TOPbase (0.6) User's Guide

Walter Cunto and Claudio Mendoza
IBM Venezuela Scientific Center
PO Box 64778, Caracas 1010A, Venezuela
Report CSC-01-92

September 25, 2003

Contents

Preface

The name *Opacity Project* (OP) refers to an international collaboration that was formed in 1984 to calculate the extensive atomic data required to estimate stellar envelope opacities. It has involved research groups from France, Germany, the United Kingdom, the United States and Venezuela. The approach adopted by the OP to calculate opacities is based on a new formalism of the equation of state [1] and on the computation of accurate atomic properties such as energy levels, f-values and photoionization cross sections [2]. In this respect, the OP data accuracy has been shown to be comparable with state-of-the-art theoretical methods and within the level of agreement with experiment attained by such methods. Moreover, the data volume generated in the process and their completeness surpass those of currently used datasets by orders of magnitude. For a general a review of the computation of the atomic data see Ref. 3. Details of the computations for isoelectronic sequences or individual ionic sytems are published in the *Journal of Physics B* in the series of papers "Atomic data for opacity calculations", the reference of which can be obtained for each ion in the present database.

Since such prime atomic data are likely to be used in several research fields, efforts are being made to ensure their availability and effective manipulation for on-line consultation and numerical intensive calculations. TOPbase [4] is a first attempt to provide a tailored interface for efficient and versatile use of the OP database and to satisfy most user data requirements. Its present stage of development is that of a robust prototype undergoing detailed auditing. The end-target is to distribute TOPbase with the OP data as a portable computer package.

During the actual developing of TOPbase we have reached several conclusions which could be of importance for future work in this area. Firstly, there seems to be a definite need in astrophysics for reliable and well presented atomic databases. Secondly, it is feasible to develop specific, efficient, portable, and affordable systems to manipulate large volumes of atomic data for general use. Finally, standard specifications and modes of use must be proposed in order to ensure platform transparency, user application consistency and guidelines for future developments.

References

1. Hummer, D.G. and Mihalas, D., 1988. *Astrophys. J.* **331**, 794

2. Seaton, M.J., 1987. *J. Phys. B* **20**, 6363.

3. Seaton, M.J., Zeippen, C.J., Tully, J.A., Pradhan, A.K., Mendoza, C., Hibbert, A. and Berrington, K.A., 1992. *Rev. Mexicana Astron. Astrofis.* **23**, 19.

4. Cunto, W. and Mendoza, C., 1992. *Rev. Mexicana Astron. Astrofis.* **23**, 107.

1 TOPbase overview

TOPbase is a read-only database system specially designed for general use of the large volume of accurate atomic data generated by the research groups involved in the OP. The tool is an attempt to provide a suitable and uniform framework capable of offering:

- The possibility of intensive use of the OP atomic data
- Access to a complete and accurate dataset
- Transparency for a wide range of computational platforms
- Time and space efficiency
- Low cost.

The present TOPbase version (0.6) has been specifically developed for interactive use of the data. It runs on UNIX[1] based platforms, and it is to handle around 0.5 Gb of compact data with special attention to efficient isonuclear, isoelectronic, ionic, spectroscopic and energy oriented manipulation requirements. The user-oriented query language has been devised so as to provide concise, yet powerful, query capabilities. Graphic processes have also been included in the code to illustrate the versatility of the tool.

The next development stage scheduled for TOPbase will incorporate suggestions and recommendations provided by a community of test users, and a graphic user interface and batch access mode will be considered.

2 TOPbase approach

TOPbase handles data related to two main atomic properties:

- Atomic bound states
- Transitions between given pairs of such bound states.

These two atomic properties condition the TOPbase structure into three entities which are referred to as

- The energy entity, `e`, containing bound states; each bound-state record is uniquely addressed by a *key* defined in terms of the following attributes
  - Chemical element nuclear charge `nz`
  - Ion electron number `ne`
  - Total quantum numbers `islp` of the spectroscopic series containing the state, defined as $\texttt{islp} = 100(2S+1) + 10L + P$ where S is the total spin quantum number, L is the total orbital angular momentum quantum number, and P the parity (0 for even and 1 for odd)

[1] UNIX is a trademark of AT&T

– The state (level) energy position within the spectroscopic series `ilv`.

Records contain for each bound state the term energy and configuration assignment.

Indexing on the energy entity `e` is organized such that a hierarchy of access is enforced. This means that a single element with nuclear charge `nz` gives rise to several ions with `ne` = `1`, `nz` electrons; each ion has several spectroscopic series defined by their total quantum numbers `islp`; and finally each series contains several levels labeled by `ilv` in ascending energy order.

This design consideration was adopted bearing in mind the types of queries most likely to be performed by users. For instance, when the nuclear charge `nz` is kept constant and `ne` is given a range, the query type is referred to as *isonuclear*. Similarly, if the number of electrons is assigned a fixed value and the nuclear charge is given a range, the query is said to be of the *isoelectronic* type. If `nz` and `ne` are given constant values, the query will address data belonging to an ion, and therefore may be called an ionic query. If `nz`, `ne` and `islp` are each given single values, the query type may be said to be spectroscopic. Finally, the user may be interested in states within an energy range, and this query type is then referred to as an *energy* query.

- The f-value entity, `f`, provides data for transitions between pairs of bound states contained in the `e` entity. Each transition record is uniquely addressed by a key defined in terms of
  - Chemical element nuclear charge `nz`
  - Ion electron number `ne`
  - Total quantum numbers `islp` and `jslp` of the initial and final spectroscopic series respectively
  - The initial and final state (level) `ilv` and `jlv` energy positions within their respective spectroscopic series.

  Indexing in the `f` entity imposes a similar access hierarchy to that described for the `e` entity. Moreover, the query types are the same with the difference that the energy query type is replaced by a wavelength query type.

- The photoionization entity `p` may be regarded as an addendum of the energy entity `e`. Both entities share the same accessing scheme; that is, each state key identifies a unique photoionization cross section. Since the cross section is a function of photoelectron energy, the number of points may vary among states. Thus, not only for convenience but also for performance purposes due to the cross-section data volume, these entities are kept physically separated although closely related.

Since access to the data is on a read-only basis, specific compact and efficient data structures have been devised for the implementation of indexes and tables. Also, access-time efficiency is ensured by a physical design tailored to save in the number of data transfers from the bulk of the database which resides in secondary storage. Data loading from secondary storage is indeed the main bottle neck in system performance, due to the volume of data that may be involved. This problem becomes eminent in the smaller platforms, e.g., workstations, with large and fast CPU but relatively poor disk-access times. These issues are detailed in the following subsection.

3 TOPbase general structure

High data compactness and fast access performance are two main features of TOPbase. In order to meet such requirements, physical media such as main and secondary storage are managed in conjunction with the logical handling of the database.

The data are organized into three entities and a set of indexes that are both resident in secondary storage. When TOPbase is invoked, the indexes are loaded into main memory such that the query processor is ready to use them for retrieving the required data from the bulk of the database. Indexes have been carefully structured so as to generally reduce to 1 the number of disk accesses in a search performed in the `e` and `f` entities and to 2 disk accesses in the case of the `p` entity. These indexes are also useful as tables of contents that can be displayed at any time to obtain summarized information of the database.

A search in the database is performed according to user selected criteria that generate a subset of highly cohesive data sharing a common meaning. This cohesive subset of data is loaded into special buffers located in main memory to allow for subsequent fast iterative manipulation in order to satisfy user real needs, specifications and output requirements. The data structure implemented in main memory to store this subset is referred to as a *view*. Each view has associated with it a *descriptor* that registers the selection criteria and view bounds. Note that the basic idea of the view is to perform only one expensive search operation in secondary storage, and from that point on to allow for inexpensive related queries on the highly cohesive data subset. Binary images of views can be archived in and restored from secondary storage. Display facilities on different output devices have been allowed for.

Finally, logical reorganizations of the data stored in a view are possible through the concept of *tables*. A table is a unidimensional array that enables or disables data items within the view according to a selection criterion, inclusion/exclusion facilities, and sorting specifications. Tables may be displayed and plotted. When plotting, graphic processing is performed by means of an interface with a graphic package. Standard graphic software should be chosen in order to preserve code portability. The current version of TOPbase at the CDS interfaces with the PLPLOT[2] graphics package.

A schematic description of the general structure of TOPbase is depicted in Fig. 1.

4 Guided examples

Example 4.1

A common query in atomic physics is to determine quantum defects for states within a spectroscopic series, in order to estimate numerical accuracy and the effect of series perturbers. In order to illustrate the TOPbase capabilities, we have chosen for this example the $2s^2np$ series of the astrophysically important C II ion.

The first operation is to access the energy entity (`e`) and to load into memory, i.e., to create a

[2]PLPLOT has been developed by T.J. Pearson (Caltech), M.J. LeBrun and G. Furnish (Texas) and T. Richardson (Duke)

view of the database, the $^2P^O$ states of C II. This may be attained by issuing the command

```
cv e (nz=6, ne=5, islp=211
```

Note that the `islp` = 211 is determined by the relation `islp` = $100(2S+1)+10L+P$, where $S = 0.5$ is the total spin quantum number, $L = 1$ is the total orbital angular momentum quantum number, and $P = 1$ denotes odd parity.

As a result, 11 states are loaded into the view buffer from disk. Their attributes may be displayed on the screen with the command `dv` (Display View command) or `dt` (Display Table command) since the view has not been logically modified, giving

I	NZ	NE	ISLP	ILV	T	IT	LN	LL	AC	E(RYD)
1	6	5	211	1	T	1	2	1		-1.78769E+00
2	6	5	211	2	T	1	3	1		-5.90082E-01
3	6	5	211	3	T	1	4	1		-3.10093E-01
4	6	5	211	4	C	2	0	0		-2.42561E-01
5	6	5	211	5	T	1	5	1		-1.89475E-01
6	6	5	211	6	T	2	3	0		-1.63384E-01
7	6	5	211	7	T	1	6	1		-1.24104E-01
8	6	5	211	8	T	1	7	1		-9.01943E-02
9	6	5	211	9	T	1	8	1		-6.82796E-02
10	6	5	211	10	T	1	9	1		-5.34454E-02
11	6	5	211	11	T	1	10	1		-4.29586E-02

The configuration assignment for every level is given in terms of the code `T.IT.LN.LL.AC`. Here `T` indicates the level type; a `T` type stands for a state that can be associated to a particular *target* (parent ion) state, i.e., the level belongs to a spectroscopic series that converges to a series limit; a `C` type denotes a state with an equivalent-electron configuration that cannot be associated to a particular series limit. The `IT` index indicates the target state in the case of a `T` state or the configuration in the case of a `C` type; both can be listed with the index command (see section 5.5 for further details)

```
di t (nz NZ ne NE
```

where `NZ` and `NE` are respectively the atomic number and electron number of the ion under consideration. The indexes `LN` and `LL` give, in the case of `T` states, the principal quantum number and orbital angular momentum quantum number of the active electron. They are both set to zero in the case of `C` states. The accuracy of the configuration assignment is indicated by the `AC` parameter: blank for firm assignments and ":" for ambiguous ones.

Therefore, we note that in the previous view there are two states that do not belong to the $2s^2np$ series, namely `ILV=4` and `ILV=6`. The two records can be logically excluded from the view, thus creating a table, by typing the commands

```
ex (ilv=4
ex (ilv=6
```

or, alternatively, in terms of the corresponding rows

```
ex 4
ex 6
```

The command

```
dt nz ne islp ilv e qd iconf > prt
```

is used to print the resulting table plus additional relevant information such as the actual quantum defects and configurations, producing

NZ	NE	ISLP	ILV	E(RYD)	QD	ICONF
6	5	211	1	-1.78769E+00	5.042E-01	2s2 2p
6	5	211	2	-5.90082E-01	3.964E-01	2s2 3p
6	5	211	3	-3.10093E-01	4.084E-01	2s2 4p
6	5	211	5	-1.89475E-01	4.053E-01	2s2 5p
6	5	211	7	-1.24104E-01	3.228E-01	2s2 6p
6	5	211	8	-9.01943E-02	3.405E-01	2s2 7p
6	5	211	9	-6.82796E-02	3.461E-01	2s2 8p
6	5	211	10	-5.34454E-02	3.488E-01	2s2 9p
6	5	211	11	-4.29586E-02	3.505E-01	2s2 10p

Finally, a plot of quantum defects (`qd`) vs energy (`e`) for states contained in the table is displayed with the graphic command

```
pt e qd (fm m nl hd 'QUANTUM DEFECTS FOR C II NP 2PO'
```

which outputs the graph in Fig. 2. To exit TOPbase, one types the command `qq`.

Example 4.2

We want to find the largest absorption gf-values in C, N, and O for transitions with wavelengths in the range 3000 Å to 3200 Å. The data are accessed from the `f` entity and loaded into memory with the command

```
cv f (nz=6,8 e=3000.0,3200.0
```

which creates a view with 441 transitions. The initial table, which overlaps the created view, is sorted in ascending order of gf-values with the command

```
so gf
```

Then, the sorted table is displayed with

```
dt
```

giving the screen

I	NZ	NE	ISLP	JSLP	ILV	JLV	GF	WL(A)
817	8	7	421	410	1	4	-2.69E+00	3.161E+03
894	8	6	311	310	2	3	-2.48E+00	3.055E+03
979	8	6	521	520	1	1	-2.28E+00	3.079E+03
1453	8	6	321	320	3	7	-1.47E+00	3.186E+03
1481	8	4	320	311	4	8	-1.43E+00	3.128E+03
1896	6	2	311	300	4	5	-1.03E+00	3.059E+03
1931	8	4	321	310	3	5	-1.00E+00	3.095E+03
2033	6	2	320	311	3	5	-9.50E-01	3.043E+03
2083	7	3	211	200	4	5	-9.22E-01	3.163E+03
2256	8	4	331	320	3	6	-8.04E-01	3.084E+03
2364	8	6	320	311	5	16	-7.50E-01	3.185E+03
2546	7	3	220	211	3	5	-6.55E-01	3.052E+03
2974	8	4	120	111	5	8	-5.04E-01	3.198E+03
3102	7	4	311	300	5	4	-4.70E-01	3.074E+03
3418	8	4	320	311	6	11	-3.91E-01	3.000E+03
3472	8	4	111	100	6	7	-3.79E-01	3.086E+03

DO YOU WANT TO SEE MORE? ==>

Example 4.3

In this example, we want to plot the photoionization cross sections of the ground states of the first four Mg-like ions. This requirement maybe fulfilled by issuing the command

```
cv p (ne=12 ilv=1
```

which creates a view with the cross sections of the ground states of the 7 Mg-like ions contained in the OP database. The initial table is displayed with command

```
dt
```

which leads to the following display on the screen

I	NZ	NE	ISLP	ILV	E(RYD)	NP
1	12	12	100	1	-5.64727E-01	669
2	13	12	100	1	-1.38689E+00	335
3	14	12	100	1	-2.46589E+00	257
4	16	12	100	1	-5.34104E+00	293
5	18	12	100	1	-9.07903E+00	275
6	20	12	100	1	-1.37687E+01	272
7	26	12	100	1	-3.35246E+01	234

Because we are only interested in the first four, they are selected with the command

```
se 1,4
```

Finally, for comparative purposes, the four cross sections may be plotted on a single graph by typing the command

```
px (ng 4
```

obtaining the graph depicted in Fig. 3.

5 TOPbase commands (version 0.6)

5.1 General command form

Since TOPbase is destined to be used on different platforms and from a large variety of terminals, its on-line user interface has been implemented as a command interpreter that recognizes a simple although powerful query language. There are five types of commands in TOPbase:

- *View* commands
- *Table* commands
- *Graphic* commands
- *Index* commands and
- *Constant* commands.

The general command syntax is:

```
<command_type> <arguments> <selector> <input/output_device>
```

In most cases, `<arguments>` and `<selector>` are sequences of `<attribute>` items, where each attribute is defined by an `<attribute_identifier>` and an `<attribute_range>`. The `<attribute_range>` may be given a single value or a range of values specified in terms of its lower and upper bounds; when an attribute is omitted or, its range is left undefined, the whole attribute domain is assumed. A `<selector>` must be preceded by a right parenthesis "`(`", an `<input_device>` by a left angle "`<`", and the `<output_device>` by a right angle "`>`". Separators such as the equal sign "`=`", commas, and redundant blanks are superfluous in any command.

For example, the command

```
cv e ( nz=6,8 ne=5 islp=201,211 ilv=1,2
```

creates a *view* of the `e` database in main storage with states having the following attribute values:

- Chemical elements with nuclear charge in the range $\texttt{nz} = 6, 8$, i.e., Carbon, Nitrogen and Oxygen
- Ions with `ne=5` electrons, i.e., B-like

- Since the state total quantum numbers are given in terms of the integer variable `islp` $= 100(2S+1)+10L+P$, where S denotes the total spin, L the total orbital angular momentum, and P its parity (0 and 1 denote even and odd parity respectively), the above search takes into account doublet odd states with L ranging from 0 to 1
- Levels in the range `ilv` $= 1, 2$; that is, the lowest two energy levels of each spectroscopic series.

The previous command can be typed without separators as

```
cv e ( nz 6 8 ne 5 islp 201 211 ilv 1 2
```

The subsequent command

```
dv
```

displays on the screen the recently created view:

I	NZ	NE	ISLP	ILV	T	IT	LN	LL	AC	E(RYD)
1	6	5	201	1	T	4	3	1		6.31842E-01
2	6	5	201	2	T	4	4	1		9.43239E-01
3	6	5	211	1	T	1	2	1		-1.78769E+00
4	6	5	211	2	T	1	3	1		-5.90082E-01
5	7	5	201	1	T	4	3	1		3.10916E-01
6	7	5	201	2	T	4	4	1		9.34160E-01
7	7	5	211	1	T	1	2	1		-3.47720E+00
8	7	5	211	2	C	2	0	0		-1.35879E+00
9	8	5	201	1	T	4	3	1		-2.38143E-01
10	8	5	201	2	T	4	4	1		7.96228E-01
11	8	5	211	1	T	1	2	1		-5.67418E+00
12	8	5	211	2	C	2	0	0		-3.02313E+00

Finally, the command

```
dv 2,10 ( islp=211 ilv=1 > prt
```

prints a selection of records from the previously created view. As shown in the next figure, the table will be the result of a selection performed between rows 2 to 10 to include only the lowest state of each $^2P^O$ spectroscopic series.

I	NZ	NE	ISLP	ILV	T	IT	LN	LL	AC	E(RYD)
3	6	5	211	1	T	1	2	1		-1.78769E+00
7	7	5	211	1	T	1	2	1		-3.47720E+00

Commands recognized by the query interpreter are discussed in turn below.

5.2 View Commands

A *view* is a user selected subset of one of the TOPbase entities which is loaded into main memory for subsequent efficient data manipulation. A structure called the *view buffer* is used to store the view in main memory. This view buffer groups unidimensional arrays, one for each attribute, and is headed by a *descriptor* that registers the attribute subdomains.

There are five commands associated with views:

1. *Create* a view
2. *Display* the current view
3. *Archive* the current view
4. *Restore* a view
5. *Display* the current view descriptor.

5.2.1 CV = Create View

When invoked, the command Create View (`cv`)

- Resets the view buffer
- Computes the entry points in the selected database (secondary storage) by means of the index structure resident in main memory
- Loads the matching data into the view buffer
- Updates the descriptor associated with the view.

Clearly this command, or a Restore View command, must be issued prior to commands concerned with view data manipulation, otherwise an empty view buffer will be encountered.

The format of the `cv` command is:

```
cv <entity> <descriptor>
```

The item `<entity>` can be substituted by one of the three following entities:

- The energy levels entity `e`
- The photoionization cross sections entity `p`
- The f-values entity `f`.

While the `e` and `p` entities contain properties that belong to atomic states, the `f` entity is concerned with properties of transitions between states included in `e`. For the sake of clarity, states will be referred to as "i-levels" and to have quantum numbers `islp`; on the other hand, a transition will be assumed to take place from state "i-level" to state "j-level" with quantum numbers `islp` and `jslp` respectively. The `i` and `j` order specified in the query may be interchanged in the TOPbase

data structure, but by convention, a negative gf-value implies an absorption for a transition from i to j.

The `<descriptor>` is a sequence of relational attributes that provide a profile of the required search. As mentioned before, they may be assigned a value or a range of values (upper and lower bounds); they are initialized to 0, and the default is taken to be the complete domain. No special positional order applies to such attributes.

When performing operations on the `e` and `p` entities, the descriptor is specified by the following attributes:

- Nuclear charge `nz` (integer attribute)
- Number of electrons `ne` (integer attribute)
- Total quantum numbers `islp` (integer attribute)
- The item i-level `ilv` (integer attribute)
- Energy in Ryd units `e` (floating point attribute).

Attributes `ilv` and `e` are mutually exclusive.

When the `f` entity is chosen, two additional attributes must be included:

- The total orbital angular momentum change `idl` such that:
  - `idl` = 1 for $\Delta l = -1$
  - `idl` = 2 for $\Delta l = 0$
  - `idl` = 3 for $\Delta l = 1$
- The item j-level `jlv` (integer attribute)

For such cases the i-level is assumed to be `ilv` and the energy attribute, `e`, is the transition wavelength measured in Å units.

Example 5.1

The command

```
cv e ( nz=6,7 ne=4,7 islp=100,110
```

creates a view consisting of singlet even states belonging to ions of Carbon and Nitrogen with 4 to 7 electrons and with total orbital angular momentum in the range $L = 0, 1$. Clearly, the case (`nz` $= 6, $`ne` $= 7$) does not arise, and not all the `islp` series will be present in each ionic system. As a result, the query interpreter issues warning messages, and 57 states with atomic parameters conforming to the descriptor attributes are loaded into main memory as shown below

```
***WARNING: DATABASE DOES NOT INCLUDE ISLP 101 FOR ION 6 4
***WARNING: DATABASE DOES NOT INCLUDE ISLP 100 FOR ION 6 5
***WARNING: DATABASE DOES NOT INCLUDE ISLP 101 FOR ION 6 5
***WARNING: DATABASE DOES NOT INCLUDE ISLP 101 FOR ION 7 4
***WARNING: DATABASE DOES NOT INCLUDE ISLP 100 FOR ION 7 5
***WARNING: DATABASE DOES NOT INCLUDE ISLP 101 FOR ION 7 5
***WARNING: DATABASE DOES NOT INCLUDE ISLP 100 FOR ION 7 7
***WARNING: DATABASE DOES NOT INCLUDE ISLP 101 FOR ION 7 7
DATA FOR 57 STATES HAVE BEEN LOADED
PLEASE TYPE COMMAND ==>
```

Example 5.2

The query

```
cv p ( nz 6 7 ne 4,7 islp 100 110
```

contains the same descriptor as that posed in Example 1, but it acts on the `p` database instead. Again, the query interpreter issues warning messages and loads the photoionization cross sections for the 57 states. Note that separators such as the equal sign "=", commas, and blanks are used arbitrarily.

Example 5.3

The command

```
cv f ( nz 6 ne 6 islp 100 e 1000.0 1500.0
```

loads 9 transitions of neutral C involving ^1S states with transition wavelengths in the range from 1000.0 Å to 1500.0 Å.

5.2.2 DV = Display View

The command Display View (`dv`) is used to display a selected part of the current view on different output devices. The selection may be performed on a range of rows and/or by specifying a set of relational attributes which are collectively referred to as the `<selector>`. The selector takes the same form as the view descriptor, but it does not lead to an update of the descriptor associated with the current view. The format of the `dv` command is

```
dv <row_range> <selector> <output_device>
```

The `<row_range>` is specified by its lower and upper bound, and the default is the complete view. Similarly, the `<selector>` default is the `<row_range>`. No ambiguity arises between the query selector attributes and those of the view descriptor, as the interpreter checks the query syntax and matches it with the descriptor before performing the selection.

One of three types of output media can be specified for <output_device>:

- scr for screen
- prt for printer and
- <file_name> to output into a disk file

The screen is assumed to be the default device. If the output is directed to an existent disk file, the query interpreter warns the user with the message

```
***ERROR: FILE ALREADY EXISTS
```

At this stage, the user is requested to continue

```
DO YOU WANT TO TRY AGAIN?
```

and to provide another file specification if the answer is positive

```
PLEASE, TYPE FILE NAME ===>
```

Example 5.4

Following the cv operation in Example 5.3, the command

```
dv
```

displays on the screen the complete view buffer content as given below

I	NZ	NE	ISLP	JSLP	ILV	JLV	GF	WL(A)
1	6	6	111	100	8	1	4.18E-03	1.500E+03
2	6	6	111	100	9	1	1.31E-02	1.498E+03
3	6	6	111	100	10	1	2.77E-03	1.481E+03
4	6	6	111	100	11	1	7.98E-03	1.480E+03
5	6	6	111	100	12	1	1.91E-03	1.470E+03
6	6	6	111	100	13	1	5.22E-03	1.469E+03
7	6	6	111	100	14	1	1.37E-03	1.462E+03
8	6	6	111	100	15	1	3.61E-03	1.461E+03
9	6	6	111	100	16	1	1.01E-03	1.456E+03

Example 5.5

If the current view is taken to be that generated in Example 5.3, the query

```
dv 2 5 > carbon.fval
```

will create the file carbon.fval in the current directory, containing records in the range I=2,5. A listing of this file is shown below

```
2  6  6  111  100   9  1  1.31E-02  1.498E+03
3  6  6  111  100  10  1  2.77E-03  1.481E+03
4  6  6  111  100  11  1  7.98E-03  1.480E+03
5  6  6  111  100  12  1  1.91E-03  1.470E+03
```

Example 5.6

The command

```
cv e (nz 6
```

loads all the states belonging to C ions. The subsequent query

```
dv (ne 6 islp 310
```

only displays on the screen the $^3P^O$ series of neutral C.

Example 5.7

The command

```
cv p ( nz 13 ne 12 islp 311
```

loads the photoionization cross sections of the $^3P^O$ states of Al II. The subsequent command

```
dv ( ilv 1 4
```

produces the output

I	NZ	NE	ISLP	ILV	E(RYD)	NP
1	13	12	311	1	-1.04647E+00	383
2	13	12	311	2	-4.21441E-01	776
3	13	12	311	3	-2.37944E-01	780
4	13	12	311	4	-1.53025E-01	762

on the screen where the column labeled `NP` tabulates the number of cross section points for each state. Note that the actual cross sections are not listed. On the the other hand, the command

```
dv ( ilv 1 4 > prt
```

will ask the user if the cross sections are to be included in the print file as shown below

```
ARE YOU SURE YOU WANT X-SECTIONS PRINTED? ==>
```

When outputing into a disk file, the actual cross sections are automatically included.

5.2.3 AV = Archive View

This command outputs a binary image of the current view into secondary storage by supplying a UNIX file specification. This command is useful to store views for future intensive use, and it is a convenient compromise between main memory space and access time when different views are to be manipulated in the same session.

The command format is

```
av <output_device>
```

If no explicit disk file is specified for `<output_device>`, the query interpreter will issue the error message

```
***ERROR: INPUT/OUTPUT DEVICE NOT SPECIFIED
```

Also, if the specified file already exists, the query interpreter avoids overwriting the file and issues the warning message

```
***ERROR: FILE ALREADY EXISTS
```

At this stage, the user is requested to continue with the command

```
DO YOU WANT TO TRY AGAIN?
```

and to provide an alternative file specification in case of a positive answer

```
PLEASE, TYPE FILE NAME ===>
```

Note that the output from this command is machine dependent and, consequently, not portable.

Example 5.8

The command

```
av > ca.fval
```

will create in the current directory the file `ca.fval` containing a binary image of the current view.

5.2.4 RV = Restore View

This command loads into main memory a previously archived view which resides in secondary storage. The command syntax is given by

```
rv <input_device>
```

If no explicit disk file is specified with the command, the query interpreter will issue an error message as shown below

```
***ERROR: INPUT/OUTPUT DEVICE NOT SPECIFIED
```

On the other hand, if the specified file does not exist, then the query interpreter will give the following message

```
***ERROR: FILE DOES NOT EXIST  DO YOU WANT TO TRY AGAIN?
```

If the answer is positive, the query interpreter will ask for a correct file name through the message

```
PLEASE TYPE FILE NAME ==>
```

Example 5.9

The command

```
rv < ca.fval
```

restores the archived view of Example 5.8 into main memory assuming that the file `ca.fval` exists.

5.2.5 DD = Display Descriptor

This command allows the user to output the current *view descriptor*. It is useful when a knowledge of the content and bounds of the current view are required. It also provides the precise list of descriptor attributes for view re-creating. For each attribute its subdomain is given, and when an attribute domain is shown as 0, the complete domain of the attribute applies.

The command format is

```
dd <output_device>
```

Three type of output devices may be specified:

- `scr` for screen
- `prt` for printer and
- `<file_name>` for a disk file.

The screen is the default output device. When outputting into an existent disc file, the query interpreter warns the user with the message

```
***ERROR: FILE ALREADY EXISTS
```

At this stage, the user is requested to continue with the command

```
DO YOU WANT TO TRY AGAIN?
```

and to provide an alternative file specification in case of a positive answer

```
PLEASE, TYPE FILE NAME ===>
```

Example 5.10

After performing the following Create View command

```
cv e (nz 26 ne 12 islp 100 321 ilv 1 5
```

each one of the following three commands

```
dd > scr
dd >
dd
```

leads to the same result on the screen as shown below

```
VIEW DESCRIPTOR
===============

DATABASE  = E
    NZ    =  26,  26
    NE    =  12,  12
    ISLP  = 100, 321
    ILV   =   1,   5
    E     =  0.0000E+00,  0.0000E+00
RECORDS LOADED =     50
```

Example 5.11

If the Display Descriptor command

```
dd > prt
```

is issued after the Create View command

```
cv f (ne 5 islp 211 idl 2 3 ilv 1 2
```

the printout would be

```
VIEW DESCRIPTOR
===============

DATABASE  = F
    NZ    =   1,  26
    NE    =   5,   5
    ISLP  = 211, 211
    IDL   =   2,   3
    ILV   =   1,   2
    JLV   =   0,   0
    E     =  0.0000E+00,  0.0000E+00
RECORDS LOADED =   1010
```

5.3 Table commands

A *table* is a logical data structure defined from a view. Although, only one table can be defined at a time, the purpose of this data structure is to allow fast and iterative modification and manipulation of the data currently stored in the view buffer, to avoid delays caused by recurrent secondary storage access, and to structure data display according to user specifications.

When a view is created or restored, the initial table comprises the complete view. At any desired moment, table rows can be selected, excluded, reinstated, sorted—all of them generating new tables—and displayed. Formally, the table commands are:

1. *Display* a table
2. *Select* a table
3. *Exclude* rows from a table
4. *Reinstate* excluded rows of a table and
5. *Sort* a table.

5.3.1 DT = Display Table

This command is employed to display the attributes contained in, or specific functions computed from the attributes of, the active rows of the current view. This means that the command does not display excluded rows (see the `ex` command below), and it can be used to display more information than that displayed by the `dv` command.

The command syntax is

```
dt <row_range> <function_arguments> <selector> <output_device>
```

Here, `<row_range>` can be a specific row, a range defined by two bounds, or simply all the table rows as a default.

The menu of functions is different for the different entities. If `<function_arguments>` are supplied with the command, they are computed from attributes of the table currently active in main memory; any combination and order of functions given in the lists below is allowed.

- For the `e` and `p` entities (i-level)

`i`:	Record number (integer)
`nz`:	Nuclear charge (integer)
`zinv`:	Inverse of the nuclear charge (floating point)
`ne`:	Number of ion electrons (integer)
`islp`:	Total quantum numbers of i-level (integer)
`ilv`:	i-level (integer)
`t`:	Level identification type, i.e., T or C (character)
`it`:	Level identification index (integer)
`ln`:	Principal quantum number of active electron (integer)
`ll`:	Orbital angular momentum of active electron (integer)
`a`:	Identification accuracy (character)
`iconf`:	i-level configuration (character)
`gi`:	Statistical weight of i-level (floating point)
`e`:	Energy in Rydberg units relative to the ionization limit (floating point)
`te`:	Energy in Rydberg units relative to the ground state (floating point)
`qd`:	Quantum defect (floating point)
`eqn`:	Effective quantum number (floating point)
`rl`:	Radiative lifetime in nsec (floating point).

- For the `p` entity only (i-level)

  `np`: Number of points in photoionization cross section (integer).

- For the `f` entity (i-level to j-level)

  `jslp`: Total quantum number of the j-level (integer)
  `jlv`: j-level (integer)
  `jconf`: j-level configuration (character)
  `gj`: Statistical weight of j-level (floating point)
  `wl`: Wavelength of the transition (floating point)
  `gf`: Weighted fvalue (floating point)
  `ga`: Weighted transition probability (floating point).

If a `<selector>` is provided with the command, then further rows in the active table are discriminated. However, the command does not change the content of the active table.

Again, three different output devices may specified: screen, printer, and a disk file. The screen is the default device.

Example 5.12

Let us assume that the $^2P^O$ series of N III is loaded into the view buffer with the command

```
cv e (nz=7 ne=5 islp=211
```

For each one of the 14 states in the resulting table, we want to tabulate

- The nuclear charge `nz`
- The number of electrons `ne`
- The SLP `islp`
- The energy relative to the ground state `te`
- The radiative lifetime `rl`
- The electronic configuration `iconf`.

The following command

```
dt nz ne islp te rl iconf
```

gives the desired result

NZ	NE	ISLP	TE(RYD)	RL(NS)	ICONF
7	5	211	0.00000E+00	1.00E+75	2s2 2p
7	5	211	2.11841E+00	2.82E-01	2p3
7	5	211	2.23232E+00	4.56E+00	2s2 3p
7	5	211	2.70608E+00	4.04E-01	2p(3PO) 3s
7	5	211	2.83230E+00	2.70E+00	2s2 4p
7	5	211	3.07201E+00	2.98E+00	2s2 5p
7	5	211	3.12379E+00	1.26E-01	2p(3PO) 3d
7	5	211	3.19842E+00	8.67E-01	2s2 6p
7	5	211	3.22553E+00	1.76E-01	2p(1PO) 3s
7	5	211	3.27808E+00	6.50E+00	2s2 7p
7	5	211	3.32566E+00	9.34E+00	2s2 8p
7	5	211	3.35763E+00	6.02E+00	2s2 9p
7	5	211	3.37650E+00	1.06E+00	2p(3PO) 4s:
7	5	211	3.38513E+00	1.96E+00	2s2 10p:

Example 5.13

In this example, we want to tabulate the first 10 transitions arising from the ground state of Ne I showing

- The nuclear charge `nz`
- The number of electrons `ne`
- The SLP for the initial and final states `islp` and `jslp` respectively
- The level index for the initial and final states `ilv` and `jlv` respectively
- The gf-values for the transitions `gf`
- The electronic configuration for the initial and final states `iconf` and `jconf` respectively.

The commands to issue are

```
cv f (nz 10 ne 10 islp 100 ilv 1 jlv 1 10
dt nz ne islp jslp ilv jlv gf iconf jconf
```

and the result obtained is

NZ	NE	ISLP	JSLP	ILV	JLV	GF	ICONF	JCONF
10	10	111	100	1	1	1.70E-01	2p5 3s	2s2 2p6
10	10	111	100	2	1	3.48E-02	2p5 4s	2s2 2p6
10	10	111	100	3	1	2.31E-02	2p5 3d	2s2 2p6
10	10	111	100	4	1	1.27E-02	2p5 5s	2s2 2p6
10	10	111	100	5	1	1.25E-02	2p5 4d	2s2 2p6
10	10	111	100	6	1	6.07E-03	2p5 6s	2s2 2p6
10	10	111	100	7	1	7.02E-03	2p5 5d	2s2 2p6
10	10	111	100	8	1	3.37E-03	2p5 7s	2s2 2p6
10	10	111	100	9	1	4.24E-03	2p5 6d	2s2 2p6
10	10	111	100	10	1	2.06E-03	2p5 8s	2s2 2p6

5.3.2 SE = SElect table

This command selects a logical table from the current view for further data manipulation. Initially, the table conforms with the complete view that has been loaded into memory, resulting from commands such as Create View or Restore View. However, whenever this command is issued, a new table is constructed starting from the previous active table.

The syntax of the command is given below

```
se <row_range> <selector>
```

The command scans the `<row_range>` of the current view and selects active rows (i.e., those that have not been previously excluded) that satisfy the selection criteria specified in `<selector>`. Both `<row_range>` and `<selector>` are specified in a similar fashion to the view commands. Similarly, if no `<row_range>` is given, all the view rows are scanned. Also, if an attribute range is no specified in the `<selector>` the whole attribute domain is assumed.

Example 5.14

If we assume that the currently active table corresponds to an `e` view containing all the states of the Aluminium ions and summarized by descriptor given below

```
VIEW DESCRIPTOR
===============
DATABASE  = E
    NZ    =  13,  13
    NE    =   1,  13
    ISLP  =   0,   0
    ILV   =   0,   0
    E     =  0.0000E+00,  0.0000E+00
RECORDS LOADED =   2774
```

the subsequent command

```
se (ne 12 islp 100
```

selects from the active table all the states of Al II in the spectroscopic series with `islp` = `100`. The resulting table is displayed (`dt`) to produce

I	NZ	NE	ISLP	ILV	T	IT	LN	LL	AC	E(RYD)
2451	13	12	100	1	C	1	0	0		-1.38689E+00
2452	13	12	100	2	T	1	4	0		-5.13458E-01
2453	13	12	100	3	C	2	0	0		-3.50974E-01
2454	13	12	100	4	T	1	5	0		-2.77667E-01
2455	13	12	100	5	T	1	6	0		-1.73487E-01
2456	13	12	100	6	T	1	7	0		-1.18691E-01
2457	13	12	100	7	T	1	8	0		-8.63389E-02
2458	13	12	100	8	T	1	9	0		-6.56270E-02
2459	13	12	100	9	T	1	10	0		-5.15666E-02
2460	13	12	100	10	T	1	11	0		-4.15855E-02

Example 5.15

Assuming that the currently active table corresponds to an `f` view containing all the transitions of all the Oxygen ions, whose descriptor is

```
VIEW DESCRIPTOR
===============

DATABASE  = F
    NZ    =   8,   8
    NE    =   1,   8
    ISLP  =   0,   0
    ILV   =   0,   0
    E     =  0.0000E+00,  0.0000E+00
RECORDS LOADED =  18037
```

the command

```
se (e 3550.  3560.
```

would select all transitions with wavelengths in the range from 3550 Å to 3560 Å. The table display command `dt` leada to the following result

I	NZ	NE	ISLP	JSLP	ILV	JLV	GF	WL(A)
926	8	4	111	100	11	8	1.95E-02	3.551E+03
1502	8	4	131	120	6	12	-4.50E-02	3.555E+03
1661	8	4	140	131	7	6	6.64E-01	3.550E+03
3351	8	5	221	210	8	6	5.82E-01	3.551E+03
6115	8	6	120	111	5	8	-4.58E-04	3.557E+03
8482	8	6	321	310	8	6	3.56E-03	3.557E+03
8926	8	6	331	320	8	4	1.67E-03	3.558E+03
11775	8	7	221	210	4	15	-1.96E-04	3.551E+03
12117	8	7	221	220	4	11	-7.49E-10	3.550E+03
13634	8	7	420	411	1	3	-1.69E-01	3.558E+03
13724	8	7	421	410	2	11	-4.92E-05	3.554E+03
13752	8	7	421	410	4	3	5.48E-04	3.553E+03
14177	8	7	430	421	7	2	1.37E-02	3.551E+03
17165	8	8	321	320	4	8	-6.18E-07	3.550E+03
17185	8	8	321	320	5	12	-2.00E-07	3.551E+03
17467	8	8	331	330	4	2	7.74E-09	3.554E+03
17482	8	8	331	330	6	3	2.57E-09	3.554E+03

5.3.3 EX = EXclude rows in a table

Excluding explicit rows from a table is a useful feature. It may be performed by a command with the following syntax:

```
ex <row_range> <selector>
```

Active rows contained in the range `<row_range>` that conform to the selection criteria given in `<selector>` are excluded from the table. If `<row_range>` is not specified all rows of the active table are excluded. Similarly, if all rows conform to the selection criteria of `<selector>` an empty table will result.

Example 5.16

Assuming that the currently active table contains all states of Al II belonging to the spectroscopic series with `islp=100`, as displayed below

I	NZ	NE	ISLP	ILV	T	IT	LN	LL	AC	E(RYD)
2451	13	12	100	1	C	1	0	0		-1.38689E+00
2452	13	12	100	2	T	1	4	0		-5.13458E-01
2453	13	12	100	3	C	2	0	0		-3.50974E-01
2454	13	12	100	4	T	1	5	0		-2.77667E-01
2455	13	12	100	5	T	1	6	0		-1.73487E-01
2456	13	12	100	6	T	1	7	0		-1.18691E-01
2457	13	12	100	7	T	1	8	0		-8.63389E-02
2458	13	12	100	8	T	1	9	0		-6.56270E-02
2459	13	12	100	9	T	1	10	0		-5.15666E-02
2460	13	12	100	10	T	1	11	0		-4.15855E-02

The command

```
ex 2451
```

excludes the first row of such table and when displayed (`dt`) leads to the following result

I	NZ	NE	ISLP	ILV	T	IT	LN	LL	AC	E(RYD)
2452	13	12	100	2	T	1	4	0		-5.13458E-01
2453	13	12	100	3	C	2	0	0		-3.50974E-01
2454	13	12	100	4	T	1	5	0		-2.77667E-01
2455	13	12	100	5	T	1	6	0		-1.73487E-01
2456	13	12	100	6	T	1	7	0		-1.18691E-01
2457	13	12	100	7	T	1	8	0		-8.63389E-02
2458	13	12	100	8	T	1	9	0		-6.56270E-02
2459	13	12	100	9	T	1	10	0		-5.15666E-02
2460	13	12	100	10	T	1	11	0		-4.15855E-02

Example 5.17

The command

```
ex (ilv=3
```

performs a further exclusion in the table obtained at the end of the previous example. If the command `dt` is then issued the following display is obtained

I	NZ	NE	ISLP	ILV	T	IT	LN	LL	AC	E(RYD)
2452	13	12	100	2	T	1	4	0		-5.13458E-01
2454	13	12	100	4	T	1	5	0		-2.77667E-01
2455	13	12	100	5	T	1	6	0		-1.73487E-01
2456	13	12	100	6	T	1	7	0		-1.18691E-01
2457	13	12	100	7	T	1	8	0		-8.63389E-02
2458	13	12	100	8	T	1	9	0		-6.56270E-02
2459	13	12	100	9	T	1	10	0		-5.15666E-02
2460	13	12	100	10	T	1	11	0		-4.15855E-02

5.3.4 RE = REinstate excluded rows in table

Excluded rows may be reactivated into the current table by issuing this command. Its syntax is given by

```
re <row_range> <selector>
```

Rows of the current view in the range `<row_range>` that conform to the selection criteria in `<selector>` are activated. If no search arguments are provided in the command, the resulting table will overlap the entire view.

Example 5.18

By typing the commands

```
re 2451
re (ilv=3
```

the two rows excluded in Examples 5.16 and 5.17 are reinstated in the current table.

Example 5.19

The command

```
re
```

produces a table congruent with the current view.

5.3.5 SO = SOrt the rows of table

This command allows the user to sort the rows of the active table in ascending order of one of the floating point attributes of the active table, thus creating a new table.
The command syntax is

```
so <sort_argument>
```

where the possible sort arguments are:

- The energy `e` for tables associated to the `e` and `p` entities.
- The gf-value `gf` and wavelength `wl` for tables associated to the `f` entity.

If no sort argument is given, then the table is reset to its original order.

Example 5.20

Let us assume that the view buffers contains all the states Be I as a result of the command

```
cv e (nz=4 ne= 4
```

To sort the initial table in ascending energy order, the following command is issued

```
so e
```

The sorted table is displayed with the usual command

```
dt
```

giving

I	NZ	NE	ISLP	ILV	T	IT	LN	LL	AC	E(RYD)
1	4	4	100	1	C	1	0	0		-6.81400E-01
2	4	4	311	1	T	1	2	1		-4.81531E-01
3	4	4	111	1	T	1	2	1		-2.90398E-01
4	4	4	300	1	T	1	3	0		-2.09971E-01
5	4	4	100	2	T	1	3	0		-1.86159E-01
6	4	4	120	1	C	2	0	0		-1.63689E-01
7	4	4	311	2	T	1	3	1		-1.47660E-01
8	4	4	111	2	T	1	3	1		-1.34543E-01
9	4	4	310	1	C	2	0	0		-1.32815E-01
10	4	4	320	1	T	1	3	2		-1.19665E-01
11	4	4	300	2	T	1	4	0		-9.71418E-02
12	4	4	120	2	T	1	3	2		-9.65492E-02
13	4	4	100	3	T	1	4	0		-9.03695E-02
14	4	4	311	3	T	1	4	1		-7.60419E-02
15	4	4	111	3	T	1	4	1		-7.35530E-02
16	4	4	320	2	T	1	4	2		-6.59733E-02

```
DO YOU WANT TO SEE MORE? ==>
```

Example 5.21

The previous sorted table can be restored to its original order by typing the command

```
so
```

The resulting table is displayed with the command

```
dt
```

giving

I	NZ	NE	ISLP	ILV	T	IT	LN	LL	AC	E(RYD)
1	4	4	100	1	C	1	0	0		-6.81400E-01
2	4	4	100	2	T	1	3	0		-1.86159E-01
3	4	4	100	3	T	1	4	0		-9.03695E-02
4	4	4	100	4	T	1	5	0		-5.33551E-02
5	4	4	100	5	T	1	6	0		-3.51993E-02
6	4	4	100	6	T	1	7	0		-2.49545E-02
7	4	4	100	7	T	1	8	0		-1.86100E-02
8	4	4	100	8	T	1	9	0		-1.44103E-02
9	4	4	100	9	T	1	10	0		-1.14870E-02
10	4	4	110	1	T	2	3	1		1.29753E-01
11	4	4	110	2	T	2	4	1		2.12135E-01
12	4	4	110	3	T	2	5	1		2.44362E-01
13	4	4	110	4	T	2	6	1		2.60494E-01
14	4	4	110	5	T	2	7	1		2.69749E-01
15	4	4	110	6	T	2	8	1		2.75555E-01
16	4	4	110	7	T	2	9	1		2.79438E-01

DO YOU WANT TO SEE MORE? ==>

Example 5.22

Let us sort the table of f-values of Si II obtained with the command

```
cv f (nz=14 ne=13
```

in ascending wavelength order

```
so wl
```

We then want to display the sorted transitions in the wavelength range 5000 Å to 5100 Å. This task may be performed through the command

```
dt (e 5000.  5100.
```

obtaining

I	NZ	NE	ISLP	JSLP	ILV	JLV	GF	WL(A)
898	14	13	421	420	6	3	1.27E-02	5.009E+03
899	14	13	421	410	2	5	-2.48E-02	5.021E+03
900	14	13	421	420	2	7	-9.76E-02	5.022E+03
901	14	13	431	420	9	3	7.43E-03	5.025E+03
902	14	13	420	411	3	11	-1.00E-02	5.046E+03
903	14	13	411	400	11	2	2.30E-02	5.058E+03
904	14	13	230	221	1	7	-1.29E-02	5.085E+03
905	14	13	421	420	2	6	-1.19E-01	5.093E+03

5.4 Graphic commands

Data visualization is an important capability to be included in databases developed for scientific applications. However, it poses important portability problems since most graphic packages are directly linked to platform characteristics. The approach we have adopted in the present development in order to attain a fair degree of platform independence is to interface with standard and widely used graphic packages.

The current TOPbase prototype implements two basic graphic commands which act upon active tables and interface with the PLPLOT graphics library. The current graphic commands are

- `pt`: this command plots the column content of two attributes or attribute functions of the active table
- `px`: plots the photoionization cross sections of each state (`p` view) contained in the active table.

These two commands are differentiated since `pt` treats each active record as a point in the resulting graph while `px` plots a graph for each active record. One, two or four graphs per screen may be plotted in the latter case.

The general format of both commands is

`<graphic_command> <abscissa> <ordinate> <graphic_options> <output_device>`

In the case of table plotting (`pt`), the `<abscissa>` and the `<ordinate>` must be specified, and they can take most table attributes or attribute functions listed in the Display Table command section. In the case of plotting cross sections (`px`), the abscissa accepts the photoelectron energy `e` (abscissa default) or photon wavelength `wl`; the ordinate is always the photoionization cross section `xsec`, the evident default. The `<graphic_options>` is an unordered list of options, preceded by a left bracket "`(`" that allow to specify:

- The abscissa range `x=a,b`
- The ordinate range `y=c,d`
- Heading `hd='heading text'`; the null heading is the default when plotting tables while the concatenation of the nuclear charge, number of electrons, SLP and the level is the default when plotting cross sections.

- Combinations of linear and log scales `sc=sx,sy`, where `sx` and `sy` may be `lin` or `log` for linear or logarithmic scales respectively; the defaults are `sx=lin` and `sy=lin` for plotting tables and `sx=lin` and `sy=log` when plotting cross sections.
- Curve tracing format specifications `fm=fm1,fm2` where `fm1` may take the values `m` or `nm` for requests of point marks or no-marks respectively. Similarly, `fm2` may be specified as `l` or `nl` for line or no-line between points respectively. Note that `fm1` and `fm2` can be interchanged, and `fm=nm,l` is taken as the default.
- Number of graphs per screen `ng`, which may be set to 1 (default), 2 or 4; this option applies only to the command `px`.
- The item `<output_device>` is prefixed by the "`>`" character and followed by the file name `fn`. TOPbase will associate the name `fn` with the PostScript[3] file `fnxx.ps` where the suffix `xx` in the file name is a number from 01 to 99. This convention follows from the fact that many graph may be generated from one single plot command. Note that no more than 99 graphs can be generated at a time.

Example 5.23

In this example the quantum defects of the $3snp\ ^3P^O$ states of Mg I are to be plotted as a function of state energy. The plotting format is such that the points are marked. The following sequence of commands

```
cv e (nz 12 ne 12 islp 311
dt nz ne islp ilv e qd iconf
```

will generate the table

NZ	NE	ISLP	ILV	E(RYD)	QD	ICONF
12	12	311	1	-3.67119E-01	1.350E+00	3s 3p
12	12	311	2	-1.25115E-01	1.173E+00	3s 4p
12	12	311	3	-6.77891E-02	1.159E+00	3s 5p
12	12	311	4	-4.24764E-02	1.148E+00	3s 6p
12	12	311	5	-2.91459E-02	1.143E+00	3s 7p
12	12	311	6	-2.12455E-02	1.139E+00	3s 8p
12	12	311	7	-1.61754E-02	1.137E+00	3s 9p
12	12	311	8	-1.27271E-02	1.136E+00	3s 10p
12	12	311	9	-1.02754E-02	1.135E+00	3s 11p

Then, in order to plot the quantum defects, the following command is introduced

```
pt e qd (fm m
```

which displays the plot in Fig. 4.

[3]PostScript is a trademark of Adobe Systems Inc.

Example 5.24

We are interested in plotting the isoelectronic trend of the f-value for the resonance transition of Be sequence. We create the desired view with the command

```
cv f (ne 4 islp 100 ilv 1 jlv 1
```

The table to plot can be displayed and examined beforehand with the command

```
dt nz ne zinv gf
```

giving

NZ	NE	1/Z	GF
4	4	2.50E-01	1.40E+00
5	4	2.00E-01	1.03E+00
6	4	1.67E-01	7.80E-01
7	4	1.43E-01	6.26E-01
8	4	1.25E-01	5.22E-01
9	4	1.11E-01	4.48E-01
10	4	1.00E-01	3.92E-01
11	4	9.09E-02	3.49E-01
12	4	8.33E-02	3.14E-01
13	4	7.69E-02	2.86E-01
14	4	7.14E-02	2.62E-01
16	4	6.25E-02	2.25E-01
18	4	5.56E-02	1.97E-01
20	4	5.00E-02	1.75E-01

The command

```
pt zinv gf (fm m hd 'RESONANCE TRANSITION OF BE ISOELECTRONIC SEQUENCE'
```

will complete the task (see Fig. 5).

Example 5.25

To plot the photoionization cross sections of the lowest four states of the $^2P^O$ series of Al I, we firstly type

```
cv p (nz 13 ne 13 islp 211 ilv 1 4
```

loading the cross sections of the four states into main memory. Secondly, all the four plots are displayed in one screen display (Fig. 6) through the command

```
px (ng 4
```

Example 5.26

If a plot of cross sections vs wavelength in the range 1000 Å to 3000 Å is required for the four states in the table of the previous example, the following command would be issued

```
px wl (ng 4 x 1000.  3000.
```

to produce the plot in Fig. 7.

Example 5.27

A plot of the radiative lifetimes for the $2snp\ ^1P^O$ states of Be I as a function of n (see Fig. 8) may be obtained with the commands

```
cv e (nz 4 ne 4 islp 111
pt ln rl (sc lin log fm m
```

5.5 Index commands

These commands allow the user to obtain a table of contents of the data included in TOPbase for each ion system. This information is valuable before considering creating views of the specific databases. They provide summaries of:

- Target representation
- The authors and source reference
- Number of SLP's and states in the `e` entity
- Number of SLP's and transitions in the `f` entity

The command format is

```
di <index_argument> <selector> <output_device>
```

The `<index_argument>` may be any from the following list

- `t` for a table of contents of the target representation used in the calculation, the authors, and source reference
- `e` for a table of contents of the `e` entity
- `f` for a table of contents of the `f` entity

If no `<index_argument>` is provided, a brief summary of the three types of information listed above is output.

The `<selector>` in this case only acts to specify the ion systems in terms of ranges of their nuclear charge (`nz`) and number of electrons (`ne`).

Devices for output follow the same convention specified in previous commands.

Example 5.28

The command

```
di (nz 6
```

gives a general summary of the data for all the Carbon ions organized according to ionic system (NZ, NE), number of states in the target representation (NT), the number of SLP's and levels in the e entity (NSLP and NLV respectively), and finally, the number of SLP's and transitions in the f entity (NSLP-SLP and NTRAN respectively).

NZ	NE	NT	NSLP	NLV	NSLP-SLP	NTRAN
6	1	1	10	55	9	330
6	2	5	6	53	4	315
6	3	2	3	26	2	153
6	4	5	16	136	18	1376
6	5	6	20	184	24	2222
6	6	10	39	361	48	4764
TOTAL			94	815	105	9160

Example 5.29

This example provides a table of contents of the target representation used in the calculation for C II. It also gives the the authors responsible for the calculation and the reference.

```
di t (nz 6 ne 5
```

I	IS	IL	IP	E(RYD)	CONFIGURATION
1	1	0	0	0.00000E+00	2s2
2	3	1	1	4.77794E-01	2p(3PO)
3	1	1	1	9.40879E-01	2p(1PO)
4	3	1	0	1.25984E+00	2p2(3PE)
5	1	2	0	1.33425E+00	2p2(1DE)
6	1	0	0	1.68671E+00	2p2(1SE)

```
N+1 CONFIGURATIONS
  1  2S 2P2           2 2P3
REFERENCE:
JA Fernley, A Hibbert, AE Kingston & MJ Seaton, to be published
```

Example 5.30

The command

```
di e (nz 6 ne 5
```

results in the table of contents of the data included for C II in the `e` entity. Data are displayed in a four-column format.

NZ = 6 NE = 5 NSLP = 20 NLV = 184

I	ISLP	NLV	I	ISLP	NLV	I	ISLP	NLV	I	ISLP	NLV
1	200	9	2	201	8	3	210	10	4	211	11
5	220	10	6	221	9	7	230	7	8	231	7
9	240	6	10	241	6	11	400	8	12	401	9
13	410	9	14	411	16	15	420	14	16	421	8
17	430	7	18	431	13	19	440	11	20	441	6

Example 5.31

The command

```
di f (nz=6 ne=4 5
```

outputs, in a three-column format, the number of transitions/SLP included for C III and C II

NZ = 6 NE = 4 NSLP-SLP = 18 NTRAN = 1376

I	ISLP	JSLP	NTRAN	I	ISLP	JSLP	NTRAN	I	ISLP	JSLP	NTRAN
1	111	100	121	2	111	110	88	3	120	111	110
4	121	110	64	5	121	120	80	6	130	121	56
7	131	120	80	8	131	130	56	9	140	131	48
10	311	300	99	11	311	310	99	12	320	311	99
13	321	310	72	14	321	320	72	15	330	321	56
16	331	320	72	17	331	330	56	18	340	331	48

NZ = 6 NE = 5 NSLP-SLP = 24 NTRAN = 2222

I	ISLP	JSLP	NTRAN	I	ISLP	JSLP	NTRAN	I	ISLP	JSLP	NTRAN
1	210	201	80	2	211	200	99	3	211	210	110
4	220	211	110	5	221	210	90	6	221	220	90
7	230	221	63	8	231	220	70	9	231	230	49
10	240	231	42	11	241	230	42	12	241	240	36
13	410	401	81	14	411	400	128	15	411	410	144
16	420	411	224	17	421	410	72	18	421	420	112
19	430	421	56	20	431	420	182	21	431	430	91
22	440	431	143	23	441	430	42	24	441	440	66

5.6 Constant commands

TOPbase allows on-line access to values of frequently used atomic constants. The command format is

```
dc <constant> <selector> <output_device>
```

The `<selector>` specifications depend on the atomic or ionic nature of the constant. At present, any of the following list of constants can be specified.

- The atomic weight constant `aw` for a range of elements
- The atomic Rydberg constant `ry` for a range of elements
- The ionic ionization potential constant `eip` for a range of ions

Output conventions are similar to those given in previous commands.

Example 5.32

The command

```
dc aw (nz 1 10
```

lists the atomic weight of Hydrogen to Neon

```
NZ =   1  AW =   1.00790
NZ =   2  AW =   4.00260
NZ =   3  AW =   6.94000
NZ =   4  AW =   9.01218
NZ =   5  AW =  10.81000
NZ =   6  AW =  12.01100
NZ =   7  AW =  14.00670
NZ =   8  AW =  15.99940
NZ =   9  AW =  18.99840
NZ =  10  AW =  20.17000
```

Example 5.33

The command

```
dc ry (nz 20
```

lists the Rydberg atomic constant of Calcium

```
NZ = 20   RY = 109735.82 CM(-1)
```

Example 5.34

The command

```
dc eip (nz 7 ne 3 6
```

lists the ionization potential ionic constant of N II to N V

```
NZ =  7  NE =  3  EIP =  7.18774E+00 RYD
NZ =  7  NE =  4  EIP =  5.68369E+00 RYD
NZ =  7  NE =  5  EIP =  3.47720E+00 RYD
NZ =  7  NE =  6  EIP =  2.17775E+00 RYD
```

5.7 Exit command

The exit command in TOPbase is `qq`.

Figures