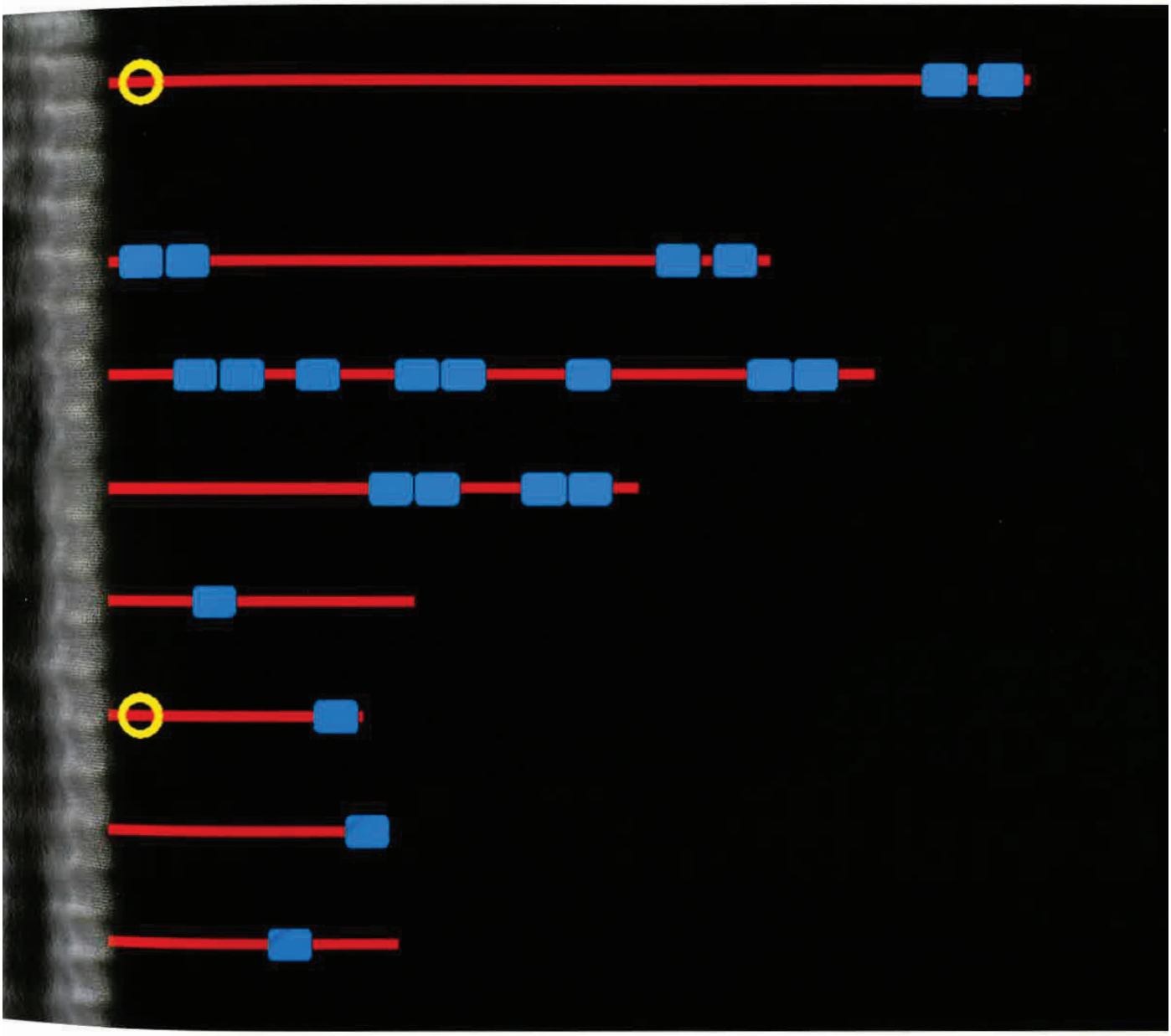
VOLUME 25 NUMBER 17 SEPTEMBER 1, 1997

Nucleic Acids Research

NAR ONLINE
http://www.oup.co.uk/nar



OXFORD UNIVERSITY PRESS

ISSN 0305 1048 Coden NARHAD

Nucleic Acids Research

LIBRARY OF CONGRESS
SEP 18 1997
COPY
COPYRIGHT OFFICE

Executive Editors

R.T.Walker, *Birmingham, UK*
R.J.Roberts, *Beverly, MA, USA*
K.Calame, *New York, NY, USA*
M.J.Gait, *Cambridge, UK*
H.J.Gross, *Würzburg, Germany*
R.I.Gumport, *Urbana, IL, USA*
R.B.Hallick, *Tucson, AZ, USA*
N.C.Martin, *Louisville, KY, USA*
R.T.Simpson, *University Park, PA, USA*

Short Methods Editors: P.B.Becker, B.Séraphin, *Heidelberg, Germany*

Editorial Board

A.Bairoch, *Geneva, Switzerland*
M.Beato, *Marburg, Germany*
M.Belfort, *Albany, NY, USA*
M.Busslinger, *Vienna, Austria*
H.Cedar, *Jerusalem, Israel*
M.L.DePamphilis, *Nutley, NJ, USA*
W.Dynan, *Augusta, GA, USA*
F.Eckstein, *Göttingen, Germany*
J.D.Engel, *Evanston, IL, USA*
I.C.Eperon, *Leicester, UK*
D.J.Finnegan, *Edinburgh, UK*
M.Gellert, *Bethesda, MD, USA*
R.Giegé, *Strasbourg, France*
L.A.Grivell, *Amsterdam, The Netherlands*
W.Gruissem, *Berkeley, CA, USA*
W.Guschlbauer, *Gif-sur-Yvette, France*
S.E.Halford, *Bristol, UK*
D.Hawley, *Eugene, OR, USA*
J.E.Hearst, *Berkeley, CA, USA*
C.Hélène, *Paris, France*
A.J.Jeffreys, *Leicester, UK*
S.-H.Kim, *Berkeley, CA, USA*
A.R.Krainer, *CSHL, NY, USA*
R.Krumlauf, *London, UK*
A.Lamond, *Dundee, UK*
D.S.Latchman, *London, UK*
C.J.Leaver, *Oxford, UK*
D.M.J.Lilley, *Dundee, UK*
S.Linn, *Berkeley, CA, USA*
J.T.Lis, *Ithaca, NY, USA*
R.Luhrmann, *Marburg, Germany*
M.McClelland, *San Diego, CA, USA*
W.R.McClure, *Pittsburgh, PA, USA*
M.Muramatsu, *Saitama, Japan*
E.Ohtsuka, *Sapporo, Japan*
H.Okayama, *Tokyo, Japan*
M.V.Olson, *Seattle, WA, USA*
M.G.Parker, *London, UK*
R.K.Patient, *London, UK*
M.Paule, *Fort Collins, CO, USA*
A.M.Pyle, *New York, NY, USA*
J.M.Rosen, *Houston, TX, USA*
H.Ruterjans, *Frankfurt, Germany*
C.W.Schmid, *Davis, CA, USA*
P.M.Sharp, *Nottingham, UK*
L.Simpson, *Los Angeles, CA, USA*
S.T.Smale, *Los Angeles, CA, USA*
C.A.Stein, *New York, NY, USA*
G.Stormo, *Seattle, WA, USA*
R.H.Symons, *Adelaide, Australia*
K.Taira, *Tsukuba, Japan*
I.Tinoco, Jr, *Berkeley, CA, USA*
B.Van Ness, *Minneapolis, MN, USA*
A.M.Weiner, *New Haven, CT, USA*
S.C.West, *South Mimms, Herts, UK*
J.A.Wise, *Cleveland, OH, USA*
G.Zon, *Foster City, CA, USA*

Editorial and Production

Joy Walker, *UK Editorial Office*
Carol Cook, *US Editorial Office*
Sarah Brennan, *Production Editor*
Jacqueline Grainger, *Production Assistant*
Colette Dunne, *Production Assistant*

OXFORD UNIVERSITY PRESS

Subscriptions

Nucleic Acids Research is published twice monthly (24 issues) with an Annual Author and Subject Index. Subscriptions are entered on a calendar year basis only and are available as a printed version only or as a printed version plus online access. Prices include postage by surface mail or, for subscribers in the USA and Canada by air freight or in India, Japan, Australia and New Zealand, by Air Speeded Post. Airmail rates are available on request.

Annual subscription rate (Volume 25, 1997):
Institutional
Print only: UK and Europe £840, Rest of World $1425.
Print and Online site licence: UK and Europe £1440, Rest of World $2425.
Personal
Print only: UK and Europe £235, Rest of World $360.
Print and Online: UK and Europe £275, Rest of World $425.
Online only: UK and Europe £60, Rest of World $95. (Only available to individuals whose libraries have a print subscription.)
Back volume prices are available on request.

Orders. Orders and payments from, or on behalf of, subscribers in the various geographical areas shown below should be sent to the Press office indicated.

The Americas: Oxford University Press Inc., 2001 Evans Road, Cary, NC 27513, USA.

Japan: available from the following agents:
Kinokuniya Company Ltd, Journal Department, PO Box 55, Chitose, Tokyo, 156 Japan.
Maruzen Company Ltd, Journal Division, PO Box 5050, Tokyo International, 100-31 Japan.
Usaco Corporation, 13-12, Shimbashi 1-chome, Minato-ku, Tokyo, 105 Japan.

Rest of the World: Journals Subscriptions Department, Oxford University Press, Great Clarendon Street, Oxford OX2 6DP, UK.
Tel: (+441865 or 01865) 267907; Fax: (+441865) 267485;
Email: jnl.info@oup.co.uk

Advertising. To advertise in *Nucleic Acids Research* contact Oxford University Press (US office) in the Americas or Oxford University Press (UK office) in the Rest of the World (see addresses above).

© Oxford University Press 1997. All rights reserved; no part of this publication may be reproduced, stored in a retrieval system, or transmitted in any form or by any means, electronic, mechanical, photocopying, recording, or otherwise without either the prior written permission of the Publishers, or a licence permitting restricted copying issued in the UK by the Copyright Licensing Agency Ltd, 90 Tottenham Court Road, London WIP 9HE, or in the USA by the Copyright Clearance Center, 27 Congress Street, Salem, MA 01970. For those in the US/Canada not registered with CCC, articles can be obtained by fax in 48 hours by calling: WISE for Medicine™ 1-800-667-WISE.

Back volumes of this journal are available in 16 mm microfilm, 35 mm microfilm and 105 microfiche from University Microfilms International, 300 North Zeeb Road, Ann Arbor, MI 48106-1346, USA. Copies of articles published are also available from UMI.

Nucleic Acids Research (ISSN 0305-1048) is published twice monthly by Oxford University Press, Oxford, UK. Annual subscription price is US$1425. Nucleic Acids Research is distributed by M.A.I.L. America, 2323 Randolph Avenue, Avenel, New Jersey, NJ 07001, USA. Periodical postage paid at Rahway, NJ, USA and additional entry points.

US POSTMASTER: send address changes to *Nucleic Acids Research*, c/o M.A.I.L. America, 2323 Randolph Avenue, Avenel, New Jersey, NJ 07001, USA.

Printed by Information Press Ltd, Oxford, UK on acid free paper.

Cover: Schematic diagram representing the location of BRCT domains (rectangles) and RING finger domains (circles) within human BRCA1 and seven recently sequenced proteins: *Arabidopsis* T10M13.12, human KIAA0259, worm T13F2.3, fission yeast SPAC6G9.12, worm C36A4.8, *Synechocystis* sp. D90904, and human Pescadillo. Using the 215 C-terminal residues of BRCA1 as query, these proteins were retrieved during multiple iterations of the PSI-BLAST program. For further information, see the paper by Altschul *et al.* in this issue [*Nucleic Acids Res.* (1997) **25**, 3389–3402].

Authors submitting an article containing a photograph which may be suitable for use on the cover of *Nucleic Acids Research* are encouraged to contact an Executive Editor.

Nucleic Acids Research

Contents

Volume 25 number 17 September 1, 1997

SURVEY AND SUMMARY

Homing endonucleases: keeping the house in order 3379–3388
M.Belfort and R.J.Roberts

RNA

Evidence for a hydroxide ion bridging two magnesium ions at the active site of the hammerhead ribozyme 3421–3427
T.Hermann, P.Auffinger, W.G.Scott and E.Westhof

Inhibition of transcription by the TAR RNA of HIV-1 in a nuclear extract of HeLa cells 3445–3450
R.Yamamoto, S.Koseki, J.Ohkawa, K.Murakami, S.Nishikawa, K.Taira and P.K.R.Kumar

Cleavage of collagen RNA transcripts by hammerhead ribozymes *in vitro* is mutation-specific and shows competitive binding effects 3451–3458
G.Grassi, A.Forlino and J.C.Marini

Specific polyadenylation and purification of total messenger RNA from *Escherichia coli* 3465–3470
R.R.Amara and S.Vijaya

The RNase P RNA from cyanobacteria: short tandemly repeated repetitive (STRR) sequences are present within the RNase P RNA gene in heterocyst-forming cyanobacteria 3471–3477
A.Vioque

Expression of bovine mitochondrial $tRNA^{Ser}_{GCU}$ derivatives in *Escherichia coli* 3503–3507
I.Hayashi, G.Kawai and K.Watanabe

Antisense oligonucleotide binding to U5 snRNP induces a conformational change that exposes the conserved loop of U5 snRNA 3508–3513
G.Ast and A.M.Weiner

pH dependence of self-splicing by the group IA2 intron in a pre-mRNA derived from the *nrd*B gene of bacteriophage T4 3543–3549
A.-S.Sjögren, R.Strömberg and B.-M.Sjöberg

MOLECULAR BIOLOGY

Effects of base mismatches on joining of short oligodeoxynucleotides by DNA ligases 3403–3407
C.E.Pritchard and E.M.Southern

***Peg5/Neuronatin* is an imprinted gene located on sub-distal chromosome 2 in the mouse** 3428–3432
F.Kagitani, Y.Kuroiwa, S.Wakana, T.Shiroishi, N.Miyoshi, S.Kobayashi, M.Nishida, T.Kohda, T.Kaneko-Ishino and F.Ishino

***Sce3*, a suppressor of the *Schizosaccharomyces pombe* septation mutant *cdc11*, encodes a putative RNA-binding protein** 3433–3439
S.Schmidt, K.Hofmann and V.Simanis

Recruitment of transcription factors to the target site by triplex-forming oligonucleotides 3459–3464
F.Svinarchuk, I.Nagibneva, D.Chern, S.Ait-Si-Ali, L.L.Pritchard, P.Robin, C.Malvy and A.Harel-Bellan

Nucleoprotein complex formation by the enhancer binding protein NifA 3478–3485
X.Y.Wang, A.Kolb, W.Cannon and M.Buck

The binding of two dimers of IciA protein to the *dnaA* promoter 1P element enhances the binding of RNA polymerase to the *dnaA* promoter 1P 3486–3489
Y.Lee, H.Lee, J.Yim and D.Hwang

Contacts between *Bacillus subtilis* catabolite regulatory protein CcpA and *amyO* target site 3490–3496
J.-H.Kim and G.H.Chambliss

Effects of diaminopurine and inosine substitutions on A-tract induced DNA curvature. Importance of the 3′-A-tract junction 3497–3502
N.E.Møllegaard, C.Bailly, M.J.Waring and P.E.Nielsen

Competition between HMG-I(Y), HMG-1 and histone H1 on four-way junction DNA 3523–3531
D.A.Hill and R.Reeves

Diversity among the primate eosinophil-derived neurotoxin genes: a specific C-terminal sequence is necessary for enhanced ribonuclease activity 3532–3536
H.F.Rosenberg and K.D.Dyer

Re-usable DNA template for the polymerase chain reaction (PCR) 3537–3542
S.N.Sheikh and P.Lazarus

CHEMISTRY

Sequence-specific targeting and covalent modification of human genomic DNA 3440–3444
E.S.Belousov, I.A.Afonina, M.A.Podyminogin, H.B.Gamper, M.W.Reed, R.M.Wydro and R.B.Meyer

STRUCTURAL BIOLOGY

A prediction of the amino acids and structures involved in DNA recognition by type I DNA restriction and modification enzymes 3408–3414
S.S.Sturrock and D.T.F.Dryden

Contents (Continued)

Volume 25 number 17 September 1, 1997

COMPUTATIONAL BIOLOGY

Gapped BLAST and PSI-BLAST: a new generation of protein database search programs 3389–3402
S.F.Altschul, T.L.Madden, A.A.Schäffer, J.Zhang, Z.Zhang, W.Miller and D.J.Lipman

ESSA: an integrated and interactive computer tool for analysing RNA secondary structure 3514–3522
F.Chetouani, P.Monestié, P.Thébault, C.Gaspin and B.Michot

METHODS

Effects of base mismatches on joining of short oligodeoxynucleotides by DNA ligases 3403–3407
C.E.Pritchard and E.M.Southern

Precise assessment of microsatellite instability using high resolution fluorescent microsatellite analysis 3415–3420
S.Oda, E.Oki, Y.Maehara and K.Sugimachi

Re-usable DNA template for the polymerase chain reaction (PCR) 3537–3542
S.N.Sheikh and P.Lazarus

A solid-phase assay to screen monoclonal antibodies against DNA-binding protein 3550–3551
N.Ishii, H.Nakayama, J.Katayama, M.Arisawa and Y.Aoki

Rapid determination of the complexity of cDNA bands extracted from DDRT–PCR polyacrylamide gels 3552–3554
N.R.Smith, A.Li, M.Aldersley, A.S.High, A.F.Markham and P.A.Robinson

Microamplification of specific chromosome sequences; an improved method for genome analysis 3555–3557
N.Ponelies, N.Stein and G.Weber

Author index

Forthcoming events

This material may be protected by Copyright law (Title 17 U.S. Code)

Nucleic Acids Research, 1997, Vol. 25, No. 17 **3389–3402**

Gapped BLAST and PSI-BLAST: a new generation of protein database search programs

Stephen F. Altschul*, Thomas L. Madden, Alejandro A. Schäffer[1], Jinghui Zhang, Zheng Zhang[2], Webb Miller[2] and David J. Lipman

National Center for Biotechnology Information, National Library of Medicine, National Institutes of Health, Bethesda, MD 20894, USA, [1]Laboratory of Genetic Disease Research, National Human Genome Research Institute, National Institutes of Health, Bethesda, MD 20892, USA and [2]Department of Computer Science and Engineering, Pennsylvania State University, University Park, PA 16802, USA

Received June 20, 1997; Revised and Accepted July 16, 1997

ABSTRACT

The BLAST programs are widely used tools for searching protein and DNA databases for sequence similarities. For protein comparisons, a variety of definitional, algorithmic and statistical refinements described here permits the execution time of the BLAST programs to be decreased substantially while enhancing their sensitivity to weak similarities. A new criterion for triggering the extension of word hits, combined with a new heuristic for generating gapped alignments, yields a gapped BLAST program that runs at approximately three times the speed of the original. In addition, a method is introduced for automatically combining statistically significant alignments produced by BLAST into a position-specific score matrix, and searching the database using this matrix. The resulting Position-Specific Iterated BLAST (PSI-BLAST) program runs at approximately the same speed per iteration as gapped BLAST, but in many cases is much more sensitive to weak but biologically relevant sequence similarities. PSI-BLAST is used to uncover several new and interesting members of the BRCT superfamily.

INTRODUCTION

Variations of the BLAST algorithm (1) have been incorporated into several popular programs for searching protein and DNA databases for sequence similarities. BLAST programs have been written to compare protein or DNA queries with protein or DNA databases in any combination, with DNA sequences often undergoing conceptual translation before any comparison is performed. We will use the *blastp* program, which compares protein queries to protein databases, as a prototype for BLAST, although the ideas presented extend immediately to other versions that involve the translation of a DNA query or database. Some of the refinements described are applicable as well to DNA–DNA comparison, but have yet to be implemented.

BLAST is a heuristic that attempts to optimize a specific similarity measure. It permits a tradeoff between speed and sensitivity, with the setting of a 'threshold' parameter, *T*. A higher value of *T* yields greater speed, but also an increased probability of missing weak similarities. The BLAST program requires time proportional to the product of the lengths of the query sequence and the database searched. Since the rate of change in database sizes currently exceeds that of processor speeds, computers running BLAST are subjected to increasing load. However, the conjunction of several new algorithmic ideas allow a new version of BLAST to achieve improved sensitivity at substantially augmented speed. This paper describes three major refinements to BLAST.

(i) For increased speed, the criterion for extending word pairs has been modified. The original BLAST program seeks short word pairs whose aligned score is at least *T*. Each such 'hit' is then extended, to test whether it is contained within a high-scoring alignment. For the default *T* value, this extension step consumes most of the processing time. The new 'two-hit' method requires the existence of two non-overlapping word pairs on the same diagonal, and within a distance *A* of one another, before an extension is invoked. To achieve comparable sensitivity, the threshold parameter *T* must be lowered, yielding more hits than previously. However, because only a small fraction of these hits are extended, the average amount of computation required decreases.

(ii) The ability to generate gapped alignments has been added. The original BLAST program often finds several alignments involving a single database sequence which, when considered together, are statistically significant. Overlooking any one of these alignments can compromise the combined result. By introducing an algorithm for generating gapped alignments, it becomes necessary to find only one rather than all the ungapped alignments subsumed in a significant result. This allows the *T* parameter to be raised, increasing the speed of the initial database scan. The new gapped alignment algorithm uses dynamic programming to extend a central pair of aligned residues in both directions. For speed, earlier heuristic methods (2,3) confined the alignments produced to a predefined strip of the dynamic

* To whom correspondence should be addressed. Tel: +1 301 496 2475; Fax: +1 301 480 9241; Email: altschul@ncbi.nlm.nih.gov

programming path graph (4). Our approach considers only alignments that drop in score no more than X_g below the best score yet seen. The algorithm is able thereby to adapt the region of the path graph it explores to the data.

(iii) BLAST searches may be iterated, with a position-specific score matrix generated from significant alignments found in round i used for round $i + 1$. Motif or profile search methods frequently are much more sensitive than pairwise comparison methods at detecting distant relationships. However, creating a set of motifs or a profile that describes a protein family, and searching a database with them, typically has involved running several different programs, with substantial user intervention at various stages. The BLAST algorithm is easily generalized to use an arbitrary position-specific score matrix in place of a query sequence and associated substitution matrix. Accordingly, we have automated the procedure of generating such a matrix from the output produced by a BLAST search, and adapted the BLAST algorithm to take this matrix as input. The resulting Position-Specific Iterated BLAST, or PSI-BLAST, program may not be as sensitive as the best available motif search programs, but its speed and ease of operation can bring the power of these methods into more common use.

After describing these refinements to BLAST in greater detail, we consider several biological examples for which the sensitivity and speed of the program are greatly enhanced.

STATISTICAL PRELIMINARIES

To analyze the BLAST algorithm and its refinements, we need first to review the statistics of high-scoring local alignments. BLAST employs a substitution matrix, which specifies a score s_{ij} for aligning each pair of amino acids i and j. Given two sequences to compare, the original BLAST program seeks equal-length segments within each that, when aligned to one another without gaps, have maximal aggregate score. Not only the single best segment pair may be found, but also other *locally optimal* pairs (3,5–7), whose scores cannot be improved by extension or trimming. Such locally optimal alignments are called 'high-scoring segment pairs' or HSPs.

For the sake of the statistical theory, we assume a simple protein model in which the amino acids occur randomly at all positions with background probabilities P_i. We require that the expected score for two random amino acids $\sum_{i,j} P_i P_j s_{ij}$ be negative. Given the P_i and s_{ij}, the basic theory (8,9) yields two calculable parameters, λ and K, which can be used to convert nominal HSP scores to normalized scores, thereby rendering all scoring systems directly comparable from a statistical perspective. The normalized score S' for an HSP is given by the equation:

$$S' = \frac{\lambda S - \ln K}{\ln 2} \qquad \mathbf{1}$$

In this paper, a nominal score is given without units, while a score normalized by equation **1** is said to be expressed in *bits* (10,11). When two random protein sequences of sufficient lengths m and n are compared, the number E of distinct HSPs with normalized score at least S' expected to occur by chance is well approximated by:

$$E = N/2^{S'} \qquad \mathbf{2}$$

where $N = mn$ is the search space size (8–10). If a protein is compared to a whole database rather than a single sequence, n is the database length in residues. Equation **2** may be inverted to yield $S' = \log_2(N/E)$, the normalized score required to achieve a particular E-value. In a typical current database search, a protein of length 250 might be compared to a protein database of 50 000 000 total residues. To achieve a marginally significant E-value of 0.05, a normalized score of ~38 bits is necessary.

While the theory just outlined has not been proved for gapped local alignments and their associated scores, computational experiments strongly suggest that it remains valid (3,12–15). The statistical parameters λ and K, however, are no longer supplied by theory but must be estimated using comparisons of either simulated or real but unrelated sequences. To distinguish below whether a given set of parameters λ and K refer to gapped or ungapped alignments, we use the subscripts g and u respectively.

When gaps are not allowed, a further important theorem states that within HSPs the aligned pair of letters (i,j) tends to occur with the 'target frequency':

$$q_{ij} = P_i P_j e^{\lambda_u s_{ij}} \qquad \mathbf{3}$$

The q_{ij} of equation **3** sum to 1; indeed, λ_u is calculated to be the unique positive number for which this is the case (8,9). The scores s_{ij} are optimal for detecting alignments with these particular target frequencies (8,10), and by inverting equation **3** to $s_{ij} = [\ln(q_{ij}/P_i P_j)]/\lambda_u$, scores may be chosen, with arbitrary scale, that correspond to any desired set of q_{ij}. The popular PAM (16,17) and BLOSUM (18) substitution matrices are constructed with explicit use of this log-odds formula. No corresponding result has been established for gapped alignment scoring systems. However, if the gap costs used are sufficiently large, it is expected that the target frequencies observed in high-scoring local alignments of random sequences will not differ greatly from those for the no-gap case.

REFINEMENT OF THE BASIC ALGORITHM: THE TWO-HIT METHOD

The central idea of the BLAST algorithm is that a statistically significant alignment is likely to contain a high-scoring pair of aligned words. BLAST first scans the database for *words* (typically of length three for proteins) that score at least T when aligned with some word within the query sequence. Any aligned word pair satisfying this condition is called a *hit*. The second step of the algorithm checks whether each hit lies within an alignment with score sufficient to be reported. This is done by extending a hit in both directions, until the running alignment's score has dropped more than X below the maximum score yet attained. This extension step is computationally quite costly; with the T and X parameters necessary to attain reasonable sensitivity to weak alignments, the extension step typically accounts for >90% of BLAST's execution time. It is therefore desirable to reduce the number of extensions performed.

Our refined algorithm is based upon the observation that an HSP of interest is much longer than a single word pair, and may therefore entail multiple hits on the same diagonal and within a relatively short distance of one another. (The *diagonal* of a hit involving words starting at positions (x_1, x_2) of the database and query sequences may be defined as $x_1 - x_2$. The *distance* between two hits on the same diagonal is the difference between their first coordinates.) This signature may be used to locate HSPs more

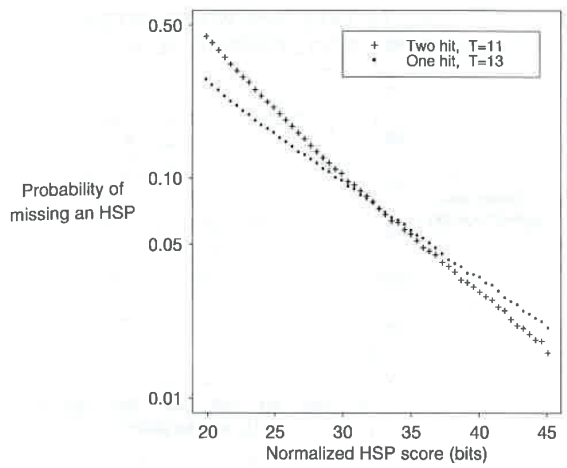


Figure 1. The sensitivity of the two-hit and one-hit heuristics as a function of HSP score. Using the BLOSUM-62 amino acid substitution matrix (18), and the target frequencies q_{ij} implied by equation **3** and the background amino acid frequencies P_i of Robinson and Robinson (20), 100 000 model HSPs were generated for each of the nominal scores 37–92, corresponding to normalized scores 19.9–45.1 bits. It was determined by inspection whether each HSP failed to contain two non-overlapping length-3 word pairs with nominal score at least 11, and within a distance 40 of one another, or a single length-3 word pair with nominal score at least 13. The corresponding probabilities of missing an HSP using the two-hit heuristic with $T = 11$, and the one-hit heuristic with $T = 13$, are plotted as a function of normalized HSP score. The two-hit method is more sensitive for HSPs with score at least 33 bits.

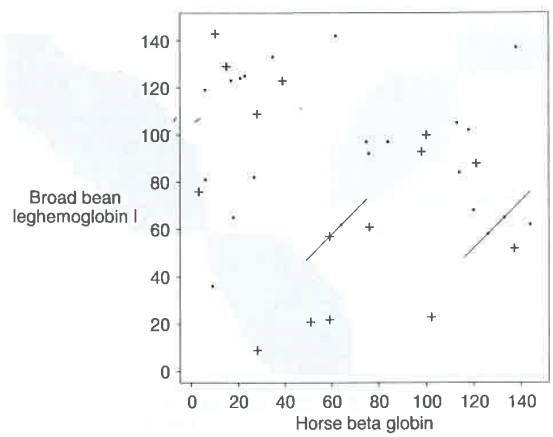


Figure 2. The BLAST comparison of broad bean leghemoglobin I (87) (SWISS-PROT accession no. P02232) and horse β-globin (88) (SWISS-PROT accession no. P02062). The 15 hits with score at least 13 are indicated by plus signs. An additional 22 non-overlapping hits with score at least 11 are indicated by dots. Of these 37 hits, only the two indicated pairs are on the same diagonal and within distance 40 of one another. Thus the two-hit heuristic with $T = 11$ triggers two extensions, in place of the 15 extensions invoked by the one-hit heuristic with $T = 13$. Because this is just one example, the relative numbers of hits and extensions at the various settings of T correspond only roughly to the ratios found in a full database search. An ungapped extension of the leftward of the two hit pairs yields an HSP with nominal score 45, or 23.6 bits, calculated using λ_u and K_u.

efficiently. Specifically, we choose a window length A, and invoke an extension only when two non-overlapping hits are found within distance A of one another on the same diagonal. Any hit that overlaps the most recent one is ignored. Efficient execution requires an array to record, for each diagonal, the first coordinate of the most recent hit found. Since database sequences are scanned sequentially, this coordinate always increases for successive hits. The idea of seeking multiple hits on the same diagonal was first used in the context of biological database searches by Wilbur and Lipman (19).

Because we require two hits rather than one to invoke an extension, the threshold parameter T must be lowered to retain comparable sensitivity. The effect is that many more single hits are found, but only a small fraction have an associated second hit on the same diagonal that triggers an extension. The great majority of hits may be dismissed after the minor calculation of looking up, for the appropriate diagonal, the coordinate of the most recent hit, checking whether it is within distance A of the current hit's coordinate, and finally replacing the old with the new coordinate. Empirically, the computation saved by requiring fewer extensions more than offsets the extra computation required to process the larger number of hits.

To study the relative abilities of the one-hit and two-hit methods to detect HSPs of varying score, we model proteins using the background amino acid frequencies of Robinson and Robinson (20), and use the BLOSUM-62 substitution matrix (18) for sequence comparison. Given these P_i and s_{ij}, the statistical parameters for ungapped local alignments are calculated to be $\lambda_u = 0.3176$ and $K_u = 0.134$. Using equation **3** above, we may calculate the q_{ij} for which the scoring system is optimized, and employ these target frequencies to generate model HSPs. Finally, we evaluate the sensitivity of the one-hit and two-hit BLAST heuristics using these HSPs.

The one-hit method will detect an HSP if it somewhere contains a length-W word of score at least T. For $W = 3$ and $T = 13$, Figure 1 shows the empirically estimated probability that an HSP is missed by this method, as a function of its normalized score. The two-hit method will detect an HSP if it contains two non-overlapping length-W words of score at least T, with starting positions that differ by no more than A residues. For $W = 3$, $T = 11$ and $A = 40$, Figure 1 shows the estimated probability that an HSP is missed by this method, as a function of its normalized score. For HSPs with score at least 33 bits, the two-hit heuristic is more sensitive.

To analyze the relative speeds of the one-hit and two-hit methods, using the parameters studied above, we note that the two-hit method generates on average ~3.2 times as many hits, but only ~0.14 times as many hit extensions (Fig. 2). Because it takes approximately one ninth as long to decide whether a hit need be extended as actually to extend it, the hit-processing component of the two-hit method is approximately twice as rapid as the same component of the one-hit method.

TRIGGERING THE GENERATION OF GAPPED ALIGNMENTS

Figure 1 shows that even when using the original one-hit method with threshold parameter $T = 13$, there is generally no greater than a 4% chance of missing an HSP with score >38 bits. While this would appear sufficient for most purposes, the one-hit default T parameter has typically been set as low as 11, yielding an execution time nearly three times that for $T = 13$. Why pay this price for what appears at best marginal gains in sensitivity? The

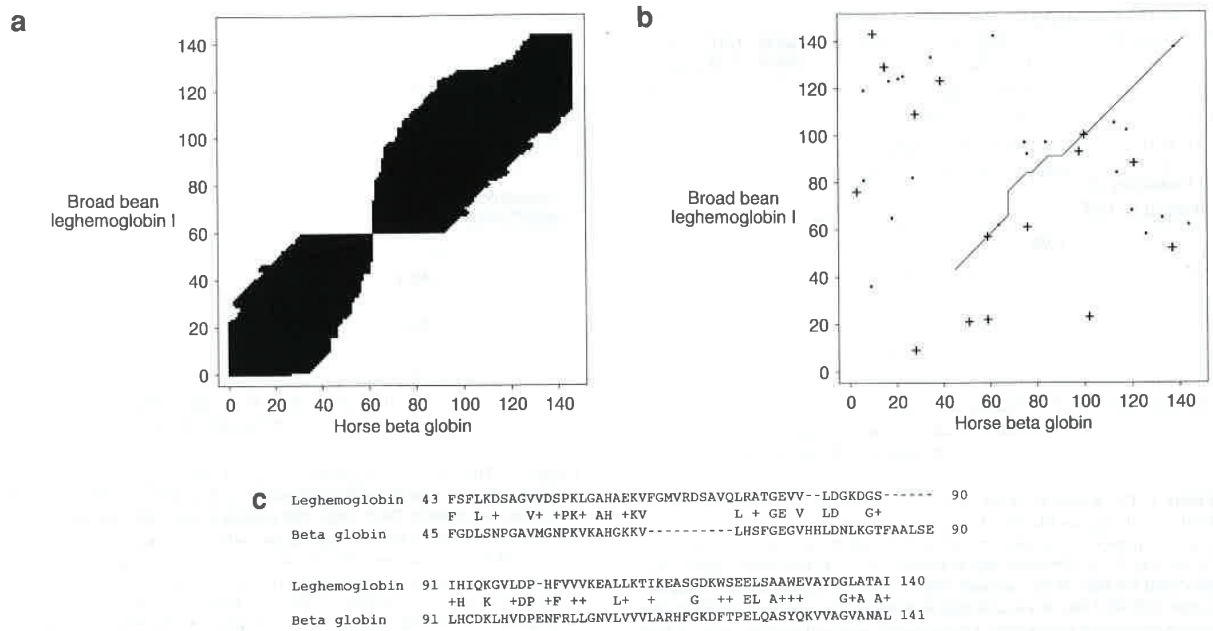


Figure 3. A gapped extension generated by BLAST for the comparison of broad bean leghemoglobin I (87) and horse β-globin (88). (**a**) The region of the path graph explored when seeded by the alignment of alanine residues at respective positions 60 and 62. This seed derives from the HSP generated by the leftward of the two ungapped extensions illustrated in Figure 2. The X_g dropoff parameter is the nominal score 40, used in conjunction with BLOSUM-62 substitution scores and a cost of $10 + k$ for gaps of length k. (**b**) The path corresponding to the optimal local alignment generated, superimposed on the hits described in Figure 2. The original BLAST program, using the one-hit heuristic with $T = 11$, is able to locate three of the five HSPs included in this alignment, but only the first and last achieve a score sufficient to be reported. (**c**) The optimal local alignment, with nominal score 75 and normalized score 32.4 bits. In the context of a search of SWISS-PROT (26), release 34 (21 219 450 residues), using the leghemoglobin sequence (143 residues) as query, the E-value is 0.54 if no edge-effect correction (22) is invoked. The original BLAST program locates the first and last ungapped segments of this alignment. Using sum-statistics with no edge-effect correction, this combined result has an E-value of 31 (21,22). On the central lines of the alignment, identities are echoed and substitutions to which the BLOSUM-62 matrix (18) gives a positive score are indicated by a '+' symbol.

reason is that the original BLAST program treats gapped alignments implicitly by locating, in many cases, several distinct HSPs involving the same database sequence, and calculating a statistical assessment of the combined result (21,22). This means that two or more HSPs with scores well below 38 bits can, in combination, rise to statistical significance. If any one of these HSPs is missed, so may be the combined result.

The approach taken here allows BLAST to simultaneously produce gapped alignments and run significantly faster than previously. The central idea is to trigger a gapped extension for any HSP that exceeds a moderate score S_g, chosen so that no more than about one extension is invoked per 50 database sequences. (By equation **2**, for a typical-length protein query, S_g should be set at ~22 bits.) A gapped extension takes much longer to execute than an ungapped extension, but by performing very few of them the fraction of the total running time they consume can be kept relatively low.

By seeking a single gapped alignment, rather than a collection of ungapped ones, only one of the constituent HSPs need be located for the combined result to be generated successfully. This means that we may tolerate a much higher chance of missing any single moderately scoring HSP. For example, consider a result involving two HSPs, each with the same probability P of being missed at the hit-stage of the BLAST algorithm, and suppose that we desire to find the combined result with probability at least 0.95. The original algorithm, needing to find both HSPs, requires $2P - P^2 \leq 0.05$, or P less than ~0.025. In contrast, the new algorithm requires only that $P^2 \leq 0.05$, and thus can tolerate P as high as 0.22. This permits the T parameter for the hit-stage of the algorithm to be raised substantially while retaining comparable sensitivity—from $T = 11$ to $T = 13$ for the one-hit heuristic. (The two-hit heuristic described above lowers T back to 11.) As will be discussed below, the resulting increase in speed more than compensates for the extra time required for the rare gapped extension.

In summary, the new gapped BLAST algorithm requires two non-overlapping hits of score at least T, within a distance A of one another, to invoke an ungapped extension of the second hit. If the HSP generated has normalized score at least S_g bits, then a gapped extension is triggered. The resulting gapped alignment is reported only if it has an E-value low enough to be of interest. For example, in the pairwise comparison of Figure 2, the ungapped extension invoked by the hit pair on the left produces an HSP with score 23.6 bits (calculated using λ_u and K_u). This is sufficient to trigger a gapped extension, which generates an alignment with score 32.4 bits (calculated using λ_g and K_g) and E-value of 0.5 (Fig. 3). The original BLAST program locates only the first and last ungapped segments of this alignment (Fig. 3c), and assigns them a combined E-value >50 times greater.

THE CONSTRUCTION AND STATISTICAL EVALUATION OF GAPPED LOCAL ALIGNMENTS

The standard dynamic programming algorithms for pairwise sequence alignment perform a fixed amount of computation per cell of a path graph, whose dimensions are the lengths of the two sequences being compared (23–25). In order to gain speed, database search algorithms such as Fasta (2) and an earlier gapped version of BLAST (3) sacrifice rigor by confining the dynamic programming to a banded section of the full path graph (4), chosen to include a region of already identified similarity. One problem with this approach is that the optimal gapped alignment may stray beyond the confines of the band explored. As the width of the band is increased to reduce this possibility, the speed advantage of the algorithm is vitiated.

We have accordingly taken a different heuristic approach to constructing gapped local alignments, which is a simple generalization of BLAST's method for constructing HSPs. The central idea is to consider only cells for which the optimal local alignment score falls no more than X_g below the best alignment score yet found. Starting from a single aligned pair of residues, called the *seed*, the dynamic programming proceeds both forward and backward through the path graph (Zheng Zhang *et al.*, manuscript in preparation) (Figs 3a and 4). The advantage of this approach is that the region of the path graph explored adapts to the alignment being constructed. The alignment can wander arbitrarily many diagonals away from the seed, but the number of cells expanded on each row tends to remain limited, and may even shrink to zero before a boundary of the path graph is encountered (Fig. 4). The X_g parameter serves a similar function to the band-width parameter of the earlier heuristic, but the region of the path graph it implicitly specifies be explored is in general more productively chosen.

An important element for this heuristic is the intelligent choice of a seed. Given an HSP whose score is sufficiently high that it triggers a gapped extension, how does one choose a residue pair to force into alignment? While more sophisticated approaches are possible, the simple procedure we have implemented is to locate, along the HSP, the length-11 segment with highest alignment score, and use its central residue pair as the seed. If the HSP itself is shorter than 11, a central residue pair is chosen. For example, the first ungapped region in the alignment of Figure 3c constitutes the HSP that triggered the alignment. The highest-scoring length-11 segment of this HSP aligns leghemoglobin residues 55–65 with β-globin residues 57–67. Thus the alanine residues at respective positions 60 and 62 are used as the seed for the gapped extension illustrated in Figure 3a. As discussed in the performance evaluation section below, this procedure is extremely good at selecting seeds that in fact participate in an optimal alignment.

Most gapped extensions are triggered by chance similarities, and are therefore likely to be of limited extent, as illustrated in Figure 4. The reverse extension in this example explores ~2000 path graph cells, so that a typical two-way gapped extension that does not encounter the end of either sequence is expected to involve ~4000 cells. Because S_g is set so that a gapped extension is invoked less than once per 50 database sequences, fewer than 80 cells need be explored per database sequence.

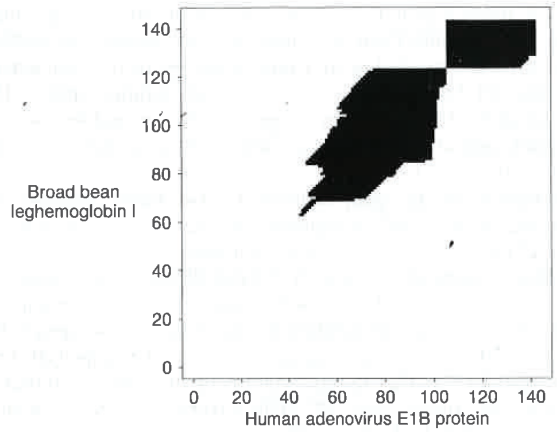


Figure 4. The path graph region explored by BLAST during a gapped extension for the comparison of broad bean leghemoglobin I and the E1B protein small T-antigen from human adenovirus type 4 (89) (SWISS-PROT accession no. P10406). The X_g dropoff parameter is the nominal score 40, used in conjunction with BLOSUM-62 substitution scores and $10 + k$ gap costs. The 22.7 bit HSP that triggers this extension, involving leghemoglobin residues 119–140 and adenovirus residues 101–122, is merely a random similarity, and not part of a larger and higher-scoring alignment. The gapped extension is seeded by the alignment of residues 124 and 106. The optimal alignment score through points in the path graph drops steadily as one moves beyond the triggering HSP, and the reverse extension terminates before the beginning of either protein is reached. A total of 2766 path graph cells are explored, with the reverse extension accounting for 2047 of these cells.

The execution time required for a gapped extension is ~500 times that for an ungapped extension. However, by triggering gapped extensions in the manner described, while simultaneously raising T for the single-hit version of BLAST from 11 to 13, approximately one gapped extension is invoked for every 4000 ungapped extensions avoided. Because the number of ungapped extensions is reduced by about two thirds, the total time spent on the extension stage of BLAST is cut by well over half. Of course, the two-hit strategy described above reduces the time needed for the ungapped extensions still further. Once program overhead is accounted for, the net speedup is a factor of about three.

For any alignment actually reported, a gapped extension that records 'traceback' information (25) needs to be executed. To increase BLAST's accuracy in producing optimal local alignments, these gapped extensions use by default a substantially larger X_g parameter than employed during the program's search stage.

Table 1. Relative times spent by the original and gapped BLAST programs on various algorithmic stages

	Overhead: database scanning, output, etc.	Calculating whether hits qualify for ungapped extension	Ungapped extensions	Gapped extensions
Original BLAST	8 (8%)		92 (92%)	
Gapped BLAST	8 (24%)	12 (37%)	5 (15%)	8 (24%)

The times required by various steps of the BLAST algorithm vary substantially from one query and one database to another. Table 1 shows typical relative times spent by the original and the gapped BLAST programs on various algorithmic stages. The 'original BLAST' program is represented, here and below, by a variant form of *blastp* version 1.4.9, modified so that it uses the same edge-effect correction (22) and background amino acid frequencies as the 'gapped BLAST'. The times represent the average for three different queries, with the time for the original BLAST program normalized in each instance to 100 units.

More concretely, to search SWISS-PROT (26), release 34 (59 576 sequences; 21 219 450 residues), with the length-567 influenza A virus hemagglutinin precursor (27) as query, the original BLAST program requires 45.8 s, and the gapped BLAST program 15.8 s. This timing experiment, and others referred to below, was run on one 200 MHz R10000 cpu processor of a lightly loaded SGI Power Challenge XL computer with 2.5 Gbytes of RAM. This machine runs the operating system IRIX, version 6.2, which is an implementation of UNIX. We used the standard SGI C compiler, with the *-O* flag for optimization, to compile all versions of the programs. The times reported are the user times given by the *time* command, and are for the better of two identical runs.

A closely related type of gapped extension routine to that used here was developed by G. Myers during the evaluation of the original BLAST algorithm. It was not included in the publicly distributed code primarily because the then current strategy of extending every hit decreased the algorithm's speed unduly for the relatively small gain in sensitivity realized (1).

As discussed above, the statistical significance of gapped alignments may be evaluated using the two statistical parameters λ_g and K_g. The current version of the Fasta program (2) estimates these parameters on each run, by analyzing the distribution of alignment scores produced by all the sequences in the database. BLAST gains speed by producing alignments for only the few database sequences likely to be related to the query, and therefore does not have the option of estimating λ_g and K_g on the fly. Instead, it uses estimates of these parameters produced beforehand by random simulation (3). A drawback of this approach is that the program may not accept an arbitrary scoring system, for which no simulation has been performed, and still produce accurate estimates of statistical significance. The original BLAST programs, in contrast, because they dealt only with ungapped local alignments, could derive λ_u and K_u from theory for any scoring matrix (8,9).

ITERATED APPLICATION OF BLAST TO POSITION-SPECIFIC SCORE MATRICES

Database searches using position-specific score matrices, also called profiles or motifs, often are much better able to detect weak relationships than are database searches that use a simple sequence as query (28–38). Employing these methods, however, frequently has involved the use of several different programs and a fair degree of expertise. Accordingly, to render the power of motif searches more readily available, we have written a procedure to construct a position-specific score matrix automatically from the output of a BLAST run, and modified BLAST to operate using such a matrix in the place of a simple query. The resulting PSI-BLAST program often is substantially more sensitive than the corresponding BLAST program, but for each iteration takes little more than the same time to run. In related work, Henikoff and Henikoff (39) have described how, short of modifying BLAST so that it may operate on a position-specific score matrix, a single artificial sequence that approximates such a matrix may be used as a query with the original BLAST programs.

The construction of a position-specific score matrix is a multi-stage process, and at each stage a choice must be made among a number of alternative routes. We have been guided by the goals of automatic operation, speed of execution, and general simplicity. The issues discussed below are: (i) general architecture of the score matrix; (ii) construction of the multiple alignment from which the matrix is derived; (iii) weights for sequences within the multiple alignment, and evaluation of the effective number of independent observations it constitutes; (iv) estimation of target frequencies, and the construction of matrix scores; (v) applying BLAST to a position-specific matrix, and the statistical evaluation of search results. We do not claim our current implementation is optimal, and it is likely that over time some of its details will change.

Score matrix architecture

The alignment of a simple sequence with a pattern embodied by a position-specific score matrix is almost completely analogous to the alignment of two simple sequences. The only real difference is that the score for aligning a letter with a pattern position is given by the matrix itself, rather than with reference to a substitution matrix. For proteins, a query of length L and a substitution matrix of dimension 20×20 are replaced by a position-specific matrix of dimension $L \times 20$. Position-specific gap costs may be defined as well (34,40). As with pairwise sequence comparison, one may choose among finding the best global alignment of the matrix and the simple sequence (23), finding the best alignment of the complete matrix with a segment of the sequence (41), and finding the best local alignment of the matrix and sequence (24).

Position-specific protein score matrices draw their power from two sources. The first is improved estimation of the probabilities with which amino acids occur at various pattern positions, leading to a more sensitive scoring system. The second is relatively precise definition of the boundaries of important motifs. By demanding the complete alignment of one or more motifs, rather than seeking an arbitrary local alignment, the size of the search space may be greatly reduced, thereby lowering the level of random noise. Unfortunately, there are many obstacles to automating well the delineation of a set of motifs from the output of a database search. The query sequence may contain a variety of different domains, and share different subsets of them with different proteins in the database. Furthermore, defining the proper extent of even a single motif may be challenging (42).

Accordingly, we have chosen to forgo the potential advantages of restricting the length of our derived matrices, and then demanding that they be completely aligned with segments of database sequences (41). Instead, each matrix we construct has length precisely equal to that of the original query sequence. When searching the database with such a matrix, we seek local alignments, in full analogy to those sought by BLAST when used for straightforward sequence–sequence comparison. Finally, we do not attempt to derive position-specific gap scores for use with our position-specific substitution scores. Instead, in each iteration

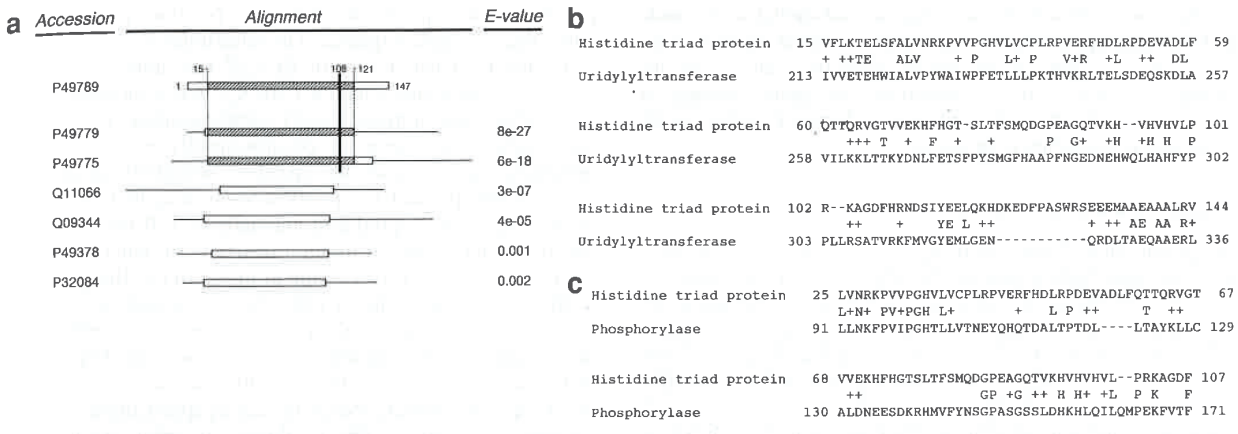


Figure 5. (**a**) The multiple alignment generated by PSI-BLAST when the human fragile histidine triad (HIT) protein (61) (SWISS-PROT accession no. P49789) is compared to SWISS-PROT. All pairwise local alignments have *E*-value ≤0.01, and are identified in SWISS-PROT as belonging to the HIT family. Thick bars within the six database sequences represent segments that align with various segments from the query. In constructing sequence weights for the indicated multiple alignment column, corresponding to residue 108 of the query, only the shaded portions of the multiple alignment are used. (**b**) A local alignment of the human HIT protein and *H.influenzae* galactose-1-phosphate uridylyltransferase (63) (SWISS-PROT accession no. P31764). In its first position-specific iteration, PSI-BLAST gives this alignment a score of 45.4 bits, corresponding to an *E*-value of 4×10^{-5}. '+' symbols reflect positive BLOSUM-62 matrix scores, even though a position-specific matrix is used to construct the alignment. (**c**) A local alignment of the human HIT protein and yeast 5′,5‴-P1,P4-tetraphosphate phosphorylase I (64) (SWISS-PROT accession no. P16550). In its second position-specific iteration, PSI-BLAST gives this alignment a score of 43.4 bits, corresponding to an *E*-value of 2×10^{-4}.

of PSI-BLAST, we employ the same gap scores that are used in the first, simple BLAST run. Our reasons are that there is no good theory for deriving gap costs from a multiple alignment and that, as will be discussed below, by eschewing position-specific gap costs we can make a reasonable estimate of the statistical significance of the resulting local alignments.

Multiple alignment construction

To produce a multiple alignment from the BLAST output, we simply collect all database sequence segments that have been aligned to the query with *E*-value below a threshold, by default set to 0.01. The query is used as a master, or template, for constructing a multiple alignment M. Any row (i.e., database sequence segment) identical to the query segment with which it aligns is purged, and only one copy is retained of any rows that are >98% identical to one another. Pairwise alignment columns that involve gap characters inserted into the query are simply ignored, so that M has exactly the same length as the query. Because we are dealing with local alignments, the columns of M may involve varying numbers of sequences, and many columns may include nothing but the query. We make no attempt to improve M by comparing database sequences with one another, or by any other true multiple alignment procedure.

As will be discussed, the matrix scores constructed for a given alignment column should depend not only upon the residues appearing there, but upon those in other columns as well. To make this dependency easy to formulate, however, we need to prune our raw multiple alignment M to a simpler 'reduced' one. This pruning is done independently for each column, so the reduced multiple alignment M_C will in general vary from one column C to the next. To construct M_C, we first specify the set R of sequences it includes to be exactly those that contribute a residue to column C. We then define the columns of M_C to be just those columns of M in which all the sequences of R are represented. By construction, the reduced multiple alignment M_C has residues or gap characters in every row and column (Fig. 5a), and is therefore amenable to the various manipulations described below.

Sequence weights

When constructing a score matrix from a multiple alignment, it is a mistake to give all sequences of the alignment equal weight. A large set of closely related sequences carries little more information than a single member, but its size alone may allow it easily to 'outvote' a small number of more divergent sequences. One way past this difficulty is to assign weights to the various sequences, with those having many close relatives receiving smaller weight. The many sequence weighting methods that have been proposed (43–51) often produce roughly equivalent results. Because of its speed and simplicity, we have implemented a modified version of the sequence weighting method of Henikoff and Henikoff (47). Gap characters are treated as a 21st distinct character, and any columns consisting of identical residues are ignored in calculating weights. In speaking of a column's observed residue frequencies f_i, we shall henceforth mean its weighted rather its raw frequencies.

In constructing matrix scores, not only a column's observed residue frequencies are important, but also the effective number of independent observations it constitutes: a column consisting of a single valine and a single isoleucine carries different information than one consisting of five independently occurring instances of each. Accordingly, we need to estimate the relative number N_C of independent observations constituted by the alignment M_C. A simple count of the number of sequences in M_C is a poor measure, for 10 identical sequences imply fewer independent observations than do 10 divergent ones. We thus propose as a simple first estimate for N_C the mean number of different residue types, including gap characters, observed in the various columns of M_C. This estimate is clearly not ideal, as it saturates at 21 no matter how many independent sequences are contained in M_C. However, for the data we are likely to encounter, N_C is typically much

smaller than 21, and therefore perhaps a good enough approximation for our purposes. As will be seen, it is not the absolute value of N_C that is important, but rather its relative value from one column to another. N_C is essentially the same measure of alignment variability as that proposed by Henikoff and Henikoff (52) for use in a different manner.

Target frequency estimation

Given a multiple alignment, many methods for generating score matrices have been advanced (28–37,42,52–54). The prescription with perhaps the best theoretical foundation is that the scores for a specific pattern position be of the form log (Q_i/P_i), where Q_i is the estimated probability for residue i to be found in that column (29,30,32,33,36,37,42,52–54). This leaves open the question how best to estimate the Q_i.

Given a multiple alignment involving a large number of independent sequences, the estimate of Q_i for a specific column should converge simply to the observed frequency of residue i in that column. However, in addition to the sequence weighting issues discussed above, factors that complicate estimating the Q_i include small sample size (30), and prior knowledge of relationships among the residues (16,37,53). Various studies suggest that the best currently available method for estimating the Q_i is that of Dirichlet mixtures (52–56). However, because it often performs nearly as well (52), and due to its relative simplicity, we have implemented the data-dependent pseudocount method introduced by Tatusov *et al.* (37). This method uses the prior knowledge of amino acid relationships embodied in the substitution matrix s_{ij} to generate residue pseudocount frequencies g_i, which are averaged with the observed frequencies f_i to estimate the Q_i.

Specifically, for a given column C, we construct pseudocount frequencies g_i using the formula:

$$g_i = \sum_j \frac{f_j}{P_j} q_{ij} \qquad \mathbf{4}$$

where the q_{ij} are the target frequencies implicit in the substitution matrix, and given by equation **3**. Intuitively, those residues favored by the substitution matrix to align with the residues actually observed receive high pseudocount frequencies. We then estimate Q_i by:

$$Q_i = \frac{\alpha f_i + \beta g_i}{\alpha + \beta} \qquad \mathbf{5}$$

where α and β are the relative weights given to observed and pseudocount residue frequencies. So that the scores we construct will reduce to s_{ij} in columns where nothing has been aligned to the query sequence, we let $\alpha = N_C - 1$. β remains an arbitrary pseudocount parameter; the larger its value, the greater the emphasis given to prior knowledge of residue relationships vis a vis observed residue frequencies. We have found empirically that, in conjunction with our method for calculating α, a reasonably good setting for β is 10.

BLAST applied to position-specific score matrices

The initial step of the BLAST algorithm is the construction of a list of words that align to query words with score at least T. Only minor modifications to the code are necessary for this step to be performed on a query consisting of a position-specific matrix rather than a simple sequence. The same holds for the ungapped and gapped extension steps of BLAST. One important issue is whether key parameters such as T and X_g, used at various heuristic stages of the algorithm and tuned to simple sequence comparison, can be applied unchanged to position-specific matrices without degrading unduly either the speed or sensitivity of database searches. We approach this problem by ensuring that the scale λ_u of the matrix scores produced internally by PSI-BLAST corresponds to that of the substitution matrix s_{ij}. In other words, we calculate the scores for a column of the matrix as $[\ln(Q_i/P_i)]/\lambda_u$.

There is no analytic theory with which to estimate the statistical significance of a gapped alignment of a position-specific score matrix and a simple sequence. However, one may hypothesize that for a score matrix constructed to the same scale as s_{ij}, a given set of gap costs should produce the same gapped alignment scale parameter λ_g as for s_{ij}. This would be convenient, because then PSI-BLAST could estimate statistical significance without expending after each iteration the substantial time required to estimate λ_g and K_g by random simulation. To test this hypothesis, we performed a number of statistical tests on PSI-BLAST generated score matrices, scaled to have $\lambda_u = 0.3176$, the value applicable to previously published BLOSUM-62 simulations (3).

First, we searched SWISS-PROT using as query the length-567 influenza A virus hemagglutinin precursor (27), and captured the score matrix constructed by PSI-BLAST from the 128 local alignments with E-value ≤ 0.01. We then compared this matrix to 10 000 random sequences of length 567, generated using the background amino acid frequencies of Robinson and Robinson (20). A gap of length k was charged a cost of $10 + k$. Counts of the optimal local alignment scores, calculated using an appropriately modified version of the Smith–Waterman algorithm (24), are plotted in Figure 6. Also shown is the best fitting extreme value distribution (3,15) which, using the edge-effect correction described by Altschul and Gish (3), has statistical parameters $\lambda_g = 0.251$ and $K_g = 0.031$. It is apparent that the distribution fits the random trial reasonably well; a χ^2 goodness-of-fit test with 34 degrees of freedom has value 41.8, which is lower than one would expect 20% of the time even were the theory precisely valid. This supports the idea that the statistical theory described above applies to local alignments of position-specific score matrices and simple sequences. Furthermore, the estimate $\lambda_g \approx 0.251 \pm 0.003$ agrees to within experimental error with the value 0.255 previously published for these gap costs (3). Similar agreement was obtained with a number of other protein sequences as initial query (results not shown), and in all cases the much less important K_g parameter could be estimated accurately as well. In general, values of λ_g for the comparison of position-specific score matrices with simple sequences appear to differ by <2% from the values for simple pairwise sequence comparison. Using these precomputed values for λ_g should thus entail an error of less than one bit for PSI-BLAST scores <50 bits, corresponding to a factor of less than two in the estimation of statistical significance.

PERFORMANCE EVALUATION

To test more directly the statistics used by PSI-BLAST, we compared query sequences from 11 large and well characterized protein families to the SWISS-PROT database, and then ran the position-specific score matrices generated against a shuffled version of the same database. For each query, we recorded the

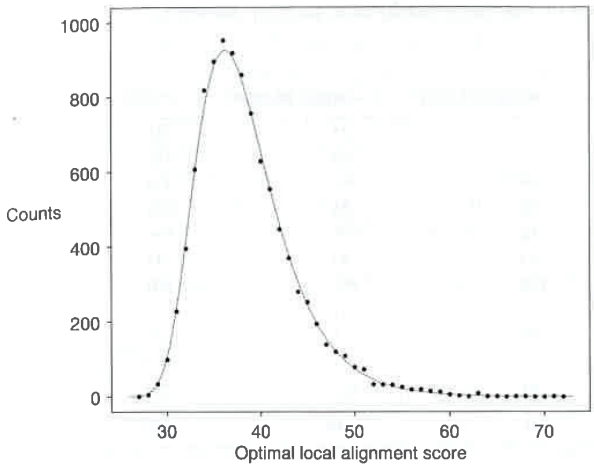


Figure 6. The distribution of optimal local alignment scores from the comparison of a position-specific score matrix with 10 000 random protein sequences. The score matrix was constructed by PSI-BLAST from the 128 local alignments with *E*-value ≤0.01 found in a search of SWISS-PROT using as query the length-567 influenza A virus hemagglutinin precursor (27) (SWISS-PROT accession no. P03435). The random sequences, each of length 567, were generated using the amino acid frequencies of Robinson and Robinson (20). Optimal local alignment scores were calculated using the position-specific matrix in conjunction with $10 + k$ gap costs. The extreme value distribution that best fits the data (3,15) is plotted. A χ^2 goodness-of-fit test with 34 degrees of freedom has value 41.8, corresponding to a *P*-value of 0.20.

lowest *E*-value found, as well as the number of shuffled sequences yielding *E*-values ≤1 and 10. For comparison, we performed the identical shuffled-database test on the gapped and original versions of BLAST. To reduce the probability that high-scoring alignments were missed due to the heuristic nature of the algorithms, we performed these tests with $T = 9$ rather than the default value of 11. The results are given in Table 2. For the 11 queries, the median of the low PSI-BLAST *E*-values was 0.87, which corresponds to a median *P*-value of 0.58 (8,9). The mean numbers of shuffled database sequences with *E*-values <1 and 10 were 1.0 and 8.7, respectively, within 20% of the expected values of 1.0 and 10.0. The equivalent tests for the ungapped and gapped versions of BLAST also yielded results that diverged from theory by <50%.

The ability to estimate with reasonable accuracy the significance of gapped local matrix-sequence alignments permits us to automate the construction of position-specific score matrices during multiple iterations of the PSI-BLAST program. After each iteration, we generate a new multiple alignment simply by collecting those alignments with *E*-value lower than a defined threshold. An interactive version of PSI-BLAST allows the user to override either the inclusion or exclusion of specific local alignments. Once a given database sequence has been used in the generation of a position-specific score matrix, low *E*-values for this sequence are virtually guaranteed in future iterations, for the sequence is to a certain extent being compared with itself. The biological relevance of PSI-BLAST output thus depends critically on avoiding the inappropriate inclusion of sequences in the multiple alignment constructed. Specifically, the utility of the score matrix produced is immediately vitiated by the inclusion of any alignment involving a region of highly biased amino acid composition (57,58).

Table 2. The comparison of various query sequences with a shuffled version of SWISS-PROT

Protein family	SWISS-PROT accession no. of query	Original BLAST Low *E*-value	No. of seqs with *E*-value ≤1	≤10	Gapped BLAST Low *E*-value	No. of seqs with *E*-value ≤1	≤10	PSI-BLAST Low *E*-value	No. of seqs with *E*-value ≤1	≤10
Serine protease	P00762	0.86	1	7	3.0	0	4	0.94	1	8
Serine protease inhibitor	P01008	3.9	0	4	0.078	1	9	1.5	0	9
Ras	P01111	3.4	0	8	3.4	0	7	1.1	0	9
Globin	P02232	2.4	0	7	2.8	0	5	8.2	0	2
Hemagglutinin	P03435	0.11	2	11	0.46	3	16	0.87	1	8
Interferon α	P05013	2.4	0	6	0.27	2	4	0.11	2	11
Alcohol dehydrogenase	P07327	1.5	0	2	0.80	1	5	1.5	0	9
Histocompatibility antigen	P10318	0.91	1	7	0.13	1	7	0.0031	2	6
Cytochrome P450	P10635	0.84	2	5	8.5	0	3	0.46	1	15
Glutathione transferase	P14942	1.0	1	10	3.3	0	3	0.30	2	9
H^+-transporting ATP synthase	P20705	0.012	1	8	0.26	2	14	0.79	2	10
Average (median or mean)		1.0	0.7	6.8	0.80	0.9	7.0	0.87	1.0	8.7

The original and gapped BLAST comparisons use BLOSUM-62 substitution scores (18). All three programs use threshold *T* parameter set to 9, but the gapped BLAST and PSI-BLAST programs use the two-hit method to trigger ungapped extensions. The original BLAST program has the *X* dropoff parameter set to nominal score 23. The gapped BLAST and PSI-BLAST comparisons charge gaps of length *k* a cost of $10 + k$. They have X_u set to 16, and X_g set to 40 for the database search stage and to 67 for the output stage of the algorithms. Gapped alignments are triggered by a score corresponding to ~22 bits. For PSI-BLAST, the query is first compared to the SWISS-PROT database, and the position-specific score matrix generated is then compared to a shuffled version of SWISS-PROT. The median is used for the average of the low *E*-values, and the mean otherwise.

Table 3. The number of SWISS-PROT sequences yielding alignments with E-value ≤0.01, and relative running times, for Smith–Waterman and various versions of BLAST

Protein family	Query	Smith–Waterman	Original BLAST	Gapped BLAST	PSI-BLAST
Serine protease	P00762	275	273	275	286
Serine protease inhibitor	P01008	108	105	108	111
Ras	P01111	255	249	252	375
Globin	P02232	28	26	28	623
Hemagglutinin	P03435	128	114	128	130
Interferon α	P05013	53	53	53	53
Alcohol dehydrogenase	P07327	138	128	137	160
Histocompatibility antigen	P10318	262	241	261	338
Cytochrome P450	P10635	211	197	211	224
Glutathione transferase	P14942	83	79	81	142
H^+-transporting ATP synthase	P20705	198	191	197	207
Normalized running time		36	1.0	0.34	0.87

To score and evaluate the significance of the alignments found, the original BLAST program uses BLOSUM-62 substitution scores (18) and sum-statistics (21,22). The Smith–Waterman and gapped BLAST programs use BLOSUM-62 substitution scores, $10 + k$ gap costs, and the statistics of equations **1** and **2**, in conjunction with the experimentally determined parameters $\lambda_g = 0.255$ and $K_g = 0.035$ (3). PSI-BLAST uses the same gap costs and λ_g, but applied to the position-specific score matrix constructed from the output of the gapped BLAST run. Only one PSI-BLAST iteration is executed. All three BLAST programs use the same parameter settings as in Table 2, except that T is set to 11. Normalized running times are the mean ratio of program running time to that for the original BLAST. The time for PSI-BLAST includes the time for the initial BLAST search.

To compare the performance of the new gapped version of BLAST and its PSI-BLAST extension to that of the Smith–Waterman algorithm (24) and the original ungapped BLAST algorithm, we employed the same 11 query sequences that were used above to investigate the accuracy of PSI-BLAST statistics. Because, as shown, these statistics are quite accurate, we may use the number of statistically significant sequences found in a database search as a reasonable measure of algorithm sensitivity. We employed the *ssearch* program, version 2.0u54, from the Fasta package (2) as our implementation of the Smith–Waterman algorithm. Using each of the 11 queries, we searched SWISS-PROT with each of the four programs. We show in Table 3 the numbers of sequences found with E value ≤0.01, as well as the average ratio of running time to that for the original BLAST program. Based upon SWISS-PROT annotation, all sequences recorded in Table 3 appear to be true family members, with the exception of one of the lowest-scoring alignments found by Smith–Waterman when applied to the histocompatibility antigen query, and the lowest-scoring alignment found by the original BLAST applied to the hemagglutinin query. While some alignments involve hypothetical proteins, the pattern of conserved residues in all such cases suggests a true positive.

As can be seen, the gapped BLAST program runs on average three times faster then the original, and in all but one case examined finds a greater number of statistically significant alignments. It runs >100 times faster than Smith–Waterman, but for the combined 11 queries misses only eight of the 1739 significant similarities found by the rigorous algorithm. Of these eight, only one has an E-value <0.001, and another appears to be a random as opposed to a biologically meaningful similarity. The scores produced by gapped BLAST for the 1731 similarities it finds differ from those produced by the Smith–Waterman algorithm in only two instances. The discrepancy arises in both cases from an X_g parameter that is too low rather than from an incorrect choice of seed. Thus despite its simplicity, the seed-selection heuristic is extremely accurate.

A search that includes a single PSI-BLAST iteration still runs faster than the original BLAST, and 40 times faster than Smith–Waterman, but can in many cases be much more sensitive. It finds every true positive returned by Smith–Waterman, but frequently many others as well. Here only a single PSI-BLAST iteration has been considered but, as will be seen below, multiple iterations can yield even better results. Furthermore, we have found PSI-BLAST to perform better on searches of the non-redundant protein sequence database maintained by the NCBI (59) than on searches of SWISS-PROT, because of the greater number of significant similarities that are found by the initial BLAST run.

For the particular examples in Table 3, the PSI-BLAST iteration takes noticeably longer than the gapped BLAST iteration, due primarily to the time needed to construct the position-specific score matrix from the large number of significant local alignments found by BLAST. For queries that return a small number of significant alignments, each PSI-BLAST iteration requires more nearly the same time as BLAST.

PSI-BLAST EXAMPLES

In many instances, PSI-BLAST is able automatically to uncover biologically interesting similarities that elude simple database searches. Multiple iterations of PSI-BLAST are sometimes required to recognize the more distantly related protein family members. We here consider two representative cases in greater detail.

HIT proteins

Holm and Sander (60) describe how a comparison of three-dimensional structures is able to identify significant similarity between histidine triad (HIT) proteins and galactose-1-phosphate uridylyltransferase (GalT) proteins. Indeed, using the human HIT protein (61) as query, a BLAST search of SWISS-PROT reveals hits with E-value <0.01 only to other HIT proteins (Fig. 5a). An alignment to

the rat GalT protein (62) has the only marginally significant *E*-value of 0.012. A PSI-BLAST search, using the score matrix generated from the six alignments illustrated in Figure 5a, can immediately cement confidence in the biological relevance of this similarity. The *E*-value of the similarity with rat GalT drops to 2×10^{-4}, and an alignment to *Haemophilus influenzae* GalT (63) (Fig. 5b) receives the even more significant *E*-value of 4×10^{-5}. These similarities, of course, are uncovered using no structural information. In addition, on the next iteration, PSI-BLAST finds a strongly significant alignment (Fig. 5c; *E*-value 2×10^{-4}) to yeast 5′,5‴-P1,P4-tetraphosphate phosphorylase I (64), for which no structure is available.

BRCT proteins

Proteins containing one or multiple copies of the BRCT domain form a superfamily many of whose members are involved in DNA damage-responsive cell cycle checkpoints (65–67). While detailed analysis is needed to delineate completely this diverse set of proteins, PSI-BLAST is able to automatically identify most of the superfamily. We used the C-terminal 215 residues of human BRCA1 (68), which includes two BRCT domains (65), as the initial query for a search of NCBI's non-redundant protein sequence database. Using the default cutoff *E*-value of 0.01, the initial BLAST search recognized as significant only alignments to other BRCA1 sequences, and the previously described BRCT protein BARD (69) (Table 4). Subsequent PSI-BLAST iterations, however, retrieved all the proteins recorded in Table 4; additional close homologs are omitted from the table. Almost all the BRCT proteins described by Bork *et al.* (66) were recognized. Not found were the retinoblastoma family, whose putative BRCT domain is particularly divergent, worm R13A5.13, which was not in the database searched, and human DNA-ligase III. PSI-BLAST did report yeast RAD9 and YGR103w, the *Kluyveromyces lactis* RAP1 homolog, worm ZK675.2, and various terminal deoxynucleotidyltransferases and poly(ADP-ribose) polymerases, but all with *E*-values >0.01 (Table 4). Detailed examination of the alignments produced suggests that the only likely false positives involved a trypanosome EST (70) and the *Methanococcus jannaschii* mutT protein (71), the latter despite its involvement in DNA repair (Table 4).

Seven recent additions to the protein databases are reported here as members of the BRCT superfamily (Table 4). (i) *Arabidopsis* T10M13.12 (72), is the first plant protein observed to contain BRCT domains. (ii) KIAA0259 (73) is a large human protein of unknown function with eight BRCT domains, the greatest number so far observed within a single protein. (iii) T13F2.3 (74) is a worm protein with a 500-residue low-complexity (57) N-terminus. (iv) SPAC6G9.12 (75) is a fission yeast protein strongly similar to the previously recognized (66) yeast BRCT protein L8543.18 (76). (v) C36A4.8 (74) is a worm protein whose C-terminus contains a single BRCT domain, and whose N-terminus, containing a RING finger domain, is strongly similar to that of BRCA1. The similar organization to BRCA1 makes this protein of particular interest. (vi) *Synechocystis* sp. D90904 (77) is the first bacterial BRCT protein that is not a bacterial ligase. While it failed to pass the cutoff *E*-value of 0.01, its C-terminal BRCT domain is very similar to that of several bacterial ligases, which presumably led to its incorrect classification as such in the databases. Most of the protein N-terminal to its BRCT domain consists of a coiled-coil domain. The actual *Synechocystis* sp. DNA ligase (77,78) is found by PSI-BLAST on the 13th iteration, with an *E*-value of 0.002.

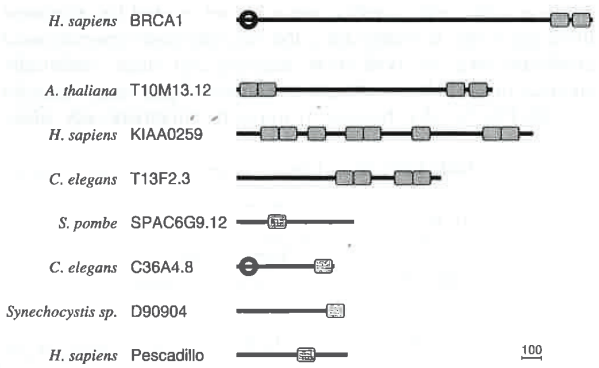


Figure 7. The location of BRCT domains within human BRCA1 (68), *Arabidopsis* T10M13.12 (72), human KIAA0259 (73), worm T13F2.3 (74), fission yeast SPAC6G9.12 (75), worm C36A4.8 (74), *Synechocystis* sp. D90904 (77) and human Pescadillo (79). BRCA1 and C36A4.8 each have, in addition, an N-terminal RING finger domain. The near identity to other worm sequences of a short region directly preceding the BRCT domain of C36A4.8 suggests the possibility that this protein has been misassembled.

(vii) Pescadillo is a human protein whose zebrafish ortholog is essential for embryonic development (79), and whose yeast ortholog YGR103w (80) has been previously recognized as a BRCT protein (66,67). It failed to pass the cutoff *E*-value of 0.01, but appeared with near-significant *E*-values in PSI-BLAST output from the 5th iteration onward. The approximate positions of the BRCT domains within BRCA1 and the seven newly identified BRCT proteins are illustrated in Figure 7.

DISCUSSION AND CONCLUSION

In addition to the major algorithmic changes described above, we have modified an aspect of the original BLAST program's output routine that on occasion caused important similarities to be overlooked. When a very large number of statistically significant alignments was found, BLAST would typically report only the top scoring 500. These alignments, however, might all involve one domain of the query that occurred frequently within the database. Interesting but weaker relationships to other regions of the query might simply be forced off the bottom of the list. Accordingly, following the general idea of Sonnhammer and Durbin (81), we have limited the number of alignments reported that involve each region of the query, but set no overall upper limit.

The BLAST programs are unlikely to remain static, and there are many possible avenues for future improvement. We discuss three of them briefly here.

Gap costs

Gapped alignments may be constructed using a variety of different types of gap cost. Because a single mutational event may insert or delete a large number of residues, it has been argued that long gaps should not cost much more than short ones, and *affine* gap costs, which assess a score $-(a + bk)$ for a gap of length k (82–85), have become the most widely used. A generalization of these costs has been proposed, that allows a gap to involve residues in both sequences rather than just one (86). Specifically, a gap in which k residues are inserted or deleted and j pairs of residues are left unaligned receives the score $-(a + bk + cj)$. The algorithm necessary

for using such costs is only a minor variant on that for traditional affine gap costs. In many cases, the new gap costs generate local alignments that are both more accurate and more statistically significant (86). These costs are potentially of particular value for use with PSI-BLAST, because by imposing alignment only where it is justified, they may lead to the construction of more sensitive position-specific score matrices. Whether it is desirable to use generalized affine gap costs as the default for general purpose database searches awaits detailed empirical study.

Table 4. PSI-BLAST protein database search results using the C-terminus of BRCA1 as query

Protein	Species	GenBank ID number	PSI-BLAST iteration	*E*-value
BARD	*Homo sapiens*	1710175	0	2e-06
T10M13.12[a]	*Arabidopsis thaliana*	2104545	1	4e-06
F26D2.b[b]	*Caenorhabditis elegans*	1914176	1	4e-04
KIAA0259[a]	*H.sapiens*	1665785	1	0.001
F37D6.1	*C.elegans*	1418521	2	4e-06
C19G10.07	*Schizosaccharomyces pombe*	1723501	2	6e-05
KIAA0170	*H.sapiens*	1136400	2	0.002
53BP1	*H.sapiens*	488592	2	0.008
T13F2.3[a]	*C.elegans*	1667334	3	2e-07
K04C2.4	*C.elegans*	470351	3	3e-07
T19E10.1	*C.elegans*	1067065	4	7e-04
Rad4/Cut5	*S.pombe*	730470	4	0.002
REV1	*Saccharomyces cerevisiae*	132409	4	0.003
ECT2	*Mus musculus*	423597	5	1e-04
XRCC1	*M.musculus*	627867	5	6e-04
Crb2	*S.pombe*	1449177	5	0.002
RAP1	*S.cerevisiae*	173558	5	0.006
TcEST030[c]	*Trypanosoma cruzi*	1536857	6	0.001
DPB11	*S.cerevisiae*	1352999	6	0.001
L8543.18	*S.cerevisiae*	1078075	6	0.010
SPAC6G9.12[a]	*S.pombe*	1644324	7	4e-04
YM8021.03	*S.cerevisiae*	1078533	7	0.005
YHR154w	*S.cerevisiae*	731729	7	0.008
C36A4.8[a]	*C.elegans*	1657667	7	0.010
UNE452	*S.cerevisiae*	1151000	8	8e-04
DNA ligase IV	*H.sapiens*	1706482	8	0.008
CDC9	*Candida albicans*	1706483	9	0.006
DNA ligase	*Thermus scotoductus*	1352293	10	0.010
GNF1	*Drosophila melanogaster*	544404	11	0.004
mutT[c]	*M.jannaschii*	2129134	15	0.008
RAD9	*S.cerevisiae*	131817	7	0.74
RAP1 homolog	*K.lactis*	422087	9	0.21
ZK675.2	*C.elegans*	599712	13	3.5
D90904[a]	*Synechocystis* sp.	1652299	15	0.17
TDT	*Mus domestica*	2149634	15	0.46
YGR103w	*S.cerevisiae*	1723693	16	0.017
Pescadillo[a]	*H.sapiens*	2194203	16	0.017
PPOL	*Sarcophaga peregrina*	1709741	16	0.060

Iteration zero refers to the initial BLAST run, using the 215 C-terminal residues of BRCA1 (68) (SWISS-PROT accession no. P38398) as query. Subsequent PSI-BLAST iterations use derived position-specific score matrices in place of the query. The score matrix for iteration $i + 1$ is constructed from alignments achieving an *E*-value ≤0.01 for iteration i. For each protein, the *E*-value is that returned during the PSI-BLAST iteration indicated, and precedes the protein's use for score matrix construction. Only one representative is listed for families of closely related proteins. On its 16th iteration PSI-BLAST uncovered no new proteins with *E*-value ≤0.01, and therefore ceased iteration. At the end of the table are shown BRCT proteins returned by PSI-BLAST with *E*-value >0.01 but ≤10, listed for the iteration in which they achieved their lowest *E*-value.

[a]Recent additions to the database, first identified as BRCT proteins here.

[b]The *C.elegans* F26D2.b protein (74) while a recent addition to the databases, is a close homolog of the previously recognized (66,67) family of *C.elegans* BRCT proteins containing, for example, F37A4.4 (90).

[c]The trypanosome EST (70) and the *M.jannaschii* mutT protein (71) are the only likely false positives.

Position-specific score matrices as input to PSI-BLAST

PSI-BLAST performs three distinct operations: it constructs a multiple alignment from BLAST output data; it processes this alignment into a position-specific score matrix; and it uses this matrix to search the database. A researcher may wish, however, to bypass the first two of these operations, and provide a score matrix as query directly to PSI-BLAST. The central difficulty is retaining the ability to calculate reliable statistics; as described above, PSI-BLAST imposes strict scaling rules on the matrices it generates, permitting the use of precomputed λ_g to assess significance. Three possible routes are open. (i) One may permit the specification of 20 target frequencies Q_i for each position rather than 20 scores. These can then be converted internally to log-odds scores with the appropriate scale for precomputed statistical parameters to apply. (ii) One may estimate by random simulation the statistical parameters for an input score matrix (3). An advantage is the applicability to a greater range of scoring systems, including the possible use of position-specific gap costs. A disadvantage is that obtaining reasonably accurate estimates of the relevant statistical parameters may increase the program's execution time unduly. (iii) One may abandon the statistical assessment of the alignments produced. This gives the greatest scope to input scoring systems, but precludes any reasonable scheme for automatic iteration of PSI-BLAST. Much experimentation will be required to determine which of these approaches is the most fruitful.

Realignment

After the initial BLAST run, or a later PSI-BLAST iteration, the multiple alignment used for subsequent iterations can be constructed in a more sophisticated manner than described above. Rather than coalescing all the pairwise alignments that pass the threshold immediately into a multiple alignment, the most significant among them can be used to build an initial multiple alignment and associated position-specific score matrix, which can then be used to rescore and realign database sequences that received lower scores. This step can be iterated several times before another full-scale database search is executed. There are several potential advantages to this procedure. (i) Weaker pairwise alignments, that may be somewhat inaccurate, can be improved and perhaps extended before they are incorporated into the evolving multiple alignment. (ii) Unrelated sequences that received chance high scores can have their scores downgraded by an improved matrix, and perhaps be rejected before they are included in the alignment. (iii) Related sequences that received relatively high alignment scores, but missed the threshold for inclusion, can have their scores increased, and perhaps be included in the multiple alignment. In short, the realignment procedure can prevent inaccurate pairwise alignments from corrupting the evolving multiple alignment, and can accelerate the recognition of related sequences, all for very little computational cost. Preliminary studies suggest this line of development to be a fruitful one.

In conclusion, the new gapped version of BLAST is both considerably faster than the original one, and able to produce gapped alignments. While the relevant statistical parameters can no longer be calculated from theory, random simulation allows them to be estimated beforehand for commonly used amino acid substitution matrices and gap costs. For many queries, the PSI-BLAST extension can greatly increase sensitivity to weak but biologically relevant sequence relationships. PSI-BLAST retains the ability to report accurate statistics, per iteration runs in times not much greater than gapped BLAST, and can be used both iteratively and fully automatically. These developments should enhance significantly the utility of database search methods to the molecular biologist.

Note

Source code for the new BLAST programs is available by anonymous ftp from the machine ncbi.nlm.nih.gov, within the directory 'blast', and the programs may be run from NCBI's web site at http://www.ncbi.nlm.nih.gov/

ACKNOWLEDGEMENTS

W.M. and Z.Z. are supported by grant LM05110 from the National Library of Medicine. We thank Dr Warren Gish for helpful conversations, Dr Eugene Koonin for assistance with the examples, and Dr Gregory Schuler for producing several of the figures.

REFERENCES

1 Altschul,S.F., Gish,W., Miller,W., Myers,E.W. and Lipman,D.J. (1990) *J. Mol. Biol.*, **215**, 403–410.
2 Pearson,W.R. and Lipman,D.J. (1988) *Proc. Natl. Acad. Sci. USA*, **85**, 2444–2448.
3 Altschul,S.F. and Gish,W. (1996) *Methods Enzymol.*, **266**, 460–480.
4 Chao,K.-M., Pearson,W.R. and Miller,W. (1992) *Comput. Appl. Biosci.*, **8**, 481–487.
5 Sellers,P.H. (1984) *Bull. Math. Biol.*, **46**, 501–514.
6 Altschul,S.F. and Erickson,B.W. (1986) *Bull. Math. Biol.*, **48**, 633–660.
7 Waterman,M.S. and Eggert,M. (1987) *J. Mol. Biol.*, **197**, 723–728.
8 Karlin,S. and Altschul,S.F. (1990) *Proc. Natl. Acad. Sci. USA*, **87**, 2264–2268.
9 Dembo,A., Karlin,S. and Zeitouni,O. (1994) *Ann. Prob.*, **22**, 2022–2039.
10 Altschul,S.F. (1991) *J. Mol. Biol.*, **219**, 555–565.
11 Altschul,S.F. (1993) *J. Mol. Evol.*, **36**, 290–300.
12 Smith,T.F., Waterman,M.S. and Burks,C. (1985) *Nucleic Acids Res.*, **13**, 645–656.
13 Collins,J.F., Coulson,A.F.W. and Lyall,A. (1988) *Comput. Appl. Biosci.*, **4**, 67–71.
14 Mott,R. (1992) *Bull. Math. Biol.*, **54**, 59–75.
15 Waterman,M.S. and Vingron,M. (1994) *Stat. Sci.*, **9**, 367–381.
16 Dayhoff,M.O., Schwartz,R.M. and Orcutt,B.C. (1978) In Dayhoff,M.O. (ed.), *Atlas of Protein Sequence and Structure*. National Biomedical Research Foundation, Washington, DC, Vol. 5, Suppl. 3, pp. 345–352.
17 Schwartz, R.M. and Dayhoff,M.O. (1978) In Dayhoff,M.O. (ed.), *Atlas of Protein Sequence and Structure*. National Biomedical Research Foundation, Washington, DC, Vol. 5, Suppl. 3, pp. 353–358.
18 Henikoff,S. and Henikoff,J.G. (1992) *Proc. Natl. Acad. Sci. USA*, **89**, 10915–10919.
19 Wilbur,W.J. and Lipman,D.J. (1983) *Proc. Natl. Acad. Sci. USA*, **80**, 726–730.
20 Robinson,A.B. and Robinson,L.R. (1991) *Proc. Natl. Acad. Sci. USA*, **88**, 8880–8884.
21 Karlin,S. and Altschul,S.F. (1993) *Proc. Natl. Acad. Sci. USA*, **90**, 5873–5877.
22 Altschul,S.F. (1997) In Suhai,S. (ed.), *Theoretical and Computational Methods in Genome Research*. Plenum Press, New York, pp. 1–14.
23 Needleman,S.B. and Wunsch,C.D. (1970) *J. Mol. Biol.*, **48**, 443–453.
24 Smith,T.F. and Waterman,M.S. (1981) *J. Mol. Biol.*, **147**, 195–197.
25 Sankoff,D. and Kruskal,J.B. (1983) *Time Warps, String Edits and Macromolecules: The Theory and Practice of Sequence Comparison*. Addison-Wesley, Reading, MA.
26 Bairoch,A. and Apweiler,R. (1997) *Nucleic Acids Res.*, **25**, 31–36.
27 Jou,W.M., Verhoeyen,M., Devos,R., Saman,E., Fang,R., Huylebroeck,D., Fiers,W., Threlfall,G., Barber,C., Carey,N. and Emtage,S. (1980) *Cell*, **19**, 683–696.
28 McLachlan,A.D. (1983) *J. Mol. Biol.*, **169**, 15–30.

29 Staden,R. (1984) *Nucleic Acids Res.*, **12**, 505–519.
30 Schneider,T.S., Stormo,G.D., Gold,L. and Ehrenfeucht,A. (1986) *J. Mol. Biol.*, **188**, 415–431.
31 Taylor,W.R. (1986) *J. Mol. Biol.*, **188**, 233–258.
32 Berg,O.G. and von Hippel,P.H. (1987) *J. Mol. Biol.*, **193**, 723–750.
33 Dodd,I.B. and Egan,J.B. (1987) *J. Mol. Biol.*, **194**, 557–564.
34 Gribskov,M., McLachlan,A.D. and Eisenberg,D. (1987) *Proc. Natl. Acad. Sci. USA*, **84**, 4355–4358.
35 Patthy,L. (1987) *J. Mol. Biol.*, **198**, 567–577.
36 Stormo,G.D. and Hartzell,G.W. III (1989) *Proc. Natl. Acad. Sci. USA*, **86**, 1183–1187.
37 Tatusov,R.L., Altschul,S.F. and Koonin,E.V. (1994) *Proc. Natl. Acad. Sci. USA*, **91**, 12091–12095.
38 Yi,T.-M. and Lander,E.S. (1994) *Protein Sci.*, **3**, 1315–1328.
39 Henikoff,S. and Henikoff,J.G. (1997) *Protein Sci.*, **6**, 698–705.
40 Bucher,P., Karplus,K., Moeri,N. and Hofmann,K. (1996) *Comput. Chem.*, **20**, 3–23.
41 Sellers, P.H. (1980) *J. Algorithms*, **1**, 359–373.
42 Lawrence,C.E., Altschul,S.F., Boguski,M.S., Liu,J.S., Neuwald,A.F. and Wootton,J.C. (1993) *Science*, **262**, 208–214.
43 Altschul,S.F., Carroll,R.J. and Lipman,D.J. (1989) *J. Mol. Biol.*, **207**, 647–653.
44 Sibbald,P.R. and Argos,P. (1990) *J. Mol. Biol.*, **216**, 813–818.
45 Sander,C. and Schneider,R. (1991) *Proteins*, **9**, 56–68.
46 Gerstein,M., Sonnhammer,E.L. and Chothia,C. (1994) *J. Mol. Biol.*, **236**, 1067–1078.
47 Henikoff,S. and Henikoff,J.G. (1994) *J. Mol. Biol.*, **243**, 574–578.
48 Thompson,J.D., Higgins,D.G. and Gibson,T.J. (1994) *Comput. Appl. Biosci.*, **10**, 19–29.
49 Eddy,S.R., Mitchison,G. and Durbin,R. (1995) *J. Comput. Biol.*, **2**, 9–23.
50 Gotoh,O. (1995) *Comput. Appl. Biosci.*, **11**, 543–551.
51 Krogh,A. and Mitchison,G. (1995) In Rawlings,C., Clark,D., Altman,R., Hunter,L., Lengauer,T. and Wodak,S. (eds.), *Proceedings of the Third International Conference on Intelligent Systems for Molecular Biology.* AAAI Press, Menlo Park, CA, pp. 215–221.
52 Henikoff,J.G. and Henikoff,S. (1996) *Comput. Appl. Biosci.*, **12**, 135–143.
53 Brown,M., Hughey,R., Krogh,A., Mian,I.S., Sjölander,K. and Haussler,D. (1993) In Hunter,L., Searls,D. and Shavlik,J. (eds.), *Proceedings of the First International Conference on Intelligent Systems for Molecular Biology.* AAAI Press, Menlo Park, CA, pp. 47–55.
54 Bailey,T.L. and Gribskov,M. (1996) In States,D.J., Agarwal,P., Gaasterland,T., Hunter,L. and Smith,R. (eds.), *Proceedings of the Fourth International Conference on Intelligent Systems for Molecular Biology.* AAAI Press, Menlo Park, CA, pp. 15–24.
55 Karplus,K. (1995) In Rawlings,C., Clark,D., Altman,R., Hunter,L., Lengauer,T. and Wodak,S. (eds.), *Proceedings of the Third International Conference on Intelligent Systems for Molecular Biology.* AAAI Press, Menlo Park, CA, pp. 188–196.
56 Sjölander,K., Karplus,K., Brown,M., Hughey,R., Krogh,A., Mian,I.S. and Haussler,D. (1996) *Comput. Appl. Biosci.*, **12**, 327–345.
57 Wootton,J.C. and Federhen,S. (1993) *Comput. Chem.*, **17**, 149–163.
58 Altschul,S.F., Boguski,M.S., Gish,W. and Wootton,J.C. (1994) *Nature Genet.*, **6**, 119–129.
59 Benson,D.A., Boguski,M.S., Lipman,D.J. and Ostell,J. (1997) *Nucleic Acids Res.*, **25**, 1–6.
60 Holm,L. and Sander,C. (1997) *Structure*, **5**, 165–171.
61 Ohta,M., Inoue,H., Cotticelli,M.G., Kastury,K., Baffa,R., Palazzo,J., Siprashvili,Z., Mori,M., McCue,P., Druck,T., Croche,C.M. and Huebner,K. (1996) *Cell*, **84**, 587–597.
62 Heidenreich,R.A., Mallee,J. and Segal,S. (1993) *DNA Seq.*, **3**, 311–318.
63 Maskell,D.J., Szabo,M.J., Deadman,M.E. and Moxon,E.R. (1992) *Mol. Microbiol.*, **6**, 3051–3063.
64 Plateau,P., Fromant,M., Schmitter,J.M., Buhler,J.M. and Blanquet,S. (1989) *J. Bacteriol.*, **171**, 6437–6445.
65 Koonin,E.V., Altschul,S.F. and Bork,P. (1996) *Nature Genet.*, **13**, 266–268.
66 Bork,P., Hofmann,K., Bucher,P., Neuwald,A.F., Altschul,S.F. and Koonin,E.V. (1997) *FASEB J.*, **11**, 68–76.
67 Callebaut,I. and Mornon,J.-P. (1997) *FEBS Lett.*, **400**, 25–30.
68 Miki,Y., Swensen,J., Shattuck-Eidens,D., Futreal,P.A., Harshman,K., Tavtigian,S., Liu,Q., Cochran,C., Bennett,L.M., Ding,W., *et al.* (1994) *Science*, **266**, 66–71.
69 Wu,L.C., Wang,Z.W., Tsan,J.T., Spillman,M.A., Phung,A., Xu,X.L., Yang,M.C., Hwang,L.Y., Bowcock,A.M. and Baer,R. (1996) *Nature Genet.*, **14**, 430–440.
70 Tanaka,T. and Tanaka, M. (1996) DDBJ accession no. D87228.
71 Bult,C.J., White,O., Olsen,G.J., Zhou,L., Fleischmann,R.D., Sutton,G.G., Blake,J.A., FitzGerald,L.M., Clayton,R.A., Gocayne,J.D., *et al.* (1996) *Science*, **273**, 1058–1073.
72 Johnson,A.F., de la Bastide,M., Lodhi,M., Hoffman,J., Hasegawa,A., Gnoj,L., Gottesman,T., Granat,S., Hameed,A., Kaplan,N., *et al.* (1997) GenBank accession no. 2104545.
73 Nagase,T., Seki,N., Ishikawa,K., Ohira,M., Kawarabayasi,Y., Ohara,O., Tanaka,A., Kotani,H., Miyajima,N. and Nomura,N. (1996) *DNA Res.*, **3**, 321–329.
74 Wilson,R., Ainscough,R., Anderson,K., Baynes,C., Berks,M., Bonfield,J., Burton,J., Connell,M., Copsey,T., Cooper,J., *et al.* (1994) *Nature*, **368**, 32–38.
75 Barrell,B.G., Rajandream,M.A. and Connor,R.E. (1996) EMBL accession no. Z81317.
76 Johnston,M., Andrews,S., Brinkman,R., Cooper,J., Ding,H., Du,Z., Favello,A., Fulton,L., Gattung,S., Greco,T., *et al.* (1994) GenBank accession no. 662142.
77 Kaneko,T., Sato,S., Kotani,H., Tanaka,A., Asamizu,E., Nakamura,Y., Miyajima,N., Hirosawa,M., Sugiura,M., Sasamoto,S., *et al.* (1996) *DNA Res.*, **3**, 109–136.
78 Tabata,S. (1996) GenBank accession no. 1651660.
79 Allende,M.L., Amsterdam,A., Becker,T., Kawakami,K., Gaiano,N. and Hopkins,N. (1996) *Genes Dev.*, **10**, 3141–3155.
80 Hernandez,K., Weber,N., Wipfli,P. and Schmidheini,T. (1996) EMBL accession no. Z72888.
81 Sonnhammer,E.L. and Durbin,R. (1994) *Comput. Appl. Biosci.*, **10**, 301–307.
82 Gotoh,O. (1982) *J. Mol. Biol.*, **162**, 705–708.
83 Fitch,W.M. and Smith,T.F. (1983) *Proc. Natl. Acad. Sci. USA*, **80**, 1382–1386.
84 Altschul,S.F. and Erickson,B.W. (1986) *Bull. Math. Biol.*, **48**, 603–616.
85 Myers,E.W. and Miller,W. (1988) *Comput. Appl. Biosci.*, **4**, 11–17.
86 Altschul,S.F. *Proteins*, in press.
87 Richardson,M., Dilworth,M.J. and Scawen,M.D. (1975) *FEBS Lett.*, **51**, 33–37.
88 Matsuda,G., Maita,T., Braunitzer,G. and Schrank,B. (1980) *Hoppe-Seyler's Z. Physiol. Chem.*, **361**, 1107–1116.
89 Tokunaga,O., Yaegashi,T., Lowe,J., Dobbs,L. and Padmanabhan,R. (1986) *Virology*, **155**, 418–433.
90 Fulton,L. and Waterston,R. (1995) GenBank accession no. 1176713.