Protein Science (1995), *4*:1145–1160. Cambridge University Press. Printed in the USA.
Copyright © 1995 The Protein Society

Comparison of methods for searching protein sequence databases

WILLIAM R. PEARSON
Department of Biochemistry, University of Virginia, Charlottesville, Virginia 22908
(RECEIVED September 28, 1994; ACCEPTED March 21, 1995)

Abstract

We have compared commonly used sequence comparison algorithms, scoring matrices, and gap penalties using a method that identifies statistically significant differences in performance. Search sensitivity with either the Smith–Waterman algorithm or FASTA is significantly improved by using modern scoring matrices, such as BLOSUM45–55, and optimized gap penalties instead of the conventional PAM250 matrix. More dramatic improvement can be obtained by scaling similarity scores by the logarithm of the length of the library sequence (ln()-scaling). With the best modern scoring matrix (BLOSUM55 or JO93) and optimal gap penalties (−12 for the first residue in the gap and −2 for additional residues), Smith–Waterman and FASTA performed significantly better than BLASTP. With ln()-scaling and optimal scoring matrices (BLOSUM45 or Gonnet92) and gap penalties (−12, −1), the rigorous Smith–Waterman algorithm performs better than either BLASTP and FASTA, although with the Gonnet92 matrix the difference with FASTA was not significant. Ln()-scaling performed better than normalization based on other simple functions of library sequence length. Ln()-scaling also performed better than scores based on normalized variance, but the differences were not statistically significant for the BLOSUM50 and Gonnet92 matrices. Optimal scoring matrices and gap penalties are reported for Smith–Waterman and FASTA, using conventional or ln()-scaled similarity scores. Searches with no penalty for gap extension, or no penalty for gap opening, or an infinite penalty for gaps performed significantly worse than the best methods. Differences in performance between FASTA and Smith–Waterman were not significant when partial query sequences were used. However, the best performance with complete query sequences was obtained with the Smith–Waterman algorithm and ln()-scaling.

Keywords: BLAST; FASTA; PAM250; sequence similarity; Smith–Waterman

The concurrent development of rapid methods for molecular cloning, DNA sequencing, high-performance computer workstations, and rapid protein and DNA sequence comparison algorithms has revolutionized the practice of molecular biology. Newly determined sequences are routinely compared against large sequence databases, and inferences about structure and function are frequently based on sequence similarity. Predicted growth of sequence databases and the advent of large-scale DNA sequencing projects have prompted increased interest in better methods for comparing protein and DNA sequences. As a result, several rapid biological sequence comparison algorithms (Pearson & Lipman, 1988; Altschul et al., 1990) have become used widely, and there has been considerable discussion of the best scoring parameters for sequence comparison algorithms (Collins et al., 1988; Karlin & Altschul, 1990; Altschul, 1991; Gonnet et al., 1992; Henikoff & Henikoff, 1992, 1993; Johnson & Overington, 1993).

Until recently, it was rare for descriptions of new algorithms and scoring matrices to be supported with a comprehensive evaluation of the approach on a wide range of protein sequences and superfamilies. Biological sequence comparison algorithms must balance sensitivity—the ability to calculate high-ranking scores for distantly related sequences—with selectivity—the ability to calculate low-ranking scores for unrelated sequences. Sensitivity and selectivity may vary for different protein families; the best algorithm or scoring parameters for finding G-protein-coupled receptors may not be the best for finding members of the hemoglobin or serine protease families (Collins et al., 1988; Altschul, 1993).

Here we examine the ability of several sequence comparison algorithms to identify distantly related (homologous) proteins by searching protein sequence databases. The goal of identifying distant relationships by database search is different from that of finding the most statistically significant sequence similarities

Reprint requests to: William R. Pearson, Department of Biochemistry, Jordan Hall #440, University of Virginia, Charlottesville, Virginia 22908; e-mail: wrp@virginia.edu.

(Collins et al., 1988; Altschul, 1991, 1993) or of finding the most accurate sequence alignments (Johnson & Overington, 1993; Vingron & Waterman, 1994). Statistical significance may not reflect homology; unrelated sequences may have statistically significant similarities due to sequence convergence, e.g., in transmembrane domains or DNA binding domains. Conversely, homologous sequences may not have statistically significant similarity scores. Likewise, the reference standard for sequence alignments—the alignment of domains of secondary and tertiary structure—may require a global alignment strategy and low gap extension penalties that are poorly suited to searches of sequence databases that contain large numbers of unrelated sequences.

In an earlier paper (Pearson, 1991), a method for evaluating different sequence comparison algorithms was introduced and the performance of FASTA, BLASTP, and Smith-Waterman was examined using query sequences from 34 families of protein sequences. In that approach, one algorithm (or scoring matrix) was better than another if it could identify more related sequences. "Identification" was operationally defined as the ability to calculate a similarity score for a related sequence that was higher than the scores of all but 0.5% of the unrelated sequences. This criterion was used to evaluate the BLOSUM series amino acid replacement scoring matrices; these matrices perform significantly better than the PAM series when used with the BLASTP and FASTA programs (Henikoff & Henikoff, 1992, 1993).

This paper extends the earlier work in four ways: (1) A new criterion for identification, the "equivalence number," is introduced. The equivalence number summarizes both the sensitivity and selectivity of a search. (2) Differences in performance are evaluated using a nonparametric statistical test, the "sign test." The sign test allows us to conclude that if BLASTP performs better than Smith-Waterman on 31 sequences but worse on 43, the difference in performance is not statistically significant. (3) Conclusions are based on 134 query sequences from 67 protein superfamilies. (4) Scoring matrices based on five different data sets are evaluated with a broad range of gap penalties.

Results

We have examined the performance of the most commonly used sequence comparison algorithms—BLAST (Altschul et al., 1990), FASTA (Pearson & Lipman, 1988), and Smith-Waterman (Smith & Waterman, 1981)—by comparing the ability of an algorithm to identify distantly related sequences using a single query sequence. Algorithms and scoring parameters were tested using two query sequences from each of 67 protein superfamilies. The query sequences and superfamilies used in this study are shown in Table 1.

A new criterion for search performance—The equivalence number

In our earlier comparisons, we considered a related sequence "found" if it obtained a score that was higher than the score of the highest-scoring 0.5% of the unrelated sequences (Pearson, 1991). For example (Table 2A), when human rhodopsin (OOHU) was compared to the augmented PIR1 database with the Smith-Waterman algorithm and the PAM250 scoring matrix, the highest-scoring unrelated sequence had a local similarity score of 138, and 62 unrelated sequences (0.5% of 12,219 sequences in the library) obtained scores of 67 or higher. Eleven members of the G-protein-coupled receptor family in the sequence database obtained scores lower than 67; thus, by this criterion, the Smith-Waterman algorithm "missed" 11 sequences.

A more informative measure of performance—the "equivalence number"—is introduced here. The equivalence number is the number of related sequences missed at a similarity score that balances the number of related sequences below the value and the number of unrelated sequences with scores at or above the value, i.e., the score where the number of false-positives equals the number of false-negatives. Thus, for the human rhodopsin sequence OOHU, there were 19 non-G-protein-coupled receptors with similarity scores equal to or greater than 119 and 19 G-protein-coupled receptors with scores less than 119 (Table 2A). The equivalence number summarizes both the sensitivity and the selectivity of a particular search and is more informative for query sequences that produce a small number of high scores for unrelated sequences. In one search with the Smith-Waterman algorithm, using the PAM250 matrix with gap penalties of −12 for the first residue in the gap and −2 (−12, −2) for each additional residue, the number of sequences missed with the equivalence number criterion was greater than or equal to the number missed at the 0.5% criterion (i.e., the criterion was as strict) for all 67 superfamilies examined (Table 2B). For this search, the equivalence number criterion was as strict as a 0.1% criterion for all but five superfamilies. Thus, the equivalence number provides a much more demanding test than the 0.5% criterion of the performance of different search algorithms and scoring parameters.

Table 3 shows the number of related sequences missed at the equivalence number for three commonly used search programs: BLASTP (Altschul et al., 1990), FASTA (Pearson & Lipman, 1988), and Smith-Waterman (Smith & Waterman, 1981). Here BLASTP[1] calculated similarity scores for 35 members of the 505-member globin family that were lower than or equal to the scores of 35 unrelated sequences (equivalence number = 35). For the same query sequence and protein database, the equivalence number was 43 for FASTA with *ktup* = 2. The equivalence number was 17 when FASTA was performed with *ktup* = 1 and optimized similarity scores were calculated (Pearson, 1990). With Smith-Waterman, the equivalence number was 16. Thus, for this member of the globin superfamily, the Smith-Waterman algorithm performed better than BLASTP or FASTA, missing only 16 of 505 members of the family; FASTA, with *ktup* = 2, performed the worst, missing 43 sequences. Differences in the performance of FASTA or Smith-Waterman with respect to BLASTP in Table 3 are indicated by a "+" if FASTA or Smith-Waterman performs better than BLASTP (the equivalence number is lower) and "−" if BLASTP performs better. The pattern of +'s and −'s in the right-most column shows that Smith-Waterman performs better than BLASTP on slightly less than 50% of the query sequences (16 of 33 sequences where the performance is different).

[1] BLASTP can rank library-sequence similarity scores based on one of three criteria: the expectation based on sum statistics (sum-*P*, the default), the expectation based on Poisson statistics (Poisson-*P*, used in earlier versions), and the score of the best high scoring segment pair (HSP). Sum-statistics rankings were used in Table 3.

Table 1. *Sequences examined*

Query1	Query2	Description/superfamily	Query length	Family size
HAHU	HBRNW	Hemoglobin α/β	141/146	505
K1HUAG	K3HU15	Ig κ chain V-I region	108/115	280
OOHU	HSSRCPT1F	G-protein-coupled receptors	348/366	165
CCHU	CCOS	Cytochrome *c*	105/104	142
N2KF1U	H3NJ1W	Snake neurotoxin	74/60	109
XURT8C	A45567	Glutathione transferase	222/211	106
TPHUCS	PVTFB3	Calcium binding EF-hand	159/109	106
OKHU2C	TVBEPN	Protein kinase, cAMP-dependent	351/390	97
FEPE	FEWT	Ferredoxin	54/97	93
RKMDS	RKRZS6	Ribulose-bisphosphate carboxylase	139/175	77
K3HU	MGMSB2	Ig κ chain C region	106/119	74
HMIVV	HMIVC2	Hemagglutinin	567/564	73
HLHUB2	HLCHBL	Histocompatibility antigen	338/231	71
IPHU	IPGP	Insulin	110/110	69
CYBOA	CYRTA	α-Crystallin chain A	173/173	67
PSHU	PSSNK1	Phospholipase A2	148/118	58
DEHUGL	DELOG3	Glyceraldehyde-3-phosphate dehydrogenase	335/333	46
TVHURA	TVBYSR	Transforming protein (N-ras)	189/219	45
TRRT1	KYVH2C	Serine protease	246/218	45
GCHU	RHHUS	Glucagon precursor	180/108	44
PWHUA	PWKMA	H^+-transporting ATP synthase	553/508	43
HNNZS	HNNZP3	Hemagglutinin-neuraminidase	576/572	42
NRBO	NRGPB	Ribonuclease	124/128	40
IVHUI6	IVHO22	Interferon α-I-6	189/195	39
AJHUQ	AJECQ	Glutamate-ammonia ligase	373/469	39
AZBR	AZALCX	Azurin	129/129	38
VGNZSV	VGNZCD	Fusion protein – Sendai virus	565/662	36
O4HUD1	O4CKA3	Cytochrome P450	497/523	35
VPXRWA	A44052	Outer capsid protein VP8	280/772	34
FOVWH3	FOLJFP	Gag polyprotein	512/450	33
KRHUE	DMHU	Keratin	471/469	32
VHIV34	VHIVM1	Nucleoprotein – influenza A	498/498	31
W6WL18	W6WL43	E6 protein papillomavirus	158/155	29
R6HUP2	R6BY22	Acidic ribosomal protein P2	115/110	29
LZHU	LAPG	Lysozyme	130/122	28
NMIV	I46347	Exo-α-sialidase	454/466	27
IJHUCN	IJBODD	*N*-cadherin	906/809	27
P2WL	P2WLHS	L2 protein papillomavirus	507/473	27
NTSRIA	AMHB	Scorpion neurotoxin	64/18	26
W7WLHS	W7WLHS	E7 protein papillomavirus	98/98	26
LWBOA	LWPMA	H^+-transporting ATP synthase	75/81	26
DEPGLH	DEBSLM	L-Lactate dehydrogenase	333/318	26
W2WLE	W2WLE	E2 protein papillomavirus	322/322	26
NKVLAH	NKVLC3	Core antigen – hepatitis B	183/188	25
XHHU3	WMVZF3	Antithrombin-III	464/148	25
KIBET	D43675	Thymidine kinase	376/364	25
CFKKA	AFKTB	Phycocyanin	162/161	25
MFNZS	A60004	Matrix protein	348/375	24
TVHUM	TVMVFV	Transforming protein (myc)	439/484	24
TYTUY2	CLHRY2	Protamine Y2	34/30	24
DEHUAA	DEPOA2	Alcohol dehydrogenase a	375/380	23
ACHUA1	ACCH2N	Ionotropic acetylcholine receptor	457/528	23
PWHU6	PWWT6	H^+-transporting ATP synthase	226/386	23
QQBE1L	VGBE2H	Glycoprotein B	857/980	23
HSHU1B	HSDU1A	Histone H1b	218/121	22
LUHU	LUJF12	Annexin I	346/316	22
MNIV2K	MNIV62	Nonstructural protein NS2	121/121	22
CYBOB	CYMSG2	β-crystallin chain Bp	204/174	21
TISYO	TINPA2	Proteinase inhibitor	71/70	21
SMHU2	SMMR	Metallothionein	61/25	21
PEHU	CMSHB	Pepsin	388/381	20
DJHUAC	DJVZ4I	DNA-directed DNA polymerase	1,462/1,006	20
LCHU	STHU	Prolactin	227/217	20
LNHU1	WMVZEL	Hepatic lectin H1	291/143	20
VGIHE2	VGIHD6	E2 glycoprotein precursor	1,447/550	20
QRECBD	QRECBD	Vitamin B_{12} transporter btuD – *E. coli*	249/249	20
UART	VART	α-2u-Globulin precursor – rat	181/201	20

Table 2. *Criteria for "finding" a library sequence*[a]

A. Criteria for "finding" sequences[b]

Criterion	Related missed	Unrelated found	Score
0.0	24	0	138
0.1	20	13	119
0.2	17	25	112
0.5	11	62	67
Equiv.	19	19	119

B. Criterion stringency[c]

Equivalence number versus criterion	More strict	Same	Less strict
0.0	0	42	25
0.1	20	42	5
0.2	29	36	2
0.5	37	30	0

[a] Five different criteria for "finding" a library sequence are shown. The 0.0%, 0.1%, 0.2%, and 0.5% criteria define a library sequence to be "missed" if its similarity score is less than or equal to the score of the highest-scoring (0.0) unrelated sequence, the 13th highest-scoring related sequence (0.1%, 13 = (0.001 × 12,219) + 1), etc. The number of sequences missed at the equivalence number (Equiv.) is also shown.

[b] The number of sequences and corresponding similarity scores for the five criteria using the human rhodopsin query sequence OOHU with the Smith–Waterman function with a PAM250 scoring matrix and gap penalties of −12, −2. One hundred sixty-five of 12,219 library sequences are related to this query.

[c] Comparison of the equivalence number criterion to the fixed percent criteria for one query sequence from each of the 67 superfamilies.

Statistical significance of differences in performance

The sign test can be used to determine whether the differences in performance for the four algorithms in Table 3 are significant. This test uses the binomial distribution to estimate the probability that the distribution of differences in the performance of two algorithms (+'s and −'s in Table 3) would occur by chance. For example, when FASTA, *ktup* = 2 (PAM250 matrix), is compared with BLASTP with 67 query sequences, FASTA performs better (has a lower equivalence number) with only 3 query sequences and performs worse with 42 query sequences. Because we are testing whether the two methods perform differently, the null hypothesis is that each algorithm performs equivalently ($P = 0.5$), and one can calculate the probability of obtaining 3 heads and 42 tails (or 3 tails and 42 heads) in 45 tosses of a fair coin. In this case, $\mu = 45 \times 0.5$, $\sigma = [45 \times 0.5 \times (1 - 0.5)]^{1/2}$, so $z = (42 - 22.5)/11.25^{1/2} = 5.8$. Using the normal approximation to the binary distribution, the probability of obtaining a z-value of 5.8 or greater with a two-tailed test is 10^{-8} and we say that the difference in performance is significant.[2]

The statistical tests used in this report are conservative. Both the use of a two-tailed test and the $2^{1/2}$ correction for duplicate samples tend to underestimate differences in performance. Thus, a scoring matrix or gap penalty that performs worse with a z-value of 1.5 might consistently perform worse on distantly related sequences. Conversely, although a method that performs significantly worse in these tests will not perform worse with every protein family, it is likely to be less effective when examining very distant evolutionary relationships.

Table 4 summarizes z- and P-values for comparisons of BLASTP (version 1.4.7 with the BLOSUM62 scoring matrix) and Smith–Waterman (PAM250, −12, −4) with a variety of commonly used comparison methods and scoring parameters. The "sum" statistics used by BLASTP 1.4 perform significantly better than ranking by the best HSP score. "Sum" statistics also perform better, but not significantly, than the older "Poisson" statistics. BLASTP/BLOSUM62 performs significantly better than FASTA with *ktup* = 2 or *ktup* = 1 and either PAM250 or BLOSUM62 if optimized similarity scores are not used to rank similarities. BLASTP performs slightly better than FASTA with optimized PAM250 scores and slightly better than Smith–Waterman with PAM250. FASTA with *ktup* = 1, the BLOSUM62 matrix, and optimized scores performs better (but not quite significantly) than BLASTP/BLOSUM62 and slightly better than Smith–Waterman with PAM250 (Table 4B, bottom row). A graphical summary of Table 4 is given in Figure 1. In the right column, gray bars extending to the left beyond −2.0 indicate that BLASTP/BLOSUM62 (Fig. 1A) or Smith–Waterman (Fig. 1B) performed significantly better than the comparison method in the left column; gray bars extending to the right indicate that the method on the left performed better than BLASTP or Smith–Waterman.

Improved searches with ln()-scaled and regression-scaled scores

When rigorous methods like the Smith–Waterman algorithm perform better than BLASTP or FASTA, one assumes that they do so because they are more sensitive; rigorous methods examine every possible alignment between two sequences. Likewise, when BLASTP or FASTA performs as well as Smith–Waterman, one assumes that the increased selectivity of these heuristic methods must offset any losses due to their decreased sensitivity. We examined the highest scoring unrelated sequences in searches with several of the query sequences and found that often the highest ranked unrelated sequences were more than twice as long as the query sequence. Long unrelated library sequences should have higher similarity scores on average simply because there are more possible alignments to a long sequence than to a short sequence. Thus, we examined several methods for increasing the selectivity of the Smith–Waterman and FASTA algorithms by normalizing similarity scores with respect to the length of the library sequence.

We first examined a very simple scaling factor, $\ln(n_q)/\ln(n_l)$, where n_q is the length of the query sequence and n_l is the length of the library sequence. With this correction, a raw similarity score of 100 between a 200-residue query sequence and a 1,000-residue library sequence would be scaled to a score of 77 by the factor $\ln(200)/\ln(1{,}000) = 0.77$. This correction has the property that it both lowers the scores of long unrelated sequences and raises the scores of short related (or unrelated) sequences. It may be appropriate to increase the scores of partial amino acid sequences that are very similar to the query sequence

[2] For most of the comparisons in this paper, two sequences from each superfamily are examined and σ is divided by $2^{1/2}$ because the two observations of the superfamily are not independent.

Table 3. *Library search sensitivity*[a]

Query	Superfamily	Family size	Related – missed/unrelated – found			
			BLAST	*ktup* = 2	*ktup* = 1 opt	Smith-Waterman
HAHU	Hemoglobin	505	35	43 –	17 +	16 +
K1HUAG	Ig κ V region	280	54	100 –	17 +	21 +
OOHU	G-protein CR	165	27	42 –	26 +	22 +
CCHU	Cytochrome *c*	142	25	29 –	27 –	27 –
N2KF1U	Snake neurotoxin	109	3	5 –	4 –	4 –
XURT8C	Glutathione transferase	106	8	20 –	8	8
TPHUCS	Calcium binding	106	8	17 –	7 +	11 –
OKHU2C	Protein kinase	97	54	52 +	37 +	37 +
FEPE	Ferredoxin	93	53	54 –	53	53
RKMDS	Ribulose-bisphosphate carboxylase	77	0	1 –	0	0
K3HU	Ig κ C region	74	22	26 –	18 +	16 +
HMIVV	Hemagglutinin	73	1	4 –	0 +	0 +
HLHUB2	Histocompatibility Ag	71	2	15 –	1 +	0 +
IPHU	Insulin	69	3	3	3	3
CYBOA	α-Crystallin	67	7	8 –	4 +	4 +
PSHU	Phospholipase A2	58	2	2	2	2
DEHUGL	G3-PDH	46	0	1 –	1 –	1 –
TVHURA	Transforming protein (N-ras)	45	1	1	1	1
TRRT1	Serine protease	45	16	14 +	12 +	12 +
GCHU	Glucagon precursor	44	5	12 –	13 –	14 –
PWHUA	H^+-transporting ATP synthase	43	2	1 +	0 +	0 +
AJHUQ	Glutamate-ammonia ligase	39	1	10 –	1	1
AZBR	Azurin	38	27	27	22 +	22 +
O4HUD1	Cytochrome P450	35	1	8 –	4 –	2 –
VPXRWA	Outer capsid VP8	34	0	1 –	1 –	1 –
FOVWH3	Gag polyprotein	33	2	3 –	3 –	3 –
KRHUE	Keratin	32	4	5 –	4	4
VHIV34	Nucleoprotein	31	0	1 –	0	0
W6WL18	E6 protein	29	0	7 –	0	0
R6HUP2	Acidic ribosomal P2	29	5	6 –	2 +	2 +
NMIV	Exo-α-sialidase	27	0	3 –	1 –	1 –
NTSRIA	Scorpion neurotoxin	26	4	8 –	5 –	5 –
W7WLHS	E7 protein	26	1	5 –	0 +	0 +
LWBOA	H^+-transporting ATP synthase	26	1	3 –	2 –	2 –
DEPGLH	L-lactate DH	26	3	13 –	3	1 +
XHHU3	Antithrombin-III	25	0	4 –	1 –	1 –
KIBET	Thymidine kinase	25	0	1 –	1 –	1 –
CFKKA	Phycocyanin	25	0	2 –	0	0
MFNZS	Matrix protein	24	0	8 –	0	0
TYTUY2	Protamine Y2	24	2	2	2	2
DEHUAA	Alcohol DH	23	0	2 –	0	0
ACHUA1	Acetylcholine receptor	23	0	2 –	0	0
PWHU6	H^+-transporting ATP synthase	23	2	9 –	4 –	7 –
HSHU1B	Histone H1b	22	2	2	2	2
LUHU	Annexin I	22	4	4	4	4
MNIV2K	Nonstructural NS2	22	2	2	2	2
CYBOB	β-Crystallin	21	0	1 –	0	0
TISYO	Proteinase inhibitor	21	2	2	2	1 +
SMHU2	Metallothionein	21	5	6 –	5	5
DJHUAC	DNA polymerase	20	1	3 –	1	2 –
LNHU1	Hepatic lectin H1	20	4	11 –	7 –	8 –
VGIHE2	E2 glycoprotein	20	0	1 –	1 –	1 –
QRECBD	Vitamin B_{12} btuD	20	4	11 –	4	4
UART	α-2u-globulin	20	9	13 –	7 +	6 +
				3 +/42 –	15 +/15 –	16 +/17 –
			z-value	5.81	0	0.17
			p-value	10^{-8}	1	0.86

[a] The number of related sequences missed (the equivalence number) for 67 query sequences with four different comparison algorithms is shown. Where shown, + or – indicates the relative performance with respect to BLASTP (version 1.4.7 with BLOSUM62 scoring matrix; sum-statistics *P*-values were used to rank scores). FASTA and Smith-Waterman algorithms used the PAM250 matrix with gap penalties of −12, −4. Numbers at the bottom of the table summarize the relative performance of the different algorithms with respect to BLASTP. The *z*-values and *P*-values were calculated as described in the text.

Table 4. *Comparison of commonly used search algorithms*

Algorithm	+	−	±	z	P	Missed
A. BLASTP, BLOSUM62, sum-P versus[a]						
BLASTP HSP	18	43	108	2.3	0.024	933
Poisson-P	12	26	119	1.6	0.11	943
FASTA, $ktup = 2$	5	84	68	5.9	10^{-8}	1,415
(opt)	31	35	96	0.35	0.73	832
BLOSUM62	5	75	78	5.5	10^{-7}	1,305
(opt)	29	24	103	0.49	0.63	861
FASTA, $ktup = 1$	10	59	89	4.2	10^{-5}	1,120
(opt)	29	31	104	0.18	0.86	804
BLOSUM62	10	50	102	3.7	10^{-4}	1,006
(opt)	37	17	108	1.9	0.054	807
B. Smith-Waterman PAM250, −12, −4 versus[b]						
BLASTP, sum-P	33	29	102	0.36	0.72	885
HSP	29	44	94	1.2	0.21	933
Poisson-P	25	40	105	1.3	0.19	943
FASTA, $ktup = 2$	4	82	68	5.9	10^{-9}	1,415
(opt)	22	23	106	0.11	0.92	832
BLOSUM62	10	75	75	5.0	10^{-6}	1,305
(opt)	33	34	95	0.09	0.93	861
FASTA, $ktup = 1$	8	60	82	4.5	10^{-5}	1,120
(opt)	15	12	122	0.41	0.68	804
BLOSUM62	21	48	90	2.3	0.022	1,006
(opt)	37	27	105	0.88	0.38	807

[a] Comparison of BLASTP 1.4.7 with probabilities based on "sum" statistics (sum-P) to BLASTP HSP scores, BLASTP "Poisson" probabilities (Poisson-P), FASTA, and Smith-Waterman. The numbers of query sequences where the alternative matrices or algorithms perform better or worse than BLASTP with BLOSUM62 are shown. In addition, the numbers of query sequences (of 134 total) where the equivalence number differs by 1 or less (±), the z-values, and the P-values are shown. The z-values shown here are also plotted in Figure 1. The "Missed" column reports the total number of related sequences missed at the equivalence number with all 134 query sequences; if every related sequence were missed, this value would be 7,094 (3,547 related sequences × 2 query sequences from each family).

[b] Comparison of Smith-Waterman with the PAM250 matrix and gap penalties of −12, −4.

(>80% identical over 10–15 residues) but too short to produce a highly ranked score. Ln()-scaling considerably improves the effectiveness of both FASTA and Smith-Waterman. Ln()-scaled optimized FASTA scores with either $ktup = 2$ or $ktup = 1$ perform significantly better than unscaled Smith-Waterman scores (Fig. 2A). However, ln()-scaling significantly improves the performance of Smith-Waterman scores as well; with ln()-scaling, none of the other algorithms, including BLASTP and ln()-scaled FASTA, perform as well as ln()-scaled Smith-Waterman.

The excellent performance of ln()-scaled scores is not predicted by current statistical theory for local alignment scores (Arratia et al., 1986; Karlin & Altschul, 1990; Mott, 1992). For local similarity scores, the mean scores for two random sequences of length m and n is predicted to increase as $\ln(mn)$, whereas the variance of the local similarity scores is independent of the length of the two sequences. Thus, scores should be corrected by *subtracting* a factor related to $\ln(n_l)$, rather than by dividing by that factor. This was confirmed indirectly by experiment; when the performances of PAM250, BLOSUM45, 50, 55, and 62 were examined with infinite gap penalties, which ensures local behavior, there was no significant difference between the unscaled and ln()-scaled scores (data not shown and Fig. 5). However, Mott (1992) has observed that for PAM250 matrix and gap extension penalties less than 4 (e.g., the −2 used extensively in this study), the alignments produced between random sequences extend farther than would be expected for a purely local alignment. Thus, he suggested that commonly used search parameters may produce alignments that have a more global than local character. As the alignment ceases to be local, the variance of the similarity score becomes a function of the length of the two sequences. P. Green (pers. comm.) has suggested that ln()-scaling partially corrects for sequence length variance of similarity scores of unrelated sequences.

To test whether ln()-scaling was simply correcting for sequence-length-dependent variance, searches with a "regression-scaled" score that removes this dependence were examined. Regression-scaled scores perform significantly better than unscaled Smith-Waterman scores (Figs. 3A, 4B), but they do not perform as well as ln()-scaled scores (Fig. 3B) for most matrices and gap penalties (Fig. 4). This was unexpected, therefore we examined the possibility that partial sequences in the PIR1 library that are related to some of the query sequences might favor ln()-scaling. Regression scaling would not increase the scores of such short partial sequences, whereas the ln()-scaled score would. The search programs were modified to ignore all library sequences shorter than 25, 50, or 75 residues (excluding 105, 426, and 1,016 sequences, respectively) and tests with BLOSUM45 −10, −2 and −12, −1 were run. Ln()-scaled scores performed significantly better than regression-scaled scores in each test when short library sequences were excluded (data not shown).

Alternatively, because the superfamily data set used for these studies contains many sequences that should share global sequence similarity, the ln()-scaling may simply be providing a correction for differences in sequence length. If so, global scoring algorithms might perform even better than Smith-Waterman. The Smith-Waterman algorithm with ln()-scaling was compared with a global comparison algorithm (Myers & Miller, 1988) that either penalized end-gaps or did not (Fig. 3). Global alignment scores that did not penalize end-gaps and used the PAM250 matrix with gap penalties of −12, −2 performed better (although not significantly) than unscaled Smith-Waterman scores. Other gap penalties, ranging from −8, −1 to −12, −4 did not perform as well as Smith-Waterman scores with PAM250 −12, −2 (data not shown). We also examined the relative performance of ln()-scaling and regression-scaling with partial sequences (Fig. 6). Lacking theoretical justification for ln()-scaling, several alternative correction factors were examined. Ln()-scaling performs significantly better than scaling based on the ratio of the sequence lengths, or the square root of that ratio (Fig. 3). Ln()-scaling performs significantly better than either the global alignments or any of the simple alternative scaling functions and better than regression-scaling with many scoring matrices.

Comparison of scoring matrices and gap penalties

The PAM250 scoring matrix (Dayhoff et al., 1978), which is used widely for protein sequence comparisons, is more than 15 years old and is based on 1,572 amino acid substitutions from a database of about 120,000 residues in 1,100 sequences. Today, the protein databases include >30,000 sequences and $>10^7$ residues. In the past 4 years, several groups have described matri-

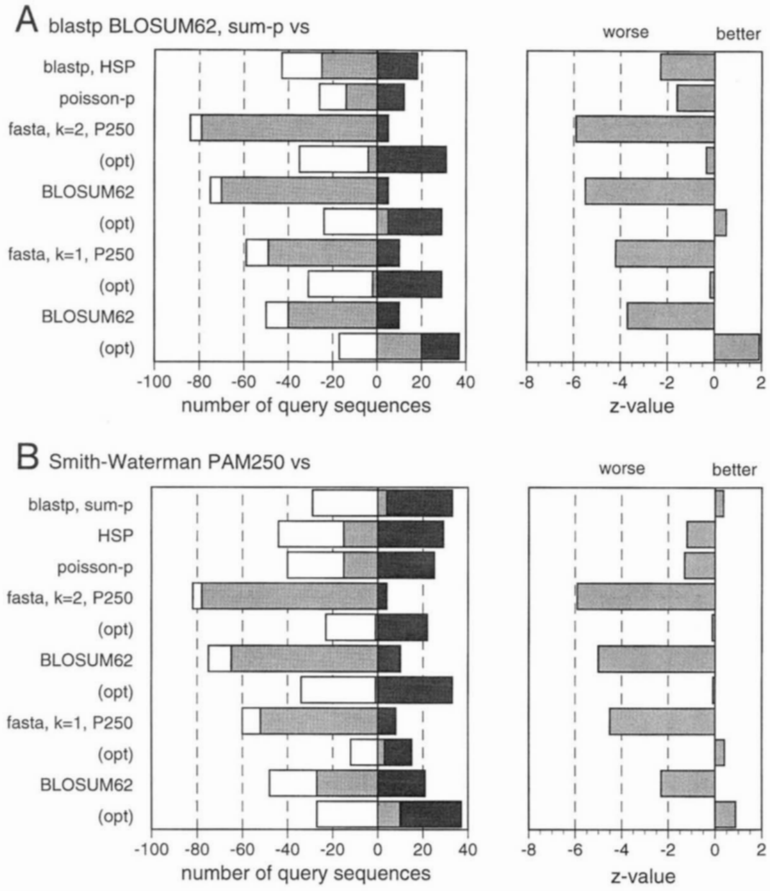

Fig. 1. Performance of commonly used sequence comparison algorithms. **A:** Comparison of BLASTP (version 1.4.7; see Table 1) with the FASTA (Pearson & Lipman, 1988) and Smith–Waterman (Smith & Waterman, 1981) algorithms. FASTA and Smith–Waterman searches were performed with the PAM250 scoring matrix and gap penalties of −12 for the first residue in the gap and −4 for each additional residue in the gap. FASTA comparisons were run with $ktup = 2$ ($k = 2$) and $ktup = 1$ ($k = 1$), with (opt) or without ranking by optimized scores (see Materials and methods) using either the PAM250 (P250) or BLOSUM62 matrix. BLASTP provides three different scores for ranking library sequences, probabilities based on "sum statistics" (Karlin & Altschul, 1993; sum-*P*), probabilities based on Poisson statistics (Poisson-*P*), or the single best high scoring segment pair score (HSP). Open bars indicate the number of query sequences (two queries per superfamily) where performance with BLASTP was better than performance with the algorithm indicated. Dark bars indicate the number of queries where the other algorithm performed better. Gray bars indicate the difference in the performance of the two algorithms. The right column shows the *z*-value for the relative performance of the algorithms. *z*-Values greater than 1.96 are statistically significant at the 0.05 level. The *P*-values associated with these *z*-values are shown in Table 4. The *P*-values associated with *z*-values of 1, 2, 3, 4, and 5 are 0.32, 0.046, 0.0027, 6×10^{-5}, and 6×10^{-7}, respectively. **B:** Performance compared with the Smith–Waterman algorithm, PAM250 matrix, gap penalties of −12, −4.

ces based on modern protein sequence databases, a number of which appear to differ substantially from PAM250. We examined the PAM250 (Dayhoff et al., 1978) matrix and several modern matrices, including JO93 (Johnson & Overington, 1993), which was derived from comparing structural alignments, Gonnet92 (Gonnet et al., 1992), which was derived from an "all-versus-all" comparison of a protein sequence database, and two families of matrices, the BLOSUM family (Henikoff & Henikoff, 1992) and a modern version of the PAM matrices (Jones et al., 1992). Because current statistical theory does not provide any guidance for the selection of gap penalties (Altschul, 1991), a range of gap penalties from −6, −1 to −16, −4 was tested for each matrix (Fig. 4).

With conventional (unscaled) Smith–Waterman scores, the best performance is obtained with the BLOSUM55, matrix and gap penalties of −12, −2. BLOSUM55 (−12, −2) performs significantly better than the widely used PAM250 matrix at every gap penalty except −16, −2, where the *z*-value was 1.9. The JO93 matrix and the JTT160 and JTT200 matrices with gap penalties from −10, −4 to −16, −2 (JO93) or −14, −2 (JTT200) also perform as well as BLOSUM55 (−12, −2). JTT250 and JTT320 perform significantly worse than BLOSUM55 (−12, −2) at every gap penalty examined. The BLOSUM45–50 and BLOSUM62.3 matrices (Fig. 4B) performed almost as well as BLOSUM55, with gap penalties ranging from −12, −2 to −16, −2. Two versions of the BLOSUM62 matrix were examined: BLOSUM62.5, which is the default matrix for the BLASTP program and has substitution values scaled in 1/2-bit units (Altschul, 1991), and BLOSUM62.3, which has entries scaled in the 1/3-bit units used by BLOSUM45-55. The BLOSUM62.5 matrix performed as well as BLOSUM62.3 and the other BLOSUM matrices when low gap penalties (−6, −4 and −8, −2) were used; when BLOSUM62.3 was examined, optimal gap penalties were in the same range as the other 1/3-bit scaled BLOSUM matrices.

When ln()-scaled Smith–Waterman scores are used, the best performance is obtained with the BLOSUM45 or the Gonnet92 matrices and gap penalties of −10, −2, −12, −1, or −12, −2 (Fig. 4A,B). The JO93 and JTT250 (data not shown) matrices performed poorly at every gap penalty; PAM250 performed poorly except at −10, −2, −12, −2, and −14, −1, where the differences were not statistically significant. Equivalent performance is obtained with BLOSUM45–62.3 and with BLOSUM62.5 if low gap penalties are used.

Figure 4 also compares performance with regression-scaled scores to performance with unscaled or ln()-scaled scores. Regression-scaling significantly improves the performance of almost every unscaled scoring matrix and gap penalty combination. In general, regression-scaled scores performed significantly worse than ln()-scaled scores; however, with the Gonnet92 matrix and several gap penalties and the BLOSUM50 matrix at −12, −1, the differences were not statistically significant. We show below that when partial sequences are used, regression-scaling is as effective as ln()-scaling (Fig. 6).

We also examined the effect of adding a constant value of +1, +2, −1, or −2 to each entry in the PAM250 and Gonnet92

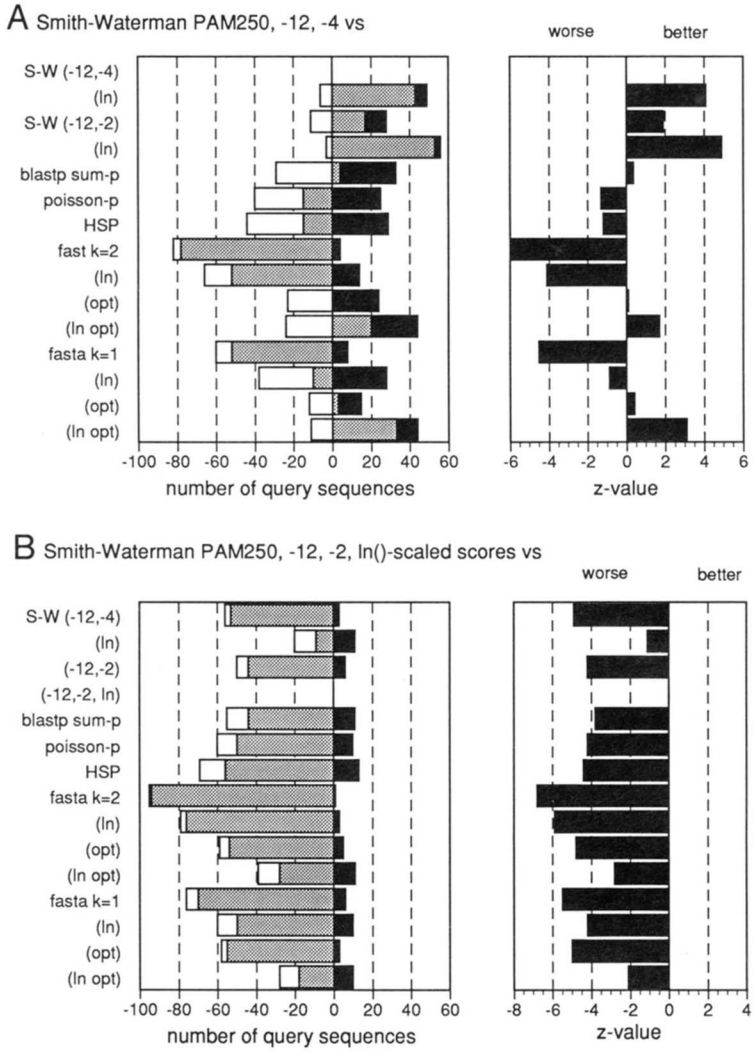

Fig. 2. Improved performance with ln()-scaling. The relative performance of the indicated algorithms is plotted as in Figure 1. **A:** Searches are compared to the standard Smith-Waterman algorithm, using PAM250, −12, −4. **B:** Comparison with Smith-Waterman, PAM250, −12, −2, ln()-scaled scores. The S-W P250 and S-W P250 (ln) rows indicate searches with gap penalties of −12, −4. FASTA searches used the PAM250 scoring matrix and gap penalties of −12, −4. BLASTP searches used the BLOSUM62 matrix. With the exception of S-W P250 −12, −4, ln(), all the methods perform significantly worse than Smith-Waterman, PAM250, −12, −2, ln()-scaling.

matrices (data not shown). Vingron and Waterman (1994) have suggested that adding a small positive value to the PAM250 matrix improves the robustness of some alignments. This is not true for similarity searches. For the PAM250 matrix, −12, −2, and unscaled scores, each offset produced significantly worse performance (data not shown). This was also true with ln()-scaled scores with the exception of an offset of −1, where performance was worse but the difference was not significant at the 0.05 level ($z = 1.2$). When +1 and +2 were added, z-values of −7 were obtained with PAM250. This dramatically lower performance for "biased" scoring matrices is consistent with the interpretation of a scoring matrix as resulting from target values for substitution frequencies (Altschul, 1991, 1993). From this perspective, an offset in the scoring matrix implies a dramatically different expectation for the number of substitutions. The same results were found with the Gonnet92 matrix; offsets of −2 . . . +2 significantly decreased performance.

Penalties of the form $q + rk$ where $r = 0$ (−12, 0; . . . ; −28, 0) performed dramatically worse than any of the affine costs, presumably because they allow very long gaps to improve the scores of unrelated sequences (Fig. 5). Likewise, penalties that charged a constant value for each residue in a gap ($q = 0$, e.g., −7, −7; −6, −6) perform significantly worse than the best affine penalties (−12, −2) with unscaled Smith-Waterman scores when the BLOSUM50 or BLOSUM62.5 matrices were used (similar results were obtained with Gonnet92 and PAM250, data not shown). With ln()-scaled scores, constant gap penalties perform almost as well as affine penalties with the BLOSUM50 and BLOSUM62.3 matrices. However, constant gap penalties do not perform as well as the best matrix and gap penalties (BLOSUM45, −12, −1) for ln()-scaled Smith-Waterman scores (Fig. 5B). The recommended scoring matrix and gap penalties (with $q = 0$) for BLITZ (mpsearch, Sturrock & Collins, 1993) were also examined (data not shown). BLITZ uses the PAM series of scoring matrices and gap penalties from −6, −6 for PAM300 to −13, −13 for PAM120. In every case, the constant penalty per residue performed worse than affine penalties, although for some matrices and gap penalties, the differences were not significant.

Figure 5 also presents comparisons of searches with $r = 10{,}000 \approx \infty$ and the three scoring methods used by BLASTP. Searches with $r \approx \infty$ with Smith-Waterman perform significantly worse than affine penalties (Fig. 5A,B) and about the same as BLASTP HSP scores (Fig. 5C). Thus, the heuristic approach

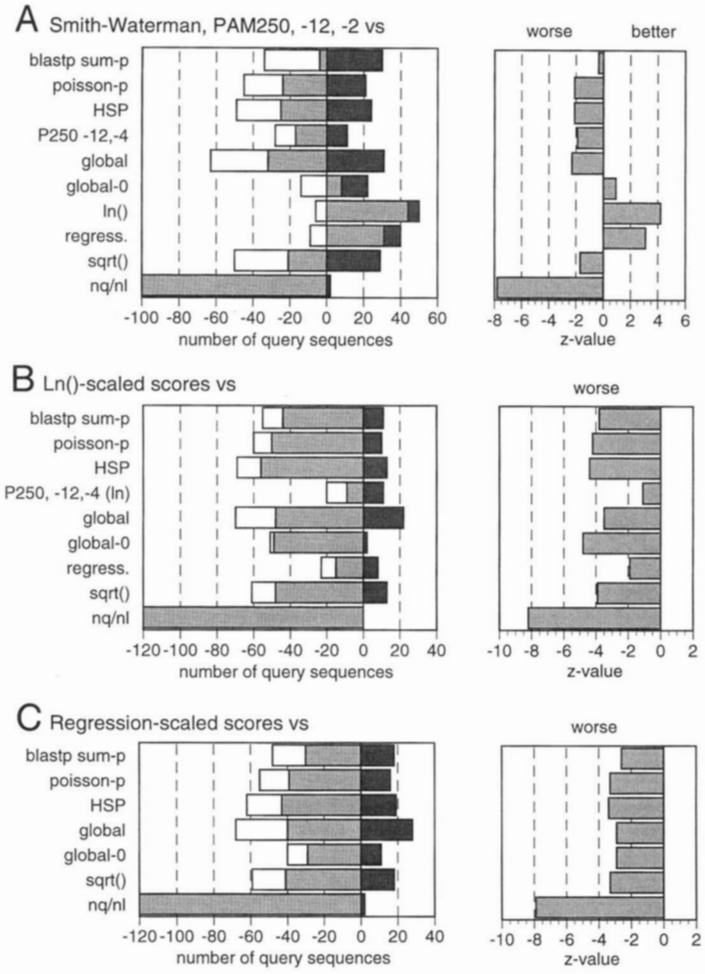

Fig. 3. Alternative normalization functions. The ln()-scaling normalization was compared with BLASTP scores, global alignment scores (global, global-0), regression scaled scores (regress.), the ratios of the lengths (n_q/n_l), and the square root of that ratio (sqrt()). Global alignment scores were calculated with (global) or without (global-0) penalties for terminal gaps (the terminal gap penalty was the same as the internal penalty). Smith–Waterman searches were done with PAM250 and gap penalties of −12, −2 unless noted. BLASTP searches used the BLOSUM62 scoring matrix. Searches with different normalization factors are compared with: (**A**) comparison with Smith–Waterman, PAM250, −12, −2; (**B**) ln()-scaled scores from Smith–Waterman, PAM250, −12, −2; (**C**) regression-scaled scores (Smith–Waterman, PAM250, −12, −2). Numbers of families for the n_q/n_l normalization rows are truncated: in A, n_q/n_l performed worse for 128 families (z-value = −7.8); in B, 133 ($z = -8.2$); and in C, 131, ($z = -7.9$).

used by BLASTP does not miss significant HSPs. The new sum-statistics-based BLASTP scores perform significantly better than HSP scores but not significantly better than the earlier Poisson statistics-based scores; thus, several HSPs must be combined to identify the most distant relationships. However, none of the BLASTP scores performed as well as either unscaled or ln()-scaled Smith–Waterman scores when optimal matrices and gap penalties are used (Fig. 5, filled symbols). Thus, the coarse segment combination approach used by BLASTP is not as effective as allowing gaps in alignments.

Performance with partial length queries

To test whether the unexpectedly good performance ln()-scaled scores reflect the "global" nature of the superfamily query sequences in the test data set, searches were performed with random portions of the query sequences. Two libraries of partial query sequences were constructed: one that simply selected a random subsequence of each query sequence, such as might be found in an EST sequencing project, and a second that embedded that partial sequence in a random sequence with the same local amino acid composition as one of the other query sequences (see Materials and methods). The embedded partial query sequences have only local sequence similarity to the entries in the PIR1 library and should provide a facsimile of sequences produced by exon-shuffling events. Because the flanking sequences have strong compositional similarity to other large families in the database, searches with this second library are much more challenging. The difference is quite dramatic; with the complete query sequence data set of 134 sequences, only 528 library sequences fall below the equivalence point with the best search parameters (BLOSUM45, −12, −1, ln()-scaled scores). In contrast, when embedded-partial sequences are used with ln()-scaled scores and the best search parameters (BLOSUM50, −12, −2) misses almost three times as many library sequences (1,452, $z = 6.2$). With partial but not embedded sequences, the best performing matrix and gap penalties miss 1,117 sequences ($z = 5.4$).

When embedded-partial query sequences are used, the best Smith–Waterman performance is obtained with higher gap penalties than seen with full-length query sequences; for unscaled scores, gap penalties of −14, −4 to −16, −2 are optimal with the BLOSUM50 scoring matrix (Fig. 6A). When either regression scaling or ln()-scaling is used, changes in gap penalties have a less dramatic effect on performance (Fig. 6B,C). This is consistent with the hypothesis that ln()-scaling and regression-scaling reduce the scores of unrelated sequences; several partial embedded query sequences have very high scores with "unrelated" library sequences because of compositional similarity in the flanking sequence. BLASTP searches, even using "sum" statistics, do not perform as well as ln()-scaled Smith–Waterman

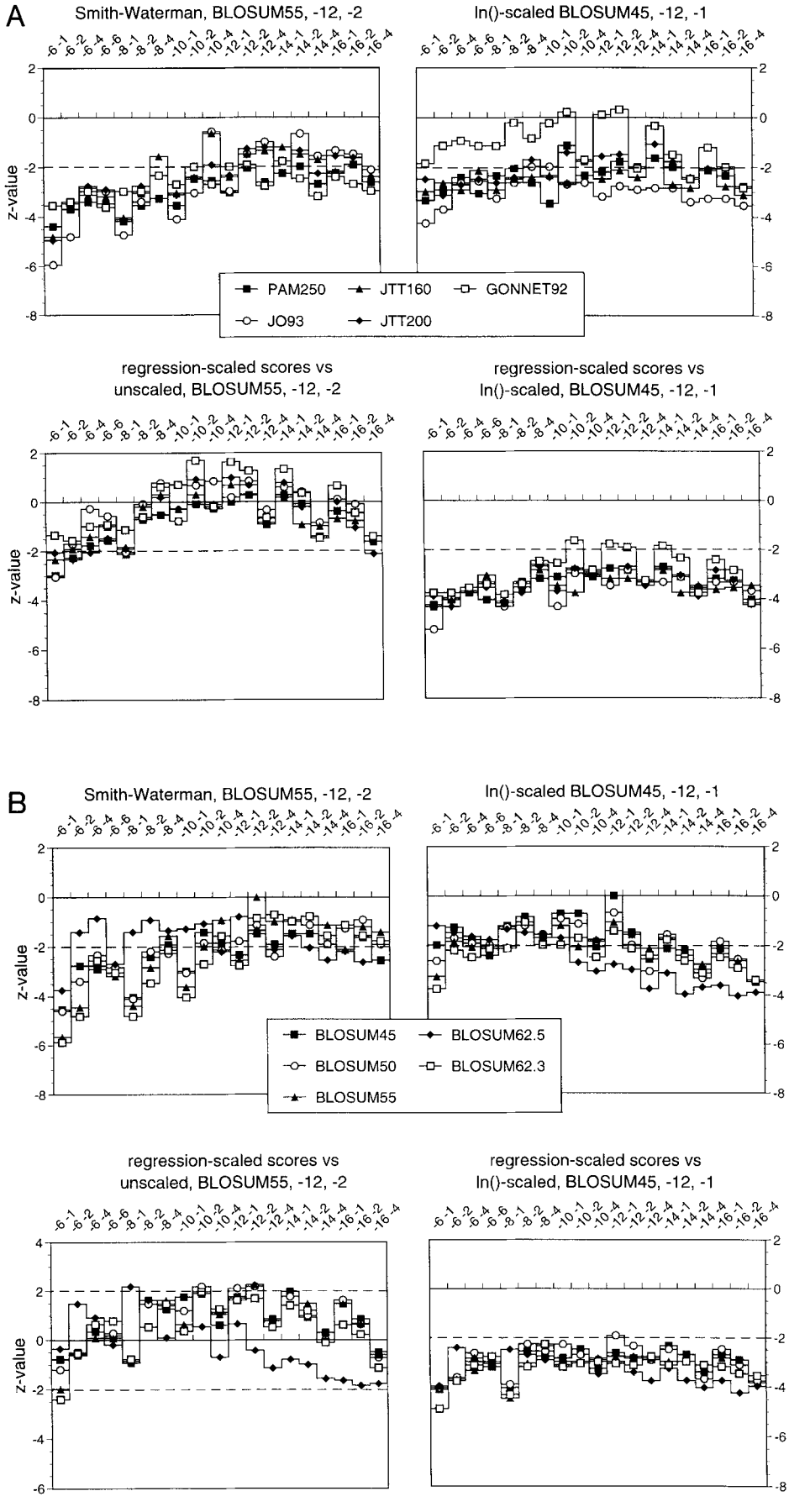

Fig. 4. Scoring matrices and gap penalties. Performance of (**A**) PAM250, JO93 (Johnson & Overington, 1993), Gonnet92 (Gonnet et al., 1992), and JTT160–200 (Jones et al., 1992), and (**B**) BLOSUM45–62 (Henikoff & Henikoff, 1992) are compared using gap penalties from −6, −1 to −16, −4. In each set (A, B) of four graphs, those on the left compare the indicated matrix and gap penalty combination with BLOSUM55, −12, −2 using unscaled Smith–Waterman scores and those on the right compare the indicated matrix and gap penalty with BLOSUM45, −12, −1 using ln()-scaled Smith–Waterman scores. For each set (A, B) of four graphs, the bottom two compare the performance of regression-scaled similarity scores to either unscaled (left) or ln()-scaled (right) scores. Performance with two different BLOSUM62 matrices is shown; BLOSUM62.5 is the standard BLOSUM62 matrix used with BLASTP with entries scaled in 1/2 bits of information per residue (Altschul, 1991). The BLOSUM62.3 matrix is scaled in 1/3-bit units, as are the other three BLOSUM matrices.

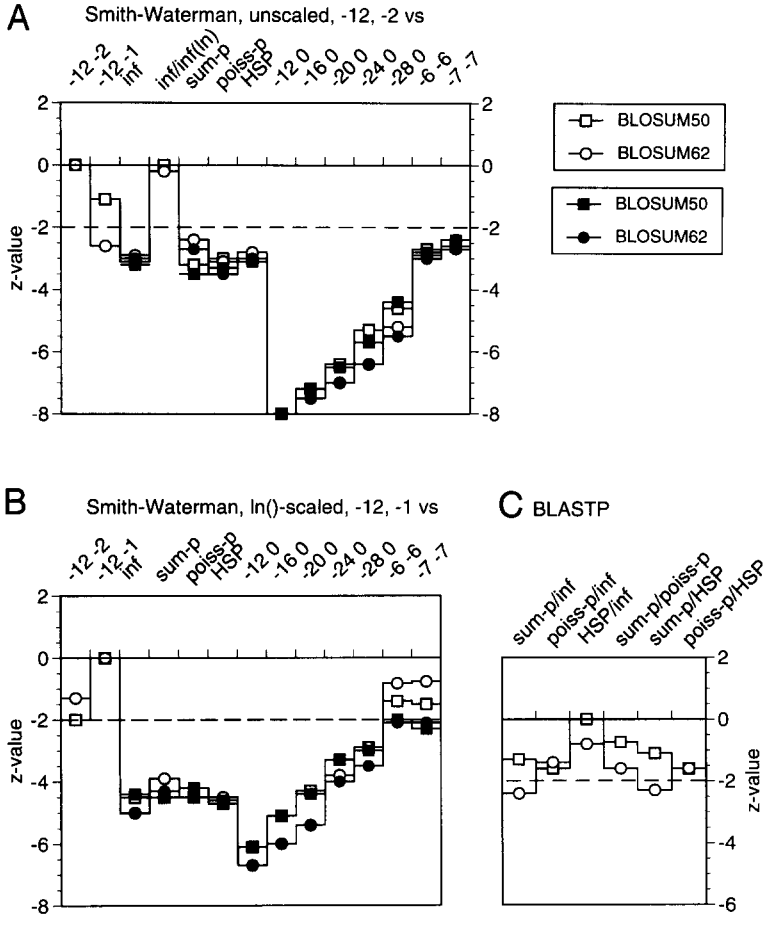

Fig. 5. Other gap penalties. Performance of BLOSUM50 and BLOSUM62.5 matrices with gap penalties of (**A**) −12, −2 (unscaled Smith-Waterman) or (**B**) −12, −1 (ln()-scaled Smith-Waterman) is compared with gap penalties of −∞, −∞ (inf); −12, 0; −16, 0; −20, 0; −24, 0; −28, 0; −7, −7; and −6, −6. For the open symbols, the same matrix is used for each series of comparisons; thus, the performance of BLOSUM50 with gap penalties of −7, −7 is compared to BLOSUM50 with −12, −2 (A) or −12, −1 (B), and BLOSUM62 gap penalties are compared with BLOSUM62.3, −12, −2 (A) or −12, −1 (B). (BLASTP searches used the default BLOSUM62.5 scoring matrix.) Filled symbols report performance with respect to Smith-Waterman BLOSUM55 unscaled (A) or BLOSUM45, ln()-scaled (B). Also included is a comparison of unscaled and ln()-scaled (inf/inf(ln)) searches (A) and a comparison of ln()-scaled Smith-Waterman to BLASTP sum-*P*, Poisson-*P*, and HSP scores (A, B). **C:** Searches using BLASTP sum-*P*, Poisson-*P*, and HSP scores are compared with Smith-Waterman, −∞, −∞, and searches using the other two BLASTP scores. When two search algorithms are shown on the column labels, e.g., inf/inf(ln) (A) or sum-*P*/inf (C), a negative *z*-value indicates that the second method performed worse than the first. Thus, BLASTP with HSP scores performed significantly worse than BLASTP sum statistics (sum-*P*/HSP) when BLOSUM62 was used.

scores. For unscaled and regression-scaled scores, the difference in performance is not significant when "HSP" scores and the BLOSUM62 matrix is used. There is no significant difference in the performance of ln()-scaled or regression-scaled Smith-Waterman scores when partial query sequences are used (Fig. 4D), but both regression-scaled and ln()-scaled Smith-Waterman scores perform significantly better than unscaled scores (data not shown).

Searching with FASTA

We compared the performance of FASTA with *ktup* = 1 and optimized scores to Smith-Waterman with optimal scoring matrices and gap penalties using either full-length or partial query sequences (Fig. 7). When unscaled scores are used to rank the sequences, FASTA performs as well as the best Smith-Waterman searches when the JO93, JTT160, or several BLOSUM matrices are used. When ln()-scaled scores are used for the searches, FASTA performs significantly worse than Smith-Waterman for most matrices. However, searches with the Gonnet92 matrix and ln()-scaled scores performed as well with FASTA at several gap penalties as with Smith-Waterman (BLOSUM45 and −12, −1).

FASTA with optimized scores and optimal matrices and gap penalties performed significantly better than BLASTP. With *ktup* = 1, unscaled optimized scores, and the JO93 matrix (−12, −2 or −14, −2), $z = 2.2$ was found for BLASTP "sum-*P*" scores and BLOSUM62; "Poisson-*P*" and HSP scores performed worse. With FASTA using BLOSUM50, −12, −2, *ktup* = 1, and optimized scores, BLASTP "sum-*P*" scores had $z = 2.6$; with ln()-scaled optimized FASTA scores, $z = 3.7$.

Surprisingly, FASTA searches with *ktup* = 2, JO93, −12, −2 or −14, −2, and unscaled optimized scores performed as well as Smith-Waterman with BLOSUM55 ($z = 1.3$, $P < 0.2$) while running 12 times faster. The best performance with FASTA required the calculation of optimized scores (a limited Smith-Waterman optimization through a 32-residue-wide band centered on the best initial region without gaps); performance using the *initn* or *init1* scores was significantly worse. For the data shown here, optimized scores were calculated for about one-half of the sequences in the database; when optimized scores were calculated for every sequence, performance did not improve significantly.

When embedded partial query sequences are used, FASTA performs as well as Smith-Waterman with both unscaled and ln()-scaled scores are used (Fig. 7C,D). When unscaled optimized FASTA scores are used, BLOSUM62.5 is one of the best matrices over a wide range of gap penalties. Henikoff and Henikoff (1993) reported the same result for a different set of test sequences. When ln()-scaled scores are used with partial sequences, many BLOSUM matrices perform well; BLOSUM50, −12, −2, is the best performer.

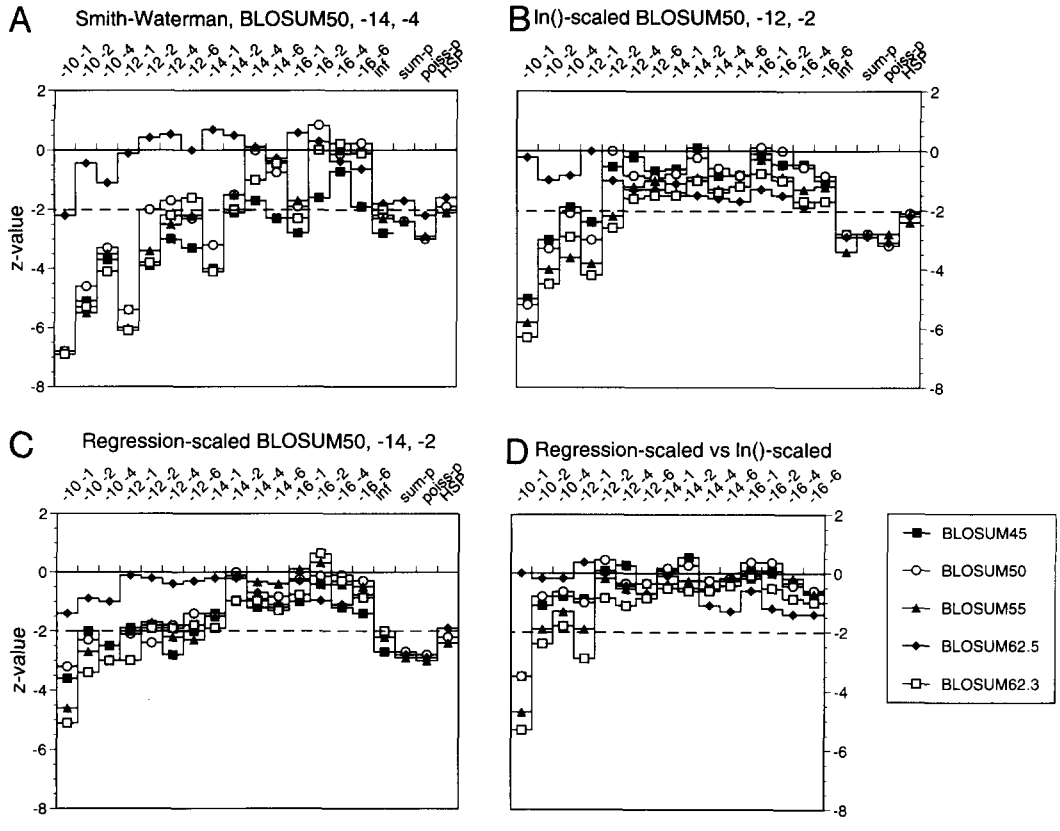

Fig. 6. Searches with partial sequences. A random portion (~70 residues) of each query sequence was embedded in the middle of ~100 residues of randomly shuffled sequence and used to search the database. Comparisons with BLOSUM50 and the indicated gap penalties are shown. **A:** Results with unscaled scores; comparisons to Smith-Waterman, BLOSUM50, −14, −4. **B:** Ln()-scaled scores; comparisons to Smith-Waterman, BLOSUM50, −12, −2. **C:** Regression-scaled scores; comparison to BLOSUM50, −14, −2. **D:** Comparison of performance with regression scaled scores to the performance of BLOSUM50, −12, −2 with ln()-scaled scores. Comparisons to Smith-Waterman with infinite gap penalties (inf) and BLASTP using sum-*P*, Poisson-*P*, or HSP scores are included in A, B, and C.

Discussion

A quantitative test for statistically significant differences in the performance of sequence comparison algorithms and scoring matrices has been developed. Using a new criterion for search performance, the equivalence number, we have shown that the performance of the sequence similarity searches can be improved dramatically if modern scoring matrices, optimized gap penalties, and normalized similarity scores are used. Using optimal algorithms and gap penalties can push back the evolutionary horizon by hundreds of millions of years. When the Gonnet92 matrix (−12, −1) is used with the Smith-Waterman algorithm and ln()-scaling to search the PIR1 database for sequences related to human α-globin, only two sequences (extracellular globins from a polychaete worm) are missed. In contrast, when the conventional PAM250 matrix with gap penalties of −12, −4 and unscaled scores are used, 16 sequences, including several leghemoglobins, bacterial globin, and several insect globins, are missed. For all 134 query sequences, the best methods typically could identify almost 300 distantly related sequences that were missed with conventional PAM250 searches (e.g., BLOSUM45, −12, −1, ln()-scaled scores missed 528; PAM250, −12, −4, unscaled missed 814).

The performance test used in this report—the ability to identify distantly related members from many diverse protein sequence families—is different from those used in several recent comparisons of amino acid replacement matrices. Johnson and Overington (1993) examined the ability of 12 scoring matrices to calculate statistically significant similarity scores and to reconstruct correct pairwise alignments for sequences whose structures are known; their results indicate that their matrix (JO93), which is derived from transition frequencies between amino acids in aligned structures, performs about as well as the BLOSUM and Gonnet92 matrices in most tests and better than the PAM250 matrix. A similar conclusion can be drawn from Figure 4A. Vingron and Waterman (1994) also used an alignment-based criterion to suggest that the PAM250 matrix with a modest positive offset performed better; our results show that unmodified matrices perform best for identifying distantly related sequences.

Our results are similar to those obtained by Henikoff and Henikoff (1993) for the BLOSUM matrices and FASTA and BLASTP. They used a similar test but a different database of related sequences—those derived from the PROSITE/Swiss-Prot databases (Bairoch, 1991; Bairoch & Boechmann, 1991). Thus, it is unlikely that our results reflect any special biases of

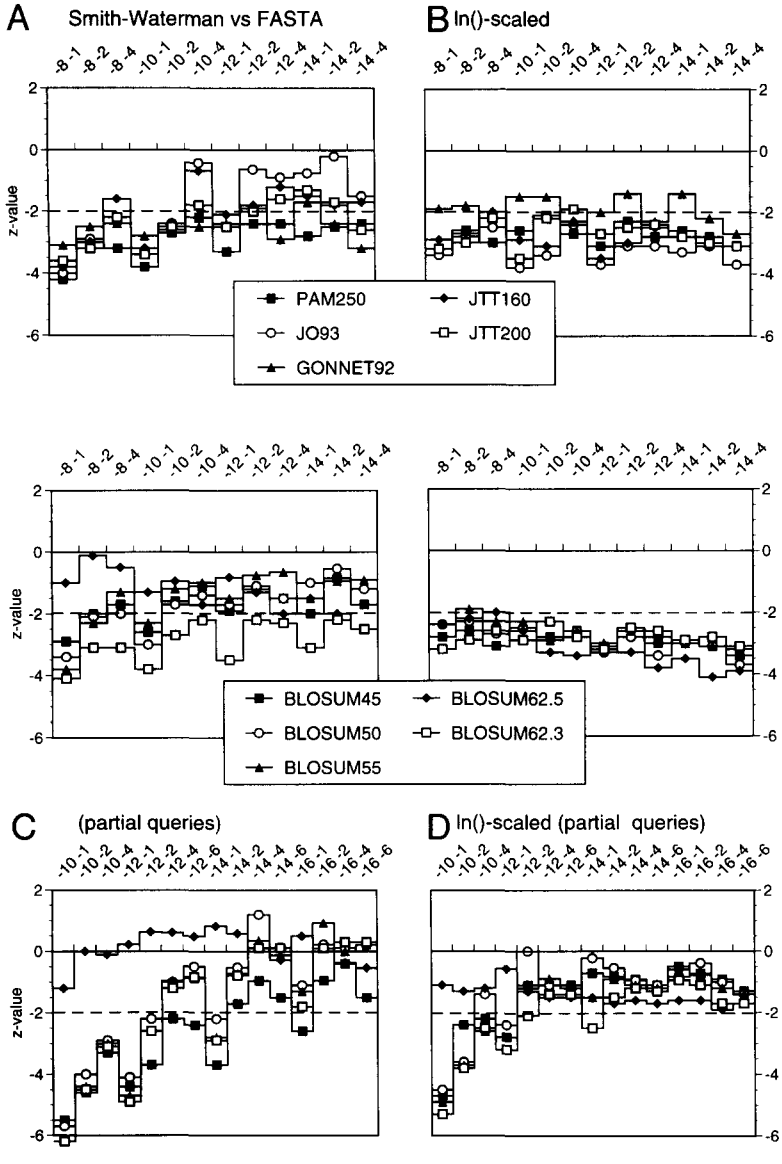

Fig. 7. Searches with FASTA. The same families of matrices shown in Figures 4 and 6 were used to search the protein database using the FASTA program with *ktup* = 1 and optimized scores. **A:** Comparison of FASTA scores calculated with the indicated matrix and gap penalty combination with unscaled Smith-Waterman scores, BLOSUM55, −12, −2. **B:** Comparison of FASTA scores calculated with the indicated matrix and gap penalty with ln()-scaled Smith-Waterman scores, BLOSUM45, −12, −1. **C:** Comparison of unscaled FASTA scores (*ktup* = 1, optimized) with unscaled Smith-Waterman (BLOSUM50, −14, −4) scores using partial query sequences and the BLOSUM45-62 scoring matrices. **D:** Comparison of ln()-scaled FASTA with ln()-scaled Smith-Waterman (BLOSUM50, −12, −2) using partial query sequences.

the PIR1 superfamily classification. Several families from the our data set (globins, glutathione transferases, α-crystallins) are missing from the PROSITE/Swiss-Prot library, whereas PROSITE data set has many more families (the Henikoffs used 257 "challenging" groups).

This study suggests that, in contrast to the observations of Henikoff and Henikoff (1993), modern extrapolated scoring matrices can perform as well as those based on direct sequence alignment when gap penalties are optimized. Thus, although BLOSUM55 is the best matrix for unscaled Smith-Waterman comparisons and performs significantly better than any matrix based on the original PAM data, JTT160 and 200, which are extrapolated from modern PAM replacement data, perform as well as BLOSUM55. For ln()-scaled comparisons, Gonnet92 is the equal of the best BLOSUM matrices, and JTT200 and 250 perform almost as well with optimal gap penalties. Gonnet92 is effectively a PAM116.5 matrix extrapolated from alignments ranging over distances from PAM6.4 to PAM100.0 (Gonnet et al., 1992). Figures 4 and 6 show that an appropriate gap penalty is essential for the best performance of a scoring matrix. The widely used −12, −4 gap penalty is often not the best when full-length query sequences are used; penalties of −12, −2 work significantly better for several of the BLOSUM matrices with unscaled scores. For ln()-scaled scores, lower gap penalties (−12, −1 or −10, −2) frequently work better than the best penalties for the corresponding unscaled search.

More importantly, the most distant sequence relationships are revealed when the underlying alignments contain gaps and ln()-scaled scores are used. Thus, even though it is more difficult to estimate accurately the probability of a match if it contains gaps (particularly if the combination of scoring matrix and gap penalties does guarantee local alignments on unrelated sequences), alignments with gaps and ln()-scaled scores are significantly more effective in identifying distantly related sequences. The exceptional performance of the Gonnet matrix and ln()-scaled scores with the globin family (two sequences missed) compares

very favorably with the best results found with profile searches (Gribskov et al., 1987) and searches based on flexible patterns (Barton & Sternberg, 1990).

It is unclear why ln()-scaling provides such a dramatic improvement over unscaled Smith-Waterman scores. The statistical properties of similarity scores that include gaps are poorly understood. Waterman (Waterman et al., 1987; Vingron & Waterman, 1994) has shown that the expected score for unrelated sequences can shift from an $\ln(n_l)$ to an n_l relationship as the scoring and gap parameters change. Mott (1992) has shown that for common choices of scoring matrices and gap parameters, the alignments can shift from local to global for unrelated sequences, with the result that the variance of the similarity score becomes dependent on n_l. P. Green has shown that ln()-scaling partially corrects for this dependence; however, the regression-scaled scores, which more accurately correct the variance, do not perform as well in general as ln()-scaled scores for full-length query sequences (Fig. 4) and do not perform significantly better for purely local searches with partial query sequences (Fig. 6). Further investigation into the statistics of similarity scores calculated with the best scoring matrices and gap penalties—whether the scores are local, partially local, or global—is clearly required.

These studies also provide a tool to investigate empirically the relationship between scoring matrices and optimal gap penalties. The relationship between the "information content" of different scoring matrices and their abilities to identify all the members of a protein family (Collins et al., 1988; Altschul, 1991, 1993) seems less important when gap penalties and scaled similarity scores are used. For example, the BLOSUM45−62.3 series of matrices can all perform very well (Figs. 4B, 6), yet they differ in relative entropy (Altschul, 1991) almost twofold, from 0.70 (BLOSUM62) to 0.38 (BLOSUM45) bits per residue pair. By decreasing the number of high-scoring unrelated sequences, ln()- and regression scaling may reduce problems encountered with a "too deep" scoring matrix. (The average length of the homologous region in our partial query sequences, 71 ± 23, was short enough that the BLOSUM45 matrix should have had difficulty detecting homologues, yet it was among the most effective when ln()-scaling was used.) Conversely, matrices that are optimal at very short evolutionary distances (e.g., PAM40, Altschul, 1993) are best suited to detecting homologous short (<30 residue) partial sequences in database searches; average-length protein sequences at short distances have high similarity scores that are easily distinguished from unrelated sequences. By increasing the scores of partial library sequences, ln()-scaling can improve the scores of homologous partial sequences in the databases. (Unfortunately, ln()-scaling can be very distracting when very short [2–4-residue] sequences are found in raw translations of DNA sequence databases; the correction is inappropriate for such sequences.)

Clearly there is a trade-off between search computation time and search performance. On the DEC Alpha workstations used in this study, the FASTA program with optimized scores and $ktup = 1$ was about six times faster than the Smith-Waterman program for all 134 sequences. In searches with a 217-residue query sequence, searching the 12,219-entry PIR1 database took about 220 s with Smith-Waterman; 51 s with FASTA, $ktup = 1$, and optimized scores, 26 s with $ktup = 2$, and optimized scores; and 14 s with BLASTP. For full-length query sequences, searches should be performed with the slower Smith-Waterman algorithm, but when partial sequences are examined, the much faster but equally effective FASTA program (with $ktup = 1$ and optimized scores) can be used. Our results suggest that, even with its recent improvements, BLASTP is not as effective as FASTA or Smith-Waterman if the latter algorithms are used with optimal scoring matrices, gap penalties, and ln()-scaled scores. However, for FASTA to be effective, optimized scores must be calculated for most of the sequences in the database. This calculation slows FASTA twofold or less for most searches, while dramatically improving performance.

We are currently exploring rapid methods for performing searches with gaps that are as sensitive as ln()-scaled Smith-Waterman. The BLASTP algorithm is an excellent starting place, because BLASTP with BLOSUM62 performs as well as Smith-Waterman with PAM250 (although not as well as Smith-Waterman with BLOSUM55, −12, −2). Our comparison of Smith-Waterman with infinite gap penalties suggests that the heuristic portion of BLASTP does not miss any significant alignments. In addition, BLASTP (with an effective $ktup = 3$) performs better than unscaled FASTA with $ktup = 1$ and no gaps; thus, the ability to examine conservative replacements at the earliest stage, as is done with BLASTP and Smith-Waterman, clearly provides a significant advantage. However, the segment-based gapping strategy of BLASTP does not perform as well as the gapped alignments computed by Smith-Waterman, even with unscaled scores. A gapped version of the BLASTP program is the best candidate for a program that is faster than, but as effective as, ln()-scaled Smith-Waterman for identifying distant relationships.

Materials and methods

Sequence libraries

All searches were performed on the annotated portion (PIR1) of the National Biomedical Research Foundation protein sequence database (Barker et al., 1990; release 39, December 31, 1993, 4,306,189 amino acid residues in 11,982 sequences). The PIR1 library is annotated with a superfamily assignment for every sequence in the library. The superfamily assignment is usually based on sequence similarity but can also reflect structural similarity or other common features if a clear case for common evolutionary ancestry (Barker et al., 1990) can be made. Sometimes a superfamily will contain several members whose common ancestry cannot be demonstrated by pairwise sequence similarity (Pearson, 1991). The PIR1 library contains only about 1/3 of the sequences in the entire NBRF-PIR protein sequence database; the other two databases (PIR2 and PIR3) do not contain complete superfamily annotations and thus cannot be used to demonstrate relatedness (common ancestry). The PIR1 library was augmented with 139 members of the G-protein-coupled receptor family and 98 glutathione transferase sequences, to make a larger library of 4,386,084 residues in 12,219 sequences.

There are 94 protein sequence superfamilies with 20 or more members in release 39 of the PIR1 sequence database. Viral polyproteins, which may contain members of several protein families, were removed from the set of query sequences. With inclusion of additional G-protein-coupled receptors and glutathione transferases, 90 families were found with more than 20 members. However, the members of 23 of these families are so closely related that none of the programs tested misses any of

the members of the family; these families were removed from the set of query sequences used because they cannot be used to discriminate between different search algorithms or scoring parameters. For the remaining 67 families, two sets of query sequences were used in these studies. The first set was built by taking the first member of each superfamily in the augmented PIR1 database. A second set was built by shuffling the protein sequence database and selecting the first member of each family found in the shuffled list. The query sequences used for these studies are listed in Table 1.

Tests were also run with partial query sequences that were embedded in longer random sequences. For each sequence, two random numbers, k and s, were generated: k was used to set the length of the partial sequence and was drawn from a normal distribution with $\mu = 100$ and $\sigma = 25$; s was then selected from uniformly distributed numbers between 0 and $n - 1$, where n was the length of the sequence being sampled. The partial sequence was then taken from residues $s + 1$ to $s + k$ of the original sequence. The original query sequences ranged in length from 16 to 1,107 (mean 230 ± 186); the partial sequences ranged from 16 to 127 (mean 71 ± 23). Each partial query sequence was embedded in two equal length random sequences. The random sequences were generated by concatenating all the residues from the 67 query sequences, shuffling this 20,000-residue sequence using a local window of 10 residues, and then generating a length for the flanking sequences from a normal distribution with $\mu = 200$, $\sigma = 100$. A sequence of this length was drawn from the shuffled query sequences; half of this random sequence was placed in front of the partial query sequence and half was placed after it. These embedded partial query sequences ranged from 89 to 462 residues (mean 284 ± 69).

Similarity searching

Searches with the FASTA (Pearson & Lipman, 1988; Pearson, 1990) and Smith–Waterman (Smith & Waterman, 1981; Pearson & Miller, 1992) algorithms were performed in parallel on a network of five DEC Alpha AXP3000 model 300 workstations using a general platform for sequence comparison on parallel computers (Despande et al., 1991) that was converted to the PVM parallel programming environment. The platform was modified to calculate the performance of the algorithm for each query sequence from within the parallel program. A derivative of version 1.6 of the FASTA program (Chao et al., 1992) was used; the implementation of the Smith–Waterman algorithm was described in Pearson and Miller (1992). The platform was extended to include a global similarity score calculated by a modification of the Needleman–Wunsch algorithm (Myers & Miller, 1988). Version 1.4.7 of the BLASTP program (October 1994; Altschul et al., 1990), which incorporates the new "sum" statistics (Karlin & Altschul, 1993), was used for BLASTP searches. BLASTP was run with the default options, except for E (2,000), hspmax (500), and V (2,000). The "E" parameter was increased to display as many HSP scores as possible. BLASTP P-values were converted to integer values similar to traditional similarity scores by the formula: $P\text{-score} = -10.0 \ln(P\text{-value})$. Equivalence numbers were calculated using either the best single HSP score, the sum-statistics ("sum-P") P-score, or the older Poisson statistics ("Poisson-P") P-score. Because the similarity score at the equivalence point is sometimes zero for BLASTP searches, the equivalence number calculation was modified slightly for BLASTP. If the equivalence point score was greater than zero, then the equivalence number was calculated in the standard way (the number of related sequences obtaining scores less than the equivalence point score). If the equivalence point score was zero, then the equivalence is the number of related sequences with a score of zero.

Different methods for normalizing local similarity scores to correct for differences between the length of the query sequence and the library sequences were examined. Raw Smith–Waterman and FASTA similarity scores were scaled by $\ln(n_q)/\ln(n_l)$, where n_q and n_l are the lengths of the query and library sequences, respectively. In addition, scaling by $(n_q/n_l)^{1/2}$ or n_q/n_l was tested.

Similarity scores were also scaled to remove the dependence of similarity scores on sequence length using a routine provided by Dr. P. Green (University of Washington). Raw similarity scores s were converted to z-scores: $z = (s - \rho \ln(n_l) - \mu)/(var)^{1/2}$. ρ and μ were determined from a fit of the line $s(n_l) = \mu + \rho \ln n_l$ and var is estimated from regressing the squared residuals $[s(n_l) - \mu - \rho \ln(n_l)]^2$ on $\ln(n_l)$. The estimation process was repeated after excluding similarity scores with $z < -5$ or $z > 5$. The z-scores calculated in this way had a mean = 0 and standard deviation = 1 for n_l from 20 to the maximum library sequence length. The z-scores were then converted back into conventional (positive) similarity scores with the relationship $s_z = 50 + 10z$. s_z are referred to as regression-scaled scores.

Scoring matrices

The PCOMPLIB programs were modified to accept the substitution matrices distributed with BLASTP (1.3.11). The PAM series (Dayhoff et al., 1978) of matrices was calculated using the PAM program distributed with this release of BLASTP, and the BLOSUM and Gonnet92 matrices were obtained from the same source. The code to generate the JTT (Jones et al., 1992) series of matrices was obtained from the authors. The JO93 (Johnson & Overington, 1993) matrix was entered by hand.

Statistical analysis

The sign-test was used to compare the performance of different algorithms and search parameters (Table 3). The equivalence number was calculated for each of the 134 query sequences with one algorithm of interest, and then this list of equivalence numbers was compared to the list generated by a second search algorithm. Differences in performance for each query sequence were scored with a "+" if the second algorithm had a lower equivalence number than the first algorithm, a "−" if the first algorithm performed better, and 0 if the two equivalence numbers were equal. The magnitude of the difference in equivalence numbers was ignored, because the observations of similarity scores are not independent within a superfamily. Thus, if an algorithm cannot detect the similarity between a human α-globin sequence and a soybean leghemoglobin sequence, it is likely that it will not detect the similarity with other leghemoglobin sequences that are closely related to the soybean sequence. If the magnitudes of the difference were considered, then superfamilies with several clusters of very closely related sequences might outweigh the results from families with more uniformly diverged sequences. By using the sign-test, errors caused by selective sampling of members of protein sequence families are reduced. A

similar approach was taken by Henikoff and Henikoff (1993), where the number of families where performance was better or worse was plotted.

Probabilities for the sign-test were calculated from a normal approximation to the binomial distribution for cases where the sum of the positive (Np) and negative (Nm) differences was 10 or greater. The z-value is calculated from the formula: $z = [\max(Np, Nm) - \mu]/\sigma$, where $\mu = (Np + Nm)P$, $\sigma = [(Np + Nm)P(1 - P)]^{1/2}$, and $P = 0.5$. Where two query sequences from each superfamily were examined (Table 4 and all figures), z-values were divided by $2^{1/2}$ because the two observations of the superfamily are not independent. P-values are reported for the more conservative two-tailed test, i.e., we test whether two strategies perform differently, not whether one performs better than the other. Both the $2^{1/2}$ correction and the use of a two-tailed test tend to underestimate the statistical significance of differences in performance.

Acknowledgments

I thank Phil Green for very helpful discussions and the code to calculate the regression-scaled scores. The "equivalence number" was suggested by Dana Richards. The use of the sign test was suggested by Joe Felsenstein. I also thank a very conscientious reviewer who provided many helpful suggestions and corrections. This work was supported by a grant from the National Library of Medicine (LM04969) with additional support from the Digital Equipment Corporation.

References

Altschul SF. 1991. Amino acid substitution matrices from an information theoretic perspective. *J Mol Biol 219*:555-565.

Altschul SF. 1993. A protein alignment scoring system sensitive at all evolutionary distances. *J Mol Evol 36*:290-300.

Altschul SF, Gish W, Miller W, Myers EW, Lipman DJ. 1990. A basic local alignment search tool. *J Mol Biol 215*:403-410.

Arratia R, Gordon L, Waterman MS. 1986. An extreme value theory for sequence matching. *Ann Stat 14*:971-993.

Bairoch A. 1991. PROSITE: A dictionary of sites and patterns in proteins. *Nucleic Acids Res 19*(Suppl):2241-2245.

Bairoch A, Boechmann B. 1991. The SWISS-PROT protein sequence data bank. *Nucleic Acids Res 19*(Suppl):2247-2249.

Barker WC, George DG, Hunt LT. 1990. Protein sequence database. *Methods Enzymol 183*:31-49.

Barton GJ, Sternberg MJE. 1990. Flexible protein sequence patterns. A sensitive method to detect weak structural similarities. *J Mol Biol 212*: 389-402.

Chao KM, Pearson WR, Miller W. 1992. Aligning two sequences within a specified diagonal band. *Comput Appl Biosci 8*:481-487.

Collins JF, Coulson AFW, Lyall A. 1988. The significance of protein sequence similarities. *Comput Appl Biosci 4*:67-71.

Dayhoff M, Schwartz RM, Orcutt BC. 1978. A model of evolutionary change in proteins. In: Dayhoff M, ed. *Atlas of protein sequence and structure, vol 5 (suppl 3)*. Silver Spring, Maryland: National Biomedical Research Foundation. pp 345-352.

Despande AS, Richards DS, Pearson WR. 1991. A platform for biological sequence comparison on parallel computers. *Comput Appl Biosci 7*: 237-247.

Gonnet GH, Cohen MA, Benner SA. 1992. Exhaustive matching of the entire protein sequence database. *Science 256*:1443-1445.

Gribskov M, McLachlan AD, Eisenberg D. 1987. Profile analysis: Detection of distantly related proteins. *Proc Natl Acad Sci USA 84*:4355-4358.

Henikoff S, Henikoff JG. 1992. Amino acid substitution matrices from protein blocks. *Proc Natl Acad Sci USA 89*:10915-10919.

Henikoff S, Henikoff JG. 1993. Performance evaluation of amino acid substitution matrices. *Proteins Struct Funct Genet 17*:49-61.

Johnson MS, Overington JP. 1993. A structural basis for sequence comparisons. An evaluation of scoring methodologies. *J Mol Biol 233*:716-738.

Jones DT, Taylor WR, Thornton JM. 1992. The rapid generation of mutation data matrices from protein sequences. *Comput Appl Biosci 8*: 275-282.

Karlin S, Altschul SF. 1990. Methods for assessing the statistical significance of molecular sequence features by using general scoring schemes. *Proc Natl Acad Sci USA 87*:2264-2268.

Karlin S, Altschul SF. 1993. Applications and statistics for multiple high-scoring segments in molecular sequences. *Proc Natl Acad Sci USA 90*: 5873-5877.

Mott R. 1992. Maximum-likelihood estimation of the statistical distribution of Smith-Waterman local sequence similarity scores. *Bull Math Biol 54*:59-75.

Myers EW, Miller W. 1988. Optimal alignments in linear space. *Comput Appl Biosci 4*:11-17.

Pearson WR. 1990. Rapid and sensitive sequence comparison with FASTP and FASTA. *Methods Enzymol 183*:63-98.

Pearson WR. 1991. Searching protein sequence libraries: Comparison of the sensitivity and selectivity of the Smith-Waterman and FASTA algorithms. *Genomics 11*:635-650.

Pearson WR, Lipman DJ. 1988. Improved tools for biological sequence comparison. *Proc Natl Acad Sci USA 85*:2444-2448.

Pearson WR, Miller W. 1992. Dynamic programming algorithms for biological sequence comparison. *Methods Enzymol 210*:575-601.

Smith TF, Waterman MS. 1981. Identification of common molecular subsequences. *J Mol Biol 147*:195-197.

Sturrock SS, Collins JF. 1993. *MPsrch version 1.3*. Edinburgh: University of Edinburgh Biocomputing Research Unit.

Vingron M, Waterman M. 1994. Sequence alignment and penalty choice. *J Mol Biol 235*:1-12.

Waterman MS, Gordan L, Arratia R. 1987. Phase transitions in sequence matches and nucleic acid structure. *Proc Natl Acad Sci USA 84*:1239-1243.