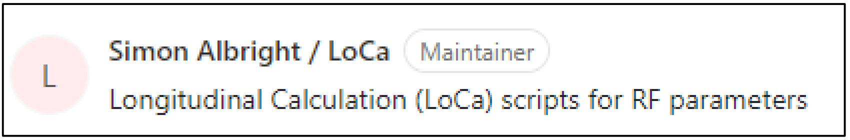
BLonD satellites

BLonD meeting – 14/06/2019

S. Albright, A. Lasheen,
Acknowledgements and contributions:
A. Farricker, B. Popovic, J. Repond, M. Schwarz

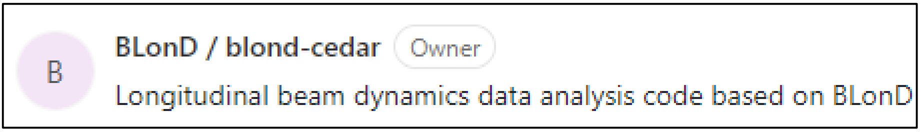
Present status, satellite projects



Simon Albright / LoCa

Longitudinal Calculation (LoCa) scripts for RF parameters

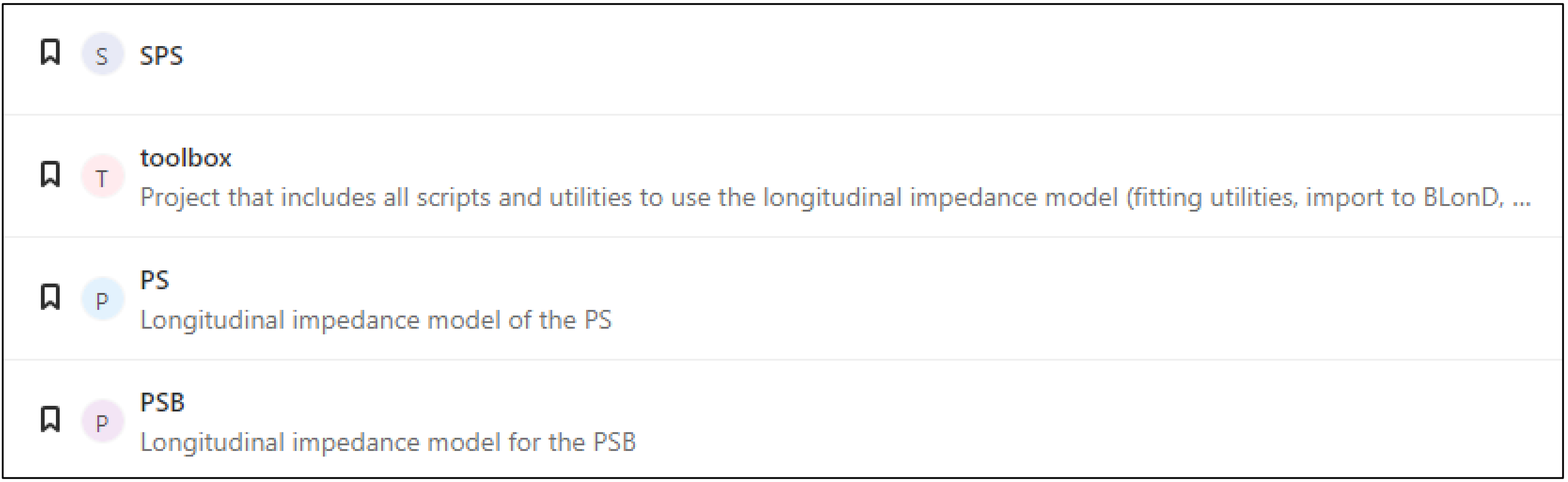
- LoCa: longitudinal calculations (buckets, emittances, ramp optimization, etc…) https://gitlab.cern.ch/salbrigh/LoCa



BLonD / blond-cedar

Longitudinal beam dynamics data analysis code based on BLonD

- Cedar: offline data analysis routines (loading data from different sources, bunch fitting, mode/spectral analysis, etc…) https://gitlab.cern.ch/blond/blond-cedar



SPS

toolbox

Project that includes all scripts and utilities to use the longitudinal impedance model (fitting utilities, import to BLonD, …

PS

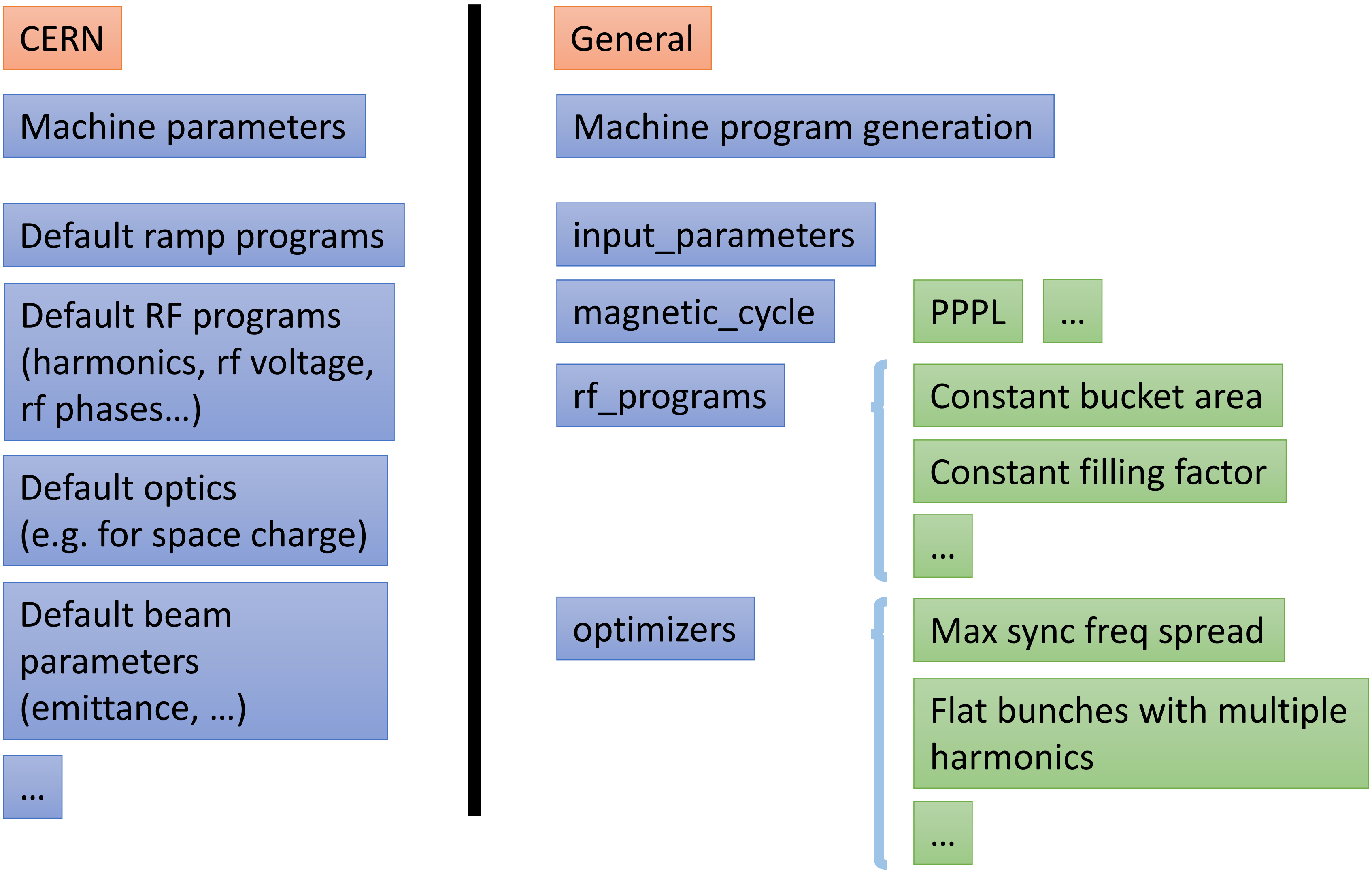
Longitudinal impedance model of the PS

PSB

Longitudinal impedance model for the PSB

- Longitudinal impedance toolbox and databases: impedance model storage and scripting (toolbox to handle different types of impedance sources, fitting with resonators, calculation of rf losses and beam spectra, includes atm PSB/PS/SPS) https://gitlab.cern.ch/longitudinal-impedance

LoCa



CERN

- Machine parameters
- Default ramp programs
- Default RF programs (harmonics, rf voltage, rf phases…)
- Default optics (e.g. for space charge)
- Default beam parameters (emittance, …)
- …

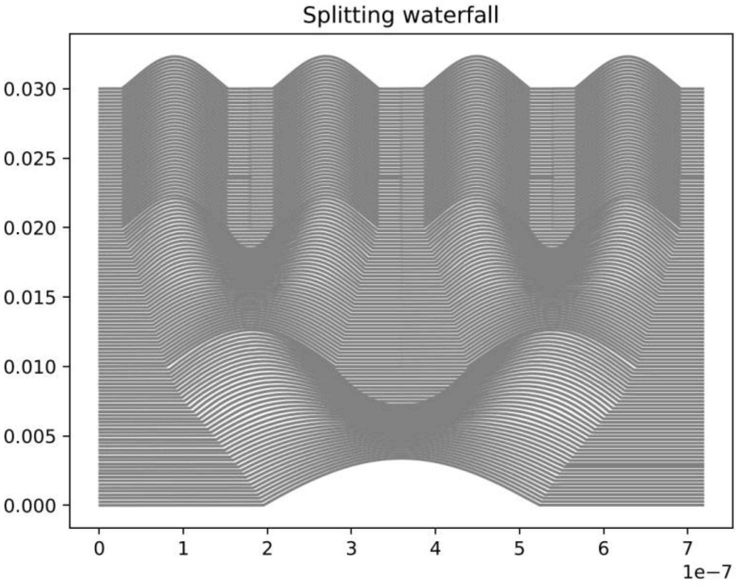
General

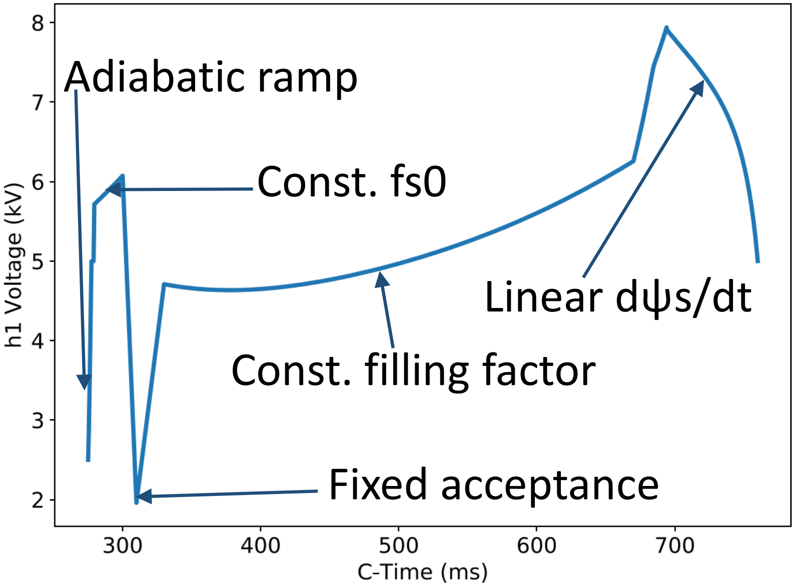
- Machine program generation
- input_parameters
- magnetic_cycle
  - PPPL
  - …
- rf_programs
  - Constant bucket area
  - Constant filling factor
  - …
- optimizers
  - Max sync freq spread
  - Flat bunches with multiple harmonics
  - …

- LoCa: longitudinal calculations (buckets, emittances, ramp optimization, etc…)

https://gitlab.cern.ch/salbrigh/LoCa

LoCa, examples


Splitting waterfall

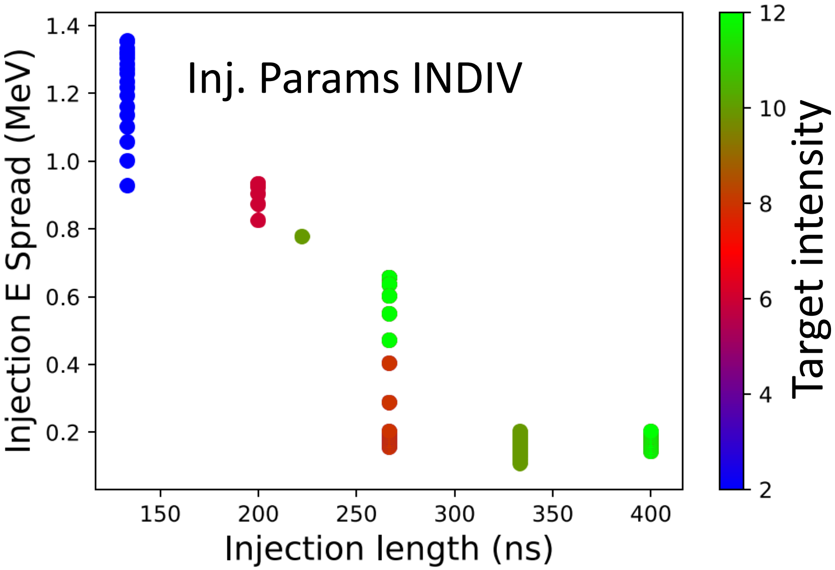

Adiabatic ramp

Const. fs0

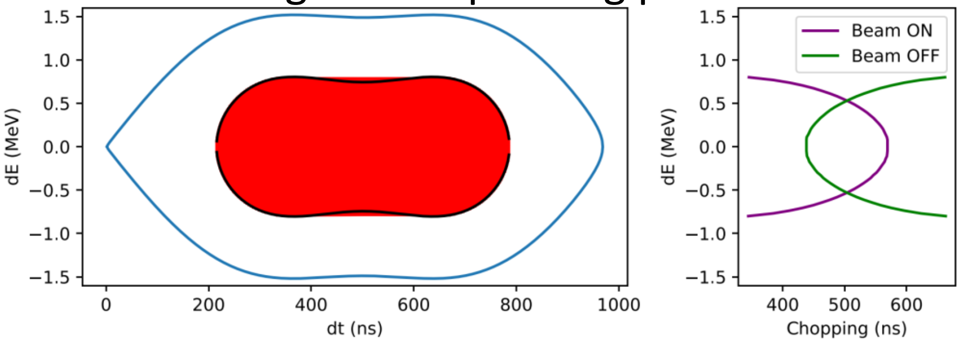
Linear dψs/dt

Const. filling factor

Fixed acceptance


Inj. Params INDIV

Longitudinal painting parameters



LoCa, present status and next steps

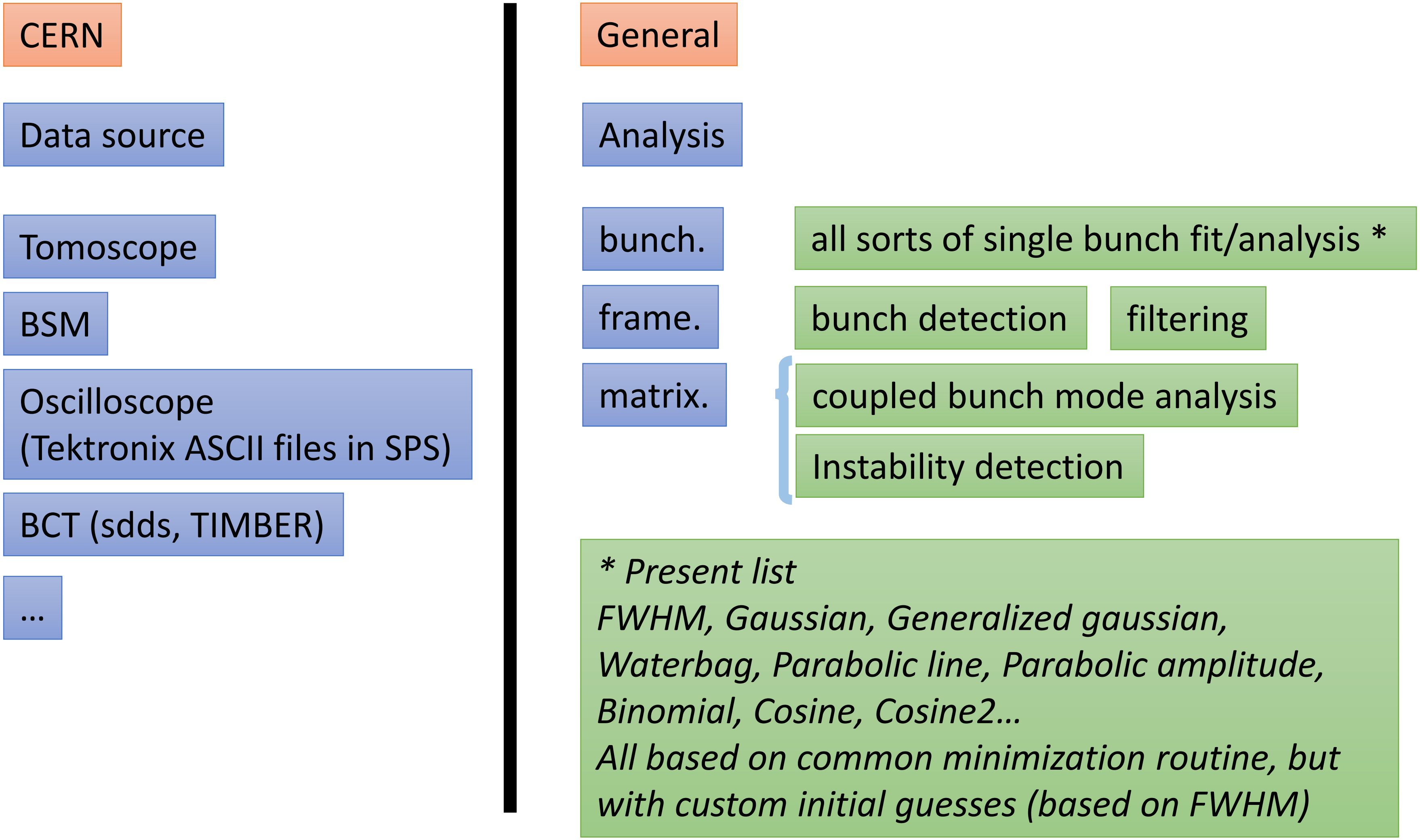
- Based on Simon/Alex’s original developments for PSB/PS/SPS machine programs calculation
- Used mostly for studies, some programs exported for operation

- +++
  - Analysis of potential wells for very complex configurations (several harmonics, rf manipulations, off-momentum configurations, …)
  - Different interpolation methods (linear, cubic spline)
  - First basis for machine program using optimizers

- ---
  - Parts also included in BLonD, duplication to be minimized
  - Need more common effort to foresee all use cases

- Next steps
  - Improve synergy with the other codes

Cedar



CERN

Data source

Tomoscope

BSM

Oscilloscope
(Tektronix ASCII files in SPS)

BCT (sdds, TIMBER)

…

General

Analysis

bunch.

frame.

matrix.

all sorts of single bunch fit/analysis *

bunch detection

filtering

coupled bunch mode analysis

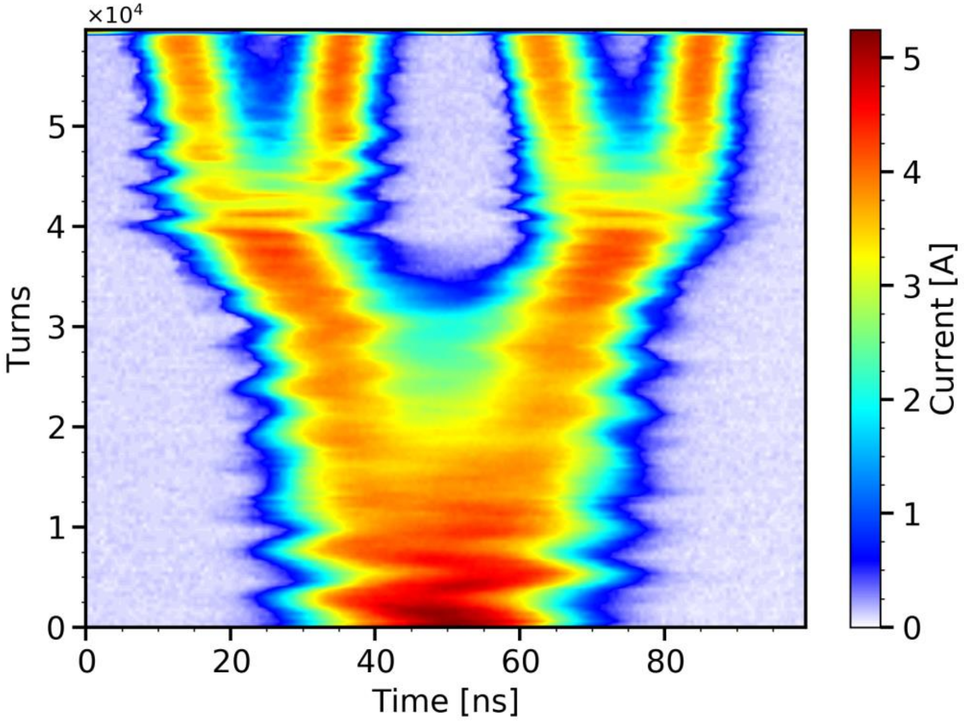
Instability detection

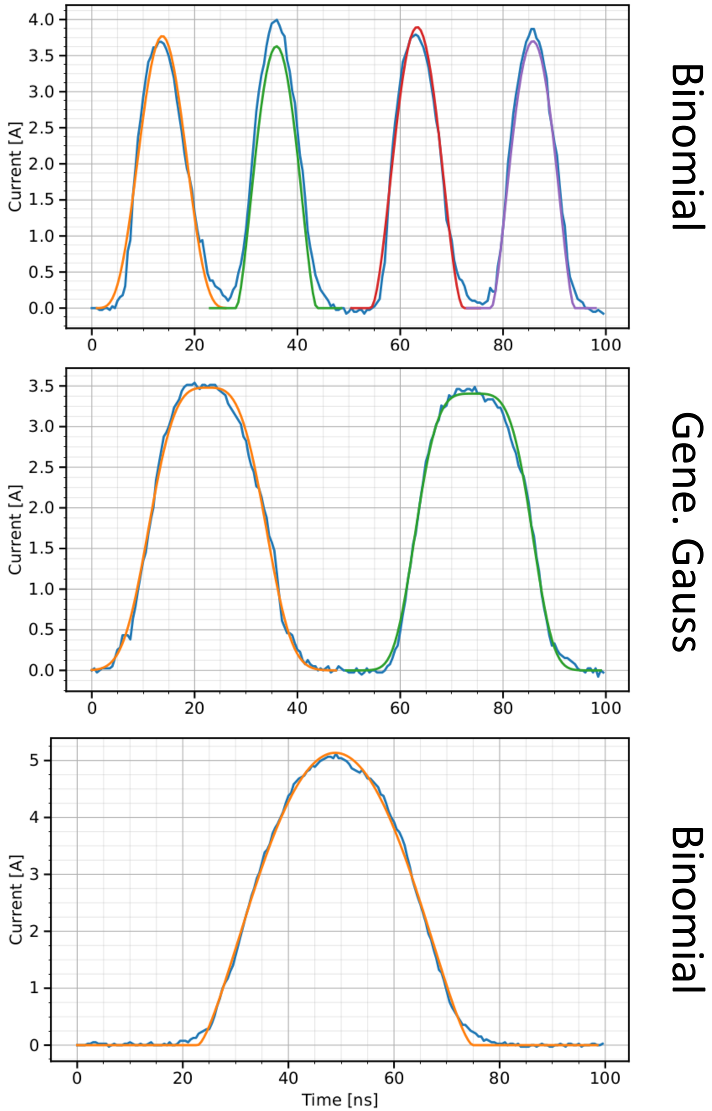
** Present list*
FWHM, Gaussian, Generalized gaussian, Waterbag, Parabolic line, Parabolic amplitude, Binomial, Cosine, Cosine2…
All based on common minimization routine, but with custom initial guesses (based on FWHM)

- Cedar: offline data analysis routines (loading data from different sources, bunch fitting, mode/spectral analysis, etc…)

https://gitlab.cern.ch/blond/blond-cedar

Cedar, example





- Test case in the PS using tomoscope reference files
- Two functions used, one to detect bunches and return their indexes, one to fit the bunches

Cedar, present status and next steps

- Based on Alex/Simon's original developments for PSB/PS/SPS, used mostly for offline data analysis
- Newdev branch started to have all the functions standalone (e.g. fitting)
- +++
  - Large amount of functionalities
  - Used for various cases (single, multi-bunch…)
  - Correlation of old data (beam/BCT) from different sources based on timestamps
- ---
  - Original structure around "MD" object, which is a blackbox
- Next steps
  - Newdev branch is already well advanced, only need to copy paste functions
  - Documentation

Impedance toolbox + database

CERN

Impedance databases and scenario

PSB PS SPS …

Default machine parameters

Beam parameters

Machine parameters

RF parameters

General

Toolbox

Handle impedance (for BLonD)

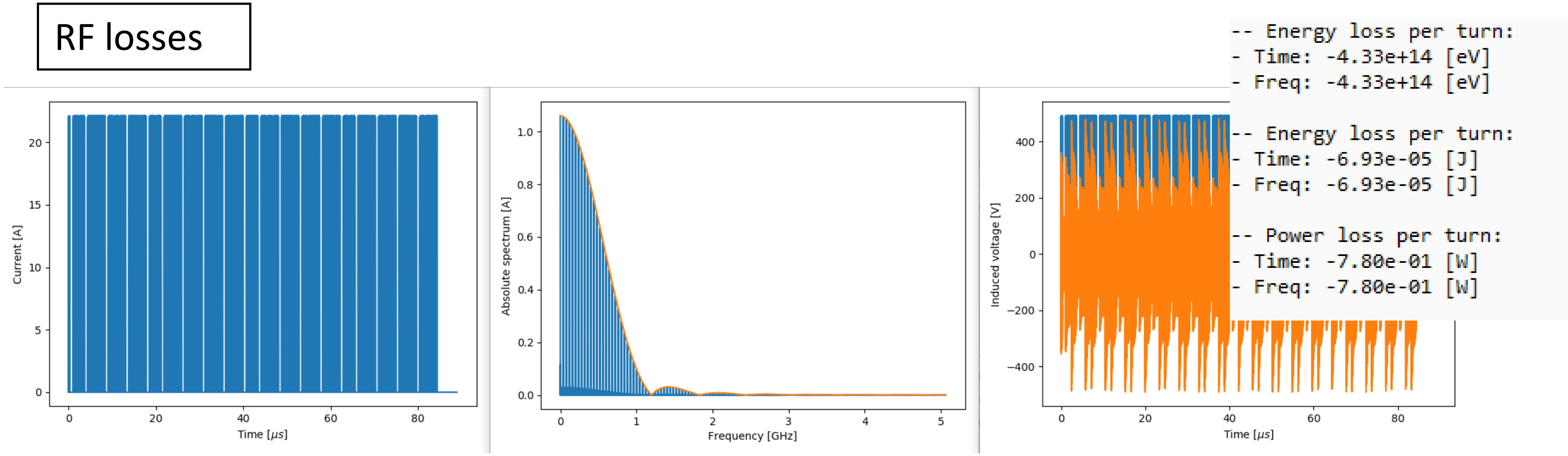
Basic effective impedance (e.g. rf losses)

Resonator fitting

- Longitudinal impedance toolbox and databases: impedance model storage and scripting (toolbox to handle different types of impedance sources, fitting with resonators, calculation of rf losses and beam spectra, includes atm PSB/PS/SPS) https://gitlab.cern.ch/longitudinal-impedance

Impedance toolbox, examples

RF losses

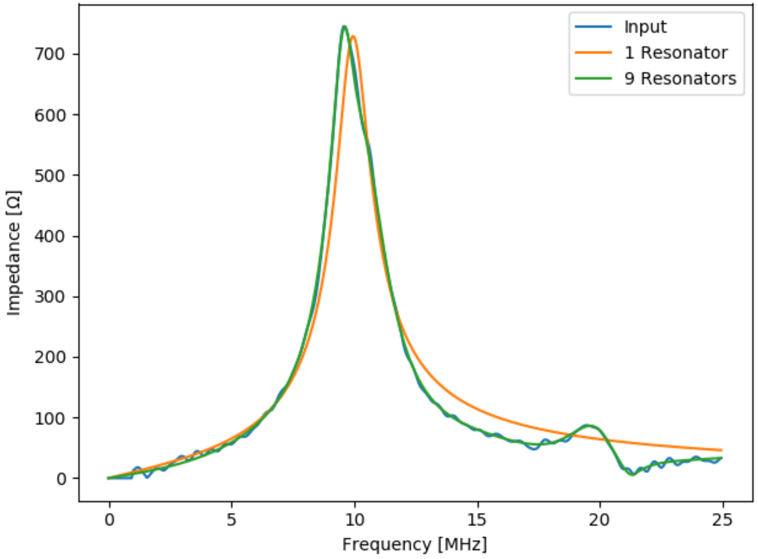


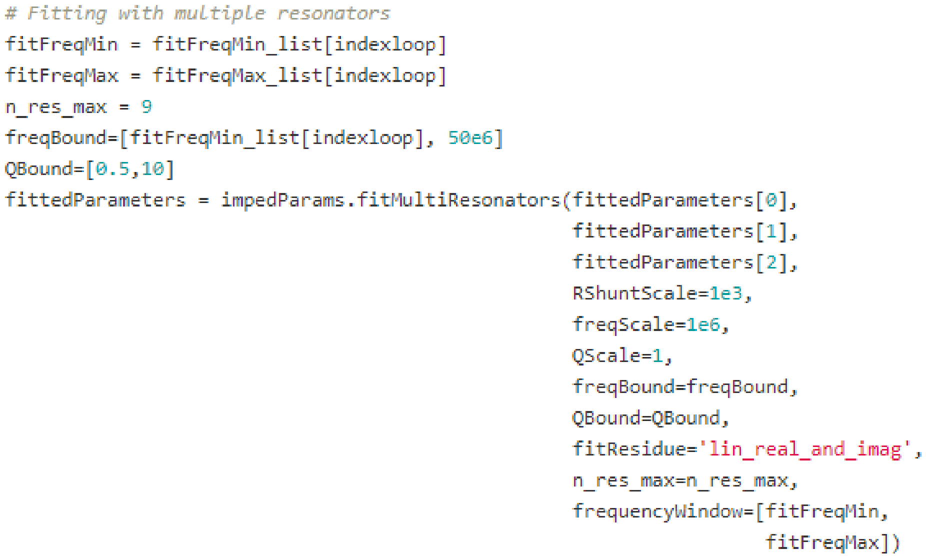
```
-- Energy loss per turn:
- Time: -4.33e+14 [eV]
- Freq: -4.33e+14 [eV]

-- Energy loss per turn:
- Time: -6.93e-05 [J]
- Freq: -6.93e-05 [J]

-- Power loss per turn:
- Time: -7.80e-01 [W]
- Freq: -7.80e-01 [W]
```

Fitting with resonators

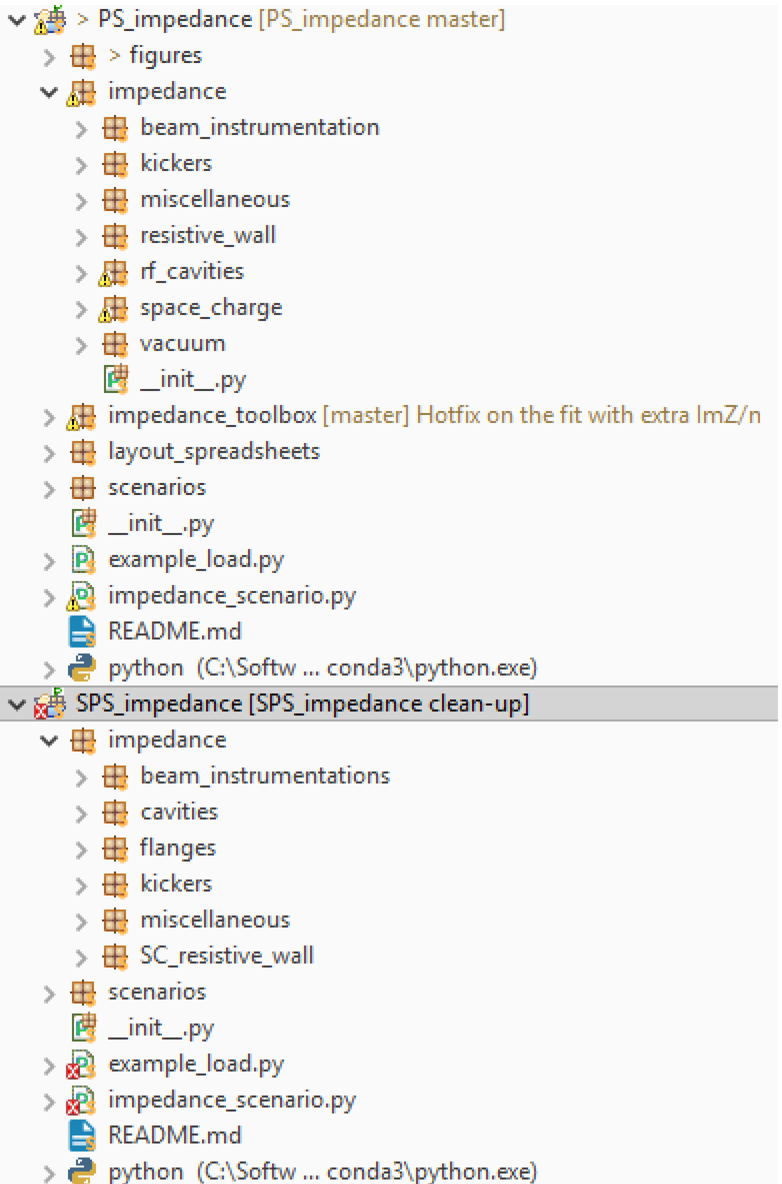




```
# Fitting with multiple resonators
fitFreqMin = fitFreqMin_list[indexloop]
fitFreqMax = fitFreqMax_list[indexloop]
n_res_max = 9
freqBound=[fitFreqMin_list[indexloop], 50e6]
QBound=[0.5,10]
fittedParameters = impedParams.fitMultiResonators(fittedParameters[0],
                                                  fittedParameters[1],
                                                  fittedParameters[2],
                                                  RShuntScale=1e3,
                                                  freqScale=1e6,
                                                  QScale=1,
                                                  freqBound=freqBound,
                                                  QBound=QBound,
                                                  fitResidue='lin_real_and_imag',
                                                  n_res_max=n_res_max,
                                                  frequencyWindow=[fitFreqMin,
                                                                   fitFreqMax])
```

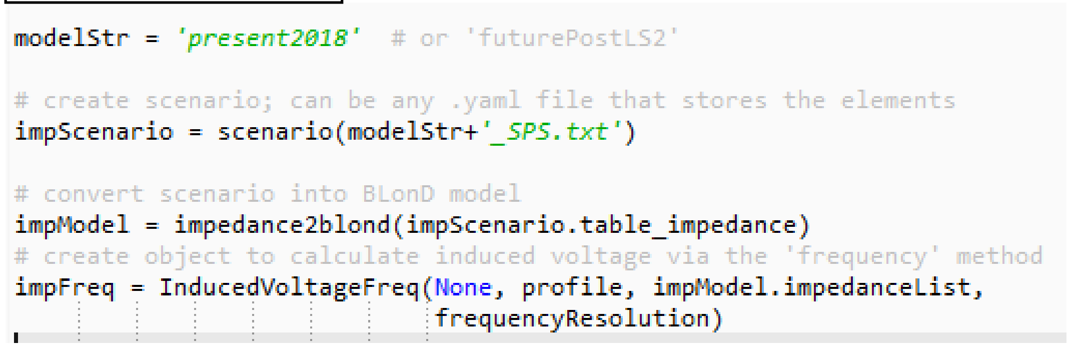
Impedance database, examples

Structure



```
PS_impedance [PS_impedance master]
    > figures
    impedance
        beam_instrumentation
        kickers
        miscellaneous
        resistive_wall
        rf_cavities
        space_charge
        vacuum
        __init__.py
    impedance_toolbox [master] Hotfix on the fit with extra ImZ/n
    layout_spreadsheets
    scenarios
    __init__.py
    example_load.py
    impedance_scenario.py
    README.md
    python  (C:\Softw ... conda3\python.exe)
SPS_impedance [SPS_impedance clean-up]
    impedance
        beam_instrumentations
        cavities
        flanges
        kickers
        miscellaneous
        SC_resistive_wall
    scenarios
    __init__.py
    example_load.py
    impedance_scenario.py
    README.md
    python  (C:\Softw ... conda3\python.exe)
```

Input to BLonD

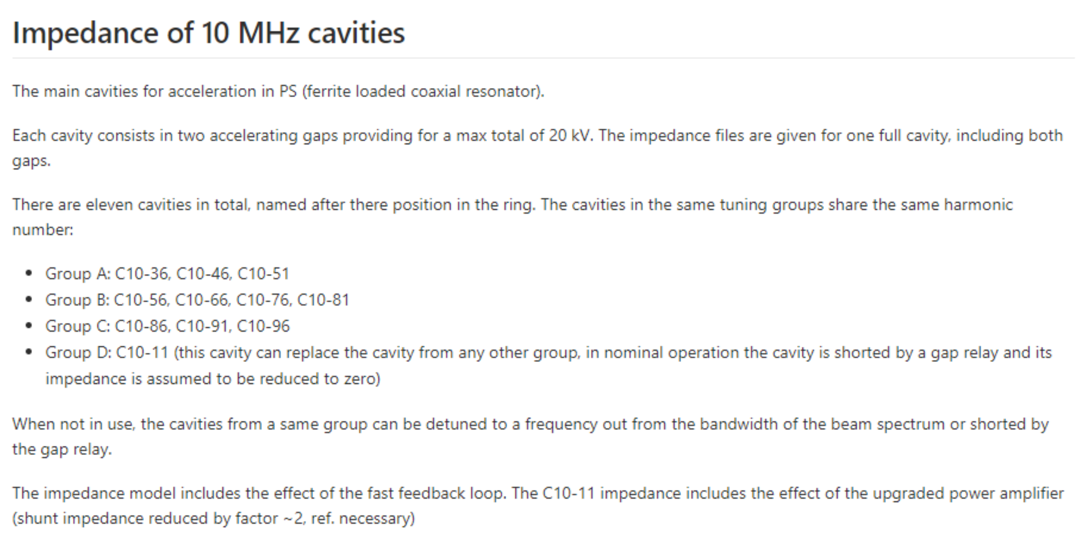


```
modelStr = 'present2018'  # or 'futurePostLS2'

# create scenario; can be any .yaml file that stores the elements
impScenario = scenario(modelStr+'_SPS.txt')

# convert scenario into BLonD model
impModel = impedance2blond(impScenario.table_impedance)
# create object to calculate induced voltage via the 'frequency' method
impFreq = InducedVoltageFreq(None, profile, impModel.impedanceList,
                             frequencyResolution)
```

Documentation



Impedance of 10 MHz cavities

The main cavities for acceleration in PS (ferrite loaded coaxial resonator).

Each cavity consists in two accelerating gaps providing for a max total of 20 kV. The impedance files are given for one full cavity, including both gaps.

There are eleven cavities in total, named after there position in the ring. The cavities in the same tuning groups share the same harmonic number:

- Group A: C10-36, C10-46, C10-51
- Group B: C10-56, C10-66, C10-76, C10-81
- Group C: C10-86, C10-91, C10-96
- Group D: C10-11 (this cavity can replace the cavity from any other group, in nominal operation the cavity is shorted by a gap relay and its impedance is assumed to be reduced to zero)

When not in use, the cavities from a same group can be detuned to a frequency out from the bandwidth of the beam spectrum or shorted by the gap relay.

The impedance model includes the effect of the fast feedback loop. The C10-11 impedance includes the effect of the upgraded power amplifier (shunt impedance reduced by factor ~2, ref. necessary)

Impedance, present status and next steps

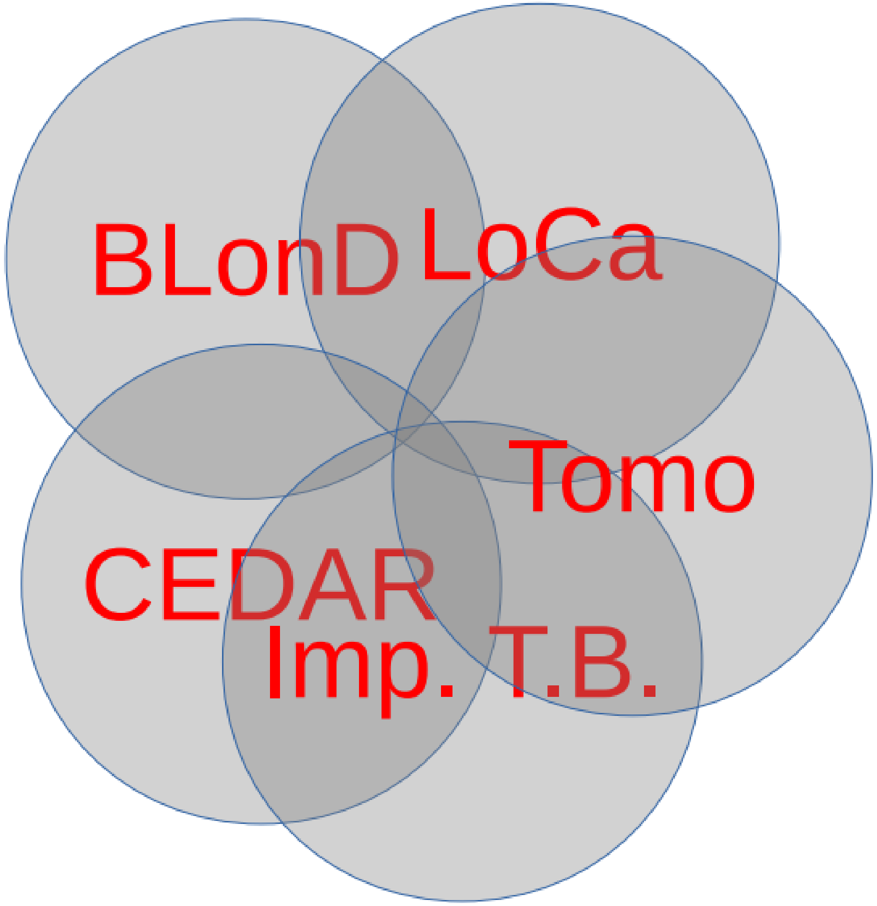
- Based on Joël’s original developments for SPS impedance
- PS and PSB followed the same structure as a start

- +++
  - Impedance import in one line for default model
  - Version control of impedance model
  - Customization of impedance model for studies (e.g. scan R, Q…)

- ---
  - Customization of impedance model through objects, not friendly

- Next steps
  - Define a better common infrastructure to handle impedance
  - Need to define a database for CST models with impedance team (for referencing)

Code overlap

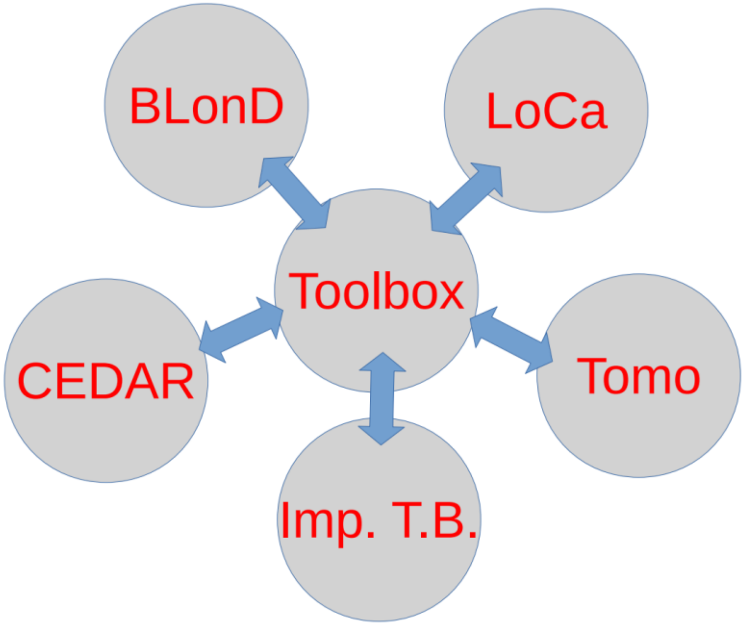


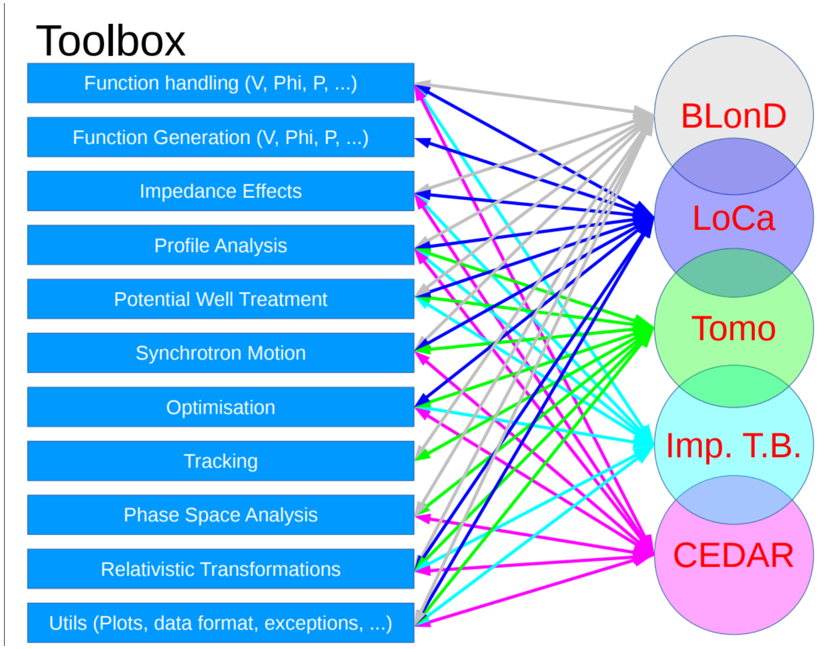
5 Independent codes
5 Independent applications
A lot of overlap

Can we use the overlap to minimise repetitive/ parallel/ duplicated development?

First attempt by Alex with submodules and subtrees showed promise, but has problems

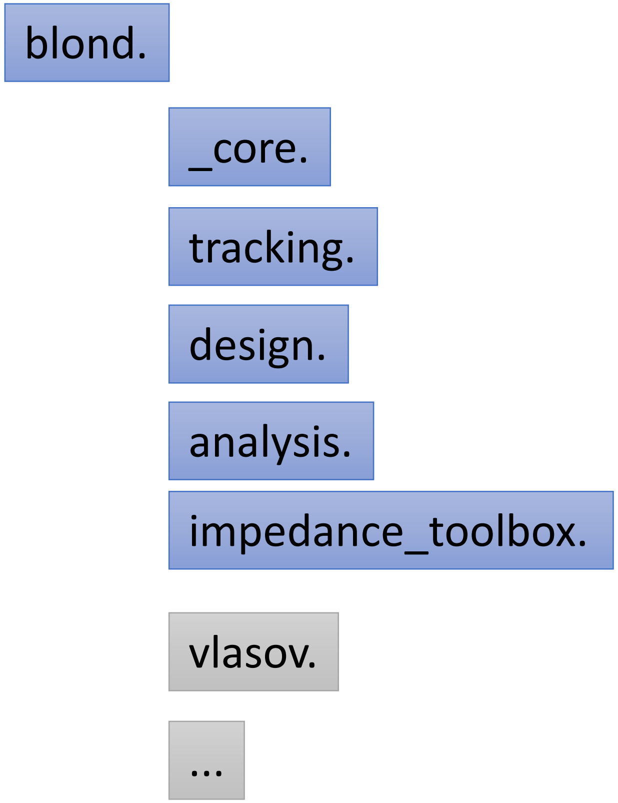
Structure proposal – Central toolbox





- One central toolbox package, which is a separate project from the existing one
- Existing projects stay independent and can pick into the toolbox.
- The toolbox is either included as a submodule, or set as a requirement for installation like for numpy, scipy etc.
- The functionalities are progressively included into the toolbox. Then the different projects are gradually updated using the toolbox as a prerequisite.
- The user only uses the projects he is interested in.

Structure proposal – “BLonD suite”



blond.

_core.

tracking.

design.

analysis.

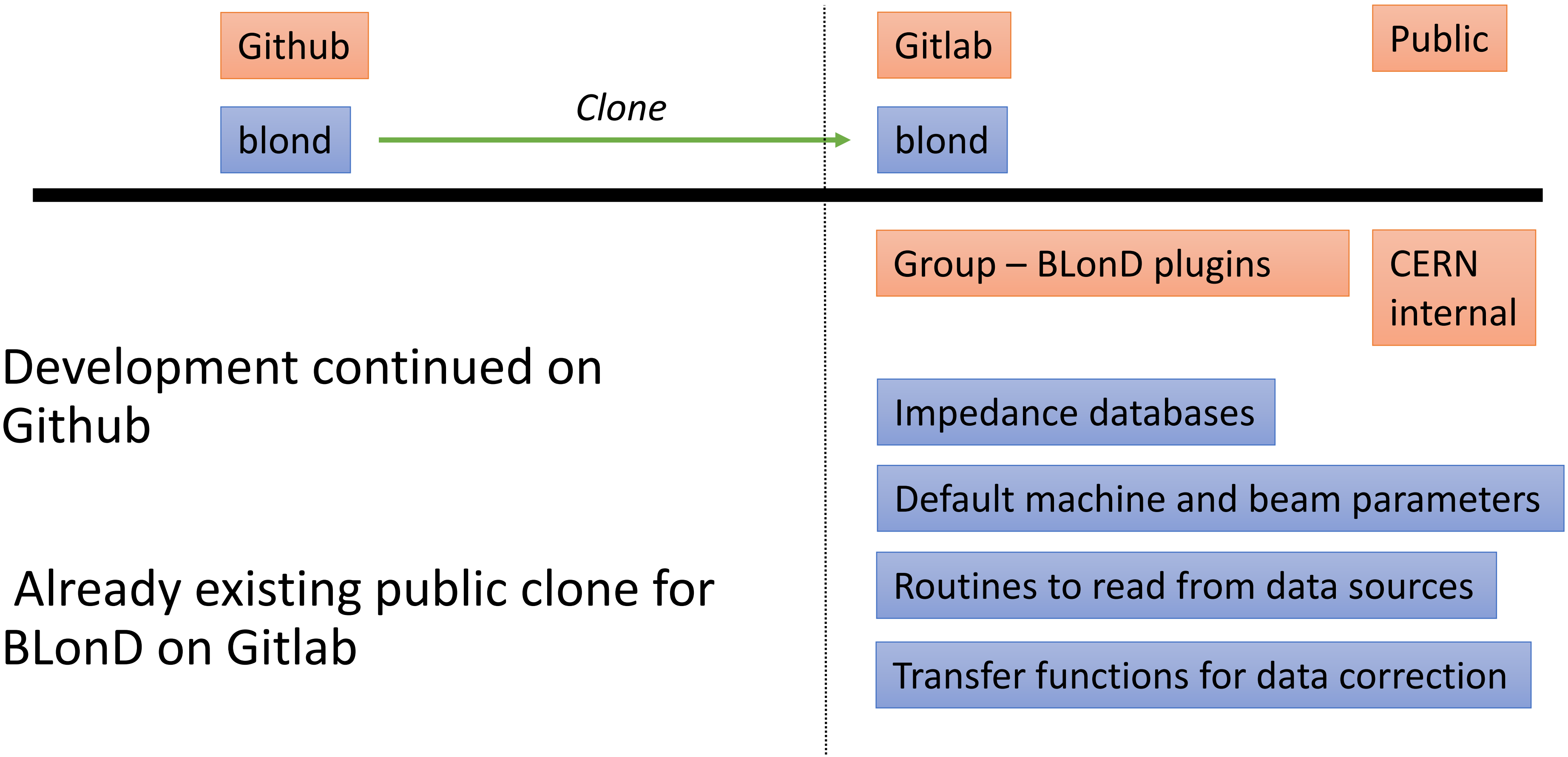
impedance_toolbox.

vlasov.

…

- One _core package, where common functions and dependencies are located (acts as the “toolbox”)
- One package per functionality, that picks functions and inherits from common objects in the _core package.
- The tracking package is the present version of BLonD, that can be encapsulated here as a whole.
- Common functions and classes can be migrated gradually to the _core package.
- The user only uses the package(s) he is interested in.
- Only the “public” parts of the codes are included, the “CERN internal” parts can be kept as plugins available on Gitlab.

Git management



Github

blond

Clone

Gitlab

blond

Public

Group – BLonD plugins

CERN internal

Impedance databases

Default machine and beam parameters

Routines to read from data sources

Transfer functions for data correction

- Development continued on Github
- Already existing public clone for BLonD on Gitlab
- CERN related packages can go in a subgroup “Plugins”, limited to CERN visibility. Can be mirrors of other existing projects on Gitlab