User Manual: Harmonic Tide Prediction

Script name: `tide.py` (*Python 3*)

Abstract

This manual explains how to run, and interpret the outputs of a non-profissional harmonic prediction tool for sea level that generates a short animation. The program reads an hourly time series in CSV format (`year,month,day,hour,level_mm`), converts levels from millimeters to meters, splits the record into two halves (training and test), fits a tidal harmonic model by least squares on the first half, and, on the second half, animates sliding 7-day windows comparing observed vs. predicted levels. It also computes and plots 4-week moving averages over the full period and saves an animation as MP4 via `ffmpeg`.

This software is not for profissional use. It was created using vibe programming to illustrate how harmonic prediction can be done using standard libraries in Python.

Contents

1 Overview

The software implements the harmonic model

$$y(t) = \beta_0 + \sum_{k=1}^{K} \left(A_k \cos(\omega_k t) + B_k \sin(\omega_k t)\right),$$

where $y(t)$ is sea level (in meters) relative to a mean offset, ω_k are angular frequencies in rad/s derived from standard speeds in degrees per hour, and (β_0, A_k, B_k) are estimated by least squares. The frequency set follows the NOS/CO-OPS *standard suite* (about 37 constituents including M2, S2, N2, K1, O1, etc.), together with shallow-water and long-period terms as encoded in `SPEEDS_DPH`.

This software is not for profissional use. It was created using vibe programming to illustrate how harmonic prediction can be done using standard libraries in Python.

Do not use for navigation or operational decisions.

2 Requirements

- **Python 3.8+**
- Python packages: `numpy`, `pandas`, `matplotlib`
- **ffmpeg** installed and on your `PATH` (for saving the animation)

3 Input data (CSV)

3.1 Format

The CSV must have **five columns without a header**:

```
year, month, day, hour, level_mm
```

- `year`, `month`, `day`, `hour` are integers in UTC.
- `level_mm` is an integer in millimeters; the program converts it to meters.
- Missing values are flagged by `-32767` mm (i.e., −32.767 m) and are **removed**.

The data available in the folder ./data/ for the cities Honolulu, Fortaleza and Salvador were obtained for University of Hawai'i Sea Level Center (UHSLC) https://uhslc.soest.hawaii.edu/data/ and the license described there for this data applies. If you use UHSLC tide gauge data in your research or applications, please cite the dataset as:

Caldwell, P. C., M. A. Merrifield, P. R. Thompson (2015), Sea level measured by tide gauges from global oceans — the Joint Archive for Sea Level holdings (NCEI Accession 0019568), Version 5.5, NOAA National Centers for Environmental Information, Dataset, doi:10.7289/V5V40S7W.

3.2 File location

- If `-csv_file` is **not** provided, the program lists `.csv` files under `./data/` and prompts you to choose one.
- If `-csv_file` is provided, the path must exist; otherwise execution aborts with an error.

4 Quick start

Minimal example 1 (Interative):

```
python tide.py
```

Minimal example 2:

```
python tide.py --csv_file ./data/Honolulu.csv --initial_year 1920 --
    final_year 2000 \
  --week_seed 42 --animation_weeks 4 --local "Honolulu" --
    animation_speed 1.0
```

On launch, the program:

1. Reads and sorts the time series; reports available year range.
2. If `-initial_year`/`-final_year` are missing, prompts for years interactively.
3. Removes missing values (`-32767`) and checks that at least 100 samples remain.
4. Splits data into two halves: training (first half) and test (second half).
5. Subtracts the *offset* (training mean) from both halves.
6. Fits harmonic coefficients by least squares on the training set.
7. Predicts over the test period and starts a 7-day sliding-window animation.
8. Computes & plots 4-week moving averages and saves a PNG.
9. Saves an MP4 animation using `ffmpeg`.

5 Command-line options

Option	Type / Default	Description
`-csv_file`	`str` / (none)	Path to the CSV file. If omitted, the program scans `./data/` and asks interactively.
`-initial_year`	`int` / (available minimum)	Start year of the time slice. Validated not to be before the earliest year. If both years are omitted, selection is interactive.
`-final_year`	`int` / (available maximum)	End year of the time slice. Validated not to exceed the latest year and to be $\geq$ start year.
`-week_seed`	`int` / 42	RNG seed for the initial window position in the test half. Ensures reproducible starting week.
`-animation_weeks`	`int` / 4	Animation duration in *weeks* (limits the number of frames). Increase to cover more of the test period.
`-local`	`str` / ""	Station/location name shown in titles and used to compose the video filename.
`-animation_speed`	`float` / 1.0	Speed multiplier for the animation (affects the per-frame step in samples). Larger values advance faster. PS: use 1.0 (there is a bug here)

Practical examples.

1. Interactive CSV and year selection:

```
python tide.py
```

2. Years 1998–2015, 12-week animation, 1x speed:

```
python tide.py --csv_file ./data/Honolulu.csv --initial_year 1998 --
    final_year 2015 \
  --animation_weeks 12 --animation_speed 1.0 --local "Honolulu"
```

3. Reproduce the same initial week with a fixed seed:

```
python tide.py --csv_file ./data/Fortaleza-Brazil.csv --week_seed 7
```

6 Harmonic model and fitting

6.1 Constituent set

The script defines a dictionary of speeds in degrees per hour (`SPEEDS_DPH`) with ≈ 37 standard constituents (M2, S2, N2, K2, K1, O1, P1, Q1, ...), shallow-water harmonics (M4, MS4, MN4, M6, M8), long-period terms (SA, SSA, MF, MM, MSF), and some additional components. You may edit this list directly to tailor the model.

6.2 Frequencies

Each speed v (deg/h) is converted to a period T in hours via $T = 360/v$. The angular frequency in rad/s is

$$\omega = \frac{2\pi}{T \cdot 3600}.$$

Time is referenced to the first training timestamp t_0 using $t = (\text{timestamp} - t_0)$ in seconds.

6.3 Design matrix and least squares

The design matrix contains a column of ones and, for each ω_k, the columns $\cos(\omega_k t)$ and $\sin(\omega_k t)$. Parameters θ are estimated by `numpy` least squares:

$$\theta = \arg\min_{\theta} \|X\theta - y\|_2^2.$$

The training mean (offset) is subtracted from both training and test so that predictions and observations share the same baseline in plots.

7 Animation: sliding 7-day window

7.1 Windowing and frames

- The *test* half is traversed with 7-day windows.
- The number of samples per window is derived from the data timestep Δt.
- The initial position is drawn using `-week_seed`.
- `-animation_weeks` limits the total number of frames.
- `-animation_speed` scales the per-frame shift (in samples).

7.2 Visual style

The script uses a dark theme (`plt.style.use('dark_background')`), solid lines for observations and dashed lines for predictions, a subtle grid, and titles summarizing the total data period, current window, and number of constituents. The y-axis limits adapt to each window with a 10% margin.

7.3 Video output

The animation is built with `matplotlib.animation.FuncAnimation` and saved via `ffmpeg`. Default filename rules:

- If `-local` is provided: `{local_lower}_weekly_average.mp4`.
- Otherwise: `{csv_basename}_weekly_average_{YYYYMMDD_HHMM}.mp4`.

8 Four-week moving averages

After the animation, the program computes 4-week averages (with a 1-week stride for time markers) across the *entire* record, using the same offset removed during training. It also fits a linear trend in time (years) and displays the slope in m/year in the legend. The figure is saved as:

`{csv_basename}_average_level_4_weeks_{YYYYMMDD_HHMM}.png`.

9 Logging

Events are recorded to `tide.log`: full command line, chosen CSV, filtered period, record counts, number of 4-week averages, basic statistics (mean, std), and, if applicable, the trend slope.

10 Key validations and messages

- **Missing CSV**: execution stops with an error.
- **Invalid year range**: start $<$ available minimum, end $>$ available maximum, or start $>$ end $\rightarrow$ error.
- **After missing-value filtering**: if fewer than 100 records remain, execution stops with an error.

11 Customization and extensions

11.1 Change constituents

Edit `SPEEDS_DPH` to add/remove constituents. Closely spaced frequencies can cause ill-conditioning; consider regularization or selecting a subset.

11.2 Window length

To animate windows other than 7 days, change `win_days` in the code and recompute `win_samples` accordingly.

11.3 Styling

Colors, line widths, and fonts can be adjusted where `line_r`, `line_p`, axes, and legends are defined.

11.4 GIF export

Switching the writer to `PillowWriter` allows saving GIFs, though MP4 via `ffmpeg` usually yields better quality and size.

12 Best practices and reproducibility

- Fix `-week_seed` to reproduce the same starting window.
- Record the exact command line (also logged in `tide.log`).

13 Troubleshooting (FAQ)

`Error: File '...' not found.`
Check the `-csv_file` path or place the CSV under `./data/`.

`No CSV files found during interactive run`
Create `data/` and move your CSV(s) there.

`Insufficient data after filtering`
After removing missing values, fewer than 100 rows remain.

Video not saved / `ffmpeg` error
Ensure `ffmpeg` is installed and on your `PATH`.

Unreadable colors/labels
Adjust the theme colors or increase `fontsize` in the plotting section.

Quick reference

Listing 1: Common commands

```
# 1) Full run with labels and 8-week animation
python tide.py --csv_file ./data/Honolulu.csv --initial_year 1993 --
    final_year 2020 \
  --local "Fortaleza-Brazil" --animation_weeks 8 --animation_speed 1.0

# 2) Interactive run (no args): select CSV and years at the prompt
python tide.py

# 3) Faster animation and fixed starting week
python tide.py --csv_file data/Fortaleza-Brazil.csv --week_seed 123 --
    animation_speed 1.0
```

Credits and license

The data available in the folder ./data/ for the cites Honolulu, Fortaleza and Salvador were obtained for University of Hawai'i Sea Level Center (UHSLC) https://uhslc.soest.hawaii.edu/data/ and the license described there for this data applies. If you use UHSLC tide gauge data in your research or applications, please cite the dataset as:

Caldwell, P. C., M. A. Merrifield, P. R. Thompson (2015), Sea level measured by tide gauges from global oceans — the Joint Archive for Sea Level holdings (NCEI Accession 0019568), Version 5.5, NOAA National Centers for Environmental Information, Dataset, doi:10.7289/V5V40S7W.

This manual was last updated on October 4, 2025.