spBayes Tutorial

Sam VanSchalkwyk

March 1, 2018

The `spBayes` Package

The `spBayes` package is used for univariate and multivariate spatial-temporal modeling.

Consider the following hierarchical linear mixed model framework:

$$p(\boldsymbol{\theta}) \times N(\boldsymbol{\beta}|\boldsymbol{\mu}_\beta, \boldsymbol{\Sigma}_\beta) \times N(\boldsymbol{\alpha}|\mathbf{0}, \boldsymbol{K}(\boldsymbol{\theta})) \times N(\boldsymbol{y}|\boldsymbol{X}\boldsymbol{\beta} + \boldsymbol{Z}(\boldsymbol{\theta})\boldsymbol{\alpha}, \boldsymbol{D}(\boldsymbol{\theta}))$$

In this model,

- $\mathbf{y}_{n\times 1}$ is a vector of possibly irregularly located observations
- $\mathbf{X}_{n\times p}$ is a known matrix of regressors
- $\mathbf{K}(\boldsymbol{\theta})_{r\times r}$ and $\mathbf{D}(\boldsymbol{\theta})_{n\times n}$ are covariance matrices ($\mathbf{K}(\boldsymbol{\theta})$ is the spatial covariance matrix)
- $\mathbf{Z}(\boldsymbol{\theta})_{n\times r}$ with $r \leq n$
- $\boldsymbol{\theta}$ is a set of unknown process parameters
- $\boldsymbol{\alpha}_{r\times 1} \sim N(\mathbf{0}, \boldsymbol{K}(\boldsymbol{\theta}))$ is a random vector for spatial random effects, which arises from a partial realization of a spatial process
- $\boldsymbol{\beta}_{p\times 1} \sim N(\boldsymbol{\mu}_\beta, \boldsymbol{\Sigma}_\beta)$ is a slope vector, where $\boldsymbol{\mu}_\beta$ and $\boldsymbol{\Sigma}_\beta$ are known.

Then assume that $\boldsymbol{\theta} \sim p(\boldsymbol{\theta})$, a proper prior distribution. The models in the spBayes package are all special cases of this model. To perform Bayesian inference, sample from the posterior distribution of $(\boldsymbol{\beta}, \boldsymbol{\alpha}, \boldsymbol{\theta})$, which is proportional to the above hierarchical model.

Models in the `spBayes` package

Let $\{w(\boldsymbol{s}) : \boldsymbol{s} \in \mathbb{R}^d\}$ be a Gaussian spatial process with positive definite covariance function $C(\boldsymbol{s}, \boldsymbol{t}, \boldsymbol{\theta})$. If $\{s_1, s_2, \ldots, s_r\}$ is a set of r locations, then $\boldsymbol{\alpha} = (w(s_1), w(s_2), \ldots, w(s_r))'$ with $r \times r$ covariance matrix $\boldsymbol{K}(\boldsymbol{\theta})$.

Gaussian univariate spatial regression

Consider the model

$$y(\boldsymbol{s}) = \boldsymbol{x}(\boldsymbol{s})'\boldsymbol{\beta} + w(\boldsymbol{s}) + \epsilon(\boldsymbol{s})$$

for locations $\mathbf{s}$. Here, $w(\boldsymbol{s})$ is a spatial process and $\epsilon(\boldsymbol{s})$ is an independent white-noise process. Assume $\epsilon(s_i)$'s are i.i.d. $N(0, \tau^2)$, where $i = 1, \ldots, n$ for n locations. Let $C(\boldsymbol{s}, \boldsymbol{t}) = \sigma^2\rho(\boldsymbol{s}, \boldsymbol{t}, \boldsymbol{\phi})$, where ρ is a correlation function such that $\rho(\boldsymbol{s}, \boldsymbol{t}, \boldsymbol{\phi}) = exp(-\boldsymbol{\phi}||\boldsymbol{s} - \boldsymbol{t}||)$, $||\boldsymbol{s} - \boldsymbol{t}||$ is the Euclidean distance between sites $\mathbf{s}$ and $\mathbf{t}$, and $\text{Var}(w(\boldsymbol{s})) = \sigma^2$.

This model can be implemented with the `spLM()` function in the `spBayes` package. Once this is done, samples from the posterior distribution of $\boldsymbol{\beta}$ and $\mathbf{w}$ can be recovered predictively, given samples of $\boldsymbol{\theta}$, by passing the `spLM` object to the function `spRecover()`.

The `spLM()` function

For more complete details on the `spLM()` function and other functions discussed here, refer to the package manual for the `spBayes` package, available on CRAN at https://cran.r-project.org/web/packages/spBayes/spBayes.pdf. The function in general is

```
spLM(formula,coords,starting,tuning,priors,cov.model,n.samples,n.report,...)
```

where some commonly specified arguments are

- **formula**, a description of the regression model to be fit
- **coords**, an $n \times 2$ matrix of the observation coordinates in $\mathbb{R}^2$
- **starting**, a list with parameter names (could be beta, sigma.sq, tau.sq, phi, or nu). The value in the list is the parameter's starting value
- **tuning**, a list with parameter names, where the value now defines the variance of the Metropolis sampler Normal proposal distribution
- **priors**, a list with parameter names (could be sigma.sq.ig, tau.sq.ig, phi.unif, nu.unif, beta.norm, or beta.flat). Variance parameters, sigma.sq and tau.sq, are assumed to follow an inverse-Gamma distribution, and phi is assumed to follow a Uniform distribution. Hyperparameters of the inverse-Gamma and Uniform are passed as a vector of length 2. If the beta vector are assumed to follow a multivariate Normal distribution, then pass the hyperparameters as a list of length 2. If beta is assumed flat then no arguments are specified. The default is a flat prior
- **cov.model**, a keyword that specifies the covariance function used to model the spatial dependence structure (could be exponential, matern, spherical, or gaussian)
- **n.samples**, the number of MCMC iterations
- **n.report**, the interval to report Metropolis sampler acceptance and MCMC progress

The `spRecover()` function

In general, the function looks like

```
spRecover(sp.obj,start=1,verbose=TRUE,...)
```

where some commonly specified arguments are

- **sp.obj**, an object returned by spLM, spMvLM, or spMisalignLM
- **start**, specifies the first sample included in the composition sampling
- **verbose**, if TRUE, model specification and progress of the sampler is printed to the screen

Using the package

Code to implement these functions is shown below. First, install the `spBayes` package:

```
install.packages(spBayes)
```

```
library(spBayes)
```

```
## Loading required package: coda
## Loading required package: magic
## Loading required package: abind
```

```
## Loading required package: Formula
```

```
# function to make sure inputs are compatible
rmvn <- function(n, mu=0, V = matrix(1)){
  p <- length(mu)
  if(any(is.na(match(dim(V),p))))
    stop("Dimension problem!")
  D <- chol(V)
  t(matrix(rnorm(n*p), ncol=p)%*%D + rep(mu,rep(n,p)))
}

set.seed(1)

# n is number of observations
n <- 100
# coords are observation coordinates for spLM()
coords <- cbind(runif(n,0,1), runif(n,0,1))
# X is matrix of regressors
X <- as.matrix(cbind(1, rnorm(n)))

# B is slope vector
B <- as.matrix(c(1,5))
# p is number of parameters
p <- length(B)

sigma.sq <- 2
tau.sq <- 0.1
phi <- 3/0.5

# D is distance matrix for coordinates
D <- as.matrix(dist(coords))
# R is correlation function
R <- exp(-phi*D)
# w is a spatial process
w <- rmvn(1, rep(0,n), sigma.sq*R)
# y is a vector of spatially referenced dependent variables
y <- rnorm(n, X%*%B + w, sqrt(tau.sq))

# number of MCMC iterations, used in spLM()
n.samples <- 2000

# starting values for parameters in spLM()
starting <- list("phi"=3/0.5, "sigma.sq"=50, "tau.sq"=1)

# variances for the Metropolis sampler in spLM()
tuning <- list("phi"=0.1, "sigma.sq"=0.1, "tau.sq"=0.1)

# priors for parameters in spLM(): betas are multivariate Normal
priors.1 <- list("beta.Norm"=list(rep(0,p), diag(1000,p)),
                 "phi.Unif"=c(3/1, 3/0.1), "sigma.sq.IG"=c(2, 2),
                 "tau.sq.IG"=c(2, 0.1))

# priors for parameters in spLM(): betas are flat
priors.2 <- list("beta.Flat", "phi.Unif"=c(3/1, 3/0.1),
```

```
                  "sigma.sq.IG"=c(2, 2), "tau.sq.IG"=c(2, 0.1))

# function for spatial dependence structure in spLM()
cov.model <- "exponential"

# interval for seeing progress of the sampler in spLM()
n.report <- 500

# model with first set of priors
m.1 <- spLM(y~X-1, coords=coords, starting=starting,
            tuning=tuning, priors=priors.1, cov.model=cov.model,
            n.samples=n.samples, n.report=n.report)
```

```
## ----------------------------------------
##  General model description
## ----------------------------------------
## Model fit with 100 observations.
##
## Number of covariates 2 (including intercept if specified).
##
## Using the exponential spatial correlation model.
##
## Number of MCMC samples 2000.
##
## Priors and hyperpriors:
##  beta normal:
##  mu: 0.000   0.000
##  cov:
##  1000.000    0.000
##  0.000   1000.000
##
##  sigma.sq IG hyperpriors shape=2.00000 and scale=2.00000
##  tau.sq IG hyperpriors shape=2.00000 and scale=0.10000
##  phi Unif hyperpriors a=3.00000 and b=30.00000
## ----------------------------------------------------
##      Sampling
## ----------------------------------------------------
## Sampled: 500 of 2000, 25.00%
## Report interval Metrop. Acceptance rate: 44.60%
## Overall Metrop. Acceptance rate: 44.60%
## ----------------------------------------------------
## Sampled: 1000 of 2000, 50.00%
## Report interval Metrop. Acceptance rate: 41.60%
## Overall Metrop. Acceptance rate: 43.10%
## ----------------------------------------------------
## Sampled: 1500 of 2000, 75.00%
## Report interval Metrop. Acceptance rate: 42.00%
## Overall Metrop. Acceptance rate: 42.73%
## ----------------------------------------------------
## Sampled: 2000 of 2000, 100.00%
## Report interval Metrop. Acceptance rate: 41.00%
## Overall Metrop. Acceptance rate: 42.30%
## ----------------------------------------------------
```

```
# model with second set of priors
m.2 <- spLM(y~X-1, coords=coords, starting=starting,
            tuning=tuning, priors=priors.2, cov.model=cov.model,
            n.samples=n.samples, n.report=n.report)
```

```
## ----------------------------------------
##  General model description
## ----------------------------------------
## Model fit with 100 observations.
##
## Number of covariates 2 (including intercept if specified).
##
## Using the exponential spatial correlation model.
##
## Number of MCMC samples 2000.
##
## Priors and hyperpriors:
##  beta flat.
##  sigma.sq IG hyperpriors shape=2.00000 and scale=2.00000
##  tau.sq IG hyperpriors shape=2.00000 and scale=0.10000
##  phi Unif hyperpriors a=3.00000 and b=30.00000
## ----------------------------------------------------
##      Sampling
## ----------------------------------------------------
## Sampled: 500 of 2000, 25.00%
## Report interval Metrop. Acceptance rate: 46.20%
## Overall Metrop. Acceptance rate: 46.20%
## ----------------------------------------------------
## Sampled: 1000 of 2000, 50.00%
## Report interval Metrop. Acceptance rate: 42.40%
## Overall Metrop. Acceptance rate: 44.30%
## ----------------------------------------------------
## Sampled: 1500 of 2000, 75.00%
## Report interval Metrop. Acceptance rate: 43.20%
## Overall Metrop. Acceptance rate: 43.93%
## ----------------------------------------------------
## Sampled: 2000 of 2000, 100.00%
## Report interval Metrop. Acceptance rate: 37.40%
## Overall Metrop. Acceptance rate: 42.30%
## ----------------------------------------------------
```

```
par(mfrow=c(2,2))
ts.plot(m.1$p.theta.samples[,1],main="sigma sq",ylab="",
        xlim=c(100,nrow(m.1$p.theta.samples)),ylim=c(0,4))
ts.plot(m.1$p.theta.samples[,2],main="tau sq",ylab="",
        xlim=c(100,nrow(m.1$p.theta.samples)),ylim=c(0,1))
ts.plot(m.1$p.theta.samples[,3],main="phi",ylab="",
        xlim=c(50,nrow(m.1$p.theta.samples)))

burn.in <- 0.5*n.samples

# recover beta and spatial random effects
m.1 <- spRecover(m.1, start=burn.in, verbose=FALSE)
m.2 <- spRecover(m.2, start=burn.in, verbose=FALSE)
```

```
# quantiles used in composition sampling
round(summary(m.1$p.theta.recover.samples)$quantiles[,c(3,1,5)],2)
```

```
##            50% 2.5% 97.5%
## sigma.sq  1.68 1.11  2.41
## tau.sq    0.07 0.02  0.35
## phi      10.13 5.39 15.55
```

```
round(summary(m.2$p.theta.recover.samples)$quantiles[,c(3,1,5)],2)
```

```
##            50% 2.5% 97.5%
## sigma.sq  1.60 1.07  2.84
## tau.sq    0.05 0.02  0.34
## phi      10.55 5.67 16.74
```

```
# quantiles of regression coefficients posterior samples
round(summary(m.1$p.beta.recover.samples)$quantiles[,c(3,1,5)],2)
```

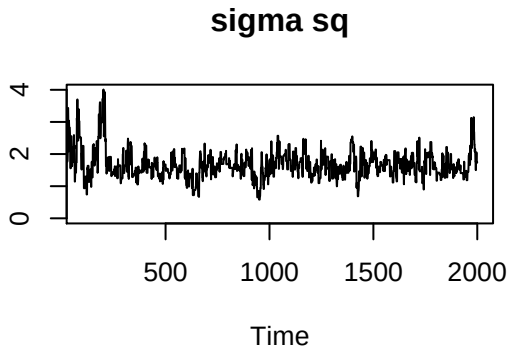
```
##     50% 2.5% 97.5%
## X1 1.64 0.98  2.24
## X2 4.91 4.73  5.10
```

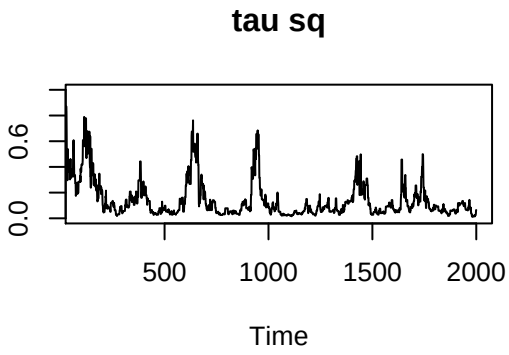
```
round(summary(m.2$p.beta.recover.samples)$quantiles[,c(3,1,5)],2)
```

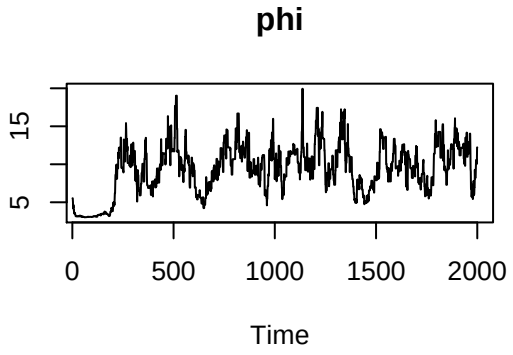
```
##     50% 2.5% 97.5%
## X1 1.66 0.99  2.32
## X2 4.90 4.72  5.09
```

```
# spatial random effects posterior samples
# rows are locations' random effects
# columns are posterior samples
m.1.w.summary <- summary(mcmc(t(m.1$p.w.recover.samples)))$quantiles[,c(3,1,5)]
m.2.w.summary <- summary(mcmc(t(m.2$p.w.recover.samples)))$quantiles[,c(3,1,5)]

par(mfrow=c(1,1))
```

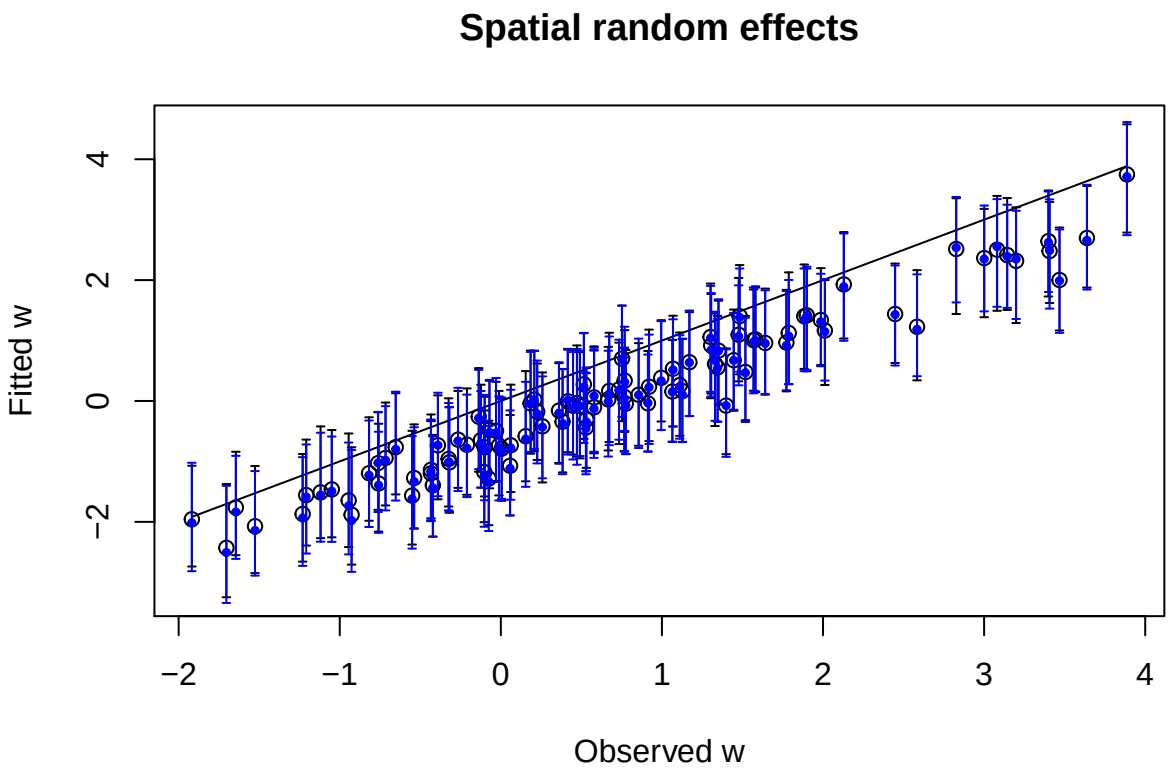







```
#plot for spatial random effects
plot(w, m.1.w.summary[,1], xlab="Observed w", ylab="Fitted w",
     xlim=range(w), ylim=range(m.1.w.summary), main="Spatial random effects")
#draw arrows for model 1 random effects
arrows(w, m.1.w.summary[,1], w, m.1.w.summary[,2], length=0.02, angle=90)
arrows(w, m.1.w.summary[,1], w, m.1.w.summary[,3], length=0.02, angle=90)
lines(range(w), range(w))

#add arrows for model 2 random effects
points(w, m.2.w.summary[,1], col="blue", pch=19, cex=0.5)
arrows(w, m.2.w.summary[,1], w, col="blue", m.2.w.summary[,2], length=0.02, angle=90)
arrows(w, m.2.w.summary[,1], w, col="blue", m.2.w.summary[,3], length=0.02, angle=90)
```



Predictive process models

The `spLM()` function can also implement predictive process models by specifying the **knots** command with a vector of length 2 or 3, with the first and second elements recording the number of columns and rows in the knot grid. The third, optional, element sets the offset of the outermost knots from the extent of the coords argument. Alternatively, the **knots** command can be specified with an $r \times 2$ matrix of knot locations.

Using the same modeling scenario as before, the predictive process model allows the user to select the number of locations, r, which yields $S^* = \{s_1^*, s_2^*, \ldots, s_r^2\}$. Then define $\tilde{w}(\boldsymbol{s}) = E[w(\boldsymbol{s})|w(\boldsymbol{s}_i^*), i = 1, 2, \ldots, r]$. We call $\tilde{w}(\boldsymbol{s})$ a predictive process.

This process is also a form of the hierarchical linear mixed model discussed before. If we let $\boldsymbol{\alpha}$ be a $r \times 1$ random vector with $w(\boldsymbol{s}_i^*)$ as its entries, then let $\boldsymbol{D}(\boldsymbol{\theta}) = \tau^2\boldsymbol{I}$, $\boldsymbol{K}(\boldsymbol{\theta}) = \boldsymbol{C}^*(\boldsymbol{\theta})$ and $\boldsymbol{Z}(\boldsymbol{\theta}) = C(\boldsymbol{\theta})'\boldsymbol{C}^*(\boldsymbol{\theta})^{-1}$, where $C(\boldsymbol{\theta})'$ is an $n \times r$ matrix whose entries are the covariances between $w(\boldsymbol{s}_i)$'s and $w(\boldsymbol{s}_j^*)$'s, and $\boldsymbol{C}^*(\boldsymbol{\theta})^{-1}$ is the $r \times r$ covariance matrix of the $w(\boldsymbol{s}_i^*)$'s.

```
############################
##Predictive process model
############################
m.1 <- spLM(y~X-1, coords=coords, knots=c(6,6,0.1), starting=starting,
            tuning=tuning, priors=priors.1, cov.model=cov.model,
            n.samples=n.samples, n.report=n.report)
```

```
## ----------------------------------------
##  General model description
## ----------------------------------------
## Model fit with 100 observations.
```

```
##
## Number of covariates 2 (including intercept if specified).
##
## Using the exponential spatial correlation model.
##
## Using modified predictive process with 36 knots.
##
## Number of MCMC samples 2000.
##
## Priors and hyperpriors:
##  beta normal:
##  mu: 0.000   0.000
##  cov:
##  1000.000    0.000
##  0.000   1000.000
##
##  sigma.sq IG hyperpriors shape=2.00000 and scale=2.00000
##  tau.sq IG hyperpriors shape=2.00000 and scale=0.10000
##  phi Unif hyperpriors a=3.00000 and b=30.00000
## -------------------------------------------------
##      Sampling
## -------------------------------------------------
## Sampled: 500 of 2000, 25.00%
## Report interval Metrop. Acceptance rate: 48.40%
## Overall Metrop. Acceptance rate: 48.40%
## -------------------------------------------------
## Sampled: 1000 of 2000, 50.00%
## Report interval Metrop. Acceptance rate: 43.20%
## Overall Metrop. Acceptance rate: 45.80%
## -------------------------------------------------
## Sampled: 1500 of 2000, 75.00%
## Report interval Metrop. Acceptance rate: 40.40%
## Overall Metrop. Acceptance rate: 44.00%
## -------------------------------------------------
## Sampled: 2000 of 2000, 100.00%
## Report interval Metrop. Acceptance rate: 46.00%
## Overall Metrop. Acceptance rate: 44.50%
## -------------------------------------------------
```

```
m.2 <- spLM(y~X-1, coords=coords, knots=c(6,6,0.1), starting=starting,
            tuning=tuning, priors=priors.2, cov.model=cov.model,
            n.samples=n.samples, n.report=n.report)
```

```
## ----------------------------------------
##  General model description
## ----------------------------------------
## Model fit with 100 observations.
##
## Number of covariates 2 (including intercept if specified).
##
## Using the exponential spatial correlation model.
##
## Using modified predictive process with 36 knots.
##
## Number of MCMC samples 2000.
```

```
##
## Priors and hyperpriors:
##  beta flat.
##  sigma.sq IG hyperpriors shape=2.00000 and scale=2.00000
##  tau.sq IG hyperpriors shape=2.00000 and scale=0.10000
##  phi Unif hyperpriors a=3.00000 and b=30.00000
## ------------------------------------------------
##      Sampling
## ------------------------------------------------
## Sampled: 500 of 2000, 25.00%
## Report interval Metrop. Acceptance rate: 41.20%
## Overall Metrop. Acceptance rate: 41.20%
## ------------------------------------------------
## Sampled: 1000 of 2000, 50.00%
## Report interval Metrop. Acceptance rate: 35.80%
## Overall Metrop. Acceptance rate: 38.50%
## ------------------------------------------------
## Sampled: 1500 of 2000, 75.00%
## Report interval Metrop. Acceptance rate: 43.20%
## Overall Metrop. Acceptance rate: 40.07%
## ------------------------------------------------
## Sampled: 2000 of 2000, 100.00%
## Report interval Metrop. Acceptance rate: 35.80%
## Overall Metrop. Acceptance rate: 39.00%
## ------------------------------------------------
```

```
par(mfrow=c(2,2))
ts.plot(m.1$p.theta.samples[50:nrow(m.1$p.theta.samples),1],main="sigma sq",ylab="",
        xlim=c(50,nrow(m.1$p.theta.samples)))
ts.plot(m.1$p.theta.samples[50:nrow(m.1$p.theta.samples),2],main="tau sq",ylab="",
        xlim=c(50,nrow(m.1$p.theta.samples)))
ts.plot(m.1$p.theta.samples[50:nrow(m.1$p.theta.samples),3],main="phi",ylab="",
        xlim=c(50,nrow(m.1$p.theta.samples)))

burn.in <- 0.5*n.samples

round(summary(window(m.1$p.beta.samples, start=burn.in))$quantiles[,c(3,1,5)],2)
```

```
##     50% 2.5% 97.5%
## X1 1.63 0.77  2.53
## X2 4.92 4.70  5.15
```

```
round(summary(window(m.2$p.beta.samples, start=burn.in))$quantiles[,c(3,1,5)],2)
```

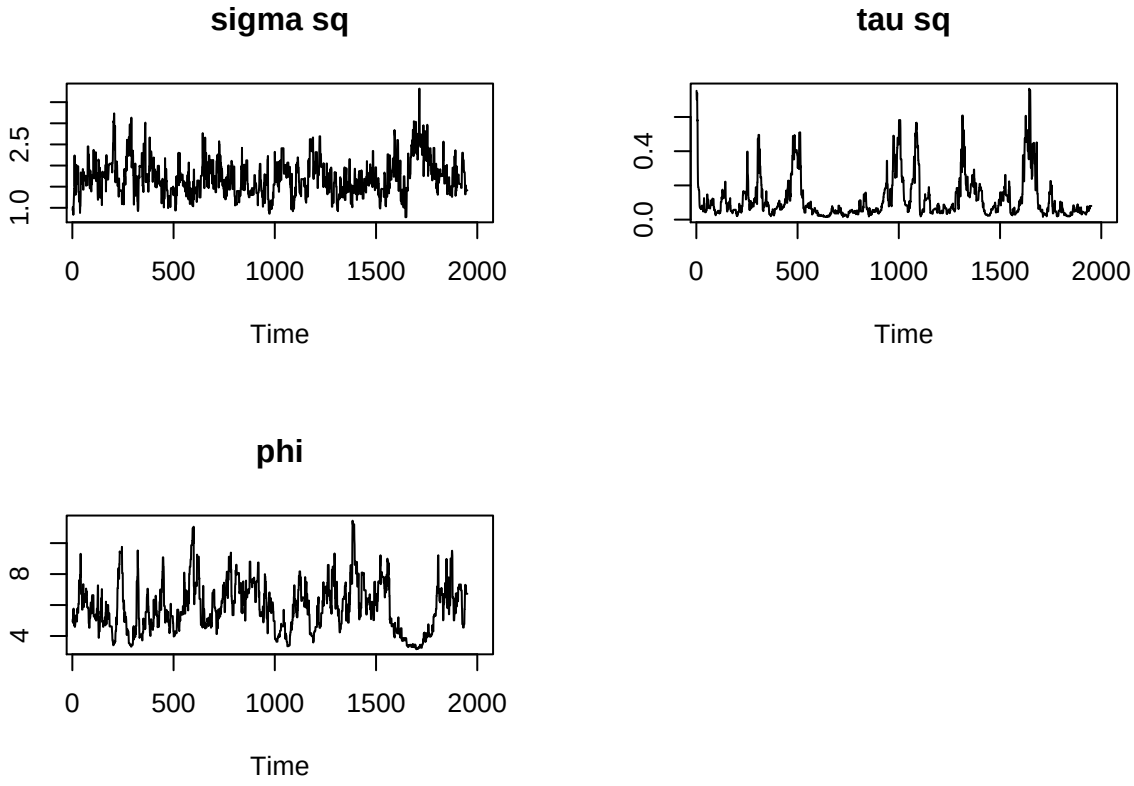
```
##     50% 2.5% 97.5%
## X1 1.62 0.69  2.56
## X2 4.92 4.70  5.15
```

```
round(summary(window(m.1$p.theta.samples, start=burn.in))$quantiles[,c(3,1,5)],2)
```

```
##           50% 2.5% 97.5%
## sigma.sq 1.69 1.01  2.70
## tau.sq   0.08 0.02  0.52
## phi      5.76 3.28  9.00
```

```
round(summary(window(m.2$p.theta.samples, start=burn.in))$quantiles[,c(3,1,5)],2)
```

```
##           50% 2.5% 97.5%
## sigma.sq 1.82 1.10  2.93
## tau.sq   0.05 0.02  0.29
## phi      5.46 3.39  9.20
```



Gaussian multivariate Bayesian spatial regression models

The `spBayes` package also extends its models to the multivariate cases. Consider m point-referenced outcomes modeled by

$$y_j(\mathbf{s}) = \mathbf{x}_j(\mathbf{s})'\boldsymbol{\beta}_j + w_j(\mathbf{s}) + \epsilon_j(\mathbf{s})$$

for $j = 1, 2, \ldots, m$. Here, $\mathbf{x}_j(\mathbf{s})$ is a $p_j \times 1$ vector of predictors for outcome j, $\boldsymbol{\beta}_j$ is a $p_j \times 1$ slope vector, and $w_j(\mathbf{s})$ and $\epsilon_j(\mathbf{s})$ are the spatial and random error processes, respectively. It is often assumed that the $\epsilon(\mathbf{s})$'s follow a zero-centered multivariate Normal distribution with zero mean and $m \times m$ dispersion matrix Ψ. The spatial variation is modeled with a $m \times 1$ Gaussian process $\mathbf{w}(\mathbf{s})$ with zero mean and cross-covariance matrix $\mathbf{C}_w(\mathbf{s}, \mathbf{t})$ where the entries are the covariance of $w_i(\mathbf{s})$ and $w_j(\mathbf{t})$. This cross-covariance matrix is specified as $\mathbf{C}_w(\mathbf{s}, \mathbf{t}) = \mathbf{AM}(\mathbf{s}, \mathbf{t})\mathbf{A}'$ where $\mathbf{A}$ is $m \times m$ lower triangular and $\mathbf{M}(\mathbf{s}, \mathbf{t})$ is $m \times m$ diagonal where the diagonals are spatial correlation functions.

Again, this model can be shown to be a special case of the original hierarchical linear mixed model. Suppose that the m outcomes have been observed in b locations, so y is $mb \times 1$. Let $\mathbf{X}$ be the $mb \times p$ matrix of predictors associated with $\mathbf{y}$, where $p = \sum_{j=1}^{m} p_j$ and $\boldsymbol{\beta}$ is $p \times 1$. Then the model is equivalent to the hierarchical model if $\mathbf{D}(\boldsymbol{\theta}) = \mathbf{I}_b \otimes \boldsymbol{\Phi}$, $\boldsymbol{\alpha}$ is $mb \times 1$, and $\mathbf{K}(\boldsymbol{\theta})$ is $mb \times mb$ partitioned into $m \times m$ blocks given by $\mathbf{AM}(\mathbf{s}, \mathbf{t})\mathbf{A}'$.

This model, as well as an analagous multivariate predictive process model, can be fitted with the `spMvLM()` function in the `spBayes` package.

In general, the function looks like

`spMvLM(formula,coords,starting,tuning,priors,cov.model,n.samples,n.report,...)`

where some commonly specified arguments are

- **formula**, a list of q symbolic regression model descriptions to be fit, where q is the number of outcomes at each location
- **coords**, an $mb \times 2$ matrix of the observation coordinates in $\mathbb{R}^2$
- **starting**, a list of tags where values are the parameter's starting values (parameters could be beta, A, phi, nu, L, or Psi)
- **tuning**, a list of tags where values are the parameter's corresponding variance for the Metropolis sampler Normal proposal distribution
- **priors**, a list with tags (could be beta.flat, beta.norm, K.iw, Psi.iw, Psi.ig, phi.unif, or nu.unif). Use Psi.iw if the non-spatial residual covariance matrix is assumed block diagonal, and otherwise, if it is assumed diagonal, then each of the q diagonal elements are assumed to follow an inverse-Gamma distribution and use Psi.ig. Hyperparameters for the cross-covariance matrix AA' K.iw are passed as a list of length 2. If Psi.ig is specified, the inverse-Gamma hyperparameters of the diagonal variance elements are passed with a list of length 2. The hyperparameters of the Uniform phi.unif and nu.unif are passed as a list of vectors
- **cov.model**, a keyword to specify the covariance function used to model the spatial dependence structure (could be exponential, matern, spherical, or gaussian)
- **n.samples**, the number of MCMC iterations
- **n.report**, an interval to report Metropolis acceptance and MCMC progress

Consider the following example for implementing this model.

```
rmvn <- function(n, mu=0, V = matrix(1)){
p <- length(mu)
if(any(is.na(match(dim(V),p)))){stop("Dimension problem!")}
D <- chol(V)
t(matrix(rnorm(n*p), ncol=p)%*%D + rep(mu,rep(n,p)))
}

set.seed(1)

# Generate some data
n <- 25 # number of locations
q <- 2 # number of outcomes at each location
nltr <- q*(q+1)/2 # number of triangular elements in the cross-covariance matrix

# matrix of observation coordinates for spMvLM()
coords <- cbind(runif(n,0,1), runif(n,0,1))

# Parameters for the bivariate spatial random effects
theta <- rep(3/0.5,q)

# Lower-triangular matrix for cross-covariance of Gaussian process
A <- matrix(0,q,q)
A[lower.tri(A,TRUE)] <- c(1,-1,0.25)
```

```
K <- A%*%t(A)

# dispersion matrix
Psi <- diag(0,q)

# calculate spatial covariance matrix
C <- mkSpCov(coords, K, Psi, theta, cov.model="exponential")

# Gaussian spatial process
w <- rmvn(1, rep(0,nrow(C)), C)

# w.1 and w.2 are every other element in w
w.1 <- w[seq(1,length(w),q)]
w.2 <- w[seq(2,length(w),q)]

# Covariate portion of the mean
x.1 <- cbind(1, rnorm(n))
x.2 <- cbind(1, rnorm(n))
# create a multivariate design matrix given q univariate design matrices
x <- mkMvX(list(x.1, x.2))

# parameters
B.1 <- c(1,-1)
B.2 <- c(-1,1)
B <- c(B.1, B.2)

# dispersion
Psi <- diag(c(0.1, 0.5))

# outcomes
y <- rnorm(n*q, x%*%B+w, diag(n)%x%Psi)

# outcomes based on every other element in y
y.1 <- y[seq(1,length(y),q)]
y.2 <- y[seq(2,length(y),q)]

# Call spMvLM
A.starting <- diag(1,q)[lower.tri(diag(1,q), TRUE)]
# number of MCMC iterations for spMvLM()
n.samples <- 1000

# tags with starting values for parameters for spMvLM()
starting <- list("phi"=rep(3/0.5,q), "A"=A.starting, "Psi"=rep(1,q))
# tags with Metropolis sampler variances for spMvLM()
tuning <- list("phi"=rep(1,q), "A"=rep(0.01,length(A.starting)), "Psi"=rep(0.01,q))
# tags with prior values
priors <- list("beta.Flat", "phi.Unif"=list(rep(3/0.75,q), rep(3/0.25,q)),
               "K.IW"=list(q+1, diag(0.1,q)), "Psi.ig"=list(c(2,2), c(0.1,0.1)))

# call spMvLM() function
m.1 <- spMvLM(list(y.1~x.1-1, y.2~x.2-1),
              coords=coords, starting=starting, tuning=tuning, priors=priors,
              n.samples=n.samples, cov.model="exponential", n.report=100)
```

```
## ----------------------------------------
##  General model description
## ----------------------------------------
## Model fit with 25 observations.
##
## Number of covariates 4 (including intercept if specified).
##
## Using the exponential spatial correlation model.
##
## Number of MCMC samples 1000.
##
## Priors and hyperpriors:
##  beta flat.
##
##  K IW hyperpriors df=3.00000, S=
##  0.100   0.000
##  0.000   0.100
##
##  Diag(Psi) IG hyperpriors
##      parameter   shape   scale
##      Psi[1,1]    2.0 0.10
##      Psi[2,2]    2.0 0.10
##
##  phi Unif hyperpriors
##      parameter   a   b
##      phi[1]      4.00000 12.00000
##      phi[2]      4.00000 12.00000
##
## -------------------------------------------------
##      Sampling
## -------------------------------------------------
## Sampled: 100 of 1000, 10.00%
## Report interval Metrop. Acceptance rate: 64.00%
## Overall Metrop. Acceptance rate: 64.00%
## -------------------------------------------------
## Sampled: 200 of 1000, 20.00%
## Report interval Metrop. Acceptance rate: 64.00%
## Overall Metrop. Acceptance rate: 64.00%
## -------------------------------------------------
## Sampled: 300 of 1000, 30.00%
## Report interval Metrop. Acceptance rate: 36.00%
## Overall Metrop. Acceptance rate: 54.67%
## -------------------------------------------------
## Sampled: 400 of 1000, 40.00%
## Report interval Metrop. Acceptance rate: 42.00%
## Overall Metrop. Acceptance rate: 51.50%
## -------------------------------------------------
## Sampled: 500 of 1000, 50.00%
## Report interval Metrop. Acceptance rate: 32.00%
## Overall Metrop. Acceptance rate: 47.60%
## -------------------------------------------------
## Sampled: 600 of 1000, 60.00%
## Report interval Metrop. Acceptance rate: 38.00%
## Overall Metrop. Acceptance rate: 46.00%
```

```
## --------------------------------------------------
## Sampled: 700 of 1000, 70.00%
## Report interval Metrop. Acceptance rate: 23.00%
## Overall Metrop. Acceptance rate: 42.71%
## --------------------------------------------------
## Sampled: 800 of 1000, 80.00%
## Report interval Metrop. Acceptance rate: 39.00%
## Overall Metrop. Acceptance rate: 42.25%
## --------------------------------------------------
## Sampled: 900 of 1000, 90.00%
## Report interval Metrop. Acceptance rate: 34.00%
## Overall Metrop. Acceptance rate: 41.33%
## --------------------------------------------------
## Sampled: 1000 of 1000, 100.00%
## Report interval Metrop. Acceptance rate: 27.00%
## Overall Metrop. Acceptance rate: 39.90%
## --------------------------------------------------
```

```
# burn in values
burn.in <- 0.75*n.samples

par(mfrow=c(3,3))
ts.plot(m.1$p.theta.samples[,1],ylab="",main="K[1,1]")
ts.plot(m.1$p.theta.samples[,2],ylab="",main="K[2,1]")
ts.plot(m.1$p.theta.samples[,3],ylab="",main="K[2,2]")
ts.plot(m.1$p.theta.samples[,4],ylab="",main="Psi[1,1]",
        xlim=c(400,nrow(m.1$p.theta.samples)),ylim=c(0,.1))
ts.plot(m.1$p.theta.samples[,5],ylab="",main="Psi[2,2]",
        xlim=c(400,nrow(m.1$p.theta.samples)),ylim=c(.01,.07))
ts.plot(m.1$p.theta.samples[,6],ylab="",main="phi[1]")
ts.plot(m.1$p.theta.samples[,7],ylab="",main="phi[2]")

# get regression coefficients and spatial random effects for model
m.1 <- spRecover(m.1, start=burn.in)
```

```
## --------------------------------------------------
##      Recovering beta and w
## --------------------------------------------------
## Sampled: 99 of 251, 39.44%
## Sampled: 199 of 251, 79.28%
```

```
# quantiles for theta and beta parameters
round(summary(m.1$p.theta.recover.samples)$quantiles[,c(3,1,5)],2)
```

```
##            50%  2.5% 97.5%
## K[1,1]    0.57  0.34  0.98
## K[2,1]   -0.55 -0.97 -0.35
## K[2,2]    0.61  0.38  1.09
## Psi[1,1]  0.02  0.02  0.04
## Psi[2,2]  0.02  0.01  0.03
## phi[1]   10.15  5.89 11.83
## phi[2]    6.28  4.04 10.44
```

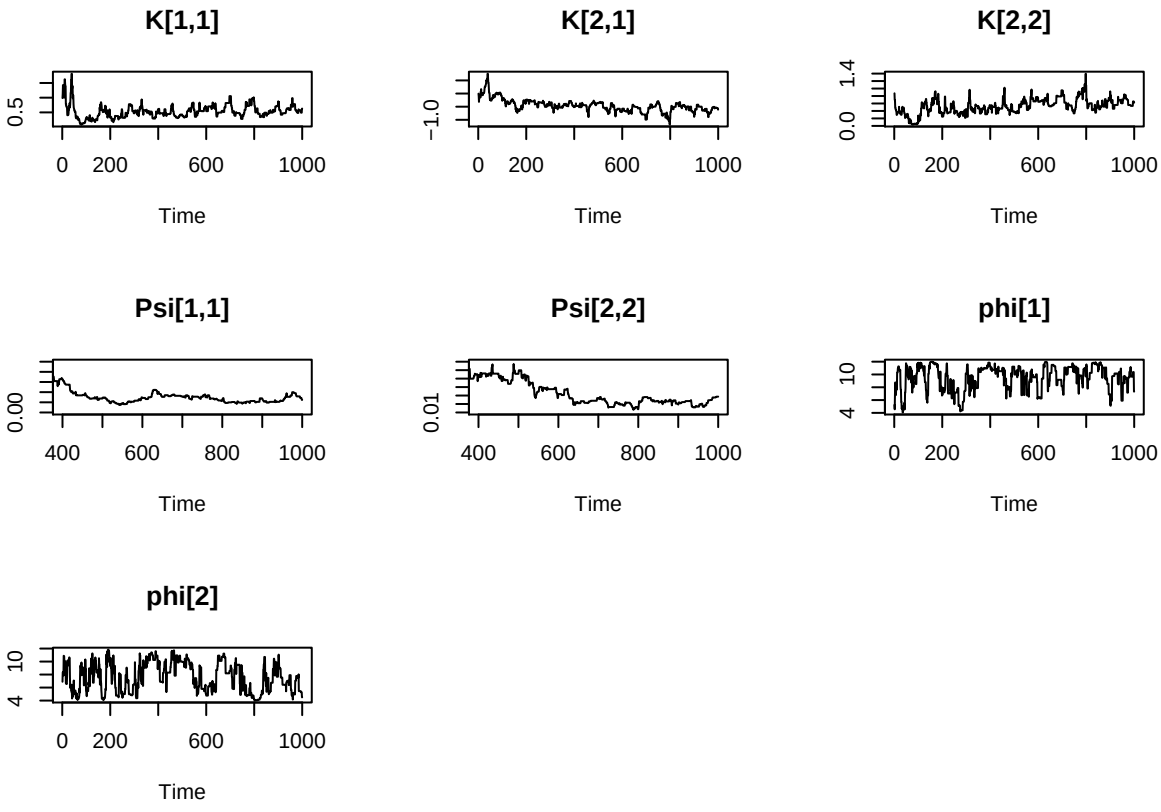
```
round(summary(m.1$p.beta.recover.samples)$quantiles[,c(3,1,5)],2)
```

```
##             50%  2.5% 97.5%
```

```
## x.11.mod1  0.97  0.58  1.45
## x.12.mod1 -0.98 -1.08 -0.87
## x.21.mod2 -0.89 -1.37 -0.49
## x.22.mod2  0.96  0.81  1.10
```

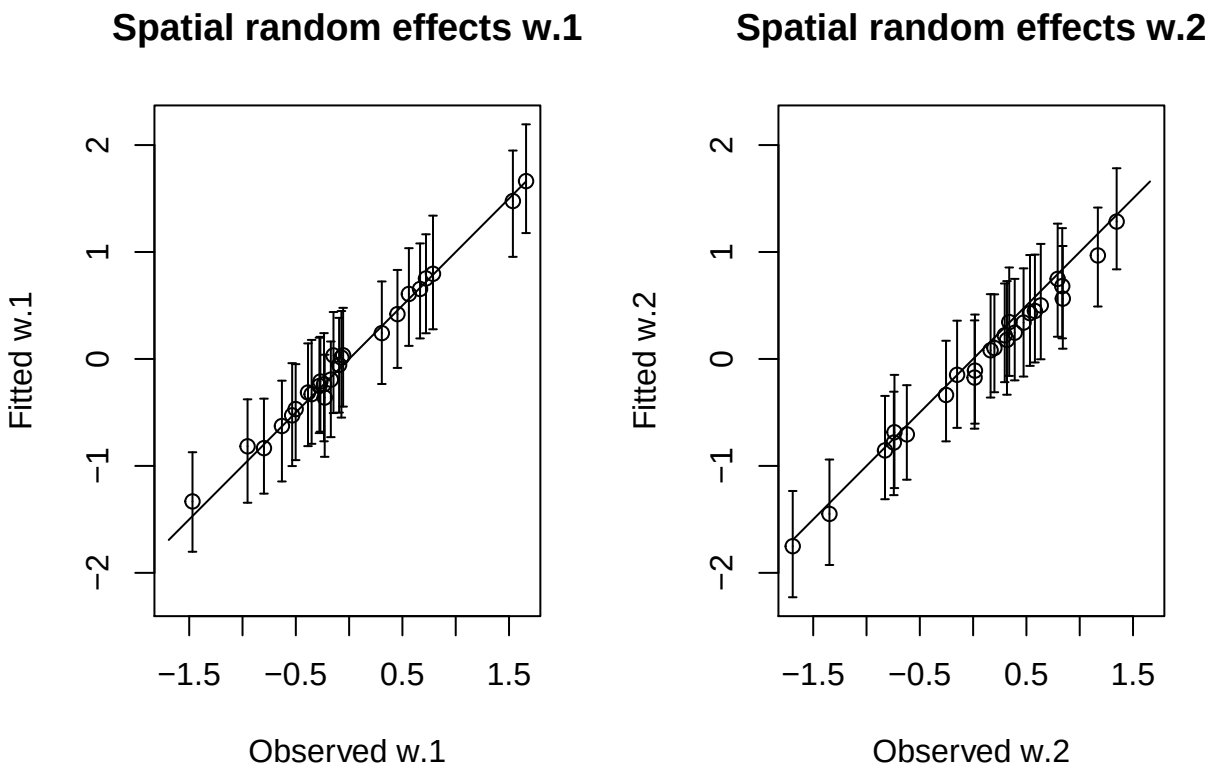
```r
# quantiles for spatial random effects
m.1.w.hat <- summary(mcmc(t(m.1$p.w.recover.samples)))$quantiles[,c(3,1,5)]
m.1.w.1.hat <- m.1.w.hat[seq(1, nrow(m.1.w.hat), q),]
m.1.w.2.hat <- m.1.w.hat[seq(2, nrow(m.1.w.hat), q),]

par(mfrow=c(1,1))
```




```r
# plot for w.1 spatial random effects
par(mfrow=c(1,2))
plot(w.1, m.1.w.1.hat[,1], xlab="Observed w.1", ylab="Fitted w.1",
     xlim=range(w), ylim=range(m.1.w.hat), main="Spatial random effects w.1")
arrows(w.1, m.1.w.1.hat[,1], w.1, m.1.w.1.hat[,2], length=0.02, angle=90)
arrows(w.1, m.1.w.1.hat[,1], w.1, m.1.w.1.hat[,3], length=0.02, angle=90)
lines(range(w), range(w))

# plot for w.2 spatial random effects
plot(w.2, m.1.w.2.hat[,1], xlab="Observed w.2", ylab="Fitted w.2",
     xlim=range(w), ylim=range(m.1.w.hat), main="Spatial random effects w.2")
arrows(w.2, m.1.w.2.hat[,1], w.2, m.1.w.2.hat[,2], length=0.02, angle=90)
arrows(w.2, m.1.w.2.hat[,1], w.2, m.1.w.2.hat[,3], length=0.02, angle=90)
lines(range(w), range(w))
```



Other models that `spBayes` can handle

This tutorial was not an exhaustive list of models that `spBayes` implements. Some topics that were not discussed here include dynamic spatio-temporal models (implemented in the function `spDynLM()`) and generalized linear models, both in the univariate and the multivariate cases (with functions `spGLM()` and `spMvGLM()`, respectively). In addition, the function `spDiag()` conducts model fit diagnostics on `spLM`, `spMvLM`, `spGLM`, or `spMvGLM` objects.

Bibliography

Finley AO, Banerjee S (2013). ***spBayes****: Univariate and Multivariate Spatial-temporal Modeling*. R package version 0.3-7, URL `http://CRAN.R-project.org/package=spBayes`.

Finley AO, Banerjee S, Gelfand AE (2013). "spBayes for Large Univariate and Multivariate Point-referenced Spatio-temporal Data Models". arXiv:1310.8192v1 [stat.CO].