Hierarchical Modeling for Non-Gaussian Spatial Data in R

Andrew O. Finley and Sudipto Banerjee

March 9, 2010

1 Synthetic data analysis

We make use of several libraries in the following example session, including:

- library(spBayes)
- library(fields)
- library(geoR)
- library(MBA)
- library(maptools)
- library(rgdal)
- library(sp)

The non-Gaussian outcome data we commonly encounter are binary, rates, or counts which can often be modeled using either a binomial or Poisson generalised linear model (GLM). The function spGLM fits the Poisson and binomial model using the log and logit link function, respectively.

Here we illustrate the use of spGLM to fit a Poisson generalized linear mixed model with spatially dependent random effects. Data from this model are generated in the code block below.

```r
> n <- 50
> coords <- cbind(runif(n, 0, 1), runif(n, 0, 1))
> phi <- 3/0.5
> sigma.sq <- 2
> R <- exp(-phi * iDist(coords))
> w <- mvrnorm(1, rep(0, n), sigma.sq * R)
> beta.0 <- 0.1
> y <- rpois(n, exp(beta.0 + w))
```

Assuming there is no spatial dependence we might fit a simple non-spatial GLM using

```r
> pois.nonsp <- glm(y ~ 1, family = "poisson")
> beta.starting <- coefficients(pois.nonsp)
> beta.tuning <- t(chol(vcov(pois.nonsp)))
```
These coefficients and the Cholesky square root of the parameters’ estimated covariances will be used as starting values and Metropolis sampler tuning values in the subsequent call to `spGLM`. In addition to the regression coefficients we specify starting values for the spatial range \( \phi \) and variance \( \sigma.sq \) as well as the random spatial effects \( w \).

Here posterior inference is based on three MCMC chains each of length 15,000. The code to generate the first of these chains is given below.

```r
> n.batch <- 300
> batch.length <- 50
> n.samples <- n.batch * batch.length
> pois.sp.chain.1 <- spGLM(y ~ 1, family = "poisson",
  + coords = coords, starting = list(beta = beta.starting,
  + phi = 3/0.5, sigma.sq = 1, w = 0), tuning = list(beta = 0.1,
  + phi = 0.5, sigma.sq = 0.1, w = 0.1), priors = list("beta.Flat",
  + phi.Unif = c(3/1, 3/0.1), sigma.sq.IG = c(2,
  + 1)), amcmc = list(n.batch = n.batch, batch.length = batch.length,
  + accept.rate = 0.43), cov.model = "exponential",
  + verbose = TRUE, n.report = 500)
```

----------------------------------------
General model description
----------------------------------------

Model fit with 50 observations.

Number of covariates 1 (including intercept if specified).

Using the exponential spatial correlation model.

Number of MCMC samples 15000.

Priors and hyperpriors:

- beta flat.
- \( \sigma.sq \) IG hyperpriors \( \text{shape}=2.00000 \text{ and scale}=1.00000 \)
- \( \phi \) Unif hyperpriors \( a=3.00000 \text{ and } b=30.00000 \)

Adaptive Metropolis with target acceptance rate: 43.0%

Sampling

Sampled: 15000 of 15000, 100.00%

Again we use the `coda` package’s `plot` function to plot chain trace plots, Figure 1.
The convergence of multiple chains can be assessed using diagnostics detailed in Gelman and Rubin (1992). The `gelman.diag` function in the `coda` package calculates the “potential scale reduction factor” for each each parameter, along with the associated upper and lower confidence limits. Approximate convergence is diagnosed when the upper confidence limit is close to 1. We can also plot the Gelman and Rubin’s shrink factor versus the number of MCMC samples, here again convergence is diagnosed when the upper confidence limit remain close to 1. Figure 2 suggests we should discard the first \( \sim 10,000 \) samples as burn in prior to summarizing the parameters’ posterior distributions.
> print(gelman.diag(samps))

Potential scale reduction factors:

<table>
<thead>
<tr>
<th></th>
<th>Point est.</th>
<th>97.5% quantile</th>
</tr>
</thead>
<tbody>
<tr>
<td>(Intercept)</td>
<td>1.07</td>
<td>1.22</td>
</tr>
<tr>
<td>sigma.sq</td>
<td>1.01</td>
<td>1.03</td>
</tr>
<tr>
<td>phi</td>
<td>1.00</td>
<td>1.01</td>
</tr>
</tbody>
</table>

Multivariate psrf

1.07

> gelman.plot(samps)
> burn.in <- 10000
> print(round(summary(window(samps, start = burn.in))$quantiles[, + c(3, 1, 5)], 2))

<table>
<thead>
<tr>
<th></th>
<th>50%</th>
<th>2.5%</th>
<th>97.5%</th>
</tr>
</thead>
<tbody>
<tr>
<td>(Intercept)</td>
<td>-0.69</td>
<td>-1.63</td>
<td>0.30</td>
</tr>
<tr>
<td>sigma.sq</td>
<td>1.61</td>
<td>0.76</td>
<td>3.60</td>
</tr>
<tr>
<td>phi</td>
<td>10.93</td>
<td>3.45</td>
<td>28.67</td>
</tr>
</tbody>
</table>
Figure 2: MCMC chain convergence diagnostics.
Given the post burn in samples, we can also generate surfaces of the estimated counts Figure 3.

```r
> samps <- as.matrix(window(samps, start = burn.in))
> sp.effects <- cbind(pois.sp.chain.1$sp.effects[, burn.in:n.samples],
+                  pois.sp.chain.2$sp.effects[, burn.in:n.samples],
+                  pois.sp.chain.3$sp.effects[, burn.in:n.samples])
> beta.0.hat <- mean(samps[, "(Intercept)"],)
> w.hat <- apply(sp.effects, 1, mean)
> y.hat <- exp(beta.0.hat + w.hat)
> par(mfrow = c(1, 2))
> surf <- mba.surf(cbind(coords, y), no.X = 100, no.Y = 100,
+                  extend = TRUE)$xyz.est
> image.plot(surf, main = "Observed counts")
> points(coords)
> surf <- mba.surf(cbind(coords, y.hat), no.X = 100,
+                  no.Y = 100, extend = TRUE)$xyz.est
> image.plot(surf, main = "Fitted counts")
> points(coords)
```

Given the posterior samples of the model parameters, we use composition sampling to draw from the posterior predictive distribution of any new location. The code block below constructs a grid to define the prediction locations then calls `spPredict`. For each new location’s posterior predictive samples we can draw a corresponding vector of realizations from `rpois`. These realization can then summarized to assess predictive performance. For example Figure 4 illustrates the median prediction over the domain.
Figure 4: Observed and predicted counts.

```r
> pred.coords <- as.matrix(expand.grid(seq(0.01, 0.99, length.out = 20), seq(0.01, 0.99, length.out = 20)))
> pred.covars <- as.matrix(rep(1, nrow(pred.coords)))
> pois.pred <- spPredict(pois.sp.chain.1, start = 10000, thin = 10, pred.coords = pred.coords, pred.covars = pred.covars, verbose = FALSE)
> y.pred <- apply(pois.pred$y.pred, 1, median)
> par(mfrow = c(1, 2))
> surf <- mba.surf(cbind(coords, y), no.X = 100, no.Y = 100, extend = TRUE)$xyz.est
> image.plot(surf, main = "Observed counts")
> points(coords)
> surf <- mba.surf(cbind(pred.coords, y.pred), no.X = 100, no.Y = 100, extend = TRUE)$xyz.est
> image.plot(surf, main = "Predicted counts")
> points(pred.coords, pch = "x", cex = 1)
> points(coords, pch = 19, cex = 1)
```
2 References


