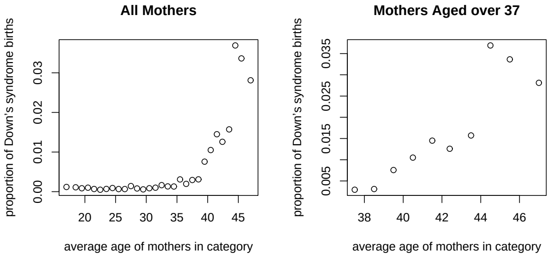
Down's Syndrome Data

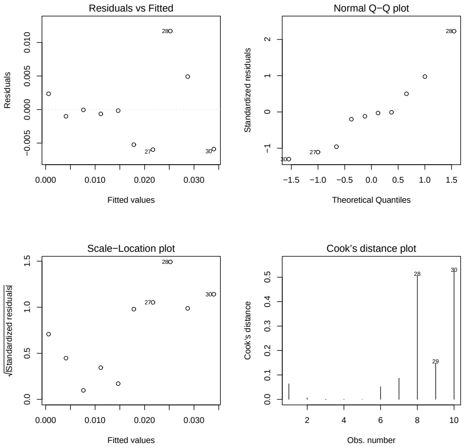
```
> library(boot)
> downs.bc[1:5,]
   age     m  r
1 17.0 13555 16
2 18.5 13675 15
3 19.5 18752 16
4 20.5 22005 22
5 21.5 23896 16
> downs.bc$phat <- downs.bc$r/downs.bc$m
> attach(downs.bc)
> plot(age, phat, xlab="average age of mothers in category",
+ ylab="proportion of Down's syndrome births")
> downs37 <- downs.bc[downs.bc$age > 37,]; attach(downs37)
> plot(age, phat, xlab="average age of mothers in category",
+ ylab="proportion of Down's syndrome births")
>
```



Simple Linear Regression

Fitting a simple linear regression doesn't work too well:

```
> lmfit <- lm(phat ~ age, data=downs37)
> summary(lmfit)
Call:
lm(formula = phat ~ age, data = downs37)
[ . . . ]
Coefficients:
              Estimate Std. Error t value Pr(>|t|)
(Intercept) -0.1313203  0.0260268  -5.046 0.000995 ***
age          0.0035176  0.0006176   5.696 0.000457 ***
[ . . . ]
Residual standard error: 0.005765 on 8 degrees of freedom
Multiple R-Squared: 0.8022,Adjusted R-squared: 0.7775
F-statistic: 32.44 on 1 and 8 DF,  p-value: 0.0004566
> plot(lmfit)
>
```



Variance Is Not Constant

The response r_i/m_i is approximately normally distributed:

$$r_i/m_i \dot\sim \mathcal{N}(p_i, p_i(1-p_i)/m_i)$$

but even if $p_i = \beta_0 + \beta_1 \text{age}_i$, the model is

$$r_i/m_i = \beta_0 + \beta_1 \text{age}_i + \epsilon_i$$

where $\epsilon_i \dot\sim \mathcal{N}(0, \sigma_i^2 = p_i(1-p_i)/m_i)$.

Some possibilities:

- Use a variance-stabilization transformation. A log-transformation of the response happens to work pretty well here but wouldn't work so well if the distribution of the m_is was substantially different or if the range of $\widehat{p}_i$s was bigger.
- Use a more appropriate theoretical framework for fitting a model to a binomial response.

Logistic Regression

For Down's Syndrome data, the theoretical model is:

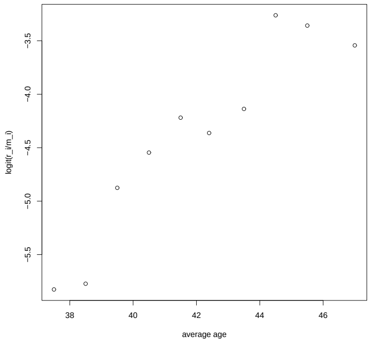
$$r_i \overset{\text{ind}}{\sim} \text{Binomial}(m_i, p_i)$$

for m_i known, p_i unknown, and where $p_i = \mathsf{E}(r_i/m_i)$ is explained by the linear component

$$\eta_i = \beta_0 + \beta_1 \text{age}_i$$

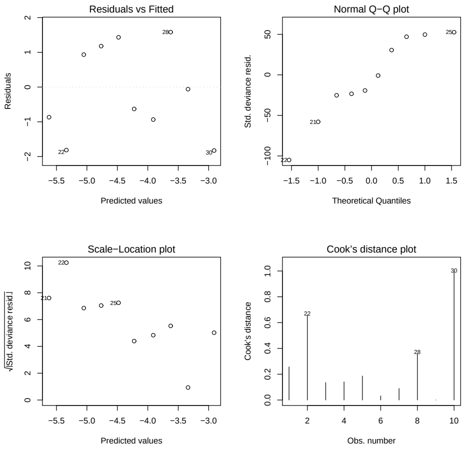
through the link $\text{logit}(p_i) = \eta_i$.

If the link and form of η are reasonable, then a plot of $\text{logit}(r_i/m_i)$ versus age_i should look roughly linear:



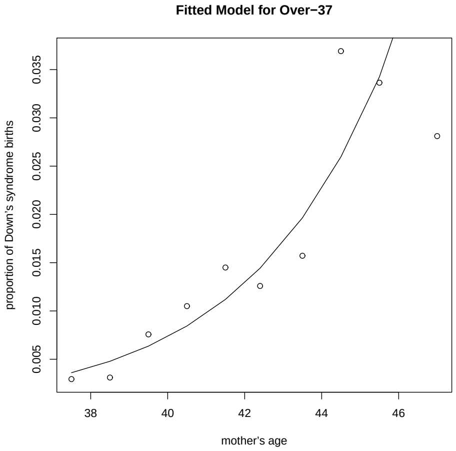
Logistic Regression

```
> glmfit <- glm(cbind(r,m-r) ~ age, family=binomial, data=downs37)
> summary(glmfit)
Call:
glm(formula = cbind(r, m - r) ~ age, family = binomial, data = downs37)
Deviance Residuals:
    Min       1Q   Median       3Q      Max
-1.8270  -0.9184  -0.3449   1.1178   1.5823
Coefficients:
             Estimate Std. Error z value Pr(>|z|)
(Intercept) -16.32407    1.09533  -14.90   <2e-16 ***
age           0.28538    0.02638   10.82   <2e-16 ***
[ . . . ]
> plot(glmfit)
>
```



The Final Fit

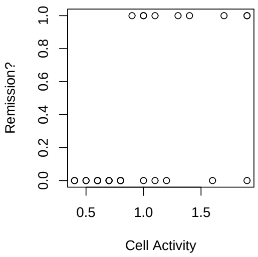
```
> attach(downs37)
> plot(age, phat, xlab="mother’s age",
+ ylab="proportion of Down’s syndrome births",
+ main="Fitted Model for Over-37")
> lines(age,fitted(glmfit))
```

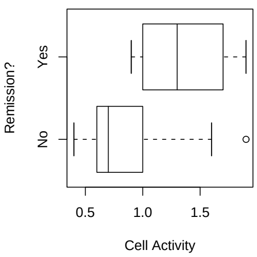


Binary (Bernoulli) Data

A common special case: $Y_i \sim$ Binomial$(1, p_i)$

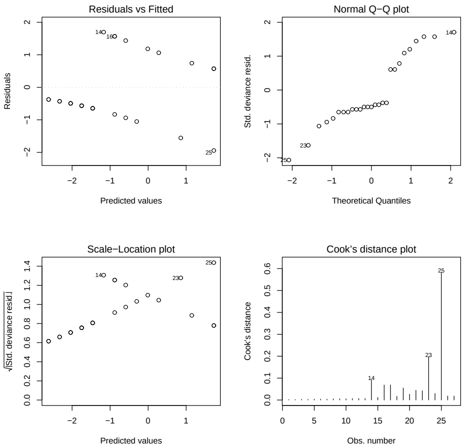
```
> library(boot)
> remission
   LI m r
1  0.4 1 0
2  0.4 1 0
[ . . . ]
26 1.9 1 1
27 1.9 1 1
> attach(remission)
> plot(LI,r, xlab="Cell Activity", ylab="Remission?")
> boxplot(LI ~ r, horizontal=T, xlab="Cell Activity",
+         ylab="Remission?", names=c("No","Yes"))
> wilcox.test(LI ~ r)
Wilcoxon rank sum test with continuity correction
[ . . . ]
W = 23.5, p-value = 0.00326
alternative hypothesis: true mu is not equal to 0
[ . . . ]
```





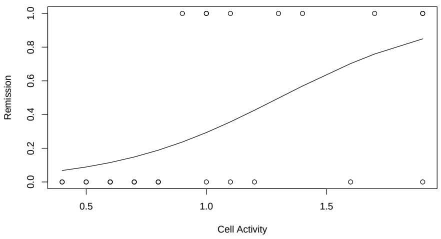
Logistic Regression

```
> glmfit <- glm(r ~ LI, family=binomial, data=remission)
> summary(glmfit)
Call:
glm(formula = r ~ LI, family = binomial, data = remission)
Deviance Residuals:
    Min       1Q   Median       3Q      Max
-1.9449  -0.6465  -0.4947   0.6571   1.6971
Coefficients:
            Estimate Std. Error z value Pr(>|z|)
(Intercept)   -3.777      1.379  -2.740  0.00615 **
LI             2.897      1.187   2.441  0.01464 *
[ . . . ]
> plot(glmfit)
>
```



The Fit

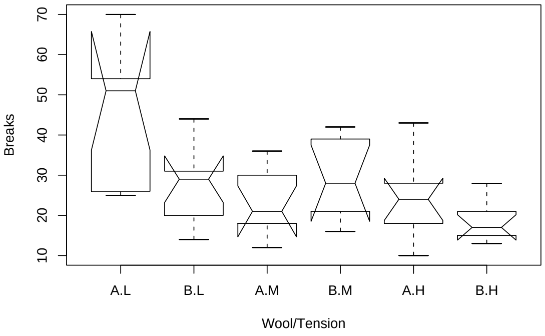
```
> plot(LI,r,xlab="Cell Activity",ylab="Remission")
> lines(LI,fitted(glmfit))
```



```
> predict(glmfit,newdata=data.frame(LI=1.0))
[1] -0.8798763
> predict(glmfit,newdata=data.frame(LI=1.0), type="response")
[1] 0.2932034
> logit(.2932034)
[1] -0.8798764
> LI.new <- seq(.2,2.0,by=.2)
> round(rbind(LI.new,
+            prob=predict(glmfit,
+               newdata=data.frame(LI=LI.new),
+               type="response")),2)
       [,1] [,2] [,3] [,4] [,5] [,6] [,7] [,8] [,9] [,10]
LI.new 0.20 0.40 0.60 0.80 1.00 1.20 1.40  1.6 1.80  2.00
prob   0.04 0.07 0.12 0.19 0.29 0.43 0.57  0.7 0.81  0.88
>
```

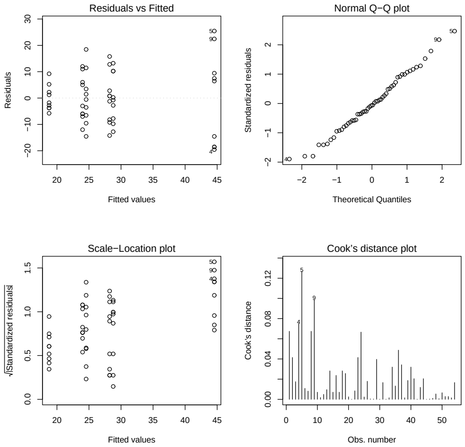
Breaks in Yarn

```
> warpbreaks
   breaks wool tension
1      26    A       L
2      30    A       L
[ . . . ]
53     16    B       H
54     28    B       H
> attach(warpbreaks)
> table(wool,tension)
    tension
wool L M H
   A 9 9 9
   B 9 9 9
> boxplot(breaks ~ wool:tension,
+ xlab="Wool/Tension",ylab="Breaks",notch=T)
```



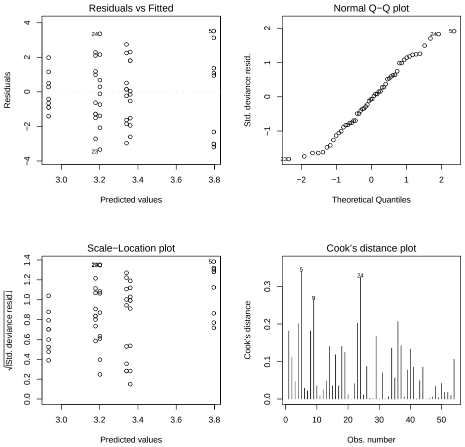
Two-Factor ANOVA

```
> lmfit <- lm(breaks ~ wool:tension-1, data=warpbreaks)
> summary(lmfit)
[ . . . ]
               Estimate Std. Error t value Pr(>|t|)
woolA:tensionL   44.556      3.647  12.218 2.43e-16 ***
woolB:tensionL   28.222      3.647   7.739 5.47e-10 ***
woolA:tensionM   24.000      3.647   6.581 3.23e-08 ***
woolB:tensionM   28.778      3.647   7.891 3.22e-10 ***
woolA:tensionH   24.556      3.647   6.734 1.88e-08 ***
woolB:tensionH   18.778      3.647   5.149 4.84e-06 ***
[ . . . ]
> plot(lmfit)
>
```



Poisson Regression

```
> glmfit <- glm(breaks ~ wool:tension-1, family=poisson,
+ data=warpbreaks)
> summary(glmfit)
[ . . . ]
               Estimate Std. Error z value Pr(>|z|)
woolA:tensionL  3.79674    0.04994   76.03   <2e-16 ***
woolB:tensionL  3.34011    0.06275   53.23   <2e-16 ***
woolA:tensionM  3.17805    0.06804   46.71   <2e-16 ***
woolB:tensionM  3.35960    0.06214   54.07   <2e-16 ***
woolA:tensionH  3.20094    0.06727   47.59   <2e-16 ***
woolB:tensionH  2.93267    0.07692   38.12   <2e-16 ***
[ . . . ]
> plot(glmfit)
>
```



Some `glm(...)` Warnings

- Don't forget the `family`! (If you do, R will default to Gaussian, and it'll be as if you used `lm`.)
- For binomial, the second column of the response is the number of *failures*, **not** the number of *trials*.
- The `fitted(...)` values are fitted responses, but the default `predict(...)`ed values are predicted η values, not predicted responses. (You must specify `type="response"` to get those.)
- There are many different types of residuals for `glm` models. `resid(...,type="response")` is the difference between response and fitted response, and it is *not* supposed to have constant variance across observations. The residuals used in the diagnostic plots are "standardized deviance residuals." These ones *should* have constant variance.