pynasonde — Ionosonde Data Analysis in Python

A Hands-On Workshop for Graduate Students

Ionospheric Remote Sensing Group

VIPIR & DIGISONDE Analysis Toolkit

March 28, 2026

`pip install pynasonde` — github.com/shibaji7/pynasonde

Outline

1. Background
2. Installation & Quick Start
3. DIGISONDE Data
4. VIPIR Data
5. Ionospheric Analysis
6. Es Layer Imaging
7. Hands-On Exercises
8. Summary

What is an Ionosonde?

- A **pulsed HF radar** ($\sim$1–30 MHz) transmitting vertically
- Each frequency reflects at the height where $f_p(h) = f$
- Records: virtual height $h'(f)$ vs frequency — the **ionogram**
- **Two instruments** in this toolkit:
  - **VIPIR** — 8-channel digital receiver, raw IQ storage
  - **DIGISONDE DPS4D** — SAO/RSF/DFT/DVL file formats

Key quantities

f	sounding frequency
$h'(f)$	virtual height (km)
f_p	plasma frequency
$N(h)$	electron density

Two Instrument Families

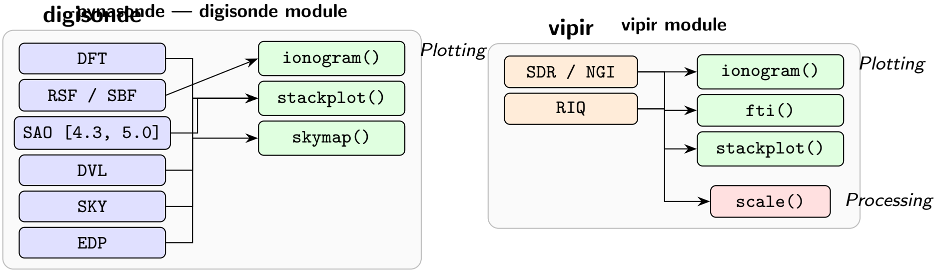
VIPIR

- Grubb et al. (2011)
- Raw IQ binary (`.RIQ`)
- 8 Rx channels, pulse-compressed
- Gate spacing: 1.499 km (WI937)
- 4 pulses/frequency (typical)
- Full analysis: Es imaging, N(h), absorption

DIGISONDE DPS4D

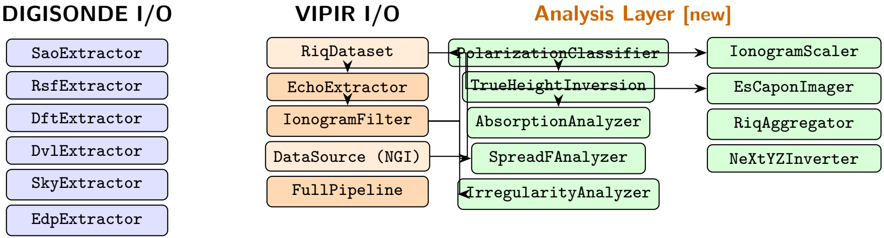
- Reinisch et al. (1997)
- Scaled data: `.SAO`
- Raw soundings: `.RSF`, `.SBF`
- Doppler spectra: `.DFT`
- Drift velocities: `.DVL`
- Sky maps: `.SKY`

pynasonde v1.0 Architecture (Chakraborty et al., SoftwareX 2026)



Adapted from Fig. 2, Chakraborty et al. (2026), *SoftwareX* 34, 102617.

Post-v1.0 Extensions — Analysis Layer



Orange = new since v1.0 publication. Green = analysis layer (entirely post-v1.0).

Installation

Terminal

```
# Option 1: PyPI
pip install pynasonde

# Option 2: developer install (recommended for this workshop)
git clone https://github.com/shibaji7/pynasonde.git
cd pynasonde
pip install -e ".[dev]"
```

Requirements

Python $\geq$ 3.11 — numpy, pandas, scipy, matplotlib, loguru

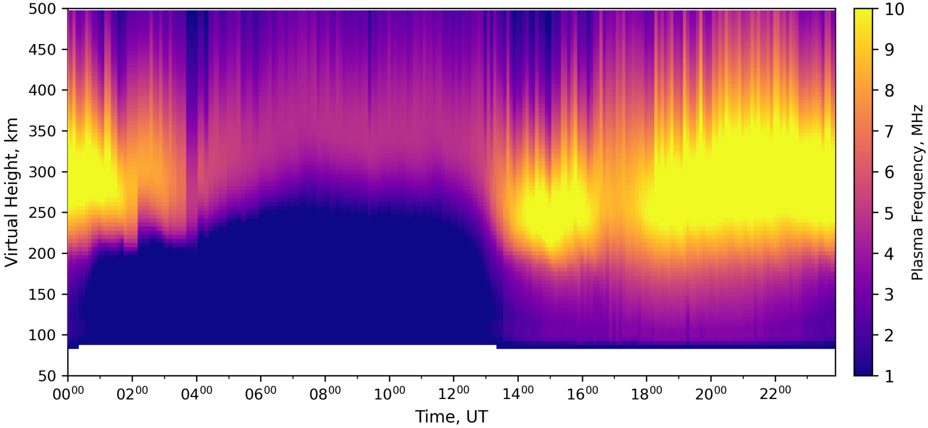
SAO — Electron Density Profiles

The `.SAO` file contains scaled ionogram parameters and height profiles:

```
from pynasonde.digisonde.parsers.sao import SaoExtractor

# Load a day of SAO files in parallel (n_procs workers)
df = SaoExtractor.load_SAO_files(
    folders=["data/KR835/"],
    func_name="height_profile",
    n_procs=4,
)
print(df.columns.tolist())
# ['datetime', 'height_km', 'plasma_freq_mhz', 'electron_density_cm3',
    ...]
```

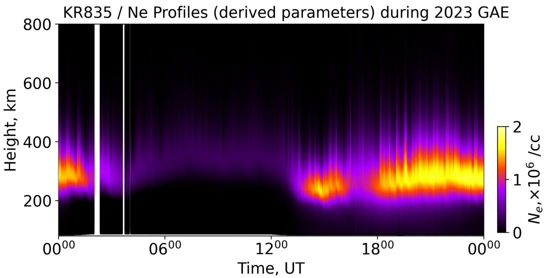
- `func_name="height_profile"` — N(h) profile per record
- `func_name="scaled_params"` — foF2, hmF2, MUF, foE

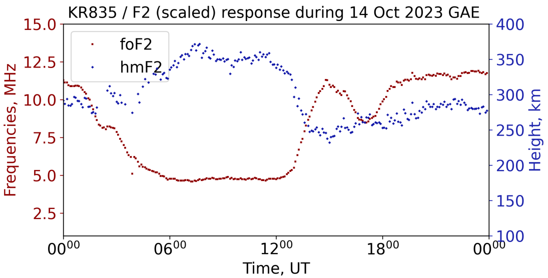
SAO — Isodensity Contour Plot



Daily isodensity contour from DPS4D SAO files at KR835. Each horizontal slice is one N(h) profile; colour encodes electron density.

SAO — N(h) Profile and F2 Parameters





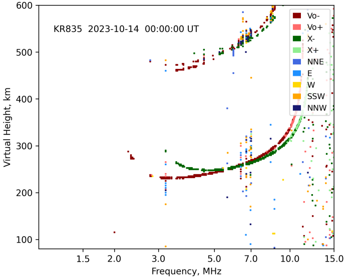
Left: stacked N(h) profiles. Right: daily foF2 and hmF2 time series.

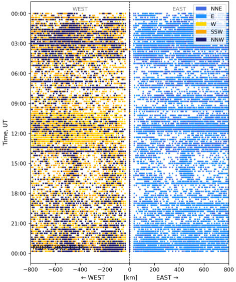
RSF — Direction-Coded Ionogram

.RSF files store raw amplitude+phase for each sounding:

```
from pynasonde.digisonde.parsers.rsf import RsfExtractor

extractor = RsfExtractor("data/KR835_20230601.RSF")
extractor.extract()
df = extractor.to_dataframe()
extractor.plot_direction_ionogram()    # direction-coded ionogram
extractor.plot_directogram()           # full-day directogram
```

RSF — Direction-Coded Ionogram & Directogram





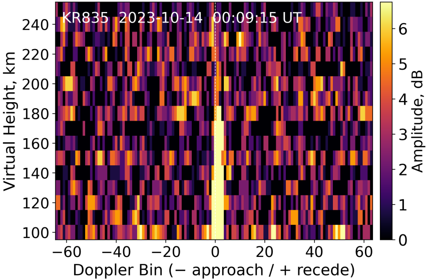
Left: direction-coded ionogram (colour = arrival azimuth). Right: daily directogram (time vs. ground distance).

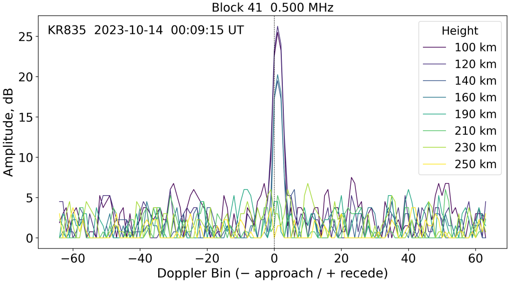
DFT — Doppler Waterfall & Spectra

.DFT files contain per-height Doppler spectra:

```
from pynasonde.digisonde.parsers.dft import DftExtractor

dft = DftExtractor("data/KR835_20230601.DFT")
dft.extract()
dft.plot_doppler_waterfall()   # time-height Doppler waterfall
dft.plot_doppler_spectra()     # per-height spectra
```

DFT — Doppler Waterfall & Spectra



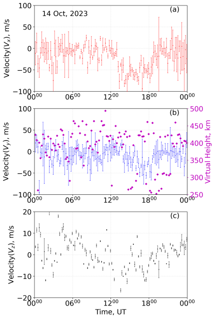


Left: Doppler velocity waterfall (height vs. time). Right: per-height Doppler spectra at one epoch.

DVL — Drift Velocity Stack Plot

.DVL files contain the ionospheric drift velocity estimates:

```
from pynasonde.digisonde.parsers.dvl import DvlExtractor

dvl = DvlExtractor.load_files(folder="data/KR835/", n_procs=4)
dvl.plot_stack()   # 3-panel Vx, Vy, Vz vs. time + virtual height overlay
```



VIPIR RIQ File Structure

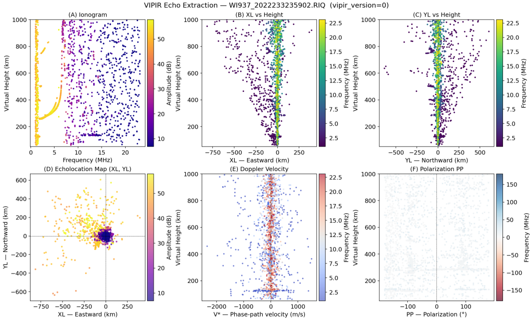
Sounding Control Table (SCT) | instrument configuration

PCT #1: frequency, time, IQ data (gates × rx)
PCT #2: ...
PCT #N: ...

```python
from pynasonde.vipir.riq.parsers.read_riq import RiqDataset,
    VIPIR_VERSION_MAP

riq = RiqDataset.create_from_file(
    "data/WI937_20220821.RIQ",
    vipir_config=VIPIR_VERSION_MAP.configs[1],   # older WI937 format
)
print(f"Gates  : {riq.sct.timing.gate_count}")   # 960
print(f"Freqs  : {riq.sct.frequency.base_steps}")
```

VIPIR — Echo Extraction

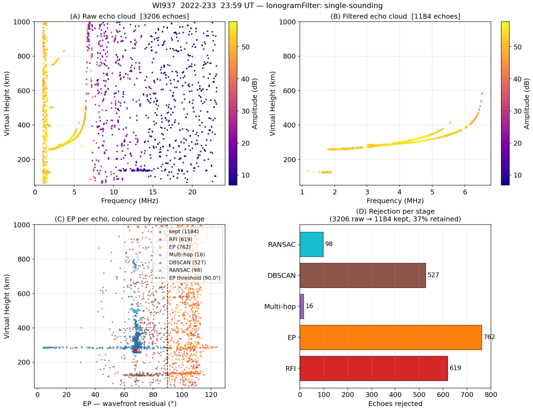
```
from pynasonde.vipir.riq.echo \
    import EchoExtractor

extractor = EchoExtractor(
  sct=riq.sct,
  pulsets=riq.pulsets,
  snr_threshold_db=3.0,
  min_rx_for_direction=3,
).extract()

df = extractor.to_dataframe()
```



Six-panel diagnostic: ionogram, XL/YL direction cosines, Doppler V*, PP polarization.

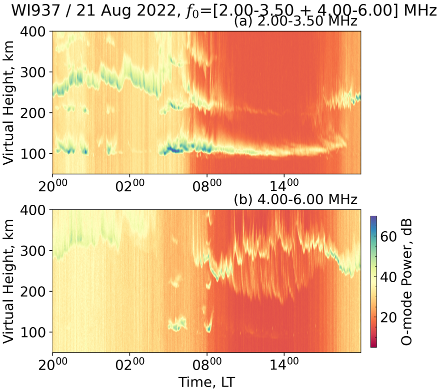
VIPIR — Ionogram Filter

```
from pynasonde.vipir.riq.parsers.\
    filter import IonogramFilter

filtered = IonogramFilter(
    snr_threshold_db=6.0,
    ep_threshold_deg=30.0,
    use_dbscan=True,
    use_ransac=True,
).fit(df)
```



6 stages: RFI $\rightarrow$ EP $\rightarrow$ multi-hop $\rightarrow$ DBSCAN $\rightarrow$ RANSAC $\rightarrow$ temporal coherence

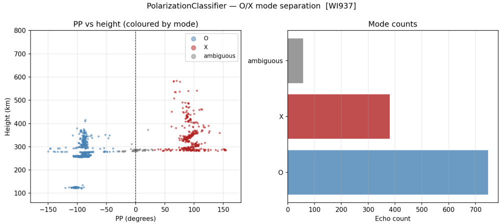
VIPIR — FTI (Frequency–Time Interval)



O-mode power FTI panels from a full day of WI937 NGI ionograms.

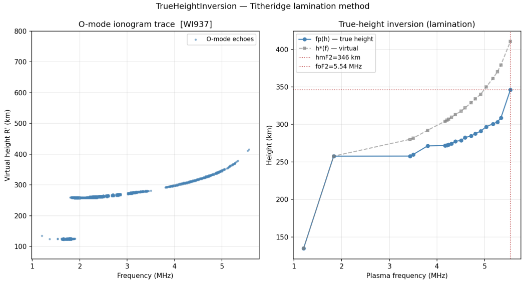
O/X Mode Separation

```python
from pynasonde.vipir.analysis \
    import PolarizationClassifier

clf = PolarizationClassifier(
  o_mode_sign=-1,        # N.
      hemisphere
  pp_ambiguous_threshold_deg=20,
)
pol = clf.fit(filtered_df)
print(pol.summary())
# O=412  X=388  ambiguous=54

o_df = pol.o_mode_df()
```



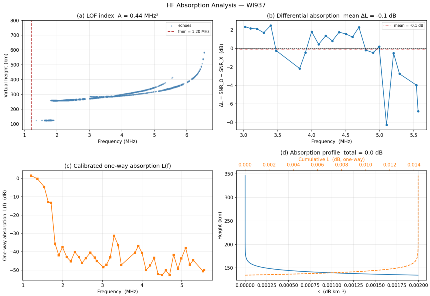
True Height Inversion → N(h)

```python
from pynasonde.vipir.analysis \
    import TrueHeightInversion

inv = TrueHeightInversion()
edp = inv.fit_from_df(o_df)

print(edp.summary())
# foF2=8.12 MHz
# hmF2=287.4 km
# NmF2=8.2e5 cm^-3

edp.plot()
```



HF Absorption

```python
from pynasonde.vipir.analysis \
    import AbsorptionAnalyzer
import numpy as np

aa = AbsorptionAnalyzer()
# 1. LOF index (no calibration)
lof = aa.lof_absorption(df)
# 2. Differential O/X
diff = aa.differential_absorption(
    pol.annotated_df)
# 3. Height-resolved profile
prof = aa.absorption_profile(
  edp,
  nu_hz=lambda h: 1e6*np.exp(-h/8))
```



Sporadic-E — Why Higher Resolution?

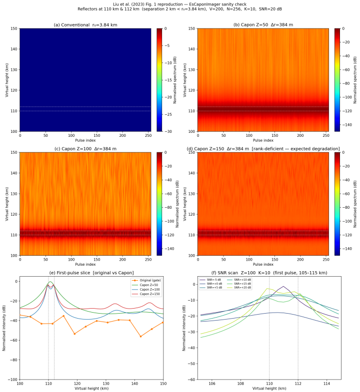
- Es: thin metallic-ion layer at 90–130 km ($<$1 km thick)
- VIPIR native gate $r_0 = 1.499$ km — **cannot resolve thin layers**
- **Capon cross-spectrum imaging** (Liu et al. 2023): **10× finer resolution**, no extra hardware
- Key: FFT of range profile as spatial frequency array; minimum-variance beamforming in *range space*

Effective resolution

$$\Delta r = \frac{r_0}{K}$$

Instrument	r_0	Δr (K=10)
WISS	3.84 km	384 m
VIPIR	1.499 km	150 m

Es Imaging Sanity Check — Liu et al. Fig. 1

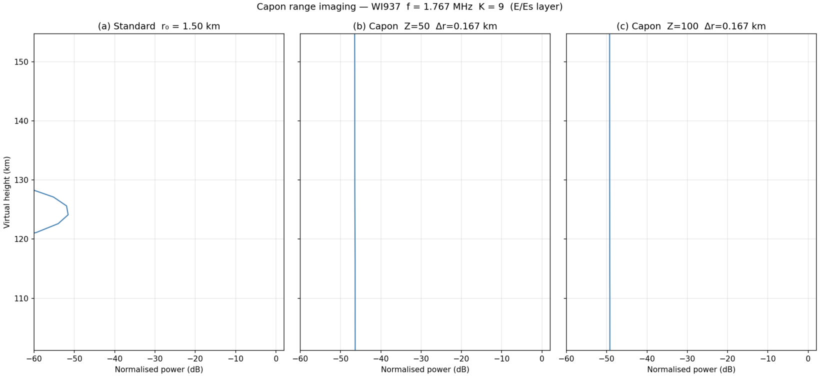


Synthetic two-layer test (110 + 112 km). $Z = 50$: unresolved. $Z = 100$: peaks at 109.8 & 111.7 km (<0.3 km error). $Z = 150$: rank-deficient (matches paper Fig. 1d).

Es Imaging — Single File

```python
from pynasonde.vipir.analysis import EsCaponImager

imager = EsCaponImager(
    n_subbands=100,           # Z  (Z <= (V+1)/2 recommended)
    resolution_factor=10,     # K  (no singularity constraint)
    gate_spacing_km=1.499,    # VIPIR native gate r0
    gate_start_km=90.0,
)
result = imager.fit(iq_cube)   # (pulses, gates) or (pulses, gates, rx)
print(result.summary())
# EsImagingResult: Z=100  K=10  r0=1.499 km -> Dr=0.150 km
result.plot(snapshot=0)
```

Es Imaging — Single File Result



Es Imaging — Multi-File A+B+C

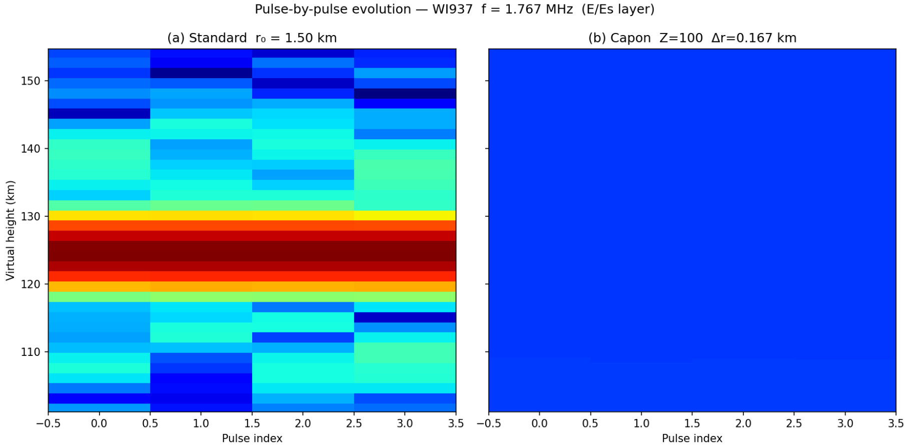
When only 4 pulses/file: combine 8 files $\times$ 4 pulses $\times$ 8 Rx $=$ 256 samples:

```
from pynasonde.vipir.analysis import RiqAggregator
import glob

files = sorted(glob.glob("data/20230601_05??.RIQ"))[:8]

agg = RiqAggregator(
    n_subbands=100, resolution_factor=10,
    output_mode="slow_rti",    # one column per file
)
result = agg.fit(files, freq_target_khz=5000.0)
result.plot()
```

SNR improvement

Option A (8 Rx coherent): +9 dB — Option B (4-pulse avg): $\div$4 var — Option C (8-file stack): $\div$8 var $\Rightarrow$ $\sim$24 dB total

Es Imaging — Slow RTI (A+B+C)



Slow RTI: one column per file (∼1 min cadence). Each column is a Capon pseudospectrum averaged over Option A+B.

Hands-On Exercises

DIGISONDE

1. Load a `.SAO` file. Print foF2 and hmF2.
2. Build a daily isodensity contour from multiple SAO files.
3. Load a `.DVL` file and plot the Vz time series.
4. Load a `.DFT` file and produce a Doppler waterfall.

VIPIR

5. Load a WI937 RIQ file; print gate count and frequency range.
6. Run `IonogramFilter`. How many echoes survive?
7. Separate O/X modes. What fraction is O-mode?
8. Run `TrueHeightInversion`. What is foF2?
9. Run `EsCaponImager` with $Z = 100$, $K = 10$. Is there a thin Es layer?
10. Compute the LOF absorption index.

Summary — I/O Layer (1/2)

DIGISONDE I/O — 8 file formats

- SaoExtractor — SAO 4.3/5.0: N(h) + scaled params
- RsfExtractor — direction-coded ionogram + directogram
- SbfExtractor — SBF binary raw soundings
- DftExtractor — Doppler waterfall + spectra
- DvlExtractor — drift velocities Vx, Vy, Vz
- SkyExtractor — polar sky maps (az/zen)
- EdpExtractor — electron density profiles
- SaoSummaryPlots, SkySummaryPlots

VIPIR I/O — 3 file formats

- RiqDataset — SCT + PCT loader, VIPIR_VERSION_MAP
- EchoExtractor — 7-param echoes (h, A, V*, EP, PP, XL, YL)
- IonogramFilter — 6-stage: RFI, EP, multi-hop, DBSCAN, RANSAC, temporal
- DataSource — NGI/SDR NetCDF loader + data cube
- AutoScaler — noise profile, segment, binary trace
- FullPipeline — end-to-end RIQ → N(h) automation

Summary — Analysis Layer (2/2)

Analysis layer — 9 classes (all post-v1.0)

- `PolarizationClassifier` — O/X via PP chirality
- `TrueHeightInversion` — Titheridge/POLAN $\rightarrow$ N(h)
- `AbsorptionAnalyzer` — LOF / O-X diff / $\kappa(z)$ profile
- `SpreadFAnalyzer` — range / freq / mixed classification
- `IrregularityAnalyzer` — $\alpha(h)$, outer scale, anisotropy
- `IonogramScaler` — foF2, foE, h'F, MUF
- `EsCaponImager` — Capon imaging ($\Delta r = 150$ m)
- `RiqAggregator` — A+B+C multi-file ($\sim$24 dB SNR)
- `NeXtYZInverter` — 3-D WSI + Hamiltonian ray-tracing

pynasonde.readthedocs.io
`pip install pynasonde`

References I

- Liu, T., Yang, G., & Jiang, C. (2023). High-resolution sporadic E layer observation. *Space Weather*, 21, e2022SW003195.
- Titheridge, J. E. (1967). A new method for the analysis of ionospheric $h'(f)$ records. *JATP*, 29, 763–778.
- Zabotin, N. A., Wright, J. W., & Zhbankov, G. A. (2006). NeXtYZ. *Radio Science*, 41, RS6S32.
- Grubb, R. N., et al. (2011). New science with the new VIPIR ionosonde. *Radio Science*.
- Reinisch, B. W., et al. (1997). The Digisonde 4D. *JASTP*, 59, 2181–2204.
- Davies, K. (1990). *Ionospheric Radio*. Peter Peregrinus.