Estimating a Dirichlet distribution

Thomas P. Minka

2000 (revised 2003, 2009)

Abstract

The Dirichlet distribution and its compound variant, the Dirichlet-multinomial, are two of the most basic models for proportional data, such as the mix of vocabulary words in a text document. Yet the maximum-likelihood estimate of these distributions is not available in closed-form. This paper describes simple and efficient iterative schemes for obtaining parameter estimates in these models. In each case, a fixed-point iteration and a Newton-Raphson (or generalized Newton-Raphson) iteration is provided.

1 The Dirichlet distribution

The Dirichlet distribution is a model of how proportions vary. Let $\mathbf{p}$ denote a random vector whose elements sum to 1, so that p_k represents the proportion of item k. Under the Dirichlet model with parameter vector $\boldsymbol{\alpha}$, the probability density at $\mathbf{p}$ is

$$\begin{aligned} p(\mathbf{p}) &\sim \mathcal{D}(\alpha_1, ..., \alpha_K) = \frac{\Gamma(\sum_k \alpha_k)}{\prod_k \Gamma(\alpha_k)} \prod_k p_k^{\alpha_k - 1} && (1) \\ \text{where } p_k &> 0 && (2) \\ \sum_k p_k &= 1 && (3) \end{aligned}$$

The parameters $\boldsymbol{\alpha}$ can be estimated from a training set of proportions: $D = \{\mathbf{p}_1, ..., \mathbf{p}_N\}$. The maximum-likelihood estimate of $\boldsymbol{\alpha}$ maximizes $p(D|\boldsymbol{\alpha}) = \prod_i p(\mathbf{p}_i|\boldsymbol{\alpha})$. The log-likelihood can be written

$$\begin{aligned} \log p(D|\boldsymbol{\alpha}) &= N \log \Gamma(\sum_k \alpha_k) - N \sum_k \log \Gamma(\alpha_k) + N \sum_k (\alpha_k - 1) \log \bar{p}_k && (4) \\ \text{where } \log \bar{p}_k &= \frac{1}{N} \sum_i \log p_{ik} && (5) \end{aligned}$$

This objective is convex in $\boldsymbol{\alpha}$ since Dirichlet is in the exponential family. This implies that the likelihood is unimodal and the maximum can be found by a simple search. A direct convexity proof has also been given by Ronning (1989). The gradient of the log-likelihood with respect to one α_k is

$$\begin{aligned} g_k &= \frac{d \log p(D|\boldsymbol{\alpha})}{d\alpha_k} = N\Psi(\sum_k \alpha_k) - N\Psi(\alpha_k) + N \log \bar{p}_k && (6) \\ \Psi(x) &= \frac{d \log \Gamma(x)}{dx} && (7) \end{aligned}$$

Ψ is known as the digamma function and is similar to the natural logarithm. As always with the exponential family, when the gradient is zero, the expected sufficient statistics are equal to the observed sufficient statistics. In this case, the expected sufficient statistics are

$$E[\log p_k] = \Psi(\alpha_k) - \Psi(\sum_k \alpha_k) \qquad (8)$$

and the observed sufficient statistics are $\log \bar{p}_k$.

A fixed-point iteration for maximizing the likelihood can be derived as follows. Given an initial guess for $\boldsymbol{\alpha}$, we construct a simple lower bound on the likelihood which is tight at $\boldsymbol{\alpha}$. The maximum of this bound is computed in closed-form and it becomes the new guess. Such an iteration is guaranteed to converge to a stationary point of the likelihood—in fact it is the same principle behind the EM algorithm (Minka, 1998). For the Dirichlet, the maximum is the only stationary point.

As shown in appendix A, a bound on $\Gamma(\sum_k \alpha_k)$ leads to the following fixed-point iteration:

$$\Psi(\alpha_k^{new}) = \Psi(\sum_k \alpha_k^{old}) + \log \bar{p}_k \tag{9}$$

This algorithm requires inverting the Ψ function—a procedure which is described in appendix C.

Another approach to finding a stationary point is Newton iteration. The second-derivatives, i.e. Hessian matrix, of the log-likelihood are given by

$$\begin{aligned} \frac{d \log p(D|\alpha)}{d\alpha_k^2} &= N\Psi'(\sum_k \alpha_k) - N\Psi'(\alpha_k) && (10) \\ \frac{d \log p(D|\alpha)}{d\alpha_k d\alpha_j} &= N\Psi'(\sum_k \alpha_k) \qquad (k \neq j) && (11) \end{aligned}$$

Ψ' is known as the trigamma function. The Hessian can be written in matrix form as

$$\begin{aligned} \mathbf{H} &= \mathbf{Q} + \mathbf{1}\mathbf{1}^{\mathrm{T}} z && (12) \\ q_{jk} &= -N\Psi'(\alpha_k)\delta(j-k) && (13) \\ z &= N\Psi'(\sum_k \alpha_k) && (14) \end{aligned}$$

One Newton step is therefore

$$\begin{aligned} \alpha^{\text{new}} &= \alpha^{\text{old}} - \mathbf{H}^{-1}\mathbf{g} && (15) \\ \mathbf{H}^{-1} &= \mathbf{Q}^{-1} - \frac{\mathbf{Q}^{-1}\mathbf{1}\mathbf{1}^{\mathrm{T}}\mathbf{Q}^{-1}}{1/z + \mathbf{1}^{\mathrm{T}}\mathbf{Q}^{-1}\mathbf{1}} && (16) \\ (\mathbf{H}^{-1}\mathbf{g})_k &= \frac{g_k - b}{q_{kk}} && (17) \\ \text{where } b &= \frac{\mathbf{1}^{\mathrm{T}}\mathbf{Q}^{-1}\mathbf{g}}{1/z + \mathbf{1}^{\mathrm{T}}\mathbf{Q}^{-1}\mathbf{1}} = \frac{\sum_j g_j/q_{jj}}{1/z + \sum_j 1/q_{jj}} && (18) \end{aligned}$$

Unlike some Newton algorithms, this one does not require storing or inverting the Hessian matrix explicitly. The same Newton algorithm was given by Ronning (1989) and by Naryanan (1991). Naryanan also derives a stopping rule for the iteration.

An approximate MLE, useful for initialization, is given by finding the density which matches the moments of the data. The first two moments of the density are

$$\begin{aligned} E[p_k] &= \frac{\alpha_k}{\sum_k \alpha_k} && (19) \\ E[p_k^2] &= E[p_k]\frac{1+\alpha_k}{1+\sum_k \alpha_k} && (20) \\ \sum_k \alpha_k &= \frac{E[p_1] - E[p_1^2]}{E[p_1^2] - E[p_1]^2} && (21) \end{aligned}$$

Multiplying (21) and (19) gives a formula for α_k in terms of moments. Equation (21) uses p_1, but any other p_k could also be used to estimate $\sum_k \alpha_k$. Ronning (1989) suggests instead using all of the p_k's via

$$\operatorname{var}(p_k) = \frac{E[p_k](1 - E[p_k])}{1 + \sum_k \alpha_k} \tag{22}$$

$$\log \sum_k \alpha_k = \frac{1}{K-1} \sum_{k=1}^{K-1} \log \left(\frac{E[p_k](1 - E[p_k])}{\operatorname{var}(p_k)} - 1 \right) \tag{23}$$

Another approximate MLE, specifically for the case $K = 2$, is given by Johnson & Kotz (1970):

$$\alpha_1 = \frac{1}{2} \frac{1 - \bar{p}_2}{1 - \bar{p}_1 - \bar{p}_2} \tag{24}$$

$$\alpha_2 = \frac{1}{2} \frac{1 - \bar{p}_1}{1 - \bar{p}_1 - \bar{p}_2} \tag{25}$$

2 Estimating Dirichlet mean and precision separately

The $\boldsymbol{\alpha}$ parameters of the Dirichlet can be understood by considering the following alternative representation:

$$s = \sum_k \alpha_k \tag{26}$$

$$\mathbf{m} = E[p_k] = \boldsymbol{\alpha}/s \tag{27}$$

Here $\mathbf{m}$ is the mean of the distribution for $\mathbf{p}$ and s can be understood as the *precision.* When s is large, $\mathbf{p}$ is likely to be near $\mathbf{m}$, and when s is small, $\mathbf{p}$ is distributed more diffusely. Interpretation of s and $\mathbf{m}$ suggests situations, such as hierarchical modeling, in which we may want to fix one parameter and only optimize the other. Additionally, s and $\mathbf{m}$ are roughly decoupled in the maximum-likelihood objective, which means we can get simplifications and speedups by optimizing them alternately. Thus, in this section we will reparameterize the distribution with $(s, \mathbf{m})$ where

$$\alpha_k = s m_k \tag{28}$$

$$\sum_k m_k = 1 \tag{29}$$

2.1 Estimating Dirichlet precision

The likelihood for s alone is

$$p(D|s) \propto \left(\frac{\Gamma(s) \exp(s \sum_k m_k \log \bar{p}_k)}{\prod_k \Gamma(s m_k)} \right)^N \tag{30}$$

whose derivatives are

$$\frac{d \log p(D|s)}{ds} = N\Psi(s) - N \sum_k m_k \Psi(s m_k) + N \sum_k m_k \log \bar{p}_k \tag{31}$$

$$\frac{d^2 \log p(D|s)}{ds^2} = N\Psi'(s) - N \sum_k m_k^2 \Psi'(s m_k) \tag{32}$$

A convergent fixed-point iteration for s is

$$1/s^{new} = 1/s - \Psi(s) + \sum_k m_k \Psi(s m_k) - \sum_k m_k \log \bar{p}_k \tag{33}$$

Proof Use the bound

$$\begin{aligned} \frac{\Gamma(s)}{\prod_k \Gamma(s m_k)} &\geq \exp(sb + \log(s) + c) \quad (34) \\ b &= \Psi(\hat{s}) - \sum_k m_k \Psi(\hat{s} m_k) - 1/\hat{s} \quad (35) \end{aligned}$$

to get

$$\log p(D|s) \geq s \sum_k m_k \log \bar{p}_k + sb + \log(s) + (\text{const.}) \tag{36}$$

from which (33) follows.

This iteration is only first-order convergent because the bound only matches the first derivative of the likelihood. We can derive a second-order method using the technique of generalized Newton iteration Minka (2000). The idea is to approximate the likelihood by a simpler function, by matching the first two derivatives at the current guess:

$$\begin{aligned} \frac{\Gamma(s)}{\prod_k \Gamma(s m_k)} &\approx \exp(sb + a\log(s) + c) \quad (37) \\ a &= -\hat{s}^2(\Psi'(\hat{s}) - \sum_k m_k^2 \Psi'(\hat{s} m_k)) \quad (38) \\ b &= \Psi(\hat{s}) - \sum_k m_k \Psi(\hat{s} m_k) - a/\hat{s} \quad (39) \end{aligned}$$

Maximizing the approximation leads to the update

$$\frac{1}{s^{new}} = \frac{1}{s} + \frac{1}{s^2}\left(\frac{d^2 \log p(D|s)}{ds^2}\right)^{-1}\left(\frac{d \log p(D|s)}{ds}\right) \tag{40}$$

This update resembles Newton-Raphson, but converges faster.

For initialization of s, it is useful to derive a closed-form approximate MLE. Stirling's approximation to Γ gives

$$\begin{aligned} \frac{\Gamma(s)\exp(s\sum_k m_k \log \bar{p}_k)}{\prod_k \Gamma(s m_k)} &\approx \left(\frac{s}{2\pi}\right)^{(k-1)/2} \prod_k m_k^{1/2} \exp(s \sum_k m_k \log \frac{\bar{p}_k}{m_k}) \quad (41) \\ \hat{s} &\approx \frac{(k-1)/2}{-\sum_k m_k \log \frac{\bar{p}_k}{m_k}} \quad (42) \end{aligned}$$

2.2 Estimating Dirichlet mean

Now suppose we fix the precision s and want to estimate the mean $\mathbf{m}$. The likelihood for $\mathbf{m}$ alone is

$$p(D|\mathbf{m}) \propto \left(\prod_k \frac{\exp(s m_k \log \bar{p}_k)}{\Gamma(s m_k)}\right)^N \tag{43}$$

Reparameterize with the unconstrained vector $\mathbf{z}$, to get the gradient:

$$
\begin{aligned}
m_k &= \frac{z_k}{\sum_k z_k} && (44)\\
\frac{d \log p(D|\mathbf{m})}{dz_k} &= \frac{Ns}{\sum_k z_k}\left(\log \bar{p}_k - \Psi(sm_k) - \sum_k m_k \left(\log \bar{p}_k - \Psi(sm_k)\right)\right) && (45)
\end{aligned}
$$

The MLE can be computed by the fixed-point iteration

$$
\begin{aligned}
\Psi(\alpha_k) &= \log \bar{p}_k - \sum_k m_k^{old} \left(\log \bar{p}_k - \Psi(sm_k^{old})\right) && (46)\\
m_k^{new} &= \frac{\alpha_k}{\sum_k \alpha_k} && (47)
\end{aligned}
$$

This update converges very quickly.

3 The Dirichlet-multinomial/Polya distribution

The Dirichlet-multinomial distribution is a compound distribution where $\mathbf{p}$ is drawn from a Dirichlet and then a sample of discrete outcomes $\mathbf{x}$ is drawn from a multinomial with probability vector $\mathbf{p}$. This compounding is essentially a Polya urn scheme, so the Dirichlet-multinomial is also called the *Polya distribution*. Let n_k be the number of times the outcome was k, i.e.

$$
n_k = \sum_j \delta(x_j - k) \qquad (48)
$$

Then the resulting distribution over $\mathbf{x}$, a vector of outcomes, is

$$
\begin{aligned}
p(\mathbf{x}|\boldsymbol{\alpha}) &= \int_{\mathbf{p}} p(\mathbf{x}|\mathbf{p})p(\mathbf{p}|\boldsymbol{\alpha})d\mathbf{p} && (49)\\
&= \frac{\Gamma(\sum_k \alpha_k)}{\Gamma(\sum_k n_k + \alpha_k)} \prod_k \frac{\Gamma(n_k + \alpha_k)}{\Gamma(\alpha_k)} && (50)
\end{aligned}
$$

This distribution is also parameterized by $\boldsymbol{\alpha}$, which can be estimated from a training set of count vectors: $D = \{\mathbf{x}_1, ..., \mathbf{x}_N\}$. The likelihood is

$$
\begin{aligned}
n_i &= \sum_k n_{ik} && (51)\\
p(D|\alpha) &= \prod_i p(\mathbf{x}_i|\alpha) && (52)\\
&= \prod_i \left(\frac{\Gamma(\sum_k \alpha_k)}{\Gamma(n_i + \sum_k \alpha_k)} \prod_k \frac{\Gamma(n_{ik} + \alpha_k)}{\Gamma(\alpha_k)} \right) && (53)
\end{aligned}
$$

The gradient of the log-likelihood is

$$
g_k = \frac{d \log p(D|\alpha)}{d\alpha_k} = \sum_i \Psi(\sum_k \alpha_k) - \Psi(n_i + \sum_k \alpha_k) + \Psi(n_{ik} + \alpha_k) - \Psi(\alpha_k) \qquad (54)
$$

The maximum can be computed via the fixed-point iteration

$$
\alpha_k^{new} = \alpha_k \frac{\sum_i \Psi(n_{ik} + \alpha_k) - \Psi(\alpha_k)}{\sum_i \Psi(n_i + \sum_k \alpha_k) - \Psi(\sum_k \alpha_k)} \qquad (55)
$$

(see appendix B).

Alternatively, there is a simplified Newton iteration as in the Dirichlet case. The Hessian of the log-likelihood is

$$\begin{aligned}\frac{d\log p(D|\alpha)}{d\alpha_k^2} &= \sum_i \Psi'(\sum_k \alpha_k) - \Psi'(n_i + \sum_k \alpha_k) + \Psi'(n_{ik} + \alpha_k) - \Psi'(\alpha_k) &(56)\\ \frac{d\log p(D|\alpha)}{d\alpha_k d\alpha_j} &= \sum_i \Psi'(\sum_k \alpha_k) - \Psi'(n_i + \sum_k \alpha_k) \qquad (k \neq j) &(57)\end{aligned}$$

The Hessian can be written in matrix form as

$$\begin{aligned}\mathbf{H} &= \mathbf{Q} + \mathbf{1}\mathbf{1}^{\mathrm{T}} z &(58)\\ q_{jk} &= \delta(j-k)\sum_i \Psi'(n_{ik} + \alpha_k) - \Psi'(\alpha_k) &(59)\\ z &= \sum_i \Psi'(\sum_k \alpha_k) - \Psi'(n_i + \sum_k \alpha_k) &(60)\end{aligned}$$

from which a Newton step can be computed as before. The search can be initialized with the moment matching estimate where p_{ik} is approximated by n_{ik}/n_i.

Another approach is to reduce this problem to the previous one via EM; see appendix D.

A different method is to maximize the leave-one-out (LOO) likelihood instead of the true likelihood. The LOO likelihood is the product of the probability of each sample given the remaining data and the parameters. The LOO log-likelihood is

$$f(\alpha) = \sum_{ik} n_{ik} \log\left(\frac{n_{ik} - 1 + \alpha_k}{n_i - 1 + \sum_k \alpha_k}\right) = \sum_{ik} n_{ik} \log(n_{ik} - 1 + \alpha_k) - \sum_i n_i \log(n_i - 1 + \sum_k \alpha_k) \quad (61)$$

Note that it doesn't involve any special functions. The derivatives are

$$\begin{aligned}\frac{df(\alpha)}{d\alpha_k} &= \sum_i \frac{n_{ik}}{n_{ik} - 1 + \alpha_k} - \frac{n_i}{n_i - 1 + \sum_k \alpha_k} &(62)\\ \frac{df(\alpha)}{d\alpha_k^2} &= \sum_i -\frac{n_{ik}}{(n_{ik} - 1 + \alpha_k)^2} + \frac{n_i}{(n_i - 1 + \sum_k \alpha_k)^2} &(63)\\ \frac{df(\alpha)}{d\alpha_k \alpha_j} &= \sum_i \frac{n_i}{(n_i - 1 + \sum_k \alpha_k)^2} &(64)\end{aligned}$$

A convergent fixed-point iteration is

$$\alpha_k^{new} = \alpha_k \frac{\sum_i \frac{n_{ik}}{n_{ik} - 1 + \alpha_k}}{\sum_i \frac{n_i}{n_i - 1 + \sum_k \alpha_k}} \quad (65)$$

Proof Use the bounds

$$\begin{aligned}\log(n + x) &\geq q \log x + (1-q)\log n - q \log q - (1-q)\log(1-q) &(66)\\ q &= \frac{\hat{x}}{n + \hat{x}} &(67)\\ \log(x) &\leq ax - 1 + \log \hat{x} &(68)\\ a &= 1/\hat{x} &(69)\end{aligned}$$

to get

$$f(\alpha) \geq \sum_i n_{ik} q_{ik} \log \alpha_k - n_i a_i \sum_k \alpha_k + (\text{const.}) \tag{70}$$

leading to (65).

The LOO likelihood can be interpreted as the approximation

$$\frac{\Gamma(x+n)}{\Gamma(x)} \approx (x+n-1)^n \tag{71}$$

4 Estimating Polya mean and precision separately

The $\boldsymbol{\alpha}$ parameters of the Polya distribution can be decomposed into mean $\mathbf{m}$ and precision s, just as in the Dirichlet case, and optimization can be done separately on each part. The decomposition also leads to an interesting interpretation of the Polya, discussed in the next subsection.

4.1 A novel interpretation of the Polya distribution

The Polya distribution can be interpreted as a multinomial distribution over a modified set of counts, with a special normalizer. To see why, consider the log-probability of the outcomes $\mathbf{x}$ under the Polya versus multinomial:

$$\begin{aligned} \log p(\mathbf{x}|\boldsymbol{\alpha}) &= \log \Gamma(s) - \log \Gamma(s+n) + \sum_k \log \Gamma(n_k + s m_k) - \log \Gamma(s m_k) && (72) \\ \log p(\mathbf{x}|\mathbf{p}) &= \sum_k n_k \log p_k && (73) \end{aligned}$$

The multinomial is an exponential family, and the Polya is not. But we can find an approximate exponential family representation of the Polya, by considering derivatives. In the multinomial case, the counts can be recovered from the expression

$$n_k = p_k \frac{d \log p(\mathbf{x}|\mathbf{p})}{d p_k} \tag{74}$$

In the Polya case, the analogous expression is

$$\begin{aligned} \tilde{n}_k &= m_k \frac{d \log p(\mathbf{x}|\boldsymbol{\alpha})}{d m_k} && (75) \\ &= \alpha_k \left(\Psi(n_k + \alpha_k) - \Psi(\alpha_k) \right) \equiv \nu(n_k, \alpha_k) && (76) \end{aligned}$$

The log-probability of $\mathbf{x}$ under the Polya can thus be approximated by

$$\log p(\mathbf{x}|\boldsymbol{\alpha}) = \sum_k \tilde{n}_k \log m_k \tag{77}$$

When $s \to \infty$, the Dirichlet-multinomial becomes an ordinary multinomial with $\mathbf{p} = \mathbf{m}$, and therefore the ‘effective’ counts are the same as the ordinary counts:

$$\nu(n_k, \infty) = n_k \tag{78}$$

At the other extreme, as $s \to 0$, the Dirichlet-multinomial favors extreme proportions, and the effective counts are a binarized version of the original counts:

$$\nu(n_k, 0) = \begin{cases} 0 & \text{if } n_k = 0, \\ 1 & \text{if } n_k > 0. \end{cases} \tag{79}$$

For intermediate values of α, the mapping ν behaves like a logarithm, reducing the influence of large counts on the likelihood (see figure 1). Thus the Polya can be understood as a multinomial with 'damped' counts.

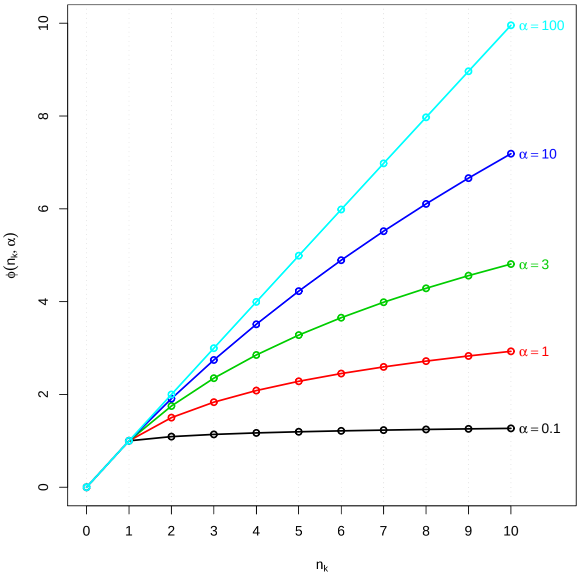

Figure 1: The 'effective' counts for the Polya distribution, as a function of the original count n_k and the parameter α_k.

This representation of the Polya also arises in the estimation of $\mathbf{m}$ when s is fixed (section 5).

4.2 Estimating Polya precision

The likelihood for s alone is

$$p(D|s) \propto \prod_i \left(\frac{\Gamma(s)}{\Gamma(n_i + s)} \prod_k \frac{\Gamma(n_{ik} + sm_k)}{\Gamma(sm_k)} \right) \tag{80}$$

The derivatives are

$$\frac{d \log p(D|s)}{ds} = \sum_i \Psi(s) - \Psi(n_i + s) + \sum_k m_k \Psi(n_{ik} + sm_k) - m_k \Psi(sm_k) \tag{81}$$

$$\frac{d^2 \log p(D|s)}{ds^2} = \sum_i \Psi'(s) - \Psi'(n_i + s) + \sum_k m_k^2 \Psi'(n_{ik} + sm_k) - m_k^2 \Psi'(sm_k) \tag{82}$$

A convergent fixed-point iteration is

$$s^{new} = s \frac{\sum_{ik} m_k \Psi(n_{ik} + sm_k) - m_k \Psi(sm_k)}{\sum_i \Psi(n_i + s) - \Psi(s)} \tag{83}$$

(the proof is similar to (55)). However, it is very slow. We can get a fast second-order method as follows. When s is small, i.e. the gradient is positive, use the approximation

$$\begin{aligned}\log p(D|s) &\approx a\log(s) + cs + k && (84)\\ a &= -s_0^2 f''(s_0) && (85)\\ c &= f'(s_0) - a/s_0 && (86)\end{aligned}$$

to get the update

$$s^{new} = -a/c = s/(1 + \frac{f'(s)}{sf''(s)}) \tag{87}$$

except when $c \geq 0$, in which case the solution is $s = \infty$. When s is large, i.e. the gradient is negative, use the approximation

$$\begin{aligned}\log p(D|s) &\approx \frac{a}{2s^2} + \frac{c}{s} && (88)\\ a &= s_0^3(s_0 f''(s_0) + 2f'(s_0)) && (89)\\ c &= -(s_0^2 f'(s_0) + a/s_0) && (90)\end{aligned}$$

to get the update

$$s^{new} = -a/c = s - \frac{f'(s)}{f''(s) + 3f'(s)/s} \tag{91}$$

For large s, the value of a tends to be numerically unstable. If $sf''(s) + 2f'(s)$ is within machine epsilon, then it is better to substitute the limiting value:

$$a \to \sum_i \frac{n_i(n_i-1)(2n_i-1)}{6} - \sum_{ik} \frac{n_{ik}(n_{ik}-1)(2n_{ik}-1)}{6m_k^2} \tag{92}$$

A even faster update for large s is possible by using a richer approximation:

$$\begin{aligned}\log p(D|s) &\approx c\log\left(\frac{s}{s+b}\right) + \frac{e}{s+b} && (93)\\ c &= \sum_{ik}\delta(n_{ik} > 0) - \sum_i \delta(n_i > 0) && (94)\\ e &= -\frac{s_0+b}{s_0}(s_0(s_0+b)f'(s_0) - cb) && (95)\\ b &= \mathrm{RootOf}(a_2b^2 + a_1b + a_0) && (96)\\ a_2 &= s_0^3(s_0 f''(s_0) + 2f'(s_0)) && (97)\\ a_1 &= 2s_0^2(s_0 f''(s_0) + f'(s_0)) && (98)\\ a_0 &= s_0^2 f''(s_0) + c && (99)\end{aligned}$$

The approximation comes from setting $c\log s$ to match $\log p(D|s)$ as $s \to 0$ and then choosing (b, e) to match the first two derivatives of f at the current s. The resulting update is

$$s^{new} = \frac{cb^2}{e - cb} = \left(\frac{1}{s} - \frac{f'(s)(s+b)^2}{cb^2}\right)^{-1} \tag{100}$$

Note that a_2 is equivalent to a above and should be corrected for stability via the same method.

The case of large dimension

An interesting special case arises when K is very large. The precision can be estimated simply by counting the number of *singleton* elements in each $\mathbf{x}$. Because the precision acts like a smoothing parameter on the estimate of $\mathbf{m}$, this result is reminiscent of smoothing methods in document modeling which are based on counting singletons.

If m_k is roughly uniform and $K >> 1$, then $\alpha_k << 1$ and we can use the approximations

$$\begin{aligned}
\Gamma(n_k + \alpha_k) &\approx \Gamma(n_k) && (101)\\
\Gamma(\alpha_k) &\approx 1/\alpha_k && (102)\\
p(\mathbf{x}|s) &\approx \frac{\Gamma(s)}{\Gamma(n+s)} \prod_{n_k>0} s m_k \Gamma(n_k) && (103)\\
&\propto \frac{\Gamma(s)s^{\hat{K}}}{\Gamma(n+s)} && (104)
\end{aligned}$$

where $\hat{K}$ is the number of unique observations in $\mathbf{x}$. The approximation does not hold if s is large, which can happen when $\mathbf{m}$ is a good match to the data. But if the dimensionality is large enough, the data will be too sparse for this to happen. The derivatives become

$$\begin{aligned}
\frac{d \log p(D|s)}{ds} &\approx \sum_i \Psi(s) - \Psi(n_i + s) + \hat{K}_i/s && (105)\\
\frac{d^2 \log p(D|s)}{ds^2} &\approx \sum_i \Psi'(s) - \Psi'(n_i + s) - \hat{K}_i/s^2 && (106)
\end{aligned}$$

Newton iteration can be used as long as the maximum for s is not on the boundary of $(0, \infty)$. These cases occur when $\hat{K} = 1$ and $\hat{K} = n$.

When the gradient is zero, we have

$$\hat{K} = s(\Psi(n+s) - \Psi(s)) = E[K|s,n] \tag{107}$$

A convergent fixed-point iteration is

$$s^{new} = \frac{\sum_i \hat{K}_i}{\sum_i \Psi(n_i + s^{old}) - \Psi(s^{old})} \tag{108}$$

Proof Use the bound

$$\begin{aligned}
\frac{\Gamma(s)}{\Gamma(n+s)} &\geq \frac{\Gamma(\hat{s})\exp((\hat{s}-s)b)}{\Gamma(n+\hat{s})} && (109)\\
b &= \Psi(n+\hat{s}) - \Psi(\hat{s}) && (110)
\end{aligned}$$

to get

$$p(D|s) \geq -s \sum_i b_i + \sum_i \hat{K}_i \log s + (\text{const.}) \tag{111}$$

leading to (108).

Applying the large K approximation to the LOO likelihood gives

$$\begin{aligned}
t &= \sum_{ik} \delta(n_{ik} - 1) \qquad \text{(number of singletons)} && (112)\\
f(s) &= t \log s - \sum_i n_i \log(n_i - 1 + s) && (113)\\
\frac{df(s)}{ds} &= \frac{t}{s} - \sum_i \frac{n_i}{n_i - 1 + s} && (114)
\end{aligned}$$

For $N = 1$:

$$\begin{aligned} s &= \frac{t(n-1)}{n-t} && (115) \\ \frac{s}{s+n} &= \frac{t(n-1)}{n^2-t} \approx \frac{t}{n} && (116) \end{aligned}$$

which is the result we wanted.

5 Estimating Polya mean

The likelihood for $\mathbf{m}$ only is

$$p(D|\mathbf{m}) \propto \prod_{ik} \frac{\Gamma(n_{ik} + sm_k)}{\Gamma(sm_k)} \tag{117}$$

The maximum can be computed by the fixed-point iteration

$$m_k^{new} \propto \sum_i \nu(n_{ik}, sm_k) \tag{118}$$

where ν is defined in (76). This update can be understood intuitively as the maximum-likelihood estimate of a multinomial distribution from effective counts $\tilde{n}_{ik} = \nu(n_{ik}, sm_k)$. The proof of this iteration is similar to (55).

For a Newton-Raphson iteration, reparameterize to get

$$\begin{aligned} m_K &= 1 - \sum_{k=1}^{K-1} m_k && (119) \\ g_k = \frac{d \log p(D|\mathbf{m})}{dm_k} &= s \sum_i \Psi(n_{ik} + sm_k) - \Psi(sm_k) - \Psi(n_{iK} + sm_K) + \Psi(sm_K) && (120) \\ \frac{d^2 \log p(D|\mathbf{m})}{dm_k^2} &= s^2 \sum_i \Psi'(n_{ik} + sm_k) - \Psi'(sm_k) + \Psi'(n_{iK} + sm_K) - \Psi'(sm_K) && (121) \\ \frac{d^2 \log p(D|\mathbf{m})}{dm_k m_j} &= s^2 \sum_i \Psi'(n_{iK} + sm_K) - \Psi'(sm_K) && (122) \end{aligned}$$

The search should be initialized at $m_k \propto \sum_i n_{ik}$, since for large s this is the exact optimum.

References

Johnson, N. L., & Kotz, S. (1970). *Distributions in statistics: Continuous univariate distributions.* New York: Hougton Mifflin.

Minka, T. P. (1998). Expectation-Maximization as lower bound maximization. http://research.microsoft.com/~minka/papers/em.html.

Minka, T. P. (2000). Beyond newton's method. http://research.microsoft.com/~minka/papers/newton.html.

Naryanan, A. (1991). Algorithm as 266: Maximum likelihood estimation of the parameters of the dirichlet distribution. *Applied Statistics*, *40*, 365–374. `http://www.psc.edu/~burkardt/dirichlet.html`.

Ronning, G. (1989). Maximum-likelihood estimation of dirichlet distributions. *Journal of Statistical Computation and Simulation*, *32*, 215–221.

A Proof of (9)

Use the bound

$$\Gamma(x) \quad \geq \quad \Gamma(\hat{x}) \exp((x-\hat{x})\Psi(\hat{x})) \tag{123}$$

to get

$$\frac{1}{N} \log p(D|\alpha) \geq (\sum_k \alpha_k)\Psi(\sum_k \alpha_k^{old}) - \sum_k \log\Gamma(\alpha_k) + \sum_k (\alpha_k - 1)\log \bar{p}_k + (\text{const.}) \tag{124}$$

leading to (9).

B Proof of (55)

Use the bound

$$\begin{aligned} \frac{\Gamma(x)}{\Gamma(n+x)} &\geq \frac{\Gamma(\hat{x})\exp((\hat{x}-x)b)}{\Gamma(n+\hat{x})} && (125) \\ b &= \Psi(n+\hat{x}) - \Psi(\hat{x}) && (126) \end{aligned}$$

and the bound

$$\begin{aligned} \frac{\Gamma(n+x)}{\Gamma(x)} &\geq cx^a \qquad \text{if } n \geq 1 && (127) \\ a &= (\Psi(n+\hat{x}) - \Psi(\hat{x}))\hat{x} && (128) \\ c &= \frac{\Gamma(n+\hat{x})}{\Gamma(\hat{x})}\hat{x}^{-a} && (129) \end{aligned}$$

to get

$$\log p(D|\alpha) \geq -(\sum_k \alpha_k - 1)\sum_i b_i + \sum_k a_{ik} \log \alpha_k + (\text{const.}) \tag{130}$$

leading to (55).

C Inverting the Ψ function

This section describes how to compute a high-accuracy solution to

$$\Psi(x) = y \tag{131}$$

for x given y. Given a starting guess for x, Newton's method can be used to find the root of $\Psi(x) - y = 0$. The Newton update is

$$x^{new} = x^{old} - \frac{\Psi(x) - y}{\Psi'(x)} \tag{132}$$

To start the iteration, use the following asymptotic formulas for $\Psi(x)$:

$$\begin{aligned} \Psi(x) &\approx \begin{cases} \log(x - 1/2) & \text{if } x \geq 0.6 \\ -\frac{1}{x} - \gamma & \text{if } x < 0.6 \end{cases} && (133) \\ \gamma &= -\Psi(1) && (134) \end{aligned}$$

to get

$$\Psi^{-1}(y) \approx \begin{cases} \exp(y) + 1/2 & \text{if } y \geq -2.22 \\ -\frac{1}{y+\gamma} & \text{if } y < -2.22 \end{cases} \tag{135}$$

With this initialization, five Newton iterations are sufficient to reach fourteen digits of precision.

D EM for estimation from counts

Any algorithm for estimation from probability vectors can be turned into an algorithm for estimation from counts, by treating the $\mathbf{p}_i$ as hidden variables in EM. The E-step computes a posterior distribution over $\mathbf{p}_i$:

$$q(\mathbf{p}_i) \sim \mathcal{D}(n_{ik} + \alpha_k) \tag{136}$$

and the M-step maximizes

$$\begin{aligned} E[\sum_i \log p(\mathbf{p}_i|\alpha)] &= N \log \Gamma(\sum_k \alpha_k) - N \sum_k \log \Gamma(\alpha_k) + N \sum_k (\alpha_k - 1) \log \bar{p}_k && (137) \\ \text{where } \log \bar{p}_k &= \frac{1}{N} \sum_i E[\log p_{ik}] && (138) \\ &= \frac{1}{N} \sum_i \Psi(n_{ik} + \alpha_k^{old}) - \Psi(n_i + \sum_k \alpha_k^{old}) && (139) \end{aligned}$$

This is the same optimization problem as in section 1, with a new definition for $\bar{p}$. It is not necessary or desirable to reach the exact maximum in the M-step; a single Newton step will do. The Newton step will end up using the old Hessian (10) but the new gradient (54). Compared to the exact Newton algorithm, this uses half as much computation per iteration, but usually requires more than twice the iterations.