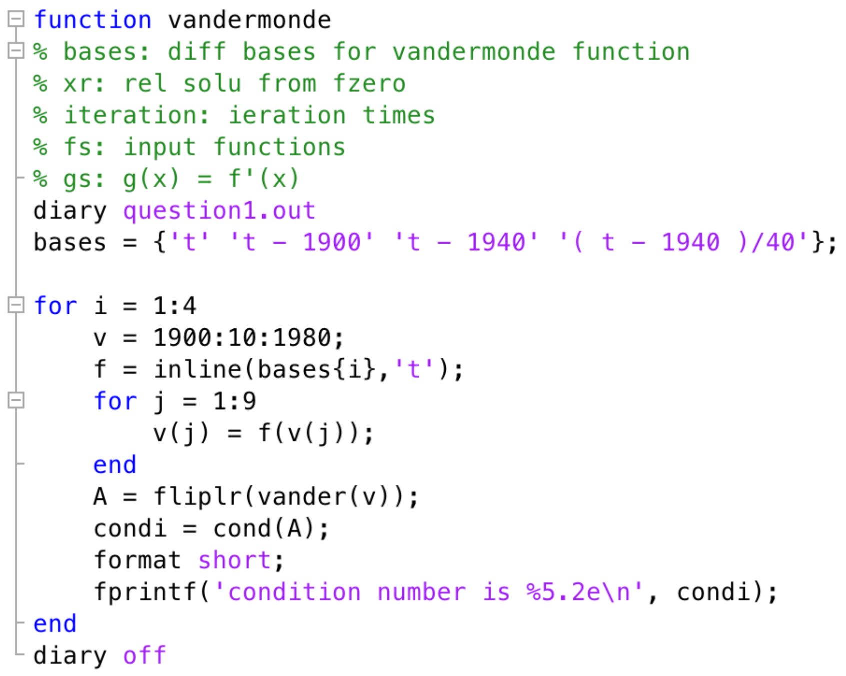
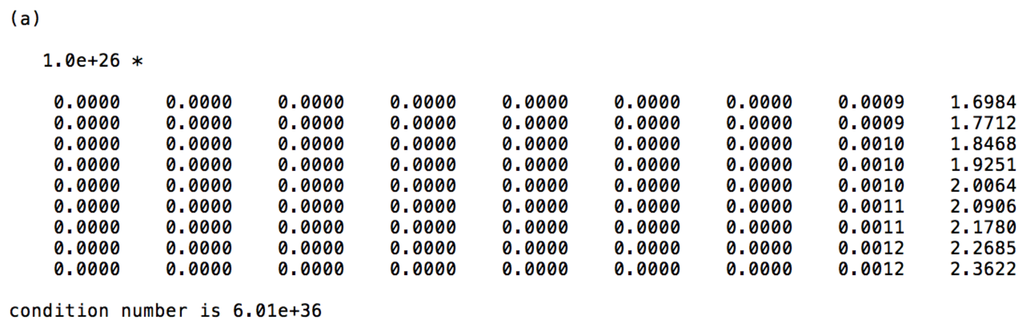
Rui Ji (1000340918)

1. solution:



```
function vandermonde
% bases: diff bases for vandermonde function
% xr: rel solu from fzero
% iteration: ieration times
% fs: input functions
% gs: g(x) = f'(x)
diary question1.out
bases = {'t' 't - 1900' 't - 1940' '( t - 1940 )/40'};

for i = 1:4
    v = 1900:10:1980;
    f = inline(bases{i},'t');
    for j = 1:9
        v(j) = f(v(j));
    end
    A = fliplr(vander(v));
    condi = cond(A);
    format short;
    fprintf('condition number is %5.2e\n', condi);
end
diary off
```

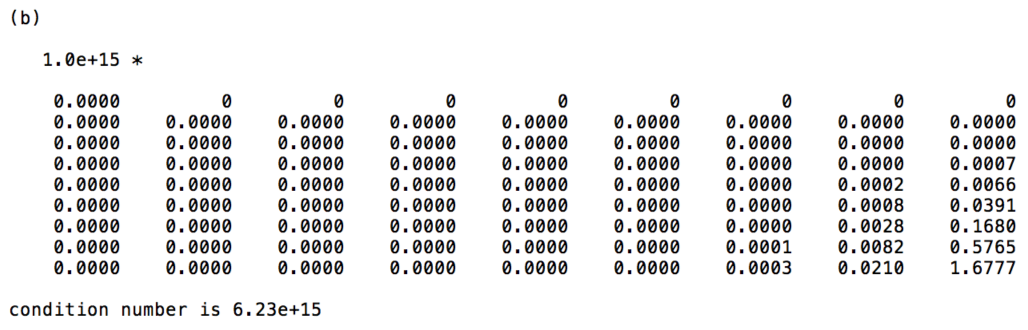


```
(a)

   1.0e+26 *

    0.0000    0.0000    0.0000    0.0000    0.0000    0.0000    0.0000    0.0009    1.6984
    0.0000    0.0000    0.0000    0.0000    0.0000    0.0000    0.0000    0.0009    1.7712
    0.0000    0.0000    0.0000    0.0000    0.0000    0.0000    0.0000    0.0010    1.8468
    0.0000    0.0000    0.0000    0.0000    0.0000    0.0000    0.0000    0.0010    1.9251
    0.0000    0.0000    0.0000    0.0000    0.0000    0.0000    0.0000    0.0010    2.0064
    0.0000    0.0000    0.0000    0.0000    0.0000    0.0000    0.0000    0.0011    2.0906
    0.0000    0.0000    0.0000    0.0000    0.0000    0.0000    0.0000    0.0011    2.1780
    0.0000    0.0000    0.0000    0.0000    0.0000    0.0000    0.0000    0.0012    2.2685
    0.0000    0.0000    0.0000    0.0000    0.0000    0.0000    0.0000    0.0012    2.3622

condition number is 6.01e+36
```

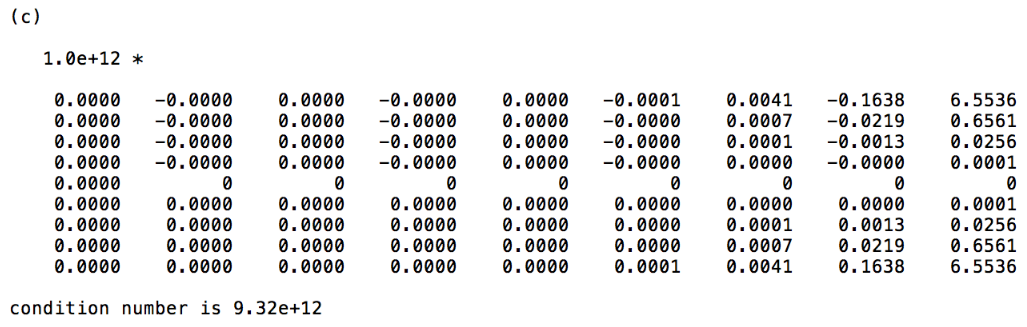


```
(b)

   1.0e+15 *

    0.0000         0         0         0         0         0         0         0         0
    0.0000    0.0000    0.0000    0.0000    0.0000    0.0000    0.0000    0.0000    0.0000
    0.0000    0.0000    0.0000    0.0000    0.0000    0.0000    0.0000    0.0000    0.0000
    0.0000    0.0000    0.0000    0.0000    0.0000    0.0000    0.0000    0.0000    0.0007
    0.0000    0.0000    0.0000    0.0000    0.0000    0.0000    0.0000    0.0002    0.0066
    0.0000    0.0000    0.0000    0.0000    0.0000    0.0000    0.0000    0.0008    0.0391
    0.0000    0.0000    0.0000    0.0000    0.0000    0.0000    0.0000    0.0028    0.1680
    0.0000    0.0000    0.0000    0.0000    0.0000    0.0000    0.0001    0.0082    0.5765
    0.0000    0.0000    0.0000    0.0000    0.0000    0.0000    0.0003    0.0210    1.6777

condition number is 6.23e+15
```

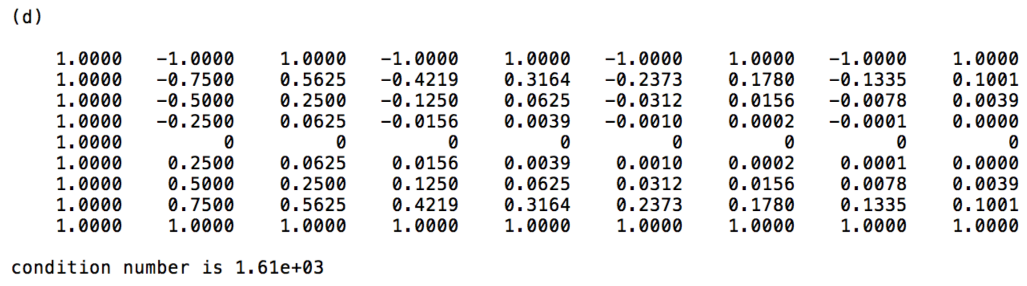


```
(c)

  1.0e+12 *

    0.0000   -0.0000    0.0000   -0.0000    0.0000   -0.0001    0.0041   -0.1638    6.5536
    0.0000   -0.0000    0.0000   -0.0000    0.0000   -0.0000    0.0007   -0.0219    0.6561
    0.0000   -0.0000    0.0000   -0.0000    0.0000   -0.0000    0.0001   -0.0013    0.0256
    0.0000   -0.0000    0.0000   -0.0000    0.0000   -0.0000    0.0000   -0.0000    0.0001
    0.0000         0         0         0         0         0         0         0         0
    0.0000    0.0000    0.0000    0.0000    0.0000    0.0000    0.0000    0.0000    0.0001
    0.0000    0.0000    0.0000    0.0000    0.0000    0.0000    0.0001    0.0013    0.0256
    0.0000    0.0000    0.0000    0.0000    0.0000    0.0000    0.0007    0.0219    0.6561
    0.0000    0.0000    0.0000    0.0000    0.0000    0.0001    0.0041    0.1638    6.5536

condition number is 9.32e+12
```



```
(d)

    1.0000   -1.0000    1.0000   -1.0000    1.0000   -1.0000    1.0000   -1.0000    1.0000
    1.0000   -0.7500    0.5625   -0.4219    0.3164   -0.2373    0.1780   -0.1335    0.1001
    1.0000   -0.5000    0.2500   -0.1250    0.0625   -0.0312    0.0156   -0.0078    0.0039
    1.0000   -0.2500    0.0625   -0.0156    0.0039   -0.0010    0.0002   -0.0001    0.0000
    1.0000         0         0         0         0         0         0         0         0
    1.0000    0.2500    0.0625    0.0156    0.0039    0.0010    0.0002    0.0001    0.0000
    1.0000    0.5000    0.2500    0.1250    0.0625    0.0312    0.0156    0.0078    0.0039
    1.0000    0.7500    0.5625    0.4219    0.3164    0.2373    0.1780    0.1335    0.1001
    1.0000    1.0000    1.0000    1.0000    1.0000    1.0000    1.0000    1.0000    1.0000

condition number is 1.61e+03
```

- First we could notice that the condition numbers for (a), (b), (c) are large and they are ill-condition. And the condition numbers are decreasing from (a) to (d); hence, clearly we should recommend (d).
- Also, The functions are exponential functions with exponent bigger than 1. Then they all grow rapidly for base number with absolute value bigger than 1.
- for (a) the base numbers are all bigger than 1900, which makes the function grows even faster. Therefore, comparing with last few columns, rest columns are too small, which make the Vandermonde matrix nearly singular. Same for (b) base numbers are still big.
- For (c), $t-1940$ makes the base number symmetric about 1940, but since the absolute value for most the of base numbers are still bigger than 10, then it still grows fast as the exponent increases, which makes the Vandermonde matrix nearly singular.
- For (d), $\frac{(t-1940)}{40}$ makes all base numbers between $[-1,1]$, then the value are changing relative slow compare with (a), (b) and (c) which have bases number bigger than 10.

For all those reasons have been discussed above, I would recommend the function d to generate an interpolant if the Vandermonde approach is used.

2. SOLUTION:

   (a) Using **matlab**'s **polyfit**($[-1,2,3,5]$, $[0,1,1,2]$, 3), we get $[\frac{1}{24}, \frac{-1}{4}, \frac{11}{24}, \frac{3}{4}]$.
   Hence, the function we get from polynomial interpolant using monomial basis is,

$$\frac{1}{24}x^3 - \frac{1}{4}x^2 + \frac{11}{24}x + \frac{3}{4}$$

(b) Using **Lagrange** polynomial interpolant we get,

$$P_3(x) = 0 + \frac{(t+1)(t-3)(t-5)}{(2+1)(2-3)(2-5)} + \frac{(t+1)(t-2)(t-5)}{(3+1)(3-2)(3-5)} + \frac{2(t+1)(t-2)(t-3)}{(5+1)(5-2)(5-3)}$$
$$P_3(x) = 0 + \frac{t^3-7t^2+7t+15}{9} + \frac{t^3-6t^2+3t+10}{-8} + \frac{2t^3-8t^2+2t+12}{36}$$
$$P_3(x) = \frac{8t^3-56t^2+56t+120}{72} + \frac{9t^3-54t^2+27t+90}{-72} + \frac{4t^3-16t^2+4t+24}{72}$$
$$P_3(x) = \frac{3t^3-18t^2+33t+54}{72}$$
$$P_3(x) = \frac{1}{24}x^3 - \frac{1}{4}x^2 + \frac{11}{24}x + \frac{3}{4}$$

Clearly, using **Lagrange** polynomial interpolant we get the same polynomial with (a).

(c) Using **Newton** polynomial interpolant (Divided Differences) we get,

$$\begin{array}{llccc}
x_0 = -1 & y_0 = 0 & & & \\
 & & \frac{1}{3} & & \\
x_1 = 2 & y_1 = 1 & & \frac{-1}{12} & \\
 & & \frac{0}{1} & & \frac{1}{24} \\
x_2 = 3 & y_2 = 1 & & \frac{1}{6} & \\
 & & \frac{1}{2} & & \\
x_3 = 5 & y_3 = 2 & & &
\end{array}$$

Hence we have,

$$P_3(x) = \frac{1}{3}(x+1) - \frac{1}{12}(x+1)(x-2) + \frac{1}{24}(x+1)(x-2)(x-3)$$
$$P_3(x) = \frac{1}{3}(x+1) - \frac{1}{12}(x^2-x-2) + \frac{1}{24}x^3 - 4x^2 + x + 6$$
$$P_3(x) = \frac{1}{24}x^3 - \frac{1}{4}x^2 + \frac{11}{24}x + \frac{3}{4}$$

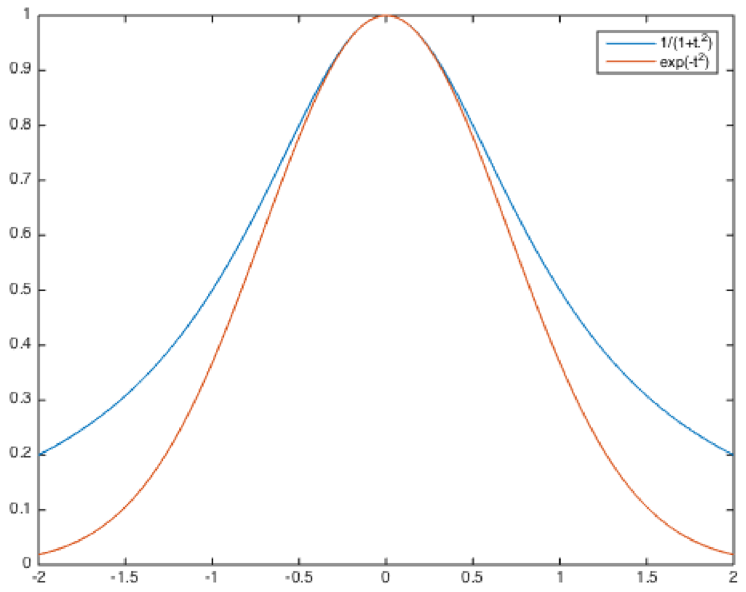
Clearly, using **Lagrange** polynomial interpolant we get the same polynomial with (a).

3. SOLUTION:

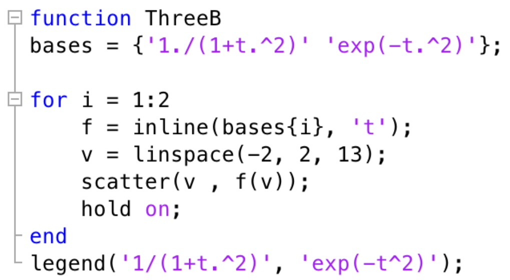
(a)

```
function ThreeA
bases = {'1./(1+t.^2)' 'exp(-t.^2)'};

for i = 1:2
    f = inline(bases{i}, 't');
    v = linspace(-2, 2, 201);
    plot(v , f(v));
    hold on;
end
legend('1/(1+t.^2)', 'exp(-t^2)');
```

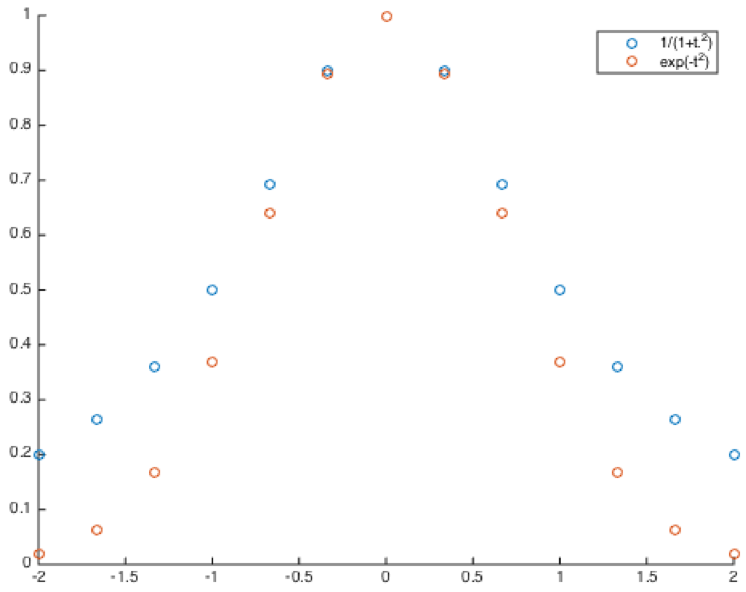


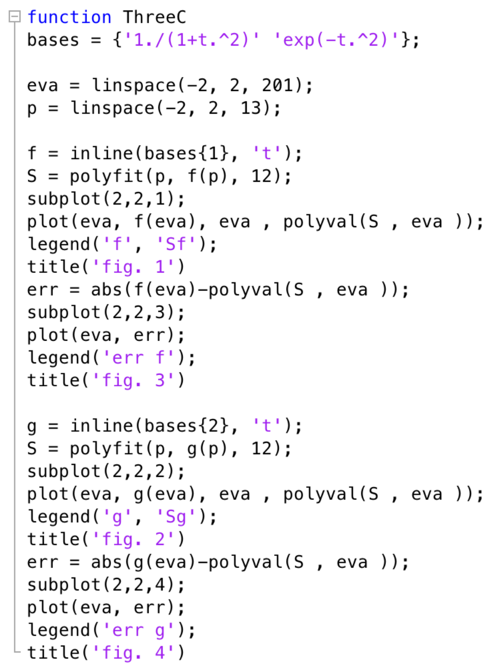
(b)



```
function ThreeB
bases = {'1./(1+t.^2)' 'exp(-t.^2)'};

for i = 1:2
    f = inline(bases{i}, 't');
    v = linspace(-2, 2, 13);
    scatter(v , f(v));
    hold on;
end
legend('1/(1+t.^2)', 'exp(-t^2)');
```





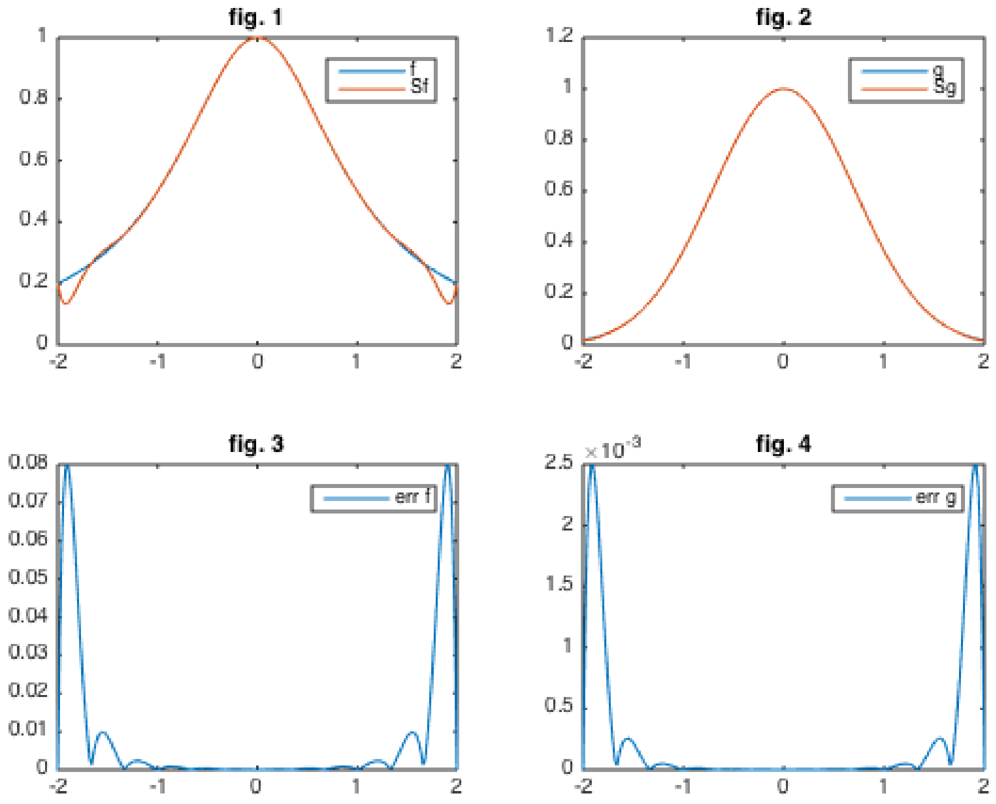
```
function ThreeC
bases = {'1./(1+t.^2)' 'exp(-t.^2)'};

eva = linspace(-2, 2, 201);
p = linspace(-2, 2, 13);

f = inline(bases{1}, 't');
S = polyfit(p, f(p), 12);
subplot(2,2,1);
plot(eva, f(eva), eva , polyval(S , eva ));
legend('f', 'Sf');
title('fig. 1')
err = abs(f(eva)-polyval(S , eva ));
subplot(2,2,3);
plot(eva, err);
legend('err f');
title('fig. 3')

g = inline(bases{2}, 't');
S = polyfit(p, g(p), 12);
subplot(2,2,2);
plot(eva, g(eva), eva , polyval(S , eva ));
legend('g', 'Sg');
title('fig. 2')
err = abs(g(eva)-polyval(S , eva ));
subplot(2,2,4);
plot(eva, err);
legend('err g');
title('fig. 4')
```

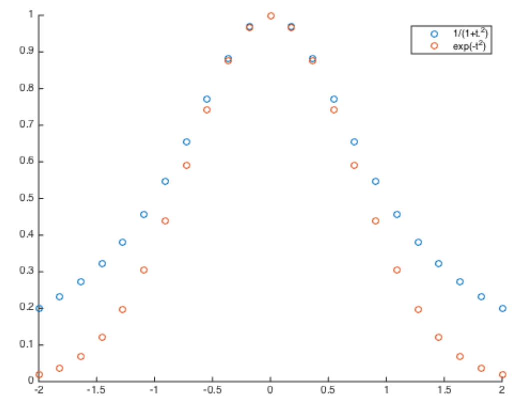
(c)

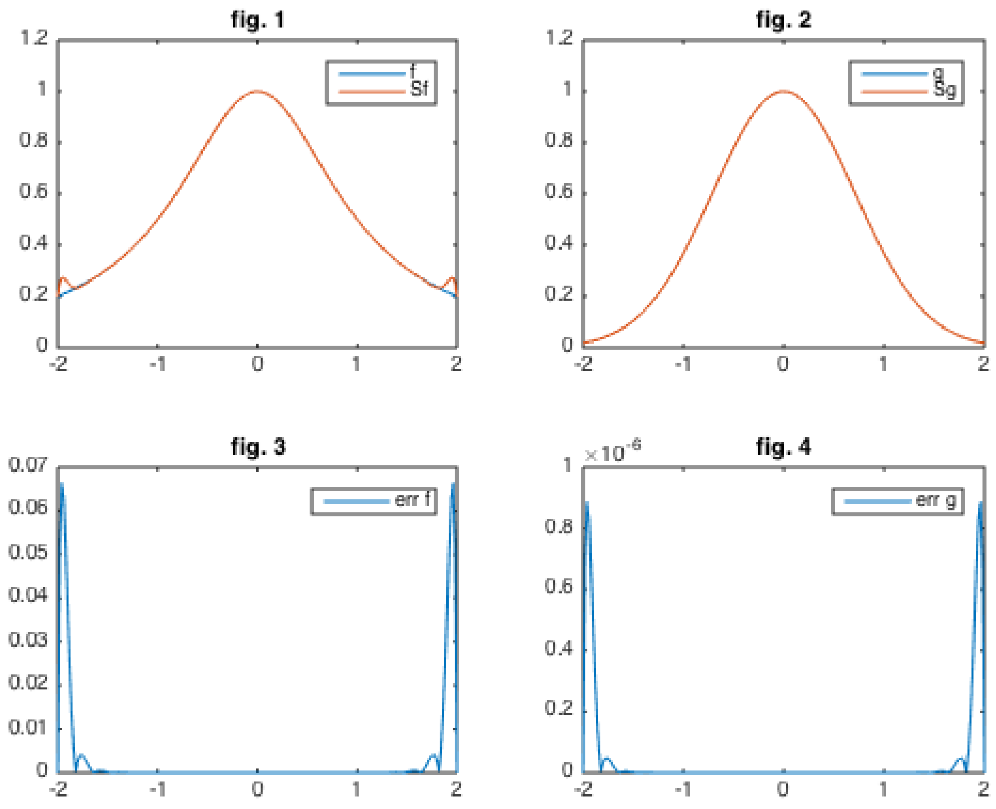


(d) It is clear that $g(t) = \exp^{(}-t^2)$ has a better approximation compare with $f(t) = \frac{1}{1+t^2}$. And we could see, for $f(t) = \frac{1}{1+t^2}$ when t is close to end points $2, -2$, the approximation become worse.

This is because the lack of the uniformly convergence of the polynomial interpolants to an underlying continuous function as the number of equally spaced points increases.

(e)





4. SOLUTION:

We know that for piecewise polynomial interpolation, the absolute error $e \leq \frac{max(\ f''(x)\ )\cdot h^2}{8}$, where $x \in [0, \pi/2]$ and h is the length of the longest subinterval in the partition.

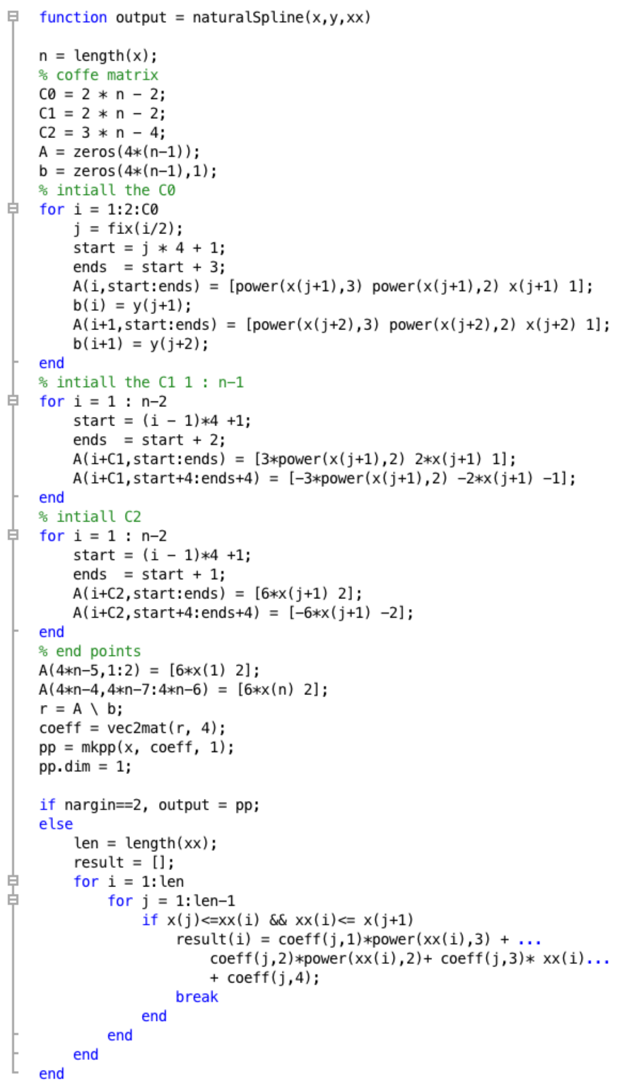
$$max(|f''(x)|) = max(|\sin''(x)|) = max(|sin(x)|) = 1$$

Since, the question asked for a absolute error within 1.0×10^{-4}, then we have,

$$\begin{aligned}
\frac{max(\ f''(x)\ )\ \cdot\ h^2}{8} &\leq 1.0 \times 10^{-4} \\
\frac{1\ \cdot\ h^2}{8} &\leq 1.0 \times 10^{-4} \\
h^2 &\leq 8.0 \times 10^{-4} \\
(\frac{\pi}{2(n-1)})^2 &\leq 8.0 \times 10^{-4} \quad n\ is\ the\ total\ number\ of\ enties \\
\frac{\pi}{2(n-1)} &\leq \sqrt{8.0 \times 10^{-4}} \\
2(n-1) &\geq \frac{\pi}{\sqrt{8.0 \times 10^{-4}}} \\
n &\geq \frac{\pi}{2\sqrt{8.0 \times 10^{-4}}} + 1 \\
n &\geq 56.5360
\end{aligned}$$

Hence, 57 entries are needed in order to make the absolute error within 1.0×10^{-4}.

5. (a)



```
function output = naturalSpline(x,y,xx)

n = length(x);
% coffe matrix
C0 = 2 * n - 2;
C1 = 2 * n - 2;
C2 = 3 * n - 4;
A = zeros(4*(n-1));
b = zeros(4*(n-1),1);
% intiall the C0
for i = 1:2:C0
    j = fix(i/2);
    start = j * 4 + 1;
    ends  = start + 3;
    A(i,start:ends) = [power(x(j+1),3) power(x(j+1),2) x(j+1) 1];
    b(i) = y(j+1);
    A(i+1,start:ends) = [power(x(j+2),3) power(x(j+2),2) x(j+2) 1];
    b(i+1) = y(j+2);
end
% intiall the C1 1 : n-1
for i = 1 : n-2
    start = (i - 1)*4 +1;
    ends  = start + 2;
    A(i+C1,start:ends) = [3*power(x(j+1),2) 2*x(j+1) 1];
    A(i+C1,start+4:ends+4) = [-3*power(x(j+1),2) -2*x(j+1) -1];
end
% intiall C2
for i = 1 : n-2
    start = (i - 1)*4 +1;
    ends  = start + 1;
    A(i+C2,start:ends) = [6*x(j+1) 2];
    A(i+C2,start+4:ends+4) = [-6*x(j+1) -2];
end
% end points
A(4*n-5,1:2) = [6*x(1) 2];
A(4*n-4,4*n-7:4*n-6) = [6*x(n) 2];
r = A \ b;
coeff = vec2mat(r, 4);
pp = mkpp(x, coeff, 1);
pp.dim = 1;

if nargin==2, output = pp;
else
    len = length(xx);
    result = [];
    for i = 1:len
        for j = 1:len-1
            if x(j)<=xx(i) && xx(i)<= x(j+1)
                result(i) = coeff(j,1)*power(xx(i),3) + ...
                    coeff(j,2)*power(xx(i),2)+ coeff(j,3)* xx(i)...
                    + coeff(j,4);
                break
            end
        end
    end
end
```

We use $[(00),(1,1),(2,8)]$ test we suppose to get,

$$p_1(t) = a_1x^3 + a_2x^2 + a_3x + a_4$$

$$p_2(t) = b_1x^3 + b_2x^2 + b_3x + b_4$$

$$a_4 = 0$$

$$a_1 + a_2 + a_3 + a_4 = 1$$

$$b_1 + b_2 + b_3 + b_4 = 1$$

$$8b_1 + 4b_2 + 4b_3 + b_4 = 8$$

$$3a_1 + 2a_2 + a_3 = 3b_1 + 2b_2 + b_3$$

$$6a_1 + 2a_3 = 6b_1 + 2b_2$$

$$2a_3 = 0$$

$$12b_1 + 2b_2 = 0$$

solve the eqaution we get,

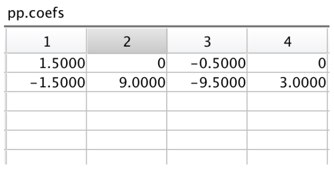
$$p_1(t) = 1.5x^3 - 0.5x$$

$$p_2(t) = -1.5x^3 + 9x^2 - 9.5x + 3$$

and the result from *naturalSpine* is same with our calculation,
matrix of coefficient:

```
0     0     0     1     0     0     0     0
1     1     1     1     0     0     0     0
0     0     0     0     1     1     1     1
0     0     0     0     8     4     2     1
3     2     1     0    -3    -2    -1     0
6     2     0     0    -6    -2     0     0
0     2     0     0     0     0     0     0
0     0     0     0    12     2     0     0
```

coefficient:

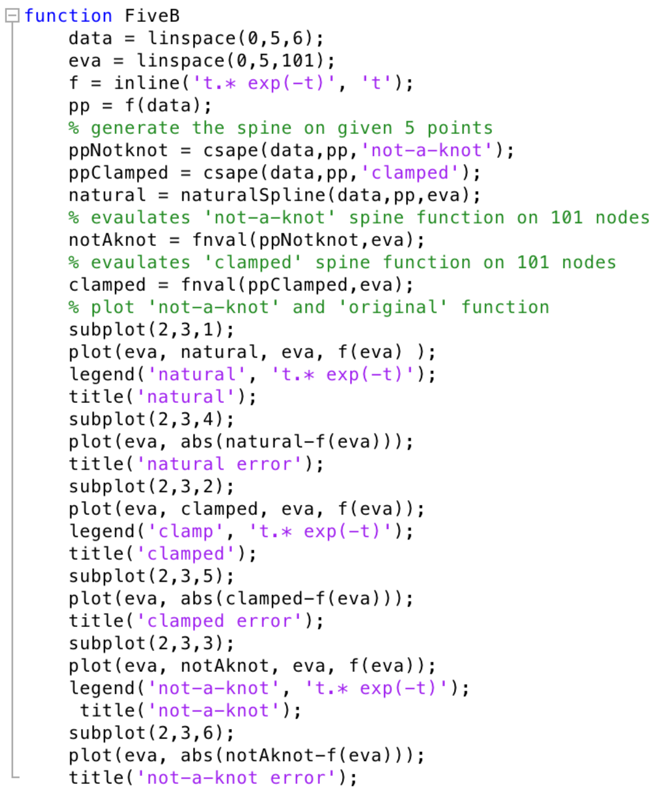

pp.coefs

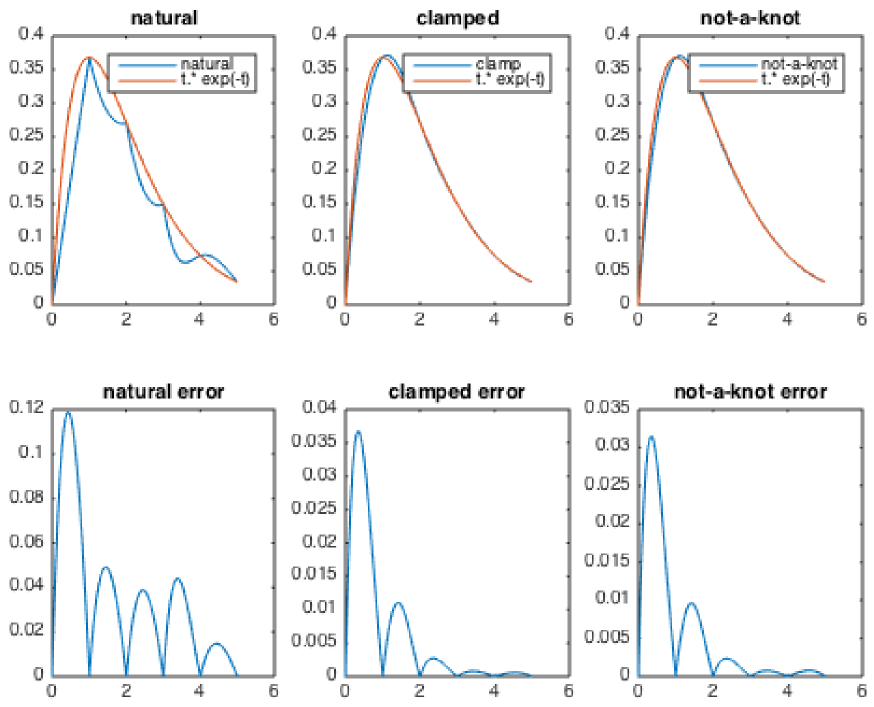
1	2	3	4
1.5000	0	-0.5000	0
-1.5000	9.0000	-9.5000	3.0000

(b) i. The results we generate from sample $\{0, 1, 2, 3, 4, 5\}$ are,

$$\{0,\ \ 0.3679,\ \ 0.2707,\ \ 0.1494,\ \ 0.0733,\ \ 0.0337\}$$

ii.


```
function FiveB
    data = linspace(0,5,6);
    eva = linspace(0,5,101);
    f = inline('t.* exp(-t)', 't');
    pp = f(data);
    % generate the spine on given 5 points
    ppNotknot = csape(data,pp,'not-a-knot');
    ppClamped = csape(data,pp,'clamped');
    natural = naturalSpline(data,pp,eva);
    % evaulates 'not-a-knot' spine function on 101 nodes
    notAknot = fnval(ppNotknot,eva);
    % evaulates 'clamped' spine function on 101 nodes
    clamped = fnval(ppClamped,eva);
    % plot 'not-a-knot' and 'original' function
    subplot(2,3,1);
    plot(eva, natural, eva, f(eva) );
    legend('natural', 't.* exp(-t)');
    title('natural');
    subplot(2,3,4);
    plot(eva, abs(natural-f(eva)));
    title('natural error');
    subplot(2,3,2);
    plot(eva, clamped, eva, f(eva));
    legend('clamp', 't.* exp(-t)');
    title('clamped');
    subplot(2,3,5);
    plot(eva, abs(clamped-f(eva)));
    title('clamped error');
    subplot(2,3,3);
    plot(eva, notAknot, eva, f(eva));
    legend('not-a-knot', 't.* exp(-t)');
     title('not-a-knot');
    subplot(2,3,6);
    plot(eva, abs(notAknot-f(eva)));
    title('not-a-knot error');
```



iii.

```
function FiveC
diary question3.out
test = [5,9,17,33,65,129];
eva = linspace(0,5,601);
f = inline('t.* exp(-t)', 't');
p = f(eva);
fprintf('    |Absolute Error: E = ||F(t)-S(t)||_2\n');
fprintf('%3s |%10s %11s %14s\n', 'n','natural','clamped','not-a-knot' );
fprintf('--------------------------------------------\n');
for i = 1:6
    data = linspace(0,5,test(i));
    pp = f(data);
    % generate the spine on given 5 points
    ppNotknot = csape(data,pp,'not-a-knot');
    ppClamped = csape(data,pp,'clamped');

    % evaulates 'not-a-knot' spine function on 601 nodes
    notAknot = fnval(ppNotknot,eva);
    % evaulates 'clamped' spine function on 601 nodes
    clamped = fnval(ppClamped,eva);
    % evaulates 'clamped' spine function on 601 nodes
    natural = naturalSpline(data,pp,eva);
    errNatual = norm(p-natural, 2);
    errNotKnot = norm(p-notAknot,2 );
    errClamped = norm(p-clamped, 2);
    fprintf('%3d |%10e %10e %10e\n', test(i), errNatual, ...
    errClamped, errNotKnot);
end
diary off
```

```
    |Absolute Error: E = ||F(t)-S(t)||_2
  n |   natural     clamped     not-a-knot
------------------------------------------
  5 |1.583456e+00 5.604828e-01 4.996096e-01
  9 |5.107820e-01 6.040910e-02 4.896355e-02
 17 |1.432833e-01 4.266713e-03 3.261746e-03
 33 |3.775102e-02 2.401296e-04 1.774617e-04
 65 |9.660809e-03 1.202700e-05 8.748925e-06
129 |2.439813e-03 5.681676e-07 4.120799e-07
```