MSc in Mathematical Engineering - Statistical Learning
Project of the course of Bayesian Statistics

# Hierarchical Bayesian Nonparametric model to smooth functional data

*Authors:*
Mirko Giovio
Riccardo Scaramuzza
Daniele Venturini
Paolo Vergottini

*Advisor:*
Prof. Raffaele Argiento

*Course Lecturers:*
Prof. Guglielmi Alessandra
Dr. Corradin Riccardo
Dr. Mario Beraha

POLITECNICO
MILANO 1863

Milan - 18 February 2021

# Contents

# 1 Introduction

This is an attempt to sum up and show all the work we carried out under the guidance of Prof. Argiento. Such work finds his framework as part of a wider study, which attempts to provide a smoothing model for clustered functional data, and its main focus in Hierarchical Dirichlet Process Mixture models and the development of tools to sample from their posterior. In this first introductory section we try to provide both a brief recap of the context in which our work takes place, and also a synthetic theoretical background about Hierarchical Dirichlet Process models.

## 1.1 Context

The whole work find is context in the field of Sport Analytics, in particular in describing and predicting the evolution of performances over time for shotput athletes.

$$y_{ij} = g_i(t_{ij}) + \varepsilon_{ij} \tag{1}$$

In (1), observed performance j of the individual i is expressed as athlete-specific function g plus a heteroskedastic noise term. It is important to note that each individual performance measurements are grouped according to season (year) of career, and such grouping is believed to provide important information to be coded by the model. The resulting idea is the split of the smoothing function $g_i$ into two different components: a smooth function $f_i$, which captures intra-seasonal variability, and a step function $\mu_i$, which basically specifies a level around which the athlete perform in each season [see Figure 1].

$$g_i = f_i + \mu_i \tag{2}$$

The focus of our work is the second component $\mu_i$. In order to capture the "seasonality effect", it is assumed constant along each period. Therefore for each athlete it can be modeled as:

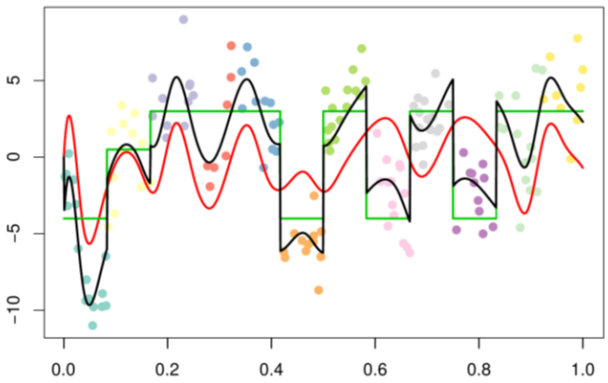



Figure 1: Smooth and step components

$$\mu_i(t) = \sum_{s=1}^{S} \theta_{is} 1_{\omega_s}(t) \tag{3}$$

The values $\vartheta_{is}$ for all t belonging to the specific season ( i.e. $\omega_s$) can be seen as an athlete-season random effect. A very straightforward interpretation for the $\vartheta$s factors is to see them as a "season-specific intercept" of the athlete performances distribution during a season. At this point it is crucial to highlight that such $\vartheta$s are not independent between each others, nor it is in our interest to treat them as if they were: it is reasonable to expect some degree of dependence for the average performance across seasons, namely we need a model to allow borrowing of information between athletes. Moreover, also a sharing of information between group of athletes in each season can be supposed to be useful. To clarify, imagine to fix a season: it is possible construct a clustering among athletes on their results, to identify a partition on levels of performances; that can be done for each season and then it becomes possible to link clusters along different seasons, including also this information in the seasonal intercepts definition. With this procedure we

can try to catch a similarity in the performance of athlete i, in season j, with the performance of another athlete i' during another season j', as if they came from the same underlying distribution which is represented by precisely those theta factors we are looking for. All of these considerations lead to the choice of Hierarchical Dirichlet Process Mixture Models.

## 1.2 Theoretical background

In this section we try to furnish necessary information about the Hierarchical Dirichlet Process and the Chinese Restaurant Franchise representation, which is fundamental as it provide the scheme on which our sampler is built upon. The notation is heavy enough, but we try to keep it as clear as possible.

Assuming the existence of capital S seasons in the observed data, each datum $y_{is}$ is sampled from a mixture distribution F, with density f, conditionally on a vector parameters $\vartheta_{is}$ (the mixture component). These mixture components are sampled from a season specific distribution Ps, generated accordingly to a Dirichlet Process, with baseline distribution P. The distribution P is itself generated by a DP with a base distribution P0.

$$Y_{1s}, .., Y_{n_s s} | \theta_{1s} ..., \theta_{n_s s} \overset{\text{i.i.d.}}{\sim} f(y_{ij} | \theta_{ij}) \quad (4)$$

$$\theta_{1s}, ..., \theta_{n_s s} | P_s \sim DP(\alpha, P) \quad (5)$$

$$P \sim DP(\gamma, P_0) \quad (6)$$

Therefore, fixing a season (i.e. fixing Ps), the data distribution is the mixture described by (4) and (5), and the desired borrowing of information is obtained thanks to (6), which defines a common hyperprior for all seasons. Indeed each season specific distribution Ps, s =1..S, is "sampled" from a Dirichlet Process with the same baseline (P0) and this allows the borrowing of information. So with the latter model we let the random effect of athlete i at season s, $\vartheta_{is}$,to be shared with the random effect of athlete i' at season s' , $\vartheta_{i's'}$.

The distribution P varies around the prior P0 with the amount of variability governed by the parameter $\gamma$ while the actual distribution Ps over the factors in the s-th group deviates from P , with the amount of variability governed by the parameter $\alpha$.

### 1.2.1 Chinese Restaurant Franchise

Since the final goal of our work is to obtain an estimate of $\vartheta_{is}$ , we wish to build an efficient and effective MCMC algorithm (in particular a Gibbs sampler). However our model does not have a parametric prior; thus, we have to deal with an infinite dimensional space. To solve this issue we relied on a particular representation of Hierarchical Dirichlet Mixture Model, known as Chinese restaurant franchise. This is an extension of the classical Chinese Restaurant representation, where multiple restaurants are allowed to share a set of dishes.

The idea is that in this extended version we have a restaurant franchise with a shared menu across all the restaurants. At each table of each restaurant, only one dish is ordered from the common menu when a first customer sits there, and it is shared among all of the customers who sit at the same table. Notice how multiple tables in multiple restaurants can serve the same dish.

We have to introduce now the notation we are going to use from now on: restaurants represent seasons and the customers correspond to the hidden parameters $\vartheta_{ji}$. We

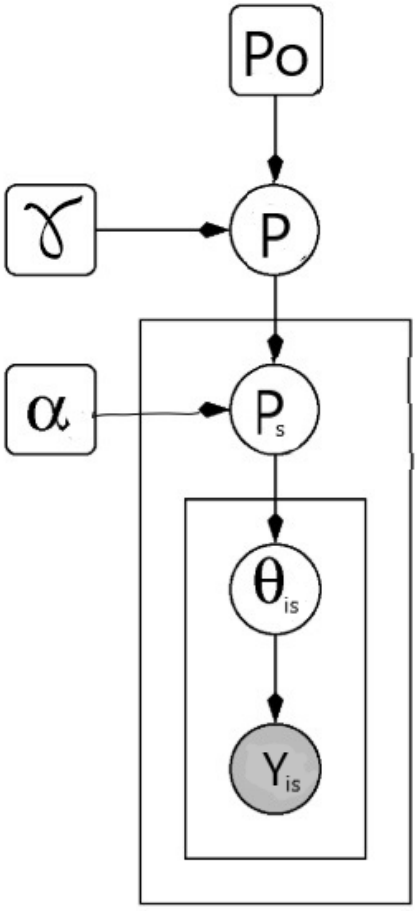



Figure 2: Graphical Model Representation of a Hierarchical DP Mixture Model . In the graphical model formalism, each node in the graph is associated with a random variable, where shading denotes an observed variable.

call $\Phi_1$, . . . , $\Phi_K$ the K dishes which compose the global menu and these values, iid random variables distributed according to P0, are the unique discrete values from the DP. We introduce variables, $\Psi_{jt}$ ,that represent dish served in restaurant j at table t. Note that each $\vartheta_{ji}$ is associated with one $\Psi_{jt}$ , whereas each $\Psi_{jt}$ is associated with one $\Phi_k$ We introduce indicators to point out these links. In particular, let $t_{ji}$ be the index of the $\Psi_{jt}$ associated with $\vartheta_{ji}$ , and let $k_{jt}$ be the index of $\Phi_k$ssociated with $\Psi_{jt}$ . In the Chinese restaurant franchise metaphor, customer i in restaurant j sits at table $t_{ji}$ whereas table t in restaurant j serves dish $k_{jt}$. These indexes play a crucial role in the coding phase, allowing to keep track of the relations between customers, tables and dishes.

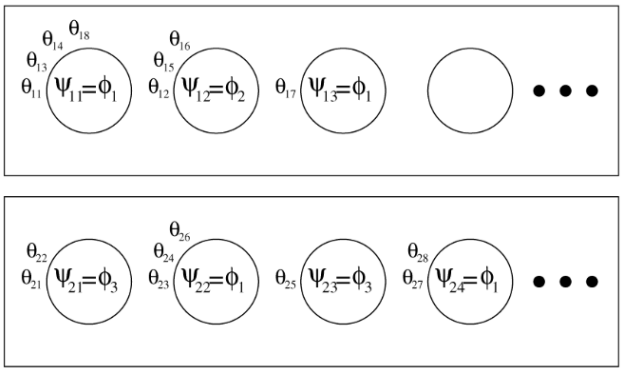



Figure 3: A Representation of a Chinese Restaurant Franchise. Each restaurant is represented by a rectangle. Customers ($\vartheta_{ji}$s) are seated at tables (circles) in the restaurants. At each table a dish is served. The dish is served from a global menu ($\Phi_k$), whereas the parameter $\Psi_{jt}$ is a table-specific indicator that serves to index items on the global menu. The customer $\vartheta_{ji}$ sits at the table to which it has been assigned

We need a notation to count customers and tables. We denote the number of customers in restaurant j at table t eating dish k with $n_{jtk}$. Then $n_{jt}$ is the number of customers in restaurant j at table t, and $n_{jk}$ the number of customers in restaurant j eating dish k. The notation $m_{jk}$ denotes the number of tables in restaurant j serving dish k. Thus $m_j$ represents the number of tables in restaurant j, $m_k$ the number of tables serving dish k, and finally m is the total number of tables with at least one customer.

To build the Gibbs sampler, we now compute the marginals of $\vartheta_{ji}$ marginalizing with respect to P0 and Ps and relying onto the

known formula of the Polya Urn Scheme.[1]

$$\theta_{ji}|\theta_{j1},...,\theta_{ji-1},P,\alpha \sim \sum_{t=1}^{m_j} \frac{n_{jt}}{i-1+\alpha}\delta_{\psi_{jt}} + \frac{\alpha}{i-1+\alpha}P \quad (8)$$

This is an analogue result. Indeed a draw from this mixture can be obtained by drawing from the distribution on the right with probabilities proportional to the cardinality of each of the group already present. If a term in the first summation is chosen, then we increment $n_{jt}$, set $\vartheta_{ji}$ $\Psi_{jt}$nd let $t_{ji}$ = t for the chosen t. If the second term is chosen, then we increment $m_j$ by one, sample the new value $\Psi_{jm_j}$rom P, and set $\vartheta_{ji}$ $\Psi_{jm_j}$nd $t_{ji} = m_j$. The following step is to integrate out P . Note that P is distributed according to a DP, so we can integrate it out by exploiting once again the Polya Urn law and write the conditional distribution of $\Psi_{jt}$s:

$$\psi_{jt}|\psi_{11},\psi_{12},...,\psi_{jt-1},P_0,\gamma \sim \sum_{k=1}^{K} \frac{m_k}{m_{..}+\gamma}\delta_{\phi_k} + \frac{\gamma}{m_{..}+\gamma}Po \quad (9)$$

If we draw $\Psi_{jt}$y choosing a term in the summation on the right side of this equation, then we set $\Psi_{jt}$ $\Phi_k$nd let $k_{jt}$ = k for the selected k. Otherwise if we choose the second term, we increment K, draw $\Phi_k$P0, and set $\Psi_{jt}$ $\Phi_k$nd $k_{jt}$ = K. We can proceed as follows, using these equations, to obtain samples of $\vartheta_{ji}$ For each j and i, we sample $\vartheta_{ji}$sing (8) and if a new sample from P is needed, then we use (9) to obtain a new draw of $\Psi_{jt}$.

[1]The Pólya urn scheme:

$$\theta_{ji}|\theta_{j1},...,\theta_{ji-1},P,\alpha \sim \sum_{t=1}^{i-1} \frac{1}{i-1+\alpha}\delta_{\theta_t} + \frac{\alpha}{i-1+\alpha}P \quad (7)$$

# 2 MCMC algorithm

Our first assumption on the data distribution F is the conjugacy with baseline P0; this simplifies the mathematical discussion and allows us to focus on the issues specific to the hierarchical DP. Moreover, we assume fixed the concentration parameters $\gamma$ and $\alpha$. With a brief recap, we first recall the variables of interest. The $y_{ji}$s are the observed data. Each $y_{ji}$ is a draw from the absolutely continuous distribution F($\vartheta_{ji}$, with density f. The values $\vartheta_{ji}$re associated with the table $t_{ji}$ in the Chinese restaurant representation. The random variables $\Psi_{jt}$ssume values $\Phi_{k_{jt}}$ P0 is the prior over the parameters $\Phi_k$ Let $z_{ji}$ the index denoting the mixture component associated with the observation $y_{ji}$ ( i.e. $z_{ji}$ = $k_{jt_{ji}}$ ) . We use the notation $n_{jt}$ for the number of customers at table t in restaurant j, $m_{jk}$ to indicate the number of tables in restaurant j serving dish k, and K to denote the total number of dishes being served in all restaurants. Notation for the marginal counts follows straightforward. When we want to indicate that we are removing a variable we attach a superscript to a vector of variables or a count ( e.g. $x^{-ji}$, $k^{-jt}$ ). We introduce the notation **y** to indicate a vector of data. Let P0 have density p0(·). Since P0 is conjugate to F, we integrate out the mixture component parameters in the sampling schemes. Denote the conditional density of $y_{ji}$ under the mixture component k given all the other data except $y_{ji}$ as the following ratio between marginals:

$$f_k^{-y_{ji}} = \frac{\int f(y_{ji}|\phi_k)\prod_{j'i'\neq ji, z_{j'i'}=k} f(y_{j'i'}|\phi_k)p_0(\phi_k)d\phi_k}{\int \prod_{j'i'\neq ji, z_{j'i'}=k} f(y_{j'i'}|\phi_k)p_0(\phi_k)d\phi_k} \quad (10)$$

Similarly let $f_k^{-\mathbf{y}_{jt}}$ denote the conditional density of the vector $\mathbf{y}_{jt}$ given all data items associated with mixture component k, but $\mathbf{y}_{jt}$ itself.

## 2.1 Sampling from the posterior

The aim now is to use Chinese restaurant franchise representation to sample from the prior distribution over the $\vartheta_{ji}$ This framework can be seen as building a Gibbs sampling scheme and sampling from the posterior given observations **y**. The basic idea is that instead of dealing directly with the $\vartheta_{ji}$ and $\Psi_{jt}$ , we can sample their indexes variables $t_{ji}$ and $k_{jt}$ . The values we are looking for can be reconstructed from the sampled indexes and the $\Phi_k$. This representation makes the MCMC sampling scheme more efficient an simple to understand. Moreover the $t_{ji}$ and the $k_{jt}$ inherit the exchangeability properties; the distributions in (8) and (9) can be adapted in terms of $t_{ji}$ and $k_{jt}$. In the metaphor of the Chinese restaurant franchise, we can describe intuitively how the algorithm is build and, in particular, how the trajectory is updated at each step. Suppose we have already allocated all customers in every restaurant; first we perform the change-table operation: one at a time each customer gets up from the table at which he was sitting and must decide whether to sit down, go to another existing table or sit at a new table properly allocated. In the last case he must also order a new dish. After repeating this operation for each customer, we perform a second level operation, the change-dish: at each table (the group of person sitting at the table is considered as a whole) in each restaurant it is asked if they would like to change the dish that is served and they must decide whether to keep the old one or to change with a new one ( already served in other tables or totally new). At the end of each iteration the partition of customers is used to sample from the full conditionals the values of interest. Let's see more in depth how this procedure works

## 2.2 Sampling t

To compute the conditional distribution of $t_{ji}$ given the rest, we can consider $t_{ji}$ as the last variable being sampled among all groups in (8) and (9), relying on the exchangeability of these variables. The probability of reassign a table already occupied by other customers is proportional to $n_{jt}^{-ji}$ times the conditional likelihood of datum $y_{ji}$ (i.e. $f_k^{-y_{jt}}(y_{jt})$ ) , while the probability of assign a new table is proportional to $\alpha$ In this case the likelihood can be computed integrating out with respect to the possible values $k_{jtnew}$ and we denote it as $f_{tnew}^{-y_{jt}}(y_{jt})$.

$$p(y_{ji}|t^{-ji}, t_{ji} = t^{new}, k) = \sum_{k=1}^{K} \frac{m_{.k}}{m_{..} + \gamma} f_k^{-y_{ji}}(y_{ji}) + \frac{\gamma}{m_{..} + \gamma} f_{k^{new}}^{-y_{ji}} \quad (11)$$

where $f_{knew}^{-y_{ji}}(y_{ji})$ is nothing but the marginal of $y_{ji}$. Thus the conditional law of $t_{ji}$ is:

$$p(t_{ji} = t|t^{(} - ji), k) \propto \begin{cases} n_{jt.}^{-ji} f_{k_{jt}}^{-y_{ji}}(y_{ji}) & \text{if t previously used} \\ \alpha p(y_{ji}|t^{-ji}, t_{ji} = t^{new}, k) & \text{if } t = t^{new} \end{cases} \quad (12)$$

and if a new value for $t_{ji}$ is sampled then we have to assign also a new dish for the new table:

$$p(k_{j_t^{new}} = k|t, k^{(} - j_t^{new})) \propto \begin{cases} m_{.k} f_k^{-y_{ji}}(y_{ji}) & \text{if k previously used} \\ \gamma f_{k^{new}}^{-y_{ji}}(y_{ji}) & \text{if } k = k^{new} \end{cases} \quad (13)$$

## 2.3 Sampling k

Sampling $k_{jt}$ means performing a change table for all the costumers sitting at table t in restaurant j. Therefore the probability of selecting a dish already served is proportional

to $m_k^{-jt}$ times $f_k^{-y_{jt}}(\mathbf{y_{jt}})$, while the probability of assigning a totally new dish is proportional to $\gamma$imes the marginal of the data sitting a the selected table $f_{knew}^{-y_{ji}}(\mathbf{y_{ji}})$.

$$p(k_{j_t} = k|t, k^{(} - jt)) \propto \begin{cases} m_{.k} f_k^{-y_{jt}}(y_{jt}) & \text{if k previously used} \\ \gamma f_{k^{new}}^{-y_{jt}}(y_{jt}) & \text{if } k = k^{new} \end{cases} \quad (14)$$

## 2.4 Prediction

The predictive density of a new customer $y_{N+1}$ can be simply obtained integrating out with respect to the posterior distribution of the other $\vartheta$. To simplify a bit the notation we call $P_{jh}^{(told)}$ the probability of assigning a table h which already exists , and $P_j^{(tnew)}$ the probability of assigning a new table. In this last case we denote as $P_m^{(dold)}$ the probability of selecting a dish already served in other tables and as $P^{(dnew)}$ the probability of ordering a new dish.

Therefore we can find the predictive of the new datum both in the case of an existing restaurant and in the case of a new restaurant:

$$\begin{aligned} p(y_{j(n_j+1)}|Y, t, k, \Phi) = \\ \sum_{h=1}^{K_j} P_{jh}^{told} f(y_{j(nj+1)}|\theta_{jh}) + \\ P_j^{tnew} \sum_{m=1}^{M} P_m^{dold} f_(y_{j(nj+1)}|\Phi_m) \\ + P_j^{tnew} P^{dnew} \int_\Theta f(y_{j(n_j+1}|\Phi) P_0(d\Phi) \end{aligned} \quad (15)$$

$$\begin{aligned} p(y_{(d+1)1}|Y, t, k, \Phi) = \\ \sum_{m=1}^{M} P_m^{dold} f_(y_{(d+1)1}|\Phi_m) \\ + P^{dnew} \int_\Theta f(y_{(d+1)1}|\Phi) P_0(d\Phi) \end{aligned} \quad (16)$$

In practice we evaluate these densities via MCMC integration.

# 3 Sampler Implementation

In this section are presented all the versions of the sampler we implemented, to fully describe all the steps in our work. Moreover, a slight introduction about the backbone of the implementations is described: actually each version can be seen as a generalization with respect to the first one. For each of them we try to follow a consistent scheme, specifying firstly the exact model shape and priors, then the goal of the implementation, and finally the result of some simulations.

## 3.1 Backbone structure

First thing to note is that the whole sampler structure is implemented in C++, due to his extremely high computational cost; instead, simulated data generation and result analysis is carried out through R.
We try to keep the sampler implementation friendly for an external user as much as possible: a specific class called ”Sampler” permits to receive input data, to initialize data structures and to tweak each parameter intuitively.
The backbone of the sampler is constituted by three sequential containers: the first one contains data provided by the user; the second one contains table indexes per each customer (i.e. each datum), while the third contains dish indexes per each customer. From these structures it is possible also to obtain the clustering at each MCMC step.
Other important quantities, necessary for marginals computation, are total number of allocated tables, number of tables per restaurant, number of tables which serve each dish, number of customers per ta-

ble, number of customer eating each dish. Please note that all of these quantities are computed automatically after the sampler initialization, and are updated according to the specific algorithm. Algorithm initialization is performed using a K-means run on each restaurant, to give a credible starting point (number of initial clusters per restaurant can be set by the user).
Briefly, at each iteration the sampler firstly updates the table allocation per each datum, then updates the dish allocation per each table, using weights computed as specified in the previous sections; this sequential approach relies on the assumption of exchangeability. Afterwards, all quantities of interest (mixture components,punctual density estimates, etc) are sampled according to the actual allocation of tables and dishes from the model full conditionals and then saved. As it can be imagined, the whole procedure is heavily time consuming.

## 3.2 First version: N-N sampler

This first version is developed according to a naïve model, which we try to keep as simple as possible:

$$Y_{1s},..,Y_{n_ss}|(\mu_{1s},...,\mu_{n_ss},\sigma_0^2) \overset{\text{i.i.d.}}{\sim} N(y_{is}|\mu_{is},\sigma_0^2) \tag{17}$$

$$\mu_{1s},...,\mu_{n_ss}|P_s \sim DP(\alpha,P) \tag{18}$$

$$P \sim DP(\gamma,P_0) \tag{19}$$

$$P_0 \overset{d}{=} N(\mu_0,\tau^2) \tag{20}$$

The sampling model is constituted by a mixture of Gaussian distributions, with fixed (and known) variance. Parameters of interest are represented by the means, over which we put the hierarchical Dirichlet Process prior. As we already pointed out, the baseline measure P0 is chosen to be conjugated, and thus it is a Gaussian. Note that good hyperparameters starting point can be guessed fairly easily through data inspection.
The first version was developed to achieve two crucial goals: the key one is, of course, to verify the sampler functioning from a structural point of view; the second was to start investigating the effect of the hierarchical structure hyperparameters: $\alpha$ and $\gamma$. To do so, we generate a set of simulated data; also here, we decide to stick to a quite simple case. Three restaurants are taken into account and, in each of them, we specified the mixtures as follows:

$$\begin{aligned} Y_1,..,Y_{150} &\overset{\text{iid}}{\sim} \\ 1/3N(y_{i1}|-2,0.3) + 1/3N(y_{i1}|0,0.3) &\\ +1/3N(y_{i1}|2,0.3) & \end{aligned} \tag{21}$$

$$\begin{aligned} Y_1,..,Y_{100} &\overset{\text{iid}}{\sim} \\ 1/2N(y_{i1}|-2,0.3) + 1/2N(y_{i1}|0,0.3) & \end{aligned} \tag{22}$$

$$\begin{aligned} Y_1,..,Y_{75} &\overset{\text{iid}}{\sim} \\ 1/2N(y_{i1}|0,0.3) + 1/2N(y_{i1}|2,0.3) & \end{aligned} \tag{23}$$

Of course, due to the neat separation of data, if the sampler produces consistent result we can just assess its proper functioning.
We now focus on how the parameters influenced the borrowing of information across groups. The general behaviour is supposed to be as follows: increasing values of the first parameter $\alpha$ produces a greater probability of allocating new tables at each step; since the dish allocation for new tables is directly related to the actual disposition of

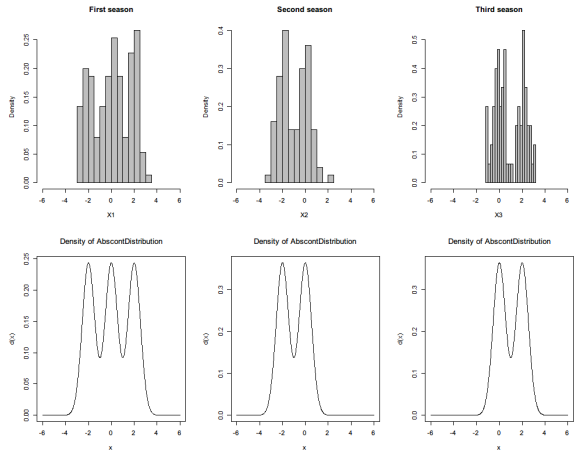



Figure 4: True data distribution in every restaurant

dishes in all restaurants, it consequently leads to an increase of the borrowing of information. Instead, $\gamma$ is connected to the probability of sampling from the baseline distribution, whenever a new dish need to be chosen, instead of sampling from the already existent ones; thus, its increase is supposed to lighten the borrowing. Here are reported the result of a small sensitivity analysis, performed on $\alpha$, to validate this belief. We can appreciate the results by looking at the predictive density of a new datum in each group (See Figures 5, 6 and 7) [2].

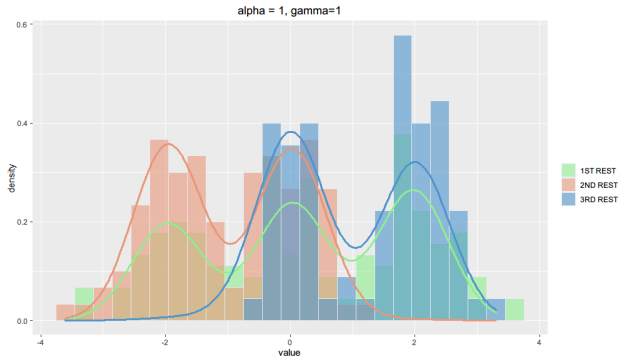



Figure 5: No sharing of information

Setting low values for the parameter $\alpha$, it is clear that the distribution of data is well captured but no information is shared among different seasons (restaurants).

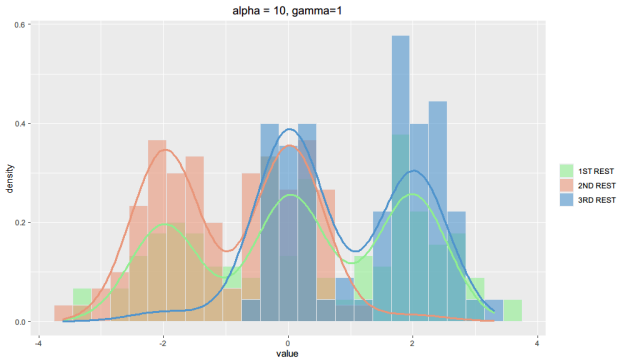



Figure 6: Increase of alpha lead to sharing of information

Increasing $\alpha$, we can appreciate the appearance of small peaks, in restaurants two and three, even if no corresponding data were provided in those seasons, testifying the borrowing from the other two restaurants.

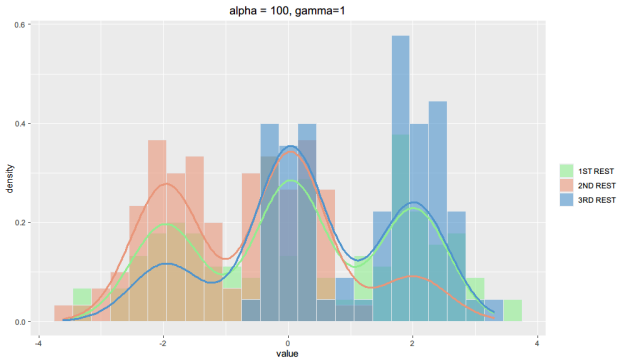



Figure 7: Very big values of alpha produce peaks

As expected, this effect is strongly amplified by the increasing of the parameter: we can actually produce big peaks by setting very high values.
As expected, it can be concluded that the tuning of these two hyperparameters covers a primary role in training such kind of models.

[2] Note that the exact same procedure can be repeated on $\gamma$.

## 3.3 Second Version: N-IG sampler

The first version is based on the too restrictive assumption of known variance: here we present a first attempt to deal with unknown variances of the mixture components. The new model basically considers a bivariate unknown parameter space, composed by the couple mean-variance, per each mixture components; also in this case we rely on conjugacy, and thus identify P0 as a Normal-Inverse Gamma distribution [3]:

$$Y_{is}|(\mu_{is},\sigma^2_{is}) \overset{\text{iid}}{\sim} N(y_{is}|\mu_{is},\sigma^2_{is}) \quad \forall i = 1,...,n_s \tag{24}$$

$$\mu_{is},\sigma^2_{is}|P_s \sim DP(\alpha,P) \quad \forall i = 1,...,n_s \tag{25}$$

$$P \sim DP(\gamma,P_0) \tag{26}$$

$$P_0 \overset{d}{=} NIG(m_0,k_0,a_0,b_0) \tag{27}$$

Also in that case the sampler is tested on simulated data. We should point out that, as expected due to the increasing computational demand, the time needed to perform each MCMC run increases. The ability of the sampler to correctly capture the underlying data distribution is confirmed.

Furthermore, we decided to test the capability to spot small subgroups in data, whose distribution could be masked by the rest of data [4]. Here it is reported the true distribution of the simulated data considered for such task: two seasons are considered, with small and concentrated subgroups and diffuse ones (see Figure 8):

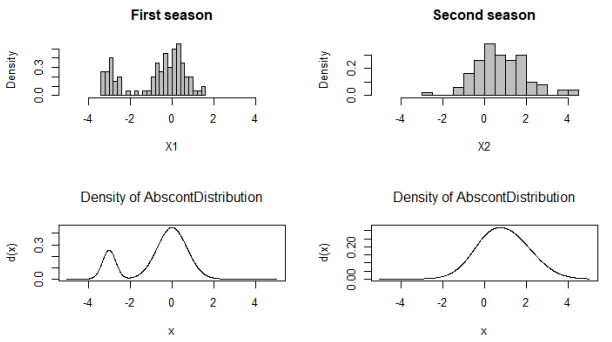



Figure 8: Simulated data distribution in both groups

$$Y_{1i} \overset{\text{iid}}{\sim} 0.2N(y_{1i}|-3,0.1) + 0.8N(y_{1i}|0,0.5) \quad \forall i = 1,...,100 \tag{28}$$

$$Y_{2i} \overset{\text{iid}}{\sim} 0.1N(y_{2i}|0,0.5) + 0.9N(y_{2i}|1,1.5) \quad \forall i = 1,...,100 \tag{29}$$

Note that the component shared by the two groups has a much higher weight in the first mixture and a very small one in the second: an independent model will likely not be able to adequately recover this component, while, through sharing of information gained from the first group, the hierarchical model is able to catch it (see Figure 9).

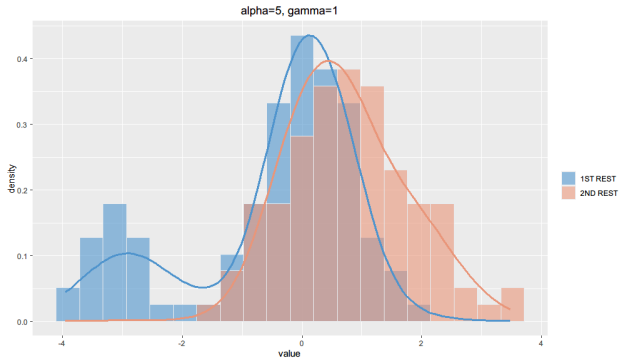



Figure 9: Predictive densities for a new datum in the two groups

[3] Starting points of the priors parameters are chosen following the same approach as [1]

[4] Such ability is relevant when treating real data

In addition, also the predictive density for an hypothetic new group is reported in Figure 10.


Figure 10: Predictive density for a new group

## 3.4 Final Version: Multiple Throws sampler

In this section is described the final version of the sampler,with which the analysis of the Shotput Dataset is performed. Once again, the sampler and the results analysis procedure are tested on a set of simulated data, which mimics the real ones.
The most challenging aspect is to account for the fact that each athlete can perform multiple throws per season: the sampling model must be tweaked, as essentially a multivariate sample of data (of unknown size) is drawn per each mixture component (note, however, that each athlete is still linked with a unique one). It is also of paramount importance to remark that the variance parameter is out of the mixture and it is provided with an Inverse Gamma prior: this allows us to find the optimal common estimate of the variance. Although assuming a unique value shared by all the components may seem a bit restrictive, this choice is made consciously, inspecting the behaviour of real data; it, also, leads to a consistent gain in terms of time required to run the sampler.

$$Y_{is} = (Y_{i1s}, .., Y_{in_{is}s}) \quad \forall i = 1, ..., 100 \tag{30}$$

$$Y_{\text{iks}}|(\theta_{is}, \sigma^2) \overset{\text{iid}}{\sim} N(y_{iks}|\theta_{is}, \sigma^2) \quad \forall k = 1, ..., n_{is} \forall i = 1, ..., 100 \tag{31}$$

$$\theta_{is}|P_s \sim DP(\alpha, P) \quad \forall i = 1, ..., 100 \tag{32}$$

$$P \sim DP(\gamma, P_0) \tag{33}$$

$$P_0 \overset{d}{=} N(\mu_0, \tau^2) \tag{34}$$

$$\sigma \overset{\text{iid}}{\sim} IG(a_0, b_0) \tag{35}$$

Clearly the data structure of the sampler has been modified accordingly, as well as functions computing marginals used to update table and dish allocation; computational cost increases even further.
As said, next step is to evaluate the performances of the model on a simulated dataset, built to resemble the real one. We considered five seasons and a number of athletes of one hundred [5]. Then, for each athlete in each season the number of trials performed is set sampling from a Poisson of parameter 2.5; in the first and second season all athletes are forced to do at least one trial, while from the third season to the fifth non participating is allowed (i.e. when zero is sampled from the Poisson). Correspondingly, times are provided for all trials performed by each athlete, divided per season,

[5] Seasons are represented by restaurants and athletes by customers; these couple of terms will be considered equivalent in the following

| Season | Active Components | Mixture coefficients |
|---|---|---|
| 1 | 2 4 6 8 10 | 0.2 0.2 0.2 0.2 0.2 |
| 2 | 2 4 6 10 | 0.2 0.2 0.3 0.3 |
| 3 | 4 6 10 | 0.3 0.4 0.3 |
| 4 | 4 6 8 | 0.4 0.2 0.4 |
| 5 | 2 4 6 8 10 | 0.1 0.2 0.4 0.2 0.1 |

Table 1: Mixture specification per each season

sampling from a uniform. The whole time span is set to be unitary. Finally, we define a general mixture of five Gaussians, centered respectively in 2,4,6,8,10, which is specified in each season according to the following table:

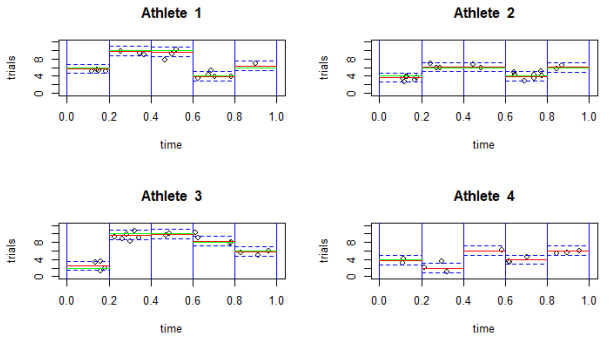



Figure 11: Comparison between real mean (green line) and corresponding estimate (red line) for 4 athletes; if only one bar is showed, values are exactly the same

As far as the variance is concerned, it is fixed at 0.5 for each component, and then jittered (0.1 range).
The obtained dataset mimics the true one, but is much more manageable; in addition, keeping trace of the true mean generating each datum, we are able to check how well the model is estimating the mixture component (i.e. the mean) per each athlete, which is actually the primary task we are required to perform on the Shotput dataset.

As it can be seen from Figure 11, such task is actually well performed by the model, which produces estimates that are very close to the real ones. The dashed blue lines (provided per each values) represent a credible 95 % span for data generated by a Gaussian centered in the guessed value, and are built upon the MCMC estimate of the variance. Checking if most of the data are contained in such span, we are able to verify if the common variance model estimate is too restrictive for the considered data or not [6].

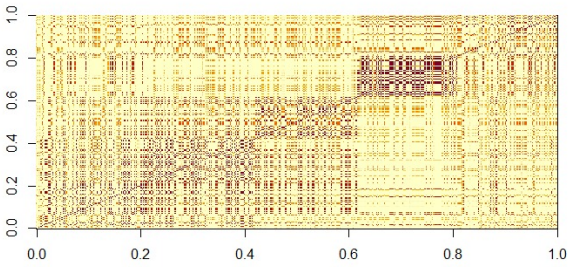


Figure 12: Posterior similarity matrix

Speaking from a pointwise perspective, we can actually be satisfied by the behaviour of the model on the simulated dataset. However, as the latter is built to keep a good separation between each component, performances may drop when considering real data. Thus, to boost our analysis, we decide to look also at the latent clustering structure produced by the model, to check if it is coherent with data.
Already by looking at the posterior similarity matrix (see Figure 12), produced from the MCMC cluster labelling, the existence of five clusters, spot by the algorithm, is clear; in Figure 13 is reported an heatmap of the posterior similarity matrix, arranged through a classical hierarchical clustering, which confirms this idea (note that this happens also because we are considering small, circumscribed dataset).

[6] Obviously, in this case it is good

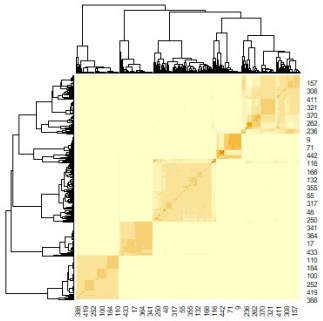


Figure 13: Heatmap of the PSM and hierarchical clustering

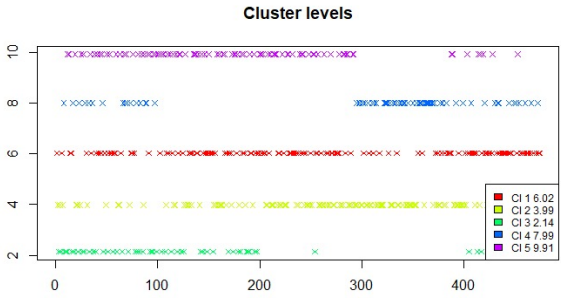



Figure 14: Estimate of the levels of clustering

To find the optimal latent partition, we firstly adopt a classical Binder loss minimization procedure, but then we decided to follow the approach suggested by Wade and Ghahramani [2]. That basically consist in minimizing the Variation of Information between groups,defined as the sum of the entropies minus two times the mutual information.

In Figure 14 is reported the labelling of each athlete obtained through the described procedure; for each identified cluster the empirical mean of all observations relative to the cluster is provided, as it can be visualized as the central value around which the estimates produced by the model are concentrated, when a particular athlete is assigned to the cluster in that season. It can be seen that actually they are very close to the true generative means of the mixture we considered to generate the data.

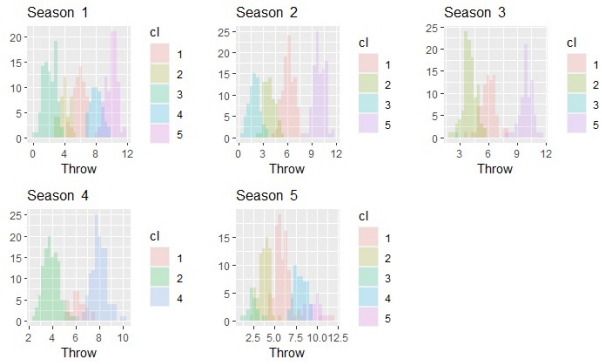



Figure 15: Histograms per cluster in each season

In Figure 15 are instead reported the histograms of observations in each season, divided as told by the optimal partition. In this case, where we know the true underlying distribution, it can be seen as a further proof of the goodness of the model; in principle, it can be useful to study a first approximation of the distribution of real data in each season, but also how the unique values are shared and distributed across season. Finally, it may be interesting to check how the intra-seasonal clustering changes tweaking the hierarchical Dirichlet process parameters.

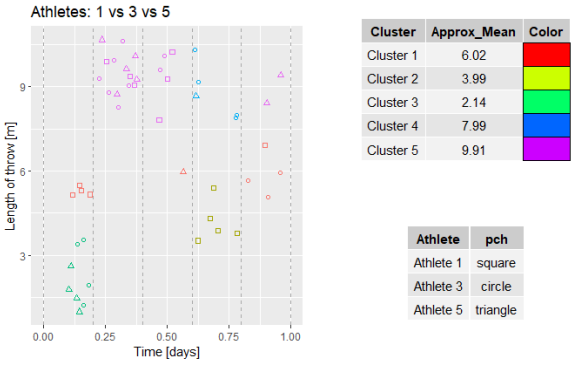



Figure 16: Visualization of the posterior clustering for 3 athletes

# 4 Application: Shotput Dataset

The dataset at our disposal is composed by 55390 observations of shotput trials, collected from 1973 to 2016. Each observation specifies athlete ID, time at which trials is taken (counted in days, starting from day of the first trial) and the relative activity season. These are the variables taken into account in order to estimate the step component of the model specified in the introduction. It is important to note, however, that other covariates are provided (gender, age, environment and year), and they supposedly may have an important role: for example, anti doping rules changed over time and so did levels of performances; male and female athletes clearly performs differently, etc (see Further directions of work for more details).

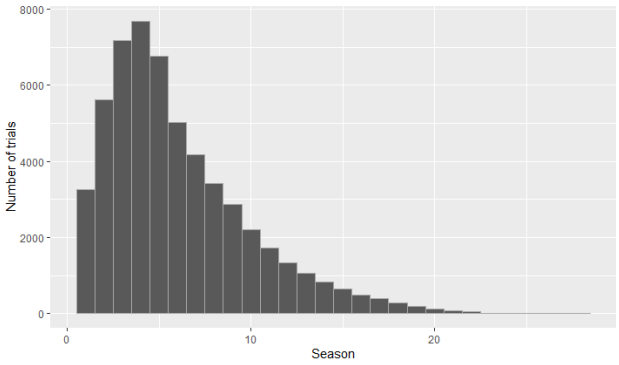



Figure 17: Number of trials per season

In order to carry out our analysis, we decided to take into account the first ten seasons of activity per each athlete, out of a total number of season of twenty seven; that is done in order to obtain a more manageable dataset, yet still broad enough. The obtained subset counts 48193 observations, with a total number of athletes of 1086. Recall that athletes presence in each season is not assured, see Table 2 for details.

| Season | Active athletes | Total throws |
|---|---|---|
| 1 | 1086 | 3261 |
| 2 | 911 | 5625 |
| 3 | 897 | 7178 |
| 4 | 859 | 7691 |
| 5 | 752 | 6754 |
| 6 | 611 | 5029 |
| 7 | 500 | 4171 |
| 8 | 405 | 3411 |
| 9 | 342 | 2875 |
| 10 | 272 | 2198 |

Table 2: Reduced dataset: Number of athletes and trials per season

We can explain the decrease of athletes due to retirement; note that an interesting trend is shown by the behaviour of Total number of throws: athletes tend to participate to few competitions the first year of activity, then number of trials per athlete grows rapidly, reaching is peak in the fourth season; then, the trend decreases (in a softer way).

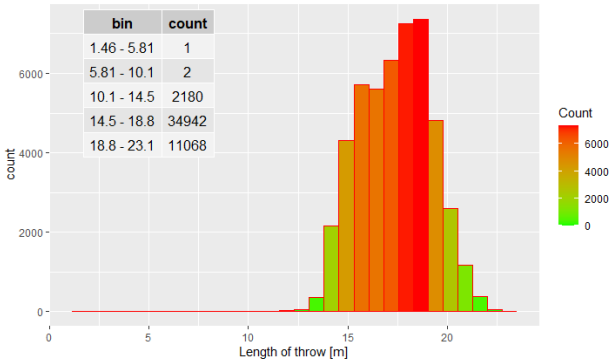



Figure 18: Histogram of reduced dataset

Another interesting trend can be identified looking at Figure 19: mean value of performances tend to increase along season of activity; moreover, from the same figure it is possible to spot the presence of few outliers, which are, probably, simple unfortunate trials. We opted not to remove them manually, as the model is believed to be capable

of dealing with them on his own.

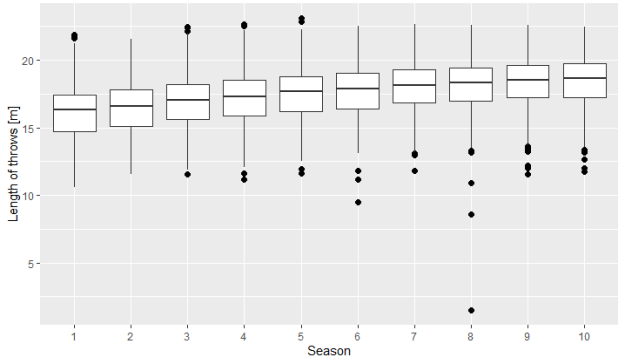



Figure 19: Seasonal evolution

| Model | $\alpha$ | $\gamma$ |
|---|---|---|
| 1 | 0.1 | 0.1 |
| 2 | 0.1 | 1 |
| 3 | 1 | 0.1 |
| 4 | 1 | 1 |
| 5 | 0.1 | 5 |
| 6 | 5 | 0.1 |
| 7 | 1 | 5 |
| 8 | 5 | 1 |
| 9 | 5 | 5 |

Table 3: Hyperparameters choices

## 4.1 Parameters setting

The baseline measure parameters (mean and variance of the conjugated Gaussian) are set in a data-driven manner (in particular, the mean is set equal to the empirical mean of data, while the variance in order to span properly the whole range of data). Inverse Gamma parameters are, instead, chosen looking at empirical variance of data. Hierarchical Dirichlet Process parameters ($\alpha$ and $\gamma$) are chosen through an a-posteriori cross validation procedure: the best model is chosen among the proposed ones (see Table 3) running the algorithm for 1000 iterations and then comparing results. Log-Pseudo Marginal Likelihood is taken into account to assess the goodness of fit, but we focus also our attention on the number of identified clusters to keep the model interpretable (and prevent overfitting).

## 4.2 Latent partition analysis

In this section we try to do a simple sensitivity analysis with respect to $\alpha$ and $\gamma$, regarding the optimal latent partition found following the same procedure described in Section 3.4. We recall that analyzing the latent partition implied by the model is useful to grasp some insights about the identified distributions of data.

First thing to note is how the number of clusters and the corresponding allocation of points varies with respect to the choice of hyperparameters. As expected, as $\alpha$ increases so does the number of unique values (clusters), while the converse happens for $\gamma$: in Figure 20 is reported the clustering of all observations across seasons of four different models (see Table 3), showing respectively two, six, nine and thirteen identified clusters. It is immediately clear at that Model 3 underestimates the true partition, while models 1 and 6 seems to behave better; Model 9 seems to present the best behaviour in terms of stratification (and actually achieved a slightly better fitting score with respect to Model 1 and 6, which are comparable), but the presence of thirteen clusters may result in a more difficult interpretation.

Actually, that concept appears even clearer looking at Figure 21, where is reported Season 3 internal clustering for each one of the previous models. Model 3 is unable to spot also the two major peaks, visible also to the eye. Model 1 performs well, identifying the principal peaks and also some smaller ones which were masked before; Model 6 behaves similarly, but is less effective on the smaller components. Model 9 is, as ex-

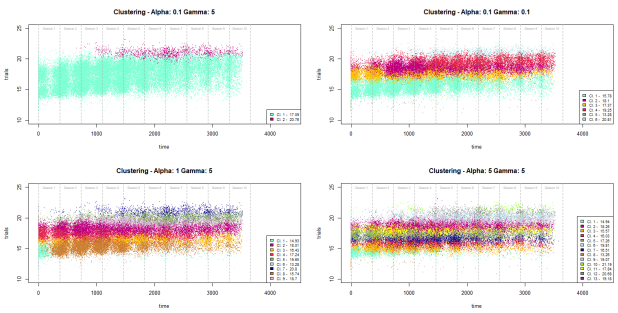


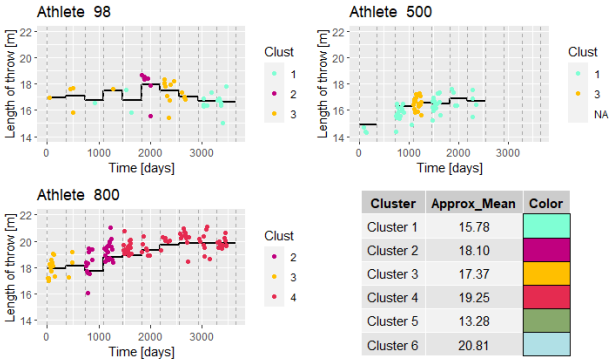

Figure 20: All points clustered - Model: 3 - 1 - 6 - 9

pected, very highly detailed, but the interpretation of the different components is almost completely lost.
Given the purpose of this report, we choose to stick to Model 1, which represent the best compromise, for the final representation of the step functions.


Figure 21: Season 3 internal clustering - Model: 3 - 1 - 6 - 9

## 4.3 Step function Estimation

In this section are reported, as an example, the final estimates of the step function for a randomly chosen set of athletes (98 - 500 - 800). In Figure 22, it is possible to see the function estimation (black bold line) across seasons, as well as trials of each athlete, coloured according to the cluster they were assigned by the algorithm.
Some interesting observations can be made: the ability of the algorithm to deal




| Cluster | Approx_Mean | Color |
|---|---|---|
| Cluster 1 | 15.78 | |
| Cluster 2 | 18.10 | |
| Cluster 3 | 17.37 | |
| Cluster 4 | 19.25 | |
| Cluster 5 | 13.28 | |
| Cluster 6 | 20.81 | |

Figure 22: Step function estimates - Athlete: 98-500-800

with outlying values is confirmed ( as it can be seen looking at Athlete 98 Season 6 or Athlete 800 Season 3), while it seems to be a bit less robust when small number of observations are available in a season. Furthermore, it is possible to give an explanation to each athlete's career: for example, athlete 500 starts from a low level of performance, then improve ( this is a typical trend we see in several cases) but still stick to a mediocre level for five seasons, and that probably led him to retirement; instead, athlete 800 keeps improving his performance levels, and thus keeps performing all throughout the ten years span considered.

## 4.4 Further directions of work

To conclude, in this paragraph are summarized some ideas which could constitute an interesting way to continue the analysis of this problem. As first consideration, it is crucial to keep in mind that the whole work on the Shotput Dataset is done mainly with exploratory purposes,considering a small number of iterations (1000) per each run, due to the time and power constraints. In the following, instead, are reported two possible enhancement of the model:

- Covariates inclusion: as pointed out

in the presentation of the Shotput Dataset, few additional covariates were provided, and some of them (sex, age, environment) likely have an effect on athletes performances. Thus, a complete additive functional model, which adds to the smooth and step components also a regression term based on such effects would be a good idea.

- Variance inclusion in the mixture: as specified in Section 3.4, the model we developed is based on a key assumption of equality of variances for all the mixture components. While looking at produced estimates the model seems to behave well, a drawback must exist. As an example, we can look at Figure 23: dashed blue lines represent the 95 % span containing data of a Gaussian centered in each guessed mean, based on the MCMC estimate of the variance. Most of the data are actually inside the identified intervals, but few of them lay still outside. Assuming this plot representative of the behaviour of almost all other athletes, and also as testified by the high values produced during the LPML computation, it would be clear that a model built along the lines of the one presented in Section 3.2 should perform better in such sense. However, we recall that the computational cost and the time required to run the algorithm is likely to increase considerably, as experienced during the analysis showed in Section 3.2.

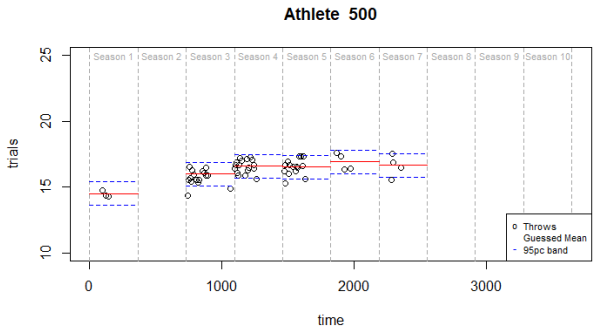



Figure 23: Athete 500: mean estimates and bands

# References


[1] Raffaele Argiento, Andrea Cremaschi, and Marina Vannucci. “Hierarchical Normalized Completely Random Measures to Cluster Grouped Data”. In: *Journal of the American Statistical Association,* (2020).

[2] Sara Wade and Zoubin Ghahramani. “Bayesian Cluster Analysis: Point Estimation and Credible Balls”. In: *Bayesian Analysis* 13.2 (2018), pp. 559–626.

[3] Peter Hoff. *A First Course in Bayesian Statistical Methods*. Springer, 2019, pp. 125–147.

[4] Peter Muller and Riten Mitra. “Bayesian Nonparametric Inference - Why and How”. In: *Bayesian Analysis* 8.2 (2013), pp. 269–302.

[5] Yee Whye Teh et al. “Hierarchical Dirichlet Processes”. In: *Journal of the American Statistical Association,* (2006).

[6] Patric Dolmeta, Raffaele Argiento, and Silvia Montagna. “Bayesian GARCH Modeling of Functional Sports Data”. In: *Unpublished* ().