Nucleic Acids Research, 2005, Vol. 33, No. 15 ***4987–4994***
doi:10.1093/nar/gki800

The Gumbel pre-factor *k* for gapped local alignment can be estimated from simulations of global alignment

Sergey Sheetlin, Yonil Park and John L. Spouge*

National Center for Biotechnology Information, National Library of Medicine, Bethesda MD 20894, USA

Received May 12, 2005; Revised July 13, 2005; Accepted August 12, 2005

ABSTRACT

The optimal gapped local alignment score of two random sequences follows a Gumbel distribution. The Gumbel distribution has two parameters, the scale parameter λ and the pre-factor *k*. Presently, the basic local alignment search tool (BLAST) programs (BLASTP (BLAST for proteins), PSI-BLAST, etc.) use all time-consuming computer simulations to determine the Gumbel parameters. Because the simulations must be done offline, BLAST users are restricted in their choice of alignment scoring schemes. The ultimate aim of this paper is to speed the simulations, to determine the Gumbel parameters online, and to remove the corresponding restrictions on BLAST users. Simulations for the scale parameter λ can be as much as five times faster, if they use global instead of local alignment [R. Bundschuh (2002) *J. Comput. Biol.*, 9, 243–260]. Unfortunately, the acceleration does not extend in determining the Gumbel pre-factor *k*, because *k* has no known mathematical relationship to global alignment. This paper relates *k* to global alignment and exploits the relationship to show that for the BLASTP defaults, 10 000 realizations with sequences of average length 140 suffice to estimate both Gumbel parameters λ and *k* within the errors required (λ, 0.8%; *k*, 10%). For the BLASTP defaults, simulations for both Gumbel parameters now take less than 30 s on a 2.8 GHz Pentium 4 processor.

INTRODUCTION

Local sequence alignment is an indispensable computational tool in modern molecular biology. It is frequently used to infer the functional, structural and evolutionary relationships of a novel protein or DNA sequence by finding similar sequences of known function in a database. Arguably, the most important sequence database search program available is BLAST (the Basic Local Alignment Search Tool) (1,2). Using a heuristic algorithm, BLAST implicitly performs a local alignment of a protein or DNA query against sequences in the corresponding database. The BLAST output then ranks each potential database match according to an *E*-value, which is derived from the corresponding local maximum score, given in bits. For each local maximum score y, the corresponding *E*-value E_y gives (under a random model) the expected number of false positives with a lower rank in the output. Thus, a small *E*-value indicates that the corresponding alignment is unlikely to occur by chance alone, whereas a large *E*-value indicates an unremarkable alignment. Without doubt, BLAST's *E*-values contribute substantially to its popularity.

Let us discuss the BLAST *E*-value E_y further here. (The Materials and Methods section also continues the discussion.) BLAST assumes a random model in which each unrelated pair of sequences $\mathbf{A}[1, m] = A_1 \cdots A_m$ and $\mathbf{B}[1, n] = B_1 \cdots B_n$ consists of random letters chosen independently from a background distribution. BLASTP (BLAST for proteins), e.g. assumes that random proteins are composed of amino acids chosen independently from the Robinson and Robinson frequency distribution (3). BLAST also requires an input, a matrix $s(A_i, B_j)$ for scoring matches between the letters A_i and B_j. BLASTP, e.g. uses the BLOSUM62 scoring matrix (4) as its default, offering as alternatives a few other PAM (5) and BLOSUM matrices. BLAST also enhances its detection of remote sequence similarities by using gapped sequence alignment. The cost of introducing a gap into an alignment is given by the 'gap penalty' $\Delta(g)$, where g is the gap length. Practical gap penalties Δ are usually super-additive, i.e. $\Delta(g) + \Delta(h) \geqslant \Delta(g + h)$, so the concatenation of optimal subsequence alignments has a score no less than the sum of their scores. (However, our theory is not restricted to super-additive gap

*To whom correspondence should be addressed. Tel: +301 402 9310; Fax: +301 480 2288; Email: spouge@ncbi.nlm.nih.gov

The authors wish it to be known that, in their opinion, the first two authors should be regarded as joint First Authors.

penalties). Affine gap penalties $\Delta(g) = a + bg$ are typical in database searches. We refer to the letter distribution, the scoring matrix, and gap penalty collectively as 'BLAST parameters'.

Throughout the paper, we assume a 'logarithmic regime' (6) where the alignment scores of long random sequences have a negative expectation. In the logarithmic regime, the BLAST E-value E_y is approximately

$$E_y \approx kmne^{-\lambda y} \quad \mathbf{1}$$

for large y. Under a Poisson approximation (7) for large y, the E-value E_y yields the P-value $P_y = 1-\exp(-E_y)$. Because of Equation 1, the tail probability P_y corresponds to a Gumbel distribution with 'scale parameter' λ and 'pre-factor' k.

For ungapped local alignment (i.e. the special case $\Delta(g) = \infty$, which disallows gaps in the optimal local alignment), a rigorous theory furnishes analytic formulas for the Gumbel parameters λ and k (7,8). For gapped local alignment, analytic results are scarce and usually come at a price: they depend on approximations whose accuracy in general is unknown (9–12). In the absence of a rigorous theory for gapped local alignment, computer simulations have confirmed the validity of Equation 1 (13–16), and in the absence of formulas, they also have provided estimates of λ and k (16–19).

Because of the exponentiation in Equation 1, errors in λ have a greater practical impact than errors in k. Thus, for use in BLAST, λ must be known to within 1–4% relative error; k, to within 10% (20). Therefore, in statements about computational speed, the following implicitly assumes that the estimation of λ and k is carried out to these accuracies, unless stated otherwise.

Presently, the BLAST program precomputes λ and k offline, using the so-called 'island method' (15,20). Because of the precomputation, users are given a narrow choice indeed of BLAST parameters. The choice of BLAST parameters would be much less restricted, if λ and k could be computed online (in, say, less than 1 s) before searching a database with arbitrary BLAST parameters. Accordingly, much recent research has been directed toward speeding estimation of λ and k.

With the ultimate aim of estimating λ and k online, Bundschuh gave some interesting conjectures about λ (21,22). He then applied them in global alignment simulations that estimated λ as much as five faster than the island method. Later, we extended his conjectures, reducing the sequence length required to estimate λ by almost a factor of 10 (23).

Despite their obvious promise, even with further improvements in speed and global alignment simulations will remain impractical for online estimation in BLAST, unless they can be made to estimate k as well. To remedy the problem, we relate k to global alignment and then exploit the relationship in simulations that estimate both λ and k.

MATERIALS AND METHODS

Notation for global sequence alignment

We denote the non-negative integers by $\mathbb{Z}_+ = \{0, 1, 2, 3, \ldots\}$. Throughout the paper, the letters g, h, i, j, m, n and the letter y are the integers.

Consider a pair $\mathbf{A} = A_1A_2\ldots$ and $\mathbf{B} = B_1B_2\ldots$ of infinite sequences. The corresponding global alignment graph Γ is a directed and weighted lattice graph in two dimensions, as follows. The vertices of Γ are $v = (i, j) \in \mathbb{Z}_+^2$, the non-negative two-dimensional integer lattice. Three sets of directed edges e come out of each vertex $v = (i, j)$: northward, northeastward and eastward. One northeastward edge goes into $(i + 1, j + 1)$ with weight $s(A_{i+1}, B_{j+1})$. For each g > 0, one eastward edge goes into $(i + g, j)$ and one northward edge goes into $(i, j + g)$; both are assigned the same weight $-\Delta(g) < 0$. For simplicity, we assume $s(A_i, B_j)$ and $\Delta(g)$ are always integers, with greatest common divisor 1.

A directed path $\pi = (v_0, e_1, v_1, e_2, \ldots e_h, v_h)$ in Γ is a finite, alternating sequence of vertices and edges that starts and ends with a vertex. We say that the path π starts at v_0 and ends at v_h. For instance, each gapped alignment of the subsequences $\mathbf{A}[i + 1, m] = A_{i+1}\ldots A_m$ and $\mathbf{B}[j + 1, n] = B_{j+1}\ldots B_n$ corresponds to exactly one directed path that starts at $v_0 = (i, j)$ and ends at $v_h = (m, n)$. The alignment's score is the 'path weight' $W_\pi = \sum_{i=1}^{h} W(e_i)$, the sum of the weights $W(e_i)$ of the edges e_i. By convention, any trivial path $\pi = (v_0)$ consisting of a single vertex has weight $W_\pi = 0$.

Let Π_{ij} be the set of all paths π starting at $v_0 = (0, 0)$ and ending at $v_h = (i, j)$. Define the 'global score' $S_{ij} = \max\{W_\pi: \pi \in \Pi_{ij}\}$. The paths π starting at v_0 and ending at v_h with weight $W_\pi = S_{ij}$ are 'optimal global paths' and correspond to 'optimal global alignments' between $\mathbf{A}[1, i]$ and $\mathbf{B}[1, j]$. The Needleman–Wunsch algorithm computes the global scores S_{ij} (24).

Let $\Pi = \cup_{(i,j)\in\mathbb{Z}_+^2} \Pi_{ij}$ be the set of all paths π starting at $v_0 = (0,0)$. Define the 'global maximum' $M = \max\{W_\pi: \pi \in \Pi\}$, which is also the maximum $M = \max\{S_{ij} : (i, j) \in \mathbb{Z}_+^2\}$ of all global scores. Let $N(y) = \#\{(i, j) \in \mathbb{Z}_+^2 : S_{ij} = y\}$ denote the number of vertices with global score y.

Define the lattice rectangle $[0, n] = \{0, 1, \ldots, n\}$. Our simulations involved a square subset $[0, n]^2$ of $\mathbb{Z}_+^2$. In particular single subscripts connote quantities for the square: $M_n = \max\{S_{ij} : (i, j) \in [0, n]^2\}$, the square's global maximum; $E_n = \max\{\max_{0\leqslant i\leqslant n} S_{in}, \max_{0\leqslant j\leqslant n} S_{nj}\}$, its edge maximum; and $N_n(y) = \#\{(i, j) \in [0, n]^2 : S_{ij} = y\}$, the number of its vertices with global score y.

The formula for k from global alignment

We can show heuristically that $k = \lim_{y\to\infty} k_y$, where

$$k_y = \frac{e^{\lambda y}}{1-e^{-\lambda}} \cdot \frac{\mathbb{P}(M = y)^2}{\mathbb{E}N(y)} \quad \mathbf{2}$$

(see our Appendix, online). Ultimately, the heuristics behind Equation 2 are based on two observations about random sequence matches. First, the two ends of a strong local alignment match are the mirrors of each other. Second, the right end of a strong alignment match looks the same for both local and global alignment.

Equation 2 computes k_y from three components: the scale parameter λ, the probability $\mathbf{P}(M = y)$ of a global maximum y, and the expected number $\mathbb{E}N(y)$ of vertices with global score $S_{ij} = y$. We now describe how our simulations determined the three components.

Numerical scheme for λ

First, we estimated λ from random global alignments (23). All simulations used to affine gap penalties $\Delta(g) = a + bg$ and

the corresponding global alignment algorithms for computing S_{ij} (25).

Recall the edge maximum E_n (defined at the end of the notation for global sequence alignment). As shown elsewhere (23), its cumulant generating function satisfies

$$\ln[\mathbb{E}\exp(\lambda E_n)] = \beta_0 + \beta_1(\lambda)n + O(\delta^n), \quad \mathbf{3}$$

where $0 \leqslant \delta < 1$. The root $\lambda = \hat{\lambda}$ of $\beta_1(\lambda) = 0$ is our estimate for λ.

To estimate $\mathbb{E}\exp(\lambda E_n)$ efficiently, we used Bundschuh's importance sampling methods (21), which apply if the gap penalty is affine. Briefly, importance sampling is a variance-reduction technique for simulating rare events. In global alignment simulations, e.g. a large edge maximum is a rare event. By simulating optimal subsequence pairs in 'hybrid alignment' (a type of optimized Bayesian local alignment) (26), we ensured that our realizations frequently generated a large edge maximum E_n. Accordingly, we simulated a pair of sequences of some 'base length' $n = \underline{l}$. After correcting for biases induced by the importance sampling distribution, we estimated $\mathbb{E}\exp(\lambda E_{\underline{l}})$.

Equation 3 corresponds to an asymptotic equality with two free parameters to β_0 and $\beta_1(\lambda)$, which we estimated with robust regression. Robust regression was originally developed as an antidote to outliers (27), which badly skew least-square regression (28–31). As noted elsewhere (23), however, robust regression is also remarkably suited for extracting asymptotic parameters like β_0 and $\beta_1(\lambda)$.

Robust regression requires the specification of an influence function, to quantify the influence of potential outliers on the regression result. Many influence functions exist (27), but the Andrews function with $a = 1.339$ [(27), p. 388; (29)] works well in asymptotic regression, because it ignores points that obviously lie outside the asymptotic regime (23).

Accordingly, we applied robust regression to Equation 3. To solve $\beta_1(\lambda) = 0$, let λ_u be the scale parameter for ungapped local alignment, which can be determined analytically. Because $0 \leqslant \lambda \leqslant \lambda_u$, with repeated bisection of the interval $[0, \lambda_u]$ yielded an estimate $\hat{\lambda}$ for the root of the equation $\beta_1(\lambda) = 0$. In practice, multiple roots did not occur.

Numerical scheme for k

Next, we estimated $\mathbb{P}(M = y)$ and $\mathbb{E}N(y)$. Importance sampling has already generated sequence-pairs of base length $\underline{l}$ for estimating λ. The bias in importance sampling tends to yield large global scores S_{ij}, ascending toward the global maximum M. To determine $N(y)$, we needed to simulate and count all vertices with global scores $S_{ij} = y$. Therefore, we extended the sequence pair beyond the base length $\underline{l}$ using random letters with the unbiased Robinson and Robinson frequencies. The global scores S_{ij} beyond the base length $\underline{l}$ became progressively smaller, thereby permitting determination of $N(y)$.

Given $\varepsilon > 0$, we simulated a random number $\bar{L}$ of unbiased letters in each sequence, until we found some total length $L = \underline{l} + \bar{L}$ such that

$$(2L + 1)\exp\{-\lambda(M_L - E_L)\} \leqslant \varepsilon. \quad \mathbf{4}$$

The edge maximum E_L is a maximum over $2L + 1$ vertices. Therefore, for small enough stringencies $\varepsilon > 0$, if the edge maximum E_L of the contributing $2L + 1$ vertices satisfies Equation 4, it is probable that $M = M_L$, because elongating the sequences is unlikely to increase the estimate of M. Similarly, the elongation does not increase the estimate of $\mathbb{E}N(y)$ much. After appropriate averaging, our simulations therefore yielded estimates $\hat{\mathbb{P}}(M = y) \approx \mathbb{P}(M_L = y)$ and $\hat{\mathbb{E}}N(y) \approx \mathbb{E}N_L(y)$ for $\mathbb{P}(M = y)$ and $\mathbb{E}N(y)$.

With the simulation estimates $\hat{\lambda}$, $\hat{\mathbb{P}}(M = y)$ and $\hat{\mathbb{E}}N(y)$ in hand, we found that errors in $\hat{\lambda}$ were negligible in practice. In contrast, the standard deviations sample (32) of $\hat{\mathbb{P}}(M = y)$ and $\hat{\mathbb{E}}N(y)$, denoted by s_M and s_N, were not.

We calculated an estimate $\hat{k}_y$ for k_y by substituting $\hat{\lambda}$, $\hat{\mathbb{P}}(M = y)$, and $\hat{\mathbb{E}}N(y)$ into Equation 2. We estimated the error $s(\hat{k}_y)$ in $\hat{k}_y$ from the equation

$$s(\hat{k}_y) = \max\left|\frac{e^{\hat{\lambda}y}}{1-e^{-\hat{\lambda}}} \cdot \frac{\left[\hat{\mathbb{P}}(M = y) \pm s_M\right]^2}{\hat{\mathbb{E}}[N(y)] \pm s_N} - \hat{k}_y\right|. \quad \mathbf{5}$$

Note that Equation 5 explicitly neglects the error in the estimate $\hat{\lambda}$.

Finally, we used robust regression to extract a summary estimate $\hat{k}$ from the estimates $\hat{k}_y \pm s(\hat{k}_y)$ for individual y. To begin with, consider a constant regression model $\boldsymbol{\eta} = \mathbf{1}\alpha + \mathbf{e}$, where $\boldsymbol{\eta}$ is a column vector consisting of the values $\hat{k}_y$, $\mathbf{1}$ is a column vector whose elements are all 1, the constant α is the summary estimate $\hat{k}$, and $\mathbf{e}$ is the column vector consisting of the errors $s(\hat{k}_y)$.

Our ultimate aim is to compute $\hat{k}$ rapidly, with as few realizations as possible. Unfortunately, for small numbers of realizations, the errors s_M and s_N are correlated with the corresponding estimates $\hat{\mathbb{P}}(M = y)$ and $\hat{\mathbb{E}}N(y)$. The correlations propagate to $s(\hat{k}_y)$, noticeably biasing the summary estimate $\hat{k}$, with $\mathbb{E}\hat{k} < k$ (see Figure 1).

To avoid the bias, we applied the constant regression model $\boldsymbol{\eta}' = \mathbf{1}\alpha' + \mathbf{e}'$ to the errors $s(\hat{k}_y)$ themselves. The elements of the column vector $\boldsymbol{\eta}'$ were the errors $s(\hat{k}_y)$, with errors in each $s(\hat{k}_y)$ is taken to be a constant s derived though a standard formula [(27), p. 387], $\mathbf{e}' = \mathbf{1}s$. Robust regression thus gave a constant estimate $\alpha' = \hat{s}(\hat{k})$ of the errors $s(\hat{k}_y)$. We substituted the constant error estimate $\mathbf{e} = \mathbf{1}\alpha' = \mathbf{1}\hat{s}(\hat{k})$ back into the constant regression $\boldsymbol{\eta} = \mathbf{1}\alpha + \mathbf{e}$ of $\hat{k}_y$ to derive a robust regression estimate $\hat{k}$ for k. Although somewhat *ad hoc*, the constant regression of the errors successfully reduced biases (see Figure 3).

Even for large simulations (e.g. 10^6 realizations), however, sampling of the event $[M = y]$ was inadequate for many large y, with $\mathbb{P}(M = y)$ likely being underestimated. Although the corresponding average was unbiased (in theory, at least), we suspect that it had a distribution whose skewing increased with y. Consequently, for large y, $\hat{k}_y$ often slightly underestimated the true k, with improbable but substantial overestimations maintaining a correct expectation $\mathbb{E}\hat{k}_y = k$ (see Figure 2). The putative skewing also made the anticipated relation $\mathbb{P}(M = y) \approx e^{\lambda}\,\mathbb{P}(M = y + 1)$ fail for large y. To avoid skewing, we therefore restricted robust regression of $\hat{k}_y$ to the range $[a, b]$ of y that minimized the function

$$f(a, b) = \frac{1}{(b - a + 1)}\sum_{y=a}^{b}\left|\frac{\mathbb{P}(M = y)}{\mathbb{P}(M \geqslant y)} - \left(1 - e^{-\lambda}\right)\right|. \quad \mathbf{6}$$

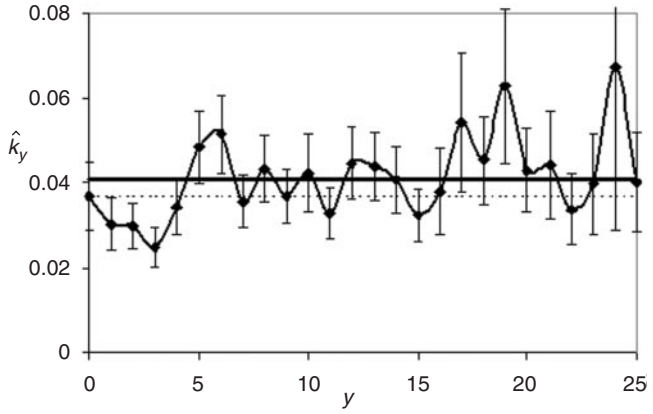

Figure 1. Plot of estimates for $\hat{k}_y$ against the global score y for the BLOSUM62 scoring matrix with an affine gap cost of $11 + g$ for a gap of length g, with random sequences whose letters are chosen according to the empirical Robinson and Robinson amino acid frequencies (3). Each point represents 30 000 random sequence-pairs generated by the importance sampling method with base length $\underline{l} = 50$ and extended to random length L using Equation 4 with $\varepsilon = 10^{-2}$. The error bars indicate the error estimate $s(\hat{k}_y)$. The horizontal thick line $k = 0.041$ represents the previous best estimate of the Gumbel pre-factor k (20). The dotted line $\hat{k} = 0.036$ shows an example of the biased summary estimate $\hat{k}$ from the robust regression, which we ascribe to the correlation between $s(\hat{k}_y)$ and $\hat{k}_y$.

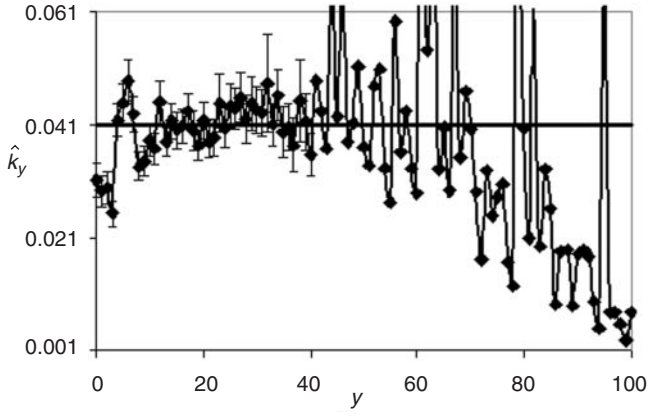

Figure 2. Plot of estimates for $\hat{k}_y$ against the global score y for 10^6 realizations. The simulation conditions were the same as in Figure 1. The error bars showing $s(\hat{k}_y)$ for the under-sampled asymptotic regime $y \in [41\ 100]$ are large and are omitted.

Software and Hardware

Computer code was written in C++ and compiled with the Microsoft® Visual C++® 6.0 compiler. The computer had a single Intel® Pentium® 4 2.8 GHz processor with 0.5 GB RAM and employed the Microsoft® Windows® 2000 operating system.

RESULTS

Tables 1 and 2 give estimates of the Gumbel parameters λ and k for all online options of the BLASTP parameters. They therefore confirm that our simulations and our formulas for k produced correct results. Other figures show results for the BLASTP default parameters, namely, the Robinson and Robinson amino acid frequencies (3), the BLOSUM62 scoring matrix and the gap cost $\Delta(g) = 11 + g$. Other BLAST parameters tested gave comparable results, unless indicated otherwise (data not shown).

Table 1. Estimates of λ for all online options of the BLASTP parameters

Scoring matrix	Gap cost $\Delta(g)$	λ	Average $\hat{\lambda}$	Standard error $\hat{\lambda}$	Relative error $\hat{\lambda}$ (%)
BLOSUM45	$15 + 2g$	0.203	0.2039	0.00061	0.30
BLOSUM62	$11 + g$	0.267	0.2678	0.00088	0.33
BLOSUM80	$10 + g$	0.299	0.3000	0.00056	0.19
PAM30	$9 + g$	0.294	0.2931	0.00035	0.12
PAM70	$10 + g$	0.291	0.2914	0.00037	0.13

All results used 100 simulations of 30 000 realizations each. In Table 1, the first and second column give the BLASTP parameter options. The third column gives λ from the online BLASTP documentation. The fourth column gives the average estimate $\hat{\lambda}$ from 100 simulations. The fifth column gives the corresponding standard error in $\hat{\lambda}$ (so the standard error mean, the actual accuracy of our results, is 0.1 times the standard error). The sixth column gives the percent relative error in $\hat{\lambda}$, as calculated from the fourth and fifth columns.

Table 2. Estimates of k for all online options of the BLASTP parameters

Scoring matrix	Gap cost $\Delta(g)$	k	Average $\hat{k}$	Standard error $\hat{k}$	Relative error $\hat{k}$ (%)
BLOSUM45	$15 + 2g$	0.041	0.0401	0.0024	5.99
BLOSUM62	$11 + g$	0.041	0.0410	0.0027	6.59
BLOSUM80	$10 + g$	0.071	0.0706	0.0044	6.23
PAM30	$9 + g$	0.110	0.1051	0.0108	10.27
PAM70	$10 + g$	0.091	0.0899	0.0079	8.79

All results used 100 simulations of 30 000 realizations each. Table 2 has the same format as Table 1.

Empirically, simulations using BLASTP default parameters needed a base length of $\underline{l} = 50$ and a stringency $\varepsilon = 10^{-2}$ for the accuracies required for (λ, 1%; k, 10%). For scoring matrices with more dominant diagonals than BLOSUM62, shorter base lengths sufficed, (e.g. for PAM30, $\underline{l} = 15$ sufficed).

Figure 1 plots the estimates $\hat{k}_y$ with their standard error bars $s(\hat{k}_y)$ against global score y, up to $y = 25$. Each point represents 30 000 realizations. The horizontal thick line represents the previous best estimate $k \approx 0.041$ and the dotted line, the biased summary estimate $\hat{k} = 0.036$ due to the positive correlation between $\hat{k}_y$ and $s(\hat{k}_y)$. Therefore Figure 1 motivated us to regress the errors in $\hat{k}_y$, to produce a constant error estimate $\hat{s}(\hat{k})$, as described in the Materials and Methods.

Figure 2 plots the estimates $\hat{k}_y$ against global score y, up to $y = 100$. Each point represents 10^6 realizations. We obtained the estimate $\hat{\lambda}$ and used it to estimate $\hat{k}_y$. The range $y \in [0, 3]$ is not asymptotic, so the $\hat{k}_y$ do not approximate the true k very well. The range $y \in [4, 40]$ is asymptotic, and it is adequately sampled, so the $\hat{k}_y$ fluctuate randomly around the true k. The range $y > 40$ is also asymptotic, but it is not adequately sampled, so the $\hat{k}_y$ usually underestimate the true k. Figure 2 motivated us to regress only in the range $[a, b]$ minimizing Equation 6, as described in the Materials and Methods.

Figure 3 plots the relative errors of the summary estimate $\hat{k}$ using $\hat{k}_y \pm s(\hat{k}_y)$ (with skewed error estimates $s(\hat{k}_y)$) and those using $\hat{k}_y \pm \hat{s}(\hat{k})$ (with constant error estimate $\hat{s}(\hat{k})$) against different numbers of realizations). All errors in $\hat{k}$ were

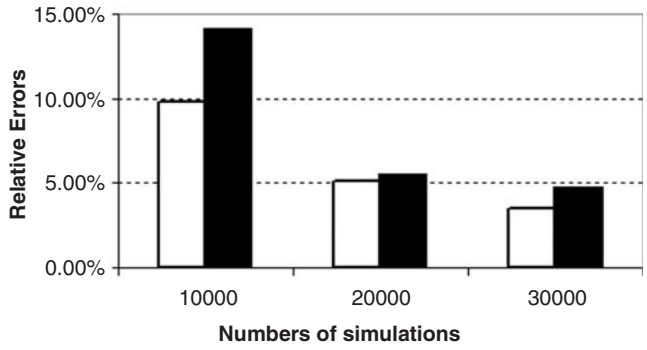

Figure 3. Plot of relative errors of estimate k obtained via robust regression using $\hat{k}_y \pm \hat{s}(\hat{k})$ and $\hat{k}_y \pm s(\hat{k}_y)$ against different numbers of simulations. Each bar represents an average over 20 absolute relative errors. The previous best estimate $k = 0.041$ is used as a basis for the relative error calculation. The relative errors from $\hat{k}_y \pm \hat{s}(\hat{k})$ are shown with white bars; the one from $\hat{k}_y \pm s(\hat{k}_y)$ with black bars.

computed relative to the approximation $k \approx 0.041$. Each error plotted is the average of the absolute relative error for 20 independent simulations, each using the indicated number of realizations. White bars show the results for $\hat{k}_y \pm \hat{s}(\hat{k})$; black bars, for $\hat{k}_y \pm s(\hat{k}_y)$. For 10 000 realizations, the constant error estimate $\hat{s}(\hat{k})$ reduces the relative errors dramatically. As the number of realizations increases, the difference in efficiency of estimation between $\hat{k}_y \pm s(\hat{k}_y)$ and $\hat{k}_y \pm \hat{s}(\hat{k})$ decreases. Figure 3 shows that 10 000 realizations estimated k with less than 10% relative error. The same 10 000 realizations also estimated $\hat{\lambda}$ with less than 0.8% relative error (data not shown).

The simulations of Figure 3 estimated $\hat{k}$ from 10 000 realizations, in less than 30 s. For comparison, the same simulations could have estimated $\hat{\lambda}$ in less than 7 s. For the PAM 30 matrix with $\Delta(g) = 9 + g$, they estimated λ and k in less than 4 s.

DISCUSSION

BLAST programs (BLASTP, PSI-BLAST, etc.) are restricted to specific scoring schemes, because time-consuming local alignment simulations for estimating the corresponding Gumbel parameters must be done offline. However, simulations of *global* alignment can estimate the Gumbel scale parameter λ for *local* alignment (6). Some global alignment methods are as much as five times faster than the best local alignment methods (21,23), so global alignment has considerable potential for online estimation of the Gumbel parameter λ.

This paper surmounts an obstacle to online estimation by demonstrating that simulations of global alignment can determine the Gumbel pre-factor k. Table 2 displays the results of global alignment simulations over a wide range of BLAST parameters, all of which gave correct estimates of the corresponding k and supported the validity of our methods for computing k.

Global alignment simulation therefore appears a feasible method for estimating both Gumbel parameters, λ and k. (The BLASTP default parameters provide a standard for quantifying speed, so the following results apply to the BLASTP defaults, unless stated otherwise.) With local alignment, estimates of λ required 40 000 sequence-pairs of minimum length 600 (21); with our methods, 5000 sequence-pairs of maximum length 50 (23). In fact, our methods attained 1.3% accuracies in λ with only 1000 sequence-pairs of maximum length 50. In our hands, k was more difficult to estimate than λ, with 10% relative errors requiring 10 000 sequence-pairs of average length 140. In summary, the methods presented here for estimating the Gumbel parameters λ and k represent at least a 3-fold improvement in speed over local alignments.

Online computation of the BLAST P-value requires more than the Gumbel parameters. It also requires an estimate of the 'finite-size effect' (10,13,33,34). Global alignment (or some variant of it) can indeed produce the required estimate (manuscript in preparation). Without the finite-size estimate in hand, however, we were not strongly motivated to incorporate technical improvements or heuristics into our methods. Bundschuh, e.g. implemented a diagonal-cutting heuristic to remove irrelevant off-diagonal elements in the global alignment matrix (21); we did not. The heuristic could probably speed our computation by a further factor of at least three.

Online BLAST estimation of the Gumbel parameters is likely just a few years away.

ACKNOWLEDGEMENTS

We would like to acknowledge helpful discussions with Dr Ralf Bundschuh and Dr Stephen Altschul. This work was supported in whole by the Intramural Research Program of the National Library of Medicine at National Institutes of Health/DHHS. Funding to pay the Open Access publication charges for this article was provided by National Library of Medicine at National Institutes of Health/DHHS.

Conflict of interest statement. None declared.

REFERENCES

1. Altschul,S.F., Gish,W., Miller,W., Myers,E.W. and Lipman,D.J. (1990) Basic local alignment search tool. *J Mol. Biol*., **215**, 403–410.
2. Altschul,S.F., Madden,T.L., Schaffer,A.A., Zhang,J., Zhang,Z., Miller,W. and Lipman,D.J. (1997) Gapped BLAST and PSI-BLAST: a new generation of protein database search programs. *Nucleic Acids Res*., **25**, 3389–3402.
3. Robinson,A.B. and Robinson,L.R. (1991) Distribution of glutamine and asparagine residues and their near neighbors in peptides and proteins. *Proc. Natl Acad. Sci. USA*, **88**, 8880–8884.
4. Henikoff,S. and Henikoff,J.G. (1992) Amino acid substitution matrices from protein blocks. *Proc. Natl Acad. Sci. USA*, **89**, 10915–10919.
5. Dayhoff,M.O., Schwartz,R.M. and Orcutt,B.C. (1978) *Atlas of Protein Sequence and Structure*. National Biomedical Research Foundation, Silver Spring, MD, Vol. 3, pp. 345–352.
6. Arratia,R. and Waterman,M.S. (1994) A phase transition for the score in matching random sequences allowing deletions. *Ann Appl Probab*., **4**, 200–225.
7. Dembo,A., Karlin,S. and Zeitouni,O. (1994) Limit distributions of maximal non-aligned two-sequence segmental score. *Ann Probab*., **22**, 2022–2039.
8. Karlin,S. and Altschul,S.F. (1990) Methods for assessing the statistical significance of molecular sequence features by using general scoring schemes. *Proc. Natl Acad. Sci. USA*, **87**, 2264–2268.
9. Mott,R. (1999) Local sequence alignments with monotonic gap penalties. *Bioinformatics*, **15**, 455–462.
10. Mott,R. (2000) Accurate formula for P-values of gapped local sequence and profile alignments. *J Mol. Biol*., **300**, 649–659.

11. Storey,J.D. and Siegmund,D. (2001) Approximate p-values for local sequence alignments: numerical studies. *J Comput. Biol.*, **8**, 549–556.
12. Siegmund,D. and Yakir,B. (2000) Approximate p-values for local sequence alignments. *Ann Stat.*, **28**, 657–680.
13. Altschul,S.F. and Gish,W. (1996) Local alignment statistics. *Methods Enzymol.*, **266**, 460–480.
14. Waterman,M.S. and Vingron,M. (1994) Rapid and accurate estimates of statistical significance for sequence data base searches. *Proc. Natl Acad. Sci. USA*, **91**, 4625–4628.
15. Olsen,R., Bundschuh,R. and Hwa,T. (1999) Rapid assessment of extremal statistics for gapped local alignment. *Proc. Int. Conf. Intell Syst Mol Biol.*, 211–222.
16. Mott,R. (1992) Maximum-Likelihood-Estimation of the Statistical Distribution of Smith-Waterman Local Sequence Similarity Scores. *B Math Biol.*, **54**, 59–75.
17. Smith,T.F., Waterman,M.S. and Burks,C. (1985) The statistical distribution of nucleic acid similarities. *Nucleic Acids Res.*, **13**, 645–656.
18. Collins,J.F., Coulson,A.F. and Lyall,A. (1988) The significance of protein sequence similarities. *Comput. Appl. Biosci.*, **4**, 67–71.
19. Mott,R. and Tribe,R. (1999) Approximate statistics of gapped alignments. *J Comput. Biol.*, **6**, 91–112.
20. Altschul,S.F., Bundschuh,R., Olsen,R. and Hwa,T. (2001) The estimation of statistical parameters for local alignment score distributions. *Nucleic Acids Res.*, **29**, 351–361.
21. Bundschuh,R. (2002) Rapid significance estimation in local sequence alignment with gaps. *J Comput. Biol.*, **9**, 243–260.
22. Grossmann,S. and Yakir,B. (2004) Large deviations for global maxima of independent superadditive processes with negative drift and an application to optimal sequence alignments. *Bernoulli*, **10**, 829–845.
23. Park,Y., Sheetlin,S. and Spouge,J.L. (2005) Accelerated convergence and robust asymptotic regression of the Gumbel scale parameter for gapped sequence alignment. *Journal of Physics A: MATHEMATICAL AND GENERAL*, **38**, 97–108.
24. Needleman,S.B. and Wunsch,C.D. (1970) A general method applicable to the search for similarities in the amino acid sequence of two proteins. *J Mol. Biol.*, **48**, 443–453.
25. Gotoh,O. (1982) An improved algorithm for matching biological sequences. *J Mol. Biol.*, **162**, 705–708.
26. Yu,Y.K. and Hwa,T. (2001) Statistical significance of probabilistic sequence alignment and related local hidden Markov models. *J Comput. Biol.*, **8**, 249–282.
27. Montgomery,D.C., Peck,E.A. and Vining,G.G. (2001) *Introduction to Linear Regression Analysis*. John Wiley & Sons, Inc. NY.
28. Andrews,D.F., Bickel,P.J., Hampel,F.R., Huber,P.J. and rogers,W.H., Tukey,K.W. (1972) *Robust Estimates of Location: Survey and advances*. Princenton University Press, Princenton, NJ.
29. Andrews,D.F. (1974) A robust method for multiple linear regression. *Technometrics*, **16**, 523–531.
30. Huber,P.J. (1964) Robust estimation of a location parameter. *Ann. Math. Statist.*, **35**, 73–101.
31. Huber,P.J. (1973) Robust regression: Asymptotics, conjectures and Monte Carlo. *Ann Stat*, **1**, 799–821.
32. Dwass,M. (1970) *Probability and Statistics*. W.A. Benjamin, NY.
33. Spouge,J.L. (2001) Finite-size corrections to Poisson approximations of rare events in renewal processes. *J Appl Probab.*, **38**, 554–569.
34. Spouge,J.L. (2005) Finite-Size Corrections to Poisson Approximations in General Renewal-Success Processes. *J Math Anal Appl.*, **301**, 401–418.
35. Spouge,J.L. (2004) Path Reversal and Islands in the Gapped Alignment of Random Sequences. *J Appl Probab.*, **41**, 975–983.
36. Smith,T.F. and Waterman,M.S. (1981) Identification of common molecular subsequences. *J. Mol. Biol.*, **147**, 195–197.
37. Aldous,D. (1989) *Probability approximations via the Poisson clumping heuristic*. 1st edn. Springer-Verlag, NY.

APPENDIX

In the Appendix, we give a heuristic derivation of Equation 2.

Notation for local sequence alignment

For local alignment, consider a pair $\hat{\mathbf{A}} = \ldots\hat{A}_{-1}\hat{A}_0\hat{A}_1\ldots$ and $\hat{\mathbf{B}} = \ldots\hat{B}_{-1}\hat{B}_0\hat{B}_1\ldots$ of doubly-infinite sequences. Their local alignment graph $\hat{\Gamma}$ is a directed, weighted lattice graph in two dimensions, as follows. The vertices v of $\hat{\Gamma}$ are $v = (i, j) \in \mathbb{Z}^2$, the *entire* two-dimensional integer lattice. In other respects, particularly with respect to the edges between its vertices, $\hat{\Gamma}$ has the same structure as the global alignment graph Γ.

We base the graph $\hat{\Gamma}$ on the entire two-dimensional integer lattice $\mathbb{Z}^2$ because of our interest in the Gumbel distribution. In intuitive terms, the BLAST E-value E_y follows the Gumbel distribution, only if the local alignment does not 'see' the ends of the sequences, so finite-size effects can be neglected (13,33).

Let $\hat{\mathbf{\Pi}}_{ij}$ be the set of all paths π ending at $v_h = (i,j)$, regardless of their starting vertex. Define the 'local score' $\hat{S}_{ij} = \max\{W_\pi : \pi \in \hat{\mathbf{\Pi}}_{ij}\}$. The paths π ending at $v_k = (i,j)$ with local score $W_\pi = \hat{S}_{ij}$ are 'optimal local paths' corresponding to 'optimal local alignments' matching subsequences of $\hat{\mathbf{A}}$ and $\hat{\mathbf{B}}$ up to and including the letters $\hat{A}_i$ and $\hat{B}_j$.

Unlike the singly-infinite sequences $\mathbf{A}$ and $\mathbf{B}$, the doubly-infinite sequences $\hat{\mathbf{A}}$ and $\hat{\mathbf{B}}$ correspond to the entire lattice $\mathbb{Z}^2$. The lattice $\mathbb{Z}^2$ is invariant under translation (i.e. it appears the same from each of its vertices). Thus, if $\hat{\mathbf{A}}$ and $\hat{\mathbf{B}}$ are sequences with independent random letters, the corresponding local scores $\hat{S}_{ij}$ are 'stationary' (i.e. their joint distribution is invariant under translation). Stationary scores carry a prime elsewhere (i.e. $\hat{S}'_{ij}$) (35), which we drop here for brevity. For many purposes, translation invariance renders all vertices in $\mathbb{Z}^2$ equivalent, so it usually suffices to define quantities below solely at the origin, (0,0). The definition at other vertices is usually left implicit.

If the sequences $\hat{\mathbf{A}}$ and $\hat{\mathbf{B}}$ were singly-infinite, the Smith–Waterman algorithm could compute the corresponding local scores $\hat{S}_{ij}$ (36). Although the algorithm is unable to compute $\hat{S}_{ij}$ for $\hat{\mathbf{A}}$ and $\hat{\mathbf{B}}$, a rigorous treatment shows that doubly-infinite sequences pose no essential difficulties in the logarithmic regime (35).

For efficiency, many simulations of random local alignments partition the vertices in $\mathbb{Z}^2$ into 'islands' (described below). To avoid technical nuisances, each vertex must belong to exactly one island, so we define the following strict total order on $\mathbb{Z}^2$: $(i',j') \prec (i,j)$, if and only if either $i' + j' < i + j$ or else, $i' + j' = i + j$ and $j' < j$.

Let us say that a vertex $(i,j) \in \mathbb{Z}^2_+$ 'belongs to' the origin if (0,0) is the greatest vertex $v_0 = (i',j')$ (under the total order $\prec$) such that $\hat{S}_{ij} = W_\pi$, for some path π starting at $v_0 = (i',j')$ and ending at $v_h = (i,j)$. The 'island' belonging to (0,0) is the set $\mathbb{B}_{00} \subseteq \mathbb{Z}^2_+$ of all vertices (i,j) belonging to (0, 0), and we say that (0, 0) 'owns' the island. [Equation 12 below uses the translate $\mathbb{B}_{-i,-j}$ of the set $\mathbb{B}_{00}$, where $\mathbb{B}_{-i,-j}$ is the set of all vertices belonging to $(-i,-j)$].

By the following reasoning, $\mathbb{B}_{00}$ is empty if and only if $\hat{S}_{00} > 0$. First, if $\hat{S}_{00} > 0$, there is some path π' ending at (0, 0) with a positive score. If (0, 0) owned any vertex (i, j), there would be a path π starting at (0, 0) and ending at (i, j) with $\hat{S}_{ij} = W_\pi$. Then, the path concatenating π and π' would have a weight exceeding $\hat{S}_{ij} = W_\pi$, contrary to the definition of $\hat{S}_{ij}$. Thus, if $(0,0)$ owns some vertex, $\hat{S}_{00} = 0$. Conversely, if $\hat{S}_{00} = 0$, then by deliberate construction, the definition of the total order $\prec$ implies that $(0,0)$ owns itself [because the weight of the trivial path containing only $(0,0)$ is 0].

Accordingly, define the 'local maximum' [implicitly, on the island $\mathbb{B}_{00}$ belonging to (0, 0)] as $\hat{M} = \max\{\hat{S}_{ij} : (i,j) \in \mathbb{B}_{00}\}$, with the default $\hat{M} = -\infty$, if $\mathbb{B}_{00}$ is empty (i.e. if $\hat{S}_{00} > 0$). Let $\hat{N}(y) = \#\{(i,j) \in \mathbb{B}_{00} : \hat{S}_{ij} = y\}$ denote the number of island vertices with local score y.

To connect our quantities explicitly to the Gumbel parameters, define $\hat{M}_{mn} = \max\{\hat{S}_{ij} : 0 \leqslant i \leqslant m,\ 0 \leqslant j \leqslant n\}$, the maximum local score in the lattice rectangle $[0, m] \times [0, n]$. Let ρ_y the density of islands yielding a local score $\hat{S}_{ij} \geqslant y$, or equivalently, the density of their owners in $\mathbb{Z}^2$. Under certain conditions in the logarithmic regime, $\mathbb{P}(\hat{M}_{mn} \geqslant y) = P_y \approx 1 - \exp(-E_y)$, where as $m, n \to \infty$,

$$E_y = \rho_y mn \approx kmne^{-\lambda y}. \quad \mathbf{7}$$

Simulations indicate that to a good approximation, islands yielding a large local score $\hat{S}_{ij}$ occur independently of each other (15). Therefore, Equation 7 asserts that $\rho_y \approx ke^{-\lambda y}$. In a Poisson approximation, ρ_y represents the intensity of the Poisson process on $\mathbb{Z}^2$ that generates the owners of islands yielding a local score $\hat{S}_{ij} \geqslant y$.

Because of translation invariance, the density ρ_y equals the probability that any particular vertex in $\mathbb{Z}^2$ [e.g. (0, 0)] owns an island yielding a local score $\hat{S}_{ij} \geqslant y$. In other words, $\mathbb{P}(\hat{M} \geqslant y) = \rho_y \approx ke^{-\lambda y}$. Thus, the limit

$$k = \lim_{y \to \infty} e^{\lambda y} \mathbb{P}(\hat{M} \geqslant y) \quad \mathbf{8}$$

exists and equals the pre-factor k.

Path reversal identity

To determine k from global alignments, we first relate the global maximum M to the local scores $\hat{S}_{ij}$ with a path reversal identity. Recall the global maximum $M = \max\{W_\pi : \pi \in \Pi\}$, where Π is the set of all paths π in $\mathbb{Z}^2_+$ starting at $v_0 = (0, 0)$. Recall also the local score $\hat{S}_{ij} = \max\{W_\pi : \pi \in \hat{\Pi}_{ij}\}$, where $\hat{\Pi}_{ij}$ is the set of all paths π in $\mathbb{Z}^2$ ending at $v_h = (i, j)$.

It is believable that for any fixed $(i, j) \in \mathbb{Z}^2$, each path in $\hat{\Pi}_{ij}$ with random edge-weights corresponds to a reversal of a path in Π with the same random edge-weights. Thus, for every $(i, j) \in \mathbb{Z}^2$, $\mathbb{P}(\hat{S}_{ij} = y) = \mathbb{P}(M = y)$, i.e. the local score and the global maximum have the same distribution. (Note: the equality is solely distributional. In any particular random instance, the local score $\hat{S}_{ij}$ and global maximum score M are unlikely to be related.)

Because the distributional equality holds for every $(i, j) \in \mathbb{Z}^2$, we drop the subscript ij on $\hat{S}_{ij}$ and write

$$\mathbb{P}(\hat{S} = y) = \mathbb{P}(M = y). \quad \mathbf{9}$$

A formal proof of Equation 9 can be found elsewhere (35).

The Poisson clumping heuristic

Consider the Poisson clumping heuristic (37)

$$\mathbb{P}(\hat{S} = y) = \mathbb{P}(\hat{M} \geqslant y)\mathbb{E}[\hat{N}(y)|\hat{M} \geqslant y]. \quad \mathbf{10}$$

Equation 10 states that at any fixed vertex $(i, j) \in \mathbb{Z}^2$, the probability that $\hat{S}_{ij} = y$ is the density of vertices with a local score y. This density equals $\rho_y = \mathbb{P}(\hat{M} \geqslant y)$, the density of islands where some local score is at least y, multiplied by $\mathbb{E}[\hat{N}(y)|\hat{M} \geqslant y]$, the expected number $\hat{N}(y)$ of island vertices (i, j) where the local score $\hat{S}_{ij} = y$, is given $\hat{M} \geqslant y$.

Equation 10 can be demonstrated as follows. First,

$$\mathbb{E}\hat{N}(y) = \mathbb{P}(\hat{M} \geqslant y)\mathbb{E}[\hat{N}(y)|\hat{M} \geqslant y], \quad \mathbf{11}$$

because if $\hat{M} < y$, then $\hat{N}(y) = 0$. Equation 11 follows, because the event $[\hat{M} < y]$ contributes nothing to the expectation on the left.

Next, define the indicator $\mathbb{I}A = 1$ if the event A occurs and $\mathbb{I}A = 0$ otherwise. Then,

$$\begin{aligned}\mathbb{E}\hat{N}(y) &= \mathbb{E}\sum_{(i,j)\in\mathbb{Z}^2_+} \mathbb{I}[\hat{S}_{ij} = y \text{ and } (i,j) \in \mathbb{B}_{00}] \\ &= \mathbb{E}\sum_{(i,j)\in\mathbb{Z}^2_+} \mathbb{I}[\hat{S}_{00} = y \text{ and } (0,0) \in \mathbb{B}_{-i,-j}] \\ &= \mathbb{P}(\hat{S}_{00} = y). \quad \mathbf{12}\end{aligned}$$

The first equality is essentially the definition of $\hat{N}(y)$, which counts the number of vertices belonging to (0,0) with local score $\hat{S}_{ij} = y$. The second equality exploits the translation invariance of the probabilities associated with $\hat{S}_{ij}$. The third inequality merely notes that in the logarithmic regime, (0,0) must belong to some vertex (35). Equation 10 follows.

Our speculations

Based on the success of our simulation results, we speculate. First,

$$\lim_{y\to\infty} \frac{\mathbb{E}[\hat{N}(y)|\hat{M} \geqslant y]}{\mathbb{E}[N(y)|M \geqslant y]} = 1. \quad \mathbf{13}$$

In fact, $lim_{y\to\infty}\mathbb{E}[\hat{N}(y)|\hat{M} \geqslant y]$ and $lim_{y\to\infty}\mathbb{E}[N(y)|M \geqslant y]$ are likely to exist as a common finite limit, but Equation 13 suffices for present purposes.

Equation 13 can be justified intuitively, as follows. As $y \to \infty$, any vertices satisfying $S_{ij} = y$ become likely to cluster on a single island that has a large maximum local score. Thus, given $M \geqslant y$, the vertices with $S_{ij} = y$ have a comparable structure to vertices with $\hat{S}_{ij} = y$ on the island belonging to (0, 0), given that the island satisfies $\hat{M} \geqslant y$. In particular, given $M \geqslant y$, the number $N(y)$ of vertices with $S_{ij} = y$ has a similar random behaviour to the number $\hat{N}(y)$ of vertices with $\hat{S}_{ij} = y$, given $\hat{M} \geqslant y$. Thus, the expectations approximate each other: $\mathbb{E}[\hat{N}(y)|\hat{M} \geqslant y] \approx \mathbb{E}[N(y)|M \geqslant y]$.

Though hardly a 'speculation', we assume that $c = \lim_{y\to\infty} e^{\lambda y}\,\mathbb{P}(M \geqslant y)$ exists. Unfortunately, still there is no rigorous proof of the limit's existence.

The formula for *k* from global alignment

Equation 11 has an analog for global alignment, with a similar demonstration:

$$\mathbb{E}N(y) = \mathbb{E}[N(y)|M \geqslant y]\mathbb{P}(M \geqslant y). \quad \mathbf{14}$$

Together, Equations 8–10, 13 and 14 yield

$$k = \lim_{y\to\infty} e^{\lambda y}\mathbb{P}\left(\hat{M} \geqslant y\right) = \lim_{y\to\infty} \frac{e^{\lambda y}\mathbb{P}(M=y)}{\mathbb{E}\left[\hat{N}(y)|\hat{M} \geqslant y\right]}$$
$$= \lim_{y\to\infty} \frac{e^{\lambda y}\mathbb{P}(M=y)}{\mathbb{E}[N(y)|M \geqslant y]} = \lim_{y\to\infty} \frac{e^{\lambda y}\mathbb{P}(M=y)\mathbb{P}(M \geqslant y)}{\mathbb{E}N(y)}.$$

Recall our assumption that $s(A_i, B_j)$ and $\Delta(g)$ are always integers:

$$\lim_{y\to\infty} \frac{\mathbb{P}(M=y)}{\mathbb{P}(M \geqslant y)} = \lim_{y\to\infty} \frac{\mathbb{P}(M \geqslant y) - \mathbb{P}(M \geqslant y+1)}{\mathbb{P}(M \geqslant y)} = 1 - e^{-\lambda}. \quad \textbf{16}$$

Let $k_y = e^{\lambda y}\mathbb{P}\left(\hat{M} \geqslant y\right)$. From Equations 15 and 16, $k = \lim_{y\to\infty} \mathrm{k}_y$, where k_y is given by Equation 2.