Introduction to the GLM for fMRI

CBBS Graduate Courses WiSe 2021/22

Prof. Dr. Dirk Ostwald

(3) The General Linear Model

Useful prior knowledge

Expectation and covariance

Statistical models

Foundations of Frequentist inference

Ordinary least squares and maximum likelihood estimation

Estimator properties

Foundations of hypothesis testing

T Tests

$\Rightarrow$ 1. FS BSc Psychologie | Wahrscheinlichkeitsheorie und Frequentistische Inferenz

Model formulation

Model estimation

Model evaluation

Model formulation

Definition (General Linear Model)

Let

$$y = X\beta + \varepsilon, \tag{1}$$

where

- y is an n-dimensional observable random vector referred to as *data*,
- $X \in \mathbb{R}^{n \times p}$ is a known matrix referred to as *design matrix*,
- $\beta \in \mathbb{R}^p$ is an unknown *weight parameter vector*,
- ε is an n-dimensional unobservable random vector referred to as *error*, for which it is assumed that

$$\varepsilon \sim N(0_n, \sigma^2 I_n) \text{ with unknown variance parameter } \sigma^2 > 0. \tag{2}$$

Then (1) is referred to as the *General Linear Model (GLM) in generative form*.

Remarks

- y is a random vector as it results from adding the non-random vector $X\beta \in \mathbb{R}^n$ to the random vector ε.
- Because the covariance matrix parameter of ε is assumed to be spherical, the $\varepsilon_1, \ldots, \varepsilon_n$ are independent Gaussian variables with equal variance parameters; because the expectation parameter of ε is assumed to be 0_n, the $\varepsilon_1, \ldots, \varepsilon_n$ are independent Gaussian variables.
- For each component $y_i, i = 1, \ldots, n$ of y (1) implies that

$$y_i = x_{i1}\beta_1 + x_{i2}\beta_2 + \cdots + x_{ip}\beta_p + \varepsilon_i \text{ with } \varepsilon_i \sim N(0, \sigma^2), \tag{3}$$

where x_{ij} denotes the ijth element of the design matrix X.

Theorem (GLM data distribution)

Let

$$y = X\beta + \varepsilon \text{ with } \varepsilon \sim N(0_n, \sigma^2) \quad (4)$$

denote the GLM in generative form. Then

$$y \sim N(X\beta, \sigma^2 I_n). \quad (5)$$

Proof

In terms of the linear-affine transformation theorem for Gaussian distributions, we have $\varepsilon \sim N(0_n, \sigma^2 I_n)$ and $y := I_n\varepsilon + X\beta$. Thus

$$y \sim N\left(I_n 0_n + X\beta, I_n(\sigma^2 I_n)I_n^T\right) = N(X\beta, \sigma^2 I_n). \quad (6)$$

Remarks

- The data y of a GLM is an n-dimensional Gaussian random vector with expectation parameter $X\beta \in \mathbb{R}^n$ and covariance matrix parameter $\sigma^2 I_n$. The components $y_i, ..., y_n$ of y are independent, but in general not identically distributed, random variables of the form $y_i \sim N(\mu_i, \sigma^2)$.
- Depending on $X\beta$ different well-known models (independent Gaussians, T-Tests, ANOVA, simple linear regression, multiple linear regression, ANCOVA) turn out to be special cases of the GLM; for fMRI analyses, multiple linear regression designs dominate first-level analyses, categorical designs dominate second-level analyses.

Example (1) Independent and identically distributed Gaussian random variables

Consider the scenario of n independent and identically distributed Gaussian random variables with expectation parameter $\mu \in \mathbb{R}$ and variance parameter σ^2 forming, for example, the statistical model of the one-sample T-Test,

$$y_i \sim N(\mu, \sigma^2) \text{ for } i = 1, ..., n. \tag{7}$$

Then based on the above (7) is equivalent to

$$y_i = \mu + \varepsilon_i, \varepsilon_i \sim N(0, \sigma^2) \text{ for } i = 1, ..., n \text{ and mutually independent } \varepsilon_i \tag{8}$$

and in vector-matrix form, we thus have

$$y \sim N(X\beta, \sigma^2 I_n) \text{ with } X := 1_n \in \mathbb{R}^{n \times 1}, \beta := \mu \in \mathbb{R}^1, \sigma^2 > 0. \tag{9}$$

Model formulation

Example (1) Independent and identically distributed Gaussian random variables

```
# utilities
import numpy as np                  # Numpy
import scipy.stats as rv            # scipy.stats

# model formulation
n      = 12                         # number of data points
p      = 1                          # number of beta parameters
X      = np.ones((n))               # design matrix
beta   = 2                          # true, but unknown, beta parameter
sigsqr = 1                          # true, but unknown, variance parameter

# model sampling (@ does not like n x 1 matrices)
y      = rv.multivariate_normal.rvs(X*beta, sigsqr*np.identity(n), 1)
print(y)
```

```
> [2.18723463 2.01492027 2.4052044  1.93676515 2.54410802 1.58414184
>  1.81553378 0.57943086 3.10735991 2.69698167 2.09505584 1.46873071]
```

Example (2) Simple linear regression

Consider the simple linear regression scenario

$$y_i = a + bx_i + \varepsilon_i, \varepsilon_i \sim N(0, \sigma^2) \text{ for } i = 1, ..., n \text{ and mutually independent } \varepsilon_i. \tag{10}$$

Then from the above (10) is equivalent to

$$y_i \sim N(\mu_i, \sigma^2) \text{ with } \mu_i := a + bx_i \text{ for } i = 1, ..., n \text{ with mutually independent } y_i \tag{11}$$

and in vector-matrix form, we thus have

$$y \sim N(X\beta, \sigma^2 I_n) \text{ with } X := \begin{pmatrix} 1 & x_1 \\ 1 & x_2 \\ \vdots & \vdots \\ 1 & x_n \end{pmatrix} \in \mathbb{R}^{n\times 2}, \beta := \begin{pmatrix} a \\ b \end{pmatrix} \in \mathbb{R}^2, \sigma^2 > 0. \tag{12}$$

Model formulation

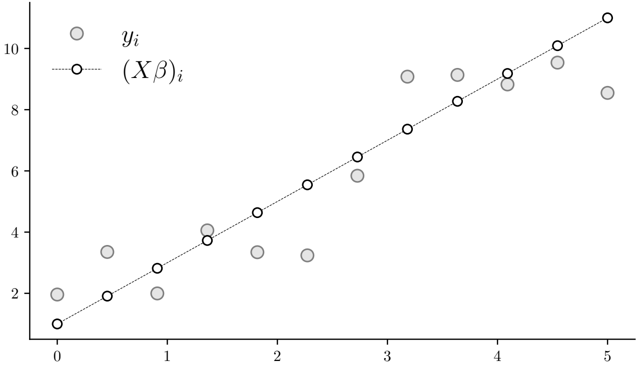
Example (2) Simple linear regression

```
# utilities
import numpy as np                                          # Numpy
import scipy.stats as rv                                    # scipy.stats

# model formulation
n      = 12                                                 # number of data points
p      = 2                                                  # number of beta parameters
x      = np.linspace(0,5,num = n)                           # regressor values
X      = np.column_stack((np.ones((n)), x))                 # design matrix
beta   = np.array([1,2])                                    # true, but unknown, beta parameter
sigsqr = 1                                                  # true, but unknown, variance parameter

# model sampling y ~ N(X\beta, \sigma^2 I_n)
y      = rv.multivariate_normal.rvs(X @ beta, sigsqr*np.identity(n), 1)
print(y)
```

```
> [ 2.14004993  2.98065167  2.31241003  2.53297888  5.8812624   6.68987246
>   7.64418444  7.83233338  8.92353783 10.44981947  9.24427335 11.37756495]
```

Model formulation

Example (2) Simple linear regression



Model formulation

Model estimation

Model evaluation

Theorem (Beta parameter estimation)

Let

$$y = X\beta + \varepsilon \text{ with } \varepsilon \sim N(0_n, \sigma^2) \tag{13}$$

denote the GLM in generative form. Then

$$\hat{\beta} := \left(X^T X\right)^{-1} X^T y \tag{14}$$

is an unbiased estimator of $\beta \in \mathbb{R}^p$.

Proof

$$\mathbb{E}(\hat{\beta}) = \mathbb{E}\left(\left(X^T X\right)^{-1} X^T y\right) = \left(X^T X\right)^{-1} X^T \mathbb{E}(y) = \left(X^T X\right)^{-1} X^T X\beta = \beta. \tag{15}$$

□

Remarks

- $\hat{\beta}$ is the ordinary least-squares estimator and the maximum likelihood estimator of β.

Model estimation

Theorem (Variance parameter estimation)

Let

$$y = X\beta + \varepsilon \text{ with } \varepsilon \sim N(0_n, \sigma^2) \tag{16}$$

denote the GLM in generative form. Then

$$\hat{\sigma}^2 := \frac{(y - X\hat{\beta})^T (y - X\hat{\beta})}{n - p} \tag{17}$$

is an unbiased estimator of $\sigma^2 > 0$.

Remarks

- We omit a proof.
- $\hat{\sigma}^2$ is the restricted maximum likelihood estimator of $\sigma^2 > 0$.
- $r := (y - X\hat{\beta}) \in \mathbb{R}^n$ is referred to as *residuals*.
- $r^T r = (y - X\hat{\beta})^T (y - X\hat{\beta})$ is called the *residual sum of squares*.
- $\hat{\sigma}^2$ is the residual sum of squares divided by $n - p$.

Model estimation

Example (1) Independent and identically distributed Gaussian random variables

Consider the scenario of n independent and identically distributed Gaussian random variables with expectation parameter $\mu \in \mathbb{R}$ and variance parameter σ^2,

$$y_i \sim N(\mu, \sigma^2) \text{ for } i = 1, ..., n. \tag{18}$$

Then, as shown below,

$$\hat{\beta} = \frac{1}{n} \sum_{i=1}^{n} y_i \text{ and } \hat{\sigma}^2 = \frac{1}{n-1} \sum_{i=1}^{n} (y_i - \bar{y})^2. \tag{19}$$

In this scenario, the beta parameter estimator thus corresponds to the sample mean and the variance parameter estimator corresponds to the sample variance, as shown below.

Model estimation

Example (1) Independent and identically distributed Gaussian random variables

For $\hat{\beta}$, we have

$$
\begin{aligned}
\hat{\beta} &= (X^T X)^{-1} X^T y \\
&= \left(\begin{pmatrix} 1 & \cdots & 1 \end{pmatrix} \begin{pmatrix} 1 \\ \vdots \\ 1 \end{pmatrix} \right)^{-1} \begin{pmatrix} 1 & \cdots & 1 \end{pmatrix} \begin{pmatrix} y_1 \\ \vdots \\ y_n \end{pmatrix} \\
&= n^{-1} \sum_{i=1}^{n} y_i \\
&= \frac{1}{n} \sum_{i=1}^{n} y_i .
\end{aligned}
$$

Example (1) Independent and identically distributed Gaussian random variables

For $\hat{\sigma}^2$, we have

$$
\begin{aligned}
\hat{\sigma}^2 & = \frac{1}{n-1}\left(y - X\hat{\beta}\right)^T\left(y - X\hat{\beta}\right) \\
& = \frac{1}{n-1}\left(\begin{pmatrix} y_1 \\ \vdots \\ y_n \end{pmatrix} - \begin{pmatrix} 1 \\ \vdots \\ 1 \end{pmatrix}\left(\frac{1}{n}\sum_{i=1}^n y_i\right)\right)^T\left(\begin{pmatrix} y_1 \\ \vdots \\ y_n \end{pmatrix} - \begin{pmatrix} 1 \\ \vdots \\ 1 \end{pmatrix}\left(\frac{1}{n}\sum_{i=1}^n y_i\right)\right) \\
& = \frac{1}{n-1}\begin{pmatrix} y_1 - \frac{1}{n}\sum_{i=1}^n y_i \\ \vdots \\ y_n - \frac{1}{n}\sum_{i=1}^n y_i \end{pmatrix}^T\begin{pmatrix} y_1 - \frac{1}{n}\sum_{i=1}^n y_i \\ \vdots \\ y_n - \frac{1}{n}\sum_{i=1}^n y_i \end{pmatrix} \\
& = \frac{1}{n-1}\sum_{i=1}^n\left(y_i - \frac{1}{n}\sum_{i=1}^n y_i\right)^2 \\
& = \frac{1}{n-1}\sum_{i=1}^n (y_i - \bar{y})^2
\end{aligned}
$$

Model estimation

Example (1) Independent and identically distributed Gaussian random variables

```
# utilities
import numpy as np                    # Numpy
import scipy.stats as rv              # scipy.stats
from numpy.linalg import inv          # matrix inversion

# model formulation
n          = 12                       # number of data points
p          = 1                        # number of beta parameters
X          = np.ones((n))             # design matrix
beta       = 2                        # true, but unknown, beta parameter
sigsqr     = 1                        # true, but unknown, variance parameter

# model sampling (@ does not like n x 1 matrices)
y          = rv.multivariate_normal.rvs(X*beta, sigsqr*np.identity(n), 1)

# model estimation
y          = np.matrix(y).T           # proper data representation
X          = np.matrix(np.ones((n,1))) # proper design matrix representation
beta_hat   = inv(X.T @ X) @ X.T @ y   # beta parameter estimator
r          = y - X @ beta_hat         # residuals
rss        = r.T @ r                  # residual sum of squares
sigsqr_hat = rss/(n-p)                # variance parameter estimator
print(beta_hat, sigsqr_hat)
```

```
> [[2.06562685]] [[1.29313485]]
```

Example (1) Independent and identically distributed Gaussian random variables

Simulation of estimator unbiasedness

```
# utilities
import numpy as np                            # Numpy
import scipy.stats as rv                      # scipy.stats
from numpy.linalg import inv                  # matrix inversion

# model formulation
n          = 12                               # number of data points
p          = 1                                # number of beta parameters
beta       = 2                                # true, but unknown, beta parameter
sigsqr     = 1                                # true, but unknown, variance parameter

# Frequentist simulations
nsim       = int(1e4)                         # number of simulations
beta_hat   = np.full((nsim), np.nan)          # beta parameter estimate array
sigsqr_hat = np.full((nsim), np.nan)          # sigma^2 parameter estimate array
for s in range(nsim):                         # simulation iterations

  # data sampling (@ does not like n x 1 matrices)
  X             = np.ones((n))                # design matrix
  y             = rv.multivariate_normal.rvs(X*beta, sigsqr*np.identity(n), 1)
  y             = np.matrix(y).T              # proper data representation

  # data analysis
  X             = np.matrix(np.ones((n,1))) # proper design matrix representation
  beta_hat[s]   = inv(X.T @ X) @ X.T @ y     # beta parameter estimator
  r             = y - X*beta_hat[s]           # residuals
  rss           = r.T @ r                     # residual sum of squares
  sigsqr_hat[s] = rss/(n-p)                   # variance parameter estimator

# estimator expectation approximation
print(np.mean(beta_hat), np.mean(sigsqr_hat))
```

```
> 1.9997606361034035 1.0012934743698905
```

Example (2) Simple linear regression

Consider the simple linear regression scenario

$$y_i = a + bx_i + \varepsilon_i, \varepsilon_i \sim N(0, \sigma^2) \text{ for } i = 1, \ldots, n \text{ and mutually independent } \varepsilon_i. \tag{20}$$

Then

$$\hat{\beta} = \begin{pmatrix} \hat{\beta}_1 \\ \hat{\beta}_2 \end{pmatrix} = \begin{pmatrix} \hat{a} \\ \hat{b} \end{pmatrix} = \begin{pmatrix} \bar{y} - \frac{c_{xy}}{s_x^2}\bar{x} \\ \frac{c_{xy}}{s_x^2} \end{pmatrix} \text{ and } \hat{\sigma}^2 = \frac{1}{n-2}\sum_{i=1}^{n}(y_i - (\hat{\beta}_0 + \hat{\beta}_1 x_i))^2 \tag{21}$$

where $\bar{x}$ and $\bar{y}$ denote the sample means of $x_1, \ldots, x_n$ and $y_1, \ldots, y_n$ and respectively, c_{xy} denotes the sample covariance of $x_1, \ldots, x_n$ and s_x^2 denotes the sample variance of $x_1, \ldots, x_n$, as shown below.

Note that for a parametric design matrix column, the beta parameter estimate is given by the sample covariance of the respective column with the data divided by the sample variance of the respective column and thus conforms to a standardized sample covariance.

Further note that we use the terms sample mean, sample variance, and sample covariance with respect to the $x_1, \ldots, x_n$ here in a mere formal manner to identify the respective formulas, but do not mean to imply that the $x_1, \ldots, x_n$ have been "sampled" in any way, as they represent known values.

Finally, note that the form of $\hat{\sigma}^2$ merely by the rules of matrix calculus.

Example (2) Simple linear regression

To show the form of the $\hat{\beta}$ estimator for simple linear regression, we first note that

$$
\begin{aligned}
c_{xy} := \sum_{i=1}^{n}(x_i - \bar{x})(y_i - \bar{y}) & = \sum_{i=1}^{n}(x_i y_i - x_i\bar{y} - \bar{x}y_i + \bar{x}\bar{y}) \\
& = \sum_{i=1}^{n} x_i y_i - \sum_{i=1}^{n} x_i\bar{y} - \sum_{i=1}^{n} \bar{x}y_i + \sum_{i=1}^{n}\bar{x}\bar{y} \\
& = \sum_{i=1}^{n} x_i y_i - \bar{y}\sum_{i=1}^{n} x_i - \bar{x}\sum_{i=1}^{n} y_i + n\bar{x}\bar{y} \\
& = \sum_{i=1}^{n} x_i y_i - \bar{y}n\bar{x} - \bar{x}n\bar{y} + n\bar{x}\bar{y} \\
& = \sum_{i=1}^{n} x_i y_i - n\bar{x}\bar{y} - n\bar{x}\bar{y} + n\bar{x}\bar{y} \\
& = \sum_{i=1}^{n} x_i y_i - n\bar{x}\bar{y},
\end{aligned}
\tag{22}
$$

Example (2) Simple linear regression

We next note that

$$
\begin{aligned}
s_x^2 := \sum_{i=1}^n (x_i - \bar{x})^2 &= \sum_{i=1}^n (x_i^2 - 2x_i\bar{x} + \bar{x}^2) \\
&= \sum_{i=1}^n x_i^2 - \sum_{i=1}^n 2x_i\bar{x} + \sum_{i=1}^n \bar{x}^2 \\
&= \sum_{i=1}^n x_i^2 - 2\bar{x}\sum_{i=1}^n x_i + n\bar{x}^2 \\
&= \sum_{i=1}^n x_i^2 - 2\bar{x}n\bar{x} + n\bar{x}^2 \\
&= \sum_{i=1}^n x_i^2 - 2n\bar{x}^2 + n\bar{x}^2 \\
&= \sum_{i=1}^n x_i^2 - n\bar{x}^2 .
\end{aligned}
\tag{23}
$$

Example (2) Simple linear regression

From the definition of $\hat{\beta}$, we have

$$
\begin{aligned}
\hat{\beta} &= (X^T X)^{-1} X^T y \\
&= \left(\begin{pmatrix} 1 & \cdots & 1 \\ x_1 & \cdots & x_n \end{pmatrix} \begin{pmatrix} 1 & x_1 \\ \vdots & \vdots \\ 1 & x_n \end{pmatrix} \right)^{-1} \begin{pmatrix} 1 & \cdots & 1 \\ x_1 & \cdots & x_n \end{pmatrix} \begin{pmatrix} y_1 \\ \vdots \\ y_n \end{pmatrix} \\
&= \begin{pmatrix} n & \sum_{i=1}^n x_i \\ \sum_{i=1}^n x_i & \sum_{i=1}^n x_i^2 \end{pmatrix}^{-1} \begin{pmatrix} \sum_{i=1}^n y_i \\ \sum_{i=1}^n x_i y_i \end{pmatrix} \\
&= \begin{pmatrix} n & n\bar{x} \\ n\bar{x} & \sum_{i=1}^n x_i^2 \end{pmatrix}^{-1} \begin{pmatrix} n\bar{y} \\ \sum_{i=1}^n x_i y_i \end{pmatrix}.
\end{aligned}
\tag{24}
$$

The inverse of $X^T X$ is given by

$$
\frac{1}{s_x^2} \begin{pmatrix} \frac{s_x^2}{n} + \bar{x}^2 & -\bar{x} \\ -\bar{x} & 1 \end{pmatrix},
\tag{25}
$$

Example (2) Simple linear regression

because

$$
\begin{aligned}
& \frac{1}{s_x^2}\begin{pmatrix} \frac{s_x^2}{n} + \bar{x}^2 & -\bar{x} \\ -\bar{x} & 1 \end{pmatrix}\begin{pmatrix} n & n\bar{x} \\ n\bar{x} & \sum_{i=1}^n x_i^2 \end{pmatrix} \\
& = \frac{1}{s_x^2}\begin{pmatrix} \frac{ns_x^2}{n} + n\bar{x}^2 - n\bar{x}^2 & \frac{s_x^2 n\bar{x}}{n} + n\bar{x}^2\bar{x} - \bar{x}\sum_{i=1}^n x_i^2 \\ -\bar{x}n + n\bar{x} & -n\bar{x}^2 + \sum_{i=1}^n x_i^2 \end{pmatrix} \\
& = \frac{1}{s_x^2}\begin{pmatrix} s_x^2 & s_x^2\bar{x} - \bar{x}\left(\sum_{i=1}^n x_i^2 - n\bar{x}^2\right) \\ 0 & \sum_{i=1}^n x_i^2 - n\bar{x}^2 \end{pmatrix} \\
& = \frac{1}{s_x^2}\begin{pmatrix} s_x^2 & s_x^2\bar{x} - \bar{x}s_x^2 \\ 0 & s_x^2 \end{pmatrix} \\
& = \frac{1}{s_x^2}\begin{pmatrix} s_x^2 & 0 \\ 0 & s_x^2 \end{pmatrix} \\
& = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}.
\end{aligned} \tag{26}
$$

Model estimation

Example (2) Simple linear regression

We thus have

$$
\begin{aligned}
\hat{\beta} = \begin{pmatrix} \frac{1}{n} + \frac{\bar{x}^2}{s_x^2} & -\frac{\bar{x}}{s_x^2} \\ -\frac{\bar{x}}{s_x^2} & \frac{1}{s_x^2} \end{pmatrix} \begin{pmatrix} n\bar{y} \\ \sum_{i=1}^n x_i y_i \end{pmatrix}
& = \begin{pmatrix} \left(\frac{1}{n} + \frac{\bar{x}^2}{s_x^2}\right) n\bar{y} - \frac{\bar{x}\sum_{i=1}^n x_i y_i}{s_x^2} \\ \frac{\sum_{i=1}^n x_i y_i}{s_x^2} - \frac{n\bar{x}\bar{y}}{s_x^2} \end{pmatrix} \\
& = \begin{pmatrix} \frac{n\bar{y}}{n} + \frac{\bar{x}^2 n\bar{y}}{s_x^2} - \frac{\bar{x}\sum_{i=1}^n x_i y_i}{s_x^2} \\ \frac{\sum_{i=1}^n x_i y_i - n\bar{x}\bar{y}}{s_x^2} \end{pmatrix} \\
& = \begin{pmatrix} \bar{y} + \frac{\bar{x}n\bar{x}\bar{y} - \bar{x}\sum_{i=1}^n x_i y_i}{s_x^2} \\ \frac{\sum_{i=1}^n x_i y_i - n\bar{x}\bar{y}}{s_x^2} \end{pmatrix} \\
& = \begin{pmatrix} \bar{y} - \frac{\sum_{i=1}^n x_i y_i - n\bar{x}\bar{y}}{s_x^2}\bar{x} \\ \frac{\sum_{i=1}^n x_i y_i - n\bar{x}\bar{y}}{s_x^2} \end{pmatrix} \\
& = \begin{pmatrix} \bar{y} - \frac{c_{xy}}{s_x^2}\bar{x} \\ \frac{c_{xy}}{s_x^2} \end{pmatrix}.
\end{aligned} \tag{27}
$$

Model estimation

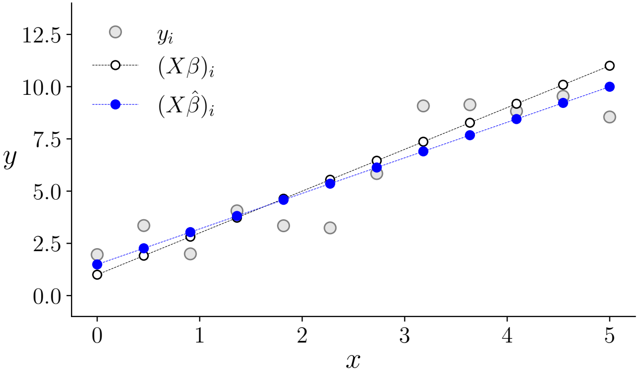
Example (2) Simple linear regression

```
# utilities
import numpy as np                                      # Numpy
import scipy.stats as rv                                # scipy.stats
from numpy.linalg import inv                            # matrix inversion

# model formulation
n          = 12                                         # number of data points
p          = 2                                          # number of beta parameters
x          = np.linspace(0,5,num = n)                   # regressor values
X          = np.column_stack((np.ones((n)), x))         # design matrix
beta       = np.array([1,2])                            # true, but unknown, beta parameter
sigsqr     = 1                                          # true, but unknown, variance parameter

# model sampling y ~ N(X\beta, \sigma^2 I_n)
y          = rv.multivariate_normal.rvs(X @ beta, sigsqr*np.identity(n), 1)

# model estimation
beta_hat   = inv(X.T @ X) @ X.T @ y                     # beta parameter estimator
r          = y - X @ beta_hat                           # residuals
rss        = r.T @ r                                    # residual sum of squares
sigsqr_hat = rss/(n-p)                                  # variance parameter estimator
print(beta_hat, sigsqr_hat)
```

```
> [0.10245232 2.37917839] 1.517653299083768
```

Model formulation

Example (2) Simple linear regression



Model formulation

Model estimation

Model evaluation

Definition (T statistic)

Let

$$y = X\beta + \varepsilon \text{ with } \varepsilon \sim N(0_n, \sigma^2 I_n) \tag{28}$$

denote the GLM and let

$$\hat{\beta} := \left(X^T X\right)^{-1} X^T y \text{ and } \hat{\sigma}^2 := \frac{(y - X\hat{\beta})^T (y - X\hat{\beta})}{n - p} \tag{29}$$

denote the beta and variance parameter estimators. Then for a vector $c \in \mathbb{R}^p$ of *contrast weights*, the *T statistic* is defined as

$$T := \frac{c^T \hat{\beta}}{\sqrt{\hat{\sigma}^2 c^T \left(X^T X\right)^{-1} c}}. \tag{30}$$

Remarks

- The T-statistic is depends on the data y via $\hat{\beta}$ and $\hat{\sigma}^2$.
- The contrast weight vector projects the beta parameter estimates onto a scalar value.
- In general, the following intuition is helpful

$$T = \frac{\text{Effect estimate}}{\text{Sample size scaled data variability}}. \tag{31}$$

- Intuitively, a T-statistic thus represents a signal-to-noise ratio.

Model evaluation

Example

Consider the GLM representation of n independent and identically distributed Gaussian random variables

$$y \sim N(X\beta, \sigma^2 I_n) \text{ with } X := 1_n \in \mathbb{R}^{n \times 1}, \beta := \mu \in \mathbb{R}^1, \sigma^2 > 0 \tag{32}$$

and let $c := 1$. As seen above, here

$$\hat{\beta} = \frac{1}{n} \sum_{i=1}^{n} y_i = \bar{y}. \tag{33}$$

Thus

$$T = \frac{c^T \bar{y}}{\sqrt{\hat{\sigma}^2 c^T \left(1_n^T 1_n\right)^{-1} c}} = \frac{\bar{y}}{\sqrt{\hat{\sigma}^2 (n)^{-1}}} = \frac{\bar{y}}{\hat{\sigma}/\sqrt{n}}, \tag{34}$$

corresponding to the well-known one-sample T-Test statistic. The GLM scenario of n independent and identically distributed Gaussian random variables is thus also referred to as a *one-sample T-Test design*.

Finally, note that

$$T = \sqrt{n} d \text{ with } d := \frac{\bar{y}}{\hat{\sigma}}. \tag{35}$$

In the current scenario, the T-statistic can thus be regarded as a sample-size adjusted effect size, as measured by Cohen's d. Vice versa, Cohen's d may be regarded as a sample-size independent T-statistic.

Theorem (T-test statistic null distribution)

Let

$$y = X\beta + \varepsilon \text{ with } \varepsilon \sim N(0_n, \sigma^2 I_n) \tag{36}$$

denote the GLM and let

$$T := \frac{c^T\hat{\beta} - c^T\beta_0}{\sqrt{\hat{\sigma}^2 c^T \left(X^T X\right)^{-1} c}}. \tag{37}$$

denote the *T-test statistic*. Then if

$$c^T\beta = c^T\beta_0, \tag{38}$$

T is distributed according to a Student t-distribution with degrees of freedom parameter $n - p$, for which we write

$$T \sim t(n-p). \tag{39}$$

Remarks

- In lieu of a proof, we exemplify the theorem using simulations below.
- A low probability of observing a given value of the T test statistic under the assumption $c^T\beta = c^T\beta_0$ as evaluated by means of $t(n-p)$ is viewed as evidence against the null hypothesis.

Model evaluation

Example

$$y \sim N(X\beta, \sigma^2 V),\ X \in \mathbb{R}^{n\times 2},\ n = 12,\ \beta := (2,2)^T,\ \sigma^2 := 1, \text{ and}$$

$$\beta_0 := (1,1) \text{ for } c_1 := (1,0)^T,\ c_2 := (0,1)^T,\ c_3 := (1,-1)^T$$

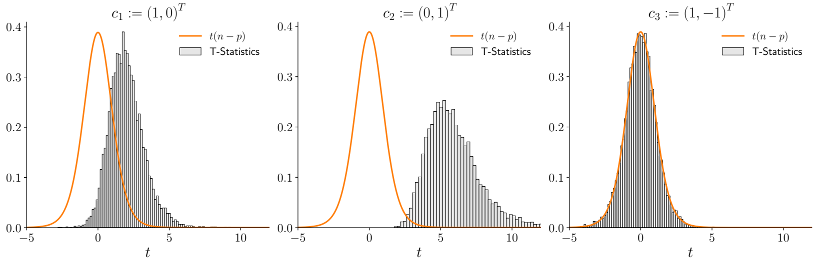
```
# model formulation
n      = 12                                                        # number of data points
p      = 2                                                         # number of beta parameters
x      = np.linspace(0,5,num = n)                                  # regressor values
X      = np.column_stack((np.ones((n)), x))                        # design matrix
beta   = np.array([2,2])                                           # true, but unknown, beta parameter
sigsqr = 1                                                         # variance parameter
Sigma  = sigsqr*np.identity(n)                                     # covariance matrix parameter
beta_0 = np.array([1,1])                                           # null hypothesis parameter
c      = np.array([[1,0], [0,1], [1,-1]]).T                        # contrast weight vectors

# simulation parameters
seed   = np.random.seed(1)                                         # random number generator
nsim   = int(1e4)                                                  # number of simulations

# simulation
T      = np.full((nsim, c.shape[1]), np.nan)                       # T-statistics array
for i in range(c.shape[1]):                                        # contrast weight vector iterations
    for s in range(nsim):                                          # simulation iterations
        y          = rv.multivariate_normal.rvs(X @ beta, Sigma, 1) # model sampling
        beta_hat   = inv(X.T @ X) @ X.T @ y                        # beta parameter estimator
        sigsqr_hat = ((y - X @ beta_hat).T @ (y - X @ beta_hat))/(n-p) # variance parameter estimator
        T[s,i]     = ((c[:,i].T @ beta_hat - c[:,i].T @ beta_0)/   # T-test statistic numerator
                     (np.sqrt(sigsqr_hat*c[:,i].T@ inv(X.T@ X)@c[:,i]))) # T-test statistic denominator

# t(n-p) density
t_min  = -5                                                        # minimum t-value
t_max  =  12                                                       # maximum t-value
t_res  = int(1e3)                                                  # t-space resolution
t      = np.linspace(t_min,t_max, t_res)                           # t-space
pt     = rv.t.pdf(t, n-p)                                          # t probability density function
```

Model evaluation

Example

$$y \sim N(X\beta, \sigma^2 V),\ X \in \mathbb{R}^{n\times 2}, n = 12, \beta := (2,2)^T, \sigma^2 := 1, \text{ and}$$

$$\beta_0 := (1,1) \text{ for } c_1 := (1,0)^T, c_2 := (0,1)^T, c_3 := (1,-1)^T$$



Model formulation

Model estimation

Model evaluation