Chapter 23
Incorporating Temporal Correlation in Seal Abundance Data with MCMC

A.A. Saveliev, M. Cronin, A.F. Zuur, E.N. Ieno, N.J. Walker, and G.M. Smith

23.1 Introduction

Common or harbour seals (*Phoca vitulina* L.) are semi-aquatic mammals that spend time onshore at terrestrial sites where they haul-out to rest, breed, moult, engage in social activity, and escape predation (Fig. 23.1).

Over one-third of harbour seals in Ireland use haul-out sites in the southwest region (Cronin et al., 2007). Most of the haul-out sites in this region are located within Bantry Bay and the Kenmare River. Special Areas of Conservation (SACs) have been designated at each of these sites in accordance with the EU Habitats Directive. Assessing the year round changes in harbour seal abundance within SACs contributes to the monitoring obligations under the Habitats Directive and to the understanding of national population trends.

Between April 2003 and November 2005, regular standardised haul-out count surveys of both bays were carried out by boat. Counts of seals at each haul-out site were carried out independently and simultaneously by two observers, initially from a distance of approximately 200 m from the haul-out site and at progressively closer ranges whilst minimising disturbance to the seals. Surveys were carried out at least monthly all year-round and weekly during the summer and autumn, weather permitting. Surveys were scheduled to occur within two hours either side of low tide and during daylight hours.

In the original analysis of these data, Cronin (2007) used additive mixed modelling techniques. Residual auto-correlation structures and residual heterogeneity structures were included, albeit in the context of a Gaussian distribution. In this chapter, we do a similar statistical analysis, but replace the Gaussian distribution with a Poisson distribution because the response variable is a count (the number of seals). However, current software for generalised linear mixed modelling do not easily allow incorporating temporal correlation, and therefore, we use Markov

A.A. Saveliev (✉)
Kazan State University, Kazan, 420008, Russia

A.F. Zuur et al., *Mixed Effects Models and Extensions in Ecology with R*,
Statistics for Biology and Health, DOI 10.1007/978-0-387-87458-6_23,

Fig. 23.1 Mother and pup hauling out. The photo was taken by Michelle Cronin

Chain Monte Carlo (MCMC) techniques to fit a model that contains a temporal correlation structure.

MCMC is based on Bayesian statistics: Something not every reader will be familiar with. Providing an introduction to Bayesian statistics is challenging and including one in a 20-page case study is near impossible. We could have simply dropped this chapter, but considered it important to include a final chapter showing there is life beyond the mixed modelling techniques discussed in earlier chapters. So, although we cannot give a detailed introduction to Bayesian statistics or MCMC, we can discuss salient points and recommend you spend some time digging a bit deeper into this approach. A good starting point for ecologists is McCarthy (2007). More technical references are given later in this chapter.

So, the aim of this chapter is to show how you can include an auto-regressive correlation structure in a generalised linear model (GLM) for count data (e.g. using the Poisson or negative binomial distribution). Our choice of a GLM instead of a generalised additive model (GAM) is that we already have various GAM case study chapters and we wanted a better balance between parametric and non-parametric case studies. Also, using the R functions from Chapter 3, a smoothing spline can easily be programmed inside a GLM; so you could program the GAM equivalent of our MCMC approach (relatively) easily yourself. Further details and references on MCMC and GAM can be found in Keele (2008). Furthermore, we only focus on the correlation aspects of the model, not on the selection of the optimal model in terms of the explanatory variables. We therefore take the explanatory variables of the optimal model presented in Cronin (2007), and use these in the GLM.

23.2 Preliminary Results

As indicated in the introduction of this chapter, the same data were analysed in Cronin (2007) using an additive mixed modelling with a Gaussian distribution. The optimal model was of the form

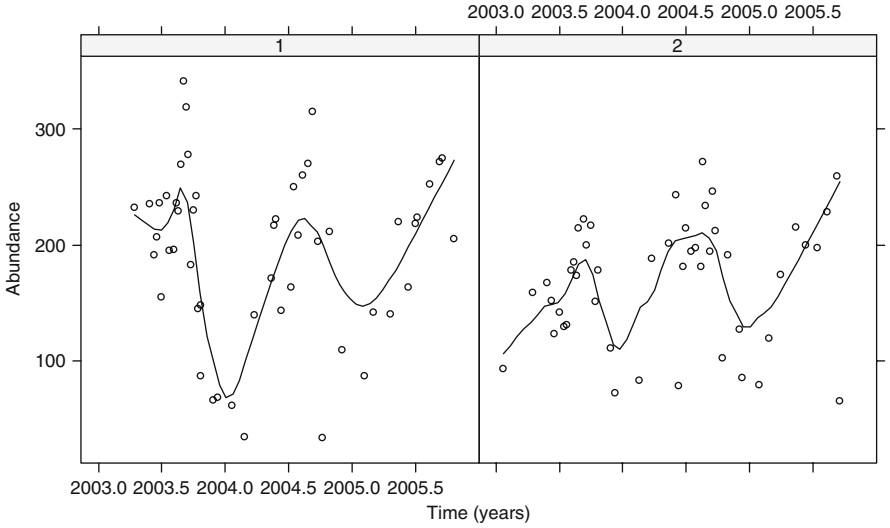

Fig. 23.2 Graph showing the observed abundances versus time (expressed as year + (week –1)/52, for both sites). To aid visual interpretation, a LOESS smoother was added. The haul-out sites are Bantry Bay (site 1) and Kenmare River (site 2). Note the seasonal pattern at both sites

$$A_i = f(\text{Month}_i, \text{TDay}_i) + \text{WindDir}_i + \text{Site}_i + \varepsilon_i \qquad \text{where } \varepsilon_i \sim N(0, \sigma^2_{\text{season}}) \tag{23.1}$$

A_i is the abundance of seals for observation number i. The explanatory variables month and time of day (expressed as TDay in the formula above) were fitted using a two-dimensional smoother $f(\text{Month}_i, \text{TDay}_i)$, allowing the day effect to change over the year. The model also contains two nominal variables: wind direction and site. Wind direction was categorised into 1 = North/Northeast, 2 = East/Southeast, 3 = South/Southwest, 4 = West/Northwest, and the haul-out sites are Bantry Bay (site 1) and Kenmare River (site 2). The residuals showed heterogeneity per season; so different residual variances per season (using the `varIdent` function from Chapter 4) were used.

Cronin (2007) also applied models with residual auto-regressive correlations of order 1 (using `corAR1`, see Chapter 6), but these were less optimal as judged by the AIC. The following R code reads the data and draws the time series plot shown in Fig. 23.2.

```
> library(AED); data(Seals)
> Seals$fSite <- factor(Seals$Site)
> Seals$Time <- Seals$Year + (Seals$Week - 1) / 52
> library(lattice)
> xyplot(Abun ~ Time | fSite, data = Seals,
     ylab = "Abundance", xlab = "Time (years)",
```

```
      panel = function(x, y){
      panel.loess(x, y, span = 0.3, col = 1)
      panel.xyplot(x, y, col = 1)})
```

We defined the variable `Time` as the year plus the week number (minus 1) divided by 52. Minus 1 ensures that an observation made in week 52 is still in the same year as an observation from week 1 or 51. The `xyplot` from the lattice package was used to make the figure, and it contains specific functions to add a LOESS smoother with a span of 0.3; higher values for the span resulted in smoothers that did not capture the seasonal pattern in the data.

Following Cronin (2007), we applied the GAM formulated in Equation (23.1), and the R code is given next. Instead of the long name `Timeofday`, we used a shorter notation, namely `TDay`. The dataset does not contain a variable Season; we created this variable using the following code:

```
> Seals$fSeason<-Seals$Month
> I1<-Seals$Month==1 | Seals$Month==2  | Seals$Month==12
> I2<-Seals$Month==3 | Seals$Month==4  | Seals$Month==5
> I3<-Seals$Month==6 | Seals$Month==7  | Seals$Month==8
> I4<-Seals$Month==9 | Seals$Month==10 | Seals$Month==11
> Seals$fSeason[I1] <- "a"
> Seals$fSeason[I2] <- "b"
> Seals$fSeason[I3] <- "c"
> Seals$fSeason[I4] <- "d"
> Seals$fSeason <- as.factor(fSeason)
```

The same was done for the wind direction:

```
> Seals$fWind2 <- Seals$Winddir
> Seals$fWind2[Seals$fWinddir==1|Seals$fWinddir==2]<-1
> Seals$fWind2[Seals$fWinddir==3|Seals$fWinddir==4]<-2
> Seals$fWind2[Seals$fWinddir==5|Seals$fWinddir==6]<-3
> Seals$fWind2[Seals$fWinddir==7|Seals$fWinddir==8]<-4
> Seals$fWind2 <- factor(Seals$fWind2)
```

Once we have this variable, the additive mixed model is run with the code below and also produces Fig. 23.3 and the numerical output.

```
> library(mgcv)
> Seals$TDay <- Seals$Timeofday
> fSeason2 <- Seals$fSeason #Avoids an error message
                            #from varIdent
> M1 <- gamm(Abun ~ s(Month,TDay) + fWind2 + fSite,
             weights = varIdent(form =~ 1 | fSeason2),
             data = Seals)
> plot(M1$gam, pers = TRUE)
> anova(M1$gam)
```

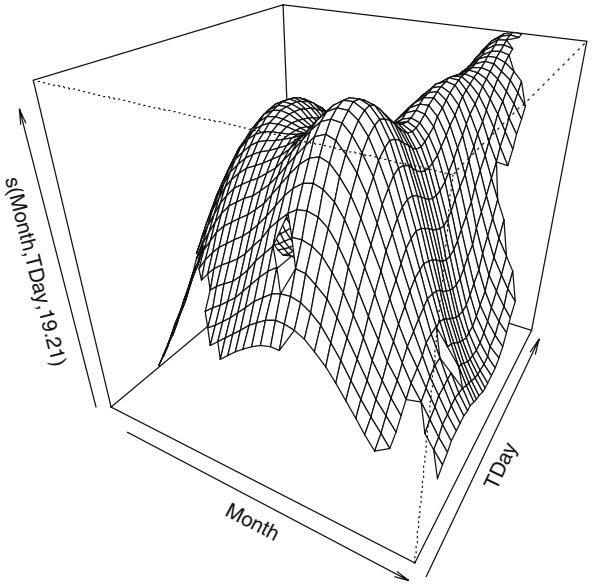

Fig. 23.3 Two-dimensional smoother for month and time of the day. The shape of the two-dimensional smoother indicates that an interaction is needed. A tunnel-type shape indicates that two additive main terms suffice

The output from the `anova` command is give below. There was a significant effect of month and time of day ($p < 0.001$), site effect ($p = 0.001$), and wind direction ($p = 0.005$) on the abundance of seals at haul-out sites in Bantry Bay and Kenmare River.

The `summary` command can be used to obtain the estimated parameters for each wind direction and site.

```
Parametric Terms:
             df      F p-value
Wind2         3  4.526 0.00573
fSite         1 10.537 0.00176
Approximate significance of smooth terms:
                edf Est.rank     F p-value
s(Month,TDay) 19.21    29.00 17.08  <2e-16
```

23.3 GLM

There are a few things that we would like to improve in Equation (23.1). First, we want to work with a distribution for count data, and the most obvious ones are the Poisson and negative binomial distributions. Recall from Chapter 9 that a GAM with a Poisson distribution is specified by the following three steps.

1. A_i is Poisson distributed with mean μ_i. In mathematical notation: $A_i \sim \mathrm{P}(\mu_i)$. As a result, we have $\mathrm{E}(Y_i) = \mu_i = \mathrm{var}(Y_i)$.
2. The systematic component is given by

$$\eta_i = f(\mathrm{Month}_i, \mathrm{TDay}_i) + \mathrm{WindDir}_i + \mathrm{Site}_i$$

3. The relationship between the mean μ_i and systematic component is specified by the logarithmic link function: $\log(\mu_i) = \eta_i$, which can also be written as $\mu_i = \exp(\eta_i)$.

In order to switch from a GAM to a GLM, we need to replace the $f(\mathrm{Month}_i, TDay_i)$ by something parametric. One option is to use

$$\eta_i = \mathrm{Month}_i + \mathrm{Month}_i^2 + \mathrm{TDay}_i + \mathrm{TDay}_i^2 + \mathrm{Month}_i \times \mathrm{TDay}_i + \mathrm{WindDir}_i + \mathrm{Site}_i$$

This predictor function uses quadratic functions for month and time of the day and an interaction between these main terms. Later, in the discussion for this chapter, alternatives are mentioned. The following code applies the GLM. The `scale` function applies a standardisation (subtract the means and divide by the standard deviation) and avoids collinearity between Month and Month^2 and also between TDay and TDay^2.

```
> Seals$Month1 <- as.double(scale(Seals$Month))
> Seals$Month2 <- Seals$Month1^2
> Seals$TDay1 <- as.double(scale(Seals$TDay))
> Seals$TDay2 <- Seals$TDay1^2
> M2 <- glm(Abun ~ Month1 + Month2 + TDay1 + TDay2 +
            Month1:TDay1 + fWind2 + fSite,
            data = Seals, family = poisson)
> summary(M2)

Coefficients:
              Estimate Std. Error z value  Pr(>|z|)
(Intercept)   5.567644   0.021865 254.640  < 2e-16
Month1       -0.016750   0.009541  -1.756  0.07916
Month2       -0.208159   0.008039 -25.894  < 2e-16
TDay1         0.012404   0.007811   1.588  0.11229
TDay2        -0.054142   0.007000  -7.734 1.04e-14
fWind22      -0.228104   0.027422  -8.318  < 2e-16
fWind23      -0.053383   0.023327  -2.288  0.02211
fWind24      -0.059348   0.021829  -2.719  0.00655
fSite2       -0.133858   0.015293  -8.753  < 2e-16
Month1:TDay1  0.108726   0.011211   9.698  < 2e-16

Dispersion parameter for poisson family taken to be 1
    Null deviance: 2502.7  on 97  degrees of freedom
Residual deviance: 1198.1  on 88  degrees of freedom
AIC: 1900.3
```

Note that there is overdispersion because the ratio of the residual deviance (1198.1) and residual degrees of freedom (88) is larger than 1! Instead of presenting a quasi-Poisson model, we will add an auto-correlation structure later in this chapter.

23.3.1 Validation

The hypothesis testing approach for the GLM also assumes independence of the residuals. Figure 23.4 shows a graph of the Pearson residuals plotted against time for both sites. Independence implies that we should not see any patterns in these panels, but we can, especially at site 1. Panel 2 also shows a general increasing trend: more negative residuals in the first year and more positive residuals in 2005. Instead of a subjective LOESS smoother, we can also use semi-variograms (or an auto-correlation function for regular spaced data) to test whether there is a significant temporal correlation, but we leave this as an exercise for the reader (see Chapters 6 and 7 for the R code). The following R code was used to create Fig. 23.4.

```
> Seals$E2 <- resid(M2, type = "pearson")
> xyplot(E2 ~ Time | fSite, data = Seals,
      ylab = "Pearson residuals", xlab = "Time (years)",
      panel = function(x, y){
      panel.loess(x, y, span = 0.5, col = 1)
      panel.xyplot(x, y, col = 1)})
```

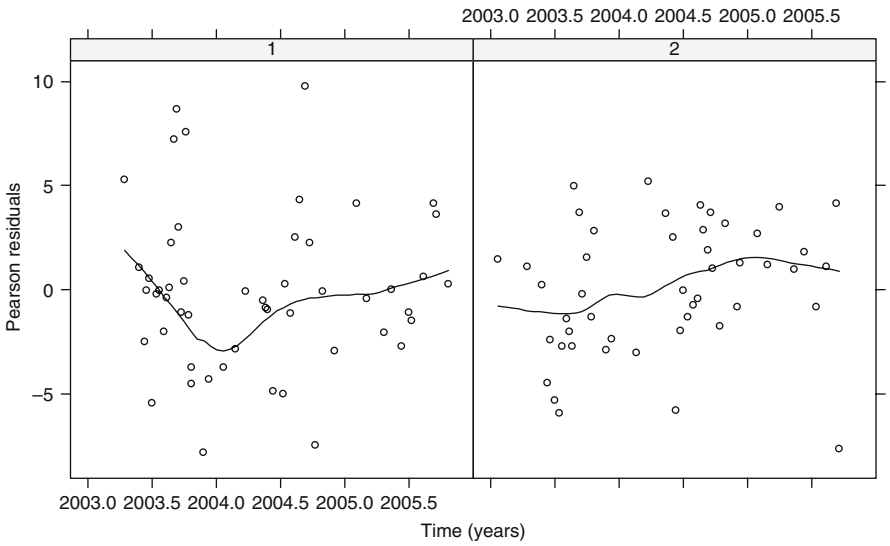

Fig. 23.4 Pearson residuals of the Poisson GLM plotted versus time for each site. The LOESS smoother indicates violation of independence over time

The first line extracts the Pearson residuals from the GLM object, and the rest is the familiar `xyplot` command with its options.

23.4 What Is Bayesian Statistics?

The methods discussed in previous chapters cannot easily be used to include a temporal correlation in a Poisson GLM, except perhaps for the `gamm` function, but it can cope with models that contain correlation structures and no random effects (e.g. random intercepts or slopes). However, as we said before, we want to do the analysis in a parametric context, and therefore, we now introduce techniques based on the Bayesian approach.

Before introducing Bayesian statistics, we will give a brief summary of the main characteristics of frequentist statistics, which is the approach used so far for analysing data. By this we mean that we have adopted a philosophy where we formulate a hypothesis for the regression parameters, apply the model in Equation (23.1) or its parametric equivalent, and estimate parameters, standard errors, 95% confidence intervals, and p-values. We can then say that if we were to repeat this experiment a large number of times, in 95% of cases, the real population regression parameters would lie inside the estimated confidence intervals. Furthermore, the p-values tell us how *often* (hence: frequency) we find an identical or larger test statistic. Key elements of frequentist statistics are as follows:

- The parameters (such as mean, variance, and regression parameters) that determine the behaviour of the population are fixed, but unknown.
- Based on observed data, these unknown parameters are then estimated in such a way that the observed data agree well with our statistical model; in other words, the parameter estimates are chosen such that the likelihood of the data is optimised (this is maximum likelihood estimation).
- Frequentist approaches are objective in that only the information contained in the current data set is used to estimate the parameters.

Bayesian statistics is based on a different philosophy. The main difference is assuming that the parameters driving the behaviour of the population are no longer fixed. Instead, it is assumed the parameters themselves follow some statistical distribution. To better explain this, we need to look at some theory.

23.4.1 Theory Behind Bayesian Statistics

The main components of Bayesian statistics are as follows.

Data. Suppose we have a stochastic variable Y with density function $f(Y \mid \boldsymbol{\theta})$, and let $\mathbf{y} = (y_1, \ldots, y_n)$ denote n observations of Y. Now $\boldsymbol{\theta}$ is a vector containing unknown parameters which are to be estimated from the observed data $\mathbf{y}$. In the case of the Poisson model described in Chapter 9, $\mathbf{y}$ would be the abundance observations and $\boldsymbol{\theta}$ the regression coefficient and overdispersion parameter.

Likelihood: The density $f(\mathbf{y} \mid \boldsymbol{\theta}) = \prod_i f(y_i \mid \boldsymbol{\theta})$ is the likelihood. When maximum likelihood estimation is carried out, $\boldsymbol{\theta}$ is chosen to maximise $f(\mathbf{y} \mid \boldsymbol{\theta})$. For example, in Chapter 9, we showed that for data following a Poisson distribution, the maximum likelihood estimates of the parameters are obtained from calculating the derivatives, setting those to zero, and solving the resulting equations.

Prior distribution: The major difference in Bayesian statistics is that instead of assuming that $\boldsymbol{\theta}$ is an unknown, but fixed, parameter vector, we now assume that $\boldsymbol{\theta}$ is stochastic. The distribution of $\boldsymbol{\theta}$ before the data are obtained, is called the prior distribution, and we denote it by $\pi(\boldsymbol{\theta})$. It reflects knowledge about $\boldsymbol{\theta}$, perhaps obtained from previous experiments, but it is also possible to choose $\pi(\boldsymbol{\theta})$ such that it reflects very little knowledge about $\boldsymbol{\theta}$ so that the posterior distribution is mostly influenced by the data. If the latter is the case, we say that the prior distribution is vague. (The term non-informative is also commonly used in reference to vague prior distributions.)

Posterior distribution: This forms the final component of a Bayesian analysis setting. Using some simple statistical theory (namely, Bayes' Theorem), the prior information is combined with information from the data to give us the posterior distribution $\pi(\boldsymbol{\theta} \mid \mathbf{y})$. It represents the information about $\boldsymbol{\theta}$ after observing the data $\mathbf{y}$:

$$\pi(\boldsymbol{\theta} \mid \mathbf{y}) = f(\mathbf{y} \mid \boldsymbol{\theta}) \times \pi(\boldsymbol{\theta}) \,/\, \pi(\mathbf{y}) \;\propto\; f(\mathbf{y} \mid \boldsymbol{\theta})\, \pi(\boldsymbol{\theta})$$

The latter follows because $\pi(\mathbf{y})$, which is the marginal density of the data, is constant. In contrast to maximum likelihood, where a point estimate for $\boldsymbol{\theta}$ is obtained, with Bayesian statistics a density of $\boldsymbol{\theta}$ is the final result. This density averages the prior information with information from the data. Gelman et al. (2003) provide an accessible book covering the basics of Bayesian analysis.

We have now discussed the three main components of Bayesian statistics: The prior distribution of $\boldsymbol{\theta}$, our observed data $\mathbf{y}$, and finally, how these two pieces of information are combined to obtain the posterior distribution of $\boldsymbol{\theta}$. In only a limited number of cases is the posterior distribution of a known form. More often than not, this distribution is complex, making it difficult to obtain summaries easily. For many years, this was the main reason why Bayesian statistics was not widely used. However, with the advent of computers, simulation tools such as Markov Chain Monte Carlo (MCMC) have become widely available and Bayesian analysis is more accessible. The development of freeware software implementing such simulation tools has also helped greatly to popularise Bayesian approaches.

23.4.2 Markov Chain Monte Carlo Techniques

The aim is to generate a sample from the posterior distribution. This is the Monte Carlo bit. Often, the exact form of the posterior distribution is unknown, but

fortunately another stochastic device, the Markov chain, can be used to deal with this. Assume we start with an initial value for $\boldsymbol{\theta}$, denoted by $\boldsymbol{\theta}_0$. Then the next state of the chain $\boldsymbol{\theta}_1$ is generated from $P(\boldsymbol{\theta}_1 \mid \boldsymbol{\theta}_0)$, where $P(. \mid .)$ is the so-called transition kernel of the chain. Then, $\boldsymbol{\theta}_2$ is generated from $P(\boldsymbol{\theta}_2 \mid \boldsymbol{\theta}_1)$, ... and $\boldsymbol{\theta}_t$ is generated from $P(\boldsymbol{\theta}_t \mid \boldsymbol{\theta}_{t-1})$. Under certain regularity conditions, the distribution of $P(\boldsymbol{\theta}_t \mid \boldsymbol{\theta}_0)$ will converge to a unique stationary distribution $\boldsymbol{\pi}(.)$. One important property of a Markov chain is that once it has reached its stationary distribution, it would have 'forgotten' about its initial starting value; so it no longer matters how inappropriate our initial value $\boldsymbol{\theta}_0$ was.

Assuming we can define an appropriate Markov chain (that is, an appropriate distribution $P(\boldsymbol{\theta}_t \mid \boldsymbol{\theta}_{t-1})$ can be constructed), we can then generate *dependent* draws (or realisations) from the posterior distribution. The samples are not independent as the distribution of $\boldsymbol{\theta}_t$ depends on the value of $\boldsymbol{\theta}_{t-1}$. In turn, the distribution of $\boldsymbol{\theta}_{t-1}$ depends on the value of $\boldsymbol{\theta}_{t-2}$ and so on. This has the following consequences:

- The initial part of the chain should be discarded (this initial part is commonly referred to as 'burn-in') so that the influence of an arbitrary initial value $\boldsymbol{\theta}_0$ is eliminated.
- The MCMC samples are less variable compared to independent samples, and therefore, the variance of estimated summary statistics, such as the sample mean, is larger than would be the case if the samples had been independent.
- When stationarity has been reached (that is, the realisations no longer depend on the initial value $\boldsymbol{\theta}_0$), a large number of samples is needed to cover the entire region of the posterior distribution as small portions of consecutive samples tend to be concentrated in small regions of the posterior distribution.

When we generate many samples after the burn-in samples have been discarded, these will then be distributed appropriately from the entire posterior distribution. This distribution can be summarised by summary statistics such as the sample mean and sample quantiles. A useful property of MCMC is that statistics calculated from the MCMC sample will converge to the corresponding posterior distribution quantities; for example, the sample mean converges to the posterior mean and the sample quantiles converge to the posterior quantiles.

It may seem complicated to generate samples from the posterior distribution, but fortunately there are algorithms available that make this task easy. The Metropolis-Hastings algorithm (Metropolis et al., 1953; Hastings, 1970) and the Gibbs sampler (Geman and Geman, 1984; Gelfand and Smith, 1990), which is a special case of the former, are two commonly used algorithms for creating appropriate Markov chains. Gilks et al. (1996) provide an accessible introduction to the various MCMC techniques illustrated with many examples. Although these algorithms are easily programmed in R, there are many technical complexities, and it is better to use specialised software, such as the freeware package WinBUGS (Lunn et al., 2000; www.mrc-bsu.cam.ac.uk/bugs/welcome.shtml). The R package `BRugs` is an interface to WinBUGS, and this is what we used for our seal abundance example.

23.5 Fitting the Poisson Model in BRugs

First, we have to recast the GLM model specification, given in Section 23.3, into a Bayesian framework. The abundance data are modelled as

$$\begin{aligned} A_i &\sim \text{Poisson}(\mu_i) \\ \eta_i &= \alpha + \beta_1 \times \text{Month}_i + \beta_2 \times \text{Month}_i^2 + \beta_3 \times \text{TDay}_i \\ &\quad + \beta_4 \times \text{TDay}_i^2 + \beta_5 \times \text{Month}_i \times \text{TDay}_i + \text{Winddir}_i + \text{Site}_i \\ \log(\mu_i) &= \eta_i \end{aligned} \tag{23.2}$$

This is the same as before. To make the model Bayesian, the unknown parameters are assigned prior distributions, as follows:

$$\begin{aligned} &\alpha \sim N(0, 10^6) \\ &\beta_1, \ldots, \beta_5 \sim N(0, 10^6) \\ &\text{Winddir, Site} \sim N(0, 10^6) \end{aligned} \tag{23.3}$$

The notation Windir, Site $\sim N(0, 10^6)$ means that we assume all the regression parameters for the individual levels of these nominal variables are normally distributed. *A priori*, we assume that the unknown parameters are zero on average, but with a large variance so that the parameters can take on values anywhere between -2000 and $+2000$. This reflects that we are rather 'vague' about what we believe about the parameters before seeing the data and is often referred to as a vague or non-informative prior distribution. To complete the Bayesian approach, we now have to obtain the posterior distribution and we will use MCMC to do this.

23.5.1 Code in R

Applying MCMC in R is relatively easy, although it takes more programming effort than the frequentist approach via the `glm` function, and it is also more time consuming in terms of computing.

First you have to install the packages `coda` and `BRugs` from the R website. To implement the Bayesian Poisson model, you need to create a couple of ASCII text files containing the model, data, and initial parameter estimates. Then the following code is run in R:

```
> library(coda)
> library(BRugs)
> modelCheck("Modelglm1.txt")
> modelData("Sealmatrix.txt")
> modelCompile(numChains = 3)
> modelInits("InitializeParam1.txt")
```

```
> modelInits("InitializeParam2.txt")
> modelInits("InitializeParam3.txt")
> #Burn in
> samplesStats("alpha")
> modelUpdate(200, thin = 50)
> plotHistory("alpha", colour = c(1, 1, 1))
> #Monitor model parameters
> dicSet()
> samplesSet("alpha")
> samplesSet("b")
> samplesSet("W")
> samplesSet("S")
> modelUpdate(10000, thin = 10)
> dicSet()
> samplesStats("alpha")
> samplesStats("b")
> samplesStats("W")
> samplesStats("S")
```

As you can see, this requires more code than the `glm` command in Chapter 9! And it also takes longer to run. Let us go over these commands in more detail. First of all, the file Sealmatrix.txt contains the data and is given on our website. The website also contains a small macro to prepare the data in the required format. The remaining components of the code are described in detail in the following sections.

23.5.2 Model Code

The file Modelglm1.txt forms the heart of the MCMC code, and contains the following lines.

```
model{
  for(i in 1:98) {
    Abun[i] ~ dpois(mu[i])
    log(mu[i]) <- alpha +
                  Month1[i] * b[1] + Month2[i] * b[2] +
                  TDay1[i] * b[3] +
                  TDay2[i] * b[4] +
                  Month1[i] * TDay1[i] * b[5] +
                  W[Wind2[i]] + S[Site[i]]
  }
  alpha ~ dnorm(0, 1.0E-6)
  for(j in 1:5) {
    b[j] ~ dnorm(0.0, 1.0E-6)
    }
```

```
  for (i in 2:4){
    W[i] ~ dnorm(0.0, 1.0E-6)
    }
  W[1] <- 0
  S[1] <- 0
  S[2] ~ dnorm(0.0, 1.0E-6)
  }
```

Assuming you have read all previous 22 chapters, this code should not appear too alien. The first block of code specifies that the abundance at site *i* is Poisson distributed with mean μ_i. The log link is used and the right hand side of the equation, following '`alpha +`', is simply the model in terms of the explanatory variables. The rest of the code, from `alpha ~ dnorm(0, 1.0E-6)` onwards specifies uninformative prior distributions for all parameters. The code including `W[1]` and `S[1]` ensures that the baseline levels of `W` (wind direction) and `S` (site) are nominal variables. It is important to note that the `dnorm` notation used in BUGS is different from the one used in R! The most important difference is that the variance is handed down to the `dnorm` function in terms of 1/variance. This is a Bayesian convention with 1/variance being the so-called precision of the distribution and is usually denoted by τ. So a variance of 10^6 is entered in the model framework as a precision of 10^{-6}. The reason for working with precision is that the posterior distribution of our parameters of interest will be a weighted combination of the prior distribution and the distribution of the data, with weights given by their respective precisions (i.e. 1/prior variance and 1/data variance). So, a large prior variance means that its precision is close to zero, and as a consequence, the prior distribution will receive almost no weight in deriving the posterior distribution of the parameters. This again reflects that our prior distribution is non-informative as it barely contributes to the posterior outcome. As a consequence, our posterior distribution should be similar to the one obtained from maximum likelihood estimation.

23.5.3 Initialising the Chains

Having formulated the model, we can now generate a sample from the posterior distribution using MCMC techniques. As described earlier, from a given starting value $\boldsymbol{\theta}_0$, the MCMC routine will generate values $\boldsymbol{\theta}_1$, $\boldsymbol{\theta}_2$, etc. When the chain is run for a sufficiently long period of time, it will have forgotten its initial starting value and from that point onward, the values drawn will represent samples from our posterior distribution of interest.

We used three chains, each of which is initialised with the `modelInits` command. The file InitializeParam1.txt contains the following text:

```
list(
  alpha=5.567643,
```

```
  b=c(-0.016750,-0.208159,0.012404,-0.054141,0.108725),
  S=c(NA, -0.133857),
  W=c(NA,-0.228104,-0.053383,-0.059347)
)
```

These are the estimated values from the ordinary GLM. The content of the file InitializeParam2.txt is not presented here, but it contains similar information; we took estimated values plus twice their standard errors from the ordinary GLM. Finally, the file InitializeParam3.txt contains the estimated values minus twice their standard errors (also from the GLM). It is also possible to add some random variation around these estimated values. Hence, the initial values are based on the maximum likelihood estimates $\pm$ 2 $\times$ standard errors. The NAs are needed for the baseline level of the factor.

The first `modelUpdate` statement in the main R code starts up the chains, and it states that every 50th realisation should be stored until 200 realisations have been kept in total per chain. The reason for not keeping all iterations is that the iterations will be auto-correlated, and therefore, do not provide much extra information on the posterior distribution. The command `plotHistory` provides a trace plot of these samples and is shown in Fig. 23.5 for one of the parameters: The intercept α. Note how the three chains start from different values and then gradually converge. The initial part, where the chains do not overlap, is the so-called burn-in period and reflects that the chains have not converged to the same stationary distribution yet. These samples will have to be discarded. Looking at Fig. 23.5 by eye, it seems that

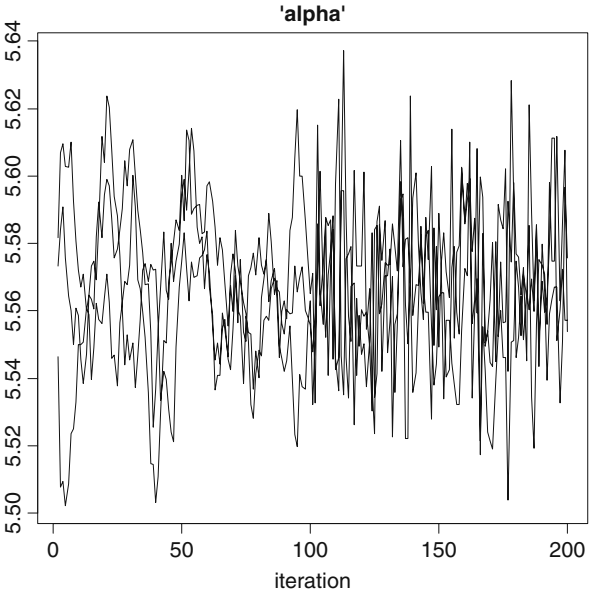

Fig. 23.5 Generated draws from the intercept α. If you run `plotHistory` with the default colours, you will see the colours mixed from different chains

a burn-in period of 150 samples (that is, $150 \times 50 = 7{,}500$ iterations) is enough, but we ran it for 10,000 samples to be sure.

More formal tests to assess the chain convergence are also available and are provided in the R package CODA (Plummer et al., 2008). One such test is the Gelman-Rubin statistic, which compares the variation between chains to the variation within a chain. Initially, the value of the Gelman-Rubin statistic will be large, but when convergence has been reached, it will have decreased to a value close to 1. Gelman (1996) suggests a value less than 1.1 or 1.2 is acceptable. This test should be applied to each of the model parameters. For our data, the Gelman-Rubin statistic was less than 1.2 for all parameters after 10,000 iterations, implying that from iteration 10,000 onwards, the draws come from the posterior distribution. Other tests (not shown here; full codes can be found at the book web site) reached similar conclusions.

23.5.4 Summarising the Posterior Distributions

Having decided on a burn-in of 10,000 iterations, we now want to formally keep the realised samples from iteration 10,000 onwards. This is achieved with the command `samplesSet`. The command `dicSet` allows for monitoring of the so-called Deviance Information Criterion, which can be used for model selection. Using the `modelUpdate` command, the chains are run for another 100,000 iterations. The R command `samplesStats` provides the following output.

```
> samplesStats("alpha")

      mean      sd    MC_error val2.5pc median val97.5pc start sample
alpha 5.568 0.02176  0.0001381    5.526  5.568      5.61   201  30000
```

The last two columns indicate when monitoring started (from the 201st sample onwards – recall that the first 200 stored samples were discarded as burn-in) and how many samples have been stored since monitoring started (10,000 samples for each of three chains). As a consequence, the summary statistics are based on 30,000 samples. To save the space, the last two columns have not been printed (start and number of samples) in the following text as they are the same as above

```
> samplesStats("b")

         mean       sd  MC_error  val2.5pc   median val97.5pc
b[1] -0.01670 0.009474 5.184e-05 -0.035220 -0.01668  0.001873
b[2] -0.20830 0.007988 4.446e-05 -0.224100 -0.20830 -0.192700
b[3]  0.01235 0.007782 4.579e-05 -0.002963  0.01238  0.027670
b[4] -0.05423 0.006975 3.980e-05 -0.067870 -0.05421 -0.040630
b[5]  0.10880 0.011200 6.361e-05  0.086900  0.10880  0.130800

> samplesStats("W")
```

```
         mean      sd  MC_error val2.5pc   median val97.5pc
W[2] -0.22850 0.02725 0.0001631 -0.28210 -0.22860 -0.175000
W[3] -0.05374 0.02324 0.0001417 -0.09905 -0.05379 -0.008206
W[4] -0.05954 0.02170 0.0001434 -0.10190 -0.05969 -0.016920

> samplesStats("S")

        mean      sd MC_error val2.5pc  median val97.5pc
S[2] -0.1339 0.01535 9.53e-05   -0.164 -0.1339   -0.1036
```

We also list the DIC statistics as an overall model fit indicator (this is similar to AIC and discussed in details at the end of chapter):

```
> dicStats()

      Dbar Dhat  DIC    pD
Abun  1890 1880 1900 9.948
total 1890 1880 1900 9.948
```

The samples from the posterior distribution are summarised by the mean, median and 2.5 and 97.5 percentiles. Note that the mean values are similar to those obtained by the `glm` command. The *standard deviation* of the posterior distribution, given under the header `sd`, is the Bayesian equivalent of the standard error of the mean (recall that the standard error of the mean is defined as the standard deviation of the mean values if the study were to repeated many times). Again, in this case, the values are similar to those obtained from `glm`.

23.5.5 Inference

The MCMC output contains thousands of realisations of all the model parameters, and these can be used to calculate various quantities of interest. For example, the correlation between the parameters can be obtained, and is shown in Table 23.1. High correlation is commonly expected for the constant and factor effects only. If

Table 23.1 Correlation between model parameters for the Poisson model

	alpha	b1	b2	b3	b4	b5	S2	W2	W3	W4
alpha	1	–0.01	–0.48	0.03	–0.32	0.03	–0.31	–0.50	–0.67	–0.71
b1		1	0.24	–0.03	0.00	0.12	0.02	–0.01	–0.17	–0.08
b2			1	0.00	0.14	0.13	–0.01	0.11	0.21	0.23
b3				1	–0.13	–0.15	–0.09	0.06	0.06	0.01
b4					1	0.03	0.07	–0.07	0.01	–0.10
b5						1	0.01	–0.10	–0.14	–0.06
S2							1	–0.08	–0.06	0.03
W2								1	0.50	0.52
W3									1	0.65
W4										1

regression coefficients associated with continuous variables (such as b[1], b[2], . . . , b[5]) show a high correlation, it is best to standardise these variables. This will reduce the correlation and will improve mixing of the MCMC chains so that consecutive realisations will be less dependent, shortening the burn-in period and the total number of iterations to be run (as instead of storing only every 20th iteration, for example, we can now keep every 5th iteration, for example).

We can also obtain Pearson residuals. The simplest way is to take the mean values from the MCMC samples and use these to calculate the Pearson residuals, but it is more informative to calculate the Pearson residuals for each MCMC realisation individually. In addition, we can also generate 'predicted' residuals for each MCMC realisation obtained from simulating abundance data from a Poisson distribution (Congdon, 2005). The latter will be properly Poisson distributed so will not display any overdispersion.

The BRugs model code (this is added to the code in the `modelglm1.txt` file presented earlier) is given below:

```
for(i in 1:N) {
      Aprd[i] ~ dpois(mu[i])
      e.obs[i] <- (Abun[i] - mu[i]) / sqrt(mu[i])
      p2.obs[i] <- e.obs[i] * e.obs[i]
      e.prd[i] <- (Aprd[i] - mu[i]) / sqrt(mu[i])
      p2.prd[i] <- e.prd[i] * e.prd[i]
  }
SS <-sum(p2.obs[1:N])
SS.prd <- sum(p2.prd[1:N])
```

In the absence of overdispersion, the sum of squares will follow a χ^2(N) distribution. The summary statistics of `SS` and `SS.prd` are

```
          mean    sd MC_error val2.5pc median val97.5pc
SS    1177    12.02  0.06561      1157   1176      1203
SS.prd  97.97 14.08  0.08001     72.47  97.23     127.7
```

For comparison, the true 2.5 and 97.5% percentiles of the χ^2(N) distribution are

```
> qchisq(c(0.025, 0.975), 98)
[1]  72.50094 127.28207
```

Note that the percentiles of the simulated `SS.prd` values correspond well with the theoretical percentiles. There are two ways of assessing overdispersion. The first one is to compare the distribution of `SS` to the distribution of the predicted SS in the absence of overdispersion `(SS.prd)`. Clearly, the two distributions do not match, indicating substantial overdispersion. The second approach is to compare the SS distribution to the $\chi^2(98)$ distribution, which gives exactly the same information. The average SS, 1177, is similar to what was observed from the `glm` fit.

Another quantity of interest is the auto-correlation in the abundance data. For each realisation of the MCMC chain, we can estimate this in R as follows (Box-Pierce auto-correlation test with Ljung-Box modification, Congdon, 2005):

```
> for(k in 1:3) {
      for (t in k+1 : N) {
        p1[k,t] <- e.obs[t] * e.obs[t - k]
      }
     auto [1, k] <- sum(p1[k, (N11 + k) : N12]) /
                               sum(p2.obs[N11: N12])
     auto2[1, k] <- auto[1,k] * auto[1,k]
     auto [2, k] <- sum(p1[k, (N21 + k) : N22]) /
                                sum(p2.obs[N21 : N22])
     auto2[2, k] <- auto[2, k] * auto[2, k]
  }
```

This code calculates the auto-correlation up to lag 3 based on the observed residuals for each of the two sites. The calculation is somewhat simplistic in that it assumes that the time series are regular, where in practice the data were collected at irregular one to two week intervals, but at this stage we are only exploring the possibility of auto-correlation. If the auto-correlation is zero, the value `BP.s1` and `BP.s1` below are distributed approximately as χ^2(k), where $k = 3$ is number of terms in sum

```
> BP.s1 <- (N12 - N11 - 1) * sum(auto2[1, ])
> BP.s2 <- (N22 - N21 - 1) * sum(auto2[2, ])
```

The summary statistics are as follows:

```
       mean     sd  MC_error val2.5pc median val97.5pc
BP.s1 9.574 0.6612 0.004345    8.377   9.54     10.96
BP.s2 3.687 0.5958 0.003407    2.616  3.653      4.94

> qchisq(c(0.025, 0.975), 3)

[1] 0.2157953 9.3484036
```

In the absence of auto-correlation, the BP statistic follows a $\chi^2(3)$ distribution. Clearly, the mean and 2.5 and 97.5 percentiles of `BP.s1` are well in excess of those of the $\chi^2(3)$ distribution, providing strong evidence of auto-correlation at this site. For site 2, the auto-correlation is not significant, as the 2.5–97.5 percentile range of `BP.s2` is covered by the corresponding range of the $\chi^2(3)$ statistic. It should be mentioned that this criterion is approximate.

23.6 Poisson Model with Random Effects

The Poisson model can be extended into a GLMM by adding a random term ε_i to the linear predictor:

$$\begin{aligned} A_i &\sim \text{Poisson}(\mu_i) \\ \log(\mu_i) &= \eta_i + \varepsilon_i \end{aligned} \tag{23.4}$$

where η_i is defined as before, see Equation (23.2), and it is assumed that

$$\varepsilon_i \sim N(0, 1/\tau_\varepsilon)$$

The BRugs model code for the abundance data now becomes

```
for(i in 1:98) {
   Abun[i] ~ dpois(mu[i])
   log(mu[i]) <- alpha +
                 Month1[i] * b[1] + Month2[i] * b[2] +
                 TDday1[i] * b[3] + TDay2[i] * b[4] +
                 Month1[i] * TDay1[i] * b[5] +
                 W[Wind2[i]] + S[Site[i]] + eps[i]
   eps[i] ~ dnorm(0.0, eps.tau)
}
```

Furthermore, in addition to the prior distributions on `alpha`, `b`, wind effect `W`, and site effect `S`, we now also need a prior distribution on τ_ε. A common choice is to assume that $\tau_\varepsilon \sim$ Gamma(0.001,0.001), which reflects vague prior information on τ_ε:

```
eps.tau ~ dgamma(0.001, 0.001)
```

The complete BRugs code can be downloaded from the book website. To investigate overdispersion, the sum of squares `SS` was monitored:

```
    mean    sd MC_error val2.5pc median val97.5pc
SS 100.5  14.2  0.07761    74.68  99.85     130.4
```

The observed range of SS values corresponds to the 2.5 and 97.5 percentiles (72.5 and 128.3, respectively) of the $\chi^2(98)$ statistic, indicating that overdispersion is no longer present. The DIC statistics have also improved:

```
       Dbar Dhat DIC    pD
Abun    783  691 875 92.02
total   783  691 875 92.02
```

The MCMC output contains realisations for ε_i, $i = 1 \ldots 98$, so that we can look at summary statistics of these for each observation i. To illustrate, for each i the mean and standard deviation of the realisations for ε_i was calculated, and then summarised in a histogram (Fig. 23.6).

Furthermore, the pooled variance of the ε_i was also calculated; it was 0.085. This agrees well with the sampled values for `eps.sigma = 1 / eps.tau`:

```
              mean      sd   MC_error val2.5pc  median val97.5pc
eps.sigma 0.08707 0.01491 9.457e-05  0.06213 0.08555    0.1205
```

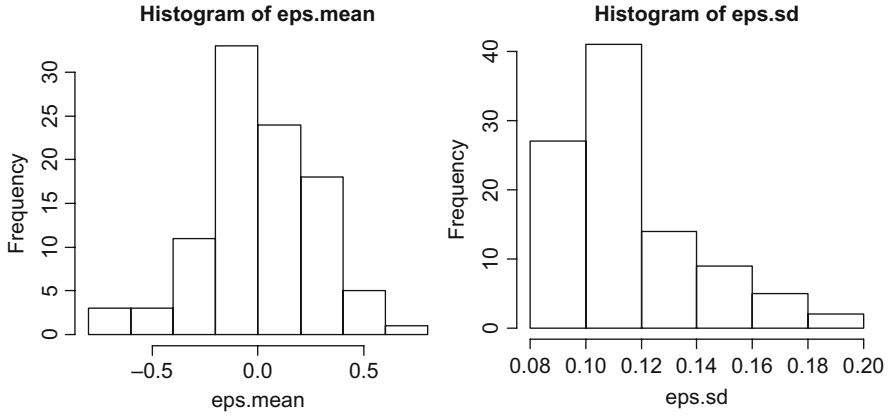

Fig. 23.6 Poisson model with random effects. For each observation i, the MCMC realisations of the residuals at the linear predictor level (ε_i, $i = 1 \ldots 98$) are summarised by their mean and standard deviation. The 98 mean values and standard deviations are then plotted as a histogram

The auto-correlation summary statistics (at abundance level) are

```
       mean    sd MC_error val2.5pc median val97.5pc
BP.s1 2.724 2.28   0.01276   0.2018  2.147     8.616
BP.s2 2.635 2.185  0.01328   0.1985  2.073     8.233
```

and correspond to the $\chi^2(3)$ distribution (2.5 and 97.5 percentiles given by 0.22 and 9.35, respectively). This suggests there is no evidence of significant auto-correlation

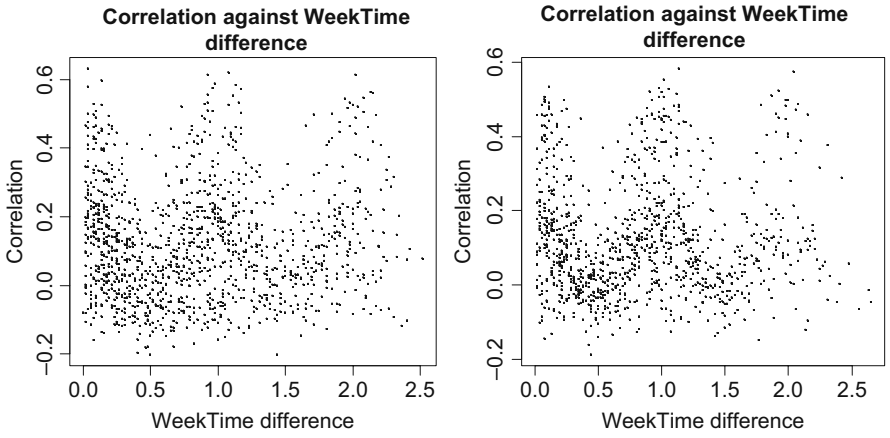

Fig. 23.7 Correlation in the random effects ε_i at the linear predictor level, calculated from MCMC results of the Poisson model with random effect, versus the lag (in weeks) between time points. The *left graph* shows the results for site 1 and the *right graph* for site 2

in the abundance data. Despite this, the correlation between the random effects, shown in Fig. 23.7, does exhibit strong pattern in time and therefore, in the next section, we investigate this further by extending the random effects model by including auto-correlation in the random effect.

23.7 Poisson Model with Random Effects and Auto-correlation

We now introduce auto-correlation into the model that allows the auto-correlation to differ between the two sites. First, we introduce a slightly different notation using subscript s for site s:

$$\begin{aligned} A_{si} &\sim \text{Poisson}(\mu_{si}) \\ \eta_{si} &= \alpha + \beta_1 \times \text{Month}_{si} + \beta_2 \times \text{Month}_{si}^2 + \beta_3 \times \text{TDay}_{si} \\ &\quad + \beta_4 \times \text{TDay}_{si}^2 + \beta_5 \times \text{Month}_{si} \times \text{TDay}_{si} + \text{Winddir}_{si} + \text{Site}_{si} \\ \log(\mu_{si}) &= \eta_{si} + \varepsilon_{si} \end{aligned} \tag{23.5}$$

and ε_{si} is a random effect. So far, this is the same model formulation as in Section 23.6. In addition, it is now assumed that ε_{si} follows an auto-regressive structure:

$$\varepsilon_{si} = \rho_s^{|W_{\text{dist}}|} \times \varepsilon_{s,i-1} + u_{si}$$

where W_{dist} is the time lag in weeks between data points $i-1$ and i. It is assumed that the u_{si} are independently and normally distributed:

$$u_{si} \sim \text{N}(0, 1/\tau_u)$$

The last part of Equation (23.5) can be rewritten into

$$\begin{aligned} \log(\mu_{si}) &= \eta_{si} + \rho_s^{|W_{\text{dist}}|} \times \varepsilon_{s,i-1} + u_{si} \\ &= \eta_{si} + \rho_s^{|W_{\text{dist}}|} \times \log(\mu_{s,i-1}) - \rho_s^{|W_{\text{dist}}|} \times \eta_{s,i-1} + u_{si} \end{aligned}$$

The full model then becomes

$$\begin{aligned} A_{si} &\sim \text{Poisson}(\mu_{si}) \\ \log(\mu_{si}) &= \eta_{si} + \rho_s^{|W_{\text{dist}}|} \times \log(\mu_{s,i-1}) - \rho_s^{|W_{\text{dist}}|} \times \eta_{s,i-1} + u_{si} \\ u_{si} &\sim N(0, 1/\tau_u) \end{aligned} \tag{23.6}$$

where η_{si} is given by Equation (23.5). Assigning prior distributions to the unknown parameters completes the Bayesian model formulation:

- $\alpha \sim \text{N}(0, 10^6)$
- $\beta_j \sim \text{N}(0, 10^6)$, for $j = 1, \ldots, 5$
- $\tau_{\text{u}} \sim \text{Gamma}(0.001, 0.001)$

- $\rho_1 \sim$ Uniform[–1, 1]
- $\rho_2 \sim$ Uniform[–1, 1]

The full model code can be downloaded from our website. The summary statistics for the posterior distributions of the model parameters are as follows.

Parameter	mean	sd	2.5%	97.5%
Alpha	5.554	0.1145	5.338	5.786
b[1]	−0.07878	0.04421	−0.1661	0.006503
b[2]	−0.2238	0.0347	−0.2933	−0.1561
b[3]	0.01604	0.02676	−0.03674	0.06891
b[4]	−0.07792	0.02602	−0.1287	−0.02693
b[5]	0.1102	0.03435	0.04245	0.1776
S[2]	−0.0702	0.1018	−0.2838	0.1186
W[2]	−0.3426	0.0914	−0.5226	−0.163
W[3]	−0.0949	0.088	−0.2682	0.08002
W[4]	−0.05306	0.07948	−0.2089	0.1023
rho1	0.6395	0.1746	0.2318	0.9199
rho2	0.3545	0.3181	−0.3075	0.8311
u.sigma	0.07798	0.01372	0.05509	0.1087

The DIC statistic is similar to the previous

```
       Dbar  Dhat   DIC    pD
Abun  784.5 693.3 875.6 91.13
total 784.5 693.3 875.6 91.13
```

There is no indication of auto-correlation at the abundance level:

```
       mean    sd MC_error val2.5pc median val97.5pc
BP.s1 2.732 2.24   0.01306   0.2005   2.16      8.526
BP.s2 2.662 2.198  0.01146   0.1916   2.096     8.332
```

The random effects auto-correlation at site 1 is significant (zero is not contained in the 2.5–97.5 percentile interval for rho1), but is not significant for site 2:

```
       mean      sd MC_error val2.5pc median val97.5pc
rho1 0.6395 0.1746 0.002414   0.2318 0.6616    0.9199
rho2 0.3545 0.3181 0.003398  -0.3075 0.4126    0.8311
```

Furthermore, there is no sign of overdispersion as the SS and SSpred intervals are similar.

```
        mean    sd MC_error val2.5pc median val97.5pc
SS     101.8  14.37  0.08356    75.45  101.3     131.9
SS.prd 97.96 14.07  0.07975    72.43  97.34     127.6
```

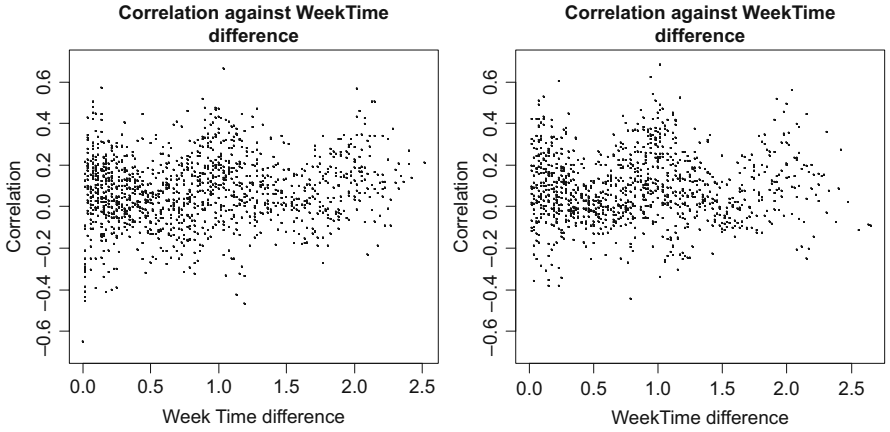

Fig. 23.8 Auto-correlation in the residuals u_{si} based on Poisson model with auto-correlated random effects (all auto-correlations over all time lags combined). The *left graph* shows the results for site 1, and the *right graph* for site 2

Finally, the auto-correlation in the residuals u_{si} was calculated and is shown in Fig. 23.8. Although the model assumes that the u_{si} are independent, the figure reveals that some auto-correlation pattern is still present (peaks at zero and multiples of year), but less severe compared to the previous model.

23.8 Negative Binomial Distribution with Auto-correlated Random Effects

Coming back to the overdispersion issue, we now use the negative binomial distribution (Chapter 9) instead of the Poisson distribution to model the overdispersion with the size parameter *size*. The variance of the abundances is given by $\mu + \mu^2/size$. To incorporate this into the Bayesian model structure, all we need to do is to change the Poisson distribution in Equation (23.6) to the negative binomial distribution:

$$A_i \sim NB(\mu_i, size) \tag{23.7}$$

As we have now introduced a new parameter, *size*, a distribution needs to be defined:

$$size \sim \text{Gamma}(0.001, 0.001)$$

This is a common choice reflecting vague prior information. The full model code can be found on our website. The posterior distributions are summarised as follows:

Parameter	mean	Sd	2.5%	97.5%
alpha	5.589	0.1055	5.39	5.81
b[1]	−0.05093	0.04148	−0.1373	0.02789
b[2]	−0.2188	0.03242	−0.2814	−0.1551
b[3]	0.02245	0.03231	−0.03867	0.08694
b[4]	−0.06466	0.03021	−0.1247	−0.005743
b[5]	0.1105	0.03911	0.03573	0.1878
S[2]	−0.1162	0.09203	−0.3104	0.04696
W[2]	−0.2827	0.1084	−0.4936	−0.06992
W[3]	−0.06669	0.0963	−0.2521	0.1244
W[4]	−0.05785	0.09231	−0.2347	0.1284
rho1	0.4544	0.4433	−0.5042	0.9796
rho2	0.2152	0.539	−0.8189	0.9669
u.sigma	0.01062	0.01133	0.0006066	0.04199
size	13.77	5.533	9.405	28.53

and the DIC statistic for the model is much larger:

```
       Dbar  Dhat   DIC    pD
Abun   1032  1009  1055 22.78
total  1032  1009  1055 22.78
```

The variance parameter 1/*size*, which allows for modelling of overdispersion, is significantly different from zero (average $size = 13.77$ against the μ of order

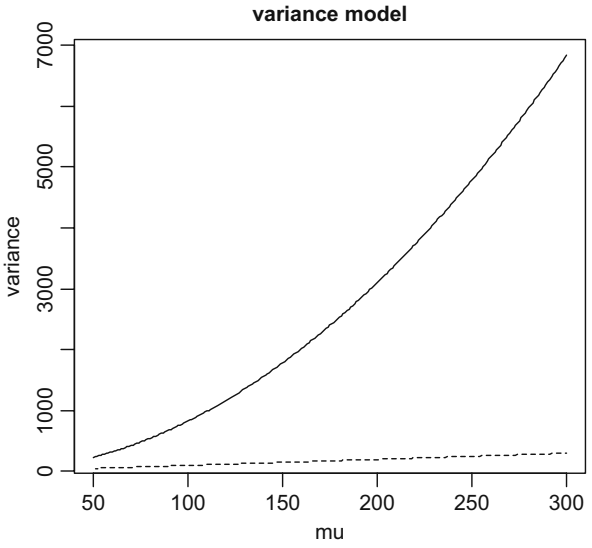

Fig. 23.9 Comparison of variance in abundance for Poisson model (*dashed line*) and negative binomial model (*solid line*)

100). The variance of the abundances is given by $\mu + \mu^2/size$, compared to μ for the Poisson model. Figure 23.9 compares the two variances and clearly shows the overdispersion for high abundance values.

As before, some evidence of auto-correlation in abundance levels is still expressed at both sites, even though the mean values almost certainly lie in the posterior confidence interval.

```
       mean    sd MC_error val2.5pc median val97.5pc
BP.s1 5.941 3.414   0.1355   0.6958  5.599     13.90
BP.s2 3.822 3.185   0.1080   0.2517  2.989     12.16
```

Residual test shows rather some signs of underdispersion:

```
         mean    sd MC_error val2.5pc median val97.5pc
SS      84.73 13.37   0.5135       64  82.77     116.2
SS.prd 97.96 14.07  0.07975    72.43  97.34     127.6
```

The auto-correlation in the residuals u_{si} is shown in Fig. 23.10. The increased auto-correlation at lag 0 and at lag multiples of year, as observed previously, has now disappeared. It is worth mentioning the significant decrease in the u_{si} variance against the previous model.

For both sites, the error auto-correlation parameters are now no longer significant:

```
       mean    sd MC_error val2.5pc median val97.5pc
rho1 0.4544 0.443  0.02364  -0.5042 0.5832    0.9796
rho2 0.2152 0.539  0.01594  -0.8189 0.2956    0.9669
```

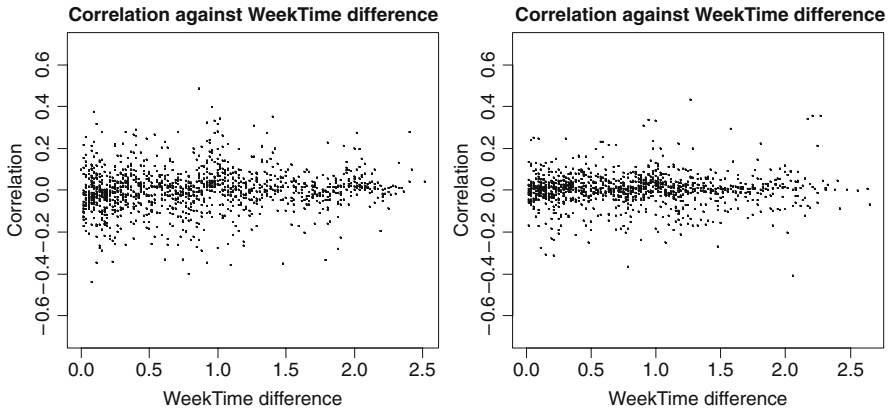

Fig. 23.10 Auto-correlation in u_{si} for negative binomial model with auto-correlated random effects. The *left graph* shows the results for site 1 and the *right graph* for site 2

and the negative binomial model without auto-correlation can also be tried. We leave it for the reader as an exercise.

23.8.1 Comparison of Models

With frequentist statistics, we have tools such as comparison of deviances (for nested models) or AIC at our disposal for comparing models. With Bayesian statistics, one such tool is the Deviance Information Criterion (DIC, Spiegelhalter et al., 2002). It is an extension of AIC and works as follows. Let D($\boldsymbol{\theta}$) be defined as –2 log(f($\mathbf{y}|\boldsymbol{\theta}$)), where $\boldsymbol{\theta}$ contains the model parameters, and f($\mathbf{y}|\boldsymbol{\theta}$) is the likelihood. D($\boldsymbol{\theta}$) is calculated for each realisation of the MCMC chain. Let $\bar{D}$ be the average of D($\boldsymbol{\theta}$). The effective number of parameters, p_D, is calculated as the posterior mean deviance minus the deviance evaluated at the posterior mean:

$$p_D = \bar{D} - D(\bar{\theta})$$

where $\bar{\theta}$ is the average of all realisations of $\boldsymbol{\theta}$. Then, analogous to AIC, the DIC is defined as

$$\text{DIC} = D(\bar{\theta}) + 2 \times p_D = \bar{D} + p_D$$

As with AIC, models with a small DIC are preferred. In R, the command `dicSet` allows for monitoring of the DIC, and the results for the various models fitted in this chapter are given below:

Model	Dbar	Dhat	DIC	pD
Poisson	1890.0	1880.0	1900.0	10.0
Poisson + random effects	783.0	691.0	875.0	92.0
Poisson + auto-correlated random effects	784.5	693.3	875.6	91.1
Negative binomial + auto-correlated random effects	1032.0	1009.0	1055.0	22.8

Note that in the output above, Dhat refers to $D(\bar{\theta})$. Comparing the DIC criteria for all four models, it can be concluded that both Poisson models with random effects are similar in fit and are much better than the Poisson and Negative Binomial models despite them showing some random effect correlation pattern. Formally, the Poisson model with the random effects will be selected as the final model.

23.9 Conclusions

Fitting the Poisson model to the seal abundance data results in overdispersion and auto-correlation. To some extent, these can be addressed by fitting a quasi-Poisson model or generalised linear mixed model. In this chapter, however, we wanted

to introduce the alternative approach of Bayesian models. These form a flexible framework allowing for random effects, auto-correlation, various types of distribution, various (non-canonical) link functions, incorporation of missing values etc. This flexibility comes at a cost though, in that time-consuming simulation methods such as MCMC need to be employed to obtain summaries of the model parameters (although all the models used take no more than half an hour to get 300,000 iterations on a 1.5 GHz PC). Fortunately, freeware software such as WinBugs and its port to R, BRugs, is available to make the MCMC implementation comparatively simple.

It is important to keep in mind that Bayesian statistics is based on a different approach to statistics. It assumes that prior information is available for the parameters, which is then combined with information contained in the data to yield the posterior distribution. This is intrinsically different from frequentist statistics, where the data are analysed in a stand-alone manner, independent from any other sources of information. Although the use of prior information may seem a drawback (as we no longer perform an independent analysis), it can also be used to our advantage. For example, if we are collecting data on an annual basis, then the models can be updated annually where the results from previous years can form the prior distribution, which is then combined with the data from the current year. Furthermore, if an 'independent' Bayesian analysis is required, then this can be achieved by choosing non-informative prior distributions, which should give results similar to those obtained from maximum likelihood estimation. There are several introductory books on Bayesian Statistics, see, for example, Gelman et al. (2003) for a general introduction.

When working with small data sets, the prior distribution can become influential in the posterior result, especially with respect to the spread of the posterior distributions, even if non-informative settings are chosen. This can especially be an issue with prior distributions on the variance components. It may be worthwhile performing a sensitivity analysis where the parameters on the prior distribution are changed and the resulting output compared to the original output.

The aim of this chapter was to give a flavour of Bayesian statistics and by no means covers all aspects of Bayesian analysis. Therefore, no section 'What to write in a paper' is given. However, we hope that it has shown that there is life beyond 'ordinary' generalised linear mixed modelling.

Acknowledgments We would like to thank Grietje Holtrop from Biomathematics & Statistics Scotland for her help with writing this chapter.