Linear and generalized linear models

Saturday 3 June, 2023

Esa Läärä

Statistical Practice in Epidemiology using **R**
2 to 7 June, 2023
University of Tartu, Estonia

Outline

- Simple linear regression.
- Fitting a regression model and extracting results.
- Predictions and diagnostics.
- Categorical factors and contrast matrices.
- Main effects and interactions.
- Modelling curved effects.
- Generalized linear models.
- Binary regression and Poisson regression.

Variables in generalized linear models

- The **outcome** or **response** variable must be numeric.
- Main types of response variables are
  - Metric or continuous (a measurement with units).
  - Binary (“yes” vs. ”no”, coded 1/0), or proportion.
  - Failure in person-time, or incidence rate.
- **Explanatory** variables or **regressors** can be
  - Numeric or quantitative variables
  - Categorical factors, represented by class indicators or contrast matrices.

The births data in Epi

`id`: Identity number for mother and baby.

`bweight`: Birth weight of baby.

`lowbw`: Indicator for birth weight less than 2500 g.

`gestwks`: Gestation period in weeks.

`preterm`: Indicator for gestation period less than 37 weeks.

`matage`: Maternal age.

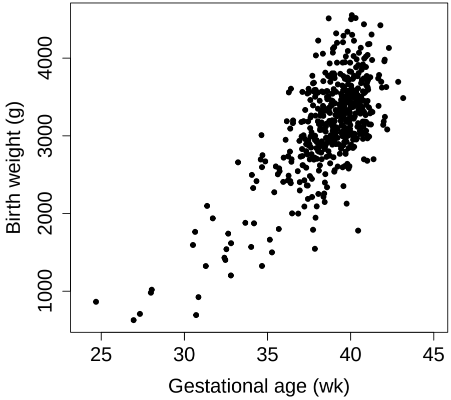
`hyp`: Indicator for maternal hypertension (0 = no, 1 = yes).

`sex`: Sex of baby (1 = male, 2 = female).

Declaring and transforming some variables as factors:

```
> library(Epi) ; data(births)
> births <- transform(births,
+    hyp = factor(hyp, labels=c("N", "H")),
+    sex = factor(sex, labels=c("M", "F")),
+    gest4 = cut(gestwks,breaks=c(20, 35, 37, 39, 45), right=FALSE)  )
> births <- subset(births, !is.na(gestwks))
```

Birth weight and gestational age



```
> with(births, plot(bweight ~ gestwks, xlim = c(24,45), pch = 16, cex.axis=1.5, cex.lab = 1.5,
+  xlab= "Gestational age (wk)", ylab= "Birth weight (g)" ) )
```

Metric response, numeric explanatory variable

Roughly linear relationship btw `bweight` and `gestwks`

→ Simple **linear regression model** fitted.

```
> m <- lm(bweight ~ gestwks, data=births)
```

- `lm()` is the function that fits linear regression models, assuming **Gaussian** distribution or **family** for **error** terms.
- `bweight ~ gestwks` is the **model formula**
- `m` is a **model object** belonging to **class** “`lm`”.

`> coef(m)` – Printing the estimated regression coefficients

```
(Intercept)      gestwks
    -4489.1        197.0
```

Interpretation of **intercept** and **slope**?

Model object and extractor functions

Model object = **list** of different elements, each being separately accessible.
– See `str(m)` for the full list.

Functions that extract results from the fitted model object

- `summary(m)` – lots of output
- `coef(m)` – beta-hats only (see above)
- `ci.lin(m)[,c(1,5,6)]` – $\widehat{\beta}_j$s plus confidence limits

  ```
              Estimate    2.5%   97.5%
  (Intercept)  -4489.1 -5157.3 -3821.0
  gestwks        197.0   179.7   214.2
  ```

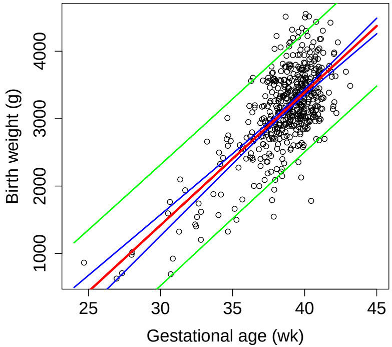
  Function `ci.lin()` is found in `Epi` package.
- `anova(m)` – Analysis of Variance Table

Other extractor functions, for example

- `fitted(m)`, `resid(m)`, `vcov(m)`, ...
- `predict(m, newdata = ..., interval=...)`
  - Predicted responses for desired combinations of new values of the regressors – `newdata`
  - Argument `interval` specifies whether **confidence** intervals for the *mean* response or **prediction** intervals for *individual* responses are returned.
- `plot(m)` – produces various diagnostic plots based on residuals (raw, standardized or studentized residuals).

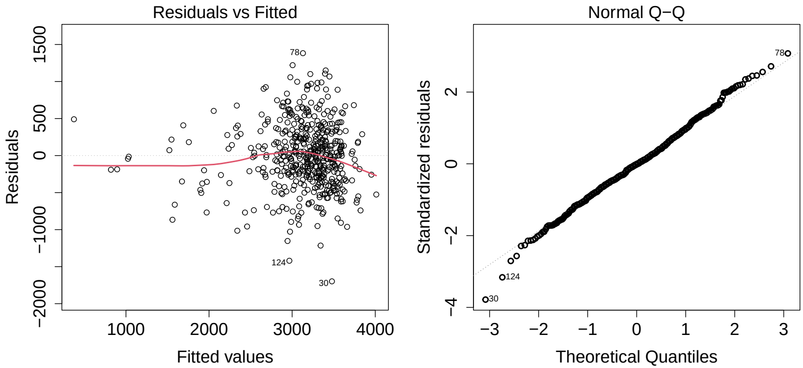
Many of these are special **methods** for certain **generic functions**, aimed at acting on objects of class "`lm`".

Fitted values, confidence & prediction intervals



```
> nd <- data.frame( gestwks = seq(24, 45, by = 0.25 ) )
> pr.c1 <- predict( m, newdata=nd, interval="conf" )
> pr.p1 <- predict( m, newdata=nd, interval="pred" )
> with(births, plot(bweight ~ gestwks, xlim = c(24,45), cex.axis=1.5, cex.lab = 1.5, xlab = 'Gestation
> matlines( nd$gestwks, pr.c1, lty=1, lwd=c(3,2,2), col=c('red','blue','blue'))
> matlines( nd$gestwks, pr.p1, lty=1, lwd=c(3,2,2), col=c('red','green','green'))
```

A couple of diagnostic plots



```
> par(mfrow=c(1,2))
> plot(m, 1:2, cex.lab = 1.5, cex.axis=1.5, cex.caption=1.5, lwd=2)
```

- Some deviation from linearity?
- Reasonable agreement with Gaussian error assumption?

Factor as an explanatory variable

- How bweight depends on maternal hypertension?

  ```
  > mh <- lm( bweight ~ hyp, data=births)

                Estimate   2.5%  97.5%
  (Intercept)     3198.9 3140.2 3257.6
  hypH            -430.7 -585.4 -275.9
  ```

- Removal of intercept → mean bweights by hyp:

  ```
  > mh2 <- lm( bweight ~ -1 + hyp, data = births)
  > coef(mh2)
    hypN   hypH
  3198.9 2768.2
  ```

- Interpretation: `-430.7 = 2768.2 - 3198.9`
  = difference between level 2 ("H") *vs.* reference level 1 ("N") of factor hyp.

Additive model with both gestwks and hyp

- Joint effect of hyp and gestwks is modelled e.g. by updating:

```
> mhg <- update(mh, . ~ . + gestwks)
              Estimate    2.5%   97.5%
(Intercept)    -4285.0 -4969.7 -3600.3
hypH            -143.7  -259.0   -28.4
gestwks          192.2   174.7   209.8
```

- The coefficient for hyp: H *vs.* N is attenuated (from -430.7 to -143.7).
- Does -143.7 estimate the **causal effect** of hyp **adjusted** for gestwks?
- No, as gestwks is most likely a **mediator**. – Much of the effect of hyp on bweight is mediated via shorter gestwks in hypertensive mothers.
- Instead, for **total causal effect** of hyp, adjustment for at least age is needed, but adjusting for gestwks is **overadjustment**.
- Yet, for **predictive modelling** it is OK to keep gestwks.

Model with interaction of `hyp` and `gestwks`

- `mhgi <- lm(bweight ~ hyp + gestwks + hyp:gestwks, ...)`

- Or with shorter formula: `bweight ~ hyp * gestwks`

```
              Estimate    2.5%   97.5%
(Intercept)    -3960.8 -4758.0 -3163.6
hypH           -1332.7 -2841.0   175.7
gestwks          183.9   163.5   204.4
hypH:gestwks      31.4    -8.3    71.1
```

- Estimated slope: 183.9 g/wk in reference group `N` of normotensive mothers and $183.9 + 31.4 =$ 215.3 g/wk in hypertensive mothers.

⇔ For each additional week the difference in mean `bweight` between `H` and `N` group increases by 31.4 g.

- *Interpretation of* `Intercept` *and "main effect"* `hypH`*?*

Model with interaction (cont'd)

More interpretable parametrization obtained if `gestwks` is **centered** at some reference value, using e.g. the **insulate** operator `I()` for explicit transformation of an original term.

-
  ```
  mi2 <- lm(bweight ~ hyp*I(gestwks-40), ...)
                          Estimate   2.5%  97.5%
  (Intercept)               3395.6 3347.5 3443.7
  hypH                       -77.3 -219.8   65.3
  I(gestwks - 40)            183.9  163.5  204.4
  hypH:I(gestwks - 40)        31.4   -8.3   71.1
  ```

- The "main effect" of `hyp` $= -77.3$ is the difference between `H` and `N` at the reference value `gestwks` $=$ 40.

- `Intercept` $=$ 3395.6 is the estimated mean `bweight` at the reference value 40 of `gestwks` in group `N`.

Factors and contrasts in R

- A categorical explanatory variable or **factor** with L **levels** will be represented by $L - 1$ linearly independent columns in the **model matrix** of a linear model.
- These columns can be defined in various ways implying alternative **parametrizations** for the effect of the factor.
- Parametrization is defined by given type of **contrasts**.
- Default: **treatment** contrasts, in which 1st class is the **reference**, and regression coefficient β_k for class k is interpreted as $\beta_k = \mu_k - \mu_1$
- Own parametrization may be tailored by function `C()`, with the pertinent **contrast matrix** as argument.
- Or, use `ci.lin(mod, ctr.mat = CM)` after fitting.

Two factors: additive effects

- Factor X has 3 levels, Z has 2 levels – Model:

$$\mu = \alpha + \beta_1 X_1 + \beta_2 X_2 + \beta_3 X_3 + \gamma_1 Z_1 + \gamma_2 Z_2$$

- X_1 (reference), X_2, X_3 are the indicators for X,
- Z_1 (reference), Z_2 are the indicators for Z.
- Omitting X_1 and Z_1 the model for mean is:

$$\mu = \alpha + \beta_2 X_2 + \beta_3 X_3 + \gamma_2 Z_2$$

with predicted means μ_{jk} $(j = 1, 2, 3; k = 1, 2)$:

		$Z = 1$	$Z = 2$
	1	$\mu_{11} = \alpha$	$\mu_{11} = \alpha + \gamma_2$
X	2	$\mu_{21} = \alpha + \beta_2$	$\mu_{22} = \alpha + \beta_2 + \gamma_2$
	3	$\mu_{31} = \alpha + \beta_3$	$\mu_{32} = \alpha + \beta_3 + \gamma_2$

Two factors with interaction

- Effect of Z differs at different levels of X:

		$Z = 1$	$Z = 2$
X	1	$\mu_{11} = \alpha$	$\mu_{12} = \alpha + \gamma_2$
	2	$\mu_{21} = \alpha + \beta_2$	$\mu_{22} = \alpha + \beta_2 + \gamma_2 + \delta_{22}$
	3	$\mu_{31} = \alpha + \beta_3$	$\mu_{32} = \alpha + \beta_3 + \gamma_2 + \delta_{32}$

- How much the effect of Z (level 2 *vs.* 1) changes when the level of X is changed from 1 to 3:

$$\begin{aligned}\delta_{32} &= (\mu_{32} - \mu_{31}) - (\mu_{12} - \mu_{11}) \\ &= (\mu_{32} - \mu_{12}) - (\mu_{31} - \mu_{11}),\end{aligned}$$

  $=$ how much the effect of X (level 3 *vs.* 1) changes when the level of Z is changed from 1 to 2.

- See the exercise: interaction of `hyp` and `gest4`.

Contrasts in R

- All contrasts can be implemented by supplying a suitable **contrast function** giving the **contrast matrix** e.g:

  ```
  > contr.cum(3)          > contr.sum(3)
  1 0 0                    1    1   0
  2 1 0                    2    0   1
  3 1 1                    3   -1  -1
  ```

- In model formula factor name `faktori` can be replaced by expression like `C(faktori, contr.cum)`.

- Function `ci.lin()` can calculate CI's for linear functions of the parameters of a fitted model `mall` when supplied by a relevant contrast matrix

  ```
  > ci.lin(mall, ctr.mat = CM)[ , c(1,5,6)]
  ```

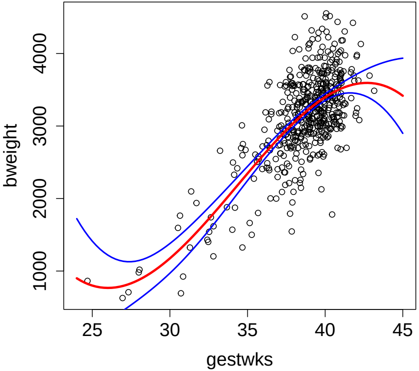
  → No need to specify contrasts in model formula!

More about numeric regressors

What if dependence of Y on X is non-linear?

- **Categorize** the values of X into a factor.
  - Continuous effects violently discretized by often arbitrary cutpoints. This is inefficient.
- Fit a low-degree (e.g. 2 to 4) **polynomial** of X.
  - Tail behaviour may be problematic.
- Use **fractional polynomials**.
  - Invariance problems. Only useful if $X = 0$ is well-defined.
- Use a **spline** model: smooth function $s(X; \beta)$. – See Martyn's lecture
  - More flexible models that act locally.
  - Effect of X reported by graphing $\widehat{s}(X; \beta)$ & its CI

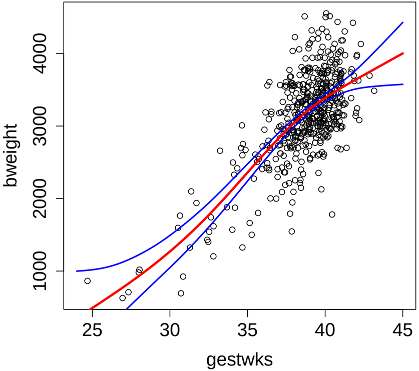
Mean `bweigth` as 3rd order polynomial of `gestwks`



```
> mp3 <- update( m, . ~ . - gestwks + poly(gestwks, 3) )
```

- The model is linear in parameters with 4 terms & 4 df.
- Otherwise good, but the tails do not behave well.

Penalized spline model with cross-validation



```
> library(mgcv)
> mpen <- gam( bweight ~ s(gestwks), data = births)
```

- Looks quite nice.
- Model df ≈ 4.2; close to 4, as in the 3rd degree polynomial model.

From linear to generalized linear models

- An alternative way of fitting our 1st Gaussian model:

```
> m <- glm(bweight ~ gestwks, family=gaussian, data=births)
```

- Function `glm()` fits **generalized linear models** (GLM).
- Requires specification of the
  - **family** – i.e. the assumed “error” distribution for Y_is,
  - **link** function – a transformation of the expected Y_i.
- Covers common models for other types of response variables and distributions, too, e.g. **logistic** regression for **binary** responses and **Poisson** regression for counts.
- Fitting: method of **maximum likelihood**.
- Many extractor functions for a `glm` object similar to those for an `lm` object.

Generalized linear models

Modelling how expected values, risks, rates, etc. depend on explanatory variables or regressors $X = (X_1, \ldots, X_p)$. – Common elements:

- Each subject i $(i = 1, \ldots, N)$ has an own **regressor profile**, i.e. vector $x_i^{\mathsf{T}} = (x_{i1}, \ldots, x_{ip})$ of values of X.
- Let vector $\beta^{\mathsf{T}} = (\beta_0, \beta_1, \ldots, \beta_p)$ contain regression coefficients. The **linear predictor** is a linear combination of β_js and x_{ij}s:

$$\eta_i = \beta_0 + \beta_1 x_{i1} + \cdots + \beta_p x_{ip}$$

- Some X_js can be **product terms** for interactions and modifications if needed, and **splines** may be used for continuous covariates.
- Further model specification depends on the type of outcome variable, assumed error distribution or family, desired interpretation of coefficients, and importance and choice of time scale(s).

Binary regression and interpretations of coefficients

- Basic model for risks $\pi(x_i) = P\{Y_i = 1|X = x_i\} = E(Y_i|X = x_i)$ with fixed risk period, complete follow-up (no censoring, nor competing events):

$$g\{\pi(x_i)\} = \beta_0 + \beta_1 x_{i1} + \ldots \beta_p x_{ip}, \quad i = 1, \ldots, N.$$

- **Link** $g(\cdot)$ and interpretation of β_js, assuming the validity of model (including homogeneity or non-modification of the coefficent in question):
  - id $\Rightarrow \beta_j =$ adjusted **risk difference** (RD) for $X_j = 1$ vs. $X_j = 0$,
  - log $\Rightarrow \beta_j =$ adjusted log of **risk ratio** (RR) – " –
  - logit $\Rightarrow \beta_j =$ adjusted log of **odds ratio** (OR), – " –

- Fitting: `glm(..., family=binomial(link=...), ...)`

- Issues with id & log links in keeping predicted $\widehat{\pi}(\cdot)$ between 0 and 1.
  – A solution for RR: Doubling the cases & logit-link! (Ning et al. 2022).
  – A solution for RD exists, too (Battey et al. 2019).

Poisson regression – model for rates

- A common outcome variable is a pair $(D, Y) =$ (no. of cases, person-time), from which the **incidence rate** $= D/Y$ (see Janne's lecture on Friday).

- **Poisson regression model** specifies, how theoretical rates or **hazards** $\lambda(x_i)$ are assumed to depend on values of X.

- Some components of X represent the relevant **time scales** (as in the exercise of today; more details in Bendix's lecture on Monday).

- Linear predictor as above – **Link** $g(\cdot)$ and interpretation of β_js:
  - id $\Rightarrow \beta_j =$ adjusted **rate difference** (RD) for $X_j = 1$ vs. $X_j = 0$,
  - log $\Rightarrow \beta_j =$ adjusted log of **rate ratio** (RR) – " –

- Fitting – our recommended approach using `Epi`:

```
glm(cbind(d,y) ~ ..., family=poisreg(link=...),...)
```

What was covered

- A wide range of models from simple linear regression to splines.
- Gaussian family for continuous outcomes, binomial for binary outcomes, and Poisson family for rates.
- Various link functions for different parametrizations.
- R functions fitting linear and generalized models: `lm()` and `glm()`.
- Parametrization of categorical explanatory factors; contrast matrices.
- Extracting results and predictions: `ci.lin()`, `fitted()`, `predict()`.
- Model diagnostics: `resid()`, `plot.lm()`, `...` .