Outline

1. Logistic regression: fitting the model
   - Components of generalized linear models
   - Logistic regression
   - Case study: runoff data
   - Case study: baby food
2. Logistic regression: Inference
   - Model fit and model diagnostics
   - Comparing models
   - Sparse data and the separation problem

Modeling non-normal data

- In all of the linear models we have seen so far, the response variable has been modeled with a normal distribution

$$(\text{response}) = (\text{fixed parameters}) + (\text{normal error})$$

- For many data sets, this model is inadequate.

  Ex: if the response variable is categorical with two possible responses, it makes no sense to model the outcome as normal.

  Ex: if the response is always a small positive integer, its distribution is also not well described by a normal distribution.

- Generalized linear models (GLMs) are an extension of linear models to model non-normal response variables. Logistic regression is for binary response variables.

The link function

Standard linear model:

$$y_i = \beta_1 x_{i1} + \beta_2 x_{i2} + \cdots + \beta_k x_{ik} + e_i, \qquad e_i \sim \mathcal{N}(0, \sigma^2)$$

The mean of expected value of the response is:

$$\mathbb{E}(y_i) = \beta_1 x_{i1} + \beta_2 x_{i2} + \cdots + \beta_k x_{ik}$$

- We will use the notation $\eta_i = \beta_1 x_{i1} + \cdots + \beta_k x_{ik}$ to represent the linear combination of explanatory variables. In a standard linear model,

$$\mathbb{E}(y_i) = \eta_i$$

- In a GLM, there is a link function g between η and the mean of the response variable:

$$g(\mathbb{E}(y_i)) = \eta_i$$

- For standard linear models, the link function is the identity function $g(y_i) = y_i$.

The link function

- It can be easier to consider the inverse of the link function:

$$\mathbb{E}(y_i) = g^{-1}(\eta_i)$$

- When the response variable is binary (with values coded as 0 or 1), the mean is simply $\mathbb{E}y = \mathbb{P}\{y = 1\}$.
- A useful function for this case is

$$\mathbb{E}y = \mathbb{P}\{y = 1\} = \frac{e^{\eta}}{1 + e^{\eta}} = g^{-1}(\eta)$$

  η can take any value, the mean is always between 0 and 1.
- The corresponding link function is called the logit function,

$$g(p) = \log\left(\frac{p}{1-p}\right) = \log\left(\frac{\mathbb{P}\{Y = 1\}}{\mathbb{P}\{Y = 0\}}\right)$$

  It is the log of the odds. Regression under this model is called logistic regression.

Deviance

- In standard linear models, we estimate the parameters by **minimizing the sum of the squared residuals**. Equivalent to finding parameters that maximize the likelihood.
- In a GLM we also fit parameters by maximizing the likelihood. The deviance is *negative two times the maximum log likelihood* up to an additive constant.

  Estimation is equivalent to finding parameter values that minimize the deviance.

Logistic regression

- Logistic regression is a natural choice when the response is categorical with two possible outcomes.
- Pick one outcome to be a "success", or "yes", where $y = 1$.
- We desire a model to estimate the probability of "success" as a function of the explanatory variables. Using the inverse logit function, the probability of success has the form

$$\mathbb{P}\{y = 1\} = \frac{e^{\eta}}{1 + e^{\eta}} = \frac{1}{1 + e^{-\eta}}$$

  Equivalent formulas:

$$e^{\eta} = \frac{\mathbb{P}\{y = 1\}}{\mathbb{P}\{y = 0\}} \qquad \eta = \log\left(\frac{\mathbb{P}\{Y = 1\}}{\mathbb{P}\{Y = 0\}}\right)$$

- We estimate the parameters so that this probability is high for cases where $y = 1$ and low for cases where $y = 0$.

Anesthesia example

- In surgery, it is desirable to give enough anesthetic so that patients do not move when an incision is made. It is also desirable not to use much more anesthetic than necessary.
- In an experiment, patients are given different concentrations of anesthetic.
- Response: whether or not they move at the time of incision 15 minutes after receiving the drug.

Anesthesia data

	Concentration					
	0.8	1.0	1.2	1.4	1.6	2.5
Move	6	4	2	2	0	0
No move	1	1	4	4	4	2
Total	7	5	6	6	4	2
Proportion	0.17	0.20	0.67	0.67	1.00	1.00

Analyze in R with `glm` twice,

- once using raw data (0's and 1's) and
- once using summarized counts $(1/7, 1/4, \ldots, 4/4, 2/2)$.

Extends **chi-square** tests.

Binomial distribution

- Logistic regression is related to the binomial distribution.
  If there are several observations with the same explanatory variable values, then the individual responses can be added up and the sum has a binomial distribution.
- Recall: the binomial distribution has parameters n and p, mean $\mu = np$ and variance $\sigma^2 = np(1-p)$.

  The probability distribution is

$$\mathbb{P}\{X = x\} = \binom{n}{x} p^x (1-p)^{n-x}$$

- Logistic regression is in the "binomial family" of GLMs.

Logistic regression in R on raw data

```
> dat = read.table("anesthetic.txt", header = T)
> str(dat)
'data.frame': 30 obs. of  3 variables:
 $ movement: Factor w/ 2 levels "move","noMove": 2 1 2 1 1 ...
 $ conc    : num  1 1.2 1.4 1.4 1.2 2.5 1.6 0.8 1.6 1.4 ...
 $ nomove  : int  1 0 1 0 0 1 1 0 1 0 ...
> dat$movement
 [1] noMove move   noMove move   move  ...
[21] ...   noMove move   noMove move   noMove
Levels: move noMove

> fit.raw = glm(movement ~ conc, data=dat, family=binomial)
> summary(fit.raw)
glm(formula = nomove ~ conc, family = binomial, data = dat)
            Estimate Std. Error z value Pr(>|z|)
(Intercept)   -6.469      2.418  -2.675  0.00748 **
conc           5.567      2.044   2.724  0.00645 **
...
    Null deviance: 41.455  on 29  degrees of freedom
Residual deviance: 27.754  on 28  degrees of freedom
AIC: 31.754
```

Fitted Model

$$\begin{aligned} \mathbb{P}\{\text{No move}\} &= \frac{e^{\eta}}{1+e^{\eta}} = \frac{1}{1+e^{-\eta}} \\ \text{with } \eta &= -6.469 + 5.567 \times \text{concentration} \end{aligned}$$

We can get predictions

- at the 'link' level: η_i
- and at the 'response' level: y, or $\mathbb{E}Y = \mathbb{P}\{Y = 1\}$

```
> predict(fit.raw, type="link")
    1     2     3     4     5     6 ...    28    29    30
-0.90  0.21  1.32  1.32  0.21  7.448 ...  0.21 -0.90  0.21

> predict(fit.raw, type="response")
   1     2     3     4     5     6 ...    28    29    30
0.29  0.55  0.79  0.79  0.55  0.999 ...  0.55  0.29  0.55
```

Plot of the logit curve

```
layout(matrix(1:2,2,1))
my.etas = seq(-8,8, by=.01)
my.prob = 1/(1+exp(-my.etas))
plot(my.etas, my.prob, type="l", bty="n",
     xlab="linear predictor: log-odds eta",
     ylab="probability of 'success'")
abline(h=0); abline(h=1);
lines(c(-10,0),c(.5,.5), lty=2)
lines(c(0,0),c(0,.5), lty=2)

my.conc = seq(0,2.5,by=.05)
my.etas = -6.469  + 5.567 * my.conc
my.prob = 1/(1+exp(-my.etas))
plot(my.conc, my.prob, type="l", bty="n", adj=1,
     xlab="", ylab="prob. no movement")
mtext("concentration", side=1, line=0.4)
mtext("eta", side=1, line=2.4)
mtext("-6.5\n(intercept)",side=1,at=0, line=4)
mtext("-0.9\n(-6.5+5.6)",side=1,at=1, line=4)
conc.5 = (0-(-6.469))/5.567
mtext("0",side=1,at=conc.5, line=3)
mtext("4.7\n(-6.5+2*5.6)",side=1,at=2, line=4)
lines(c(-1,conc.5),c(.5,.5), lty=2)
lines(c(conc.5,conc.5),c(0,.5), lty=2)
```

Plot of movement probability versus concentration

```
plot(movement ~ conc, data=dat)
plot(movement ~ as.factor(conc), data=dat)
plot(nomove ~ conc, data=dat)
plot(jitter(nomove) ~ conc, data=dat)
plot(jitter(nomove,amount=.02) ~ conc, data=dat)

myconc = seq(0.8,2.5,by=.05)
lines(myconc, predict(fit.raw, type="response",
                      list(conc = myconc)))
```

Logistic regression in R on summary data

```
> with(dat, table(movement, conc))
          conc
movement 0.8 1 1.2 1.4 1.6 2.5
  move     6 4   2   2   0   0
  noMove   1 1   4   4   4   2

> dat2 = data.frame( conc = c(.8,1,1.2,1.4,1.6,2.5),
+                    total = c(7,5,6,6,4,2),
+                    prop = c(1/7,1/5,4/6,4/6,4/4,2/2)
+                  )

> fit.tot = glm(prop ~ conc, data=dat2, weights=total,
+                  family=binomial)
> predict(fit.tot, type="link")
    1     2     3     4     5     6
-2.02 -0.90  0.21  1.32  2.44  7.45
> predict(fit.tot, type="response")
    1     2     3     4     5     6
 0.12  0.29  0.55  0.79  0.92  1.00
```

Logistic regression in R on summary data

```
> summary(fit.tot)
glm(formula = prop ~ conc, family=binomial, data=dat2,
    weights = total)
            Estimate Std. Error z value Pr(>|z|)
(Intercept)   -6.469      2.419  -2.675  0.00748 **
conc           5.567      2.044   2.724  0.00645 **
...
    Null deviance: 15.4334  on 5  degrees of freedom
Residual deviance:  1.7321  on 4  degrees of freedom
AIC: 13.811

> plot(prop ~ conc, data=dat2)
> lines(myconc, predict(fit.raw, type="response",
                        list(conc=myconc))
+       )
```

Runoff data set

- Data collected over a 4-year period from a Madison home.
- Outcome: indicator if a rain storm produces runoff.
- Multiple predictors. From graphical examinations: the *total amount of precipitation* and various measures of *storm intensity* are good predictors.
  Storm duration and time since the previous storm are less predictive.

```
runoff = read.table("runoff.txt",header=T)

plot(       RunoffEvent              ~ Precip, data=runoff)
plot(jitter(RunoffEvent,amount=.02) ~ Precip, data=runoff)

library(lattice)
densityplot(~ StormDuration,groups=factor(RunoffEvent),
            data=runoff,auto.key=list(columns=2))
densityplot(~ Precip,groups=factor(RunoffEvent),
            data=runoff,auto.key=list(columns=2))
```

Fitting a logistic model in R: `glm`

We first study a model with storm total precipitation as a single predictor: `Precip`, in inches.

```
> fit1 = glm(RunoffEvent ~ Precip, data=runoff,
+            family=binomial)
> summary(fit1)

glm(formula = RunoffEvent ~ Precip, family=binomial,
                                    data=runoff)
(Intercept)  -3.6418     0.4152  -8.771  < 2e-16 ***
Precip        3.8059     0.5801   6.560 5.37e-11 ***

    Null deviance: 227.82  on 230  degrees of freedom
Residual deviance: 148.13  on 229  degrees of freedom
AIC: 152.13
```

Fitted Model

The general logistic regression formula is

$$\mathbb{P}\{y_i = 1\} = \frac{e^{\eta_i}}{1 + e^{\eta_i}} = \frac{1}{1 + \exp(-\eta_i)}$$

where $\eta_i = X_i\hat{\beta}$. So the probability of runoff in this model is:

$$\mathbb{P}\{\text{runoff}\} = \frac{1}{1 + \exp(-(-3.64 + 3.81 * \text{Precip}))}$$

To plot the prediction curve:

```
plot(jitter(RunoffEvent, amount=.02) ~ Precip, data=runoff)
lines(myprecip, predict(fit1,  list(Precip = myprecip),
                        type="response")
      )
```

Finding the 50/50 point

In general:

$$p = \frac{1}{1 + \exp(-\eta)} \quad \text{or equivalently} \quad \eta = \log\left(\frac{p}{1-p}\right)$$

At the 50/50 point, there is a 50% chance of runoff and 50% chance of no runoff. The odds are 50:50, or 1:1 or just $p/(1-p) = 1$, and the log of the odds is $\eta = \log(1) = 0$.

With one predictor (plus an intercept), we want to solve:

$$\hat{\eta} = \hat{\beta}_1 + \hat{\beta}_2 * \text{Precip} = \log(1) = 0$$

so

$$\text{Precip} = -\frac{\hat{\beta}_1}{\hat{\beta}_2} = -\frac{-3.64}{3.81} = 0.96 \text{ in}$$

Interpreting coefficients

- Intercept: related to predictions when the predictor has value 0. Here we estimate $\mathbb{P}\{\text{runoff}|\text{precip} = 0\} = 0.025$.
- Slope: determines how steeply the probability of runoff moves from 0 to 1, as precipitation increases. Roughly:

  $$\text{slope}/4 \approx \text{change in probability, around the 50:50 point}$$

  Here: $3.81/4 = 0.95$. Because this is so high, we need to consider smaller changes than one unit. When the precipitation is **near the 50:50 point** (near one inch), an increase of 0.1 inch of precipitation increases the runoff probability by about 0.09.

Predictions

At the linear ‘link’ level, or at the response level:

```
> newdat = data.frame(Precip=c(0, 0.25, 0.5, 0.75, 1.0, 1.1,
+                              1.25, 1.5, 1.75, 2.0, 4.0)
+                     )
> predict(fit1, newdat)
   1    2    3     4     5     6     7     8     9    10   11
-3.6 -2.7 -1.7 -0.79  0.16  0.54  1.12  2.07  3.02  3.97 11.6
> predict(fit1, newdat, type="response")
   1    2    3    4    5    6    7    8    9   10   11
0.03 0.06 0.15 0.31 0.54 0.63 0.75 0.89 0.95 0.98  1.0
```

Adding another predictor

Maximum intensity at 10 minutes: in/hr

```
> fit2 = glm(RunoffEvent ~ Precip + MaxIntensity10,
+            data=runoff, family=binomial)
> summary(fit2)
glm(formula=RunoffEvent ~ Precip + MaxIntensity10,
    family=binomial, data=runoff)
               Estimate Std. Error z value Pr(>|z|)
(Intercept)     -4.9017     0.6157  -7.961 1.70e-15 ***
Precip           2.8148     0.6750   4.170 3.05e-05 ***
MaxIntensity10   1.8377     0.3753   4.896 9.78e-07 ***
...
    Null deviance: 227.82  on 230  degrees of freedom
Residual deviance: 116.11  on 228  degrees of freedom
AIC: 122.11
```

What is the equation for η? for the probability *p* of runoff?

Including an interaction

```
> fit3 = glm(RunoffEvent ~ Precip * MaxIntensity10,
+            data=runoff, family=binomial)
> summary(fit3)
glm(formula=RunoffEvent ~ Precip * MaxIntensity10,
    family=binomial, data=runoff)
                      Estimate Std. Error z value Pr(>|z|)
(Intercept)            -5.4276     0.8581  -6.325 2.53e-10 ***
Precip                  3.5900     1.0376   3.460  0.00054 ***
MaxIntensity10          2.4211     0.6911   3.503  0.00046 ***
Precip:MaxIntensity10  -0.8447     0.7707  -1.096  0.27308
...
    Null deviance: 227.82  on 230  degrees of freedom
Residual deviance: 115.27  on 227  degrees of freedom
AIC: 123.27
```

What is the equation for η? for the probability *p* of runoff?

Plots

Without interaction, the curves are parallel: just shifted.

With interaction: some curves are steeper than others.

```
plot(jitter(RunoffEvent,amount=.02) ~ Precip, data=runoff,
     ylab="Probability of runoff event")
legend("right",pch=1,col=c("blue","darkblue","black"),
        legend=c("1.0","0.8","0.24"),title="MaxIntensity10")

myprecip = seq(0,5,0.02)              # calculate predictions
prob1 = predict(fit2,type="response",
             data.frame(Precip=myprecip, MaxIntensity10=0.24))
prob2 = predict(fit2,type="response",
             data.frame(Precip=myprecip, MaxIntensity10=0.80))
prob3 = predict(fit2,type="response",
             data.frame(Precip=myprecip, MaxIntensity10=1.00))

lines(myprecip, prob1, col="black")  # draw prediction curves
lines(myprecip, prob2, col="darkblue")
lines(myprecip, prob3, col="blue")

abline(h=0,lty=2)                    # Add horizontal lines
abline(h=1,lty=2)
```

Case study: Baby food

Number of infant respiratory disease (bronchitis or pneumonia) in their first year of life:

	Bottle only	Some breast with supplement	Breast only
Boys	77/458	19/147	47/494
Girls	48/384	16/127	31/464

How could we test an effect of food

- ignoring a possible gender effect?
- among boys only?

How could we test an effect of gender, ignoring a possible food effect?

Case study: Baby food

```
> babyfood = read.table("babyfood.txt", header=T)

# re-ordering the food levels, non-alphabetically:
> babyfood$food = factor(babyfood$food,
+                        levels = c("bottle","mixed","breast"))

# calculate number of non-disease cases:
> babyfood$nondisease = with(babyfood, total - disease)

> xtabs(disease/total ~ sex+food, babyfood)
      food
sex         bottle      mixed     breast
  boy  0.16812227 0.12925170 0.09514170
  girl 0.12500000 0.12598425 0.06681034

> plot(xtabs(disease/total ~ sex+food, babyfood),
+      main="Respiratory disease incidence in 1st year")
> plot(xtabs(disease/total ~ food+sex, babyfood),
+      main="Respiratory disease incidence in 1st year")
```

Chi-square test of association

Inappropriate if gender effect, which we don't know yet.

```
> l1 = with(babyfood, tapply(disease,    food, sum))
> l2 = with(babyfood, tapply(nondisease, food, sum))
> l1
bottle  mixed breast
   125     35     78
> l2
bottle  mixed breast
   717    239    880
> cbind(l1, l2)
         l1  l2
bottle 125 717
breast  78 880
mixed   35 239

> chisq.test(cbind(l1,l2))
        Pearson's Chi-squared test
data:  cbind(l1, l2)
X-squared = 20.348, df = 2, p-value = 3.815e-05
```

Logistic model

```
> fit = glm(disease/total ~ sex + food, weight=total,
+             family=binomial, data=babyfood)
> fit = glm(cbind(disease, nondisease) ~ sex + food,
             family=binomial, data=babyfood)

> summary(fit)

...
Coefficients:
            Estimate Std. Error z value Pr(>|z|)
(Intercept)  -1.6127     0.1124 -14.347  < 2e-16 ***
sexgirl      -0.3126     0.1410  -2.216   0.0267 *
foodmixed    -0.1725     0.2056  -0.839   0.4013
foodbreast   -0.6693     0.1530  -4.374 1.22e-05 ***
...
    Null deviance: 26.37529  on 5  degrees of freedom
Residual deviance:  0.72192  on 2  degrees of freedom
AIC: 40.24
```

Interpretation of coefficients: odds

```
             Estimate Std. Error z value Pr(>|z|)
(Intercept)  -1.6127     0.1124 -14.347  < 2e-16 ***
sexgirl      -0.3126     0.1410  -2.216   0.0267 *
foodmixed    -0.1725     0.2056  -0.839   0.4013
foodbreast   -0.6693     0.1530  -4.374 1.22e-05 ***
```

Let $p =$ probability of infant respiratory disease. With $o = \mathrm{e}^{\eta}$,

$$p = \frac{o}{1+o}, \quad o = \frac{p}{1-p} = \frac{\mathbb{P}\{\text{disease}\}}{\mathbb{P}\{\text{no disease}\}}$$

$$\eta = \log(o) = \begin{cases} -1.61 & \text{bottle-fed boys} \\ -1.61 - 0.31 = -1.92 & \text{bottle-fed girls} \\ -1.61 - 0.31 - 0.67 = -2.60 & \text{breast-fed girls} \end{cases}$$

or, the odds of respiratory disease are:

$$o = \begin{cases} \exp(-1.61) \sim 1/5 & \text{bottle-fed boys} \\ \exp(-1.61)\exp(-0.31) \sim 1/7 & \text{bottle-fed girls} \\ \exp(-1.61)\exp(-0.31)\exp(-0.67) \sim 1/14 & \text{breast-fed girls} \end{cases}$$

exp(coefficient) is the multiplicative change in odds.

Interpretation of coefficients: odds

Quiz:

Odds	log odds (η)	probability
$o = 100$	$\log(100) = 4.6$	$p =$
$o = 10$	$\log(10) = 2.3$	$p =$
$o = 9$	$\log(9) = 2.2$	$p =$
$o = 7$	$\log(7) = 1.94$	$p =$
$o = 1$	$\log(1) = 0$	$p =$
$o = 1/7$	$\log(1/7) = -1.94$	$p =$
$o = 1/9$	$\log(1/9) = -2.2$	$p =$
$o = 0.1$	$\log(0.1) = -2.3$	$p =$

$\exp(-0.6693) = 0.512$: breastfeeding reduces the odds of respiratory disease to 51% of that for bottle feeding:
For girls: from $o \approx 1/7$ ($p = 0.13$) to $o \approx$
For boys: from $o \approx 1/5$ ($p = 0.17$) to $o \approx$

Outline

1. Logistic regression: fitting the model
   - Components of generalized linear models
   - Logistic regression
   - Case study: runoff data
   - Case study: baby food
2. Logistic regression: Inference
   - Model fit and model diagnostics
   - Comparing models
   - Sparse data and the separation problem

Model fit and the residual deviance

If the model is correct and when n_i's are large, the residual deviance D has a chi-square distribution approximately:

$$\text{residual } D \sim \chi^2_{\text{dfResid}}$$

If D is too large, or p-value too small: the model does not capture all the features in the data.

Example: baby food.

```
> summary(fit)
... Null deviance: 26.37529  on 5  degrees of freedom
Residual deviance:  0.72192  on 2  degrees of freedom
> pchisq(0.72192, df=2, lower.tail=F)
[1] 0.6970069
```

No sign of lack of fit: the model fits well enough. This test is valid because sample sizes n_i are large:

```
> babyfood$total
[1] 458 147 494 384 127 464
```

Model fit and the residual deviance

Warning: The chi-square approximation is very bad when n_i's are small. This chi-square test is worthless when all $n_i = 1$.

Example: anesthesia data, fit on raw 0/1's versus grouped totals:

```
> summary(fit.raw)
... Residual deviance: 27.754  on 28  degrees of freedom
> summary(fit.tot)
... Residual deviance:  1.7321  on 4  degrees of freedom

> pchisq(27.754, df=28, lower.tail=F)
[1] 0.4775395   # don't trust this one
> pchisq(1.7321, df=4, lower.tail=F)
[1] 0.7848787   # this one is more trustworthy (but how much?)
```

Response residuals

$$y_i - \hat{y}_i$$

Example: anesthetic on raw 0/1 data:

observed	1	0	1	0	...
predicted	0.29	0.55	0.79	0.79	...
residual	0.71	-0.55	0.21	-0.79	...

on group totals:

observed	0.14	0.20	0.67	0.67	1.00	1.00
predicted	0.12	0.29	0.55	0.79	0.92	0.9994
residual	0.03	-0.09	0.11	-0.12	0.08	0.0006

```
> residuals(fit.raw, type="response")[1:4]
> residuals(fit.tot, type="response")
```

But we expect unequal variances: smaller when p is close to 0 or 1, larger when $p \sim 0.5$: $\text{var}(y_i) = p(1-p)/n_i$

Pearson's residuals

$$\frac{y_i - \hat{y}_i}{\sqrt{\text{var}(\hat{y}_i)}}$$

Example: anesthetic on raw 0/1 data:

observed	1	0	1	0	...
predicted	0.29	0.55	0.79	0.79	...
residual	1.57	-1.11	0.52	-1.94	...

on group totals:

observed	0.14	0.20	0.67	0.67	1.00	1.00
predicted	0.12	0.29	0.55	0.79	0.92	0.9994
residual	0.21	-0.44	0.56	-0.74	0.59	0.03

```
> residuals(fit.raw, type="pearson")
> residuals(fit.tot, type="pearson")
```

Their variance should be more uniform.

Deviance residuals

$$r_i^D = \text{sign}(y_i - \hat{y}_i) * \sqrt{d_i}$$

where d_i is the contribution of observation i to the (residual) deviance:

$$d_i = 2\left(y_i \log\frac{y_i}{\hat{y}_i} + (n_i - y_i)\log\frac{n_i - y_i}{n_i - \hat{y}_i}\right)$$

They are the default in R, and often quite similar to Pearson's residuals:

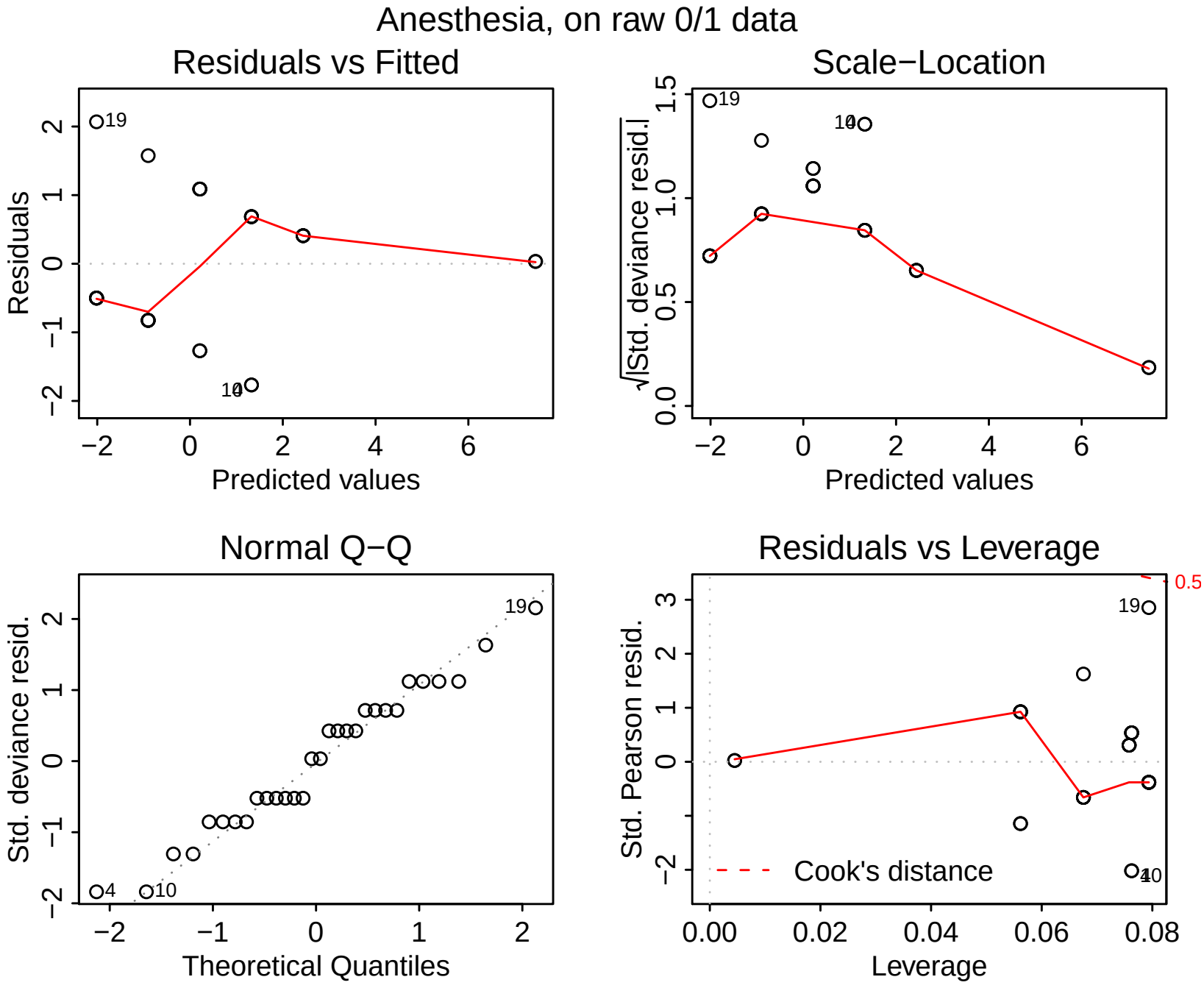
```
> residuals(fit.raw)
    1     2     3     4
 1.58 -1.27  0.69 -1.77 ...
> residuals(fit.tot)
    1     2     3     4     5     6
 0.20 -0.45  0.57 -0.70  0.82  0.05
```

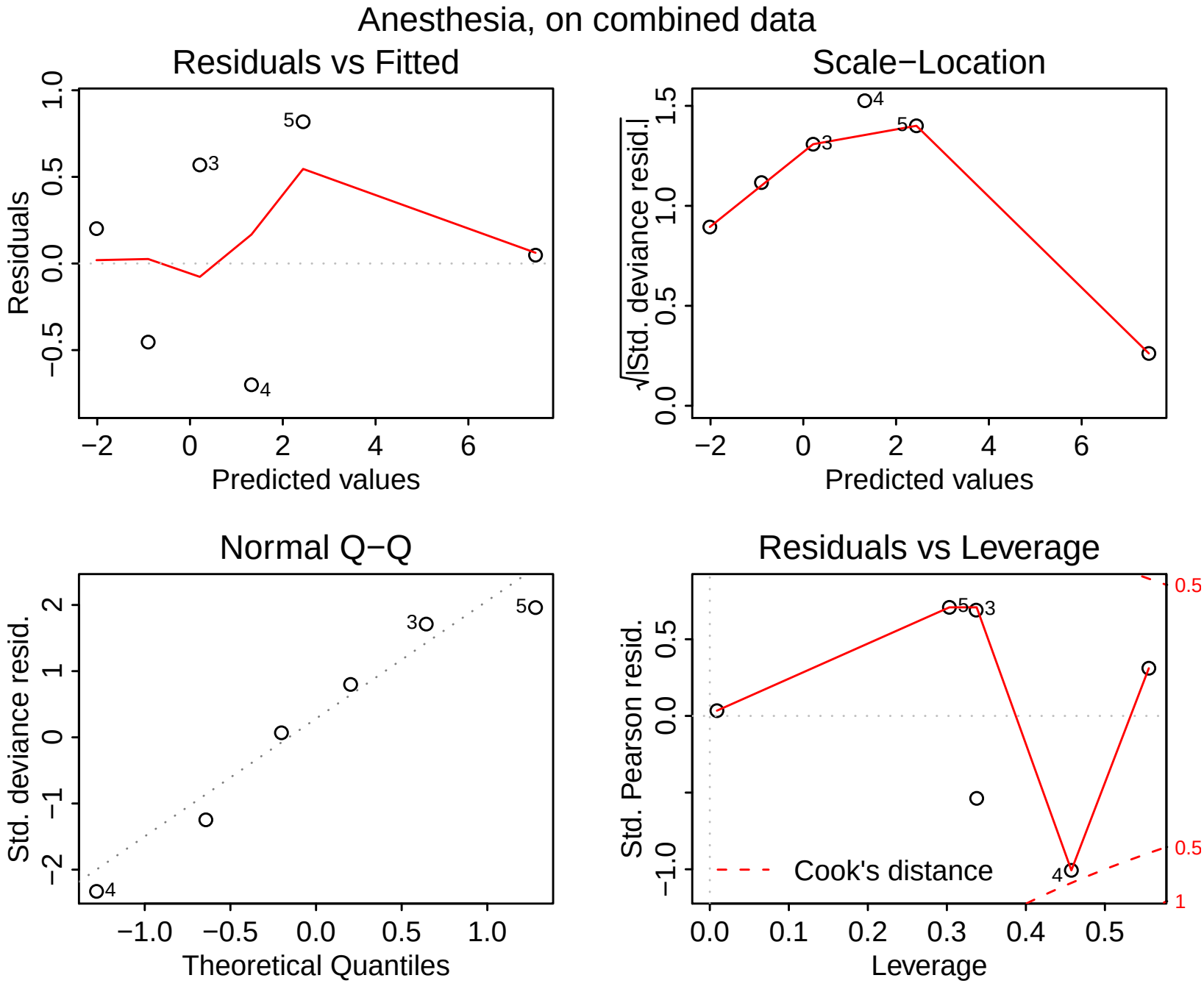
In standard linear models, these residuals coincides.

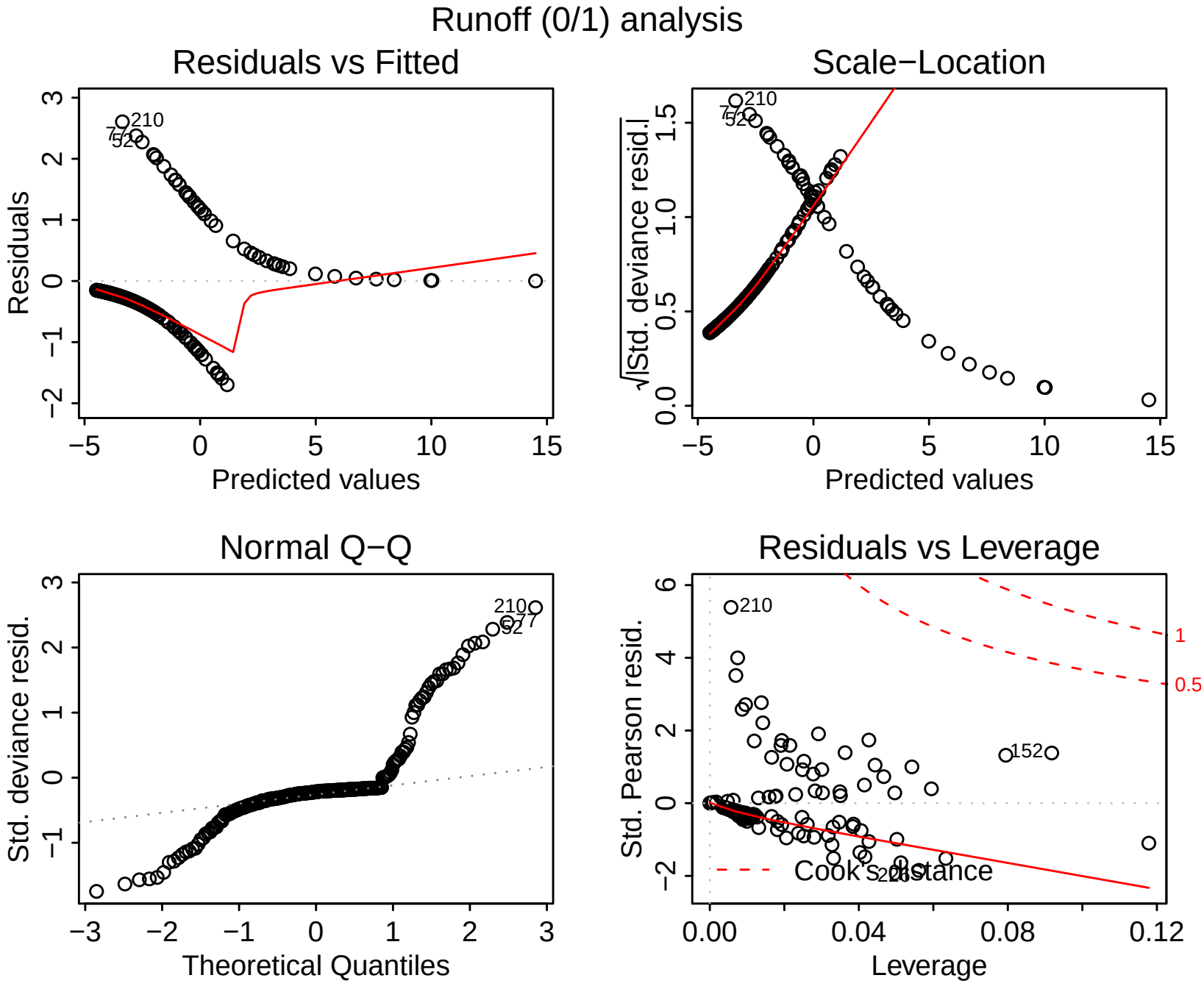
Residual plots

- Deviance residuals are most appropriate for residual plots.
- Plotting predicted values on the linear (link) scale is best.
- Residual plots are almost useless when $n_i = 1$: predictable pattern

```
> layout(matrix(1:4,2,2))
> plot(fit.raw)
> plot(fit.tot)
> plot(fit2)   # from runoff data: were 0/1 response values
```


Anesthesia, on raw 0/1 data
Residuals vs Fitted
Residuals
Predicted values
Scale−Location
√|Std. deviance resid.|
Predicted values
Normal Q−Q
Std. deviance resid.
Theoretical Quantiles
Residuals vs Leverage
Std. Pearson resid.
Cook's distance
0.5
Leverage


Anesthesia, on combined data
Residuals vs Fitted
Residuals
Predicted values
Scale−Location
√|Std. deviance resid.|
Predicted values
Normal Q−Q
Std. deviance resid.
Theoretical Quantiles
Residuals vs Leverage
Std. Pearson resid.
Cook's distance
Leverage


Runoff (0/1) analysis
Residuals vs Fitted
Residuals
Predicted values
210
77
52
Scale−Location
√|Std. deviance resid.|
Predicted values
Normal Q−Q
Std. deviance resid.
Theoretical Quantiles
Residuals vs Leverage
Std. Pearson resid.
Leverage
Cook's distance
1
0.5
152
226

Why is the deviance is too large?

A large residual deviance (as compared to a chi-square distribution) suggests a bad fit. Ways to correct this:

- include the correct predictors in the model
- transform predictors appropriately
- detect if there are a few outliers or a few points with undue influence, using residual plots
- if all/many n_i's are small: the residual deviance is not approximately χ^2, so it is useless to assess goodness of fit.
- if none of the above: consider overdispersion. More later.

Comparing models: chi-square likelihood ratio test

The deviance always goes down as more predictors are added to the model, just like RSS goes down (R^2 goes up) in linear models.

χ^2 test (LRT) for nested models

If the reduced model is true, then

$$D_{\text{reduced}} - D_{\text{full}} \sim \chi^2_{\text{d}}$$

approximately, when d is the difference in degrees of freedom between the two models.

- Much more reliable than the χ^2 test for goodness of fit.
- This is a likelihood-ratio test (LRT)

Comparing models: chi-square test

```
> summary(fit1)
... glm(formula = RunoffEvent ~ Precip,
          family = binomial, data = runoff)
... Residual deviance: 148.13  on 229  degrees of freedom

> summary(fit2)
... glm(formula = RunoffEvent ~ Precip + MaxIntensity10,
         family = binomial, data = runoff)
... Residual deviance: 116.11  on 228  degrees of freedom

> pchisq(148.13-116.11, df=229-228, lower.tail=F)
[1] 1.525e-08

> anova(fit1, fit2, test="Chisq")
Analysis of Deviance Table
Model 1: RunoffEvent ~ Precip
Model 2: RunoffEvent ~ Precip + MaxIntensity10
  Resid. Df Resid. Dev  Df Deviance P(>|Chi|)
1       229    148.129
2       228    116.106   1   32.023 1.524e-08
```

Comparing models: chi-square test

```
> anova(fit2, test="Chisq")  # Warning! sequential

Analysis of Deviance Table
Model: binomial, link: logit
Response: RunoffEvent
Terms added sequentially (first to last)

                Df Deviance Resid. Df Resid. Dev P(>|Chi|)
NULL                              230    227.820
Precip           1   79.691       229    148.129 4.378e-19
MaxIntensity10   1   32.023       228    116.106 1.524e-08

> drop1(fit2, test="Chisq") # each term against the full model

Single term deletions
Model: RunoffEvent ~ Precip + MaxIntensity10
               Df Deviance     AIC     LRT   Pr(Chi)
<none>             116.106 122.106
Precip          1  136.717 140.717  20.611 5.628e-06 ***
MaxIntensity10  1  148.129 152.129  32.023 1.524e-08 ***
```

Comparing models: chi-square test

```
> anova(fit1, fit2, fit3, test="Chisq")
Analysis of Deviance Table

Model 1: RunoffEvent ~ Precip
Model 2: RunoffEvent ~ Precip + MaxIntensity10
Model 3: RunoffEvent ~ Precip * MaxIntensity10
  Resid. Df Resid. Dev  Df Deviance P(>|Chi|)
1       229    148.129
2       228    116.106   1   32.023 1.524e-08
3       227    115.273   1    0.833     0.361
```

AIC = Deviance $+2p$, where p = total # coefficients

```
> extractAIC(fit1)
[1]   2.0000 152.1287
> extractAIC(fit2)
[1]   3.0000 122.1059
> extractAIC(fit3)
[1]   4.0000 123.2725
```

Wald test for coefficients

- Standard errors for coefficients obtained as in linear models, using matrix algebra.
- Wald test: z-test here. Approximate. Roughly speaking, a coefficient will be **statistically significant** if it is **at least two standard errors away from zero**.
- The chi-square test using deviances is more reliable.
- It rarely makes sense to test the intercept.

```
> summary(fit2)
                Estimate Std. Error z value Pr(>|z|)
(Intercept)      -4.9017     0.6157  -7.961 1.70e-15 ***
Precip            2.8148     0.6750   4.170 3.05e-05 ***
MaxIntensity10    1.8377     0.3753   4.896 9.78e-07 ***
```

Confidence intervals for coefficients, Wald-based

- Confidence intervals associated with Wald test: on the linear scale.
- Transform with `exp` to have CI for the change in odds.
- Symmetric interval around the coefficient, not symmetric on the odds scale.

```
> summary(fit)
...
             Estimate Std. Error z value Pr(>|z|)
sexgirl       -0.3126     0.1410  -2.216   0.0267 *
foodmixed     -0.1725     0.2056  -0.839   0.4013
foodbreast    -0.6693     0.1530  -4.374 1.22e-05 ***

# CI for breastfeeding effect:
> c(-0.6693 - 2*0.1530, -0.6693 + 2*0.1530)
[1] -0.9753 -0.3633

# CI for change in odds due to breastfeeding:
> exp(c(-0.6693 - 2*0.1530, -0.6693 + 2*0.1530))
[1] 0.3770792 0.6953778
```

Confidence intervals from profile likelihood

- Profile likelihood-based method: include in the interval all the ‘plausible’ values that are not rejected by a LRT.
- This is preferable to Wald-based CI.

```
> library(MASS)
> confint(fit)
Waiting for profiling to be done...
                 2.5 %      97.5 %
(Intercept) -1.8376014 -1.39661429
sexgirl     -0.5912751 -0.03778236
foodmixed   -0.5878196  0.22028446
foodbreast  -0.9723573 -0.37176239

> exp(confint(fit))
Waiting for profiling to be done...
                2.5 %    97.5 %
(Intercept) 0.1591988 0.2474333
sexgirl     0.5536209 0.9629225
foodmixed   0.5555372 1.2464312
foodbreast  0.3781905 0.6895181
```

Sparse data and the separation problem

Growth of *Staphylococcus aureus* in vacuum-packaged ready-to-eat meats. (work with Darand Borneman and Steve Ingham)

data for 68 products:

```
ph:  pH
aw:  water activity
wps:  percent water phase salt
mpr:  moisture protein ratio
growth:  0 (no growth) or 1 (growth)
```

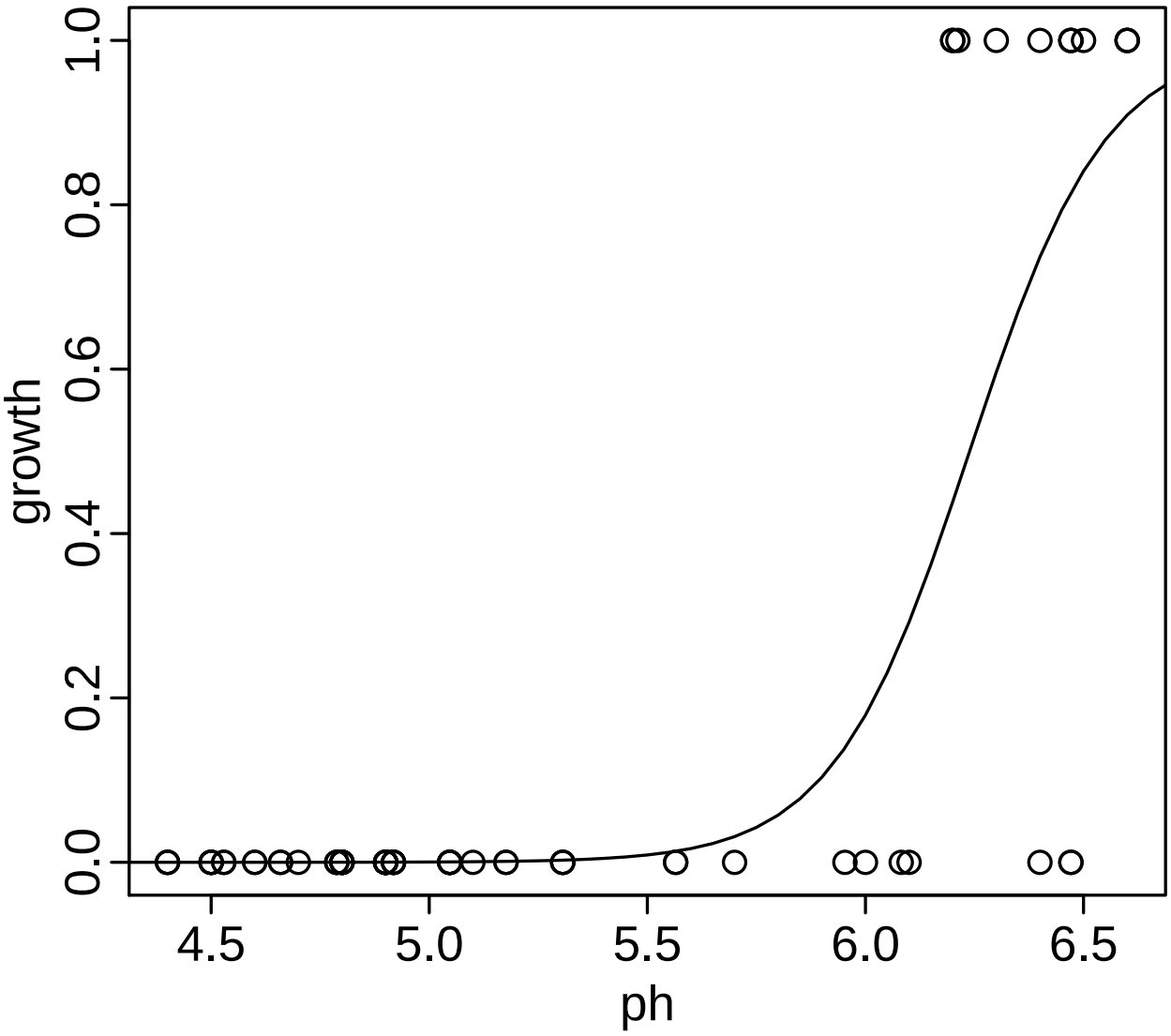
We would like to predict growth of *S. aureus*, and find the best variables to make this prediction.

S. aureus example

Let's predict *S. aureus* growth using pH alone:

```
> rte = read.table("rte.txt", header=T)
> fit.ph = glm(growth ~ ph, family=binomial, data=rte)
> summary(fit.ph)
ph              6.38       2.55   2.502   0.0123 *
Residual deviance: 20.226  on 66  degrees of freedom
AIC: 24.226
> plot(growth ~ ph, data=rte)
> mypH = seq(4,7,by=.05)
> lines(mypH, predict(fit.ph, type="response", list(ph=mypH)))
```

S. aureus growth explained by pH

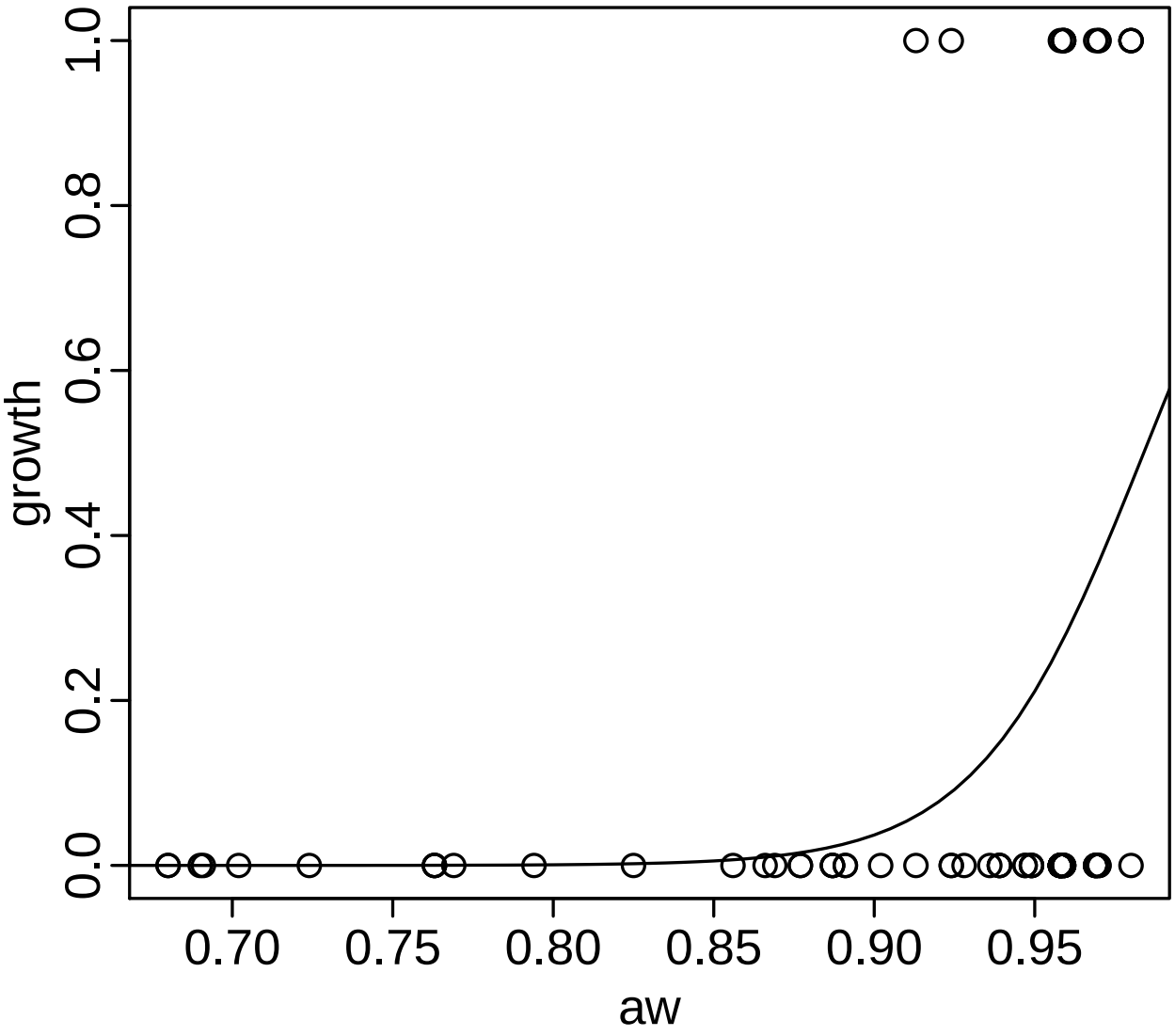


S. aureus growth

Using water activity alone:

```
> fit.aw = glm(growth ~ aw, family=binomial, data=rte)
> summary(fit.aw)
aw              39.48      19.04   2.074   0.0381 *
Residual deviance: 50.871  on 66  degrees of freedom
AIC: 54.871
> plot(growth ~ aw, data=rte)
> myaw = seq(.65,1,by=.005)
> lines(myaw, predict(fit.aw, type="response", list(aw=myaw)))
```

S. aureus growth explained by water activity



S. aureus growth

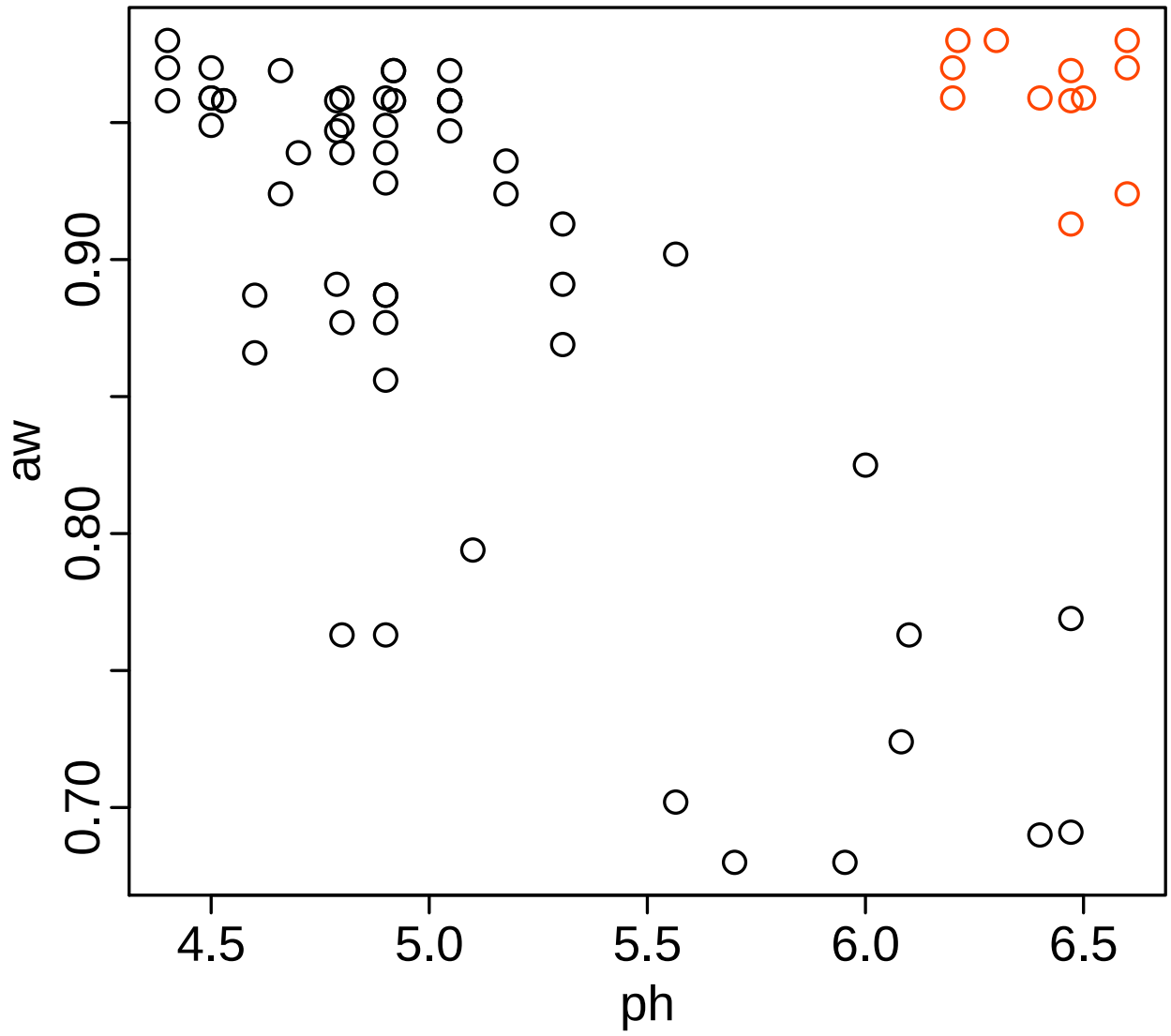
Using both pH and water activity:

```
> fit.awph = glm(growth ~ aw+ph, family=binomial, data=rte)
1: In glm.fit(x=X, y=Y, weights=weights, start=start, etastart=
  algorithm did not converge
2: In glm.fit(x=X, y=Y, weights=weights, start=start, etastart=
  fitted probabilities numerically 0 or 1 occurred
```

What is going on? Let's look at the data (something that should be done before...)

```
> growthcolor = rep(NA, 68)
> growthcolor[rte$growth==0] = "black"
> growthcolor[rte$growth==1] = "orangered"
> plot(aw~ph, data=rte, col=growthcolor)
```

S. aureus growth explained by both pH and aw



Sparse data and the separation problem

When the 0/1 are perfectly separated by a linear combination of the predictors,

- we could fit many, many curves, all providing perfect fit.

  diagnostic: Residual deviance$= 0$.
- the coefficient values providing maximum likelihood are infinite: infinitely steep curve, or step-shaped curve.

  diagnostic: huge SE for individual coefficients and $p = 1$ from Wald test.

Sparse *S. aureus* data diagnostic

```
> fit.awph = glm(growth ~ aw+ph, family=binomial, data=rte)
... error messages
> summary(fit.awph)
             Estimate Std. Error z value Pr(>|z|)
(Intercept)   -556.69  664238.24  -0.001        1
aw             316.63  540050.65   0.001        1
ph              44.78   51071.97   0.001        1
...
    Null deviance: 6.3376e+01  on 67  degrees of freedom
Residual deviance: 1.3577e-09  on 65  degrees of freedom
```

Still, LRT indicates that both aw and pH are significant predictors:

```
> drop1(fit.awph, test="Chisq")
       Df Deviance    AIC    LRT   Pr(Chi)
<none>       0.000  6.000
aw      1   20.229 24.229 20.229 6.870e-06 ***
ph      1   50.995 54.995 50.995 9.262e-13 ***
```

Sparse data and the separation problem

Possible corrections:

- Increase the sampling in the separation zone, so as to obtain some overlap between the cloud of 0's and the cloud of 1's.
- Use a "bias-reduction" approach, which penalizes large coefficients, i.e. penalizes steep curves. The theoretical basis is a reduction bias in estimated coefficients.

S. aureus growth with bias-reduction analysis

brglm package: for ‘bias-reduction’ glm. In active development.

```
> library(brglm)

> fit.awph = brglm(growth ~ aw+ph, family=binomial, data=rte)
> summary(fit.awph)
Coefficients:
             Estimate Std. Error z value Pr(>|z|)
(Intercept)  -53.009     15.095  -3.512 0.000445 ***
aw            25.592     10.773   2.376 0.017517 *
ph             4.948      1.444   3.426 0.000613 ***

    Null deviance: 56.2075  on 67  degrees of freedom
Residual deviance:  3.0725  on 65  degrees of freedom
Penalized deviance: 8.09377
AIC:  9.0725
```

Visualize *S. aureus* growth estimated probability

data and estimated region of 1:1, 4:1 and 1:4 odds of growth:

```
> co =coef(fit.awph)
> co
(Intercept)          aw          ph
  -53.00912    25.59206     4.94773

> b = -co["ph"]/co["aw"]    # slope of line on a aw~ph plot
> a50 = -co[1]/co["aw"]     # intercept of line with 1:1 odds
> a80 = ( log(4) -co[1])/co["aw"] # intercept        4:1 odds
> a20 = (-log(4) -co[1])/co["aw"] # intercept        1:4 odds

> plot(aw~ph, data=rte, col=growthcolor)
> abline(a80,b, col="orangered", lty=3)
> abline(a50,b, col="orangered4")
> abline(a20,b, col="black", lty=3)
> legend("bottomleft", lty=c(3,1,3),title="odds of growth",
+        col=c("orangered","orangered4","black"),
+        legend=c("4:1","1:1","1:4"))
```

S. aureus growth explained by water activity