Bayesian Nonparametric Regression with a Dirichlet Process Mixture of Generalized Linear Models

Lawrence W. Manning

May 11, 2010
Contents

1 Introduction 2

2 Dirichlet Process Mixtures of Generalized Linear Models 5
  2.1 Dirichlet Process Mixture Models 5
  2.2 Formulation of the DP-GLM Model 8
  2.3 DP-GLM Regression 9

3 Implementation: Practical Adjustments 11
  3.1 Conjugate Prior Distributions 11
  3.2 Hyperparameter Selection 14

4 Implementation: Stepping into the Algorithm 14
  4.1 Data Input & Formatting 15
  4.2 The Cluster 16
  4.3 The Main Loop 16
    4.3.1 Gibbs Sampling 16
    4.3.2 Prediction 17
  4.4 The Algorithm in Review 18

5 Extensions & Future Work 19
  5.1 Dynamic Prediction 19
  5.2 Coping with Missing Fields 20
  5.3 Future Work 20
1 Introduction

The algorithm explored within this paper is a Bayesian nonparametric solution to the general regression problem, which models a response variable $Y$ as dependent on a set of $d$-dimensional covariates $X$:

$$Y \mid X \sim f(m(X)).$$

(1)

Here, $m$ is a deterministic function that specifies the mean of the response $Y$ conditioned on the covariates $X$, and $f$ is a distribution that describes the deviation from the mean (what some might refer to as the “noise”). Both the mean function and the parameters of the noise distribution may be estimated from the set of paired covariate-response observations $\{(x_i, y_i)\}_{i=1}^n$. Given a new covariate observation, $x_{n+1}$, the response $y_{n+1}$ can be predicted by the conditional expectation $E[Y \mid x_{n+1}]$. For Bayesian regression models such as the one addressed here, one seeks to compute the posterior expectation conditioned on the observed data.

In both Bayesian and frequentist statistics, the most frequently used method of regression is the linear model. It assumes that the response distribution is Gaussian with fixed variance and that the mean of the response variable is a linear combination of the covariates. Linear regression seeks to estimate both the linear coefficients, which are usually denoted by $\vec{\beta}$, and the response variance. The strict assumptions imposed by this model, however, limit the use of both Bayesian linear regression and ordinary least squares (OLS) regression for an exceptionally rich class of non-linear problems.
Generalized linear models (GLMs) extend linear regression to a wider set of response types and distributions. GLMs are characterized by a link-function, which may be non-linear, and a response distribution that may be any member of the exponential family. A wide range of parametric models may be expressed as GLMs, including linear, logistic, multinomial, and Poisson regressions. The form of distributions in the exponential family is:

\[ f(y|\eta) = \exp\left(\frac{y \eta - b(\eta)}{a(\phi)} + c(y, \phi)\right), \quad \eta = g(\mu) \quad (2) \]

where \(a, b, \& c\) are form-specific functions that are well-known for various members of the exponential family, \(g\) is the link function, \(\mu\) is the mean of the response, and \(\phi\) is an arbitrary scale parameter. It may be shown that the function \(b\) has the property that \(b'(\eta) = \mu\). Using that fact and setting the covariates to be a linear predictor of the canonical parameter,

\( \eta = X\beta, \)

one may see that

\( b'(\eta) = g^{-1}(X\beta). \quad (3) \)

Equation 3 links the canonical parameter to the linear predictor and holds for all GLMs, defining the entire family of diverse, versatile, and computationally tractable parametric regression models. However, GLMs are still restricted by their dependence upon their parameters (the linear coefficients), which are situated in finite-dimensional space and therefore have a finite capacity for adaptation to the data. The model is assumed to apply uniformly
across all possible covariates, which will often not be the case for realistic data. In contrast, nonparametric methods operate in infinite dimensional space and can adapt well to data sets that stymie parametric competitors. This increased adaptability comes with the downside that it may over-fit the data. Nonparametric methods are also usually characterized by increased computational complexity, which often reduces the feasibility of using such models on high dimensional data.

Dirichlet process mixtures of generalized linear models (DP-GLMs) are Bayesian nonparametric models that place individual data points into clusters, within each of which a GLM is fitted to the response and conditioned on the covariates. The Dirichlet mixing process, which is responsible for the clustering, produces an infinite clustering model: a given observation can be attributed to any of the clusters containing other observations or to a new cluster that is drawn from a hyperdistribution (also referred to as the base measure). This base measure places a probability measure on probability measures, essentially assigning a distribution to the parameters of the distributions used within each cluster. The DP-GLM regression algorithm then calculates a prediction based upon both a weighted sum of the predictions from each of the observed clusters and a term to account for the infinite number of yet-unseen clusters available through the base measure. The ability to generate an arbitrary number of new clusters makes the DP-GLM a powerful and flexible method for analyzing data with minimal prior assumptions concerning the form of the response.

The goal of this paper is to give a detailed overview of the DP-GLM regression algorithm. Section 2 presents the model formulation as given in
the paper by Hannah, Blei, and Powell [4], which proved the unbiasedness of the estimator. Section 3 discusses additional assumptions and adjustments made for the sake of computational efficiency and simplicity of use. Section 4 looks at a case-study implementation in detail. Finally, Section 5 surveys both useful extensions to the implementation and areas that hold potential for future work.

2 Dirichlet Process Mixtures of Generalized Linear Models

To formulate the DP-GLM algorithm, it is first necessary to review Dirichlet process mixture models to complement the previous discussion of generalized linear models.

2.1 Dirichlet Process Mixture Models

Dirichlet process mixture models represent the true density of a random variable as a mixture of parametric densities conditioned on the unobserved (multidimensional) parameter $\theta$, which for notational convenience will be partitioned such that $\theta = (\theta_x, \theta_y)$, corresponding to the two complementary subsets of $\theta$ that are respectively associated with the covariates $X$ and response $Y$. Because Dirichlet process mixtures are Bayesian models, the parameter $\theta$ is imbued with a prior distribution $P$. The marginal probability of an observation $(x, y)$ is given by the continuous mixture:

$$f_0(x, y) = \int_T f(x, y | \theta) P(d\theta),$$  \hspace{1cm} (4)
where $\mathcal{T}$ is the set of all possible parameters. Uncertainty about the prior density $P$ can be modeled with a *Dirichlet process* (DP), which is described in Ferguson (1973)[1]. If $P$ is drawn from a Dirichlet process with base measure $\mathcal{G}_0$ and positive scaling parameter $\alpha$, then for any finite partition $A_1, \ldots, A_k$ of $\mathcal{T}$,

$$(P(A_1), \ldots, P(A_k)) \sim \text{Dir}(\alpha \mathcal{G}_0(A_1), \ldots, \alpha \mathcal{G}_0(A_k)),$$

where $\text{Dir}(a_1, \ldots, a_k)$ represents the Dirichlet distribution with strictly positive parameters $(a_1, \ldots, a_k)$, and $\mathcal{G}_0$ is a measure on $\mathcal{T}$. The Dirichlet distribution is a conjugate prior to the multinomial distribution; that is, it gives the probability that the probabilities of $k$ discrete events are $p_i$ for $i = 1, \ldots, k$, given that each event has been observed $a_i - 1$ times. For a given $\vec{p}$ on the $(k-1)$-simplex of $k$-dimensional vectors whose elements sum to unity, the probability density function of the Dirichlet distribution with $k$-dimensional parameter $\vec{a}$ is

$$f(\vec{p}, \vec{a}) = \frac{1}{B(\vec{a})} \prod_{i=1}^{K} p_{i}^{a_{i}-1}. \quad (5)$$

Here, $B(\vec{a})$ is a normalizing constant calculated from the multinomial beta function:

$$B(\vec{a}) = \frac{\prod_{i=1}^{K} \Gamma(a_i)}{\Gamma \left( \sum_{i=1}^{K} a_i \right)}$$

The random variable $P = (P_1, \ldots, P_k)$ generated by the Dirichlet process above is a *random measure*. In accordance with its interpretation as a distribution over a finite partition of space with a histogram $\vec{a}$ of observations,
the expectation of $E[P_i]$ is simply the maximum likelihood estimator:

$$E[P_i] = \frac{a_i}{\sum_{i=1}^{n} a_i}$$

An analytically convenient property of measures drawn from Dirichlet processes is that they may be integrated out of the conditional distribution of $\Theta_n$ given $\theta_{1:(n-1)}$. The random variable $\Theta_n$ has a Pólya’s Urn distribution,

$$\Theta_n \mid \theta_{1:(n-1)} \sim \frac{1}{\alpha + n - 1} \sum_{i=1}^{n-1} \delta_{\theta_i} + \frac{\alpha}{\alpha + n - 1} G_0. \tag{6}$$

In Equation 6, $\delta_{\theta_i}$ represents an atom at $\theta_i$, a measure that assigns a probability mass of one to that point [1, 5]. The clustering effect of the Dirichlet process is intuitively clear from the conditional distribution of $\Theta_n$, as there is a strictly positive probability that each $\Theta_n$ will take on the value of a previous $\theta_i$ for $i = 1, \ldots, n - 1$. Otherwise, $\Theta_n$ will be drawn independently of $\theta_{1:(n-1)}$ from the base measure $G_0$. The parameter $\alpha$ determines how likely $\Theta_n$ is to be newly-drawn instead of assuming one of the previously observed values. Additionally, as $n$ approaches infinity, the conditional distribution of $\Theta_n$ converges almost surely and in mean to a limiting vector with a finite number of elements.

In turn, $\theta$ is a latent parameter to an observed datum $x$,

$$P \sim DP(\alpha G_0)$$

$$\Theta_i \sim P$$

$$x_i \mid \theta_i \sim f(\cdot \mid \theta_i). \tag{7}$$
Using Bayes’ Theorem, one can derive the posterior distribution of $\theta_{1:n}$ given $x_{1:n}$, which renders apparent the interpretation as an infinite clustering model. Observations are grouped by their common parameters, but the number of unique parameters is random and unknown, and newly examined data points can cause the generation of new clusters. [4]

2.2 Formulation of the DP-GLM Model

The underlying assumptions of the DP-GLM are that the covariates $X$ may be modeled with some mixture of exponential family distributions, that the response $Y$ may be modeled as a GLM conditioned on the covariates $X$, and that with each component of the mixture there may be associated a set of GLM coefficients. Partitioning $\theta = (\theta_x, \theta_y)$ as before and letting $G_0$ denote a base measure on the space of both $X$ & $Y$, the full model is:

\[
P \sim DP(\alpha G_0),
\]

\[
(\theta_{x,i}, \theta_{y,i}) \sim P,
\]

\[
X_i \mid \theta_{x,i} \sim f_x(\cdot \mid \theta_{x,i}), \tag{8}
\]

\[
Y_i \mid x_i, \theta_{y,i} \sim GLM(\cdot \mid x_i, \theta_{y,i}). \tag{9}
\]

The random measure $P$ is generated from a Dirichlet process and in turn $\theta$ is distributed according to $P$, as in Section 2.1. $f_x$ specifies the distribution of the covariates and is, in turn, specified by $\theta_x$. The GLM for $Y$ is dependent upon the parameter $\theta_y$ and further varies by the form of the response (i.e., continuous, categorical, etc.) and the link function. Equations 8 and 9 are taken over $i = 1, \ldots, n$, though there generally will not be $n$ distinct
submodels due to the clustering effect of the Dirichlet process.

When covariate-response pairs \( \{(x_i, y_i)\}_{i=1}^n \) may be fully observed, the Dirichlet process clusters covariates that share similar relationships to their responses. During that phase, the model is considered to be “in training.” If only the covariates are observed, the point may still be assigned to a cluster based upon its covariates, and a prediction of the response may be made from the GLM associated with that cluster.

### 2.3 DP-GLM Regression

Thus, given a set of covariate-response pairs \( \{(x_i, y_i)\}_{i=1}^n \), the DP-GLM model can be used to predict the response for a new set of covariates \( x \).

Conditional to the latent parameters \( \theta_{1:n} \) that generated the observed data, the expectation of the response is:

\[
E[Y \mid x, \theta_{1:n}] = \frac{\alpha \int_T E[Y \mid x, \theta] f_x(x \mid \theta) G_0(d\theta) + \sum_{i=1}^n E[Y \mid x, \theta_i] f_x(x \mid \theta_i)}{\alpha \int_T f_x(x \mid \theta) G_0(d\theta) + \sum_{i=1}^n f_x(x \mid \theta_i)}
\]  

(10)

Hannah et al. [4, 3] suggest that this may be approximated with an average of \( M \) Monte Carlo samples of the expectation conditioned on \( \theta_{1:n} \),

\[
E[Y \mid x, (X_i, Y_i)_{1:n}] = E[Y \mid x, \theta_{1:n}]
\]

\[
\approx \frac{1}{M} \sum_{m=1}^M E[Y \mid x, \theta^{(m)}_{1:n}]
\]  

(11)

The computation in Equation 11 depends entirely on the ability to generate posterior draws of the hidden parameters \( \theta_{1:n} \). To accomplish this end, one may use Markov Chain Monte Carlo (MCMC) simulation [6].
In particular, a special case of the Hastings-Metropolis algorithm known as the Gibbs Sampler offers a computationally feasible method for sampling from an unknown joint distribution by using known conditional distributions. The Gibbs Sampler sequentially uses the conditional distributions of the components of its state vector to generate a Markov chain with the desired joint distribution as its ergodic distribution. For example, given the state vector \((\phi_1, \ldots, \phi_n)\), one iteration would draw \(\phi_i\) and hold \((\phi_1, \ldots, \phi_{i-1}, \phi_{i+1}, \ldots, \phi_n)\) constant. The subsequent iteration might do the same for \(\phi_{i+1}\), and so on ad infinitum [7].

In this case, the state vector is the vector of hidden parameters \((\theta_x)_{1:n}\).

The general form of the conditional distribution was given in Equation 6 in Section 2.1. Recall that this conditional distribution is determined by the hyperparameter \(\alpha\) and the base measure \(G_0\). The regression procedure is given in Algorithm 1.

**Algorithm 1 DP-GLM Regression**

*Data:* Observations \((X_i, Y_i)_{1:n}\), functions \(f_x, f_y\), number of posterior samples \(M\), query \(x\)

*Result:* Mean function estimate at \(x\), \(\bar{m}(x)\)

1. Initialization;
2. for \(m = 1\) to \(M\) do
   1. Obtain posterior sample \(\theta^{(m)}_{1:n} \mid (X_j, Y_j)_{1:n}\) through Gibbs Sampling;
   2. Compute \(\mathbb{E}[Y \mid x, \theta^{(m)}_{1:n}]\);
3. end for
4. Set \(\bar{m}(x) = \frac{1}{M} \sum_{m=1}^{M} \mathbb{E}[Y \mid x, \theta^{(m)}_{1:n}]\);

Two difficulties arise from the use of a Markov chain for the generation of posterior samples: first, successive samples are not independent; second, an indeterminate number of iterations are required for the Markov chain
to converge to the ergodic distribution. The first issue may be resolved by pulsed sampling; that is, one may simply take a sample after every period of some fixed number of iterations. The second is best handled by choosing a “good” starting estimate (presumably using a priori information) and then discarding the initial iterations as part of some predetermined burn-in phase. However, convergence is only guaranteed in the limit as the number of iterations approaches infinity [4, 7]. Thus one may observe inconsistencies between runs using the same data but random number generators with different seeds, even for large burn-in and pulse selections. This instability is an apparent shortcoming of the algorithm.

3 Implementation: Practical Adjustments

The formulation of the DP-GLM given in Section 2 offers a great degree of generality (and in some ways, ambiguity) in certain aspects of practical implementation. This section attempts to address some of those decisions.

3.1 Conjugate Prior Distributions

Equation 10 in Section 2.3 reveals a computationally important aspect in the selection of $G_0$. It includes the term

$$\alpha \int_T f_x(x \mid \theta)G_0(d\theta),$$

which may be computed analytically if the base measure of $\theta_x$ is conjugate to $f_x(\cdot \mid \theta)$. Otherwise the integration must be performed numerically,
which is considerably less efficient and computationally infeasible for high dimensional spaces.

A conjugate prior also gives a closed form solution for the conditional distribution used in the Gibbs Sampler, making that subroutine more efficient as well. Because of the significant computational gains, it is in one’s interest to assume such a prior when implementing the DP-GLM.

For the continuous components of \( \theta_x \), it is convenient to assume a Gaussian posterior model with a Normal-Inverse Chi-Squared (or equivalently, a Normal-Inverse Gamma) prior. For purely notational convenience, this paper will use the former. A random variable \( X_1 \) distributed according to a Normal-Inverse Chi-Squared prior is specified as follows:

\[
\sigma^2 \sim \text{Inv-Chi}^2(\nu_0, \sigma_0^2) \\
\mu \mid \sigma^2, \mu_0, \kappa_0 \sim \mathcal{N}\left(\mu_0, \frac{\sigma^2}{\kappa_0}\right) \\
X_1 \sim \mathcal{N}(\mu, \sigma)
\]

Assuming that the data has been normalized (for each covariate component) to have a mean of zero and variance of one, a suitable set of prior parameters is \( \{\mu_0, \sigma_0^2, \nu_0, \kappa_0\} = \{0, 1, 1, 2\} \). Given a set of observations \( x_{1:n} \) with sample mean \( \bar{x} \) and sample variance \( s^2 \), the Bayesian updating equations [2] for
these parameters are:

\[ \mu_n = \frac{\kappa_0}{\kappa_0 + n} \mu_0 + \frac{n}{\kappa_0 + n} \bar{x} \]

\[ \kappa_n = \kappa_0 + n \]

\[ \nu_n = \nu_0 + n \]

\[ \nu_n \sigma_n^2 = \nu_0 \sigma_0^2 + (n - 1)s^2 + \frac{\kappa_0 n}{\kappa_0 + n} (\bar{x} - \mu_0)^2 \]

Similarly, the multinomial model with a Dirichlet prior may be chosen for categorical inputs. Recall the Dirichlet distribution from Equation 5 in Section 2.1. A random variable \( X_2 \) with a Dirichlet prior parameterized by \( \vec{a}_0 \) may be written:

\[ X_2 \sim Dir(\vec{a}_0). \]

Given a vector representation \( \vec{x} \) of a histogram of \( n \) categorical observations, the posterior \[1\] is simply:

\[ \vec{a}_n = \vec{a}_0 + \vec{x} \]

\[ X_2 \sim Dir(\vec{a}_n) \]

For a general implementation, one may do well to select a relatively uninformed prior such as \( \vec{a} = \vec{1} \).

Under the assumption of these conjugate priors, the calculations of both the conditional distributions of each component and of the value of the base measure term in the prediction are made substantially faster, and for well-behaved and properly-formatted data, these assumptions are flexible enough.
to justify the sacrifice in generality.

3.2 Hyperparameter Selection

Though the hyperparameters of the base measure $G_0$ may be assigned statically, assuming proper data cleaning and normalization, the remaining hyperparameter $\alpha$ cannot be dealt with so easily. Recall that $\alpha$ defines the likelihood of a given data point being assigned to a previously observed cluster as opposed to one drawn from the base measure; the value of this parameter is, in general, problem dependent and therefore cannot be preset. The best available compromise, as suggested by Hannah et al. [4], is to draw $\alpha$ from a Gamma distribution and periodically resample.

The Gamma distribution is fitting because it produces a flexible range of strictly positive values with a specifiable mean. In the absence of a priori information, this seems to be a reasonable solution, though this issue is an interesting topic for future research. A good approach might be to impose a Bayesian prior (which might still be a Gamma distribution) and then view it as an online optimization problem.

4 Implementation: Stepping into the Algorithm

In this section, a robust implementation (written in Java) will be explored step-by-step in order to study the mechanics of the DP-GLM. The sample implementation supports both categorical and continuous covariates as well as a continuous, univariate response, which is modeled with simple Bayesian linear regression.
A notational comment: upper case names of implementation components refer specifically to Java Objects, rather than to the abstract constructs which they represent. That said, the object structure of the implementation being examined is designed to intuitively correspond to that of the original algorithm itself.

4.1 Data Input & Formatting

It is assumed that the client partitions the available data into training and testing data. The former is used to determine clustering, and the covariates of the latter are used to produce predictions of their respective responses. If the true response values are available, then these test points may also be used to calculate a whole range of error metrics.

To begin, one must properly normalize all continuous data to have a mean of zero and a variance of one. This is done independently for each continuous covariate, and the original means and variances are stored, as they are necessary for proper rescaling of predictions. Each observation is stored as a Point object, which is simply an internal representation that both allows each datum to be manipulated as a Java object and stores properties of each of its components (continuous, categorical, range, etc). All of the Points may then be added to the single, initial Cluster. Thus, on the first iteration, the DP-GLM regression algorithm reduces to mere Bayesian linear regression.
4.2 The Cluster

Each Cluster object holds a collection of Points that are believed to share the same hidden parameter $\theta$. It maintains an internal record of both the sufficient statistics and the parameters $\theta_x$ of the covariates of the Points in the Cluster. These parameters are calculated according to the updating equations given in Section 3.1. Additionally, each Cluster is responsible for calculating its own GLM parameters; thus each Cluster is entirely self-contained and self-descriptive. For efficiency, the Cluster maintains an internal flag signaling a change in the constituent points; the GLM coefficients are then recalculated when they are next needed.

4.3 The Main Loop

The main loop is set to run through 5000 iterations of the Gibbs Sampler, sampling every ten iterations ($\text{pulse} = 10$) after a burn-in phase of 2000 iterations. A Gamma(2,2) distribution is used for alpha and is resampled every 10 iterations as well.

4.3.1 Gibbs Sampling

At each iteration of the main loop, the algorithm completes a full cycle over all of the training points with the Gibbs Sampler. Each training point $x_i$ corresponds to a single $\theta_i$, though the collection $\{\theta_i\}_{i=1}^n$ need have only one unique element. (Indeed, on the first iteration, this is the case by design: all of the $\theta_i$’s are equal.) For each Point $x_i$, the algorithm calculates the probability of remaining in its current Cluster, moving into another existing
Cluster (not an option at the first iteration), and creating a new Cluster. These probabilities are calculated from the base measure, which was specified in Section 3.1, and Equation 6 in Section 2.1.

The decision amongst Clusters is, in actuality, a decision amongst retaining the current value of $\theta_i$, switching to another previously observed value, or drawing a new value altogether from the base measure. Keep in mind that the $\theta_i$ values being discussed are the true values. Any time a Point is reassigned, the estimators for the $\theta$’s in the leaving and entering Clusters must be recalculated. It is assumed, however, that the true value remains the same. In practice, though, it is often difficult to distinguish between Cluster objects with largely varying estimators and transient values of $\theta$.

### 4.3.2 Prediction

On each tenth iteration after the burn-in phase, a prediction of the response for each of the Points in the testing set is made based upon the current set $\{\theta_i\}_{i=1}^n$ (i.e., the current set of Cluster assignments and the derived parameters). This prediction is a weighted sum of the predictions from the GLM within each Cluster. The weightings are determined by the probability of being associated with each of the current Clusters, as well as to a hitherto uncreated Cluster. These probabilities all have closed form expressions derived, once again, from Equation 6 in Section 2.1 with a conjugate base measure.

That prediction is then added to a collection, from which a mean prediction is calculated for the given Point. This aggregate collection, $\{\bar{m}(x_j)\}_{j=1}^N$, where the $x_j$ represent the Points in the testing set, is an unbiased predictor.
and the sought-after solution to the regression problem [4].

4.4 The Algorithm in Review

The DP-GLM is reasonably straightforward in the end; the algorithm identifies points with similar response relationships, clusters them, and then runs a Bayesian linear regression through each cluster. It relies on the asymptotic convergence of the Markov Chain Monte Carlo simulation to distribute the Points into the proper Clusters. It then finds the mean of the weighted means of the GLMs from pulsed clustering samples to calculate the unbiased mean predictor. The implementation version of the algorithm is given in Algorithm 2.

Algorithm 2 DP-GLM Regression

Require: testing data, training data, number of iterations
 Normalize training data
 Normalize testing data
 for $i = 1$ to $iter$ do
   for $j = 1$ to $n$ do
     Remove the $j^{th}$ Point in training data from its Cluster
     Assign it to a new Cluster
     Recalculate parameters of modified Clusters
   end for
   if $i \% pulse == 0$ && $i >= burnin$ then
     Calculate probability of association with each existing Cluster
     Calculate probability of association with a newly-drawn Cluster
     Calculate prediction as probability-weighted sum of GLM predictions
   end if
end for
Calculate means of prediction samples
5 Extensions & Future Work

This section discusses extensions that may be implemented to enhance both the usability and the versatility of the DP-GLM algorithm, as well as potential areas for enhancement in the future.

5.1 Dynamic Prediction

A practical implementation assumes that a set of testing data (that is, data for which predictions will be calculated) will be available at initialization. If so, then the main loop may be run to completion at initialization and prediction samples may be taken on a pulsed basis per Algorithm 2. However, if a prediction is desired for points not in the original testing set, then the original DP-GLM regression algorithm would require a second round of MCMC, which is extremely inefficient, particularly in the case in which points are being queried one at a time.

The easiest solution is to store the sufficient statistics and parameters of the clusters in use at each pulse during the original run of the main loop. Then a prediction may be made by revisiting the sequence of historical states and taking a prediction sample based upon those records. This method avoids additional MCMC altogether, yielding speed increases of several orders of magnitude and providing internal consistency amongst predictions. While this does cause a substantial relative increase in the amount of memory expended on object storage, the absolute impact on initialization runtime is minimal. This modification then opens DP-GLM regression up for a variety of practical applications, with the foremost perhaps being
dynamic programming.

5.2 Coping with Missing Fields

Additionally, the DP-GLM regression algorithm can be modified to accept both training and testing data with incomplete fields. An implicit assumption of the model is that each component of the covariates $X$ (and therefore of $\theta_x$) is independent of all other components. Additionally, all probabilities are simply weightings determined by relative log-likelihoods, which are linearly additive functions of the components of $X$. If a field is missing from a given data point $x_i$, then the corresponding term in the log-likelihood calculation may simply be omitted. Since it is factored out of all of the log-likelihood values used for calculating a given set of probabilities, this does not bias or otherwise adversely affect the model. For the GLM calculation, one should use the maximum likelihood estimator of the value of that field, which is simply the sample mean of that component within the Cluster.

From the perspective of real-world applications, the ability to obtain an unbiased estimator on incomplete data is invaluable. Without such a modification, it would be impossible to handle realistic data sets in which measurement attempts might have failed or data collection was inconsistent.

5.3 Future Work

There are several aspects of the DP-GLM that might be interesting for further research. First, selection of the hyperparameter $\alpha$ could be made to adapt to the data instead of being generated randomly, as was mentioned in Section 3.2. Second, specifying an appropriate number of iterations, pulse
size, and burn-in length for the Gibbs sampler is difficult and problem dependent. More iterations gives a greater likelihood of convergence but comes at the cost of speed; a good strategy for continuing or halting MCMC after a certain threshold could help to improve the stability of the algorithm without unnecessarily compromising its efficiency. Finally, the robustness of the DP-GLM still needs to be tested for a wider variety of problem classes, particularly those with a response that is not continuous and univariate.
References


