Acta Cryst. (1998). D**54**, 1071–1077

Iditis: Protein Structure Database

Stephen Gardner[a]* and Janet Thornton[b]

[a]*Synomics Ltd, 1 Cambridge Business Park, Cambridge CB4 0WZ, England, and* [b]*Department of Biomolecular Structure, University College London, Gower Street, London WC1E 7HX, England.*
E-mail: sgardner@synomics.com

(*Received 27 February 1998; accepted 18 May 1998*)

Abstract

The validation, enrichment and organization of the data stored in PDB files is essential for those data to be used accurately and efficiently for modelling, experimental design and the determination of molecular interactions. The *Iditis* protein structure database has been designed to allow the widest possible range of queries to be performed across all available protein structures. The *Iditis* database is the most comprehensive protein structure resource currently available, and contains over 500 fields of information describing all publicly deposited protein structures. A custom-written database engine and graphical user interface provide a natural and simple environment for the construction of searches for complex sequence- and structure-based motifs. Extensions and specialized interfaces allow the data generated by the database to used in conjunction with a wide range of applications.

1. Introduction

Protein structure data plays a pivotal role in the understanding of the mechanisms of molecular interaction. When available, it often becomes the key to unlocking the mechanisms of a disease or other molecular process and provides a springboard for the investigation of candidate compounds that may bind to and modify the action of target receptors or enzymes. Many new technologies that aim to accelerate our understanding of novel molecular systems are extending the utility of, and our dependence on, accurate protein structure data. Large-scale sequencing of microbial and other pathogenic genomes to identify sequences related to known disease-causing agents is now routinely performed. High-throughput screening relies largely on the identification of enzymes, receptors or larger cellular complexes which can be used to model a specific disease process and identify potentially active compounds. Rapid genotyping of individuals for specific polymorphic alleles will become an increasingly common method of screening patients for therapies. All these techniques are required to assign putative homologies and to describe the functional anatomy of new proteins and their mutations in a semi-automated fashion, for which thorough knowledge of all existing data is critical.

The number of known protein structures deposited at the Protein Data Bank (PDB) (Bernstein *et al.*, 1977; Abola *et al.*, 1987) has increased rapidly over the last few years to its current level of well over 6500 structures. These structures are currently distributed as formatted data files, one for each protein structure. The protein structure data deposited in these files are reduced to a very simple representation (PDB, 1992), which can obscure the biologically significant information contained in the data set. Whilst these structure files can be used to examine individual structures in detail, for example using molecular graphics, this organization of data does not allow the whole body of known structures to be searched in detail to identify specific features of interest from all available proteins. More recently the PDB has provided a limited relational catalogue of the protein structures available. The information available for searching is, however, limited to protein name, function, source and simple experimental information, and does not include detailed derived structural information.

In order to be able to identify or characterize structures, folds, or motifs from all proteins at once, it is necessary to extend the range of the structure data substantially, and to organize the fullest possible range of structural information into a fully functional database. This has been achieved in the *Iditis* protein structure database, where 500 fields of information are stored for all deposited PDB structures.

2. Background

The first generation of databases of protein structure was based around commercially available relational database management systems (RDBMSs) (Isogai *et al.*, 1987; Akrigg *et al.*, 1988; Islam & Sternberg, 1989) although some researchers attempted to use logic programming methods (Clark *et al.*, 1990; Paton & Gray, 1988; Gray *et al.*, 1988). These predominantly relational systems attempted to organize the protein structure data

Table 1. *Major data-derivation programs*

Program	Authors	Purpose	References & methods
ACCESS	S. J. Hubbard	Calculate the solvent-accessible surface area of proteins	Chothia, 1976; Lee & Richards, 1971; Satav *et al.*, 1980
ALTERNATES	E. G. Hutchinson	Builds ALTERNATE table.	
ATMNAMES	D. K. Smith	Determines atom- and residue-based properties	
BRKALN	D. K. Smith; E. G. Hutchinson	Checks *ATOM* records against *SEQRES* records	
BRKCLN	D. K. Smith	Cleans raw PDB files	IUPAC-IUB, 1970
BVCALC	D. Naylor	Calculates average *B* or *U*2 values per residue.	
CALPHA	D. K. Smith	Builds the *CALPHA* table	Nishikawa & Ooi, 1980, 1986
DISULF	D. K. Smith	Builds the *DISULPH* table	
HBOND	D. T. Jones	Identifies all potential hydrogen bonds in a protein	Baker & Hubbard, 1984
HLXTBL	T. P. Flores	Builds the HELIX and HELIXINT tables	Barlow & Thornton, 1988; Chothia *et al.*, 1981 Richmond & Richards, 1978; Kabsch & Sander, 1983
LIGAND	E. G. Hutchinson	Generates the LIGAND and WATER tables	
NEIGHBOUR	D. T. Jones	Identifies all non-bonded interactions in a protein	Narayana & Argos, 1984
NEWAMINOA	D. K. Smith, E. G. Hutchinson	Correlates records to build up the AMINO table	Kabsch & Sander, 1983; Nishikawa & Ooi, 1980, 1986; Efimov, 1986
NEWATOM	E. G. Hutchinson	Builds the ATOM table	
NEWCHAIN	D. K. Smith	Builds the CHAIN table	
NEWDBSHEET	E. G. Hutchinson	Builds the SHEET and STRAND tables	Kabsch & Sander, 1983; Richardson, 1976, 1977
NEWTAB2DS	D. K. Smith	Builds the GAMMATURN and BETATURN tables	Lewis *et al.*, 1973; Efimov, 1986; Milner-White *et al.*, 1988
NMRCLUST	L. A. Kelley	Clusters models within NMR ensembles	Kelley *et al.*, 1996
PROCHECK	R. A. Laskowski	Analyses stereochemical quality of protein structures	Laskowski *et al.*, 1993
PROTIN	D. K. Smith	Builds the PROTEIN table	
SALTBR	D. K. Smith	Builds the SALT table	Barlow & Thornton, 1983
SITE		Builds the SITE table	
SSTRUC	D. K. Smith	Determines information pertaining to the secondary structure of the protein	Kabsch & Sander, 1983; Nishikawa & Ooi, 1980, 1986
SUMMARY	A. L. Morris	Builds the SSSUM table	

in order to allow them to be used by researchers in a rational manner. Most groups made attempts to provide a richer set of data than was available in the PDB file by calculating derived data fields, such as secondary-structure assignments, Cα distance matrices and torsion angles. The RDBMSs provided search tools to identify chosen subsets of the data bank corresponding to particular search criteria.

The most successful of the first-generation relational systems were undoubtedly BIPED (Islam & Sternberg, 1989) and SESAM (Huysmans *et al.*, 1991), although both were limited by the underlying relational technology. BIPED was designed around the ORACLE RDBMS, and stored structural (and later homologous sequence) data for all available proteins, whilst SESAM contained a set of protein structures and their homologous sequences, and was based around the SYBASE RDBMS. At roughly the same time as these relational systems were being developed, computer scientists at Aberdeen University were creating an object-oriented version of the database (P/FDM) using the BIPED raw data files (Gray *et al.*, 1990).

At the time of development of *Iditis* (Thornton & Gardner, 1993; Oxford Molecular, 1997), using relational systems to store protein structure data was attractive, although they had some major limitations, notably lack of record order, inefficient storage and non-extensible query languages. At that time also, object-oriented systems were in their infancy, and were complex, inefficient and slow. Much progress with object-oriented systems has since been made (Gray *et al.*, 1996), although fully operational systems are still to come to fruition. Given the state of currently available technology, *Iditis* was designed as an extended pseudo-

relational database management system with a comprehensive data schema.

3. The *Iditis* family

The *Iditis* protein structure database has a number of components.

(i) *PROCHECK*, a protein-structure data-validation toolkit.

(ii) *NMRCLUST* and *NMRCORE*, automatic clustering tools for atoms and models in an NMR ensemble.

(iii) *Iditis Architect*, a structural data derivation toolkit.

(iv) *Iditis Data*, the comprehensive database of validated, derived protein structure data.

(v) *Iditis RDBMS*, a database management and search engine.

4. *Iditis Architect*

Iditis Architect is run each time new data becomes available to enter into the *Iditis* database. This may either be from a new release of the PDB, or when a new proprietary structure has been solved. The *Iditis Architect* data-derivation suite controls and executes the data validation and derivation. The 'raw' PDB files are first run through a clean-up program, *BRKCLN*, which identifies stereochemical inconsistencies and labelling errors within the PDB file. The *PROCHECK* (Laskowski *et al.*, 1993) suite arose out of the *BRKCLN* data-validation tools, and was subsequently extended and converted to its present form by Roman Laskowski on a grant from Oxford Molecular. *PROCHECK* has subsequently been made freely available and is distributed by PDB and CCP4.

The validation process is essential to ensure that the coordinate data from which all other derived information is calculated are as reliable and consistent as possible. It is imperative that, when performing comparative analysis using data from many structures derived from different sources, the data used should conform exactly to the same standards and protocols.

All changes that are made by the *BRKCLN* program are flagged in an output 'clean' PDB file, and those flags are incorporated in the *Iditis* database. No changes to atomic coordinate information are made, other than the relabelling of mislabelled atoms. No attempt is made, as was performed in other data-preparation routines, to correct incorrect data, or to fill in, by modelling and minimization, areas of the structure whose electron density is poorly defined. These areas of structure are simply flagged, and reported as being poorly defined in *Iditis*.

Once *BRKCLN* and its companion *BRKALN*, which realigns the *SEQRES* records to match the reported structure, have been run on a file, it is ready to be passed through the data-derivation programs themselves. The programs that are used are mainly implementations of standard, published techniques, and generate information in ASCII output files capable of being read into *Iditis*. As the final step, the intermediate files are reformatted into data 'stripes' that can then be loaded into *Iditis* using the data-definition language tools.

In total, there are over 45 programs which interact to produce the final set of data stripes for the *Iditis* database. The important derivation programs that are used are shown in Table 1.

The use of NMR methods for solving small to medium size polypeptides has become much more feasible in recent years. One of the advantages of NMR methods is the presentation of multiple solution structures. Frequently, ensembles of over 30 model solutions of the structure are presented in a single PDB file, and occasionally over 50. This allows more detailed analysis of the movements of a structure in solution, and may avoid some of the time averaging and crystal contact problems associated with crystallography.

However, it makes very little sense to store all 30 sets of secondary-structure assignments, φ/ψ angles *etc.* for each of the models in the ensemble, as this information would overload a researcher in a practical situation. Both because of the need to store data efficiently, and the need to present the user with useful, usable infor-

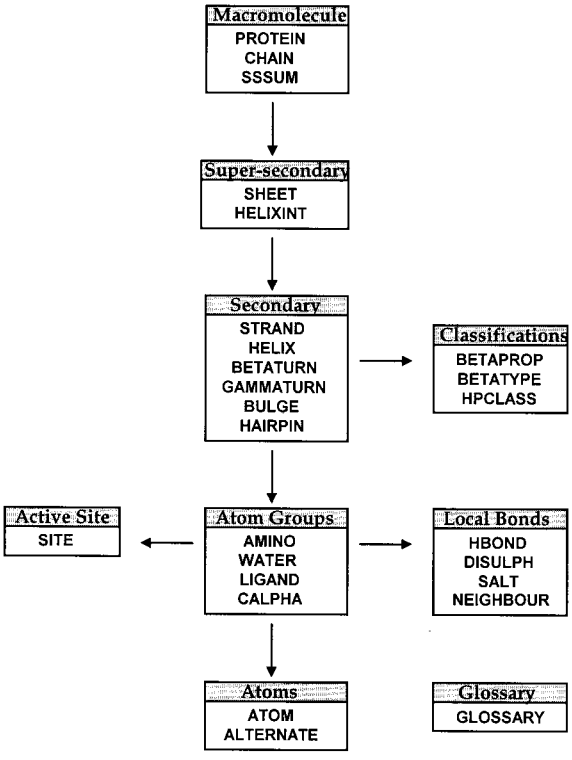

Fig. 1. The *Iditis* database schema.

mation, a small number of structures representative of the conformation/s in the ensemble must be chosen for the protein.

To choose representatives of the major conformational clusters within the ensemble, the automated clustering package *NMRCLUST* is used. *NMRCLUST* (Kelley *et al.*, 1996) and *NMRCORE* (Kelley *et al.*, 1997), which are both freely available, were used automatically to select the most representative models for each conformational cluster to enter into the *Iditis* database. This usually leads to five or less models representing an ensemble of 30–50 models. Entering data for only the representative models removes redundancy in the database, whilst retaining most of the diversity of the conformational information in a manageable form.

5. The *Iditis* data schema

The *Iditis* database includes all X-ray, neutron, and electron diffraction and NMR-derived protein data sets from the PDB, and is updated in synchrony with PDB releases. The *Iditis* database schema (Fig. 1) is designed to remain as close to the user's conceptual understanding of protein structure as possible. This means that queries can be asked in as simple a way as possible, without the need for the user to learn artificial database constructs.

Iditis uses a data representation that is very close to a true binary copy of the data. There is no extra information stored in a table other than the reformatted raw data itself. This data is then compressed using a version of the gzip algorithm, and accessed in compressed form during database queries. Consequently a protein structure stored in *Iditis* occupies only 60% of the space required by the PDB file it is derived from, despite *Iditis* holding 500 fields of structural data, and being able to recreate exactly the original or validated PDB structure.

The storage requirements for the *Iditis* schema (6335 proteins, January 1998 *Iditis* data release) in its different forms are shown in Fig. 2.

Data for each protein in *Iditis* are stored in consecutive rows in the tables, and can be manipulated on a protein-by-protein basis. This allows whole protein entries in the database to be added, removed, or updated independently, so that only entries that have changed from the previous release of the database have to be distributed. This mechanism ensures that proprietary or other unreleased structures do not have to be re-entered into the database every time a new version of the publicly available data is released.

The headers and footers of the original PDB structures are stored in *Iditis* to allow the regeneration of PDB files. The user can regenerate either an exact image of the original PDB file, or a 'cleaned' version of that file, which resulted from the validation programs.

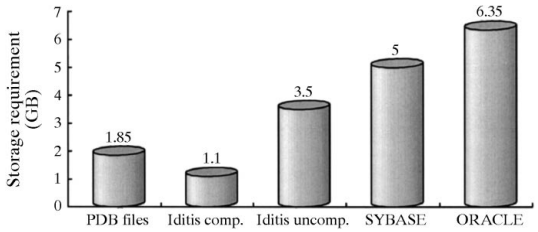

Fig. 2. Database storage requirement for protein structure data in various formats.

6. The *Iditis* database management system

One of the main design goals for *Iditis* was to allow users to ask natural protein-based queries as easily as possible. To this end a graphical user interface based on a series of forms was developed to allow users easily to enter required query constraints. These are then automatically converted by *Iditis* into a query that can be run against the database. Any required table joins are, as far as possible, determined by the system, or the user is prompted in plain English for clarification of the meaning of the query. Hits are returned in textual, molecular or numeric graphical form. Execution times for searches are typically in the order of seconds to minutes depending on their complexity.

The main graphical user interface allows the creation of a wide variety of queries regarding any combination of the following.

(i) Protein properties, including name, function, authors, dates, resolution, *R* factor *etc.*

(ii) Sequence, including regular expressions and physicochemically related sets.

TABLE A

AMINOID	S
155C/-0001-	N
155C/-0002-	E
155C/-0003-	G
155C/-0004-	D
155C/-0005-	A
155C/-0006-	A
155C/-0007-	K
155C/-0008-	G
155C/-0009-	E

TABLE B

L	AMINOID	S
1	155C/-0001-	N
2	155C/-0002-	E
3	155C/-0003-	G
4	155C/-0004-	D
5	155C/-0005-	A
6	155C/-0006-	A
7	155C/-0007-	K
8	155C/-0008-	G
9	155C/-0009-	E

TABLE C

AMINOID	S	SM3	SM2	SM1	SP1	SP2	SP3
155C/-0001-	N	.	.	.	E	G	D
155C/-0002-	E	.	.	N	G	D	A
155C/-0003-	G	.	N	E	D	A	A
155C/-0004-	D	N	E	G	A	A	K
155C/-0005-	A	E	G	D	A	K	G
155C/-0006-	A	G	D	A	K	G	E
155C/-0007-	K	D	A	A	G	E	K
155C/-0008-	G	A	A	K	E	K	E
155C/-0009-	E	A	K	G	K	E	F

Fig. 3. Relational fixes.

(iii) Secondary structure, including regular expressions, and structurally related sets.

(iv) Hydrogen-bonding interactions.

(v) Non-bonded neighbour interactions.

(vi) Disulfide bridges.

(vi) Ligand groups and active sites.

(vii) Torsion angles, including φ, ψ, ω.

(viii) Structural quality, including whole data set and residue-by-residue quality metrics.

(ix) Accessibility, including absolute and relative for a variety of atom groups.

Whilst extensive, these represent only a small fraction of the data available through *Iditis*. Other less commonly used information can be queried through a generic query builder interface. In addition the queries can be limited to defined sets of proteins, or the results of previous queries can be combined logically through the interface.

One key factor in the simplicity of this interface is an implicit knowledge of order of elements within a query. The *Iditis* DBMS uses an extended version of the Structured Query Language (SQL) (ANSI, 1986; ANSI/ISO, 1989) which is capable of recognising record order within its tables. This order-based SQL (OBSQL) provides extensions to standard SQL that allow relative positions of elements within a motif or fragments to be easily defined and efficiently searched.

6.1. *Order in a pseudo-relational database*

Many queries on protein structure data reflect the fact that protein structures are made up of a series of serially linked amino acids. The order of the data (amino acids) must, therefore, be easily accessible to the protein structure database. Because standard relational systems do not store the order of records, different 'fixes' have to be applied to allow order to be used. There are two ways around this problem in a standard relational system, which are illustrated in Fig. 3.

Table *A* in Fig. 3 illustrates a simple table (AMINO) with an identifier code (AMINOID) and an amino-acid sequence (*S*). Ideally the data would be stored simply, as in Table *A*, and the implicit order of the records could be used in queries. This is not however possible in standard relational databases. Instead there are two strategies by which this table can be altered to allow the retrieval of ordered fragments with a given sequence. The first, illustrated in Table *B* of Fig. 3, requires explicit row numbers that can be used in conjunction with self-join operations or with external programs. When looking for a target sequence (*e.g.*, *ACDEF*), all examples of *A* are found in one copy of the table, then all examples of *C* in another copy of the table, and so on with *D*, *E* and *F*. Finally all five lists are joined by complex self-joins or external 'shell' applications. Unfortunately, neither of these processes can be made transparent to the user, and they are extremely inefficient both in terms of additional complexity and query processing time.

Another means of overcoming the order problem, illustrated in Table *C* of Fig. 3, is to add extra fields of information that duplicate values for preceding and following entries (SM3–SP3). This 'environmental' information is then available on a single row and can be accessed easily. This designed redundancy of information is however extremely costly in storage and leads to complex queries.

Iditis, in contrast, stores the data as in Table *A*, and uses the implict record order through simple extensions to the SQL syntax allow the position of elements in the query to be specified, as shown below in a query to find all β-turns in γ-crystallin. This query specifies that the residues at positions i+1 and i+2 should not be in a helical conformation, whilst the distance between the Cαs of residues i and i+3 should be less than 7 Å. (This query would normally be generated through a graphical user interface, rather than by a user typing SQL statements, and could be generated more simply from the BETATURN table, Fig. 4.)

Flexible order constraints allow target values to occur within a range of records. Searches using flexible order constraints can be used to answer two types of query. An example of the first type of query would be to scan for all instances of a proline residue within four residues of the N-terminus of an α-helix. This query would take the form

```
SELECT AMINOID, SEQ1[−5:5], XSS[−5:5]
FROM AMINO
WHERE SEQ1[0, 3] =′ P′ANDXSS =′ hH′
```

The key constraint is that the proline should lie within four residues of the N-terminus of the helix, allowing four possible acceptable positions for the proline.

The second type of search made possible by the flexible column offsets will return a range of residues that all have a certain property. An example of a query of this form is shown below

```
1> SELECT AMINOID, SEQ[:3], CADP3, PHI[1:2], PSI[1:2]
2> FROM AMINO
3> WHERE BROOKID='1GCR' AND CADP3 < 7.0
4> AND XSS[1] = '+NONHELICAL++NONHELICAL+'

AMINOID     SEQ[:3] CADP3   PHI[1]  PHI[2]  PSI[1]  PSI[2]
----------- ------- ------ ------- ------- ------- -------
1GCR/-0008-    DRGF    6.5   -77.3    66.2   168.2    24.3
1GCR/-0009-    RGFQ    6.0    66.2    69.2    24.3    33.2
1GCR/-0022-    CPNL    6.5   -66.7  -156.3   -24.7   107.0
1GCR/-0025-    LQPY    5.5   -52.1   -57.9   -41.7   -35.2
1GCR/-0026-    QPYF    5.8   -57.9   -90.2   -35.2   -24.6
1GCR/-0047-    RPNY    6.5   -65.5    68.1   170.1    30.1
1GCR/-0048-    PNYQ    5.9    68.1    63.4    30.1    47.8
1GCR/-0096-    DDFR    6.0    68.4    55.9    29.5    36.0
1GCR/-0109-    CPSL    6.4   -85.3  -155.8    -9.6   105.9
1GCR/-0136-    MPSY    6.2   -66.4    63.2   159.5    35.6
1GCR/-0137-    PSYR    6.2    63.2    68.6    35.6    39.6

12 hits match query specifications
```

Fig. 4. Select all β-turns in γ-crystallin.

```
SELECT AMINOID, SEQ1[−2:7], XSS[−2:7], PACC[:5]
FROM AMINO
WHERE NOT PACC[, 5]<10 AND XSS =′ hH′
```

This query looks for all α-helices that begin with a stretch of at least six accessible residues. The query translates literally as, 'having identified all N-termini of α-helices, exclude any region which, between residues i to i+5, contains one or more residues with greater than 10% residue accessibility'.

Both types of queries mentioned in this section are impossible to execute in a standard relational environment. It may be possible to write a post-processing utility that will take the results of a set of queries along with row-number information, and correlate the results to give the same effect, but this is a very inefficient and complex way to achieve the same result.

6.2. *Result display*

In addition to standard text displays, *Iditis* provides a mechanism to pass query results to a series of protein-specific applications, such as Ramachandran plots (Fig. 5) (Ramachandran *et al.*, 1963), protein visualizers and molecular-alignment systems. Application triggers can be registered in the database, so that on typing a user-defined command keyword, specified data fields for the current hit list can be exported to an intermediate data file and the relevant analysis application executed.

Further application interfaces such as a loop candidate selection system for homology modelling have been added to *Iditis*. This provides a visual interface to align sequences to be modelled with their structural homologues, from which the system can automatically identify loop regions and allow the user to 'paint' additional constraints, such as accessibility, physicochemical types, distances, interactions, or quality onto the loop query. The necessary query is then generated and a set of candidate loops is extracted from the database for further use in modelling programs.

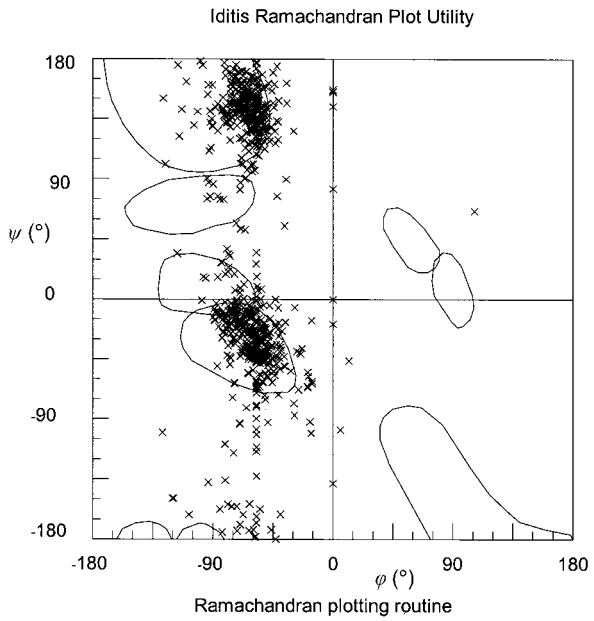

Fig. 5. Ramachandran plot created from results of *Iditis* query.

The authors would like to acknowledge the contributions of the following people and groups at various times to the *Iditis* project; at Birkbeck College, David K. Smith, Tom Blundell, Peter Murray-Rust, Mike Sternberg, Ian Tickle, Suhail Islam, Fiona Hayes, Roman Laskowski, Geoff Barton, Carrie Wilmot; at Daresbury Laboratories, Alan Bleasby; at University College, Gail Hutchinson, Christine Orengo, Simon Hubbard, Mark Swindells, David Jones, Louise Morris, Malcolm MacArthur, Tom Flores; at Oxford Molecular, Pete Bennett, Rob Nelson, Rob Scoffin, Dave Walker, Chris Brown; and at Leicester University, Mike Sutcliffe, and Lawrence Kelley. *Iditis* is distributed by Oxford Molecular Ltd, Medawar Centre, Oxford Science Park, Oxford OX4 4GA, UK.

References

Abola, E. E., Bernstein, F. C., Bryant, S. H., Koetzle, T. F. & Weng, J. (1987). In *Crystallographic Databases – Information Content, Software Systems, Scientific Applications*, edited by F. H. Allen, G. Bergerhoff & R. Sievers, pp. 107–132. Bonn/Cambridge/Chester: Data Commission of the International Union of Crystallography.

Akrigg, D., Bleasby, A. J., Dix, N. I. M., Findlay, J. B. C., North, A. C. T., Parry-Smith, D., Wooton, J. C., Blundell, T. L., Gardner, S. P., Hayes, F., Islam, S. A., Sternberg, M. J. E., Thornton, J. M. & Tickle, I. J. (1988). *Nature (London)*, **335**, 745–746.

ANSI (1986). ANSI X3.135–1986.

ANSI/ISO (1989). ANSI X3H2-89–151/ISO DBL Can-3a.

Baker, E. N. & Hubbard, R. E. (1984). *Prog. Biophys. Mol. Biol.* **44**, 97–179.

Barlow, D. J. & Thornton, J. M. (1983). *J. Mol. Biol.* **168**, 867–885.

Barlow, D. J. & Thornton, J. M. (1988). *J. Mol. Biol.* **201**, 601–619.

Bernstein, F. C., Koetzle, T. F., Williams, G. J. B., Meyer, D. F. Jr, Brice, M. D., Rodgers, J. R., Kennard, O., Shimanouchi, T. & Tasumi, M. (1977). *J. Mol. Biol.* **112**, 535–542.

Chothia, C. (1976). *J. Mol. Biol.* **105**, 1–14.

Chothia, C., Levitt, M. & Richardson, D. (1981). *J. Mol. Biol.* **145**, 215–250.

Clark, D. A., Barton, G. J. & Rawlings, C. J. (1990). *J. Mol. Graphics*, **8**, 94–107.

Efimov, A. V. (1986). *Mol. Biol. (Moscow)*, **20**, 2250–2260.

Gray, P. M. D., Kemp, G. J. L., Rawlings, C. J., Sander, C., Thornton, J. M. & Wodak, S. J. (1996). *Trends Biochem. Sci.* **21**, 251–256.

Gray, P. M. D., Moffat, D. S. & Paton, N. W. (1988). *Advances in Database Technology. Proceedings of EDBT 1988 (Venice)*, pp. 34–48. Berlin: Springer-Verlag.

Gray, P. M. D., Paton, N. W., Kemp, G. J. L. & Fothergill, J. E. (1990). *Protein Eng.* **3**, 235–243.
Huysmans, M., Richelle J. & Wodak, S. J. (1991). *Proteins Struct. Funct. Genet.* **11**, 59–76.
Islam, S. A. & Sternberg, M. J. E. (1989). *Protein Eng.* **2**, 431–442.
Isogai, Y., Toma, K., Tagashira, M., Watanabe, H. & Go, N. (1987). *Tanpakusitu Kakusan Kouso*, **32**. (In Japanese.)
IUPAC-IUB (1970). *J. Mol. Biol.* **52**, 1–17.
Kabsch, W. & Sander, C. (1983). *Biopolymers*, **22**, 2577–2637.
Kelley, L. A., Gardner, S. P. & Sutcliffe, M. J. (1996). *Protein Eng.* **9**, 1063–1065.
Kelley, L. A., Gardner, S. P. & Sutcliffe, M. J. (1997). *Protein Eng.* **10**, 737–741.
Laskowski, R. A., MacArthur, M. W., Moss, D. S. & Thornton, J. M. (1993). *J. Appl. Cryst.* **26**, 283–291.
Lee, B. & Richards, F. M. (1971). *J. Mol. Biol.* **55**, 379–400.
Lewis, P. N., Momany, F. A. & Scheraga, H. A. (1973). *Biochim. Biophys. Acta*, **303**, 211–229.
Milner-White, E. J., Ross, B. M., Ismail, R., Belhadj-Mostefa, K. & Poet, R. (1988). *J. Mol. Biol.* **204**, 777–782.
Narayana, S. V. L. & Argos, P. (1984). *Int. J. Pept. Protein Res.* **24**, 25–39.
Nishikawa, K. & Ooi, T. (1980). *Int. J. Pept. Protein Res.* **16**, 19–32.
Nishikawa, K. & Ooi, T. (1986). *J. Biochem.* **100**, 1043–1047.
Oxford Molecular (1997). *Iditis Version 3.1 User Manual*, Oxford Molecular Ltd, Medawar Centre, Oxford Science Park, Oxford OX4 4GA, UK.
Paton, N. W. & Gray, P. M. D. (1988). *PROLOG and Databases: Implementations and New Directions*, edited by P. M. D. Gray & R. J. Lucas, pp. 251–266. Chichester: Ellis Horwood.
PDB (1992). Brookhaven National Laboratory, Long Island, New York, USA.
Ramachandran, G. N., Ramakrishan, C. & Sasisekharan, V. (1963). *J. Mol. Biol.* **7**, 95–99.
Richardson, J. (1976). *Proc. Natl Acad. Sci. USA*, **73**, 2619–2623.
Richardson, J. (1977). *Nature (London)*, **268**, 495–500.
Richmond, T. J. & Richards, F. M. (1978). *J. Mol. Biol.* **119**, 537–555.
Satav, Y., Watanabe, Y. & Mitsui, Y. (1980). *J. Biochem.* **88**, 1739–1755.
Thornton, J. M. & Gardner, S. P. (1993). *Chem. Des. Autom. News*, **8**(4), 18–22.