BNPlib for density estimation:
A nonparametric C++ library

Bruno Guindani
Elena Zazzetti

November 22$^{\text{nd}}$, 2019

POLITECNICO
MILANO 1863

https://github.com/poliprojects/BNPlib

Non-Parametric Statistics

- Goal: density estimation
- **Infinite-dimensional** parameters
- For example: functions
- Bayesian Non-Parametric (**BNP**) model:

$$y_i | G \overset{\text{iid}}{\sim} G, \quad i = 1, \ldots, n$$
$$G \sim \mathscr{P} \quad (G \text{ random pr. measure})$$

Dirichlet Process Prior

$$y_i | G \overset{\text{iid}}{\sim} G, \quad i = 1, \ldots, n$$
$$G \sim \mathscr{P} = DP(MG_0)$$

- Parameters: $M > 0$, $G_0 \in M(S)$

- Defining property: $\forall \{B_{1:k}\}$ partition of S,

$$[G(B_1), \ldots, G(B_k)] \sim \text{Dir}\left(MG_0(B_1), \ldots, MG_0(B_k)\right)$$

- **Discreteness** (stick-breaking): $G(\cdot) = \sum_{k=1}^{+\infty} w_k \delta_{m_k}(\cdot)$

- **Conjugacy**: $G | \mathbf{y} \sim DP(MG_0 + \sum_{i=1}^{n} \delta_{y_i}) \implies$ density estimation

Continuous Density Estimation

- **Mixtures** (kernel F + mixing distribution G):

$$y_i|G \sim F_G(y) = \int F(y, \vartheta) \, G(\mathrm{d}\vartheta), \quad i = 1, \ldots, n$$
$$G \sim DP(MG_0)$$

- Model name: **DPM model**
- Equivalent to:

$$y_i|\vartheta_i \overset{\perp\!\!\!\perp}{\sim} F(\cdot, \vartheta_i), \quad i = 1, \ldots, n$$
$$\vartheta_i|G \overset{\text{iid}}{\sim} G, \quad i = 1, \ldots, n$$
$$G \sim DP(MG_0)$$
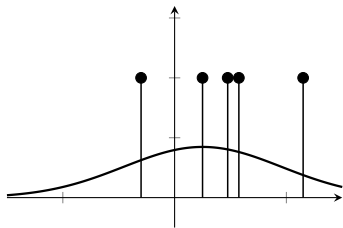
- ϑ_i are the *latent variables*, one for each data unit

Clustering In The DPM

- Discreteness: the ϑ_i have one of k **unique values** ϕ_j
- Data units: $i = 1, \ldots, n$
- Unique values: $j = 1, \ldots, k \simeq M \log n \ll n$
- c_i **allocation** parameters to the clusters: $c_i = j$ if $\vartheta_i = \phi_j$

Clustering In The DPM

- Discreteness: the ϑ_i have one of k **unique values** ϕ_j
- Data units: $i = 1, \ldots, n$
- Unique values: $j = 1, \ldots, k \simeq M \log n \ll n$
- c_i **allocation** parameters to the clusters: $c_i = j$ if $\vartheta_i = \phi_j$
- Conditional prior for ϑ_i, $i = 1, \ldots, n$:

$$\mathcal{L}(\vartheta_i | \boldsymbol{\vartheta}_{-i}) \propto \sum_{j=1}^{k^-} n_j^- \delta_{\phi_j^-}(\vartheta_i) + M G_0(\vartheta_i)$$

$$\uparrow \qquad\qquad \uparrow$$

- Conditional posterior for ϑ_i:

$$\mathcal{L}(\vartheta_i | \boldsymbol{\vartheta}_{-i}, y_i) \propto \sum_{j=1}^{k^-} F(y_i, \vartheta) \, \delta_{\phi_j^-}(\vartheta_i) + M \, r_i \, G_0(\vartheta_i | y_i)$$

Neal's Algorithm 2

Gibbs sampling algorithm:

- $(\phi, \mathbf{c})$ is the **state** of a Markov chain
- For $i = 1, \ldots, n$: update c_i
    - If c_i allocates ϕ_i to a singleton, remove ϕ_{c_i} from the state
    - Sample c_i as follows:

If $c = c_j$ for some $j \neq i$: $\mathbb{P}(c_i = c | \mathbf{c}_{-i}, y_i, \boldsymbol{\phi}) \propto \dfrac{n_{-i,c}}{n-1-M} F(y_i, \phi_c)$

$$\mathbb{P}(c_i \neq c_j \text{ for all } j | \mathbf{c}_{-i}, y_i, \boldsymbol{\phi}) \propto \frac{M}{n-1-M} \int F(y_i, \phi) \, G_0(\mathrm{d}\phi)$$

   - If the new c_i allocates ϕ_i to a singleton, draw $\phi_{c_i} \sim G_0(\cdot | y_i)$ and add it to the state
- For $c \in \{c_1, \ldots, c_n\}$: update ϕ_c, given all the y_i with $c_i = c$

Advantages

- Feasible if we can compute the $\int F(y_i, \phi)G_0(\mathrm{d}\phi)$ and sample from the $G_0(\cdot|y_i)$ (conjugate case)
- Change the ϑ for more than one observation simultaneously

Neal's Algorithm 8

- **Gibbs sampling** on the state, which is extended by the addition of m **auxiliary parameters**
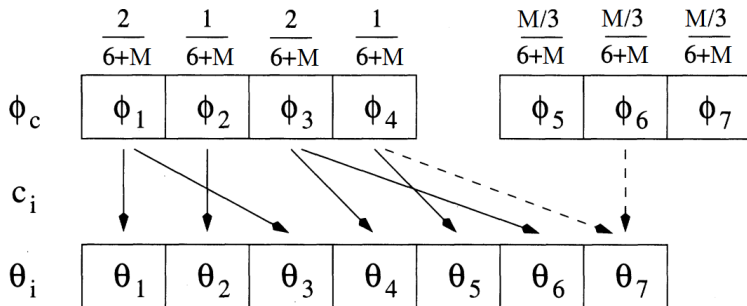
- Prior for c_i:

If $c = c_j$ for some j: $\mathbb{P}(c_i = c | \mathbf{c}_{-i}) = \dfrac{n_{-i,c}}{n - 1 - M}$

$$\mathbb{P}(c_i \neq c_j \text{ for all } j) = \dfrac{M}{n - 1 - M} \Rightarrow \quad \begin{array}{l} \textbf{split} \text{ among the } m \\ \text{auxiliary parameters} \end{array}$$

Neal's Algorithm 8

Algorithm:

- For $i = 1, \ldots, n$: update c_i
  - ▶ Sample auxiliary parameters:
    - ◦ $c_i = c_j$ for some $j \Rightarrow$ no connection
    - ◦ $c_i \neq c_j \Rightarrow$ association to one of m

    The other ϕ values drawn from G_0
  - ▶ Draw c_i as follows:

  $$P(c_i = c | \mathbf{c}_{-i}, y_i, \phi_1, ..., \phi_h) \propto \begin{cases} \frac{n_{-i,c}}{n-1-M} F(y_i, \phi_c), & \text{for } 1 \leq c \leq k^- \\ \frac{M/m}{n-1-M} F(y_i, \phi_c), & \text{for } k^- + 1 < c \leq h \end{cases}$$

  - ▶ Discard values in ϕ not associated to any ϑ_j
- For $c \in \{c_1, .., c_n\}$: update ϕ_c given y_i such that $c_i = c$

Advantages

- Models with non-conjugate priors
- As $m \to +\infty$ it approaches Algorithm 2 but equilibrium distribution is exact
- More efficient than similar algorithms (e.g. no-gaps)
- Hierarchical extensions

Stick-Breaking Priors

$$\mathscr{P}(\cdot) = \sum_{k=1}^{N} p_k \delta_{Z_k}(\cdot)$$

with $k = 1, \ldots, N$ and:

- $Z_k \overset{\text{iid}}{\sim} H$ (allocations)
- $V_k \overset{\text{iid}}{\sim} \text{Beta}(a_k, b_k)$ with $\mathbf{a} = (a_1, a_2, \ldots)$ and $\mathbf{b} = (b_1, b_2, \ldots)$
- $p_k = (1 - V_1)(1 - V_2) \cdots (1 - V_{k-1}) V_k$ (weights),

  with $0 \leq p_k \leq 1$, $\sum_{k=1}^{N} p_k = 1$

Dimension:

- $N < +\infty$: $\mathscr{P}_N(\mathbf{a}, \mathbf{b})$
  - $\mathbf{p} \sim \mathscr{GD}(\mathbf{a}, \mathbf{b})$ (Generalized Dirichlet)
  - e.g. all finite dimensional Dirichlet priors
- $N = +\infty$: $\mathscr{P}_\infty(\mathbf{a}, \mathbf{b})$
  - e.g. Dirichlet process, the two-parameter Poisson-Dirichlet process

Blocked Gibbs Algorithm

- Assumption: **finite-dimensional** prior $P \sim \mathscr{P}_N(\mathbf{a}, \mathbf{b})$
- Finite number of variables $\Rightarrow$ *blocks of parameters*
- Model:

$$(y_i | \phi, \mathbf{c}) \overset{\perp\!\!\!\perp}{\sim} F(\cdot, \phi_{c_i}), \quad i = 1, \dots, n$$

$$(c_i | \mathbf{p}) \overset{\text{iid}}{\sim} \sum_{k=1}^{N} p_k \delta_k(\cdot), \quad i = 1, \dots, n$$

$$\mathbf{p} \sim \mathscr{G}\mathscr{D}(\mathbf{a}, \mathbf{b})$$

$$\phi_c \sim G_0, \quad c \in \{c_1, \dots, c_n\}$$

Blocked Gibbs Algorithm

Algorithm:

- Repeatedly draw values from the conditional distributions of the blocked variables:

$$\phi \sim \mathcal{L}(\phi|\mathbf{c}, \mathbf{y})$$
$$\mathbf{c} \sim \mathcal{L}(\mathbf{c}|\phi, \mathbf{p}, \mathbf{y})$$
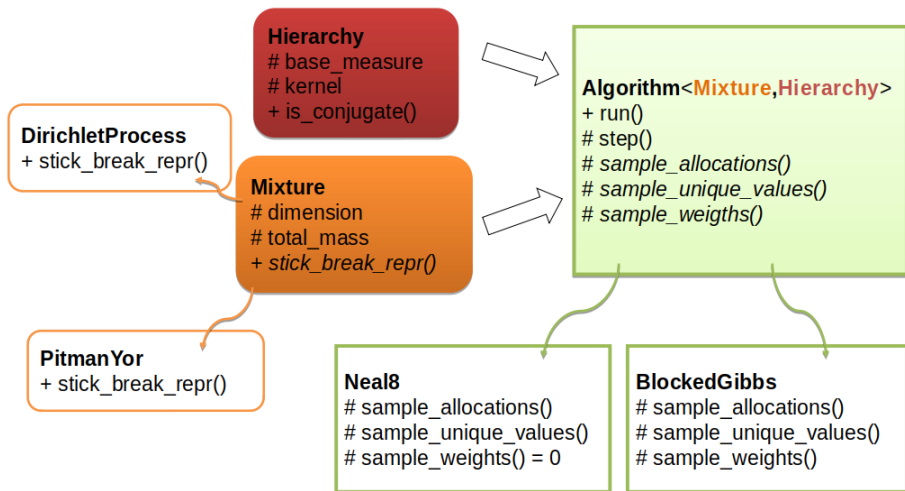$$\mathbf{p} \sim \mathcal{L}(\mathbf{p}|\mathbf{c})$$

Direct sampling of the **posterior** $\mathscr{P}(\cdot|\mathbf{y})$:

- The algorithm produces draws from $(\phi, \mathbf{c}, \mathbf{p}|\mathbf{y})$

- Each draw $(\phi, \mathbf{c}, \mathbf{p})$ defines a measure $P(\cdot) = \sum_{k=1}^{N} p_k \delta_{\phi_k}(\cdot)$

- Each P is a drawn from $\mathscr{P}(\cdot|\mathbf{y})$

Advantages

- Handles the issue of conjugacy
- Good mixing
- Hierarchical extensions

Code Structure

Hierarchy
\# base_measure
\# kernel
+ is_conjugate()

DirichletProcess
+ stick_break_repr()

Mixture
\# dimension
\# total_mass
+ *stick_break_repr()*

PitmanYor
+ stick_break_repr()

Algorithm<**Mixture**,**Hierarchy**>
+ run()
\# step()
\# *sample_allocations()*
\# *sample_unique_values()*
\# *sample_weigths()*

Neal8
\# sample_allocations()
\# sample_unique_values()
\# sample_weights() = 0

BlockedGibbs
\# sample_allocations()
\# sample_unique_values()
\# sample_weights()

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*