PmmSuite - Gaussian_Ppmx

2023-12-15

```
source("include.R")
```

```
##
## Caricamento pacchetto: 'crayon'
```

```
## Il seguente oggetto è mascherato da 'package:ggplot2':
##
##     %+%
```

```
## hash-2.2.6.3 provided by Decision Patterns
## ❤ Loaded agrimonia dataset. Available as df_agri_full.
## Removed it as useless now.
##
## ❤ Loaded cleaned dataset. Available as df_agri.
## ❤ Loaded 2018 dataset. Available as df_2018.
## Converted column Time (of all dfs) to Date variable type, format: year-month-day.
## Created DATE_FORMAT = "%Y-%m-%d" variable for date comparisons.
## Usage example: df_agri$Time >= as.Date("2017-01-01",DATE_FORMAT).
## Actually also this works: df_agri$Time >= as.Date("2017-01-01").
##
## ❤ Loaded weekly divided dataset(s). Available as df_weekly.
## Uniformed names of stations (some were STA-ecc and some STA.ecc; now are all STA.ecc).
## Only of df_weekly and the following df_weekly_scaled_centered (ie df_wsc).
## This name change was also needed for the graph cluster plot function.
##
## ❤ Created scaled df_weekly dataset. Available as df_weekly_scaled_centered (or df_wsc).
## Scaled variables were c(3,4,6,8:10,12:20,22:37)
## Untouched variables were
##    (col 1) X,
##    (col 2) IDStations,
##    (col 5) Time,
##   (col 11) WE_mode_wind_direction_10m,
##   (col 21) WE_mode_wind_direction_100,
##   (col 38) day,
##   (col 39) week
## ------------------------------------------------------
##   (col 7) AQ_pm10 has been centered, not scaled
## (col 3&4) Latitude and Longitude have also been scaled
##           (fits were better in this way)
##
## ❤ Created stations split function Available as create_df_stat(df).
## Use it as my_df_stat = create_df_stat(df_2018).
## Then for example my_df_stat[["1264"]] retrieves the dataset for station 1264.
##
## ❤ Created function to get color palettes. Available as colora(len, seed, show).
## Try for example colora(10,56,1).
## ❤ Created utility to explain covariates. Available as spiega(string).
## Try for example spiega("wind").
```

Plot

```
source("plot functions/plotter.R")
```

```
## Linking to GEOS 3.11.2, GDAL 3.6.2, PROJ 9.2.0; sf_use_s2() is TRUE
```

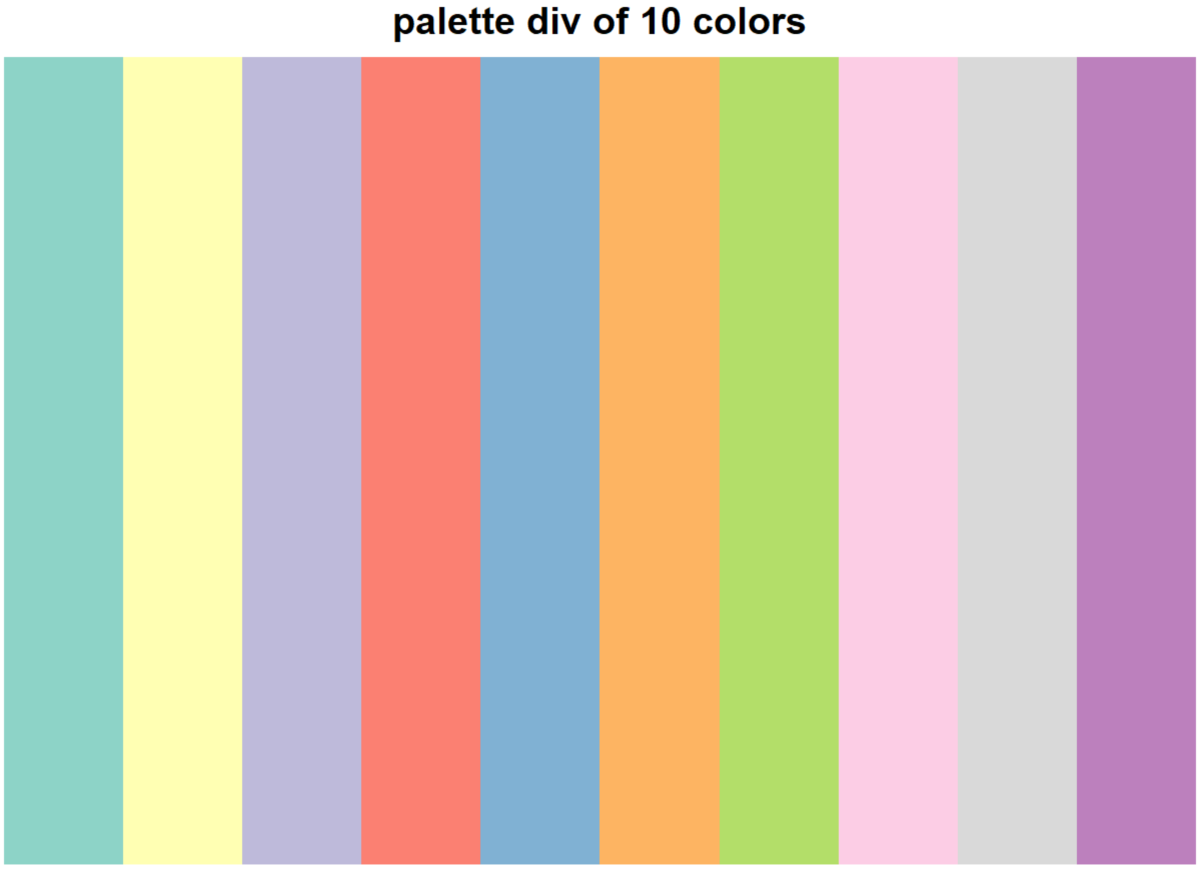
```
##
## Caricamento pacchetto: 'lubridate'
```

```
## I seguenti oggetti sono mascherati da 'package:base':
##
##     date, intersect, setdiff, union
## Loading AGC dataset, may took some time.
## Reading layer `gadm40_ITA_3' from data source
##   `C:\Users\alessandro\Documents\GitHub\progetto-bayesian\src\plot functions\italia\gadm40
_ITA_3.shp'
##   using driver `ESRI Shapefile'
## Simple feature collection with 8100 features and 14 fields
## Geometry type: MULTIPOLYGON
## Dimension:     XY
## Bounding box:  xmin: 6.630879 ymin: 35.49292 xmax: 18.52069 ymax: 47.09096
## Geodetic CRS:  WGS 84
## Reading layer `gadm40_ITA_2' from data source
##   `C:\Users\alessandro\Documents\GitHub\progetto-bayesian\src\plot functions\italia\gadm40
_ITA_2.shp'
##   using driver `ESRI Shapefile'
## Simple feature collection with 110 features and 12 fields
## Geometry type: MULTIPOLYGON
## Dimension:     XY
## Bounding box:  xmin: 6.630879 ymin: 35.49292 xmax: 18.52069 ymax: 47.09096
## Geodetic CRS:  WGS 84
## Reading layer `gadm40_ITA_1' from data source
##   `C:\Users\alessandro\Documents\GitHub\progetto-bayesian\src\plot functions\italia\gadm40
_ITA_1.shp'
##   using driver `ESRI Shapefile'
## Simple feature collection with 20 features and 11 fields
## Geometry type: MULTIPOLYGON
## Dimension:     XY
## Bounding box:  xmin: 6.630879 ymin: 35.49292 xmax: 18.52069 ymax: 47.09096
## Geodetic CRS:  WGS 84
```

cluster plots stuff

Stuff needed for the plot functions to work

```
cols = colora(10,"div")[-2] # divergent palette; togliamo il giallino
```



```
stations = unique(df_weekly$IDStations)
y=data.frame()

for(st in stations){
  y_we_pm10=cbind(as.data.frame(st),t(df_weekly[which(df_weekly$IDStations==st),"AQ_pm10"]))
  y=rbind(y,y_we_pm10)
}

rownames(y) = NULL
colnames(y)<- c("id",paste0("w", 1:53))
# this y needs to not be overwritten
```

```
library(salso)
```

```
## Warning: il pacchetto 'salso' è stato creato con R versione 4.3.1
```

```
library(ppmSuite)
```

```
## Warning: il pacchetto 'ppmSuite' è stato creato con R versione 4.3.2
```

```
##
## Caricamento pacchetto: 'ppmSuite'
## Il seguente oggetto è mascherato da 'package:maps':
##
##     ozone
```

Sometimes it fails the execution (dont know why). In that case: - go to the file include_clusters_function.R and run it all (ctrl+alt+R) - come back here and now it should run finely

gaussian_ppmx

All Covariate into the model

```
all_cov <- c(
  "Altitude", "WE_temp_2m", "WE_wind_speed_10m_mean",
  "WE_wind_speed_10m_max", "WE_mode_wind_direction_10m", "WE_tot_precipitation",
  "WE_precipitation_t", "WE_surface_pressure", "WE_solar_radiation",
  "WE_rh_min", "WE_rh_mean", "WE_rh_max",
  "WE_wind_speed_100m_mean", "WE_wind_speed_100m_max", "WE_mode_wind_direction_100m",
  "WE_blh_layer_max", "WE_blh_layer_min", "EM_nh3_livestock_mm",
  "EM_nh3_agr_soils", "EM_nh3_agr_waste_burn", "EM_nh3_sum",
  "EM_nox_traffic", "EM_nox_sum", "EM_so2_sum",
  "LI_pigs", "LI_bovine", "LI_pigs_v2",
  "LI_bovine_v2", "LA_hvi", "LA_lvi",
  "LA_land_use"
)
n <- length(all_cov)
```

Selection of the covariate

In order to choose the covariate to include in the model, i found the ‘starting’ variable comparing the length(all_cov) model. I selected the covariate relating to the model with the highest LPML.

For these analysis, I used the 5th week.

```
id <- which(LPML  == max(LPML))
all_cov[id] # Altitude
```

```
## [1] "Altitude"
```

Valutiamo modello migliore

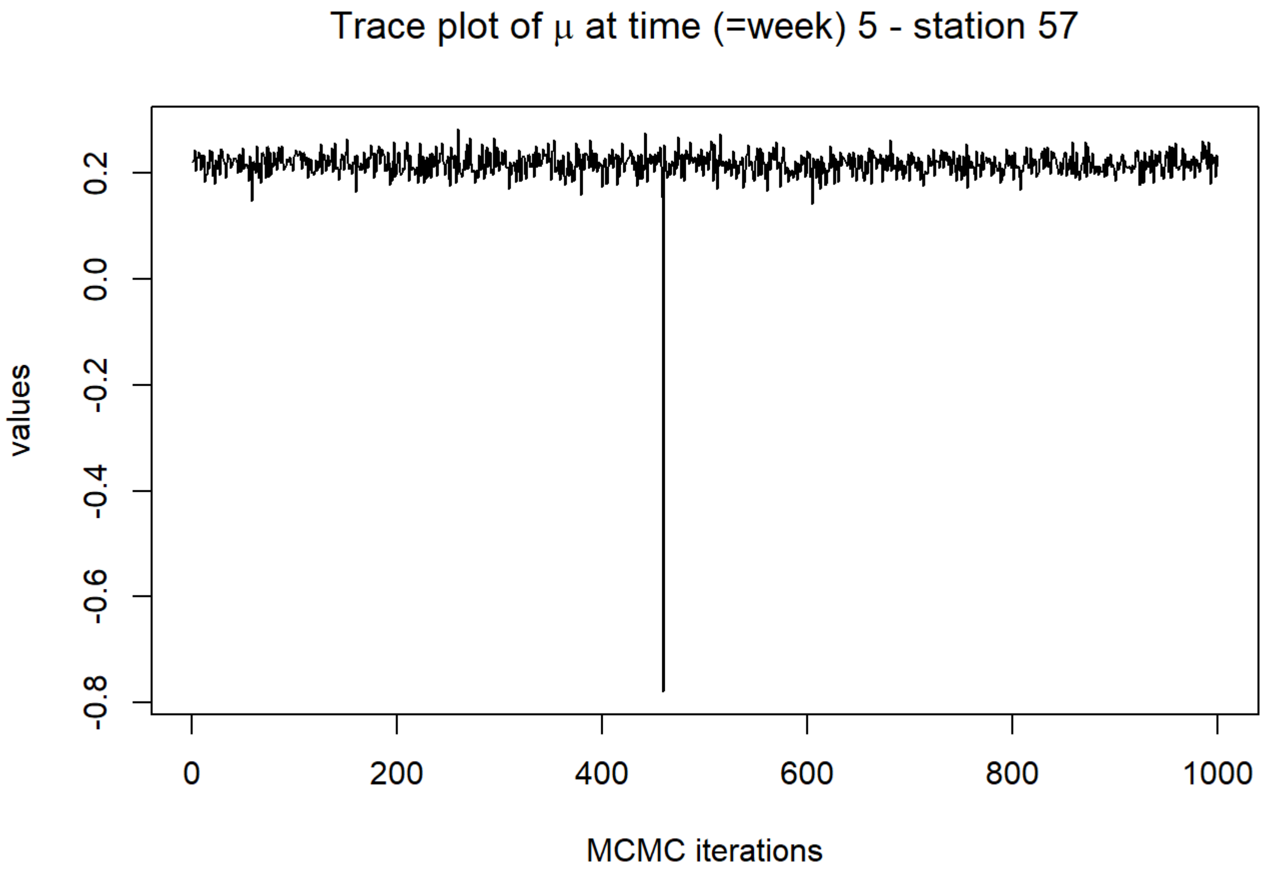
```
sampled_station = floor(runif(1,0,105))
cat("sampled_station =",sampled_station,"- that is station called",unique(df_weekly$IDStation
s)[sampled_station])
```

```
## sampled_station = 57 - that is station called 693
```

mu (Posterior mean of mu relativa a soggetto si) ->

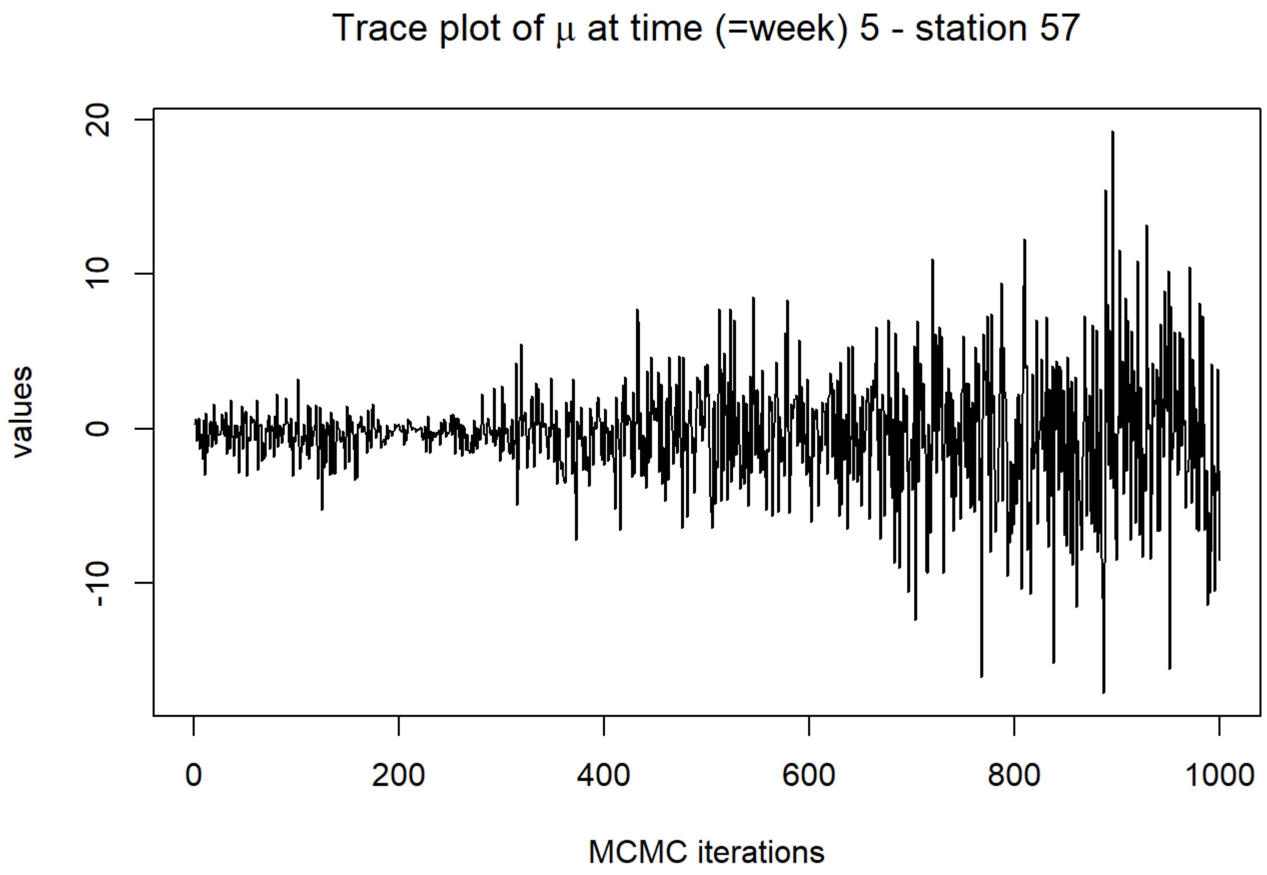
associata a muj

```
plot(fit$mu[,sampled_station],type="l",
         main=bquote("Trace plot of " * mu * " at time (=week) " * .(5) *
                         " - station " * .(sampled_station)),
         xlab = "MCMC iterations",ylab="values")
```



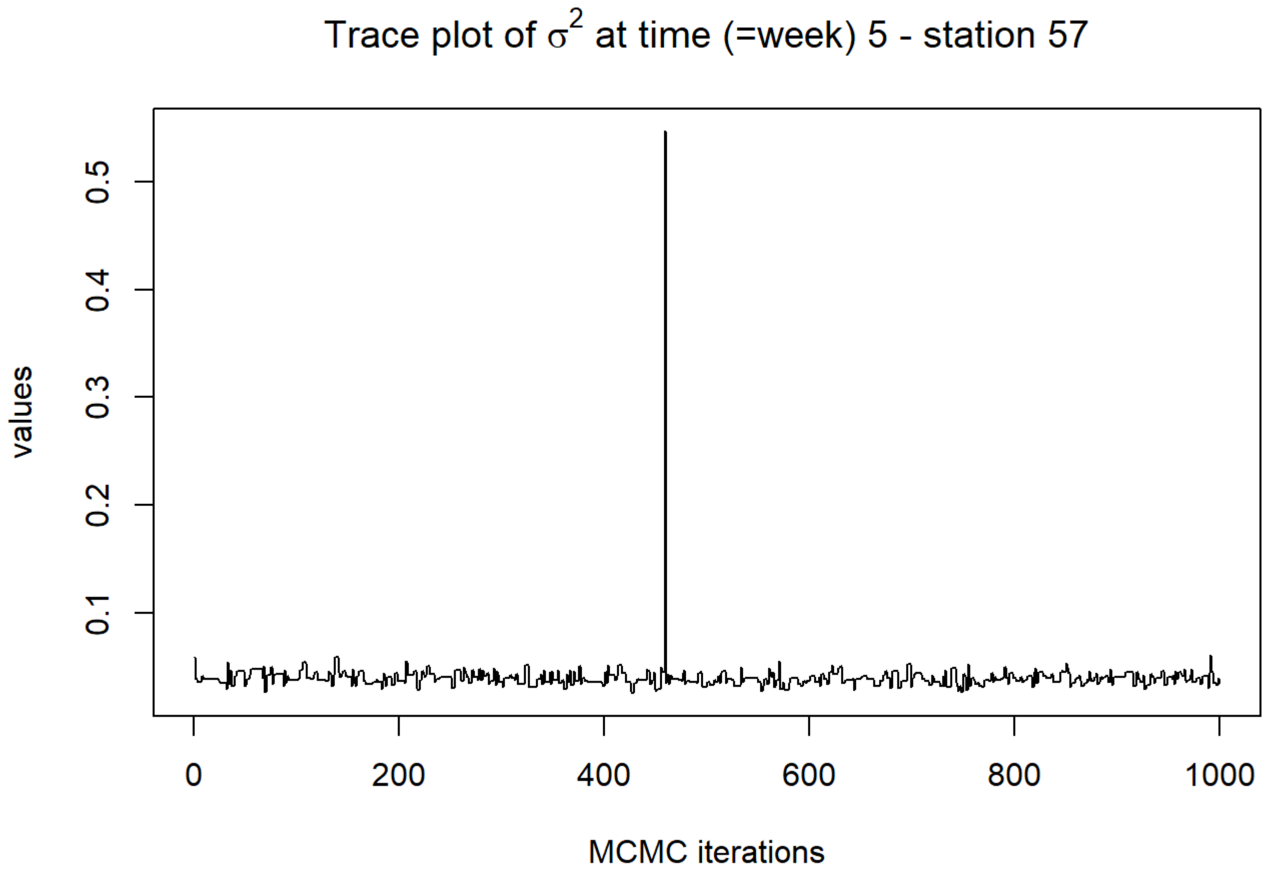
mu0 (Posterior values of mu0) –> associata a mu0

```
plot(fit$mu0,type="l",
         main=bquote("Trace plot of " * mu * " at time (=week) " * .(5) *
                         " - station " * .(sampled_station)),
         xlab = "MCMC iterations",ylab="values")
```



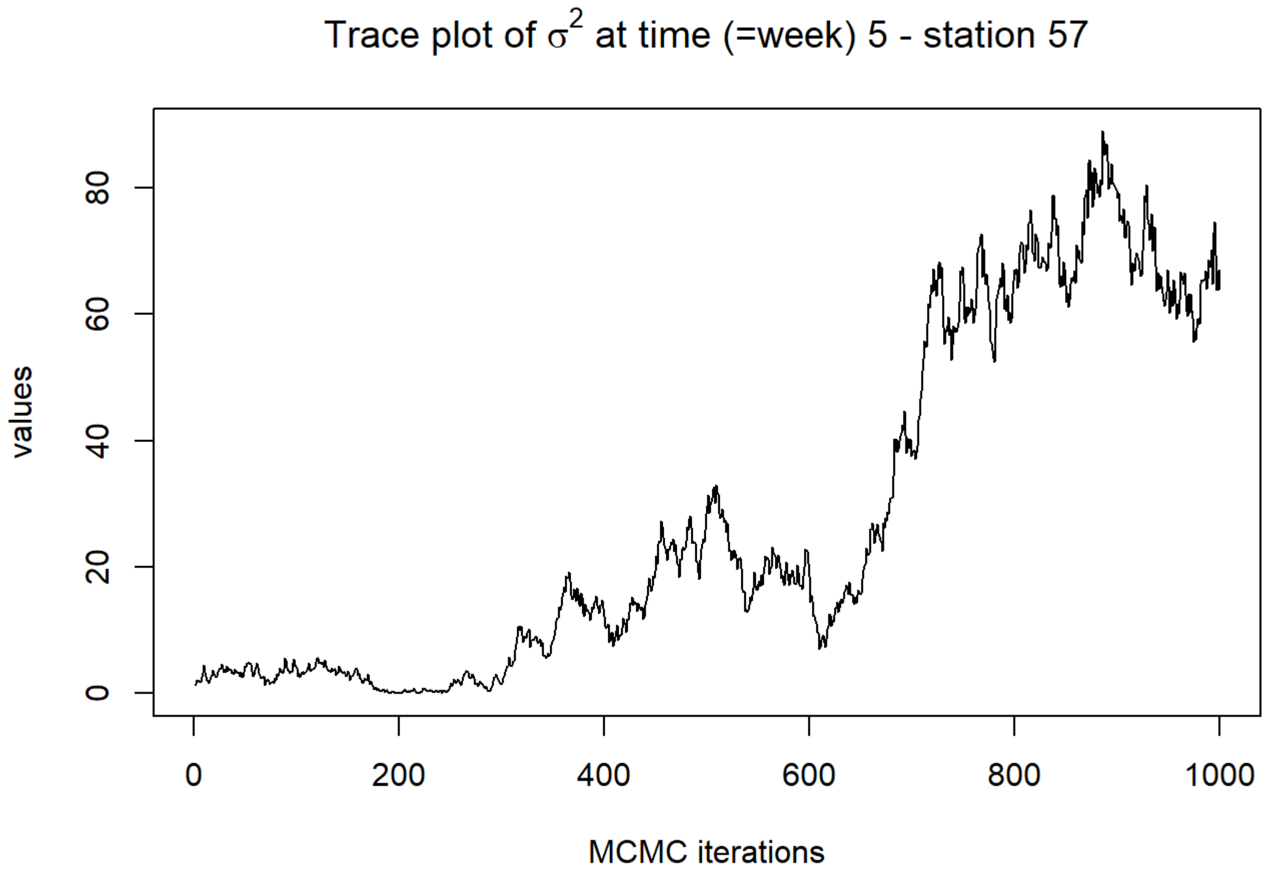
sigma2 (posterior of sigma^2 relativa a soggetto si) –> associata a sigmaj

```
plot(fit$sig2[,sampled_station],type="l",
        main=bquote("Trace plot of "* sigma^2 * " at time (=week) " * .(5) *
                     " - station " * .(sampled_station)),
        xlab = "MCMC iterations",ylab="values")
```



sig20 (Posterior value of sigma0^2) –> associata sigma0

```
plot(fit$sig20,type="l",
         main=bquote("Trace plot of " * sigma^2 * " at time (=week) " * .(5) *
                         " - station " * .(sampled_station)),
         xlab = "MCMC iterations",ylab="values")
```



Selezioniamo altre covariate includendo una ad una le covariate nel modello e le seleziono solo se LMPL aumenta. Parto da covariata selezionata precedentemente.

Per selezionare covariate uso settimana 5

```
set.seed(1)
all_cov_altitude <- c( "WE_temp_2m", "WE_wind_speed_10m_mean",
  "WE_wind_speed_10m_max", "WE_mode_wind_direction_10m", "WE_tot_precipitation",
  "WE_precipitation_t", "WE_surface_pressure", "WE_solar_radiation",
  "WE_rh_min", "WE_rh_mean", "WE_rh_max",
  "WE_wind_speed_100m_mean", "WE_wind_speed_100m_max", "WE_mode_wind_direction_100m",
  "WE_blh_layer_max", "WE_blh_layer_min", "EM_nh3_livestock_mm",
  "EM_nh3_agr_soils", "EM_nh3_agr_waste_burn", "EM_nh3_sum",
  "EM_nox_traffic", "EM_nox_sum", "EM_so2_sum",
  "LI_pigs", "LI_bovine", "LI_pigs_v2",
  "LI_bovine_v2", "LA_hvi", "LA_lvi",
  "LA_land_use"
)

all_cov_NOAlt_sample <- sample(all_cov_altitude)
n <- lenght(all_cov_NOAlt_sample)
LPML.cov <- numeric(0)
```

loop fit

Le variabili selezionate sono

```
var_selezionate
```

```
##  [1] "Altitude"                    "WE_temp_2m"
##  [3] "WE_wind_speed_10m_mean"      "WE_mode_wind_direction_100m"
##  [5] "EM_nh3_agr_soils"            "EM_nh3_agr_waste_burn"
##  [7] "LI_bovine_v2"                "LA_land_use"
##  [9] "EM_nox_sum"                  "WE_wind_speed_100m_max"
## [11] "LI_pigs_v2"                  "EM_nh3_sum"
## [13] "WE_wind_speed_10m_max"
```

```
variabili_selezionate <- c("Altitude", "WE_temp_2m", "WE_wind_speed_10m_mean", "WE_mode_wind_
direction_100m",
               "EM_nh3_agr_soils", "EM_nh3_agr_waste_burn", "LI_bovine_v2", "LA_land_use",
               "EM_nox_sum", "WE_wind_speed_100m_max", "LI_pigs_v2", "EM_nh3_sum",
               "WE_wind_speed_10m_max")
```

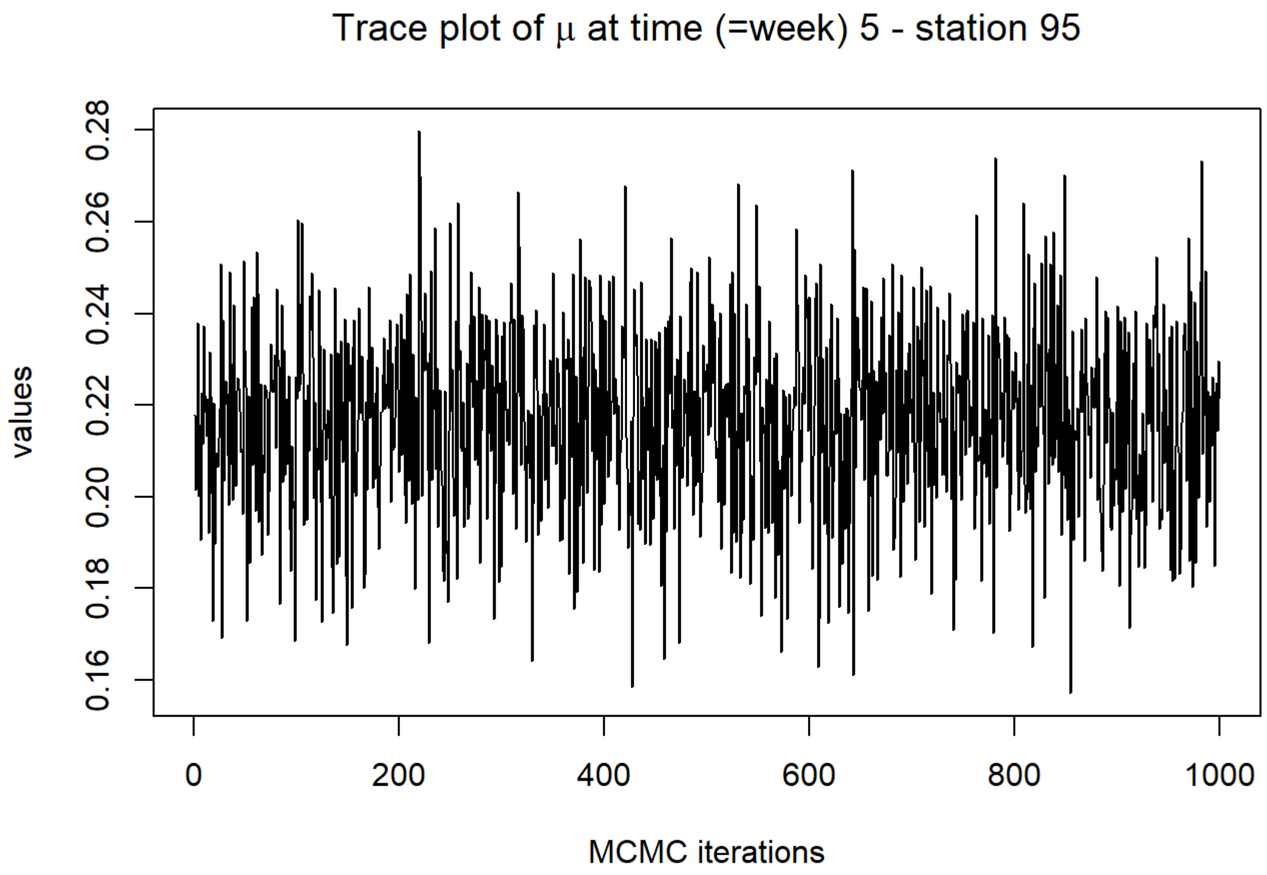
Vediamo convergenza

```
sampled_station = floor(runif(1,0,105))
cat("sampled_station =",sampled_station,"- that is station called",unique(df_weekly$IDStation
s)[sampled_station])
```

```
## sampled_station = 95 - that is station called STA.IT1917A
```

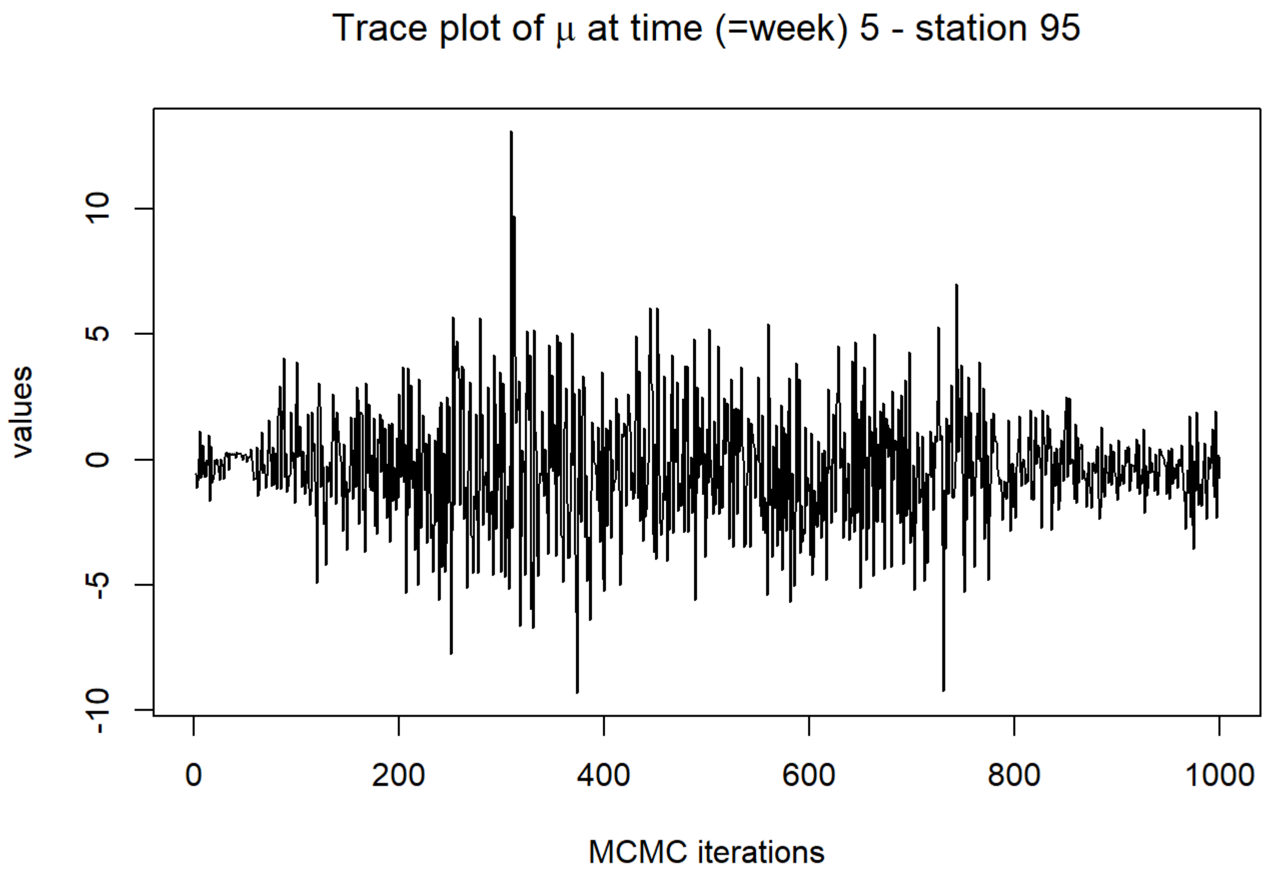
mu (Posterior mean of mu relativa a soggetto si) -> associata a muj

```
plot(fit$mu[,sampled_station],type="l",
         main=bquote("Trace plot of " * mu * " at time (=week) " * .(5) *
                         " - station " * .(sampled_station)),
         xlab = "MCMC iterations",ylab="values")
```



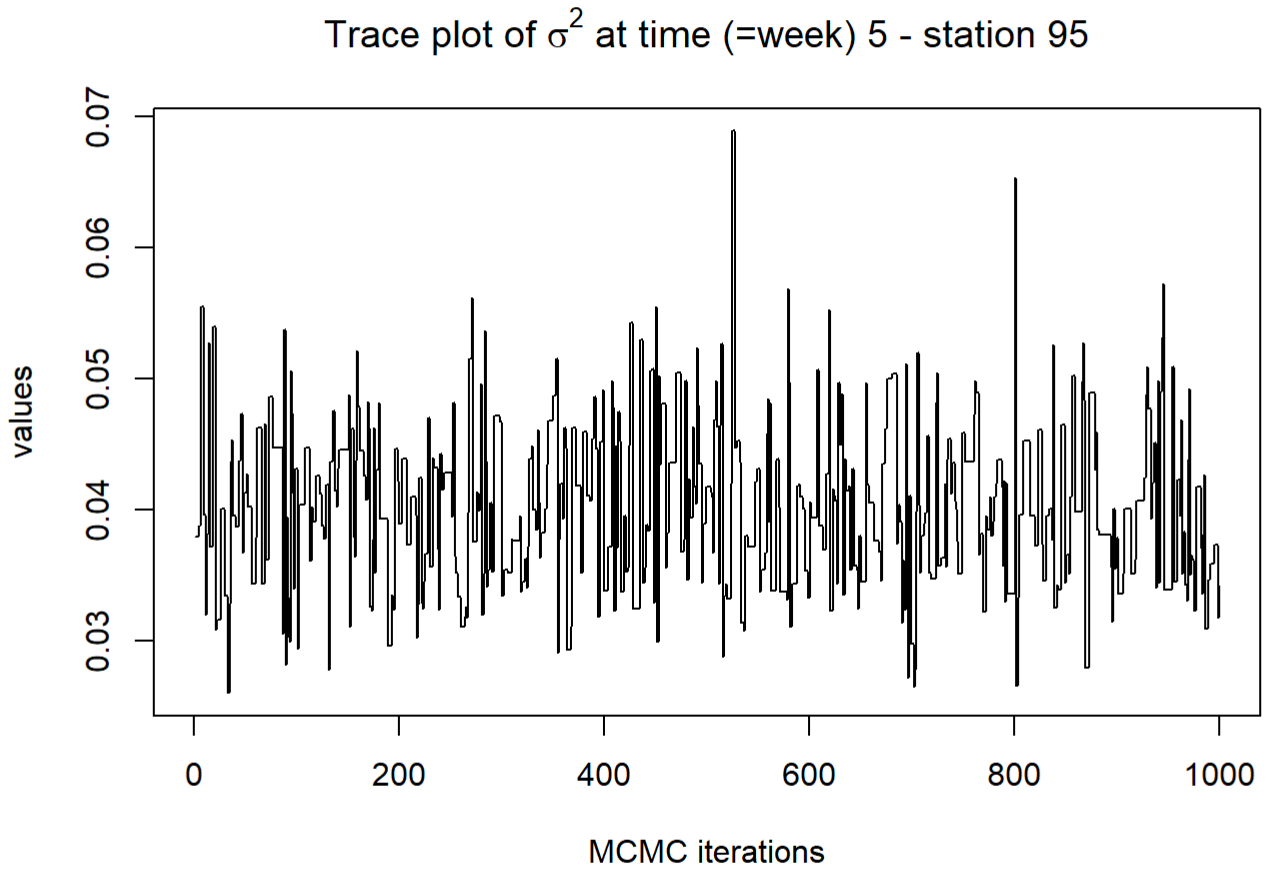
mu0 (Posterior values of mu0) –> associata a mu0

```
plot(fit$mu0,type="l",
         main=bquote("Trace plot of " * mu * " at time (=week) " * .(5) *
                        " - station " * .(sampled_station)),
         xlab = "MCMC iterations",ylab="values")
```



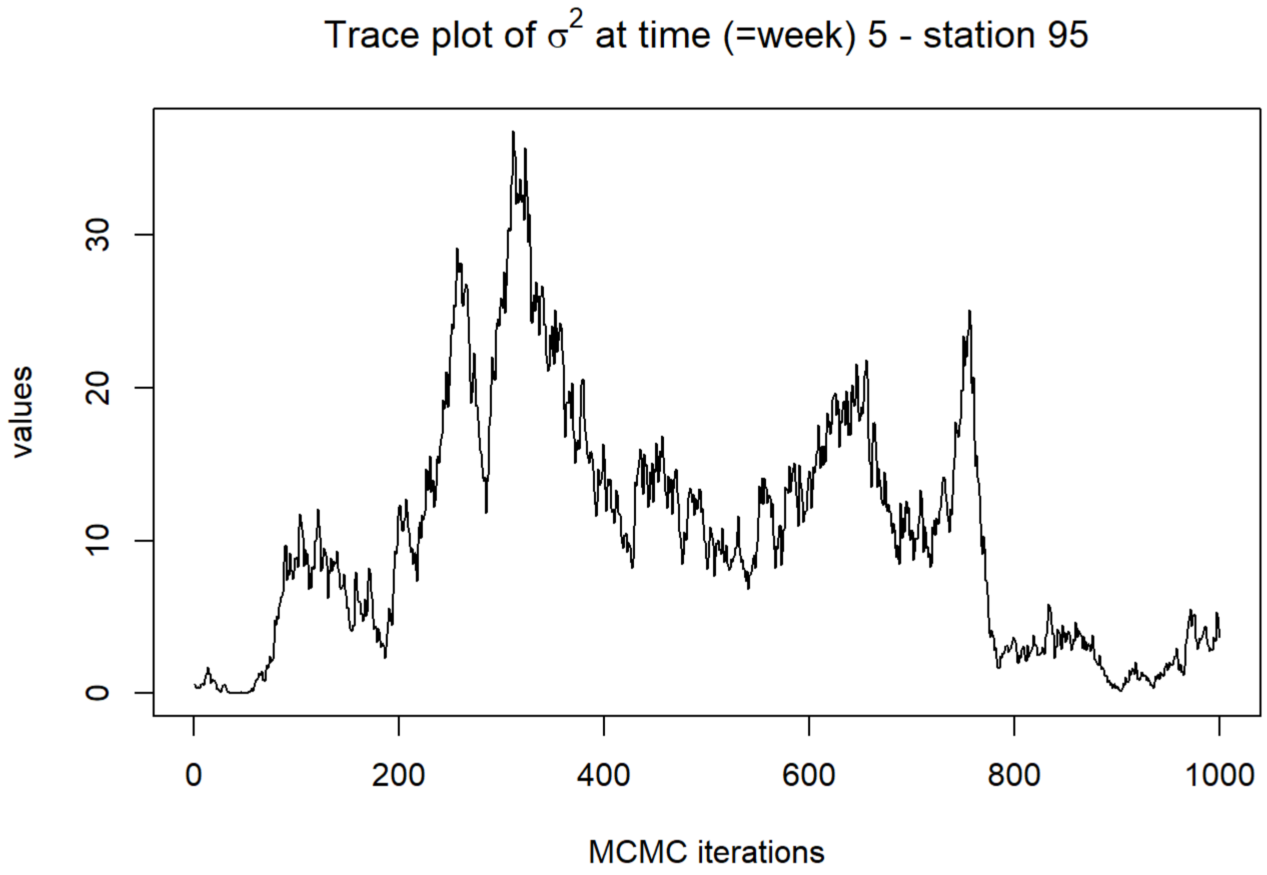
sigma2 (posterior of sigma^2 relativa a soggetto si) –> associata a sigmaj

```
plot(fit$sig2[,sampled_station],type="l",
         main=bquote("Trace plot of "* sigma^2 * " at time (=week) " * .(5) *
                        " - station " * .(sampled_station)),
         xlab = "MCMC iterations",ylab="values")
```



sig20 (Posterior value of sigma0^2) –> associata sigma0

```
plot(fit$sig20,type="l",
         main=bquote("Trace plot of " * sigma^2 * " at time (=week) " * .(5) *
                         " - station " * .(sampled_station)),
         xlab = "MCMC iterations",ylab="values")
```



Clusters

Trovo clusters finali assegnando ad ogni stazione il clusters che la contraddistingue di più

Per vedere i trace plot guarda quelli della sezione precedente dove considero una settimana sola. Il modello è lo stesso.

```
colnames(mat_cluster)[1] = 1
clus <- numeric(0)
for (i in 1:dim(mat_cluster)[1])
{ clus[i] <- as.numeric(names(table(mat_cluster[i,]))[which.max(table(mat_cluster[i,]))])
}
clus
```

```
##   [1] 1 2 2 1 2 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 2 1 1 2 2 2 2 2 2 2 2 1
##  [38] 1 1 1 1 1 2 2 2 2 2 1 1 2 1 1 2 1 1 1 1 2 1 2 1 1 1 1 1 1 1 1 1 1 1 1 1 1
##  [75] 2 2 1 2 1 1 1 1 1 1 1 1 1 1 1 1 1 1 2 2 1 1 2 2 1 1 1 1 1 2 2
```

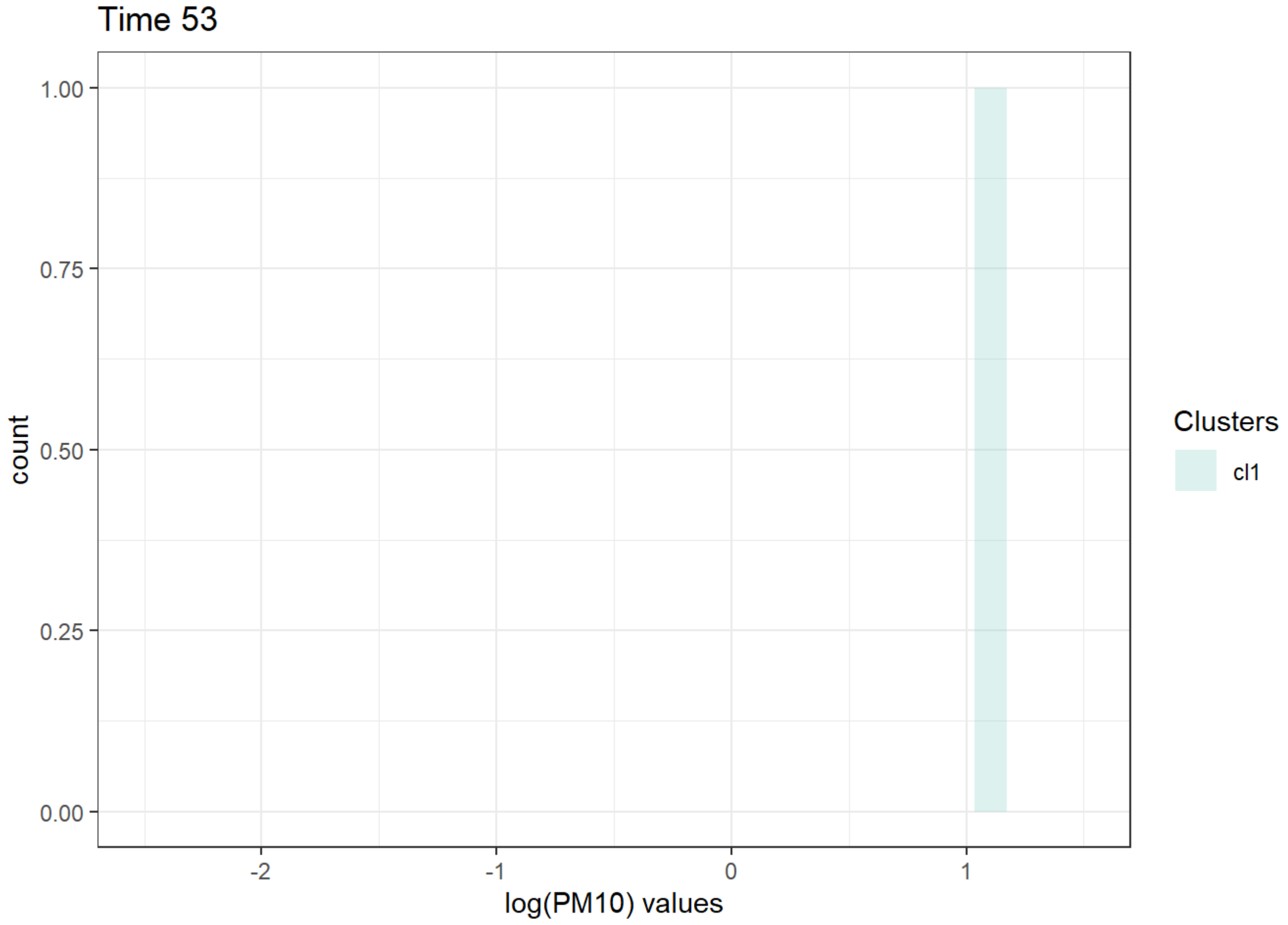
Plots of the clusters –> perchè mi da’ week 53, dove lo tolgo.

```
df_temp = data.frame(
    Longitude = unique(df_weekly$Longitude),
    Latitude = unique(df_weekly$Latitude),
    clusters = clus
)

df_temp$Time = rep(1,dim(df_temp)[1])
df_cluster_cut = df_temp


### Hist plot
# p = get_hist_color_plot(df_cluster_cut)
p = get_hist_fill_plot(df_cluster_cut) # choose one of these two
print(p)
```

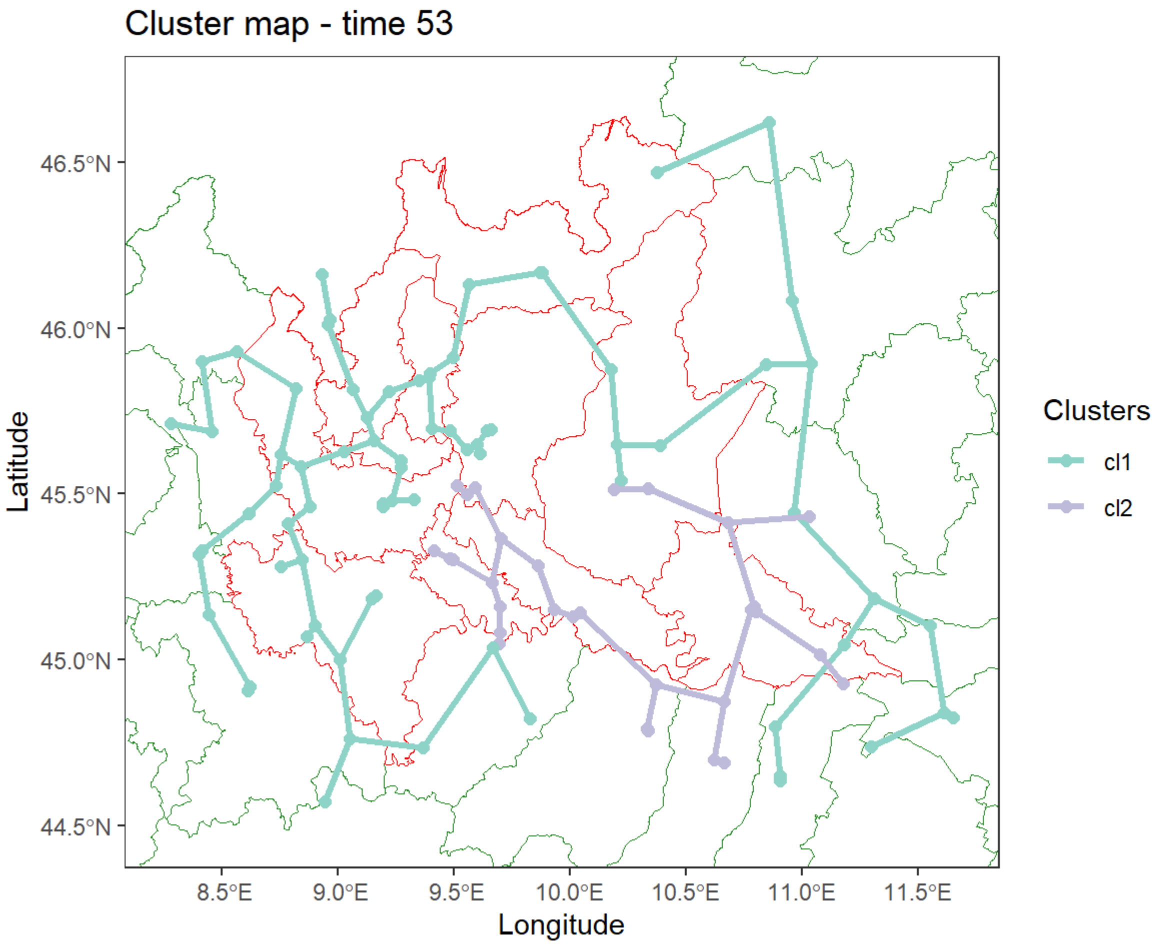
```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```

```
## Warning: Removed 104 rows containing non-finite values (`stat_bin()`).
```

```
## Warning: Removed 2 rows containing missing values (`geom_bar()`).
```



```
### Graph plot
q = get_graph_plot(df_cluster_cut)
print(q)
```



```
# or both together with
# plot_graph_and_hist(df_cluster_cut)
```

A livello annuale vengono fuori 2 clusters molto sproporzionati.

Clusters for seasons

Winter: 1 - 11, 52, 53 Spring: 12- 25 Summer: 26 - 38 Autumn: 39 - 51

WINTER

```
clus_WINTER <- numeric(0)
for (i in 1:dim(mat_cluster)[1])
{
    clus_WINTER[i] <- as.numeric(names(table(mat_cluster[i,c(1:11, 52,53)]))[which.max(table
(mat_cluster[i,c(1:11, 52,53)]))])
}
clus_WINTER
```

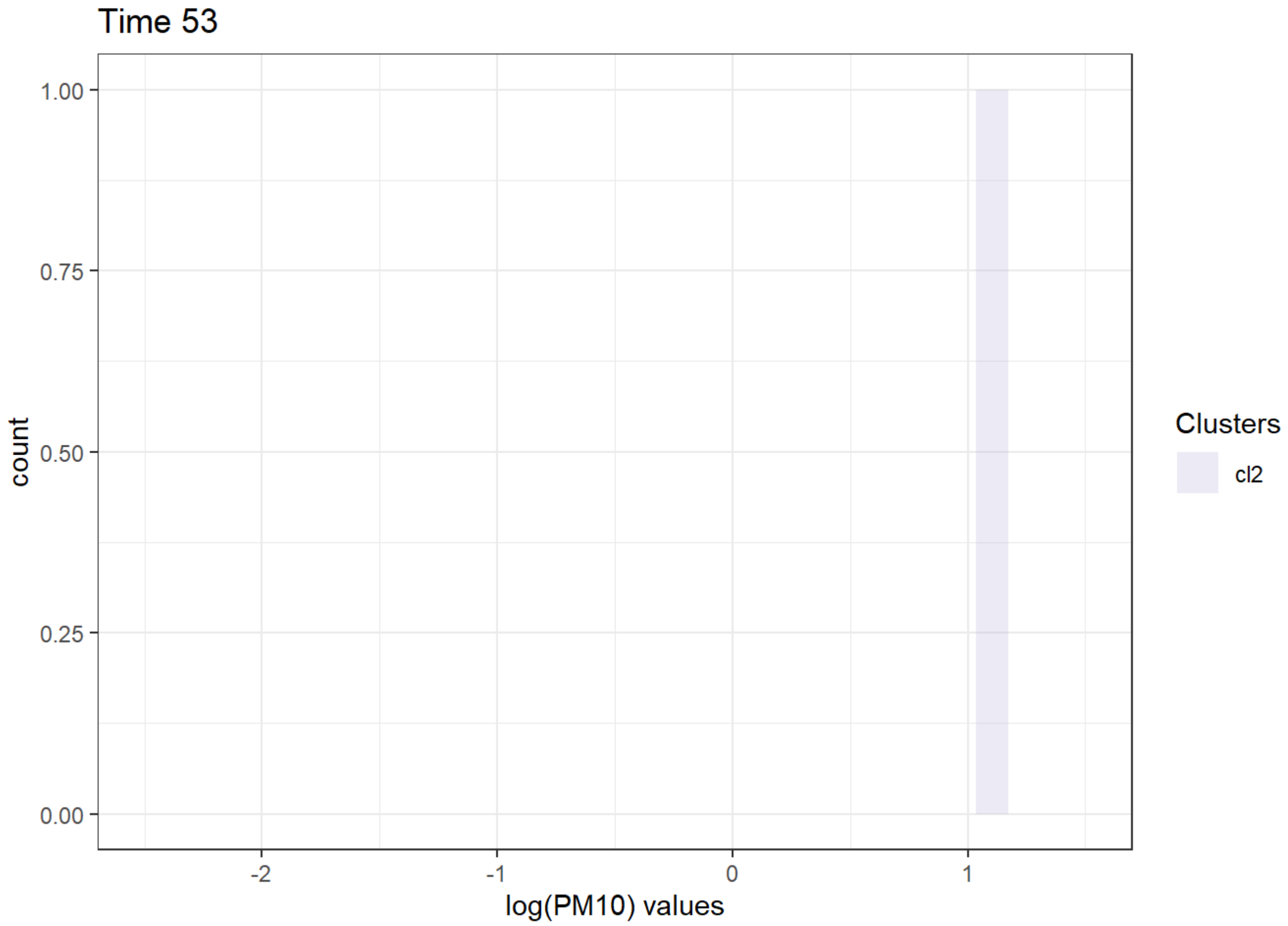
```
##   [1] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 2 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
##  [38] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 2 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
##  [75] 1 1 1 1 1 2 1 1 1 1 1 1 1 1 1 1 1 2 1 1 1 1 1 1 1 1 2 1 1 1 1
```

```
df_temp = data.frame(
    Longitude = unique(df_weekly$Longitude),
    Latitude = unique(df_weekly$Latitude),
    clusters = clus_WINTER
)

df_temp$Time = rep(1,dim(df_temp)[1])
df_cluster_cut = df_temp



### Hist plot
# p = get_hist_color_plot(df_cluster_cut)
p = get_hist_fill_plot(df_cluster_cut) # choose one of these two
print(p)
```

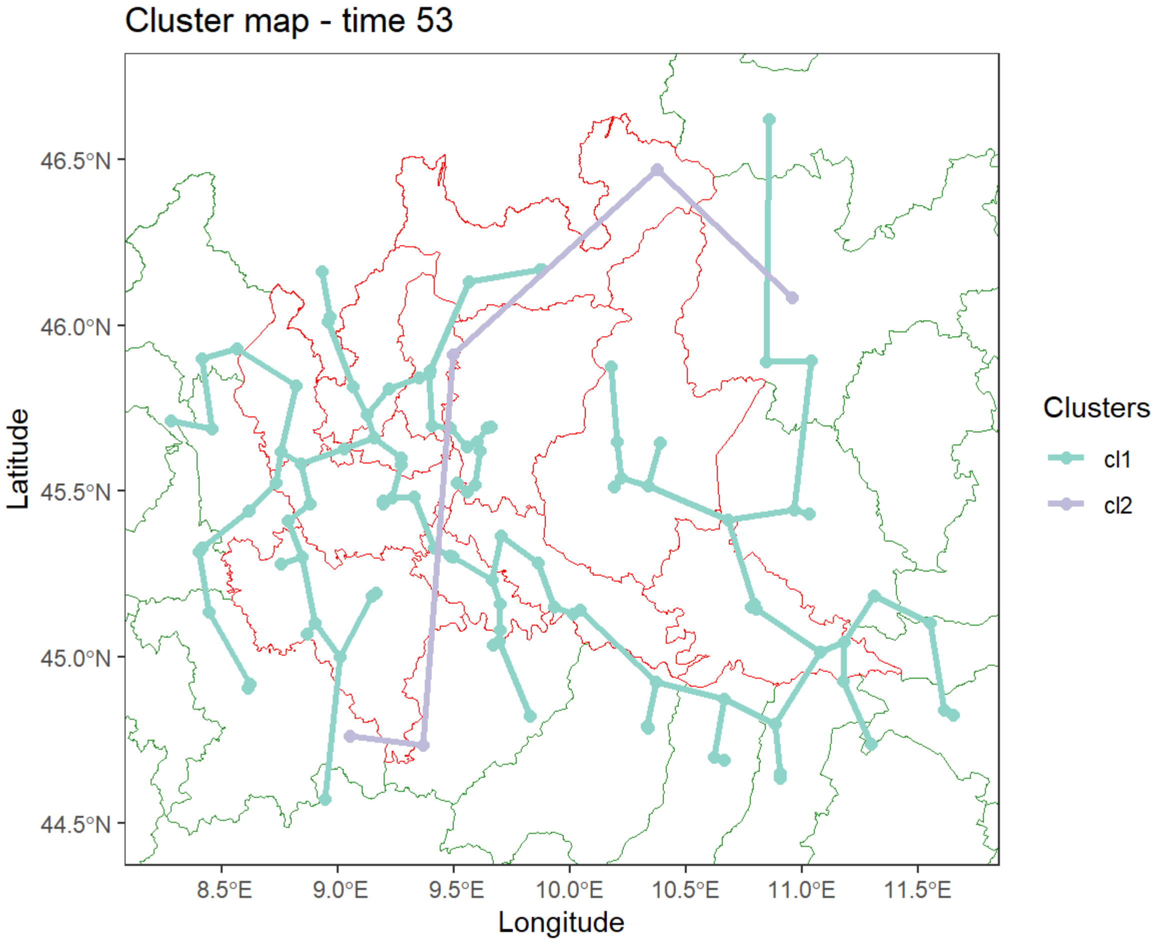
```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```

```
## Warning: Removed 104 rows containing non-finite values (`stat_bin()`).
```

```
## Warning: Removed 2 rows containing missing values (`geom_bar()`).
```



```
### Graph plot
q = get_graph_plot(df_cluster_cut)
print(q)
```



```
# or both together with
# plot_graph_and_hist(df_cluster_cut)
```

SPRING

```
clus_SPRING <- numeric(0)
for (i in 1:dim(mat_cluster)[1])
{ clus_SPRING[i] <- as.numeric(names(table(mat_cluster[i,c(12:25)]))[which.max(table(mat_clus
ter[i,c(12:25)]))])
}
clus_SPRING
```

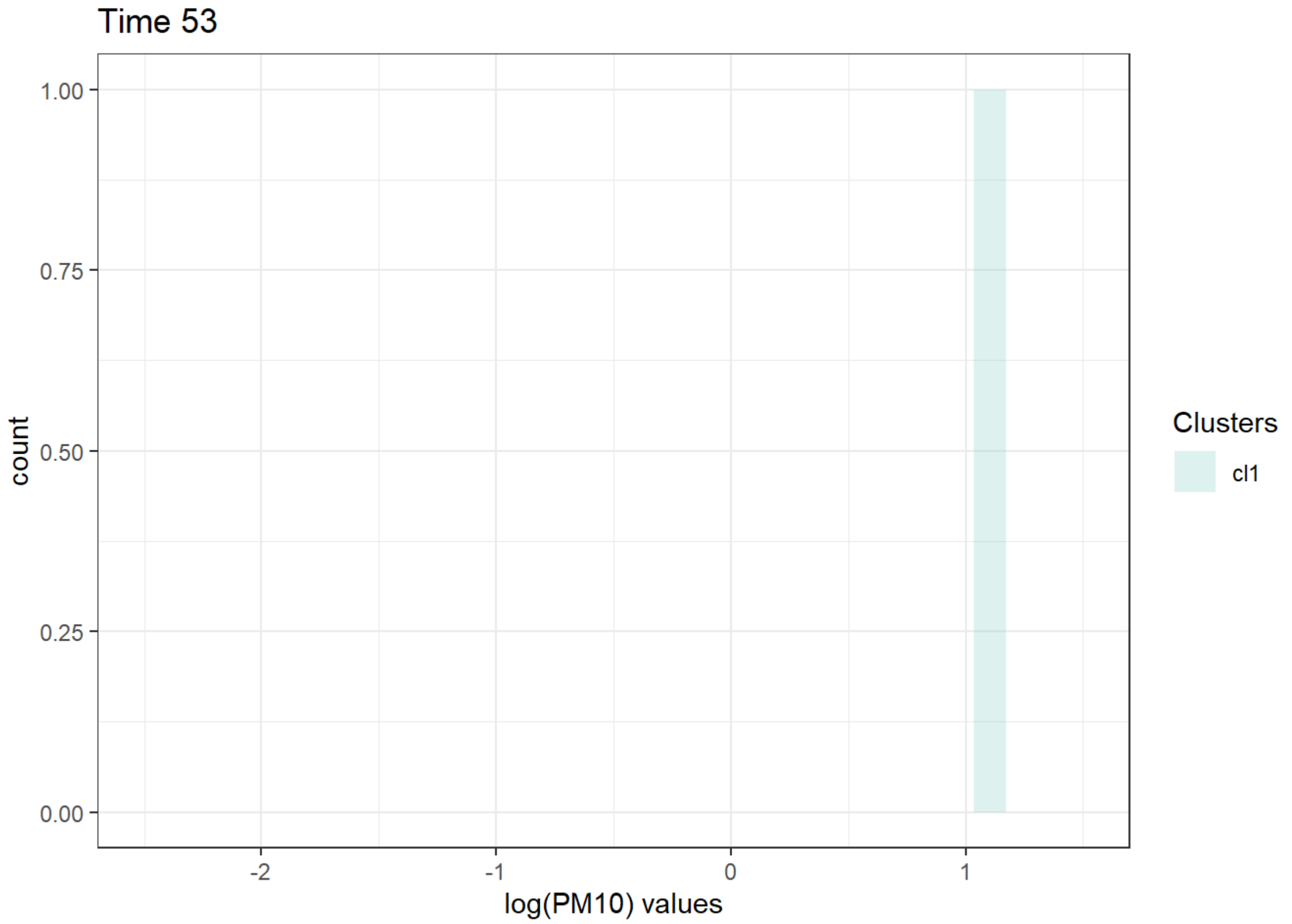
```
##   [1] 1 2 2 2 2 2 2 2 2 2 2 2 2 1 1 1 1 1 1 1 1 1 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
##  [38] 2 2 1 1 1 2 2 2 2 2 2 2 2 1 1 2 2 2 2 2 2 2 2 2 1 2 2 1 1 1 2 1 2 1 2 2 1
##  [75] 2 2 1 2 2 1 2 2 1 1 2 2 2 2 2 2 2 1 2 2 2 2 2 2 2 2 1 1 1 2 2
```

```
df_temp = data.frame(
    Longitude = unique(df_weekly$Longitude),
    Latitude = unique(df_weekly$Latitude),
    clusters = clus_SPRING
)

df_temp$Time = rep(1,dim(df_temp)[1])
df_cluster_cut = df_temp




### Hist plot
# p = get_hist_color_plot(df_cluster_cut)
p = get_hist_fill_plot(df_cluster_cut) # choose one of these two
print(p)
```

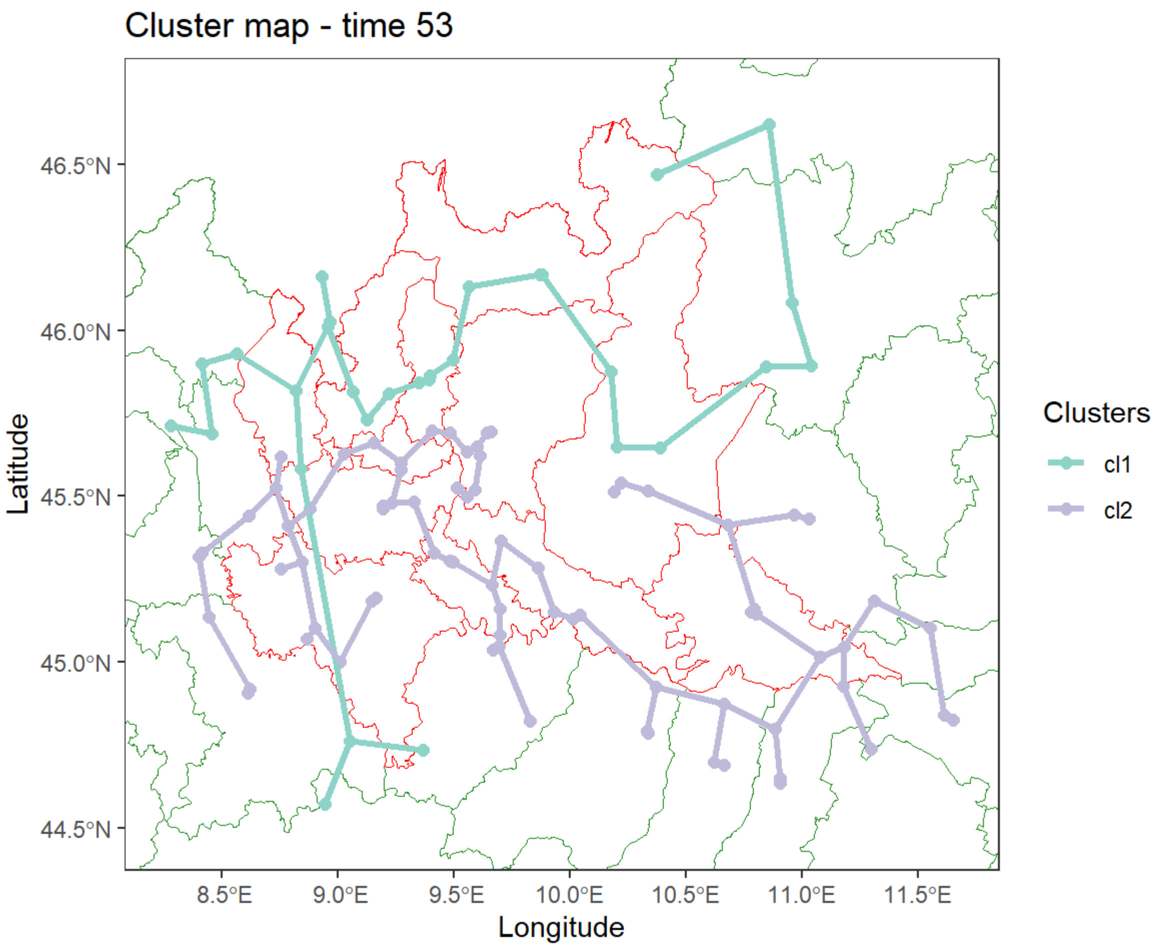
```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```

```
## Warning: Removed 104 rows containing non-finite values (`stat_bin()`).
```

```
## Warning: Removed 2 rows containing missing values (`geom_bar()`).
```



```
### Graph plot
q = get_graph_plot(df_cluster_cut)
print(q)
```



```
# or both together with
# plot_graph_and_hist(df_cluster_cut)
```

SUMMER

```
clus_SUMMER <- numeric(0)
for (i in 1:dim(mat_cluster)[1])
{ clus_SUMMER[i] <- as.numeric(names(table(mat_cluster[i,26:38]))[which.max(table(mat_cluster
[i,26:38]))])
}
clus_SUMMER
```

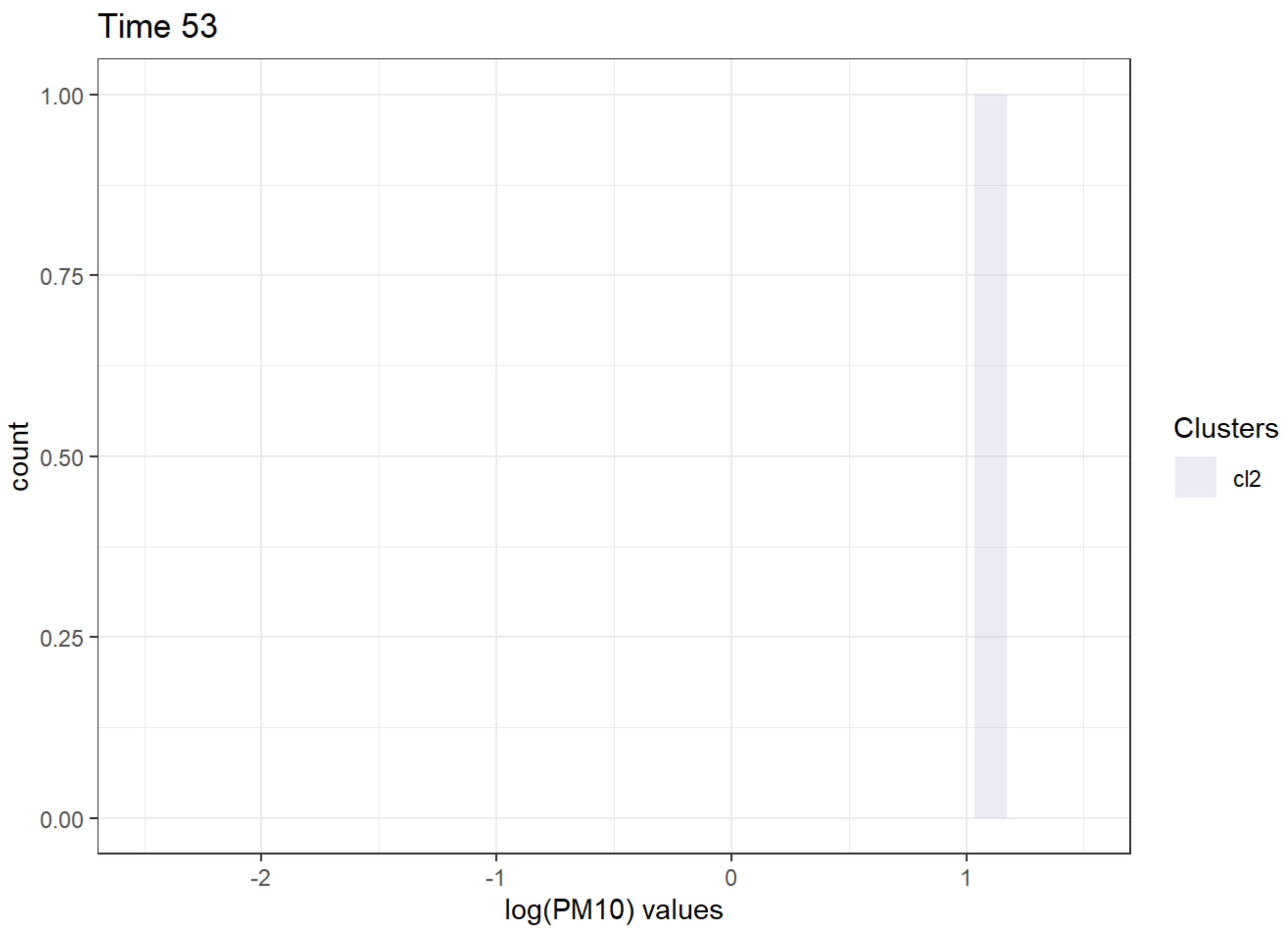
```
##   [1] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 2 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
##  [38] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 2 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
##  [75] 1 1 1 1 1 2 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
```

```
df_temp = data.frame(
    Longitude = unique(df_weekly$Longitude),
    Latitude = unique(df_weekly$Latitude),
    clusters = clus_SUMMER
)

df_temp$Time = rep(1,dim(df_temp)[1])
df_cluster_cut = df_temp



### Hist plot
# p = get_hist_color_plot(df_cluster_cut)
p = get_hist_fill_plot(df_cluster_cut) # choose one of these two
print(p)
```

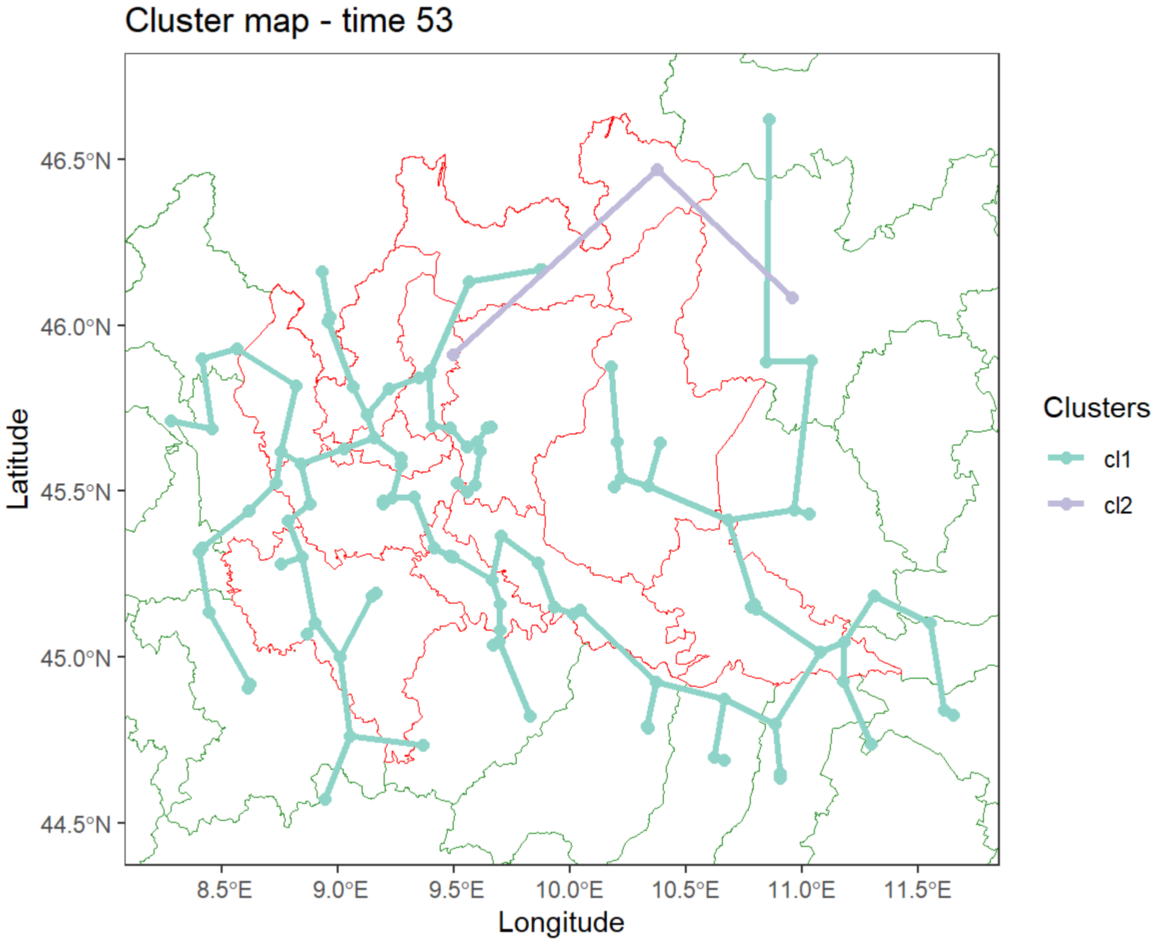
```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```

```
## Warning: Removed 104 rows containing non-finite values (`stat_bin()`).
```

```
## Warning: Removed 2 rows containing missing values (`geom_bar()`).
```



```
### Graph plot
q = get_graph_plot(df_cluster_cut)
print(q)
```



```
# or both together with
# plot_graph_and_hist(df_cluster_cut)
```

AUTUMN

```
clus_AUTUMN <- numeric(0)
for (i in 1:dim(mat_cluster)[1])
{ clus_AUTUMN[i] <- as.numeric(names(table(mat_cluster[i,39:51]))[which.max(table(mat_cluster
[i,39:51]))])
}
clus_AUTUMN
```

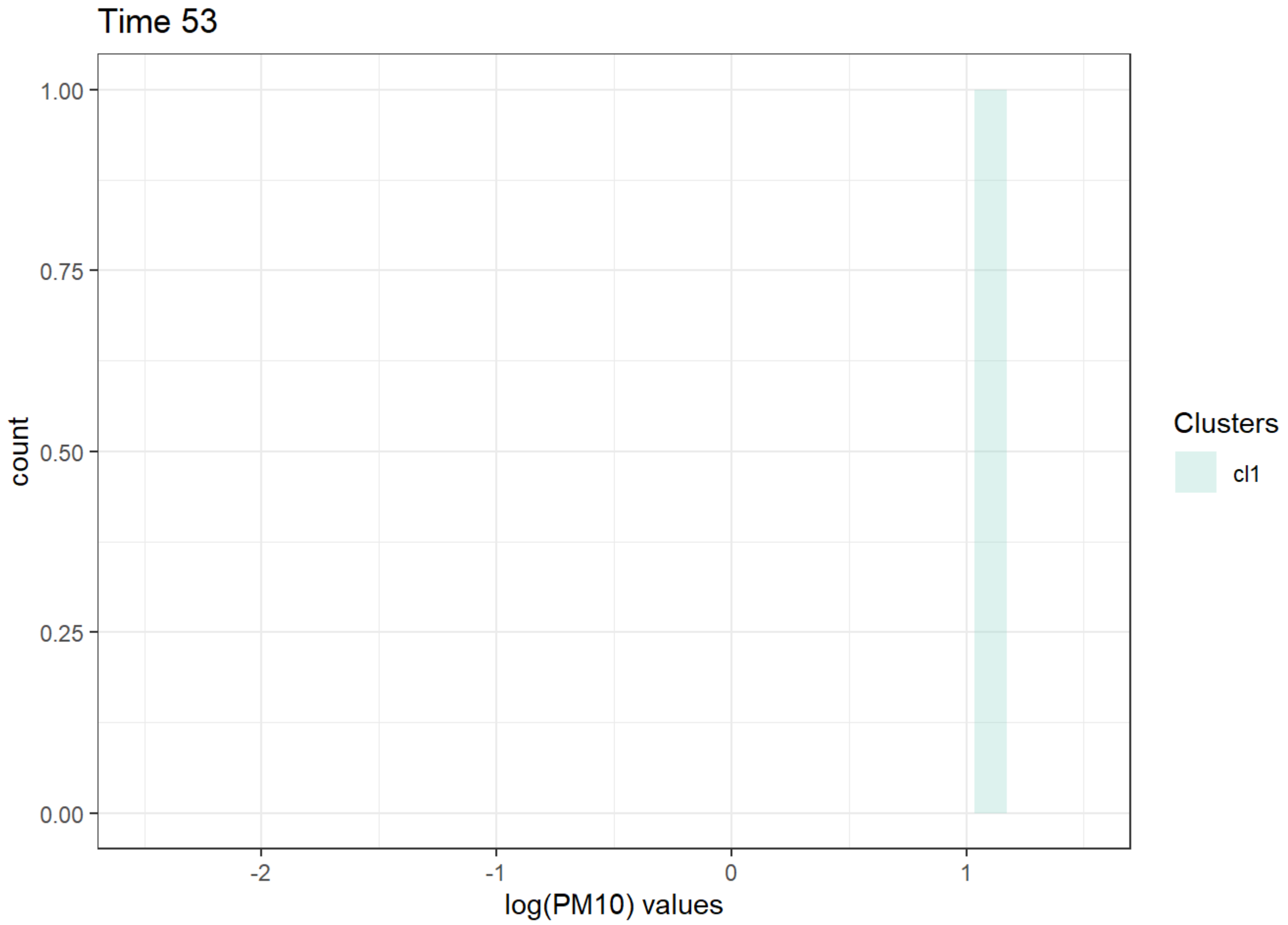
```
##   [1] 1 2 2 2 2 2 2 2 2 2 2 2 2 2 1 1 1 2 1 1 1 1 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
##  [38] 2 2 1 1 1 2 2 2 2 2 2 2 2 1 1 2 2 2 2 2 2 2 2 2 1 2 2 1 1 1 2 1 2 1 2 2 1
##  [75] 2 2 1 2 2 1 2 2 1 1 2 2 2 2 2 2 2 1 2 2 2 2 2 2 2 2 1 1 1 2 2
```

```
df_temp = data.frame(
    Longitude = unique(df_weekly$Longitude),
    Latitude = unique(df_weekly$Latitude),
    clusters = clus_AUTUMN
)

df_temp$Time = rep(1,dim(df_temp)[1])
df_cluster_cut = df_temp




### Hist plot
# p = get_hist_color_plot(df_cluster_cut)
p = get_hist_fill_plot(df_cluster_cut) # choose one of these two
print(p)
```

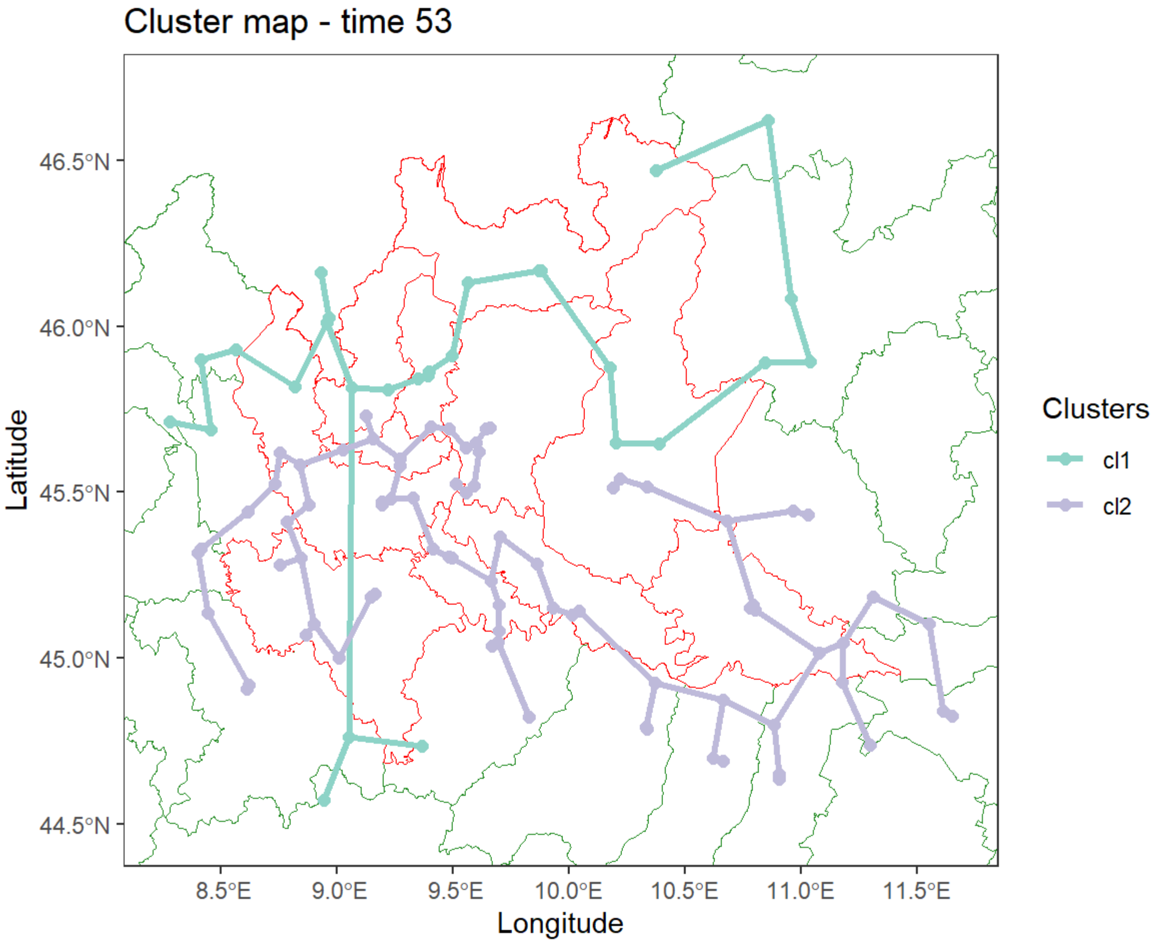
```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```

```
## Warning: Removed 104 rows containing non-finite values (`stat_bin()`).
```

```
## Warning: Removed 2 rows containing missing values (`geom_bar()`).
```



```
### Graph plot
q = get_graph_plot(df_cluster_cut)
print(q)
```



```
# or both together with
# plot_graph_and_hist(df_cluster_cut)
```