International Atomic Energy Agency
Nuclear Data Services
DOCUMENTATION SERIES OF THE IAEA NUCLEAR DATA SECTION

IAEA-NDS-229
August 20 , 2019

PREPRO 2019

2019 ENDF/B Pre-processing Codes
(ENDF/B-VIII Verified)
Owned, Maintained and Distributed
by
The Nuclear Data Section
International Atomic Energy Agency
P.O. Box 100
A-1400, Vienna, Austria

Originally Written
by
Dermott E. Cullen
National Nuclear Data Center, BNL, alumnus
Nuclear Data Section, IAEA, Vienna, alumnus
University of California, LLNL, retired

Present Address
1466 Hudson Way
Livermore, CA 94550
E.Mail: RedCullen1@comcast.net
Website: http://redcullen1.net/HOMEPAGE.NEW/

Abstract: The codes are named "the Pre-processing" codes, because they are designed to pre-process ENDF formatted data, for later, further processing for use in applications. This is a modular set of computer codes, each of which reads and writes evaluated nuclear data in the ENDF format. Each code performs one or more independent operations on the data, as described below. These codes are designed to be computer independent, and are presently operational on every type of computer from large mainframe computer to small personal computers, such as IBM-PC and MAC (OSX). The codes are available from the IAEA Nuclear Data Section, free of charge and can be downloaded from http://www-nds.iaea.org/ndspub/endf/prepro/

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

e-mail: services@iaeand.iaea.org
fax: (43-1) 26007
cable: INATOM VIENNA
telex: 1-12645
telephone: (43-1) 2600-21710

Online: TELNET or FTP: iaeand.iaea.org
username: IAEANDS for interactive Nuclear Data Information System
username: ANONYMOUS for FTP file transfer;
Web: http://www-nds.iaea.org

Note

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

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

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

Citation guidelines

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

This computer code package should be cited as follows: D.E. Cullen, "PREPRO 2019: 2019 ENDF/B Pre-processing Codes", report IAEA-NDS-229, August 20, 2019.

Contents

1. NUCLEAR DATA SECTION INTRODUCTION

Here I attempt to distinguish between the ENDF **format**, and the **data in the format.** ENDF is the internationally agreed upon **format** for dissemination of evaluated nuclear data; it now been through six (6) versions, ENDF-1 through the current ENDF-6 **formats**. In contrast, the ENDF/B **data library** has now been through eight (VIII) versions; the latest identified as ENDF/B-VIII. Until recently the **format** and **data in the format** corresponded because ENDF/B-I through ENDF/B-VI **data** were in the ENDF-1 through ENDF-6 **format**. However, for later libraries, such as the ENDF/B-VIII data library the formats were not changed, so that the VIII **data** is in the ENDF-6 **format**.

Documentation for the current ENDF format and convention is available in ENDF-102, from the National Nuclear Data Center, Brookhaven National Laboratory
http://www.nndc.bnl.gov/nndcscr/documents/endf/endf102/

The 2019 ENDF/B Pre-processing codes process nuclear data formatted in any version of the ENDF **formats**; ENDF-1 through ENDF-6 evaluations, i.e. all versions of the ENDF/B **data**, I through VIII. These codes can be used on virtually any computer: everything from large mainframe computers, to workstations, to IBM-PC (Windows or Linux) and MAC (OSX).

These codes are available free of charge upon request from the Nuclear Data Section (see addresses on cover page) or downloaded from the Nuclear Data Section Web page.

The present documentation (revision 19) completely supersedes all previous documentation of earlier versions of the ENDF/B Pre-processing data. **It is strongly recommended that you use ONLY the latest 2019 version of the PREPRO codes. Failure to heed this warning can lead to completely erroneous results**.

2. CONDITIONS FOR USE OF THE CODES

Any comments on the use of the codes, including difficulties encountered or any suggestions should be sent to the IAEA Nuclear Data Section. If any results obtained from using these codes are used or referenced in a publication, a copy of the publication should be sent to the IAEA Nuclear Data Section.

3. DEDICATION

Regardless of whose name appears on the cover of this report, much of the work involved in maintaining, testing and distributing the previous and current versions of the PREPRO codes, was done by **Kevin McLaughlin** (NDS, IAEA, Vienna). For over 30 years Kevin has played an invaluable role in updating and testing the PREPRO codes. After all these years I am sorry to have to report that Kevin has now retired. I think I can speak for all present and past members of the Nuclear Data

Section and nuclear data community in saying that Kevin will be greatly missed both as a co-worker and as a good friend.

4. ACKNOWLEDGEMENT

Today's PREPRO 2019 codes represent the efforts of many more people than just the author. First, I must acknowledge the contributions of those who prepared the PREPRO 2019 package for release. I thank **Jesse David Norris** (LLNL), for preparing the MAC-OSX and LINUX executables for PREPRO2019. I thank those who prepared executables for earlier version of PREPRO and tested the 2019 version: this includes **Jean Christophe Sublet** (NDS, IAEA), for MAC (OSX) executables and **Andrej Trkov** (NDS,IAEA) and **Bojan Zefran** (IJS, Slovenia), for the LINUX executables. I express thanks to **Arjan Koning** (NDS, IAEA) for testing the codes using many different compilers, and for doing a truly amazing job finding incredibly subtle faults, which because of his efforts have now all been corrected. I also thank **Andrej Trkov** and **Kira Nathani** (Nuclear Data Section, IAEA, Vienna), who are now in charge of PREPRO at IAEA, and prepared the final PREPRO 2019 package for release on the NDS website.

I thank **Bob MacFarlane** (LANL), for the decades of cooperation and coordination between NJOY and PREPRO that we have shared; together we recognized that our codes are far too complicated to allow us to assume that hard work and good intentions are enough to make them accurate and cooperation benefitted both of us and our codes – **only by code comparisons can we be confident of our results**. Now that Bob has retired I will greatest miss this cooperative effort and Bob's keen insight.

I gratefully acknowledge the contribution of **S. Ganesan** (BARC, India), **Andrej Trkov** (NDS, IAEA) and **Jean Christophe Sublet** (NDS, IAEA) in continuing to propose interesting and useful extensions to these codes; keep those great ideas coming guys I also acknowledge **Janice Arwood** and **Mark Baird** (RSICC, Oak Ridge), for their review of the documentation and testing of codes prior to their distribution through RSIC. I am sorry to announce that **Jennie Manneschmidt**, who worked on these codes at RSICC for so many years, and if my memory serves me right is the one who thought up the name PREPRO, has now retired; she will be greatly missed by the entire computer code and nuclear data communities.

5. HISTORY AND TERMINOLOGY

Originally the Evaluated Nuclear Data File (ENDF) was divided into two different **formats**: **ENDF/A** which was designed to contain partial evaluations that might later be incorporated into complete evaluations, and **ENDF/B** which was designed to contain complete evaluations for use in applications. Originally these were physically two completely different formats, but circa 1970, when I worked at the NNDC, Brookhaven, I realized that two different formats were not needed, so we abandoned the ENDF/A format and adopted the ENDF/B format for both partial (A) and complete (B) evaluations; this is what we today call the **ENDF format**. Here I distinguish between the format, such as the current **ENDF-6 format**, and the data in the format, such as the current **ENDF/B-VIII data**.

I try to distinguish between the **ENDF-6 format**, and **the ENDF/B data** that is coded into this format. The **ENDF-6** format is now used universally to store evaluated nuclear data: in the United Stated the data is referred to as **ENDF/B-VIII**, in Western Europe, **JEFF**, in Japan, **JENDL**, in China, **CENDL**, in Russia, **BROND**, etc. Here I will not be concerned with the differences between the **contents** of these data libraries. My only concern will be with the common **ENDF-6 format**, that all these data libraries use. The PREPRO codes are designed to process evaluated data in any version of the ENDF format. The **ENDF format** has now been through six major versions, with the current format defined as **ENDF-6**. In contrast the United States **ENDF/B data library** has now been through eight major versions, with the current data library defined by **ENDF/B-VIII**.

6. FEATURES OF 2019 VERSION

6.1. What is New

The 2019 version is a major change in both the overall direction and actual coding of the PREPRO computer codes. The PREPRO codes have a long history extending back to the origins of the ENDF effort over 50 years ago. During the early years of ENDF, from 1967 through 1972, I worked at what is now the National Nuclear Data Center (NNDC). During these years I started PREPRO to help the then new ENDF system conform to the new ENDF formats and procedure, we were all then learning.

At the time the overall direction was for the PREPRO codes to carefully check for problems and whenever possible correct the evaluated data. The objective was always an attempt to guarantee ENDF formatted data conformed to ENDF formats and conventions and gave a unique interpretation of ENDF formatted data, e.g., without a unique interpretation meaningful data testing and comparing the results from a variety of nuclear and atomic data processing and application codes would be impossible, and indeed can be misleading. The approach from the very start of PREPRO was to use Howerton's first law: "The are in one rush for the wrong answer"; we tried to design PREPRO results to be as precise as possible regardless of how long it took to run our computer codes.

This effort did indeed contribute to improving our nuclear data, but it is fair to admit that the search for a "unique" interpretation has not been completely successful, because much of the PREPRO processing and correction was done AFTER the evaluations were completed and distributed to users. Unfortunately, by then many processing codes were making their attempts to "uniquely" interpret the data, not always using the same rules as PREPRO.

Starting with PREPRO 2019 the overall direction has been changed to reflect that today ENDF is mature, and by now rather than attempting to interpret and correct ENDF formatted data, the PREPRO does much more extensive checking and report problems to users, in the hope that these WARNING and ERROR messages would be used by evaluators, to improve and correct their data BEFORE their evaluations are distributed. To give but a few examples:

PREPRO/RECENT has always reconstructed energy dependent cross sections (MF=3) from a combination of resonance parameters (MF=2 data) and tabulated "background" cross sections (MF=3). However, the reconstructed cross sections from MF=2 are defining at 0 Kelvin temperature and can only to combined with the tabulated background (MF=3) if it is also at 0 Kelvin temperature. If both were not at 0 K temperature, earlier version of PREPRO/RECENT printed a WARNING message that it would ignore the background and only output the resonance parameter contribution. There are at least three problems with this approach: a) Obviously the MF=3 was incorrect, b) The temperature is now defined in MF=1, so it was output as input (greater than 0 K), making the MF=3 even further incorrect by identifying it as at temperature a temperature greater than 0 K, c) unfortunately, these days fewer and fewer code users read the WARNING and ERROR messages output by PREPRO codes so they had no idea that the output MF=3 data was incorrect. **Starting with 2019 PREPRO/RECENT if the MF=1 temperature is not 0 K, the code will print an ERROR message and TERMINATE, without producing any reconstructed cross sections; hopefully users will note this result.**

A second example, earlier version of several of the PREPRO codes would extend tabulated cross sections that ended below 20 MeV up to 20 MeV as constant equal to the last tabulated value. This was useful with early versions of ENDF formatted data, but by today most evaluations extend above 20 MeV and having PREPRO codes make these extensions can be misleading. **Starting with 2019 the PREPRO codes check for consistency that all data extends up to the same maximum energy or ends at a lower energy with zero cross section, and print both the maximum energy, and a tables of MF/MT data that stops at a lower energy.** For example, for ENDF/B-VIII this check found that for U235 and U238 the cross sections extend up to 30 MeV, but the tabulated fission spectra extend up to about 32 MeV; what users are supposed to do with neutrons emitted above 30 MeV, where we do not have any cross sections is no clear (possibly leading to a non-unique interpretation).

Another example is that earlier versions of PREPRO/GROUPIE output self-shielded cross sections and multi-band parameters in the resolved resonance region and tabulated energy range, but only unshielded cross section in the unresolved resonance region. **Starting with 2019 PREPRO/GROUPIE output also included self-shielded cross section and multi-band parameters for the unresolved resonance region, in both a simple tabulated format and a pseudo-ENDF format for use with NJOY/MCNP.**

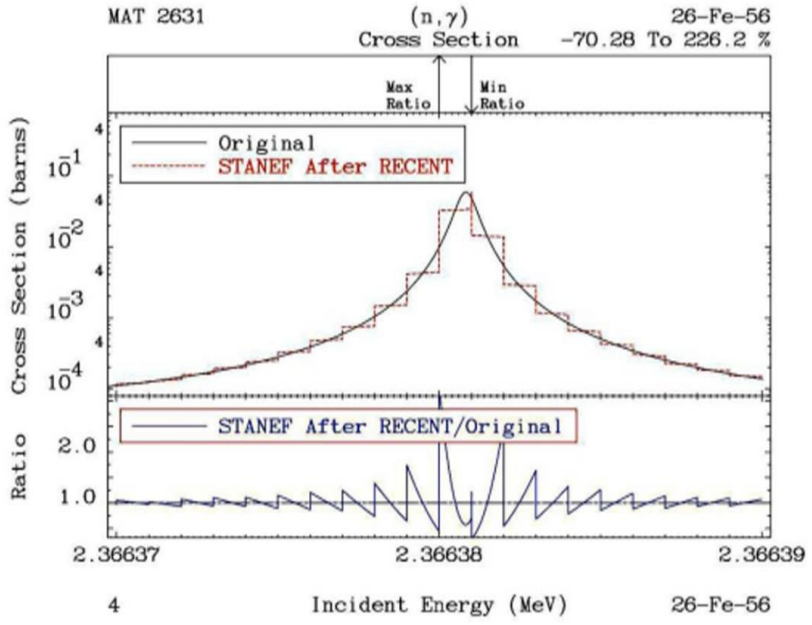
Much of the following refers to the 2019 version. Compared to earlier versions of these codes the 2019 version has the following features.

Codes Modernized: The codes have ALL been modernized and updated to make them even more compatible for use on ANY COMPUTER. All of the codes are now designed to be 100% compatible for use on 32 or 64 bit computers. All the codes have been further optimized and improved; in most cases the improvements are based on feedback from code users; this feedback is most appreciated and benefits all of us, by ensuring that the codes are as compatible as possible with our needs.

The 2019 versions are bigger and faster than preceding versions, in line with the ever-increasing size and speed of our computers. This will allow you to treat much

larger and more complex problems in a reasonable amount of time. It has also led to improved precision of the tabulated cross sections, particularly for very narrow resonances. Code Modernization (described above) allows the codes to be further sped up by using very aggressive optimization; this was fully tested and verified.

PREPRO uses 9 or 10 digit precision for all ENDF output. For example, consider: 9 digits: 12324.56789, versus the same number rounded to 7 digits: 1.234568+E3. Here in we can see that the 9 digit output is a hundred times more precise compared to the 7 digit output. This is very important for narrow milli-eV wide resonances in the keV or today even in the MeV energy range. Below I show the difference between PREPRO 2019 output data using 9 or 10 digits of accuracy compared to STANEF using 7 digits. For this narrow resonance the differences are very dramatic, with PREPRO showing a smooth resonance shape and STANEF showing a Ziggurat (i.e., a stepped pyramid). In both cases the numbers are identical INSIDE the computer, and the differences seen here are solely due to the precision each code uses to OUTPUT data in the ENDF format, resulting in **differences up to over a factor of 2, i.e., not 2%, a factor of 2, over 100% differences.**



FORTRAN, C and C++ Compatible ENDF results: I have added the ENDF2C code to PREPRO, to ensure that ALL PREPRO output in the ENDF format are completely FORTRAN, C and C++ compatible. As of today (August 2019) evaluated data even from major code centers are still not completely FORTRAN, C and C++ compatible. Therefore, when I begin pre-processing any evaluation the first PREPRO code I run is ENDF2C to ensure that ALL ENDF formatted output in subsequent codes are completely compatible. This is a very important step: it would be such a shame if after all the effort invested to produce accurate results it cannot be accurately read and used by application codes. If as recommended you ALWAYS

use ENDF2C first you will be able to avoid this problem. To maintain 100% ENDF-6 format compatible PREPRO2019 also uses the current ENDF convention that sequence numbers start at 1 for each section (MAT/MF/MT), instead of the older convention starting at 1 for each material (MAT).

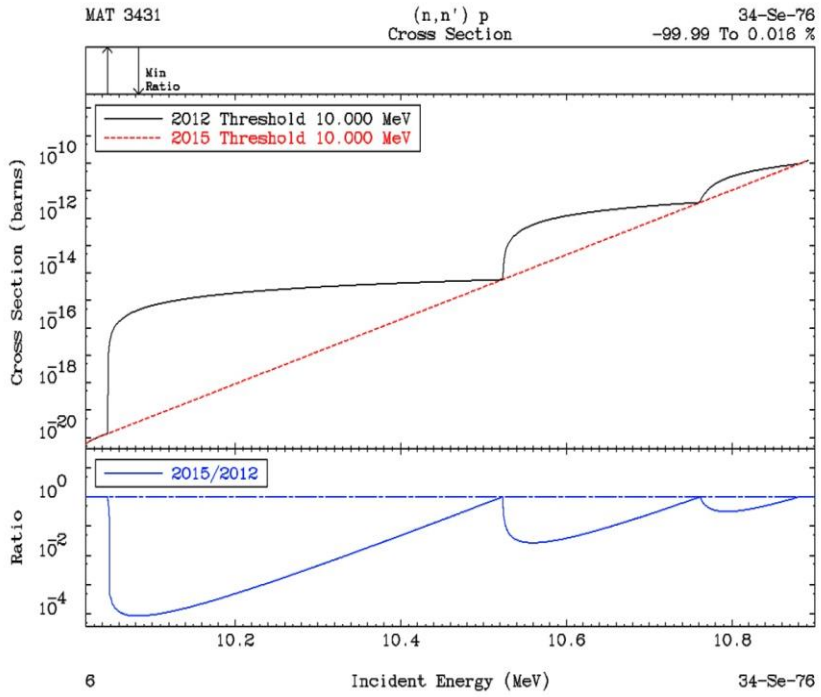
Before ENDF2C

```
1.002000+3 1.996800+0          0          0          0          0 128 3  1  125
0.000000+0 0.000000+0          0          0          1        149 128 3  1  126
       149          2                                              128 3  1  127
1.000000-5 3.420300+0 1.000000-4 3.403000+0 2.530000-2 3.395510+0 128 3  1  128
1.000000+2 3.395010+0 1.000000+3 3.394900+0 2.000000+3 3.394800+0 128 3  1  129
3.000000+3 3.394400+0 4.000000+3 3.389400+0 5.000000+3 3.385000+0 128 3  1  130
1.000000+4 3.367000+0 2.000000+4 3.342000+0 3.000000+4 3.321000+0 128 3  1  131
4.000000+4 3.302000+0 5.000002+4 3.285000+0 6.000002+4 3.270000+0 128 3  1  132
```

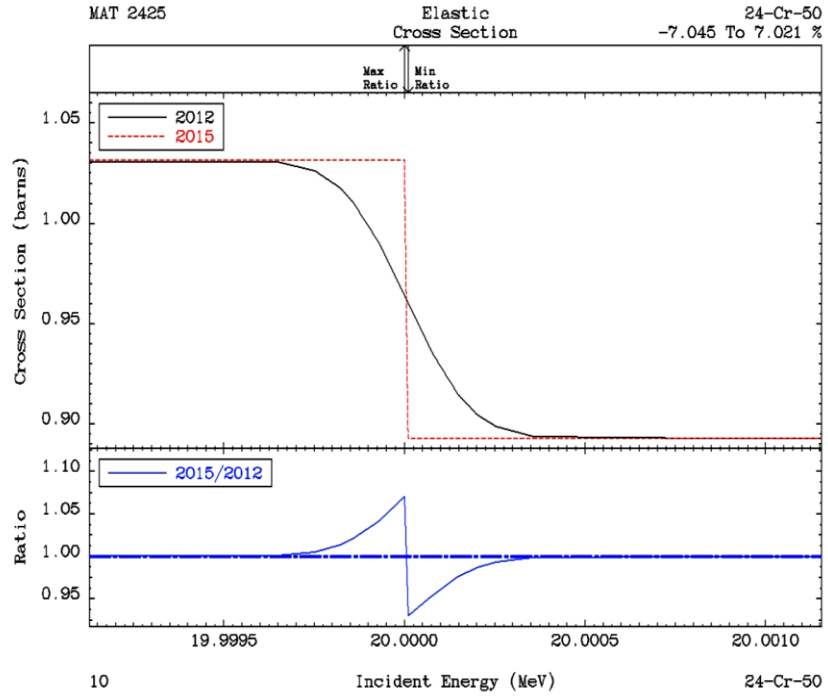
After ENDF2C

```
1002.00000 1.99680000          0          0          0          0 128 3  1    1
0.0        0.0                 0          0          1        149 128 3  1    2
       149          2                                              128 3  1    3
1.00000E-5 3.42030000 1.00000E-4 3.40300000 .025300000 3.39551000 128 3  1    4
100.000000 3.39501000 1000.00000 3.39490000 2000.00000 3.39480000 128 3  1    5
3000.00000 3.39440000 4000.00000 3.38940000 5000.00000 3.38500000 128 3  1    6
10000.0000 3.36700000 20000.0000 3.34200000 30000.0000 3.32100000 128 3  1    7
40000.0000 3.30200000 50000.0200 3.28500000 60000.0200 3.27000000 128 3  1    8
```

Improved BEST Input Parameters, based on extensive use of the earlier versions of the PREPRO codes. Of particular note is decreasing the minimum cross section from 10^{-10} to 10^{-30} barns to be linearized (tabulated data below the minimum are copied, ignoring the ENDF interpolation code). This has a rather dramatic effect, particularly on (neutron, charged particle) reactions, which often have long, slowly decreasing tails toward the reaction threshold. Here the cross section can be quite small, but extend over a large energy range, so there might be an integral effect; in the below plot interpolated values differ by up to a factor of 1 million – **let me repeat that: a factor of 1 million**. Since this extension has only a minor effect on the overall size of the pre-processed ENDF data it is now accurately included.



Doppler Broadening High Energy Cutoff: Today many modern evaluations extend to very high energies well above the traditional ENDF 20 MeV high energy limit. In these cases, the theoretical models used for the evaluations change at or near 20 MeV, which can cause an abrupt change (a non-physical discontinuity) in cross sections. To compensate for the "intent" of the evaluators, PREPRO Doppler broadening now only extends up to 10 MeV. This has the effect of making the "discontinuities" in the cross section at or near 20 MeV**, temperature independent**, which I judge to be the "intent" of the evaluators.



ENDF/B Tested: All of ENDF/B-VIII evaluations have been processed to high precision at many temperatures to create **POINT2018 (Version V**III) data, and the results are now available on-line at NNDC. BNL.

http://www.nndc.bnl.gov/endf/b8.0/POINT2018\

ENDF2C is designed to ensure that ALL PREPRO output in the ENDF format are completely FORTRAN, C and C++ compatible. As of today (August 2019) evaluated data even from major code centers are still not completely FORTRAN, C and C++ compatible. Therefore, when I begin pre-processing any evaluation the first PREPRO code I run is ENDF2C to ensure that ALL ENDF formatted output in subsequent PREPRO codes are completely compatible. This is a very important step: it would be such a shame if after all the effort invested to produce accurate results it cannot be accurately read and used by application codes.

SPECTRA was a new code for PREPRO 2010, which starting from models and tabulated data, linearizes and tabulates neutron emission spectra (MF=5); it is similar to and is an extension of the LINEAR code that performs a similar function for cross sections (MF=3). It has been extended for 2019.

RECENT for 2019 is extended to handle multiple resolved resonance energy ranges for the general Reich-Moore (LRF=7) resolved resonance formalism. The other resolved formalisms calculate and output total, elastic, capture and fission cross sections. The general Reich-Moore allows many more output channels; RECENT 2019 has been further extended and allows up to 10 output channels, and outputs cross sections are all these channels in the ENDF format. The extension to multiple LRF=7 resonance regions makes PREPRO capable of handling all current ENDF/B-VIII and planned evaluations.

SIGMA1 for 2019 has been updated for improved low energy treatment, as well as improved accuracy and consistency throughout. For 2019 Doppler broadening is now restricted to an upper limit of 10 MeV, i.e., all tabulated cross sections at energies higher than this are assumed to be temperature independent and are copied exactly as read from the ENDF input to the ENDF output.

SIXPAK and **ACTIVATE** have been extended to handle newer data that can now be coded using MF=3, 6, 9, and 10 formats.

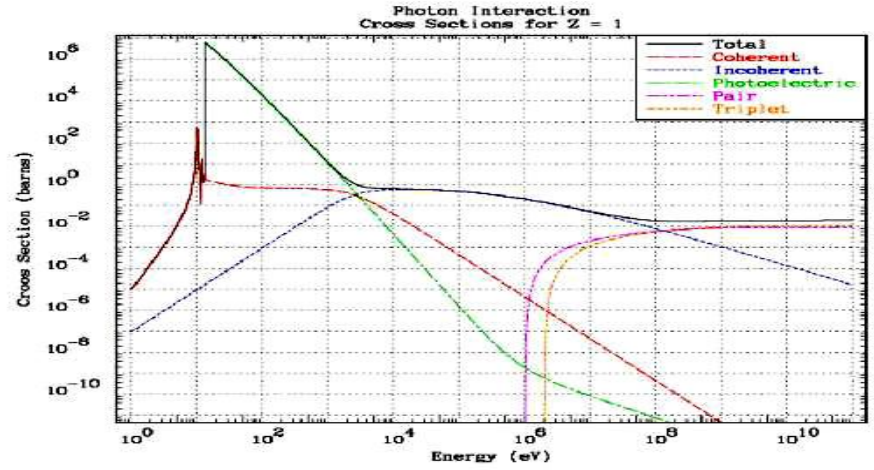
BEST INPUT is provided separately for all codes. As distributed PREPRO includes a series of test cases to quickly run each code to insure it is operating correctly. The input for these test runs are designed to allow adequate testing in a reasonable amount of time; as such this input may not correspond to what we recommend for your production work. **What we recommend you use** for each code for your actual applications is provided in a separate directory. **After you have tested the codes it is recommended that you use the BEST INPUT, which is distributed with PREPRO 2019. This BEST** input will guarantee that PREPRO results will be generated to high accuracy needed for use in applications.

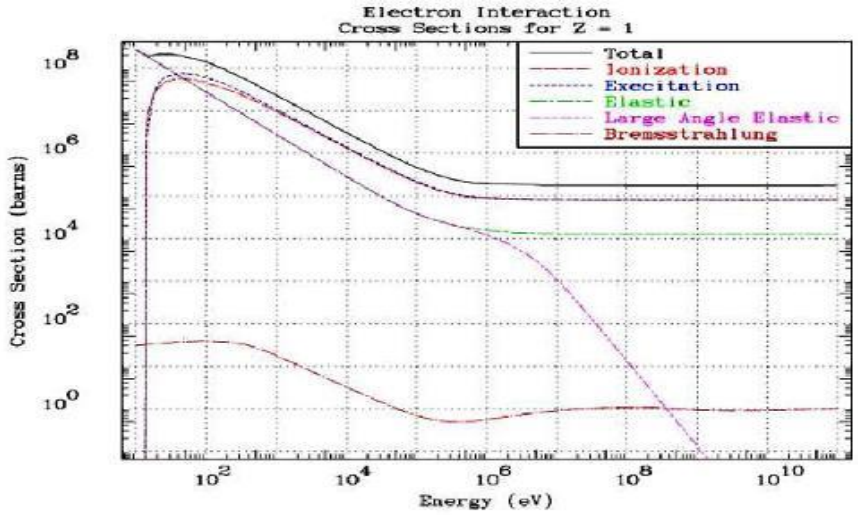
LINKING and TRACKING sequences of codes, has now been simplified by having each code identify itself when it starts and when it finishes correctly in its output report (.LST files), and if it does not finish correctly each code will identify the problem that caused it to terminate and print an ERROR message rather than the code name. This allows users to automate and run long sequences of codes and still easily monitor performance. For example, to create the POINT2009 (VII.0), POINT2015 (VII.1), and POINT2019 (VIII) data libraries only required me to run one non-interactive batch file to process hundreds of evaluations, requiring thousands of code executions (each evaluation was processed using a series of codes). Using this approach, what used to take months to accomplish can now be done in a day.

More Complete packages are included for each type of computer; in particular the graphics codes EVALPLOT and COMPLOT are now included so that users can quickly view nuclear data on their computer screen and/or produce Postscript files for later use, i.e., as in reports. Interactive graphics are a powerful tool that allows us

to quickly check the enormous amount of data currently included in modern nuclear data libraries.

More Graphics Output, so that users can "see" and check data using the PREPRO plotting codes COMPLOT and EVALPLOT. This extension applies to all the ENDF/B-VIII Electron and Photon data, EPICS2017, as illustrated below.





6.2. Recommended Accuracy

There is almost no cross section for any material, at any energy and temperature that we know to better than an uncertainty of about 1%. With the PREPRO codes we try to introduce an additional UNCERTAINTY due to processing that is much less than the basic UNCERTAINTY in the cross-section data itself. This is done to ensure that the final combined uncertainty is essentially equivalent to only the UNCERTAINTY in the basic data itself, i.e., **our data processing introduced no significant additional uncertainty.**

We therefore recommend that integral **cross sections** be processed using an allowable uncertainty of 0.1%, and in the thermal, low energy range 0.01%; we use the latter because the thermal range is better known (at least in an integral sense), and has few if any resonances, so that the cross sections are smoothly varying and can be very accurately represented using a relatively small number of tabulated data points.

Angular and energy distributions are less well known and normalized. We therefore recommend that these be reconstructed to within an accuracy of 1%.

You, the PREPRO code user need not memorize these recommendations, because these criteria are included in the **BEST INPUT** supplied with PREPRO 2019. **All you need remember is to use the BEST INPUT.**

6.3. WARNING About File Formats

We have made every effort to ensure that the files for each computer are in the correct native format for that computer. But you should be WARNED that some files may be ONLY in IBM-PC (DOS) format. If you successfully verify the installation on your computer, you can be confident that the files are in the correct format for use on your computer.

If you have an installation problem, it may be due to file formats. This may cause problems if you attempt to use them in this form on other types of computers. The major difference between files is how the end of each line is defined. The characters: carriage return (CR) and/or Line Feed (LF) are used at the end of line. For example,

Windows (DOS)	CR/LF	= 2 characters
UNIX	LF	= 1 character – for this and all below systems
LINUX	LF	
MAC (OSX)	LF	
VMS	LF (implied; UNIX compatible)	
MAC Classic	CR (no longer supported)	

This means the ENDF 80 characters/line format, will include 82 characters for Windows (DOS), or 81 characters for the other systems listed above, e.g. you can easily see this if you open any ENDF format data file, note how many lines are in the file, and compare this to the listed byte size of the file; you will find that the file size is either 81 or 82 times the number of lines in the file.

For PREPRO 2019 we supply both DOS and UNIX compatibles files; this combination will work on all the computers that we supply packages for: Windows,

LINUX (UNIX) and MAC (OSX). We have found that DOS executables can read MAC or LINUX formatted **DATA** files, and MAC and LINUX executables can read DOS formatted **DATA** files. It is only sources codes, Makefiles, etc. that we find to be system dependent.

Failure to insure files are in the correct format can lead to unreliable results. Fortunately, converting files from DOS to other formats for use on other types of computers is usually simple and straightforward. For example,

1) On UNIX type computers (today this includes LINUX and MAC), you can use **dos2UNIX** to convert files and **chmod** if a file is to be executed, e.g.,

```
dos2UNIX sun.mak SUN.MAK
chmod 777 SUN.MAK
```

2) On other types of computers, you need merely use a word processing code to edit the file – check the end of each file and delete any blank lines that you may find; generally when you then close the file it will be saved in the local format for use on your computer.

6.4 Running Time

It wasn't too many years ago that in order to process major ENDF/B evaluations we needed super, million dollar computers, and even then, it could take days to process a large evaluation, such as U-238.

Need I say it: **those days are gone forever**. Today even small personal computers can be used to quickly process any ENDF evaluations. For example, on an IBM-PC, a $ 300 computer, I can process U-238 in the time it takes me to go and get a cup of coffee - and with the next generation, it will require even less time.

So I am not going to list typical running times for the codes, for two reasons: 1) the running times have now become trivial, and 2) by the time you get a copy of this report any times I quote here will be outdated by the availability of newer, faster, and cheaper computers.

Bottom line: **running time is no longer a major concern in processing ENDF data**, and even small personal computers are now powerful enough to be used to process all ENDF evaluations. This has allowed us to change our focus to producing much more reliable results, to high precision, e.g., 9 or 10 digit floating point output numbers in the ENDF format.

6.5. All ENDF Formats and Procedures

These codes can automatically determine the ENDF format version that each evaluation is coded in (ENDF-1 through ENDF-6 format), and use the appropriate procedures. It should be particularly noted, that these codes now handle all ENDF formats and procedures through ENDF-6, and they have been tested with all the ENDF/B-VIII evaluations.

WARNING: The 2019 codes include extensions to handle all current ENDF formats and procedures, and corrections to problems that existed in earlier versions of these

codes. **As such the 2019 codes completely supersede all earlier versions of PREPRO and it is strongly recommended that all users of these codes only use the 2019 version of these codes. Failure to heed this warning can lead to completely erroneous results.**

6.6. Consistent Handling of All ENDF Formatted Data

All the PREPRO codes now use the same routines to handle all ENDF formatted input and output. This has resulted in a completely consistent interpretation of all ENDF formatted data by all the codes and has also allowed the precision of the ENDF output to be consistently extended in all codes. For 2019 if you follow the recommended procedures, and first use ENDF2C, the ENDF output will be completely consistent for input into FORTRAN, C and C++ codes, while still maintaining the accuracy of the data.

6.7. Optional Input Parameters

All PREPRO codes, except ENDF2C, now allow input parameter files and allow ALL input parameters to be optional; all input parameters now have built-in default values. Of note is that allowable uncertainties are now optional input. This allows us to select what we consider the best choices, based on the most recent advances in the speed and size of computer, and we can change these default values built into the codes, without you having to be aware of these changes or you having to change your input parameters.

6.8. Computer Independence

The PREPRO2019 codes are now 100% compatible for use on IBM-PC, LINUX or MAC computers/systems. For other types of computers, the only computer dependence in the 2019 codes is to define running time. Routines to define running time are supplied for most types of computers, and instructions are provided in this report to help you define a timing routine for any other type of computer.

6.9. 32 versus 64 bit Computers

All the codes are now designed to be 100% compatible for use on 32 or 64 bit computers. Executables are provided for,

1) **32 bit Windows** – these will execute on 32 and 64 bit Window systems. So far, I have not found any significant speed difference between 32 and 64 bit executables, and both give the same answers.

2) **64 bit MAC** – MAC computers have been 64 bit for many years, so today we deem it only necessary to supply 64 bit executables.

3) **32 and 64 bit LINUX** – executables are **not** completely interchangeable between 32 and 64 systems using LINUX, so we provide both; choose whichever meets your needs.

6.10. MAC OSX Executables

Earlier versions of PREPRO supplied executables for MAC OS9. The current PREPRO supplies executable for MAC OSX (there are no executables for OS9 included). Under OSX the codes run much faster than under OS9. Under OSX the

codes appear to the user very similar to how they appear on a UNIX or LINUX computer.

6.11. Bigger, Faster, Improved Accuracy

In line with the enormous increase in computer sizes during the last few years, the 2019 versions are **bigger**, allowing more complicated problems to be run much more efficiently, and in general allowing each problem to be run much **faster**.

All of the codes now use double precision throughout, resulting in **improved accuracy**. Compared to the earlier versions that used a mixture of single and double precision, with modern compilers and hardware, using double precision throughout has also contributed to making the codes **faster**.

6.12. Optimization

Be WARNED that if you create your own executables using optimization options can affect the precision of calculations; in many cases the executables will only use 32 bit, rather than 64 bit, precision. Optimization is designed by make codes run faster, but be WARNED that this may be accomplished by sacrificing accuracy. **As such use caution and verify results whenever using optimization.**

6.13. On-Line Reports

All of the codes now include an on-line report to your screen (except for the graphics codes, that use the screen), and a report to an output file; the on-line report allows users to monitor the progress of each code as it executes. Earlier versions of some codes had no on-line report; as far as what the user saw, the code started and ran to conclusion without printing anything on-line. This made it impossible to monitor the progress of each code, and for long running problems often resulted in users terminating the codes before they completed execution, because it appeared that the codes weren't doing anything.

6.14. Execution Timing

ALL the codes now include a timer, to print execution time at the end of processing each evaluation (MAT), and at the end of execution.

7. FEATURES OF ALL VERSIONS

7.1. Code Documentation

These codes are designed to be self-documenting, in the sense that the latest documentation for each code is included as comments at the beginning of each code. Printed documentation, such as this report, is periodically published and consists mostly of a copy of the comment lines from the beginning of each code.

The user should be aware that the comment lines within the codes are continually updated to reflect the most recent status of the codes and these comments within the codes should always be considered to be the most recent documentation for the codes and may supersede published documentation, such as this document. **Therefore, users are advised to always read the documentation within the actual code that is being used.**

7.2. Data Documentation

It is essential that the pedigree of the evaluated data you use in your applications be documented. This is the purpose of the comment lines at the beginning of each ENDF/B evaluation (in MF/MT=1/451). The PREPRO codes are designed to supplement the evaluator supplied comments by documenting any operations that they perform on ENDF/B data, that changes the evaluated data in any way. If one of these codes produces ENDF/B formatted output which in any way effects the actual evaluated data, what the code did is documented by adding additional comment lines at the end of the comment lines at the beginning of each evaluation, defining the code and input parameters that it used. The sequence of all such comments completely documents all of the operations that have been performed on the data. **Code users are advised that it is very important to leave this documentation directly inside each evaluation, i.e., please do not modify the PREPRO codes or the evaluations to remove this documentation. WARNING: It is VERY IMPORTANT that you know the PEDIGREE of the data you are using, which is EXACTLY what these PREPRO added comments are designed to do for you. This is particularly true of the allowable uncertainty used by each code, which if not correctly used may affect the results you obtain using the ENDF formatted data in your applications.**

7.3. Obtaining the Codes

These codes are available free of charge upon request from the Nuclear Data Section (see addresses on cover page) or downloaded from the Nuclear Data Section Web page

http://www-nds.iaea.org/ndspub/endf/prepro/

7.4. Your Feedback is IMPORTANT!!!

We are trying to develop a set of codes that are as computer independent as possible. In this effort your feedback is IMPORTANT!!! It is impossible for us to test these codes on all available computer/compiler combinations. Therefore your experience, on your specific computer/compiler can help us to improve the computer independence of these codes. It is also in your best interest to share your experience with us, since it will insure that future versions of these codes are as compatible as possible to meet your needs.

Please send all feedback via e. mail at,

mailto:services@iaeand.iaea.org

8. IMPLEMENTING THE CODES

8.1. What Computers do the codes run on?

The codes are designed to run on virtually any computer. The exceptions to this rule are the interactive graphics codes **complot** and **evalplot**, which are designed to produce on-screen graphics on UNIX workstations, IBM-PC, MAC (OSX), LINUX 32 and 64 bit), i.e., not mainframe computers. However, even these codes can be used in their non-interactive mode, named **comhard** and **evalhard** (note the names to indicate **hard**copy output), to produce Postscript formatted files that can be printed on any Postscript printer or viewed on many computer screens.

For use on IBM-PC running Windows (DOS) or Linux (32 or 64 bit), and on MAC (OSX), the distribution includes executables, ready to use immediately. For use on a variety of UNIX, LINUX and MAC based computers, the distribution includes a batch file for each type of computer, to compile and load all programs. For other types of computers, see the section below on, Details of Compiling and Loading Codes

8.2. The Most Up-to-Date Installation Instructions

The most up-to-date installation instructions, documentation, and the codes, can be downloaded from the website,

http://www-nds.iaea.org/ndspub/endf/prepro/

Read the text and then select “Download Codes” or “Download Documentation”

We try to maintain these installation instructions as up-to-date as possible, based on user feedback. So if you have any problems or suggestions regarding installation please e.mail them to the Nuclear Data Section at,

mailto:services@iaeand.iaea.org

8.3. Register as a User

We try to maintain these codes and data as up-to-date as possible. So if you are using any of these codes it is important that you tell us about this, so that the Nuclear Data Section can put your name on the distribution list to inform you about the latest updates. This is a FREE!!! service which is provided to users of these codes. We have tried to make this as easy as possible for you - PLEASE take a moment to e.mail to mailto:services@iaeand.iaea.org, and tell what codes you are using, and what type of computer(s) you are using - it’s as simple as that.

9. USE OF CODES

9.1. Read the Output Reports

MOST IMPORTANT! You cannot use these codes like a black box and assume that everything is perfect. Don't make the mistake of assuming that all ENDF/B data is perfect, or that these codes are perfect. It's up to you, the code user, to check and be sure that the data output by these codes is accurate and can be used in applications. If you don't, you are wasting your time, and will produce inaccurate results in your applications.

You can do this by reading the output reports produced by each code. These output reports will generally be quite small. They are intended to be used by you to quickly scan through them and look for **WARNING** or **ERROR** messages - these indicate problems with the ENDF/B data that you should check before using the data in any applications. You need not read each output report in detail; it is sufficient to merely search for the words **WARNING** or **ERROR** – these will always accompany important messages.

Checking these output reports doesn't take very much time, but failing to check them can cause you to waste an awful lot of YOUR time and can cause you headaches later, if you try to use data that a code has clearly indicated to be bad. If there are errors in the ENDF/B data, you are clearly in a "garbage in, garbage out" situation as far as the result you calculate in your subsequent applications. **Caveat Emptor!**

9.2. Standard and Variable Filenames

Currently all input files and input parameters are optional, and have built-in default values.

All of the codes have standard, built-in, filenames that they will use by default, unless input parameters explicitly define other filenames.

The default filenames have been defined to make it easy for you to remember, and to be compatible with as many operating systems as possible, e.g., DOS, that may only allow short filenames, and UNIX, that allows longer filenames. The default filenames are all of the form NAME.EXT, where NAME identifies a program name, and EXT identifies the type of file. All default filenames use **ONLY UPPER CASE** characters. The basic filenames include,

1) **???.INP** - The **INP**ut parameters for each code, where ??? is the name of the code. For example, the input parameters for **RECENT** are in a file named **RECENT.INP**. This name cannot be changed by input. Currently these input files are optional; if they are not present default values are used for all input parameters.

2) **???.LST** - The output **L**i**ST**ing from each code, where ??? is the name of the code. For example, the output listing from **RECENT** is in a file named **RECENT.LST**. This name cannot be changed by input.

3) **???.IN** - ENDF formatted data to be read (**IN**put) by each code, where ??? is the name of the code. For example, the ENDF/B data read by **RECENT** are in a file named **RECENT.IN**. This name can be changed by input.

4) **???.OUT** - ENDF formatted data written (**OUT**put) by each code, where ??? is the name of the code. For example, the ENDF/B data written by **RECENT** are in a file named **RECENT.OUT**. This name can be changed by input.

The above simple filename conventions will allow you to easily remember for each code, where the input parameters and output report are located, as well as where the ENDF/B data that is read and written by the code are located.

By input you can change the filenames of the ENDF formatted data files; data read and/or written – the exception being **ENDF2C** which uses fixed names for the ENDF input and output filenames = ENDFB.IN and ENDFB.OUT, and thus avoids the need for any input parameters.

If you input blank filenames the codes will use the default names (described above).

If you input anything else, the code will use the filenames you have defined. Variable filenames for each code can be up to 72 characters long. This allows you to specify directory structures, so that you can store your ENDF/B data in some rational way within a directory file structure.

For example, if you store all the ENDF/B-VII data files in a directory named ENDFB8, the following input filename used with **linear** will read a file named za092238 on an IBM-PC,

\ENDFB8\ORIGINAL\za092238

or on a UNIX workstation,

/ENDFB8/ORIGINAL/za092238

Warning - generally on UNIX workstations you will have to include the complete path to files. For example, the path to my files on my workstation may be /home/pd11/cullen, in which case my filename should be,

/home/pd11/cullen/ENDFB8/ORIGINAL/za092238

The ability to directly reference file structures is a very powerful facility that you should not overlook in organizing your ENDF/B data.

9.3. Brief Description (in the recommended order to use)

Endf2c	- Convert ENDF data to FORTRAN, C and C++, compatible form
Linear	- Linearize cross sections
Recent	- Reconstruct cross sections from resonance parameters
Sigma1	- Doppler broaden cross sections
Activate	- Generate activation cross sections (MF=10) from MF=3 and 9 data
Legend	- Calculate/correct angular distributions
Sixpak	- Convert double differential data (MF=6) to single differential
Spectra	- Convert model and general tabulation to linearized spectra (MF=5)
Fixup	- Correct format and cross sections, define cross sections by summation
Dictin	- Create reaction dictionary (MF=1, MT=451)
Merger	- Retrieve and/or Merge evaluated data
Groupie	- Calculate group averages and multi-band parameters
Complot	- Plot comparisons of cross sections (MF=3, 23); Comhard for hardcopy
Evalplot	- Plot evaluated data (MF=3, 4, 5, 23, 27); Evalhard for hardcopy
Mixer	- Calculate mixtures of cross sections
Virgin	- Calculated transmitted uncollided (virgin) flux and reactions
Convert	- Convert codes for computer/precision/compiler
Relabel	- Relabel and sequence programs

9.4. Detailed Description

The codes can be used to: 1) extensively check and correct evaluated data prior to using them in applications, particularly using graphics, 2) pre-process the data into a form that will make subsequent use of the data much easier, e.g., a processing code can avoid having to start from an original evaluation, and instead start processing PREPRO output where the data has been linearized, resonance contribution added, and cross sections Doppler broaden.

The normal sequence in which the codes are used is described below. **WARNING** - this is the recommended sequence of codes that you should run to produce **LEGAL** ENDF formatted data that conforms to **ALL** ENDF formats and conventions. Note in particular that if you do not run **ENDF2C** first the ENDF data may not be in FORTRAN, C and C++ compatible format, and if you do not run **FIXUP** and **DICTIN** at the end of this sequence the resulting ENDF data **WILL NOT** conform to all ENDF formats/conventions and may cause problem if you subsequently try to use the data.

1) **ENDF2C -** is designed to ensure that ALL PREPRO output in the ENDF format are completely FORTRAN, C and C++ compatible. As of today (August 2019) evaluated data even from major code centers are still not completely FORTRAN, C and C++ compatible. Therefore when I begin pre-processing any evaluation the first PREPRO code I run is ENDF2C to insure that ALL ENDF formatted output in subsequent codes are completely compatible. This is a very important step: it would be such a shame if after all of the effort invested to produce accurate results it cannot be accurately read and used by application codes.

2) **LINEAR** - Linearize cross sections. ENDF format allows evaluated cross sections to be represented as tables of data points using a number of different interpolation laws between tabulated points; to obtain accurate results it is important to interpret

the data using these interpolation laws. The interpolation laws are very useful during evaluation, but can present problems when they are used in applications. The subsequent use of the data can be greatly simplified and the accuracy of results improved by first linearizing all of the cross sections, i.e., replace the original evaluated tabulated data points and interpolation law by a new table where one can use linearly interpolation between tabulated points to within any required accuracy; the accuracy we use today is much, much smaller than the inherent uncertainly in today's ENDF evaluated data, so we do not add any additional significant uncertainly.

3) **RECENT** - Add the contribution of resonances to the cross sections. ENDF format allows cross sections to be represented as a contribution of resonance parameters and tabulated background corrections. This code will add the resonance contribution to the background cross sections in order to define the cross sections as linearly interpolable tables at 0 Kelvin (cold). Therefore, subsequent codes need only deal with tabulated, linearly interpolable, 0 Kelvin cross sections, i.e., no resonance parameters or non-linear interpolation.

4) **SIGMA1** - Doppler broaden cross sections to any temperature of interest for use in applications. As in the case of LINEAR and RECENT all cross sections read and written by this code are tabulated, linearly interpolable. All subsequent codes need not explicitly consider temperature effects and need only deal with tabulated, linearly interpolable cross sections at a given temperature.

5) **ACTIVATE** – Combine neutron interaction cross sections (MF=3) and multipliers (MF=9) to create activation cross sections (MF=10). LINEAR and GROUPIE have been updated to process multipliers (MF=9) and activation cross sections (MF=10). The sequence of codes LINEAR, ACTIVATE, and GROUPIE allow you to produce group averaged activation cross sections.

6) **LEGEND** - Convert tabulated angular distributions and Legendre coefficients to linearly interpolable angular distribution tables (MF=4, similar to what **LINEAR** does for MF=3 cross sections). Check all angular distributions and Legendre coefficients, in particular check for negative angular distributions and if found, optionally correct the distributions to make them positive. Note, negative angular distributions can lead to numerical instabilities and unreliable results if they are used in applications. **For 2019 a new summation of Legendre moments has been implemented to minimize round-off, by adding moments from highest order to lowest.**

7) **SIXPAK** - ENDF-6 format introduced double differential data (MF=6) into the ENDF/B system for the first time. If your application codes have not yet been updated to handle double differential data, you can use **SIXPAK** to obtain single differential (MF=4 and 5) approximations to double differential data. Earlier versions of SIXPAK only output results for outgoing (emitted) neutrons and photons, however currently SIXPAK will output angular distributions for discrete charged particle levels. **Recently SIXPAK was extended to also create MF/MT=9/5 output from the yields of MF/MT=6/5, which can be used as input to ACTIVATE to define activation cross sections.**

8) **SPECTRA** – Linearize and tabulate neutron emission spectra (MF=5). ENDF format allows neutron spectra to be represented as nuclear models or tables of data points using a number of different interpolation laws between tabulated points; in order to obtain accurate results it is important to interpret the data using these interpolation laws. The nuclear models and tables with interpolation laws are very useful during evaluation, but can present problems when they are used in applications. The subsequent use of the data can be greatly simplified and the accuracy of results improved by first linearizing all of the spectra, i.e., replace the original nuclear model or tabulated data points and interpolation law by a new table where one can use linearly interpolation between tabulated points to within any required accuracy.

9) **FIXUP** - Define all cross sections to be consistently exactly equal to the sum of their parts, make format corrections, and a number of other tests and corrections to the data to insure it is complete and in legal ENDF format and conventions, **BEFORE the data is actually used in applications**. It is extremely important for use in applications to guarantee that the cross sections are exactly consistent. For example, the total cross section MUST to defined as equal to the sum of its parts at all energies that appear in one or more of the contributing parts. In addition, it should be mentioned that the total will be equal to the sum of its parts at all energies (not just the energies at which the total is tabulated), **only if all the cross sections are linearly interpolable**; this illustrates the importance of the steps described above in processing data through each of these codes, e.g., use LINEAR. Note, if FIXUP's option to output all cross sections on a uniform energy grid is used, the FIXUP output is compatible for use as **NJOY** input.

10) **DICTIN** - Update the section index in MF=1, MT=451. This step need only be run if the subsequent codes that use the data refer to this index. If you are not sure this is the case, it is always best to include this step, since relative to the other codes described above this step requires very little running time. Be aware, that it least in principle the data is not in legal ENDF format/conventions unless this section index is correct, so it is strongly suggested that this step ALWAYS be used as the final code of any sequence of PREPRO codes.

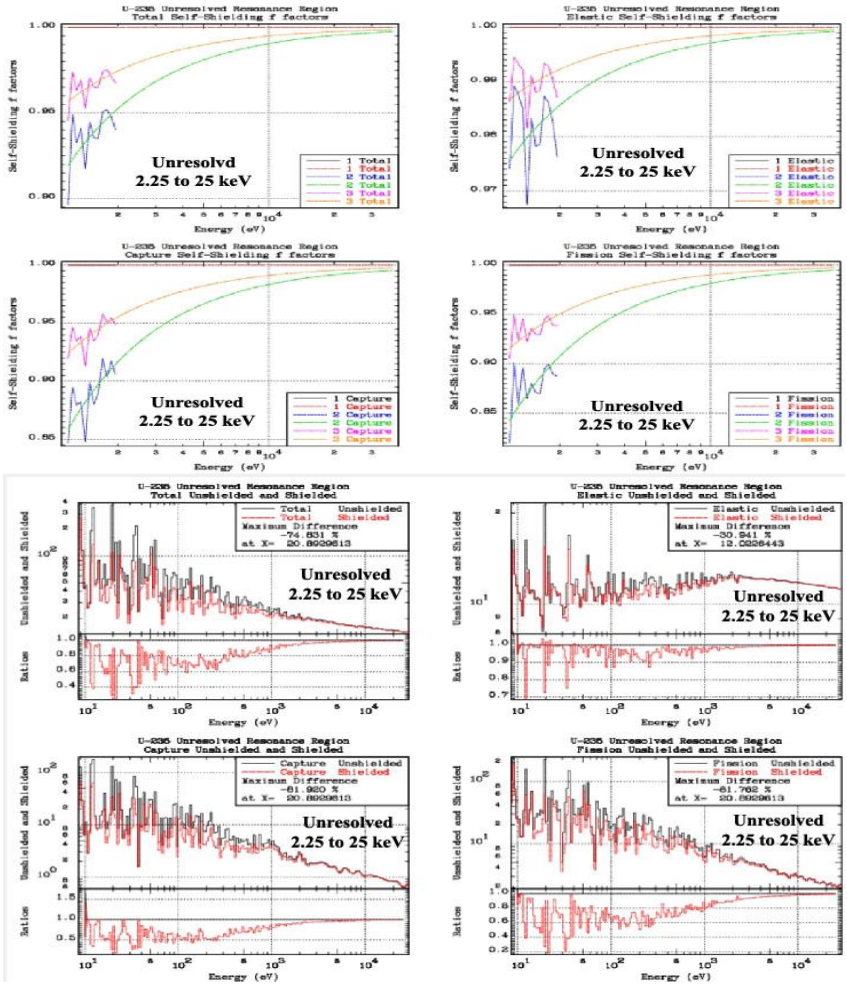
After this sequence of codes has been run the results will be evaluated data that has been carefully checked for consistency and has been reduced to a form that can be used more easily and reliably in subsequent applications.

In addition to the codes mentioned above, this PREPRO package includes a number of useful utility codes including,

1) **MERGER** - Retrieve and/or combine evaluated data. This code can be used to create a single file of data in the ENDF format from a number of different files, each of which is in the ENDF format. It can also be used to retrieve specific evaluated data from a larger ENDF/B library in order to simplify and optimize the subsequent use of the data in applications, e.g., if you have an entire ENDF/B library, but will only be using five evaluations for your applications, you can first use this code to create a mini-library containing only the five evaluations that you need for your application.

2) **GROUPIE** - Calculates Bondarenko self-shielded, multigroup cross sections and multiband parameters. This code can be used as a simple and very economical means of obtaining multigroup cross sections, in the ENDF format, which can be used in many applications where only multigroup cross sections are required, e.g., dosimetry. For comparing data using **COMPLOT** this code can be used to reduce evaluations that have many resonances, to a form in which integral differences through the resonance region can be more easily seen.

A new GROUPIE feature in 2019 is calculation of self-shielding in the unresolved resonance region; this is included in both self-shielded multigroup cross sections, and multiband parameters. Output is optionally available in listings of self-shielded data, and a new pseudo-ENDF format (used by NJOY/MCNP), as well as PLOTTAB format, so we can "see" the results. Below are PLOTTAB results for ENDF/B-VIII U-235: First, the f-factors extrapolated from the resolved to unresolved; next the unshielded and shielded cross sections.



Calculation of multigroup cross sections require a "guess" as to the shape of energy dependent flux; here referred to as the weighting spectrum. As input users may choose,

a) Input an arbitrary tabulated linearly interpolable weighting spectrum, or,
b) "flat" or constant weight in each group, or,
c) 1/E across section entire energy range, or,
d) Maxwellian to 1/E to Fission to Constant

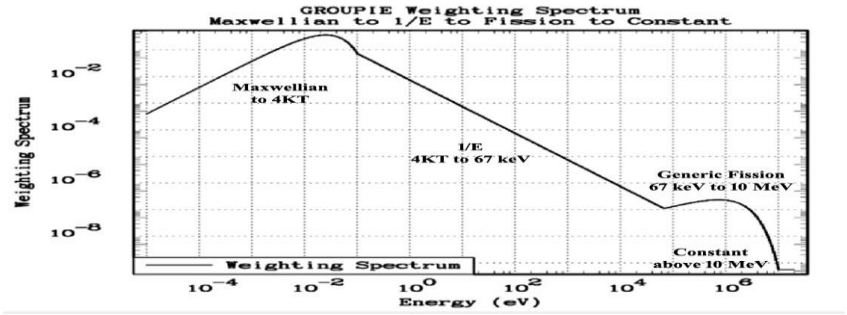
Details of d) are,
Maxwellian = 0 eV to 4KT (T = Temperature read from MT=1 of ENDF data).
1/E = 4KT to 67 keV
Fission = 67 keV to 10 MeV
Constant = 10 MeV and above
This is normalized to unity integrated over the entire energy range, with each segment normalized to make the spectrum continuous, as shown in the below figure.

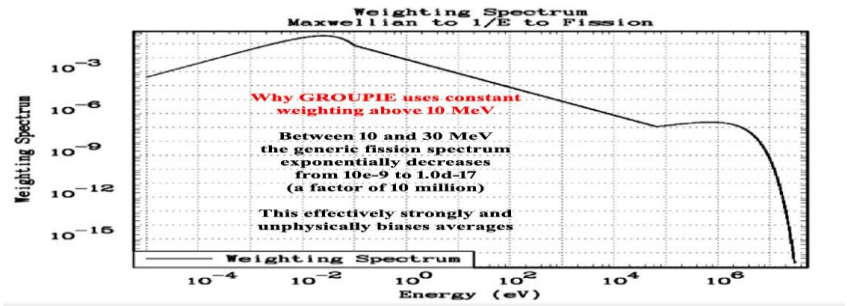
WARNING – the low energy Maxwellian does not include binding effects, and the high energy generic fission spectrum is not unique (i.e., actual fission spectra vary from one isotope to another and also vary with incident neutron energy); it is intended only as a rough estimate to weight fine groups in the keV to MeV energy range. Above 10 MeV a constant is physically a better "guess" than the fission spectra. The first figure below shows the spectrum used by GROUPIE.

If there are any neutrons above 10 MeV they are not from fission, e.g., 14.1 MeV fusion neutrons. Between 10 and 30 MeV if the generic fission spectrum were used by GROUPIE it would exponentially decrease from 10^{-10} to 10^{-17}; a factor of 10 million. In this energy range this would be a completely unrealistic "guess" for the weighting spectrum. The second figure below shows what this would look like (not to worry; GROUPIE does not use this). Hopefully this explains why the GROUPIE's weighting spectrum above 10 MeV is defined to be constant; we judge this to be the best approximation to calculate fine group constants above 10 MeV, i.e., above 10 MeV there are either few if any neutrons and averages are meaningless, or there is some higher energy neutron source, for which a constant rather than fission spectrum is a better guess.

Above 10 MeV, the difference between a fission spectrum (a bad guess), and constant that GROUPIE uses, will obviously effect averages of all data above 10 MeV, but particularly all the data above 20 MeV in newer evaluations. This difference can also have an ENORMOUS effect on wide energy average for any reaction with a high energy threshold, e.g., (n,3n); **here fission spectrum average will be many times smaller than a constant spectrum average.**

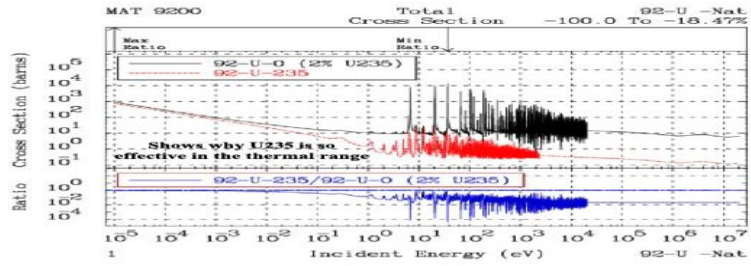
Commented [DC1]:





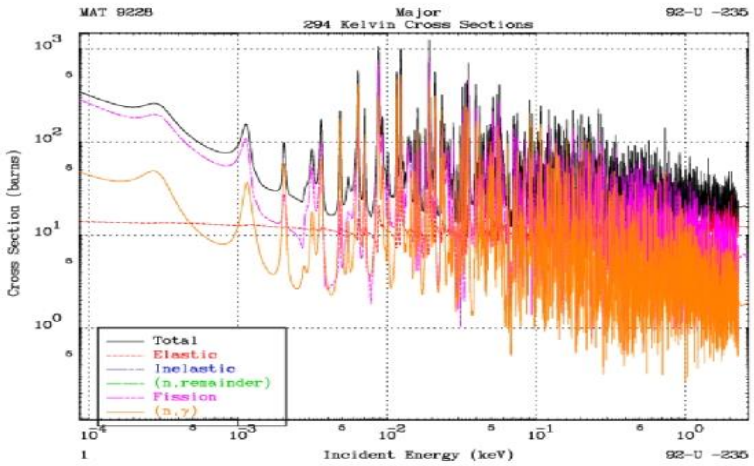
3) **COMPLOT** – Plot a comparison of cross sections from two different evaluations. This code can be used to compare cross sections, for each reaction, to define exactly how two evaluations differ. This can be extremely important if one has already used a given evaluation in applications and wishes to quickly and inexpensively determine whether a newer evaluation can be expected to produce significantly different results when used in your applications. It is also an excellent and simple means of documenting the differences between two evaluations, e.g., what's the difference between the ENDF/B-VI, Release 4 and 5, U-235 cross sections? See the above comments under **GROUPIE** for suggestions concerning comparing evaluations that have many resonances.

This code can be used as a simple means of visually checking all cross section data types and is often very useful to help understand the results obtained when data is used in applications. In addition, the graphic Postscript output can serve as a part of the documentation for evaluations, as shown below. Below the total cross section for uranium enriched to 2 % U235 as compared to 2% U235. Here we can see that at high energy the U235 is only a small fraction of the total, whereas at low energy it is most of the cross section, e.g., that's what makes thermal reactors work.



Three versions of the same code are provided: **complot** to produce on-screen graphics, as well as Postscript hardcopy output: **comhard** (each Postscript file in a different output file), and **comhard1** (all Postscript files in a single output file).

4) **EVALPLOT** - Plot cross sections (MF=3), angular distributions (MF=4), Legendre coefficients (MF=4) and/or energy distributions (MF=5), for neutron interaction data, as well as neutron induced photon production data, photon interaction, and electron interaction data. This code can be used as a simple means of visually checking all these data types and is often very useful to help understand the results obtained when data is used in applications. In addition, the graphic Postscript output can serve as a part of the documentation for evaluations, as shown below. Here we see the ENDF/B-VIII U235 cross sections in the resolved resonance region; note the high ratio of fission to capture at low energies, hence thermal reactors.



Three versions of the same code are provided: **evalplot** to produce on-screen graphics, as well as Postscript hardcopy output: **evalhard** (each Postscript file in a different output file), and **evalhard1** (all Postscript files in a single output file).

5) **MIXER** - Can be used to define the cross sections for a combination of materials, e.g., stainless steel. This code can be used in combination with **COMPLOT** to see which energy ranges are important for each material and each constituent of a material. This code can also be used to define the correct total cross section for use in transmission calculations (see, **VIRGIN**), as well as in Bondarenko self-shielding calculations (see, **GROUPIE**), to avoid the approximations normally incoherent in the Bondarenko method of self-shielding, e.g., the assumption that the resonance structure in each isotope is completely independent of that in all other isotopes. Since ENDF/B-VI, VII and VIII have moved in the direction of representing separate isotopes for each element, this code is particularly useful if your applications only require a natural mixture of isotopes, e.g., use MIXER to combine isotopes into the natural element, such as natural Fe from its isotopes.

6) **VIRGIN** - Can be used to perform exact uncollided (virgin) transmission calculations (exact, assuming the tabulated, linearly interpolable cross sections are

exact - no other approximations are used). By using the data that has been prepared by a combination of **LINEAR, RECENT, SIGMA1, MIXER,** etc., this code can be used to simulate planar transmission through any given material, or layers of different materials, at any given temperature. The results include both transmitted flux and reaction rates (as measured in self-indication measurements) vs. material thickness. The results can be obtained either on a continuous energy basis, or they can be binned (energy integrated) to simulate any given experimental resolution.

In addition, there are two utility codes that operate on the codes, rather than on ENDF/B data.

1) **RELABEL** - Is a file maintenance code used to maintain all the codes in this package. This code will normally not be used by users, unless they plan to modify the PREPRO codes. Users should be WARNING that I (D. E. Cullen) extensively use RELABEL to maintain my codes based on my very conservative use of FORTRAN, but it is not intended for general use = **CAVEAT EMPTOR!!!!**

2) **CONVERT** - Format and optimize codes for use at any given computer installation. This code is no longer required by the PREPRO, since the codes are now completely computer independent. It is still included in this package only because users have found it useful for other purposes. Generally, this code is used only once to format all the codes prior to their first use on any given computer/system.

9.5. Verifying Implementation

This distribution comes with a file named VERIFY (or verify), which is designed to run the codes, one after another, with the final two steps being to run EVALPLOT and COMPLOT, so that you can see the final results. VERIFY is a simple text file; its contents are shown below,

```
endf2c
linear
recent
sigma1
activate
legend
fixup
dictin
groupie
mixer
virgin
evalplot
complot
```

When executed as a batch file, this will run the codes in the order indicated. The distributed input parameters have been defined so that each code reads the ENDF formatted data file produced by the preceding code, and writes the ENDF formatted data file that will be read by the following code.

To verify implementation immediately after you have installed the codes, DO NOT change any input parameters for ANY codes, and execute VERIFY.BAT. It will take between 5 minutes and an hour (depending on the speed of your computer), to run all the codes. When you get to the final two graphics codes, EVALPLOT and COMPLOT, you can be assured that all the codes have run successfully.

COMPLOT will compare the cross sections calculated by you on your computer to a standard set of results distributed with PREPRO 2019. In both cases cross sections are calculated by each code to within an accuracy of 1 %. Therefore, when COMPLOT compares the results you may find differences of about 1 % (up to 2%). This difference is o.k., and merely indicates the differences due to precision to which the cross sections have been calculated. Subsequently, for use in your applications you can feel free to modify the input parameters for each code to meet the precision that you require, e.g. **I recommend you subsequently use the BEST input for each code.**

WARNING – for UNIX users - some UNIX systems now include **diction** as a system command. To avoid this conflict, in PREPRO 2019 the code previously named **diction** has been renamed **dictin**.

9.6. Use of the Codes in Combination

Almost any computer will allow you to submit a batch job, in which case you can perform any number of operations one after the other, as is done in the above verification. These computers can utilize this facility to run any number of these codes in combination, minimize the total amount of disk space used, and most important, optimize the use of YOUR time/effort.

In order to run any number of codes one after the other, all you need is the facility to: 1) start a program, 2) rename a file, 3) delete a file, if you want to minimize disk space.

For example, if I want to run the sequence of codes**, ENDF2C**, **LINEAR, RECENT, SIGMA1, ACTIVATE, LEGEND, FIXUP** and **DICTIN** and only keep the original data read by **ENDF2C** and the final results output by **DICTIN**, I can use the standard ENDF filenames for the data read and written by each code, and submit the following batch file on an IBM-PC,

```
endf2c
rename ENDFB.OUT LINEAR.IN
linear
rename LINEAR.OUT RECENT.IN
recent
delete RECENT.IN
rename RECENT.OUT SIGMA1.IN
sigma1
delete SIGMA1.IN
rename SIGMA1.OUT ACTIVTE.IN
activate
delete ACTIVATE.IN
```

```
rename ACTIVATE.OUT LEGEND.IN
legend
delete LEGEND.IN
rename LEGEND.OUT FIXUP.IN
fixup
delete FIXUP.IN
rename FIXUP.OUT DICTIN.IN
dictin
delete DICTIN.IN
```

Note, when each code finishes the above batch deck renames the ENDF formatted data **output** by the code to the filename of the ENDF formatted data **input** to the next code. When the next code finishes, the ENDF formatted data input to it is deleted (we no longer need it), and the cycle starts for the next code. More efficiently you could have defined ENDF input and output file names in the input parameter files for each code to link them together, e.g., instead of copying LINEAR.OUT to RECENT.IN, you could have defined the input file to RECENT to be named LINEAR.OUT.

The result will be the original data read by **ENDF2C** is still in the file named **ENDFB.IN**, and the result is in the file named **DICTIN.OUT**. All other intermediate files have been deleted.

On any other system, such as UNIX, the names **delete** and **rename** may be different, but the basic idea remains the same.

An alternative to the above approach is to use the facility of the codes to read and write files from any file structure. For example, assume I have a directory named ENDFB7, and within this directory I have three sub-directories: ORIGINAL, TMP, and K300 (data Doppler broadened to 300 Kelvin). What I can do is first copy a file from ENDFB7/ORIGINAL to ENDFB.IN, the standard **ENDF2C** ENDF input data file (ENDF2C is the only PREPRO code that uses fixed ENDF input and output filenames), define input parameters to **LINEAR, RECENT, SIGMA1, ACTIVATE, LEGEND** and **FIXUP** to produce ENDF output in ENDFB7/TMP, and have each code read the output from the preceding code. Finally, I can define **DICTIN** input parameters to write the ENDF output into ENDFB7/K300, with its final filename. In this case if I do not worry about deleting the intermediate files, the batch input need only be the names of the codes to run, i.e.,

```
endf2c
linear
recent
sigma1
activate
legend
fixup
dictin
```

Using a batch approach can save you a great deal of YOUR precious time/effort. You don't have to sit there and babysit your terminal in order to start each code until the preceding one finishes. You can use batch jobs to combine code executions, and go off to work (or play) until the sequence of codes finishes. If you then want to be

sure that everything ran correctly, you can read the output reports from each code, i.e., see the **???.LST** from each code, e.g., for RECENT see RECENT.LST.

It is HIGHLY Recommended that you always read these OUTPUT REPORT files (???.LST). Never make the mistake of assuming the ENDF data or the PREPRO are "perfect"; always check the report from each code (???.LST) for WARNINGS and ERRORS This minor investment of your time can save you a great deal of time and effort if you "misuse" data that has clearly been documented in the ???.LST files to be deficient or in ERROR.

10. DETAILS OF COMPILING AND LOADING CODES

For use on IBM-PC running Windows or Linux (32 or 64 bit), and on MAC (OSX), the distribution includes executables, ready to immediately use. For use on a variety of UNIX, LINUX and MAC based computers, the distribution includes a batch file for each type of computer, to compile and load all programs, and to then clean up by deleting everything not required to execute the programs. Only for other types of computers need you be concerned with the details concerning compiling and loading the codes, which are described here.

10.1. Parts of the Codes

The codes have now been divided into several parts that should be combined when compiling and loading; see, example compile/load instructions below. The parts are,

1) The basic code

2) Include files to define code storage

3) Routines to allow all codes to now uniformly treat all ENDF formatted input and output (**endfio.f**)

4) Routines to allow scratch files to be defined either with or without file names,
scratcha.f = with file name
scratchb.f = without file name

Most compilers/computers allow scratch files to be defined without scratch file names, so use either **scratcha.f** or **scratchb.f**. However, some compilers/system combinations get confused when there are multiple scratch files without file names, e.g., Lahey on IBM-PC (use **scratcha.f**), and some compilers do not allow scratch files with file names, e.g., ABSOFT on IBM-PC and MAC (OSX) (use **scratchb.f**).

4) A timer, to define the execution time for each code. The standard timer routine (**timer.f**) distributed with the codes uses the standard Windows, LINUX and MAC routine **ETIME**; on some computers you will have to consult the on-line manual to see how to link to **ETIME**, e.g., HP.

If you are not using a UNIX based computer, you will have to supply your own timing routine. It is recommended that you use the distributed version of **timer.f**, and add a function **ETIME**, that defines the execution time on your computer - see, the timing routines included for a variety of UNIX computers

If you do define a non-standard timer, try to define EXECUTION - NOT WALL CLOCK time - on some computers this isn't possible, e.g., IBM-PC running DOS - in which case use whatever you can.

If you can't figure out how to define running time, or you don't want the codes to print running time, instead of using the distributed **timer.f**, define and use the following dummy routine,

```
SUBROUTINE TIMER
RETURN
END
```

If you do define a non-standard timer, PLEASE send us a copy, identifying what computer/compiler you are using - over a period of time we intend to build up a library of timer routines for as many different computers as possible - which we will then distribute with the codes = future versions will be more compatible to meet YOUR needs.

5) A graphics interface, for complot and evalplot.

10.2. Compiling/Loading

This section applies to all the codes, except the graphics codes, **complot** and **evalplot**; see, below under graphics codes. Below is an example of how to compile/load the codes on a UNIX based computer. For this example, I illustrate how to create sixteen (16) executables on a SUN workstation; timing routines are provided for most types of computers. Note,

1) No special libraries are used by these codes, so that compile/load instructions are very simple.
2) How the pieces are combined.
3) Use the OPTIMIZATION available on your computer - this can make a BIG difference in running time, but please verify results
4) sun.f is the timing routine to use on a SUN workstation. Similar timing routines are provided for most types of computers.

```
f77 -o endf2c    -O endf2c.f    endfio.f scratchb.f timer.f sun.f
f77 -o activate  -O activate.f  endfio.f scratchb.f timer.f sun.f
f77 -o linear    -O linear.f    endfio.f scratchb.f timer.f sun.f
f77 -o recent    -O recent.f    endfio.f scratchb.f timer.f sun.f
f77 -o sigma1    -O sigma1.f    endfio.f scratchb.f timer.f sun.f
f77 -o fixup     -O fixup.f     endfio.f scratchb.f timer.f sun.f
f77 -o spectra   -O spectra.f   endfio.f scratchb.f timer.f sun.f
f77 -o legend    -O legend.f    endfio.f scratchb.f timer.f sun.f
f77 -o sixpak    -O sixpak.f    endfio.f scratchb.f timer.f sun.f
f77 -o mixer     -O mixer.f     endfio.f scratchb.f timer.f sun.f
f77 -o merger    -O merger.f    endfio.f scratchb.f timer.f sun.f
f77 -o dictin    -O dictin.f    endfio.f scratchb.f timer.f sun.f
f77 -o virgin    -O virgin.f    endfio.f scratchb.f timer.f sun.f
f77 -o groupie   -O groupie.f   endfio.f scratchb.f timer.f sun.f
f77 -o relabel   -O relabel.f                      timer.f sun.f
f77 -o convert   -O convert.f                      timer.f sun.f
```

10.3. Graphics Codes

The graphics codes - **complot** and **evalplot** - can be used to produce either,

1) Postscript output files for printed hardcopy, using executables named **comhard** and **comhard1** and **evalhard** and **evalhard1**. For 2019, 2 variations are included: **comhard** create a separate file for each plot and **comhard1** create a single file containing all of the plots. Similarly for **evalhard** and **evalhard1**.

2) On screen graphics, using executables named **complot** and **evalplot**.

The 3 executables, **complot**, **comhard** and **comhard1**, are exactly the same code, loaded with different graphics interfaces; all executables use the same input and output files, **COMPLOT.INP** and **COMPLOT.LST**. Similarly, the 3 executables, **evalplot**, **evalhard and evalhard1**, are exactly the same code, loaded with different graphics interfaces; both executables use the same input and output files, **EVALPLOT.INP** and **EVALPLOT.LST**.

10.4. Postscript Output Files

The Postscript graphics interface should be completely computer independent, and as such should run on any computer.

It will create a series of output files - none of which are sent to your printer during execution of the code.

Output for each plot is saved on disk, so when the code ends all the plot files will still be on disk, and you can then view them (use a Postscript file viewer), send them to your printer, and/or, save them for later use.

WARNING - the codes always use the same file names, **PLOT0001.ps, PLOT0002.ps**, etc. So that repeatedly running a code will overwrite any files that you previously created. If you want to save files, moved them or rename them before running a code again.

To use this method to create these Postscript files use **saveps.f** or **saveps1.f** with the codes.

For Postscript graphics, no special libraries are used, and an example of how to compile/load the codes on a UNIX based computer is shown below - this is very similar to the compile instructions shown above, with the addition of saveps.f,

```
f77 -o comhard  -O complot.f  endfio.f scratchb.f timer.f saveps.f sun.f
f77 -o evalhard -O evalplot.f endfio.f scratchb.f timer.f saveps.f sun.f
```

Note, that here the executables are given the names for the hardcopy versions of the codes, **comhard** and **evalhard**.

10.5. On Screen Graphics

For on screen graphics the codes are loaded with **screen.f**, in contrast to the hardcopy version of the codes, described above, for Postscript graphics that are loaded with **saveps.f**.

Example Makefiles are included for a variety of UNIX, LINUX and MAC systems.

On screen graphics is VERY computer dependent, so on UNIX computers you may have to modify the UNIX Makefile - this should only involve finding out where the X11 graphics library is stored on your computer and setting the correct path in the Makefile.

If you do have to modify the Makefile, please send me a copy of the modified file, identifying your computer/compiler, so that we can build up a library of Makefiles to be distributed with the codes; this will make future versions as compatible as possible with your needs.

The codes are distributed with graphics interfaces for,

1) UNIX, LINUX, MAC (OSX), and openVMS systems, using the X11 graphics library (**screen.f**, **nodash.c, dash.c**)

2) If you are using any other system, you will have to supply your own graphics interface - see, **screen.f** for a description of the simple interface used by these codes.

10.6. Interacting with Graphics

When you are using **evalplot** there is no true on-screen interaction with the plots. If you wish to view different data over different energy ranges your only option is to change your input parameters in the file **EVALPLOT.INP**.

When you are using **complot** you can interact with the on-screen plots. Once a plot has completed displaying on your screen if you would like to see a portion of the energy range of the plot in greater detail, you can do this by using your mouse to zoom in by indicating the lower and upper energy limits of the energy range you would like to see. As soon as you select the energy range, the next zoomed plot will appear on your screen, with the same data as on the previous plot, but only over the energy range that you have selected. WARNING – **complot** only generates plots when the two evaluations differ by more than the allowable uncertainty you define by input in the file **COMPLOT.INP**. This also applies when you interact with the plots. Therefore, if you use your mouse to select an energy range over which the two evaluations do not differ by more than your allowable uncertainty a zoomed plot will not be produced, but the results of the comparison will be reported in the output file COMPLOT.LST, and **complot** will proceed to its next comparison.

11. COMMENTS FROM CODES

These codes are designed to be self-documenting, in the sense that the most up-to-date documentation is included as comments at the beginning of each code. Periodically documentation, such as this report, is published. But the user is warned that the comments in the codes are continuously updated and it is these comments within the codes that should be considered to be the most up-to-date documentation, and the user should read these comments before, and while, using these codes.

The following section contains a listing of the comments from the codes as of the publication date of this report (comments for each code are now included separately as part of the PREPRO system, in Word, PDF, and TEXT formats). The comments are listed for each code alphabetically according to the name of the code, including,

ACTIVATE
CONVERT
COMPLOT
DICTIN
ENDF2C
EVALPLOT
FIXUP
GROUPIE
LEGEND
LINEAR
MERGER
MIXER
RECENT
RELABEL
SIGMA1
SIXPAK
SPECTRA
VIRGIN