FES 524 Winter 2022 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 interpret interactions between categorical explanatory variables.

Load R packages needed today

```
library(dplyr)
library(ggplot2)
library(nlme)
library(emmeans)
```

Read in the dataset

We will be working with the dataset `lab4.example.data.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.data.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: chr  "Jan2" "Jan2" "Jan2" "Jan2" ...
 $ stock    : chr  "contain" "contain" "contain" "contain" ...
 $ 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 and then make `stock` and `nursery` factors.

```
growthdata$plantdate = factor(growthdata$plantdate,
                              levels = c("Jan2","Jan28","Feb25"),
                              labels = c("Jan 2","Jan 28","Feb 25"))

# Make stock type and nursery factors
growthdata = mutate(growthdata,
                    stock = factor(stock),
                    nursery = factor(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 the 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 must occur 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* designs. If the factors in a study are not crossed, as can sometimes happen in both observational studies and certain field experiments, 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`. In the output of `xtabs()` below you can see that we have perfect crossing with 5 observations per factor combination. We knew this should be true based on the study design.

```
# Examine the number of observations in the groups
xtabs(~stock + plantdate, data = 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. This uses the specialty function `n()` from **dplyr**.

Since we now have multiple grouping variables like we did in week 2, we'll makes sure to `"drop"` groups after calculating the summary statistics.

```
( 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),
                  .groups = "drop") )
```

```
# A tibble: 6 x 7
  plantdate stock       n  mean    sd   min   max
  <fct>     <fct>   <int> <dbl> <dbl> <dbl> <dbl>
1 Jan 2     barert      5  2.32 0.55   1.52  2.79
2 Jan 2     contain     5  2.05 0.52   1.18  2.48
3 Jan 28    barert      5  2.16 0.59   1.19  2.65
4 Jan 28    contain     5  2.42 0.62   1.4   3.08
5 Feb 25    barert      5  1.14 0.570  0.43  1.67
6 Feb 25    contain     5  1.05 0.52   0.22  1.42
```

Graphical data exploration

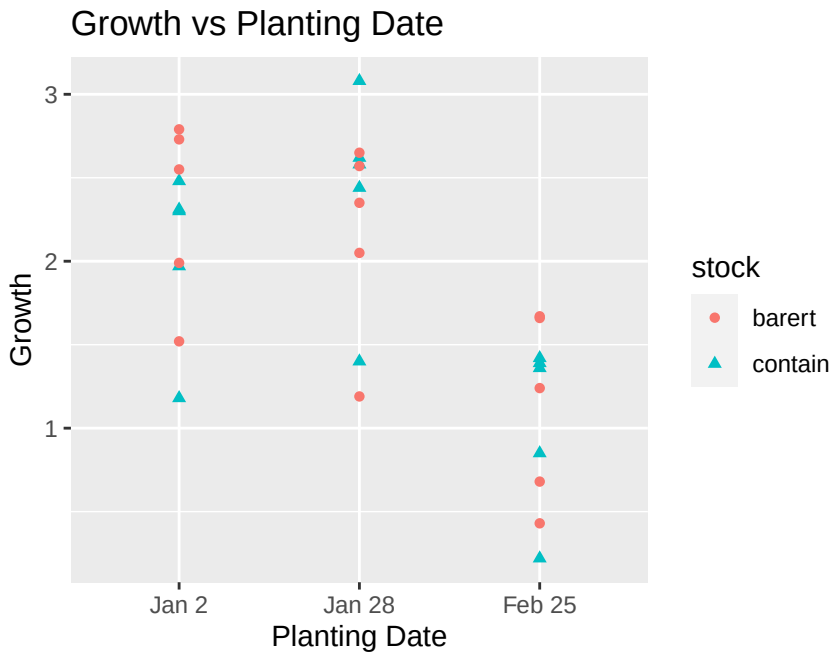
We'll explore our data, looking at scatterplots of our response variable versus each other variables (`plantdate`, `stock`, `nursery`). We'll use color and shapes to help explore patterns. Think about any patterns you see along with the research questions.

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 planning future research based on a similar study design.

In this case we see a lot of noise around the effect of the factor combinations among nurseries, although in all nurseries February 25 had the lowest observed growth. This indicates a fair amount of unexplained variation and/or an unexpected nursery effects. Maybe something like localized weather came into play here that caused this?

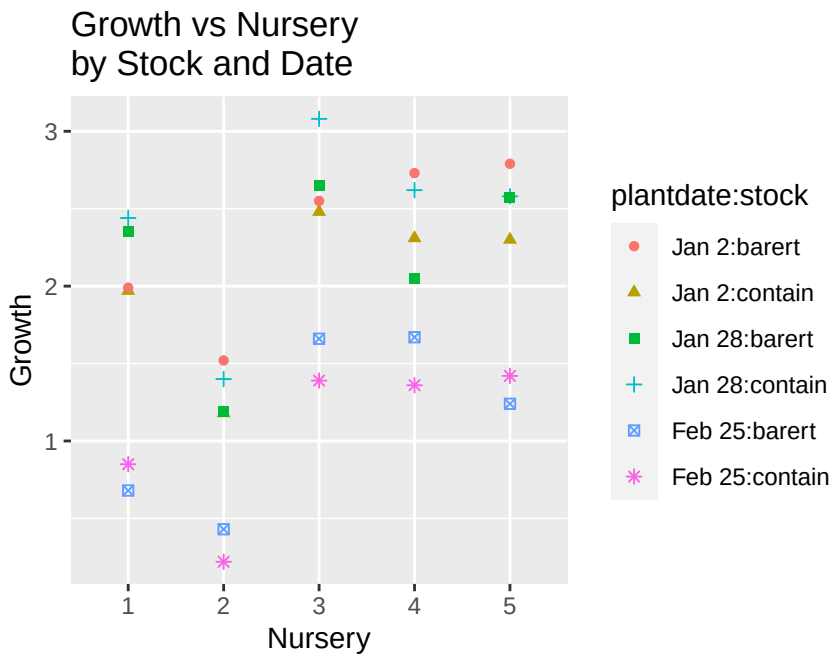
```
# Plot the raw data

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

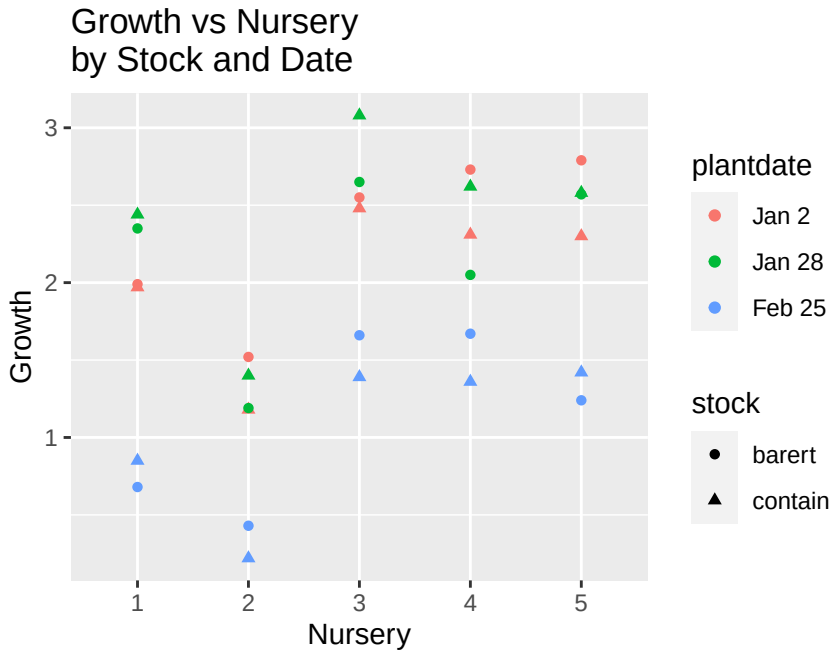


```
# Scatter plot of growth vs stock,
    # color/shape by plantdate
qplot(x = stock, y = growth,
      color = plantdate,
      shape = plantdate,
      data = growthdata,
      xlab = "Stock Type",
      ylab = "Growth",
      main = "Growth vs Stock by Date")
```


```
# 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(x = nursery, y = growth,
      color = plantdate:stock,
      shape = plantdate:stock,
      data = growthdata,
      xlab = "Nursery",
      ylab = "Growth",
      main = "Growth vs Nursery\nby Stock and Date")
```




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



The interaction plot

Researchers did a good job of thinking through and expressing their expectations prior to doing the analysis. In particular, they expected there to be an interaction between planting date and stock type (you'll note that all research questions involve both factors). 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 effect of planting date (i.e., the difference in mean growth between any two

planting dates) changes based on which stock type is planted.

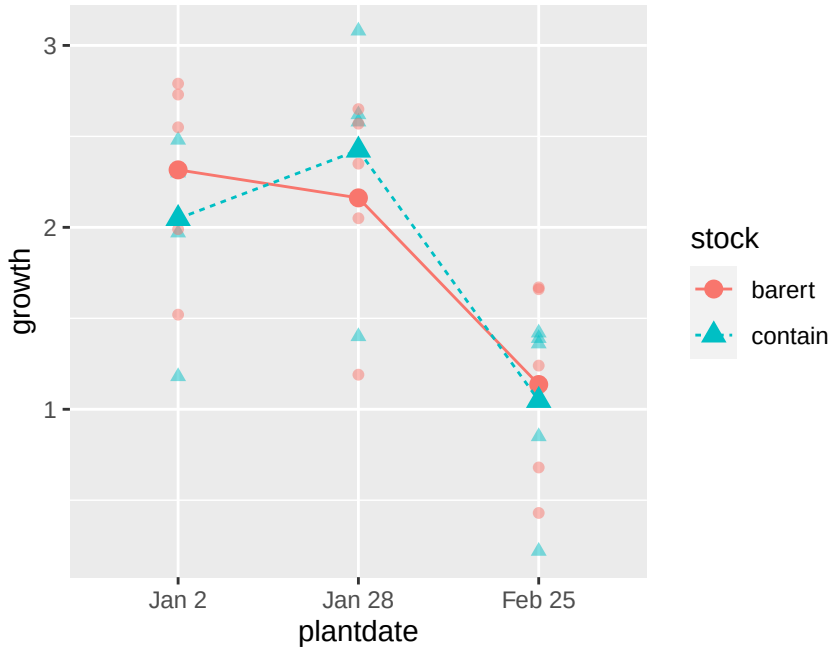
An interaction plot of the means can help us visualize the possibility of an interaction when exploring a dataset. Note that an interaction plot like this is an *exploratory* graphic. The plot does not have error bars and so such plots are not appropriate for making inference unless additional information is added. 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 to show the mean response vs one of the categorical explanatory variables. The second explanatory variable is represented by line types 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 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,
                       color = stock,
                       shape = stock) ) +
    geom_point(alpha = .5, size = 1.75) + # Add raw data
    stat_summary(fun = mean, geom = "point", size = 3) + # Add points for means
    stat_summary(fun = mean, geom = "line") # Connect points with lines
```



What do you think about the possibility of an interaction here? Do you think the plot shows that the effect of planting date could depend on stock type or not?

Fitting a factorial linear mixed model with `lme()`

We will fit a linear mixed model using `lme()` from package **nlme**, where the response variable is `growth`, `nursery` is a random effect and `plantdate` and `stock` are fixed effects. Remember that our observation-level random effect, the residual error, is included automatically in linear mixed models fit with `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 models in R. 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 for symmetry of the residuals with a boxplot.

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 response residuals. We use the `type` argument to get the Pearson residuals. See the help page for `residuals.lme` for a reminder.

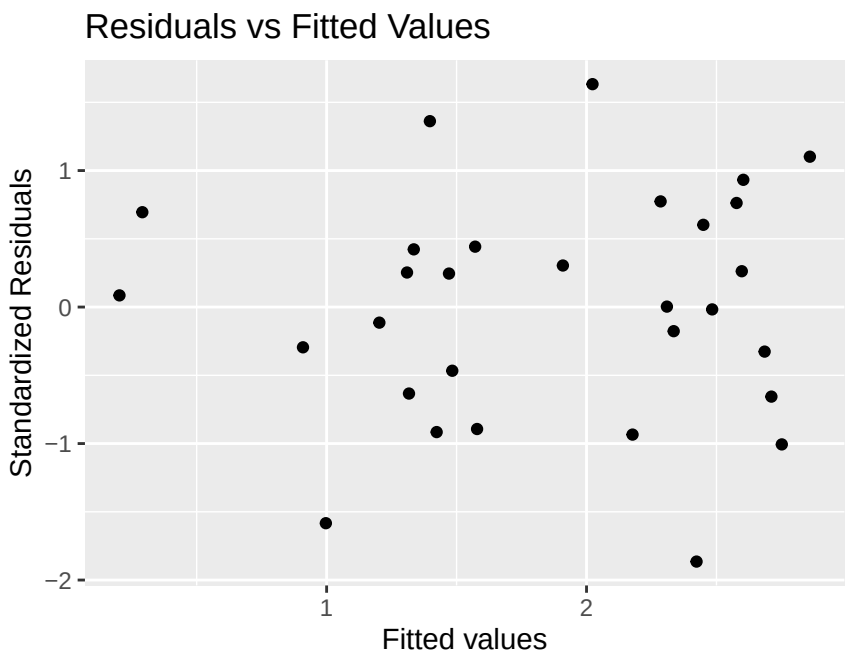
In these plots we are checking the assumption of constant variance across groups and looking for any other unusual patterns.

You should have noted that the two values on the far left of the residuals vs fitted values look a little unusual since they are "far" from the rest of the points. If these were our own data, where we knew more about it, we should investigate those points to see what might be going on.

In addition, it looks like there is a possibility of nonconstant variances among stock types. The `barert` group spread goes from about -2 to 1.5 and the `contain` groups goes from about -1 to 1. This is just under a doubling in spread. I'm going to move forward making inference from the model, but this is a case where someone could decide to relax the assumption of constant variances for that variable.

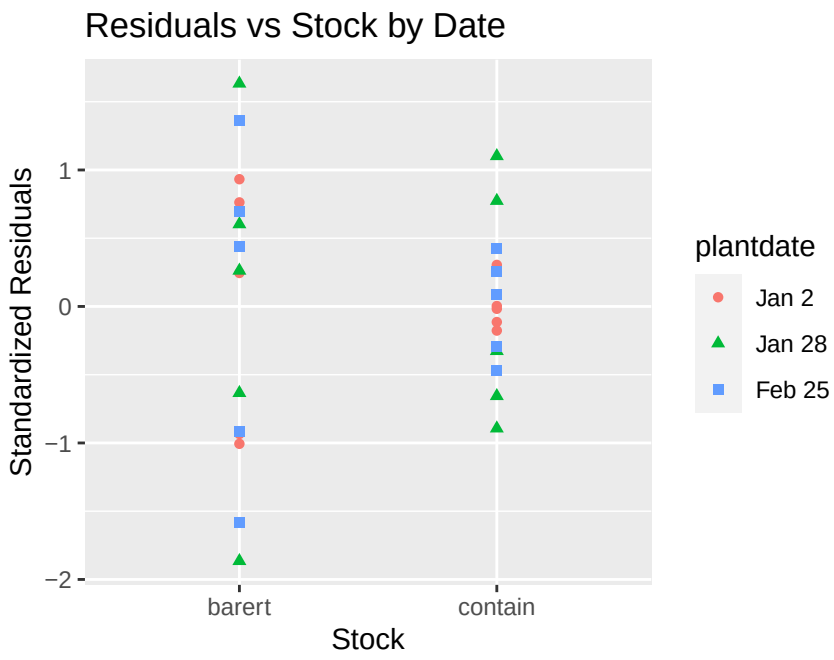
```
growthdata$res = residuals(model1, type = "pearson")
growthdata$fit = fitted(model1)

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




```
# Residuals vs stock type
qplot(x = stock, y = res,
      color = plantdate,
      shape = plantdate,
      data = growthdata,
      xlab = "Stock",
```

```
      ylab = "Standardized Residuals",
      main = "Residuals vs Stock by Date")
```

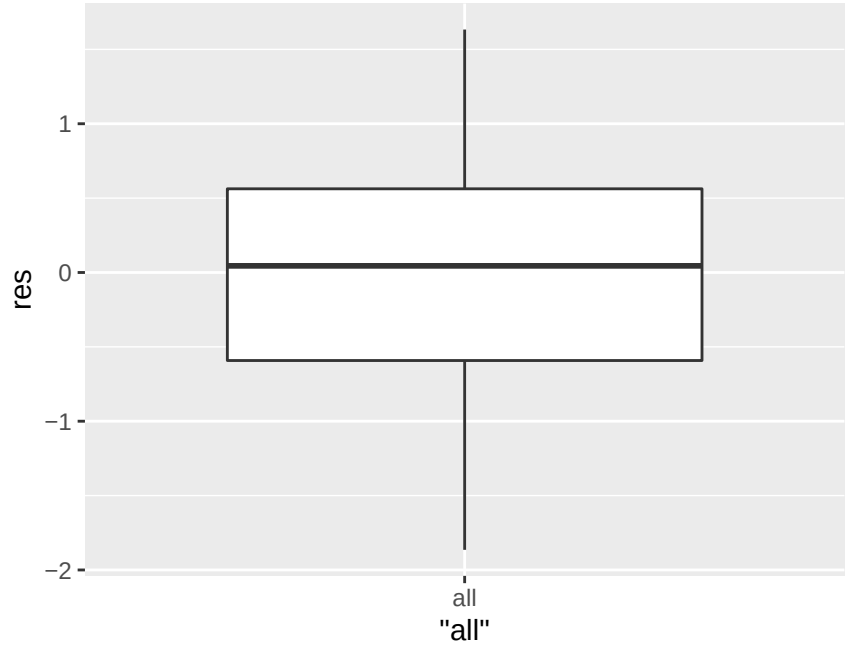



```
# Residuals vs plantdate
qplot(plantdate, res,
      color = stock,
      shape = stock,
      data = growthdata,
      xlab = "Planting Date",
      ylab = "Standardized Residuals",
      main = "Residuals vs Date by Stock")
```

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. This looks fine.

```
# Check normality of residuals with boxplot
```

```
qplot(x = "all", y = res,
      data = growthdata, geom = "boxplot")
```



Extending the 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. See section 5.5 in the *Statistical Sleuth* for a discussion about the effect of differing standard deviations among groups. We are using a sample to try to determine if the *populations* have the same spread, which is extremely difficult to do for these small sample sizes.

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

Relaxing the assumption of constant 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 vital 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.

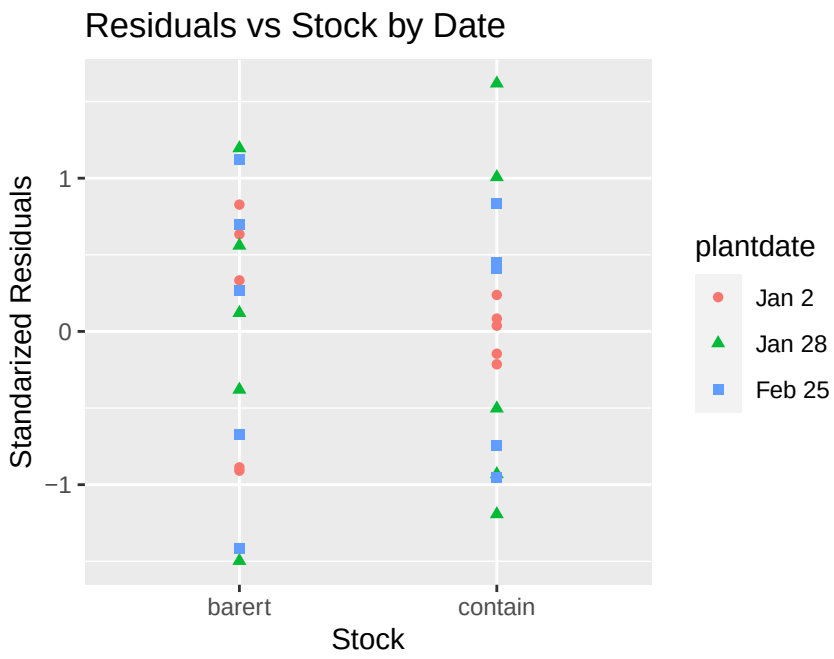
I'll just put the residuals vs explanatory variable plots here so you can see how they changed.

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

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(x = stock, y = res2,
      color = plantdate,
      shape = plantdate,
      data = growthdata,
```

```
      xlab = "Stock",
      ylab = "Standarized Residuals",
      main = "Residuals vs Stock by Date")
```




```
# Residuals vs plantdate
    # Notice this plot looks more-or-less the same
qplot(x = plantdate, y = res2,
      color = stock,
      shape = 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()` and then set up the comparisons of interest.

If we want, we can use p-value for the interaction to give evidence against the null hypothesis of no interaction. This might be useful to report since the researchers expected the effect of planting date to depend on which type of stock was planted, although we need estimates and confidence intervals to talk about the practical importance of any results.

The `summary()` output is not particularly useful once we have multiple factors and some factors have >2 levels.

```
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
```

```
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 for each stock type?
2. Which planting date/stock type combinations have different growth compared to the control group (Jan 2, bare root)?

Custom contrasts with emmeans

To get estimates for addressing the first question we will need to write *custom contrasts* in **emmeans**. This means there are no built-in tools to easily calculate the comparisons of interest.

The first step to achieve this is to use `emmeans()` to calculate the estimated marginal means for each planting date-stock type combination. Notice that this uses the formula code like last week but we didn't put anything on the left side of the tilde (like `pairwise` or `trt.vs.ctrl`). The output is only the estimated marginal means; no comparisons are done.

```
( emm_mod1 = emmeans(model1, specs = ~ stock*plantdate) )
```

```
 stock   plantdate emmean    SE df lower.CL upper.CL
 barert  Jan 2       2.32 0.251  4    1.619     3.01
 contain Jan 2       2.05 0.251  4    1.351     2.75
 barert  Jan 28      2.16 0.251  4    1.465     2.86
```

```
 contain Jan 28      2.42 0.251  4    1.727     3.12
 barert  Feb 25      1.14 0.251  4    0.439     1.83
 contain Feb 25      1.05 0.251  4    0.351     1.75

Degrees-of-freedom method: containment
Confidence level used: 0.95
```

The order of the factor combinations in the above output is key to the process we're going to use. To build the specific comparisons between January 28 and February 25 for each stock type, we will need to pull out the means for each group that will be involved in the comparisons. We can do this by writing out a vector of 0 and 1 values, one value for each mean. A `1` is how we ask for a specific mean, while a `0` indicates a value we don't want.

This is easier to see with an example. We will need the January 28-bare root group mean for one of the comparisons. The marginal mean for the January28-bare root group is the third estimate in `emm_mod1`. Therefore we will use a `1` in the third element of the vector. All other elements are `0`.

Note this vector must be the same length as the number of group combinations/marginal means. In this case, we have 6 values total.

This vector represents the estimated January 28 planted-bare root stock mean.

```
jan28_bare = c(0, 0, 1, 0, 0, 0)
```

We can do the same thing for the other three groups that we are involved in the comparisons of interest. Each vector is length 6. I carefully name these so I know which group mean I'm pulling out.

```
jan28_contain = c(0, 0, 0, 1, 0, 0)
feb25_bare = c(0, 0, 0, 0, 1, 0)
feb25_contain = c(0, 0, 0, 0, 0, 1)
```

Now that we have vectors that represent group means, we can take differences among vectors to calculate the differences among groups means. Differences among group means is the information we will want to report to help address the research question.

The work is done in `contrast()` from the **emmeans** package (see `?contrast.emmGrid`). The comparisons are calculated from `emm_mod1` above. The comparisons, done by taking a difference between the vectors that make up the comparisons, are given as a list to the `method` argument.

```
contrast(emm_mod1, method = list(jan28_contain - feb25_contain,
                                 jan28_bare - feb25_bare) )
```

```
 contrast             estimate    SE df t.ratio p.value
 c(0, 0, 0, 1, 0, -1)     1.38 0.127 20  10.867  <.0001
 c(0, 0, 1, 0, -1, 0)     1.03 0.127 20   8.103  <.0001

Degrees-of-freedom method: containment
```

The result is the comparisons we wanted, but it's difficult to tell which comparison is which. I find it works better for me to give the comparisons some meaningful names.

```
contrast(emm_mod1, method = list("Jan 28 minus Feb 25 contain" = jan28_contain - feb25_contain,
                                 "Jan 28 minus Feb 25 bare" = jan28_bare - feb25_bare) )
```

```
 contrast                    estimate    SE df t.ratio p.value
 Jan 28 minus Feb 25 contain     1.38 0.127 20  10.867  <.0001
 Jan 28 minus Feb 25 bare        1.03 0.127 20   8.103  <.0001

Degrees-of-freedom method: containment
```

The researchers wanted to be fairly conservative with these results because they have management implications, and so they decided to use adjustments for multiple comparisons. I'll use `"sidak"` for a family of two comparisons via the `adjust` argument for these comparisons. Given there are only two comparisons this shouldn't be extremely conservative.

You won't see much difference with this adjustment because right now we are only seeing an adjustment of the p-values, which are already very small. The adjustment is more apparent if we compared the confidence interval limits with and without the adjustment.

```
contrast(emm_mod1, method = list("Jan 28 minus Feb 25 contain" = jan28_contain - feb25_contain,
                                  "Jan 28 minus Feb 25 bare" = jan28_bare - feb25_bare),
         adjust = "sidak")
```

```
 contrast                    estimate    SE df t.ratio p.value
 Jan 28 minus Feb 25 contain     1.38 0.127 20  10.867  <.0001
 Jan 28 minus Feb 25 bare        1.03 0.127 20   8.103  <.0001

Degrees-of-freedom method: containment
P value adjustment: sidak method for 2 tests
```

The `contrast()` function can be used with `summary()` the same way we used it last week. This allows us to get CI for the comparisons and then output the results into a nice format for plotting and tables. I'll use `infer = TRUE` to demonstrate keeping both hypothesis tests and confidence intervals in the output.

I'll use a pipe in order to clearly use all these functions, naming the resulting data.frame `diff_q1`.

```
( diff_q1 = emm_mod1 %>%
        contrast(method = list("Jan 28 minus Feb 25 contain" = jan28_contain - feb25_contain,
                               "Jan 28 minus Feb 25 bare" = jan28_bare - feb25_bare),
                 adjust = "sidak") %>%
        summary(infer = TRUE ) )
```

```
 contrast                    estimate    SE df lower.CL upper.CL t.ratio p.value
 Jan 28 minus Feb 25 contain     1.38 0.127 20     1.07     1.68  10.867  <.0001
 Jan 28 minus Feb 25 bare        1.03 0.127 20     0.72     1.33   8.103  <.0001

Degrees-of-freedom method: containment
Confidence level used: 0.95
Conf-level adjustment: sidak method for 2 estimates
P value adjustment: sidak method for 2 tests
```

All vs control comparisons in emmeans

The second question is one that involves each group vs a control group. The **emmeans** package has a built-in option for this, which we can use via the `contrast()` function.

Since I made the control the reference group in my model, I can use the built-in option `trt.vs.ctrl` to get the comparisons I want.

If my reference group was the very last group instead of the first I would use `trt.vs.ctrlk`. If the control was neither first nor last but in some other position I would need to use `trt.vs.ctrlk` and then indicate the index number of the control group with `ref`. For example, if the control group was the third group in `emm_mod1` I would use `ref = 3`.

Here's what using `trt.vs.ctrl` looks like. I again use `emm_mod1` as the basis of the comparisons in `contrast()`.

You'll notice this uses a Dunnett adjustment for multiple comparisons, which is pretty standard when comparing each group to some control group. I'll use this default on my family of comparisons. Since I used two different adjustments for multiple comparisons, I will have to be careful to clearly state this in my "Methods" section along with the size of each family of comparisons. This information will also need to be included with any tables or graphs.

```
emm_mod1 %>%
    contrast(method = "trt.vs.ctrl")
```

```
 contrast                      estimate    SE df t.ratio p.value
 contain Jan 2 - barert Jan 2    -0.268 0.127 20  -2.117  0.1715
 barert Jan 28 - barert Jan 2    -0.154 0.127 20  -1.216  0.6126
 contain Jan 28 - barert Jan 2    0.108 0.127 20   0.853  0.8187
 barert Feb 25 - barert Jan 2    -1.180 0.127 20  -9.319  <.0001
 contain Feb 25 - barert Jan 2   -1.268 0.127 20 -10.014  <.0001

Degrees-of-freedom method: containment
P value adjustment: dunnettx method for 5 tests
```

Again I will want to get CI and put things in a data.frame. I'll be using these results to make a graphic.

```
( diff_q2 = emm_mod1 %>%
        contrast(method = "trt.vs.ctrl") %>%
        summary(infer = c(TRUE, FALSE) ) )
```

```
 contrast                       estimate    SE df lower.CL upper.CL
 contain Jan 2 - barert Jan 2     -0.268 0.127 20   -0.617   0.0806
 barert Jan 28 - barert Jan 2     -0.154 0.127 20   -0.503   0.1946
 contain Jan 28 - barert Jan 2     0.108 0.127 20   -0.241   0.4566
 barert Feb 25 - barert Jan 2     -1.180 0.127 20   -1.529  -0.8314
 contain Feb 25 - barert Jan 2    -1.268 0.127 20   -1.617  -0.9194

Degrees-of-freedom method: containment
Confidence level used: 0.95
Conf-level adjustment: dunnettx method for 5 estimates
```

Wrapping up an analysis

A table can be a nice way to display the results instead of or in addition to 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. I show some of the work I did to clean up a table for potential inclusion in the assignment in the "Bonus graphics" section this week.

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 the line types of the confidence intervals. To get this to work I had to add both factors to the dataset I was plotting from, the `diff_q2` dataset. The 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, what adjustment and size any confidence intervals are, and the order that 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. If that's the case, go back to lab 3 to get code for graphing.

```
# This is the dataset I'm making the graphic with
diff_q2
```

```
 contrast                       estimate    SE df lower.CL upper.CL
 contain Jan 2 - barert Jan 2     -0.268 0.127 20   -0.617   0.0806
 barert Jan 28 - barert Jan 2     -0.154 0.127 20   -0.503   0.1946
 contain Jan 28 - barert Jan 2     0.108 0.127 20   -0.241   0.4566
 barert Feb 25 - barert Jan 2     -1.180 0.127 20   -1.529  -0.8314
 contain Feb 25 - barert Jan 2    -1.268 0.127 20   -1.617  -0.9194

Degrees-of-freedom method: containment
Confidence level used: 0.95
Conf-level adjustment: dunnettx method for 5 estimates
```

```
# We need to add our two factor variables to this dataset
    # so we can tell the groups apart

# First a plantdate variable to the dataset
    # and then setting the level order to something chronological
diff_q2$plantdate = factor(c("Jan2", "Jan28", "Jan28", "Feb25", "Feb25"),
                           levels = c("Jan2", "Jan28", "Feb25"),
```

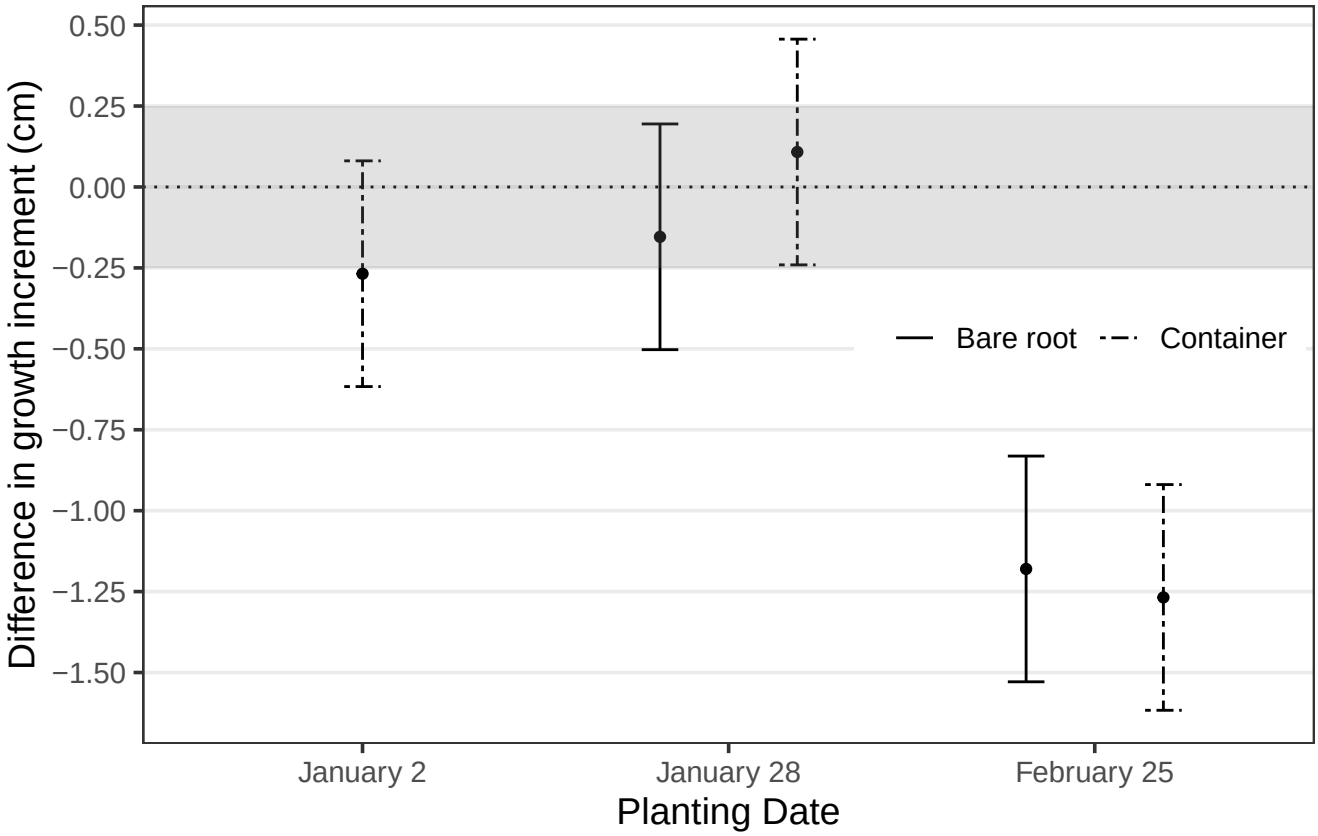
```
                          labels = c("January 2", "January 28", "February 25") )

# Second add a stocktype variable to diffcontrol
# I will allow ggplot to set the order of these alphabetically
diff_q2$stocktype = c("C", "B", "C", "B", "C")
diff_q2
```

```
 contrast                      estimate    SE df lower.CL upper.CL plantdate   stocktype
 contain Jan 2 - barert Jan 2    -0.268 0.127 20   -0.617   0.0806 January 2   C
 barert Jan 28 - barert Jan 2    -0.154 0.127 20   -0.503   0.1946 January 28  B
 contain Jan 28 - barert Jan 2    0.108 0.127 20   -0.241   0.4566 January 28  C
 barert Feb 25 - barert Jan 2    -1.180 0.127 20   -1.529  -0.8314 February 25 B
 contain Feb 25 - barert Jan 2   -1.268 0.127 20   -1.617  -0.9194 February 25 C

Degrees-of-freedom method: containment
Confidence level used: 0.95
Conf-level adjustment: dunnettx method for 5 estimates
```

```
( g1 = ggplot(diff_q2, 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(ymin = lower.CL, ymax = upper.CL,
                            linetype = stocktype,
                            width = c(.1, .2, .2, .2, .2)),
                        position = position_dodge(width = .75) ) +
          # Add error bars, dodge by group
          theme_bw(base_size = 14) +
          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.direction = "horizontal", # make legend horiz
                legend.position = c(.8, .55), # change legend position
                panel.grid.minor = element_blank(), # Remove gridlines
                panel.grid.major.x = element_blank() ) +
          scale_y_continuous(breaks = seq(-1.5, .5, .25) ) ) # Add more breaks on the y axis
```


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