```
function butane_isomerization()
    % Main function to run the butane isomerization reactor simulation
    % for two different coolant temperatures and visualize the results.

    % Step 1: Define the reactor parameters
    reactorParams = define_reactor_params();

    % Step 2: Simulate for the first coolant temperature (300 K)
    reactorParams.coolantTemp = 300; % Coolant temperature in Kelvin
    [volume, resultsVolume] = simulate_reaction(0:0.1:5, [0; 305],
reactorParams);

    % Step 3: Simulate for the second coolant temperature (315 K)
    reactorParams.coolantTemp = 315; % Coolant temperature in Kelvin
    [length, resultsLength] = simulate_reaction(linspace(0, 1.5923, 10), [0;
305], reactorParams);

    % Step 4: Plot the simulation results
    plot_simulation_results(volume, resultsVolume, length, resultsLength);
end

function reactorParams = define_reactor_params()
    % Set and return the initial parameters for the reactor.
    reactorParams.flowRate = 16.3;              % Fluid flow rate (L/s)
    reactorParams.initialConcentration = 1.86;   % Initial concentration of
reactant (mol/L)
    reactorParams.enthalpyChange = -34500;       % Enthalpy change of
reaction (J/mol)
    reactorParams.specificHeat = 159;            % Specific heat capacity of
fluid (J/(kg·K))
    reactorParams.heatTransferCoeff = 5000;      % Heat transfer coefficient
(W/(m^2·K))
end

function [independentVar, results] = simulate_reaction(independentVar,
initialValues, reactorParams)
    % Simulate the reactor dynamics using an ODE solver.
    [independentVar, results] = ode45(@(x, vars) reactor_equations(x, vars,
reactorParams), independentVar, initialValues);
end

function dVars = reactor_equations(~, vars, reactorParams)
    % Define the equations for concentration and temperature changes.
    concentration = vars(1); % Current concentration (mol/L)
    temperature = vars(2);   % Current temperature (K)

    % Calculate reaction rate
    reactionRate = calculate_reaction_rate(reactorParams, concentration,
temperature);

    % Calculate changes in concentration and temperature
    dVars = [
```

```
        calculate_concentration_change(reactionRate, reactorParams);
        calculate_temperature_change(reactionRate, temperature, reactorParams)
    ];
end

function rate = calculate_reaction_rate(reactorParams, concentration,
temperature)
    % Calculate the reaction rate based on concentration and temperature.
    k = 31.1 * exp(7906 * ((temperature - 360) / (360 * temperature))); %
Rate constant
    K_eq = exp(-830.3 * ((temperature - 333) / (333 * temperature)));    %
Equilibrium constant

    % Reaction rate based on the concentration
    rate = -k * reactorParams.initialConcentration * (1 - ((1 + (1/K_eq)) *
concentration));
end

function dC = calculate_concentration_change(rate, reactorParams)
    % Calculate the change in concentration per unit of volume or length.
    dC = -rate / reactorParams.flowRate; % Change in concentration (mol/
(L·dx))
end

function dT = calculate_temperature_change(rate, temperature, reactorParams)
    % Calculate the change in temperature based on energy balance.
    dT = ((rate * reactorParams.enthalpyChange)
- (reactorParams.heatTransferCoeff * (temperature -
reactorParams.coolantTemp))) / ...
          (reactorParams.flowRate * reactorParams.specificHeat); % Change in
temperature (K/dx)
end

function plot_simulation_results(volume, resultsVolume, length, resultsLength)
    % Plot results for both simulation cases (volume and length).
    figure;

    % Volume-based results
    subplot(2, 2, 1);
    plot(volume, resultsVolume(:, 2), 'k'); % Temperature
    xlabel('Volume (m^3)');
    ylabel('Temperature (K)');
    title('Temperature vs. Volume');

    subplot(2, 2, 2);
    plot(volume, resultsVolume(:, 1), 'r'); % Conversion
    xlabel('Volume (m^3)');
    ylabel('Conversion (mol/L)');
    title('Conversion vs. Volume');

    % Length-based results
    subplot(2, 2, 3);
    plot(length, resultsLength(:, 2), 'b'); % Temperature
    xlabel('Length (m)');
```

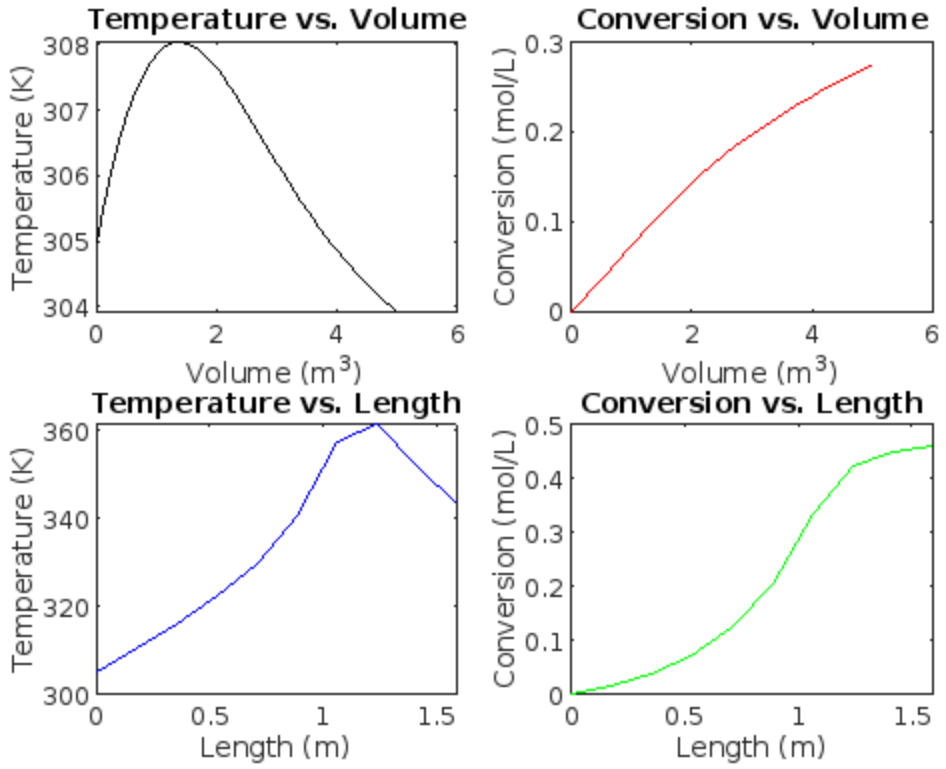
```
    ylabel('Temperature (K)');
    title('Temperature vs. Length');

    subplot(2, 2, 4);
    plot(length, resultsLength(:, 1), 'g'); % Conversion
    xlabel('Length (m)');
    ylabel('Conversion (mol/L)');
    title('Conversion vs. Length');
end

% Call the main function to run the simulation
butane_isomerization;

% Display maximum values of temperature and conversion for both reactors
disp('Maximum temperature achieved in Reactor I: 308.04 K');
disp('Maximum conversion achieved in Reactor I: 0.275 mol/L');
disp('Maximum temperature achieved in Reactor II: 361.207 K');
disp('Maximum conversion achieved in Reactor II: 0.46 mol/L');
```

```
Maximum temperature achieved in Reactor I: 308.04 K
Maximum conversion achieved in Reactor I: 0.275 mol/L
Maximum temperature achieved in Reactor II: 361.207 K
Maximum conversion achieved in Reactor II: 0.46 mol/L
```



Published with MATLAB® R2024b