
Pitman-Yor Diffusion Trees

David A. Knowles, Zoubin Ghahramani
University of Cambridge

Abstract

We introduce the Pitman Yor Diffusion Tree (PYDT) for hierarchical clustering, a generalization of the Dirichlet Diffusion Tree (Neal, 2001) which removes the restriction to binary branching structure. The generative process is described and shown to result in an exchangeable distribution over data points. We prove some theoretical properties of the model and then present two inference methods: a collapsed MCMC sampler which allows us to model uncertainty over tree structures, and a computationally efficient greedy Bayesian EM search algorithm. Both algorithms use message passing on the tree structure. The utility of the model and algorithms is demonstrated on synthetic and real world data, both continuous and binary.

Tree structures play an important role in machine learning and statistics. Learning a tree structure over data points gives a straightforward picture of how objects of interest are related. Trees are easily interpreted and intuitive to understand. Sometimes we may know that there is a true underlying hierarchy: for example species in the tree of life or duplicates of genes in the human genome, known as paralogs. Typical mixture models, such as Dirichlet Process mixture models, have independent parameters for each component. We might expect for example that certain clusters are similar, for example are sub-groups of some large group. By learning this hierarchical similarity structure, the model can share statistical strength between components to make better estimates of parameters using less data.

Classical hierarchical clustering algorithms employ a bottom up “agglomerative” approach (Duda et al., 2001) which hides the statistical assumptions being made. Heller and Ghahramani (2005) use a principled

probabilistic model in lieu of a distance metric but simply view the hierarchy as a tree consistent mixture over partitions of the data. If instead a full generative model for both the tree structure and the data is used (Williams, 2000; Neal, 2003b; Teh et al., 2008; Blei et al., 2010) Bayesian inference machinery can be used to compute posterior distributions over the tree structures themselves. Such models can also be used to learn hierarchies over latent variables (Rai and Daumé III, 2008).

Both heuristic and generative probabilistic approaches to learning hierarchies have focused on learning binary trees. Although computationally convenient this restriction may be undesirable: where appropriate, arbitrary trees provide a more interpretable, clean summary of the data. Some recent work has aimed to address this. Blundell et al. (2010) extend Heller and Ghahramani (2005) by removing the restriction to binary trees. However, as for Heller and Ghahramani (2005) the lack of a generative process prohibits modeling uncertainty over tree structures. Williams (2000) allows nonbinary trees by having each node independently pick a parent in the layer above, but requires one to pre-specify the number of layers and number of nodes in each layer. Blei et al. (2010) use the nested Chinese restaurant process to define probability distributions over tree structures. Each data point is drawn from a mixture over the parameters on the path from the root to the data point, which is appropriate for mixed membership models but not standard clustering. An alternative to the PYDT to obtain unbounded trees is given by Adams et al. (2010). They use a nested stick-breaking process to construct the tree, which is then endowed with a diffusion process. Data live at internal nodes of the tree, rather than at leaves as in the PYDT.

We introduce the Pitman Yor Diffusion Tree (PYDT), a generalization of the Dirichlet Diffusion Tree (Neal, 2001) to trees with arbitrary branching structure. While allowing atoms in the divergence function of the

DDT can in principle be used to obtain multifurcating branch points (Neal, 2003b), our solution is both more flexible and more mathematically and computationally tractable. An interesting property of the PYDT is that the implied distribution over tree structures corresponds to the multifurcating Gibbs fragmentation tree (McCullagh et al., 2008), known to be the most general process generating exchangeable and consistent trees (here consistency can be understood as coherence under marginalization of subtrees).

Our contributions are as follows. In Section 1 we describe the generative process corresponding to the PYDT. In Section 2 we derive the probability of a tree, and in Section 3 show some important properties of the process. Section 4 describes our hierarchical clustering models utilising the PYDT. In Section 5 we present an MCMC sampler and a greedy EM algorithm, which we developed for the DDT in Knowles et al. (2011). We present results demonstrating the utility of the PYDT in Section 6. In the supplementary material we describe how to sample from the PYDT¹.

1 Generative process

We will describe the generative process for the data in terms of a diffusion process in fictitious “time” on the unit interval. The observed data points (or latent variables) correspond to the locations of the diffusion process at time $t = 1$. The first datapoint starts at time 0 at the origin in a D -dimensional Euclidean space and follows a Brownian motion with variance σ^2 until time 1. If datapoint 1 is at position $x_1(t)$ at time t , the point will reach position $x_1(t + dt) \sim N(x_1(t), \sigma^2 Idt)$ at time $t + dt$. It can easily be shown that $x_1(t) \sim \text{Normal}(0, \sigma^2 It)$. The second point x_2 in the dataset also starts at the origin and initially follows the path of x_1 . The path of x_2 will diverge from that of x_1 at some time T_d after which x_2 follows a Brownian motion independent of $x_1(t)$ until $t = 1$, with $x_i(1)$ being the i -th data point. The probability of diverging in an interval $[t + dt]$ is determined by a “divergence function” $a(t)$ (see Equation 1 below) which is analogous to the hazard function in survival analysis.

The generative process for datapoint i is as follows. Initially $x_i(t)$ follows the path of the previous data-points. If at time t the path of $x_i(t)$ has not diverged, it will diverge in the next infinitesimal time interval $[t, t + dt]$ with probability

$$\frac{a(t)\Gamma(m - \beta)dt}{\Gamma(m + 1 + \alpha)} \quad (1)$$

where m is the number of datapoints that have previously followed the current path and $0 \leq \beta \leq 1, \alpha \geq -2\beta$ are parameters of the model. In the special case of integer $\alpha \in \mathbb{N}$ and $\beta = 0$ the probability of divergence reduces to $a(t)dt/[m(m + 1) \dots (m + \alpha - 1)(m + \alpha)]$. For example for $\alpha = 1$ this gives $a(t)dt/[m(m + 1)]$, and for $\alpha = \beta = 0$ the DDT expression $a(t)dt/m$ is recovered. If x_i does not diverge before reaching a previous branching point, it may either follow one of the previous branches, or diverge at the branch point (adding one to the degree of this node in the tree). The probability of following one of the existing branches k is

$$\frac{b_k - \beta}{m + \alpha} \quad (2)$$

where b_k is the number of samples which previously took branch k and m is the total number of samples through this branch point so far. The probability of diverging at the branch point and creating a new branch is

$$\frac{\alpha + \beta K}{m + \alpha} \quad (3)$$

where K is the number of branches from this branch point. By summing Equation 2 over $k = \{1, \dots, K\}$ with Equation 3 we get 1 as required. This reinforcement scheme is analogous to the Pitman Yor process (Teh, 2006) version of the Chinese restaurant process (Aldous, 1985). For the single data point $x_i(t)$ this process is iterated down the tree until divergence, after which $x_i(t)$ performs independent Brownian motion until time $t = 1$. The i -th observed data point is given by the location of this Brownian motion at $t = 1$, i.e. $x_i(1)$.

2 Probability of a tree

We refer to branch points and leaves of the tree as nodes. The probability of generating a specific tree structure with associated divergence times and locations at each node can be written analytically since the specific diffusion path taken between nodes can be ignored. We will need the probability that a new data point does not diverge between times $s < t$ on a branch that has been followed m times by previous data-points. This can straightforwardly be derived from Equation 1:

$$P \left(\begin{array}{c} \text{not diverging} \\ \text{in } [s, t] \end{array} \right) = \exp \left[(A(s) - A(t)) \frac{\Gamma(m - \beta)}{\Gamma(m + 1 + \alpha)} \right], \quad (4)$$

where $A(t) = \int_0^t a(u)du$ is the cumulative rate function.

¹<http://mlg.eng.cam.ac.uk/dave/>

Consider the tree of $N = 4$ data points in Figure 1. The probability of obtaining this tree structure and associated divergence times is:

$$\begin{aligned}
& e^{-A(t_a)} \frac{\Gamma(1-\beta)}{\Gamma(2+\alpha)} \frac{a(t_a)\Gamma(1-\beta)}{\Gamma(2+\alpha)} \\
& \times e^{-A(t_a)} \frac{\Gamma(2-\beta)}{\Gamma(3+\alpha)} \frac{1-\beta}{2+\alpha} e^{-[A(t_a)-A(t_b)]} \frac{\Gamma(1-\beta)}{\Gamma(2+\alpha)} \frac{a(t_b)\Gamma(1-\beta)}{\Gamma(2+\alpha)} \\
& \times e^{-A(t_a)} \frac{\Gamma(3-\beta)}{\Gamma(4+\alpha)} \frac{\alpha+2\beta}{3+\alpha}
\end{aligned}$$

The first data point does not contribute to the expression. The second point contributes the first line: the first term results from not diverging between $t = 0$ and t_a , the second from diverging at t_a . The third point contributes the second line: the first term comes from not diverging before time t_a , the second from choosing the branch leading towards the first point, the third term comes from not diverging between times t_a and t_b , and the final term from diverging at time t_b . The fourth and final data point contributes the final line: the first term for not diverging before time t_a and the second term for diverging at branch point a .

The component resulting from the divergence and data locations for the tree in Figure 1 is

$$\begin{aligned}
& N(x_1; 0, \sigma^2) N(x_2; x_a, \sigma^2(1-t_a)) \\
& \times N(x_3; x_b, \sigma^2(1-t_b)) N(x_4; x_a, \sigma^2(1-t_a))
\end{aligned}$$

where each data point has contributed a term. We can rewrite this as:

$$\begin{aligned}
& N(x_a; 0, \sigma^2 t_a) N(x_b; x_a, \sigma^2(t_b - t_a)) \\
& \times N(x_1; x_b, \sigma^2(1-t_b)) \times N(x_2; x_a, \sigma^2(1-t_a)) \\
& \times N(x_3; x_b, \sigma^2(1-t_b)) N(x_4; x_a, \sigma^2(1-t_a)) \quad (5)
\end{aligned}$$

to see that there is a Gaussian term associated with each branch in the tree.

3 Theory

Now we present some important properties of the PYDT generative process.

Lemma 1. *The probability of generating a specific tree structure, divergence times, divergence locations and corresponding data set is invariant to the ordering of data points.*

Proof. The probability of a draw from the PYDT can be decomposed into three components: the probability of the underlying tree structure, the probability of the divergence times given the tree structure, and the probability of the divergence locations given the divergence times. We will show that none of these

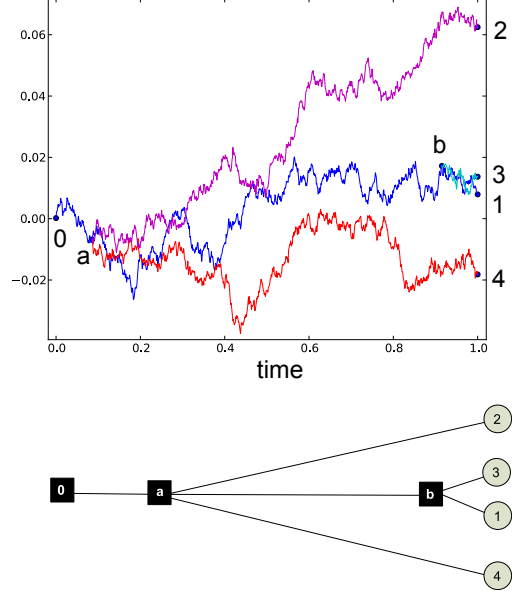


Figure 1: A sample from the Pitman-Yor Diffusion Tree with $N = 4$ datapoints and $a(t) = 1/(1-t)$, $\alpha = 1$, $\beta = 0$. Top: the location of the Brownian motion for each of the four paths. Bottom: the corresponding tree structure. Each branch point corresponds to an internal tree node.

components depend on the ordering of the data. Consider the tree \mathcal{T} as a set of edges $\mathcal{S}(\mathcal{T})$ each of which we will see contributes to the joint probability density. The tree structure \mathcal{T} contains the counts of how many datapoints traversed each edge. We denote an edge by $[ab] \in \mathcal{S}(\mathcal{T})$, which goes from node a to node b with corresponding locations x_a and x_b and divergence times t_a and t_b . Let $m(b)$ be the number of samples to have passed through b . Denote by $\mathcal{S}'(\mathcal{T}) = \{[ab] \in \mathcal{S}(\mathcal{T}) : m(b) \geq 2\}$ the set of all edges traversed by $m \geq 2$ samples (for divergence functions which ensure divergence before time 1 this is the set of all edges not connecting to leaf nodes).

Probability of the tree structure. For segment $[ab]$, let i be the index of the sample which diverged to create the branch point at b , thereby contributing a factor

$$\frac{a(t_b)\Gamma(i-1-\beta)}{\Gamma(i+\alpha)}. \quad (6)$$

Let the number of branches from b be K_b , and the number of samples which followed each branch be $\{n_k^b : k \in [1 \dots K_b]\}$. The total number of datapoints which traversed edge $[ab]$ is $m(b) = \sum_{j=1}^{K_b} n_j^b$. It can be shown (see Appendix A) that the factor associated with this branching structure for the data points after i is

$$\frac{\prod_{k=3}^{K_b} [\alpha + (k-1)\beta] \Gamma(i+\alpha) \prod_{l=1}^{K_b} \Gamma(n_l^b - \beta)}{\Gamma(i-1+\beta) \Gamma(m(b)+\alpha)}$$

Multiplying by the contribution from data point i in Equation 6 we have

$$\frac{a(t_b) \prod_{k=3}^{K_b} [\alpha + (k-1)\beta] \prod_{l=1}^{K_b} \Gamma(n_l^b - \beta)}{\Gamma(m(b) + \alpha)} \quad (7)$$

Each segment $[ab] \in \mathcal{S}'(\mathcal{T})$ contributes such a term. Since this expression does not depend on the ordering of the branching events, the overall factor does not either.

Probability of divergence times. The $m(b) - 1$ points that followed the first point along this path did not diverge before time t_b (otherwise $[ab]$ would not be an edge), which from Equation 4 we see contributes a factor

$$\begin{aligned} & \prod_{i=1}^{m(b)-1} \exp \left[(A(t_a) - A(t_b)) \frac{\Gamma(i - \beta)}{\Gamma(i + 1 + \alpha)} \right] \\ &= \exp \left[(A(t_a) - A(t_b)) H_{m(b)-1}^{\alpha, \beta} \right] \end{aligned} \quad (8)$$

where we define $H_n^{\alpha, \beta} = \sum_{i=1}^n \frac{\Gamma(i - \beta)}{\Gamma(i + 1 + \alpha)}$. All edges $[ab] \in \mathcal{S}'(\mathcal{T})$ contribute the expression in Equation 8, resulting in a total contribution

$$\prod_{[ab] \in \mathcal{S}'(\mathcal{T})} \exp \left[(A(t_a) - A(t_b)) H_{m(b)-1}^{\alpha, \beta} \right] \quad (9)$$

This expression does not depend on the ordering of the datapoints.

Probability of node locations. Generalizing Equation 5 it is clear that each edge contributes a Gaussian factor, resulting an overall factor:

$$\prod_{[ab] \in \mathcal{S}(\mathcal{T})} \mathcal{N}(x_b; x_a, \sigma^2(t_b - t_a)I) \quad (10)$$

The overall probability of a specific tree, divergence times and node locations is given by the product of Equations 7, 9 and 10, none of which depend on the ordering of the data. \square

The term $\prod_{k=3}^{K_b} [\alpha + (k-1)\beta]$ in Equation 7 can be calculated efficiently depending on the value of β . For $\beta = 0$ we have $\prod_{k=3}^{K_b} \alpha = \alpha^{K-2}$. For $\beta \neq 0$ we have

$$\begin{aligned} \prod_{k=3}^{K_b} [\alpha + (k-1)\beta] &= \beta^{K_b-2} \prod_{k=3}^{K_b} [\alpha/\beta + (k-1)] \\ &= \frac{\beta^{K_b-2} \Gamma(\alpha/\beta + K_b)}{\Gamma(\alpha/\beta + 2)} \end{aligned}$$

Theorem 1. *The Pitman-Yor Diffusion Tree defines an infinitely exchangeable distribution over data points.*

Proof. Summing over all possible tree structures, and integrating over all branch point times and locations, by Lemma 1 we have infinite exchangeability. \square

Corollary 1. *There exists a prior ν on probability measures on \mathbb{R}^D such that the samples x_1, x_2, \dots generated by a PYDT are conditionally independent and identically distributed (iid) according to $\mathcal{F} \sim \nu$, that is, we can represent the PYDT as*

$$PYDT(x_1, x_2, \dots) = \int \left(\prod_i \mathcal{F}(x_i) \right) d\nu(\mathcal{F})$$

Proof. Since the PYDT defines an infinitely exchangeable process on data points, the result follows directly by de Finetti's Theorem (Hewitt and Savage, 1955). \square

Another way of expressing Corollary 1 is that data points x_1, \dots, x_N sampled from the PYDT could equivalently have been sampled by first sampling a probability measure $\mathcal{F} \sim \nu$, then sampling $x_i \sim \mathcal{F}$ iid for all i in $\{1, \dots, N\}$. For divergence functions such that $A(1)$ is infinite, the probability measure \mathcal{F} is continuous almost surely.

Lemma 2. *The PYDT reduces to the Diffusion Dirichlet Tree (Neal, 2001) in the case $\alpha = \beta = 0$.*

Proof. This is clear from the generative process: for $\alpha = \beta = 0$ there is zero probability of branching at a previous branch point (assuming continuous cumulative divergence function $A(t)$). The probability of diverging in the time interval $[t, t + dt]$ from a branch previously traversed by m datapoints becomes:

$$\frac{a(t)\Gamma(m-0)dt}{\Gamma(m+1+0)} = \frac{a(t)(m-1)!dt}{m!} = \frac{a(t)dt}{m} \quad (11)$$

as for the DDT. \square

It is straightforward to confirm that the DDT probability factors are recovered when $\alpha = \beta = 0$. In this case $K = 2$ since non-binary branch points have zero probability, so Equation 7 reduces as follows:

$$\frac{a(t_b) \prod_{l=1}^{K=2} \Gamma(n_l^b - 0)}{\Gamma(m(b) + 0)} = \frac{a(t_b)(b_1 - 1)!(b_2 - 1)!}{(m(b) - 1)!}$$

as for the DDT. Equation 9 also reduces to the DDT expression since

$$H_n^{0,0} = \sum_{i=1}^n \frac{\Gamma(i-0)}{\Gamma(i+1+0)} = \sum_{i=1}^n \frac{(i-1)!}{i!} = \sum_{i=1}^n \frac{1}{i} = H_n$$

where H_n is the n -th Harmonic number.

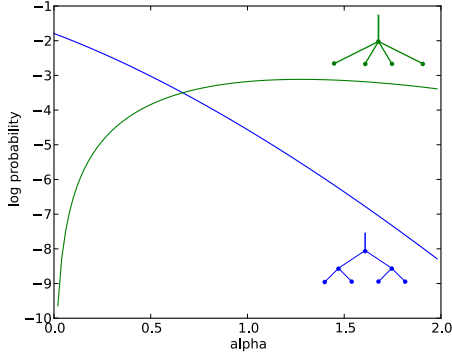


Figure 2: The effect of varying α on the log probability of two tree structures, indicating the types of tree preferred. Small $\alpha < 1$ favors binary trees while larger values of α favors higher order branching points.

For the purpose of this paper we use the divergence function $a(t) = \frac{c}{1-t}$, with “smoothness” parameter $c > 0$. Larger values $c > 1$ give smoother densities because divergences typically occur earlier, resulting in less dependence between the datapoints. Smaller values $c < 1$ give rougher more “clumpy” densities with more local structure since divergence typically occurs later, closer to $t = 1$. For this divergence function we have $A(t) = -c \log(1-t)$.

Equation 9 factorizes into a term for t_a and t_b . Collecting such terms from the branches attached to an internal node b the factor for t_b for the divergence function $a(t) = c/(1-t)$ is

$$P(t_b|\mathcal{T}) = a(t_b) \exp \left[A(t_b) \left(\sum_{k=1}^{K_b} H_{n_k^b-1}^{\alpha,\beta} - H_{m^{(b)}-1}^{\alpha,\beta} \right) \right] = c(1-t_b)^{cJ_{n^b}^{\alpha,\beta}-1} \quad (12)$$

where $J_{n^b}^{\alpha,\beta} = H_{\sum_{k=1}^K n_k^b-1}^{\alpha,\beta} - \sum_{k=1}^K H_{n_k^b-1}^{\alpha,\beta}$ with $\mathbf{n}^b \in \mathbb{N}^K$.

This generalization of the DDT allows non-binary tree structures to be learnt. By varying α we can move between flat (large α) and hierarchical clusterings (small α), as shown in Figure 2.

4 Model

To complete the model we must specify a likelihood function for the data given the leaf locations of the PYDT, and priors on the hyperparameters. We use a Gaussian observation model for multivariate continuous data and a probit model for binary vectors. We specify the following priors on the hyperparameters:

$$\begin{aligned} \alpha &\sim G(a_\alpha, b_\alpha) & \beta &\sim \text{Beta}(a_\beta, b_\beta) \\ c &\sim G(a_c, b_c) & 1/\sigma^2 &\sim G(a_{\sigma^2}, b_{\sigma^2}) \end{aligned}$$

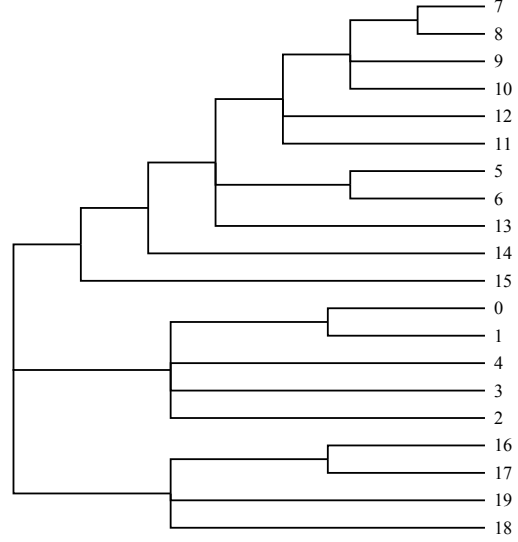
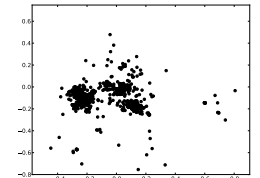
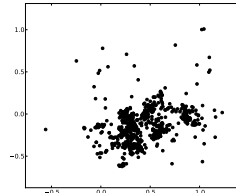
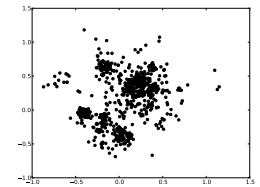
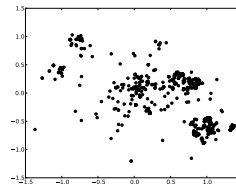


Figure 3: A sample from the Pitman-Yor Diffusion Tree with $N = 20$ datapoints and $a(t) = 1/(1-t)$, $\alpha = 1, \beta = 0$ showing the branching structure including non-binary branch points.



(a) $c = 1, \alpha = 0, \beta = 0$ (b) $c = 1, \alpha = 0.5, \beta = 0$ (DDT)



(c) $c = 1, \alpha = 1, \beta = 0$ (d) $c = 3, \alpha = 1.5, \beta = 0$

Figure 4: Samples from the Pitman-Yor Diffusion Tree with $N = 1000$ datapoints in $D = 2$ dimensions and $a(t) = c/(1-t)$. As α increases more obvious clusters appear.

where $G(a, b)$ is a Gamma distribution with shape a and rate b . In all experiments we used $a_\alpha = 2, b_\alpha = .5, a_\beta = 1, b_\beta = 1, a_c = 1, b_c = 1, a_{\sigma^2} = 1, b_{\sigma^2} = 1$.

5 Inference

We propose two inference algorithms: an MCMC sampler and a more computationally efficient greedy EM algorithm. Both algorithms marginalize out the locations of internal nodes using belief propagation, and are capable of learning the hyperparameters c, σ^2, α and β if desired.

5.1 MCMC sampler

We construct an MCMC sampler to explore the posterior over the tree structure, divergence times and hyperparameters. To sample the structure and divergence times a subtree is chosen uniformly at random to be detached (the subtree may be a single leaf node). To propose a new position in the tree for the subtree, we follow the procedure for generating a new sample on the remaining tree. The subtree is attached wherever divergence occurred, which may be on a segment, in which case a new parent node is created, or at an existing internal node, in which case the subtree becomes a child of that node. If divergence occurred at a time later than the divergence time of the root of the subtree we must repeat the procedure until this is not the case. The marginal likelihood of the new tree is calculated, marginalizing over the internal node locations, and excluding the structure and divergence time contribution since this is accounted for by having sampled the new location according to the prior. The ratio to the marginal likelihood for the original tree gives the Metropolis factor used to determine whether this move is accepted. Unfortunately it is not possible to slice sample the position of the subtree as in Neal (2003b) because of the atoms in the prior at each branch point.

Smoothness hyperparameter c . From Equation 12 the Gibbs conditional for c is

$$G\left(a_c + |\mathcal{I}|, b_c + \sum_{i \in \mathcal{I}} J_{n_i}^{\alpha, \beta} \log(1 - t_i)\right) \quad (13)$$

where \mathcal{I} is the set of internal nodes of the tree.

Data variance σ^2 . It is straightforward to sample $1/\sigma^2$ given divergence locations. Having performed belief propagation it is easy to jointly sample the divergence locations using a pass of backwards sampling. From Equation 10 the Gibbs conditional for the preci-

sion $1/\sigma^2$ is then

$$G(a_{\sigma^2}, b_{\sigma^2}) \prod_{[ab] \in \mathcal{S}(\mathcal{T})} G\left(D/2 + 1, \frac{\|x_a - x_b\|^2}{2(t_b - t_a)}\right) \quad (14)$$

where $\|\cdot\|$ denotes Euclidean distance.

Pitman-Yor hyperparameters α, β . We use slice sampling (Neal, 2003a) to sample α and β . We reparameterize in terms of the logarithm of α and the logit of β to extend the domain to the whole real line. The terms required to calculate the conditional probability are those in Equations 7 and 9.

5.2 Greedy Bayesian EM algorithm

As an alternative to MCMC here we use a Bayesian EM algorithm to approximate the marginal likelihood for a given tree structure, which is then used to drive a greedy search over tree structures, following our work in Knowles et al. (2011).

EM algorithm. In the E-step, we use message passing to integrate over the locations and hyperparameters. In the M-step we maximize the lower bound on the marginal likelihood with respect to the divergence times. For each node i with divergence time t_i we have the constraints $t_p < t_i < \min(t_l, t_r)$ where t_l, t_r, t_p are the divergence times of the left child, right child and parent of i respectively.

We jointly optimize the divergence times using LBFGS (Liu and Nocedal, 1989). Since the divergence times must lie within $[0, 1]$ we use the reparameterization $s_i = \log[t_i/(1 - t_i)]$ to extend the domain to the real line, which we find improves empirical performance. From Equations 10 and 12 the lower bound on the log evidence is a sum over all branches $[pi]$ of expressions of the form:

$$\langle c \rangle J_{n_i}^{\alpha, \beta} - 1 \log(1 - t_i) - \frac{D}{2} \log(t_i - t_p) - \langle \frac{1}{\sigma^2} \rangle \frac{b_{[pi]}}{t_i - t_p} \quad (15)$$

where $b_{[pi]} = \frac{1}{2} \sum_{d=1}^D \mathbb{E}[(x_{di} - x_{dp})^2]$, x_{di} is the location of node i in dimension d , and p is the parent of node i . The full lower bound is the sum of such terms over all nodes. The expectation required for $b_{[pi]}$ is readily calculated from the marginals of the locations after message passing. Differentiating to obtain the gradient with respect to t_i is straightforward so we omit the details. Although this is a constrained optimization problem (branch lengths cannot be negative) it is not necessary to use the log barrier method because the $1/(t_i - t_p)$ terms in the objective implicitly enforce the constraints.

Hyperparameters. We use variational inference to learn Gamma posteriors on the inverse data variance $1/\sigma^2$ and smoothness c . The variational updates for c and $1/\sigma^2$ are the same as the conditional Gibbs distributions in Equations 13 and 14 respectively. We optimize α and β by coordinate descent using golden section search on the terms in Equations 7 and 9.

Search over tree structures The EM algorithm approximates the marginal likelihood for a fixed tree structure \mathcal{T} . We maintain a list of K -best trees (typically $K = 10$) which we find gives good empirical performance. Similarly to the sampler, we search the space of tree structures by detaching and re-attaching subtrees. We choose which subtree to detach at random. We can significantly improve on re-attaching at random by calculating the local contribution to the evidence that would be made by attaching the root of the subtree to the midpoint of each possible branch and at each possible branch point. We then run EM on just the three best resulting trees. We found construction of the initial tree by sequential attachment of the data points using this method to give very good initializations.

5.3 Predictive distribution

To calculate the predictive distribution for a specific tree we compute the distribution for a new data point conditioned on the posterior location and divergence time marginals. Firstly, we calculate the probability of diverging from each branch according to the data generating process described in Section 1. Secondly we draw several (typically three) samples of when divergence from each branch occurs. Finally we calculate the Gaussian at the leaves resulting from Brownian motion starting at the sampled divergence time and location up to to $t = 1$. This results in a predictive distribution represented as a weighted mixture of Gaussians. Finally we average the density from a number of samples from the sampler or the K -best trees found by the EM search algorithm.

5.4 Likelihood models

Connecting our PYDT module to different likelihood models is straightforward: we use a Gaussian observation model and a probit model for binary vectors. The MCMC algorithm slice samples auxiliary variables and the EM algorithm uses EP (Minka, 2001) on the probit factor, implemented using the runtime component of the Infer.NET framework Minka et al. (2010).

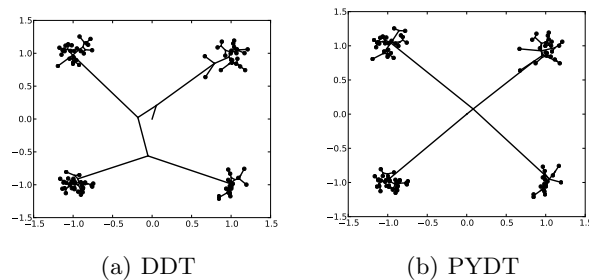


Figure 5: Optimal trees learnt by the greedy EM algorithm for the DDT and PYDT on a synthetic dataset with $D = 2, N = 100$.

6 Results

We present results on synthetic and real world data, both continuous and binary.

6.1 Synthetic data

We first compare the PYDT to the DDT on a simple synthetic dataset with $D = 2, N = 100$, sampled from the density

$$f(x, y) = \frac{1}{4} \sum_{\bar{x} \in [-1, 1]} \sum_{\bar{y} \in [-1, 1]} N(x; \bar{x}, 1/8) N(y; \bar{y}, 1/8)$$

The optimal trees learnt by 100 iterations of the greedy EM algorithm are shown in Figure 5. While the DDT is forced to arbitrarily choose a binary branching structure over the four equi-distant clusters, the PYDT is able to represent the more parsimonious solution that the four clusters are equally dependent. Both models find the fine detail of the individual cluster samples which may be undesirable; investigating whether learning a noise model for the observations alleviates this problem is a subject of future work.

6.2 Density modeling

In Adams et al. (2008) the DDT was shown to be an excellent density model on a $D = 10, N = 228$ dataset of macaque skull measurements, outperforming a kernel density and infinite mixture of Gaussians, and sometimes the Gaussian process density sampler itself. We compare the PYDT to the DDT on the same dataset, using the same data preprocessing and same three train test splits ($N_{\text{train}} = 200, N_{\text{test}} = 28$) as Adams et al. (2008). The performance using the MCMC sampler is shown in Figure 6. The PYDT finds trees with higher marginal likelihood than the DDT, which corresponds to a moderate improvement in predictive performance. Inference in the PYDT is actually slightly more efficient computationally than

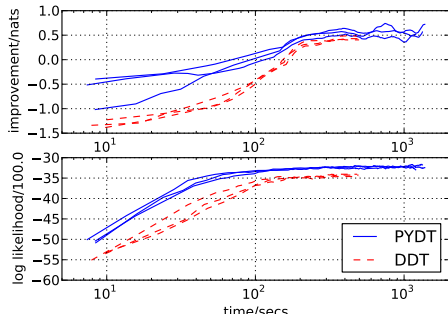


Figure 6: Density modeling of the $D = 10, N = 200$ macaque skull measurement dataset of Adams et al. (2008). *Top*: Improvement in test predictive likelihood compared to a kernel density estimate. *Bottom*: Marginal likelihood of current tree. The shared x-axis is computation time in seconds.

in the DDT because the on average smaller number of internal nodes reduces the cost of belief propagation over the divergence locations, which is the bottleneck of the algorithm.

6.3 Binary example

To demonstrate the use of an alternative observation model we use a probit observation model in each dimension to model 102-dimensional binary feature vectors relating to attributes (e.g. being warm-blooded, having two legs) of 33 animal species from Kemp and Tenenbaum (2008). The MAP tree structure learnt using EM, is shown in Figure 7, is intuitive, with subtrees corresponding to land mammals, aquatic mammals, reptiles, birds, and insects (shown by colour coding). Note that penguins cluster with aquatic species rather than birds, which is not surprising since there are attributes such as “swims”, “flies” and “lives in water”.

7 Conclusion

We have introduced the Pitman-Yor Diffusion Tree, a Bayesian nonparametric prior over tree structures with arbitrary branching structure at each branch point. We have shown the PYDT defines an infinitely exchangeable distribution over data points. We demonstrated an MCMC sampler and Bayesian EM with greedy search, both using message passing on the tree structure. More advanced MCMC methods could be of use here. Quantitatively we have shown a modest improvement relative to the DDT on a density estimation task. However, we see improved interpretability as the key benefit of removing the restriction to binary trees, especially since hierarchical clustering is typically used as a data exploration tool. Qualitatively,

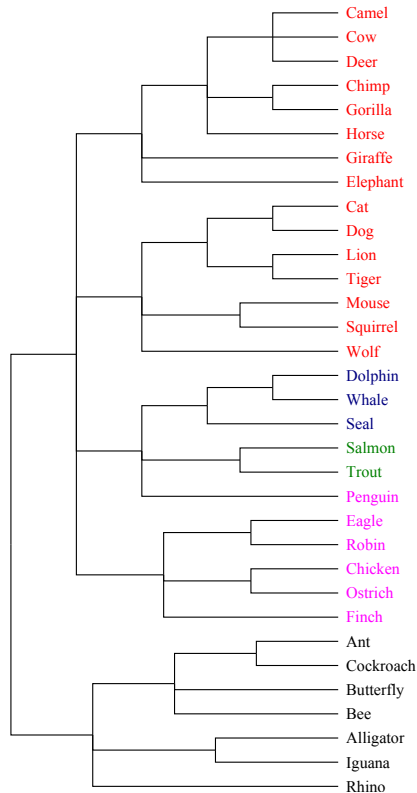


Figure 7: Tree structure learnt for the animals dataset of Kemp and Tenenbaum (2008).

we have shown the PYDT can find simpler, more interpretable representations of data than the DDT. To encourage the use of the PYDT by the community we will make our code publicly available.

In ongoing work we use the PYDT to learn hierarchical structure over latent variables in models including Hidden Markov Models, specifically in part of speech tagging (Kupiec, 1992) where a hierarchy over the latent states aids interpretability, and Latent Dirichlet Allocation, where it is intuitive that topics might be hierarchically clustered (Blei et al., 2004). Another interesting direction would be to use the prior over branching structures implied by the PYDT in the annotated hierarchies model of Roy et al. (2007).

A Probability of tree structure

For segment $[ab]$, recall that i is the index of the sample which created the branch point at b . Thus $i - 1$ samples did not diverge at b so do not contribute any terms. Let the final number of branches from b be K_b , and the number of samples which followed each branch be $\mathbf{n}^k := \{n_k^b : k \in [1 \dots K_b]\}$. The probability of the i -th sample having diverged to form the branch point is $\frac{a(t_b)\Gamma(i-1-\beta)}{\Gamma(i+\alpha)}$. Now we wish to calculate the probability of the final branching structure at b . Following the divergence of sample i there are $K_b - 2$ samples who

form new branches from the same point, contributing $\alpha + (k-1)\beta$ to the numerator for $k \in \{3, \dots, K_b\}$. Let c_l be the number of samples having previously followed path l , so that c_l ranges from 1 to $n_l^b - 1$ (apart from c_1 which only ranges from $i-1$ to $n_1^b - 1$). The j -th sample contributes a factor $j-1+\alpha$ to the denominator. The total number of datapoints which traversed edge $[ab]$ is $m(b) = \sum_{j=1}^{K_b} n_k^b$. The factor associated with this branch point is then:

$$\begin{aligned} & \frac{\prod_{k=3}^{K_b} [\alpha + (k-1)\beta] \prod_{c_1=i-1}^{n_1^b-1} (c_1 - \beta) \prod_{l=2}^{K_b} \prod_{c_l=1}^{n_l^b-1} (c_l - \beta)}{\prod_{j=i+1}^{m(b)} (j-1+\alpha)} \\ &= \frac{\prod_{k=3}^{K_b} [\alpha + (k-1)\beta] \prod_{c_1=1}^{K_b} \prod_{c_l=1}^{n_l^b-1} (c_l - \beta)}{\prod_{j=i+1}^{m(b)} (j-1+\alpha) \prod_{c_1=1}^i (c_1 - \beta)} \\ &= \frac{\prod_{k=3}^{K_b} [\alpha + (k-1)\beta] \Gamma(i+\alpha) \prod_{l=1}^{K_b} \Gamma(n_l^b - \beta)}{\Gamma(m(b) + \alpha) \Gamma(i-1 + \beta)} \end{aligned}$$

References

- Adams, R., Murray, I., and MacKay, D. (2008). The Gaussian process density sampler. In *Advances in Neural Information Processing Systems*, volume 21. MIT Press.
- Adams, R. P., Ghahramani, Z., and Jordan, M. I. (2010). Tree-structured stick breaking for hierarchical data. In *Advances in Neural Information Processing (NIPS) 23*.
- Aldous, D. (1985). Exchangeability and related topics. In *Ecole d'Été de Probabilités de Saint-Flour*, volume XIII, pages 1–198.
- Blei, D. M., Griffiths, T. L., and Jordan, M. I. (2010). The nested Chinese restaurant process and Bayesian nonparametric inference of topic hierarchies. *Journal of the ACM*, 57.
- Blei, D. M., Griffiths, T. L., Jordan, M. I., and Tenenbaum, J. B. (2004). Hierarchical topic models and the nested Chinese restaurant process. *Advances in Neural Information Processing Systems*, 16:106.
- Blundell, C., Teh, Y. W., and Heller, K. A. (2010). Bayesian rose trees. In *Proceedings of the International Conference on Uncertainty in Artificial Intelligence*.
- Duda, R. O., Hart, P. E., and Stork, D. G. (2001). *Pattern Classification*. Wiley-Interscience, 2nd edition.
- Heller, K. A. and Ghahramani, Z. (2005). Bayesian hierarchical clustering. In *Proceedings of the 22nd International Conference on Machine Learning*, page 304.
- Hewitt, E. and Savage, L. J. (1955). Symmetric measures on Cartesian products. *Transactions of the American Mathematical Society*, 80:470–501.
- Kemp, C. and Tenenbaum, J. B. (2008). The discovery of structural form. In *Proceedings of the National Academy of Sciences*, volume 105(31), pages 10687–10692.
- Knowles, D. A., Gael, J. V., and Ghahramani, Z. (2011). Message passing algorithms for Dirichlet diffusion trees. In *Proceedings of the 28th Annual International Conference on Machine Learning*.
- Kupiec, J. (1992). Robust part-of-speech tagging using a hidden markov model. *Computer Speech & Language*, 6(3):225 – 242.
- Liu, D. C. and Nocedal, J. (1989). On the limited memory BFGS method for large scale optimization. *Mathematical Programming*, 45:503–528.
- McCullagh, P., Pitman, J., and Winkel, M. (2008). Gibbs fragmentation trees. *Bernoulli*, 14(4):988–1002.
- Minka, T. P. (2001). Expectation propagation for approximate bayesian inference. In *Uncertainty in Artificial Intelligence*, volume 17.
- Minka, T. P., Winn, J. M., Guiver, J. P., and Knowles, D. A. (2010). Infer.NET 2.4. Microsoft Research Cambridge. <http://research.microsoft.com/infernet>.
- Neal, R. (2003a). Slice sampling. *The annals of statistics*, 31(3):705–741.
- Neal, R. M. (2001). Defining priors for distributions using Dirichlet diffusion trees. Technical Report 0104, Dept. of Statistics, University of Toronto.
- Neal, R. M. (2003b). Density modeling and clustering using Dirichlet diffusion trees. *Bayesian Statistics*, 7:619–629.
- Rai, P. and Daumé III, H. (2008). The infinite hierarchical factor regression model. In *Advances in Neural Information Processing Systems*, volume 21. MIT Press.
- Roy, D., Kemp, C., Mansinghka, V., and Tenenbaum, J. (2007). Learning annotated hierarchies from relational data. *Advances in Neural Information Processing Systems*, 19:1185.
- Teh, Y. W. (2006). A hierarchical Bayesian language model based on Pitman-Yor processes. In *Proceedings of the 21st International Conference on Computational Linguistics*, page 992. Association for Computational Linguistics.
- Teh, Y. W., Daumé III, H., and Roy, D. M. (2008). Bayesian agglomerative clustering with coascents. *Advances in Neural Information Processing Systems*, 20.
- Williams, C. (2000). A MCMC approach to hierarchical mixture modelling. *Advances in Neural Information Processing Systems*, 13.