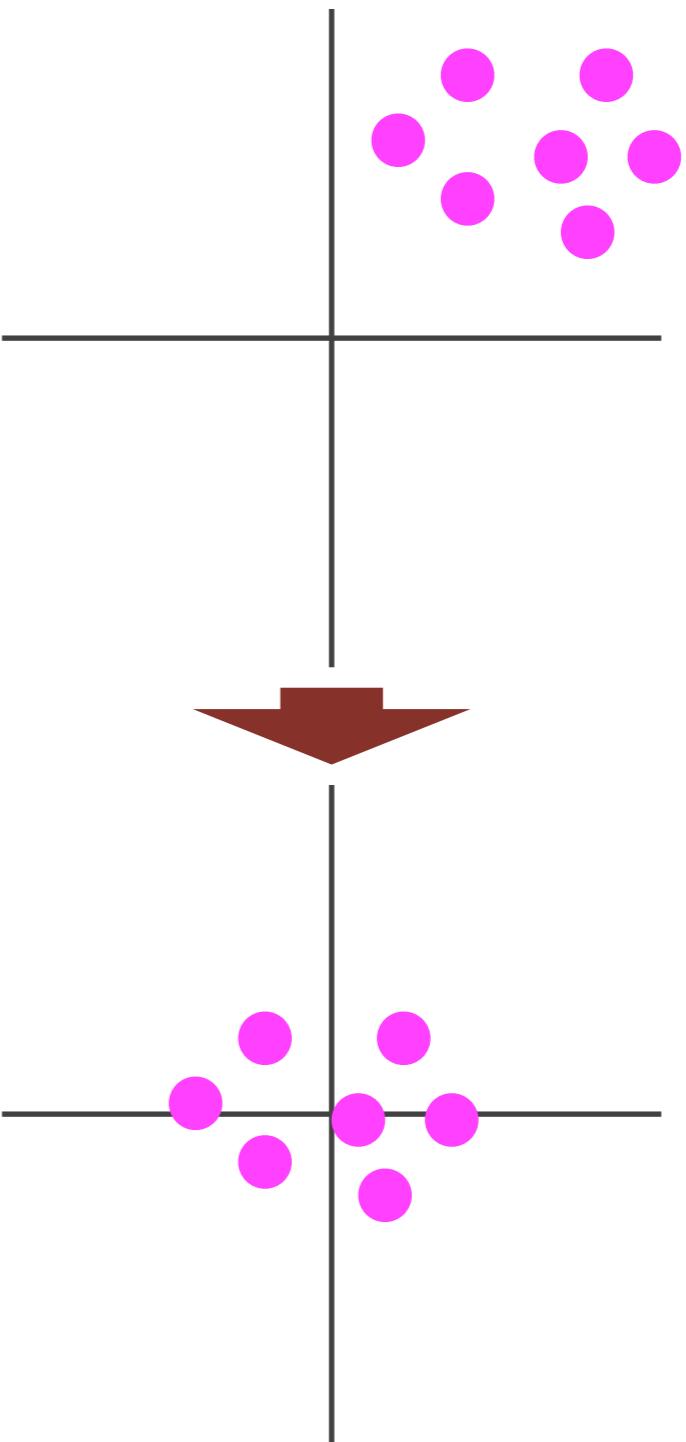
$$\Sigma = \begin{bmatrix} \sigma(X_1, X_1) & \sigma(X_1, X_2) & \dots & \sigma(X_1, X_n) \\ \sigma(X_2, X_1) & \sigma(X_2, X_2) & \dots & \sigma(X_2, X_n) \\ \vdots & \vdots & \ddots & \vdots \\ \sigma(X_n, X_1) & \sigma(X_n, X_2) & \dots & \sigma(X_n, X_n) \end{bmatrix}$$

The covariance matrix is a **square symmetric matrix**

Demo: 2D covariance

Mean Centring

- Mean Centring is the process of computing the mean (across each dimension independently) of a set of vectors, and then subtracting the mean vector from every vector in the set.
 - All the vectors will be translated so their average position is the origin



Covariance matrix again

$$Z = \begin{bmatrix} v_{11} & v_{12} & v_{12} & \dots \\ v_{21} & v_{22} & v_{22} & \dots \\ v_{31} & v_{32} & v_{32} & \dots \\ v_{41} & v_{42} & v_{42} & \dots \end{bmatrix}$$

Each row is a **mean centred** featurevector

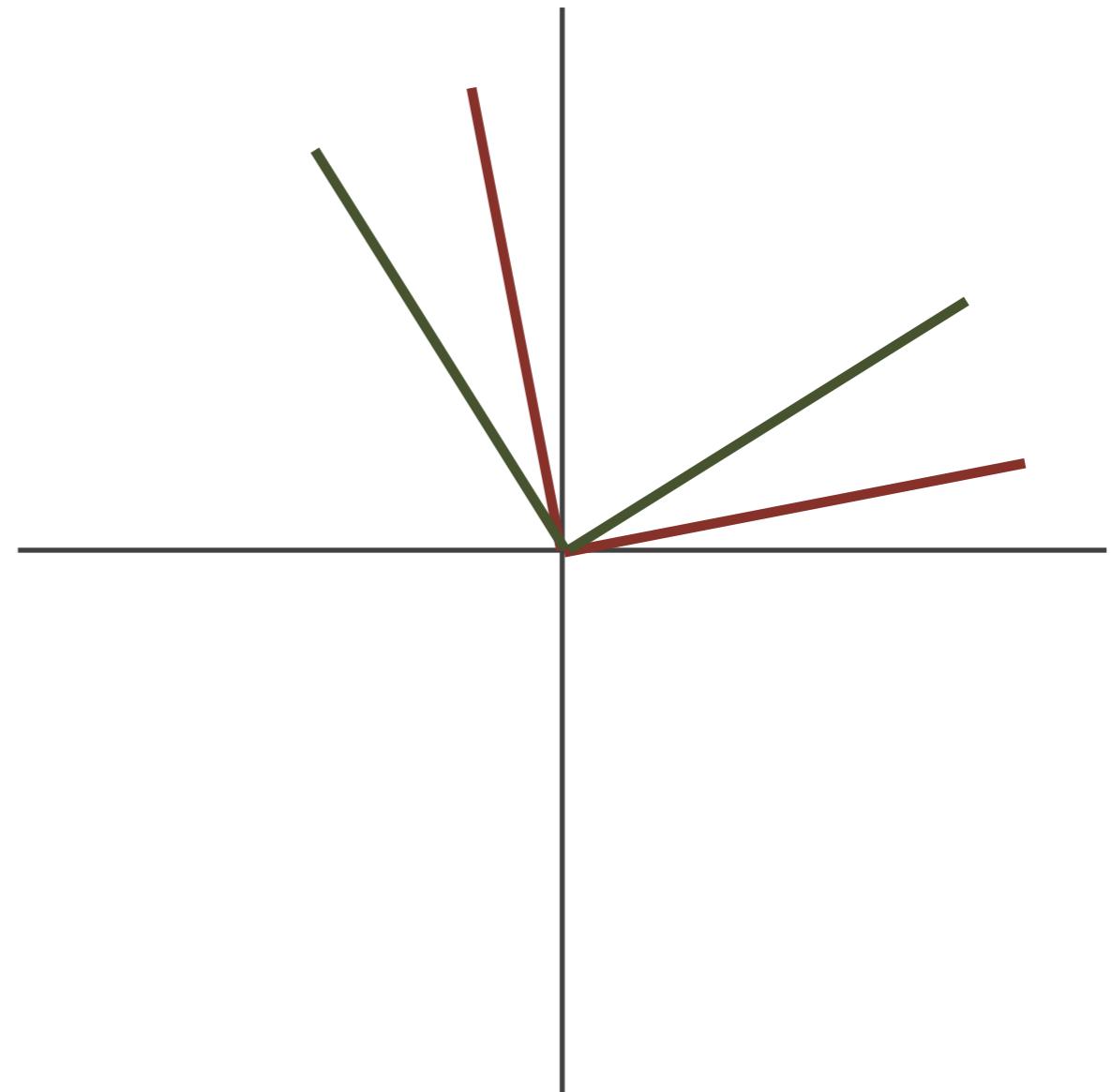
Then

$$\Sigma \propto Z^T Z$$

Principal axes of variation

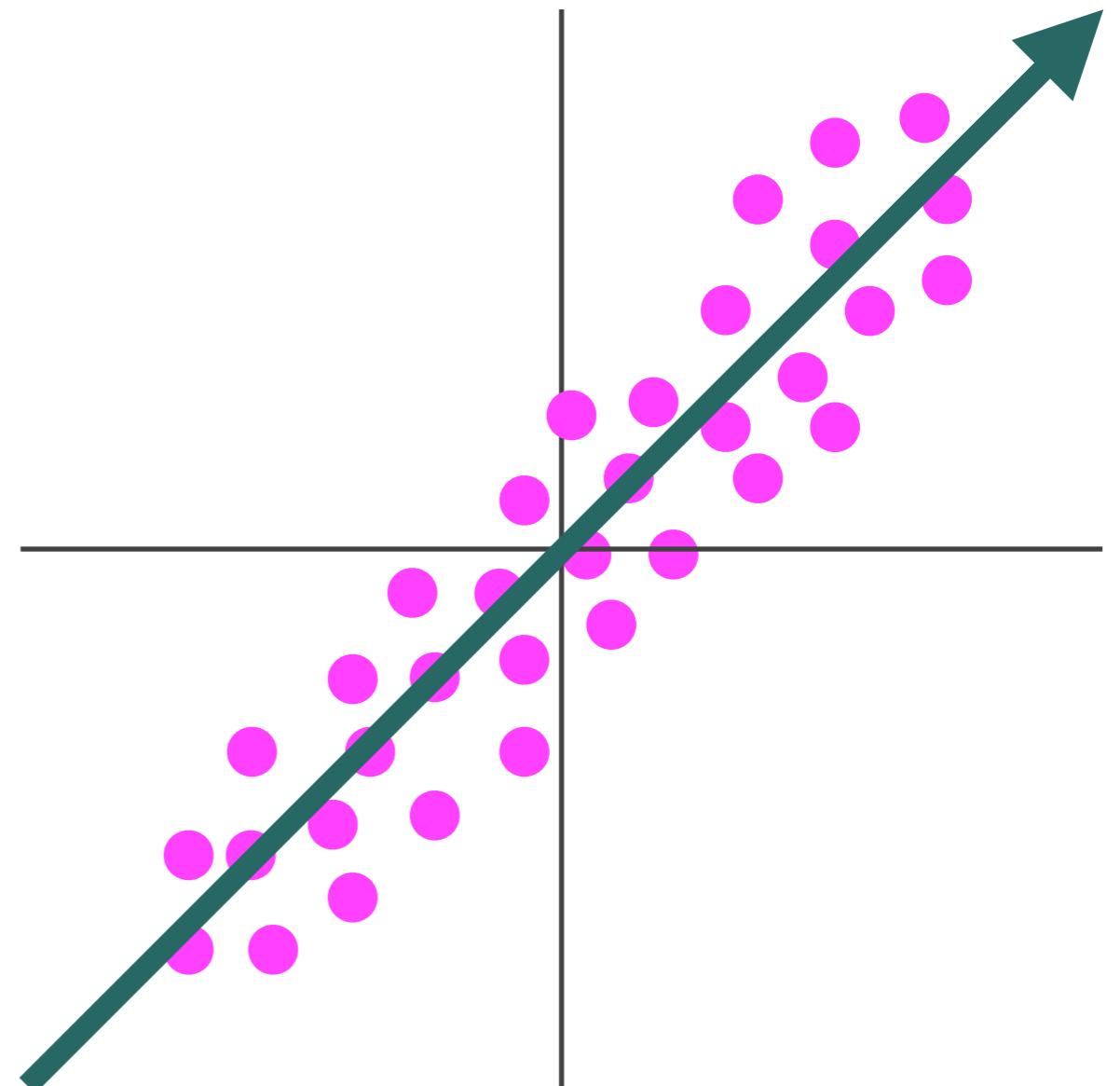
Basis

- A basis is a set of n **linearly independent** vectors in an n dimensional space
 - The vectors are **orthogonal**
 - They form a “coordinate system”
 - There are an infinite number of possible basis



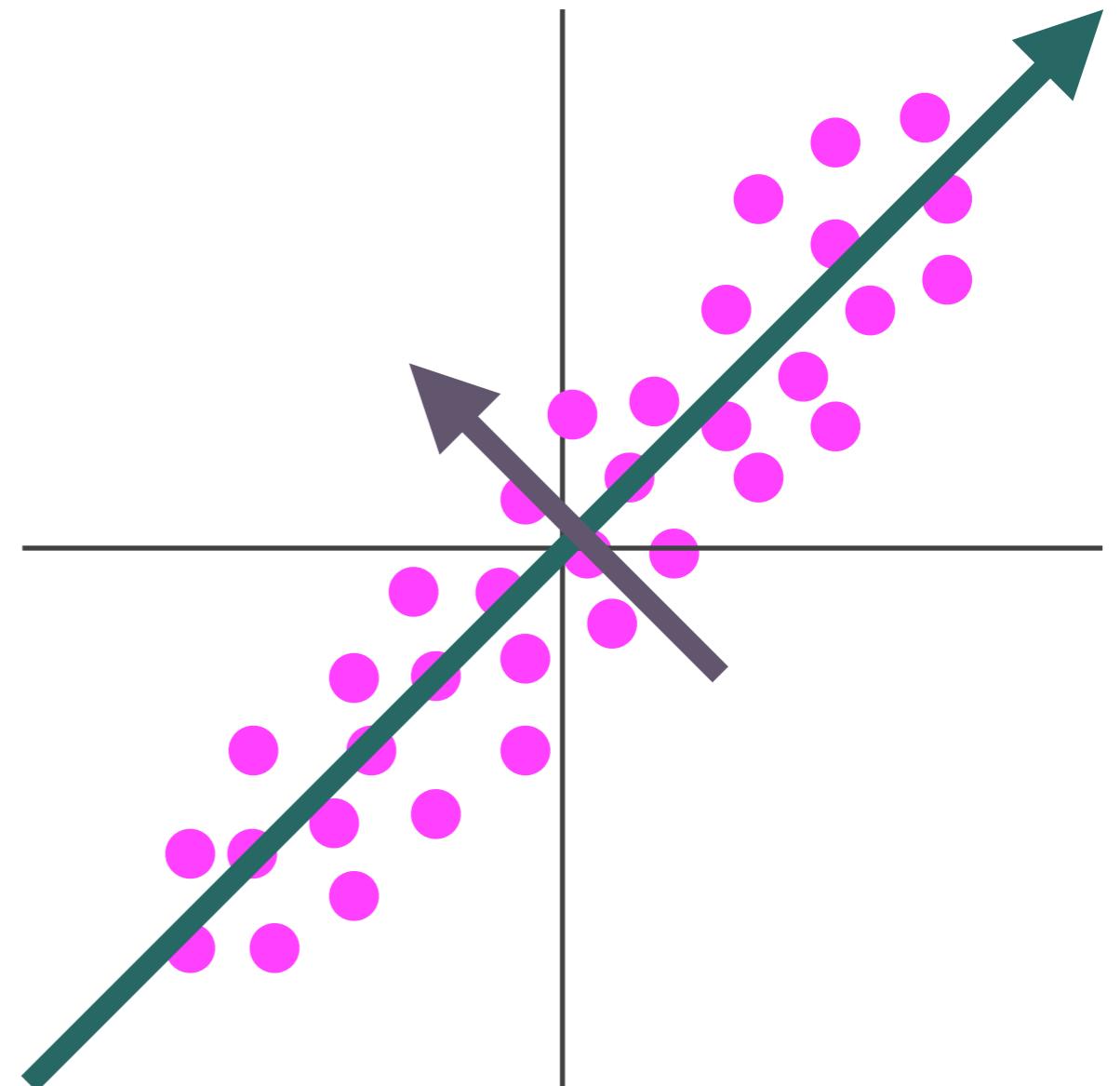
The first principal axis

- For a given set of n dimensional data, the ***first principle axis*** (or just ***principal axis***) is the vector that describes the direction of **greatest variance**.



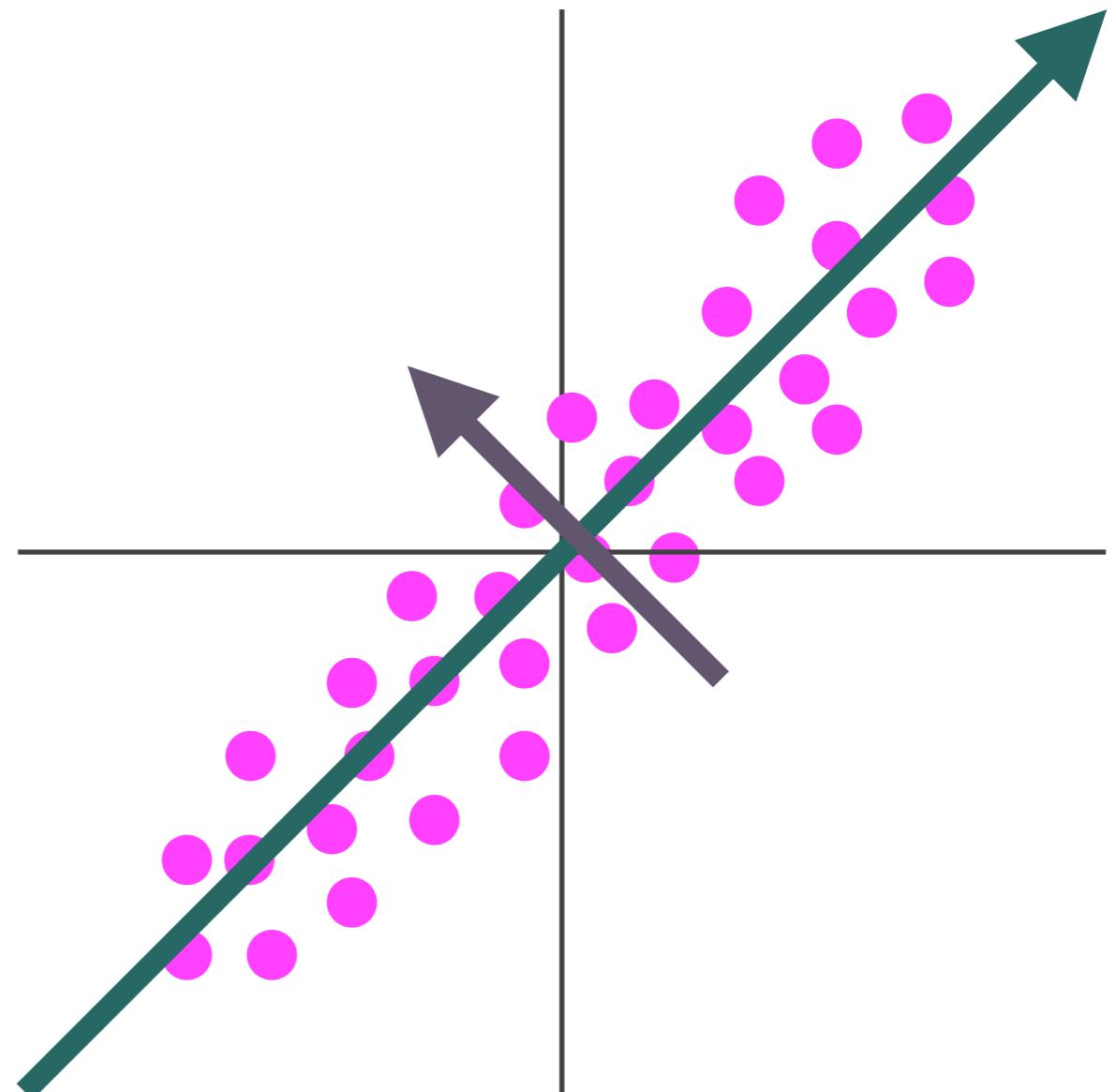
The second principal axis

- The **second principal axis** is a vector in the direction of greatest variance orthogonal (perpendicular) to the first major axis.



The third principal axis

- In a space with 3 or more dimensions, the third principal axis is the direction of greatest variance orthogonal to both the first and second principal axes.
 - The forth... ... and so on...
 - The set of n **principal axes** of an n dimensional space are a **basis**



Eigenvectors, Eigenvalues and Eigendecomposition

Eigenvectors and Eigenvalues

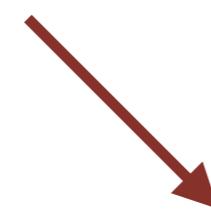
Very important equation!

$$Av = \lambda v$$



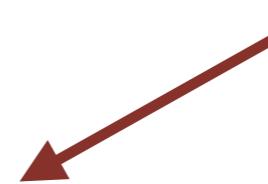
Eigenvectors and Eigenvalues

a $n \times n$ square matrix



$$Av = \lambda v$$

a scalar value,
known as an
eigenvalue



an n dimensional vector,
known as an **eigenvector**

Eigenvectors and Eigenvalues

$$\mathbf{A}v = \lambda v$$

There are at most n eigenvector-eigenvalue pairs

If \mathbf{A} is symmetric, then the set of eigenvectors is orthogonal

Can you see where this is going?

Eigenvectors and Eigenvalues

$$Av = \lambda v$$

If **A** is a **covariance matrix**, then the eigenvectors are the **principal axes**.

The eigenvalues are proportional to the **variance** of the data along each eigenvector.

The eigenvector corresponding to the **largest** eigenvalue is the first P.C.

Finding the EVecs and EVals

- For small matrices ($n \leq 4$) there are algebraic solutions to finding all the eigenvector-eigenvalue pairs
- For larger matrices, numerical solutions to the **Eigendecomposition** must be sought.

Eigendecomposition (aka EVD)

columns of \mathbf{Q} are the eigenvectors

$$\mathbf{A} = \underbrace{\mathbf{Q}}_{\text{columns of } \mathbf{Q} \text{ are the eigenvectors}} \mathbf{\Lambda} \underbrace{\mathbf{Q}^{-1}}_{\text{diagonal eigenvalue matrix } (\Lambda_{ii} = \lambda_i)}$$

diagonal eigenvalue matrix ($\Lambda_{ii} = \lambda_i$)

Eigendecomposition

$$\mathbf{A} = \underline{\mathbf{Q}} \boldsymbol{\Lambda} \underline{\mathbf{Q}}^{-1}$$

If \mathbf{A} is *real symmetric* (i.e. a covariance matrix), then $\mathbf{Q}^{-1} = \mathbf{Q}^T$ (i.e. eigenvectors are orthogonal), so:

$$\mathbf{A} = \underline{\mathbf{Q}} \boldsymbol{\Lambda} \underline{\mathbf{Q}}^T$$

Eigendecomposition

In summary, the Eigendecomposition of a covariance matrix \mathbf{A} :

$$\mathbf{A} = \mathbf{Q}\Lambda\mathbf{Q}^T$$

Gives you the principal axes and their relative magnitudes

Ordering

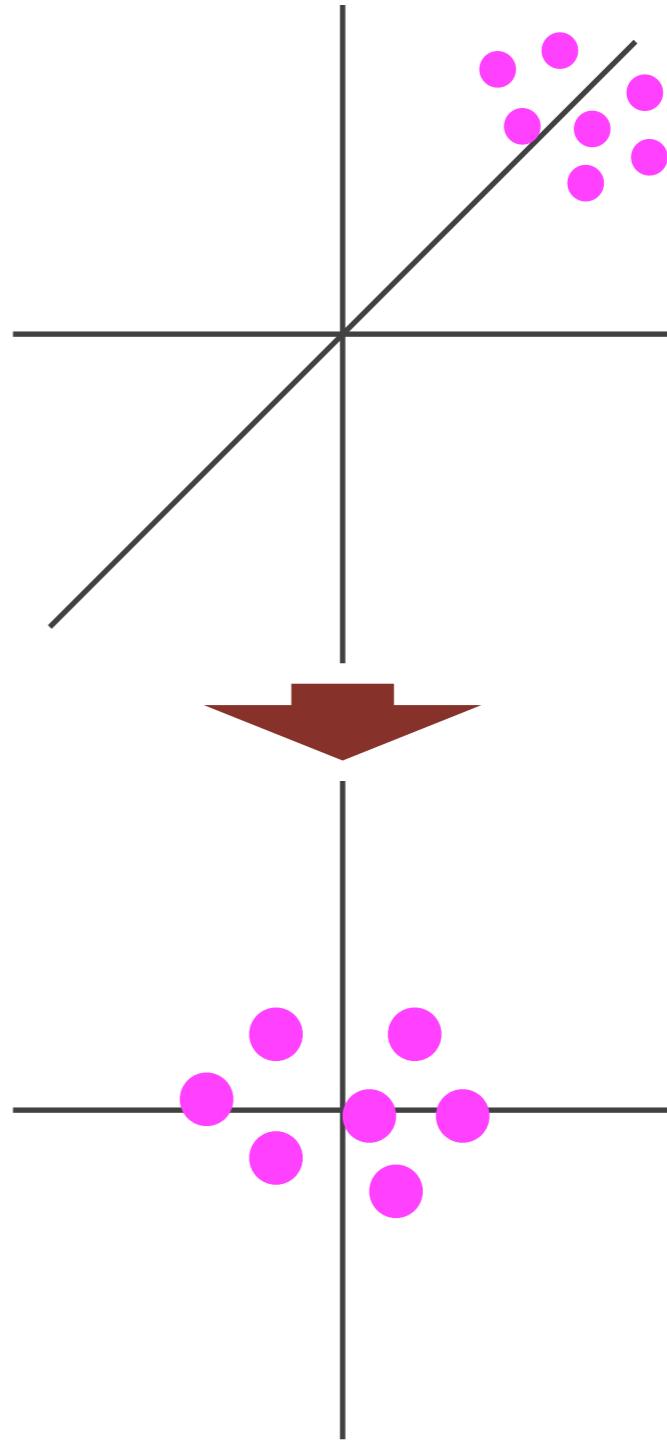
- Standard Eigendecomposition implementations will order the eigenvectors (columns of \mathbf{Q}) such that the eigenvalues (in the diagonal of Λ) are sorted in order of decreasing value.
- Some solvers are optimised to only find the top k eigenvalues and corresponding eigenvectors, rather than all of them.

Demo: Covariance, Eigendecomposition and principal axes

Principal Component Analysis

Linear Transform

- A linear transform \mathbf{W} projects data from one space into another:
 - $\mathbf{T} = \mathbf{Z}\mathbf{W}$
 - Original data stored in the rows of \mathbf{Z}
 - \mathbf{T} can have fewer dimensions than \mathbf{Z} .

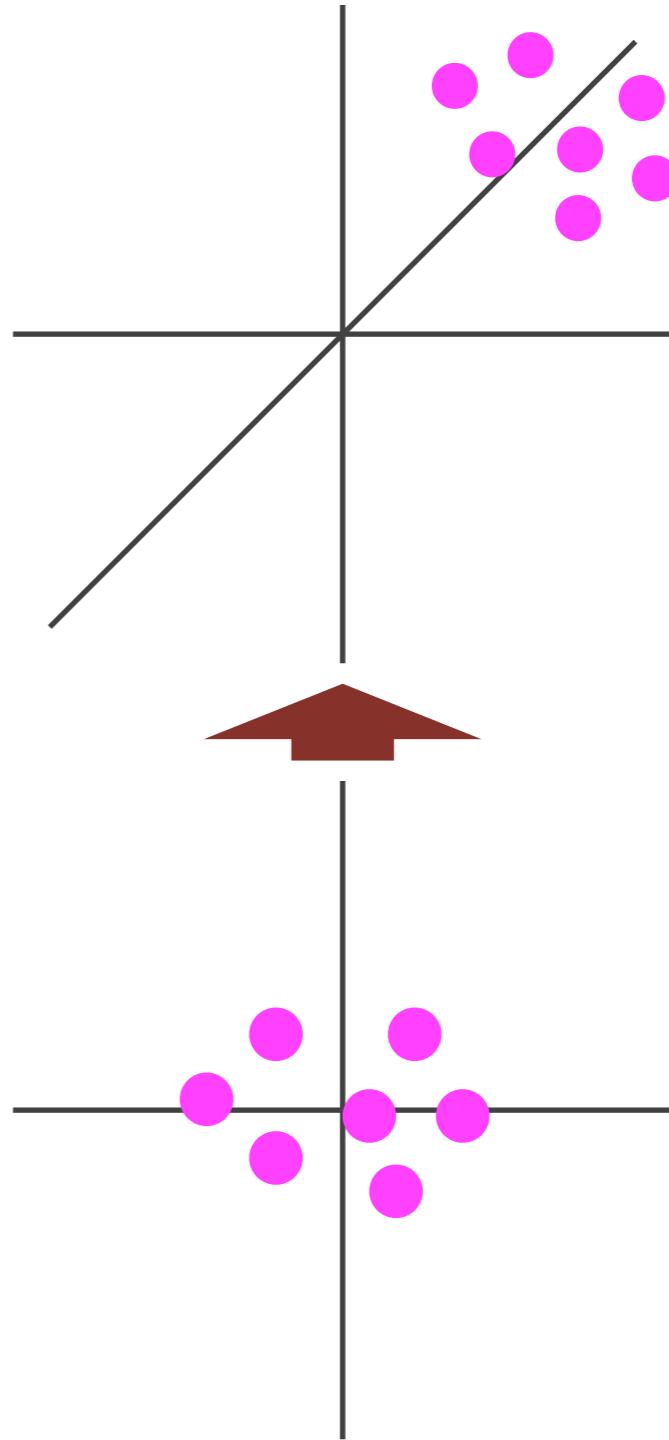


Linear Transforms

- The effects of a linear transform can be reversed if **W** is **invertible**:

- $Z = TW^{-1}$

- A lossy process if the dimensionality of the spaces is different



PCA

- PCA is an **Orthogonal Linear Transform** that maps data from its original space to a space defined by the principal axes of the data.
- The transform matrix **W** is just the eigenvector matrix **Q** from the Eigendecomposition of the covariance matrix of the data.
- Dimensionality reduction can be achieved by removing the eigenvectors with low eigenvalues from **Q** (i.e. keeping the first L columns of **Q** assuming the eigenvectors are sorted by decreasing eigenvalue).

PCA Algorithm

1. Mean-centre the data vectors
2. Form the vectors into a matrix Z , such that each row corresponds to a vector
3. Perform the Eigendecomposition of the matrix $Z^T Z$, to recover the eigenvector matrix Q and diagonal eigenvalue matrix Λ : $Z^T Z = Q \Lambda Q^T$
4. Sort the columns of Q and corresponding diagonal values of Λ so that the eigenvalues are decreasing.
5. Select the L largest eigenvectors of Q (the first L columns) to create the transform matrix Q_L .
6. Project the original vectors into a lower dimensional space, T_L : $T_L = Z Q_L$

Demo: PCA

Singular Value Decomposition

Singular Value Decomposition

(Complex Valued M)

$$M = U \Sigma V^*$$

Conjugate Transpose



(Real Valued M)

$$M = U \Sigma V^T$$


Note that this is a different Σ to the one used earlier

(Real) Singular Value Decomposition

Orthogonal matrix:

$$\mathbf{U}^T \mathbf{U} = \mathbf{U} \mathbf{U}^T = \mathbf{I}$$

Orthogonal matrix:

$$\mathbf{V}^T \mathbf{V} = \mathbf{V} \mathbf{V}^T = \mathbf{I}$$

$$\mathbf{M} = \mathbf{U} \Sigma \mathbf{V}^T$$

```
graph TD; A[Orthogonal matrix: U^T U = U U^T = I] --> U[U]; B[Orthogonal matrix: V^T V = V V^T = I] --> VT[V^T]; C[Real Diagonal Matrix] --> Sigma[\Sigma]
```

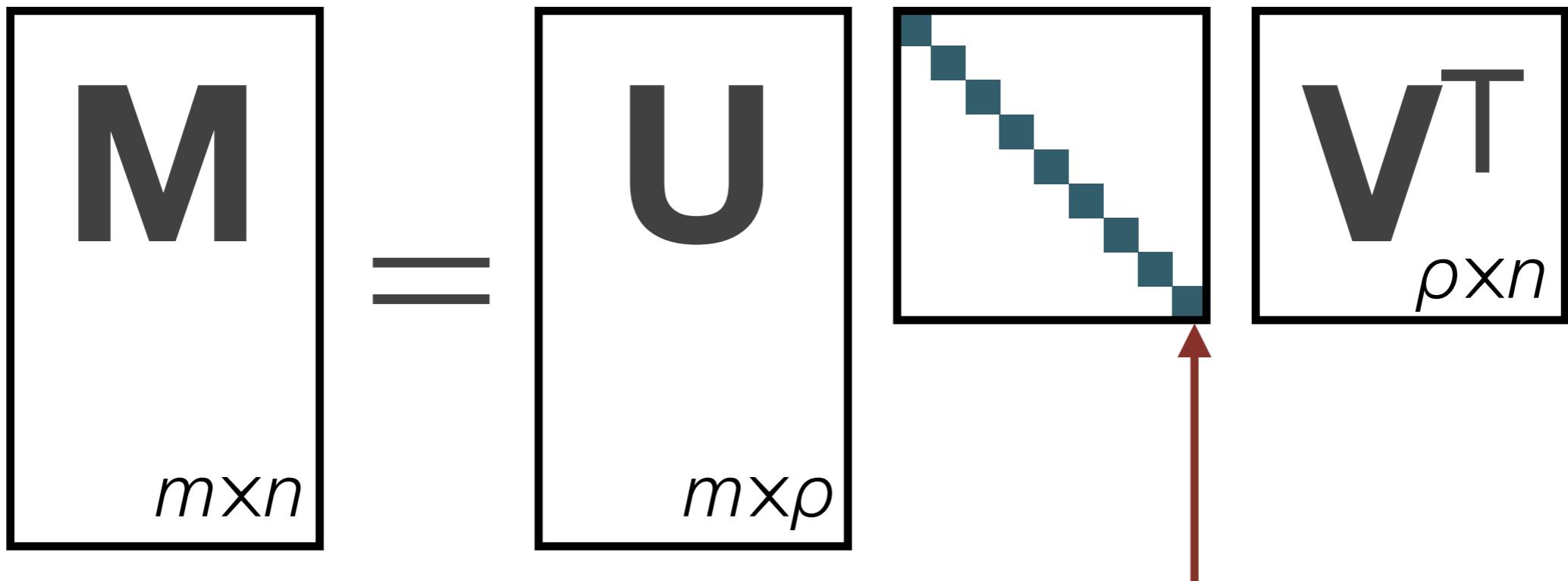
Real Diagonal Matrix

Singular Value Decomposition

$$\begin{matrix} \mathbf{M} \\ m \times n \end{matrix} = \begin{matrix} \mathbf{U} \\ m \times p \end{matrix} \begin{matrix} \Sigma \\ p \times p \end{matrix} \begin{matrix} \mathbf{V}^T \\ p \times n \end{matrix}$$

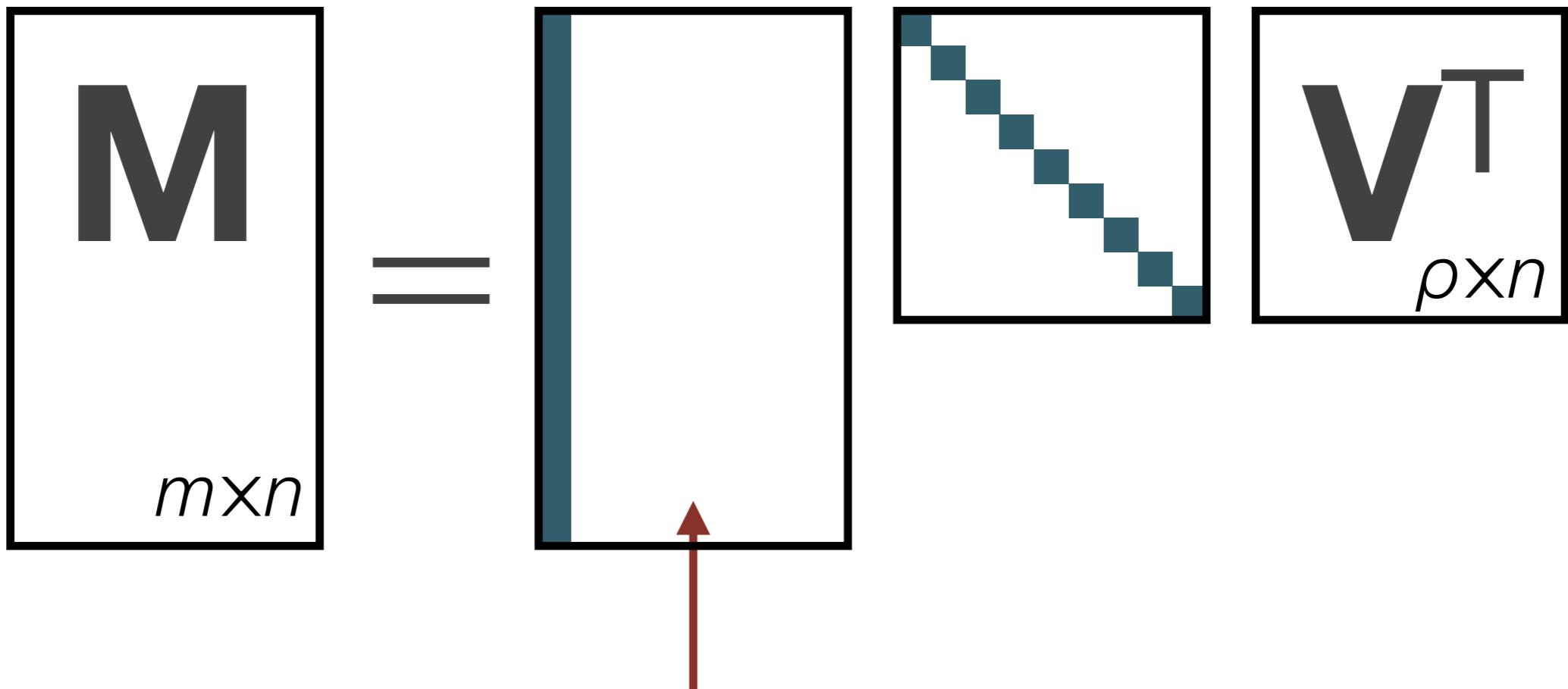
where ρ is the rank of \mathbf{M}
(i.e. the number of linearly independent rows or columns)

Singular Value Decomposition



“Singular Values”
(Usually organised/computed so they monotonically decrease)

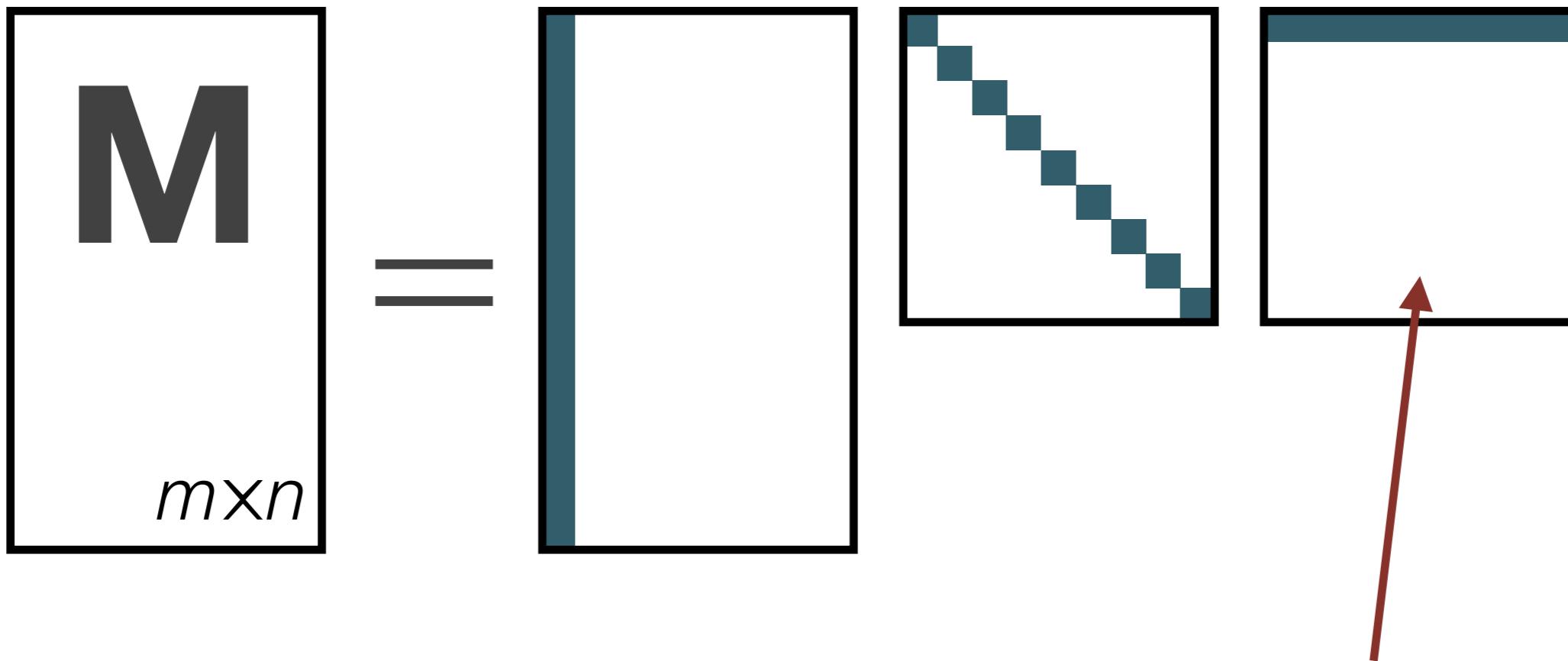
Singular Value Decomposition



Columns of U are called the “left singular vectors”

The left singular vectors are the eigenvectors of MM^T

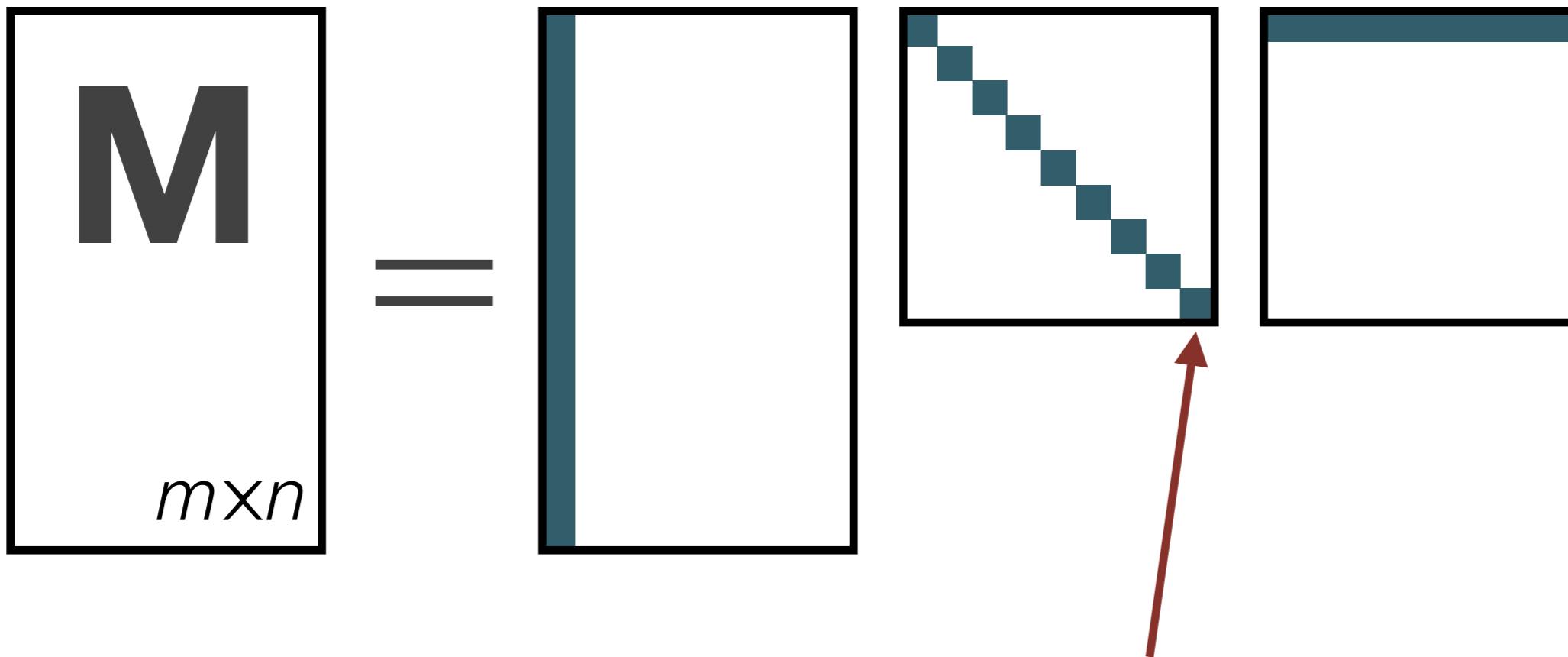
Singular Value Decomposition



Rows of V^T (columns of V) are called the “right singular vectors”

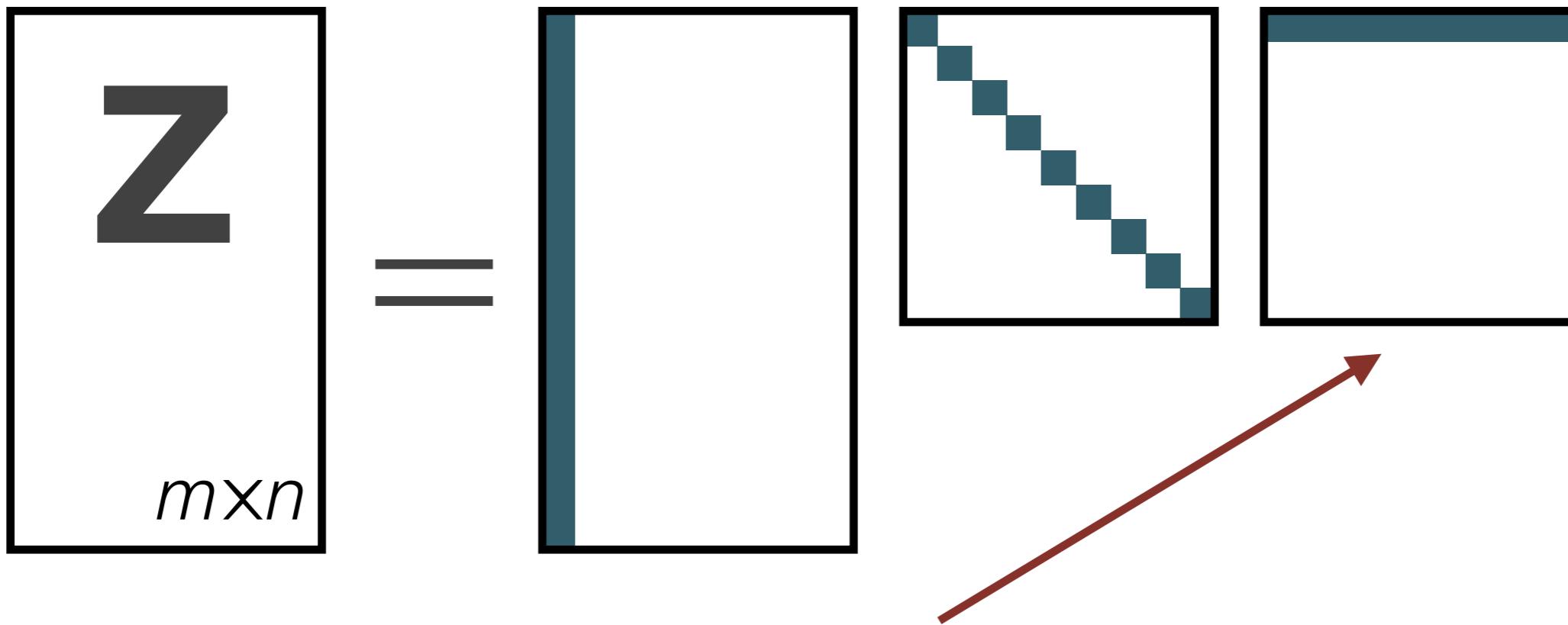
The right singular vectors are the eigenvectors of $M^T M$

Singular Value Decomposition



The singular values are the square roots of the eigenvalues of the corresponding eigenvectors of $\mathbf{M}^T\mathbf{M}$ and $\mathbf{M}\mathbf{M}^T$

Singular Value Decomposition



If \mathbf{Z} is the matrix of mean centred featurevectors that we introduced earlier, then the columns of \mathbf{V} are the eigenvectors of $\mathbf{Z}^T\mathbf{Z}$ - that is they are the **principle components of \mathbf{Z} !**

SVD-based PCA Algorithm

1. Mean-centre the data vectors
2. Form the vectors into a matrix Z , such that each row corresponds to a vector
3. Perform the SVD of the matrix Z , to recover the right singular vector matrix V and diagonal singular value matrix Σ : $Z = U\Sigma V^T$
4. Sort the columns of V and corresponding diagonal values of Σ so that the singular values are decreasing.
5. Select the first L right singular vectors from V (the first L columns) to create the transform matrix Q_L .
6. Project the original vectors into a lower dimensional space,
 T_L : $T_L = ZQ_L$

Why SVD-based PCA

- Numerical stability
 - Computing the covariance matrix might be disastrous - e.g. consider the Läuchli Matrix:
- Performance
 - SVD *can* be faster than computing covariance followed by EVD

$$\begin{pmatrix} 1 & 1 & 1 \\ \epsilon & 0 & 0 \\ 0 & \epsilon & 0 \\ 0 & 0 & \epsilon \end{pmatrix}^T$$

Truncated Singular Value Decomposition

$$\begin{matrix} M \\ \text{mxn} \end{matrix} \approx \begin{matrix} U_r \\ \text{mxr} \end{matrix} \begin{matrix} \Sigma_r \\ r \times r \end{matrix} \begin{matrix} V_r^T \\ \text{rxn} \end{matrix}$$

*Truncated SVD considers only the **largest** r singular values (and corresponding left & right vectors)*

Low Rank Approximation

$$\begin{matrix} \mathbf{M} \\ m \times n \end{matrix} \approx \begin{matrix} \mathbf{U}_r \\ m \times r \end{matrix} \begin{matrix} \Sigma_r \\ r \times r \end{matrix} \begin{matrix} \mathbf{V}_r^T \\ r \times n \end{matrix}$$

It can be shown that $\tilde{\mathbf{M}} = \mathbf{U}_r \Sigma_r \mathbf{V}_r^T$ is the “best” possible rank- r approximation to \mathbf{M} in the sense that it minimises the Frobenius Norm $\|\tilde{\mathbf{M}} - \mathbf{M}\|_F$

(Frobenius Norm of \mathbf{A} is just the square root of the sum of all the elements of \mathbf{A} squared)

What else is SVD good for?

- Pseudoinverse
 - $\mathbf{A}^+ = \mathbf{V}\Sigma^{-1}\mathbf{U}^T$
 - use to solve $\mathbf{Ax} = b$ for x where $\|\mathbf{Ax} - b\|_2$ is minimised: $x = \mathbf{A}^+ b$
(i.e. solve least squares regression!)
- Solving homogeneous linear equations
 - $\mathbf{Ax} = 0$
- Data mining
 - e.g. model based CF recommender systems, latent factors, ...
 - We'll see lots more in coming lectures

Computation of EVD and SVD

Classical EVD/SVD algorithms I

- All general E.V. algorithms are iterative
 - “Power Iteration”
$$b_{k+1} = \frac{Ab_k}{\|Ab_k\|}$$
 - Iterative process to find the biggest E.V.
 - Repeated for each E.V. by rotating and truncating the input to remove the effect of the previous E.V.
- “Arnoldi Iteration” and the “Lanczos Algorithm”
 - More efficient variants of Power Iteration
 - use the “Gram-Schmidt” process to find the orthonormal basis of the top-r E.Vs

Classical EVD/SVD algorithms II

- Those algorithms are good for finding the **largest r E.V.s or S.V.s**
 - Lots of standard efficient implementations in LAPACK/ ARPACK
 - including for relatively large, sparse matrices
 - **but not really massive matrices...**
 - Variations exist for finding the **smallest non-zero E.V.s**

Summary

- Covariance measures the “shape” of data by measuring how different dimensions change together.!
- The principle axes are a basis, aligned such they describe most directions of greatest variance.!
- The Eigendecomposition of the covariance matrix produces pairs of eigenvectors (corresponding to the principal axes) and eigenvalues (proportional to the variance along the respective axis).!
- PCA aligns data with its principal axes, and allows dimensionally reduction by discounting axes with low variance.
- SVD is a general matrix factorisation tool with lots of uses