Bayesian mixture model for environmental application

P. Bogani, P. Botta, S. Caresana, R. Carrara, G. Corbo, L. Mainini

Synthetic Data

First we include all the codes and create the synthetic data following this model:

$$\mathbf{y}_i = \mathbf{Z}\alpha_i + \theta_i + \epsilon_i$$
$$\epsilon_i' = (\epsilon_{i1}, \dots, \epsilon_{iT}) \sim \mathrm{N}_T\left(\mathbf{0}, \sigma_{\epsilon_i}^2 \mathbf{I}\right),$$
$$\theta_{it} = \rho_i \theta_{i,t-1} + \nu_{it} \quad \text{with } \nu_{it} \sim \mathrm{N}\left(0, \sigma_i^2\right),$$

We suppose that stations belonging to the same cluster are characterized by the same value of ρ_i and σ_i. For example, if stations 1 and 2 come from the same cluster $\rho_1 = \rho_2$ and $\sigma_1 = \sigma_2$.

To verify the correctness of the algorithm we set some random values for the unknown parameters and we generated θ_{it} and y_i accordingly.

Synthetic data generation

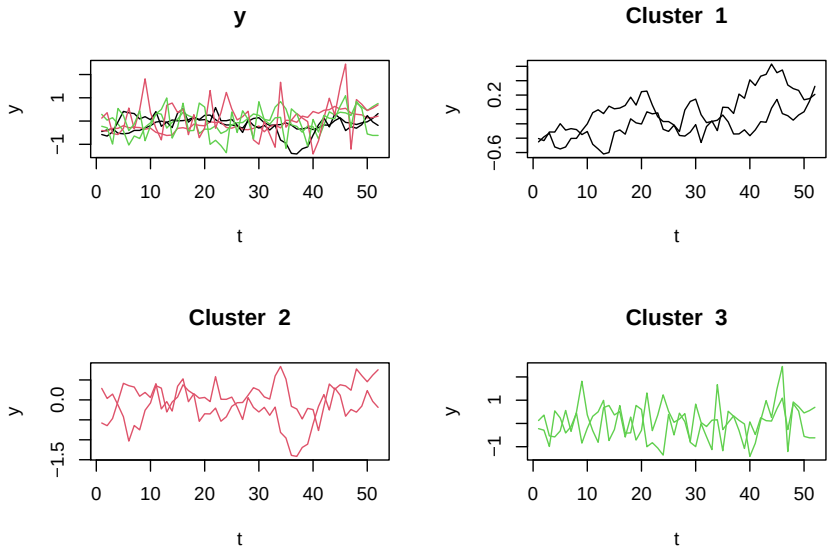
The data are generated in the function `generate synthetic data`, sampling from the normal distributions described:

```
for(j in 1:n_c[i]){
    k <- sum(n_c[1:i])-n_c[i] + j
    theta[k,] <- mvrnorm(n=1, mu=rep(0,T), Sigma=R)
    y[k,] <- mvrnorm(n=1, mu=Z%*% alpha[k,]+theta[k,], Sigma=sigma2_eps[k]*diag(T))
}
```

To check its correctness we created 6 observations equally divided in three clusters. $\alpha_i = 0$ for each i, and $\sigma_{\epsilon_i} = 0.0001$ for each i (i.e. low noise)

1. The first cluster has $\rho = 0.9$ and $\sigma = 0.1$
2. The second cluster has $\rho = 0.9$ and $\sigma = 0.5$
3. The third cluster has $\rho = 0.1$ and $\sigma = 0.5$

```
s = generate_synthetic_data(n_c=c(2,2,2), rho_c=c(0.9, 0.9, 0.1),
        sigma2_c=c(0.1,0.5,0.5), sigma2_eps=0.0001, alpha=c(0, 0, 0, 0), T=52)
par(mfrow=c(2,2))
matplot(s$data, type='l', col=unique(s$labels), xlab='t', ylab='y', lty=1, main='y')
for (j in 1:length(unique(s$labels))){
  matplot(s$data[,s$labels==j], type='l', col=j, xlab='t', ylab='y'
          , lty=1, main=paste('Cluster ',j))
}
```

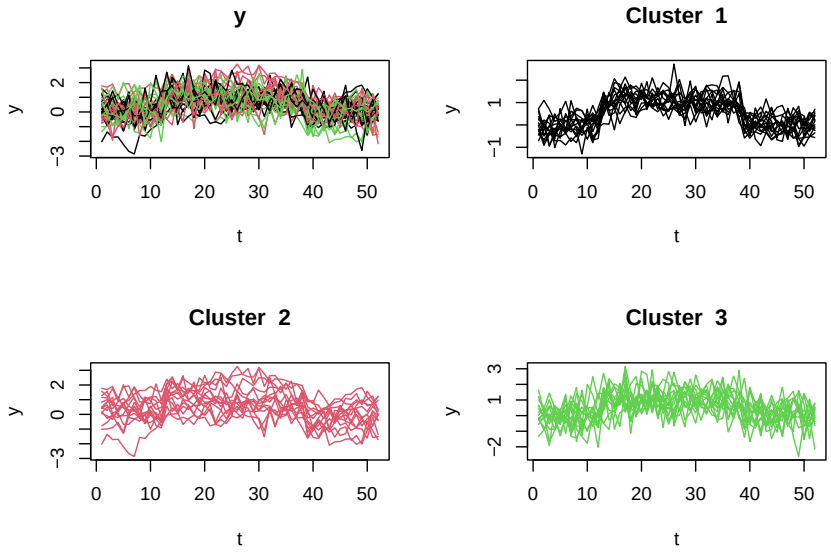


The generated data in this small example follow our expectations: stations from the first cluster have a strong persistence, while stations from the third cluster are practically white noises.

Posterior sampling

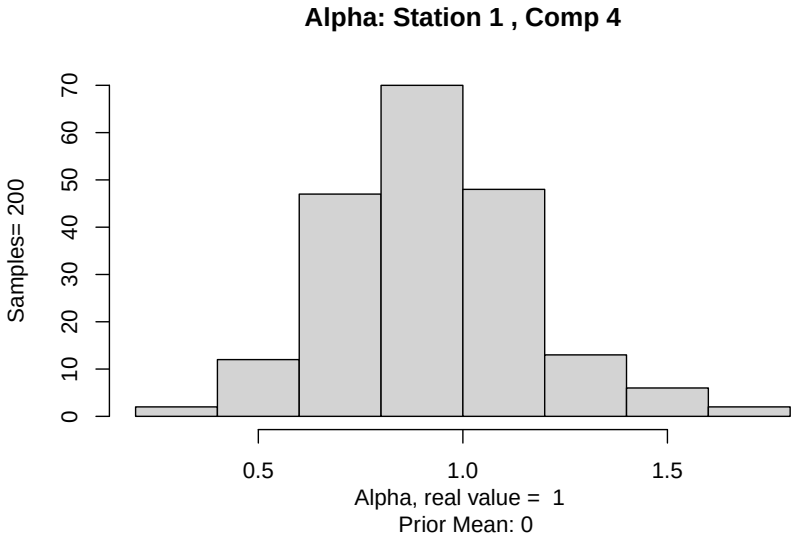
To verify the correctness of the posteriors we used a larger synthetic dataset with 45 stations and three clusters (with the same ρ_i and σ_i as before). This time, $\sigma_{\epsilon_i} = 0.1$ and $(\alpha_i)_4 = 1$ for each i. This induces a seasonality and a higher noise.

```
s = generate_synthetic_data(n_c=c(15,15,15), rho_c=c(0.9, 0.9, 0.1),
    sigma2_c=c(0.1,0.5,0.5), sigma2_eps=0.1, alpha=c(0, 0, 0, 1), T=52)
par(mfrow=c(ceiling((length(unique(s$labels))+1)/2),2))
matplot(s$data, type='l', col=unique(s$labels), xlab='t', ylab='y', lty=1, main='y')
for (j in 1:length(unique(s$labels))){
  matplot(s$data[,s$labels==j], type='l', col=j, xlab='t', ylab='y',
          lty=1, main=paste('Cluster ',j))
}
```

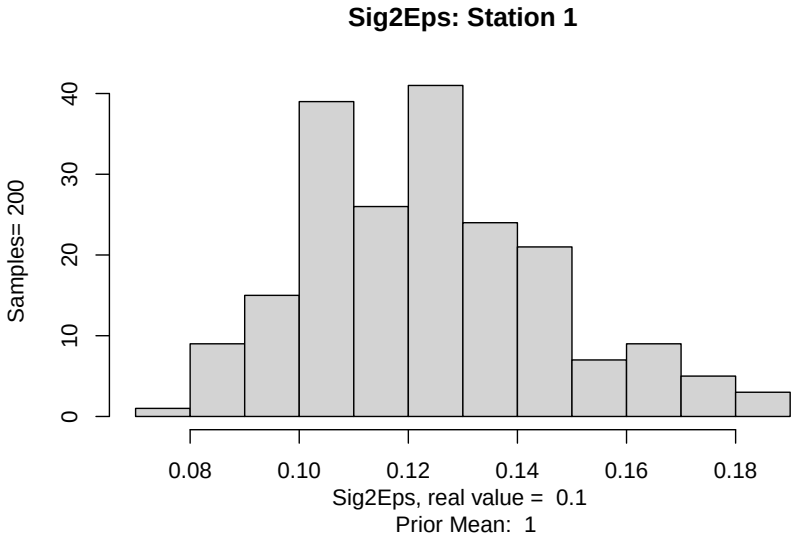


We executed the algorithm on this data and we plotted some posterior samples from $(\alpha_1)_4$ and σ_{ϵ_1}. Of course, we could also choose another station different from the first one.

```
debug_alpha(s, 1, 4, 200)
```




```
debug_sig2eps(s, 1, 200)
```



As we said, the real value of $(\alpha_1)_4$ which generated the data is 1. So, we expect a posterior mean approximately equal to 1. As you can see, the majority of our posterior samples vary between 0.5 and 1.5. The mass is not centered in 1 but it is slightly shifted to the left. That's correct considering that the posterior mean is also affected by the prior (mean), which is equal to 0.

Similarly, the real value of σ_{ϵ_1} is 0.1 but the posterior samples are slightly shifted to the right due to the prior (mean) equals to 1. Thus, our posteriors are perfectly able to estimate the real parameters.

Testing

Finally, we applied the algorithm on a synthetic dataset with 3 clusters with dimensions 9,15,5 and respectively ρ 0.8,0.6,0.1 and σ^2 0.018,0.028,0.015. Following the guidelines of … we selected a high value for the shape hyper-parameter of the Inverse-Gamma distribution of σ^2 ($a = 17$) and a low value for the rate hyper-parameter ($b = 0.3$). We have then selected the cluster structure minimizing the Binder loss function with respect to the known real cluster we created.

```
rho_c <- c(0.8,0.6,0.2)
sigma2_c <- c(0.015,0.028,0.015)
synthetic = generate_synthetic_data(n_c=c(9,15,5), rho_c=rho_c, sigma2_c=sigma2_c,
                                    sigma2_eps=0.0001, alpha=c(0, 0, 0, 0), T=52)

data = synthetic$data
labels = synthetic$labels
coord = synthetic$coord
```

We run the algorithm:

```
pb=progress_bar$new(total=300)
invisible(pb$tick(0))
tseriesc.out = tseriesclust(data, maxiter=300, thinning=1,
                            scale=TRUE, a=17,b=0.3,ab=3,bb=3,seed=4)
```

We then used a Binder loss function to find the best cluster structure

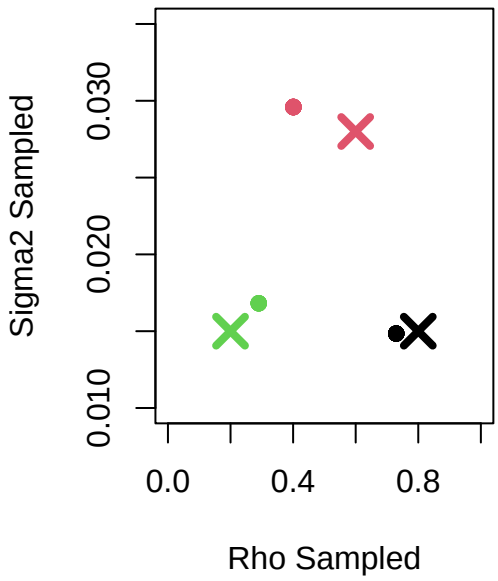
```
jstar <- tseriesc.out$jstarsample
memory <- tseriesc.out$memorygn

real_cluster <- c(rep(1,9),rep(2,15),rep(3,5))
sample_clust <- binderfunction(real_cluster,memory)
gnstar <- sample_clust[[1]]
index <- sample_clust[[2]]

jstar[index,]
rho <- tseriesc.out$rhosample[index,]
sigma2 <- tseriesc.out$sig2thesample[index,]
```

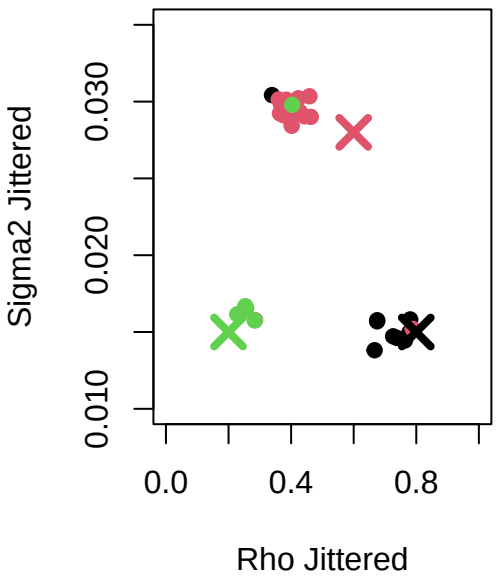
and plotted the real values with respect to the sampled ones:

```
plot(rho,sigma2,col=gnstar ,pch=19 ,xlim = c(0,1),ylim = c(0.01,0.035)
     ,xlab = 'Rho Sampled',ylab = 'Sigma2 Sampled')
points(rho_c,sigma2_c, col=c(1,2,3),pch=4 ,cex=2,lwd=4)
```



and the real values with respect to the jittered values of ρ and σ just to visual purposes:

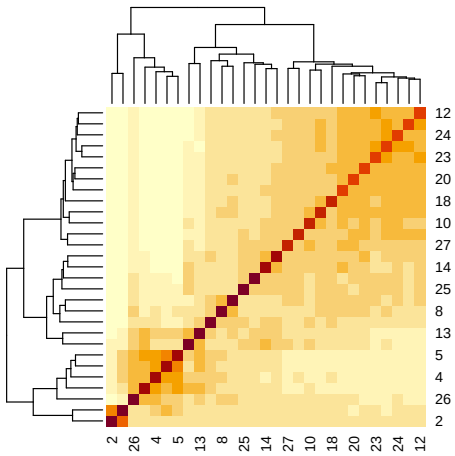
```
jf <- 3
set.seed(28) #just for the jittering
plot(jitter(rho,jf),jitter(sigma2,jf),col=real_cluster
     ,pch=19 ,xlim = c(0,1),ylim = c(0.01,0.035)
     ,xlab = 'Rho Jittered',ylab = 'Sigma2 Jittered')
points(rho_c,sigma2_c, col=c(1,2,3),pch=4 ,cex=2,lwd=4)
```



The points in the figures represent the sampled values of ρ and σ, and as expected we obtained 3 clusters with values acceptably similar to the real and original value of the ρ and σ of the 3 clusters (plotted as $\times$).

Finally, we plotted the similarity matrix of the cluster obtained:

```
simm <- comp.psm(memory)
heatmap(as.matrix(simm))
```



We can indeed denote from the dendrogram the formation of the three clusters.