Wavelet Animation Script User Manual
Daniel Smania

Version: `1.0` – Last updated: July 18, 2025

Contents

1 Introduction

This script produces a frame-by-frame reconstruction of a one-dimensional function using discrete wavelet coefficients.[1] It is aimed at educators and researchers who wish to visualise how individual wavelets build up a signal.

Key features:

- Generates random functions of several types (smooth, discontinuous, etc.).
- Supports a wide range of discrete wavelets from PyWavelets.
- Saves high-quality GIF or MP4 animations (optional).
- Logs every execution for reproducibility (`wavelet_history.log`).

2 Prerequisites

- **Python** ≥ 3.8
- Packages: `numpy`, `matplotlib`, `pywt` (PyWavelets), `argparse` (standard), `pillow`.
- To export MP4 you also need FFmpeg in your `PATH`.

Install the scientific stack with:

```
pip install numpy matplotlib pywavelets pillow

# Optional for MP4 export
sudo apt-get install ffmpeg # Debian/Ubuntu
```

3 Quick Start

```
python wavelet.py --wavelet_type db4 --function_type smooth_periodic --save gif
```

This command animates the first 64 `db4` wavelets reconstructing a periodic smooth signal and saves a GIF in the working directory.

4 Command-Line Options

`-frames_per_wavelet` Frames allocated to each wavelet coefficient. Controls animation speed. Default: 12.

`-wavelet_type` Wavelet family used by PyWavelets. Supported: `db4`, `db6`, `db8`, `haar`, `bior2.2`, `bior4.4`, `coif2`, `coif4`, `sym4`, `sym8`, `dmey`. Default: `haar`.

`-function_type` Type of synthetic function: `smooth`, `piecewise_linear`, `discontinuous`, `smooth_periodic`, `mix`. Default: `smooth_periodic`.

[1] Source file: `wavelet.py`.

`-function_seed` Integer seed for reproducible randomness. Default: `38324`.

`-number_wavelets` Maximum number of coefficients to animate. Default: `64`.

`-save` Format for saving animation: `gif`, `mp4`, or `none`. Default: `none`.

5 Output Files

Animation

When `-save` is `gif` or `mp4`, the script writes a file named:

```
wavelet-<YYYY-MM-DD-HH-MM-SS>-<wavelet>_<func>_<N>w_<seed>s_<F>f.(gif|mp4)
```

Run History

Each command is logged to `wavelet_history.log`. Only the 100 most recent entries are kept.

6 Examples

1. Fast preview without saving

```
python wavelet.py --wavelet_type sym4 --frames_per_wavelet 5 --number_wavelets 32
```

2. High-quality MP4

```
python wavelet.py \
  --wavelet_type bior4.4 \
  --function_type discontinuous \
  --frames_per_wavelet 15 \
  --number_wavelets 120 \
  --save mp4
```

7 Workflow Internals

1. Generate input signal f using `random_function()`.
2. Compute wavelet decomposition via `pywt.wavedec`.
3. For each animation frame:
   (a) Build partial coefficient list with added wavelets.

(b) Reconstruct $f_{textpartial}$.

(c) Overlay orange fill (wavelet being added) and blue fill (cumulative).

4. Optionally export via Pillow (GIF) or FFmpeg (MP4).

8 Troubleshooting

Missing FFmpeg Ensure `ffmpeg` is installed and in your `PATH`.

Large GIF files Increase `frames_per_wavelet` or reduce `number_wavelets`.

Memory errors Reduce `dpi` or figure size in the script.

9 Extending the Script

- Add new function types by editing `random_function()`.
- Try other PyWavelets wavelet families.
- Replace generated signal with real data arrays.

License

MIT License – Feel free to adapt and redistribute with attribution.