ABSTRACT

A FORTRAN program to analyze homology of letter strings (nucleotide or amino acid sequences) and to display the result in the form of a dot matrix is presented. The program is generally usable, user-friendly and has a number of options (filtering, "fudging," i.e., consideration of groups of homologous residues, and screening, i.e., display of only particular groups of residues) which greatly potentiate its analytical power.

INTRODUCTION

The current inflation of primary structural data has created an urgent need for methods which make it possible to comprehend the global features of nucleotide and amino acid sequences in their entirety. One of the most elegant and powerful ways to display graphically regularities and homologies of DNA sequences is the dot matrix introduced in two recent papers. J. Maizel (1) and, independently, T. Hunkapillar (2) developed programs which, formally resembling the distance plots of protein tertiary structures (3), allow a comparison of two sequences, generally m and n residues long, in the form of a matrix consisting of m rows and n columns. Thus, each letter (nucleotide) of the first sequence is compared to all the letters of the other, and the resulting matrix becomes output on the plotter; the matrix element $M(i,j)$ is a dot if the corresponding $i$-th and $j$-th letters are identical, being blank otherwise. If the sequences compared are identical, the dots form a diagonal running from the element $M(0,0)$ to $M(m,n)$. Homology with deletions and insertions gives rise to a series of short, parallel diagonal lines; palindromes form lines perpendicular to the main diagonal; homotactic tracts [e.g., oligo(A)] appear as lines parallel to one of the matrix axes [e.g., elements $M(a,i)$, $M(a,j)$, $M(a,k)$, etc.] and often give rise to solid square boxes. Weak and imperfect homologies, very difficult to detect otherwise, uncover themselves as faint lines with some degree of parallelism to the main diagonal.

To reduce the background noise inevitably present in comparisons of long nucleotide sequences, it was found useful not to compare the single letters, but doublets, triplets, etc., of one sequence to the other. This procedure, known as filtering,
Figure 1. Comparison of the nucleotide sequence of the immunoglobulin heavy chain γ2A cDNA (5) to itself. The cDNA is 1000 nucleotides long. The main diagonal, short palindromes and homotaetic tracts are well apparent. Filter = 3 (i.e., trinucleotides were compared at each position).

Figure 2. Comparison of the immunoglobulin heavy chain γ2A cDNA to its subsequence, namely, the 3'-proximal 430 nucleotides coding for the third constant domain of the heavy chain (CH3). Filter = 2. Depending on relative lengths of the sequences compared, the program automatically adjusts the output matrix into corresponding square or rectangular forms.
Figure 3. Comparison of amino acid sequences of the immunoglobulin heavy chain γ2A and its CH3 domain (5). Unfiltered data.

Figure 4. An example of "fudging." The amino acid sequence of the immunoglobulin gamma chain CH3 domain (5) was compared to the variable region of an anti-(iodo-nitrophenyl)-antibody (6). (A) Unfudged, unfiltered data; no homology is apparent. (B) Sequences compared in terms of three broadly defined homology sets of amino acid residues: [1] hydrophilic, small side chain residues with a propensity for polypeptide chain bends (Ser, Gly, Pro, Thr, Asn, Ala); [2] bulky, polar residues occurring on protein surface (Asp, Glu, Asn, Gln, Arg, Lys, His); [3] lipophilic residues buried within the protein structure (Leu, Ile, Val, Met, Ala, Tyr, Phe, Trp). Distant homology of the two primary structures now appears as a diagonal running approximately from residue 17 of the CH3 sequence to residue 106 of the V domain sequence. Homology requires multiple deletions and insertions.
Figure 5. The nucleotide sequence of the immunoglobulin CH3 domain compared to itself: the pattern of G-C base-pairs (the A-T base-pair had been screened out).

eliminates most of the random, fortuitous similarities while preserving the features of true homology.

To my knowledge, none of the existing matrix programs has yet been published. The aim of the present paper, therefore, is to make public a single FORTRAN source code of an efficient program which has been generalized to analyze not only the nucleotide sequences, but also amino acid sequences and, optionally, to perform filtering and "fudging." In fudging, the operator defines groups of letters (nucleotide or amino acid residues) which he wants to be treated as identical. Fudging is a powerful tool for visualizing a distant protein homology. Alternatively, the operator may decide that he wants only particular residues to be displayed; he can thus study such features as GC-rich tracts or hydrophobic acid residues in various primary structures, etc.

TECHNICAL CONSIDERATIONS

The program is written in "VAX-11 FORTRAN IV-PLUS," the FORTRAN IV dialect running on Digital VAX/VMS computers. The only non-standard features used are the OPEN and CLOSE statements associated with reading the input files containing letter
strings. The graphic output of the matrix itself is realized by calls to the plotting subroutines of the Versatex plotter "Versaplot 07." The calls would have to be changed if the matrix output is directed to a different plotter. However, since the calls conform to a standard format widely used by a large number of other plotting devices, it is simple to adjust the program to different environments.

In its present form, the program can handle strings up to 1000 letters long. Redimensioning for longer strings does not pose any problem, particularly on virtual memory machines like VAX/VMS. The letter strings are read into the program from terminal-formatted files which consist of records (lines) not longer than 80 letters. The reading is performed by calling the ARRFIL routine of the DNA-handling software package of Roger Staden (4) (the package, one of the most widely used, is freely available from Roger Staden).

The program is fully conversational and is intended for persons with a minimal knowledge of computers. It runs reasonably fast in the Digital VAX/VMS operation system. A matrix of the complexity $2 \times 10^5$ dots takes approximately 7 minutes real time to compute and some 13 minutes to be written from the main memory on the disc (the CPU time is several times shorter).

Persons interested in implementing, running or adjusting the program to their local environment are encouraged to communicate with the author.

ACKNOWLEDGEMENTS

I wish to thank Dr. Edgar Haber (Chief, Cardiac Unit), John Newell (Head, Cardiac Computer Center) and David Murphy (System Manager, CCC) for their support and help.

REFERENCES

1. Heiter, P.A., Max, E.E., Seidman, J.G., Maizel, J.V., and Leder, P. (1980) Cell 22, 197-207
2. Steinmetz, M., Freilinger, J.G., Fisher, D., Hunkapillar, T., Pereira, D., Weissman, S.M., Uehara, H., Nathenson, S. and Hood, L. (1981) Cell 24, 125-134
3. Rossmann, M.G. and Liljas, A. (1974) J. Mol. Biol. 85, 177-181
4. Staden, R. (1977) Nucleic Acid Res. 4, 4037-4051
5. Sikorav, J.L., Auffray, C., and Rougeon, F. (1980) Nucleic Acid Res. 8, 3143-3155
6. Bothwell, A.L.M., Paskind, M., Reth, M., Imanishi-Kari, T., Rajewski, K., and Baltimore, D. (1981) Cell 24, 625-637
