# Spatio – Temporal Methods in Environmental Epidemiology: Supplementary Material for Chapter 11

Gavin Shaddick and James V Zidek

**Sampling from the full conditional of the overall mean parameter, $\beta_p$**

The regression coefficient $\beta_p$ is assumed to have a normal hyper-prior distribution

$$p(\beta_p) \sim N(m_\beta, s_\beta)$$

The full conditional distribution of $\beta_p$ can be written as

$$p(\beta_p|\theta_{..}, m_{..}, n_{..}, \Sigma_u, y_{...}) \propto \prod_{s=1}^{S}\prod_{t=1}^{T} p(y_{stp}|w_{tp}, m_{sp}, n_{sp}, \beta_p, \sigma^2_{u_{sp}}) \times p(\beta_p)$$

where $p(y_{spt}|\theta_{tp}, m_{sp}, n_{sp}, \beta_p, \sigma^2_{u_{sp}}) \sim N(\theta_{tp} + m_{sp} + I_{sp}n_{sp} + \beta_p, \sigma^2_{u_{sp}})$ for $s = 1, ...S$, $t = 1, .., P$ and $p = 1, ..., P$.

As a result the full conditionals distribution of $\beta_p$ is

$$\beta_p \sim N(\mu_{\beta_p}, s_{\beta_p})$$

where

$$s_{\beta_p} = \frac{1}{\sum_{s=1}^{S} \frac{T}{\sigma^2_{u_{sp}}}} \text{ and } \mu_\beta = \sum_{t=1}^{T} \frac{\sum_{s=1}^{S}(y_{stp} - \theta_{tp} - m_{sp} - I_{sp}(n_s))}{\sigma^2_{u_{sp}}} s_{\beta_p}$$

### *0.0.1 Sampling from the full conditional of the measurement error variance $\sigma^2_{u_{sp}}$*

The hyper-prior for the precision of the measurement error, $\sigma^{-2}_{u_{sp}}$ is a Gamma distribution, parameterisations can found in equation 3.2.

$$p(\sigma^{-2}_{u_{sp}}) \sim Gam(a_u, b_u)$$

therefore the full conditional can be written as

$$p(\sigma^{-2}_{u_{sp}}|\theta_{..}, m_{..}, n_{..}, \beta_{.}, y_{...}) \propto \prod_{s=1}^{S}\prod_{t=1}^{T} p(y_{stp}|\theta_{tp}, m_{sp}, n_{sp}, \beta_p, \sigma^2_{u_{sp}}) \times p(\sigma^{-2}_{u_{sp}})$$

$$\sigma^{-2}_{u_{sp}} \sim Gam\left(a_u + \frac{T}{2}, b_u + \frac{1}{2}\sum_{t=1}^{T}(Y_{stp} - \beta_p - \theta_{tp} - m_{sp} - I_{sp}(n_s))^2\right)$$

### *0.0.2 Sampling from the full conditional of the covariance matrix of temporal effects, $\Sigma_w$*

$$\begin{aligned} p(\Sigma_w|\theta_{..}, \alpha_{.}) &\propto \left\{\textstyle\prod_{t=2}^{T} MVN_P(\alpha_{.}\theta_{.(t-1)}, \Sigma_w)\right\} \times IW_p(Q_w, d) \\ &\propto IW\left(Q_w + \textstyle\sum_{t=2}^{T}(\theta_{.t} - \alpha_{.}\theta_{.(t-1)})(\theta_{.t} - \alpha_{.}\theta_{.(t-1)})', T - 1 + d\right) \end{aligned}$$

Thus an Inverse Wishart distribution is used to sample the temporal covariance matrix.

### 0.0.3 Sampling from the full conditional of the temporal effects, $\theta_{..}$

The prior distribution of $\theta_{.t}$ is a multivariate normal distribution as described in Section 4.1. The full conditional distributions for the temporal effects can be written as

$$p(\theta_{..}|y_{...},\beta_{.},\Sigma_u,\Sigma_w,\alpha_{.},m_{..},n_{..}) \propto$$
$$\propto \left\{\prod_{s=1}^{S}\prod_{t=1}^{T} MVN_P(\beta_{.}+\theta_{.t}+m_{s\cdot}+I_s.n_{s\cdot},\Sigma_u)\right\} \times \left\{\prod_{t=2}^{T} MVN_P(\alpha_{.}\theta_{.(t-1)},\Sigma_w)\right\}$$
$$\propto \exp\left\{-\tfrac{1}{2}\sum_{s=1}^{S}\sum_{t=1}^{T}(y_{s.t}-(\beta.+\theta_{.t}+m_{s}.+I_s.n_{s}.))\Sigma_u^{-1}(y_{s.t}-(\beta.+\theta_{.t}+m_{s}.+I_s.n_{s}.))'\right\}$$
$$\times \exp\left\{-\tfrac{1}{2}\sum_{t=2}^{T}(\theta_{.t}-\alpha_{.}\theta_{.(t-1)})\Sigma_w^{-1}(\theta_{.t}-\alpha_{.}\theta_{.(t-1)})'\right\}$$
$$\propto MVN_P(\mu_{\theta_t},S_{\theta_t}), \text{ where}$$

$S_{\theta_t} = \left(\Sigma_u^{-1}+\alpha_{.}\Sigma_w^{-1}\alpha_{.}'\right)^{-1}$ for $t=1$,

$\left(\Sigma_u^{-1}+\Sigma_w^{-1}+\alpha_{.}\Sigma_w^{-1}\alpha_{.}'\right)^{-1}$ for $t=2,...,T-1$,

$\left(\Sigma_u^{-1}+\Sigma_w^{-1}\right)^{-1}$ for $t=T$,
$\mu_{\theta_t} = S_{\theta_t}\left(\sum_{s=1}^{S}(y_{s.t}-(\beta.+\theta_{.t}+m_{s}.+I_s.n_{s}.))\Sigma_u^{-1}+\theta_{.(t+1)}\Sigma_w^{-1}\alpha_{.}\right)$ for $t=1$,
$S_{\theta_t}\left(\sum_{s=1}^{S}(y_{s.t}-(\beta.+\theta_{.t}+m_{s}.+I_s.n_{s}.))\Sigma_u^{-1}+(\theta_{.(t-1)}+\theta_{.(t+1)})\Sigma_w^{-1}\alpha_{.}\right)$ for $t=2,...,T-1$,
$S_{\theta_t}\left(\sum_{s=1}^{S}(y_{s.t}-(\beta.+\theta_{.t}+m_{s}.+I_s.n_{s}.))\Sigma_u^{-1}+\theta_{.(t-1)}\Sigma_w^{-1}\alpha_{.}\right)$ for$t=T$,

### 0.0.4 Sampling from the full conditional of the $\alpha_{.}$ parameter

The $\alpha_p$ parameters, $p=1,2,...,P$, have Uniform distributions as their priors and will not have a full conditional of closed form. In such cases, a Metropolis–with–Gibbs algorithm can be used with proposed value of $\alpha_p$ accepted with probability

$$c = min\left[1,\frac{p(\theta_{p}.|\alpha_p',\Sigma_w)}{p(\theta_{p.}|\alpha_p^c,\Sigma_w)}\right]$$

where $\alpha_p'$ is the proposed value and $\alpha_p^c$ the current value of $\alpha_p$ for $p=1,...,P$. As proposed value, $\alpha_p'$, a random value from the prior distribution, $Unif(-1,1)$, is used.

### 0.0.5 Sampling from the full conditional distribution of the spatial parameters for the background process, $\sigma_{mp}^{-2},\phi_{mp}$

There are two parameters that controls the spatial process of the data, $\sigma_{mp}^2$ is the between site spatial variance and the parameter $\phi_{mp}$ controls the strength of the correlation between the sites.

The hyper-prior for the between site precision, $\sigma_{mp}^{-2}$ is a Gamma distribution

$$p(\sigma_{mp}^{-2}) \sim Gam(a_{mp}, b_{mp})$$

$$p(\sigma_{mp}^{-2}|m_{.p}) \propto p(m_{.}p|\sigma_{mp}^{2}, \phi_{mp}) \times p(\sigma_{mp}^{-2})$$

where $p(m_{.}|\sigma_{m}^{2}, \phi_{m}) \sim MVN_S(0_S, \sigma_{m}^{2}\Sigma_{m})$. As a result the full conditional distribution is

$$\sigma_{mp}^{-2} \sim Gam\left(a_{mp} + \frac{S}{2}, b_{mp} + \frac{1}{2} m_{.p}\Sigma_{mp}m_{.p}^{'}\right)$$

The parameter $\phi_{mp}$ does not the have full conditional distribution available in closed form, so the Metropolis-Hasting algorithm is used with proposed values from the range $(a_{\phi}, b_{\phi})$, since the prior distribution of $\phi_{mp}$ is $Unif(a_{\phi}, b_{\phi})$. The proposed values of $\phi_{mp}$ is accepted with probability

$$c = min\left[1, \frac{p(m_{.p}|\phi_{mp}', \sigma_{mp}^{2})}{p(m_{.p}|\phi_{mp}^{c}, \sigma_{mp}^{2})}\right]$$

where $\phi_{mp}'$ is the proposed value and $\phi_{mp}^{c}$ the current value of $\phi_{mp}$.

### 0.0.6 *Sampling from the full conditional distribution of spatial effects, $m_{.p}$*

The spatial effects have a zero mean multivariate normal distribution as a prior distribution

$$p(m_{.p}|\sigma_{mp}^{2}, \phi_{mp}) \sim MVN_S(0_S, \sigma_{mp}^{2}\Sigma_{mp})$$

The assumption that the spatial effects has zero mean prior distribution is valid since we have the $\beta$ parameter in the model. This prior distribution is controlled by algorithms initial values that will be chosen for parameters $(\sigma_{mp}^{2}, \phi_{mp})$.

The full conditional distribution of $m_{.p}$ can be written as

$$p(m_{.p}|\sigma_{up}^{2}, n_{.p}, \sigma_{mp}^{2}, \phi_{mp}, \theta_{.p}, y_{..p}) \propto \prod_{t=1}^{T}\prod_{s=1}^{S} p(y_{stp}|\theta_{tp}, m_{sp}, \beta, n_{sp}, \sigma_{up}^{2}) \times p(m_{.p}|\sigma_{mp}^{2}, \phi_{mp})$$

As a result the full conditional of $m_{.p}$ can be written in two forms, one for single updating and one for block updating. The single updating full conditional distribution is

$$m_{sp} \sim N(\mu_{m_{sp}}, s_{m_{sp}})$$

where

$$s_{m_{sp}} = \frac{1}{T\sigma_{up}^{-2} + \sigma_{mp}^{-2}} \text{ and } \mu_{m_{sp}} = \sum_{t=1}^{T}(y_{spt} - \beta_p - I_{sp}n_{sp})\sigma_{up}^{-2}s_{m_s}$$

The block updating posterior is given by

$$m_{.p} \sim MVN_S\left(\sum_{t=1}^{T}(y_{.}tp - \beta_p - I_{sp}n_{.p}))\Sigma_u^{-1}s_m, s_m\right)$$

where

$$s_m = \left(T\Sigma_u^{-1} + \left(\sigma_{mp}^2\Sigma_{mp}\right)^{-1}\right)^{-1}$$

where $\Sigma_u$ is a diagonal matrix.

#### *0.0.6.1 Sampling from the full conditional distribution of the spatial parameters for the additional process,* $\sigma_{np}^{-2}, \phi_{np}$

The additional spatial process is independent from the background spatial process but the posterior distributions of its two parameters have similar form, $\sigma_{np}^2$ is the between site spatial variance of the specific group and the parameter $\phi_{np}$ controls the strength of the correlation between the sites.

$$p(\sigma_{np}^{-2}) \sim Gam(a_{np}, b_{np})$$

$$p(\sigma_{np}^{-2}|n_{.p}) \propto p(n_{.p}|\sigma_{np}^2, \phi_{np}) \times p(\sigma_{np}^{-2})$$

where $p(n_{.p}|\sigma_{np}^2, \phi_{np}) \sim MVN_{S^*}(0_{S^*}, \sigma_{np}^2\Sigma_{np})$. As a result the full conditional distribution is

$$\sigma_{np}^{-2} \sim Gam\left(a_{np} + \frac{S^*}{2}, b_{np} + \frac{1}{2}n_{.p}\Sigma_{np}n_{.p}^{'}\right)$$

The parameter $\phi_{np}$ does not the have full conditional distribution available in closed form, so the Metropolis-Hasting algorithm is used with proposed values from the range $(a_\phi, b_\phi)$, since the prior distribution of $\phi_{np}$ is $Unif(a_\phi, b_\phi)$. The proposed values of $\phi_{np}$ is accepted with probability

$$c = min\left[1, \frac{p(n_{.p}|\phi_{np}', \sigma_{np}^2)}{p(n_{.p}|\phi_{np}^c, \sigma_{np}^2)}\right]$$

where $\phi_{np}'$ is the proposed value and $\phi_{np}^c$ the current value of $\phi_{np}$.

### 0.0.7 *Sampling from the full conditional distribution of spatial effects, $n_{.p}$*

The spatial effects have a zero mean multivariate normal distribution as a prior distribution

$$p(n_{.p}|\sigma^2_{np}, \phi_{np}) \sim MVN_{S^*}(0_{S^*}, \sigma^2_{np}\Sigma_{np})$$

The assumption that the spatial effects has zero mean prior distribution is valid since we have the $\beta_p$ parameter in the model which represents the overall mean of this process. This prior distribution is controlled by algorithm initial values that will be chosen for parameters $(\sigma^2_{np}, \phi_{np})$.

The full conditional distribution of $n_{.p}$ can be written as

$$p(n_{.p}|\beta_p, \sigma^2_u, m_{.p}\sigma^2_{np}, \phi_{np}, \theta_{.p}y_{..p}) \propto \prod_{t=1}^{T}\prod_{s\in S^*} p(y_{stp}|m_{sp}, \beta_p, n_{sp}, \theta_{tp}, \sigma^2_u) \times p(n_{.p}|\sigma^2_{np}, \phi_{np})$$

As a result the posterior distribution of $n_{.p}$ can be written in two forms, one for single updating and one for block updating. The single updating full conditional distribution is

$$n_{sp} \sim N(\mu_{n_{sp}}, s_{n_{sp}})$$

where

$$s_{n_{sp}} = \frac{1}{T\sigma_u^{-2} + \sigma_{np}^{-2}} \text{ and } \mu_{n_{sp}} = (y_{spt} - \beta_p - m_{sp} - \theta_{tp})\sigma^{-2}_{up} s_{n_{sp}}$$

The block updating posterior is given by

$$n_{.p} \sim MVN_{S^*}\left(s_{np}\sum_{t=1}^{T}(y_{t.p} - \beta_p - m_{.p} - \theta_{tp})\Sigma^{-1}_{up}, s'_{np}\right)$$

where

$$s_{np} = \left(T\Sigma^{-1}_{up} + \left(\sigma^2_{np}\Sigma_{np}\right)^{-1}\right)^{-1}$$

where $\Sigma_u$ is a diagonal matrix.

### 0.0.8 *Implementation using WinBUGS*

```
model {
# beginning of t loop
              for (t in 2:(n-1)) {

                              for (poll in 1:4) {
                                     for (site in 1:8) {
```

```
# 4x8 ys
    arise
    from the
    4
    underlying
     thetas,
    & 8 site
    effects &

    measurement
     error
y.mat[t,poll,site] ~ dnorm(
    mean.poll.site[t,poll,
    site],tau.v[poll,site])
    mean.poll.
        site[t,
        poll,site
        ] <-
        theta[t,
        poll]  +m
        .adj[poll
        ,site]
           + temp
        .effect[t
        ,poll]
# end of site
     loop
}

# all of  the

    underlying
     thetas
    are
    averages
    of the
    two
    neighbours

 tmp.theta[t,
     poll]<-
     (theta[t
     -1,poll
     ]+theta[
     t+1,poll
     ])/2
        for (
            poll2

            in

            1:4)

            {
```

```
Sigma.p.like[t,poll,poll2] <- (theta[t,poll]-theta[t-1,poll]) * (theta[t,poll2]-theta[t-1,poll2])

temp.effect[t,poll] <- (beta.temp[
```

```
                                                      poll]*
                                                      temp.adj
                                                      [t])

                                                      +(beta.
                                                      temp2[
                                                      poll]*
                                                      temp2.
                                                      adj[t])
# end of poll loop
                                                                          }

                                      theta[t,1:4] ~ dmnorm(tmp.
                                          theta[t,1:4],Sigma.p2
                                          [1:4,1:4])
# temp effects
temp.adj[t]<-temp[t]-temp.bar
temp2.adj[t]<- temp2[t]-temp2.bar

# end of t loop
                                                          }


# Set up the priors for 'edges' of the underlying AR process
     for theta
      theta[1,1:4]~dmnorm(theta[2,1:4],Sigma.p[1:4,1:4])
      theta[n,1:4]~dmnorm(theta[n-1,1:4],Sigma.p[1:4,1:4])

# Set up the priors for the 'edges' of the y's
               for (poll in 1:4) {
                               for (site in 1:8) {
                       y.mat[1,poll,site] ~ dnorm(theta[1,
                           poll],tau.v[poll,site])
                       y.mat[n,poll,site] ~ dnorm(theta[n,
                           poll],tau.v[poll,site])
                                                                          }


                                                 }

# Likelihoods for the 'edges'

               for (poll1 in 1:4) {
                       for (poll2 in 1:4) {
                               Sigma.p.like[1,poll1,poll2]<-
                                   0
                               Sigma.p.like[n,poll1,poll2]<-
                                   (theta[n,poll1]-theta[n
                                   -1,poll1]) *  (theta[n,
                                   poll2]-theta[n-1,poll2])
                                                                   }


                                                        }
```

For the parameters of the Wishart distribution, $d$ was chosen to be equal to four, the dimension of $\Sigma_P$;

*D* was then chosen so that the diagonals of the expected value ($D/d$) represent a 10% coefficient of variation. The off-diagonals were taken to be zero.

```
# Likelihoods for the Wishart parameter
# initial values of the priors
R[1,1] <- 0.2
R[1,2] <- 0.01
R[1,3] <- 0.01
R[1,4] <- 0.01

R[2,2] <- 0.2
R[2,1] <- 0.01
R[2,3] <- 0.01
R[2,4] <- 0.01

R[3,3] <- 0.2
R[3,1] <- 0.01
R[3,2] <- 0.01
R[3,4] <- 0.01

R[4,4] <- 0.2
R[4,1] <- 0.01
R[4,2] <- 0.01
R[4,3] <- 0.01

for (poll1 in 1:4) {
       for (poll2 in 1:4) {
               Rn[poll1,poll2] < - R[poll1,poll2] + sum(
                   Sigma.p.like[1:n,poll1,poll2])
                                       }
                                                 }

K <-2
Kn <- K+ n

       Sigma.p[1:4,1:4] ~ dwish(Rn[1:4,1:4],Kn)

# mutiply the precision by 2, as variance needs to be
    divided by 2 (average of 2 thetas)

for (i in 1:4){
                       for (j in 1:4){
                                                         Sigma
                                                            .
                                                            p2
                                                            [
                                                            i
                                                            ,
                                                            j
                                                            ]

                                                            <-

                                                            Sigma
                                                            .
                                                            p
```

```
                                                       [
                                                       i
                                                       ,
                                                       j
                                                       ]
                                                       *
                                                       2
                                                   }
                       }

# put in the inverse stuff here, for the sd matrix   /
    correlation

for (i in 1:4){
                       for (j in 1:4){
                                                   var.p
                                                       [
                                                       i
                                                       ,
                                                       j
                                                       ]

                                                       <-

                                                       inverse
                                                       (

                                                       Sigma
                                                       .
                                                       p
                                                       [,],
                                                       i
                                                       ,
                                                       j
                                                       )
                                                   }
                       }

for (poll in 1:4) {
  sigma.theta[poll] <- sqrt(var.p[poll,poll])
               for (poll2 in 1:4) {
                       corr.theta[poll,poll2] <-  var.p[poll
                           ,poll2] / (sigma.theta[poll]*
                           sigma.theta[poll2])
                                              }
                                       }




# Set up the pollutant/site specific observation precisions
       for (poll in 1:4) {
               for (site in 1:8) {
                   tau.v[poll,site] ~ dgamma(1,0.01)
```

```
                   sigma.v[poll,site] <-1/sqrt(tau.v[poll,
                       site])
                                                          }
                                           }

# Set up the priors for the site specific parameters
#    set them up as spatial.exp prior, different for each
     site
for (poll in 1:4) {
m[poll,1:8] ~ spatial.exp(xcoords[],ycoords[],tau.m[poll],
    phi1[poll],phi2)
}

for (poll in 1:4) {
sigma.m[poll] <- 1/sqrt(tau.m[poll])
}

# and to constrain the sums to be zero - CHECK for quicker
    approach
for (poll in 1:4) {
for (site in 1:8) {
m.adj[poll,site] <- m[poll,site]-mean(m[poll,1:8])
}
}

phi2 <- 1
for (poll in 1:4) {
phi1[poll]~ dunif(0.0026,0.115)
tau.m[poll] ~ dgamma(1,0.01)
}



# priors for temp
temp.bar<-mean(temp[])
temp2.bar <- mean(temp2[])
for (poll in 1:4) {
beta.temp[poll] ~ dnorm(0,0.001)
beta.temp2[poll] ~ dnorm(0,0.001)
}

# Calculate the mean and sd of the thetas
for (poll in 1:4) {
                  mean.theta[poll] <- mean(theta[1:n,poll])
                  sd.theta[poll] <-sd(theta[1:n,poll])
                                           }




# end of model
}
```

# References