**HW 1**

**(MATH/CS 375)**

**Jacob Hurst**

**01/25/18**

**Problem #1: z = [10 40 50 80 30 70 60 90]**

1. >> z = z(z>=50)  
     
   z = [50 80 70 60 90]
2. >> z(1:2:7) = zeros(1,4)  
     
   z = [0 40 0 80 0 70 0 90]
3. >> z(7:-2:1) = fliplr(z(1:2:7))  
     
   z = [10 40 50 80 30 70 60 90]
4. >> z([3 4 8 1]) = []  
     
   z = [40 30 70 60]

**Problem #2: linspace function vs. colon notation**

1. >> t = 1:6:37

t = 1 7 13 19 25 31 37

>> t\_lin = linspace(1, 37, 7)

t\_lin = 1 7 13 19 25 31 37

1. >> x = -2:4  
     
   x = -2 -1 0 1 2 3 4  
     
     
   >> x\_lin = linspace(-2, 4, 7)

x\_lin = -2 -1 0 1 2 3 4

**Problem #3: colon notation vs. linspace function**

1. >> v = linspace(-3, 0, 5)  
     
   v = -3.0000 -2.2500 -1.5000 -0.7500 0  
     
     
   >> v\_colon = (-3:0.75:0)  
     
   v\_colon = -3.0000 -2.2500 -1.5000 -0.7500 0
2. >> r = linspace(2,4,7)

r = 2.0000 2.3333 2.6667 3.0000 3.3333 3.6667 4.0000

>> r\_colon = (2:(1/3):4)

r\_colon = 2.0000 2.3333 2.6667 3.0000 3.3333 3.6667 4.0000

**Problem #4: CPU time**

1. Code:

% Using a for-loop

clc;

tic;

x = 0:(pi/1e8):pi;

y = cos(x);

summation = 0;

for k = 1:length(x)

summation = summation + (x(k) \* y(k));

end

toc;

disp(summation);

Output:  
Elapsed time is 2.181191 seconds.

-6.3662e+07

1. Code:

% Using a built-in sum command

clc;

tic;

x = 0:(pi/1e8):pi;

y = cos(x);

summation = sum(x\*y');

toc;

disp(summation);

Output:

Elapsed time is 1.497960 seconds.

-6.3662e+07

1. Code:

% Using vectorized implementation

clc;

tic;

summation=sum((0:(pi/1e8):pi).\*cos((0:(pi/1e8):pi)));

toc;

disp(summation);

Output:

Elapsed time is 0.210878 seconds.

-6.3662e+07

**Problem #5: Plotting 2x2 subfigures.**   
% Some figures involving x, x^2, x^3, x^4.  
N = 1000;

x = linspace(0, 1, N);

y1 = x;

y2 = x.^2;

y3 = x.^3;

y4 = x.^4;

% plot - standard plot, not a large range without cramming x & y axis, useful for linear functions

subplot(2, 2, 1);

set(gca, 'fontsize', 20);

plot(x, y1, 'r-', 'Linewidth', 2);

hold on;

plot(x, y2, 'g-', 'Linewidth', 2);

plot(x, y3, 'b-', 'Linewidth', 2);

plot(x, y4, 'k-', 'Linewidth', 2);

xlabel('x');

ylabel('y (plot)');

legend('y = x', 'y = x^2', 'y = x^3', 'y = x^4');

axis([0 1 0 1]);

% semilogx - allows for very large scale on x axis, useful for substantial change in x axis

subplot(2, 2, 2);

set(gca, 'fontsize', 20);

semilogx(x, y1, 'r-', 'Linewidth', 2);

hold on;

semilogx(x, y2, 'g-', 'Linewidth', 2);

semilogx(x, y3, 'b-', 'Linewidth', 2);

semilogx(x, y4, 'k-', 'Linewidth', 2);

xlabel('x');

ylabel('y (semilogx)');

axis([0 1 0 1]);

% semilogy - allows for very large scale on y axis, useful for timescales

subplot(2, 2, 3);

set(gca, 'fontsize', 20);

semilogy(x, y1, 'r-', 'Linewidth', 2);

hold on;

semilogy(x, y2, 'g-', 'Linewidth', 2);

semilogy(x, y3, 'b-', 'Linewidth', 2);

semilogy(x, y4, 'k-', 'Linewidth', 2);

xlabel('x');

ylabel('y (semilogy)');

axis([0 1 0 1]);

**Problem #5 (Continued): Plotting 2x2 subfigures**

% loglog - allows for very large scale on x & y axis, useful for exponential functions

subplot(2, 2, 4);

set(gca, 'fontsize', 20);

loglog(x, y1, 'r-', 'Linewidth', 2);

hold on;

loglog(x, y2, 'g-', 'Linewidth', 2);

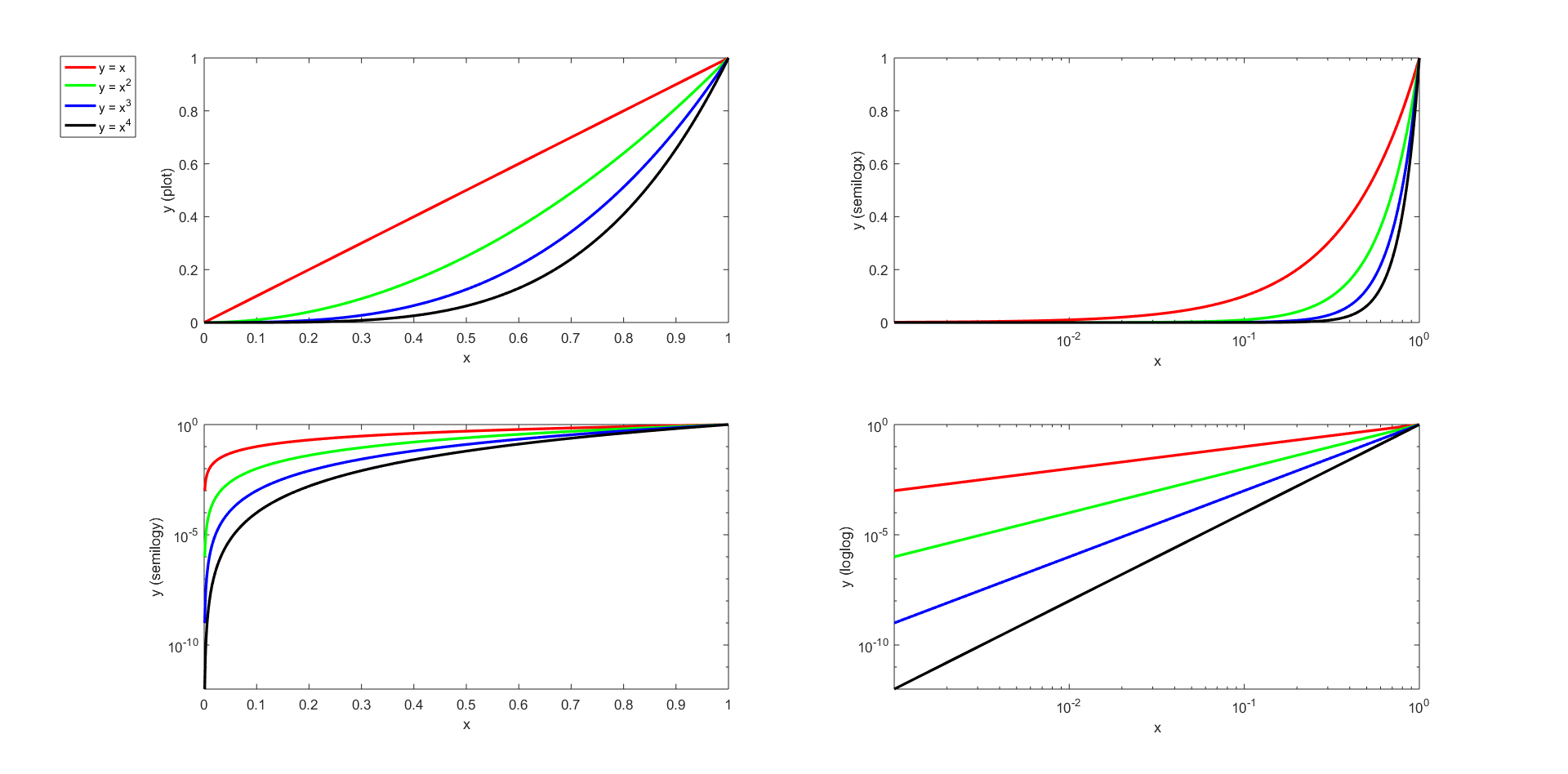
loglog(x, y3, 'b-', 'Linewidth', 2);

loglog(x, y4, 'k-', 'Linewidth', 2);

xlabel('x');

ylabel('y (loglog)');

axis([0 1 0 1]);

**Figure 5:**

**Problem #6: Functions**

1. **Code:**% my\_mean function

function output = my\_mean(fun, a, b, N)

if(a > b)

error('Invalid interval [' + a + ', ' + b + ']');

end

u = linspace(a, b, N);

v = fun(u);

output = (1/N)\*(sum(v));

end

**Output:**

>>my mean(@exp, 0, 5, 100)

Ans = 29.9411

1. **Code:**

% my\_fun function

function output = my\_fun(a) %#ok<INUSD>

output = @(a)(a.\*exp(a));

end

**Output:**

>>my\_fun(x)

ans = function\_handle with value: @(a)(a.\*exp(a))

**Problem #6 (continued): Functions**

1. **Code:**

clc;

a = -1;

b = 1;

N = [10 20 40 80 160 320 640 1280];

x = linspace(a, b, length(N));

real = 1/exp(1);

M = [0 0 0 0 0 0 0 0];

err = [0 0 0 0 0 0 0 0];

xlabel('N');

ylabel('Error');

set(gca, 'fontsize', 10);

hold on;

for n = N

M(N==n) = my\_mean(my\_fun(x), a, b, n);

err(N==n) = abs(M(N==n) - real);

semilogy(n, err(N==n), '-s', 'MarkerSize', 10,...

'MarkerFaceColor', [1, .6, .6],...

'MarkerEdgeColor', 'red');

end

T = table(N', M', err');

T.Properties.VariableNames = {'N' 'M' 'Error'};

**Problem #6 (continued): Functions**

A picture containing sky

Description generated with high confidence**Figure 6c:**

**Problem #6 (continued): Functions**

N M (approximation) Error (abs(1/e – M))

10 0.458663902492169 0.0907844613207265

20 0.410629165764081 0.0427497245926385

40 0.388643266967069 0.0207638257956271

80 0.378114329516468 0.0102348883450257

160 0.372960818849529 0.00508137767808697

320 0.370411197949211 0.00253175677776873

640 0.369143097014304 0.00126365584286170

1280 0.368510714758770 0.000631273587327397

Observations:   
  
The size in error decreases proportionately to the number of digits in N,   
e.g. N = 10,…,40 -> error of 0.0x, N = 160, 320, 640 -> error of 0.00x, N = 1280 -> error of 0.000x.

The level of accuracy increases significantly as N gets large. As N->inf, the error would approach 0.