Growth Analysis Documentation Draft

Matthew-Pierre Rogers

2022-09-14

Documentation for Plant Growth Analysis

This is documentation. You'll see how I treat with some results, but this is not the analysis of the actual results =)

Last modified 23/09/2022

Introduction

This Document serves as the documentation for the analysis of plant growth to be implemented later.

The document will be set up using the `Orange` dataset included in R.
This dataset is not identical to the data on seaweed which is to eventually be analyzed by the associated code. However it bears certain similarities.

Namely, both:

- Have a continuous measurement variable for growth(circumference as a stand in for mass and length)
- Have unique identifiers for variates
- Track units over some time period in days

Objectives

There are a number of objectives which should be met by this code and its associated documentation. These objectives include:

1. Visualizing the Growth of the variates
2. Determining the growth rates of the plants
    - determine the mean, and standard deviations of these growth rates
3. Identifying any trends in the growth rates
4. Determining if there is a signficant difference between growth rates at the sites
5. See how effectively length and mass can be used to predict each other

These objectives will be accomplished using functional programming as best as possible. This is beneficial for Reuse, and because i still think like i'm doing C.

Load The Necessary Packages

A number of packages beyond base R will be employed. They are as follows:

```
library(readxl) #for importing data
library(lubridate)

##
## Attaching package: 'lubridate'

## The following objects are masked from 'package:base':
##
##     date, intersect, setdiff, union

library(tidyverse) # for data manipulation and visualization with the
remaining packages

## ── Attaching packages
## ─────────────────────────────────────
## tidyverse 1.3.2 ──

## ✓ ggplot2 3.3.6      ✓ purrr   0.3.4
## ✓ tibble  3.1.8      ✓ dplyr   1.0.10
## ✓ tidyr   1.2.1      ✓ stringr 1.4.1
## ✓ readr   2.1.2      ✓ forcats 0.5.2
## ── Conflicts ──────────────────────────────────────
tidyverse_conflicts() ──
## ✖ lubridate::as.difftime() masks base::as.difftime()
## ✖ lubridate::date()        masks base::date()
## ✖ dplyr::filter()          masks stats::filter()
## ✖ lubridate::intersect()   masks base::intersect()
## ✖ dplyr::lag()             masks stats::lag()
## ✖ lubridate::setdiff()     masks base::setdiff()
## ✖ lubridate::union()       masks base::union()

library(gganimate)
library(gifski)
library(ggpubr)
```

Until the actual data is imported from excel/google sheets, `readxl::read_xls` is not needed. When it is, it should be saved in the same folder as the underlying code.

Structure of the Data

As the same team who gathered and collected the data, we should have be familiar with how it is structured. That said, its still a good practice to treat with how the data looks. For the `Orange` dataset used as a test run, this can be done as follows:

```
glimpse(Orange)

## Rows: 35
## Columns: 3
## $ Tree          <ord> 1, 1, 1, 1, 1, 1, 1, 2, 2, 2, 2, 2, 2, 2, 3, 3, 3,
3, 3,…
## $ age           <dbl> 118, 484, 664, 1004, 1231, 1372, 1582, 118, 484,
664, 10…
## $ circumference <dbl> 30, 58, 87, 115, 120, 142, 145, 33, 69, 111, 156,
172, 2…
```

The actual Seaweed dataset is to be structured differently.

Intended Data Structure

Variable	Type	Description
Identifier	Categorical	Identifies each Seedling
Site	Factor/Categorical	Identifies which site the seedlin was planted out at
Day	Measurement(discrete)	Tells the date on which data was taken
Mass	Measurement(continuous)	Mass of the Seedling in Grams
Length	Measurement(continuous)	Length of the Seedling in centimetres

There will also be room for comments but that is more qualitative and not to be considered too heavily in this particular analysis.

The circumference is similar enough to Mass and Length in the Seaweed data to be used as a good approximation in this Draft.

Checking and Cleaning the Data

Being familiar with the data, I may need to treat with missing Values and ensure everything is tidy. Mistakes do occur along the way. The `Orange` Dataset has no missing values. But Seaweed might. Specifically i want `NA` for the seaweed data to represent those lost from the line. I'd need some other method of dealing with them otherwise.

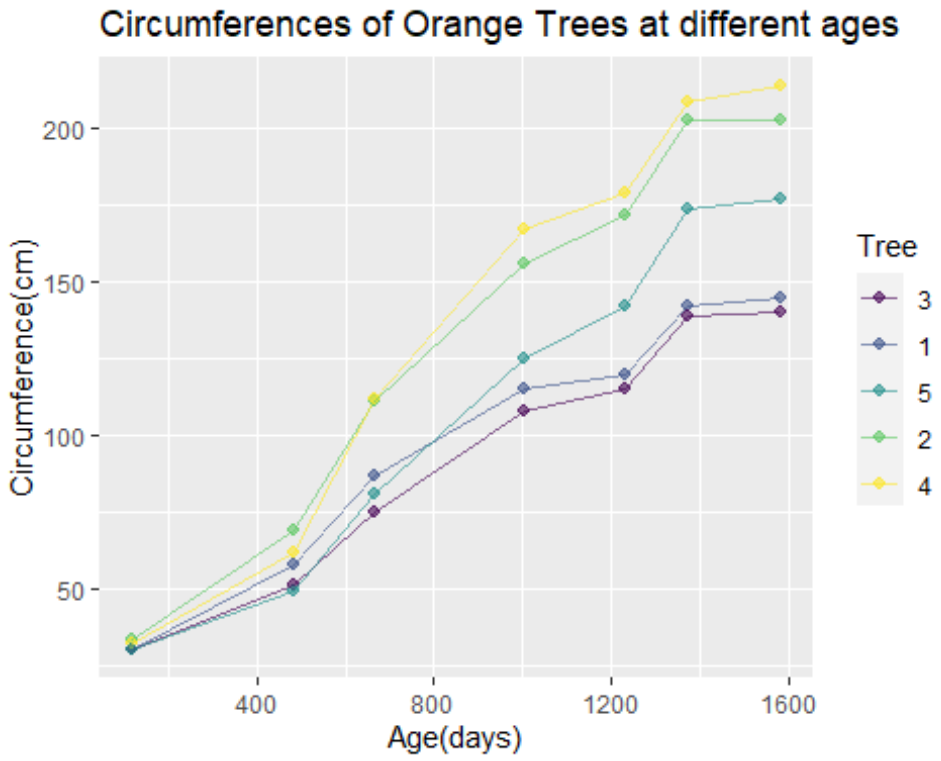
```
#Assess the data to see if it is missing anything and reveal the formatting
  data<-Orange
  glimpse(data)

## Rows: 35
## Columns: 3
## $ Tree          <ord> 1, 1, 1, 1, 1, 1, 1, 2, 2, 2, 2, 2, 2, 2, 3, 3, 3,
3, 3,…
## $ age           <dbl> 118, 484, 664, 1004, 1231, 1372, 1582, 118, 484,
664, 10…
## $ circumference <dbl> 30, 58, 87, 115, 120, 142, 145, 33, 69, 111, 156,
172, 2…

  full.record<-complete.cases(data)
  data<-data[full.record,] # Removes NA. will have to figure out how to treat
with that
```

This is more specific to the `Orange` data. I would treat with NA's differently in the seaweed data

Preliminary Analysis and Visualization

This Stage gets some of the basic visualization done. By observing the change in the circumference/length/mass, we get an idea of how the plants grow over time.

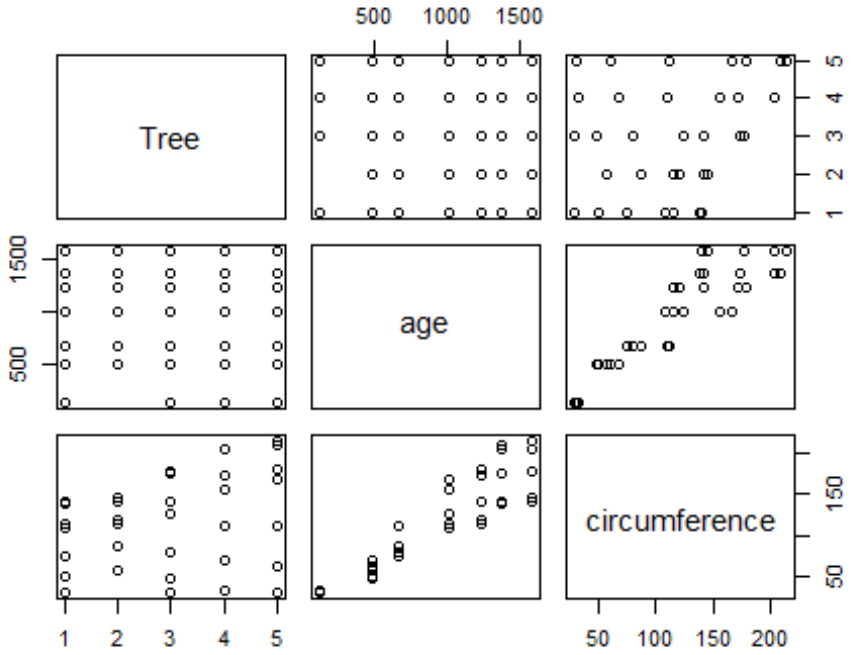
```
data<-Orange
  print(summary(data))

##  Tree       age         circumference
##  3:7   Min.   : 118.0   Min.   : 30.0
##  1:7   1st Qu.: 484.0   1st Qu.: 65.5
##  5:7   Median :1004.0   Median :115.0
##  2:7   Mean   : 922.1   Mean   :115.9
##  4:7   3rd Qu.:1372.0   3rd Qu.:161.5
##        Max.   :1582.0   Max.   :214.0

  glimpse(data)

## Rows: 35
## Columns: 3
## $ Tree          <ord> 1, 1, 1, 1, 1, 1, 1, 2, 2, 2, 2, 2, 2, 2, 3, 3, 3,
3, 3,…
## $ age           <dbl> 118, 484, 664, 1004, 1231, 1372, 1582, 118, 484,
664, 10…
## $ circumference <dbl> 30, 58, 87, 115, 120, 142, 145, 33, 69, 111, 156,
172, 2…

  p<- data %>% ggplot(mapping=aes(x = age, y = circumference, colour =
Tree))+
    geom_point(alpha = 0.6, size = 1.8)+
    geom_line(alpha = 0.6)+
    #geom_abline(intercept = 17.39, slope = 0.1068)+#taken from a linear
model used later, remove
    labs(title = "Circumferences of Orange Trees at different ages",
x="Age(days)", y="Circumference(cm)")
  print(p)
```




```
 print(paste(" Pairwise comparisons of the variables are as follows:"))
```

```
## [1] " Pairwise comparisons of the variables are as follows:"
```

```
 p2<-pairs(data[-1,])
```




```
p3<-cor(data$circumference, data$age)

print(paste("The correlation bwteen age and circumference is:", p3))
```

```
## [1] "The correlation bwteen age and circumference is: 0.913518852891591"
```

From the pairwise comparison, we get a sense of which variables have a relationship at a glance, in this case its age and circumference.

For the seaweed data, I'll also visualize length vs age, to see if it follows similar trends. This general shape is also what I expect from the seaweed data, if we have enough data points. Its sort of a sigmoid curve, growth followed by rapid growth, and finally a plateau.

Calculating Growth Rate

The growth rate is calculated according to the equations:

$$k = \delta Circumference/\delta Days$$

Assuming growth can be described by:

$$FinalSize = InitialSize^{(1+k)t}$$

whereas the time needed for doubling can be calculated as:

$$DoublingTime = log(2)/(1+k)$$

Where k is expressed as a decimal

The following code allows for the calculation of growth rates using the equation above. Also included is a percentage growth rate.

```
data<- Orange %>% group_by(Tree) %>% mutate(prev.age = lag(age, 1, default =
NA))
  data<- data %>% mutate(prev.circumference = lag(circumference, 1, default =
NA))
  data<- data %>% mutate(k = (circumference - prev.circumference)/(age-
prev.age))
  data<- data %>% mutate(percent.k = (k/prev.circumference)*100)
```

From here some further calculation is done to find the mean and standard deviation of the growth rates

```
# work with growth in percentage terms
    perc.growth.rates<-na.omit(data$percent.k)
    perc.growth.rates<-perc.growth.rates[is.finite(perc.growth.rates)]
    avg.percent.growth<-round(mean(perc.growth.rates),4)
    sd.percent.growth<-round(sd(perc.growth.rates),4)


    #work with growth in absolute terms
    growth.rates<-na.omit(data$k)
    growth.rates<-growth.rates[is.finite(growth.rates)]
    avg.growth.rate<-round(mean(growth.rates),4)
    sd.growth.rate<-round(sd(growth.rates),4)


    doubling.time<-(log(2)/log((1+(avg.percent.growth/100))))# this eqn might
need to be modified.
    doubling.time<-round(doubling.time)
    #where % k is used in calculating doubling time, it must be expressed as
a decimal again, divide by 100
```

These results are expressed below:

```
print(paste("The mean growth rate is to be", avg.growth.rate, " cm per day + or -", sd.growth.rate))
```

```
## [1] "The mean growth rate is to be 0.1082  cm per day + or - 0.0776"
```

```
    print(paste("The mean percentage growth rate is to be", avg.percent.growth, " % per day + or -", sd.percent.growth))
```

```
## [1] "The mean percentage growth rate is to be 0.147  % per day + or - 0.1202"
```

```
    print(paste("The estimated mean time to double in size is ", doubling.time, " days(to the nearest day)"))
```

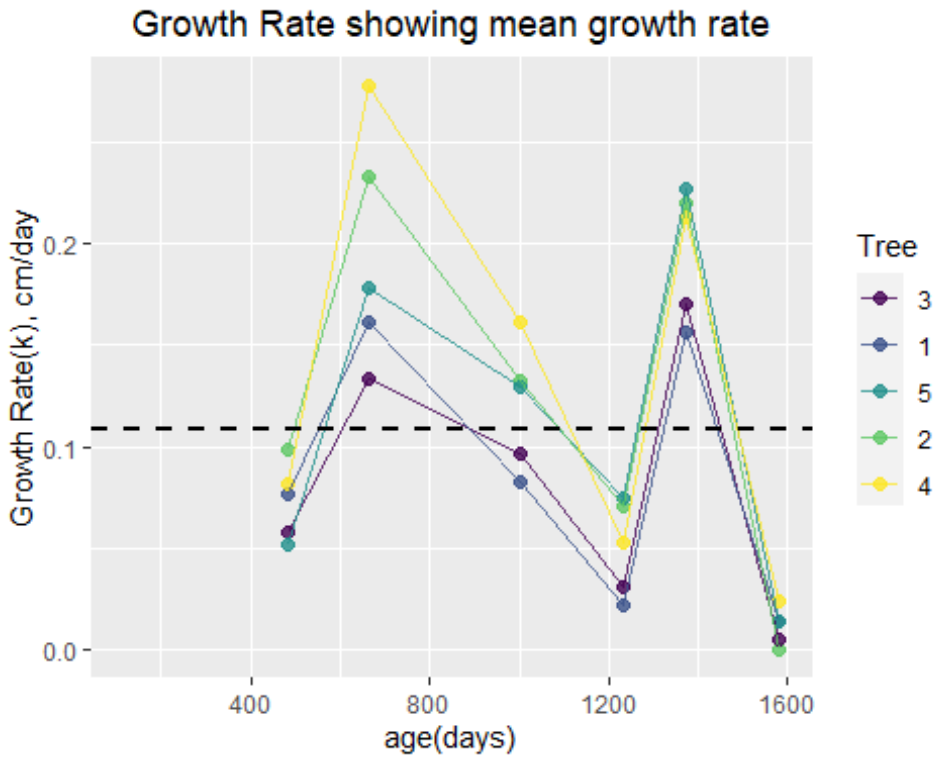
```
## [1] "The estimated mean time to double in size is  472  days(to the nearest day)"
```

Finally, the results are vsualized for ease of understanding/any intutive analysis here

```
    graph.2<-ggplot(data, mapping = aes(x =age, y = k, colour = Tree))+
      geom_point(alpha = 0.8, size = 2)+
      geom_line(alpha = 0.8)+
      geom_hline(yintercept = avg.growth.rate, linetype = "dashed", size = 1)+
      labs(x="age(days)", y = "Growth Rate(k), cm/day", title="Growth Rate showing mean growth rate")+
      theme(plot.title = element_text(hjust = 0.5))
    print(graph.2)
```

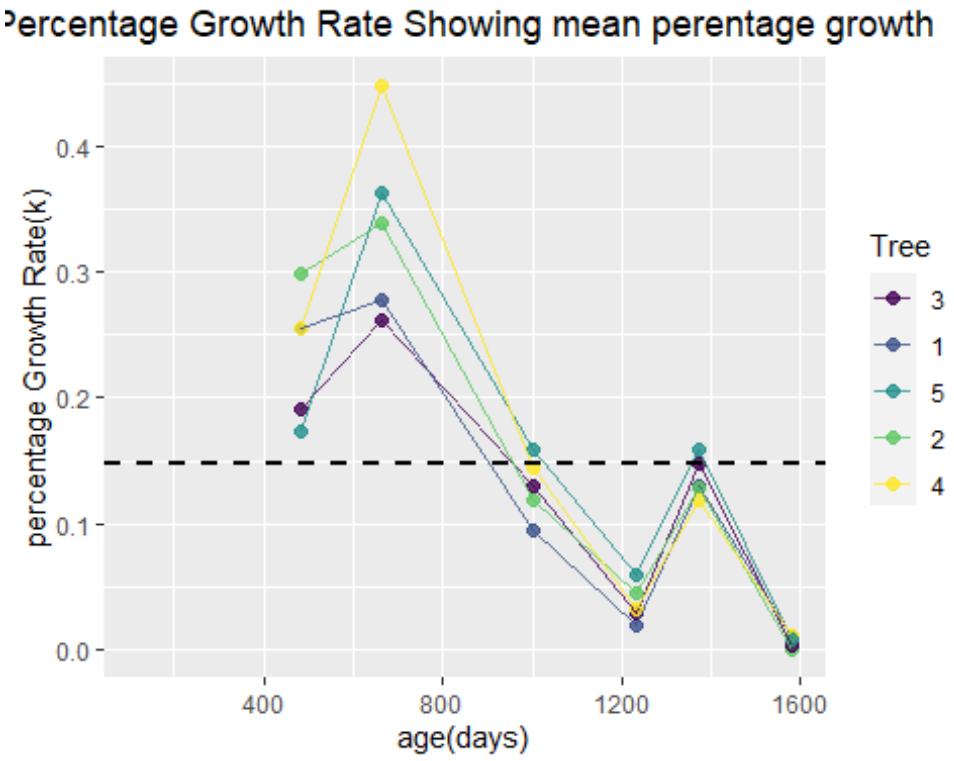
```
## Warning: Removed 5 rows containing missing values (geom_point).
```

```
## Warning: Removed 5 row(s) containing missing values (geom_path).
```



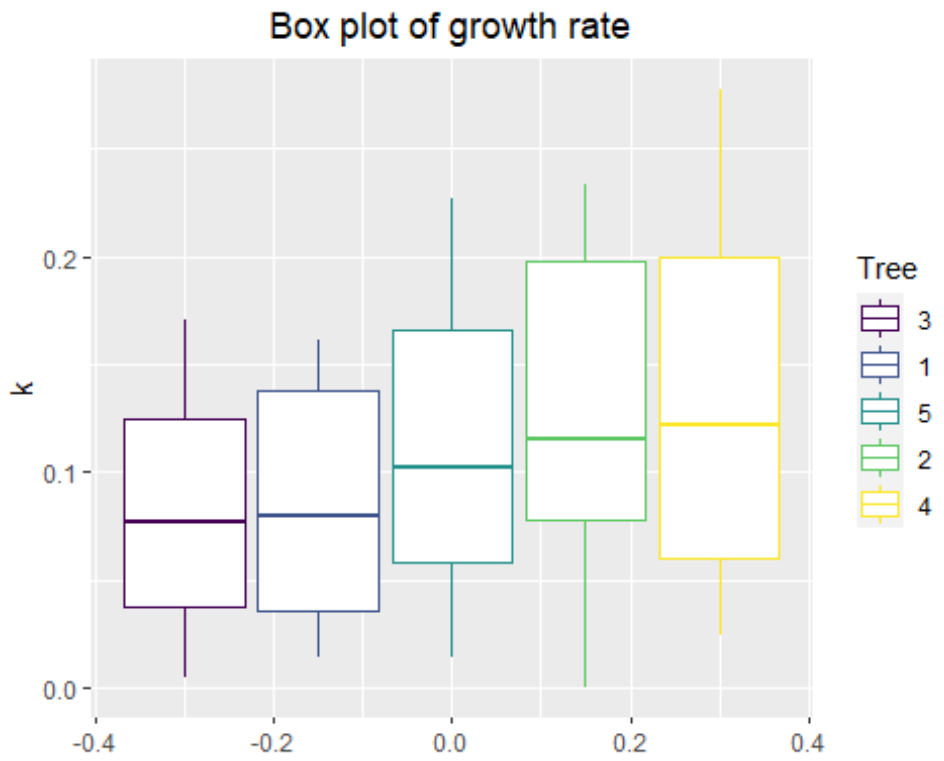

```
    graph.3<-ggplot(data, mapping = aes(x =age, y = percent.k, colour =
Tree))+
      geom_point(alpha = 0.8, size = 2)+
      geom_line(alpha = 0.8)+
      labs(x="age(days)", y = "percentage Growth Rate(k)", title="Percentage
Growth Rate Showing mean perentage growth")+
      geom_hline(yintercept = avg.percent.growth, linetype = "dashed", size =
1)+
      theme(plot.title = element_text(hjust = 0.5))
    print(graph.3)
```

```
## Warning: Removed 5 rows containing missing values (geom_point).
## Removed 5 row(s) containing missing values (geom_path).
```



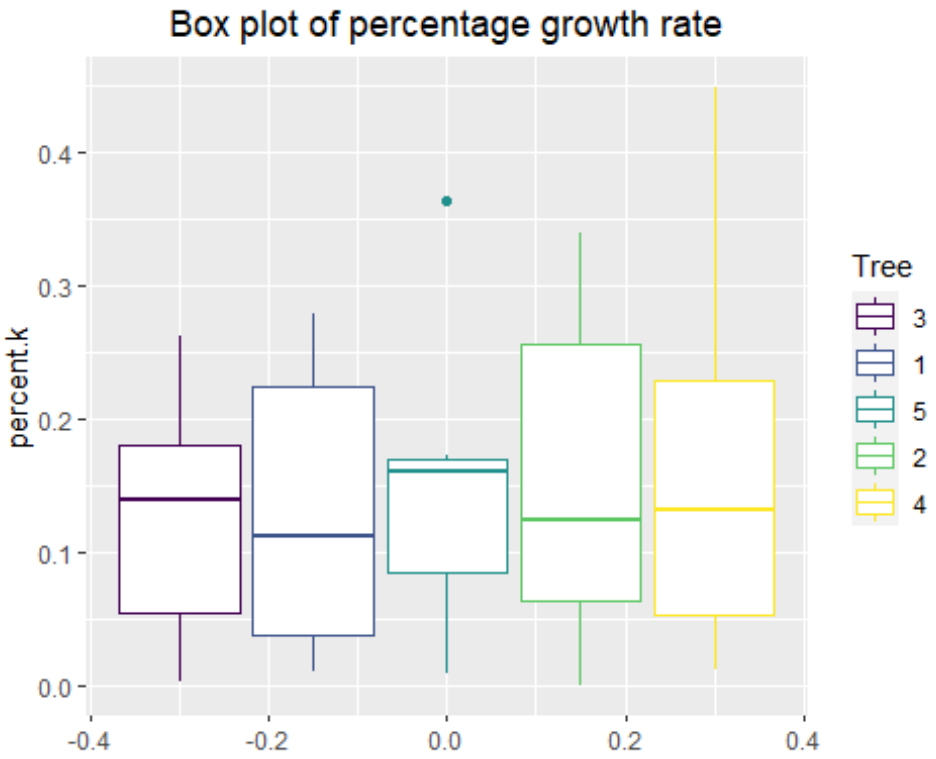

```
    graph.4<-data %>% ggplot(mapping = aes(y = k, colour = Tree))+
      geom_boxplot()+
      labs(title = "Box plot of growth rate")+
      theme(plot.title = element_text(hjust = 0.5))
    print(graph.4)
```

```
## Warning: Removed 5 rows containing non-finite values (stat_boxplot).
```



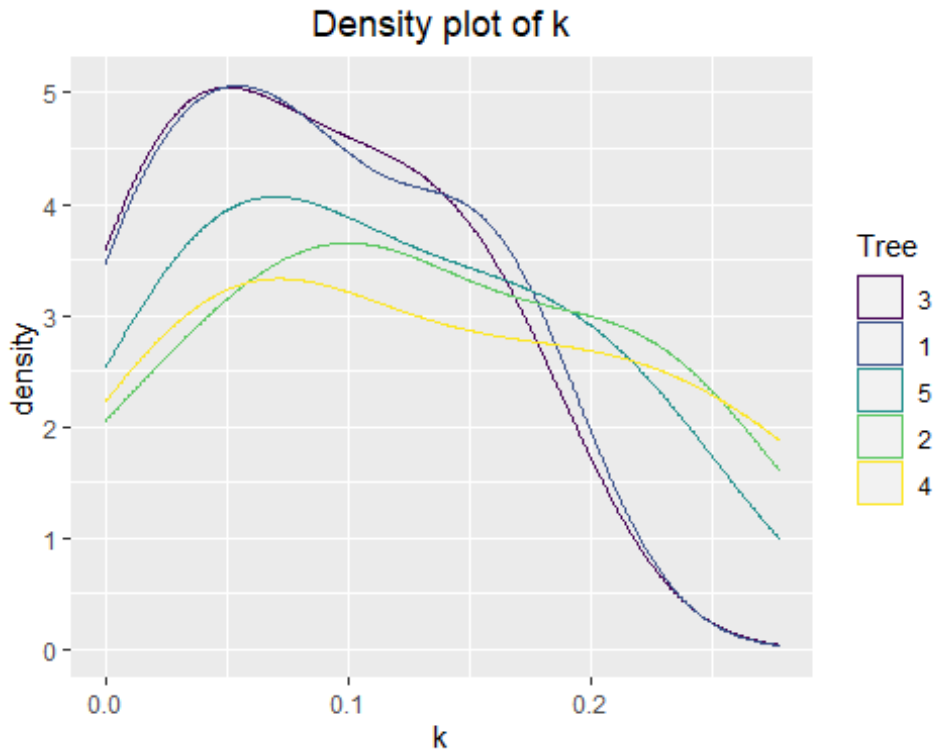

```
    graph.5<-data %>% ggplot(mapping = aes(y = percent.k, colour = Tree))+
      geom_boxplot()+
      labs(title = "Box plot of percentage growth rate")+
      theme(plot.title = element_text(hjust = 0.5))
    print(graph.5)
```

```
## Warning: Removed 5 rows containing non-finite values (stat_boxplot).
```




```
    graph6<-data |> ggplot(mapping = aes(x = k, colour = Tree))+
      geom_density()+
      labs(title = "Density plot of k")+
      theme(plot.title = element_text(hjust = 0.5))
    print(graph6)
```

```
## Warning: Removed 5 rows containing non-finite values (stat_density).
```



Please note these growth rates arent constant as the formula (and even first graph) showed. These are estimates, even without all the deviation and noise.

Note to self As the graph above indicates, growth rates aren't constant, if so graph 1 would be linear. It changes, throughout the period observed. That raises the question of where to take a value for growth rate. My options are basically to take it at the mean or maximum value. The mean(or median) seems to be the best fit.

Signficance between two growth sites

For this section I will run an ANOVA between sites 1 and 2(for both length and mass), to see if there is a significant($p<0.05$) difference between the sites. to be fair i could probably run a 2 sample t-test but meh.

```
#mass.site.aov<-aov(mass ~ Site, data = seaweedData)
#length.site.aov<-aov(length ~ Site, data = seaweedData)
#summary(mass.site.aov)
#summary(length.site.aov)
```

Attempt at Model Fitting

I attempt to see which model accurately describes the data, i attempt from a linear(1st order polynomial) up to a 4th order polynomial. I could throw in some visualization here. In fact i should, But thats down the line.

```
  d<-data[,c(2,3)]
  age2<-d$age^2
  age3<-d$age^3
  age4<-d$age^4
  model1<-lm(circumference ~ age, data = d)
  model2<-lm(circumference ~ age + age2,data = d)
  model3<-lm(circumference ~ age + age2 + age3,data = d)
  model4<-lm(circumference ~ age + age2 + age3 + age4,data = d)
  model1.summary<-aov(model1)
  model2.summary<-aov(model2)
  model3.summary<-aov(model3)

 # view the models for comparing whats needed and whats overfitting
  print("Summary of the linear model")
```

```
## [1] "Summary of the linear model"
```

```
  print(summary(model1.summary))
```

```
##             Df Sum Sq Mean Sq F value   Pr(>F)
## age          1  93772   93772   166.4 1.93e-14 ***
## Residuals   33  18595     563
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
  print("Summary of the quadratic model")
```

```
## [1] "Summary of the quadratic model"
```

```
  print(summary(model2.summary))
```

```
##             Df Sum Sq Mean Sq F value   Pr(>F)
## age          1  93772   93772 163.956 3.89e-14 ***
## age2         1    293     293   0.512    0.479
## Residuals   32  18302     572
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
  print("Summary of the cubic model")
```

```
## [1] "Summary of the cubic model"
```

```
  print(summary(model3.summary))
```

```
##             Df Sum Sq Mean Sq F value  Pr(>F)
## age          1  93772   93772 166.200 5.4e-14 ***
```

```
## age2         1    293     293   0.519   0.477
## age3         1    811     811   1.438   0.240
## Residuals   31  17490     564
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
  print("Summary of the best model")
```

```
## [1] "Summary of the best model"
```

```
  print(summary(model1))# model 1 in this case, but whichever model is best
```

```
##
## Call:
## lm(formula = circumference ~ age, data = d)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -46.310 -14.946  -0.076  19.697  45.111
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 17.399650   8.622660   2.018   0.0518 .
## age          0.106770   0.008277  12.900 1.93e-14 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 23.74 on 33 degrees of freedom
## Multiple R-squared:  0.8345, Adjusted R-squared:  0.8295
## F-statistic: 166.4 on 1 and 33 DF,  p-value: 1.931e-14
```

In this case only age is needed and a linear model is the best fit. The addition of a 2nd(and hence 3rd order polynomial) does not explain significantly more.

Length - Mass Regression

This section aims to see how accurately length and mass can be used as predictors of each other. the aim is, hopefully, that with a high enough relationship, we can measure out length, which is easier and less labour/time intensive, to get a decent approximation of biomass(since product will be sold per unit mass.)

```
#length.mass.model<-dataset %>% lm(mass ~ length)
#View(summary(length.mass.model))
```

The first thing to look at is the R^2 value, if it is sufficiently high, we can use the intercept and coefficients to come up with a decent equation to relate mass to length.

Concluding Comments

There is almost always room for improvement.

- Figure out how im treating with `NA` where seaweed is lost
- Add in the Model Fittings
- There is stuff in place for the length-mass regression and ANOVA between sites(but it needs real data, the `Orange` data is not a perfect fit)

Information of Version of Software Used

```
sessionInfo()
```

```
## R version 4.2.1 (2022-06-23 ucrt)
## Platform: x86_64-w64-mingw32/x64 (64-bit)
## Running under: Windows 10 x64 (build 22000)
##
## Matrix products: default
##
## locale:
## [1] LC_COLLATE=English_United States.utf8
## [2] LC_CTYPE=English_United States.utf8
## [3] LC_MONETARY=English_United States.utf8
## [4] LC_NUMERIC=C
## [5] LC_TIME=English_United States.utf8
##
## attached base packages:
## [1] stats     graphics  grDevices utils     datasets  methods   base
##
## other attached packages:
##  [1] ggpubr_0.4.0    gifski_1.6.6-1  gganimate_1.0.8 forcats_0.5.2
##  [5] stringr_1.4.1   dplyr_1.0.10    purrr_0.3.4     readr_2.1.2
##  [9] tidyr_1.2.1     tibble_3.1.8    ggplot2_3.3.6   tidyverse_1.3.2
## [13] lubridate_1.8.0 readxl_1.4.1
##
## loaded via a namespace (and not attached):
##  [1] prettyunits_1.1.1   assertthat_0.2.1    digest_0.6.29
##  [4] utf8_1.2.2          R6_2.5.1            cellranger_1.1.0
##  [7] backports_1.4.1     reprex_2.0.2        evaluate_0.16
## [10] highr_0.9           httr_1.4.4          pillar_1.8.1
## [13] rlang_1.0.5         progress_1.2.2      googlesheets4_1.0.1
## [16] rstudioapi_0.14     car_3.1-0           rmarkdown_2.16
## [19] labeling_0.4.2      googledrive_2.0.0   munsell_0.5.0
## [22] broom_1.0.1         compiler_4.2.1      modelr_0.1.9
## [25] xfun_0.32           pkgconfig_2.0.3     htmltools_0.5.3
## [28] tidyselect_1.1.2    viridisLite_0.4.1   fansi_1.0.3
## [31] crayon_1.5.1        tzdb_0.3.0          dbplyr_2.2.1
## [34] withr_2.5.0         grid_4.2.1          jsonlite_1.8.0
## [37] gtable_0.3.1        lifecycle_1.0.2     DBI_1.1.3
## [40] magrittr_2.0.3      scales_1.2.1        carData_3.0-5
## [43] cli_3.3.0           stringi_1.7.8       ggsignif_0.6.3
## [46] farver_2.1.1        fs_1.5.2            xml2_1.3.3
## [49] ellipsis_0.3.2      generics_0.1.3      vctrs_0.4.1
## [52] tools_4.2.1         glue_1.6.2          tweenr_2.0.2
## [55] hms_1.1.2           abind_1.4-5         fastmap_1.1.0
## [58] yaml_2.3.5          colorspace_2.0-3    gargle_1.2.1
## [61] rstatix_0.7.0       rvest_1.0.3         knitr_1.40
## [64] haven_2.5.1
```