What is the SIaD tool?

The **S**tability and **I**nteraction **a**ssessment in the frequency-**D**omain tool (**SIaD**) is a free-access tool designed to analyze small-signal stability and interactions in large modern power systems. It helps identify potential interactions and assess closed-loop stability using advanced frequency-domain analysis techniques.

SIaD consists of two main modules:

1. Scanning Module: Determines the system's frequency response through impedance and admittance matrices.
2. Analysis Module: Applies the Generalized Nyquist Criterion and other specialized algorithms to evaluate system stability and interactions.

Key features and capabilities:

- Identification of interaction frequencies and stability across a wide frequency range (including subsynchronous and harmonic regimes).
- Determination of stability limits between interconnected systems.
- Prediction of oscillations and their damping levels.
- Analysis of multiple operating points.
- Free-access software for Transmission System Operators (TSOs) and academic institutions.

Where to Get the SIaD tool?

The **SIaD** tool is freely available for download from the following GitHub repository:

https://github.com/luisangelgare/FD_Scanning_Tool

As this is an free-access software, any use, modification, or contribution based on this tool must properly acknowledge the original authors and contributors. To cite this work, please use the following reference:

García-Reyes, L. A., Gomis-Bellmunt, O., Prieto-Araujo, E., Lacerda, V. A., & Chea-Mañé, M., "SIaD Tool: A Comprehensive Frequency-Domain Tool for Small-Signal Stability and Interaction Assessment in Modern Power Systems", in IEEE Transactions on Power Delivery, submitted for review.

Requirements to Start Using the SlaD Tool

MATLAB Version:

- MATLAB R2022b or newer
- Simulink R2022b or newer
- Control System Toolbox
- Simscape Electrical

Python/PSCAD Version:

- Python 3.12.7
- PSCAD 5.0.2

Libraries:

- mhi, numpy, pandas, sys, decimal, matplotlib, scipy

Recommended Hardware:

- Intel Core i5 processor or higher
- 8 GB RAM or more

How to Use the SlaD Tool?

Once the SlaD tool source files have been downloaded and extracted, the usage process is divided into two separate guides depending on the working environment:

Using MATLAB/Simulink:

After extracting the contents of the .zip file, you will find four main files, as shown in Figure 1.

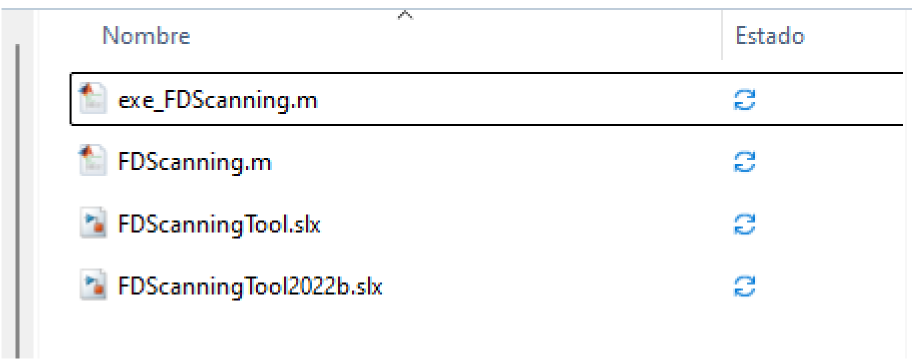

Figure 1. Files included in the SlaD tool for MATLAB/Simulink.

Each file serves the following purpose:

- *exe_FDScanning.m*: The main executable script that launches the **SlaD** Tool and contains all necessary settings and parameters for scanning and stability analysis.
- *FDScanning.m*: The master script containing the full algorithm for open-loop system scanning. This file must be placed in the working directory without modification.
- *FDScanningTool.slx*: A Simulink file that includes the scanning block. It contains essential parameter configuration information for proper use of the **SlaD** tool. Versions are available for R2024b and R2022b.

Steps to Install and Use the SlaD Tool (MATLAB/Simulink)

Step 1: Set Up the Working Environment

Copy the *FDScanning.m* file into your main working directory (see Figure 2). This folder should include all the necessary files to run the simulation, such as:

- Scripts for pre-calculating simulation parameters.
- Auxiliary functions used by the main script and its dependencies.
- The Simulink EMT model file (.slx) of the system to be identified.

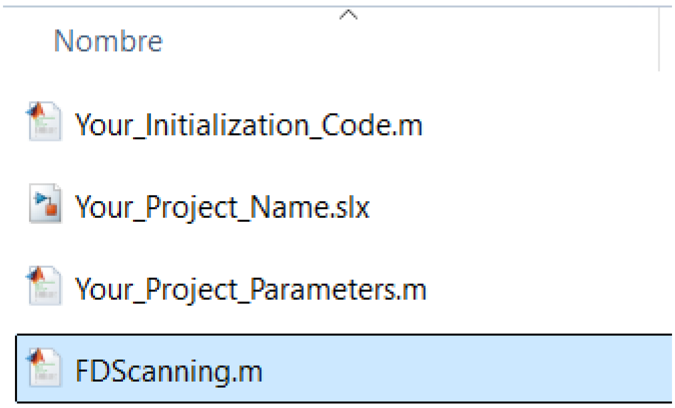

Figure 2. Copying the master file *FDScanning.m* to your working directory.

Step 2: Configure the Execution Script

Also copy the *exe_FDScanning.m* file into the same working directory mentioned earlier (see Figure 3). Once the file is in place, open it to define the scanning and system identification parameters.

This script allows you to configure key aspects of the analysis, such as the frequency range, type of perturbation, and other parameters required to successfully run the identification process.

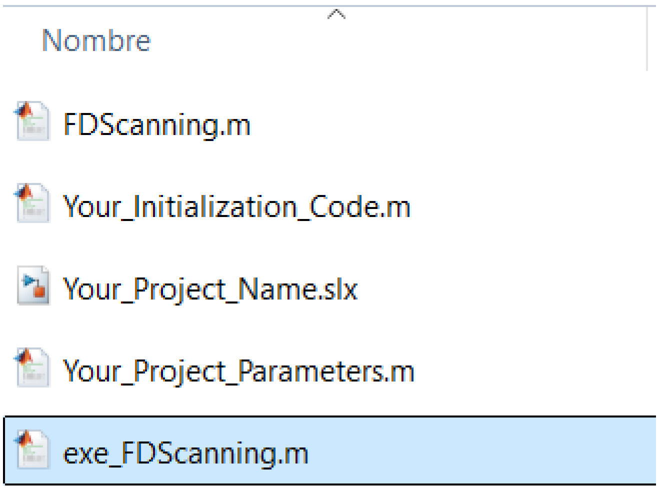

Figure 3. Copying the file *exe_FDScanning.m* to your working directory.

Step 3: Define Global Variables in the Execution Script

Open the *exe_FDScanning.m* file. Inside the script, you will find a section for defining the global variables used by the tool. These variables must be **unique within the working environment**, so make sure they are **not duplicated** in your initialization scripts, parameter files, or especially within the Simulink EMT model.

The variables to be defined are shown in Figure 4 and are explained below.

```
%% Simulation parameters

program='GFM_Nonlinear.slx'; % Simulink program selection
model='GFM_Nonlinear'; % Main canvas name of your project (same as program usually)
Tinit=6; % Initialization time (arbitray for select the steady state)
fs=1; % Sampling frequency for FTT (specify the value in Hz for minimal freq)
delta_t=25E-6; % Fixed step time
fd0=unique(round(logspace(0,log10((1/delta_t)/4),5)));  % Perturbation frequencies in Hz

%% Steady state and disturbance sources data

f0=50; % Fundamental base frequency
w=2*pi*f0; % Base angular frequency (rad/s)
Sbase=2.75E6; % Base power (VSC or System to idenfify)
Vbase=690; % Base voltage (V)
Vpeak=(Vbase/sqrt(3))*sqrt(2); % Fundamental peak voltage at the connection
Vq_ss=0; % q-component steady state voltage at bus under analysis
Vd_ss=0; % d-component steady state voltage at bus under analysis
Ipeak=Sbase/Vpeak; % Fundamental peak current through the link
Iq_ss=0; % q-component steady state current at link under analysis
Id_ss=0; % d-component steady state current at link under analysis
Vperturbation=0.03; % Percentage of nominal voltage perturbation
Iperturbation=0.03; % Percentage of nominal current perturbation
```

Figure 4. Global variables definition in *exe_FDScanning.m*.

3.1 Simulation Parameters

The first set of parameters in the *exe_FDScanning.m* script is divided into two sections. The first section includes the simulation parameters, which must be carefully defined to ensure proper execution of the analysis.

- `program`: This is where you insert the name of the .slx file containing the EMT model of the system to be analyzed. For example, in the application shown in Figure 3, the file name would be "*Test_Network.slx*".
- `model`: Repeat the same name as above, but without the .slx extension, assuming the main canvas in Simulink uses the same name.
- `Tinit`: This parameter defines the time at which the system reaches a steady state. It is set arbitrarily by the user based on prior observation of the system's transient behavior. In other words, it should correspond to the moment when the initial transient has passed and all controllers and system dynamics are active. Figure 5 shows a practical example of how to identify this time in a simple transient. As illustrated, T_{init} would be the point where the system is stable and ready to receive perturbations for small-signal identification.

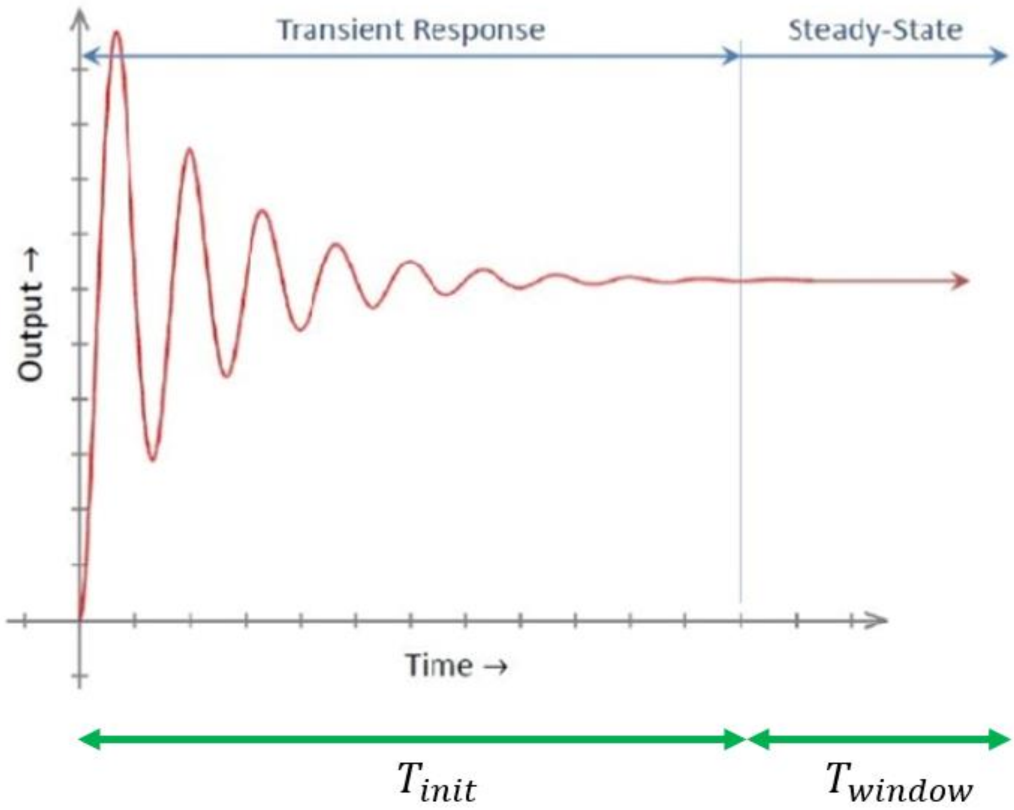

Figure 5. Details in the selection of T_{init}.

- `fs`: This parameter defines the desired frequency resolution in the frequency domain, i.e., the minimum frequency to be identified in the system. It directly determines the time window (T_{window}) used during the simulation, based on the inverse relationship:

$$fs = \frac{1}{T_{window}}$$

  Therefore, a lower resolution frequency `fs` requires a longer time window T_{window} to accurately capture the system dynamics within that frequency range. When the fs parameter is set, **the tool automatically** calculates the additional simulation time required to cover the necessary observation window. This total simulation time is referred to as `Tobs`, and is defined as:

$$Tobs = T_{init} + T_{window}$$

  For example:

  - If a 1 Hz spectral resolution is desired, the time window will be 1 second, so the simulation must run until $T_{init} + 1\ s$.
  - If a 0.1 Hz resolution is required, the window will be 10 seconds, and the total simulation time will be $T_{init} + 10\ s$.

  This parameter can also be interpreted as the **minimum frequency to be identified**, and its selection depends on the type of study being conducted, such as:

  - Subsynchronous interactions (require lower frequencies, hence longer windows).
  - Harmonic interactions (may require higher frequencies, with shorter windows).

- `delta_t`: This is the discrete integration step defined by the user for EMT simulations and system identification. While arbitrary, it must be consistent with the type of study:

  - For harmonic interactions (high frequency), a high resolution is recommended, e.g., 10 µs.
  - For low-frequency interactions, a larger step can be used to speed up the study, with a maximum recommended value of 50 µs.

- `fd0`: This is a frequency vector defined by the user to select the frequency band for system identification. By default, a *single-tone* identification is used, with a logarithmic scale vector from 1 Hz up to the maximum sampling frequency allowed. It is crucial not to exceed the sampling limit to avoid aliasing errors:

$$fd_{max} \; < \frac{fsampling}{4} \; ; \; where \; fsampling = \frac{1}{delta_t}$$

This vector can be modified to specify all desired frequencies for system perturbation. If *multi-tone* perturbations are used, the same vector format must be maintained, but the tool will only consider the initial and final frequencies to synthesize a sinusoidal or random binary waveform covering that spectrum. For better low-frequency identification, a narrow range is recommended (e.g., 1 to 100 Hz), and for high-frequency studies, a wider range.

Recommendation: The authors suggest using *single-tone* perturbations due to their higher accuracy.

3.2 Steady-State and Disturbance Source Data

This section includes variables required for the **ideal source** to properly identify the operating point of the element under study. You must enter the values used in the EMT simulation:

- If the .slx model uses **per-unit (p.u.)** values, enter them in p.u. here.
- If **real units** (e.g., volts or amperes) are used, maintain those units.

For `Ipeak`, it is recommended to first extract the current measurement from the link previous **SIaD** and then use that value to improve identification accuracy.

The `Vperturbation` and `Iperturbation` variables define the **percentage of disturbance** applied to voltage and current. These should always be kept between **0.01 and 0.03** to preserve linearity and the system's operating point.

Step 4: Select the Scanning Options

Next, you must select the available scanning options within the script. Figure 6 shows the scanning methodologies integrated into the tool.

All available combinations can currently be used. However, to properly tailor each methodology to your specific study, it is highly recommended to consult the reference article associated with the tool. Doing so will help you achieve more accurate and relevant results for your analysis.

```matlab
%% Scanning options available

% Scanner settings:
% 1 -> Voltage perturbation
% 2 -> Current perturbation
scanner_type=1;

% 1 -> Single-tone perturbation
% 2 -> RBS perturbation
% 3 -> Multi-tone perturbation
signal_type=3;

% 1 -> ABC scan
% 2 -> qd0 scan
% 3 -> pn0 scan
scanner_selector=2;

% 1 -> linear response exists
% 0 -> no linear response is available
linear=0;
```

Figure 6. Scanning options available in the **SIaD** tool.

In addition to selecting the scanning methodology, users can choose between different types of perturbation and identification configurations:

- Perturbation type:
  - **Voltage-based** or **current-based**.
  - Identification signals can be either **random binary** or **multi-frequency sine**.
- Reference sequence:
  - Identification can be performed in different **reference sequences** (e.g., positive-negative or dq), depending on the analysis requirements.
- Linear validation:
  - If a **theoretical linear response** of the system is available, the "linear" option can be enabled to perform an additional validation of the identification results.

All available options are clearly numbered and described within the script, allowing for a very straightforward and user-friendly selection process.

Step 5: Finalize the Executable Script Configuration

The final part of the *exe_FDScanning.m* script includes the necessary components to properly execute the identification process:

1. Load the system parameter program: If your project requires **pre-processing calculations** before initialization or simulation (e.g., parameter estimation, controller setup), this is where you should include the corresponding script.
2. Project initialization: If a **power flow** or other initialization process is required before running the EMT simulation, it should be added here.
3. Run the master script: Finally, the **main identification script** is called, which automatically executes the entire scanning, simulation, and analysis process.

The complete identification workflow is illustrated in the diagram shown in Figure 7, which summarizes the steps from setup to results.

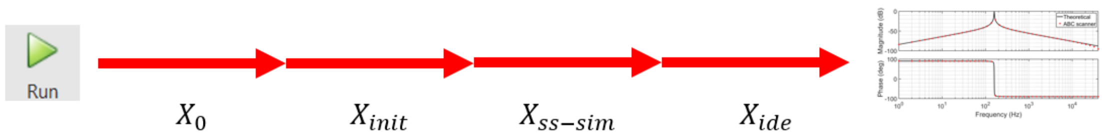

Figure 7. SIaD tool full identification workflow.

5.1 Steady-State Parameter Calculation (X_0)

The first stage of the identification process, labeled X_0 in Figure 7, corresponds to the calculation of the system's steady-state parameters. This step is equivalent to running your custom script, for example: *Your_Project_Parameters.m*. This script should include all necessary calculations to generate the variables required for the EMT simulation, such as initial conditions, electrical parameters, controller settings, and more.

Executing this step ensures that the system is properly parameterized before starting the simulation and applying perturbations. Figure 8 provides a detailed illustration of this process.

```
%% Insert here your file with your system initialization data (without run)

%-----------------
% Here is your code for parameters file and initizalization
Your_Project_Parameters;
Your_Initialization_Code;
%-----------------

% Loading the FDs main code...
FDScanning;
```

Figure 8. Initialization of the SlaD tool.

Your_Project_Parameters.m **must not contain any commands that start the EMT simulation**, especially if the SlaD block is already connected in the .slx mode (see **Step 6** for reference on integrating the SlaD block in Simulink).

5.2 Stage X_{init}: Model Initialization

If your project requires a **power flow** or other pre-simulation calculation to obtain necessary variables, include tour script: *Your_Initialization_Code.m*. This script must be placed in the **same folder** as the rest of your project files (refer to **Figure 3** for the file placement).

5.3 Stages $X_{ss\text{-}sim}$ and X_{ide}: Simulation and Identification

This script stage automatically executes the last two stages:

- $X_{ss\text{-}sim}$: Runs a simulation to calculate the steady-state values of **voltage and current** at the bus where the SlaD tool is connected.
- X_{ide}: Performs the **complete system identification**.

The duration of this stage depends on the type of perturbation signal used:

- Random binary (RBS) and multi-frequency sine signals allow for faster identification.
- Single-tone signals take longer but provide higher accuracy.

Recommendation: Single-tone signals are recommended for more precise identification results.

Step 6: Integrate the Identification Block in Simulink

To integrate the block that enables system identification within your Simulink model, follow these steps:

1. Open the *FDScanningTool.slx* file, located in the **root folder** where the tool was downloaded (see **Figure 1** for a reference of the folder structure).
2. Inside this file, you will find the block named **"Frequency Scanner"**, colored in **red** for easy identification.
3. This block should be **copied and inserted** into your project's EMT model (.slx) at the point in the system where you want to perform the identification.

Figure 9 shows the visual location and appearance of the block within the original file.

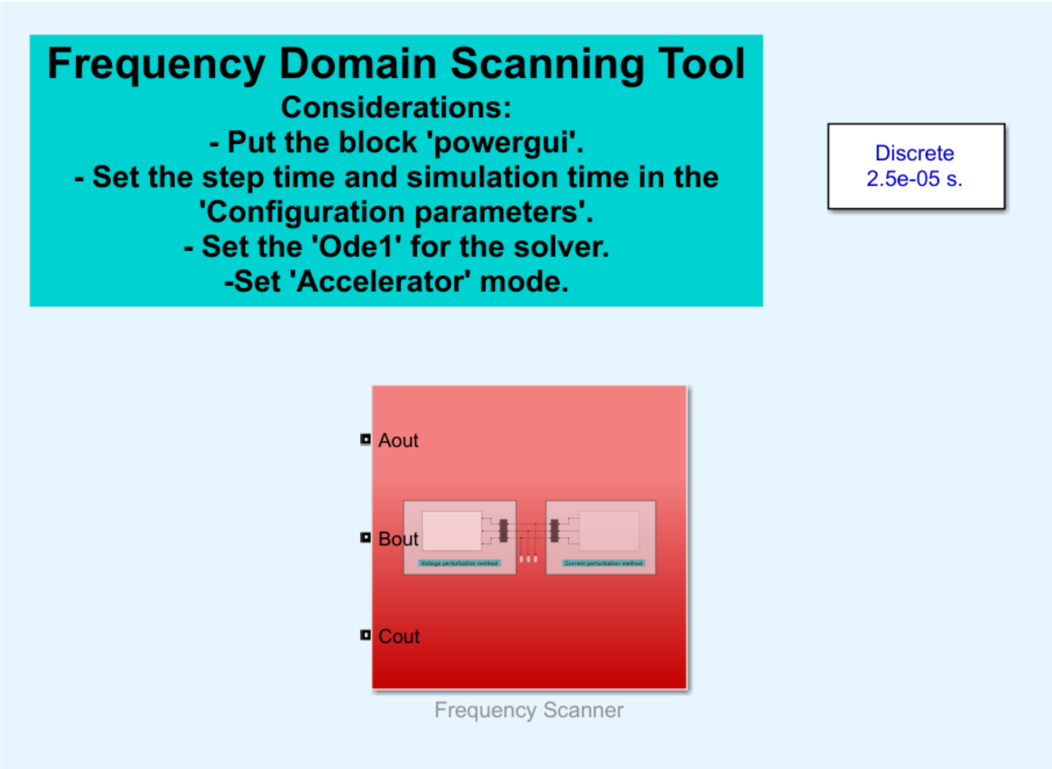

Figure 9. Frequency Scanner block.

Once the **Frequency Scanner** block is located, it must be **copied and inserted** into your simulation project: *Your_Project_Name.slx*. Next, you must **properly connect the terminals** of the block to the **three-phase bus** at the point where system identification is to be performed. This connection point should correspond to the location in the system where the frequency response is to be measured.

For example, in a typical case study showed in the Figure 10, the scanner is connected at the Point of Connection (PoC) of a wind farm located at an offshore node, in order to identify its dynamic behavior.

Since the connection points are **three-phase**, it is essential to ensure that **each phase (A, B, and C)** is connected correctly. Figure 10 illustrates the proper way to make these connections.

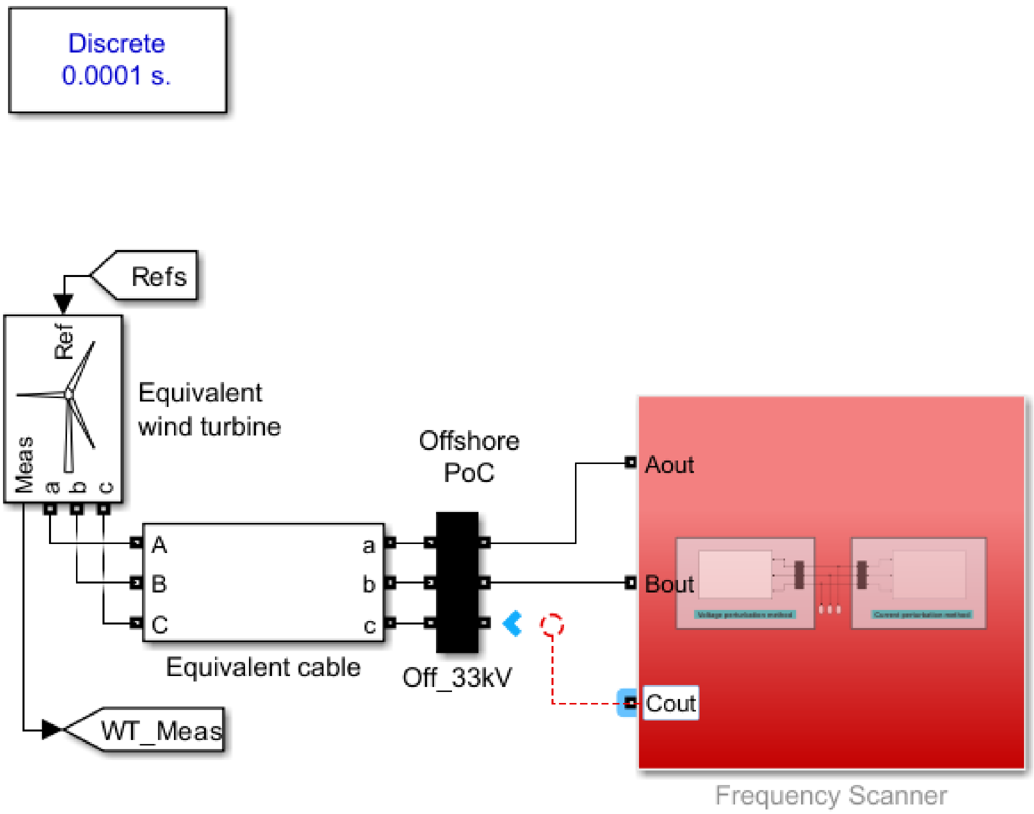

Figure 10. Offshore wind farm example of connection.

Step 7: Configuring the Simulink Environment for the SIaD Tool

Following the instructions shown in the **Frequency Scanner** block interface (see Figure 9), the following steps must be taken to properly integrate the tool into your project:

7.1. Configure solver parameters

Due to the nature of the identification method, your .slx model must operate in **discrete time**. To do this:

- Open your model in Simulink.
- Click on the **“Model Settings”** icon ( ) and go to **Solver** options

- Set the variables `Tobs` and `delta_t` (previously defined in the *exe_FDScanning.m* script).
- Select the **“Fixed-step”** option.
- Choose the solver **ode1 (Euler)**, which is recommended by the authors for its simplicity and compatibility with the identification method.

Once these parameters are configured, click **“Apply”** to save the changes and then **“OK”**. This process is illustrated in **Figure 11**.

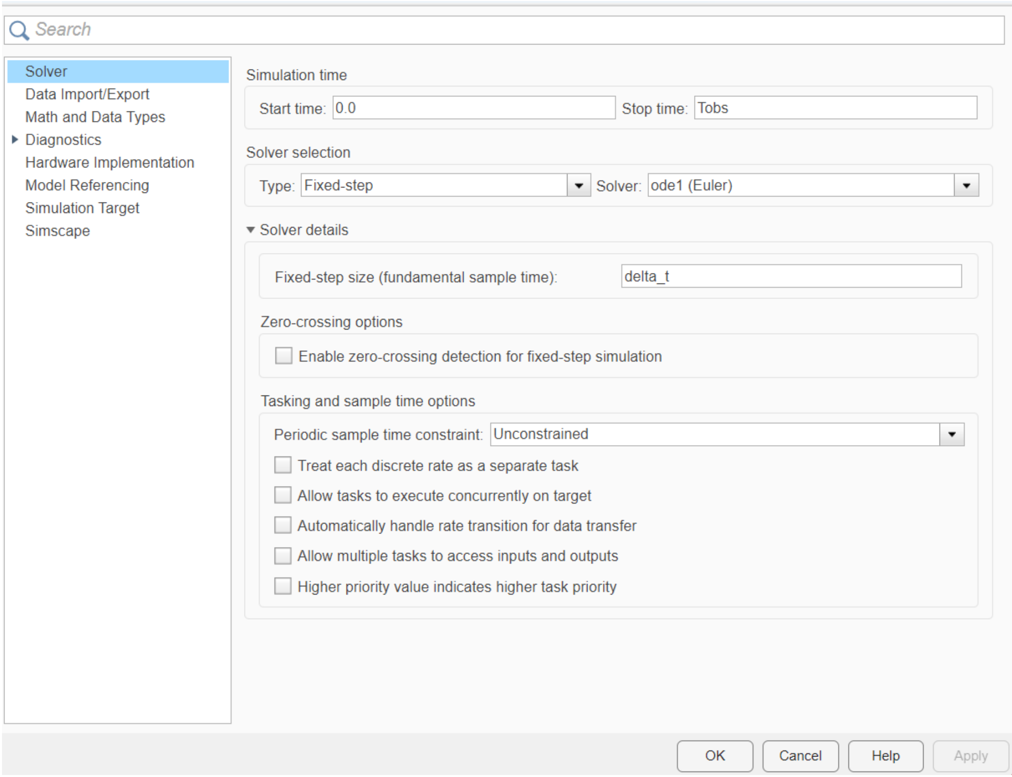

Figure 11. Model settings adjustments for the SIaD tool.

7.2. Enable “Accelerator” Simulation Mode

As a final step, it is recommended to enable the **“Accelerator”** simulation mode to improve performance during model execution (see Figure 12).

To do this:

- Go to the **“Simulation”** section in the Simulink top menu.
- Then, under the **“Simulate”** subsection, just below the **“Stop Time”** field, open the dropdown menu and select **“Accelerator”**.

This setting allows Simulink to compile the model before running it, which can significantly reduce simulation time, especially for complex systems.

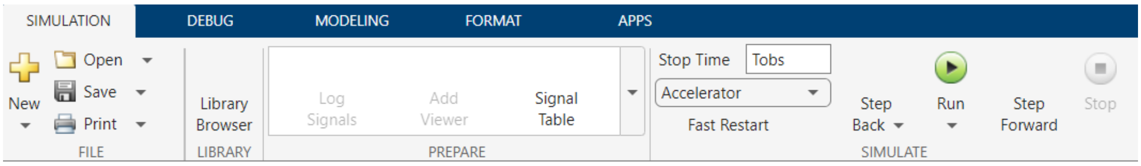

Figure 12. Simulation acceleration option selection.

Once all changes are made, **save the model**. At this point, the environment is fully configured and ready to begin working with the SIaD tool.

Step 8: Running the Study and Viewing Results

Once all **7 steps** of this manual have been completed, click **“RUN”** in the executable script (*exe_FDScanning.m*). The tool will automatically perform the identification study and generate a **plot with the results**.

The **identification variables** will be saved in the MATLAB **Workspace**, organized according to the selected **reference frame**. For example:

- Yxx or Zxx, where x can be:
    - ABC (phase sequence),
    - dq0 (synchronous reference frame),
    - 0pn (modal sequence: zero, positive, negative).

These variables will be available for further analysis depending on your study needs.

Who to Contact for Support or Follow-Up?

For questions, follow-up, or technical support regarding the **SIaD** tool, please contact:

luis.reyes@upc.edu

Important Note

The SIaD tool is still under development. Any **improvements, suggestions, or issues** you identify are highly appreciated and should be reported to the authors to help **refine and enhance** the tool for the benefit of the community.

Acknowledgements

This work has received funding from the ADOreD project under the European Union’s Horizon Europe Research and Innovation Programme under the Marie Skłodowska-Curie Grant Agreement No. 101073554.