wradlib 1.13. Getting Started Cheat Sheet

Learn more about *wradlib* at https://wradlib.org

wradlib Introduction

The *wradlib* project has been initiated in order facilitate the use of weather radar data as well as to provide a common platform for research on new algorithms. *wradlib* is an open source library which is well documented and easy to use. It is written in the free programming language Python.

Installation

We recommend using ***conda/mamba*** package manager alongside the ***conda-forge*** community channel:

- Install Mambaforge [1]
- Create dedicated *wradlib* environment:
  `$ mamba create -name wradlib python=3.9`
- Activate *wradlib* environment:
  `$ conda activate wradlib`
- Install *wradlib* and other needed packages:
  `(wradlib)$ mamba install wradlib jupyter`

If you want to test the most recent *wradlib* developments, then you need to get the latest **master** from *github.com* in addition:

- Clone *wradlib* repository
  `$ git clone https://github.com/wradlib/wradlib.git`
- Install *wradlib* from sources:
  `$ python -m pip install .`

If you want to test the provided example notebooks, you need to download the example data [2] and extract it to an arbitrary directory. Finally set the **WRADLIB_DATA** environment variable:

- `$ export WRADLIB_DATA=/full/path/to/wradlib-data`

[1] https://github.com/conda-forge/miniforge/releases
[2] https://github.com/wradlib/wradlib-data/archive/master.zip

Getting Started

`>>> import wradlib as wrl`	Import using wrl as alias
`>>> wrl.__version__`	Print wradlib version

References

[1] Maik Heistermann, Stephan Jacobi, and Thomas Pfaff. Technical note: An open source library for processing weather radar data (wradlib). *Hydrol. Earth Syst. Sci.*, 16:863–871, 2013.

Reading Radar Data

- all functions imported from **wrl.io**
- Polar Radar Data Reader

`>>> img, meta = read_dx(f)`	DWD's DX
`>>> data = open_odim_dataset(f)`	ODIM_H5
`>>> data = open_gamic_dataset_(f)`	GAMIC
`>>> data = open_cfradial1_dataset(f)`	CfRadial1
`>>> data = open_cfradial2_dataset(f)`	CfRadial2
`>>> data = open_rainbow_dataset(f)`	Rainbow5
`>>> data = open_iris_dataset(f)`	Sigmet

- Gridded Radar Data Reader

`>>> data = open_radolan_dataset(f)`	RADOLAN
`>>> data = read_rainbow(f)`	Rainbow5
`>>> data = read_iris(f)`	Sigmet

- Generic Data Format Reader

`>>> data = read_generic_hdf5(f)`	HDF5
`>>> data = read_generic_netcdf(f)`	NetCDF

- Raster Data Reader using GDAL

`>>> ds = open_raster(f)`	open raster
`>>> img, crd, proj = wrl.georef.extract_raster_dataset(ds)`	extract raster data

Visualizing Radar Data

- functions imported from **wrl.vis** or **WradlibAccessor**
- Plot Polar Radar Data **da=xarray.DataArray**

`>>> da.wradlib.plot_ppi()`	plot simple PPI
`>>> da.wradlib.plot_ppi(proj="cg")`	Curvelinear Grid
`>>> da.wradlib.plot_rhi()`	plot simple RHI
`>>> da.wradlib.plot_rhi(proj="cg")`	Curvelinear Grid

- Plot Polar Radar Data **img(nrays, nbins)**

`>>> plot_ppi(img)`	plot simple PPI
`>>> plot_ppi(img, proj="cg")`	Curvelinear Grid
`>>> plot_rhi(img)`	plot simple RHI
`>>> plot_rhi(img, cg=True)`	Curvelinear Grid

- Plot Gridded Radar Data **img(nrows, ncols)**

`>>> import matplotlib.pyplot as plt`	matplotlib
`>>> pl.imshow(img)`	use imshow
`>>> pl.pcolormesh(img)`	use MeshPlot

Data Transformation

`>>> y = wrl.trafo.rvp_to_dbz(x)`	RVP6 in dBZ
`>>> dBZ = wrl.trafo.decibel(Z)`	decibel
`>>> Z = wrl.trafo.idecibel(dBZ)`	inverse decibel
`>>> RR = wrl.trafo.kdp_to_r(KDP)`	Rainrate from KDP
`>>> RR = wrl.zr.z_to_r(Z)`	Rainrate from Z
`>>> Z = wrl.zr.r_to_z(RR)`	Z from RainRate

Data Classification

• wrl.clutter.filter_gabella()	Clutter id filter by Gabella
• wrl.clutter.filter_cloudtype()	Filter based on cloud type
• wrl.clutter.filter_window_distance()	2D filter large gradients
• wrl.clutter.histo_cut()	Histogram clutter id
• wrl.clutter.classify_echo_fuzzy()	Dual-Pol fuzzy method

Data Correction

Gate-by-Gate approaches **wrl.atten**

• correct_attenuation_hb()	Hitschfeld&Bordan
• correct_attenuation_constrained()	iterative Kraemer (ext. by Jacobi)

Phase Processing

Phase Unfolding

• wrl.dp.unfold_phi()	unfolds ambiguous phase
• wrl.dp.unfold_phi_vulpiani()	KDP based unfolding

KDP Retrieval

• wrl.dp.kdp_from_phidp()	Lanczos derivative
• wrl.dp.process_raw_phidp_vulpiani()	2-step PHIDP/KDP

Data Compositing

• wrl.comp.togrid()	polar to grid
• wrl.comp.compose_ko()	quality knockout criterion
• wrl.comp.compose_weighted()	quality weighted average

Other Resources

Check out the other available *wradlib* Cheat Sheets which will be available shortly. Those will cover amongst others Visualisation, Georeferencing, Interpolation, Classification, Correction, Phase Processing, Compositing, Zonal Statistics, Gage Adjustment.

Contact

Web wradlib.org
Email wradlib@wradlib.org
Gitter gitter.im/wradlib