

CIVL 8126 / CERI- 8104- Data Analysis

Aftershock decay rates and power-law fits (50 points)

For this assignment, submit a short report with figures documenting your result! You will also have to submit your Python code with comments.

Required submission files:

- 1) Report (including the following figures: Prague seismicity map, aftershock rates, map of injection wells and earthquakes in OK, seismicity rate in Oklahoma)**
- 2) Python code**

The vast majority of earthquakes are aftershocks, i.e. earthquakes that are caused by stress perturbations of preceding events. Thus, the study of aftershocks is of key importance in seismology and provides a critical component for operational earthquake forecasts and short-term hazard assessments. Aftershocks are special in the sense that the average expected rate is well described by an empirical relationship called Omori's law:

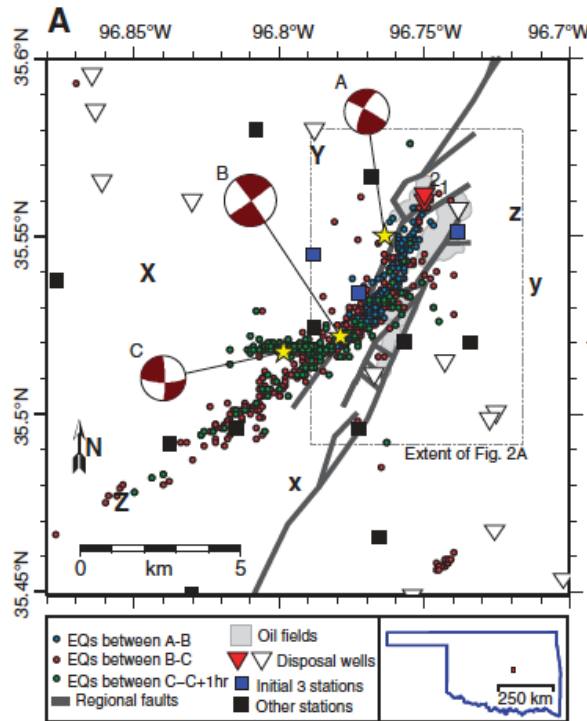
$$dN/dt = K/(c+t)^p$$

where dN/dt is the event rate after the mainshock, t is time after the mainshock, and K , c and p are constants describing the general productivity and decay rate of the aftershock sequence. This empirical relationship provides the opportunity to predict the average expected rate of earthquakes from hours to years after the mainshock.

Consider the record of seismic events within the central U.S., constrained by the following parameters:

Table 1:

Time range	MAG range	Longitude in °	Latitude in °
2010/01/01 to 2012/12/31	2.0 to 7	-100 to -95	34.3 to 37



Exemplary figure of the M5.7 Prague earthquake as well as fore and aftershocks in map view. The mainshock is highlighted by the yellow star and focal mechanism labeled by the letter B. Two additional M5 events are labeled A and C.

from: Keranen, K.M., Savage, H.M., Abers, G. a., and Cochran, E.S., 2013, Potentially induced earthquakes in Oklahoma, USA: Links between wastewater injection and the 2011 Mw 5.7 earthquake sequence: *Geology*, v. 41, no. 6, p. 699–702, doi: 10.1130/G34045.1.

Download:

1. Download the earthquake catalog from: <https://earthquake.usgs.gov/earthquakes/map/>, using the parameter specified in Table 1 (you can also go to: <https://earthquake.usgs.gov/earthquakes/search/>). You can either go to the website and input the parameters manually or you can use the python model `os.system` and the bash command 'wget'. You should end up with an ASCII data file that contains 364 events.

Data processing:

2. Write a script that imports the earthquake catalog! This is a nice exercise because you are dealing with somewhat complex, multivariate data that includes date, time, magnitude type and regular floating point numbers. You can focus on loading the first 6 columns (Date to Mag). Useful commands for this exercise are `numpy.loadtxt`, `numpy.genfromtxt` and standard ASCII file imports with `file_obj = open([file_name], 'r')`

Another way to load ASCII tables is with **Pandas**, a Python package that is designed to mimic sql-like queries of complex relational databases. Pandas finds much application in machine learning and data science and you should be familiar with the basic syntax, such as:

```
>>>import pandas as pd
>>>df = pd.read_csv([file], use_cols =..., delim_whitespace=True)
```

3. Select the data of interest:

- a. Look at the downloaded datafile and identify the M5.7 Prague main-shock in 2011
- b. **Spatial Filtering:** Select the area of interest using the following empirical scaling relation to identify the area of influence of a mainshock:

$$R_{\max} = 10^{(0.25M-0.22)} [\text{km}] *$$

There are several ways to filter the data to only include events within R_{\max} such as:

i) create an array of Boolean values (syntax is identical to MATLAB, use `numpy.logical_and` and `numpy.logical_or` for upper/lower bound conditions)

ii) use the `numpy.where()` function

*From: Gardner, J.K., and Knopoff, L., 1974, Is the sequence of earthquakes in southern California, with aftershocks removed, Poissonian? Bull. Seismol. Soc. Am, v. 64, no. 5, p. 1363–1367.

You can project the data from spherical into Cartesian or into an equal distance projection, e.g., using `matplotlib.basemap`. Alternatively, you can directly use the haversine formula to compute distances between your lon/lat vectors from the `seis_utils` module (https://en.wikipedia.org/wiki/Haversine_formula).

- c. **Time Filtering:** Select only events after the mainshock.
- d. Convert date-time to days relative to the mainshock occurrence. You can either convert all datetime to decimal years or use the `datetime` function. The time difference is then:

$$t_{\text{as}} = t_{\text{eq}} - t_{\text{MS}}, \text{ where } t_{\text{as}} \text{ are the aftershock times (in days) relative to the mainshock, } t_{\text{MS}}$$

Note that Pandas will automatically recognize DateTime strings and create a TimeDelta object when subtracting the respective origin times of all events from the mainshock origin time.

Aftershock rates and rate-decay fitting:

4. Compute the aftershock rate decay using the following equation for linear density estimates:

Linear Density (forward looking window):

$$\rho_i^{\text{eq}} = \frac{k}{t_{i+k} - t_i}; \text{ with } i = 0 \text{ to } (n - k) \quad **;$$

Centered:

$$\rho_{i+\frac{k}{2}}^{\text{eq}} = \frac{k}{t_{i+k} - t_i}; \text{ with } i = 0 \text{ to } (n - k);$$

where k is the sample size with higher values of k resulting in smoother density estimates, and $t_{i+k} - t_i$ is the distance between the i^{th} and the $i+k^{\text{th}}$ event. This is essentially a sliding sample window from your first to your $n-k^{\text{th}}$ event with a step size of 1.

**See: Silverman, B.W., 1986, DENSITY ESTIMATION FOR STATISTICS AND DATA ANALYSIS: Monographs on Statistics and Applied Probability, p. 1–22.

5. Compare the temporal decay from the above equation to a simple histogram of events after the mainshock! What are some advantages of using statistical density estimates vs. a binned representation of the data?
6. Fit the power-law portion of the time decay using:

$$dN/dt \sim t^p$$

Note that there are several steps involved in determining the fit, including: i) log-transformation of the data, ii) selecting the correct time range over which the data shows power-law behavior and iii) finally determining the fit in a least squares sense. You can use:

```
>>>import scipy.stats
>>>slope, inter, R2, p, err = scipy.stats.linregress(
np.log10(a_t), np.log10( a_rate) )
```

7. Plot the fit and the data on double logarithmic scales! Report the aftershock decay exponent, p , and the R2-value.
8. Discuss your results, observations and implications!