

***Numerical Methods for Engineers and Scientists
Using MATLAB 2nd edition Esfandiari Solutions
Manual***

SKU: 9781498777421-SOLUTIONS

Problem Set (Chapter 2)

All calculations must be performed in MATLAB.

1 Evaluate the function $g(x, y) = \frac{1}{2}e^{-2x/3} \tan(y+1)$ for $x = 0.3, y = -0.7$

- (a) Using the `subs` command,
- (b) By conversion into a MATLAB function.

Solution

```
(a)
>> g = sym('exp(-2*x/3)*tan(y+1)/2');
>> x = 0.3; y = -0.7; subs(g)

>> double(subs(g))

ans =

    0.1266
```

```
(b)
>> G = matlabFunction(g);
>> G(0.3,-0.7)

ans =

    0.1266
```

2 Evaluate the function $h(x, y) = \cos(\frac{1}{3}x - 1)\sin(y + \frac{1}{2})$ for $x = \frac{3}{4}, y = 1$ using

- (a) The `subs` command,
- (b) An anonymous function.

Solution

```
(a)
>> h = sym('cos(x/3-1)*sin(y+1/2)');
>> x = 3/4; y = 1; double(subs(h))

ans =

    0.7299
```

```
(b)
>> H = @(x,y) (cos(x/3-1)*sin(y+1/2));
>> H(3/4,1)

ans =

    0.7299
```

3 Evaluate the vector function $f(x, y) = \begin{Bmatrix} x-1 \\ 2y+x \end{Bmatrix}$ for $x = 2, y = \frac{2}{3}$ using

- (a) The `subs` command,
- (b) An anonymous function.

Solution

```
(a)
>> f = sym('[x-1;2*y+x]');
>> x = 2; y = 2/3; double(subs(f))
```

```
ans =
```

```
1.0000
3.3333
```

(b)

```
>> F = @(x,y) ([x-1; 2*y+x]);
>> F(2, 2/3)
```

```
ans =
```

```
1.0000
3.3333
```

4 Evaluate the matrix function $f(x, y) = \begin{bmatrix} 1-2x & x+y \\ 0 & \cos y \end{bmatrix}$ for $x=1, y=-1$

(a) Using the `subs` command,

(b) By conversion into a Matlab function.

Solution

(a)

```
>> f = sym(' [1-2*x x+y; 0 cos(y)] ');
>> x = 1; y = -1; double(subs(f))
```

```
ans =
```

```
-1.0000    0
         0    0.5403
```

(b)

```
>> F = matlabFunction(f);
>> F(1,-1)
```

```
ans =
```

```
-1.0000    0
         0    0.5403
```

5 Consider $g(t) = t \sin(\frac{1}{2}t) + \ln(t-1)$. Evaluate dg/dt at $t = \frac{4}{3}$

(a) Using the `subs` command,

(b) By conversion into a Matlab function.

Solution

(a)

```
>> g = sym('t*sin(t/2)+log(t-1)');
>> dg = diff(g);
>> t = 4/3;
>> double(subs(dg))
```

```
ans =
```

```
4.1423
```

(b)

```
>> dG = matlabFunction(dg);
>> dG(4/3)
```

```
ans =
```

```
4.1423
```

6 Consider $h(x) = 3^{x-2} \sin x + \frac{2}{3}e^{1-2x}$. Evaluate dh/dx at $x = -0.3$

- (a) Using the `subs` command,
 (b) By conversion into a Matlab function.

Solution

(a)

```
>> h = sym('3^(x-2)*sin(x)+2*exp(1-2*x)/3');
>> dh = diff(h); x = -0.3; double(subs(dh))

ans =

-6.5536
```

(b)

```
>> dH = matlabFunction(dh); dH(-0.3)

ans =

-6.5536
```

7 Evaluate $\left[x^2 + e^{-a(x+1)}\right]^{1/3}$ when $a = -1$, $x = 3$ using an anonymous function in another anonymous function.

Solution

```
>> A = @(a,x) (x^2+exp(-a*(x+1)));
>> B = @(a,x) (A(a,x)^(1/3));
>> B(-1,3)

ans =

3.9916
```

8 Evaluate $\sqrt{\left|x + \ln\left|1 - e^{(a+2)x/3}\right|\right|}$ when $a = -3$, $x = 1$ using an anonymous function in another anonymous function.

Solution

```
>> A = @(a,x) (x+2*log(abs(1-exp((a+2)*x/3))));
>> B = @(a,x) (sqrt(abs(A(a,x))));
>> B(-3,1)

ans =

1.2334
```

In Problems 9 through 12 write a script file that employs any combination of the *flow control commands* to generate the given matrix.

$$9 \quad \mathbf{A} = \begin{bmatrix} 1 & 0 & -1 & 0 & 0 & 0 \\ 0 & 2 & 0 & -1 & 0 & 0 \\ 2 & 0 & 3 & 0 & -1 & 0 \\ 0 & 2 & 0 & 4 & 0 & -1 \\ 0 & 0 & 2 & 0 & 5 & 0 \\ 0 & 0 & 0 & 2 & 0 & 6 \end{bmatrix}$$

Solution

```

clear
clc
A = zeros(6,6);
for i = 1:6,
    for j = 1:6,
        A(i,i) = i;
        if j == i+2,
            A(i,j) = -1;
        elseif i == j+2,
            A(i,j) = 2;
        end
    end
end
end

```

```
>> A
```

```

A =
     1     0    -1     0     0     0
     0     2     0    -1     0     0
     2     0     3     0    -1     0
     0     2     0     4     0    -1
     0     0     2     0     5     0
     0     0     0     2     0     6

```

$$10 \quad A = \begin{bmatrix} 4 & 1 & -2 & 3 & 0 & 0 \\ 0 & 4 & -1 & 2 & 3 & 0 \\ 0 & 0 & 4 & 1 & -2 & 3 \\ 0 & 0 & 0 & 4 & -1 & 2 \\ 0 & 0 & 0 & 0 & 4 & 1 \\ 0 & 0 & 0 & 0 & 0 & 4 \end{bmatrix}$$

Solution

```

clear
clc
A = 4*eye(6);
for i = 1:6,
    for j = 1:6,
        if j == i+1,
            A(i,j) = (-1)^(i+1);
        elseif j == i+2,
            A(i,j) = 2*(-1)^i;
        elseif j == i+3,
            A(i,j) = 3;
        end
    end
end
end

```

```
>> A
```

```

A =
     4     1    -2     3     0     0
     0     4    -1     2     3     0
     0     0     4     1    -2     3
     0     0     0     4    -1     2
     0     0     0     0     4     1
     0     0     0     0     0     4

```

$$11 \quad \mathbf{B} = \begin{bmatrix} 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & -2 & 2 & 0 & 0 & 0 \\ -1 & 0 & 3 & 3 & 0 & 0 \\ 0 & 1 & 0 & -4 & 4 & 0 \\ 0 & 0 & -1 & 0 & 5 & 5 \\ 0 & 0 & 0 & 1 & 0 & -6 \end{bmatrix}$$

Solution

```
clear
clc
B = zeros(6,6);
for i = 1:6,
    for j = 1:6,
        B(i,i) = (-1)^(i+1)*i;
        if j == i+1,
            B(i,j) = i;
        elseif i == j+2,
            B(i,j) = (-1)^i;
        end
    end
end
```

```
>> B
```

```
B =
```

```

1      1      0      0      0      0
0     -2      2      0      0      0
-1      0      3      3      0      0
0      1      0     -4      4      0
0      0     -1      0      5      5
0      0      0      1      0     -6
```

$$12 \quad \mathbf{B} = \begin{bmatrix} 0 & 0 & -1 & 0 & 0 \\ 0 & -1 & 0 & -2 & 0 \\ 4 & 0 & 2 & 0 & -3 \\ 0 & 5 & 0 & -3 & 0 \\ 0 & 0 & 6 & 0 & 4 \end{bmatrix}$$

Solution

```
clear
clc
B = zeros(5,5);
for i = 1:5,
    for j = 1:5,
        B(i,i) = (-1)^(i+1)*(i-1);
        if j == i+2,
            B(i,j) = -i;
        elseif i == j+2,
            B(i,j) = i+1;
        end
    end
end
```

```
>> B
```

```
B =
```

```

0     0    -1     0     0
0    -1     0    -2     0
4     0     2     0    -3
0     5     0    -3     0
0     0     6     0     4
```

13 Using any combination of commands `diag`, `triu` and `tril`, construct matrix **B** from **A**.

$$\mathbf{A} = \begin{bmatrix} 2 & 1 & -1 & 2 \\ 3 & 0 & 4 & 1 \\ 1 & 5 & -1 & 3 \\ 0 & 2 & 6 & 1 \end{bmatrix}, \quad \mathbf{B} = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 3 & 0 & 0 & 0 \\ 1 & 5 & 0 & 0 \\ 0 & 2 & 6 & 0 \end{bmatrix}$$

Solution

```
>> A = [2 1 -1 2; 3 0 4 1; 1 5 -1 3; 0 2 6 1]
```

```
A =
```

```

2     1    -1     2
3     0     4     1
1     5    -1     3
0     2     6     1
```

```
>> D = diag(diag(A))
```

```
D =
```

```

2     0     0     0
0     0     0     0
0     0    -1     0
0     0     0     1
```

```
>> B = tril(A) - D
```

```
B =
```

```

0     0     0     0
3     0     0     0
1     5     0     0
0     2     6     0
```

14 Using any combination of commands `diag`, `triu` and `tril`, construct matrix **B** from **A**.

$$\mathbf{A} = \begin{bmatrix} 2 & 1 & -1 & 2 \\ 3 & 0 & 4 & 1 \\ 1 & 5 & -1 & 3 \\ 0 & 2 & 6 & 1 \end{bmatrix}, \quad \mathbf{B} = \begin{bmatrix} 0 & 1 & -1 & 2 \\ 3 & 0 & 4 & 1 \\ 0 & 5 & 0 & 3 \\ 0 & 0 & 6 & 0 \end{bmatrix}$$

Solution

```

>> A = [2 1 -1 2; 3 0 4 1; 1 5 -1 3; 0 2 6 1];
>> D = diag(diag(A)); C = triu(A) - D;
>> B = C + diag(diag(A,-1),-1)
```

B =

0	1	-1	2
3	0	4	1
0	5	0	3
0	0	6	0

15 Plot $\int_1^t e^{t-2x} \sin x dx$ versus $-1 \leq t \leq 1$, add grid and label.

Solution

```
>> syms t x
>> integ = int(exp(t-2*x)*sin(x), x, 1, t);
>> ezplot(integ, [-1, 1])
```

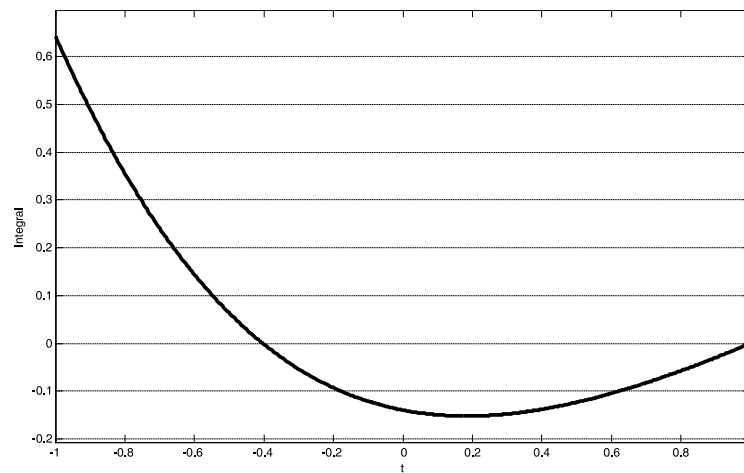


Figure Solution of Problem 15

16 Plot $\int_0^t (x+t)^2 e^{-(t-x)} dx$ versus $-2 \leq t \leq 1$, add grid and label.

Solution

```
>> syms x t
>> integ = int((x+t)^2*exp(-(t-x)), x, 0, t);
>> ezplot(integ, [-2, 1])
```

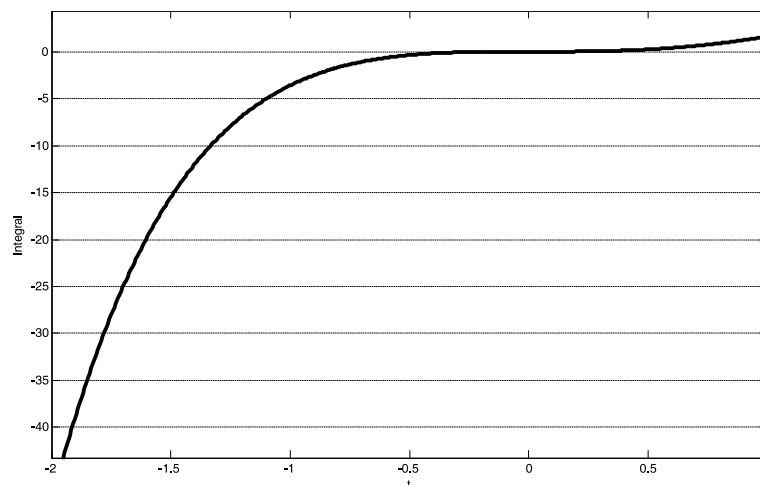


Figure Solution of Problem 16

17 Plot $y_1 = \frac{1}{3}e^{-t} \sin(t\sqrt{2})$ and $y_2 = e^{-t/2}$ versus $0 \leq t \leq 5$ in the same graph. Add grid, and label.

Solution

```
syms t
y1 = sym('(1/3)*exp(-t)*sin(sqrt(2)*t)');
y2 = sym('exp(-t/2)');
ezplot(y1,[0,5])
hold on
ezplot(y2,[0,5])
```

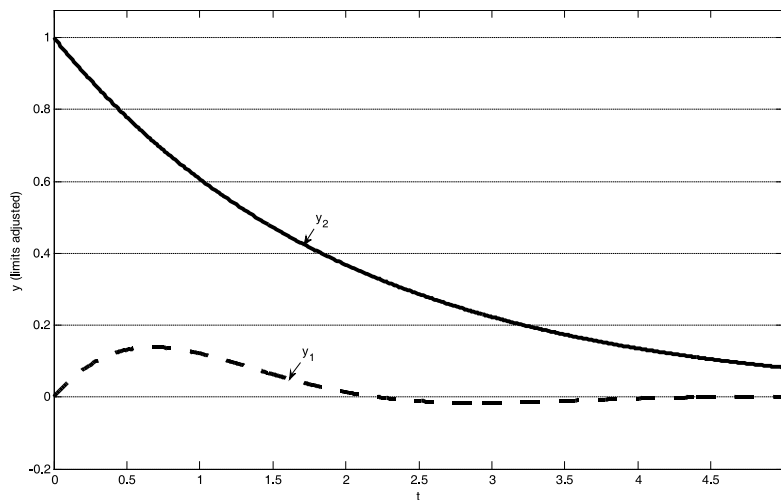


Figure Solution of Problem 17

18 Generate 100 points for each of the two functions in Problem 17 and plot versus $0 \leq t \leq 5$ in the same graph. Add grid, and label.

Solution

```
y1 = @(t) ((1/3)*exp(-t)*sin(sqrt(2)*t));
y2 = @(t) (exp(-t/2));
t = linspace(0,5);
for i = 1:100,
    yy1(i) = y1(t(i));
    yy2(i) = y2(t(i));
end
plot(t,yy1,t,yy2)
```

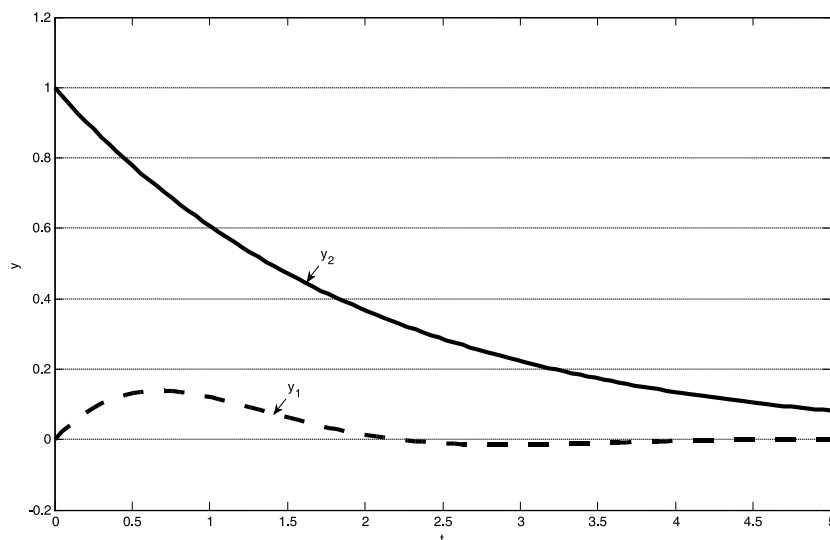


Figure Solution of Problem 18

19 Evaluate $\int_0^{\infty} \frac{\sin \omega}{\omega} d\omega$.

Solution

```
>> syms w
>> int(sin(w)/w,w,0,inf)
```

```
ans =
```

```
pi/2
```

20 Plot $u(x,t) = \cos(1.7x)\sin(3.2t)$ versus $0 \leq x \leq 5$ for four values of $t = 1, 1.5, 2, 2.5$ in a 2×2 tile. Add grid and title.

Solution

```
x = linspace(0,5); t = 1:0.5:2.5;
for i = 1:4,
    for j = 1:100,
        u(j,i) = cos(1.7*x(j))*sin(3.2*t(i)); % Generate 100 values of u for each t
    end
end

% Initiate figure
subplot(2,2,1), plot(x,u(:,1)), title('t = 1')
subplot(2,2,2), plot(x,u(:,2)), title('t = 1.5')
subplot(2,2,3), plot(x,u(:,3)), title('t = 2')
subplot(2,2,4), plot(x,u(:,4)), title('t = 2.5')
```

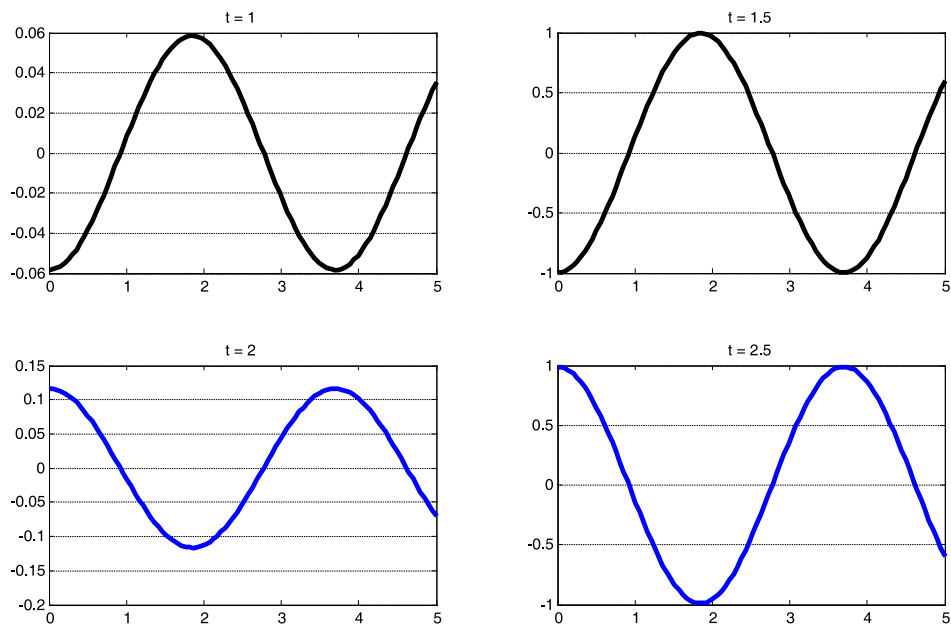


Figure Solution of Problem 20

21 Plot $u(x,t) = (1 - \sin x)e^{-(t+1)}$ versus $0 \leq x \leq 5$ for two values of $t = 1, 3$ in a 1×2 tile. Add grid and title.

Solution

```
x = linspace(0,5); t = [1,3];
for i = 1:2,
    for j = 1:100,
        u(j,i) = (1-sin(x(j)))*exp(-(t(i)+1)); % Generate 100 values of u for each t
    end
end

% Initiate figure
subplot(1,2,1), plot(x,u(:,1)), title('t = 1')
subplot(1,2,2), plot(x,u(:,2)), title('t = 3')
```

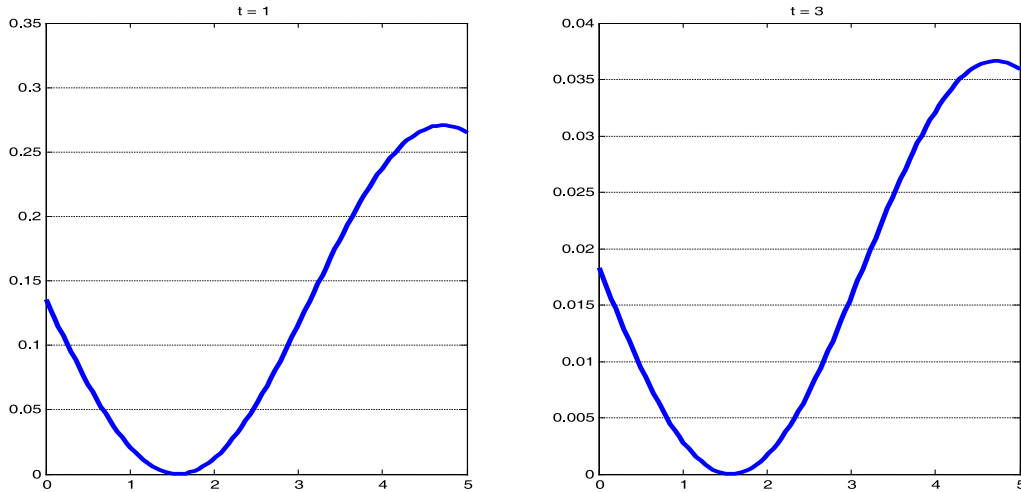


Figure Solution of Problem 21

22 Given that $f(x) = e^{-2x} + \cos(x+1)$, plot $f'(x)$ versus $0 \leq x \leq 8$.

Solution

```
>> f = sym('exp(-2*x)+cos(x+1)');
>> df = matlabFunction(diff(f));
>> ezplot(df,[0,8])
```

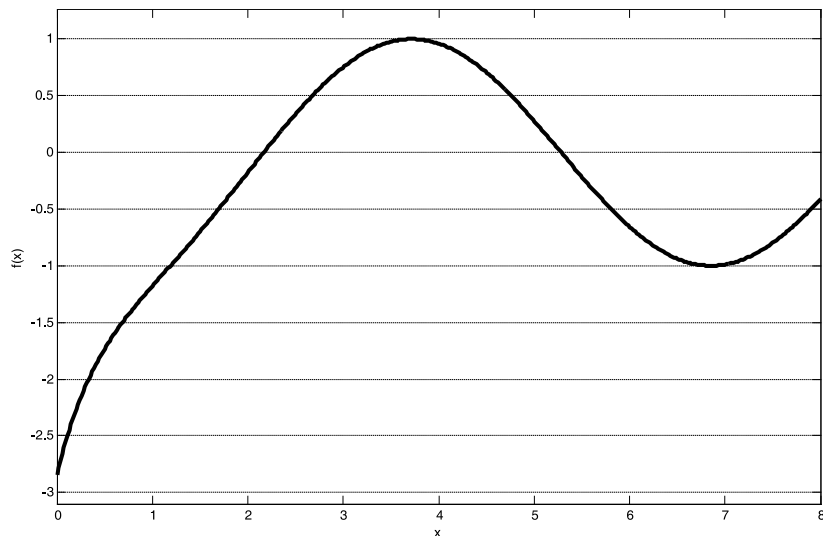


Figure Solution of Problem 22

- 23** Write a user-defined function with function call `val = f_eval(f,a,b)` where f is an anonymous function, and a and b are constants such that $a < b$. The function calculates the midpoint m of the interval $[a,b]$ and returns the value of $f(a) + \frac{1}{2}f(m) + f(b)$. Execute `f_eval` for $f = e^{-x/3}$, $a = -4$, $b = 2$.

Solution

```
function val = f_eval(f,a,b)
m = (a + b)/2;
val = f(a) + f(m)/2 + f(b);
```

```
>> f = @(x) (exp(-x/3));
>> val = f_eval(f,-4,2)
```

```
val =

    5.0049
```

- 24** Write a user-defined function with function call `m = mid_seq(a,b,tol)` where a and b are constants such that $a < b$, and `tol` is a specified tolerance. The function first calculates the midpoint m_1 of the interval $[a,b]$, then the midpoint m_2 of $[a,m_1]$, then the midpoint m_3 of $[a,m_2]$, and so on. The process terminates when two successive midpoints are within `tol` of each other. Allow a maximum of 20 iterations. The output of the function is the sequence $m_1, m_2, m_3, \dots$. Execute the function for $a = -4$, $b = 10$, $\text{tol} = 10^{-3}$.

Solution

```
function m = mid_seq(a,b,tol)
m = zeros(20); % Pre-allocate
m(1) = (a + b)/2;
for i = 2:20,
    m(i) = (a + m(i-1))/2;
    if abs(m(i) - m(i-1)) < tol,
        break
    end
end
```

```
>> m = mid_seq(-4,10,1e-3)
```

```
m =

Columns 1 through 8

    3.0000    -0.5000    -2.2500    -3.1250    -3.5625    -3.7813    -3.8906    -3.9453

Columns 9 through 14

   -3.9727   -3.9863   -3.9932   -3.9966   -3.9983   -3.9991
```

- 25** Write a user-defined function with function call `C = temp_conv(F)` where F is temperature in Fahrenheit, and C is the corresponding temperature in Celsius. Execute the function for $F = 87$.

Solution

```
function C = temp_conv(F)
C = (F-32)*100/180;
```

```
>> C = temp_conv(87)
```

```
C =

    30.5556
```

- 26** Write a user-defined function with function call $P = \text{partial_eval}(f, a)$ where f is a function defined symbolically, and a is a constant. The function returns the value of $f' + f''$ at $x = a$. Execute the function for $f = 3x^2 - e^{x/3}$, and $a = 1$.

Solution

```
function P = partial_eval(f,a)
del = diff(f,'x') + diff(f,2,'x');
x = a; P = double(subs(del));
```

```
>> f = sym('3*x^2-exp(x/3)'); P = partial_eval(f,1)

P =
    11.3797
```

- 27** Write a user-defined function with function call $P = \text{partial_eval2}(f, g, a)$ where f and g are functions defined symbolically, and a is a constant. The function returns the value of $f' + g'$ at $x = a$. Execute the function for $f = x^2 + e^{-x}$, $g = \sin(0.3x)$, and $a = 0.8$.

Solution

```
function P = partial_eval2(f,g,a)
del = diff(f,'x') + diff(g,'x');
x = a; P = double(subs(del));
```

```
>> f = sym('x^2+exp(-x)'); g = sym('sin(0.3*x)'); P = partial_eval2(f,g,0.8)

P =
    1.4421
```

- 28** Write a user-defined function with function call $[r, k] = \text{root_finder}(f, x_0, k_{\max}, \text{tol})$ where f is an anonymous function, x_0 is a specified value, $k_{\max}$ is the maximum number of iterations, and tol is a specified tolerance. The function sets $x_1 = x_0$, calculates $|f(x_1)|$, and if it is less than the tolerance, then x_1 approximates the root r . If not, it will increment x_1 by 0.01 to obtain x_2 , repeat the procedure, and so on. The process terminates as soon as $|f(x_k)| < \text{tol}$ for some k . The outputs of the function are the approximate root and the number of iterations it took to find it. Execute the function for $f(x) = x^2 - 3.3x + 2.1$, $x_0 = 0.5$, $k_{\max} = 50$, $\text{tol} = 10^{-2}$.

Solution

```
function [r, k] = root_finder(f,x0,kmax,tol)
x(1) = x0;
if abs(f(x(1))) < tol,
    r = x(1);
end
for k = 2:kmax,
    x(k) = x(k-1) + 0.01;
    if abs(f(x(k))) < tol,
        r = x(k); break; end
end
```

```
>> f = @(x) (x^2-3.3*x+2.1); [r, k] = root_finder(f,0.5,50,1e-2)

r =
    0.8600

k =
    37
```

29 Repeat Problem 28 for $f(x) = 3 + \ln(2x-1) - e^x$, $x_0 = 1$, $k_{\max} = 25$ $\text{tol} = 10^{-2}$.

Solution

```
function [r, k] = root_finder(f, x0, kmax, tol)
x(1) = x0;
if abs(f(x(1))) < tol,
    r = x(1);
end
for k = 2:kmax,
    x(k) = x(k-1) + 0.01;
    if abs(f(x(k))) < tol,
        r = x(k); break, end
end
```

```
>> f = @(x) (3+log(2*x-1)-exp(x));
>> [r, k] = root_finder(f, 1, 25, 1e-2)
```

```
r =
    1.2100
```

```
k =
    22
```

30 Write a user-defined function with function call `[opt, k] = opt_finder(fp, x0, kmax, tol)` where `fp` is the derivative (as a MATLAB function) of a given function f , x_0 is a specified value, $k_{\max}$ is the maximum number of iterations, and `tol` is a specified tolerance. The function sets $x_1 = x_0$, calculates $|fp(x_1)|$, and if it is less than the tolerance, then x_1 approximates the critical point `opt` at which the derivative is near zero. If not, it will increment x_1 by 0.1 to obtain x_2 , repeat the procedure, and so on. The process terminates as soon as $|fp(x_k)| < \text{tol}$ for some k . The outputs are the approximate optimal point and the number of iterations it took to find it. Execute the function for $f(x) = x + (x-2)^2$, $x_0 = 1$, $k_{\max} = 50$ $\text{tol} = 10^{-3}$.

Solution

```
function [opt, k] = opt_finder(fp, x0, kmax, tol)
x = zeros(kmax); % Pre-allocate
x(1) = x0;
if abs(fp(x(1))) < tol,
    opt = x(1);
end
for k = 2:kmax,
    x(k) = x(k-1) + 0.1;
    if abs(fp(x(k))) < tol,
        opt = x(k); break, end
end
```

```
>> f = sym('x + (x-2)^2');
>> fp = matlabFunction(diff(f));
>> [opt, k] = opt_finder(fp, 1, 50, 1e-3)
```

```
opt =
    1.5000
```

```
k =
    6
```