User’s and experts’ data corrections in X4Pro

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.

Introduction EXFOR data renormalization/correction system

EXFOR data correction system (2009+)

(re-normalization system)

Goal:

To create tools for re-calculation absolute values from EXFOR according to

a) today's knowledge (new standards, decay data, abundances)

b) evaluators' experience based on additional information about experiments

Main tasks:

1) to re-normalize data using old monitors and new standards and decay data
2) to create a tool for data modifications: multiply data by a factor, correct wrong units, set up uncertainties, delete part of a data set, recalculate data using isotope abundances, etc.
3) to develop software to generate automatic corrections using EXFOR information
4) to collect experts' corrections to a database

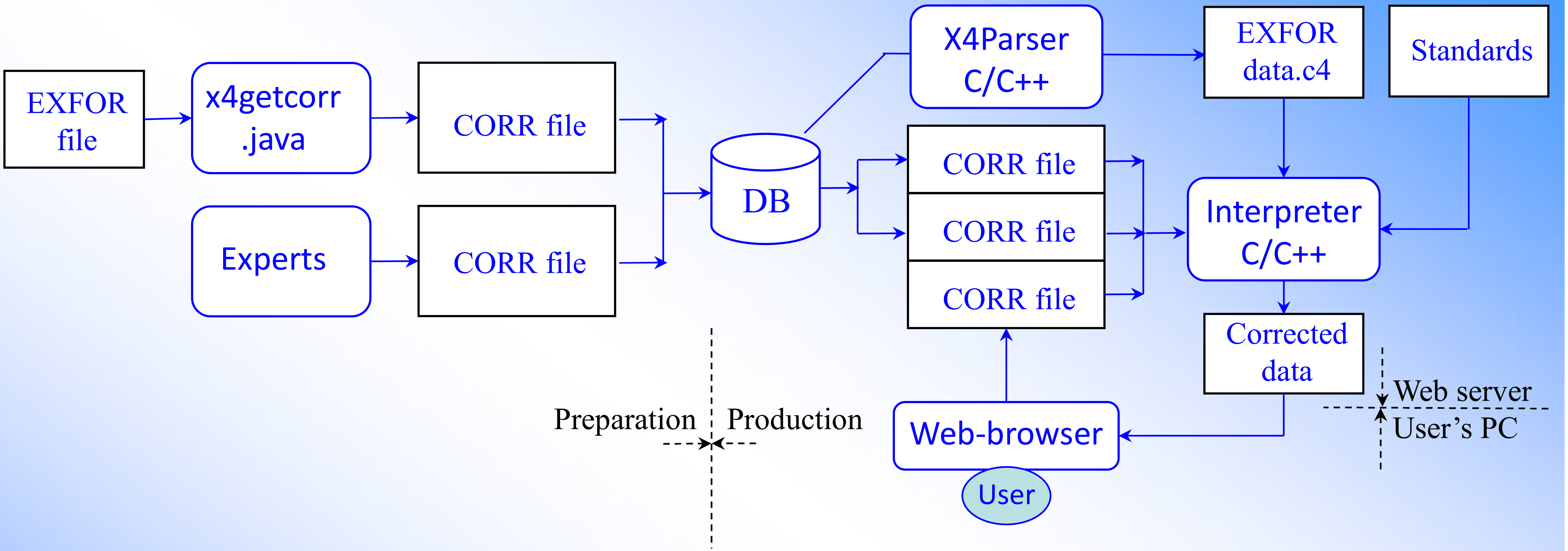
"Automatic" corrections are based on the information given in EXFOR file: keywords MONITOR and MONIT-REF, monitor data in the DATA and COMMON sections. This method is objective.

"Manual" corrections are based user's knowledge and experience – they include subjective judgment.

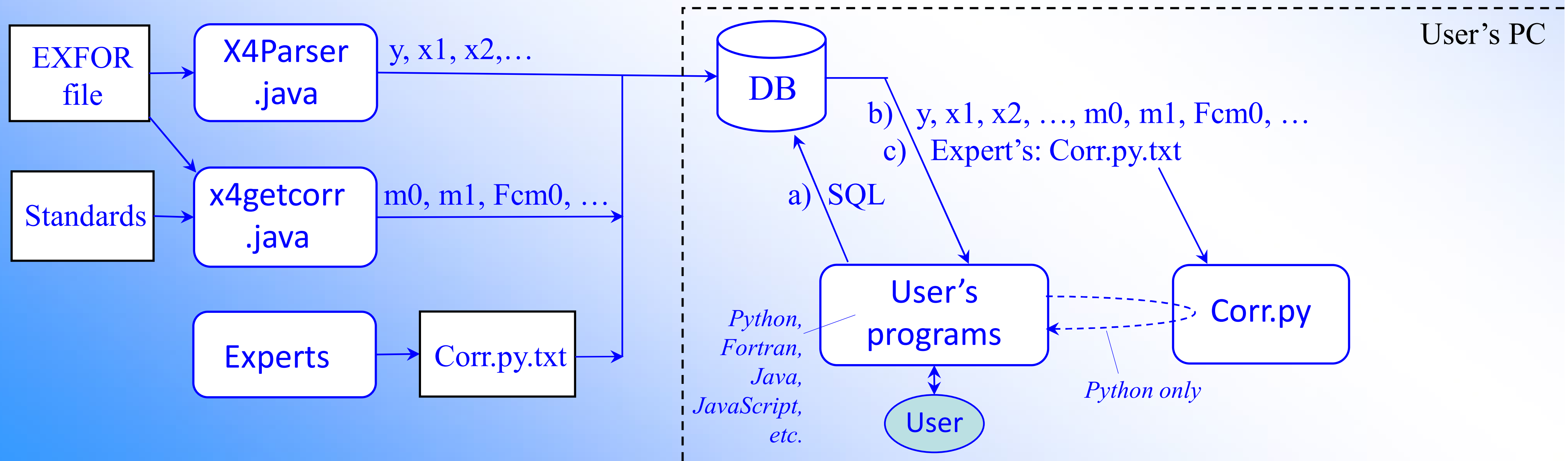
"Experts" corrections: "trusted corrections". NDS is collecting database of experts corrections since 2011.

"Flagging" system. There are files provided from experts and programs detecting whether or not some data were accepted/rejected from by evaluators or programs. This information is stored in the database and used in the Web retrieval system (since 2019).

Now: EXFOR-correction system on Web



X4Pro-corrections: implementation



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/C++)
2. Corr. interpreter (C/C++)
3. Archive of monitors
4. Database of new standards

X4Pro: table x4pro_c5dat

MySQL Query Browser - Connection: / x4mysql5nds

File Edit View Query Script Tools Window MySQL Enterprise Help

Go back Next Refresh

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

Execute Stop

Resultset 1

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)

Edit Apply Changes Discard Changes First Last Search

1: 1

Part II.

User’s data corrections using X4Pro

Example of expert's corrections: ^{239}Pu/^{235}U(n,f)

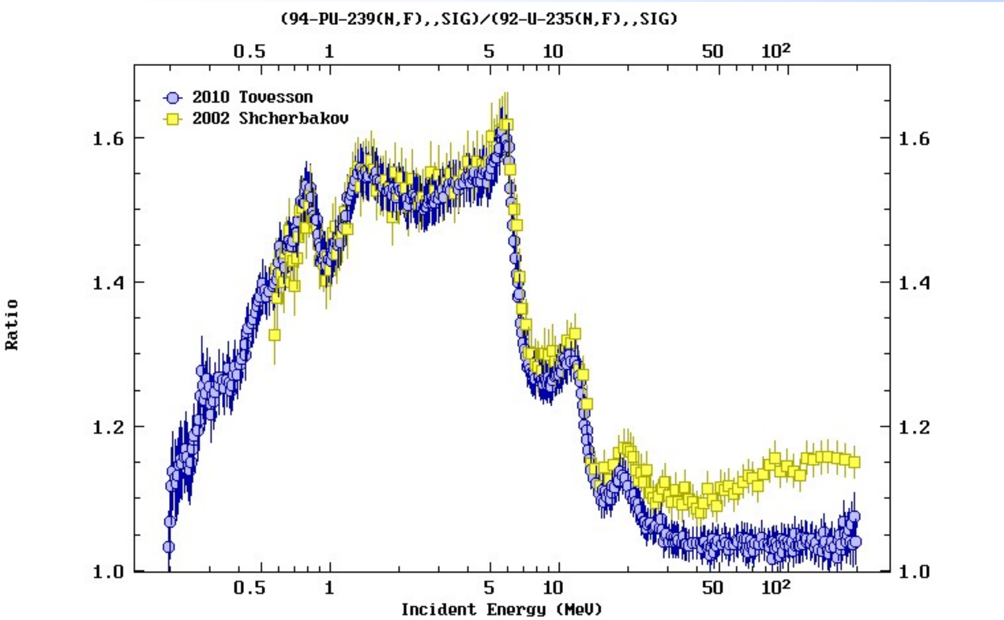
Following [1], we can define tasks:

1. Remove Tovesson's data above 13MeV (1)
2. Renormalize Scherbakov's data and include missing uncertainties (2)
3. Store and share this information between evaluators (3)

[1] D.Neudecker, SG50, 2021-06-21

Solution in EXFOR Web Retrieval system:

```
$C 2021-09-24, V.Zerkin for SG50 2021, 239Pu(n,f)sig/235U(n,f)sig        (3)

142710031  x4u:20201215  #2010,F.Tovesson                                 (1)
#Reaction: (94-PU-239(N,F),,SIG)/(92-U-235(N,F),,SIG)
e:13e6 *; del; #data above 13 MeV rejected in Neutron Standard evaluation (2017)

41455005  x4u:20170724  #2002,O.Shcherbakov                               (2)
# REACTION    ((94-PU-239(N,F),,SIG)/(92-U-235(N,F),,SIG))
# MONITOR     ((94-PU-239(N,F),,SIG)/(92-U-235(N,F),,SIG))
# MONIT-REF   (,,3,JENDL-3.2,,1994)
# COMMENT     Of Authors.
#             The fission cross-section ratio normalization
#             has been done in the 1.75-4.0 MeV energy interval
#             using data of JENDL-3.2.
dy=dy/y;          #convert abs. uncertainty in cs-ratio to rel. uncertainty
a0=1.535;         #used ratio normalization factor (using data of JENDL-3.2), E:1.75-4.0 MeV
c0=1.668/100;     #1.535 +-1.668%  (this uncertainty is not included to error analys)
a1=1.5393;        #ratio normalization factor (using data of ENDF/B-VIII.0), E:1.75-4.0 MeV
c1=2.82/100;      #1.5393 +-2.82%  (uncertainty should be added)
fc=a0/a1;         #total correction factor
y=y*fc;           #correction exp. cs
dy=dy**2+c1**2;   #calc. new quadrature of total uncertainty
dy=dy**0.5*y;     #back to absolute uncertainty
```



Before correction

```
$C 2021-09-24, V.Zerkin for SG50 2021, 239Pu(n,f)sig/235U(n,f)sig
142710031  x4u:20201215  #2010,F.Tovesson
#Reaction: (94-PU-239(N,F),,SIG)/(92-U-235(N,F),,SIG)
e:13e6 *; del; #data above 13 MeV rejected in Neutron Standard evaluation (2017)

41455005  x4u:20170724  #2002,O.Shcherbakov
# REACTION   ((94-PU-239(N,F),,SIG)/(92-U-235(N,F),,SIG))
# MONITOR    ((94-PU-239(N,F),,SIG)/(92-U-235(N,F),,SIG))
# MONIT-REF  (,,3,JENDL-3.2,,1994)
# COMMENT    Of Authors.
#            The fission cross-section ratio normalization
#            has been done in the 1.75-4.0 MeV energy interval
#            using data of JENDL-3.2.
dy=dy/y;        #convert abs. uncertainty in cs-ratio to rel. uncertainty
a0=1.535;       #used ratio normalization factor (using data of JENDL-3.2), E:1.75-4.0 MeV
c0=1.668/100;   #1.535 +-1.668%  (this uncertainty is not included to error analys)
a1=1.5393;      #ratio normalization factor (using data of ENDF/B-VIII.0), E:1.75-4.0 MeV
c1=2.82/100;    #1.5393 +-2.82%  (uncertainty should be added)
fc=a0/a1;       #total correction factor
y=y*fc;         #correction exp. cs
dy=dy**2+c1**2; #calc. new quadrature of total uncertainty
dy=dy**0.5*y;   #back to absolute uncertainty
```

Corrections protocol

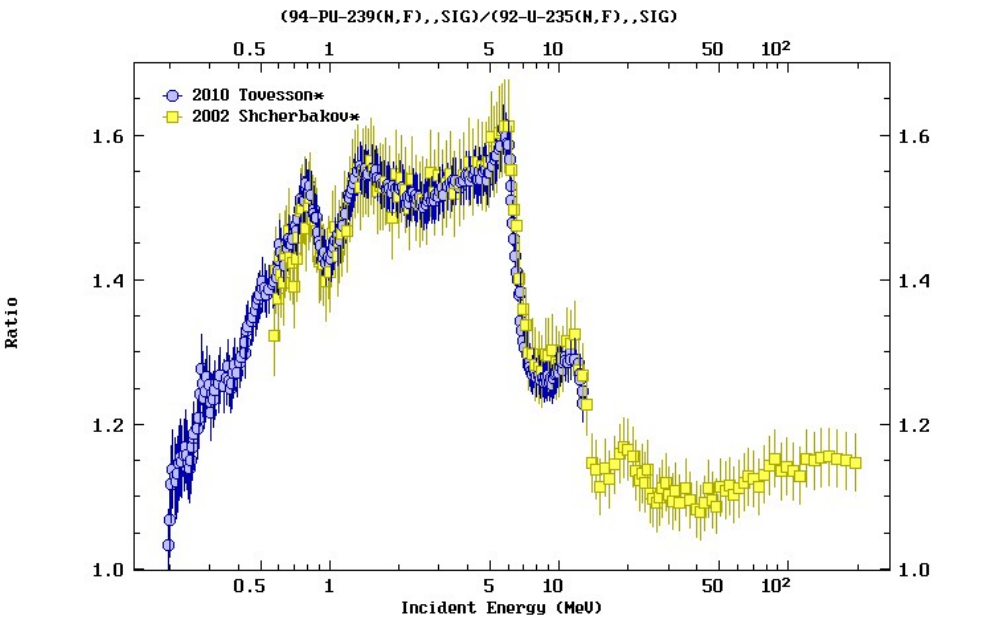
Applied corrections. Datasets: 2

1) EXFOR:#142710031 Ref:F.Tovesson, (10) Corrected_Points:0 Deleted_Points:238 Unchanged_Points:362

2) EXFOR:#41455005 Ref:O.Shcherbakov, (02) Corrected_Points:166 yFactor_Ave:0.997207 yFactor_Min:0.997206 yFactor_Max:0.997207

142710031 X4U:20201215; E:1.3e+7 *; Del;

41455005 X4U:20170724; dY=dY/Y; a0=1.535; c0=1.668/100; a1=1.5393; c1=2.82/100; Fc=a0/a1; Y=Y*Fc; dY=dY^2+c1^2; dY=dY^0.5*Y;



After correction

```
Data check:
 . . . . . . . . . . .  Y(ratio*1000)
-50   En(MeV)=1.773  Y=1541     dY=40.4767 (2.63%)  41455005  O.Shcherbakov,
+50                  Y=1536.7   dY=59.221  (3.85%)  41455005  *Fc=0.997207
-51   En(MeV)=1.829  Y=1531     dY=39.9504 (2.61%)  41455005  O.Shcherbakov,
+51                  Y=1526.72  dY=58.6578 (3.84%)  41455005  *Fc=0.997206
-52   En(MeV)=1.887  Y=1491     dY=38.9867 (2.61%)  41455005  O.Shcherbakov,
+52                  Y=1486.84  dY=57.1796 (3.85%)  41455005  *Fc=0.997207
-53   En(MeV)=1.949  Y=1552     dY=40.7403 (2.63%)  41455005  O.Shcherbakov,
+53                  Y=1547.66  dY=59.6265 (3.85%)  41455005  *Fc=0.997206
 . . . . . . . . . . .
-154  En(MeV)=83.32  Y=1147     dY=23.0894 (2.01%)  41455005  O.Shcherbakov,
+154                 Y=1143.8   dY=39.63   (3.46%)  41455005  *Fc=0.997207
-155  En(MeV)=88.22  Y=1157     dY=23.2452 (2.01%)  41455005  O.Shcherbakov,
+155                 Y=1153.77  dY=39.9491 (3.46%)  41455005  *Fc=0.997206
-156  En(MeV)=93.57  Y=1139     dY=22.965  (2.02%)  41455005  O.Shcherbakov,
+156                 Y=1135.82  dY=39.3748 (3.47%)  41455005  *Fc=0.997206
-157  En(MeV)=99.46  Y=1146     dY=22.6142 (1.97%)  41455005  O.Shcherbakov,
+157                 Y=1142.8   dY=39.3335 (3.44%)  41455005  *Fc=0.997206
```

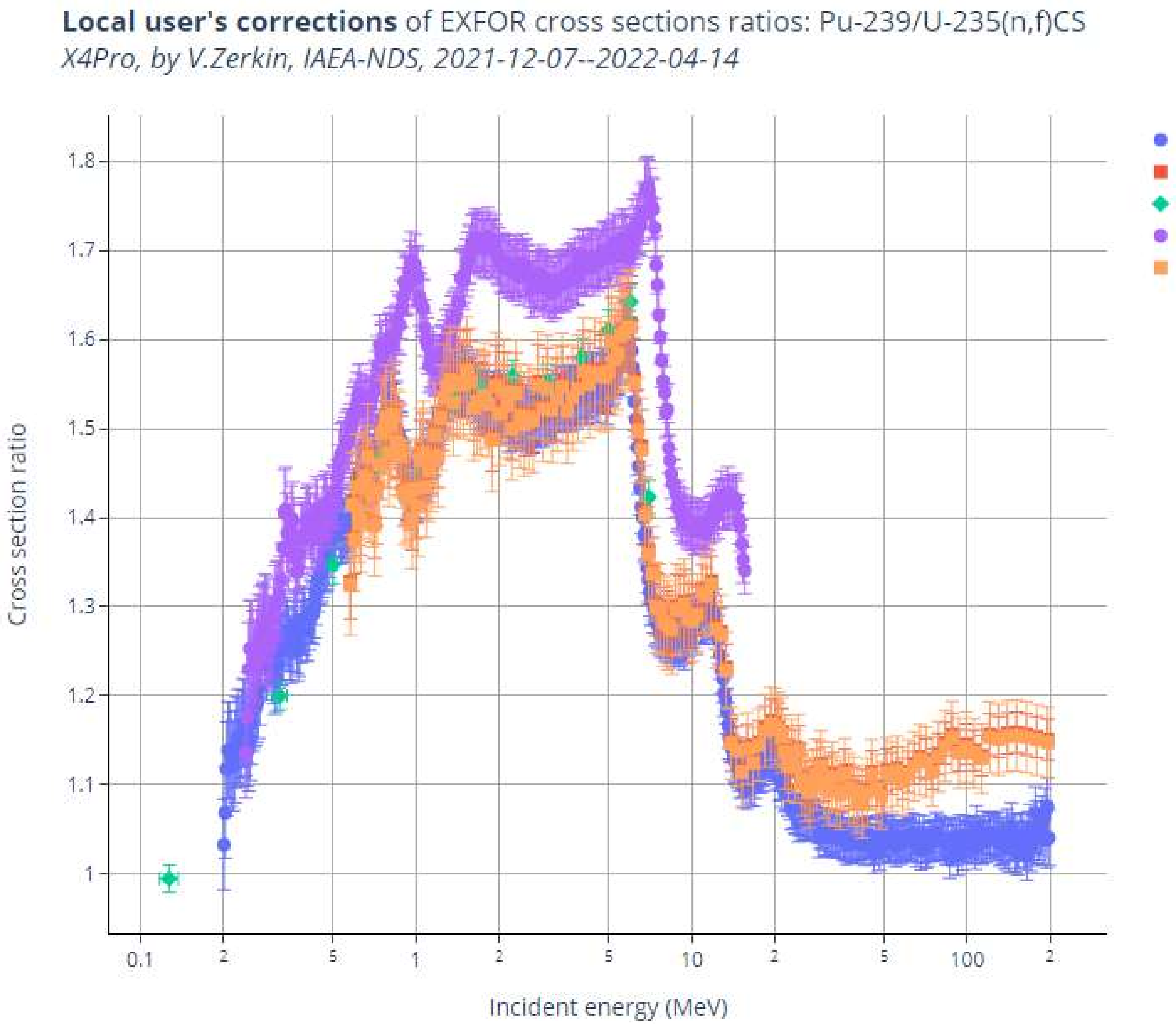
X4Pro: user's data modifications by Python

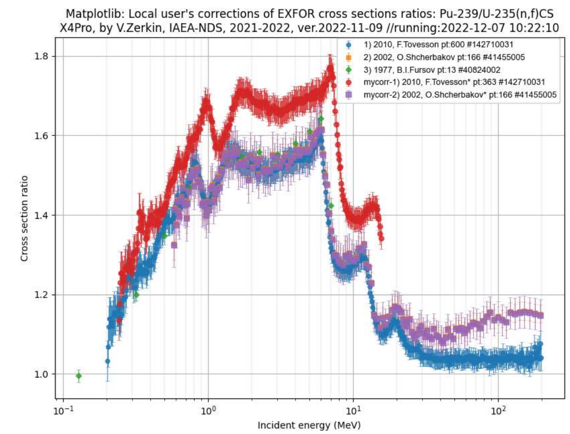
```
import sys
sys.path.append('./')
from myCorr1 import *

myFuncs={
          '142710031':fcorr_142710031
         ,'41455005' :fcorr_41455005
}

def corr_dataset(rows,cursor):
   newrows=[]
   if len(rows)<=0: return newrows
   row=rows[0]
   DatasetID=row['DatasetID']
   print("____________corr_dataset:::["
   +DatasetID+']'+' len='+str(len(rows)))
   myFunc=myFuncs.get(DatasetID)
   if myFunc is None: return newrows
   for row in rows:
      #print(row)
      iupd=myFunc(row)
      if iupd>0:
         newrows.append(row);
         row['corrected']=1
   return newrows
```

Example:
part3-2-user1/

```
#file: myCorr1.py
def fcorr_142710031(row):              #2010,F.Tovesson
  #Reaction: (94-PU-239(N,F),,SIG)/(92-U-235(N,F),,SIG)
  ene=row['En']
  if ene>13e6:
    print("___fcorr_142710031:"+' En='+str(row['En']/1e6)+'MeV'+' -REJECTED-')
    return -1                #data above 13 MeV rejected in Neutron Standard evaluation (2017)
  row['En']=row['En']*1.2               #just for example
  row['y']=row['y']*1.1                 #just for example
  #print("___fcorr_142710031:"+' En='+str(row['En']/1e6)+'MeV'+' -ACCEPTED-')
  return 1                   #updated == accepted

def fcorr_41455005(row):               #x4u:20170724 #2002,O.Shcherbakov
  # REACTION   ((94-PU-239(N,F),,SIG)/(92-U-235(N,F),,SIG))
  # MONITOR    ((94-PU-239(N,F),,SIG)/(92-U-235(N,F),,SIG))
  # MONIT-REF  (,,3,JENDL-3.2,,1994)
  # COMMENT    Of Authors.
  #          The fission cross-section ratio normalization
  #          has been done in the 1.75-4.0 MeV energy interval
  #          using data of JENDL-3.2.
  y=row['y']
  dy=row['dy']
  dy=dy/y;                   #convert abs. uncertainty in cs-ratio to rel. uncertainty
  a0=1.535;                  #used ratio normalization factor (using data of JENDL-3.2), E:1.75-4.0 MeV
  c0=1.668/100;              #1.535 +-1.668% (this uncertainty is not included to error analys)
  a1=1.5393;                 #ratio normalization factor (using data of ENDF/B-VIII.0), E:1.75-4.0 MeV
  c1=2.82/100;               #1.5393 +-2.82% (uncertainty should be added)
  fc=a0/a1;                  #total correction factor
  y=y*fc;                    #correction exp. cs
  dy=dy**2+c1**2;            #calc. new quadrature of total uncertainty
  dy=dy**0.5*y;              #back to absolute uncertainty
  print("____________fcorr_41455005:"+' En='+str(row['En'])
           +' y0='+str(row['y'])+' y1='+str(round(y,5))+' Fc='+str(round(fc,5))
           +' dy0='+str(row['dy'])+'dy1='+str(round(dy,5)))
  row['y']=y                 #save y
  row['dy']=dy               #save dy
  return 1                   #update
```

User's modifications of EXFOR data





Part III.

Experts' data corrections in X4Pro: usage and sharing via database

Expert's data corrections

Expert's correction is stored in database as python text (table x4pro_expertcorr, e.g. 10224002)
Main Python program loads `strcorr` from x4pro_expertcorr to working file "corr_subr.py", reload subroutine and calls it in the loop on all data points.

```
load_corr_subr('corr_subr.py',DatasetID,cursor,row)
reload(corr_subr)
```

```
select * from x4pro_expertcorr
```

	DatasetID	author	itype	fileID	fileDate	dbDate	**strcorr**
1	13597002	K.Zolotarev 2011	x4expert	6	2016-09-22	2011-05-16	...
2	22209012	K.Zolotarev 2011	x4expert	6	2016-09-22	2011-05-16	...
3	10224002	K.Zolotarev 2011	x4expert	6	2016-09-22	2001-03-09	...

```
Execution finished without errors.
Result: 3 rows returned in 5ms
At line 1:
select * from x4pro_expertcorr
```

```

def corr_point(row):
    y  =row["y"]
    dy =row["dy"]
    if dy is None: dy=0
            #[K.Zolotarev 2011]
    #10224002   #1972 D.C.Santry+
            #measurements with T(p,n)He3 neutron source
            #monitor BF3 long counter
    a0=0.91582; #experimental data were renormalized to the integral of
            #cross-section calculated from experimental data of Mannhart
            #and Schmidt 2007 in the overlapping energy
            #range 1.500 - 3.958 MeV, a0=0.91582
    a1=0.0115;  #error in b+ mode in Cu64 decay - 1.15%
    a2=0.03;    #error in normalization value        - 3%
    a3=0.03;    #error in angular neutron intensity - 3%
            #error in the cs data due to the error in En center pozition
            #of 0.17 - 20.09% are not taken into accout
    dy=dy/y;    #relative uncertainty in original cs for Zn64(n,p)Cu64 reaction
    fc=a0;      #total correction factor
    y=y*fc;     #correction exp. cs
    dy=dy**2+a1**2+a2**2+a3**2; #determination the quadrature of new total error
    dy=dy**0.5*y; #determination the absolute error in new Zn64(n,p) cs
    row["y"]=y
    row["dy"]=dy
    return 1
```

Implementation of experts' corrections stored in the database x4pro_expertcorr, retrieved and used "on the fly"

File: expert_corr.py

```
from importlib import reload
import corr_subr

def corr_dataset(rows,cursor):
    newrows=[]
    if len(rows)<=0: return newrows
    row=rows[0]
    DatasetID=row['DatasetID']
    print("___________corr_dataset:::"+DatasetID+' ldat:'+str(len(rows)))
    load_corr_subr('corr_subr.py',DatasetID,cursor,row)
    reload(corr_subr)
    for row in rows:
        str0=DatasetID+' En='+str(row['En'])+' y0='+str(round(row['y'],5))
        ierr=corr_subr.corr_point(row)
        if ierr>0:
            newrows.append(row);
            row['corrected']=1
            str0=str0+'   y1='+str(round(row['y'],5))
                 +'       dy1='+str(round(row['dy'],5))
            print("___________corr_data:::"+str0)
    return newrows
```

File: expert_corr.py (cont.)

```
def load_corr_subr(file_py,DatasetID,cursor,datarow):
    sql="SELECT strcorr,author from x4pro_expertcorr \n\
         WHERE (DatasetID like '"+DatasetID+"')      \n\
         ORDER BY fileDate desc"
    try:
        cursor.execute(sql)
        rows=cursor.fetchall()
    except Exception as ex:
        print("___1___execute-SQL error: "+str(ex)+"\n"+sql)
        rows=[]
    strcorr=''
    if len(rows)>0:
        row=rows[0]
        strcorr=row['strcorr']
        author=row['author']
        datarow['corr_author']=author
    else:
        strcorr="\
def corr_point(row):\n\
    return 0 #unchanged\n\
"

    my_file=open(file_py,'w')
    my_file.write("#Dataset:"+DatasetID+"\n")
    my_file.write(strcorr)
    my_file.write("\n")
    my_file.close()
    print("________load_corr_subr:::"+DatasetID+"\n"+strcorr)
```

Corrections for 10224002:[K.Zolotarev, 2011]

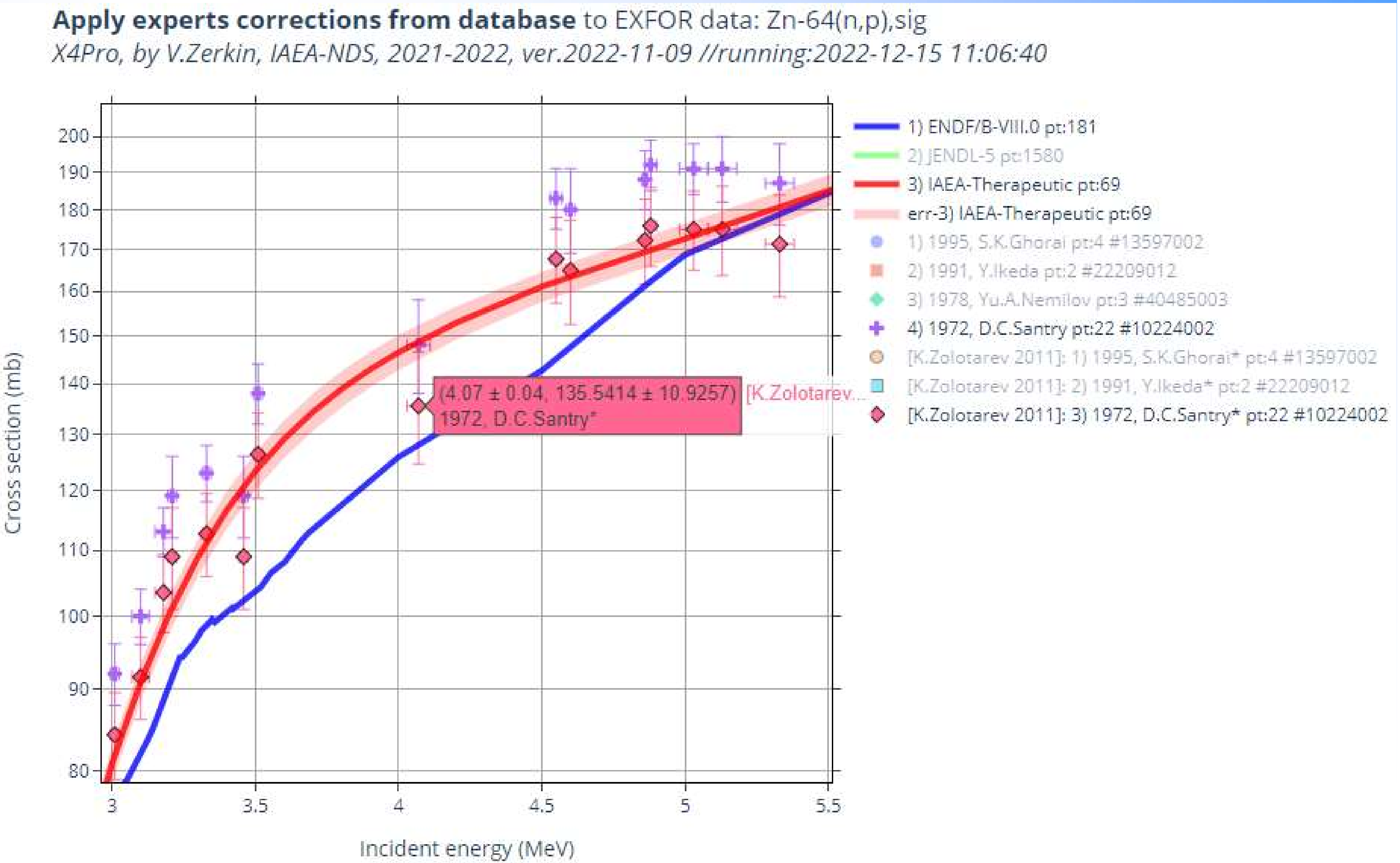
Additions for Python implementation

Corrections in “my-language”

```
def corr_point(row):
    y  =row["y"]
    dy =row["dy"]
    if dy is None: dy=0
                #[K.Zolotarev 2011]
    #10224002   #1972 D.C.Santry+
                #measurements with T(p,n)He3 neutron source
                #monitor BF3 long counter
    a0=0.91582; #experimental data were renormalized to the integral of
                #cross-section calculated from experimental data of Mannhart
                #and Schmidt 2007 in the overlapping energy
                #range 1.500 - 3.958 MeV, a0=0.91582
    a1=0.0115;  #error in b+ mode in Cu64 decay - 1.15%
    a2=0.03;    #error in normalization value       - 3%
    a3=0.03;    #error in angular neutron intensity - 3%
                #error in the cs data due to the error in En center pozition
                #of 0.17 - 20.09% are not taken into accout
    dy=dy/y;    #relative uncertainty in original cs for Zn64(n,p)Cu64 reaction
    fc=a0;      #total correction factor
    y=y*fc;     #correction exp. cs
    dy=dy**2+a1**2+a2**2+a3**2; #determination the quadrature of new total error
    dy=dy**0.5*y; #determination the absolute error in new Zn64(n,p) cs
    row["y"]=y
    row["dy"]=dy
    return 1
```

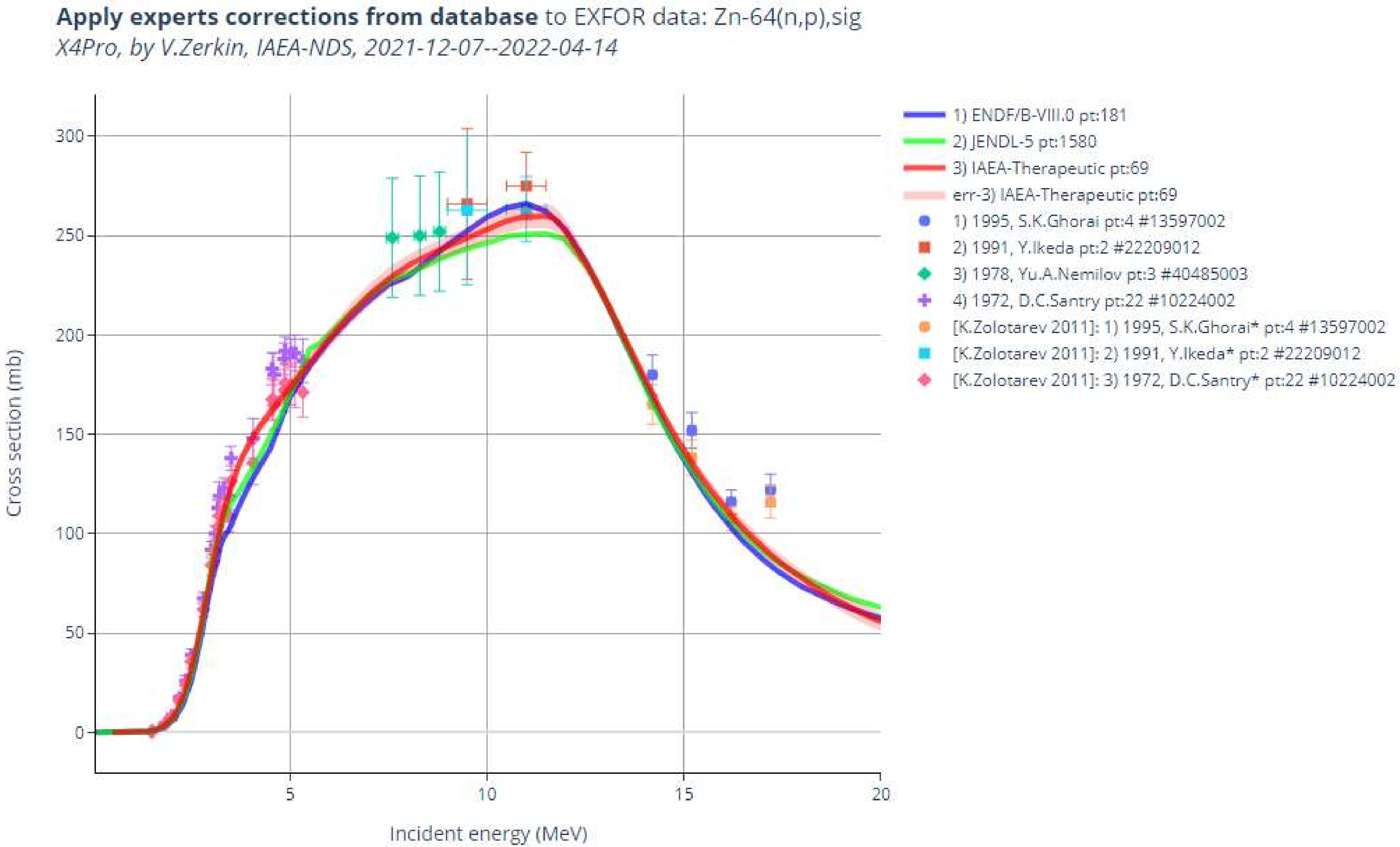
Experts' corrections in Python

example for 10224002: [K.Zolotarev, 2011]



Experts' corrections in Python

example in part3-2-user1/



Concluding remarks

1. X4Pro provides infrastructure for storage/sharing automatic and experts’ data corrections
2. X4Pro provides examples of implementing automatic and users data corrections on SQL level and experts’ data corrections using Python coding
3. Future potential problem [?]
   Experts’ corrections on Web using Python instructions

Thank you.