lab 9: Gradient and Seasonality

The aim of this practical exercise is

- to learn how to replot data to emphasise changes in gradient, in two different ways.
- We will also use **grouping and faceting** to visualise seasonal variations.

Carbon dioxide (CO2),

- is one of three main greenhouse gases on Earth, along with methane and water vapour.
- Monitoring the concentration of these gases in the atmosphere is an extremely important matter in the 21st century because of the influence they have on temperature and weather patterns that affect water and food supply.

This practical uses data on the concentration of CO2 in the atmosphere, measured at Mauna Loa in Hawaii.

Mauna Loa has the oldest continuous record of atmospheric carbon dioxide measurements. This famous data set is a continuous series of measurements started in 1959 by Keeling, who realised that CO2 concentrations are increasing and it would be valuable to have an accurate and consistent series of measurements.

The data provided is from:

https://climatedataguide.ucar.edu/climate-data/overview-carbon-dioxide-co2-data-sets

Please look at this page; please also briefly use Google to look up the significance of the concentration of CO2 in the air, and also why CO2 concentration varies seasonally. (I ask you to do this because you should never investigate a data set without knowing anything about it!)

Get the dataset from ftp://aftp.cmdl.noaa.gov/products/trends/co2/co2_mm_mlo.txt save the file as **co2_header.txt**.

Edit the file and change the heading as necessary - The columns are:

- `year` and `month` are integers;
- `date` is year and month as a decimal year;
- `interp` is interpolated raw data (all there);
- `seasonal` is seasonally adjusted data that has been statistically processed to remove the yearly seasonal variations in CO2 concentration.
- `Days` is the number of days during the month on which measurements were taken (I think – look at the documentation!)

Read in the data from the file `co2_header.txt` using the function read.table

(You will have to read the documentation on read.table to do this. You need to set the header option to be TRUE to get the column names.)

`?read.table` to see the R documentation on read.table

```
#?read.table
#data <- read.table(file = "co2_header.txt", header = TRUE)
my.data <- read.delim("co2_header.txt", header = TRUE,sep = "")

str(my.data)
```

```
## 'data.frame':    768 obs. of  8 variables:
##  $ year    : int  1958 1958 1958 1958 1958 1958 1958 1958 1958 1958 ...
##  $ month   : int  3 4 5 6 7 8 9 10 11 12 ...
##  $ date    : num  1958 1958 1958 1958 1959 ...
##  $ interp  : num  316 317 318 317 316 ...
##  $ seasonal: num  314 315 315 315 315 ...
##  $ days    : int  -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 ...
##  $ st.dev  : num  -9.99 -9.99 -9.99 -9.99 -9.99 -9.99 -9.99 -9.99 -9.99 -9.99 ...
##  $ unc.of  : num  -0.99 -0.99 -0.99 -0.99 -0.99 -0.99 -0.99 -0.99 -0.99 -0.99 ...
```

```
names(my.data)
```

```
## [1] "year"     "month"    "date"     "interp"   "seasonal" "days"     "st.dev"
## [8] "unc.of"
```

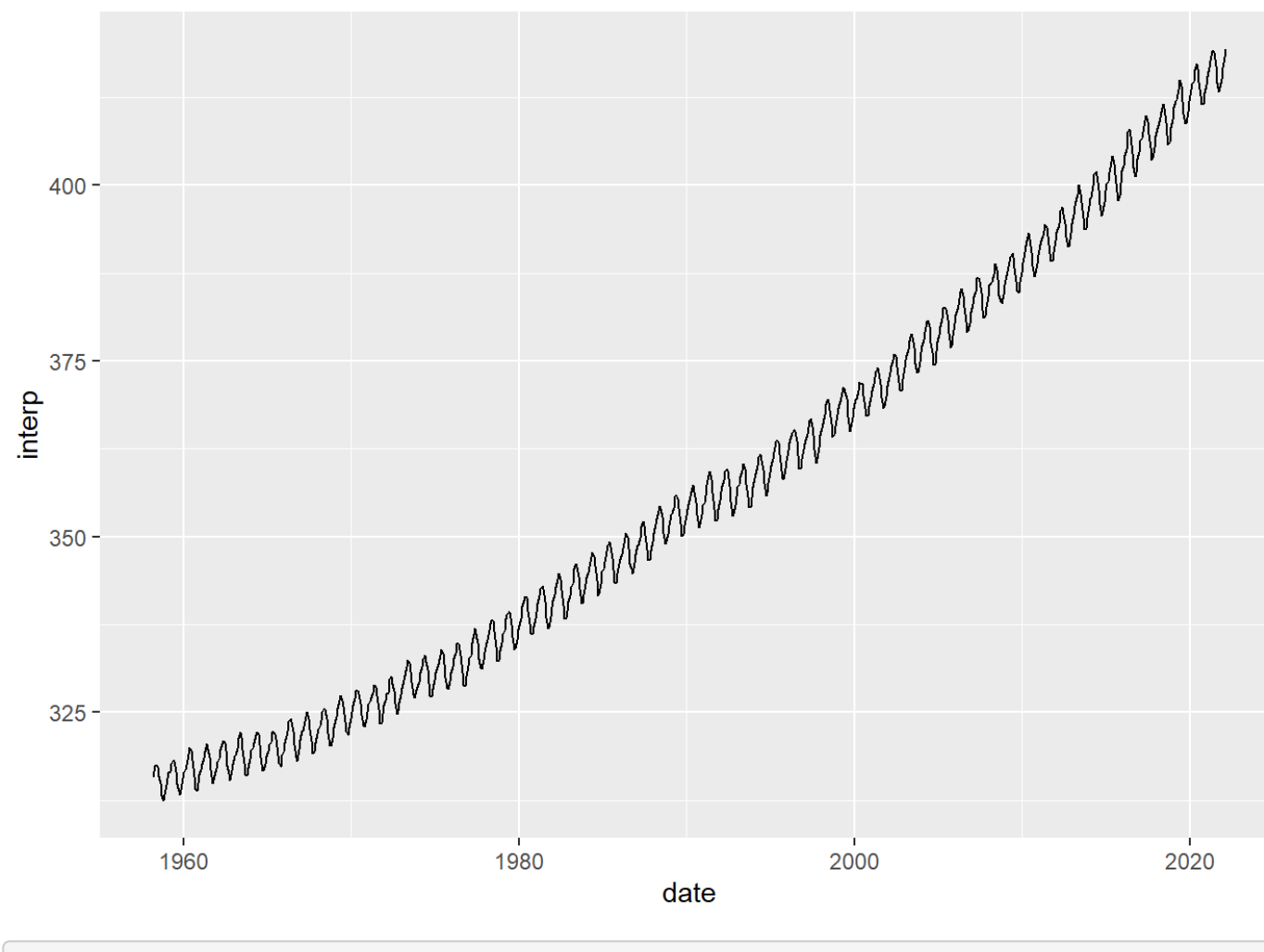
```
head(my.data, 10)
```

```
##    year month     date interp seasonal days st.dev unc.of
## 1  1958     3 1958.203 315.70   314.43   -1  -9.99  -0.99
## 2  1958     4 1958.288 317.45   315.16   -1  -9.99  -0.99
## 3  1958     5 1958.370 317.51   314.71   -1  -9.99  -0.99
## 4  1958     6 1958.455 317.24   315.14   -1  -9.99  -0.99
## 5  1958     7 1958.537 315.86   315.18   -1  -9.99  -0.99
## 6  1958     8 1958.622 314.93   316.18   -1  -9.99  -0.99
## 7  1958     9 1958.707 313.20   316.08   -1  -9.99  -0.99
## 8  1958    10 1958.789 312.43   315.41   -1  -9.99  -0.99
## 9  1958    11 1958.874 313.33   315.20   -1  -9.99  -0.99
## 10 1958    12 1958.956 314.67   315.43   -1  -9.99  -0.99
```

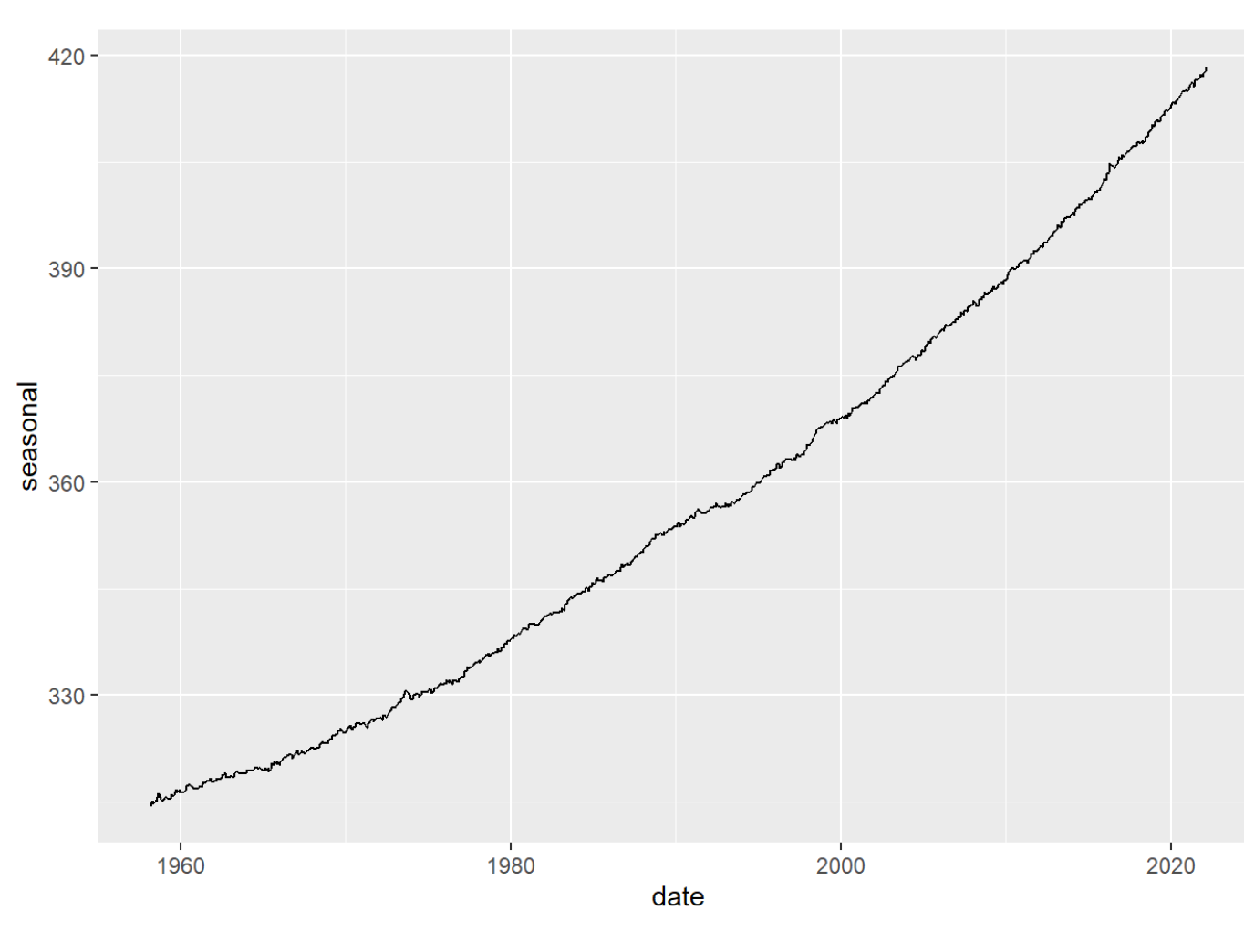
```
library(ggplot2)
```

```
## Warning: package 'ggplot2' was built under R version 4.0.5
```

```
p <- ggplot(my.data,aes(x=date,y=interp)) + geom_line()
p
```



```
p <- ggplot(my.data,aes(x=date,y=seasonal)) + geom_line()
p
```



The series exhibit strong seasonality and an obvious trend. A natural question is whether the rate at which CO2 has been increasing has changed over time. How can we plot this?

Method 1: Calculate the gradient then plot it: plotting differences

Find out about the r function diff Try diff(c(2,3,9,1 3)) has length one less than the vector.

```
diff(c(2,3,9,1, 3))
```

```
## [1]  1  6 -8  2
```

Now calculate differences of interp and seasonal

```
head(diff(my.data$interp),10)
```

```
##  [1]  1.75  0.06 -0.27 -1.38 -0.93 -1.73 -0.77  0.90  1.34  0.91
```

```
head(diff(my.data$seasonal),10)
```

```
##  [1]  0.73 -0.45  0.43  0.04  1.00 -0.10 -0.67 -0.21  0.23  0.12
```

```
my.data$interpdiff <- c(NA,diff(my.data$interp))
my.data$seasonaldiff <- c(NA,diff(my.data$seasonal))
```

Now plot a line graph of interpdiff.

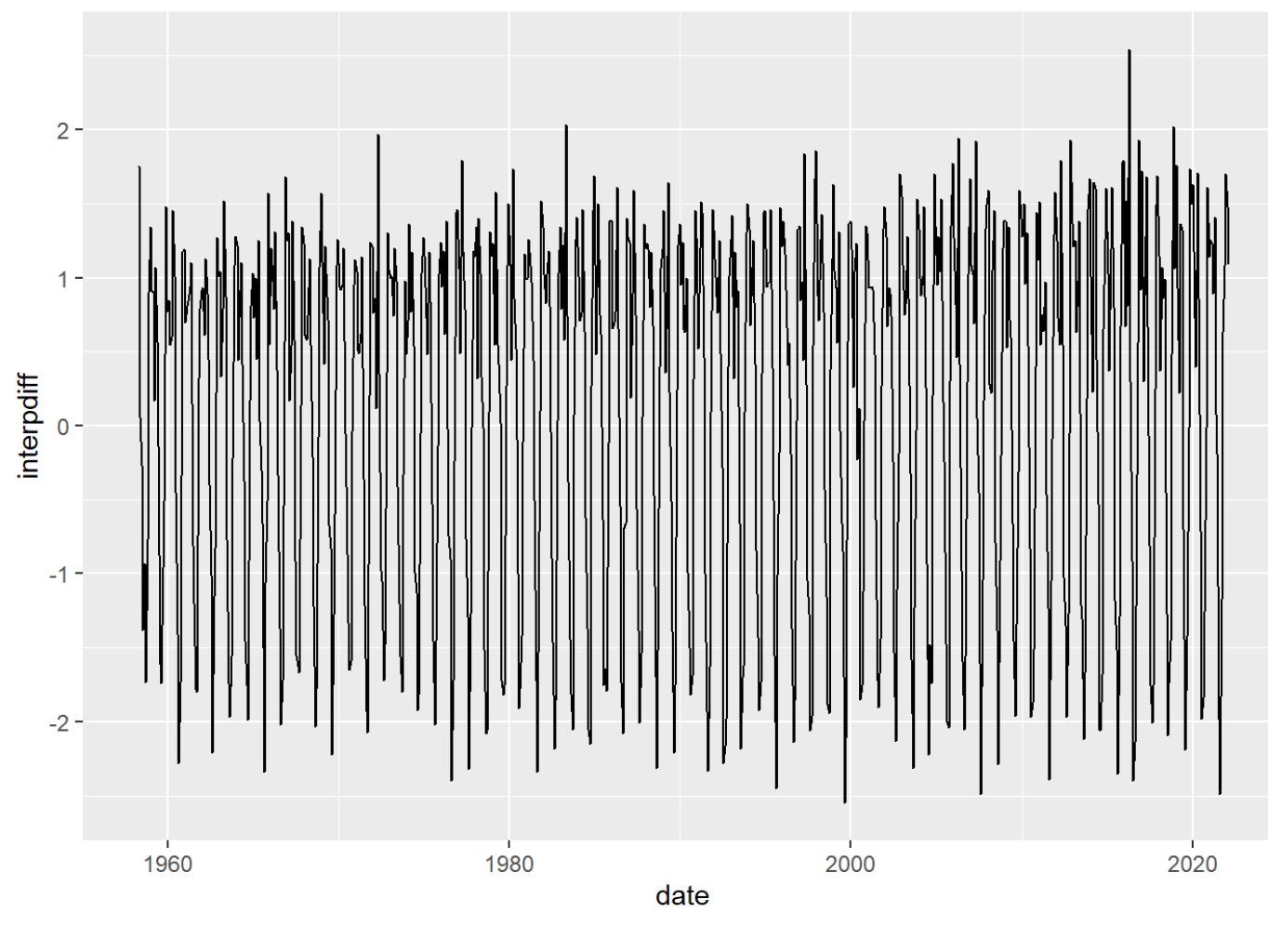
- Is this useful for telling if the gradient has increased over time?

```
head(my.data)
```

```
##   year month     date interp seasonal days st.dev unc.of interpdiff
## 1 1958     3 1958.203 315.70   314.43   -1  -9.99  -0.99         NA
## 2 1958     4 1958.288 317.45   315.16   -1  -9.99  -0.99       1.75
## 3 1958     5 1958.370 317.51   314.71   -1  -9.99  -0.99       0.06
## 4 1958     6 1958.455 317.24   315.14   -1  -9.99  -0.99      -0.27
## 5 1958     7 1958.537 315.86   315.18   -1  -9.99  -0.99      -1.38
## 6 1958     8 1958.622 314.93   316.18   -1  -9.99  -0.99      -0.93
##   seasonaldiff
## 1           NA
## 2         0.73
## 3        -0.45
## 4         0.43
## 5         0.04
## 6         1.00
```

```
p <- ggplot(my.data,aes(x=date,y=interpdiff)) + geom_line()
p
```

```
## Warning: Removed 1 row(s) containing missing values (geom_path).
```

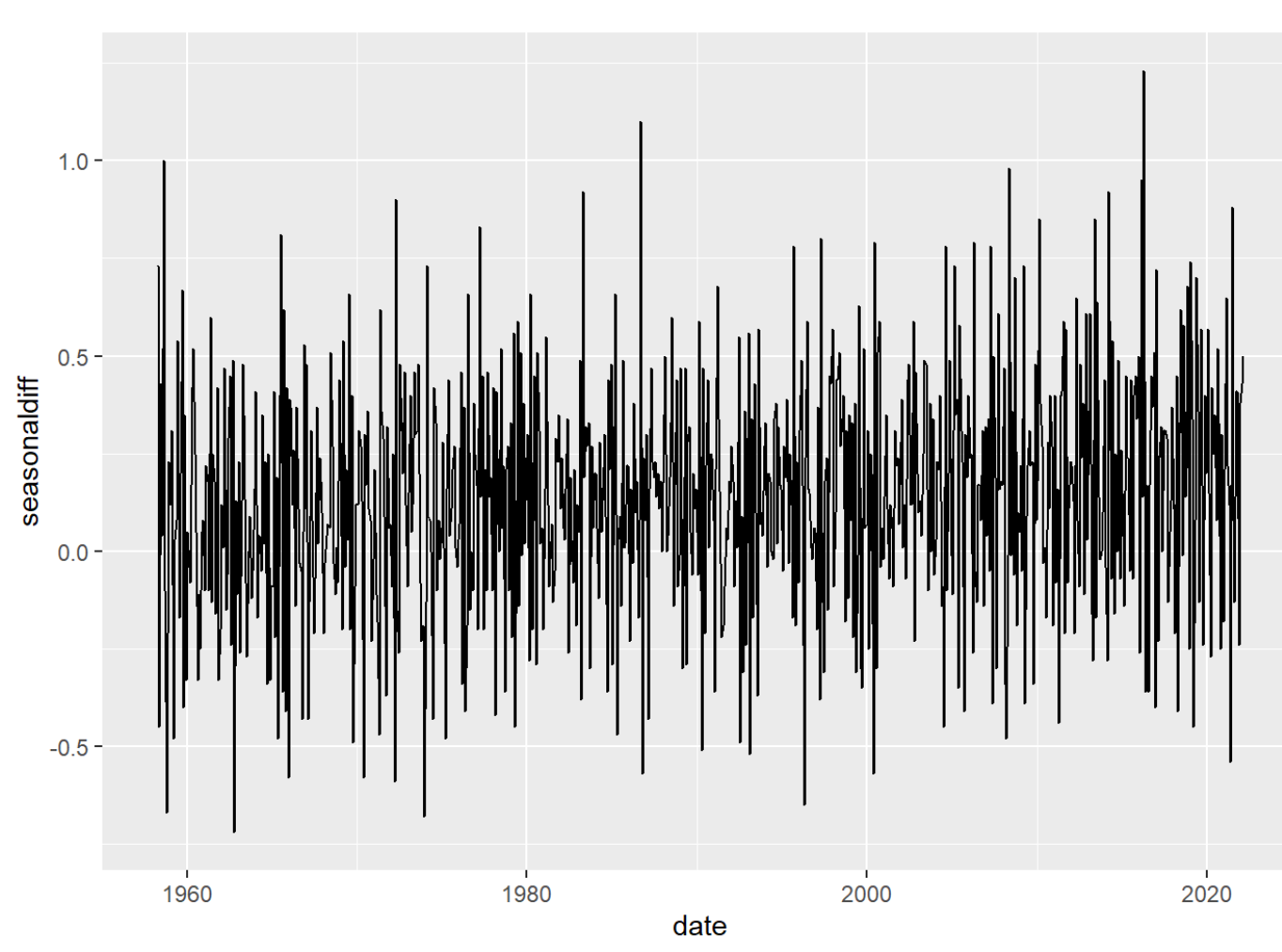


Do the same thing for seasonaldiff.

- Can you see whether gradient has increased over time?
- Is the plot clear?
- Does it show the gradient?
- What is the problem?

```
p <- ggplot(my.data,aes(x=date,y=seasonaldiff)) + geom_line()
p
```

```
## Warning: Removed 1 row(s) containing missing values (geom_path).
```



For what value of differencing – for both interp and seasonal – do you get a sensible looking result?

- (Hint: why is 12 more sensible than other long lags – look at the vertical scale.)

Method 2: Fit a line, then plot deviations from the line

```
model <- lm(interp ~ date, my.data)
model
```

```
##
## Call:
## lm(formula = interp ~ date, data = my.data)
##
## Coefficients:
## (Intercept)         date
##   -2835.453        1.604
```

Here ‘interp ~ date’ is a formula object. See: https://www.r-bloggers.com/forming-formulas/to find out what a formula object is.

What is ‘model’?

It is a list of many things:

- the intercept and slope of the line;
- the predicted values;
- the residuals;
- the dataframe itself, and a number of statistical diagnostics which are beyond the scope of this course.

```
str(model)
```

```
## List of 12
##  $ coefficients : Named num [1:2] -2835.5 1.6
##   ..- attr(*, "names")= chr [1:2] "(Intercept)" "date"
##  $ residuals    : Named num [1:768] 10.02 11.63 11.56 11.16 9.64 ...
##   ..- attr(*, "names")= chr [1:768] "1" "2" "3" "4" ...
##  $ effects      : Named num [1:768] -9892.12 821.33 10.58 10.17 8.66 ...
##   ..- attr(*, "names")= chr [1:768] "(Intercept)" "date" "" "" ...
##  $ rank         : int 2
##  $ fitted.values: Named num [1:768] 306 306 306 306 306 ...
##   ..- attr(*, "names")= chr [1:768] "1" "2" "3" "4" ...
##  $ assign       : int [1:2] 0 1
##  $ qr           :List of 5
##   ..$ qr   : num [1:768, 1:2] -27.7128 0.0361 0.0361 0.0361 0.0361 ...
##   .. ..- attr(*, "dimnames")=List of 2
##   .. .. ..$ : chr [1:768] "1" "2" "3" "4" ...
##   .. .. ..$ : chr [1:2] "(Intercept)" "date"
##   .. ..- attr(*, "assign")= int [1:2] 0 1
##   ..$ qraux: num [1:2] 1.04 1.06
##   ..$ pivot: int [1:2] 1 2
##   ..$ tol  : num 1e-07
##   ..$ rank : int 2
##   ..- attr(*, "class")= chr "qr"
##  $ df.residual  : int 766
##  $ xlevels      : Named list()
##  $ call         : language lm(formula = interp ~ date, data = my.data)
##  $ terms        :Classes 'terms', 'formula'  language interp ~ date
##   .. ..- attr(*, "variables")= language list(interp, date)
##   .. ..- attr(*, "factors")= int [1:2, 1] 0 1
##   .. .. ..- attr(*, "dimnames")=List of 2
##   .. .. .. ..$ : chr [1:2] "interp" "date"
##   .. .. .. ..$ : chr "date"
##   .. ..- attr(*, "term.labels")= chr "date"
##   .. ..- attr(*, "order")= int 1
##   .. ..- attr(*, "intercept")= int 1
##   .. ..- attr(*, "response")= int 1
##   .. ..- attr(*, ".Environment")=<environment: R_GlobalEnv>
##   .. ..- attr(*, "predvars")= language list(interp, date)
##   .. ..- attr(*, "dataClasses")= Named chr [1:2] "numeric" "numeric"
##   .. .. ..- attr(*, "names")= chr [1:2] "interp" "date"
##  $ model        :'data.frame':   768 obs. of  2 variables:
##   ..$ interp: num [1:768] 316 317 318 317 316 ...
##   ..$ date  : num [1:768] 1958 1958 1958 1958 1959 ...
##   ..- attr(*, "terms")=Classes 'terms', 'formula'  language interp ~ date
##   .. .. ..- attr(*, "variables")= language list(interp, date)
##   .. .. ..- attr(*, "factors")= int [1:2, 1] 0 1
##   .. .. .. ..- attr(*, "dimnames")=List of 2
##   .. .. .. .. ..$ : chr [1:2] "interp" "date"
##   .. .. .. .. ..$ : chr "date"
##   .. .. ..- attr(*, "term.labels")= chr "date"
##   .. .. ..- attr(*, "order")= int 1
##   .. .. ..- attr(*, "intercept")= int 1
##   .. .. ..- attr(*, "response")= int 1
##   .. .. ..- attr(*, ".Environment")=<environment: R_GlobalEnv>
##   .. .. ..- attr(*, "predvars")= language list(interp, date)
##   .. .. ..- attr(*, "dataClasses")= Named chr [1:2] "numeric" "numeric"
##   .. .. .. ..- attr(*, "names")= chr [1:2] "interp" "date"
##  - attr(*, "class")= chr "lm"
```

model$residuals is a vector of the differences between the observed values of interp and the values predicted by the model – and in this case these are just the vertical distances between the values and the line of best fit.

Having looked at your model, you will find that the intercept and the slope are the first two coefficients in `model$coefficients`.

Now look up how to add a line onto your plot – the geom to do this is `geom_abline`, and the documentation is at http://docs.ggplot2.org/current/geom_abline.html.

(There is also a section on how to do this in the R Graphics Cookbook at page 152.) Advice: you can find the intercept and slope in the array in model$coefficients. However, these are stored as a special data-type of ‘labelled number’. Many people have reported problems in putting these array elements into the geom_abline function: I am not yet sure what is causing this R quirk, but for now I advise you to work around it by simply typing in the numbers by hand. Sorry about that.

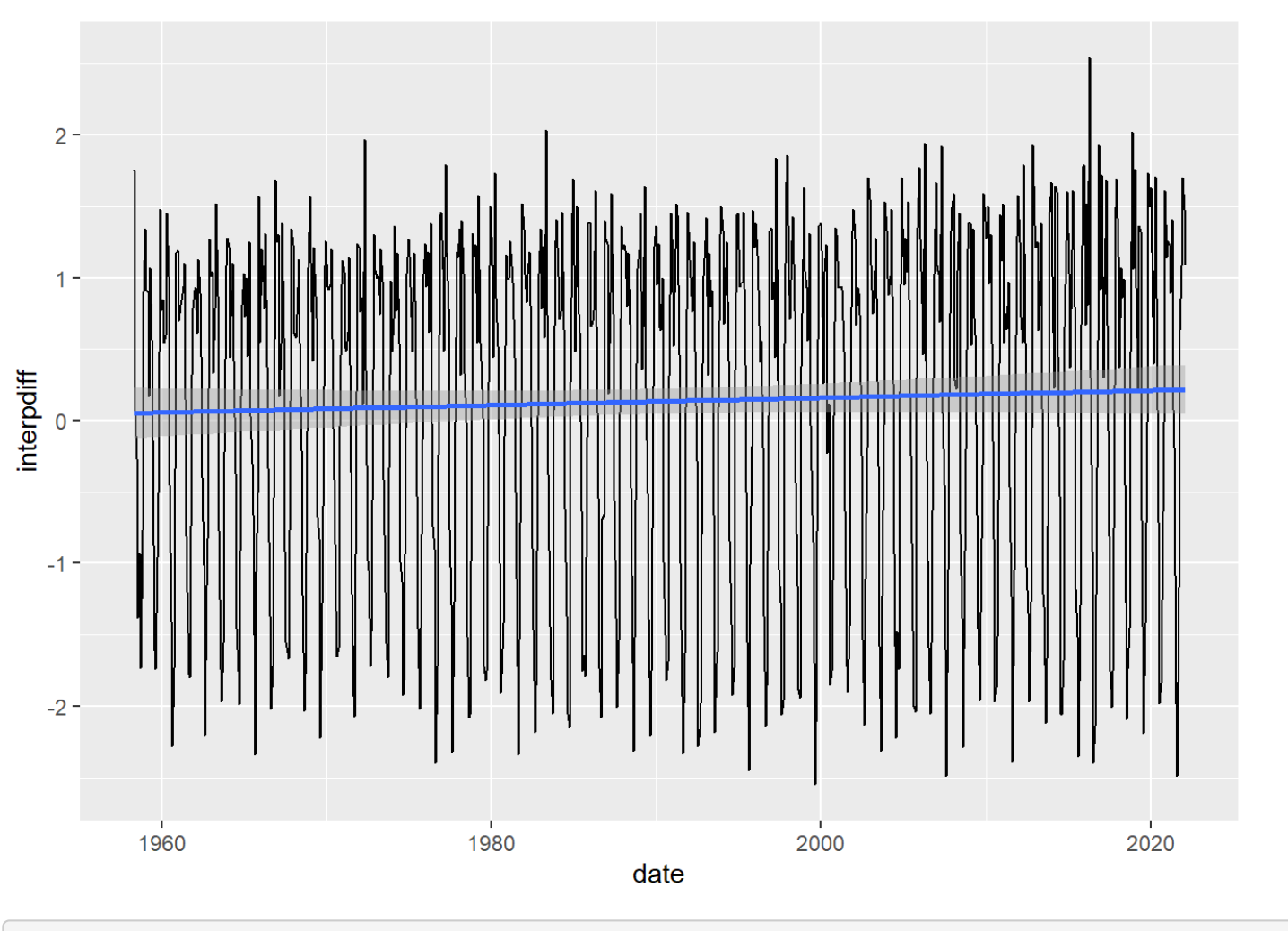
Check that the line you have added does look like a good fit to the plot! (You can get ggplot2 to add a line using geom_smooth(method="lm"))

```
p <- ggplot(my.data,aes(x=date,y=interpdiff)) + geom_line()+geom_abline()+geom_smooth(method="lm")
p
```

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

```
## Warning: Removed 1 rows containing non-finite values (stat_smooth).
```

```
## Warning: Removed 1 row(s) containing missing values (geom_path).
```

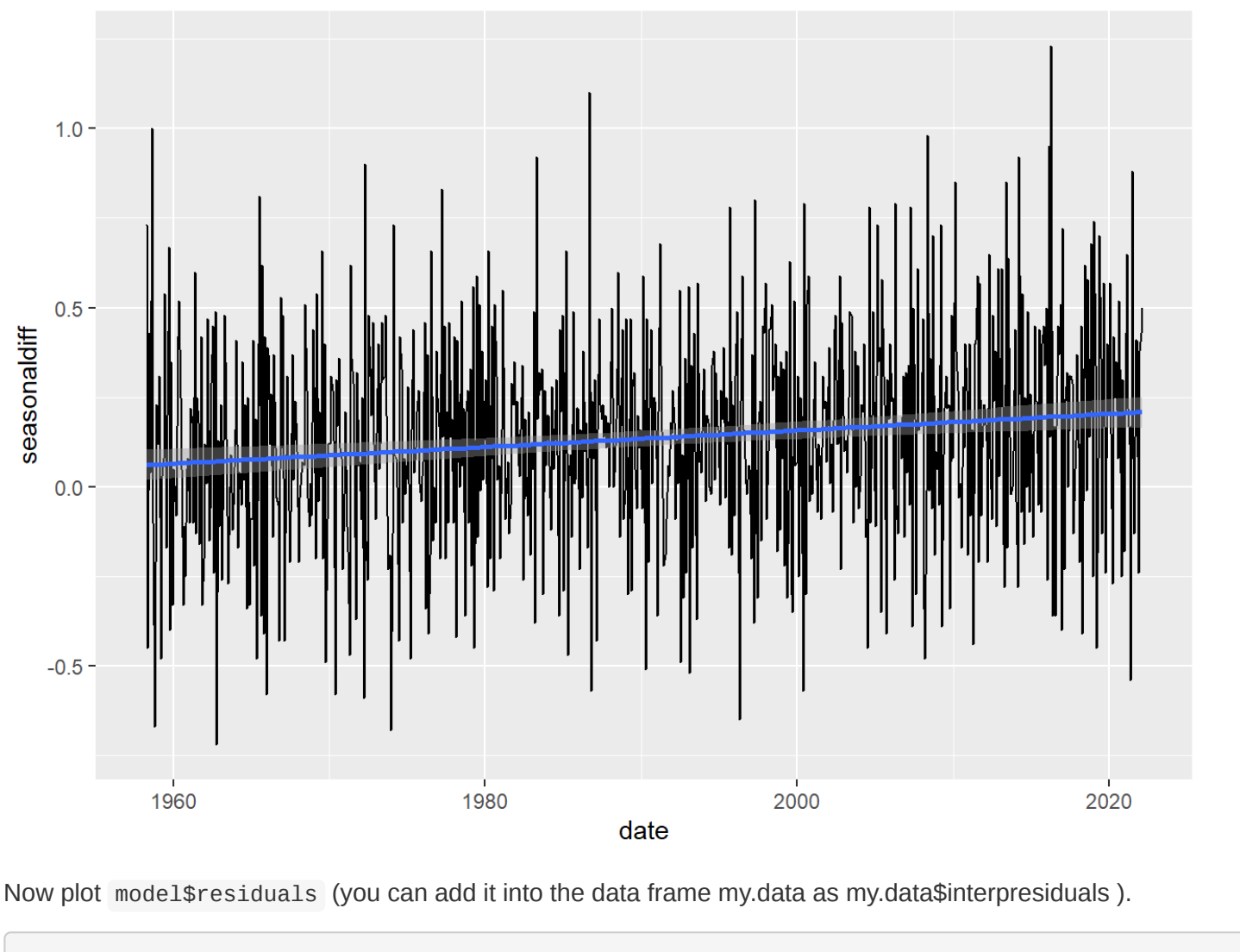


```
p <- ggplot(my.data,aes(x=date,y=seasonaldiff)) + geom_line()+geom_smooth(method="lm")
p
```

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

```
## Warning: Removed 1 rows containing non-finite values (stat_smooth).
```

```
## Warning: Removed 1 row(s) containing missing values (geom_path).
```



Now plot `model$residuals` (you can add it into the data frame my.data as my.data$interpresiduals).

```
model$residuals <- my.data$interpresiduals
names(my.data)
```

```
##  [1] "year"         "month"        "date"         "interp"       "seasonal"
##  [6] "days"         "st.dev"       "unc.of"       "interpdiff"   "seasonaldiff"
```

```
head(my.data)
```

```
##   year month     date interp seasonal days st.dev unc.of interpdiff
## 1 1958     3 1958.203 315.70   314.43   -1  -9.99  -0.99         NA
## 2 1958     4 1958.288 317.45   315.16   -1  -9.99  -0.99       1.75
## 3 1958     5 1958.370 317.51   314.71   -1  -9.99  -0.99       0.06
## 4 1958     6 1958.455 317.24   315.14   -1  -9.99  -0.99      -0.27
## 5 1958     7 1958.537 315.86   315.18   -1  -9.99  -0.99      -1.38
## 6 1958     8 1958.622 314.93   316.18   -1  -9.99  -0.99      -0.93
##   seasonaldiff
## 1           NA
## 2         0.73
## 3        -0.45
## 4         0.43
## 5         0.04
## 6         1.00
```

What do you see? Are changes in gradient clearer? Why?

Try the same procedure with `df$seasonal`. That is, fit a linear model to `df$seasonal`, add the residuals into the data frame, and then plot these residuals against date.

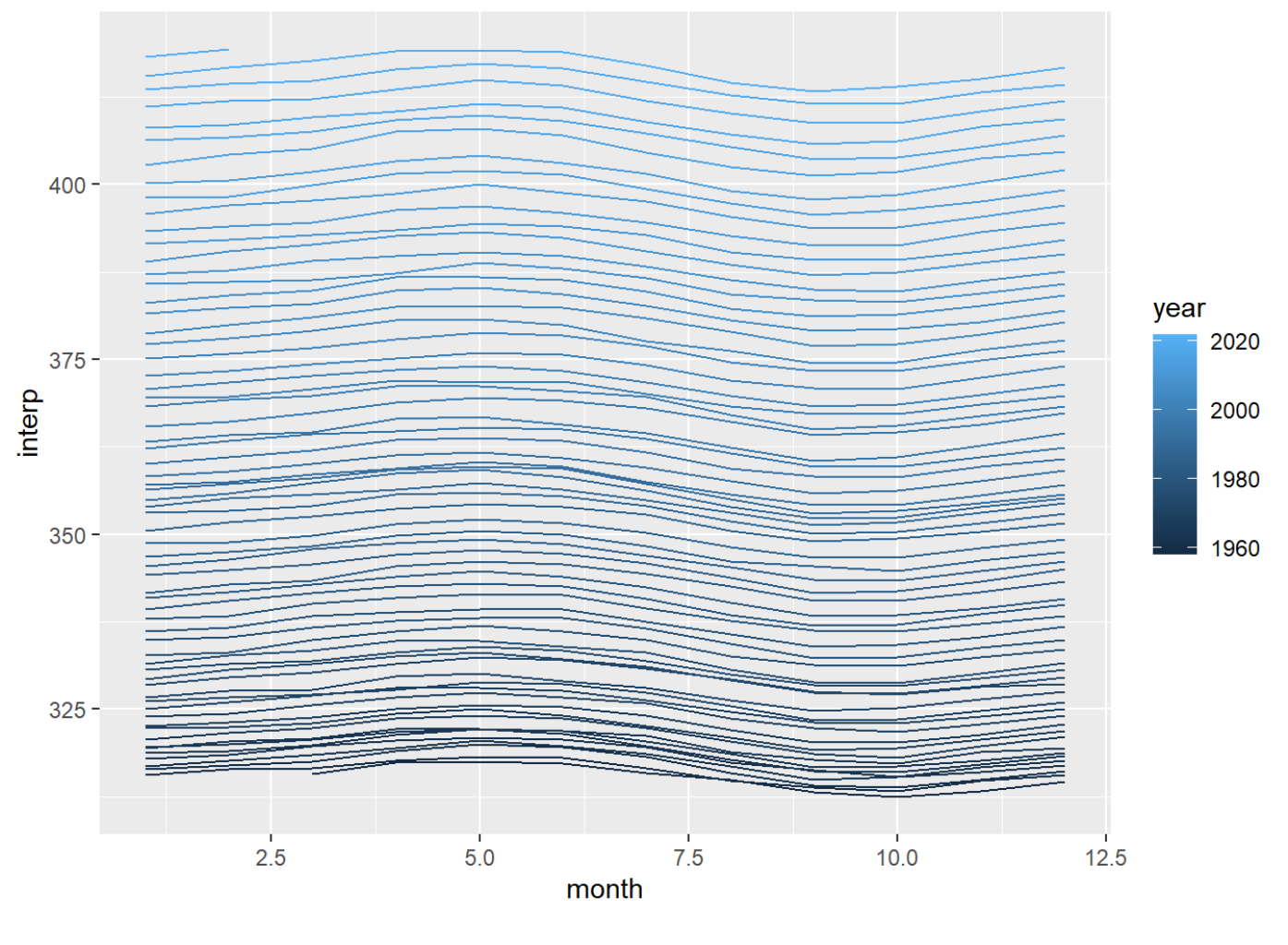
Why does this graph first slope down and then up?

Can you estimate the gradient for the first one-third of the time period, and the gradient for the last one-third of the time-period?

Seasonality via Faceting

Let's compare the variation during the year, in different years. That is, let's do a calendar plot.

```
p <- ggplot(my.data, aes(x=month, y=interp, group=year,colour=year)) + geom_line()
p
```



This plots all years above each other. The shape – and depth of variation – looks pretty similar.

We can adjust all the year means to be equal to zero using ddply:

```
library( plyr )
```

```
## Warning: package 'plyr' was built under R version 4.0.5
```

```
dfzeroyearmeans <- ddply(my.data,.(year), summarise, adjinterp=interp-mean(interp))
head(dfzeroyearmeans)
```

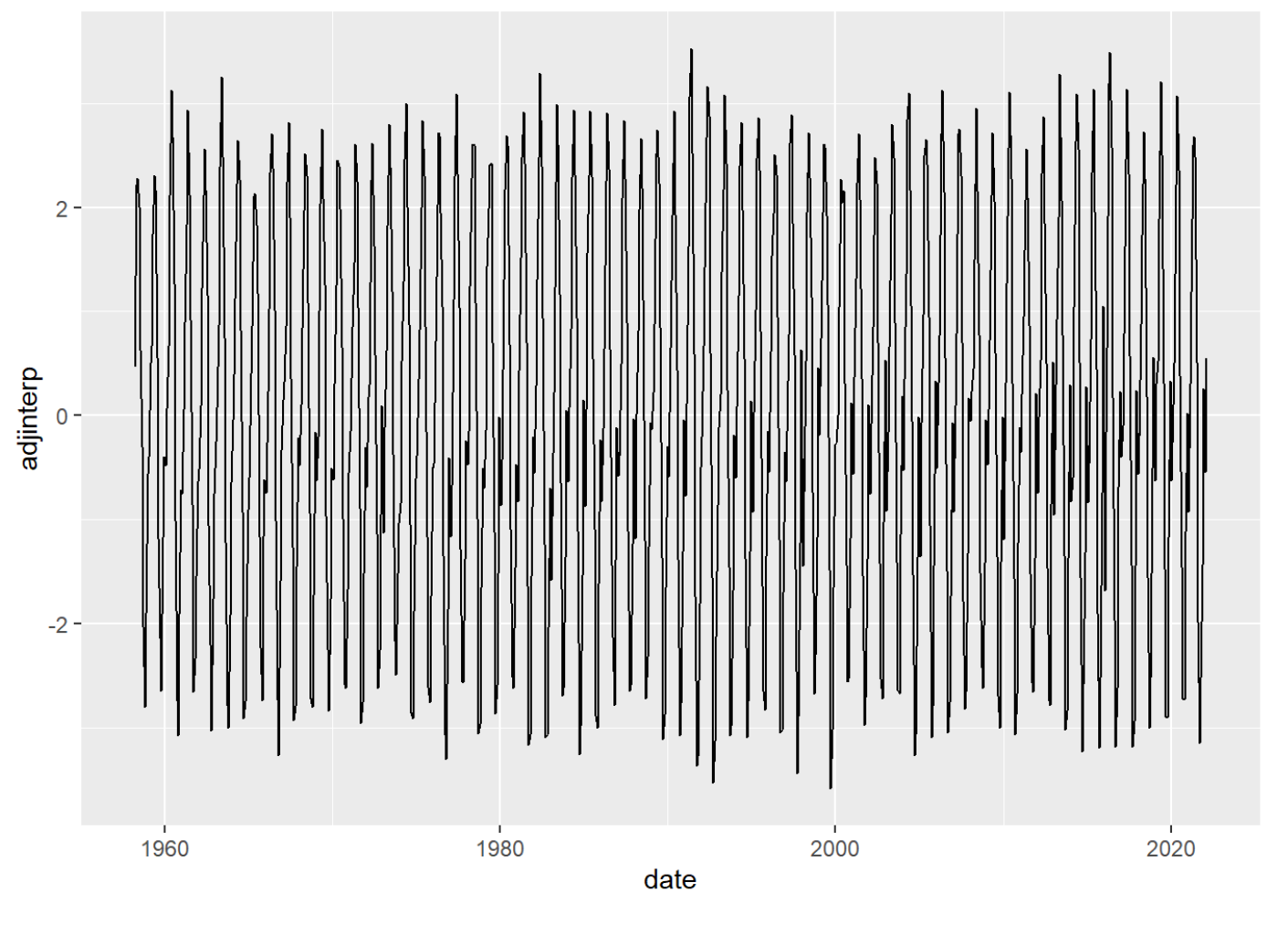
```
##   year adjinterp
## 1 1958     0.468
## 2 1958     2.218
## 3 1958     2.278
## 4 1958     2.008
## 5 1958     0.628
## 6 1958    -0.302
```

```
my.data$adjinterp <- dfzeroyearmeans$adjinterp
head(my.data$adjinterp)
```

```
## [1]  0.468  2.218  2.278  2.008  0.628 -0.302
```

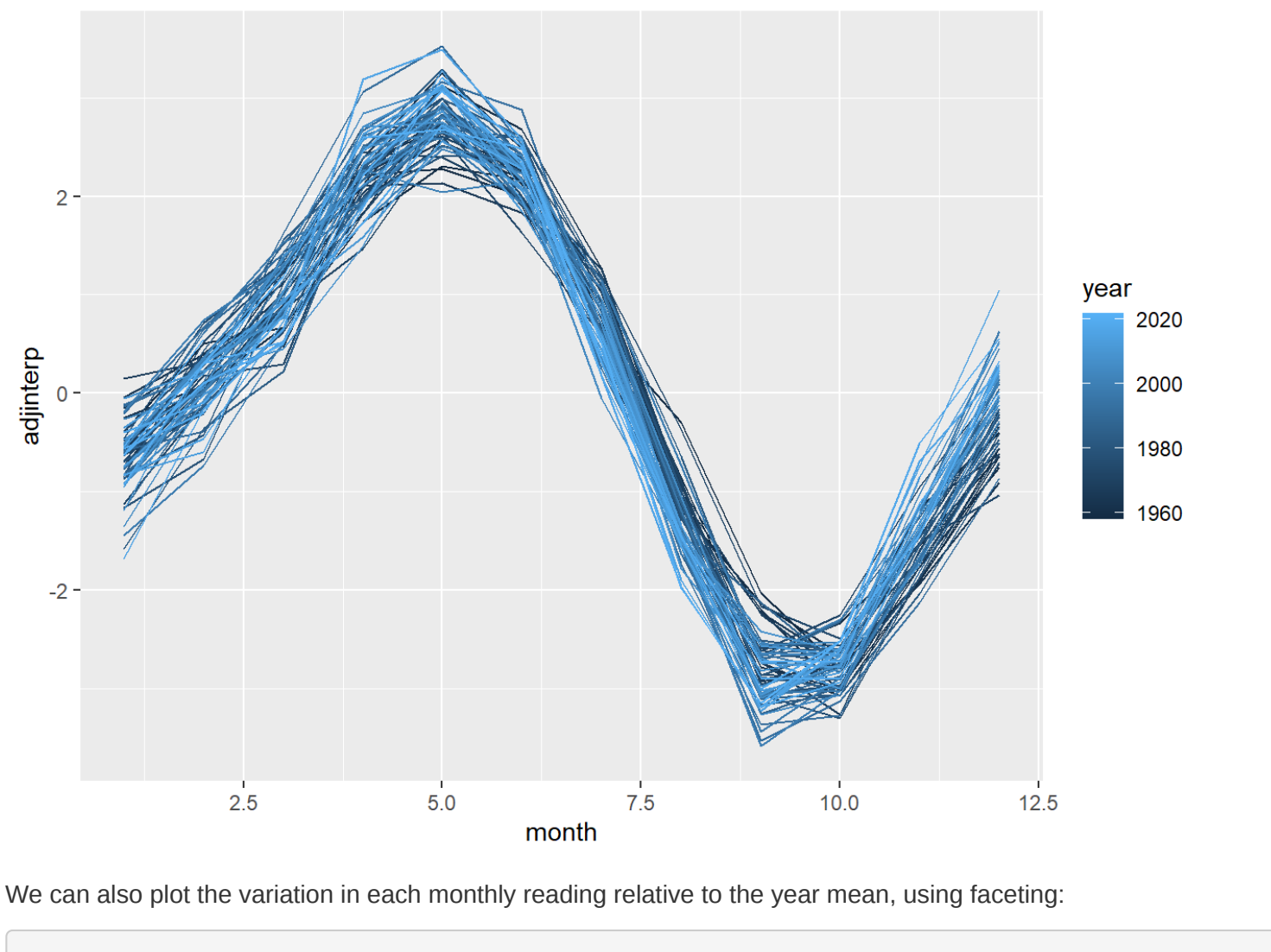
Plot adjinterp versus date to see what has happened. (They should all be zeroed.)

```
p <- ggplot(my.data, aes(x=date,y=adjinterp)) + geom_line()
p
```



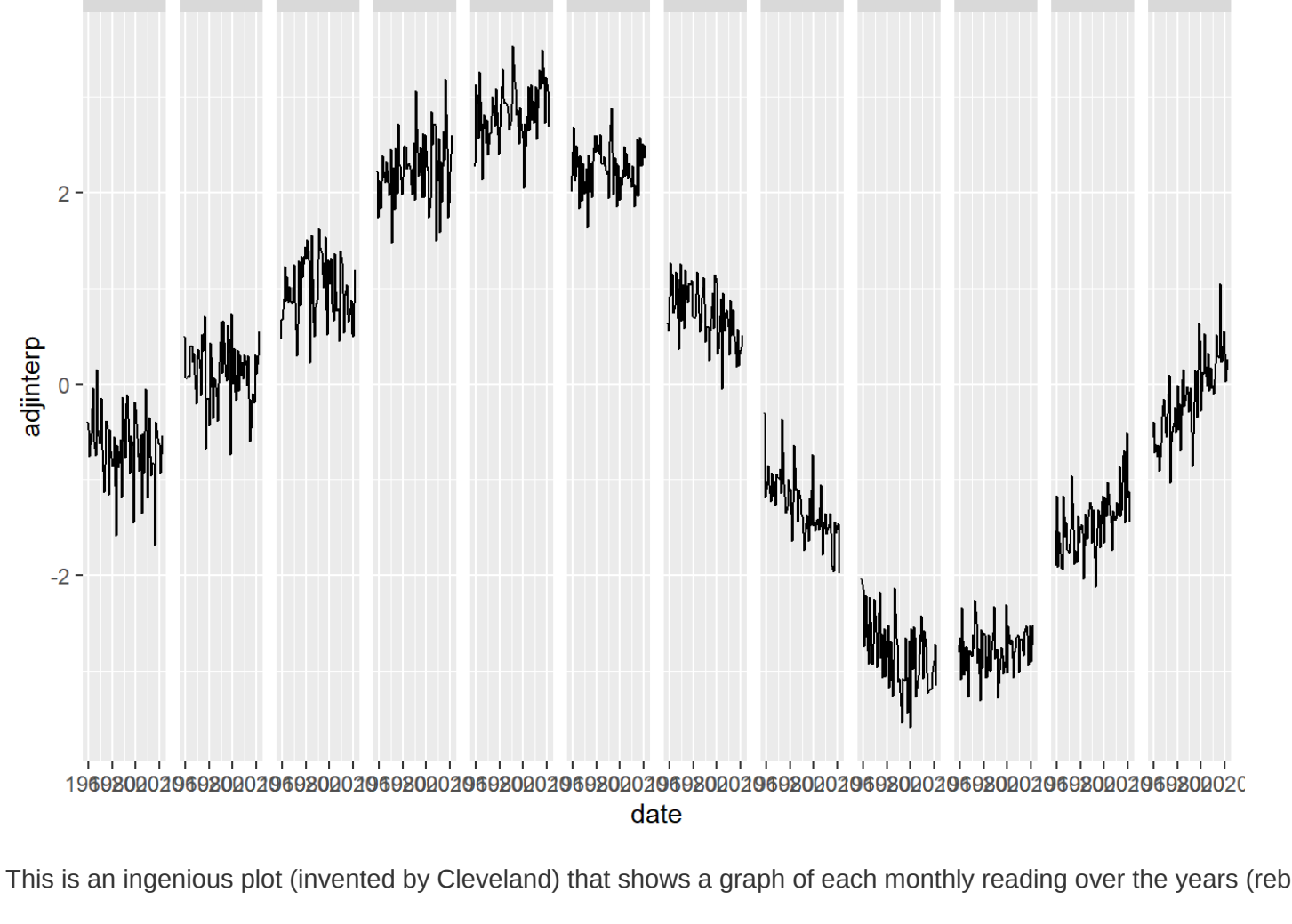
This is not a very good plot because the year-ends do not properly match up; let's plot all curves on top of each other:

```
p <- ggplot(my.data,aes(x=month,y=adjinterp, group=year, colour=year)) + geom_line()
p
```



We can also plot the variation in each monthly reading relative to the year mean, using faceting:

```
p <- ggplot(my.data,aes(x=date,y=adjinterp)) + geom_line() + facet_grid( . ~ month )
p
```



This is an ingenious plot (invented by Cleveland) that shows a graph of each monthly reading over the years (rebased relative to the yearly mean).

Can you explain why CO2 concentration increases in the winter?