SCHOOL OF ENGINEERING, HAROHALLI-562112

DEPARTMENT OF ELECTRONICS AND COMMUNICATION

SKILL ENHANCEMENT COURSE -I MATLAB AND SIMULINK

TOPIC:SOLAR POWER GENERATION FOR HOME USING MATLAB

NAME: Karthik H M

CLASS: ECE ‘3A’ - 3 rd Semester

USN: ENG22EC0025

SUBMITTED TO: Dr.Shirshendu Roy

Dr. Arun Ananthanarayanan

- **INTRODUCTION**

MATLAB, which stands for MATrix LABoratory, is a high-performance programming language and environment primarily used for numerical computing, data analysis, signal processing, and other scientific computing tasks.

- **Here are some key aspects of MATLAB**:

1.Matrix-Based Language: MATLAB is designed around the concept of matrices. It allows easy manipulation of arrays, which makes it well-suited for tasks involving linear algebra and matrix operations.

2.Programming Environment: MATLAB provides an interactive environment for algorithm development, data analysis, and visualization. It includes a command-line interface, a script language, and a graphical user interface (GUI).

3.Numerical Computing: MATLAB excels in numerical computing and is widely used in various scientific and engineering disciplines. It provides a rich set of built-in functions for mathematical operations, linear algebra, statistics, optimization, and more.

4.Data Visualization: MATLAB offers powerful tools for visualizing data, including 2D and 3D plotting functions. Users can create a wide range of plots and customize them to suit their needs.

5.Toolboxes: MATLAB provides additional functionality through toolboxes, which are collections of functions for specific application areas (e.g., image processing, signal processing, machine learning). These toolboxes extend MATLAB's capabilities.

6.Simulink: Simulink is a graphical programming environment for modeling, simulating, and analyzing multidomain dynamical systems. It is widely used for designing and simulating control systems, signal processing systems, and more.

7.Application Deployment: MATLAB code can be deployed to create standalone executables or integrate with other programming languages (e.g., C, C++, Java). This allows users to share their applications without requiring others to have MATLAB installed.

8.Community and Documentation: MATLAB has a large and active user community. MathWorks, the company behind MATLAB, provides comprehensive documentation, tutorials, and online forums to support users.

9.Scripting and Automation: MATLAB supports scripting, allowing users to write scripts and functions to automate tasks. This is useful for developing and testing algorithms.

10.Supported Platforms: MATLAB is available for Windows, macOS, and Linux. It is widely used in academia, industry, and research.

11.Programming Languages: While MATLAB has its own programming language, it also supports integration with other languages such as C, C++, and Java.

*It's worth noting that MATLAB is a commercial product developed by MathWorks, and a license is required for its use. There are also open-source alternatives for numerical computing, such as Octave and Scilab, which offer similar functionality to MATLAB.

- **INFORMATION-**

- PV Home On-Grid Solar System

This example shows the operation of a photovoltaic (PV) residential system connected to the electrical grid.

- PV Strings

The PV strings section implements a home installation of six PV array blocks in series that can produce 2400 W of power at a solar irradiance of 1000 W/m2. In the Advanced tab of the PV blocks, the robust discrete model method is selected, and a fixed operating temperature is set to 25 degrees C.

- Two-Stage Converter

The power produced by the PV strings is fed to the house and utility grid using a two-stage converter: a boost DC-DC converter and a single-phase DC-AC full-bridge converter. Both converters are PWM-controlled with a switching frequency of 20 kHz. They are using the Switching Function method with PWM pulse averaging, allowing a simulation sample time of 5 microseconds and good accuracy on harmonics generated.

- <u>Control Systems</u>

The MPPT Controller: The Maximum Power Point Tracking (MPPT) controller is based on the Perturb and Observe technique with scanning capability. The MPPT system automatically varies the duty cycle of the boost converter to generate the required voltage across the PV string in order to extract maximum power. Under partial shading condition, a duty cycle is initiated to find the global maximum power point (GMPP) among various local maximum power points (LMPP).

- <u>The Inverter Controller</u>

  The inverter control maintain the DC link voltage at 400 V while keeping a unity power factor. The controller uses a voltage regulator outer loop and a fast inner loop current regulator to generate the appropriate reference voltage (Vref) for the PWM generator controlling the full-bridge converter.

- <u>Load & Utility Grid</u>

The grid is modeled using a typical pole-mounted transformer and an ideal AC source of 14.4 kVrms. The transformer 240 volt secondary winding is center-tapped and the central neutral wire is grounded. The inverter, the 2500 W residential load as well as the neighbors' load are connected to the 240V secondary winding.

- ## Simulation

Run the simulation and observe the resulting signals on the various scopes.

(1) At 0.25s, with a solar irradiance of 1000 W/m2 on all PV modules, steady state is reached. The solar system generates 2400 Watts and the DC link is maintained at 400 volts with a small 120-Hz ripple due to the single-phase power extracted from the PV string. The Utility meter indicates that the system takes almost no power from the grid to supply the home total load.

(2) At 0.3s, a partial shading condition is created by reducing the irradiance on some PV modules. When steady-state is reached at 0.35s, the MPPT controller has set the boost duty cycle at 0.44, generating a PV string voltage of 225 V. With this voltage, 920 W is extracted from the PV string. As you can see on the PV curve characteristic, the system is operating at a local maximum power point but not at the global maximum power point.

(3) At 0.4s, a duty cycle scan of 0.25 seconds is performed by the MPPT controller to find the GMPP point.

(4) At 0.7s, the MPPT controller has set the boost duty cycle at 0.58 generating a PV string voltage of 168 V. With this voltage, 1364 W is extracted from the PV string which is the GMPP value. The Utility meter indicates that it takes now around 1100 W (2500 W residential load - 1364 W supplied by PV) from the grid to supply the home total load.

CODE:

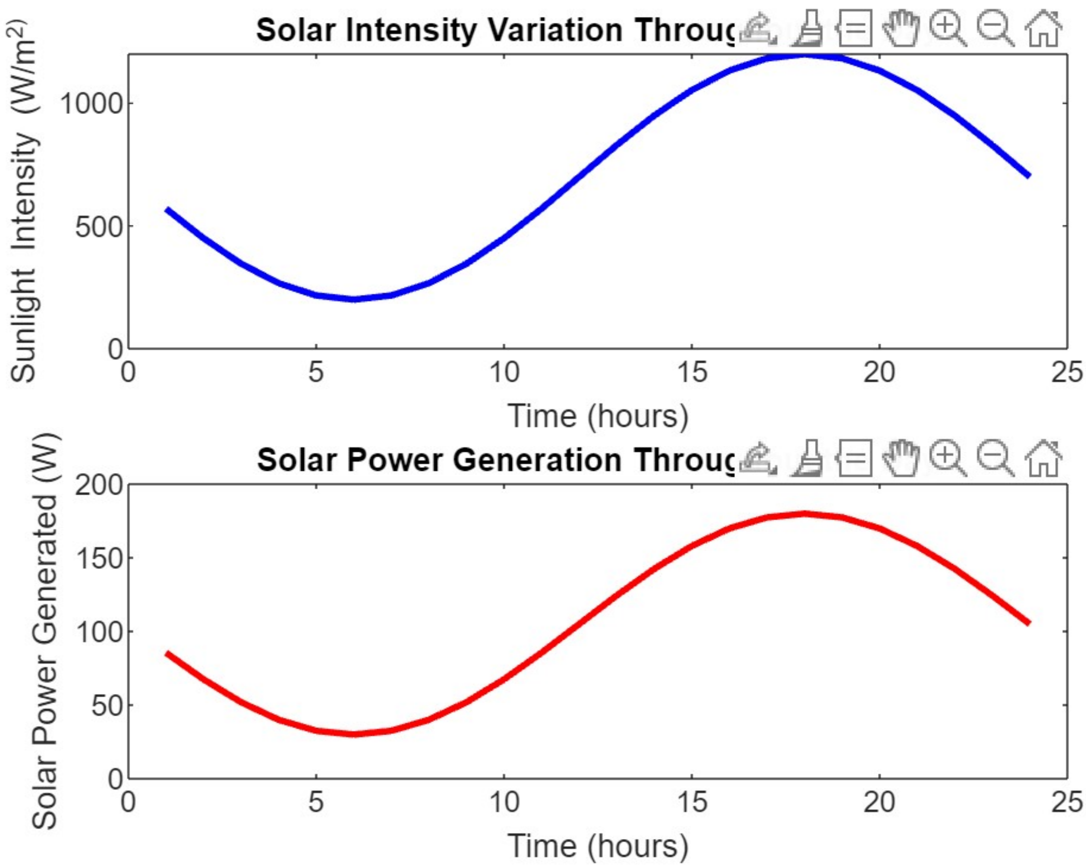
```
% Solar Power Generation Simulation in MATLAB
% Constants
solarPanelArea = 1; % in square meters
solarPanelEfficiency = 0.15; % efficiency of the solar panel
solarConstant = 1361; % solar constant in W/m^2 (average value)
ambientTemperature = 25; % in Celsius
% Time settings
time = 1:24; % 24 hours
sunlightIntensity = zeros(size(time));
% Simulate sunlight intensity (you can replace this with actual
data)
% For simplicity, this example assumes a sinusoidal variation
throughout the day
sunlightIntensity = 500 * sin(2 * pi * (time - 12) / 24) + 700;
% Simulate temperature variation throughout the
day (you can replace this with actual data)
% For simplicity, this example assumes a linear
variation
temperature = ambientTemperature + (time - 12);
```

```
% Calculate solar power generation
 solarPower = solarPanelEfficiency * solarPanelArea *
 sunlightIntensity;
% Plot results
 figure;
 subplot(2, 1, 1);
 plot(time, sunlightIntensity, 'b', 'LineWidth', 2);
 xlabel('Time (hours)');
 ylabel('Sunlight Intensity (W/m^2)');
 title('Solar Intensity Variation Throughout the Day');
subplot(2, 1, 2);
 plot(time, solarPower, 'r', 'LineWidth', 2);
 xlabel('Time (hours)');
 ylabel('Solar Power Generated (W)');
 title('Solar Power Generation Throughout the Day');
% Display total energy generated in a day
 totalEnergy = trapz(time, solarPower);
```

disp(['Total energy generated in a day: ', num2str(totalEnergy), ' Wh']);

- **OUTPUT:**

Total energy generated in a day: 2424.7057 Wh



This MATLAB code simulates solar power generation throughout the day based on sunlight intensity and temperature. Let's go through the code step by step:

1. Constants:

- `solarPanelArea`: Area of the solar panel in square meters.
- `solarPanelEfficiency`: Efficiency of the solar panel (fractional value).
- `solarConstant`: Solar constant in W/m^2, representing the average solar irradiance outside Earth's atmosphere.
- `ambientTemperature`: Ambient temperature in Celsius.

2. Time settings:

- `time`: An array representing 24 hours.
- `sunlightIntensity`: An array initialized to zeros to store simulated sunlight intensity values.

3. Simulate sunlight intensity:

- The code simulates sunlight intensity throughout the day. In this example, it assumes a sinusoidal variation with a peak at noon (12:00 PM).

- `sunlightIntensity` is updated using the formula: `500 * sin(2 * pi * (time - 12) / 24) + 700;`.

  - The term `(time - 12) / 24` normalizes the time to a range of [-0.5, 0.5].

  - The sine function creates a sinusoidal variation.

  - The coefficients scale and shift the function to achieve the desired range and peak intensity.

4. Simulate temperature variation:

- The code simulates the temperature variation throughout the day. In this example, it assumes a linear variation.

- `temperature` is calculated as `ambientTemperature + (time-12);`.

5. Calculate solar power generation:

- `solarPower` is calculated using the formula: `solarPanelEfficiency * solarPanelArea * sunlightIntensity;`.

  - It represents the solar power generated by the solar panel at each hour.

6. Plotting:

- Two subplots are created to visualize the variation in sunlight intensity and solar power throughout the day.

- The first subplot shows the variation in sunlight intensity, and the second subplot shows the corresponding solar power generation.

7. Display total energy generated:

- The code calculates the total energy generated in a day using the trapezoidal rule (`trapz` function) to integrate the solar power values over time.

- The result is displayed using `disp`.

This code is a simple simulation and can be enhanced by incorporating real data for sunlight intensity and temperature. Additionally, it assumes idealized conditions and doesn't consider factors like shading, panel degradation, and other environmental influences that affect actual solar power generation.

QUOTE:

"MATLAB IS THE HEART OF ENGINEERING, THE SOUL OF SCIENCE ,AND THE MAGIC BEHIND DATA ANALYSIS."

THANK YOU