MATH 543 – Homework 6

Parham Khodadi

TB-18.1

Problem

Consider the example $A = \begin{bmatrix} 1 & 1 \\ 1 & 1.0001 \\ 1 & 1.0001 \end{bmatrix}$, $\vec{b} = \begin{bmatrix} 2 \\ 0.0001 \\ 4.0001 \end{bmatrix}$.

A) What are the matrices A^+ and P for this example? Give exact answers.

B) Find the exact solutions x and $y = Ax$ to the least squares problem $Ax \approx b$.

C) What are $\kappa(A)$, θ, and η? From here on, numerical answers are acceptable.

D) What are the four conidition numbers of Theorem 18.1?

E) Give examples of perturbations δb and δA that approximately attain these four condition numbers.

Solution

MATLAB code found in Appendix.

A)

$$A^+ = (A^T A)^{-1} A^T = \begin{bmatrix} 10001 & -5000 & -5000 \\ -10000 & 5000 & 5000 \end{bmatrix}$$

$$P = AA^+ = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 0.5 & 0.5 \\ 0 & 0.5 & 0.5 \end{bmatrix}$$

B)

$$\vec{x} = A^+ \vec{b} = \begin{bmatrix} 1 \\ 1 \end{bmatrix}$$

$$\vec{y} = A\vec{x} = \begin{bmatrix} 2 \\ 2.0001 \\ 2.0001 \end{bmatrix}$$

C)

$\kappa(A) = \frac{\sigma_1}{\sigma_n} \approx 42429.235416083058$

$\theta = cos^{-1}(\frac{||\vec{y}||_2}{||\vec{b}||_2}) \approx 0.68470283674839516 rad \approx 39.2305827663180509\,\text{deg}$

$\eta = \frac{||A||_2||\vec{x}||_2}{||\vec{y}||_2} \approx 1.0000000005535632$

D)

$\kappa(b \to y) = 1.2909771975928801$

$\kappa(b \to x) = 54775.175403141970$

$\kappa(A \to y) = 54775.175433463490$

$\kappa(A \to x) = 1469883143.3101680$

E)

I wasn't sure how to do this part, so I asked Gemini for help. Check the Appendix for more info.

To approximately attain these four condition numbers, we construct specific worst-case perturbations using the Singular Value Decomposition $A = U\Sigma V^T$ and the residual $\vec{r} = \vec{b} - \vec{y}$. Let $\epsilon > 0$ be a small scaling scalar (e.g., 10^{-8}).

- $\kappa(b \to y)$**:** The perturbation δb must lie entirely in the column space of A. We choose $\delta b = \epsilon \frac{\vec{y}}{||\vec{y}||_2} ||\vec{b}||_2$.
- $\kappa(b \to x)$**:** The perturbation δb must align with the left singular vector corresponding to the smallest singular value. We choose $\delta b = \epsilon \vec{u}_n ||\vec{b}||_2$.
- $\kappa(A \to y)$**:** The perturbation δA must map the most sensitive right singular vector ($\vec{v}_n$) directly into the residual space. We choose $\delta A = \epsilon \left(\frac{\vec{r}}{||\vec{r}||_2}\right) \vec{v}_n^T ||A||_2$.
- $\kappa(A \to x)$**:** The perturbation δA must map $\vec{v}_n$ into a specific linear combination of $\vec{u}_n$ and $\vec{r}$. We choose $\delta A = \epsilon \left(c_1 \vec{u}_n + c_2 \frac{\vec{r}}{||\vec{r}||_2}\right) \vec{v}_n^T ||A||_2$, where c_1 and c_2 are optimally weighted based on the pseudoinverse and residual norms.

From MATLAB: $\delta b_y = \begin{bmatrix} 2.5820e-08 \\ 2.5821e-08 \\ 2.5821e-08 \end{bmatrix}$

$\delta b_x = \begin{bmatrix} 3.6516e-08 \\ -1.8257e-08 \\ -1.8257e-08 \end{bmatrix}$

$\delta A_y = \begin{bmatrix} -3.6512e-16 & 3.6509e-16 \\ -1.2248e-08 & 1.2247e-08 \\ 1.2248e-08 & -1.2247e-08 \end{bmatrix}$

$\delta A_x = \begin{bmatrix} -3.7870e-16 & 3.7868e-16 \\ -1.2248e-08 & 1.2247e-08 \\ 1.2248e-08 & -1.2247e-08 \end{bmatrix}$

PB-14.1

Problem

Consider the vector $\vec{x} \in \mathbb{R}^{101}$ consisting of equi-spaced points in the interval $[0, 1]$, e.g. $x = linspace(0, 1, 101)'$; and let $A_k \in \mathbb{R}^{101\times(k+1)}$ be the matrix consisting of columns formed by component-wise powers $0, ..., k$ of the x-values (a Vandermonde Matrix). Let $c_l = k(A_l)$ be components of the vector $\vec{c}$ containing the collection of condition numbers for these matrices. Let $l \in 0, ..., L$, and make L large enough that you see something interesting.

- Plot $\vec{c}$ (use a log scale)
- We could use these matrices A_k to least-squares-fit polynomials (of matching degree k) to some data-set with 101 measurements. Is it necessarily better to have more model parameters (i.e. fitting a higher degree polynomial)? — Discuss.

Solution

Plot

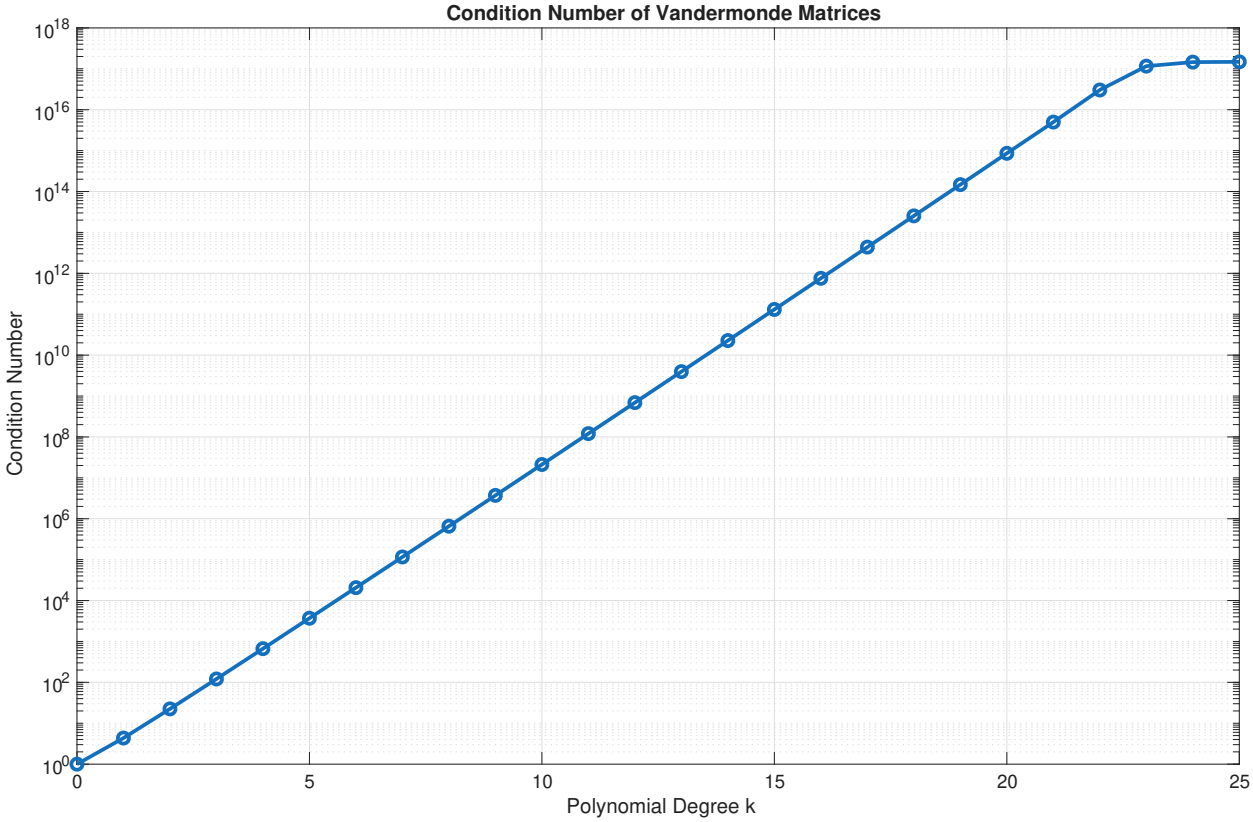


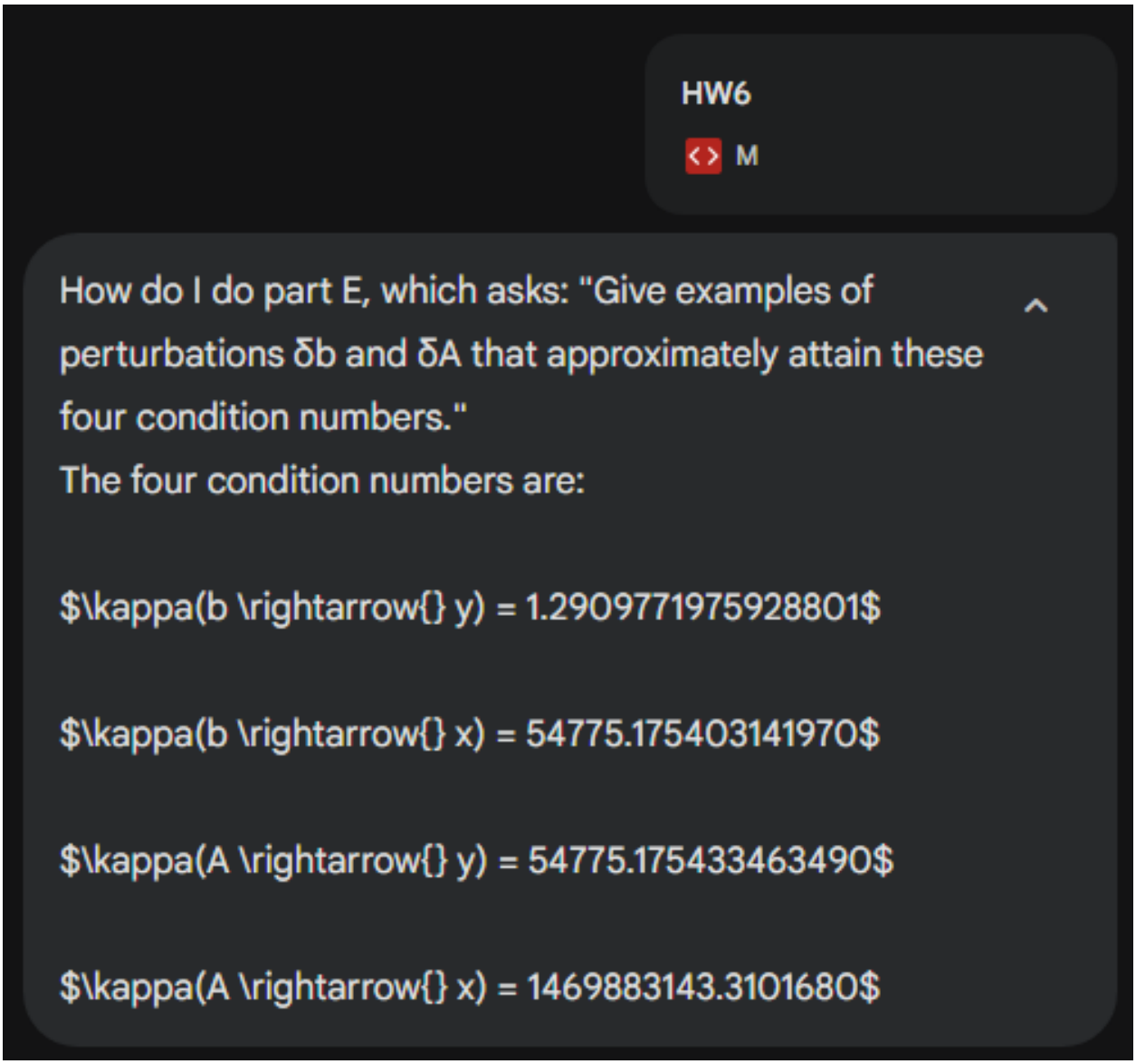
Figure 1: Condition number of Vandermonde matrices as a function of polynomial degree k.

Discussion

No, it is not better to use more model parameters (a higher degree polynomial) to fit your data. As you increase the polynomial degree k, the condition number $\kappa(A_k)$ of the Vandermonde matrix

grows exponentially. Conditioning is the measure of sensitivity of solutions to perturbations , such as uncertainties or floating-point representation errors in the data. The condition number for finding the least squares solution vector $\vec{x}$ contains the square of the condition number of the matrix A, given by the formula $\kappa(A \mapsto \vec{x}) = \kappa(A) + \tan(\theta)\kappa(A)^2\eta$.

Appendix: Gemini AI



Appendix: MATLAB Code

```
%% HW6 - TB-18.1 and PB-14.1
% By Parham Khodadi
clear; clc; close all;

%% ============================
%  (1) TB-18.1
%  ============================
fprintf('=================================================================\n');
fprintf('(1) TB-18.1\n');
fprintf('=================================================================\n');

% Problem data
A = [1, 1; ...
     1, 1.0001; ...
     1, 1.0001];

b = [2; 0.0001; 4.0001];

% A^+ = (A^T A)^{-1} A^T
Api_exact = (A' * A)^(-1) * A';

% P = A A^+
P_exact = A*Api_exact;

% x = A^+ b
x_exact = Api_exact * b;

% y = A x
y_exact = A*x_exact;

fprintf('\nPart A:\n');
fprintf('A^+ (exact) =\n');
disp(Api_exact);
fprintf('P = A*A^+ (exact) =\n');
disp(P_exact);

fprintf('\nPart B:\n');
fprintf('x (exact) = \n');
disp(x_exact);
fprintf('y (exact) = \n');
disp(y_exact);

%% Part C/D
[U,S,V] = svd(A);
sigma = diag(S);
sigma1 = sigma(1);
sigman = sigma(2);

kappaA = sigma1 / sigman;
cos_theta = norm(y_exact,2) / norm(b,2);
theta = acos(cos_theta);
```

```
eta = norm(A,2) * norm(x_exact,2) / norm(y_exact,2);

fprintf('\nPart C:\n');
fprintf('sigma_1                =  %.16e\n', sigma1);
fprintf('sigma_n                =  %.16e\n', sigman);
fprintf('kappa(A)               =  %.16e\n', kappaA);
fprintf('cos(theta)             =  %.16e\n', cos_theta);
fprintf('theta (radians)        =  %.16e\n', theta);
fprintf('theta (degrees)        =  %.16f\n', theta*180/pi);
fprintf('eta                    =  %.16e\n', eta);

k_b_to_y = 1 / cos_theta;
k_b_to_x = kappaA / (eta * cos_theta);
k_A_to_y = kappaA / cos_theta;
k_A_to_x = kappaA + (kappaA^2) * tan(theta) / eta;

fprintf('\nPart D:\n');
fprintf('kappa(b -> y)          =  %.16e\n', k_b_to_y);
fprintf('kappa(b -> x)          =  %.16e\n', k_b_to_x);
fprintf('kappa(A -> y)          =  %.16e\n', k_A_to_y);
fprintf('kappa(A -> x)          =  %.16e\n', k_A_to_x);

%% Part E

fprintf('\nPart E:\n')

% Set a small relative perturbation size
epsilon = 1e-8;

% --- 1. b -> y ---
% Worst-case db must be entirely in the column space of A.
db_y = (y_exact / norm(y_exact, 2)) * epsilon * norm(b, 2);

% --- 2. b -> x ---
% Worst-case db aligns with the left singular vector for the smallest
   singular value.
u_n = U(:, 2);
db_x = u_n * epsilon * norm(b, 2);

% --- 3. A -> y ---
% Worst-case dA maps the most sensitive right singular vector (v_n) to the
    residual space.
r = b - y_exact;
u_r = r / norm(r, 2);
v_n = V(:, 2);
dA_y = u_r * v_n' * epsilon * norm(A, 2);

% --- 4. A -> x ---
% Worst-case dA maps v_n to a specific linear combination of u_n and u_r.
w1 = -(v_n' * x_exact) / sigman;
w2 = norm(r, 2) / (sigman^2);
norm_w = sqrt(w1^2 + w2^2);

c1 = w1 / norm_w;
```

```
c2 = w2 / norm_w;

u_opt = c1 * u_n + c2 * u_r;
dA_x = u_opt * v_n' * epsilon * norm(A, 2);

fprintf('\\delta b_y = \\begin{bmatrix} %.4e \\\\ %.4e \\\\ %.4e \\end{
    bmatrix}\n\n', db_y(1), db_y(2), db_y(3));
fprintf('\\delta b_x = \\begin{bmatrix} %.4e \\\\ %.4e \\\\ %.4e \\end{
    bmatrix}\n\n', db_x(1), db_x(2), db_x(3));
fprintf('\\delta A_y = \\begin{bmatrix} %.4e & %.4e \\\\ %.4e & %.4e \\\\
    %.4e & %.4e \\end{bmatrix}\n\n', dA_y(1,1), dA_y(1,2), dA_y(2,1), dA_y
    (2,2), dA_y(3,1), dA_y(3,2));
fprintf('\\delta A_x = \\begin{bmatrix} %.4e & %.4e \\\\ %.4e & %.4e \\\\
    %.4e & %.4e \\end{bmatrix}\n', dA_x(1,1), dA_x(1,2), dA_x(2,1), dA_x
    (2,2), dA_x(3,1), dA_x(3,2));

%% ==========================
%  (2) PB-14.1
%  ==========================
fprintf('\n================================================================\n'
    );
fprintf('(2) PB-14.1\n');
fprintf('================================================================\n');

x = linspace(0,1,101)';
L = 25;
c = zeros(L+1,1);

for k = 0:L
    A_k = x.^(0:k);
    c(k+1)=cond(A_k);
end

semilogy(0:L, c, '-o', 'LineWidth', 2)
xlabel('Polynomial Degree k')
ylabel('Condition Number')
title('Condition Number of Vandermonde Matrices')
grid on
exportgraphics(gcf, 'Figures/PB_14_1.eps', 'ContentType', 'vector')
```