Model

Nicholas Nagle

Load the data

```
library(dplyr)
library(Matrix)
library(INLA)

# Load the precip data:
load('~/Dropbox/git_root/climate-bayes/data/anomaly.Rdata')
# Load the tree ring data:
load('~/Dropbox/git_root/climate-bayes/data/itrdb_meta.Rdata')
# Load the spatial matrices
load('~/Dropbox/git_root/climate-bayes/data/spatial_fields.Rdata')

# Convert species_code to a facter
tree.meta <- tree.meta %>% mutate(species = as.factor(species_code))

# Count the number of distinct species:
num.species <- nlevels(tree.meta$species)

# Create the design matrix, which has ntrees rows and num.species columns
tree.design <- sparse.model.matrix(data=tree.meta, ~species -1 )


# alpha_t is a 580 x 1 vector
# y_t is a 914 tree plus 106 precip locs = 1020 x 1 vector

# beta has
 # 18 intercepts for each species
 # 18 slopes for each species
 # 1 mean for alpha

# Z_grid = 106 x 580 matrix
# Z_tree is a 914 x 580 matrix: diag(beta) %*% A_tree

## Convert the monthly data to annual data:
anom.df <- anom.df %>% mutate(year=substr(time, 1, 4)) %>% group_by(year, lat, lon) %>%
  summarise(precip = mean(precip), SID=mean(SID)) %>% arrange(year, SID) %>% ungroup
nobs.df <- nobs.df %>% mutate(year=substr(time, 1, 4)) %>% group_by(year, lat, lon) %>%
  summarise(nobs = mean(nobs), SID=mean(SID)) %>% arrange(year, SID) %>% ungroup
```

Process level

The true temperature field (sampled at a finite number of points) $\boldsymbol{\alpha}_t$ follows a Vector Autoregressive process:

$$\boldsymbol{\alpha}_t = \mu\mathbf{1} + \rho(\boldsymbol{\alpha}_{t-1} - \mu\mathbf{1}) + \boldsymbol{\epsilon}_t$$

$$\boldsymbol{\alpha}_t = \mu(1-\rho)\mathbf{1} + \rho\boldsymbol{\alpha}_{t-1} + \boldsymbol{\epsilon}_t$$

where $\boldsymbol{\epsilon}_t \sim N(\mathbf{0}, \boldsymbol{\Sigma}_\epsilon)$ for all time periods t.

We will assume a GMRF so that $\mathbf{\Sigma}_\epsilon^{-1} = \mathbf{Q}$ is sparse and has parameters $\boldsymbol{\theta}_\epsilon$.

```
##
# Parameters for the spde:
sigma0 <- .2 # standard deviation
range0 <- .1 # range
# convert into tau and kappa
kappa0 = sqrt(8)/range0
tau0 = 1/(sqrt(4*pi)*kappa0*sigma0)

spde1 <- inla.spde2.matern(mesh=mesh2,
                           B.tau=cbind(log(tau0),1,0),
                           B.kappa=cbind(log(kappa0), 0, 1),
                           theta.prior.mean=c(0,0),
                           theta.prior.prec=c(1,1))
```

Evaluate a good prior for theta. I want to make sure that theta translates into good values for sigma and range:

```
data.frame(theta1=rnorm(1000,0, sd=1/1), theta2=rnorm(1000, 0, 1/1)) %>%
  mutate(logtau=(log(tau0)+theta1), logkappa=(log(kappa0)+theta2)) %>%
  mutate(range= sqrt(8)/exp(logkappa), sigma=1/(sqrt(4*pi)*exp(logtau+logkappa))) %>%
  select(range, sigma) %>% summary
```

```
##      range              sigma
##  Min.   :0.005449   Min.   : 0.003868
##  1st Qu.:0.053829   1st Qu.: 0.083663
##  Median :0.103582   Median : 0.205436
##  Mean   :0.174819   Mean   : 0.527369
##  3rd Qu.:0.204790   3rd Qu.: 0.530377
##  Max.   :2.667202   Max.   :17.905475
```

That seems good.

```
# Set precision:
Q <- inla.spde2.precision(spde1, theta=c(0,0))
```

```
## Note: method with signature 'diagonalMatrix#sparseMatrix' chosen for function '%*%',
##  target signature 'ddiMatrix#dgTMatrix'.
##  "Matrix#TsparseMatrix" would also be valid
## Note: method with signature 'TsparseMatrix#Matrix' chosen for function '%*%',
##  target signature 'dgTMatrix#ddiMatrix'.
##  "sparseMatrix#diagonalMatrix" would also be valid
```

```
LQ <- Cholesky(Q)
```

Data Level

The Instrument and Proxy data follow the model

$$\mathbf{Y}_t = \mathbf{X}_t\boldsymbol{\beta}_\mathbf{0} + \mathbf{Z}_t\boldsymbol{\alpha}_t + \boldsymbol{\nu}_t$$

where $\boldsymbol{\nu}_t \sim N(\mathbf{0}, \boldsymbol{\Sigma}_{\nu,t})$. We assume that $\boldsymbol{\Sigma}_{\nu,t}$ is diagonal with parameters $\boldsymbol{\theta}_\eta$. $\boldsymbol{\theta}_\eta$ contains σ^2_{ps} for each proxy species and σ^2_i for an instrument record.

The instruments have the simple model: $y_t = A\alpha_t + \nu_t$, where A is the spatial basis matrix that translates spatial climate grid points to instrument locations. We can thus write that $\mathbf{X}_t = \mathbf{0}$ and $\mathbf{Z}_t = \mathbf{A}_t$

```
# Z = A.precip
Z <- A.precip
```

The tree proxies have the model: $y_t = \beta_{0s} + \beta_{1s}A\alpha_t + \nu_t$, where again A is a matrix that translates spatial climate grid points to a tree proxy location (hence, $A\alpha$ is the climate at the tree location).

Prior Level:

prior on $\boldsymbol{\alpha}_1$

$\boldsymbol{\alpha}_1 \sim N(\boldsymbol{\mu}_1, \boldsymbol{\Sigma}_1)$. We set $\mu_1 = 0$ and $\boldsymbol{\Sigma}_1) = \sigma_1^2\mathbf{I}$, where $\sigma_1^2 = 2^2$ or some other value suitable to weakly constrain the range of the beginning temperature field.

```
mu_1 <- 0
sigma_sq_1 <- 4
```

prior on ρ

$\rho \sim U(0, 1)$.

prior on μ

$\mu \sim N(\mu_0, \sigma^2_\mu)$. we set the prior mean of the climate field to have mean $\mu_0 = 0$ and the standard deviation to $\sigma^2_\mu = 5$. This is a large prior and should not dominate the posterior.

```
mu_0 = 0
sigma_sq_mu = 5
```

prior on σ^2_I

$\sigma^2_I \sim IG(\lambda_I, \nu_I)$, i.e. $P(\sigma^2_I) \propto (\sigma^2_I)^{-(\lambda_I+1)} \exp(\nu_I/\sigma^2_I)$. Following Tingley and Gelman, this prior corresponds to $2\lambda_I$ observations with average squared deviation (ν_I/λ_I). We set each to .5.

```
nu_I <- .5
lambda_I <- .5
```

prior on σ^2_{Ps}

Same as for σ^2_I.

Posterior:

We can write the joint distribution as:

$$\frac{n-1}{2}\log|Q(\boldsymbol{\Theta})| - \frac{1}{2}\sum_{t=1}^{n-1}(\boldsymbol{\alpha}_{t+1} - \mu\mathbf{1} - \rho(\boldsymbol{\alpha}_t - \mu\mathbf{1}))^T\mathbf{Q}(\boldsymbol{\Theta})(\boldsymbol{\alpha}_{t+1} - \mu\mathbf{1} - \rho(\boldsymbol{\alpha}_t - \mu\mathbf{1})) +$$

$$-\frac{1}{2}\sum_{t=1}^{n}\log|\boldsymbol{\Sigma}_{\nu,t}| - \frac{1}{2}\sum_{t=1}^{n}(\mathbf{Y}_t - \mathbf{X}_t\boldsymbol{\beta}_{\mathbf{0}} - \mathbf{Z}_t\boldsymbol{\alpha}_t)^T\Sigma_{\nu,t}^{-1}(\boldsymbol{\Theta})(\mathbf{Y}_t - \mathbf{X}_t\boldsymbol{\beta}_{\mathbf{0}} - \mathbf{Z}_t\boldsymbol{\alpha}_t)$$

$$-\frac{1}{2}\log|\boldsymbol{\Sigma}_0| - \frac{1}{2}(\boldsymbol{\alpha}_1 - \boldsymbol{\mu}_0)^T\boldsymbol{\Sigma}_0^{-1}(\boldsymbol{\alpha}_1 - \boldsymbol{\mu}_0)$$

$$-\frac{1}{2}\log\sigma_\mu^2 - \frac{1}{2}\frac{(\mu-\mu_0)^2}{\sigma_\mu^2}$$

Posterior of μ

$$P(\mu \mid \cdot) \propto -\frac{1}{2}\sum_{t=1}^{n-1}((\boldsymbol{\alpha}_{t+1} - \rho\boldsymbol{\alpha}_t) - \mathbf{1}(1-\rho)\mu)^T Q((\boldsymbol{\alpha}_{t+1} - \rho\boldsymbol{\alpha}_t) - \mathbf{1}(1-\rho)\mu) - \frac{1}{2}\frac{(\mu-\mu_0)^2}{\sigma_\mu^2}$$

$\mu \mid \cdot \sim N(V_\mu c_\mu, V_\mu)$, where

$$V_\mu^{-1} = \frac{1}{\sigma_\mu^2} + (n-1)(1-\rho)^2\mathbf{1}^T Q\mathbf{1}$$

$$c_\mu = \frac{\mu_0}{\sigma_\mu^2} + \sum_{t=1}^{n-1}((1-\rho)\mathbf{1})^T Q(\boldsymbol{\alpha}_{t+1} - \rho\boldsymbol{\alpha}_t)$$

Posterior of ρ

$$P(\rho \mid \cdot) \propto -\frac{1}{2}\sum_{t=1}^{n-1}(\boldsymbol{\alpha}_{t+1} - \mu\mathbf{1} - (\boldsymbol{\alpha}_t - \mu\mathbf{1})\rho)Q(\boldsymbol{\alpha}_{t+1} - \mu\mathbf{1} - (\boldsymbol{\alpha}_t - \mu\mathbf{1})\rho)$$

over the range (0,1).

$\rho \mid \cdot \sim TN_{0,1}(V_\rho c_\rho, V_\rho)$ where

$$V_\rho^{-1} = \sum_{t=1}^{n-1}(\boldsymbol{\alpha}_t - \mu\mathbf{1}))^T Q(\boldsymbol{\alpha}_t - \mu\mathbf{1}))$$

$$c_\rho = \sum_{t=1}^{n-1}(\boldsymbol{\alpha}_t - \mu\mathbf{1}))^T Q(\boldsymbol{\alpha}_{t+1} - \mu\mathbf{1}))$$

Posterior of σ_I^2

$$\propto (\frac{1}{\sigma_I^2})^{\lambda_I+1}\exp(-\nu_i/\sigma_I^2)\prod\prod(\frac{1}{\sigma_I^2 n_{it}})^{1/2}\exp(\frac{(Y_{it}-\hat{Y}_{it})^2}{2\sigma_I^2 n_{it}})$$

$$\propto (\frac{1}{\sigma_I^2})^{\lambda_I+1}\exp(-\nu_i/\sigma_I^2)\left(\frac{1}{\sigma_I^2}\right)^{1/2\sum N_{It}}\exp\left(\sum\sum\frac{(Y_{it}-\hat{Y}_{it})^2}{2\sigma_i^2 n_{it}}\right)$$

$$\propto \left(\sigma_I^2\right)^{-1/2\sum N_{It}-\lambda_I-1}\exp\left(-\frac{1}{\sigma_I^2}\left(-\nu_I - \sum\sum\frac{(Y_{it}-\hat{Y}_{it})^2}{2n_{it}}\right)\right)$$

which is $IG(\lambda_I + 1/2\sum_t N_{It}, \nu_I + \sum_t\sum_{i\in N_{It}}\frac{(Y_{it}-\hat{Y}_{it})^2}{2n_{it}})$

Posterior of τ, κ

τ and κ don't have a closed form posterior distribution.

$$\propto \log \pi(\boldsymbol{\Theta}) - \frac{n-1}{2} \log |Q(\boldsymbol{\Theta})| - \frac{1}{2} \sum_{t=1}^{n-1} (\boldsymbol{\alpha}_{t+1} - \mu \mathbf{1} - \rho(\boldsymbol{\alpha}_t - \mu \mathbf{1}))^T \mathbf{Q}(\boldsymbol{\Theta}) (\boldsymbol{\alpha}_{t+1} - \mu \mathbf{1} - \rho(\boldsymbol{\alpha}_t - \mu \mathbf{1}))$$

```
# y has a 106 x 1 vector
# alpha has a 580 x 1 vector
# beta is the mean mu of alpha.
# there is no X.
# W = (1-rho) * mu
# T = rho*I
years <- anom.df %>% ungroup %>%
  distinct(year) %>% arrange(year) %>%
  mutate(year=as.numeric(year)) %>% .[["year"]]
N <- length(years)
Size <- mesh2$n

# Markov Chain Setup
NMC <- 1
# Save the storage objects
mc <- list(theta = matrix(NA, NMC+1, 2),
           alpha = matrix(NA, NMC+1, Size),
           mu = matrix(NA, NMC+1, 1),
           rho = matrix(NA, NMC+1,1),
           sigma_I = matrix(NA, NMC+1, 1))
mc$theta[1,] <- c(0,0)
mc$alpha[1,] <- rep(0, Size)
mc$mu[1] <- 0
mc$rho[1] <- .9
mc$sigma_I[1] <- 1

# We don't have a precip observation each time period, so let's precompute
#  which observation exist for each time period
y <- vector(mode='list', length=N) # the precip data for each time period
data_ids <- vector(mode='list', length=N) # The IDs for the precip data each period
A11 <- vector(mode='list', length=N) # A11 is the precision matrix of the data.  It
# depends on the number of observations.  It is the A11 matrix in McCausland et al 2011
# !!! A11 NEEDS TO BE MULTIPLIED BY sigma_sq_I AT EACH ITERATION !!!!!!!!!!!!!!!!!!!!!!
for(i in 1:N){
  y[[i]] <- anom.df %>% filter(year==years[i]) %>%
    arrange(SID) %>% filter(is.finite(precip))
  data_ids[[i]] <- y[[i]][['SID']]
  A11[[i]] <- nobs.df %>% filter(year==years[i]) %>%
    filter(SID %in% y[[i]][['SID']]) %>% .[["nobs"]]
  A11[[i]] <- .sparseDiagonal(A11[[i]], n=length(A11[[i]]))
}
```

```
##########################################################################
# Markov Chain Loop
for(mciter in 1:NMC){
```

```
}

# A11 is diagonal
# The var of a tile is sigma^2/nobs
tau_y <- 1/sigma_y
# A22 is Q

################################################################################
A11 <- vector(mode='list', length=N)
y <- vector(mode='list', length=N)
data_ids <- vector(mode='list', length=N)
Omega_tt <- vector(mode='list', length=N)

# A22 is Q
# TA22 is rho*Q*rho
rho=.8
TA22 <- rho^2*Q

Omega_tt[[1]] <- t(Z[data_ids[[1]],]) %*% A11[[1]] %*% Z[data_ids[[1]],] + TA22 + Q
for(t in 2:N-1){
  Omega_tt[[t]] <- t(Z[data_ids[[t]],]) %*% A11[[t]] %*% Z[data_ids[[t]],] + TA22 + Q
}
Omega_tt[[N]] <- t(Z[data_ids[[N]],]) %*% A11[[N]] %*%  Z[data_ids[[N]],] + Q

Omega_t1 <- -rho *Q # Omega_{t,t+1}

mu=0
Wb <- drop0(matrix((1-rho)*mu, nrow(Q)))
Q1 <- Q
a1 <- matrix(0, nrow(Q))
c <- vector(mode='list', length=N)
c[[1]] <- t(Z[data_ids[[1]],]) %*% drop0(A11[[1]] %*% (y[[1]]$precip-mu)) - rho*Q %*% Wb + Q1 %*% a1
for(t in 1:N){
    c[[t]] <- t(Z[data_ids[[t]],]) %*% drop0(A11[[t]] %*% (y[[t]]$precip-mu)) - rho*Q %*% Wb + Q %*% Wb
}
c[[N]] <- t(Z[data_ids[[N]],]) %*% drop0(A11[[N]] %*%y[[N]]$precip) + Q %*% Wb


################################################################################
Chol_omega <- vector(mode='list', length=N)
Chol_omega[[1]] <- Cholesky(Omega_tt[[1]])
# Note, that when following McCausland, that t(Omega_t-1,t) = Omega_t1
for(t in 2:N){
  temp <- Omega_tt[[t]] - (Omega_t1) %*% solve(Chol_omega[[t-1]], t(Omega_t1))
  Chol_omega[[t]] <- Cholesky(drop0(zapsmall(temp)),super=NA)
}

# LO = Lambda_t \ Omega_t1
LO <- vector(mode='list', length=N)
for(t in 1:N){LO[[t]] <- drop0(zapsmall(solve(Chol_omega[[t]], Omega_t1, system='Lt')))}

# OSO = Omega_{t,t+1}^T \Sigma_t Omega_{t,t+1}
OSO <- vector(mode='list', length=N)
```

```r
for(t in 1:N){OSO[[t]] <- crossprod(LO[[t]])}

m <- vector(mode='list', length=N)
m[[1]] <- solve(Chol_omega[[1]], c[[1]], system='A')
for(t in 2:N){
  m[[t]] <- solve(Chol_omega[[t]], c[[t]] - Omega_t1 %*% m[[t-1]])
}

alpha <- vector(mode='list', length=N)
epsilon <- rnorm(n=nrow(Q))
alpha[[N]] <- m[[N]] + solve(Chol_omega[[N]], epsilon, system='Lt')
for(t in seq(N-1, 1, -1)){
  epsilon <- rnorm(n=nrow(Q))
  alpha[[t]] <- m[[t]] + solve(Chol_omega[[t]], epsilon - LO[[t]] %*% alpha[[t+1]])

}

plot(colMeans(do.call(cBind,alpha)))
```