Prefactor - The LOFAR pre-facet calibration pipeline

Overview

The official repository of prefactor is now on GitHub: https://github.com/lofar-astron/prefactor

It consists of parsets for the genericpipeline that do the first calibration of LOFAR data. Originally in order to prepare said data for the Factor facet calibration (https://github.com/lofar-astron/factor), but also useful if you don't plan to run Factor.

It includes:

- clock-TEC separation with transfer of clock from the calibrator to the target
- some flagging and averaging of amplitude solutions
- diagnostic plots
- at least some documentation

There are several pipeline parsets in this repository:

- **Pre-Facet-Cal.parset** : The “standard” pre-facet calibration pipeline, works on pre-NDPPP'ed data
- **Pre-Facet-Cal-RawData-Single.parset** : A pre-facet pipeline to work on raw (non NDPPP'ed) data
- **Pre-Facet-Cal-RawData-PreAvg.parset** : A pre-facet pipeline to work on raw (non NDPPP'ed) data that does the subband concatenating in the first NDPPP step. (To reduce the number of files on systems where this is a problem, e.g. JURECA)
- **Initial-Subtract.parset** : A pipeline that generates full FoV images and subtracts the sky-models from the visibilities. (Needed for facet-calibration, this could also be done as the first step of Factor.)

Software requirements

- the full “offline” LOFAR software installation version >= 2.15
  (With small modifications the Pre-Facet-Cal pipelines can be run with older versions, but that is not supported by the authors anymore.)
- LoSoTo
- LSMTool (see https://github.com/darafferty/LSMTool)
- Python-PP (see http://www.parallelpython.com/ or https://pypi.python.org/pypi/pp)
- Python matplotlib
- WSClean (for Initial-Subtract, version >=1.9)

Documentation

Installation and setup is explained in the preliminary version of a cookbook chapter at:
https://github.com/lofar-astron/prefactor/blob/pdf-doc/docs/cookbook_prefacet.pdf

Please read that first!

Usage Notes

- Don't edit the original parset files directly. Make a copy with a descriptive name (e.g. `Pre-Facet-Cal-calibrator-3c295.parset`) and edit that copy.
  - There is also no need to change the `runtime_directory` and `working_directory` entries in the `pipeline.cfg` for different pipeline runs. The pipeline framework will generate sub-directories with the job-name in there.
- calibrator and target data have to match:
  - observed with the same selection of stations (If calibrator and target observations have the same number of stations but different stations, then the pipeline will not fail but produce wrong results.)
  - observed close enough in time that calibration values can be transferred
- For each observation you should process all the calibrator data at once together. (Clock/TEC separation and flagging of bad amplitudes work better with the full bandwidth.)
  - The target data can be processed one time- or frequency- block at a time.
- Yes, updating the LOFAR software can be annoying. But it is worth ist! Trust me. 
- Get someone to write a better documentation for the genericpipeline.
- Don't forget to update the recipe_directories entry in the pipeline.cfg to include the pre-factor plugins, for example:

```
 recipe_directories =
[%(pythonpath)s/lofarpipe/recipes,/home/rvweeren/software/prefactor]
```

FAQ

BLAS Core affinity : Your pipeline runs slow. All NDPPP-/BBS-/whatever- processes use only little CPU time and only one core of the node is busy.
On clusters like CEP-3 the OpenBLAS library is built with threading affinity. This means that by default the different processes all try to use the same core(s). The `use LofIm` and `use Lofar` scripts set an environment variable that disables this threading affinity, but if the `pipeline.cfg` file does not have the `[remote]` section included, then this environment variable is not forwarded to the processes that are started by the pipeline.
So please set the `[remote]` section in your `pipeline.cfg`.

KeyError 'mapfile' : Your pipeline run fails like that:

```
2016-02-07 14:48:58 ERROR   genericpipeline:
****************************************
2016-02-07 14:48:58 ERROR   genericpipeline: Failed pipeline run: Pre-Facet-
Cal
2016-02-07 14:48:58 ERROR   genericpipeline: Detailed exception information:
2016-02-07 14:48:58 ERROR   genericpipeline: <type 'exceptions.KeyError'>
2016-02-07 14:48:58 ERROR   genericpipeline: 'mapfile'
2016-02-07 14:48:58 ERROR   genericpipeline:
```

```
*****************************************
```

That happens when one step didn't generate a mapfile. Usually that means that the pipeline was looking for its input data, but couldn't find any files that match. Please check your `*_input_path` and `*_input_pattern` in the parset file! (Note: `ls -d *_input_path/*_input_pattern` should find your data.)

PipelineStep_* missing : Your pipeline run fails like that:

```
2016-02-04 13:33:56 ERROR   genericpipeline:
*****************************************
2016-02-04 13:33:56 ERROR   genericpipeline: Failed pipeline run: Pre-Facet-
Cal
2016-02-04 13:33:56 ERROR   genericpipeline: Detailed exception information:
2016-02-04 13:33:56 ERROR   genericpipeline: <type 'exceptions.ImportError'>
2016-02-04 13:33:56 ERROR   genericpipeline: No module named
PipelineStep_createMapfile
2016-02-04 13:33:56 ERROR   genericpipeline:
*****************************************
```

(The exact name of the missing module varies.) You are probably missing one of the entries in the `recipe_directories` setting in your `pipeline.cfg`, or one of those entries doesn't work. Make sure both entries point to the correct directories, and that the missing module can be found in the `plugins` subdirectory of one of those two entries.

Missing “h5imp_cal_losoto.h5”: Your pipeline run fails with an “`executable_args failed`” error and in the logfile you can find something like:

```
"/usr/local/lofar/losoto/current/lib/python2.7/site-packages/losoto-1.0.0-
py2.7.egg/losoto/h5parm.py",
line 40, in __init__
    raise Exception('Missing file '+h5parmFile+'.')
Exception: Missing file ./h5imp_cal_losoto.h5.
```

That was caused by a bug in an old version of the genericpipeline. Update the software, make sure you use the new version of the software when starting the pipeline, and check the pathes in `pipeline.cfg` that they use the new version!

To do list

Pre factor

- RMwriter
- Write script for automatic flagging on the calibrator and the in the pipeline rerunning of the calibrator after this flagging
- TGSS global sky model (not really something to develop)
- Collect phase calibration solutions and fit to these and apply the fits (Reinout has script).
- Check fix for observations without continuous frequency coverage
- Move to NDPPP (waiting for NDPPP developments).

Subtract pipeline

- Create a deep image by stacking 10subband images and use this image or catalog of this image to help create facets and avoid sources at the boundaries.

FACTOR

- Time correlation settings given by user.
- Aplpy plots (Tammo has script on his github but change noise to 1mJy (current 5mJy))
- All WSclean (any aliasing issues?)
- pybdsm boxes from restoring beam size.
- outer uvrange option (maybe good for help create models for selfcalbration when working at full resolution)
- Check that amplitudes are properly normalised
- Second major iteration
- High dynamic range option
- optimize number of IO-intesive (NDPPP) processes