Class Test (model solution)

1. Write a MATLAB function called `banded_matrix(j,k,n)` which outputs a square matrix of size `n` with ones in the main diagonal, ones in `j` first lower diagonals, ones in the `k` first upper diagonals, and zero everywhere else. For example, `banded_matrix(1,2,6)` should output,

$$\begin{pmatrix} 1 & 1 & 1 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 & 0 & 0 \\ 0 & 1 & 1 & 1 & 1 & 0 \\ 0 & 0 & 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 & 1 & 1 \end{pmatrix}$$

Test your code with `k = 3, j = 4, n = 8`.

Solution:

```
function A = banded_matrix(k,l,n)
if k>n-1 || l>n-1
    error('wrong dimensions')
else
A = eye(n);
for i=1:l
    A = A + diag(ones(1,n-i),i);
end
for i=1:k
    A = A + diag(ones(1,n-i),-i);
end
end
```

```
>> banded_matrix(3,4,8)

ans =

     1     1     1     1     1     0     0     0
     1     1     1     1     1     1     0     0
     1     1     1     1     1     1     1     0
     1     1     1     1     1     1     1     1
     0     1     1     1     1     1     1     1
     0     0     1     1     1     1     1     1
     0     0     0     1     1     1     1     1
     0     0     0     0     1     1     1     1
```

2. An iterative method to compute $\sqrt{a}$ has the form

$$x_n = \frac{1}{2}\left(x_{n-1} + \frac{a}{x_{n-1}}\right),$$

$n = 1, 2, \ldots,$ with $x_0 = a$. Write a function `mySqrt(a,n)` which implements this calculation in a recursive function. Test your function for `a = 15129` and `n = 10`.

Solution:

```
function root = mySqrtRec(a,n)
if n == 0
root = a;
else
root = (mySqrtRec(a,n-1) + a/mySqrtRec(a,n-1))/2;
end

>> root = mySqrtRec(15129,10)
root =
123.0000
```

3. Write a function `taylor_log(n,x)` which outputs the Taylor approximation $\log(1+x) \approx -\sum_{k=1}^{n} \frac{(-x)^k}{k}$. The function should be able to take `x` as a `linspace` array. Plot `taylor_log(5,x)` and `log(1+x)`, on the same plot, for `x = linspace(-1/2,1,100)`.

   Solution:

   ```
   function l = taylor_log(n,x)
   l = (-sum((-x').^(1:n)./(1:n),2))';
   end
   ```

   And the plot:

   ```
   x = linspace(-1/2,1,100);
   plot(x, log(1+x)), hold on
   plot(x, taylor_log(5,x),'--');
   xlabel('x axis');
   ylabel('y axis');
   title('log(1+x)');
   legend('exact', 'Taylor approximation')
   ```

4. Using MATLAB, classify the solutions of the following systems of equations into unique, doesn't exist, or infinite. Explain your reasoning. Calculate the solution(s) if (they) exist.

   a)

$$x - 2y + z = -2$$
$$2x + 2y - 2z = 4$$
$$3x - z = 2$$

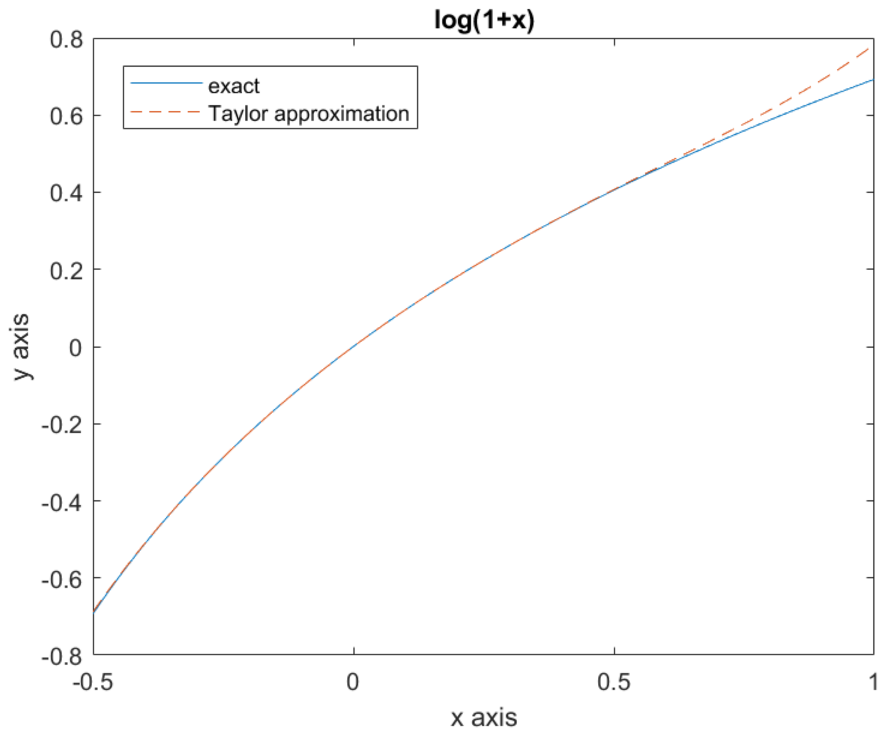

Figure 1: Q3

b)

$$\begin{aligned} 2x - y + 2z - w &= 0 \\ x + y - z - w &= 0 \\ x - y - z + w &= 0 \\ 2x - y + z - 2w &= 1 \end{aligned}$$

Solution: We analyze the system `Ax = b`:

a)

```
>> A = [1 -2 1; 2 2 -2; 3 0 -1]; b = [-2;4;2];
>> rank(A)

ans =

     2

>> rank([A b])

ans =

     2
```

Since the rank of the augmented matrix is the same as the rank of `A` and since `A` is not full rank, there are infinite solutions:

```
p = null(A)

p =

    0.2673
    0.5345
    0.8018

>> x_p = pinv(A)*b

x_p =

    0.4286
    0.8571
   -0.7143
```

The solutions are of the form `x_p + alpha*p`, for any constant `alpha`. b) The system is homogeneous so we investigate its null space

```
>> A = [2 -1 2 -1; 1 1 -1 -1; 1 -1 -1 1; 2 -1 1 -2]; b =
   [0;0;0;1];
>> rank(A)

ans =

     4
```

The matrix is full rank so there is a unique solution:

```
>> x = A\b

x =

   -0.3333
   -0.6667
   -0.3333
   -0.6667
```

5. Create a data set using `x = linspace(0,2*pi,100)` and `y = sin(x)+0.2*rand(size(x))`. Write a script file which plots this data set and regressions curve for the estimation function

$$\hat{y}(x) = \alpha_n x^n + \alpha_{n-1} x^{n-1} + \ldots + \alpha_0$$

for `n = 2,4,6`.

Solution:

```
x = linspace(0,2*pi,100); y = sin(x) + 0.2*randn(size(x
   )); plot(x, y,'.'); hold on
A = x'.^(6:-1:0);
alpha = pinv(A)*y';
plot(x,A*alpha);
title ( ' Plot of data set with regression curve ');
xlabel('x axis');
ylabel('y axis');
```

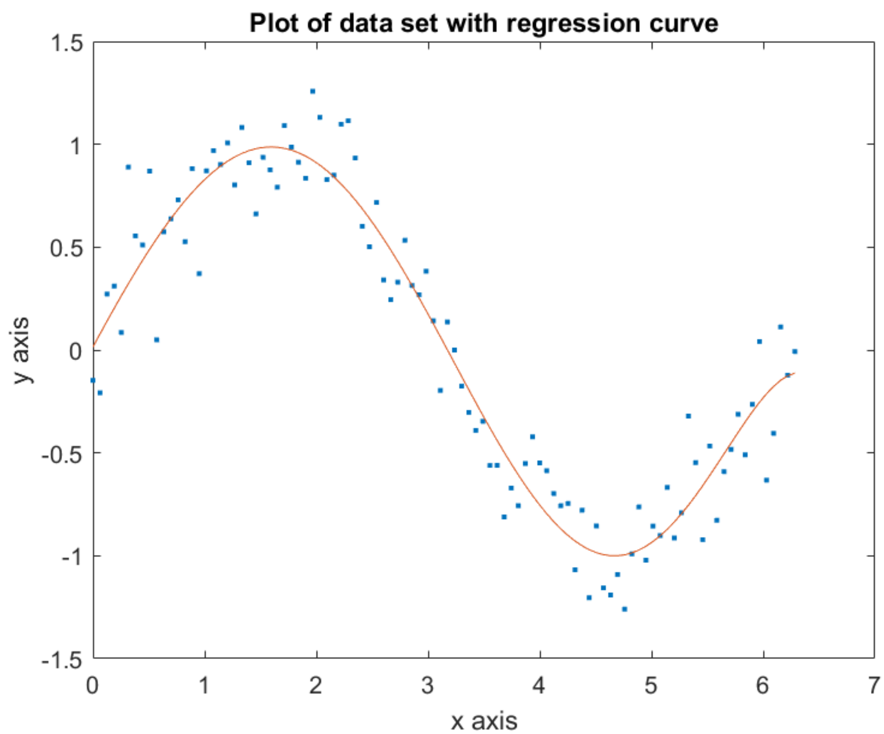

Figure 2: Q5