ExtractRINKO (Python Port)

This repo ports the ExtractRINKO Fortran workflow to Python and adds quality-of-life improvements (validation, clearer I/O, chunked output, configurable resampling, and reproducible processing).

Goals

- Reproduce ExtractRINKO results from the Nortek Vector + RINKO EC sensor pipeline.
- Provide a clean, well-tested Python implementation.
- Make inputs/outputs explicit and easier to validate.

Status

Early setup: repo scaffolding and documentation.

Project layout (planned)

- `src/`: Python implementation
- `docs/`: reference documents (ignored by git)
- `sample_data/`: example inputs/outputs (external)

What this does

`src/vector_formatter.py` processes a converted Nortek Vector data file (text, 18 columns) plus a configuration INI and writes calibrated, resampled output suitable for Aquatic Eddy Covariance (AEC) analysis. It:

- Converts velocities to cm/s, pressure to mbar (with offset), and converts RINKO counts to temperature and oxygen.
- Applies air-pressure correction (via elevation or measured atmospheric pressure).
- Applies optional time-based and concentration-based calibration factors to O2.
- Downscales data to a target frequency by averaging, downsampling, or interpolation.
- Splits output into 30-minute chunks with timestamped filenames.

Similarities vs ExtractRINKO

- Uses the same core formulas as the Fortran version for temperature, oxygen saturation, density, and oxygen concentration.
- Supports the same calibration inputs and air-pressure correction options.
- Matches ExtractRINKO outputs (within rounding) when configured equivalently.

Differences vs ExtractRINKO

- Configuration is **INI-based** rather than the legacy `.def` format.
- Output is **CSV** with labeled header and unit row.
- Output columns are **fixed** (time, U/V/W, PW, TW, O2, O2_SAT) rather than selectable.
- Output is **chunked** into 30-minute files named with a start-timestamp.
- Resampling is **opt-in** and configurable (average, downsample, interpolate).

Configuration

Configuration uses `vector_formatter.ini`:

- `[paths]` includes `input_file` and `output_id` (default `reformatted`).
- `[sampling]` includes `hour_start_input`, `hour_start_output`, `hour_end_output`, `input_frequency`, `output_frequency`, `resampling_method`, and optional `start_date`/`start_time` for timestamped filenames.
- `[sampling]` also includes `hour_end_output`, which stops processing once the output time exceeds this value.
- When `start_date` and `start_time` are provided (e.g., `2025-02-10` and `15:00`), each output file name uses the chunk start time formatted as `YYYYMMDDTHHMM`, and the TIMESTAMP column is written as `yyyymmddTHHMM.sssssss`.
- `[environment]` includes `pressure_reference`, `pressure_offset`, `salinity`, `elevation`, `atmospheric_pressure`.
- Calibration coefficients are explicit per-coefficient keys under:
  - `[temperature_calibration_coefficients]` (`AT`, `BT`, `CT`, `DT`)
  - `[o2_calibration_coefficients]` (`AO2`…`HO2`)
- Calibration factors live under `[calibration_time]` and `[calibration_concentration]`.

def_to_ini tool

`src/def_to_ini.py` converts legacy ExtractRINKO `.def` files to the new INI schema:

- Reads the `.def` format (including two/three time-range lines).
- Writes an INI with explicit coefficient fields and `output_id`.
- Output defaults to the same path with a `.ini` extension.

Quick start

Convert a legacy `.def` file to INI:

```
python src/def_to_ini.py
C:\Users\geryatejina\OneDrive\dev\aec_project\sample_data\sample.def
```

Basic usage (default = no resampling):

```
python src/vector_formatter.py
C:\Users\geryatejina\OneDrive\dev\aec_project\sample_data\sample.ini
```

Enable resampling (downsample/average/interpolate):

```
python src/vector_formatter.py
C:\Users\geryatejina\OneDrive\dev\aec_project\sample_data\sample.ini --enable-
resample --resample average --target-hz 16
```

References

See `DETAILS.md` for a concise breakdown of inputs/outputs and processing steps.