Notes on AllelicCapSeg

Samuel K. Lee
Broad Institute, 75 Ames Street, Cambridge, MA 02142[*]
(Dated: October 21, 2015)

Some notes on the methods used in the current python implementation and proposed methods for the hellbender port.

I. INTRODUCTION

[SL: A brief intro here would be nice.]

II. ReCapSeg OVERVIEW

We first summarize the portions of the ReCapSeg workflow that generate the input for AllelicCapSeg. The below is copied/paraphrased from the documentation at http://gatkforums.broadinstitute.org/discussion/5640/recapseg-overview.

ReCapSeg is a copy-number–variant detector that runs on user-defined target regions, which can correspond to exomes, gene panels, or arbitrary windows. ReCapSeg uses a Panel of Normal (PoN) samples to model noise and normalize the coverage calls of the target sample. These methods were designed for copy-number calling (amplification and deletion) of somatic events with a resolution of two or more targets. ReCapSeg does not need a matched normal, but operates on a panel of normal samples representing similar library preparation to agnostically remove noise. ReCapSeg is the production version of the CapSeg algorithm.

Given an DNA-Seq alignment BAM file and a BED file of target genomic intervals, the ReCapSeg algorithms estimate copy ratio by performing the following steps:

1. Generate proportional coverage: First, the per sample normalized coverage is calculated by normalizing the read coverage spanning a target segment with the total number of aligned reads (for every read group: number of reads over segment/total number of aligned reads). The proportional coverage is then calculated by normalizing every segment with the median normalized coverage across the PoN for the given segment.

2. Tangent normalization: This normalization procedure projects the sample proportional coverage to a hyperplane defined by the PoN. This normalization procedure results in a copy-ratio estimate with reduced noise.

3. Segment: The target regions are then merged into continuous segments that represent the same copy-number event. The segmentation is performed by a circular-binary-segmentation (CBS) algorithm described by Olshen et al. 2004 that was originally developed to segment noisy array copy-number data.[1] Currently, ReCapSeg considers only segments that include two or more targets (a target usually represents a single exon).

To summarize, given coverage data for a set of targets, ReCapSeg produces 1) $\log_2$ copy-ratio estimates for each target, and 2) corresponding segments, which are specified by a set of genomic intervals. The segment files produced by ReCapSeg also contain segment "means," which are given by the mean linear copy ratio of all targets contained within each segment.

[*] Electronic address: slee@broadinstitute.org

[1] Specifically, the CBS implementation provided by the R package DNACopy is used. Note that even though the DNACopy parameter data.type is set to logratio, the tangent-normalized *linear* copy ratios are used instead. [SL: Check this? If true, it is unclear to me how this affects the resulting segmentation, if at all.]

III. CURRENT HELLBENDER `AllelicCapSeg` WORKFLOW

A. Segmented model

We want a generative model for allelic fractions that infers its parameters from the data. We observe alt and ref read counts for each het site and wish to infer the minor allelic fraction of every segment. Let's consider what other hidden variables belong in the model. Read counts obey an overdispersed binomial distribution in which the probability of an alt read is a site-dependent random variable. Letting θ_j be the probability that a mapped read at het j is an alt we have

$$P(a_j, r_j|\theta_j) = \binom{a_j + r_j}{a_j} \theta_j^{a_j} (1 - \theta_j)^{r_j} = \binom{n_j}{a_j} \theta_j^{a_j} (1 - \theta_j)^{r_j}, \tag{1}$$

where a_j and r_j are alt and ref read counts and $n_j = a_j + r_j$ is the total read count at site j. Now we consider θ_j. Suppose site j belongs to a segment with minor allelic fraction f and is alt minor, such that $P(\text{alt}) = f$ and $P(\text{ref}) = 1 - f$ are the probabilities that a random DNA fragment will contain the alt and ref alleles. Let $x_j^{\text{alt(ref)}} = P(\text{mapped}|\text{alt(ref)})$ be the probabilities that an alt (ref) DNA fragment at site j eventually gets sequenced and mapped. Then θ_j is the conditional probability that a mapped read comes from an alt fragment:

$$\theta_j = P(\text{alt}|\text{mapped}) = \frac{P(\text{alt})P(\text{mapped}|\text{alt})}{P(\text{alt})P(\text{mapped}|\text{alt}) + P(\text{ref})P(\text{mapped}|\text{ref})} \tag{2}$$

$$= \frac{f x_j^{\text{alt}}}{f x^{\text{alt}} + (1 - f) x_j^{\text{ref}}} = \frac{f}{f + (1 - f)\lambda_j}, \tag{3}$$

where $\lambda_j = x_j^{\text{ref}} / x_j^{\text{alt}}$ is the "bias ratio" of ref to alt sequenceability and mappability at site j. A similar result for ref minor sites follows from substituting $f \leftrightarrow 1 - f$. In addition to the bias ratio λ_j we need an indicator variables z_j with three states, alt minor, ref minor, and an outlier state that gives robustness to anomalous events. For this outlier state we average the binomial likelihood over all θ to get:

$$P(a_j, r_j|\text{outlier}) = \binom{n_j}{a_j} \int_0^1 \theta_j^{a_j} (1 - \theta_j)^{r_j} \, d\theta_j = \binom{n_j}{a_j} \frac{a_j! r_j!}{(n_j + 1)!} \tag{4}$$

For notational convenience we give z_j a one-of-K encoding $z_j = (z_{ja}, z_{jr}, z_{jo})$ in which one component equals 1 and the rest 0.

The contribution of site j to the likelihood is

$$P(a_j, r_j|f_s, \lambda_j, z_j) = \binom{n_j}{a_j} \left[\frac{f_s^{a_j} (1 - f_s)^{r_j} \lambda_j^{r_j}}{(f_s + (1 - f_s)\lambda_j)^{n_j}} \right]^{z_{ja}} \left[\frac{(1 - f_s)^{a_j} f_s^{r_j} \lambda_j^{r_j}}{(1 - f_s + f_s \lambda_j)^{n_j}} \right]^{z_{jr}} \left[\frac{a_j! r_j!}{(n_j + 1)!} \right]^{z_{jo}} \tag{5}$$

where f_s is the minor allele fraction of the segment containing site j. We will consider f to be drawn from a uniform distribution on $[0, 1/2]$ – that is, we give it a flat prior – but in the future we can obtain some sort of clustering behavior, representing the fact that events in the same subclone that exhibit the same integer copy numbers will have the same minor allelic fractions, by drawing f_s from a Dirichlet process.

We assume that the bias ratios come from a common Gamma distribution with parameters α, β:

$$P(\lambda_j|\alpha, \beta) = \frac{\beta^\alpha}{\Gamma(\alpha)} \lambda_j^{\alpha - 1} e^{-\beta \lambda_j} \tag{6}$$

Note that bias ratios tend to be near 1.0 and so the choice of distribution is not too important as long as it has adjustable mean and standard deviation. We choose the Gamma distribution because it is the simplest such distribution on $\mathbb{R}^+$. We will give the parameters α and β a flat prior $P(\alpha, \beta) \propto 1$.

Finally, the indicator z_j is a multinomial random variable distributed according to parameter vector π:

$$P(z_{ja(r,o)} = 1|\pi) = \pi_{a(r,o)} \tag{7}$$

We set the alt and ref minor probabilities equal so that the only free parameter is π_o, with $\pi_{a(r)} = (1 - \pi_o)/2$. The Bayesian network corresponding to this model is shown in Figure III A.

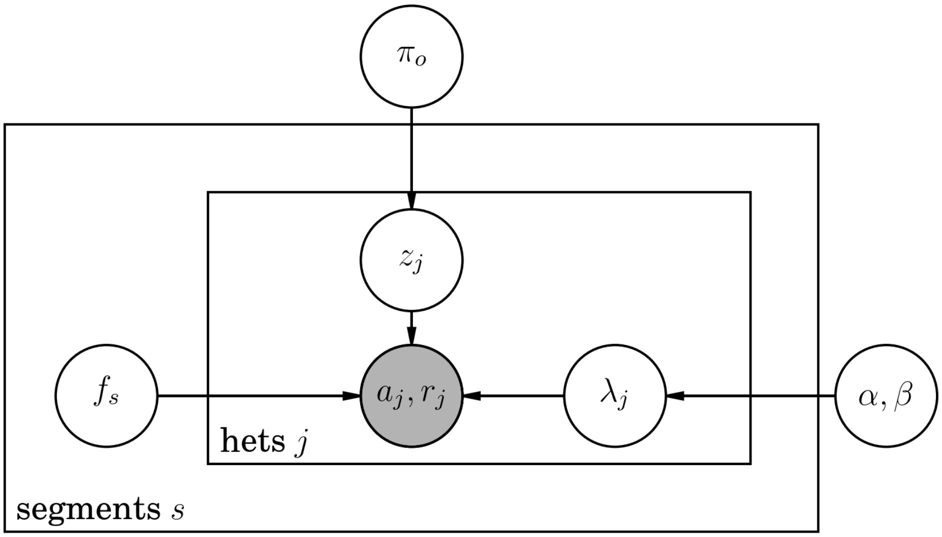

FIG. 1: Graphical model for `AllelicCapSeg`

As with the other parameters, we put a flat prior on π. Putting all the pieces together the complete-data likelihood is

$$P(f,a,r,z,\lambda,\alpha,\beta) = \prod_s \prod_{j \in s} P(a_j, r_j | f_s, \lambda_j, z_j) P(z_j|\pi) P(\lambda_j|\alpha,\beta), \tag{8}$$

where the shorthand $j \in s$ means that het site j belongs to segment s.

We can read off the conditional probabilities necessary for Gibbs sampling from the model. Up to constant factors the conditional on the outlier probability is,

$$P(\pi_o|\{z_j\}) \propto \pi_o^{N_o}(1-\pi_o)^{N-N_o}, \tag{9}$$

where $N_o = \sum_j z_{jo}$ is the number of indicator variables in the outlier state. Thus the conditional distribution on π_o given $\{z_j\}$ is $\mathrm{Beta}(N_o, N - N_o)$.

The conditional distribution on f_s is the product of likelihoods over sites $j \in s$:

$$P(f_s|a,r,\lambda,z) \propto \prod_{j \in s} \left[\frac{f_s^{a_j}(1-f_s)^{r_j}}{(f_s + (1-f_s)\lambda_j)^{n_j}} \right]^{z_{ja}} \left[\frac{(1-f_s)^{a_j} f_s^{r_j}}{(1-f_s+f_s\lambda_j)^{n_j}} \right]^{z_{jr}} \tag{10}$$

Since, in general $\lambda_j \approx 1$, this conditional approximately reduces to $\mathrm{Beta}(N^s_{\mathrm{AA}} + N^s_{\mathrm{RR}}, N^s_{\mathrm{AR}} + N^s_{\mathrm{RA}})$, where e.g. $N_{AR} = \sum_{j \in s} z_{ja} r_j$ is the total number of ref reads in alt minor sites. This is the posterior distribution of the minor allele fraction given $N^s_{\mathrm{AA}} + N^s_{\mathrm{RR}}$ observations of minor alleles and $N^s_{\mathrm{AR}} + N^s_{\mathrm{RA}}$ of non-minor alleles. Actually using this approximation would defeat the purpose of modeling the allelic bias, but it gives us a good sense of the shape of the posterior – it is very likely to be unimodal, hence amenable to simple univariate slice sampling.

The conditional distribution on λ_j is the product of the likelihood term for site j with $\mathrm{Gamma}(\alpha,\beta)$.

$$P(\lambda_j|a_j,r_j,f_s,z_j) \propto \left[\frac{\lambda_j^{r_j}}{(f_s + (1-f_s)\lambda_j)^{n_j}} \right]^{z_{ja}} \left[\frac{\lambda_j^{r_j}}{(1-f_s+f_s\lambda_j)^{n_j}} \right]^{z_{jr}} \lambda_j^{\alpha-1} e^{-\beta\lambda_j} \tag{11}$$

$$= (f_s + (1-f_s)\lambda_j)^{-n_j(z_{ja}+z_{jr})} \lambda_j^{r_j(z_{ja}+z_{jr})+\alpha-1} e^{-\beta\lambda_j} \tag{12}$$

This conditional distribution is also easily sampled from via univariate slice sampling.

The conditional on β is

$$P(\beta|\alpha,\lambda) \propto \beta^{N\alpha} e^{-\sum_j \lambda_j \beta}, \tag{13}$$

which is the gamma distribution $\Gamma(N\alpha + 1, \sum_j \lambda_j)$.

The conditional on α is

$$P(\alpha|\beta,\lambda) \propto \frac{\exp\left[\left(N \ln \beta + \sum_j \ln \lambda_j\right) \alpha\right]}{\Gamma(\alpha)^N} \tag{14}$$

which is not of a standard form but is efficient to compute and thus amenable to univariate slice sampling.

Finally, z_j is a categorical variable with priors given by π, and its posterior can be obtained simply by plugging in its three possible values into the likelihood.

Thus we have completely specified an MCMC scheme for this model, given by Algorithm 1:

Algorithm 1 MCMC algorithm for `AllelicCapSeg`

```
Initialize $z_j$ to be alt minor if $a_j < r_j$, ref minor otherwise.
Initialize $f_s = (N^s_{\rm AA} + N^s_{\rm RR}) \,/\, (N^s_{\rm AR} + N^s_{\rm RA})$
Initialize $\lambda$ randomly and approximately equal to 1.
repeat
    Slice sample each $f_s$ from the conditional Eq. 10.
    for all sites $j$ do
        Slice sample $\lambda_j$ from the conditional Eq. 12.
        Sample $z_j$ from the categorical distribution defined by Eq. 5.
    end for
    Sample $\pi_o$ from the beta conditional Eq. 9.
    Sample $\beta$ from the gamma conditional Eq. 13.
    Slice sample $\alpha$ from the conditional Eq. 14.
until convergence
```

B. Target/SNP segment union

[SL: DB can fill this in.]

C. Small-segment merging

[SL: SL to update this to the new method.]

Using CBS to segment the targets in `ReCapSeg` results in segments that are [SL: always?] larger than ∼2–3 targets. However, after taking the union of target and SNP segments, small segments with less than ∼2–3 targets may be introduced. To be consistent with CBS and `ReCapSeg`, `AllelicCapSeg` treats these small segments as spurious, and removes them by merging them with adjacent segments.

A segment is considered to be small if the number of targets it contains is strictly less than a threshold number of targets n_t; we take $n_t = 3$. The small-segment merging algorithm checks each ith segment in turn, starting with the first, leftmost segment. If the segment is small, it is repeatedly merged with the adjacent segment that is closer in the L_1 distance $|\tau_i - \tau_{i\pm1}| + |f_i - f_{i\pm1}|$ in copy-ratio–allele-fraction space until it is no longer small.[2] Exceptions occur for adjacent segments on different chromosomes, which are never merged; in practice, this is enforced by setting the L_1 distance between segments on different chromosomes to be infinite. After all segments have been checked and merged, any remaining small segments (which will be present if any chromosome contains less than n_t targets) are dropped.

[2] To be explicit, segments are reindexed after each merge, so that the new segment formed by merging segment i and segment $i \pm 1$ retains the index i.

D. Parameter optimization

E. Similar-segment merging

F. Final parameter optimization

IV. PROPOSED METHODS FOR hellbender

A. Bayesian het pulldown

We are given a large data set of ref and alt read counts over many potential SNP sites and we wish to infer which sites are hets and with what probabilities. This problem is naturally represented as a mixture model in which the hidden labels are genotypes – hom ref, het, and hom alt. Since the observed data are ref and alt counts is seems natural to use a binomial mixture model in which the binomial parameters are the probability of an alt read. Then the binomial parameters are the error rate for hom ref genotypes, 1/2 times the allelic bias for het genotypes, and 1 minus the error rate for hom alt genotypes. However, actual data are overdispersed because the error rate and allelic bias are random variables, not single parameters. For example, sequencing error rates and allelic bias (for concreteness, consider mapping bias) depend on context. Thus a beta-binomial mixture model is more appropriate. A maximum-likelihood (MLE) approach will yield posterior probabilities on the genotypes at each locus, in particular the het probability. It also gives the parameters of a beta distribution of allelic bias, which is useful downstream in ACS.

For generality and at no cost in complexity, consider a Dirichlet-multinomial mixture (DMM) with K mixture components and M classes of observed data. For our purposes there are $K = 3$ genotypes and $M = 2$ types of read, ref and alt. The beta-binomial distribution is the $M = 2$ case of the Dirichlet-multinomial. The observed data are counts n_{jm}, the number of times class m was seen in observation j. For us, each potential SNP site is a datum j. Let $N_j = \sum_m n_{jm}$ denote the total read count at site j. For our purposes, $\{N_j\}$ are constants – we are not trying to model depth of coverage here, just the proportions of the coverage allotted to ref and alt reads.

We represent hidden labels via the 1-of-K encoding $\mathbf{z}_j = (0, 0, \ldots 1, 0, 0 \ldots)$, so $z_{jk} = 1$ when datum j comes from component k. The hidden labels are multinomially distributed as $P(\mathbf{z}_j) = \mathrm{Mult}(\mathbf{z}_j|\pi)$, where π_k is the probability of component k and $\sum_k \pi_k = 1$. Finally, the observed counts for mixture component k are drawn from a Dirichlet-multinomial distribution with parameter α_k:

$$P(\mathbf{n}_j \mid z_{jk} = 1, \alpha_k) = \frac{\Gamma(A_k)}{\Gamma(A_k + N_j)} \prod_m \frac{\Gamma(\alpha_{km} + n_{jm})}{\Gamma(\alpha_{km})}, \tag{15}$$

where $A_k = \sum_m \alpha_{km}$.

The EM algorithm for MLE estimates of $\{\pi_k\}$ and $\{\alpha_{km}\}$ requires the complete-data likelihood (CDL), that is, the joint likelihood of the observed data and hidden labels given the parameters. In contrast, a direct approach maximizes the likelihood marginalized over the hidden variables. The CDL of of the DMM is

$$P(\mathbf{z}, \mathbf{n} \mid \pi, \alpha) = P(\mathbf{z} \mid \pi) P(\mathbf{n} \mid \mathbf{z}, \alpha) \tag{16}$$

$$= \prod_{jk} \left[\pi_k \frac{\Gamma(A_k)}{\Gamma(A_k + N_j)} \prod_m \frac{\Gamma(\alpha_{km} + n_{jm})}{\Gamma(\alpha_{km})} \right]^{z_{jk}} \tag{17}$$

In the E step of the EM algorithm, we obtain the posterior distribution on $P(\mathbf{z} \mid \mathbf{n}, \pi, \alpha)$ from Eq. (17). By inspection the posterior is a product of independent multinomials

$$\bar{z}_{jk} \equiv P(z_{jk} = 1 \mid \mathbf{n}, \pi, \alpha) \propto \pi_k \frac{\Gamma(A_k)}{\Gamma(A_k + N_j)} \prod_m \frac{\Gamma(\alpha_{km} + n_{jm})}{\Gamma(\alpha_{km})}, \tag{18}$$

with a normalization constant determined by the condition $\sum_k \bar{z}_{jk} = 1$.

In the M step of the EM algorithm we take the expectation of the log-CDL with respect to the posterior on $\mathbf{z}$ and maximize with respect to π and α. That is, we maximize

$$\sum_{jk} \bar{z}_{jk} \left\{ \log \pi_k + \log \frac{\Gamma(A_k)}{\Gamma(A_k + N_j)} + \sum_m \log \frac{\Gamma(\alpha_{km} + n_{jm})}{\Gamma(\alpha_{km})} \right\}. \tag{19}$$

Maximizing with respect to π_k with a Lagrange multiplier for the constraint $\sum_k \pi_k = 1$

$$\pi_k = \frac{\sum_j \bar{z}_{jk}}{\sum_{j\ell} \bar{z}_{j\ell}} \tag{20}$$

To maximize with respect to α we use the fact that if we are trying to maximize $f(\mathbf{x})$ and have a current guess of $\mathbf{x}_0$, then an improved guess may be obtained by maximizing $g(\mathbf{x})$, where $g(\mathbf{x}_0) = f(\mathbf{x}_0)$ and $g(\mathbf{x}) \leq f(\mathbf{x})$ for all $\mathbf{x}$. Furthermore, repeating this gives an iterative optimization that converges to a local maximum. Using bounds (A7) and (A9) and dropping additive constants, we find that the iterative step is to maximize the lower bound

$$\sum_{jk} \bar{z}_{jk} \left\{ -\left(\psi(\hat{A}_k + N_j) - \psi(\hat{A}_k) \right) A_k + \sum_m \hat{\alpha}_{km} \left(\psi(\hat{\alpha}_{km} + n_{jm}) - \psi(\hat{\alpha}_{km}) \right) \log(\alpha_{km}) \right\}. \tag{21}$$

with respect to α_{km} treating the "old" guesses $\hat{\alpha}_{km}$ as constants. This maximization is a straightforward matter of setting the derivative to zero and gives the fixed-point iteration

$$\alpha_{km} = \hat{\alpha}_{km} \frac{\sum_j \bar{z}_{jk} \left(\psi(\hat{\alpha}_{km} + n_{jm}) - \psi(\hat{\alpha}_{km}) \right)}{\sum_j \bar{z}_{jk} \left(\psi(\hat{A}_k + N_j) - \psi(\hat{A}_k) \right)} \tag{22}$$

As is often the case with mixture models, we risk converging to a bad local maximum if parameters are initialized poorly. Following the approach used by Thomas Minka in his FastFit software, we obtain a good initial guess by fitting a Dirichlet mixture model (as opposed to a Dirichlet-multinomial model) on effective multinomial pseudodata. That is, instead of working with *counts* n_{jm}, work with *proportions* $p_{jm} = n_{jm}/N_j$. Since $\sum_m p_{jm} = 1$, $\mathbf{p}_j$ can be interpreted as a multinomial distribution drawn from a Dirichlet mixture. This preprocessing step maps the original count data onto the $(M-1)$-dimensional simplex, and we can then assign the pseudo-multinomials $\{\mathbf{p}_j\}$ to K clusters via the K-means algorithm. Define the indicator variable $\chi_{jk} = 1$ if pseudo-multinomial $\mathbf{p}_j$ is assigned to cluster k and let $\chi_{jk} = 0$ otherwise.

We initialize π_k as the empirical proportion of mixture component k in the clustering step. That is

$$\pi_k = \frac{\sum_j \chi_{jk}}{N} = \frac{N_k}{N} \tag{23}$$

where N_k is the number of pseudo-multinomials assigned to cluster k.

Then for each component k we initialize the Dirichlet parameter vector $\alpha_{\mathbf{k}}$ via moment matching. Parameterize $\alpha_{\mathbf{k}}$ as $\alpha_k = s_k \theta_k$, where $\sum_m \theta_{km} = 1$ is the mean of the Dirichlet distribution and s is its concentration. Since multinomials $S_k = \{\mathbf{p}_j : \chi_{jk} = 1\}$ are presumed drawn from Dirichlet distribution with parameter α_k, we set the theoretical mean θ_k to the empirical mean of S_k:

$$\theta_{km} = \langle \mathbf{p}_j \in S_k \rangle = \frac{1}{N_k} \sum_j \chi_{jk} p_{jm} \tag{24}$$

Moment matching of the m-th diagonal component of the covariance gives

$$\begin{aligned} \frac{\alpha_{km} \left(\sum_\ell \alpha_{k\ell} - \alpha_{km} \right)}{\left(\sum_\ell \alpha_{k\ell} \right)^2 \left(\sum_\ell \alpha_{k\ell} + 1 \right)} &= \mathrm{cov}(S_k)_{mm} = \left\langle p_{jm}^2 \in S_k \right\rangle - \left\langle p_{jm} \in S_k \right\rangle^2 && (25) \\ \frac{\theta_{km}(1 - \theta_{km})}{s_k + 1} &= \frac{1}{N_k} \sum_j \chi_{jk} p_{jm}^2 && (26) \\ s_k &= \frac{\theta_{km} - \frac{1}{N_k} \sum_j \chi_{jk} p_{jm}^2}{\frac{1}{N_k} \sum_j \chi_{jk} p_{jm}^2 - \theta_{km}^2} && (27) \end{aligned}$$

Since these M estimates s_k do not need to agree, we simply take their average.

The EM algorithm for DMM inference is summarized in Algorithm 2.

Returning to our original task, we obtain three mixture components with Dirichlet parameters $(\alpha_{k1}, \alpha_{k2})$. The mean proportion of alt reads (WLOG we choose $m = 1$ to be alt and $m = 2$ to be ref) are $\alpha_{k1}/(\alpha_{k1} + \alpha_{k2})$, so we can assign mixture labels $k = 1, 2, 3$ to genotypes by comparing these proportions to 0 (hom ref), 1/2 (het) and 1 (hom alt). The posterior probability $\bar{z}_{jk}$ is the probability that site j has genotype k, which is exactly what we need for a probabilistic het pulldown.

Algorithm 2 EM algorithm for Dirichlet-multinomial mixture model

```
Form pseudo-multinomial data $p_{jm} = n_{jm}/N_j$
Find $K$ clusters of this pseudodata via the $K$-means algorithm.
Initialize $\pi$ via Eq. 23
Initialize $\{\alpha_{km}\}$ via Eqs. 24 and 27
repeat
    Update $\bar{z}_{jk}$ via Eq. 18
    Update $\pi$ via Eq. 20
    repeat
        update $\{\alpha_{km}\}$ via Eq. 22
    until convergence
until convergence
```

B. Improving `AllelicCapSeg` model sampling

The `AllelicCapSeg`model likelihood has thorny factors of the form $(f_s + (1 - f_s)\lambda_j)^{-n_j}$ (and " $f \leftrightarrow 1 - f$ "). Other than these, everything comes from exponential family distributions, is log-concave etc. If we could rid ourselves of this term and replace it with a standard distribution we could greatly speed our Gibbs sampling. Consider the identities

$$(f_s + (1 - f_s)\lambda_j)^{-n_j} = \frac{1}{\Gamma(n_j)} \int_0^\infty w_j^{n_j - 1} e^{-(f_s + (1 - f_s)\lambda_j) w_j} \, dw_j \tag{28}$$

$$(1 - f_s + f_s\lambda_j)^{-n_j} = \frac{1}{\Gamma(n_j)} \int_0^\infty w_j^{n_j - 1} e^{-(1 - f_s + f_s\lambda_j) w_j} \, dw_j \tag{29}$$

$$1 = \frac{1}{\Gamma(n_j)} \int_0^\infty w_j^{n_j - 1} e^{-w_j} \, dw_j \tag{30}$$

Applying these to the likelihood terms when $z_{ja} = 1$, $z_{zr} = 1$, and $z_{jo} = 1$ respectively we see that if we augment the state space with an auxiliary variable w_j for each het site we may replace the likelihood Eq. 5 with the more tractable

$$P(a_j, r_j | f_s, \lambda_j, w_j, z_j) \propto w_j^{n_j - 1} \left[f_s^{a_j} (1 - f_s)^{r_j} \lambda_j^{r_j} e^{-(f_s + (1 - f_s)\lambda_j) w_j} \right]^{z_{ja}} \left[(1 - f_s)^{a_j} f_s^{r_j} \lambda_j^{r_j} e^{-(1 - f_s + f_s\lambda_j) w_j} \right]^{z_{jr}} \left[\frac{a_j! r_j!}{(n_j + 1)!} e^{-w_j} \right]^{z_{jo}} \tag{31}$$

The complete-data likelihood becomes

$$P(f, a, r, z, \lambda, w, \alpha, \beta) = \prod_s \prod_{j \in s} P(a_j, r_j | f_s, \lambda_j, w_j, z_j) P(z_j | \pi) P(\lambda_j | \alpha, \beta), \tag{32}$$

with the terms $P(z_j|\pi)$ and $P(\lambda_j|\alpha, \beta)$ unchanged from above. The virtues of this "de-marginalization" are apparent when we consider conditional probabilities. The conditional on w_j is

$$P(w_j | a_j, r_j, f_s, \lambda_j, z_j) \propto w_j^{n_j - 1} e^{-[(f_s + (1 - f_s)\lambda_j) z_{ja} + (1 - f_s + f_s\lambda_j) z_{jr} + z_{jo}] w_j}, \tag{33}$$

which is a gamma distribution. The conditional on λ_j is

$$\lambda_j^{(z_{ja} + z_{jr}) r_j + \alpha - 1} e^{-[(1 - f_s) w_j z_{ja} + f_s w_j z_{jr} + \beta] \lambda_j}, \tag{34}$$

another gamma distribution. The conditional on f_s is

$$P(f_s | a, r, \lambda, w, z) = f_s^{\sum_{j \in s} (a_j z_{ja} + r_j z_{jr})} (1 - f_s)^{\sum_{j \in s} (a_j z_{jr} + r_j z_{ja})} e^{\sum_{j \in s} (\lambda_j w_j - 1)(z_{ja} - z_{jr}) f_s}. \tag{35}$$

Although this is not a standard distribution, it is the product of only three factors, as opposed to one factor for each $j \in s$, a great improvement. Furthermore, it is of the form $f^a (1 - f)^b e^{cf}$. It turns out that functions of this form on $[0, 1]$ can be approximated extremely well by a beta distribution[3] proportional to $f^\sigma (1 - f)^\tau$. Thus if we use this

[3] I have an interactive SAGE notebook verifying this claim for a wide range of parameters, but now that I know it empirically to be true an analytic proof shouldn't be too hard.

beta distribution as a proposal distribution for Metropolis-Hastings sampling of the conditional Eq. 35 we will obtain essentially perfect, uncorrelated samples with a single step. That is, we can achieve the ideal state of Gibbs sampling. To obtain the desired approximation, we match first and second derivatives of the log-conditional with the log of the beta approximation at the mode of the log-conditional. This can all be done analytically. The mode f_0 is given by

$$f_0 = \frac{-(a+b) + c + \sqrt{(a+b)^2 + 2(a-b)c + c^2}}{2c} \tag{36}$$

and the result of matching derivatives is

$$\sigma = a + cf_0^2,\ \tau = -cf_0^2 + 2cf_0 + b - c. \tag{37}$$

The higher-level parameters α, β, and π_o, and the indicators z_j are unaffected by this change and are sampled as before. To summarize what we have accomplished by adding auxiliary variables, we now have perfect, uncorrelated draws from the conditional distribution of every variable using standard distributions (beta, gamma, categorical), the parameters of which are obtained from simple arithmetic. The only exception is α, which we must still slice-sample, but that only occurs once per Gibbs sweep. The computational savings ought to be enormous.

C. `AllelicCapSeg` hyperparameter sampling

In general, high-level parameters, such as the hyperparameters α and β of our `AllelicCapSeg` model are hard to sample because consecutive MCMC samples tend to be highly correlated leading to low effective sample sizes even with many iterations. This is seen as follows: at any iteration we have values of α and β that define some mean (α/β^2) of the $\{\lambda_j\}$. When we Gibbs sample the $\{\lambda_j\}$, the sampling mean will then tend to be near α/β^2. Then α and β will in turn be sampled so as to reproduce this mean and the sampling gets entangled in its current value. This effect worsens as N increases.

This could be averted if we could sample from the joint conditional $P(\alpha, \beta, \lambda|w, f, z)$, thereby exactly accounting for any correlations. This is intractable, but we could obtain the same benefits by sampling α and β from the *marginalized* conditional

$$P(\alpha, \beta|w, f, z) = \int P(\alpha, \beta, \lambda|w, f, z)\, d^N\lambda. \tag{38}$$

This approach is called *collapsed Gibbs sampling.* It is not equivalent to completely marginalizing the $\{\lambda_j\}$ out of our model, because this would render the conditionals on the $\{w_j\}$ intractable. Rather, they are marginalized only for sampling the hyperparameters. This is made possible by the auxiliary hyperparameters $\{w_j\}$, which render the $\{\lambda_j\}$ conditionally gamma and thus easy to integrate out analytically.

$$P(\alpha, \beta|w, f, z) = \int P(\alpha, \beta, \lambda|w, f, z)\, d^N\lambda \tag{39}$$

$$= \prod_j \int \frac{\beta^\alpha}{\Gamma(\alpha)} \lambda_j^{\alpha_j+\alpha-1} e^{-(\beta_j+\beta)\lambda_j}\, d\lambda_j \tag{40}$$

$$= \prod_j \frac{\Gamma(\alpha_j+\alpha)}{\Gamma(\alpha)} \frac{\beta^\alpha}{(\beta_j+\beta)^{\alpha_j+\alpha}} \tag{41}$$

where $\alpha_j = (z_{ja} + z_{jr})r_j$ and $\beta_j = ((1-f_s)z_{ja} + fz_{jr})\, w_j$.

Since α and β are correlated, naively alternating Gibbs sampling of α and β may be slow. A natural choice of variables is the mean μ and standard deviation σ (of Gamma(α, β), in terms of which $\alpha = \mu^2/\sigma^2$ and $\beta = \mu/\sigma^2$. Intuitively these ought to be weakly-coupled. Empirically we find that the conditionals on μ and σ are always unimodal[4] and thus sampling is straightforward. Slice sampling is a reasonable method to use.

D. Model-comparison test for segment merging

[SL: I'll fill this in once it's worked out.]

[4] For μ this is actually pretty simple to show analytically.

Appendix A: Finding Tight Lower Bounds on Convex and Non-convex Functions

Given a function $f(x)$ and an arbitrary value x_0, we seek a lower bound $g(x) \leq f(x)$ that is tight at x_0, that is, $g(x_0) = f(x_0)$. If $f(x)$ is convex, that is, if $f'(x)$ is non-decreasing, the linearization $g(x) = f(x_0) + f'(x_0)(x - x_0)$ is such a bound. More generally, suppose $h(x)f'(x)$ is non-decreasing. Instead of approximating $f'(x) \approx f'(x_0)$, perhaps we can approximate $h(x)f'(x) \approx h(x_0)f'(x_0)$ via a candidate lower bound $g(x)$ for which

$$g'(x) = \frac{h(x_0)f'(x_0)}{h(x)} \tag{A1}$$

$$g(x) = f(x_0) + h(x_0)f'(x_0)\int_{x_0}^{x} \frac{dt}{h(t)} \tag{A2}$$

Lemma 1. *Such a function $g(x)$ is a tight lower bound on $f(x)$ if $h(x)f'(x)$ is non-decreasing for some non-negative function $h(x)$.*

Proof. By the Fundamental Theorem of Calculus,

$$f(x) - g(x) = \int_{x_0}^{x} \left(f'(t) - g'(t)\right) dt \tag{A3}$$

$$= \int_{x_0}^{x} \frac{h(t)f'(t) - h(x_0)f'(x_0)}{h(t)}\, dt \tag{A4}$$

By the monotonicity of $h(x)$, the integral is non-positive for $x > x_0$ and non-negative for $x < x_0$. Either way, the resulting integral is negative, so $g(x) \leq f(x)$. □

For Dirichlet-multinomial inference, we use the special case $h(x) = x$.

Corollary 1. *If $xf'(x)$ is non-decreasing for $x > 0$ then for any $x_0 > 0$*

$$f(x) \geq f(x_0) + x_0 f'(x_0)\left(\log(x) - \log(x_0)\right) \tag{A5}$$

Two important cases are $f_1(x) = \log\Gamma(x)/\Gamma(x+n)$ and $f_2(x) = \log\Gamma(x+n)/\Gamma(x) = -f_1(x)$, where n is a whole number. Using the recursive identity $\Gamma(n+1) = n\Gamma(n)$ we have

$$f_1(x) = -\sum_{k=0}^{n-1}\log(x+k), \quad f_1'(x) = -\sum_{k=0}^{n-1}\frac{1}{x+k}, \quad f_1''(x) = \sum_{k=0}^{n-1}\frac{1}{(x+k)^2} \tag{A6}$$

from which we see that $f_1(x)$ is convex and the usual linearization bound holds

$$\log\frac{\Gamma(x)}{\Gamma(x+n)} \geq \log\frac{\Gamma(x_0)}{\Gamma(x_0+n)} - \left(\psi(x_0+n) - \psi(x_0)\right)(x - x_0) \tag{A7}$$

where $\psi(x) = \frac{d}{dx}\log\Gamma(x)$ is the digamma function. As for $f_2(x) = -f_1(x)$, it is not convex, but

$$xf_2'(x) = \sum_{k=0}^{n-1}\frac{x}{x+k} = \sum_{k=0}^{n-1}\frac{1}{1+k/x}, \tag{A8}$$

which is increasing since each denominator is decreasing. Thus we apply the above corollary to obtain

$$\log\frac{\Gamma(x+n)}{\Gamma(x)} \geq \log\frac{\Gamma(x_0+n)}{\Gamma(x_0)} + x_0\left(\psi(x_0+n) - \psi(x_0)\right)\left(\log(x) - \log(x_0)\right) \tag{A9}$$