A Fast and Numerically Robust Method for Exact Multinomial Goodness-of-Fit Test

Uri KEICH and Niranjan NAGARAJAN

Evaluating the significance of goodness-of-fits tests for multinomial data in general, and estimating the p value of the log-likelihood ratio statistic G^2 in particular, have been studied extensively in the statistical literature. The methods for computing the latter can be broadly divided into two categories: asymptotic approximations and exact methods. In many applications in bioinformatics one needs a fast algorithm to estimate vanishingly small p values which led to a renewed interest in exact methods.

We introduce a new algorithm (bagFFT) for computing the p value of G^2 which is based on the algorithm by Baglivo, Olivier, and Pagano. The original algorithm tends to suffer from large numerical errors, rendering it inappropriate for our purposes. We give theoretical and empirical evidence that by applying appropriate exponential shifts we obtain an algorithm that is: (a) as stable numerically as Hirji's network algorithm, and (b) asymptotically faster than both Baglivo et al.'s and Hirji's algorithm: $O(QKN \log N)$ versus $O(QKN^2)$, where Q is the size of the lattice, K is the number of categories, and N is the size of the sample. We conclude by showing how bagFFT can be combined with our earlier work to obtain a fast and accurate algorithm to compute the significance of biological sequence motifs.

Key Words: DFT; Entropy score; FFT; Information content; Large deviation; Likelihood ratio statistic; p value.

1. INTRODUCTION

In a review article Cressie and Read (1989) wrote:

> The importance of developing useful and appropriate statistical methods for analyzing discrete multivariate data is apparent from the enormous amount of attention this subject has commanded in the literature over the last thirty years. Central to these discussions has been Pearson's X^2 statistic and the loglikelihood ratio statistic G^2.

The methods for computing the p value of the latter statistic can be broadly divided into two categories: asymptotic approximations and exact methods. We introduce a new exact

Uri Keich is Assistant Professor, and Niranjan Nagarajan is a Ph.D. Student, Computer Science Department, Cornell University, 4130 Upson Hall, Ithaca, NY 14850 (E-mail: *keich@cs.cornell.edu* and *niranjan@cs.cornell.edu*).

Journal of Computational and Graphical Statistics, Volume 15, Number 4, Pages 779–802
DOI: 10.1198/106186006X159377

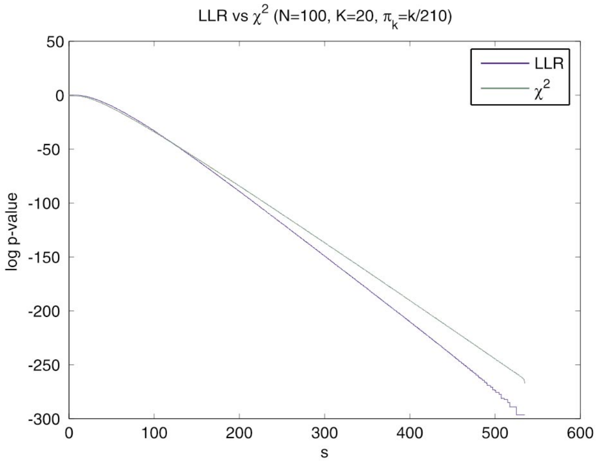


Figure 1. Inaccuracy of the χ^2 approximation.

method for estimating the p value of the G^2 statistic although our method might be applicable to Pearson's X^2 as well.

The classical approach to estimating the p value of G^2 relies on the asymptotic result $P_{H_0}(G^2 \geq s) \underset{[N\to\infty]}{\longrightarrow} \chi^2_{K-1}(s)$, where H_0 is the null multinomial distribution specified by $\pi = (\pi_1, \ldots, \pi_K)$ and N is the multinomial sample size (e.g., Cressie and Read 1989). Although the χ^2 approximation is a valid asymptotic result, in applications where N is fixed as s approaches the tail of the distribution the approximation can be quite poor. For example, as can be seen in Figure 1 for $K = 20$, $\pi_i = i/210$ and $N = 100$, the χ^2 approximation can be more than a factor of 10^{10} off in the tail of the distribution. The χ^2 approximation can be improved by adding second-order terms (Cressie and Read 1989). However, the resulting values (Siotani and Fujikoshi 1984; Cressie and Read 1984) are only accurate to $O(N^{-3/2})$ which is often significantly bigger than the p values that have to be estimated. In particular, this is typically the case for applications in bioinformatics, some of which are mentioned in Section 2.

Baglivo, Olivier, and Pagano (1992) addressed this problem by suggesting a lattice based exact method. The idea is to estimate the p value directly from the underlying multinomial distribution. More specifically, as explained in Section 3, they computed the characteristic function of a latticed version of G^2 in $O(QKN^2)$ time where Q is the size of the lattice that controls the accuracy of the estimated p value. Later Hirji (1997) proposed an algorithm based on Mehta and Patel's (1983) network algorithm. Although Hirji's, essentially branch-and-bound, algorithm can be implemented without resorting to a lattice (see also Bejerano, Friedman, and Tishby 2004), only in the latticed case is it guaranteed to have polynomial complexity. In that case Hirji's algorithm shares the same worst-case runtime as that of Baglivo et al.'s: $O(QKN^2)$. As far as the space overhead, Baglivo et al.'s algorithm is better with a space overhead of $O(Q + N)$ as opposed to $O(QN)$ for Hirji's. However, Baglivo et al.'s algorithm is prone to large numerical errors (see Section 3) which make it unusable for computing the small p values that are of most interest in this discussion, while Hirji's algorithm can be shown to be numerically stable. In this article, we present a new algorithm that yields the exact (up to lattice errors) p value of G^2 in $O(QKN \log N)$ time and $O(Q + N)$ space.

After a brief overview of applications in bioinformatics we present Baglivo et al.'s algorithm in Section 3 and, in Section 4, modify it using the shifted-FFT technique (Keich 2005) to control the numerical errors in the algorithm. This results in a $O(QKN^2)$ algorithm that can accurately compute small p values. We also present a mathematical analysis of the total roundoff error in computing the p value. We then use shifted-FFT based convolutions to reduce the runtime to $O(QKN \log N)$ and obtain the bagFFT algorithm in Section 5 (with error analysis). Both variants share Baglivo et al.'s space requirement of $O(Q + N)$. In Section 6 we present experimental results that demonstrate the reliability and improved efficiency of bagFFT in comparison to Hirji's algorithm. Finally, in Section 7 we discuss ways to combine it with the work in Keich (2005) to compute the significance of sequence motifs.

2. MOTIVATION FROM BIOINFORMATICS

In the analysis of multiple-sequence alignments one often evaluates the significance of an alignment column using a goodness-of-fit test between the column's residue distribution and a given background distribution. Commonly one computes the information content, or generalized log-likelihood ratio of the column defined as $I = G^2/2 = \sum_{j=1}^{K} n_j \log \frac{n_j/N}{\pi_j}$, where K is the size of the alphabet, n_j is the number of occurrences of the jth letter in the column, π_j is its background frequency, and N is the depth of the column. The p value of I serves as a uniform measure of the column's significance that can be compared between columns of varying sizes and background distributions. For example, Rahmann (2003) used p values to design a conservation index for alignment columns. These indices can then be used to compare and visualize (as sequence logos) the conservation profile for alignments of different sizes. In Sadreyev and Grishin (2004), a similarly defined p value is suggested as a means to detect misaligned regions in sequence alignments (among other applications). Extending this technique to distributions of residue-pairs, Bejerano, Friedman, and Tishby (2004) discussed its use for detecting correlated columns that serve as signatures for binding sites and RNA base pairs.

Motif-finding programs such as MEME (Bailey and Elkan 1994) and Consensus (Hertz and Stormo 1999) seek statistically significant (ungapped) alignments in the input sequences. These alignments are presumably the instances of the putative motif. The alignments are scored with $I_A = \sum_{j=1}^{L} I_j$, where I_j is the information content of the jth of the alignment's L columns (Stormo 2000). In order to compare two alignments of varying L and N (number of sequences in the alignment) one assumes the columns are iid and replaces I_A with its p value. One way to compute this p value is by convoluting the pmf of the individual I_j whose computation is the subject of this article. This application is studied further in Section 7.

Typically, in the applications mentioned here, there are many competing columns (or sets of columns) that need to be evaluated for their significance. The two-fold consequences are: first, to compensate for a huge number of multiple hypotheses these algorithms need to reliably compute extremely small p values corresponding to the significant and putatively more interesting columns. Second, the runtime efficiency of the algorithm is very important. Indeed, these explain the interest the bioinformatics community has shown in exact methods for computing the p value of I or, equivalently, of G^2 (e.g., Hertz and Stormo 1999; Bejerano, Friedman, and Tishby 2004; Rahmann 2003).

3. BAGLIVO ET AL.'S ALGORITHM

We begin with a formal introduction of the problem. Given a null multinomial distribution $\pi = (\pi_1, \dots, \pi_K)$ and a random sample $n = (n_1, \dots, n_K)$ of size $N = \sum n_k$ let $s = I(n) = \sum_k n_k \log \frac{n_k}{N\pi_k}$ and note that $I = G^2/2$. The p value of s is $P_{H_0}(I \geq s)$. Since for a given N and an arbitrary π the range of I can have an order of N^{K-1} distinct points, strictly exact methods are typically impractical even for moderately sized K. Thus, we are forced to map the range of I to a lattice and compute exact probabilities on the lattice. Explicitly, let $\pi_{\text{min}} = \min\{\pi_k\}$ and let $I_{\text{max}} = N \log \pi_{\text{min}}^{-1}$ be the maximal entropy value. Given the size of the lattice, Q, let $\delta = \delta(Q) = I_{\text{max}}/(Q-1)$ be the mesh size. Our surrogate for $I(n)$ is the integer valued

$$I_Q(n) = \sum_k \text{round}\left[\delta^{-1} n_k \log(n_k/(N\pi_k))\right],$$

so that $\delta I_Q \approx I$. (Note that, due to rounding effects, I_Q might be negative but we shall ignore this as the arithmetic we perform is modulo Q. The concerned reader can redefine $\delta = I_{\text{max}}/(Q - 1 - \lceil K/2 \rceil)$.) Let p_Q be the pmf of I_Q then, clearly, for any s,

$$L(s) = \sum_{j \geq \lceil s/\delta + K/2 \rceil} p_Q(j) \leq \ P(I \geq s) \ \leq \sum_{j \geq \lfloor s/\delta - K/2 \rfloor} p_Q(j) = U(s), \tag{3.1}$$

which allows us to estimate the p value and control the lattice error via adjustments to Q.

Baglivo et al. computed p_Q by inverting its characteristic function. More precisely, they computed the DFT (Discrete Fourier Transform; Press, Teukolsky, Vetterling, and Flannery 1992) of p_Q, Φ, where:

$$\Phi(l) := (Dp_Q)(l) = \sum_{j=0}^{Q-1} p_Q(j) e^{i\omega_0 jl} \quad \text{for} \quad l = 0, 1, \dots, Q-1,$$

where $\omega_0 = 2\pi/Q$ and recover p_Q by applying D^{-1}, the inverse-DFT:

$$p_Q(j) = (D^{-1}\Phi)(j) = \frac{1}{Q} \sum_{l=0}^{Q-1} \Phi(l) e^{-i\omega_0 lj}.$$

In order for this procedure to be useful, one must be able to efficiently compute Φ, keeping in mind that p_Q is unknown. Baglivo et al. accomplish this based on the observation that a multinomial distribution can be represented as the distribution of independent Poisson random variables conditioned on their sum being N. Explicitly, let $\lambda_k = N\pi_k$, that is, the mean number of occurrences of the kth letter or category, let $s_k(n_k) = \text{round}[\delta^{-1} n_k \log(n_k/\lambda_k)]$, that is, the contribution to I_Q from the kth letter appearing n_k times, let p_k denote the Poisson $\lambda = \lambda_k$ pmf, and let X_+ be a Poisson $\lambda = N$ random variable. Finally, let $\psi_{k,l}(n) = \sum_y \prod_{j=1}^k p_j(y_j) e^{il\omega_0 s_j(y_j)}$, where the sum extends over all $y \in \mathbb{Z}_+^K$ for which $\sum_{j=1}^k y_j = n$. It is not difficult to check that $\psi_{k,l}$ satisfy the following recursive formula:

$$\psi_{k,l}(n) = \sum_{x=0}^{n} p_k(x) e^{il\omega_0 s_k(x)} \psi_{k-1,l}(n-x), \tag{3.2}$$

and since as explained by Baglivo, Olivier, and Pagano (1992),

$$\Phi(l) = \frac{1}{P(X_+ = N)} \sum_{x \in \mathbb{Z}_+^K : \sum x_j = N} \prod_{j=1}^{K} p_j(x_j) e^{i\omega_0 l s_j(x_j)} = \frac{\psi_{K,l}(N)}{P(X_+ = N)},$$

$\Phi(l)$ can be recovered from (3.2) in $O(KN^2)$ steps for each l separately and hence $O(QKN^2)$ overall (to see this, note that we need to compute $\psi_{k,l}(n)$ for $k \in [1..K]$ and $n \in [0..N]$ and each computation takes $O(N)$ time). Finally, using an FFT (Fast Fourier Transform (Press et al. 1992), a fast algorithm for DFT with a runtime of $O(n \log n)$ for a vector of size n) implementation of DFT Baglivo et al. get an estimate of p_Q in an additional $O(Q \log Q)$ steps, which should typically be absorbed in the first term (as observed by Rahmann (2003), Q has to grow linearly with N in order to preserve the bound on the distance between p_Q and our real subject of interest, p_I, the pmf of I).

The algorithm as it is has a serious limitation in that the numerical errors introduced by the DFTs can readily dominate the calculations. An example of this phenomena can be observed with the parameter values, $Q = 8192$, $N = 100$, $K = 20$, and $\pi_k = 1/20$, where this algorithm yields a *negative* p value ($= -2.18 \cdot 10^{-14}$) for $P(I \geq 60)$.

4. ERROR CONTROL USING SHIFTED-FFT

The numerical instability of Baglivo et al.'s algorithm is illustrated by the following simple example. Let $p(x) = e^{-x}$ for $x \in \{0, 1, \ldots, 255\}$ and $q = \widetilde{D^{-1}}(\widetilde{D}p)$, where $\widetilde{D}$ and $\widetilde{D^{-1}}$ are the machine-implemented FFT and inverse FFT operators. As can be seen in Figure 2, while theoretically equal, in practice the two differ significantly. The analogous situation in Baglivo et al.'s algorithm is that $p = p_Q(j)$ and we compute $q = \widetilde{D^{-1}}(\widetilde{D}_{\text{bag}} p)$, where $\widetilde{D}_{\text{bag}}$ is the recursive DFT computation in the algorithm. As the example suggests we cannot compute the smaller entries of p_Q using Baglivo et al.'s algorithm. This limitation arises from the fact that we work with fixed-precision arithmetic on computers and therefore can only approximate the real arithmetic that we wish to do. For example, in the double precision arithmetic that we usually work with $1 \widetilde{+} 10^{-16} = 1$ and therefore performing a DFT on p_Q discards the information about the entries of p_Q that are smaller than $10^{-16} \cdot \max\{p_Q\}$.

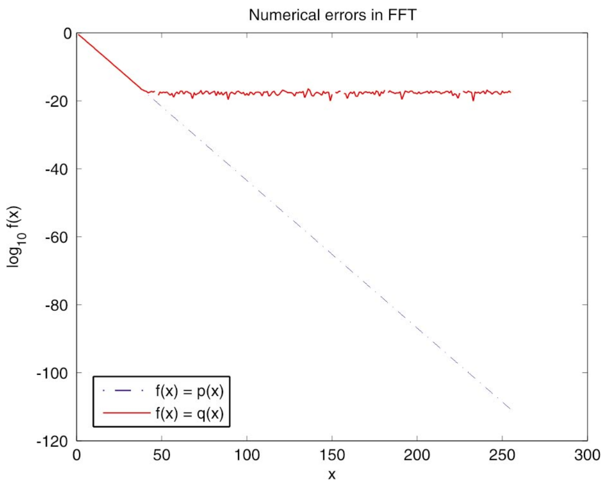


Figure 2. The destructive effects of numerical roundoff errors in FFT. This figure illustrates the potentially overwhelming effects of numerical errors in applications of FFT. $p(x) = e^{-x}$ *for* $x \in \{0, 1, \dots, 255\}$ *is compared with what should (in the absence of numerical errors) be the same quantity:* $q = \widetilde{D^{-1}}(\widetilde{D}p)$, *where* $\widetilde{D}$ *and* $\widetilde{D^{-1}}$ *are the machine implemented FFT and inverse FFT operators, respectively. This dramatic difference all but vanishes when we apply the correct exponential shift prior to applying D.*

One possible remedy for the numerical errors is to move to higher precision arithmetic. However, this only postpones the problem to smaller p values and also significantly slows down the runtime of the algorithm (due to a typical lack of hardware support for higher precision arithmetic). A better solution (in the spirit of Keich 2005) is suggested by the following extension to the example above: let $p_\theta(x) = p(x)e^{\theta x}$ and $q_\theta = \widetilde{D^{-1}}\big(\widetilde{D}p_\theta\big)$. For $\theta = 1$, we experimentally get $\max_x |\log \frac{q_\theta(x)e^{-\theta x}}{p(x)}| < 1.78 \cdot 10^{-15}$, showing that using this mode of computation we can "recover" p (from $q_\theta(x)e^{-\theta x}$) almost up to machine precision ($\varepsilon_0 \approx 2.2 \cdot 10^{-16}$). This solution is based on the intuition that by applying the correct exponential shift we "flatten" p so that the smaller values are not overwhelmed by the largest ones during the computation of the Fourier transforms.

Needless to say this exponential shift will not always work. However, the following bounds due to Hoeffding (1965) suggest that for fixed N and K, "to first order", the p values and hence p_Q behave like e^{-s}:

$$c_0 N^{-(K-1)/2} \exp(-s) \le P(I \ge s) \le \binom{N+K-1}{K-1} \exp(-s), \tag{4.1}$$

where c_0 is a positive absolute constant which can be taken to be $1/2$. This suggests that we would benefit from applying an exponential shift to p_Q. Let

$$p_\theta(j) = \frac{p_Q(j)e^{\theta\delta j}}{M(\theta)},$$

where $M(\theta) = Ee^{\theta\delta I_Q}$, the MGF (moment generating function) of δI_Q, serves to normalize p_θ and avoid numerical under/overflows. Figure 3 shows an example of the flattening effect

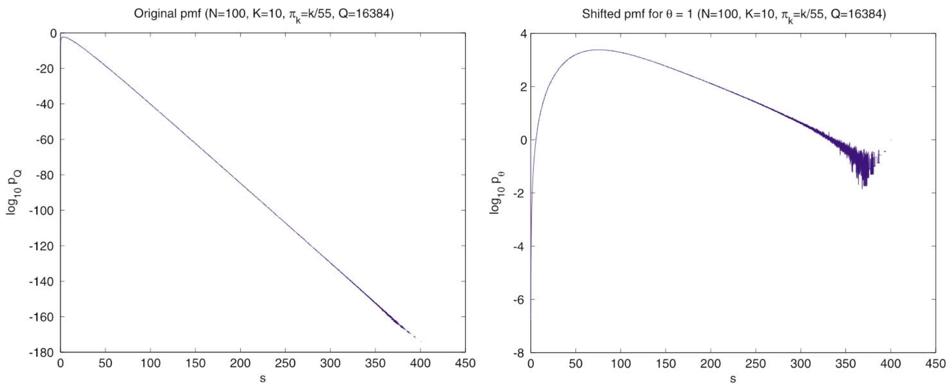


Figure 3. How can an exponential shift help? The graph on the left is that of $\log_{10} p_Q(s/\delta)$ *where N* $= 100$*, K* $= 10$*,* $\pi_k = k/55$*, and Q* $= 16{,}384$*. The graph on the right is of the log of the shifted pmf,* $\log_{10} p_\theta(s/\delta)$ *where* $\theta = 1$*. Note the dramatic flattening effect of the exponential shift (keeping in mind the fact that the scales of the* y*-axes are different).*

such a shift has on p_Q. As can be seen in the figure, the range of values in p_θ is much smaller and therefore the largest values of p_θ are no longer expected to overwhelm the smaller values (in the context of FFTs).

The discussion so far implicitly assumed that we know p_Q which of course we do not. However, we can essentially compute $\Phi_\theta = Dp_\theta$ by incorporating the shift into the recursive computation in (3.2). We do so by replacing the Poisson pmfs p_k with a shifted version

$$p_{k,\theta}(x) = p_k(x) e^{\theta \delta s_k(x)}, \tag{4.2}$$

and obtain the following recursion for $\psi_{k,l,\theta}(n) = \psi_{k,l}(n) e^{\theta \delta s_k(n)}$, the shifted version of $\psi_{k,l}(n)$:

$$\psi_{k,l,\theta}(n) = \sum_{x=0}^{n} p_{k,\theta}(x) e^{il\omega_0 s_k(x)} \psi_{k-1,l,\theta}(n-x). \tag{4.3}$$

where $\psi_{1,l,\theta}(n) = p_{1,\theta}(n)$. This allows us to compute $\widetilde{\psi}_{K,l,\theta}(N)$, an estimate of $\psi_{K,l,\theta}(N)$ (due to unavoidable roundoff errors we cannot expect to recover $\psi_{K,l,\theta}(N)$ precisely) in the same $O(KN^2)$ steps for each fixed l (due to unavoidable roundoff errors we cannot expect to recover $\psi_{K,l,\theta}(N)$ precisely). We then compute an estimate $\widetilde{p_Q}$ of p_Q based on

$$p_Q(j) = \left(D^{-1}\psi_{K,\bullet,\theta}(N)\right)(j) \frac{e^{-\theta\delta j}}{P(X_+ = N)}. \tag{4.4}$$

An additional feature of this approach (that is absent in Baglivo et al.'s algorithm) is that we can directly estimate $\log p_Q(j)$, in cases where computing $p_Q(j)$ would create an underflow. This could be important in applications where very small p values are common, for example, in a typical motif-finding situation. Finally, the p value is estimated using (3.1) (or the logarithmic version of the summation).

Remark 1. In practice, to avoid under/overflows we normalize $p_{k,\theta}(x)$ in (4.2) so that it sums up to 1. These constants are then compensated for when computing p_Q in (4.4). We ignore these factors throughout the article.

Remark 2. As pointed out by an alert referee, for computing a single p value, we can avoid inverting Φ_θ by noting that for $n \in [0..Q-1]$,

$$\begin{aligned}\sum_{j \geq n} p_Q(j) &= \sum_{j \geq n} p_\theta(j) e^{-\theta\delta j} M(\theta) = \sum_{j \geq n} e^{-\theta\delta j} M(\theta) \sum_{l=0}^{Q-1} \frac{\Phi_\theta(l) e^{-i\omega_0 l j}}{Q} \\ &= \frac{M(\theta)}{Q} \sum_{l=0}^{Q-1} \Phi_\theta(l) \frac{e^{z(l)n} - e^{z(l)Q}}{1 - e^{z(l)}},\end{aligned}$$

where $z(l) = -(\theta\delta + i\omega_0 l)$. This version of the algorithm is however only marginally more efficient while having a relative error that is more than 10 times worse, in some cases, than that for the presented algorithm (and so we do not pursue it further here).

4.1 CHOOSING θ

An obvious choice for θ that is suggested by inequality (4.1) is to set it to 1 and indeed it typically yields the widest range of j's for which $\widetilde{p_Q}(j)$ provides a "decent" approximation of $p_Q(j)$. However, for computing the p value of a given s there would typically be a better choice of θ. As we can see from Figure 4, a shift of $\theta = 1$ could lead to the loss of values in the tail of the pmf during the DFT computation. If we wish to compute a p value in this region then setting $\theta = 1$ would perform poorly. Intuitively, we wish to choose a θ to ensure that the entries of p_θ around $\lfloor s/\delta \rfloor$ are not overwhelmed during the DFT computation. The solution we adopt is borrowed from the theory of large-deviation: choose θ so as to "center" p_θ about s, or more precisely, set the mean of p_θ to s. This can be accomplished by setting θ to (Dembo and Zeitouni 1998):

$$\theta_s = \underset{\theta}{\operatorname{argmin}} \left[-\theta s + \log M(\theta)\right] \tag{4.5}$$

The minimization procedure in (4.5) can be carried out numerically (a crude approximation of θ_s would typically suffice for our purpose) by using, for example, Brent's method (Press et al. 1992). The runtime cost for this is essentially a constant factor of the cost of evaluating $M(\theta)$. The latter can be reliably estimated in $O(KN^2)$ steps by replacing $e^{il\omega_0 s_k(x)}$ with $e^{\theta\delta s_k(x)}$ in (3.2). The runtime of the shifted-FFT based algorithm is therefore still $O(QKN^2)$.

The following claim, whose technical proof is outlined in the Appendix (p. 796), allows us to gauge the magnitude of the numerical errors introduced by our algorithm.

Claim 1.

$$|\widetilde{p_Q}(j) - p_Q(j)| \leq C(KN\log N + \log Q)\varepsilon_0 e^{-\theta\delta j + \log M(\theta)} + CN\log N\, \widetilde{p_Q}(j)\varepsilon_0 + O(\varepsilon_0^2), \tag{4.6}$$

where C is a small universal constant and ε_0 is the machine precision.

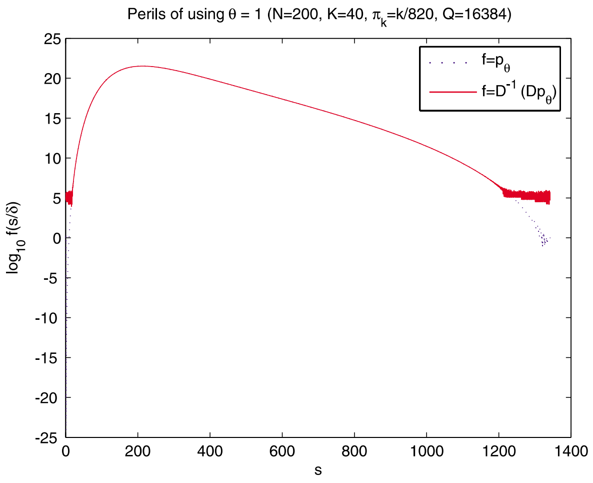


Figure 4. *Numerical errors in estimating p_θ with $\theta = 1$.*

Remark 3. The $O(\varepsilon_0^2)$ term refers to all higher order terms in an ε_0 power series expansion of the accumulated roundoff error. The bound in (4.6) is only useful when it is $\ll p_Q(j)$. In that case the propagation of roundoff errors is essentially linear and therefore the $O(\varepsilon_0^2)$ term is negligible compared to the $O(\varepsilon_0)$ term (e.g., Tasche and Zeuner 2001).

Remark 4. The Claim only holds in the absence of intermediate over/under-flows. In practice remark 1 guarantees this condition but in any case such events are detectable.

Summing over j in (4.6) yields an upper bound on the error in computing the p value. Note that if $s = \delta j$, the upper bound in Claim 1 is essentially minimized for $\theta = \theta_s$ (as the relative error term of $CN \log N \, \widetilde{p_Q}(j)\varepsilon_0$ is typically negligible), thus giving us another justification for our choice of θ. In Section 6 we show that this choice of θ works well in practice and that the theoretical error bounds there can be applied fruitfully.

5. IMPROVING THE RUNTIME

The algorithm presented in Section 4 is free of the large numerical errors that plague Baglivo et al.'s algorithm while preserving its time and space complexity. Observing that (4.3) can be expressed as a convolution between the vectors

$$p_{k,l,\theta}(x) = p_{k,\theta}(x)e^{il\omega_0 s_k(x)} \tag{5.1}$$

and $\psi_{k-1,l,\theta}$ allows us to improve the runtime of our algorithm as follows. A naively implemented convolution requires $O(N^2)$ steps and hence that factor in the overall runtime complexity. Alternatively, we can carry out an FFT-based convolution, based on the identity $\big(D(u * v)\big)(j) = (Du)(j)(Dv)(j)$ (Press et al. 1992), where $u * v$ is the convolution of the vectors u and v (a special case of the identity for the characteristic function of a sum of two independent random variables, say X and Y: $\phi_{X+Y} = \phi_X \phi_Y$). This would only require $O(N \log N)$ steps (as the FFT of a vector of size N can be computed in $O(N \log N)$ time), cutting down the overall complexity to $O(QKN\log N + Q\log Q + KN^2)$. Typically the last two terms are small compared to the runtime cost of the main loop thus giving us a $O(QKN\log N)$ algorithm.

Simply implementing (4.3) using an FFT-based convolution, however, reintroduces the severe numerical errors that were corrected for in Section 4. The following example illustrates the situation: for $\theta = 1$ one can verify that $|p_{k,l,\theta}(x)| \approx e^{-N\pi_k + x}/\sqrt{2\pi x}$. Computing $Dp_{k,l,\theta}$ therefore faces essentially the same problem as the one demonstrated in our example of FFT applied to e^{-x}. Once again the solution we propose is to apply an appropriate exponential shift: for a vector u let $u_\alpha(x) = u(x)e^{-\alpha x}$ and let $u \odot v$ denote the pointwise product of u and v, then one can readily show that

$$(u * v)_\alpha \equiv D^{-1}\left[Du_\alpha \odot Dv_\alpha\right].$$

Based on the last identity we replace the shifted convolution of (4.3) with its doubly shifted Fourier version:

$$\psi_{k,l,\theta,\theta_2}(n) = D^{-1}\left[Dp_{k,l,\theta,\theta_2} \odot D\psi_{k-1,l,\theta,\theta_2}\right](n) \qquad n = 0, 1, \ldots, N-1, \tag{5.2}$$

where

$$p_{k,l,\theta,\theta_2}(x) = p_{k,l,\theta}(x)e^{-\theta_2 x} \quad \psi_{k,l,\theta,\theta_2}(x) = \psi_{k,l,\theta}(x)e^{-\theta_2 x}.$$

One final detail is that p_{k,l,θ,θ_2} and $\psi_{k-1,l,\theta,\theta_2}$ are padded with zeros (otherwise, you get cyclic convolution; Press et al. 1992) so that they are now vectors of length $N_2 = 2N-1$ and $D = D_{N_2}$.

Analogous to (4.4) we recover p_Q from

$$p_Q(j) = \left(D^{-1}\psi_{K,\bullet,\theta,\theta_2}(N)\right)(j)\frac{e^{-\theta\delta j + \theta_2 N}}{P(X_+ = N)}, \tag{5.3}$$

and here $D^{-1} = D_Q^{-1}$.

5.1 Analysis of the Convolution Error

Let

$$\Delta_k^p = \Delta_k^p(\theta, \theta_2) = \max_l \|p_{k,l,\theta,\theta_2} - \widetilde{p}_{k,l,\theta,\theta_2}\|_2/\varepsilon_0,$$

and inductively define Δ_k^ψ as: $\Delta_1^\psi = \Delta_1^p$ and for $k = 2, \ldots, K$

$$\Delta_k^\psi = \|p_\mu\|_1\left((2C_F\log_2 N_2 + 5)\|\psi_\mu\|_2 + \Delta_{k-1}^\psi\right) + \|\psi_\mu\|_1(C_F\log_2 N_2\|p_\mu\|_2 + \Delta_k^p), \tag{5.4}$$

where μ stands for $(k, 0, \theta, \theta_2)$, and C_F is a constant < 5 that controls the l_2 norm of the numerical errors introduced by the FFT (Tasche and Zeuner 2001) (see also (A.5) p. 797).

Claim 2. *Let $\widetilde{\psi}_{k,l,\theta,\theta_2}$ denote the estimate of $\psi_{k,l,\theta,\theta_2}$ computed by (5.2). For $k = 1, \ldots, K$:*

$$\max_l \|\widetilde{\psi}_{k,l,\theta,\theta_2} - \psi_{k,l,\theta,\theta_2}\|_2 \le \Delta_k^\psi\varepsilon_0 + O(\varepsilon_0^2).$$

Remarks:

- Δ_k^p depends on the particular implementation of computing p_{k,l,θ,θ_2}. The only delicate point is when computing $\exp(il\omega_0 s_k(x))$ one should compute $ls_k(x)$ mod Q, otherwise Δ_k^p will grow linearly with Q. With this in mind, a naive computation of the other factors would result in
$$\Delta_k^p \le CN\log N\|p_{k,0,\theta,\theta_2}\|_2,$$
where C is some small constant. This can be readily proved in the spirit of the proofs in the appendix.
- Analogous to Remark 1, in practice we normalize p_{k,l,θ,θ_2} so that $\|p_{k,l,\theta,\theta_2}\|_1 = 1$. Again, we ignore this practical step in the discussion below.
- The remarks following Claim 1 are valid here as well.

Proof of Claim: By induction on k. For $k = 1$ the claim follows immediately from the definitions. For the inductive step we need the following lemma which is proved in the Appendix (p. 796).

Lemma 1. *Suppose that for $x, y, \widetilde{x}, \widetilde{y} \in \mathbb{R}^N$*
$$\|x-\widetilde{x}\|_2 \le m_x\varepsilon_0 \qquad \|y-\widetilde{y}\|_2 \le m_y\varepsilon_0.$$
Choose $N_2 \ge 2N-1$ and with $D = D_{N_2}$, the corresponding DFT operator, let
$$\tau = Dx \qquad \nu = Dy \qquad \widetilde{\tau} = \widetilde{D}\widetilde{x} \qquad \widetilde{\nu} = \widetilde{D}\widetilde{y},$$
where the vectors are padded with zeros. Then,
$$|\widetilde{D^{-1}\tilde{\tau}\,\widetilde{\odot}\,\tilde{\nu}} - D^{-1}\tau\odot\nu\|_2 \le \varepsilon_0\big[(2C_F\log_2 N_2+5)\|x\|_1\|y\|_2$$
$$+C_F\log_2 N_2\|y\|_1\|x\|_2 + \|y\|_1 m_x + \|x\|_1 m_y\big] + O(\varepsilon_0^2),$$
where $(u\odot v)(k) = u(k)v(k)$, $\widetilde{\odot}$ is the machine computation of $\odot$, and the remarks following Claim 1 are valid here as well.

Let $x = p_{k,l,\theta,\theta 2}$ and $y = \psi_{k-1,l,\theta,\theta 2}$. Clearly, $\|x-\widetilde{x}\|_2 \le \Delta_k^p\varepsilon_0$ and by the inductive hypothesis $\|y-\widetilde{y}\|_2 \le \Delta_{k-1}^{\psi}\varepsilon_0 + O(\varepsilon^2)$. The claim follows from the lemma, $\|p_{k,l,\theta,\theta 2}\|_i = \|p_{k,0,\theta,\theta 2}\|_i$ and $\|\psi_{k,l,\theta,\theta 2}\|_i \le \|\psi_{k,0,\theta,\theta 2}\|_i$, for $i = 1, 2$. □

Using the last claim, we establish in the appendix the following error bound:

Claim 3. *Let $\widetilde{p_Q}$ be computed according to (5.3). Let*
$$\Delta_{p_\theta} = \left[\frac{\Delta_K^{\psi} e^{\theta_2 N}}{M(\theta)P(X_+ = N)} + C_F\log_2 Q\right],$$
then
$$|\widetilde{p_Q}(j) - p_Q(j)| \le \varepsilon_0\Delta_{p_\theta}e^{-\theta\delta j + \log M(\theta)} + CN\log N\,\widetilde{p_Q}(j)\varepsilon_0 + O(\varepsilon_0^2), \tag{5.5}$$

Given N, K, π, Q, and s, bagFFT:

1. Computes θ by numerically solving (4.5) (using Brent's method).
2. Computes θ_2 by minimizing $\Delta_K^{\psi} e^{\theta_2 N}$ computed from (5.4) (using Brent's method).
3. For each $l = 0, 1 \ldots, Q-1$, recursively computes $\psi_{K,l,\theta,\theta_2}(N)$ using (5.2).
4. Using FFT computes $u = D^{-1}\psi_{K,\bullet,\theta,\theta_2}(N)$.
5. Computes $p_Q(j) = u(j)\frac{e^{-\theta\delta j+\theta_2 N}}{P(X_+=N)}$, or $\log p_Q(j) = \log\frac{u(j)}{P(X_+=N)} - \theta\delta j + \theta_2 N$.
6. Returns $L(s)$ and $U(s)$, computed using (3.1), as the lower and upper bounds on the p value respectively (or the logarithmic version of the sum).
7. Computes the theoretical error bounds, $E_L(s)$ and $E_U(s)$ for $L(s)$ and $U(s)$ respectively, using Corollary 1.

Figure 5. The bagFFT algorithm.

where C is a small universal constant.

Remarks:

- The remarks following Claim 1 are valid here as well.
- When computing Δ_K^{ψ} from (5.4) we plug in $\widetilde{p}_\mu$ and $\widetilde{\psi}_\mu$ for p_μ and ψ_μ, respectively. Still, (5.5) holds since by Claim 2 and its following remark the difference can be absorbed in the $O(\varepsilon_0^2)$ term.

Corollary 1. *For $n \in [0..Q-1]$ and a small universal constant C,*

$$\Big|\sum_{j\geq n}\widetilde{\widetilde{p_Q}(j)} - \sum_{j\geq n} p_Q(j)\Big| \leq \sum_{j\geq n}\left[\Delta_{p_\theta} e^{-\theta\delta j+\log M(\theta)} + (Q + CN\log N)\widetilde{p_Q}(j)\right]\varepsilon_0 + O(\varepsilon_0^2).$$

Remarks:

- The proof of the corollary follows from Claim 3 and Lemma 2 (in the Appendix).
- The relative error term, $\sum_{j\geq n}(Q + CN\log N)\widetilde{p_Q}(j)\varepsilon_0$, tends to be negligible in practice.
- A tighter bound can be obtained here from analysis of the l^2-norm of the error (using (5.3) and Claim 2) and from more careful summations.

Minimizing the bound in (5.5) is in principle a two-dimensional optimization problem. However, we found that first solving (4.5) for θ and then choosing θ_2 that minimizes $\Delta_K^{\psi} e^{\theta_2 N}$ works sufficiently well in practice. We present a summary of the bagFFT algorithm in Figure 5. We also present an illustrated example for the algorithm in the Appendix. As the θ_2 computation adds only $O(KN\log N)$ to the runtime, the runtime of this algorithm is $O(QKN\log N)$.

Table 1. Range of Test Parameters for Comparing bagFFT with Hirji's Algorithm. *Uniform* refers to the distribution where $\pi_k = 1/K$, *Sloped* refers to the case where $\pi_k = k/(K*(K+1)/2)$, and *Blocked* refers to the case where $\pi_k = 3/(4\lfloor K/4 \rfloor)$ if $k \leq \lfloor K/4 \rfloor$ and $\pi_k = 1/(4*(K-\lfloor K/4 \rfloor))$ otherwise.

Parameter	*Values*
K	4, 10, 20
N	50, 100, 200, 400
π	Uniform, Sloped, Blocked
s	$\frac{i}{21} * I_{max}$ $i \in [1..20]$

6. RESULTS

6.1 Accuracy

As a test of accuracy for bagFFT we compared its results to those from a lattice version of Hirji's algorithm (which can be proven to be numerically stable). The range of parameters for the comparison is given in Table 1. The comparison was done using C implementations and with double precision arithmetic. For the set of 720 test cases defined by table 1 and with Q set to 16,384 we found that bagFFT agreed with Hirji's algorithm to more than 12 decimal places in all cases. The same experiment was also repeated with values of s that are much closer to $I_{\max}$: an interval halving procedure on the range $[(\frac{20}{21} * I_{\max})..I_{\max}]$ was used to get eight values of s. The agreement was again to more than 12 decimal places. In addition, in both these experiments the theoretical error bounds from Figure 5 guarantee nearly six decimal places of accuracy in all cases.

The set of parameters in Table 1 is restricted to small values of N and K and one reason this is so is because these are the typical ranges that are of interest in bioinformatics applications. However, there is also a practical reason, which is that Hirji's algorithm is quite slow for large values of N and K (and it also requires a substantial amount of memory). For example, for $N = 10{,}000$, $K = 20$, and $Q = 16{,}384$, we estimated that Hirji's algorithm would take at least 40 hours while bagFFT takes about 25 minutes (for optimized C implementations). Fortunately, we can compute error bounds for bagFFT to confirm that the computed values are accurate. To verify that bagFFT is useful even for large values of N and K we conducted two sets of tests. In the first test we allowed N to vary over $\{1{,}000, 2{,}000, 5{,}000, 10{,}000\}$ where the other parameters vary as before. In this case, the theoretical error bounds from (5.5) guarantee more than four decimal places of accuracy in all cases. In the second test, we varied K over $\{50, 75, 100, 200\}$ with the other parameters varying as before. For this experiment, the guarantee is still more than three decimal places for all the cases tested. We report the results of a few more tests of accuracy in the Appendix (p. 796).

Besides serving to confirm the accuracy of computed p values, the theoretical error bounds are also useful for identifying the regions of the pmf that are accurately computed.

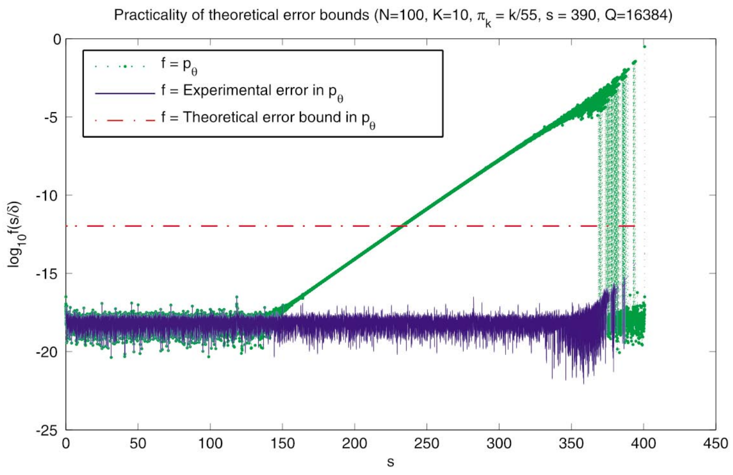


Figure 6. Practicality of theoretical error bounds (5.5) for estimating the error in p_θ. Note the plotted values for p_θ are those computed using the bagFFT algorithm. The region where these values are much larger than the theoretical error bound defines the entries of p_θ which can be trusted in practice. As can be seen, this approach can be used to recover a large proportion of the reliable entries of p_θ.

An example of this can be seen in Figure 6. Here the theoretical bounds, while being conservative by design, can still be used to recover nearly 60% of the correct entries of p_θ (where we want both theoretical and actual relative error to be less than 10%).

6.2 Runtime

For runtime comparisons we implemented bagFFT and Hirji's algorithm in C with particular attention to optimizing the runtime of the programs. Based on our experiments we observed that while Hirji's algorithm is efficient for small values of N, bagFFT is faster as N increases. In particular, for $K = 20$, bagFFT is faster for $N > 30$. The asymptotic behavior of the algorithms can be clearly seen in Figure 7 where we plot the runtime of the two algorithms with increasing N for a fixed choice of the other parameter values (the graph is similar looking for other choices of the parameter values as well).

In columns 1 and 2 of Table 2 we present the runtime of Hirji's algorithm and bagFFT for a set of parameter values that demonstrate the typical behavior of the algorithms. As can be seen from lines 2 and 4, while the choice of π does not affect the runtime of bagFFT it does affect the runtime of Hirji's algorithm. For Hirji's algorithm, $\pi =$ Uniform is the worst case and the runtime decreases for other choices of π. Also, as can be seen from lines 2, 3, and 5, as K increases, the "crossover point" between the runtime curves for bagFFT and Hirji's algorithm becomes smaller. In other words, bagFFT becomes more efficient sooner, with respect to N, as K increases. Finally, lines 6 and 7 demonstrate the substantial difference in runtime between Hirji's algorithm and bagFFT as N and K become large.

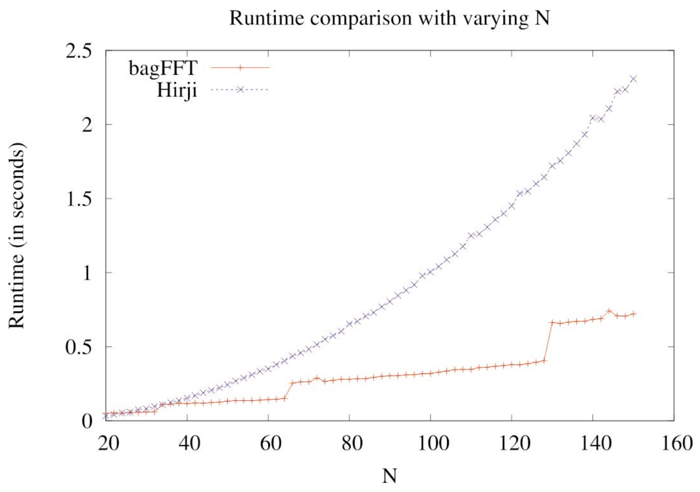


Figure 7. Runtime comparison of bagFFT and Hirji's algorithm, for varying N. The parameter values used in this comparison are $K = 20$, $\pi_j = j/(K * (K + 1)/2)$, and $Q = 1{,}024$. The runtimes reported are averaged over 10 evenly spaced s values in the range $[0..I_{\max}]$. Note that the discontinuities in the curve for bagFFT are due to the fact that our implementation of FFT works with arrays whose sizes are powers of 2.

7. RECOVERING THE ENTIRE PMF AND ITS APPLICATION

So far our goal was to compute a single p value, however, we often need to evaluate many different values of I. In such cases it would be better to compute the entire pmf, p_Q, in advance. Hirji's algorithm can be modified to compute p_Q in the same $O(QKN^2)$ time it can take to compute a single p value. The difference, however, is that in the case of a single p value $O(QKN^2)$ is a worst case analysis and in many cases the computation is significantly faster. These savings which apply only for computing a single p value are due to the pruning that any network algorithm (Mehta and Patel 1983) such as Hirji's employs.

Although bagFFT was designed for computing a single p value, in practice it can be easily adapted to reliably estimate p_Q in its entirety. In some cases it already does that: for example, for $s = 100$, $N = 100$, $K = 4$, $\pi_k = 1/4$, and $Q = 16{,}384$ we get a reliable

Table 2. Runtime in Seconds for bagFFT, Hirji's Algorithm, and a version of Hirji's Algorithm Without Pruning (see Section 7), for Various Parameter Values. Here, Q is set to 1,024 and the runtimes reported are averaged over s values in the range $\{\frac{i}{11} * I_{\max} | i \in [1..10]\}$ (except for the last line where $Q = 16{,}384$ and $s = 3{,}000$).

Parameters			*Hirji's algorithm*	*bagFFT*	*Hirji without pruning*
$N = 50$,	$K = 4$,	$\pi =$ Uniform	0.006	0.022	0.01
$N = 400$,	$K = 4$,	$\pi =$ Uniform	0.4	0.4	1.3
$N = 1{,}600$,	$K = 4$,	$\pi =$ Uniform	13.1	4.7	44.5
$N = 400$,	$K = 4$,	$\pi =$ Sloped	0.3	0.4	1.7
$N = 50$,	$K = 20$,	$\pi =$ Uniform	0.3	0.13	0.7
$N = 400$,	$K = 20$,	$\pi =$ Uniform	7.4	2.7	77.9
$N = 1{,}600$,	$K = 20$,	$\pi =$ Uniform	$4.5 \cdot 10^3$	110.2	$> 1.9 \cdot 10^4$

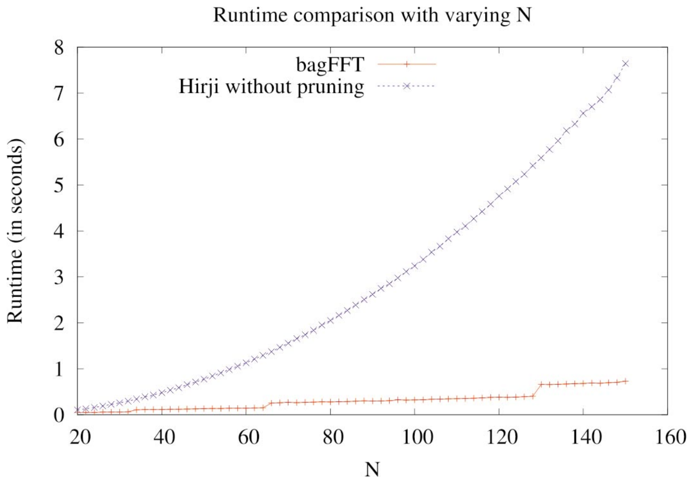


Figure 8. Runtime comparison of bagFFT and Hirji's algorithm without pruning, for varying N. The parameter values used in this comparison are the same as in Figure 7.

estimate for all the entries in p_Q (with relative error $< 10^{-9}$). In all cases that we tried we could reliably recover the entire range of values of p_Q using as little as two to three different s values, or equivalently, θ's: recall that each estimate has an error bound, based on (5.5), which allows us to choose the estimate which has better error guarantees. This approach is typically still significantly cheaper than running Hirji's algorithm, especially since without pruning the latter is significantly slower than bagFFT (even for much smaller N) as demonstrated in Figure 8 and Table 2.

As mentioned in Section 2, an important application for recovering p_Q in its entirety is the computation of the p value of a sum of entropy scores, $I_A = \sum_j I(j)$, from L independent columns of an alignment. The sFFT algorithm (Keich 2005) applies an exponential shift to p_Q so that it can use FFT to compute the L-fold convolution p_Q^{*L}. In the original implementation of sFFT, p_Q was computed using naive enumeration. Here we present a modification to sFFT that uses bagFFT to compute p_Q.

As suggested above, typically, a few applications of bagFFT can be used to recover all the entries of p_Q accurately. However, this approach may expend too much effort in recovering entries of p_Q that do not contribute significantly to the p value for a particular score. Indeed, from Keich (2005) we know that the entries of p_Q that are most relevant to computing the p value of $I_A = s_A$ are centered about s_A/L, suggesting the algorithm summarized in Figure 9. The runtime for this algorithm is $O(QKN\log N + LQ\log(LQ))$.

The following claim (proved in the Appendix, p. 796) bounds the magnitude of the accumulated roundoff error in our computation.

Claim 4.

$$|\widetilde{p_Q{}^{*L}}(j) - p_Q{}^{*L}(j)| \le \varepsilon_0 \left[L\Delta_{p_\theta} + (L+1)C_F \log(LQ)\right] e^{-\theta\delta j + L\log M(\theta)} + C\widetilde{p_Q{}^{*L}}(j)\varepsilon_0 + O(\varepsilon_0^2)$$

```
Given $N, K, L, \pi, Q$, and $s_A$, the algorithm:
 1. Executes steps 1–4 of Figure 5 with $s = s_A/L$
 2. Computes $q_\theta(j) = \begin{cases} u(j)\frac{e^{\theta_2 N}}{P(X_+=N)} = p_\theta(j)M(\theta) & j = 0,\dots,Q-1 \\ 0 & j = Q,\dots,LQ-1 \end{cases}$.
 3. For $l = 0, 1, \dots, LQ-1$, computes $y(l) = \left[(Dq_\theta)(l)\right]^L$, where $D = D_{LQ}$.
 4. Computes $w = D^{-1}y$.
 5. Computes $p_Q^{*L}(j) = w(j)e^{-\theta\delta j}$ (or the logarithmic version).
 6. Returns $\sum_{j\geq\lceil s_A/\delta+LK/2\rceil} p_Q^{*L}(j)$ and $\sum_{j\geq\lfloor s_A/\delta-LK/2\rfloor} p_Q^{*L}(j)$ as the lower
    and upper bounds respectively for the $p$ value (or the logarithmic version).
```

Figure 9. An algorithm for computing the p value of the information content of an alignment.

where C is a small universal constant and with Δ_{p_θ} as in Claim 3.

The reliability of this algorithm was tested by comparison to the numerically stable, naive convolution based algorithm (NC) in Hertz and Stormo (1999) on a typical range of parameters as described in Table 3. We found that in all 1,600 cases the combination of bagFFT and sFFT is in agreement with the results from NC to at least 11 decimal places and the theoretical bounds (from Claim 4 and analogous to Corollary 1) guarantee accuracy to at least five decimal places.

8. CONCLUSION AND FUTURE WORK

The bagFFT algorithm is asymptotically the fastest algorithm for computing the exact p value of the G^2 statistic for goodness-of-fit tests. We complement the algorithm with a rigorous analysis of the accumulation of roundoff errors in it. Moreover, we show empirically that for a wide range of parameters these error bounds are useful to guarantee the quality of the computed p value. We demonstrate the utility of our approach by combining bagFFT and sFFT to provide a fast, new algorithm for estimating the significance of sequence motifs. The bagFFT algorithm is available at *http://www.cs.cornell.edu/~keich/*.

We are still working on certain algorithmic refinements to bagFFT. In particular, we wish to optimize bagFFT for computing a single p value. This is motivated by Hirji's algo-

Table 3. Range of Test Parameters for Testing the Combination of bagFFT and sFFT. Here $K = 4$, *Uniform* refers to the case where $\pi_k = 1/4$, *Sloped* refers to $\pi_k = k/10$, *Blocked* refers to $\pi = [0.2, 0.2, 0.3, 0.3]$ and *Perturbed Uniform* refers to $\pi = [0.2497, 0.2499, 0.2501, 0.2503]$.

Parameter	*Values*
L	5, 10, 15, 30
N	5, 10, 15, 20, 50
π	*Uniform*, *Sloped*, *Blocked*, *Perturbed Uniform*
s	$\frac{i}{21} * L * I_{\max}\ i \in [1..20]$

rithm, which as a network algorithm, is optimized for computing a single p value based on pruning strategies described by Hirji (1997) [another strategy was described by Bejerano, Friedman, and Tishby (2004)]. This pruning is one of the main reasons Hirji's algorithm is still faster than bagFFT for smaller N. We are currently working on providing similar runtime gains for bagFFT. Our future goals include designing a "stitched" algorithm that can choose among a range of existing algorithms so as to be optimal for any given set of parameter values and a desired level of accuracy. We would also like to explore the applicability of bagFFT for Pearson's X^2 and for log-linear models, as well as a generalization to two-dimensional contingency tables, as is the case for Baglivo et al.'s algorithm (Baglivo, Olivier, and Pagano 1992).

The bagFFT algorithm serves as another demonstration of the effectiveness of the shifted-FFT technique (Keich 2005) to accurately compute vanishingly small p values. We plan on studying the applicability of this method to nonparametric tests such as the Mann-Whitney as well.

APPENDIX

A.1 PROOF OF CLAIM 1

The following lemma can be readily derived from the results in Keich (2005; see Lemmas 1–3, equations (20) and (21)). For $\alpha \in \mathbb{C}$ we denote by $\widetilde{\alpha}$ its machine estimator and define $e_\alpha = \widetilde{\alpha} - \alpha$. For $\alpha, \beta \in \mathbb{C}$, we define

$$e_{\alpha+\beta} = \widetilde{\widetilde{\alpha} + \widetilde{\beta}} - (\alpha + \beta),$$

and similarly for $e_{\alpha\beta}$.

Lemma 2. *If* $|e_\alpha| < c_\alpha \alpha \varepsilon_0$ *and* $|e_\beta| < c_\beta \beta \varepsilon_0$*, then*

$$\begin{aligned} |e_{\alpha+\beta}| &\leq (\max\{c_\alpha, c_\beta\} + 1)(|\alpha| + |\beta|)\varepsilon_0 \\ |e_{\alpha\beta}| &\leq (c_\alpha + c_\beta + 5)(|\alpha\beta|)\varepsilon_0. \end{aligned}$$

We can now prove the claim. Similarly to the remark following Claim 2, from (5.1) and the fact that $|e^{i\phi}| = 1$, we have

$$|\widetilde{p}_{k,l,\theta}(n) - p_{k,l,\theta}(n)| \leq CN \log N |p_{k,l,\theta}(n)|\varepsilon_0 = CN \log N p_{k,0,\theta}(n)\varepsilon_0.$$

Combining this bound with the previous lemma one can use (4.3) to prove by induction on k that

$$|\psi_{k,l,\theta}(n) - \widetilde{\psi}_{k,l,\theta}(n)| \leq (CkN \log N)\psi_{k,0,\theta}(n)\varepsilon_0.$$

In particular, with $\rho(l) = \psi_{K,l,\theta}(N)$

$$|\rho(l) - \widetilde{\rho}(l)| \leq (CKN \log N)M(\theta)P(X_+ = N)\varepsilon_0. \tag{A.1}$$

Let D be the m-dimensional DFT operator. It is easy to show that for $v \in \mathbb{C}^m$

$$\|Dv\|_\infty \le \|v\|_1 , \quad \|D^{-1}v\|_\infty \le \frac{1}{m}\|v\|_1 \le \|v\|_\infty. \tag{A.2}$$

Let $\widetilde{D}$ be D's FFT machine implementation, then it follows from (A.5) below and $\|v\|_\infty \le \|v\|_2 \le \sqrt{m}\|v\|_\infty$ that:

$$\|(D^{-1} - \widetilde{D^{-1}})v\|_\infty \le C_F \log_2(m)\varepsilon_0\|v\|_\infty + O(\varepsilon_0^2). \tag{A.3}$$

Using the triangle inequality, (A.1), (A.2), and (A.3) we get

$$\begin{aligned}\|D^{-1}\rho - \widetilde{D^{-1}}\widetilde{\rho}\|_\infty &\le \|D^{-1}(\rho-\widetilde{\rho})\|_\infty + \|(D^{-1} - \widetilde{D^{-1}})\widetilde{\rho}\|_\infty \\ &\le \|\rho - \widetilde{\rho}\|_\infty + C_F \log_2 Q\varepsilon_0\|\rho\|_\infty + O(\varepsilon_0^2) \\ &\le C(KN\log N + \log_2 Q)M(\theta)P(X_+ = N)\varepsilon_0 + O(\varepsilon_0^2).\end{aligned}$$

Claim 1 now follows from multiplying by $e^{-\theta\delta j}/P(X_+ = N)$ (see (4.4)).

A.2 PROOF OF LEMMA 1

Let D be the m-dimensional DFT. The discrete Parseval identity (e.g., Press, Teukolsky, Vetterling, and Flannery 1992) states that for $v \in \mathbb{C}^m$,

$$\|D^{-1}v\|_2 = \frac{1}{\sqrt{m}}\|v\|_2 , \quad \|Dv\|_2 = \sqrt{m}\|v\|_2. \tag{A.4}$$

Let $\widetilde{D}$ denote the FFT machine implementation of the DFT. Then, there exists a constant $C_F < 5$ such that (Tasche and Zeuner 2001):

$$\begin{aligned}\|(\widetilde{D^{-1}} - D^{-1})v\|_2 &\le \frac{1}{\sqrt{m}}C_F \log_2(m)\varepsilon_0\|v\|_2 + O(\varepsilon_0^2) \\ \|(\widetilde{D} - D)v\|_2 &\le \sqrt{m}C_F \log_2(m)\varepsilon_0\|v\|_2 + O(\varepsilon_0^2).\end{aligned} \tag{A.5}$$

The following bound on the norm of a convolution is used repeatedly below. Let $u, v \in \mathbb{C}^m$, then it follows from (A.2) and (A.4) (with $\odot$ being the pointwise product operator) that

$$\frac{1}{\sqrt{N_2}}\|Du \odot Dv\|_2 \le \frac{1}{\sqrt{N_2}}\|Du\|_2\|Dv\|_\infty \le \frac{1}{\sqrt{N_2}}\|Du\|_2\|v\|_1 = \|u\|_2\|v\|_1. \tag{A.6}$$

We are now ready to prove the lemma:

$$\|\widetilde{D^{-1}}\widetilde{\tau}\,\widetilde{\odot}\,\widetilde{\nu} - x * y\|_2 \le \underbrace{\|D^{-1}(\tau\odot\nu - \widetilde{\tau}\,\widetilde{\odot}\,\widetilde{\nu})\|_2}_{\alpha} + \underbrace{\|(\widetilde{D^{-1}} - D^{-1})\widetilde{\tau}\,\widetilde{\odot}\,\widetilde{\nu}\|_2}_{\beta}. \tag{A.7}$$

From (A.2)–(A.6) and Lemma 2 we have

$$\begin{aligned}\alpha &= \frac{1}{\sqrt{N_2}}\|\tau\odot\nu - \widetilde{\tau}\,\widetilde{\odot}\,\widetilde{\nu}\|_2 \\ &\le \underbrace{\frac{1}{\sqrt{N_2}}\|(\tau-\widetilde{\tau})\odot\nu\|_2}_{\alpha_1} + \underbrace{\frac{1}{\sqrt{N_2}}\|\widetilde{\tau}\odot(\nu-\widetilde{\nu})\|_2}_{\alpha_2} + \underbrace{\frac{1}{\sqrt{N_2}}\|\widetilde{\tau}\odot\widetilde{\nu} - \widetilde{\tau}\,\widetilde{\odot}\,\widetilde{\nu}\|_2}_{\alpha_3},\end{aligned}$$

where

$$\begin{aligned}
\alpha_1 &\leq \frac{1}{\sqrt{N_2}} \|\tau - \widetilde{\tau}\|_2 \|y\|_1 \\
&\leq \left[\frac{1}{\sqrt{N_2}} \|D(x - \widetilde{x})\|_2 + \frac{1}{\sqrt{N_2}} \|(D - \widetilde{D})\widetilde{x}\|_2 \right] \|y\|_1 \\
&\leq \varepsilon_0 \big[m_x + C_F \log_2 N_2 \|\widetilde{x}\|_2 \big] \|y\|_1 + O(\varepsilon_0^2).
\end{aligned}$$

$$\begin{aligned}
\alpha_2 &\leq \frac{1}{\sqrt{N_2}} \|\nu - \widetilde{\nu}\|_2 \|\widetilde{\tau}\|_\infty \\
&\leq \left[\varepsilon_0 \left(m_y + C_F \log_2 N_2 \|\widetilde{y}\|_2 \right) + O(\varepsilon_0^2) \right] \left[\|(\widetilde{D} - D)\widetilde{x}\|_\infty + \|D\widetilde{x}\|_\infty \right] \\
&\leq \varepsilon_0 \left[m_y + C_F \log_2 N_2 \|\widetilde{y}\|_2 \right] \|\widetilde{x}\|_1 + O(\varepsilon_0^2).
\end{aligned}$$

$$\begin{aligned}
\alpha_3 &\leq 5\varepsilon_0 \frac{1}{\sqrt{N_2}} \|\widetilde{\tau} \odot \widetilde{\nu}\|_2 \\
&\leq 5\varepsilon_0 \frac{1}{\sqrt{N_2}} \|\widetilde{\nu}\|_2 \|\widetilde{\tau}\|_\infty \\
&\leq 5\varepsilon_0 \left[\frac{1}{\sqrt{N_2}} \|(\widetilde{D} - D)\widetilde{y}\|_2 + \frac{1}{\sqrt{N_2}} \|D\widetilde{y}\|_2 \right] \left[\|\widetilde{x}\|_1 + O(\varepsilon_0) \right] \\
&\leq 5\varepsilon_0 \|\widetilde{x}\|_1 \|\widetilde{y}\|_2 + O(\varepsilon_0^2).
\end{aligned}$$

Finally, by the same type of arguments

$$\beta \leq \varepsilon_0 C_F \log_2 N_2 \|\widetilde{\widetilde{\tau} \odot \widetilde{\nu}}\|_2 \leq \varepsilon_0 C_F \log_2 N_2 \|\widetilde{x}\|_1 \|\widetilde{y}\|_2 + O(\varepsilon_0^2).$$

The proof is completed by collecting all the terms into (A.7) and noting that the differences between $\|y\|$ and $\|\widetilde{y}\|$ (or $\|x\|$ and $\|\widetilde{x}\|$) are absorbed in the $O(\varepsilon_0^2)$ term.

A.3 Proof of Claim 3

For $l = 0, \ldots, Q-1$ let $\rho(l) = \psi_{K,l,\theta,\theta_2}(N)$. Then

$$\begin{aligned}
\|D^{-1}\rho - \widetilde{D^{-1}}\widetilde{\rho}\|_2 &\leq \|D^{-1}(\rho - \widetilde{\rho})\|_2 + \|(D^{-1} - \widetilde{D^{-1}})\widetilde{\rho}\|_2 \\
&\leq \frac{1}{\sqrt{Q}} \|\rho - \widetilde{\rho}\|_2 + \frac{1}{\sqrt{Q}} C_F \log_2 Q \varepsilon_0 \|\widetilde{\rho}\|_2 + O(\varepsilon_0^2) \\
&\leq \|\rho - \widetilde{\rho}\|_\infty + C_F \log_2 Q \varepsilon_0 \|\widetilde{\rho}\|_\infty + O(\varepsilon_0^2) \\
&\leq \left[\Delta_K^\psi + C_F \log_2 Q M(\theta) P(X_+ = N) e^{-\theta_2 N} \right] \varepsilon_0 + O(\varepsilon_0^2),
\end{aligned} \tag{A.8}$$

where the last inequality follows from Claim 2 and

$$|\psi_{K,l,\theta,\theta_2}(N)| \leq \psi_{K,0,\theta,\theta_2}(N) = M(\theta) P(X_+ = N) e^{-\theta_2 N}.$$

The proof now follows from

$$p_Q(j) = (D^{-1}\rho)(j) \frac{e^{-\theta\delta j + \theta_2 N}}{P(X_+ = N)}.$$

A.4 PROOF OF CLAIM 4

By the same arguments as above, with $D = D_{LQ}$ and $w = D^{-1}y$ as in Figure 9,

$$\begin{aligned} \|w - \widetilde{w}\|_2 &\leq \|D^{-1}(y - \widetilde{y})\|_2 + \|(D^{-1} - \widetilde{D^{-1}})\widetilde{y}\|_2 \\ &\leq \frac{1}{\sqrt{LQ}}\|y - \widetilde{y}\|_2 + \frac{1}{\sqrt{LQ}} C_F \log_2(LQ)\varepsilon_0 \|\widetilde{y}\|_2 + O(\varepsilon_0^2). \end{aligned} \tag{A.9}$$

Let $y(l) = [(Dq_\theta)(l)]^L$ and let $q_\theta \equiv \frac{e^{\theta_2 N}}{P(X_+ = N)} u 1_{[0,\ldots,Q-1]} \equiv M(\theta) p_\theta 1_{[0,\ldots,Q-1]}$ as in Figure 9. By (A.2)

$$\|y\|_\infty \leq \|Dq_\theta\|_\infty^L \leq \|q_\theta\|_1^L = [M(\theta)]^L.$$

It follows that,

$$\frac{1}{\sqrt{LQ}}\|\widetilde{y}\|_2 \leq [M(\theta)]^L + \frac{1}{\sqrt{LQ}}\|y - \widetilde{y}\|_2 \tag{A.10}$$

and since $|(a+h)^L - a^L| \leq L|h||a|^{L-1} + O(|h|^2)$, that

$$|\widetilde{y}(l) - y(l)| \leq LM(\theta)^{L-1} |(Dq_\theta)(l) - \widetilde{D}\widetilde{q}_\theta(l)| + O(\varepsilon_0^2).$$

Therefore,

$$\|y - \widetilde{y}\|_2 \leq LM(\theta)^{L-1}\|Dq_\theta - \widetilde{D}\widetilde{q}_\theta\|_2 + O(\varepsilon_0^2). \tag{A.11}$$

As $u \equiv D_Q^{-1}\left[\psi_{K,\bullet,\theta,\theta_2}(N)\right]$ it follows from (A.8) that

$$\begin{aligned} \|q_\theta - \widetilde{q}_\theta\|_2 &\leq \varepsilon_0 \Big[\frac{\Delta_K^\psi e^{\theta_2 N}}{P(X_+ = N)} + C_F M(\theta) \log_2 Q\Big] + O(\varepsilon_0^2) \\ &\leq \varepsilon_0 \Delta_{p_\theta} M(\theta) + O(\varepsilon_0^2), \end{aligned}$$

and since

$$\|q_\theta\|_2 \leq \|q_\theta\|_1 = M(\theta),$$

it follows that

$$\begin{aligned} \|Dq_\theta - \widetilde{D}\widetilde{q}_\theta\|_2 &\leq \|D(q_\theta - \widetilde{q}_\theta)\|_2 + \|(D - \widetilde{D})\widetilde{q}_\theta)\|_2 \\ &\leq \sqrt{LQ}\left[\|q_\theta - \widetilde{q}_\theta\|_2 + C_F \log_2(LQ)\varepsilon_0\|\widetilde{q}_\theta\|_2\right] + O(\varepsilon_0^2) \\ &\leq \sqrt{LQ}\left[\Delta_{p_\theta} M(\theta) + C_F \log_2(LQ) M(\theta)\right]\varepsilon_0 + O(\varepsilon_0^2). \end{aligned} \tag{A.12}$$

Plugging (A.11), (A.12), and (A.10) back into (A.9) we get:

$$\begin{aligned} \|w - \widetilde{w}\|_2 &\leq LM(\theta)^{L-1}\left[\Delta_{p_\theta} M(\theta) + C_F \log_2(LQ) M(\theta)\right]\varepsilon_0 \\ &\quad + C_F \log_2(LQ)\|\widetilde{y}\|_\infty \varepsilon_0 + O(\varepsilon_0^2) \\ &\leq \left[L\Delta_{p_\theta} + (L+1)C_F \log_2(LQ)\right] M(\theta)^L \varepsilon_0 + O(\varepsilon_0^2). \end{aligned}$$

The proof is now immediate from $p_Q^{*L}(j) = w(j)e^{-\theta\delta j}$.

A.5 AN ILLUSTRATION OF THE BAGFFT ALGORITHM

In Figure A.1 we present an illustrated example for the core of the bagFFT algorithm, that is, computing $\psi_{k,l,\theta,\theta_2}$ starting from the p_k's. The parameters used in this example are $N = 100, K = 10, \pi = \{(10 - i)/55 | i \in [0..9]\}, s = 100$, and $Q = 16{,}384$.

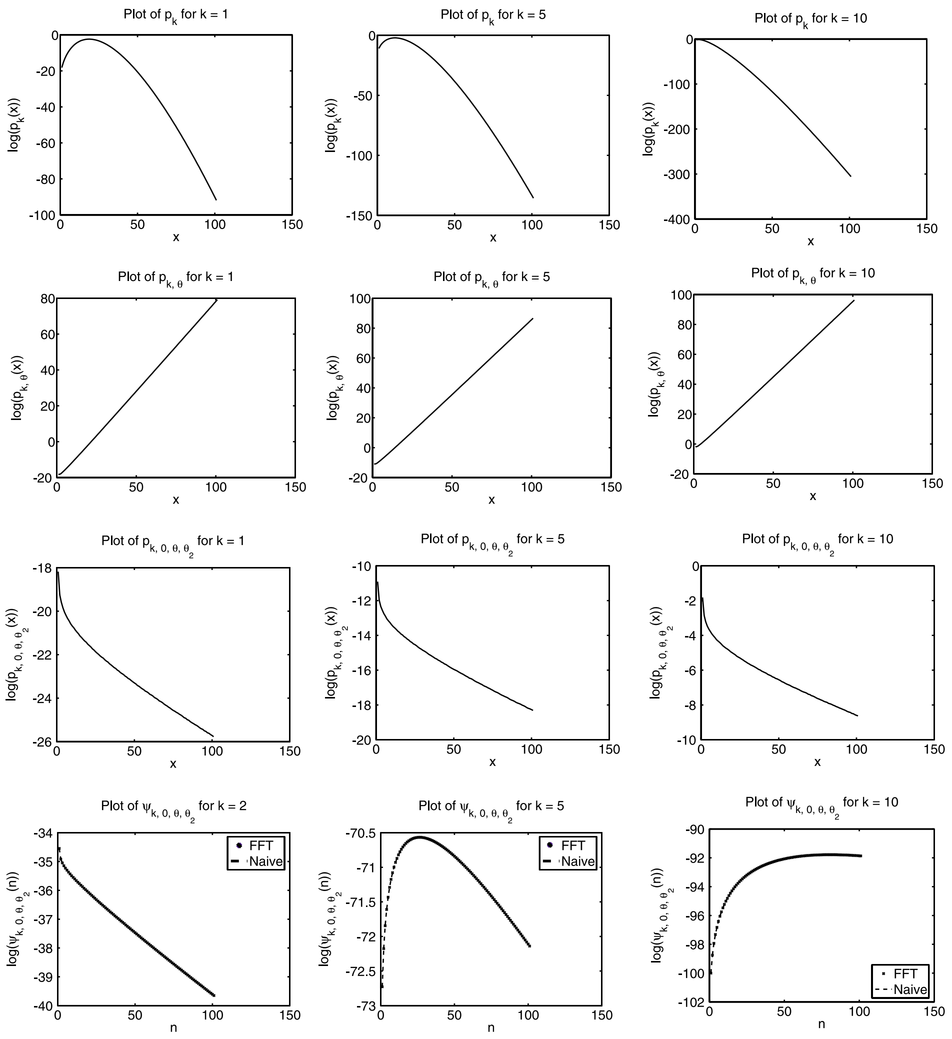
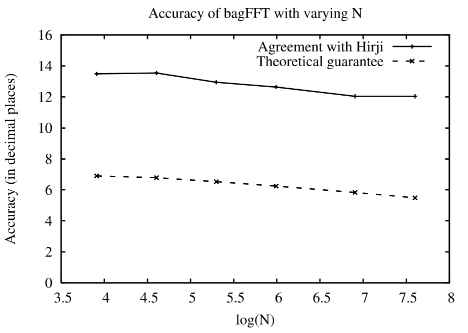
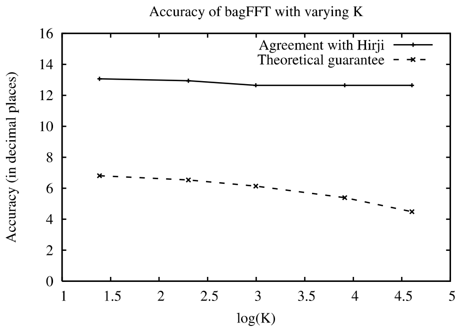


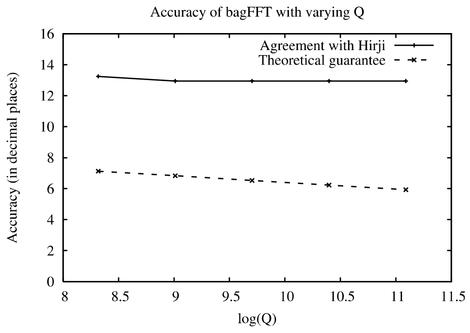
Figure A.1. Graphical illustration of the bagFFT algorithm. Computation using the p_k's shown in row 1 leads to the roundoff errors described in Figure 2. So a shift with $\theta = 1$ is applied to get the $p_{k,\theta}$'s shown on row 2. To aid FFT-convolutions using the $p_{k,\theta}$'s, they are shifted with $\theta_2 = 1.05$ to get the $p_{k,0,\theta,\theta_2}$'s on row 3 (note the different scale from the previous row). These are now convolved (using FFTs) to accurately recover the $\psi_{k,0,\theta,\theta_2}$'s, as can be seen from row 4 (by comparison to the curves from naive convolution that overlap very well). Note that corresponding FFT-convolutions with the $p_{k,\theta}$'s (without the second shift) does not recover any of the entries of $\psi_{k,0,\theta}$ accurately (data not shown).



(a) $K = 10$, $\pi = Uniform$, $Q = 16{,}384$ and N varies over $\{50, 100, 200, 400, 1{,}000, 2{,}000\}$



(b) $N = 200$, $\pi = Uniform$, $Q = 16{,}384$ and K varies over $\{4, 10, 20, 50, 100\}$



(c) $K = 10$, $N = 200$, $\pi = Uniform$ and Q varies over $\{4{,}096, 8{,}192, 16{,}384, 32{,}768, 65{,}536\}$

Figure A.2. Accuracy of bagFFT as a function of N, K, and Q. The values reported here are the minimum values for s in the range $\{\frac{i}{21} * I_{\max} | i \in [1..20]\}$.

A.6 ACCURACY OF BAGFFT AS A FUNCTION OF N, K, AND Q

The behavior of the theoretical error bounds and the agreement between bagFFT and Hirji's algorithm, as a function of N, K and Q, is illustrated in Figure A.2. Here we define agreement with Hirji's algorithm as

$$-\log_{10}(\max(|L_H(s) - L(s)|/L_H(s), |U_H(s) - U(s)|/U_H(s)))$$

where L_H and U_H are the corresponding lattice bounds for the p value reported by Hirji's algorithm. Correspondingly, the theoretical error guarantee is calculated as

$$-\log_{10}(\max(E_L(s)/|L(s) - E_L(s)|, E_U(s)/|U(s) - E_U(s)|)).$$

An important trend to note here is that the agreement with Hirji's algorithm is essentially constant with increasing Q. In the rest of the cases the trend is that accuracy decreases roughly linearly as a function of $\log N$, $\log K$, and $\log Q$. The results therefore indicate that both the error bounds and the agreement with Hirji's algorithm are relatively stable for increasing N, K, or Q.

ACKNOWLEDGMENTS

The authors thank Gill Bejerano for initial discussion on this subject and for generously sharing his code with us. We also thank the anonymous referees for their valuable comments and suggestions.

[Received June 2005. Revised February 2006.]

REFERENCES

Baglivo, J., Olivier, D., and Pagano, M. (1992), "Methods for Exact Goodness-of-Fit Tests," *Journal of the American Statistical Association*, 87, 464–469.

Bailey, T. L., and Elkan, C. (1994), "Fitting a Mixture Model by Expectation Maximization to Discover Motifs in Biopolymers," in *Proceedings of the Second International Conference on Intelligent Systems for Molecular Biology*, Menlo Park, CA, pp. 28–36.

Bejerano, G., Friedman, N., and Tishby, N. (2004), "Efficient Exact p-value Computation for Small Sample, Sparse and Surprising Categorical Data," *Journal of Computational Biology*, 11, 867–886.

Cressie, N., and Read, T. R. C. (1984), "Multinomial Goodness-of-Fit Tests," *Journal of the Royal Statistical Society*, Series B, 46, 440–464.

——— (1989), "Pearson's χ^2 and the Log-Likelihood Ratio Statistic g^2: A Comparative Review," *International Statistical Review*, 57, 19–43.

Dembo, A., and Zeitouni, O. (1998), *Large Deviation Techniques and Applications*, New York: Springer.

Hertz, G. Z., and Stormo, G. D. (1999), "Identifying DNA and Protein Patterns with Statistically Significant Alignments of Multiple Sequences," *Bioinformatics*, 15, 563–577.

Hirji, K. A. (1997), "A Comparison of Algorithms for Exact Goodness-of-Fit Tests for Multinomial Data," *Communications in Statistics—Simulation and Computations*, 26, 1197–1227.

Hoeffding, W. (1965), "Asymptotically Optimal Tests for Multinomial Distributions," *Annals of Mathematical Statistics*, 36, 369–408.

Keich, U. (2005), "Efficiently Computing the p-value of the Entropy Score," *Journal of Computational Biology*, 12, 416–430.

Mehta, C. R., and Patel, N. R. (1983), "A Network Algorithm for Performing Fisher's Exact Test in $r \times c$ Contingency Tables," *Journal of the American Statistical Association*, 78, 427–434.

Press, W. H., Teukolsky, S. A., Vetterling, W. T., and Flannery, B. P. (1992), *Numerical Recipes in C. The Art of Scientific Computing* (2nd ed.), Cambridge, UK: Cambridge University Press.

Rahmann, S. (2003), "Dynamic Programming Algorithms for Two Statistical Problems in Computational Biology," in *Proceedings of the Third International Workshop on Algorithms in Bioinformatics (WABI-03)*, volume 2812 of *Lecture Notes in Computer Science*, eds. G. Benson and R. D. M. Page, Budapest, Hungary, 2003, New York: Springer, pp. 151–164.

Sadreyev, R. I., and Grishin, N. V. (2004), "Estimates of Statistical Significance for Comparison of Individual Positions in Multiple Sequence Alignments," *BMC Bioinformatics*, 5.

Siotani, M., and Fujikoshi, Y. (1984), "Asymptotic Approximations for the Distributions of Multinomial Goodness-of-Fit Statistics," *Hiroshima Mathematical Journal*, 14, 115–124.

Stormo, G. D. (2000), "DNA Binding Sites: Representation and Discovery," *Bioinformatics*, 16, 16–23.

Tasche, M., and Zeuner, H. (2001), "Worst and Average Case Roundoff Error Analysis for FFT," *BIT*, 41, 563–581.