Politecnico di Milano

Mathematical Engineering - Bayesian Statistics

A.Y. 2023/2024

Vehicles entering the gates for area C in Milano

Authors:

Marco Vivian
Margherita Brogi
Rossana Cometa
Ardiana Prekazi
Alessandra Sala
Nick Usubelli

Tutors:

Alessandro Carminati
Michela Frigeri

Milan, February 15th 2024

GitHub repository:
https://github.com/nickselv/Bayesian_Statistics_Project

Contents

1 Introduction

This project focuses on the spatio-temporal analysis of vehicle entries in the gates of *Area C* in Milan, a pivotal element in the city's mobility dynamics.

The *Area C* in Milan is a restricted traffic zone in the city center, implemented to reduce traffic congestion and improve air quality. The area includes the historical center and some surrounding zones. The access to this zone from Monday to Friday and at specific hours requires the payment of a fee, while on weekends and holidays, entry is unrestricted. The imposed fee seeks to discourage unnecessary use of private vehicles, encouraging more sustainable alternatives such as public transportation, bicycles, and electric scooters.

The goal of the project is to understand space and time patterns and trends of vehicles entries, taking into account the diverse factors that could characterize the *Area C* gates.

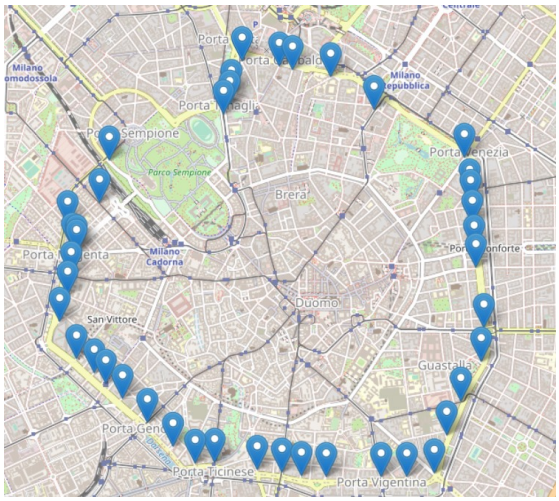


Figure 1: Area C gates

2 The Data

The data used for this analysis have been extracted from the official website of *Comune di Milano*. Information on the entries from the 40 gates surrounding Area C, such as the number and the times of passages, the gates, the vehicle categories and more, were collected monthly and scattered among 12 datasets.

Given the complexity and the large amount of information provided by the datasets at our disposal (over 1.1 million rows), a meticulous work of data selection and filtering was done trough the implementation of few user-defined functions. We grouped all entries daily without distinguishing between vehicle categories or fuel types. We also did not differentiate between entries for residents and non-residents or for different categories of vehicles. As for spatial data, we combined data containing latitude and longitude coordinates for each gate. We introduced a *festivo* column to distinguish between weekdays and holidays. This addition allows us to analyze vehicle entries more effectively.

In pursuit of our project objective, it is essential to identify the factors that might impact the number of entries in Area C. While exploring potential influences, we deemed it reasonable to incorporate weather data, specifically precipitation levels in millimeters and temperature. These information have been extracted from the website of *Arpa Lombardia*, which is the Regional

Agency for Environmental Protection. The data were collected hourly, grouped on a daily basis and finally merged with the other data. Precipitation was considered as the sum of millimeters recorded in a day, while temperature was averaged, aiming to highlight different trends.

3 Exploratory Analysis

The work begins with a thorough exploratory analysis which takes into consideration both the spatial component and the temporal component of the data.

During the analysis, it could be important to categorize the vehicle entries based on whether Area C was active or inactive at the time of passage. For this reason, we have decided to inspect the distributions of the total entries at the same gates considered earlier, dividing the two scenarios. We only considered a two week period for simplicity. We divided the metropolitan area of Milan in four zones, each corresponding to a cardinal point, and then we randomly selected a gate for each zone: Volta to the north, Porta Genova to the south, San Vittore to the west, and Mascagni to the east.

Figure 2: Area C on and off, respectively

In the first graph of Figure 2, the number of entries is zero during the weekend because Area C remains inactive throughout the day, while it remains relatively constant from Monday to Friday. The second one, on the other hand, exhibits an opposite trend during the weekend, showing peaks of accesses, while over the course of the week, it seems to display a slightly increasing trend in entries.

If we want to make this distinction, it is more appropriate to check the trend when the temporal component is determined by the time slot rather than the days because, within the same day, Area C is active at certain times and inactive at others. The graph in Figure 3 shows the number of entries for a week at Porta Genova gate. The time component is given by 1-hour time slots. Passages during the weekdays (blue lines) increase when Area C is deactivated: in the morning until 7:30 AM and in the evening after 7:30 PM. When Area C is active, entries slightly decrease after the initial peak and then remain relatively constant. On weekends (red lines), there is a more uniform trend as Area C is always inactive, with higher entries in the evenings.

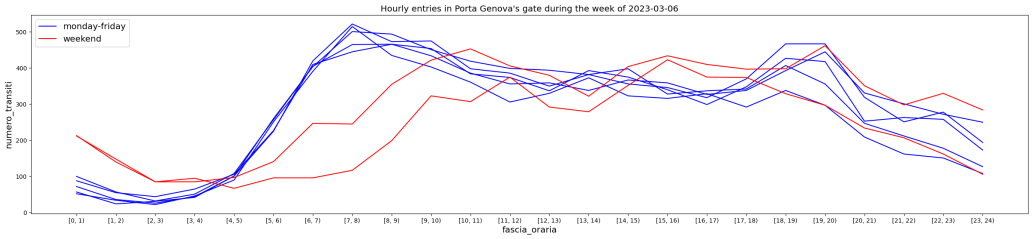


Figure 3: Area C hourly entries over one week

In Figure 4, we plotted a time series representing the total number of entries in Area C day by day over the 12 months considered, without distinguishing between active and inactive Area C. It is evident that for all the considered gates, despite a significant difference in the total number of passages at the represented gates, the trend is similar: the number of entries decreases during the summer period, especially in the month of August, and also during the Easter holidays. On a weekly basis, we observe minima during the weekend, particularly on Sundays. This is reasonable, as the majority of entries can be presumed to be associated with work-related purposes. It is noteworthy that Porta Genova appears to be the most crowded area, as expected, aligning with its notoriety.

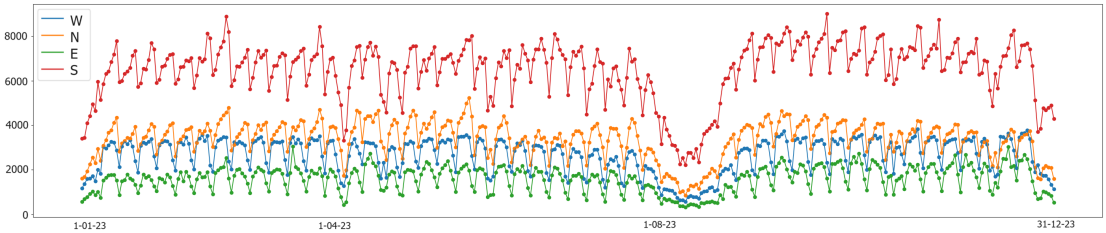
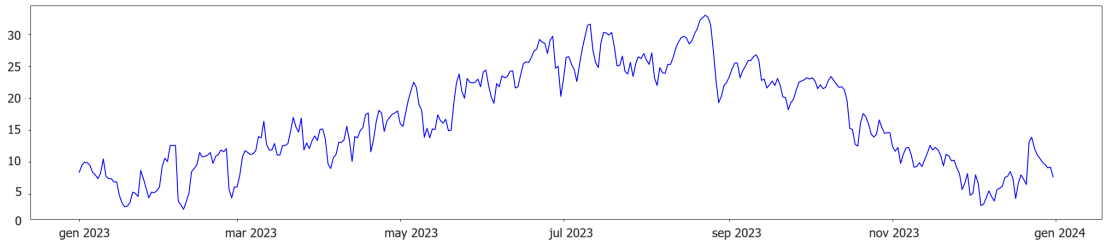


Figure 4: Area C entries time series

As previously mentioned, factors that could impact the number of entries are the temperature and the rain. We can think that, logically, people may be more inclined to use their cars when the temperature is lower or when there is some form of precipitation. The first graph in Figure 5 represents the time series of the temperature collected hourly during the time frame considered, where each blue bar ranges from the minimal to the maximal temperature registered daily. On the other hand, the second graph represents the rainfall trend in Milan throughout the same period.



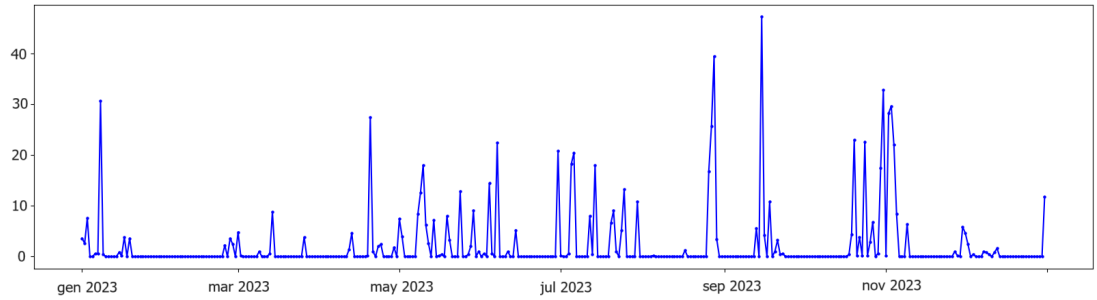


Figure 5: Temperature and rain time series

By visually comparing Figures 4 and 5, it cannot be definitively stated that there is a strong correlation between the meteorological variable and the number of entries. However, their relationship will need to be analyzed through a regression model to understand the seasonal impact on the number of entries in Area C.

4 Spatio-Temporal Model

In order to pursue the goal of describing the number of entries in Area C in Milan we will make use of Bayesian spatio-temporal models. Specifically, a regression model in which the response variable takes on count values is the Poisson model. This will allow us to monitor the number of passages registered through the 40 gates of Area C, exploiting their spatial correlation. The time component will also play a pivotal role in the analysis.

4.1 Dataset and model

To prepare for subsequent analysis, we merged all the mentioned data into a single dataset containing the following variables:

- **data**: date from January 1st 2023 to December 31st 2023;
- **numero_transiti**: number of passages in Area C;
- **festivo**: binary variable where 0 indicates a weekday and 1 is for weekends and holidays;
- **longitude**: the longitude coordinate;
- **latitude**: the latitude coordinate;
- **nome varco**: the name of the gate;
- **precipitazioni**: sum in millimeters of precipitation recorded;
- **av_temp**: average temperature recorded, in Celsius degrees;

Starting from those initial covariates, we made some key changes in order to make our model more accurate and efficient:

- We added two dummy variables: **estate** and **inverno**, which are equal to 1 when **data** is respectively a summer day and a winter day, since, as the exploratory analysis shows, the entries situation drastically changes during summer. It's plausible to think that way fewer people will drive to Milan during summer, both because they won't go there at all and because they're more likely to choose a different type of transport rather than driving.

- We chose to standardise the rain and temperature to properly confront their effects in spite of their different units of measure (mm vs °C)

4.2 First glance of the model

$$Y_{it}|\lambda_{it} \overset{\text{ind}}{\sim} Pois(\lambda_{it})$$
$$log(\lambda_{it}) = f(t) + c_i$$

where:

- $\mathbf{i} \in \{1, \ldots, 40\}$ gates
- $\mathbf{t} \in \{1, 2, \ldots, 365\}$ time instants corresponding to each day of the year
- $\boldsymbol{f}(t)$ time-dependent function
- $\mathbf{c_i}$ gate-specific intercept

4.3 Modeling priors for $\mathbf{c_i}$

To begin, the idea is to model the intercept c_i, $i = 1, .., 40$, so that each gate i is shifted according to some gate-specific factor. This can be achieved by fitting an appropriate regression model with the package *CARBayesST* in R. *CARBayesST* allows to implement spatio-temporal generalised linear mixed models for areal unit data, with inference in a Bayesian setting using Markov Chain Monte Carlo (MCMC). The spatio-temporal autocorrelation is modeled by random effects, which are assigned conditional autoregressive (CAR) style prior distributions.

CARBayesST can fit the following generalized linear mixed model for Poisson distributed data:

$$Y_{it}|\lambda_{it} \overset{\text{ind}}{\sim} Pois(\lambda_{it})$$
$$log(\lambda_{it}) = \mathbf{x}_{it}^T\boldsymbol{\beta} + O_{it} + \phi_{it}$$

where Y_{it} are the number of entries in gate $i, i = 1, ..., 40$ and on day $t, t = 1, ..., 365$, λ_{it} denotes the expectation of the response variable, $\boldsymbol{\beta}$ is the vector of covariates, O_{it} is the offset for gate i and day t, and ϕ_{it} is a latent component for areal unit i and time period t encompassing one or more sets of spatio-temporally autocorrelated random effects. For the purpose of this part of the work, where we aim to determine a shift parameter for subsequent analysis, the average number of entries in each gate of the previous year (2022) is used as the offset term ($O_{it} = log(\lambda_{it})$), without incorporating any additional covariate except the intercept.

Spatial autocorrelation is controlled by a symmetric non-negative KxK (with $K = 40$ in this case) proximity matrix $\mathbf{W} = [w_{ij}]$, where w_{ij} represents the spatial closeness between two areal units (s_i, s_j). The larger the values the closer the two areal units considered are. Very small values correspond to areas which are not spatially close. It is often assumed that $\mathbf{W}$ is binary with $w_{ij} = 1$ if (s_i, s_j) share a common border. In addition, $w_{ii} = 0$ for all areal units i.
To determine the most suitable coefficients c_i, we initially construct three distinct proximity matrices:

1. $\mathbf{W_1}$: binary proximity matrix where $w_{ij} = 1$ if the gates i and j are adjacent, $w_{ij} = 0$ otherwise;
2. $\mathbf{W_2}$: we assign decreasing values starting from 1 for adjacent gates, 0.5 if gate i is one gate away from j, 0.25 if gate i is two gates away from j and so forth;

3. $\mathbf{W_3}$: we assign values according to the distance between gates (Euclidean distance computed with latitude and longitude coordinates).

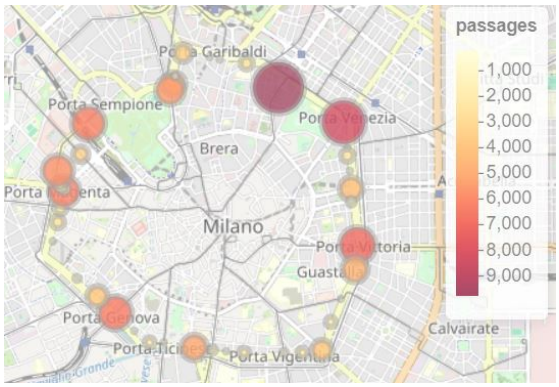


Figure 6: Average number of entries in Area C gates

Before proceeding, we only need to choose the most suitable model for $\boldsymbol{\phi}$. Among the various models provided in the package we are using, the *ST.CARar* model seems to be the most appropriate. This model, leveraging a first-order temporal autoregressive process, is well-suited if one wishes to estimate the evolution of the spatial structure in the data over time, which is our case.

One model for each different proximity matrix $\mathbf{W}$ is then fitted, and the model fit criteria results are stored in Table 1.

	W_1	W_2	W_3
DIC	138040.004	137866.878	137831.428
WAIC	135937.018	135386.108	135198.682
LMPL	-72195.480	-71979.841	-71778.294
Log-likelihood	-58053.025	-57867.418	-57789.947

Table 1: Model fit criteria

It is clear that the three models perform similarly. The third model displays the lowest values of the Deviance Information Criterion (DIC), Widely Applicable Information Criterion (WAIC), and the highest values of Log Marginal Predictive Likelihood (LMPL) and Log-likelihood, making it the best one. This aligns with our expectations, as the model's proximity matrix $\boldsymbol{W_3}$ is constructed to take into account the actual distances between the Area C gates. Hence, c_i estimates of this model will be used as prior means in the following analysis.

Further specifications of the models are included in the Appendix 6.1.

4.4 STAN model

In this section, we didn't consider *latitude* and *longitude* and replaced them with a new index $\mathbf{i} = 1, ...40$ that indicates the label of the specific gate. We chose to not take into account the actual spatial coordinates of the gates but only a label, since the ring structure of the gates was already taken into account when modelling the priors of the $\mathbf{c_i}$'s and our model aims to analyse the daily entries through the 40 gates, rather than performing a Bayesian Kriging on a new location.

$$Y_{it}|\lambda_{it} \overset{\text{ind}}{\sim} Pois(\lambda_{it})$$
$$log(\lambda_{it}) = f(t) + \beta_i X_t + c_i$$

where:

- $\mathbf{i} \in \{1, \dots, 40\}$
- $\mathbf{t} \in \{1, 2, \dots, 365\}$ time instants corresponding to each day of the year
- $\boldsymbol{f}(t)$ Fourier series
- $\mathbf{c_i}$ gate-specific intercept
- $\mathbf{X_t}$ values of weather conditions on day t
- $\beta_\mathbf{i}$ coefficient that takes into account the effect of $\mathbf{X_t}$ on the different gates

4.4.1 Priors and model choices

- $\mathbf{f}(t) = a_0 sin(\omega_1 t) + b_0 cos(\omega_1 t) + a_1 sin(\omega_1 t) + b_1 cos(\omega_2 t)$, where
  - $\boldsymbol{\omega}_1 = 2\pi/7$ to account for the weekly periodicity
  - $\boldsymbol{\omega}_2 = 2\pi/226$ for the seasonal periodicity
  - $a_0, b_0, a_1, b_1 \overset{\text{iid}}{\sim} N(0, 6)$
- $\mathbf{c}_i \overset{\text{ind}}{\sim} N(log(\mu_c), 0.3)$ where μ_c is the vector of values estimated in section 4.2
- $\beta_\mathbf{i} \overset{\text{iid}}{\sim} N(0, 1)$

4.4.2 Preliminary analysis on $\beta_\mathbf{i}$

Starting from model (1), we plotted the 95% credible intervals of the different $\beta_\mathbf{i}$ along the gates.

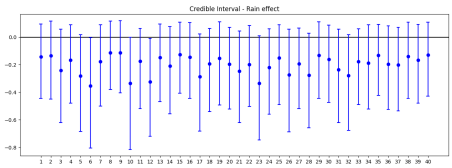


Figure 7: CI of $\beta_{\mathbf{pioggia}}$

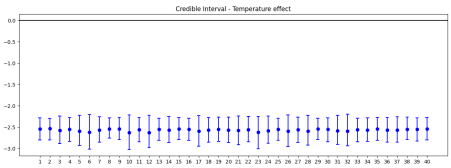


Figure 8: CI of $\beta_{\mathbf{temperatura}}$

From the $\beta_{\mathbf{pioggia}}$ and $\beta_{\mathbf{temperatura}}$ credible intervals we can see how they are all very similar to each other, meaning that rain has similar influence among the different gates on the number of vehicles entries. The same holds for $\beta_{\mathbf{temperatura}}$.

As for $\beta_{\mathbf{estate}}$ and $\beta_{\mathbf{inverno}}$ we can notice how they both slightly vary between gates, which leads us to assume that seasonality has a different effect on different gates. However, for the sake of computational efficiency, we preferred to consider a β independent of i, since considering $\beta_\mathbf{i}$ would add an additional complexity to the model without resulting in a better explanation of the variability of our data.

Hence, for our analysis, we proceeded by considering a 1x4 vector β (where the effect of weather conditions is assumed to be equal for all the gates) that is multiplied by X_t.
Therefore, the new model can be written as:

$$Y_{it}|\lambda_{it} \overset{\text{ind}}{\sim} Pois(\lambda_{it})$$
$$log(\lambda_{it}) = f(t) + \beta X_t + c_i$$

Traceplots

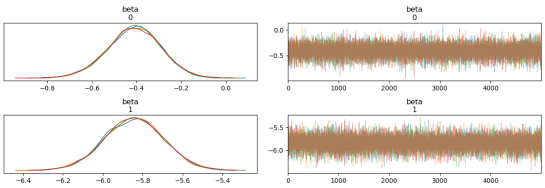


Figure 9: Traceplots of $\beta_{\textbf{pioggia}}, \beta_{\textbf{temperatura}}$

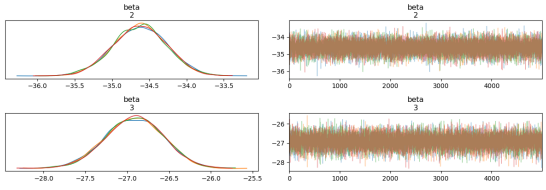


Figure 10: $\beta_{\textbf{estate}}, \beta_{\textbf{inverno}}$

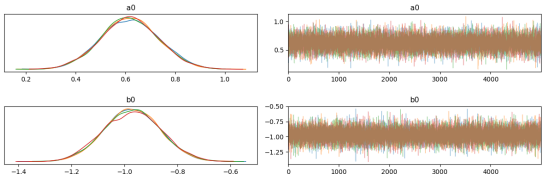


Figure 11: Traceplots of weekly coefficients

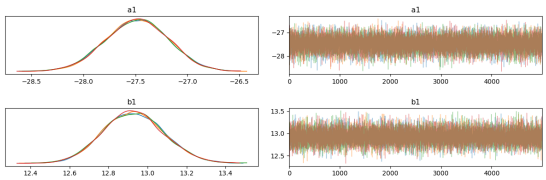


Figure 12: Traceplots of seasonal coefficients

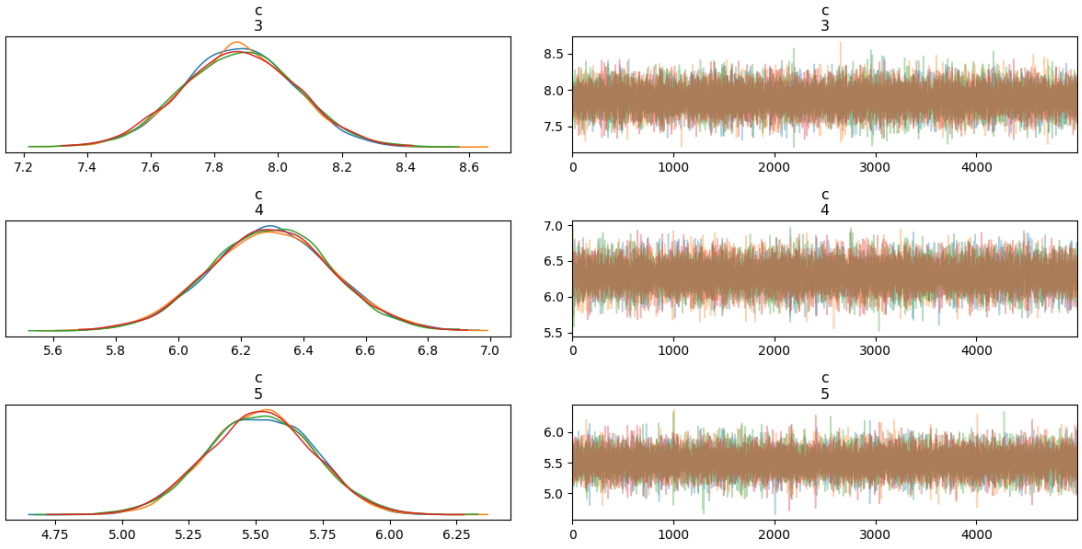


Figure 13: Traceplots of c[i]

4.4.3 Comments on the fitted model

The fit process was performed using 1000 as burn-in iterations and 5000 as sampling iterations. Looking at the traceplots it's clear how all the distributions we reached seem to be stationary,

thus indicating the convergence of the chains. In particular all the c_i are centered around positive values indicating that their presence contributes to increase significantly the parameter lambda of our model. Moreover some of them tend to assume different values indicating some disparity between gates. We'll analyse their possible cluster structure in the following section.

Talking about the $\beta_{\mathbf{pioggia}}$ and $\beta_{\mathbf{temperature}}$ they're centered around values which are negative and close to the zero, indicating a little (negligible) contribution to the lambda parameter. On the other hand $\beta_{\mathbf{estate}}$ and $\beta_{\mathbf{inverno}}$ are concentrated around negative and definitely far from zero values; This agrees with the fact that summer and winter days correspond to fewer vehicles entering Area C on average.

As last thing, a comparison between the coefficients of the Fourier series can be made trough the traceplots too: the coefficients a_0 and b_0 appear to be on average way closer to zero with respect to a_1 and b_1. This can be interpreted as the seasonality playing a much more important rhole in our model with respect to the weekly pattern.

5 Clustering

After looking at the traceplots and at the different patterns of the time series reported above, our goal here is to perform a univariate clustering on $\mathbf{c_i}$'s, employing a Dirichlet Process Mixture Model grouping the 40 gates into plausible clusters, and possibly give an interpretation on the gate similarities.

5.1 Clustering on $\mathbf{c_i}$

Before performing clustering on the gate-specific intercepts c_i, we plotted the corresponding 95% credible intervals, based on our model.

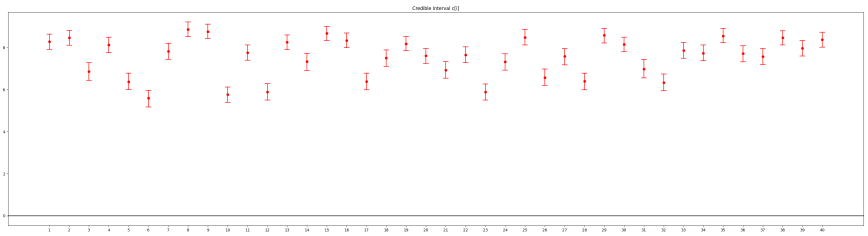


Figure 14: Credible intervals of gate specific intercepts (c[i])

We can observe some similarities among the c_i; which makes us deduce that there might exist some clusters.

5.1.1 Dirichlet Process Mixture Model

We then performed clustering on the c_i's using our STAN model. We followed a Bayesian non-parametric model-based approach, employing a Dirichlet process mixture model with Stick breaking construction. This approach implies a prior distribution on the random partition of the experimental units, determined by the latent variables. The Dirichlet process mixture model allows us to compute the posterior of the random partition of the $\mathbf{c_i}$'s given the data and subsequently estimate their clustering structure through a summary of that posterior distribution.

$$
\begin{aligned}
c_i &\overset{iid}{\sim} \sum_{k=1}^{C} w_k N(\mu_k, \sigma_k^2) && i = 1, \ldots, 40 \\
w_k &= v_k \prod_{i=1}^{k-1} (1 - v_i) && k = 2, \ldots, C \\
w_1 &= v_1
\end{aligned}
$$

with:

- $\mu_k \sim N(0, 10\sigma_k)$
- $\sigma_k^2 \sim \text{InvGamma}(4, 0.5)$
- $v_k \sim \text{Beta}(1, 2)$
- C=10

Further details on our implementation of the DPMM in our STAN model can be found in Appendix 6.4.

5.2 Final results and interpretation

After simulating the MCMC chains with 500 iterations, we can observe that the algorithm is working correctly since the number of non null-clusters is varying at each iteration.

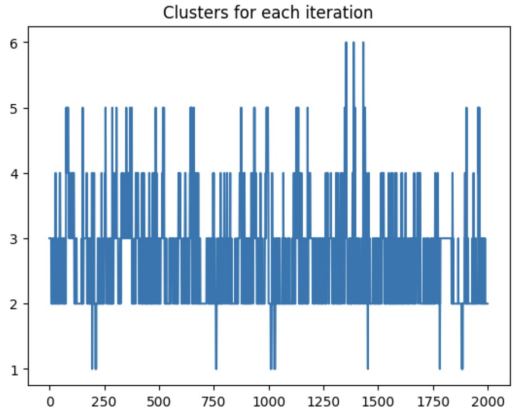


Figure 15: Clusters at each iterations

We managed to find the clusters and the labelling thanks to the SALSO library in R, which is minimizing the posterior expectation of the Binder's loss function. The result is the optimal number of clusters, which is equal to 2, and the vector of labels for each gate specific intercept c. We again plotted the 95% credible intervals, distinguishing between the two different clusters they belong to by highlighting them in two different colors.

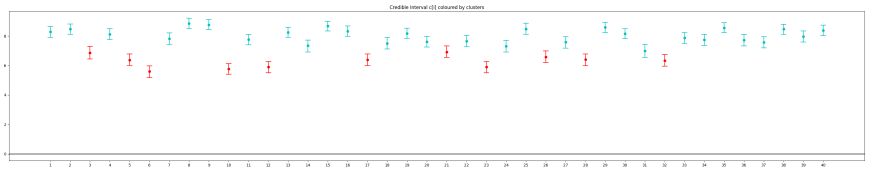


Figure 16: Credible intervals of gate specific intercepts (c[i]) after clustering

To better interpret our results, we decided to also plot the time series of the entries in the 40 gates of the Area C, giving them different colours according to the two clusters.

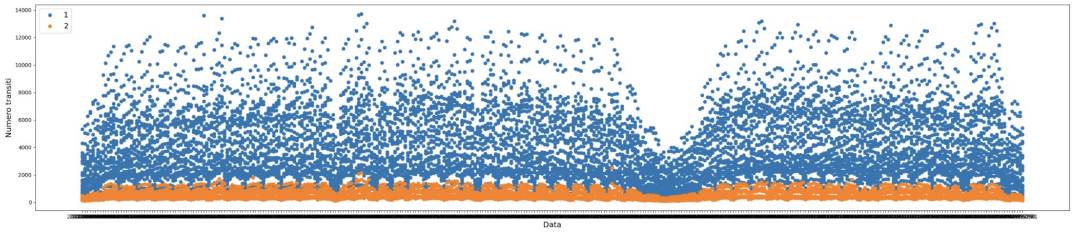


Figure 17: Area C entries time series after clustering

Moreover, for a more visual interpretation of the clustering, we colored the gates on the map according to the same labeling.

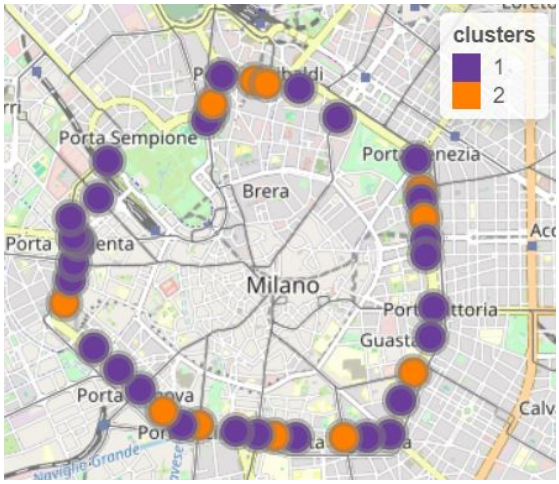


Figure 18: Area C entries map after clustering

Therefore, we can infer from the two plots above that the clustering yields consistent results with the data. Specifically as can be immediately observed, the less numerous cluster corresponds to gates where there are fewer entries.

Moreover, we found out that some of these gates are reserved exclusively for public transport and we inferred that some gates might be less frequented because the areas they're located at are well-served by public transportation.

6 Appendix

6.1 $\mathbf{c_i}$

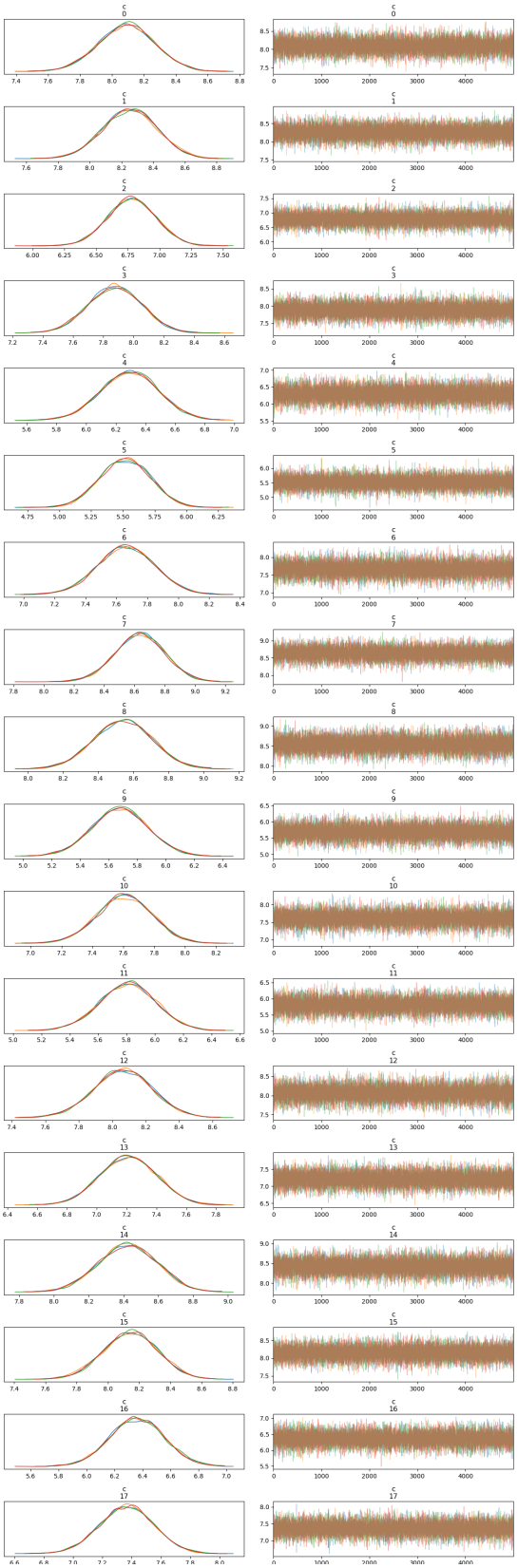
	W_1	W_2	W_3
(Intercept)	-0.0536	-0.0535	-0.0537
τ^2	0.0171	0.0389	0.0369
ρ_s	0.9765	0.9763	0.9800
ρ_t	0.5693	0.5338	0.5373

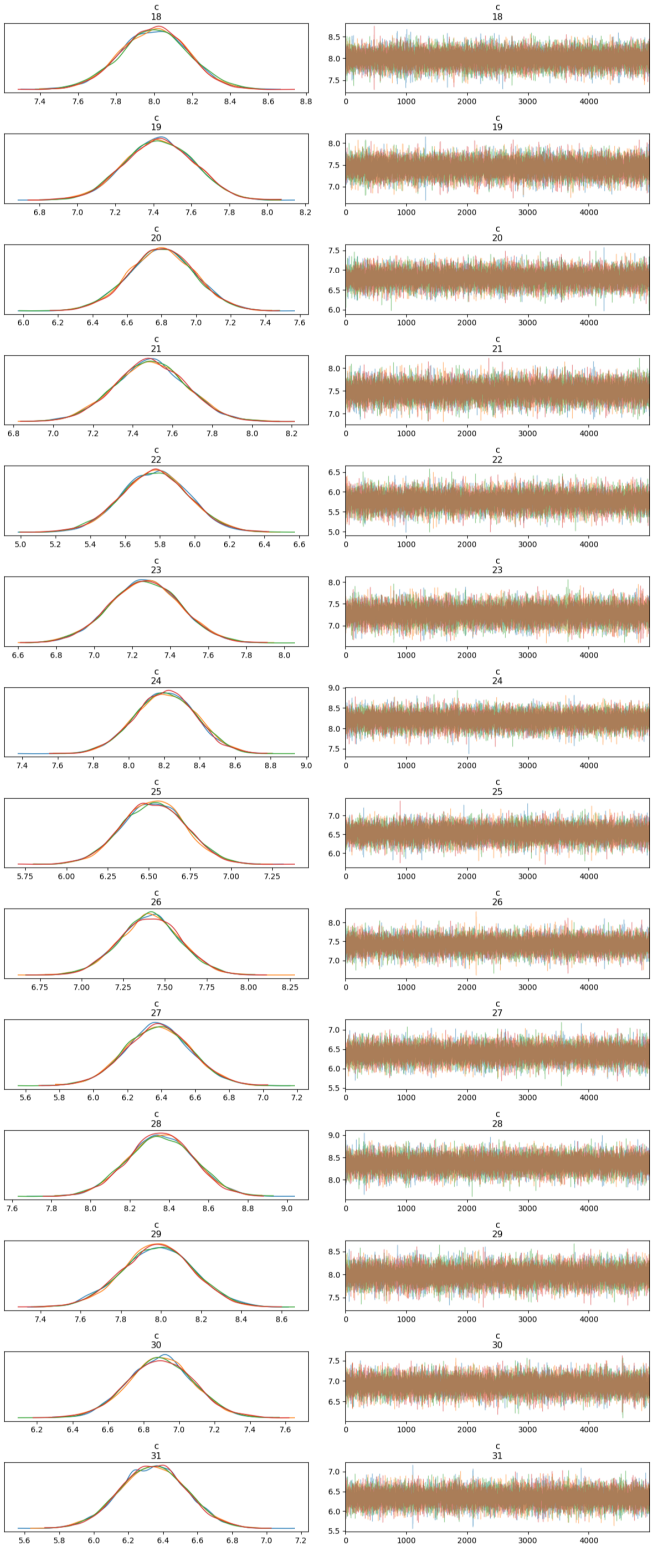
Table 2: Posterior medians of the parameters

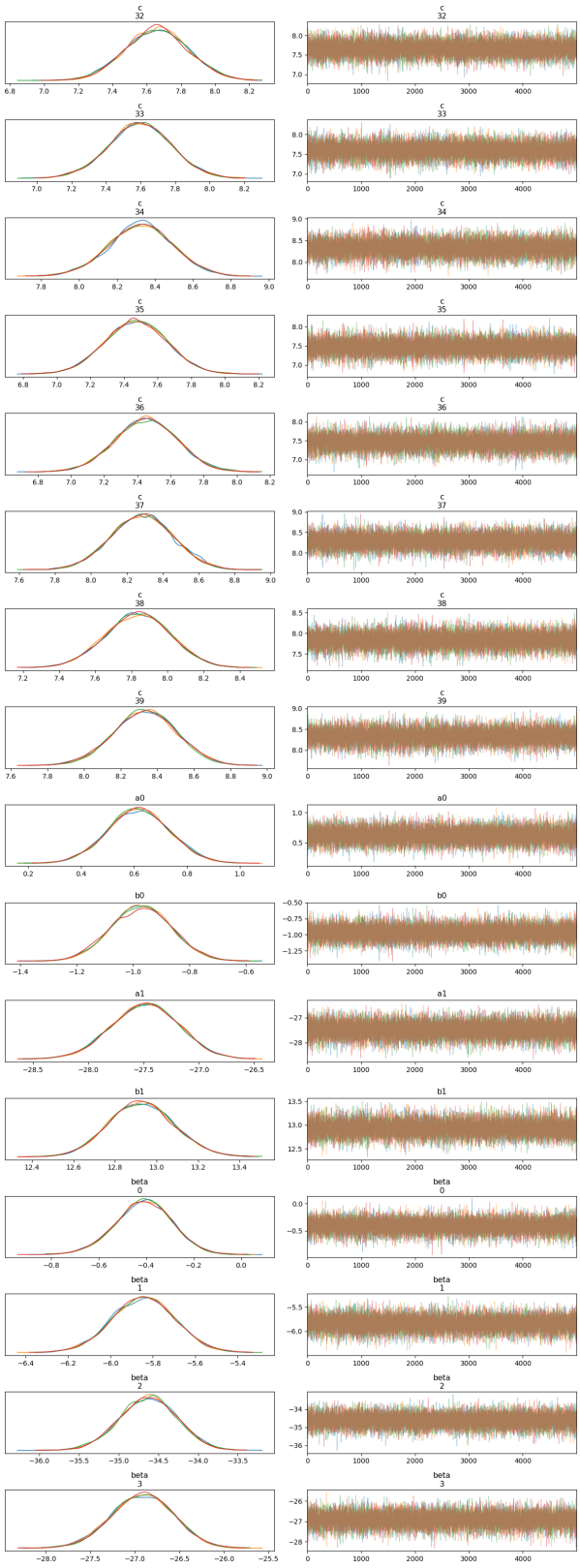
Legend:

- τ^2: quantifies the variability of the spatially correlated random effects across different locations. Lower value indicates less variability in the spatially correlated random effects, implying that neighboring locations have similar random effects.
- ρ_s: quantifies the strength of the spatial correlation between neighboring locations. A higher value indicates stronger spatial autocorrelation, implying that nearby locations are more similar to each other.
- ρ_t: measures the strength of the temporal correlation between consecutive time points. A higher value of indicates stronger temporal autocorrelation, suggesting that observations at adjacent time points are more correlated.

6.2 Further traceplots




c
18
19
20
21
22
23
24
25
26
27
28
29
30
31


c 32
c 33
c 34
c 35
c 36
c 37
c 38
c 39
a0
b0
a1
b1
beta 0
beta 1
beta 2
beta 3

6.3 STAN code

```
stan_code = """
functions {
  matrix f_matr(real a, real b, real c, real d, real w1, real w2, matrix
      t) {
    return a*sin(w1 * t) + b*cos(w1 * t) + c*sin(w2 * t) + d*cos(w2 * t);
  }

}


data {
    int<lower=0> N;
    int<lower=0> T;
    int<lower=0> I;
    array[I,T] real count;
    matrix[I,T] time;
    array[N] real area;
    array[N] real precipitazioni;
    array[N] real av_temp;
    matrix[4,T] Xt;
    matrix[1,I] Xi;
    row_vector[I] mu_c;
    array[I * T] int<lower=0> output_flat;
    array[I,T] int<lower=0>  output;
    real fixed_w1;
    real fixed_w2;

}

parameters {

    row_vector[I] c;
    real a0;
    real b0;
    real a1;
    real b1;
    row_vector[4] beta;

}

model {

    // Priors
    c ~ normal(log(mu_c), 0.2);

    beta ~ normal(0, 2);

    a0 ~ normal(0, 6);
    b0 ~ normal(0, 6);
    a1 ~ normal(0, 6);
    b1 ~ normal(0, 6);


    matrix[I, T] lambda_matrix = f_matr(a0, b0, a1, b1, fixed_w1,
        fixed_w2, time) + rep_matrix(c, T)' + rep_matrix(beta*Xt, I);
```

```
54
55      row_vector[I*T] lambda_row = to_row_vector(lambda_matrix);
56
57      output_flat ~ poisson(exp(lambda_row));
58
59  }
```

6.4 Dirichlet Process Mixture Model in STAN

```
1  stan_code = """
2  functions {
3    matrix f_matr(real a, real b, real c, real d, real w1, real w2, matrix
       t) {
4      return a*sin(w1 * t) + b*cos(w1 * t) + c*sin(w2 * t) + d*cos(w2 * t);
5    }
6
7  }
8
9
10 data {
11      int<lower=0> N;
12      int<lower=0> T;
13      int<lower=0> I;
14      array[I,T] real count;
15      matrix[I,T] time;
16      array[N] real area;
17      array[N] real precipitazioni;
18      array[N] real av_temp;
19      matrix[4,T] Xt;
20      matrix[1,I] Xi;
21      row_vector[I] mu_c;
22      array[I * T] int<lower=0> output_flat;
23      array[I,T] int<lower=0>  output;
24      real fixed_w1;
25      real fixed_w2;
26
27      //clustering:
28      //number of components
29      int<lower=1> C;
30
31      // DP PARAMATER
32      real <lower=0> param;
33
34      //P0 PARAMETER (mu0,s0,a,b)
35      //real mu0;
36      //real<lower=0> s0;
37
38      real<lower=0> aa;
39      real<lower=0> bb;
40 }
41
42 parameters {
43
44      row_vector[I] c;
45      real a0;
46      real b0;
```

```
47      real a1;
48      real b1;
49      row_vector[4] beta;
50
51      //clustering:
52      vector<lower=0>[C] vars;
53      vector[C] m;
54      vector<lower=0, upper=1>[C-1] nus;
55  }
56
57  transformed parameters {
58      //compute standard deviations
59      vector<lower=0>[C] s = sqrt(vars);
60
61      vector<lower=0, upper=1>[C - 1] cumprod_one_minus_nu;
62      cumprod_one_minus_nu = exp(cumulative_sum(log1m(nus)));
63
64      //DP weights
65      simplex[C] ws;
66      ws[1] = nus[1];
67      ws[2:(C - 1)] = nus[2:(C - 1)] .* cumprod_one_minus_nu[1:(C - 2)];
68      ws[C] = cumprod_one_minus_nu[C - 1];
69  }
70
71  model {
72
73      // Priors
74      c ~ normal(log(mu_c), 0.2);
75
76      beta ~ normal(0, 2);
77
78      a0 ~ normal(0, 6);
79      b0 ~ normal(0, 6);
80      a1 ~ normal(0, 6);
81      b1 ~ normal(0, 6);
82
83
84      matrix[I, T] lambda_matrix = f_matr(a0, b0, a1, b1, fixed_w1,
            fixed_w2, time) + rep_matrix(c, T)' + rep_matrix(beta*Xt, I);
85
86      row_vector[I*T] lambda_row = to_row_vector(lambda_matrix);
87
88      output_flat ~ poisson(exp(lambda_row));  //poisson(to_row_vector(
            rep_matrix(coeff,T)').*exp(lambda_row));
89
90
91      // Clustering
92      //prior
93      m[1:C] ~ normal(0, 10*s[1:C]);
94      vars ~ inv_gamma(aa, bb);
95      nus ~ beta(1, param);
96
97      // Finite Mixture Model
98      //likelihood
99      for (i in 1:I) {
100       array[C] real log_probs;
```

```
      for (h in 1:C) {
          log_probs[h] = log(ws[h]) + normal_lpdf(c[i] | m[h], s[h]);
      }
      target += log_sum_exp(log_probs);
    }
}

generated quantities {

    vector[I] cluster_allocs;
    for (i in 1:I) {
      vector[C] log_probs;
      for (h in 1:C) {
          log_probs[h] = log(ws[h]) + normal_lpdf(c[i] | m[h], s[h]);
      }
      cluster_allocs[i] = categorical_rng(softmax(log_probs));
    }
}
```

7 Bibliography

References:

1. David B. Hitchcock, "Bayesian Count Regression Models", *STAT535: Chapter 12*, 2022.
2. D. Lee, A. Rushworth, G. Napier,, "Spatio-Temporal Areal Unit Modeling in R with Conditional Autoregressive Priors Using the CARBayesST Package", 2018.
3. "Bayesian Hierarchical Poisson Regression Model for Overdispersed Count Data", *SAS/-STAT Examples*.
4. Gary L. Rosner, Purushottan W. Laud, Wesley O. Johnson , "Bayesian Thinking in Biostatistics", Chapter 9 (pages 241-264), 2021.
5. M. Frigeri, A. Guglielmi, "Spatio-temporal models for particulate matter in the Po valley", 2022.