BNPlib for density estimation:
A nonparametric C++ library (part 2)

Bruno Guindani
Elena Zazzetti

January 8th, 2020

POLITECNICO MILANO 1863

https://github.com/poliprojects/BNPlib

Model

$$\begin{aligned}(Y_i|\vartheta_i) &\sim F(\cdot,\vartheta_i)\\ (\vartheta_i|G) &\sim G\\ G &\sim DP(M,G_0)\end{aligned} \qquad \overset{K\to\infty}{\Longleftrightarrow} \qquad \begin{aligned}(Y_i|\phi,c_i) &\sim F(\cdot,\phi_{c_i})\\ (c_i|\mathbf{p}) &\sim \sum_{k=1}^{K} p_k\delta_k(\cdot)\\ \phi_c &\sim G_0\\ \mathbf{p} &\sim \mathrm{Dir}(M/K,\ldots,M/K)\end{aligned}$$

(hierarchical model)

(K-discrete model)

with $\boldsymbol{\vartheta} \leftrightsquigarrow (\boldsymbol{\phi},\mathbf{c})$

Algorithms

- Neal2, Neal8, blocked Gibbs
- **Gibbs sampling** procedures
- General structure:
  - sample **allocations** c from some conditional distribution
  - sample **unique values** ϕ from some conditional distribution
  - (sample **weights** p of the unique values deltas)

General structure

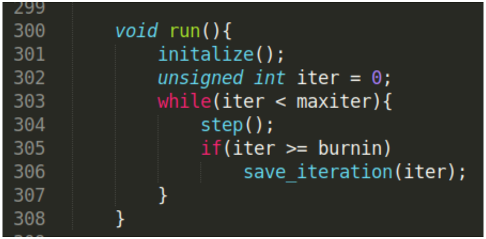
- Libraries: Stan + Eigen
- `Algorithm<Hierarchy, Mixture, Hypers>`
- Specializations

Classes

- `Hypers`:
  - ...
- `Hierarchy`: ...
- `Mixture`

The algorithms in C++

`Algorithm<Hierarchy, Mixture, Hypers>`



```cpp
    void run(){
        initalize();
        unsigned int iter = 0;
        while(iter < maxiter){
            step();
            if(iter >= burnin)
                save_iteration(iter);
        }
    }
```

- Example: `Hierarchy = Normal-NormalInvGamma`, `Hypers = HypersFixed`
- `initalize()`: random allocation
- `step()`
  - `sample_allocations()`: vector card of cardinalities of clusters
  - 4 cases handled separately: singleton vs !singleton, aux vs old
  - `sample_unique_values()`: vector `clust_idxs` to record which data are in each cluster
- Actual cluster structures?

Impending extensions

- **Hyperpriors**: objects of class `Hypers` store pointers to objects of class `HypersFixed`
- **Non-conjugacy**: via Stan's HMC sampler
- **R interface**: via **protocol buffers**

Protocol Buffers

- API developed by Google
- Data is saved in XML-like structures, called **messages**, that are defined in `.proto` files
- Each message corrresponds to a class in C++
- The `protoc` compiler produces the C++ files that make up the API
- RProtoBuf
- Compromise between efficiency and human-readibility

A general library?

Fully abstract library for all distributions?

- Hierarchies
- Updates
- Non-conjugacy

...but Stan functions cannot take vectors of parameters
$\Longrightarrow$ argument unpacker?

Bibliography

- Muller, Quintana, *Bayesian Nonparametric Data Analysis*
- Neal (2000), *Markov Chain Sampling Methods for Dirichlet Process Mixture Models*
- Ishwaran, James (2001), *Gibbs Sampling Methods for Stick-Breaking Priors*
- https://developers.google.com/protocol-buffers/docs/cpptutorial