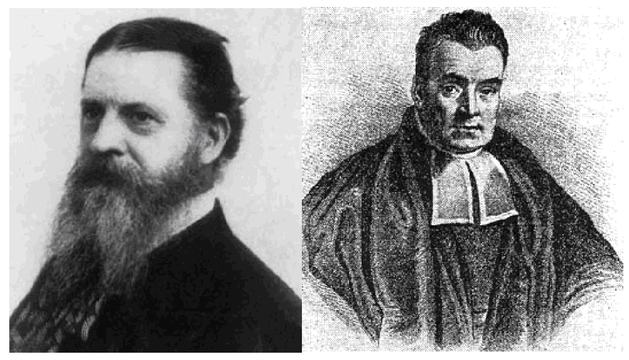
# Five probabilistic programs in `peircebayes`

**Calin Rares Turliuc**[1]*, **Luke Dickens**[2]
**Alessandra Russo**[1], **Krysia Broda**[1]
[1]Department of Computing, Imperial College London
[2]Department of Information Studies, University College London
* ct1810@imperial.ac.uk



**Abstract**

We present `peircebayes`, a probabilistic abductive logic programming language tailored for inference in models with discrete observed and latent variables endowed with Dirichlet priors. As with other probabilistic programming languages, `peircebayes` attempts to reduce the effort in developing probabilistic models by supporting programs that encode a generative model of the data and, when run, applying general inference algorithms to these models. In this poster, we show experiments with `peircebayes` on five probabilistic models: three flavours of Latent Dirichlet Allocation (LDA), namely (1) vanilla LDA, (2) LDA with seed constraints, and (3) hierarchical LDA; a model for inference over preferences, called the Repeated Insertion Model (RIM); and the Bayesian Prevalence Model (BPM), a model that has been used for deception rate estimation in positive hotel reviews. Our results show that `peircebayes` can be used to encode a variety of probabilistic models within its scope and to efficiently infer parameters of and make predictions with these models.

## Introduction

We present experiments with `peircebayes` [4], a probabilistic abductive logic programming language designed for inference in probabilistic models with categorical variables and Dirichlet priors. The plate notation of the models expressible in `peircebayes` is shown in the figure.

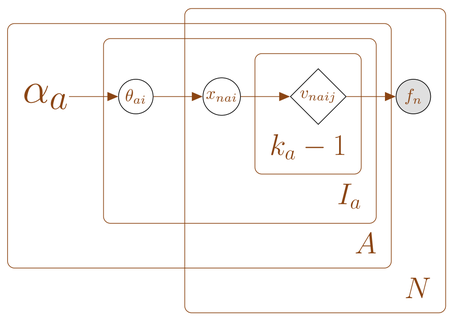



**Figure 1:** The PB plate model. Unbounded nodes are constants, circle nodes are latent variables, shaded nodes are observed variables, diamond nodes are deterministic variables. $A$, $I_a$, $N$, and $k_a$ are positive integers, $k_a \geq 2$.

$$\theta|\alpha \sim \text{Dirichlet}(\alpha) \qquad x|\theta \sim \text{Categorical}(\theta)$$

$$P(v_{nai*}|x_{nai} = l) = \begin{cases} \overline{v_{nai1}} \dots \overline{v_{nail-1}} v_{nail} & \text{, if } l < k_a \\ \overline{v_{nai1}} \dots \overline{v_{nail-1}} & \text{, if } l = k_a \end{cases}$$

where $v_{naij}$ is a boolean and $\overline{v}$ denotes boolean negation.

$$P(f_n|v_{n*}) = [f_n = \text{Bool}_n(v_{n*})] \qquad n = 1, \dots, N$$

where $\text{Bool}_n(v)$ denotes an arbitrary boolean function of variables $v$, and $[i = j]$ is the Kronecker delta function $\delta_{ij}$.

## BPM [3]

### Model and experimental setup

$$\pi|\alpha \sim \text{Beta}(\alpha) \qquad \eta|\beta \sim \text{Beta}(\beta) \qquad \theta|\gamma \sim \text{Beta}(\theta)$$
$$y|\pi \sim \text{Bernoulli}(\pi) \qquad x|\eta, \theta, y \sim y\,\text{Bernoulli}(\eta) + (1 - y)\,\text{Bernoulli}(1 - \theta)$$

We generate 1000 synthetic data points according to a model with priors $\alpha = (30, 70)$, $\beta = (21, 81)$, $\gamma = (11, 91)$, then use the same $\beta$ and $\gamma$ for inference, but a non-informative prior on $\alpha$.

### Program and results

```
observe(1, 424). observe(2, 576).
pb_dirichlet([1, 1], pi, 2, 1).
pb_dirichlet([21, 81], eta, 2, 1).
pb_dirichlet([11, 91], theta, 2, 1).
pb_plate( [observe(Class, Count)],
Count, [generate(Class)] ).
generate(Class) :- Y in 1..2, pi(Y, 1), generate(Class, Y).
generate(1, 1) :- theta(2, 1).
generate(2, 1) :- theta(1, 1).
generate(Class, 2) :- eta(Class,1).
```

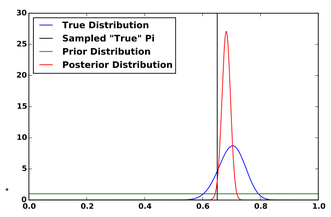



**Figure 2:** Results on synthetic BPM.

## LDA [1]

### Model and experimental setup

$$\theta|\alpha \sim \text{Dirichlet}(\alpha) \qquad \phi|\beta \sim \text{Dirichlet}(\beta) \qquad z|\theta \sim \text{Categorical}(\theta) \qquad x|z, \phi \sim \text{Categorical}(\phi_z)$$

We use the computing related (comp.*) newsgroups in the 20 newsgroups dataset. We select only test documents, tokenize and remove stop words to obtain a corpus with 35850 unique tokens in 1911 documents with average length of $\approx 108$ tokens. We set $T = 10$ topics and priors $\alpha = 50/T$, $\beta = 0.1$.

### Program and results

```
pb_dirichlet(5, theta, 10, 1911). pb_dirichlet(0.1, phi, 27206, 20).
pb_plate( [observe(d(Doc), TokenList), member((w(Token), Count), TokenList)],
    Count, [Topic in 1..10, theta(Topic,Doc), phi(Token,Topic)]).
observe(d(1), [(w(20600),1),(w(3987),1),(w(19837),1), ... ]). ...
```

We show wordclouds for three particularly coherent topics:

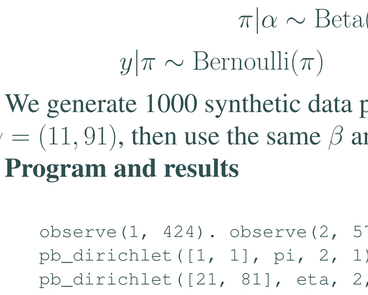



**Figure 3:** Linux, shell.

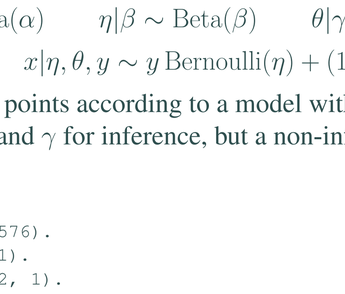



**Figure 4:** Images, graphics formats.

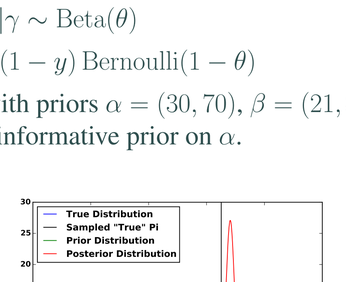



**Figure 5:** Monitors, displays.

## Seed LDA

### Model and experimental setup

Same as LDA, except $x$:

$$x_{NotSeed}|z, \phi \sim \text{Categorical}(\phi_z) \qquad x_{Seed}|s, \phi \sim \text{Categorical}(\phi_s)$$

We use the same dataset as for LDA, only we select all documents, tokenize, lemmatize and remove stop words to obtain a corpus with 27206 unique tokens in 4777 documents with average length $\approx 72$ tokens. We set $T = 20$ topics and priors $\alpha = 50/T$, $\beta = 0.01$. We seed two topics with hardware (hardware, machine, memory, cpu), and software (software, program, version, shareware) related terms.

### Results

We notice that both topics give high weights to the seed words, and other hardware (floppy, video, speed) or software (sun, character, viewer) related terms.


**Figure 6:** Hardware.


**Figure 7:** Software.

## Cluster LDA

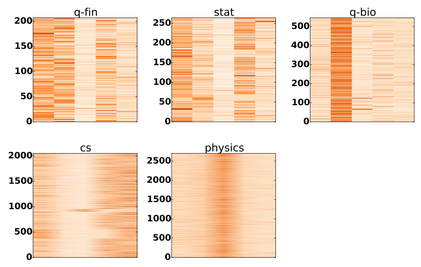
### Model and experimental setup

$$\psi|\gamma \sim \text{Dirichlet}(\gamma) \qquad \theta|\alpha \sim \text{Dirichlet}(\alpha) \qquad \phi|\beta \sim \text{Dirichlet}(\beta)$$
$$y|\psi \sim \text{Categorical}(\psi) \qquad z|\theta, y \sim \text{Categorical}(\theta_{Cluster(y)}) \qquad x|z, \phi \sim \text{Categorical}(\phi_z)$$

We collect all abstracts on arXiv submitted in 2007, from five categories: quantitative finance (q-fin), statistics (stats), quantitative biology (q-bio), computer science (cs), and physics (physics). We tokenize and remove stop words to obtain a corpus with 26834 unique tokens in 5769 documents with average length $\approx 80$ tokens.

### Results

We observe that q-bio and physics are clustered in the same topic cluster, computer science is characterized by cluster 5, but also 1 and 4, while q-fin and stats are very similar, consisting mainly of clusters 1,2 and 4. The latter effect is due to the small number of documents in both q-fin and stats, as well as the fact that most q-fin papers focus on statistical methods.




| | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| important | 0.0092 | physical | 0.0099 | scaling | 0.0115 | simulations | 0.0098 | structural | 0.0096 |
| expression | 0.0092 | proteins | 0.0092 | free | 0.0105 | response | 0.0093 | short | 0.0095 |
| model | 0.0092 | interaction | 0.0092 | similar | 0.0089 | activity | 0.0087 | stability | 0.0094 |
| fluctuations | 0.0085 | individual | 0.008 | genetic | 0.0079 | mean | 0.0084 | global | 0.009 |
| large | 0.0082 | scales | 0.0072 | transfer | 0.0067 | diffusion | 0.0084 | equilibrium | 0.0087 |
| mechanism | 0.0077 | transition | 0.0072 | agent | 0.0062 | rate | 0.0081 | studies | 0.0081 |
| specific | 0.0077 | mathematical | 0.0069 | brain | 0.006 | present | 0.008 | role | 0.008 |
| factors | 0.0075 | investigate | 0.0066 | chemical | 0.0058 | temporal | 0.0075 | experimental | 0.0078 |
| recent | 0.0074 | process | 0.0066 | exhibit | 0.0056 | mechanics | 0.0075 | statistics | 0.0074 |
| highly | 0.0073 | dynamical | 0.0062 | normal | 0.0056 | correlations | 0.0073 | influence | 0.0071 |

**Table 1:** Topic Cluster 2 (top 10 words).

**Figure 8:** Cluster mixture for each category (x - topic clusters, y - documents, darker colour - higher probability).

## RIM

### Model and experimental setup

$$\pi|\alpha \sim \text{Dirichlet}(\alpha) \qquad p_i|\beta_i \sim \text{Dirichlet}(\beta_i) \qquad i = 2, \dots, N$$

Inspired by [2], we use 5000 complete orderings of $N = 10$ Sushi ingredients to infer $K = 6$ preference profiles, i.e. distributions over permutations of $N$ items, using $\alpha = 50/K$, $\beta = 0.1$.

### Results

We reach similar conclusions as [2]: fatty tuna is universally preferred, cucumber roll is universally disliked and there is a strong positive correlation between sea urchin and salmon roe.

| $\pi_1 = 0.144$ | $\pi_2 = 0.191$ | $\pi_3 = 0.153$ | $\pi_4 = 0.185$ | $\pi_5 = 0.187$ | $\pi_6 = 0.138$ |
|---|---|---|---|---|---|
| fatty tuna | fatty tuna | fatty tuna | fatty tuna | fatty tuna | fatty tuna |
| shrimp | tuna | sea urchin | sea urchin | tuna | tuna |
| sea eel | shrimp | salmon roe | salmon roe | shrimp | salmon roe |
| squid | squid | sea eel | shrimp | sea eel | shrimp |
| tuna | sea eel | tuna | sea eel | squid | squid |
| tuna roll | tuna roll | shrimp | squid | tuna roll | sea eel |
| salmon roe | egg | tuna roll | tuna roll | egg | tuna roll |
| sea urchin | cucumber roll | squid | tuna | salmon roe | sea urchin |
| egg | salmon roe | egg | egg | cucumber roll | egg |
| cucumber roll | sea urchin | cucumber roll | cucumber roll | sea urchin | cucumber roll |

**Table 2:** Preference profile mixture and mode (top - most liked).

## Concluding remarks

Please see the missing `peircebayes` programs here: `https://goo.gl/3cfMhj`. For more information on `peircebayes`, see `http://raresct.github.io`. In future work, we aim to express and to test more models, to study inductive learning in `peircebayes` and to develop a non-parametric `peircebayes` based on work in Dirichlet processes.

## References

[1] David M. Blei, Andrew Y. Ng, Michael I. Jordan, and John Lafferty. Latent dirichlet allocation. *Journal of Machine Learning Research*, 3:2003, 2003.

[2] Tyler Lu and Craig Boutilier. Effective sampling and learning for mallows models with pairwise-preference data. *Journal of Machine Learning Research*, 15:3783–3829, 2014.

[3] Myle Ott, Claire Cardie, and Jeff Hancock. Estimating the prevalence of deception in online review communities. In *Proceedings of the 21st International Conference on World Wide Web*, WWW '12, pages 201–210, New York, NY, USA, 2012. ACM.

[4] C.R. Turliuc, L. Dickens, A. Russo, and K. Broda. Probabilistic abductive logic programming using dirichlet priors. In *Proceedings of the Second Workshop on Probabilistic Logic Programming*, 2015.