Honors Numerical Analysis

Lecture 15

Computing the SVD

Recall: the SVD of <u>any</u> matrix $A \in \mathbb{R}^{m \times n}$, $m \geq n$

for now, is $A = U S V^T$

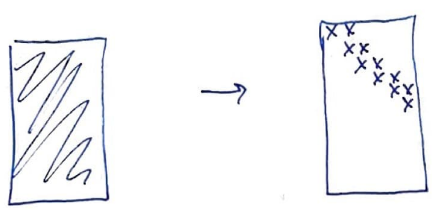
U orth.

S diag

V orth.

Instead of finding eigenvalues of $A^T A = V S^2 V^T$,

we can instead do the following:

$A =$  via Householder reflections

dense

bidiagonal

B_k

$$\Rightarrow \quad U^T A V = B_k$$

composition of <u>different</u> Householder reflections.

$\Leftrightarrow \quad A = U B V^T$

If the SVD of B is $B = U_B S V_B^T$, then

$$A = (U U_B) S (V V_B)^T$$

and A and B share the same singular values.

To compute the SVD of B, we form:

$$H = \begin{pmatrix} 0 & B^T \\ B & 0 \end{pmatrix} = \begin{pmatrix} 0 & \diagdown\!\diagdown \\ \diagdown\!\diagdown & 0 \end{pmatrix}$$

Note that H is symmetric, and is diagonalizable:

$$H \begin{pmatrix} V_B & V_B \\ U_B & -U_B \end{pmatrix} = \begin{pmatrix} V_B & V_B \\ U_B & -U_B \end{pmatrix} \begin{pmatrix} S & 0 \\ 0 & -S \end{pmatrix}$$ is the

eigendecomposition.

Then, permuting H gives a symmetric tridiagonal matrix:

$$P^T H P = \begin{pmatrix} 0 & x & & & \\ x & 0 & x & & \\ & x & \ddots & \ddots & \\ & & \ddots & \ddots & x \\ & & & x & 0 & x \\ & & & & x & 0 \end{pmatrix}$$ $\leftarrow$ Apply QR Algorithm to this, which yields its eigenvalues.

$$\Rightarrow \tilde{Q}^T P^T H P \tilde{Q} \approx \begin{pmatrix} S & 0 \\ 0 & -S \end{pmatrix}$$

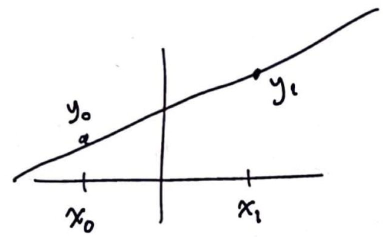
Then the SVD of B can be computed from $\tilde{Q}$, P, and S.

(exercise for homework.)

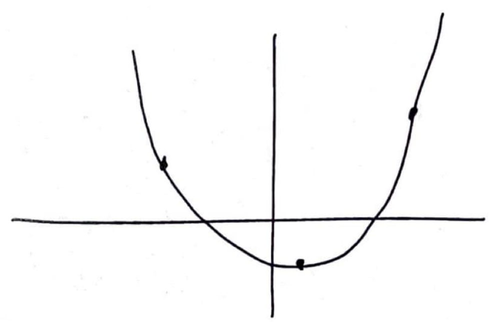
Polynomial Interpolation

Since computers can only multiply/add, basically the only functions that your computer can evaluate are polynomials.

Ex: • 2 points in the xy-plane define a line.



• 3 points define a parabola (a deg 2 polynomial)



In general, $n+1$ unique points in the xy-plane define a polynomial of degree n:

$$p(x) = \underbrace{a_n x^n + a_{n-1} x^{n-1} + \dots + a_1 x + a_0}$$

$$(*) \quad \left.\begin{aligned} p(x_0) &= y_0 \\ p(x_1) &= y_1 \\ &\vdots \\ p(x_n) &= y_n \end{aligned}\right\} \text{ $n+1$ equations for the $n+1$ unknowns a_j in}$$

The point of Interpolation

$\boxed{\text{Function Approximation}}$ Most functions (i.e., $\cos x$, solutions to differential equations, etc.) are not polynomials, and do not have closed form solutions. However, most of the time they can be locally approximated via polynomials (just think Taylor series).

$\Rightarrow$ Polynomial interpolation is at the core of Numerical Analysis.

With this in mind, given (x_j, y_j)'s, $j = 0, \dots, n$ with $x_j \in [a,b]$, how do we compute the interpolant:

Option 1 Solve for the coefficients in $p_n(x) = a_0 + a_1x + \dots + a_nx^n$.

(*) Can be written in matrix form as:

$$\underbrace{\begin{pmatrix} 1 & x_0 & x_0^2 & \cdots & x_0^n \\ 1 & x_1 & x_1^2 & \cdots & x_1^n \\ & & \vdots & & \\ 1 & x_n & x_n^2 & \cdots & x_n^n \end{pmatrix}}_{\text{Vandermonde Matrix } = A} \begin{pmatrix} a_0 \\ a_1 \\ \vdots \\ a_n \end{pmatrix} = \begin{pmatrix} y_0 \\ y_1 \\ \vdots \\ y_n \end{pmatrix}$$

Unless the x_j's are chosen very carefully, the relative condition number K grows exponentially.

So: If the x_j's are distinct, A is formally invertible, but horribly ill-conditioned. Never try to numerically invert it.

Option 2 The coefficients a_j usually don't matter – the goal is usually to evaluate p_n at some new point $x = x_j$.

While the polynomial p_n is unique, there are many ways to construct / evaluate it, the most common of which is the Lagrange Interpolating Polynomial.

Idea: Construct a sequence of polynomials $L_k \in P_n$ such that:

particular polynomial / subspace of polynomials of degree $\leq n$

$$L_k(x_j) = \begin{cases} 1 & \text{if } j = k \\ 0 & \text{if } j \neq k \end{cases}$$

If this is possible, then $p_n(x) = \sum_{k=0}^{n} y_k L_k(x)$ is the interpolating polynomial for the data (x_j, y_j), $j = 0 \ldots n$.

The construction of such polynomials L_k is straightforward:

$$L_k(x) = \prod_{\substack{j=0 \\ j\neq k}}^{n} \frac{x - x_j}{x_k - x_j} \qquad (*)$$

$$= \left(\prod_{\substack{j=0 \\ j\neq k}}^{n} \frac{1}{(x_k - x_j)} \right) \left(\prod_{\substack{j=0 \\ j\neq k}}^{n} (x - x_j) \right)$$

Theorem Given data (x_j, y_j) $j = 0 .. n$, there exists a unique polynomial $p_n \in P_n$ such that $p_n(x_j) = y_j$.

Proof Existence: Immediately follows from the Lagrange formula $(*)$

Uniqueness: See textbook.

The form of the interpolating polynomial $p_n(x) = \sum L_k(x) y_k$ is referred to as the "Lagrange interpolation formula of degree n".