Bayesian Methods - Ex4 - Exam project - Poisson Mixture Model

Jan Vávra

This assignment is can be solved via JAGS and `library(runjags)`. List through the manual to find what you need. However, solutions with original manual Gibbs sampling implementation are welcomed.

Your solution to this problem will be discussed during exam. Send it to vavraj@karlin.mff.cuni.cz and komarek@karlin.mff.cuni.cz latest at 7:00 on day D-2, where D is your exam term.

Data and model description

We will work with the `velibCount` data from `library(MBCbook)`, see `help(velibCount)` for more details. The `velibCount` object is a `list` of the following structure:

```
library(MBCbook)
data(velibCount)
names(velibCount)
```

```
## [1] "data"     "position" "dates"    "names"
```

It consists of 1213 city-bike rental stations in the vicinity of Paris. Each hour of one week in September 2014 the number of bikes stationed at each place has been recorded.

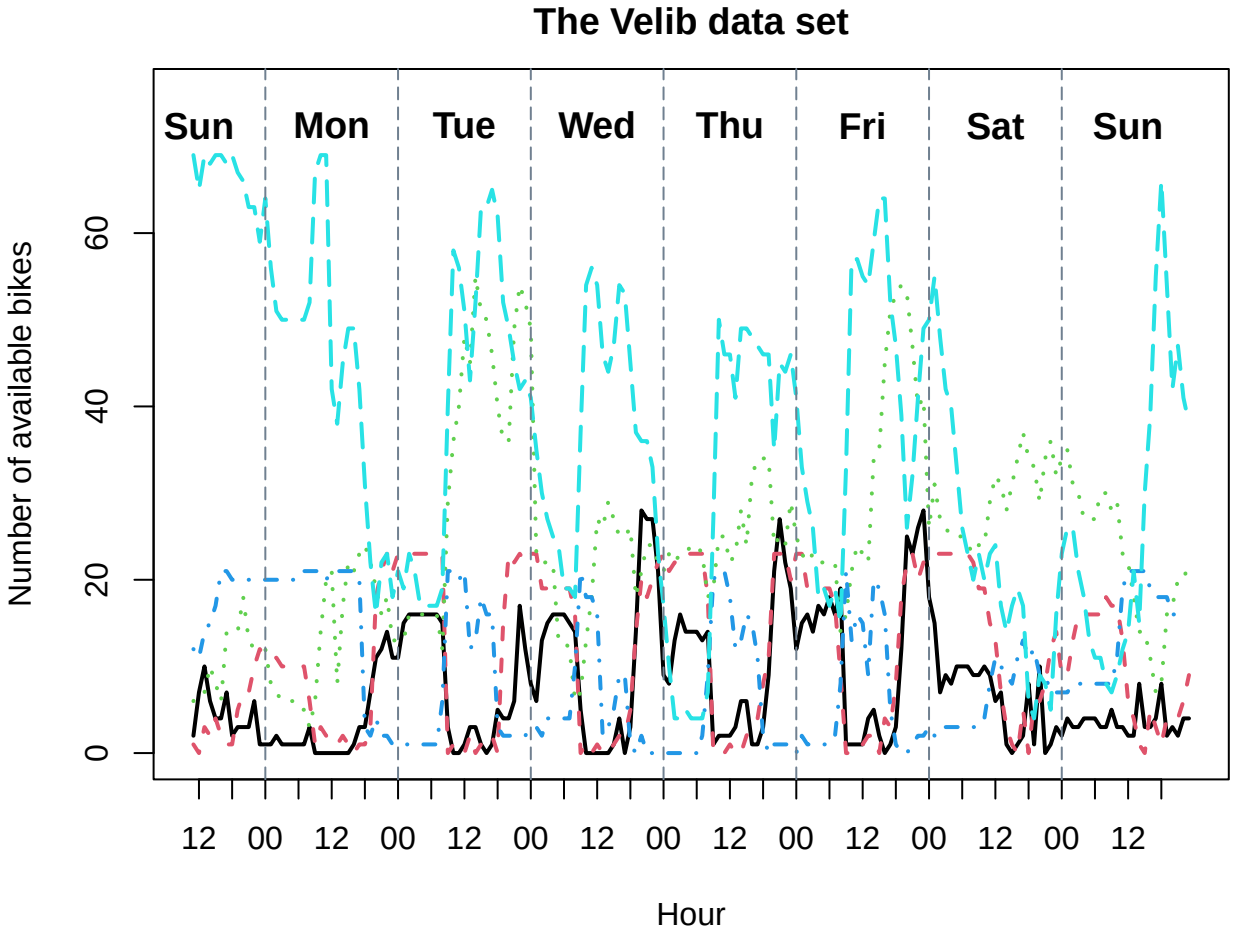
- `data` - matrix containing the bike counts stationed within specific rental station (row) at a specific time (column),
- `position` - matrix containing the coordinates of the rental stations,
- `dates` - vector of dates (in seconds from `"1970-01-01"`) at which `data` were counted,
- `names` - vector of names of the rental bike stations.

Your task is to discover different evolution curves of the intensity for the stationed bikes during a working day and cluster the stations based on similar profiles.

Let us first have a look on several stations:

```
X = velibCount$data
Sys.setlocale("LC_TIME", "en_US.UTF-8")
dates = strftime(as.POSIXct(velibCount$dates, origin = "1970-01-01"), format = "%a-%I%P")
days = strftime(as.POSIXct(velibCount$dates, origin = "1970-01-01"), format = "%u")
hours = strftime(as.POSIXct(velibCount$dates, origin = "1970-01-01"), format = "%H")

par(mar = c(4,4,2.5,0.5))
matplot(t(X[1:5,]), type = "l", lwd = 2,
        main = "The Velib data set", xaxt = "n", xlab = "Hour",
        ylab="Number of available bikes", ylim = c(0, 1.1*max(X[1:5,])))
axis(1, at = seq(2, length(hours), by = 6), labels = hours[seq(2, length(hours), by = 6)])
abline(v = seq(14, length(hours), by = 24), col = "slategrey", lty = 5)
text(seq(2, length(hours), by = 24), 1.05*max(X[1:5,]),
     # labels = c("Sunday", "Monday", "Tuesday", "Wednesday",
     #            "Thursday", "Friday", "Saturday", "Sunday"),
     labels = c("Sun", "Mon", "Tue", "Wed", "Thu", "Fri", "Sat", "Sun"),
     font = 2, cex = 1.2)
```

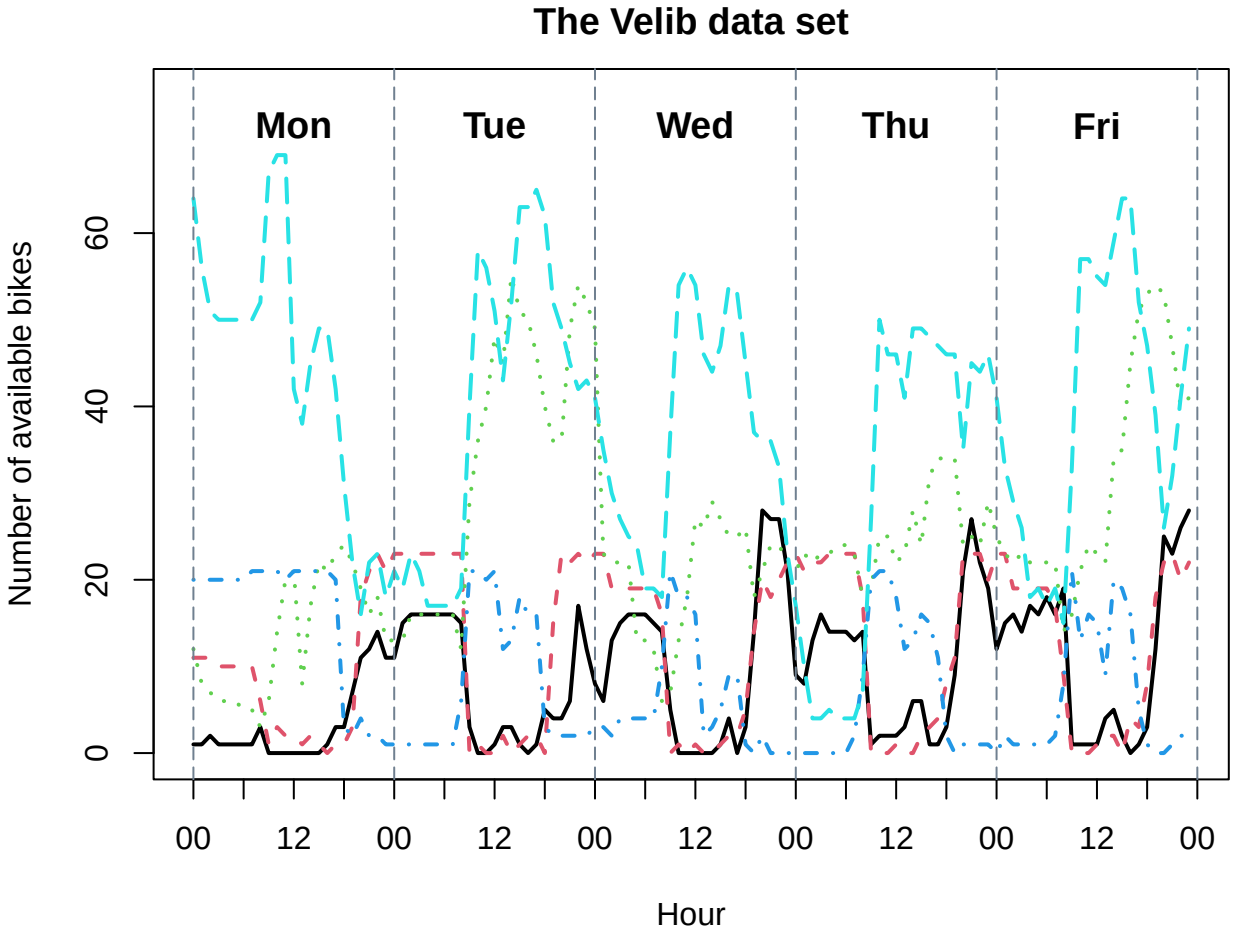



```
# legend(24*3+14, max(X[1:5,]), legend = velibCount$names[1:5],
#        col = 1:5, lty = 1:5, lwd = 2, xjust = 0.5, yjust = 1)
```

Some stations are filled during night, others are filled during the day. These patterns can possibly vary in many ways. The profiles within working days look very similar, but weekends tend to disobey the regular trends. Therefore, we eliminate observations from Saturday and Sunday from the analysis. Thus, only $24 \cdot 5 = 120$ observations should remain:

```
Y = X[, days %in% 1:5]
dim(Y)
hours = hours[days %in% 1:5]

par(mar = c(4,4,2.5,0.5))
matplot(t(Y[1:5,]), type = "l", lwd = 2,
        main = "The Velib data set", xaxt = "n", xlab = "Hour",
        ylab="Number of available bikes", ylim = c(0, 1.1*max(Y[1:5,])))
axis(1, at = seq(1, length(hours)+1, by = 6),
     labels = c(hours[seq(1, length(hours), by = 6)], "00"))
abline(v = seq(1, length(hours)+1, by = 24), col = "slategrey", lty = 5)
text(seq(13, length(hours), by = 24), 1.05*max(Y[1:5,]),
     # labels = c("Monday", "Tuesday", "Wednesday", "Thursday", "Friday"),
     labels = c("Mon", "Tue", "Wed", "Thu", "Fri"),
     font = 2, cex = 1.2)
```




```
# legend(24*2+13, max(Y[1:5,]), legend = velibCount$names[1:5],
#        col = 1:5, lty = 1:5, lwd = 2, xjust = 0.5, yjust = 1)
```

We will assume that the evolution pattern (from 00:00 to 23:00) in a working day is the same regardless of the day of the week. In particular, the expected curves for Monday, ..., Friday are the same.

Moreover, we will expect G different types of stations. Each type of station will be distinguished from the others by specific evolution during a working day. These curves and station types are **not known** in advance. We will perform *unsupervised clustering* based on mixture of Poisson distributions.

Poisson Mixture Model

Let $U_i \in \{1, \ldots, G\}$ be the unknown cluster label for station $i \in \{1, \ldots, n\}$. This label can be equivalently represented with vector of indicators $\mathbf{Z_i} = \left(\mathbb{I}_{(U_i=1)}, \ldots, \mathbb{I}_{(U_i=G)}\right)^\top$, which have number 1 at position g when $U_i = g$ and zeros elsewhere. We will specify the distribution of observed counts through conditional distribution given cluster label.

We will assume that bike storage counts follow Poisson distribution independently of each other. In mathematical notation $Y_{ihd} \,|\, U_i = g \sim \mathsf{Pois}\,(\mu_{ihdg})$, where Y_{ihd} is the number of bikes stored in station i at $h-1$ hours on day d and μ_{ihdg} is its expectation parameter. Naturally, $h \in \{1, \ldots, 24\}$, $d \in \{1, 2, 3, 4, 5\}$ and $g \in \{1, \ldots, G\}$.

Assuming complete generality of $\boldsymbol{\mu}$ would require $120nG$ unknown parameters. We will decompose $\boldsymbol{\mu}$ to decrease the number of parameters, but still capture the essentials. Namely, $\mu_{ihdg} = w_i \cdot \lambda_{hg}$, where

- λ_{hg} are elements of $24 \times G$ matrix, where each column describes the evolution of intensity proportions during a working day in cluster g,
- w_i scales the intensity for each bike station.

For identifiability purposes we have to enforce condition

$$\sum_{h=1}^{24} \lambda_{hg} = 1, \quad \text{for each } g \in \{1, \ldots, G\}. \tag{1}$$

otherwise we could obtain arbitrary rescaling, e.g. $\mu_{ihdg} = (w_i \cdot c) \cdot (c^{-1} \cdot \lambda_{hg})$. As discussed above, all days are assumed to follow the same evolution curve, hence, none parameter depends on d.

Assume multinomial distribution for $\mathbf{Z}_i \sim \mathsf{Mult}_G(1, \boldsymbol{\tau})$, which is equivalent to discrete categorical distribution $\mathsf{P}(U_i = g) = \tau_g$, where parameter $\boldsymbol{\tau} \in (0,1)^G$ is vector of G cluster probabilities, that sum up to 1, $\sum_{g=1}^{G} \tau_g = 1$.

Posit the following independent prior distributions for model parameters:

- $w_i \sim \Gamma(a, b)$,
- $\lambda_{hg} \sim \Gamma(c, d)$,
- $\boldsymbol{\tau} \sim \mathsf{Dir}(\alpha(1, 1, \ldots, 1)^\top)$,

where hyperparameters a, b, c, d and α are chosen to induce weakly informative prior. Random vector $\boldsymbol{\tau}$ follows Dirichlet distribution with parameter $\boldsymbol{\alpha}$ on a simplex (constraints $0 < \tau_g < 1$ and $\tau_1 + \cdots + \tau_G = 1$) if $p(\boldsymbol{\tau}|\boldsymbol{\alpha}) \propto \prod_{g=1}^{G} \tau_g^{\alpha_g - 1}$. Search for `ddirch` in JAGS manual.

Task 1 - Full-conditional distributions and Gibbs sampling

Describe in detail how the data augmentation works in this model. What would be the observed likelihood in standard frequentistic approach?

Write down the full-conditional probability density functions. Do they belong to well-known parametric families? If so, write down the values of the parameters. Comment on how the choice of hyperparameters for the prior distribution effects these full-conditional distributions. Choose their values appropriately.

Write down the algorithm of Gibbs sampling for this specific model.

Task 2 - JAGS implementation

Implement the Poisson Mixture Model in JAGS for a given number of clusters G. Show the code and explain the format of inputs. Enforce condition (1) above by distinguishing the unrestricted and restricted parameters λ_{hg}.

Would you be able to implement the Gibbs sampling on your own without the use of JAGS?

Task 3 - Running JAGS

Choose some reasonable value of number of clusters G ($G = 10$ used in source literature, but $G = 2$ or $G = 3$ could be enough). Sample two Markov chains using JAGS (or your own implementation) to approximate the posterior distribution $p(\mathbf{U}, \boldsymbol{\tau}, \mathbf{w}, \boldsymbol{\lambda} \,|\, \mathsf{data})$. Initialize them from completely different points. Choose appropriate `burnin` by monitoring some of the trajectories. Do the chains converge to the same solution? What could be the danger here when approximating posterior distribution from both chains?

Hints: The parametric space is huge. Sampling can take even hours! Start by low number of samples, estimate how long it would take you to produce MCMC of reasonable length (`...$timetaken`). Continue in sampling with function `extend.jags`, which will skip the model compilation. You are allowed to do any reasonable shortcuts to speed up your sampling: parallelization, low G, single day of the week only, subset of stations, low sample length. Defend your shortcuts. Implementing it on your own may severely reduce the computation time.

Task 4 - Posterior approximation via MCMC

Decide whether you can use samples from all chains to approximate the posterior. Provide summaries including ET and HPD intervals for a representative subset of primary model parameters.

Task 5 - Plotting different evolution profiles during the day

Estimate the posterior mean for parameters λ_{hg} and plot the profiles for each g into one plot. Depict also the cluster size. Interpret the discovered clusters.

Estimate the posterior median for station-specific intensity w_i and plot histogram.

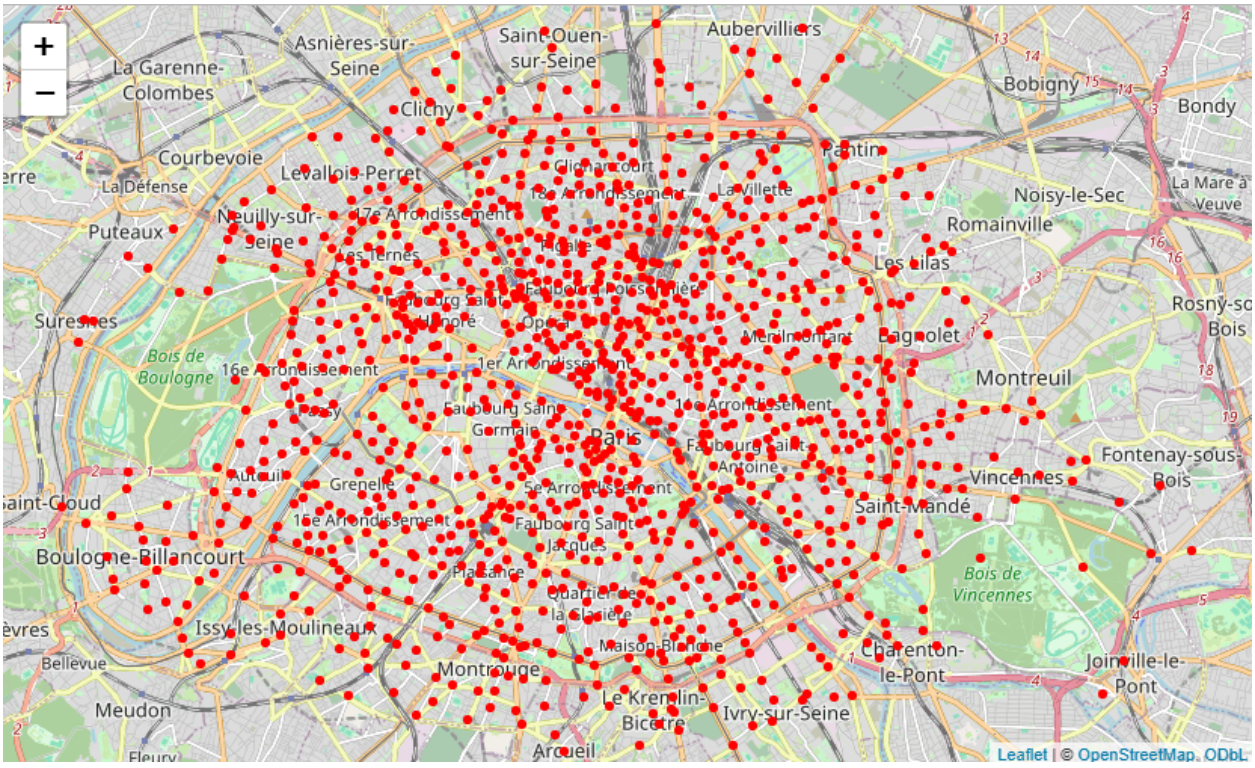
Task 6 - Clustering the stations

Using the sampled cluster labels U_i assign each station to the cluster with the highest frequency. You may apply a custom rule to keep a station unclustered if there are two or more closely competing clusters. Take the locations of the stations and plot them into a real world map. Distinguish clusters by different colors. Do you see some logical pattern?

Hint: How to plot a real world map using R? Use either `mapview` *or* `leaflet` *library. ‘*

```
library(leaflet)
library(htmlwidgets)
library(webshot2)

Paris <- leaflet(velibCount$position) %>%
  addTiles() %>%
  addCircleMarkers(color = "red", radius = 3, fillOpacity = 1, stroke = FALSE) %>%
  fitBounds(lng1 = min(velibCount$position$longitude),
            lat1 = min(velibCount$position$latitude),
            lng2 = max(velibCount$position$longitude),
            lat2 = max(velibCount$position$latitude))
saveWidget(Paris, file.path(FIG, "Paris_velib_stations_unclustered.html"),
           selfcontained = TRUE)
webshot(file.path(FIG, "Paris_velib_stations_unclustered.html"),
        file = file.path(FIG, "Paris_velib_stations_unclustered.png"),
        vwidth = 500, vheight = 300)
```



Task 7 - Clustering a newly observed station

To which cluster would you assign a newly observed station $\mathbb{Y}_{\text{new}}$ if

- $Y_{\text{new}\,dh} = 0, \qquad \forall d = 1, \ldots, 5, \forall h = 1, \ldots, 24,$
- $Y_{\text{new}\,dh} = 3, \qquad \forall d = 1, \ldots, 5, \forall h = 1, \ldots, 24,$
- $Y_{\text{new}\,dh} = 3\mathbb{I}_{(6<h\leq 18)}, \qquad \forall d = 1, \ldots, 5, \forall h = 1, \ldots, 24,$
- $Y_{\text{new}\,dh} = 3\mathbb{I}_{(1\leq h\leq 6)|(18<h\leq 24)}, \qquad \forall d = 1, \ldots, 5, \forall h = 1, \ldots, 24?$

Start by writing down the general formula for posterior probability $\mathsf{P}(U_{\text{new}} = g|\mathbb{Y}_{\text{new}})$ and express it as a function of primary model parameters. Explore the posterior distribution of these probabilities and classify new stations based on a reasonable rule.

Prepare for discussion

Here are some points to be discussed:

- How would you use EM algorithm to estimate such model. What are the similarities and differences between EM approach and Gibbs sampling?
- In what ways could the model be improved with the current data? Critically judge all the assumptions made along the way.
- Could also the coordinates participate in clustering?
- Could you reduce the number of parameters needed to capture the evolution over day?
- Imagine you would have data for more days (even more than a year) and abundance of computational power, how would you extend the model to capture weekends or seasonality?
- Could the information about capacity of the stations be any useful?
- How would you choose the number of clusters more rigorously?