Wind Farm Passivity Analysis — User manual

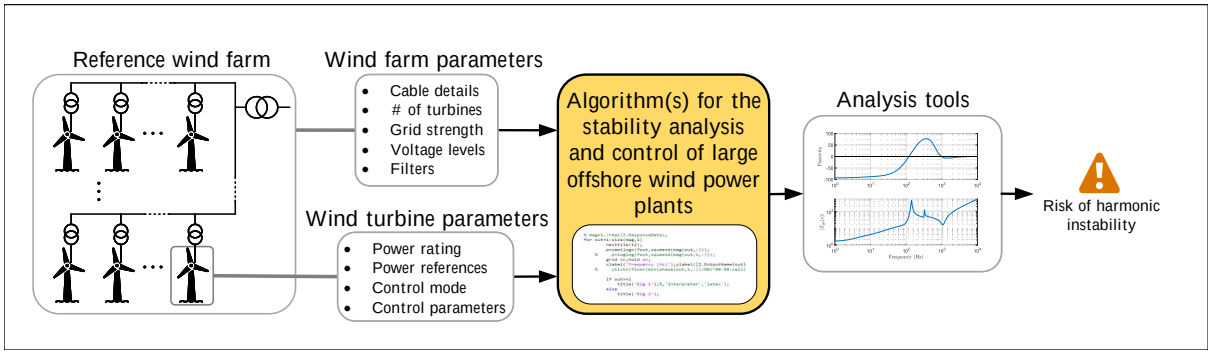
Andrew Macmillan Smith
andrew.smith@ntnu.no

December 23, 2025

Petronas-NTNU-IEL Collaboration within the SAFER Project

Stability and control of large offshore wind power plants

Equipment and Methods for Harmonic Mitigation in Offshore Wind Power Plants



This manual and the related code is a work in progress, which is continuously improved by the authors. It is not a finished work and may therefore contain defects or "bugs" inherent to this type of development. For this reason the work is provided without warranties of any kind concerning the work, including without limitation merchantability, fitness for a particular purpose, absence of defects or errors, and accuracy.

Contents

1 Code Capabilities

This document explains how to use the `WF_parameter_variation.m` script to run parameter sweeps and visualize wind-farm (WF) impedance and passivity properties for stability assessments. The code allows for quick and easy sweeping of parameters and recalculation of the passivity and impedances for a variety of conditions. Included are several grid-forming implementations as well as a grid-following control implementation.

2 Prerequisites

- MATLAB (tested with R2023a)
- Control System Toolbox recommended (for `ss` and `frd`)
- The Simulink model `WF_offshore.slx` if you plan to run Simulink blocks

2.1 Quick Start

1. Open MATLAB and set current folder to the repository root (where `WF_parameter_variation.m` exists).
2. Run:
   `WF_parameter_variation.m`
3. The script opens figures showing passivity and impedance plots.

3 Detailed Instructions

This section gives step-by-step, practical instructions for running and adapting the script.

3.1 Prepare environment

1. Open MATLAB and set the current folder to the repository root (the folder that contains `parameters_WF_default.m`).
2. Add repository to MATLAB path if needed:
3. Ensure the Simulink models `WF_onshore.slx` and `WF_offshore.slx` is available if you plan to run Simulink-based checks.

3.2 Run a quick nominal check

1. Load default parameters and compute the nominal WF model:

```
run('parameters_WF_default.m');
WF = calculate_WF(WF);
```

2. Plot nominal passivity and impedance of input 1 and output 1 (Z_{pp}):

```
f = figure;
opts.plot_pass=1;
```

```
opts.plot_imp=1;
plot_p_vary(f, WF.Imp.Z_wf, WF.Imp.Z_wf_grid, 1, 1,opts);
```

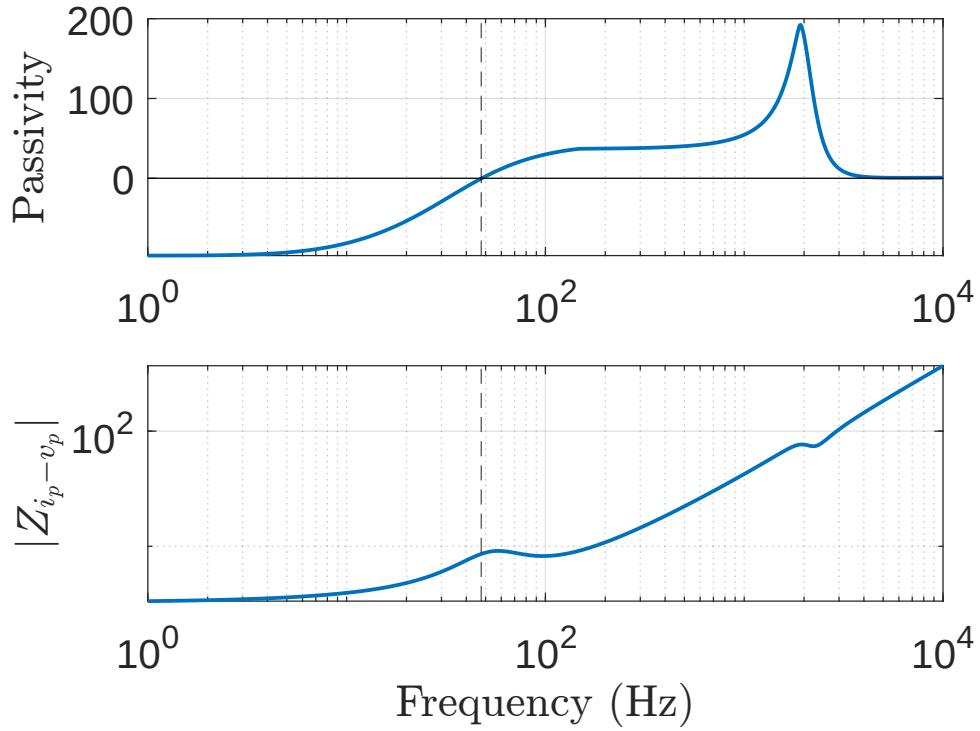


Figure 1: Nominal passivity plot.

3.3 Changing a wind farm parameter

Wind farm parameters can be changed by simply setting the field in the `WF` structure. For example, the wind farm location can be changed from onshore to offshore by:

```
WF.location= 'offshore';
```

The structure of the wind farm is divided into the following substructures:

- `WF.Trans`: Transmission system of the wind farm. Included in this substructure are parameters related to transmission such as transmission length, effective transmission line impedance values, and equivalent grid impedance.
- `WF.WT`: Individual and aggregated wind turbine parameters. Included in this substructure are parameters related to the wind turbine including physical filter and transformer values, as well as control parameters. Included in this release are 6 different control schemes: 5 grid-forming schemes and 1 grid-following scheme. These can be selected by setting `WF.WT.cvtr.parameters.model_selector` to 1 through 6.
- `WF.Imp`: Various wind farm impedances are stored in this substructure including the wind farm and wind turbine impedance, total impedance including transmission and grid, and base values for per unit.

More details of each parameter can be found by examination of the default parameter

file, `parameters_WF_default.m`.

As an example, to plot the passivity and impedance of an onshore vs. offshore wind farm the following can be used:

```
WF.location='onshore';
WF=calculate_WF(WF);
Z1_on=WF.Imp.Z_wf;
Z2_on=WF.Imp.Z_wf_grid;

WF.location='offshore';
WF=calculate_WF(WF);

Z1_off=WF.Imp.Z_wf;
Z2_off=WF.Imp.Z_wf_grid;

f=figure;
plot_p_vary(f,Z1_on,Z2_on,1,1);
plot_p_vary(f,Z1_off,Z1_off,1,1);
```

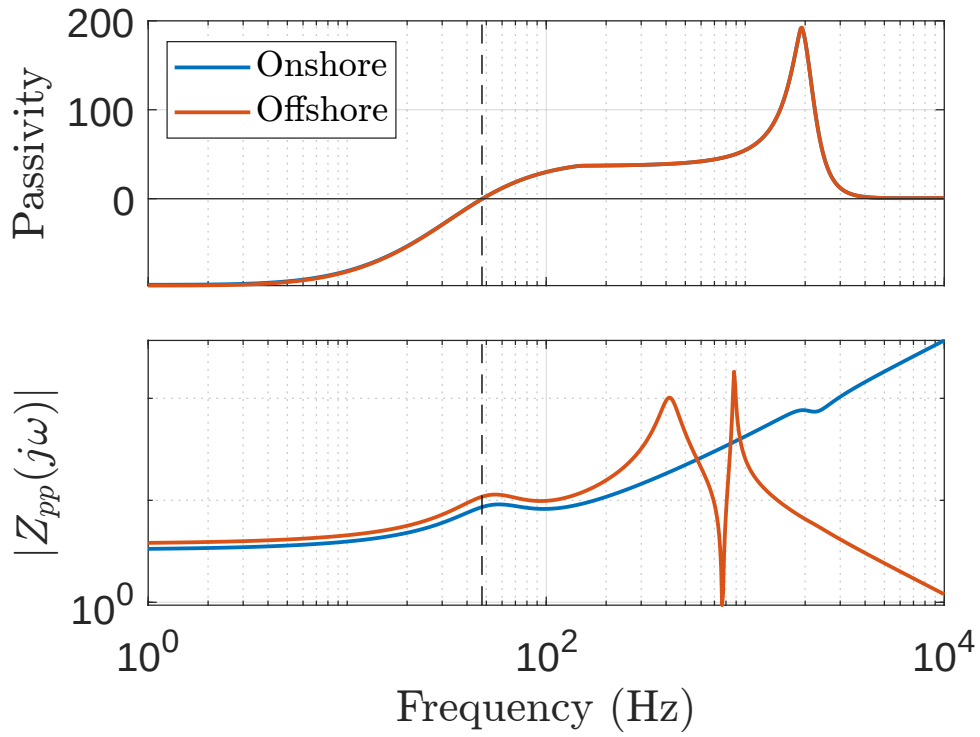


Figure 2: Passivity and impedance plot for onshore and offshore cases.

3.4 Running a parameter sweep

To run a quick parameter sweep, a short for loop can be used. Example: vary distance over a small range and plot:

```
run('parameters_WF_default.m');
WF.location='offshore';
p_vary.name = 'Trans.D_C';
```

```
p_vary.val = 10:4:150;
for i = 1:length(p_vary.val)
  WF = setNestedField(WF, p_vary.name, p_vary.val(i));
  WF = calculate_WF(WF);
  Z1_array{i} = WF.Imp.Z_wf;
  Z2_array{i} = WF.Imp.Z_wf_grid;
end
f=figure;
plot_p_vary(f, Z1_array, Z2_array, 1, 1);
```

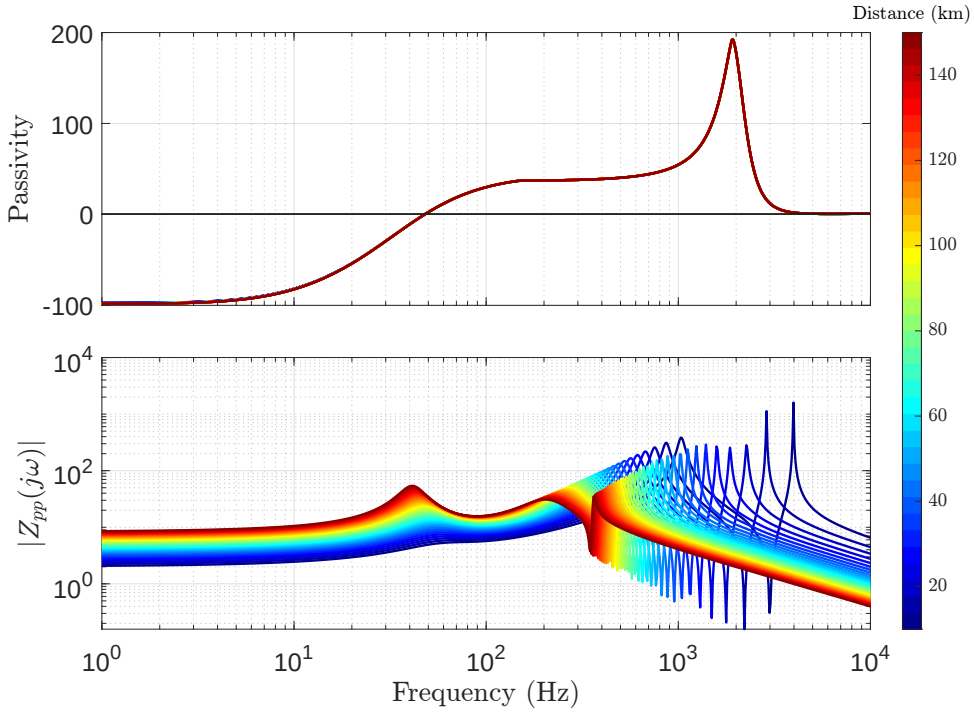


Figure 3: Parameter sweep example (distance sweep).

Change `p_vary.name` to any field in the `WF` struct (dot-separated path) and `p_vary.val` to your sweep vector.

3.5 Included tests

- Onshore vs. offshore and GFL vs. GFM
- Sweep of transmission distance
- Sweep of number of turbines connected
- Sweep of grid impedance
- Sweep of active and reactive power references
- Enabled/disabled voltage feedforward
- Sweep of active damping
- Sweep of virtual impedance (GFM)

- Sweep of droop gains and VSM parameters

4 Useful Functions

There are several useful functions included to help with analysis and plotting of the results besides the ones directly used for calculating the wind farm impedances. They are:

- `plot_p_vary`: Plots the passivity and impedance of either a single impedance or array of impedances as a function of the frequency. The impedances should be a frequency response object, `frd`. This uses the function `mimo_passivity` to calculate the passivity.
- `mimo_passivity`: Calculates the equation: eig $\left[\mathbf{Z}(j\omega) + \mathbf{Z}^H(j\omega)\right] > 0$ for a MIMO impedance defined as an `frd`.
- `setNestedField`: Sets an arbitrary field that may be nested by listing the full field name as a string. This is useful for parameter sweeps.