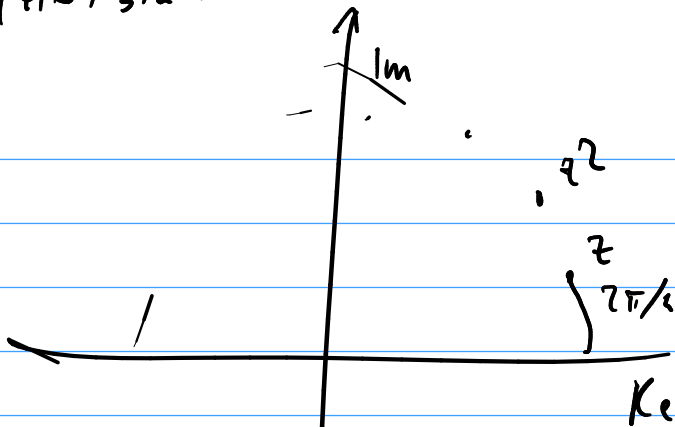


$$e^{i\varphi} = \cos(\varphi) + i \sin \varphi$$



$$z = e^{2\pi i/n}$$

$$n = 7$$

$$|z| = 1$$

# Overview

- SVB

↳ lsq

↳ norms

↳ CNA

- Interpolation

$$PA = L U$$

↑

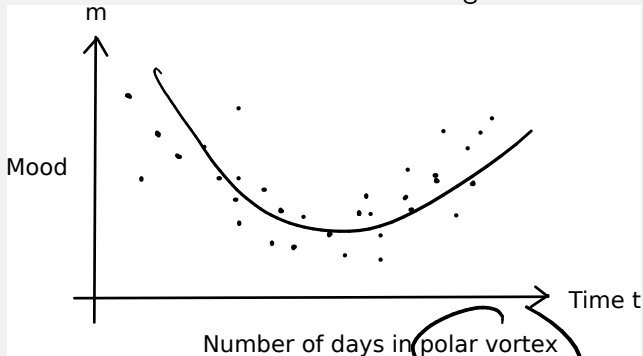
$$A = P^T L U$$

↑

"PLU" factorization

# Fitting a Model to Data

How can I fit a model to measurements? E.g.:



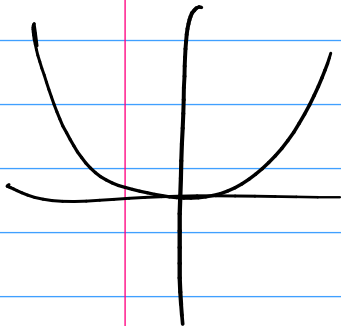
$$p(t) = \alpha_0 + \alpha_1 t + \alpha_2 t^2$$

$$V \begin{matrix} \vec{\alpha} \\ \parallel \\ \vec{y} \end{matrix}$$

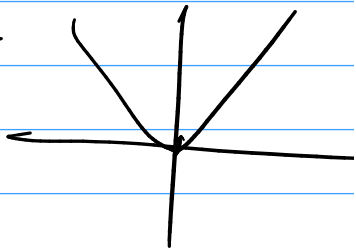
$$Ax \approx b \rightarrow \min_x \|Ax - b\|_2$$



$l_2$ -minimization



$l_1$ -



## Demo: Data Fitting using Least Squares

## Meaning of the Singular Values

What do the singular values mean? (in particular the first/largest one)

$$A = U \begin{pmatrix} \sigma_1 & & \\ & \ddots & \\ & & \sigma_n \end{pmatrix} V^T$$

$$\|A\|_2 = \max_{\|x\|_2=1} \|Ax\|_2$$

$$= \max_{\|x\|_2=1} \|U \Sigma V^T x\|_2$$

$$\begin{aligned}
 & - \max_{\|x\|_2=1} \|\Sigma V^T x\|_2 & \left. \begin{array}{l} \rightarrow \|x\|_2=1 \\ \|V^T x\|_2=1 \end{array} \right\} \\
 & = \max_{\|V^T x\|_2=1} \|\Sigma V^T x\|_2 \\
 & = \max_{\|y\|_2=1} \|\Sigma y\|_2 \\
 & \quad \hat{y} = V^T x \\
 & = \|\Sigma\|_2 = \sigma_1
 \end{aligned}$$



$$\|Qx\|_2^2 = \|x\|_2^2$$

## Condition Numbers

How would you compute a 2-norm condition number?

$$\text{cond}_2(A) = \|A\|_2 \cdot \|A^{-1}\|_2$$

$$= \sigma_1 \cdot \frac{1}{\sigma_n}$$

$$A^{-1} = V \Sigma^{-1} U^T = V \begin{pmatrix} 1/\sigma_1 & & \\ & \dots & \\ & & 1/\sigma_n \end{pmatrix} U^T$$

↑

# Outline

Python, Numpy, and Matplotlib  
Making Models with Polynomials  
Making Models with Monte Carlo  
Error, Accuracy and Convergence  
Floating Point  
Modeling the World with Arrays  
    The World in a Vector  
    What can Matrices Do?  
    Graphs  
    Sparsity  
Norms and Errors  
The 'Undo' Button for Linear Operations: LU  
Repeating Linear Operations:  
Eigenvalues and Steady States  
Eigenvalues: Applications

Approximate Undo: SVD and Least Squares

## SVD: Applications

Solving Funny-Shaped Linear Systems  
Data Fitting  
Norms and Condition Numbers  
Low-Rank Approximation

Interpolation

Iteration and Convergence

Solving One Equation

Solving Many Equations

Finding the Best: Optimization in 1D

Optimization in  $n$  Dimensions

# SVD as Sum of Outer Products

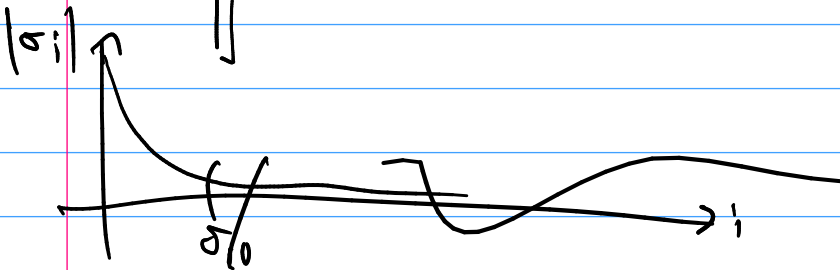
What's another way of writing the SVD?

$$\begin{aligned} A = U \Sigma V^T &= \begin{matrix} \swarrow m \times h \\ \begin{matrix} \begin{matrix} | & & | \\ u_1 & \dots & u_m \\ | & & | \end{matrix} \\ \hline \begin{matrix} | & & | \\ \vdots & & \vdots \\ | & & | \end{matrix} \\ \hline \begin{matrix} | & & | \\ \vdots & & \vdots \\ | & & | \end{matrix} \end{matrix} \end{matrix} \begin{matrix} \begin{matrix} \begin{matrix} | & & | \\ \sigma_1 & \dots & \sigma_n \\ | & & | \end{matrix} \\ \hline \begin{matrix} | & & | \\ 0 & \dots & 0 \\ | & & | \end{matrix} \\ \hline \begin{matrix} | & & | \\ \sigma_1 v_1 & & \\ | & & \\ \sigma_n v_n & & \end{matrix} \end{matrix} \begin{matrix} \begin{matrix} | & & | \\ -v_1 & & \\ | & & \\ -v_n & & \end{matrix} \end{matrix} \end{matrix} \\ &= \begin{matrix} \begin{matrix} | & & | \\ u_1 & \dots & u_m \\ | & & | \end{matrix} \\ \hline \begin{matrix} | & & | \\ \vdots & & \vdots \\ | & & | \end{matrix} \\ \hline \begin{matrix} | & & | \\ \vdots & & \vdots \\ | & & | \end{matrix} \end{matrix} \begin{matrix} \begin{matrix} | & & | \\ \sigma_1 & & \\ | & & \\ 0 & \dots & 0 \\ | & & \\ \sigma_n & & \end{matrix} \end{matrix} \begin{matrix} \begin{matrix} | & & | \\ -v_1 & & \\ | & & \\ -v_n & & \end{matrix} \end{matrix} \\ &= \begin{matrix} \begin{matrix} | & & | \\ u_1 & \dots & u_m \\ | & & | \end{matrix} \\ \hline \begin{matrix} | & & | \\ \vdots & & \vdots \\ | & & | \end{matrix} \\ \hline \begin{matrix} | & & | \\ \vdots & & \vdots \\ | & & | \end{matrix} \end{matrix} \begin{matrix} \begin{matrix} | & & | \\ \sigma_1 v_1 & & \\ | & & \\ \sigma_n v_n & & \end{matrix} \end{matrix} \end{matrix}$$

$$= \underbrace{\sigma_1 u_1 v_1^T}_{\text{rank 1}} + \dots + \underbrace{\sigma_h u_h v_h^T}_{\text{rank 1}}$$

$$A_1 = \sigma_1 u_1 v_1^T$$

$$= \begin{matrix} \sigma_1 \\ \downarrow \end{matrix} \begin{matrix} \left[ u_1 \right] \\ \left[ v_1 \right] \end{matrix}$$



## Low-Rank Approximation (I)

What is the *rank* of  $\sigma_1 \mathbf{u}_1 \mathbf{v}_1^T$ ?

|

What is the *rank* of  $\sigma_1 \mathbf{u}_1 \mathbf{v}_1^T + \sigma_2 \mathbf{u}_2 \mathbf{v}_2^T$ ?

2

**Demo:** Image Compression

## Low-Rank Approximation

What can we say about the low-rank approximation

$$A_k = \sigma_1 \mathbf{u}_1 \mathbf{v}_1^T + \cdots + \sigma_k \mathbf{u}_k \mathbf{v}_k^T$$

to

$$A = \sigma_1 \mathbf{u}_1 \mathbf{v}_1^T + \sigma_2 \mathbf{u}_2 \mathbf{v}_2^T + \cdots + \sigma_n \mathbf{u}_n \mathbf{v}_n^T?$$

For simplicity, assume  $\sigma_1 \geq \sigma_2 \geq \cdots \geq \sigma_n > 0$ .

$$\min_{\text{rank}(B)=k} \|A - B\|_2 = \|A - A_k\|_2$$

$$\text{rank}(B) = k$$

$$\min_{\text{rank}(B)=k} \|A - B\|_F = \|A - A_k\|_F$$

$$\text{rank}(B) = k$$

Eckart-Young Mirsky theorem



$$A = \begin{pmatrix} a_1 & \dots & a_n \end{pmatrix}$$

$$\approx \begin{pmatrix} u_1 & \dots & u_m \end{pmatrix} \underbrace{\begin{pmatrix} \sigma_1 & & \\ & \dots & \\ & & \sigma_{10} \end{pmatrix}} \begin{pmatrix} V^T \end{pmatrix}$$

# Outline

Python, Numpy, and Matplotlib  
Making Models with Polynomials  
Making Models with Monte Carlo  
Error, Accuracy and Convergence  
Floating Point  
Modeling the World with Arrays  
    The World in a Vector  
    What can Matrices Do?  
    Graphs  
    Sparsity  
Norms and Errors  
The 'Undo' Button for Linear Operations: LU  
Repeating Linear Operations:  
Eigenvalues and Steady States  
Eigenvalues: Applications

Approximate Undo: SVD and Least Squares

SVD: Applications

Solving Funny-Shaped Linear Systems  
Data Fitting  
Norms and Condition Numbers  
Low-Rank Approximation

**Interpolation**

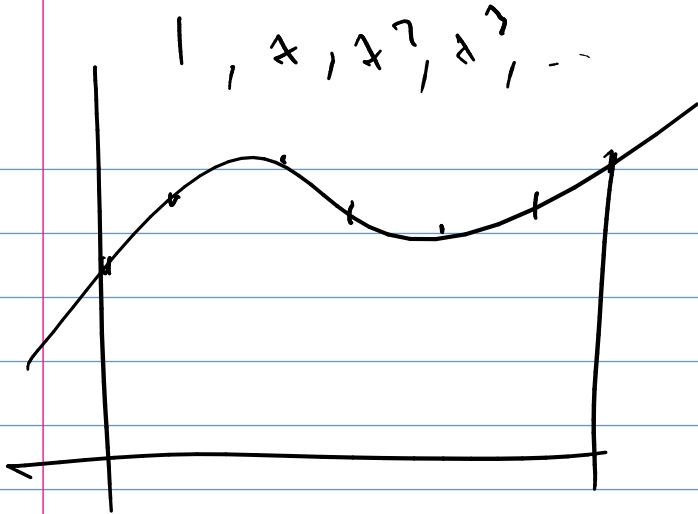
Iteration and Convergence

Solving One Equation

Solving Many Equations

Finding the Best: Optimization in 1D

Optimization in  $n$  Dimensions



## Recap: Interpolation

Starting point: Looking for a linear combination of functions  $\varphi_i$  to hit given data points  $(x_i, y_i)$ .

Interpolation becomes solving the linear system:

$$y_i = f(x_i) = \sum_{j=0}^{N_{\text{func}}} \alpha_j \underbrace{\varphi_j(x_i)}_{V_{ij}} \quad \Leftrightarrow \quad V\boldsymbol{\alpha} = \mathbf{y}.$$

Want unique answer: Pick  $N_{\text{func}} = N \rightarrow V$  square.

$V$  is called the (generalized) Vandermonde matrix.

Main lesson:

$$V \text{ (coefficients)} = \text{(values at nodes)}.$$