Weather Data Analysis

STAT 129 Big Data Project Report (Spring 2024)

Austin Melendez
California State University, Sacramento
Sacramento, CA, USA

ACM Reference Format:
Austin Melendez. 2024. Weather Data Analysis: STAT 129 Big Data Project Report (Spring 2024). In . ACM, New York, NY, USA, 6 pages.

1 Introduction

There are 106,200 stations from around the world containing five weather common measurements: precipitation total (PRCP), snowfall (SNOW), snow base depth (SNWD), max temperature (TMAX), min temperature (TMIN) for each day of the year. Other additional meteorological measurements are included for stations where the data is available. This dataset contains data from 1750 to 2023. This totals over 108 GB of data when uncompressed.

1.1 The Dataset

The NOAA Global Historical Climatology Network Daily (GHCN-Daily) dataset contains daily observations over global land areas. It contains station-based measurements from land-based stations worldwide. GHCN-Daily is a composite of climate records from numerous sources that were merged together and subjected to a common suite of quality assurance reviews. The directory is structured by year from 1750 to present, with each file named after the respective year.

2 Problem

Manipulate the dataset to retrieve the median of 'TMAX' for a specific subset of weather stations for all 365 days for each available year. Make a plot showing the average annual temperatures for 5 weather stations. Use parallel programming and a single pass through the data to make the data processing as fast as it can possibly be.

- The 5 weather stations with longest recorded data for 'TMAX'

2.1 Hypothesis

With global temperatures rising due to global warming, the maximum temperature recorded at different stations will trend upwards over time.

2.2 Approach Overview

I selected the 5 weather stations with the most overall data to analyze:

- **Selecting Stations**: Using zgrep, determine which station ID's have 'TMAX' data for the longest amount of time into the past.
- **Python Script**: A python script to obtain all 365 days of data for each given year then calculate the median temperature at a given station for each year.
- **Bash Script**: Finds all 'TMAX' records in each year file for each given station ID. Feeds data into python script to calculate median and saves result to CSV.
- **CSV Output**: The output of both scripts in [Station ID], [Year], [Median Temperature] format for easy graphing.

3 System/Algorithm Design

3.1 Data Preprocessing

The preprocessing of the data from the dataset is handled in the Shell script. We manipulate the data from it's original CSV format, removing unnecessary columns using zgrep and cut. The values we need for our calculations are sorted and saved in a text file for use in our other scripts.

3.2 System Architecture

The dataset for this project was stored on an external Sacramento State server for the Statistics Department. The directory is structured by year from 1750 to present, with each file named after the respective year. The data is available in CSV file format and as '.csv.gzip' files, so any particular year will be named 'yyyy.csv' and 'yyyy.csv.gz'.

3.3 Python Script

The Python script is used to calculate the median temperature at a specific station for each full year. The script first fetches the preprocessed data file from the beginning of our Bash script, each input file follows the naming convention 'out-stationid-year.txt'. After opening the file, a for loop is used to iterate through every line in the file. In this loop we make sure to catch any edge cases of blank strings or empty lines, ensuring they are handled appropriately. Otherwise, the current line is stripped and appended to an array called 'total', which holds a sorted array of every temperature value. A counter variable, 'count' is then also incremented to keep track of the total number of values in this specific file. Lastly, when all the temperatures have been read from the file, the overall median temperature is calculated.

Even total elements:

$$Median = total[count/2] + total[(count/2) + 1]/2$$

Odd total elements:

$$Median = total[count/2]$$

3.4 Bash Script

The Bash script first saves the station ID's that you want to work on as variables at the top of the file. Next, it sets up the formatting of the output CSV file by creating a blank file with only the

columns names defined. As previously mentioned, the script then preprocesses the data to extract only the 'TMAX' values for each station, these stripped values are stored in a temporary text file titled 'out-stationid-year.txt'. While calling the Python script we need to input 'stationid-year' as an argument so the Python file can find the correct temporary file generated earlier. The output of the Python script is then saved into another temporary text file titled 'median-stationid-year.txt'. The script then saves the calculated value generated in 'median-stationid-year.txt' for each station. This is outputted into the CSV file by using the echo command to place the station id, year, and median temperature into a single string line separated by commas, which is then appended to the CSV file. Lastly, after the final output is completed, all of the temporary files generated throughout the script are removed from memory.

3.5 CSV Output

The output file is structed as a CSV file with only three columns. The first column contains the station ID, the second column contains the year, and the third column contains the median temperature. After running my scripts on this dataset, my output file contained 1315 rows, meaning there are 1315 total data points between the five weather stations.

946	EZE00100082	1960	13.3
947	GM000010962	1954	9.75
948	USW00013994	1944	20.0
949	EZE00100082	1961	16.05
950	GM000010962	1955	9.8
951	USW00013994	1945	19.4
952	ITE00100554	1949	19.25
953	EZE00100082	1962	12.55
954	EZE00100082	1963	14.4

Figure 1: Subset of CSV Output

4 Evaluation

4.1 Methodology

My scripts worked efficiently and captured the data from the server that was requested for the median calculation. However, there is some erroneous data graphed. I suspect the extreme dips found at certain years along the graph correspond to missing or faulty data. This can be confirmed by checking the dataset for the value under the "quality" flag for those years. To clean-up this erroneous data Tableau was used to remove any outlier points and create an interactive data dashboard.

5 Conclusion

Historical data trends upward for all five weather stations in various parts of the world. This supports my initial hypothesis that the effects of global warming can be seen by plotting the median temperature, thus removing the influence of outliers, for every year. In our Tableau dashboard, you can also see the overall trend line has a positive slope, indicating a climb in overall temperatures over time. This can also be confirmed by looking at each station's individual trend line, again each trend line has a positive slope, indicating an increase in overall temperatures over time.

6 Learning Experience

In this project I learned how to manipulate large datasets, by connected through SSH, to obtain the exact data I want. To achieve this I created two scripts which can easily be edited to obtain different types of data other than 'TMAX' as well as any specific weather stations as long as you know the station ID. Additionally, I structured the scripts to ensure they could efficiently run using parallel programming to further reduce the runtime of the scripts. Since this dataset contained 108 GB of data when uncompressed, it was essential to reduce runtime as much as possible. By implementing these parallel programming techniques, I was able to reduce total runtime by 70 percent overall.

7 *Data Visualization*

7.1 Pandas Output

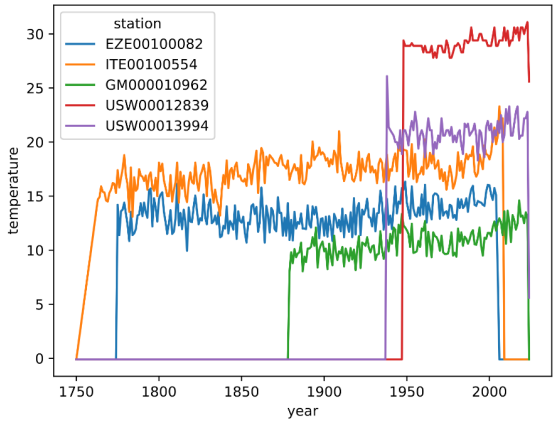

Figure 2: Median Temperature at Five Unique Stations

7.2 Tableau Output

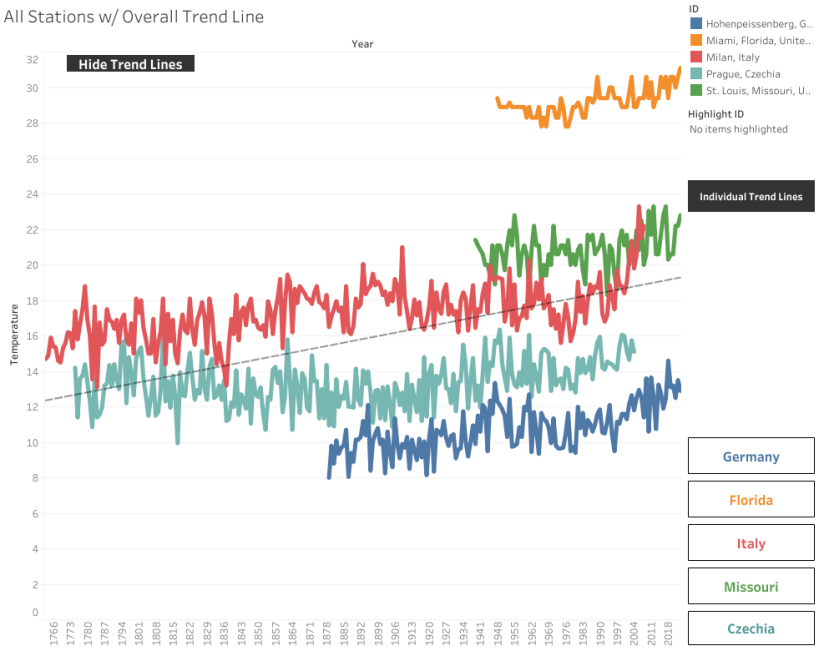

Figure 3: Overall Trend Line

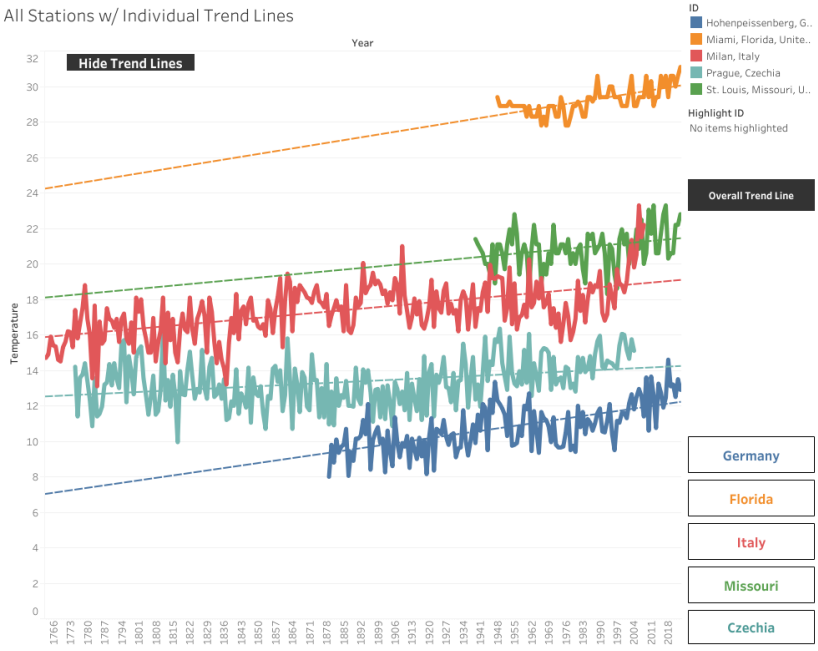

Figure 4: Individual Trend Lines

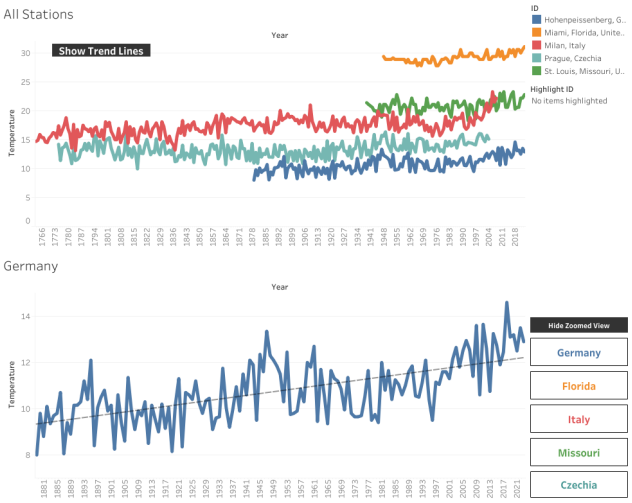

Figure 5: Zoomed Germany Graph

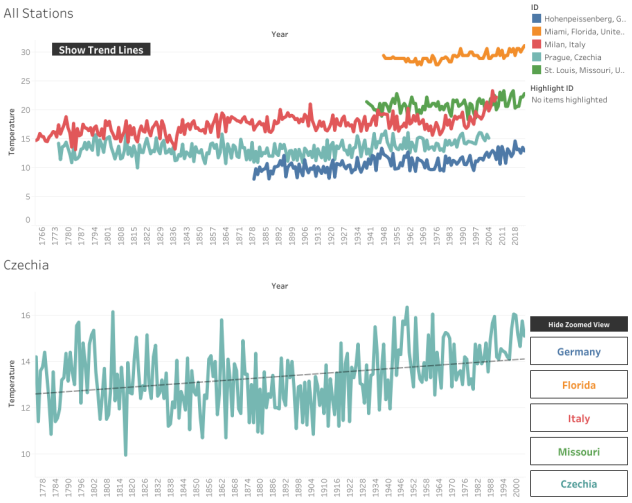

Figure 6: Zoomed Czechia Graph

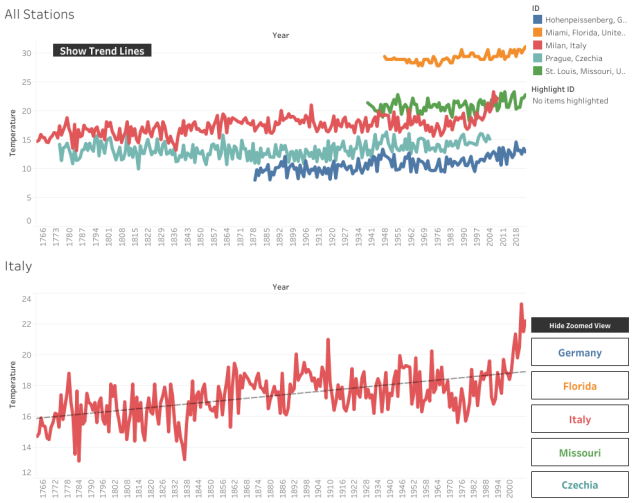

Figure 7: Zoomed Italy Graph

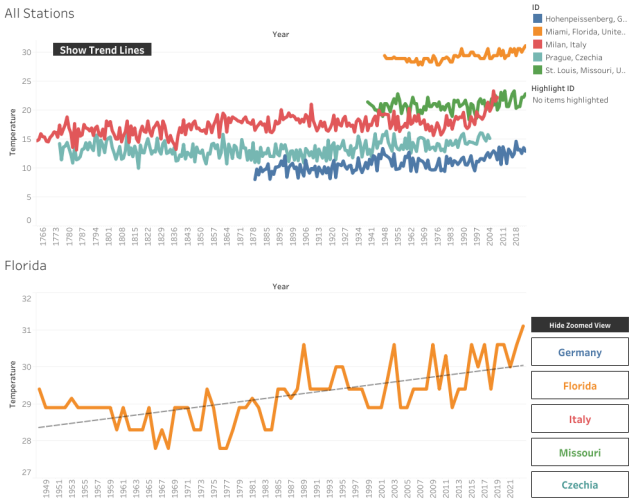

Figure 8: Zoomed Florida Graph

```python
#CALC MEDIAN
import sys
import subprocess

def temptocelcius(temp):
    celcius = temp/10
    return celcius

def adjustscale(data):
    pass

def calcmed(name):
    med = 0
    count = 0
    total = []

    filename = "out"+name+".txt"
    with open(filename,"r") as file:
        for i in file:
            line = file.readline()
            if line == "":
                break
            if line == "\n":
                break
            else:
                total.append(line.strip())
                count += 1

    #IF EVEN
    if(count != 0):
        if((count%2) == 0):
            med = ((int(total[int(count/2)]) + int(total[int(count/2)+1]))/2)
        else:
            med = (int(total[int(count/2)]))
    else:
        med = 0

if __name__ == "__main__":
    calcmed(sys.argv[1])
```

Figure 9: Python Script

```
#Save year from each file we are working on
year=${1:9:4}

#Station 1
station1="EZE00100082"
#Station 2
station2="UK000047811"
#Station 3
station3="ITE00100554"
#Station 4
station4="GM000004204"
#Station 5
station5="BE000006447"


#INITAL SETUP CSV FILE FOR GRAPHING LATER
#echo "station,year,temperature" > station-year-temp.csv

zgrep -e $station1 $1 | zgrep "TMAX" | cut -d, -f4 | sort -n > out$station1$year.txt
python3 -u calcmedian.py $station1$year > median$station1$year.txt
echo $station1","$year","$(<median$station1$year.txt) >> station-year-temp.csv

zgrep -e $station2 $1 | zgrep "TMAX" | cut -d, -f4 | sort -n > out$station2$year.txt
python3 -u calcmedian.py $station2$year > median$station2$year.txt
echo $station2","$year","$(<median$station2$year.txt) >> station-year-temp.csv

zgrep -e $station3 $1 | zgrep "TMAX" | cut -d, -f4 | sort -n > out$station3$year.txt
python3 -u calcmedian.py $station3$year > median$station3$year.txt
echo $station3","$year","$(<median$station3$year.txt) >> station-year-temp.csv

zgrep -e $station4 $1 | zgrep "TMAX" | cut -d, -f4 | sort -n > out$station4$year.txt
python3 -u calcmedian.py $station4$year > median$station4$year.txt
echo $station4","$year","$(<median$station4$year.txt) >> station-year-temp.csv

zgrep -e $station5 $1 | zgrep "TMAX" | cut -d, -f4 | sort -n > out$station5$year.txt
python3 -u calcmedian.py $station5$year > median$station5$year.txt
echo $station5","$year","$(<median$station5$year.txt) >> station-year-temp.csv

#REMOVE ALL TEMPORARY FILES
rm out$station1$year.txt
rm median$station1$year.txt
rm out$station2$year.txt
rm median$station2$year.txt
rm out$station3$year.txt
rm median$station3$year.txt
rm out$station4$year.txt
rm median$station4$year.txt
rm out$station5$year.txt
rm median$station5$year.txt
```

Figure 10: Bash Script