

Assignment 2023

1. (30 marks)

The cost of refined oil when shipped via the Malacca Straits to Japan in dollars per kiloliter was given (Uchiyama, 1968) as the linear sum of the crude oil cost, the insurance, customs, freight cost for the oil, loading and unloading cost, sea berth cost, submarine pipe cost, storage cost, tank area cost, refining cost, and freight cost of products as

$$\begin{aligned} c = & c_c + c_i + c_x + \frac{2.09 \times 10^4 t^{-0.3017}}{360} + \frac{1.064 \times 10^6 a t^{0.4925}}{52.47 q(360)} \\ & + \frac{4.242 \times 10^4 a t^{0.7952} + 1.813 i p (n t + 1.2 q)^{0.861}}{52.47 q(360)} \\ & + \frac{4.25 \times 10^3 a (n t + 1.2 q)}{52.47 q(360)} + \frac{5.042 \times 10^3 q^{-0.1899}}{360} \\ & + \frac{0.1049 q^{0.671}}{360} \end{aligned}$$

where a = annual fixed charges, fraction (0.20)

c_c = crude oil price, \$/kL (12.50)

c_i = insurance cost, \$/kL (0.50)

c_x = customs cost, \$/kL (0.90)

i = interest rate (0.10)

n = number of ports (2)

p = land price, \$/m² (7000)

q = refinery capacity, bbl/day

t = tanker size, kL

Given the values indicated in the parentheses, use a computer code to compute the minimum cost of oil and the optimum tanker size t and refinery size q by Newton's method. (Note that 1 kL = 6.29bbl)

Code:

```
% Define the symbols
syms tanker_size refinery_size
% Constants
amplitude = 0.20;
cc_value = 12.50;
ci_value = 0.50;
cx_value = 0.90;
i_value = 0.10;
n_value = 2;
p_value = 7000;
% Define the cost equation
cost_eq = cc_value + ci_value + cx_value + (2.09 * 10^4 .* tanker_size.^-0.3017 / 360) + (1.064 * 10^6 .* amplitude .* tanker_size.^0.4925 / (52.47 .* refinery_size .* 360)) + ((4.242 * 10^4 .* amplitude .* tanker_size.^0.7952 + 1.813 .* i_value .* p_value .* (n_value + 1.2 .* refinery_size).^0.861) / (52.47 .* refinery_size .* 360)) + (4.25 * 10^3 .* amplitude .* (n_value + 1.2 .* refinery_size) / (52.47 .* refinery_size .* 360)) + (5.04 * 10^3 .* refinery_size.^-0.1899 / 360) + (0.1049 .* refinery_size.^0.671 / 360);
% Create a function handle for optimization
cost_func = @(x) double(subs(cost_eq, [tanker_size, refinery_size], x));
% Set initial guesses
initial_estimates = [1.0, 1.0];
% Perform optimization
optim_options = optimoptions('fminunc', 'Display', 'iter', 'Algorithm', 'quasi-newton');
[optimal_params, min_cost] = fminunc(cost_func, initial_estimates, optim_options);
% Display results
optimal_tanker_size = optimal_params(1);
optimal_refinery_size = optimal_params(2);

disp(['The minimum cost of oil is ', num2str(min_cost)])
disp(['The optimum tanker size t is ', num2str(optimal_tanker_size)])
disp(['The optimum refinery size q is ', num2str(optimal_refinery_size)])
```

Output:

```
17          54          25.5055          1          0.00176
18          57          24.3462          1          0.00105
19          60          23.3148          1          0.000631
                                     First-order
Iteration  Func-count      f(x)      Step-size  optimality
20          63          22.3984          1          0.000378
21          66          21.5863          1          0.000226
22          69          20.869          1          0.000136
23          72          20.2394          1          8.17e-05
24          75          19.6915          1          4.93e-05
25          78          19.2211          1          2.99e-05
26          81          18.8253          1          1.84e-05
27          84          18.5022          1          1.14e-05

Local minimum found.

Optimization completed because the size of the gradient is less than
the value of the optimality tolerance.
|
<stopping criteria details>
The minimum cost of oil is 18.5022
The optimum tanker size t is 51620.9834
The optimum refinery size q is 43253.9819
```

2. (30 marks)

Enzymatic reactions are used extensively to characterize biologically mediated reactions in environmental engineering. Proposed rate expressions for an enzymatic reaction are given below where $[S]$ is the substrate concentration and v_0 is the initial rate of reaction. Which formula best fits the experimental data? (Here k and K are fitting parameters.)

$$v_0 = k[S] \quad v_0 = \frac{k[S]}{K + [S]} \quad v_0 = \frac{k[S]^2}{K + [S]^2} \quad v_0 = \frac{k[S]^3}{K + [S]^3}$$

[S], M	Initial Rate, 10^{-6} M/s
0.01	6.3636×10^{-5}
0.05	7.9520×10^{-3}
0.1	6.3472×10^{-2}
0.5	6.0049
1	17.690
5	24.425
10	24.491
50	24.500
100	24.500

Code:

```
% Define the substrate concentration [substrate] and initial rate init_rate
substrate = [0.01, 0.05, 0.1, 0.5, 1, 5, 10, 50, 100];
init_rate = [6.363e-5, 7.9520e-3, 6.3472e-2, 6.0049, 17.690, 24.425, 24.491, 24.500, 24.500];

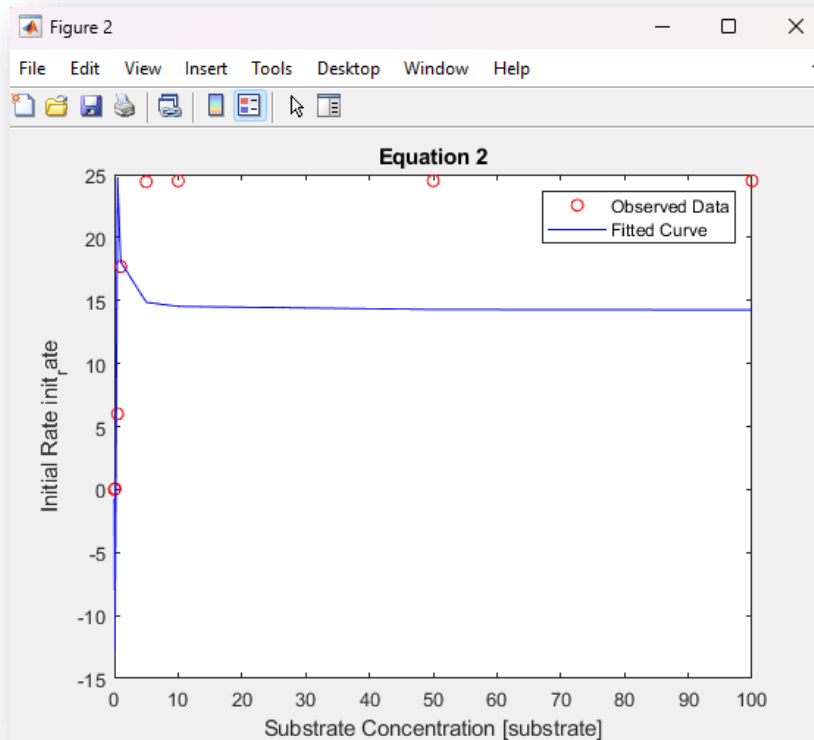
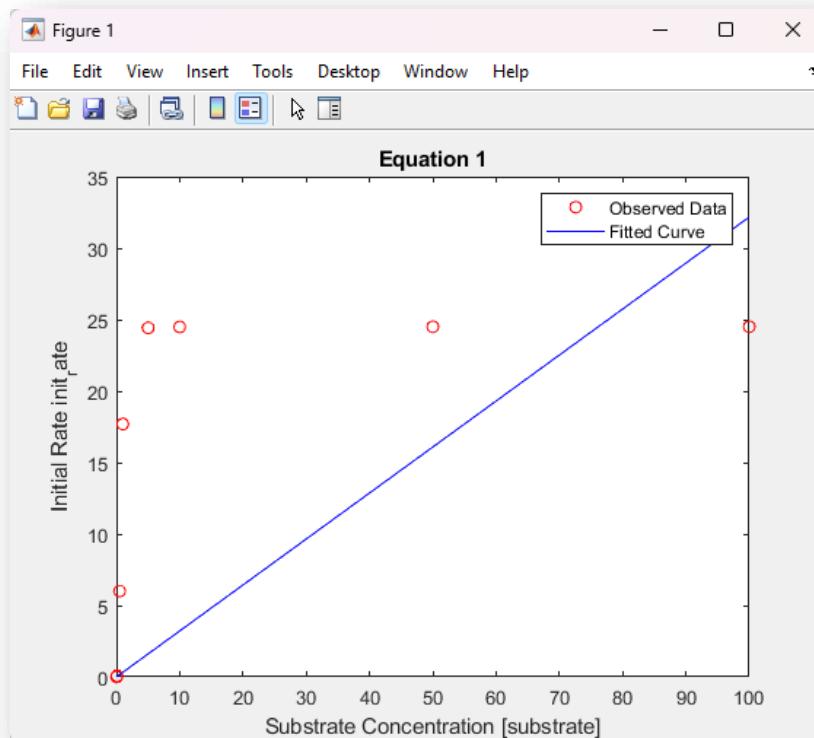
% Define the equations
equations = {
    @(params, substrate) params(1)*substrate,
    @(params, substrate) params(1)*substrate./(params(2)+substrate),
    @(params, substrate) params(1)*substrate.^2./(params(2)+substrate.^2),
    @(params, substrate) params(1)*substrate.^3./(params(2)+substrate.^3)
};

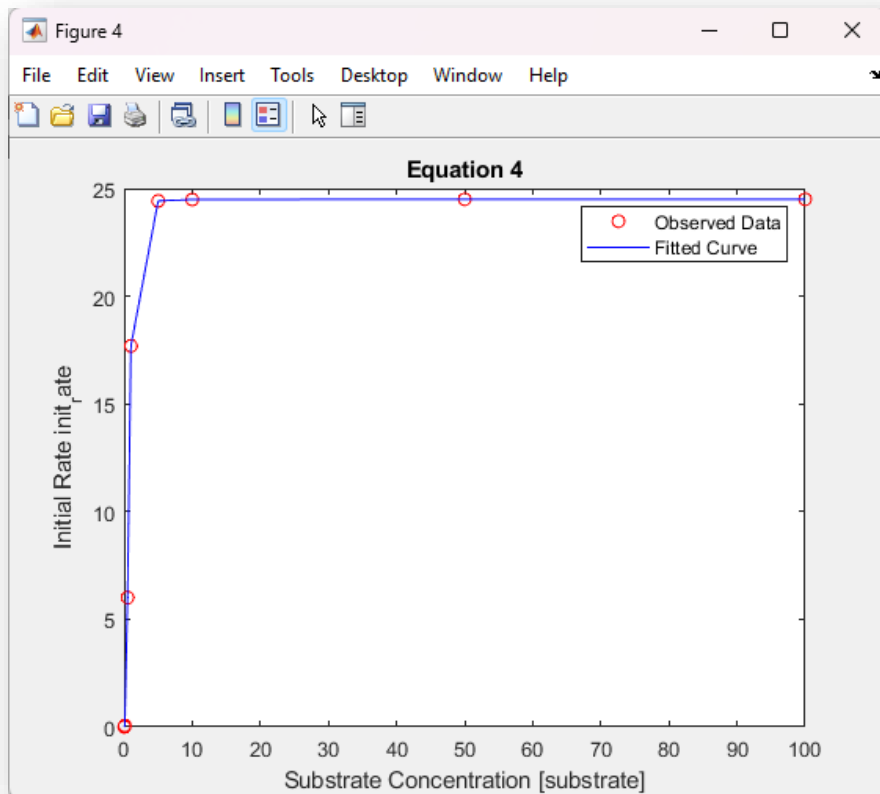
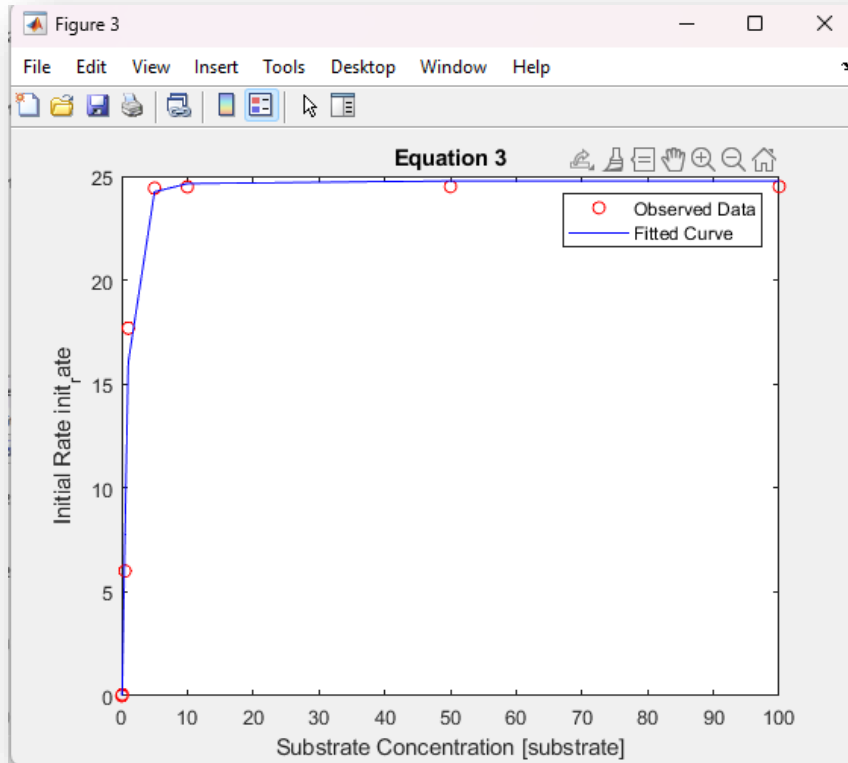
% Initialize the parameters
init_params = [1, 1];

% Perform the curve fitting for each equation
for i = 1:length(equations)
    equation = equations{i};
    params = lsqcurvefit(equation, init_params, substrate, init_rate);
    init_rate_fit = equation(params, substrate);

    % Plot the observed and fitted values
    figure;
    plot(substrate, init_rate, 'o', 'Color', 'red'); hold on;
    plot(substrate, init_rate_fit, '-', 'Color', 'blue');
    xlabel('Substrate Concentration [substrate]');
    ylabel('Initial Rate init_rate');
    title(['Equation ', num2str(i)]);
    legend('Observed Data', 'Fitted Curve');
end
```

Output:





3. (20 marks)

A drug is encapsulated in a polymer and then released slowly into the bloodstream of patients. $100\mu\text{g}$ of a drug in a controlled release capsule is injected into a patient. The drug is released at a rate of $8e^{-0.1t^2}\mu\text{g}/\text{h}$, where t is hours after injection. Use Matlab toolbox to calculate what fraction of the drug remains in the capsule after 24 h?

Code:

```
% Define the initial quantity of the drug
drug_quantity = 100; % in grams

% Define the discharge rate function
discharge_rate = @(time) 8 * exp(-0.1 * time.^2); % in grams per hour

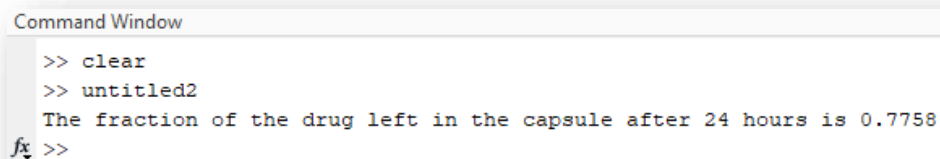
% Use numerical integration to calculate the total quantity of drug discharged
after 24 hours
total_discharged = integral(discharge_rate, 0, 24);

% Calculate the quantity of drug left in the capsule
leftover = drug_quantity - total_discharged;

% Calculate the fraction of the drug left in the capsule
fraction_leftover = leftover / drug_quantity;

% Display the result
disp(['The fraction of the drug left in the capsule after 24 hours is ',
num2str(fraction_leftover)])
```

Output:



```
Command Window

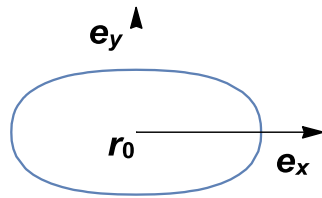
>> clear
>> untitled2
The fraction of the drug left in the capsule after 24 hours is 0.7758
fx >>
```

4. (20 marks)

A superellipse is defined by the inequality

$$\left(\frac{x}{a}\right)^n + \left(\frac{y}{b}\right)^n \leq 1,$$

where x and y are the Cartesian coordinates, and a and b are the length of long and short axes with $n > 2$ the deformation parameter. An example of a superellipse centered at \mathbf{r}_0 with $a = 2$, $b = 1$ and $n = 2.5$ is shown in the figure below, where \mathbf{e}_x and \mathbf{e}_y are the unit vectors pointing to the direction of long and short axes, respectively, and $\mathbf{e}_x \perp \mathbf{e}_y$. All vectors here are column vectors



Use Matlab toolbox numerically to calculate the area S of a superellipse for given a and b . For $a = 2$, $b = 1$, plot S as a function of $n \in [2, 10]$, and compare your result with the analytical formula

$$S = \frac{4^{1+\frac{1}{n}} ab \sqrt{\pi} \Gamma(\frac{1}{n})}{\Gamma(1+\frac{1}{n})^2},$$

where $\Gamma(\cdot)$ is the Gamma function.

Code

```
% Define the initial parameters
param_a = 2;
param_b = 1;
n_vals = 2:20;
S_vals = zeros(size(n_vals));
analytical_S_vals = zeros(size(n_vals));

% Define the grid size for the Riemann sum approximation
grid_sz = 0.01;
[x_vals, y_vals] = meshgrid(-param_a:grid_sz:param_a, -param_b:grid_sz:param_b);

for idx = 1:length(n_vals)
    n_val = n_vals(idx);

    % Calculate the area of the superellipse using a Riemann sum approximation
    z_vals = (abs(x_vals/param_a).^n_val + abs(y_vals/param_b).^n_val) <= 1;
    S_vals(idx) = sum(z_vals(:)) * grid_sz^2;

    % Calculate the area using the analytical formula
    analytical_S_vals(idx) = (4^(1+1/n_val)*param_a*param_b*sqrt(n_val)*gamma(1+1/n_val))/gamma(1/2+1/n_val);
end
```

```

% Plot the results
figure;
plot(n_vals, S_vals, 'g', 'LineWidth', 2);
hold on;
plot(n_vals, analytical_S_vals, 'm--', 'LineWidth', 2);
xlabel('n values');
ylabel('S values');
title('Superellipse Area Calculation');
legend('Numerical Calculation', 'Analytical Calculation');
grid on;

```

Output:

