A Python-based Software Tool for Power System Analysis

Dr. Federico Milano

E-mail: Federico.Milano@ucd.ie

School of Electrical, Electronic and Communications Engineering

University College Dublin

Ireland

Why Python?

- Python is a modern scripting language which merges together the flexibility of class-oriented programming (such as Java) and neat functional programming schemes (such as Haskell).
- It includes a **huge** number of third-party modules for mathematical applications, 2D and 3D plotting, html and xml parsing, etc.
- However, Python is an interpreted language, hence it is **slow** . . .

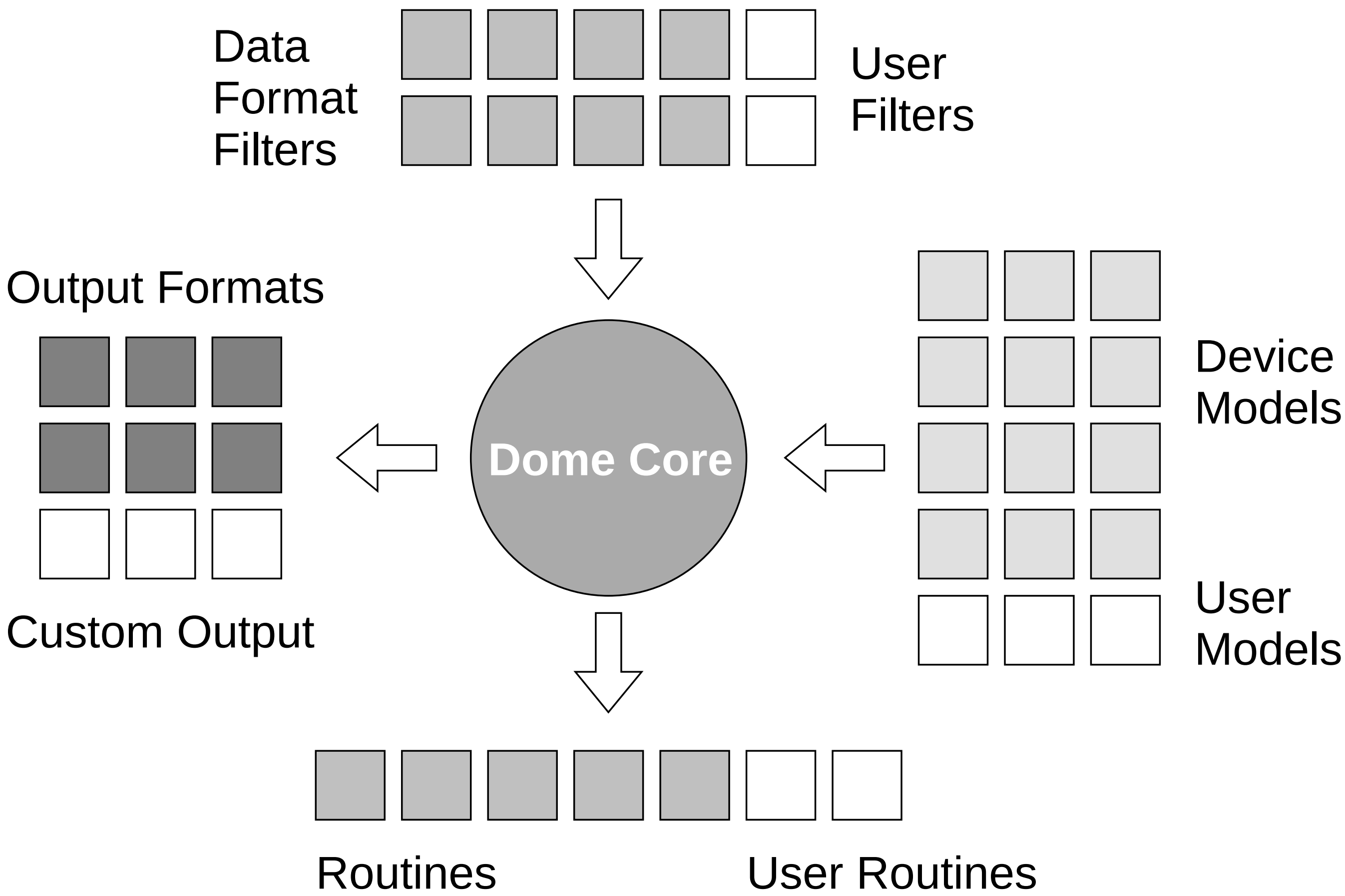
Python is just a Glue!

- Another property of Python is that it can be easily linked to C, C++ and Fortran libraries.
- So, Python can be used as a **glue** to link together efficient (compiled) mathematical libraries.
- In conclusion Dome can be quite **fast**!
- For example, to solve the power flow analysis of the UCTE test system (1254 buses and 1944 lines) takes about 0.05 seconds (neat CPU time of the NR routine) on a standard Dell workstation.

Dome: Objectives

- A Python-based Tool for Power System Analysis.
- Dome has been designed with the following objectives in mind:
  - Modularity
  - Fast prototypying of models, algorithms and libraries
  - Parallel computing
  - Lazyness

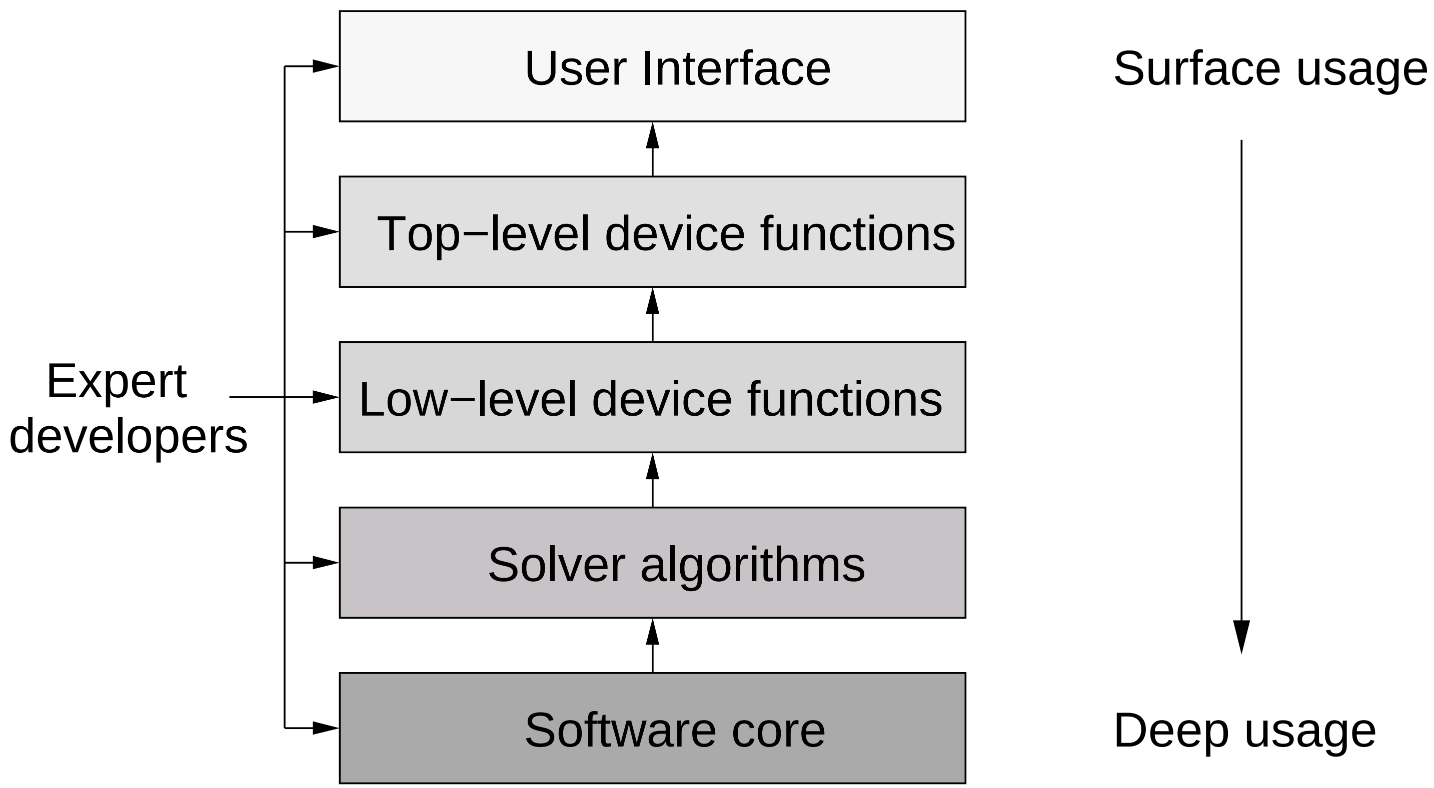
Dome: Structure



Dome: Methodological Rules

- Dome has been designed based on systematic methodological (even *philosophical*) rules:
  - Bottom-up approach (or *divide et impera*)
  - No code duplicates (heavily based on Python classes)
  - *Everything* can be customized (Hegel's approach!)
  - Separation between *models* and *solvers*
  - Minimizing the amount of *essential* code sections
  - Never complain (use defaults and warn the user!)
  - Code and documentation live (and update) together
  - Layered structure
  - The *world* is described by differential algebraic equations and/or if-then rules.

Layered Structure



Mathematical Model: DAE

- Dome requires a set of nonlinear differential algebraic equations (DAE) with discrete variables, as follows:

$$\begin{aligned} \dot{\boldsymbol{x}} &= \boldsymbol{f}(\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{\eta}, \boldsymbol{u}, t) \\ \boldsymbol{0} &= \boldsymbol{g}(\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{\eta}, \boldsymbol{u}, t) \end{aligned} \quad (1)$$

where $\boldsymbol{x}$ ($\boldsymbol{x} \in \mathbb{R}^{n_x}$) indicates the vector state variables, $\boldsymbol{y}$ ($\boldsymbol{y} \in \mathbb{R}^{n_y}$) are the algebraic variables, $\boldsymbol{\eta}$ ($\boldsymbol{\eta} \in \mathbb{R}^{n_\eta}$) are the controllable parameters, $\boldsymbol{u}$ ($\boldsymbol{u} \in \mathbb{R}^{n_u}$) are discrete variables, $\boldsymbol{f}$ ($\varphi : \mathbb{R}^{n_x} \times \mathbb{R}^{n_y} \times \mathbb{R}^{n_\eta} \times \mathbb{R}^{n_u} \times \mathbb{R}^{+} \mapsto \mathbb{R}^{n_x}$) are the differential equations, and $\boldsymbol{g}$ ($\varphi : \mathbb{R}^{n_x} \times \mathbb{R}^{n_y} \times \mathbb{R}^{n_\eta} \times \mathbb{R}^{n_u} \times \mathbb{R}^{+} \mapsto \mathbb{R}^{n_y}$).

- Discrete variables $\boldsymbol{u}$ can be often translated into *if-then* rules.

Mathematical Model: SDAE

- Dome can take into account stochastic processes leading to stochastic differential algebraic equations (SDAE):

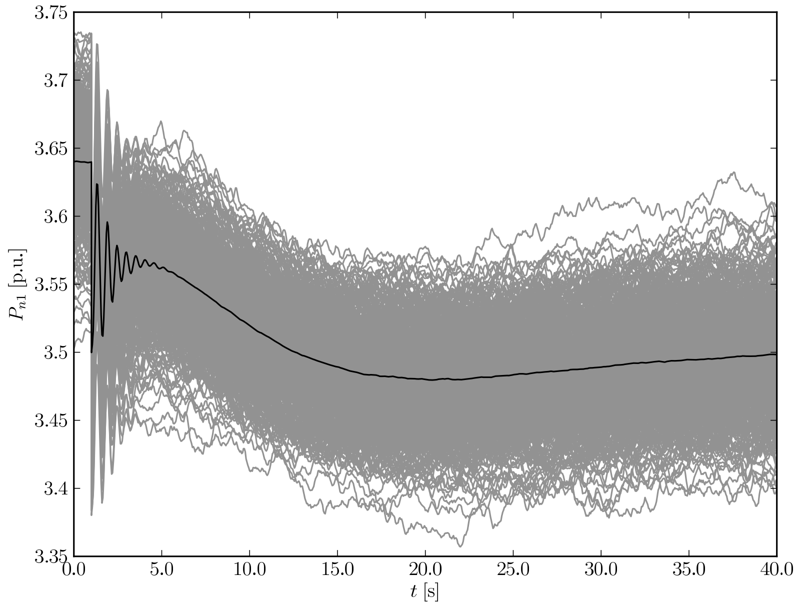
$$\begin{aligned} \dot{\boldsymbol{x}} &= \boldsymbol{f}(\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{\eta}, \boldsymbol{u}, t) + \boldsymbol{B}(\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{\eta}, \boldsymbol{u}, t)\boldsymbol{\xi} \\ \mathbf{0} &= \boldsymbol{g}(\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{\eta}, \boldsymbol{u}, t) \end{aligned} \tag{2}$$

where $\boldsymbol{\xi}$ are *white noises*, i.e., the time derivative of Wiener's processes:

$$\boldsymbol{\xi} = \frac{d\boldsymbol{W}}{dt}$$

and $\boldsymbol{B}$ is the *diffusion* tensor.

Stochastic Time Domain Analysis



- Dome allows solving time domain simulations in parallel.
- Solving 1000 simulations with a fixed time step of 0.05 s and a standard dishonest NR solver for a 200-variable system takes about 8.5 seconds on a 12-CPU workstation.

Mathematical Model: DDAE

- Dome can take into account delays leading to delayed differential algebraic equations (DDAE):

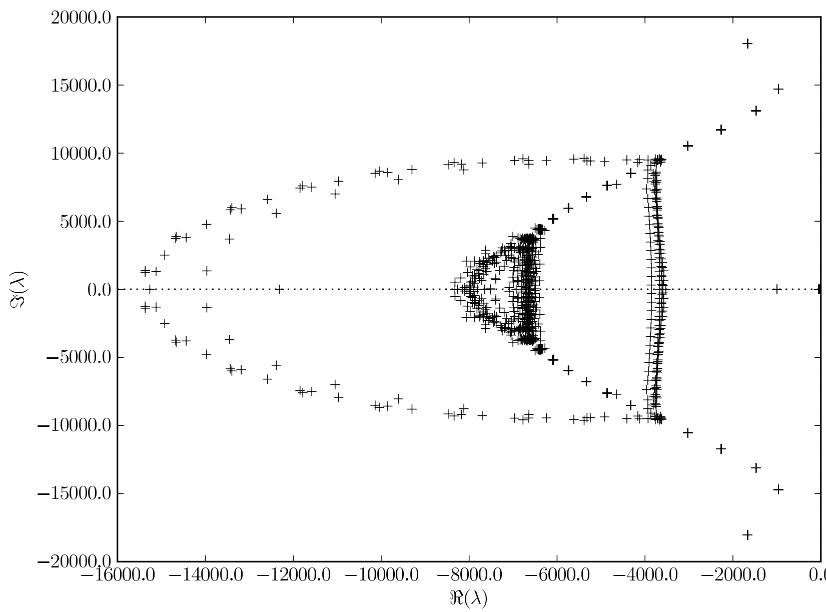
$$\begin{aligned} \dot{\boldsymbol{x}} &= \boldsymbol{f}(\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{x}_d, \boldsymbol{y}_d, \boldsymbol{\eta}, \boldsymbol{u}, t) \\ \boldsymbol{0} &= \boldsymbol{g}(\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{x}_d, \boldsymbol{y}_d, \boldsymbol{\eta}, \boldsymbol{u}, t) \end{aligned} \tag{3}$$

where $\boldsymbol{x}_d$ and $\boldsymbol{y}_d$ are delayed variables:

$$\begin{aligned} \boldsymbol{x}_d &= \boldsymbol{x}(t - \tau) \\ \boldsymbol{y}_d &= \boldsymbol{y}(t - \tau) \end{aligned}$$

and τ are the delays (non-necessarily time-independent).

Small-Signal Stability Analysis of Delayed Power Systems



- Dome allows finding approximated solutions of the characteristic equation of a DDAE.
- It turns to be the solution of a *huge* eigenvalue problem (thousands of eigenvalues!).

Some Statistics

- Dome currently includes:
  - **45** data format (e.g., PSS/E, GE PSLF, DigSilent, etc.)
  - **357** device models (e.g., synchronous machines and their primary controllers, FACTS, wind turbines, DERs, energy storage systems, etc.)
  - **10** power flow algorithms ranging from the standard NR to a GPU-based BFS.
  - **13** mathematical libraries (e.g., KLU, UMFPACK, SLUDIST, PETSC, MAGMA, etc.)
  - **Several** static and dynamic analysis tools (eigenvalue analysis, CPF, OPF, time domain analysis, short-circuit analysis, equivalencing techniques, polynomial recasting, electric vehicles management, etc.).

Supported Models

- Dome currently supports:
  - Standard quasi-static phasors representing single-phase equivalents of three-phase balanced and symmetrical devices
  - Park vectors
  - EMT models
  - DC, AC and power electronics devices
  - Three-phase unbalanced systems
  - Physical micro-controller devices (beta!)
  - A mix of all the above (with proper interfaces!)

Research

- Dome is an efficient research tool.
- It is currently used by a small group of researchers under the supervision of Federico Milano to study the following topics:
  - Modeling power systems through stochastic differential equations.
  - Stability of power system controllers including functional delays.
  - Parallellization of power flow analysis and time domain integration.
  - Detailed modeling of DERs and energy storage devices.

Power System Education

- Dome proved to be a challenging educational tool.
- It has been used at the University of Castilla-La Mancha for the lab activities of the course *Power System Control*.
- It is currently used for teaching basic power system control and stability concepts at Stage 4 of the Electrical Engineering program at UCD.

Benchmarking

- Dome can be used to test mathematical libraries.
- For example, one can test the performance of libraries for sparse matrix factorization.
- The table below refers to the power flow solution of the 1254-bus 1944-line network that models that UCTE 2002 Winter Off-peak.

Library	Total CPU time [s]	1^{st} fact. time [s]	Next fact. time [s]
KLU	0.0933	0.0044	0.0026
CXSPARSE	0.0936	0.0043	0.0027
UMFPACK	0.1750	0.0126	0.0095
SUPERLU	0.1927	0.0247	0.0082
LUSOL	0.3112	0.0360	0.0195

Consulting

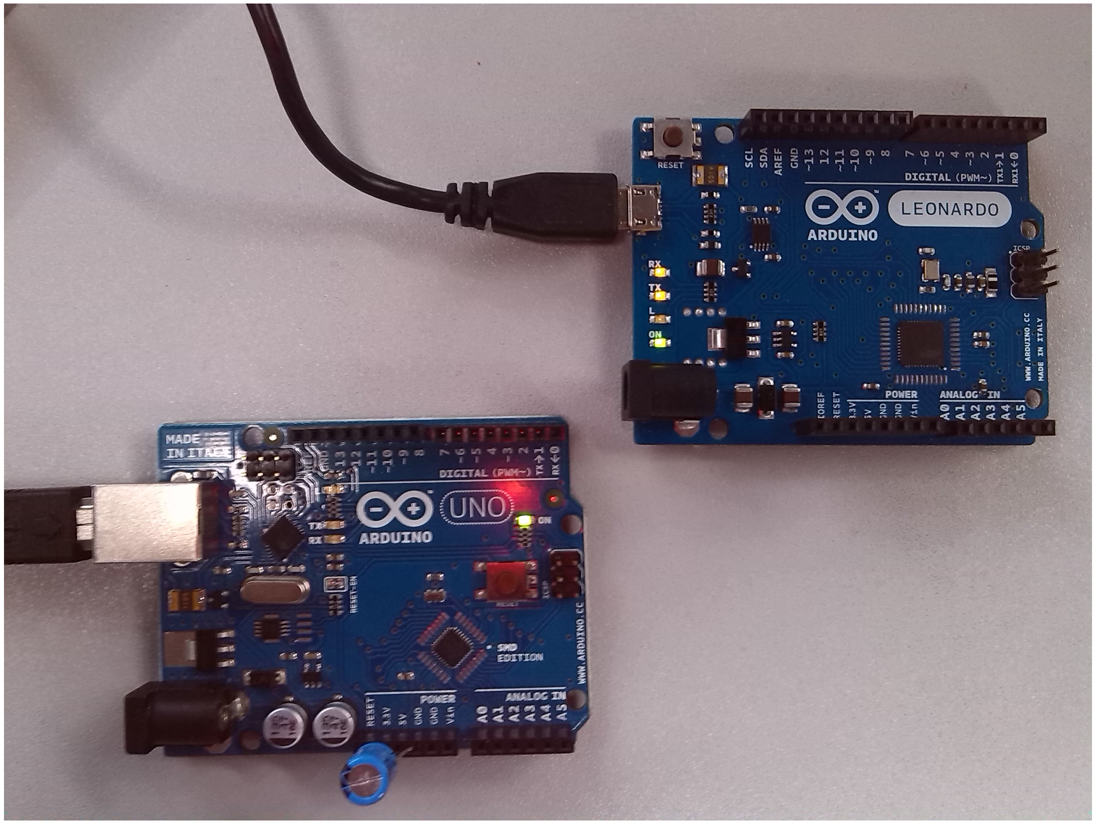
- Dome can be easily extended, can run on any platform supported by Python and the resulting code is 100% open source.
- These features makes Dome ideal for consulting activities.
- It has been successfully used to develop an optimal load management algorithm, data format filtering, stability analysis, etc., for a variety of European and American companies.
- A spin-out based on Dome is under evaluation at UCD.

Current Challenges

- Extensions that are currently under study and/or development are:
  - Real-time simulation capability
  - *Hardware-in-the-loop*
  - Capability of being a OPC (OLE for Process Control) server
  - Include diverse energy system models (e.g., gas and water)
  - Parallel computing based on heterogeneous architectures (CPUS and GPUs)

Hardware-in-the-loop with Arduino

- A very first attempt to include hardware-in-the-loop: Arduino micro-controllers.



To-Do List

- Graphical user interface (originally omitted by purpose!)
- Full support on Windows operating systems
- Improve documentation (e.g., automatic generation of device equations)
- Support for discrete-event-simulation (DES) models (see for example the tools developed at Oak Ridge National Laboratory, US)
- . . .

Try Dome!

- Further details and a demo version of Dome are available at:

  http://faraday1.ucd.ie/dome.html

- If you are interested in giving Dome a try, just let me know and I will open an account for you on my server.

UCD-ERC Software Tools

- A new page that collects information on software tools developed at ERC has been included in the ERC web-page:

  http://erc.ucd.ie/outputs/software

- The page can be accessed from the ERC main page:

  Menu Options $\longrightarrow$ Sotware Tools

Thanks for your attention!