Pairwise comparisons usually follow the application of some sort of linear or generalised linear model; in this setting, the 'emmeans' package (Lenth, 2020) is very handy, as it uses a very logical approach. However, we can find ourselves in the need of making pairwise comparisons between the elements of a vector, which does not came as the result of linear model fitting.

For example, we may happen to have an old table of means with standard errors and have lost the original raw data. Or, we may happen to have a vector of parameters from a nonlinear regression model, fitted with the `nls()` function. How do we make pairwise comparisons? Experienced users can make profit of the `glht()` function in the 'multcomp' package, although this is not immediate and, at least for me, it takes always some attempts to recall the exact syntax.

Therefore, I have built the `pairComp()` wrapper, which is available within the 'aomisc' package, the accompanying package for this website. Let's see how this function works by using a typical example.

A case-study

This is a real-life example, taken from a research published by Vischetti et al. in 1996 (we have used this example in other posts, before). That research considered three herbicides for weed control in sugar beet, i.e. metamitron (M), phenmedipham (P) and chloridazon (C). Four soil samples were contaminated, respectively with: (i) M alone, (ii) M + P (iii) M + C and (iv) M + P + C. The aim was to assess whether the degradation speed of metamitron in soil depended on the presence of co-applied herbicides. To reach this aim, the soil samples were incubated at 20°C and sub-samples were taken in different times after the beginning of the experiment. The concentration of metamitron in those sub-samples was measured by HPLC analyses, performed in triplicate. The resulting dataset is available within the 'aomisc' package.

In the box below. we install the 'aomisc' package from gitHub (if necessary), load it and load the 'metamitron' dataset.

```
# library(devtools)
# install_github("OnofriAndreaPG/aomisc")
library(aomisc)
data(metamitron)
head(metamitron)
##   Time Herbicide   Conc
## 1    0         M  92.00
## 2    0         M 118.64
## 3    0         M  89.58
## 4    7         M  59.32
## 5    7         M  62.95
## 6    7         M  62.95
#...
#...
tail(metamitron)
##    Time Herbicide  Conc
## 91   55       MPC 35.75
## 92   55       MPC 37.83
## 93   55       MPC 27.41
## 94   67       MPC 23.38
```

```
## 95   67       MPC 28.41
## 96   67       MPC 18.92
```

The first step we take is to fit a first-order degradation model, as follows:

$$C_{t, h} = A_h \, \exp \left(-k_h \, t \right)$$ where C is the concentration at time t for metamitron in the h^{th} combination (M alone, M + P, M + C and M + P + C), A is the initial concentration for the metamitron in the h^{th} combination, k is the degradation rate for metamitron in the h^{th} combination. This model is nonlinear and, therefore, we can use the `nls()` function for nonlinear least squares regression. The code is given below: please, note that the two parameters are followed by the name of the factor variable in square brackets (i.e.: A[Herbicide] and k[Herbicide]). This is necessary, to fit a different parameter value for each level of the 'Herbicide' factor.

```
#Fit nls grouped model
modNlin <- nls(Conc ~ A[Herbicide] * exp(-k[Herbicide] * Time),
               start=list(A=rep(100, 4), k=rep(0.06, 4)),
               data=metamitron)
summary(modNlin)
##
## Formula: Conc ~ A[Herbicide] * exp(-k[Herbicide] * Time)
##
## Parameters:
##     Estimate Std. Error t value Pr(>|t|)
## A1 9.483e+01  4.796e+00   19.77   <2e-16 ***
## A2 1.021e+02  4.316e+00   23.65   <2e-16 ***
## A3 9.959e+01  4.463e+00   22.31   <2e-16 ***
## A4 1.116e+02  4.184e+00   26.68   <2e-16 ***
## k1 4.260e-02  4.128e-03   10.32   <2e-16 ***
## k2 2.574e-02  2.285e-03   11.26   <2e-16 ***
## k3 3.034e-02  2.733e-03   11.10   <2e-16 ***
## k4 2.186e-02  1.822e-03   12.00   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 9.701 on 88 degrees of freedom
##
## Number of iterations to convergence: 5
## Achieved convergence tolerance: 7.136e-06
```

We can retrieve the degradation rates for the four herbicides (*k1*, *k2*, *k3* and *k4*) together with standard errors and load them into two vectors, as shown in the box below. In order to make pairwise comparisons, we also need to retrieve an estimate of the residual degrees of freedom, which we can also extract from the model fit object.

```
tab <- summary(modNlin)
dRates <- tab$coef[5:8,1]
SEs <- tab$coef[5:8,2]
dfr = tab$df[2]
dRates
##         k1         k2         k3         k4
## 0.04260044 0.02573512 0.03033803 0.02185935
SEs
```

```
##          k1          k2          k3          k4
## 0.004128447 0.002284696 0.002733498 0.001822218
dfr
## [1] 88
```

Now we have one vector of estimates to be compared and one vector of standard errors. In this situation, we can make pairwise comparisons by using the `pairComp()` function in the 'aomisc' package. We just have to pass the vector of model parameters, the vector of standard errors, and, optionally, the names of parameters (we do not need this, as 'dRates' is a named vector), the number of residual degrees of freedom (defaults to 'Inf') and the multiplicity adjustment method, as in the 'multcomp' package (defaults to "single-step").

```
cp <- pairComp(dRates, SEs, dfr = dfr, adjust = "holm")
cp$pairs
##
##   Simultaneous Tests for General Linear Hypotheses
##
## Linear Hypotheses:
##             Estimate Std. Error t value Pr(>|t|)
## k1-k2 == 0  0.016865   0.004718   3.574  0.00286 **
## k1-k3 == 0  0.012262   0.004951   2.477  0.04604 *
## k1-k4 == 0  0.020741   0.004513   4.596 8.58e-05 ***
## k2-k3 == 0 -0.004603   0.003563  -1.292  0.37639
## k2-k4 == 0  0.003876   0.002922   1.326  0.37639
## k3-k4 == 0  0.008479   0.003285   2.581  0.04604 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
## (Adjusted p values reported -- holm method)
```

We can also obtain a letter display, by taking the 'Letters' slot in the 'cp' object. In this case, we might like to change the yardstick protection level, by passing the 'level' argument in 'pairComp()', that defaults to 0.05.

```
cp$Letters
##          Mean          SE CLD
## k1 0.04260044 0.004128447   a
## k2 0.02573512 0.002284696  bc
## k3 0.03033803 0.002733498   b
## k4 0.02185935 0.001822218   c
```

Please, note that the `pairComp()` function can be flexibly used in every situation where we have a vector of estimates and a vector of standard errors. It yields correct results whenever the elements of the vector of estimates are uncorrelated. Hope this is useful. Thanks for reading!