A statistical approach to predicting a tsunami's wave height

Can the height of a tsunami's wave be predicted based on the distance from and the magnitude of a preceding earthquake?

Word count: TBD

Contents

1 Introduction

1.1 Background

Naturally occurring disasters such as tsunamis are devastating for people and resources residing in risk areas. These tsunamis are caused by many different types of natural phenomena such as landslides, collapse of seamount, or more rarely, the impact of a meteorite. This paper however will focus on one of the more common cases of tsunami; one caused by the earthquakes. This is due to earthquakes causing around 72% of earthquakes [1], and as the time between an earthquake event and its tsunami is relatively high compared to events such as landslides [2], leaving adequate time for prediction and correct preparation of the events properties, such as the wave height, as discussed in this paper.

1.2 Personal Engagement

I have chosen to write about this subject as I have recently become interested by the statistics and its implications in real-life matters. This research has proved very interesting for me as it has abstracted the field of statistics for me. Furthermore, as I enjoy programming, the research I have done has helped me learn about the tools and technologies used by data analysts and other professionals of the matter.

2

3 Statistical Model

4 Uses of this research

5 Conclusion

Works Cited

[1] *What Causes a Tsunami?* URL: https://tsunami.org/what-causes-a-tsunami/.

[2] Langford P Sue, Roger I Nokes, and Roy A Walters. *Modelling of Tsunami Generated By Underwater Landslides*, p. 5. URL: https://www.eqc.govt.nz/sites/public_files/1596-Modelling-tsunami-generated-by-underwater-landslides.pdf.

A Appendix

A.1 Python program

qsd

```
import numpy as np
import pandas as pd
from matplotlib import pyplot as plt


def filter_time(df, threshold):
    filter_df = df.loc[df.year > threshold].copy()
    return filter_df
```

```
def filter_distance(df, threshold):
    filter_df = df.loc[df['distance from source (km)'] < threshold].copy()
    return filter_df


def filter_doubtful(df):
    filter_df = df.loc[df['doubtful runup'] != 'y'].copy()
    return filter_df


def calculate_mean(x):
    x['weight'] * x['maximum water height (m)']


def weighted_mean_height(df):
    return df.groupby('earthquake magnitude', as_index=True).apply(lambda x: (x['w


def get_droppable(df, whitelist):
    to_drop = []
    for (columnName, columnData) in df.iteritems():
        if columnName not in whitelist:
            to_drop.append(columnName)
    return to_drop


if __name__ == "__main__":
    df = pd.read_table("dataset/runups.tsv", sep='\t')
    df.columns = df.columns.str.lower()
```

```
# Filter data points before installation of
filter_df = filter_time(df, threshold=1960)

# No doubtful runups.
filter_df = filter_doubtful(filter_df)

# Drop all entries with no reported wave height
filter_df = filter_df.dropna(how='any', subset=['maximum water height (m)'])

# Filter waves originating from a certain distance away
# filter_df = filter_distance(filter_df, 30)

filter_df = filter_df.drop(get_droppable(filter_df,
                                         ['year', 'mo', 'earthquake magnitude',
                                          'tsunami event validity',
                                          'maximum water height (m)']), axis=1)

filter_df['weight'] = 1 - filter_df['distance from source (km)'] / filter_df['

filter_df = weighted_mean_height(filter_df)

filter_df.to_csv("test.csv")

print(filter_df)

plt.scatter(filter_df['earthquake magnitude'], filter_df['maximum water height
plt.show()
```