Schweizerische Eidgenossenschaft
Confédération suisse
Confederazione Svizzera
Confederaziun svizra

Eidgenössisches Departement des Innern EDI
Bundesamt für Meteorologie und Klimatologie MeteoSchweiz

Meteorological Support for MALSplus

Data Processing Cookbook

Version: 0.3.0
Date: October, 18, 2017

Document History

Responsible People

Creation/Edition	J. Grazioli
Revision	
Approval	
Further information	Document editors

Versionenkontrolle

Version	**Edited by**	**Date**	**Activity**
0.2.0	J. Grazioli	2017-08-23	Creation begun
0.3.0	J. Grazioli	2017-10-18	check products section

Contents

1 Introduction

1.1 Motivation for Separating Code and Configuration

- A future "final" version of the code should not be touched.
- Clear change and versioning control of the code.
- No local file paths and personal settings in the code.
- Developing and test can be done using local config files.
- Parameters are not hard coded and can be changed easily.
- High flexibility. All settings are made by changing only 1–3 config files.

1.2 Rationale and other sources of information

The processing is based on the PyRad framework. Usually pyrad is cloned in the home directories of each users in the appropriate servers. As an example, for zueub222 and user jgr:

```
jgr@zueub222:pyrad$ pwd
/home/lom/users/jgr/pyrad
```

The purpose of this document is to allow users to process data by manipulating only configuration files. The focus is therefore here on the processing output and not on PyRad itself. For an overview of the functionalities of PyRad, its installation, and development, please refer to the documentation available at:

```
pyrad/doc/
```

2 Data Processing

2.1 Data Processing

The information about this section is complementary to Sec. 2 of the document:

```
pyrad/doc/pyrad_user_manual.docx
```

The data processing can be started from the linux shell, after activation of the proper conda environment. Depending on the server, the appropriate environment may be "root" (i.e. for zueub222):

```
jgr@zueub222:~$ source activate root
(root) jgr@zueub222:~$
```

or it can be "pyrad" (i.e. for CSCS and cirrus servers). The python scripts used to process the radar data can be called from the directory:

```
pyrad/src/pyrad_proc/scripts/
```

The scripts that are useful for this document are the following processing and realtime scripts:

- main_process_data.py
- main_process_data_rt.py
- main_process_data_period.py
- process_trajectory.py

they can all be called from the linux shell.

2.1.1 Process data real time (main_process_data_rt.py)

This script is designed to process data in real time. The main feature that makes him real-time is the capability of "listening" some data folders and immediately process new data se they appear. Verbatim, from the help page of the script:

```
This program performs real time processing of the data

To run the processing framework type:
    python main_process_data.py [config_files]
     --starttime [process_start_time] --endtime [process_end_time]
     --cfgpath [cfgpath] --proc_period [proc_period]

If startime or endtime are specified the program will start processing at the
specified time and end at the specified time. Otherwise the program ends when
the user interrupts it.
cfgpath is an optional argument with default: '$HOME/pyrad/config/processing/'
proc_period is the time that has to pass before attempting to restart the
processing in s
if proc_finish is not none it indicates the time the program is allowed to ran
berfore forcing it to end


Example:
```

```
    python main_process_data.py 'paradiso_fvj_vol.txt' 'paradiso_fvj_rhi.txt'
    --starttime '20140523000000' --endtime '20140523001000'
    --cfgpath '$HOME/pyrad/config/processing/' --proc_period 60 --proc_finish 120
```

```
usage: main_process_data_rt.py [-h] [--starttime STARTTIME]
                               [--endtime ENDTIME] [--cfgpath CFGPATH]
                               [--proc_period PROC_PERIOD]
                               [--proc_finish PROC_FINISH]
                               cfgfiles [cfgfiles ...]

Entry to Pyrad processing framework

positional arguments:
  cfgfiles              name of main configuration file

optional arguments:
  -h, --help            show this help message and exit
  --starttime STARTTIME
                        starting time of the data to be processed. Format
                        YYYYMMDDhhmmss
  --endtime ENDTIME     end time of the data to be processed. Format
                        YYYYMMDDhhmmss
  --cfgpath CFGPATH     configuration file path
  --proc_period PROC_PERIOD
                        Period between processing rounds (s)
  --proc_finish PROC_FINISH
                        Processing time allowed before shutdown (s)
```

TODO: explain better the last two parameters -jgr- -fvj-

2.1.2 Process data (main_process_data.py)

Stadard data processing (i.e., usually non real time) is performed by this script. Verbatim from the help page:

```
This program processes and post-processes data over a time span

To run the processing framework type:
    python main_process_data.py [config_file] --starttime [process_start_time] --endtime
     [process_end_time] --postproc_cfgfile [postproc_config_file] --cfgpath [cfgpath]

If startime and endtime are not specified the program determines them from
the trajectory file or the last processed volume.
postproc_cfgfile is an optional argument with default: None
cfgpath is an optional argument with default: '$HOME/pyrad/config/processing/'

Example:
    python main_process_data.py 'paradiso_fvj_vol.txt' --starttime '20140523000000'
    --endtime '20140523001000' --postproc_cfgfile 'paradiso_fvj_vol_postproc.txt' --cfgpath
     '$HOME/pyrad/config/processing/'
```

```
usage: main_process_data.py [-h] [--starttime STARTTIME] [--endtime ENDTIME]
                            [--postproc_cfgfile POSTPROC_CFGFILE]
                            [--cfgpath CFGPATH] [-i INFOSTR] [-t TRAJFILE]
                            proc_cfgfile

Entry to Pyrad processing framework

positional arguments:
  proc_cfgfile          name of main configuration file

optional arguments:
  -h, --help            show this help message and exit
  --starttime STARTTIME
                        starting time of the data to be processed. Format
                        YYYYMMDDhhmmss
  --endtime ENDTIME     end time of the data to be processed. Format
                        YYYYMMDDhhmmss
  --postproc_cfgfile POSTPROC_CFGFILE
                        name of main post-processing configuration file
  --cfgpath CFGPATH     configuration file path
  -i INFOSTR, --infostr INFOSTR
                        Information string about the actual data processing
                        (e.g. 'RUN57'). This string is added to the filenames
                        of the product files.
  -t TRAJFILE, --trajfile TRAJFILE
                        Definition file of plane trajectory. Configuration of
                        scan sector, products, ...
```

2.1.3 Process data (main_process_data_period.py)

This script is used in post-processing to process data over long periods of time, usually several days. According to its help entry:

```
This program does the daily processing and post-processing over a period of time.

To run the processing framework type:
    python main_process_data.py [config_file] [process_start_date] [process_end_date]
    --starttime [process_start_time] --endtime [process_end_time] --postproc_cfgfile
     [postproc_config_file] --cfgpath [cfgpath]

starttime is an optional argument with default: '000000'
endtime is an optional argument with default: '235959'
postproc_cfgfile is an optional argument with default: None
cfgpath is an optional argument with default: '$HOME/pyrad/config/processing/'

Example:
    python main_process_data.py 'paradiso_fvj_vol.txt' '20140523' '20140525'
    --starttime '000000' --endtime '001000' --postproc_cfgfile 'mals_emm_vol_postproc.txt'
    --cfgpath '$HOME/pyrad/config/processing/'
```

```
usage: main_process_data_period.py [-h] [--starttime STARTTIME]
                                   [--endtime ENDTIME]
                                   [--postproc_cfgfile POSTPROC_CFGFILE]
                                   [--cfgpath CFGPATH]
                                   proc_cfgfile startdate enddate

Entry to Pyrad processing framework

positional arguments:
  proc_cfgfile          name of main configuration file
  startdate             starting date of the data to be processed. Format
                        YYYYMMDD
  enddate               end date of the data to be processed. Format YYYYMMDD

optional arguments:
  -h, --help            show this help message and exit
  --starttime STARTTIME
                        starting date of the data to be processed. Format
                        hhmmss
  --endtime ENDTIME     end date of the data to be processed. Format hhmmss
  --postproc_cfgfile POSTPROC_CFGFILE
                        name of main post-processing configuration file
  --cfgpath CFGPATH     configuration file path
```

TODO: explain better why this script is needed, conceptually. It seems very similar to the main_process_data one

2.1.4 Process trajectories (process_trajectory.py)

This script is used in post-processing to process trajectory data. IT is therefore often used in the days/weeks following a flight test or a RTS test. The usage of this script is:

```
usage: process_trajectory.py [-h] [-c CFGFILE]
                             [--preproc_cfgfile PREPROC_CFGFILE] [-i INFOSTR]
                             trajfile [starttime] [endtime]

Create PYRAD products using a plane trajectory

positional arguments:
  trajfile              Definition file of plane trajectory. Configuration of
                        scan sector, products, ...
  starttime             Starting time of the data to be processed. Format:
                        YYYYMMDDhhmm[ss]. If not given, the time of the first
                        sample is used.
  endtime               End time of the data to be processed. Format:
                        YYYYMMDDhhmm[ss]. If not given, the time of the last
                        sample is used.

optional arguments:
  -h, --help            show this help message and exit
```

```
  -c CFGFILE, --cfgfile CFGFILE
                        Main configuration file. Defines the
  --preproc_cfgfile PREPROC_CFGFILE
                        name of main pre-processing configuration file
  -i INFOSTR, --infostr INFOSTR
                        Information string about the actual data processing
                        (e.g. 'RUN57'). This string is added to the filenames
                        of the product files.

Example:
  process_trajectory.py -c $HOME/pyrad/config/processing/mals_emm_rw22_traj.txt
     --preproc_cfgfile $HOME/pyrad/config/processing/mals_emm_rw22_traj_preproc.txt
     -i TS011 /data/mals_plane_traj/EMM/gnv_20161026_ts011_seat_emmen_flt01_ADS.txt
```

3 Configuration

The configuration of the data processing is divided into three files. The *main configuration file* (see Section 3.1), the *location configuration file* (see Section 3.2) describing the location of the weather radar and the used scans. The *product configuration file* describes the datasets and products. As this is bit more complicated it is described in its own Section 4.

The configuration files are located in *malsgit/config_pyrad/processing/*.

3.1 Main Configuration File

Tha main configuration file is used to define the global settings, notably the paths to the different sources of data. The parameters of the main configuration file are described in Table 2

Table 2: Configuration parameters of the main configuration file

Name	Type	Description
name	STRING	Name of the data processing. This name is used in the path of the saved products in the following manner: <saveimgbasepath>/<name>/<YYYY-MM-DD>/<datasetname>/<prodname>/ <outputname>
datapath	STRING	Base directory of the rainbow raw data. This field must have a trailing '/'. The raw data files of a scan can be found using the following file path: <datapath>/<scanname>/<YYYY-MM-DD>/<YYYYMMDDHHMMSS00datatype>.<ext>
psrpath	STRING	Base directory of the power spectrum (psr) raw files. By default: empty string
iqpath	STRING	Base directory of the IQ data files. By default: empty string
configpath	STRING	Base directory of the configuration files. This directory contains clutter maps, filter coefficients, antenna pattern, and the data processing configuration files.
cosmopath	STRING	Base directory of the COSMO data files.
dempath	STRING	Base directory of the Digital Elevation Model (DEM) files. Basically to load the radar visibility (Optional)
smnpath	STRING	Base directory of the SwissMetNet stations data. Used in the comparison between radar data and rain gauges (Optional)
disdropath	STRING	Base directory of the disdrometer data. Used in the comparison between radar data and disdrometers (Optional)
solarfluxpath	STRING	Base directory of the solar flux data. Used to plot the calibration bias based on sun monitoring (Optional)
rad4alppath	STRING	Base directory of the averaged reflectivity of the rad4alp radars. Used in the inter-comparison between radars (Optional)
rad4alprawpath	STRING	Base directory of raw rad4alp radars data. Used in the inter-comparison between radars (Optional)
statuspgr	STRING	path to program that reads rad4alp status files. Used in the calculation of the radar noise (Optional)
mapspath	STRING	path to directory where background maps are stored. Used in the plotting of PPI and CAPPI data (Optional)

Continued on next page

Table 2 – *Continued from previous page*

Name	Type	Description
mxpolname	STRING	Name of the directory where MXPOL processed data is stored. Used in the inter-comparison between radars (Optional)
mxpolrawname	STRING	Name of the directory where MXPOL raw data is stored. Used in the inter-comparison between radars (Optional)
locationConfigFile	STRING	File name (with full path) of the location configuration file. Described in Section 3.2.
productConfigFile	STRING	File name (with full path) of the product configuration file. Described in Section 4.
lastStateFile	STRING	File name (with full path) of the file containing the time of the last processed scan. Used in particular for real time processing.
saveimg	INT	Set to 1, if all generated images shall be written to a file. If set to 0, the graphical output is displayed on screen and not saved.
imgformat	STRING	File format of the images. The following formats are supported: eps, png and jpg (**TODO: check if applicable -jgr- -fvj-**). If *saveimg* is set to 0, this field is not used.
saveimgbasepath	STRING	Base directory for the images to save. The directory structure looks as follows: <saveimgbasepath>/<name>/<YYYY-MM-DD>/<datasetname>/<prodname>/ <outputname> If *saveimg* is set to 0, this field is not used.
loadbasepath	STRING	OPTIONAL. Base path of saved data. By default, this field is set to *saveimgbasepath*.
loadname	STRING	OPTIONAL. Name of the saved data processing. Used for saved volume loading. By default, this field is set to *name*.

TODO: discuss together -jgr- -fvj- to see which entry should i add or remove

3.2 Location Configuration File

The location configuration files describes some parameters that are depending on the specific location of a radar (type of scans we want to measure, radar name, etc). The location of the weather radar (its position) itself, is instead usually read from the radar metadata directly and it is not necessarily defined in this file. The fields are described in Table 3.

Table 3: Configuration parameters of the location configuration file

Name	Type	Description
RadarName	STRING	Short version name of a rad4alp radar (i.e. A, D, L, P) or DX50, MXPol for the X-band radars.
RadarRes	STRING	rad4alp radar resolution (H or L). Only necessary if rad4alp data is processed
RadarBeamwidth	FLOAT	Radar antenna beam width [Deg]
AntennaGaindB	FLOAT	antenna gain [dB]

Continued on next page

Table 3 – *Continued from previous page*

Name	Type	Description
ScanList	STRARR	A list with the scans used for this data processing. Note that the first scan in this list is used as master scan. The master scan must be the first (temporal) scan of the corresponding rainbow task. In case of composite volumes the master scan is usually a PPI and the following are RHIs. If the radar processed is rad4alp the scan list consists of the radar elevation (i.e. from 001 to 020). All scan names must have a trailing '/' except if rad4alp data is processed
ScanPeriod	FLOAT	Repetition period of each scan in minutes.
Azimtol	FLOAT	Tolerance in azimuth for irregular data. (0.5 is a good value)
clutterMap	STRING	Clutter map of the data processing. The clutter map is located at <configpath>/clutter/<clutterMap>
iso0setup	STRUCT	Setup to read the iso zero height from cosmo model.
xcoord	STRING	CH1903 X-coordinate of the radar position [km].
ycoord	STRING	CH1903 Y-coordinate of the radar position [km].
CosmoRunFreq	INT	Frequency of a COSMO model run in hours.
CosmoForecasted	INT	Hours forecasted by the COSMO model.
RcovFile	STRING	Name (without path) of the R-coefficients for the Kalman filtering. This file and path must be <configpath>/filter/<RcovFile>
PcovFile	STRING	Name (without path) of the P-coefficients for the Kalman filtering. This file and path must be <configpath>/filter/<PcovFile>
IQWindow	STRING	Possible values: HAMMING, HANNING, BLACKMAN, NONE. Default NONE. When processing IQ data, window type applied before computing the spectrum.
Clipping	FLOAT	Default 0. In spectral processing: dB above noise level below which the spectra is clipped. If 0 no clipping is applied
SWindow	INT	Default 0. In spectral processing: Size (in Doppler cells) of the Gaussian smoothing window used to smooth the Doppler spectrum. If 0 no smoothing is applied.
Average	INT	Boolean: 0,1. Default 0. Flag indicating whether the Doppler of a volume have to be averaged or not. A single PRF is assumed to perform the average.
ppiImageConfig	STRUCT	Structure defining the PPI image generating. The following 6 fields are described below:
rhiImageConfig	STRUCT	Structure defining the RHI image generating. The following 6 fields are described below:
xsize	INT	Number of horizontal pixels of the picture (without frame).
ysize	INT	Number of vertical pixels of the picture (without frame).

Continued on next page

Table 3 – *Continued from previous page*

Name	Type	Description
xmin	FLOAT	Distance of the left image boundary to the radar in km.
xmax	FLOAT	Distance of the right image boundary to the radar in km.
ymin	FLOAT	Distance of the lower image boundary to the radar in km.
ymax	FLOAT	Distance of the upper image boundary to the radar in km.
ppiMapImageConfig	STRUCT	Structure defining the PPI image overlayed on a map. The following 9 fields are described below:
rngRing	FLOAT	Distance between range rings (0 means no range ring) [km].
xsize	FLOAT	Image size (inches) [ich].
ysize	FLOAT	Image size [ich].
lonmin	FLOAT	Minimum WGS84 longitude [°].
lonmax	FLOAT	Maximum WGS84 longitude [°].
latmin	FLOAT	Minimum WGS84 latitude [°].
latmax	FLOAT	Maximum WGS84 latitude [°].
mapres	STRING	Map resolution. Accepted strings are: "10m", "50m", "110m"
maps	STRARR	String array of possible maps to overplot. Accepted entries include: relief, countries, provinces, urban_areas, roads, railroads, coastline, lakes, lakes_europe, rivers, rivers_europe
rvsazImageConfig	STRUCT	Structure defining the range versus azimuth image. The following 4 fields are described below:
xmin	FLOAT	Min angle on horizontal axis [Deg].
xmax	FLOAT	Max angle on horizontal axis [Deg].
ymin	FLOAT	Min range on vertical axis [km].
ymax	FLOAT	Max range on vertical axis [km].
rvselImageConfig	STRUCT	Structure defining the range versus elevation image. It contains the same 4 fields as rvsazImageConfig.
sunhitsImageConfig	STRUCT	Structure defining the sun hits image. The following 6 fields are described below:
xsize	INT	Number of horizontal pixels of the picture (without frame).
ysize	INT	Number of vertical pixels of the picture (without frame).
xmin	FLOAT	Minimum azimuth angle difference (between sun and radar).
xmax	FLOAT	Maximum azimuth angle difference (between sun and radar).
ymin	FLOAT	Minimum elevation angle difference (between sun and radar).
ymax	FLOAT	Maximum azimuth angle difference (between sun and radar).
wgs84ImageConfig	STRUCT	Structure defining the image in WGS84 coordinates. The following 6 fields are described below:
north	FLOAT	North limit of image [deg].

Continued on next page

Table 3 – *Continued from previous page*

Name	Type	Description
south	FLOAT	South limit of image [deg].
east	FLOAT	East limit of image [deg].
west	FLOAT	West limit of image [deg].
dlat	FLOAT	latitude resolution [deg].
dlon	FLOAT	longitude resolution [deg].
ch1903ImageConfig	STRUCT	Structure defining the image in Swiss coordinates. The following 6 fields are described below:
north	FLOAT	North limit of image [m].
south	FLOAT	South limit of image [m].
east	FLOAT	East limit of image [m].
west	FLOAT	West limit of image [m].
dchx	FLOAT	chx resolution [m].
dchy	FLOAT	chy resolution [m].
par_azimuth_antenna	STRUCT	Structure defining the PAR azimuth antenna. It contains the following 3 fields:
par_elevation_antenna	STRUCT	Structure defining the PAR elevation antenna. It contains the following 3 fields:
azPatternFile	STRING	Name of the azimuth pattern file of the antenna. This file and path must be <config-path>/antenna/<azPatternFile>
elPatternFile	STRING	Name of the elevation pattern file of the antenna. This file and path must be <config-path>/antenna/<elPatternFile>
fixed_angle	FLOAT	Fixed angle of a PAR antenna in degrees. For the PAR azimuth antenna this is the elevation angle. For the elevation antenna is is the azimuth angle.
RAINBOW-OLDMERGE	INT	OPTIONAL. If set, uses the old method to read the elevation angles from a raw rainbow file. This method just uses the elevation start angle. The new version take the mean between start and end elevation angle.
RAINBOW-SECTORMERGE	INT	OPTIONAL. Special method to merge the PPI's of a sector volume scan. Because the start and end azimuth angles of different PPI slices of a non-360 degrees volume scan do not correspond very well.

TODO: help needed here to define the complete list for pyrad and see what to remove -jgr -fvj-

4 Product Generation Configuration

This section describes the product configuration.

4.1 Basic Concept

The concept is based on three stages: 1. input or raw data volume, 2. datasets and 3. products.

The center point of the data processing is the dataset. A dataset can be generated from one or more than one input data volumes (e.g. a rainrate dataset uses 5 different raw data volume to be generated).

There are several different formats of a dataset. For example, a dataset can be a volume, a trajectory, a volume composite or a time series. Section 5 summarizes the possible datasets.

From a dataset the products are generated. What products can be generated from a dataset depends on the type (volume, volume composite, trajectory or time series) of the dataset.

Figure 1 shows a schema of an example of the basic concept of the product generation. From different input data a dataset is generated and from this dataset m products are created.

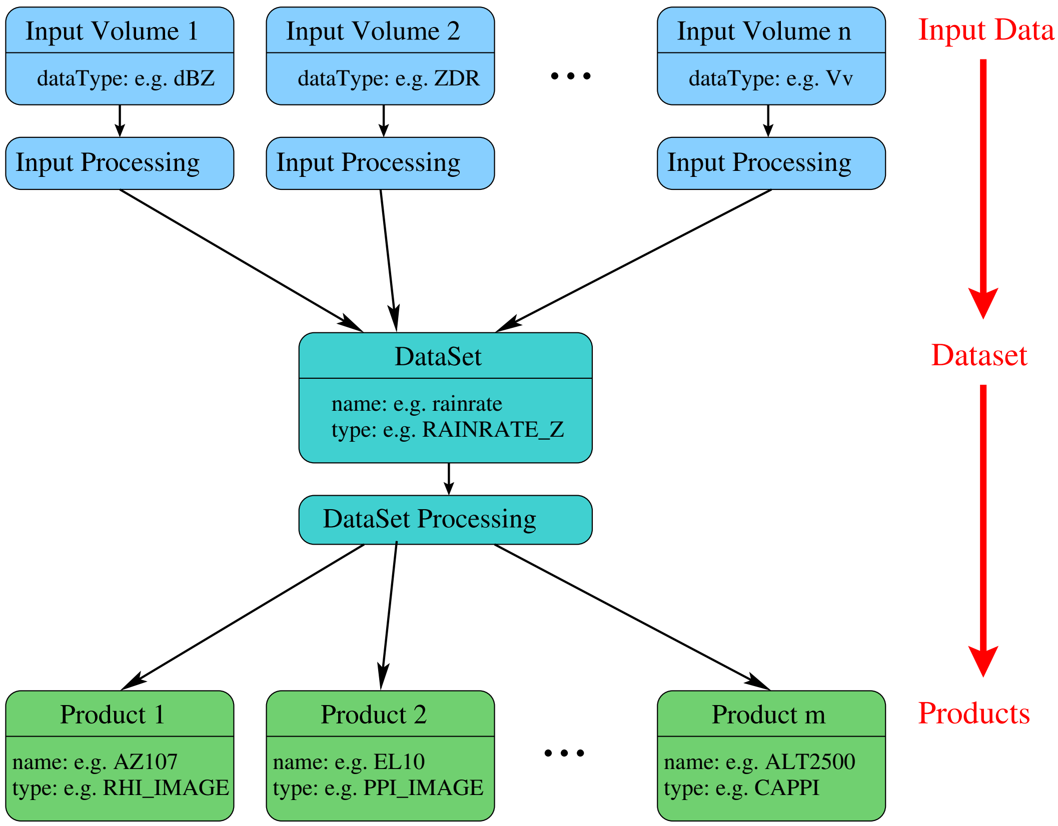

Figure 1: Basic concept for product generation from input (also called raw) data.

The dataset types are described in Section 5 and the product types in Section 6

4.2 The concept of processing level

TODO: add explanation of concept of processing level -jgr- -fvj-

4.3 Input Volume Datatypes

There are different group of input volume datatypes that can be read as input data for a dataset. These groups are summarized in Table 4. The possible volume datatypes for each datatype group are listed in Table 5.

The specification of the datatype for a dataset is done by first writing the datatype group, followed by a ':' and then the name of the datatype. If no ':' is given, it is assumed that the datatype belongs to the group *RAW*. For a datatype from the group *SAVED*, the dataset and product name must be specified. This information is separated by ','. Note that *NETCDF* datatypes cannot be mixed with other datatypes.

Caveat: please do not mistake the input volume datatype with the dataset type (entry "type" of the dataset structure)

```
<datasetname> STRUCT 2
   type      STRING <datasettype>
   datatype STRARR
      RAW:dBZ
      PSR:NhDBM
      COSMO:TEMP
      SAVED:RR_h,<olddataset>,<oldproduct>
```

Group	Description
RAW	Raw data generated by the DX50. Stored in rainbow data format.
PSR	Power spectra data generated by the signal processor of the DX50. Stored in Gematronik file format.
IQ	data based on IQ raw data. For the moment only from rad4alp radars. The data is stored in a binary format in a .dat file.
SAVED	A dataset generated by this dataprocessing procedure. As additional information the dataset and the product name of the previous dataset must be specified. Stored in IDL file format.
COSMO	Data created by COSMO. Converted to polar radar coordinates and stored in rainbow file format.
DEM	Digital Elevation Model data (DEM). Basically visibility in rainbow file format
NETCDF	External data in netcdf format.
NETCDFCOSMO	COSMO data in netcdf format.
RAD4ALP	rad4alp data.
RAD4ALPDEM	rad4alp visibility data.
RAD4ALPCOSMO	a binary file with COSMO data for rad4alp processing.
PROC	indicates that the dataset is the result of the preprocessing of raw data. (i.e. it will be created on the fly).
PSRPROC	indicates that the dataset is the result of the preprocessing of psr data. (i.e. it will be created on the fly).
RAINBOW	rainbow data types. **TODO: check this -jgr-**

Table 4: List of input volume datatype groups

TODO: (1) check entries. -jgr- -fvj-. Isaw for example that in pyrad we could have "RADAR001:CFRADIAL:..." types. (2) check that all the entries make sense

Group	Datatypes
RAW	dBZ, dBZv, dBuZ, dBuZv, V, Vv, Vu, Vvu, W, Wv, Wu, Wvu, KDP, uKDP, uKDPu, PhiDP, uPhiDP, uPhiDPu, RhoHV, RhoHVu, uRhoHV, L, ZDR, ZDRu, SQI, SQIv, SQIu, SQIvu, SNRh, SNRv, CDR
PSR	NhADU, NvADU, NhDBM, NvDBM, PhADU, PvADU, PhDBM, PvDBM, PhDBZ, PvDBZ, ShADU_HZ, ShADU_MS, ShDBADU_HZ, ShDBADU_MS, ShDBM_HZ, ShDBM_MS, ShDBZ_HZ, ShDBZ_MS, SvADU_HZ, SvADU_MS, SvDBADU_HZ, SvDBADU_MS, SvDBM_HZ, SvDBM_MS, SvDBZ_HZ, SvDBZ_MS, UhADU_HZ, UvADU_HZ, UhADU_MS, UvADU_MS, UhCPX_HZ, UvCPX_HZ, . . .
IQ	Same as PSR plus WhADU, WvADU, WhDBADU, WvDBADU, WhDBM, WvDBM, WhDBZ, WvDBZ, IhCPX, IvCPX, IhRAW, IvRAW, IhADU, IvADU, IhDBADU, IvDBADU, IhDBM, IvDBM, IhDBZ, IvDBZ, IhDEG, IvDEG
SAVED	(Some examples, not complete) dV, dVv, dVu, dVvu, RR_Zh, RR_Ah, RR_Kdp, Att, SAN, TRAJ, HEIGHT, WP, WPDIFF, WPRELDIFF, dtfilter, RAINEXT, RCS
COSMO	ISO0, TEMP
DEM	VIS
NETCDF	
NETCDFCOSMO	ISO0, TEMP
RAD4ALP	dBZ, ZDR, RhoHV, uPhiDP, V, W, SNRh, SNRv, L, CDR.
RAD4ALPDEM	VIS.
RAD4ALPCOSMO	ISO0, TEMP
PROC	dBZc, dBZvc, ZDRc, PhiDPc, KDPc, RhoHVc, Ah, Adp.
PSRPROC	sZh_MS, sZv_MS, sZDR_MS, sRhoHV_MS, sPhiDP_MS.

Table 5: List of possible datatypes

TODO: check if those are up to date

4.4 Product Configuration File

The product configuration files describes the products that are generated for the data processing. The fields are described in Table 6.

Table 6: Configuration parameters of the product configuration file

Name	Type	Description
dataSetList	STRARR	A list of the datasets that are generated for the data processing. There must be a structure in the product configuration file defining the dataset and its product for each dataset in this list. The list of datasets may include the processing level. **TODO: add link to the definition of processing level**

Continued on next page

Table 6 – *Continued from previous page*

Name	Type	Description
<datasetname>	STRUCT	A structure defining a dataset. The <datasetname> must be a member of the *dataSetList* list. The structure defines the type of the dataset, its parameters and the products that are applied to this dataset. The <datasetname> can be freely chosen. Just make sure the spelling is the same in the dataSetList. The <datasetname> is used for the path to store the products: <saveimgbasepath>/<name>/<YYYY-MM-DD>/<datasetname>/<prodname>/ <outputname>. The mandatory fields of this structure are described below. The fields depending on the dataset type are described in Section 5.
type	STRING	Type of the dataset. The tables in Section 5 list all possible dataset types.
datatype	STRARR	Raw (or input) datatype. The dataset is generated using rainbow raw files of this datatypes or the dataset is generated using multiple raw files with different datatypes.
IGNORE_-MISSING_VOLS	INT	OPTIONAL. If set, the function processing the dataset is called if not all input data volumes could be selected. For example this could be used for the sanity check. In such a case only the checks are done with the available input volumes. By default, this option is off. **TODO: can be removed?**
DSSAVENAME	STRING	OPTIONAL. Usually the product files are stored under the name of the dataset. If this parameter is saved, the files are stored under this name instead of the dataset name.
INPUT-PROCESSING	STRUCT	OPTIONAL. Input processing of one or more input volumes. See section 4.5 for details. **TODO: can be removed?**
DATASET-PROCESSING	STRUCT	OPTIONAL. Dataset processing of a volume dataset. See section 4.5 for details. **TODO: can be removed?**
products	STRUCT	This structure contains a list of products. Each product is a structure named as <prodname>. For the product description see Section 6.
<prodname>	STRUCT	Structure defining a product of the dataset. The <prodname> can be freely chosen. The <prodname> is used to store the output products: <saveimgbasepath>/<name>/<YYYY-MM-DD>/<datasetname>/<prodname>/ <outputname>. The mandatory fields of this structure are described below. The fields depending on the product type are described in Section 6.
type	STRING	Type of the product. Section 6 describes all possible product types.

Example: A simple product config file is listed below:

```
#
# Product generation configuration
```

```
#

# List of datasets to generate.
# The detailed specification of each dataset is given below.
dataSetList STRARR 12
   l0:TEMP
   l0:reflectivity
   l0:ZDR
   l0:RhoHV
   l0:echoID
   l1:echoFilter
   l3:echoFilter_Ah
   l2:outlierFilter
   l2:Att_ZPhi
   l3:hydroclass
   l4:rainrate
   l3:wind


# ==========================================================================
#               COSMO data
# ==========================================================================
TEMP STRUCT 6
    type STRING COSMO_LOOKUP
    datatype STRARR 1
        dBZ
    cosmo_type STRING TEMP
    regular_grid INT 0
    lookup_table INT 1
    MAKE_GLOBAL INT 1


# ==========================================================================
#                 raw data processing
# ==========================================================================
reflectivity STRUCT 3
   type     STRING RAW
   datatype STRING dBZ
   products STRUCT 4
      EL03_0 STRUCT 3
         type  STRING PPI_IMAGE
         anglenr INT 0
         voltype STRING dBZ
      EL04_0 STRUCT 3
         type  STRING PPI_IMAGE
         anglenr INT 1
         voltype STRING dBZ
      EL05_7 STRUCT 3
         type  STRING PPI_IMAGE
         anglenr INT 2
         voltype STRING dBZ
```

```
      SAVESTATE STRUCT 2
         type STRING SAVESTATE
         voltype STRING dBZ

ZDR STRUCT 3
   type     STRING RAW
   datatype STRING ZDR
   products STRUCT 3
      EL03_0 STRUCT 3
         type  STRING PPI_IMAGE
         anglenr INT 0
         voltype STRING ZDR
      EL04_0 STRUCT 3
         type  STRING PPI_IMAGE
         anglenr INT 1
         voltype STRING ZDR
      EL05_7 STRUCT 3
         type  STRING PPI_IMAGE
         anglenr INT 2
         voltype STRING ZDR

RhoHV STRUCT 3
   type     STRING RAW
   datatype STRING RhoHV
   products STRUCT 3
      EL03_0 STRUCT 3
         type  STRING PPI_IMAGE
         anglenr INT 0
         voltype STRING RhoHV
      EL04_0 STRUCT 3
         type  STRING PPI_IMAGE
         anglenr INT 1
         voltype STRING RhoHV
      EL05_7 STRUCT 3
         type  STRING PPI_IMAGE
         anglenr INT 2
         voltype STRING RhoHV


# ==============================================================================
#                  echo identification
# ==============================================================================
echoID STRUCT 3
    type STRING SAN
    datatype STRARR 4
        dBZ
        ZDR
        uPhiDP
        RhoHV
    MAKE_GLOBAL INT 1
```

```
# ==========================================================================
#                 clutter and noise suppression
# ==========================================================================
# echo type 3 : precip, 2 : clutter, 1 : noise
echoFilter STRUCT 4
    type STRING ECHO_FILTER
    datatype STRARR 8
        PROC:echoID
        dBZ
        ZDR
        RhoHV
        PhiDP
        KDP
V
W
    echo_type INT 3
    MAKE_GLOBAL INT 1

echoFilter_Ah STRUCT 4
    type STRING ECHO_FILTER
    datatype STRARR 2
        PROC:echoID
        PROC:Ah
    echo_type INT 3
    MAKE_GLOBAL INT 1


# ==========================================================================
#                 outlier filter
# ==========================================================================
outlierFilter STRUCT 8
    type STRING OUTLIER_FILTER
    datatype STRARR 1
        PROC:Vc
    threshold FLOAT 10.
    nb INT 2
    nb_min INT 3
    percentile_min FLOAT 5.
    percentile_max float 95.
    MAKE_GLOBAL INT 1


# ==========================================================================
#                 Attenuation
# ==========================================================================
Att_ZPhi STRUCT 5
    type STRING ATTENUATION
    datatype STRARR 4
        PROC:dBZc
        PROC:ZDRc
```

```
        PROC:PhiDPc
        PROC:TEMP
    ATT_METHOD STRING ZPhi
    fzl FLOAT 2000.
    MAKE_GLOBAL INT 1


# ============================================================================
#                 hydrometeor classification products
# ============================================================================
hydroclass STRUCT 5
    type STRING HYDROCLASS
    datatype STRARR 5
        PROC:dBZc
        PROC:ZDRc
        PROC:RhoHVc
        PROC:KDPc
        PROC:TEMP
    HYDRO_METHOD STRING SEMISUPERVISED
    RADARCENTROIDS STRING DX50
    MAKE_GLOBAL INT 1


# ============================================================================
#               rainfall rate
# ============================================================================
rainrate STRUCT 5
    type STRING RAINRATE
    datatype STRARR 3
        PROC:dBZc
        PROC:Ahc
        PROC:hydro
    RR_METHOD STRING hydro
    MAKE_GLOBAL INT 1
    products STRUCT 3
      EL03_0 STRUCT 3
         type  STRING PPI_IMAGE
         anglenr INT 0
         voltype STRING RR
      EL04_0 STRUCT 3
         type  STRING PPI_IMAGE
         anglenr INT 1
         voltype STRING RR
      EL05_7 STRUCT 3
         type  STRING PPI_IMAGE
         anglenr INT 2
         voltype STRING RR


# ============================================================================
#               wind velocity
```

```
# ==========================================================================
wind STRUCT 5
    type STRING WIND_VEL
    datatype STRARR 1
        PROC:Vc
    vert_proj INT 0
    MAKE_GLOBAL INT 1
    products STRUCT 3
      EL03_0 STRUCT 3
         type  STRING PPI_IMAGE
         anglenr INT 0
         voltype STRING wind_vel_h_az
      EL04_0 STRUCT 3
         type  STRING PPI_IMAGE
         anglenr INT 1
         voltype STRING wind_vel_h_az
      EL05_7 STRUCT 3
         type  STRING PPI_IMAGE
         anglenr INT 2
         voltype STRING wind_vel_h_az
```

The example product configuration files defines twelve datasets: let us take the example of *l0:reflectivity* and *l4: rainrate*.

The *reflectivity* dataset is a *RAW* dataset generated using the raw datatype *dBZ*. Four products are generated for this dataset: the products *EL03_ 0*, *EL04_ 0*, *EL05_ 7* which are of type *PPI_ IMAGE* (the first, second and third PPIs in a volume scan, as given in field *anglenr*). The images are stored in <saveimgbasepath>/<name>/<YYYY-MM-DD>/reflectivity/EL0X_X/. The fourth product saves the volume in the path given by *loadbasepath* fo the main configuration file. **TODO: check that**

The *rainrate* dataset is a *RAINRATE* dataset generated using the processed datatypes *dBZc*, *Ahc*, *hydro*, by means of the *HYDRO* retrieval method. In analogous way with respect to the *reflectivity* dataset, PPI images are generated as products.

4.5 Input- and DataSet-Processing

TODO: -jgr- -fvj- if i understand correctly, this is outdated. Need to see what is the equivalent for pyrad.

5 Datasets

Each dataset has a format. The possible formats are listed in Table 7. The dataset *type* is then a subset of these possible formats. A list with all possible dataset types for each dataset format is given in the following tables. If a dataset type starts with *TRAJ*, it uses a plane trajectory to generate the dataset.

TODO: check in pyrad_proc/proc/process_aux.py to see which datasets exist. Does the concept of dataset type still exist?

Table 7: List of all possible formats of a dataset

Name	**Description**
VOL	A structure with a volume with 3-dimensional data (azimuth, elevation and range).
RAY	As a volume structure without range dimension. The data are two-dimensional as function of (azimuth, elevation). **TODO: still there? or to be done?**
TIMESERIES	A structure with an array with 2 dimensions: time and one or more values (e.g. mean, min, max, quantiles, ...)
TIMEARRAY	A structure with a two-dimensional array: time and another parameter (e.g. altitude)
TIMEARRAY_ARRAY	A structure with a three-dimensional array: type, time and another parameter (usually altitude) **TODO: Still there?**
TRAJ_ONLY	Trajectory data only. A structure containing a timeseries structure with range, azimuth and elevation position of the plane. **TODO: still there? or to be done?**
TRAJ_VOL_COMPOSITE	A volume structure (azimuth, elevation and range). It is a composite of several raw volume projected to the plane trajectory. **TODO: still there? or to be done?**
VOLUMEGROUP	XXX to be described **TODO: still there? or to be done?**
PSR	A structure with a volume with 4-dimensional data. For each (azimuth, elevation and range) tuple a vector with values for each spectrum sample is given. **TODO: still there? or to be done?**
ARRAY_PSR	A structure with a volume with 5-dimensional data. For each (data type, azimuth, elevation and range) tuple a vector with values for each spectrum sample is given. It is a concatenation of spectra generated by a processing routine. **TODO: still there? or to be done?**
MONITORING	A structure with the data resultant from the data quality monitoring. For each parameter monitored there is a space-time matrix containing the results of the monitoring for each volume and several statistical parameters.
ARRAY_VOL	A structure with a volume with a 4-dimensional data (type,azimuth,elevation,range). It is a concatenation of volumes generated together by a processing routine.

Continued on next page

Table 7 – *Continued from previous page*

Name	Description
SUN_MONITORING	This dataset is computed offline. It generates two different structures. When processing data, whenever there is a possible sun hit it generates a volume array with sun hit information (channel power, zdr, position of sun hit, ...). At the end of the data processing generates a structure with the parameter datatype_array=SUN_HITS, the sun hits file name, the sun retrieval file name and the parameters of the fitting for horizontal sun power, vertical sun power and sun differential reflectivity. **TODO: is it still there or integrated somewhere else?**
XVSC	Used for the intercomparison of averaged reflectivity. A structure that contains a list of the radars with which the processed radar is compared and the volume time. **TODO: is it still there?**
INTERCOMP	Used for the inter-comparison of any radar variable. A structure that contains a list of the radars with which the processed radar is compared, the volume time and the data type. **TODO: see comments below**
HISTOGRAM	A structure with 2D and 1D histograms of radar variables. **TODO: see comments below**
COMPOSITE_VOLUME	A nested structure that can embed two sub-structures. Each sub-structure is a volume with 3D data (see VOL) or 4D data (see ARRAY_VOL). Usually one sub-structure includes PPI scans and the second one RHI scans. **TODO: still of interest?**

Name	Description	Reference
SAN	Sanity check dataset. This dataset is used to preprocess other raw data for other datasets.	Section 5.1.1
RAW	Use a rainbow raw volume directly as dataset.	Section 5.1.2
WINDSHEAR	Calculate the windshear.	Section 5.1.3
RAINRATE	Rainrate retrieval with a selected method.	Section 5.1.4
BACKSCATTER	Calculate the radar backscattering cross setion from the reflectivity. **TODO: still there? or to be done?**	Section 5.1.5
SPEC_QUALITY	Compare V and W estimations with the Doppler spectrum **TODO: still there? or to be done?**	Section 5.1.6
SPHERECALIBRATION	Processing of the sphere calibration measurement **TODO: still there? or to be done?**	Section 5.1.7
IQ_RHOHV	Computes RhoHV from raw IQ data. **TODO: still there? or to be done?**	Section 5.1.8
IQ_ZDR	Computes Zdr from raw IQ data. **TODO: still there? or to be done?**	Section 5.1.9

Table 8: List of all possible dataset types for VOL datasets

Name	Description	Reference
RAINEXTENSION	Calculate the rain extension. **TODO: still there? or to be done?**	Section 5.2.1

Table 9: List of all possible dataset types for RAY datasets

Name	Description	Reference
TRAJ_ATPLANE	A trajectory is combined with saved rainbow volumes.	Section 5.3.1
TRAJ_ANTENNA_PATTERN	A trajectory is combined with saved rainbow volumes.	Section 5.3.2
POINT_MEASUREMENT	The radar measurement over one specific point	Section 5.3.3

Table 10: List of all possible dataset types for TIMESERIES datasets

Name	Description	Reference
TIME_HEIGHT	Make a time-height profile. **TODO: still there? or to be done?**	Section 5.4.1
TIME_VECTOR	Make a time-vector dataset. **TODO: still there? or to be done?**	Section 5.4.2
QVP	Make a Quasi Vertical Profile (QVP) **TODO: still there? or to be done?**	Section 5.4.3

Table 11: List of all possible dataset types for TIMEARRAY datasets

Name	Description	Reference
TIME_SUMMARY_HEIGHT	Make a set of time-height profiles. Each profile is a statistical moment (over height levels) of an RHI image. **TODO: still there? or to be done?**	Section 5.5.1

Table 12: List of all possible dataset types for TIMEARRAY_ARRAY

Name	Description	Reference
TRAJ	Uses a trajectory without radar data. Note, just a few product types can be applied on this dataset.	Section 5.6.1
TRAJ_PLANEHITS	Uses a trajectory to find plane hits in radar data.	Section 5.6.2

Table 13: List of all possible dataset types for TRAJ_ONLY datasets **TODO: still there? or to be done?**

Name	Description	Reference
TRAJ_VOL	A trajectory is combined with raw rainbow volumes.	Section 5.7.1

Table 14: List of all possible dataset types for TRAJ_VOL_COMPOSITE datasets **TODO: still there? or to be done?**

Name	Description	Reference
QUANTILE	Quantile calculation. **TODO: still there? or to be done?**	Section 5.8.1

Table 15: List of all possible dataset types for VOLUMEGROUP datasets

Name	Description	Reference
PSR_RAW	The raw spectra values.	Section 5.9.1
PSR_MERGE	Merges every two spectra with different PRF to one.	Section 5.9.2
PSR_VERTICAL	XXX to be described	Section 5.9.3
PSR_SZH	Computes the spectral horizontal reflectivity	Section 5.9.4
PSR_SZV	Computes the spectral vertical reflectivity	Section 5.9.5
PSR_SZZDR	Computes the spectral differential reflectivity	Section 5.9.6
PSR_SZDR_TEXT	Computes the texture of the spectral differential reflectivity	Section 5.9.7
PSR_SRHOHV	Computes the spectral co-polar correlation coefficient	Section 5.9.8
PSR_SPHIDP	Computes the spectral differential phase	Section 5.9.9
PSR_SPHIDP_TEXT	Computes the texture of the spectral differential phase	Section 5.9.10
PSR_DSD	Computes the drop size distribution (DSD) from spectral parameters.	Section 5.9.11

Table 16: List of all possible dataset types for PSR datasets **TODO: not there yet, but to be done?**

Name	Description	Reference
PSR_FILTER	Computes spectral polarimetric variables and filters out clutter contaminated Doppler bins	Section 5.10.1
PREPROC_PSR	Computes spectral polarimetric variables and optionally filters out clutter contaminated Doppler bins. The clean polarimetric variables are then usable by other functions.	Section 5.10.2

Table 17: List of all possible dataset types for ARRAY_PSR datasets **TODO: still there? or to be done?**

Name	Description	Reference
MONITORING	The data obtained by the data quality monitoring process	Section 5.11.1

Table 18: List of all possible dataset types for MONITORING datasets **TODO: changed from MON to MONITORING. Ok?**

Name	Description	Reference
PREPROCESSING	Preprocessing of raw radar data. **TODO: still there? or to be removed?**	Section 5.12.1
MEANZ	Time-averaged reflectivity, differential phase (weighted by reflectivity) and a flag matrix containing information on clutter and attenuation within the time of averaging. **TODO: still there? or to be removed?**	Section 5.12.2
XVSC_POSITION	A boolean matrix with 1 indicating the range bins usable for intercomparison with each radar. **TODO: still there? or changed name?**	Section 5.12.3
ATTENUATION	Attenuation estimation and attenuation correction on the polarimetric variables.	Section 5.12.4
HYDROCLASS	Hydrometeor classification.	Section 5.12.5
PHIDP_KDP	K_{dp} and Φ_{dp} estimated from raw noisy Ψ_{dp}. **TODO: still there? or name is now splitted in several smaller ones?**	Section 5.12.6
PSR_POLVARIABLES	Computes the polarimetric variables from the spectra. **TODO: Not there yet, i guess**	Section ??
PSR_DSDBASED_PARAM	Computes parameters based on integrated DSD such as liquid water content (LWC) and rainfall rate. **TODO: Not there yet, i guess**	Section ??

Table 19: List of all possible dataset types for ARRAY_VOL datasets

Name	Description	Reference
SUN_MONITORING	sun monitoring data	Section 5.13.1

Table 20: List of all possible dataset types for sun monitoring datasets. **TODO: is it still there?**

Name	Description	Reference
XVSC	inter-comparison of time-averaged reflectivity between radars.	Section 5.14.1

Table 21: List of all possible dataset types for XVSC datasets **TODO: is it still there?**

Name	Description	Reference
INTERCOMP	inter-comparison of instantaneous values of polarimetric parameters between radars.	Section 5.15.1

Table 22: List of all possible dataset types for INTERCOMP datasets. **TODO: is it still there? or does it correspond to the current INTERCOMP_TIME_AVG ?**

Name	Description	Reference
HISTOGRAM	1D and 2D histograms of polarimetric data.	Section 5.16.1

Table 23: List of all possible dataset types for HISTOGRAM datasets. **TODO: still there as datasets?**

Name	Description	Reference
———	COMPOSITE_VOLUME are just combination of VOL datasets	Section 5.17

Table 24: List of all possible dataset types for COMPOSITE_VOLUME datasets. **TODO: still there?**

5.1 VOL Datasets

5.1.1 SAN

A sanity check dataset. From four raw input data volumes (dBZ, ZDR, uPhiDP and RhoHV) each radar cell is checked if its value is feasible. The output sanity volume contains a mask for each radar cell indicating if the cell is noise, clutter or melting layer. This dataset can be stored to a global variable and can be used to preprocess the raw input data of other datasets.

Note, that this dataset must be at the first place in the dataset list!

The parameters of a *SAN* dataset are listed in Table 25. The general parameters *type*, *datatype*, *ignore_missing_vols* and *products* are not listed here, see Table 6 for description of these parameters.

TODO: here i checked in process_echoclass.py and i am not sure the configuration inputs below still apply. Do they?

Parameter	Type	Description
NO_MELTINGLAYER	INT	OPTIONAL. If set to 1, the melting layer cells are not marked. By default, this option is not set.
RHOMIN	FLOAT	OPTIONAL. Default 0.75. The minimum value of Rhohv to flag the gate as suspected melting layer.
RHOMAX	FLOAT	OPTIONAL. Default 0.94. The maximum value of Rhohv to flag the gate as suspected melting layer.
NML_POINTS_MIN	INT	OPTIONAL. Default 1500. The minimum number of suspected melting layer radar bins to compute a valid melting layer.
PERCENTILE_BOTTOM	FLOAT	OPTIONAL. Default 0.3. The percentile of radar bins from which the melting layer bottom is considered to start.
NO_SANITYCHECK	INT	OPTIONAL. If set to 1, the clutter cells determined by the sanity check are not marked. By default, this option is not set.
DOTEXTURE	INT	OPTIONAL. Default 1. If set the sanity check uses information on texture of the polarimetric variables to identify clutter.
TEXTPHI	FLOAT	OPTIONAL. Default 0.266 [Deg/m]. The threshold on the texture of Phidp to consider the radar bin clutter contaminated.
TEXTRHO	FLOAT	OPTIONAL. Default 0.004 [1/m]. The threshold on the texture of Rhohv to consider the radar bin clutter contaminated.
TEXTZDR	FLOAT	OPTIONAL. Default 0.038 [dB/m]. The threshold on the texture of Zdr to consider the radar bin clutter contaminated.
TEXTZH	FLOAT	OPTIONAL. Default 0.107 [dBz/m]. The threshold on the texture of Zh to consider the radar bin clutter contaminated.
NNEIGHBOUR_MIN	INT	OPTIONAL. Default 0. If set to a value larger than 0 speckle filtering is performed. The filter checks how many valid neighbors a valid range bin has. If it is less than what is specified by this parameter the range bin is set to clutter.
NO_NOISE	INT	OPTIONAL. If set to 1, the noise cells are not marked. By default, this option is not set.
NO_CLUTTERMAP	INT	OPTIONAL. If set to 1, the clutter cells determined by the clutter map are not marked. By default, this option is not set.
MAKE_GLOBAL	INT	OPTIONAL. If set to 1, the sanity dataset is made global and can be used for the input processing for other datasets. By default, this option is not set.

Table 25: Parameters of a RAW dataset (without the general parameters).

5.1.2 RAW

Use a rainbow raw volume directly as dataset.

The general parameters *type*, *datatype*, *ignore_missing_vols* and *products* are not listed here, see Table 6 for description of these parameters.

5.1.3 WINDSHEAR

Calculate the windshear from a Doppler velocity raw volume.

The general parameters *type*, *datatype*, *ignore_missing_vols* and *products* are not listed here, see Table 6 for description of these parameters.

5.1.4 RAINRATE

Rain rate retrieval. Several rain rate retrievals are implemented. Depending on the retrieval different input data types are necessary. Each retrieval is called by using the dataset structure entry *RR_METHOD* (i.e. "hydro"). Table 26 describes the possible rain rate datasets. In retrievals that have a frequency dependence the parameter BANDPARAMS can be set. It is a string with possible values S, C and X and default value X.

Retrieval	Input Data Type	Output Data Type	Description
Z	dB(u)Z(c)	RR_Zh	Single Z-R relation.
KDP	KDP(c)	RR_Kdp	Kdp-R relation.
A	Ah(c)	RR_Ah	Ah-R relation.
ZA	dB(u)Z(c), Ah(c)	**TODO, complete**	
ZKDP	dB(u)Z(c), KDP(c)	**TODO, complete**	Use of Kdp-R or Z-R relation
ZPoly	**TODO, complete**		
hydro	dB(u)Z(c), Ah(c)	**TODO, complete**	

Table 26: Description of RAINRATE dataset types.

5.1.5 BACKSCATTER

TODO: is it still valid for pyRad?

5.1.6 SPEC_QUALITY

TODO: is it still valid for pyRad?

5.1.7 SPHERECALIBRATION

TODO: is it still valid for pyRad?

5.1.8 IQ_RHOHV

TODO: is it still valid for pyRad?

5.1.9 IQ_ZDR

TODO: is it still valid for pyRad?

5.2 RAY Datasets

5.2.1 RAINEXTENSION

TODO: is it still valid for pyRad?

5.3 TIMESERIES Datasets

5.3.1 TRAJ_ATPLANE

The values nearest to the plane during a trajectory are calculated and returned as a time series.

The parameters of a *TRAJ_ATPLANE* dataset are listed in Table 27. The general parameters *type*, *ignore_missing_vols* and *products* are not listed here, see Table 6 for description of these parameters.

TODO: if understood correctly from process_traj.py, it seems that now there is a default behavior and cube, cube_varbin are obsolete. Correct?

Parameter	Type	Description
datatype	STRING	Date type of a volume dataset. A list of generated datatypes is given in Table 5. For example the datatyp *RR_Zh* is possible to use a rainrate volume.
CUBE	INT	If set to 1, use a cube around to nearest cell (to the plane). Derive min, max, median values from this cube.
CUBE_VARBIN	INT	If set to 1, determine the size of the cube according the distance from the radar. If set to 0, use a fixed cube size.

Table 27: Parameters of a TRAJ_ATPLANE dataset (without the general parameters).

5.3.2 TRAJ_ANTENNA_PATTERN

Transform the radar cells to the view of another antenna (an antenna pattern must be given). A plane trajectory is used. And a time series of the antenna transformation is returned.

The parameters of a *TRAJ_ANTENNA_PATTERN* dataset are listed in Table 28. The general parameters *type*, *ignore_missing_vols* and *products* are not listed here, see Table 6 for description of these parameters.

Parameter	Type	Description
datatype	STRING	Date type of a volume dataset. A list of generated datatypes is given in Table 5. For example the datatyp *RR_Zh* is possible to use a rainrate volume.
ANTENNATYPE	STRING	Select the antenna type, either ”AZIMUTH” or ”ELEVATION”. The antenna pattern files are stored in the config directory and its names are defined in the location config file.
PATTERN_THRES	FLOAT	OPTIONAL. The transforming algorithm weights the radar cell by the antenna pattern. If there are undefined values, the sum of all weights must be larger than this threshold.
USE_NANS	INT	OPTIONAL. If set to 1, the NaNs are set to a fixed number (by default 0.0) and are used for the antenna pattern transformation.
NAN_VALUE	FLOAT	OPTIONAL. The value for NaNs if USE_NANS is set. By default this value is 0.0.
RANGE_ALL	INT	OPTIONAL. If set to 1, all range gates up to the plane position are used for the dataset generation (*line of sight*). By default, only the range gate of the plane is used.

Table 28: Parameters of a TRAJ_ANTENNA_PATTERN dataset (without the general parameters).

5.3.3 POINT_MEASUREMENT

Obtains the value of a polarimetric variable at a particular elevation closest to a point of interest (latitude, longitude, altitude). The input data type can be any volume. The parameters of a this dataset are listed in Table 29.

TODO: here the parameters are quite different in pyRad. -jgr- check with -fvj-. Example, config file mals_mei_dataquality_prod.txt

Parameter	Type	Description
datatype	STRING	any volume data type.
lat	FLOAT	latitude of the point of interest [decimal Deg].
lon	FLOAT	longitude of the point of interest [decimal Deg].
alt	FLOAT	altitude of the point of interest [m].
elind	INT	index of the radar elevation.

Table 29: Parameters of a POINT_MEASUREMENT dataset (without the general parameters).

5.4 TIMEARRAY Datasets

5.4.1 TIME_HEIGHT

TODO: not there, right?

5.4.2 TIME_VECTOR

TODO: not there yet, right?

5.4.3 QVP

Creates a quasi vertical profile (QVP) of data. A QVP is an azimuthal average of the data at a particular elevation angle presented as a time-height 2D plot. The parameters for QVP datasets are listed in Table 30.

TODO: check if parameters are all ok. It seems so but some spelling may differ.

Parameter	Type	Description
type	STRING	must be QVP.
datatype	STRING	Any data type.
h_max	FLOAT	desired maximum height of the QVP [m MSL]. Default=50
h_res	FLOAT	desired height resolution [m]. Default=10000
ind_el	INT	index of the elevation angle used for the QVP. Default= 0

Table 30: Parameters of a QVP dataset.

5.5 TIMEARRAY_ARRAY Datasets

5.5.1 TIME_SUMMARY_HEIGHT

TODO: not there yet, right?

5.6 TRAJ_ONLY Datasets

5.6.1 TRAJ

Use a trajectory without any radar data. Note, just a few product types as plotting of the trajectory range, azimuth and elevation can be applied on this dataset.

The general parameters *type*, *datatype*, *ignore_missing_vols* and *products* are not listed here, see Table 6 for description of these parameters.

5.6.2 TRAJ_PLANEHITS

TODO: not there yet, right?

5.7 TRAJ_VOL_COMPOSITE Datasets

5.7.1 TRAJ_VOL

TODO: not there yet, i guess. And not clear to me this dataset. -jgr-

5.8 VOLUMEGROUP Datasets

TODO: not there yet, i guess.

5.8.1 QUANTILE

TODO: not there yet? or obsolete?

5.9 PSR Datasets

TODO: to clarify the plan for PSR

5.9.1 PSR_RAW

5.9.2 PSR_MERGE

5.9.3 PSR_VERTICAL

5.9.4 PSR_SZH

5.9.5 PSR_SZV

5.9.6 PSR_SZDR

5.9.7 PSR_SZDR_TEXT

5.9.8 PSR_SRHOHV

5.9.9 PSR_SPHIDP

5.9.10 PSR_SPHIDP_TEXT

5.9.11 PSR_DSD

5.10 ARRAY_PSR Datasets

5.10.1 PSR_FILTER

5.10.2 PREPROC_PSR

5.11 MONITORING Datasets

5.11.1 MONITORING

TODO -jgr-: just add a reference to latest Jordi documentation

Polarimetric data quality monitoring. The inputs can be dB(u)Z(c), ZDR(c), uPhiDP, PhiDP(c), RhoHV(c) and ISO0. The function performs 4 different monitoring routines: 1) monitoring of the phidp offset, 2) monitoring of the Zdr in moderate rain (expected 0.2 dB),

3) monitoring of the 80th quantile in rain (expected well above 0.98), 4) monitoring of the Zh bias by means of self-consistency in rain. The routines are user selectable. By default all the monitoring routines are performed. The ISO0 information is used in monitoring routines 2 to 4. By default no monitoring is performed if ISO0 information is missing but the user may select to allow missing ISO0 volumes. The ISO0 is then assumed to be at a fix height of 2000 m. The user-selectable parameters for this function are summarized in Table 31.

This process is intended to be performed off-line. It does not provide any output until all the volumes (Typically all the volumes generated in a day) have been processed. The output is a structure of type "MON" that contains general information of the volumes used (azimuth angles, elevation angles, range bins and range resolution), as well as the information for each polarimetric variable that has been processed (number of volumes used, time of each volume, space-time matrix with the data and number of samples and statistical parameters.

TODO: it seems that less parameters are now used. Correct? Which one is a good example as a config file? Checking now mals_mei_dataquality_prod

Parameter	Type	Description
IGNORE_MISSING_VOL	INT	Boolean (0,1), default: 0. If set to 1, preprocessing is performed despite missing volumes.
MONPHIDP0	INT	Boolean (0,1), default: 1. If set to 1, Phidp offset monitoring is performed.
MONZDRBIAS	INT	Boolean (0,1), default: 1. If set to 1, monitoring of the Zdr in moderate rain is performed.
MONRHOAV	INT	Boolean (0,1), default: 1. If set to 1, monitoring of the RhoHV value in rain is performed.
MONZHBIAS	INT	Boolean (0,1), default: 1. If set to 1, monitoring of the Zh bias is performed.
ALLOW_MISSING_ISO0	INT	Boolean (0,1), default 0. If set to 1, monitoring is performed even if ISO0 volumes are missing. The ISO0 height is assume to be 2000 m
ZDRBIAS0	FLOAT	Default: 0. ZDR bias. This parameter is used in the monitoring of Zdr when ZDRc is the input to uncorrect for any possible bias that has been previously corrected
ZDRBIAS0	FLOAT	Default: 0. Zh bias. This parameter is used in the monitoring of Zh bias when dBZc is the input to uncorrect for any possible bias that has been previously corrected

Table 31: Parameters of MONITORING (without the general parameters).

5.12 ARRAY_VOL Datasets

5.12.1 PREPROCESSING

TODO: is this replaced by the “processing levels” somehow? Should check one by one the preprocessing here (IDL) to see the equivalent in pyrad -jgr- -fvj-

Pre-processing of the raw polarimetric data. The inputs can be dB(u)Z, dBZ(u)v, ZDR, (u)PhiDP, RhoHV, KDP, CDR and the auxiliary volumes SNRh (used in phidp processing), ISO0 (used in attenuation correction processing) and VIS (used for partial beam blockage correction).

The pre-processing performed is user selectable and can consist in clutter suppression, partial beam blocking correction, Zdr and Zh bias correction, kdp and phidp retrievals and/or attenuation correction.

The output is an array of volumes that may contain some or all of these data types: dBZc, dBZvc, ZDRc, PhiDPc, KDPc, RhoHVc, CDRc, Ah or Adp, depending on the user selected processing and whether it has been successful[1]. The output volumes are added to the raw volumes to be processed and can be used by other functions specifying for example: PROC:dBZc The user may decide not to issue any product for this data set. The parameters of PREPROCESSING are listed in Table 32.

IMPORTANT: This dataset must be processed right after the sanity volume and before any other data set so that the output volumes can be used by others.

Parameter	Type	Description
IGNORE_MISSING_VOL	INT	Boolean (0,1), default: 0. If set to 1, preprocessing is performed despite missing volumes.
CLUTTERSUPPRESSION	INT	Boolean (0,1), default: 0. If set to 1, clutter suppression is performed.
VISIBILITYCORRECTION	INT	Boolean (0,1), default: 0. If set to 1, partial beam blockage correction is performed.
VISMIN	FLOAT	Possible values: 0 to 100. default: 0. Minimum visibility. Any range bin that has a lower visibility than this value will be set to -INF.
BIASCORRECTION	INT	Boolean (0,1), default: 0. If set to 1, dBZ and ZDR are corrected with the value specified by the user.
ZDRBIAS0	FLOAT	ZDR bias in dB
ZHBIAS0	FLOAT	dBZ bias in dBZ
DOKDPPHIDP	INT	Boolean (0,1). If set to 1, uPhiDP is processed to get PhiDP and KDP. **If set, any parameter accepted by the ARRAY_VOLPHIDP_KDP dataset, can be passed to define and tune the estimation methods (refer to Section 5.12.6).**
ATTENUATIONENABLE	INT	Boolean (0,1). If set to 1, dBZ and ZDR are corrected for rain-induced attenuation. **If set, any parameter accepted by the ARRAY_VOLATTENUATION dataset, can be passed to define and tune the methods (refer to Section 5.12.4).**

Table 32: Parameters of PREPROCESSING (without the general parameters).

An example of application of PREPROCESSING, inside a product configuration file is the following one:

```
preprocessing STRUCT 9
   TYPE STRING PREPROC
   DATATYPE STRARR 6
      dBZ
      ZDR
      RhoHV
      PhiDP
      KDP
```

[1]Note the suffix -c- in processed data.

```
   COSMO:ISO0
IGNORE_MISSING_VOLS INT 1
CLUTTERSUPPRESSION INT 1
DOKDPPHIDP INT 0
ATTENUATIONENABLE INT 1
ATTCORRMETHOD STRING zphi
BIASCORRECTION INT 1
ZDRBIAS0 FLOAT 0.2
```

5.12.2 MEANZ

TODO: outdated?

5.12.3 XVSC_POSITION

TODO: outdated? or replaced by something with different name?

5.12.4 ATTENUATION

TODO: (1) checking the function in process_phase.py (function named process_attenuation.py) it seems that now only zphi can be used. Correct? (2) check if input parameters in table below are outdated. It seems so.

The parameters for this dataset are listed in Table 33.

Parameter	Type	Description
attcorrmethod	STRING	The attenuation estimation method. One of the following: selfcons, zphi, philinear
mu	FLOAT	Only for selfcons method. The assumed mu parameter of the gamma distribution
zhbias0	FLOAT	Only for selfcons method. The Zh bias to be corrected before applying the method
zdrbias0	FLOAT	Only for selfcons method. The Zdr bias to be corrected before applying the method
phiparammethod	STRING	Possible: MFranceX, MFranceC, MFranceS, RyzhkovX. Default MFranceX. Selects the parameters for attenuation correction

Table 33: Parameters of ATTENUATION.

5.12.5 HYDROCLASS

TODO: here now is replaced by semisupervised, right? How can i find proper parameters?

Hydrometeor classification performed with a selected method (among the available ones). Hydrometeor classification maps a set of volume inputs (radar-based or from external sources) into a discrete set of hydrometeor classes (indicated with integers). Each method has its own conventions.

This dataset is an array_volume because side products may be provided: for instance the uncertainty of classification, the second-likely hydrometeor class and so on.

5.12.6 PHIDP_KDP

TODO: in general, does still hold the definition of ARRAY_VOL dataset? TODO2:

this should be splitted in all the methods -jgr-. Need some support from -fvj- on how to find the accepted parameters with some certainty.

K_{dp} and Φ_{dp} estimated from noisy Ψ_{dp} (Ψ_{dp} includes two unwanted components: (i) system noise, and (ii) differential phase shift upon backscattering δ_{hv}). Usually Φ_{dp} is estimated first, and then K_{dp} in series. Any combination of Φ_{dp} and K_{dp} methods can be used. This dataset expects a sanity volume. This ARRAY_VOL dataset is composed by the KDP volume (datatype 'KDP') and PHIDP volume (datatype 'PHIDP').

HERE HERE HERE

5.13 SUN_MONITORING Datasets

5.13.1 SUN_MONITORING

TODO: substituded by "sun_hits", right?

5.14 XVSC Datasets

5.14.1 XVSC

TODO: is it outdated?

5.15 INTERCOMP Datasets

5.15.1 INTERCOMP

TODO: i found INTERCOMP in the function process_calib.py but still would like to find a way to understand all the parameters that are accepted through config file (similar question for other datasets/products). Could not find an example of this dataset in the config files.

Compares a polarimetric variable of a master radar with a slave radar. The data type can be any data type recognized by both radars. This dataset makes use of the list of collocated points obtained by XVSC_POSITION. The points used in the comparison are stored in a text file with the following information: time in format YYYYYMMDDhhmmss, elevation, range and azimuth indices of the master radar, polarimetric variable value of master radar, elevation, range and azimuth indices of slave radar, polarimetric variable value of slave radar. The list of parameters for this dataset can be seen in Table 34.

Parameter	Type	Description
RADAR	STRARR	list of radars that are compared to processed radar (Ex. D, MXPOL, etc.).
RADARRES	STRING	Radar resolution (H or L).
ScanList	STRARR	List of MXPOL radar scans where to look for data.

Table 34: Parameters of INTERCOMP.

5.15.2 INTERCOMP_TIME_AVG

TODO: add description (novelty with respect to IDL?)

5.16 HISTOGRAM Datasets

5.16.1 HISTOGRAM

TODO: does this still exist as dataset or is it a product?

5.17 COMPOSITE_VOLUME datasets

TODO: check how this would fit in the context of pyrad.

6 Products

For each dataset format several products can be generated. The following tables list the possible products for each dataset format.

Name	Description	Reference
PPI_IMAGE	PPI image of constant elevation	Section 6.1.1
PPI_MAP	PPI image on a map	**TODO: describe it**
PSEUDOPPI_IMAGE	**TODO: describe**	**TODO: describe it**
PSEUDOPPI_MAP	**TODO: describe**	**TODO: describe it**
RHI_IMAGE	RHI image of constant azimuth	Section 6.1.4
RHI_PROFILE	Averaged height profile	Section 6.1.5
PSEUDORHI_IMAGE	**TODO: describe**	**TODO: describe it**
CAPPI_IMAGE	Constant altitude PPI image	Section 6.1.6
PLOT_ALONG_COORD	**TODO: describe**	**TODO: describe it**
BSCOPE_IMAGE	**TODO: describe**	**TODO: describe it**
TIME_RANGE	**TODO: describe**	**TODO: describe it**
HISTOGRAM	**TODO: describe**	**TODO: describe it**
QUANTILES	**TODO: describe**	**TODO: describe it**
FIELD_COVERAGE	**TODO: describe**	**TODO: describe it**
CDF	**TODO: describe**	**TODO: describe it**
SAVEVOL	Save the generated dataset volume	Section 6.1.10
SAVEALL	**TODO: describe**	**TODO: describe it**
SAVESTATE	Save the time of the processed volume.	Section 6.1.11
	TODO: check -jgr- and -fvj- all the products below, what to do.	
WGS84_IMAGE	Image in WGS84 coordinates	Section 6.1.2
CH1903_IMAGE	Image in Swiss coordinates	Section 6.1.3
CAPPI_ASCII	XXX to be described	Section 6.1.7
PLOT_LINES	Plot values along a coordinate, holding the other two fixed.	Section 6.1.8
CDF_STAT	Cumulative distribution function	Section 6.1.9
NETCDF_CONV	Save data in netcdf file format	Section 6.1.12
MELTLAYER_IMAGE	Azimuth-Height graphic indicating the areas suspected to belong to the melting layer	Section 6.1.13
MELTLAYER_TS	Time series plot with the evolution of the melting layer.	Section 6.1.14
SAVE_DEM	Save a PPI in DEM format.	Section 6.1.15
RNGVSANG_IMAGE	Range versus angle image at a particular elevation or azimuth.	Section 6.1.16
SAVESLICE_PPI_ASCII	XXX to be described	Section 6.1.17
CONST_RANGE_IMAGE	Make a azimuth elevation plot at fixed range.	Section 6.1.18
CONTOUR_RANGE_IMAGE	make a contour plot (azimuth vs. elevation) at a fixed range gate	Section 6.1.19
CONTOUR_RANGE_IMAGE_3D	make a contour plot (azimuth vs. elevation) at a fixed range gate	Section 6.1.20
WRITE_BIN	Write bin values to a file.	Section 6.1.21
WRITE_MEAN	[illegible]	Section 6.1.22

Name	Description	Reference
WRITE_SUN_HITS	**TODO: describe**	**TODO**
PLOT_SUN_HITS	**TODO: does it replace psunhits below?**	**TODO**
WRITE_SUN_RETRIEVAL	**TODO: describe**	**TODO**
PLOT_SUN_RETRIEVAL	**TODO: describe**	**TODO**
PLOT_SUN_RETRIEVAL_TS	**TODO: describe**	**TODO**
	TODO: check what to do with the IDL products below	
PSUNHITS_IMAGE	Creates a 2D plot where the x axis is the difference between azimuth position of the radar and the azimuth position of the sun, the y axis is the difference between elevation position of the radar and elevation position of the sun and color coded us the estimated sun hits power	Section 6.2.1
PSUNRETRIEVAL_IMAGE	As above but with the retrieved sun power	Section 6.2.2
ZDRSUNHITS_IMAGE	As above but with the Zdr of the sun hits	Section 6.2.3
ZDRSUNRETRIEVAL_IMAGE	as above but with the retrieved Zdr of the sun	Section 6.2.4
SUNRETRIEVAL_TS	Plot a time series showing the evolution of a sun retrieval parameter	Section 6.2.5

Table 36: List of specific product types for dataset with SUN_HITS format. In addition to them, all the product types for dataset with VOL format can be applied.

Name	Description	Reference
	TODO: outdated? or replaced?	
POLAR_AZ_EL_IMAGE	A polar azimuth elevation plot	Section 6.3.1
DISTANCE_VS_AZIMUTH_IMAGE	Make a 2D "data" vs azimuth plot	Section 6.3.2
QUANTILE_STAT	Make "data" vs azimuth statistics	Section 6.3.3

Table 37: List of all possible product types for dataset with RAY format

Name	Description	Reference
PLOT_AND_WRITE_POINT	Plots a time series with the evolution of a variable at a particular point. Writes the same information in a file.	Section 6.4.2
PLOT_CUMULATIVE_POINT	Plots a time series with the accumulation in time of a variable at a particular point.	Section 6.4.3
COMPARE_POINT	Time series plot showing the evolution of a radar variable and a variable from another sensor placed at a particular point.	Section 6.4.4
COMPARE_CUMULATIVE_POINT	Time series plot showing the accumulated value of a radar variable and a variable from another sensor placed at a particular point.	Section 6.4.5
COMPARE_TIME_AVG	**TODO**	**TODO**
PLOT_AND_WRITE	Make a plot of a timeseries. And write the timeseries to a file.	Section 6.4.1

Table 38: List of all possible product types for dataset with TIMESERIES format

Name	Description	Reference
	TODO: is it still of interest? is it replaced by QVP?	
TIME_ARRAY_IMAGE	Plot time-array plot (e.g. a time-height array).	Section 6.5.1
TIME_ARRAY_FILE	XXX to be described	Section 6.5.2

Table 39: List of all possible product types for dataset with TIMEARRAY format

Name	Description	Reference
	TODO: is it still of interest?	
TIME_ARRAY_IMAGE	Plot time-array plot (e.g. a time-height array).	Section 6.6.1
TIME_ARRAY_MAP	Plot time-array on a map	Section 6.6.2

Table 40: List of all possible product types for dataset with TIMEARRAY_ARRAY format

Name	Description	Reference
	TODO: now TRAJ and TRAJ_ONLY are merged in Pyrad, right?	
TRAJ_PLOT	Plot the range, elevation, azimuth of a plane trajectory.	Section 6.7.1
TRAJ_TEXT	Write the range, elevation, azimuth of a plane trajectory to a text file.	Section 6.7.2
TRAJ_ANTENNA	Plot the radar antenna movement.	Section 6.7.3
TRAJ_DX50_ANTENNA_HITS	Compare the DX50 antenna movement with a plane trajectory. Possible hits are listed.	Section 6.7.4

Table 41: List of all possible product types for dataset with TRAJ_ONLY format

Name	Description	Reference
	TODO: of interest?	
RHI_TRAJ	Overplot the trajectory over a RHI image with azimuth according to the planes azimuth angle.	Section 6.8.1
PPI_TRAJ	Overplot the trajectory over a PPI image with elevation according to the planes elevation angle.	Section 6.8.2

Table 42: List of all possible product types for dataset with TRAJ_VOL_COMPOSITE format

Name	Description	Reference
	TODO: of interest?	
SPECTRUM_IMAGE	Plot of a single spectrum	Section 6.9.1
VERTICAL_SPECTRUM_IMAGE	XXX to be described	Section 6.9.2
SINGLE_SPECTRUM_IMAGE	XXX to be described	Section 6.9.3
SAVE_PSR_VOLUME	XXX to be described	Section 6.9.4
SPEC_ALONG	Plot the spectrum along the range, along the azimuth or along an elevation	Section 6.9.5
IQ_ALONG	Plot the IQ data along the range, along the azimuth or along an elevation	Section 6.9.6

Table 43: List of all possible product types for dataset with PSR format

Name	Description	Reference
VOL_HISTOGRAM	**TODO: describe**	**TODO: describe**
PPI_HISTOGRAM	**TODO: describe**	**TODO: describe**
ANGULAR_DENSITY	**TODO: describe**	**TODO: describe**
VOL_TS	**TODO: describe**	**TODO: describe**
SAVEVOL	**TODO: describe**	**TODO: describe**
	TODO: check what to do with legacy products below	
WRITE_PHIDP0	Writes the information of the monitoring of the differential phase offset.	Section 6.10.1
WRITE_ZDRBIAS	Writes the information of the monitoring of the Zdr value in moderate rain.	Section 6.10.2
WRITE_RHOAV	Writes the information of the monitoring of the Rhohv in rain.	Section 6.10.3
WRITE_ZHBIAS	Writes the information of the monitoring of the reflectivity bias.	Section 6.10.4
PLOT_PHIDP0	Plots the differential phase offset as a function of azimuth for one elevation.	Section 6.10.5
PLOT_ZDRBIAS	Plots the Zdr in moderate rain as a function of azimuth for one elevation.	Section 6.10.6
PLOT_ZHBIAS	Plots the Zh bias as a function of azimuth for one elevation.	Section 6.10.7
PLOT_PHIDP0_DAY_TS	Plots the instantaneous evolution of the differential phase offset.	Section 6.10.8
PLOT_ZDRBIAS_DAY_TS	Plots the instantaneous evolution of the Zdr in moderate rain.	Section 6.10.9
PLOT_RHOAV_DAY_TS	Plots the instantaneous evolution of the RhoHV in rain.	Section 6.10.10
PLOT_ZHBIAS_DAY_TS	Plots the instantaneous evolution of the Zh bias.	Section 6.10.11
PLOT_AND_WRITE_PHIDP0_TS	Plots the evolution on a daily basis of the phidp offset. Writes the same information in a file	Section 6.10.12
PLOT_AND_WRITE_ZDRBIAS_TS	Plots the evolution on a daily basis of the Zdr in moderate rain. Writes the same information in a file	Section 6.10.13
PLOT_AND_WRITE_RHOAV_TS	Plots the evolution on a daily basis of the RhoHV in rain. Writes the same information in a file	Section 6.10.14
PLOT_AND_WRITE_ZHBIAS_TS	Plots the evolution on a daily basis of the Zh bias. Writes the same information in a file	Section 6.10.15

Table 44: List of all possible product types for dataset with MONITORING format

Name	Description	Reference
	TODO: Are XVSC substituted/merged by INTERCOMP format?	
PLOT_INTERCOMP	scatter plot between the pairs of time-averaged reflectivity data at each (or all) elevations. Computation of various statistics	Section 6.11.1
PLOT_AND_WRITE_INTERCOMP_TS	Computes various statistics and plots the daily evolution of the median bias	Section 6.11.2

Table 45: List of all possible product types for dataset with XVSC format

Name	Description	Reference
PLOT_INTERCOMP	scatter plot between the pairs of polarimetric data at each (or all) elevations. Computation of various statistics	Section 6.12.1

Table 46: List of all possible product types for dataset with INTERCOMP format

Name	Description	Reference
	TODO: Are those merged in VOL?	
WRITE_2DHIST	writes a 2D histogram in a file	Section 6.13.1
PLOT_2DHIST	plots a 2D histogram	Section 6.13.2
WRITE_HIST	writes a 1D histogram in a file	Section 6.13.3
PLOT_HIST	plots a 1D histogram	Section 6.13.4

Table 47: List of all possible product types for dataset with HISTOGRAM format

Name	Description	Reference
	TODO: what to do with them?	
PPI_RHI_IMAGE	Plot one PPI and up to 3 RHIs on the same image.	Section 6.14.1.
CELL_3D_IMAGE	Plot in 3D composite volume data inside a bounding box centered along a given RHI direction	Section 6.14.2.
RADAR_3D_IMAGE	Plot in 3D composite volume data over a given spatial domain	Section 6.14.3

Table 48: List of all possible product types for COMPOSITE_VOLUME datasets.

TODO: in Pyrad there are also: COLOCATED_GATES_PRODUCTS, QVP_PRODUCTS, GRID_PRODUCTS, COSMO_COORD_PRODUCTS. Clarify how those relate with the IDL framework.

TODO: uncomment the description of the products below here, when defined which ones to keep

6.1 VOL Products