Hierarchical Modeling for Univariate Spatial Data in R

Andrew O. Finley and Sudipto Banerjee

October 11, 2012

1 Data preparation and initial exploration

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

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

We will use forest inventory data from the U.S. Department of Agriculture Forest Service, Bartlett Experimental Forest (BEF), Bartlett, NH. This dataset holds 1991 and 2002 forest inventory data for 437 plots. Variables include species specific basal area and total tree biomass; inventory plot coordinates; slope; elevation; and tasseled cap brightness (TC1), greenness (TC2), and wetness (TC3) components from spring, summer, and fall 2002 Landsat images.

We use these data to demonstrate some basics of spatial data manipulation, visualization, and univariate spatial regression analysis. The regression model will be used to make prediction of biomass for every image pixel across the BEF.

We begin by removing non-forest inventory plots, converting biomass measurements from kilograms per hectare to the log of metric tons per hectare, and taking a look at plot locations across the forest.

2 Spatial data visualization

```
> data(BEF.dat)
> BEF.dat <- BEF.dat[BEF.dat$ALLBIO02_KGH > 0, ]
> bio <- BEF.dat$ALLBIO02_KGH * 0.001
> log.bio <- log(bio)
> coords <- as.matrix(BEF.dat[, c("XUTM", "YUTM")])
> plot(coords, pch = 19, cex = 0.5, xlab = "Easting (m)",
+     ylab = "Northing (m)")
```

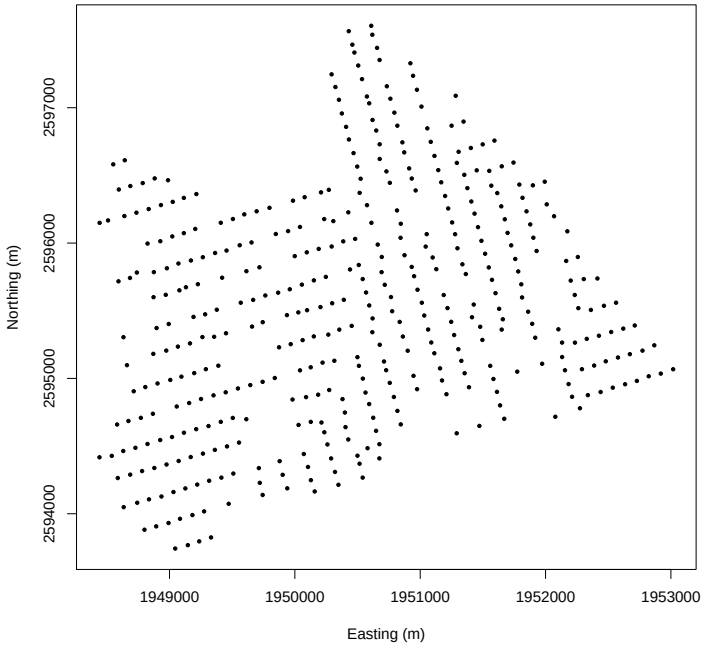

Figure 1: Forest inventory plot locations across the BEF.

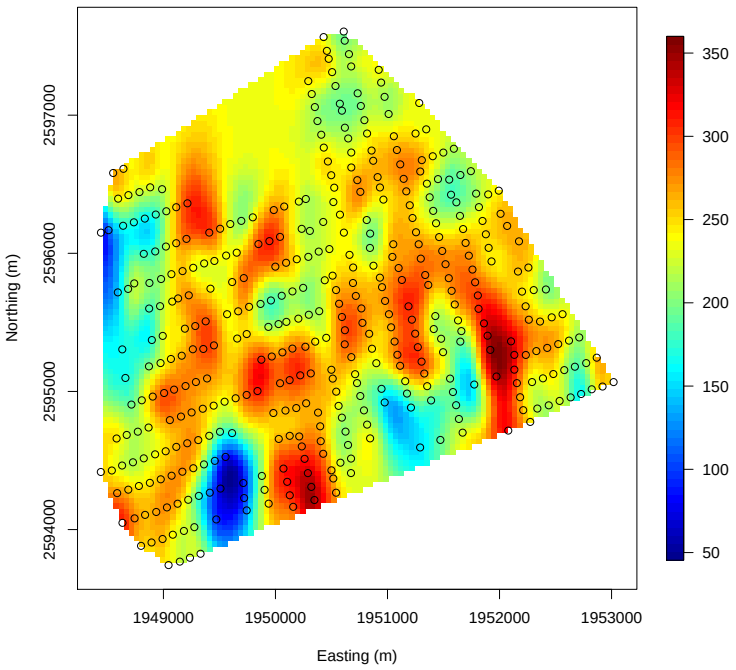

Figure 2: Interpolation of log metric tons of biomass using a Multilevel B-spline.

Our objective is to obtain an estimate, with an associated measure of uncertainty, of biomass as a continuous surface over the domain. We can gain a non-statistical estimate of this surface using **MBA** package which provides efficient interpolation of large data sets using multilevel B-splines. The result of the `mba.surf` function can be passed to `image` or `image.plot` to produce $\Re^2$ depictions, Figure 2.

```
> x.res <- 100
> y.res <- 100
> surf <- mba.surf(cbind(coords, bio), no.X = x.res,
+     no.Y = y.res, h = 5, m = 2, extend = FALSE)$xyz.est
> image.plot(surf, xaxs = "r", yaxs = "r", xlab = "Easting (m)",
+     ylab = "Northing (m)")
> points(coords)
```

In addition to the image plots, functions in the **rgl** package can produce informative $\Re^3$ depictions of biomass over the BEF.

```
> col <- rbind(0, cbind(matrix(drape.color(surf[[3]]),
```

```
+     x.res - 1, y.res - 1), 0))
> surface3d(surf[[1]], surf[[2]], surf[[3]], col = col)
> axes3d()
> title3d(main = "Biomass", xlab = "Easting (m)", ylab = "Northing (m)",
+     zlab = "Log metric tons of biomass")
```

We hope to improve prediction by using elevation and slope topographic variables and variables derived from a summer date of 30×30 m resolution Landsat ETM+ satellite imagery. These predictor variables are included in the `BEF.dat` dataset. In the code block below we regress biomass onto the set of predictor variables, then use `mba.surf` to make image plots of the residuals and predictor variables, Figure 3.

```
> lm.bio <- lm(log.bio ~ ELEV + SLOPE + SUM_02_TC1 +
+     SUM_02_TC2 + SUM_02_TC3, data = BEF.dat)
> summary(lm.bio)

Call:
lm(formula = log.bio ~ ELEV + SLOPE + SUM_02_TC1 + SUM_02_TC2 +
    SUM_02_TC3, data = BEF.dat)

Residuals:
     Min       1Q   Median       3Q      Max
-1.79092 -0.14262  0.04728  0.20521  0.67744

Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept)  0.7534253  0.6812872   1.106 0.269426
ELEV         0.0006276  0.0001923   3.264 0.001190 **
SLOPE       -0.0104332  0.0029867  -3.493 0.000529 ***
SUM_02_TC1   0.0094663  0.0054483   1.737 0.083054 .
SUM_02_TC2   0.0068053  0.0035748   1.904 0.057649 .
SUM_02_TC3   0.0241059  0.0049732   4.847 1.78e-06 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 0.3166 on 409 degrees of freedom
Multiple R-squared: 0.1966,        Adjusted R-squared: 0.1868
F-statistic: 20.02 on 5 and 409 DF,  p-value: < 2.2e-16

> bio.resid <- resid(lm.bio)
> par(mfrow = c(2, 3))
> surf <- mba.surf(cbind(coords, bio.resid), no.X = x.res,
+     no.Y = y.res, h = 5, m = 2, extend = FALSE)$xyz.est
> image.plot(surf, xaxs = "r", yaxs = "r", main = "Log metric tons of biomass residuals",
+     xlab = "Easting (m)", ylab = "Northing (m)")
> covars <- c("ELEV", "SLOPE", "SUM_02_TC1", "SUM_02_TC2",
```

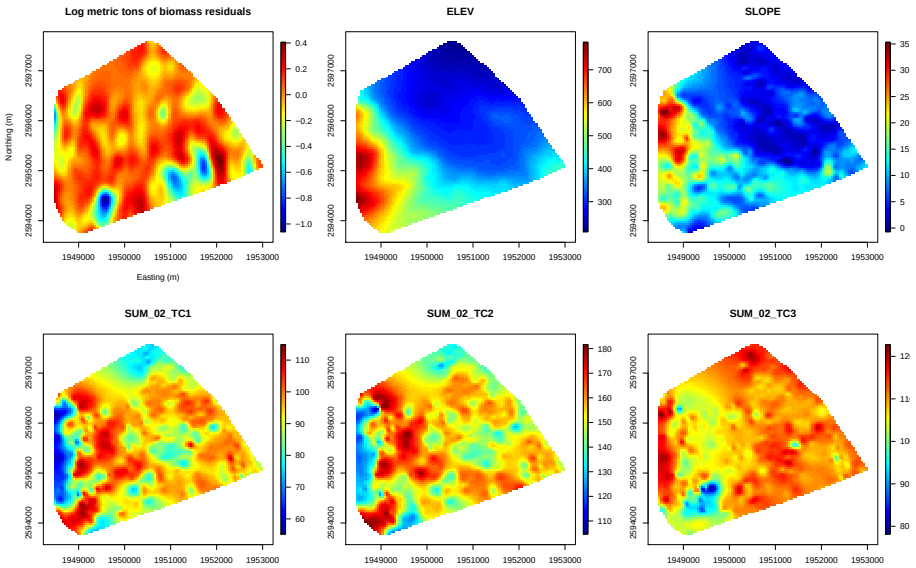

Figure 3: Interpolation of log metric tons of biomass and predictor variables using a Multilevel B-spline.

```
+     "SUM_02_TC3")
> for (i in 1:5) {
+     surf <- mba.surf(cbind(coords, BEF.dat[, covars[i]]),
+         no.X = x.res, no.Y = y.res, extend = FALSE)$xyz.est
+     image.plot(surf, xaxs = "r", yaxs = "r", main = covars[i])
+ }
```

The residual image plot in Figure 3 suggests that there is spatial dependence even after accounting for the predictor variables. These patterns can be more formally examined using empirical semivariograms. In the code block below, we first fit an exponential variogram model to log biomass then fit a second variogram model to the regression residuals. The resulting variograms are offered in Figure 4. Here the upper and lower horizontal lines are the *sill* and *nugget*, respectively, and the vertical line is the effective range (i.e., that distance at which the correlation drops to 0.05).

```
> max.dist <- 0.5 * max(iDist(coords))
> bins <- 20
> vario.bio <- variog(coords = coords, data = log.bio,
+     uvec = (seq(0, max.dist, length = bins)))

variog: computing omnidirectional variogram
```

```
> fit.bio <- variofit(vario.bio, ini.cov.pars = c(0.08,
+     500/-log(0.05)), cov.model = "exponential", minimisation.function = "nls",
+     weights = "equal")

variofit: covariance model used is exponential
variofit: weights used: equal
variofit: minimisation function used: nls

> vario.bio.resid <- variog(coords = coords, data = bio.resid,
+     uvec = (seq(0, max.dist, length = bins)))

variog: computing omnidirectional variogram

> fit.bio.resid <- variofit(vario.bio.resid, ini.cov.pars = c(0.08,
+     500/-log(0.05)), cov.model = "exponential", minimisation.function = "nls",
+     weights = "equal")

variofit: covariance model used is exponential
variofit: weights used: equal
variofit: minimisation function used: nls

> par(mfrow = c(1, 2))
> plot(vario.bio, main = "Log metric tons of biomass")
> lines(fit.bio)
> abline(h = fit.bio$nugget, col = "blue")
> abline(h = fit.bio$cov.pars[1] + fit.bio$nugget,
+     col = "green")
> abline(v = -log(0.05) * fit.bio$cov.pars[2], col = "red3")
> plot(vario.bio.resid, main = "Log metric tons of biomass residuals")
> lines(fit.bio.resid)
> abline(h = fit.bio.resid$nugget, col = "blue")
> abline(h = fit.bio.resid$cov.pars[1] + fit.bio.resid$nugget,
+     col = "green")
> abline(v = -log(0.05) * fit.bio.resid$cov.pars[2],
+     col = "red3")
```

Figure 4 corroborates what is shown in the image plots. Specifically, there is substantial spatial dependence in biomass across the inventory plots and that this dependence persists, but to a lesser degree, after accounting for the predictor variables. Therefore, we expect that both the predictor variables and spatial proximity to inventory plots will improve prediction.

3 Fitting the spatial regression model

The empirical semivariograms estimates of the partial sill, σ^2, nugget, τ^2, and range ϕ provide good starting values to use in the **spBayes** univariate spatial regression functions `bayesGeostatExact` and `spLM`.

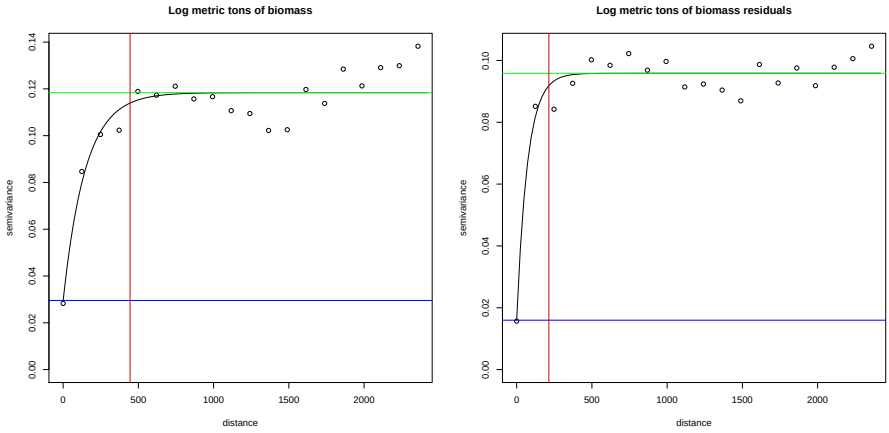

Figure 4: Isotropic semivariograms for log metric tons of biomass and residuals from a linear regression.

```
> p <- 6
> beta.prior.mean <- as.matrix(rep(0, times = p))
> beta.prior.precision <- matrix(0, nrow = p, ncol = p)
> phi <- 0.014
> alpha <- 0.016/0.08
> sigma.sq.prior.shape <- 2
> sigma.sq.prior.rate <- 0.08
> sp.exact <- bayesGeostatExact(log.bio ~ ELEV + SLOPE +
+     SUM_02_TC1 + SUM_02_TC2 + SUM_02_TC3, data = BEF.dat,
+     coords = coords, n.samples = 1000, beta.prior.mean = beta.prior.mean,
+     beta.prior.precision = beta.prior.precision,
+     phi = phi, alpha = alpha, sigma.sq.prior.shape = sigma.sq.prior.shape,
+     sigma.sq.prior.rate = sigma.sq.prior.rate, sp.effects = FALSE)

------------------------------------------------
        General model description
------------------------------------------------
Model fit with 415 observations.
Number of covariates 6 (including intercept if specified).
Using the exponential spatial correlation model.

------------------------------------------------
               Sampling
------------------------------------------------
Sampled: 1000 of 1000, 100%

> round(summary(sp.exact$p.samples)$quantiles, 3)
```

```
              2.5%    25%    50%    75%  97.5%
(Intercept) -0.315  0.670  1.153  1.723  2.715
ELEV         0.000  0.000  0.000  0.001  0.001
SLOPE       -0.015 -0.011 -0.009 -0.007 -0.002
SUM_02_TC1  -0.002  0.006  0.010  0.015  0.023
SUM_02_TC2  -0.003  0.003  0.006  0.009  0.013
SUM_02_TC3   0.010  0.017  0.021  0.025  0.033
sigma.sq     0.072  0.078  0.082  0.086  0.095
tau.sq       0.014  0.016  0.016  0.017  0.019
```

A more robust, but computationally demanding, alternative to `bayesGeostatExact` is the `spLM` function. For brevity, we only take 10 MCMC sample in the code block below, however, we subsequently load samples from a previous run of 10,000 samples less 1,000 burn in samples (which took ∼10 minutes to run).

```
> n.samples <- 10
> bef.sp <- spLM(log.bio ~ ELEV + SLOPE + SUM_02_TC1 +
+     SUM_02_TC2 + SUM_02_TC3, data = BEF.dat, coords = coords,
+     starting = list(phi = 3/200, sigma.sq = 0.08,
+         tau.sq = 0.02), sp.tuning = list(phi = 0.1,
+         sigma.sq = 0.05, tau.sq = 0.05), priors = list(phi.Unif = c(3/1500,
+         3/50), sigma.sq.IG = c(2, 0.08), tau.sq.IG = c(2,
+         0.02)), cov.model = "exponential", n.samples = n.samples,
+     sub.samples = c(1, n.samples, 1), verbose = TRUE,
+     n.report = 100)

----------------------------------------
        General model description
----------------------------------------
Model fit with 415 observations.

Number of covariates 6 (including intercept if specified).

Using the exponential spatial correlation model.

Number of MCMC samples 10.

Priors and hyperpriors:
        beta flat.
        sigma.sq IG hyperpriors shape=2.00000 and scale=0.08000
        tau.sq IG hyperpriors shape=2.00000 and scale=0.02000
        phi Unif hyperpriors a=0.00200 and b=0.06000
------------------------------------------------
                Sampling
------------------------------------------------
Sampled: 10 of 10, 100.00%
------------------------------------------------
```

```
> load(file = "R-data/bef.splm")

> par(mfrow = c(3, 2))
> plot(bef.sp$p.samples[, 1:6], auto.layout = TRUE,
+     density = FALSE)

> par(mfrow = c(2, 2))
> effective.range <- 3/bef.sp$p.samples[, 9]
> plot(mcmc(cbind(bef.sp$p.samples[, 7:9], effective.range)),
+     auto.layout = TRUE, density = FALSE)

> round(summary(mcmc(bef.sp$p.samples))$quantiles,
+     3)

              2.5%    25%    50%    75% 97.5%
(Intercept) -0.317  0.774  1.356  1.947 3.136
ELEV         0.000  0.000  0.000  0.001 0.001
SLOPE       -0.015 -0.010 -0.008 -0.005 0.000
SUM_02_TC1  -0.003  0.006  0.011  0.015 0.025
SUM_02_TC2  -0.004  0.002  0.005  0.008 0.014
SUM_02_TC3   0.007  0.016  0.020  0.024 0.033
sigma.sq     0.038  0.062  0.079  0.092 0.109
tau.sq       0.005  0.012  0.025  0.042 0.063
phi          0.005  0.008  0.010  0.011 0.016


> w.hat.mu <- apply(bef.sp$sp.effects, 1, mean)
> w.hat.sd <- apply(bef.sp$sp.effects, 1, sd)
> par(mfrow = c(1, 2))
> surf <- mba.surf(cbind(coords, bio.resid), no.X = x.res,
+     no.Y = y.res, extend = FALSE)$xyz.est
> z.lim <- range(surf[[3]], na.rm = TRUE)
> image.plot(surf, xaxs = "r", yaxs = "r", zlim = z.lim,
+     main = "LM residuals")
> surf <- mba.surf(cbind(coords, w.hat.mu), no.X = x.res,
+     no.Y = y.res, extend = FALSE)$xyz.est
> image.plot(surf, xaxs = "r", yaxs = "r", zlim = z.lim,
+     main = "Mean spatial effects")

> par(mfrow = c(1, 2))
> surf <- mba.surf(cbind(coords, bio.resid), no.X = x.res,
+     no.Y = y.res, extend = FALSE)$xyz.est
> image.plot(surf, xaxs = "r", yaxs = "r", main = "LM residuals")
> surf <- mba.surf(cbind(coords, w.hat.sd), no.X = x.res,
+     no.Y = y.res, extend = FALSE)$xyz.est
> image.plot(surf, xaxs = "r", yaxs = "r", main = "SD spatial effects")
```

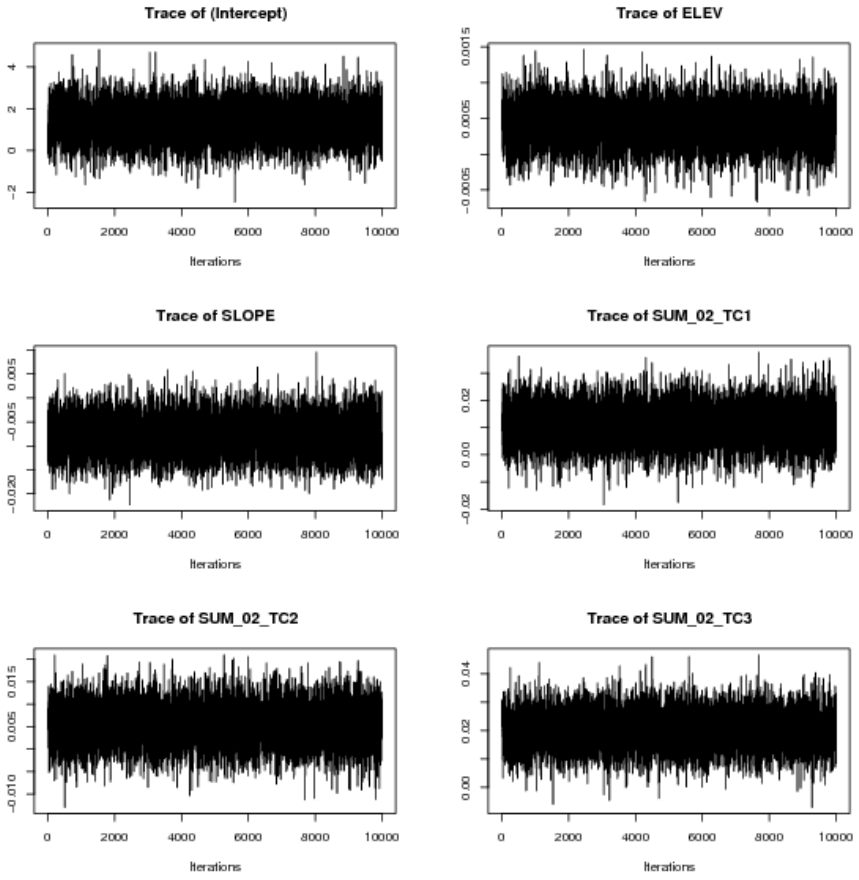

Figure 5: MCMC trace plots of $\boldsymbol{\beta}$

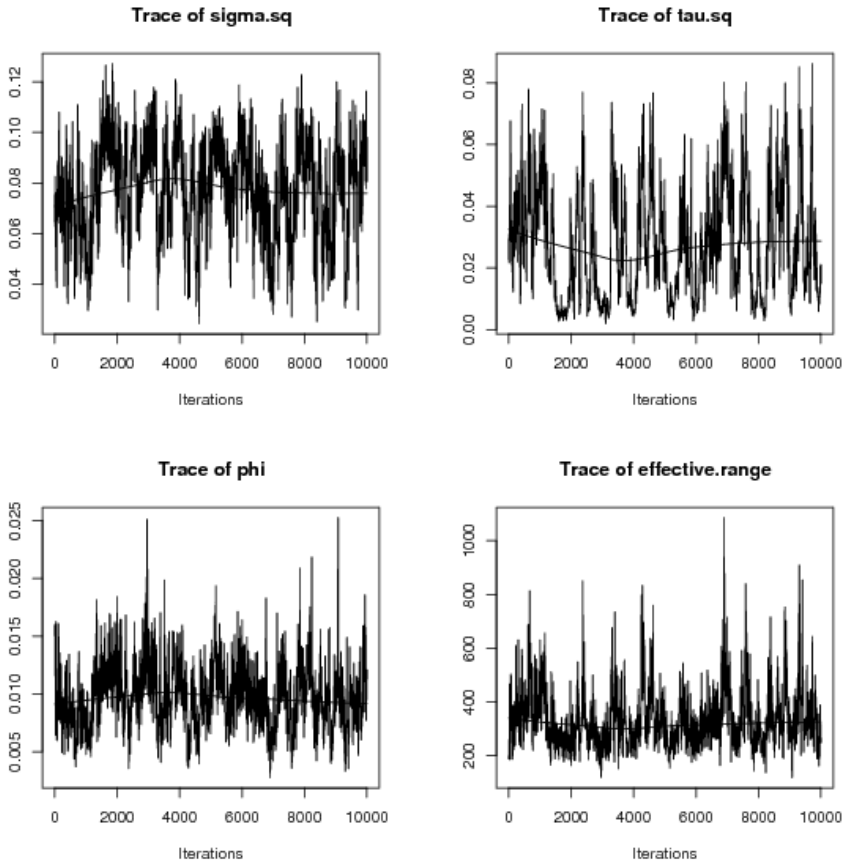

Figure 6: MCMC trace plots of spatial parameters, σ^2, τ^2, ϕ, and effective range.

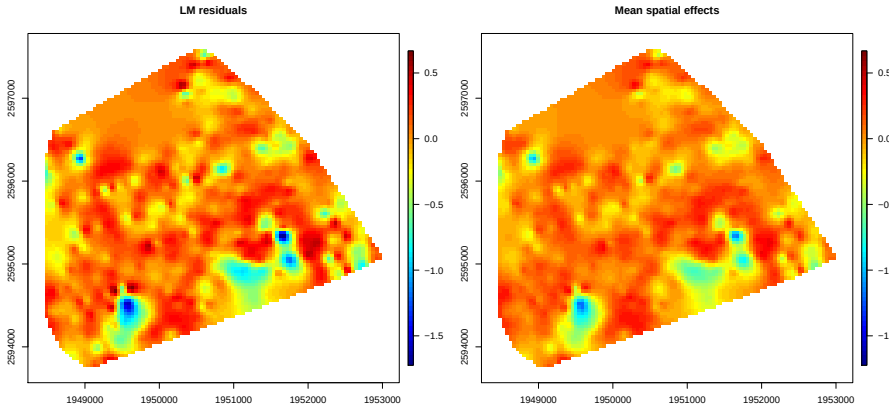

Figure 7: Interpolated surface of the non-spatial model residuals and the mean of the random spatial effects posterior distribution.

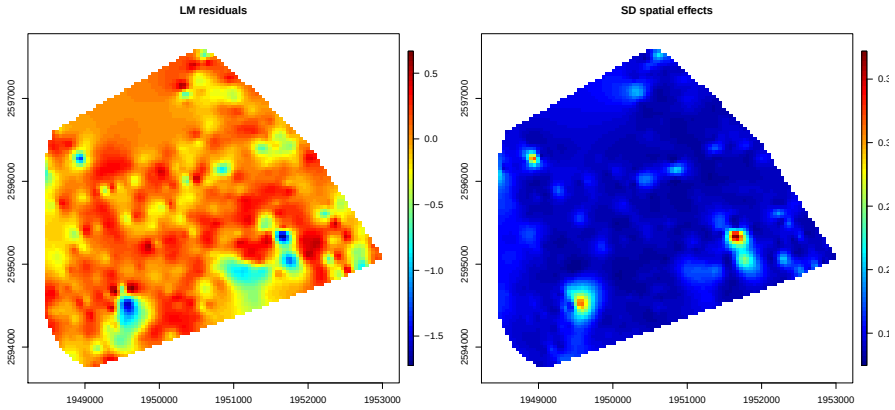

Figure 8: Interpolated surface of the non-spatial model residuals and the standard deviation of the random spatial effects posterior distribution.

4 Prediction

Given the samples from the parameters' posterior distribution we can now turn to prediction. Using the `spLM` object and predictor variables from *new* locations, the function `spPredict` allows us to sample from the posterior predictive distribution of every pixel across the BEF. We are only interested in predictions within the BEF; however, the predictor variable grid extends well beyond the BEF bounds. Therefore, we would like to *clip* the predictor grid to the BEF bounding polygon. The code block below makes use of the `readShapePoly` function from the **maptools** package and `readGDAL` function from the **rgdal** package to read the bounding polygon and predictor variable grid stack, respectively. We then construct the prediction design matrix for the entire grid extent. Then extract the coordinates of the BEF bounding polygon vertices and use the `pointsInPoly` **spBayes** function to obtain the desired subset of the prediction design matrix and associated prediction coordinates (i.e., pixel centroids). Finally, the `spPredict` function is called and posterior predictive samples are stored in `bef.bio.pred`. Again for brevity, we are reading in samples from a previous run of `spPredict` (i.e., note we specify `start=1` and `start=2` only to save computing time in this illustration).

```
> BEF.shp <- readShapePoly("BEF-data/BEF_bound.shp")
> BEF.poly <- as.matrix(BEF.shp@polygons[[1]]@Polygons[[1]]@coords)
> BEF.grids <- readGDAL("BEF-data/dem_slope_lolosptc_clip_60.img")

BEF-data/dem_slope_lolosptc_clip_60.img has GDAL driver GTiff
and has 81 rows and 81 columns

> pred.covars <- cbind(BEF.grids[["band1"]], BEF.grids[["band2"]],
+     BEF.grids[["band3"]], BEF.grids[["band4"]], BEF.grids[["band5"]])
> pred.covars <- cbind(rep(1, nrow(pred.covars)), pred.covars)
> pred.coords <- SpatialPoints(BEF.grids)@coords
> pred.covars <- pred.covars[pointsInPoly(BEF.poly,
+     pred.coords), ]
> pred.coords <- pred.coords[pointsInPoly(BEF.poly,
+     pred.coords), ]
> bef.bio.pred <- spPredict(bef.sp, start = 1, end = 2,
+     pred.coords = pred.coords, pred.covars = pred.covars,
+     verbose = FALSE)
> load(file = "R-data/bef.splm.predict")
```

With access to each pixel's posterior predictive distribution we can map any summary statistics of interest. In Figure 9 we compare the log metric tons of biomass interpolated over the observed plots to that of the pixel-level prediction. The generation of this image plot requires some additional code to clip the interpolation grid produced by `mba.surf` to the BEF polygon. Here we also demonstrate the **sp** function `overlay` to subset the grid (that is an alternative approach to using `pointsInPoly`). Figure 10 shows the standard

deviation of the posterior predictive distributions. From this image plot it is clear that predictions closer to observed inventory plots have higher precision, as expected.

```
> bef.bio.pred.mu <- apply(bef.bio.pred$y.pred, 1,
+     mean)
> bef.bio.pred.sd <- apply(bef.bio.pred$y.pred, 1,
+     sd)
> surf <- mba.surf(cbind(coords, log.bio), no.X = x.res,
+     no.Y = x.res, extend = TRUE, sp = TRUE)$xyz.est
> surf <- surf[!is.na(overlay(surf, BEF.shp)), ]
> surf <- as.image.SpatialGridDataFrame(surf)
> z.lim <- range(surf[["z"]], na.rm = TRUE)
> pred.grid <- as.data.frame(list(pred.coords, pred.mu = bef.bio.pred.mu,
+     pred.sd = bef.bio.pred.sd))
> coordinates(pred.grid) = c("x", "y")
> gridded(pred.grid) <- TRUE
> pred.mu.image <- as.image.SpatialGridDataFrame(pred.grid["pred.mu"])
> par(mfrow = c(1, 2))
> image.plot(surf, axes = TRUE, zlim = z.lim, col = tim.colors(25),
+     xaxs = "r", yaxs = "r", main = "Log metric tons of biomass")
> plot(BEF.shp, add = TRUE)
> image.plot(pred.mu.image, zlim = z.lim, col = tim.colors(25),
+     xaxs = "r", yaxs = "r", main = "Mean predicted log metric tons of biomass")
> plot(BEF.shp, add = TRUE)

> pred.sd.image <- as.image.SpatialGridDataFrame(pred.grid["pred.sd"])
> image.plot(pred.sd.image, axes = TRUE, col = tim.colors(25),
+     xaxs = "r", yaxs = "r")
> plot(BEF.shp, add = TRUE)
> points(coords, cex = 1)
```

Finally, we often want to combine our prediction surface with other spatial data within a full featured Geographic Information system (e.g., GRASS, QGIS, ArcGIS, etc.). **sp** grid data objects can be exported using functions within **rgdal** as described in the code block below.

```
> writeGDAL(pred.grid["pred.mu"], "BEF_Pred_mu_biomass.tif")
> writeGDAL(pred.grid["pred.sd"], "BEF_Pred_sd_biomass.tif")
```

4.1 Model selection

To compare several alternative models with varying degrees of richness, we might use the GP criterion (Gelfand and Ghosh, 1998) or the Deviance Information Criterion (DIC) (Spiegelhalter et al., 2002). Letting Ω be the generic set of parameters being estimated for each model (including random effects), we compute the expected posterior deviance $\overline{D(\Omega)} = E_{\Omega|\mathbf{Y}}\{-2\log L(Data|\ \Omega)\}$, where

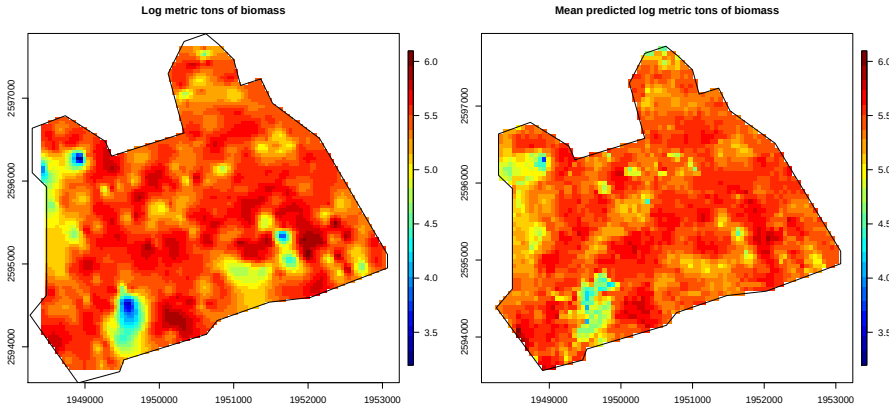

Figure 9: Interpolated surface of observed log metric tons of biomass and mean of pixels' posterior predictive distribution.

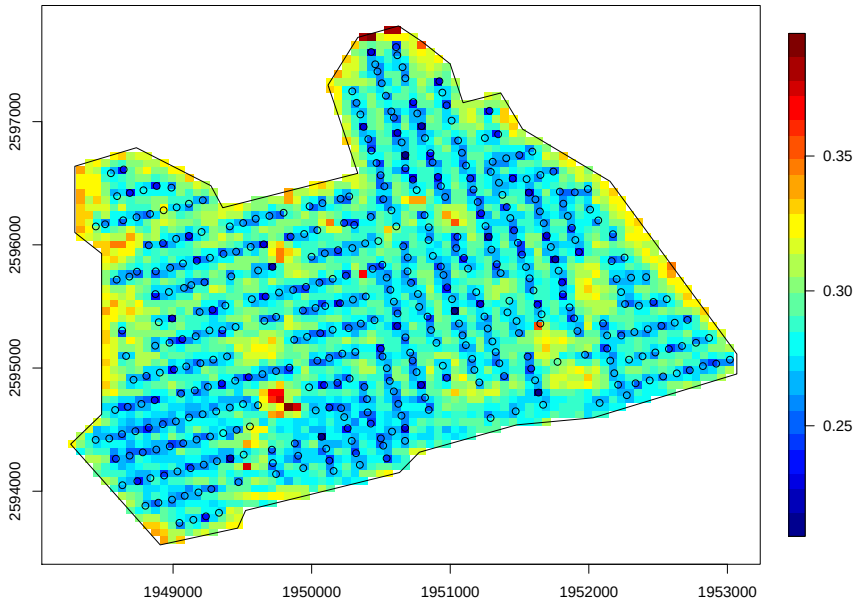

Figure 10: Interpolated surface of the standard deviation of pixels' posterior predictive distribution.

$L(Data|\ \Omega)$ is the first stage Gaussian likelihood from the respective model and the effective number of parameters (as a penalty) as $p_D = \overline{D(\Omega)} - D(\bar{\Omega})$, where $\bar{\Omega}$ is the posterior mean of the model parameters. The DIC is then given by $\overline{D(\Omega)} + p_D$ and is easily computed from the posterior samples with lower values indicating better models. Again, we will load samples from previous runs of length 10,000 samples, less 1,000 burn in and thinned.

```
> set.seed(1)
> n.samples <- 10
> lm.obj <- lm(log.bio ~ ELEV + SLOPE + SUM_02_TC1 +
+     SUM_02_TC2 + SUM_02_TC3, data = BEF.dat)
> cand.1 <- bayesLMRef(lm.obj, n.samples)
> load(file = "R-data/cand.1")
> cand.2 <- spLM(log.bio ~ 1, data = BEF.dat, coords = coords,
+     starting = list(phi = 3/200, sigma.sq = 0.08,
+         tau.sq = 0.02), sp.tuning = list(phi = 0.1,
+         sigma.sq = 0.05, tau.sq = 0.05), priors = list(phi.Unif = c(3/1500,
+         3/50), sigma.sq.IG = c(2, 0.08), tau.sq.IG = c(2,
+         0.02)), cov.model = "exponential", n.samples = n.samples,
+     sub.samples = c(1, n.samples, 1), verbose = FALSE)
> load(file = "R-data/cand.2")
> cand.3 <- spLM(log.bio ~ ELEV + SLOPE + SUM_02_TC1 +
+     SUM_02_TC2 + SUM_02_TC3, data = BEF.dat, coords = coords,
+     starting = list(phi = 3/200, sigma.sq = 0.08,
+         tau.sq = 0.02), sp.tuning = list(phi = 0.1,
+         sigma.sq = 0.05, tau.sq = 0.05), priors = list(phi.Unif = c(3/1500,
+         3/50), sigma.sq.IG = c(2, 0.08), tau.sq.IG = c(2,
+         0.02)), cov.model = "exponential", n.samples = n.samples,
+     sub.samples = c(1, n.samples, 1), verbose = FALSE)
> load(file = "R-data/cand.3")
> cand.1.DIC <- spDiag(cand.1, start = 1000, thin = 10,
+     verbose = FALSE)
> cand.2.DIC <- spDiag(cand.2, verbose = FALSE)
> cand.3.DIC <- spDiag(cand.3, verbose = FALSE)
```

	bar.D	D.bar.Omega	pD	DIC
Non spatial	-538.30	-545.50	7.20	-531.10
Spatial intercept only	-1177.60	-1399.70	222.10	-955.50
Spatial with predictors	-1167.00	-1356.00	188.90	-978.10

Table 1: Candidate model comparison using DIC

	G	P	D
Non spatial	41.00	42.40	83.30
Spatial intercept only	3.00	18.70	21.70
Spatial with predictors	3.60	19.60	23.30

Table 2: Candidate model comparison using GP

5 References

Banerjee, S., Carlin, B.P., and Gelfand, A.E. (2004). *Hierarchical Modeling and Analysis for Spatial Data*, Boca Raton, FL: Chapman and Hall/CRC Press.

Bivand, R.B., Pebesma, E.J., and Gómez-Rubio, V. (2008). *Applied Spatial Data Analysis with R*, UseR! Series, Springer.

Diggle, P.J. and Riberio, P.J. (2007). *Model-based Geostatistics*, Series in Statistics, Springer.

Gelfand A.E. and Ghosh, S.K. (1998). Model choice: a minimum posterior predictive loss approach. *Biometrika*. 85:1-11.

Gelman, A., Carlin, J.B., Stern, H.S., and Rubin, D.B. (2004). *Bayesian Data Analysis*. Second Edition. Boca Raton, FL: Chapman and Hall/CRC Press.

Spiegelhalter, D.J., Best, N.G., Carlin, B.P., and van der Linde, A. (2002). Bayesian measures of model complexity and fit (with discussion). *Journal of the Royal Statistical Society, Series B* **64**, 583–639.