Linear Algebra and Unsupervised Dimensionality Reduction via PCA

David I. Inouye

Linear Algebra: The Foundation of High-Dimensional Transformations and Modern AI

From Single Numbers to High-Dimensional Worlds

In your early math education, the focus was primarily on **scalars** (single numbers) and functions that operate on them. You learned to reason in one dimension.

However, the world of AI and Machine Learning is fundamentally **high-dimensional**. Data is almost never a single number; it's a **vector**.

- **Images:** A grid of pixels flattened into a vector (e.g., a 100x100 image is a 10,000-dimensional vector).
- **Text:** Sentences represented as vectors in "embedding" spaces.
- **User Data:** A person's preferences represented as a vector of ratings.

The Challenge of High Dimensions

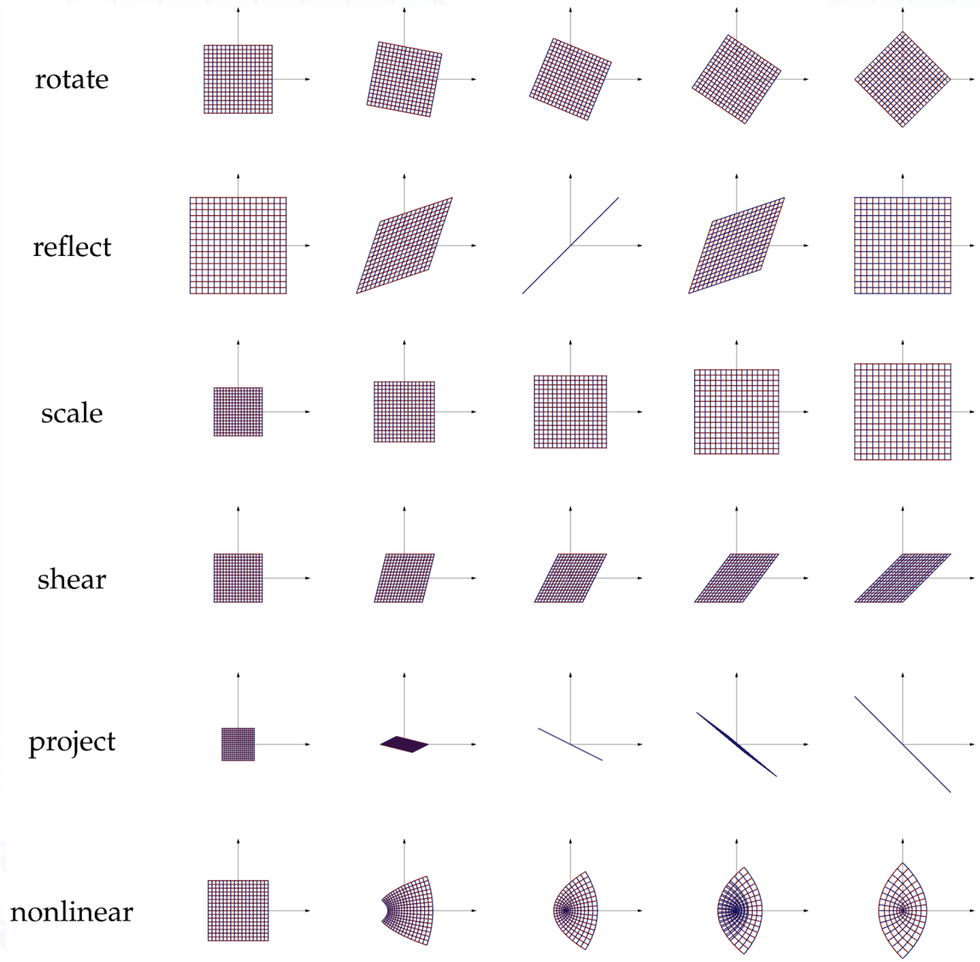
Our intuition is well-suited for 2D or 3D space, but it often breaks down in high dimensions. Concepts like distance, volume, and shape behave differently than we might expect.

- **The Goal:** We need a formal language and a core set of tools to manipulate and reason about high-dimensional data.
- **The Solution: Linear Algebra** provides this language.

At its heart, linear algebra is the study of **vectors** and the **linear transformations** that act upon them.

What is a Matrix Multiplication, Really?

- The fundamental operation in linear algebra is **matrix multiplication**.
- It's not just a mechanical process; it's a way to apply a **linear transformation** to data. 💪
- A matrix A is an operator that transforms an input vector x into an output vector y via the equation $y = Ax$.
- These transformations include operations like **rotating, scaling, and shearing** the data space itself.
- While seemingly simple, these linear operators are the fundamental building blocks for creating far more complex transformations in high dimensions.



Linear operators (e.g., matrix multiplication) can scale, rotate, and shear the original space and are the simple building block for complex non-linear transformations (bottom).

Eigenvalues
Eigenvectors
Solution
B

Linear Transformations Enable Understanding Invertibility in High Dimensions

Linear algebra helps us understand when transformations preserve or discard information.

- **Reversible (Invertible) Transformations:** If a matrix is **invertible**, you can apply an inverse transformation to perfectly recover the original vector. No information is lost.
- **Irreversible (Singular) Transformations:** If a matrix is **not invertible** (i.e., singular), the transformation is a one-way street. Information is lost.
- Projecting to a **lower dimension** is an inherent information loss; you cannot perfectly reconstruct the original data. Think of collapsing a 3D object into a 2D shadow.
- Projecting to a **higher dimension** doesn't create new information, but it can rearrange data into a space where patterns become easier to find, especially before applying a non-linear function.

The Building Block of Modern AI

Virtually all modern AI models are built upon matrix multiplication as their fundamental computational unit.

- **Two-Layer MLP:** A simple but powerful neural network.
  - $y = W_2\sigma(W_1x + b_1) + b_2$
- **Attention (Transformers/LLMs):** The mechanism that allows models to weigh the importance of different inputs.
  - $\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V$
- **Convolution (Images/Audio):** A specialized operation for grid-like data.
  - *As a convolution operator:* $Y = f * X$ (filter convoluted with image)
  - *As a matrix-vector product:* $y = A_f x$, where A_f is a special matrix built from the filter f and $x = \text{vec}(X)$ is the flattened image.

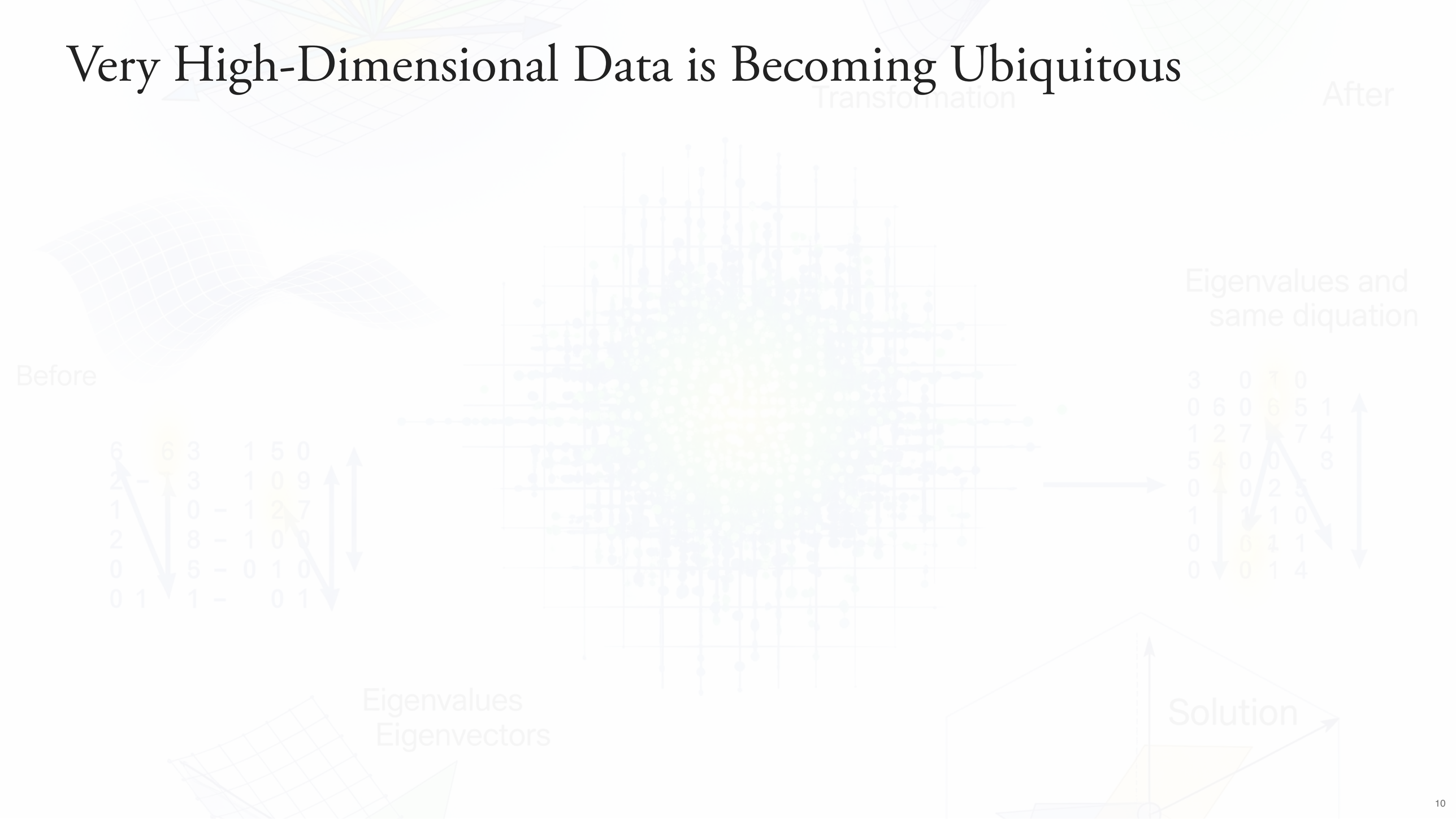
From Theory to Practice: Dimensionality Reduction

Now that we've established the fundamentals of how linear algebra manipulates data, let's apply it to a core task in unsupervised machine learning: **dimensionality reduction**.

We will now explore **Principal Component Analysis (PCA)**, a powerful technique that directly leverages linear transformations to find the most informative, lower-dimensional representation of your data. 🔎

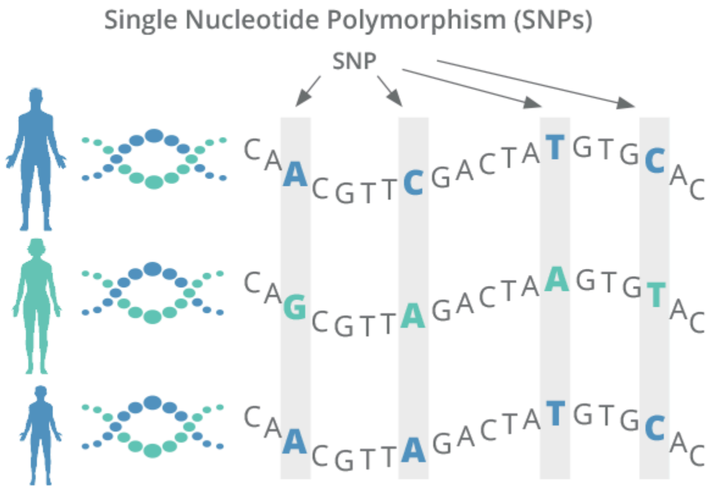
Unsupervised Dimensionality Reduction via PCA

Very High-Dimensional Data is Becoming Ubiquitous



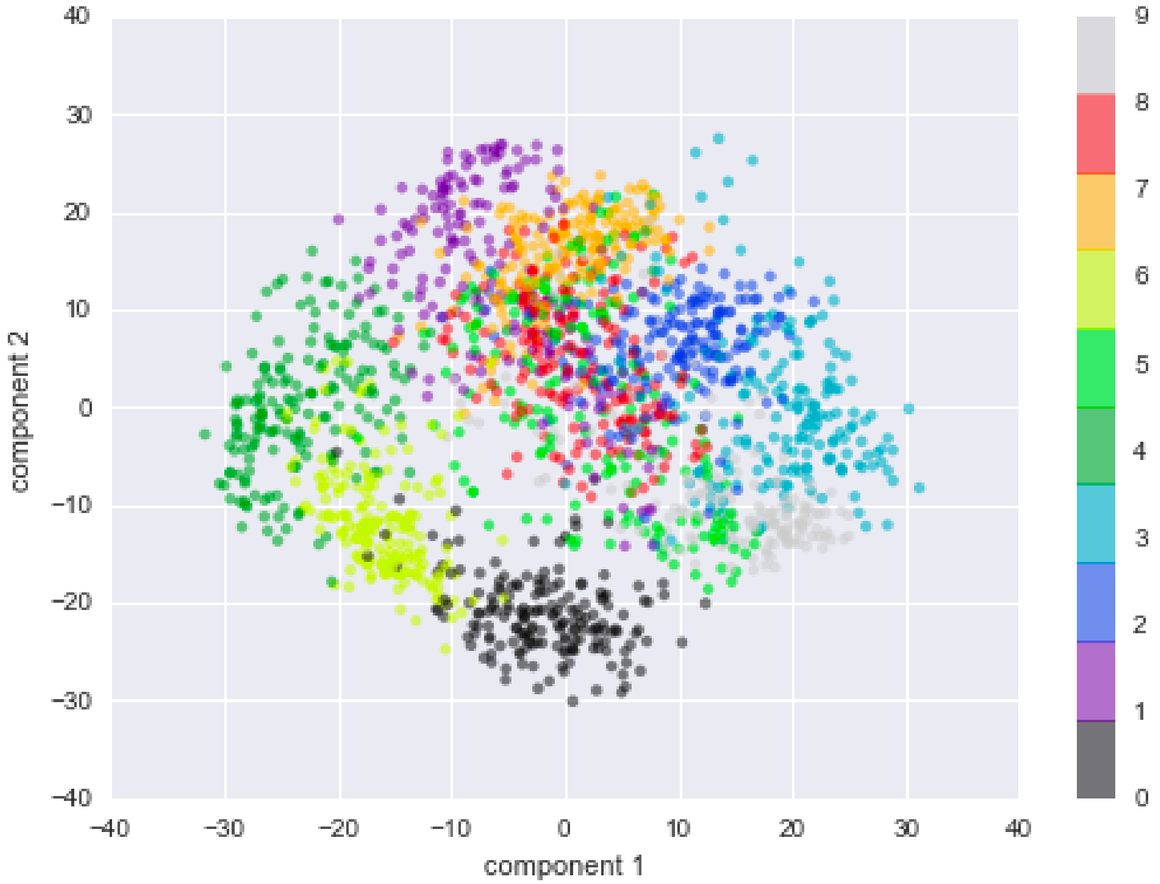
Why Dimensionality Reduction? Lower Computation Costs

- Suppose original dimension is large like $d = 100000$ (e.g., images, DNA sequencing, or text).
- If we reduce to $k = 100$ dimensions, the training algorithm can be sped up by $1000\times$.



Why Dimensionality Reduction? Visualization

- Allows 2D scatterplot visualizations even of high-dimensional data (2D projection of digits).



Why Dimensionality Reduction? Noise Reduction Via Reconstruction

Outline of Principal Components Analysis (PCA)

- Motivation for dimensionality reduction
- Formal PCA problem: Min reconstruction
- Derive PCA formulation for 1D
  - Least error 1D projection is orthogonal
  - Sum over all data points
- Solution is based on truncated SVD
- Alternative problem: Max variance

Math: PCA Minimizes The Linear Reconstruction Error Using Only $k \leq d$ Dimensions

PCA can be formalized as:

$$\min_{Z,W} \|X_c - ZW^T\|_F^2$$

where

- $X_c = X - \mathbf{1}_n \mu_x^T \in \mathbb{R}^{n \times d}$ (centered input data)
- $Z \in \mathbb{R}^{n \times k}$ (latent representation or "scores")
- $W^T \in \mathbb{R}^{k \times d}$ (principal components)
- $w_s^T w_t = 0, w_s^T w_s = \|w_s\|_2^2 = 1, \forall s, t \neq s$ (orthogonal constraint)

Review of Linear Algebra and Introduction to Numpy Python Library

See Jupyter notebook, which can be opened and run in Google Colab.

Transformation

After

Before

Eigenvalues and
same diquation

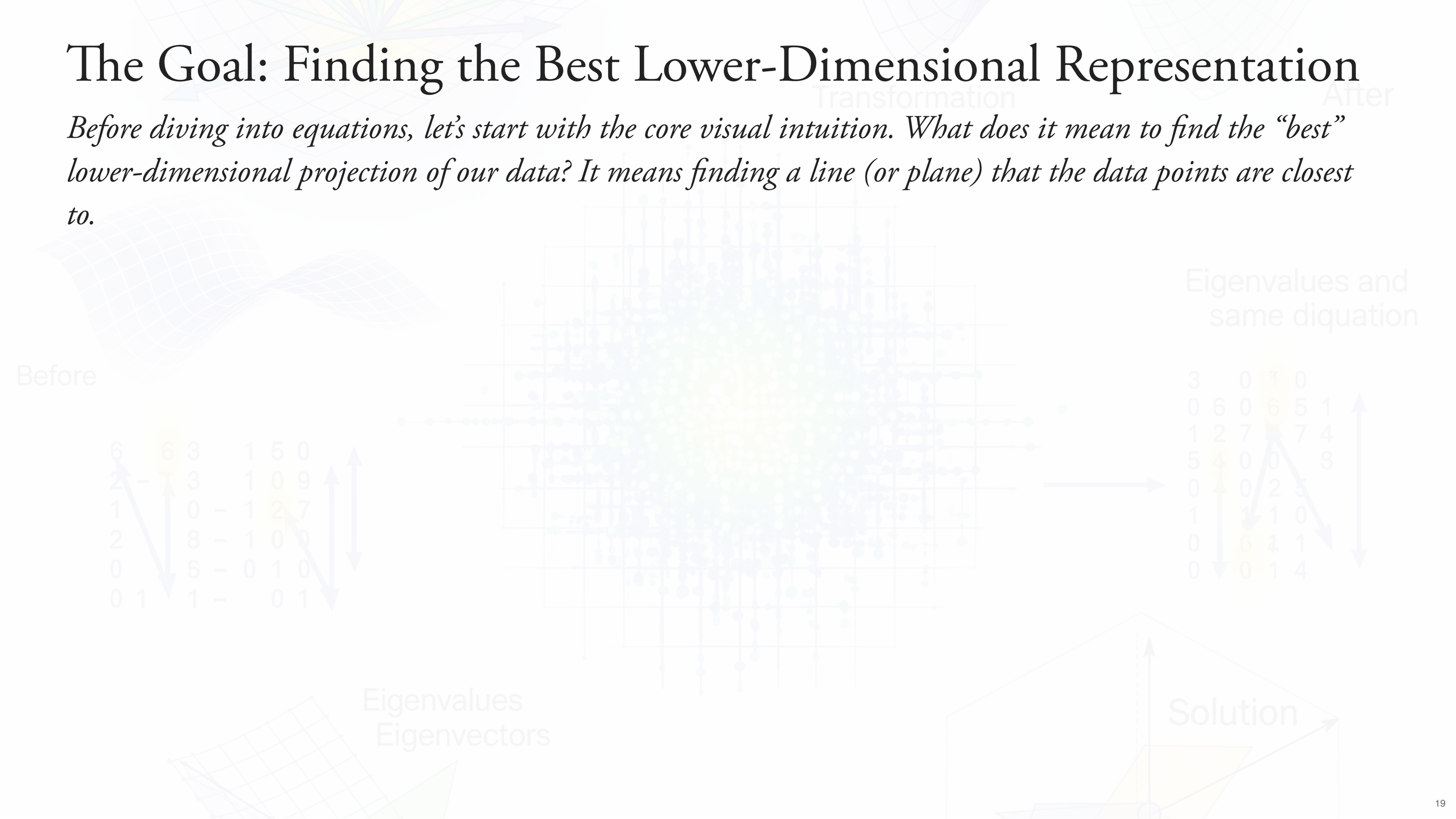
Eigenvalues
Eigenvectors

Solution

PCA Math and Intuition: A Reorganized Flow

The Goal: Finding the Best Lower-Dimensional Representation

Before diving into equations, let's start with the core visual intuition. What does it mean to find the "best" lower-dimensional projection of our data? It means finding a line (or plane) that the data points are closest to.



Intuition: Principal Component Analysis Finds The Best Linear Projection Onto a Lower-Dimensional Space

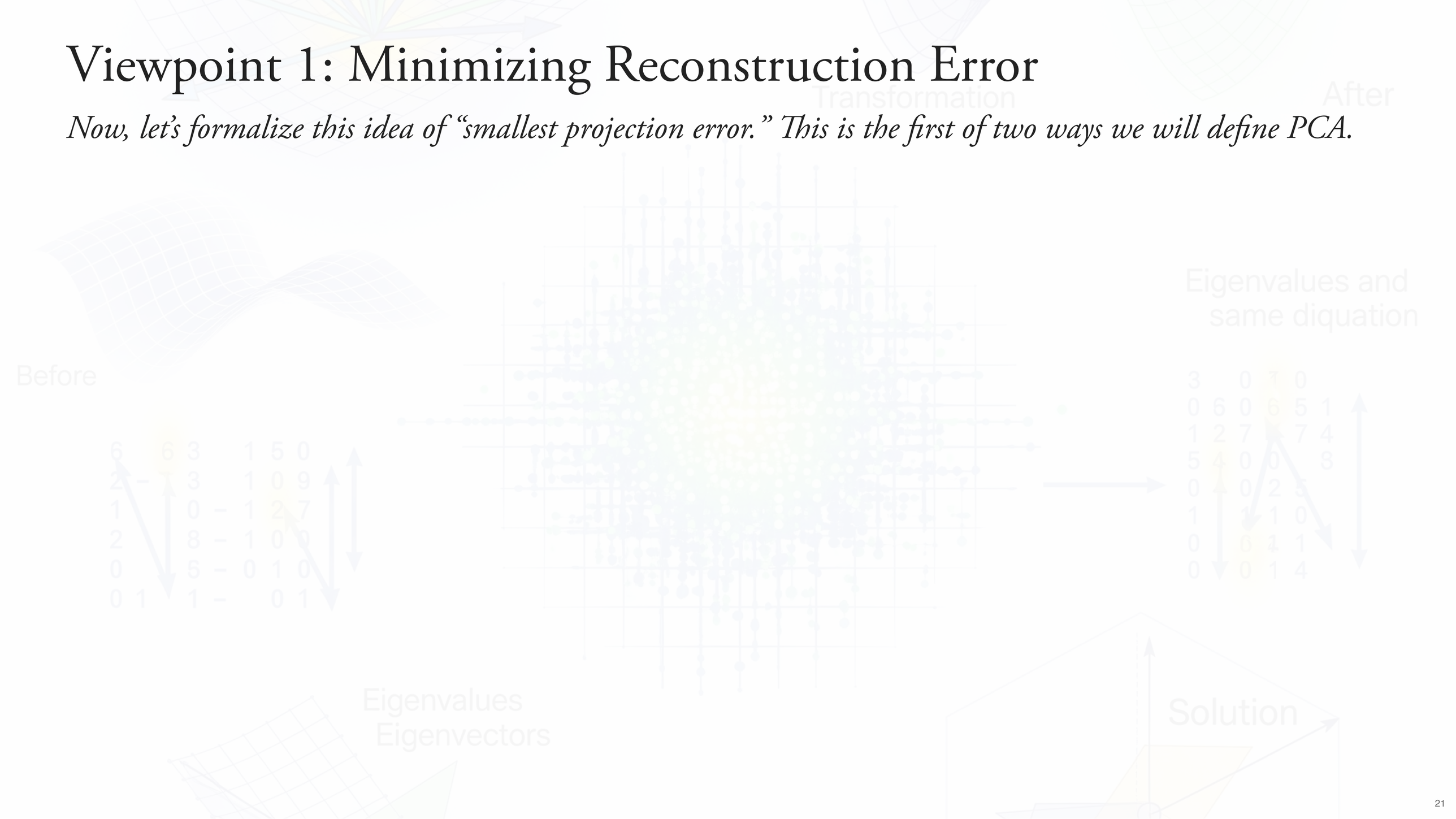

Plot showing 2D data points (blue) being projected onto a 1D line. The red lines indicate the projection error.

- 2D to 1D projection: Red lines show the projection error onto 1D lines.
- PCA finds the line that has the smallest projection error (in this example, when it aligns with the purple).

$$\min_{w:\|w\|_2=1} \|X_c - zw^T\|_F^2 \quad \text{where} \quad z = X_c w$$

Viewpoint 1: Minimizing Reconstruction Error

Now, let's formalize this idea of "smallest projection error." This is the first of two ways we will define PCA.



Math: PCA Minimizes The Linear Reconstruction Error Using Only $k \leq d$ Dimensions

PCA can be formalized as:

$$\min_{Z,W} \|X_c - ZW^T\|_F^2$$

where

- $X_c = X - \mathbf{1}_n \mu_x^T \in \mathbb{R}^{n \times d}$ (centered input data)
- $Z \in \mathbb{R}^{n \times k}$ (latent representation or "scores")
- $W^T \in \mathbb{R}^{k \times d}$ (principal components)
- $w_s^T w_t = 0, w_s^T w_s = \|w_s\|_2^2 = 1, \forall s, t \neq s$ (orthogonal constraint)

Math: PCA Minimizes The Linear Reconstruction Error Using Only $k \leq d$ Dimensions

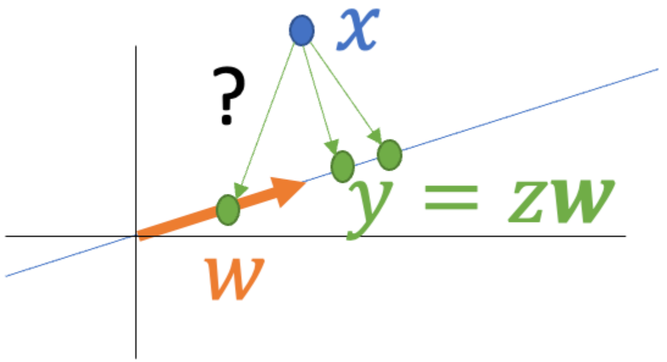
$$\min_{Z \in \mathbb{R}^{n \times k}, W \in \mathbb{R}^{d \times k}} \|X_c - ZW^T\|_F^2 \quad \text{s.t.} \quad W^T W = I_k$$

- Let's stare at this equation some more 🙂
- Why is this dimensionality reduction?
- What does the orthogonal constraint mean?
- Why minimize the squared Frobenius norm?
- $\|X_c - ZW^T\|_F^2 = \sum_{i=1}^{n} \|x_i^T - z_i^T W^T\|_2^2 = \sum_{i=1}^{n} \|x_i - W z_i\|_2^2$
- For analysis, let's simplify to a single dimension (i.e., $k = 1$):
  - $\sum_{i=1}^{n} \|x_i - z_i w\|_2^2$ where z_i is a scalar

What is The Best Projection Given a Fixed Subspace (Line in 1D Case)?

- If we are given w, what is the best z (i.e. minimum reconstruction error) for a given x?

$$\min_z \|x - zw\|_2^2$$



- The orthogonal projection!
  - $z = x^T w = \|x\|\|w\| \cos\theta = \|x\| \cos\theta$
  - $z = \|x\| \cos\theta = \text{hyp} \cdot \frac{\text{adj}}{\text{hyp}} = \text{adj}$
- zw is a scaled vector along the line defined by w.

Thus, We Can Simplify to Only Minimizing Over W

$$\min_{z,w:\|w\|_2=1} \sum_{i=1}^{n} \|x_i - z_i w\|_2^2 = \min_{w:\|w\|_2=1} \sum_{i=1}^{n} \|x_i - (x_i^T w)w\|_2^2$$

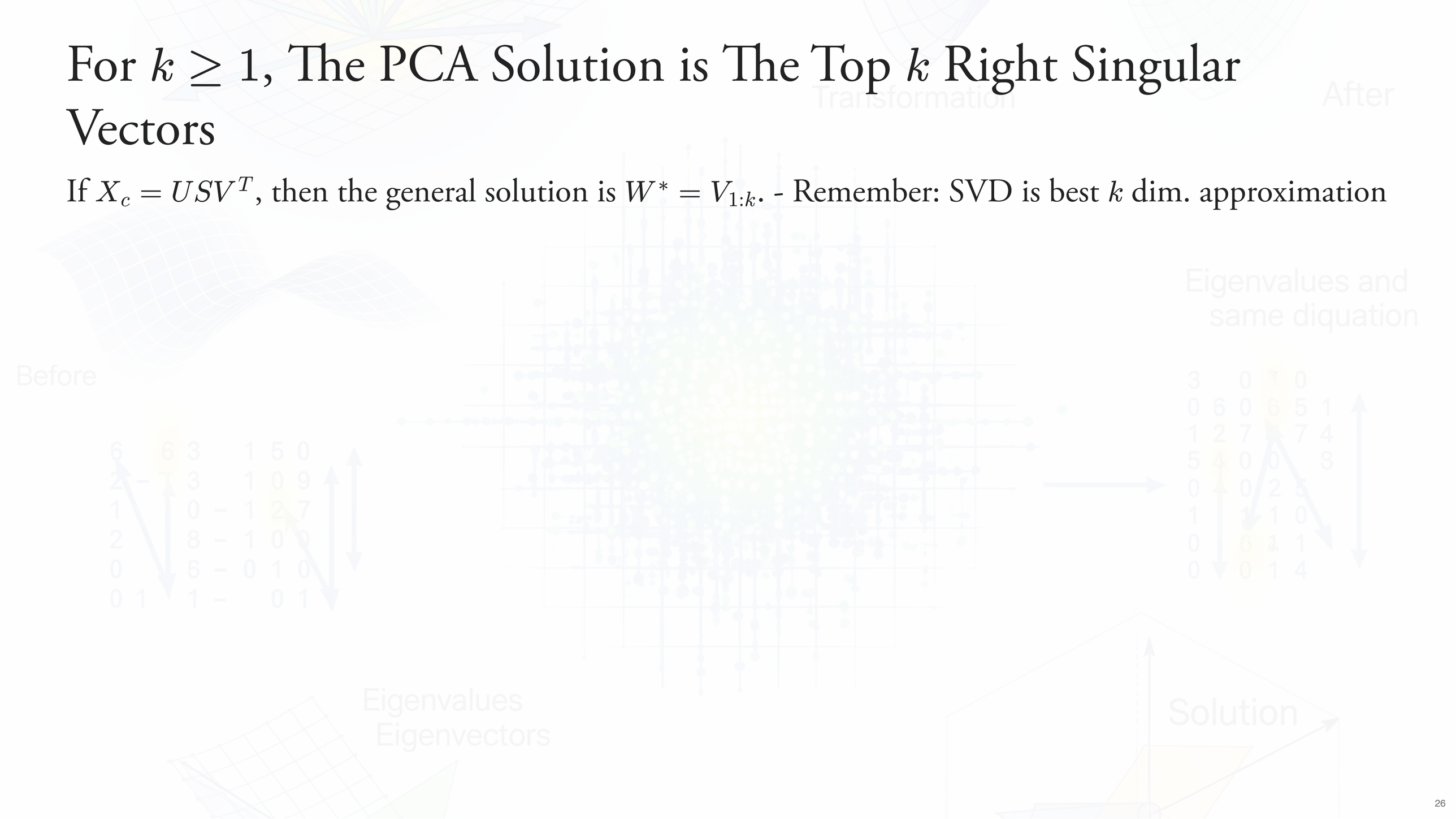
Now we can return to the Frobenius norm:

$$\min_{w:\|w\|_2=1} \|X_c - zw^T\|_F^2 \quad \text{where} \quad z = X_c w$$

- What is zw^T? Have we seen something like this before?
- This is the best low-rank approximation to X_c, which is given by the SVD!
- $w = v_1$ and $z = \sigma_1 u_1$, where σ_1, u_1, v_1 are the first singular value, left singular vector and right singular vector respectively.

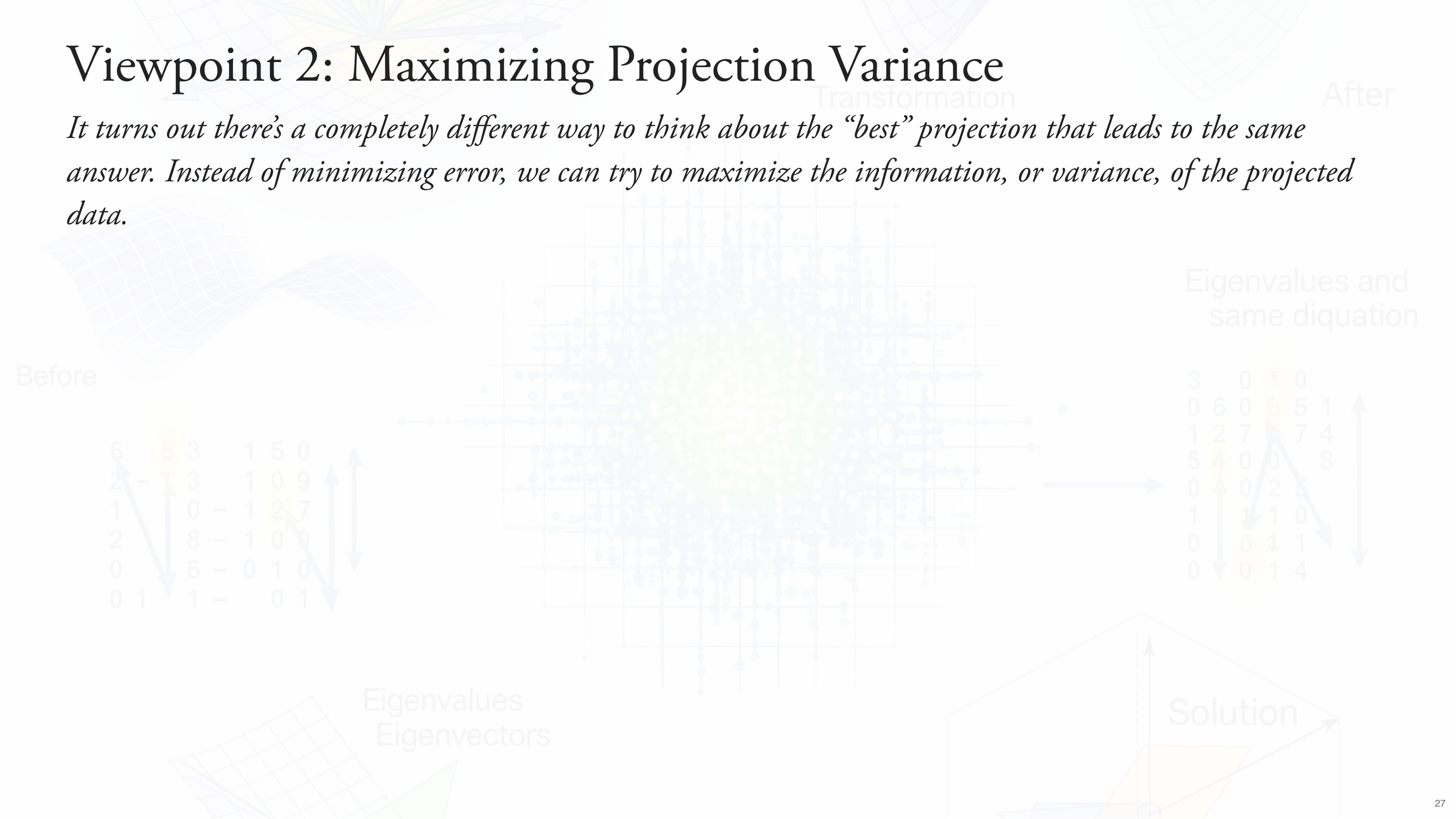
For $k \geq 1$, The PCA Solution is The Top k Right Singular Vectors

If $X_c = USV^T$, then the general solution is $W^* = V_{1:k}$. - Remember: SVD is best k dim. approximation

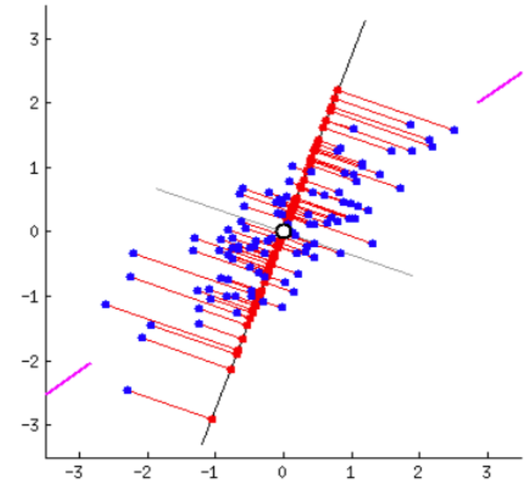


Viewpoint 2: Maximizing Projection Variance

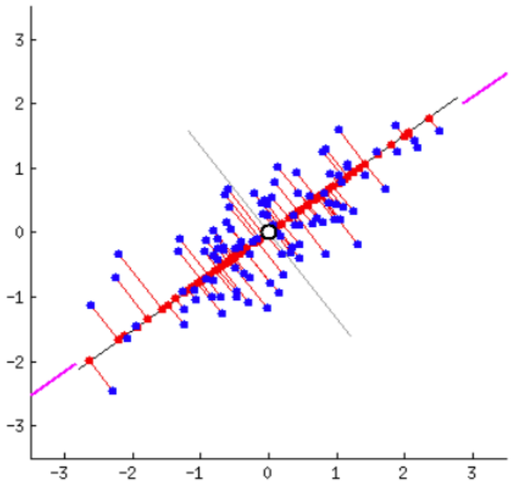
It turns out there's a completely different way to think about the "best" projection that leads to the same answer. Instead of minimizing error, we can try to maximize the information, or variance, of the projected data.



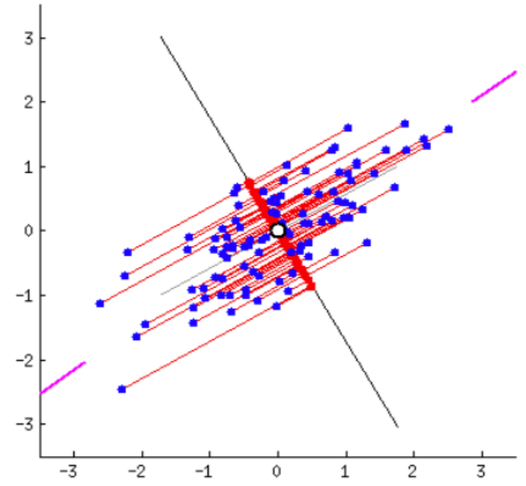
Minimizing Reconstruction Error (Red Lines) is Equivalent to Maximizing The Variance of Projection (Spread of Red Points)



Max reconstruction error, Min variance



Min reconstruction error, Max variance



Max reconstruction error, Min variance

Proving the Equivalence in 1D (Part 1)

This visual equivalence isn't just a coincidence. Let's formally prove it for the 1D case. The first step is to rewrite the squared reconstruction error for a single point x_i in a more useful form.

$$\begin{aligned}\|x_i - (x_i^T w)w\|_2^2 &= (x_i - (x_i^T w)w)^T(x_i - (x_i^T w)w) \\ &= x_i^T x_i - 2(x_i^T w)w^T x_i + (x_i^T w)^2 w^T w \\ &= \|x_i\|^2 - 2(x_i^T w)^2 + (x_i^T w)^2\|w\|^2 \\ &= \|x_i\|^2 - (x_i^T w)^2\end{aligned}$$

Proving the Equivalence in 1D (Part 2)

Now we plug that result back into the total error minimization problem. Notice that minimizing the error is equivalent to maximizing the term we are subtracting.

$$
\begin{aligned}
\arg\min_w \sum_i \|x_i - (x_i^T w)w\|_2^2 &= \arg\min_w \sum_i \left(\|x_i\|^2 - (x_i^T w)^2\right) \\
&= \arg\min_w \sum_i -(x_i^T w)^2 \\
&= \arg\max_w \frac{1}{n} \sum_i (x_i^T w)^2 \\
&= \arg\max_w \frac{1}{n} \sum_i z_i^2 \\
&= \arg\max_w \sigma_z^2
\end{aligned}
$$

Note z is already centered so mean of squares is variance.

Reframing the Problem: Maximizing Variance

We've shown that minimizing reconstruction error is the same as maximizing the variance of the projected points. Let's formalize this new maximization problem. The term σ_z^2 can be rewritten using the data covariance matrix $\hat{\Sigma}$.

Let's rewrite this last term:

$$\sigma_z^2 = \frac{1}{n}\sum_i (z_i)^2 = \frac{1}{n}\sum_i (x_i^T w)^2 = \frac{1}{n}(X_c w)^T (X_c w) = w^T \left(\frac{1}{n} X_c^T X_c\right) w = w^T \hat{\Sigma} w$$

- Thus, our problem can be formulated as:

$$\max_{w:\|w\|=1} \; w^T \hat{\Sigma} w$$

- The solution is the eigenvector q_1 of $\hat{\Sigma} = Q \Lambda Q^T$ corresponding to the largest eigenvalue λ_1. $w^* = q_1$

Generalizing the Max Variance View to k Dimensions

This max-variance formulation generalizes neatly from one dimension to k dimensions. But what does the new objective function mean?

Minimize reconstruction error

$$\min_{W:W^TW=I_k} \|X_c - (X_cW)W^T\|_F^2$$

Alternative problem

$$\max_{W:W^TW=I_k} \mathrm{Tr}(W^TX_c^TX_cW)$$

$$\begin{aligned}
\mathrm{Tr}(W^TX_c^TX_cW) &= \mathrm{Tr}((X_cW)^T(X_cW)) \\
&= \mathrm{Tr}(Z^TZ) \\
&= \sum_{j=1}^{k} z_j^Tz_j \\
&= n\sum_{j=1}^{k}\frac{1}{n}\sum_{i=1}^{n} z_{i,j}^2 \\
&= n\sum_{j=1}^{k}\sigma_{z,j}^2
\end{aligned}$$

where $\sigma^2_{z,j}$ is the variance of the j-th latent dimension. - Given this, what does the optimization problem mean? - Answer: This objective maximizes the sum of variances of the data projected onto W.

For $k > 1$, We Maximize The Sum of Variances for Each Latent Dimension

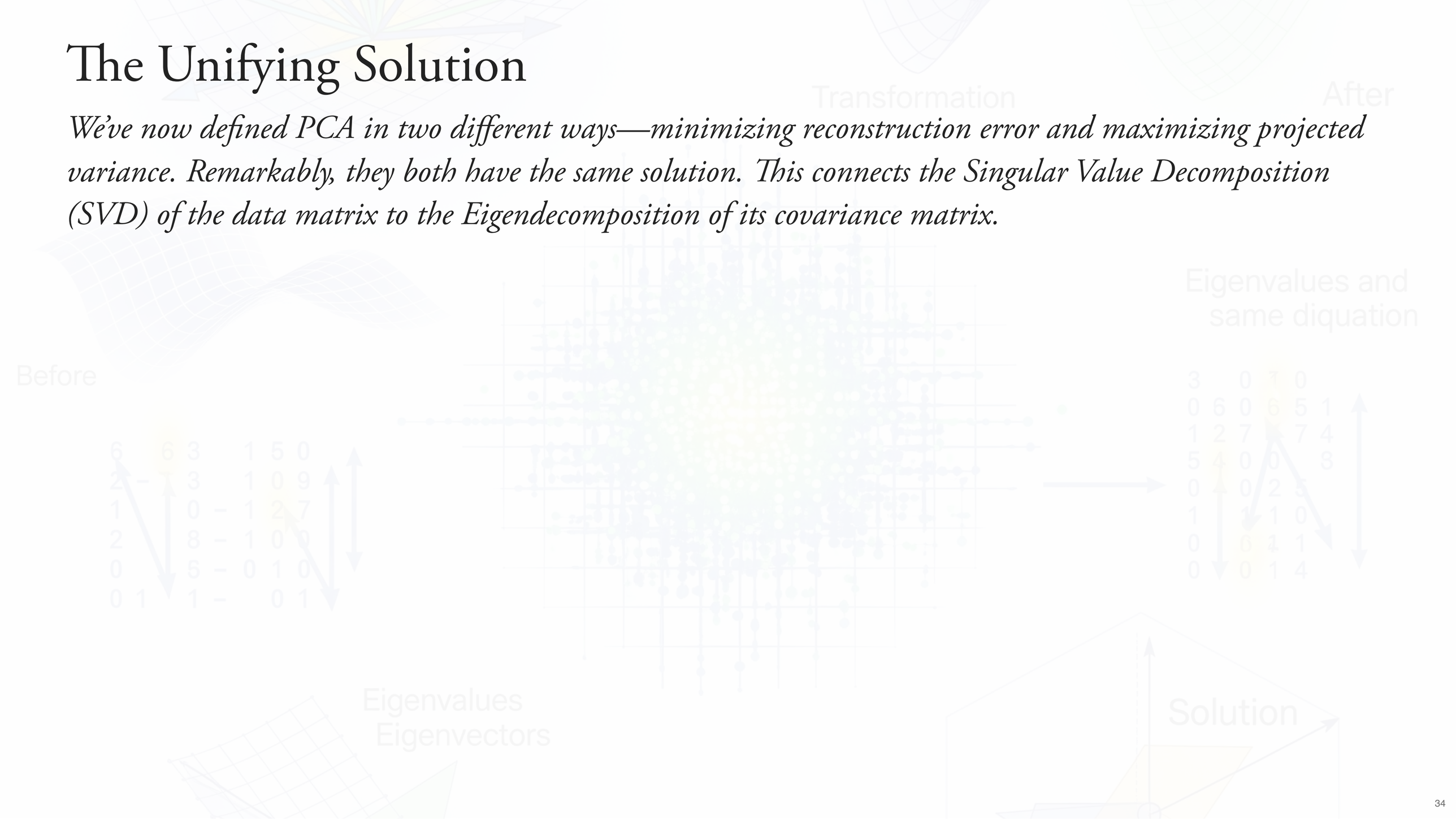
More generally we can formulate this as:

$$\begin{aligned}\max_{W:W^TW=I_k} \sum_{j=1}^{k} \sigma_{z_j}^2 &= \max_{W:W^TW=I_k} \sum_{j=1}^{k} w_j^T \hat{\Sigma} w_j \\ &= \max_{W:W^TW=I_k} \mathrm{Tr}(W^T \hat{\Sigma} W) \\ &= \max_{W:W^TW=I_k} \frac{1}{n} \mathrm{Tr}(W^T X_c^T X_c W)\end{aligned}$$

- The solution is the top k eigenvectors of $\hat{\Sigma} = Q \Lambda Q^T$. $W^* = Q_{1:k}$

The Unifying Solution

We've now defined PCA in two different ways—minimizing reconstruction error and maximizing projected variance. Remarkably, they both have the same solution. This connects the Singular Value Decomposition (SVD) of the data matrix to the Eigendecomposition of its covariance matrix.



Equivalent Solutions: The Solution to Both Problems is The Top k Right Singular Vectors of X_c

Minimize reconstruction error

$$\min_{W:W^TW=I_k} \|X_c - (X_cW)W^T\|_F^2$$

- Singular value decomposition (SVD) of X_c $= USV^T$
- Solution: $W^* = V_{1:k}$

Maximize variance of latent projection (equivalent solution)

$$\max_{W:W^TW=I_k} \text{Tr}(W^T\hat{\Sigma}W)$$

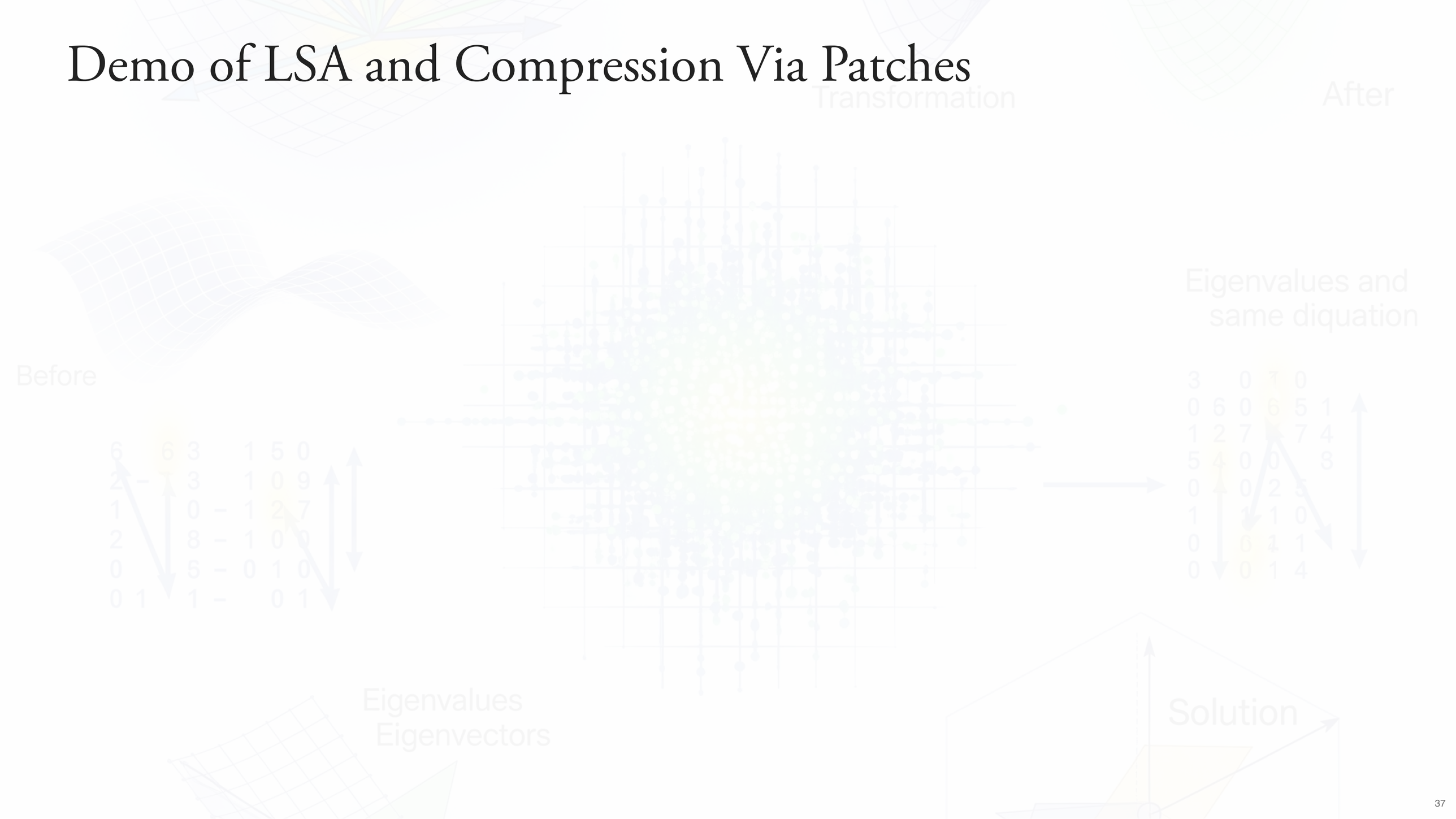
where $\hat{\Sigma} := \frac{1}{n}X_c^TX_c$ is the covariance matrix

- $n\hat{\Sigma} = X_c^TX_c = (USV^T)^T(USV^T)$
- $= VSU^TUSV^T = VS(U^TU)SV^T = VS^2V^T$
- $= Q\Lambda Q^T$
- Solution: $W^* = Q_{1:k} \equiv V_{1:k}$!

Demo of PCA Via Sklearn (Time Permitting)

- Random projections vs PCA projections
- Visualizations of
  - Minimum reconstruction error
  - Maximum variance
- Explained variance based on k
- Code examples
  - Digits
  - Eigenfaces

Demo of LSA and Compression Via Patches



Questions?


Transformation
After
Eigenvalues and
same diquation
Before
Eigenvalues
Eigenvectors
Solution