Regression Models For Count Data

Statistics 149

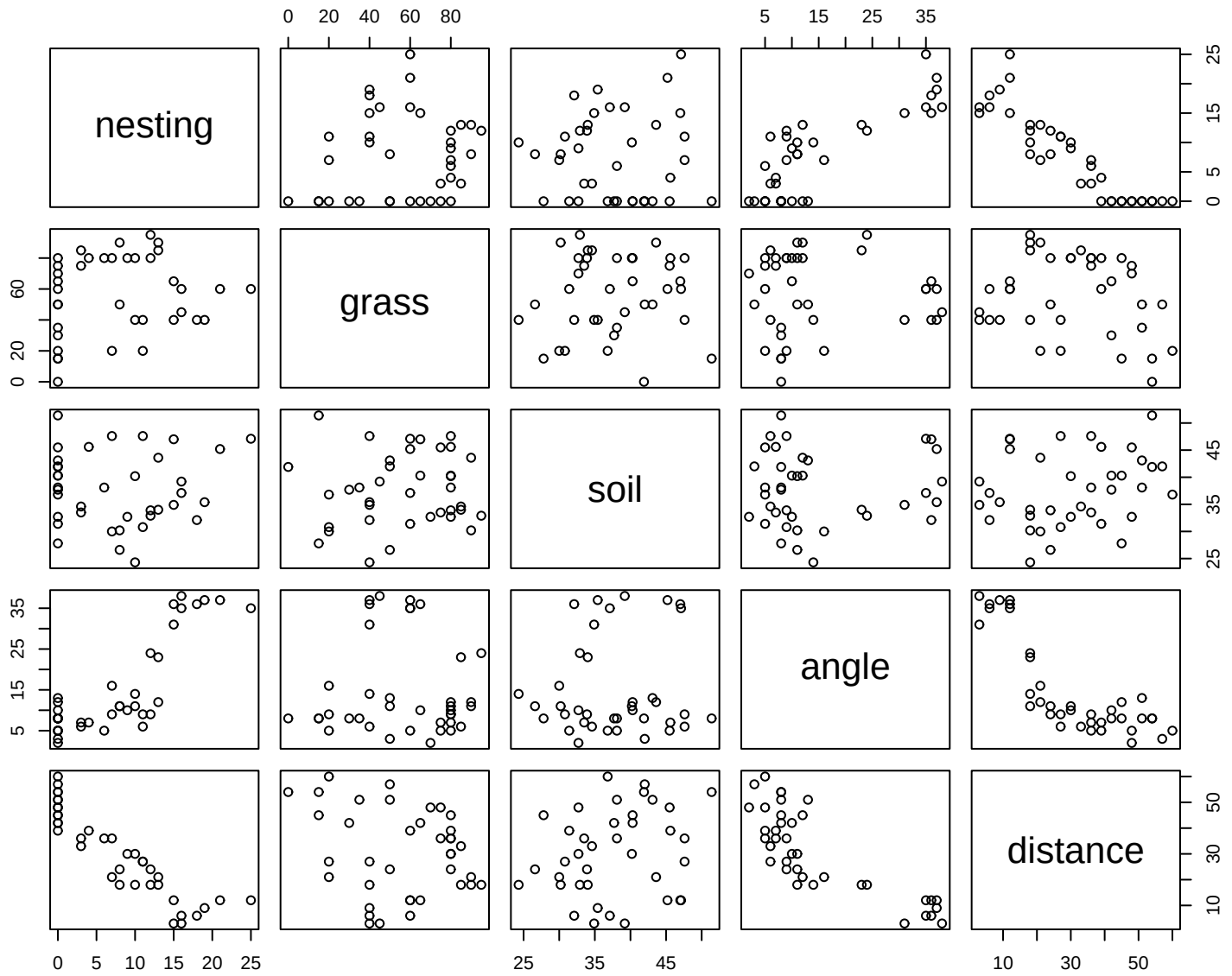
Spring 2006

Poisson Regression Model

Puffin Nesting: Based on the dataset from the article "Breeding Success of the Common Puffin on Difference Habitats at Great Island, Newfoundland"

Four variables where considered in trying to describe the nesting frequency of the common puffin in a $3m \times 3m$ grid of plots.

- nesting: number of nests per $9m^2$
- grass: grass cover percentage
- soil: mean soil depth in cm
- angle: angle of slope in degrees
- distance: distance from cliff edge in m


nesting
grass
soil
angle
distance

Poisson Regression Model

- Distribution: $y_i|x_{1i}, \ldots, x_{pi} \overset{ind}{\sim} Poisson(\mu_i)$
- Link function: $g(\mu_i) = \beta_0 + \beta_1 x_{1i} + \ldots + \beta_p x_{pi}$

  The common choices for the link function are

  - `log`: $g(\mu) = \log \mu$
    This is the canonical link and leads to the multiplicative model as

    $$\mu(x+1) = e^{\beta_0 + \beta_1 (x+1)} = e^{\beta_0 + \beta_1} e^{\beta_1}$$

    For many datasets involving count data, this multiplicative model is reasonable and this happens to be the most popular link function.
  - `identity`: $g(\mu) = \mu$
    The link function is usually only useful when y is bounded away from 0 over the range of interest for the predictor variables as $X\beta$ can go negative, which isn't compatible with the Poisson (or any count distribution).

– `sqrt` (Square root): $g(\mu) = \sqrt{\mu}$
  This is an analog to the variance stabilizing transformation which has been used in the past. By the delta rule, if $\text{Var}(Y|X) = E[Y|X] = \mu(X)$,

$$\text{Var}(\sqrt{Y}|X) \approx \frac{1}{4}$$

Note that for the the `log` and `sqrt` link functions, additive models on the linear predictor scale lead to interactions on the mean scale.

For the `log` link,

$$\mu(x_1, x_2) = e^{\beta_0} e^{\beta_1 x_1} e^{\beta_2 x_2}$$

In this case the effect of x_1 depends on the level of x_2 (and vice-versa).

Similarly, for the `sqrt` link

$$\mu(x_1, x_2) = \beta_0^2 + 2\beta_0\beta_1 x_1 + 2\beta_0\beta_2 x_2 + +2\beta_1\beta_1 x_1 x_2 + \beta_1^2 x_1^2 + \beta_2^2 x_2^2$$

- Variance function: $V(\mu_i) = \mu_i$

As we have seen before, this data can easily analyzed by

```
> puffin.glm <- glm(nesting ~ grass + soil + angle + distance,
    data=puffin, family=poisson())

> summary(puffin.glm)

Call:
glm(formula = nesting ~ grass + soil + angle + distance,
    family = poisson(), data = puffin)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-2.3263  -1.2984  -0.6617   0.8119   2.5304
```

```
Coefficients:
             Estimate Std. Error z value Pr(>|z|)
(Intercept)  3.069973   0.452568   6.783 1.17e-11 ***
grass        0.005441   0.003104   1.753  0.07960 .
soil         0.033441   0.010822   3.090  0.00200 **
angle       -0.030077   0.010724  -2.805  0.00504 **
distance    -0.089399   0.010680  -8.371  < 2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for poisson family taken to be 1)

    Null deviance: 310.427  on 37  degrees of freedom
Residual deviance:  68.765  on 33  degrees of freedom
AIC: 183.38

Number of Fisher Scoring iterations: 6
```

```
> anova(puffin.glm)
Analysis of Deviance Table

Model: poisson, link: log

Response: nesting

Terms added sequentially (first to last)


         Df Deviance Resid. Df Resid. Dev
NULL                        37    310.427
grass     1    6.393         36    304.033
soil      1    0.033         35    304.000
angle     1  159.343         34    144.657
distance  1   75.892         33     68.765
```

Inference in Poisson Regression

As with logistic regression, inference can be based on Wald procedures for confidence intervals and tests on single βs and Likelihood Ratio drop in deviance tests on one or more parameters.

For example, an approximate confidence interval for β_j is

$$\hat{\beta}_j \pm z^*_{\alpha/2} SE(\hat{\beta}_j) = (L, U)$$

A confidence interval for e^{β_j} is given by

$$(e^L, e^U)$$

For example, for the effect of `distance`, 95% confidence intervals for β_4 and e^{β_4} are

```
> betahat <- coef(puffin.glm)[5]
> se <- sqrt(vcov(puffin.glm)[5,5])
> cibeta <- c(betahat-qnorm(0.975)*se, betahat+qnorm(0.975)*se)
> betahat
   distance
-0.08939925
> cibeta
   distance    distance
-0.11033098 -0.06846753
> exp(betahat)
 distance
0.9144804
> exp(cibeta)
 distance  distance
0.8955377 0.9338238
```

In addition, we can get confidence intervals for mean responses by a similar approach. The usual estimate of the mean response with predictor vector X is

$$\hat{\mu}(X) = g^{-1}(X\hat{\beta})$$

So for the log link, this is

$$\hat{\mu}(X) = e^{X\hat{\beta}}$$

A confidence interval for $g(\mu(X))$ is

$$X\hat{\beta} \pm z^*_{\alpha/2}\sqrt{X\widehat{\mathrm{Var}}(\hat{\beta})X^T} = (L, U)$$

which is then transformed

$$(g^{-1}(L), g^{-1}(U))$$

to give a confidence interval for $\mu(X)$.

The basic components can be calculated in **R** with the `predict` function, using the `type="link"` option.

For independent observations, the log-likelihood is

$$l(\beta) = \sum_{i=1}^{n} y_i \log \mu_i - \mu_i - \log y_i!$$

This gives a deviance for Poisson regression of

$$G^2 = 2 \sum_{i=1}^{n} y_i \log \frac{y_i}{\hat{\mu}_i} - y_i + \hat{\mu}_i$$

(This is set so the saturated model has a deviance of 0.)

Two models can be compared with the test statistic

$$X^2 = G^2(\text{Reduced Model}) - G^2(\text{Full Model})$$

which is compared to a χ^2_{df} distribution where df is the difference in the number of parameters between the 2 models.

For example, lets examine the effect of `grass` and `soil` on the number of nests

```
> anova(puffin2.glm, puffin.glm, test="Chisq")
Analysis of Deviance Table

Model 1: nesting ~ angle + distance
Model 2: nesting ~ grass + soil + angle + distance
  Resid. Df Resid. Dev Df Deviance P(>|Chi|)
1        35     85.050
2        33     68.765  2   16.285 0.0002909
```

As in Logistic regression, the Wald and Drop in Deviance tests aren't the same when a single β is examined.

```
> anova(puffin3.glm, puffin.glm, test="Chisq")
Analysis of Deviance Table

Model 1: nesting ~ soil + angle + distance
Model 2: nesting ~ grass + soil + angle + distance
  Resid. Df Resid. Dev Df Deviance P(>|Chi|)
1        34     71.888
2        33     68.765  1    3.123     0.077
> summary(puffin.glm)
Coefficients:
             Estimate Std. Error z value Pr(>|z|)
(Intercept)  3.069973   0.452568   6.783 1.17e-11 ***
grass        0.005441   0.003104   1.753  0.07960 .
soil         0.033441   0.010822   3.090  0.00200 **
angle       -0.030077   0.010724  -2.805  0.00504 **
distance    -0.089399   0.010680  -8.371  < 2e-16 ***
```

Model Diagnostics

To examine the fit of the model, similar approaches are taken as in logistic regression. As before, there are two types of residuals used

- Deviance residual

$$Dres_i = \text{sign}(Y_i - \hat{\mu}_i)\sqrt{2\left[y_i \log \frac{y_i}{\hat{\mu}_i} - y_i + \hat{\mu}_i\right]}$$

  Note that the form of residual changes as deviance residuals depend on the form of the log likelihood.

  As before these can be calculated in **R** by `resid(glm.object)`.

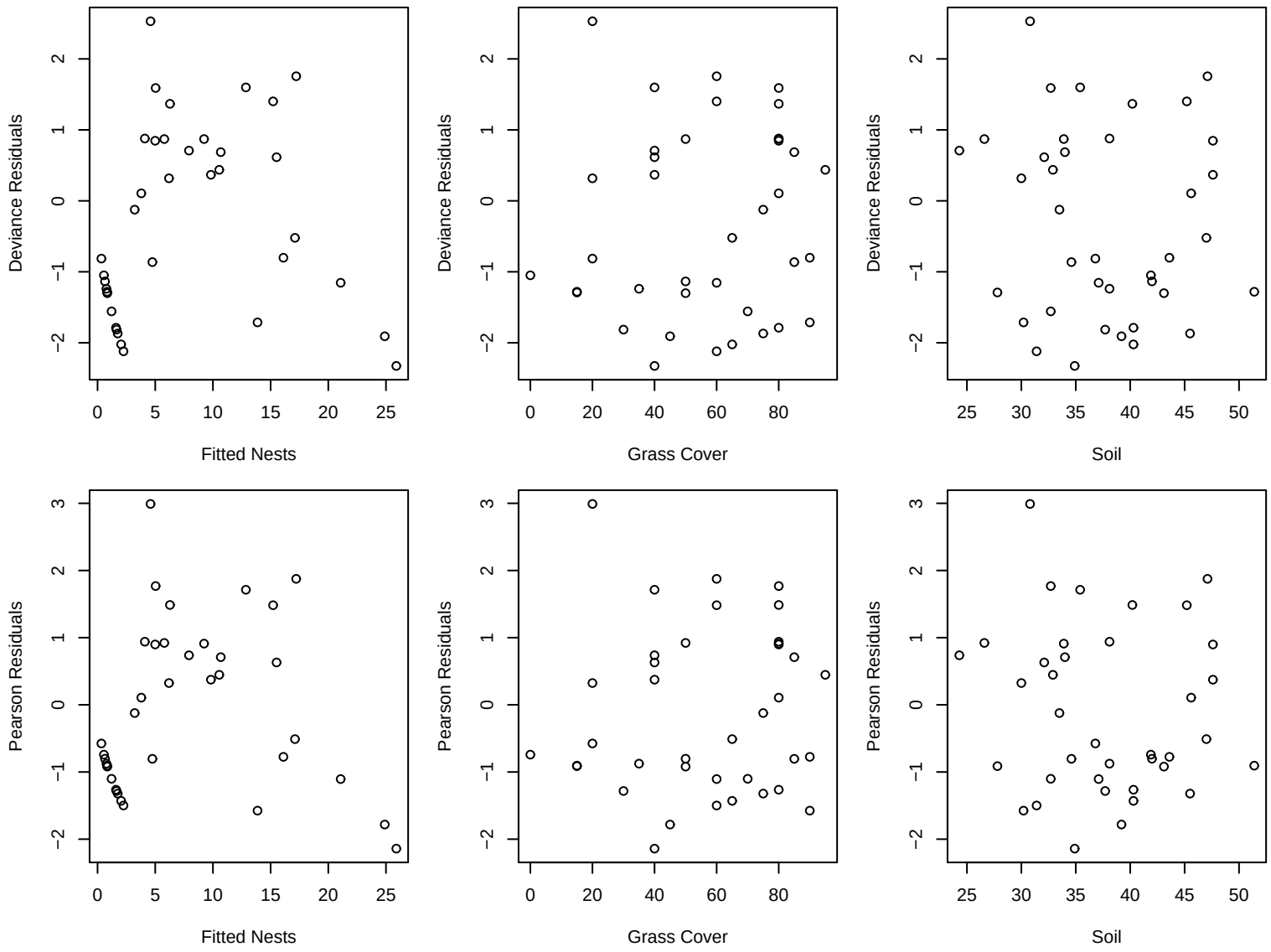
- Pearson residuals

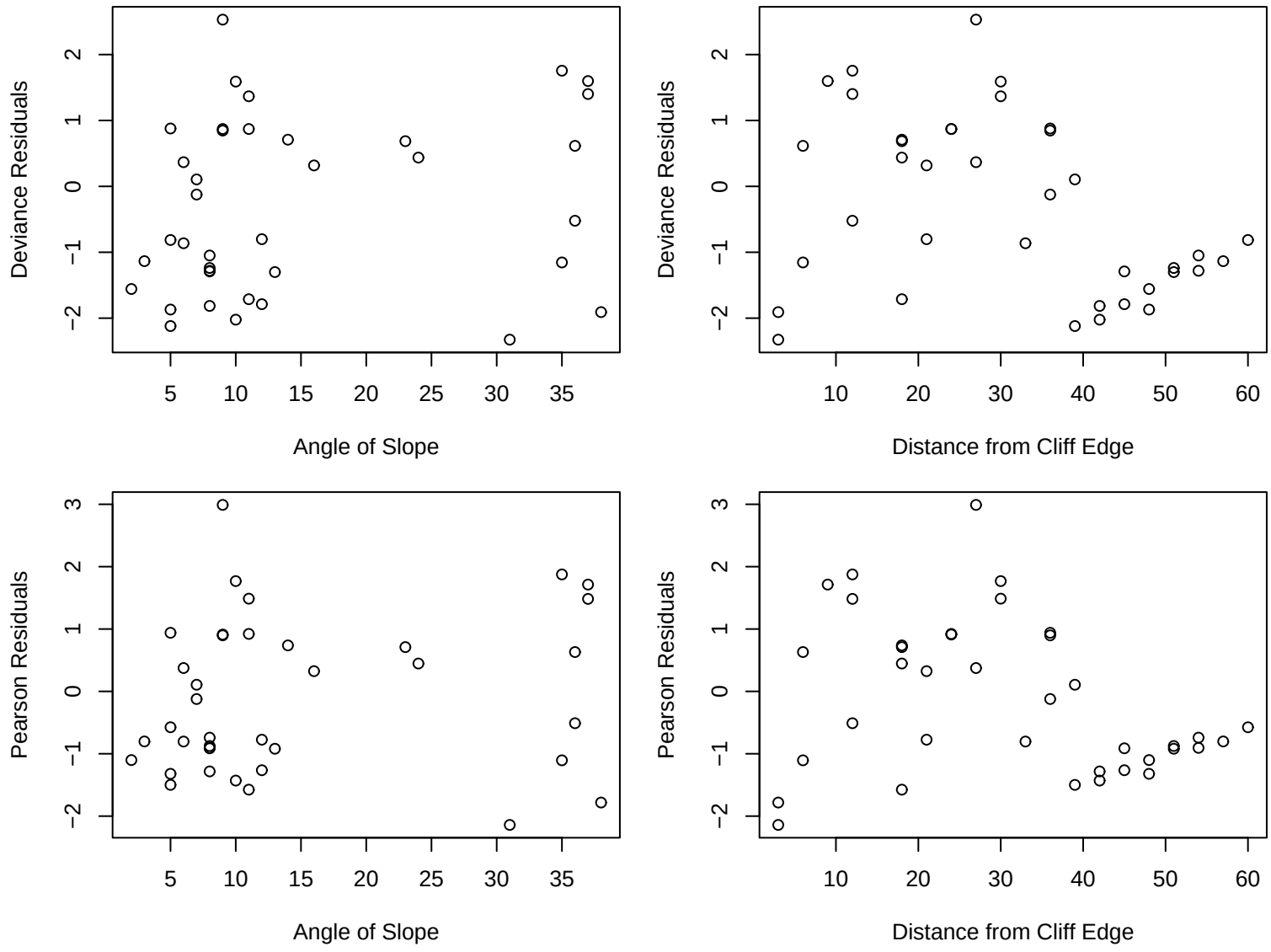
$$Pres_i = \frac{Y_i - \hat{\mu}_i}{\sqrt{\hat{\mu}_i}}$$

  This is of a similar form as seen before, except for the change in the form of $\mathrm{Var}(Y_i)$ that occurs in the denominator.

  Note that it ends up that these can be easily gotten in **R** by `resid(glm.object, type="pearson")`. This works for any generalized model, including logistic regression.

While usually not useful here, it is also possible to get the raw residuals $y_i - \hat{\mu}_i$ in **R** with the command `resid(glm.object, type="response")`.


Deviance Residuals
Fitted Nests
Deviance Residuals
Grass Cover
Deviance Residuals
Soil
Pearson Residuals
Fitted Nests
Pearson Residuals
Grass Cover
Pearson Residuals
Soil


Deviance Residuals
Angle of Slope
Deviance Residuals
Distance from Cliff Edge
Pearson Residuals
Angle of Slope
Pearson Residuals
Distance from Cliff Edge

These look pretty good. There is one observation that stands out, with a Pearson residual of 3. Interestingly, it doesn't stand out strongly in the scatterplot matrix

```
> cbind(puffin,pear.resid, dev.resid, fitted(puffin.glm))[24,]
   nesting grass soil angle distance pear.resid dev.resid
24      11    20 30.8     9       27   2.990383  2.530391
   fitted(puffin.glm)
24           4.591954
```

If the underlying μ_i are large, both the deviance and Pearson residuals are approximately $N(0,1)$ distributed. However if many of the $\hat{\mu}_i < 5$, the usual normal based cutoffs are questionable.

Not surprisingly, it is also possible to perform Goodness of Fit tests for Poisson data.

- Deviance Goodness of Fit test:

$$X^2 = 2\sum_{i=1}^{n} y_i \log \frac{y_i}{\hat{\mu}_i} - y_i + \hat{\mu}_i$$
$$= \sum_{i=1}^{n} Dres_i^2 = G^2$$

This is compared to a χ^2_{n-p} distribution.

The information for this test is given in the `summary` of the glm. For the example,

```
> summary(puffin.glm)

    Null deviance: 310.427  on 37  degrees of freedom
Residual deviance:  68.765  on 33  degrees of freedom
```

```
> pchisq(deviance(puffin.glm), df.residual(puffin.glm),
  lower.tail=F)
[1] 0.0002572599
```

- Pearson Goodness of Fit test:

$$X_p^2 = \sum_{i=1}^{n} \frac{(Y_i - \hat{\mu}_i)^2}{\hat{\mu}_i}$$
$$= \sum_{i=1}^{n} Pres_i^2$$

Similarly, it is also compared to a χ^2_{n-p} distribution.

```
> pchisq(sum(pear.resid^2), df.residual(puffin.glm),
  lower.tail=F)
[1] 0.006662672
```

Small p-values for these tests can be caused by many things, including

- Incorrect mean model, e.g. missing predictors
- Poisson model is wrong. For example, maybe $\mathrm{Var}(Y|X) > \mu(Y|X)$.
- Outliers contaminating the data

As with binomial data, these tests break down if there many observations with small Poisson means (e.g. $\hat{\mu}_i < 5$).

In cases like this, such as the example, other tools for examining the fit, such as testing extra terms (e.g. βx^2), should be used.

It is also possible to examine for influence using similar approaches as before. For this example, not much stands out, except for one observation that has large leverage $\left(> 3\sqrt{\frac{p}{n-p}}\right)$.

```
  dfb.1 dfb.gr dfb.so dfb.ang dfb.dst  dffit cov.r  cook.d   hat
5 0.068 -0.192  0.253 -0.2490 -0.1667  0.332  2.19 2.3e-02 0.478
```

```
> cbind(puffin,pear.resid, dev.resid, fitted(puffin.glm))[5,]
  nesting grass soil angle distance pear.resid dev.resid
5      11    40 47.6     6       27  0.3740736 0.3669813
  fitted(puffin.glm)
5           9.827332
```

Offsets

Often with count data, the expected counts will depend on an observation time, or an observation area. The idea being, if you observe for twice as long, you expect the count to double. So often the mean model will look like

$$\mu_i = t_i \times r_i$$

where t_i is observation time (or equivalent) and r_i is the rate (expected count per observation unit).

The log-linear model works well in this situation as

$$\begin{aligned}\log \mu_i &= \log(t_i r_i) \\ &= \log t_i + \log r_i \\ &= \log t_i + \beta_0 + \beta_1 x_{i1} + \ldots + \beta_p x_{ip}\end{aligned}$$

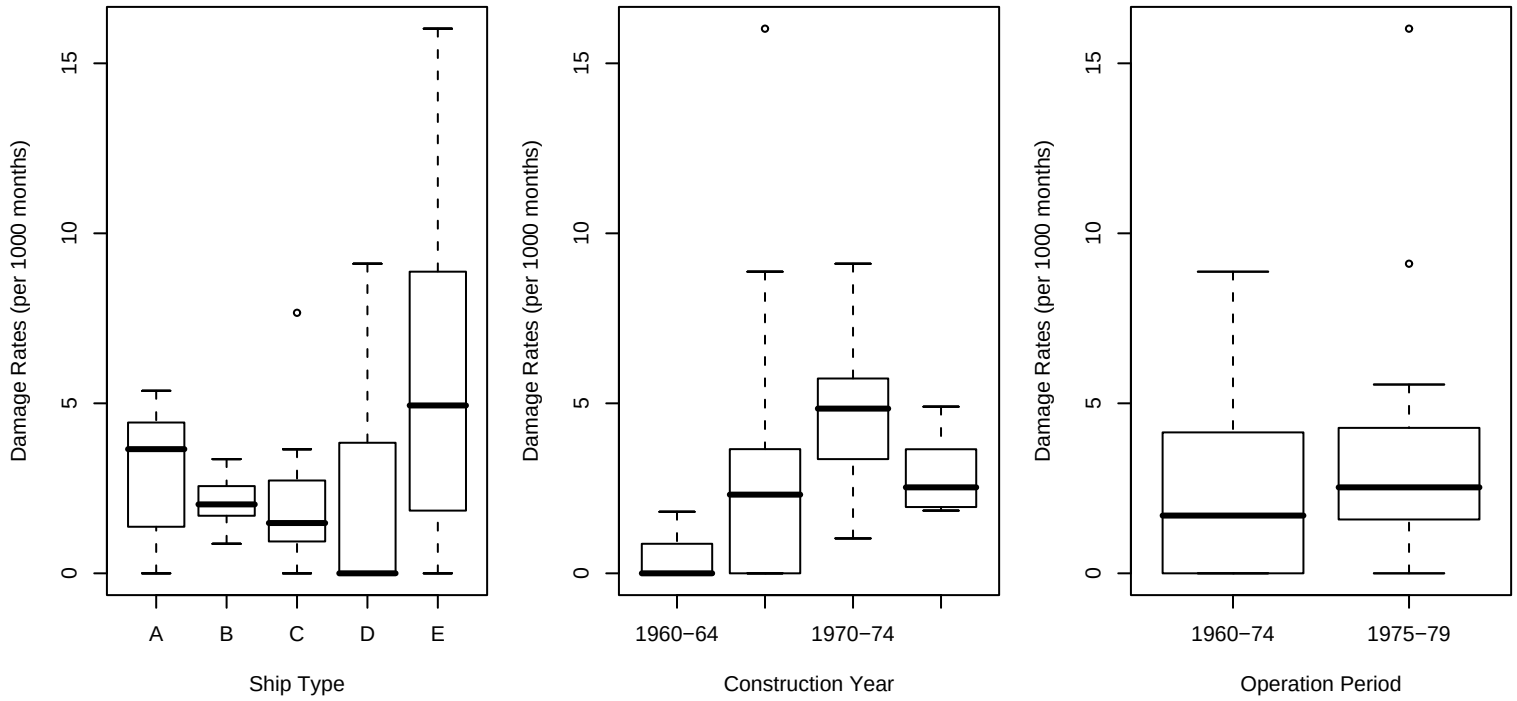
The quantity $\log t_i$ is often referred to as the offset.

Wave Damage to Cargo Ships: Data was collected by Lloyd's Register of Shipping investigating the damage caused by waves to the forward section of certain cargo-carrying vessels. Three factors are believed to affect the number of damage incidents

- Ship type: A - E
- Year of construction: 1960-64, 1965-69, 1970-74, 1975-1979
- Period of operation: 1960-74, 1975-1979

The observation times varied greatly (45 to 44882 months) and thus must be taken account of in the analysis. For example

Ship Type	Year of Construction	Period of Operation	Aggregate Service Time	Damage Damage	Damage Rate (per 1000 months)
B	1960-64	1960-74	44882	39	0.869
B	1960-64	1975-79	17176	29	1.688
B	1965-69	1960-74	28609	58	2.027
B	1965-69	1975-79	20370	53	2.602



Lets examining the model

$$\log \mu = \beta_0 + \beta_1 Type + \beta_2 Construct + \beta_3 Operation + \log Service$$

This can be done in **R** by using the `offset` option to `glm`.

```
> wave.glm <- glm(Damage ~ Type + Construct + Operation,
    offset=log(Service), data=wave2, family=poisson())

> summary(wave.glm)

Call:
glm(formula = Damage ~ Type + Construct + Operation,
    family = poisson(), data = wave2, offset = log(Service))

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-1.6768  -0.8293  -0.4370   0.5058   2.7912
```

```
Coefficients:
                 Estimate Std. Error z value Pr(>|z|)
(Intercept)      -6.40590    0.21744 -29.460  < 2e-16 ***
TypeB            -0.54334    0.17759  -3.060  0.00222 **
TypeC            -0.68740    0.32904  -2.089  0.03670 *
TypeD            -0.07596    0.29058  -0.261  0.79377
TypeE             0.32558    0.23588   1.380  0.16750
Construct1965-69  0.69714    0.14964   4.659 3.18e-06 ***
Construct1970-74  0.81843    0.16977   4.821 1.43e-06 ***
Construct1975-79  0.45343    0.23317   1.945  0.05182 .
Operation1975-79  0.38447    0.11827   3.251  0.00115 **
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for poisson family taken to be 1)

    Null deviance: 146.328  on 33  degrees of freedom
Residual deviance:  38.695  on 25  degrees of freedom
AIC: 154.56
```

In this example, all three factors seem to be important as

```
> anova(wave.t.glm, wave.glm, test="Chisq")
Analysis of Deviance Table

Model 1: Damage ~ Construct + Operation
Model 2: Damage ~ Type + Construct + Operation
  Resid. Df Resid. Dev Df Deviance P(>|Chi|)
1        29     62.365
2        25     38.695  4   23.670   9.3e-05

> anova(wave.c.glm, wave.glm, test="Chisq")
Analysis of Deviance Table

Model 1: Damage ~ Type + Operation
Model 2: Damage ~ Type + Construct + Operation
  Resid. Df Resid. Dev Df Deviance P(>|Chi|)
1        28     70.103
2        25     38.695  3   31.408 6.975e-07
```

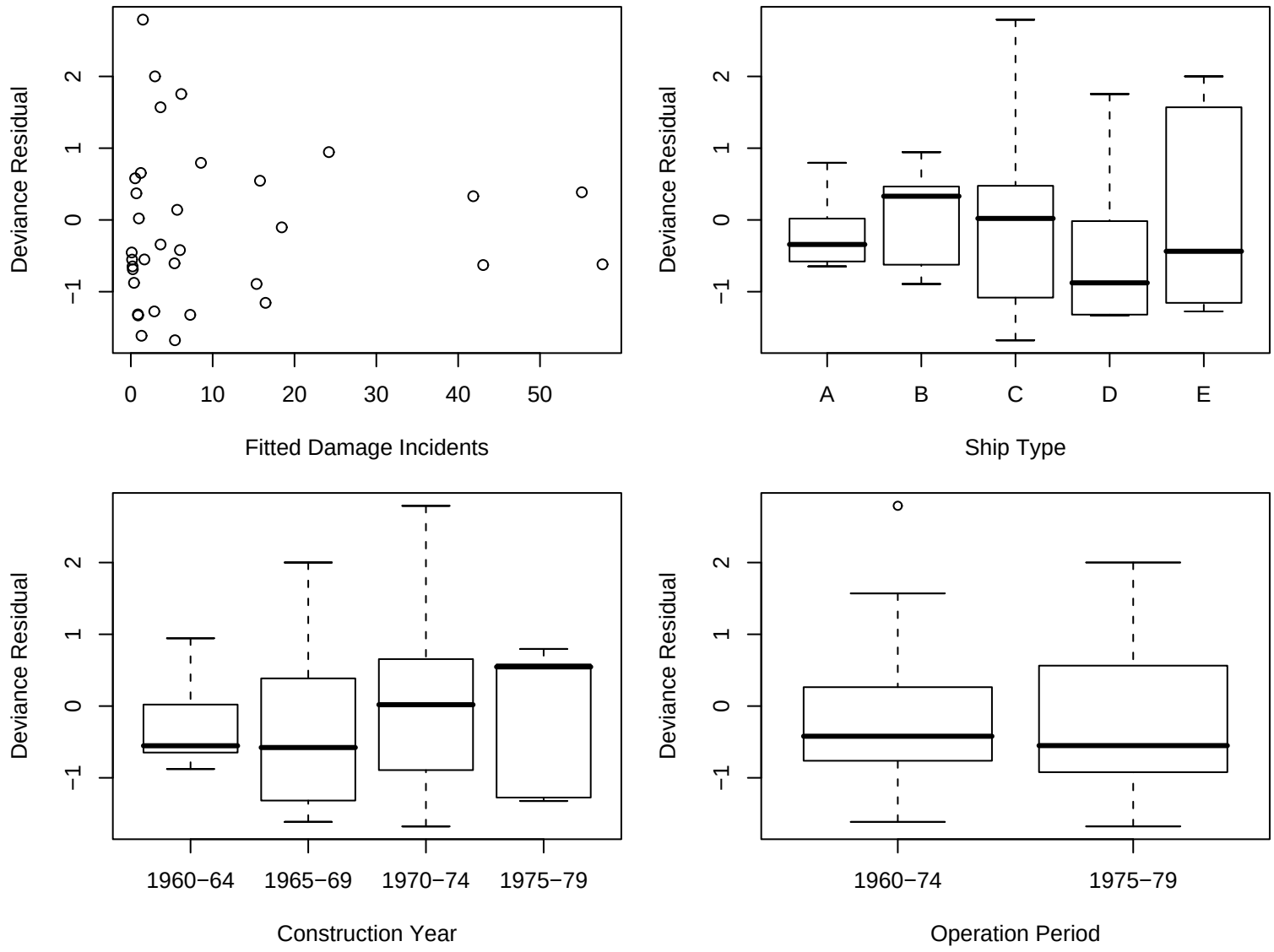
```
> anova(wave.o.glm, wave.glm, test="Chisq")
Analysis of Deviance Table

Model 1: Damage ~ Type + Construct
Model 2: Damage ~ Type + Construct + Operation
  Resid. Df Resid. Dev Df Deviance P(>|Chi|)
1        26     49.355
2        25     38.695  1   10.660     0.001
```

This should be taken with some grain of salt as it appears that there is a bit of a problem with the model.

First there are a couple of influential points as

```
> wave.dfbetas[dfbetamax > 1,]
   (Intercept)       TypeB        TypeC       TypeD       TypeE
22   0.1611307 -0.09398073 -1.573890166 -0.01157439  0.01292938
30  -0.0583331  0.01572651 -0.037365326  1.35894226 -0.05260388
32   0.0569754 -0.05843084 -0.054483664 -1.12959509 -0.06231752
38   0.1067719 -0.06268673  0.008085228 -0.02112444 -1.12917764
   Construct1965-69 Construct1970-74 Construct1975-79 Operation1975-79
22     0.0009727626      -0.21026708     -0.002521478      -0.15803220
30    -0.0247803096       0.17300816     -0.377954582       0.15951317
32     0.0034987641       0.08859403     -0.486453042       0.02191973
38     0.1116555777      -0.08841786      0.164985538      -0.28430484
```

More problematic is the residual plot


Deviance Residual
Fitted Damage Incidents
Ship Type
A
B
C
D
E
1960−64
1965−69
1970−74
1975−79
Construction Year
1960−74
1975−79
Operation Period