*Mathl. Comput. Modelling* Vol. 16, No. 6/7, pp. 37–43, 1992
Printed in Great Britain. 

0895-7177/92 $5.00 + 0.00


# PROSET—A FAST PROCEDURE TO CREATE NON-REDUNDANT SETS OF PROTEIN SEQUENCES

VOLKER BRENDEL
Department of Mathematics
Stanford University, Stanford, CA 94305, U.S.A.

**Abstract**—The PROSET computer program described efficiently eliminates redundant entries from a set of proteins. It finds repeats between all the proteins of the original set, derives a block identity score between any two sequences that share at least one sufficiently long exact repeat, and discards the shorter of the two sequences if the score exceeds a user-defined threshold. The program finds application in generating control sets for statistical evaluation of sequence patterns. It may also serve to reduce the amount of redundancy in available databases.

A common problem in biomolecular sequence analysis is to find an appropriate control set of sequences against which a particular hypothesis concerning the occurrence of a certain pattern may be tested. For example, one would like to know how frequent motifs like zinc-fingers [1], the leucine zipper [2], or acidic blobs [3] are among a general class of proteins, say all mammalian proteins, or all mammalian nuclear proteins. Such information is necessary for prudent sequence interpretation. Initially, one might be tempted to use all matching entries of the standard databases (GenBank, EMBL, NBRF, SWISS-PROT) as a control set. However, such approach would give biased results due to the considerable redundancy in the databases. For the purposes described, one would not want to include multiple entries of closely related sequences in the control set.

I shall present a computer program that serves to generate non-redundant sets of protein sequences. The user specifies a list of protein files and a similarity criterion; the program then produces a new list which retains only the longer sequence of any two similar protein files. As a sample application, the list of 2,281 human, mouse and rat protein sequences of lengths 200 to 2500 residues from SWISS-PROT Release 17 is reduced to a list of 1,050 proteins, any two of which are less than about 25% identical. Several statistics are compiled for this set.

## PROGRAM DESCRIPTION

PROSET was written in the C-Programming Language and implemented under the UNIX operating system on a Sun-4/110 workstation.

The program is called by typing "proset *listn rmin imin* [*rmax*]", where *listn* is the name of a file containing the names of data files to be processed. Each data file consists of an arbitrary number of comment lines followed by a delimiter (. .) and a protein sequence in the standard one-letter code. *rmin* is an integer (typically 7) which gives the desired "core length" of the comparisons; any two sequences that contain at least one common *rmin*-mer will be subjected to a detailed pairwise comparison. *imin* is another integer (typically 25) setting the threshold "block identity" (defined below) for sequence elimination. If two sequences are pairwise compared and their block identity exceeds *imin*% then the smaller of the two sequences is scratched from the list. *rmax* is an optional argument (by default set equal to 50) that should be set to the expected maximal repeat size (if less than 50) in the data set.

This work was supported by NIH grant GM39907-01. I would like to thank Drs. P. Bucher, G. Schachtel, and S. Karlin for discussion.

PROSET is constructed such that the terminal command invokes a C-shell, which cyclically calls a number of programs. Initially, all proteins are identified that have a 50-mer (or, if specified, *rmax*-mer) in common. These proteins are compared in more detail, their block identity established, and if the block identity exceeds *imin*%, the shorter of the two sequences is marked for elimination from the list. This cycle is repeated for core lengths 40, 30, ..., down to *rmin* as specified by the user. The reason for this stepwise approach is to reduce the number of pairwise comparisons. Proteins that have a 50- or 40-mer in common are most likely to exceed the *imin*% block identity; thus the elimination of one of them quickly reduces the list.

*Finding Exact Repeats*

The PROSET algorithm to find exact repeats is essentially as described by Karlin *et al.* [4,5]. The initial step is the creation of an array for 4-mer repeat positions. The amino acids are given numbers 0 to 19, corresponding to their approximate order in decreasing abundance. Each 4-mer $wxyz$ is represented by the unique number $w * 8000 + x * 400 + y * 20 + z$, where $w$, $x$, $y$, $z$ are its amino acids in numerical representation. An array of length $20^4$ is used to store the position of the most recent occurrence of each successive 4-mer as the proteins are scanned one by one from N- to C-terminus. An array of size equal to the total length of the concatenated proteins is created; the $i^{\text{th}}$ element of this array gives the position of the most recent occurrence of the 4-mer starting in position $i$ (or 0 if it has not occurred previously). Thus this array links all common 4-mers, the last occurrence in the concatenated sequence pointing successively to all previous occurrences.

The array for 4-mers is then extended to give the same information for longer repeats. To get the array for 8-mers, the previous occurrence positions of 4-mer $i$ and 4-mer $(i-4)$ are screened until a pair $j$, $(j-4)$ is found; if no such pair is found then the 8-mer starting in position $(i-4)$ is unique. Similarly, the list could be extended by 1, say going from 8-mers to 9-mers. If an exact repeat cannot be further extended and exceeds the specified core length *rmin*, its positions are passed to a subroutine which will try to extend it with allowance for errors.

*Extending Exact Repeats*

Each exact repeat of length exceeding *rmin* is treated as the core of a block identity. A finite automata programming loop is used to extend this core to the left and right. The error criteria were set as follows: any mismatch, or gap of size one, has to be followed by two perfect matches, any two mismatches, or gaps of size two, have to be followed by three perfect matches. Any number of such error blocks is allowed, any disruption (e.g., three consecutive mismatches) terminates the identity block. Care is taken that any two exact repeats linked by an error block are only reported once.

*Block Identity*

For any two sequences that have at least one repeat exceeding *rmin* in common, a pairwise comparison is performed with the same algorithm described above, except that the core length is lowered to 5. Each of the two sequences is then represented by a string of dots, where each dot represents a hexamer (i.e., a sequence of 600 amino acids would be represented by a string of 100 dots). At the coordinates where the error-extended repeats occur, these dots are replaced by lowercase letters, "a" for the first repeat detected by the program, "b" for the next, etc., such that the later letters stand for the longer repeats. Should the number of distinct repeats exceed 26, then printing of repeats of the initial core size is suppressed and letters are re-assigned to longer repeats. Thus, each of the two sequences becomes a string of dots and lowercase letters, the dots delineating unique regions, and the letters delineating common sequences (Figure 1). The block identity of the two sequences is calculated as follows: first all dots are deleted (corresponding to no punishment for gaps); then the two strings of letters are compared position by position; the number of matching positions, divided by number of non-overlapping hexamers in the shorter sequence times 100 gives the percent block identity for this alignment. To make this score independent of domain shuffling, the eventual score is taken to be the maximum of the scores calculated as above for all circular permutations of one sequence against the other.

This definition of block identity is a rough measure for the similarity of two sequences, appropriate to the task of eliminating clearly related sequences (but not suitable to finding weak similarities). It gives more conservative scores than similarity measures obtained from optimal alignment programs.

```
**************************************************************************
Protein    1 (File: /ASQ/SPROT/IL2_MOUSE)
>SW;IL2_MOUSE: INTERLEUKIN-2 PRECURSOR (IL-2) (T-CELL GROWTH FACTOR) (TCGF).

number of residues:  169
       1  MYSMQLASCV TLTLVLLVNS APTSSSTSSS TAEAQQQQQQ QQQQQQHLEQ LLMDLQELLS
      61  RMENYRNLKL PRMLTFKFYL PKQATELKDL QCLEDELGPL RHVLDLTQSK SFQLEDAENF
     121  ISNIRVTVVK LKGSDNTFEC QFDDESATVV DFLRRWIAFC QSIISTSPQ

**************************************************************************
Protein    2 (File: /ASQ/SPROT/IL2_HUMAN)
>SW;IL2_HUMAN: INTERLEUKIN-2 PRECURSOR (IL-2) (T-CELL GROWTH FACTOR) (TCGF).

number of residues:  153

       1  MYRMQLLSCI ALSLALVTNS APTSSSTKKT QLQLEHLLLD LQMILNGINN YKNPKLTRML
      61  TFKFYMPKKA TELKHLQCLE EELKPLEEVL NLAQSKNFHL RPRDLISNIN VIVLELKGSE
     121  TTFMCEYADE TATIVEFLNR WITFCQSIIS TLT

**************************************************************************

Listed are all repeats containing at least one exact repeat of length >=  5

a   P#   2     23 -     27:  TSSST
 =  P#   1     27 -     31:  TSSST
-------------------------------------------------
b   P#   2     55 -     86:  KL t RMLTFKFY m PK k ATELK h LQCLE e EL k
 PL
 =  P#   1     69 -    100:  KL p RMLTFKFY l PK q ATELK d LQCLE d EL g
 PL
-------------------------------------------------
c   P#   2    137 -    151:  FL n RWI t FCQSIIST
 =  P#   1    152 -    166:  FL r RWI a FCQSIIST
-------------------------------------------------
d   P#   2     16 -     27:  LV t NSAPTSSST
 =  P#   1     17 -     27:  LV _ NSAPTSSST
-------------------------------------------------

Protein 1 (top) - Protein 2 (bottom) alignment:

...dd......bbbbb.........ccc

...dd....bbbbb.........ccc

/ASQ/SPROT/IL2_MOUSE ( 169) /ASQ/SPROT/IL2_HUMAN ( 153):  38 % block identity.
```

Figure 1. Sample output of a pairwise comparison based on identity blocks of length at least 5. Each identity block (extended with errors) is identified by a single letter, which are displayed with appropriate spacing as an alignment; note that the identity block labeled "a" is overwritten by the longer (and more appropriate) identity block "d."

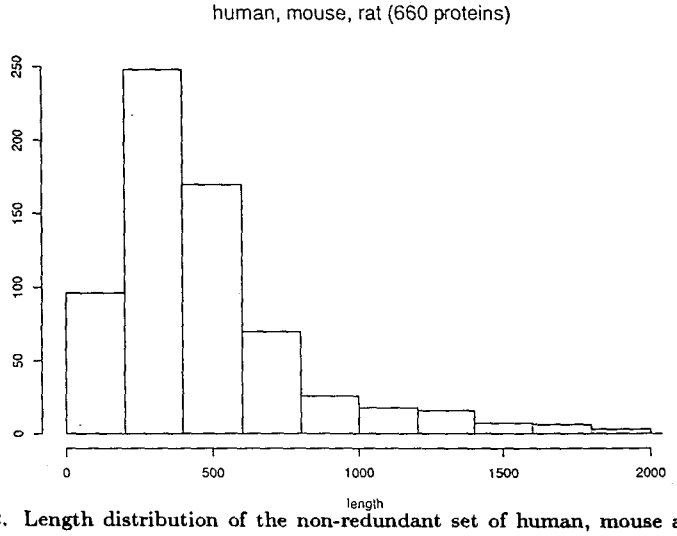


Figure 2. Length distribution of the non-redundant set of human, mouse and rat protein sequences (sequences less than 200 residues or longer than 2,500 residues being excluded).

Table 1. Average amino acid composition of the representative mammalian protein set. Displayed are the sample mean and standard deviation.

| amino acid | mean | std |
|---|---|---|
| L (leucine) | 9.4 | 2.8 |
| S (serine) | 7.5 | 2.5 |
| G (glycine) | 7.4 | 3.9 |
| A (alanine) | 7.0 | 2.5 |
| E (glutamate) | 6.6 | 2.5 |
| V (valine) | 6.3 | 1.9 |
| P (proline) | 6.1 | 3.5 |
| K (lysine) | 5.9 | 2.5 |
| T (threonine) | 5.5 | 2.0 |
| R (arginine) | 5.2 | 2.0 |
| D (aspartate) | 4.9 | 1.8 |
| I (isoleucine) | 4.6 | 1.9 |
| Q (glutamine) | 4.5 | 2.1 |
| F (phenylalanine) | 3.9 | 1.5 |
| N (asparagine) | 3.9 | 1.4 |
| Y (tyrosine) | 3.0 | 1.3 |
| H (histidine) | 2.4 | 1.3 |
| C (cysteine) | 2.3 | 1.7 |
| M (methionine) | 2.2 | 1.0 |
| W (tryptophane) | 1.4 | 0.9 |

## APPLICATION: A NON-REDUNDANT SET OF MAMMALIAN PROTEINS

Release 17 of the SWISS-PROT database (February 1991) contained 2,281 files with protein sequences derived from human, mouse, or rat cells of lengths between 200 and 2,500 residues (total length 1,126,544 residues). PROSET with *rmin* set to 7 and *imin* set to 25 reduced the original list of files to 1,050, roughly half of the original set. This striking redundancy results from many exact duplications (e.g., sequences entered by different authors), or very similar entries (different versions of a protein, homologous proteins from different sources, protein family members, etc.). The program ran close to the maximum of memory availability for about a couple of hours on a Sun-4/110.

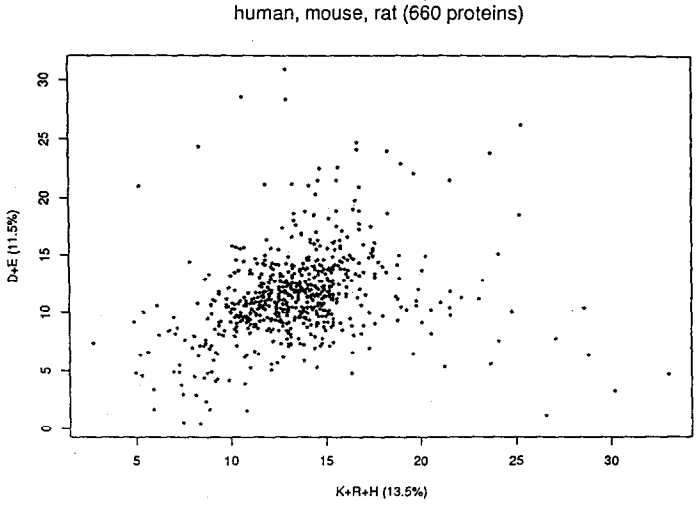


Figure 3. Negative versus positive charge content in the representative mammalian protein set. Proteins of high net negative charge are in the upper left corner; proteins of high total charge are in the upper right corner; proteins of high positive charge are in the lower right corner; and proteins of low total charge are in the lower left corner.

Table 2. Occurrence of charge clusters in the representative mammalian protein set. + indicates a positive charge cluster, − a negative charge cluster, and ± a mixed charge cluster (high in both basic and acidic residues). +/± and −/± indicate positive and negative charge clusters that also extend to mixed charge clusters, respectively. For example, 34 of the 1,050 proteins in the set have a single positive charge cluster.

| Protein with a single charge cluster | |
|---|---|
| + | 34 |
| − | 56 |
| ± | 89 |
| +/± | 10 |
| −/± | 8 |
| sum | 197 (18.8%) |
| **Proteins with multiple charge clusters** | |
| + + | 0 |
| − − | 4 |
| ± ± | 8 |
| + − | 4 |
| + ± | 4 |
| − ± | 1 |
| others | 4 |
| sum | 25 (2.4%) |

*Amino Acid Composition*

The PROSET generated set of non-redundant human, mouse and rat proteins gives a fair representation of mammalian proteins sequenced to date. Average characteristics can now be calculated based on this set. I shall give but a few examples. Figure 2 gives a histogram of the length distribution; the median length is 433 residues (note, though, that proteins shorter than 200 residues or longer than 2,500 residues were excluded to start with). Table 1 shows the average amino acid composition, and Figure 3 gives the plot of negative versus positive charge content for each protein.

Table 3. Occurrence of long runs of charged residues in the representative mammalian protein set. The position and length of each run, and the number of errors (mismatches, insertions, deletions) are given in column 2. For each class, the about 5 longest charge runs in the set of proteins are displayed. The proteins are: GCF, GC-rich sequence DNA-binding factor; A1AB, α-1b adrenergic receptor; IF2B, translation initiation factor β-subunit; PDGA, platelet-derived growth factor A-chain precursor; NUCL, nucleolin; HMG1, high mobility group protein 1; B232, nucleolar phosphoprotein B23; CENB, major centromere autoantigen; HUBF, nucleolar transcription factor; RDP, RD protein; CRTC, calreticulin precursor; SPBP, prostatic spermine-binding protein precursor; RU17, U1 small nuclear ribonucleoprotein 70 Kd; and HS9A, heat shock protein 86 Kd.

| file | from-to: | length(errors) | sequence |
|---|---|---|---|
| Positive charge runs: | | | |
| GCF_HUMAN | 22- 30: | 9(0) | RKKKKKKKR |
| A1AB_RAT | 371-378: | 8(0) | RRRRRRRR |
| IF2B_HUMAN | 14- 21: | 8(0) | KKKKKKKK |
| | 79- 87: | 8(1) | KKKKKKTKK |
| PDGA_HUMAN | 201-207: | 7(0) | KKRKRKR |
| Negative charge runs: | | | |
| NUCL_MOUSE | 240-272: | 33(0) | EEEDDEEEDEDDEDEDDEEEDDEDDDEEEEEEE |
| | 189-214: | 25(1) | EDEEDDEDEDDEEDDDEEEEDDSEEE |
| | 142-167: | 24(2) | EDSDEDEDEEDEDDSDEDEDDEEEDE |
| HMG1_HUMAN | 185-214: | 30(0) | EEEEDEEDEEDEEEEEDEEDEDEEEDDDDE |
| B232_HUMAN | 161-185: | 24(1) | DEDDDDDDEEDDDEDDDDDDFDDEE |
| CENB_HUMAN | 510-533: | 23(1) | EEEDDEEEDDEDEDDDDDEEDGDE |
| HUBF_HUMAN | 678-698: | 21(0) | EEDDEEDEDDEDEDEEEEDDE |
| | 722-742: | 20(1) | DENEEDDEDEDDDEDDDEDED |
| Mixed charge runs: | | | |
| RDP_MOUSE | 196-247: | 52(0) | RDRDRDKERDRDRDRDRDRDRDKDKDRDRDRDRDKERDRDRDRDRDRERDRE |
| CRTC_RAT | 368-407: | 39(1) | KEEEEDKKRKEEEEAEDKEDEDDRDEDEDEEDEKEEDEED |
| SPBP_RAT | 192-230: | 35(4) | DDDDDDKEDDNEEDVDDERDDKDDDEEDDDNDKENDKDD |
| RU17_HUMAN | 408-433: | 25(1) | RDRDRDRERERRERSRERDKERERRR |
| | 538-557: | 19(1) | ERERRRDRDRDRDREHKR |
| HS9A_MOUSE | 231-254: | 22(2) | DDEAEEKEEKEEEKEKEEKESDDK |
| | 263-279: | 16(1) | DEEEEEKKDGDKKKKKK |

*Charge Clusters*

The occurrence of charge clusters among viral and cellular proteins has been studied in detail by Karlin *et al.* [6–10]. It was noted that charge clusters are often found in viral and cellular transactivators, as well as in nuclear oncogene products. Table 2 lists the numbers of proteins in the mammalian set with single and multiple charge clusters, confirming, in particular, that proteins with multiple charge clusters are rare (found in less than 3% of the sequences).

*Charge Runs*

Table 3 gives the longest contiguous charge runs found in the set of mammalian proteins. It is of note that the longest negative charge runs are much longer than the longest positive charge runs. Most of the proteins containing long negative or mixed charge runs are among the nuclear autoantigens targeted in patients suffering from systemic autoimmune diseases [11].

*Leucine Zipper Motif*

The leucine zipper motif $LX_6LX_6LX_6L$ is characteristic of a protein domain that facilitates dimerization [2]. Given that leucine is by far the most abundant amino acid (see Table 1), one may suspect that such a pattern is actually not particularly rare. Indeed, for a protein sequence of 500 residues and leucine content 10%, the probability of finding a leucine zipper motif by chance is about 4% [12]. Among the 1,050 mammalian proteins, there are 86 sequences (8.2%)

with a leucine zipper motif. Thus, caution in speculating a dimerization function for a leucine zipper motif in a sequence in the absence of additional evidence seems warranted (cf. [12]).

## SUMMARY

PROSET (available for the UNIX environment upon request) is an efficient program to generate non-redundant sets of protein sequences. It quickly identifies repeats shared between any two sequences, and eliminates the shorter sequence if the repeats take up a large proportion of its length. The speed of the program is facilitated by concentrating on sequences for pairwise comparison only if they share at least one sufficiently long exact repeat. This requirement is reasonable for the purpose of eliminating redundancies (for detecting weak similarities, programs like FASTA of Pearson and Lipman [13] are more suitable). Statistics derived on non-redundant data sets provide controls for statistical sequence interpretation. Thus, it is hoped that PROSET or similar programs will be employed widely to rigorize the evaluation of patterns in sequences.

## REFERENCES

1. J. Berg, Proposed structure for the zinc-binding domains from transcription factor IIIA and related proteins, *Proc. Natl. Acad. Sci. U.S.A.* **85**, 99–102 (1988).
2. W.H. Landschulz, P.F. Johnson and S.L. McKnight, The leucine zipper: A hypothetical structure common to a new class of DNA binding proteins, *Science* **240**, 1759–1764 (1988).
3. P. Sigler, Acid blobs and negative noodles, *Nature* **333**, 210–212 (1988).
4. S. Karlin, M. Morris, G. Ghandour and M.-Y. Leung, Efficient algorithms for molecular sequence analysis, *Proc. Natl. Acad. Sci. U.S.A.* **85**, 841–845 (1988).
5. S. Karlin, M. Morris, G. Ghandour and M.-Y. Leung, Algorithms for identifying local molecular sequence features, *CABIOS* **4** (1), 41–51 (1988).
6. S. Karlin, B.E. Blaisdell, E.S. Mocarski and V. Brendel, A method to identify distinctive charge configurations in protein sequences, with application to human herpesvirus polypeptides, *J. Mol. Biol.* **205**, 165–177 (1989).
7. S. Karlin and V. Brendel, Charge configurations in viral proteins, *Proc. Natl. Acad. Sci. U.S.A.* **85**, 9396–9400 (1988).
8. V. Brendel and S. Karlin, Association of charge clusters with functional domains of cellular transcription factors, *Proc. Natl. Acad. Sci. U.S.A.* **86**, 5698–5702 (1989).
9. S. Karlin and V. Brendel, Charge configurations in oncogene products and transforming proteins, *Oncogene* **5** (1), 85–95 (1990).
10. S. Karlin, B.E. Blaisdell and V. Brendel, Identification of significant sequence patterns in proteins, *Methods Enzymol.* **183**, 388–402 (1990).
11. V. Brendel, J. Dohlman, B.E. Blaisdell and S. Karlin, Very long charge runs in systemic lupus erythematosus associated autoantigens, *Proc. Natl. Acad. Sci. U.S.A.* **88**, 1536–1540 (1991).
12. V. Brendel and S. Karlin, Too many leucine zippers?, *Nature* **341**, 574–575 (1989).
13. W.R. Pearson and D.J. Lipman, Improved tools for biological sequence comparison, *Proc. Natl. Acad. Sci. U.S.A.* **85**, 2444–2448 (1988).