Do weather changes matter?

Marc Los Huertos

July 1, 2016

1 Introduction

According the the IPCC, the temperature has been changing about 0.X degrees per XX years – but how do these changes "map" onto a community that you care about? Can we find out how these changes will affect specific communities we care about? In other words, do weather changes matter?

1.1 Goals of this Document

1. Describe the goals and approach for the project;
2. Provide or point to resources to prepare for and conduct the project; and
3. Describe how we will evaluate the projects.

1.2 Learning Goals

For this project, you will use weather data to the question "do weather changes matter". How you answer the question is largely up to you, however, there are some learning goals associated with this project:

- Learn how to download and process weather data;
- Evaluate the trends in weather data;
- Determine the impact of weather in a human or non-human community; and
- Communicate your conclusions to the public.

Throughout this project, your team and instructor will develop the strategies and skills to address this question and help you make some conclusions and preset the results ot the public.

1.3 Driving Question

Projects can often be structured as questions, but sometimes it it worth phrasing the questions in a number of ways – this might help you find ways that you might find the question more provactive and interesting, For example,

- Is my region's climate changing?
- How is climate change affecting my community?

1.4 Public Product

Science is a social project. From the questions we ask, to the results and their presentation, science is embedded in a culture of norms. Thus, as part of this project, students will produce a narrative blog with the following characterics:

- appropriate and thoughful statistical analysis;
- Professionally appearing graphics; and
- provide interactive graphics as part of a public presentation.

1.5 Approach

Students will have the following tools available.

- NOAA website where data can be downloaded;
- R Studio Server with some scripts to help you develop analyses;
- Gighub to store project codes; and
- Shiny app templates that might be used as a container for interactive content.

2 Project Stages (i.e. Scafolding)

2.1 Day 1: How is temperature data collected?

2.1.1 Land Based

2.1.2 Marine Based

2.2 Day 2: How are the data store, curated and checked for quality?

3 Data 3: Data Sources

3.1 NOAA

3.2 Others

3.3 File Types and Software Tools

4 Using RStudio

4.1 Why R, Why Rstudio, and Why Open Source

Excel was not designed to handle large datasets, i.e. over 1 million rows. For most purposes, this might be enough. However, in many climate science data often exceed this number of samples.

5 R Coding an Analysis

For this code, I suggest the using the R base package plus some libraries for assorted specialized tools. When these are used, I can explain them, but for now, I suggest you make sure these files are 1) conveinient and 2) useful.

```
library(tidyr)
library(dplyr)

##
## Attaching package:  'dplyr'
##
## The following objects are masked from 'package:stats':
##
##     filter, lag
##
## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
library(stringr)
library(reshape2)
```

We will also use a customized function, which can be called automatically if you have the source code in your directory with the following:

```
source("summarySE.R")
```

Or you can download this file from http:... and run code to create the function manually.

5.1 Accessing the Data

First, we you may find you need to download the data...

Once you have downloaded it, the files will need to be pre-processed to be imported into R and/or post-process to create a useful dataset.

One preprocessing task might be to uncompress the data for example:

```
# Uncompress the files.
# ghcnd_all
```

```
stationfile = "/home/CAMPUS/mwl04747/github/Climate_Change_Narratives/Data/ghcnd-stations.tx
```

5.2 Read Station Data into R

```
# read.table(stationfile, header=F, fill=T, row.names=NULL); head(stations)
stations = (read.fwf(stationfile, fill=T, widths= c(11, 9, 10, 7, 3, 32, 3, 4, 9), ))
names(stations)= c("ID", "LAT", "LONG", "ELEV", "STATE", "NAME", "GSN", "HCN_CRN", "WHOID")

head(stations)
```

```
##            ID     LAT     LONG  ELEV STATE
## 1 ACW00011604 17.1167 -61.7833  10.1
## 2 ACW00011647 17.1333 -61.7833  19.2
## 3 AE000041196 25.3330  55.5170  34.0
## 4 AEM00041194 25.2550  55.3640  10.4
## 5 AEM00041217 24.4330  54.6510  26.8
## 6 AEM00041218 24.2620  55.6090 264.9
##                               NAME GSN HCN_CRN WHOID
## 1  ST JOHNS COOLIDGE FLD                          NA
## 2  ST JOHNS                                       NA
## 3  SHARJAH INTER. AIRP             GSN         41196
## 4  DUBAI INTL                                  41194
## 5  ABU DHABI INTL                              41217
## 6  AL AIN INTL                                 41218
```

Example of data:
AG000060680 22.8000 5.4331 1362.0 TAMANRASSET GSN 60680

5.3 Selecting and Example Location

Here's what the data look like:

ID 1-11 Character YEAR 12-15 Integer MONTH 16-17 Integer ELEMENT 18-21 Character VALUE1 22-26 Integer MFLAG1 27-27 Character QFLAG1 28-28 Character SFLAG1 29-29 Character VALUE2 30-34 Integer MFLAG2 35-35 Character QFLAG2 36-36 Character SFLAG2 37-37 Character . . . . VALUE31 262-266 Integer MFLAG31 267-267 Character QFLAG31 268-268 Character SFLAG31 269-269 Character

Here's an example of data from Arizona...

```
stations[stations$ID=="US1AZMR0019",]

##                ID     LAT      LONG  ELEV STATE
## 48124 US1AZMR0019 33.5902 -111.9712 418.5    AZ
##                                    NAME GSN HCN_CRN WHOID
## 48124  SCOTTSDALE 8.8 SW                                NA

# head(stations[stations£HCN_CRN==" CRN",])
```

Let's get the a different site into R

I often forget how to make loops, so I often use simple examples that help me remember, for example,

```
# practicing loops
for (year in c(2010,2011,2012,2013,2014,2015)){
  print(paste("The year is", year))
}

## [1] "The year is 2010"
## [1] "The year is 2011"
## [1] "The year is 2012"
## [1] "The year is 2013"
## [1] "The year is 2014"
## [1] "The year is 2015"
```

Since the data have a re-occuring set of variable names, I decided to create a vector of variable names, many of which are nearly the same. So, as you'll see, I had to create a loop to avoid having to type a ton (or 31 :-)) of different variables.

```
# Create New Varible Names
MFLAG=NA; QFLAG=NA; SFLAG=NA; VALUE=NA
```

```r
for (i in 1:31){
VALUE[i] = paste("DATE", i, sep="")
MFLAG[i] = paste("MFLAG", i, sep="")
QFLAG[i] = paste("QFLAG", i, sep="")
SFLAG[i] = paste("SFLAG", i, sep="")
}

# Vector of variable names converted from a transposed matrix
tmp = as.vector(t(matrix(data=c(VALUE, MFLAG, QFLAG, SFLAG), ncol=4)))
Names = c("ID", "YEAR", "MONTH", "ELEMENT", tmp); length(Names)

## [1] 128
```

5.4 Process Selected Data Files

```r
setwd("/home/CAMPUS/mwl04747/github/Climate_Change_Narratives/Data")

dly_list = list.files(pattern="*.dly"); head(dly_list)

## [1] "AGM00060515.dly" "US1AZCN0021.dly"

#for (i in 1:length(dly_list))
for (i in 1:1){
tmp <- read.fwf(dly_list[i], widths = c(11, 4, 2, 4, rep(c(5, 1, 1, 1),31)))
names(tmp) <- Names
assign(dly_list[i], subset(tmp, ELEMENT=="TMAX", select=c(1:4, seq(5, by = 4, length.out=31)
}


tmp1 = melt(AGM00060515.dly, id=c("ID", "YEAR", "MONTH", "ELEMENT"))
head(tmp1)

##            ID YEAR MONTH ELEMENT variable value
## 1 AGM00060515 1984     3    TMAX    DATE1 -9999
## 2 AGM00060515 1984     4    TMAX    DATE1   190
## 3 AGM00060515 1984     5    TMAX    DATE1 -9999
## 4 AGM00060515 1984     6    TMAX    DATE1 -9999
## 5 AGM00060515 1984     7    TMAX    DATE1   430
## 6 AGM00060515 1984     8    TMAX    DATE1 -9999

tmp1$Day = as.numeric(str_sub(tmp1$variable,6,7)); head(tmp1)

##            ID YEAR MONTH ELEMENT variable value Day
## 1 AGM00060515 1984     3    TMAX    DATE1 -9999  NA
## 2 AGM00060515 1984     4    TMAX    DATE1   190  NA
```

```
## 3 AGM00060515 1984     5    TMAX    DATE1 -9999   NA
## 4 AGM00060515 1984     6    TMAX    DATE1 -9999   NA
## 5 AGM00060515 1984     7    TMAX    DATE1   430   NA
## 6 AGM00060515 1984     8    TMAX    DATE1 -9999   NA
```

```
tmp1$value[tmp1$value==-9999] = NA; head(tmp1)
```

```
##            ID YEAR MONTH ELEMENT variable value Day
## 1 AGM00060515 1984     3    TMAX    DATE1    NA   NA
## 2 AGM00060515 1984     4    TMAX    DATE1   190   NA
## 3 AGM00060515 1984     5    TMAX    DATE1    NA   NA
## 4 AGM00060515 1984     6    TMAX    DATE1    NA   NA
## 5 AGM00060515 1984     7    TMAX    DATE1   430   NA
## 6 AGM00060515 1984     8    TMAX    DATE1    NA   NA
```

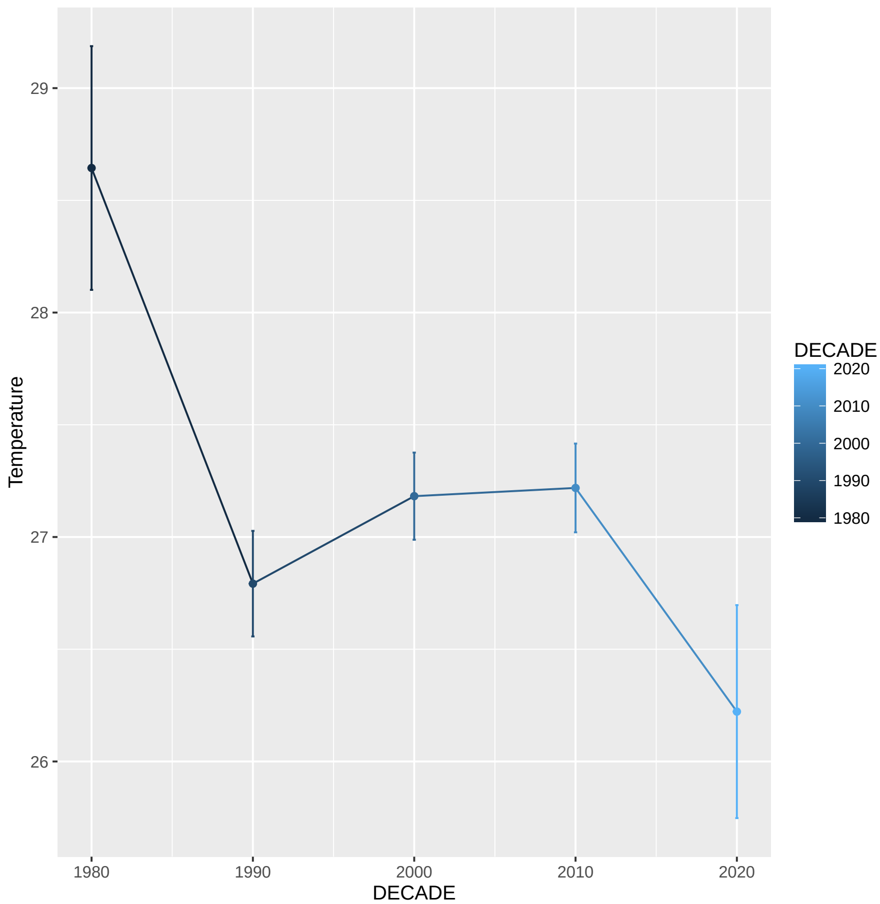
```
tmp1$Temperature = tmp1$value/10

drops <- c("variable","value")
tmp1 <-tmp1[ , !(names(tmp1) %in% drops)]
tmp1$DECADE = round(tmp1$YEAR, -1)
# names(tmp1)
```

6 Presenting the Results

```
# call summarySE function....somehow...


library(ggplot2)

summarydf <- summarySE(tmp1, "Temperature", "DECADE", na.rm=T)
```

```
## ------------------------------------------------------------------------
## You have loaded plyr after dplyr - this is likely to cause problems.
## If you need functions from both plyr and dplyr, please load plyr
first, then dplyr:
## library(plyr); library(dplyr)
## ------------------------------------------------------------------------
##
## Attaching package:  'plyr'
##
## The following objects are masked from 'package:dplyr':
##
##    arrange, count, desc, failwith, id, mutate, rename, summarise,
##    summarize
```

```r
# Think the color=DECADE thing can be deleted, but I haven't tried it yet. In any case, the

ggplot(summarydf, aes(y=Temperature, x=DECADE, color= DECADE)) + geom_errorbar(aes(ymin=Temp
```



6.1 NOAA dataset

New NOAA Directory – ftp://ftp.ncdc.noaa.gov/pub/data/noaa/

```r
library(raster)

## Loading required package:  sp
##
## Attaching package:  'raster'
##
## The following object is masked from 'package:dplyr':
##
```

```
##     select
##
## The following object is masked from 'package:tidyr':
##
##     extract
```

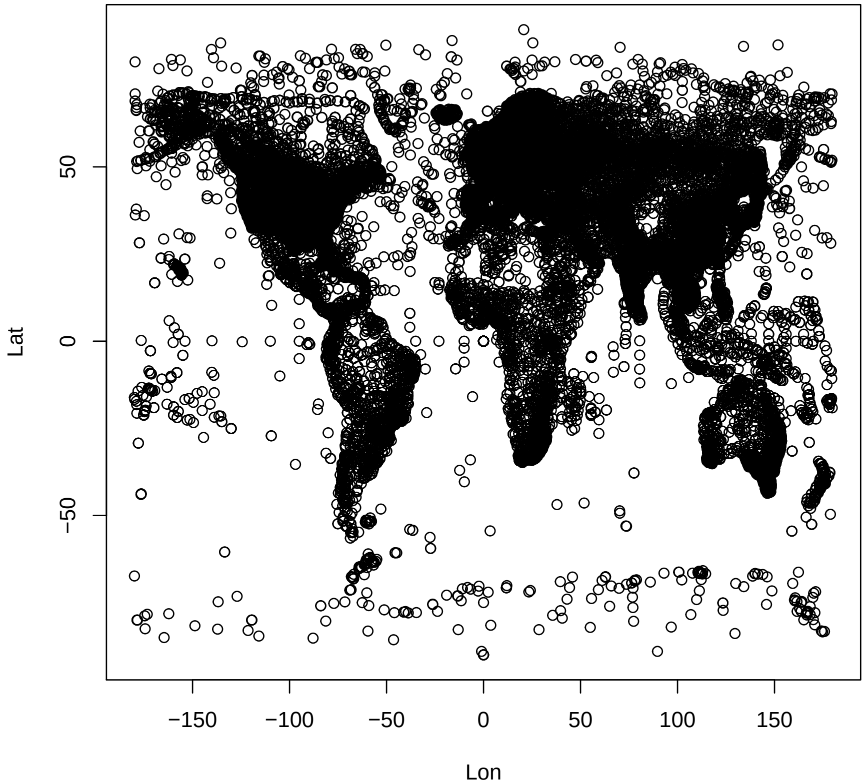
```
library(XML)

coords.fwt <- read.fwf("ftp://ftp.ncdc.noaa.gov/pub/data/noaa/isd-history.txt",widths=c(6,1,
Names = c("USAF", "X1", "WBAN", "X2", "STATION_NAME", "X3", "CTRY", "X4", "ST", "X5", "CALL"
Widths = c(6,       1,    5,      1,        29,         1,    2,      3,    2,    1,    4,

coords.fwt <- read.fwf("ftp://ftp.ncdc.noaa.gov/pub/data/noaa/isd-history.txt",widths=Widths
```

```
coords <- data.frame(ID=paste(as.factor(coords.fwt[,1])),WBAN=paste(as.factor(coords.fwt[,3]
```

```
## Warning in data.frame(ID = paste(as.factor(coords.fwt[, 1])), WBAN
= paste(as.factor(coords.fwt[, :  NAs introduced by coercion
## Warning in data.frame(ID = paste(as.factor(coords.fwt[, 1])), WBAN
= paste(as.factor(coords.fwt[, :  NAs introduced by coercion
```

NOAA Locations

```
plot(Lat ~ Lon, data=coords, xlim=c(-180, 180) )
```



7 Evaluating Narratives

7.1 Examples

7.2 Developing Criteria for Project Models

7.3

7.4