Part\_3\_Temp\_Phenology\_Data\_Together

## R Markdown

This is an R Markdown document. Markdown is a simple formatting syntax for authoring HTML, PDF, and MS Word documents. For more details on using R Markdown see <http://rmarkdown.rstudio.com>.

## Load Required Libraries

Also, set your working directory

# Load required libraries  
library(ggplot2)  
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(gridExtra)

##   
## Attaching package: 'gridExtra'

## The following object is masked from 'package:dplyr':  
##   
## combine

library(scales)  
  
options(stringsAsFactors=F) #keep strings as character type not factors  
  
# set working directory to ensure R can find the file we wish to import and where  
# we want to save our files. Be sure to move the download into your working directory!  
wd <- "~/Desktop/Data\_Intensive\_Ecology\_2021/Phenology\_Case\_Study/" # Change this to match your local environment

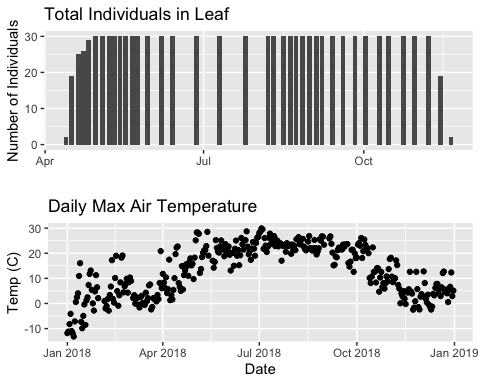
# Read in data ->  
temp\_day <- read.csv(paste0(wd,'NEON-pheno-temp-timeseries/NEONsaat\_daily\_SCBI\_2018.csv'))  
  
phe\_1sp\_2018 <- read.csv(paste0(wd,'NEON-pheno-temp-timeseries/NEONpheno\_LITU\_Leaves\_SCBI\_2018.csv'))  
  
# Convert dates  
temp\_day$Date <- as.Date(temp\_day$Date)  
# use dateStat - the date the phenophase status was recorded  
phe\_1sp\_2018$dateStat <- as.Date(phe\_1sp\_2018$dateStat)

### Separate Plots, Same Panel

n this dataset, we have phenology and temperature data from the Smithsonian Conservation Biology Institute (SCBI) NEON field site. There are a variety of ways we may want to look at this data, including aggregated at the site level, by a single plot, or viewing all plots at the same time but in separate plots. In the Work With NEON’s Plant Phenology Data and the Work with NEON’s Single-Aspirated Air Temperature Data tutorials, we created separate plots of the number of individuals who had leaves at different times of the year and the temperature in 2018.

However, plot the data next to each other to aid comparisons. The grid.arrange() function from the gridExtra package can help us do this.

# first, create one plot   
phenoPlot <- ggplot(phe\_1sp\_2018, aes(dateStat, countYes)) +  
 geom\_bar(stat="identity", na.rm = TRUE) +  
 ggtitle("Total Individuals in Leaf") +  
 xlab("") + ylab("Number of Individuals")  
  
# create second plot of interest  
tempPlot\_dayMax <- ggplot(temp\_day, aes(Date, dayMax)) +  
 geom\_point() +  
 ggtitle("Daily Max Air Temperature") +  
 xlab("Date") + ylab("Temp (C)")  
  
# Then arrange the plots - this can be done with >2 plots as well.  
grid.arrange(phenoPlot, tempPlot\_dayMax)

 Now, we can see both plots in the same window. But, hmmm… the x-axis on both plots is kinda wonky. We want the same spacing in the scale across the year (e.g., July in one should line up with July in the other) plus we want the dates to display in the same format(e.g. 2016-07 vs. Jul vs Jul 2018).

### Format Dates in Axis Labels

The date format parameter can be adjusted with scale\_x\_date. Let’s format the x-axis ticks so they read “month” (%b) in both graphs. We will use the syntax:

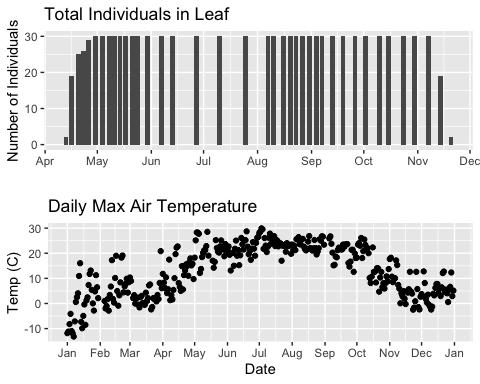
scale\_x\_date(labels=date\_format(“%b”")

Rather than re-coding the entire plot, we can add the scale\_x\_date element to the plot object phenoPlot we just created.

Data Tip:

1. You can type ?strptime into the R console to find a list of date format conversion specifications (e.g. %b = month). Type scale\_x\_date for a list of parameters that allow you to format dates on the x-axis.
2. If you are working with a date & time class (e.g. POSIXct), you can use scale\_x\_datetime instead of scale\_x\_date.

# format x-axis: dates  
phenoPlot <- phenoPlot +   
 (scale\_x\_date(breaks = date\_breaks("1 month"), labels = date\_format("%b")))  
  
tempPlot\_dayMax <- tempPlot\_dayMax +  
 (scale\_x\_date(breaks = date\_breaks("1 month"), labels = date\_format("%b")))  
  
# New plot.   
grid.arrange(phenoPlot, tempPlot\_dayMax)

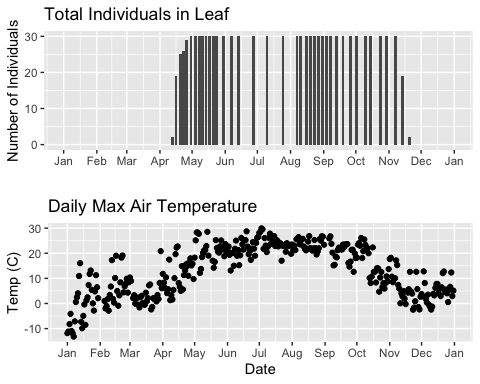
 But this only solves one of the problems, we still have a different range on the x-axis which makes it harder to see trends.

###Align data sets with different start dates

Now let’s work to align the values on the x-axis. We can do this in two ways, 1) setting the x-axis to have the same date range or 2) by filtering the dataset itself to only include the overlapping data. Depending on what you are trying to demonstrate and if you’re doing additional analyses and want only the overlapping data, you may prefer one over the other. Let’s try both.

Set range of x-axis Alternatively, we can set the x-axis range for both plots by adding the limits parameter to the scale\_x\_date() function.

# first, lets recreate the full plot and add in the   
phenoPlot\_setX <- ggplot(phe\_1sp\_2018, aes(dateStat, countYes)) +  
 geom\_bar(stat="identity", na.rm = TRUE) +  
 ggtitle("Total Individuals in Leaf") +  
 xlab("") + ylab("Number of Individuals") +  
 scale\_x\_date(breaks = date\_breaks("1 month"),   
 labels = date\_format("%b"),  
 limits = as.Date(c('2018-01-01','2018-12-31')))  
  
# create second plot of interest  
tempPlot\_dayMax\_setX <- ggplot(temp\_day, aes(Date, dayMax)) +  
 geom\_point() +  
 ggtitle("Daily Max Air Temperature") +  
 xlab("Date") + ylab("Temp (C)") +  
 scale\_x\_date(date\_breaks = "1 month",   
 labels=date\_format("%b"),  
 limits = as.Date(c('2018-01-01','2018-12-31')))  
  
# Plot  
grid.arrange(phenoPlot\_setX, tempPlot\_dayMax\_setX)



Now we can really see the pattern over the full year. This emphasizes the point that during much of the late fall, winter, and early spring none of the trees have leaves on them (or that data were not collected - this plot would not distinguish between the two).

##Subset one data set to match other

Alternatively, we can simply filter the dataset with the larger date range so the we only plot the data from the overlapping dates.

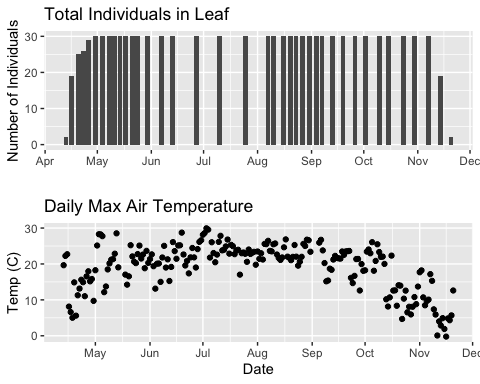
# filter to only having overlapping data  
temp\_day\_filt <- filter(temp\_day, Date >= min(phe\_1sp\_2018$date) &   
 Date <= max(phe\_1sp\_2018$date))  
  
# Check   
range(phe\_1sp\_2018$date)

## [1] "2018-04-13" "2018-11-20"

## [1] "2018-04-13" "2018-11-20"  
  
range(temp\_day\_filt$Date)

## [1] "2018-04-13" "2018-11-20"

## [1] "2018-04-13" "2018-11-20"  
  
#plot again  
tempPlot\_dayMaxFiltered <- ggplot(temp\_day\_filt, aes(Date, dayMax)) +  
 geom\_point() +  
 scale\_x\_date(breaks = date\_breaks("months"), labels = date\_format("%b")) +  
 ggtitle("Daily Max Air Temperature") +  
 xlab("Date") + ylab("Temp (C)")  
  
  
grid.arrange(phenoPlot, tempPlot\_dayMaxFiltered)



With this plot, we really look at the area of overlap in the plotted data (but this does cut out the time where the data are collected but not plotted).

###Same plot with two Y-axes

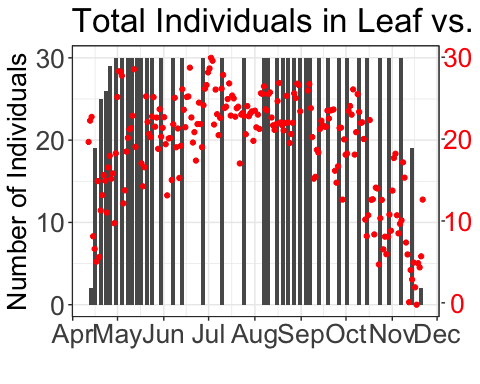
What about layering these plots and having two y-axes (right and left) that have the different scale bars?

Some argue that you should not do this as it can distort what is actually going on with the data. The author of the ggplot2 package is one of these individuals. Therefore, you cannot use ggplot() to create a single plot with multiple y-axis scales. You can read his own discussion of the topic on this StackOverflow post. <https://stackoverflow.com/questions/3099219/ggplot-with-2-y-axes-on-each-side-and-different-scales/3101876#3101876>

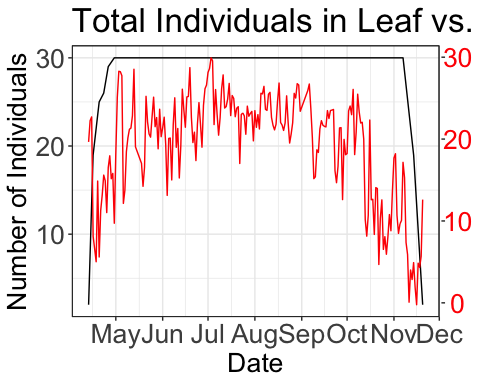
However, individuals have found work arounds for these plots. The code below is provided as a demonstration of this capability. Note, by showing this code here, we don’t necessarily endorse having plots with two y-axes.

This code is adapted from code by Jake Heare. <http://heareresearch.blogspot.com/2014/10/10-30-2014-dual-y-axis-graph-ggplot2_30.html>

# Source: http://heareresearch.blogspot.com/2014/10/10-30-2014-dual-y-axis-graph-ggplot2\_30.html  
  
# Additional packages needed  
library(gtable)  
library(grid)  
  
  
# Plot 1: Pheno data as bars, temp as scatter  
grid.newpage()  
phenoPlot\_2 <- ggplot(phe\_1sp\_2018, aes(dateStat, countYes)) +  
 geom\_bar(stat="identity", na.rm = TRUE) +  
 scale\_x\_date(breaks = date\_breaks("1 month"), labels = date\_format("%b")) +  
 ggtitle("Total Individuals in Leaf vs. Temp (C)") +  
 xlab(" ") + ylab("Number of Individuals") +  
 theme\_bw()+  
 theme(legend.justification=c(0,1),  
 legend.position=c(0,1),  
 plot.title=element\_text(size=25,vjust=1),  
 axis.text.x=element\_text(size=20),  
 axis.text.y=element\_text(size=20),  
 axis.title.x=element\_text(size=20),  
 axis.title.y=element\_text(size=20))  
  
  
tempPlot\_dayMax\_corr\_2 <- ggplot() +  
 geom\_point(data = temp\_day\_filt, aes(Date, dayMax),color="red") +  
 scale\_x\_date(breaks = date\_breaks("months"), labels = date\_format("%b")) +  
 xlab("") + ylab("Temp (C)") +  
 theme\_bw() %+replace%   
 theme(panel.background = element\_rect(fill = NA),  
 panel.grid.major.x=element\_blank(),  
 panel.grid.minor.x=element\_blank(),  
 panel.grid.major.y=element\_blank(),  
 panel.grid.minor.y=element\_blank(),  
 axis.text.y=element\_text(size=20,color="red"),  
 axis.title.y=element\_text(size=20))  
  
g1<-ggplot\_gtable(ggplot\_build(phenoPlot\_2))  
g2<-ggplot\_gtable(ggplot\_build(tempPlot\_dayMax\_corr\_2))  
  
pp<-c(subset(g1$layout,name=="panel",se=t:r))  
g<-gtable\_add\_grob(g1, g2$grobs[[which(g2$layout$name=="panel")]],pp$t,pp$l,pp$b,pp$l)  
  
ia<-which(g2$layout$name=="axis-l")  
ga <- g2$grobs[[ia]]  
ax <- ga$children[[2]]  
ax$widths <- rev(ax$widths)  
ax$grobs <- rev(ax$grobs)  
ax$grobs[[1]]$x <- ax$grobs[[1]]$x - unit(1, "npc") + unit(0.15, "cm")  
g <- gtable\_add\_cols(g, g2$widths[g2$layout[ia, ]$l], length(g$widths) - 1)  
g <- gtable\_add\_grob(g, ax, pp$t, length(g$widths) - 1, pp$b)  
  
grid.draw(g)



# Plot 2: Both pheno data and temp data as line graphs  
grid.newpage()  
phenoPlot\_3 <- ggplot(phe\_1sp\_2018, aes(dateStat, countYes)) +  
 geom\_line(na.rm = TRUE) +  
 scale\_x\_date(breaks = date\_breaks("months"), labels = date\_format("%b")) +  
 ggtitle("Total Individuals in Leaf vs. Temp (C)") +  
 xlab("Date") + ylab("Number of Individuals") +  
 theme\_bw()+  
 theme(legend.justification=c(0,1),  
 legend.position=c(0,1),  
 plot.title=element\_text(size=25,vjust=1),  
 axis.text.x=element\_text(size=20),  
 axis.text.y=element\_text(size=20),  
 axis.title.x=element\_text(size=20),  
 axis.title.y=element\_text(size=20))  
  
tempPlot\_dayMax\_corr\_3 <- ggplot() +  
 geom\_line(data = temp\_day\_filt, aes(Date, dayMax),color="red") +  
 scale\_x\_date(breaks = date\_breaks("months"), labels = date\_format("%b")) +  
 xlab("") + ylab("Temp (C)") +  
 theme\_bw() %+replace%   
 theme(panel.background = element\_rect(fill = NA),  
 panel.grid.major.x=element\_blank(),  
 panel.grid.minor.x=element\_blank(),  
 panel.grid.major.y=element\_blank(),  
 panel.grid.minor.y=element\_blank(),  
 axis.text.y=element\_text(size=20,color="red"),  
 axis.title.y=element\_text(size=20))  
  
g1<-ggplot\_gtable(ggplot\_build(phenoPlot\_3))  
g2<-ggplot\_gtable(ggplot\_build(tempPlot\_dayMax\_corr\_3))  
  
pp<-c(subset(g1$layout,name=="panel",se=t:r))  
g<-gtable\_add\_grob(g1, g2$grobs[[which(g2$layout$name=="panel")]],pp$t,pp$l,pp$b,pp$l)  
  
ia<-which(g2$layout$name=="axis-l")  
ga <- g2$grobs[[ia]]  
ax <- ga$children[[2]]  
ax$widths <- rev(ax$widths)  
ax$grobs <- rev(ax$grobs)  
ax$grobs[[1]]$x <- ax$grobs[[1]]$x - unit(1, "npc") + unit(0.15, "cm")  
g <- gtable\_add\_cols(g, g2$widths[g2$layout[ia, ]$l], length(g$widths) - 1)  
g <- gtable\_add\_grob(g, ax, pp$t, length(g$widths) - 1, pp$b)  
  
grid.draw(g)



You can save an image of this graph by right-clicking or highlighting and using the “Edit” menu up above to copy it. Now the fun. Interpret and extend!!!