Intelligent Power Source Selector

1. Wind speed modelling using Weibull distribution
   a. Generating Simulated Wind Speed data

% Example code for generating simulated wind speed data
num_samples = 365 * 24; % Number of samples for one year (assuming hourly data)
shape_parameter = 2;   % Adjust as needed
scale_parameter = 6;   % Adjust as needed
simulated_wind_speed = wblrnd(scale_parameter, shape_parameter, [num_samples, 1]);

- num_samples: This variable is set to 365 days multiplied by 24 hours per day, resulting in the total number of samples for one year. It's assumed that you want hourly data for one year.

- shape_parameter: The shape parameter (k) of the Weibull distribution. It determines the shape of the distribution. In this example, it's set to 2, but you can adjust it based on your desired distribution shape.

- scale_parameter: The scale parameter (c) of the Weibull distribution. It affects the scale or location of the distribution. It's set to 6 in this example, but you can adjust it as needed.

- simulated_wind_speed: This line generates synthetic wind speed data following a Weibull distribution with the specified shape and scale parameters. The resulting data is stored in the simulated_wind_speed variable.

   b. Defining the Wind Turbine Power Curve Function

% Define a wind turbine power curve function (example curve)
turbine_power_curve = @(wind_speed) (0.5 * wind_speed.^3)

- Here, you define a wind turbine power curve function. The function relates wind speed (in meters per second, m/s) to power output (in Watts, W). In this

example, the power curve is a simple cubic relationship, where the power is calculated as 0.5 times the cube of the wind speed (wind_speed^3). Note that this is just an example power curve; you should use the actual power curve specific to your wind turbine.

c. Calculating Hourly Wind Power

```
% Define the number of hours in a day
hours_per_day = 24;
% Initialize an array to store hourly power generation
hourly_power = zeros(hours_per_day, 1);
```

- hours_per_day: This variable defines the number of hours in a day, which is set to 24 for a typical day.
- hourly_power: An array is initialized to store the hourly power generation data. It's set to all zeros initially.

d. Calculating and Storing Hourly Power

```
% Calculate hourly power generation
for hour = 1:hours_per_day
    % Filter wind speeds for the current hour
    wind_speed_hourly = simulated_wind_speed((hour - 1) * num_samples /
hours_per_day + 1 : hour * num_samples / hours_per_day);

    % Calculate power for each wind speed using the power curve
    power_hourly = turbine_power_curve(wind_speed_hourly);

    % Sum the hourly power generation
    hourly_power(hour) = sum(power_hourly);

    % Add data to the table
    power_table.Hour(hour) = hour;
    power_table.WindPower(hour) = hourly_power(hour);
end
```

- This loop iterates through each hour of the day (from 1 to 24) and performs the following steps for each hour:

- It filters the wind speed data to include only the data points corresponding to the current hour.
- It calculates the power generated at each wind speed for that hour using the wind turbine power curve function.
- It sums up the power generated at each wind speed to obtain the total hourly power generation for that hour, which is stored in the hourly_power array.
- It also adds the hour and the corresponding hourly power value to the power_table table.

e. Displaying the table

% Display the table

disp(power_table);

- This line displays the power_table table in the MATLAB console, allowing you to see the hourly wind power data.

f. Plotting the hourly power generation and wind speed distribution

% Plot the hourly power generation

plot(1:hours_per_day, hourly_power);

xlabel('Hour of the Day');

ylabel('Wind Power (Watts)');

title('Hourly Wind Power Generation');

- This code generates a plot of the hourly wind power generation data. The x-axis represents the hour of the day (from 1 to 24), and the y-axis represents the wind power in Watts. The title and axis labels are added for clarity.

% Plot the wind speed distribution

subplot(2, 1, 2);

histogram(simulated_wind_speed, 'Normalization', 'pdf'); % 'pdf' for probability density

xlabel('Wind Speed (m/s)');

ylabel('Probability Density');

title('Wind Speed Distribution');

- These lines create a subplot with two rows and one column. In the second subplot (lower part), it plots a histogram of the simulated wind speed data. The

'Normalization' option is set to 'pdf' to display the probability density function. This subplot visualizes the distribution of wind speeds.

% Adjust the subplot layout

sgtitle('Hourly Wind Power and Wind Speed Distribution');

- This line adds a title to the overall figure, summarizing the content of both subplots.