

# BNPlib for density estimation:

A nonparametric C++ library  
(part 2)

**Bruno Guindani**  
**Elena Zazzetti**

January 8<sup>th</sup>, 2020



**POLITECNICO**  
MILANO 1863

<https://github.com/poliprojects/BNPlib>

# Model

$$(Y_i|\vartheta_i) \sim F(\cdot, \vartheta_i)$$

$$(\vartheta_i|G) \sim G$$

$$G \sim DP(M, G_0)$$

(hierarchical model)

$$(Y_i|\phi, c_i) \sim F(\cdot, \phi_{c_i})$$

$$(c_i|\mathbf{p}) \sim \sum_{k=1}^K p_k \delta_k(\cdot)$$

$$\phi_c \sim G_0$$

$$\mathbf{p} \sim \text{Dir}(M/K, \dots, M/K)$$

( $K$ -discrete model)

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

# Algorithms

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

# C++ Implementation

## Libraries:

- Stan Math: C++ template library for automatic differentiation of any order. It includes a range of built-in functions for probabilistic modeling, linear algebra, and equation solving.
  - ▶ Intel TBB library
  - ▶ Sundials library
- Eigen: C++ template library for linear algebra.

# Structure of Algorithm

**Structure:** Algorithm<Hierarchy, Mixture>

```
Algorithm<Hierarchy, Mixture>
```

```
Mixture mixture  
vector<data_type> data  
vector<unsigned int> allocations  
vector<Hierarchy> unique_values
```

```
Hierarchy<Hypers>
```

```
state_type state  
shared_ptr <Hypers> hypers  
void draw()  
void sample_given_data()
```

# Specializations

**Hierarchy** : NNIGHierarchy (Normal-NormalInvGamma)

- draw()
  - ▶ stan::math::inv\_gamma\_rng
  - ▶ stan::math::normal\_rng
- sample\_given\_data()
  - ▶ NormalGammaUpdate
  - ▶ stan::math::inv\_gamma\_rng
  - ▶ stan::math::normal\_rng

**Mixture** : SimpleMixture  $\rightarrow$  TotalMass

**Hypers** : HypersFixed  $\rightarrow \mu_0 \lambda_0 \alpha_0 \beta_0$

**Algorithm** Base → **Neal8** Derived

Neal8<Hierarchy, Mixture>
Mixture mixture vector<data_type> data vector<unsigned int> allocations vector<Hierarchy> unique_values vector<Hierarchy> aux_unique_values

# The algorithms in C++

Algorithm<Hierarchy, Mixture>

```
299
300     void run(){
301         initialize();
302         unsigned int iter = 0;
303         while(iter < maxiter){
304             step();
305             if(iter >= burnin)
306                 save_iteration(iter);
307         }
308     }
```

- Example: Hierarchy = Normal-NormalInvGamma, Hypers = HypersFixed
- initialize(): 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\_idx 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 corresponds to a class in C++
- The protoc compiler produces the C++ files that make up the API
- RProtoBuf
- Compromise between efficiency and human-readability

# A general library?





Fully abstract library for all distributions?

- Hierarchies
- Updates
- Non-conjugacy

... but Stan functions cannot take vectors of parameters

⇒ variadic template + 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>