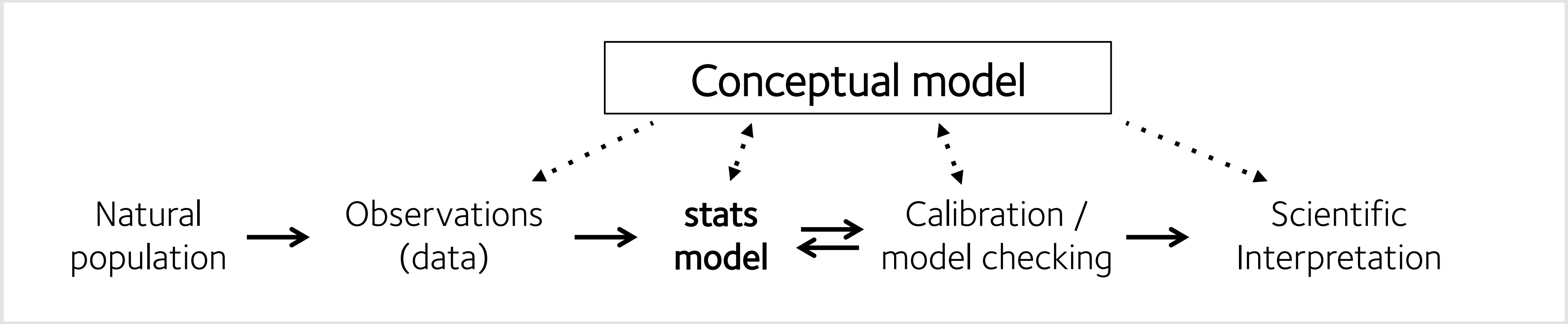
Introduction to statistical modelling

Gavin Band
gavin.band@well.ox.ac.uk

UNIVERSITY OF OXFORD

Last week

- Implemented your first (log)likelihood function in R for a 2x2 table reflecting binomial sampling in rows.
- Drew distinction between a statistical ("small world"*) model, which gives us formal statistics like estimates and P-value and Bayes factors, and our overall conceptual ("large world") model, which is how we interpret results.
- Two examples using genome-wide data to check calibration of the statistical model.



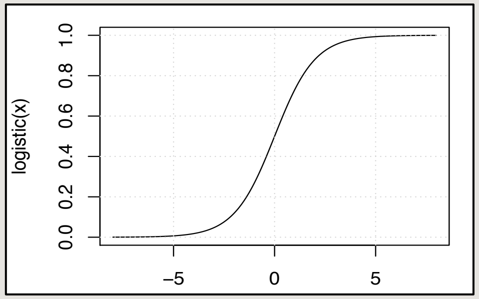
*"Statistical Rethinking", https://xcelab.net/rm/statistical-rethinking/

Last week

We also covered some technical stuff:

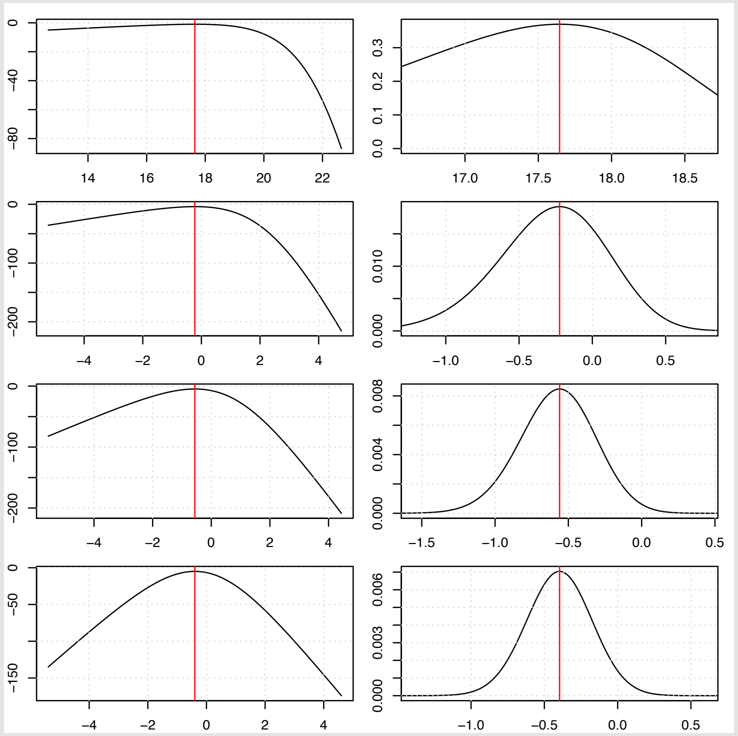
$$\text{logistic}(x) = \frac{e^x}{1+e^x}$$

- the `logistic()` function and parameterisation of 2x2 tables in terms of an *effect size parameter* (log odds ratio) – often what we're most interested in.



- we plotted the likelihood function as "data" quantities grow:

- log-likelihoods become **more quadratic** (likelihoods become closer to Gaussian) as data quantities grow.



Distributions

Binomial distribution (e.g. socks in drawer, alleles in a population, …)

$$Y \sim \text{binomial(n,p)}$$

$$P(Y=y|n,p) = \binom{n}{y} p^y (1-p)^{n-y}$$

(where n is total number of things drawn and they are 'successful' with probability p)

Normal or Gaussian distribution (e.g. sums of errors and lots more….)

$$X \sim N(\mu, \sigma^2)$$

$$P(X=x|\mu,\sigma^2) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{(x-\mu)^2}{2\sigma^2}}$$

(where μ is the mean and σ^2 is the variance).

Multivariate Normal Distribution (e.g. sums of errors and lots more….)

$$X \sim \text{MVN}(\mu, \Sigma)$$

$$P(X=x|\mu,\Sigma) = \frac{1}{\sqrt{(2\pi)^k |\Sigma|}} \cdot e^{-\frac{1}{2}(x-\mu)^t \Sigma (x-\mu)}$$

(where μ is the k-dimensional mean vector, and Σ is the $k \times k$ covariance matrix).

This week

- Implement logistic regression
- More on asymptotics
- Meta-analysis
- Leading toward bayesian analysis

Logistic regression

Last week we considered a 2x2 table parameterised like this:

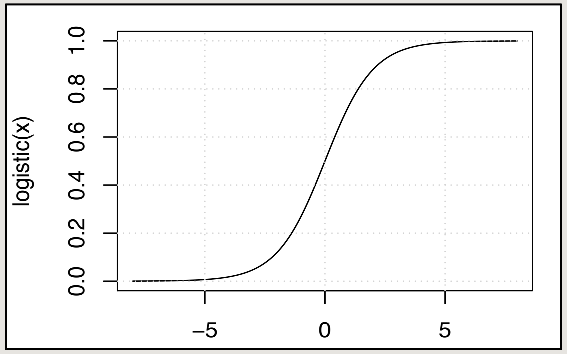
	A	B
controls	$1-\theta_1$	θ_1
cases	$1-\theta_2$	θ_2

$$\theta_1 = \text{logistic}(\mu)$$
$$\theta_2 = \text{logistic}(\mu + \beta)$$

μ is the log-odds of outcome B in controls
β is the log odds ratio (the 'effect size' of interest).

As several of you pointed out this model is too simplistic – how do we account for measured covariates or other confounders?

$$\text{logistic}(x) = \frac{e^x}{1 + e^x}$$



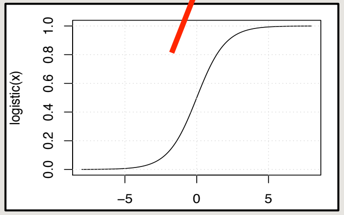
	O blood group	**ethnic group**	**sex**	**principal component 1**	**...**
sample 1	0	FULA	M	0.1	...
sample 2	0	FULA	F	0.15	...
sample 3	1	JOLA	F	-0.02	...
...					
	β_1	β_2	β_3	β_4	...

Logistic regression gives each predictor and covariate its own parameter (β_j), allowing it to contribute to modelling the outcome.

Logistic regression

Specifically the outcome Y_i for each individual is modelled in terms of a linear combination of the predictors and covariates:

$$Y_i \sim \text{binomial}(1, \theta_i)$$
$$\theta_i = \text{logistic}(x_i \cdot \beta)$$



"design matrix"

parameters

$$X = \begin{pmatrix} 1 & x_{11} & x_{12} & \dots & x_{1k} \\ 1 & x_{21} & x_{22} & \dots & x_{2k} \\ \vdots & & & \ddots & \\ 1 & x_{n1} & x_{n2} & \dots & x_{nk} \end{pmatrix} \quad \text{and} \quad \beta = \begin{pmatrix} \beta_0 \\ \beta_1 \\ \vdots \\ \beta_k \end{pmatrix}$$

$x_{i,}$ = the row containing the predictors and covariates for sample i

$\boldsymbol{\beta}_j$ is interpreted as: *the change in log-odds that Y_i=1 for a unit increase in the jth predictor.*

or if you don't like matrix notation:

$$\theta_i = \text{logistic}(\beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2} + \beta_3 x_{i3} + \cdots)$$

where x_{ij} is the value of the *j*-th predictor for sample *i*.

Logistic regression as a special case

Logistic regression and linear regression are both special cases of *generalised linear models*, for which:

1. The expected (i.e. mean) value of Y_i is a function of the linear predictor:

$$E(Y_i) = f(x_{i\cdot}\beta)$$

f is called the "**mean function**".
E.g. f = identity for for linear regression, while f = logistic for logistic regression.

2. The distribution of Y_i around its mean is chosen to reflect the problem at hand. E.g.:

$$Y_i \sim N(x_{i\cdot}\beta, \sigma^2)$$

Linear regression; models the mean of a continuous quantity – e.g. gene expression levels.

$$Y_i \sim \text{binomial}(\text{logistic}(x_{i\cdot}\beta), 1)$$

Logistic regression; models the frequency of binary (or more generally categorical) outcomes - e.g. case/control status.

$$Y_i \sim \text{Poisson}(e^{x_{i\cdot}\beta})$$

Poisson regression; models the rate of occurrence of discrete events - e.g. sequence reads along a genome.

Challenge #1: implement logistic regression

$$X = \begin{pmatrix} 1 & x_{11} & x_{12} & \dots & x_{1k} \\ 1 & x_{21} & x_{22} & \dots & x_{2k} \\ \vdots & & & \ddots & \\ 1 & x_{n1} & x_{n2} & \dots & x_{nk} \end{pmatrix} \quad \text{and} \quad \beta = \begin{pmatrix} \beta_0 \\ \beta_1 \\ \vdots \\ \beta_k \end{pmatrix}$$

design matrix

parameters

$$Y_i \sim \text{binomial}(1, \theta_i)$$
$$\theta_i = \text{logistic}(x_i \cdot \beta)$$

```
logistic.regression.ll <- function(
    Y,
    design.matrix,
    params
) {
  ...
}
```

Hint #1: you have written `binomial.ll(y,n,p)` and **`logistic()`** already. You can call them for each sample.

Hint #2: you can compute the vector of linear predictors as:

```
linear.predictor = design.matrix %*% params
```

This is matrix multiplication of the design matrix times the column of parameters.

Challenge #1: implement logistic regression

$$X = \begin{pmatrix} 1 & x_{11} & x_{12} & \dots & x_{1k} \\ 1 & x_{21} & x_{22} & \dots & x_{2k} \\ \vdots & & & \ddots & \\ 1 & x_{n1} & x_{n2} & \dots & x_{nk} \end{pmatrix} \quad \text{and} \quad \beta = \begin{pmatrix} \beta_0 \\ \beta_1 \\ \vdots \\ \beta_k \end{pmatrix}$$

design matrix

parameters

$$Y_i \sim \text{binomial}(1, \theta_i)$$
$$\theta_i = \text{logistic}(x_i \cdot \beta)$$

```
logistic.regression.ll <- function (
    Y,
    design.matrix,
    params
) {
    predictor = design.matrix %*% params
    lls = binomial.ll( Y, 1, logistic( predictors ))
    return( sum( lls ))
}
```

Plotting

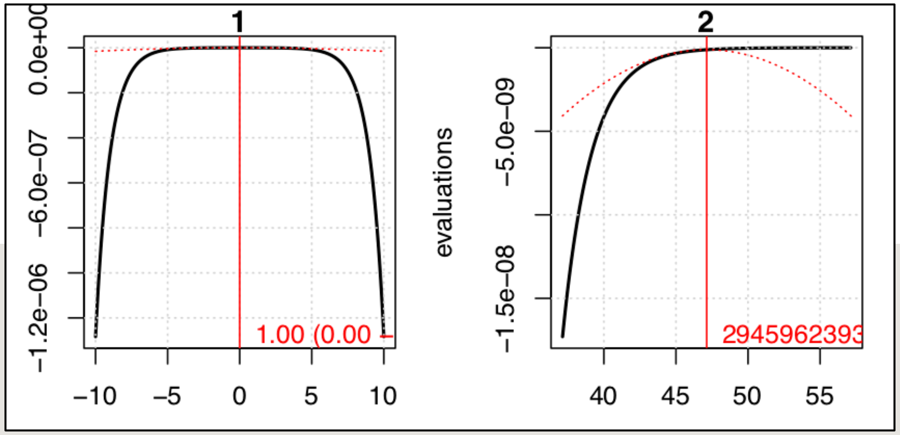
Let’s look at a simple example with one case and one control:

```
Y = c( 0, 1 )
design.matrix = matrix( c( 1, 0, 1, 1 ), nrow = 2, byrow = T )

# Plot it:
at = seq( from = -10, to = 10, by = 0.01 )
evaluations = sapply( at, function( x ) {
    logistic.regression.ll( Y, design.matrix, c( 0, x ))
} )
plot( x, evaluations, type = "l" )
grid()
```

I have made a slightly more advanced plotting function in solutions part 2.R:

```
plot.loglikelihood(
    Y,
    design.matrix,
    logistic.regression.ll
)
```



Let’s try it out on the O blood group example

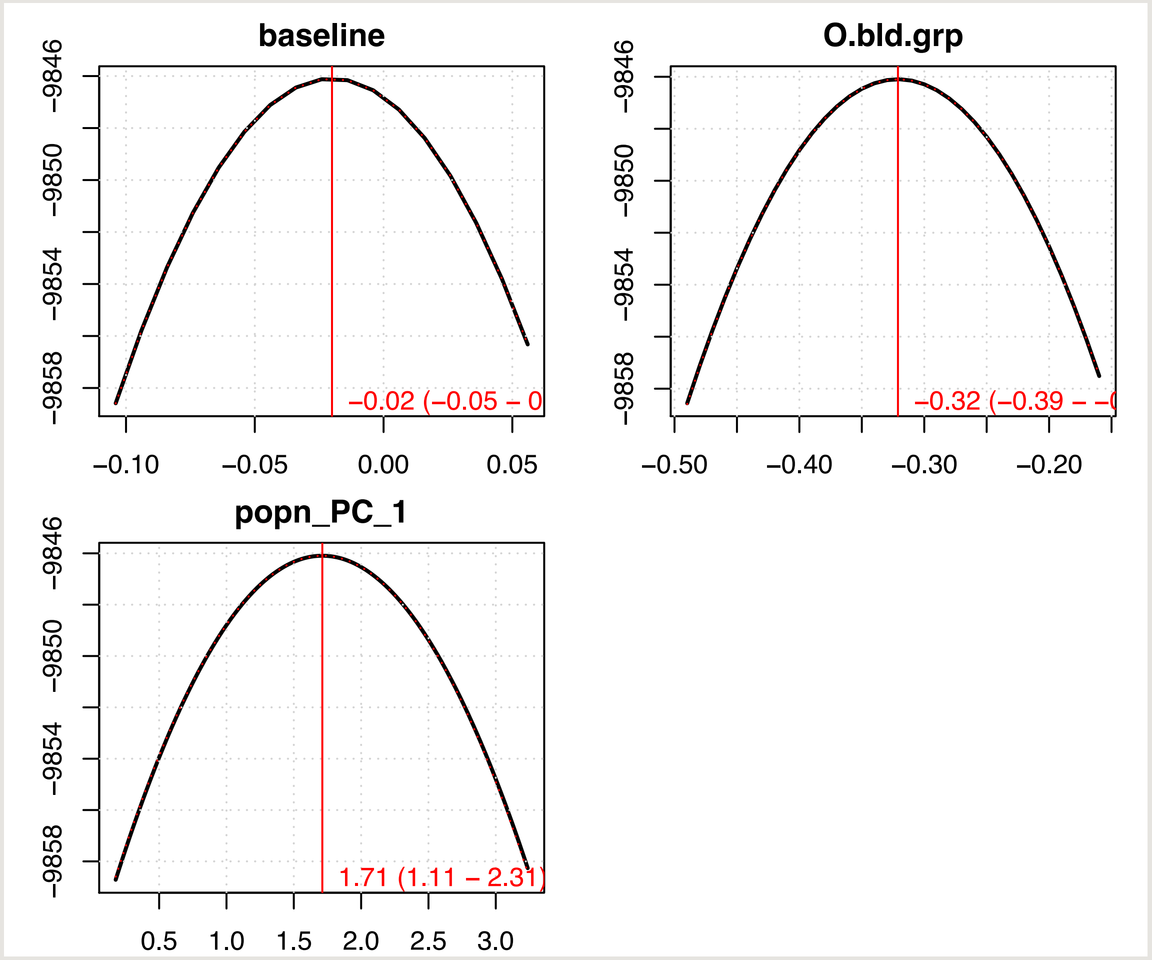
```
X = read.csv(
    "practicals/o_bld_grp.csv",
    header = T
)

head(X) # take a look at it
```

To use our plot function we need the design matrix:

```
design.matrix = as.matrix(
    cbind(
        baseline = rep( 1, nrow(X) ),  # repeat 1 N times
        X[, c("o.bld.grp", "popn_PC_1" )]
    )
)

plot.loglikelihood(
    X$severe.malaria,
    design.matrix,
    logistic.regression.ll
)
```


baseline
−0.02 (−0.05 – 0
O.bld.grp
−0.32 (−0.39 – −(
popn_PC_1
1.71 (1.11 – 2.31)
−9846
−9850
−9854
−9858
−0.10
−0.05
0.00
0.05
−0.50
−0.40
−0.30
−0.20
0.5
1.0
1.5
2.0
2.5
3.0

Summary #1

- You have now implemented a useful logistic regression log-likelihood in plain R.
- This can be used for plotting, as a starting point for generalisations, or as a building block for other more complex models.
- Of course R provides its own functions for fitting logistic regression models. The core one is `glm()`. We will now use these along with our log-likelihood function to investigate some real datasets. General form:

```
glm(
   outcome ~ predictor1 + predictor2 + ..., # formula
   family = “binomial”     # for logistic regression
   data = X                # data frame to take values from
)
```

Challenge #2: Investigate the O blood group data

```
head(X)

fit = glm(
   severe.malaria ~ O.bld.grp + popn_PC_1,
   family = "binomial",
   data = X
)

summary(fit)$coeff
```

What variables are in the data?

Can you interpret the regression output?

Challenge #2: Investigate the O blood group data

```
head(X)

fit = glm(
    severe.malaria ~ o.bld.grp + country,
    family = "binomial",
    data = X
)

summary(fit)$coeff
```

Should get something like this:

```
                    Estimate Std. Error         z value     Pr(>|z|)
(Intercept)      0.3397772948 0.05708759  5.951859102 2.651136e-09
o.bld.grp       -0.3202151077 0.03318353 -9.649820248 4.924197e-22
countryCameroon -0.3367451583 0.07914892 -4.254577040 2.094445e-05
countryGambia   -0.2475559812 0.06219679 -3.980204798 6.885592e-05
```

Maximum likelihood estimate

Standard error of mle – c.f. quadratic approximation to log-likelihood

"Wald test" P-value, computed from standard error

Challenge #3: investigate covariates

Do covariates explain the association?

```
fit = glm(
    severe.malaria ~ O.bld.grp + [other variables...],
    family = "binomial",
    data = X
)
```

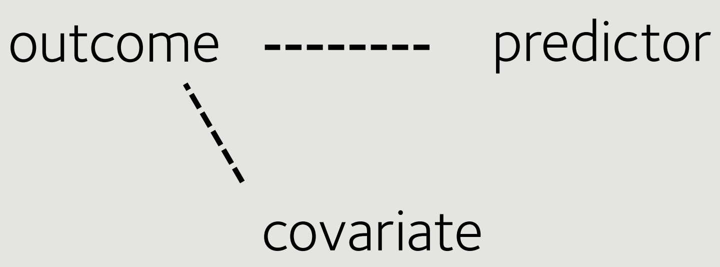
- Can you interpret the output? Which variables are important? Are there confounders? Can you destroy the O blood group signal?

Challenge #2: investigate covariates

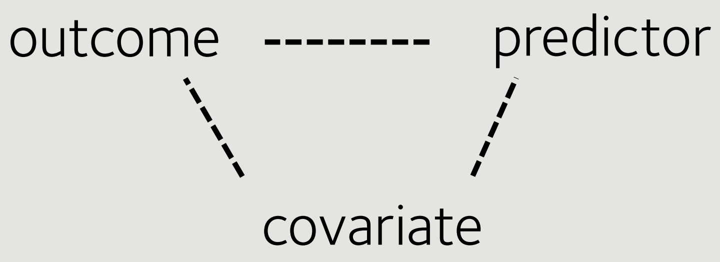
```
                                            Estimate    Std. Error   z value     Pr(>|z|)
(Intercept)                               0.34842369   0.05992995  5.81384898 6.105250e-09
O.bld.grp                                -0.31801617   0.03378603 -9.41265363 4.837662e-21
sexM                                     -0.01771729   0.03361826 -0.52701395 5.981839e-01
countryCameroon                           0.06286589   0.35629700  0.17644238 8.599464e-01
countryGambia                            -0.39190253   0.29389695 -1.33346920 1.823779e-01
countryGhana                              0.14052738   0.17105356  0.82154026 4.113386e-01
countryKenya                             -0.05845274   0.31788063 -0.18388267 8.541055e-01
countryMalawi                            -0.29907305   0.06854128 -4.36340044 1.280562e-05
countryTanzania                          -0.70762110   0.33138860 -2.13532120 3.273478e-02
ethnicityBANTU                           -0.63555997   0.36349040 -1.74849178 8.037891e-02
ethnicityCHONYI                          -0.66250151   0.31985343 -2.07126591 3.833395e-02
ethnicityFRAFRA_NANKANA_GRUSHIE_KUSASI[UER] 13.16537398 153.86479143  0.08556457 9.318126e-01
ethnicityFULA                            -0.43721358   0.29752670 -1.46949360 1.416990e-01
ethnicityGIRIAMA                          0.18014826   0.31710040  0.56811112 5.699595e-01
ethnicityJOLA                             0.40485488   0.29877379  1.35505485 1.754001e-01
ethnicityKASEM                           -0.38898939   0.20356483 -1.91088702 5.601910e-02
ethnicityKAUMA                           -0.43941549   0.33390479 -1.31599038 1.881773e-01
ethnicityMANDINKA                         0.22526535   0.29309377  0.76857777 4.421440e-01
ethnicityMZIGUA                           0.58827051   0.35624494  1.65130911 9.867547e-02
ethnicityMZIGUA_MIXED                     0.70428511   0.43176348  1.63118268 1.028518e-01
ethnicityNANKAM                          -0.41023415   0.21980755 -1.86633327 6.199475e-02
ethnicityNORTHERNER                      13.17322089 160.70677606  0.08197054 9.346701e-01
ethnicityOTHER                            0.48722082   0.28269623  1.72347829 8.480207e-02
ethnicitySEMI_BANTU                      -0.46971604   0.36181028 -1.29823849 1.942054e-01
ethnicityWABONDEI                         0.46229005   0.38797213  1.19155477 2.334359e-01
ethnicityWABONDEI_MIXED                   0.69429031   0.41617170  1.66827853 9.526045e-02
ethnicityWASAMBAA                         0.44671325   0.36119028  1.23678094 2.161684e-01
ethnicityWASAMBAA_MIXED                   0.61407421   0.39491416  1.55495617 1.199565e-01
ethnicityWOLLOF                          -0.10248243   0.29847414 -0.34335446 7.313318e-01
```

Covariate types

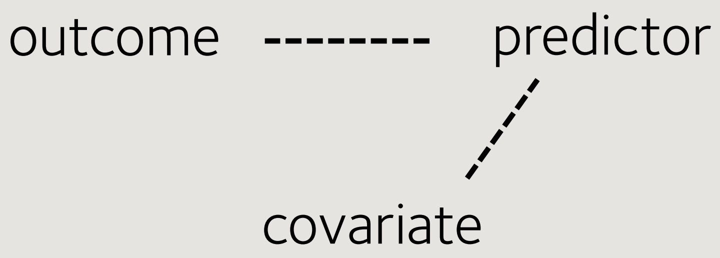
- Three types of covariates you might encounter:
  (lines indicate correlations)



non-confounder – may help to include by explaining some of the variation in the outcome.



confounder – mandatory to include!



irrelevant variable– unlikely to be helpful (and may be harmful)

Challenge #4: investigate covariates in second example

```
X = read.csv(
   "practicals/practicals/logistic_regression_example.csv",
   header = T
)
head(X)

fit = glm(
   outcome ~ predictor + [...],
   family = "binomial",
   data = X
)

summary(fit)$coeff
```

Fit covariates (I suggest starting one at a time). What is your conclusion?

Rules of thumb

	Changes estimated predictor effect?	**Changed estimated predictor standard error?**	**Include?**
Confounding variable e.g. covariate3	Yes	Yes	Yes
Non-confounding explanatory variable E.g. covariate2	No	No	Depends*
Irrelevant variable E.g. covariate1 and 4	No	Yes if colinear with predictor	No

*For linear regression in unselected samples: **usually good to include**. Covariate may explain some of the noise in the outcome and lead to more precise estimates.

*For logistic regression in a selected (e.g. case-control) sample: **depends on effect sizes and frequency**. Case-control sam"pling induces correlation between variables that are not correlated in the population. See Pirinen et al Nature Genetics 2012 https://doi.org/10.1038/ng.2346.

Summary #1

- The likelihood function captures all the information about the parameters in a dataset

$$\text{likelihood} = P(data|parameters)$$

- The log-likelihood becomes approximately quadratic (the likelihood becomes approximately normal) as data volumes grow.

- This makes it practically useful to summarise the log-likelihood by its mode (maximum likelihood estimate) and its covariance (i.e. standard error).

- But even if this fails - you can always use a computer to plot and employ the likelihood directly. (E.g. simulate from a logistic regression model)

Asymptotics

- See board.

$$\log P(\text{data}|\text{parameters}{=}x) \approx \text{const} - \left(x - \hat{\beta}\right)^t I^{-1}(x - \hat{\beta})$$

$$\hat{\beta} \sim N(\beta, I)$$

$$I \approx I^{\infty}/n$$

loglikelihood is quadratic with curvature I^{-1}

mle is normally distributed with covariance I

I itself is approx equal to constant shrinking at rate n (standard errors shrink at rate $\sqrt{n}$)

Conclusion: practically we often only need to know the maximum likelihood estimate (beta hat) and its second derivative $H = -I^{-1}$.

2nd derivatives are computable using maths or numerically (c.f. `logistic.regression.ddll()` in `solutions part 2.R`)

Full statements are somewhat technical because approximations are distributional, c.f "Local asymptotic normality" and

How to think of estimates

- Maximum likelihood estimate = True effect + noise*

$$\hat{\beta} = \beta + \epsilon$$

- When the asymptotics hold,

$$\epsilon \sim N(0, \mathrm{se}^2)$$

- And se ~ const / √n

*here "noise" means the formal statistical sampling noise, or uncertainty in the likelihood.
Real experiments have other, possibly unmodelled sources of noise!

Meta-analysis

Suppose we are given summary statistics from K independent studies

$$\hat{\beta}_1, \hat{\beta}_2, \cdots, \hat{\beta}_K$$

with standard errors

$$\mathrm{se}_1, \mathrm{se}_2, \cdots, \mathrm{se}_K$$

$$L_i \approx N(\hat{\beta}_i, \mathrm{se}_i^2)$$

(approx) likelihood in study i

$$L(\beta) = \prod_i L_i(\beta_i)$$

overall likelihood for a vector β of “true” effects

$$\mathrm{posterior}(\beta) \propto L(\beta) \times \mathrm{prior}(\beta)$$

What we'd like to know – what are the true effects?

prior encodes our model of true effects

Example: fixed-effect meta-analysis

- Suppose we believe the true effect is identical in all studies – sometimes termed a ‘fixed effect’

- Then $\boldsymbol{\beta} = (\beta, \beta, \ldots, \beta)$, and we are just taking a product of gaussian distributions evaluated at the same point β.

- Apply a crucial lemma…

Example: “random effect meta-analysis”

- Assume the true effects are distributed around some common mean

$$\boldsymbol{\beta} = (\beta_1, \beta_1, \ldots, \beta_k)$$

where

$$\beta_i \sim N(\ \alpha, \delta^2\ )$$

- Then estimate **α and δ^2**

E.g. cf **meta** or **metafor** packages in R.

Crucial property of gaussians

Products of gaussian distributions are gaussian:

Lemma 1. *If μ_1,μ_2 are two means and Σ_1, Σ_2 are covariance matrices then:*

$$MVN(x;\mu_1,\Sigma_1) \times MVN(x;\mu_2,\Sigma_2) = const \times MVN(x;a,A)$$

where

$$A = \left(\Sigma_1^{-1} + \Sigma_2^{-1}\right)^{-1} \qquad and \qquad a = A\left(\Sigma_1^{-1}\mu_1 + \Sigma_2^{-1}\mu_2\right)$$

1 dimensional version:

Lemma 2. *If μ_1,μ_2 are two means and σ_1^2, σ_2^2 are variances then:*

$$N(x;\mu_1,\sigma_1^2) \times N(\mu_1;b,\sigma_2^2) = const \times N(x;w,W)$$

where

$$W = \frac{1}{\frac{1}{\sigma_1^2} + \frac{1}{\sigma_2^2}} \qquad and \qquad w = W \times \left(\frac{\mu_1}{\sigma_1^2} + \frac{\mu_1}{\sigma_2^2}\right)$$

Conclusion: can find the meta-analysis estimate as a weighted average of per-study estimates.

Fixed-effect meta-analysis

We are given summary statistics from K independent studies $\hat{\beta}_1, \hat{\beta}_2, \cdots, \hat{\beta}_K$

with standard errors $\mathrm{se}_1, \mathrm{se}_2, \cdots, \mathrm{se}_K$

$$W = \frac{1}{\frac{1}{se_1^2} + \frac{1}{se_2^2} + \cdots} \qquad and \qquad w = W \times \left(\frac{\hat{\beta}_1}{se_1^2} + \frac{\hat{\beta}_2}{se_2^2} + \cdots \right)$$

Meta-analysis variance

Meta-analysis estimate

Challenge #4: implement fixed-effect meta-analysis

$$\underbrace{W}_{\text{Meta-analysis variance}} = \frac{1}{\frac{1}{se_1^2} + \frac{1}{se_2^2} + \cdots} \qquad \textit{and} \qquad \underbrace{w}_{\text{Meta-analysis estimate}} = W \times \left(\frac{\hat{\beta}_1}{se_1^2} + \frac{\hat{\beta}_2}{se_2^2} + \cdots \right)$$

```
fixed.effect.meta <- function(
    betas,
    ses
) {
    W = ...
    scaled_betas = ... # suggestion
    return( list(
        meta.beta = ...,
        meta.se =
    ))
}
```

Challenge #4: implement fixed-effect meta-analysis

$$W = \frac{1}{\frac{1}{se_1^2} + \frac{1}{se_2^2} + \cdots} \qquad \textit{and} \qquad w = W \times \left(\frac{\hat{\beta}_1}{se_1^2} + \frac{\hat{\beta}_2}{se_2^2} + \cdots \right)$$

Meta-analysis variance

Meta-analysis estimate

```
fixed.effect.meta <- function(
    betas,
    ses
) {
    W = 1 / sum( 1/ses^2 )
    scaled_betas = betas / ses^2
    return( list(
        meta.beta = W * sum( scaled_betas ),
        meta.se = sqrt(W)
    ))
}
```

Interpreting fixed-effect meta-analysis

$$W = \frac{1}{\frac{1}{se_1^2} + \frac{1}{se_2^2} + \cdots} \qquad and \qquad w = W \times \left(\frac{\hat{\beta}_1}{se_1^2} + \frac{\hat{\beta}_2}{se_2^2} + \cdots \right)$$

Meta-analysis estimate is **at least as precise** as the most precise study estimate. (Studies with the smallest variance dominate the expression).

Meta-analysis estimate is a **weighted average** of per-study effect estimates. The **most precise** estimates get the **largest weights**.

Our derivation makes it clear this gives a Gaussian approximation to the full data loglikelihood, obtained by assuming a Gaussian approximation in each study.

Interpreting fixed-effect meta-analysis

```
X = read.csv( "practicals/o_bld_grp.csv", header = T )
fit = glm(
    severe.malaria ~ o.bld.grp + country,
    family = "binomial",
    data = X
)
summary( fit )$coeff
```

```
meta.data = read.csv(
    "practicals/o_bld_grp_per_country.csv",
    header = T, as.is = T
)
head( meta.data )

fixed.effect.meta( meta.data$beta, meta.data$se )
```

Compare effect size and standard error.

Conclusions

- You can implement (log)-likelihoods

- You can fit and interpret regression models

- In many cases all we care about are an estimate and its standard error. These are interpreted **either** as giving an approximation to the likelihood, **or** as a sampling distribution of the estimate.

- If asymptotics don't hold – use computers to plot and simulate.

- A reminder that the above is all part of the formal statistical ('small world') model. Model checking and calibration against understanding is always needed c.f. examples last week

Tomorrow

So far we have concentrated on the likelihood function.

But right back at the start we noted that we are really interested in the *posterior* distribution (distribution of the parameters of interest)

$$P(\text{mass}|\text{data}) \propto P(\text{data}|\text{mass}) \times P(\text{mass})$$

Posterior Likelihood Prior

c.f. the Bayesian analysis session this morning.

A prior is **necessary** when there isn't much information about particular parameters of interest. And it is **useful** when expressing more complex models than the likelihood encodes. And it is **natural**, since it is essentially the right* way to make estimates.

*For some definitions of 'right'

Tomorrow

So far we have concentrated on the likelihood function.

But right back at the start we noted that we are really interested in the *posterior* distribution (distribution of the parameters of interest)

$$\underbrace{P(\text{mass}|\text{data})}_{\text{Posterior}} \propto \underbrace{P(\text{data}|\text{mass})}_{\text{Likelihood}} \times \underbrace{P(\text{mass})}_{\text{Prior}}$$

c.f. the Bayesian analysis session this morning.

Tomorrow we'll aim to tie this up with practicals on Bayesian analysis & bayesian meta-analysis.

Also: we'll give you homework for the GWAS session.