Homework 7
CS 259 Numerical Methods for Data Science
Prof. **David Bindel**
TA. **Yurong You, Xinran Zhu**

Hongyu Yan (516030910595)
ACM Class, Zhiyuan College, SJTU
Due Date: July 2, 2018
Submit Date: July 26, 2018

Problem 1

$X = [x_1, x_2, \cdots, x_n]$. In order to minimize $\|Ac - f_X\|^2$, we have $c = A^{\dagger} f_X$. In this problem, we have $\mathcal{A}[f_X] = T(x)^T A^{\dagger} f_X$, where $T(x) = \begin{bmatrix} T_0(x) \\ T_1(x) \\ \vdots \\ T_{n-1}(x) \end{bmatrix}$, so:

$$\begin{aligned} &\|\mathcal{A}[f_X + u] - \mathcal{A}[f_X]\|_\infty \\ =&\|T(x)^T A^{\dagger}(f_X + u - f_X)\|_\infty \\ \leq&\|T(x)^T\|_\infty \|A^{\dagger}\|_\infty \|u\|_\infty \end{aligned}$$

When $x \in [-1, 1]$, $T(x) \leq 1$, then $\|T(x)^T\|_\infty \leq n$. So $\|\mathcal{A}[f_X + u] - \mathcal{A}[f_X]\|_\infty \leq n\|A^{\dagger}\|_\infty \|u\|_\infty$, which means $L \leq n\|A^{\dagger}\|_\infty$

Here is my MATLAB code for ploting relationship between $n\|A^{\dagger}\|_\infty$ versus n.

```
X = zeros(3,1);
Y = zeros(3,1);
m = 7;
for k = 1:3
  n = k*m;
  x = linspace(-1, 1, n);
  A = chebmatrix(x, n);
  X(k) = n;
  Y(k) = n * norm(pinv(A), inf);
end
semilogy(X, Y, '--o');
xlabel('n');
ylabel('n||A||_{\infty}');
```

The plot is as below, which is approximately a straight line:

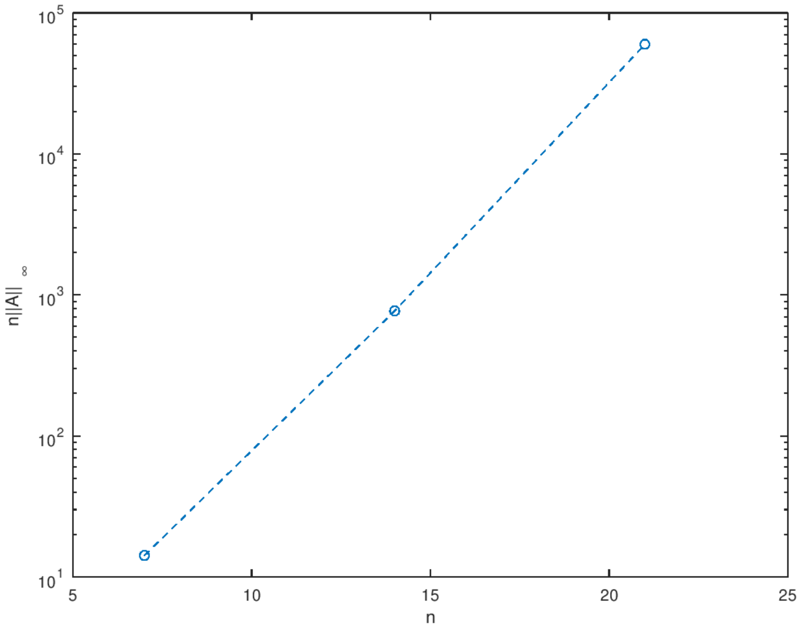

Figure 1: $n\|A^{\dagger}\|_{\infty}$ versus n(m=7)

Problem 2

My code is as below:

```
x = linspace(-1, 1, 10);
fx = sin(2 * pi * x);

% generate Kxx, P
Kxx = zeros(10, 10);
for i = 1:10
  for j = 1:10
    Kxx(i,j) = abs((x(i)-x(j))^3);
  end
end
P = [ones(10, 1) x'];

A = [Kxx, P; P' zeros(2, 2)];
b = [fx'; zeros(2, 1)];
tmp = A\b;

d1 = tmp(11);
d2 = tmp(12);
% f(x) = d1 + d2*x + sum(c_i*|x_i-x_j|^3)

x_fit = linspace(-1, 1, 100);
y = f(tmp, x_fit);
y_true = sin(2 * pi * x_fit);
```

```

figure(1);
plot(x_fit, y, x_fit, y_true);
xlabel('x');
ylabel('y');
legend('interpolation␣curve', 'ground␣truth');

figure(2);
plot(x_fit, (y - y_true).^2);
xlabel('x');
ylabel('approximation␣error');
```

```
function [y] = f(tmp, X)
  x = linspace(-1, 1, 10);
  y = tmp(11) * ones(1, length(X)) + tmp(12) * X;
  for i = 1:10
    y = y + tmp(i) * (abs(X - x(i)).^3);
  end
```

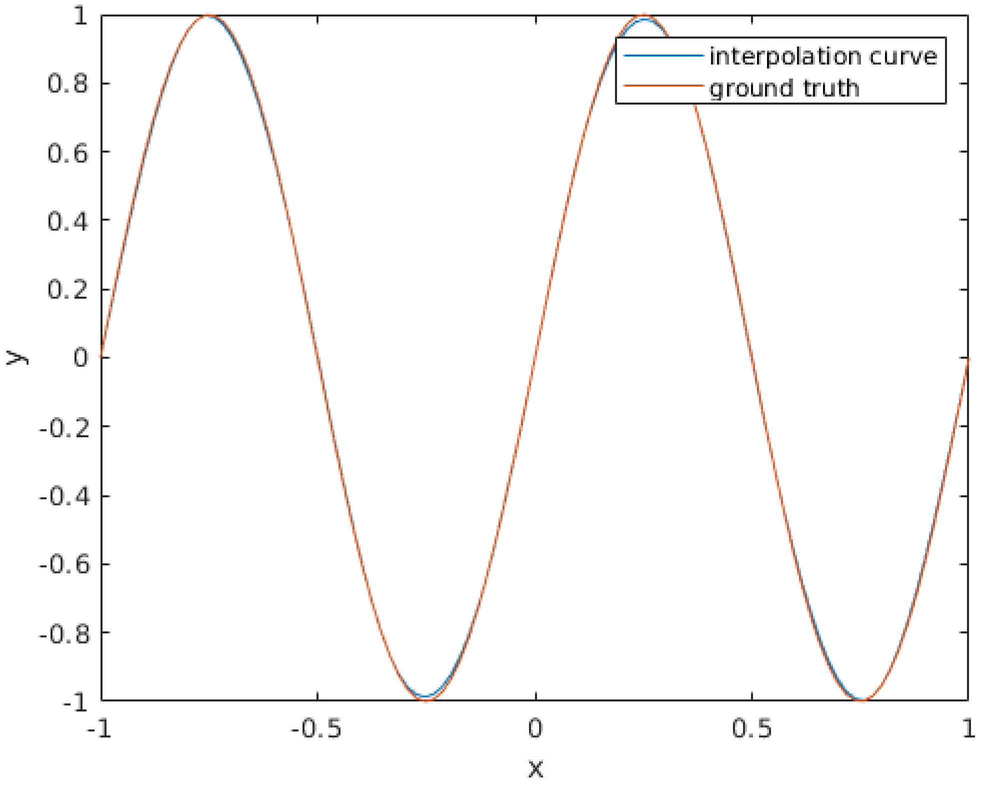

Figure 2: spline interpolation

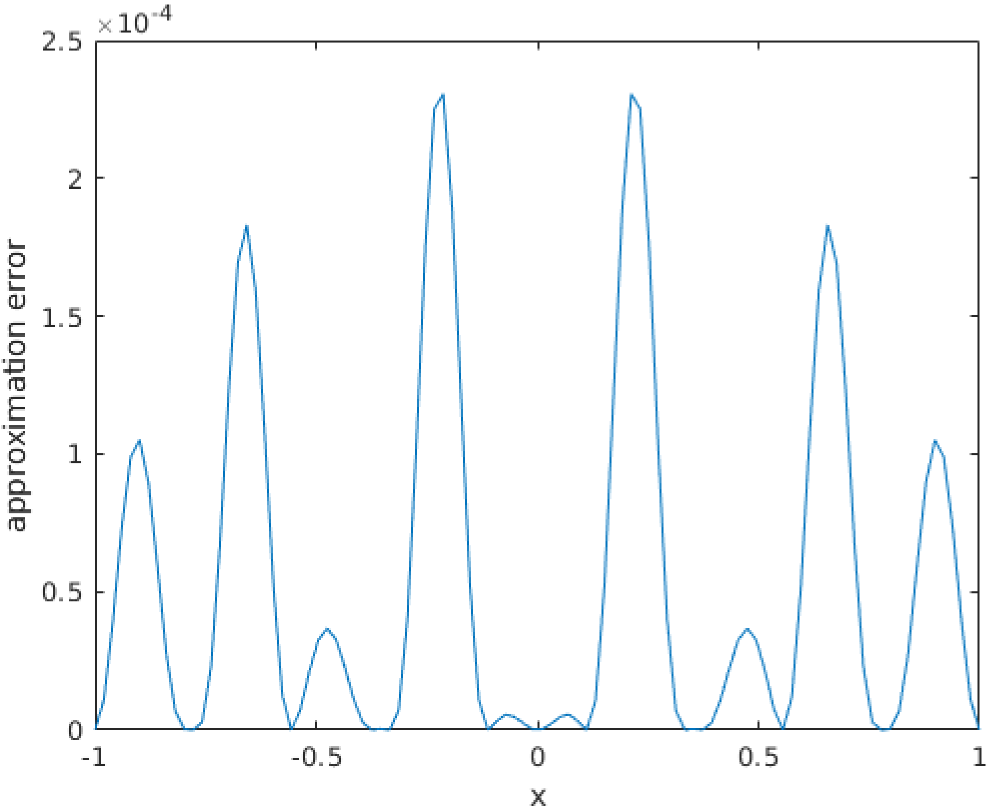

Figure 3: approximation error