UCSF MidasPlus

Molecular Interactive Display And Simulation

USER'S MANUAL

May 1991

Computer Graphics Laboratory
School of Pharmacy
University of California, San Francisco

Additional copies of this manual are available for $10 each, including postage, by writing to MIDAS Software Distribution, Computer Graphics Laboratory, School of Pharmacy, University of California, San Francisco, CA 94143-0446. Please include a check or money order payable to The Regents of the University of California with your request. Sorry, but purchase orders cannot be accepted. Orders received without payment enclosed will be returned unprocessed.

Copyright (c) 1983, 1985, 1989, 1991 by The Regents of the University of California.
All rights reserved.

MidasPlus User's Manual

Table of Contents

Introduction 1
Part I: Molecular Modeling with MIDAS 5
- Important Concepts in MIDAS 5
- Connectivity and Templates 5
- Coordinates 7
- Protein Data Bank Format 7
- Examples of PDB Format 9
- Common Errors in PDB Format Files 12
- Building a MIDAS Database 14
- Using midas.in 14
- Building and Modifying MIDAS Templates 15
- More on PDB format versus MIDAS databases 16
- Displaying the Model 17
- Manipulating the Model 17
- On-line Help 19

Part II: Advanced Concepts: Surfaces and Hydrogen Atoms 21
- Molecular Surfaces 21
- VDW Surfaces 21
- Solvent Accessible Surfaces 22
- Modeling Hydrogen Atoms 25
- Hydrogens in Protein Data Bank Files 25
- Making Videos 26

Part III: Command Reference Guide 27
- Referencing Models, Residues, and Atoms 27
- Command Overview 30
- MIDAS Start-Up 30
- Command Synopsis 30
- Addaa 35
- Addgrp 35
- Alias 35
- Align 36
- Angle 36
- Assign 36
- Bond 37
- Brotation 37
- Cd 38
- Center 38
- Chain 38
- Clip 38
- Cofr 38
- Color 39

Conic 40
Copy 41
Delegate 41
Delete 41
Devopt 42
Display 42
Distance 42
Echo 43
Fix 43
Fixreverse 43
Freeze 43
Getcrd 43
Help 44
Intensity 44
Label 44
Link 44
Match 45
Matrixcopy/Matrixget/Matrixset 45
Midaspush/Midaspop 45
Move 46
Open 46
Pdbrun 47
Push/Pop 49
Read 49
Record 49
Redraw 49
Reset 50
Reverse 50
Ribbon 50
Rlabel 51
Rock 51
Roll 51
Rotation 52
Run 52
Save 52
Savepos 53
Scale 53
Section 53
Select 54
Set/Unset 54
Setcom 58
Show 58
Sleep 58
Source 58
Speed 59
Stereo 59
Stop 59

Surface 59
Swapaa 60
Swapna 60
System 60
Thickness 61
Turn 61
Update 61
Vdw 61
Vdwopt 62
Wait 62
Watch/Watchopt 63
Window 63
Write 63
Version 63
Zone 64
Appendix 1: Available Protein Data Bank Models 65
Appendix 2: PDB Atom Naming Conventions for Amino Acids 77
Appendix 3: Special Characters and Symbols Used in MIDAS 78
Appendix 4: Default Options, Aliases and Device Assignments 80
Appendix 5: The MidasPlus Delegate Mechanism 81
Appendix 6: MIDAS Utilities 85

Command Index by Keyword

abbreviate ..... alias
access ..... open
accessible ..... surface
acid ..... addaa swapaa swapna
active ..... select
add ..... addaa addgrp
alias ..... alias
align ..... align match
amino ..... addaa swapaa
angle ..... angle rotation
antialias ..... devopt
append ..... addaa addgrp
atom ..... color display section show thickness
backbone ..... chain
background ..... set
backwards ..... brotation fixreverse reverse
base ..... swapna
batch ..... read source
big ..... scale set
bond ..... assign bond brotation color display fix fixreverse reverse rotation section show thickness
brightness ..... intensity
bye ..... stop
center ..... center cofr set setcom window
clipping ..... assign clip section thickness
coincide ..... match
color ..... color intensity set
command ..... delegate pdbrun read run set source system
connelly ..... surface
contact ..... watch
continue ..... midaspop
control ..... set
coordinate ..... getcrd
copy ..... copy
cpk ..... conic
database ..... write
delete ..... delete
density ..... vdwopt
depth-cueing ..... intensity
dihedral ..... angle
directory ..... cd
display ..... center open show surface window
distance ..... distance match watch
end ..... stop
enlarge ..... scale set
escape ..... system
exchange ..... swapaa swapna
execute ..... delegate pdbrun run system
exit ..... stop
expand ..... scale set

extend ..... addaa addgrp
file ..... read record source write
filenames ..... set
finish ..... stop
font ..... set
freeze ..... freeze select
fullscreen ..... set
function ..... devopt set
group ..... addgrp
halfbond ..... color set
halt ..... freeze
hardcopy ..... copy
helix ..... ribbon
help ..... help
hide ..... display label section show thickness
hither ..... assign clip section thickness
horizontal ..... move
hue ..... intensity
identifier ..... label
image ..... push pop reset save savepos
inactive ..... select
information ..... help
inhibit ..... select
input ..... pdbrun run
intensity ..... intensity
interpret ..... pdbrun run
interrupt ..... midaspush
join ..... bond
label ..... color label rlabel set
left ..... move
length ..... distance
line ..... set
link ..... bond
mainchain ..... chain
mass ..... cofr setcom
match ..... match
message ..... echo
model ..... open
mouse ..... set
move ..... rock select
ms ..... surface
mutate ..... swapaa swapna
name ..... label rlabel
nucleic ..... swapna
object ..... open
open ..... open
opposite ..... brotation fixreverse reverse
option ..... devopt set
orientation ..... push pop reset save savepos
orthographic ..... set
output ..... pdbrun run
pair ..... stereo
panel ..... set

pause ..... midaspush sleep wait
PDB ..... pdbrun write
picture ..... push pop reset save savepos
pop ..... midaspop pop
position ..... getcrd push pop reset save savepos
prevent ..... select
printout ..... copy
program ..... delegate pdbrun run
prune ..... delete
push ..... midaspush push
quit ..... stop
radius ..... vdwopt
reaction ..... speed
read ..... read source
record ..... record
redirect ..... pdbrun run
reduce ..... match scale set
remember ..... push record savepos
remove ..... delete fix rlabel
replace ..... swapaa swapna
reply ..... echo set
reset ..... pop reset
residue ..... addaa rlabel swapaa swapna
restart ..... midaspop
resume ..... midaspop
retain ..... push savepos
retrieve ..... pop reset
return ..... midaspop
reverse ..... brotation fixreverse reverse
ribbon ..... ribbon
Richardson ..... ribbon
right ..... move
RMS ..... match
rock ..... freeze rock
roll ..... freeze roll
rotation ..... assign brotation cofr fix fixreverse freeze reverse roll rotation set turn
run ..... delegate pdbrun run system
saturation ..... intensity
save ..... push record save savepos write
scale ..... scale set
scaling ..... assign
screen ..... center window
script ..... read source
secondary ..... ribbon
select ..... select
sensitivity ..... speed
session ..... save
shadowed ..... conic
sheet ..... ribbon
shell ..... midaspush system
short ..... alias
show ..... center display label open set show window
shrink ..... scale set

size ..... scale set
slider ..... assign
small ..... scale set
smooth ..... devopt
solvent ..... surface
source ..... read source
space-filling ..... conic
speed ..... speed
sphere ..... set
stereo ..... stereo
steric ..... watch
stop ..... freeze midaspush select stop
structure ..... ribbon
substitute ..... alias
superimpose ..... match
surface ..... color open surface vdw vdwopt write
swap ..... swapaa swapna
text ..... set
thickness ..... set
trackball ..... set
translate ..... move
translation ..... assign
turn ..... turn
VDW ..... vdw vdwopt
vertical ..... move
view ..... push pop reset save savepos
visible ..... center display label set show window
wait ..... sleep wait
window ..... center midaspush midaspop window
words ..... set
write ..... record save write
x ..... assign getcrd move rock roll turn
y ..... assign getcrd move rock roll turn
yon ..... assign clip section thickness
z ..... align assign getcrd move rock roll turn

(This page intentionally left blank)

Introduction

Background

The Molecular Interactive Display and Simulation (**MIDAS**) System is a collection of programs developed by the Computer Graphics Laboratory at the University of California, San Francisco. The major component of the **MIDAS** system is an interactive graphics display program, **MidasPlus**, designed for the display and manipulation of macromolecules such as proteins and nucleic acids. Several ancillary programs are also part of the system and allow for such features as computing the surface of a molecule, the selection of an active site region within a molecule, computation of electrostatic charge potentials, etc. At the core of **MIDAS** is an unusually coherent hierarchical database system, designed specifically for macromolecules and both compact in its storage requirements and fast in its data access.

MIDAS is the most recent in a series of interactive molecular graphics systems whose direct lineage extends back to the first developments in molecular graphics at Project MAC, Massachusetts Institute of Technology, in 1964.[1,2] National Institutes of Health (NIH) support began with the formation of the Computer Graphics Laboratory at Princeton University in 1969 and resulted in a number of pioneering developments including CAAPS (Computer Aided Analysis of Protein Structure).[3] In 1976 this NIH research resource moved to UCSF. A new graphics package[4] was designed to operate under the UNIX[5] operating system and a new molecular graphics system (MMS) designed, initially in collaboration with the group under Professor J. Kraut at UC San Diego. This evolved into a system, MIDS, which was good enough to accommodate the new developments in color and which also made possible the display of interacting surfaces.[6] The MIDS system was used by numerous visitors to the our laboratory in the late 1970's.

In 1980 we decided to redesign the system completely, making use of the lessons learned over the previous 15 years. The result, **MIDAS**,[7,8] emphasized highly interactive display and manipulation, with a data structure designed for very fast access to large and complex molecules such as proteins and nucleic acids.[9] A great deal of effort was expended on the interactive selection, manipulation and docking of drugs and receptors. Over 200 publications have resulted from work at the UCSF Computer Graphics Laboratory using **MIDAS**.

MIDAS was developed in a university research environment and has grown considerably in functionality and size as a response to new ideas; often these ideas have come from **MIDAS** users themselves, now numbering over 120 from 27 states and 12 countries. **MIDAS** was originally developed on the UNIX operating system for use with an Evans and Sutherland Picture System 2 display, however between 1982 and 1989 **MIDAS** has run on a variety of graphics display engines (PS2, MPS, PS300 family, and Silicon Graphics IRIS family) and operating systems (BSD UNIX, System V UNIX and VMS).

MidasPlus

Due to the substantial advances in graphics display technology and workstation functionality and performance in the mid-1980's, we decided in early 1989 that it was time to "rewrite" the display portion of the **MIDAS** system in order to take advantage of these advances in technology. This work was done during the summer of 1989 and the resulting program was named **MidasPlus**. **MidasPlus** has significantly increased functionality and performance when compared to the previous version of the **MIDAS** display program; many commands execute 5x - 10x faster.

---

[1] R. Langridge and A.W. MacEwan, in *Proceedings, IBM Scientific Computing Symposium on Computer Aided Experimentation* (1965).

[2] C. Levinthal, *Scientific American* **214** (6), 42-52 (1966).

[3] R. Langridge, *Federation Proceedings of the American Society of Experimental Biology* **33**, 2322-2328 (1974).

[4] T.E. Ferrin and R. Langridge, *Computer Graphics* **13**, 320-331 (1980).

[5] D.M. Ritchie and K. Thompson, *Communications of the ACM* **17**, 7 (1974). UNIX is a registered trademark of AT&T Bell Laboratories.

[6] R. Langridge, T.E. Ferrin, I.D. Kuntz, M.L. Connolly, *Science* **211**, 661-666 (1981).

[7] T.E. Ferrin *et.al.*, *J. Mol. Graphics* **2**, 55 (1984);

[8] T.E. Ferrin *et.al.*, *J. Mol. Graphics* **6**, 13-27 (1988).

[9] T.E. Ferrin *et.al.*, *J. Mol. Graphics* **6**, 2-12 (1988).

This was accomplished while maintaining complete compatibility with the original **MIDAS** command language and **MIDAS** database and session files. New features in **MidasPlus** include:

- Full Protein Data Bank (PDB) support. Anything that can be done with a **MIDAS** database can now be done directly from within **MidasPlus**. For example, **MidasPlus** can open PDB format files for input directly, without needing to use the *midas.in* program beforehand. "Compressed" PDB files are also supported.
- Graphical-based user interaction, including a "virtual trackball" technique for interactive rotations and direct manipulation of eye position and hither and yon clipping planes.
- Ability to generate space filling images with shadows cast from multiple light sources.
- Ability to generate "ribbon" drawings for depicting secondary protein structure.
- Ability to spawn and interactively communicate with other programs from within **MidasPlus**. Data is transferred as a PDB format file of the currently displayed image. The spawned "delegate" program can manipulate the currently displayed image through standard **MidasPlus** commands.
- Enhanced support for stereo viewing of images, including generating images with either positive or negative horizontal parallax.
- Support for Spatial Technology's six degree-of-freedom "Spaceball" interactive input device.
- Enhanced control of Van der Waals surfaces with immediate updating of the screen image. Non-standard atom radii can be defined directly on the **MidasPlus** command line.
- Interactive monitoring for interatomic contacts during bond and dihedral angles rotations.
- Direct support of MS surface files, including "compressed" MS files. There is no need to create a separate **MIDAS** *.srf* database any longer.
- Full integration with the standard IRIS 4Sight windowing system.

Manual Organization

The MidasPlus User's Manual is divided into several sections. Part I discusses the important concepts in building **MIDAS** databases. Since the input data used by **MidasPlus** is critical to a productive modeling session, careful reading of this section is essential. Part II discusses many of the advanced concepts in **MIDAS** such as molecular surface displays, computing electrostatic potential surfaces, adding hydrogen atoms to a database, etc. Beginning users may skip this section on first reading. Part III of the manual is intended as a reference guide and gives a concise description of each of the commands available in **MidasPlus**. Appendices at the end of this manual describe such things as available models in the Protein Data Bank library, atom naming conventions, special characters and symbols used by **MidasPlus**, default options, aliases and device assignments, and differences between **MidasPlus** and previous versions of **MIDAS**. Special attention should be given to Appendix 6, as this section describes the many **MIDAS** utility programs that are part of the overall system.

Recent changes to the MidasPlus User's Manual are now indicated with a vertical bar in the right margin area of the manual, such as shown in this paragraph. Changes, additions and bug fixes are also detailed in the "Release Notes" document. Thus users already familiar with **MidasPlus** can quickly determine what all has changed when a new release is distributed.

Future Development Plans

The **MIDAS** system is used daily at UCSF as part of an active research program in pharmaceutical chemistry. As such, changes and enhancements are are made from time to time, often in response to new ideas from our users. While we make no promises as to when, if ever, new versions of **MIDAS** will be released, we do encourage your feedback. If you discover bugs or have ideas you think would be particularly useful to others, please send us electronic mail at **midas-ideas@cgl.ucsf.edu**. If the mail concerns a program bug, please include the version of **MidasPlus** you are using as determined from the output of the "version" command.

Acknowledgements

The original version of **MIDAS** was written by Prof. Thomas Ferrin and Dr. Conrad Huang. Dr. Huang is also the principle author of **MidasPlus**. Eric Pettersen, Greg Couch and Laurie Jarvis have all contributed to the significant effort required in programming, testing and documenting the many features the **MIDAS** system offers. All of these individuals are still affiliated with our laboratory. The Laboratory Director is Prof. Robert Langridge. Thanks are due to many of our existing MIDAS users for countless suggestions over the years that have lead to improvements in the system. This work is supported by the National Institutes of Health, National Center for Research Resources (NCRR), RR-01081.

Because a substantial portion of the funding for our laboratory comes from a NCRR grant (RR-01081) from the National Institutes of Health, it is important that publications resulting from work using the **MIDAS** system or incorporating graphical images produced with **MidasPlus** acknowledge our laboratory. We ask that a statement similar to the following be used:

> Molecular graphics images were produced using the MidasPlus software system from the Computer Graphics Laboratory, University of California, San Francisco.

The article which describes the **MIDAS** system and should be included in your references is:

> T.E. Ferrin *et.al.*, ''The MIDAS display system'', *J. Mol. Graphics* **6**, 13-27 (1988).

We would also appreciate receiving two reprints of any publications resulting from your work with the **MIDAS** system.

(This page intentionally left blank)

Part I: Molecular Modeling with MIDAS

1. Important Concepts in MIDAS

This introductory section describes some concepts important to molecular modeling with MIDAS. Understanding these concepts and the philosophy of MIDAS is crucial to success, and all readers are encouraged to read this section thoroughly.

1.1. Getting Started

MIDAS is capable of displaying molecular structures from information contained in either a Protein Data Bank* (PDB) format file or a binary MIDAS database (created from a PDB file using the **midas.in** program). Both file formats have their advantages. Using PDB format files directly provides for maximum convenience, since only a single file needs to be manipulated, the file is readable by humans, and the **ribbon** command requires a PDB format file to work properly. On the other hand, MIDAS database files provide for minimum system startup time and minimum data storage requirements on disk. MIDAS databases consist of three files with a common root name but differing suffixes (*.ndx*, *.tpl*, and *.dat*). For instance, the MIDAS database referred to as *1gcn* would consist of the files *1gcn.ndx*, *1gcn.tpl*, and *1gcn.dat*. Which file format to use can usually be determined on a case-by-case basis. Large protein structures that are referenced frequently are probably best converted to a MIDAS database before being displayed, since this reduces waiting time. Small or medium sized proteins are probably best left in their original PDB format. The only functional difference between the two file formats occurs with the **ribbon** command, as this command operates only on PDB format files since it requires secondary structure information in the form of HELIX and SHEET records that are not stored as part of a MIDAS database. The following sections discuss the details of PDB file format and how to construct a MIDAS database using the **midas.in** program. Even if you only intend to use PDB format files in MidasPlus it is instructive to throughly read the following manual sections dealing with data formats and connectivity, since subtles in PDB file record formats are the most common cause of unexpected results when displaying models.

If you already have a PDB format file that is known to be correct, or you want to try out MidasPlus with one of the test files included with the software distribution, you may temporarily skip ahead to section 4, "Displaying the Model".

1.2. Building Your Own MIDAS Database

As discussed above, if the model you want to examine is not available in the */usr/mol/midas* directory, you may wish to prepare a MIDAS database from the coordinates. If the coordinates are in Protein Data Bank format, the **midas.in** program may be used to convert them to MIDAS database format. The MIDAS database produced consists of the three files model.*ndx*, model.*tpl* and model.*dat*, where *model* is the molecule name. When building a MIDAS database, a corresponding "template" file is required for each residue specified in the Protein Data Bank file. This template describes the connectivity of atoms in a residue. It is a map of how atoms are connected and which atoms act as links to other residues. *For the successful building of a MIDAS database, every residue name in the Protein Data Bank file must have a corresponding template file in the directory /usr/mol/models or the user's private template directory.* Templates are described in detail in the following section.

1.3. Connectivity and Templates

Many macromolecules in nature, such as proteins and DNA, are built from component molecules which are chained linearly into large structures. Amino acids and nucleic acids are the building blocks from which complex biomolecules are made. MIDAS uses the same approach in building molecular models. Small component molecules are chained together to build images of complex models. In MIDAS,

* Bernstein, F.C. The Protein Data Bank: a Computer Archive. J Mol. Biol. 112, 535-542 (1977). For information on obtaining copies of coordinates from the data bank write to Ms. F.C. Bernstein, Chemistry Department, Brookhaven National Laboratory, Upton, New York 11973 USA.

these building blocks are called *residues*.

Each residue is made up of *atoms* which are connected in a specific pattern. Each atom in the residue has a unique name, usually combining a key letter, such as C for carbon, N for nitrogen, etc. and additional digits to uniquely identify it. For example, atoms may be named CA, C1, C2, N1, O2, etc. Thus, a residue contains a fixed number of atoms with specific names.

The pattern of bonding between the atoms is termed *connectivity*. Each atom of a residue is connected to one or more atoms in the same residue. Since the atoms are uniquely named, the bonding can be defined in an unambiguous manner. The definition of a MIDAS residue includes both connectivity and atom naming information.

When residues are used to build models, the connectivity of atoms is the same for each occurrence of any given residue in the model. For example, if we build a residue named "gly", then each time a "gly" residue appears in the model protein, it will have the same basic pattern of connectivity of atoms and the same atom names.

MIDAS uses files called *templates* to name each residue's atoms and the connectivity of those atoms. Each template consists of a map which describes how the atoms are connected and which atoms link the residue to other residues. This includes the following information:

(1) the residue name,

(2) the starting atom of the residue (important for connecting this residue to the previous one),

(3) the ending atom of the residue (important for connecting this residue to the next residue), and

(4) connections between all atoms in the residue.

Template file names are typically the same as the residue name, but for the most part this is only a matter of convention. It is the template file name that is used to determine the connectivity and atom naming information when building MIDAS databases. The residue name within the file itself is used only for labeling information in the picture which MIDAS displays. The following example of a template instruction file for glycine illustrates these ideas:

```
RESIDUE      GLY
START N
DRAW  CA
DRAW  C
DRAW  O
MOVE  C
DRAW  OXT
END   C
```

Notice that:

(1) The residue has a name, GLY, a starting atom, N, and an ending atom C. Thus, if GLY were found in a chain of amino acids, this residue would be connected to the previous residue in the chain via atom N and to the next residue in the chain via atom C. All MIDAS databases which include this residue information will refer to this residue via the character string "GLY".

(2) If you follow the DRAW and MOVE instructions on paper with a pencil (starting at N, draw a bond to CA, draw a bond from CA to C, draw a bond from C to O, lift up the pencil and move it back to C, draw a bond to OXT), you will find a pattern of connectivity for glycine such as can be found in any standard biochemistry text.

(3) Templates specify connectivity only, and do not give any information about bond angle or bond length. *In MIDAS, connectivity is determined by the MOVE and DRAW instructions in the template files, not by the distance between atoms.* If the user provides coordinates for glycine in which the distance from the C to the O is 10 angstroms, MIDAS is constrained to draw the long bond because of the connectivity specification in the glycine template. It is the user (who hopefully knows some chemistry) who defines the templates and ultimately controls which atoms are connected.

(4) The observant reader may have noticed that the carboxyl acid group contains two oxygen atoms, OXT and O. If this *gly* residue is the terminal amino acid in a peptide chain, then both oxygen atoms are appropriate to the model. If, however, the residue appears within a sequence of amino acids, then only one oxygen is appropriate and the bond to the atom named OXT is not drawn but instead a bond is drawn between the "END" atom (in this example a carbon) and the "START" atom of the next residue (typically a nitrogen in the case of amino acids). MIDAS decides which bonds not to draw in this case by ignoring atoms which appear in the template but for which the user provides no coordinates. Thus, only if the user provides a coordinate value for the atom OXT will MIDAS draw OXT. If OXT does not appear in the template, however, providing coordinates for OXT is *not* sufficient for adding the additional atom since the associated connectivity information must also be provided. In other words, an atom must be included in the template in order to appear in the MIDAS model.

(5) Note that none of the hydrogen atoms contained within a real glycine amino acid are included in the above template specification. Historically hydrogen atom coordinates are not included in protein and nucleic acid structures because of the limited resolution provided by x-ray crystallography techniques. For structures where hydrogen atom information is available, uniquely named atoms (*e.g.* H1, H2...) can be added to the residue definitions and thus incorporated into the MIDAS model database in the same manner as other atoms in the template.

One can rightly conclude from this description that the construction of templates is an integral part of MIDAS modeling. Templates must be built with care to insure an accurate model.

Most commonly used templates have already been constructed and reside in a library accessible to MIDAS (*/usr/mol/models*). The naming conventions used in these templates are those specified by the Brookhaven Protein Data Bank and include amino acids, nucleic acids, and many prosthetic groups. Thus, for the most part all the residues needed to build a molecular model are already available. For the exception, however, the user must construct his or her own set of MIDAS templates and use these alone or in combination with the existing MIDAS library of templates. A program exists to automatically construct new template files when needed; see **gentpl** in Appendix 6 for details.

1.4. Coordinates

Since templates tell MIDAS only the pattern with which to connect atoms and nothing about their relative (coordinate) positions, the user must also provide information as to the spacial relationships of the atoms in the model. The three-dimensional position of the atoms is specified by *coordinate data* which consists of a Cartesian coordinate system x, y, and z value for each atom in the model (*i.e.* orthogonal angstrom unit coordinates). [MIDAS displays increasingly positive z values as nearer the viewer]. This data is typically derived from x-ray crystallography techniques or from theoretical calculations which generate coordinate data. MIDAS uses this data along with the connectivity information provided in the templates to construct an image of a three dimensional model. *Thus, the user provides two forms of data to MIDAS: first, the template or connectivity information for each residue type in the model and secondly, the coordinate data for each atom appearing in the model.*

Coordinate data must be provided to MIDAS in a specific format called Protein Data Bank format (PDB).[†] The Protein Data Bank at Brookhaven National Laboratory (BNL) is a clearing house for macromolecular coordinate data, and distribution tapes are written in a standardized format specified by BNL. A complete and concise description of the format is given in the document "Protein Data Bank Atomic Coordinate Entry Format Description", which is published and distributed by Brookhaven National Laboratory. A newsletter is also published on a periodic basis by BNL. The following section contains a shortened description of PDB format sufficient for use in creating MIDAS databases.

2. Protein Data Bank Format

The complete PDB file structure contains a wealth of information including source, journal citations, and identification of substructures such as disulfide bonds, helices, beta sheets, and active sites. Since the entire

[†] The U.C.S.F Computer Graphics Laboratory has adopted this format as a standard for all coordinate text files.

Protein Data Bank entry contains much more information on a macromolecule than is needed for model building, knowledge of a subset of the format is sufficient for MIDAS users. Users should bear in mind, however, that adhering strictly to the format specifications is key to successful model building. The modeling programs are very unforgiving about incorrect input formats, and much time and frustration can be saved by diligence in data preparation.

2.1. Description

Protein Data Bank format is a character oriented format which consists of lines of information in a file. One file generally contains enough information to characterize a single molecule or model. Each line of information in the file is called a *record*. There are usually several different types of records present in the same file, such as **ATOM** records, which contain coordinate values for atoms, **SSBOND** records, which contain disulfide linkage information, and **TER** records which signal the end of a chain of residues. These records are arranged in a specific sequence to characterize the molecule.

Protein Data Bank Record Types Recognized by MIDAS

Record Type	Data Provided by Record
ATOM	atomic coordinate record containing the x,y,z orthogonal angstrom coordinates for the given atom
HETATM	atomic coordinate record containing x,y,z coordinates for non-standard atoms. These record types are used by Brookhaven to distinguish standard residues, such a amino acids and nucleic acids, from non-standard groups, such as inhibitors, substrates, and saccharides. MIDAS does not distinguish between ATOM and HETATM records, so the user may use ATOM records exclusively, if desired.
SSBOND	defines disulfide bond linkages between amino acid residues. Notice that the templates as described thus far allow only for linkage to the next residue in the chain and to the previous residue in the chain. SSBOND records are special case links which are handled separately in MIDAS and bond rotations about these links are not allowed.
TER	indicate the end of a chain of residues. For example, a hemoglobin molecule consists of 4 subunit chains which are not connected. TER indicates the end of a chain and prevents a connection (line) to the next chain. This record type is also used to prevent connection of substrates to other displayed parts of the model.

The following table describes the format for each record type. The *record type* appears in columns 1 to 6 of each line of a PDB file. This *record type* determines the kind and format of information on the remainder of that line. Note that the data appears in specific *columns*. This refers to the spaces on the line in which the data appears. For example, in an ATOM record, the first four spaces contain the record type, "ATOM". The next two spaces are blank. The 7th through 11th spaces contain the atom serial number. The serial number is right justified, so if the serial number is "1", for example, the digit 1 will appear in the 11th space and the other spaces will be blank. For the atom with serial number "100" the number will appear in spaces 9 through 11 leaving space 7 and 8 blank. It is necessary to reproduce this format *exactly* in order for MIDAS to interpret properly the data. Any deviation is likely to cause errors preventing the successful building of a model database.

Protein Data Bank Format			
Record Type	Columns	Data	Justification
ATOM	1-4	"ATOM"	left
	7-11	Atom serial number	right
	13-16	Atom name	left*
	17	Alternate location indicator	character
	18-20	Residue name	right
	22	Chain identifier	character
	23-26	Residue sequence number	right
	27	Code for insertions of residues	character
	31-38	X orthogonal Å coordinate	right
	39-46	Y orthogonal Å coordinate	right
	47-54	Z orthogonal Å coordinate	right
	55-60	Occupancy	right
	61-66	Temperature factor	right
TER	1-3	"TER"	
	7-11	Serial number(optional)	right
	18-20	Residue name(optional)	right
	22	Chain identifier(optional)	
	23-26	Residue sequence number(optional)	right
HETATM	1-6	"HETATM"	
	7-66	same as ATOM records	
SSBOND	1-6	"SSBOND"	left
	8-10	Sequence number (optional)	integer
	12-14	Residue name (CYS)	right
	16	Chain identifier	
	18-21	Residue sequence number	right
	26-28	Residue name (CYS)	right
	30	Chain identifier	
	32-35	Residue sequence number	right

For those who are familiar with the FORTRAN programming language, the following format descriptions will be meaningful. For those users unfamiliar with FORTRAN, ignore this gibberish:

ATOM and **HETATM** Format (A6,I5,1X,A4,A1,A3,1X,A1,I4,A1,3X,3F8.3,2F6.2)

SSBOND Format (A6,1X,I3,1X,A3,1X,A1,1X,I4,4X,A3,1X,A1,1X,I4)

2.2. Examples of PDB Format

Consider the following simple example:

Glucagon is a small protein of 29 amino acids in a single chain. The first residue is the amino terminal amino acid, histidine, which is followed a serine residue and then a glutamine. The beginning portion of the PDB file appears thus:

*Atoms whose chemical symbols (as distinct from remoteness indicator) are one letter long are left justified in columns 14-16. Those which are 2 characters long (e.g. zinc symbol "ZN") are left justified starting in column 13. Refer to the Brookhaven document "Protein Data Bank File Record Formats" for details.

ATOM	1	C	HIS	1	49.169	26.701	10.917	16.00
ATOM	2	CA	HIS	1	50.197	25.578	10.784	16.00
ATOM	3	CB	HIS	1	51.312	26.048	9.843	16.00
ATOM	4	CD2	HIS	1	51.797	26.043	7.286	16.00
ATOM	5	CE1	HIS	1	49.691	26.152	6.454	17.00
ATOM	6	CG	HIS	1	50.958	26.068	8.340	16.00
ATOM	7	N	HIS	1	49.668	24.248	10.436	25.00
ATOM	8	ND1	HIS	1	49.636	26.144	7.860	16.00
ATOM	9	NE2	HIS	1	51.046	26.090	6.098	17.00
ATOM	10	O	HIS	1	48.241	26.524	11.749	16.00
ATOM	11	C	SER	2	47.713	29.006	10.110	15.00
ATOM	12	CA	SER	2	49.138	29.147	10.620	15.00
ATOM	13	CB	SER	2	49.875	29.930	9.569	16.00
ATOM	14	N	SER	2	49.788	27.850	10.784	16.00
ATOM	15	O	SER	2	46.740	29.251	10.864	15.00
ATOM	16	OG	SER	2	49.145	31.057	9.176	19.00
ATOM	17	C	GLN	3	45.406	27.172	8.963	14.00
ATOM	18	CA	GLN	3	46.287	28.193	8.308	14.00
ATOM	19	CB	GLN	3	46.489	27.963	6.806	18.00

Notice that each line or *record* begins with the record type, ATOM. The atom serial number is the next item in each record. Although each atom in the file is given a unique serial number, this information is not required by MIDAS.

The atom name is the third item in the record. Notice that the first one or two characters of the atom name consists of the chemical symbol for the atom type. All the atom names beginning with "C" are carbon atoms; "N" indicates a nitrogen and "O" indicates oxygen. The next character is the remoteness indicator code which is transliterated according to:

α	A
β	B
δ	D
ε	E
γ	G
η	H
ζ	Z

The last character of the atom name is a branch indicator, if required.

The next data field is the residue type. Notice that *each* record contains the residue type. In this example, the first residue in the chain is HIS (histidine) and the second residue is a SER (serine).

The next data field contains the residue sequence number. Notice that as the residue changes from histidine to serine, the residue number changes from "1" to "2". Two like residues may be adjacent to one another, so the residue number is very important for distinguishing between them.

The next three data fields contain the X, Y, and Z coordinate values, respectively. The final data item is an optional atom temperature factor.

The glucagon data file continues in this manner until the final residue is reached:

(see next page)

ATOM	239	C	THR	29	0.826	19.943	12.332	23.00
ATOM	240	CA	THR	29	2.014	19.761	13.283	21.00
ATOM	241	CB	THR	29	1.845	20.667	14.505	21.00
ATOM	242	CG2	THR	29	3.180	20.968	15.185	21.00
ATOM	243	N	THR	29	3.391	19.940	12.762	21.00
ATOM	244	O	THR	29	0.932	19.600	11.133	30.00
ATOM	245	OG1	THR	29	1.214	21.893	14.153	21.00
ATOM	246	OXT	THR	29	-0.317	20.109	12.824	25.00
TER	247		THR	29				

Note that this residue includes the extra oxygen atom, OXT, on the terminal carboxyl. The "TER" record terminates the amino acid chain.

A more complicated protein, fetal hemoglobin, consists of two amino acid chains (alpha and gamma) and two heme groups. The first ten lines of the PDB file for this molecule appear as:

ATOM	1	C	VAL	A	1	8.436	18.338	4.977	0.00
ATOM	2	CA	VAL	A	1	6.948	18.508	4.671	0.00
ATOM	3	CB	VAL	A	1	6.317	19.598	5.527	0.00
ATOM	4	CG1	VAL	A	1	6.959	20.999	5.376	0.00
ATOM	5	CG2	VAL	A	1	4.819	19.636	5.383	0.00
ATOM	6	N	VAL	A	1	6.280	17.225	4.929	0.00
ATOM	7	O	VAL	A	1	8.813	17.657	5.941	0.00
ATOM	8	C	LEU	A	2	11.156	20.058	5.187	0.00
ATOM	9	CA	LEU	A	2	10.715	18.872	4.330	0.00
ATOM	10	CB	LEU	A	2	11.420	18.882	2.960	0.00

This data file appears initially much the same as the file for glucagon with the exception that the fifth data field now contains the single character chain indicator. In this case, the chain indicator is "A", indicating the alpha chain of the hemoglobin molecule. This field was simply blank in the glucagon example. At the end of chain A, the heme group records appear:

ATOM	1059	CA	ARG	A	141	-8.044	12.831	-10.214	0.00
ATOM	1060	CB	ARG	A	141	-8.579	11.531	-9.580	0.00
ATOM	1061	CD	ARG	A	141	-8.727	10.045	-7.568	0.00
ATOM	1062	CG	ARG	A	141	-8.386	11.441	-8.054	0.00
ATOM	1063	CZ	ARG	A	141	-9.268	8.931	-5.414	0.00
ATOM	1064	N	ARG	A	141	-6.576	12.834	-10.275	0.00
ATOM	1065	NE	ARG	A	141	-9.095	10.056	-6.143	0.00
ATOM	1066	NH1	ARG	A	141	-8.602	8.795	-4.282	0.00
ATOM	1067	NH2	ARG	A	141	-10.097	7.962	-5.830	0.00
ATOM	1068	O	ARG	A	141	-7.591	15.139	-9.671	0.00
ATOM	1069	OXT	ARG	A	141	-8.973	13.984	-8.310	0.00
TER	1070		ARG	A	141				
HETATM	1071	C1A	HEM	A	1	9.434	9.659	-17.555	0.00
HETATM	1072	C1B	HEM	A	1	9.982	10.168	-13.320	0.00
HETATM	1073	C1C	HEM	A	1	7.911	6.467	-12.607	0.00
HETATM	1074	C1D	HEM	A	1	7.362	5.958	-16.842	0.00

The last residue in the alpha chain is an "ARG" (arginine). There is a coordinate value given for atom "OXT" in the terminal carboxyl. Notice that the "TER" record terminates the alpha chain, thus separating it from the heme group. This is important in preventing MIDAS from drawing a connection between the last atom in ARG and the first atom of the HEM residues. The atom number, residue type,

chain and residue number in the "TER" record are optional.

The heme group is a single residue made up of "HETATM" records. These record types are interchangeable with "ATOM" records in MIDAS. Note that the residue numbering begins again with "1" as the new chain begins.

At the end of the heme group associated with the alpha chain, the gamma chain begins:

HETATM	1110	O1A	HEM	A	1	9.575	12.251	-21.906	0.00
HETATM	1111	O1D	HEM	A	1	9.693	5.683	-22.895	0.00
HETATM	1112	O2A	HEM	A	1	9.716	14.207	-21.247	0.00
HETATM	1113	O2D	HEM	A	1	8.276	7.153	-23.229	0.00
TER	1114		HEM	A	1				
ATOM	1115	C	ACE	G	0	7.896	-18.462	-1.908	0.00
ATOM	1116	CH3	ACE	G	0	9.415	-18.301	-1.832	0.00
ATOM	1117	O	ACE	G	0	7.246	-18.839	-0.922	0.00
ATOM	1118	C	GLY	G	1	7.139	-19.112	-2.930	0.00
ATOM	1119	CA	GLY	G	1	5.904	-18.282	-3.283	0.00
ATOM	1120	N	GLY	G	1	7.354	-18.174	-3.077	0.00
ATOM	1121	O	GLY	G	1	7.026	-20.248	-2.448	0.00
ATOM	1122	C	HIS	G	2	10.808	-18.990	-3.748	0.00
ATOM	1123	CA	HIS	G	2	9.565	-19.224	-2.889	0.00

Once again the "TER" record signals the end of a chain. The new chain identifier is "G". The file continues in the same pattern as before until the entire gamma chain and its associated heme group have been specified.

Remember that the spacing of the data fields is crucial. Refer to the table given previously to determine the precise columns in which data *must* appear. If a data field does not apply, it should be left blank. For example, a protein which consists of a single amino acid chain has no chain identifier and thus column 22 is blank.

From this example, it is apparent that Protein Data Bank format relies on the concept of *residues* much in the same way as templates. The same rules apply for PDB residues and template residues. These can be summarized as:

(1) All atoms within a single residue must have unique names. For example, residue "VAL" may have only one atom named "CA". Other residues may also have a "CA" atom but not more than one "CA" may appear in "VAL".

(2) Residue names are a maximum of three characters long and uniquely identify the residue type. Thus, all residues of a given name in a file will be the same type residue and have the same structure. For example, residue "SER" has a certain connectivity specified in the template for "SER". Each occurrence of a serine residue in the Protein Data Bank file will conform to this pattern of connectivity. Atoms may be deleted in the PDB file, but may not be added if they do not appear in the corresponding template file. If a residue requires additional atoms or a different pattern of connectivity, such as in homoserine, a new template must be built and the PDB file must use a different residue name (*i.e.* other than "SER") in specifying the coordinates for the new residue.

2.3. Common Errors in PDB Format Files

If a data file fails to produce a model or produces an incorrect model in MIDAS, it is sometimes difficult to determine where in the hundreds of lines of data the mistake occurred. One common pitfall is failure to uniquely name all atoms within a given residue. Notice in the following example that two atoms in residue VAL are named CA.

ATOM	1	C	VAL	A	1	8.436	18.338	4.977	0.00
ATOM	2	CA	VAL	A	1	6.948	18.508	4.671	0.00
ATOM	3	CA	VAL	A	1	6.317	19.598	5.527	0.00
ATOM	4	CG1	VAL	A	1	6.959	20.999	5.376	0.00
ATOM	5	CG2	VAL	A	1	4.819	19.636	5.383	0.00
ATOM	6	N	VAL	A	1	6.280	17.225	4.929	0.00
ATOM	7	O	VAL	A	1	8.813	17.657	5.941	0.00
ATOM	8	C	LEU	A	2	11.156	20.058	5.187	0.00
ATOM	9	CA	LEU	A	2	10.715	18.872	4.330	0.00
ATOM	10	CB	LEU	A	2	11.420	18.882	2.960	0.00

This error often does not become apparent until a MIDAS database is built and the resulting model is found to be missing a "CB" atom. If the **midas.in** program is used to construct the MIDAS database, then the program will issue a warning message and use the first set of coordinates for "CA" and ignores all others. In other words, no atom will be drawn with the extra coordinate values. If the MidasPlus program is used to (implicitly) construct and simultaneously display the MIDAS database, then it will draw a bond to the closest atom and silently ignore the other set of coordinates. This is one of the reasons it is advisable to always construct a MIDAS database using **midas.in,** since this program is more rigorous in checking for input errors; see section 3.3 for more details.

In the following example, notice that the second residue (SER) appearing in the file is erroneously numbered residue 5. MIDAS will accept this input without complaint. The resulting MIDAS database will have residue 5 connected to residue 1 and residue 3. This is all well and good, but only if it is what was originally intended. If, however, residue number 5 was to appear between residue 4 and residue 6, then it should have appeared in that order in the PDB file. Thus, if one finds that residues are connected in an incorrect order, the ordering and not the numbering in the data file should be changed.

ATOM	1	C	HIS	1	49.169	26.701	10.917	16.00
ATOM	2	CA	HIS	1	50.197	25.578	10.784	16.00
ATOM	3	CB	HIS	1	51.312	26.048	9.843	16.00
ATOM	4	CD2	HIS	1	51.797	26.043	7.286	16.00
ATOM	5	CE1	HIS	1	49.691	26.152	6.454	17.00
ATOM	6	CG	HIS	1	50.958	26.068	8.340	16.00
ATOM	7	N	HIS	1	49.668	24.248	10.436	25.00
ATOM	8	ND1	HIS	1	49.636	26.144	7.860	16.00
ATOM	9	NE2	HIS	1	51.046	26.090	6.098	17.00
ATOM	10	O	HIS	1	48.241	26.524	11.749	16.00
ATOM	11	C	SER	5	47.713	29.006	10.110	15.00
ATOM	12	CA	SER	5	49.138	29.147	10.620	15.00
ATOM	13	CB	SER	5	49.875	29.930	9.569	16.00
ATOM	14	N	SER	5	49.788	27.850	10.784	16.00
ATOM	15	O	SER	5	46.740	29.251	10.864	15.00
ATOM	16	OG	SER	5	49.145	31.057	9.176	19.00
ATOM	17	C	GLN	3	45.406	27.172	8.963	14.00
ATOM	18	CA	GLN	3	46.287	28.193	8.308	14.00

Another common error often arises in the data entry process. Sometimes the letter "l" may be erroneously substituted for the number "1". This error has different repercussions depending upon in what data field the error occurs. If the letter "l" appears in the residue number, MIDAS does not complain, but then the letter "l" (rather than the number "1") must be used in all subsequent references to that residue. This can be very confusing, especially when all the other residues in the model are numbered. If the letter "l" appears in place of a "1" in the coordinate values, MIDAS accepts the input and sets those coordinate values equal to 0.000. Thus, if the user finds some atoms grossly misplaced in the MIDAS model, the corresponding error in the PDB file may be the use of "l" instead of "1". Such errors may be readily located if the text of the data file appears in upper case, since the editor may be invoked to search for all occurrences of the lower case letter "l".

3. Building a MIDAS Database

As discussed in sections 1.1 and 1.2, a MIDAS database is a group of binary files which contains all the information the MIDAS display program needs to display a model. These binary files contain both the coordinate and connectivity data of a molecule and are in a format readable only by MIDAS.

3.1. Using midas.in

To build a MIDAS database, the user must begin with a Protein Data Bank file as described in the previous section and use the program "**midas.in**" to then generate the binary database. Suppose, for example, the Protein Data Bank file is named *gcn.pdb*. An appropriate command for generating the MIDAS database would be:

midas.in -i gcn.pdb -o gcn

Assuming that all the residues in the data file *gcn.pdb* are standard amino acid (or nucleic acid) residues as defined by the PDB format, this command will produce a MIDAS database.

If **midas.in** is successful, the MIDAS database will appear in the user's directory as a group of three files. In this example these three files will be named:

gcn.ndx gcn.tpl gcn.dat

Notice that the initial part of the file names correspond to the output file name specified with the *-o* flag of the **midas.in** command (*i.e. gcn*). The suffixes, *.ndx*, *.tpl*, and *.dat* indicate a MIDAS index file, template file and data file, respectively. All MIDAS databases will contain at least three files having these three suffixes. Again, the files are in binary format and cannot be examined with, for example, a text editor.

Midas.in requires both a Protein Data Bank file and templates as input. If standard PDB residues are used, **midas.in** is able to use its own library of templates located in */usr/mol/models*. **Midas.in** automatically searches this directory for available templates matching the residue names given in the data file. If the residue and atom names used in the residues correspond to those found in the template, **midas.in** uses the template for connectivity. For example, if the PDB input file contains residue "VAL.tpl", then in order to use the library template:

(1) a template named "VAL" must be present in the directory */usr/mol/models*, and

(2) this template must contain atom names corresponding to those atom names found for "VAL" in the PDB file.

Now suppose a user's PDB file contains one or more residues which do not appear in the system template library, */usr/mol/models*. In this case, **midas.in** will try to build a template from the information provided in the PDB file itself.

Midas.in builds a template by calculating the distances between all the atoms in the given residue and determining from those distances which atoms should be connected. Using this information, the program generates a template which appears in the user's models directory. *All users should either define a MODELS environment variable or create a directory named* "models" *in their main home directory so that* **midas.in** *can create templates in this directory if needed.* A private "models" directory can be specified as follows:

(1) If the user has defined a pathname MODELS in his program environment (see *environ(5)* in the UNIX Programmer's Manual), this directory is searched first. The user may define MODELS in his *.login* file thus:

setenv MODELS *path_name*

(2) If the environment variable MODELS is not defined, a directory named "models" in the user's home directory is searched.

The templates which appear in the user's "models" directory consist of two files:

(1) a text file of connectivity instructions whose file name is suffixed with *.ins*, and

(2) a binary file whose file name is suffixed with *.tpl*.

Consider the following example: If **midas.in** were unable to find a template for residue VLX in its library of templates, it would generate a template and place two files, "VLX.ins" and "VLX.tpl" in the user's "models" directory. This template would then be used to generate the MIDAS database. In all subsequent runs of **midas.in**, the user's "models" directory is searched so that template "VLX" is accessed if residue VLX occurs in other models.

3.2. Building and Modifying MIDAS Templates

Occasionally it is necessary to modify existing templates. For example, **midas.in** sometimes generates templates with incorrect START and END atoms. This results in the use of the wrong atoms to connect the current residue to the previous and next residues in the chain. To correct this error, the user may edit the template directly. To do so, invoke the standard system text editor on the *.ins* file in the user's private "models" directory. Since *.ins* suffix files contain only simple character text they can be edited directly. (The *.tpl* suffix files contain binary data and cannot be edited with a text editor.) The name of the START atom and the END atom can be changed as needed; see section 1.3, "Connectivity and Templates", for additional information on the format of template files. For example, suppose **midas.in** generated the following template:

```
RESIDUE      FEA
START FE1
DRAW O
DRAW FE2
DRAW NA
DRAW NB
DRAW NC
END          NC
```

The start atom, *FE1* is correct but the connection to the neighboring residue should be made via *FE2* instead of *NC*. The file can be edited to read:

```
RESIDUE      FEA
START FE1
DRAW O
DRAW FE2
DRAW NA
DRAW NB
DRAW NC
END          FE2
```

Once the instruction file has been changed, the binary file must be updated as well. To do this, invoke the **maketpl** program:

maketpl -i FEA.ins -o FEA

This command uses the connectivity instructions in file *FEA.ins* to create a new MIDAS binary template file, *FEA.tpl*. The *.ins* file is optimized to produce the minimum number of MOVE and DRAW commands and may be updated by **maketpl** as well. Note that this is done in the user's "models" directory, *not* in the MIDAS system library of templates. The system library templates may not be changed by general users, but may be copied into a user's private "models" directory and modified.

After the template has been modified using a text editor and a new *.tpl* file generated by the **maketpl** program, **midas.in** can be run again so that the new template is incorporated into the MIDAS database. This is done using the same **midas.in** command as before. The corrected template is automatically included in the new MIDAS database. Remember that the MIDAS database contains all the information needed to display a model. Thus, the templates that existed at the time the database was generated are the

ones incorporated into the database. Merely modifying a template does not change the MIDAS database contents.

Sometimes **midas.in** is unable to generate a template from the PDB file because the atoms in the file are too far apart to determine the correct connectivity. An error message appears indicating "unreachable atoms". **Midas.in** uses a standard list of atomic radii for determining connectivity. This list is located in */usr/local/lib/midas/connect.tpl*. Atoms falling within the standard distances in the list are joined. Any atoms which are not joined to at least one other atom or any portion of the residue not connected to the main body of the residue, result in the "unreachable atoms" error message.

One approach to solving the "unreachable atoms" problem is changing the standard distances to larger values so that atoms which failed to bond before can be appropriately connected. To do this one can:

(1) Make a copy of the atomic radii file in your own private directory using the command:

cp /usr/local/lib/midas/connect.tpl connect

(2) Edit this connect file to increase the atom radii so that the appropriate connections can be made.

(3) Use the **gentpl** program to create the new template. For example, if the residue for which we want to build a template is VLX and this residue occurs in the PDB file *glx.pdb*, the command used is:

gentpl -r VLX -i glx.pdb -c connect

where *connect* is the name of the atomic radii file just edited. **Gentpl** searches through the PDB file for the specified residue and uses the atomic radii in the *connect* file to generate the connectivity of atoms and the corresponding MIDAS template.

If this method fails to produce an accurate template, the user must resort to creating the template instruction file with a text editor. Refer to section 1.3 of this document, "Connectivity and Templates", for a description and example of the connectivity instruction file. The manual page for the **maketpl** command is also helpful (see Appendix 6). Look at some of the *.ins* files in the template library (*/usr/mol/models*) to use as examples. Do not worry about the order in which the DRAW and MOVE commands appear in your instruction file as long as they correctly specify the connectivity and all atoms are connected to at least one other atom in the residue. The **maketpl** program will optimize the "connectivity" of atoms for you. To generate the required binary template file (*e.g.* VLX.tpl), run **maketpl** as before.

3.3. More on PDB format versus MIDAS databases

Now that the reader has a good understanding of what comprises a PDB format data file and how a MIDAS database is built, additional details about differences between using a PDB format file directly within MidasPlus and using a MIDAS database can be presented.

When MidasPlus uses a PDB file to display an image, it uses either the CONNECT records stored within the PDB file or, if these records are absent, it attempts to determine connectivity by utilizing *.ins* connectivity files based on residue type. These connectivity files must reside in either the user's current directory or */usr/mol/models*. If neither means of determining atom connectivity exists, then MidasPlus will determine feasible atom connections based on atom type and bond lengths obtained from tables taken from the CRC Handbook of Chemistry and Physics. If atoms still exist that cannot form bonds of "reasonable" length, then MidasPlus will form a connection to the closest atom within the residue under consideration, regardless of the length of the resulting bond.

Midas.in determines its connectivity information first from *.ins* connectivity files. If these files do not exist, then the **gentpl** program is invoked and connectivity is determined based on atomic radii as specified in the file */usr/local/lib/midas/connect.tpl*. If two atoms are more distant from one another than the sum of their radii, then **gentpl** reports that these atoms cannot be connected and the database building process terminates.

In summary, both MidasPlus and **midas.in** behave in a similar and predictable fashion in the normal situation where atom coordinates are properly specified and *.ins* and/or PDB CONNECT records exist. If atom coordinates are incorrectly specified, either intentionally or due a error in the PDB database, then

MidasPlus and **midas.in** behave differently; **midas.in** will report an error in this case while MidasPlus will always form a bond to a neighboring atom.

4. Displaying the Model

Regardless of which approach is used to provide the required input coordinate data (PDB file or MIDAS database), the main MidasPlus display program is used to display the resulting model. To invoke the MidasPlus display program give the command:

midas *or* **midas -f**

If the **-f** flag is given, MIDAS will use the entire screen for displaying the model, otherwise the user indicates the size of the MIDAS window by depressing the left mouse button to indicate one window corner, then dragging the mouse to the location of the opposite window corner and releasing the mouse button. This is the same mechanism used by standard IRIS applications to allow the user to indicate a desired window size. The MIDAS window may be resized at any time by using the standard IRIS technique of "grabbing and pulling" a window corner with the mouse.

After a few seconds delay, the graphics screen should display a prompt, "COMMAND:", on the lower left indicating MIDAS is ready to receive input from the user. The box in the lower right corner of the MIDAS window is called the *control panel*. It is described in detail under **Manipulating the Model**.

Once the "COMMAND:" prompt appears, characters typed on the keyboard are interpreted directly as commands to MIDAS (if you started MIDAS within a window, you must select the window first by positioning the mouse cursor within the window).

The first step in displaying a model is to "open" the database. If the name of the PDB file or MIDAS database is *gcn*, the command used to open the database is:

open 0 gcn

If the *gcn* file is a MIDAS database, then the actual files associated with this database are "gcn.tpl", "gcn.ndx", and "gcn.dat". The *database* name is "gcn" (without the suffixes), and this name is used to open the database. The database name may also be a pathname to a database in another directory. If both a MIDAS database and a PDB format file exists and the user does explicitly specify that the PDB file is to be used (via the "pdb" option to **open**), then MidasPlus will use the MIDAS database for its input. The 0 (zero) in the above example is the model number. Since more than one model may be open at a time in MIDAS, a model number (a digit between 0 and 11, inclusive) must be provided by the user. If a second model, say *gcx*, were to be opened now, the command would be:

open 1 gcx

This model is now referenced as model number one in MIDAS.

4.1. Manipulating the Model

After execution of the **open** command completes, the model should appear on the graphics display screen. The left and middle mouse buttons allow direct manipulation of the selected model(s) (the right button calls up a system menu). Depressing and holding the left button controls model rotation. While the button is held down, a dashed blue circle is displayed on the screen. Moving the mouse outside the circle results in model rotation about the z axis. The area inside the circle is a "virtual trackball"; when the mouse is within this area it "grabs" the trackball and rotates it. The model rotates as if it were inside the trackball. After familiarity is gained with this interaction method, display of the dashed circle can be suppressed, if desired, by typing **~set showsphere**. The "icon" used for the mouse cursor changes when the left mouse button is depressed and also changes dependent on whether or not the mouse is inside or outside the trackball circle, regardless of whether or not the actual circle is displayed. Thus the mouse icon can be effectively used as a visual feedback clue to indicate the type of rotational motion currently in effect. Depressing and holding the middle mouse allows dragging the model(s) left/right and up/down. In other words, the middle button controls global translation of the selected model(s). Depressing and holding

the left and middle mouse buttons simultaneously results in z translation of the selected model(s).

The *control panel*, located in the lower right corner is divided into two parts.

The upper part shows a stylized representation of the molecule(s), as viewed from the side. Also represented are the viewer's eyepoint (the box at the junction of the convergent horizontal lines) and the hither and yon clipping planes (the left and right vertical lines, respectively). A typical side-view area is shown in Figure 1.

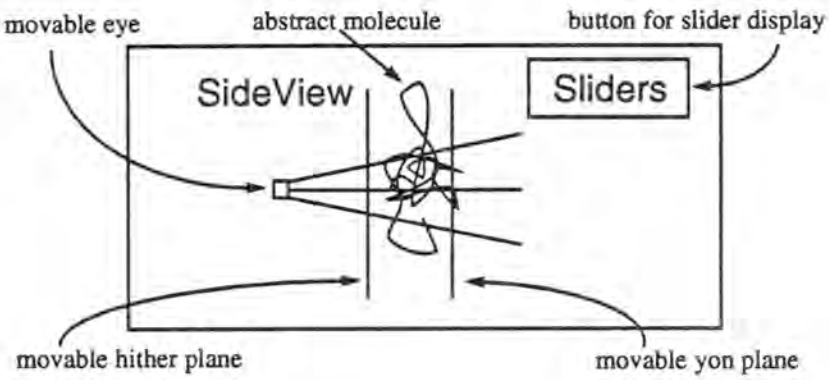

Figure 1. Side-view

The eyepoint and/or clipping planes can be moved by positioning the mouse over them, depressing the left button, and dragging the item. Moving the eyepoint changes the apparent size (scale) of the image. Clicking on the box labelled *Sliders* replaces the side-view with 8 rectangular boxes, called *pseudo-sliders* or *sliders*, numbered from 0 to 7. Each slider can be assigned a function, such as model rotation or translation, clipping plane manipulation or bond rotation adjustment. See the **assign** command in Part III for further details. To utilize a slider function, click and hold down the left mouse button over the slider labeled with the function of interest. For example, to translate a model along the z axis, click over the slider labeled "z tran". The left half of the slider moves the model further away, while the right half brings it closer. As one holds the button down, the rate of the translation, and even its direction, can be adjusted by moving the mouse within the slider region. Clicking on the *SideView* box returns you to the side-view representation.

The lower part of the control panel contains twelve small boxes, numbered from 0 to 10 plus one marked "All", referred to as *pseudo-switches* or simply *switches*. These control whether the corresponding model number is *selected*. Only models that are selected can be manipulated. When a model is first opened, it is automatically selected and the corresponding switch is highlighted. It may be necessary to deselect a model if one model is to be moved relative to another. Clicking over a switch toggles the selection mode of the corresponding model. Clicking over "All" selects all open models. Clicking on "All" again will return to the previous selection state.

In order to provide "backwards" compatibility with previous versions of MIDAS, an alternative interaction method is available if one types **set sphere.** *This interaction mode is less intuitive than the virtual trackball described above and new users are discouraged from using it.* To get the desired motion, hold down the appropriate combination of mouse buttons as shown in Figure 2 and move the mouse to get a corresponding change in model orientation. For X/Y translation and rotation, both left-right and up-down mouse motion is effective. For Z translation and rotation, only left-right mouse motion is effective. Since this method uses the right mouse button for translation control, the system menu is unavailable. *This is a limitation inherent with this mode of interaction.*

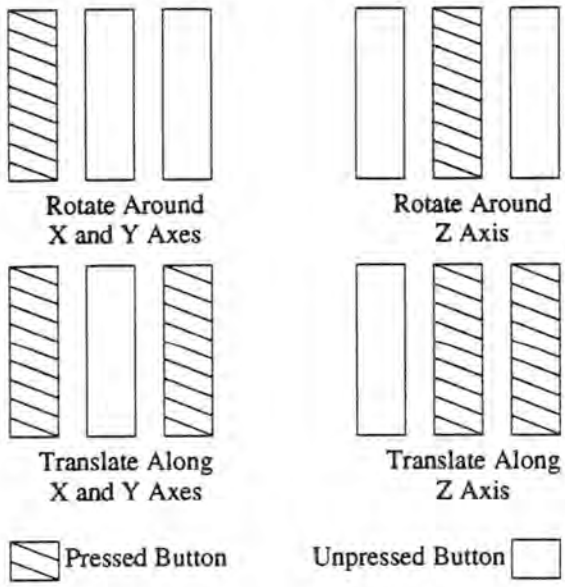

Figure 2. Mouse button interpretation when **not** using virtual trackball interaction mode

From this point on, there are many MIDAS commands available for manipulating the model. These are described in detail in Part III, "Command Reference Guide", of this document. The following categories of commands are provided to help you get started:

MIDAS Commands by Category	
For information on:	The pertinent commands are:
Adding groups, new residues	**addgrp addaa delete swapaa swapna**
Bond rotations	**rotat brotat reverse assign angle**
Coloring models	**color intensity set**
Coordinate recovery	**fix getcrd save write**
Interactive manipulation of models	**assign select**
Labeling model components	**label rlabel**
Making movies	**rock roll set sleep source wait**
Molecular surface display	**surface vdw vdwopt**
Selective display of model components	**chain display show zone**
Stereo	**set**

5. On-line Help

MIDAS features an on-line help system. All commands documented in Part III of this manual are also available on-line.

help *command*

will produce a short synopsis of the specific command in question. The command

help

without any arguments produces a list of all available MIDAS commands. The help facility is very useful for both the novice and experienced MIDAS user and alleviates constantly referring the User's Manual for every command.

Part II: Advanced Concepts: Surfaces and Hydrogen Atoms

This section describes some additional features of MIDAS and some ancillary programs which may be used in conjunction with MIDAS for specific modeling problems. Included are descriptions of:

- Calculation and display of a molecular surface
- Calculation of electrostatic potential for a surface

Note: The descriptions include sample command lines. In these examples the model name used is *dfr*, an acronym for dihydrofolate reductase, a system which has been extensively modeled. The file names containing 'dfr' should be substituted with the user's own file names when running commands.

6. Molecular Surfaces

MIDAS represents molecules by drawing three dimensional "wire frame" models on the screen. Users often want to characterize the surfaces of these molecules as well. This can be accomplished in several ways within MIDAS: the two basic types are space filling "solid" models and surfaces represented by a densely spaced dot matrix. Surfaces represented by dot matrices offer the most utility, since they can be manipulated in real time in the same way as wire frame models. Space filling models, on the other hand, offer shading and shadows and produce photographic quality images; these surfaces are generated using the **conic** command (see Appendix 6 for full details).

There are two different types of dot matrix surfaces available:

(1) A *vdw* or van der Waals surface uses the van der Waals radii of the atoms in the model. MIDAS assigns a van der Waals radius to each atom according to its atom type as determined by the Protein Data Bank naming conventions. (*e.g.* if the first letter encountered in atom name is C, then the atom is carbon, and so forth for N (nitrogen), O (oxygen), etc.) MIDAS creates a surface around each atom, clipping off areas of overlap to create a Corey-Pauling-Koltun (cpk) type model.

(2) A *solvent accessible* surface may be calculated for any molecule. This surface is defined by rolling a theoretical water "probe" around the van der Waals surface of the molecule and using the contact reentry points to determine the surface. This type of surface provides a smooth surface, free of the "seams" between atoms, since the surface is determined by the water molecule.

These two types of molecular surface representation differ significantly in their speed of generation and utility. Vdw surfaces are generated quickly and do not break or "tear" with bond rotations. Solvent accessible surfaces take approximately 100 times longer to compute, usually overnight for large molecules, and the surface tears with bond rotations. Users who choose to make solvent accessible surface calculations should have good reason to do so.

6.1. VDW Surfaces

The **vdw** command in MIDAS will display the van der Waals surface for all atoms of a MIDAS model or a subset of atoms as described in the "Command Reference Guide" section of this document. The user selects those atoms, residues, and models for which the surface is displayed.

Often the user does not want the surfaces of all atoms, both interior and exterior, displayed, but rather only the surfaces of those atoms which comprise the surface of the molecule or the surface of an active site. It can be a tedious process in MIDAS to select each of these surface atoms individually or even residue by residue. Both the MIDAS command **zone** and the MIDAS utility program **irs** provide ways to select specific sites of interest and/or remove the surfaces of interior atoms.

Example 1: Interior atom removal

Given a MIDAS database, *dfr*, the user wishes to display surfaces only for those atoms on the exterior surface of the molecule. The command:

irs -i dfr -o cmds

will do two things. First, it will modify the MIDAS database, *dfr*, such that the vdw surfaces for those atoms on the exterior surface of the molecule are displayed. Thus, when the user displays the model with MIDAS after running the **irs** program, these surfaces will be displayed and the surfaces for all other atoms are turned off. Secondly, **irs** records the MIDAS commands used to display the surface atoms in the named file *cmds* (the user may, of course, substitute a file name of his own choosing). Thus, if the user modifies the pattern of surface display in the MIDAS database and later wishes to display only exterior surface atoms, this file can be run in MIDAS (using the **source** command) to display the surface for this set of atoms without having to rerun the **irs** program. Note the commands in the *cmds* file assume that *dfr* is model number 0 (zero) in the MIDAS session. If the user wishes to use another model number, the *-m* flag is used with a model number argument. For example:

irs -i dfr -o cmds -m 1

causes the commands in the *cmds* file to reference model number 1.

Example 2: Site selection

Suppose the user wants to display the surfaces of only those atoms within a certain distance of a particular set of atoms or coordinate positions, as in determining an active site of a molecule. This can be accomplished by supplying **irs** with a MIDAS database of test coordinates to be used to select the site. For example, if such a MIDAS database were named *probe*, then the command:

irs -i dfr -o cmds -t probe

modifies the MIDAS database *dfr* such that surfaces are displayed for only those atoms within 8 angstroms of any atom (coordinate value) in data base *probe*. If the user wishes to change the test radius from 8 angstroms to 12 angstroms, for example, the command line is:

irs -i dfr -o cmds -t probe -r 12

See the **irs** manual page in Appendix 6 of this document for additional details. Also, the **zone** command described in Part III provides another method of achieving similar results from directly within MidasPlus.

6.2. Solvent Accessible Surfaces

Creating a solvent accessible surface for display with MIDAS is a multi-step process and generally requires an overnight run of the **ms** program.[†] Because of the time required to generate the surface, it is important to prepare files carefully to prevent time-consuming errors.

(1) Begin with a model which has been displayed and is known to be correct. At UCSF, the coordinates may be in either Protein Data Bank format or a MIDAS database.

(2) The surface may be calculated for the entire model or specific sections. To select specific sections, prepare a file containing the residues and/or atoms of interest. The format for this file is a series of lines containing the residue name, residue number and atom specification:

(see next page)

[†] The **ms** program was written at U.C.S.F. by Michael Connolly. The program is available from Quantum Chemistry Program Exchange (QCPE) at: Chemistry Department, Room 204, Indiana University, Bloomington, Indiana 47401. The exact commands for using the QCPE version of **ms** may differ somewhat from those given here.

Site Selection File Format		
Columns	Data	Justification
1-3	Residue name	left
5-8	Residue number	right
10-12	Atom specification (atom name, '*', 'FRM', or 'TO')	left

As an example, a file appearing as:

```
ASN   15   FRM
LYS   21   TO
HIS  123   *
GLU  141   CB
GLU  141   CG
```

selects residues 15 through 21, inclusive, residue 123 (the "*" selects all the atoms of the residue), and atoms 'CB' and 'CG' of residue 141. The surface is calculated only for these specified atoms and residues.

Usually the selected portion of the model is an active site. The user may determine which atoms are of interest by one of two methods:

1. display the model and choose the atoms visually

2. use **irs** to select atoms within a given radius of a ligand.

(1) Use the **ms** program to generate the surface. An example command for creating a surface is:

ms *hb1* **-a -m -d 0.5 -g** *logfile* **-i** *site* **-o** *hb1.ms*

where:

hb1	is the name of the MIDAS database.
-a	indicates that all atoms are to be included in the calculation. If this flag is missing only amino acids are included in the calculation.
-m	indicates that the input file (*hb1*) is a MIDAS database.
-d *0.5*	indicates that the density of points on the surface.
-g *logfile*	directs informative messages from the program to the file *logfile*.
-i *site*	directs **ms** to calculate the surface for the entire molecule but report the surface only those atoms and residues selected in the file *site*.
-o *hb1.ms*	directs the output from the program (*i.e.* the surface) to the file *hb1.ms*.

At UCSF, the **ms** program should always be run in the background using the **submit** command. Thus, the command line for the above example becomes:

submit ms *hb1* **-a -m -d 0.5 -g** *logfile* **-i** *site* **-o** *hb1.ms*

If the coordinates are in Protein Data Bank format rather than a MIDAS database, the *-m* flag is deleted:

submit ms *hb1* **-a -d 0.5 -g** *logfile* **-i** *site* **-o** *hb1.ms*

where the input file, *hb1.pdb*, is a Protein Data Bank format file.

(2) The progress of the **ms** program may be monitored in two ways:

1. Use the **ps** command to see if the program is running.

2. Read the logfile (*i.e.* the file name given with the *-g* flag) and the *submit.out* file created by **submit**. If the program is left to run overnight it is a good idea to check these two files before leaving the laboratory to make sure the program didn't run into immediate difficulty and stop.

The completed **ms** output file is quite large, so be sure adequate disk space is available.

(3) Use the **makesurf** program to convert the **ms** output into a MIDAS surface database. Continuing the above example, the appropriate command is:

makesurf -i *hb1.ms* **-o** *hb1.s*

which uses the **ms** generated file *hb1.ms* as input and creates a MIDAS surface database named *hb1.s*. Like MIDAS databases, this surface database is a group of binary files. In this example, the file names generated are:

hb1.s.dat
hb1.s.ndx
hb1.s.tpl
hb1.s.srf

Note that the **makesurf** program requires MIDAS templates for all residues found in the **ms** file. If the model has been previously displayed with MIDAS, then the templates exist and are accessible. If not, **midas.in** should be run with the PDB file as input to generate the needed templates.

Most users find it convenient to name surface databases using the same name as the corresponding coordinate database with a ".s" suffix. This eliminates ambiguity in associating a MIDAS database with its companion surface database.

(4) To display the surface, invoke the **midas** display program. Open the coordinate database first using the **open** command as described in the "Command Reference Guide" section of this document. Then using the same model number, open the surface database. For example:

open 1 hb1
open s 1 hb1.s

opens database *hb1* as model 1 and associates the surface *hb1.s* with it. To display the surface, use the **surface** command:

surface #1

7. Electrostatic Potential Molecular Surfaces

Users who wish to display electrostatic potential molecular surfaces should run the **ms** program as described above with an additional flag, *-n*. This flag generates surface normals in the **ms** output file. The **ms** output file may be used to generate the electrostatic potential of the surface using the program **esp**.

Esp requires a **ms** output file and a MIDAS database as input. Assuming the **ms** output file name is *2hbl.ms* and the MIDAS database name is *2hbl*, the command to calculate electrostatic potential is:

esp -i 2hbl.ms -o 2hbl.srf -a 2hbl

The program generates a MIDAS surface database, which in this example is named *2hbl.srf*.

The atomic charges for the various residue types are held in the system file */usr/local/lib/midas/charges.esp*. The user may substitute his or her own file of charges if desired. Use the system file as a format guide to generate the appropriate file. Include this file in the command line:

esp -i 2hbl.ms -o 2hbl.srf -q chrgs -a 2hbl

where *chrgs* is the name of the charges file. The *-q* flag and charges file must precede the name of the MIDAS database in the command line. Note that the electrostatic surface potential computed by **esp** depends crucially on the charge data in the charges file, */usr/local/lib/midas/charges.esp*. Users should verify the data given in this file is correct for their particular application.

The **esp** program has several other options which are described in the **esp** manual page; see Appendix 6 of this document.

8. Modeling Hydrogen Atoms

8.1. Hydrogens in Protein Data Bank Files

Users who have coordinate values for hydrogen atoms may include those values in the PDB data file using the Brookhaven Protein Data Bank convention for hydrogen atoms. The conventions for hydrogen atoms in PDB files are as follows:

(1) Hydrogen atoms appear as ATOM records following the ATOM records of all other atoms of a particular residue.

(2) The name of each hydrogen atom is determined by the name of the atom to which it is connected:

The first space of the name (column 13) is an optional digit to be used if two or more hydrogens are attached to the same atom.

The second column, 14, is used for the chemical symbol, "H".

The next two columns contain the remoteness indicator (one or two characters) of the atom to which the hydrogen is attached.

For example,

ATOM	1	N	VAL	1	0.330	15.770	15.090	3.30	1.46
ATOM	2	CA	VAL	1	1.650	16.390	15.360	0.96	1.50
ATOM	3	C	VAL	1	2.670	15.670	16.230	3.18	1.43
ATOM	4	O	VAL	1	3.170	16.250	17.200	-0.72	1.48
ATOM	5	CB	VAL	1	1.760	17.680	16.180	1.77	1.47
ATOM	6	CG1	VAL	1	3.120	18.310	15.900	4.31	1.50
ATOM	7	CG2	VAL	1	0.630	18.680	15.930	2.42	1.51
ATOM	8	D	VAL	1	-0.250	15.310	14.410	4.96	1.46
ATOM	9	HA	VAL	1	2.200	16.520	14.420	-2.30	1.50
ATOM	10	HB	VAL	1	1.750	17.420	17.260	-2.08	1.62
ATOM	11	1HG1	VAL	1	3.210	18.510	14.820	-0.21	1.60
ATOM	12	2HG1	VAL	1	3.230	19.250	16.460	-0.92	1.57
ATOM	13	3HG1	VAL	1	3.910	17.600	16.200	-1.79	1.56
ATOM	14	1HG2	VAL	1	-0.270	18.120	15.640	-3.19	1.55
ATOM	15	2HG2	VAL	1	0.440	19.240	16.860	-0.54	1.56
ATOM	16	3HG2	VAL	1	0.940	19.370	15.130	2.66	1.57

Note in this example that:

- All hydrogens appear after the other atoms of the residue.
- Atom 9, "HA" is attached to atom 2, "CA". The remoteness indicator, "A" is the same for both these atoms.
- There are three hydrogen atoms connected to "CG1". These three all have the same remoteness indicator, but contain a distinguishing digit in column 13. Thus, each has a unique name.

- It is not necessary to use a digit as a prefix to the atom name when only one hydrogen is attached to a given atom.

9. Making Videos

Many MIDAS commands are useful for producing video tapes and movies, and commands such as **move**, **rock**, **roll** and **turn** have optional parameters for the number of image update frames over which to carry out the command execution and the number of frames to wait before beginning execution. There are no stop motion "frame by frame" movie features (*i.e.* image generation synchronized with a camera).

MIDAS videos are produced from "scripts" or command files which can be created by the user with a text editor or recorded during a MIDAS session using the **record** command. By using command scripts the disk space required for files is minimized. To review a script give a **push** command, run the script, and do a **pop** to return to the original picture. Note that MIDAS cannot be stopped and restarted in the *middle* of a script.

PART III: Command Reference Guide

10. Referencing Models, Residues and Atoms

10.1. Models, Residues and Atoms

MIDAS uses a hierarchical command syntax developed in 1980 by the UCSF Computer Graphics Laboratory for referencing models, residues and atoms. In each MIDAS model, molecules are made up of residues. The residues are chained together in a specific sequence to form the molecule. Each residue is made up of atoms organized according to the coordinates and connectivity information stored in the database. This scheme reflects nature's organization of biomolecules: amino acid chains make up protein molecules and nucleotide chains make up DNA molecules.

MIDAS allows the user to display multiple models (molecules) simultaneously. These molecules are assigned a model number by the user with the **open** command or a default by MIDAS. Each molecule consists of one or more residues, each of which has a unique associated residue number according to its location in the residue sequence. The atoms which make up the residue each have associated atom types which are unique within any single residue. Thus, any displayed atom may be uniquely described by its model number, residue number and atom type.

The residue names and atom types are determined at the time the database is built, and generally match the standard Brookhaven Protein Data Bank (PDB) residue and atom names. The symbols for these reference levels are defined as follows: †

Atom Specification Symbols

Symbol	Reference Level	Definition
#	model number	a number assigned to the displayed model by the user via the **open** command
:	residue	a residue type (standard Protein Data Bank abbreviation) *or* residue sequence number *or* range of sequence numbers
@	atom	an atom name (standard Protein Data Bank abbreviation)

The following examples illustrate the use of these symbols for referencing models, residues and atoms. Note that the lack of either a residue specifier or an atom specifier or both is interpreted to mean "all" units of the associated reference level.

#0	(all atoms in all residues in model 0)
#1:50	(all atoms of residue 50 in model 1)
#0:12@CA	(alpha carbon of residue 12 in model 0)

Groups of atoms or residues may be specified. For example:

† Note: A summary of all special symbols described in this section appears in Appendix 3 of this document.

#0:12@CA@N	(alpha carbon of residue 12 in model 0 and nitrogen of residue 12 in model 0)
#0:12@CA,N	(alpha carbon and nitrogen of residue 12 in model 0)
#1:LYS	(all lysine residues in model 1)
#3:45-83	(range of residues 45 through 83 in model 3)

Notice in the above example that the two atoms, 'CA' and 'N', may be delimited by either a comma or the symbol '@'. MIDAS interprets each atom name specified with the '@' symbol as a separate entity. Thus, the @N is interpreted to mean "use the preceding (most recent) residue and molecule information to determine the atom." The comma delimiter indicates a group of atoms which are not single entities. Thus, '@CA,N' means "use the preceding (most recent) residue and molecule information to determine *both* carbon and nitrogen." Thus, in specifications where the *order* of appearance of the atoms is significant (*e.g.* the **match** command), the separate entity notation should be used. For residues, the same hierarchical notation is followed. For example, for atoms on different residues but same model:

#1:12,14@CA	(alpha carbon in residue 12 and residue 14)
#1:12:14@CA	(all atoms in residue 12 and alpha carbon in residue 14)
#1:12-20@CA:14@N	(alpha carbon in residues 12 through 20 and nitrogen in residue 14)
:LYS@CA	(alpha carbons in all lysine residues)

In the example above, the first statement gives two residues which make up a single residue specification. Therefore, the carbon atoms in both residues are selected. In the second example, the entire residue 12 and only the carbon in residue 14 are selected.

10.2. Wildcard Symbols

The global wildcard symbol "*" matches all atoms in a residue or all residues in a model. It stands alone as a symbol, *i.e.* it cannot be used to match parts of names or sequences, such as G* or *A. To do that, use the "=" wildcard character. For example, **color red @c=** means to color all atoms whose names begin with the letter "c". This works for residue names too (but not residue sequence numbers). The single character wildcard symbol "?" is used to select atom *names* and residue *names* whose names follow patterns. "?" cannot be used to match sequence numbers. For example:

#1:12@*	(all atoms in residue 12 of model 1)
#0,1,2:50-*@CA	(all alpha carbon atoms in residues 50 to the end of models 0, 1 and 2)
#2:G??	(all three character residue names which begin with the letter 'G' in model 2)
#0:*@H@H?@H??	(all hydrogen atoms with one, two or three letter names in model 0)

The percent symbol, "%", may be used to specify every *n*th item where *n* is an integer. For example, #1:*%5 selects every fifth residue in model 1; #1:HIS@*%4 selects the fourth atom of each HIS residue.

10.3. Atom Properties

Atom properties are specified with the / operator. The currently supported properties include **display**, **label**, **vdw**, **surface**, and **visible**. The last property is true if an atom actually appears on the screen. If a property name is preceded by a !, it means the atom must *not* have that property. The following specifier selects all atoms named CA which appear on the screen but are not labeled:

@ca/visible,!label

10.4. Zone Specifiers

Zone specifiers are used to select atoms and residues that are within a given distance of the referenced atom(s). **z<** and **zr<** specify all *residues* within the given distance from the referenced atoms. **za<** specifies all *atoms* within the given distance. **z>**, **zr>**, and **za>** yield the complementary set to their '<' counterpart. For example,

#1 za<10

selects all atoms within 10 angstroms of model 1.

10.5. Temperature Factors and Electrostatic Potentials

Atoms may be selected by temperature factor and electrostatic potential using the "<" and ">" symbols. Electrostatic potential selection requires the symbol "e>" or "e<" to select potentials above or below a specified value, respectively. Temperature factor selection requires the symbol "b>" or "b<" to select factors above or below a specified value, respectively. These symbols follow the *entire* atom specification and *must* have a preceding blank. For example,

#1:HIS b>25.0	(all atoms in histidine residues in model 1 which have temperature factors exceeding 25.0)
#1 e>10 e<20.0	(all surface points in model 1 with potentials between 10 and 20 kcal/mole)

Note that electrostatic potentials only apply to surface points and are calculated and incorporated into the MIDAS database using the **esp** utility program; see Appendix 6 of this document for details.

10.6. Atom Intersections

Intersections of groups of atoms are handled with the **&** operator. For example, one may want all atoms in model 1 which are within 10 angstroms of model 0:

#1 & #0 z<10

10.7. Atom Picking

Atom picking allows a user to select atoms using the mouse instead of typing in the atom names on the keyboard. This is useful for identifying atoms whose names are not known and for "non-typists". To use this mode, type the MidasPlus command substituting the symbol "+" in place of each desired atom specification. When you type the "+", the cursor will change shape into a picking arrow. Use the mouse to move the picking arrow displayed on the screen to the desired atom and press the left mouse button. MidasPlus substitutes the name of the selected atom for the first occurrence of the symbol "+". Each subsequent "picked atom" is likewise substituted for a "+" symbol in the typed command. As an alternative to typing a "+" and then picking an atom with the mouse, an atom can be picked by clicking the left or middle mouse button on it while the keyboard ALT key is depressed. Its atom specifier will then be inserted in front of the command line cursor.

The command may be edited at any point using the editing commands detailed in Appendix 3. The command is executed when the user hits the RETURN key. For example, to label two atoms, type:

label + +

Then "pick" the atoms with the mouse. After the atom names are substituted for the "+ +", press the RETURN key to execute the command. When picking atoms, it may be necessary to rotate the molecule so that the desired atom is not obscured by other atoms, or so that neighbor atoms are not picked by mistake.

11. Commands

11.1. Command Overview

MIDAS commands allow the user a variety of modes of execution. The user may:

(1) Type in commands at the graphics system keyboard,

(2) Set up a command file named *.midasrc* which is automatically read each time MIDAS is executed,

(3) Set up a command file which can be executed via the **source** or **read** commands,

(4) Control movements via the mouse or auxiliary devices such as a joystick or "Spaceball".

Commands which are typed in at the graphics system keyboard are echoed at the very bottom of the graphics display screen. Replies generated by MIDAS appear just above this line. There may be several lines of reply messages. For example:

```
Reply:  Clipping plane is missing
Usage: assign [ slider_num function [ direction ] ]
Command: assign 0 clipping
```

appears on the screen when the user inadvertently forgets to supply the required clipping plane argument to the **assign** command.

11.2. MIDAS Start-Up

The MIDAS commands in the start-up file *.midasrc*, are executed each time MIDAS is executed. There is also a system start-up file, */usr/local/lib/midas/midas.rc*, executed each time. The order of execution of start-up files proceeds as follows:

(1) The system start-up file OR the file specified by the user's environment variable "MIDASRC". (See Appendix 4 of this document.)

(2) The *.midasrc* in the user's main directory.

(3) The *.midasrc* in the user's present working directory.

Start-up files are conveniently used for assigning pseudo-sliders as well as defining aliases and setting display options. (See the **assign**, **alias**, and **set** commands.) Any legal MIDAS command, however, may be included in a start-up file.

MidasPlus starts up in window mode with a default window size of 645 x 484 pixels (good for creating NTSC standard video tapes). To get a full-screen window with no border, use the **–f** command line option. If you always want full screen behavior when running MIDAS, you can put the following line in your *.cshrc* file:

```
alias midas midas -f
```

On the Silicon Graphics IRIS, MidasPlus now will refuse to start up if you're not logged in on the console. You can force opening a window on the console by giving the **–F** command line option.

11.3. Command Synopsis

MIDAS commands may be grouped together on one line using the ";" character as a delimiter. For example:

label #1; color 32,s #1; color green,b #1

If a large system is being displayed, it might be advantageous to use such a compound command since the graphics image is drawn only after the entire command has been executed.

Each description of the MIDAS commands in this document contains a line indicating the correct usage of the command. The usage includes the command name appearing in **boldface** print followed by command line parameters in *italic* or roman print. Parameters appearing in *italics* require substitution of the appropriate name, digit, etc. by the user. Parameters appearing in roman print are literals and should be typed in as they appear in the usage line. Parameters which appear inside square brackets, "[...]", are optional. All parameters not appearing inside square brackets are required for the command to execute. Keyword parameters are sometimes separated by vertical bars ("|"), which indicate that the keywords are mutually exclusive. Parameters named *atom_specification* always refer to a selection of atoms, residues and/or models as described in section 10.1 and 10.2 of this document.

MIDAS accepts abbreviated forms for all commands. The abbreviation consists of the shortest substring which unambiguously identifies the command. For example, "rec" may be substituted for **record**. Typing "re" for **reverse** will not work because it could mean **read**, **record**, **reset** or **reverse**.

The commands available in MIDAS are summarized in the following tables and described individually in detail on subsequent pages:

(see next page)

MidasPlus Commands	
Command	Function
addaa	add an amino acid to the end of a molecule
addgrp	add a new group to a residue
alias	set command aliases
align	align two atoms along the z-axis
angle	calculate the angle between three atoms
assign	assign functions to pseudo-sliders
bond	make a bond between two atoms
brotation	initiate a "backwards" bond rotation
cd	change current working directory
center	specify center of image
chain	chain specified atoms together
clip	move clipping planes
cofr	change center of rotation
color	color bonds, labels and surfaces
conic	display shadowed, space-filling image
copy	send display image to a printer or file
delegate	specify an action involving delegate program(s)
delete	delete a group from a residue
devopt	set device specific option
display	display specified molecules, residues, atoms
distance	display atom distances
echo	display text in reply area
fix	make bond rotations permanent
fixreverse	fix bond rotations and reverse rotation
freeze	stop rock or roll motion
getcrd	return <x,y,z> coordinates for an atom
help	show information about MidasPlus commands
intensity	set depth cue intensity at hither and yon clipping planes
label	label atoms and residues
link	join two residue chains
match	superimpose two models
matrixcopy	copy transformation matrix from one model to another
matrixget	output a tranformation matrix to a file
matrixset	set a tranformation matrix from a file
midaspop	pop MIDAS window to front of other screen windows
midaspush	push MIDAS window behind other screen windows
move	translate selected models
open	open a MIDAS database or PDB file for display
pdbrun	pipe PDB file describing current models to arbitrary command
push/pop	push or pop images on the picture stack

(continued on next page)

MidasPlus Commands
(continued)

Command	Function
read	read a command file
record	record all executed MidasPlus commands in a file
reset	reset all models to original orientations
reverse	reverse the direction of a rotation
rlabel	enable residue labeling
rock	rock a structure about the x, y or z axis
roll	roll a structure or bond rotation about the x, y, or z axis
rotation	activate a bond rotation
run	execute a shell command and send output to MIDAS
save	save a MIDAS session
savepos	save a model's current orientation
scale	apply a scaling factor to all models
section	change sectioning of the display
select	select models for move, rock, roll, or turn commands
set/unset	set options
setcom	set molecule's center of mass
show	display specified atoms and no others
sleep	temporarily suspend all display activity
source	read and execute a command file
speed	set the control speed of pseudo-sliders and spaceball
stereo	specify whether to use stereo and in what manner
stop	terminate the current MIDAS session
surface	display a model's "ms" surface
swapaa	exchange one amino acid for another
swapna	exchange a nucleotide for another
system	execute a UNIX shell command
thickness	change thickness of the displayed image cross section
turn	turn a structure about the x, y, or z axis
update	change coordinates of a model from a PDB file
vdw	display van der Waals surface
vdwopt	set van der Waals surface options
wait	interrupt processing until model has stopped moving
watch	graphically monitor interatomic distances
watchopt	specify parameters used by **watch** command
window	display the entire molecule on the screen
write	output a model as a MIDAS database or PDB file
version	report MIDAS version number
zone	display only those atoms within specified distance of others

The action of many MIDAS commands may be reversed by preceding the command with the tilde character "~". This is essentially an "undo" for any of the following commands:

Reverse Command Functions	
Command	Function
~alias	delete an alias
~angle	remove an angle monitor
~assign	deactivate pseudo-sliders
~bond	remove a bond between two atoms
~brotation	remove a "backwards" rotation
~chain	break chaining for all atoms listed
~clip	halt an ongoing clipping operation
~cofr	use default center of rotation
~display	delete atoms from the display
~distance	remove a distance calculation
~link	break a residue chain into two parts
~label	remove atom and residue labels
~move	stop an ongoing move operation
~open	close a model
~pop	equivalent to **push**
~push	equivalent to **pop**
~rlabel	don't display residue labels
~rock	terminate **rock** motion
~roll	terminate **roll** motion
~rotation	remove a rotation
~savepos	delete saved position
~scale	stop ongoing scaling operation
~section	stop an ongoing section operation
~select	deselect a model
~set	unset an option
~setcom	use default center of mass
~show	delete atoms from the display
~stereo	equivalent to "**stereo** off"
~surface	remove a "ms" surface display
~thickness	stop an ongoing thickness operation
~turn	terminate **turn** motion
~vdw	remove a van der Waals surface display
~watch	terminate **watch** monitoring

11.4. Addaa

Usage: **addaa** *residue_type*, *residue_sequence* [, *conformation*] *residue*

Addaa adds an amino acid of type *residue_type*, with sequence number *residue_sequence*, in the specified conformation after the specified *residue*. *Conformation* may be one of:

EXT	extended (default)
ALPHA	alpha helix
PBETA	parallel beta sheet
ABETA	antiparallel beta sheet

Currently, *residue* may only be the very last residue of a molecule. (This restriction may be removed in future versions of MIDAS.)

The temperature factor for the new residue is set to the highest currently found in the model.

Examples:

addaa tyr,30 #0:29

Add **tyr** as residue **30** after residue **#0:29**

11.5. Addgrp

Usage: **addgrp** *group*, *bond_length*, *bond_angle* [, *dihedral_angle* [, *new_resname*]] *atom1 atom2 atom3*

Addgrp adds a new chemical group whose position is determined by the three specified atoms. The parameters required are the group name (which corresponds to a file in the directory */usr/mol/groups* or a file in the user's "groups" directory), the *bond_length* from *atom1* to the first atom of the added group, and the *bond_angle* formed by the group being added, *atom1* and *atom2*. The *dihedral_angle*, which defaults to 0, and the *new_resname*, which defaults to the old residue name, are optional. The *dihedral_angle* must be a positive value between 0 and 360, inclusive.

Users may create groups in a private directory named *groups* in their home directory. Use the files in the system directory, */usr/mol/groups*, as formating guides. A group name must contain only alphanumeric characters.

Note that adding a group creates a new residue which is colored white and has no labels. The temperature factor for the new residue is set to the highest currently found in the model.

See also: **delete**

11.6. Alias

Usage: **alias** [*name* [*wordlist...*]]

Alias assigns to *name* the specified *wordlist*. All subsequent appearances of the space delimited *name* will be substituted with the *wordlist*. The *wordlist* may contain multiple commands separated by semicolons in which case the *wordlist* must be imbedded in double quotes, like this:

alias name "command1 ; command2"

The **alias** command without any arguments reports all current aliases. The **alias** command with *name* only reports the alias for that *name*. ~**alias** *name* deletes the alias for that *name*.

11.7. Align

Usage: **align** *atom1 atom2*

Align positions the selected model(s) such that the two specified atoms lie along the z-axis at the center of the screen. *Atom1* is positioned in the front and *atom2* is in the back.

See also: **reset, push/pop, savepos, window**

11.8. Angle

Usage: **angle** [*angle_no*] *atom1 atom2 atom3* [*atom4*]

Angle monitors the angle in degrees between the three specified atoms. If four atoms are specified, the dihedral angle is monitored. The atoms need not be connected and no diagnostic is given if the atoms are not connected. Up to 16 angles (0–15) may be monitored simultaneously. If the angle number is not specified, MIDAS will assign one for you.

~**angle** *angle_no* will remove the indicated angle monitor.

11.9. Assign

Usage: **assign** [*slider_number function* [*direction*]]

Assign is used to activate control panel pseudo-sliders. Each slider's number is displayed just to the left or right of the slider itself on the control panel. Usually only one slider may be assigned to any one function and reassignment of a function cancels the previous assignment. This default action of cancelling a previous assignment can be disabled via the **unset reassign** command (see **set/unset**).

Function may be any of the following key words or the first two letters of the appropriate key word:

Slider Functions

Keyword	Function
translation	model translation
rotation	model and bond rotation
clipping	clipping planes
scaling	changes size of selected models
section	moves hither and yon planes in the same direction at the same rate
thickness	moves hither and yon planes in the opposing directions at the same rate
nothing	for unassigning sliders

The *direction*, if applicable, may be any of:

Slider Directions		
Direction	Designation	Applicability
x	the x axis	translation, rotation
y	the y axis	translation, rotation
z	the z axis	translation, rotation
h	hither plane	clipping, section, thickness
y	yon plane	clipping, section, thickness
0–15	rotation number	rotation

Note that if *direction* is an integer, the corresponding intramolecular bond rotation is assigned to the slider. *Direction* is required for the functions **translation, rotation**, and **clipping**.

The default device assignments are located in */usr/local/lib/midas/midas.rc* and listed in Appendix 4 of this document. The user may find it convenient to make additional automatic assignments by constructing a ".midasrc" file in his or her home directory and/or present working directory. Each time MIDAS is executed, any ".midasrc" files in the user's home directory *and* the current working directory are executed before user commands are processed.

The **assign** command without any arguments reports all current assignments.

Example: **assign** 0 clipping h

See also: **select, (b)rotation**

11.10. Bond

Usage: **bond** *atom1 atom2*

The **bond** command tells MIDAS that *atom1* and *atom2* are bonded. *Atom1* and *atom2* must be in the same model.

~**bond** breaks an existing bond between two atoms.

See also: **link**

11.11. Brotation

Usage: **brotation** [*rotation_number*] *atom1, atom2*[, *atom3, atom4*]

Brotation produces a "backwards" rotation, *i.e.* the portion of the molecule which remains fixed in a **rotation** command is rotated in the **brotation** command and vice versa. See **rotation** for an explanation of the syntax.

See also: **rotation, reverse**

11.12. Cd

Usage: **cd** *path_name*

Cd changes the current working directory to *path_name*. If you are not familiar with the concept of directories and path names, see "UNIX for Beginners" (Kernighan) and the description of the "cd" command in section 1 of the UNIX User's Manual. Note that all subsequent **record** and **save** commands are executed in the new working directory.

Example: **cd** ../crodna

11.13. Center

Usage: **center** *atom_specification*

Center places the center of the atoms in the *atom_specification* at the center of the current view.

See also: **window**

11.14. Chain

Usage: **chain** *atom_specification*

The **chain** command draws pseudobonds between the specified atoms, undisplaying all others. This is particularly useful for displaying carbon atoms in a protein.

Example: **chain** @ca chain all alpha carbons

See also: **display, show, zone**

11.15. Clip

Usage: **clip** *plane units* [*frames* [*wait_frames*]]

The clipping planes may be moved relative to their current position by a specified number of angstrom *units*. *Units* is a positive or negative number in angstroms. A positive number moves the plane towards the user. A negative number moves it away from the user. *Plane* may be either **hither**, or **yon.**

Frames moves the clipping plane in the specified manner for the specified number of image update frames. *Wait_frames*, if specified, indicates the number of frames to wait before beginning the move. ~**clip** will halt an ongoing clip. *Frames* and *wait_frames* default to 1 and 0, respectively. These parameters are useful for controlling the rate of clipping and are helpful when constructing MIDAS command scripts and making videos.

See also: **intensity**

11.16. Cofr

Usage: **cofr** [*atom_specification* | view]

When invoked without any arguments, the **cofr** command reports the current center of rotation.

If given an *atom_specification*, the **cofr** command sets the center of the bounding box of the given atom(s) as the center of subsequent rotations.

If the **view** keyword is given instead, the center of rotation is set to be the center of the current view.

~cofr causes MIDAS to use its default center of rotation, instead of any previous user-specified one.

11.17. Color

Usage: **color** *color_designation*[,s][,l][,b][,v] *atom_specification* [e>*potential*] [b>*temperature*]

The **color** command allows the user to selectively color bonds, labels and surfaces by model, residue, and atom. *Color_designation* is an integer ranging from 0 to 64, inclusive, or a keyword as follows:

Numerical Color Mapping

Keyword	Equivalent integer
green	1
cyan	8
blue	16
magenta	24
red	32
yellow	48
white	0
black	64
gray/grey	65

Another special color keyword is **byatom**, which is a simple attempt at determining color from atom name. Atom name prefixes and their corresponding colors are as follows:

CL/BR	magenta
FE	gray
C	gray
O	red
N	light blue
H/D/digit	white
S	yellow
I/F	magenta
B	gray
others	blue

These colors are coded into MidasPlus and cannot be easily changed.

Optionally, the user may specify one of the designations **s**, **l**, **b** or **v** to color surfaces, labels, bonds, and van der Waals surfaces, respectively. In the absence of these specifiers, MIDAS colors everything, *e.g.* bonds, surfaces, labels and van der Waals surfaces of all specified atoms.

Note that the solvent accessible surface and the van der Waals surface may be displayed simultaneously, and thus colored separately. See the **surface** and **vdw** commands in this document.

The *atom_specification* uses the standard MIDAS syntax. Note that the **~color** command will not work, *i.e.* models may not be "uncolored".

On Silicon Graphics IRIS systems with the "GT" graphics option, bond colors will smoothly change from atom to atom due to the GT's automatic color blending if "**devopt** blend" is *on*. On other systems, the entire bond will be given the color of the bond's initial atom. To get half bond coloring on these

systems, use **set halfbond**. This may result in decreased interactive response due to the increased number of vectors that have to be drawn.

If no *atom_specification* is given, all open models are colored.

Examples:

color red #0:4@*	color all bonds in the fourth residue of model 0 red.
color 20,s #3	color model 3 surface color 20 (midway between blue and magenta).
color green #2:HIS	color all histidine residues on model 2 green.
color blue,s #1 e<5	color all atom surfaces on model 1 with electrostatic potential less than 5 the color blue.

Ranges of colors may be mapped onto models based on either surface electrostatic potential or atom temperature factor. A range of color is specified by either keywords or numbers, such as *blue-red* or *16-32*. To color models by electrostatic potential use the surface designation, **s**, with the color range. Colors are mapped onto the surface from lowest to highest potential based on a scale determined by the lowest and highest potential values *of all open models*. If boundary conditions are given *i.e.* **e>**, **e<**), the colors are mapped within the boundaries. Similarly, for temperature factors, use the **l**, **b**, or **v** designation and specify boundary conditions using **b<** and **b>**. The default designation maps bond color by temperature factor and surface color by electrostatic potential.

For finer distribution of potentials and temperature factors, broader color ranges should be used. For example, *blue-red* maps a range of 17 colors whereas *cyan-yellow* maps 41 colors.

Examples:

color blue-red,s #0	color lowest potential surface blue and highest red. Intermediate potentials are mapped to shades of magenta.
color red-blue,b #0	color the bonds of the lowest temperature factor atoms red and highest blue.
color 8-32,s #0 e>-20 e<20	color *ms* surfaces with potential -20 kcal/mole cyan (color 8) and with potential +20 kcal/mole orange (color 40). Intermediate potentials are mapped to shades of blue, magenta, and red.

11.18. Conic

Usage: **conic** *options*

The **conic** command produces a space-filling rendering of the current molecule(s). The current orientation and coloring is retained. Each atom is depicted as a sphere of the appropriate radius with realistic highlighting and shadowing effects. This image is not interactive and takes anywhere from 30 seconds to a few minutes to produce. Clicking the left button returns to MidasPlus.

There are many options, which are detailed fully in the *conic* manual page included in Appendix 6 of this document.

See also: **ribbon**

11.19. Copy

Usage: **copy** [date] [box] [flat] [bg|background *color*] [intensity *0-1*] [printer *printer*] [file *file*] [title *title* ...]

The **copy** command sends a PostScript description of the current picture to a printer or disk file.

The optional keyword **date** puts today's date in the lower left hand corner of the copy. The optional keyword **box** draws a heavy border around the copy. The **flat** option forces all vectors (line segments) to be the same thickness, thereby disabling the pseudo-depthcueing effect created with variable width lines.

The **background** option sets the background copy color without changing what is shown on the screen. Similarly, the **intensity** option controls the color intensity of the copy background without affecting the screen background.

If the **printer** keyword is given, the copy is sent to *printer*. If the **file** keyword is given, the Postscript is saved in *file*. If neither is given, the copy is sent to the default system printer.

If the **title** keyword is given, the entire rest of the line is taken to be the title for the picture. Therefore, if **title** is specified, it must be the last thing in the **copy** command. If no title is given, you will be prompted for the title. If no title is desired, simply hit RETURN at the prompt. The title is centered at the bottom of the picture.

Users of an IRIS 4DGT should note that in order to get halfbond coloring on color hardcopies, halfbond mode will have to be set before issuing the **copy** command (and probably unset immediately after the copy, since MIDAS on GT's looks worse with halfbond mode set).

11.20. Delegate

Usage: **delegate** list
Usage: **delegate** *name* start *command* [*command_arguments*]
Usage: **delegate** *name* stop

The **list** option lists all the active delegates.

The **start** and **stop** options, start up and terminate a delegate of the given *name*, respectively.

The delegate mechanism allows other programs to extend the interactive capabilities of MidasPlus. The delegate mechanism is discussed in detail in appendix 5 of this manual. For a detailed description of an application developed using delegates, see "Automated Site-Directed Drug Design Using Molecular Lattices" in the *Journal of Molecular Graphics*, in press [R. A. Lewis, *et al.*].

See also: **pdbrun, run, system**

11.21. Delete

Usage: **delete** *new_residue_type atom_specification*

Delete removes a branch of atoms from a residue. All atoms from *atom_specification* to the end of the side chain are deleted from the residue. Deleting from a main chain atom produces unpredictable results. Because the structure of the residue changes after deletion, a *new_residue_type* must be specified to differentiate between the original and new residue types.

See also: **addgrp**

11.22. Devopt

Usage: **devopt** [*option* [on | off]]

Devopt sets device-specific options. This mechanism replaces the **set** syntax that was used previously, *e.g.*, **set smooth** is now **devopt smooth on**.

Issuing a **devopt** command with no arguments will list the options available on your machine as well as each option's current setting.

The **smooth** option turns antialiasing on and off.

The **colormap** option turns on and off colormap mode. This mode is provided because on many models of the IRIS antialiased lines are *much* faster in colormap mode than in RGB mode.

The **blend** option turns on and off color-blending. Color blending is off by default because neither the Personal IRIS nor the IRIS VGX support depthcued color blending.

The **antialias_points** option controls whether points used to depict molecular surfaces are antialiased when the **smooth** option is on. Having **antialias_points** off will improve graphics performance and may improve or degrade image quality, depending on the specific model of workstation that MidasPlus is running on.

Setting the **ucsf_stereo** option on prevents the **stereo sequential** command from automatically switching the monitor to 120 Hz interlaced refresh rate. The monitor must instead be switched manually.

See also: **set**

11.23. Display

Usage: **display** *atom_specification*

Display allows the user to selectively display the atoms of a model. The *atom_specification* may be any combination of molecules, residues and atoms for the currently open models. To display only selected portions of a model the user may use **~display** to delete the unwanted atoms and labels. When used with the **label** and **surface** commands, the user is able to display any combination of bonds, labels and atom surfaces.

See also: **chain, show, label, surface**

11.24. Distance

Usage: **distance** [*distance_number*] *atom1 atom2*

Distance dynamically calculates and displays distances in angstroms between specified atoms. A dotted line is drawn between each pair of atoms for which a distance calculation is active. The user may assign a *distance_number* between 0 and 15, inclusive, to each active distance calculation. If *distance_number* is not given, one will be assigned for you. The distance calculation may be deactivated using **~distance** *distance_number*. *Atom1* and *atom2* may be any two atoms on the displayed models in standard MIDAS syntax.

Examples:

distance 1 #1:12@CA #0:47@CA	assign distance 1
~**distance** 0,1	remove distances 0 and 1

See also: **watch**

11.25. Echo

Usage: **echo** *text*

echo places all of the *text* argument into the MidasPlus reply buffer visible at the bottom of the graphics window. **echo** may be used by custom MidasPlus scripts to send messages back the user.

11.26. Fix

Usage: **fix** *rotation_number*

The **fix** command removes a previously activated torsional bond rotation and leaves the structure as currently displayed. It is necessary to **fix** a rotation before giving a **save** command if the current position of the rotation is to be reflected in the **saved** PDB file of the model with the active rotation.

11.27. Fixreverse

Usage: **fixreverse** *rotation_number*

Fixreverse fixes the named bond rotation and activates the reverse bond rotation using the same *rotation_number*. This command insures that the model will not move relative to other displayed models as is often the case when the **reverse** command is used.

Warning: In the case of multiple bond rotations, *if* the set of atoms rotated by a given bond rotation includes the pivotal atom of another bond rotation and *if* the set of atoms rotated by this second rotation includes the pivotal atom of the first, *then* MIDAS will not allow such rotations. In executing **fixreverse**, reversing the bond rotation may cause such a conflict and be disallowed. Reversing bond rotations must be done in an appropriate order such that intermediate combinations are legal. (See **rotation**.)

11.28. Freeze

Usage: **freeze**

Freeze stops all motion on the screen.

See also: **rock, roll**

11.29. Getcrd

Usage: **getcrd** *atom_specification*

The **getcrd** command returns the untransformed x, y and z coordinates for the atom specified. The *atom_specification* must select one atom only using the standard #:@ syntax. The coordinates are returned in the command reply area at the bottom of the MIDAS window.

11.30. Help

Usage: **help** [*topic*]

The **help** command displays information on the selected topic. If no *topic* is specified, MIDAS displays a list of all available topics.

11.31. Intensity

Usage: **intensity** *hither_value* [*yon_value*]

Intensity changes the value of the intensity for the hither and yon clipping planes. The values range from 0 to 1. Default values are 1 for the hither (maximum intensity) to 0.2 for the yon (minimum intensity). Note that the location of the hither and yon clipping planes may be changed with the **clip** command or by assigning thickness and section to the control panel sliders (which is done by default). For backwards compatibility, it is legal to use intensity values in the range 0-255 (which are then mapped to 0-1) but this functionality is obsolescent and should not be relied upon to exist in future versions of MIDAS.

See also: **clip**

11.32. Label

Usage: **label** *atom_specification*

Atoms and residues are labeled appropriately. *Atom_specification* may be any displayed atoms, residues or models. The residue name appears after the first labeled atom in the residue.

Examples:

label #3	label everything in model 3
label #2:HIS	label all histidine residues in model 2
~label #2:40	unlabel the 40th residue in model 2

See also: **rlabel**

11.33. Link

Usage: **link** *residue*

This command is obsolete and is provided for backwards compatibility with old MIDAS command scripts. Use **bond** instead.

See also: **bond**

11.34. Match

Usage: **match** [selected] *atom_specification*

The **match** command uses the least squares fit method to superimposes two models. The *atom_specification* should contain an equal number of atoms from two different models.

The atoms are matched according to the order in which they are specified, *i.e.* the first atom of the first model is matched to the first atom of the second model, second atom to second atom, etc. The MIDAS command syntax allows much flexibility in specification, *i.e.* atom specifications can use all the shorthands available for related atoms. For example, the user might match models 1 and 2 thus:

match #1:3@C1@C2@P@O2 #2:3@C1@C2@P@O2

MIDAS will transform the second model so that its atoms overlay those of the first model. Specifying the **selected** option will make **match** transform not only the second model but all selected models as well, using the same transformation applied to the second model.

The user should be aware that the order in which atoms are specified in a list does not necessarily force the order in the match. For example,

match #1:3@C1,C2,P,O2 #2:3,C1,C2,P,O2

is not specific as to the order of the atoms C1, C2, P and O2. In this case, MIDAS will order the atoms as they occur in the template for residue 3. If residue 3 of model 1 and residue 3 of model 2 are the same, this is not a concern. If they different, however, then the order is not specific and the models may not be superimposed in the way the user would expect. Ordering may be forced by using the @ designation:

match #1:3@C1@C2@P@O2 #2:3@C1@C2@P@O2

The RMS error value from the least squares fit is returned in the command reply area at the bottom of the MIDAS window.

11.35. Matrixcopy/Matrixget/Matrixset

Usage: **matrixcopy** *from_model to_model*
Usage: **matrixget** *filename*
Usage: **matrixset** *filename*

Matrixcopy makes the 4x4 transformation matrix of model *to_model* the same as model *from_model*.

Matrixget prints the current 4x4 transformation matrices to a file named *filename*.

Matrixset reads matrices from the file named *filename* and sets the current transformation matrices (using the same file format as **matrixget**).

See also: **cofr, getcrd**

11.36. Midaspush/Midaspop

Usage: **midaspush**
Usage: **midaspop**

Midaspush pushes the MIDAS display window behind all other screen windows and icons. These windows and icons can then be used normally. **Midaspop** brings MIDAS back to the top. Make sure that the mouse cursor is over the MIDAS window when **midaspop** is typed because MIDAS will not receive keystrokes typed with the mouse cursor positioned over other screen windows. If you accidentally type with the mouse incorrectly positioned, erase your input in the other window, then move the mouse over the MIDAS window and type **midaspop**.

Note that if MIDAS is not using the full screen, and therefore has a window frame around it, clicking the right mouse button over the window frame title bar will bring up a menu from which it is also possible to push and pop the MIDAS window. This is a standard IRIS menu for applications using windows.

11.37. Move

Usage: **move** *axis units* [*frames* [*wait_frames*]]

The **move** command translates the selected molecule(s) along the specified *axis*. *Axis* may be *x*, *y* or *z*. *Units* is a floating point number in angstroms. A positive value for *units* indicates translation to the right, up, or toward the user and corresponds to the x, y, and z axes, respectively.

Frames moves the models in the specified manner for the specified number of image update frames. *Wait_frames*, if specified, indicates the number of frames to wait before beginning the move. ~**move** will halt an ongoing move. *Frames* and *wait_frames* default to 1 and 0, respectively. These parameters are useful for controlling the rate of motion and are helpful when constructing MIDAS command scripts and making videos.

If the **move** command does not work, it is likely you have failed to **select** the target models.

See also: **select**

11.38. Open

Usage: **open** [model | surface | pdb | ms | object | midas] [*model_number*] *filename* [*surface_database*]

Open causes the contents of the file *filename* to be read and shown as model *model_number* (if *model_number* is omitted, the lowest available model number is used). The model number is used to uniquely reference the model in subsequent commands and therefore should be remembered.

How MIDAS interprets *filename* is controlled by the optional keyword specifier. If no keyword is given, or the keyword **model** is specified, MIDAS will first try to open a MIDAS database that has *filename* as its root name. If unsuccessful, it will then assume that *filename* is a PDB file and try to open that. The keyword **pdb** forces the latter action and prevents any attempt to open *filename* as a MIDAS database whereas the keyword **midas** has the reverse effect. Note that opening a MIDAS database is slightly faster than opening the corresponding PDB file and that the program for creating databases (**midas.in**) is more accurate in determining connectivity, although MIDAS generally gets it correct also. See Part I of this document for further detail on using the **midas.in** program. Once a PDB file is open, it can be written out as a MIDAS database using the **write** command, and vice versa.

The **ms** keyword indicates the *filename* is the output of the **ms** utility, used to generate solvent accessible surfaces, and that the surface should be associated with the indicated *model_number* (*model_number* cannot be omitted in this case).

MIDAS can read **ms** and PDB files that have been compressed using the UNIX "compress" command. Compressed **ms** and PDB files use substantially less disk space than regular files. There is no need to first uncompress the files or to specify any special keyword on the **open** command line.

The obsolescent **surface** keyword indicates that *filename* is a surface prepared with the **makesurf** utility, a postprocessor for **ms**. Since MIDAS can now open **ms** files directly, there is no reason to use

makesurf, and the **surface** keyword is only retained for backwards compatibility. Once a surface file or database is open, the **surface** command may be used to display the surface. Optionally, a MIDAS surface database, *surface_database*, may be specified as a fourth argument. This surface is associated with the open model.

If **open** is used with the **object** keyword, the file is assumed to specify a non-molecule graphic object, and can be opened as a new model, or associated with an existing model. Each line in the object file is a command or text. If it is text, then it is displayed in the current color and font at the current position. All of the commands start with a period and are as follows:

.comment *text*	comments
.c *text*	comments
.font *name size*	set font and size
.color *color_designation*	set color
.cmov *x y* [*z*]	set character position
.dot *x y* [*z*]	show dot at position
.marker *x y* [*z*]	show marker at position
.m *x y* [*z*]	move to location
.move *x y* [*z*]	move to location
.d *x y* [*z*]	draw line from last location
.draw *x y* [*z*]	draw line from last location

If the **autocolor** option has been set (using the command ***set autocolor***), then each newly opened model will be given a unique color so that different models can be easily distinguished. Note that MIDAS databases have their color stored with them, so autocolor has no effect on MIDAS databases.

To close a model, use **~open** followed by either the model number or name. If the name is used, it must match exactly the name used in the open command.

In MidasPlus, if a model is closed and another model opened with the same model number then none of the transformations applied to the previous model are applied to the newly opened model. Note that this is in direct contrast to previous versions of MIDAS where all the transformations were applied, in order to facilitate docking. If docking is necessary, two models can be placed in approximately the same orientation with the **match** command.

Filename may be a pathname to a database/file or the name of a database/file in the current working directory. To change directories, use the **cd** command.

See also: **cd, match, write, midas.in, surface**

11.39. Pdbrun

Usage: **pdbrun** [**all**] [**conect**] *command* [*command_args...*]

Pdbrun causes *command* and *command_args* to be passed to the user's preferred UNIX shell for execution. A PDB file describing the current models is also passed as standard input. Normal shell metacharacters (notably output redirection) can be used. Errors are ignored and any shell output is interpreted as MIDAS commands and executed (*a la* **run**).

The **all** option specifies that all atoms, not just those shown, labeled, vdw'ed, or surfaced should be sent to the given *command*.

The **conect** [*sic*] option specifies that PDB-standard CONECT records should be generated for all residues, even if they have standard connectivity.

Control returns to MIDAS when the **pdbrun** command terminates. This command facilitates extensions to the normal MIDAS command set. In particular, both the **conic** and the **ribbon** commands are implemented using **pdbrun.**

TER records are inserted normally in the PDB file and END records follow each model. USER records are also added to give additional information. These remarks are of the general form:

USER *keyword information*

The following table describes each *keyword* and its associated *information*.

Pdbrun Keywords		
Keyword	Where Found	Information
PDBRUN	First line of file	Pdbrun version number
EYEPOS	Second line of file	Where the viewer is assumed to be positioned (x,y,z) for purposes of calculating what to display.
ATPOS	Third line of file	The location the viewer is assumed to be looking towards for determining line of sight.
WINDOW	Fourth line of file	View volume displayed, relative to the line of sight. The 6 numbers are, respectively, x_{left}, x_{right}, y_{bottom}, y_{top}, z_{hither}, z_{yon}. Since in MIDAS the view volume is symmetric about the line of sight, x_{left} and y_{bottom} are both negative and equal in magnitude respectively to x_{right} and y_{top}, which are both positive. z_{hither} and z_{yon} are positive distances from the viewer to the *hither* and *yon* clipping planes, respectively.
VIEWPORT	Fifth line of file	Extent of the MIDAS window, in screen coordinates. $(x_{min}, x_{max}, y_{min}, y_{max})$
FILE	After each MODEL record	Name of file corresponding model was opened from
COLOR	After each ATOM record	MIDAS color number and red, green, and blue values in the range 0 to 1
RADIUS	After each ATOM record	Atomic radius in angstroms

A typical set of records describing an atom looks like this:

```
ATOM      1  C   HIS     1      49.168  26.701  10.916  1.00 16.00
USER  COLOR 10 0.000 0.667 1.000
USER  RADIUS 1.800
```

which indicates the C atom of histidine residue 1 is colored an off-blue (color 10, no red, two-thirds full green, full blue) and has an atomic radius of 1.8 angstroms.

See also: **delegate, run, system, write**

11.40. Push/Pop

Usage: **push**
Usage: **pop**

Push saves the current orientation of all open models on an image stack.[†] **Pop** retrieves the previous "pushed" orientation. Any number of model orientations may be saved on the stack, memory allowing, and retrieved on a last-in first-out basis.

See also: **align, reset, savepos, window**

11.41. Read

Usage: **read** *filename*

Read executes the contents of the file *filename* as a command file. **Read** differs from **source** in that **source** updates the display after each command while **read** only updates the display after all commands are done.

See also: **record, source**

11.42. Record

Usage: **record** *filename*

Record saves all the commands remembered by the **set record** command in the file *filename*. Saved commands can later be re-executed with a **read** or **source** command, although the command file should first be editted since the **record** command itself will be in the file. **Record** is very useful for creating demonstration scripts for later playback. Note that there is normally a **set record** in the system startup file distributed with MIDAS so that, by default, all typed user input is remembered. Thus all commands used since MIDAS startup can easily saved with **record**. This is useful for reproducing results or applying commands used on one set of models to a different set of models.

See also: **read, set, source**

11.43. Redraw

Usage: **Redraw** *atom_specification*

This command no longer does anything. It is provided for backwards compatibility with old MIDAS command scripts.

[†] For those unfamiliar with the concept of a "stack", think of a pile of pictures which is created by "pushing" pictures one at a time onto the *top* of the pile and in which pictures are retrieved one at a time by taking the *top* picture off the pile.

11.44. Reset

Usage: **reset** [*view_name*]

Reset returns models back to saved orientations. In each MIDAS session, the original orientation of the *first* model(s) opened is saved as *view_name* "default". This orientation may be retrieved by the command **reset** or the command **reset default**. Other orientations may be saved using the **savepos** command:

savepos *view_name*

and retrieved using the **reset** command:

reset *view_name*

For a list of existing *view_names*, give the command:

reset list

See also: **align, push/pop, savepos, window**

11.45. Reverse

Usage: **reverse** *rotation_number*

Reverse reverses the direction of an assigned rotation. If the direction is reversed, then the portion of the molecule which rotated previously remains fixed and vice versa in subsequent rotations. Reverse rotations may also be attained by using the **brotation** command.

Warning: In the case of multiple bond rotations, *if* the set of atoms rotated by a given bond rotation includes the pivotal atom of another bond rotation and *if* the set of atoms rotated by this second rotation includes the pivotal atom of the first, *then* MIDAS will not allow such rotations. In executing **reverse**, reversing the bond rotation may cause such a conflict and be disallowed. Reversing bond rotations must be done in an appropriate order such that all intermediate combinations are legal. (See **rotation**.)

See also: **brotation, rotation**

11.46. Ribbon

Usage: **ribbon** *options*

The **ribbon** command produces an aesthetic representation of the secondary structure of the current molecule(s), in a manner reminiscent of a "Jane Richardson drawing." The current model orientation and coloring is retained. Clicking the left mouse button returns to MidasPlus.

In order for the ribbon command to know what the secondary structure of the model(s) is, each model must have been opened from a PDB file, and that PDB file has to have correct `HELIX` and `SHEET` records. Models that were opened from MIDAS databases are rendered as garden hoses, since they lack this secondary structure information.

There are many options, which are detailed fully in the **cartoon** manual page included in Appendix 6 of this document. (The **ribbon** command is actually an alias that executes the **pdbrun** command and sends data to the **cartoon** program.)

See also: **conic, ribbon**

11.47. Rlabel

Usage: **rlabel** *atom_list*

Rlabel enables residue labeling of the first displayed atom of each residue in *atom_list*. This condition is true by default in MIDAS. **~rlabel** may be used to turn off display of residue labels. This is particularly useful for showing the position of water molecules without displaying a long label.

See also: **label**

11.48. Rock

Usage: **rock** [*axis* [*angle* [*frames* [*wait_frames*]]]]

Selected structures may be rocked about the *x*, *y* or *z axis*. The *angle* indicates the number of degrees the structure rotates at the fastest point in the sinusoidal period. (An *angle* value of 9 corresponds approximately to a 90 degree arc.)

The cycle time for **rock** is approximately 2.4 seconds. This corresponds to 18 frames forward and 18 frames back [15 frames per second].

If arguments are omitted, defaults will be used. These defaults are: *wait_frames* – *0*, *frames* – infinite, *angle* – *3*, *axis* – *y*.

If **rock** does not work, it is likely you have failed to **select** the target models.

See also: **freeze, select**

11.49. Roll

Usage: **roll** [*axis* [*angle* [*frames* [*wait_frames*]]]]

Roll will roll the selected structures about the *x*, *y*, or *z axis*. If *axis* is an integer, it refers to the corresponding bond rotation.

If *angle* is specified, the structure is rolled through that angle (given in degrees) for each frame. If *angle* has a negative value the direction of the rotation is reversed. *Frames*, if specified, indicates the number of image update frames over which the **roll** operation is carried out. If *frames* is not specified, the structure continues to roll until explicitly turned off with the command **~roll** *axis*. Note that two or three **rolls** may be active at the same time; that is, the user can roll the structure around two or three axes at the same time. Each must be turned off explicitly.

Wait_frames, if specified, indicates the number of frames to wait before beginning the roll. This is useful for making videos. For example,

roll x 2 90 30

rolls the model 180 degrees over 90 image updates (*i.e.* 2 degrees on each frame update) after waiting 30 image update cycles before beginning.

The default values for *axis* and *angle*, if omitted, are *y* and *3*, respectively.

If **roll** does not work, it is likely you have failed to **select** the target models.

See also: **freeze, select, turn**

11.50. Rotation

Usage: **rotation** [*rotation_number*] *atom1,atom2* [,*atom3,atom4*]

Rotation activates a bond rotation. All rotations are assigned a *rotation_number* between 0 and 15 inclusive, either by the user or automatically by MIDAS. Once a bond rotation is activated, the rotation may be manipulated/controlled by assigning the rotation to a pseudo-slider via the **assign** command.

The rotation number and current bond angle are reported on the top of the display screen. If only *atom1* and *atom2* are specified, the angle reported is relative to the beginning position, *i.e.* the position when the assignment was made. If four atoms are specified, then the torsional angle is reported. The **~rotation** command removes the rotation and returns the bond to its original conformation (use the **fix** command to preserve the new conformation).

In assigning multiple bond rotations to a model, MIDAS does not allow the set of atoms rotated by a given bond rotation to include the pivotal atom of another bond rotation which includes in its set of atoms the pivotal atom of the former rotation. In other words, each rotation affects some set of atoms. If a second rotation is added to a model, it may affect the pivotal atom of the existing rotation. If this is true and it is also true that the pivotal atom of the existing rotation affects the pivotal atom of the new rotation, then the rotation is illegal.

Examples:

rotation 1 #1:1@c8,c9	assign rotation 1 to the bond between c8 and c9 in the first residue of model 1
rotation 3 #1:2@205:3@1	assign rotation 3 to the bond between the terminal atom of residue 2 (atom 205) and the first atom of residue 3

See also: **brotation, reverse, assign, fix**

11.51. Run

Usage: **run** *cmd* [*cmd_args...*]

Run passes its arguments *cmd* and *cmd_args* to the shell for execution and takes the output from these as a series of MIDAS commands.

The user may interrupt the execution of the MIDAS commands by pressing the ESC key. Execution of the current command is completed before processing the interrupt.

See also: **pdb, un, system**

11.52. Save

Usage: **save** *session_name*

Save stores the model orientation, rotation, distance calculations, slider assignments and user options for the current session in a MIDAS session file. The model coordinates are saved in PDB format. Since the model orientations are stored in the session file and not the data files, PDB files produced with **save** do not have their coordinates transformed. To get transformed coordinates, use the **write** command.

To restart a saved session, use the command:

% **midas** *session_name*

Note that if bond rotations have been made and the user wishes to save the new orientation(s), the **fix** command must be invoked before the session is saved.

See also: **fix, stop, write**

11.53. Savepos

Usage: **savepos** [*view_name*]

Savepos saves the current view and associates *view_name* with it. If *view_name* is missing, the name "default" is used. The view may be retrieved using the **reset** command.

A view may be "forgotten" using the command **~savepos** *view_name*. This saves space in session files.

For a list of all existing *view_names*, give the command:

savepos list

See also: **align, reset, push/pop, window**

11.54. Scale

Usage: **scale** *factor* [*frames* [*wait_frames*]]

Scale multiplies the size of the displayed models by the specified scaling *factor*. The scaling factor must be positive.

Frames scales the models in the specified manner for the specified number of image update frames. *Wait_frames*, if specified, indicates the number of frames to wait before beginning the scaling. **~scale** will halt an ongoing scale. *Frames* and *wait_frames* default to 1 and 0, respectively. These parameters are useful for controlling the rate of scaling and are helpful when constructing MIDAS command scripts and making videos.

11.55. Section

Usage: **section** *units* [*frames* [*wait_frames*]]

Section moves the hither and yon clipping planes the specified number of angstrom *units*. This has the effect of displaying a different serial cross section of the displayed model(s). A positive number of units moves the cross section toward the user, whereas a negative number moves the cross section away from the user.

Frames moves the clipping planes in the specified manner for the specified number of image update frames. *Wait_frames*, if specified, indicates the number of frames to wait before beginning the clip. ~**section** will halt an ongoing section. *Frames* and *wait_frames* default to 1 and 0, respectively. These parameters are useful for controlling the rate of clipping and are helpful when constructing MIDAS command scripts and making videos.

See also: **thickness**

11.56. Select

Usage: **select** *atom_specification*
Usage: **select** *model_number ... | model_range*
Usage: **select** all

Select selects a model or models for subsequent **move**, **rock**, **roll** and **turn** commands as well as interactive mouse manipulations.

If the argument to **select** is an atom specification, then the model(s) containing those atoms will be (de)selected.

If **select** is used with the keyword "all", then all models will be selected. ~**select all** will then revert to the previous selection state. This feature is convenient for switching back and forth between moving models relative to one another and global motion of all models.

Otherwise the argument(s) to **select** should be one or more model numbers or ranges (of the form #-#), separated by spaces. Note that the *model* number does not permit a # symbol to be included in the number.

The numbered boxes on the lower part of the "control panel" portion of the display (referred to as pseudo-switches) are used to toggle the selection status of the corresponding model number. Clicking on the box will select a model if currently unselected (turning the box green), or deselect if selected (turning the box red). The box labelled "All" works in an analogous manner to the typed **select all**. Clicking on "All" once will cause all open models to become selected and turn the "All" box green. Clicking on the box again will return to the previous selection status and turn the "All" box red.

Examples:

select 1	selects model 1
select 1 5–8	selects models 1 and 5 through 8

See also: **assign**

11.57. Set/Unset

Usage: **set** *keyword* [*value*]
Usage: **unset** *keyword*

There are two types of display options in MIDAS, those that act as off/on toggles and those that vary over a range of values. For the toggle type of option, **set** with the appropriate *keyword* enables the option and ~**set** *keyword* or **unset** *keyword* disables the option. For value type options, **set** *keyword value* sets the value while **set** *keyword* displays the current value.

Although initially all toggle options are disabled, MIDAS sets certain options on at the beginning of each session by reading the initialization file */usr/local/lib/midas/midas.rc*. See Appendix 4 of this document for a list of the display options enabled during initialization.

See also: **devopt**

The available MIDAS display options are:

(see next page)

Set/Unset Toggle Options	
Keyword	Function
autocolor	Give each newly opened model a different color if set. Note that MIDAS databases have their color stored with them, so **autocolor** will have no effect on them.
cofg	Puts a '+' at the center of rotation for the selected models. The '+' corresponds to the center of gravity if there is only one molecule or if the rotations are independent (center of mass of selected molecules).
control	If set, display the control panel.
filenames	When set, make MIDAS display the filenames of the open models in the top left corner of the window above any bond rotation monitors.
fullscreen	When set, make MIDAS resize itself to use the full screen. Useful if MIDAS is started without a desired *-f* option. If then unset, MIDAS will revert to its original size.
halfbond	If set, atoms are colored by halfbond connections to other atoms instead of each atom having one whole bond associated with it. Note that using halfbond mode may degrade response time.
independent	If set, models rotate about their independent centers of mass, otherwise models rotate about the common center of mass.
labels	Turns on distance, rotation, and angle monitoring labels.
ortho	Use orthographic instead of perspective projection.
pair	Obsolete: use "**stereo** walleye" instead.
record	Initiates remembering of subsequent typed user commands. Note that commands that implicitly generate additional commands (*e.g.* **read**, **source**, **run**, **pdbrun**) are remembered but not expanded. Issuing a new **set record** when record mode is already set resets remembering from scratch. ~**set record** clears the command memory and prevents subsequent commands from being remembered. This will save some time and disk space since the commmands are remembered in a temporary disk file.
showsphere	Controls whether a circle defining the transition between *x,y* versus *z* rotation is shown when MidasPlus is in "virtual trackball" manipulation mode.
smooth	Obsolete: use "**devopt** smooth on" instead.
sphere	If set, MidasPlus uses a "virtual trackball" method for model manipulation. Otherwise, model interaction is controlled via various combinations of mouse buttons. This is explained in more detail in section 4.1 of this document.
stereo	Obsolete: use "**stereo** sequential" instead.
text	Activates the COMMAND and REPLY text lines on the bottom of the display screen. This option can be turned off using **unset text** when taking photographs.
verbose	MIDAS prints confirmation messages after each successful command. If verbose is **unset** these messages will not appear.

Set/Unset Value Options	
Keyword	Function
bg_color	Sets the MIDAS background color. *Value* can be either a color keyword or color index as described in detail under **color**.
bg_intensity	Controls the brightness of the background color. *Value* can vary from 0 (black) to 1 (full intensity). For purposes of backwards compatibility, *value* can be in the range 0-255, in which case it will be interpolated into the range 0-1 and handled appropriately. This latter functionality is obsolescent and should not be relied upon to exist in future MIDAS releases.
eyesep	The separation between the centers of the viewer's eyes, in inches. This information is necessary to compute the projections for stereo viewing. It is rarely necessary to change the default setting for most adults, but it might be necessary for children viewing stereo.
font	This allows atom labels to be in any font style. The *value* is a font name concatenated with a point size, *i.e.*, **Helvetica10** would set the font to be Helvetica, and the point size to 10.
linewidth	This controls the thickness with which bonds are drawn. The default is 1, and larger values produce thicker lines (and slower interaction).
nameplate	This controls the placement of the MidasPlus logo on the bottom of the screen. A *value* of 0.0 puts it to the extreme left; 1.0 puts it to the right.
viewdist	The distance from the viewer to the screen, in inches. This information is needed to correctly compute stereo projections. Its default setting is appropriate for most modeling work, but the value may have to be increased if a demonstration is being given where many people are further from the screen than normal.
vpsep	Amount of vertical separation between left and right eye images in stereo mode, in scan lines. This option should only have to be set once for each machine with stereo. It controls the vertical convergence of the left and right images when the stereo system is turned on. Once the correct *value* is determined (empirically), it should be put in */usr/local/lib/midas/midas.rc*.
walleye_scale	This scales the size of walleye-type stereo image pairs (see **stereo** command). The default size is correct when using opticomechanical stereo viewers, while a larger scale is useful for taking pictures for publications.

11.58. Setcom

Usage: **setcom** *model x_coord y_coord z_coord* [*radius* [*natoms*]]

When MIDAS is asked to rotate one or more models, it needs to know the center of mass of the model(s). **Setcom** is used to change the center of mass parameters in the rare case where the ones automatically computed by MIDAS are unacceptable.

X_coord, *y_coord*, and *z_coord* specify the new center of mass for *model*. If *radius* is given, it should be the shortest radius (in angstroms) from the new center of mass that encloses the model. This helps MIDAS do a better job of framing the models in the graphics window. Specifying *natoms* essentially tells MIDAS how much weight this model has when computing a group center of mass for multiple models.

~**setcom** will cause MIDAS to revert to the default center of mass that it normally computes.

11.59. Show

Usage: **show** *atom_specification*

Show displays the specified atoms and deletes all others from the display. Note that this differs from the **display** command in that the **show** command displays *only* those atoms specified (and will undisplay all others). ~**Show** removes the specified atoms completely from the display.

See also:

chain, display

11.60. Sleep

Usage: **sleep** *number_of_seconds*

Sleep causes MIDAS to pause for *number_of_seconds* seconds and then resume operation. This command is useful in command scripts where a break in the action is required.

See also: **wait**

11.61. Source

Usage: **source** *filename*

Source reads a command file of MIDAS commands. **Source** differs from **read** in that **source** will display the results of each command as it is executed while **read** only updates the display after all commands have completed.

The user may interrupt the execution of the *source* file by pressing the ESC key. MIDAS completes execution of the current command before processing the interrupt.

See also: **read**

11.62. Speed

Usage: **speed** *value*

Speed changes the speed of functions activated by pseudo-sliders or "spaceball". Thus, the sensitivity of the devices are altered for scaling and rotation functions. *Value* is a positive or negative integer which reflects the *relative* change in speed. The absolute range is 2 to 14 with default value of 10.

11.63. Stereo

Usage: **stereo** off | sequential | walleye | crosseye | lefteye | righteye

Stereo specifies how images are displayed. **Off** indicates a monocular image. **Sequential** indicates a time sequential stereographic image of the type that is utilized by "Crystal Eyes" stereo systems. Note that this command requires special hardware to work properly and not all workstations may support this hardware. **Walleye** indicates side-by-side stereo pairs having positive horizontal parallax. That is, the left eye image is shown on the left hand side of the display window and the right eye image is shown on the right hand side of the window. This is the classic method of generating stereo image pairs, but the size of the images are constrained not to overlap. **Crosseye** indicates side-by-side stereo pairs having negative horizontal parallax. That is, the left eye image is shown on the right hand side of the display window and the right eye image is shown on the left hand side of the window. The user must look "crosseyed" at the images to perceive the stereopsis effect. Image size may fill the entire window in this mode. **Lefteye** and **righteye** show the left and right view of a stereo image, respectively. This is useful for taking stereo slides since you can photograph each eye view individually.

The **stereo** command effectively obsoletes the **set stereo** and **set pair** options. (These obsolete commands will be removed in a future version of MidasPlus.) The command **~stereo** is equivalent to **stereo off**.

11.64. Stop

Usage: **stop**

Stop terminates the current MIDAS session without saving any of the currently displayed models.

See also: **save**

11.65. Surface

Usage: **surface** *atom_specification*

Surface selectively displays solvent accessible surface points for models. The syntax is the same as for the **display** command, except that it applies to surface points rather than to bonds.

To prepare a solvent accessible surface for display see "Solvent Accessible Surfaces" in Part II of this manual. The **open** command is used to open prepared MIDAS solvent accessible surface files followed by the **surface** command to display the surface.

Alternatively, the **vdw** command may be used to display the model's van der Waals surface. Display of this surface requires no advance calculation and is more convenient unless the solvent accessible surface is specifically desired.

Examples:

surface #0	display the surface for model 0
surface #1:5	display the surface for model 1 residue 5
~**surface** #1:5,32,64	remove surface for model 1 residues 5, 32 and 64

See also: **vdw**

11.66. Swapaa

Usage: **swapaa** *new_residue_type* [,preserve] *residue*

Swapaa replaces *residue* with a *new_residue_type* residue. Both the old and new residues must be standard amino acids. The side chain of the new residue will be in standard conformation unless the **preserve** keyword is given, in which case as much existing conformation as possible is saved. This can cause rings to become non-planar (*e.g.* swapping in a PHE for a LYS while preserving conformation).

The temperature factor for the new residue is set to the highest currently found in the model.

11.67. Swapna

Usage: **swapna** *new_residue_type* [,preserve] *residue*

Swapna replaces *residue* with a *new_residue_type* residue. Both the old and new residues must be standard nucleotides : A, T, G, C, or U.

The temperature factor for the new residue is set to the highest currently found in the model.

The keyword **preserve** is recognized, although it currently does not affect **swapna** behavior in any way.

11.68. System

Usage: **system** *command*

System executes the UNIX *command* under the user's preferred shell. *Command* may not be an interactive program. The output from *command* will appear as a REPLY on the graphics display screen. If the expected reply is more than five lines, the user should instead give the command:

!*command*

(note "!" mark) which directs output to the user's shell window.

As an alternative to the **system** command, the **midaspush** command may be used to access other screen windows or icons (*i.e.* go off and do something else for awhile and then return to MIDAS). If MIDAS was started in its own window, the standard IRIS window manipulation menu may also be use for this purpose. Lastly, one could also use the **save** command to retain orientations, rotation and slider assignments, *etc.* and then quit MIDAS entirely and return later for another modeling session.

See also: **run, pdbrun, midaspush, save**

11.69. Thickness

Usage: **thickness** *units* [*frames* [*wait_frames*]]

The **thickness** command changes the distance between the hither and yon clipping planes by the specified number of angstrom *units*. A positive *unit* value increases the distance between the clipping planes, whereas a negative value decreases the it. This results in displaying an increased or decreased serial cross section of the current model(s).

Frames moves the clipping planes in the specified manner for the specified number of image update frames. *Wait_frames*, if specified, indicates the number of frames to wait before beginning the move. **~thickness** will halt an ongoing thickness. *Frames* and *wait_frames* default to 1 and 0, respectively. These parameters are useful for controlling the rate of clipping and are helpful when constructing MIDAS command scripts and making videos.

See also: **section**

11.70. Turn

Usage: **turn** [*axis* [*angle* [*frames* [*wait_frames*]]]]

Turn functions the same as **roll** except that the default values for the *angle* and number of *frames* are 5 degrees and 1 frame, respectively. Thus, the command **turn** *y* will generate a left-eye stereo image, although **stereo lefteye** and **stereo righteye** are preferable since these commands generate a more technically accurate stereo image pair.

See also: **roll, stereo**

11.71. Update

Usage: **Update** transformed | original *filename*

Update changes the coordinates of a subset of atoms in a model. The user must supply either the **transformed** or **original** keywords and a PDB filename containing atom records with new atom coordinates. If PDB `MODEL` record(s) are present in the file, then the indicated models are updated, otherwise the lowest numbered model is updated. If the keyword is **transformed**, then the new coordinates are considered as having already been transformed by the current rotation and translation matrices. If the keyword is **original**, then the new coordinates are treated as untransformed coordinates and the current rotation and translation matrices are applied to the new coordinates before they are integrated with the rest of the model.

11.72. Vdw

Usage: **vdw** *atom_specification*

Vdw displays the van der Waals surface for the selected atoms. The syntax is the same as for the **display** command, except that it applies to surface points instead of bonds.

The **vdw** command may be interrupted by the ESC key. Atoms whose vdw surfaces were already computed when the interrupt occurred will have their surfaces displayed. Since computing the vdw surface is much faster than displaying it, pressing the ESC key might not help.

Note that the default vdw radii used by MIDAS assume that no explicit hydrogens are present in the model(s). This behavior can be changed with the **vdwopt** command.

See also: **vdwopt, surface**

11.73. Vdwopt

Usage: **vdwopt** [radii *file*] [density *value*] [hydrogen | default] [extend *leng*] [define *atom_type radius*]

Vdwopt sets user options for displaying van der Waals surfaces. The options are as follows:

radii *file*	Indicates a file containing alternate van der Waals radii. The alternate file must contain a complete set of radii for all atoms in the model. The file contains a series of records consisting of an atom name (character string) followed by a space and the atom radii in angstroms (floating point number). This is the same format as used by **ms** program (see Appendix 6). Optionally, a third field can specify the residue type. Note that once alternate radii have been selected the user may not "undo" the selection.
density *value*	The user may change the dot density of the displayed surface relative to the initial value of 1, which corresponds to 5 dots per square Angstrom. The density of dots varies linearly with the *value* provided. Thus, a density *value* of 2 gives 10 dots per square angstrom.
hydrogen **nuc** **prot**	Indicates that hydrogen atoms are included in the model and van der Waals radii should not compensate for missing hydrogens. The result is smaller atom radii and distinct hydrogen atoms. (**Hydrogen** is the preferred keyword. **Nuc** and **prot** are included for backwards compatibility.)
default	Indicates that van der Waals radii should compensate for missing hydrogen atoms. **Vdwopt default** essentially undoes the effects of **vdwopt hydrogen**.
extend *len*	Increases all van der Waals radii by the constant *len* angstroms.
define *atom_type radius*	Indicates that all atoms with an atomic symbol of *atom_type* should be assigned a van der Waals radius of *radius* angstroms.

See also: **vdw**

11.74. Wait

Usage: **wait**

Wait interrupts command processing until all model movement has ceased. Thus, if a **roll** has been implemented for a given number of frames, that motion is completed before the next command in the command file is executed.

The **wait** may be interrupted by pressing the ESC key. This breaks out of **wait** but does not freeze the screen (*i.e.* any active motion continues until completion).

See also: **sleep**

11.75. Watch/Watchopt

Usage: **watch** *atom_specification*
Usage: **watchopt** [distance *distance*]

Watch monitors interatomic distances. By default, it checks for distances less than the sum of the van der Waals radii of two atoms. Close contacts are displayed as yellow dotted lines, in the same manner as distance monitors. One can specify a fixed distance by using the **distance** option to the **watchopt** command, where *distance* is in Angstroms. A *distance* of zero means to use the default vdw radii for comparison. Only distances that potentially vary are checked, *i.e.*, atoms that are fixed relative to each other are not checked.

~**watch** will terminate **watch** monitoring.

See also: **distance**

11.76. Window

Usage: **window** [*atom_specification*]

With no arguments, **window** puts the entire image on the screen. The orientation of the structure is not changed. If the image has drifted off the screen, this is an excellent way of making it visible again.

If given an *atom_specification* as an argument, **window** will recalculate the view to enclose just those atoms instead of all of the models.

See also: **align, center, push/pop, reset, savepos**

11.77. Write

Usage: **write** [pdb | midas | ms | surface] [relative *n*] *model_number* [*filename*] [relative *n*]

This command causes the specified *model* to be written out. The format in which the model will be written out, PDB file or MIDAS database, will be the same as the format from which the model was opened. This default behavior can be overridden by supplying the **pdb** or **midas** keyword on the command line. Current bond rotations must be fixed before the **write** if they are to be reflected in the saved file(s).

The **relative** option specifies that the atomic coordinates written out are relative to the untransformed atomic coordinates of model *n*.

The **surface** option will rewrite the surface file in whatever format it was read in. The **ms** option will write the surface file in MS format.

Note that the **write** command is very slow when writing out a MIDAS database for a model that was opened as a PDB file. Instead, it is suggested that one should **write** the model as a PDB file and use the **midas.in** program (see Appendix 6) to produce the MIDAS database.

See also: **fix, save**

11.78. Version

Usage: **version**

Version reports information about which version of MIDAS is currently being executed and is displayed in the reply area of the MIDAS window. It is useful to supply this information when reporting MIDAS bugs, so that it is possible to determine if the problem has already been solved in a more recent version of the program.

11.79. Zone

Usage: **zone** *dist atoms*

Note that the zone command is obsolete (since zones can be indicated in atom specifiers now) and will be removed in a future release. It is implemented with an alias:

alias zone show z<

Consequently, the formerly operative keyword *preserve* no longer works.

Zone displays all atoms within *dist* angstroms of the specified *atoms* and undisplays all others. This is useful for depicting only the part of a protein near a substrate, for example.

See also: **display, show**

Appendix 1: Available Protein Data Bank Models

Id	Å	Molecule	Source	Depositors
3HVP	2.8	(ABA67,95)-HIV-1 PROTEASE (SF2 ISOLATE)	SYNTHETIC ENZYME ...	A.WLODAWER
1XY2	1.20	1 BETA-MERCAPTOPROPIONATE-OXYTOCIN (DRY FORM)	SYNTHETIC	S.COOPER
1XY1	1.04	1 BETA-MERCAPTOPROPIONATE-OXYTOCIN (WET FORM)	SYNTHETIC	J.HUSAIN
1PPD	2.0	2-HYDROXYETHYLTHIOPAPAIN (E.C.3.4.22.2)- CRYSTAL FORM D	PAPAYA (CARICA ...)	J.N.JANSONIUS
1KGA	3.5	2-KETO-3-DEOXY-6-PHOSPHOGLUCONATE (KDPG) ALDOLASE (E.C.4.1.2.14)	(PSEUDOMONAS PUTIDA)	A.TULINSKY
3INS	1.5	2ZN-INSULIN (JOINT X-RAY AND NEUTRON REFINEMENT)	PIG (SUS SCROFA)	A.WLODAWER,H.SAVAGE
4FAB	2.7	4-4-20 (IGG2A/KAPPA/) FAB FRAGMENT - FLUORESCEIN (DIANION) ...	MOUSE (MUS MUSCULUS)	J.N.HERRON,X.HE
2CRO	2.35	434 CRO PROTEIN	PHAGE 434	A.MONDRAGON
1R69	2.0	434 REPRESSOR (AMINO-TERMINAL DOMAIN) (R1-69)	PHAGE 434	A.MONDRAGON
2OR1	2.5	434 REPRESSOR (AMINO-TERMINAL DOMAIN) (R1-69) COMPLEX WITH ...	PHAGE 434	A.K.AGGARWAL
1HSC	2.2	44K ATPASE FRAGMENT (N-TERMINAL) OF 70K HEAT-SHOCK COGNATE ...	BOVINE (BOS TAURUS) BRAIN	K.M.FLAHERTY
9DNA	1.8	A-DNA-5(PRIME)-D(GPCPCPCGPGPGPC)-3(PRIME)	SYNTHETIC DNA	U.HEINEMANN
4APE	2.1	ACID PROTEINASE (E.C.3.4.23.10), ENDOTHIAPEPSIN	CHESTNUT BLIGHT ...	L.H.PEARL,B.T.SEWELL
3APP	1.8	ACID PROTEINASE (PENICILLOPEPSIN) (E.C.3.4.23.7)	FUNGUS (PENICILLIUM ...)	A.R.SIELECKI,M.N.G.JAMES
2APR	1.8	ACID PROTEINASE (RHIZOPUSPEPSIN) (E.C.3.4.23.6)	BREAD MOLD (RHIZOPUS ...)	K.SUGUNA,D.R.DAVIES
3APR	1.8	ACID PROTEINASE (RHIZOPUSPEPSIN) (E.C.3.4.23.6) COMPLEX WITH ...	BREAD MOLD (RHIZOPUS ...)	K.SUGUNA,D.R.DAVIES
6ACN	2.5	ACONITASE (E.C.4.2.1.3) (ACTIVATED (4FE-4S) CLUSTER FORM)	PIG (SUS SCROFA) HEART	A.H.ROBBINS,C.D.STOUT
5ACN	2.1	ACONITASE (E.C.4.2.1.3) (INACTIVE (3FE-4S) CLUSTER FORM)	PIG (SUS SCROFA) HEART	A.H.ROBBINS,C.D.STOUT
2ACT	1.7	ACTINIDIN (SULFHYDRYL PROTEINASE) (E.C. NUMBER NOT ASSIGNED)	CHINESE GOOSEBERRY ...	E.N.BAKER
1ACX	2.0	ACTINOXANTHIN	(ACTINOMYCES ...)	V.Z.PLETNEV,A.P.KUZIN
3ADK	2.1	ADENYLATE KINASE (E.C.2.7.4.3)	PORCINE (SUS SCROFA) ...	G.E.SCHULZ
1AGA	3.0	AGAROSE (AN ALTERNATING COPOLYMER OF 3-LINKED ...)	RED SEAWEED ...	S.ARNOTT
1AMT	1.5	ALAMETHICIN	(TRICHODERMA VIRIDE)	R.O.FOX,F.M.RICHARDS
5CHA	1.67	ALPHA CHYMOTRYPSIN A (E.C.3.4.21.1)	COW (BOS TAURUS)	R.A.BLEVINS,A.TULINSKY
6CHA	1.8	ALPHA CHYMOTRYPSIN A (E.C.3.4.21.1) COMPLEX WITH ...	COW (BOS TAURUS)	A.TULINSKY,R.A.BLEVINS
2CHA	2.0	ALPHA CHYMOTRYPSIN A (TOSYLATED) (E.C.3.4.21.1)	COW (BOS TAURUS)	J.J.BIRKTOFT,D.M.BLOW
1CTX	2.8	ALPHA COBRATOXIN	COBRA (NAJA NAJA ...)	W.SAENGER,M.D.WALKINSHAW
1HOE	2.0	ALPHA-AMYLASE INHIBITOR HOE-467A	(STREPTOMYCES TENDAE ...)	J.W.PFLUGRATH
2ABX	2.5	ALPHA-BUNGAROTOXIN	BRAIDED KRAIT ...	R.LOVE,R.STROUD
4CHA	1.68	ALPHA-CHYMOTRYPSIN (E.C.3.4.21.1)	COW (BOS TAURUS)	H.TSUKADA,D.M.BLOW
1CHO	1.8	ALPHA-CHYMOTRYPSIN (E.C.3.4.21.1) COMPLEX WITH TURKEY ...	BOVINE (BOS TAURUS) ...	M.FUJINAGA
1COH	2.9	ALPHA-FERROUS-CARBONMONOXY, BETA-COBALTOUS-DEOXY HEMOGLOBIN ...	HUMAN (HOMO SAPIENS)	B.LUISI
1ALC	1.7	ALPHA-LACTALBUMIN	BABOON (PAPIO ...)	K.R.ACHARYA
2ALP	1.7	ALPHA-LYTIC PROTEASE (E.C.3.4.21.12)	(LYSOBACTER ENZYMOGENES)	M.FUJINAGA
1P07	2.25	ALPHA-LYTIC PROTEASE (E.C.3.4.21.12) (MUTANT WITH MET 192 ...)	(LYSOBACTER ...)	R.BONE,D.A.AGARD
1P08	2.25	ALPHA-LYTIC PROTEASE (E.C.3.4.21.12) (MUTANT WITH MET 192 ...)	(LYSOBACTER ...)	R.BONE,D.A.AGARD
1P09	2.20	ALPHA-LYTIC PROTEASE (E.C.3.4.21.12) (MUTANT WITH MET 213 ...)	(LYSOBACTER ...)	R.BONE,D.A.AGARD
1P10	2.25	ALPHA-LYTIC PROTEASE (E.C.3.4.21.12) (MUTANT WITH MET 213 ...)	(LYSOBACTER ...)	R.BONE,D.A.AGARD
1P01	2.0	ALPHA-LYTIC PROTEASE (E.C.3.4.21.12) COMPLEX WITH ...	(LYSOBACTER ...)	R.BONE,D.A.AGARD
1P02	2.0	ALPHA-LYTIC PROTEASE (E.C.3.4.21.12) COMPLEX WITH ...	(LYSOBACTER ...)	R.BONE,D.A.AGARD
1P03	2.15	ALPHA-LYTIC PROTEASE (E.C.3.4.21.12) COMPLEX WITH ...	(LYSOBACTER ...)	R.BONE,D.A.AGARD
1P04	2.55	ALPHA-LYTIC PROTEASE (E.C.3.4.21.12) COMPLEX WITH ...	(LYSOBACTER ...)	R.BONE,D.A.AGARD
1P05	2.10	ALPHA-LYTIC PROTEASE (E.C.3.4.21.12) COMPLEX WITH ...	(LYSOBACTER ...)	R.BONE,D.A.AGARD
1P06	2.34	ALPHA-LYTIC PROTEASE (E.C.3.4.21.12) COMPLEX WITH ...	(LYSOBACTER ...)	R.BONE,D.A.AGARD
1TPA	1.9	ANHYDRO-TRYPSIN (E.C.3.4.21.4) COMPLEX WITH PANCREATIC ...	BOVINE (BOS TAURUS) ...	R.HUBER,W.BODE
2GD1	2.5	APO-D-GLYCERALDEHYDE-3-PHOSPHATE DEHYDROGENASE (E.C.1.2.1.12)	(BACILLUS ...)	T.SKARZYNSKI,A.J.WONACOTT
4GPD	2.8	APO-D-GYCERALDEHYDE-3-PHOSPHATE DEHYDROGENASE (E.C.1.2.1.12)	LOBSTER (HOMARUS ...)	J.P.GRIFFITH,S.SONG
5DFR	2.3	APO-DIHYDROFOLATE REDUCTASE (E.C.1.5.1.3) (DHFR)	(ESCHERICHIA COLI) ...	C.BYSTROFF,J.KRAUT
1LDB	2.8	APO-L-LACTATE DEHYDROGENASE (E.C.1.1.1.27)	(BACILLUS ...)	K.PIONTEK,M.G.ROSSMANN
2LDX	2.96	APO-LACTATE DEHYDROGENASE (E.C.1.1.1.27), ISOENZYME C/4/	MOUSE (MUS MUSCULUS) ...	J.P.GRIFFITH,M.G.ROSSMANN

Id	Å	Molecule	Source	Depositors
5ADH	2.9	APO-LIVER ALCOHOL DEHYDROGENASE (E.C.1.1.1.1) COMPLEX WITH ...	HORSE (EQUUS ...)	H.EKLUND,T.A.JONES
8ADH	2.4	APO-LIVER ALCOHOL DEHYDROGENASE (E.C.1.1.99.8)	HORSE (EQUUS ...)	T.A.JONES,H.EKLUND
2PCY	1.8	APO-PLASTOCYANIN (PH 6.0)	POPLAR (POPULUS ...)	T.P.J.GARRETT
1APD	N/A	APOLIPOPROTEIN D (MODEL)	HUMAN (HOMO SAPIENS)	M.C.PEITSCH,M.S.BOGUSKI
1TRM	2.3	ASN102TRYPSIN (E.C.3.4.21.4) (MUTANT WITH ASP 102 REPLACED ...)	RAT (RATTUS RATTUS)	S.SPRANG,T.STANDING
2TRM	2.8	ASN102TRYPSIN (E.C.3.4.21.4) (MUTANT WITH ASP 102 REPLACED ...)	RAT (RATTUS RATTUS)	R.M.STROUD,J.FINER-MOORE
2AAT	2.8	ASPARTATE AMINOTRANSFERASE (E.C.2.6.1.1) MUTANT K258A ...	(ESCHERICHIA COLI)	D.SMITH,S.ALMO
1AT1	2.8	ASPARTATE CARBAMOYLTRANSFERASE (ASPARTATE TRANSCARBAMYLASE) ...	(ESCHERICHIA COLI)	J.E.GOUAUX,W.N.LIPSCOMB
2AT1	2.8	ASPARTATE CARBAMOYLTRANSFERASE (ASPARTATE TRANSCARBAMYLASE) ...	(ESCHERICHIA COLI)	J.E.GOUAUX,W.N.LIPSCOMB
2ATC	3.0	ASPARTATE CARBAMOYLTRANSFERASE (ASPARTATE TRANSCARBAMYLASE) ...	(ESCHERICHIA COLI)	R.B.HONZATKO
3AT1	2.8	ASPARTATE CARBAMOYLTRANSFERASE (ASPARTATE TRANSCARBAMYLASE) ...	(ESCHERICHIA COLI)	J.E.GOUAUX,W.N.LIPSCOMB
4AT1	2.6	ASPARTATE CARBAMOYLTRANSFERASE (ASPARTATE TRANSCARBAMYLASE) ...	(ESCHERICHIA COLI)	R.C.STEVENS
5AT1	2.6	ASPARTATE CARBAMOYLTRANSFERASE (ASPARTATE TRANSCARBAMYLASE) ...	(ESCHERICHIA COLI)	R.C.STEVENS
6AT1	2.6	ASPARTATE CARBAMOYLTRANSFERASE (ASPARTATE TRANSCARBAMYLASE) ...	(ESCHERICHIA COLI)	R.C.STEVENS
7AT1	2.8	ASPARTATE CARBAMOYLTRANSFERASE (ASPARTATE TRANSCARBAMYLASE) ...	(ESCHERICHIA COLI)	J.E.GOUAUX
8AT1	2.8	ASPARTATE CARBAMOYLTRANSFERASE (ASPARTATE TRANSCARBAMYLASE) ...	(ESCHERICHIA COLI)	J.E.GOUAUX
8ATC	2.5	ASPARTATE CARBAMOYLTRANSFERASE (ASPARTATE TRANSCARBAMYLASE) ...	(ESCHERICHIA COLI)	H.KE,W.N.LIPSCOMB
1PPT	1.37	AVIAN PANCREATIC POLYPEPTIDE	TURKEY (MELEAGRIS ...)	T.L.BLUNDELL
1AZU	2.7	AZURIN	(PSEUDOMONAS AERUGINOSA)	E.T.ADMAN,L.C.SIEKER
2AZA	1.8	AZURIN (OXIDIZED)	(ALCALIGENES ...)	E.N.BAKER,G.E.NORRIS
1BD1	1.6	B-DNA-5(PRIME)-D(CPCPAPGPGPCPCPTPGPG)-3(PRIME)	SYNTHETIC DNA	U.HEINEMANN
3BCL	1.9	BACTERIOCHLOROPHYLL-A PROTEIN	(PROSTHECOCHLORIS ...)	D.TRONRUD,M.F.SCHMID
2BDS	N/A	BDS-I (NMR, 42 SIMULATED ANNEALING STRUCTURES)	SEA ANEMONE ...	G.M.CLORE
1BDS	N/A	BDS-I (NMR, MINIMIZED MEAN STRUCTURE)	SEA ANEMONE ...	G.M.CLORE
1BMV	3.0	BEAN POD MOTTLE VIRUS (MIDDLE COMPONENT)	BOUNTIFUL BEAN	J.E.JOHNSON
1REI	2.0	BENCE-JONES IMMUNOGLOBULIN REI VARIABLE PORTION	HUMAN (HOMO SAPIENS)	O.EPP,E.E.LATTMAN
2RHE	1.6	BENCE-JONES PROTEIN (LAMBDA, VARIABLE DOMAIN)	HUMAN (HOMO SAPIENS) ...	W.FUREY JUNIOR
4PTP	1.34	BETA TRYPSIN, DIISOPROPYLPHOSPHORYL INHIBITED (E.C.3.4.21.4)	BOVINE (BOS TAURUS) ...	J.L.CHAMBERS
3BLM	2.0	BETA-LACTAMASE (E.C.3.5.2.6)	(STAPHYLOCOCCUS ...)	O.HERZBERG,J.MOULT
2BLM	2.0	BETA-LACTAMASE (PENICILLINASE) (E.C.3.5.2.6)	(BACILLUS ...)	P.C.MOEWS,J.R.KNOX
3PTB	1.7	BETA-TRYPSIN (BENZAMIDINE INHIBITED) AT PH7 (E.C.3.4.21.4)	BOVINE (BOS TAURUS) ...	W.BODE,P.SCHWAGER
1TPP	1.4	BETA-TRYPSIN (E.C.3.4.21.4) COMPLEX WITH ...	BOVINE (BOS TAURUS) ...	J.WALTER,W.BODE,R.HUBER
2PTC	1.9	BETA-TRYPSIN (E.C.3.4.21.4) COMPLEX WITH PANCREATIC TRYPSIN ...	BOVINE (BOS TAURUS) ...	R.HUBER,J.DEISENHOFER
1TLD	1.5	BETA-TRYPSIN (ORTHORHOMBIC) AT PH 5.3 (E.C.3.4.21.4)	BOVINE (BOS TAURUS) ...	H.D.BARTUNIK
1TPO	1.7	BETA-TRYPSIN (ORTHORHOMBIC) AT PH5.0 (E.C.3.4.21.4)	BOVINE (BOS TAURUS) ...	W.BODE,J.WALTER,R.HUBER
6PTI	1.7	BOVINE PANCREATIC TRYPSIN INHIBITOR (BPTI,CRYSTAL FORM III)	BOVINE (BOS TAURUS) ...	A.WLODAWER
2P21	2.2	C-H-RAS P21 PROTEIN CATALYTIC DOMAIN	TRANSFORMED ...	S.-H.KIM
3P21	2.2	C-H-RAS P21 PROTEIN CATALYTIC DOMAIN (MUTANT WITH GLY 12 ...)	TRANSFORMED ...	S.-H.KIM
1CBH	N/A	C-TERMINAL DOMAIN OF CELLOBIOHYDROLASE I (CT-CBH I) ...	CHEMICALLY ...	G.M.CLORE,A.M.GRONENBORN
2CBH	N/A	C-TERMINAL DOMAIN OF CELLOBIOHYDROLASE I (CT-CBH I) ...	SYNTHETIC ...	G.M.CLORE,A.M.GRONENBORN
1CDP	1.6	CADMIUM-SUBSTITUTED CALCIUM-BINDING PARVALBUMIN B	CARP (CYPRINUS CARPIO)	A.L.SWAIN
4CPV	1.5	CALCIUM-BINDING PARVALBUMIN (PI=4.25)	CARP (CYPRINUS CARPIO)	V.D.KUMAR,L.LEE
5CPV	1.6	CALCIUM-BINDING PARVALBUMIN B	CARP (CYPRINUS CARPIO)	A.L.SWAIN
3ICB	2.3	CALCIUM-BINDING PROTEIN (VITAMIN D-DEPENDENT, MINOR A FORM) ...	BOVINE (BOS TAURUS) ...	D.M.E.SZEBENYI,K.MOFFAT
1PP2	2.5	CALCIUM-FREE PHOSPHOLIPASE A/2/ (E.C.3.1.1.4)	WESTERN DIAMONDBACK ...	S.BRUNIE,P.B.SIGLER
3CLN	2.2	CALMODULIN	RAT (RATTUS RATTUS)	Y.S.BABU,C.E.BUGG
2APK	N/A	CAMP DEPENDENT PROTEIN KINASE (E.C.2.7.1.37) TYPE II, DOMAIN ...	BOVINE (BOS ...)	I.T.WEBER
2BPK	N/A	CAMP DEPENDENT PROTEIN KINASE (E.C.2.7.1.37) TYPE II, DOMAIN ...	BOVINE (BOS ...)	I.T.WEBER
1APK	N/A	CAMP DEPENDENT PROTEIN KINASE (EC 2.7.1.37) TYPE I, DOMAIN A ...	BOVINE (BOS ...)	I.T.WEBER
1BPK	N/A	CAMP DEPENDENT PROTEIN KINASE (EC 2.7.1.37) TYPE I, DOMAIN B ...	BOVINE (BOS ...)	I.T.WEBER
1CAP	3.0	CAPSULAR POLYSACCHARIDE	(ESCHERICHIA COLI) ...	S.ARNOTT
2CAB	2.0	CARBONIC ANHYDRASE FORM B (CARBONATE DEHYDRATASE) (E.C.4.2.1.1)	HUMAN (HOMO SAPIENS) ...	K.K.KANNAN
1CA2	2.0	CARBONIC ANHYDRASE II (CARBONATE DEHYDRATASE) (HCA II) ...	HUMAN (HOMO SAPIENS) ...	A.E.ERIKSSON
2CA2	1.9	CARBONIC ANHYDRASE II (CARBONATE DEHYDRATASE) (HCA II) ...	HUMAN (HOMO SAPIENS) ...	A.E.ERIKSSON
3CA2	2.0	CARBONIC ANHYDRASE II (CARBONATE DEHYDRATASE) (HCA II) ...	HUMAN (HOMO SAPIENS) ...	A.E.ERIKSSON

Id	Å	Molecule	Source	Depositors
5CPA	1.54	CARBOXYPEPTIDASE A/ALPHA/ (COX) (E.C.3.4.17.1)	BOVINE (BOS TAURUS) ...	W.N.LIPSCOMB
3CPA	2.0	CARBOXYPEPTIDASE A/ALPHA/ (COX) (E.C.3.4.17.1) COMPLEX WITH ...	BOVINE (BOS TAURUS) ...	W.N.LIPSCOMB
4CPA	2.5	CARBOXYPEPTIDASE A/ALPHA/ (COX) (E.C.3.4.17.1) COMPLEX WITH ...	BOVINE (BOS TAURUS) ...	W.N.LIPSCOMB,D.C.REES
1CPB	2.8	CARBOXYPEPTIDASE B (E.C.3.4.12.3) FRACTION II	BOVINE (BOS TAURUS) ...	M.F.SCHMID,J.R.HERRIOTT
3GAP	2.5	CATABOLITE GENE ACTIVATOR PROTEIN - CYCLIC AMP COMPLEX (CAP)	(ESCHERICHIA COLI)	I.T.WEBER,T.A.STEITZ
2GAP	N/A	CATABOLITE GENE ACTIVATOR PROTEIN - DNA COMPLEX (MODEL)	(ESCHERICHIA COLI)	I.T.WEBER,T.A.STEITZ
4CAT	3.0	CATALASE (E.C.1.11.1.6)	(PENICILLIUM VITALE)	B.K.VAINSHTEIN
7CAT	2.5	CATALASE (E.C.1.11.1.6)	BEEF (BOS TAURUS) LIVER	M.R.N.MURTHY
8CAT	2.5	CATALASE (E.C.1.11.1.6)	BEEF (BOS TAURUS) LIVER	M.R.N.MURTHY
1CD4	2.3	CD4 (1 - 183 PLUS ASP - THR) (D1D2) (N-TERMINAL FRAGMENT OF ...)	RECOMBINANT HUMAN ...	S.-E.RYU,P.D.KWONG
2CD4	2.4	CD4(1-182) (N-TERMINAL FRAGMENT OF CD4 CONSISTING OF ...)	HUMAN (HOMO SAPIENS) ...	J.WANG,Y.YAN
3CBH	2.0	CELLOBIOHYDROLASE II CORE PROTEIN (E.C.3.2.1.91) (CBHII)	(TRICHODERMA REESEI)	T.A.JONES,J.ROUVINEN
2CHY	2.7	CHEY (MUTANT WITH SER 56 REPLACED BY CYS) (S56C)	(SALMONELLA ...)	J.M.MOTTONEN
2CLA	2.35	CHLORAMPHENICOL ACETYLTRANSFERASE (E.C.2.3.1.28) (CAT/III/) ...	(ESCHERICHIA COLI), ...	M.R.GIBBS
1C4S	3.0	CHONDROITIN-4-SULFATE (AN ALTERNATING COPOLYMER OF ...)	BOVINE (BOS TAURUS) ...	S.ARNOTT
2C4S	3.0	CHONDROITIN-4-SULFATE (AN ALTERNATING COPOLYMER OF ...)	SWARM RAT ...	S.ARNOTT
1CMS	2.3	CHYMOSIN B (FORMERLY KNOWN AS RENNIN) (E.C.3.4.23.4)	BOVINE (BOS TAURUS) ...	G.L.GILLILAND
2CI2	2.0	CHYMOTRYPSIN INHIBITOR 2 (CI-2)	BARLEY (HORDEUM ...)	C.A.MCPHALEN,M.N.G.JAMES
1CHG	2.5	CHYMOTRYPSINOGEN A	COW (BOS TAURUS)	S.T.FREER,J.KRAUT
2CGA	1.8	CHYMOTRYPSINOGEN A	BOVINE (BOS TAURUS) ...	D.WANG,W.BODE,R.HUBER
2CTS	2.0	CITRATE SYNTHASE (E.C.4.1.3.7) - (COA, CITRATE) COMPLEX	PIG (SUS SCROFA) HEART	S.REMINGTON
3CTS	1.7	CITRATE SYNTHASE (E.C.4.1.3.7) - (COA, CITRATE) COMPLEX	CHICKEN (GALLUS ...)	S.REMINGTON
1CTS	2.7	CITRATE SYNTHASE (E.C.4.1.3.7) - CITRATE COMPLEX	PIG (SUS SCROFA) HEART	S.REMINGTON
6CTS	2.5	CITRATE SYNTHASE (E.C.4.1.3.7) - CITRYLTHIOETHER - COENZYME ...	CHICKEN (GALLUS ...)	M.KARPUSAS
4CTS	2.9	CITRATE SYNTHASE (E.C.4.1.3.7) - OXALOACETATE COMPLEX	PIG (SUS SCROFA) HEART	S.REMINGTON
5CTS	1.9	CITRATE SYNTHASE (E.C.4.1.3.7)- OXALOACETATE - CARBOXYMETHYL ...	CHICKEN (GALLUS ...)	M.KARPUSAS
2CNA	2.0	CONCANAVALIN A	JACK BEAN (CANAVALIA ...)	G.N.REEKE JUNIOR
3CNA	2.4	CONCANAVALIN A	JACK BEAN (CANAVALIA ...)	K.D.HARDMAN,C.F.AINSWORTH
1CN1	3.2	CONCANAVALIN A (DEMETALLIZED)	JACK BEAN (CANAVALIA ...)	M.SHOHAM,A.YONATH
1CRN	1.5	CRAMBIN	ABYSSINIAN CABBAGE ...	W.A.HENDRICKSON
1CRO	2.2	CRO REPRESSOR	BACTERIOPHAGE (LAMBDA)	D.H.OHLENDORF
2SOD	2.0	CU,ZN SUPEROXIDE DISMUTASE (E.C.1.15.1.1)	BOVINE (BOS TAURUS) ...	J.A.TAINER
1CBP	2.5	CUCUMBER BASIC PROTEIN	CUCUMBER (CUCUMIS ...)	J.M.GUSS
3B5C	1.5	CYTOCHROME B5 (OXIDIZED)	BOVINE (BOS TAURUS) ...	F.S.MATHEWS,R.C.E.DURLEY
1CCR	1.5	CYTOCHROME C	RICE EMBRYOS (ORYZA ...)	H.OCHI,Y.HATA
3CYT	1.8	CYTOCHROME C (OXIDIZED)	ALBACORE TUNA ...	T.TAKANO
5CYT	1.5	CYTOCHROME C (REDUCED)	ALBACORE TUNA ...	T.TAKANO
2CYP	1.7	CYTOCHROME C PEROXIDASE (E.C.1.11.1.5) (FERROCYTOCHROME C ...)	BAKER,S YEAST ...	B.C.FINZEL
2CCY	1.67	CYTOCHROME C(PRIME)	(RHODOSPIRILLUM ...)	B.C.FINZEL,P.C.WEBER
2C2C	2.0	CYTOCHROME C/2/ (OXIDIZED)	(RHODOSPIRILLUM RUBRUM)	G.BHATIA,B.C.FINZEL
3C2C	1.68	CYTOCHROME C/2/ (REDUCED)	(RHODOSPIRILLUM RUBRUM)	G.BHATIA,B.C.FINZEL
1CY3	2.5	CYTOCHROME C/3/	(DESULFOVIBRIO ...)	R.HASER,M.FREY,F.PAYAN
2CDV	1.8	CYTOCHROME C/3/	(DESULFOVIBRIO ...)	Y.HIGUCHI,M.KUSUNOKI
1CC5	2.5	CYTOCHROME C/5/ (OXIDIZED)	(AZOTOBACTER VINELANDII)	C.D.STOUT,D.C.CARTER
351C	1.6	CYTOCHROME C/551/ (OXIDIZED)	(PSEUDOMONAS AERUGINOSA)	Y.MATSUURA,T.TAKANO
451C	1.6	CYTOCHROME C/551/ (REDUCED)	(PSEUDOMONAS AERUGINOSA)	Y.MATSUURA,T.TAKANO
155C	2.5	CYTOCHROME C550	(PARACOCCUS ...)	R.TIMKOVICH
3CPP	1.9	CYTOCHROME P450CAM (CAMPHOR MONOOXYGENASE) (E.C.1.14.15.1) - ...	(PSEUDOMONAS PUTIDA)	R.RAAG,T.L.POULOS
2CPP	1.63	CYTOCHROME P450CAM (CAMPHOR MONOOXYGENASE) (E.C.1.14.15.1) ...	(PSEUDOMONAS PUTIDA)	T.L.POULOS
4MDH	2.5	CYTOPLASMIC MALATE DEHYDROGENASE (E.C.1.1.1.37)	PORCINE (SUS SCROFA) ...	J.J.BIRKTOFT,L.J.BANASZAK
1AAT	2.8	CYTOSOLIC ASPARTATE AMINOTRANSFERASE (E.C.2.6.1.1) COMPLEX ...	CHICKEN (GALLUS ...)	E.G.HARUTYUNYAN
2CP1	N/A	CYTOTOXIC T-LYMPHOCYTE PROTEINASE 1 (CCP1) (MODEL)	MOUSE (MUS MUSCULUS)	M.MURPHY,M.N.G.JAMES
256B	1.4	CYTROCHROME B562 (OXIDIZED)	(ESCHERICHIA COLI)	K.HAMADA,P.H.BETHGE
1PTE	2.8	D-ALANYL-D-ALANINE CARBOXYPEPTIDASE(SLASH)TRANSPEPTIDASE	(STREPTOMYCES R61)	J.A.KELLY,J.R.KNOX
2GBP	1.9	D-GALACTOSED-GLUCOSE BINDING PROTEIN (GGBP)	(ESCHERICHIA COLI ...)	N.K.VYAS,M.N.VYAS

Id	Å	Molecule	Source	Depositors
1PGI	3.5	D-GLUCOSE 6-PHOSPHATE ISOMERASE (E.C.5.3.1.9)	PORCINE (SUS SCROFA) ...	H.MUIRHEAD
3GPD	3.5	D-GLYCERALDEHYDE-3-PHOSPHATE DEHYDROGENASE (E.C.1.2.1.12)	HUMAN (HOMO SAPIENS) ...	H.C.WATSON,J.C.CAMPBELL
1GPD	2.9	D-GYCERALDEHYDE-3-PHOSPHATE DEHYDROGENASE (E.C.1.2.1.12)	LOBSTER (HOMARUS ...)	D.MORAS,K.W.OLSEN
1XIA	2.3	D-XYLOSE ISOMERASE (E.C.5.3.1.5)	(ARTHROBACTER, ...)	D.M.BLOW
2XIA	3.5	D-XYLOSE ISOMERASE (E.C.5.3.1.5)	(STREPTOMYCES ...)	H.L.CARRELL
3XIA	3.0	D-XYLOSE ISOMERASE (E.C.5.3.1.5)	(STREPTOMYCES ...)	G.FARBER,G.PETSKO
4XIA	2.3	D-XYLOSE ISOMERASE (E.C.5.3.1.5), D-SORBITOL COMPLEX	(ARTHROBACTER, ...)	K.HENRICK
5XIA	2.5	D-XYLOSE ISOMERASE (E.C.5.3.1.5), XYLITOL COMPLEX	(ARTHROBACTER, ...)	K.HENRICK
4TSI	2.5	DES-(ILE 318-ARG 417)-TYROSYL-TRANSFER RNA SYNTHETASE ...	(BACILLUS ...)	P.BRICK,D.M.BLOW
2INS	2.5	DES-PHE B1 INSULIN	BOVINE (BOS TAURUS)	G.D.SMITH,W.L.DUAX
8DFR	1.7	DIHYDROFOLATE REDUCTASE (E.C.1.5.1.3)	CHICKEN (GALLUS ...)	D.A.MATTHEWS
2DHF	2.3	DIHYDROFOLATE REDUCTASE (E.C.1.5.1.3) (DHFR) COMPLEX WITH ...	HUMAN (HOMO SAPIENS) ...	J.F.DAVIES II,J.KRAUT
7DFR	2.5	DIHYDROFOLATE REDUCTASE (E.C.1.5.1.3) (DHFR) COMPLEX WITH ...	(ESCHERICHIA COLI) ...	C.BYSTROFF
1DHF	2.3	DIHYDROFOLATE REDUCTASE (E.C.1.5.1.3) (DHFR) COMPLEX WITH FOLATE	HUMAN (HOMO SAPIENS) ...	J.F.DAVIES II,J.KRAUT
6DFR	2.4	DIHYDROFOLATE REDUCTASE (E.C.1.5.1.3) (DHFR) COMPLEX WITH NADP+	(ESCHERICHIA COLI) ...	C.BYSTROFF
4DFR	1.7	DIHYDROFOLATE REDUCTASE (E.C.1.5.1.3) COMPLEX WITH METHOTREXATE	(ESCHERICHIA COLI ...)	D.J.FILMAN
3DFR	1.7	DIHYDROFOLATE REDUCTASE (E.C.1.5.1.3) COMPLEX WITH NADPH AND ...	(LACTOBACILLUS ...)	D.J.FILMAN
1DNN	N/A	DNA (5(PRIME)-D((APTPCPGPGPCPTPAPAPG))- 3(PRIME)) MODEL	NOT APPLICABLE ...	I.L.SUSSMAN,E.N.TRIFONOV
1D13	2.0	DNA (5(PRIME)-D(APCPCPGPGPCPCPGPGPT)-3(PRIME))	SYNTHETIC DNA	C.A.FREDERICK
1D12	1.7	DNA (5(PRIME)-D(CPGPAPTPCPG)-3(PRIME)) COMPLEX WITH ADRIAMYCIN	SYNTHETIC DNA	C.A.FREDERICK
1D10	1.5	DNA (5(PRIME)-D(CPGPAPTPCPG)-3(PRIME)) COMPLEX WITH DAUNOMYCIN	SYNTHETIC DNA	C.A.FREDERICK
1DN9	2.2	DNA (5(PRIME)-D(CPGPCPAPTPAPTPAPTPGPCPGP)-3(PRIME))	SYNTHETIC	C.YOON,R.E.DICKERSON
1DNF	1.5	DNA (5(PRIME)-D(CPGPCPGPUFPGP)-3(PRIME))	SYNTHETIC	M.COLL,D.SAAL
1D14	1.5	DNA (5(PRIME)-D(CPGPTPAPCPG)-3(PRIME)) (A 4 IS ...)	SYNTHETIC DNA	L.D.WILLIAMS,M.EGLI
1D11	1.2	DNA (5(PRIME)-D(CPGPTPAPCPG)-3(PRIME)) COMPLEX WITH DAUNOMYCIN	SYNTHETIC DNA	A.H.-J.WANG
5ANA	2.25	DNA (5(PRIME)-D(GPTPAPCPGPTPAPCP)-3(PRIME))	SYNTHETIC	F.TAKUSAGAWA
1DNS	2.0	DNA (5(PRIME)-D(GPTPGPTPAPCPAPCP)) COMPLEX WITH SPERMINE	SYNTHETIC	M.SUNDARALINGAM
1ANA	2.1	DNA (A, 5(PRIME)-D(IGPG)-3(PRIME))	SYNTHETIC DNA	B.N.CONNER,R.E.DICKERSON
2ANA	2.5	DNA (A,5(PRIME)-D(GPGPGPGPCPCPCPC)-3(PRIME))	SYNTHETIC DNA	M.MCCALL,T.BROWN
6BNA	2.21	DNA (B, 5(PRIME)- D(CPGPCPGPAPAPTPTP=/BR/=CPGPCPG)-3(PRIME)) ...	SYNTHETIC DNA	M.L.KOPKA,C.YOON
3BNA	3.0	DNA (B, 5(PRIME)- D(CPGPCPGPAPAPTPTPBRCPGPCPG)-3(PRIME)) (60 ...)	SYNTHETIC DNA	M.L.KOPKA
4BNA	2.3	DNA (B, 5(PRIME)- D(CPGPCPGPAPAPTPTPBRCPGPCPG)-3(PRIME)) (60 ...)	SYNTHETIC DNA	M.L.KOPKA
8BNA	2.2	DNA (B, 5(PRIME)- D(CPGPCPGPAPAPTPTPCPGPCPG)-3(PRIME)) ...	SYNTHETIC DNA	P.PJURA
2BNA	2.7	DNA (B, 5(PRIME)-D(CPGPCPGPAPAPTPTPCPGPCPG)-3(PRIME)) (16 ...)	SYNTHETIC DNA	H.R.DREW,R.E.DICKERSON
1BNA	1.9	DNA (B, 5(PRIME)-D(CPGPCPGPAPAPTPTPCPGPCPG)-3(PRIME)) (290 ...)	SYNTHETIC DNA	H.R.DREW,R.E.DICKERSON
7BNA	1.9	DNA (B, 5(PRIME)-D(CPGPCPGPAPAPTPTPCPGPCPG)-3(PRIME)) (290 ...)	SYNTHETIC DNA	S.R.HOLBROOK
5BNA	2.6	DNA (B, 5-D(CPGPCPGPAPAPTPTPCPGPCPG)-3) COMPLEX WITH CISPLATIN	SYNTHETIC DNA	R.WING,P.PJURA
1ZNA	1.6	DNA (Z, 5-D(CPGPCPG)-3, HIGH SALT)	SYNTHETIC DNA	H.R.DREW,R.E.DICKERSON
2ZNA	N/A	DNA (Z-I, 5-D(PCPGPCPGPCP6PCPGPCPGPCPG)-3)	SYNTHETIC DNA	A.H.-J.WANG
3ZNA	N/A	DNA (Z-II, 5-D(PCPGPCPGPCPGPCPGPCPGPCPG)-3)	SYNTHETIC DNA	A.H.-J.WANG
1DPI	2.8	DNA POLYMERASE I (KLENOW FRAGMENT) (E.C.2.7.7.7) - DCMP COMPLEX	(ESCHERICHIA COLI)	L.BEESE,D.OLLIS,T.STEITZ
2DND	2.2	DNA-5(PRIME)- D(CPGPCPAPAPAPTPTPTPGPCPG)-3(PRIME)) COMPLEX ...)	SYNTHETIC DNA	M.COLL,C.A.FREDERICK
1DNH	2.25	DNA-5(PRIME)- D(CPGPCPGPAPAPTPTPCPGPCPG)-3(PRIME)) COMPLEX ...)	SYNTHETIC DNA	M.-K.TENG,N.USMAN
4DNB	2.0	DNA-5(PRIME)- D(CPGPCPGPAPM6APTPTPCPGPCPG)-3(PRIME)	SYNTHETIC DNA	C.A.FREDERICK
1DNE	2.4	DNA-5(PRIME)- D(CPGPCPGPAPTPAPTPCPGPCPG)-3(PRIME)) COMPLEX ...)	SYNTHETIC DNA	M.COLL,J.AYMAMI
1D16	2.1	DNA-5(PRIME)- D(CPGPCPGPCPGPTPTPTPTPTPCPGPCPGP G)-3(PRIME)	SYNTHETIC DNA	R.CHATTOPADHYAYA
1DN7	N/A	DNA-5(PRIME)-D(GPGPGPGGPGPGPGGPGPGPGGPGPGPG) ...	SYNTHETIC	M.J.MCCALL,T.BROWN
1DN4	1.4	DNA-5(PRIME)-D(5BRCPGP5BRCPGP5BRCPGP)-3(PRIME)) (18 DEGREES C)	SYNTHETIC DNA	B.CHEVRIER,A.C.DOCK
1DN5	1.4	DNA-5(PRIME)-D(5BRCPGP5BRCPGP5BRCPGP)-3(PRIME)) (37 DEGREES C)	SYNTHETIC DNA	B.CHEVRIER,A.C.DOCK
3DNB	1.3	DNA-5(PRIME)-D(CPCPAPAPGPAPTPTPGPG)-3(PRIME)	SYNTHETIC DNA	G.G.PRIVE,R.E.DICKERSON
1DCG	1.0	DNA-5(PRIME)-D(CPGPCPGPCPG)-3(PRIME) COMPLEX WITH MAGNESIUM	SYNTHETIC DNA	R.V.GESSNER
2DCG	0.9	DNA-5(PRIME)-D(CPGPCPGPCPG)-3(PRIME) COMPLEX WITH MAGNESIUM ...	SYNTHETIC DNA	A.H.-J.WANG
1DN6	3.0	DNA-5(PRIME)-D(GPGPAPTPGPGPGPAPG)-3(PRIME)	SYNTHETIC DNA	M.J.MCCALL,T.BROWN
3ANA	2.5	DNA-5(PRIME)-D(GPGPGPAPTPCPCPC)-3(PRIME) (A CONFORMATION)	SYNTHETIC DNA	U.HEINEMANN,H.LAUBLE
1DN8	1.5	DNA-5(PRIME)-D(PCPGPTPAPCPGPTPAPCPG) - COBALT HEXAMMINE COMPLEX	SYNTHETIC DNA	M.SUNDARALINGAM

Id	Å	Molecule	Source	Depositors
2EST	2.5	ELASTASE (E.C.3.4.21.11) COMPLEX WITH TRIFLUOROACETYL ...	PORCINE (SUS SCROFA) ...	L.C.SIEKER,D.L.HUGHES
1ETU	2.9	ELONGATION FACTOR TU (DOMAIN I) - GUANOSINE DIPHOSPHATE COMPLEX	(ESCHERICHIA COLI B)	B.F.C.CLARK
2ER0	3.0	ENDOTHIA ASPARTIC PROTEINASE (ENDOTHIAPEPSIN) (E.C.3.4.23.6) ...	CHESTNUT BLIGHT ...	J.B.COOPER
2ER6	2.0	ENDOTHIA ASPARTIC PROTEINASE (ENDOTHIAPEPSIN) (E.C.3.4.23.6) ...	CHESTNUT BLIGHT ...	J.B.COOPER
2ER7	1.6	ENDOTHIA ASPARTIC PROTEINASE (ENDOTHIAPEPSIN) (E.C.3.4.23.6) ...	CHESTNUT BLIGHT ...	B.VEERAPANDIAN
2ER9	2.2	ENDOTHIA ASPARTIC PROTEINASE (ENDOTHIAPEPSIN) (E.C.3.4.23.6) ...	CHESTNUT BLIGHT ...	J.B.COOPER
4ER1	2.0	ENDOTHIA ASPARTIC PROTEINASE (ENDOTHIAPEPSIN) (E.C.3.4.23.6) ...	CHESTNUT BLIGHT ...	J.W.QUAIL,J.B.COOPER
4ER2	2.0	ENDOTHIA ASPARTIC PROTEINASE (ENDOTHIAPEPSIN) (E.C.3.4.23.6) ...	CHESTNUT BLIGHT ...	D.BAILEY
5ER1	2.0	ENDOTHIA ASPARTIC PROTEINASE (ENDOTHIAPEPSIN) (E.C.3.4.23.6) ...	CHESTNUT BLIGHT ...	J.B.COOPER
2ENL	2.25	ENOLASE (E.C.4.2.1.11) (2-PHOSPHO-D-GLYCERATE HYDROLASE)	BAKER'S YEAST ...	L.LEBIODA,B.STEC
5EBX	2.0	ERABUTOXIN A	SEA SNAKE (LATICAUDA ...)	P.W.R.CORFIELD
3EBX	1.4	ERABUTOXIN B	SEA SNAKE (LATICAUDA ...)	J.L.SMITH
1IGE	N/A	FC FRAGMENT (IGE(PRIME)CL) (MODEL)	HUMAN (HOMO SAPIENS)	E.A.PADLAN,D.R.DAVIES
1FC1	2.9	FC FRAGMENT (IGG1 CLASS)	HUMAN (HOMO SAPIENS) ...	J.DEISENHOFER
1FDX	2.0	FERREDOXIN	(PEPTOCOCCUS AEROGENES)	E.T.ADMAN,L.C.SIEKER
1FXB	2.3	FERREDOXIN	(BACILLUS ...)	K.FUKUYAMA
3FXC	2.5	FERREDOXIN	(SPIRULINA PLATENSIS)	M.KAKUDO,T.TSUKIHARA
4FD1	1.9	FERREDOXIN	(AZOTOBACTER ...)	C.D.STOUT
1FD2	1.9	FERREDOXIN (MUTANT WITH CYS 20 REPLACED BY ALA) (C20A)	(AZOTOBACTER ...)	C.D.STOUT
2FD2	1.9	FERREDOXIN (MUTANT WITH CYS 24 REPLACED BY ALA) (C24A)	(AZOTOBACTER ...)	C.D.STOUT
1CYC	2.3	FERROCYTOCHROME C	BONITO (KATSUWONUS ...)	N.TANAKA,T.YAMANE
1FCB	2.4	FLAVOCYTOCHROME B/2/ (E.C.1.1.2.3)	YEAST (SACCHAROMYCES ...)	F.S.MATHEWS,Z.-X.XIA
1FX1	2.0	FLAVODOXIN	(DESULFOVIBRIO VULGARIS)	K.D.WATENPAUGH
3FXN	1.9	FLAVODOXIN (OXIDIZED FORM)	(CLOSTRIDIUM MP)	M.L.LUDWIG
4FXN	1.8	FLAVODOXIN (SEMIQUINONE FORM)	(CLOSTRIDIUM MP)	M.L.LUDWIG
1GBP	3.0	GALACTOSE-BINDING PROTEIN	(SALMONELLA ...)	S.L.MOWBRAY,G.A.PETSKO
3GCH	1.9	GAMMA CHYMOTRYPSIN (E.C.3.4.21.1) COMPLEX WITH ...	BOVINE (BOS TAURUS) ...	B.L.STODDARD,D.RINGE
4GCH	1.9	GAMMA CHYMOTRYPSIN (E.C.3.4.21.1) COMPLEX WITH ...	BOVINE (BOS TAURUS) ...	B.L.STODDARD,D.RINGE
6GCH	2.1	GAMMA CHYMOTRYPSIN (E.C.3.4.21.1) WITH ...	BOVINE (BOS TAURUS) ...	K.BRADY,A.WEI
7GCH	1.8	GAMMA CHYMOTRYPSIN (E.C.3.4.21.1) WITH ...	BOVINE (BOS TAURUS) ...	K.BRADY,D.RINGE
2GCH	1.9	GAMMA CHYMOTRYPSIN A (E.C.3.4.21.1)	BOVINE (BOS TAURUS) ...	G.H.COHEN,D.R.DAVIES
2GCR	2.3	GAMMA IVA-CRYSTALLIN	BOVINE LENS (BOS TAURUS)	H.E.WHITE
1GCR	1.6	GAMMA-II CRYSTALLIN	CALF (BOS TAURUS) ...	C.SLINGSBY,P.LINDLEY
2GN5	2.3	GENE 5 DNA BINDING PROTEIN	FILAMENTOUS ...	G.D.BRAYER,A.MCPHERSON
1GCN	3.0	GLUCAGON (PH 6 - PH 7 FORM)	PORCINE (SUS SCROFA) ...	T.L.BLUNDELL
2GLS	3.5	GLUTAMINE SYNTHETASE (E.C.6.3.1.2)	(SALMONELLA TYPHIMURIUM)	D.EISENBERG
1GP1	2.0	GLUTATHIONE PEROXIDASE (E.C.1.11.1.9)	BOVINE (BOS TAURUS) ...	O.EPP,R.LADENSTEIN
3GRS	1.54	GLUTATHIONE REDUCTASE (E.C.1.6.4.2), OXIDIZED FORM (E)	HUMAN (HOMO SAPIENS) ...	G.E.SCHULZ,P.A.KARPLUS
1GOX	2.0	GLYCOLATE OXIDASE (E.C.1.1.3.1)	SPINACH (SPINACIA ...)	Y.LINDQVIST
1GMA	0.86	GRAMICIDIN A	(BACILLUS BREVIS)	D.A.LANGS
1HF1	N/A	HANNUKA FACTOR (MODEL)	HUMAN (HOMO SAPIENS)	M.MURPHY,M.N.G.JAMES
5HMG	3.2	HEMAGGLUTININ (D112(B)G) (BROMELAIN DIGESTED) (MUTANT WITH ...)	INFLUENZA VIRUS ...	W.I.WEIS
2HMG	3.0	HEMAGGLUTININ (G146(A)D) (BROMELAIN DIGESTED) (MUTANT WITH ...)	INFLUENZA VIRUS ...	W.I.WEIS
3HMG	2.9	HEMAGGLUTININ (L226(A)Q) (BROMELAIN DIGESTED) (MUTANT WITH ...)	INFLUENZA VIRUS ...	W.I.WEIS
4HMG	3.0	HEMAGGLUTININ (L226(A)Q) (BROMELAIN DIGESTED) (MUTANT WITH ...)	INFLUENZA VIRUS ...	W.I.WEIS
1HMZ	2.0	HEMERYTHRIN (AZIDO, MET)	SIPUNCULID WORM ...	R.E.STENKAMP
1HR3	5.5	HEMERYTHRIN (AZIDO,MET)	(SIPHONOSOMA SPECIES ...)	J.L.SMITH
1HMQ	2.0	HEMERYTHRIN (MET)	SIPUNCULID WORM ...	R.E.STENKAMP
1HRB	5.5	HEMERYTHRIN B	MARINE WORM ...	W.A.HENDRICKSON,K.B.WARD
1HCO	2.7	HEMOGLOBIN (CARBONMONOXY)	HUMAN (HOMO SAPIENS)	J.M.BALDWIN
2HCO	2.7	HEMOGLOBIN (CARBONMONOXY)	HUMAN (HOMO SAPIENS)	J.M.BALDWIN
2HHB	1.74	HEMOGLOBIN (DEOXY)	HUMAN (HOMO SAPIENS)	G.FERMI,M.F.PERUTZ
3HHB	1.74	HEMOGLOBIN (DEOXY)	HUMAN (HOMO SAPIENS)	G.FERMI,M.F.PERUTZ
4HHB	1.74	HEMOGLOBIN (DEOXY)	HUMAN (HOMO SAPIENS)	G.FERMI,M.F.PERUTZ
1FDH	2.5	HEMOGLOBIN (DEOXY, HUMAN FETAL F/II/)	HUMAN FETUS (HOMO ...)	J.A.FRIER JUNIOR

Id	Å	Molecule	Source	Depositors
1ECA	1.4	HEMOGLOBIN (ERYTHROCRUORIN, AQUO MET)	(CHIRONOMOUS THUMMI ...)	W.STEIGEMANN,E.WEBER
1ECO	1.4	HEMOGLOBIN (ERYTHROCRUORIN, CARBONMONOXY)	(CHIRONOMOUS THUMMI ...)	W.STEIGEMANN,E.WEBER
1ECN	1.4	HEMOGLOBIN (ERYTHROCRUORIN, CYANO MET)	(CHIRONOMOUS THUMMI ...)	W.STEIGEMANN,E.WEBER
1ECD	1.4	HEMOGLOBIN (ERYTHROCRUORIN, DEOXY)	(CHIRONOMOUS THUMMI ...)	W.STEIGEMANN,E.WEBER
2MHB	2.0	HEMOGLOBIN (HORSE, AQUO MET)	HORSE (EQUUS CABALLUS)	R.C.LADNER
2DHB	2.8	HEMOGLOBIN (HORSE,DEOXY)	HORSE (EQUUS CABALLUS)	M.F.PERUTZ ET AL.
1HDS	1.98	HEMOGLOBIN (SICKLE CELL)	VIRGINIA ...	E.L.AMMA,R.L.GIRLING
1THB	1.5	HEMOGLOBIN (T STATE, PARTIALLY OXYGENATED)	HUMAN (HOMO SAPIENS)	D.A.WALLER,R.C.LIDDINGTON
1HHO	2.1	HEMOGLOBIN A (OXY)	HUMAN (HOMO SAPIENS)	B.SHAANAN
1HBS	3.0	HEMOGLOBIN S (DEOXY)	HUMAN (HOMO SAPIENS)	E.A.PADLAN,W.E.LOVE
2LHB	2.0	HEMOGLOBIN V (CYANO,MET)	SEA LAMPREY ...	R.B.HONZATKO
1HKG	3.5	HEXOKINASE A AND GLUCOSE COMPLEX (E.C.2.7.1.1)	YEAST (SACCHAROMYCES ...)	W.S.BENNETT JUNIOR
6HIR	N/A	HIRUDIN (MUTANT WITH LYS 47 REPLACED BY GLU) (K47E) (NMR, ...)	LEECH (HIRUDO ...)	G.M.CLORE,A.M.GRONENBORN
4HIR	N/A	HIRUDIN (MUTANT WITH LYS 47 REPLACED BY GLU) (K47E) (NMR,32 ...)	LEECH (HIRUDO ...)	G.M.CLORE,A.M.GRONENBORN
5HIR	N/A	HIRUDIN (WILD-TYPE)	LEECH (HIRUDO ...)	G.M.CLORE,A.M.GRONENBORN
2HIR	N/A	HIRUDIN (WILD-TYPE) (NMR,32 SIMULATED ANNEALING STRUCTURES)	LEECH (HIRUDO ...)	G.M.CLORE,A.M.GRONENBORN
2HVP	3.0	HIV-1 PROTEASE	ESCHERICHIA COLI (IN ...)	M.A.NAVIA
4HVP	2.3	HIV-1 PROTEASE (HIV-1 PR) COMPLEX WITH INHIBITOR ...	SYNTHETIC ENZYME ...	M.MILLER,J.SCHNEIDER
1HVP	N/A	HIV-1 PROTEASE COMPLEX WITH SUBSTRATE (MODEL)	HIV-1 RETROVIRUS ...	L.T.WEBER
1GD1	1.8	HOLO-D-GLYCERALDEHYDE-3-PHOSPHATE DEHYDROGENASE (E.C.1.2.1.12)	(BACILLUS ...)	T.SKARZYNSKI
6ADH	2.9	HOLO-LIVER ALCOHOL DEHYDROGENASE (E.C.1.1.1.1) COMPLEX WITH ...	HORSE (EQUUS ...)	H.EKLUND
1HLA	3.5	HUMAN CLASS I HISTOCOMPATIBILITY ANTIGEN A2 (HLA-A2, HUMAN ...)	HUMAN (HOMO SAPIENS) ...	P.J.BJORKMAN
3HLA	2.6	HUMAN CLASS I HISTOCOMPATIBILITY ANTIGEN A2.1 (HLA-A2.1 ...)	HUMAN (HOMO SAPIENS) ...	M.A.SAPER
2HLA	2.6	HUMAN CLASS I HISTOCOMPATIBILITY ANTIGEN AW 68.1 (HLA-AW ...)	HUMAN (HOMO SAPIENS) ...	T.P.J.GARRETT
1HNE	1.84	HUMAN NEUTROPHIL ELASTASE (HNE) (E.C.3.4.21.37) (ALSO ...)	HUMAN (HOMO SAPIENS) ...	M.A.NAVIA
1HYA	3.0	HYALURONIC ACID (POLY D-GLUCURONIC ...)	SYNTHESIZED BY ...	S.ARNOTT
2HYA	3.0	HYALURONIC ACID (POLY D-GLUCURONIC ...)	HUMAN (HOMO SAPIENS) ...	S.ARNOTT
3HYA	3.0	HYALURONIC ACID (POLY D-GLUCURONIC ...)	HUMAN (HOMO SAPIENS) ...	S.ARNOTT
4HYA	3.0	HYALURONIC ACID (POLY D-GLUCURONIC ...)	HUMAN (HOMO SAPIENS) ...	S.ARNOTT
2FBJ	1.95	IGA FAB FRAGMENT (J539) (GALACTAN-BINDING)	MOUSE (MUS MUSCULUS)	T.N.BHAT,E.A.PADLAN
1FVB	N/A	IGA FV FRAGMENT (19.1.2, ANTI-ALPHA(1(RIGHT ARROW)6) ...)	MOUSE (MUS MUSCULUS)	E.A.PADLAN,E.A.KABAT
2FVB	N/A	IGA FV FRAGMENT (19.1.2, ANTI-ALPHA(1(RIGHT ARROW)6) ...)	MOUSE (MUS MUSCULUS)	E.A.PADLAN,E.A.KABAT
1FVW	N/A	IGA FV FRAGMENT (W3129, ANTI-ALPHA(1(RIGHT ARROW)6) DEXTRAN) ...	MOUSE (MUS MUSCULUS)	E.A.PADLAN,E.A.KABAT
2FVW	N/A	IGA FV FRAGMENT (W3129, ANTI-ALPHA(1(RIGHT ARROW)6) DEXTRAN) ...	MOUSE (MUS MUSCULUS)	E.A.PADLAN,E.A.KABAT
3HFM	3.0	IGG1 FAB FRAGMENT (HYHEL-10) AND LYSOZYME (E.C.3.2.1.17) COMPLEX	MOUSE (MUS MUSCULUS) ...	E.A.PADLAN,D.R.DAVIES
2HFL	2.54	IGG1 FAB FRAGMENT (HYHEL-5) AND LYSOZYME (E.C.3.2.1.17) COMPLEX	BALB(SLASH)C MOUSE ...	S.SHERIFF,D.R.DAVIES
1HFM	N/A	IGG1 FV FRAGMENT (HYHEL-10) (MODEL)	MOUSE (MUS MUSCULUS)	S.J.SMITH-GILL
2HFM	N/A	IGG1 FV FRAGMENT (HYHEL-10) AND LYSOZYME (E.C.3.2.1.17) ...	MOUSE (MUS MUSCULUS) ...	S.J.SMITH-GILL
2FB4	1.9	IMMUNOGLOBULIN FAB	HUMAN (HOMO SAPIENS) ...	M.MARQUART,R.HUBER
1MCP	2.7	IMMUNOGLOBULIN FAB FRAGMENT (MCPC603)	MOUSE (MUS MUSCULUS)	Y.SATOW,G.H.COHEN
1FC2	2.8	IMMUNOGLOBULIN FC AND FRAGMENT B OF PROTEIN A COMPLEX	HUMAN (HOMO SAPIENS) ...	J.DEISENHOFER
2IG2	3.0	IMMUNOGLOBULIN G1	HUMAN (HOMO SAPIENS) ...	M.MARQUART,R.HUBER
1MCW	3.5	IMMUNOGLOBULIN HETEROLOGOUS LIGHT CHAIN DIMER (MCG-WEIR HYBRID)	HUMAN (HOMO SAPIENS)	K.R.ELY,J.N.HERRON
3MCG	2.0	IMMUNOGLOBULIN LAMBDA LIGHT CHAIN DIMER (MCG) (ORTHORHOMBIC FORM)	HUMAN (HOMO SAPIENS)	K.R.ELY,J.N.HERRON
2MCG	2.0	IMMUNOGLOBULIN LAMBDA LIGHT CHAIN DIMER (MCG) (TRIGONAL FORM)	HUMAN (HOMO SAPIENS)	K.R.ELY,J.N.HERRON
2MCP	3.1	IMMUNOGLOBULIN MCPC603 FAB-PHOSPHOCHOLINE COMPLEX	MOUSE (MUS MUSCULUS)	E.A.PADLAN,G.H.COHEN
1PYP	3.0	INORGANIC PYROPHOSPHATASE (E.C.3.6.1.1)	BAKER,S YEAST ...	E.H.HARUTYUNYAN
4INS	1.5	INSULIN	PIG (SUS SCROFA)	G.G.DODSON
1GF1	N/A	INSULIN-LIKE GROWTH FACTOR I (IGF I) (SOMATOMEDIN)	HUMAN (HOMO SAPIENS)	T.L.BLUNDELL
1GF2	N/A	INSULIN-LIKE GROWTH FACTOR II (IGF II) (SOMATOMEDIN)	HUMAN (HOMO SAPIENS)	T.L.BLUNDELL
2TMV	2.9	INTACT TOBACCO MOSAIC VIRUS (FIBER DIFFRACTION STUDY)	TOBACCO MOSAIC VIRUS ...	G.STUBBS
1IL8	N/A	INTERLEUKIN 8 (IL-8) (NEUTROPHIL ACTIVATION PROTEIN) NAP ...	HUMAN (HOMO SAPIENS) ...	G.M.CLORE,A.M.GRONENBORN
2IL8	N/A	INTERLEUKIN 8 (IL-8) (NEUTROPHIL ACTIVATION PROTEIN) NAP ...	HUMAN (HOMO SAPIENS) ...	G.M.CLORE
1I1B	2.0	INTERLEUKIN-1BETA (IL-1BETA)	HUMAN (HOMO SAPIENS) ...	B.C.FINZEL
2I1B	2.0	INTERLEUKIN-1BETA (IL-1BETA)	HUMAN (HOMO SAPIENS) ...	J.P.PRIESTLE

Id	Å	Molecule	Source	Depositors
4I1B	2.0	INTERLEUKIN-1BETA (IL-1BETA)	HUMAN (HOMO SAPIENS) ...	B.VEERAPANDIAN
1CAR	3.0	IOTA CARRAGEENAN (AN ALTERNATING COPOLYMER OF ...)	RED SEAWEED ...	S.ARNOTT
3ICD	2.5	ISOCITRATE DEHYDROGENASE (E.C.1.1.1.42)	(ESCHERICHIA COLI)	J.H.HURLEY
7ADH	3.2	ISONICOTINIMIDYLATED LIVER ALCOHOL DEHYDROGENASE (E.C.1.1.1.1)	HORSE (EQUUS ...)	B.PLAPP,H.EKLUND
2PKA	2.05	KALLIKREIN A (E.C.3.4.21.8)	PORCINE (SUS SCROFA) ...	W.BODE,Z.CHEN
2KAI	2.5	KALLIKREIN A (E.C.3.4.21.8)BOVINE PANCREATIC TRYPSIN ...	PORCINE (SUS SCROFA) ...	W.BODE,Z.CHEN
1KES	3.0	KERATAN SULFATE (SULFATED POLY(GALACTOSYL-N-ACETYL GLUCOSAMINE))	BOVINE (BOS TAURUS) ...	S.ARNOTT
1ABP	2.4	L-ARABINOSE-BINDING PROTEIN	(ESCHERICHIA COLI)	F.A.QUIOCHO,G.L.GILLILAND
1LLC	3.0	L-LACTATE DEHYDROGENASE (E.C.1.1.1.27) COMPLEX WITH ...	(LACTOBACILLUS CASEI)	M.BUEHNER,H.J.HECHT
2LDB	3.0	L-LACTATE DEHYDROGENASE (E.C.1.1.1.27) COMPLEX WITH NAD AND ...	(BACILLUS ...)	K.PIONTEK,M.G.ROSSMANN
1CTF	1.7	L7(SLASH)L12 50 S RIBOSOMAL PROTEIN (C-TERMINAL DOMAIN)	(ESCHERICHIA COLI, ...)	M.LEIJONMARCK,A.LILJAS
3LDH	3.0	LACTATE DEHYDROGENASE (E.C.1.1.1.27) M4 ENZYME, TERNARY ...	DOGFISH (SQUALUS ...)	J.L.WHITE
5LDH	2.7	LACTATE DEHYDROGENASE H/4/ AND S-LAC-NAD+ COMPLEX ...	PIG (SUS SCROFA) HEART	U.M.GRAU,M.G.ROSSMANN
3FAB	2.0	LAMBDA IMMUNOGLOBULIN FAB(PRIME)	HUMAN (HOMO SAPIENS) ...	R.J.POLJAK,L.M.AMZEL
1LRP	3.20	LAMBDA REPRESSOR (N-TERMINAL DOMAIN)	(LAMBDA)	C.PABO,M.LEWIS
1LRD	2.5	LAMBDA REPRESSOR-OPERATOR COMPLEX	BACTERIOPHAGE (LAMBDA)	S.JORDAN,C.PABO
1LH1	2.0	LEGHEMOGLOBIN (ACETATE,MET)	YELLOW LUPIN ...	B.K.VAINSHTEIN
2LH1	2.0	LEGHEMOGLOBIN (ACETATE,MET)	YELLOW LUPIN ...	B.K.VAINSHTEIN
1LH2	2.0	LEGHEMOGLOBIN (AQUO,MET)	YELLOW LUPIN ...	B.K.VAINSHTEIN
2LH2	2.0	LEGHEMOGLOBIN (AQUO,MET)	YELLOW LUPIN ...	B.K.VAINSHTEIN
1LH3	2.0	LEGHEMOGLOBIN (CYANO,MET)	YELLOW LUPIN ...	B.K.VAINSHTEIN
2LH3	2.0	LEGHEMOGLOBIN (CYANO,MET)	YELLOW LUPIN ...	B.K.VAINSHTEIN
1LH4	2.0	LEGHEMOGLOBIN (DEOXY)	YELLOW LUPIN ...	B.K.VAINSHTEIN
2LH4	2.0	LEGHEMOGLOBIN (DEOXY)	YELLOW LUPIN ...	B.K.VAINSHTEIN
1LH5	2.0	LEGHEMOGLOBIN (FLUORO,MET)	YELLOW LUPIN ...	B.K.VAINSHTEIN
2LH5	2.0	LEGHEMOGLOBIN (FLUORO,MET)	YELLOW LUPIN ...	B.K.VAINSHTEIN
1LH6	2.0	LEGHEMOGLOBIN (NICOTINATE,MET)	YELLOW LUPIN ...	B.K.VAINSHTEIN
2LH6	2.0	LEGHEMOGLOBIN (NICOTINATE,MET)	YELLOW LUPIN ...	B.K.VAINSHTEIN
1LH7	2.0	LEGHEMOGLOBIN (NITROSOBENZENE)	YELLOW LUPIN ...	B.K.VAINSHTEIN
2LH7	2.0	LEGHEMOGLOBIN (NITROSOBENZENE)	YELLOW LUPIN ...	B.K.VAINSHTEIN
2LIV	2.4	LEUCINE(SLASH)ISOLEUCINE(SLASH)VALINE-BINDING PROTEIN (LIVBP)	(ESCHERICHIA COLI)	J.S.SACK,M.A.SAPER
2LBP	2.4	LEUCINE-BINDING PROTEIN (LBP)	(ESCHERICHIA COLI) ...	J.S.SACK
2RNT	1.8	LYS 25-RIBONUCLEASE T/1/ (LYS 25-RNASE T/1/) (E.C.3.1.27.3 ...)	(ASPERGILLUS ORYZAE)	W.SAENGER,J.KOEPKE
3RNT	1.8	LYS 25-RIBONUCLEASE T/1/ (LYS 25-RNASE T/1/) (E.C.3.1.27.3 ...)	(ASPERGILLUS ORYZAE)	D.KOSTREWA,H.-W.CHOE
1RSM	2.0	LYS-7-(DINITROPHENYLENE)-LYS-41 CROSS-LINKED RIBONUCLEASE A ...	BOVINE (BOS TAURUS) ...	P.C.WEBER,S.SHERIFF
1LZ2	2.8	LYSOZYME	TURKEY (MELEAGRIS ...)	R.BOTT,R.SARMA
1LYM	2.5	LYSOZYME (E.C.3.2.1.17)	HEN (GALLUS GALLUS) ...	J.HOGLE,S.T.RAO
1LYZ	2.0	LYSOZYME (E.C.3.2.1.17)	HEN (GALLUS GALLUS) ...	R.DIAMOND
1LZ1	1.5	LYSOZYME (E.C.3.2.1.17)	HUMAN (HOMO SAPIENS)	P.J.ARTYMIUK,C.C.F.BLAKE
2LYZ	2.0	LYSOZYME (E.C.3.2.1.17)	HEN (GALLUS GALLUS) ...	R.DIAMOND
2LZ2	2.2	LYSOZYME (E.C.3.2.1.17)	TURKEY (MELEAGRIS ...)	M.R.PARSONS
2LZM	1.7	LYSOZYME (E.C.3.2.1.17)	(ESCHERICHIA COLI ...)	L.H.WEAVER,B.W.MATTHEWS
3LYZ	2.0	LYSOZYME (E.C.3.2.1.17)	HEN (GALLUS GALLUS) ...	R.DIAMOND
3LZM	1.7	LYSOZYME (E.C.3.2.1.17)	(ESCHERICHIA COLI ...)	K.WILSON,R.FABER
4LYZ	2.0	LYSOZYME (E.C.3.2.1.17)	HEN (GALLUS GALLUS) ...	R.DIAMOND
5LYZ	2.0	LYSOZYME (E.C.3.2.1.17)	HEN (GALLUS GALLUS) ...	R.DIAMOND
6LYZ	2.0	LYSOZYME (E.C.3.2.1.17)	HEN (GALLUS GALLUS) ...	R.DIAMOND
2LYM	2.0	LYSOZYME (E.C.3.2.1.17) (1 ATMOSPHERE, 1.4 M NACL)	HEN (GALLUS GALLUS) ...	C.E.KUNDROT,F.M.RICHARDS
3LYM	2.0	LYSOZYME (E.C.3.2.1.17) (1000 ATMOSPHERES, 1.4 M NACL)	HEN (GALLUS GALLUS) ...	C.E.KUNDROT,F.M.RICHARDS
1L01	1.7	LYSOZYME (E.C.3.2.1.17) (DOUBLE MUTANT WITH THR 155 REPLACED ...)	BACTERIOPHAGE T4 ...	S.DAO-PIN,T.ALBER
1L24	1.7	LYSOZYME (E.C.3.2.1.17) (MUTANT WITH ALA 82 REPLACED BY PRO) ...	BACTERIOPHAGE T4 ...	H.NICHOLSON,B.W.MATTHEWS
1L34	1.9	LYSOZYME (E.C.3.2.1.17) (MUTANT WITH ARG 96 REPLACED BY HIS) ...	BACTERIOPHAGE T4 ...	L.H.WEAVER,B.W.MATTHEWS
1L20	1.85	LYSOZYME (E.C.3.2.1.17) (MUTANT WITH ASN 144 REPLACED BY ...)	BACTERIOPHAGE T4 ...	H.NICHOLSON,B.W.MATTHEWS
1L21	1.85	LYSOZYME (E.C.3.2.1.17) (MUTANT WITH ASN 55 REPLACED BY GLY) ...	BACTERIOPHAGE T4 ...	H.NICHOLSON,B.W.MATTHEWS
1L16	1.7	LYSOZYME (E.C.3.2.1.17) (MUTANT WITH GLY 156 REPLACED BY ...)	BACTERIOPHAGE T4 ...	T.M.GRAY,B.W.MATTHEWS

Id	Å	Molecule	Source	Depositors
1L23	1.7	LYSOZYME (E.C.3.2.1.17) (MUTANT WITH GLY 77 REPLACED BY ALA) ...	BACTERIOPHAGE T4 ...	H.NICHOLSON,B.W.MATTHEWS
1L18	1.7	LYSOZYME (E.C.3.2.1.17) (MUTANT WITH ILE 3 REPLACED BY TYR) (I3Y)	BACTERIOPHAGE T4 ...	M.MATSUMURA
1L35	1.8	LYSOZYME (E.C.3.2.1.17) (MUTANT WITH ILE 3 REPLACED BY TYR, ...)	BACTERIOPHAGE T4 ...	P.E.PJURA
1L17	1.7	LYSOZYME (E.C.3.2.1.17) (MUTANT WITH ILE 3 REPLACED BY VAL) (I3V)	BACTERIOPHAGE T4 ...	M.MATSUMURA
1L22	1.7	LYSOZYME (E.C.3.2.1.17) (MUTANT WITH LYS 124 REPLACED BY ...)	BACTERIOPHAGE T4 ...	H.NICHOLSON,B.W.MATTHEWS
1L25	1.8	LYSOZYME (E.C.3.2.1.17) (MUTANT WITH PRO 86 REPLACED BY ALA) ...	BACTERIOPHAGE T4 ...	J.A.BELL,S.DAO-PIN
1L31	1.8	LYSOZYME (E.C.3.2.1.17) (MUTANT WITH PRO 86 REPLACED BY ARG) ...	BACTERIOPHAGE T4 ...	J.A.BELL,S.DAO-PIN
1L27	1.8	LYSOZYME (E.C.3.2.1.17) (MUTANT WITH PRO 86 REPLACED BY ASP) ...	BACTERIOPHAGE T4 ...	J.A.BELL,S.DAO-PIN
1L28	1.9	LYSOZYME (E.C.3.2.1.17) (MUTANT WITH PRO 86 REPLACED BY GLY) ...	BACTERIOPHAGE T4 ...	J.A.BELL,S.DAO-PIN
1L29	1.7	LYSOZYME (E.C.3.2.1.17) (MUTANT WITH PRO 86 REPLACED BY HIS) ...	BACTERIOPHAGE T4 ...	J.A.BELL,S.DAO-PIN
1L30	1.7	LYSOZYME (E.C.3.2.1.17) (MUTANT WITH PRO 86 REPLACED BY LEU) ...	BACTERIOPHAGE T4 ...	J.A.BELL,S.DAO-PIN
1L32	1.7	LYSOZYME (E.C.3.2.1.17) (MUTANT WITH PRO 86 REPLACED BY SER) ...	BACTERIOPHAGE T4 ...	J.A.BELL,S.DAO-PIN
1L19	1.7	LYSOZYME (E.C.3.2.1.17) (MUTANT WITH SER 38 REPLACED BY ASP) ...	BACTERIOPHAGE T4 ...	H.NICHOLSON,B.W.MATTHEWS
1L02	1.7	LYSOZYME (E.C.3.2.1.17) (MUTANT WITH THR 157 REPLACED BY ...)	BACTERIOPHAGE T4 ...	S.DAO-PIN,T.ALBER
1L04	1.7	LYSOZYME (E.C.3.2.1.17) (MUTANT WITH THR 157 REPLACED BY ...)	BACTERIOPHAGE T4 ...	S.DAO-PIN,T.ALBER
1L05	1.7	LYSOZYME (E.C.3.2.1.17) (MUTANT WITH THR 157 REPLACED BY ...)	BACTERIOPHAGE T4 ...	S.DAO-PIN,T.ALBER
1L06	1.7	LYSOZYME (E.C.3.2.1.17) (MUTANT WITH THR 157 REPLACED BY ...)	BACTERIOPHAGE T4 ...	S.DAO-PIN,K.WILSON
1L07	1.7	LYSOZYME (E.C.3.2.1.17) (MUTANT WITH THR 157 REPLACED BY ...)	BACTERIOPHAGE T4 ...	S.DAO-PIN,R.FABER
1L08	1.7	LYSOZYME (E.C.3.2.1.17) (MUTANT WITH THR 157 REPLACED BY ...)	BACTERIOPHAGE T4 ...	S.DAO-PIN,T.ALBER
1L09	1.7	LYSOZYME (E.C.3.2.1.17) (MUTANT WITH THR 157 REPLACED BY ...)	BACTERIOPHAGE T4 ...	S.DAO-PIN,J.BELL
1L10	1.7	LYSOZYME (E.C.3.2.1.17) (MUTANT WITH THR 157 REPLACED BY ...)	BACTERIOPHAGE T4 ...	S.DAO-PIN,K.WILSON
1L12	1.7	LYSOZYME (E.C.3.2.1.17) (MUTANT WITH THR 157 REPLACED BY ...)	BACTERIOPHAGE T4 ...	S.DAO-PIN,T.ALBER
1L13	1.7	LYSOZYME (E.C.3.2.1.17) (MUTANT WITH THR 157 REPLACED BY ...)	BACTERIOPHAGE T4 ...	S.DAO-PIN,T.ALBER
1L14	1.7	LYSOZYME (E.C.3.2.1.17) (MUTANT WITH THR 157 REPLACED BY ...)	BACTERIOPHAGE T4 ...	S.DAO-PIN,T.ALBER
1L15	1.7	LYSOZYME (E.C.3.2.1.17) (MUTANT WITH THR 157 REPLACED BY ...)	BACTERIOPHAGE T4 ...	S.DAO-PIN,T.ALBER
1L33	1.7	LYSOZYME (E.C.3.2.1.17) (MUTANT WITH VAL 131 REPLACED BY ...)	BACTERIOPHAGE T4 ...	H.NICHOLSON,B.W.MATTHEWS
8LYZ	2.5	LYSOZYME (E.C.3.2.1.17) IODINE-INACTIVATED	HEN (GALLUS GALLUS) ...	C.R.BEDDELL
7LYZ	2.5	LYSOZYME (E.C.3.2.1.17) TRICLINIC CRYSTAL FORM	HEN (GALLUS GALLUS) ...	J.MOULT,A.YONATH
1LZT	1.97	LYSOZYME (E.C.3.2.1.17), TRICLINIC CRYSTAL FORM	HEN (GALLUS GALLUS) ...	J.M.HODSDON
2LZT	1.97	LYSOZYME (E.C.3.2.1.17), TRICLINIC CRYSTAL FORM	HEN (GALLUS GALLUS) ...	M.RAMANADHAM
1LZH	6.0	LYSOZYME (MONOCLINIC) (E.C.3.2.1.17)	HEN (GALLUS GALLUS) ...	P.J.ARTYMIUK
9LYZ	2.5	LYSOZYME (NAM-NAG-NAM SUBSTRATE ONLY) (E.C.3.2.1.17)	HEN (GALLUS GALLUS) ...	J.A.KELLY,M.N.G.JAMES
2LZH	6.0	LYSOZYME (ORTHORHOMBIC) (E.C.3.2.1.17)	HEN (GALLUS GALLUS) ...	P.J.ARTYMIUK
6LDH	2.0	M/4/ APO-LACTATE DEHYDROGENASE (E.C.1.1.1.27)	DOGFISH (SQUALUS ...)	C.ABAD-ZAPATERO
8LDH	2.8	M/4/ APO-LACTATE DEHYDROGENASE (E.C.1.1.1.27) COMPLEX WITH ...	DOGFISH (SQUALUS ...)	C.ABAD-ZAPATERO
1LDM	2.1	M/4/ LACTATE DEHYDROGENASE (E.C.1.1.1.27) TERNARY COMPLEX ...	DOGFISH (SQUALUS ...)	J.P.GRIFFITH,M.G.ROSSMANN
2MLT	2.0	MELITTIN	HONEY BEE (APIS ...)	D.EISENBERG
2MEV	3.0	MENGO ENCEPHALOMYOCARDITIS VIRUS COAT PROTEIN	MONKEY BRAIN,MIDDLE ...	M.G.ROSSMANN
1MAD	2.25	METHYLAMINE DEHYDROGENASE (MADH) (E.C.1.4.99.3)	GRAM NEGATIVE ...	F.M.D.VELLIEUX,W.G.J.HOL
7API	3.0	MODIFIED ALPHA/1/-ANTITRYPSIN (MODIFIED ALPHA/1/-PROTEINASE ...)	HUMAN (HOMO SAPIENS)	H.LOEBERMANN
8API	3.1	MODIFIED ALPHA/1/-ANTITRYPSIN (MODIFIED ALPHA/1/-PROTEINASE ...)	HUMAN (HOMO SAPIENS)	H.LOEBERMANN
9API	3.0	MODIFIED ALPHA/1/-ANTITRYPSIN (MODIFIED ALPHA/1/-PROTEINASE ...)	HUMAN (HOMO SAPIENS)	H.LOEBERMANN
1NTP	1.8	MODIFIED BETA TRYPSIN (MONOISOPROPYLPHOSPHORYL INHIBITED) ...	BOVINE (BOS TAURUS) ...	A.A.KOSSIAKOFF
1MON	2.75	MONELLIN	SERENDIPITY ...	S.-H.KIM
1MLE	2.5	MUCONATE LACTONIZING ENZYME (CIS,CIS MUCONATE ...)	(PSEUDOMONAS PUTIDA, ...)	A.GOLDMAN,D.L.OLLIS
1MLI	3.3	MUCONOLACTONE ISOMERASE (E.C.5.3.3.4)	(PSEUDOMONAS PUTIDA)	S.K.KATTI,B.A.KATZ
1MLP	N/A	MUREIN LIPOPROTEIN	(ESCHERICHIA COLI)	A.D.MCLACHLAN
1PMB	2.5	MYOGLOBIN (AQUOMET, PH 7.1)	PORCINE (SUS SCROFA) ...	S.J.SMERDON
2MBA	2.0	MYOGLOBIN (AZIDE COMPLEX) (PH 7.0)	SEA HARE (APLYSIA ...)	M.BOLOGNESI,S.ONESTI
1MB5	1.8	MYOGLOBIN (CARBONMONOXYMYOGLOBIN) (NEUTRON STUDY)	SPERM WHALE ...	J.C.HANSON
5MBN	2.0	MYOGLOBIN (DEOXY)	SPERM WHALE ...	T.TAKANO
1MBD	1.4	MYOGLOBIN (DEOXY, PH 8.4)	SPERM WHALE ...	S.E.V.PHILLIPS
1MBC	1.5	MYOGLOBIN (FE II, CARBONMONOXY, 260 DEGREES K)	SPERM WHALE ...	J.KURIYAN,G.A.PETSKO
1MBN	2.0	MYOGLOBIN (FERRIC IRON - METMYOGLOBIN)	SPERM WHALE ...	H.C.WATSON,J.C.KENDREW
3MBA	2.0	MYOGLOBIN (FLUORIDE COMPLEX) (PH 7.0)	SEA HARE (APLYSIA ...)	M.BOLOGNESI,S.ONESTI

Id	Å	Molecule	Source	Depositors
4MBA	2.0	MYOGLOBIN (IMIDAZOLE COMPLEX) (PH 7.0)	SEA HARE (APLYSIA ...)	M.BOLOGNESI,S.ONESTI
1MBS	2.5	MYOGLOBIN (MET)	COMMON SEAL (PHOCA ...)	H.SCOULOUDI
4MBN	2.0	MYOGLOBIN (MET)	SPERM WHALE ...	T.TAKANO
1MBW	1.9	MYOGLOBIN (MET) (MUTANT WITH INITIATOR MET AND WITH ASP 122 ...)	SYNTHETIC GENE FOR ...	G.N.PHILLIPS JUNIOR
1MBA	1.6	MYOGLOBIN (MET) (PH 7.0)	SEA HARE (APLYSIA ...)	M.BOLOGNESI,S.ONESTI
1MBO	1.6	MYOGLOBIN (OXY, PH 8.4)	SPERM WHALE ...	S.E.V.PHILLIPS
2MHR	1.71.3	MYOHEMERYTHRIN	SIPUNCULAN WORM ...	S.SHERIFF,W.A.HENDRICKSON
3EST	1.65	NATIVE ELASTASE (E.C.3.4.21.11)	PORCINE (SUS SCROFA) ...	E.F.MEYER,G.COLE
1NXB	1.38	NEUROTOXIN B (PROBABLY IDENTICAL TO ERABUTOXIN B)	SEA SNAKE (LATICAUDA ...)	D.TSERNOGLOU,G.A.PETSKO
2CLN	N/A	NZ115 TRIMETHYLCALMODULIN COMPLEX WITH TRIFLUOPERAZINE (MODEL)	BOVINE (BOS TAURUS) BRAIN	N.C.J.STRYNADKA
1OVO	1.9	OVOMUCOID THIRD DOMAIN	JAPANESE QUAIL ...	E.WEBER,E.PAPAMOKOS
2OVO	1.5	OVOMUCOID THIRD DOMAIN	SILVER PHEASANT ...	W.BODE,O.EPP
1HIP	2.0	OXIDIZED HIGH POTENTIAL IRON PROTEIN (HIPIP).	(CHROMATIUM ...)	C.W.CARTER JUNIOR
2PHH	2.7	P-HYDROXYBENZOATE HYDROXYLASE (PHBH) (E.C.1.14.13.2) - ...	(PSEUDOMONAS FLUORESCENS)	J.M.VAN DER LAAN
1PHH	2.3	P-HYDROXYBENZOATE HYDROXYLASE (PHBH) (E.C.1.14.13.2) - FAD - ...	(PSEUDOMONAS FLUORESCENS)	H.A.SCHREUDER,J.DRENTH
1PAD	2.8	PAPAIN (E.C.3.4.22.2) -ACETYL-ALANYL-ALANYL- ...	PAPAYA (CARICA ...)	J.DRENTH,K.H.KALK
6PAD	2.8	PAPAIN (E.C.3.4.22.2) -BENZYLOXYCARBONYL- ...	PAPAYA (CARICA ...)	J.DRENTH,K.H.KALK
5PAD	2.8	PAPAIN (E.C.3.4.22.2) -BENZYLOXYCARBONYL-GLYCYL- ...	PAPAYA (CARICA ...)	J.DRENTH,K.H.KALK
2PAD	2.8	PAPAIN (E.C.3.4.22.2) -CYSTEINYL DERIVATIVE OF CYSTEINE-25 ...	PAPAYA (CARICA ...)	J.DRENTH,K.H.KALK
4PAD	2.8	PAPAIN (E.C.3.4.22.2) -TOSYL-METHYLENYLLYSYL DERIVATIVE OF ...	PAPAYA (CARICA ...)	J.DRENTH
9PAP	1.65	PAPAIN (E.C.3.4.22.2) CYS-25 OXIDIZED	PAPAYA (CARICA ...)	I.G.KAMPHUIS,J.DRENTH
1TN2	3.0	PB(II)-TRANSFER RIBO-NUCLEIC ACID (YEAST,PHE) TRNA (PH 5.0)	YEAST (SACCHAROMYCES ...)	J.C.DEWAN,R.S.BROWN
1TN1	3.0	PB(II)-TRANSFER RIBO-NUCLEIC ACID (YEAST,PHE) TRNA (PH 7.4)	YEAST (SACCHAROMYCES ...)	J.C.DEWAN,R.S.BROWN
2LTN	1.7	PEA LECTIN	GARDEN PEA (PISUM ...)	F.L.SUDDATH
3PEP	2.3	PEPSIN (E.C.3.4.23.1)	PIG (SUS SCROFA)	C.ABAD-ZAPATERO
4PEP	1.8	PEPSIN (E.C.3.4.23.1)	PIG (SUS SCROFA)	N.ANDREEVA
5PEP	2.34	PEPSIN (E.C.3.4.23.1)	PORCINE (SUS SCROFA)	J.B.COOPER,G.KHAN
1PSG	1.65	PEPSINOGEN	PORCINE (SUS SCROFA)	J.A.HARTSUCK
1PFC	3.125	PFC(PRIME) FRAGMENT OF AN IGG1	GUINEA PIG (CAVIA ...)	S.H.BRYANT,L.M.AMZEL
1PHS	3.0	PHASEOLIN	FRENCH BEAN ...	M.C.LAWRENCE
2PFK	2.4	PHOSPHOFRUCTOKINASE (E.C.2.7.1.11)	(ESCHERICHIA COLI)	W.R.RYPNIEWSKI,P.R.EVANS
3PFK	2.4	PHOSPHOFRUCTOKINASE (E.C.2.7.1.11)	(BACILLUS ...)	P.R.EVANS,P.J.HUDSON
5PFK	7.0	PHOSPHOFRUCTOKINASE (E.C.2.7.1.11) (INHIBITED T-STATE ...)	(BACILLUS ...)	P.R.EVANS
1PFK	2.4	PHOSPHOFRUCTOKINASE (E.C.2.7.1.11) (R-STATE) COMPLEX WITH ...	(ESCHERICHIA COLI)	Y.SHIRAKIHARA,P.R.EVANS
4PFK	2.4	PHOSPHOFRUCTOKINASE (E.C.2.7.1.11) COMPLEX WITH ...	(BACILLUS ...)	P.R.EVANS,P.J.HUDSON
3PGK	2.5	PHOSPHOGLYCERATE KINASE (E.C.2.7.2.3) COMPLEX WITH ATP, ...	BAKERS ...	P.J.SHAW,N.P.WALKER
2PGK	3.0	PHOSPHOGLYCERATE KINASE (HORSE,MUSCLE) (E.C.2.7.2.3)	HORSE (EQUUS ...)	R.D.BANKS
3PGM	2.8	PHOSPHOGLYCERATE MUTASE (E.C.2.7.5.3) DE-PHOSPHO ENZYME	DRIED BAKER,S YEAST ...	J.W.CAMPBELL
3BP2	2.1	PHOSPHOLIPASE A/2/ (E.C.3.1.1.4) (PHOSPHATIDE ...)	BOVINE (BOS TAURUS ...)	B.W.DUKSTRA,J.DRENTH
1BP2	1.7	PHOSPHOLIPASE A/2/ (E.C.3.1.1.4) (PHOSPHATIDE ACYL-HYDROLASE)	BOVINE (BOS TAURUS ...)	B.W.DUKSTRA
1P2P	2.6	PHOSPHOLIPASE A/2/ (E.C.3.1.1.4) (PHOSPHATIDE ACYL-HYDROLASE)	PORCINE (SUS SCROFA) ...	B.W.DUKSTRA
3P2P	2.1	PHOSPHOLIPASE A/2/ (PHOSPHATIDE-2-ACYL-HYDROLASE) MUTANT ...	PORCINE (SUS SCROFA) ...	B.W.DUKSTRA
4ICD	2.5	PHOSPHORYLATED ISOCITRATE DEHYDROGENASE (E.C.1.1.1.42)	(ESCHERICHIA COLI)	J.H.HURLEY,A.M.DEAN
1PHY	2.4	PHOTOACTIVE YELLOW PROTEIN	(ECTOTHIORHODOSPIRA ...)	D.E.MCREE,J.A.TAINER
5GCH	2.7	PHOTOLYSIS PRODUCT OF P-DIETHYLAMINO-O-HYDROXY-ALPHA-METHYL ...	BOVINE (BOS TAURUS) ...	B.L.STODDARD,D.RINGE
1PRC	2.3	PHOTOSYNTHETIC REACTION CENTER	(RHODOPSEUDOMONAS ...)	J.DEISENHOFER,O.EPP
7PCY	1.8	PLASTOCYANIN	GREEN ALGA ...	C.A.COLLYER,J.M.GUSS
4PCY	2.15	PLASTOCYANIN (CROSS-LINKED WITH GLUTERALDEHYDE, CU1+, PH 7.8)	POPLAR (POPULUS ...)	J.M.GUSS,H.C.FREEMAN
6PCY	1.90	PLASTOCYANIN (CU1+,PH 3.8)	POPLAR (POPULUS ...)	J.M.GUSS,H.C.FREEMAN
5PCY	1.80	PLASTOCYANIN (CU1+,PH 7.0)	POPLAR (POPULUS ...)	J.M.GUSS,H.C.FREEMAN
1PCY	1.6	PLASTOCYANIN (CU2+, PH 6.0)	POPLAR (POPULUS ...)	J.M.GUSS,H.C.FREEMAN
3PCY	1.9	PLASTOCYANIN (HG2+ SUBSTITUTED)	POPLAR (POPULUS ...)	W.B.CHURCH,J.M.GUSS
2PLV	2.88	POLIOVIRUS (TYPE 1, MAHONEY STRAIN)	HUMAN (HOMO SAPIENS) ...	D.J.FILMAN,J.M.HOGLE
2PAB	1.8	PREALBUMIN (HUMAN PLASMA)	HUMAN (HOMO SAPIENS)	S.J.OATLEY,C.C.F.BLAKE
2BP2	3.0	PROPHOSPHOLIPASE A/2/	BOVINE (BOS TAURUS ...)	B.W.DUKSTRA

Id	Å	Molecule	Source	Depositors
2SGA	1.5	PROTEINASE A (COMPONENT OF THE EXTRACELLULAR FILTRATE ...)	(STREPTOMYCES ...)	M.N.G.JAMES,A.R.SIELECKI
1SGC	1.8	PROTEINASE A COMPLEX WITH CHYMOSTATIN	(STREPTOMYCES ...)	L.T.J.DELBAERE,G.D.BRAYER
3SGB	1.8	PROTEINASE B FROM STREPTOMYCES GRISEUS (SGPB) (E.C. NUMBER ...)	(STREPTOMYCES ...)	R.J.READ,M.FUJINAGA
2PRK	1.5	PROTEINASE K (E.C.3.4.21.14)	FUNGUS (TRITIRACHIUM ...)	C.BETZEL,G.P.PAL
2PAZ	2.0	PSEUDOAZURIN (CUPREDOXIN)	(ALCALIGENES ...)	E.T.ADMAN,K.PETRATOS
1PAZ	1.55	PSEUDOAZURIN (OXIDIZED CU ++ AT PH 6.8)	(ALCALIGENES ...)	K.PETRATOS,Z.DAUTER
1PYK	2.6	PYRUVATE KINASE (E.C.2.7.1.40)	CAT MUSCLE (FELIS ...)	H.MUIRHEAD,M.LEVINE
1F19	2.8	R19.9 (IGG2B/K/, CRI-/A/) FAB FRAGMENT	MOUSE (MUS MUSCULUS) ...	M.B.LASCOMBE
3RP2	1.9	RAT MAST CELL PROTEASE II (RMCPII)	RAT (RATTUS RATTUS) ...	R.REYNOLDS
1RLX	N/A	RELAXIN	PIG (SUS SCROFA) OVARY	N.W.ISAACS,G.DODSON
2RLX	N/A	RELAXIN	PIG (SUS SCROFA) OVARY	N.W.ISAACS,G.DODSON
3RLX	N/A	RELAXIN	PIG (SUS SCROFA) OVARY	N.W.ISAACS,G.DODSON
4RLX	N/A	RELAXIN	PIG (SUS SCROFA) OVARY	N.W.ISAACS,G.DODSON
4RHV	3.0	RHINOVIRUS 14 (HRV14)	HUMAN (HOMO SAPIENS) ...	E.ARNOLD,M.G.ROSSMANN
1RMU	3.0	RHINOVIRUS 14 (HRV14) (MUTANT WITH CYS 1 199 REPLACED BY ...)	HUMAN (HOMO SAPIENS) ...	J.BADGER
2RMU	3.0	RHINOVIRUS 14 (HRV14) (MUTANT WITH VAL 1 188 REPLACED BY ...)	HUMAN (HOMO SAPIENS) ...	J.BADGER
2RR1	3.0	RHINOVIRUS 14 (HRV14) COMPLEX WITH ANTIVIRAL AGENT WIN I(R)	HUMAN (HOMO SAPIENS) ...	J.BADGER,T.J.SMITH
2RS1	3.0	RHINOVIRUS 14 (HRV14) COMPLEX WITH ANTIVIRAL AGENT WIN I(S)	HUMAN (HOMO SAPIENS) ...	J.BADGER,T.J.SMITH
2RM2	3.0	RHINOVIRUS 14 (HRV14) COMPLEX WITH ANTIVIRAL AGENT WIN II(SR)	HUMAN (HOMO SAPIENS) ...	J.BADGER,T.J.SMITH
2RS3	3.0	RHINOVIRUS 14 (HRV14) COMPLEX WITH ANTIVIRAL AGENT WIN III(S)	HUMAN (HOMO SAPIENS) ...	J.BADGER,T.J.SMITH
2R04	3.0	RHINOVIRUS 14 (HRV14) COMPLEX WITH ANTIVIRAL AGENT WIN IV	HUMAN (HOMO SAPIENS) ...	J.BADGER,T.J.SMITH
2RS5	3.0	RHINOVIRUS 14 (HRV14) COMPLEX WITH ANTIVIRAL AGENT WIN V(S)	HUMAN (HOMO SAPIENS) ...	J.BADGER,T.J.SMITH
2R06	3.0	RHINOVIRUS 14 (HRV14) COMPLEX WITH ANTIVIRAL AGENT WIN VI	HUMAN (HOMO SAPIENS) ...	J.BADGER,T.J.SMITH
2R07	3.0	RHINOVIRUS 14 (HRV14) COMPLEX WITH ANTIVIRAL AGENT WIN VII	HUMAN (HOMO SAPIENS) ...	J.BADGER,T.J.SMITH
1R08	3.0	RHINOVIRUS 14 (HRV14) COMPLEX WITH ANTIVIRAL AGENT WIN VIII	HUMAN (HOMO SAPIENS) ...	J.BADGER,T.J.SMITH
1R1A	3.2	RHINOVIRUS SEROTYPE 1 (HRV1) COAT PROTEIN	HUMAN (HOMO SAPIENS) ...	S.KIM,M.G.ROSSMANN
1RHD	2.5	RHODANESE (E.C.2.8.1.1)	BOVINE (BOS TAURUS) LIVER	W.G.J.HOL
1RN3	1.45	RIBONUCLEASE A (E.C.3.1.27.5)	BOVINE (BOS TAURUS) ...	N.BORKAKOTI,D.S.MOSS
5RSA	2.0	RIBONUCLEASE A (E.C.3.1.27.5) (JOINT NEUTRON AND X-RAY)	BOVINE (BOS TAURUS) ...	A.WLODAWER
6RSA	2.0	RIBONUCLEASE A (E.C.3.1.27.5) COMPLEX WITH URIDINE VANADATE ...	BOVINE (BOS TAURUS) ...	A.WLODAWER
7RSA	1.26	RIBONUCLEASE A (PHOSPHATE-FREE) (E.C.3.1.27.5)	BOVINE (BOS TAURUS) ...	A.WLODAWER,G.L.GILLILAND
1RBB	2.5	RIBONUCLEASE B(E.C.3.1.4.22)	BOVINE (BOS TAURUS) ...	R.L.WILLIAMS
1RNT	1.9	RIBONUCLEASE T/1/(E.C.3.1.27.3) ISOZYME-2(PRIME)-GUANYLIC ...	(ASPERGILLUS ORYZAE)	W.SAENGER,R.ARNI
1RNS	2.0	RIBONUCLEASE-S (E.C.3.1.4.22)	BOVINE (BOS TAURUS) ...	F.M.RICHARDS,H.W.WYCKOFF
2RSP	2.0	ROUS SARCOMA VIRUS PROTEASE (RSV PR)	ROUS SARCOMA VIRUS ...	A.WLODAWER,M.MILLER
2RUB	1.7	RUBISCO (RIBULOSE-1,5-BISPHOSPHATE ...)	(RHODOSPIRILLUM RUBRUM)	G.SCHNEIDER
1RDG	1.4	RUBREDOXIN	(DESULFOVIBRIO GIGAS)	M.FREY,L.C.SIEKER,F.PAYAN
3RXN	1.5	RUBREDOXIN	(DESULFOVIBRIO VULGARIS)	E.T.ADMAN,L.C.SIEKER
6RXN	1.5	RUBREDOXIN	(DESULFOVIBRIO ...)	R.E.STENKAMP
5RXN	1.20	RUBREDOXIN (OXIDIZED, FE(III)) (CONSTRAINED MODEL)	(CLOSTRIDIUM ...)	K.D.WATENPAUGH
4RXN	1.20	RUBREDOXIN (OXIDIZED, FE(III)) (UNCONSTRAINED MODEL)	(CLOSTRIDIUM ...)	K.D.WATENPAUGH
1SCP	3.0	SARCOPLASMIC CALCIUM-BINDING PROTEIN	SANDWORM (NEREIS ...)	W.J.COOK,S.E.EALICK
2STV	2.50	SATELLITE TOBACCO NECROSIS VIRUS	COAT PROTEIN OF ...	T.A.JONES,L.LILJAS
1SN3	1.8	SCORPION NEUROTOXIN (VARIANT 3)	SCORPION ...	R.J.ALMASSY
1SRN	1.8	SEMISYNTHETIC RIBONUCLEASE A (RNASE 1-118(COLON)111-124) ...	BOVINE (BOS TAURUS) ...	P.D.MARTIN
4SGB	2.1	SERINE PROTEINASE B COMPLEX WITH THE POTATO INHIBITOR PCI-1	(STREPTOMYCES ...)	M.JAMES,H.GREENBLATT
4SBV	2.8	SOUTHERN BEAN MOSAIC VIRUS COAT PROTEIN	SOUTHERN BEAN MOSAIC ...	M.G.ROSSMANN
1SNC	1.65	STAPHYLOCOCCAL NUCLEASE (E.C.3.1.31.1) COMPLEX WITH A ...	(STAPHYLOCOCCUS ...)	P.J.LOLL,E.E.LATTMAN
2SNS	1.5	STAPHYLOCOCCAL NUCLEASE (E.C.3.1.4.7) COMPLEX WITH ...	(STAPHYLOCOCCUS ...)	M.J.LEGG,F.A.COTTON
2SSI	2.6	STREPTOMYCES SUBTILISIN INHIBITOR	(STREPTOMYCES ...)	Y.MITSUI,Y.SATOW
1SBT	2.5	SUBTILISIN BPN (E.C.3.4.21.14)	PROBABLY BACILLUS ...	R.A.ALDEN
1SIC	2.0	SUBTILISIN BPN(PRIME) (E.C.3.4.21.14) COMPLEX WITH ...	PROBABLY (BACILLUS ...)	Y.MITSUI,S.HIRONO
1S01	1.7	SUBTILISIN BPN(PRIME) 8350 (E.C.3.4.21.14) (MUTANT WITH MET ...)	(BACILLUS ...)	M.WHITLOW,A.J.HOWARD
1CSE	1.2	SUBTILISIN CARLSBERG (E.C.3.4.21.14) (COMMERCIAL PRODUCT ...)	(BACILLUS SUBTILIS) ...	W.BODE
2SEC	1.8	SUBTILISIN CARLSBERG (E.C.3.4.21.14) COMPLEX WITH ...	(BACILLUS SUBTILIS) ...	C.A.MCPHALEN,M.N.G.JAMES

Id	Å	Molecule	Source	Depositors
1SBC	2.5	SUBTILISIN CARLSBERG (SUBTILOPEPTIDASE A) (E.C.3.4.21.14)	(BACILLUS SUBTILIS)	D.J.NEIDHART,G.A.PETSKO
2SBT	2.8	SUBTILISIN NOVO (E.C.3.4.21.14)	PROBABLY BACILLUS ...	I.DRENTH,W.G.J.HOL
2SNI	2.1	SUBTILISIN NOVO (E.C.3.4.21.14) COMPLEX WITH CHYMOTRYPSIN ...	(BACILLUS ...)	C.A.MCPHALEN,M.N.G.JAMES
1L03	1.7	SGAMMA157-BETA-MERCAPTOETHANOL-LYSOZYME (E.C.3.2.1.17) ...	BACTERIOPHAGE T4 ...	S.DAO-PIN,K.WILSON
1L26	1.7	SGAMMA86-BETA-MERCAPTOETHANOL-LYSOZYME (E.C.3.2.1.17) ...	BACTERIOPHAGE T4 ...	J.A.BELL,S.DAO-PIN
1L11	1.7	SGAMMA97-BETA-MERCAPTOETHANOL-LYSOZYME (E.C.3.2.1.17) ...	BACTERIOPHAGE T4 ...	S.DAO-PIN,K.WILSON
1LYD	2.0	T4-LYSOZYME	SYNTHETIC CODING DNA ...	D.R.ROSE
2TAA	3.0	TAKA-AMYLASE A (E.C.3.2.1.1)	(ASPERGILLUS ORYZAE)	M.KUSUNOKI
2AIT	N/A	TENDAMISTAT	(STREPTOMYCES TENDAE)	A.D.KLINE,W.BRAUN
1THI	3.2	THAUMATIN I	KETEMFE ...	S.-H.KIM
1TEC	2.2	THERMITASE (E.C.3.4.21.14) COMPLEX WITH EGLIN-C	(THERMOACTINOMYCES ...)	P.GROS,B.W.DIJKSTRA
3TLN	1.6	THERMOLYSIN (E.C.3.4.24.4)	(BACILLUS ...)	B.W.MATTHEWS,M.A.HOLMES
1TMN	1.9	THERMOLYSIN (E.C.3.4.24.4) COMPLEX WITH ...	(BACILLUS ...)	A.F.MONZINGO,B.W.MATTHEWS
2TMN	1.6	THERMOLYSIN (E.C.3.4.24.4) COMPLEX WITH ...	(BACILLUS ...)	D.E.TRONRUD
5TLN	2.3	THERMOLYSIN (E.C.3.4.24.4) COMPLEX WITH ...	(BACILLUS ...)	B.W.MATTHEWS,M.A.HOLMES
6TMN	1.6	THERMOLYSIN (E.C.3.4.24.4) COMPLEX WITH ...	(BACILLUS ...)	D.E.TRONRUD
5TMN	1.6	THERMOLYSIN (E.C.3.4.24.4) COMPLEX WITH CBZ-GLYP-LEU-LEU ...	(BACILLUS ...)	H.M.HOLDEN
4TMN	1.7	THERMOLYSIN (E.C.3.4.24.4) COMPLEX WITH CBZ-PHEP-LEU-ALA ...	(BACILLUS ...)	H.M.HOLDEN
7TLN	2.3	THERMOLYSIN (E.C.3.4.24.4) COMPLEX WITH CH2CO(N-OH)LEU-OCH3	(BACILLUS ...)	B.W.MATTHEWS
7TMN	N/A	THERMOLYSIN (E.C.3.4.24.4) COMPLEX WITH GLY-TPH-LEU-LEU ...	(BACILLUS ...)	H.M.HOLDEN
4TLN	2.3	THERMOLYSIN (E.C.3.4.24.4) COMPLEX WITH L-LEUCYL-HYDROXYLAMINE	(BACILLUS ...)	B.W.MATTHEWS,M.A.HOLMES
1TLP	2.3	THERMOLYSIN (E.C.3.4.24.4) COMPLEX WITH PHOSPHORAMIDON	(BACILLUS ...)	D.E.TRONRUD
3TMN	1.7	THERMOLYSIN (E.C.3.4.24.4) COMPLEX WITH VAL-TRP (VW)	(BACILLUS ...)	H.M.HOLDEN,B.W.MATTHEWS
1SRX	2.8	THIOREDOXIN (E.C.1.6.4.5) (OXIDIZED FORM)	(ESCHERICHIA COLI B)	B.-O.SODERBERG
2TBV	2.90	TOMATO BUSHY STUNT VIRUS	TOMATO BUSHY STUNT VIRUS	S.C.HARRISON
1TON	1.8	TONIN (E.C. NUMBER NOT ASSIGNED)	RAT (RATTUS RATTUS) ...	M.FUJINAGA,M.N.G.JAMES
1EST	2.5	TOSYL-ELASTASE (E.C.3.4.21.11)	PORCINE (SUS SCROFA) ...	L.SAWYER,D.M.SHOTTON
1TRA	3.0	TRANSFER RIBO-NUCLEIC ACID (YEAST, PHE), TRNA	YEAST (SACCHAROMYCES ...)	
2TRA	3.0	TRANSFER RIBO-NUCLEIC ACID (YEAST,ASP) TRNA (A FORM)	YEAST (SACCHAROMYCES ...)	E.WESTHOF,P.DUMAS,D.MORAS
3TRA	3.0	TRANSFER RIBO-NUCLEIC ACID (YEAST,ASP) TRNA (B FORM)	YEAST (SACCHAROMYCES ...)	E.WESTHOF,P.DUMAS,D.MORAS
4TNA	2.5	TRANSFER RIBO-NUCLEIC ACID (YEAST,PHE) TRNA	YEAST (SACCHAROMYCES ...)	A.JACK,J.E.LADNER,A.KLUG
4TRA	2.7	TRANSFER RIBO-NUCLEIC ACID (YEAST,PHE),TRNA	YEAST (SACCHAROMYCES ...)	E.WESTHOF,P.DUMAS,D.MORAS
6TNA	2.7	TRANSFER RIBO-NUCLEIC ACID (YEAST,PHE),TRNA	YEAST (SACCHAROMYCES ...)	J.L.SUSSMAN
1TGL	1.9	TRIACYLGLYCEROL ACYLHYDROLASE (E.C.3.1.1.3)	(RHIZOMUCOR MIEHEI)	L.BRADY
1TIM	2.5	TRIOSE PHOSPHATE ISOMERASE (E.C.5.3.1.1)	CHICKEN (GALLUS ...)	D.W.BANNER
1YPI	1.9	TRIOSE PHOSPHATE ISOMERASE (TIM) (E.C.5.3.1.1)	YEAST (SACCHAROMYCES ...)	T.ALBER,E.LOLIS
2YPI	2.5	TRIOSE PHOSPHATE ISOMERASE (TIM) (E.C.5.3.1.1) COMPLEX WITH ...	YEAST (SACCHAROMYCES ...)	E.LOLIS,G.A.PETSKO
2TMA	15.0	TROPOMYOSIN	RABBIT (ORYCTOLAGUS ...)	G.N.PHILLIPS JUNIOR
1TNC	N/A	TROPONIN - CALCIUM-BINDING COMPONENT	RABBIT (ORYCTOLAGUS ...)	R.H.KRETSINGER,C.D.BARRY
4TNC	2.0	TROPONIN C	CHICKEN (GALLUS ...)	M.SUNDARALINGAM
5TNC	2.0	TROPONIN-C	TURKEY (MELEAGRIS ...)	O.HERZBERG,M.N.G.JAMES
3WRP	1.8	TRP APOREPRESSOR	(ESCHERICHIA COLI)	R.-G.ZHANG,P.B.SIGLER
2WRP	1.65	TRP REPRESSOR (ORTHORHOMBIC FORM)	(ESCHERICHIA COLI)	C.L.LAWSON,P.B.SIGLER
1WRP	2.2	TRP REPRESSOR (TRIGONAL FORM)	(ESCHERICHIA COLI)	R.W.SCHEWITZ
2PTN	1.55	TRYPSIN (ORTHORHOMBIC, 2.4 M AMMONIUM SULFATE) (E.C.3.4.21.4)	BOVINE (BOS TAURUS) ...	J.WALTER
1SGT	1.7	TRYPSIN (SGT) (E.C.3.4.21.4)	(STREPTOMYCES ...)	R.J.READ,M.N.G.JAMES
3PTN	1.7	TRYPSIN (TRIGONAL, 2.4 M AMMONIUM SULFATE) (E.C.3.4.21.4)	BOVINE (BOS TAURUS) ...	J.WALTER
4PTI	1.5	TRYPSIN INHIBITOR	BOVINE (BOS TAURUS) ...	R.HUBER,D.KUKLA
5PTI	1.0	TRYPSIN INHIBITOR (CRYSTAL FORM II)	BOVINE (BOS TAURUS) ...	A.WLODAWER,R.HUBER
1EFM	2.7	TRYPSIN-MODIFIED ELONGATION FACTOR TU (EF-TU-GDP)	(ESCHERICHIA COLI)	F.JURNAK
1TGN	1.65	TRYPSINOGEN	BOVINE (BOS TAURUS) ...	A.A.KOSSIAKOFF,R.M.STROUD
1TGC	1.8	TRYPSINOGEN (0.50 METHANOL, 0.50 WATER)	BOVINE (BOS TAURUS) ...	J.WALTER
2TGT	1.7	TRYPSINOGEN (103 DEGREES K, 0.70 METHANOL, 0.30 WATER)	BOVINE (BOS TAURUS) ...	J.WALTER
1TGT	1.7	TRYPSINOGEN (173 DEGREES K, 0.70 METHANOL, 0.30 WATER)	BOVINE (BOS TAURUS) ...	J.WALTER
2TGA	1.8	TRYPSINOGEN (2.4 M MAGNESIUM SULFATE)	BOVINE (BOS TAURUS) ...	J.WALTER

Id	Å	Molecule	Source	Depositors
2TPI	2.1	TRYPSINOGEN - PANCREATIC TRYPSIN INHIBITOR - ILE-VAL COMPLEX ...	BOVINE (BOS TAURUS) ...	J.WALTER
2TGP	1.9	TRYPSINOGEN COMPLEX WITH PANCREATIC TRYPSIN INHIBITOR	BOVINE (BOS TAURUS) ...	R.HUBER,W.BODE
3TPI	1.9	TRYPSINOGEN COMPLEX WITH PANCREATIC TRYPSIN INHIBITOR AND ILE-VAL	BOVINE (BOS TAURUS) ...	R.HUBER,W.BODE
1TGS	1.8	TRYPSINOGEN COMPLEX WITH PORCINE PANCREATIC SECRETORY ...	BOVINE (BOS TAURUS) ...	M.BOLOGNESI,G.GATTI
4TPI	2.2	TRYPSINOGEN COMPLEX WITH THE ARG15--ANALOGUE OF PANCREATIC ...	BOVINE (BOS TAURUS) ...	W.BODE,J.WALTER
2TGD	2.1	TRYPSINOGEN, DIISOPROPYLPHOSPHORYL INHIBITED	BOVINE (BOS TAURUS) ...	M.O.JONES,R.M.STROUD
1TGB	1.8	TRYPSINOGEN-CA FROM PEG	BOVINE (BOS TAURUS) ...	W.BODE,H.FEHLHAMMER
1WSY	2.5	TRYPTOPHAN SYNTHASE (E.C.4.2.1.20)	(SALMONELLA ...)	C.HYDE,S.AHMED
1TNF	2.6	TUMOR NECROSIS FACTOR-ALPHA (CACHECTIN)	HUMAN (HOMO SAPIENS) ...	M.J.ECK,S.R.SPRANG
1CLA	2.34	TYPE III CHLORAMPHENICOL ACETYLTRANSFERASE (CAT/III/) ...	(ESCHERICHIA COLI), ...	M.R.GIBBS,A.G.W.LESLIE
3CLA	1.75	TYPE III CHLORAMPHENICOL ACETYLTRANSFERASE (CAT/III/) ...	(ESCHERICHIA COLI), ...	A.G.W.LESLIE
2TS1	2.3	TYROSYL-TRANSFER RNA SYNTHETASE (E.C.6.1.1.1)	(BACILLUS ...)	P.BRICK,T.N.BHAT,D.M.BLOW
3TS1	2.7	TYROSYL-TRANSFER RNA SYNTHETASE (E.C.6.1.1.1) COMPLEXED WITH ...	(BACILLUS ...)	C.MONTEILHET,P.BRICK
1UBQ	1.8	UBIQUITIN	HUMAN (HOMO SAPIENS) ...	S.VIJAY-KUMAR
2UTG	1.64	UTEROGLOBIN	UTERINE SECRETIONS ...	R.BALLY,J.DELETTRE
1UTG	1.34	UTEROGLOBIN (OXIDIZED)	RABBIT (ORYCTOLAGUS ...)	I.MORIZE,E.SURCOUF
7WGA	2.0	WHEAT GERM AGGLUTININ (ISOLECTIN 1)	WHEAT (TRITICUM ...)	C.S.WRIGHT
1WGC	2.2	WHEAT GERM AGGLUTININ (ISOLECTIN 1) COMPLEX WITH ...	WHEAT (TRITICUM ...)	C.S.WRIGHT
9WGA	1.8	WHEAT GERM AGGLUTININ (ISOLECTIN 2)	WHEAT (TRITICUM ...)	C.S.WRIGHT
2WGC	2.2	WHEAT GERM AGGLUTININ (ISOLECTIN 2) COMPLEX WITH ...	WHEAT (TRITICUM ...)	C.S.WRIGHT
2YHX	2.1	YEAST HEXOKINASE B (E.C.2.7.1.1) COMPLEX WITH ...	BAKERS YEAST ...	T.A.STEITZ

Appendix 2: PDB Atom Naming Conventions for Amino Acids

Alanine

— CB

Arginine

— CB — CG — CD — NE — CZ — NH1, NH2

Asparagine

— CB — CG — OD1, ND2

Aspartic acid

— CB — CG — OD1, OD2

Cysteine

— CB — SG

Glutamic acid

— CB — CG — CD — OE1, OE2

Glutamine

— CB — CG — CD — OE1, NE2

Glycine

— X

Histidine

— CB — CG — ND1
CD2 — CE1
NE2

Hydroxyproline

— N — CA —
CD — CB
CG
OD

Isoleucine

— CB — CG2, CG1 — CD1

Leucine

— CB — CG — CD2, CD1

Lysine

— CB — CG — CD — CE — NZ

Methionine

— CB — CG — SD — CE

Proline

— N — CA —
CD — CB
CG

Serine

— CB — OG

Threonine

— CB — OG1, CG2

Valine

— CB — CG1, CG2

Tyrosine

— CB — CG — CD1 — CE1, CD2 — CE2 — CZ — OH

Phenylalanine

— CB — CG — CD1 — CE1, CD2 — CE2 — CZ

Tryptophan

— CB — CG — CD2 — CE3 — CZ3
CD1 — CE2 — CH2
NE1 — CZ2

Appendix 3: Special Characters and Symbols Used in MIDAS

The following table describes symbols which have special meaning to MIDAS. In addition to these symbols, several special characters are available for use in command line editing (user's of the EMACS text editor will readily recognize most of these special editing characters).

Symbol	Function	Usage
#	model number	# *model_number* where *model_number* is an integer.
:	residue	: *residue* where *residue* is a residue name, residue sequence number, or range of residues.
@	atom name	@ *atom_name* where *atom_name* is an atom name or range of atoms.
–	range	specifies a range of atoms such as @CB-* (beta carbon to the last atom in model), a range of residues such as :35-66 (residues 35 through 66) or a range of colors such as red-blue (shades of red, magenta, and blue).
,	name separator	separates names or ranges in an atom specifier *e.g.* #1@CA,CB,CG *or* :21-30,45
*	whole wildcard match	matches whole atom or residue names. For example, #0:*@CA selects the alpha carbon atoms of all residues.
=	partial wildcard match	matches partial atom or residue names. *e.g.* #0:*@C= matches all carbon atoms of all residues.
?	single character wildcard	used for atom and residue *names* only. For example :G?? selects all three letter residue names beginning with "G".
%	every *nth* residue or atom	For example, :%5 selects every fifth residue in the sequence.
z>	zone specifier	**z>***zone* and **zr>***zone* select all residues within *zone* angstroms of the indicated atoms. **za>***zone* selects all atoms (not residues) within *zone* angstroms. Using < instead of > results in the complementary set of atoms.
b>	temperature factor	**b>** *temp_factor* selects all atoms with temperature factors greater than *temp_factor*. **b<** *temp_factor* selects all atoms with temperature factors less than *temp_factor*. For example, **b>**20 **b<**25 selects all atoms with temperature factors greater than 20 and less than 25.

Symbol	Function	Usage
e>	electrostatic potential	e> *potential* selects all atoms with electrostatic potentials greater than *potential*. e< *potential* selects all atoms with electrostatic potentials less than *potential*. For example, e>10 e<20 selects all atoms with electrostatic potentials between 10 and 20 kcal/mole.
+	atom picking	enables use of the mouse to select atoms whose names are substituted for the leftmost appearance of "+" symbol in the MIDAS command line.
;	command separator	separates multiple commands on a single line.
RETURN	return	accept the line.
LINEFEED	linefeed	accept the line.
RUBOUT	backspace	erase the character before the cursor.
CTRL-H	backspace	erase the character before the cursor.
CTRL-U	line kill	erase the whole line.
CTRL-W	word kill	erase the word before the cursor.
CTRL-D	delete	erase the character under the cursor.
CTRL-K		erase to end of line.
CTRL-P	history	retrieve previous command(s).
CTRL-N	history	retrieve next command(s).
CTRL-A		go to beginning of line.
CTRL-E		go to the end of line.
CTRL-B		move back a single character.
CTRL-F		move forward a single character.
CTRL-L		move cursor one word left.
CTRL-R		move cursor one word right.
CTRL-G		insert next character without interpretation.
ESC	break	break after the completion of the current command. (see **source** command.)

Note: Control characters are typed by holding down the CTRL key on the keyboard and typing the corresponding alphabetic character. For example, to type CTRL-H hold down the key marked "CTRL" while at the same time striking the "H" key.

Appendix 4: Default Options, Aliases and Device Assignments

The following is a list of the default options, aliases and device assignments made by MidasPlus at the start of each MidasPlus session. These default assignments are stored in */usr/local/lib/midas/midas.rc*. The user may replace these defaults with an alternate file defined by the UNIX environment variable MIDASRC. If this variable is set to a legal source file name, it is executed *instead of* the MidasPlus default file. This is especially useful for making video tapes, as the user may not want extraneous text appearing on the screen at the beginning of the session.

The default assignments are:

```
assign 0 scaling                                  assign useful functions to the first few sliders
assign 1 section
assign 2 thickness

alias model #                                     a few useful aliases
alias molecule #
alias residue :
alias atom @
alias sidechain @ cb - *
alias mainchain @ n,ca,c,o
alias close ~open
alias conic pdbrun /usr/local/bin/conic           conic and ribbon
alias ribbon pdbrun /usr/local/bin/cartoon            are implemented using pdbrun
set record                                        start remembering commands for later recording
set text                                          show command line and reply area
set labels                                        show distance, angle, and rotation monitors
set control                                       show control panel
set verbose autocolor
set vpsep 42                                      the optimal value for vpsep on your workstation can
                                                      be determined by following the procedure
                                                      outlined in the MidasPlus Installation Guide.
```

Appendix 5: The MidasPlus Delegate Mechanism

What are delegates?

MidasPlus has a rich command language, and users can specify nearly all operations by typing them in on the keyboard. Occasionally, however, it is useful to have other computer programs compose the commands. This functionality is partially provided by the **pdbrun** command, which sends the current transformed coordinates of molecules to an user-specified program, and treats the output of that program as MidasPlus commands. **pdbrun** is most useful for "one-shot" type computations, such as volume rendering or coloring atoms according to packing density. For computations that require information at several different times during a single MidasPlus session, however, **pdbrun** is too inefficient.

The solution is a mechanism that allows designated programs, called **delegates**, to communicate with MidasPlus while executing in parallel. For example, there is a rotation delegate that supports two commands: **snapshot** and **interpolate**. Users can use the **snapshot** command to save molecule positions at selected points during a session, and the **interpolate** command to smoothly interpolate between two saved positions. While it was possible to add the **snapshot** and **interpolate** commands directly to MidasPlus, this approach requires one to have both an understanding of the internal program structure, and permission to change source files. Writing the rotation delegate required no modifications to MidasPlus, and, once the matrix arithmetic was solved, took less that a day to implement. Less than 300 lines of new code (including comments) were written.

The delegate facility provides an alternative to modifying MidasPlus everytime additional functionality is desired. Users who are willing to program can easily implement their own flavors of delegates and need not wait for MidasPlus developers to implement desired new features. In addition, delegates can use information sources other than MidasPlus. Thus, they can, potentially, inject some much needed chemistry information into MidasPlus.

User Perspective

Delegates are controlled from MidasPlus using the *delegate* command, which supports three operations: **start**, **stop**, and **list**. New delegates are invoked with the command

delegate start *delegate_name command arguments...*

where *delegate_name* is the name that MidasPlus will use to refer to the started delegate process, and *command* and *arguments* are the Unix command to execute the delegate program. Once a delegate is running, the user can send commands to it by prefixing the command with the name of the delegate, *i.e.*,

delegate_name delegate_command arguments...

To terminate an active delegate, the user can issue the command

delegate stop *delegate_name*

Finally, the command

delegate list

lists the names of all active delegates.

Implementor Perspective

When a delegate is executed, its standard input (C standard I/O library **stdin** or Unix file descriptor 0) and standard output (**stdout** or descriptor 1) are connected to MidasPlus. Data sent to **stdout** will be interpreted by MidasPlus. Data received on **stdin** are either commands from the user (via the *delegate_name* mechanism), or replies from MidasPlus commands (which are normally displayed to the user).

The communications protocol between the delegate and MidasPlus is very simple. An event diagram of the protocol is shown in Figure 1. On start up, the delegate gets to send lines of commands to MidasPlus for execution. For each line of command, MidasPlus sends back to the delegate all the reply lines, followed by a line containing only the word **SYNC**. When the delegate is finished with its initialization, it informs MidasPlus by sending a line containing only the word **SYNC**. At this point, the delegate should wait for user commands, which will arrive on **stdin** via MidasPlus.

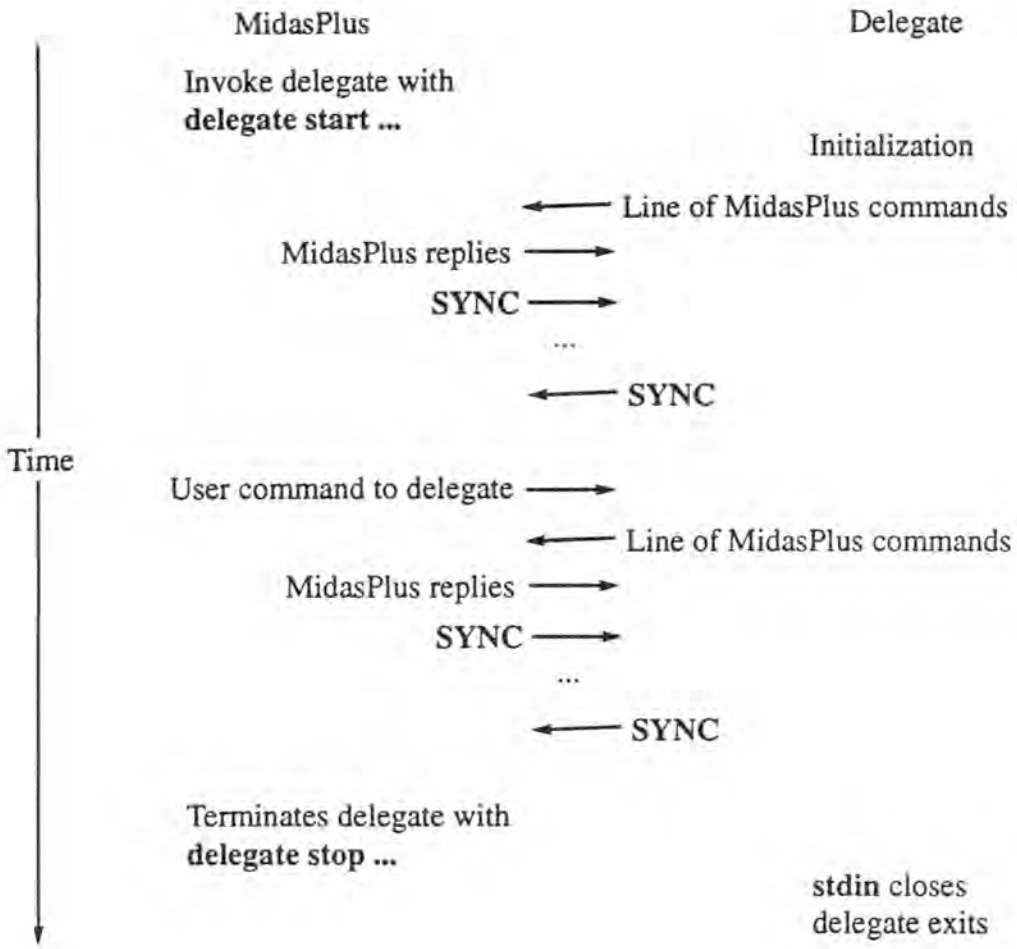

Figure 1 – Event diagram of delegate communications protocol

When the delegate receives an user command, it, once again, gets to send lines of commands to MidasPlus, using exactly the same synchronization as on start up. Between the time it forwards an user command to the delegate and the time it receives the terminating **SYNC** message from the delegate, MidasPlus will not accept any input from the user. Thus, a buggy delegate can cause MidasPlus to appear to hang until the delegate exits. When the delegate receives an end-of-file indication on **stdin**, it should terminate gracefully without trying to communicate with MidasPlus, as **stdout** may already been closed.

While waiting for an user command, a delegate may spontaneously generate MidasPlus commands. This is useful for programs that need to run for a substantial period of time before yielding results and can let the user manipulate the model while the computation occurs. These delegates can send the **SYNC** message before entering

background computation mode, and then spontaneously generate MidasPlus commands once the calculation completes. The only drawback to using background computation is that the user may try to send more commands to the delegate, resulting in a race condition, where the user command may be interpreted as a reply to a MidasPlus command. Implementors of this type of delegates should clearly document which user commands use background computation modes, and possibly warn the user at run-time that further input is not advisable.

An example of a delegate that simply echoes user input follows.

```
#include <stdio.h>
#include <stdarg.h>

#ifndef TRUE
#define TRUE    1
#define FALSE   0
#endif

FILE    *record_fp;

/*
 * Sample MidasPlus delegate
 */
main(int ac, char **av)
{
        register int    c;
        int             verbose = FALSE;
        char            buf[BUFSIZ];
        extern int      optind;
        extern char     *optarg;
        static int      send_command(char *, ...);

        /*
         * Process command line arguments
         */
        while ((c = getopt(ac, av, "vr:")) != EOF)
                switch (c) {
                  case 'v':
                        verbose = TRUE;
                        break;
                  case 'r':
                        record_fp = fopen(optarg, "w");
                        if (record_fp != NULL)
                                setbuf(record_fp, (char *) NULL);
                        break;
                }

        /*
         * First we sync with MidasPlus, which is expecting us to notify
         * it of proper start up.
         */
        (void) printf("SYNC\n");
        (void) fflush(stdout);

        /*
         * We simply echo back any commands to MidasPlus.  If we are
         * in verbose mode, we notify the user before and after
         * each command
         */
        while (fgets(buf, sizeof buf, stdin) != NULL) {
                if (verbose)
                        send_command("echo Delegate executing %s", buf);
                send_command(buf);
                if (verbose)
                        send_command("echo Delegate done\n");
                (void) printf("SYNC\n");
                (void) fflush(stdout);
        }

        exit(0);
}
```

```
/*
 * send_command:
 *      Send a command to MidasPlus and wait until the
 *      synchronizing string comes back
 */
static
int
send_command(char *fmt, ...)
{
        va_list args;
        char    buf[BUFSIZ];

        va_start(args, fmt);
        (void) vfprintf(stdout, fmt, args);
        if (record_fp != NULL) {
                fputs("> ", record_fp);
                vfprintf(record_fp, fmt, args);
        }
        va_end(args);
        (void) fflush(stdout);          /* Make sure buffered data get sent */

        /*
         * Keep reading until we see a line containing only SYNC,
         * which denotes end of MidasPlus replies
         */
        while (fgets(buf, sizeof buf, stdin) != NULL) {
                if (record_fp != NULL)
                        fprintf(record_fp, "< %s", buf);
                if (strcmp(buf, "SYNC\n") == 0)
                        break;
        }

        return;
}
```

Appendix 6: MIDAS Utilities

Several MIDAS utilities are available for generating MIDAS databases, preparing surface files, building templates and preparing models for display. These utilities are typically run directly from the UNIX shell and *not* from the graphics system keyboard during a MIDAS session. The manual pages for MidasPlus and all associated utility programs are included here and the table below summarizes the function of each program.

MIDAS Utility Programs

Program Name	Function
cartoon	generate ribbon representation of proteins
conic	generate CPK-style molecular models with shadows
esp	calculate electrostatic potential
fixatname	correct AMBER pseudo-PDB files so they are in standard PDB format
gentpl	generate a MIDAS template from a Protein Data Bank coordinate file
ilabel	label an IRIS image with arbitrary text
irs	interior atom removal and site selection
makesurf	convert **ms** format surface files to MIDAS surface databases
maketpl	create MIDAS residue templates
midas	MidasPlus molecular interactive display program
midas.dump	print information about MIDAS databases
midas.in	convert Protein Data Bank format to MIDAS format
midas.misc	miscellaneous MIDAS database maintenance utilities
midas.out	convert a MIDAS database to Protein Data Bank format
midas.tty	terminal based version of MidasPlus display program
ms	calculate a solvent accessible molecular surface

Each utility description contains a synopsis line indicating the correct usage of the command. The usage includes the command name in **boldface** type followed by command line parameters. These command line parameters appear in:

boldface print — indicating a flag. The flag is usually a single character and is preceded by a "–" character and is typed as is.

Roman print — indicating a parameter for which the user substitutes the appropriate name, digit, *etc.* In the text of the manual page, these parameters appear in *italic* print.

NAME

cartoon – generate ribbon representation of proteins

SYNOPSIS

cartoon [**–a**] [**–c** *config-file*] [**–f**] [**–h**] [**–o** *output-file*] [**–p**] [**–s** *residue-file*] [**–u** *count*] [**–v** *count*] [**–A** *file*] [**–N**] [**–R**] [**–P** *printer*] [**–X** *cross-section-file*] [**–W**] [*PDB-file*]

DESCRIPTION

Cartoon reads a Protein Data Bank file and generates ribbon image of the molecule. The PDB file may carry extra atom information such as color and radius in the same fashion described in the *conic*(1) manual page (under section "Coloring the Molecule"). The color of the ribbon is the same as the color of the α-carbons. If no PDB file is specified, standard input is used.

Atoms and bonds may also be optionally displayed as balls and sticks. When the ball-and-stick option is selected, most mainchain atoms, including N, C, O, and OXT, are not displayed; the α-carbon, CA, is displayed if it is connected to a displayed atom; all other atoms are displayed. The bonds are derived either from CONECT records if they exist in the PDB file, or from drawing templates found in *midas*(1) template directories.

COMMAND LINE FLAGS

The command line flags interpreted by *cartoon* are:

–a Display all atoms using balls and sticks. The only atoms that will **not** appear in the image are N, C, O, and OXT of amino acids that form the ribbon.

–c *file* Use *file* as the *cartoon* configuration file.

–f Use full screen mode. Set the image size to use the entire screen.

–h Generate a PostScript file instead of displaying the image on the screen. The PostScript image is different from the screen image in that the ribbon for the former case is strictly two dimensional, while the ribbon for the latter case has a non-zero cross-section. The PostScript output will be sent to the default printer queue via the *lpr*(1), unless either the **–o** or **–P** flag is specified.

–o *file* When used with the **–h** flag, the PostScript output is sent to *file* instead of the default printer.

–p Use preview mode. Set the image size to 645x484.

–s *file* The content of *file* is a list of residues which should not be used in the construction of ribbons. The atoms in these residues are still displayed with the **–a** flag. The residues are specified one per line in *file*, and may be specified either by sequence or by type.

–u *count*
The ribbon is formed using a series of bicubic patches between residues. Each patch is subdivided into quadrilaterals, which are rendered on the screen. The **–u** flag specifies the number of divisions to use length-wise along ribbon. The default value is 10.

–v *count*
Specify the number of divisions to use width-wise across the ribbon. The default value is 5.

–A *file* Default atom radius and color information are read from *file*. The format of the file is the same as that used by *conic*(1).

–N Display normal vectors along the ribbon. The resulting image has been compared to porcupines and cacti.

–P*printer*
PostScript output is sent to *printer* instead of the default printer. If both **–o** and **–P** are specified, the output will be sent to the file rather than the printer.

–R Display rectangular patches along ribbon for PostScript output. Mostly used for debugging although this rendition produces a better silhouette.

–W Force *midas*(1) to wait until cartoon has exited before continuing.

–X *file* Use *file* as the cross-section file rather than the default version.

CONFIGURATION FILE

The image computed by *cartoon* is described by a list of options in a configuration file. If the configuration file is absent, or the option is omitted, then a default value will be used. Lines beginning with '#' are comments and are ignored. All other lines are options, which begin with a keyword and are followed by space-separated values. The available options are listed below.

alpha_only
Use only the α-carbons for generating the ribbon. The image generated this way is generally inferior to using both α-carbons and carboxy oxygens.

arrow_size *length width*
Each β-sheet is terminate with an arrow. The length of the arrow is defined as a fraction of the inter-residue distance, and the width is defined as a fraction of the sheet width. The default length and width are 0.5 and 3 respectively.

atinfo *file*
Use the given file as the **atom information** file, which contains information on how each type of atom should be colored. Coloring the molecule is described in the *conic*(1) manual page under section "Coloring the Molecule."

background *r g b* [*r g b*]
Set the background color for the image. If only one RGB value is given, then the entire background is set in that color. If two RGB values are given, then the background is interpolated between the two colors from bottom to top. The default background color is (0 0 0).

bs_size *sphere_size cylinder_size*
Atoms and bonds may be displayed as balls and sticks in the computed image (see the **show_atoms** option below). The radius of the balls (spheres) and the diameter of the sticks (cylinders) are specified as fractions of the atomic radii. The default values are 0.3 for both sizes.

hardcopy
Generate PostScript output instead of displaying image on screen (same as the **–h** command line flag).

helix_color *r g b*
Specify the ribbon color of residues which are part of α-helices. If this option is not specified, the residue color will be the same as the α-carbon in the residue.

location *x y*
Specifies the location of the lower left corner of the display window. Due to a limitation of the IRIS graphics library, this option is only effective when the window size is specified as well (see the **size** option below).

matprop *ambient diffuse specular*
Specifies the material property of ribbons, balls, and sticks. The three values are the ambient, diffuse, and specular reflectance of the material and should all range between 0 and 1. The default values are 0.2, 0.4 and 0.2 respectively.

output *file*
Send PostScript output to *file*. Same as the **–o** command line flag.

precision *u v*
Specify the number of divisions per bicubic patch. The first parameter controls the number of divisions along the ribbon and the second controls the number across. The default values are 10 and 5 respectively. Same as the **–u** and **–v** command line flags.

printer *printer_name*
Send PostScript output to printer *printer_name*. Same as the **–P** command line flag.

scale *scale_factor*

Scale the picture. The degree of scaling is proportional to the scale factor.

sheet_color *r g b*
Specify the ribbon color of residues which are part of β-sheets. If this option is not specified, the residue color will be the same as the α-carbon in the residue.

show_atoms
Display atoms and bonds as balls and sticks. Same as the **–a** command line flag.

show_rectangles
Show rectangles in PostScript output. Same as the **–R** command line flag.

size *width height*
Specify the display window width and height. When used in conjunction with the **location** option, the window and image will appear without user intervention (*i.e.*, having to sweep out a window).

turn_color *r g b*
Specify the ribbon color of residues which are part of turns. If this option is not specified, the residue color will be the same as the α-carbon in the residue.

SEE ALSO

Carson, M and Bugg, C E, **Algorithm for ribbon models of proteins**, *Journal of Molecular Graphics* Vol 4 (1986) pp 121-122.
conic(1), lpr(1), midas(1).

FILES

/usr/local/lib/cartoon/atinfo – default atom information file
/usr/local/lib/cartoon/xsection – default cross-section file
/usr/mol/models/*.ins – default residue drawing templates

BUGS

This program has such a stupid name because all the good ones were taken.

AUTHORS

Conrad Huang
UCSF Computer Graphics Laboratory

NAME

conic – generate CPK-style molecular models with shadows

SYNOPSIS

conic [**–p**] [**–f**] [**–s**] [**–a** *mode*] [**–o** *output-file*] [**–x** *pixels-wide*] [**–y** *pixels-high*] [**–c** *config-file*] [**–v**] [**–W**] [*PDB-file*]

DESCRIPTION

Conic reads a Protein Data Bank file and generates a Corey-Pauling-Koltun style image of the molecule. If no PDB file is specified, standard input is used. There can be an arbitrary number of light sources. Specular highlights, diffuse reflections, and shadows are all computed properly.

COMMAND LINE FLAGS

The command line flags interpreted by *conic* are:

–p Use preview mode. Set the image size to 645x484 and antialias mode to **none** (see below).

–f Set the image size to be the full screen.

–s Invoke *ipaste*(1) on the computed image file. This flag is only meaningful when used in conjunction with the **–o** flag or the **output** option.

–a *mode*
Set the antialias mode. *Mode* is the same as the argument to the **antialias** option in the configuration file (see below).

–o *file* Store the computed image in *file* . The image is not displayed unless the **–s** flag is also specified.

–x *size* Set the horizontal image size to *size* pixels.

–y *size* Set the vertical image size to *size* pixels.

–c *file* Use *file* as the *conic* configuration file.

–v Print progress messages.

–W Force *midas*(1) to wait until cartoon has exited before continuing.

NeXT DIFFERENCES

The **-s** option is not supported because all output is in EPSF format (Encapsulated PostScript File). You should place the output in a file whose name ends with **.eps**, so the Workspace may open it correctly. *Midasplus*(1G) simulates the effects of the **-s** option for compatibility with other systems.

CONFIGURATION FILE

The scene computed by *conic* is described by a list of options in a configuration file. If the configuration file is absent, or the option is omitted, then a default value will be used. Lines beginning with '#' are comments and are ignored. All other lines are options, which begin with a keyword and are followed by space-separated values. The available options are listed below.

ambient *r g b*
Set the ambient light to the given RGB value, which is three floating point intensities ranging from 0 to 1. The default ambient lighting is (0.2 0.2 0.2).

antialias *mode*
Set the antialiasing algorithm. *Mode* may be **none** , for no antialiasing; **3/2** , for mapping 3x3 calculation pixels onto 2x2 image pixels; or **2x2** , for mapping 2x2 calculation pixels onto single image pixels. Antialiasing improves the picture quality at the expense of computation time. The time increase is proportional to the number of pixels computed modulo the startup time. Thus, for small molecules, which have low startup times, going from mode **none** to **2x2** will increase the computation time four-fold. The relative increase is less for large molecules since the startup time for large molecules is a significant fraction of total computation time. The default antialias mode is **none** .

atinfo *file*

Use the given file as the **atom information** file, which contains information on how each type of atom should be colored. Coloring the molecule is described in greater detail below.

background *r g b [r g b]*
Set the background color for the image. If only one RGB value is given, then the entire background is set in that color. If two RGB values are given, then the background is interpolated between the two colors from bottom to top. The default background color is (0 0 0).

cone *x y z r g b dx dy dz angle*
Define a cone light. The absolute Cartesian coordinate of the light source is given by (*x y z*). The color of the light is given by (*r g b*). The Cartesian direction of the cone light is given by (*dx dy dz*). And the half-angle of the cone is *angle* degrees.

eye *r b g*
Conic places an additional point light source which coincides with the eye position. The purpose of this light source is to weakly illuminate shadowed areas so that they have discernible features rather than a uniform color. The **eye** option sets the color of the point light source. The default value is (0.3 0.3 0.3).

fov *angle*
Sets the field-of-view half-angle, in degrees. The default value is 15 degrees.

input *file*
Use *file* as the Protein Data Bank file.

light *x y z r g b*
Add an infinite light source to the scene being computed. The direction of the light source is specified by (*x y z*). The color of the light source is specified by (*r g b*). By default, *conic* defines a light source with direction (1 1 1) and color (1 1 1). The default light source is removed if other sources are specified via the **light** option.

matprop *kd ks power*
Define default material properties. *Kd* is the diffuse reflection coefficient. *Ks* is the specular reflection coefficient. *Power* controls how sharply defined a specular light is, and must be a positive even integer. The higher the value of *power* , the smaller the specular reflection area. The default values are 0.5, 0.25, and 8, respectively.

output *file*
Store the computed image in *file* . The file is generated using the Silicon Graphics Image Library routines, and may be displayed using the *ipaste*(1) program. By default no image file is used and the computed image is displayed on the IRIS directly.

point *x y z r g b*
Define a point light source. The arguments are the same as those for the **light** option, except that (*x y z*) defines the light position rather than direction.

rcone *x y z r g b dx dy dz angle*
Rcone is to **cone** as **rpoint** is to **point.**

rpoint *x y z r g b*
Define a point light source relative to the scene, similar to **point.** The (*x y z*) coordinate is relative to the center of the scene, with lengths normalized such that the distance from the eye to the center of the scene is 1. Thus, the option

rpoint 0 0 1 1 1

would define a point light source that coincided with the eye.

rspot *x y z r g b dx dy dz power*
Rspot is to **spot** as **rpoint** is to **point.**

size *x y* Sets the image size. The default image size is 1280x1024.

spot *x y z r g b dx dy dz power*

Define a spot light. The absolute Cartesian coordinate of the light source is given by (*x y z*). The color of the light is given by (*r g b*). The Cartesian direction of the spot light is given by (*dx dy dz*). The intensity of the spot light drops off as the angle between the spot light direction and the pixel direction; the rate of decrease is the cosine of the angle raised to the *power*th power. *Power* must be an even integer; odd integers will be incremented silently.

COLORING THE MOLECULE

Conic uses two sources of atom radius and coloring information. If neither source of information yields radius and color for an atom, then the atom is ignored.

The first source is embedded in the input to *conic*, which is an extended Protein Data Bank format. The format is identical to standard PDB format except that **ATOM** and **HETATM** records may be followed by **USER** records, whose text field contains a keyword and some values. (The **pdbrun** command of *midas*(1) generates output of this format.) The keywords that *conic* uses are **COLOR**, **RADIUS**, and **MATPROP**. **COLOR** is followed by an integer color index and three floating point RGB intensities. **RADIUS** is followed by a floating point number representing the atom radius in angstroms. If the radius is absent, then the color information is considered invalid. **MATPROP** is followed by the three parameters to the **matprop** option in the configuration file. An example of the extended format follows.

```
ATOM      1  C   HIS     1      49.168  26.701  10.916  1.00 16.00
USER  COLOR 1 0.000 1.000 0.000
USER  RADIUS 1.800
USER  MATPROP 0.5 0.25 16
```

If the input fails to specify the color and radius of an atom, *conic* uses an **atom information** file to supply simple default values. The file contain comment lines, which begin with '#', and information lines, which have five fields: atom type, radius, and RGB value. The atom type is either one or two characters and is used to match the atom type in the PDB input. The atom type '*' is a special case and matches any atom which does not match any other information lines. Using an **atom information** file, simple color-by-type images may be generated from raw PDB files.

The default **atom information** file contains the following lines:

```
C     1.8   0.5   0.5   0.5
N     1.8   0     0     1
O     1.5   1     0     0
S     1.85  1     1     0
H     1.0   1     1     1
P     1.9   1     0.5   0
F     1.35  0     1     0
CL    1.8   0     1     0
BR    1.95  0     1     0
I     2.15  0     1     0
B     1.8   0.5   0     0
FE    0.64  0.5   0     0
CU    1.28  0.5   0     0
ZN    1.38  0.5   0     0
```

BUGS

Light intensity does not attenuate with distance.

SEE ALSO

ipaste(1), midas(1), ilabel(1)

FILES

/usr/local/lib/conic.atinfo – default atom information file

AUTHORS

Eric F. Pettersen, Conrad Huang, Gregory S. Couch
UCSF Computer Graphics Laboratory

NAME

esp – calculate electrostatic potential

SYNOPSIS:

esp -i ms_file **-o** midas_srf [**-q** file] **-a** midas_db [**-r**] [**-n**] [**-c** cutoff] [**-e** epsilon] [**-p** len] [**-v**] [**-w**]

DESCRIPTION:

Esp calculates the electrostatic potential of a solvent accessible surface and stores it in a MIDAS surface database. This information can then be used by MIDAS to selectively color molecular surfaces based on electrostatic charge. *Esp* prints out a summary of the conditions used to calculate the potential.

-i *ms_file* specifies a *ms* surface input file. The electrostatic potential is calculated for all points of this surface. Use the *ms(1)* program with the *-n* flag (to calculate normals) to generate *ms_file*. If the **-p** flag (see below) is given a value of 0, the normals need not be calculated by *ms(1)*.

-o *midas_srf* indicates the MIDAS surface database to which the calculated electrostatic surface is output.

-q *charge_file* The option **-q** *charge_file* supplies an alternate charge file for residue types. The default file used is */usr/local/lib/midas/charges.esp*. Instructions for constructing alternate charge values are contained in the default file. The **-q** flag must *precede* the name of the MIDAS database (**-a** flag) to which the alternate charge file is applied. Thus, a series of command line parameters, **-q** *file1* **-a** *db1* **-q** *file2* **-a** *db2* may be used to associate alternate charge files with specific models.

-a *midas_db* is the name of the MIDAS database containing the coordinates for the associated *ms* file and any other atoms which should be included in the electrostatic calculation. The potential is calculated *only* for those surface points in *ms_file*, but *all* atoms in *midas_db* are used in the calculation. Use *midas.in(1)* to generate a MIDAS database.

-r indicates that the dielectric constant is dependent on the distance from the atom or charge to each surface point.

-n specifies neutral spheres. Charges are summed within the *cutoff* radius defining the sphere, and an equal and opposite charge is spread uniformly across the sphere surface.

-c *cutoff* indicates the cut off radius in angstroms. The default value is 10.0.

-e *epsilon* indicates the value of epsilon. The default value is 1.0.

-p *len* calculates the potential at a distance *len* angstroms from the surface. Positive values of *len* lie outside of the surface, while negative values lie within the surface. The default value of *len* is 1.4 angstroms.

-w indicates that the electrostatic potential of each point should be appended to the corresponding line in the *ms_file*.

-v indicates verbose mode. Auxillary information (e.g. the number of points found for each atom) is reported.

SEE ALSO

ms(1), midas.in(1), midas(1), makesurf(1), MidasPlus User's Manual

AUTHOR

Conrad Huang
UCSF Computer Graphics Laboratory

The idea of neutral spheres came from Paul Weiner while a graduate student in the Department of Pharmaceutical Chemistry, UCSF.

NAME

fixatname – correct AMBER pseudo-PDB files so they are in standard PDB format

SYNOPSIS:

fixatname [file1 ...]

DESCRIPTION:

Versions of AMBER prior to version 3.0 revision A produced PDB files that had atom names aligned in the wrong columns (see MidasPlus User's Manual, Part I, *Protein Data Bank Format* for details of PDB format). *Fixatname* corrects this mis-alignment. AMBER PDB files have their atom names left-adjusted in columns 13-16 of each line. *Fixatname* left-adjusts the atom name in columns 14-16, with the former contents of column 16 being placed in column 13. This results in atom names that conform to the PDB standard with the rather infrequent exception of atom names with a two character atomic symbol (such as iron, *FE*). Such cases can be corrected by hand after processing by *fixatname*.

Fixatname reads from standard input if no file names are specified. *Fixatname* prints the corrected PDB file on standard output.

BUGS

Atoms with two character atomic symbols are mis-aligned, as mentioned above.

SEE ALSO

MidasPlus User's Guide

AUTHOR

Conrad Huang
UCSF Computer Graphics Laboratory

NAME

gentpl – generate a MIDAS template from a Protein Data Bank coordinate file

SYNOPSIS:

gentpl -r residue [**-i** infile] [**-c** radiifile]

DESCRIPTION:

Gentpl is a utility program for generating a MIDAS template from a Protein Data Bank coordinate file. Standard input is assumed unless otherwise specified. Two files are produced: the ASCII instruction file, *residue*.ins, and the binary MIDAS template, *residue*.tpl, where "*residue*" is the residue name specified on the command line. These files are placed in the directory defined by the MODELS variable in the user's program environment (see part I of the MidasPlus User's Manual).

-r *residue* specifies the *residue* name. This name must correspond to the residue name as it appears in the Protein Data Bank input file.

-i *infile* specifies an input file. The input *must* be in standard Brookhaven Protein Data Bank format. The file may contain data other than the coordinate data for the specified residue, but these extraneous records are ignored.

-c *radiifile* specifies a file containing the radii of the atoms used to calculate the connectivity. If no file specified, the program uses as default */usr/local/lib/midas/connect.tpl*. The format of the radii file is a series of records containing the atom name followed by the atom radius in angstroms. At least one space must appear between the atom name and the radius.

-v indicates verbose mode and is useful for tracking down errors.

BUGS

The radii file must be ordered such that in the case of overlapping atoms names, longest names appear before shorter ones. For example, if the file contains the radius for both B and BR, BR must appear before B in file.

SEE ALSO

maketpl(1), midas(1), midas.in(1)
Protein Data Bank File Record Formats, December 1981

AUTHOR

Laurie Jarvis
UCSF Computer Graphics Laboratory

NAME

ilabel – label an IRIS image with arbitrary text

SYNOPSIS

ilabel [**–f**] [**–i** file] [**–o** file] image_file

DESCRIPTION

ilabel displays the given *image_file* and lets the user put labels over the image. The labels are drawn using the IRIS Font Manager, which supports many fonts in arbitrary sizes. Labels may be saved to a file for reuse via the **–o** flag. The stored attributes of labels include color, vertical and horizontal justification, font, size, and position relative to the image. The saved label files may be displayed again using the **–i** flag. There may be several **–i** files but at most one **–o** file. The output file is created when the user selects **Exit** from the pop-up menu. If the file already exists, the user is asked whether the file should be overwritten (unless the **-f** flag was specified, in which case no question is asked).

To add a new label, simply click the left mouse button, which should make a triangular cursor appear, and type in the label. Labels containing multiple lines may be created by typing either RETURN or LINEFEED at the appropriate place. The left button is also used to select other labels so that they may be edited. The middle button is used to select and move labels. The right button displays a menu which contains options to show and hide the defaults panel (see below), redraw the images and labels, turn the mouse cursor on and off, and quit.

The default display attributes for labels are as follows:

font	Times Roman
size	14
color	white
justification	bottom left

All these values may be changed via the defaults panel, which is shown when the user selects the **Show Defaults** option from the right-button menu. When the mouse cursor is over the defaults panel, the left mouse button is used to select the justification mode and label color; the font size may be entered via the keyboard; and new fonts may be selected from the right-button menu. Also, colors may be selected from anywhere on the screen. If the mouse button is depressed over the color selection area, the mouse may be moved anywhere and a color will not be selected until the button is *released.*

BUGS

Cannot display arrows.

AUTHOR

Conrad Huang
UCSF Computer Graphics Laboratory

NAME

irs – interior atom removal and site selection

SYNOPSIS

irs –i midas_db [**–a**] [**–v**] [**–t** midas_tst [**–r** radius]] [**–c** cell_size] [**–o** out_file] [**–m** model]

DESCRIPTION

Irs selects exterior surface atoms of a MIDAS molecular model for display (surface selection), or, alternatively, selects atoms of residues within a given distance of test coordinates (site selection). The input MIDAS database, *midas_db*, is modified such that only those atoms and surface points selected by *irs* are set for display within *MIDAS*. Additionally, *irs* also produces a list of MIDAS commands which may be executed during a MIDAS session to display these same selected atoms and surface points. This feature is useful for re-generating a display that has changed during a MIDAS work session.

Irs performs exterior surface selection by default. In this mode, approximately 60-70 percent of the total atoms in a globular protein are selected. If a file of 'test' coordinates is provided, *irs* runs in site selection mode, selecting those residues that contain atoms within a given test radius of the coordinates in the test file. This provides a useful mechanism for displaying the active site of a model. The *-t* and optional *-r* flags specify the coordinate file and radius, respectively.

Atoms selected for display may be displayed along with their solvent accessible 'molecular' surface. This type of surface display requires calculation of the molecular surface using the *ms(1)* utility and conversion of the output to MIDAS database format using *makesurf(1)*. The resulting molecular surface may then be displayed during a MIDAS session. Van der Waals surfaces can also be selected for display by specifying the *-v* flag.

Irs command line parameters are:

–i *midas_db* specifies the input MIDAS database. This database is modified as described above. If any atoms in the input MIDAS database are not displayed (i.e. they were turned off in a previous MIDAS session), *irs* does not test them unless the **-a** flag is present.

–t *midas_tst* specifies a MIDAS database file of test coordinates used to select a site in the MIDAS model. Only those atoms displayed in the test database are used for site selection. The default test radius is 8.0 angstroms unless otherwise specified by the **-r** flag. Note that all residues are selected, not just those on the surface.

–c *cell_size* *Irs* projects the model onto a 3-D Cartesian grid which is searched for discontinuities in order to identify surface atoms. *Cell_size* is a number defining the distance between grid points. The default size is 1.6 angstroms.

–o *out_file* produces a read file of "surf" or "vdw" MIDAS commands for the atoms selected for display. The model number used in these commands may be changed from the default number 0 by using the **-m** flag. If the **-o** flag is omitted, the MIDAS commands are sent to standard output.

SEE ALSO

midas.in(1), ms(1), makesurf(1), **zone** command in MidasPlus User's Manual

AUTHORS

Paul Bash and Conrad Huang
UCSF Computer Graphics Laboratory

NAME

makesurf – convert *ms* format surface files to MIDAS surface databases

SYNOPSIS:

makesurf -i ms_file **-o** midas_db [**-l**digit] [**-v**]

DESCRIPTION:

Makesurf prepares a MIDAS surface database from a *ms* format surface file. The MIDAS surface database generated consists of four files, *midas_db*.dat, *midas_db*.ndx, *midas_db*.tpl, *midas_db*.srf.

-o *midas_db*	specifies the MIDAS surface database name.
-i *ms_file*	specifies the *ms* format input file.
-l*digit*	specifies the density of points in the surface database. Density ranges from 0 (lowest) to 9 (highest, default).
-v	specifies the verbose option.

SEE ALSO

ms(1), esp(1), MidasPlus User's Manual

AUTHOR

Conrad Huang
UCSF Computer Graphics Laboratory

NAME

maketpl – create MIDAS residue templates

SYNOPSIS

maketpl -o outfile [**-i** infile]

DESCRIPTION

Maketpl creates a MIDAS template file from an ascii connectivity file. The connectivity file may be created either "automatically" via the gentpl(1) program or "by hand" using a text editor. There are five basic record types in a template specification:

RESIDUE residue_name
: The RESIDUE record specifies the residue name which identifies the template. Since the residue name will ultimately be the residue label in the MIDAS display program, residue names should be chosen thoughtfully. This is the first record in the input file.

START atom_name
: The START record indicates the first or "chief" atom of the template and is where the residue would link to the preceding residue in a chain.

DRAW atom_name

MOVE atom_name
: The DRAW and MOVE records specify the connectivity of atoms. The DRAW record indicates a bond between the named atom and the named atom in the preceding record. The MOVE record indicates return to a previously named atom without drawing a bond.

END The END record indicates the end of the template and the "linkage" atom, ie. the atom which bonds to the next residue in a chain.

Maketpl produces two output files named *outfile*.ins and *outfile*.tpl. The ".tpl" file is the MIDAS binary template. The ".ins" file is an ascii file of the same format as the input file except the number of **DRAW** and **MOVE** commands has been minimized to give the most efficient representation of the connectivity.

The MIDAS utilities which access templates expect to find binary template files for residues in the user's models directory or the system directory */usr/mol/models*. The cataloged templates are named *residue*.tpl, and the associated ascii files are named *residue*.ins, where *residue* is the three letter residue name.

FILES

/usr/mol/models

SEE ALSO

gentpl(1), midas.in(1), MidasPlus User's Manual

AUTHOR

Conrad Huang
UCSF Computer Graphics Laboratory

NAME

midas – MidasPlus molecular interactive display program

SYNOPSIS

midas [**-w**] [*file...*]

DESCRIPTION

Midas, or MidasPlus as the second generation program is called, is an interactive molecular modeling system for 3D graphics displays. The system is designed to generate easily manipulated views of large molecules, primarily proteins and nucleic acids. *Midas* normally uses the entire graphics display screen when running, but if the **–w** flag is used, then *midas will start up in a* whose size is interactively indicated by the user with the mouse.

If one or more *files* are specified on the *midas* command line, then *midas* opens then first file as model 0, the second as model 1, etc. (see **open** in the Command Reference Guide section of the MidasPlus User's Manual).

Midas commands consist of a command word followed by 0 or more command arguments. The command arguments typically form a hierarchical specification for an area of interest within a molecule. The following symbols are part of the specification hierarchy:

#n or *#name*	indicates a particular model; *n* is a digit indicating the model number (assigned when model was first opened), or, alternatively, the 3 or 4 alphanumeric abbreviation for the molecule (*e.g.*, **cpa** for carboxypeptidase). This part of the command argument is optional; if it is omitted, all models are processed.
:n or *:name*	indicates a particular residue (*e.g.*, **102**) or a particular group of residues (*e.g.*, **tyr**).
@name	indicates a particular atom (*e.g.*, **ca** for alpha-carbon)

Note that space characters between arguments are ignored and may be freely inserted (if desired) for readability. Omitting a portion of an argument is the same as specifying "*" (matches anything). In addition to these symbols, the following symbols may be used to modify command arguments:

%n	specifies a skip count (*n* is a digit)
–	used to specify an inclusive range of items
*	matches anything
?	matches a single alpha-numeric character
,	used for grouping
~	logical complement of command
;	separator for multiple commands on the same line
<RETURN>	separator for commands

EXAMPLES

label #2:248a@*	label all atoms in residue **248a** of model **2**
label #2:248a	same as above
dist 1 #1:248a@oh#2:1b@n	establish a distance calculation between atom **oh** of residue **248a** in model **1** and atom **n** of residue **1b** of model **2**
rot 0 #1:248a@ca,cb	establish a bond rotation about the **ca->cb** bond of residue **248a** in model **1**
~rot 0	remove the above rotation
~dist 1	disable the above distance calculation

SEE ALSO

MidasPlus User's Manual

AUTHORS

Conrad Huang and Thomas Ferrin, UCSF Computer Graphics Laboratory
Paul Bash contributed vdw style surfaces and the **addaa**, **addgrp**, **swapaa** and **swapna** commands.

NAME

midas.dump – print information about MIDAS databases

SYNOPSIS

midas.dump [**-h**] [**-s**] [**-m**] files...

DESCRIPTION

Midas.dump provides a variety of information about a MIDAS database. The flags indicate the type of information requested:

-h prints out MIDAS file header information for each MIDAS database file, file.*dat*, file.*tpl*, file.*ndx*. For each file, the filetype, residue count, template count, and file size are reported. In addition, for MIDAS index files (.*ndx*) the sequence numbers of the first and last residues and the atom size are reported.

-s prints status information for one or more MIDAS databases. The following information is provided:

Total number of residues in the database
Total atoms present (ie. those which exist)
Total atoms missing (ie. those which do not exist)
Number of residues which contain one or more hydrogen atoms
Total hydrogens present
Total hydrogens missing
Atoms with temperature factors (ie. non-zero)
Total number disulfide bonds

If more than one MIDAS database name is provided, a grand total summary table is included at the end.

-m prints only grand total status information for the specified MIDAS databases.

default If no flags are specified, *midas.dump* goes into interactive mode which allows the user to select specific residues for reporting. Residues are selected by residue sequence number, residue type, or relative position (ie. + or -). For each atom in a selected residue, the atom name, x, y, z coordinates, temperature factor, status bits and color bits are reported, if present. Disulfide type bonds may be reported by requesting residue type "BOND".

Midas.dump accepts either explicit MIDAS database file names (i.e. those with *.tpl*, *.dat*, or *.ndx* suffixes) or implicit MIDAS database names (i.e. those without the *.tpl*, *.dat*, *.ndx*, suffixes). Thus, giving the command *midas.dump* * in a directory which contains several MIDAS databases produces the appropriate output.

SEE ALSO

midas.in(1)

AUTHOR

Laurie Jarvis
UCSF Computer Graphics Laboratory

MIDAS.IN (1)

NAME

midas.in – convert Protein Data Bank format to MIDAS format

SYNOPSIS:

midas.in [**-i** in_file] [**-o** midas_db] [**-v**] [**-c**] [**-C**] [**-a** alternates] [**-m** map_file]

DESCRIPTION:

Midas.in is a utility program for converting protein databank (PDB) format files to MIDAS database format. Standard input is assumed unless otherwise specified. (See the **-i** option below.) Three files are produced, *file*.ndx, *file*.data and *file*.tpl, where *file* is the corresponding PDB file name found on the HEADER record or the *midas_db* name provided with the **-o** flag or a default name provided by the program. The output file name is converted to lower case.

Midas.in requires each residue in the input file to have a corresponding connectivity template. These template files can be stored in two places: a systemwide directory, */usr/mol/models*, of all common templates (*i.e.* amino acids and nucleic acid bases) or, alternatively, in a user's individual template directory. The default location of a user's individual templates is a directory named *models* found in the user's home directory. This default can be overridden by specifying a value for the MODELS variable in the user's program environment.

In the event that no template for a given residue exists, *midas.in* will generate a template file in the user's individual template directory. This file will also be created if an existing residue template is found, but does not contain all the atoms necessary for building the model.

Residue templates consist of two files, a *.tpl* file and a *.ins* file. The template files are not removed after use so other programs may use them; but one may safely remove the *.tpl* file since it can be quickly reconstructed from the *.ins* file with *maketpl(1)*. Templates may also be generated directly by using *gentpl(1)* and a PDB file.

Midas.in acts as a filter by default, accepting standard input. While *midas.in* was originally developed to handle PDB tapes as distributed by Brookhaven National Laboratory, users with local coordinate files may find the following options useful.

-i directs input from the named file, *in_file*. If the named file is "-", standard input is assumed.

-o directs output to the named database, *midas_db*. This is useful when the user wishes to override the filename given on the HEADER card or to assign a file name when no HEADER card is included in the input.

-v sets verbose mode. *Midas.in* reports atom names which were changed to match the template atom names, alternate atom sites, and disulfide bonds when this flag is set. The standard database summary is also printed.

-V passes verbose flag to *gentpl* when generating templates.

-a tells which atoms with alternate indicators to put in the generated database.

-c causes all residues to connected except for waters and ca/na ions. Normally, residues with HETATM records are not connected.

-C sets cleanup mode. All local templates which were generated are removed (*i.e.* the *.tpl* and *.ins* files).

-m causes the named file to be used as the atom name mapping file. By default, */usr/mol/midas/map_file* is used. This is useful if the input file contains non-standard atom names within a given residue and you are unwilling to modify the PDB file.

DIAGNOSTICS

Programs and files not in PDB format are skipped.

Midas.in reports unknown residues (residues for which there is no MIDAS template), unknown atoms (atoms which do not occur in the MIDAS template), and atoms with alternate locations.

MIDAS.IN (1)

SEE ALSO

midas.out(1), midas(1), gentpl(1), maketpl(1)
Protein Databank Newsletter

AUTHORS

Laurie Jarvis and Greg Couch
UCSF Computer Graphics Laboratory

BUGS

If a residue has names that are mapped and an anomaly is detected (*e.g.* a deuterium instead of a hydrogen atom), then *gentpl* (which doesn't map atom names) will generate a template that *midas.in* cannot use. Try rerunning *midas.in* with *-m /dev/null*.

NAME

midas.misc – miscellaneous MIDAS database maintenance utilities

SYNOPSIS

midas.bond dbname
midas.link -i midas_db **-r** residue [**-u**]

DESCRIPTION

Midas.link connects two chains into a single chain or breaks a chain into two chains. Two chains may be linked together at contiguous residues only; i.e. the residues must already be next to one another in the MIDAS database. This is equivalent to deleting a TER record from a Protein Data Bank format file. Note that chief and linkage atoms of the residues are determined by the template from which they were built (see *maketpl(1)*). Required flags are:

-i *midas_db* MIDAS database

-r *residue* residue sequence number of the residue which begins the new chain.

The optional flag **-u** does the *unlinking* operation rather than the linking operation. This is equivalent to inserting a TER record in a Protein Data Bank format file. The residue number specified with the **-r** flag then becomes the first residue of the new chain when the chain is broken.

Midas.bond adds a disulfide type bond between any two atoms in the named MIDAS database. This type of bond is displayed and colored by MIDAS, but no bond rotations are allowed. Such a bond is useful for displaying hydrogen bonds, links between branched saccharides, as well as disulfide bonds. *Midas.bond* prompts the user for the residue sequence number and atom of each bonded pair.

SEE ALSO

midas.in(1), midas.edit(1), maketpl(1)

AUTHOR

Laurie Jarvis
UCSF Computer Graphics Laboratory

NAME

midas.out – convert a MIDAS database to Protein Data Bank format

SYNOPSIS:

midas.out -i midas_db [**-o** filename] [**-m** model] [**-csIP**]

DESCRIPTION:

Midas.out is a utility program for converting MIDAS database format to Protein Data Bank format.

The *midas_db* must be specified and indicates either a MIDAS database or a MIDAS saved session. In the case of a MIDAS database, the database *name* corresponds to the *name*.dat, *name*.ndx and *name*.tpl files. In the case of a saved MIDAS session file, *name* corresponds to the *name* of the *name*.ses file. Note that a session file will have associated with it one or more MIDAS databases. MIDAS applies the rotation matrices stored in the session file to the original coordinates.

Command line options interpreted by *midas.out* are:

-m *model* indicates a particular *model* number in the session. Models other than the one specified are not converted.

-s instructs *midas.out* to convert only those atoms which were displayed at the time the session file was created.

-c causes *midas.out* to output lines in a card image format (fixed length records 80 characters long).

-P annotates the output with PDB USER records which contain MIDAS state information for each atom.

-I changes the output format to internal coordinates. The format is as follows:

atom names(4) residue sequence residue type bond angle dihedral angle

The last three of the 4 named atoms are used to determine the bond angle. The residue sequence and type are that of the *last* atom of the four named.

As an example of *midas.out* usage, consider the saved session "gcn" which consists of the following files: *gcn*.ses, *gcn*.0.dat, *gcn*.0.tpl, *gcn*.0.ndx, *gcn*.1.dat, *gcn*.1.tpl, and *gcn*.1.ndx.

The command *midas.out -i gcn* will produce coordinates with the rotations applied from the saved session, whereas the command *midas.out -i gcn.0* will produce the original coordinates. The command *midas.out -i gcn -m 1 -s* will produce coordinates with the rotations applied for the displayed atoms of model 1 only.

Midas.out directs its output to standard output unless a *filename* is provided with the **-o** flag. A named file of "–" also directs the output to standard output.

Since a MIDAS database does not contain all the information contained in a Brookhaven Protein Data Bank file, the PDB file produced by *midas.out* will only contain ATOM, HETATM, TER, SSBOND, USER, MODEL, and ENDMDL records.

SEE ALSO

midas.in(1)
Protein Databank Newletter

AUTHORS

Laurie Jarvis and Greg Couch
UCSF Computer Graphics Laboratory

NAME

midas.tty – terminal based version of MidasPlus display program

SYNOPSIS

midas.tty [*file...*]

DESCRIPTION

Midas.tty is a version of MidasPlus that runs on any CRT terminal. *Midas.tty* does everything that the *midas* interactive display program does except generate a three dimensional image of the molecular model. In particular, it accepts all normal MidasPlus commands (*e.g.* open, chain, label, distance, copy, save...). This is useful for editing a MIDAS database or setting up session files without requiring access to the graphics display on a workstation.

If either standard input or standard output is redirected (*i.e.* is not a terminal), then *midas.tty* runs in a line-oriented batch mode. Thus it can be called from within a shell script for mass processing of MIDAS databases.

Midas.tty supersedes its cruder predecessor *midas.edit*.

SEE ALSO

midas(1), MidasPlus User's Manual

AUTHOR

Greg Couch, UCSF Computer Graphics Laboratory

BUGS

Since the session file format is device dependent, there must be a different version of *midas.tty* for each different make of workstation.

Interactive distances/angles are not shown in batch mode.

If a terminal emulation program is used to remotely log onto the graphics workstation, then the aspect ratio of the terminal emulation window affects the size and orientation of hardcopy images generated with the **copy** command.

NAME

ms – calculate a solvent accessible molecular surface

SYNOPSIS

ms file [**-a**] [**-d** density] [**-e** file] [**-g** file] [**-i** file] [-s file] [**-k**] [**-p**] [**-m**] [**-n**] [**-r** b-e] [**-w** radius] **-o** file

DESCRIPTION

Ms calculates the molecular surface of a molecule. The molecular surface resembles the van der Waals surface of a molecule, except that crevices between atoms are smoothed over and interstices too small to accommodate the probe are eliminated. The surface includes cavities in the interior of the molecule, even if they are not accessible to a solvent molecule coming from the outside.

The molecular surface calculated is that defined by F. M. Richards (1977, *Ann. Rev. Biophys. Bioeng.*). According to Richards' definition the molecular surface consists of two parts: *contact surface* and *reentrant surface*. The contact surface is made up of "those parts of the molecular van der Waals surface that can actually be in contact with the surface of the probe." The reentrant surface is defined by "the interior-facing part of the probe when it is simultaneously in contact with more than one atom."

File is an input file of coordinates. The input file may be in one of three formats: Kraut (using the **-k** flag), Protein Data Bank (using the **-p** flag), or MIDAS (using the **-m** flag). The first letter or first two letters of the atom name is used to determine the element type. Implicit hydrogens are always included for carbon, nitrogen and oxygen atoms, thus aromatic carbons and nitrogens will have van der Waals radii that are somewhat too big. Note that only amino acid residues will be included unless **-a** is also specified. Because coordinates are multiplied by 100 and stored as integers, coordinates must have absolute values smaller than 327.67.

The flags may be in any order. The meanings of the flags are described below:

-a Include all atoms, not just those in amino acid residues.

-d Change the density of points on the surface. *Density* is a factor affecting the density of points on the surface: the default of 1.0 produces about 5 points per square Angstrom. Only values between 0.1 and 1.0 are permitted. For large proteins, a density of 0.5 is recommended.

-e Calculate only the surface lying within the ellipsoid specified in *file*. *File* consists of 5 lines which define an ellipsoid. The five lines are: the ellipsoid center (line 1), an orthogonal matrix representing the orientation of the ellipsoid (lines 2-4), and the lengths of the three semiaxes (line 5). The normalized vectors of the semiaxes form the columns of the orthogonal matrix. It is recommended that you use a sphere so the matrix will be a unit matrix and all three lengths will just be the sphere radius.

-g Write all the informative messages to *file*, instead of the standard error output. Genuine errors still go to the standard error output. This file is not rewound at any time, so messages from several runs may be accumulated.

-i Calculate the molecular surface only for those residues and atoms specified in *file*, but keeping the rest of the molecule for collision checks. This is equivalent to calculating the molecular surface for the entire molecule and then selecting out the surface belonging to the specified residues and atoms. The file consists of a series of lines such as the following:

```
ASP 205 CA
TYR  13 *
GLY 116 FRM
HIS 178 TO
```

The asterisk means all atoms of the residue and the "FRM" and "TO" mean all residues from 116 to 178 inclusive. These records are read with a FORTRAN "a3,1x,a4,1x,a3" format statement and the residue name, sequence number, and atom name (if specified) must match those of the input file exactly. The residue name must be left justified, and the sequence number must be right justified. The sequence number may contain letters. Up to one hundred such records may be used. The **-e** and **-i** flags are compatible. The surface generated using the **-i** flag is not always the

same as the surface generated by running the entire molecule and afterwards selecting out the desired atoms. The first surface will not include reentrant surface lying between an atom in the **-i** file and atoms not in the file. (The QCPE version of *ms* does not have this bug.)

-s Use the supplied file argument to specify the atomic radii. The format of the file is an atomic symbol followed by its radius, one per line. */usr/local/lib/midas/connect.tpl* uses this format.

-k The input file is in kraut format. Only amino acid residues are read, unless **-a** is specified. If the file name begins with a digit, */usr/mol/kr*, rather than the current directory, is searched.

-m The input file is in MIDAS database format.

-p The input file is in Protein Data Bank format.

-n Include the unit normals to the surface with each surface point record.

-o The output is written to *file*. This flag is not optional.

-r Only residues numbered *b* through *e* inclusive are used in the calculation. This is quite different from the **-i** flag. Residue sequence numbers involving letters may cause problems.

-w Change the water probe radius from the default radius of 1.4 Angstroms. This parameter must be between 1.0 and 2.0.

The output consists of a series of atom and surface point records, with the same format for the first 6 fields. Each atom is followed by the surface points (if any) which belong to it. These first 6 fields are in the following format: residue name, sequence number, atom name, x coordinate, y coordinate, z coordinate. For an atom record, the seventh field is "A". For a surface point record, the seventh field begins with an "S", followed by a "C" or "R" according to whether the point is part of contact or reentrant surface. This is followed a digit used for depicting different density levels. The eighth field is the molecular surface area associated with the point in square Angstroms. If the **-n** flag is specified, the next three fields are the unit normal vector pointing outward from the surface. Informative messages and errors are written to the standard error output unless a **-g** file is specified. The calculation takes about 5 seconds per atom for molecules of fewer than 1000 atoms and 7 seconds per atom for larger molecules (timings are for a VAX780).

The chemical elements that the program can currently handle are those which the author has found to occur in molecules of interest *and* whose van der Waals radii could be located in the literature. The atoms currently recognized are:

Element	Radius
H	1.20
C	1.90
N	1.50
O	1.40
F	1.35
P	1.90
S	1.85
Cl	1.8
Fe	0.64
Cu	1.28
Zn	1.38
Br	1.95
I	2.15

AUTHOR

Michael Connolly
University of California, San Francisco

LIMITATIONS

Six thousand atoms. Also, there cannot be more than 8000 waters involved in overlapping reentrant surface removal. This is generally not a problem, unless you have a large protein with many internal cavities and you have not used the **-d** flag to reduce the density.

FILES

r??????	intermediate file of reentrant surface points - binary
a??????	intermediate file of reentrant surface points - ascii
s??????	intermediate file of reentrant surface points - sorted
c??????	intermediate file of contact surface points - ascii
o??????	intermediate file of all surface points
k??????	intermediate file from getkraut
e??????	intermediate file of atoms inside ellipsoid
b??????	binary file of coordinates for buffering

DIAGNOSTICS

Many and varied. Be sure to examine the **-g** file and "submit.out" before you leave a background job running overnight.