Locus Regression

Allen W. Lynch

February 1, 2023

1 Introduction

The generative model explains the observed mutations in a dataset $\mathbb{D}$ of G genomes, where mutations in a genome are composed of a distribution over K signatures, or processes, each of which has a characteristic bias with respect to which genomic loci and nucleotides it affects. Mutations in genome g are expressed as tuples of (m, ℓ), where $m \in \{m \in \mathbb{Z} | 1 \leq m \leq 96\}$ is the trinucleotide context and the mutation (of which there are 96 possibilities) and $\ell \in \{\ell \in \mathbb{Z} | 1 \leq \ell \leq L\}$ is the genomic locus or window at which that mutation occurred, with L possible windows. In the generative process outlined below, $\pi_g \in \mathbb{R}^K_{(0,1)}, \sum_{k=1}^{K} \pi_{gk} = 1$ is the composition over processes that generated the mutations in a genome g, N_g is the number of mutations in g, and z is an indicator variable which describes which process generated the gi^{th} mutation:

Algorithm 1 Generative Process

```
for all $g \in 1...G$ do
    $\pi_g \sim \text{Dirichlet}(\alpha)$
    for all $i \in 1...N_g$ do
        $z \sim \text{Categorical}(\pi_g)$
        $\ell \sim \text{Categorical}(\theta_z^{(g)})$
        $m \sim \text{Categorical}(\psi_z^{(\ell)})$
        yield $(m, \ell)$
    end for
end for
```

Like the closely-related process for generating a textual corpus utilized by Latent Dirichlet Allocation (LDA), we sample the mutations in a genome by iteratively choosing a process (z), then a mutation (m, ℓ) from categorical distributions. Unlike LDA, however, the distribution over loci θ is conditioned on both process *and* sample effects - for a given process, the distribution over loci may be different across samples. For example, a process which affects nucleotides in heterochromatin will have a different distribution over loci for two samples with different epigenetic states. Learning a distribution over loci for each process in each sample by estimating θ for each z, g would have extremely high variance. Therefore, we estimate θ using a data efficient parameterization which describes how a process acts across loci and samples with respect to genomic correlates (see section **3**).

Likewise, the distribution over mutations, ψ, is conditioned on the process *and* the locus, since different genomic windows may have different distributions over trinucleotide contexts on which a process acts. Again, we must construct a data-efficient parameterization for this distribution (see section **4**).

2 Parameterizing θ

As outlined above, the distribution of mutations over loci for each process and sample is high-dimensional and impractical to compute directly. Instead, we paramterize this distribution in terms of a simpler model of process-dependent mutation rates. We assume that the probability of a mutation occurring at a locus given a process z and a sample g is proportional to the estimated mutation rate for that locus. We predict mutation rate using a linear model where loci are associated with genomic features $X^{(g)} \in \mathbb{R}^{F,L}$, where L is the number of loci and F is the number of features, and processes have coefficients $\beta_z \in \mathbb{R}^F$. Thus,

$$p(\ell|z, g) = (\theta_{z,g})_\ell \propto l_\ell \text{exp}(\sum_{f=1}^{F} \beta_{zf} X_{f\ell}^{(g)}) = l_\ell \text{exp}(\beta_z^T X_\ell^{(g)}) \tag{1}$$

Above, l_ℓ is the length in nucleotides for that window, but this fixed value can also represent user-specified "exposure" variables. For instance, l may adjust for technical effects which influence the sensitivity to call mutations within a window independent of the mutation rate.

Notably, the features X are dependent on the sample g, while β only depends on the process. In this way, this parameterization can capture jointly process and sample-specific variation in mutation rate across the genome using few parameters. To specify a valid probability distribution over ℓ, we must introduce some multiplicative normalizer to the mutation rate model. That normalizer is the sum of predicted mutation rates across all loci:

$$p(\ell|z, g) = \frac{1}{\sum_{\ell'=1}^{L} l_{\ell'} \text{exp}(\beta_z^T X_{\ell'}^{(g)})} l_\ell \text{exp}(\beta_z^T X_\ell^{(g)}) \tag{2}$$

Finally, to regularize the coefficients and address uncertainty in the posterior estimate of θ, we treat β as a random variable with prior:

$$\beta_{zf} \sim \text{Normal}(0, \tau_z^2) \tag{3}$$

We estimate the value of τ_z empirically during parameter inference.

3 Parameterizing ψ

The distribution ψ gives the probability of each mutation - defined by the trinucleotide context and alternative nucleotide - occuring given some process z and genomic window ℓ. We assume that the dependence on ℓ is solely the result of differences in available contexts to mutate within windows, and that the θ distribution fully explains all other locus-based effects.

We factorize the process of generating a single mutation into two components: a distributions over contexts, c, from the set $\mathbb{C}$ of trinucleotide contexts (of which there are 32) and a distribution over alternative nucleotides, a, conditioned on that context. The set of alternative nucleotides $\mathbb{A}_c$ is defined as $\{a \in \{\mathrm{A}, \mathrm{T}, \mathrm{C}, \mathrm{G}\} | a \neq c_{\mathrm{ref}}\}$ which excludes the middle nucleotide of the context (c_{ref}), since a nucleotide cannot mutate to itself. As an example, the probability of a mutation expressed in the COSMIC notation is given by:

$$P(m = \mathrm{A[C{:}T]G}|z, \ell) = P(a = \mathrm{T}|z, c = \mathrm{ACG})P(c = \mathrm{ACG}|z, \ell) \tag{4}$$

We assume that each mutational process acts on each nucleotide context at some fixed rate, $\lambda_{zc} \in \mathbb{R}_{(0,\mathrm{inf})}$, such that over some arbitrary timestep and acting in a locus ℓ with $t_{\ell c}$ such contexts, $\lambda_{zc} \cdot t_{\ell c}$ of them are mutated.

Assuming the reserve of contexts cannot be depleted, the portion of mutated contexts of some type is given by:

$$p(c|z, \ell) = \frac{\lambda_{zc} t_{\ell c}}{\sum_{c' \in \mathbb{C}} \lambda_{zc'} t^T_{c'\ell}} \tag{5}$$

Above, we cast the problem as inferring absolute rates, which is impossible from the data. If instead we treat $\lambda \in \mathbb{R}^{K \times |\mathbb{C}|}_{(0,1)}$ as a collection of Gamma-distributed random variables, and we infer relative rates:

$$p(c|z, \ell) = \left(\frac{\frac{1}{\sum_{c'=1}^{C} \lambda_{zc'}}}{\frac{1}{\sum_{c'=1}^{C} \lambda_{zc'}}} \right) \cdot \frac{\lambda_{zc} t_{\ell c}}{\sum_{c'=1}^{C} \lambda_{zc'} t^T_{c'\ell}} \tag{6}$$

Then the random variable $\frac{\lambda_{zc}}{\sum_{c'=1}^{C} \lambda_{zc'}} \forall c \in \{1...|\mathbb{C}|\}$ is Dirichlet-distributed under the condition the underlying Gamma distributions share the same rate variable. Therefore, we assume that λ_z is a compositional random variable with prior Dirichlet($\mathbf{1}$).

Finally, given a context and a process, the distribution over alternative nucleotides is:

$$p(a|z, c) = \rho_{zca}, \sum_{a \in \mathbb{A}_c} \rho_{zca} = 1 \tag{7}$$

where ρ_{zc} is another compositional random variable with prior Dirichlet($\mathbf{1}$)

4 Variational inference

For a corpus of genomes with mutations, each genome has a composition over mutational processes $\pi_g \forall g \in \{1...G\}$ drawn from a Dirichlet prior: $\pi_g \sim \mathrm{Dir}(\alpha)$. Each genome has mutations m_{gi} at a locus ℓ_{gi} associated with a hidden process variable $z_{gi} \forall i \in \{1...N_g\}$. Finally, the random variables associated with the nucleotide signature of each mutational process, ρ and λ, are drawn from uniform Dirichlet priors, and the random variable β is drawn from a prior parameterized by the parameter τ. Altogether, the joint PDF of the model is:

$$p_{\alpha,\tau}(m, l, z, \beta, \pi, \lambda, \rho) = p(\lambda|\mathbf{1})p(\rho|\mathbf{1})p(\beta|\tau)$$
$$\prod_{g=1}^{G} p(\pi_g|\alpha) \prod_{i=1}^{N_g} p(m_{gi}|\ell_{gi}, \lambda_{z_{gi}}, \rho_{z_{gi}})p(\ell_{gi}|\beta_{z_{gi}})p(z_{gi}|\pi_g)$$

Direct inference of the posterior distribution over model parameters is intractable, so we employ variational inference to infer an approximate posterior. Particularly, we assume the parameter posterior densities are independent and are specified by variational parameters such that:

$$p(\pi, \lambda, \rho, \beta, z|m, l) \approx \prod_{k=1}^{K} q(\lambda_k; \delta_k)q(\rho_k; \omega_k)q(\beta_k; \mu_k, \nu_k) \quad (8)$$
$$\prod_{g=1}^{G} q(\pi_g; \gamma_g) \prod_{i=1}^{N_g} q(z_{gi}; \phi_{gi}) \quad (9)$$

$$\begin{aligned} \lambda_k &\sim q(.; \delta_k) = \text{Dir}(\delta_k) \\ \rho_k &\sim q(.; \omega_k) = \text{Dir}(\omega_k) \\ \beta_k &\sim q(.; \mu_k, \nu_k) = \text{Normal}(\mu_k, \nu_k^2) \\ \pi_g &\sim q(.; \gamma_g) = \text{Dir}(\gamma_g) \\ z_{gi} &\sim \text{Categorical}(\phi_{gi}) \end{aligned}$$

To infer the optimal values of the variational parameters, $\Omega^* = \{\delta^*, \omega^*, \mu^*, \nu^*, \gamma^*, \phi^*\}$, we optimize with respect to:

$$\Omega^* = \text{argmin}_\Omega D_{KL}(q(.; \Omega)||p(.|m, \ell)) \quad (10)$$

Or equivalently:

$$\Omega^* = \text{argmax}_\Omega E_{q(.;\Omega)}[\log p_{\alpha,\tau}(m, l, \pi, \lambda, \beta, z)] + \mathcal{H}(q(.; \Omega)) \quad (11)$$

Expanding the terms in (11), we define the objective function ($\mathcal{H}$ is the entropy):

$$\begin{aligned} &\mathcal{L}(\delta, \omega, \mu, \nu, \gamma, \phi) = \\ &E_q[\log p(\beta|\tau)] + E_q[\log p(\pi|\alpha)] + E_q[\log p(z|\pi)] + \\ &E_q[\log p(\ell|z, \beta)] + E_q[\log p(m|\ell, z, \lambda, \rho)] \\ &+\mathcal{H}(q(z; \phi)) + \mathcal{H}(q(\lambda; \delta)) + \mathcal{H}(q(\rho; \omega)) + \mathcal{H}(q(\beta; \mu, \nu)) + \mathcal{H}(q(\pi|\gamma)) \end{aligned} \quad (12)$$

Above, we omit the terms $E_q[\log p(\rho|\mathbf{1})]$ and $E_q[\log p(\lambda|\mathbf{1})]$. Since the prior gives a uniform distribution over all values of ρ, λ these are constant terms.

We optimize the variational parameters in $\mathcal{L}$ using the "Expectation Maximization" (EM) algorithm: we iteratively fix global parameters δ, ω, μ, ν and find optimal local parameters γ, ϕ (E-step), then fix ϕ and find optimal global parameters (M-step). This process is guaranteed to find a stable local optimum with respect to $\mathcal{L}$.

4.1 M-step: Optimization of μ, ν

Here we describe the method to find the optimal distribution for β, parameterized by μ,ν, while holding the other parameters fixed.

First, we subset (12) to only terms which depend on μ, ν, plug in (1), and propagate the expectation by linearity. Below, the notation $X^{(g)}_{[\ell]_{gi}}$ refers to the operation of selecting the row of $X^{(g)}$ which corresponds with the observed locus of the gi^{th} mutation:

$$\begin{aligned}\mathcal{L}^{(\beta)} &= E_q[p(\beta|\tau^2)] + E_q[\log p(\ell|\beta, z)] + \mathcal{H}(q(\beta;\mu,\nu)) \\ &= -\frac{1}{2\tau^2}\sum_{k=1}^{K}\sum_{f=1}^{F}\left(\mu_{kf}^2 + \nu_{kf}^2\right) \\ &+ \sum_{g=1}^{G}\sum_{i=1}^{N_g}\sum_{k=1}^{K}\phi_{gik}\left[E_{\beta_k\sim q(.;\mu_k,\nu_k)}[\beta_k^T X^{(g)}_{[\ell]_{gi}}] - E_{\beta_k\sim q(.;\mu_k,\nu_k)}[\log\sum_{\ell'=1}^{L} l_{\ell'}\exp(\beta_k^T X^{(g)}_{\ell'})]\right] \\ &+ \sum_{k=1}^{K}\sum_{f=1}^{F}\log\nu_{kf}\end{aligned} \tag{13}$$

Via Jensen's inequality, we lower bound the objective function:

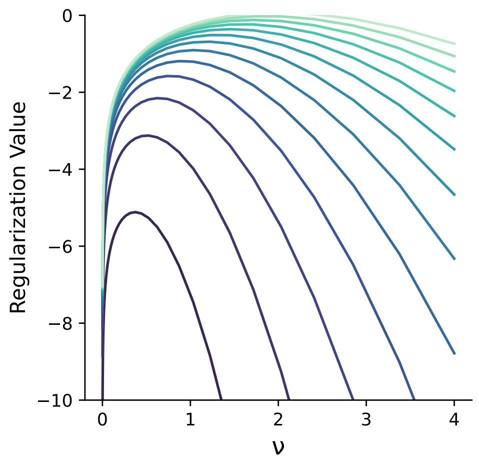
$$\begin{aligned}\mathcal{L}^{(\beta)} \geq \sum_{k=1}^{K}\sum_{f=1}^{F} -\frac{1}{2\tau^2}\left(\mu_{kf}^2 + \nu_{kf}^2\right) + \log\nu_{kf} + \\ \sum_{g=1}^{G}\sum_{i=1}^{N_g}\sum_{k=1}^{K}\phi_{gik}\left[\mu_k^T X^{(g)}_{[\ell]_{gi}} - \log\sum_{\ell'=1}^{L} l_{\ell'}\exp\left(\mu_k^T X^{(g)}_{\ell'} + \frac{1}{2}(\nu_k^T X^{(g)}_{\ell'})^2\right)\right]\end{aligned} \tag{14}$$

We cannot find optimal values for μ,ν analytically, so we use gradient descent. We define another lower-bound on the objective with the introduction of the $\zeta \in \mathbb{R}^K$ parameter, which removes the $\sum\exp$ term from the log and greatly simplifies calculation of the derivative. Below, we consider the scenario where every sample has the same genomic correlates (all samples have homogenous gene expression, chromatin accessibility, etc.), which further simplifies the calculations.

$$\mathcal{L}^{(\beta)} \geq \hat{\mathcal{L}}_\zeta^{(\beta)}(\mu,\nu) = \sum_{k=1}^{K}\sum_{f=1}^{F} -\frac{1}{2\tau^2}(\mu_{kf}^2+\nu_{kf}^2) + \log \nu_{kf}$$
$$+ \sum_{\ell=1}^{L}\left(\sum_{g=1}^{G}\sum_{i=1}^{N_g}\phi_{gik}\mathbb{I}_{[\ell]_{gi}=\ell}\right)\mu_k^T X_\ell \qquad (15)$$
$$-\frac{\sum_{g=1}^{G}\sum_{i=1}^{N_g}\phi_{gik}}{\zeta_k}\sum_{\ell'=1}^{L} l_{\ell'}\exp\{\mu_k^T X_{\ell'} + \frac{1}{2}(\nu_k^T X_{\ell'})^2\} + 1 - \log\zeta_k$$

The first line of (15) is $E_{\beta\sim q}[\log p(\beta|\tau)] + \mathcal{H}(q(\beta))$, and serves to regularize the values of μ and ν. Intuitively, the regularizer of ν is a concave function defined over the positive reals with a maximum at $\nu = \tau$.

Figure 1: Function $-\frac{1}{2\tau^2}\nu^2 + \log\nu$ for decreasing values of τ



The second line of (15) shows the sufficient statistics to update the variational parameters of β are the number of mutations that fall within each locus weighted by the probability that mutation was generated by process k. To optimize, we first set ζ:

$$\zeta_k = \sum_{\ell'=1}^{L} l_{\ell'}\exp\left\{\mu_k^T X_{\ell'} + \frac{1}{2}(\nu_k^T X_{\ell'})^2\right\} \qquad (16)$$

then provide $\hat{\mathcal{L}}_\zeta^{(\beta)}$ and analytical derivatives $\frac{\partial\hat{\mathcal{L}}_\zeta^{(\beta)}}{\partial\mu}$, and $\frac{\partial\hat{\mathcal{L}}_\zeta^{(\beta)}}{\partial\nu}$ to *scipy*'s L-BFGS-B optimization function.

4.2 M-step: optimization of δ, ω

Work in progress!

$E_q[\log p(m|\ell, z, \lambda, \rho)] = E_{\rho\sim q(.;\omega)}[\log p(a|z, c, \omega)] + E_{\lambda\sim q(.;\delta)}[\log p(c|z, \ell, \lambda)]$
$\mathcal{L}^{(\delta)} = E_{\lambda\sim q(.;\delta)}[\log p(c|z, \ell, \lambda)] + \mathcal{H}(q(\lambda; \delta))$

$$\begin{aligned}\mathcal{L}^{(\delta)} \geq \sum_{k=1}^{K}\sum_{c=1}^{|\mathbb{C}|}\left(\sum_{g=1}^{G}\sum_{i=1}^{N_g}\phi_{gik}\mathbb{I}_{[c]_g i=c}\right)\left(\Psi(\delta_{kc}) - \Psi(\sum_{c'=1}^{|\mathbb{C}|}\delta_{kc'})\right) \\ -\sum_{\ell=1}^{L}\left(\sum_{g=1}^{G}\sum_{i=1}^{N_g}\phi_{gik}\mathbb{I}_{[\ell]_{gi}=\ell}\right)\log\sum_{c'=1}^{|\mathbb{C}|}\lambda_{kc'}(\sum\lambda_k)^{-1}t_{c'\ell} \\ +\mathcal{H}(q(.;\delta))\end{aligned} \tag{17}$$

$$\text{argmax}_{\omega_{ca}} E_{\rho\sim q(.;\omega)}[\log p(a|z, c, \omega)] + \mathcal{H}(q(\rho; \omega)) = \sum_{g=1}^{G}\sum_{i=1}^{N_g}\sum_{k=1}^{K}\phi_{gik}\mathbb{I}_{[c]_{gi}=c,[a]_{gi}=a} + 1 \tag{18}$$

4.3 E-step: Optimization of γ, ϕ

The ϕ parameter represents the posterior distribution over generative processes for each mutation $p(z = k|m, l)$, and is given by:

$$\phi_k \propto \exp\left(E_q[\log p(m|\ell, z = k, \lambda, \rho)] + E_q[\log p(\ell|\beta, z = k)] + E_q[\log p(z|\pi)]\right) \tag{19}$$

To update γ:

$$\gamma_g k = \alpha_k + \sum_{i=1}^{N_g}\phi_{gik} \tag{20}$$

Because the ϕ update depends on γ, and vice versa, these two parameters are optimized iteratively until γ converges to a stable value.

5 Stochastic Variational Inference

The procedure outlined in **Algorithm 2** converges to a stable fixed point local optimum over many iterations through the dataset. The most time-consuming step of each iteration is the update of the β distribution's variational parameters, since computation of the gradient of (17) requires the summation of $E_q[\beta^T X]$ over all loci for each iteration of gradient descent.

Algorithm 2 Inference

```
initialize $\Omega = \{\delta, \omega, \mu, \nu, \gamma, \phi\}$
initialize parameters $\alpha, \tau$
while $\Omega$ not converged do
    E-step:
    for all $g \in 1...G$ do
        while $\gamma_g$ not converged do
            update $\phi_{gi} \forall i \in \{1, ..., N_g\}$
            update $\gamma_g$ with $\phi_g$
        end while
    end for
    M-step:
    update $\mu, \nu$ with $\phi$
    update $\delta, \rho$ with $\phi$
    if empirical Bayes: then
        update $\alpha, \tau$
    end if
end while
return $\Omega, \alpha, \tau$
```

In this section, we derive a stochastic variational update step which linearly reduces the time to iterate through the dataset, leads to faster convergence, and produces models with greater marginal likelihoods.

Stochastic variational inference involves iteratively randomly downsampling the original dataset, finding optimal parameters on that subset as if the entire dataset had the same distribution, then updating a dataset-level copy of each parameter based on a weighted combination of each subsets' optimal parameters. Usually, this algorithm works by downsampling along the samples axis to calculate the updates, but here, we randomly downsample the number of *loci* to model in each update. Importantly, each parameter update equation derived above is an unbiased estimator for the data-level paramter value when calculated on a subset of loci in the dataset.

Stochastic variational inference will likewise converged to a stable fixed point local optimum with respect to (15) if the dataset-level parameters, for example, λ, are updated at the t^{th} iteration according to:

$$\lambda^t = (1 - d(t))\lambda^{t-1} + d(t) \cdot \hat{\lambda}^t \tag{21}$$

where $\hat{\lambda}^t$ is the optimal value calculated from the downsampled dataset at iteration t, and $d(t)$ is a the learning rate at epoch t: $d(t) = t^{-0.5}$.

In practice, the SVI procedure (**Algorithm 3**) not only coverges a magnitude faster, but produces models with superior marginal likelihoods and test set generalization than standard variational inference (**Figure 2**). This may be because stochastic subsampling regularizes and smooths the objective space, which allows the model to avoid shallow local optima.

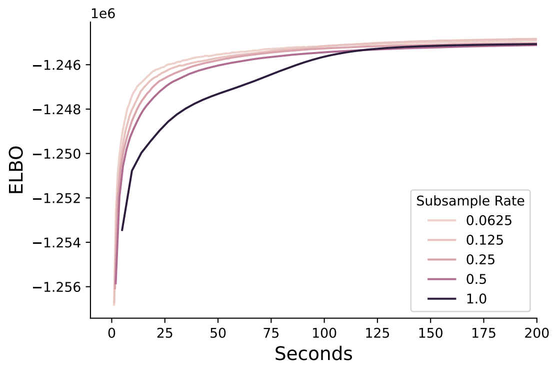
6 Future Directions

A key limitation of the current generative model is that the local mutation rate for each process z in each locus ℓ is modeled as proportional to $\exp(\beta_z^T X_\ell)$,

Algorithm 3 Stochastic Variational Inference

```
initialize dataset-level $\Omega = \{\delta, \omega, \mu, \nu, \gamma, \phi\}$
initialize parameters $\alpha, \tau$
set t = 1
set subsample = 0.1
while $\Omega$ not converged do
    $\hat{\mathbb{D}}$ = downsampleLoci($\mathbb{D}$, subsample )
    $\hat{\Omega}$ = inference($\hat{\mathbb{D}}, \Omega, \alpha, \tau$)
    set $\Omega = (1 - d(t))\Omega + d(t)\hat{\Omega}$
    set $t = t + 1$
    if empirical Bayes: then
        update $\alpha, \tau$
    end if
end while
```

Figure 2: Bound convergence on training samples for PCAWG Esophogeal Adenocarcinoma Chr1 mutations dataset, colored by locus sampling rate (1 is no downsampling.)



which assumes that the mutation rate for each locus is independent. There are some work-arounds that can be achieved even within the framework outlined here. The easiest solution is to model the genomic correlates at multiple resolutions (e.g. 10Kb, 100Kb, 1Mb), which can capture attributes about the wider regulatory neighborhood. Another option is to model the correlates using a convolutional kernel, where coefficients are learned to weight features not just in the current locus, but neighboring loci as well:

$$\lambda_{z\ell} \propto \exp] \left(\sum_{j=-w}^{w} \sum_{f=1}^{F} \beta_{zjf} X_{(\ell+j),f} \right) \tag{22}$$

The convolutional kernel above learns a coefficient β for each feature at each locus which is within w windows upstream or downstream of ℓ. This model is equivalent to a one-layer Bayesian convolutional neural network. Generalizing this convolutional kernel to include nonlinearities, one could use the ”kernel trick”:

$$\begin{aligned}\lambda_{z\ell} &\propto \exp\left(\sum_{\ell'=1}^{L} \beta_{zl} \cdot \mathrm{K}(X_\ell, X_{\ell'})\right) \\ \mathrm{K}(X_\ell, X_{\ell'}) &= \exp\left(\sum_{j=-w}^{w} \sum_{f=1}^{F} \left(X_{(\ell+j),f} - X_{(\ell'+j),f}\right)^2\right)\end{aligned} \tag{23}$$

The kernel matrix K is an L×L matrix, but this dimensionality could be reduced by Nystrom kernel approximation. More broadly, one could substitute a complex model (kernels, deep CNNs, transformers, etc.) which maximizes:

$$\mathcal{L}^{(\beta)} \geq E_q[p(.|\text{prior})] + \mathcal{H}(q) + \\ \sum_{g=1}^{G} \sum_{i=1}^{N_g} \sum_{k=1}^{K} \phi_{gik} \left[E_{\lambda_k \sim q(.|X^{(g)}, z=k)}[\log \lambda_{k[\ell]_{gi}}] - \log \sum_{\ell'=1}^{L} l_{\ell'} E_{\lambda_k \sim q(.;X^{(g)}, z=k)}[\lambda_{k\ell'}] \right] \tag{24}$$

The more general form above shows that the posterior distribution over mutation rates must be defined for both an expectation and log expectation in each window. This suggests the model for mutation rate at locus ℓ may be parameterized by:

$$\lambda_{z\ell} \sim \text{Gamma}(a(X_\ell), b(X_\ell)) \tag{25}$$

Where a, b are functions which predict the parameters of the Gamma distribution using the features of the locus. This direct inference of uncertainty at each locus admits a negative binomial distribution over mutations at each locus, and an overdispersed dirichlet-multinomial distribution across all loci.