Automated Parsing and Visualization of Surface-Wave Dispersion Data Using Python

Isaac Smith

PHY 591

Final Project Report

Abstract

Surface-wave dispersion analysis is a foundational step in near-surface seismic investigations, particularly for multichannel analysis of surface waves (MASW). In practice, dispersion picks are often generated using specialized software and exported in text-based formats that are not immediately suitable for further analysis or visualization. This project presents a Python-based workflow for automatically parsing, organizing, and visualizing dispersion data stored in .LST files generated by SurfSeis/Multimodes-style processing. The code imports dozens of dispersion files simultaneously, extracts frequency–velocity picks along with quality weights and mode numbers, exports clean and reusable CSV files, and produces publication-quality dispersion plots using spline-interpolated curves. While no inversion is performed, this workflow addresses a critical preprocessing step required before quantitative surface-wave modeling and demonstrates a reproducible approach to handling real seismic datasets.

Introduction

Surface waves contain information about the elastic structure of the near subsurface through their frequency-dependent phase velocities. In MASW workflows, dispersion curves are typically identified by picking energy ridges in the frequency–velocity domain. These picks are then used as inputs to inversion algorithms that estimate shear-wave velocity as a function of depth.

However, the dispersion-picking step often produces output files that are difficult to analyze programmatically, limiting reproducibility and large-scale comparisons across surveys or processing runs.

The goal of this project is to develop a Python-based framework that converts raw dispersion-pick output files into readable, graphable, and reusable formats. Rather than performing a full inversion, this project focuses on automating the parsing, cleaning, visualization, and export of dispersion data. This scope emphasizes computational data handling and visualization, aligning with the core objectives of the course while reflecting realistic seismic data workflows.

Data Description

The input data consist of multiple .LST files produced by surface-wave processing software. Each .LST file contains a tabulated list of dispersion picks with the following information:

- Frequency (Hz)
- Rayleigh-wave phase velocity
- Pick weight (a measure of confidence or quality)
- Mode number

These files are text-based and include extensive header information, requiring careful parsing to isolate the numeric dispersion table. Multiple .LST files are analyzed together to allow comparison across different shot gathers, processing runs, or survey locations.

Methods

- **File Parsing and Data Extraction**

The Python code automatically scans a user-defined directory and identifies all .LST files. For each file, the program locates the start of the dispersion table by searching for the line containing

the header "data F. (Hz)". Numeric rows following this header are parsed using regular expressions and converted into NumPy arrays containing frequency, velocity, weight, and mode information.

Duplicate frequency entries, which can occur in some .LST outputs, are handled by averaging values at repeated frequencies to ensure stable curve visualization. All dispersion picks are sorted by frequency so that subsequent plotting operations behave correctly.

- **Data Export**

For transparency and reuse, the program exports two types of CSV files:

Per-file CSVs, containing dispersion data from individual .LST files.

A combined CSV, containing dispersion picks from all files with an additional filename column.

These outputs allow the processed data to be easily imported into other analysis tools or used in future inversion workflows.

- **Visualization and Curve Smoothing**

Dispersion curves are visualized using frequency on the horizontal axis and Rayleigh-wave velocity on the vertical axis. To avoid jagged, piecewise-linear connections between discrete picks, cubic spline interpolation is applied to generate smooth dispersion curves. The original picks are plotted simultaneously as points to preserve the raw data.

Three types of plots are generated:

- Combined dispersion curves for all files
- Dispersion curves separated by mode number
- Pick weight versus frequency plots for quality control

Spline smoothing is used only for visualization; no additional physical assumptions are introduced.

Results

- **Combined Dispersion Curves**

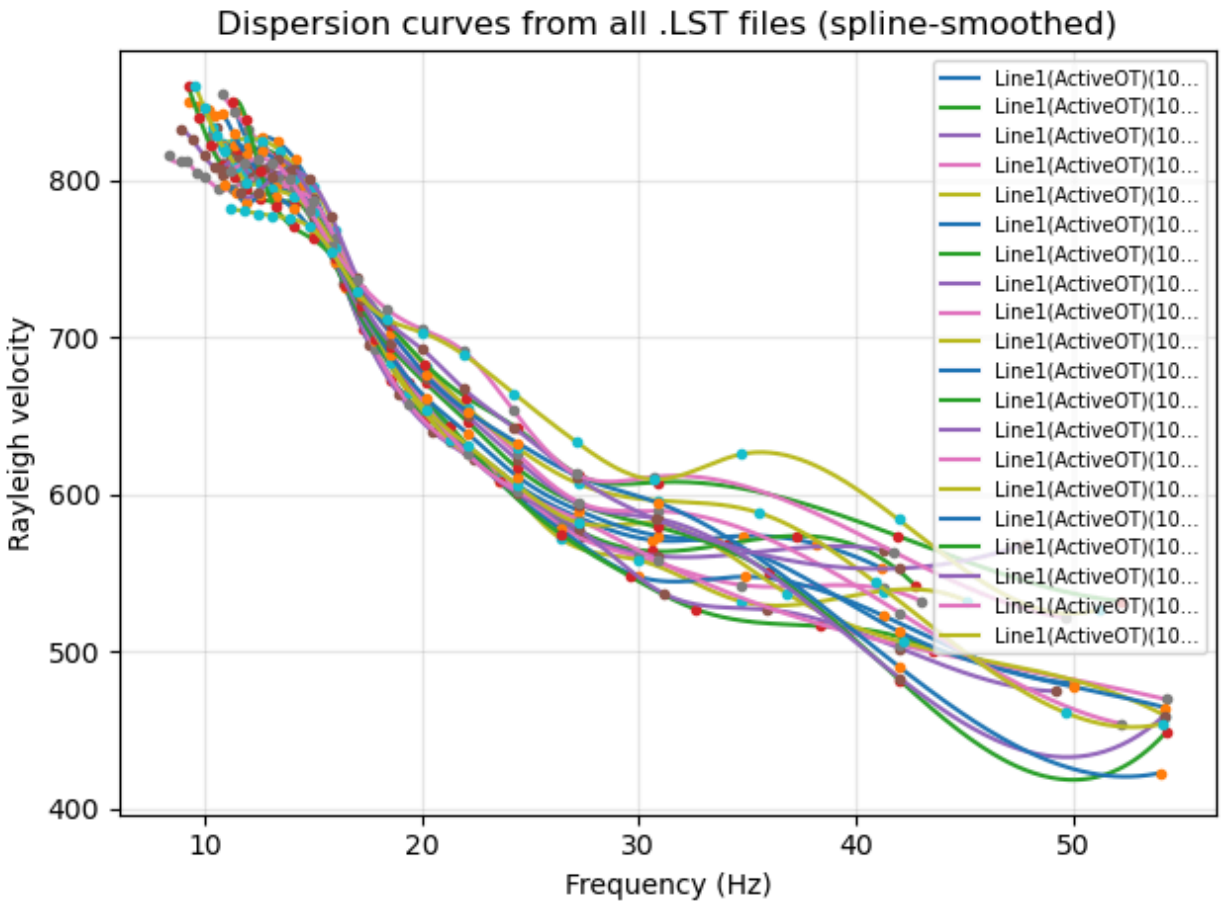


Figure 1 shows the dispersion curves extracted from all analyzed .LST files. The smooth trends illustrate consistent frequency-dependent velocity behavior, while the overlaid points indicate the original picked data. Differences between curves likely reflect variations in acquisition geometry, subsurface structure, or processing conditions.

- **Mode-Separated Dispersion Curves**

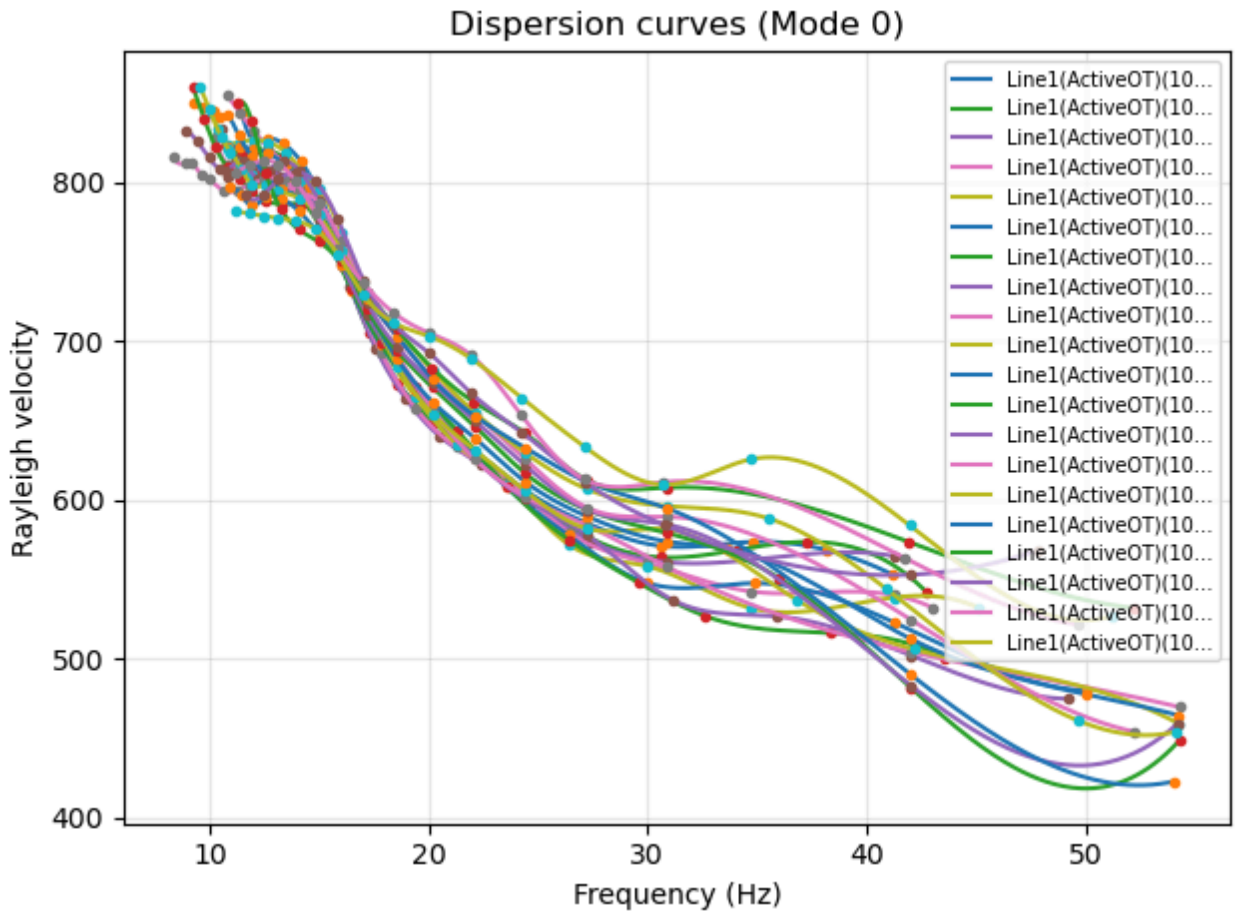


Figure 2 presents dispersion curves grouped by mode number. Separating modes clarifies the structure of the dispersion data and helps identify potential mode mixing or mispicks that would complicate inversion.

- **Pick Weight Analysis**

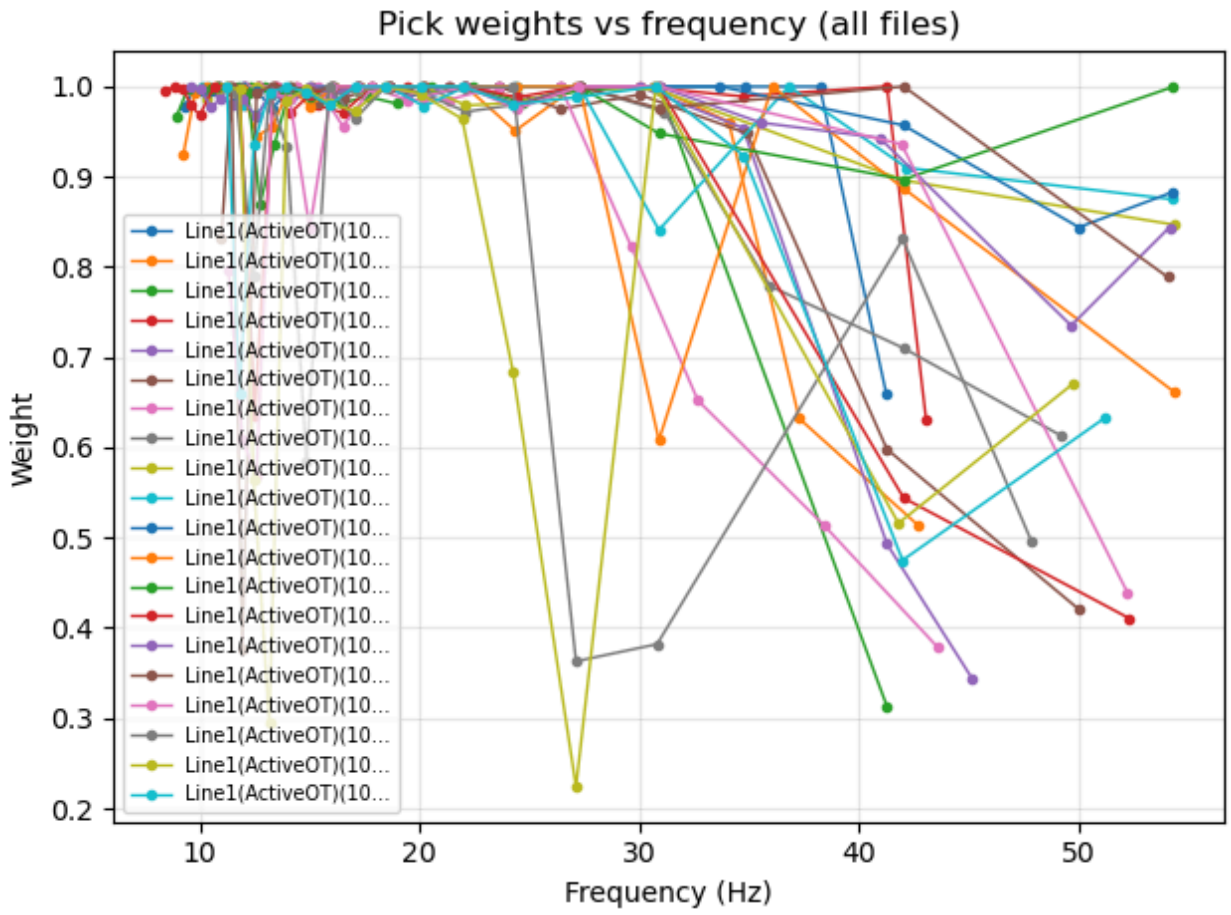


Figure 3 illustrates the distribution of pick weights as a function of frequency. Higher weights generally occur at intermediate frequencies, where surface-wave energy is strongest, while lower weights appear at frequency extremes. This plot provides a diagnostic tool for identifying unreliable portions of the dispersion curves.

Discussion

The results demonstrate that the Python workflow successfully converts raw dispersion-pick outputs into a clean, interpretable format suitable for further analysis. The use of spline interpolation improves the visual clarity of dispersion curves without obscuring the underlying data. The weight-versus-frequency plots add an important quality-control element that is often overlooked in simple visualization routines.

Although no inversion is performed, this project addresses a critical preprocessing step in surface-wave analysis. In practice, reliable dispersion parsing and visualization are prerequisites for any quantitative modeling. The modular design of the code allows it to be easily extended to include automated inversion or integration with external modeling software.

Limitations and Future Work

This project does not perform shear-wave velocity inversion and therefore does not produce a subsurface velocity model. Additionally, the dispersion picks are assumed to be valid outputs from prior processing, and no attempt is made to re-pick or modify them. Future extensions of this work could include automated filtering based on pick weight, uncertainty propagation into inversion results, or direct coupling with MASW inversion algorithms.

Conclusion

This project demonstrates a reproducible Python-based approach for parsing, visualizing, and exporting surface-wave dispersion data from .LST files. By focusing on data handling and visualization, the workflow fills a practical gap between proprietary processing software and quantitative modeling tools. The results show that dispersion data from multiple files can be efficiently analyzed and compared, providing a strong foundation for future inversion-based studies.