EFFSIMMSI Project Wrap-Up

Author: Danilo Dordevic

Date: August 2025

Affiliation: ETH Zürich - Department of Earth and Planetary Sciences / Swiss Seismological Service

This document summarizes the work conducted as part of the EFFSIMMSI project, with a focus on Distributed Acoustic Sensing (DAS) data. The notebook performs detailed analyses across multiple levels.

This wrap-up is intended as a reference for future researchers or collaborators continuing work on DAS signal processing, interpretation, and machine learning. The content is structured to be both reproducible and extendable.

Contents

1. People Involved

This is a summary of key academic and industry-related contacts relevant to the project at the beginning:

1.1. ETH Collaborators

- Doktorvater: Prof. Dr. Stefan Wiemer
- Principal Investigator: Dr. Peidong Shi, *expert in ML data preparation, model development, training, and transfer learning*
- Collaborators: Dr. Federica Lanza, *expert in DAS measurement and analysis, will provide inputs on DAS data preparation*

1.2. Geo-Energie Suisse AG

Geo-Energie Suisse AG is a Swiss company specializing in the development of deep geothermal energy projects. Their work spans subsurface exploration, geophysical monitoring, data acquisition, and the integration of innovative technologies such as Distributed Acoustic Sensing (DAS) into geothermal research. The company plays a key role in advancing sustainable energy solutions in Switzerland and Europe.

In the context of the **FORGE DAS recordings**, Geo-Energie Suisse was one of the primary contributors responsible for **collecting, managing, and pre-processing** the DAS datasets.

The following collaborators from Geo-Energie Suisse have been instrumental in the FORGE DAS efforts and remain key points of contact for project-related inquiries:

- **Dimitrios Karvounis,**

  📧 d.karvounis@geo-energie.ch

  Dimitrios has extensive experience in seismic monitoring and DAS data interpretation. He coordinated data logistics and interfaced with SED research teams for analysis support.

- **Ben Dyer,**

  📧 b.dyer@geo-energie.ch

  Ben contributed to the setup and operation of DAS hardware and was involved in the technical aspects of data handling, format conversion, and collaboration with external partners.

Their contributions have been critical in enabling downstream research using the FORGE DAS April 2024 dataset, particularly in tasks such as phase picking, and machine learning.

2. Project Introduction

Key milestones up until this point have been:

1. Gathering of the labeled and unlabeled DAS dataset, namely the FORGE April 2024 dataset
2. Development of the foundational model for DAS data

The key ideas of the project proposal, as well as the briefing on current progress, are summarized below:

2.1. EFFMMSI Proposal

The project proposal is the outline of the key points of the project. As part of the EFFSIMMSI project on induced earthquake forecasting, the PhD student contributed to the development of machine-learning-based tools for processing and analyzing Distributed Acoustic Sensing (DAS) data collected during hydraulic stimulations at the FORGE geothermal site.

The student's primary responsibilities are the **compilation, preprocessing, and organization of DAS datasets** to be integrated into the cross-scale WaveOcean database, which supports the training and evaluation of ML models for seismic event detection and phase picking.

This work was carried out in close collaboration with DAS expert Dr. Federica Lanza to ensure consistency with physical interpretations and data quality standards. In parallel with the PI's work on geophone, AE, and seismometer datasets, the student led the DAS component of the project and initiated the development of **DASNet**, a deep learning model specifically designed to process spatially coherent DAS recordings with high sampling rates and large channel counts.

The first milestone towards this goal is the compilation of a dataset which will be used to train this model. The current dataset covers the FORGE April 2024 DAS recordings, gathered from the GDR repository and with collaboration with Geo-Energie Suisse. Since the data is raw, some processing was needed to make the data appropriate for neural network training. The data is contained in a remote Petabyte storage, with each file representing a 12 second recording of microseismic activity. One of the utility functions that allows for easier analysis and extraction of relevant data is `slice_das_segment`, contained in the `utils.py` file, located on the github https://github.com/danilodjor/forge-downsampling. Given the location of the set of HDF5 file recordings, it takes as input start and end date and time, and extracts the exact measurements corresponding to the window between those two points in time. This can be useful in training of neural networks, where only noise or only signal can be extracted using the data from the catalog.

2.2. Foundational Model

The development of DASNet was grounded in **the foundational model approach**, motivated by recent advances in large-scale representation learning and inspired by techniques from large language model (LLM) training. The key insight driving this strategy was that DAS experiments collect **vast amounts of unlabeled wavefield data,** particularly during hydraulic stimulations, that remain largely untapped in the current literature.

By leveraging self-supervised learning techniques on this unlabeled data, the foundational model can acquire a general understanding of wave propagation patterns embedded in the DAS recordings, independent of any specific downstream task. This results in a flexible, scale-aware representation model capable of generalizing across different spatial resolutions, sampling rates, and recording geometries.

The next phase in the development of DASNet involves **fine-tuning of the learned representations on a smaller labeled dataset** for seismic phase picking, enabling high accuracy with limited labeled data while benefiting from the broad generalization capabilities of the pre-trained model. One of the key advantages of this approach is the development of the first foundation model for DAS data that can be used by researchers around the world for any other downstream task they need in a relatively straight-forward way, which includes just fine-tuning on a smaller set. This way, general wave propagation knowledge is extracted and exploited for concrete downstream tasks, yielding higher information utilization.

You may find the elaborated proposal for the foundational model here: link. It includes the proposal of the dataset, training methods, as well as the concrete architectures to be used in the project. It highlights the motivation behind choosing this data and this model.

The proposed method mentioned in the proposal includes the POSEIDON model, described in this paper. The key idea was to restructure the neural network architecture to model seismic wave development along a 1-D array rather than in full 3-D space, aligning with the linear geometry and spatial sampling of DAS systems. The main modifications would be adjusting the loss function to a different one, one which would support self-supervised learning on our task, such as reconstruction loss. Moreover, as the SwinTransformer blocks expect 2-D input data, one way would be to include as input to the network not just a single slice of the DAS recording, but multiple slices concatenated along the time dimension.

The code for this can be found on Euler, since it contains GPUs and allows for testing of neural networks. The path to the code is `/cluster/project/sed/ddordevic/poseidon` . You may clone the original github https://github.com/camlab-ethz/poseidon into your folder under `/cluster/project/sed/username/` .

3. Datasets

3.1. Links

https://gdr.openei.org/submissions/1680

https://utahforge.com/

3.2. About the Datasets

The current project utilizes DAS data from the **Utah FORGE GDR repository**, totaling over **90 TB** of continuous recordings. The main dataset spans from **April 7 to April 23, 2025**, recorded at a **10 kHz sampling rate** in **12-second windows**. This data has been fully downloaded and stored on the **Swiss Seismological Service's (SED) Petabyte storage system**, located at: `/bedrettolab/E1B/DAS/2024_FORGE/DATA_RAW_fromOpenei/April_2024/v1.0.0/`

To reduce storage requirements and to match the sampling rate of data provided by Geo-Energie Suisse (GES), the dataset has also been **downsampled to 4 kHz**. The downsampled version is available at: `/bedrettolab/E1B/DAS/2024_FORGE/DATA_RAW_fromOpenei/April_2024/v2.0.0/` .

The downsampled version can be accessed from any of the **Bigstar0x clusters**.

In addition, the dataset includes **Geo-Energie Suisse (GES) recordings**, which cover the period from **April 1 to April 6, 2025**, as well as the **triggered DAS data** collected throughout the experiment. These GES datasets are stored on **two dedicated NAS devices**, each with a capacity of 18 TB**.** These can be mounted onto the Petabyte storage, or the Euler. For mounting options, the best point of contact is Leandra Eberle.

The recordings are found in two datatypes: **HDF5**, and **SEGY**:

- To read and analyze the HDF5 files, one may use the following repo, which contains the scripts for reading, analyzing and downsampling the HDF5 data: https://github.com/danilodjor/forge-downsampling. The most relevant and comprehensive file that contains code for doing this is `spectral_analysis.ipynb` , which contains code snippets for reading the files, manipulating them (e.g. zooming in on a given time window, extracting only a certain channel, extracting metadata, and so on).
- To read the SEGY data, from the GES harddrives, one may use the code contained here: https://github.com/motionsignaltechnologies/reading-forge-DAS-segy-Apr2024

The specifications of the DAS recordings located in HDF5 files in Petabyte storage are:

- **Gauge Length:** 1.0210914611816406 meters
- **File Format:** HDF5 (h5)
- **Data Type & Shape:** Numpy arrays of size 120,000 × 1,496. First dimension is time, and second dimension is the DAS cable channel (there are 1496 channels).
- **Number Format:** float32

3.3. Data Analysis and Processing

Detailed statistical and spectral analysis of signals is performed. Relevant scripts are:

- `plot_events.ipynb`, which plots events from the catalog according to their location, and their moment magnitude
- `spectral_analysis.ipynb` , notebook performs a comprehensive spectral analysis of Distributed Acoustic Sensing (DAS) data recorded along an array of equidistant channels. Key functionalities include:
    - **Channel-wise spectral analysis:** Computes and visualizes frequency spectra for each channel along the DAS cable.
    - **Signal vs. noise comparison:** Analyzes and contrasts the spectral characteristics of noise segments with those of seismic signal segments.
    - **Phase-specific spectral comparison:** Separately evaluates the frequency content of P-wave, S-wave, and noise segments within

individual seismic recordings.

  - **Frequency-Wavenumber (F-K) analysis:** Conducts spatial-temporal spectral analysis across all channels simultaneously to capture wave propagation properties.
  - **Spectrogram generation:** Produces time-frequency spectrograms for selected channels to visualize signal evolution over time.
  - **Amplitude spectra visualization:** Presents one-dimensional amplitude spectra for all channels to assess their individual frequency responses.

  This notebook is structured to provide both detailed visualizations and comparative insights into the spectral properties of different seismic phases and noise, facilitating in-depth DAS data interpretation. It is also used to determine and justify the best frequency to downsample the data to.

- `visualization.ipynb`, which visualizes some signals and their 2-D Fourier trasnforms.
- `downsample.ipynb`, which validates the downsampling procedure, before applying it on the whole 90 TB dataset. This notebook compares different methods for downsampling HDF5 files containing DAS (Distributed Acoustic Sensing) data by a factor of 2 (e.g., from 10kHz to 5kHz sampling rate). The code implements and evaluates three downsampling approaches:

  1. **Method 1**: Basic resampling using scipy's `resample()` function directly on the HDF5 dataset
  2. **Method 2**: Concatenating three consecutive data patches, resampling the combined data, then extracting only the middle portion to reduce edge effects
  3. **Method 3**: Using DASCore's built-in `decimate()` function as a reference

  The notebook performs comprehensive validation through:

  - **Visual analysis**: Frequency domain plots using DFT to examine spectral content
  - **Error metrics**: Computing reconstruction errors and comparing frequency spectra
  - **Statistical testing**: Multiple statistical tests (t-tests, Kolmogorov-Smirnov, Mann-Whitney U, Wilcoxon, chi-square) to determine if the

downsampled signals maintain similar statistical properties to the original

The main finding appears to be that while all methods reduce file size as expected, they produce slightly different results in terms of spectral content and statistical properties, with Method 2 (concatenation approach) potentially providing better anti-aliasing by avoiding edge effects during the resampling process.

- `downsample.py` - This tool reduces the file size of DAS (Distributed Acoustic Sensing) HDF5 datasets by downsampling the temporal resolution while preserving signal quality. It processes collections of timestamped HDF5 files and reduces their storage footprint by a configurable factor (default: 2.5x reduction).

  ### *How It Works*

  The tool implements a sophisticated approach to avoid common artifacts that occur during downsampling. Instead of processing each file individually, it temporarily combines each target file with its chronological neighbors (the files immediately before and after it in time). This three-file concatenation is then downsampled as a unit, and only the middle portion - corresponding to the original target file - is kept. This approach eliminates edge effects that would otherwise degrade signal quality at file boundaries.

  The process handles the first and last files in a sequence specially, using only two files (since they lack a complete set of neighbors). All processed files have their metadata automatically updated to reflect the new sampling rate and time intervals.

  ### *Key Features*

  - **Edge handling**: Prevents signal artifacts by processing files with their temporal neighbors
  - **Configurable downsampling**: Adjustable up/down sampling ratios (default 2:5 for 2.5x reduction)
  - **Automatic metadata updates**: Correctly adjusts sampling rates and time intervals in HDF5 attributes
  - **Batch processing**: Processes entire directories automatically with progress tracking

    - **Space management**: Removes original files after successful processing to free up storage
    - **Robust file handling**: Sorts files chronologically and handles boundary cases appropriately

  ### ***Usage***

  The tool is designed for command-line operation and requires specifying source and target directories. Optional parameters control the processing range, sampling ratios, and logging behavior. Files are processed sequentially in chronological order based on their timestamps.

  ### ***Important Considerations***

  This is a **destructive operation** - original files are deleted after successful processing to manage disk space. Ensure adequate backups exist before running. The tool requires temporary storage space approximately equal to three times the size of the largest file being processed. Processing must maintain chronological order, so files cannot be processed independently or in parallel.

- `associate_catalog_dataset.py` : This tool links earthquake event records in CSV catalogs with corresponding DAS HDF5 data files from the Petabyte storage by matching timestamps. It identifies the most recent DAS file that was recorded before each earthquake event occurred (i.e. it identifies the exact DAS file that contains the event).

  ### ***How It Works***

  The script processes earthquake catalog CSV files that contain trigger dates and times for seismic events. For each earthquake record, it searches through a directory of timestamped HDF5 files to find the DAS file with the latest timestamp that still precedes the earthquake occurrence time. The tool handles timezone conversion (local time to UTC) and adds a new "Matched File" column to each catalog containing the filename of the corresponding DAS data file.

  ### ***Key Processing Steps***

  1. **Timestamp Extraction and Conversion**: Parses date/time information from CSV records, combines trigger date and time fields, and converts from local time to UTC by adding 6 hours offset.

2. **File Matching Algorithm**: For each earthquake event, scans all HDF5 filenames to extract their timestamps, then identifies the file with the most recent timestamp that is still earlier than or equal to the earthquake time.
3. **Data Quality Filtering**: Skips earthquake records that occur before April 7, 2024, and handles missing or invalid timestamp data gracefully.
4. **Batch Processing**: Automatically processes all FORGE catalog CSV files in a specified directory with progress tracking.

Key Features

- **Automated batch processing**: Handles multiple CSV catalog files in a single run
- **Timestamp matching logic**: Finds the most temporally relevant DAS file for each earthquake
- **Timezone handling**: Properly converts between local and UTC time zones
- **Non-destructive updates**: Adds new columns without modifying existing data
- **Resumable processing**: Checks for existing "Matched File" columns to avoid reprocessing

Usage

The tool requires specifying a directory containing FORGE earthquake catalog CSV files and a directory containing timestamped HDF5 DAS data files. It processes all qualifying files automatically and updates each CSV with the matched filenames, enabling downstream analysis that correlates seismic events with their corresponding DAS recordings.

Output

Each processed CSV file gains a new "Matched File" column containing the filename of the temporally closest preceding DAS data file, creating a direct link between earthquake events and their associated continuous waveform data for further analysis.

- `channel_interpolation/channel_interpolation.ipynb` - This tool processes well survey data to create precise 3D positioning for DAS (Distributed Acoustic Sensing) cable sensors and enables backprojection of earthquake events onto the cable for analysis. It interpolates irregular survey measurements to uniform sensor spacing and calculates travel times for seismic waves from earthquake locations to each sensor position. Elaborated in section 3.4. DAS Cable Channel Interpolation and Arrival Time Calculation.

How It Works

The script loads well trajectory survey data from Excel files, cleans and converts units from feet to meters, then performs spatial interpolation to generate uniform sensor positions along the cable path. Two interpolation methods are implemented: measured depth (MD) based interpolation and true 3D arc-length based interpolation. The tool creates detailed 3D visualizations comparing original survey points, interpolated sensor positions, and anchor locations.

Key Processing Steps

Survey Data Processing: Loads Excel survey files, cleans numeric data by removing commas and whitespace, converts measurements from feet to meters, and handles missing data points appropriately.

Spatial Interpolation: Uses cubic spline interpolation to generate uniform sensor positions with 1.021095-meter spacing along the cable path, supporting both measured depth and true arc-length based approaches for different accuracy requirements.

3D Visualization: Creates interactive 3D plots showing the cable trajectory with original survey points in red, interpolated sensor positions in blue, and anchor points in green, with depth displayed as increasing downward for geological convention.

Seismic Event Integration: Loads earthquake catalogs from multiple sources, filters for valid events, and prepares for backprojection analysis using P-wave and S-wave velocities to calculate travel times from earthquake locations to each DAS sensor.

Key Features

- **Dual interpolation methods**: Both measured depth and arc-length based interpolation for different use cases
- **Visualization**: 3D cable plots and 2D depth profiles showing easting/northing vs depth relationships

- **Travel time calculation**: Implements seismic wave propagation modeling for event-to-sensor timing
- **Flexible coordinate systems**: Handles UTM coordinates with proper unit conversions

Applications

The tool enables correlation of seismic events with DAS recordings by providing precise sensor locations and calculating expected arrival times for seismic waves. The uniform sensor positioning is essential for accurate seismic processing and interpretation of distributed acoustic sensing data.

Output Files

Generates CSV files with interpolated sensor positions, high-resolution visualization plots, and provides the foundation for seismic event backprojection analysis linking earthquake catalogs with their corresponding DAS waveform signatures.

3.4. DAS Cable Channel Interpolation and Arrival Time Calculation

This notebook `channel_interpolation/channel_interpolation.ipynb` implements spatial interpolation methods to accurately reconstruct the physical positions of Distributed Acoustic Sensing (DAS) cable channels at the Utah FORGE geothermal site. By estimating precise channel locations and computing seismic wave arrival times along the cable, the notebook improves spatial understanding of DAS data. These results serve as pseudo-labels for downstream seismic phase-picking and interpretation tasks.

3.4.1. What It Does

- Reads and processes well survey and reference positioning data related to the DAS cable installation.
- Performs spatial interpolation (cubic spline interpolation) to generate a dense, smooth estimate of channel coordinates in 3D space (Easting, Northing, Depth).
- Integrates well trajectory data and anchor points to improve spatial modeling accuracy.
- Calculates expected seismic wave arrival times at each DAS channel by backprojecting known event origin times, using planar wavefront

assumptions and actual event coordinates. This is done in the following way:

$$t_i^S = t_{origin} + \frac{\text{distance}(\mathbf{x_i}, \mathbf{x_{origin}})}{v_s}$$

$$t_i^P = t_{origin} + \frac{\text{distance}(\mathbf{x_i}, \mathbf{x_{origin}})}{v_p}$$

t_i^S represents the arrival time of the S wave at channel i

t_i^P represents the arrival time of the P wave at channel i

$\mathbf{x_i}$ is the location of the DAS channel i, given as a triplet (EASTING, NORTHING, TRUE VERTICAL DEPTH)

$\mathbf{x_{origin}}$ is the location of the origin of the event, given as a triplet (EASTING, NORTHING, TRUE VERTICAL DEPTH), taken from the catalogs (see code for concrete implementation).

- Visualizes the interpolated channel positions and arrival time distributions in 2D and 3D plots for data exploration and validation.

3.4.2. Inputs

- Well survey CSV files containing channel positions and depths.
- Anchor point CSV files serving as reference locations for interpolation.
- Known seismic event origin times and locations from the seismic catalog.
- DAS channel numbering and geometry metadata.

3.4.3. Outputs

- Interpolated channel positions with increased spatial resolution.
- Arrival time estimates for seismic events at each DAS channel.
- Visualizations of channel layouts and arrival time maps.
- CSV files or dataframes containing interpolated channel coordinates and timing data for subsequent processing.

3.4.4. Contribution & Relevance

This notebook enhances the seismic data analysis pipeline by providing precise spatial context to DAS recordings, critical for phase picking. By creating pseudo-labels for arrival times, it enables scalable machine learning model training on the vast DAS dataset without requiring exhaustive manual annotation.

4. Computing at ETH

ETH Zurich and the Swiss Seismological Service (SED) maintain substantial computing infrastructure through their respective clusters: **Euler** (ETH-wide) and **Bigstar** (SED-exclusive).

Euler is a high-performance computing cluster available to all ETH members and supports both CPU and GPU workloads. It operates on a job scheduling system using *Slurm*, requiring users to submit jobs to a queue for execution.

In contrast, **Bigstar** is a CPU-only cluster dedicated exclusively to SED users and does *not* use a job scheduler, allowing for more straightforward, interactive job execution. Bigstar is divided into seven sub-clusters, with users logging into one at a time. Among them, Bigstar07 is the newest and most capable machine, offering the best performance. For account creation, access permissions, and initial setup on Bigstar, please contact **Leandra Eberle**.

5. Miscellaneous

5.1. Github and Useful Links

https://github.com/danilodjor/forge-das-processing: Contains the code for FORGE April DAS HDF5 data analysis and downsampling, catalog-file association, as well as channel interpolation.

https://github.com/motionsignaltechnologies/reading-forge-DAS-segy-Apr2024: Contains the code for reading FORGE April DAS SEGY files.

https://polybox.ethz.ch/index.php/s/nCmig84F72kZJtY: Contains other materials related to the project.

5.2. Python Environments

Dascore:

This Python environment was set up to support advanced time series analysis and scientific computing, with a focus on seismic and Distributed Acoustic Sensing (DAS) data. It includes core geophysical libraries like *obspy* for seismic processing and *segyio* for working with SEG-Y files, along with *dascore* (installed from the latest GitHub commit) for handling and analyzing DAS data.

Standard data science tools such as *numpy*, *pandas*, *matplotlib*, and *scipy* are included for numerical analysis and visualization, while *tables*, *h5py*, and *blosc2* enable efficient I/O and storage of large datasets. The environment also includes support for cloud integration via AWS (*boto3*, *s3transfer*), and structured data handling through *pydantic*, *jsonschema*, and *SQLAlchemy*.

Overall, this environment is well-suited for research workflows that involve seismic data processing, DAS signal analysis, and interactive exploration in Jupyter.

Installation:

```
python -m venv ~/pyenv/dascore-env
source ~/pyenv/dascore-env/bin/activate
pip install -r requirements.txt
```