Immagine che contiene testo, Carattere, logo, schermata

Il contenuto generato dall'IA potrebbe non essere corretto.

List of Examples and Descriptions

# Glossary

**VirtualLab:** The environment where all the modules and examples are stored.

**SimPla:** Simulated Plasma is a simple module to simulate the equilibrium given the target separatrix and the functionality between the toroidal density current and the magnetic surfaces. Kinetic profiles are then evaluated using analytical formulation.

**SynDiag**: Module for synthetic diagnostics. It calculates the expected measurements given the plasma state.

# SinPla

# Example 1

## General Description

This script provides a comprehensive workflow for setting up and simulate a Tokamak equilibrium scenario using the Tokalab framework. The process begins with the initialization of a Tokamak structure, followed by loading the standard Tokalab geometry, scenario, and kinetic profiles.

Once the core data is loaded, the script imports the geometry object, like the grid and the wall. The Equilibrium class is then employed to prepare all necessary components for solving the Grad–Shafranov equation, enabling equilibrium simulation.

Finally, synthetic diagnostics are applied to the resulting equilibrium, offering a way to generate and visualize expected diagnostic outputs based on the current configuration.

## MATLAB script

Before running a MATLAB script or example, remember to run the function:

VirtualLab\_init()

This will add all the VirtualLab paths to run the various modules. At the beginning of each script, typically you want to clear the workspace and the command window, done by clear and clc functions:

clear; clc;

Then, you initialise the tokamak class:

% initialise the class tokamak

tok = tokamak;

If you want to use standard Tokalab geometry, scenario and kinetic profiles (single null case), you can just run machine\_upload, scenario\_upload and kinetic\_upload, without any inputs:

% upload the geometry information of your tokamak

tok = tok.machine\_upload();

tok = tok.scenario\_upload();

tok = tok.kinetic\_upload();

Once the fundamental configuration information has been uploaded in the tokamak class (here named tok), you can upload your geometry class and run the function import\_geometry (takes information from tok), build\_geometry (prepare the grid), and inside\_wall (evaluate the variable wall.inside, useful for fast calculation during equilibrium solving).

% initialise the class geometry

geo = geometry;

geo = geo.import\_geometry(tok);

geo = geo.build\_geometry();

geo = geo.inside\_wall();

Once the geometry is prepared, the equilibrium class initialised, and some fundamental information from classes tokamak (tok) and geometry (geo) are uploaded.

% initialise the class equilibrium

equi = equilibrium;

equi = equi.import\_configuration(geo,tok.config);

equi = equi.import\_classes();

Then, the (target) separatrix curve is created:

equi.separatrix = equi.separatrix.build\_separatrix(equi.config.separatrix,equi.geo);

To plot the wall and the target separatrix, it is possible to run built in plotting functions:

% show uploaded geometry and target separatrix

figure(1)

clf

geo.plot\_wall()

hold on

equi.plot\_separatrix()

and this should be the output:

% solve equilibrium

equi = equi.solve\_equilibrium();

% post processing (Opoint, Xpoint, LFCS)

equi = equi.equi\_pp();

% mhd and kinetic profiles

equi = equi.compute\_profiles();