Assessing the effect of an exposure on multiple outcomes (with R code)

Brice Ozenne

June 27, 2018

Abstract

We propose two strategies to assess the relationship between an exposure (e.g. a disease, a genetic factor) and a set of outcomes (e.g. a set of psychological outcomes or the binding potential measured in several brain regions) while accounting for possible risk factors and confounders. The outcomes can be continuous, categorical, or a mixture of the two but their number should be smaller than the number of observations (low-dimensional setting). This document intends to give a basic understanding of the two strategies, their limitations, and how to implement them. For now, only the "multiple univariate regressions" is presented in this document. In particular we don't claim that the proposed strategies give valid or optimal results in every application (they probably do not). Many other approaches exists (e.g. MANOVA) but won't be discussed here.

Strategies:

- *Multiple univariate regressions*: for each outcome we use a separate model to model its relationship to the exposure. An adjustment for multiple comparisons is used for drawing inference.
- *Joint model*: we model the relationship between the outcomes and the exposure in a single model. A single test can be used to test whether there is an association between the exposure and any of the outcomes.

By explicitly modeling the correlation between the outcomes, the "joint model" strategy, when valid, will provide more powerful tests compared to the "multiple univariate regressions" strategy. Another benefit of the "joint model" strategy is the use of latent variables that may reflect unmeasured biological mechanisms and therefore help the interpretation of the results. There are however drawbacks with this approach: it relies on more assumptions and a more complex statistical modeling.

Inference: The two strategies enable to quantify the association between the exposure and the outcomes and to test hypotheses such as:

- there no association between the exposure and the outcomes. When rejected, it can further test for which outcomes there is evidence for an association while adjusting for multiple comparisons in an efficient way.

- the association between the exposure and the outcomes is the same for all outcomes.

An adjustment is proposed in small sample sizes (e.g. n<100) to improve the control of the type 1 error rate. This adjustment has been shown to beneficial in several settings (using simulation studies) but does not always perfectly control the type 1 error rate. It is advised to check that validity of the adjustment when using very small samples or models with many parameters. Resampling methods (permutation or bootstrap (Carpenter and Bithell, 2000)) may be preferable in some settings.

Diagnostics: The proposed strategies uses on parametric models that rely on a set of assumptions. The violation of some of the assumptions can be tested using the observed data (often with limited power). Tools to test these assumptions are presented.

Software: we advise to use the R software to implement these strategies. It can be downloaded at https://cloud.r-project.org/. R studio provide a convenient user interface that can be downloaded at https://www.rstudio.com/products/rstudio/. The R code used to carry out the two strategies will be display in boxes:

```
1+1 ## comment about the code
```

```
[1] 2
```

while the R output will be displayed in dark red below the box.

We also recommend the following packages:

- *data.table*: for data management. See https://cran.r-project.org/web/packages/data.table/vignettes/datatable-intro.html for an introduction and https://github.com/Rdatatable/data.table/wiki for more documentation. A synthetic description of the functionalities can be found at https://s3.amazonaws.com/assets.datacamp.com/img/blog/data+table+cheat+sheet.pdf
- *ggplot2*: for data visualization. See http://r4ds.had.co.nz/data-visualisation.html for an introduction. A synthetic description of the functionalities can be found at https://www.rstudio.com/wp-content/uploads/2015/03/ggplot2-cheatsheet.pdf
- *lava*: for defining and estimating latent variable models.
- *multcomp*: for adjustments for multiple comparisons.

Note that this document should be understandable even though the reader is not familiar with these packages.

Contents

1 R settings **4**

2 Data **5**

3 Definitions and notations **6**
- 3.1 Variables . . . 6
- 3.2 Statistical model . . . 6
- 3.3 Parameter of interest . . . 8
- 3.4 Inference . . . 9
  - 3.4.1 One parameter . . . 9
  - 3.4.2 Linear combination of parameters . . . 10
  - 3.4.3 Linear combination of parameters - using a contrast matrix . . . 11
  - 3.4.4 Testing simultaneously several combination of parameters . . . 12

4 Using a separate model for each outcome **14**
- 4.1 Univariate linear regression . . . 14
  - 4.1.1 Estimation of the model . . . 14
  - 4.1.2 Fitted values . . . 15
  - 4.1.3 Diagnostics . . . 15
- 4.2 Multiple linear regressions . . . 21
- 4.3 Power and type 1 error of the procedure . . . 23

5 References **24**

A Power and type 1 error **25**
- A.1 Multiple linear regression: no adjustment vs. Bonferroni vs. Dunnett . . . 25

1 R settings

When starting a fresh R session, only the core functionalities of R are available. Additional functionalities called packages can be downloaded from the CRAN using the command `install.packages` and from github using the command `devtools::install_github` (you need to install the package *devtools* to be able to use this functionality). We will need the following packages:

```
## data management
library(data.table)

## graphical display
library(ggplot2)

## latent variable model
library(lava)

## mixed models
library(nlme)
library(lme4)

## small sample correction for variable models
library(lava)
library(lavaSearch2)

## adjustment for multiple comparisons
library(multcomp)

## diagnostics
library(qqtest) ## qqplot
library(car) ## levene test
library(butils) ## cor.testDT
## install using devtools::install_github("bozenne/butils")
library(gof) ## diagnostics based on cumulative residuals

## other
library(pbapply)
```

Even though we will not use it explicitly, it is also useful to specify the working directory:

```
path <- "c:/Users/hpl802/Documents/Github/lavaSearch2/vignettes/"
setwd(path)
```

This is the directory where, by default, R will export and import files and pictures.

2 Data

To be able to assess the validity of the proposed strategies, we will use simulated data containing:

- a variable identifying each patient: `Id`
- 10 outcomes per patient: `Y1` to `Y10`.
- 3 possible exposures per patient: `E0` that is not related to the outcomes, `E1` that has the same effect on all outcomes, and `E2` that has a different effect per outcome.

We use the `lvm` function from the *lava* package to define these variables:

```
m.sim <- lvm(Y1 ~ 0*E0 + 0.25*E1 + 0.1*E2 + 1*eta,
         Y2[0:2] ~ 0*E0 + 0.25*E1 + 0.2*E2 + 2*eta,
         Y3 ~ 0*E0 + 0.25*E1 + 0.15*E2 + 3*eta,
         Y4[0:0.5] ~ 0*E0 + 0.25*E1 + 0.175*E2 + 1*eta,
         Y5[0:3] ~ 0*E0 + 0.25*E1 + 0.075*E2 + 2*eta
         )
transform(m.sim, Id ~ eta) <- function(x){paste0("n",1:NROW(x))}
latent(m.sim) <- ~eta
```

From the code above we can see that the variance of the outcomes differs between outcomes and that the correlation between pairs of outcomes is also variable. We now simulate data using `lava::sim`:

```
set.seed(10)
dfW <- lava::sim(m.sim, n = 50, latent = FALSE)
```

We convert the resulting dataset into a `data.table` object:

```
dtW <- as.data.table(dfW)
```

and re-order its columns:

```
setcolorder(dtW, neworder = c("Id",setdiff(sort(names(dtW)),"Id")))
```

We can now display first lines of the dataset:

```
head(dtW)
```

```
   Id         E0         E1         E2         Y1         Y2         Y3        Y4         Y5
1: n1 -0.4006375 -0.7618043 -0.3911042  1.0046984  3.6867259  4.899969 0.3295657  2.073620
2: n2 -0.3345566  0.4193754 -0.2498675  0.2264810  2.0343610  1.650403 1.2908295  2.794710
3: n3  1.3679540 -1.0399434  1.1551047 -0.1255308  0.6857108  3.453420 0.1974658  6.393710
4: n4  2.1377671  0.7115740 -0.8647272  0.3643000  0.4676576  3.456675 1.6921803  2.560649
5: n5  0.5058193 -0.6332130 -0.8666783 -1.0312430 -3.4554301 -3.405372 0.1354576 -3.663814
6: n6  0.7863424  0.5631747 -2.3210170  0.7943079  2.1717773  1.602861 0.5332534 -2.430374
```

3 Definitions and notations

3.1 Variables

We can differentiate several types of random variables: outcomes, exposure, risk factors, confounders, and mediators. To explicit the difference between these types of variables we consider a set of random variables (Y, E, X_1, X_2, M) whose relationships are displayed on **??**:

- **outcome** (Y): random variables that are observed with noise. It can be for instance the 5HT-4 binding in a specific brain region. When considering several outcomes we will denote in bold variable that stands for a vector of random variables: $\mathbf{Y} = (Y_1, Y_2, \ldots, Y_m)$. This happens for instance when studying the binding in several brain regions. In such a case we expect the outcomes to be correlated.

- **exposure** (E): a variable that may affect the outcome or be associated with the outcome *and* we are interested in studying this effect/association. It can for instance be a genetic factor that is hypothesized to increase the 5HT-4 binding, or a disease like depression that is associated with a change in binding (we don't know whether one causes the other or whether they have a common cause, e.g. a genetic variant).

- **risk factor/confounder** (X_1, X_2): a variable that may affect the outcome or be associated with the outcomes *but* we are *not* interested in studying their effect/association. Risk factors (denoted by X_1) are only associated with the outcomes and confounders that are both associated with the outcome and the exposure. We usually need to account for confounders the statistical model in order to obtain unbiased estimates while accounting for risk factors only enables to obtain more precise estimates (at least in linear models).

- **mediator** (M): a variable that modulate the effect of the exposure, i.e. stands on the causal pathway between the exposure and the outcome. For instance, the permeability of the blood-brain barrier may modulate the response to drugs and can act as a mediator. It is important to keep in mind that when we are interested in the (total) effect of E on Y, we should *not* adjust the analysis on M[1]. Doing so we would remove the effect of E mediated by M and therefore bias the estimate of the total effect (we would only get the direct effect).

In the following we will assume that we do not measure any mediator variable and therefore ignore this type of variable. Also we will call covariates the variables E, X_1, X_2.

3.2 Statistical model

We will use a statistical model to relate the observed variables based a priori assumptions:

- **causal assumptions**: saying which variables are related and in which direction. This can be done by drawing a path diagram similar to **??**. In simple univariate models it may seems unnecessary to draw the path diagram since the system of variables is very simple to visualize. In multivariate model, it is often very useful to draw it. Some of these assumptions are untestable, e.g. often we cannot decide whether it is E that impacts Y or whether it is Y that impacts E just based on the data.

[1] This may not be true in specific types of confounding but we will ignore that.

- **modeling assumptions**: specifying the type of relationship between variables (e.g. linear) and the marginal or joint distribution (e.g. Gaussian). Often these assumptions can be tested and relaxed using a more flexible model. While appealing, there are some drawbacks with using a very flexible model: more data are needed to get precise estimates and the interpretation of the results is more complex.

A statistical model $\mathcal{M}$ is set of possible probability distributions. For instance when we fit a Gaussian linear model for Y_1 with just an intercept $\mathcal{M} = \left\{\mathcal{N}\left(\mu, \sigma^2\right); \mu \in \mathbb{R},\ \sigma^2 \in \mathbb{R}^+\right\}$: $\mathcal{M}$ is the set containing all possible univariate normal distributions.

The model parameters are the (non random) variables that enable the statistical model to "adapt" to different settings. They will be denoted Θ. They are the one that are estimated when we fit the statistical model using the data or that we specify when we simulate data. In the previous example, we could simulate data corresponding to a Gaussian linear model using the `rnorm` function in R:

```
rnorm
```

```
function (n, mean = 0, sd = 1)
.Call(C_rnorm, n, mean, sd)
<bytecode: 0x000000001d7eb938>
<environment: namespace:stats>
```

We would need to specify:

- n the sample size
- $\Theta = (\mu, \sigma^2)$ the model parameters, here μ corresponds to `mean` and σ to `sd`.

The true model parameters are the model parameters that have generated the observed data. They will be denoted Θ_0. For instance if in reality the binding potential is normally distributed with mean 5 and variance $2^2 = 4$, then $\Theta_0 = (\mu_0, \sigma_0^2) = (5, 4)$. Then doing our experiment we observed data such as:

```
set.seed(10)
Y_1.XP1 <- rnorm(10, mean = 5, sd = 2)
Y_1.XP1
```

```
[1] 5.037492 4.631495 2.257339 3.801665 5.589090 5.779589 2.583848 4.272648 1.746655 4.487043
```

If we were to re-do the experiment we would observe new data but Θ_0 would not change:

```
Y_1.XP2 <- rnorm(10, mean = 5, sd = 2)
Y_1.XP2
```

```
[1] 7.203559 6.511563 4.523533 6.974889 6.482780 5.178695 3.090112 4.609699 6.851043 5.965957
```

The estimated parameters are the parameters that we estimate when we fit the statistical model. They will be denoted Θ_0. We usually try to find parameters whose value maximize the chance of simulating the observed data under the estimated model (maximum likelihood estimation, MLE). For instance in the first experiment all values are positive so we would not estimate a negative mean value. In our example, $\hat{\mu}$ the MLE of μ reduces to the empirical average and $\hat{\sigma}^2$ the MLE of σ^2 to the empirical variance:

```
Theta_hat.XP1 <- c(mu_hat = mean(Y_1.XP1),
           sigma2_hat = var(Y_1.XP1))
Theta_hat.XP1
```

```
  mu_hat sigma2_hat
4.018686   1.959404
```

Clearly the estimated coefficients vary across experiments:

```
Theta_hat.XP2 <- c(mu_hat = mean(Y_1.XP2),
           sigma2_hat = var(Y_1.XP2))
Theta_hat.XP2
```

```
  mu_hat sigma2_hat
5.739183   1.799311
```

3.3 Parameter of interest

The statistical model may contain many parameters, most of them are often not of interest but are needed to obtain valid estimates (e.g. account for confounders). In most settings, the parameter of interest is one (or several) model parameter(s) - or simple transformation of them. For instance if we are interested in the average binding potential in the population our parameter of interest is μ.

Often, the aim of a study is to obtain the best estimate of the parameter of interest μ. Best means:

- **unbiased**: if we were able to replicate the study many times, i.e. get several estimates $\hat{\mu}_1, \hat{\mu}_2, \ldots, \hat{\mu}_K$, the average estimate $< \hat{\mu} >= \frac{\hat{\mu}_1 + \hat{\mu}_2 + \ldots + \hat{\mu}_K}{K}$ would coincide with the true one μ_0.
- **minimal variance**: if we were able to replicate the study many times, the variance of the estimates $\frac{(\hat{\mu}_1 - <\hat{\mu}>)^2 + \ldots + (\hat{\mu}_K - <\hat{\mu}>)^2}{K-1}$ should be as low as possible.

There will often be a trade-off between these two objectives. A very flexible method is more likely to give an unbiased estimate (e.g. being able to model non-linear relationship) at the price of greater uncertainty about the estimates. Often we favor unbiasedness over minimal variance. Indeed, if several studies are published with the same parameter of interest, one can pool the results to obtain an estimate with lower variance. Note that we have no guarantee that it will reduce the bias.

3.4 Inference

In addition to estimate the parameter(s) of interest, we often want to test hypotheses about the parameter(s) of interest.

3.4.1 One parameter

Imagine we fit a univariate linear model relating outcome 1 ($Y1$) to exposure 1 ($E1$):

$$Y1 = \alpha + \beta E1 + \varepsilon$$

where ε are the residuals that are assumed to be independent and identically distributed (iid) as well as normally distributed. Under causal assumptions (mainly no unobserved confounder) β denotes the effect of exposure 1, i.e. the change in $Y1$ for each unit increase in $E1$. In R code we can fit the linear model using:

```
e.lm <- lm(Y1 ~ E1, data = dtW)
```

```
Call:
lm(formula = Y1 ~ E1, data = dtW)

Coefficients:
(Intercept)           E1
   -0.34331     -0.08887
```

and output the estimated α and β using `coef`:

```
coef(e.lm)
```

```
(Intercept)          E1
-0.34330792 -0.08886769
```

Imagine we want to test whether there is any association between the exposure and the exposure. We want to test the null hypothesis:

$$(\mathcal{H}_0)\ \beta = 0$$

Since the parameters are estimated by ML and assuming that the model is correctly specified, we know that the asymptotic distribution of the parameter is Gaussian. This means that for large sample size, the fluctuation of the estimated values follows a normal distribution. For instance:

$$\hat{\beta} \underset{n\to\infty}{\sim} \mathcal{N}\left(\beta, \sigma^2_\beta\right)$$

where σ^2_β is the variance of the MLE, i.e. the incertainty surrounding our estimation of the association. It follows that:

$$t_\beta = \frac{\hat{\beta} - \beta}{\sigma^2_\beta} \underset{n\to\infty}{\sim} \mathcal{N}(0, 1) \tag{1}$$

So under the null hypothesis of no association between the outcome and the exposure the statistic t_β should follow a standard normal distribution. Very low or very large values are unlikely to be observed and would indicate that the null hypothesis does not hold. This is called a (univariate) Wald test. The result of this tests can be obtained using the `summary` method:

```
summary(e.lm)$coef
```

```
             Estimate Std. Error    t value   Pr(>|t|)
(Intercept) -0.34330792  0.1720656 -1.9952158 0.05171268
E1          -0.08886769  0.1787466 -0.4971712 0.62133806
```

Note: in reality R is automatically performing a correction that improves the control of the type 1 error. Indeed we usually don't know σ^2_β and plugging-in its estimate in equation (1) modifies the distribution of t_β in small samples. The correction uses a Student's t distribution instead of a Gaussian distribution.

3.4.2 Linear combination of parameters

Imagine we now want to test whether the effect of exposure 1 is different from the effect of exposure 2. We consider the following univariate linear model:

$$Y1 = \alpha + \beta_1 E1 + \beta_2 E2 + \varepsilon$$

where ε are assumed to be iid and normally distributed. In R code we can fit the linear model using:

```
e.lm <- lm(Y1 ~ E1 + E2, data = dtW)
```

We want to test the null hypothesis:

$$(\mathcal{H}_0)\ \beta_2 - \beta_1 = 0$$

Once more we use that the asymptotic distribution of the parameters is a normal distribution:

$$\begin{bmatrix}\hat{\beta}_1 \\ \hat{\beta}_2\end{bmatrix} \underset{n\to\infty}{\sim} \mathcal{N}\left(\begin{bmatrix}\hat{\beta}_1 \\ \hat{\beta}_2\end{bmatrix}, \begin{bmatrix}\sigma_1^2 & \sigma_{12}^2 \\ \sigma_{12}^2 & \sigma_2^2\end{bmatrix}\right)$$

Then $\beta_1 - \beta_2$ also follows a normal distribution:

$$\hat{\beta}_B - \hat{\beta}_A \underset{n\to\infty}{\sim} \mathcal{N}\left(\beta_B - \beta_A, \sigma_B^2 + \sigma_A^2 - 2\sigma_{AB}^2\right)$$

so:

$$t_{\beta_B-\beta_A} = \frac{\hat{\beta}_B - \hat{\beta}_A - (\beta_B - \beta_A)}{\sigma_B^2 + \sigma_A^2 - 2\sigma_{AB}^2} \underset{n\to\infty}{\sim} \mathcal{N}(0,1)$$

As before we can compute $t_{\beta_B-\beta_A}$ under the $\mathcal{H}_0$ and large or low values would be evidence for rejecting $\mathcal{H}_0$. The package *multcomp* provides a convenient interface to test linear combinations of parameters. First we specify the null hypothesis using the `glht` method:

```
e.glht <- glht(e.lm, linfct = c("E2 - E1 = 0"))
```

Then we can use the `summary` method to obtain perform the test:

```
summary(e.glht)
```

```
         Simultaneous Tests for General Linear Hypotheses

Fit: lm(formula = Y1 ~ E1 + E2, data = dtW)

Linear Hypotheses:
             Estimate Std. Error t value Pr(>|t|)
E2 - E1 == 0 -0.05015    0.21814   -0.23    0.819
(Adjusted p values reported -- single-step method)
```

3.4.3 Linear combination of parameters - using a contrast matrix

Null hypotheses can be defined using a contrast matrix. If we denote by:

$$\Theta = [\alpha\ \beta_1\ \beta_2]$$

the vector of parameters, we can define a 1-row contrast matrix:

$$c = [0\ -1\ 1]$$

such that:

$$(\mathcal{H}_0)\ c\Theta = \beta_2 - \beta_1 = 0$$

Indeed

$$c^\intercal\Theta = 0 * \alpha + -1 * \beta_1 + 1 * \beta_2 = \mu_B - \mu_A$$

So in the previous example we could have also defined a contrast matrix, e.g. using `createContrast`:

```
C <- createContrast(e.lm, par = c("E2 - E1 = 0"),
            add.variance = FALSE, rowname.rhs = FALSE)
C$contrast
```

```
          (Intercept) E1 E2
- E1 + E2           0 -1  1
```

and then call `glht` using this contrast matrix:

```
ebis.glht <- glht(e.lm, linfct = C$contrast)
summary(ebis.glht)
```

```
         Simultaneous Tests for General Linear Hypotheses

Fit: lm(formula = Y1 ~ E1 + E2, data = dtW)

Linear Hypotheses:
               Estimate Std. Error t value Pr(>|t|)
- E1 + E2 == 0 -0.05015    0.21814   -0.23    0.819
(Adjusted p values reported -- single-step method)
```

3.4.4 Testing simultaneously several combination of parameters

When we want to test several hypotheses we can stack the contrast vector of each hypothesis into the contrast matrix:

$$C = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

Then $C\Theta = 0$ is equivalent to:

$$(\mathcal{H}_{0,I})\ \beta_1 = 0$$
$$(\mathcal{H}_{0,II})\ \beta_2 = 0$$

For instance we could want to test the null hypothesis that the effect of both exposures is null:

```
C <- createContrast(e.lm, par = c("E1 = 0","E2 = 0"),
            add.variance = FALSE, rowname.rhs = FALSE)$contrast
C
```

```
   (Intercept) E1 E2
E1           0  1  0
E2           0  0  1
```

We can then call `glht` with this new contrast matrix:

```
e.glht <- glht(e.lm, linfct = C)
e.glht
```

```
         General Linear Hypotheses

Linear Hypotheses:
        Estimate
E1 == 0  -0.1628
E2 == 0  -0.2130
```

Since we perform multiple tests, we need to say to the software how to adjust for multiple comparisons. The *multcomp* package implement several approaches such as:

- **no adjustment**:

```
summary(e.glht, test = adjusted("none"))
```

```
         Simultaneous Tests for General Linear Hypotheses

Fit: lm(formula = Y1 ~ E1 + E2, data = dtW)

Linear Hypotheses:
        Estimate Std. Error t value Pr(>|t|)
E1 == 0  -0.1628     0.1905  -0.855    0.397
E2 == 0  -0.2130     0.1925  -1.107    0.274
(Adjusted p values reported -- none method)
```

- **Bonferroni** adjustment (conservative approach)

```
summary(e.glht, test = adjusted("bonferroni"))
```

```
         Simultaneous Tests for General Linear Hypotheses

Fit: lm(formula = Y1 ~ E1 + E2, data = dtW)

Linear Hypotheses:
        Estimate Std. Error t value Pr(>|t|)
E1 == 0  -0.1628     0.1905  -0.855    0.794
E2 == 0  -0.2130     0.1925  -1.107    0.548
(Adjusted p values reported -- bonferroni method)
```

- **single step Dunnett** adjustment (more efficient approach relying on asymptotic results)

```
summary(e.glht, test = adjusted("single-step"))
```

```
         Simultaneous Tests for General Linear Hypotheses

Fit: lm(formula = Y1 ~ E1 + E2, data = dtW)

Linear Hypotheses:
        Estimate Std. Error t value Pr(>|t|)
E1 == 0  -0.1628     0.1905  -0.855    0.620
E2 == 0  -0.2130     0.1925  -1.107    0.456
(Adjusted p values reported -- single-step method)
```

multcomp also implements approaches that are more powerful than the single step Dunnett but does not propose the corresponding confidence intervals (e.g. `adjusted("free")`). For this reason, they will not be presented here.

4 Using a separate model for each outcome

In this section, we will model the relationship between each of the five outcomes and the covariates using a separate linear model:

$$\begin{bmatrix} Y_1 & = \alpha_{X_1} + \beta_{Y_1,X_0} X_0 + \beta_{Y_1,X_1} X_1 + \beta_{Y_1,X_2} X_2 + \varepsilon_{Y_1} \\ Y_2 & = \alpha_{X_2} + \beta_{Y_2,X_0} X_0 + \beta_{Y_2,X_1} X_1 + \beta_{Y_2,X_2} X_2 + \varepsilon_{Y_2} \\ Y_3 & = \alpha_{X_3} + \beta_{Y_3,X_0} X_0 + \beta_{Y_3,X_1} X_1 + \beta_{Y_3,X_2} X_2 + \varepsilon_{Y_3} \\ Y_4 & = \alpha_{X_4} + \beta_{Y_4,X_0} X_0 + \beta_{Y_4,X_1} X_1 + \beta_{Y_4,X_2} X_2 + \varepsilon_{Y_4} \\ Y_5 & = \alpha_{X_5} + \beta_{Y_5,X_0} X_0 + \beta_{Y_5,X_1} X_1 + \beta_{Y_5,X_2} X_2 + \varepsilon_{Y_5} \end{bmatrix}$$

where $\varepsilon_1, \varepsilon_2, \varepsilon_3, \varepsilon_4, \varepsilon_5$ are the residual errors. The outcomes are assumed to have zero mean and finite variance, respectively, $\sigma_1^2, \sigma_2^2, \sigma_3^2, \sigma_4^2, \sigma_5^2$. Here we make no assumption on the correlation structure between the residuals.

4.1 Univariate linear regression

First, we focus on the first outcome, i.e. the first equation:

$$Y_1 = \alpha_{X_1} + \beta_{Y_1,X_0} X_0 + \beta_{Y_1,X_1} X_1 + \beta_{Y_1,X_2} X_2 + \varepsilon_{Y_1}$$

4.1.1 Estimation of the model

We can estimate the model parameters $(\alpha_{X_1}, \beta_{Y_1,X_0}, \beta_{Y_1,X_1}, \beta_{Y_1,X_2}, \sigma^2_{Y_1})$ using `lm`:

```
e.lm <- lm(Y1 ~ E0+E1+E2, data = dtW)
```

The estimate model parameters $(\hat{\alpha}_{X_1}, \hat{\beta}_{Y_1,X_0}, \hat{\beta}_{Y_1,X_1}, \hat{\beta}_{Y_1,X_2}, \hat{\sigma}^2_{Y_1})$ can be output using `summary`:

```
summary(e.lm)$coef
```

```
             Estimate Std. Error     t value   Pr(>|t|)
(Intercept) -0.38648123  0.1783576 -2.16689009 0.03545788
E0          -0.01097614  0.1799681 -0.06098939 0.95163188
E1          -0.16366671  0.1929686 -0.84815191 0.40074529
E2          -0.21367875  0.1948854 -1.09643293 0.27859637
```

or `coef`:

```
coef(e.lm)
```

```
(Intercept)          E0          E1          E2
-0.38648123 -0.01097614 -0.16366671 -0.21367875
```

Confidence intervals can be obtained with the `confint` method:

```
confint(e.lm)
```

```
                2.5 %      97.5 %
(Intercept) -0.7454964 -0.02746608
E0          -0.3732331  0.35128081
E1          -0.5520924  0.22475899
E2          -0.6059627  0.17860517
```

4.1.2 Fitted values

A fitted value, denoted $\hat{Y}$, is the expected outcome value estimated by the model. When there are covariates included in the model (e.g. X_0, X_1, X_2), the fitted value depends on the value of the covariates. In a linear model, the fitted values can be computed using:

$$\hat{Y} = \hat{\alpha}_{X_1} + \hat{\beta}_{Y_1,X_0} X_0 + \hat{\beta}_{Y_1,X_1} X_1 + \hat{\beta}_{Y_1,X_2} X_2$$

This can be done in R using the `fitted` method:

```
dtW$fit.lm <- fitted(e.lm)
dtW$fit.lm[1]
```

```
[1] -0.1738311
```

One can also compute them using:

```
coef(e.lm)["(Intercept)"] + coef(e.lm)["E0"] * dtW$E0[1] + coef(e.lm)["E1"] * dtW$E1[1]
     + coef(e.lm)["E2"] * dtW$E2[1]
```

```
(Intercept)
 -0.1738311
```

4.1.3 Diagnostics

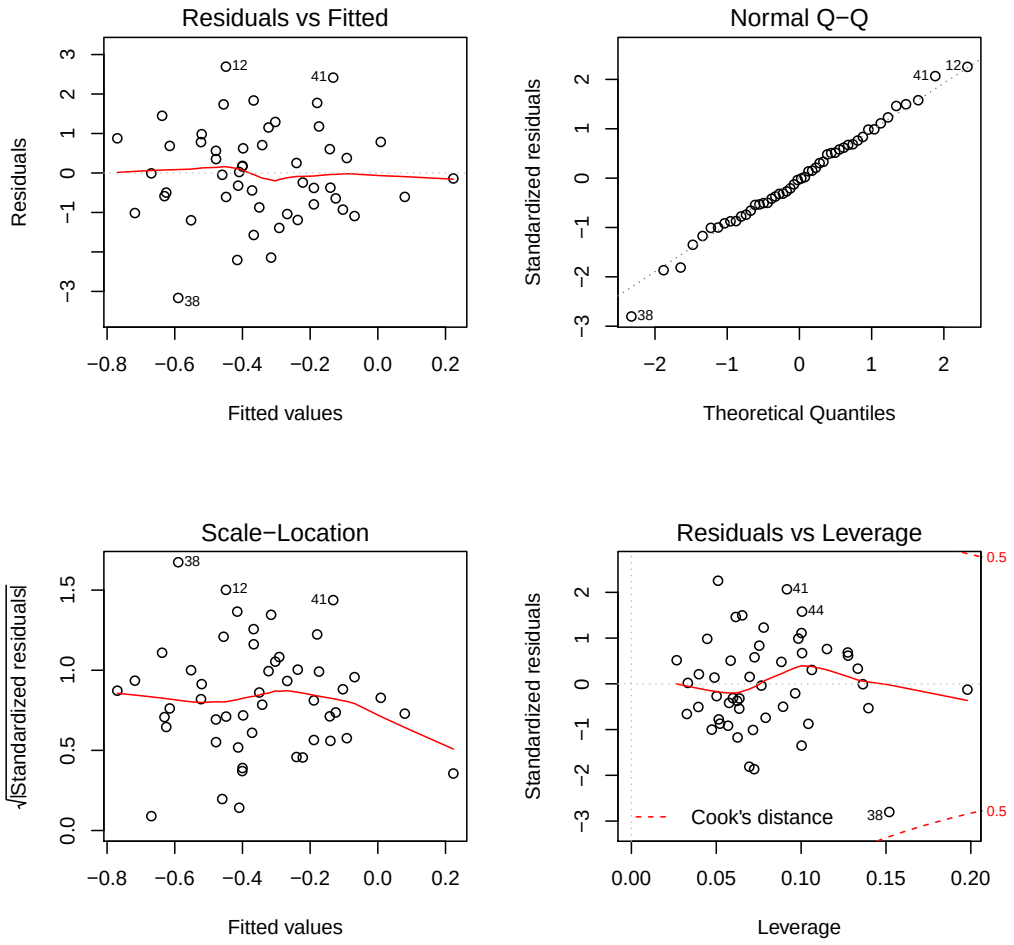
Inference using a univariate linear model rely on several assumptions:

- residuals independent and identically distributed (iid).
- residuals normally distributed.
- correct specification of the linear predictor, e.g. linearity of the effect, no interaction.
- no unobserved confounders (this is usually an untestable assumption).

Moreover, explanatory variables that are very correlated may lead to instable results.

By default R provides a graphical display that enables to check several of these assumptions:

```
par(mfrow = c(2,2))
plot(e.lm)
```



The top left plot is useful to detect a misspecification of the linear predictor (e.g. a U shape would indicate a missing quadratic effect). The top right plot enable to check the normality of the residuals, we will describe a more informative qqplot below. The bottom left can be used to detect heteroschedasticity (e.g. a trumpet shape) and the bottom right plot can be used to identify observation that have a huge influence on the fitted values.

The **qqtest** package provides a more readable qqplot. To use it, we first need to extract the residuals. This can be achieved using the `residuals` method:

```
dtW$resid.lm <- residuals(e.lm, type = "response")
```

The `type` argument indicates the type of residuals we want to extract. Raw residuals are $\hat{\varepsilon} = Y - \hat{Y}$, i.e. the observed minus the fitted values. In models more complex than a univariate linear regression, the raw residuals may not be iid. This makes it difficult to assess the validity of

the assumptions. This is why we usually display diagnostics for normalized residuals that, if the assumptions of the model are correct, should follow a standard normal distribution. For instance in the case of a univariate linear model, we can use the Pearson residuals:

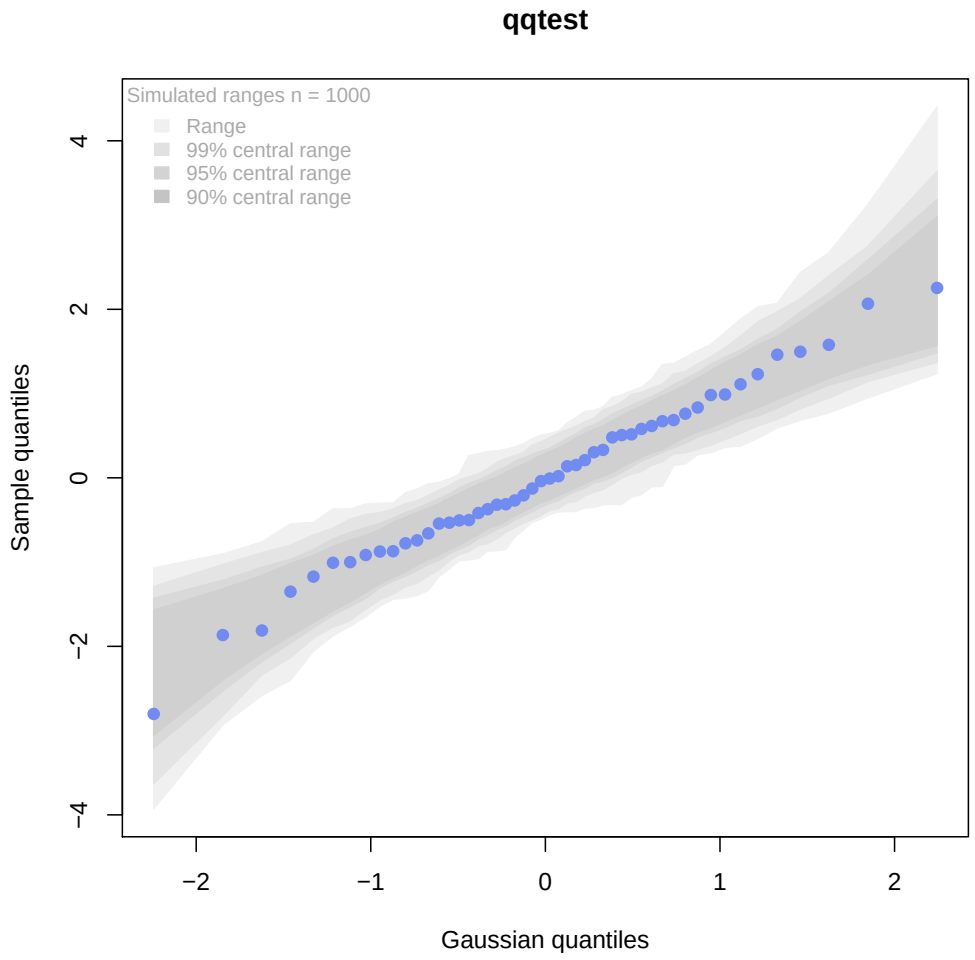
$$\hat{\varepsilon}^{Pearson} = \frac{Y - \hat{Y}}{\sqrt{\mathbb{V}ar\left[\hat{Y}\right]}} \sim \mathcal{N}(0,1)$$

In R:

```
sd.resid <- sigma(e.lm)*sqrt(1-lm.influence(e.lm, do.coef = FALSE)$hat)
dtW$resid.norm.lm <- dtW$resid.lm/sd.resid
```

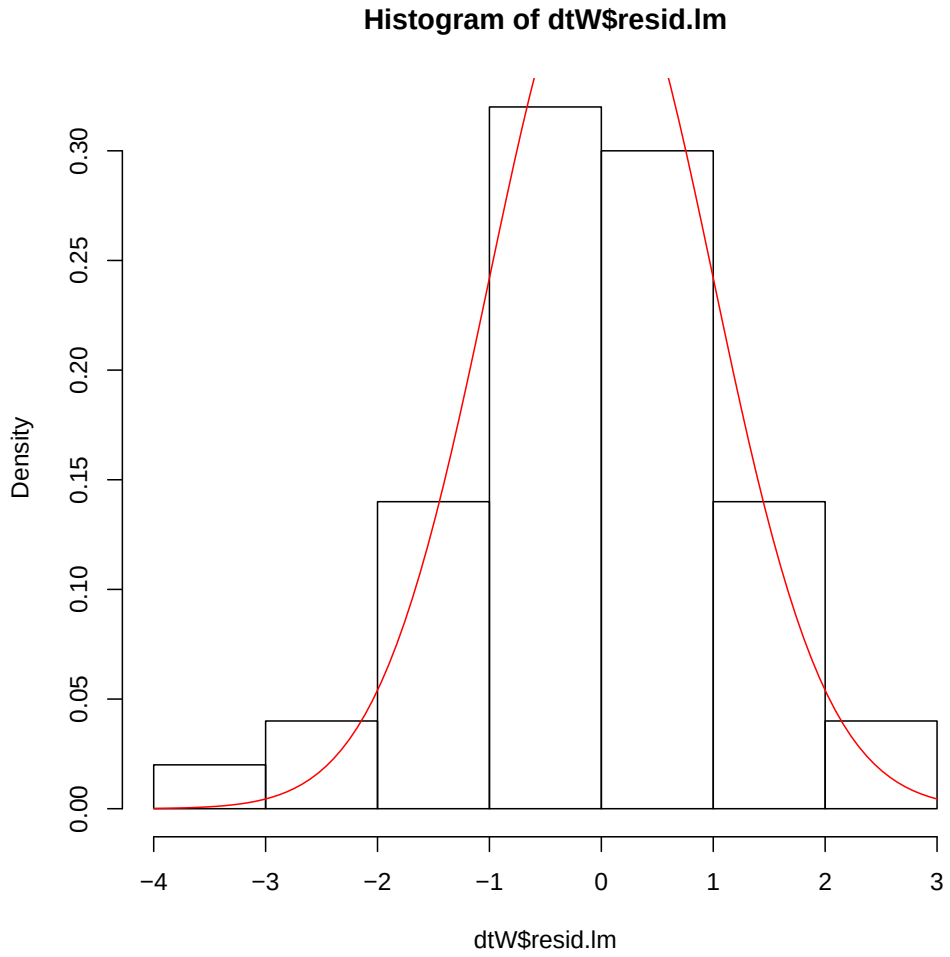
We can then obtain the qqplot using the `qqtest` function:

```
qqtest(dtW$resid.norm.lm)
```



The shaded area indicates where, if the normality assumption was correct, we would expect to observe the points. Alternatively, an histogram of the residuals can be used to assess the normality of the residuals:

```
hist(dtW$resid.lm, prob=TRUE)
curve(dnorm(x, mean=0, sd=1), add=TRUE, col = "red")
```



Statistical tests can also be used to assess deviation from normality:

```
shapiro.test(dtW$resid.lm)
```

```
        Shapiro-Wilk normality test

data:  dtW$resid.lm
W = 0.99284, p-value = 0.9899
```

Here the null hypothesis is that the residuals follow a normal distribution.

The `influence` method can be used to output what is the impact of each observation on each estimated parameter:

```
head(influence(e.lm)$coefficient)
```

```
  (Intercept)           E0           E1           E2
1  0.023113692 -0.014298361 -0.026854342 -0.014861558
2  0.013331967 -0.005188019  0.005660896  0.001314494
3  0.009519952  0.011090606 -0.004793503  0.010612459
4  0.012117058  0.036241781  0.011800078 -0.006132586
5 -0.015026578 -0.007331698  0.020875655  0.021232975
6  0.008123937  0.012157294 -0.003719040 -0.042740303
```

A statistical test can also be used to assess whether there is evidence for a more complex functional form for the linear predictor:

```
cumres(e.lm)
```

```
Kolmogorov-Smirnov-test: p-value=0.749
Cramer von Mises-test: p-value=0.764
Based on 1000 realizations. Cumulated residuals ordered by predicted-variable.
---
Kolmogorov-Smirnov-test: p-value=0.278
Cramer von Mises-test: p-value=0.27
Based on 1000 realizations. Cumulated residuals ordered by E0-variable.
---
Kolmogorov-Smirnov-test: p-value=0.413
Cramer von Mises-test: p-value=0.529
Based on 1000 realizations. Cumulated residuals ordered by E1-variable.
---
Kolmogorov-Smirnov-test: p-value=0.436
Cramer von Mises-test: p-value=0.762
Based on 1000 realizations. Cumulated residuals ordered by E2-variable.
---
```

Finally, an excessive correlation among the explanatory variables can be detected using the VIF (variance inflation factor):

```
vif(e.lm)
```

```
      E0       E1       E2
1.006096 1.146089 1.144345
```

Values higher than 5 are (arbitrarily) considered as high.

4.2 Multiple linear regressions

We now estimate all the 5 models and store them into a list:

```
ls.lm <- list(Y1 = lm(Y1 ~ E0+E1+E2, data = dtW),
              Y2 = lm(Y2 ~ E0+E1+E2, data = dtW),
              Y3 = lm(Y3 ~ E0+E1+E2, data = dtW),
              Y4 = lm(Y4 ~ E0+E1+E2, data = dtW),
              Y5 = lm(Y5 ~ E0+E1+E2, data = dtW)
              )
```

The next step would be to check the hypotheses relative to each of the models (see section 4.1.3). To perform inference in this setting we first need to define our null hypothesis or null hypotheses. Let imagine that we want to test the effect of the exposure E0 and say whether any outcome is related to the exposure. We first define a contrast matrix:

```
resC <- createContrast(ls.lm, var.test = "E1", add.variance = TRUE)
```

Since we consider separate model, we will use the element `mlf` in the output of `createContrast`:

```
resC$mlf
```

```
$Y1
   (Intercept) E0 E1 E2 sigma2
E1           0  0  1  0      0

$Y2
   (Intercept) E0 E1 E2 sigma2
E1           0  0  1  0      0

$Y3
   (Intercept) E0 E1 E2 sigma2
E1           0  0  1  0      0

$Y4
   (Intercept) E0 E1 E2 sigma2
E1           0  0  1  0      0

$Y5
   (Intercept) E0 E1 E2 sigma2
E1           0  0  1  0      0

attr(,"class")
[1] "mlf"
```

is the left hand side of the null hypothesis and:

```
resC$null
```

```
Y1: E0 Y2: E0 Y3: E0 Y4: E0 Y5: E0
     0      0      0      0      0
```

is the right hand side of the null hypothesis. We can now call `glht2`. To do so we first need to convert the list into a `mmm` object:

```
class(ls.lm) <- "mmm"
e.glht_lm <- glht2(ls.lm, linfct = resC$contrast, rhs = resC$null)
e.glht_lm
```

```
         General Linear Hypotheses

Linear Hypotheses:
            Estimate
Y1: E1 == 0 -0.16367
Y2: E1 == 0 -0.09144
Y3: E1 == 0 -0.22923
Y4: E1 == 0  0.02596
Y5: E1 == 0 -0.73954
```

We can now correct for multiple comparisons using the methods presented in section 3.4.4:

```
summary(e.glht_lm, test = adjusted("single-step"))
```

```
         Simultaneous Tests for General Linear Hypotheses

Linear Hypotheses:
            Estimate Std. Error t value Pr(>|t|)
Y1: E1 == 0 -0.16367    0.19297  -0.848    0.766
Y2: E1 == 0 -0.09144    0.40563  -0.225    0.998
Y3: E1 == 0 -0.22923    0.44975  -0.510    0.949
Y4: E1 == 0  0.02596    0.18949   0.137    1.000
Y5: E1 == 0 -0.73954    0.45807  -1.614    0.284
(Adjusted p values reported -- single-step method)
```

When all the linear model have the same number of degrees of freedom, the unadjusted p-values of this procedure will match the p-values of each model. Otherwise an approximation is made and the results may differ slightly, especially if the sample size is small. Confidence intervals can be obtained using the `confint` function:

```
confint(e.glht_lm)
```

```
         Simultaneous Confidence Intervals

Fit: NULL

Quantile = 2.4946
95% family-wise confidence level


Linear Hypotheses:
            Estimate lwr      upr
```

```
Y1: E1 == 0 -0.16367 -0.64505  0.31772
Y2: E1 == 0 -0.09144 -1.10335  0.92046
Y3: E1 == 0 -0.22923 -1.35119  0.89272
Y4: E1 == 0  0.02596 -0.44675  0.49868
Y5: E1 == 0 -0.73954 -1.88226  0.40318
```

Note that the `confint` function output confidence intervals using the (single step) Dunnett correction.

4.3 Power and type 1 error of the procedure

See appendix A.1

5 References

References

Carpenter, J. and Bithell, J. (2000). Bootstrap confidence intervals: when, which, what? a practical guide for medical statisticians. *Statistics in medicine*, 19(9):1141–1164.

A Power and type 1 error

A.1 Multiple linear regression: no adjustment vs. Bonferroni vs. Dunnett

Fonction replicating the analysis for a given sample size:

```
warper_type1power <- function(n.sample){

    ## simulate data
    iDf <- lava::sim(m.sim, n = n.sample, latent = FALSE)

    ## fit model
    iLs <- list(Y1 = lm(Y1 ~ E0+E1+E2, data = iDf),
        Y2 = lm(Y2 ~ E0+E1+E2, data = iDf),
        Y3 = lm(Y3 ~ E0+E1+E2, data = iDf),
        Y4 = lm(Y4 ~ E0+E1+E2, data = iDf),
        Y5 = lm(Y5 ~ E0+E1+E2, data = iDf)
        )
    class(iLs) <- "mmm"

    ## type 1 error
    iC.E0 <- createContrast(iLs, var.test = "E0", add.variance = TRUE)
    iGlht.E0 <- glht2(iLs, linfct = iC.E0$contrast, rhs = iC.E0$null)

    ## power
    iC.E1 <- createContrast(iLs, var.test = "E1", add.variance = TRUE)
    iGlht.E1 <- glht2(iLs, linfct = iC.E1$contrast, rhs = iC.E1$null)

    ## export
    vec.minP <- c("type1.none" = min(summary(iGlht.E0, test = adjusted("none"))$test$
    pvalues),
          "type1.bonferroni" = min(summary(iGlht.E0, test = adjusted("bonferroni"))$
    test$pvalues),
          "type1.dunnett" = min(summary(iGlht.E0, test = adjusted("single-step"))$test$
    pvalues),
          "power.none" = min(summary(iGlht.E1, test = adjusted("none"))$test$pvalues),
          "power.bonferroni" = min(summary(iGlht.E1, test = adjusted("bonferroni"))$
    test$pvalues),
          "power.dunnett" = min(summary(iGlht.E1, test = adjusted("single-step"))$test$
    pvalues))
    return(vec.minP)
}
```

Perform simulation study:

```
set.seed(10)
n.cpus <- 4
n.sim <- 1e3

cl <- snow::makeSOCKcluster(n.cpus)
```

```
doSNOW::registerDoSNOW(cl)

pb <- txtProgressBar(max = n.sim, style=3)
opts <- list(progress = function(n) setTxtProgressBar(pb, n))

ls.res <- foreach::`%dopar%`(
               foreach::foreach(i=1:n.sim,
                    .options.snow=opts,
                    .packages = c("multcomp","lavaSearch2")), {
                        warper_type1power(50)
                    })

parallel::stopCluster(cl)
M.p <- Reduce(rbind,ls.res)
```

Type 1 error:

```
colMeans(M.p[,1:3]<=0.05)
```

```
type1.none type1.bonferroni    type1.dunnett
     0.165            0.034            0.057
```

Power:

```
colMeans(M.p[,4:6]<=0.05)
```

```
power.none power.bonferroni    power.dunnett
     0.381            0.137            0.178
```