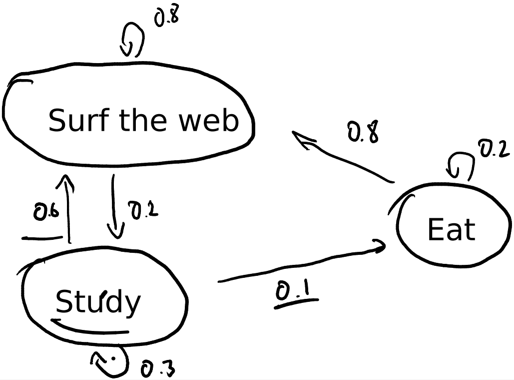
Overview

- Eigen: app
- SVD
- Csq

Markov chains and Eigenvalue Problems

Recall our example of a Markov chain:



$$A = \begin{array}{c|ccc} & \text{Study} & \text{Surf} & \text{Eat} \\ \hline S & 0.3 & & \\ S & 0.6 & & \\ E & 0.1 & & \end{array}$$

Suppose this is an accurate model of the behavior of the average student. :) How likely are we to find the average student in each of these states?

$$A\vec{p} = \vec{p}$$

Demo: Finding an equilibrium distribution using the power method

Understanding Time Behavior

Many important systems in nature are modeled by describing the time rate of change of something.

- E.g. every bird will have 0.2 baby birds on average per year.
- But there are also foxes that eat birds. Every fox present decreases the bird population by 1 birds a year.

  Meanwhile, each fox has 0.3 fox babies a year. And for each bird present, the population of foxes grows by 0.9 foxes.

Set this up as equations and see if eigenvalues can help us understand how these populations will evolve over time.

$$\frac{\partial b}{\partial t} \quad \cancel{\frac{\Delta b}{\Delta t}} = 0.2b - 1f$$

$$\frac{\partial f}{\partial t} \quad \cancel{\frac{\Delta f}{\Delta t}} = 0.9b + 0.3f$$

$$\vec{s} = \begin{pmatrix} b \\ f \end{pmatrix}$$

$$\frac{\partial \vec{s}}{\partial t} = \overbrace{\begin{pmatrix} 0.2 & -1 \\ 0.9 & 0.3 \end{pmatrix}}^{A} \vec{s}$$

$$\vec{s}(t) = e^{\lambda t} \cdot \vec{s}_0$$

$$\lambda \cancel{e^{\lambda t}} \vec{s}_0 = A \cancel{e^{\lambda t}} \vec{s}_0$$

Demo: Understanding the birds and the foxes with eigenvalues

In-class activity: Eigenvalues 2

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

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

Singular Value Decomposition

What is the Singular Value Decomposition ('SVD')?

$$A = \underset{\text{orth.}}{U} \; \underset{\text{diagonal}}{\Sigma} \; \underset{\text{orth}}{V^T}$$

"Full":

$$\left.\begin{array}{l} A: m \times n \\ U \quad m \times m \\ \Sigma: m \times n \\ V^T: n \times n \end{array}\right] \text{full}$$

$Ax = b \qquad A = U\Sigma V^T$

$U\Sigma V^T x = b \quad \Big| \; U^T \cdot$

$\Sigma V^T x = U^T b \quad \Big| \; \Sigma^{-1}$

$V^T x = \Sigma^{-1} U^T b \quad \Big| \; V \cdot$

$x = V\left(\Sigma^{-1}\left(U^T b\right)\right)$

Computing the SVD

How can I compute an SVD of a matrix A?

A^TA : symmetric, pos. semidef.

$\rightarrow$ eigv. ≥ 0

$A^TA\vec{v}_i = \lambda_i \vec{v}_i$

$$V = \begin{pmatrix} | & & | \\ \vec{v}_1 & \cdots & v_n \\ | & & | \end{pmatrix}$$

$$A^TAV = V\begin{pmatrix} \lambda_i & & \\ & \ddots & \\ & & \lambda_1 \end{pmatrix}$$

$\sigma_i = \sqrt{\lambda_i}$ "singular values"

$$\Sigma = \begin{pmatrix} \sqrt{\lambda_1} & & \\ & \ddots & \\ & & \sqrt{\lambda_n} \end{pmatrix} = \begin{pmatrix} \sigma_1 & & \\ & \ddots & \\ & & \sigma_n \end{pmatrix}$$

$$A^T A V = V \Sigma^2 \Leftrightarrow V^{-1} A^T A V = \Sigma^2$$

eigenvectors of a symm. matrix are orth.: V orth.

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

$$A V \Sigma^{-1} = U$$

$$I \stackrel{!}{=} U^T U = \Sigma^{-1} \underbrace{V^{-1} A^T A V}_{\Sigma^2} \Sigma^{-1} = I$$

Demo: Computing the SVD

-

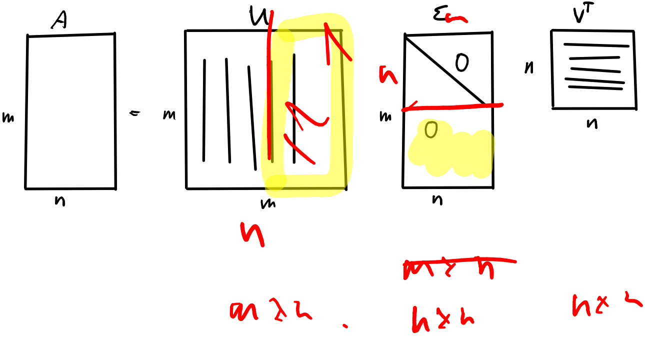
How Expensive is it to Compute the SVD?

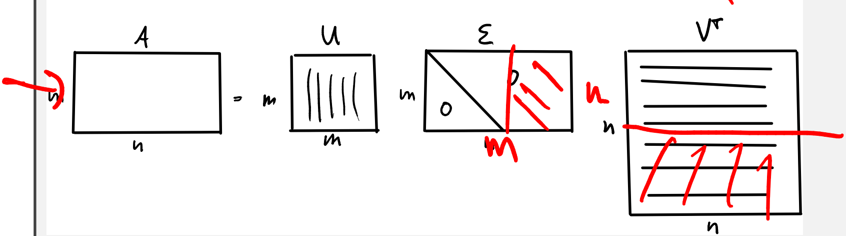
Demo: Relative Cost of Matrix Factorizations

‘Reduced’ SVD

'Reduced' SVD (II)

Is there a 'reduced' factorization for non-square matrices?





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

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

Solve Square Linear Systems

Can the SVD $A = U\Sigma V^T$ be used to solve *square* linear systems?
At what cost (once the SVD is known)?