FES 524 Winter 2018 Lab 4

Contents

Factorial Treatment Designs with Random Effects

In lab 4, you will learn to fit a mixed model with more than one categorical explanatory variable and learn how to test for and interpret interactions between categorical explanatory variables.

Load R packages needed today

```
library(dplyr)
library(ggplot2)
library(nlme)
library(gmodels)
```

Read in the dataset

We will be working with the dataset `lab4.example.stock.txt`, so make sure you have this file and have changed your working directory appropriately. As usual when we read in a dataset we'll take a look at the structure. We will also do any factoring/relabeling of factors as needed.

```
growthdata = read.table("lab4.example.stock.txt", header = TRUE)
head(growthdata) # First six lines
```

```
  nursery plantdate   stock growth
1       1      Jan2 contain   1.97
2       2      Jan2 contain   1.18
3       3      Jan2 contain   2.48
4       4      Jan2 contain   2.31
5       5      Jan2 contain   2.30
6       1      Jan2  barert   1.99
```

Check the structure of the dataset in the Environment pane.

```
'data.frame':	30 obs. of  4 variables:
 $ nursery  : int  1 2 3 4 5 1 2 3 4 5 ...
 $ plantdate: Factor w/ 3 levels "Feb25","Jan2",..: 2 2 2 2 2 2 2 2 2 2 ...
 $ stock    : Factor w/ 2 levels "barert","contain": 2 2 2 2 2 1 1 1 1 1 ...
 $ growth   : num  1.97 1.18 2.48 2.31 2.3 1.99 1.52 2.55 2.73 2.79 ...
```

```
# Note that the factor plantdate has the levels in a non-useful order. Let's
    # change the order so the levels are ordered chronologically
    # We'll relabel while we're at it
growthdata$plantdate = factor(growthdata$plantdate,
                 levels = c("Jan2","Jan28","Feb25"),
                 labels = c("Jan 2","Jan 28","Feb 25"))

# Make nursery a factor variable instead of integer
growthdata$nursery = factor(growthdata$nursery)
```

Check the structure of the dataset to see the changes.

```
'data.frame':	30 obs. of  4 variables:
 $ nursery  : Factor w/ 5 levels "1","2","3","4",..: 1 2 3 4 5 1 2 3 4 5 ...
 $ plantdate: Factor w/ 3 levels "Jan 2","Jan 28",..: 1 1 1 1 1 1 1 1 1 1 ...
 $ stock    : Factor w/ 2 levels "barert","contain": 2 2 2 2 2 1 1 1 1 1 ...
 $ growth   : num  1.97 1.18 2.48 2.31 2.3 1.99 1.52 2.55 2.73 2.79 ...
```

Initial data exploration

We'll start by calculating summary statistics by group and creating some initial figures to explore our dataset graphically.

Check if two factors are crossed

When we have two categorical variables, we need to check if the factors are *crossed* or not. To be crossed, every level of one categorical variable occurs with every level of another categorical variable. For example, in this study we need to check that every level of transplanting date occurs with every level of stock type and *vice versa*. In designed experiments, fixed effects factors are often crossed and there is specific interest, as in today's example, in how the combination of factors affects our mean response variable. Such studies are referred to as having *factorial* treatment designs. If the factors in a study are not crossed, as can sometimes happen in observational studies, it is difficult to impossible to answer questions about the combined effect of the factors.

Here we can use `xtabs` and/or functions from **dplyr** to check if each level of `plantdate` occurs with every level of `stock`.

```
# Examine the number of observations in the groups
xtabs(~stock + plantdate, growthdata)
```

```
         plantdate
stock     Jan 2 Jan 28 Feb 25
  barert      5      5      5
  contain     5      5      5
```

With **dplyr** we can get the number of observations in each combined group along with the rest of our summary statistics of interest to get an idea of what the dataset looks like.

```
( sumdat = growthdata %>%
        group_by(plantdate, stock) %>%
        summarise(n = n(),
                mean = round(mean(growth), 2),
                sd = round(sd(growth), 2),
                min = min(growth),
                max = max(growth) ) )
```

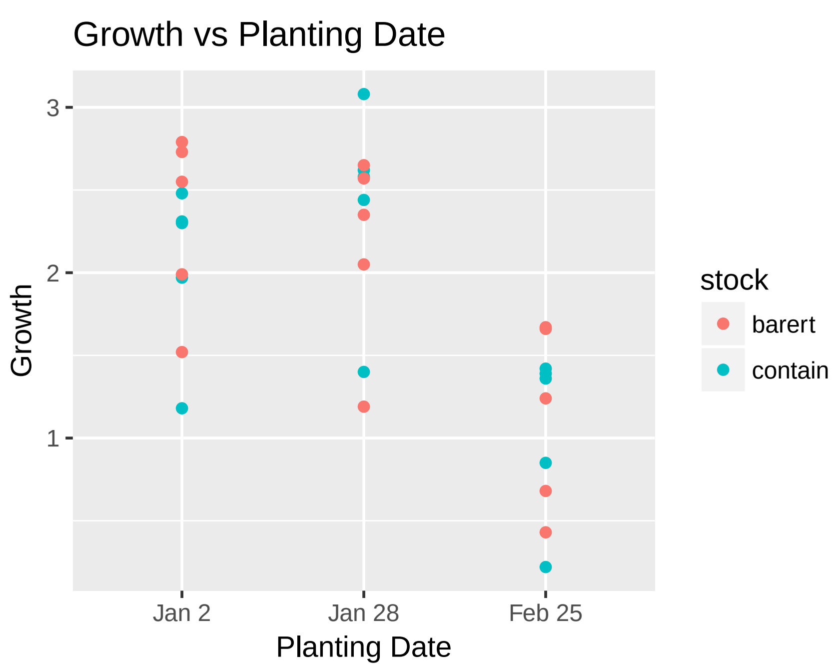
```
# A tibble: 6 x 7
# Groups: plantdate [?]
  plantdate stock       n  mean    sd   min   max
  <fct>     <fct>   <int> <dbl> <dbl> <dbl> <dbl>
1 Jan 2     barert      5  2.32 0.550 1.52   2.79
2 Jan 2     contain     5  2.05 0.520 1.18   2.48
3 Jan 28    barert      5  2.16 0.590 1.19   2.65
4 Jan 28    contain     5  2.42 0.620 1.40   3.08
5 Feb 25    barert      5  1.14 0.570 0.430  1.67
6 Feb 25    contain     5  1.05 0.520 0.220  1.42
```

Graphical data exploration

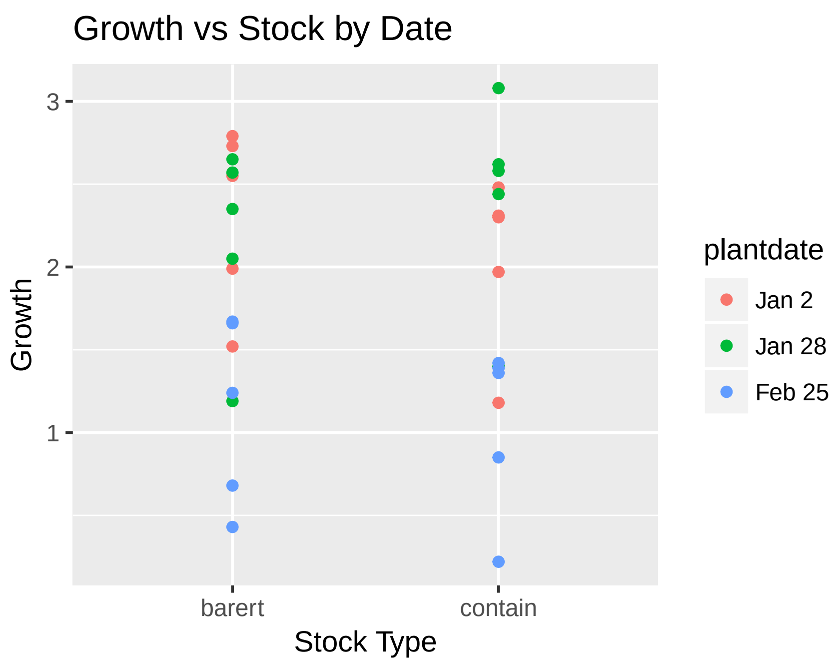
We'll explore our data, looking at scatterplots of our response versus each other variable (`plantdate`, `stock`, `nursery`). We'll use color and shapes to help explore patterns.

While `nursery` isn't of specific research interest, we are plotting it in order to understand the dataset better. We're looking to see if the pattern among factor levels is similar among all nurseries. If this wasn't the case, this would be a discussion point when talking about future research with a similar study design.

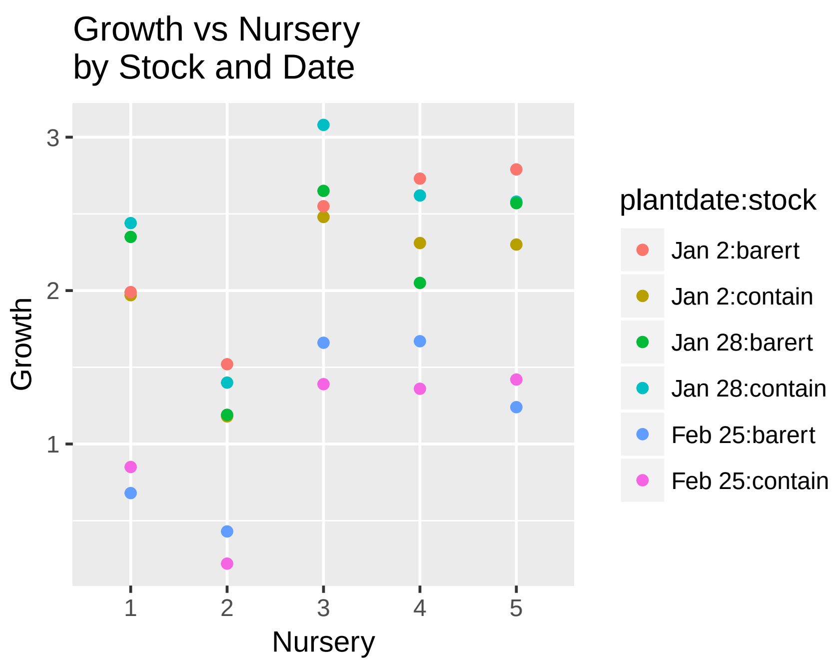
```r
# Plot the raw data

# Scatterplot of growth vs planting date,
    # color by stock type
qplot(plantdate, growth, color = stock, data = growthdata,
      xlab = "Planting Date",
      ylab = "Growth",
      main = "Growth vs Planting Date")
```



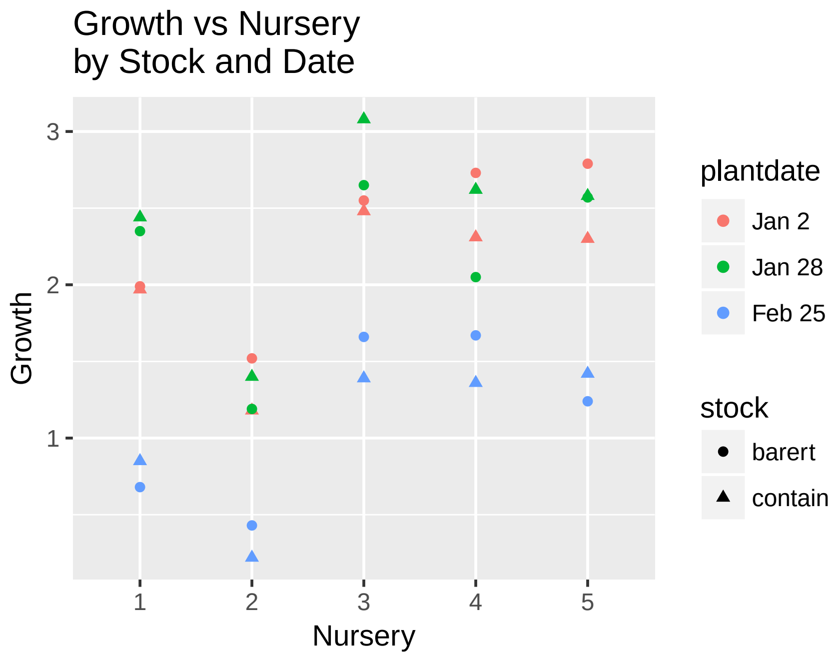

```r
# Scatter plot of growth vs stock,
    # color by plantdate
qplot(stock, growth, color = plantdate, data = growthdata,
      xlab = "Stock Type",
      ylab = "Growth",
      main = "Growth vs Stock by Date")
```




```r
# Scatter plot of growth vs nursery, coloring by combined factor levels
# Is the observed pattern among treatment combinations
    # more or less the same among nurseries?
qplot(nursery, growth, color = plantdate:stock, data = growthdata,
      xlab = "Nursery",
      ylab = "Growth",
      main = "Growth vs Nursery\nby Stock and Date")
```




```r
# An alternative is to use colors for date and shapes for stock
qplot(nursery, growth, color = plantdate, shape = stock, data = growthdata,
      xlab = "Nursery",
      ylab = "Growth",
      main = "Growth vs Nursery\nby Stock and Date")
```



The interaction plot

One of the big questions that we'll need to answer is whether there is evidence of an interaction between the two factors, stock type and transplanting date. An interaction means that the effect of the levels of one factor *depends* on the level of the other factor (and *vice versa*). For example, here it would mean that the difference in mean growth between the bare root and the container stock types differed across the three planting dates.

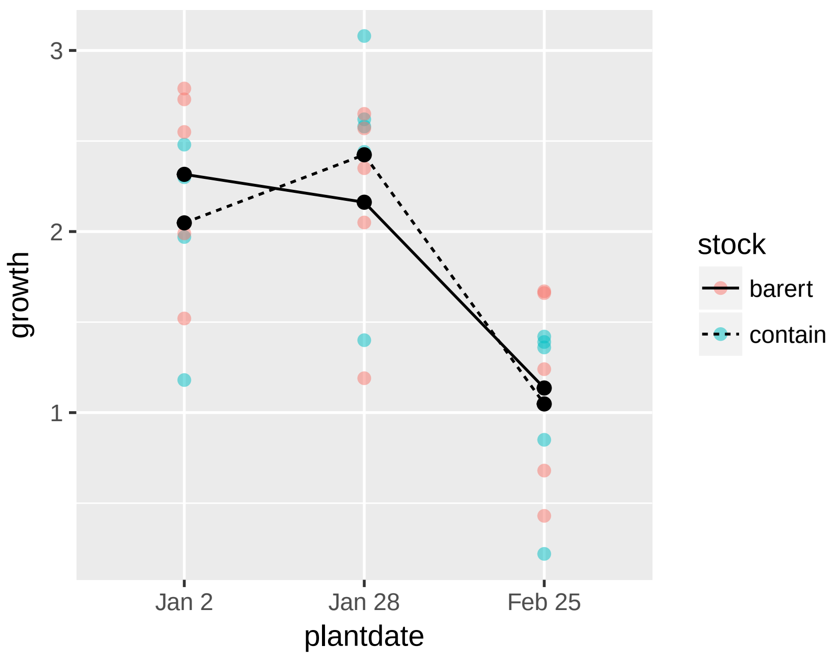
An interaction plot of the means can help us visualize the possibility of a detectable interaction when exploring a dataset. Note that an interaction plot is an exploratory graphic. They do not have error bars on them and so are not appropriate for making inference unless additional information is added to them. We'll add the raw data onto the graphic to make it clearer that this is an exploratory plot, not an inferential one.

To make an interaction plot, we need to calculate the group means for each combination of factor levels. We then make a scatterplot show the mean response vs one of the categorical explanatory variables. The second explanatory variable is represented by linetypes or colors or shapes. We connect the means associated with specific levels of the second factor with lines to make the interaction plot easier to read. Making use of colors/shapes/line types helps us distinguish between groups.

We will make this plot using `ggplot` directly, as the plot is a little too complicated to be considered a "quick" plot.

The key for plotting a summary statistic from the data like the mean is to use `stat_summary`. Notice we are using two `geoms`, points and lines, and so use `stat_summary` twice. Note also that we could have just plotted the means from the `sumdat` dataset we already made rather than using `stat_summary` for plotting the original dataset. It is often simpler in **ggplot2** to create and plot a summary dataset than to figure out how to use `stat_summary` properly.

```
ggplot(growthdata, aes(x = plantdate, y = growth,
                   group = stock, linetype = stock)) +
    geom_point(alpha = .5, size = 1.75, aes(color = stock) ) + # Add raw data
    stat_summary(fun.y = mean, geom = "point", size = 2) + # Add points for means
    stat_summary(fun.y = mean, geom = "line") # Connect points with lines
```



Fitting a factorial linear mixed model with `lme`

We will fit a linear mixed model using `lme` from package **nlme**, where `nursery` is a random effect and `plantdate` and `stock` are fixed effects. Remember that our observation-level random effect, the residual term, is included automatically in `lme`.

We will include both categorical explanatory variables and a term for the interaction between `plantdate` and `stock` in the model. We use a colon (`:`) between the two variable names to indicate an interaction in `lme`. We add explanatory variables as fixed effects using the plus sign, `+`.

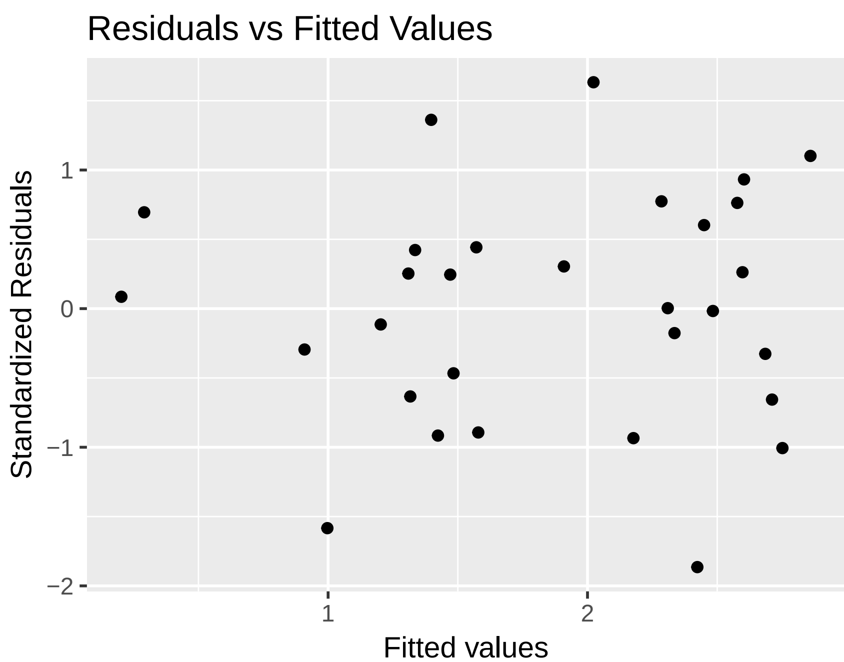
```
model1 = lme(growth ~ stock + plantdate + stock:plantdate,
           random = ~1|nursery, data = growthdata)
```

Checking model assumptions

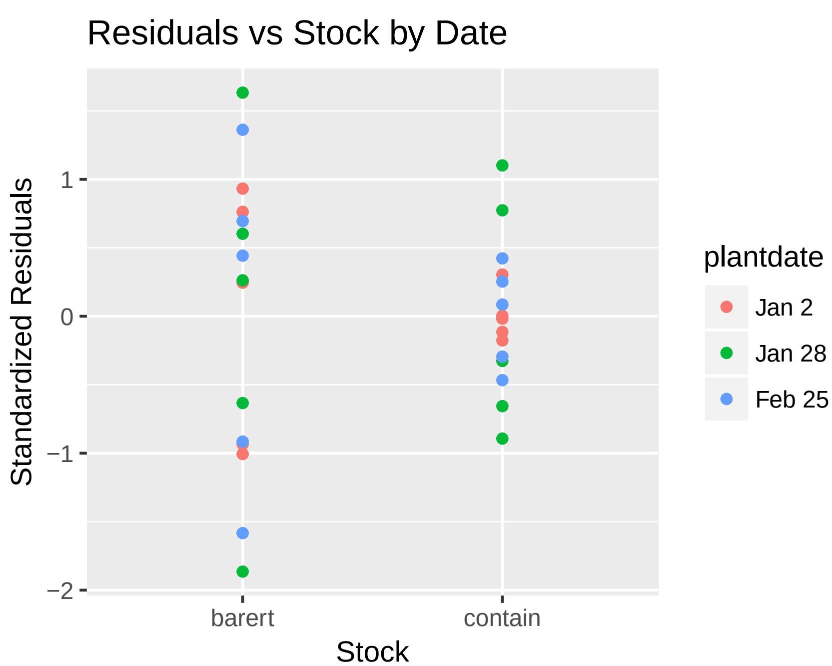
As always, we'll need to check the assumptions of the model using the residual and fitted values from the model. We can add these to the dataset `growthdata`, and then plot the residuals vs the fitted values, the residuals vs each fixed effect variable, and check the normality of the residuals with a quantile-quantile plot.

This week we will be plotting with *standardized* residuals (which are often called *pearson* residuals in R). Remember that the default residual type for `lme` objects is *raw* residuals. We use the `type` argument to get the Pearson residuals. See the help page for `residuals.lme` for a reminder.

```
growthdata$res = residuals(model1, type = "pearson")
growthdata$fit = fitted(model1)

# Plot residuals vs fitted values
qplot(fit, res, data = growthdata,
     main = "Residuals vs Fitted Values",
     xlab = "Fitted values",
     ylab = "Standardized Residuals")
```



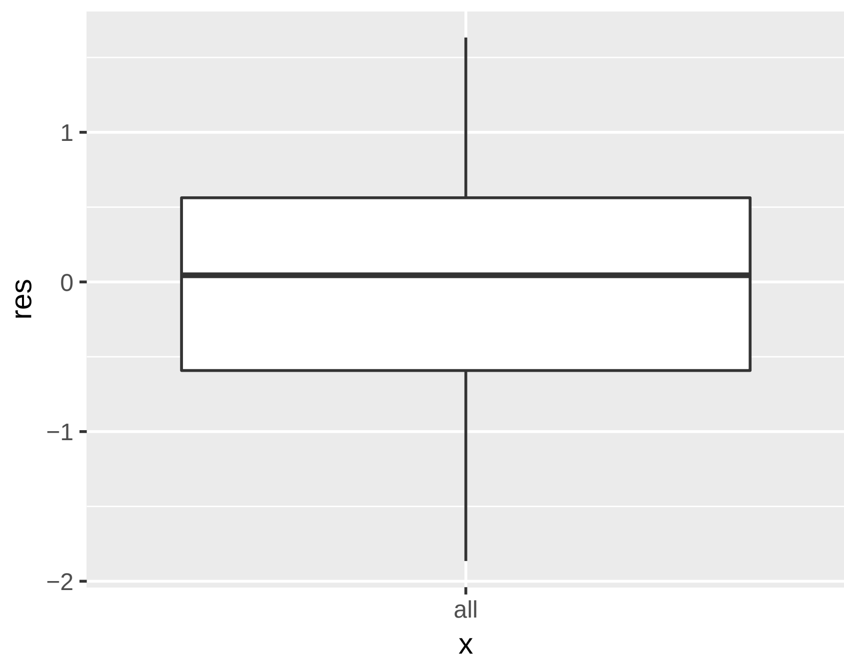

```r
# Residuals vs stock type
qplot(stock, res, color = plantdate, data = growthdata,
      xlab = "Stock",
      ylab = "Standardized Residuals",
      main = "Residuals vs Stock by Date")
```




```r
# Residuals vs plantdate
qplot(plantdate, res, color = stock, data = growthdata,
      xlab = "Planting Date",
      ylab = "Standardized Residuals",
      main = "Residuals vs Date by Stock")
```

I'm using a boxplot to check for residual normality/symmetry today, but you can always use a histogram or a quantile-quantile plot here instead.

```
# Check normality of residuals with normal probability plot
qplot(x = "all", y = res,
      data = growthdata, geom = "boxplot")
```



Extending the basic linear mixed model

You should have noticed that the variation between the container stock type and the bare root stock type looks like it could be a bit different. Now, the magnitude of the difference is small enough (based on the spread of the residuals) that I am not overly concerned, particularly because we have a balanced design (i.e., same number of samples in each group). See section 3.2.3 in the *Statistical Sleuth* for a discussion about the effect of differing standard deviations among groups.

However, I want to take a moment to show you that we have options for extending the basic linear mixed model. This is so you can see that such tools exist - it is not something we are expecting you to learn this week or use on your assignment.

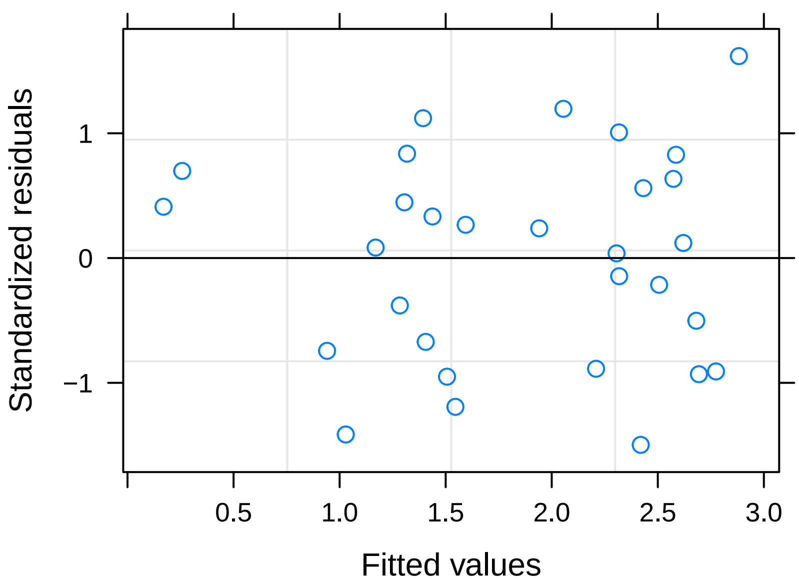
Relaxing the assumption of equal variances in `lme`

We can relax the assumption that the variances are constant within the `stock` groups by adding the `weights` argument with the `varIdent` function. The `varIdent` function is a **nlme** function specifically for relaxing the assumption of constant variance among groups.

```
model2 = lme(growth ~ stock + plantdate + stock:plantdate,
             random = ~1|nursery,
             weights = varIdent(form = ~1|stock),
             data = growthdata)
```

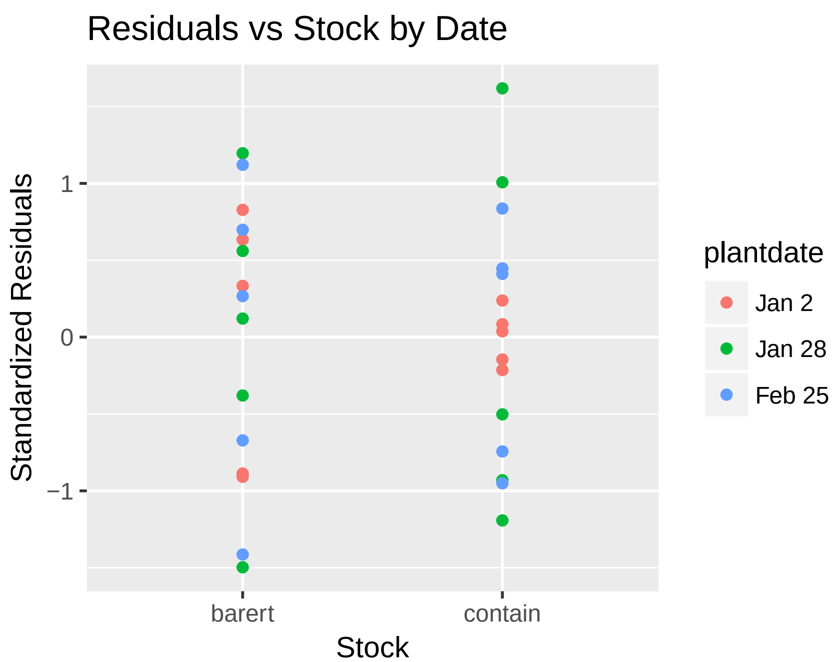
Let's take a look at these residual plots for this model. Using Pearson residuals is very important once you start using the `weights` argument. If you use the raw residuals you will not be able to judge how well the model fits.

```
# Compute and save pearson residuals
growthdata$res2 = residuals(model2, type = "pearson")

# Residuals vs fitted values plots
plot(model2)
```

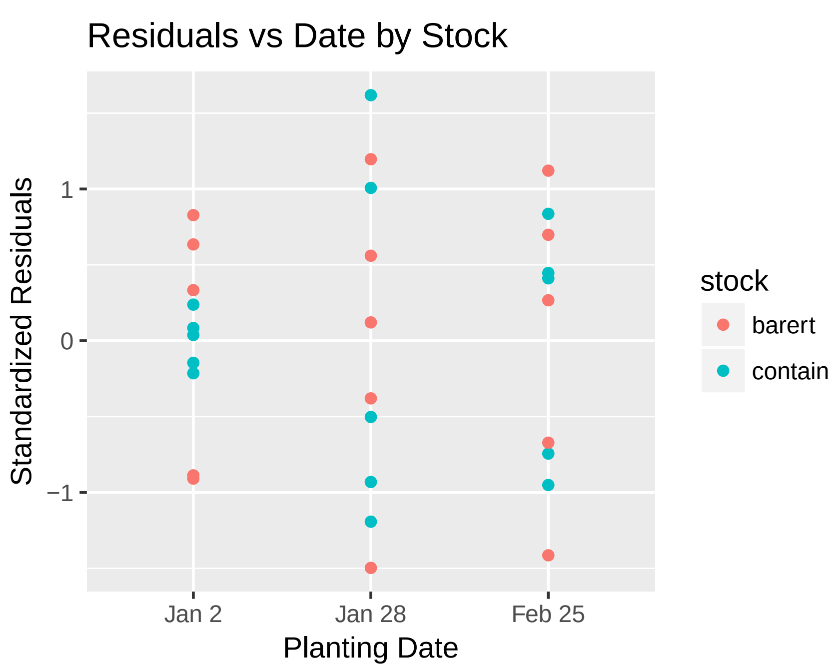


Not surprisingly, the plot of the residuals vs `stock` is the one that has really changed substantially. After allowing for nonconstant variances among levels of this variable, the residual spread is very similar between the two `stock` levels.

```
# Residuals vs stock type
# This is the residual plot that looks fairly different now
qplot(stock, res2, color = plantdate, data = growthdata,
      xlab = "Stock",
      ylab = "Standardized Residuals",
      main = "Residuals vs Stock by Date")
```




```
# Residuals vs plantdate
    # Notice this plot looks more-or-less the same
qplot(plantdate, res2, color = stock, data = growthdata,
      xlab = "Planting Date",
      ylab = "Standardized Residuals",
      main = "Residuals vs Date by Stock")
```



Take a look at the summary of this model, as well. Since we allowed different variances per group, we now have a section called **Variance function** in the summary output. This section shows us the ratios of the variances of the groups. In this case, the bare root group had a standard deviation estimated to be about 2.02 times larger than the container group.

```
summary(model2)
```

```
Linear mixed-effects model fit by REML
 Data: growthdata
       AIC      BIC    logLik
```

```
  28.88107 39.48356 -5.440536

Random effects:
 Formula: ~1 | nursery
        (Intercept)  Residual
StdDev:   0.5356277 0.1222186

Variance function:
 Structure: Different standard deviations per stratum
 Formula: ~1 | stock
 Parameter estimates:
 contain   barert
1.000000 2.018806
Fixed effects: growth ~ stock + plantdate + stock:plantdate
                              Value Std.Error DF   t-value p-value
(Intercept)                   2.316 0.2637330 20  8.781608  0.0000
stockcontain                 -0.268 0.1231389 20 -2.176404  0.0417
plantdateJan 28              -0.154 0.1560494 20 -0.986867  0.3355
plantdateFeb 25              -1.180 0.1560494 20 -7.561708  0.0000
stockcontain:plantdateJan 28  0.530 0.1741447 20  3.043446  0.0064
stockcontain:plantdateFeb 25  0.180 0.1741447 20  1.033623  0.3136
 Correlation:
                             (Intr) stckcn plnJ28 plnF25 st:J28
stockcontain                 -0.375
plantdateJan 28              -0.296  0.634
plantdateFeb 25              -0.296  0.634  0.500
stockcontain:plantdateJan 28  0.265 -0.707 -0.896 -0.448
stockcontain:plantdateFeb 25  0.265 -0.707 -0.448 -0.896  0.500

Standardized Within-Group Residuals:
       Min         Q1        Med         Q3        Max
-1.4972045 -0.7257183  0.1024148  0.6160325  1.6184330

Number of Observations: 30
Number of Groups: 5
```

Model results

Overall F-tests

While each of us would have had to decide which model to use for inference, I am going to go back to the original linear mixed model, `model1`. Once we've decided on a model where the assumptions have been reasonably met, we can report any model results of interest from `anova` and/or `summary`.

Make note of statistical evidence for the interaction in the overall F-tests from `anova`.

```r
anova(model1)
```

```
                numDF denDF   F-value p-value
(Intercept)         1    20  61.11828  <.0001
stock               1    20   0.18371  0.6728
plantdate           2    20 109.89596  <.0001
stock:plantdate     2    20   4.53045  0.0238
```

```r
summary(model1)
```

```
Linear mixed-effects model fit by REML
 Data: growthdata
       AIC      BIC    logLik
  31.52795 40.95238 -7.763976
```

```
Random effects:
 Formula: ~1 | nursery
        (Intercept)  Residual
StdDev:   0.5244308 0.2002016

Fixed effects: growth ~ stock + plantdate + stock:plantdate
                              Value Std.Error DF   t-value p-value
(Intercept)                   2.316 0.2510412 20  9.225579  0.0000
stockcontain                 -0.268 0.1266186 20 -2.116593  0.0470
plantdateJan 28              -0.154 0.1266186 20 -1.216251  0.2380
plantdateFeb 25              -1.180 0.1266186 20 -9.319327  0.0000
stockcontain:plantdateJan 28  0.530 0.1790657 20  2.959807  0.0077
stockcontain:plantdateFeb 25  0.180 0.1790657 20  1.005218  0.3268
 Correlation:
                             (Intr) stckcn plnJ28 plnF25 st:J28
stockcontain                 -0.252
plantdateJan 28              -0.252  0.500
plantdateFeb 25              -0.252  0.500  0.500
stockcontain:plantdateJan 28  0.178 -0.707 -0.707 -0.354
stockcontain:plantdateFeb 25  0.178 -0.707 -0.354 -0.707  0.500

Standardized Within-Group Residuals:
        Min          Q1         Med          Q3         Max
-1.86471825 -0.59191154  0.04450435  0.56257609  1.63314505

Number of Observations: 30
Number of Groups: 5
```

Estimating specific group differences in mean response

The specific research questions of interest in this study are:

1. How is growth affected when trees are planted in late February versus late January?
2. Which planting date/stock type combinations have different growth compared to the control group (Jan 2, bare root)?

Calculate the size of the family of comparisons

While the first question is only about transplanting date, we'll need to answer this question separately for each stock type for two reasons: first, we have statistical evidence of an interaction; second (and more importantly), based on the wording of the research question *the researchers thought the combination of two factors would affect growth differently than each factor alone.*

So we will be doing a total of seven comparisons today to answer the two research questions. We can use the Bonferroni correction to control the Type I error rate for the whole family of comparisons (i.e., control the familywise error rate). Again, Bonferroni is very conservative, especially as we start doing many comparisons, and I would recommend you explore other options such as the false discovery rate (FDR) method in your own work if you are doing many comparisons.

Here are the confidence interval size we would need to make to control the familywise error rate using the Bonferroni correction. If you use p-values for each comparison in your write-up, remember to compare them to the Bonferroni-adjusted alpha value.

```r
# Bonferroni correction for all CI, 1 - alpha/k
1 - (.05/7)
```

```
[1] 0.9928571
```

Writing the linear combinations of coefficients for group means

We'll be using the function `estimable` from package **gmodels** to get estimates for our comparisons that are needed to answer the research questions, and will need to write out the appropriate linear combinations of coefficients ourselves.

The first thing to do is look at the fixed effects summary output of the model again. We will use the order of the output of the coefficients in the summary to help us write out our vectors of 0's and 1's that represent the mean growth for every `plantdate`/`stock` combination. Review your lecture notes for an explanation of how to do this.

```
summary(model1)$tTable
```

```
                              Value Std.Error DF   t-value      p-value
(Intercept)                   2.316 0.2510412 20  9.225579 1.205324e-08
stockcontain                 -0.268 0.1266186 20 -2.116593 4.702800e-02
plantdateJan 28              -0.154 0.1266186 20 -1.216251 2.380494e-01
plantdateFeb 25              -1.180 0.1266186 20 -9.319327 1.021218e-08
stockcontain:plantdateJan 28  0.530 0.1790657 20  2.959807 7.744803e-03
stockcontain:plantdateFeb 25  0.180 0.1790657 20  1.005218 3.267999e-01
```

```
# Writing out linear combinations of coefficients for each group mean
jan2b = c(1, 0, 0, 0, 0, 0)
jan2c = c(1, 1, 0, 0, 0, 0)
jan28b = c(1 ,0, 1, 0, 0, 0)
jan28c = c(1, 1, 1, 0, 1, 0)
feb25b = c(1, 0, 0, 1, 0, 0)
feb25c = c(1, 1, 0, 1, 0, 1)
```

Using `estimable` for the comparisons of interest

Once we have the vectors that represent the means for each factor combination, we can use these to create vectors that represent the comparisons that will answer our research questions. To answer the question about differences in mean growth between late January and late February, we will perform two comparisons (one for each stock type).

```
# Comparisons to answer the first question
j28minf_b = jan28b - feb25b
j28minf_c = jan28c - feb25c

# Make the estimates of the differences, remembering to stack the vectors with rbind()
( compj28f = estimable(model1, rbind(j28minf_b, j28minf_c), conf.int = 0.993) )
```

```
          Estimate Std. Error   t value DF     Pr(>|t|)  Lower.CI Upper.CI
j28minf_b    1.026  0.1266186  8.103076 20 9.559372e-08 0.6455377 1.406462
j28minf_c    1.376  0.1266186 10.867283 20 7.675653e-10 0.9955377 1.756462
```

To answer our second question, about each group compared to the control, we will use the factor combination group mean vectors to create vectors that represent the five comparisons.

```
# Comparisons to answer the second questions
jan2c.vs.cont = jan2c - jan2b
jan28c.vs.cont = jan28c - jan2b
jan28b.vs.cont = jan28b - jan2b
feb25c.vs.cont = feb25c - jan2b
feb25b.vs.cont = feb25b - jan2b

# Stack the vectors with rbind() for use in estimable
compcont = rbind(jan2c.vs.cont, jan28c.vs.cont, jan28b.vs.cont,
    feb25c.vs.cont, feb25b.vs.cont)

# Make the estimates of the differences
( diffcontrol = estimable(model1, compcont, conf.int = 0.993) )
```

```
               Estimate Std. Error    t value DF     Pr(>|t|)   Lower.CI   Upper.CI
jan2c.vs.cont    -0.268  0.1266186 -2.1165929 20 4.702800e-02 -0.6484623  0.1124623
jan28c.vs.cont    0.108  0.1266186  0.8529553 20 4.037838e-01 -0.2724623  0.4884623
```

```
jan28b.vs.cont   -0.154  0.1266186  -1.2162511 20 2.380494e-01 -0.5344623  0.2264623
feb25c.vs.cont   -1.268  0.1266186 -10.0143275 20 3.088505e-09 -1.6484623 -0.8875377
feb25b.vs.cont   -1.180  0.1266186  -9.3193268 20 1.021218e-08 -1.5604623 -0.7995377
```

Wrapping up an analysis

Cleaning up results to put in a table

A table can be a nice way to display the results instead of a graphic. We are not covering how to make nice tables in R, so you'll likely want to do this in a program like Excel. Here is an example of how I might neaten a table up in R that I could take to Excel for final "prettifying".

```
# Make a table of results of interest,
    # with the estimated differences in means and confidence intervals
diffconttab = round( diffcontrol[,c("Estimate", "Lower.CI", "Upper.CI")], 2)

# I am going to use paste() to make a single column for the confidence intervals
    # Using sprintf() to force 2 decimal places on Upper.CI
diffconttab$ci = with(diffconttab, paste(Lower.CI, sprintf("%.2f", Upper.CI), sep = ", ") )

# I need to make the row names and column names looks nicer
rownames(diffconttab) = c("Jan 2 container minus control",
                     "Jan 28 container minus control",
                     "Jan 28 bare minus control",
                     "Feb 25 container minus control",
                     "Feb 25 bare minus control")

colnames(diffconttab) = c("Difference in mean growth increment (cm)",
                     "Lower", "Upper", "99.3% CI")
```

Here is an example table for the write-up (although it could be improved by wrapping the long column name).

```
diffconttab[,c(1, 4)]
```

	Difference in mean growth increment (cm)	99.3% CI
Jan 2 container minus control	-0.27	-0.65, 0.11
Jan 28 container minus control	0.11	-0.27, 0.49
Jan 28 bare minus control	-0.15	-0.53, 0.23
Feb 25 container minus control	-1.27	-1.65, -0.89
Feb 25 bare minus control	-1.18	-1.56, -0.80

Graphic of results

Today I am just going to make one graphic, showing the results of the comparisons of mean growth increment of each transplanting date/stock type combination compared to the control. Graphics can become more complicated once we have multiple factor variables.

Here I make a graphic that has one of the fixed effects on the x axis and the other fixed effect indicated by confidence intervals made with different line types per group. To do this, I had to add both factors to the dataset I was plotting from, the `diffcontrol` dataset. Key to making this graphic is the use of `position_dodge` so the two stock types for one planting date don't fall on top of each other In addition, I had to carefully set the `width` of the error bars in this comparatively complicated graphic. I also moved the legend inside the graphic for the first time, which you can see is done in `theme`.

In your write-up, don't forget an appropriate caption explaining any/all the elements of the graphic such as what the error bars represent and which the order the comparisons were done (i.e., which group was subtracted). This particular graph might need more explanation than previous graphs.

It is likely that you won't need a graphic this complicated for your assignment write-up. If that's the case, go back to lab 3 to get code for graphing.

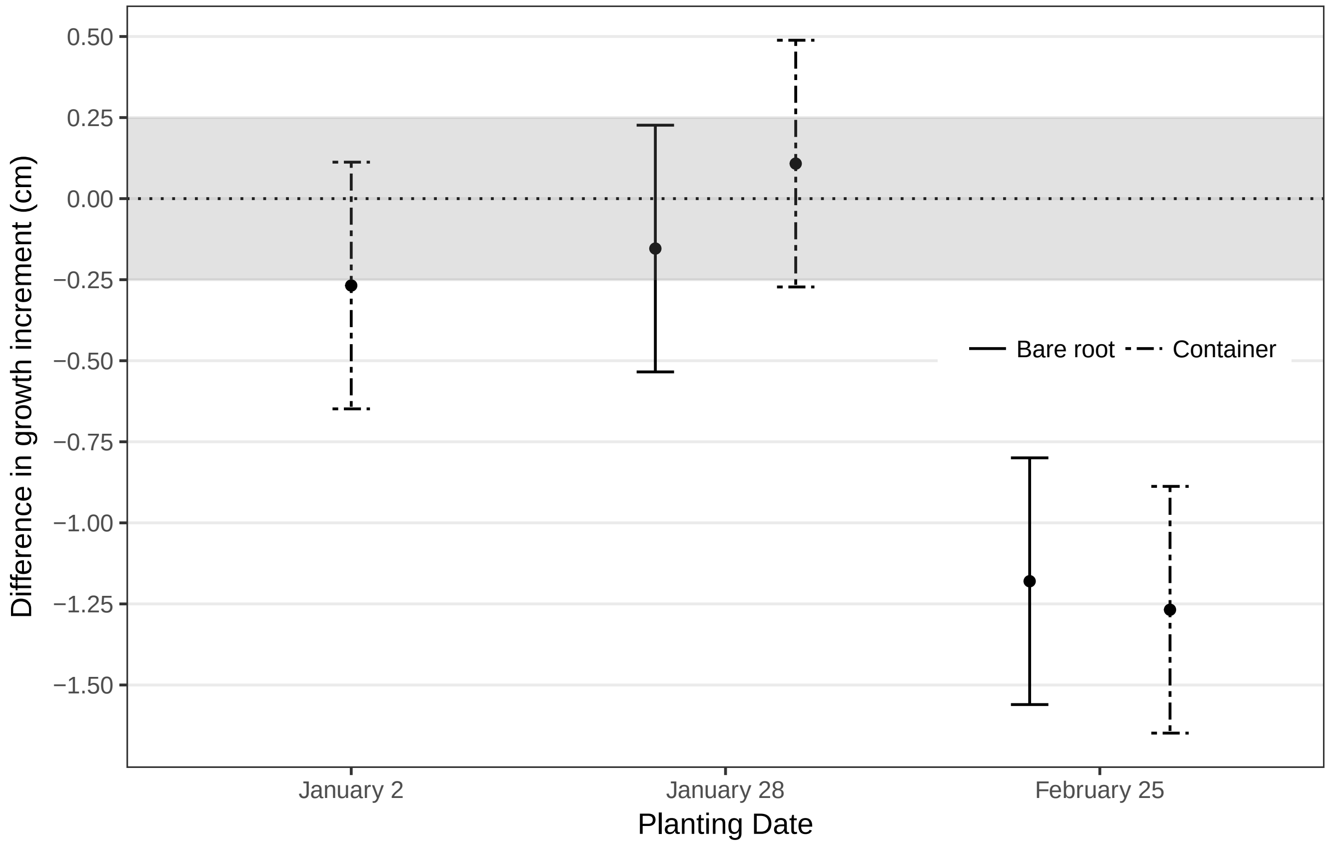
```r
# The dataset I'm making the graphic with
diffcontrol
```

```
               Estimate Std. Error     t value DF     Pr(>|t|)   Lower.CI   Upper.CI
jan2c.vs.cont    -0.268  0.1266186  -2.1165929 20 4.702800e-02 -0.6484623  0.1124623
jan28c.vs.cont    0.108  0.1266186   0.8529553 20 4.037838e-01 -0.2724623  0.4884623
jan28b.vs.cont   -0.154  0.1266186  -1.2162511 20 2.380494e-01 -0.5344623  0.2264623
feb25c.vs.cont   -1.268  0.1266186 -10.0143275 20 3.088505e-09 -1.6484623 -0.8875377
feb25b.vs.cont   -1.180  0.1266186  -9.3193268 20 1.021218e-08 -1.5604623 -0.7995377
```

```r
# Add plantdate variable to the dataset
    # and then setting the level order to something chronological
diffcontrol$plantdate = factor(c("Jan2", "Jan28", "Jan28", "Feb25", "Feb25"),
                   levels = c("Jan2", "Jan28", "Feb25"),
                   labels = c("January 2", "January 28", "February 25") )

# Add a stocktype variable to diffcontrol
diffcontrol$stocktype = c("C", "C", "B", "C", "B")
diffcontrol
```

```
               Estimate Std. Error     t value DF     Pr(>|t|)   Lower.CI   Upper.CI   plantdate
jan2c.vs.cont    -0.268  0.1266186  -2.1165929 20 4.702800e-02 -0.6484623  0.1124623   January 2
jan28c.vs.cont    0.108  0.1266186   0.8529553 20 4.037838e-01 -0.2724623  0.4884623  January 28
jan28b.vs.cont   -0.154  0.1266186  -1.2162511 20 2.380494e-01 -0.5344623  0.2264623  January 28
feb25c.vs.cont   -1.268  0.1266186 -10.0143275 20 3.088505e-09 -1.6484623 -0.8875377 February 25
feb25b.vs.cont   -1.180  0.1266186  -9.3193268 20 1.021218e-08 -1.5604623 -0.7995377 February 25
               stocktype
jan2c.vs.cont          C
jan28c.vs.cont         C
jan28b.vs.cont         B
feb25c.vs.cont         C
feb25b.vs.cont         B
```

```r
( g1 = ggplot(diffcontrol, aes(y = Estimate, x = plantdate, group = stocktype)) +
    # Define stock as group this week as well as set x and y axes
    geom_point(position = position_dodge(width = .75)) + # Add points, dodge by group
    geom_errorbar(aes(ymax = Upper.CI, ymin = Lower.CI,
                   linetype = stocktype,
                   width = c(.1, .2, .2, .2, .2)),
                position = position_dodge(width = .75)) + # Add errorbars, dodge by group
    theme_bw() +
    labs(y = "Difference in growth increment (cm)",
        x = "Planting Date") +
    scale_linetype_manual(values = c("solid", "twodash"),
                      name = "", # Change names in legend
                      labels = c("Bare root", "Container")) +
    geom_hline(yintercept = 0, linetype = 3) + # Add horizontal line at 0
    geom_rect(xmax = Inf, xmin = -Inf, ymax = .25, ymin = -.25,
            fill = "grey54", alpha = .05) + # Add grey rectangle
    theme(legend.position = c(.825, .55), # change legend position
         legend.direction = "horizontal", # make legend horiz
         panel.grid.minor = element_blank(),
         panel.grid.major.x = element_blank() ) + # Remove gridlines
    scale_y_continuous(breaks = seq(-1.5, .5, .25) ) ) # Add more breaks on the y axis (every .25 cm)
```


0.50
0.25
0.00
−0.25
−0.50
−0.75
−1.00
−1.25
−1.50
Difference in growth increment (cm)
Bare root
Container
January 2
January 28
February 25
Planting Date