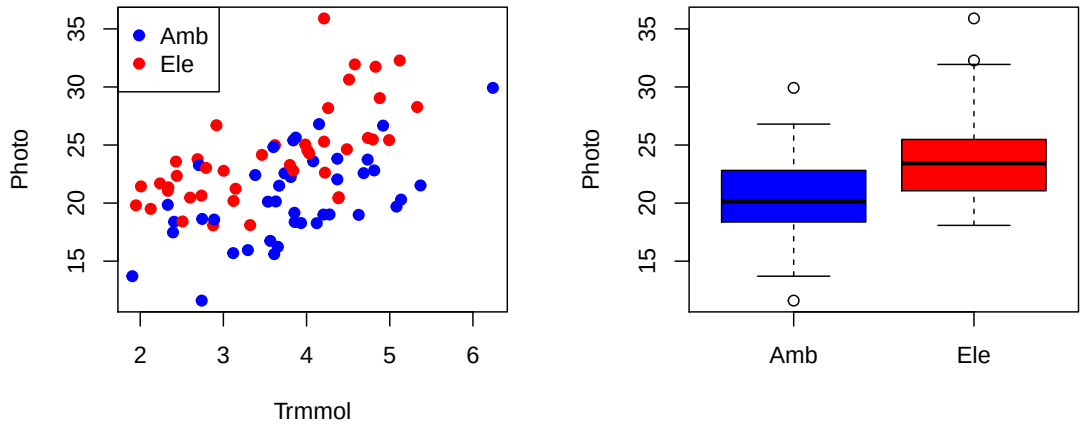

Figure 7.12: Photosynthesis and leaf transpiration rate (Trmmol) for leaves in elevated (red) and ambient (blue) CO2 concentration.

7.4.2 Using predicted effects to make sense of model output

Next we show an example of a model where using predicted effects as introduced in Section 7.4.1 is very helpful in understanding model output. For this example, we use measurements of photosynthesis and transpiration of tree leaves in the EucFACE experiment (see Section A.23 for a description of the data). We are interested in the relationship between photosynthesis and transpiration (the ratio of which is known as the water-use efficiency), and whether this relationship differs with CO_2 treatment. The data are shown in Fig. 7.12.

```
eucgas <- read.csv("eucface_gasexchange.csv")

palette(c("blue","red"))
with(eucgas, plot(Trmmol, Photo, pch=19, col=CO2))
legend("topleft", levels(eucgas$CO2), pch=19, col=palette())

boxplot(Photo ~ CO2, data=eucgas, col=palette(), ylab="Photo")
```

It seems quite clear from Fig. 7.12 that, at a given `Trmmol`, `Photo` is higher in the elevated (`Ele`) CO_2 treatment. From the boxplot on the right, it also seems more than reasonable to expect that overall, `Photo` is higher in `Ele` (when not accounting for the `Trmmol` covariate).

Let's fit a linear model to confirm this effect.

```
# A linear model with a continuous and a factor predictor, including the interaction.
lmfit <- lm(Photo ~ CO2*Trmmol, data=eucgas)

# Significance of overall model terms (sequential anova)
anova(lmfit)
```

```
## Analysis of Variance Table
##
## Response: Photo
##             Df Sum Sq Mean Sq F value    Pr(>F)
## CO2          1 263.01  263.01 26.6447 1.78e-06 ***
```

```
## Trmmol      1 490.17  490.17 49.6574 6.08e-10 ***
## CO2:Trmmol  1   3.10    3.10  0.3143   0.5767
## Residuals  79 779.82    9.87
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

And the coefficients table from `summary`:

```
##                 Estimate Std. Error   t value Pr(>|t|)
## (Intercept)   11.4840889  2.1580560 5.3214972    0e+00 ***
## CO2Ele         2.6896268  2.8295735 0.9505414  0.34474
## Trmmol         2.3550052  0.5478918 4.2983031    5e-05 ***
## CO2Ele:Trmmol  0.4132841  0.7372231 0.5605957  0.57666
```

```
# Significance of the individual coefficients:
summary(lmfit)
```

Look at the 'coefficients' table in the `summary` statement. Four parameters are shown, they can be interpreted as, 1) the intercept for 'Amb', 2) the slope for 'Amb', 3) the *difference* in the intercept for 'Ele', compared to 'Amb', 4) the *difference* in the slope for 'Ele', compared to 'Amb'.

It seems that neither the intercept or slope effect of `CO2` is significant here, which is surprising. Also confusing is the fact that the `anova` statement showed a clear significant effect of `CO2`, so what is going on here?

First recall that the sequential anova tests each term against a model that includes *only the terms preceding it*. So, since we added `CO2` as the first predictor, its test in the `anova` is tested against a model that has no predictors. This is similar in approach to simply performing a *t*-test on `Photo` vs. `CO2` (which also shows a significant effect), in other words testing for separation in the right-hand panel of Fig. 7.12. It is clearly a different test from those shown in the `summary` statement.

To understand the tests of the coefficients, we will plot predictions of the model, together with confidence intervals. The following code makes Fig. 7.13, and we introduce the use of the `predict` function to estimate fitted values, and confidence intervals, from a fitted model.

```
# Set up a regular sequence of numbers, for which 'Photo' is to be predicted from
xval <- seq(0, max(eucgas$Trmmol), length=101)

# Two separate dataframes, one for each treatment/
amb_dfr <- data.frame(Trmmol=xval, CO2="Amb")
ele_dfr <- data.frame(Trmmol=xval, CO2="Ele")

# Predictions of the model using 'predict.lm'
# The first argument is the fitted model, the second argument a dataframe
# containing values for the predictor variables.
predamb <- as.data.frame(predict(lmfit, amb_dfr, interval="confidence"))
predele <- as.data.frame(predict(lmfit, ele_dfr, interval="confidence"))

# Plot. Set up the axis limits so that they start at 0, and go to the maximum.
palette(c("blue","red"))
with(eucgas, plot(Trmmol, Photo, pch=19, col=CO2,
                  xlim=c(0, max(Trmmol)),
                  ylim=c(0, max(Photo))))

# Add the lines; the fit and lower and upper confidence intervals.
with(predamb, {
  lines(xval, fit, col="blue", lwd=2)
```

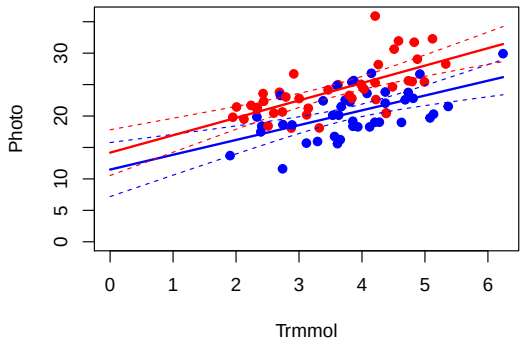

Figure 7.13: Predicted relationship between Photo and Trmmol for ambient and elevated CO_2 concentrations in the EucFACE leaf gas exchange dataset. Dashed lines are confidence intervals for the regression line.

```
  lines(xval, lwr, col="blue", lwd=1, lty=2)
  lines(xval, upr, col="blue", lwd=1, lty=2)
})

with(predele, {
  lines(xval, fit, col="red", lwd=2)
  lines(xval, lwr, col="red", lwd=1, lty=2)
  lines(xval, upr, col="red", lwd=1, lty=2)
})
```

Try this yourself The above plot can also be easily made with the `visreg` package, as we have seen already. Use the code `visreg(lmfit, "Trmmol", by="CO2", overlay=TRUE)` to make a similar plot. Set the x-axis limit with `xlim` to include the intercept.

The intercept in a linear model is of course the value of the Y variable where X is zero. As we can see in Fig. 7.13, the confidence intervals for the regression lines overlap when `Trmmol` is zero - which is the comparison made in the `summary` statement for the intercept. We now see why the intercept was not significant, but it says very little about the treatment difference *in the range of the data*.

Perhaps it is more meaningful to test for treatment differences at a mean value of `Trmmol`. There are four ways to do this.

Centering the predictor

The first approach is to recenter the predictor so that the intercept can be interpreted as the value where the predictor (in our case, `Trmmol`) is at its mean value.

```
# Rescaled transpiration rate
# This is equivalent to Trmmol - mean(Trmmol)
eucgas$Trmmol_center <- scale(eucgas$Trmmol, center=TRUE, scale=FALSE)

# Refit using centered predictor
lmfit2 <- lm(Photo ~ Trmmol_center*CO2, data=eucgas)
```

The coefficients table in the `summary` statement now shows a highly significant effect for `CO2Ele`, a difference of about 4.22 units. It is also possible to compute confidence intervals on the coefficients via `confint(lmfit2)`, try this yourself.

```
# Summary of the fit:
summary(lmfit2)
```

```
##                          Estimate Std. Error    t value Pr(>|t|)
## (Intercept)          20.2096075  0.4958600 40.7566799  < 2e-16 ***
## Trmmol_center         2.3550052  0.5478918  4.2983031    5e-05 ***
## CO2Ele                4.2208837  0.6963167  6.0617295    0e+00 ***
## Trmmol_center:CO2Ele  0.4132841  0.7372231  0.5605957  0.57666
```

Using the effects package

Another way is to compute the `CO2` effect at a mean value of `Trmmol`. This avoids having to refit the model with centered data, and is more flexible.

```
# The effects package calculates effects for a variable by averaging over all other
# terms in the model
library(effects)
Effect("CO2", lmfit)
```

```
##
##  CO2 effect
## CO2
##      Amb      Ele
## 20.20961 24.43049
```

```
# confidence intervals can be obtained via
summary(Effect("CO2", lmfit))
```

```
##
##  CO2 effect
## CO2
##      Amb      Ele
## 20.20961 24.43049
##
##  Lower 95 Percent Confidence Limits
## CO2
##      Amb      Ele
## 19.22262 23.45745
##
##  Upper 95 Percent Confidence Limits
## CO2
##      Amb      Ele
## 21.19659 25.40353
```

The `effects` package is quite flexible. For example, we can calculate the predicted effects at any specified value of the predictors, like so (output not shown):

```
# For example, what is the CO2 effect when Trmmol was 3?
summary(Effect("CO2", lmfit, given.values=c(Trmmol=3)))
```

Least-square means

The effect size while holding other predictors constant at their mean value is also known as the 'least-square mean' (or even 'predicted marginal means'), and is implemented as such in the `lsmeans` package. It is a powerful package, also to make sense of models that are far more complex than the one in this example.

```
library(lsmeans)
summary(lsmeans(lmfit, "CO2"))

## NOTE: Results may be misleading due to involvement in interactions

##  CO2   lsmean        SE df lower.CL upper.CL
##  Amb 20.20961 0.4958600 79 19.22262 21.19659
##  Ele 24.43049 0.4888557 79 23.45745 25.40353
##
## Confidence level used: 0.95

# lsmeans warns that perhaps the results are misleading - this is true for more
# complex models but not a simple one as shown here.
```

Using the `predict` function

Finally, we show that the effects can also be obtained via the use of `predict`, as we already saw in the code to produce Fig. 7.13.

```
# Predict fitted Photo at the mean of Trmmol, for both CO2 treatments
predict(lmfit, data.frame(Trmmol=mean(eucgas$Trmmol),
                          CO2=levels(eucgas$CO2)),
        interval="confidence")

##        fit      lwr      upr
## 1 20.20961 19.22262 21.19659
## 2 24.43049 23.45745 25.40353
```

Further reading This example shows that interpretation of main effects (in this case, `CO2`) is not at all straightforward when the model also includes an interaction term (`CO2:Trmmol`). A readable review of this problem is Engqvist 2005, *Animal Behaviour* 70(4):967-971. There, it is shown that many studies misinterpret the main effects in cases like this.