Lab 7_Solutions

Lab 7

In this lab we will practice working with raster data, in this case in the context of climate models. I have given you 4 sets of data:

1. Climate Model Data_Historic - this is a NetCDF file with output from a climate model. Data is monthly average air temperature for 1920-2005
2. Climate Model Data_Future - this is a NetCDF file with climate model output for the period 2006-2080
3. Observed Temp Data - this is gridded data based on weather station and satellite data. Data is monthly for 1991-2000
4. Population data - gridded counts of population for a number of years

The first part of the lab will compare modeled and observed climate data for major cities around the world. The second part of the lab will combine the population data and future climate data to project future changes in global temperature.

#Part 1

1a. Read in the historic climate model data as a SpatRaster. Use "TREFHT" (temperature at reference height) in the subds (sub-dataset) argument.

```
library(terra)
```

```
## terra 1.7.9
```

```
#reading historic climate data
setwd("C:\\Users\\Benny Panjaitan\\Documents\\GitHub\\esp106-Naomi\\W7 Lab\\Data")
hist_clim <- list.files("Climate Model Data_Historic", pattern="nc$", full.names=TRUE)

#making historic climate data as spatial raster data under the name hc (historic climate)
hc <- rast(hist_clim, "TREFHT")
hc
```

```
## class       : SpatRaster
## dimensions  : 192, 288, 1032  (nrow, ncol, nlyr)
## resolution  : 1.25, 0.9424084  (x, y)
## extent      : -0.625, 359.375, -90.4712, 90.4712  (xmin, xmax, ymin, ymax)
## coord. ref. :
## source      : b.e11.B20TRC5CNBDRD.f09_g16.002.cam.h0.TREFHT.192001-200512.nc:TREFHT
## varname     : TREFHT (Reference height temperature)
## names       : TREFHT_1, TREFHT_2, TREFHT_3, TREFHT_4, TREFHT_5, TREFHT_6, ...
## unit        :        K,        K,        K,        K,        K,        K, ...
## time (days) : 1920-02-01 to 2006-01-01
```

1b. Use ext() to see the longitude and latitude range of the SpatRaster you created. Note that the longitude goes form 0 to 360 (ish) instead of the more typical -180 to 180. This will cause a problem later on so use the rotate() function to change the longitude coordinates. Use extent again on the rotated object to check the longitude goes from -180 to 180 (ish)

```
ext(hc)
```

```
## SpatExtent : -0.625, 359.375, -90.4712041884817, 90.4712041884817 (xmin, xmax, ymin, ymax)
```

Initially, the longitude of the "hc" SpatRaster goes from 0 to 360 (-0.625 to 359.375)

```
hc <- rotate(hc)
ext(hc)
```

```
## SpatExtent : -180.625, 179.375, -90.4712041884817, 90.4712041884817 (xmin, xmax, ymin, ymax)
```

After we rotate it, it goes from 0 to 180 (-180.625 to 179.375)

2a. Use `rnaturalearth::ne_download()` function to get a sf object of major cities ("populated_places"). Use `vect` to coerce this to a SpatVector, and subset it to get just the 10 most populous cities based on 2020 population (POP2020 column)

```
#Hint 1: Check the object type of the POP2020 column. If necessary convert to a numeric vector using as

#Hint 2: The function order() will give you the index value corresponding to the ascending or descendin
```

```
#install.packages("rnaturalearth")

pop <- rnaturalearth::ne_download(
scale = 110,
type = "populated_places"
)

pop <- terra::vect(pop, geom=c("lon", "lat"), crs="", keepgeom=FALSE)
class(pop)
```

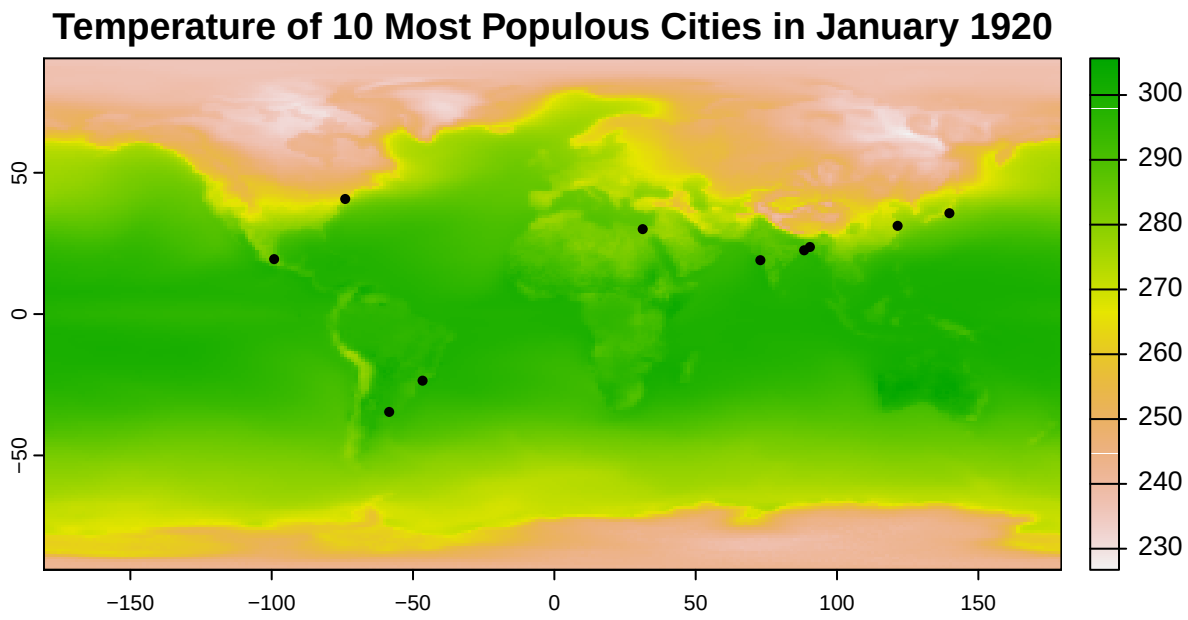
```
## [1] "SpatVector"
## attr(,"package")
## [1] "terra"
```

```
pop10 <- pop[order(pop$POP2020, decreasing = TRUE), ]
pop10 <- pop10[1:10,]
pop10$NAME
```

```
##  [1] "Tokyo"        "Mumbai"       "São Paulo"    "Mexico City"  "New York"
##  [6] "Shanghai"     "Kolkata"      "Dhaka"        "Cairo"        "Buenos Aires"
```

2b. Make a plot of the temperature data for January 1920 and overlay the 10 major cities.

```
plot(hc[[1]], main="Temperature of 10 Most Populous Cities in January 1920")
points(pop10)
```



2c. What about the plot gives you confidence this is actually showing temperature data from a January? What are the units of the temperature data?

Answer: Because the plot calls the first data in hc SpatRaster. The temperature unit was initially in Kelvin, but I change it to Celcius as below.

```
hc <- (hc-273.15)
units(hc) <- "C"
hc
```

```
## class       : SpatRaster
## dimensions  : 192, 288, 1032  (nrow, ncol, nlyr)
## resolution  : 1.25, 0.9424084  (x, y)
## extent      : -180.625, 179.375, -90.4712, 90.4712  (xmin, xmax, ymin, ymax)
## coord. ref. :
## source(s)   : memory
## names       :  TREFHT_1,  TREFHT_2,  TREFHT_3,  TREFHT_4,  TREFHT_5,  TREFHT_6, ...
## min values  : -46.41039, -47.89117, -54.69149, -62.76728, -70.50164, -71.77230, ...
## max values  :  32.50683,  32.08068,  32.16586,  33.47888,  39.08105,  40.24868, ...
## unit        :         C,         C,         C,         C,         C,         C, ...
## time (days) : 1920-02-01 to 2006-01-01
```

3a. Read in the observed temperature data as a SpatRaster, using “tmp” for the sub-dataset argument

```
#reading observed temperature data
setwd("C:\\Users\\Benny Panjaitan\\Documents\\GitHub\\esp106-Naomi\\W7 Lab\\Data")
obs_temp <- list.files("Observed Temp Data", pattern="nc$", full.names=TRUE)

#making observed temperature data as spatial raster data under the name ot (observed temperature)
ot <- rast(obs_temp, subds="tmp", lyrs=NULL, drivers=NULL, opts=NULL, win=NULL)
ot
```

```
## class       : SpatRaster
## dimensions  : 360, 720, 120  (nrow, ncol, nlyr)
## resolution  : 0.5, 0.5  (x, y)
## extent      : -180, 180, -90, 90  (xmin, xmax, ymin, ymax)
## coord. ref. : lon/lat WGS 84
## source      : cru_ts4.03.1991.2000.tmp.dat.nc:tmp
## varname     : tmp (near-surface temperature)
## names       :           tmp_1,           tmp_2,           tmp_3,           tmp_4,           tmp_5,
## unit        : degrees Celsius, degrees Celsius, degrees Celsius, degrees Celsius, degrees Celsius, de
## time (days) : 1991-01-16 to 2000-12-16
```

3b. Note that this climate model data is for 1920-2005 but the observation data is only from 1991-2000. Subset the climate model data to just the years 1991-2000. Also change the units to match those of the observed climate data.

```
# hint: have a look at ?terra::time to see how to get years
# terra::time(hc, format="years")
```

I see the time in years format from terra::time function, but hide the results because there are so many of them to show.

Then, I subset row numbers 852 to 971, based on looking at years format time above, to get only data for years 1991-2000.

```
hco <- subset(hc, 852:971)
hco
```

```
## class       : SpatRaster
## dimensions  : 192, 288, 120  (nrow, ncol, nlyr)
## resolution  : 1.25, 0.9424084  (x, y)
## extent      : -180.625, 179.375, -90.4712, 90.4712  (xmin, xmax, ymin, ymax)
## coord. ref. :
## source(s)   : memory
## names       : TREFHT_852, TREFHT_853, TREFHT_854, TREFHT_855, TREFHT_856, TREFHT_857, ...
## min values  :  -44.42032,  -48.08484,  -47.13795,  -56.85676,  -61.24125,  -66.53745, ...
## max values  :   31.83303,   33.57092,   30.72607,   32.64471,   35.56732,   38.02841, ...
## unit        :          C,          C,          C,          C,          C,          C, ...
## time (days) : 1991-01-01 to 2000-12-01
```

4. Use terra::extract() to produce two data-frames, one with observed and one with modeled temperature values for each city.

```
city_obs <- terra::extract(ot, pop10, ID=FALSE)
city_sim <- terra::extract(hco, pop10, ID=FALSE)
```

We have to do a bit of data-wrangling to compare modeled and observed temperature data for each city.

5a. transpose, use the city names for column names and add a time column to both data-frames

```
city_obs$ID <- NULL
city_sim$ID <- NULL

city_obs <- data.frame(t(city_obs))
city_sim <- data.frame(t(city_sim))

colnames(city_obs) <- pop10$NAME
colnames(city_sim) <- pop10$NAME

city_obs$time <- time(ot)
city_sim$time <- time(hco)
```

5b. Use pivot_longer() from the tidyr package to turn both data-frames into tidy data-frames, with one row for each unique city-month combination

```
#Hint: you want to use the first 10 columns (cities) to pivot (cols argument in the pivot_longer functi
library(tidyr)
```

```
##
## Attaching package: 'tidyr'
```

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

```
city_obs <- pivot_longer(city_obs, cols=1:10, values_to="observed")
city_sim <- pivot_longer(city_sim, cols=1:10, values_to="simulated")
```

5c. Notice that the modeled and observed rasters have used slightly different conventions for naming the months. You can see this in the "name" column of the two data frames you made in 5b. The model output uses the first of the month (e.g. 1991.02.01) whereas the observational data uses the middle of the month (e.g. 1991.01.16). This is a problem since we want to merge together the two data frames to compare observed and simulated data.

To merge the two data frames together, first we need to "chop off" the last two digits in the month ids in both data frames. One way to do this is to use the substr() function to return some subset of a character vector.

change the variable "time" from Date to "yearmon" (character)

```
city_obs$time <- substr(city_obs$time,1,7)
city_sim$time <- substr(city_sim$time,1,7)

#sub("-02"," Feb", city_obs$time)
```

5d. Merge the observed and modeled city data into a single data-frame. In this case you could use `cbind`, but that it is safer to use `merge`

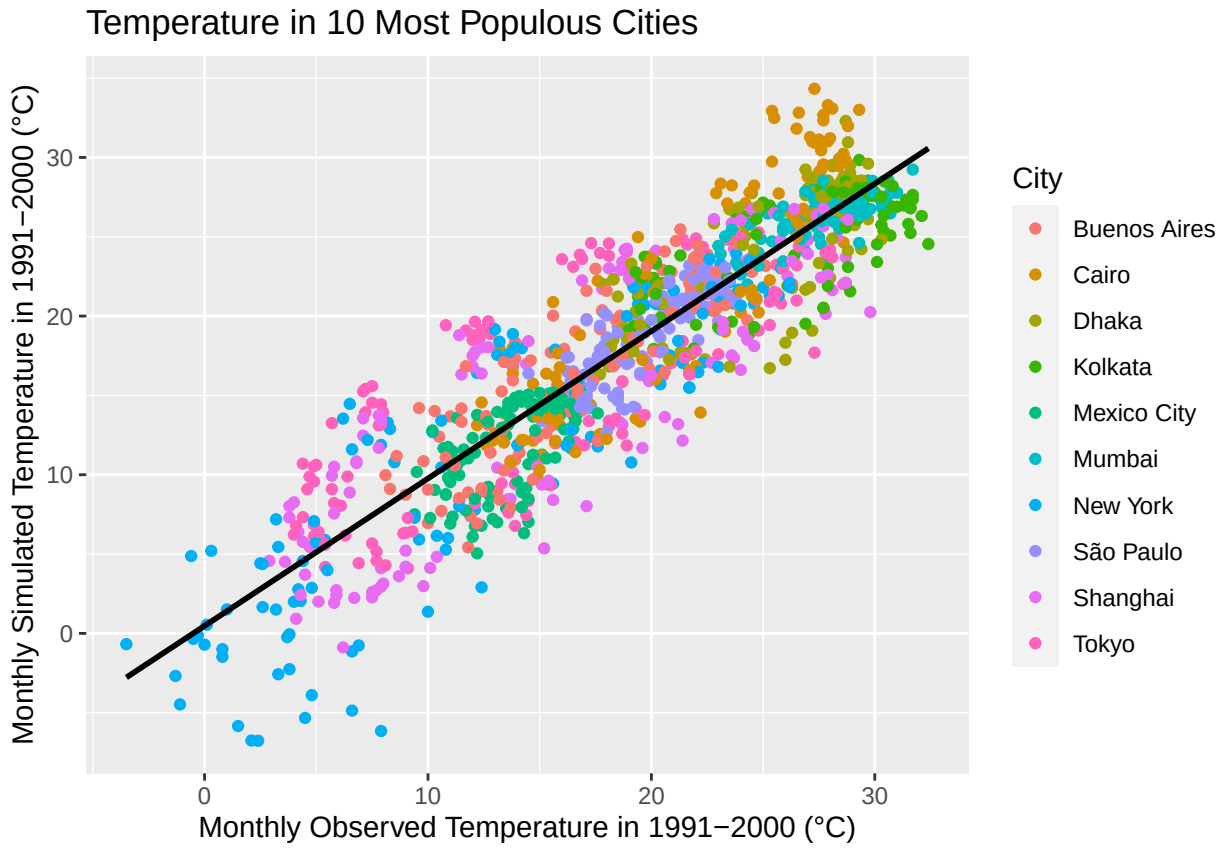
```
#Hint: you will want to specify two columns in the "by" argument in merge(). Think about what two colum

city_merge <- data.frame ()
city_merge <- merge.data.frame(city_obs, city_sim, by = c("time", "name"))
```

5e. Make a plot showing observed vs modeled temperature for the 10 cities. Add a 1:1 line which showing the exact match between observed and modeled data. You can use base plot or ggplot.

```
library(ggplot2)
ggplot(city_merge, aes(x=observed, y=simulated, color=name)) +
  geom_point() +
  geom_smooth(method=lm , color="black", se=F) +
  labs(title="Temperature in 10 Most Populous Cities", x="Monthly Observed Temperature in 1991-2000 (°C)
```

```
## `geom_smooth()` using formula = 'y ~ x'
```



#Part 2

In the second part of the lab, we will use projections of future temperature change (until 2080) and a map of the distribution of population in 2020 to get global, population-weighted projected warming.

6a. Read in the netCDF file with projected climate model temperature (in the "Climate Model Data_Future" directory) as a SpatRaster. Use the rotate() function again as you did in 1b to transform the coordinates

to -180 to 180 and the units to C. Use `subds="TREFHT"`. This has gridded projections of monthly global temperature between 2006 and 2020 under a high-emissions scenario (referred to as RCP8.5).

```
#read the projected climate model temperature
setwd("C:\\Users\\Benny Panjaitan\\Documents\\GitHub\\esp106-Naomi\\W7 Lab\\Data")
futr_clim <- list.files("Climate Model Data_Future", pattern="nc$", full.names=TRUE)
fc <- rast(futr_clim, "TREFHT")

#rotate the coordinates to -180 to 180
fc <- rotate(fc)

#change the units from Kelvin to Celcius degree
fc <- (fc-273.15)
units(fc) <- "C"

#check the SpatRaster of Climate Model in the Future (fc)
fc
```

```
## class       : SpatRaster
## dimensions  : 192, 288, 900  (nrow, ncol, nlyr)
## resolution  : 1.25, 0.9424084  (x, y)
## extent      : -180.625, 179.375, -90.4712, 90.4712  (xmin, xmax, ymin, ymax)
## coord. ref. :
## source(s)   : memory
## names       :  TREFHT_1,  TREFHT_2,  TREFHT_3,  TREFHT_4,  TREFHT_5,  TREFHT_6, ...
## min values  : -45.29093, -46.80746, -56.69919, -63.87290, -68.79848, -62.21677, ...
## max values  :  32.85928,  32.91741,  34.10244,  34.25445,  37.39761,  40.88925, ...
## unit        :         C,         C,         C,         C,         C,         C, ...
## time (days) : 2006-02-01 to 2081-01-01
```

6b. Compute the projected *annual* trend in global climate. Use `tapp` for this temporal aggregation.

```
fc <- tapp(fc, index="years", fun=mean)
fc
```

```
## class       : SpatRaster
## dimensions  : 192, 288, 76  (nrow, ncol, nlyr)
## resolution  : 1.25, 0.9424084  (x, y)
## extent      : -180.625, 179.375, -90.4712, 90.4712  (xmin, xmax, ymin, ymax)
## coord. ref. :
## source(s)   : memory
## names       :    y_2006,    y_2007,    y_2008,    y_2009,    y_2010,    y_2011, ...
## min values  : -57.85912, -55.56080, -55.82983, -56.11691, -54.26283, -55.08581, ...
## max values  :  30.03227,  30.04908,  29.50337,  29.93165,  31.11958,  29.89142, ...
## time (years): 2006 to 2081
```

7a. Read in the netCDF data on population in the "Population" directory as a SpatRaster. (There is only one variable in this netCDF, so you do not need to specify the variable name this time). This is gridded population count at 15 arc minute resolution.

```
#read the data of population
setwd("C:\\Users\\Benny Panjaitan\\Documents\\GitHub\\esp106-Naomi\\W7 Lab\\Data")
population <- list.files("Population", pattern="nc$", full.names=TRUE)
```

```
#make a spatial raster data of population
pplt <- rast(population)
pplt
```

```
## class       : SpatRaster
## dimensions  : 720, 1440, 20  (nrow, ncol, nlyr)
## resolution  : 0.25, 0.25  (x, y)
## extent      : -180, 180, -90, 90  (xmin, xmax, ymin, ymax)
## coord. ref. : lon/lat WGS 84
## source      : gpw_v4_population_count_adjusted_rev11_15_min.nc
## varname     : UN WPP-Adjusted Population Count, v4.11 (2000, 2005, 2010, 2015, 2020): 15 arc-minutes
## names       : UN WP~ter=1, UN WP~ter=2, UN WP~ter=3, UN WP~ter=4, UN WP~ter=5, UN WP~ter=6, ...
## unit        :     Persons,     Persons,     Persons,     Persons,     Persons,     Persons, ...
```

7b. We want only the 5th layer in this SpatRaster, which corresponds to population count in 2020. (Note - I know this from some associated files that came with the netCDF file. Take a look at the csv file in the directory to see this documentation). Pull out just the population in 2020.

```
pop2020 <- pplt[[5]]
pop2020
```

```
## class       : SpatRaster
## dimensions  : 720, 1440, 1  (nrow, ncol, nlyr)
## resolution  : 0.25, 0.25  (x, y)
## extent      : -180, 180, -90, 90  (xmin, xmax, ymin, ymax)
## coord. ref. : lon/lat WGS 84
## source      : gpw_v4_population_count_adjusted_rev11_15_min.nc
## varname     : UN WPP-Adjusted Population Count, v4.11 (2000, 2005, 2010, 2015, 2020): 15 arc-minutes
## name        : UN WPP-Adjusted Population Cou~2020): 15 arc-minutes_raster=5
## unit        :                                                        Persons
```

8a. Now we want to eventually match the population grid to the projected temperature grid. But the problem is that the grid size of the climate model is much larger than the grid size of the population data. How many rows and columns does the climate model data have? And how many rows and columns does the population data have? Use code to show that.

```
dim(fc)
```

```
## [1] 192 288  76
```

```
nrow(fc)
```

```
## [1] 192
```

```
ncol(fc)
```

```
## [1] 288
```

```
nlyr(fc)
```

```
## [1] 76
```

Annual-based Future Climate SpatRaster (fc) has 192 rows, 288 columns, and 76 layers

```
dim(pop2020)
```

```
## [1]  720 1440    1
```

```
nrow(pop2020)
```

```
## [1] 720
```

```
ncol(pop2020)
```

```
## [1] 1440
```

```
nlyr(pop2020)
```

```
## [1] 1
```

Population in 2020 SpatRaster (pop2020) has 720 rows, 440 columns, and 1 layer

8b. To fix this problem we can aggregate the population raster up to the resolution of the climate model using the aggregate() function. The population data you have is the population count (i.e. number of people in each grid cell). What function should we use to aggregate to larger grid cells? What function would we use instead if we had population density data instead of population count?

Answer: sum / mean

If it is a population count data, the function should be "sum"

On the other hand, if it is a population density data, the function should be "mean"

8c. Aggregate the population data to a higher level of resolution, as close as possible to the climate model data.

```
pop2020a <- aggregate(pop2020, fact=c(4,5), fun="sum", method="near")
dim(pop2020a)
```

```
## [1] 180 288   1
```

8d. If everything has gone according to plan, we would expect that summing up all the cells in the population raster should give us something close to the current population on the planet. Calculate that sum from your aggregated population data and compare to the total population today.

Answer:

```
sum(pop2020)
```

```
## class       : SpatRaster
## dimensions  : 720, 1440, 1  (nrow, ncol, nlyr)
## resolution  : 0.25, 0.25  (x, y)
## extent      : -180, 180, -90, 90  (xmin, xmax, ymin, ymax)
## coord. ref. : lon/lat WGS 84
## source(s)   : memory
## name        :      sum
## min value   :        0
## max value   : 13191638
```

```
sum(pop2020a)
```

```
## class       : SpatRaster
## dimensions  : 180, 288, 1  (nrow, ncol, nlyr)
## resolution  : 1.25, 1  (x, y)
## extent      : -180, 180, -90, 90  (xmin, xmax, ymin, ymax)
## coord. ref. : lon/lat WGS 84
## source(s)   : memory
## name        :      sum
## min value   :        0
## max value   : 42462471
```

The total number between the initial population data (13,191,638) and the aggregated population data (42,462,472) show big differences.

9a. Now we will use the population data to do a weighted averaging of the projected temperature data, to get the monthly temperature experienced by the average person between 2006 and 2080.

One problem is that even after the aggregation, the grids of the population data still don't quite match. Use terra::resample() to resample the aggregated population data to the climate model grid.

```
pop2020a <- terra::resample(pop2020a, fc, method="near")
pop2020a
```

```
## class       : SpatRaster
## dimensions  : 192, 288, 1  (nrow, ncol, nlyr)
## resolution  : 1.25, 0.9424084  (x, y)
## extent      : -180.625, 179.375, -90.4712, 90.4712  (xmin, xmax, ymin, ymax)
## coord. ref. : lon/lat WGS 84
## source(s)   : memory
## name        : UN WPP-Adjusted Population Cou~2020): 15 arc-minutes_raster=5
## min value   :                                                            0
## max value   :                                                     42462472
```

Now, the row and column dimension of population in 2020 (pop2020a) and future climate (fc) are the same.

9b. Now we can use the population raster to do a weighted average of the climate model data. Use the global() function to calculate both the global and and the population-weighted average temperature for each year.

```
avg_temp <- global(fc, fun="mean", na.rm=TRUE, weight=pop2020a)

avg_temp <- rast(avg_temp, type="x", crs="", digits=6, extent=NULL)
```

avg_temp is a data frame containing annual average temperature based on population in 2020.

Make a graph showing the projected annual trend in global climate. On the same graph show the temperature trend for the entire world, and weighted by population.

```
plot(pop2020a)
```

```
## Warning: [is.lonlat] coordinates are out of range for lon/lat
```

```
plot(avg_temp, add=T)
```