REPUBLIQUE ALGERIENNE DEMOCRATIQUE ET POPULAIRE
الجمهورية الجزائرية الديمقراطية الشعبية

MINISTERE DE L'ENSEIGNEMENT SUPERIEUR ET DE LA RECHERCHE SCIENTIFIQUE
وزارة التعليم العالي والبحث العلمي

ECOLE SUPERIEURE DES SCIENCES APPLIQUEES
--T L E M C E N--
المدرسة العليا في العلوم التطبيقية
-تلمسان-

Ecole Supérieure en sciences appliquées de Tlemcen

Département deuxième cycle

Renewable Energy Systems Modeling and Simulation

Dr. Abdelkarim CHEMIDI **2025/2026** **4ème année Electrotechnique**

Preamble

This document serves as a course companion for Renewable Energy Systems Modeling and Simulation. The content is divided into four chapters, each addressing specific aspects of this field. Chapter 1 introduces the fundamental concepts and principles underlying renewable energy systems modeling and simulation. Chapter 2 present an overview about Matlab/Simulink environment. Chapters 3 and 4 delve deeper into the modeling of photovoltaic and wind energy conversion systems, respectively.

Learning Objectives

Upon completion of this course, students will be able to:

- Develop proficiency in using MATLAB/Simulink for modeling and simulation tasks.
- Apply various modeling techniques to complex electrical systems.
- Model and simulate real-world electrical systems with accuracy.

Contents

Introduction

The rapid depletion of fossil fuels, coupled with the environmental consequences of greenhouse gas emissions, has accelerated the global transition toward renewable energy sources. Among the various alternatives, wind and solar photovoltaic (PV) energy have emerged as the most promising technologies due to their abundance, maturity, and cost competitiveness. However, the intrinsic intermittency of wind speed and solar irradiance poses significant challenges to the reliability and stability of standalone renewable energy systems.

To overcome these limitations, hybrid renewable energy systems have been proposed, combining complementary energy sources in order to enhance power availability and reduce dependence on a single resource. In particular, hybrid systems based on wind turbines and photovoltaic arrays represent a highly attractive solution. While solar energy is abundant during the day, wind resources may prevail at night or during cloudy conditions, leading to a more balanced and continuous energy supply when both resources are integrated.

The modeling of such systems requires a multi-domain approach, as it involves the representation of aerodynamics, semiconductor physics, electrical machines, and power electronics. The wind turbine must be modeled through aerodynamic power capture and mechanical-to-electrical conversion via induction or synchronous generators. The PV array, in turn, is represented using diode-based models that describe its nonlinear current–voltage (I–V) characteristics. Power electronic converters play a central role in interfacing both sources with the DC bus and ultimately the load or utility grid, ensuring maximum power extraction and system stability.

Simulation tools such as MATLAB/Simulink with Simscape Power Systems provide a powerful environment for analyzing the dynamic performance of hybrid wind–PV systems. Through simulation, researchers and engineers can evaluate the effects of varying irradiance, wind speed fluctuations, and load demands, while testing different control strategies such as Maximum Power Point Tracking (MPPT), pitch control, and supervisory energy management algorithms.

The study of hybrid wind–PV systems is therefore not only of academic importance but also of significant practical relevance, as such systems form the backbone of modern microgrids and distributed generation networks. Their modeling and simulation constitute essential steps toward the design, optimization, and real-world deployment of reliable, efficient, and environmentally sustainable energy solutions.

Chapter 1:
Introduction and
Hands-on Experience

Purpose of the Chapter

- This chapter serves as an introduction to the world of modeling and simulation. The intermediate objectives for this chapter are:
- To familiarize students with the fundamental concepts underlying modeling and simulation.
- To provide a hands-on experience with MATLAB, one of the most widely used software tools in engineering and scientific applications.

Specific Objectives:

Upon completion of this chapter, students will be able to:

- Understand the basic principles of modeling and simulation.
- Familiarize themselves with Simulink, a graphical interface for modeling systems.
- Explore the capabilities of SimPowerSystems, a library specifically designed for electrical power system modeling.

I. Introduction

The increasing complexity of modern systems has led to the need for advanced modeling and simulation tools. As technology continues to evolve at a rapid pace, it is essential to have software that can simulate the behavior of complex electrical systems.

Modeling and simulating these systems involves considering their overall structure, control mechanisms, and environmental interactions. This approach enables engineers to manage exchanges between subsystems and obtain results that take into account most or even all physical phenomena governing the system's operation.By mastering simulation tools and software, engineers can develop new skills that are essential for designing efficient and reliable renewable energy systems.

In this course, we will delve deeper into the world of modeling and simulation in the context of electrical engineering. We will explore various techniques and methods for simulating complex systems using MATLAB/Simulink and SimPowerSystems, with a focus on renewable energy applications.

II. Why Renewable Energy?

The world's energy needs are growing exponentially, with global demand increasing by a staggering 30% in the past two decades. This surge in consumption has placed immense pressure on traditional fossil fuel-based energy systems.

As it stands today, nearly all (around 90%) of our energy comes from non-renewable sources that not only contribute to climate change and air pollution but also deplete finite resources at an alarming rate. The consequences are dire: increased greenhouse gas emissions, rising temperatures, and the exhaustion of fossil fuel reserves within a few generations. It is against this backdrop that renewable energy has emerged as a vital solution for meeting our future energy needs sustainably. By harnessing the power of sun, wind, water, and geothermal heat, we can reduce our reliance on finite resources and create cleaner, more sustainable energy options for all.

III. Modeling and Simulation Engineer

The modeling and simulation engineer is an expert who applies advanced mathematical models and computational tools to simulate the behavior of complex systems. Their primary goal is to analyze and predict the performance of products or machines under various operating conditions.

By using specialized software, such as MATLAB/Simulink or ANSYS, they create virtual replicas of real-world systems. These digital models enable them to:

- Identify performance bottlenecks: By simulating different scenarios, they can pinpoint areas where a system's efficiency is compromised.
- Optimize design parameters: Through iterative simulations, engineers can refine the design of products or machines to meet specific requirements or performance goals.
- Reduce testing and validation time: Modeling and simulation allow for rapid prototyping and virtual testing, significantly shortening the development cycle.

III.1. Missions:

As a modeling and simulation engineer, your primary missions include:

- **Product design and development**: Utilize software tools to conceptualize, model, and simulate new products or systems.
- **Computational experimentation**: Conduct virtual experiments using computational models to test various parameters (e.g., stress, strain, thermal analysis) on a given system, device or product.
- **Performance benchmarking**: Evaluate the performance of modeled systems or products against set standards or targets.
- **Design optimization**: Refine and optimize the shape and structure of products to enhance their functionality, efficiency, and overall performance.

Additionally, as an expert in modeling and simulation:

- **Software development**: Create new software tools for modeling, analysis, and simulation purposes.
- **Collaboration**: Work closely with cross-functional teams (e.g., design engineering, manufacturing) to ensure that modeled products meet the necessary requirements and standards.

By executing these missions effectively, the engineer contribute significantly to product innovation and efficiency improvement.

IV. Modeling Tools

Simulation software empowers product development by minimizing prototyping phases, enabling designers, engineers, and analysts to:

- **Rapidly iterate designs**: Test and refine product concepts quickly using virtual prototypes.
- **Analyze complex systems**: Simulate the behavior of intricate assemblies or mechanisms with ease.

IV.1.SimSolid

With SimSolid, you can:

- **Simulate complex assemblies**: Analyze complete CAD assemblies in minutes, enabling faster design iterations and improved product performance.
- **Collaborate more effectively**: Share simulation results with cross-functional teams, fostering better communication and decision-making.

IV.2. Wolfram Mathematica

Wolfram Mathematica is a powerful computational platform that offers an extensive range of capabilities to tackle complex problems in various fields. Its suite of tools includes:

- **Image processing**: Enhance and analyze images using advanced algorithms and techniques.
- **Geometry**: Perform geometric calculations, visualize 3D models, and create interactive simulations.
- **Visualization**: Create stunning visualizations, animations, and interactive dashboards to communicate complex data insights.
- **Machine learning**: Develop predictive models using machine learning algorithms, including neural networks and decision trees.
- **Data mining**: Explore large datasets, identify patterns, and uncover hidden trends.

By harnessing Wolfram Mathematica's capabilities, you can:

- **Accelerate research**: Streamline data analysis, simulation, and visualization tasks to accelerate discovery.
- **Improve decision-making**: Provide stakeholders with accurate insights and recommendations based on thorough analysis.

IV.3. SolidWorks

SolidWorks is a powerful simulation solution that helps businesses evaluate various aspects of product design. Its capabilities include:

- **Structural analysis**: Analyze the stress and strain on materials under different loads to ensure optimal performance.
- **Dynamic loading**: Simulate real-world scenarios, such as vibrations or impacts, to predict how products will behave in extreme conditions.
- **Composite materials**: Model and analyze the behavior of composite materials, including their mechanical properties and response to various stimuli.

IV.4. Matlab

Matlab is a comprehensive environment for programming and data analysis that enables users to:

- **Develop algorithms**: Create custom algorithms for specific tasks, such as image processing or machine learning.
- **Analyze data**: Work with large datasets, identifying trends, patterns, and insights using various statistical and numerical techniques.
- **Visualize results**: Present findings in a clear and concise manner through interactive visualizations.
- **Perform numerical calculations**: Run complex simulations and models to predict outcomes or optimize processes.

IV.5.HOMER

The Hybrid Optimization Model for Electric Renewables (HOMER) is a powerful tool that helps optimize the configuration of renewable energy systems. Developed by the National Renewable Energy Laboratory, HOMER uses simulation to identify the most efficient and cost-effective way to integrate solar, wind, hydro, or other forms of renewable energy into an existing grid.

Key features of HOMER include:

- **Simulation-based optimization:** HOMER uses a sophisticated algorithm to simulate different configurations and scenarios, allowing users to visualize the potential outcomes.
- **Hourly time step**: The simulator takes into account hourly variations in weather conditions, load demand, and other factors that can impact renewable energy production.
- **Environmental data integration:** HOMER incorporates real-world environmental data, such as solar radiation, wind speed, and temperature, to ensure accurate predictions.

However:

- **Limited user control**: Unlike some other optimization tools, HOMER does not allow users to select specific components for their system. Instead, it uses a pre-defined library of devices and models to simulate different configurations.
- **Assumes grid-connected scenario**: HOMER is designed primarily for grid-connected scenarios, which may not be suitable for all applications.

Despite these limitations:

- **Provides valuable insights**: By running various simulations and analyzing the results, users can gain a deeper understanding of how their renewable energy system will perform in different conditions.
- **Can help reduce costs**: Optimizing the configuration of renewable energy systems using HOMER can lead to cost savings by minimizing excess capacity, reducing wear and tear on components, and improving overall efficiency.

IV.6. TRNSYS

The Transient Energy System Simulation Program is a software for simulating energy systems developed at the University of Wisconsin and the University of Colorado in 1975. Initially designed to simulate thermal systems, TRNSYS has evolved into a hybrid system simulator, including photovoltaic, solar thermal and other types of renewable energy systems.

The standard library of TRNSYS includes many commonly used components in thermal and electrical energy systems. The simulation is performed with high precision, allowing for accurate visualization of the results through graphical plots.

Key features of TRNSYS include:

- **Transient simulations**: TRNSYS can simulate transient behavior in energy systems, including dynamic responses to changes in load or operating conditions.
- **Hybrid system modeling**: The software can model complex hybrid systems that integrate multiple energy sources and storage technologies.
- **High precision simulation**: TRNSYS performs simulations with high accuracy, allowing for precise visualization of results through graphical plots.

However:

- **Limited optimization capabilities**: As mentioned earlier, TRNSYS does not allow for optimization of system parameters or design.
- **Steep learning curve**: The software has a complex interface and requires significant expertise to use effectively.

Despite these limitations:

- **Valuable tool for detailed analysis**: TRNSYS remains a valuable tool for performing detailed analyses of energy systems, including transient behavior and hybrid system interactions.
- **Can be used in combination with other tools**: Users can combine TRNSYS with other software or techniques, such as optimization algorithms or machine learning models, to gain further insights into energy system performance.

In this course, we are focusing on Matlab as a tool that provides an environment for both simulation and verification/validation of hardware. These functions are combined into a single environment, making it an ideal platform for system design and development. Moreover, Matlab contains a vast library of tools and functions specifically designed for the modeling and simulation of renewable energy systems.

V. Modeling definition

To model a system requires a thorough understanding of its fundamental operating principles.The modeling process involves observing and representing the behavior of a phenomenon or process in an approximate, yet sufficiently accurate, manner. This representation focuses on capturing the most significant characteristics relevant to a given context or application.

There are two primary approaches to modeling:

- **Black Box Modeling**: This approach relies solely on observing how a system responds to known input signals without considering its internal mechanisms or workings. It is based on empirical data and observations, rather than any specific knowledge of the underlying laws or principles.
- **White Box Modeling (Physical Modeling)**: In contrast, this approach involves creating models that accurately represent the underlying laws and principles governing the behavior of a system. This includes using mathematical equations, physical laws, and other relevant theories to describe how the system functions.

While black box modeling is useful for predicting general trends or behaviors, white box modeling provides more detailed insights into the underlying mechanisms driving those behaviors.

V.1. Effective Utilization of Modeling and Simulation Tools

The effective utilization of modeling and simulation software enables the implementation of various approaches to model and obtain rapid, valid results. By leveraging these tools, users can explore different configurations, parameters, and scenarios without incurring the costs or risks associated with physical prototypes.

These software solutions facilitate:

- **Risk reduction**: by simulating system behavior prior to actual deployment.
- **Design optimization**: by adjusting and testing various configurations to achieve optimal performance.
- **Process acceleration**: by allowing rapid exploration of different hypotheses and selection of the most promising options.

Through these tools, users can obtain:

- **Accurate and reliable results**: thanks to validated simulations that accurately reproduce real-world system behavior.
- **Flexibility**: in modeling and simulating various levels of complexity, from simple processes to multi-dimensional systems.

- **Improved decision-making**: By providing a deeper understanding of system, dynamics and the potential consequences of different design choices.
- **Increased efficiency**: by streamlining the development process and reducing the need for costly physical prototypes or iterative testing.

VI. Modeling Approaches

There are various approaches to creating numerical models that replicate the behavior of complex systems. Each method has its own advantages and disadvantages, and can be adapted to different contexts.

VI.1. White Box Modeling

This approach involves writing mathematical equations that describe the system's behavior. Subsequently, a numerical tool is used to solve these equations, yielding an initial approximation of the model structure. However, in most cases, the parameters characterizing the system's behavior are not fully known. To overcome this limitation, some parameters must be neglected or estimated during simulation. The resulting output should then be compared with experimental data to optimize and validate the model.

VI.2. Multi-Physics Modeling

This approach involves assembling pre-modeled components from a software library, eliminating the need for writing equations and yielding significant time savings. Libraries of pre-defined components enable modeling across various physical domains (mechanics, electricity, hydraulics, thermodynamics, etc.). It becomes relatively simple to link different domains together to obtain a comprehensive model of an entire system.

VI.3. Gray Box Modeling

This approach complements white box or multi-physical models by estimating unknown parameters through comparison between the model's response and actual system behavior. Automated algorithms can facilitate this process, assigning values to previously unknown parameters in order to eliminate discrepancies between simulated and real-world performance.

VI.4. Black Box Modeling (Identification)

This approach involves associating a mathematical model with experimentally measured behavior. Numerical tools automate this task by proposing complex identification models.

This method produces only a global behavioral model of the system, without concern for separate parameter influences on performance.

VII. Example

Modeling a DC Motor involves establishing the relationship between the voltage across the motor's inductor and its rotational speed. Specifically, as the input voltage increases, so does the motor's speed.

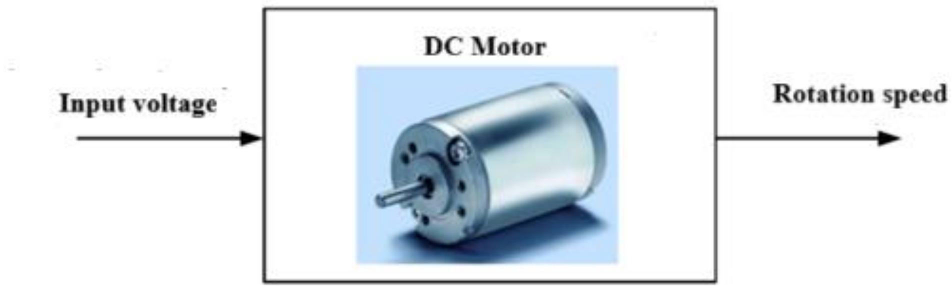


Figure I.1: Input/ output's for DC motor

VII.1. White Box Modeling

The induction of a DC motor can be represented by the following figure:

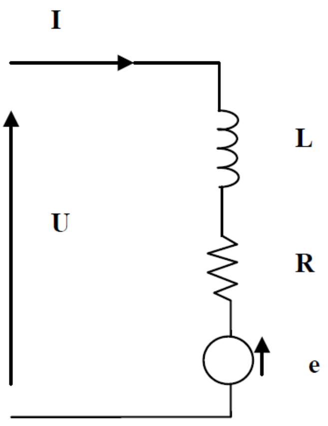


Figure I.2: Induction of a DC motor

R: Internal resistance of the motor's winding.

L: Internal inductance of the motor's winding.

e: Back-EMF (Electromotive Force) generated by the motor.

Electromechanical Equations of the DC motor

Electrical Equations

The mesh law allows us to write:

$$u(t) = R.i(t) + L\frac{di}{dt} + e(t)$$

With $e = K_e.\Omega$

Mechanical Equations

The mechanical fundamental equation of the shaft written as:

$$J\frac{d\Omega}{dt} + f\Omega = C_{em} - C_r$$

With $C_{em} = K_m.I$

Among the parameters that characterize the behavior of the motor, some are well-known and others remain unknown. The inductance L and resistance R of the winding are provided in the technical documentation of the motor, as are the coefficients K_e and K_t. However, the viscous friction coefficient f is not necessarily known, particularly if the motor is subsequently coupled to a mechanical system. In this case, it may be necessary either to neglect this parameter or estimate its value in order to obtain valid results with the model.

Modeling Using Laplace Transform

After applying the Laplace transform to these equations, it is straightforward to obtain a numerical model in the form of a block diagram that can then be modeled and solved using numerical simulation software.

$$u(s) = R.i(s) + sLi(s) + K_e\Omega(s)$$

Then:

$$I(s) = \frac{U(s) - K_e\Omega(s)}{Ls + R}$$

Laplace transformation of the mechanical equation:

$$Js\Omega(s) + f\Omega(s) = K_m I(s) - C_r$$

Then:

$$\Omega(s) = \frac{K_m I(s) - C_r}{Js + f}$$

From the previous equations, it is possible to construct the functional block diagram of the DC machine.

This diagram represents the dynamic behavior of the motor by clearly illustrating the interaction between the electrical and mechanical subsystems. It serves as the foundation for numerical modeling using simulation software such as MATLAB/Simulink, allowing for the analysis of system responses under various input conditions.

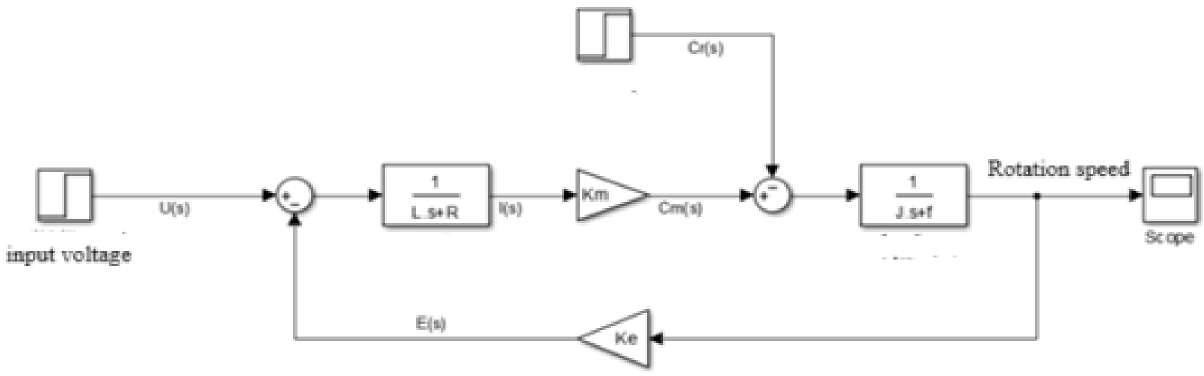


Figure I.3: Block diagram-based modeling of the DC motor

At this stage, the physical system can be substituted with its mathematical model, as shown in the following block diagram. This allows for simulation and analysis under various operating conditions.

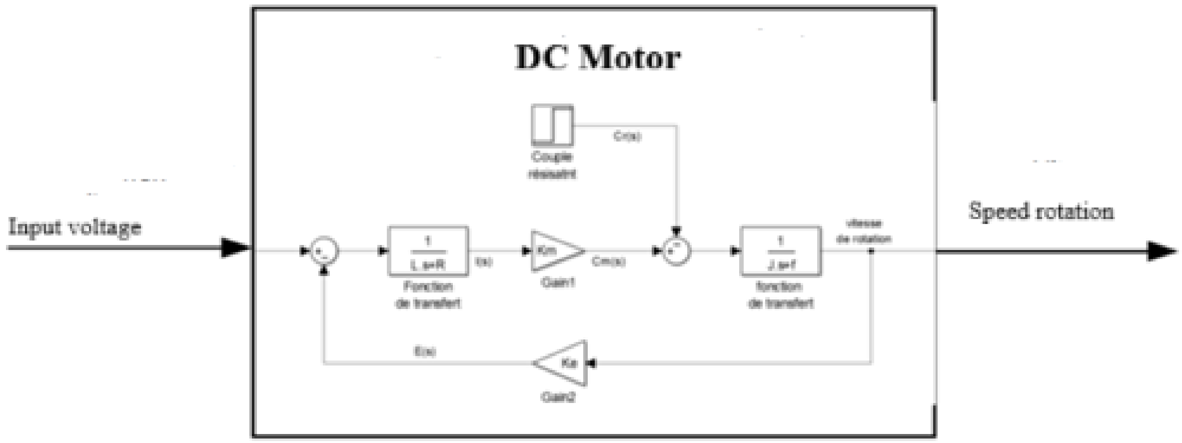


Figure I.4 : White box modeling of DC motor

Multi-Physics Modeling

Multi-physics modeling adopts an acausal approach, allowing systems to be constructed by assembling components from predefined libraries. The physical behavior of each component is inherently managed by the software, eliminating the need to manually derive and implement the differential equations governing the system's dynamics. This approach significantly simplifies the modeling process, especially for complex systems involving multiple physical domains.

The multi-physics model of the DC motor is illustrated in the following figure.

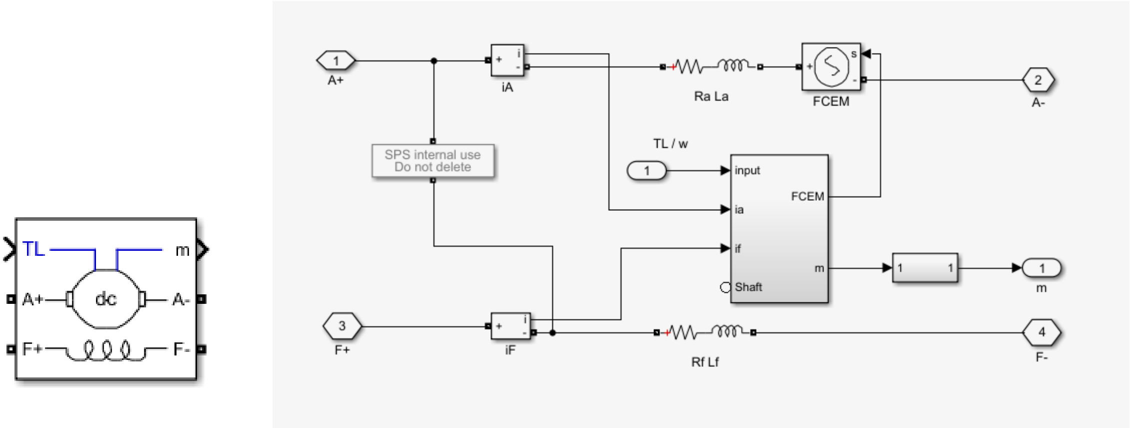


Figure I.5 :Multi-physics modeling of the DC motor using component assembly

At this point, the physical system may be substituted by its representative model, to facilitate simulation and analysis.

Model Simulation

Simulation and numerical resolution of models using simulation software require all parameter values to be defined. In our example, all parameters are known except for the coefficient *f*, which represents friction. As a first approximation, this parameter will be assumed to be zero, effectively neglecting the influence of friction in the system.

Regardless of the modeling approach—whether based on a white-box or multi-physics methodology—the numerical simulation results are identical (Figure 6). In both cases, the same physical phenomena are taken into account; the primary difference lies in the method used to construct the model and the numerical resolution techniques employed.

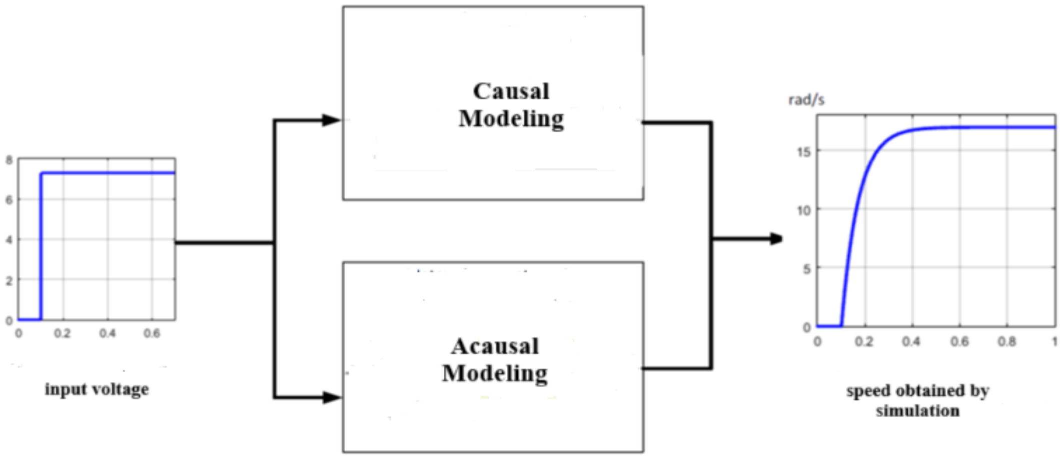


Figure I.6 : Simulation results of DC motor

The simulation results provide a first approximation of the motor's rotational speed as a function of time.However, in general, this approach is often insufficient. The only reliable way to assess the model's validity is to compare the simulated performance with actual measurements obtained from the real system. By applying the same voltage across the terminals of the motor armature as in the simulation, the real-time evolution of the motor's rotational speed can be recorded, as illustrated in Figure I.7.

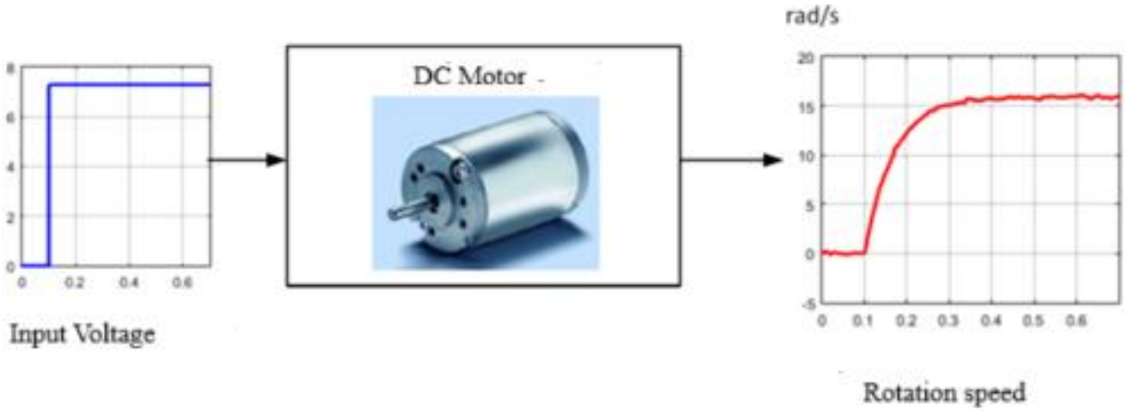


Figure I.7 :Measurement of the evolution of motor speed over time

Comparison of Simulated and Measured Performance

Figure I.8 presents a comparison between the simulated and experimentally measured performance.

At this stage, it is common to observe discrepancies between the model and the real system behavior. These differences must be carefully evaluated to identify possible ways to improve the model—either by incorporating additional physical effects or by adjusting specific parameter values to better reflect the system's dynamics.In this example, the observed mismatch indicates that the model is not yet validated. The main source of error lies in the omission of frictional effects, which significantly influence the motor's behavior. To address this, the unknown parameter—here, the friction coefficient *f*—must be estimated.

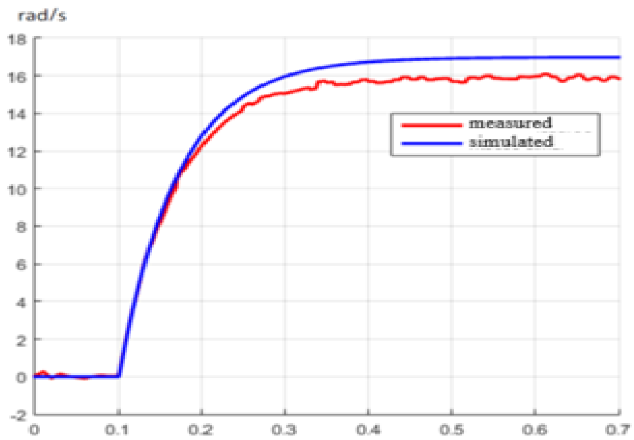


Figure I.8 :Assessment of the gap between simulated and measured performance

A **gray-box modeling approach** becomes particularly useful in such cases. It allows for refining the existing model by combining physical insights with parameter estimation techniques, ultimately leading to a more accurate and validated representation of the real system.

Gray-Box Modeling

The principle of gray-box modeling is to estimate the unknown parameters of a model by comparing simulated performance with measured performance. A built-in algorithm within the simulation software performs this task by analyzing discrepancies between the model and the real system, and then proposing plausible values for the unknown parameters that minimize or eliminate these discrepancies.

To carry out this estimation, the same input signal is applied to both the model and the real system. The software then adjusts the unknown parameters to align the simulated response with the measured one. The objective is for the model to reach the target response. This optimization process can be applied using either a causal (equation-based) or an acausal (component-based) model.

The figure below illustrates this process:

- The **target response** (in red) represents the behavior of the real system.
- The **initial model response** (in blue) reflects the system behavior before parameter estimation.
- The **optimized model response** (in green) shows the system behavior after parameter adjustment.

This method offers a clear advantage: it yields a validated model whose behavior closely mirrors that of the actual system. The discrepancies between measured and simulated performance are effectively minimized, resulting in a reliable and accurate representation of the system dynamics.

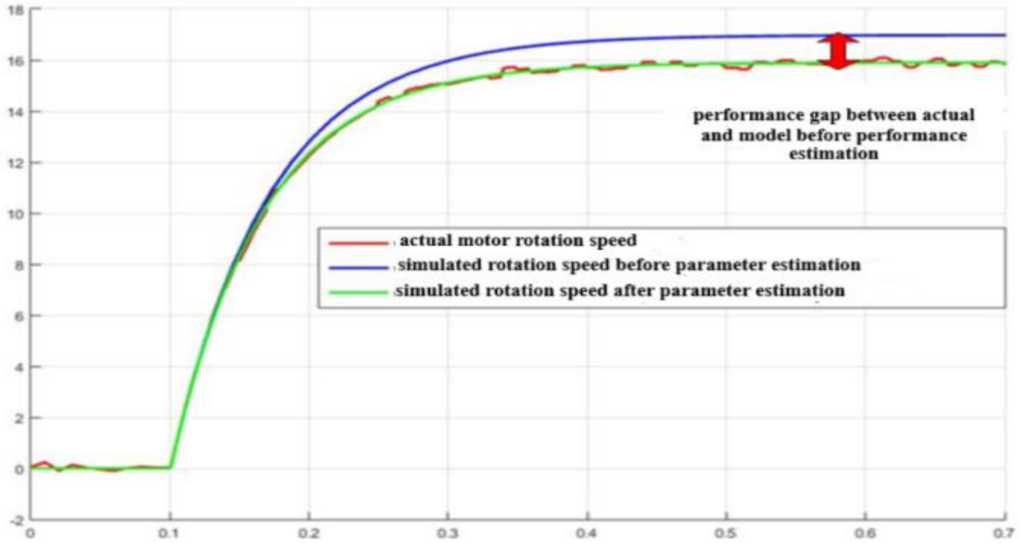


Figure I.9 :The model responses before and after estimation of the unknown parameters

At the end of this phase, the project benefits from validated models.

This allows the physical hardware (in this case, the motor) to be replaced by its digital model. The validated model can then be used for further analysis, control design, or integration within larger simulation frameworks, offering a reliable and cost-effective alternative to physical experimentation.

VIII. Conclusion

In this first chapter, we have laid the foundations of modeling and simulation, clarifying their definitions, objectives, and scope of application. By distinguishing between models as abstract representations of reality and simulation as the dynamic experimentation of these models, we have established the conceptual framework necessary for deeper study.

This introduction highlights the importance of modeling as a tool for understanding complex systems and simulation as a means of predicting behavior, testing hypotheses, and supporting decision-making. The concepts presented here will serve as the basis for the subsequent chapters, where we will move from theoretical principles to practical methods and applications.

Chapter 2: Introduction and Overview of Modeling Tools

Purpose of the Chapter

This chapter provides an overview of the main modeling and simulation tools used in the field of renewable energy systems. It aims to help students understand the roles, functionalities, and application domains of different software environments. By exploring both general-purpose and domain-specific tools, students will be able to make informed decisions when selecting appropriate platforms for modeling, simulation, and system analysis. This foundational knowledge is essential before diving into the detailed modeling of photovoltaic and wind energy systems in later chapters.

Intermediate Objectives

By the end of this chapter, students will:

- Understand the importance of choosing the right modeling tool for a specific application.
- Distinguish between different categories of modeling tools
- Recognize the advantages and limitations of various modeling environments commonly used in renewable energy engineering.
- Become familiar with acausal and causal modeling paradigms, as well as component-based vs. equation-based modeling.

Specific Objectives

Upon completion of this chapter, students will be able to:

- Navigate the interfaces and libraries of major modeling tools especially MATLAB/Simulink andSimscape.
- Explain the basic principles behind simulation solvers and numerical resolution in MATLAB/Simulink andSimscape.
- Set up a basic simulation scenario using at least one tool (e.g., create a simple thermal or electrical system in Simulink).
- Interpret simulation results and assess their relevance to real-world system behavior.

I. Introduction

After having defined the fundamental concepts of modeling and simulation in Chapter 1, we now turn to the methodological and practical aspects that enable their effective implementation. Chapter 2 is dedicated to exploring the different types of models, their construction processes, and the principles guiding their validation and use.This chapter emphasizes the importance of choosing the right modeling approach depending on the system under study—whether deterministic or stochastic, continuous or discrete, analytical or numerical. It also introduces the notion of abstraction levels, highlighting how simplification and representation choices directly influence the accuracy and applicability of simulations.

Furthermore, we will examine the simulation process as a structured methodology: from problem formulation and model design to experimentation and interpretation of results. Special attention will be given to the role of computational tools, such as MATLAB/Simulink, which provide powerful environments for building and testing models in real time.

II. MATLAB Software

MATLAB is a high-level programming environment and numerical computing platform widely used in engineering and scientific research. It enables users to develop algorithms, perform data analysis, visualize results, and implement system models through both command-line scripts and graphical interfaces.

One of MATLAB's strengths lies in its vast ecosystem of **Toolboxes**—specialized libraries that extend its core functionality for specific application areas. These toolboxes are continuously updated to support modern challenges in engineering and technology. Examples of commonly used tool boxes include:

- **Simulink**: A graphical environment for modeling, simulating, and analyzing dynamic systems.
- **Simscape**: Enables multi-domain physical modeling using physical connections and acausal modeling techniques.
- **Simscape Electrical** (formerly SimPowerSystems): Provides component libraries for modeling electrical power systems, including converters, machines, and control systems.

- **Control System Toolbox**: Used for designing and analyzing feedback control systems with tools for root locus, Bode plots, and automatic tuning.
- **Signal Processing Toolbox**: Offers advanced tools for signal analysis, filtering, spectral analysis, and signal transformation.
- **Machine Learning Toolbox**: Facilitates classification, regression, clustering, and dimensionality reduction using machine learning algorithms.
- **Optimization Toolbox**: Provides algorithms for solving constrained and unconstrained optimization problems.
- **Deep Learning Toolbox**: Supports the design and training of deep neural networks for AI and data science applications.

The following figures show an example of a MATLAB script and the corresponding simulation results, highlighting the platform's capacity for integrated modeling and data visualization.

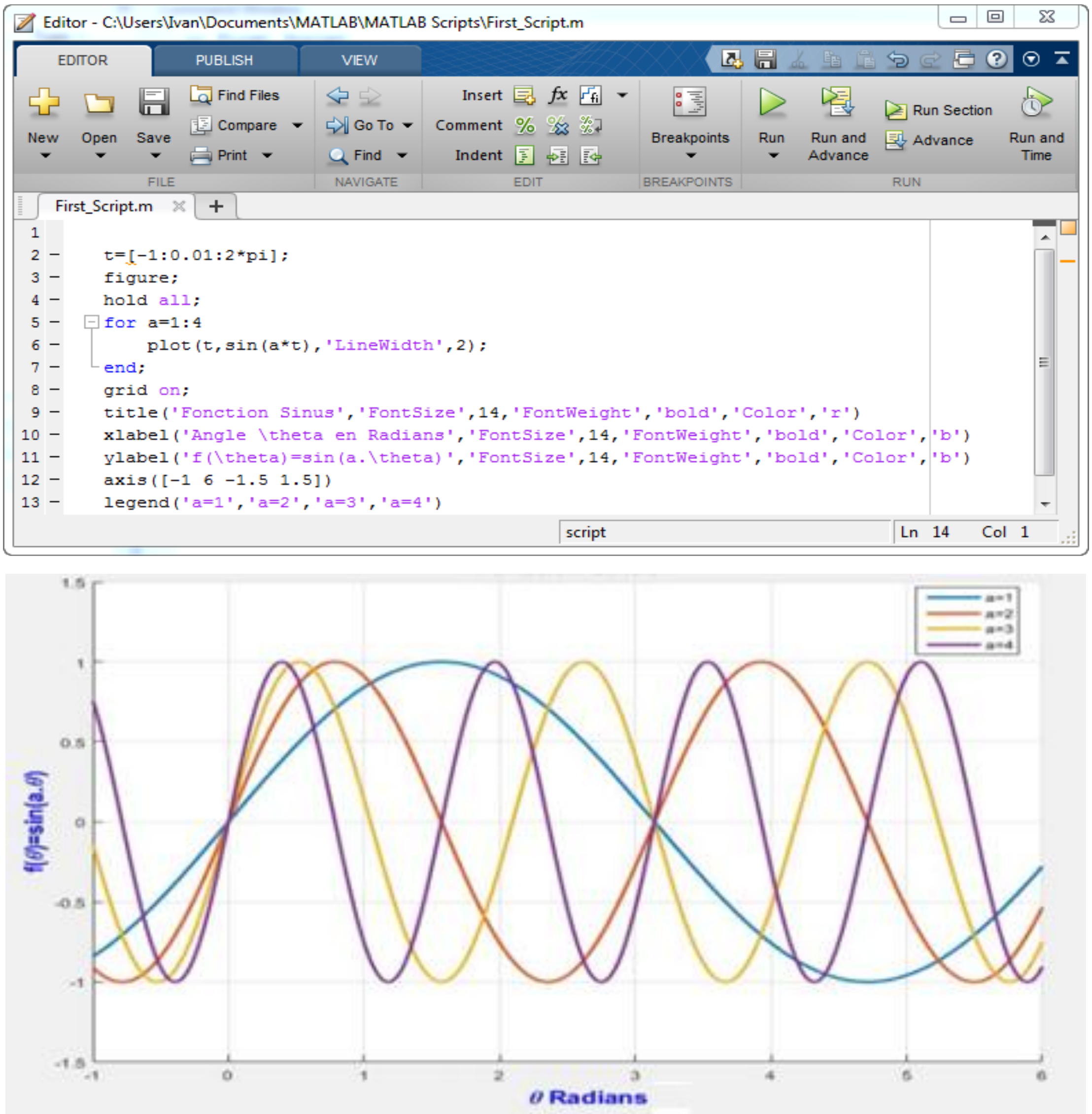


Figure II.1: Example of a MATLAB script

II.1. Simulink

Simulink is MATLAB's graphical environment for modeling, simulating, and analyzing dynamic systems without the need for manual coding of command-line instructions. It allows users to build models by connecting blocks from extensive **block libraries**, organized into thematic **Blocksets** or specialized toolboxes.

Simulink adopts a **causal modeling** approach, where the dynamic behavior of a system is represented by blocks containing mathematical representations—such as transfer functions, state-space equations, or predefined physical components—with clearly defined inputs and outputs. The data flowing between blocks is carried by **oriented digital signals**, and the block outputs are computed numerically at each simulation step based on the transformation defined within the block.

Modern Simulink includes a wide range of specialized libraries, such as:

- **Simscape**: Enables multi-domain physical modeling (electrical, mechanical, thermal, and hydraulic) using acausal connections.
- **Simscape Electrical** (formerly SimPowerSystems): Dedicated to modeling electrical power systems, converters, machines, and renewable energy sources.
- **Simulink Control Design**: Provides tools for designing, tuning, and validating control systems directly within the Simulink environment.
- **Simulink Real-Time**: Allows real-time testing and Hardware-in-the-Loop (HIL) simulations using dedicated target hardware.
- **Stateflow**: Facilitates modeling of event-driven logic, finite-state machines, and complex decision-making systems.
- **Simulink Test**: Enables test case creation, execution, and automated verification of model behavior.

This modeling approach, widely adopted in engineering and industry, requires a solid understanding of the physical laws governing system behavior. A typical workflow begins with formulating the differential equations of the physical phenomena under study, followed by deriving the transfer functions of each subsystem and implementing them in Simulink.

With the integration of Simscape and real-time capabilities, Simulink has evolved from a pure block-diagram modeling tool to a complete simulation platform that bridges numerical analysis, physical modeling, and experimental validation.

Causal vs. Acausal Modeling in MATLAB/Simulink

Causal Modeling (Simulink Standard Blocks)

- **Principle:** The direction of signal flow is explicitly defined from input to output.

- **Representation:** Systems are described using transfer functions, state-space equations, or mathematical operators.

- **Example:** A transfer function block representing a motor's dynamics where voltage is the input and speed is the output.

- **Advantages:**
  - Clear signal flow, easy to trace system behavior.
  - Direct integration with control system design workflows.
- **Limitations:**
  - Requires deriving mathematical equations beforehand.
  - Less intuitive for multi-domain physical systems.

Acausal Modeling (Simscape and Physical Modeling)

- **Principle:** Connections are defined by physical interactions (e.g., voltage-current, force-velocity) without predefined signal direction.

- **Representation:** Components are connected using physical ports that enforce conservation laws automatically.

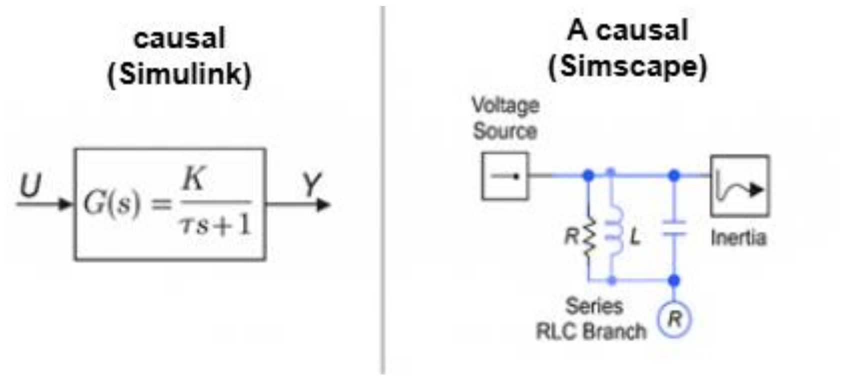
- **Example:** A DC motor modeled by connecting an electrical winding component, a rotational mechanical shaft, and a load, without explicitly writing the equations.

- **Advantages:**
  - Direct physical representation of systems.
  - Ideal for multi-domain modeling (electrical, mechanical, thermal, hydraulic).
  - Equations are generated automatically by the solver.

- **Limitations:**
    - ➢ Less control over solver details.
    - ➢ May be computationally heavier for large systems.

Summary Table

Feature	Causal (Simulink)	Acausal (Simscape)
Signal direction	Explicit input → output	Determined by physics
Equation derivation	Required manually	Generated automatically
Multi-domain support	Limited	Strong
Best for	Control and signal flow	Physical multi-domain models



- **Simulink Blocs**

All representations in Simulink are created using **blocks**, each defined by its specific function and characterized by its inputs and outputs.

These blocks are connected by **time-based signals** that transmit data from one component to another within the model, ensuring the correct flow of information during simulation.

In Simulink, several types of signals can be used:

- **Scalar signals**: Represent a single time-varying value (e.g., motor speed, voltage).
- **Vector signals**: Contain multiple time-varying elements grouped together (e.g., three-phase voltages).
- **Matrix signals**: Represent multidimensional data, useful in advanced system modeling or image processing.

- **Bus signals**: Bundle multiple signals of different types into a single connection, improving model readability and reducing clutter.
- **Virtual signals**: Used internally by Simulink for organizing data without affecting simulation results.

Understanding how to select and configure these signal types is essential for building efficient, accurate, and readable models. Proper signal management ensures that data flows correctly between blocks and that simulation outputs reflect the intended system behavior.

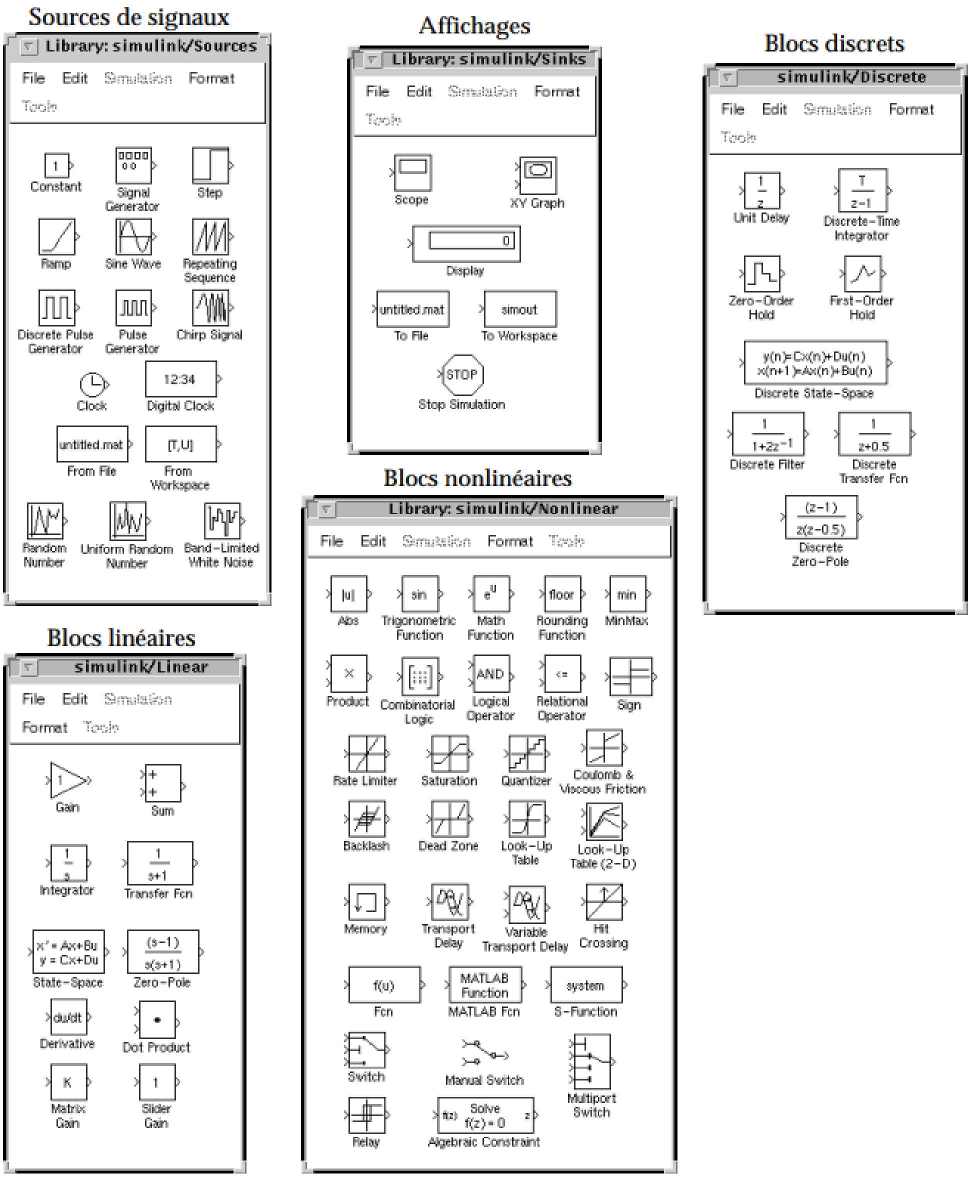


Figure II.2: Simulink blocs

II.1.1.Simulation with Simulink

Choosing Simulation Parameters

Before running a simulation, it is essential to select parameters that are appropriate for the system model. In the Simulation menu, selecting Parameters opens the *Simulation Parameters* window. From this window, the user can configure key settings such as Solver**,** Workspace I/O**,** and Diagnostics**.**

a. Solver Parameters

The Solver defines how Simulink performs the numerical integration of the model's equations over time. Choosing the right solver and step size is critical for ensuring both the accuracy and efficiency of the simulation. Solverscanbebroadlycategorizedinto:

- **Fixed-step solvers**: Use a constant step size throughout the simulation. These are often required for real-time applications, hardware-in-the-loop (HIL) simulations, and discrete-time control systems.
- **Variable-step solvers**: Automatically adjust the step size based on system dynamics, allowing for higher accuracy when simulating systems with fast or slow transients.

The choice between fixed-step and variable-step depends on the nature of the system, the desired accuracy, and the computational resources available.

b. Workspace I/O Parameters

The Workspace I/O settings determine how simulation data is exchanged between Simulink and the MATLAB workspace. Configuring the se parameter sallows the user to:

- **Log simulation data** (e.g., time series of signals, states, or outputs) to the MATLAB workspace for further analysis, plotting, or post-processing.
- Define **variable names** under which the simulation results will be stored.
- Choose the **format of stored data** (e.g., structure with time, array, or structure without time).

Proper configuration of Workspace I/O ensures that the results can be easily reused for validation, optimization, or report generation.

c. Diagnostics Parameters

The Diagnostics settings control how Simulink handles errors, warnings, and potential issues during model execution. They help ensure model robustness and detect configuration or numerical problems early. Diagnostics canbeused to:

- Detect **algebraic loops**, inconsistent initial conditions, or sample-time mismatches.
- Flag **invalid model settings** or missing data.
- Monitor for **parameter overflows**, division by zero, or other numerical errors.
- Configure Simulink to stop the simulation, issue a warning, or ignore the event depending on the severity of the detected issue.

Well-configured diagnostics are essential for debugging complex models and ensuring simulation reliability.

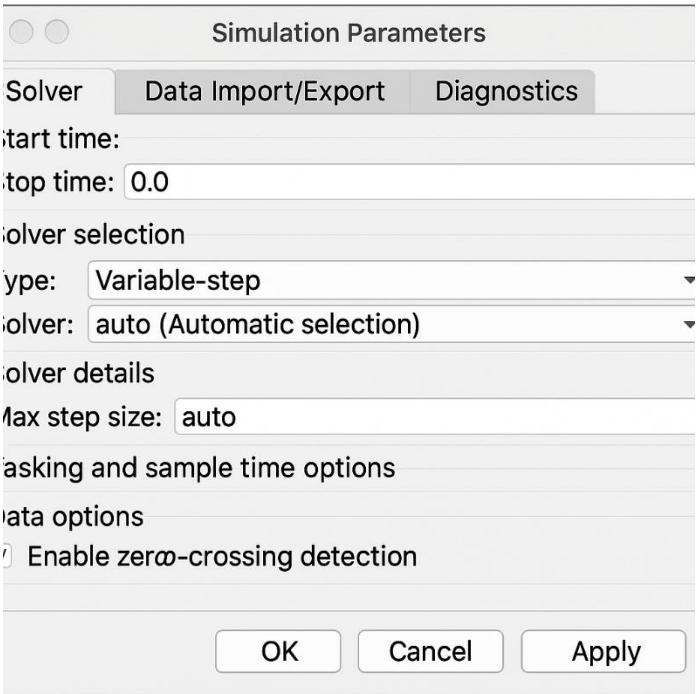


Figure II.3: Simulink window parameters

II.2 Simscape

Simscape adopts an acausal modeling approach, enabling system modeling through the assembly of predefined physical components. The physical behavior of each component is inherently handled by the software, making it possible to model complex systems without manually writing the differential equations that govern their dynamics.

Simscape includes a comprehensive library of basic components spanning multiple physical domains—mechanical, electrical, hydraulic, pneumatic, thermal, and magnetic—and is structured into specialized add-onmodules:

- **Simscape Multibody**: Allows importing 3D CAD models and integrating them directly into a multi-physics simulation or control system. The kinematic properties of the parts are taken into account, and their dynamic behavior is automatically incorporated.
- **Simscape Electrical** (formerly Power Systems): Contains components for modeling electrical power systems, including various motor technologies, power supplies, filters, actuators, and power electronics.
- **Simscape Electronics**: Provides components for electronic and mechatronic systems, such as servo motors, PWM controllers, H-bridges, converters, and logic gates.
- **Simscape Fluids**: Enables modeling of hydraulic and pneumatic circuits, including pumps, valves, distributors, cylinders, and fluid control devices.
- **Simscape Driveline**: Offers components for modeling mechanical power transmission systems, such as gearboxes, brakes, clutches, differentials, and couplings.

By combining these modules, engineers can build complete multi-domain models that simulate the interactions between electrical, mechanical, thermal, and fluid subsystems in a single environment. This capability makes Simscape a powerful tool for both concept designandsystem validation**.**

II.3 Launching the software

To start MATLAB, double-click the **MATLAB** icon located on the desktop or search for it in the operating system's application menu. Upon launch, MATLAB initializes its integrated development environment (IDE), preparing all necessary toolboxes and paths.

II.3.1. Matlab Environment Window

When MATLAB opens, the user is presented with an integrated workspace composed of several main panels and tabs. The default layout typically includes:

1. **Command Window**
    - ➢ The central interface for executing MATLAB commands interactively.
    - ➢ Users can run scripts, call functions, and visualize immediate results.
2. **CurrentFolder**
    - ➢ Displays the contents of the current working directory.

- Allows quick navigation between project folders and files.

3. **Workspace**
   - Shows all variables currently loaded in memory.
   - Provides information such as variable size, type, and value.
4. **Editor (Script/FunctionWindow)**
   - Used for writing, editing, and saving MATLAB programs (.m files).
   - Includes syntax highlighting, debugging tools, and code folding.
5. **Command History***(optional panel)*
   - Records all commands entered in the Command Window.
   - Enables re-executing or editing past commands.

Additional Interface Features in Recent Versions

- **Home Tab**: Quick access to file management, variable import/export, add-on installation, and preferences.
- **Plots Tab**: Tools for creating and customizing 2D and 3D plots.
- **Apps Tab**: Direct access to MATLAB's interactive applications (e.g., Curve Fitting, Signal Analyzer, Control System Designer).
- **Live Editor**: Enables the creation of **Live Scripts** (*.mlx* files) that combine code, output, equations, and formatted text for richer documentation and interactive reports.

I.3.2. the Simulink Environment Window

To launch **Simulink**, click on the **Simulink icon** located in the MATLAB toolbar or type `simulink` in the MATLAB Command Window. This action opens the **Simulink Library Browser**, which provides access to all component libraries and preconfigured blocksets available in your MATLAB installation.

From the Library Browser, you can:

- Browse **built-in libraries** (e.g., Sources, Sinks, Math Operations, Continuous, Discrete).
- Access specialized **toolbox libraries** such as Simscape, Simscape Electrical, Stateflow, and Control System Toolbox.
- Search directly for blocks using the **search bar** to quickly locate components.
- Drag and drop blocks into a new or existing model workspace to build your system.

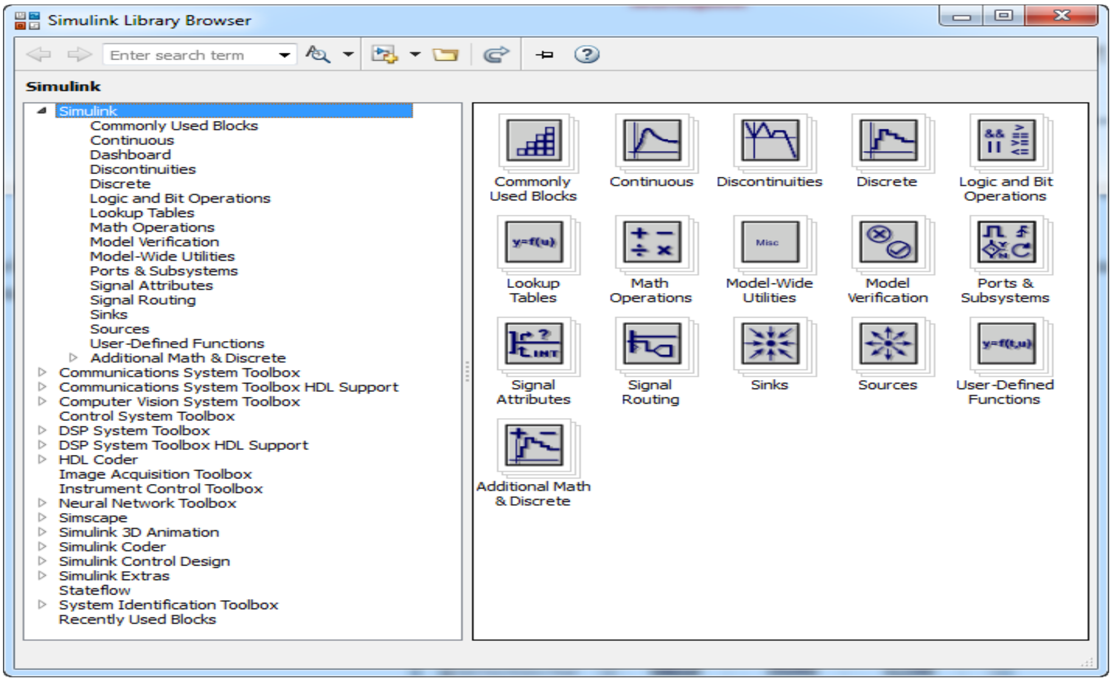


Figure II.4 : Simulink library window

a. Naming Files in MATLAB/Simulink

You can assign any name of your choice to a file in MATLAB or Simulink, provided that you follow these rules:

- Do not use **spaces** in file names.
- Do not use **accented characters** or special symbols.

b. The MATLAB Path

The MATLAB *path* is the set of folders that MATLAB can access during the current session. Any file located in one of these folders can be executed or called without explicitly specifying its full directory path.

To ensure smooth access to your working files, it is important to **add all relevant project folders** to the MATLAB path. This allows you to execute scripts or call functions simply by typing their names in the Command Window, without having to navigate to their location.

However, some rules must be observed:

- Avoid having **two scripts or functions with the same name** anywhere in the MATLAB path, as MATLAB will execute only the first one it finds.
- Keep your project files organized in dedicated folders to prevent conflicts.
- Use the MATLAB **Set Path** tool or the `addpath` function to manage folders dynamically.

This path management strategy improves workflow efficiency and reduces errors related to missing or inaccessible files.

III. Conslusion

In this second chapter, we have moved from the conceptual foundations of modeling and simulation toward their methodological and practical dimensions. By examining the different types of models—deterministic, stochastic, continuous, and discrete—we have highlighted how the choice of representation directly influences the accuracy and relevance of results.

We also emphasized the simulation process as a structured methodology, from problem formulation and model construction to experimentation and interpretation. This systematic approach ensures that simulations are not only technically sound but also meaningful for analysis and decision-making.

Finally, the introduction of computational tools such as MATLAB/Simulink illustrates how theory can be translated into practice, enabling real-time experimentation and validation. These tools bridge abstraction and application, making simulation a powerful instrument for engineers, scientists, and decision-makers.

Chapter 3: Modeling and Simulation of a Photovoltaic System

Purpose of the Chapter

This chapter focuses on the theoretical and practical aspects of modeling and simulating photovoltaic (PV) systems using MATLAB/Simulink and Simscape Electrical. It aims to provide students with a solid understanding of solar energy conversion principles and their application in modern renewable energy systems. The modeling process will cover the PV cell, module, and array levels, as well as the associated power conditioning systems, such as DC-DC converters and maximum power point tracking (MPPT) algorithms.

Intermediate Objectives

By the end of this chapter, students will be able to:

- Understand the physical principles of photovoltaic energy conversion.
- Identify the key parameters influencing PV system performance.
- Model PV cells, modules, and arrays in MATLAB/Simulink.
- Integrate power converters and control algorithms into the PV system model.

Specific Objectives

Upon completing this chapter, students will be able to:

- Derive the mathematical model of a PV cell based on the single-diode or double-diode model.
- Use MATLAB/Simulink and Simscape Electrical to implement PV models.
- Simulate the effect of environmental conditions (irradiance, temperature) on PV output.
- Design and implement an MPPT algorithm (e.g., Perturb& Observe, Incremental Conductance).
- Validate PV system performance through numerical simulation.

I. Introduction

Solar photovoltaic technology has become a key component of the global renewable energy mix, offering a clean and sustainable way to generate electricity. The rapid growth of PV installations worldwide has driven the need for accurate and reliable modeling tools to predict system performance under varying environmental conditions.

Modeling a PV system involves representing the electrical behavior of solar cells, integrating them into modules and arrays, and coupling them with power electronic converters for grid connection or stand-alone applications. Simulation allows engineers to test different configurations, evaluate control strategies, and optimize system performance before physical implementation.

In this chapter, we will start by reviewing the fundamental operating principles of PV cells, followed by the development of their mathematical model. We will then implement this model in MATLAB/Simulink, add a DC-DC and DC-AC converters, and integrate an MPPT controller to optimize energy extraction from the solar array.

II. The Photovoltaic Effect

When a semiconductor material is exposed to sunlight, the atoms within the material are “bombarded” by photons—the elementary particles that compose light. Under the impact of this photon bombardment, the electrons located in the outer energy layers (known as valence electrons) tend to be “dislodged” from their atomic bonds.

If a dislodged electron returns to its initial state, the energy gained is dissipated as heat, resulting in a slight increase in the material’s temperature. In this case, the kinetic energy of the photon is transformed into thermal energy.

However, in photovoltaic (PV) cells, a fraction of these electrons do not return to their original state. Instead, the dislodged electrons are separated and create a small direct current (DC)voltage. In this process, part of the photons’ kinetic energy is directly converted into electrical energy—a phenomenon known as the photovoltaic effect.

The photovoltaic effect is thus the direct conversion of solar radiation energy into electrical energy, typically using silicon-based cells. To obtain a sufficient power output, multiple PV cells are connected together to form a solar module.

This effect—the generation of electricity directly from light—was first observed in 1839 by the French physicist Edmond Becquerel. However, it was not until the 1950s that researchers at Bell Laboratories in the United States succeeded in manufacturing the first practical photovoltaic cell, the basic building block of a PV system.

III. The Photovoltaic Cell

Photovoltaic cells are manufactured using silicon as the base material. The process begins with silicon ingots, which are sliced into thin wafers of either P-type or N-type semiconductor. A solar cell is then created by forming a junction between two regions of opposite types, known as a PN junction.

Near the junction, an electric field is established, which ensures the separation of positive and negative charges generated by the incident light. To enable current collection, metallic contacts in the form of a fine grid are deposited on the front surface, while a full metallic contact is applied to the rear surface of the cell. This grid structure allows sunlight to penetrate the cell while ensuring efficient charge extraction.

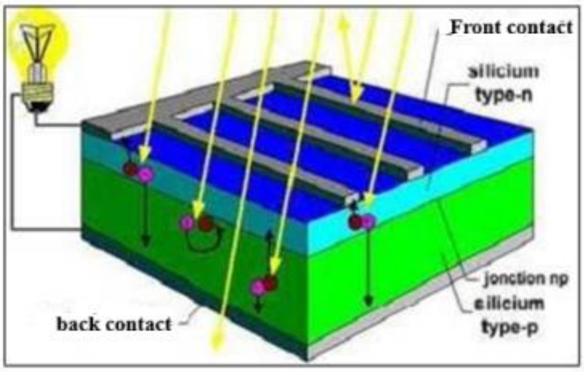


Figure III.1: Schematic representation of solar cell

A photovoltaic cell is a device that converts solar energy into electrical energy. This conversion isbased on three main mechanisms:

1. Absorption of photons by the material composing the device.
2. Conversion of photon energy into electrical energy, corresponding to the creation of electron-hole pairs within the semiconductor material.
3. Collection of the generated charge carriers within the device.

To collect these generated carriers, an electric field is required to separate the created electron-hole pairs. This is most commonly achieved through the use of a PN junction, which provides the necessary internal electric field for efficient charge separation.

III.1. Electrical Characteristics of a Photovoltaic Cell

Figure (III.1) shows the equivalent circuit diagram of a photovoltaic cell under illumination. The model consists of a photogenerated current source (I_{ph}) connected in parallel with a diode. Additionally, two parasitic resistances are included in the model to account for non-idealities.

These resistances influence thecurrent-voltage (I–V) characteristic of the cell as follows:

- **Series Resistance (Rs)**: Represents the internal resistance of the cell. It primarily depends on the resistivity of the semiconductor material and the contact resistance of the metallic collector grids. A high series resistance reduces the fill factor and limits current flow at high load conditions.
- **Shunt Resistance (Rsh)**: Accounts for leakage currents across the junction. It is typically caused by manufacturing defects, such as micro-cracks, impurities, or flaws in the cell structure. A low shunt resistance reduces the open-circuit voltage and decreases cell efficiency.

IV. Modeling a Photovoltaic Cell

A variety of mathematical models have been developed to represent the behavior of the semiconductor junctions that form the basis of photovoltaic (PV) cells. In the literature, several representative PV cell models can be found, differing in their formulation, level of complexity, and the number of parameters involved in calculating the final voltage and current of the photovoltaic generator.

IV.1. Basic Model

The principle of the photovoltaic effect is essentially that of a diode. When the **PN junction** is exposed to light and the cell is connected to a load resistance (*Rch*), a current (*I*) flows through the load, and a voltage (*V*) appears across its terminals. This electrical behavior is governed by the combined effects of the light-generated current and the diode's intrinsic characteristics.

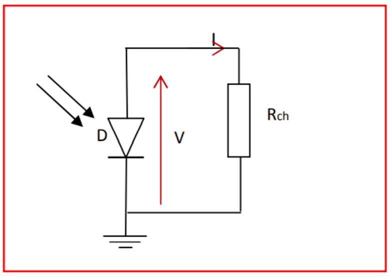


Figure III.2: Schematic diagram of the photovoltaic effect

IV.2. Ideal Model

The ideal model is the simplest representation of a solar cell, as it considers only the diffusion phenomenon and neglects all parasitic effects. In this simplified equivalent circuit, the solar cell is modeled as a current source connected in parallel with a diode [43]. The current source represents the light-generated current (*Ipv*), while the diode models the PN junction's intrinsic behavior under illumination.

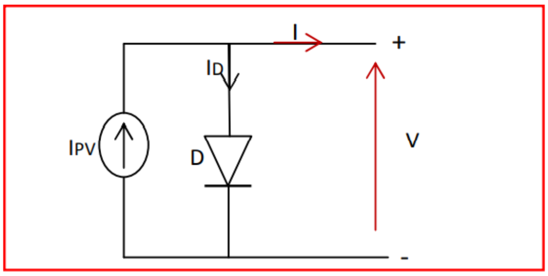


Figure III.3: Equivalent circuit of a PV cell - ideal model -

In the ideal model, the output current (*I*) of the solar cell is obtained by applying Kirchhoff's current law to the equivalent circuit:

$$I = I_{ph} - I_D$$

where:

- Ipv : the photo generated current (proportional to incident irradiance).
- I_D : the diode current, given by the Shockley diode equation:

$$I_D = I_0[\exp(\frac{qV}{nkT}) - 1]$$

Combining the two expressions gives the I–V equation of the ideal PV cell:

$$I = I_{ph} - I_0[\exp(\frac{qV}{nkT}) - 1]$$

where:

- I_0 : diode reverse saturation current (A)
- q : electron charge (1.602×10^{-19} C)
- V : terminal voltage of the cell (V)
- n : diode ideality factor (dimensionless, typically 1 to 2)
- k : Boltzmann constant (1.381×10^{-23} J/K)
- T : cell absolute temperature (K)

Interpretation:

- At **short-circuit condition** (V=0), the cell delivers its maximum current, approximately equal to I_{ph}.
- At **open-circuit condition** (I=0), the voltage is:

$$V_{oc} \approx \frac{nkT}{q} \ln\left(\frac{I_{ph}}{I_0} + 1\right)$$

This model ignores series and shunt resistances, meaning it overestimates the real performance of the PV cell.

IV.3.Model with Ohmic Losses

This model takes into account the resistivity of the semiconductor material and the ohmiclosses that occur at the metallic contacts of the photovoltaic cell. These non-idealities are represented in the equivalent circuit by introducing a series resistance (R_s).

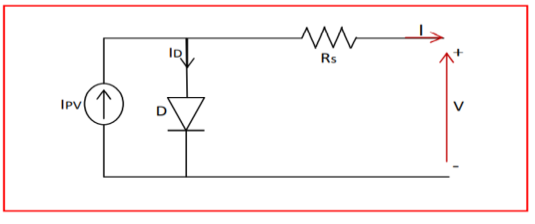


Figure III.4: Equivalent circuit of a PV cell

The output current equation of the PV cell then becomes:

$$I = I_{ph} - I_0[\exp(\frac{q(V + IR_s)}{nkT}) - 1]$$

where:

- I_{ph} : photocurrent (A), proportional to solarirradiance
- I_0 : diode reverse saturation current (A)
- R_s : series resistance (Ω), representing ohmic losses in the semiconductor and contacts
- V : output voltage (V)
- n : diode ideality factor
- k : Boltzmann constant
- T : absolutetemperature (K)
- q : electron charge

Interpretation:

- The presence of $R_s$reduces the slope of the I–V curve near the short-circuit condition.
- A high series resistance causes significant voltage drops inside the cell, reducing both the fill factor (FF) and the maximum power output (P_{max}).
- This model is more realistic than the ideal model, but still incomplete since it neglects leakage currents.

IV.4.SingleDiode Model

The single-diode model extends the PV cell representation by also accounting for leakage currents, which are modeled through a parallel (or shunt) resistance R_p. This model is widely adopted in practice, as it provides a good balance between accuracy and simplicity.

It is the standard model used by manufacturers when publishing the technical specifications of their solar cells and modules in data sheets. In fact, the single-diode model is generally considered sufficiently accurate and serves as a reference model for cataloging and simulating the behavior of commercial PV modules.

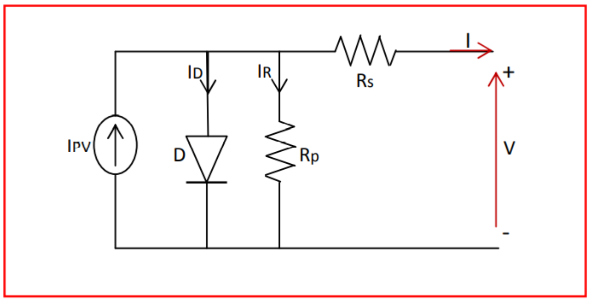


Figure III.5: Equivalent circuit of a PV cell – 1-D model -

The corresponding current–voltage relationship is expressed as:

$$I = I_{ph} - I_0[\exp(\frac{q(V + IR_s)}{nkT}) - 1] - \frac{V + IR_s}{R_p}$$

where:

- R_s : series resistance (Ω), representing ohmic/contact losses
- R_p : shunt resistance (Ω), representing leakage currents through the junction
- other parameters remain as previously defined

Interpretation:

- A low R_p value leads to high leakage currents, reducing the open-circuit voltage and overall efficiency.
- A high R_s reduces the short-circuit current and fill factor.
- With appropriate parameter extraction, this model can accurately reproduce the I–V and P–V curves of real PV modules under different operating conditions.

IV.5.TwoDiode Model

The two-diode model is considered the most accurate representation of the real behavior of a solar cell. Unlike the single-diode model, it accounts for the charge transfer mechanisms within the cell. The additional diode enables the equivalent circuit to reproduce the recombination effects of electrons within the semiconductor material, which are neglected in the single-diode approach.

Single-diode models are based on the assumption that electron recombination losses are negligible. However, in practice, these effects can significantly impact the performance of PV cells, especially under low irradiance or non-ideal operating conditions.

The inclusion of a second diode provides greater modeling accuracy but also increases the number of parameters to be estimated—typically up to seven parameters. This higher level of complexity complicates the parameter extraction process, making it more challenging to implement in practice.

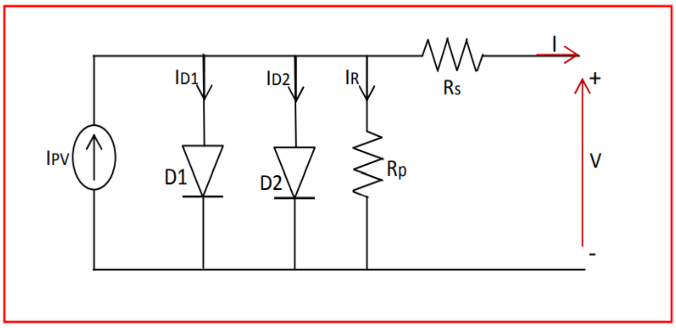


Figure III.6: Equivalent circuit of a PV cell – 2-D model -

The current–voltage equation of the two-diode model is expressed as:

$$I = I_{ph} - I_{01}\left[\exp\left(\frac{q(V+IR_s)}{n_1 kT}\right)-1\right] - I_{02}\left[\exp\left(\frac{q(V+IR_s)}{n_2 kT}\right)-1\right] - \frac{V+IR_s}{R_p}$$

where:

- I_{01}, I_{02} : reverse saturation currents of the two diodes
- n_1, n_2 : ideality factors of each diode
- R_s : series resistance (Ω)
- Rp : shunt resistance (Ω)
- other symbols are as previously defined

V. Equivalent Circuit of a PV Cell

The electrical behavior of a photovoltaic (PV) cell under illumination can be represented by the single-diode model, also known as the five-parameter model. Its equivalent circuit is shown in Figure (III.7). It consists of:

- A photocurrent source (I_{ph}) that represents the current generated by incident solar radiation.
- A diode that models the PN junction behavior of the solar cell.

- A series resistance (R_s) accounting for ohmic losses due to the semiconductor material and metallic contacts.
- A shunt resistance (R_{sh}) representing leakage currents across the junction.

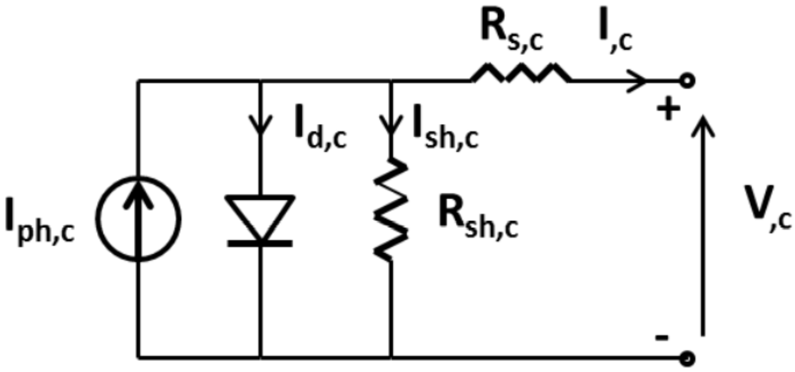


Figure III.7: Equivalent circuit of a PV cell

V.1. Mathematical Model of the I–V Characteristic

The output current–voltage (I–V) relationship of the PV cell is expressed as:

$$I = I_{ph} - I_0[\exp(\frac{q(V + IR_s)}{aN_s kT}) - 1] - \frac{V + IR_s}{R_{sh}}$$

where:

- I_{ph} : photocurrent (A), proportional to solarirradiance G
- I0 : reverse saturation current of the diode (A)
- Ns : number of cells in series
- V : terminal voltage of the module (V)
- I : output current of the module (A)
- T : ambient/celltemperature (K)
- a : diode ideality factor (typically 1–2)
- G : solarirradiance (W/m²)
- Rs : series resistance (Ω)
- Rsh : shunt resistance (Ω)

V.2. Physical Interpretation

- Increasing irradiance (**G**) increases I_{ph}, shifting the I–V curve upward.
- Increasing temperature **(T)** decreases the open-circuit voltage (V_{oc}), shifting the curve left.

- A high R_s reduces the slope near short-circuit current and lowers maximum power.
- A low R_{sh} increases leakage currents, reducing V_{oc} and efficiency.

Thus, the I–V characteristicof a PV cell is strongly influenced by both its internal parameters (I_{ph},I_0,R_s,R_{sh}) and the environmental conditions (temperature, irradiance).

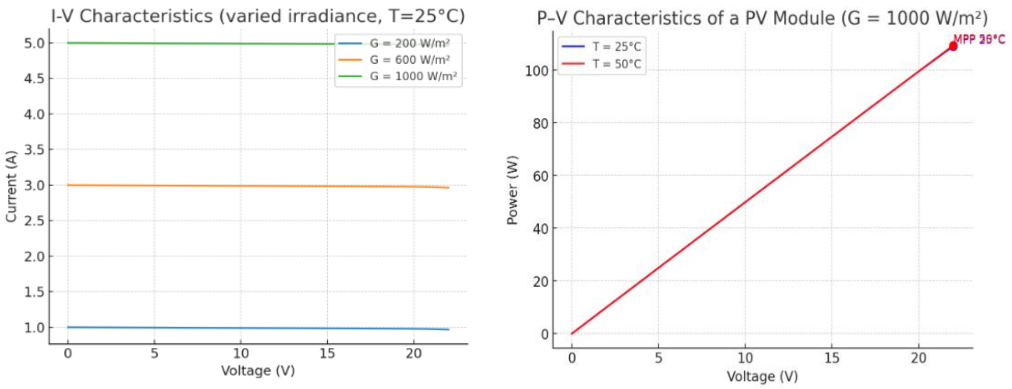


Figure III.8.I–V and P–V characteristics

V.3. Interpretation of the I–V and P–V Characteristics

1. **Characteristics under Different Irradiance Levels**

- As irradiance increases (from 200 W/m² to 1000 W/m²), the photogenerated current**I_{ph}** increases proportionally.
- This results in a higher short-circuit current **(I_{sc})** and a steeper I–V curve.
- The open-circuit voltage (V_{oc}) is only slightly affected by irradiance.
- Thus, higher irradiance leads to greater power generation capability.

2. **P–V Characteristics under Different Temperatures**

- At constant irradiance (1000 W/m²), increasing temperature (from 25°C to 50°C) reduces the open-circuit voltage **(V_{oc})**.
- The maximum power point **(MPP)** shifts toward lower voltages as temperature rises.
- Consequently, the maximum output power **(P_{max})** decreases at higher temperatures, demonstrating the negative temperature coefficient of PV cells.

Summary:

- **Irradiance** mainly affects the **current** and therefore the amplitude of the I–V curve.
- **Temperature** mainly affects the **voltage** and reduces the maximum power output.

VI. Modeling of the Buck Converter

The Buck converter (also called a step-down chopper) is a DC–DC power converter that reduces the input voltage V_i to a lower output voltage V_o, while ideally maintaining power balance. It is widely used in photovoltaic (PV) systems for **Maximum Power Point Tracking (MPPT)** and efficient energy conversion.

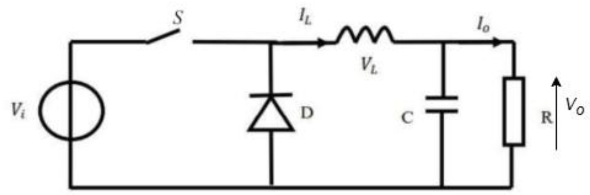


Figure III.9: Buck converter

The converter consists of:

- a controlled switch (typically a MOSFET or IGBT),
- a diode,
- an inductor L,
- a capacitor C, and
- a resistive load R.

Its operation is governed by the duty cycle $\alpha \in [0,1]$, which controls the fraction of the switching period T for which the switch is ON.

VI.1. Operating Principle

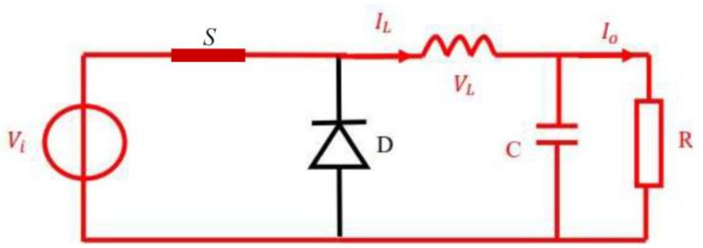
The Buck converter operates in two modes within each switching cycle:

1. **Switch ON interval (0≤t<αT):**
   The inductor is directly connected to the source, storing energy while simultaneously supplying the load.
2. **Switch OFF interval (αT≤t<T):**
   The switch is open, the diode conducts, and the inductor releases its stored energy to the load.

VI.2. Mathematical Modeling

a- ON Interval: 0≤t<αT

When the switch is closed:



$$v_L = v_i - v_o \Rightarrow \frac{di_L}{dt} = \frac{v_i - v_0}{L}$$

The capacitor current is:

$$i_c = i_L - \frac{v_0}{R} \Rightarrow \frac{dv_0}{dt} = \frac{1}{C}(i_L - \frac{v_0}{R})$$

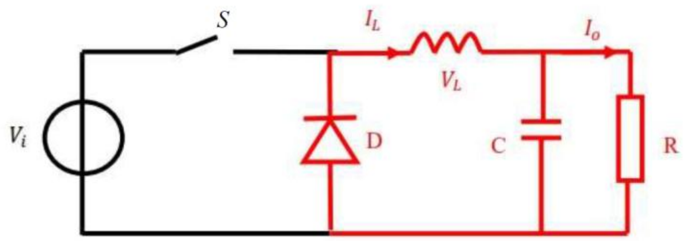
State-Space (ON):

$$\dot{x} = A_{on}x + B_{on}V_i, \quad x = \begin{bmatrix} i_L \\ v_0 \end{bmatrix}$$

$$A_{on} = \begin{bmatrix} 0 & -\frac{1}{L} \\ \frac{1}{C} & -\frac{1}{RC} \end{bmatrix}, \quad B_{on} = \begin{bmatrix} \frac{1}{L} \\ 0 \end{bmatrix}$$

b- OFF Interval: αT≤t<T

When the switch is open:



$$v_L = -v_o \Rightarrow \frac{di_L}{dt} = \frac{-v_0}{L}$$

The capacitor equation remains:

$$\frac{dv_0}{dt} = \frac{1}{C}(i_L - \frac{v_0}{R})$$

State-Space (OFF):

$$\dot{x} = A_{off}\,x, \quad A_{off} = \begin{bmatrix} 0 & -\frac{1}{L} \\ \frac{1}{C} & -\frac{1}{RC} \end{bmatrix}$$

c- Inductor Current Ripple

During ON:

$$\Delta i_L^{on} \approx \frac{V_i - V_o}{L}(\alpha T)$$

During OFF:

$$\Delta i_L^{off} \approx \frac{-V_o}{L}(1-\alpha)T$$

For steady-state periodic operation:

$$\Delta i_L^{on} + \Delta i_L^{off} = 0$$

d- Steady-State Relation

Applying the ripple balance condition:

$$\frac{V_i - V_o}{L}(\alpha T) - \frac{V_o}{L}(1-\alpha)T = 0$$
$$\Rightarrow V_o = \alpha V_i$$

Thus, in Continuous Conduction Mode (CCM), the output voltage is directly proportional to the input voltage and the duty cycle.

VI.3. Average State-Space Model

By averaging over one switching period, we obtain the continuous-time averaged model:

$$\frac{di_L}{dt} = \frac{1}{L}(\alpha V_i - V_o),$$
$$\frac{dv_o}{dt} = \frac{1}{C}(i_L - \frac{V_o}{R}),$$

This averaged model is particularly useful for control design and system-level simulation in Simulink.

where:

- i_L : inductor current (A)
- u =Vo : output voltage (V)
- Vi : input voltage (V)
- R : loadresistance (Ω)
- L : inductance (H)
- C : capacitance (F)
- $\alpha \in [0,1]$: duty cycle of the switching signal

VI.4. Key Observations

- The Buck converter always steps down the voltage (V_o<V_i).
- The duty cycle **α** controls the output voltage.
- The inductor ensures current continuity, while the capacitor reduces voltage ripple.
- Two operation modes exist:
    - **CCM (Continuous Conduction Mode):** inductor current never falls to zero, relation $V_o=\alpha V_i$holds.

- **DCM (Discontinuous Conduction Mode):** inductor current reaches zero, requiring a different analysis.

From these equations, the following Simulink block diagram of the Buck converter can be established:

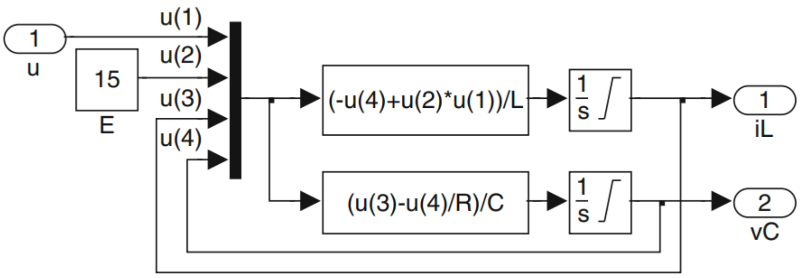


Figure III.10 :Simulink model for buck converter

VI.5. Modeling of the Buck Converter based on transfer function

We start from the averaged state equations in Continuous Conduction Mode (CCM):

$$\frac{di_L}{dt} = \frac{1}{L}(\alpha V_i - V_o),$$
$$\frac{dv_o}{dt} = \frac{1}{C}(i_L - \frac{V_o}{R}),$$

a- Define steady-state operating point

At steady state:

$$I_L = \frac{V_o}{R}, V_o = DV_i$$

Where D=α is the duty cycle.

We now consider small perturbations around this operating point:

$$i_L = I_L + \hat{i}_L, \quad v_o = V_O + \hat{v}_o, \quad d = D + \hat{d}, \quad v_i = V_i + \hat{v}_i$$

Here, the variables with hats (^) represent small-signal variations.

b- Linearize the equations

Substitute into the state equations and neglect second-order small terms:

1. **Inductor equation**:

$$\frac{d\hat{i}_L}{dt} = \frac{1}{L}(V_i\hat{d} + D\hat{v}_i - \hat{v}_o)$$

2. **Capacitor equation:**

$$\frac{d\hat{v}_o}{dt} = \frac{1}{C}\left(\hat{i}_L - \frac{\hat{v}_o}{R}\right)$$

c- Laplace transform

Taking the Laplace transform:

$$s\hat{I}_L(s) = \frac{1}{L}(V_i\hat{d}(s) + D\hat{V}_i(s) - \hat{V}_o(s)),$$
$$s\hat{V}_o(s) = \frac{1}{C}\left(\hat{I}_L(s) - \frac{\hat{V}_o(s)}{R}\right)$$

From the capacitorequation:

$$\hat{I}_L(s) = Cs\hat{V}_o(s) + \frac{\hat{V}_o(s)}{R}$$

Substitute into the inductor equation:

$$s\left(Cs\hat{V}_o(s) + \frac{\hat{V}_o(s)}{R}\right) = \frac{1}{L}(V_i\hat{d}(s) + D\hat{V}_i(s) - \hat{V}_o(s))$$

Rearrange equation :

$$\left(LCs^2 + \frac{L}{R}s + 1\right)\hat{V}_o(s) = V_i\hat{d}(s) + D\hat{V}_i(s)$$

From this general form, we can directly read the transfer functions:

- Control-to-Output (duty to output voltage):

$$G_{vd}(s) = \frac{\hat{V}_o(s)}{\hat{d}(s)} = \frac{V_i}{LC}\frac{1}{s^2 + \frac{1}{RC}s + \frac{1}{LC}}$$

- Input-to-Output:

$$G_{vg}(s) = \frac{\hat{V}_o(s)}{\hat{V}_i(s)} = \frac{D}{LC} \frac{1}{s^2 + \frac{1}{RC}s + \frac{1}{LC}}$$

- Load-to-Output (output impedance):

  If we introduce a perturbation load current, we obtain:

$$G_{vi}(s) = \frac{\hat{V}_o(s)}{\hat{I}_R(s)} = -\frac{R}{1 + sRC + s^2 LC}$$

VII. Modeling of the Boost Converter

We consider an ideal Boost converter with input voltage v_i, duty cycle d, inductor current i_L, output voltage v_o, inductor L, capacitor C, and load R.

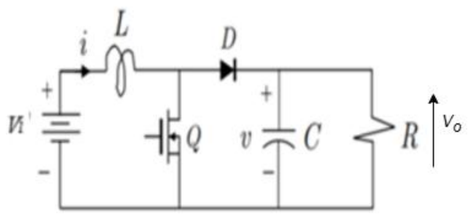
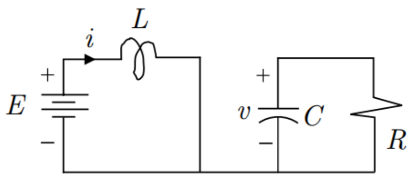


Figure III.11: Boost converter

ON interval (0≤t<dT)

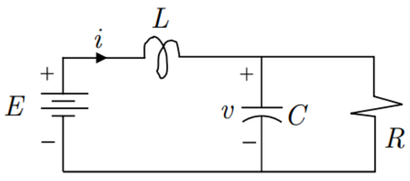
- Switch closed, diode OFF.



$$\frac{di_L}{dt} = \frac{v_i}{L}, \quad \frac{dv_o}{dt} = -\frac{v_o}{RC}$$

OFF interval (dT≤t<T)

- Switch open, diode ON.



(b) The switch at the position $u = 0$.

$$\frac{di_L}{dt} = \frac{v_i - v_o}{L}, \; \frac{dv_o}{dt} = \frac{1}{C}(i_L - \frac{v_o}{R})$$

Averaged model over one switching period

$$\frac{di_L}{dt} = \frac{1}{L}(v_i - (1-D)v_o),$$
$$\frac{dv_o}{dt} = \frac{1}{C}((1-D)i_L - \frac{v_o}{R})$$

a- Steady-State Operating Point

At steady state:

$$\overline{v_L} = 0 \Rightarrow V_i = (1-D)V_O$$
$$\overline{i_C} = 0 \Rightarrow (1-D)I_L = \frac{V_O}{R}$$

Thus the DC conversion law is:

$$V_o = \frac{V_i}{1-D}$$

This shows that the output voltage is always **greater** than the input in CCM (D<1).

b- Small-Signal Linearization

We perturb around steady state:

$$i_L = I_L + \hat{i}_L, \quad v_o = V_O + \hat{v}_o, \quad d = D + \hat{d}, \quad v_i = V_i + \hat{v}_i$$

Inductor equation:

$$\frac{di_L}{dt} = \frac{1}{L}(v_i - (1-d)v_o)$$

Substitute decomposed variables:

$$\frac{d(I_L + \hat{i}_L)}{dt} = \frac{1}{L}[(V_i + \hat{v}_i) - (1 - D - \hat{d})(V_O + \hat{v}_o)]$$

Expanding and neglecting second-order small terms:

$$\frac{d\hat{i}_L}{dt} = \frac{1}{L}(\hat{v}_i + V_O\hat{d} - (1-D)\hat{v}_o)$$

Capacitor equation:

$$\frac{dv_o}{dt} = \frac{1}{C}\left((1-d)i_L - \frac{v_o}{R}\right)$$

Substitute decomposed variables:

$$\frac{d(V_O+\hat{v}_o)}{dt}=\frac{1}{C}\left[(1-D-\hat{d})(I_L+\hat{i}_L)-\frac{V_O+\hat{v}_o}{R}\right]$$

Expanding and ignoring second-order terms:

$$\frac{d\hat{v}_o}{dt}=\frac{1}{C}\left((1-D)\hat{i}_L-\frac{\hat{v}_o}{R}-I_L\hat{d}\right)$$

c- Small-Signal Model in State-Space Form

$$\frac{d}{dt}\begin{bmatrix}\hat{i}_L\\ \hat{v}_o\end{bmatrix}=\begin{bmatrix}0 & -\frac{1-D}{L}\\ \frac{1-D}{C} & -\frac{1}{RC}\end{bmatrix}\begin{bmatrix}\hat{i}_L\\ \hat{v}_o\end{bmatrix}+\begin{bmatrix}\frac{1}{L} & \frac{V_o}{L}\\ 0 & -\frac{I_L}{C}\end{bmatrix}\begin{bmatrix}\hat{v}_i\\ \hat{d}\end{bmatrix}$$

d- Control-to-Output Transfer Function

Eliminating $\hat{I}_L$, the duty-to-output transfer function becomes:

$$G_{vd}(s)=\frac{\hat{V}_o(s)}{\hat{d}(s)}=\frac{V_O}{1-D}\cdot\frac{1-\frac{sL}{R(1-D)^2}}{LCs^2+\frac{L}{R(1-D)^2}s+1}$$

VIII. Modeling of the inverter/rectifier

A three-phase voltage source inverter (VSI) consists of six power switches, typically IGBTs or MOSFETs, arranged in a bridge configuration. The inverter converts a DC voltage source into a set of balanced three-phase AC voltages, which can be applied to an electrical machine or injected into the grid. The idealized circuit representation is shown in the following figure.

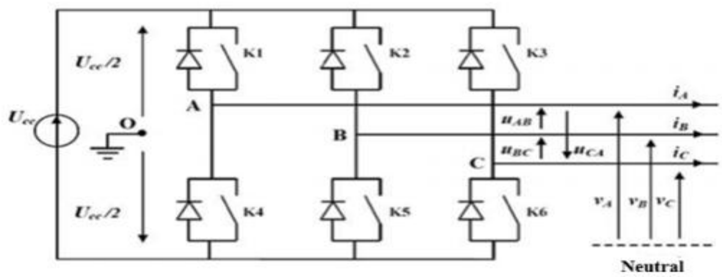


Figure III.12. Inverter equivalent circuit

Assumptions

- Two-level inverter with ideal switches (IGBTs or MOSFETs with antiparallel diodes), neglecting switching losses, dead-times, and voltage drops.
- Balanced three-phase load connected in star (Y), with isolated neutral.
- The model applies to both inverter and rectifier operation (only the power flow direction changes).

Notations

- V_{dc}: DC-link voltage.
- $S_a,S_b,S_c \in \{0,1\}$: switching functions of each inverter leg (1 = upper switch ON, lower switch OFF; 0 = upper switch OFF, lower switch ON).
- v_{xN}: pole voltage of phase $x \in \{a,b,c\}$ with respect to the negative DC bus terminal N.
- v_{x0}: load phase voltage (phase x with respect to the load neutral point 0).
- $v_{xy}=v_{x0}-v_{y0}$: line-to-line voltages.

Each leg generates a binary voltage depending on the switching state:

$$v_{aN} = S_a V_{dc},\ v_{bN} = S_b V_{dc},\ v_{cN} = S_c V_{dc}$$

Introducing the neutral shift voltage v_{0N} :

$$v_{a0} = v_{aN} - v_{0N},\ v_{b0} = v_{bN} - v_{0N},\ v_{c0} = v_{cN} - v_{0N}$$

In a three-phase system where the load is connected in star (Y) and the neutral point is isolated (not connected to the DC bus midpoint or to ground), the neutral potential is not fixed. Instead, it “floats” according to the instantaneous balance of the three phase voltages.

This physical condition imposes the following constraint:

$$v_{a0} + v_{b0} + v_{c0} = 0$$

In other words, at any instant, the sum of the three load phase voltages must equal zero. This relationship ensures that Kirchhoff’s Voltage Law is satisfied around the star connection.

By applying this constraint, the floating neutral potentialv_{0N} can be expressed as:

$$v_{0N} = \frac{v_{aN} + v_{bN} + v_{cN}}{3}$$

By summing and imposing the constraint, we obtain:

$$v_{0N} = \frac{v_{aN} + v_{bN} + v_{cN}}{3} = \frac{V_{dc}}{3}(S_a + S_b + S_c)$$

Substituting into the previous equations, the phase voltages can be expressed in matrix form:

$$\begin{bmatrix} v_{a0} \\ v_{b0} \\ v_{c0} \end{bmatrix} = \frac{V_{dc}}{3} \begin{bmatrix} 2 & -1 & -1 \\ -1 & 2 & -1 \\ -1 & -1 & 2 \end{bmatrix} \begin{bmatrix} S_a \\ S_b \\ S_c \end{bmatrix}$$

This compact matrix representation highlights the dependence of the phase voltages on the switching states of each leg.

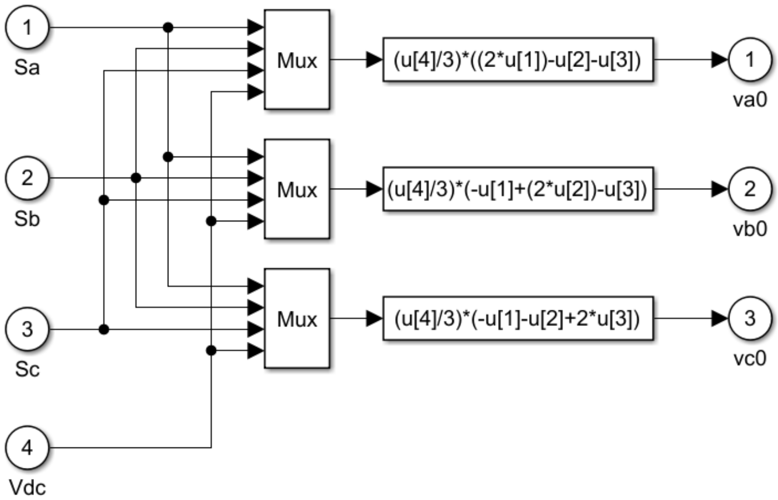


Figure III.13 : Inverter simulink model

IX. Conclusion

In this third chapter, we have extended our exploration of modeling and simulation by focusing on their implementation and application in practice. Beyond the theoretical and methodological foundations established earlier, we examined how models are translated into executable simulations, highlighting the importance of computational environments and specialized tools.

We emphasized the role of validation and verification, ensuring that models not only reproduce expected behaviors but also provide reliable insights for decision-making. The discussion of case studies and practical examples demonstrated how simulation serves as a bridge between abstract representation and real-world problem solving.

Finally, this chapter underscored the necessity of balancing simplicity and accuracy: while models must remain tractable, they should also capture the essential dynamics of the systems they represent. This balance is what makes simulation a powerful tool for engineers, scientists, and managers alike

Chapter 4: Modeling and Simulation of a Wind Energy Conversion System (WECS)

Purpose of the Chapter

This chapter provides a comprehensive framework to understand the interaction between mechanical and electrical subsystems in WECS.

Intermediate Objectives

To develop mathematical models of the different components of a Wind Energy Conversion System (WECS) and to simulate their interaction in order to analyze system behavior under different operating conditions.

Specific Objectives

Upon completing this chapter, students will be able to:

- To model the **aerodynamic conversion** from wind speed to turbine mechanical power.
- To establish the **mechanical dynamics** of the turbine and drive-train, including gearbox effects.
- To model the **electrical generator** (synchronous, induction, or DFIG) in both steady-state and dynamic forms.
- To simulate the complete conversion chain and validate its performance through MATLAB/Simulink.
- To compare the impact of control strategies (e.g., MPPT – Maximum Power Point Tracking, vector control) on the overall system efficiency and dynamic response.

I. Introduction

Wind energy has become one of the most promising renewable energy sources due to its abundance, sustainability, and continuous technological advancements. A Wind Energy Conversion System (WECS) transforms the kinetic energy of the wind into mechanical energy through the wind turbine, and then into electrical energy using a generator. This electrical energy is finally conditioned and adapted by power electronic converters to meet the requirements of either stand-alone or grid-connected applications.

I.1. Definition of Wind Energy

A wind turbine, also referred to as anaerogenerator, is a device that converts part of the kinetic energyof the wind (a moving fluid) into mechanical energy on a drive shaft, and subsequently into electrical energy through a generator (Figure IV.1).

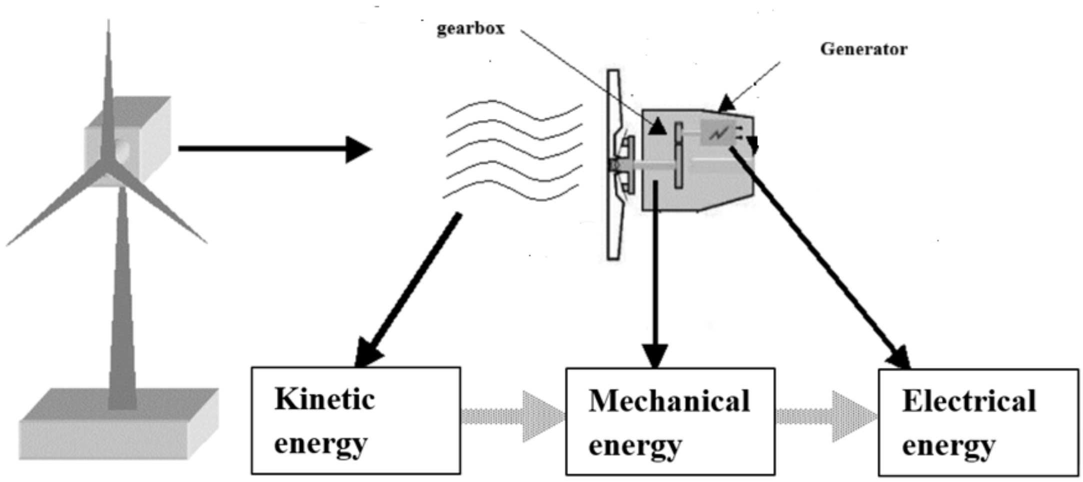


Figure IV.1: Conversion of wind kinetic energy

Wind energy is therefore a process of energy conversion involving three main stages:

1. **Aerodynamic conversion** – the rotor blades capture the kinetic energy of the wind.

2. **Mechanical conversion** – the rotational energy is transmitted through the shaft and, if present, a gearbox.
3. **Electrical conversion** – the generator transforms the mechanical energy into usable electrical power.

This transformation chain makes wind turbines one of the most promising technologies for large-scale renewable energy production.

I.2. Types of Wind Turbines

Wind turbines are commonly classified according to the geometrical arrangement of the shaft on which the rotor blades are mounted. Two main categories can be distinguished: vertical-axis wind turbines (VAWTs)and horizontal-axis wind turbines (HAWTs)**.**

I.2.1. Horizontal-Axis wind turbines (HAWTs).

A horizontal-axis wind turbine has a rotor shaft aligned horizontally and oriented toward the wind, similar to the propellers of airplanes or traditional windmills. It is typically mounted on the top of a tall tower, allowing it to capture a larger fraction of the available wind energy.

Most of the commercially deployed wind turbines worldwide belong to this category. The popularity of HAWTs is largely due to their advantages, such as:

- Low cut-in wind speed, enabling early energy production at relatively low wind conditions.
- High power coefficient (Cp), i.e., a favorable ratio between the extracted power and the kinetic power of the incident air mass.

However, HAWTs also present some limitations: the gearbox and electrical generator must be installed at the top of the tower, which introduces mechanical challenges, increases structural loads, and can lead to higher installation and maintenance costs.

According to the number of blades, a horizontal-axis wind turbine (HAWT) can be classified as single-blade**,** two-blade**,** three-blade**,** ormulti-blade

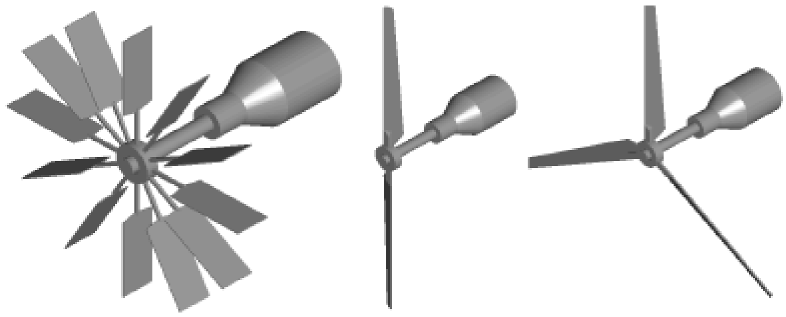


Figure IV.2: Horizontal axis wind turbines

There are two main categories of horizontal-axis wind turbines (HAWTs):

- **Upwind turbines**: In this configuration, the wind strikes the **front side** of the blades before reaching the nacelle. The blades are rigid, and the rotor is actively oriented into the wind direction using a yaw control system.
- **Downwind turbines**: In this case, the wind passes the **nacelle first** and then strikes the **back side** of the blades. The rotor is more flexible and can be self-oriented with respect to the wind direction.

Among these two configurations, the upwind arrangement is the most widely used, particularly for high-power turbines. This is due to severaladvantages:

- Elimination of the need for a tail vane (yaw is actively controlled).
- Reduced maneuvering forces acting on the rotor.
- Better structural stability under high wind conditions.

It should be noted that the blades of horizontal-axis turbines must continuously face the wind. For this reason, wind turbines are equipped with yaw control systems, which automatically adjust the nacelle orientation according to the prevailing wind direction.

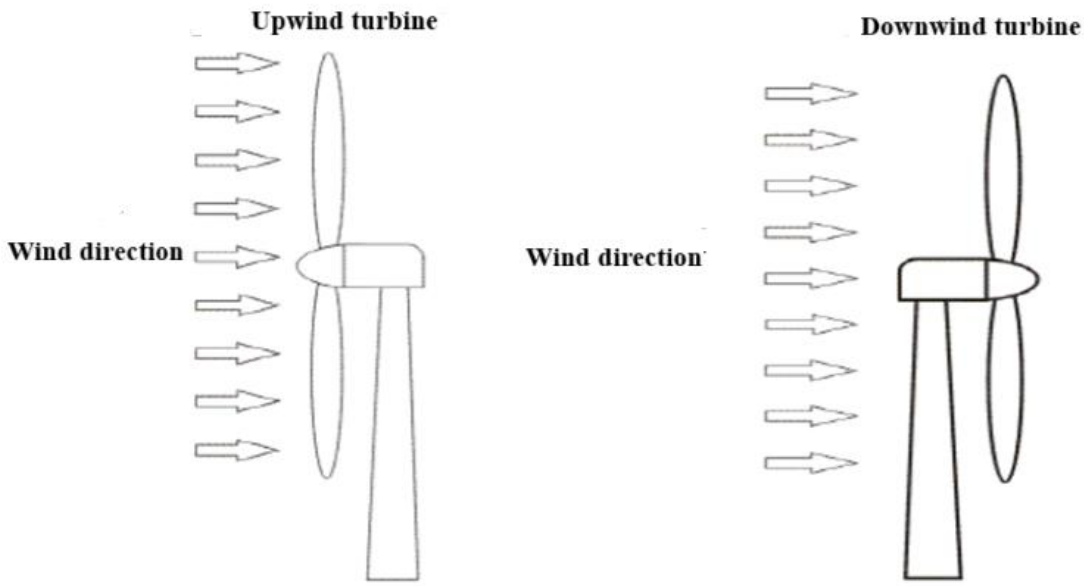


Figure IV.3 :Upwind and downwind turbines

I.2.2. Vertical-Axis wind turbine (VAWT)

The rotation axis of a vertical-axis wind turbine (VAWT) is oriented vertically to the ground and perpendicular to the wind direction. This configuration allows the turbine to receive wind from any direction, eliminating the need for yaw mechanisms. Moreover, both the generator and gearbox can be conveniently located at ground level, which simplifies the design and reduces installation and maintenance costs. Maintenance operations are easier since they are performed on the ground, and VAWTs typically do not require blade pitch control systems like some HAWTs.

Figure IV.4 :Vertical axis wind turbines

However, VAWTs also suffer from several drawbacks:

- Some designs require an auxiliary starting device to initiate rotation.
- Certain VAWTs operate based on aerodynamic drag forces rather than lift forces (which allow airplanes to fly). This results in alower power coefficient and reduced efficiency.
- The majority of VAWTs operate at low rotational speeds, which is problematic for grid-connected electricity generation (50 or 60 Hz). Consequently, the gearbox must provide a high step-up ratio, leading to increased complexity and mechanical losses.

Additionally, two major limitations reduce the competitiveness of VAWTs compared to HAWTs:

1. Their low aerodynamic efficiency, and
2. The reduced wind speeds available close to the ground where they are typically installed.

I.3. Main Components of a Wind Turbine

A wind turbine is generally composed of three main elements:

- The tower,
- The nacelle**,**
- The rotor.

The Tower

The tower is usually made of steel tubular structures or, in some cases, metal lattice frames. Its height must be as large as possible to minimize wind disturbances caused by ground-level turbulence. Increasing tower height enables the turbine to capture stronger and more stable wind flows.

The Nacelle

The nacelle houses all the mechanical and electromechanical components required to couple the rotor to the electrical generator. These include:

- Low-speed and high-speed shafts,
- Bearings **to reduce friction,**
- Gearbox (speed multiplier)**,**
- Control and coolingsystems**,**
- Wind vane **(to provide the yaw system with wind direction),**

- Anemometer **(to measure wind speed and signal when conditions are suitable for efficient electricity production),**
- Disc brake system**, which complements the aerodynamic braking system and ensures safe stopping of the turbine during overload or emergency conditions,**
- Generator**, which is typically either a** synchronous machine **or an** induction machine**,**
- Hydraulic or electrical systems **for** blade pitch control **(aerodynamic brake) and** yaw control **(to keep the rotor swept area perpendicular to the wind direction).**

The Rotor

The rotor consists of the blades mounted on the hub.

- The blades are responsible for capturing wind energy and transferring it into rotational motion.
- The hub serves as the blade support structure. It must withstand significant mechanical stresses, especially during turbine startup or sudden changes in wind speed. For this reason, hubs are typically manufactured as solid cast components rather than welded assemblies, despite the increased manufacturing difficulty

II. Modeling of the Wind Turbine

II.1. Wind Energy Conversion

II.1.1. Power of a wind turbine

The kinetic power of the wind passing through a circular area swept by the turbine blades (of radius *R*) in one second is expressed as:

$$P_{wind} = \frac{1}{2}\rho A V^3$$

where:

- ρ : air density (kg/m³),
- $A=\pi R^2$: swept area of the rotor (m²),
- V : wind speed (m/s).

This relation shows that the available wind power increases linearly with air density and swept area, and cubically with wind speed, which highlights the crucial importance of wind velocity in power production.

II.1.2. Mechanical Power Extracted by the Turbine

Not all the kinetic energy of the wind can be converted into useful power, since part of it must remain in the airflow to allow it to continue moving downstream. The mechanical power extracted by a wind turbine is therefore expressed as:

$$P_t = \frac{1}{2}\rho A V^3 C_p(\lambda,\beta)$$

where:

- $C_p(\lambda,\beta)$: power coefficient, a dimensionless factor that characterizes the efficiency of the turbine in extracting energy from the wind,
- λ : tip-speed ratio (TSR), the ratio between the blade tip speed and the wind speed,
- β : pitch angle of the blades, which adjusts the aerodynamic capture.

II.1.3. Betz Limit

According to **Betz's law**, the theoretical maximum value of C_p is:

$$C_{p,max}=0.593 \quad (59.3\%)$$

This means that no wind turbine can convert more than 59.3% of the kinetic energy of the wind into mechanical energy, regardless of design. In practice, modern turbines reach maximum values of C_p between0.4 and 0.5.

Several numerical approximations have been proposed in the literature to compute the power coefficient C_p, since it depends on both the tip-speed ratio λ and the blade pitch angle β. Different mathematical expressions exist, each offering a trade-off between accuracy and simplicity.

Below, we present four commonly used formulations of the power coefficient $C_p(\lambda,\beta)$

1) $C_p(\lambda,\beta) = [0.5-0.00167(\beta-2)]\sin\left[\frac{\pi(\lambda+0.1)}{18.5-0.3(\beta-2)}\right] - 0.00814(\lambda-3)(\beta-2)$
2) $C_p(\lambda,\beta) = C_1\left(C_2\left(\frac{1}{\lambda+0.08\beta} - \frac{0.0035}{\beta+1}\right)C_3\beta - C_4\right)\mathrm{Exp}\left(-C_5\left(\frac{1}{\lambda+0.08\beta} - \frac{0.0035}{\beta+1}\right)\right) + C_6\lambda$

Where : $C_1 = 0.5109$; $C_2 = 116$; $C_3 = 0.4$; $C_4 = 5$; $C_5 = 21$; $C_6 = 0.0068$;

3) $C_p(\lambda,\beta) = 0.22\left(\frac{116}{\lambda'} - 0.4\beta - 5\right)e^{\frac{-12.5}{\lambda'}}$ et $\frac{1}{\lambda'} = \frac{1}{\lambda+0.08\beta} - \frac{0.035}{\beta+1}$
4) $C_p(\lambda,\beta) = 7.95633\ 10^{-5}\lambda^5 - 17.375\ 10^{-4}\lambda^4 + 9.86\ 10^{-3}\lambda^3 - 9.4\ 10^{-3}\lambda^2 + 6.38\ 10^{-2}\lambda + 0.001$

With :

β: Blade Pitch Angle

The angle at which the blades are oriented with respect to the incoming wind. It is a crucial control variable that regulates the amount of aerodynamic power captured by the rotor.

λ: Tip-Speed Ratio (TSR)

Defined as the ratio between the linear speed of the blade tips and the wind speed. It characterizes the aerodynamic efficiency of the turbine and is expressed as:

$$\lambda = \frac{\Omega_{turbine}.R}{V_{wind}}$$

$\Omega_{turbine}$: Turbine Speed

The rotational speed of the turbine expressed in radians per second. It determines both the tip-speed ratio and the mechanical torque developed by the rotor.

II.1.4. Aerodynamic Torque of the Wind Turbine

The mechanical (aerodynamic) torque delivered by the rotor is obtained from the extracted power:

$$T_t = \frac{P_t}{\Omega_{turbine}} = \frac{\frac{1}{2}\rho A V^3 C_p(\lambda,\beta)}{\Omega_{turbine}}$$

Using:

$$\lambda = \frac{\Omega_{turbine}.R}{V_{wind}}$$

This can be written in the more convenient forms:

$$T_t = \frac{1}{2}\rho A V^2 \underbrace{\frac{C_p(\lambda,\beta)}{\lambda}}_{C_q(\lambda,\beta)}$$

Where $C_q(\lambda,\beta)$: is the torque coefficient.

Notes

- T_t scales with V^2_{wind} (for fixed λ), while P_t scales with V^3_{wind}.

- At very low speed ($\Omega_{turbine} \rightarrow 0$), use the C_q form to avoid division by $\Omega_{turbine}$ in simulations.
- For a given blade design, $C_q(\lambda,\beta)$ reaches a maximum near the optimal λopt used for MPPT.

II.1.5. Mechanical Subsystem Modeling

In this section, we consider a three-blade wind turbine. The blades are mounted on a main shaft, which transfers the aerodynamic torque to a gearbox with a transmission ratio GGG. The gearbox, in turn, drives the generator shaft through a high-speed shaft.

The overall mechanical model of the wind energy conversion system can therefore be decomposed into three interconnected subsystems:

- The Turbine
- The Gearbox (Multiplier)
- The Generator Shaft

a. Turbine Modeling

The dynamic study established previously shows that, for a given wind speed V_{wind} crossing the rotor swept area, the aerodynamic torque applied to the turbine shaft can be expressed as:

$$T_t = \frac{1}{2}\rho A V_{wind}{}^2 \frac{C_p(\lambda,\beta)}{\lambda}$$

This equation directly links the mechanical torque generated by the wind turbine to the aerodynamic parameters (λ,β) and the environmental condition (V_{wind}).

For the considered wind turbine, the power coefficient C_p is defined by the following empirical relation:

$$C_p(\lambda,\beta) = [0.5 - 0.00167(\beta-2)]\cdot\sin\left(\frac{\pi(\lambda+0.1)}{18.5-0.3(\beta-2)}\right) - 0.00814(\lambda-3)(\beta-2)$$

This nonlinear expression captures the combined influence of the tip-speed ratio and thepitch angleon the turbine’s aerodynamic performance. The function provides a realistic representation of the turbine’s efficiency curve, with a well-defined maximum at the optimal operating point(λ_{opt},β_{opt}), which is typically exploited by MPPT (Maximum Power Point Tracking) control strategies.

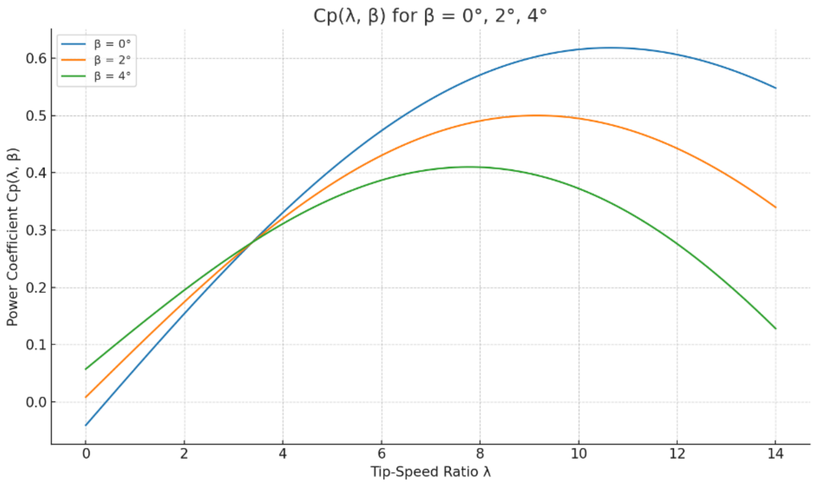


Figure IV.5 :Power coefficient $C_p$versusλ and β

The figure shows the variation of the power coefficient C_p as a function of the tip-speed ratio λ for three different blade pitch angles (β=0°,2°,4°).

- For β=0°, the turbine achieves its highest efficiency, with a maximum $C_p\approx0.48$at an optimal tip-speed ratio $\lambda\approx7$.
- When the pitch angle increases (β=2°,4°), both the maximum valueof C_p and the corresponding optimal λ decrease. This indicates that blade pitch introduces aerodynamic losses and reduces the turbine's efficiency.
- The shift and reduction of the C_p peak explain why active pitch control is typically used: at low wind speeds, β is kept near zero to maximize energy capture, while at high wind speeds, β is increased intentionally to limit the aerodynamic power and protect the turbine.

Thus, the Cp–λ characteristics are fundamental for defining the maximum power point tracking (MPPT) strategy and for understanding how pitch control regulates turbine performance.

b. Gearbox Modeling

The gearbox adapts the low turbine speed to the higher generator speed.Assuming an ideal (lossless) gearbox, the kinematic and torque relations are:

$$\Omega_g = G.\Omega_t$$
$$T_g = \frac{T_t}{G}$$

Where:

G: gearbox (multiplier) ratio.

T_g : torque at the generator (high-speed) shaft.

T_t: aerodynamic torque at the turbine shaft.

Ω_g: generator angular speed (rad/s).

Ω_t: turbine angular speed (rad/s).

c. Mechanical model

The proposed mechanical model considers the total inertiaJ, obtained by referring the turbine inertia to the high-speed (generator) side and adding the generator inertia:

$$J = \frac{J_t}{G^2} + J_{mec}$$

where

$J_{turbine}$: turbine inertia (referred to the turbine shaft),

J_{mec} : generator (mechanical) inertia referred to the generator shaft,

G : gearbox ratio (speed multiplier, Ω_g=G Ω_t).

$\Omega_g = \Omega_{mec}$

Applying the fundamental equation of rotational dynamics to the generator (fast) shaft gives:

$$J\frac{d\Omega_{gen}}{dt} = T_{mec} - T_{em} - T_{vis}$$

Where :

T_{mec}: mechanical torque transmitted through the gearbox (from the turbine),

T_{em}: electromagnetic torque developed by the electrical machine,

T_{vis}: viscous friction torque (mechanicallosses).

The viscous friction torque is assumed proportional to the rotational speed:

$$T_{vis} = f\Omega_{gen}$$

Where

f : the viscous friction coefficient of the rotating assembly.

II.1.6. Global Model of the Wind Energy Conversion System

From the previously established equations, the overall dynamics of the wind turbine, gearbox, and generator can be represented in the form of a block diagram.

The global model links the wind speed V_{wind} to the electromagnetic torque T_{em} of the generator through the following chain of relations:

- Aerodynamic Power and Torque (turbine):

$$P_t = \frac{1}{2}\rho A C_p(\lambda, \beta) V_{wind}^3, \quad T_t = \frac{P_t}{\Omega_t}$$

- Gearbox Relations (transformation to generator shaft):

$$\Omega_{gen} = G\Omega_t, \quad T_{mec} = \frac{T_t}{G}$$

- Mechanical Dynamics of the Generator Shaft (with total inertia and friction):

$$J\frac{d\Omega_{gen}}{dt} = T_{mec} - T_{em} - f\Omega_{gen}$$

These relations allow the global block diagram of the wind energy conversion system (WECS) to be constructed, where the input is the wind speed V_{wind}, and the outputs are the mechanical speed Ω_{gen} and the electromagnetic torque T_{em}.

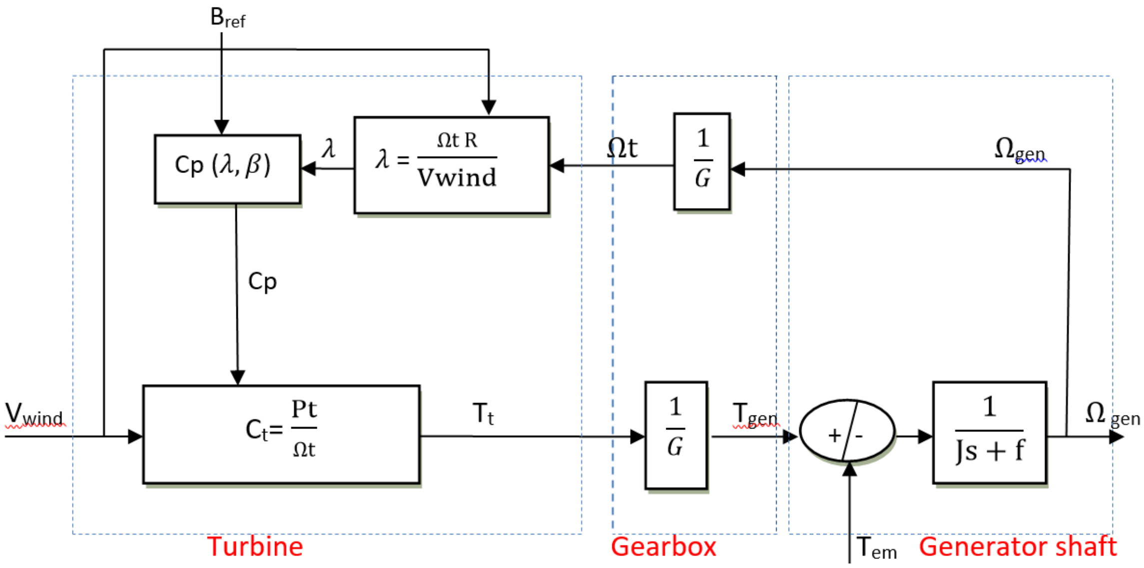


Figure IV.6 :Block diagram of the wind turbine model

III. Algorithms for Maximum Power Extraction

III.1. Optimization of Power Extraction

The optimal operation of a wind turbine is achieved when the power coefficient C_p reaches its maximum value. This maximum occurs for a specific tip speed ratio $\lambda=\lambda_{opt}$. Consequently, the optimal rotational speed of the turbine can be expressed as:

$$\Omega_{turbine} = \frac{\lambda.V_{wind}}{R}$$

where:

λ_{opt} = optimal tip speed ratio,

V_{wind} = wind speed (m/s),

R = turbine radius (m).

The variation of the optimal turbine speed $\Omega_{turbine}$, with respect to wind speed is shown in Figure IV.7. Based on physical and technological constraints, the operation of the wind turbine can be divided into three zones:

- **Zone 1**: Corresponds to very low wind speeds, which are insufficient to rotate the turbine. No useful power is extracted in this region.
- **Zone 2**: This is the linear operating region, directly following relation (1.34). In this zone, a Maximum Power Point Tracking (MPPT) algorithm is employed to ensure that the turbine operates at its optimal tip speed ratio, thereby extracting maximum available power. The algorithm ensures that the operating point follows the optimal curve illustrated in Figure 33.
- **Zone 3**: Corresponds to high wind speeds. In this region, the turbine's rotational speed is intentionally limited to a maximum allowable value in order to protect the turbine and generator from mechanical and electrical damage.

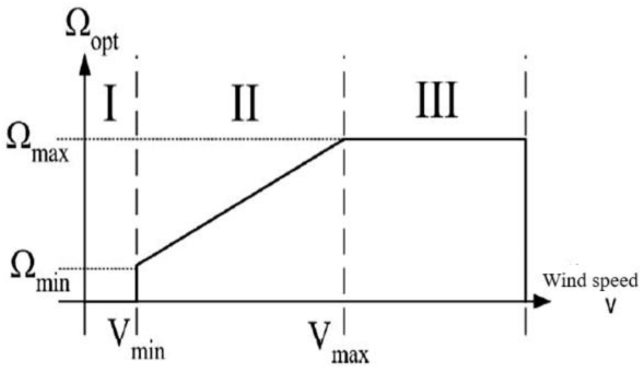


Figure IV.7. Variable speed wind turbine control law

III.1.1. Algorithms for Maximizing Extracted Power

The main objective of variable-speed control is to extract the maximum possible wind power. To achieve this, a dedicated algorithm must act on the reference variables so that the studied system operates at its highest efficiency.

In practice, the rotational speed of the turbine is controlled through the electromagnetic torque. Two control strategies are commonly distinguished in the literature:

- **Control with mechanical speed feedback** (closed-loop speed control),
- **Control without mechanical speed feedback** (sensorless or indirect control).

a. Control with Mechanical Speed Feedback

This approach aims to determine the turbine speed that ensures maximum generated power. From the relation, which links the turbine speed to the wind speed for maximum C_p, a reference mechanical speed can be calculated. The control structure is then designed to track this reference speed, ensuring that the turbine consistently operates near the maximum power point.

The reference electromagnetic torque is computed as:

$$T_{em\_ref} = REG(\Omega_{mec\_ref} - \Omega_{mec})$$

where:

REG is the speed controller,

Ω_{mec} is the reference mechanical speed.

This reference speed depends on the desired turbine speed $\Omega_{turbine,ref}$, which is set to maximize the extracted power. Taking into account the gearbox ratio G, we obtain:

$$\Omega_{mec\_ref} = G.\Omega_{turbine\_ref}$$

The following figure illustrates the block diagram of maximum power extraction using mechanical speed feedback control:

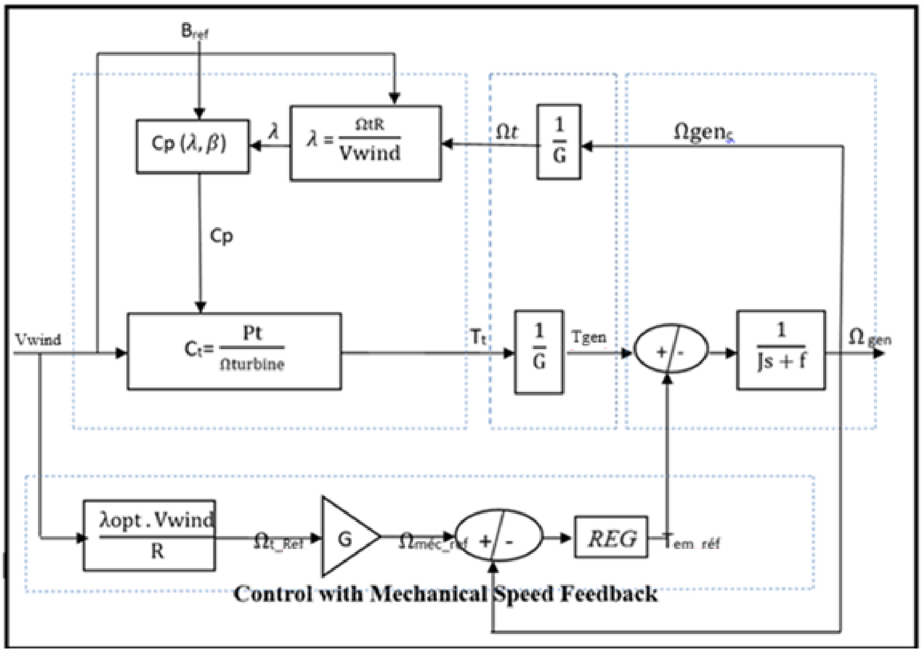


Figure IV.8: Block diagram of power maximization with mechanical speed control

b. Control without Mechanical Speed Feedback

Accurately measuring the wind speed is challenging because it is a highly fluctuating variable. An inaccurate measurement may lead to imposing an unsuitable reference value, resulting in degraded captured power.

To establish this control structure, it is assumed that wind speed varies very little in steady-state operation.

From the turbine dynamics, the steady-state equation of the wind generator can be written as:

$$J\frac{d\Omega_{mec}}{dt} = \sum T = 0 = T_{mec} - T_{em} - T_{vis}$$

If the viscous torque is neglected ($T_{visc} \approx 0$), we obtain:

$$T_{mec} = T_{em}$$

Thus, the reference electromagnetic torque is estimated based on the aerodynamic torque:

$$T_{mec\_ref} = \frac{T_{aero}}{G}$$

The aerodynamic torque is itself estimated as a function of wind speed and turbine speed:

$$T_{aero,est} = \frac{1}{2}\rho C_p(\lambda,\beta) S \frac{V_{vent,est}^3}{\Omega_{turbine}}$$

The estimated turbine speed is derived from the measured mechanical speed as:

$$\Omega_{turbine\_ref} = \frac{\Omega_{mec}}{G}$$

Since direct measurement of wind speed is difficult, it can be estimated as:

$$V_{vent\_est} = \frac{\Omega_{turbine\_est}.R}{\lambda}$$

From these relations, the global torque control law can be expressed as:

$$T_{em,ref} = \frac{1}{2} C_p(\lambda,\beta)\rho\pi R^5 \frac{\Omega_{mec}^2}{G^3\lambda^3}$$

To extract the maximum available power, the tip speed ratio must be fixed at its optimal value λ_{opt}, which corresponds to the maximum power coefficient $C_{p,max}$.

Therefore, the reference electromagnetic torque must be adjusted to:

$$T_{em,ref} = \frac{1}{2} C_{p,max} \rho \pi R^5 \frac{\Omega_{mec}^2}{G^3 \lambda_{opt}^3}$$

The block diagram representation of this control method is shown in the following figure:

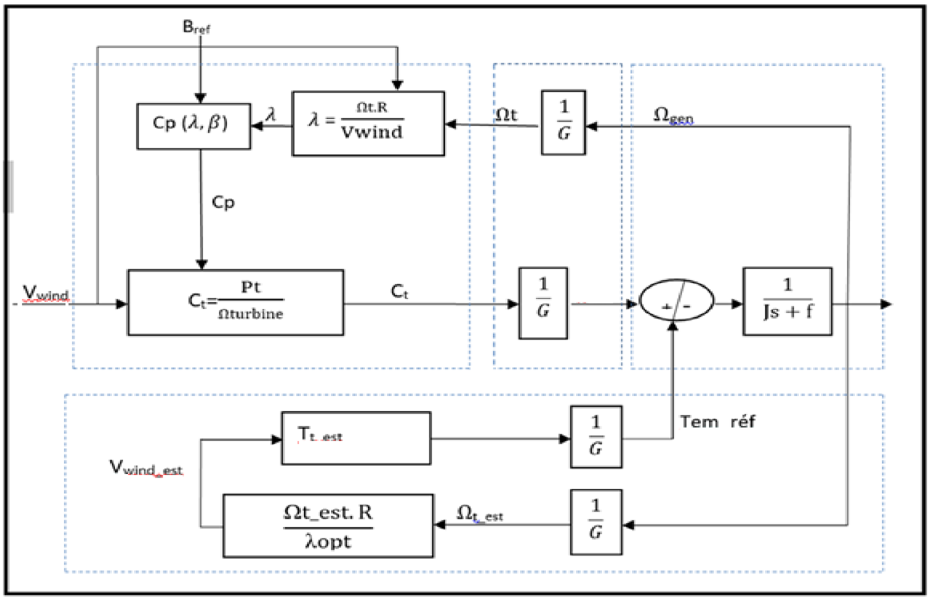


Figure IV.9: Block diagram of power maximization without mechanical speed control

III.2. Pitch Control System

The pitch control system is used to adjust the blade pitch angle in order to maintain the output power close to its rated value. By regulating the pitch angle, the aerodynamic performance of the wind turbine is modified, specifically affecting the power coefficient C_p.

The pitch control system generates a reference pitch angle β_{ref}. In practice, the actuator dynamics are usually represented by a first-order transfer function:

$$\beta(s) = \frac{1}{\tau s + 1} \beta_{ref}(s)$$

Where

τ : the time constant of the pitch actuator.

When modeling the pitch system, it is essential to consider the rate of change of the pitch angle. Due to the mechanical stress experienced by the blades, the pitch rate is generally limited to:

$$\frac{d\beta}{dt} \leq 10^{\circ} / s$$

Moreover, the maximum pitch angle is limited to 90°, which corresponds to the feathered position of the blades.

By adjusting the pitch angle, the turbine performance is modified. At low wind speeds, the blades are aligned directly with the wind (β=0°) to maximize aerodynamic efficiency. At high wind speeds, the blades are pitched to reduce the aerodynamic lift and degrade the power coefficient C_p, thus preventing mechanical overload and maintaining safe operation.

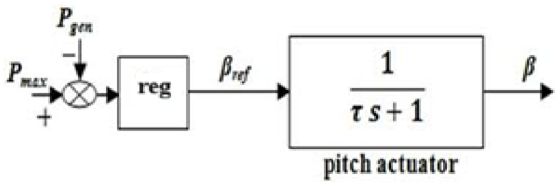


Figure IV.10: Pitch control

IV. Modeling of Squirrel-Cage Induction Machines (SCIM)

IV.1. Introduction

Induction machines (IM), particularly the squirrel-cage type, represent the backbone of modern electrical drives and energy conversion systems. They are widely used due to their robustness, low cost, and minimal maintenance requirements compared to synchronous machines.

In wind energy systems, the induction generator—often in squirrel-cage form—plays a central role in converting mechanical energy from the turbine into electrical energy. Understanding the machine's mathematical model is therefore essential for dynamic analysis, control design, and system optimization.

IV.2. Three-Phase Model in abc Variables

Assumptions: Magnetic linearity, sinusoidal distribution, balanced system, and neglect of core losses.

Stator voltage equations:

$$\begin{bmatrix} v_{as} \\ v_{bs} \\ v_{cs} \end{bmatrix} = \begin{bmatrix} R_s & 0 & 0 \\ 0 & R_s & 0 \\ 0 & 0 & R_s \end{bmatrix} \begin{bmatrix} i_{as} \\ i_{bs} \\ i_{cs} \end{bmatrix} + \frac{d}{dt} \begin{bmatrix} \varphi_{as} \\ \varphi_{bs} \\ \varphi_{cs} \end{bmatrix}$$

Rotor voltage equations (squirrel cage → short-circuited):

$$\begin{bmatrix} v_{ar} \\ v_{br} \\ v_{cr} \end{bmatrix} = \begin{bmatrix} R_r & 0 & 0 \\ 0 & R_r & 0 \\ 0 & 0 & R_r \end{bmatrix} \begin{bmatrix} i_{ar} \\ i_{br} \\ i_{cr} \end{bmatrix} + \frac{d}{dt} \begin{bmatrix} \varphi_{ar} \\ \varphi_{br} \\ \varphi_{cr} \end{bmatrix}$$

With

$v_{ar}=v_{br}=v_{cr}=0$

IV.3. Flux-Current Relationships in abc

The relationships between Flux and current are given by the following equation:

$$\begin{bmatrix} \varphi_s \\ \varphi_r \end{bmatrix} = \begin{bmatrix} L_s & M_{sr} \\ M_{rs} & L_r \end{bmatrix} . \begin{bmatrix} I_s \\ I_r \end{bmatrix}$$

IV.4. Concordia Transformation (abc → αβ)

The Concordia Transformation from abc toαβ reference is:

$$[X_{\alpha\beta h}] = \begin{bmatrix} 1 & -\frac{1}{2} & -\frac{1}{2} \\ 0 & \frac{\sqrt{3}}{2} & -\frac{\sqrt{3}}{2} \\ \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \end{bmatrix} [X_{abc}]$$

IV.5. Park Transformation (αβ → dq)

To get dq equations Park transformation is applied as follow:

$$\left[X_{dqh}\right]=\begin{bmatrix}\cos(\psi) & \sin(\psi) & 0\\ -\sin(\psi) & \cos(\psi) & 0\\ 0 & 0 & 1\end{bmatrix}\left[X_{\alpha\beta h}\right]$$

dq Voltage Equations:

$$\begin{cases} v_{ds} = R_s i_{ds} + \varphi_{ds}^{'} - \omega_s \varphi_{qs} \\ v_{qs} = R_s i_{qs} + \varphi_{qs}^{'} + \omega_s \varphi_{ds} \\ 0 = R_r i_{dr} + \varphi_{dr}^{'} - \omega_r \varphi_{qr} \\ 0 = R_r i_{qr} + \varphi_{qr}^{'} + \omega_r \varphi_{dr} \end{cases}$$

Flux-Current Relationships in dq

$$\begin{cases} \varphi_{ds} = L_s I_{ds} + MI_{dr} \\ \varphi_{qs} = L_s I_{qs} + MI_{qr} \\ \varphi_{dr} = L_r I_{dr} + MI_{ds} \\ \varphi_{qr} = L_r I_{qr} + MI_{qs} \end{cases}$$

$$\begin{bmatrix} I_{ds} \\ I_{qs} \\ I_{dr} \\ I_{qr} \end{bmatrix} = \frac{1}{\sigma L_s L_r}\begin{bmatrix} L_r & 0 & -M & 0 \\ 0 & L_r & 0 & -M \\ -M & 0 & L_s & 0 \\ 0 & -M & 0 & L_s \end{bmatrix}\begin{bmatrix} \varphi_{ds} \\ \varphi_{qs} \\ \varphi_{dr} \\ \varphi_{qr} \end{bmatrix}$$

Electromagnetic Torque Equation

$$T_{em} = p\left[i_{qs}\varphi_{ds} - i_{ds}\varphi_{qs}\right]$$

Mechanical Dynamics

$$J\frac{d\Omega}{dt} = T_{mec} - T_{em} - f\Omega$$

IV.6. Equivalent dq Model

The dq model is typically represented by a block diagram including resistances, leakage inductances, mutual inductances, and flux/current interactions.

IV.7. Application in Wind Energy Systems

In wind turbine applications:

- SCIMs can operate directly connected to the grid (fixed speed).
- With power electronics, they can operate at variable speed, enabling MPPT.
- dq modeling is essential for vector control strategies.

IV.8. Conclusion

Modeling in (dq) provides a compact and powerful representation of the SCIM. It is the foundation for advanced control techniques. Essential in renewable energy integration (wind turbine systems).

V. Modeling of double fed Induction generator (DFIG)

V.1. Introduction

The Doubly-Fed Induction Generator (DFIG) is one of the most widely used technologies in modern wind energy conversion systems. It allows variable-speed operation while keeping the grid frequency constant. The stator is directly connected to the grid, while the rotor is fed through slip rings and a back-to-back converter. This design reduces converter rating (typically 25–30% of rated power), lowering costs and improving efficiency.

V.2. Three-Phase Model in abc Variables

The DFIG can be modeled in the abc frame using stator and rotor voltage equations. These equations describe resistive drops and flux derivatives. The rotor voltages are expressed relative to the rotor reference frame, which rotates at rotor speed ω_r.

The schematic representation of the double-fed asynchronous machine in the three-phase coordinate system is shown in the following figure.

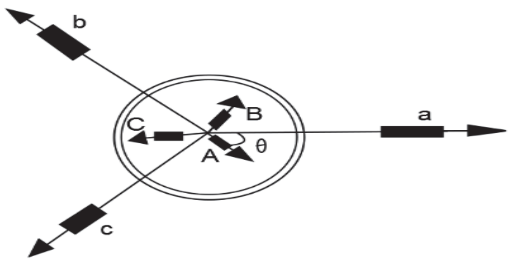


Figure IV.11: Schematic representation of DFIG

Stator voltage equations:

$$\begin{bmatrix} v_a \\ v_b \\ v_c \end{bmatrix} = \begin{bmatrix} R_s & 0 & 0 \\ 0 & R_s & 0 \\ 0 & 0 & R_s \end{bmatrix} \begin{bmatrix} i_a \\ i_b \\ i_c \end{bmatrix} + \frac{d}{dt} \begin{bmatrix} \varphi_a \\ \varphi_b \\ \varphi_c \end{bmatrix}$$

Rotor voltage equations:

$$\begin{bmatrix} v_A \\ v_B \\ v_C \end{bmatrix} = \begin{bmatrix} R_r & 0 & 0 \\ 0 & R_r & 0 \\ 0 & 0 & R_r \end{bmatrix} \begin{bmatrix} i_A \\ i_B \\ i_C \end{bmatrix} + \frac{d}{dt} \begin{bmatrix} \varphi_A \\ \varphi_B \\ \varphi_C \end{bmatrix}$$

V.3. Need for Park Transformation

In the abc frame, voltages and currents are sinusoidal and time-varying, making analysis complex. The Park transformation converts these three-phase sinusoidal variables into DC quantities in the dq synchronous frame. This simplifies modeling by removing time-varying inductances and enables advanced control strategies.

Voltage equations in dq reference:

$$\begin{cases} v_{ds} = R_s i_{ds} + \varphi_{ds}^{'} - \omega_s \varphi_{qs} \\ v_{qs} = R_s i_{qs} + \varphi_{qs}^{'} + \omega_s \varphi_{ds} \\ v_{dr} = R_r i_{dr} + \varphi_{dr}^{'} - \omega_r \varphi_{qr} \\ v_{dr} = R_r i_{qr} + \varphi_{qr}^{'} + \omega_r \varphi_{dr} \end{cases}$$

Flux linkages are given by:

$$\begin{cases} \varphi_{ds} = L_s I_{ds} + MI_{dr} \\ \varphi_{qs} = L_s I_{qs} + MI_{qr} \\ \varphi_{dr} = L_r I_{dr} + MI_{ds} \\ \varphi_{qr} = L_r I_{qr} + MI_{qs} \end{cases}$$

Electromagnetic Torque Equation

$$T_{em} = p\left[i_{qs}\varphi_{ds} - i_{ds}\varphi_{qs} \right]$$

Mechanical Dynamics

$$J\frac{d\Omega}{dt} = T_{mec} - T_{em} - f\Omega$$

V.4. Application in Wind Energy

The DFIG is ideal for variable-speed wind turbines since it enables maximum power extraction while maintaining synchronous grid frequency. Its converter rating is lower than that of PMSG systems, which reduces costs. However, the use of slip rings increases maintenance requirements.

VI. Modeling of Permanent Magnet Synchronous Generator (PMSG)

VI.1. Introduction

Permanent Magnet Synchronous Generators (PMSG) are increasingly popular in direct-drive wind turbines. They eliminate the need for slip rings and external excitation systems, as permanent magnets provide the rotor field. This leads to higher efficiency and lower maintenance, but the high cost of rare-earth magnets remains a challenge.

VI.2. Three-Phase abc Model

The stator equations in the abc frame are:

$$\begin{bmatrix} v_{as} \\ v_{bs} \\ v_{cs} \end{bmatrix} = \begin{bmatrix} R_s & 0 & 0 \\ 0 & R_s & 0 \\ 0 & 0 & R_s \end{bmatrix} \begin{bmatrix} i_{as} \\ i_{bs} \\ i_{cs} \end{bmatrix} + \frac{d}{dt} \begin{bmatrix} \varphi_{as} \\ \varphi_{bs} \\ \varphi_{cs} \end{bmatrix}$$

VI.3. Need for Park Transformation

In the abc frame, the voltages and currents are sinusoidal and time-dependent. Applying the Park transformation aligns the d-axis with the rotor permanent magnet flux. As a result, the flux along the d-axis is constant, greatly simplifying control and analysis.

Voltage equations in dq reference

$$\begin{cases} v_{ds} = R_s i_{ds} + \varphi_{ds}^{'} - \omega_s \varphi_{qs} \\ v_{qs} = R_s i_{qs} + \varphi_{qs}^{'} + \omega_s \varphi_{ds} \end{cases}$$

Flux Equations

$$\begin{cases} \varphi_{ds} = L_d I_d + \varphi_f \\ \varphi_{qs} = L_q I_q \end{cases}$$

Electromagnetic Torque Equation

$$T_{em} = p\left[i_q \varphi_d - i_d \varphi_q \right]$$

Mechanical Dynamics

$$J \frac{d\Omega}{dt} = T_{mec} - T_{em} - f\Omega$$

VI.4. Application in Wind Energy

PMSGs are especially suitable for direct-drive wind turbines. They remove the gearbox, reducing mechanical complexity and improving reliability. A full-scale power converter is required to interface with the grid, enabling flexible control of both active and reactive power. PMSG systems offer high efficiency, making them attractive despite their higher cost.

VII. Modeling of a Hybrid Wind–Photovoltaic (PV) Power System

VII.1. System overview (functional blocks)

A typical grid-connected or stand-alone hybrid system includes the following blocks:

Wind subsystem

- Wind turbine (aerodynamics → mechanical torque)
- Drive train / gearbox
- Electrical machine (DFIG, SCIM or PMSG) + generator control / converter (partial or full power converter)

PV subsystem

- PV array (series/parallel strings)
- DC–DC converter (MPPT — boost or buck–boost)

DC link / DC bus

- Common DC bus (linking PV converter, rotor converter if applicable, battery and inverter)
- DC-link capacitor and control (voltage regulation)

Energystorage (optional)

- Battery model (Thevenin / equivalent circuit)
- Battery DC–DC charger/discharger

Power electronics&grid interface

- DC–AC inverter (grid-tied or islanded)
- Filters (L or L–C)
- Grid synchronization (PLL) and current control

Energy Management System (EMS)

- Supervisory logic deciding power flow (PV → load/grid, wind → load/grid, battery charging/discharging)
- Priorities, constraints, and MPPT coordination

Loads

- AC load (or grid), possibly DC loads

The hybrid operates by coordinating MPPT for PV and wind, storing excess energy in battery, and injecting remaining power to the load or grid

VII.2. Wind subsystem model (summary)

(You can reuse the turbine / gearbox / generator models from your wind chapter; here's the compact chain)

The expression of aerodynamic power and torque

$$P_t = \frac{1}{2}\rho A V_{wind}^3 C_p(\lambda, \beta)$$
$$T_t = \frac{P_t}{\Omega_t}$$

The expression of gearbox (ideal)

$$\Omega_{gen} = G\Omega_t, \quad T_{em} = \frac{T_t}{G}$$

The expression of mechanical dynamics (generator shaft)

$$J\frac{d\Omega}{dt} = T_{mec} - T_{em} - f\Omega$$

The expression of electrical machine

Choose one: DFIG (partial converter), PMSG (full converter), or SCIM (with full converter for variable speed). Use the dq model syoual ready have:

- stator/rotor dq equations, flux linkages, torque equation T_{em}.
- converter control: rotor-side & grid-side for DFIG, full converter (back-to-back) for PMSG.

Implementation note: the wind side provides a reference torque or speed computed by MPPT; the generator / converter enforces that torque/speed via current control.

VII.3. PV subsystem model

Single-diode (one-cell) PV model (fundamental equations)

For a cell:

$$I = I_{ph} - I_0[\exp(\frac{(V + IR_s)}{nT}) - 1] - \frac{V + IR_s}{R_p}$$

For an array with Ns cells in series and Np in parallel:

$$V_{array} = N_s . V_{cell},$$

$$I_{array} = N_p . I_{cell}$$

PV DC–DC converter (MPPT)

- Typical converter: boost (step up) or buck-boost (for wide operating range).
- MPPT algorithms: Perturb & Observe (P&O), Incremental Conductance (IncCond), or more advanced (extremum seeking, hill-climb, model-based).
- The MPPT provides a duty cycle D_{pv} to operate the PV at V_{mp} or the voltage corresponding to maximum power.

VII.4. Battery / storage model

Simple Thevenin 1st-order model

Open-circuit voltage $V_{oc}$and internal resistor R_{int}, plus RC dynamics for transient behavior:

$$V_{battery} = V_{oc}(SoC) - I_{batt}R_{int} - V_{RC}$$

$$\dot{V}_{RC} = -\frac{1}{R_c C_c}V_{RC} + \frac{1}{C_c}I_{batt}$$

$$\dot{SoC} = -\frac{I_{batt}}{Q_{nom}}$$

VII.5. DC bus & power balance

Let DC bus voltage V_{dc} be regulated. Power flows:

- $P_{PV,dc} = V_{pv}I_{pv}$
- $P_{wind,dc}$: power extracted from the generator via converter (rotor or full converter)
- P_{bat} (battery charge/discharge)
- P_{inv} (inverter power to AC load/grid)
- P_{loss} (converter & system losses)

Energy balance on DC bus:

$$C_{dc}\frac{dV_{dc}}{dt} = \frac{1}{V_{dc}}(P_{PV,dc} + P_{wind,dc} - P_{inv} - P_{bat} - P_{loss})$$

DC bus control keeps V_{dc} atsetpoint via inverter or battery converter actions.

VII.6. Grid interface: inverter model

Inverter + L / L–C filter

A three-phase inverter controlled with PWM.

Control

- Grid-tied: current control in dq frame (synchronous to grid via PLL). Outer loop regulates active power (P) or DC-bus voltage; inner loop controls AC currents to track P & Q references.
- Islanded: voltage & frequency control (VSI voltage control).

VII.7. Supervisory energy management (EMS)

EMS decides how to share power among PV, wind, battery and grid/load:

Typical priorities (example):

1. Meet local load from PV & wind (instantaneous).
2. If generation > load: charge battery (if $S_{oC}<S_{oC\_max}$) else export to grid.
3. If generation < load: discharge battery (if $S_{oC}>S_{oC\_min}$) else import from grid or curtail loads.

MPPT coordination: both PV and wind controllers attempt MPPT independently unless battery constraints or grid constraints require curtailment. EMS can override MPPT to maintain DC link or to meet grid codes.

VII.8. Control strategies — details

PV MPPT

- **P&O**: perturb voltage, observe change in power; simple but oscillates at steady state.
- **IncCond**: computes dI/dV and compares to -I/V for MPP; better at changing irradiance.
- **Implementation**: controller outputs duty cycle for DC–DC converter.

Wind MPPT

Two main approaches:

- **Rotor speed control** (speed-feedback): compute $\Omega_{turbine,opt}=(\lambda_{opt}V_{wind})/R$(if V_{wind} measured or estimated), make generator follow Ω_{opt} with torque controller/regulator.
- **Power control / torque control without wind sensor** (sensorless): compute reference torque from measured turbine/generator speed using equation:

$$T_{em,ref} = \frac{1}{2} C_p(\lambda,\beta)\rho\pi R^5 \frac{\Omega_{mec}^2}{G^3\lambda^3}$$

and set $\lambda=\lambda_{opt}$

Generator/converter control

- **DFIG:** rotor-side converter controls rotor currents to manage active/reactive power; grid-side converter maintains DC-link and grid injection.
- **PMSG:** full converter controls stator currents in dq frame; outer speed or torque loop for MPPT, inner current loops for torque.
- **SCIM with full converter:** functionally similar to PMSG but with field dynamics different.

DC bus voltage regulation

- In grid-tied systems, inverter may regulate DC bus by adjusting $P_{\_ref}$; battery converter used to stabilize DC bus in islanded systems.

VII.9. Simulation implementation notes

- **Model granularity:** start with averaged converter models (no switching) for faster simulation and easier control design. Use switching models only when assessing harmonics, or detailed switching losses.

- **Solvers:** use variable-step stiff solvers for detailed models; for averaged ODEs ode23t/ode23tb (MATLAB) or appropriate Simulink solvers.
- **Sample times:** control loops (inner current loops) require much faster sampling than outer MPPT/EMS loops.
- **Initial conditions:** set initial SoC, DC bus voltage, generator speed consistent with power balance to avoid large transients.
- **Clamping& saturation:** enforce duty cycle limits, current limits, SoC bounds.
- **Noise & disturbances:** include realistic irradiance and wind profiles for test cases.

VII.10 Recommended reading / blocks to use in Simulink

- Use Simscape Electrical blocks: PV array (built-in), controlled converters, batteries.
- For machines: use synchronous/induction machine blocks (for PMSG/DFIG/SCIM) in Simscape or write custom dq models with ODEs.
- Implement MPPT and supervisory logic in standard Simulink function blocks.

Conclusion

The modeling and simulation of a hybrid wind–photovoltaic (PV) power system provide a comprehensive framework for understanding the integration of multiple renewable sources into a single energy conversion chain. By combining the aerodynamic and electrical dynamics of the wind turbine with the nonlinear I–V behavior of photovoltaic arrays, we can represent the hybrid system as an interconnected set of subsystems linked through a common DC bus and coordinated by power electronics.

The use of mathematical models — including the turbine's aerodynamic equations, dq machine models for induction or synchronous generators, the single-diode PV model, and averaged converter representations — allows engineers to study both steady-state and dynamic performance. Control strategies such as Maximum Power Point Tracking (MPPT), pitch control, and DC-bus regulation ensure that each subsystem operates at its optimum point while preserving the stability of the overall hybrid system.

Furthermore, the inclusion of energy storage and supervisory energy management highlights the practical challenges of balancing intermittent renewable generation with fluctuating load and grid requirements. Through simulation, one can analyze diverse operating scenarios such as variable irradiance, wind gusts, or battery state-of-charge limits, and design robust control algorithms to guarantee reliability and efficiency.

In summary, hybrid wind–PV systems illustrate the multidisciplinary nature of renewable energy conversion, combining aerodynamics, power electronics, electrical machines, control theory, and system integration. Their modeling not only supports academic study but also provides essential tools for engineers aiming to design, optimize, and deploy sustainable energy solutions for the future.

General conclusion

This polycopie has progressively guided us from the conceptual foundations of modeling and simulation, through their methodological frameworks, to their practical applications. By defining key notions, exploring types of models, and detailing the simulation process, we have established a coherent path that links theory to practice.

The integration of computational tools such as MATLAB/Simulink has shown how abstract models can be transformed into operational simulations, capable of supporting analysis, prediction, and decision-making in diverse domains. Case studies and examples have further illustrated the relevance of these techniques in engineering, science, and industry.

Ultimately, this document demonstrates that modeling and simulation are not only academic exercises but also indispensable instruments for understanding complexity, optimizing systems, and innovating in real-world contexts. The knowledge acquired here provides a solid foundation for deeper exploration of advanced techniques and specialized applications, ensuring that learners are well-prepared to harness the full potential of these approaches.

References

[1] Djamila Rekioua, *Hybrid Renewable Energy Systems: Optimization and Power Management Control*, Springer: Green Energy and Technology, 2020.

[2] Ahmed Masmoudi, *Control-Oriented Modelling of AC Electric Machines*, Springer Briefs in Electrical and Computer Engineering, 2018.

[3] W. -Z. Shen, W.-S. Wen, et al, *Wind Turbine Aerodynamics*, MDPI Books, 2019.

[4] Roger A. Messenger, Jerry Ventre, *Photovoltaic Systems Engineering*, CRC Press, 2004.

[5] Ivan Liebgott, Modeling and Simulation of Multi-Physics Systems with MATLAB, 2018.

[6] Djamila Rekioua , Ernest Matagne, Optimization of Photovoltaic Power Systems Modelization, Simulation and Control, Springer: Green Energy and Technology, 2012.

[7] Mundu, M.M., Nnamchi, S.N., Sempewo, J.I. et al. Simulation modeling for energy systems analysis: a critical review. Energy Inform 7, 75, 2024.

https://doi.org/10.1186/s42162-024-00374-8

[8] Moustakas K, Loizidou M, Rehan M, NizamiA, A review of recent developments in renewable and sustainable energy systems: key challenges and future perspective. Renew Sustain Energy Rev, 2020. https://doi.org/10.1016/j.rser.2019.109418

[9] Ethmane Isselem Arbih Mahmoud, Etude, modélisation et simulation d'une turbine éolienne, Éditions universitaires européennes

Annex

Wind Turbine Power Extraction

%% Wind Turbine Model

Vwind = 5:1:15; % wind speed range (m/s)

```
Omega_t = 0:1:300; % turbine speed range (rad/s)

fori = 1:length(Vwind)

lambda = (Omega_t * R) ./ Vwind(i);

Cp = Cp_max * (lambda./lambda_opt) .* exp(1-lambda/lambda_opt);

Pw(i,:) = 0.5 * rho * pi * R^2 * Vwind(i)^3 .* Cp;

end

% Plot wind turbine power vs speed

figure;

surf(Omega_t, Vwind, Pw);

xlabel('Turbine Speed (rad/s)');

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

zlabel('Power (W)');

title('Wind Turbine Power Extraction');
```

MPPT Algorithm (Perturb & Observe)

```
%% MPPT using Perturb & Observe (P&O)

Vpv = 0; Ipv = 0; % Initial values

duty = 0.5;       % Initial duty cycle
```

```
deltaD = 0.01;    % Perturbation step

Pold = 0;         % Previous power

for k = 1:100

Vpv = Vmp + (rand-0.5); % Simulated PV voltage with noise

Ipv = Imp + (rand-0.5); % Simulated PV current

   P = Vpv * Ipv;          % Power

if P >Pold

duty = duty + deltaD;

else

duty = duty - deltaD;

end

duty = min(max(duty,0),1); % Limit between 0 and 1

Pold = P;

End
```

PV Model (I–V Characteristics)

```
%% PV Cell I-V Model

V = linspace(0, Voc, 100);

I = Isc - (Isc/ (exp(1)-1)) .* (exp((V)/(Ns*0.5)) - 1) - V/Rp - Rs*Isc;
```

```
% Plot I-V curve

figure;

plot(V, I, 'b', 'LineWidth', 2);

xlabel('Voltage (V)');

ylabel('Current (A)');

title('PV Cell I-V Characteristic');

grid on;

% P-V curve

P = V .* I;

figure;

plot(V, P, 'r', 'LineWidth', 2);

xlabel('Voltage (V)');

ylabel('Power (W)');

title('PV Cell P-V Characteristic');

grid on;
```