Multiple Regression
==========

This R markdown document provides an example of performing a multiple regression using the lm() function in R and compares the output with the linReg() function in the jmv (Jamovi) package.

Package management in R

```r
# keep a list of the packages used in this script
packages <- c("tidyverse","rio","jmv")
```

This next code block has eval=FALSE because you don’t want to run it when knitting the file. Installing packages when knitting an R notebook can be problematic.

```r
# check each of the packages in the list and install them if they're not installed already
for (i in packages){
  if(! i %in% installed.packages()){
    install.packages(i,dependencies = TRUE)
  }
  # show each package that is checked
  print(i)
}
```

```r
# load each package into memory so it can be used in the script
for (i in packages){
  library(i,character.only=TRUE)
  # show each package that is loaded
  print(i)
}
```

```
    ## -- Attaching packages --------------------------------------- tidyverse 1.3.0 --

    ## v ggplot2 3.3.3     v purrr   0.3.4
    ## v tibble  3.0.6     v dplyr   1.0.4
    ## v tidyr   1.1.2     v stringr 1.4.0
    ## v readr   1.4.0     v forcats 0.5.1

    ## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
    ## x dplyr::filter() masks stats::filter()
    ## x dplyr::lag()    masks stats::lag()

    ## [1] "tidyverse"
    ## [1] "rio"
    ## [1] "jmv"
```

Multiple Regression

Multiple regression is predicting a continuous outcome variable (dependent variable) with several predictor variables (independent variables). You can perform regressions using categorical predictor variables, but we'll talk more about that later.

Open data file

The rio package works for importing several different types of data files. We're going to use it in this class. There are other packages which can be used to open datasets in R. You can see several options by clicking on the Import Dataset menu under the Environment tab in RStudio. (For a csv file like we have this week we'd use either From Text(base) or From Text (readr). Try it out to see the menu dialog.)

```r
# Using the file.choose() command allows you to select a file to import from another folder.
# dataset <- rio::import(file.choose())
dataset <- rio::import("Album Sales.sav")
```

Get R code from Jamovi output

You can get the R code for most of the analyses you do in Jamovi.

1. Click on the three vertical dots at the top right of the Jamovi window.
2. Click on the Syndax mode check box at the bottom of the Results section.
3. Close the Settings window by clicking on the Hide Settings arrow at the top right of the settings menu.
4. you should now see the R code for each of the analyses you just ran.

lm() function in R

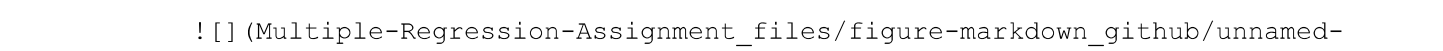
Many linear models are calculated in R using the lm() function. We'll look at how to perform a multiple regression using the lm() function since it's so common.

Visualization

```r
# This code creates a scatter matrix
library(GGally)
```

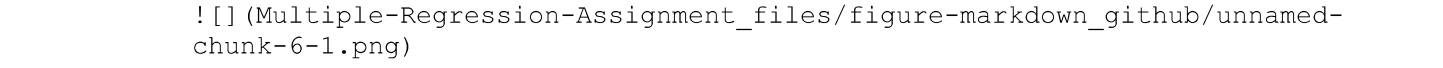
```
## Registered S3 method overwritten by 'GGally':
##   method from
##   +.gg   ggplot2
```

```r
GGally::ggpairs(dataset, columns=c('Sales','Adverts','Airplay','Image'), lower
= list(continuous = "smooth"))
```



```r
# This code creates a scatterplot between a single pair of variables
ggplot(dataset, aes(x = Adverts, y = Sales)) +
  geom_point() +
  stat_smooth(method = lm)
```

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



Computation

```r
model <- lm(formula = Sales ~ Adverts + Airplay + Image, data = dataset)
model
```

```
    ##
    ## Call:
    ## lm(formula = Sales ~ Adverts + Airplay + Image, data = dataset)
    ##
    ## Coefficients:
    ## (Intercept)      Adverts      Airplay        Image
    ##   -26.61296      0.08488      3.36743     11.08634
```

Model assessment

```r
summary(model)
```

```
    ##
    ## Call:
    ## lm(formula = Sales ~ Adverts + Airplay + Image, data = dataset)
    ##
    ## Residuals:
    ##      Min       1Q   Median       3Q      Max
    ## -121.324  -28.336   -0.451   28.967  144.132
    ##
    ## Coefficients:
    ##               Estimate Std. Error t value Pr(>|t|)
    ## (Intercept) -26.612958  17.350001  -1.534    0.127
    ## Adverts       0.084885   0.006923  12.261  < 2e-16 ***
    ## Airplay       3.367425   0.277771  12.123  < 2e-16 ***
    ## Image        11.086335   2.437849   4.548 9.49e-06 ***
```

```
    ## ---
    ## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
    ##
    ## Residual standard error: 47.09 on 196 degrees of freedom
    ## Multiple R-squared:  0.6647, Adjusted R-squared:  0.6595
    ## F-statistic: 129.5 on 3 and 196 DF,  p-value: < 2.2e-16
```

Standardized residuals from lm()

You might notice lm() does not provide the standardized residuals. Those must me calculated separately.

``` r
standardized = lm(scale(Sales) ~ scale(Adverts) + scale(Airplay) +
scale(Image), data=dataset)
summary(standardized)
```

```
    ##
    ## Call:
    ## lm(formula = scale(Sales) ~ scale(Adverts) + scale(Airplay) +
    ##     scale(Image), data = dataset)
    ##
    ## Residuals:
    ##      Min       1Q   Median       3Q      Max
    ## -1.50342 -0.35113 -0.00559  0.35895  1.78605
    ##
    ## Coefficients:
    ##                   Estimate Std. Error t value Pr(>|t|)
    ## (Intercept)    -5.586e-17  4.126e-02   0.000        1
    ## scale(Adverts)  5.108e-01  4.166e-02  12.261  < 2e-16 ***
    ## scale(Airplay)  5.120e-01  4.223e-02  12.123  < 2e-16 ***
    ## scale(Image)    1.917e-01  4.215e-02   4.548 9.49e-06 ***
    ## ---
    ## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
    ##
    ## Residual standard error: 0.5835 on 196 degrees of freedom
    ## Multiple R-squared:  0.6647, Adjusted R-squared:  0.6595
    ## F-statistic: 129.5 on 3 and 196 DF,  p-value: < 2.2e-16
```

function in Jamovi

Compare the output from the lm() function with the output from the function in the jmv package.

``` r
jmv::linReg(
  data = dataset,
  dep = Sales,
  covs = vars(Adverts, Airplay, Image),
  blocks = list(list("Adverts"),
                list("Airplay","Image")),
  refLevels = list(),
  modelTest = TRUE,
  anova = TRUE,
```

```
  ci = TRUE,
  stdEst = TRUE,
  ciStdEst = TRUE)
```

```
    ##
    ##  LINEAR REGRESSION
    ##
    ##  Model Fit Measures
    ##
-------------------------------------------------------------------------------
    ##    Model    R            R²           F            df1    df2    p
    ##
-------------------------------------------------------------------------------
    ##        1    0.5784877    0.3346481     99.58687      1    198    <
.0000001
    ##        2    0.8152715    0.6646677    129.49827      3    196    <
.0000001
    ##
-------------------------------------------------------------------------------
    ##
    ##
    ##  Model Comparisons
    ##
-------------------------------------------------------------------------------
    ##    Model          Model    <U+0394>R²          F           df1    df2
p
    ##
-------------------------------------------------------------------------------
    ##        1    -        2    0.3300196    96.44738      2    196    <
.0000001
    ##
-------------------------------------------------------------------------------
    ##
    ##
    ##  MODEL SPECIFIC RESULTS
    ##
    ##  MODEL 1
    ##
    ##  Omnibus ANOVA Test
    ##
-------------------------------------------------------------------------------
    ##                 Sum of Squares    df     Mean Square    F           p
    ##
-------------------------------------------------------------------------------
    ##    Adverts            433687.8      1     433687.833    99.58687    <
.0000001
    ##    Residuals          862264.2    198       4354.870
    ##
-------------------------------------------------------------------------------
    ##    Note. Type 3 sum of squares
    ##
    ##
    ##  Model Coefficients - Sales
    ##
-------------------------------------------------------------------------------
```

```
    ##    Predictor    Estimate        SE             Lower           Upper
t            p             Stand. Estimate    Lower        Upper
    ##
-------------------------------------------------------------------------------------------
    ##    Intercept    134.13993781    7.536574679    119.27768082
149.0021948    17.798528    < .0000001
    ##    Adverts        0.09612449    0.009632366      0.07712929
0.1151197     9.979322    < .0000001          0.5784877    0.4641726
0.6928029
    ##
-------------------------------------------------------------------------------------------
    ##
    ##
    ##  MODEL 2
    ##
    ##  Omnibus ANOVA Test
    ##
-----------------------------------------------------------------------------
    ##                  Sum of Squares    df     Mean Square    F            p
    ##
-----------------------------------------------------------------------------
    ##    Adverts           333332.44      1     333332.444    150.33820    <
.0000001
    ##    Airplay           325859.87      1     325859.869    146.96795    <
.0000001
    ##    Image              45853.30      1      45853.296     20.68056
0.0000095
    ##    Residuals         434574.58    196       2217.217
    ##
-----------------------------------------------------------------------------
    ##    Note. Type 3 sum of squares
    ##
    ##
    ##  Model Coefficients - Sales
    ##
-------------------------------------------------------------------------------------------
    ##    Predictor    Estimate        SE              Lower           Upper
t            p             Stand. Estimate    Lower        Upper
    ##
-------------------------------------------------------------------------------------------
    ##    Intercept    -26.61295836    17.350000565    -60.82960967
7.60369295    -1.533888     0.1266698
    ##    Adverts        0.08488483     0.006923017      0.07123166
0.09853799    12.261248    < .0000001          0.5108462    0.4286800
0.5930125
    ##    Airplay        3.36742517     0.277770832      2.81962186
3.91522848    12.123034    < .0000001          0.5119881    0.4286993
0.5952769
    ##    Image         11.08633520     2.437849265      6.27855218
15.89411823     4.547588     0.0000095          0.1916834    0.1085566
0.2748103
    ##
-------------------------------------------------------------------------------------------
```