



INTERNATIONAL ATOMIC ENERGY AGENCY

# NUCLEAR DATA SERVICES

DOCUMENTATION SERIES OF THE IAEA NUCLEAR DATA SECTION

---

## ForEX: Utility Codes for EXFOR

Naohiko Otuka  
IAEA Nuclear Data Section, Vienna, Austria

October 2024

**Note:**

The IAEA-NDS-reports should not be considered as formal publications. When a nuclear data library is sent out by the IAEA Nuclear Data Section, it will be accompanied by an IAEA-NDS-report which should give the data user all necessary documentation on contents, format and origin of the data library.

IAEA-NDS-reports are updated whenever there is additional information of relevance to the users of the data library.

For citations care should be taken that credit is given to the author of the data library and/or to the data centre which issued the data library. The editor of the IAEA-NDS-report is usually not the author of the data library.

Neither the originator of the data libraries nor the IAEA assume any liability for their correctness or for any damages resulting from their use.

96/11

**Citation guideline:**

When quoting the EXFOR Utility Codes in a publication this should be done in the following way:

N. Otuka, “ForEX: Utility Codes for EXFOR”, report IAEA-NDS-0244 Rev. 2024/10, International Atomic Energy Agency, 2024 (<https://doi.org/10.61092/iaea.y0xh-2a8y>).

## **ForEX: Utility Codes for EXFOR**

Naohiko Otuka  
IAEA Nuclear Data Section, Vienna, Austria

### **Abstract**

Descriptions are given for a package of utility codes operating on the experimental nuclear reaction data files in the EXFOR format. This program package is written in Python and may be downloaded from the NRDC website (<http://nds.iaea.org/nrdc/>).

October 2024



## **Introduction**

The Utility Codes for EXFOR Utility (ForEX) are written to process EXFOR Entry files and EXFOR/CINDA Dictionary files. Currently, the following 14 codes (Python scripts) are included in this package:

- DIC227: Produce Archive Dictionary 227 from a NUBASE file.
- DICA2J: Convert Archive dictionaries to a JSON Dictionary.
- DICDIS: Prepare Archive and Backup dictionaries for distribution.
- DICJ2A: Convert a JSON Dictionary to Archive dictionaries.
- DICJ2T: Convert a JSON Dictionary to a Transmission dictionary.
- DIRINI: Split an EXFOR library tape into EXFOR entry files.
- DIRUPD: Update the EXFOR entry files with an EXFOR transmission tape.
- J4TOX4: Convert a J4 file to an EXFOR file.
- MAKLIB: Merge EXFOR entry files into a single library tape.
- MAKTAB: Produce a data table and covariance matrix from a J4 file.
- POIPOI: Remove pointers from a J4 file.
- SEQADD: Add record identification to an EXFOR file.
- SPELLS: Check English spell in free text in EXFOR format.
- X4TOJ4: Convert an EXROF file to a J4 file.

(“J4” means EXFOR in JSON.)

This document explains how to use these codes. Users need to install Python3 in their environments prior to run these codes. Any comments on the use of the codes, including difficulties encountered or any possible bug reports and suggestions are welcome.

## **Option available in all codes**

- -h Display help information
- -v Display the version
- -f Never prompt

## **Acknowledgements**

The developer would like to thank David Brown, Oscar Cabellos, Georg Schnabel and Nicolas Soppera for their comments and proposals.

### **History (major revisions only)**

2023-10-23:

- First release of 4 scripts (DIRINI, DIRUPD, MAKLIB, SEQADD)

2023-11-02:

- First release of IAEA-NDS-0244.

2024-05-03:

- First release of 6 scripts (DIC227, DICA2J, DICDIS, DICJ2A, DICJ2T, SPELLS)

2024-06-25:

- Addition of -c option to MAKLIB.
- Update of DICA2J to implement format changes of Dictionaries 25, 209 and 227 concluded in the NRDC 2024 meeting (C12 and C13).

2024-10-07:

- First release of 4 scripts (J4TOX4, MAKTAB, POIPOI, X4TOJ4).

## DIC227

This code reads a NUBASE file and a supplemental dictionary file (compilation of properties of elementary particles and natural elements in the Archive dictionary format), and convert them to Archive Dictionary227.

### Input

- NUBASE file<sup>1</sup> (-i)
- supplemental input file (-s)

### Output

- Archive Dictionary 227 file (-o)

### Option

- `-i file_nubase` Specify the NUBASE file to read
- `-s file_suppl` Specify the supplemental dictionary file to read
- `-o file_arc227` Specify the archive dictionary file to write

### Example

Convert a NUBASE file *nubase\_4.mass20.txt* and a supplemental dictionary file *dict\_arc\_sup.227* to the Archive Dictionary 227 *dict\_arc\_new.227*:

```
python3 x4_dic227.py -i nubase_4.mass20.txt -s dict_arc_sup.227  
-o dict_arc_new.227
```

---

<sup>1</sup> The NUBASE file is distributed from the Atomic Mass Data Center (<https://amdc.impcas.ac.cn/>, <https://www.anl.gov/phy/atomic-mass-data-resources>, <https://nds.iaea.org/amdc/>).



## **DICA2J**

This code reads Archive Dictionaries and convert them to a JSON Dictionary.

### **Input**

- Archive Dictionaries (-i)

### **Output**

- JSON Dictionary (-o)

### **Option**

- `-n dict_ver` Specify the dictionary version (transmission ID)
- `-i dir_archive` Specify the directory of Archive Dictionaries to read
- `-o dir_json` Specify the directory of JSON Dictionary to write

### **Example**

Convert Archive Dictionaries *input/dict\_arc.top*, *input/dict\_arc\_new.001* etc. to a JSON Dictionary *json/dict\_9130.json*.

```
python3 x4_dica2j.py -n 9130 -i input -o json
```

## DICDIS

This code reads Archive and JSON Dictionaries and process them for distribution after removal of records with the alteration flag D (deletion) and updating the year + month field (YYYYMM). At the same time, this code also produces the Backup Dictionary.

### Input

- Archive and JSON Dictionaries (-i)

### Output

- Archive, Backup and JSON Dictionaries for distribution (-o)

### Option

- `-n dict_ver` Specify the dictionary version (transmission ID)
- `-a dir_archive` Specify the directory of Archive dictionaries to read
- `-j dir_json` Specify the directory of JSON dictionary to read
- `-o dir_dist` Specify the directory of dictionaries to write

### Example

Read Archive and JSON Dictionaries *input/dict\_arc.top*, *input/dict\_arc\_new.001* etc. and *json/dict\_9130.json*, process them for distribution, and output them under the same names but under the directory *dist*. At the same time, it also produces the Backup Dictionary *dist/dan\_back\_new.9130*.

```
python3 x4_dicdis.py -n 9130 -a input -j json -o dist
```

## **DICJ2A**

This code reads JSON Dictionary and convert it to Archive Dictionaries.

### **Input**

- JSON Dictionary (-i)

### **Output**

- Archive Dictionaries (-o)

### **Option**

- `-n dict_ver` Specify the dictionary version (transmission ID)
- `-i dir_json` Specify the directory of JSON Dictionary to read
- `-o dir_archive` Specify the directory of Archive Dictionaries to write

### **Example**

Convert a JSON Dictionary *json/dict\_9130.json* to Archive Dictionaries *output/dict\_arc.top*, *output/dict\_arc\_new.001* etc..

```
python3 x4_dicj2a.py -n 9128 -i json -o output
```

## **DICJ2T**

This code reads JSON Dictionary and convert it to a Transmission (TRANS) Dictionary.

### **Input**

- JSON Dictionary (-i)

### **Output**

- TRANS Dictionary (-o)

### **Option**

- `-n trans_ID` Specify the dictionary version (transmission ID)
- `-i dir_json` Specify the directory of JSON Dictionary to read
- `-o dir_trans` Specify the directory of TRANS Dictionary to write

### **Example**

Convert JSON Dictionary *json/dict\_9130.json* to TRANS Dictionary *dist/trans.9130*.

```
python3 x4_dicj2t.py -n 9130 -i json -o dist
```

## DIRINI

This code reads an EXFOR library tape (e.g., a master file<sup>2</sup>) in EXFOR formats with a MASTER, LIB or REQUEST record as the first record, splits it into entries, and saves each entry file in an entry storage. It initialises the storage (i.e, namely delete the files in the storage directory) at the beginning of processing.

### Input

- library tape (-l)

### Output

- entry files (-d)
- log file (dirupd.log)

### Option

- -c Delete (1) trailing blanks in col. 12-66, (2) line sequential number (col.67-80) and (3) N2 of ENDBIB/ENDCOMMON/ENDSUBENT/ENDENTRY.
- -l *file\_lib* Specify the library tape to read
- -d *dir\_storage* Specify the entry storage directory to write

### Example

Initialise the entry storage directory *entry* by loading the input library tape *lib/library.txt* (without elimination of the record identification and bookkeeping if the input library has them):

```
python3 x4_dirini.py -l lib/library.txt -d entry
```

At the end of processing, one obtains entry files under *entry/1/*, *entry/2/*, etc. and a log file with a new line like

Seq.	Update date/time	Trans (N1)	Trans (N2)	Centre	Tape
0	2023-10-04 00:48:30.849567	0001	20231004		lib/library.txt

### Note

A storage initialized by DIRINI with the most recent EXFOR Master File and updated by DIRUPD with all TRANS files released after the EXFOR Master File is equivalent to the zipped version of the EXFOR Entry Files distributed from the NRDC website (<https://nds.iaea.org/nrdc/exfor-master/entry/>).

---

<sup>2</sup> The NRDC release the EXFOR Master File on an annual basis on the NRDC website (<https://nds.iaea.org/nrdc/exfor-master/>).

## **DIRUPD**

This code reads a trans tape<sup>3</sup> (starting from the TRANS record) and adds or updates the entry files in the local storage.

### **Input**

- trans tape (-t)
- entry files (-d)

### **Output**

- entry files (-d)
- log file (dirupd.log)

### **Option**

- -c (Same as DIRINI. Use this option if you use it for DIRINI.)
- -t *file\_trans* Specify the trans tape to read
- -d *dir\_storage* (Same as DIRINI)

### **Example**

Update of the entry storage *entry* by a tape *trans/trans.txt*:

```
python3 x4_dirupd.py -t trans/trans.txt -d entry
```

At the end of processing, one obtains new and/or updated entry files under *entry/1/* etc. and a log file with a new line like

Seq.	Update date/time	Trans (N1)	Trans (N2)	Centre	Tape
0	2023-10-04 00:48:30.849567	0001	20231004		lib/library.txt
1	2023-10-04 00:48:31.725707	1234	20160121	NNDC	trans.txt

---

<sup>3</sup> The NRDC assembles a set of new and revised EXFOR entries in a TRANS file on a regular basis, and it is released on the NRDC website (<https://nds.iaea.org/nrdc/exfor-master/trans/>)

## **J4TOX4** (*New*)

This code reads a J4 file and convert it to an EXFOR file (starting from an ENTRY, MASTER, REQUEST or TRANS record).

### Input

- J4 file (-i)
- JSON Dictionary (-d)

### Output

- EXFOR file (-o)

### Option

- `-i file_j4` Specify the J4 file to read
- `-d file_dict` Specify the directory of JSON Dictionary to read
- `-o file_x4` Specify the EXFOR file to write

### Example

Convert a J4 file *exfor.json* to an EXFOR file *exfor.txt* with the dictionary *dict\_9130.json*:

```
python3 x4_j4tox4.py -i exfor.json -d dict_9130.json -o exfor.txt
```

## **MAKLIB**

This reads and combines the entry files in the entry storage and create a single library tape.

### Input

- entry files (-d)

### Output

- library tape (-l).

### Option

- -a      Add “19” to two-digit year in N2 of ENTRY/SUBENT/NOSUBENT.
- -c      (Same as DIRINI.)
- -n      Exclude dictionaries
- -d *dir\_storage*      (Same as DIRINI)
- -l *file\_lib*      Specify the library tape to write
- -i *tape\_ID*      Specify an integer printed at cols 12-22 of the first record

### Example

Create a library tape *lib/library.txt* by merging entry files in the storage *entry* with the tape ID *0001*:

```
python3 x4_maklib.py -d entry -l library.txt -i 0001
```

This operation does not add a new line in the log file.

*Note:* The format of the output library tape depends on the format of the files in the entry storage. If user maintains the entry files in the storage without record identifications etc. (e.g., with -c option of DIRINI and DIRUPD), then the produced library tape also does not have them. The record identifications can be added later by processing the output library tape by SEQADD.



## **MAKTAB** (*New*)

This reads a J4 file processed by `POIPOI`, and print a four-column ( $x$ ,  $\Delta x$ ,  $y$ ,  $\Delta y$ ) table and correlation coefficients (upper triangle matrix).

### Input

- J4 file processed by `POIPOI` (`-i`)
- “HED file” (See Appendix of this manual for its details)
- JSON Dictionary (`-d`)

### Output

- a file of a four-column table and correlation coefficients (`-o`)
- log file (`-g`)

### Option

- `-s`      treat the dataset as a shape (normalization free) dataset
- `-r`      print  $\Delta y/y$  (%) rather than  $\Delta y$
- `-i file_j4`      Specify the J4 file to read
- `-j file_hed`      Specify the “HED file” to read
- `-e data_id`      Specify the dataset ID<sup>4</sup>
- `-o file_tab`      Specify the table file to write
- `-g file_log`      Specify the log file to write
- `-l x_min`      Specify the lower boundary of independent variable to process
- `-u x_max`      Specify the upper boundary of independent variable to process

### Example

Read a J4 file *exfor.json*, HED file *exfor\_hed.txt*, JSON dictionary *dict\_9130.json*, and create a table file *exfor\_tab.txt* for a dataset *22742.004.1*:

```
python3 x4_maktab.py -i exfor.json -j exfor_hed.txt -d dict_9130.json  
-e 22742.004.1 -o exfor_tab.txt
```

---

<sup>4</sup> To identify the corresponding dataset ID (e.g., 22742.004.1) in the J4 file, the J4 file must be created by `POIPOI` without the option `-d` (delete pointer).

## **POIPOI (*New*)**

This reads a J4 file, extract the information relevant to a particular dataset, and create a J4 file. It removes the pointer structure in the input J4 file. If wish, one can (1) merge the information originally compiled in the common (001) subentry and the data subentry, and (2) select the keywords (e.g., TITLE, AUTHOR) to be kept in the output J4 file.

### Input

- J4 file (-i)
- JSON Dictionary (-d)

### Output

- J4 file (-o)

### Option

- -k *keywords* Specify the BIB keywords to keep<sup>5</sup>
- -p Delete the pointer
- -1 Keep the common (001) and data subentries separately
- -i *file\_inp* Specify the J4 file to read
- -e *data\_ID* Specify the dataset ID (or “all” to process all datasets in the input)
- -o *file\_out* Specify the J4 file to write

### Example

Read a J4 file *exfor.json*, and JSON dictionary *dict\_9130.json*, and create another J4 file *exfor\_poi.json* for a dataset *22742.004.1*: by keeping the BIB records under AUTHOR and REFERENCE:

```
python3 x4_poipoi.py -i exfor.json -d dict_9128.json -e 22742.004.1  
-o exfor_poi.json
```

Same as above, but remove all data descriptions other than those under AUTHOR and REFERENCE:

```
python3 x4_poipoi.py -i exfor.json -d dict_9130.json -e 22742.004.1  
-o exfor_poi.json -k AUTHOR REFERENCE
```

---

<sup>5</sup> The REACTION information is always kept.

## **SEQADD**

This code adds and/or updates record identifications (cols.67-79) and bookkeeping information such as N1 and N2 of BIB and ENDBIB. records. (Similar to ORDER developed at NNDC and ZORDER developed at NDS).

### Input

- Entry, trans or library tape (-i)

### Output

- Entry, trans or library tape (-o)

### Option:

- -m Do not add “19” to two-digit year in N2 of ENTRY/SUBENT/NOSUBENT, and do not alter N2 of ENDBIB/ENDCOMMON/ENDDATA/ENDSUBENT/ENDENTRY/ENDSUBDICT records.
- -i *file\_inp* Specify the input EXFOR file
- -o *file\_out* Specify the output EXFOR file

### Example

Process an EXFOR file *exfor.txt* by adding and updating the record identification and bookkeeping information, and print the output in *exfor\_ord.txt*.

```
python3 x4_seqadd.py -i exfor.txt -o exfor_ord.txt
```

t

## SPELLS

This code checks English spells in the free text field in the EXFOR format. It checks each set of lower characters in free text (i.e., the first word of a sentence is not checked). The default dictionary does not know nuclear physics technical terms, and the user should add the to the dictionary to minimize the output.

Execution requires a Python module **spellchecker**, which may be installed by the following command if pip (a standard Python package manager) is installed in your computer:

```
pip install pyspellchecker
```

### Input

- Entry, trans or library tape (-i)
- Dictionary listing known technical words (-d)

Example of the dictionary file (a plain text file to be updated by the user by adding more technical terms etc.)

```
atm
deadtime
decoupler
epithermal
fluence
linac
nonuniformity
subentry
```

### Output

- log file summarizing typos (can be an argument following -l)

### Option:

- -i *file\_x4* Specify the EXFOR file to read
- -d *file\_dict* Specify the dictionary of known words to read
- -l *file\_log* Specify the log file name to write

### Example

Check spells in an EXFOR entry file *exfor.txt* with a dictionary *x4\_spells.dic* and record the checking result in *x4\_spells.log*.

```
python3 x4_spells.py -i exfor.txt -d x4_spells.dic -l x4_spells.log
```

## X4TOJ4 (*New*)

This code reads an EXFOR file (starting from an `ENTRY`, `MASTER`, `REQUEST` or `TRANS` record) and convert it to a J4 file. One can select the keywords (e.g., `TITLE`, `AUTHOR`) to be kept in the J4 file.

### Input

- EXFOR file (-i)
- JSON Dictionary (-d)

### Output

- J4 file (-o)

### Option

- `-c` Check record identification of the system identifiers
- `-k keywords` BIB keywords to be kept<sup>6</sup>
- `-g` Keep the flag at column 80 of each record of the DATA sections
- `-s` Keep real numbers as strings
- `-i file_x4` Specify the EXFOR file to read
- `-d file_dict` Specify the directory of JSON Dictionary to read
- `-o file_j4` Specify the J4 file to write

### Example

Read an EXFOR file *exfor.txt* and JSON dictionary *dict\_9130.json* and write a J4 file *exfor.json*:

```
python3 x4toj4.py -i exfor.txt -d dict_9130.json -o exfor.json
```

Same as above, but remove all data descriptions other than those under `REACTION`:

```
python3 x4_poipoi.py -i exfor.json -d dict_9130.json -e 22742.004.1  
-o exfor_poi.json -k REACTION
```

---

<sup>6</sup> The `REACTION` information is always kept.

## **Appendix 1: HED file used in MAKTAB**

(See also N. Otuka, O. Iwamoto, INDC(SEC)-0112(Rev.) Sect.2.3)

### **Basic structure**

The HED file defines the independent (x) and dependent (y) variables for tabulation by MAKTAB. It may have the following lines:

- L.1: Heading for the lower boundary of the independent variable.
- L.2: Heading for the upper boundary of the independent variable (may be repetition of the heading on L.1).
- L.3: Heading for the uncertainty or resolution of the independent variable (optional). The number under this heading is ignored if it is smaller than the difference of the numbers under the headings specified in L.1 and L.2.
- L.4: Heading for the dependent variable. The heading specified in L.4 is typically DATA or DATA-CM.
- L.5: Heading for the reference variable (optional). The heading specified in L.5 is typically MONIT. The number under this heading is used when conversion of the number to or from the fractional (%) uncertainty is required.
- L.6: Heading for the total uncertainty in the dependent variable (optional if L.7 is given). The heading specified in L.6 is typically ERR-T or DATA-ERR.
- L.7: Heading for the 1st partial uncertainty in the dependent variable (optional if L.6 is given). The heading specified in L.7 and below is typically ERR-S, ERR-1, ERR-2 etc.
- L.8: Heading for the 2nd partial uncertainty in the dependent variable (optional)

etc.

The column 1 is reserved for a flag. It must be empty in Line 1 to 6 must be empty. In Line 7 and below, this column can be empty or

- “S”: indicates the uncertainty must be ignored when the dataset is treated as a shape dataset (MAKTAB with option -s)
- “#”: indicates it is a comment line (always ignored).

The column 2 is for the first character of a heading if a heading is given on the line.

When only a range of the partial uncertainty is given under ERR-ANALYS, its lower boundary is treated as a constant of the dataset. The user may add an extra uncertainty following an ad-hoc heading such as ERR-A, ERR-B.

### **Tabulation without estimation of correlation coefficients**

When the only tabulation in the four-column (x,  $\Delta x$ , y,  $\Delta y$ ) structure is necessary, L.5 always has a correlation flag “+”. If the partial uncertainties are in EXFOR without the total uncertainty, the partial uncertainties under the heading in L.6 and below must be flagged by “-“.  $\Delta y$  is calculated by adding these partial uncertainties in % quadratically.

#### Example 1

$\Delta x$  is calculated by the difference between EN-MIN and EN-MAX.  $\Delta y$  is taken from ERR-T.

```
1: EN-MIN
2: EN-MAX
3:
4: DATA
5:
6: ERR-T      +
```

#### Example 2

$\Delta x$  is taken from EN-ERR.  $\Delta y$  is obtained by the quadrature sum of ERR-S, ERR-1 and ERR-2.

```
1: EN
2: EN
3: EN-ERR
4: DATA
5:
6:      +
7: ERR-S      0
8: ERR-1      0
9: ERR-2      0
```

#### Example 3

$\Delta x$  is always zero.  $\Delta y$  is obtained by the quadrature sum of ERR-S, ERR-1 and MONIT-ERR. The MONIT-ERR coded in other than % (e.g., in barn) is divided by MONIT for conversion to the fractional (%) uncertainty. The flag “S” on the L.9 indicates that the MONIT-ERR value is ignored when the dataset is treated as a shape dataset (i.e., MAKTAB is used with the option -s).

```
1: EN
2: EN
3:
4: DATA
5: MONIT
6:      +
7: ERR-S      0
8: ERR-1      0
9: SMONIT-ERR 0
```

#### Tabulation with estimation of correlation coefficients

The correlation coefficients are calculated on the assumption that each partial uncertainty is fully correlated or uncorrelated between two data points ( $x_1, y_1$ ) and ( $x_2, y_2$ ).

- The correlation property of each partial uncertainty is specified by a flag 1 (fully correlated) or 0 (uncorrelated).
- When all partial uncertainties are given in EXFOR, then the heading of the total uncertainty should not be specified in L.6.

When the total uncertainty is known but not all partial uncertainties are known, the “residual uncertainty” (the difference of the quadrature sum of all known partial uncertainties) is calculated for which the user must assign a correlation property.

L6 is used to specify the heading of the total uncertainty, the “residual uncertainty” (the difference of the quadrature sum of all known partial uncertainties) is calculated. The heading of the total uncertainty specified in L.6 must be followed by blanks and one of the following flags:

\*: Estimation of the correlation coefficients is skipped (e.g., when the correlation coefficients estimated by the experimentalists are available in EXFOR)

F: The residual uncertainty is fully correlated.

U: The residual uncertainty is uncorrelated.

L7 and below is used for the heading of a partial uncertainty, the heading must be followed by one of the following flags:

0: This partial uncertainty as uncorrelated. (currently not allowed when the total uncertainty heading is flagged with U)

1: This partial uncertainty is fully correlated. (currently not allowed when the total uncertainty heading is flagged with F)

-: This partial uncertainty is subtracted from the total uncertainty. (This is used only when the total uncertainty heading is specified in L.6. The 1st column must be flagged by S.)

#### Example 4

The partial uncertainty coded under ERR-S is specified as uncorrelated. The residual uncertainty is specified as fully correlated.

```
1: EN
2: EN
3: EN-ERR
4: DATA
5:
6: ERR-T      F
7: ERR-S      0
```

#### Example 5

The partial uncertainty coded under ERR-1 and ERR-2 is specified as fully uncorrelated. The residual uncertainty is specified as uncorrelated.

```
1: EN
2: EN
3: EN-ERR
4: DATA
5:
6: ERR-T      U
7: ERR-1      1
8: ERR-2      1
```



### Example 6

The partial uncertainty coded under `ERR-S` is specified as uncorrelated, and those under `ERR-1` and `ERR-2` is specified as fully uncorrelated. The residual uncertainty is specified as uncorrelated.

```
1: EN
2: EN
3: EN-ERR
4: DATA
5:
6:
7: ERR-S      0
8: ERR-1      1
9: ERR-2      1
```

### Example 7

Same as Example 6, but `ERR-2` is ignored.

```
1: EN
2: EN
3: EN-ERR
4: DATA
5:
6:
7: ERR-S      0
8: ERR-1      1
9: #ERR-2     1
```

### Example 8

Same as Example 6, but `MONIT-ERR` instead of `ERR-2`. The flag “S” on the L.9 indicates that the `MONIT-ERR` value is ignored when the dataset is treated as a shape dataset (i.e., `MAKTAB` is used with the option `-s`).

```
1: EN
2: EN
3: EN-ERR
4: DATA
5:
6:
7: ERR-S      0
8: ERR-1      1
9: SMONIT-ERR 1
```

### Example 9

Same as Example 4, but L.8 indicates that the `MONIT-ERR` is subtracted from the total uncertainty when the dataset is treated as a shape dataset.

```
1: EN
2: EN
3: EN-ERR
4: DATA
5:
6: ERR-T      F
7: ERR-S      0
8: SMONIT-ERR -
```

### Example 10

The partial uncertainty coded under ERR-S is specified as uncorrelated, and a fully correlated partial uncertainty of 1.00% not in EXFOR is added with ERR-A in L.8.

```
1: EN
2: EN
3: EN-ERR
4: DATA
5:
6:
7: ERR-S      0
8: ERR-A      1    1.00 # Normalization uncertainty 1% added
```

### Numerical example

Numerical examples are given for the correlation coefficient and total uncertainty  $\Delta y$  calculated with the various setting of correlation flags for a dataset consisting of two data points having the same total and partial uncertainties coded by

```
ERR-T      ERR-S      ERR-1      MONIT-ERR
PER-CENT    PER-CENT    PER-CENT    PER-CENT
6.          4.          1.          2.
```

for which is the L.6 to L.9 of the HED file are

```
6: ERR-T      ?
7: ERR-S      ?
8: ERR-1      ?
9: SMONIT-ERR ?
```

The next table summarizes the obtained correlation coefficient between the two data points and the total uncertainty  $\Delta y$  for various combination of the flags. The first column shows use of -s option (treat as a shape dataset) in MAKTAB. “( # )” indicates that the line is commented out.

Shape	ERR-T	ERR-S	ERR-1	MONIT-ERR	Cor	$\Delta y$
No	U	1	1	1	0.583	6.000
Yes	U	1	1	1	0.531	5.657
No	F	0	0	(#)	0.528	6.000
No	U	(#)	1	1	0.139	6.000
Yes	U	(#)	1	1	0.031	5.657
No		0	1	1	0.238	4.583
Yes		0	1	1	0.059	4.123
No	F	0	0	-	0.528	6.000
Yes	F	0	0	-	0.469	5.657

## **Appendix 2: EXFOR revision for cancellation of multiple reaction formalism**

By processing an EXFOR entry by X4TOJ4, POIPOI and J4TOX4, one can easily cancel the multiple reaction formalism. Below is a Python script and input/output demonstrating extraction of the dataset coded under the pointer A from EXFOR X0001.002. Three files x4\_x4toj4.py, x4\_poipoi.py and x4\_j4tox4.py must be placed in the same folder along with the JSON Dictionary dict\_9130.json and the EXFOR file x0001.txt. The output (x0001\_002\_a.txt) must be further processed by SEQADD to get the proper EXFOR format.

### **Script (delpoi.py)**

```
import os
import x4_x4toj4
import x4_poipoi
import x4_j4tox4

force    = True    # never prompt
chkrid   = False   # record identificaiton not checked in X4TOJ4
keykeep  = "all"    # keep all keywords in X4TOJ4 and POIPOI
keepflg  = False   # ignore flags at col.80 in X4TOJ4
outstr   = True    # print real numbers as strings in X4TOJ4
delpoin  = True    # delete pointer in the output in POIPOI
keep001  = True    # keep common subentry in POIPOI

# Conversion to J4 format by X4TOJ4
file_inp=input("EXFOR file to read (e.g., 'exfor_inp.txt'  -----> ")
file_dic=input("EXFOR/CINDA JSON Dict. (e.g., 'dict_9130.json') -> ")
file_out = "exfor0.json"
x4_x4toj4.main(file_inp, file_dic, file_out, force, chkrid, keykeep, keepflg,
outstr)

# Conversion to J4 format without pointer structure by POIPOI
file_inp = "exfor0.json"
file_out = "exfor1.json"
data_id=input("Dataset to extract (e.g., 'X0001.002.A') -----> ")
x4_poipoi.main(file_inp, file_dic, data_id, file_out, force, keykeep, delpoin,
keep001)

# Conversion to X4 format by J4TOX4
file_inp = "exfor1.json"
file_out=input("EXFOR file to write (e.g., 'exfor_out.txt') -----> ")
x4_j4tox4.main(file_inp, file_dic, file_out, force)

os.remove("exfor0.json")
os.remove("exfor1.json")

print("Processing completed")
```

### **Execution**

```
% python3 ./delpoi.py
EXFOR file to read (e.g., 'exfor_inp.txt'  -----> x0001.txt
EXFOR/CINDA JSON Dict. (e.g., 'dict_9130.json') -> dict_9130.json
ENTRY          X0001    20241006
X4TOJ4: Processing terminated normally.
Dataset to extract (e.g., 'X0001.002.A') -----> X0001.002.A
** Processing X0001.002.A
POIPOI: Processing terminated normally.
EXFOR file to write (e.g., 'exfor_out.txt') -----> x0001.002.a.txt
J4TOX4: Processing terminated normally.
Processing completed
```

## Input (x0001.txt)

ENTRY	X0001	20241006				X000100000001
SUBENT	X0001001	20241006				X000100100001
BIB	8	11				X000100100002
TITLE	Cross sections of reactions due to the action of protons having an energy of 50 MeV on C-12, N-14, O-16 nuclei					X000100100003
AUTHOR	(A.I.Vdovin, I.G.Golikov, M.N.Zhukov, I.I.Loshchakov, V.I.Ostroumov)					X000100100004
INSTITUTE	(4RUSLPI,4RUSITE)					X000100100005
REFERENCE	(J,IZV,43,148,1979)					X000100100006
FACILITY	(SYNCY,4RUSITE)					X000100100007
DETECTOR	(PLATE)					X000100100008
SAMPLE	Nuclear emulsion					X000100100009
HISTORY	(20241006C)					X000100100010
ENDBIB	11	0				X000100100011
COMMON	2	3				X000100100012
EN	EN-ERR					X000100100013
MEV	MEV					X000100100014
50.	1.0					X000100100015
ENDCOMMON	3	0				X000100100016
ENDSUBENT	18	0				X000100100017
SUBENT	X0001002	20241006				X000100100018
BIB	4	10				X000100100019
REACTION	A(6-C-12(P,2P)5-B-11,,SIG)					X000100199999
	B(6-C-12(P,P+2A)2-HE-4,,SIG)					X000100200001
	C(6-C-12(P,P+D)5-B-10,,SIG)					X000100200002
PART-DET	A(P)					X000100200003
	B(P,A)					X000100200004
	C(P,D)					X000100200005
ERR-ANALYS	(DATA-ERR) Uncertainties include error for reactions on Ag and Br, error of identification and statistical error.					X000100200006
STATUS	(TABLE,,A.I.Vdovin+,J,IZV,43,148,1979) Table 1					X000100200007
ENDBIB	10	0				X000100200008
NOCOMMON	0	0				X000100200009
DATA	6	1				X000100200010
DATA	ADATA-ERR	ADATA	BDATA-ERR	BDATA	CDATA-ERR	X000100200011
MB	MB	MB	MB	MB	MB	X000100200012
47.	9.	37.	2.3	37.	8.	X000100200013
ENDDATA	3	0				X000100200014
ENDSUBENT	18	0				X000100200015
ENDENTRY	2	0				X000100200016

## Output (x0001\_002\_a.txt)

ENTRY	X0001	20241006
SUBENT	X0001001	20241006
BIB	8	0
TITLE	Cross sections of reactions due to the action of protons having an energy of 50 MeV on C-12, N-14, O-16 nuclei	
AUTHOR	(A.I.Vdovin, I.G.Golikov, M.N.Zhukov, I.I.Loshchakov, V.I.Ostroumov)	
INSTITUTE	(4RUSLPI,4RUSITE)	
REFERENCE	(J,IZV,43,148,1979)	
FACILITY	(SYNCY,4RUSITE)	
DETECTOR	(PLATE)	
SAMPLE	Nuclear emulsion	
HISTORY	(20241006C)	
ENDBIB	0	0
COMMON	2	0
EN	EN-ERR	
MEV	MEV	
50.	1.0	
ENDCOMMON	0	0
ENDSUBENT	0	0
SUBENT	X0001???	20241006
BIB	4	0
REACTION	(6-C-12(P,2P)5-B-11,,SIG)	
PART-DET	(P)	
ERR-ANALYS	(DATA-ERR) Uncertainties include error for reactions on Ag and Br, error of identification and statistical error.	
STATUS	(TABLE,,A.I.Vdovin++J,IZV,43,148,1979) Table 1	
ENDBIB	0	0
NOCOMMON	0	0
DATA	2	1
DATA	DATA-ERR	
MB	MB	
47.	9.	
ENDDATA	0	0
ENDSUBENT	0	0
ENDENTRY	2	0





---

Nuclear Data Section  
International Atomic Energy Agency  
P.O. Box 100  
A-1400 Vienna  
Austria

---

e-mail: [nds.contact-point@iaea.org](mailto:nds.contact-point@iaea.org)  
fax: (43-1)26007  
telephone: (43-1)2600-21710  
web: <http://nds.iaea.org/>