SampleScript.r

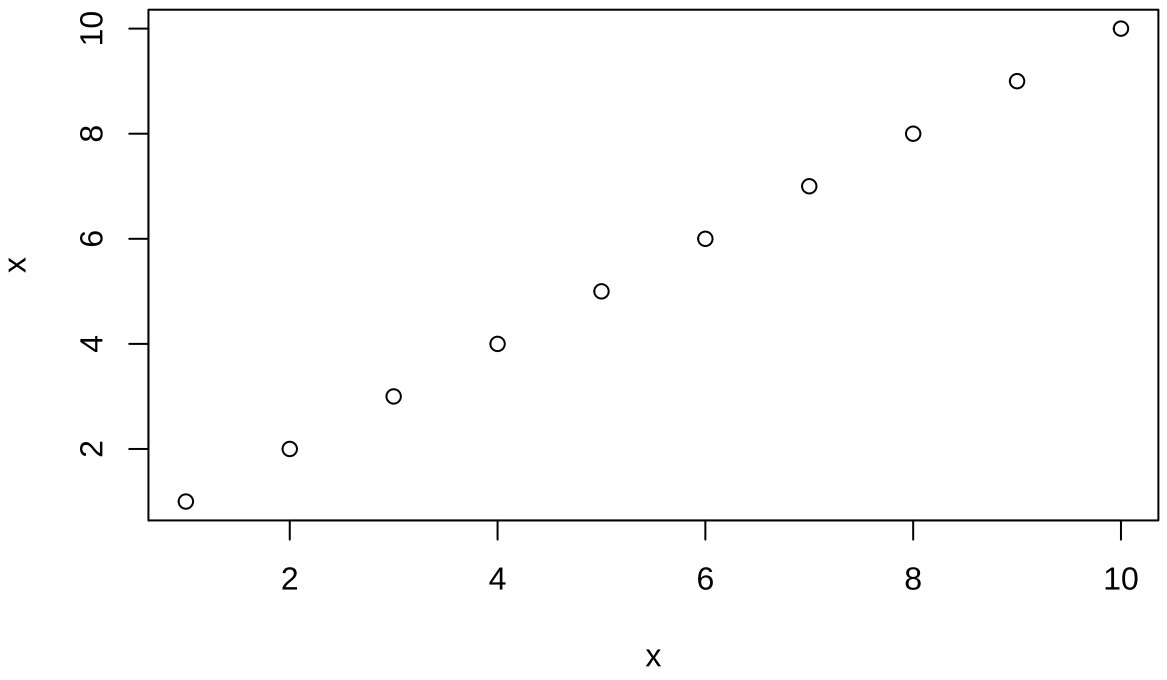
cschwarz

Tue Jan 22 19:21:16 2019

```
# This will be used as an example of creating a notebook in HTML, DOC, or PDF formats.

# See
#     http://rmarkdown.rstudio.com/articles_report_from_r_script.html
# for more informaton




options(useFancyQuotes=FALSE) # renders summary output corrects
#source("schwarz.functions.r")
source('http://www.stat.sfu.ca/~cschwarz/Stat-650/Notes/MyPrograms/schwarz.functions.r')

# This is a quick demo of using Rstudio
x <- 1:10
x
```

```
##  [1]  1  2  3  4  5  6  7  8  9 10
```

```
plot(x,x)
```



```
# This script will read in the cereal data set,
#    do a simple listing,
#    fit a regression line,
#       draw a scatter plot and add the line to the plot
#    do a single factor crd anova
#       get the compact letter display
#       make some plots


# load required libraries
library(ggplot2)
library(emmeans)
library(readxl)

# Read in the cereal data from a csv file
cereal <- read.csv('cereal.csv',
              header=TRUE, as.is=TRUE, strip.white=TRUE)

cereal2 <- readxl::read_excel('ALLofDATA.xls',
                     sheet='cereal',
                     skip=7)
names(cereal2) <- make.names(names(cereal2))


# Define new variables and factors (for categorical variables). CHeck the structure of the data frame
cereal$shelfF <- factor(cereal$shelf)
cereal$Calories.fr.Protein <- cereal$protein * 4;

str(cereal)
```

```
## 'data.frame':    77 obs. of  17 variables:
##  $ name               : chr  "100%_Bran" "100%_Natural_Bran" "All-Bran" "All-Bran_with_Extra_Fiber"
##  $ mfr                : chr  "N" "Q" "K" "K" ...
##  $ type               : chr  "C" "C" "C" "C" ...
##  $ calories           : int  60 110 80 50 110 110 110 140 90 90 ...
##  $ protein            : int  4 3 4 4 2 2 2 3 2 3 ...
##  $ fat                : int  1 5 1 0 2 2 0 2 1 0 ...
##  $ sodium             : int  130 15 260 140 200 180 125 210 200 210 ...
##  $ fiber              : num  10 2 9 14 1 1.5 1 2 4 5 ...
##  $ carbo              : num  5 8 7 8 14 10.5 11 18 15 13 ...
##  $ sugars             : int  6 8 5 0 8 10 14 8 6 5 ...
##  $ shelf              : int  3 3 3 3 3 1 2 3 1 3 ...
##  $ potass             : int  280 135 320 330 NA 70 30 100 125 190 ...
##  $ vitamins           : int  25 0 25 25 25 25 25 25 25 25 ...
##  $ weight             : num  1 1 1 1 1 1 1 1.33 1 1 ...
##  $ cups               : num  0.331 NA 0.33 0.5 0.75 0.75 1 0.75 0.67 0.67 ...
##  $ shelfF             : Factor w/ 3 levels "1","2","3": 3 3 3 3 3 1 2 3 1 3 ...
##  $ Calories.fr.Protein: num  16 12 16 16 8 8 8 12 8 12 ...
```

```
# List  the first few records
cereal[1:5,]
```

```
##                        name mfr type calories protein fat sodium fiber
## 1                 100%_Bran   N    C       60       4   1    130    10
## 2         100%_Natural_Bran   Q    C      110       3   5     15     2
## 3                  All-Bran   K    C       80       4   1    260     9
## 4 All-Bran_with_Extra_Fiber   K    C       50       4   0    140    14
## 5            Almond_Delight   R    C      110       2   2    200     1
##   carbo sugars shelf potass vitamins weight  cups shelfF
## 1     5      6     3    280       25      1 0.331      3
## 2     8      8     3    135        0      1    NA      3
## 3     7      5     3    320       25      1 0.330      3
## 4     8      0     3    330       25      1 0.500      3
## 5    14      8     3     NA       25      1 0.750      3
##   Calories.fr.Protein
## 1                  16
## 2                  12
## 3                  16
## 4                  16
## 5                   8
```

```
# List some variables
cereal$calories
```

```
##  [1]  60 110  80  50 110 110 110 140  90  90 120 110 130 100 110 110 110
## [18] 100 110 110 100 100  90 100 100 110  90 120 130 100 100 100 100 110
## [35] 110 130 110 120 100 140 100 100 110 110 150 150 160  90 120 140  90
## [52] 130 130  90  40  50 100  90 120  90  90 110 100  80  80  90 110 100
## [69]  80 100 150 110 100 110 100  90 110
```

```
cereal[,"calories"]
```

```
##  [1]  60 110  80  50 110 110 110 140  90  90 120 110 130 100 110 110 110
## [18] 100 110 110 100 100  90 100 100 110  90 120 130 100 100 100 100 110
## [35] 110 130 110 120 100 140 100 100 110 110 150 150 160  90 120 140  90
## [52] 130 130  90  40  50 100  90 120  90  90 110 100  80  80  90 110 100
## [69]  80 100 150 110 100 110 100  90 110
```

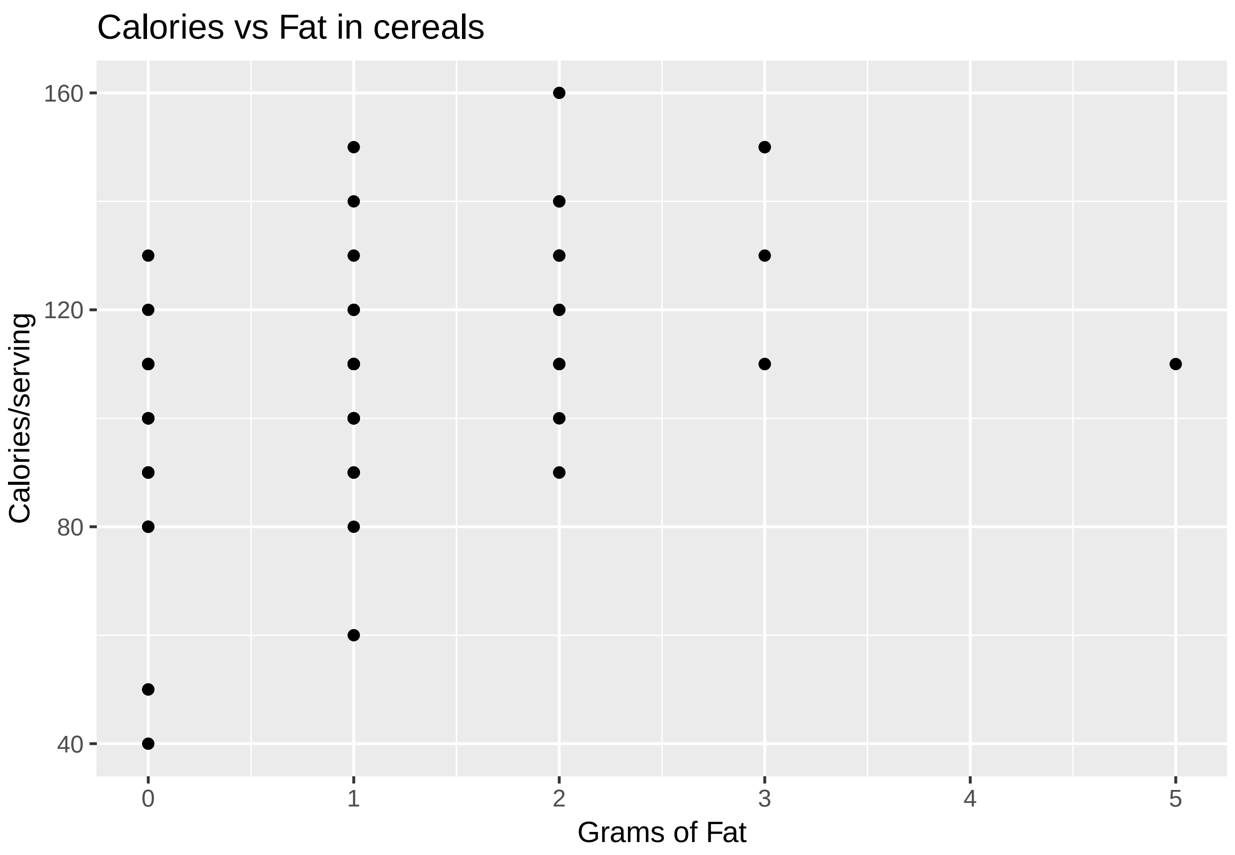
```
cereal$fat
```

```
##  [1] 1 5 1 0 2 2 0 2 1 0 2 2 3 2 1 0 0 0 1 3 0 0 1 0 1 0 0 2 0 1 0 1 1 0 3
## [36] 2 1 0 1 1 1 2 1 1 3 3 2 1 1 2 0 2 1 0 0 0 1 2 1 2 0 0 0 0 0 0 1 0 0 1
## [71] 1 1 1 1 1 1 1
```

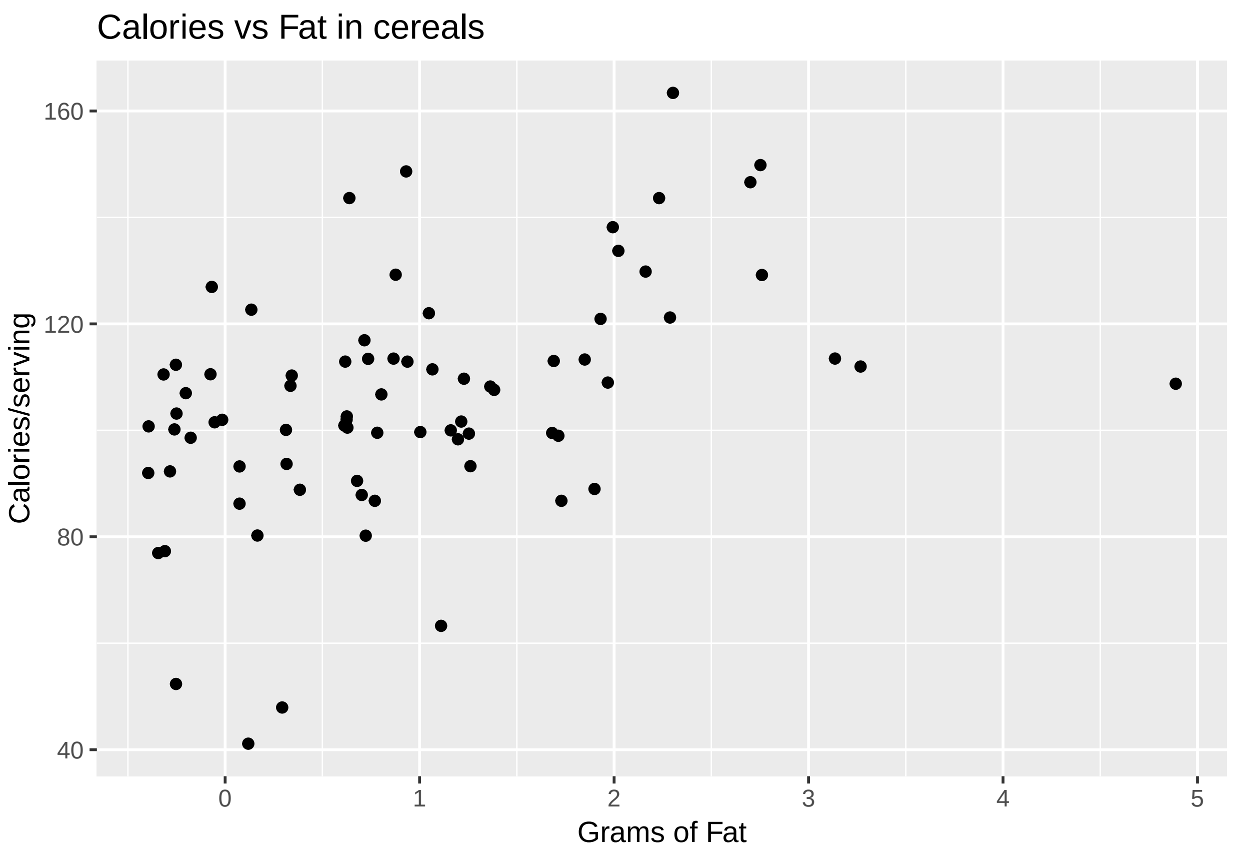
```
cereal[1:5,c("name","fat","calories")]
```

```
##                          name fat calories
## 1                   100%_Bran   1       60
## 2           100%_Natural_Bran   5      110
## 3                    All-Bran   1       80
## 4 All-Bran_with_Extra_Fiber   0       50
## 5              Almond_Delight   2      110
```

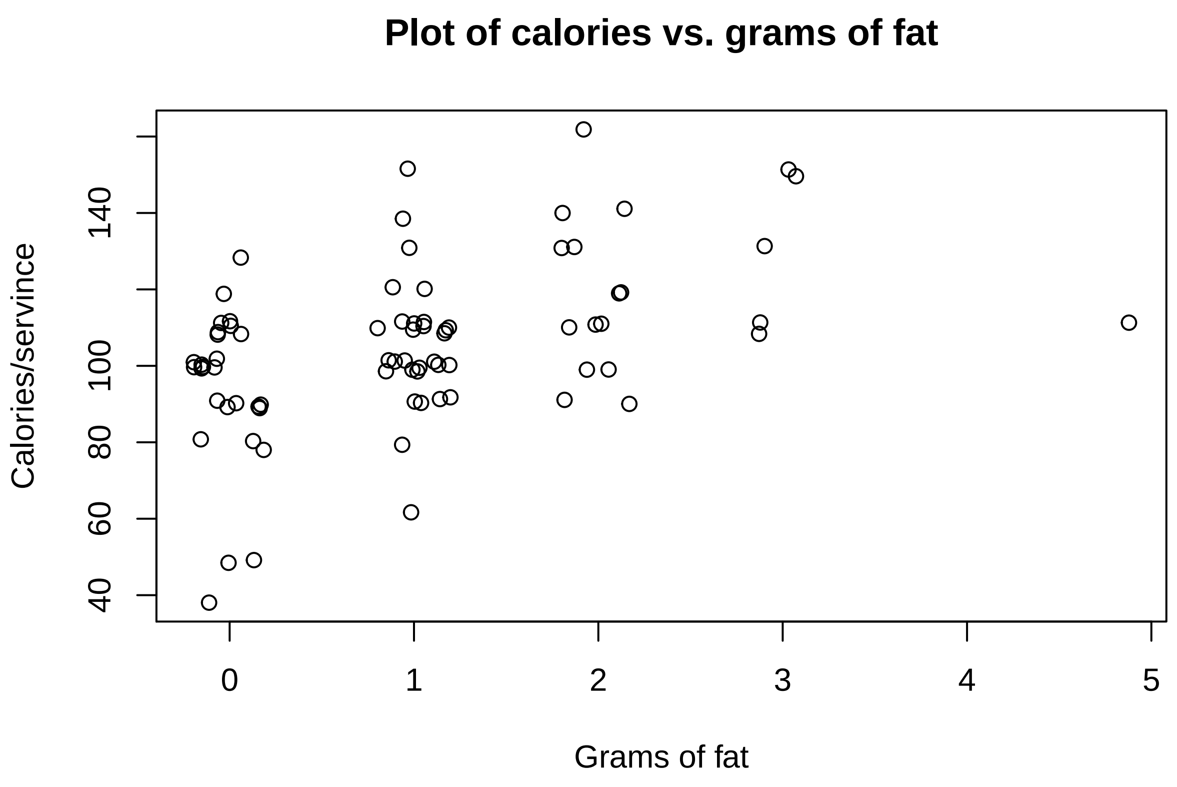
```
# Make a basic scatter plot
plotbasic <- ggplot(data=cereal, aes(x=fat, y=calories))+
      ggtitle("Calories vs Fat in cereals")+
      xlab("Grams of Fat")+ylab("Calories/serving")+
      geom_point()
plotbasic
```



```
plotbasic2 <- ggplot(data=cereal, aes(x=fat, y=calories))+
      ggtitle("Calories vs Fat in cereals")+
      xlab("Grams of Fat")+ylab("Calories/serving")+
      geom_jitter()
plotbasic2
```



```
# Same plot in base R graphics (ugh) Try to avoid using Base R graphics
plot(jitter(cereal$fat), jitter(cereal$calories),
   main="Plot of calories vs. grams of fat",
   xlab="Grams of fat", ylab='Calories/servince')
```



```r
# Fit a regression between calories and grams of fat
fit.calories.fat <- lm( calories ~ fat, data=cereal)
summary(fit.calories.fat)
```

```
##
## Call:
## lm(formula = calories ~ fat, data = cereal)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -55.132  -5.132   4.868  14.868  45.256
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)   95.132      3.141  30.285  < 2e-16 ***
## fat            9.806      2.207   4.443 3.01e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 19.36 on 75 degrees of freedom
## Multiple R-squared:  0.2084, Adjusted R-squared:  0.1978
## F-statistic: 19.74 on 1 and 75 DF,  p-value: 3.009e-05
```

```r
anova(fit.calories.fat) # careful Type I SS
```

```
## Analysis of Variance Table
##
## Response: calories
##           Df  Sum Sq Mean Sq F value    Pr(>F)
## fat        1  7402.9  7402.9  19.743 3.009e-05 ***
## Residuals 75 28121.8   375.0
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
coef(fit.calories.fat)
```

```
## (Intercept)         fat
##   95.131579    9.806005
```

```
sqrt(diag(vcov(fit.calories.fat))) # extract the SE
```

```
## (Intercept)         fat
##    3.141224    2.206897
```

```
confint(fit.calories.fat) # confidence intervals on parameters
```

```
##                 2.5 %    97.5 %
## (Intercept) 88.873939 101.38922
## fat          5.409642  14.20237
```

```
names(summary(fit.calories.fat))
```

```
##  [1] "call"          "terms"         "residuals"     "coefficients"
##  [5] "aliased"       "sigma"         "df"            "r.squared"
##  [9] "adj.r.squared" "fstatistic"    "cov.unscaled"
```

```
summary(fit.calories.fat)$r.squared
```

```
## [1] 0.2083875
```

```
summary(fit.calories.fat)$sigma
```

```
## [1] 19.36381
```

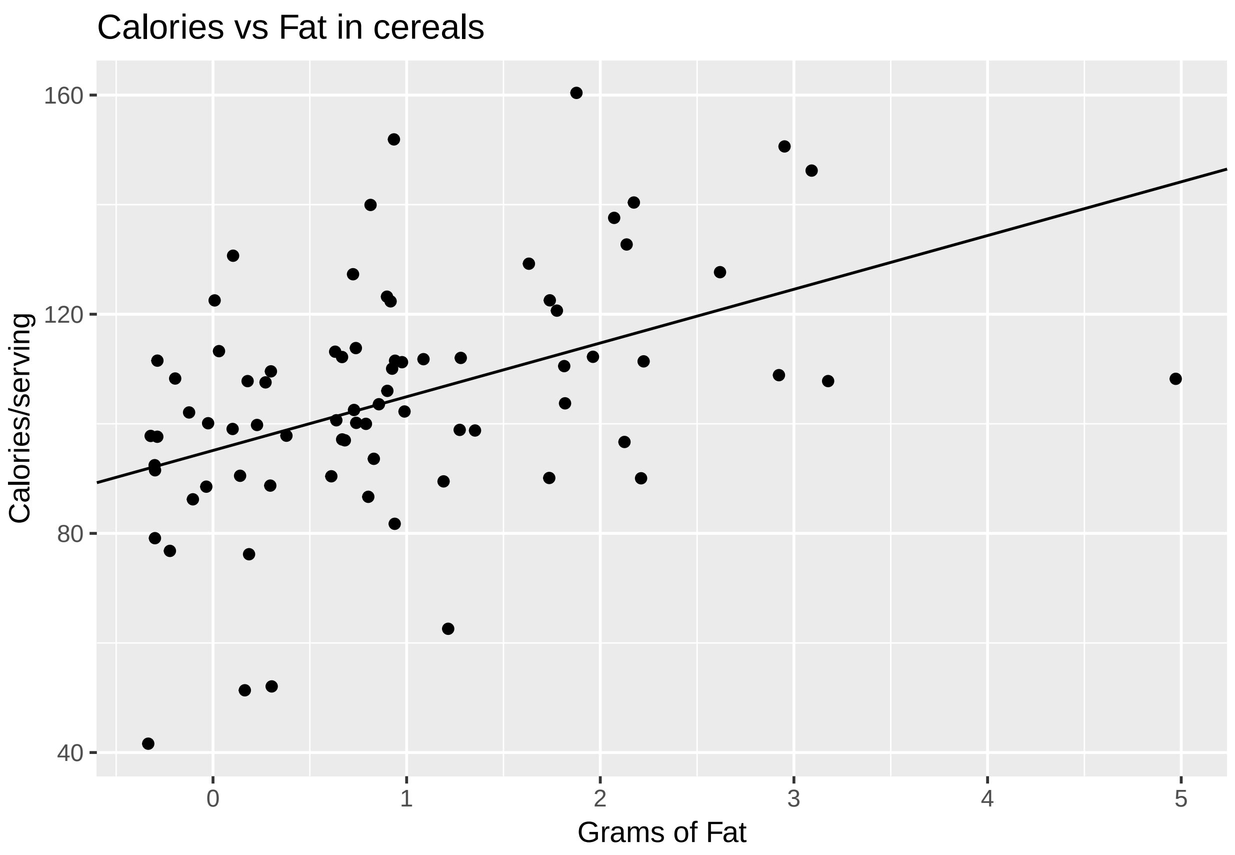
```
class(fit.calories.fat)
```

```
## [1] "lm"
```

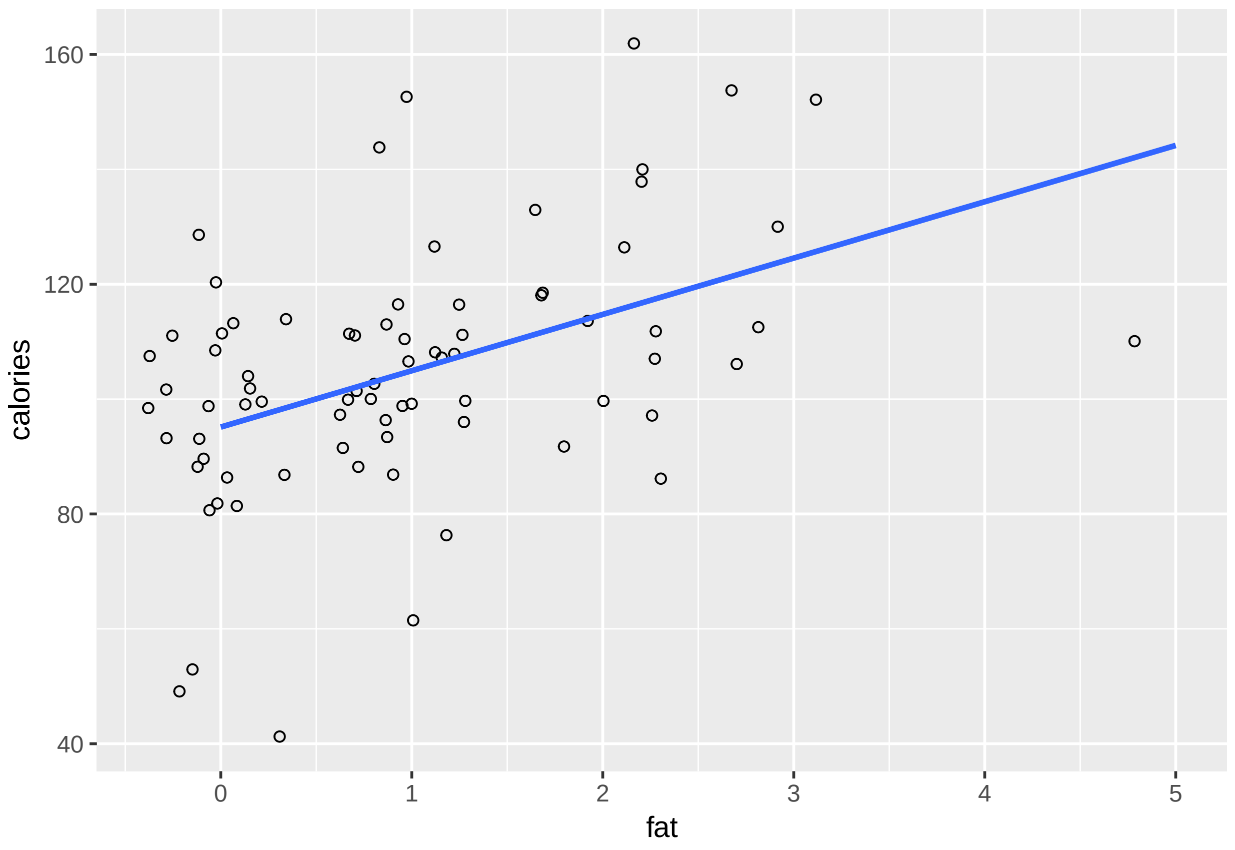
```
methods(class=class(fit.calories.fat))
```

```
##  [1] add1           alias          anova          case.names
##  [5] coerce         confint        cooks.distance deviance
##  [9] dfbeta         dfbetas        drop1          dummy.coef
## [13] effects        emm_basis      extractAIC     family
## [17] formula        fortify        hatvalues      influence
## [21] initialize     kappa          labels         logLik
## [25] model.frame    model.matrix   nobs           plot
## [29] predict        print          proj           qr
## [33] recover_data   residuals      rstandard      rstudent
## [37] show           simulate       slotsFromS3    summary
## [41] variable.names vcov
## see '?methods' for accessing help and source code
```

```
# Add the fitted line to the scatter plot; and save
plotline <- plotbasic2 +
  geom_abline(intercept=coef(fit.calories.fat)[1],
              slope    =coef(fit.calories.fat)[2])
plotline
```



```
# Or, if you don't want' to do the actual fit, use ggplot directly
plot.calories.fat <- ggplot(data=cereal, aes(x=fat, y=calories)) +
    geom_jitter(shape=1) +     # Use hollow circles
    geom_smooth(method=lm,    # Add linear regression line
                se=FALSE)     # Don't add shaded confidence region
plot.calories.fat
```



```
# Make a nicer scatter plot and add the fitted line in base R graphics. Ugh. Not recommended to use Bas
png("cal-vs-fat3-base.png")
plot(jitter(cereal$fat), jitter(cereal$calories),
   main="Plot of calories vs. grams of fat",
   xlab="Grams of fat", ylab='Calories/servince')
abline(fit.calories.fat)
dev.off()
```

```
## pdf
##   2
```

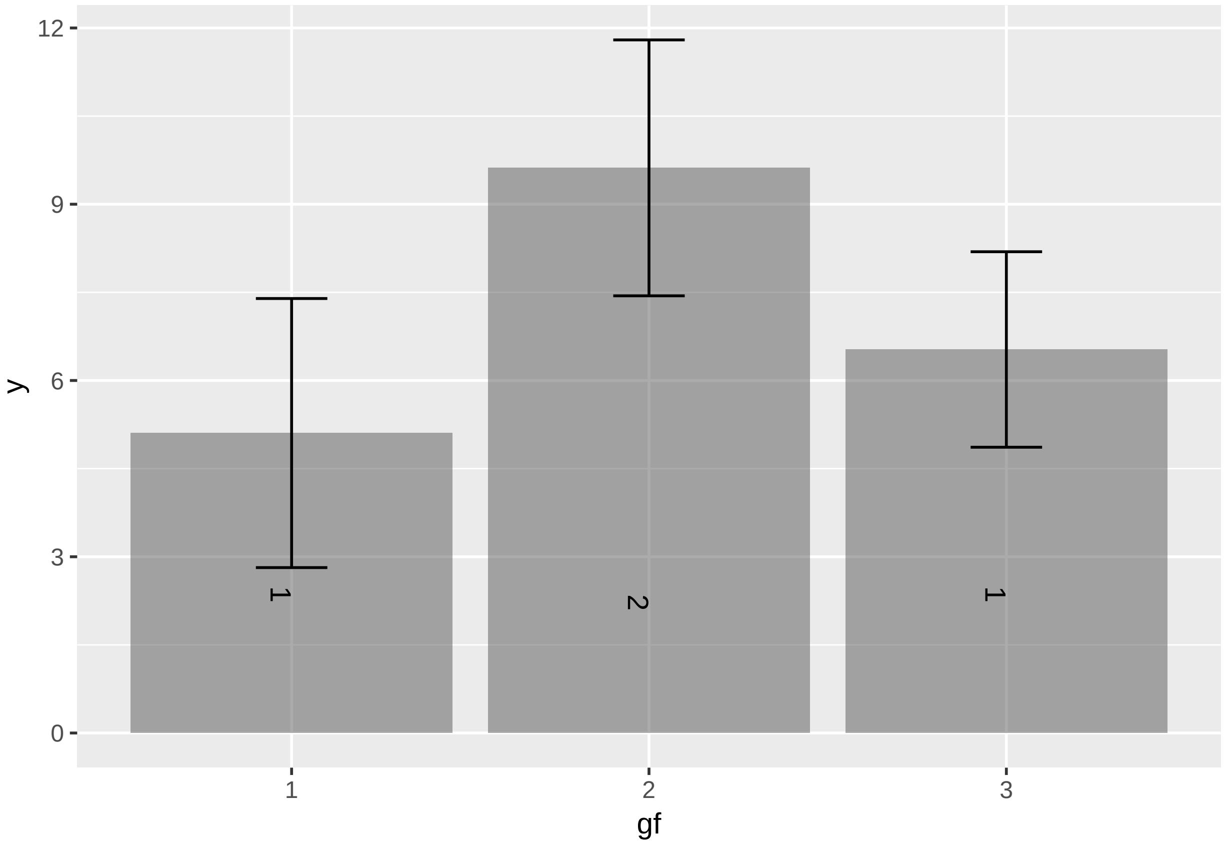
```
# Do a simple single factor ANOVA
# Is the mean number of calories the same for all shelves
# Need to use a FACTOR variable for the categorical variable
fit.sugars.shelf <- lm( sugars ~ shelfF, data=cereal)
anova(fit.sugars.shelf)
```

```
## Analysis of Variance Table
##
## Response: sugars
##           Df  Sum Sq Mean Sq F value   Pr(>F)
## shelfF     2  220.23 110.117  6.6013 0.002316 **
## Residuals 73 1217.71  16.681
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
# Estimate the marginal means along with confidence limits and Tukey multiple comparison.
fit.sugars.shelf.lsmo <- emmeans::emmeans(fit.sugars.shelf, ~shelfF)
fit.sugars.shelf.cld <- CLD(fit.sugars.shelf.lsmo, adjust='tukey')
fit.sugars.shelf.cld
```

```
##   shelfF   emmean        SE df lower.CL  upper.CL .group
##   1      5.105263 0.9369889 73 2.815493  7.395034  1
##   3      6.527778 0.6807066 73 4.864298  8.191257  1
##   2      9.619048 0.8912542 73 7.441041 11.797054   2
##
## Confidence level used: 0.95
## Conf-level adjustment: sidak method for 3 estimates
## P value adjustment: tukey method for comparing a family of 3 estimates
## significance level used: alpha = 0.05
```

```
cld.plot <- sf.cld.plot.bar(fit.sugars.shelf.cld, "shelfF", order=FALSE)
cld.plot
```



```
# Estimate the pairwise differences
pairs(fit.sugars.shelf.lsmo)
```

```
##   contrast  estimate       SE df t.ratio p.value
##   1 - 2    -4.513784 1.293168 73  -3.490  0.0023
##   1 - 3    -1.422515 1.158149 73  -1.228  0.4405
##   2 - 3     3.091270 1.121470 73   2.756  0.0199
##
## P value adjustment: tukey method for comparing a family of 3 estimates
```