Beam Instrumentation Simulation in Python

M. Gonzalez-Berges, D. Alves, A. Boccardi, V. Chariton, I. Degl'Innocenti, S. Jackson, J. Martínez Samblas
Accelerator Systems Department, European Organization for Nuclear Research (CERN)
Geneva, Switzerland

Abstract

The **design of acquisition electronics** for particle accelerator systems relies on simulations in various domains. System level simulation frameworks can integrate the results of specific tools with **analytical models** and **stochastic analysis**. This allows the designer to estimate the **performance of different architectures**, compare the results, and ultimately optimise the design. These simulation frameworks are often made of custom scripts for specific designs, which are hard to share or reuse. Adopting a **standard interface** for **modular components** can address these issues. Also, providing a **graphical interface**, where these components can be easily **configured**, **connected** and the **results visualised**, eases the creation of simulations. This paper identifies which characteristics **ISPy (Instrumentation Simulation in Python)** should fulfil as a simulation framework. It subsequently proposes a standard format for signal-processing simulation modules. **Existing environments** which allow script integration and have an intuitive graphical interface have then been evaluated with the **KNIME Analytics Platform** being the proposed solution. Additionally, the need to handle **parameter sweeps** for all simulation parameters and the need for a bespoke **visualisation tool** will be discussed. **Python** has been chosen for all of these developments due to its flexibility and its wide adoption in the scientific community. The ensuing performance of the tool will also be discussed.

Keywords: simulation, system design, digital twin, Python, performance

Requirements & Choice of Environment

- Main Requirements
  - Based on components representing acquisition system elements
    - E.g. beam characteristics, filters, cables, amplifiers
  - Extendable without technical knowledge of the tool internals
  - Interaction through a GUI
    - Define schematic & parameters + Explore & analyze results
  - Possibility to run simulations outside the tool (CLI)
  - Open-source principles
  - Python based or facilities to integrate Python modules
- Based on our requirements several candidates were considered:

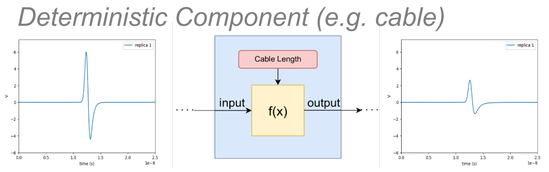
PyQt custom application — KNIME Analytics Platform — Kepler Project

- KNIME finally selected:
  - Large and dynamic community
  - Extensive functionality
  - Python extensions

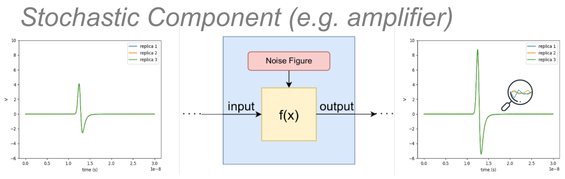
Integration in KNIME

Component Standardization:
- Python source code
- Configuration files (XML-SXD)

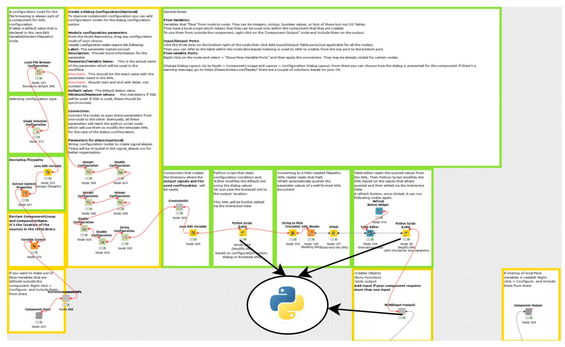
Deterministic Component (e.g. cable)



Stochastic Component (e.g. amplifier)

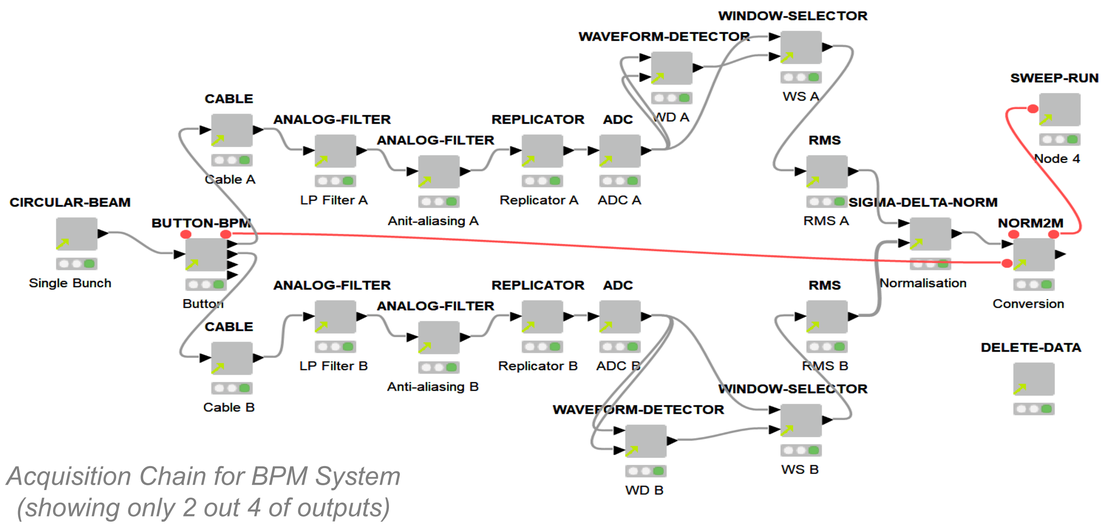


KNIME template for including new components



First Experience

- Full chain simulation of a Beam Position Monitoring (BPM) system
  - Horizontal & Vertical positions = 4 processing chains
- Mix of deterministic (e.g. cable) and stochastic modules (e.g. ADC)
- Several parameters are swept:
  - Beam horizontal position, beam number of charges



Acquisition Chain for BPM System (showing only 2 out 4 of outputs)

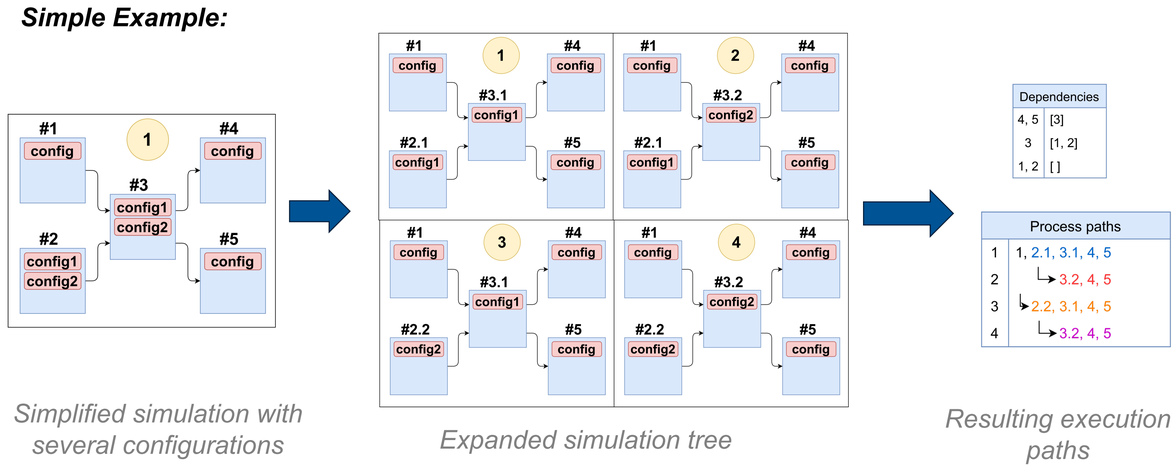
RowID	unit_multiplier	unit	start	stop	python_function	num / step
#4(AnalogComponents)_LP_Wn	K	Hz	0.1	1	arange(start, stop, step)	1
#2(Monitors)_x	m	m	-3	-3	linspace(start, stop, num)	3
#2(Monitors)_y	m	m	-2	2	linspace(start, stop, num)	3

Sweep parameters configuration example

Parameter Sweep Simulations

- Standard technique to find optimal parameters
- Each component can have multiple configurations
  - Defined individually or with some logic (e.g. linear range, logarithmic steps)
- Optimized execution to avoid duplicates
  - modified **Depth First Search (DFS)** algorithm on a directed graph

Simple Example:



Simplified simulation with several configurations

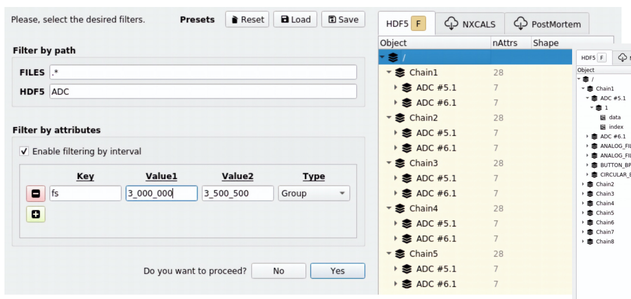
Expanded simulation tree

Resulting execution paths

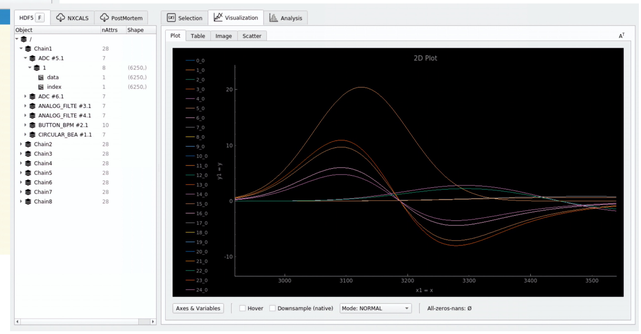
Results Storage & Visualization

Storage: HDF5 format selected:
- Group together all results & metadata
- Tools & APIs (Python, Java, C/C++)
- Extensive functionality (e.g. caching, compression, etc)

Visualization:
- DAVIT (Data Analysis and Visualization Tool)
- General purpose tool
- Python based (PyQtGraph, Pandas, etc)
- Integrate data from many sources: simulation results, logging system, post-mortem



Filtering of File System & HDF5 structure



Visualization of Simulation with Parameter Sweep

Conclusion

- Complex and iterative process for designing acquisition electronics involves optimizing a large number of parameters
- Instrument Simulation in Python (ISPy) developed as a modular framework to handle simulations, including a user-friendly GUI, efficient handling and visualization of results, and parameter sweeping
- KNIME Analytics platform chosen to due to its open-source model and available functionality
- Keep open-source principles for the development
- In use for the Beam Position system for the future LHC upgrade
  - The library will be extended to other systems (e.g. Fast Beam Current Transformers)

CERN Accelerator Systems Department
Beam Instrumentation Group (BI)

Contact: manuel.gonzalez@cern.ch

Contribution ID: THPDP060

ICALEPCS CAPE TOWN SOUTH AFRICA 2023