X4Pro: introduction and overview

Viktor Zerkin

International Atomic Energy Agency, Nuclear Data Section

EXFOR Workshop on the Compilation of Experimental Nuclear Reaction Data
13-16 December 2022, IAEA Headquarters, Vienna, Austria

Part I.

EXFOR systems.

EXFOR Relational database.

EXFOR today

Format. EXFOR - EXchange FORmat for compiling and exchange experimental reaction data between members of Nuclear Reaction Data Centers network (NRDC, 1970-2022). Format is always under development; changes are fixed on annual NRDC meetings.

Library. EXFOR library includes EXFOR Master file (25,489 Entries, 1966-2022), EXFOR-CINDA Dictionaries (41), Manuals: EXFOR Basics, Formats and Lexfor.

The network strictly regulates data format, rules of compilation, distribution of work and data exchange between data centers according to a special Protocol.

NRDC

Distribution of data is mainly the responsibility of nuclear data centers and is not directly regulated by the NRDC.

Database(s). Based on EXFOR library databases implemented in specific computer environment, starting from CODASYL-DBMS (VMS, NNDC/BNL, 80-90s) and later: MySQL/MS-Access/SQLite (IAEA-NDS, NNDC), H2 (NEA-DB) and others

Web system(s). Web interface to EXFOR database: several systems; IAEA-NDS EXFOR-ENDF Web system: https://www-nds.iaea.org/exfor

Off-line system(s) with GUI and command line interface. IAEA-NDS packages: “EXFOR for Applications”, “EXFOR-CINDA for Windows”, “X4Apps/SQLite”; NEA-DB: JANIS

Computational plain file(s). IAEA-NDS: XC4 for Model codes and other Apps (since 2007)

Users | Center

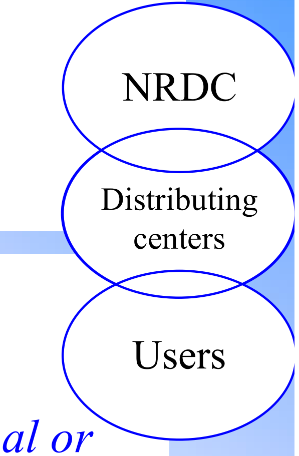
EXFOR: tasks and problems

International Network of Nuclear Reaction Data Centres (NRDC)

The primary goal of the Network is the dissemination of nuclear reaction data and associated documentation to users.

EXFOR format and library are constructed to have structure similar to original publications in order to:

a) simplify compilation process, be human readable,
b) reduce number of mistakes in compilation,
c) simplify process of cross-checking done by other centers before official release.



Distributing centers, users' community

Problem-1. EXFOR storage-search-reading/filtering/sorting

Storage of EXFOR data can be organized using database management systems, e.g. relational or NoSQL databases, or even under directory/file structures. Decisions in system design are driven by understanding of tasks, strategic plans and IT trends. Note: regular data updates are needed to be in sync with official EXFOR.

Problem-2. EXFOR parser-converter

Structure of EXFOR Entry follows the logic of original article – to be simple for compiling, but not necessarily simple for programming. After parsing, in order to be comparable, data should be converted into a universal form and into the same conditions (units, Lab-CM, etc.), which is not always trivial. So, writing a comprehensive parser-converter is complicated task. Additional problem - programming language. There are EXFOR parser-converters in Fortran, C/C++, Java, Python. New language – new parser, and all work from scratch.

Problem-3. Delivering data

Depending on EXFOR system, data are usually stored in one specific way, but delivered to user and/or application in various ways (Web/offline/GUI/API/command line interface), formats (EXFOR original and interpreted, JSON, XML) with different options and with different hierarchy. Fixing output data structure/ hierarchy (e.g. sorting order in a dataset) can make some output formats very inconvenient for other data users and applications.

Database technologies

Database. /Wikipedia/

« In computing, a database is an organized collection of data stored and accessed electronically. Small databases can be stored on a file system, while large databases are hosted on computer clusters or cloud storage. The design of databases spans formal techniques and practical considerations, including data modeling, efficient data representation and storage, query languages, security and privacy of sensitive data, and distributed computing issues, including supporting concurrent access and fault tolerance.

A database management system (DBMS) is the software that interacts with end users, applications, and the database itself to capture and analyze the data. The DBMS software additionally encompasses the core facilities provided to administer the database. The sum total of the database, the DBMS and the associated applications can be referred to as a database system. Often the term "database" is also used loosely to refer to any of the DBMS, the database system or an application associated with the database.

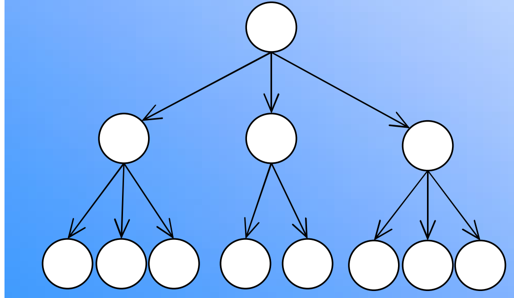
Computer scientists may classify database management systems according to the database models that they support. Relational databases became dominant in the 1980s. These model data as rows and columns in a series of tables, and the vast majority use SQL for writing and querying data. In the 2000s, non-relational databases became popular, collectively referred to as NoSQL, because they use different query languages.

The subsequent development of database technology can be divided into three eras based on data model or structure: navigational, SQL/relational, and post-relational.

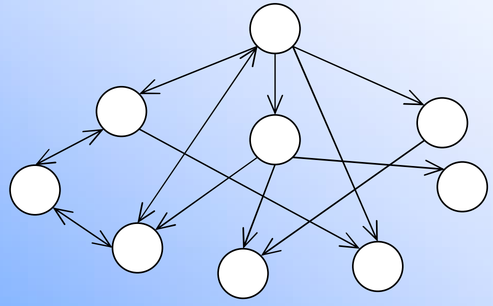
The two main early navigational data models were the hierarchical model and the CODASYL model (network model). These were characterized by the use of pointers (often physical disk addresses) to follow relationships from one record to another.

The relational model, first proposed in 1970 by Edgar F. Codd, departed from this tradition by insisting that applications should search for data by content, rather than by following links. The relational model employs sets of ledger-style tables, each used for a different type of entity. Only in the mid-1980s did computing hardware become powerful enough to allow the wide deployment of relational systems. »

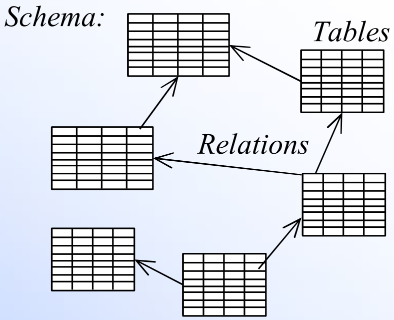
Hierarchical model



Network model



Relational model (SQL)



NoSQL

Types:

- document databases
- key-value databases
- wide-column databases
- graph databases

Features:

- Flexible schemas (no schema)
- Horizontal scaling
- Fast queries due to the data model (?)
- Ease of use for developers (?)

Relational databases

SQL (Structured Query Language) is a declarative programming language used to create, modify and manage data in a relational database managed by a database management system. SQL consists of a collection of operators, statements and calculated functions.

Types of SQL statements:

- *Data Definition Language (DDL):*
  - CREATE *creates a database object (database, table, view, user, and so on),*
  - ALTER *changes the object,*
  - DROP *deletes the object;*
- *Data Manipulation Language (DML):*
  - SELECT *selects data that meets the specified conditions*
  - INSERT *adds new data,*
  - UPDATE *changes existing data,*
  - DELETE *deletes data;*
- *Data Control Language (DCL):*
  - GRANT *grants the user (group) permissions for certain operations with the object,*
  - REVOKE *revokes previously issued permits,*
  - DENY *sets a ban that has priority over resolution;*
- *Transaction Control Language (TCL):*
  - COMMIT *applies the transaction,*
  - ROLLBACK *rolls back all changes made in the context of the current transaction,*
  - SAVEPOINT *divides the transaction into smaller sections.*

```
create table ENTRY (
    EntryID          integer NOT NULL,
    Entry            char(5),
    origEntry        char(5) null,
    Area             char(1),
    expArea          char(1),
    CenterID         smallint    null,
    DateDebut        date,
    UpdateNo         smallint,
    TransID          char(5)     null,
    TransDate        char(8)     null,
    TransFile        varchar(20) null,
    nInstitutes      smallint    null,
    Institute1       char(7)     null,
    nAuthors         smallint    null,
    Author1Ini       varchar(55) null,
    Author1          varchar(55) null,
    nReferences      smallint    null,
    Reference1       varchar(55) null,
    Ref1             varchar(32) null,
    YearRef1         smallint    null,
    Publication1     varchar(55) null,
    stdFileName      varchar(40) null,
    TypeRef1         char    (1) null,
    NsrKeyNo         varchar(8)  null,
    DOI              varchar(40) null,
    CompilerID       varchar(40) null,
    PRIMARY KEY      (EntryID)
)
```

```
alter table AUTHORS add column FullName varchar (80) null;
create index AUTHORS_FullName on AUTHORS(FullName);
update AUTHORS set FullName=trim(concat(trim(AuthorIni),' ',trim(Author)));
```

```
SELECT * FROM ENTRY where Author1='Korzh';
SELECT DISTINCT Reference1,nAuthors FROM ENTRY where Author1='Korzh' and YearRef1>=1977 order by nAuthors;
```

```
select distinct SUBENT.Entry,SUBENT.DateCompil,ENTRY.Reference1,KEYWORD.FreeText as Title
 from SUBENT inner join ENTRY on ENTRY.EntryID=SUBENT.EntryID
 inner join KEYWORD on KEYWORD.EntryID=SUBENT.EntryID
 where SUBENT.SubAcc like '%001' and ((SUBENT.Entry like '2%') or (SUBENT.Entry like 'O%'))
 and KEYWORD.KeyWord='TITLE' and binary upper(KEYWORD.FreeText)<>KEYWORD.FreeText
 order by SUBENT.Entry
```

MySQL.
Search Entries with UPPER case Title from Area 2 compiled between 1990 and 1999

SQLite. Remove **binary**

Project “EXFOR Relational”

1. Planned features of the system (2000)

1. *All information in EXFOR should be available for search in any order (direct access)*
2. *Execution time of typical request should be within 2-3 sec*
3. *The system should be really platform independent (simplest: no stored procedures, no foreign keys, etc.)*
4. *The system should guarantee integrity of original data*
   - *usage of BLOBs to store SUBENT in their original form*
   - *convincing other centers to switch to central database maintenance*
5. *Whole system (database and programs) should fit to CD-ROM=640Mb (storage of zipped BLOBs)*
6. *The database should be easy deployed to mirror-sites (MyISAM, MDB) without maintenance system*
7. *Extendable set of tables and columns in the tables*
8. *System should allow usage of programs on several languages (including legacy codes) and extensions*
9. *Modularity and robustness of software, re-use of modules*
10. *Interactive multiplatform plotting*

2. Allowed to achieve

1. *Merging EXFOR libraries to common "EXFOR Master file” (2002-2005)*
2. *Global EXFOR maintenance system in the IAEA-NDS (since 2005): TRANS files and fixed Master file for every update*
3. *Optimising of efforts in NRDC*
4. *Common (robust) EXFOR Web retrieval system: IAEA-NDS, NNDC (USA), India, China, Russia*
5. *Integrating with EXFOR compilation control system*

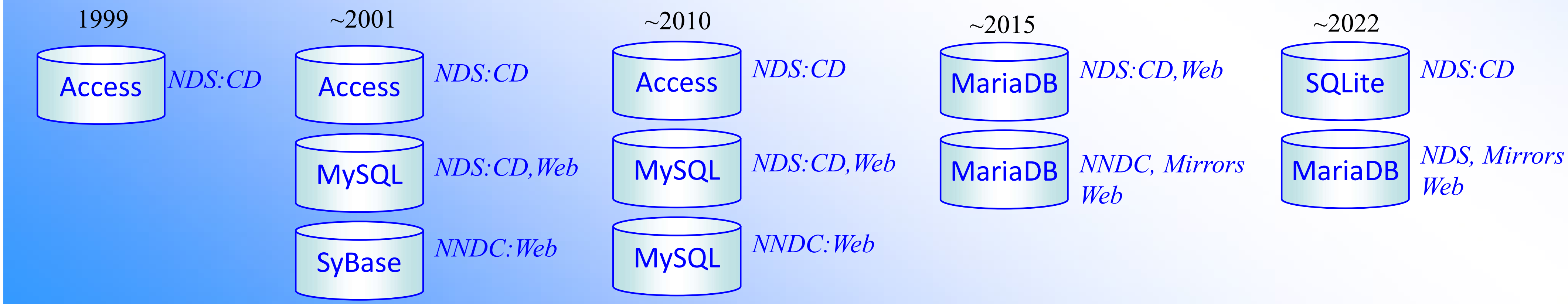
3. Not foreseeing extensions (2007-2022)

1. *EXFOR-NSR PDF database (with authorised Web access)*
2. *Connection and import from NSR*
3. *Export to R33 (IBANDL)*
4. *EXFOR data re-normalization/corection system*
5. *Construction covariance matrices using uncertainties*
6. *Uploading system for remote data checking and processing (for EXFOR compilers)*
7. *Web system without Internet*
8. *X4Lite (EXFOR-Relational on SQLite) and X4Pro*

Current status of EXFOR-Relational

1. *Relational EXFOR database: common between NDS-NNDC*
    - a) *schema based on "EXFOR-Access CD-ROM": discussed and initially agreed between NDS, NNDC, CNPD on NNDC-2000 Workshop "Nuclear database: migration to relational database and Java technology"*
    - b) *existing and maintained at NDS and NNDC from 2000 to 2021:*
    - c) *OS: Windows, Linux, MacOS*
    - d) *DBMS: MS-Access (2000), MySQL (2001), SyBase (2005), SQLite (2020)*
    - e) *Web: NDS, NNDC, 3 Mirrors (India, China, Russia)*
    - f) *deployed to Mirror-sites and on CD-ROM to individual users*
2. *EXFOR-CINDA Web Retrieval system:*
   official NRDC Web retrieval system since 2008
3. *Current versions of EXFOR output to C4, C5, JSON, XML:*
    - a) *easier to use in users' applications than EXFOR*
    - b) *have fixed format, require converter*

History of EXFOR-Relational



1999	~2001	~2010	~2015	~2022
Access *NDS:CD*	Access *NDS:CD*	Access *NDS:CD*	MariaDB *NDS:CD,Web*	SQLite *NDS:CD*
	MySQL *NDS:CD,Web*	MySQL *NDS:CD,Web*	MariaDB *NNDC, Mirrors Web*	MariaDB *NDS, Mirrors Web*
	SyBase *NNDC:Web*	MySQL *NNDC:Web*		

EXFOR relational database: structure and content (IAEA-NDS, NNDC, 2000-2022)

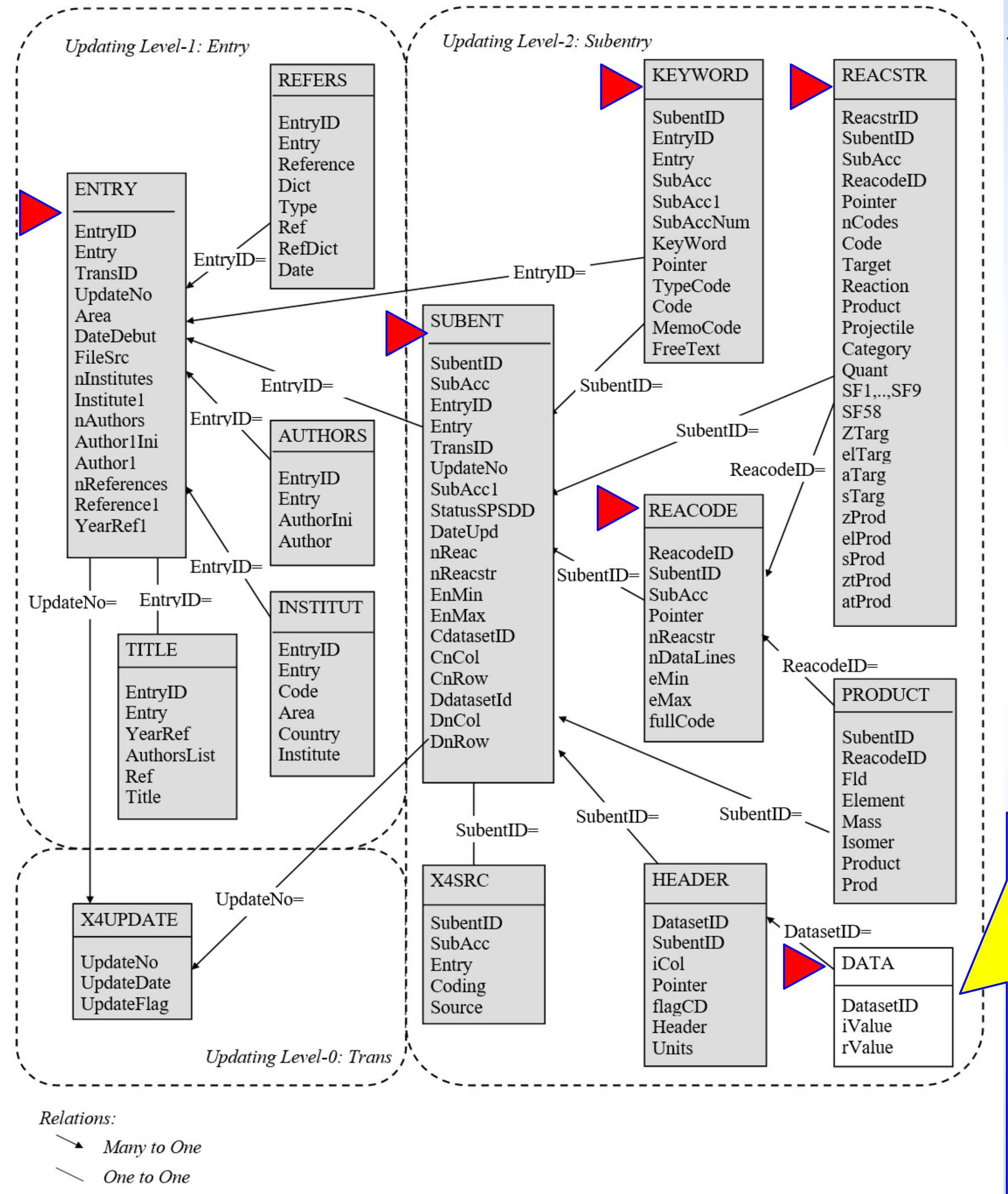


Fig.1. EXFOR Relational: Schema (November-2003)

Initial database: EXFOR + Dictionaries

Database extensions:

Corrections

Created: 2010
Records: 17,23
Size: 11.6 Mb

Automatic and experts' corrections. Available online via C4, TAB, Plots.

X4-NSR PDF

Created: 2012
Records: 224,632
Size: 192 Gb

PDF files of publications – source of EXFOR and NSR. Full contents available online for authorized users.

Test search

Created: 2014
Records: 1,514,453
Size: 198 Mb

Google-like search in interpreted EXFOR, incl. free text, keywords, codes and their interpretation from dictionaries.

EXFOR Archive

Created: 2014
Entries: 103,4317
Subent: 813,341
Size: 0.9 Gb

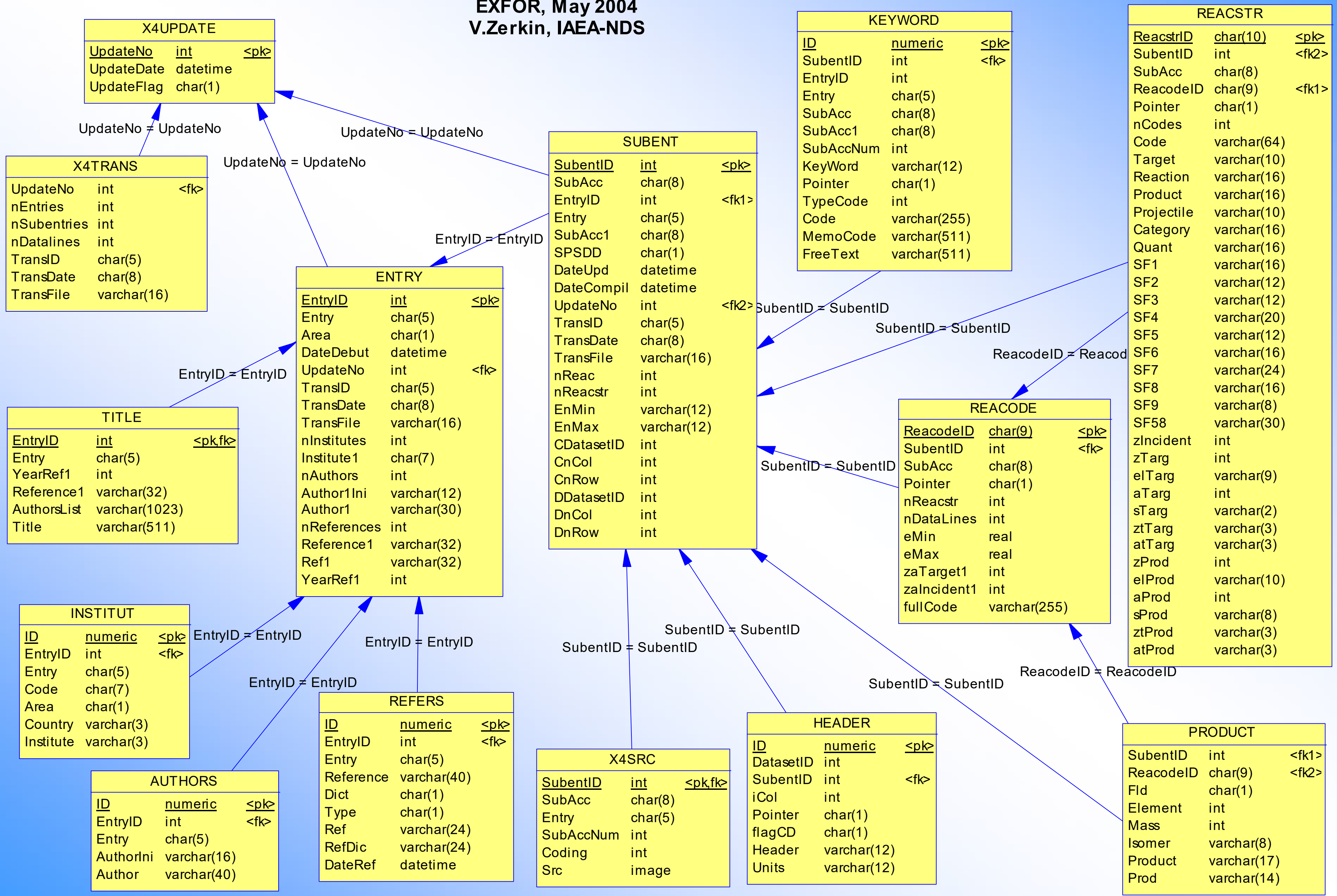
Contains current and all previous versions of every SUBENT. Available online.

Note. The table DATA was not implemented, because there was no effective solution to store and operate with such a huge variety of data as presented in EXFOR. Situation has been changed when popular relational database management systems integrated JSON as column type to their databases providing functionality for storing, presenting, indexing, querying.

This is key point of X4Pro approach.

EXFOR database schema as of 2004, SyBase

EXFOR, May 2004
V.Zerkin, IAEA-NDS



X4UPDATE

UpdateNo	int	<pk>
UpdateDate	datetime	
UpdateFlag	char(1)	

X4TRANS

UpdateNo	int	<fk>
nEntries	int	
nSubentries	int	
nDatalines	int	
TransID	char(5)	
TransDate	char(8)	
TransFile	varchar(16)	

ENTRY

EntryID	int	<pk>
Entry	char(5)	
Area	char(1)	
DateDebut	datetime	
UpdateNo	int	<fk>
TransID	char(5)	
TransDate	char(8)	
TransFile	varchar(16)	
nInstitutes	int	
Institute1	char(7)	
nAuthors	int	
Author1Ini	varchar(12)	
Author1	varchar(30)	
nReferences	int	
Reference1	varchar(32)	
Ref1	varchar(32)	
YearRef1	int	

TITLE

EntryID	int	<pk,fk>
Entry	char(5)	
YearRef1	int	
Reference1	varchar(32)	
AuthorsList	varchar(1023)	
Title	varchar(511)	

INSTITUT

ID	numeric	<pk>
EntryID	int	<fk>
Entry	char(5)	
Code	char(7)	
Area	char(1)	
Country	varchar(3)	
Institute	varchar(3)	

AUTHORS

ID	numeric	<pk>
EntryID	int	<fk>
Entry	char(5)	
AuthorIni	varchar(16)	
Author	varchar(40)	

REFERS

ID	numeric	<pk>
EntryID	int	<fk>
Entry	char(5)	
Reference	varchar(40)	
Dict	char(1)	
Type	char(1)	
Ref	varchar(24)	
RefDic	varchar(24)	
DateRef	datetime	

SUBENT

SubentID	int	<pk>
SubAcc	char(8)	
EntryID	int	<fk1>
Entry	char(5)	
SubAcc1	char(8)	
SPSDD	char(1)	
DateUpd	datetime	
DateCompil	datetime	
UpdateNo	int	<fk2>
TransID	char(5)	
TransDate	char(8)	
TransFile	varchar(16)	
nReac	int	
nReacstr	int	
EnMin	varchar(12)	
EnMax	varchar(12)	
CDatasetID	int	
CnCol	int	
CnRow	int	
DDatasetID	int	
DnCol	int	
DnRow	int	

X4SRC

SubentID	int	<pk,fk>
SubAcc	char(8)	
Entry	char(5)	
SubAccNum	int	
Coding	int	
Src	image	

HEADER

ID	numeric	<pk>
DatasetID	int	
SubentID	int	<fk>
iCol	int	
Pointer	char(1)	
flagCD	char(1)	
Header	varchar(12)	
Units	varchar(12)	

KEYWORD

ID	numeric	<pk>
SubentID	int	<fk>
EntryID	int	
Entry	char(5)	
SubAcc	char(8)	
SubAcc1	char(8)	
SubAccNum	int	
KeyWord	varchar(12)	
Pointer	char(1)	
TypeCode	int	
Code	varchar(255)	
MemoCode	varchar(511)	
FreeText	varchar(511)	

REACODE

ReacodeID	char(9)	<pk>
SubentID	int	<fk>
SubAcc	char(8)	
Pointer	char(1)	
nReacstr	int	
nDataLines	int	
eMin	real	
eMax	real	
zaTarget1	int	
zaIncident1	int	
fullCode	varchar(255)	

REACSTR

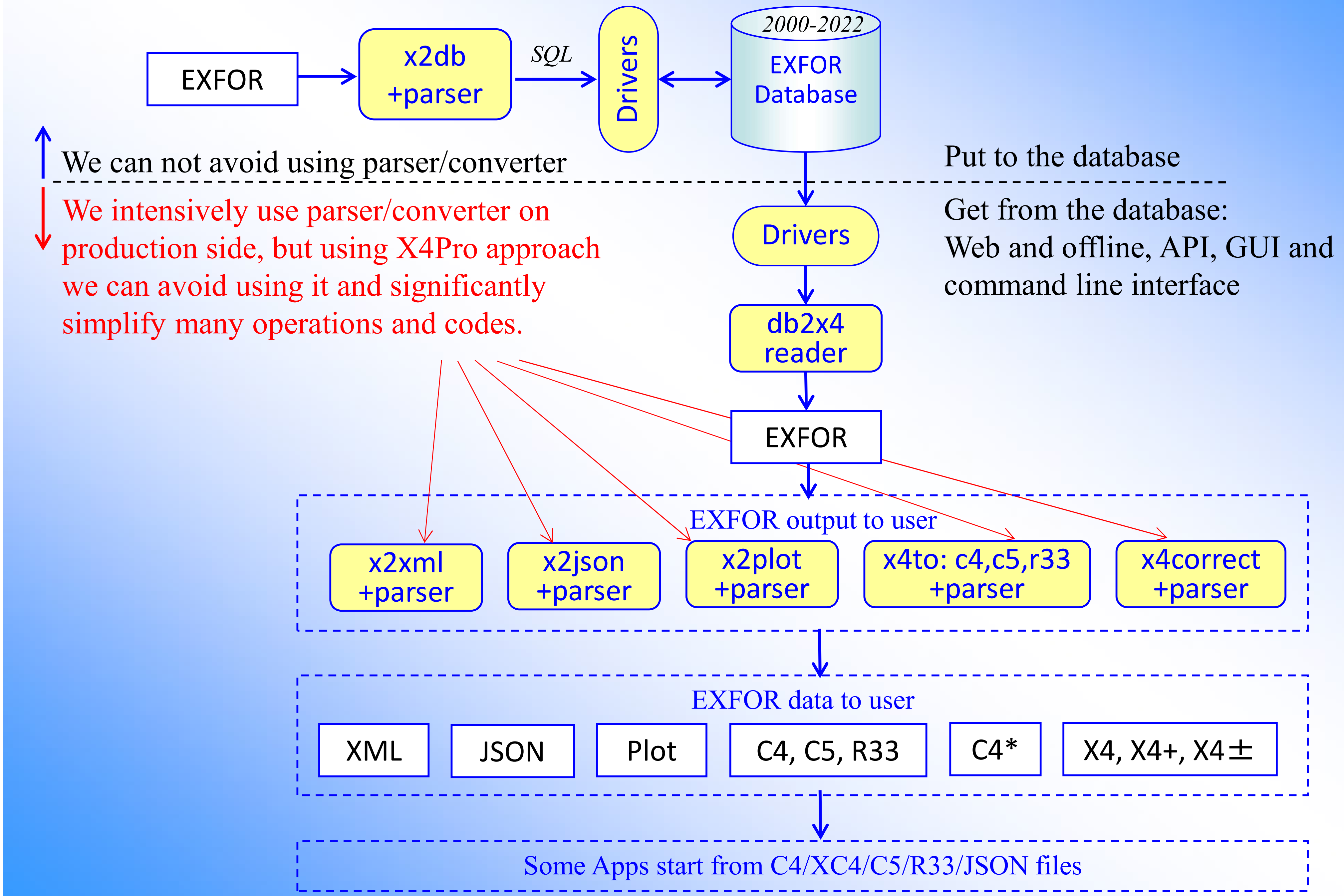
ReacstrID	char(10)	<pk>
SubentID	int	<fk2>
SubAcc	char(8)	
ReacodeID	char(9)	<fk1>
Pointer	char(1)	
nCodes	int	
Code	varchar(64)	
Target	varchar(10)	
Reaction	varchar(16)	
Product	varchar(16)	
Projectile	varchar(10)	
Category	varchar(16)	
Quant	varchar(16)	
SF1	varchar(16)	
SF2	varchar(12)	
SF3	varchar(12)	
SF4	varchar(20)	
SF5	varchar(12)	
SF6	varchar(16)	
SF7	varchar(24)	
SF8	varchar(16)	
SF9	varchar(8)	
SF58	varchar(30)	
zIncident	int	
zTarg	int	
elTarg	varchar(9)	
aTarg	int	
sTarg	varchar(2)	
ztTarg	varchar(3)	
atTarg	varchar(3)	
zProd	int	
elProd	varchar(10)	
aProd	int	
sProd	varchar(8)	
ztProd	varchar(3)	
atProd	varchar(3)	

PRODUCT

SubentID	int	<fk1>
ReacodeID	char(9)	<fk2>
Fld	char(1)	
Element	int	
Mass	int	
Isomer	varchar(8)	
Product	varchar(17)	
Prod	varchar(14)	

EXFOR relational system

The system is functioning for public at the IAEA-NDS and NNDC since 2004



EXFOR

x2db +parser

SQL

Drivers

2000-2022

EXFOR Database

We can not avoid using parser/converter

Put to the database

We intensively use parser/converter on production side, but using X4Pro approach we can avoid using it and significantly simplify many operations and codes.

Drivers

Get from the database:
Web and offline, API, GUI and command line interface

db2x4 reader

EXFOR

EXFOR output to user

x2xml +parser

x2json +parser

x2plot +parser

x4to: c4,c5,r33 +parser

x4correct +parser

EXFOR data to user

XML

JSON

Plot

C4, C5, R33

C4*

X4, X4+, X4±

Some Apps start from C4/XC4/C5/R33/JSON files

Part II.

X4Pro database: concept and implementation

X4Pro concept

What is wrong now?

EXFOR relational database present EXFOR meta-data in tables accessible for SQL commands, but data points (numerical data) are stored only as part of original EXFOR SUBENT in BLOB. In order to be used, numerical data need to be extracted from BLOB to EXFOR file, parsed and converted to universal form. Thus, our current EXFOR relational database forces us always to work with original EXFOR and requires additional software on production stage: EXFOR reader, parser, converter.

Presenting EXFOR data in C4, C5, R33, JSON, XML forms usually works well for specific tasks and users' communities but has problems trying to present whole EXFOR library. For example, structure of C4/C5 work well for evaluators because use ENDF compatible designation (MF:MT) but have limited number of independent variables and finally cover from 60 to 80% of the entire library.

Another task is to provide needed information and recipe for automatic renormalization of EXFOR data using new cross section standards and decay data and for data corrections shared by experts.

Dream

It would be nice to store and directly access all data in standard relational form without intermediate storage and operations, but there are some problems with rational representation of EXFOR data in relational form associated with a large variety of the data and volume (now EXFOR has more than 500 types of data headers and 18 million data points).

Solution

Storage of all data points in modern relational database supporting JSON data type and providing direct SQL access could solve all these problems. It can also (a) solve problem of accessing the data from different programming languages, and (b) avoid problems with software distribution, installation, licences, etc.

X4Pro concept

1. Continue relational model. *Traditional SQL database storing data in tables. Continues and extends "EXFOR-Relational" project, 2000-2022.*
2. Fully relational. *All meta-data and numerical values presented in tables and accessible by SQL commands.*
3. Multi-model. *Table cells contain single values and also many values as semi-structured data in JSON. Note: "JSON" data type supported by modern relational DBMS via functions extending SQL commands (since ~2015).*
4. Universal. *Flexible SQL search, filtering, sorting allows to produce any data hierarchy on the fly; data in original and computational forms; includes monitor and decay data to be used for automatic renormalization, instructions for data modifications from experts. Implemented in MariaDB and SQLite, tested on Windows, Linux, MacOS. Can be used as starting point for other projects: from students with homework to professionals with advanced tasks.*
5. Powerful. *Oriented to programming users: they can do much more then using Web and GUI interfaces with fixed functionality.*
6. Rational. *No need EXFOR parsers for new languages. Can be used by programs on any language supporting SQL: Python, Java, JavaScript, Fortran, Perl, etc.*

X4Pro offers

1. EXFOR data without EXFOR format.
   - *All data points, data for corrections, meta-data are provided in the database.*
   - *No need in original EXFOR for end-users.*
   - *No need in new EXFOR parsers/converters for new programming languages.*
   - *No need in intermediate files and formats with fixed structure (C5, XML, JSON).*
   - *Simple for programming on any language supporting SQL for data search, filtering, sorting, retrieval, renormalization.*

2. Local EXFOR database for programmatic access.

   Providing data for various tasks required automatization, "non-so-general" to be implemented under Web/GUI interface proposed by data centres; packages required access to all experimental data at once; evaluation software required data corrections

3. Examples.

   24 examples of Fortran and Python programs provided with source code (MIT licence) and "run-me" scripts retrieving and plotting data from local X4Pro and remote ENDF database via Web-API interface.

4. X5-JSON.

   Comprehensive EXFOR data presentation in JSON form.
   Can be used for creating another systems built on JSON objects (e.g. NoSQL databases).
   Example of building CouchDB is provided.

EXFOR Relational data formats overview

X4+ EXFOR-Interpreted; X4± Interactive Tree

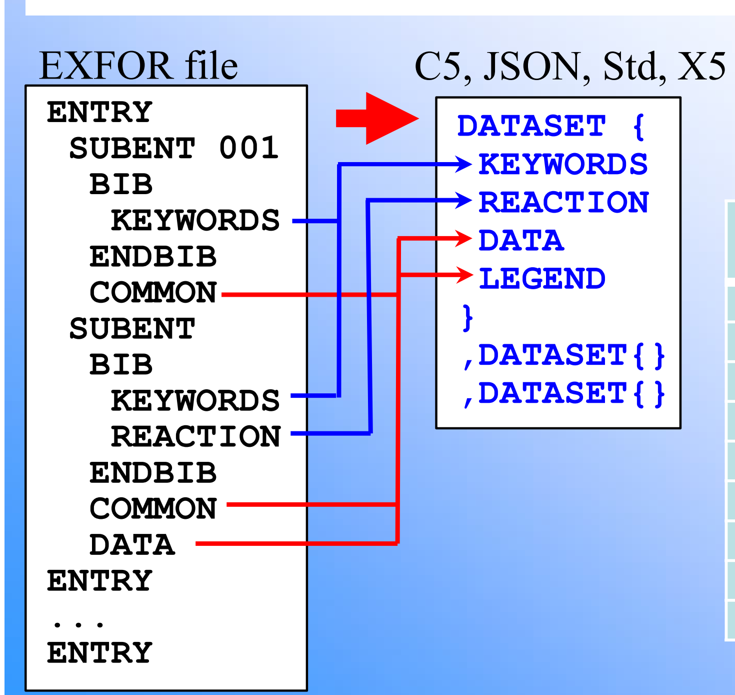
1. Presents EXFOR as it is + extra lines with information from Dictionaries, NSR, etc.
2. Numbers in traditional style
3. No limit on the number of values per line

XML

1. Repeats structure of EXFOR file using nested <elements>; includes information from EXFOR Dictionaries explaining codes
2. Numbers are presented in traditional style (no more E-less Fortran format for numbers)

Concept of Dataset //C5, JSON, JSON_FY, Std_out, X5Z

1. File contains Datasets; no text blocks for ENTRY, SUBENT, BIB; no Pointers
2. Dataset is identified by DatasetID:="SUBENT+Pointer"; includes all information related to one reaction: Reaction-code, selected/all Keywords from SUBENT-1 and current SUBENT, Data-section and Legend
3. Data-section: all data from DATA and COMMON from SUBENT-1 and current SUBENT
4. Data are presented as table function *f(x1,x2,…,par)*, where variables are sorted according to Dictionary-213 and Dict.24
5. Legend and Keywords contain EXFOR codes and their interpretation (e.g. basic-units and conversion factors)
6. C5, JSON_FY and X5Z contain computational data values; StdOut, XML and JSON (as of now) – only original values



EXFOR file

```
ENTRY
  SUBENT 001
   BIB
     KEYWORDS
   ENDBIB
   COMMON
  SUBENT
   BIB
     KEYWORDS
     REACTION
   ENDBIB
   COMMON
   DATA
ENTRY
. . .
ENTRY
```

C5, JSON, Std, X5

```
DATASET {
  KEYWORDS
  REACTION
  DATA
  LEGEND
}
,DATASET{}
,DATASET{}
```

Comparison of formats: summary

Nucl. data format	Numbers' format /Language	Sequence (main block)	Meta data	Interpret. from Dict.	Orig. data	Comput. data	Renorm. data
EXFOR	Fixed-len, E-less	ENTRY	yes	no	yes	no	no
C4	Fixed-fmt lines	SUBENT	no	no	no	yes	no
C5	Fixed-fmt lines	Datasets	yes	yes	no	yes	yes
X4+	Flex. fields /HTML	ENTRY	yes	yes	yes	no	no
XML	Flex. fields /XML	ENTRY	yes	yes	yes	no	no
JSON	Flex. fields /JSON	Datasets	yes	yes	yes	no	no
JSON_FY	Flex. fields / JSON	Datasets	yes	yes	no	yes	no
JSON_X4	Flex. fields / JSON	Datasets	yes	yes	yes	yes	no
X5Z.JSON	Flex. fields / JSON	Datasets	yes	yes	yes	yes	yes

X4Pro implementation

Translation from MariaDB to SQLite is done automatically by a bash script (Dec.2022) working ~4 hours and producing a single 8Gb file x4sqlite1.db with 10 new tables:

1.	x4pro_ds	*Datasets*
2.	x4pro_hdr	*Headers*
3.	x4pro_kw	*Keywords*
4.	x4pro_x4data	*EXFOR data points (json)*
5.	x4pro_x4cdat	*EXFOR data points in basic units (json)*
6.	x4pro_c5dat	*Computational data points (real)* *+ total sys/stat/partial errors* *+ old and new monitor CS data*
7.	x4pro_autocorr	*Decay data for product and monitor*
8.	x4pro_expertcorr	*Experts' corrections (Python)*
9.	x4pro_x4z	*Subentries in X4Z.JSON*
10.	x4pro_x5z	*Datasets in X5Z.JSON*

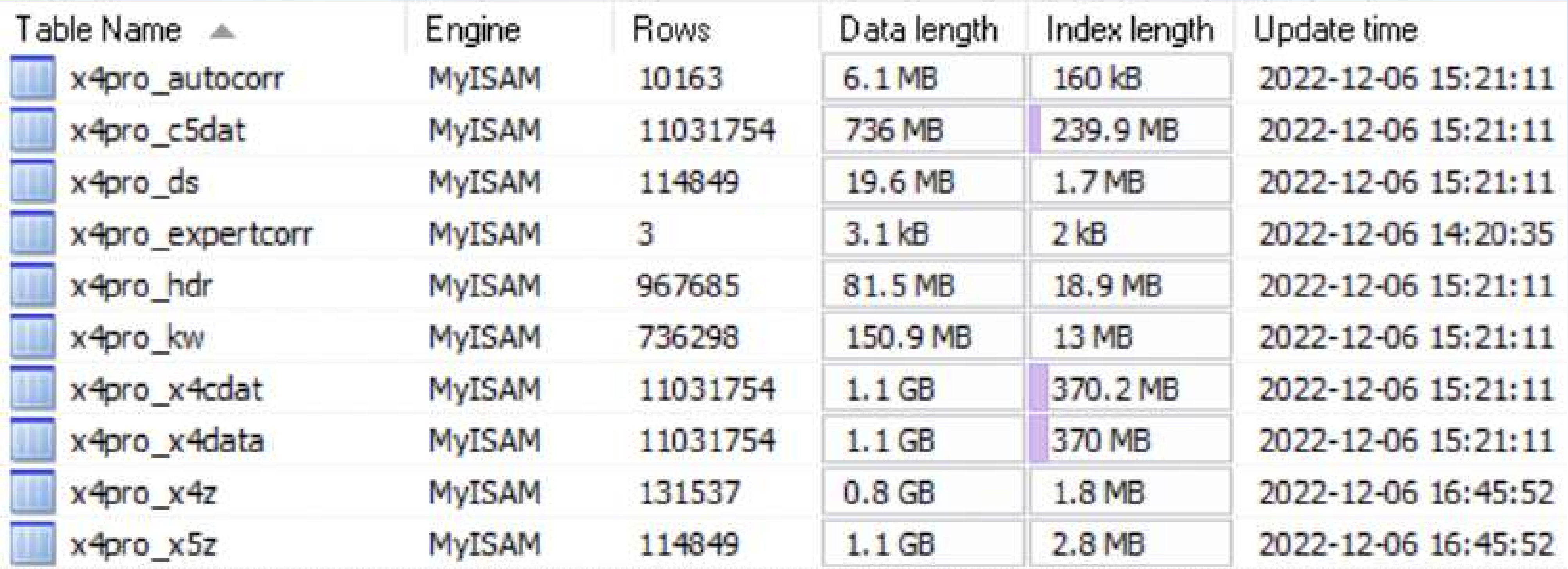


Table Name	Engine	Rows	Data length	Index length	Update time
x4pro_autocorr	MyISAM	10163	6.1 MB	160 kB	2022-12-06 15:21:11
x4pro_c5dat	MyISAM	11031754	736 MB	239.9 MB	2022-12-06 15:21:11
x4pro_ds	MyISAM	114849	19.6 MB	1.7 MB	2022-12-06 15:21:11
x4pro_expertcorr	MyISAM	3	3.1 kB	2 kB	2022-12-06 14:20:35
x4pro_hdr	MyISAM	967685	81.5 MB	18.9 MB	2022-12-06 15:21:11
x4pro_kw	MyISAM	736298	150.9 MB	13 MB	2022-12-06 15:21:11
x4pro_x4cdat	MyISAM	11031754	1.1 GB	370.2 MB	2022-12-06 15:21:11
x4pro_x4data	MyISAM	11031754	1.1 GB	370 MB	2022-12-06 15:21:11
x4pro_x4z	MyISAM	131537	0.8 GB	1.8 MB	2022-12-06 16:45:52
x4pro_x5z	MyISAM	114849	1.1 GB	2.8 MB	2022-12-06 16:45:52

Maintenance of EXFOR relational. X4Pro production.

The system is functioning at the IAEA-NDS and NNDC since 2004

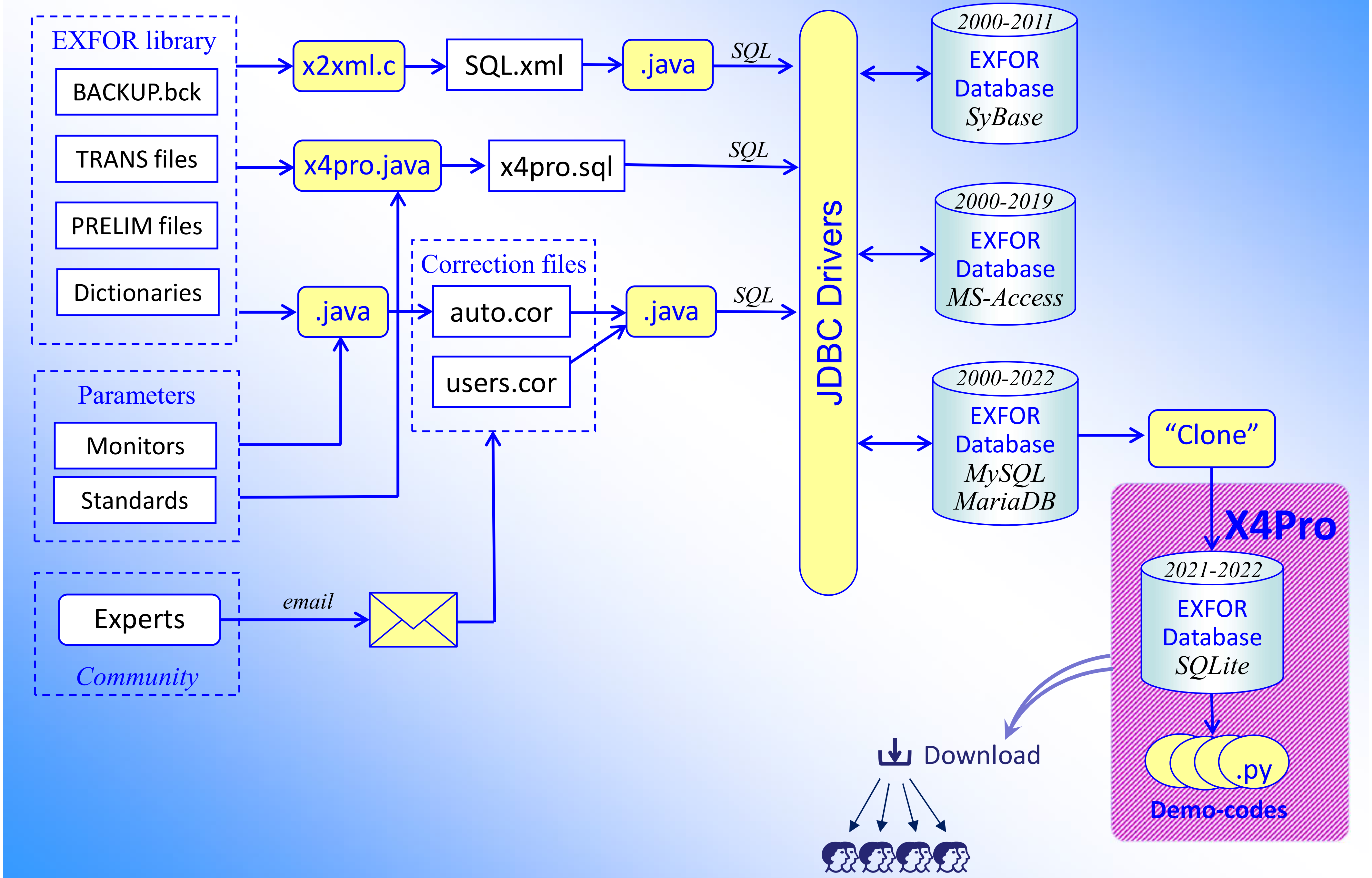
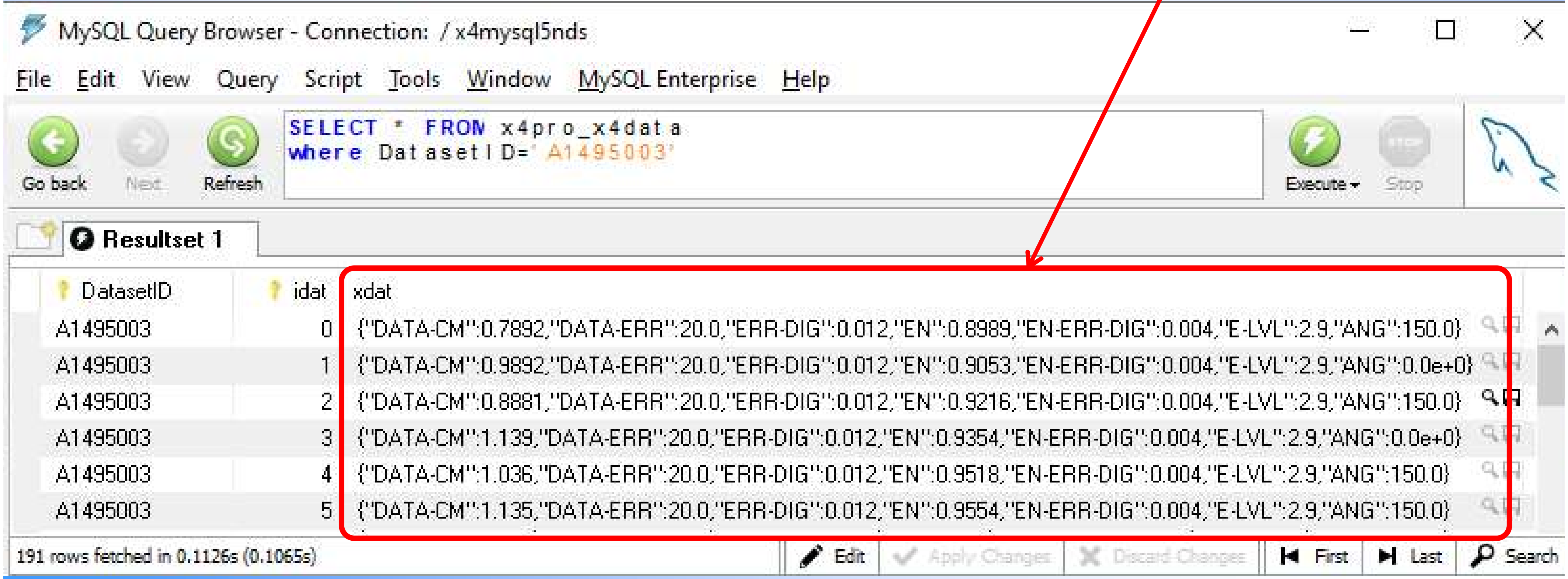


Table x4pro_x4data

```
create table x4pro_x4data (
   DatasetID   varchar(9) not null,
   idat        integer null,
   xdat        JSON null,
   primary key (DatasetID,idat)
)
```

“semi-structured” data: type JSON

Using MySQL Query Browser



MySQL Query Browser - Connection: / x4mysql5nds
File Edit View Query Script Tools Window MySQL Enterprise Help
Go back
Next
Refresh
Execute
Stop
Resultset 1
Edit
Apply Changes
Discard Changes
First
Last
Search

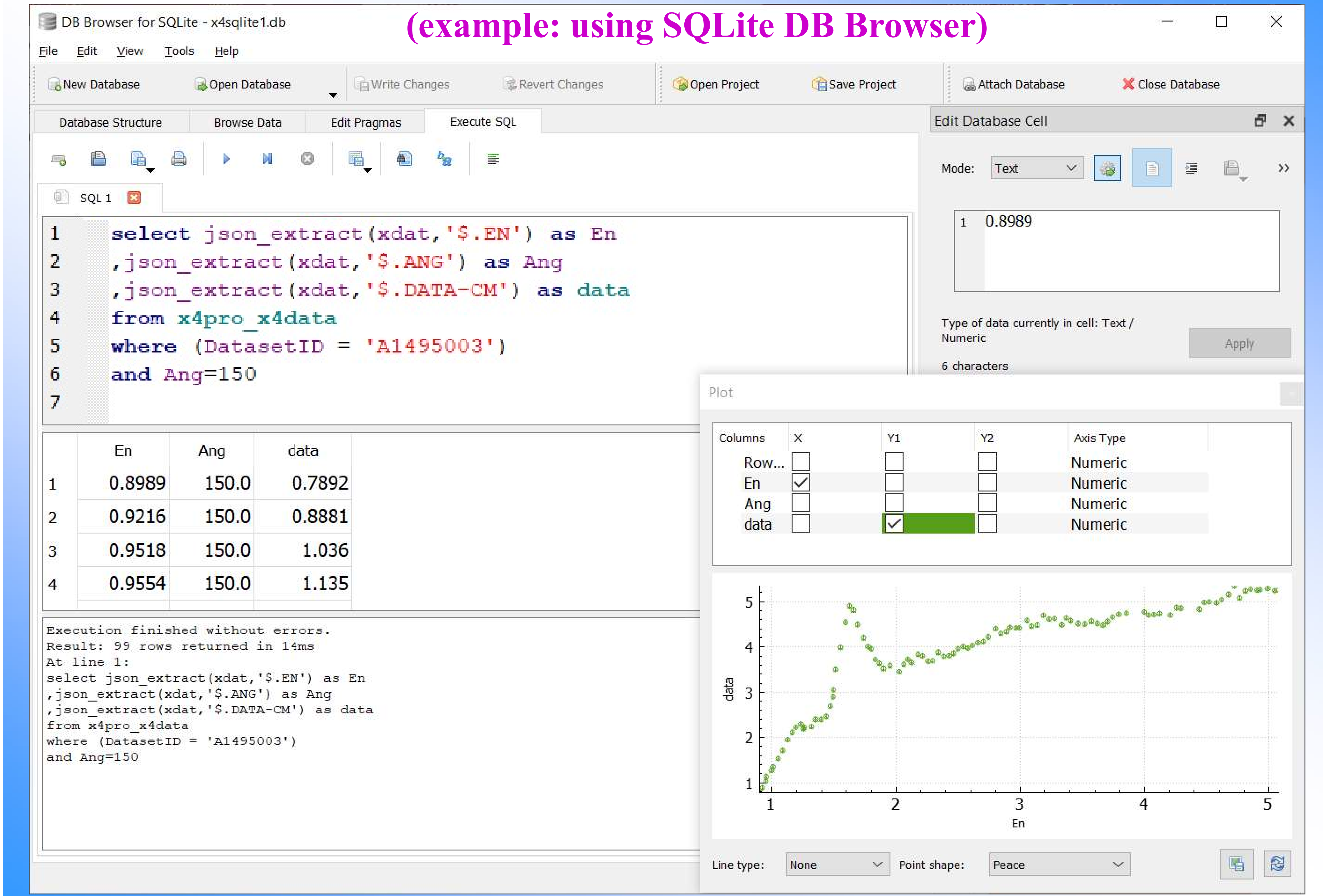
```
SELECT * FROM x4pro_x4data
where DatasetID='A1495003'
```

DatasetID	idat	xdat
A1495003	0	{"DATA-CM":0.7892,"DATA-ERR":20.0,"ERR-DIG":0.012,"EN":0.8989,"EN-ERR-DIG":0.004,"E-LVL":2.9,"ANG":150.0}
A1495003	1	{"DATA-CM":0.9892,"DATA-ERR":20.0,"ERR-DIG":0.012,"EN":0.9053,"EN-ERR-DIG":0.004,"E-LVL":2.9,"ANG":0.0e+0}
A1495003	2	{"DATA-CM":0.8881,"DATA-ERR":20.0,"ERR-DIG":0.012,"EN":0.9216,"EN-ERR-DIG":0.004,"E-LVL":2.9,"ANG":150.0}
A1495003	3	{"DATA-CM":1.139,"DATA-ERR":20.0,"ERR-DIG":0.012,"EN":0.9354,"EN-ERR-DIG":0.004,"E-LVL":2.9,"ANG":0.0e+0}
A1495003	4	{"DATA-CM":1.036,"DATA-ERR":20.0,"ERR-DIG":0.012,"EN":0.9518,"EN-ERR-DIG":0.004,"E-LVL":2.9,"ANG":150.0}
A1495003	5	{"DATA-CM":1.135,"DATA-ERR":20.0,"ERR-DIG":0.012,"EN":0.9554,"EN-ERR-DIG":0.004,"E-LVL":2.9,"ANG":150.0}

191 rows fetched in 0.1126s (0.1065s)

Querying data from inside JSON

(example: using SQLite DB Browser)



```
select json_extract(xdat,'$.EN') as En
,json_extract(xdat,'$.ANG') as Ang
,json_extract(xdat,'$.DATA-CM') as data
from x4pro_x4data
where (DatasetID = 'A1495003')
and Ang=150
```

	En	Ang	data
1	0.8989	150.0	0.7892
2	0.9216	150.0	0.8881
3	0.9518	150.0	1.036
4	0.9554	150.0	1.135

```
Execution finished without errors.
Result: 99 rows returned in 14ms
At line 1:
select json_extract(xdat,'$.EN') as En
,json_extract(xdat,'$.ANG') as Ang
,json_extract(xdat,'$.DATA-CM') as data
from x4pro_x4data
where (DatasetID = 'A1495003')
and Ang=150
```

Mode: Text

1 0.8989

Type of data currently in cell: Text / Numeric

6 characters

Apply

Columns	X	Y1	Y2	Axis Type
Row...	□	□	□	Numeric
En	☑	□	□	Numeric
Ang	□	□	□	Numeric
data	□	☑	□	Numeric

Line type: None   Point shape: Peace

Simplify SQL commands using Views

Using Tables only

```
select x4data_cdat.DatasetID,REACODE.fullCode
 ,ENTRY.YearRef1,ENTRY.Author1Ini,ENTRY.Author1
 ,x4data_cdat.idat as iPoint
 ,x4data_cdat.y    as Sig
 ,x4data_cdat.dy   as dSig
 ,x4data_cdat.x1   as En
 ,x4data_cdat.dx1  as dEn
 ,x4data_cdat.x2   as Eout
 ,x4data_cdat.dx2  as dEout
 ,x4data_cdat.x3   as An
 ,x4data_cdat.dx3  as dAn
from x4data_cdat
 inner join REACODE on
  REACODE.ReacodeID=x4data_cdat.DatasetID
 inner join REACSTR on
  REACSTR.ReacodeID=REACODE.ReacodeID
 inner join SUBENT on REACODE.SubentID=SUBENT.SubentID
 inner join ENTRY on ENTRY.EntryID=SUBENT.EntryID
where (REACSTR.Target like 'F-19')
  and (REACSTR.Reaction like 'n,x')
  and (REACSTR.Quant like 'DAE')
  and (REACODE.outParticles like '[n]')
  and (REACSTR.SF8='')
  and ((REACSTR.SF9='') or (REACSTR.SF9='EXP'))
  and (REACODE.nReacstr=1)
  and (An>=25) and (An<=45)
order by REACODE.fullCode,
  ENTRY.YearRef1 desc,x4data_cdat.DatasetID
  ,En,An,Eout,x4data_cdat.idat
```

Using Views

```
select * from dae1
where Target like 'F-19’
 and Reaction like 'n,x’
 and outParticles like '[n]’
 and (An>=25) and (An<=45)
order by fullCode,
 YearRef1 desc,DatasetID,
 En,An,Eout,iPoint
```

Example: SIG

```
select * from sig1
where Target='Mn-55’
and MT=107
```

```
select * from sig1
where Target='Mn-55’
and Reaction like 'n,a'
```

View “sig1”

```
CREATE VIEW sig1 AS
 select x4pro_c5dat.DatasetID
 ,x4pro_c5dat.idat as iPoint
 ,REACODE.fullCode
 ,REACODE.Pointer,ENTRY.Entry,REACODE.SubAcc as Subent
 ,ENTRY.YearRef1,ENTRY.nAuthors,ENTRY.Author1Ini,ENTRY.Author1
 ,REACSTR.Target, REACSTR.Reaction
 ,lower(REACSTR.Projectile) as Projectile
 ,REACSTR.sProd,REACSTR.sTarg
 ,REACODE.zaTarget1,REACODE.zaIncident1
 ,REACODE.outParticles,REACODE.MF,REACODE.MT
 ,x4pro_c5dat.x1  as En
 ,x4pro_c5dat.dx1 as dEn
 ,x4pro_c5dat.y   as Sig
 ,x4pro_c5dat.dy  as dSig
from x4pro_c5dat
 inner join REACODE on REACODE.ReacodeID=x4pro_c5dat.DatasetID
 inner join REACSTR ON REACSTR.ReacodeID=REACODE.ReacodeID
 inner join SUBENT on REACODE.SubentID=SUBENT.SubentID
 inner join ENTRY on ENTRY.EntryID=SUBENT.EntryID
where
      (REACSTR.SF58 like ',SIG')
  and (REACSTR.SF8='')
  and ((REACSTR.SF9='') or (REACSTR.SF9='EXP'))
  and (REACODE.nReacstr=1)
order by
  REACODE.fullCode,ENTRY.YearRef1 desc,x4pro_c5dat.DatasetID
  ,En,x4pro_c5dat.idat
```

Automatic renormalization of EXFOR data

(part of EXFOR data correction system, 2010-2022)

1. Renormalization using old and new monitor cross sections

m0, dm0: “old monitor” – monitor-reaction cross sections used by authors

Source of data:

1) DATA, COMMON sections: MONIT, MONIN-ERR (EN, EN-NRM)
2) MONIT-REF pointing to another EXFOR data
3) MONIT-REF pointing to ENDF library (e.g., ENDF/B-5 Standards sub-library)
4) MONIT-REF pointing to “a publication” //--> Archive of old Monitors

m1, dm1: “new monitor” – monitor-reaction cross sections “recommended” now

Source of data:

1) IAEA Standards-2017
2) IRDFF-II (2019)

2. Renormalization of EXFOR data using newer Decay data

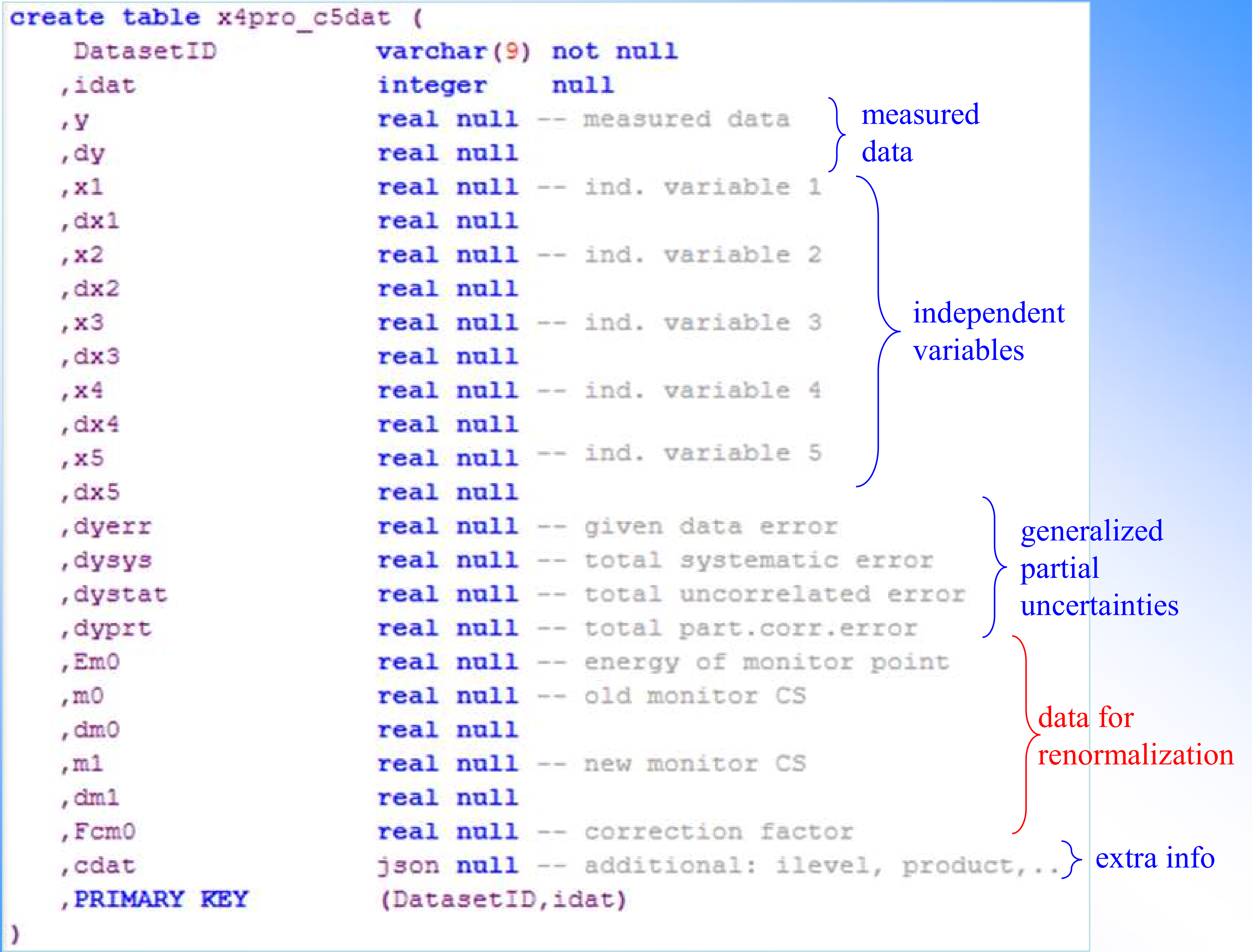
1) “AR” 511 keV annihilation decay data (intensity)
2) “DR” gamma line intensity

- EXFOR keywords: DECAY-DATA and DECAY-MON
- Data renormalized to the current ENSDF data - thanks to M.Verpelli (IAEA-NDS)

3. Data types available for automatic renormalization

“SIG”, “DA”, “DE”, “DAE”, “FY”.
Total number of Datasets where auto-renormalization: 17,395 (~10% of whole EXFOR)

Table x4pro_c5dat



```
create table x4pro_c5dat (
    DatasetID             varchar(9) not null
   ,idat                  integer    null
   ,y                     real null -- measured data
   ,dy                    real null
   ,x1                    real null -- ind. variable 1
   ,dx1                   real null
   ,x2                    real null -- ind. variable 2
   ,dx2                   real null
   ,x3                    real null -- ind. variable 3
   ,dx3                   real null
   ,x4                    real null -- ind. variable 4
   ,dx4                   real null
   ,x5                    real null -- ind. variable 5
   ,dx5                   real null
   ,dyerr                 real null -- given data error
   ,dysys                 real null -- total systematic error
   ,dystat                real null -- total uncorrelated error
   ,dyprt                 real null -- total part.corr.error
   ,Em0                   real null -- energy of monitor point
   ,m0                    real null -- old monitor CS
   ,dm0                   real null
   ,m1                    real null -- new monitor CS
   ,dm1                   real null
   ,Fcm0                  real null -- correction factor
   ,cdat                  json null -- additional: ilevel, product,..
   ,PRIMARY KEY           (DatasetID,idat)
)
```

measured data

independent variables

generalized partial uncertainties

data for renormalization

extra info

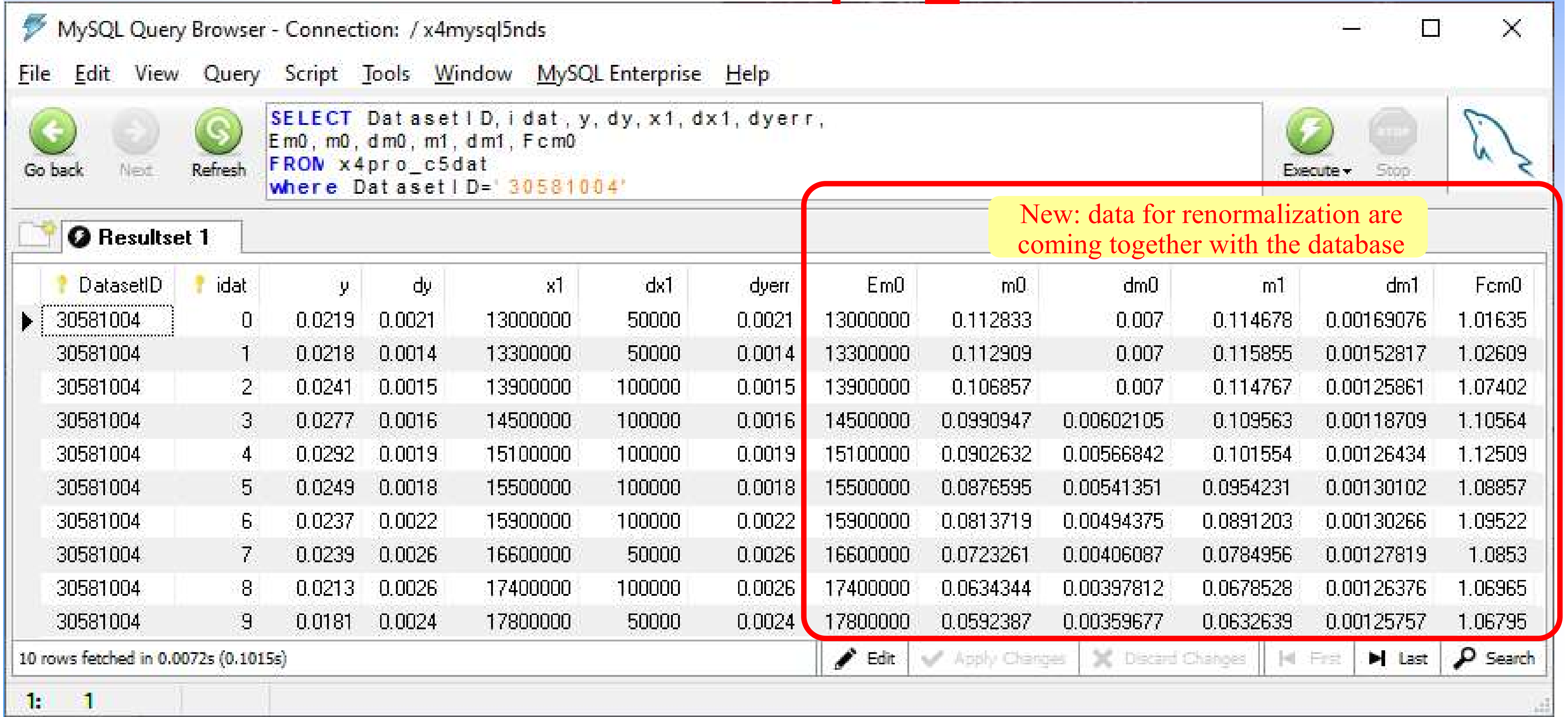
Now on EXFOR-Web:

```
30581004  x4u:20090506  #1980,Zupranska #Pts:10
 #[0]#---Monitor xs-data
 #[0]#Reaction: 25-MN-55(N,A)23-V-52,,SIG
 #[0]#Monitor:  26-FE-56(N,P)25-MN-56,,SIG
 #m0: {20377002,H.LISKIEN+,J,JNE/AB,19,73,196502} $ fe56np;#[0]#old monit-ref
 m0: exfor$20377002_fe56np;      #[0]#old monitor(energy) in EXFOR
 m1: recom$fe56np;               #[0]#new monitor(energy)
 dy=dy/y;                        #[0]#to rel. uncertainties----
 y=y/m0*m1;                      #[0]#renormalizing CS
 dy=(dy**2-dm0**2+dm1**2)**0.5;  #[0]#replace monitor uncertainties
 dy=dy*y;                        #[0]#to abs. uncertainties
```

Currently implemented using:

1. EXFOR Parser (C)
2. Interpreter (C)
3. Archive of monitors
4. Database of new standards

X4Pro: table x4pro_c5dat



```
SELECT DatasetID, idat, y, dy, x1, dx1, dyerr,
Em0, m0, dm0, m1, dm1, Fcm0
FROM x4pro_c5dat
where DatasetID='30581004'
```

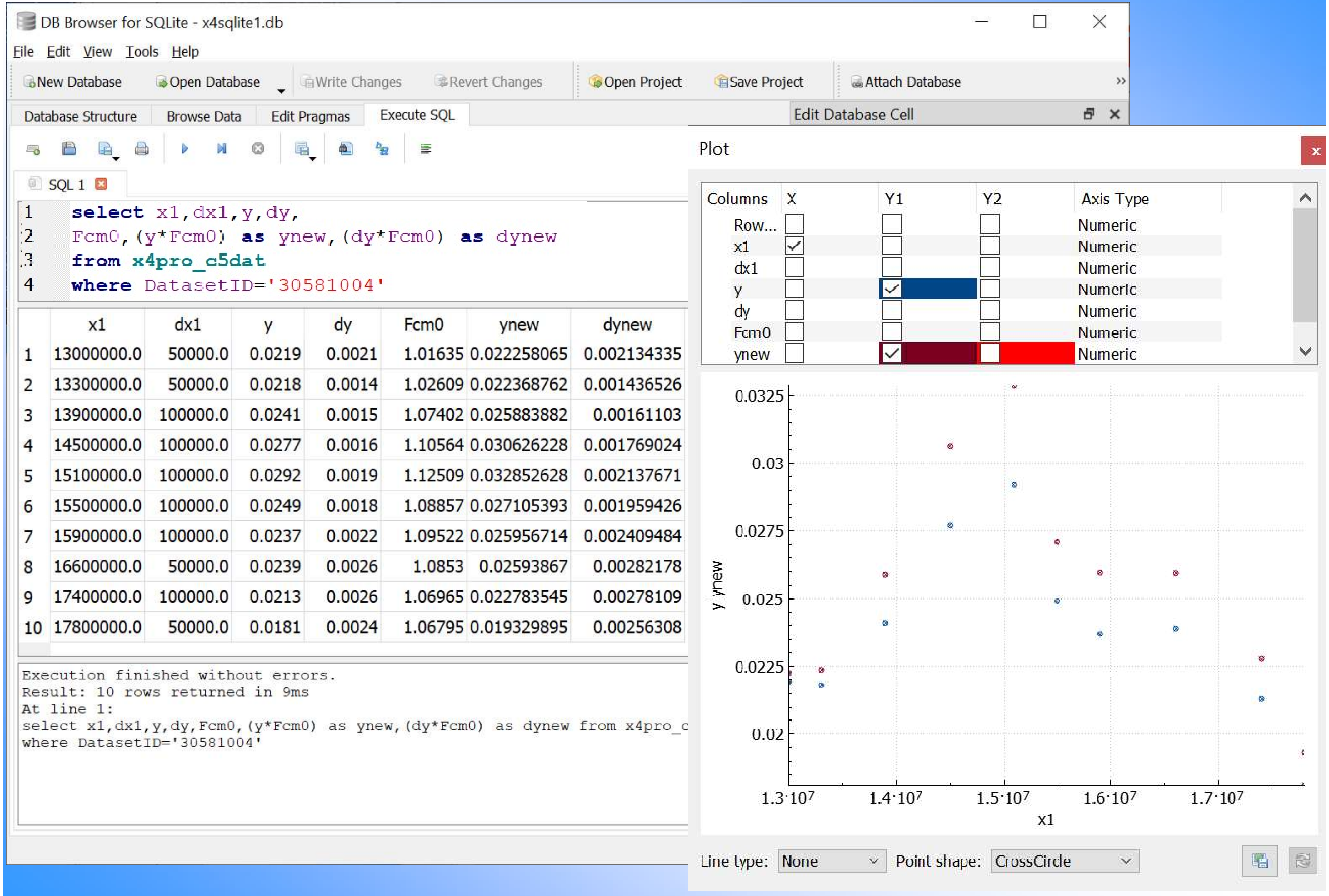
New: data for renormalization are coming together with the database

DatasetID	idat	y	dy	x1	dx1	dyerr	Em0	m0	dm0	m1	dm1	Fcm0
30581004	0	0.0219	0.0021	13000000	50000	0.0021	13000000	0.112833	0.007	0.114678	0.00169076	1.01635
30581004	1	0.0218	0.0014	13300000	50000	0.0014	13300000	0.112909	0.007	0.115855	0.00152817	1.02609
30581004	2	0.0241	0.0015	13900000	100000	0.0015	13900000	0.106857	0.007	0.114767	0.00125861	1.07402
30581004	3	0.0277	0.0016	14500000	100000	0.0016	14500000	0.0990947	0.00602105	0.109563	0.00118709	1.10564
30581004	4	0.0292	0.0019	15100000	100000	0.0019	15100000	0.0902632	0.00566842	0.101554	0.00126434	1.12509
30581004	5	0.0249	0.0018	15500000	100000	0.0018	15500000	0.0876595	0.00541351	0.0954231	0.00130102	1.08857
30581004	6	0.0237	0.0022	15900000	100000	0.0022	15900000	0.0813719	0.00494375	0.0891203	0.00130266	1.09522
30581004	7	0.0239	0.0026	16600000	50000	0.0026	16600000	0.0723261	0.00406087	0.0784956	0.00127819	1.0853
30581004	8	0.0213	0.0026	17400000	100000	0.0026	17400000	0.0634344	0.00397812	0.0678528	0.00126376	1.06965
30581004	9	0.0181	0.0024	17800000	50000	0.0024	17800000	0.0592387	0.00359677	0.0632639	0.00125757	1.06795

10 rows fetched in 0.0072s (0.1015s)

X4Pro: data renormalization with SQL SELECT

(example: using SQLite DB Browser)



```sql
select x1,dx1,y,dy,
Fcm0,(y*Fcm0) as ynew,(dy*Fcm0) as dynew
from x4pro_c5dat
where DatasetID='30581004'
```

	x1	dx1	y	dy	Fcm0	ynew	dynew
1	13000000.0	50000.0	0.0219	0.0021	1.01635	0.022258065	0.002134335
2	13300000.0	50000.0	0.0218	0.0014	1.02609	0.022368762	0.001436526
3	13900000.0	100000.0	0.0241	0.0015	1.07402	0.025883882	0.00161103
4	14500000.0	100000.0	0.0277	0.0016	1.10564	0.030626228	0.001769024
5	15100000.0	100000.0	0.0292	0.0019	1.12509	0.032852628	0.002137671
6	15500000.0	100000.0	0.0249	0.0018	1.08857	0.027105393	0.001959426
7	15900000.0	100000.0	0.0237	0.0022	1.09522	0.025956714	0.002409484
8	16600000.0	50000.0	0.0239	0.0026	1.0853	0.02593867	0.00282178
9	17400000.0	100000.0	0.0213	0.0026	1.06965	0.022783545	0.00278109
10	17800000.0	50000.0	0.0181	0.0024	1.06795	0.019329895	0.00256308

```
Execution finished without errors.
Result: 10 rows returned in 9ms
At line 1:
select x1,dx1,y,dy,Fcm0,(y*Fcm0) as ynew,(dy*Fcm0) as dynew from x4pro_c
where DatasetID='30581004'
```

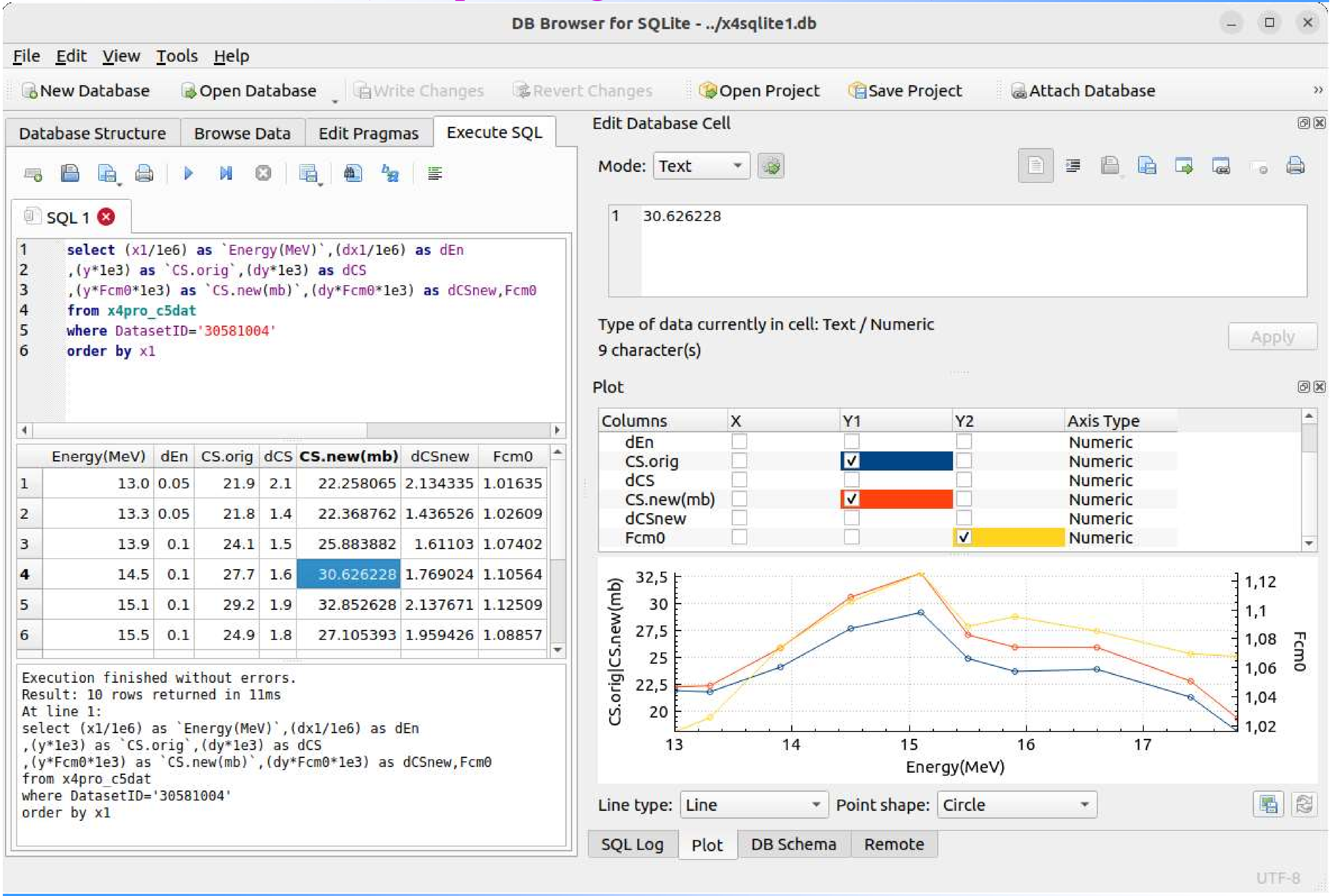
Columns	X	Y1	Y2	Axis Type
Row...	□	□	□	Numeric
x1	☑	□	□	Numeric
dx1	□	□	□	Numeric
y	□	☑	□	Numeric
dy	□	□	□	Numeric
Fcm0	□	□	□	Numeric
ynew	□	☑	□	Numeric

Line type: None

Point shape: CrossCircle

X4Pro: data renormalization with SQL SELECT

(example: using SQLite DB Browser)



```sql
select (x1/1e6) as `Energy(MeV)`,(dx1/1e6) as dEn
,(y*1e3) as `CS.orig`,(dy*1e3) as dCS
,(y*Fcm0*1e3) as `CS.new(mb)`,(dy*Fcm0*1e3) as dCSnew,Fcm0
from x4pro_c5dat
where DatasetID='30581004'
order by x1
```

	Energy(MeV)	dEn	CS.orig	dCS	CS.new(mb)	dCSnew	Fcm0
1	13.0	0.05	21.9	2.1	22.258065	2.134335	1.01635
2	13.3	0.05	21.8	1.4	22.368762	1.436526	1.02609
3	13.9	0.1	24.1	1.5	25.883882	1.61103	1.07402
4	14.5	0.1	27.7	1.6	30.626228	1.769024	1.10564
5	15.1	0.1	29.2	1.9	32.852628	2.137671	1.12509
6	15.5	0.1	24.9	1.8	27.105393	1.959426	1.08857

```
Execution finished without errors.
Result: 10 rows returned in 11ms
At line 1:
select (x1/1e6) as `Energy(MeV)`,(dx1/1e6) as dEn
,(y*1e3) as `CS.orig`,(dy*1e3) as dCS
,(y*Fcm0*1e3) as `CS.new(mb)`,(dy*Fcm0*1e3) as dCSnew,Fcm0
from x4pro_c5dat
where DatasetID='30581004'
order by x1
```

Mode: Text

```
1 30.626228
```

Type of data currently in cell: Text / Numeric
9 character(s)

Columns	X	Y1	Y2	Axis Type
dEn	□	□	□	Numeric
CS.orig	□	☑	□	Numeric
dCS	□	□	□	Numeric
CS.new(mb)	□	☑	□	Numeric
dCSnew	□	□	□	Numeric
Fcm0	□	□	☑	Numeric

Line type: Line
Point shape: Circle

Part III.

Illustrating usage of X4Pro database

Demo codes: Python, Fortran, JavaScript

Visualization: Python/Plotly+Matplotlib
JavaScript/Plotly

Platforms: Windows, Linux, MacOS

Tests, demo-codes, platforms and technologies

I. Retrieve experimental data from local X4Pro with evaluated data from Web ENDF retrieval system: Python3 with Plotly or Matplotlib

1. *Cross sections (MF3 + uncertainties from MF33)*
2. *Angular distributions (MF4)*
3. *Emission spectra (MF5)*
4. *Double differential cross sections (MF6)*
5. *Fission yield (MF1)*
6. *Hidden EXFOR data: Mass×TKE distribution, EXFOR data correlations*

for young

II. Retrieve data from local X4Pro using GFortran and GCC

1. *Cross sections (MF3), output C4 file*
2. *Double differential cross sections (MF6)*
3. *Retrieve LEG/RS and SIG from different Subent and generate DA → C4*

for old

III. Data renormalization/modification on Python + Plotly or Matplotlib

1. *Automatic renormalization*
2. *User's modifications*
3. *Experts' modifications (taken from database)*
4. *Ratios to cross sections recalculations*
5. *Retrieve Legendre coefficient L[0] and calculate cross sections*
6. *Retrieve LEG/RS and SIG from different Subent and generate DA*
7. *Retrieve LEG generate DA output draft of R33 (for IBANDL)*

for evaluators+

IV. Populating CouchDB database using X5-JSON in Python

1. *Retrieve X4Z.JSON from table x4pro_x4z store in local CouchDB*

study

V. Data retrievals from local X4Pro using javascript (+ENDF +Plotly)

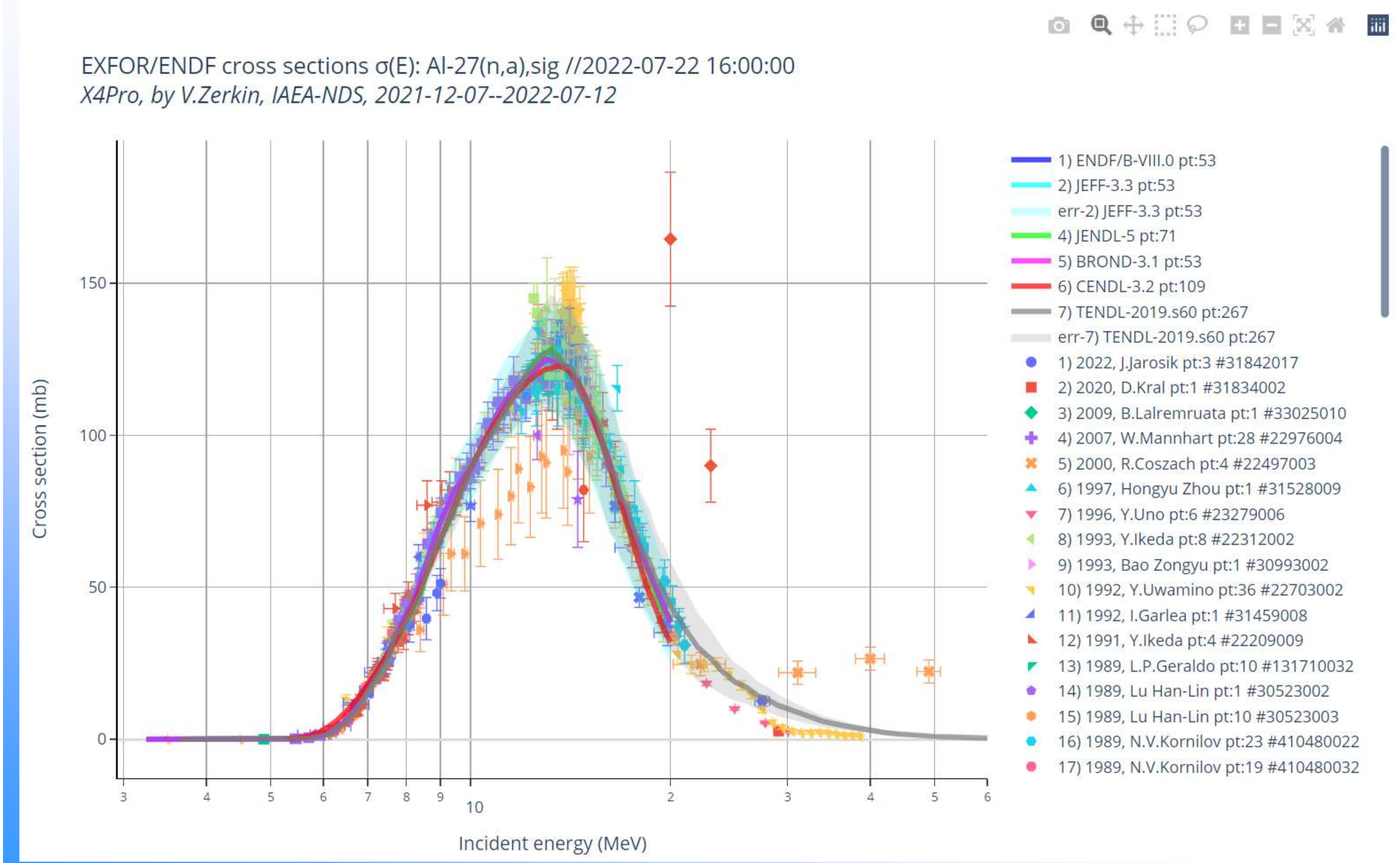
1. *Cross sections (MF3) with GUI/Html5*
2. *Retrievals from javascript under Node.js*

for fun

Note. Trial distribution: database file ~8Gb, python codes are built on modules containing ~100 lines each; fortran codes 100-200 lines; item V is not included

Cross sections: EXFOR + ENDF.MF3/MF33

Demo code: python3 + requests + plotly

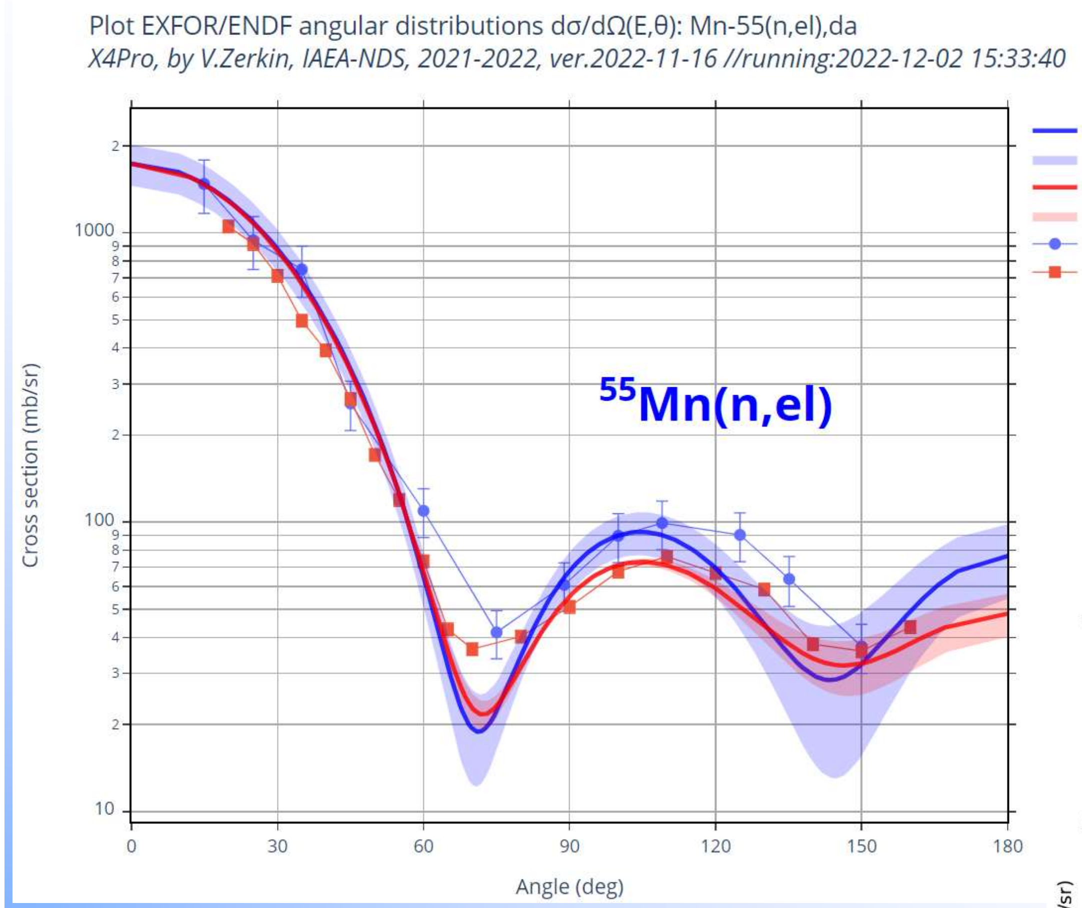


EXFOR/ENDF cross sections σ(E): Al-27(n,a),sig //2022-07-22 16:00:00

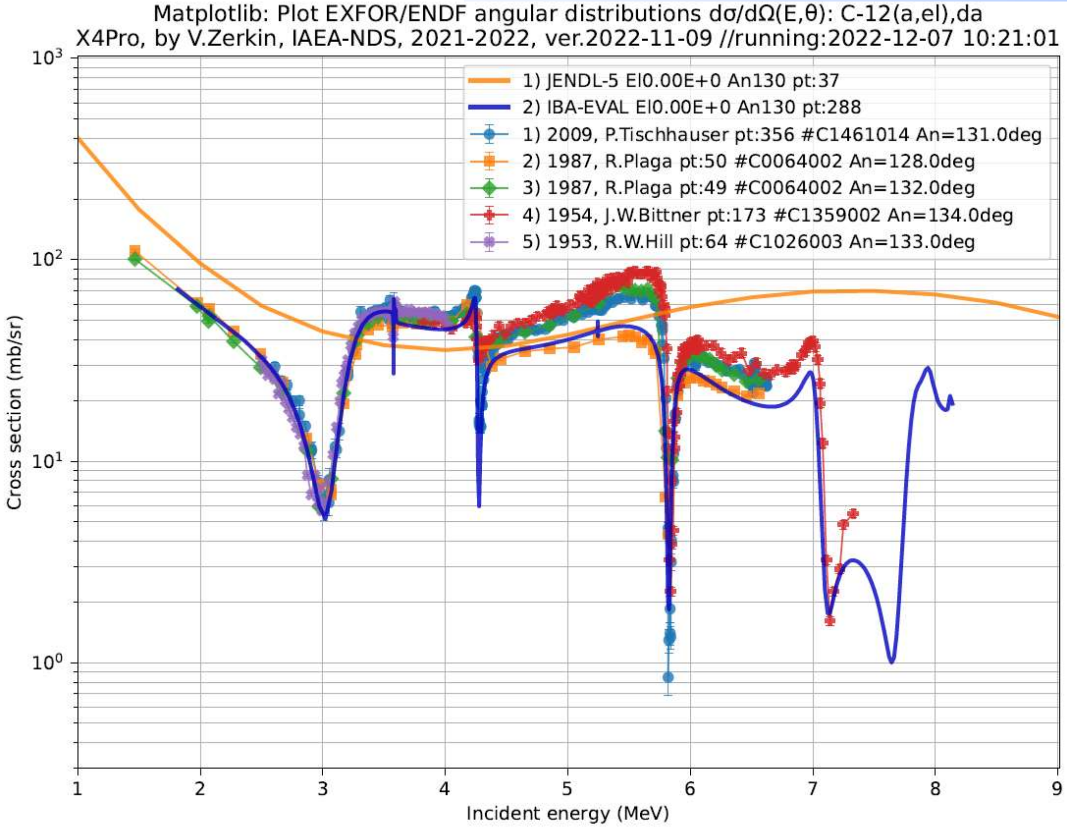
X4Pro, by V.Zerkin, IAEA-NDS, 2021-12-07--2022-07-12

Angular distributions (DA+MF4.MF34)

Demo-code: python3 + requests + plotly or matplotlib

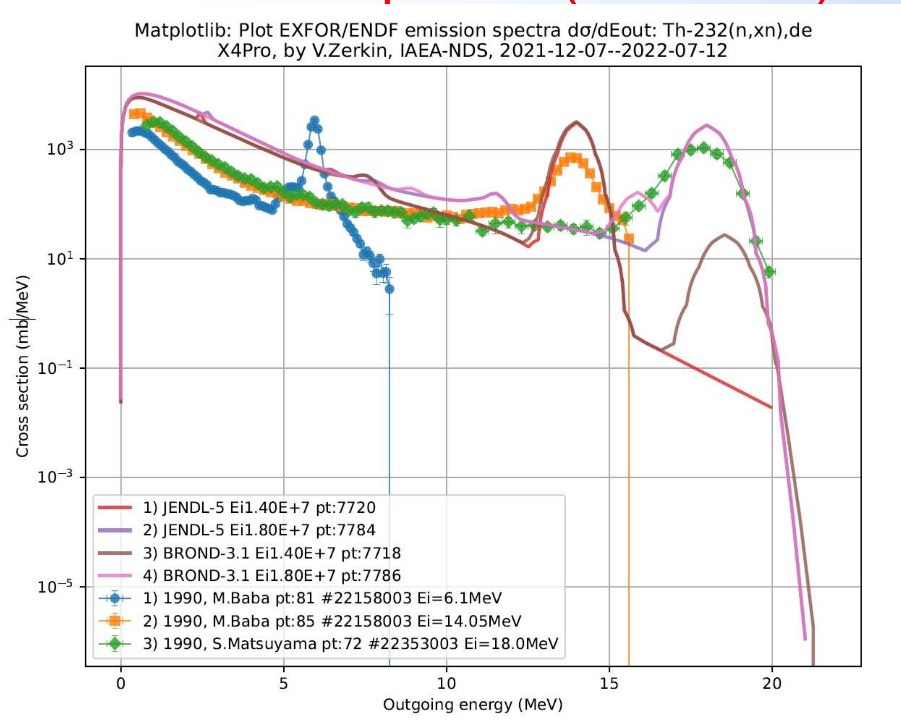

Plot EXFOR/ENDF angular distributions dσ/dΩ(E,θ): Mn-55(n,el),da

X4Pro, by V.Zerkin, IAEA-NDS, 2021-2022, ver.2022-11-16 //running:2022-12-02 15:33:40

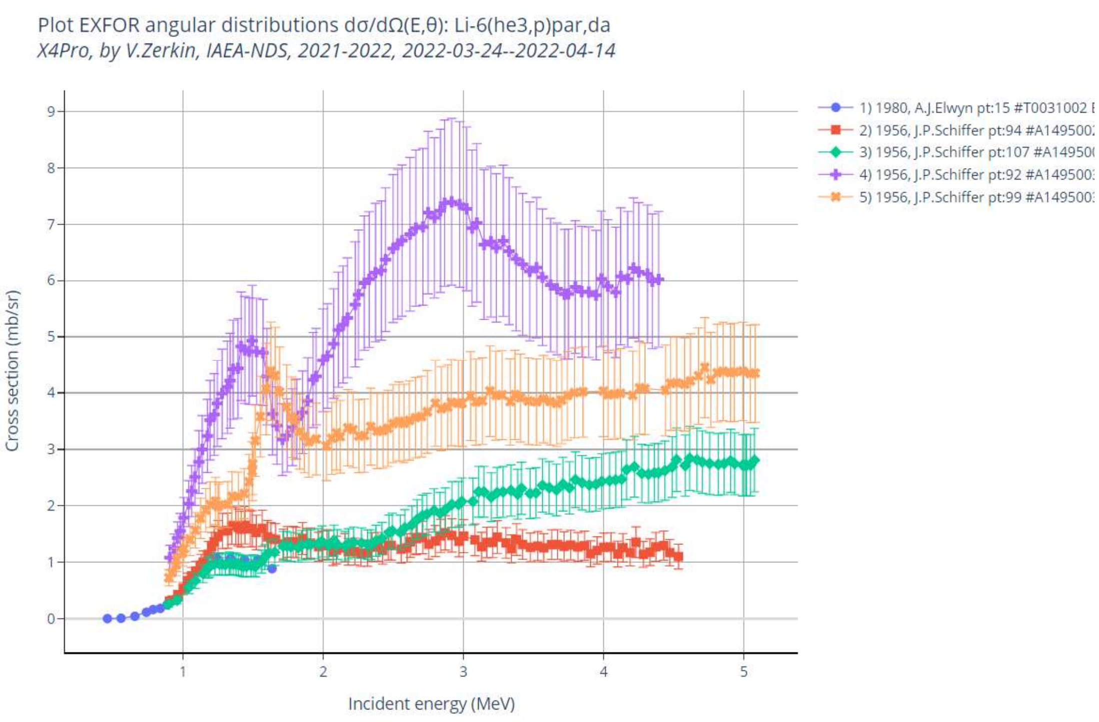

Matplotlib: Plot EXFOR/ENDF angular distributions dσ/dΩ(E,θ): C-12(a,el),da

X4Pro, by V.Zerkin, IAEA-NDS, 2021-2022, ver.2022-11-09 //running:2022-12-07 10:21:01

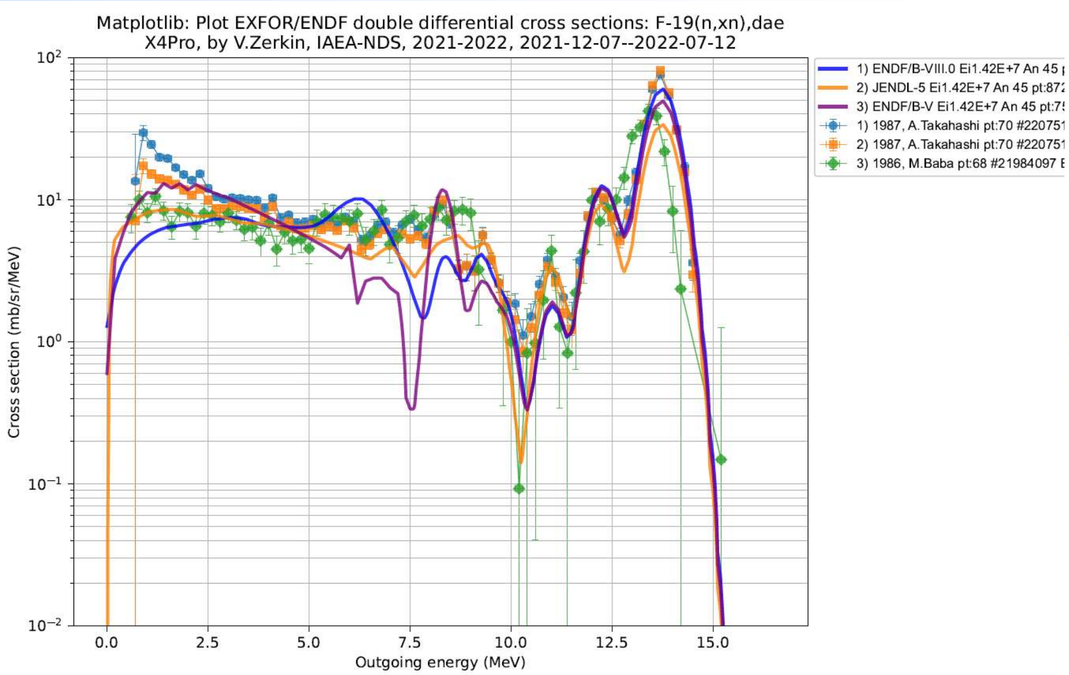
Emission spectra (DE+MF5)



Angular distribution (partial: DAP)

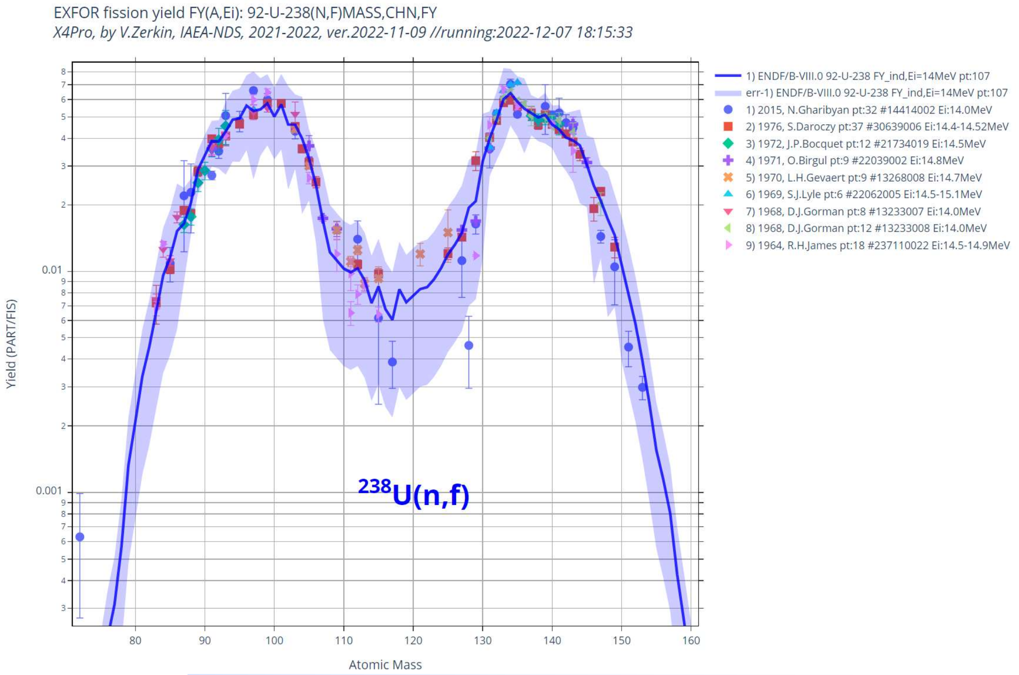


Double differential cross sections (MF6)



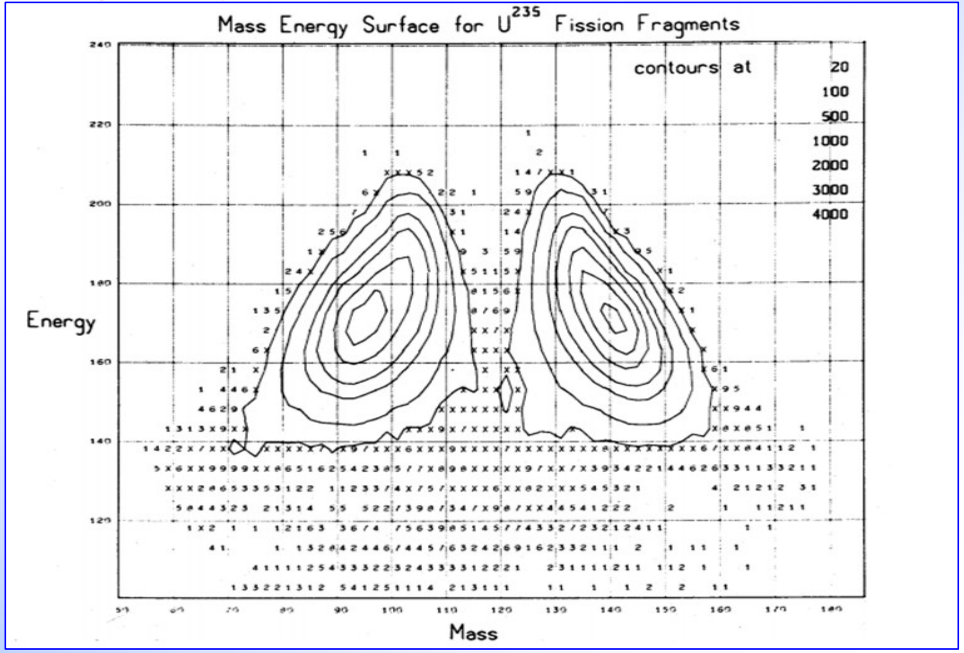
Fission-product yield from EXFOR (FY)

92-U-238(N,F)MASS,CHN,FY : Total chain yield of fission products



Example of “Native” EXFOR plotting (Mass-TKE)

Original publication:



EXFOR: #21095008: 92-U-235(N,F)MASS,PR/FRG,NU/TKE

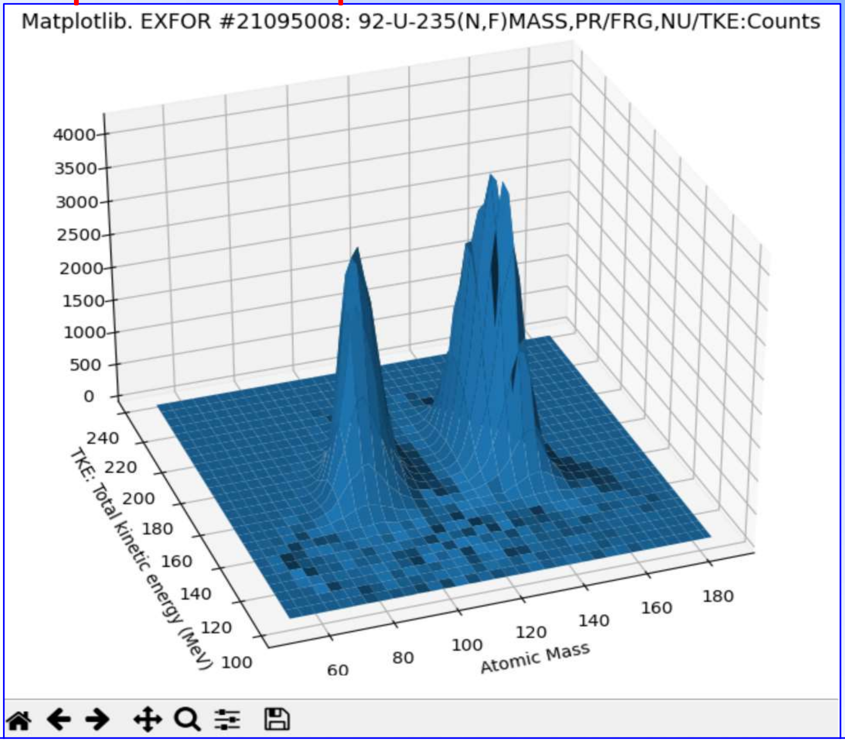
Mass-Energy distribution for both fission fragments

X: MASS(NO-DIM):Atomic mass of nuclide

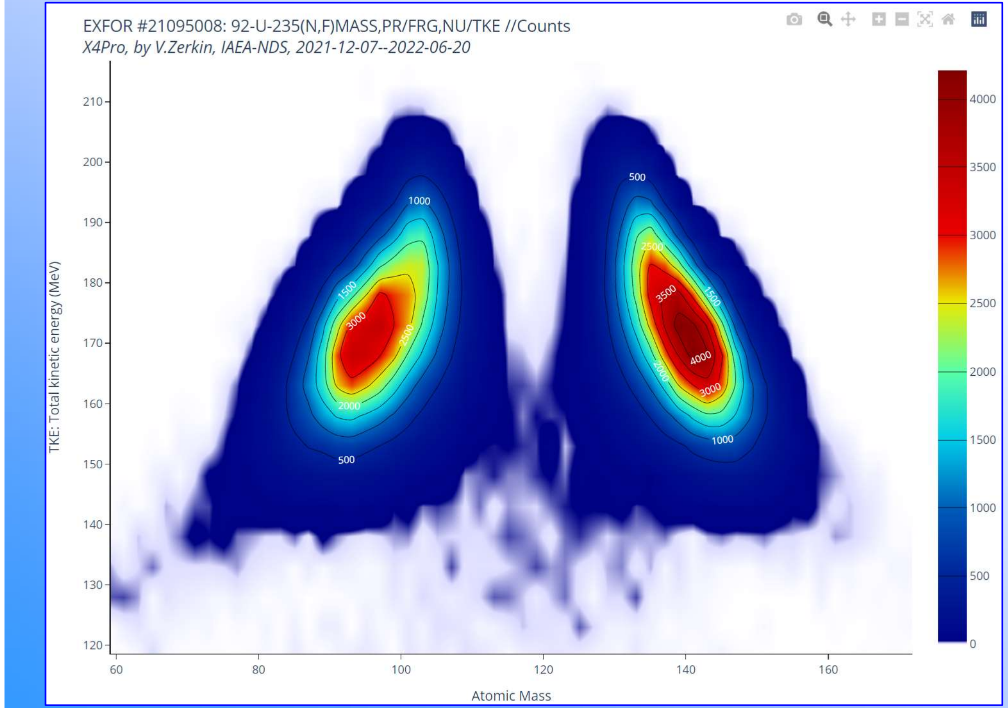
Y: E(MEV):Energy of outgoing particle, lab. system

Z: MISC(NO-DIM):Number of events detected

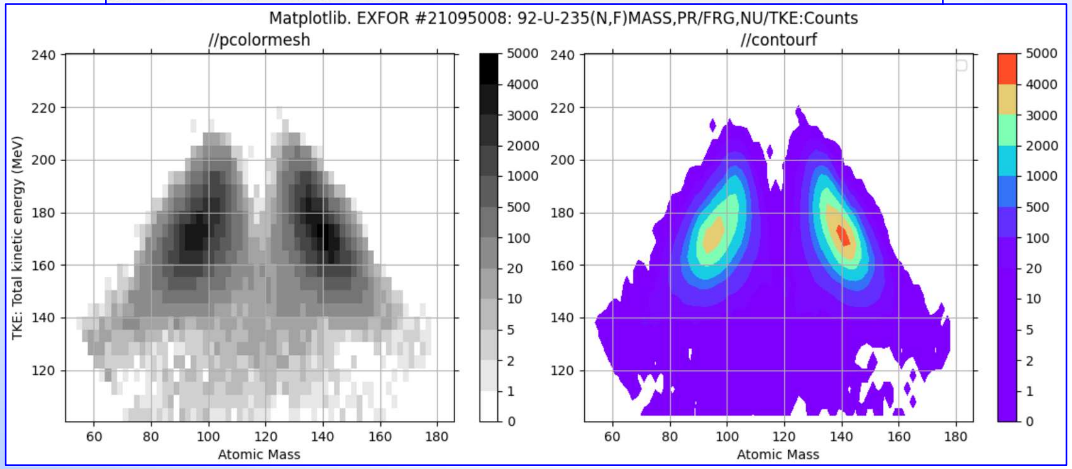
X4pro → Matplotlib:


Matplotlib. EXFOR #21095008: 92-U-235(N,F)MASS,PR/FRG,NU/TKE:Counts

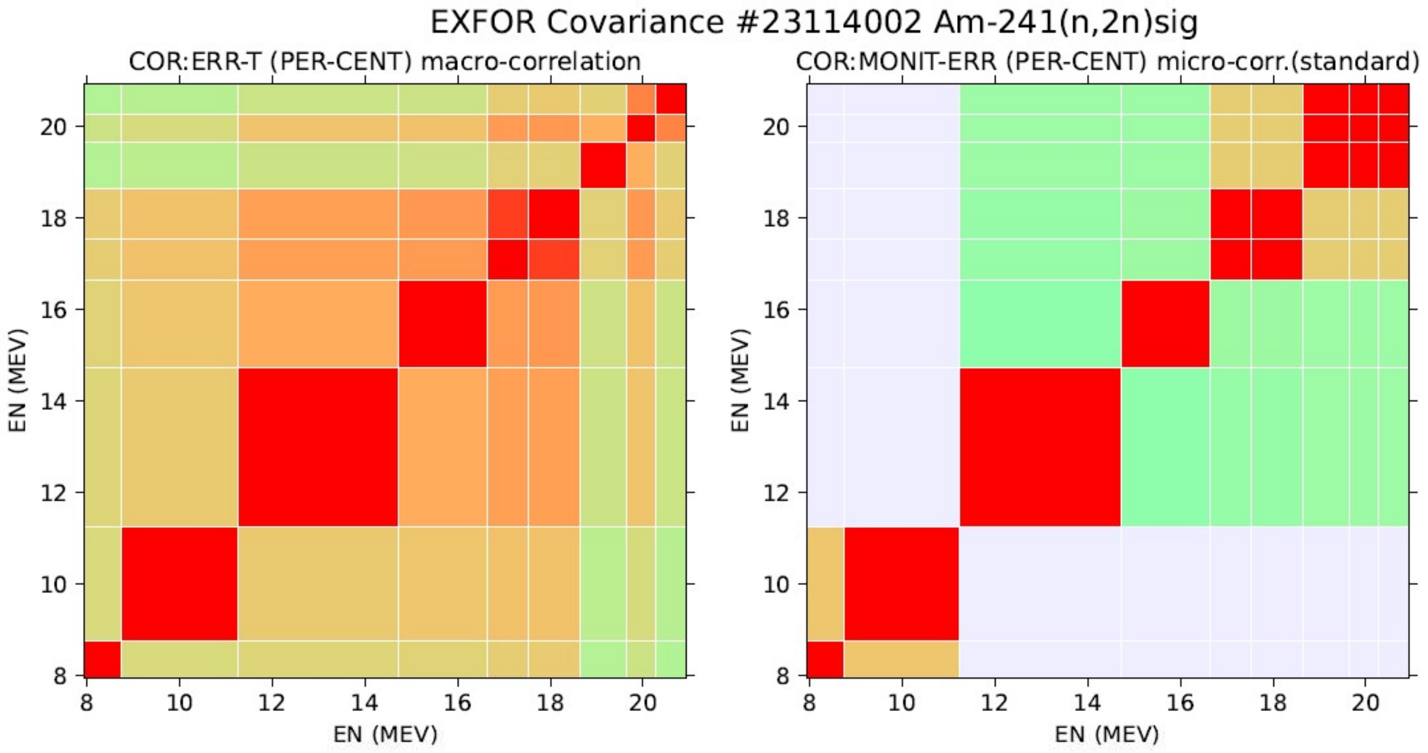
X4pro → Plotly:

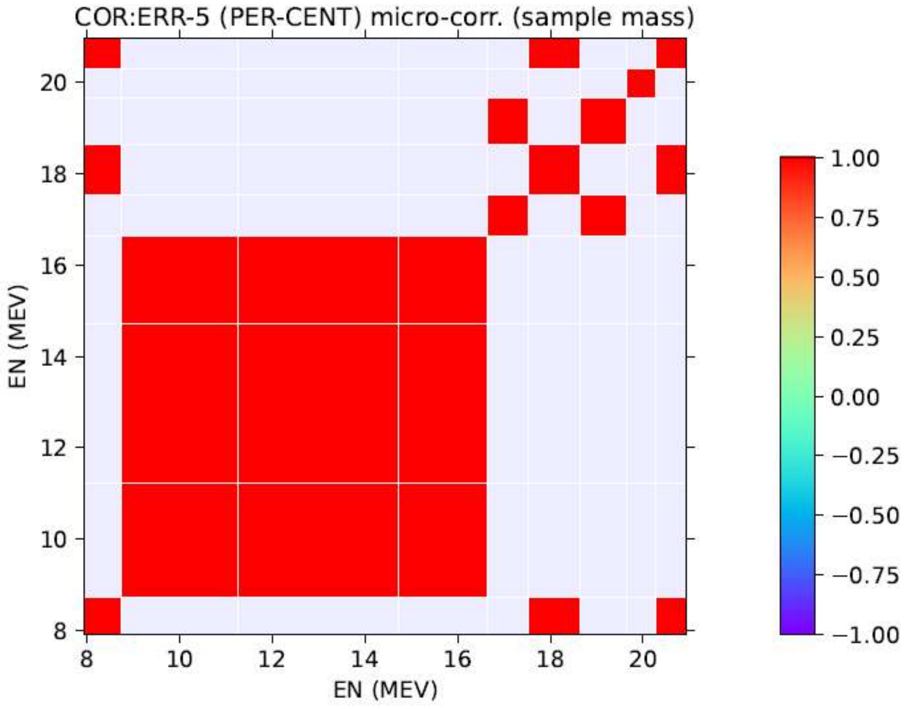

EXFOR #21095008: 92-U-235(N,F)MASS,PR/FRG,NU/TKE //Counts

X4Pro, by V.Zerkin, IAEA-NDS, 2021-12-07--2022-06-20


Matplotlib. EXFOR #21095008: 92-U-235(N,F)MASS,PR/FRG,NU/TKE:Counts

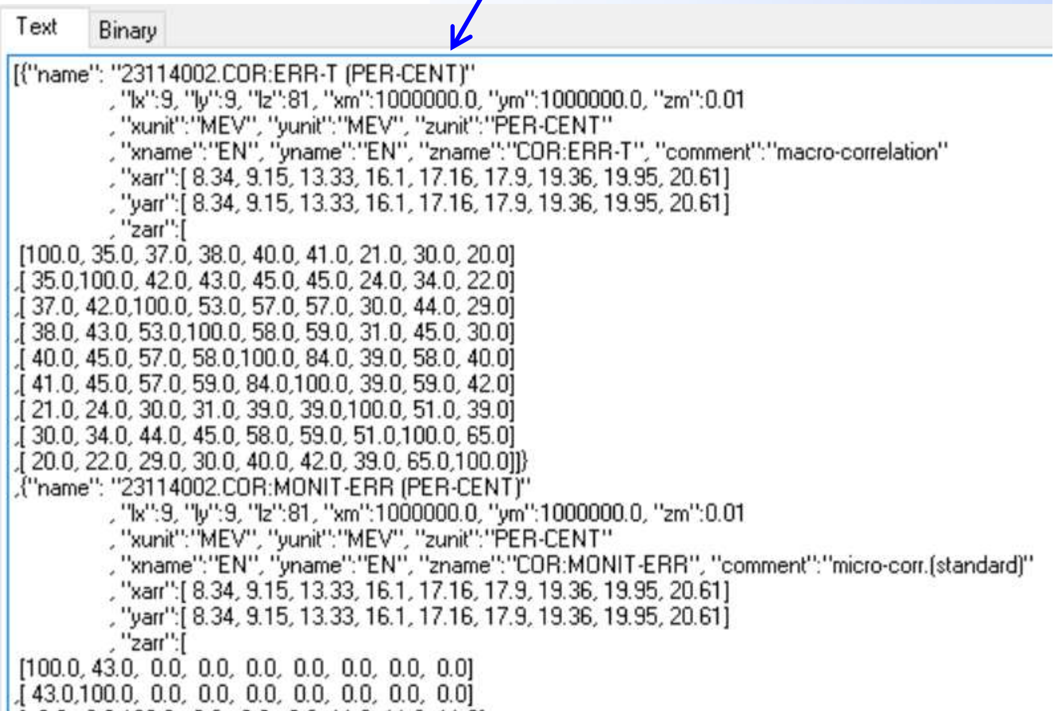
Covariance data coded in EXFOR





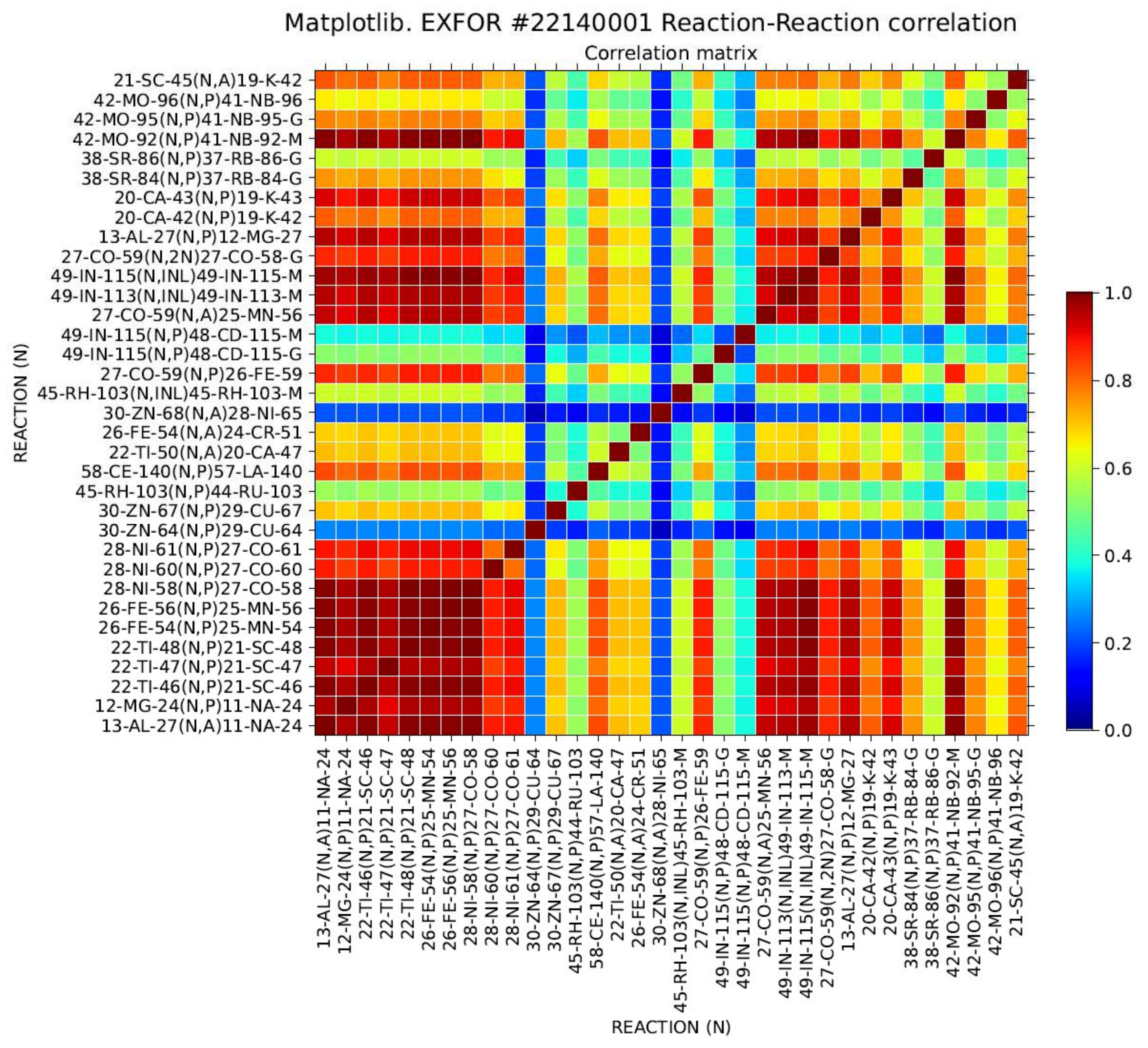
Demo code: python3 + matplotlib

SQL command:
select jFreeData
from x4pro_kw
where DatasetID='23114002’
and Keyword like 'covariance'



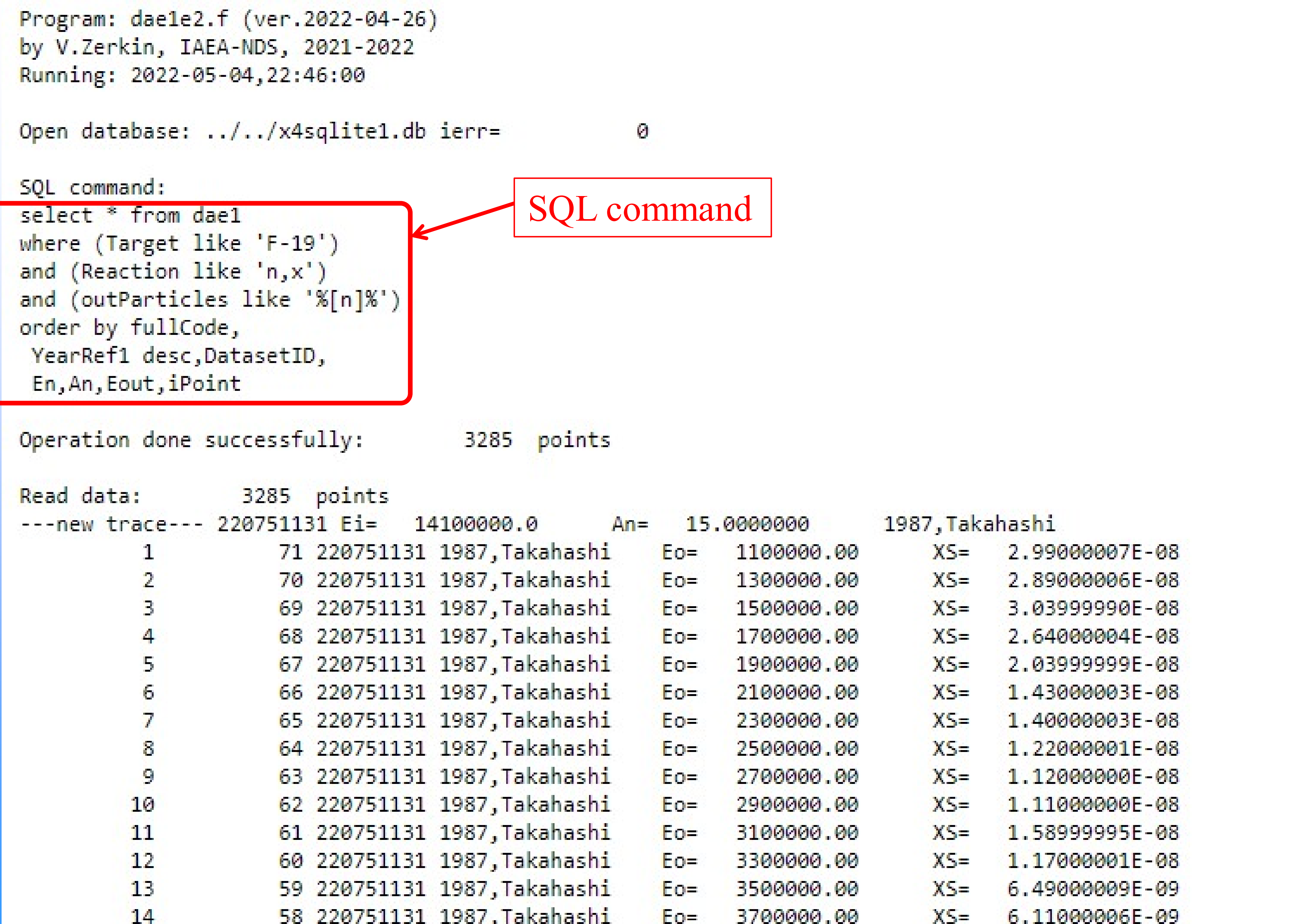
```
[{"name": "23114002.COR:ERR-T (PER-CENT)"
          , "lx":9, "ly":9, "lz":81, "xm":1000000.0, "ym":1000000.0, "zm":0.01
          , "xunit":"MEV", "yunit":"MEV", "zunit":"PER-CENT"
          , "xname":"EN", "yname":"EN", "zname":"COR:ERR-T", "comment":"macro-correlation"
          , "xarr":[ 8.34, 9.15, 13.33, 16.1, 17.16, 17.9, 19.36, 19.95, 20.61]
          , "yarr":[ 8.34, 9.15, 13.33, 16.1, 17.16, 17.9, 19.36, 19.95, 20.61]
          , "zarr":[
 [100.0, 35.0, 37.0, 38.0, 40.0, 41.0, 21.0, 30.0, 20.0]
,[ 35.0,100.0, 42.0, 43.0, 45.0, 45.0, 24.0, 34.0, 22.0]
,[ 37.0, 42.0,100.0, 53.0, 57.0, 57.0, 30.0, 44.0, 29.0]
,[ 38.0, 43.0, 53.0,100.0, 58.0, 59.0, 31.0, 45.0, 30.0]
,[ 40.0, 45.0, 57.0, 58.0,100.0, 84.0, 39.0, 58.0, 40.0]
,[ 41.0, 45.0, 57.0, 59.0, 84.0,100.0, 39.0, 59.0, 42.0]
,[ 21.0, 24.0, 30.0, 31.0, 39.0, 39.0,100.0, 51.0, 39.0]
,[ 30.0, 34.0, 44.0, 45.0, 58.0, 59.0, 51.0,100.0, 65.0]
,[ 20.0, 22.0, 29.0, 30.0, 40.0, 42.0, 39.0, 65.0,100.0]]}
,{"name": "23114002.COR:MONIT-ERR (PER-CENT)"
          , "lx":9, "ly":9, "lz":81, "xm":1000000.0, "ym":1000000.0, "zm":0.01
          , "xunit":"MEV", "yunit":"MEV", "zunit":"PER-CENT"
          , "xname":"EN", "yname":"EN", "zname":"COR:MONIT-ERR", "comment":"micro-corr.(standard)"
          , "xarr":[ 8.34, 9.15, 13.33, 16.1, 17.16, 17.9, 19.36, 19.95, 20.61]
          , "yarr":[ 8.34, 9.15, 13.33, 16.1, 17.16, 17.9, 19.36, 19.95, 20.61]
          , "zarr":[
 [100.0, 43.0,  0.0,  0.0,  0.0,  0.0,  0.0,  0.0,  0.0]
,[ 43.0,100.0,  0.0,  0.0,  0.0,  0.0,  0.0,  0.0,  0.0]
```

Covariance data coded in EXFOR



Matplotlib. EXFOR #22140001 Reaction-Reaction correlation

Double differential cross sections in Fortran



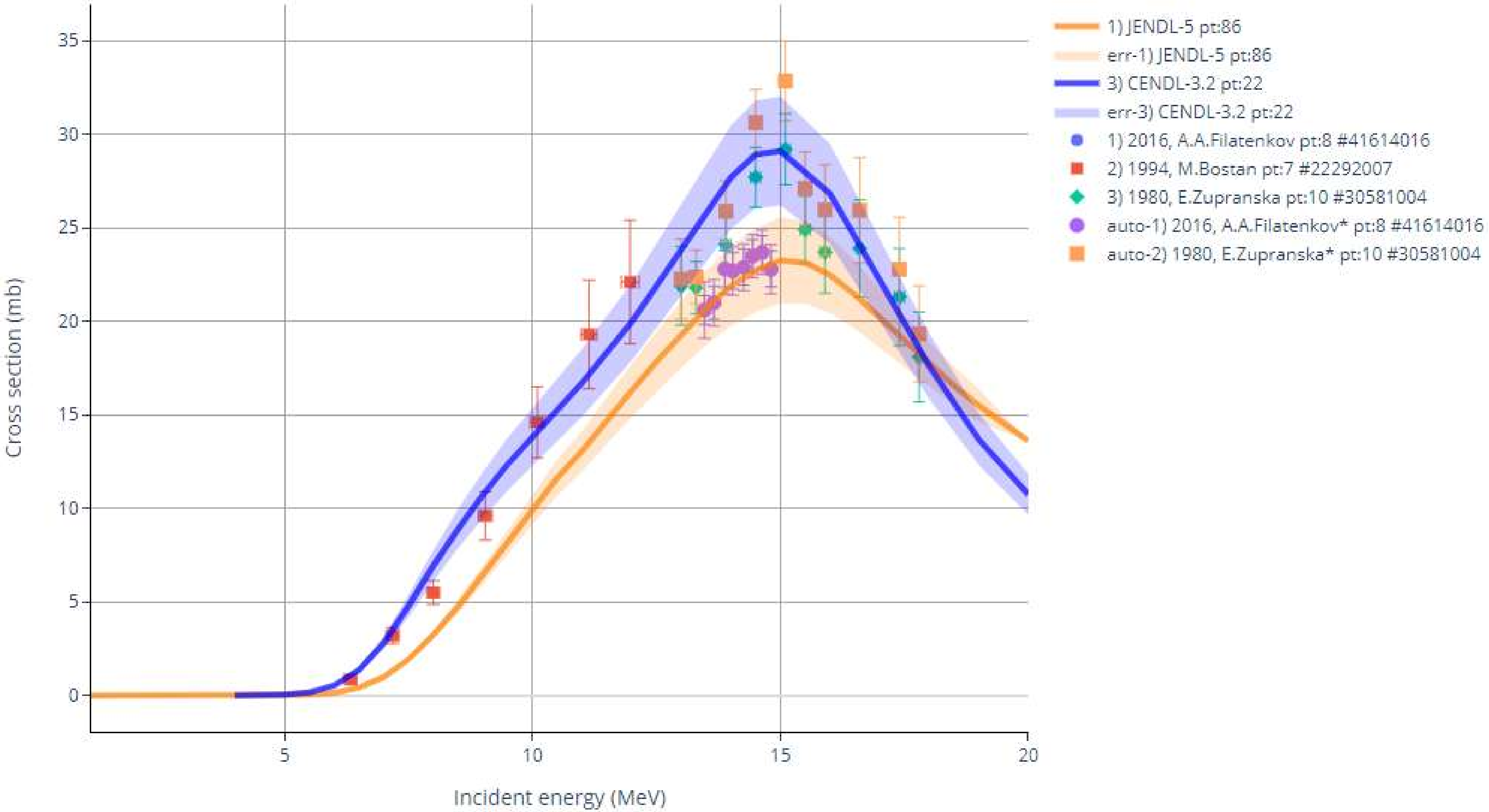
```
Program: daele2.f (ver.2022-04-26)
by V.Zerkin, IAEA-NDS, 2021-2022
Running: 2022-05-04,22:46:00

Open database: ../../x4sqlite1.db ierr=           0

SQL command:
select * from dae1
where (Target like 'F-19')
and (Reaction like 'n,x')
and (outParticles like '%[n]%')
order by fullCode,
 YearRef1 desc,DatasetID,
 En,An,Eout,iPoint

Operation done successfully:        3285  points

Read data:        3285  points
---new trace--- 220751131 Ei=   14100000.0      An=   15.0000000      1987,Takahashi
           1           71 220751131 1987,Takahashi    Eo=   1100000.00      XS=   2.99000007E-08
           2           70 220751131 1987,Takahashi    Eo=   1300000.00      XS=   2.89000006E-08
           3           69 220751131 1987,Takahashi    Eo=   1500000.00      XS=   3.03999990E-08
           4           68 220751131 1987,Takahashi    Eo=   1700000.00      XS=   2.64000004E-08
           5           67 220751131 1987,Takahashi    Eo=   1900000.00      XS=   2.03999999E-08
           6           66 220751131 1987,Takahashi    Eo=   2100000.00      XS=   1.43000003E-08
           7           65 220751131 1987,Takahashi    Eo=   2300000.00      XS=   1.40000003E-08
           8           64 220751131 1987,Takahashi    Eo=   2500000.00      XS=   1.22000001E-08
           9           63 220751131 1987,Takahashi    Eo=   2700000.00      XS=   1.12000000E-08
          10           62 220751131 1987,Takahashi    Eo=   2900000.00      XS=   1.11000000E-08
          11           61 220751131 1987,Takahashi    Eo=   3100000.00      XS=   1.58999995E-08
          12           60 220751131 1987,Takahashi    Eo=   3300000.00      XS=   1.17000001E-08
          13           59 220751131 1987,Takahashi    Eo=   3500000.00      XS=   6.49000009E-09
          14           58 220751131 1987,Takahashi    Eo=   3700000.00      XS=   6.11000006E-09
```

SQL command

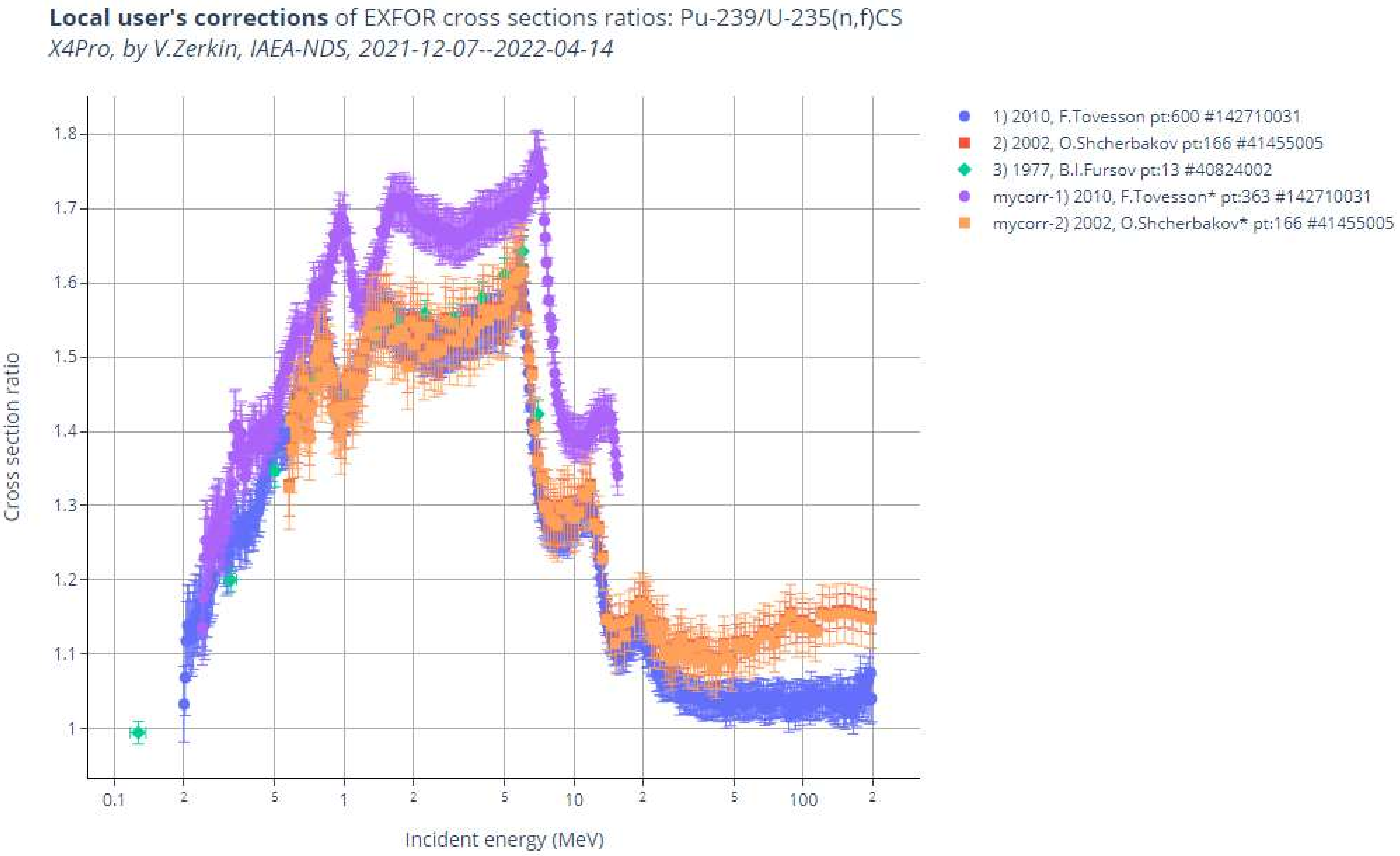
Automatic renormalization in Python

Automatic correction of EXFOR cross sections: Mn-55(n,a),sig

X4Pro, by V.Zerkin, IAEA-NDS, 2021/12/07-2022/03/24



User's modifications of EXFOR data



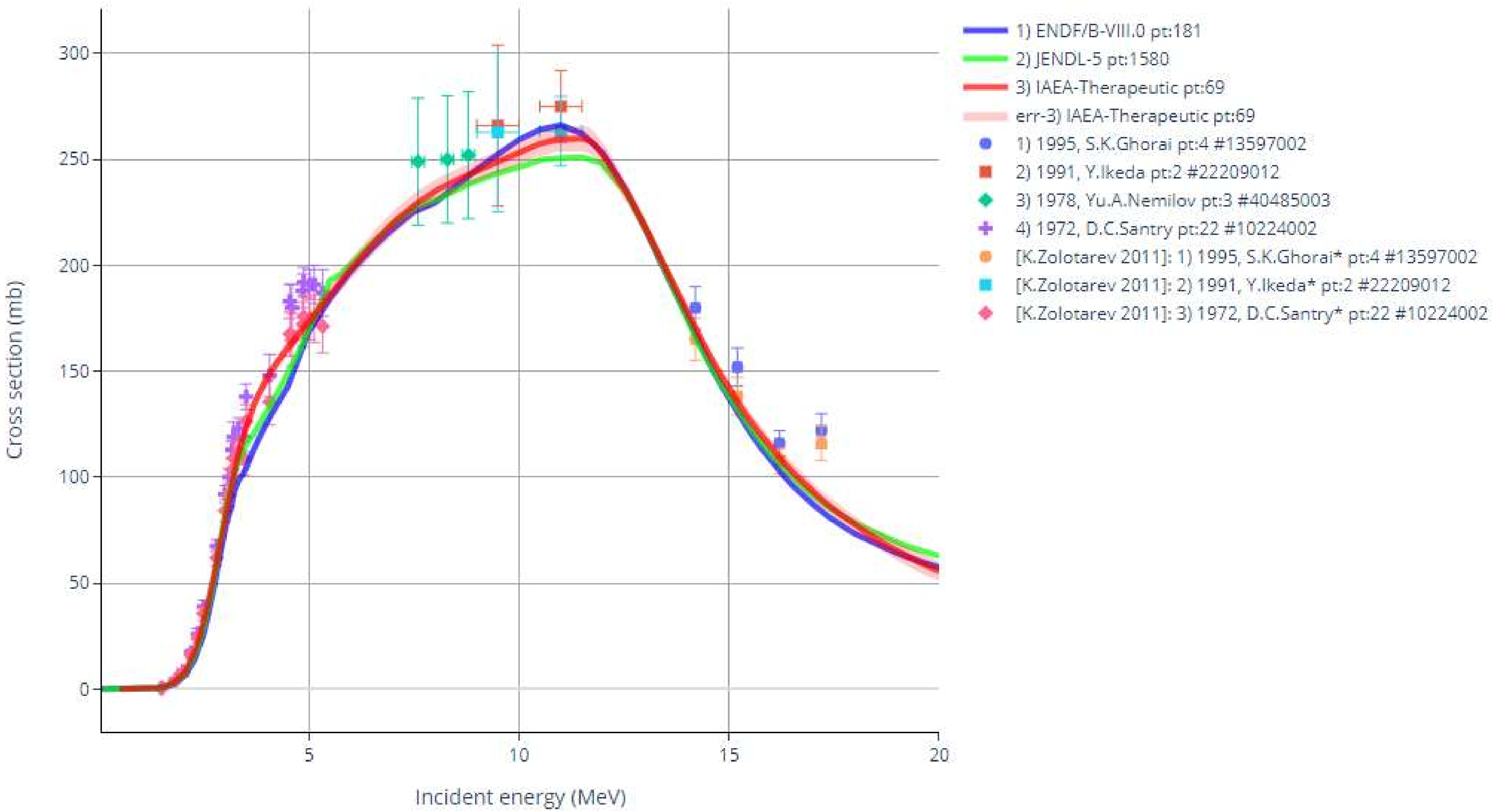
Local user's corrections of EXFOR cross sections ratios: Pu-239/U-235(n,f)CS

X4Pro, by V.Zerkin, IAEA-NDS, 2021-12-07--2022-04-14

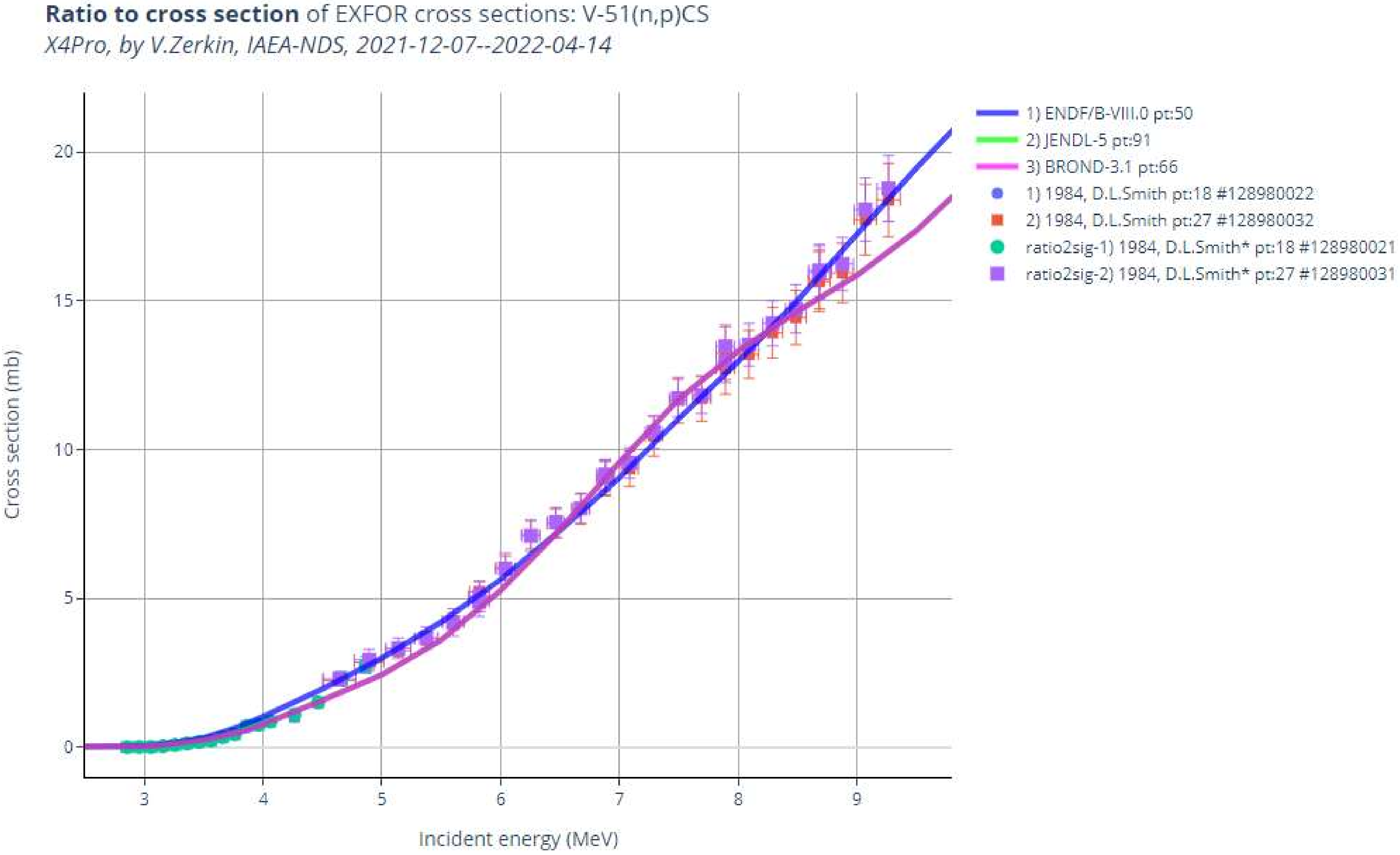
Experts' data corrections in Python

Apply experts corrections from database to EXFOR data: Zn-64(n,p),sig

X4Pro, by V.Zerkin, IAEA-NDS, 2021-12-07--2022-04-14



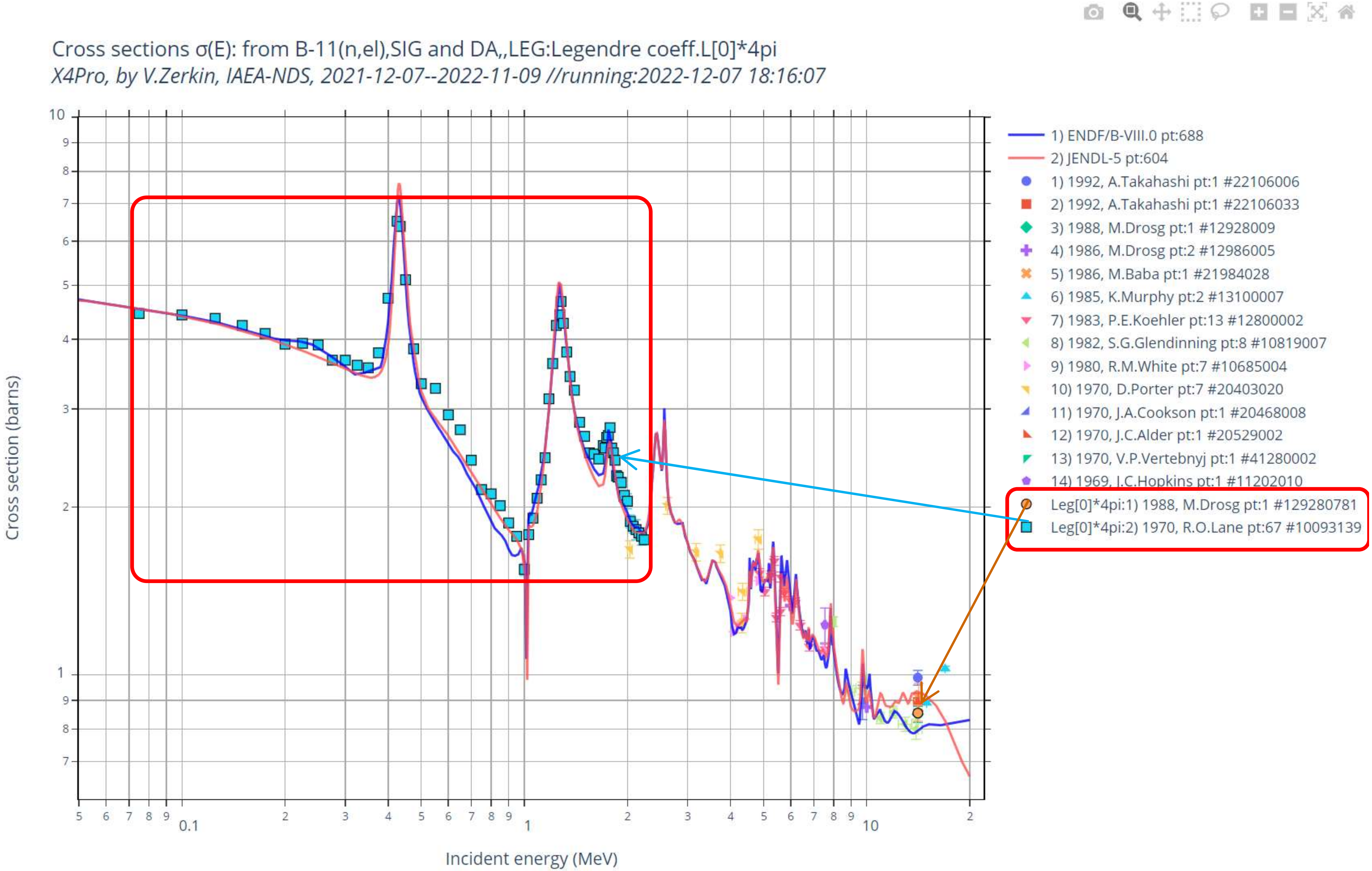
Ratios to cross sections in Python



Ratio to cross section of EXFOR cross sections: V-51(n,p)CS

X4Pro, by V.Zerkin, IAEA-NDS, 2021-12-07--2022-04-14

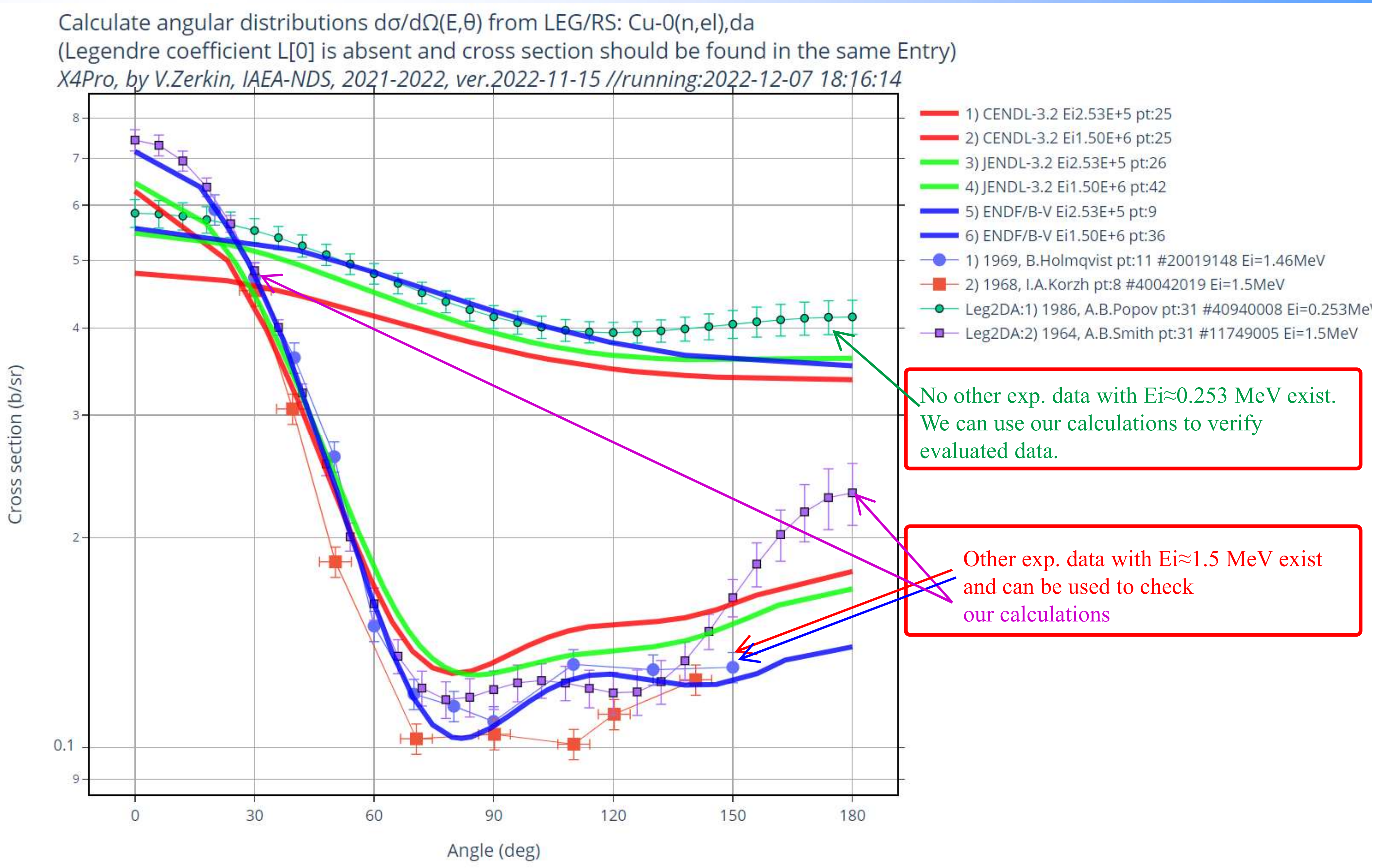
Cross section DATA from [,SIG] together with search/filter/calc.: 4π·L(0) from [,DA,,LEG]



Cross sections σ(E): from B-11(n,el),SIG and DA,,LEG:Legendre coeff.L[0]*4pi

X4Pro, by V.Zerkin, IAEA-NDS, 2021-12-07--2022-11-09 //running:2022-12-07 18:16:07

Retrieve LEG/RS and SIG from the same ENTRY to form L(0..n) and calculate DA

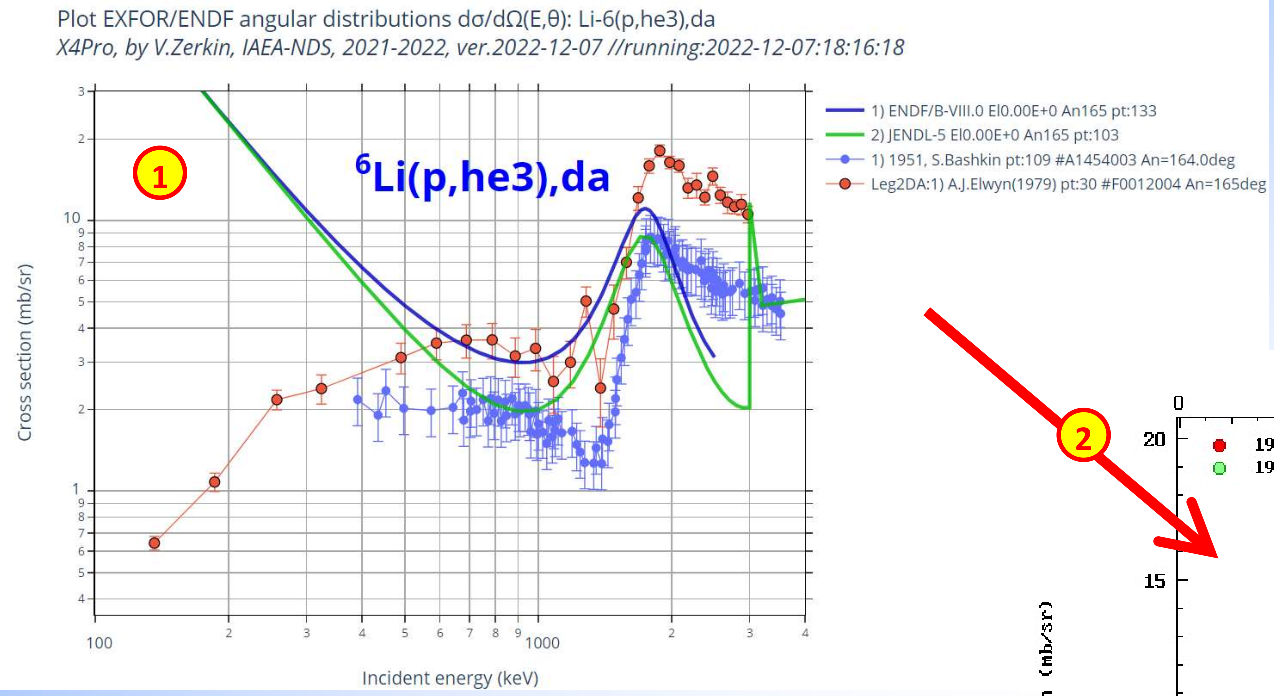


Calculate angular distributions dσ/dΩ(E,θ) from LEG/RS: Cu-0(n,el),da
(Legendre coefficient L[0] is absent and cross section should be found in the same Entry)
X4Pro, by V.Zerkin, IAEA-NDS, 2021-2022, ver.2022-11-15 //running:2022-12-07 18:16:14

No other exp. data with Ei≈0.253 MeV exist. We can use our calculations to verify evaluated data.

Other exp. data with Ei≈1.5 MeV exist and can be used to check our calculations

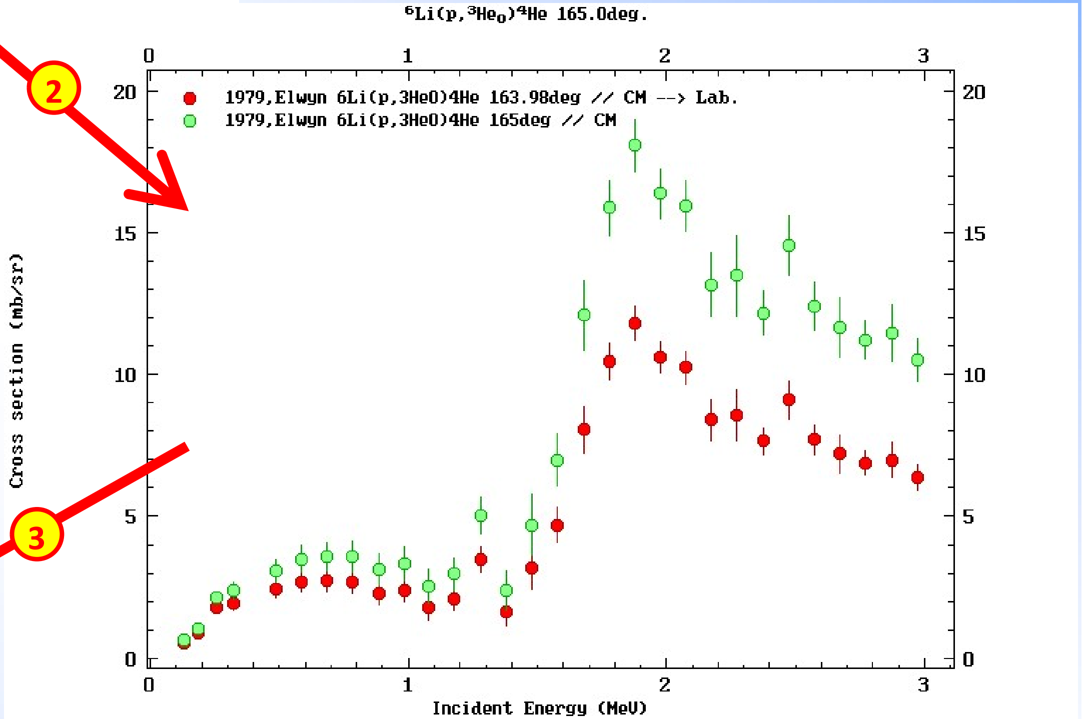
(1) Retrieve LEG(0..n), calculate DA-CM and output to R33, (2) upload to IBANDL, (3) convert to DA-Lab/R33

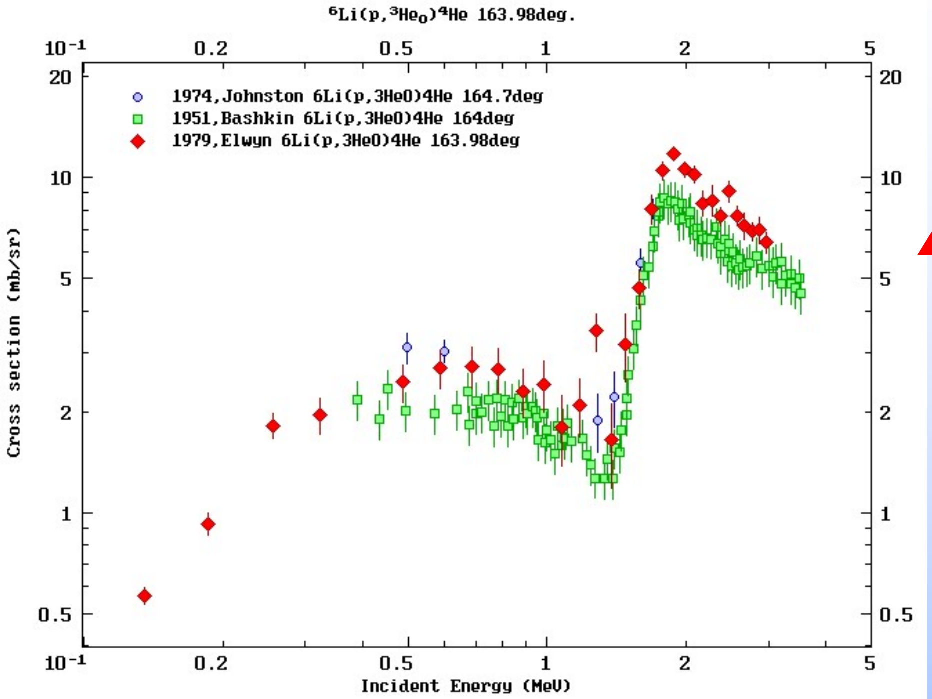

Plot EXFOR/ENDF angular distributions dσ/dΩ(E,θ): Li-6(p,he3),da

X4Pro, by V.Zerkin, IAEA-NDS, 2021-2022, ver.2022-12-07 //running:2022-12-07:18:16:18

#cm2lab: SigFactor=0.607908 0.8800943

#cm2lab: Theta_CM=165.0 Theta_Lab=163.98





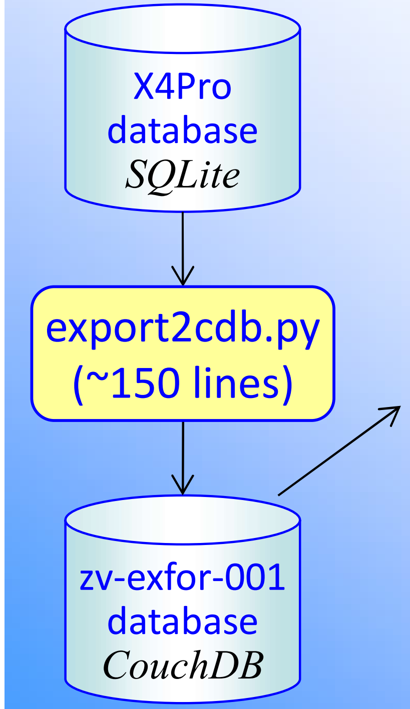
Part IV. X4-JSON, CouchDB

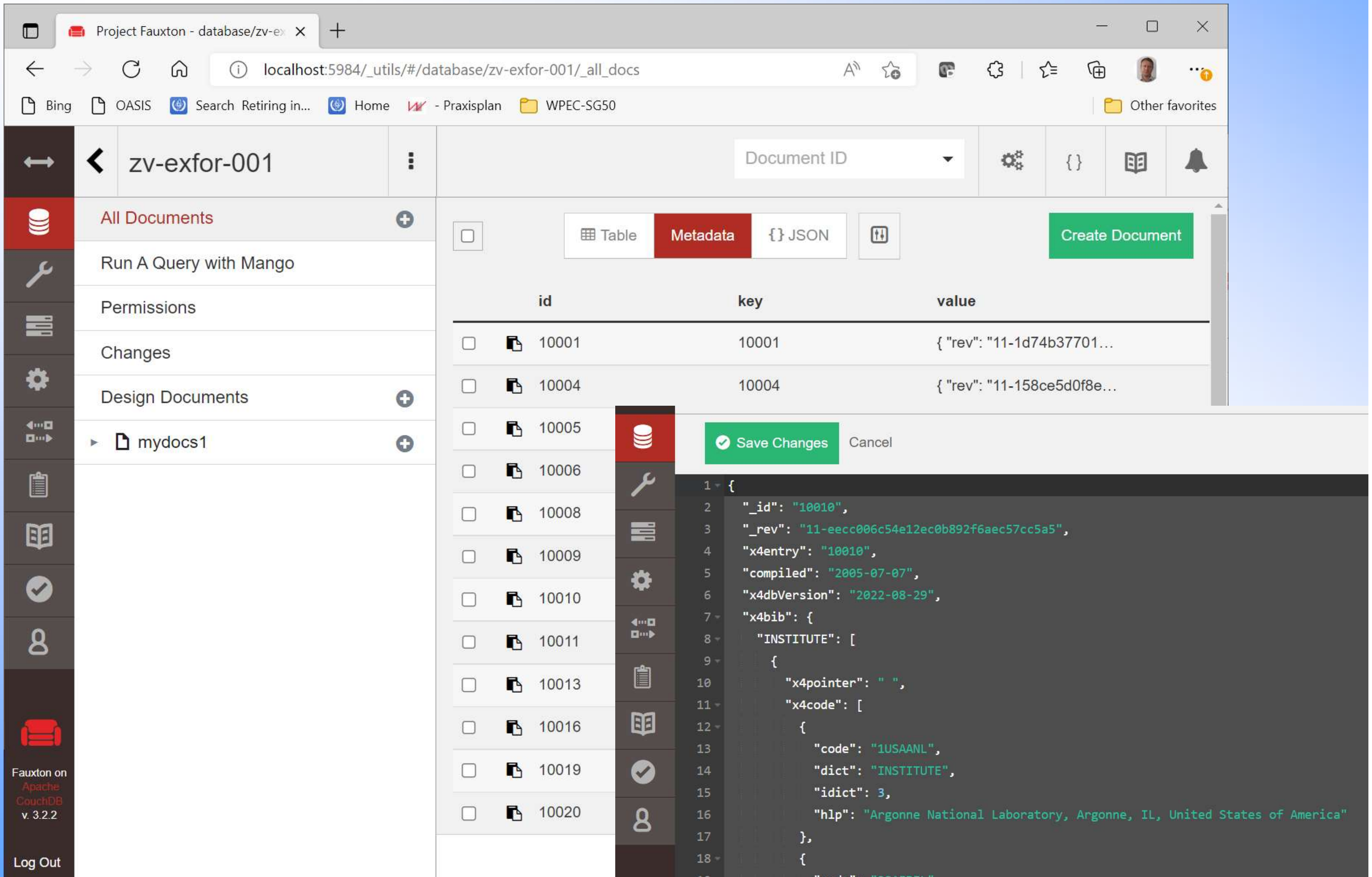
X5-JSON presents meta and numerical data:

1. *from EXFOR and Dictionaries structured as they are in EXFOR - to be useful by compilers*
2. *computational data by Datasets (~C5) including data for automatic correction by new monitor and decay data*

Available on Web-EXFOR as X4Z and X5Z

Example in X4Pro:

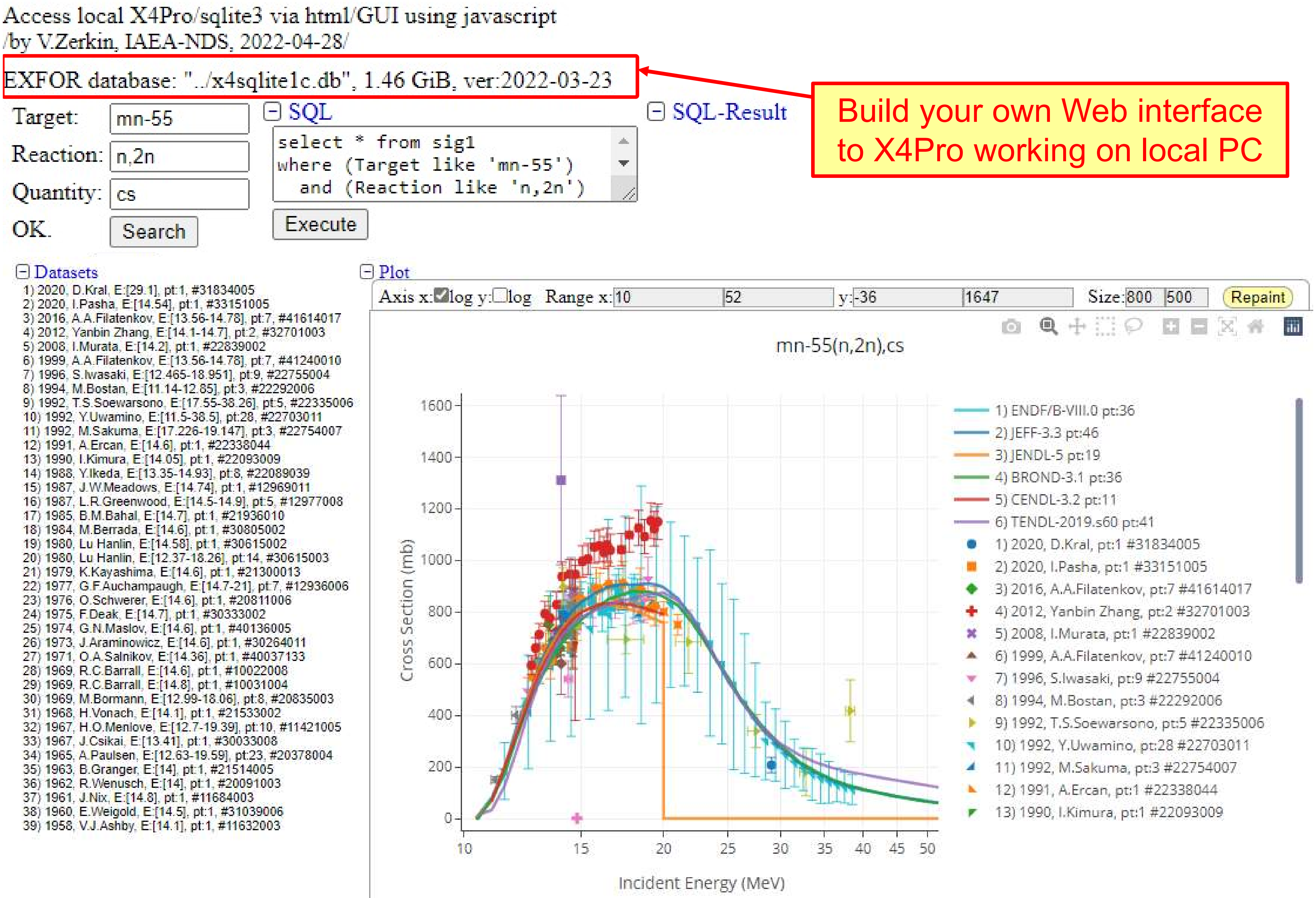




	id	key	value
□	10001	10001	{ "rev": "11-1d74b37701…
□	10004	10004	{ "rev": "11-158ce5d0f8e…
□	10005		
□	10006		
□	10008		
□	10009		
□	10010		
□	10011		
□	10013		
□	10016		
□	10019		
□	10020		

```
{
  "_id": "10010",
  "_rev": "11-eecc006c54e12ec0b892f6aec57cc5a5",
  "x4entry": "10010",
  "compiled": "2005-07-07",
  "x4dbVersion": "2022-08-29",
  "x4bib": {
    "INSTITUTE": [
      {
        "x4pointer": " ",
        "x4code": [
          {
            "code": "1USAANL",
            "dict": "INSTITUTE",
            "idict": 3,
            "hlp": "Argonne National Laboratory, Argonne, IL, United States of America"
          },
          {
```

Part V. Retrieval on JavaScript with GUI/html5



Access local X4Pro/sqlite3 via html/GUI using javascript
/by V.Zerkin, IAEA-NDS, 2022-04-28/

EXFOR database: "../x4sqlite1c.db", 1.46 GiB, ver:2022-03-23

Build your own Web interface to X4Pro working on local PC

Target: mn-55
Reaction: n,2n
Quantity: cs
OK. Search

⊟ SQL

```
select * from sig1
where (Target like 'mn-55')
  and (Reaction like 'n,2n')
```

Execute

⊟ SQL-Result

⊟ Datasets

1) 2020, D.Kral, E:[29.1], pt:1, #31834005
2) 2020, I.Pasha, E:[14.54], pt:1, #33151005
3) 2016, A.A.Filatenkov, E:[13.56-14.78], pt:7, #41614017
4) 2012, Yanbin Zhang, E:[14.1-14.7], pt:2, #32701003
5) 2008, I.Murata, E:[14.2], pt:1, #22839002
6) 1999, A.A.Filatenkov, E:[13.56-14.78], pt:7, #41240010
7) 1996, S.Iwasaki, E:[12.465-18.951], pt:9, #22755004
8) 1994, M.Bostan, E:[11.14-12.85], pt:3, #22292006
9) 1992, T.S.Soewarsono, E:[17.55-38.26], pt:5, #22335006
10) 1992, Y.Uwamino, E:[11.5-38.5], pt:28, #22703011
11) 1992, M.Sakuma, E:[17.226-19.147], pt:3, #22754007
12) 1991, A.Ercan, E:[14.6], pt:1, #22338044
13) 1990, I.Kimura, E:[14.05], pt:1, #22093009
14) 1988, Y.Ikeda, E:[13.35-14.93], pt:8, #22089039
15) 1987, J.W.Meadows, E:[14.74], pt:1, #12969011
16) 1987, L.R.Greenwood, E:[14.5-14.9], pt:5, #12977008
17) 1985, B.M.Bahal, E:[14.7], pt:1, #21936010
18) 1984, M.Berrada, E:[14.6], pt:1, #30805002
19) 1980, Lu Hanlin, E:[14.58], pt:1, #30615002
20) 1980, Lu Hanlin, E:[12.37-18.26], pt:14, #30615003
21) 1979, K.Kayashima, E:[14.6], pt:1, #21300013
22) 1977, G.F.Auchampaugh, E:[14.7-21], pt:7, #12936006
23) 1976, O.Schwerer, E:[14.6], pt:1, #20811006
24) 1975, F.Deak, E:[14.7], pt:1, #30333002
25) 1974, G.N.Maslov, E:[14.6], pt:1, #40136005
26) 1973, J.Araminowicz, E:[14.6], pt:1, #30264011
27) 1971, O.A.Salnikov, E:[14.36], pt:1, #40037133
28) 1969, R.C.Barrall, E:[14.6], pt:1, #10022008
29) 1969, R.C.Barrall, E:[14.8], pt:1, #10031004
30) 1969, M.Bormann, E:[12.99-18.06], pt:8, #20835003
31) 1968, H.Vonach, E:[14.1], pt:1, #21533002
32) 1967, H.O.Menlove, E:[12.7-19.39], pt:10, #11421005
33) 1967, J.Csikai, E:[13.41], pt:1, #30033008
34) 1965, A.Paulsen, E:[12.63-19.59], pt:23, #20378004
35) 1963, B.Granger, E:[14], pt:1, #21514005
36) 1962, R.Wenusch, E:[14], pt:1, #20091003
37) 1961, J.Nix, E:[14.8], pt:1, #11684003
38) 1960, E.Weigold, E:[14.5], pt:1, #31039006
39) 1958, V.J.Ashby, E:[14.1], pt:1, #11632003

⊟ Plot

Axis x:☑log y:☐log Range x:10 52 y:-36 1647 Size:800 500 Repaint

mn-55(n,2n),cs
Cross Section (mb)
1600
1400
1200
1000
800
600
400
200
0
Incident Energy (MeV)
10
15
20
25
30
35
40
45
50
1) ENDF/B-VIII.0 pt:36
2) JEFF-3.3 pt:46
3) JENDL-5 pt:19
4) BROND-3.1 pt:36
5) CENDL-3.2 pt:11
6) TENDL-2019.s60 pt:41
1) 2020, D.Kral, pt:1 #31834005
2) 2020, I.Pasha, pt:1 #33151005
3) 2016, A.A.Filatenkov, pt:7 #41614017
4) 2012, Yanbin Zhang, pt:2 #32701003
5) 2008, I.Murata, pt:1 #22839002
6) 1999, A.A.Filatenkov, pt:7 #41240010
7) 1996, S.Iwasaki, pt:9 #22755004
8) 1994, M.Bostan, pt:3 #22292006
9) 1992, T.S.Soewarsono, pt:5 #22335006
10) 1992, Y.Uwamino, pt:28 #22703011
11) 1992, M.Sakuma, pt:3 #22754007
12) 1991, A.Ercan, pt:1 #22338044
13) 1990, I.Kimura, pt:1 #22093009

Concluding remarks

1. What is X4Pro?

   Extended EXFOR relational database without EXFOR format.
   EXFOR relational database extended with EXFOR data points in original and computational form, data for renormalizing EXFOR data (monitor and decay data) and instructions for data corrections; implemented in MariaDB and SQLite; can be used on server side and on user's PC with Windows, Linux, MacOS.

2. Download X4Pro-trial/SQLite:

   https://www-nds.iaea.org/cdroms/#x4pro1trial

3. Advantages of X4Pro:

   a) universal, flexible, platform-independent, efficient, robust

   b) no need in original EXFOR: all info and data can be taken from the database

   c) no need in EXFOR parsers/converter on user's side

   d) no need for intermediate (C4/C5/JSON) files with fixed structure: application create needed objects on the fly

   e) simple for programming on any programming language supporting SQL for data search, filtering, sorting, retrieval and even renormalization

4. X4Pro status and plans-2022/23:

   a) started public distribution of trial version
   b) presented on NRDC-2022, ND-2022, proposed for testing and feedback
   c) …to take part in EXFOR workshop IAEA-2022 (practicing, feedback)…
   d) to continue development
   e) to coordinate distribution with NRDC-2023

Thank you.