Exercise 1:

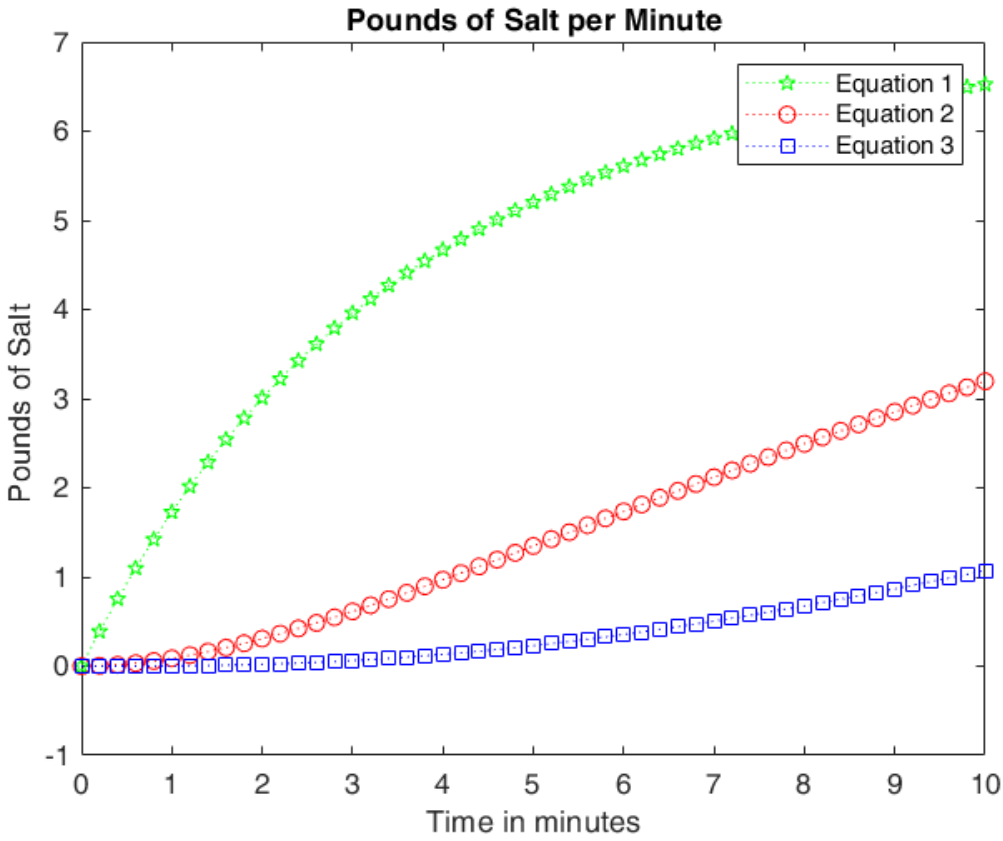
Code:

```
[Q] = [14.4721 10.0000 5.5279; 8.9443 10.0000 8.9443; 5.5279 10.0000 14.4721];
t = (0:0.2:10);

% Equation 1
x = 10 - 14.4721 * exp(-0.2618 * t) + 10 * exp(-0.2 * t) - 5.5279 * exp(-0.0382 * t);
plot(t, x, 'g:p', 'DisplayName', 'Equation 1');

% Equation 2
y = 10 + 8.9443 * exp(-0.2618 * t) - 10 * exp(-0.2 * t) - 8.9443 * exp(-0.0382 * t);
hold on;
plot(t, y, 'r:o', 'DisplayName', 'Equation 2');

% Equation 3
z = 10 - 5.5279 * exp(-0.2618 * t) + 10 * exp(-0.2 * t) - 14.4721 * exp(-0.0382 * t);
plot(t, z, 'b:s', 'DisplayName', 'Equation 3');

title('Pounds of Salt per Minute');
xlabel('Time in minutes');
ylabel('Pounds of Salt ');
legend('show');
```

Explanation

The code defines three equations modeling the rate of salt dissolution over time in three tanks, each with distinct coefficients and exponential decay rates. It then plots the solutions of these equations over a time range of 0 to 10, taking 0.2 steps to plot data.

Result:



Exercise 2:

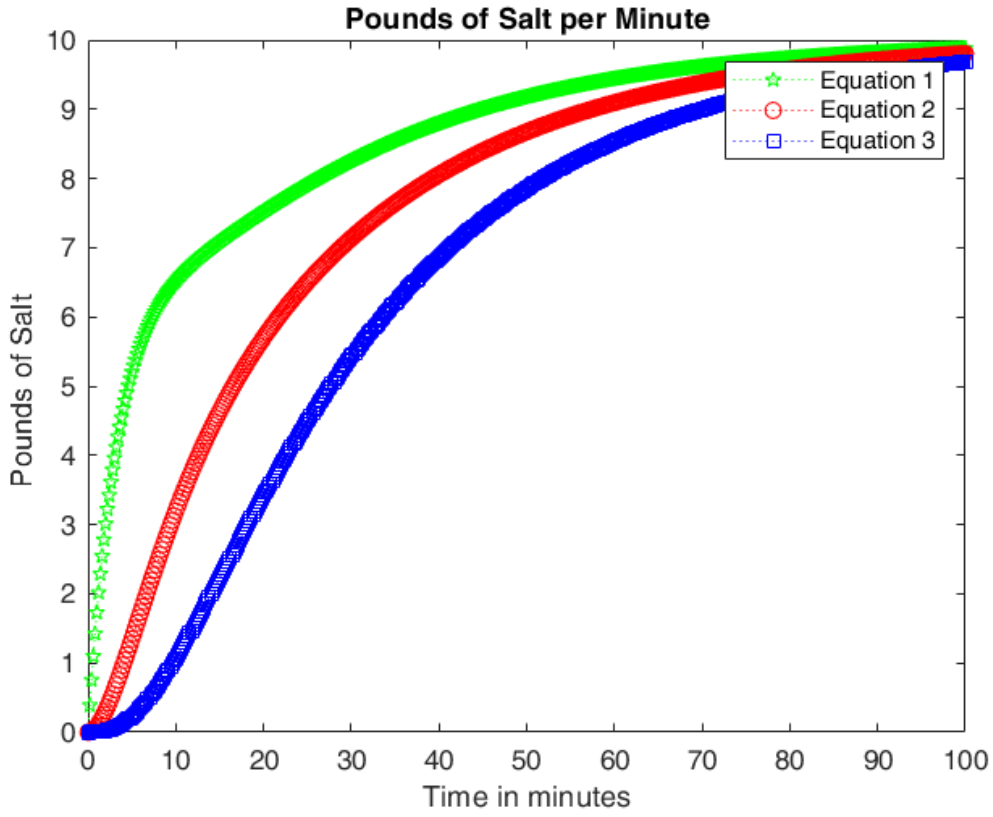
Code:

```
[Q] = [14.4721 10.0000 5.5279; 8.9443 10.0000 8.9443; 5.5279 10.0000 14.4721];
t = (0:0.2:100);

% Equation 1
x = 10 - 14.4721 * exp(-0.2618 * t) + 10 * exp(-0.2 * t) - 5.5279 * exp(-0.0382 * t);
plot(t, x, 'g:p', 'DisplayName', 'Equation 1');

% Equation 2
y = 10 + 8.9443 * exp(-0.2618 * t) - 10 * exp(-0.2 * t) - 8.9443 * exp(-0.0382 * t);
hold on;
plot(t, y, 'r:o', 'DisplayName', 'Equation 2');

% Equation 3
z = 10 - 5.5279 * exp(-0.2618 * t) + 10 * exp(-0.2 * t) - 14.4721 * exp(-0.0382 * t);
plot(t, z, 'b:s', 'DisplayName', 'Equation 3');

title('Pounds of Salt per Minute');
xlabel('Time in minutes');
ylabel('Pounds of Salt ');
legend('show');
```

Explanation

Nearly identical to exercise 1. The only difference is that it steps through a larger time range, 0 to 100, taking 0.2 steps.

Result:



Exercise 3:

Code:

```
% Define the matrix (A) to be solved
A = [-0.3, 0, 0.1;
     0.1, -0.1, 0;
     0, 0.1, -0.1];

% Compute eigenvalues and eigenvectors of matrix A
[eigenvectors, diagonal_matrix] = eig(A);

% Display the eigenvectors
disp('Eigenvectors (P):');
disp(eigenvectors);

% Display the diagonal matrix containing the eigenvalues
disp('Diagonal Matrix (D):');
disp(diagonal_matrix);

% Extract the eigenvalues
eigenvalues = diag(diagonal_matrix);

% Extract the first eigenvector and corresponding eigenvalue
first_eigenvector = eigenvectors(:,3);
first_eigenvalue = eigenvalues(1);
second_eigenvalue = eigenvalues(2);
third_eigenvalue = eigenvalues(3);

% First part
selector = [1,0,0;
            0,0,0;
            0,0,0];
multiplier = eigenvectors * selector * inv(eigenvectors);
disp(['e^(', num2str(first_eigenvalue), 't) * ']);
disp(multiplier);

% Second part
selector = [0,0,0;
            0,1,0;
            0,0,0];
multiplier = eigenvectors * selector * inv(eigenvectors);
disp(['e^(', num2str(second_eigenvalue), 't) * ']);
disp(multiplier);

% Third part
selector = [0,0,0;
            0,0,0;
            0,0,1];
multiplier = eigenvectors * selector * inv(eigenvectors);
disp(['e^(', num2str(third_eigenvalue), 't) * ']);
disp(multiplier);
```

Explanation

This code calculates P and D matrix associated with matrix A. Then it displays the exponential matrix using the format: $e^{\lambda t} * P * A * P^{-1}$

Result:

e^(-0.2618t) *

1.8944	0.7236	-1.1708
-1.1708	-0.4472	0.7236
0.7236	0.2764	-0.4472

e^(-0.2t) *

-1.0000	-1.0000	1.0000
1.0000	1.0000	-1.0000
-1.0000	-1.0000	1.0000

e^(-0.038197t) *

0.1056	0.2764	0.1708
0.1708	0.4472	0.2764
0.2764	0.7236	0.4472