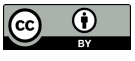
Applying Random Sampling methods to data analysis for uncertainty production, with an Open source and Open science outlook

Greg Henning



Université de Strasbourg

UNIVERSITÉ DE STRASBOURG

Collège Doctoral

Applying Random Sampling methods to data analysis for uncertainty production, with an Open source and Open science outlook.

Manuscript submitted to the jury for the Habilitation à Diriger les Recherches examination.

By Greg Henning,

Defended on June 27th, 2024, in front of the jury:

- Arnd Junghans (HZRD), referee
- Emmeric Dupont (CEA), referee
- Nicolas Arbor (IPHC), referee
- Anne Ruimy (EDP Science)
- Sandrine Courtin (IPHC)
- Maëlle Kerveno (IPHC), guarantor

Acknowledgments

This work was funded in part by the CNRS multipartner NEEDS program, PACEN/GEDEPEON, and by the European Commission within the Sixth Framework Program through I3-EFNUDAT (EURATOM contract n◦ 036434) and NUDAME (Contract FP6-516487), within the Seventh Framework Program through EUFRAT (EURATOM contract n◦ FP7-211499), through ANDES (EURATOM contract n◦ FP7-249671), from the Euratom research and training program 2014-2018 under grant agreement n◦ 847594 (ARIEL) and under grant agreement n◦ 847552 (SANDA).

CONTENT:

By Greg Henning

Warning: The preferred way to read this document is as a webpage.

Disclaimer

The Large Language Model predictor GPT (versions 3.5 and 4) from OpenAI have been used to proofread part of this document. The texts, data and figures are the product of my own work. The ideas and conclusions are my own.

Note: For clarity, the text has been written by following the New York Times elements of style[1] as much as possible (for example, acronyms are spelled as a proper nouns, initialisms in full upper case).

Warning: As of July 2024, the outside links included in the document are valid. As much as possible, I tried to use *URL* that should have a certain degree of permanence, but I cannot guarantee that all the links will stay accessible in the distant future.

[1] FAQs on Style

CHAPTER

ONE

PREAMBLE

À une vache près, hein, c'est pas une science exacte..

Karadoc (Jean-Christophe Hembert), *Kaamelott*, Livre II, épisode "Sept cent quarante-quatre", écrit par Alexandre Astier.

1.1 Thanks and acknowledgments

There are many people to thank for their help, support and contribution, (in diverse ways) to the work presented here.

A big thank you to Maëlle Kerveno and Philippe Dessagne who welcomed me in the *DNR* (at the time *Grace*) team in 2013. Since then, they have always been pushing me forward, encouraging me to aim for higher quality results and never settle for the simple "it works". They were very helpful in the preparation of this manuscript.

At their side, I can thank the postdocs and students that spent some time in the team and brought new ideas and challenges. In particular Antoine Bacquias, Eliot Party, François Claeys.

Thank you to all the members of the formal and informal collaborations we are part of: Marc Dupuis, Stephane Hilaire, Pascal Romain, Cyrille de Saint Jean, David Bernard, Gilles Noguere, Catalin Borcea, Andu Negret, Adina Olacel, Marian Boromiza, Arjan Plompen, Carlos Paradela Dobarro, Markus Nyman, Roberto Capote, Toshihiko Kawano.

I owe a big thank you to the laboratory directors (C. Roy, R. Barillon and S. Courtin) and associate directors for their support to the *DNR* group and my many activities in the lab. Special thanks to Sandrine Courtin and Gilbert Duchene, who, at an Agata@*Ganil* workshop in February 2013, informed me of a possible position opening in *IPHC*, and by that led me to where I am today. My most sincere gratitude to the administrative and technical staff at IPHC, they are our partners in research, and we could not do much without you.

Thank you to Arnd Junghans, Emmeric Dupont, Nicolas Arbor for agreeing to be reviewer of my manuscript, and to Anne Ruimy and Sandrine Courtin for also being part of the jury.

Thank you to Noemie Cobolet, Aurelie Hourlier-Fargette, Stephanie Cheviron, and all the other members of the University Library, IT and Data services, who helped me and became partners in my evolution from an *Open Science* curious observer into a dedicated actor. Without them, the *Open Science* part of this manuscript would not be more than an anecdotal paragraph in the appendix[1]. Thank you to Noémie and Stéphanie Cheviron for their help in writing the *OS* chapter.

A subtle thanks to JP, Marie, Nicolas, Momo, … companions of many coffee breaks and unofficial support group for Notilus/Goelett users.

Thanks to all the people who helped me write and correct this manuscript, give them credit for everything that is right. For the stuff that's wrong, I take the blame.

Last but certainly not least, thank you to Corinne and Valentin for their support, accepting my absences during my many work trips over the years, as well as the necessary work sessions at home, after dinner and during the weekends.

[1] Still, don't blame them.

1.2 Foreword

by the author.

It took me some time to start writing this HDR manuscript, for the simple reasons that I was unsure of what to put in it. The physics done in the *DNR* group is already well explained in past articles and thesis manuscripts, and even more in Maëlle Kerveno's own HDR[2], it would have been pointless to repeat it.

After some *soul-searching*, I finally realized that I could write about ~~my obsessions~~ what I focus on in my work: producing *uncertainties*, using the *Monte-Carlo method*, and writing reliable analysis code. Additionally, I will take this manuscript as an opportunity to write about methods, *work organization*, and *Open Science*. These are topics of great interest for me and *work* well together: publishing in *Open Access* and *Open Source* requires specific *methods* and structure in the work, ...

This manuscript will therefore be somewhat unusual (compared to most of the HDRs), in tone and in content. If a thesis manuscript is a scientific document with cold hard description of facts and methods, I intend to make this paper one with personality. There are not a lot of opportunities in a scientist career to put "*on the records*" some methods and reflection on what one thinks is a good way to do *things*. Still, it won't be an opinion piece. The methods and works presented here will not necessarily be truths universally acknowledged, but you'll find in these pages statements that hold true in the particular cases discussed. However, they may not be the **only** valid ones.

The physics (producing inelastic neutron scattering on ^{183}W) is a support for the description of the *Monte-Carlo method* as well as code writing strategy. Indeed, one important point to take away is that the code cannot be considered anymore as a *black box* that magically turn data recorded with a well characterized setup into detailed experimental result. It is as much part of the process as the detectors and should be described and tested with the same degree of care.

Moreover, it is much more than a simple *PDF* file[3] with text and figure. True to my focus on *Open Science* and *Open Source*, the whole document, as well as any associated scripts, and codes used or referred to in the text, will be made available.

The main part of the document (from "*Context and motivation*" to "*Results and discussions*") is sequential, but the *Open Science* or *Management* chapters, as well as the sections in the *appendix*, can be read independently. The text is rich in hyperlinks connecting sections together. Don't hesitate to click and Rabbit-hole your way into the topics.

[2] Maëlle Kerveno. "NACRE: Le Noyau Au Coeur du RéactEur". Université de Strasbourg, 2018. ⟨tel-03013645⟩ <https://hal.science/tel-03013645>

[3] In fact, the preferred way to read the document is not the *PDF* file, but *HTML* format pages.

CHAPTER

TWO

CONTEXT AND MOTIVATION

This section presents the general context and motivation of my work.

2.1 Nuclear data for application

Today, nuclear production for electricity uses mainly the neutron induced fission of the 235 isotope of uranium (Z=92). This method of production has disadvantages: it requires isotopic enrichment in ^{235}U and generates radioactive waste (fission fragments and actinides) for which no definitive solution yet exists. In order to imagine a sustainable and clean electronuclear cycle, new generation reactors are being studied[1]. These reactors will have to be adapted to meet economic constraints (reactor construction costs, reprocessing of used fuel), safety (accident prevention, proliferation) and sustainability (resources, waste storage). These new reactor concepts and new fuel cycles will produce fewer actinides, use other fissile nuclei (such as the isotopes ^{233}U or ^{239}Pu), or allow transmutation of waste.

The design of these systems requires precise numerical simulations in order to integrate all the mechanical, thermal and nuclear physics constraints and to ensure the safety conditions in their operation. These simulations use evaluated databases[2,3,4] which contain the physical quantities describing the various processes considered. They gather the cross sections, the angular distributions, the spectra of emitted particles… for the nuclear reactions induced by neutron or charged particles on the elements of the fuel or of the structure. These evaluations are the result of both theoretical and experimental work, and represent the best estimate of the value of these physical quantities. Sensitivity studies[5] show that the current uncertainties in the evaluated databases prevent reaching the expected accuracy objectives on the simulation of core parameters.

These uncertainties are explained by a lack of experimental data and by a need for theoretical developments for a better description of the reaction mechanisms. In order to consider industrial applications, the current uncertainties must be significantly reduced. This will necessarily require new measurement campaigns for the reactions of interest. The nuclear physics experiments will provide new constraints for the models and will allow improvements of the evaluations and the theoretical description of the processes of the nucleon-nucleus interaction.

[1] The Generation IV International Forum

[2] ENDF/B-VII.1 Evaluated Nuclear Data Library

[3] Joint Evaluated Fission and Fusion Nuclear Data Library (JEFF)

[4] The Japanese Evaluated Nuclear Data Library

[5] Uncertainty and target accuracy assessment for innovative systems using recent covariance data evaluations. Report from Working Party on International Nuclear Data Evaluation Co-operation sub group 26

2.1.1 Inelastic scattering

Among the reactions that take place in the reactor, the inelastic neutron scatterings (n, xn) are important. Indeed, they modify the energy distribution and the number of neutrons, and, for $x > 1$, create new isotopes in the medium. They therefore have a significant impact on the behavior of the reactor. However, the effective cross sections of reactions (n, xn) on uranium isotopes are only known with accuracies of 20 % in certain energy domains. This leads to significant uncertainties about the power, reactivity, or criticality of next-generation reactors. For example, sensitivity studies[Page 5, 5] have determined that the uncertainty in the neutron multiplication factor k_{eff} in new lead-cooled fast reactor designs is 1.4 % in total, of which 0.7% is due to the inelastic scattering cross section on the ^{238}U core (see reference[Page 5, 5] and Figure 1).

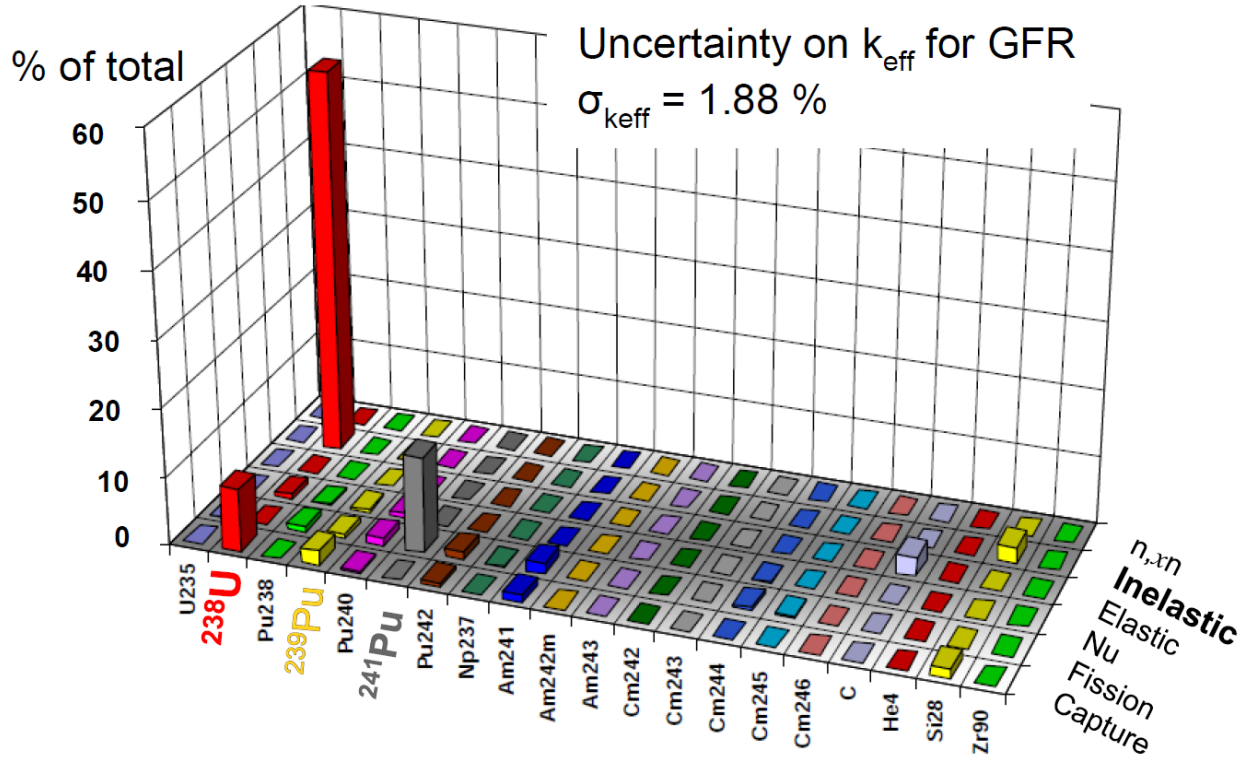


Figure 1: Break down of uncertainty contributions (in %) associated with the total combined uncertainty on the k_{eff} of a Gas Fast Reactor (GFR) by isotope and reaction channel. (From reference[Page 5, 5].)

These uncertainties come, for one part, from a lack of experimental data (see Figure 2) to constrain the evaluation, and, on the other hand, from deficiencies in models. To improve the reaction models and their ingredients (level density, transition strength functions, ...) theoretical work is needed, but this work requires new, more *microscopic* data to guide and/or test the new reaction description.

Following sensitivity analysis works[Page 5, 5], one can establish a list of required improvement, with target uncertainty in the final evaluation. The *NEA* **Nuclear Data High Priority Request List**[6] is such a list, compiling the nuclei and reactions of interest to be improved, including quantity (cross-section, yield, angular distribution...), energy range and motivation. For example, request number 18H lists the inelastic neutron scattering off ^{238}U cross-section as a quantity of interest.

[7] A.J. Koning, D. Rochman, J.-Ch. Sublet, N. Dzysiuk, M. Fleming, S. van der Marck. "TENDL: Complete Nuclear Data Library for Innovative Nuclear Science and Technology" Nuclear Data Sheets 155 (2019) 1-55 https://doi.org/10.1016/j.nds.2019.01.002.

[6] High Priority Request list (HPRL)

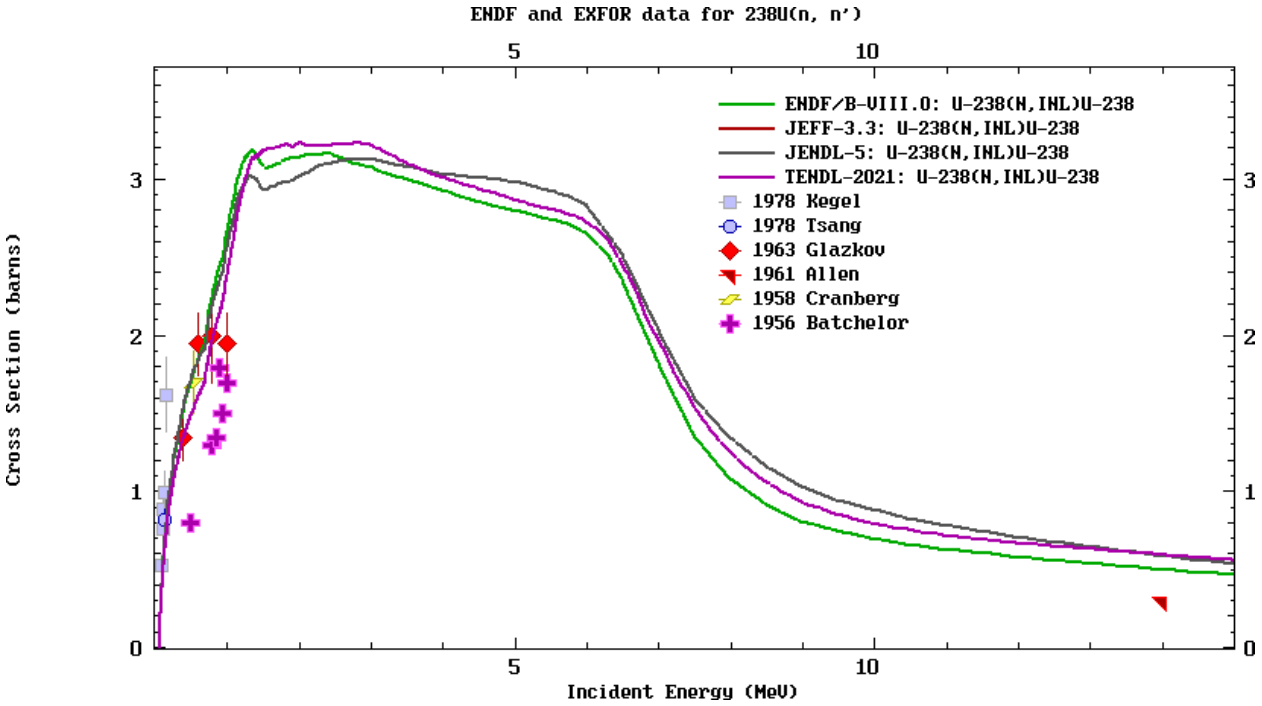


Figure 2: Cross section of inelastic neutron scattering off Uranium 238 according to four principal evaluations (JEFF-3.3[Page 5, 3] and TENDL-2021[Page 6, 7] are the same), compared with the experimental data (data collected from *Exfor* and *ENDF*).

Recently, the importance of quality nuclear structure information for reliable nuclear data as emerged. Indeed, in experiments, or in the evaluation process, the level scheme of the nucleus is often used as input to interpret or process the measured data. Unfortunately, just as the reaction databases may present defaults, nuclear structure databases[8,9] also present uncertainties and *holes* in the level schemes of nuclei of interest[10]. It's typical to observe transitions with large uncertainties on intensities, or just a limit on the intensity given, as well as noting a significantly low number of states listed in certain spin/parity/excitation energy regions[11]. This will have an impact on the accuracy of the derived nuclear data, even if the underlying experimental data has a very good precision[Page 7, 10].

2.1.2 Nuclear Data study program at IPHC

The Nuclear Data for Reactors group (*DNR*, formerly "Grace") at the *IPHC*[12] has started since 2000 a program of measurements and studies of $(\mathrm{n},\ x\mathrm{n}\ \gamma)$ reactions. The experiments are carried out at the *Gelina* neutron beam at the *JRC-Geel* (Belgium)[13,14,15] with the GeRmanium array for Actinides PrEcise MEasurements (*Grapheme*), seen in

[8] Reference Input Parameter Library (RIPL-3)

[9] Evaluated Nuclear Structure Data File (ENSDF)

[10] Greg Henning, Maëlle Kerveno, Philippe Dessagne, François Claeys, Nicolas Dari Bako, et al. On the need for precise nuclear structure data for high quality (n, n' γ) cross-section measurements. EPJ Web of Conferences, 2023, 284, pp.01022. ⟨https://dx.doi.org/10.1051/epjconf/202328401022⟩. ⟨https://hal.science/hal-04124945⟩

[11] Citation needed (by which I mean that it has been mentionned in papers, presentations, discussions, but I have not a particular reference article to cite as illustration of the point).

[12] Données Nucléaires pour les Réacteurs (DNR)

[13] GELINA The European Commission's Linear Electron Accelerator Facility

[14] "GELINA: A modern accelerator for high resolution neutron time of flight experiment". A. Bensussan, J.M. Salome, Nucl. Instrum. Methods 155, 11 (1978) https://doi.org/10.1016/0029-554X(78)90181-7

[15] "Global characterisation of the GELINA facility for high-resolution neutron time-of-flight measurements by Monte Carlo simulations". D. Ene, C. Borcea, S. Kopecky, W. Mondelaers, A. Negret, A.J.M. Plompen, Nucl. Instrum. Methods Phys. Res., Sect. A 618, 54 (2010) https://doi.org/10.1016/j.nima.2010.03.005

Figure 3 and, later, Figure 12. The latter consists (today) of 6 planar germanium detectors - one of which is segmented into 36 pixels, connected to digital electronics[16].

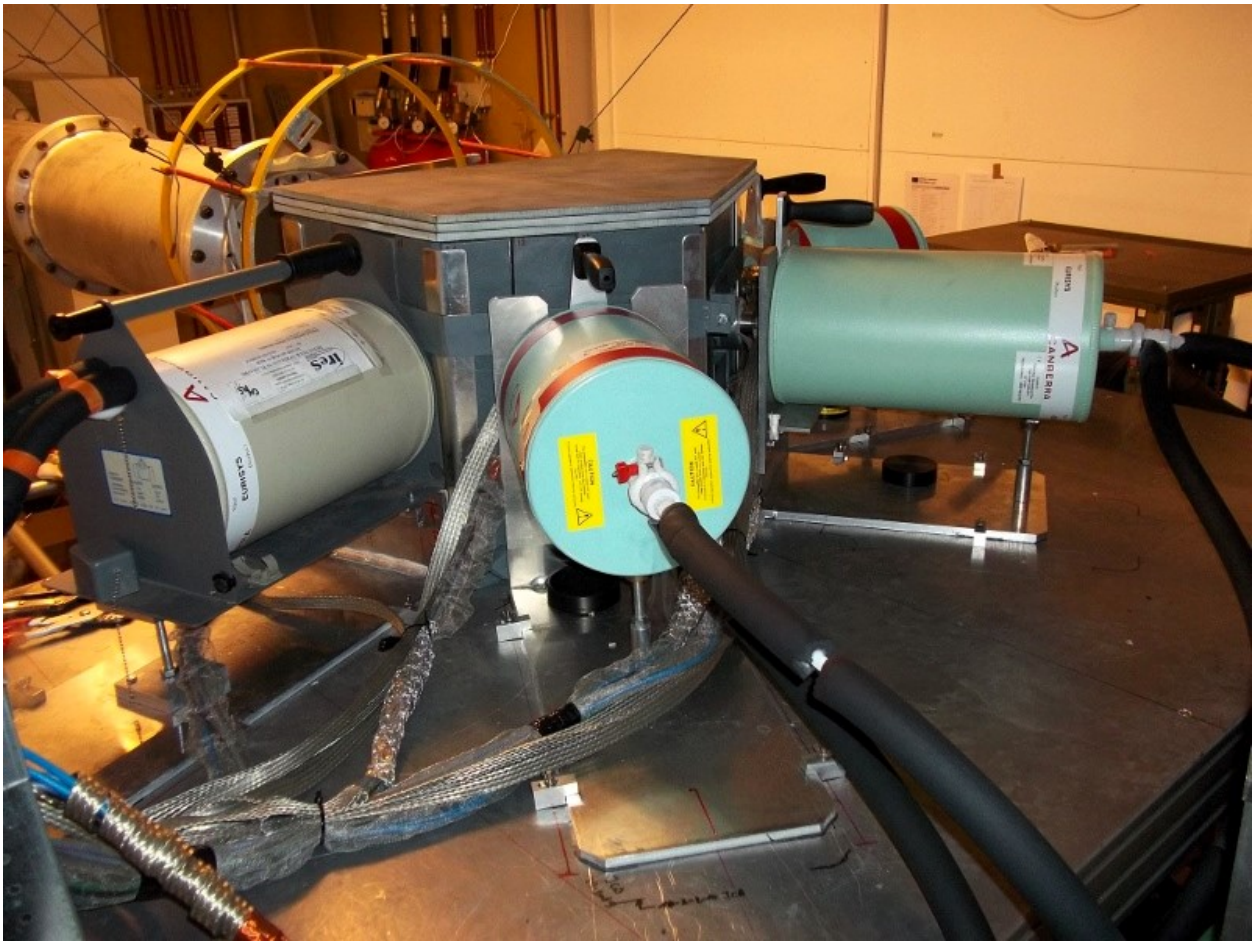


Figure 3: Photography of *Grapheme* in 2010, at the time when the *data analyzed later in this manuscript* was recorded. The liquid nitrogen dewars of the HPGe detectors can be seen sticking out of a lead and copper castle shield, with the active Ge crystal being inside, surrounding the target. The neutron beam comes, in the air, from the right and pass through a gap in the shielding (not distinguishable on the picture).

The experimental method combines prompt γ-ray spectroscopy with time-of-flight neutron energy measurement[17,18]. The accuracy on the measured $(\mathrm{n},\ \mathrm{n}'\gamma)$ cross sections is 3-15 $\%$ depending on the considered γ ray, the neutron energy , ...

[16] "GRAPhEME: A setup to measure (n, xn γ) reaction cross sections" G. Henning et al., 4th International Conference on Advancements in Nuclear Instrumentation Measurement Methods and their Applications (ANIMMA), 2015, pp. 1-9, doi: 10.1109/ANIMMA.2015.7465505. https://doi.org/10.1109/ANIMMA.2015.7465505

[17] "How to produce accurate inelastic cross sections from an indirect measurement method?" Maëlle Kerveno, Greg Henning, Catalin Borcea, Philippe Dessagne, Marc Dupuis, et al. EPJ N - Nuclear Sciences & Technologies, 2018, 4, pp.23. ⟨10.1051/epjn/2018020⟩. https://doi.org/10.1051/epjn/2018020

[18] "From γ emissions to $(\mathrm{n},\ x\mathrm{n})$ cross sections of interest: The role of GAINS and GRAPhEME in nuclear reaction modeling." M. Kerveno, A. Bacquias, C. Borcea, Ph. Dessagne, G. Henning, et al. European Physical Journal A, 2015, 51 (12), pp.167. https://doi.org/10.1140/epja/i2015-15167-y

The $(\mathrm{n},\ \mathrm{n}')$ and $(\mathrm{n},\ 2\mathrm{n})$ reactions were studied on the isotopes of $^{\mathrm{nat}}$Zr, $^{\mathrm{nat,182,183,184,186}}$W[19], ^{233,235,238}U[20,21], and ^{232}Th[22].

The experimental $(\mathrm{n},\ x\mathrm{n}\,\gamma)$ cross sections are compared to model predictions and used to improve reaction codes such as *Talys*[23,24], *Empire*[25], or *CoH3*[26]. The interpretation of differences between measures and calculations is a way to reveal the points needing refinement in the codes. In particular, we showed, first in ^{238}U[Page 9, 20], then in even-even tungsten isotopes[27], the importance of the spin distribution in the pre-equilibrium steps of the reaction modeling[28].

Important: One of the key issues in producing experimental values is to publish them with their uncertainties (and covariances) so that they can be fully exploited during evaluation. And beyond just giving the uncertainties and covariances, the method for obtaining them needs to be clearly explained, so that evaluators can take them properly into account in their work. (See later, *The important thing about uncertainties*)

The work presented in this manuscript offers *one way* to produce such experimental results, along with *methods* and *tools* to make the process accessible to be studied.

Perspective for the DNR group

In the near future, the *DNR* team will follow several research paths.

First, at *Gelina*, data as been recorded for ^{233}U and results will be published soon[29]. Data taking for the experimental study of $(\mathrm{n},\ x\mathrm{n}\,\gamma)$ reactions on a ^{239}Pu target is in progress (since mid-2023, including lengthy technical beam shutdown periods).

Additionally, to study the highly converted transitions in the actinides, not available experimentally in $(\mathrm{n},\ x\mathrm{n}\,\gamma)$ study, we are developing, in collaboration with *Ifin-HH* and *JRC-Geel*, the *Delco* setup (**Detecteur d'ELectrons de COnversion**)[30].

Finally, the new neutron beam facility *NFS* at *Ganil* recently started operations. The neutron beam there has an average

[19] "MEASUREMENT OF ^{182,184,186} W $(\mathrm{n},\ \mathrm{n}'\,\gamma)$ CROSS SECTIONS AND WHAT WE CAN LEARN FROM IT." G. Henning, Antoine Bacquias, Catalin Borcea, Mariam Boromiza, Roberto Capote, et al.. EPJ Web of Conferences, 2021, 247, pp.09003. https://doi.org/10.1051/epjconf/202124709003

[20] "Measurement of ^{238}U $(\mathrm{n},\ \mathrm{n}'\,\gamma)$ cross section data and their impact on reaction models." M. Kerveno, M. Dupuis, A. Bacquias, F. Belloni, D. Bernard, et al.. Physical Review C, 2021, 104 (4), pp.044605. ⟨10.1103/PhysRevC.104.044605⟩. https://doi.org/10.1103/PhysRevC.104.044605

[21] "Measurement of ^{235}U $(\mathrm{n},\ \mathrm{n}'\,\gamma)$ and ^{235}U $(\mathrm{n},\ 2\mathrm{n}\,\gamma)$ reaction cross sections." M. Kerveno, J. Thiry, A. Bacquias, C. Borcea, P. Dessagne, et al.. Physical Review C, 2013, 87 (2), ⟨10.1103/PhysRevC.87.024609⟩. https://doi.org/10.1103/PhysRevC.87.024609

[22] "Neutron inelastic scattering of 232 Th: measurements and beyond." Eliot Party, Catalin. Borcea, Philippe Dessagne, Xavier Doligez, Grégoire. Henning, et al.. 5th International Workshop On Nuclear Data Evaluation for Reactor applications (WONDER-2018), Oct 2018, Aix en Provence, France. pp.03005. https://doi.org/10.1051/epjconf/201921103005

[23] "TALYS-1.0" A. Koening, S. Hilaire, M. Duijvestijn, in International Conference on Nuclear Data for Science and Technology (2007), Vol. EDP Sciences, 2008, pp. 211–214. https://doi.org/10.1051/ndata:07767

[24] "TALYS: modeling of nuclear reactions" Arjan Koning, Stephane Hilaire, Stephane Goriely. Eur. Phys. J. A 59 (6) 131 (2023) https://doi.org/10.1140/epja/s10050-023-01034-3

[25] "EMPIRE: Nuclear Reaction Model Code System for Data Evaluation" M. Herman, R. Capote, B. Carlson, P. Obložinský, M. Sin, A. Trkov, H. Wienke, V. Zerkin, Nucl. Data Sheets 108, 2655 (2007) https://doi.org/10.1016/j.nds.2007.11.003

[26] "Statistical Hauser-Feshbach theory with width-fluctuation correction including direct reaction channels for neutron-induced reactions at low energies" T. Kawano, R. Capote, S. Hilaire, P. Chau Huu-Tai, Phys. Rev. C 94, 014612 (2016) https://doi.org/10.1103/PhysRevC.94.014612

[27] "Improving the accuracy of 182,184,186W(n, n'γ) cross sections calculations" G. Henning, M. Dupuis, M. Kerveno, R. Capote, Ph. Dessagne, S. Hilaire, T. Kawano, P. Romain, C. de Saint Jean, P. Scholtes, P. Tamagno. In preparation for Physical Review C. (2024)

[28] "Microscopic modeling of direct pre-equilibrium emission from neutron induced reactions on even and odd actinides". M Dupuis, S Hilaire, S Péru, E Bauge, M Kerveno, P Dessagne, G Henning. EPJ Web of Conferences 146, 12002 (2017) https://doi.org/10.1051/epjconf/201714612002

[29] Measurement and evaluation of partial and total (n,n' γ) reaction cross-sections on highly radioactive nuclei of interest for energy production application. François Claeys, Philippe Dessagne, Maëlle Kerveno, Cyrille De Saint Jean, Catalin Borcea, Marian Boromiza, Roberto Capote, Nicolas Dari Bako, Marc Dupuis, Greg Henning, Stéphane Hilaire, Alexandru Negret, Gilles Noguère, Markus Nyman, Adina Olacel and Arjan Plompen. EPJ Web of Conf., 284 (2023) 01014. https://doi.org/10.1051/epjconf/202328401014

[30] Markus Nyman, Thomas Adam, Catalin Borcea, Marian Boromiza, Philippe Dessagne, *et al.* "New equipment for neutron scattering cross-section measurements at GELINA." ND 2019: International Conference on Nuclear Data for Science and Technology, 2019, Pékin, China. pp.17003, ⟨10.1051/epjconf/202023917003⟩.

energy higher than the one at *Gelina* and this will allow us to study $(\mathrm{n},\ 2\mathrm{n})$ and $(\mathrm{n},\ 3\mathrm{n})$ reactions[31,32].

Important: For these future works, the availability of the *full Monte Carlo analysis code* described in *this manuscript* will be an asset. First, because it is designed to be *flexible* and accommodate more detector inputs, or in different file format. Then, because it will be completely *open*, with a *detailed documentation*, it will meet the *criteria for accurate and meaningful evaluation of experimental values*.

2.2 ^{183}W

Tungsten is not an active element in nuclear reactors, but, because of its chemical and mechanical properties[1], it is used in many alloys. The interaction of neutrons with tungsten is therefore of importance for reactor physics, in particular for fusion reactors[2], in which tungsten is one of the most exposed materials to high energy neutrons. From a theoretical point of view, a better description of $(\mathrm{n},\ x\mathrm{n})$ reactions on tungsten nuclei allows an improvement of models for other key nuclei in reactor fuel. Indeed, tungsten isotopes are deformed like actinides[3], but also easier to describe as they do not present a neutron-induced fission channel (theoretical fission barrier for Tungsten isotopes is around 21 MeV[4]). Still, there are very few measurements available today to test evaluations. Our experimental data[5] will provide an extensive and constraining test to the predictability of models. Figure 4 shows a simplified level scheme for the isotope.

Data for the naturally occurring even-even isotopes (182, 184 and 186) have already been published[6,Page 10, 5, and7]. The *same setup* has been used to record data with a ^{183}W target. The experimental cross-sections for the odd-N isotope will bridge the gap between ^{182} and ^{184}W by connecting the $(\mathrm{n},\ \mathrm{n}')$ and $(\mathrm{n},\ 2\mathrm{n})$ channel of the three isotopes together. (Note: the $^{184}\mathrm{W}(\mathrm{n},\ 2\mathrm{n})$ cross section has not yet been analyzed.) Furthermore, because of the unpaired nucleon in ^{183}W, this isotope offers unique insights into nuclear structure, reactions, and forces, and the experimental $(\mathrm{n},\ x\mathrm{n}\ \gamma)$ cross section will be very valuable to improve our understanding of nuclear reactions and structure.

2.2.1 Current knowledge

At the present time (spring 2024), there is only a limited amount of data available on *pure* neutron induced reactions on ^{183}W (that excludes ratios, ...).

In particular, **no** data is reported in the *Exfor* database for inelastic scattering $(\mathrm{n},\ \mathrm{n}')$ and just a couple of data set exists for partial cross sections in the inelastic channel.

[31] GRAPhEME: performances, achievements (@EC-JRC/GELINA facility) and future (at GANIL/SPIRAL2/NFS facility). Maëlle Kerveno, Catalin Borcea, Marian Boromiza, Roberto Capote, François Claeys, Nicolas Dari Bako, Cyrille De Saint Jean, Philippe Dessagne, Jean Claude Drohé, Marc Dupuis, Greg Henning, Stéphane Hilaire, Toshihiko Kawano, Alexandru Negret, Markus Nyman, Adina Olacel, Carlos Paradela, Arjan Plompen and Ruud Wynants. EPJ Web of Conf., 284 (2023) 01005. https://doi.org/10.1051/epjconf/202328401005

[32] "Studies of (n,2n), (n,3n) reactions at the new *GANIL*/SIPRAL2/*NFS* facility, using prompt gamma spectroscopy." PhD Thesis subject.

[1] Haynes, W.M. (Ed.). (2016). CRC Handbook of Chemistry and Physics (97th ed.). CRC Press. https://doi.org/10.1201/9781315380476

[2] M.R.Gilbert et al "An integrated model for materials in a fusion power plant: transmutation, gas production, and helium embrittlement under neutron irradiation," Nuclear Fusion, vol. 52, no. 8, p. 083019, 2012. https://dx.doi.org/10.1088/0029-5515/52/8/083019

[3] P.Moller, J.R. Nix, W.D. Myers, W.J. Swiatecki, Nuclear Ground-State Masses and Deformations, Atomic Data and Nuclear Data Tables, Volume 59, Issue 2, 1995, Pages 185-381, https://doi.org/10.1006/adnd.1995.1002 .

[4] G.Henning. A python reimplementation of A. Sierk's BARFIT. 2021. ⟨hal-03132426v2⟩

[5] "Improving the accuracy of 182,184,186W(n, n'γ) cross sections calculations" G. Henning, M. Dupuis, M. Kerveno, R. Capote, Ph. Dessagne, S. Hilaire, T. Kawano, P. Romain, C. de Saint Jean, P. Scholtes, P. Tamagno. In preparation for Physical Review C. (2024)

[6] "MEASUREMENT OF ^{182,184,186} W $(\mathrm{n},\ x\mathrm{n}\ \gamma)$ CROSS SECTIONS AND WHAT WE CAN LEARN FROM IT." G. Henning, Antoine Bacquias, Catalin Borcea, Mariam Boromiza, Roberto Capote, et al.. EPJ Web of Conferences, 2021, 247, pp.09003. ⟨10.1051/epjconf/202124709003⟩.

[7] HENNING, Greg, 2024, "Experimental (n, n' γ) cross sections for isotopes 182,184 and 186W", https://doi.org/10.57745/JRCNEJ, Recherche Data Gouv, V1

[8] Level scheme designer in python

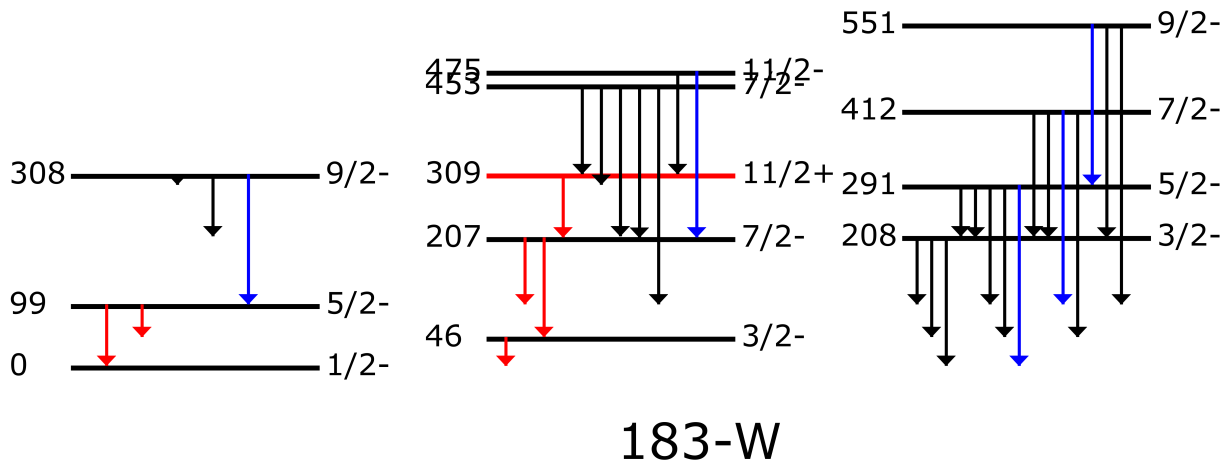


Figure 4: Simplified level scheme for ^{183}W. The 5.30 seconds, $\frac{11}{2}^+$ isomeric level at 309.5 keV is represented in red, with all transitions below in red too. The transitions studied in this work are in blue. For readability, some transitions of low intensity have been ignored. Level scheme drawn with `pylevelscheme`[8].

(n, 2n) cross section data

J.Frehaut, *et al.*[9] measured the (n, 2n) cross section off ^{183}W (along with many other isotopes) in 1974 with neutrons in the range 7 to 15 MeV (Figure 5).

The evaluations are overall compatible with experimental data, although there is a spread in model values in the 6 to 12 MeV neutron energy range.

(n, n′ γ) data

Exfor lists one data set related to (n, n′ γ) cross section. The 1996 data[10] is presented as the cross section of 210±30 keV γ rays (detected with NaI scintillator) at $E_n = 3$ MeV. The 210 ± 30 keV energy can be matched to 8 γ rays in the output of *Talys* calculations[11] for (n, n′) reaction on ^{183}W (a few more transitions with energy within the 30 keV range around 210 keV can be found, from high excitation energy states or *stopped* by long live isomers):

- decay from the 208.8 keV $\frac{3}{2}^-$ state to the ground state ($E_\gamma = 208.8$ keV),
- from the 291.7 keV $\frac{5}{2}^-$ state to the 99 keV $\frac{5}{2}^-$ one ($E_\gamma = 192.6$ keV),
- from the 308.9 keV $\frac{9}{2}^-$ state to the 291.7 keV $\frac{5}{2}^-$ one ($E_\gamma = 209.9$ keV),
- from the 412.1 keV $\frac{7}{2}^-$ state to the 208.8 keV $\frac{3}{2}^-$ one ($E_\gamma = 203.3$ keV),
- from the same $\frac{7}{2}^-$ state to the 207.0 keV $\frac{7}{2}^-$ one ($E_\gamma = 205.1$ keV),
- from the 739.95 keV $\frac{11}{2}^-$ state to the 551.2 keV $\frac{9}{2}^-$ one ($E_\gamma = 188.8$ keV),
- from the 849.9 keV $\frac{15}{2}^-$ state to the 631.1 keV $\frac{13}{2}^-$ one ($E_\gamma = 219.4$ keV),

[9] http://www-nds.iaea.org/EXFOR/20416.108
[10] http://www-nds.iaea.org/EXFOR/41245.022
[11] Done by Pascal Romain (CEA/DAM/DIF) in 2016

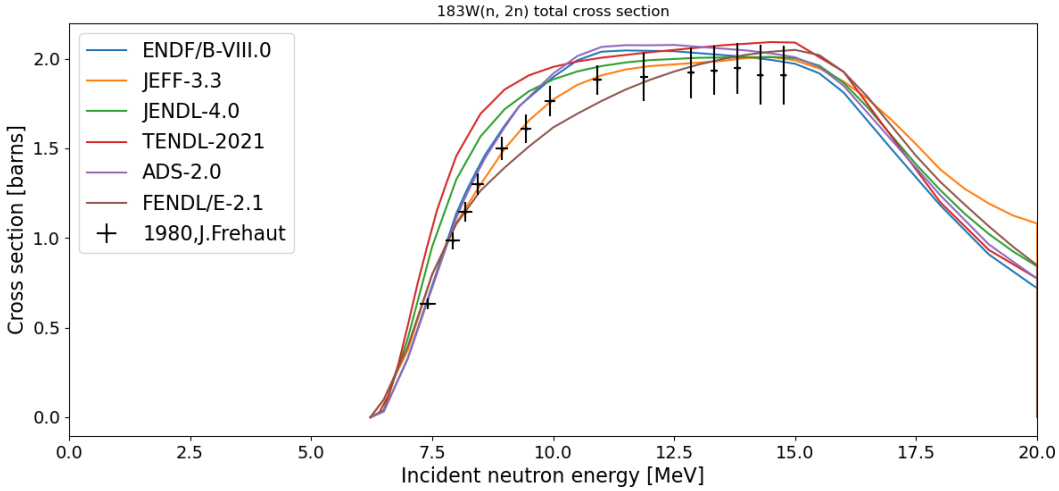


Figure 5: Current experimental data points for $(\mathrm{n},\ 2\mathrm{n})$ data[9], compared with several evaluations. (Data obtained from *Exfor* and *ENDF*.)

- and from the 965.13 keV $\frac{15}{2}^-$ state (at 973.9 keV according to the *Talys* structure file) to the 766.3 keV $\frac{11}{2}^-$ one (772 keV in Talys) ($E_\gamma = 198.6$ keV (201.9 keV in *Talys*)).

The quoted cross section in the *Exfor* entry is 2523 ± 730 mbarns[Page 11, 10]. This value is of the order of the total $(\mathrm{n},\ \mathrm{n}')$ cross section (as shown in Figure 6), but is much lower than the sum of all possible 210 ± 30 keV γ contributions. Different *Talys* calculations lead to the same results, so the discrepancy does not come from specific parameters in the computation.

Unfortunately, when compared to γ ray cross section (predicted by *Talys*), there's no obvious match (Figure 6). As the original paper cited as source in *Exfor* is not available[12], it's difficult to conclude about this, but we can guess there's an error somewhere, whether in the numerical value, in the actual quantity reported, ...

Note: This case is actually an example why the integration of data in the Exfor database is not enough to ensure that the data can be understood later. The *Exfor format* may not contain all the information needed to interpret the value. Or the referenced articles might not be available anymore (or be very hard to find).

Update

See in *Appendix* for updated information on this *Exfor* entry.

---

[12] I tried very hard, through different channels: University of Strasbourg library services, New York State Library's Ask a Librarian, Shadow library websites, I even asked on 𝕏-*formerly-known-as-twitter*, ... to no avail. See and *update on that topic int he appendix*.

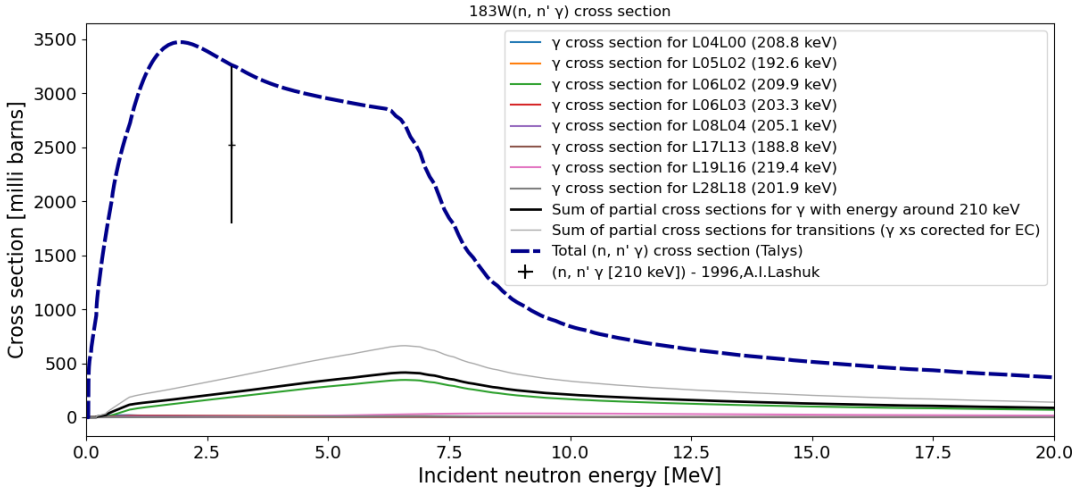


Figure 6: Experimental data from the I. Lashuk, 1996's data[Page 11, 10], as seen in *Exfor*. The data point is compared to *Talys* calculations by P. Romain[Page 11, 11]. The transitions matching the 210 ± 30 keV criteria are plotted in full color line. The sum of these $(\mathrm{n},\ \mathrm{n}'\ \gamma)$ cross sections is in full black line. The sum of transitions cross sections (corrected for electronic conversion) is in full gray line. The total $(\mathrm{n},\ \mathrm{n}')$ cross section is in dashed blue line.

Population of an isomer in $(\mathrm{n},\ \mathrm{n}')$ reactions

The final data set of interest available in *Exfor* is an isomer decay cross-section at 14.6 MeV. The $\frac{11}{2}^+$ state at 309.5 keV has a 5.30 seconds lifetime and the production has been measured by B. Anders, *et al.*[13,14]. The reference states a 127 ± 14 mbarns for the 108 keV γ ray (which decays from the $\frac{7}{2}^-$ state at 207 keV, which is fed by the only transition decaying from the $\frac{11}{2}^+$ isomer state). In Figure 7, we compare the data point to *Talys* predictions for the direct excitation of the $\frac{11}{2}^+$ state, the total cross section of populating the $\frac{11}{2}^+$ (either directly or from feeding by states above), and the 108 keV γ ray, with a very different value predicted by the code. However, the fact that this is an isomer may confuse the code and the outputted cross section might not be the same quantity as has been measured. We note that the value by[14] is on the same order of magnitude than the *plateau* cross section at lower incident neutron energy, so, some interpretation of the effective energy of the interaction might be playing an effect here.

2.2.2 $(\mathrm{n},\ \mathrm{n}'\ \gamma)$ cross sections measurements for ^{183}W

Using *Grapheme*, the neutron inelastic scattering off ^{183}W was *investigated* by measuring $(\mathrm{n},\ \mathrm{n}'\ \gamma)$ *cross sections*.

The data presented *later*, will fill a gap in the (as we saw) sparse data on $(\mathrm{n},\ \mathrm{n}')$ reaction. Furthermore, in conjunction with $(\mathrm{n},\ \mathrm{n}'\ \gamma)$ and $(\mathrm{n},\ 2\mathrm{n}\ \gamma)$ on *even-even isotopes*[Page 10, 5] (in particular ^{182}W and ^{184}W) will allow a *chaining* of experimental results along the isotopic chain, thanks to overlapping $(\mathrm{n},\ \mathrm{n}'\ \gamma)$ and $(\mathrm{n},\ 2\mathrm{n}\ \gamma)$ measurements.

This will provide a valuable data set to constrain the models and parameters for the reactions code.

[13] http://www-nds.iaea.org/EXFOR/21818.007

[14] Anders, B., Pepelnik, R., Fanger, HU. (1983). Application of a Novel 14-Mev Neutron Activation Analysis System for Cross-Section Measurements with Short-Lived Nuclides. In: Böckhoff, K.H. (eds) Nuclear Data for Science and Technology. Springer, Dordrecht. https://doi.org/10.1007/978-94-009-7099-1_193

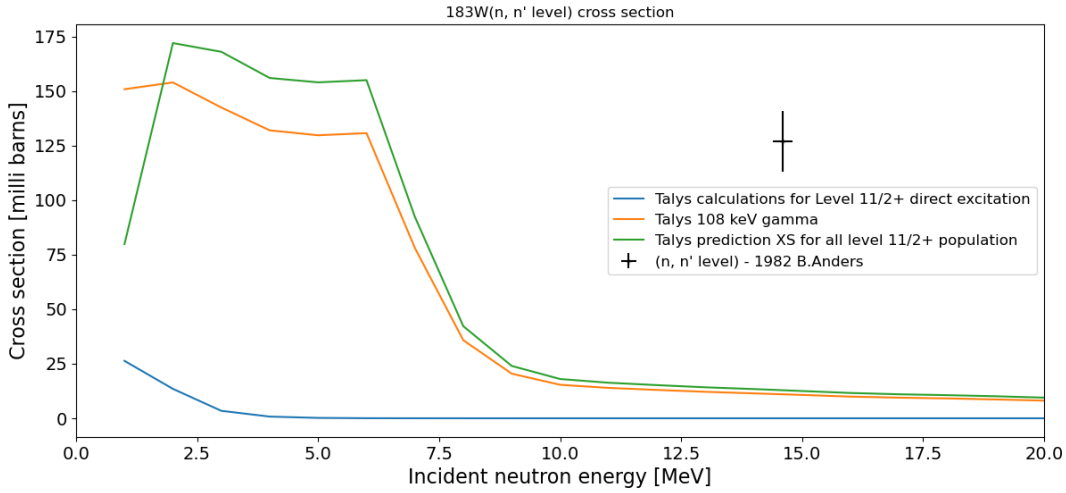


Figure 7: Current experimental data point[13 andPage 13, 14] for $(\mathrm{n},\ \mathrm{n}')$ reaction on the $\frac{11}{2}^+$ isomeric level, compared with *Talys* calculations[Page 11, 11].

CHAPTER

THREE

USING RANDOM SAMPLING TO PRODUCE UNCERTAINTIES AND CORRELATION MATRICES

3.1 Uncertainty and covariance ?

As in many other scientific fields, nuclear physics measurements come with a degree of *uncertainty*. The concept of uncertainty is used to express the level of confidence in the measurement results. It can be characterized by a range of values, a probability distribution, or a standard deviation. The sources of uncertainty can be numerous and may include the precision of the instrument, the limited statistics, or the unknowns on some parameters. Whatever the source, the proper characterization of uncertainty is essential for the correct interpretation of experimental results and for the comparison of these results with other measurements and/or theoretical predictions.

The discussion in this section focuses on the *practical* point of view, and the quoted formulas might differ from the *pure* mathematic ones found in statistics textbooks.

Writing convention

In the following, the uncertainty on a quantity x will be noted u_x in the case of a Gaussian probability density function and δ_x for a uniform one[1]. The central value of x is noted $\bar{x}$.

3.1.1 Uncertainties : definitions, usage, methods

The intrinsic uncertainties in nuclear physics experiment (and in particular in the case of *inelastic neutron scattering cross-section determination*) come from many different sources: time interval between clock-ticks, limited precision on distances, unknown contaminants in target materials, trigger rate limitation and signal processing in the acquisition (a.k.a. *pile-up* and *deadtime*), ...

All the *original* uncertainties in a physics analysis have to be evaluated (it is the nature of uncertainty that it cannot be measured[2]) to correctly take them into account. There are basically two ways to perform the uncertainty estimation, associated with uncertainty *Type*[3].

Type A
: The first type of uncertainty is determined by statistical methods. Mostly, via a repeated set of measurements. The standard deviation of the obtained distribution of results will be kept as the uncertainty.

Type B
: The other type of uncertainty is ... *any other methods*. For type B uncertainty, the estimated variance (i.e. ${u_x}^2$)

[1] Greg Henning. Uncertainty notation guidelines for scientific publications. 2019. ⟨hal-02115623v2⟩

[2] But *bias* (i.e. a systematic and deterministic effect) can be. (In which case, the bias correction method will most probably have its own uncertainty attached to it.)

[3] Guide to the expression of uncertainty in measurement (ISO/IEC Guides 98-3)

is evaluated using the knowledge about the setup used for the determination of the value, the use of a model or prescription, ...

Important: The fact that one uncertainty is evaluated via type **A** or **B** method does not imply that the source of the uncertainty is *systematic* or *statistic*. Indeed, this categorization is often misleading, as the *statistical* uncertainty in one case might be *systematic* in another (see *later*).

In addition to a central value and its associated standard deviation, the uncertainty is more generally defined by a probability density function (pdf, not *PDF*) that will generally follow a generic mathematical shape (Gaussian, uniform over an interval, Triangular, log-normal, Poisson, Maxwell-Boltzmann, ... See examples in Figure 8). The standard deviation is usually defined for these shapes, but does not fully characterize them.

Therefore, one should consider all values for which an uncertainty is associated (whether it is the result of a measurement, or a parameter used in the analysis) as a *random variable*. That means that a measured value of x is one *realization* of the random variable X, defined by a probability density function for which a mean value $\bar{x}$ and a standard deviation u_x can be computed[4].

Combination of uncertainties

The combination of uncertainties is usually done with the *square summation* formula. Considering the quantity a, obtain from two independent variables x and y with the function $f(x,\ y)$, the variance of a is given by :

$$ {u_a}^2 = \left(\frac{\partial f}{\partial x}(\bar{x},\ \bar{y})\right)^2 \times {u_x}^2 + \left(\frac{\partial f}{\partial y}(\bar{x},\ \bar{y})\right)^2 \times {u_y}^2 $$

The formula can be generalized to the case of many variables. One can easily obtain it by first order series expansion on the function f around $(\bar{x},\ \bar{y})$, using the definition that $\left\langle (\bar{x} - x)^2 \right\rangle = {u_x}^2$ (i.e. the variance of x is the average of the squares of the spread of the realizations of x around the random variable mean value)[5].

In cases where the variables are not independent, an extra term appears: $+2 \times \frac{\partial f}{\partial x}(\bar{x},\ \bar{y}) \times \frac{\partial f}{\partial y}(\bar{x},\ \bar{y}) \times \langle (x - \bar{x})(y - \bar{y}) \rangle$ (note that this term may be negative). Where we define $cov(x, y) = \langle (x - \bar{x})(y - \bar{y}) \rangle$ the covariance between the two random variables x and y (the variance ${u_x}^2$ is by definition equal to $cov(x, x)$). Additionally, the correlation between the two variables is $corr(x, y) = \frac{cov(x,y)}{u_x u_y}$.

Note: Two variables can be strongly correlated and still have a covariance equal to 0.

For example, if x is randomly distributed along the real axis with a central value of 0, and $y = x^2$, then $cov(x, y) = 0$ despite y being clearly correlated to x.

But[6], the previous formulas for uncertainty combinations apply only in cases of symmetrical, small deviations around the mean values. In practice, it's not always the case. (Normalization for example may prevent symmetric variations around a central value.) The exact combination of uncertainties should be done by convolution of probability density functions (Using a *random sampling* approach can be an efficient way to perform the uncertainty combination). This makes sure that any shape and type of uncertainties are correctly combined. In general, if one considers two random variables x and y, with probability density functions P_x and P_y, the distribution corresponding to possible values of a, with $a = f(x, y)$, will be $P_a(a) = \int \delta(a - f(x, y)) \times P_x(t)dx \times P_y(y)dy$.

[4] "Analyse de données en sciences expérimentales", Benoit Clément, Dunod, 2012. ISBN:9782100575695

[5] It's true, make the calculations yourself, or at least ask the students under your supervision to, so they can learn where it comes from and stop using it blindly in cases where it does not apply.

[6] Of course, there is a *but*.

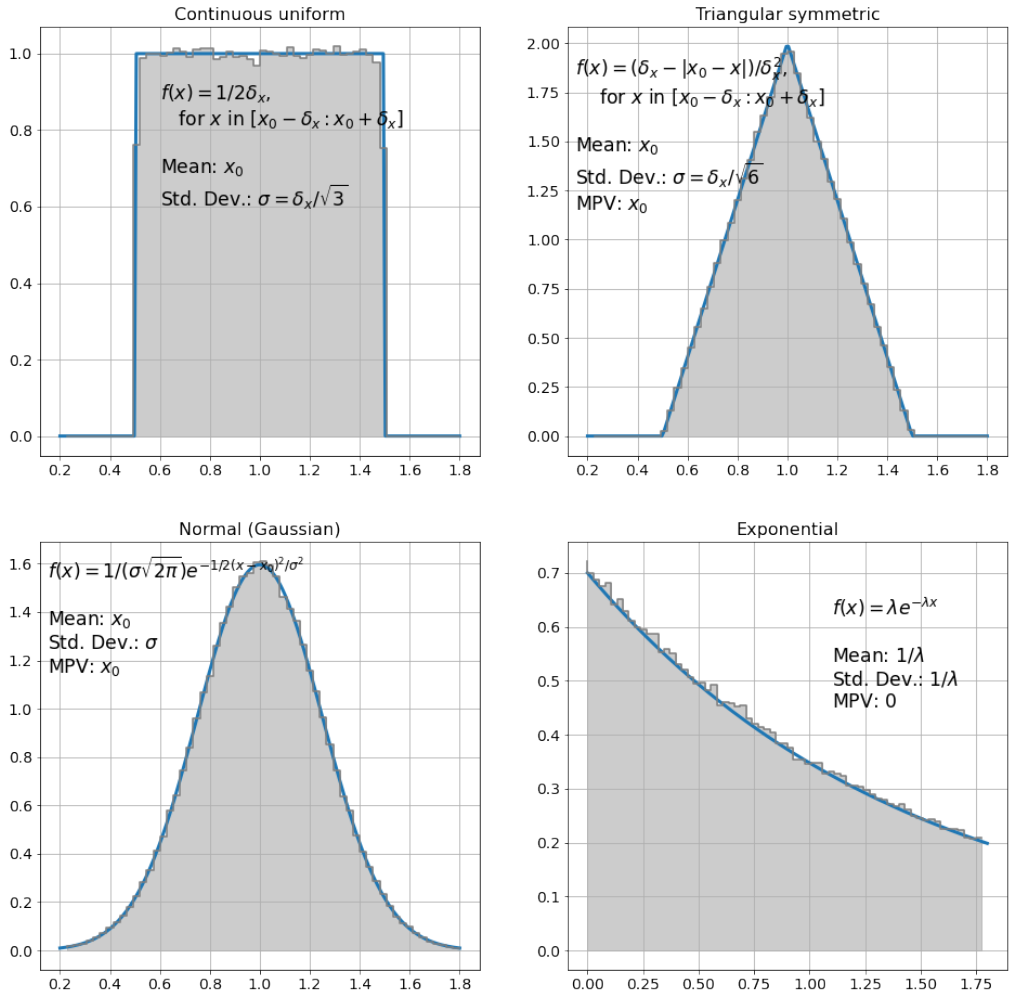


Figure 8: Example of four typical probability density distributions (uniform or *flat*, triangular, Gaussian or normal, and exponential), with their main characteristics: mean value, standard deviation and most probable value (MPV) when defined.

Statistical vs. Systematic uncertainties

It is very common in physics results to separate between the so-called *statistical* and *systematic* uncertainties. However, it is not always clear what is meant by these words. Let's clarify in the following paragraphs.

True statistical uncertainty

Literally, the *statistical uncertainty* could be understood as **uncertainty determined from the statistical distribution of results** (for example, read "Introduction to Statistical vs. Systematic Uncertainty", one of the top results in a Google search for **statistical uncertainty**). In other words, it is a *Type A uncertainty*.

This uncertainty will follow the *Statistics-101* formula $\bar{x} = \frac{1}{N}\sum_i x_i$, with the uncertainty u_x being equated to the standard deviation σ_x: ${\sigma_x}^2 = \frac{1}{N-1}\sum_i (x_i - \bar{x})^2$.

Uncertainty from limited Statistics

In (nuclear) physics, *statistical uncertainty* is more generally understood as **uncertainty arising from the limited statistics**, in other words "*we are not exactly sure of the value of the quantity of interest because we could not run the experiment forever*". This *statistical* uncertainty typically arises from counting something (events, atoms, ...) and will propagate through the usual *uncertainty combination* to the final value.

The default estimation for the statistical uncertainty for a quantity N is usually taken as $\sqrt{N}$. It can be (and usually is) true, but that's not a universal truth. The $\sqrt{N}$ is actually a limit case starting way back to a binomial law: if N_0 independent events happen, each with a probability p to be observed, the (discrete) probability density function of the observed number of events N is $P(N) = \frac{N_0!}{N!(N_0-N)!}p^N(1-p)^{N_0-N}$. The average N according to this distribution is $\bar{N} = N_0 \times p$, with a standard deviation $\sigma_N = \sqrt{N_0 \times p \times (1-p)}$. When $p \longrightarrow 0$ the distribution $P(N)$ can first be approximated by a Poisson distribution (if N is not too big) and ultimately (if $N_0 \longrightarrow \infty$) in a Normal (i.e. Gaussian) distribution centered on $\bar{N} = N_0 \times p$: $P(N) = \frac{1}{\sigma\sqrt{2\pi}}\exp(-\frac{(N-N_0)^2}{2\sigma^2})$ with $\sigma = \sqrt{N}$.

Therefore, when one declares that the statistical uncertainty on a measured quantity N_m is $u_{N_m} = \sqrt{N_m}$, the quoted uncertainty is of *type B*. It is assumed that the observed number is the mean value of a Gaussian distribution and that the associated standard deviation (taken as the uncertainty) is $\sqrt{N_m}$. (If you prefer to use 2σ or 3σ as the uncertainty on a quantity, then your statistical uncertainty should be 2 or $3\sqrt{N_m}$).

It is important to remember where the $\sqrt{N}$ uncertainty comes from. It does not hold for very low number of N (use Poisson distribution for these cases), and is even less valid if the observation probability p is high (going back to the binomial law, if $p = 0.5$ for example, $\sigma = 0.5\sqrt{N_0}$).

Finally, not all *numbers* measured follow an *event-observation* logic. If a number of atoms is determined by weighting, there is absolutely no justification to use $\sqrt{N}$ to estimate the uncertainty.

Systematic uncertainty

This covers any source of uncertainty that is not related to limited statistics.

Generally, it comes from the limited precision of an instrument, and the value is always the same (over repeated measurements) and cannot be reduced further down by recording more data in the same experimental condition. As it is, the Systematic uncertainty is *irreducible*, while the so called *statistical* one, by nature, can be reduced.

The systematic uncertainty may still be determined by statistical methods (*type A*).

Warning: Reducing the overall uncertainty is not always just a matter of increasing the statistics. A more intense beam, longer recording times, ... may come with increased systematic uncertainties (from increase background or parasitic reactions, ...). In other words, a gain in statistics may come at a price in systematics.

Note: The *statistical* uncertainty on a measurement can become a *systematic* source of uncertainty on another one. For example, when determining the activity of a source, the limited statistics will be a *statistical* uncertainty on the recorded value for the activity. Later, when using the source to determine an efficiency, the uncertainty on the activity

will be a systematic one (but multiple measurement might be used to determine the systematic Type A uncertainty on the efficiency via statistical method).

Note: For a value obtained at the end of a data analysis process, one may prefer the labels *intrinsic* and *parametric* uncertainties (which we will use later in this manuscript). The latter is the uncertainty associated with the uncertainties on analysis parameters (cuts and selections, scaling and normalizations, ...), while the first is the uncertainty that comes directly from the analysis process (e.g. results of fits, ...) and is "intrinsic" to the dataset (including its limited statistics).

Whatever the scale of the different sources of uncertainties into the final results, the total combined uncertainty on the measurement is the quantity that matters. It is not particularly important whether most of the uncertainty is due to systematics factors or the limited statistics. Given the same final combined uncertainty, one measurement will not be "better" if it boasts a smaller systematic uncertainty.

However, splitting between the two *kind* may be justified for data evaluation, where they can be processed differently.

3.1.2 Covariance, why it matters?

We hinted at covariance just above. Covariances are, to put it simply, the links between values. For illustration, let's imagine we measure a quantity (let's say a cross-section) at two different energies.

Our experiment and data analysis will yield two values σ_1 and σ_2, with their respective uncertainties u_{σ_1} and u_{σ_2}. The quantities σ_i are derived from number of counts N_i (and the associated uncertainty u_{N_i}) by $\sigma_i = N_i/\epsilon$ with ϵ a common parameter between both values. Since the two values have been obtained with the same setup, using the same procedure of data analysis, their uncertainties are *linked* together by the common parameter ϵ.

The efficiency itself has an uncertainty u_ϵ, and we can express the uncertainty on our cross section (here we use the simplified square summation formula) by ${u_{\sigma_i}}^2 = \left(\frac{u_{N_i}}{\epsilon}\right)^2 + \left(\frac{u_\epsilon N_i}{\epsilon^2}\right)^2$. If the dominant source of uncertainty in our experiment is the detection efficiency, then we can neglect the uncertainty on N_i.

The relative uncertainty u_{σ_i}/σ_i is therefore approximately given by u_ϵ/ϵ. It is easy then to understand that if we underestimate the central value of ϵ, both σ_1 and σ_2 will be overestimated by the same relative amount.

Or, to summarize: the common source of uncertainty creates a *leverage* between uncertainties of different variables.

Covariances to interpret experimental data

The knowledge of covariance is a powerful tool for interpretation of experimental data. Let's take the example in Figure 9 with two experimental points (the σ_i from before), that are compared to two theoretical model predictions.

If one tries to distinguish which of the two models fits better the experimental data, without considering the covariance, one can for example computes the χ^2 value for both models and compare. In the presented example, the values are 0.782 for model 1, and 0.784 for model 2. In other words, without covariance information, both models are compatible with the data with the same level of compatibility.

But we know that the uncertainties are linked, just as if uncertainty bars had a *direction* (in other words, it's more like uncertainty arrows). The graph in Figure 9 is therefore better when represented as in Figure 10.

Generalized χ^2 formula using covariance

The generalized formula for computing the χ^2 using covariances uses a matrix notation: $\chi^2 = (X - \bar{X})^T Cov^{-1} (X - \bar{X})$ with X the vector of values, $\bar{X}$ the vector of model values to compare to and Cov the covariance matrix.

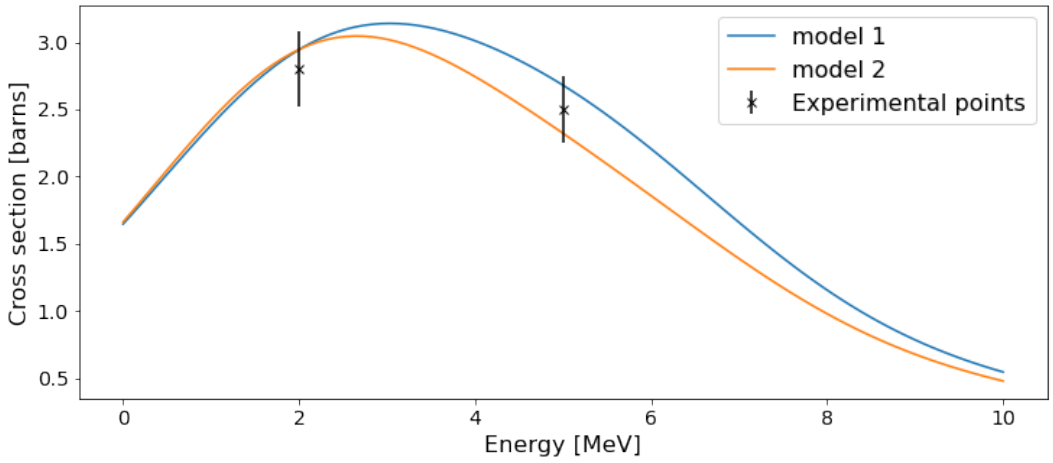


Figure 9: Example of two experimental points compared to two different models.

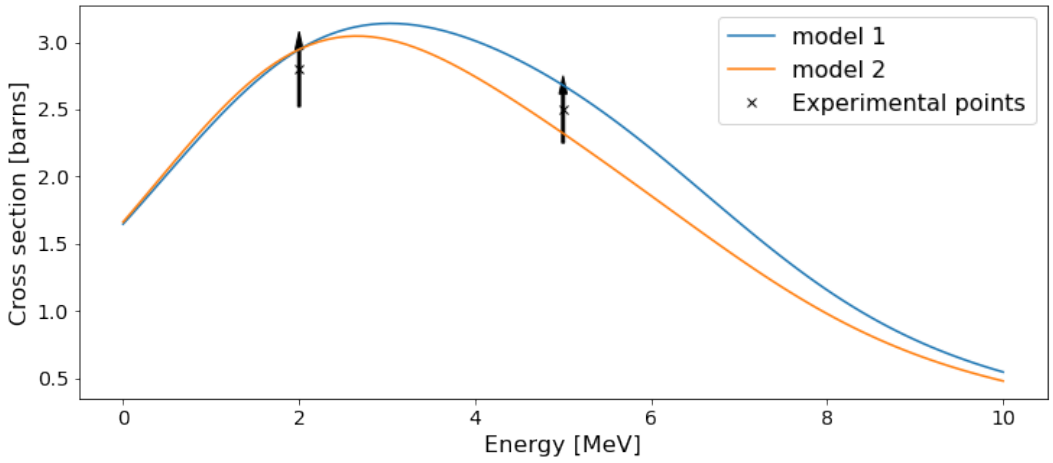


Figure 10: Example of two experimental points compared to two different models (like Figure 9). Here, the error bars are replaced with *arrows* that represent the covariance between the two points.

Tip: For two variables only, x and y with uncertainties u_x and u_y respectively, and $r_{x,y}$ the correlation between the two variables (i.e. $cov(x,y) = r_{x,y} \times u_x \times u_y$), the generalized χ^2 is computed by the formula: $\chi^2 = \frac{{u_y}^2}{{u_x}^2{u_y}^2-(r_{x,y}u_xu_y)^2}(x-\bar{x})^2 + \frac{{u_x}^2}{{u_x}^2{u_y}^2-(r_{x,y}u_xu_y)^2}(y-\bar{y})^2 - \frac{r_{x,y}u_xu_y}{{u_x}^2{u_y}^2-(r_{x,y}u_xu_y)^2}(x-\bar{x})(y-\bar{y})$. It's a bit of a mouthful, but if $r_{x,y} = 0$, we fall back on the classic uncorrelated χ^2

When we take into account the covariance (arbitrarily assuming a 0.95 correlation between the two variables), the obtained χ^2 values are respectively 0.0002 and 1.5704 for model 1 and 2 respectively[7]. **Just by considering the extra information of covariance, we can now say that the first model has a high compatibility with the data, while the second one is much less compatible.**

3.1.3 The important thing about uncertainties

By definition, the uncertainties cannot be *known exactly*[8], they are *estimated* or *evaluated*. In any case, to correctly use the information from the uncertainties, as well as the covariance and correlation matrices, it is important to know how they have been constructed. To put it another way: the way the uncertainties are obtained is as much important as the uncertainty values themselves.

That is why it is important to fully document the sources of uncertainty in a result, and how they are combined to produce the final, published, uncertainty.

The principles of *Open Science*, whether it is *Open Data* or *Open Source*, are a great way to ensure the maximum documentation on how the uncertainties were produced.

Following this idea, the *Implementation of the Monte Carlo analysis* chapter will describe in details how a *full Monte Carlo analysis code* is designed and produce central values, uncertainties, covariance and correlation matrices. That code is available in *Open access* and can be completely examined to understand the way the computations are performed.

Such a fully open code, with the ability to be re-executed, answers the need for detail documentation on how the data is analyzed, and the uncertainties produced, that is required for *accurate and meaningful evaluation of experimental values*.

Note: The amount of time during which the data, code, ... should be archive depends on the importance of the data and the ease of repeating the measurement. Still, a properly archived data set and code should be easy to retrieve and reuse.

Dissemination via *Open* channels is also a way to maximize the future availability of the archives, as they are more likely to be copied in different places.

3.2 The Monte Carlo method

3.2.1 Principle

The name *Monte Carlo* was coined by Nicholas Metropolis, John von Neumann and Stanislaw Ulam during their work on the Manhattan Project, as a reference to the Monte Carlo Casino in Monaco[1].

The principle of Monte Carlo calculations is to generate random numbers, use these numbers to perform a simulation, repeat the simulation many times to generate a large sample of results, and finally use statistical methods to estimate the desired quantities (their central values, plus the standard deviations, covariances, ...).

[7] I know the $\chi^2 \approx 0$ can be shocking to some. But here, we are looking at a simplified example, so I made things very clear cut.
[8] Obviously
[1] The Monte Carlo method - Wikipedia.

This type of simulation provides an efficient way to study complex problems by taking into account the uncertainty and possible correlations between input parameters and variables (when done right).

There is not **one** strict definition of what a Monte Carlo calculation should look like, but the general principle generally reproduces the following steps:

1. Draw random numbers from a given *probability density function* to create a new set of parameters.
2. Compute one or several values using the parameters obtained in the previous step, as well as other data. Each computed value(s) is stored in a *stack*.
3. Repeat steps *1* and *2* many times (until convergence).
4. From the values stored in the stack, the average value, standard deviation and (when applicable) covariance matrix are calculated.

The use of Monte Carlo methods is very widespread in nuclear physics. One can note in particular that transport codes like *MCNP*[2] or *Geant4*[3] use Monte Carlo methods. The same method can be used for some evaluation processes[4] or to recreate a nucleus' level scheme[5].

3.2.2 Convergence

At the end of a Monte Carlo calculation, one typically ends up with a series of values $x_1, x_2, x_3, ..., x_N$ with N being a *large* number. The next step is to compute the mean value $\bar{x} = 1/N \sum_{i=1}^{N} x_i$ and standard deviation $\sigma_x = \sqrt{1/(N-1) \sum_{i=1}^{N} (x - \bar{x})^2}$ (the uncertainty associated to x, u_x is taken to be equal to the standard deviation σ_x.

It is obvious that for these values to be relevant, one needs N larger than 3... But how large should it be?

Unfortunately, there is no universal answer to that, and a particular attention should be paid to this issue of *convergence*.

Convergence accomplished?

We say that the Monte Carlo calculation has *converged* when the sufficient number of iterations has been performed (i.e. N is large enough), so that the x_i set reach a certain convergence criteria. The criteria to test for, however, is not unique, and is left to the appreciation of the person performing the calculation. Depending on the situation, one or more criteria will be appropriate to apply.

Warning: An important thing to keep in mind when choosing a criteria is that the stricter it is, the harder it will be to achieve. That means calculations may run for a very long time and require an increasingly big amount of CPU time and memory storage (for the intermediate results stack). A poorly chosen criteria might even make your Monte Carlo code run *forever*. A failsafe should always be implemented.

We will now describe some of the possible criteria.

[2] The MCNP Code .

[3] "Geant4—a simulation toolkit", J. Allison *et al.*, Nucl. Instrum. Meth. A 835 (2016) 186-225 https://doi.org/10.1016/j.nima.2016.06.125

[4] Citation needed

[5] Litaize, O., Serot, O. & Berge, L. "Fission modelling with FIFRELIN". Eur. Phys. J. A 51, 177 (2015). https://doi.org/10.1140/epja/i2015-15177-9

Number of iterations

The simplest convergence criteria is setting a strict number of iteration to run. It offers the safety of always stopping at the same point, but does not guarantee that the number chosen will be large enough to have a statistically relevant $\{x_i\}$ set to work with.

Having a (very) large maximum number of iterations in combination with another criteria is a good failsafe for calculation. It ensures the code will eventually stop.

Target uncertainty on the mean value

The uncertainty on the mean value $\bar{x}$ (we are not talking about the standard deviation here) is $\sqrt{1/N \sum_i^N var(x_i)}$ where $var(x_i)$ is the square of the uncertainty associated with the obtained value x_i[6]. If all x_i have the same uncertainty ς_x, then the uncertainty on the mean value is $\varsigma_x/\sqrt{N}$.

This is a well known results that the more data points one have, the easiest it is to determine the mean value of the points.

This criteria will lead to a reasonably quick convergence (for example, $N = 100$ leads to an uncertainty on the $\bar{x} \approx \varsigma_x/10$). However, it tells us nothing on the standard deviation or covariance aspect of the result.

Target standard deviation

The Monte Carlo calculation may quickly converge on the mean value, but the standard deviation of the $\{x_i\}$ set might still be very large (for example, if the x_i values are evenly distributed around $\bar{x}$ but with a very broad distribution). Let's remember that it's the standard deviation of the final $\{x_i\}$ set that counts, much more than the ease of finding the central value.

So, it is a good idea to target the standard deviation and continue the Monte Carlo iterations until we reach $\sqrt{1/(N-1) \sum_{i=1}^N (x - \bar{x})^2} \leq \sigma_{\text{target}}$.

One has then to choose the target standard deviation σ_{target} wisely. If it is too small, the value will never be reached and the code may run forever. Also, and more importantly, it has to have significance: is the target standard deviation the value needed to be compared to another results? To discriminate between theories?

Step to step change in mean value

What if we do not have a way to set a significant target for the mean value uncertainty or standard value? That may happen when there is no other value to compare to.

In that case, one can *observe* the step by step (possible averaged over some number of steps) change in the result. Convergence can be considered achieved once the calculated mean value is *stable*, i.e. $|\frac{1}{P} \sum_{i=1}^{P} x_i - \frac{1}{P'} \sum_{i=1}^{P'} x_i| \leq \epsilon$, with a value of ϵ chosen accordingly.

With this convergence test, one may add the condition of a minimum number of steps, as the random process might create the first few steps to lead to mean values close enough to *trigger* the convergence.

[6] "Analyse de données en sciences expérimentales", Benoit Clément, Dunod, 2012. ISBN:9782100575695

Step to step change in standard deviation

The same method can be applied to the standard deviation, with a target *stability* in the σ_x. Both *step-to-step* changes on $\bar{x}$ and σ_x are complementary and can be applied together.

When to test for convergence

The convergence test may be applied "dynamically" (at the end of each step, to see if it is necessary to continue), or *a posteriori*, to check if the number of performed iterations was enough.

When applying the same Monte Carlo calculation to several quantities, it's important that the convergence criteria and number of iterations between each stay consistent. Ideally, one will process all with the maximum number of iterations needed to ensure that convergence is reached for all.

3.2.3 Full Monte Carlo data analysis to produce uncertainties and covariances

By *Full Monte Carlo*, we understand that the data analysis is completely done within a Monte Carlo loop. It is opposed to *partial* Monte Carlo analysis, where only a subset of analysis steps are performed using random sampling. For example, for the *DNR* group, I developed with F. Claeys during his PhD. (2019-2023) an *intermediate data* semi-Monte Carlo code[7,8and9].

The *workflow* of a *full Monte Carlo analysis* is given in Figure 11.

Starting with the distributions associated with each parameter, as well as the raw data (for performance's sake, all the preparatory steps and anything that does not require sampling parameters is done before entering the Monte Carlo loop) are prepared.

At the start of each iteration, the relevant parameters are sampled according to their given distribution. Then, *the analysis is performed* automatically (if the analysis requires human input, the iteration process will be much slower - but it is still possible), starting from the most *raw* spectra, up to the final value. If / When uncertainties from some analysis steps are generated in the analysis (typically, uncertainty from a peak fit in a spectrum), this *intrinsic uncertainty* is carried over and stored with the final value to be combined with the Monte-Carlo uncertainty (from the dispersion of results, i.e. *parametric uncertainty*).

Once all the iterations are done (according to the chosen *Convergence criteria*), a *final stage* gathers all the iteration results, computes their central values, standard deviation (which will be labeled *parametric uncertainty*), and covariances (the covariance is only relevant to the parametric uncertainty, all intrinsic uncertainties are normally independent).

We note that with this method, the correlations from parameters are *automatically* taken into account. That is one of the major advantage of it.

[7] Greg Henning, Maëlle Kerveno, Philippe Dessagne, François Claeys, Nicolas Dari Bako, et al.. Using the Monte-Carlo method to analyze experimental data and produce uncertainties and covariances. 15th International Conference on Nuclear Data for Science and Technology (ND2022), Jul 2022, Online Conference, United States. pp.01045, ⟨10.1051/epjconf/202328401045⟩.

[8] "Producing uncertainties and covariance matrix from intermediate data using a Monte-Carlo method." Greg Henning, François Claeys, Nicolas Dari Bako, Philippe Dessagne and Maëlle Kerveno. EPJ Web of Conf., 294 (2024) 05002 https://doi.org/10.1051/epjconf/202429405002

[9] Francois Claeys, "Mesure, modélisation et évaluation de sections efficaces à seuil (n, xn γ) d'intérêt pour les applications de l'énergie nucléaire". Thèse de doctorat, 2023. Chapter 2.4 "Resimulateur" https://theses.fr/2023STRAE027

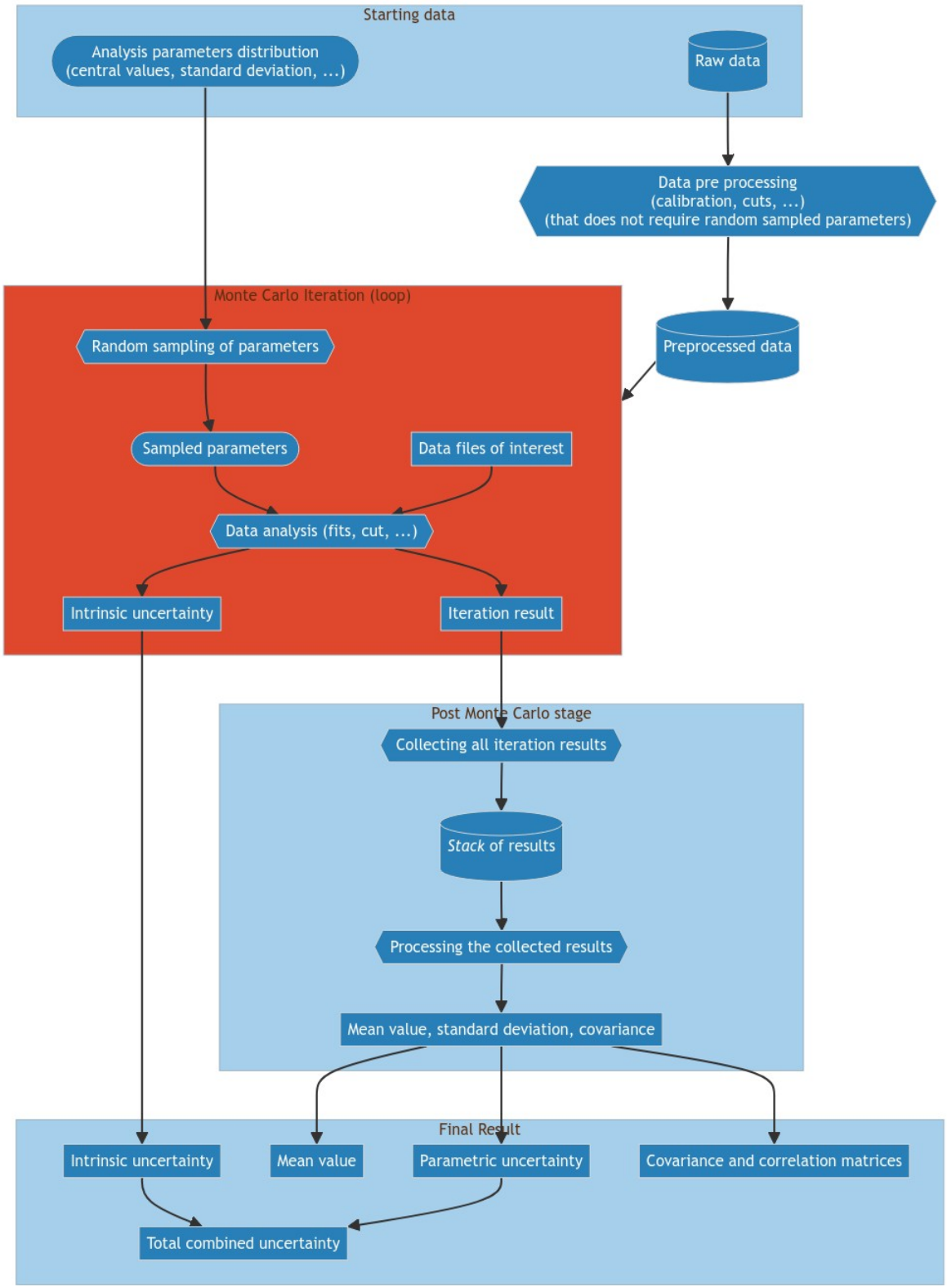


Figure 11: Schematic *flow chart* of a full Monte Carlo Analysis. The *Monte Carlo "loop"* is highlighted in red.

CHAPTER

FOUR

ANALYSIS METHOD AND PRACTICAL IMPLEMENTATION

4.1 Method of experimental data analysis

Note: The following section describes the overall principles of analysis for context. More detailed description can be found in numerous references from past thesis and articles[1,2, and3].

The analysis method for *Grapheme*[4 and5] (or other similar setups[6]) combines precise γ-ray spectroscopy with Germanium detectors, neutron detection with a fission chamber, *time-of-flight* measurements (for neutron energy determination), and the *Gauss method* for solid angle integration[7].

4.1.1 The Grapheme setup

For the recording of the ^{183}W data, *Grapheme* was consisting of :

- Two fission chambers, upstream from the target. Both use a highly enriched ($\geq 99.5\ \%$) ^{235}U, thin (≈ 600 nm), in the form of UF_4 and U_3O_8 deposits[Page 27, 1].
- Four planar High Purity Germanium detectors with diameters between 3.5 and 6 cm, and thickness 2 to 3 cm. The detectors are named after colors in French: *Bleu*, *Vert*, *Rouge* and *Gris*.

Figure 12 gives a schematic view of the setup. Reference[1] extensively describes the detectors.

The detectors are connected to a digital acquisition[8] which records the events in *list* mode, without condition of multiplicity, but requiring a time coincidence (within 7 microseconds) with the beam pulse signal (that is also connected to the acquisition input).

[1] Jean-Claude Thiry. Measurement of (n, xn γ) reaction cross sections of interest for the Generation IV reactors. Université de Strasbourg, 2010. <http://www.theses.fr/2010STRA6144>

[2] Eliot Party. Etude des réactions (n, xn) pour les noyaux fertiles / fissiles du cycle du combustible innovant au Thorium. Université de Strasbourg, 2019. ⟨tel-02496853⟩

[3] Francois Claeys. Mesure, modélisation et évaluation de sections efficaces (n, xn γ) d'intérêt pour les applications de l'énergie nucléaire. Université de Strasbourg, 2023. https://theses.fr/2023STRAE027

[4] Greg Henning et al., "GRAPhEME: A setup to measure (n, xn γ) reaction cross sections," 2015 4th International Conference on Advancements in Nuclear Instrumentation Measurement Methods and their Applications (ANIMMA), Lisbon, Portugal, 2015, pp. 1-9, doi: 10.1109/ANIMMA.2015.7465505.

[5] "How to produce accurate inelastic cross sections from an indirect measurement method?" Maëlle Kerveno, Greg Henning, Catalin Borcea, Philippe Dessagne, Marc Dupuis, et al. . EPJ N - Nuclear Sciences & Technologies, 2018, 4, pp.23. ⟨10.1051/epjn/2018020⟩.

[6] Maëlle Kerveno, Bacquias, A., Borcea, C. et al. "From emissions to (n,xn) cross sections of interest: The role of GAINS and GRAPhEME in nuclear reaction modeling. "Eur. Phys. J. A 51, 167 (2015). <https://doi.org/10.1140/epja/i2015-15167-y>

[7] C.R Brune, "Gaussian quadrature applied to experimental γ-ray yields". Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment, Volume 493, Issues 1–2, 2002, Pages 106-110, doi: 10.1016/S0168-9002(02)01552-8

[8] L. Arnold, R. Baumann, E. Chambit, M. Filliger, C. Fuchs, C. Kieber, D. Klein,P. Medina, C. Parisel, M. Richer, C. Santos, and C. Weber. in Proceedings of RealTime Conference, 14th IEEE-NPSS (2005) pp. 265-269, 2005.

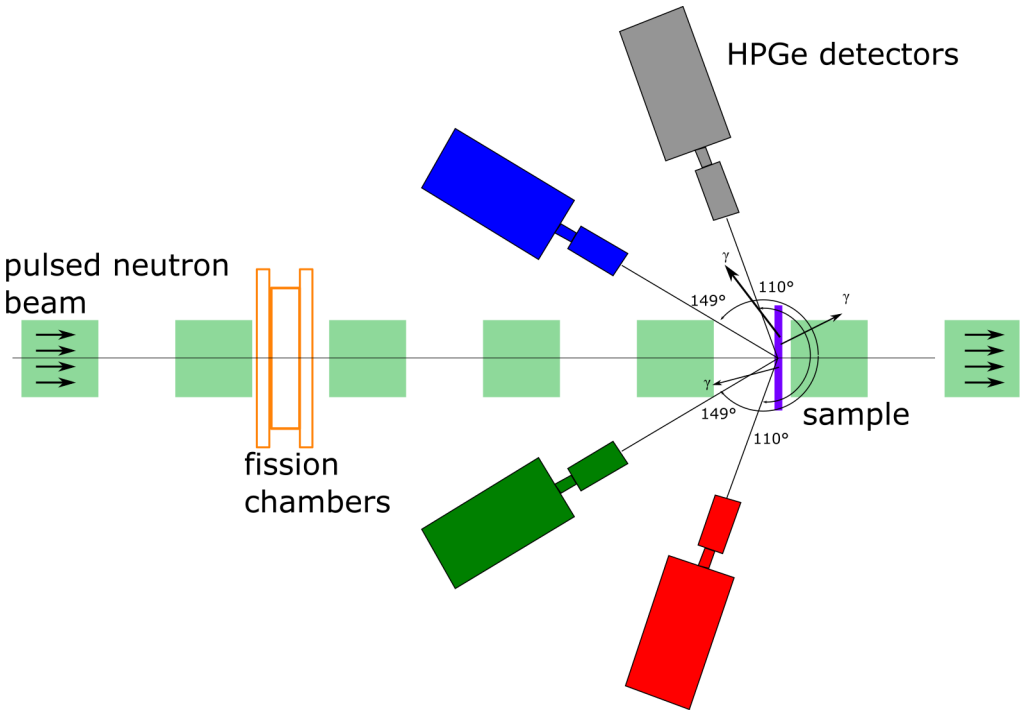


Figure 12: Schematic representation of the *Grapheme* setup. (Adapted from[1].)

The isotopically enriched (80.9 %) ^{183}W target, with a 7.1 centimeters diameter (39.6 cm^2 area) (compared to $\approx$ 55 mm diameter of the neutron beam[Page 27, 1]) and 1.2 millimeters thickness, was placed in the beam axis, at the center of the HPGe detectors (the distance from the target center to the front of the HPGe detectors were between 13 and 21 centimeters).

The data is processed into *2D histograms*: time-of-flight *vs.* γ energy (Figure 13).

4.1.2 Selection of the reaction channel with γ rays

The use of High Purity Germanium detectors in the setup allows for a clear selection of the reaction channel. (Although, it may become *tricky* for highly radioactive and/or fissile isotopes because of a large number of contaminant γ lines. Elements like ^{183}W, with many low-intensity transitions, also require a precise selection work in the γ spectrum.) Using the structure information[9], we know what particular γ ray to look for in the spectrum from the HPGe detector, signing unequivocally a particular $(\mathrm{n},\ x\mathrm{n}\ \gamma)$ channel (see in Figure 13).

In the cases of non isotopically pure targets, the γ from a $(\mathrm{n},\ \mathrm{n'})$ reaction on one isotope may overlap with the *same* γ from the $(\mathrm{n},\ 2\mathrm{n})$ reaction on the one neutron heavier isotope in the material, but both contribution can usually be disentangle with incident neutron energy consideration. In some very rare cases, the level scheme presents two γ rays with energies so near that they cannot be separated, in that case, a *sum* cross section will be extracted.

[9] Evaluated Nuclear Structure Data File

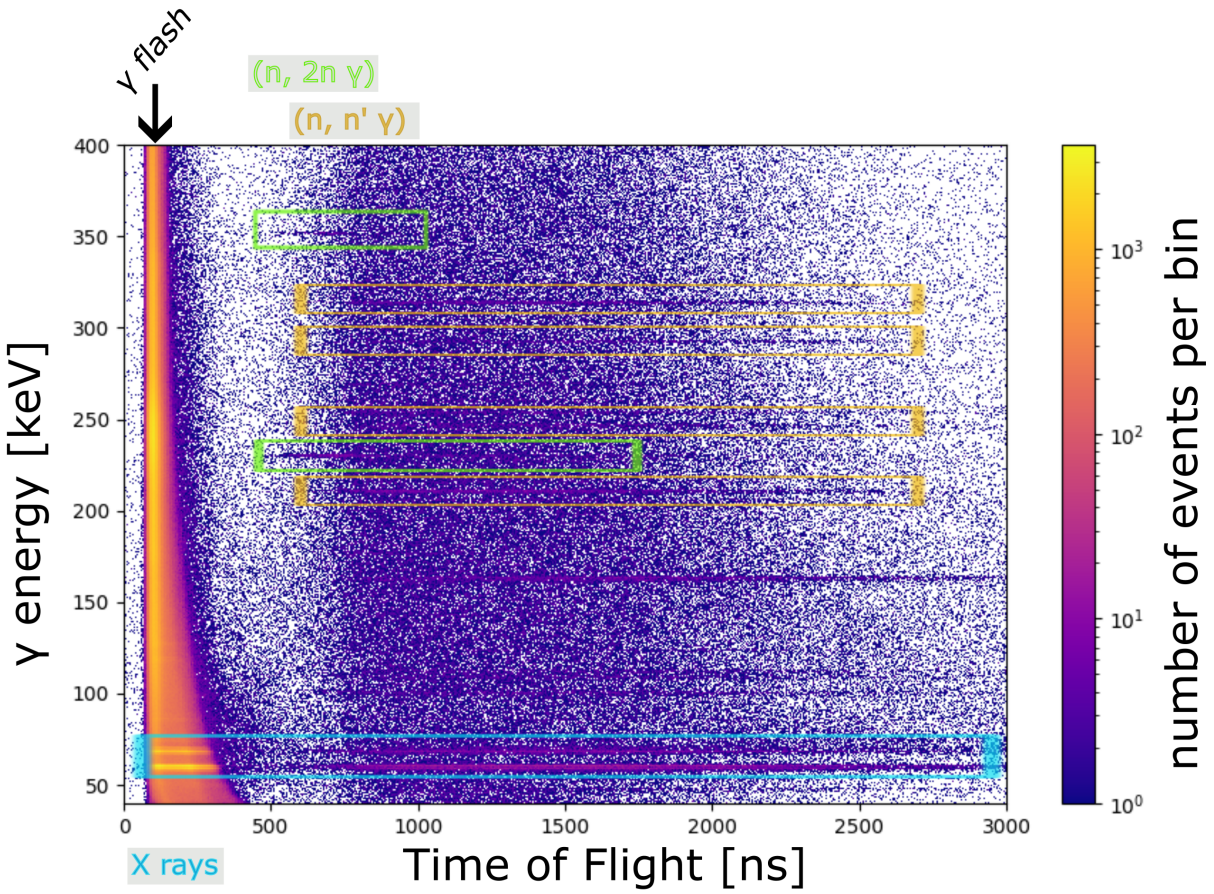


Figure 13: Example of time-of-flight *vs.* γ energy matrix, from the detector "*Vert*". The different reaction channels can be identified. Two $(\mathrm{n},\ 2\mathrm{n}\ \gamma)$ transitions are identified in green, and four $(\mathrm{n},\ \mathrm{n}'\ \gamma)$ in orange. The big concentration of events at the earliest time of flight is the γ *flash*. The low energy, continuous lines at the bottom are X rays from the environment (in particular, the lead shielding around *Grapheme*).

4.1.3 Determination of the incident neutron energy by *Time of flight*

There is no direct measurement of the energy of the neutron that induced the reaction. We only detect the outgoing γ rays. Furthermore, the neutron beam at the *Gelina* Facility[10,11 and12] is *white*, i.e. there is a continuous spectrum of neutron energy (read more in[Page 30, 12]). To determine (with a given accuracy) the energy of the neutron that induced the inelastic scattering for which a γ ray is detected, we rely on the neutron time of flight. Indeed, the accelerator that beams electrons on the neutron production target provides a *pulse* signal that is inputted in our digital acquisition setup and used as `start` signal. The detection of events in our HPGe detectors is considered as the `stop` and from the time elapsed between the two (after a correction for electronic processing and signal travel, that is determined using the γ flash detected in our setup), one can easily find the energy of the neutron E_{n}, using the formula $E_{\mathrm{n}} = m_{\mathrm{n}}c^2\left(\left[1-\left(\frac{DoF/ToF}{c}\right)^2\right]^{-1/2}-1\right)$ (where m_{n} is the mass of the neutron in MeV/c^2, *DoF* and *ToF* the distance and time of flight, respectively).

Note: Because of this formulation, a constant uncertainty on the time (or distance), will have an energy dependent impact on the neutron energy uncertainty. For a distance of about 30 meters, and a time resolution of 10 nanoseconds, the energy resolution will go from 1 keV for a 100 keV neutron, up to around 2.7 MeV for 20 MeV neutrons.

Below is a table giving a few reference uncertainties.

Neutron Energy (keV)	Neutron energy uncertainty (keV)
100	1
500	10
1000	30
5000	350
10000	1000
20000	2700

4.1.4 Normalization to neutron flux with fission chamber

The absolute number of neutron going through the target is determined using a *fission chamber*. It is, in fact, a ionization chamber with a thin ($\approx$ 600 nm) deposit of Uranium 235[Page 27, 1]. The neutron passing through will, with a small probability due to the low amount of material, induce fission of the Uranium in the deposit. The fission fragment (usually only one of the two will get out of the material, for geometric reasons) will then escape into the gas within the high voltage bias region and induce an electric signal that will be detected. Counting the number of recorded signals with an amplitude high enough to be the signature of a fission fragment (against a constant background of α particle emission), also combined with the time of flight determination, gives us a yield of fission events as a function of neutron energy, after correction for the fission chamber efficiency ($\varepsilon_{\mathrm{FC}}$) which is of the order of 90 %. Normalization by material quantity ($N_{^{235}U}$) and the reference cross section of neutron induced fission of ^{235}U ($\sigma_{^{235}\mathrm{U(n,\,f)}}(E_n)$) allows us to reconstruct the spectrum of incident neutrons, both in distribution of energy and absolute value (i.e. the number of neutrons that went through our setup during the time of the experiment).

The number of neutrons at and energy E_{n} that traversed the fission chamber can be expressed as :

$$N_{\mathrm{n}}(E_{\mathrm{n}}) = \frac{1}{N_{^{235}U}}\frac{N_{\mathrm{FC}}(E_{\mathrm{n}})}{\varepsilon_{\mathrm{FC}}}\frac{1}{\sigma_{^{235}\mathrm{U(n,f)}}(E_n)}$$

[10] GELINA The European Commission's Linear Electron Accelerator Facility

[11] "GELINA: A modern accelerator for high resolution neutron time of flight experiment". A. Bensussan, J.M. Salome, Nucl. Instrum. Methods 155, 11 (1978). doi: 0.1016/0029-554X(78)90181-7.

[12] "Global characterisation of the GELINA facility for high-resolution neutron time-of-flight measurements by Monte Carlo simulations". D. Ene, C. Borcea, S. Kopecky, W. Mondelaers, A. Negret, A.J.M. Plompen, Nucl. Instrum. Methods Phys. Res., Sect. A 618, 54 (2010) doi: 10.1016/j.nima.2010.03.005.

To get the proper flux, one just needs to divide $N_{\mathrm{n}}(E_{\mathrm{n}})$ by the beam spot area and the time of recording.

4.1.5 Angle integration of the partial cross sections

After determining the number of γ rays for given neutron energy intervals (*which translates to time-of-flight intervals*) at a given angle (the determination is usually down by *fitting* a peak shape on the E_γ histogram, in order to remove background and possible parasitic peaks contributions), the $N_\gamma(E_n;\theta)$ values are combined with the neutron flux and normalization factor related to the amount of material in the target (N_{target}), as well as efficiencies ($\varepsilon(E_\gamma)$), or pile up corrections, we can obtain a partial cross-section at the given angle of the detector, following the formula:

$$\frac{d\sigma}{d\Omega}(E_n,\gamma;\theta) = \frac{1}{4\pi}\,\frac{1}{N_{\text{target}}}\,\frac{N_\gamma(E_n;\theta)}{\varepsilon(E_\gamma)}\,\frac{1}{N_{\mathrm{n}}(E_{\mathrm{n}})}$$

The angle integrated value is then obtained by using the Gauss method, that uses the decomposition of the spatial emission probability of the γ rays (according to their multipolarity) in Legendre polynomials to perform a solid angle integration via a simple weighted sum of partial cross section at specific angles[Page 27, 7].

With two detectors at 110 and 150 degrees in respect to the beam axis[Page 27, 7], the angle integrated cross-section is

$$\sigma(En,\gamma) = 4\pi\left(w_{110^\circ}\frac{d\sigma}{d\Omega}(E_n,\gamma;110^\circ) + w_{150^\circ}\frac{d\sigma}{d\Omega}(E_n,\gamma;150^\circ)\right)$$

With $w_{110^\circ} = 0.6521$ and $w_{150^\circ} = 0.3479$.

This relation is generally true for γ rays connecting levels with well defined parity, it breaks down only if the γ multipolarity is strictly larger than 3 **and** the γ decays for a state with spin larger than 3[Page 27, 7].

4.2 Implementation of the Monte Carlo analysis

4.2.1 Motivation for writing a new analysis code

A previous full Monte Carlo analysis code has been written for the analysis of $^{\text{nat,182,184,186}}$W[1 and2]. Written entirely in C++/*Root*, it works well, but suffers from its own history. Indeed, it had been written piece by piece over the years, without long term planning. Today, it is a very long source file (1140 lines of code), calling many dependencies, using a code that is not fully compatible with the most recent versions of *Root*. Worse, no documentation has been written. Furthermore, it mixes analysis and display functions, contains hard coded values, rely on a *homemade* input parameter file format that lacks flexibility, does not output covariances (correlation matrices can still be obtained using *tricks* and additional work), and does not save intermediate results, making debugging or fine tuning of the analysis cumbersome.

As the *old style* C++/*Root* code was created during the analysis of the $^{\text{nat,182,184,186}}$W data, it is still today very relevant, and the best option, to analyze or reanalyze these data sets. However, the readaptation of the configuration files to a new dataset is so complex that only preliminary data for γ rays from the ^{183}W dataset could be extracted (see later, *Results*). This is typically an example of software development that has been done for a specific purpose (analyzing the even-even tungsten isotopes data), and that is *locked-in* with these datasets[3].

For all this reasons, it has been decided to create a new code from the ground up, relying on the experience gained in writing the previous one. The main objectives of this new analysis software are:

1. Be portable to all platforms by using mostly *Python* and standard libraries, while ensuring version compatibility in the future.

[1] "MEASUREMENT OF ^{182,184,186} W (n, xn γ) CROSS SECTIONS AND WHAT WE CAN LEARN FROM IT." G. Henning, Antoine Bacquias, Catalin Borcea, Mariam Boromiza, Roberto Capote, et al.. EPJ Web of Conferences, 2021, 247, pp.09003. ⟨10.1051/epjconf/202124709003⟩.

[2] Greg Henning, Maëlle Kerveno, Philippe Dessagne, François Claeys, Nicolas Dari Bako, et al.. Using the Monte-Carlo method to analyze experimental data and produce uncertainties and covariances. 15th International Conference on Nuclear Data for Science and Technology (ND2022), Jul 2022, Online Conference, United States. pp.01045, ⟨10.1051/epjconf/202328401045⟩.

[3] It is entirely my fault. I am the one who created this previous code, and have no one to blame but my foolish past-self.

2. Export all intermediate results for later inspection.
3. Use input parameter files in a standardized format (here *Yaml*).
4. Analyze all transitions of interest at the same time.
5. Be designed to be run in a *full Monte Carlo* manner, in order to produce uncertainties and covariance matrices.
6. Be versatile and flexible so that it can be easily ported and reused in the future.

The last point will be possible thanks to a division of the analysis workflow in small tasks and packages, that can be replaced by others for different experiments. For example, the reading of raw TNT[4] data files can be swapped for a *Faster*[5] file reading module, leaving the rest of the procedure the same. If one prefers to avoid long computation time with a full Monte-Carlo, it is also possible to *plug* at the end of the analysis workflow the *intermediate data* semi-Monte Carlo code I developed for the *DNR* group with F. Claeys and described in references[2,6, and7], and *earlier*.

Furthermore, the *philosophy* of the new code is inspired by the *CI/CD* use of *pipeline* chaining stages and tasks together. This is done by, in one part, *cutting down* the whole work in small tasks, but also by running them in inter-dependent stages one after the other (read later: *The orchestration of the different subtasks*).

Finally, by being completely *Open Source*, and coming with a *full documentation*, the code meets the *requirements for accurate and meaningful evaluation* of the values it produces.

4.2.2 Platform and requirements

The analysis has been fully conducted on the *IN2P3* computation center (CC).

At the time of writing this line, the gateway server operates a `CentOS Linux release 7.9.2009` distribution, with 40 `Intel(R) Xeon(R) Silver 4210R CPU @ 2.40GHz` on a `x86_64` architecture[1].

The strength of the IN2P3's CC is the possibility to run batches of commands on a computing platform. Jobs are submitted and managed by the `Slurm` system[2], with computing nodes having the same characteristics as the interactive gateway server. In addition, the running jobs have access to the same file tree as in interactive mode, making testing and preparing the jobs very easy, as no special packaging of the files is needed before launching a job. The downside is that it is not possible to run parallel jobs on the same files without specific duplication of said files into temporary directories for independent running of the tasks (see *Job queuing in Monte Carlo iterations* for the workaround).

Environment

The code written for the analysis is mostly in *Python*, with just a small part in C++ for tasks that would require too much execution time and memory otherwise (plus the use of gnuplot for some simple plotting tasks).

The software, their version and dependencies are managed with the CC `module` tool that load specific version of available softwares. In our analysis, we use *Python* version 3.9.1 as well as `gcc` version 10.3.0, `gnuplot` version 5.4.3 is also loaded to perform some quick and basic plotting.

Note: The default version of different softwares available on the CC platform are tied to different dependencies version. Therefore, one may need to settle for an *older* version of, say, `gcc` in order to accommodate both `Python` and `gnuplot`

[4] L. Arnold, R. Baumann, E. Chambit, M. Filliger, C. Fuchs, C. Kieber, D. Klein,P. Medina, C. Parisel, M. Richer, C. Santos, and C. Weber. in Proceedings of RealTime Conference, 14th IEEE-NPSS (2005) pp. 265-269, 2005.

[5] But not necessarily quicker…

[6] "Producing uncertainties and covariance matrix from intermediate data using a Monte-Carlo method." Greg Henning, François Claeys, Nicolas Dari Bako, Philippe Dessagne and Maëlle Kerveno. EPJ Web of Conf., 294 (2024) 05002 https://doi.org/10.1051/epjconf/202429405002

[7] Francois Claeys, "Mesure, modélisation et évaluation de sections efficaces à seuil (n, xn γ) d'intérêt pour les applications de l'énergie nucléaire". Thèse de doctorat, 2023. Chapter 2.4 "Resimulateur" https://theses.fr/2023STRAE027

[1] Information obtained with the use of *lsb_release -a* and *lscpu* commands

[2] Slurm workload manager

in the environments.

The rest of the analysis tools are python modules that are loaded and installed with the `pip` *tool*[3]. The required modules are listed, as per tradition, in a `requirements.txt` file, as in Listing 1.

Listing 1: `requirements.txt` file for the analysis code. The specific versions of the dependencies are specified to avoid deprecation issues in future versions.

```
doit==0.36.0

PyYAML==6.0

matplotlib==3.7.1
numpy==1.25.0
scipy==1.11.1

pyhisto==0.1.1
```

The python modules are installed into a dedicated *virtual environment* using the standard python library `venv`[4].

The orchestration of the different subtasks is done with the python module `doit`[5], a *task runner* that is in someways similar to `make`[6], but the commands are written in *Python* and allow actual python code to be used when defining tasks. In particular, generic rules using `Path`, loops and configuration files are easy to set up.

Several *tasks* list of commands are defined, each run in order by `doit`, calling on different python scripts. The idea has been to define data processing in scripts outside the `doit` *pipeline*, even if sometime, for the sake of simplicity (*practicality beats purity*), this principle has been twisted a bit.

The parameters of the analysis steps are spread over many different configuration files in *Yaml* format. Using many files, each with a specific parameter, instead of one file gathering all information is the best way to avoid rerunning all tasks dependent on the configuration, even if just one value has been changed.

The overall analysis file structure is in Listing 2.

Listing 2: Top level directory structure of the analysis code repository. As represented by the command `tree -L 1 -F`.

```
.
|-- bin/
|-- data/
|-- etc/
|-- output/
|-- scripts/
|-- tasks/
|-- README.md
|-- cc_setup.sh
|-- requirements.txt
|-- env.run*
|-- run_on_cc.sh
|-- run_task*
```

[3] The package installer for Python
[4] venv — Creation of virtual environments
[5] DoIt Automation tool
[6] GNU Make

The directories and files have pretty self explanatory names.

- `bin/` contains the binary executables files for the small part of the analysis in C++.
- `data/` contains the raw data (TNT binary files[7]), organized by *week* of recording.
- `etc/` stores all the configurations files,
- `outputs/` will contain all the outputs produced by the analysis.
- `scripts/` the python (and bash) scripts that are executed in the pipeline,
- `tasks/` contains the tasks definition files for `doit`.
- `cc_setup.sh` and `run_on_cc.sh` are `bash` scripts to setup the *environment* and run the analysis *pipeline* on the CC machines.
- `requirements.txt` lists the python dependencies to install (Listing 1).
- `env.run` and `run_task` are wrapper executables to launch tasks and scripts easily within the proper python virtual environment (marked with a * in Listing 2, because they have executable rights).

Data handling backbone `pyhisto`

The general data handling is done using the `pyhisto`[8] Histogram structure, which is an histogram library for python, created *in house* to produce histograms that are stored in ascii format[9]. The goal of the module is to be able to handle histograms in python codes (many modules allows the processing of data lists into histograms, such as `pandas`[10], `numpy`[11], ... but the histogram structure, as used in nuclear physics, and *à la Root*[12] are not common[13]).

Data processing

The statistical and random aspects of the Monte Carlo implementation will be taken care of by the `numpy`[Page 34, 11] package. It is a module that is dedicated to the handling of large size arrays. It can therefore compute the arithmetic average (`mean`) and standard deviation (`std`) of a series, as well as covariance (`cov`) and correlation (`corrcoef`) matrices.

By delegating the *mathematics* to an existing library, we use a well tested and documented module, which prevents introducing our own errors. Additionally, this allows us to focus on the rest of the analysis code.

[7] L. Arnold, R. Baumann, E. Chambit, M. Filliger, C. Fuchs, C. Kieber, D. Klein,P. Medina, C. Parisel, M. Richer, C. Santos, and C. Weber. in Proceedings of RealTime Conference, 14th IEEE-NPSS (2005) pp. 265-269, 2005.

[8] Python Histogram module

[9] Ascii format for histograms

[10] Python for Data Analysis

[11] Package for scientific computing with Python

[12] ROOT Data Analysis Framework

[13] The library `hist` (H. Schreiner, S. Liuand A. Goel, "hist". doi: 10.5281/zenodo.4057112.) could do the job, but 1. I became aware of its existence after I started using my own and 2. the export options are limited and not optimized for interoperability (see *later*).

Job queuing in Monte Carlo iterations

The `slurm` job orchestration system runs the jobs directly in the user's *directories*. As *mentioned before*, this has great advantages, as there is no need to package the code in a specific way before launching the job, allowing to have a testing and running environment exactly similar. But that also makes running jobs in parallel more difficult.

The user can either prepare the necessary data and run each job in a dedicated temporary directory, but that brings us back to packaging the code in such specific way and making sure that everything is collected and cleaned up afterward.

Or, and that is the solution that I used in this particular case, the jobs can be run chained back-to-back, one after the other. To launch many jobs in one command, while ensuring job $n + 1$ waits until the job n has finished to start, we use the `--dependency` option of the *slurm* job starting command (`sbatch`), that requires a previous job to terminate before starting the one concerned.

A script that start a given number of time the same job in a dependent *queue* has been written in bash: `start_chain_jobs.sh`.

4.2.3 Analysis flowchart

The general flow of the analysis is given in Figure 14. It is obviously a very simplistic view, each step of this general workflow is divided in many small tasks.

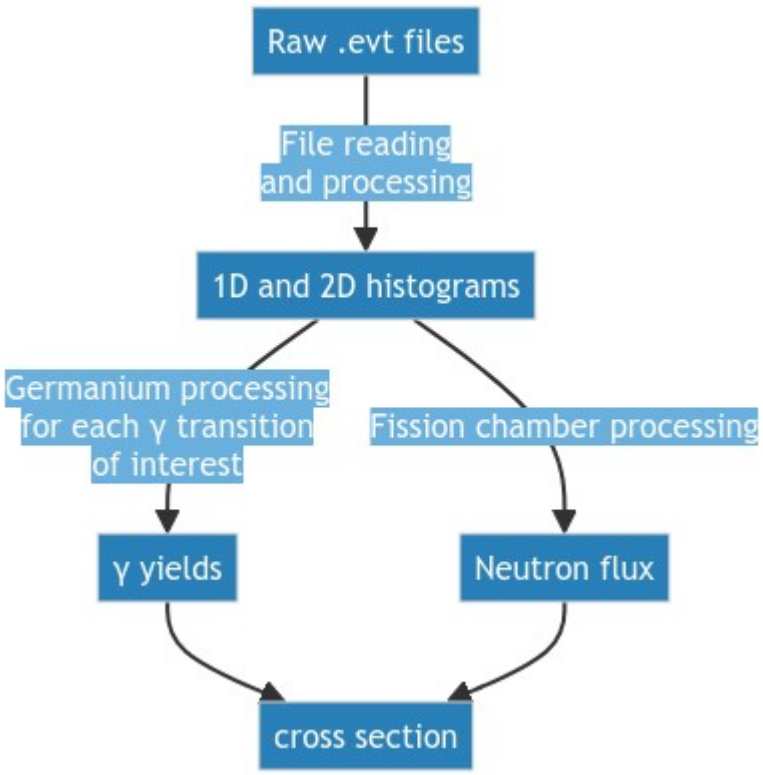


Figure 14: General workflow of data analysis.

The division in small tasks has many benefits. First, it is easier to build, since an *atomic* (in the sense of "*Unable to be split or made any smaller*"[1]) code is much simpler to envision, create, test and maintain than a very large piece of code performing a great number of tasks. Second, it allows a *pipeline* like run, with only the necessary task executed

[1] Definition number 4 in "atomic" on Wiktionary.org.

again, rather than reloading and re-running the whole analysis code. Third, and finally, debugging and fine-tuning of the analysis is easier when all intermediate steps are accessible independently.

Raw files to Histograms conversion

The raw files to histograms (1D and 2D) conversion in itself relies on many steps, including listing the files of interest, converting them from one format to another, performing gain matching (i.e. calibration) and alignments, merging individual files. Additionally, and along side these steps, checks and control are also performed to ensure the quality of the data.

The first steps of file conversion, represented in the flowchart of Figure 15 uses a data list and external parameters to perform the conversion. The data, once converted into histograms (2D and 1D) are checked (by observing the position of the maximum in the spectra for example). In the end, a list of validated data files is constructed, and will be reused in the next steps of the analysis.

For a given week of data recording, there are plenty (like hundreds) of data file (the TNT acquisition[2] splits files in 200 MB chunks). Each raw data file is analyzed (using the `evt2histo.py` python script) to produced four files per channels (two time vs. energy 2D histograms and two 1D histograms), with *four Ge detector channels and 2 fission chamber channels*. In total 24 output files for each 200 MB raw data. Therefore, following the direct conversion, the job of gain matching and aligning histograms starts.

The file `do_raw2histo.py` is the `doit` input files for the generation of histograms from raw data files.

Time alignment and gain matching

Time alignment is relatively easy, since the time spectra contain a characteristic and noticeable peak corresponding to the γ *flash*. Still, the processing has to be done differently for the Germanium detectors and fission chambers. This is because the raw time spectra of the fission chamber display oscillations that make the automatic detection of the γ *flash* position impossible (Figure 16), since no simple "rule" top find the γ flash can be devised for the program. But once the time spectrum has been clean with a simple cut, automatic determination is easy (Figure 17). There's no such issue for Germanium detectors.

Time alignment of Fission chambers

The direct time alignment from a raw time spectrum for a fission chamber is not possible, because of the wide variations *masking* the true γ *flash* (see Figure 16). Therefore, we need first to perform an arbitrary energy cut on the 2D time *vs.* energy histogram to get a *clean* time distribution and find the γ flash easily. (Figure 17). At this point in the analysis, the position of the energy cut does not have a great importance, and it is *eyeballed* by the *user* and entered in a configuration file that is read by the task manager.

Once a *clean* time spectra is obtained (Figure 17), it is easy to automate the search for the γ *flash* (as the first bin in the spectrum above a threshold value). The position of the γ *flash* (expressed in timestamp) is stored in a dedicated file. An intermediate step of controlling the extracted position is then done (by checking that the extracted position is stable from file to file), to ensure the quality of the data (this feeds back into the validated data list).

Finally, from the position of the γ flash in the spectrum and the known distance between the neutron production target and the fission chamber, the time calibration parameters for the spectra are computed and sorted in a file. The parameters files are then read to individually perform the time alignment on all the fission chamber histograms (2D and 1D time).

The flow chart of fission chamber time alignment is given in Figure 18.

The *task file* `do_fc_timealign.py` manages the time alignment of fission chambers.

[2] L. Arnold, R. Baumann, E. Chambit, M. Filliger, C. Fuchs, C. Kieber, D. Klein,P. Medina, C. Parisel, M. Richer, C. Santos, and C. Weber. in Proceedings of RealTime Conference, 14th IEEE-NPSS (2005) pp. 265-269, 2005.

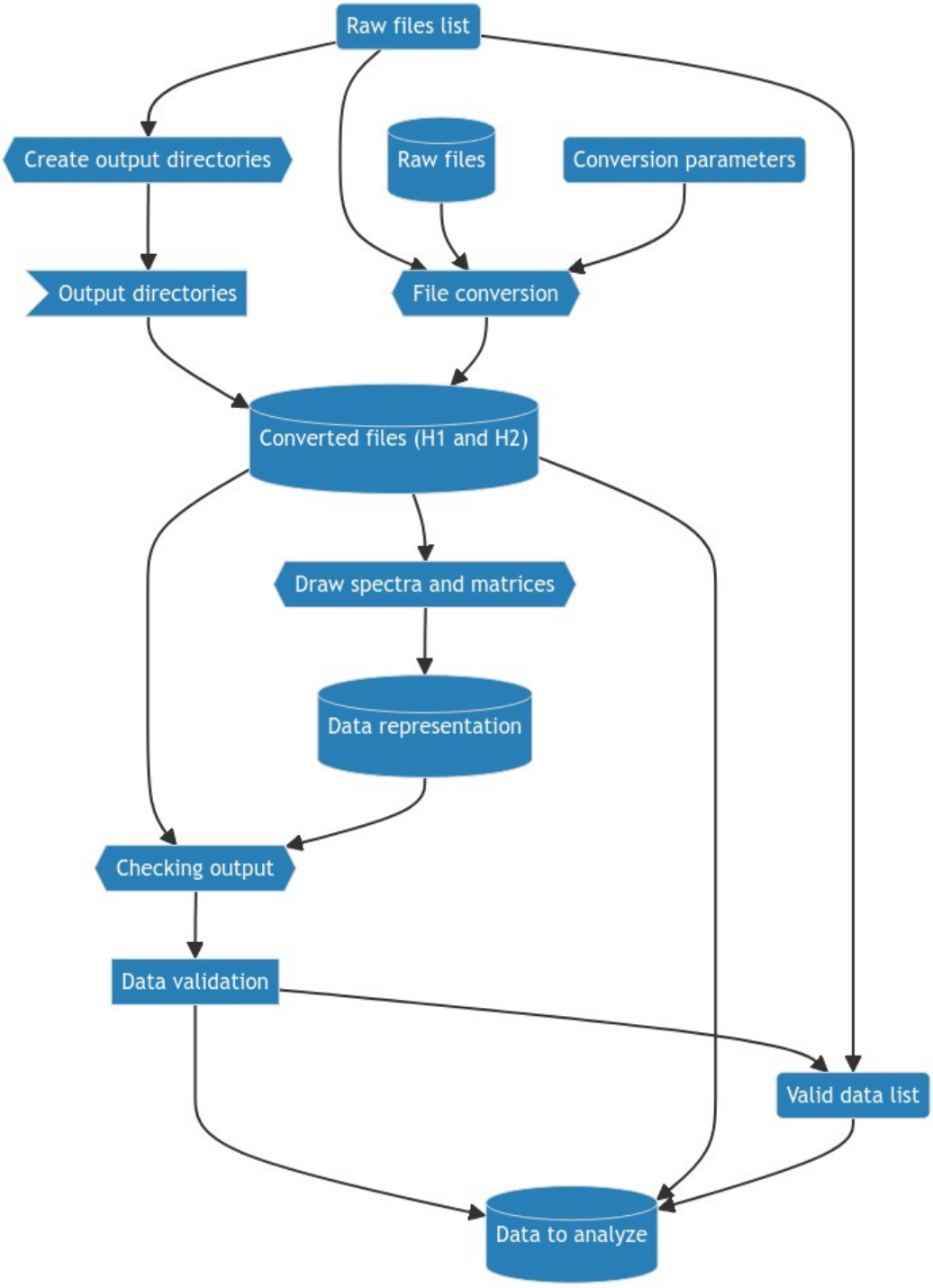


Figure 15: Workflow for file conversion.

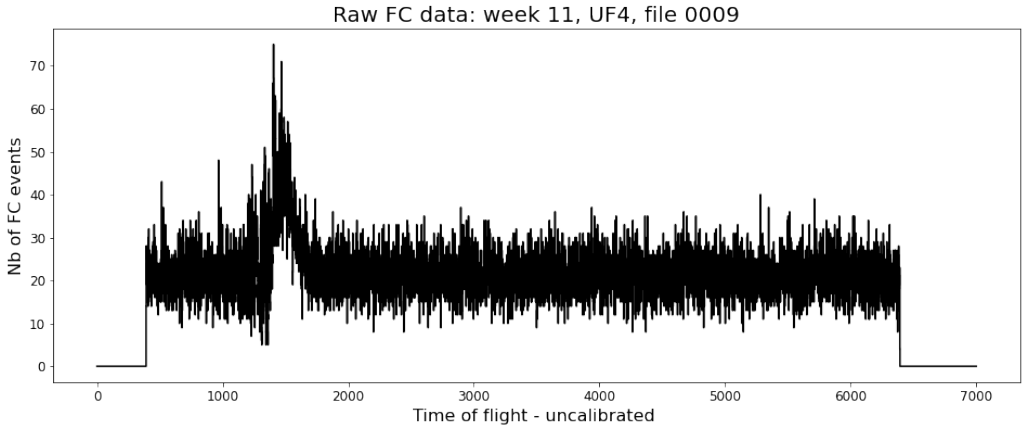


Figure 16: Raw time spectrum for events in the UF_4 fission chamber in one `.evt` 200 MB data file.

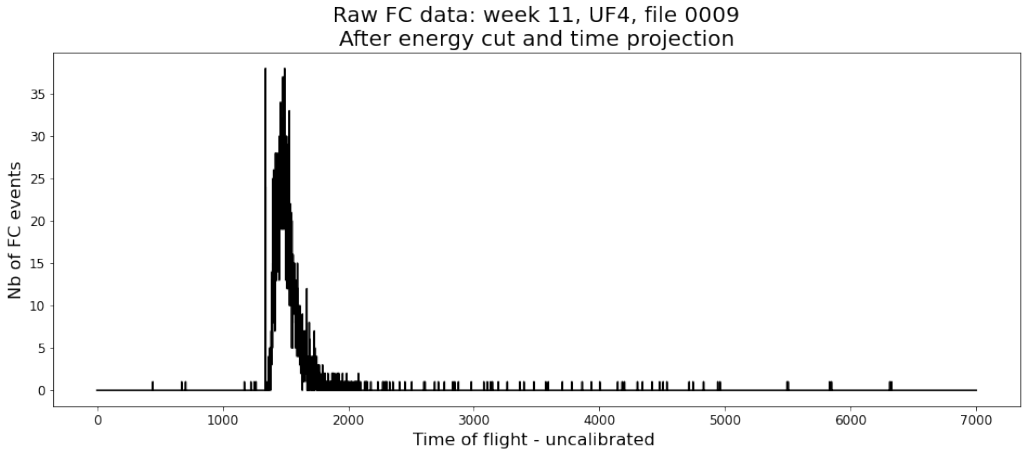


Figure 17: Raw time spectrum for events in the UF_4 fission chamber, after energy cut in the 2D histogram and projection. (Same data as in Figure 16.)

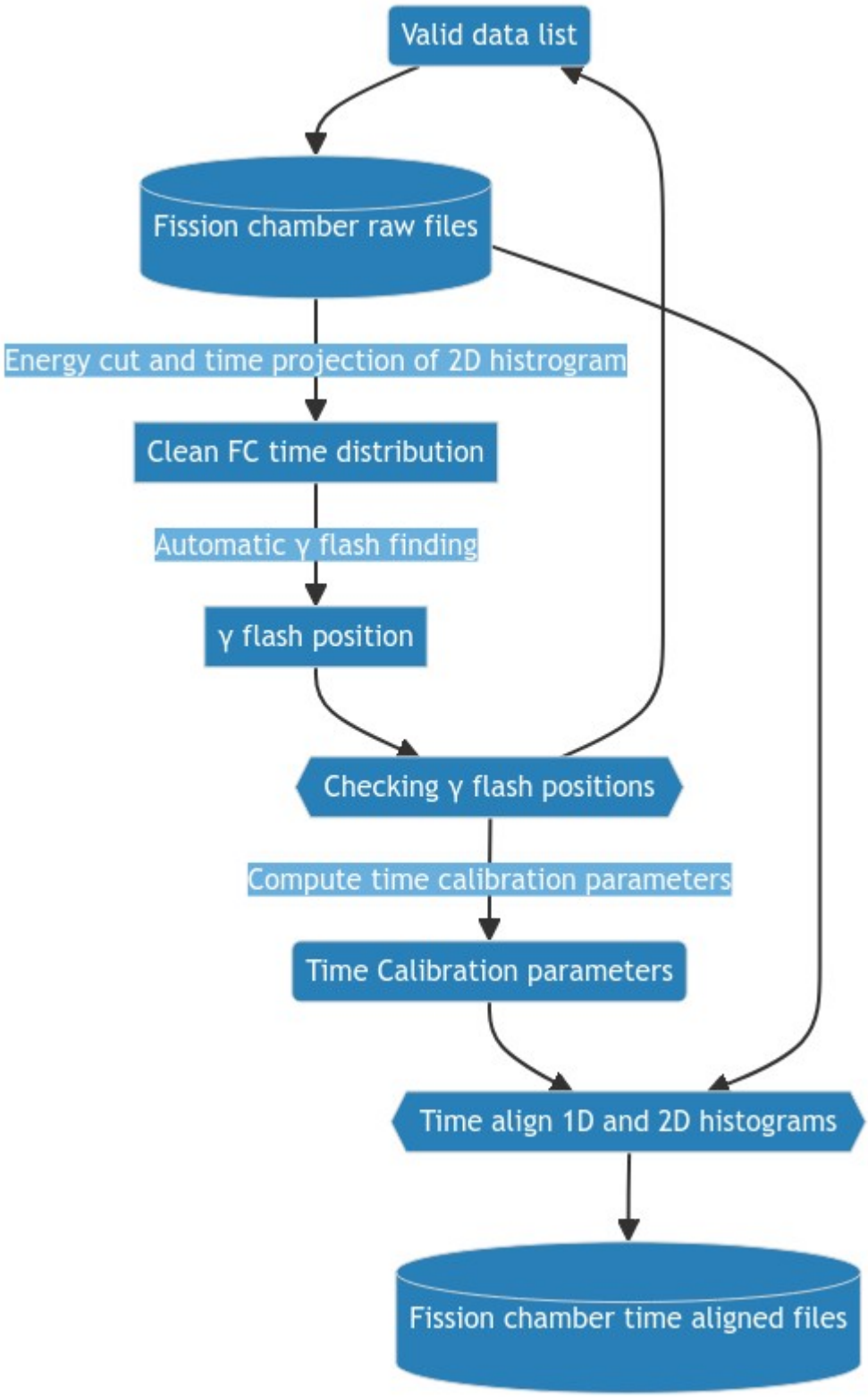


Figure 18: Flow chart for time alignment of fission chamber files.

Gain matching for fission chambers

The resolution of the *energy spectra from the fission chambers* does not require a high precision, as long as the α events at low energy can be distinguished from the fission fragments (much higher energies) (see Figure 23). This is ensured by the correct bias voltage on the fission chambers and the constant gas flow in the chamber. Furthermore, we are not interested in extracting the energy of the fragment. Therefore, no gain matching is necessary for the fission chamber.[3]

Time alignment of Germanium spectra

In germanium detectors spectra, the γ *flash* is directly visible (because the Germanium detectors are there to detect γ rays, which is not the case of fission chambers) (Figure 19). The same procedure as for fission chambers can be used, skipping the 2D histogram energy cut and time projection: Figure 20 .

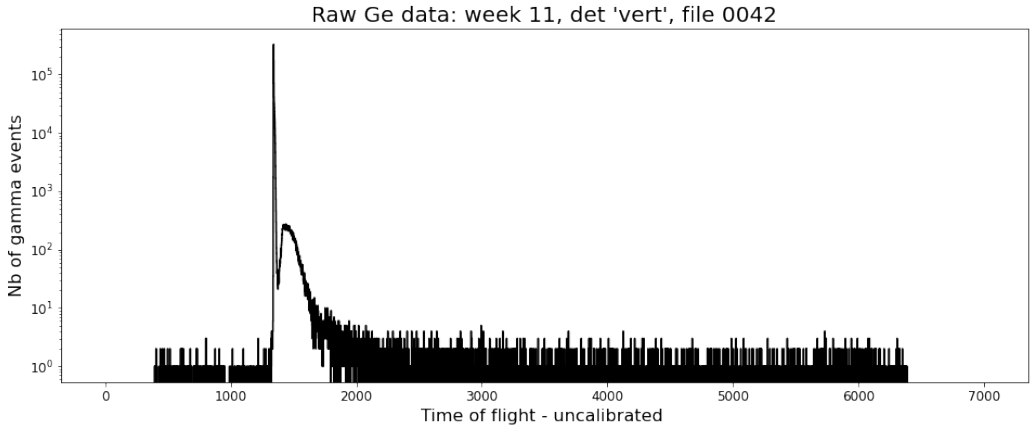


Figure 19: Raw time spectrum for all events in a germanium detector.

The *task file* `do_ge_timealign.py` manages the time alignment of Germanium detectors.

Note: The Germanium detector signals processing last for 4000 nanoseconds, which is reflected in the *pile up* of events. However, the deadtime is negligible compared to the signal rise time (20 to 50 nanoseconds), and given the low count rate in the detectors, there is no loss of data in the recording.

Gain matching for Germanium detectors

In the course of the data recording (several weeks), the gain in the Germanium detectors acquisitions chain (High voltage supply, preamplifier, digital acquisitions input, ...) varies slightly, changing the correspondence between *channels* (as computed by the acquisition) and the actual energy of the detected γ rays. Fortunately, the TNT[Page 36, 2] acquisition used for recording the data, splits the data in 200 MB files. Each file contains enough events to extract a reasonable energy spectrum from the Ge detectors and perform calibration, and corresponds to a length of time small enough (of the order of around 3 hours) that the gain drift is minimal within one file.

Therefore, we can recalibrate the files one by one. This is obviously done automatically, using a rough guess of the calibration coefficients, and a list of peak of interest that will be easily identified in the raw spectra.

[3] This is probably the easiest paragraph to write in the whole manuscript.

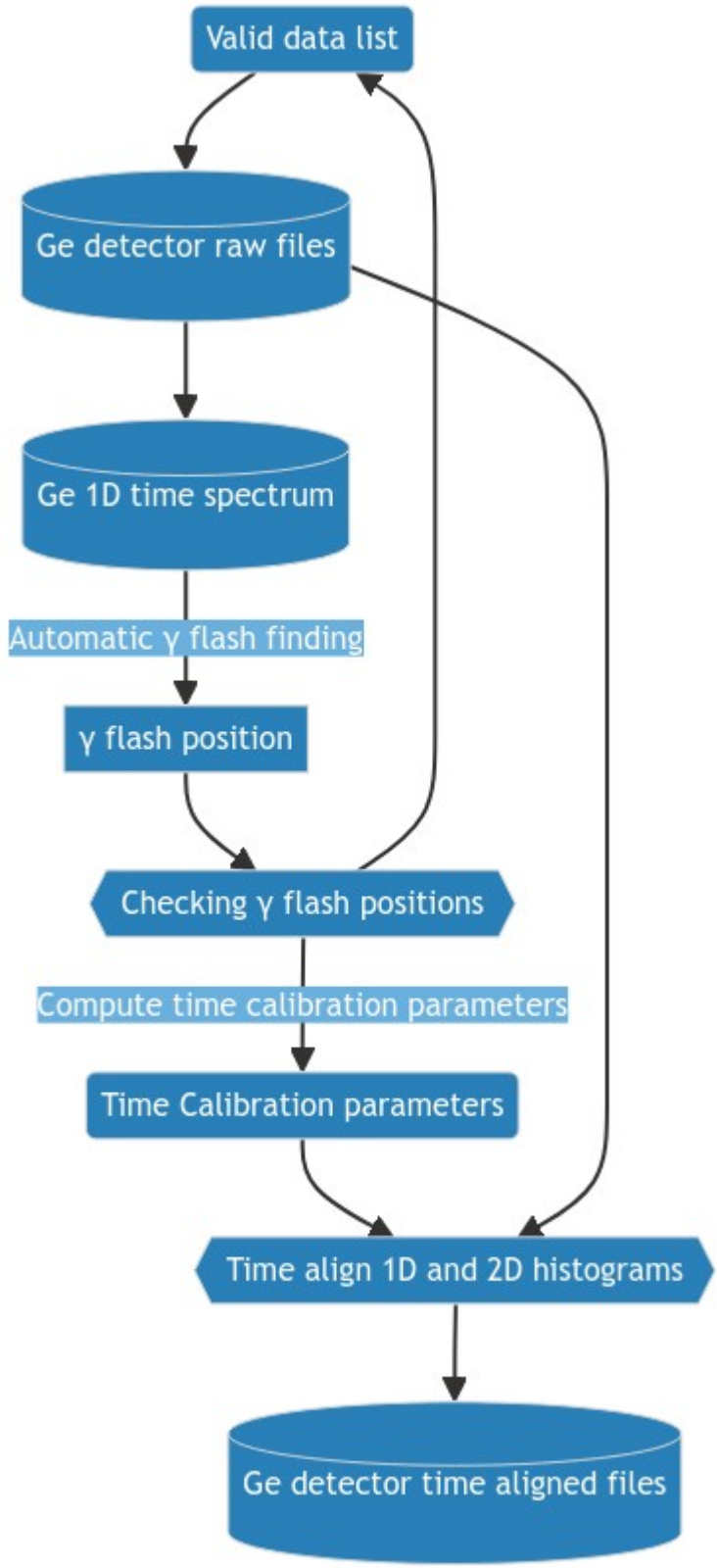


Figure 20: Flow chart for time alignment of Germanium detectors files.

The script `auto-calib.py` performs this, looking for the peaks of interest in the uncalibrated spectra (using the calibration coefficient guess), precisely getting their position and computing the actual calibration coefficients. Once the coefficients are stored in a dedicated file, calibration of the spectrum file is easy (see flowchart in Figure 21).

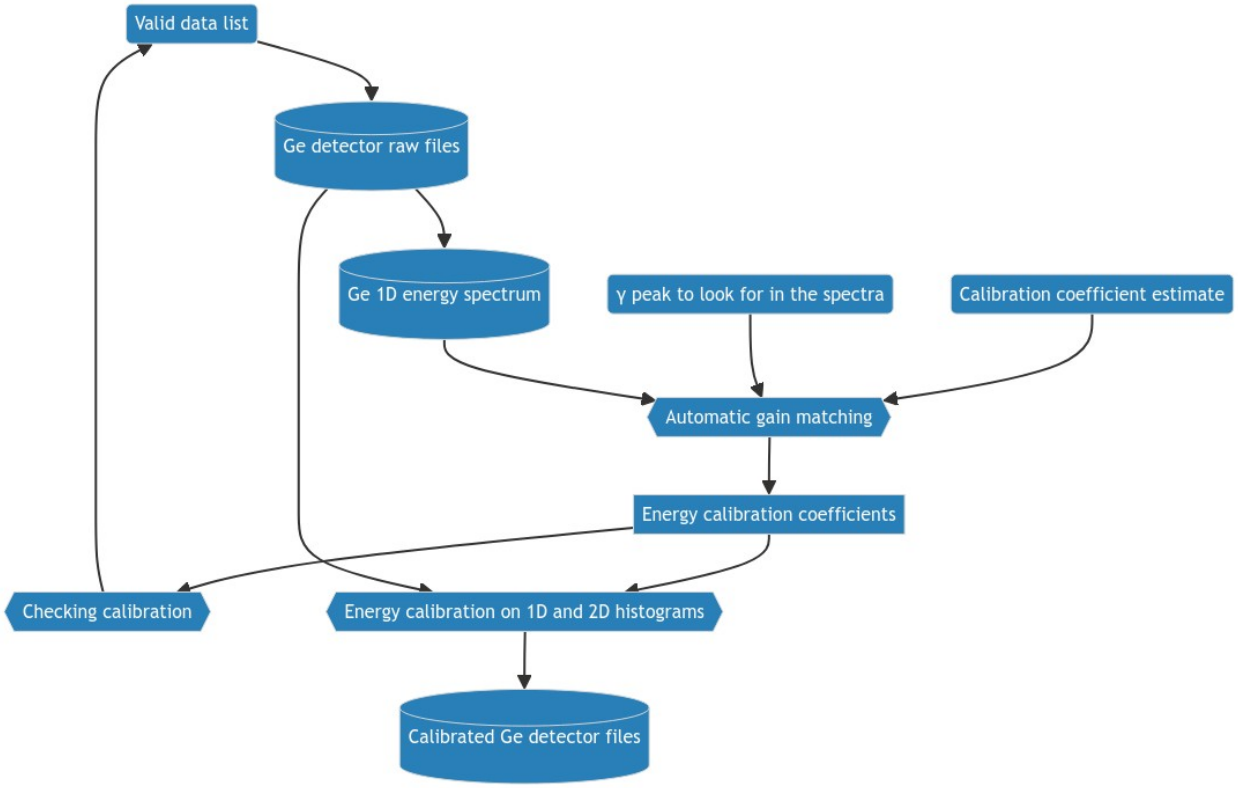


Figure 21: Flow chart for calibration of Germanium detectors spectra.

Calibrated and aligned files merging

In the end, once the files are individually calibrated, aligned, … it is time to put them all together into one, *a.k.a* merging.

The calibration step is relatively easy, with just the scale of the histograms to change (i.e. lower and upper bounds of bins). But because of that, it means different calibrated files will have different bounds and bins. So the merging process requires unifying this.

It is, therefore, done by *filling* an histogram with chosen limits and number of bins, from the many different calibrated ones, as shown in Figure 22.

The 1D histogram merging is performed by a python script `fill_into_histogram.py`. for the sake of speed, the 2D histogram merging is done with a C++ program `fill_into_h2`. (A python option exists, but it runs much more slowly.)

The *task files* `do_ge_calibrate.py` and `do_ge_calibrate_2D.py` are *in charge* of finding the calibration coefficients, calibrating the files and merging them.

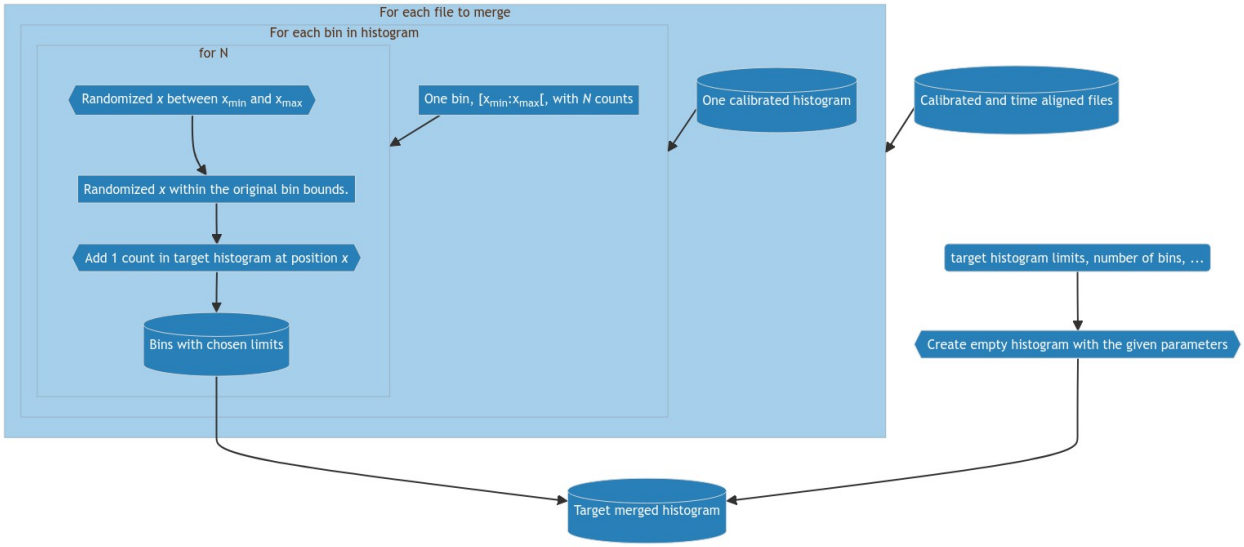


Figure 22: Flow chart for merging spectra.

Neutron Flux determination

Note: We use the term *flux* loosely as we are not going to compute a value in neutrons$/cm^2/s^1$, and rather the absolute number of neutrons that went through the target during the experiment time.

The neutron flux is determined from the fission chamber data by counting the number of events identified as fission as a function of time.

α vs. fission event energy cut

The energy spectra of a fission chamber (projected for all time of flight) is shown in Figure 23. At low energy, there is a significant contribution from the α decays of the Uranium in the fission chamber. The α contribution is constant in time and cannot be removed by time of flight selection. Therefore, an energy cut is necessary.

Since the upper energy tail of the α contribution overlaps with the low energy tail of the (broad) fission peak, the definition of the energy cut between the two is not trivial. To decide where the energy cut should be done in a systematic way, we use a fit of the curve over the two regions and extrapolate the fitted curves to their point of intersection, which is kept as the energy of the cut, as shown in Figure 24. The underlying thinking is that, at the point of intersection of these two curves, the number of α events spilling over the energy cut and being wrongly counted as fission is roughly equal to the number of fission events below the energy limit and not being counted.

Past analysis have shown that this method leads to consistent neutron count in the fission chamber, with no strong dependence of the neutron count to the cut value.

The *task file* `do_fc_ecut.py` contains a single task that determine the energy where to cut, using a simple `gnuplot` command.

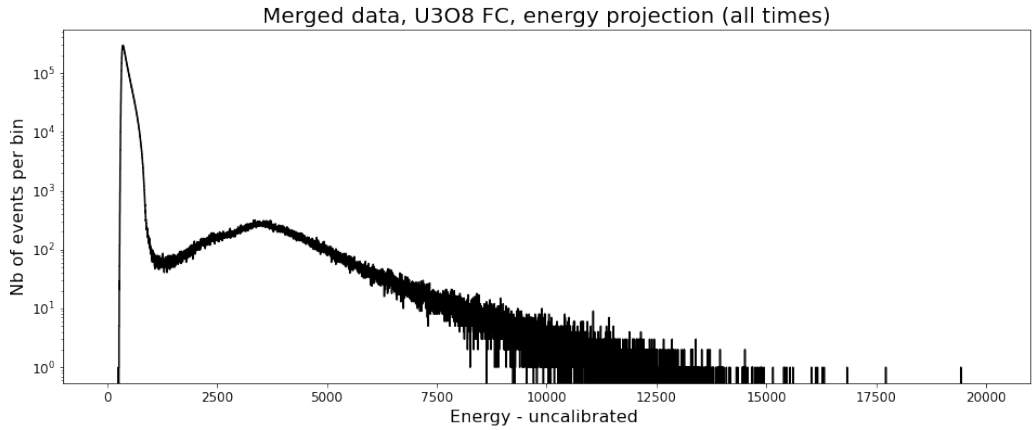


Figure 23: Histogram of the energy projection of the U_3O_8 fission chamber for all time of flight. The α constant contribution is the high statistics peak at low energy (cut sharply by the acquisition threshold). The higher energy *bump* is the distribution of fission fragment energy deposits.

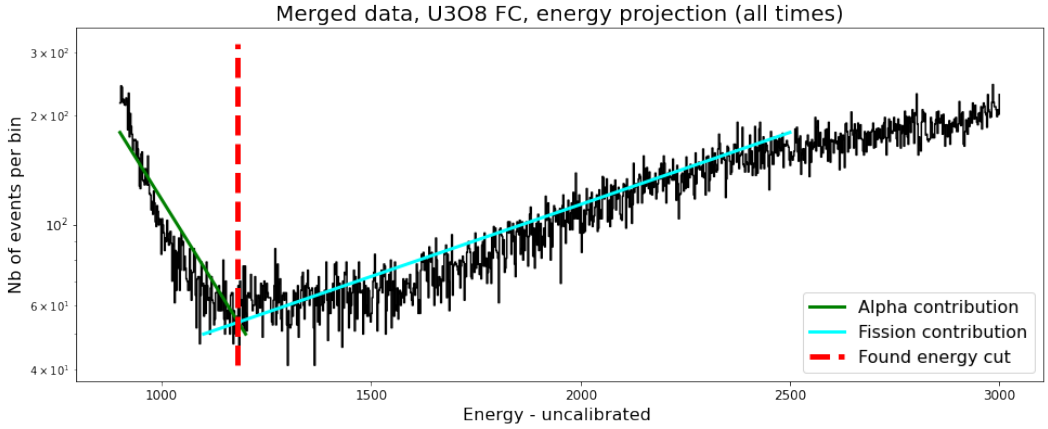


Figure 24: Zoom on the histogram of the energy projection of the U_3O_8 fission chamber (see Figure 23), with the fitted line for α and fission contribution. The point of intersection for the two lines (vertical dashed red line) is taken as the energy cut.

Fission chamber pile up rate

The pile up of events in the fission chambers is overall low, but still has to be considered. Mostly, it comes from the constant α signals that may happen in random coincidence with fission events, leading to wrongly assigned energy or lost event.

The pile up is considered as an average value over the whole time of flight range (previous work showed that there is no time dependency to the pile up in the fission chambers). To compute the pile up rate, we simply divided the number of *clean* events (i.e. not marked as pileup) by the number of all events (without conditions), subtracted by one for convenience (Equation (1)). This provides a coefficient that will be applied to numbers of events extracted on the *clean* spectra to correct for the pile up.

$$\text{pile up rate} = \frac{\sum\limits_{ToF\text{bins}} N_{\text{all}}}{\sum\limits_{ToF\text{bins}} N_{\text{clean}}} - 1 \tag{1}$$

The *task file* `do_fc_pileup.py` contains the pile up rate correction computation.

Flux determination

For the final steps of the flux determination, we first apply the determined energy cut to the 2D matrix and project it on the time of flight axis.

The next step is to remove the background, i.e. events that are within the energy window but might be random signal. To assess the number of count per time bin, we look at the count number in the projected time spectrum after energy cut outside the time of flight region of interest (before the neutron events and the γ flash). The small (below 1 count per 10 ns bin) rate of background is then subtracted to the time projection.

The time spectrum, corrected for the background, is then up scaled by a factor corresponding to the pile up, and fission chamber efficiency.

Finally, the corrected time distribution is converted into a distribution in neutron energy, by applying the formula $E_{\text{n}} = m_{\text{n}}c^2 \left(\left[1 - \left(\frac{DoF/ToF}{c} \right)^2 \right]^{-1/2} - 1 \right)$ (see *Determination of the incident neutron energy by Time of flight*). Once in the *neutron energy* domain, the number of fission events in the FC is converted in number of neutron passing through by dividing, at each neutron energy, by the standard $\sigma_{^{235}U(n,f)}$ cross section from ENDF/B-VIII.0[4] [and5] , and multiplying by the ^{235}U nuclear density in the fission chamber deposits. The values are then corrected for neutron loss in the air between the fission chambers and the target (estimated with MCNP simulations), so that we obtain the number of neutrons hitting the target. Finally, the *flux* spectra is smoothed to avoid big variations from bins to bin (the effect of smoothing is transparent, since later we will integrate over neutron energy range, but it makes the visualization better) and interpolated into a keV wide bins spectrum with the number of neutrons per keV as bin content.

Figure 25 shows the steps to arrive at the number of neutron per kev arriving on target during the recording period of the experiment.

The *task files* `do_fc_cut_and_project.py` performs the cut and projections of the fission chambers 2D histograms, as well as performing the background removal and pile-up correction. The file `do_fc_to_flux.py` does the *time to energy* conversion, as well as the interpolation *per keV* and the final "*physics*" scaling (neutron loss, density of matter, $\sigma_{^{235}U(n,f)}$ division, ...).

[4] ENDF/B-VIII.0 Evaluated Nuclear Data Library https://www.nndc.bnl.gov/endf-b8.0/

[5] ENDF/B-VIII.0: The 8th Major Release of the Nuclear Reaction Data Library with CIELO-project Cross Sections, New Standards and Thermal Scattering Data, D.A. Brown, M.B. Chadwick, R. Capote, A.C. Kahler, A. Trkov, M.W. Herman, A.A. Sonzogni, et al. Nuclear Data Sheets, 148 (2018) 1-142 https://doi.org/10.1016/j.nds.2018.02.001.

Figure 25: Flow chart for the lasts steps of the *flux* determination.

γ yields extraction

The extraction of the γ-rays yields is more complex than for the processing of the fission chamber events. Indeed, we look at specific γ-rays energies, over a selected range of neutron energy (i.e. time of flights). Additionally, the correction for pile up is time dependent and not a constant.

Pile up time profile

The pile up time profile is independent on the γ transition that will be looked at, so it is determined once and for all in a dedicated stage. The Figure 26 shows the different steps in the producing of the pile up profile.

For each Germanium detector, the 2D histogram with all events (including pile-up ones) and only *clean* events (excludes pile-up and overflow), are projected on the time axis. In the time region of interest (starting after the γ flash and going up to the largest possible neutron time of flight [corresponding to the lowest neutron energy]), the ratio of the time projections is done: N_{all}/N_{clean} (in fact, since the ratio is always 1 or larger, the stored value in the file is $\frac{N_{all}}{N_{clean}} - 1$).

The ratio is computed time bin per time bin, which can lead to some significant variations from one bin to the other, so a smoothing of the curve is applied to obtain a profile easier to *inspect* (as for the flux, the time profile will be averaged over ranges of time of flights / energies, so the smoothing is essentially cosmetic and won't affect the data processing). Figure 27 shows an example of the pile up factor calculations on the *Vert* detector.

Finally, the pile up profile is kept both in "time of flight" form, as well as transformed in neutron energy dependent profile.

In Figure 27, we observe a pileup rate of about 60 to 70 %, which is a rather large value. This is due to the relatively high thickness of the target (*about 1.2 millimeters*), which scatters a large portion of the γ rays from the γ *flash*. The high pile up comes indeed after the γ *flash*, and lasts about 4000 nanoseconds (covering the ToF range up down to an equivalent neutron energy of about 250 keV), the shaping time of the trapezoidal filter in the TNT acquisition[Page 36, 2].

The *doit file* `do_ge_pileup.py` is in charge of computing the pile up profiles for the Germanium detectors.

γ transitions of interest

The transitions of interest to extract from the data are listed, with all the relevant information in a dedicated *Yaml* file. Each transition is given a key name that will be used for output directories and files.

The code below (Listing 3) shows the information for one transition looked at in ^{183}W.

Listing 3: Excerpt from the *Yaml* configuration file defining the parameters to extract the yield of the γ ray at 209.8 keV, for the inelastic reaction on ^{183}W.

```yaml
inel209:
   energy: 209.8
   name: 183L06L02
   threshold: 308.
   proj_limits: [207.001, 213.901]
   width: 1.
   parasits: [208.5]
   time_shift:
      bleu: 5.9
      vert: 6.0
      rouge: 10.4
      gris: 18.8
```

(continues on next page)

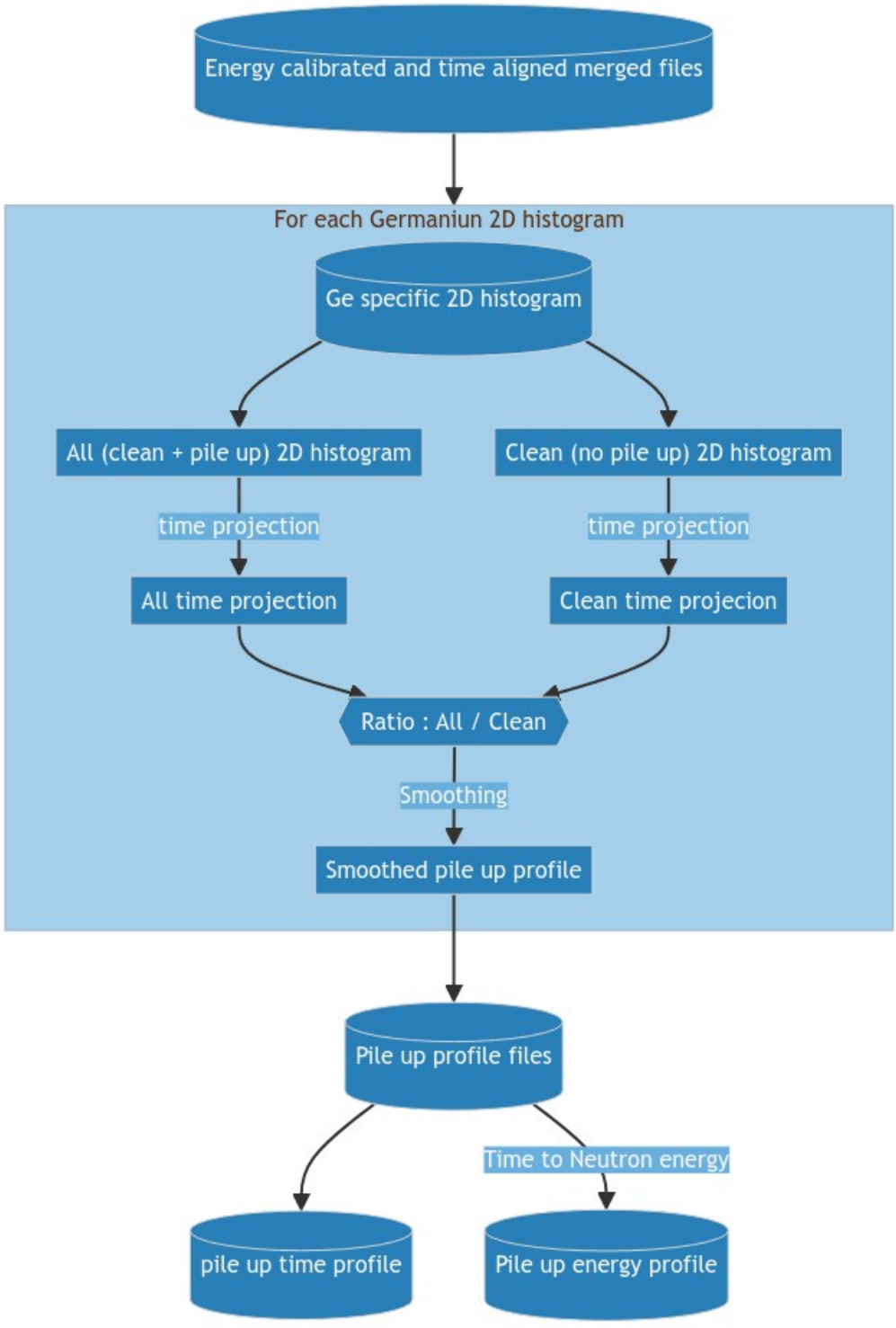


Figure 26: Flowchart of the pile up determination for each germanium detector.

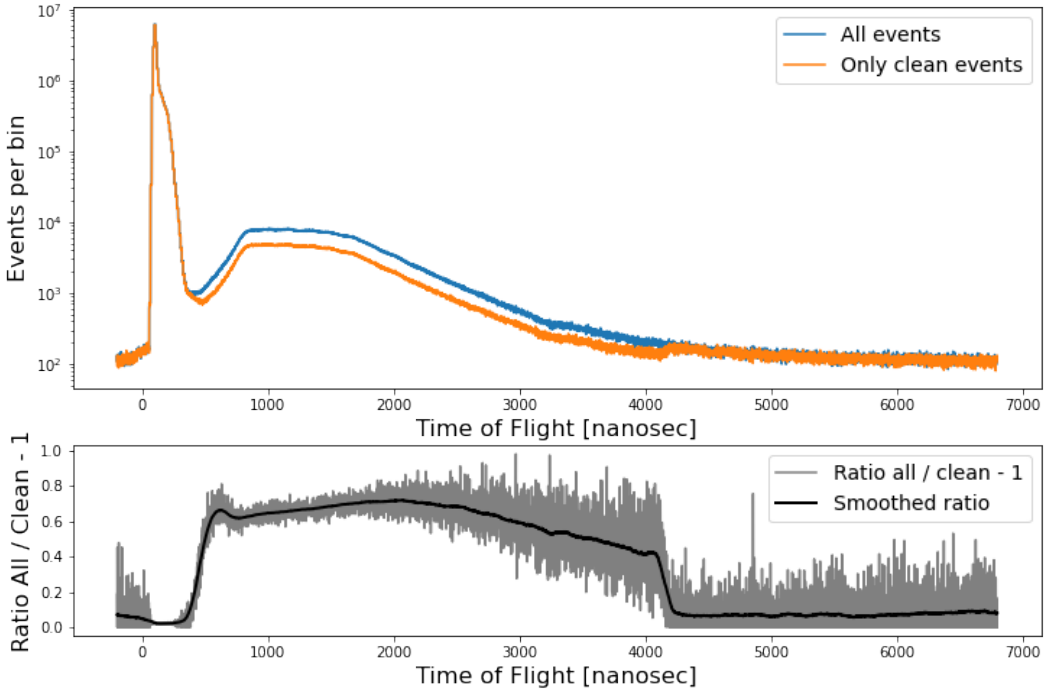


Figure 27: Example of pileup for the *Vert* Ge detector, with the data from the ^{183}W dataset. The `all` (blue) and `clean` (no pilup, no overflow, ...) (orange) time distribution of events (no energy cuts) are represented on the top panel. The ratio $\frac{\text{All}}{\text{Clean}} - 1$ (the -1 is there as a convenience, since there are fewer *clean* events than *all* events considered together) is plotted on the bottom panel in gray, with the smoothed (rolling 60 bins wide window average) distribution in full black line.

(continued from previous page)

```
    En limits: [300., 800., 1000., 1500., 2000., 2500., 3000., 3500., 4000., 4500., 5000.
↪,
                6000., 7000., 8000., 9000., 12000., 16000., 25000.]
    prev_data: ['../old_style/209keV.txt']
```

The key `energy` indicates the energy of the γ ray for the transition (in keV). The `name` key is an arbitrary name, here, chosen to be the corresponding label in the *Talys* output files, in order to facilitate the comparisons. `threshold` is the threshold energy for the transitions (i.e. the excitation energy of the level from which it decays), it is given here for information and not used in the analysis.

The keys `proj_limits`, `width` and `parasits`[6] are relevant for the γ ray peak fitting in the histogram (along with `energy`). `proj_limits` indicates the energy range (in keV) over which to perform the fit. Eventual nearby peaks that have to be taken into account to get a clean fit are listed in `parasits` and the `width` parameter is the initial guess for the peak width (in keV).

`time_shift` indicates, for each detector, the time correction to apply when going from time of flight to energy (see later, *Germanium timing resolution and \gamma energy dependence*). `En limits` lists the limits in incident neutron energy (in keV) over which to project and fit the 2D histogram (each $[E_n,\ E_{n+1}]$ interval will give one data point). The neutron energy windows are chosen (arbitrarily) by the user.

Finally, the list `prev_data` indicates the path to files with the cross section for the same transitions obtained by other codes, persons. Like the *Talys* label indication, it is useful *in the end* for comparison and not used in the analysis.

Germanium detectors efficiencies

The Germanium detectors efficiencies for each transitions of interest are determined by precise *MCNP* calculations, adjusted on source measurement. Indeed, we have to take into account the extended nature of the source (*a disk about 5 cm wide in diameter*) and the absorption of the γ rays in the target, since they are emitted from the *inside*, and the *target is thick* (about 1.2 millimeters) in regard to the usual nuclear physics ones (of the order of micrometers thickness in ion beam experiments).

Ph. Dessagne does this work in the *DNR* group. The output of his calculations is a file that contains the computed efficiencies (with uncertainties given by MCNP) for the detectors, for each energy of interest.

The file is processed to create one file per detector, and adjust the uncertainties, to include the original uncertainty of about 2 % in the activity of the source used to adjust the *MCNP* simulations.

The efficiencies for each transition and for each detector are then drawn from this detector files, with random sampling around the central value according to the uncertainty (for the *Monte Carlo processing*).

Finally, for each transition and for each detector, a file in *yaml* format is written with the efficiency of the detector at the γ-ray energy.

The script `get_transition_efficiency.py` is the main part of this processing.

[6] The writing `parasits` is gramatically incorrect, but the error has been kept from the first versions, to ensure continuous operation. It will be replace in the future by the correct `parasites`.

Germanium timing resolution and γ energy dependence

Due to the charge deposition and collection dynamics in the Germanium crystal, as well as the inner working of the signal amplification chain and processing, γ rays with *low* energies (below $\approx$ 200 keV) will suffer, in the time of flight spectrum from a degraded resolution and a delayed trigger.

Figure 28 shows this deformation of the time distribution, dependent on the γ energy, as observed on the γ *flash* in the recorded data.

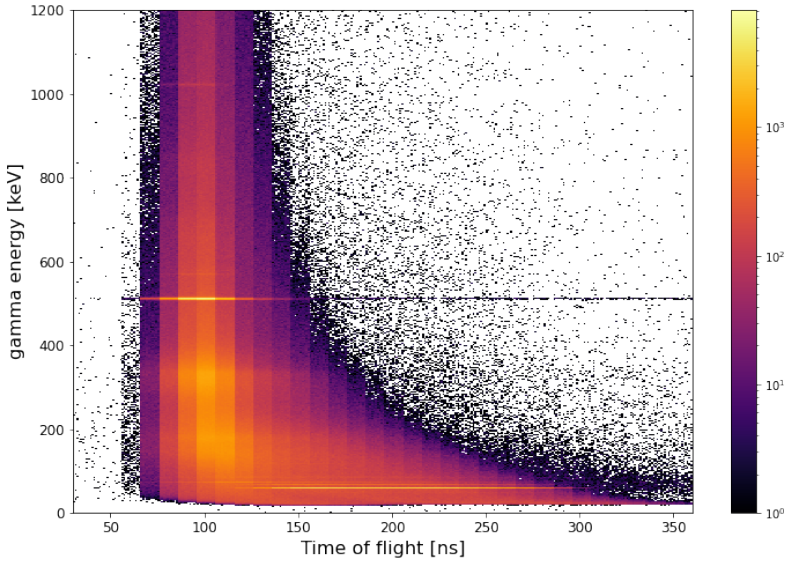


Figure 28: γ energy vs. time of flight 2D histogram of the γ *flash* for the detector *rouge*. The spread of the flash at low γ energies is clearly seen. (The horizontal line correspond to the 511 keV signal from the e^-/e^+ pair creation process)

We can evaluate the loss of resolution as a broadening of the γ *flash* (from below 15 ns wide at higher γ-ray energies to around 40 ns at 100 keV) and a shift to higher delays (i.e., *later*) (about 40 ns later). Figure 29 shows projection of the γ *flash* for different γ energy selections. The mean value and standard deviation of the time distribution are represented, illustrating the time shift and spread.

Although some methods have been proposed to correct for this effect[7], such an attempt at recovering the original time distribution involves adding numerous parameters to the analysis, each with its uncertainty, and lead to higher correlations across the time distribution. Therefore, in this work, the choice was made to perform a correction for the time shift and use large time of flight windows when studying low energy γ transitions, in order to reduce the impact of the time distribution broadening.

The time shift correction, with the shift to be applied listed in the *configuration file* is determined with the help of time projections for γ energy from 20 keV to 1 MeV, with 10 keV intervals (a python script does the job). The mean value for each projection is compared to the one for the 800-1000 keV interval, used as reference, and the shift is computed: $\text{mean}_{E_\gamma} - \text{mean}_{\text{ref}}$. The values are written into a file (one for each detector, as each detector as slightly different behavior). An example (first 10 lines) of the file is given in Listing 4.

[7] Eliot Party. "Etude des réactions (n, xn) pour les noyaux fertiles / fissiles du cycle du combustible innovant au Thorium." Thèse de doctorat. Université de Strasbourg, 2019. ⟨NNT: 2019STRAE020⟩, ⟨tel-02496853⟩.

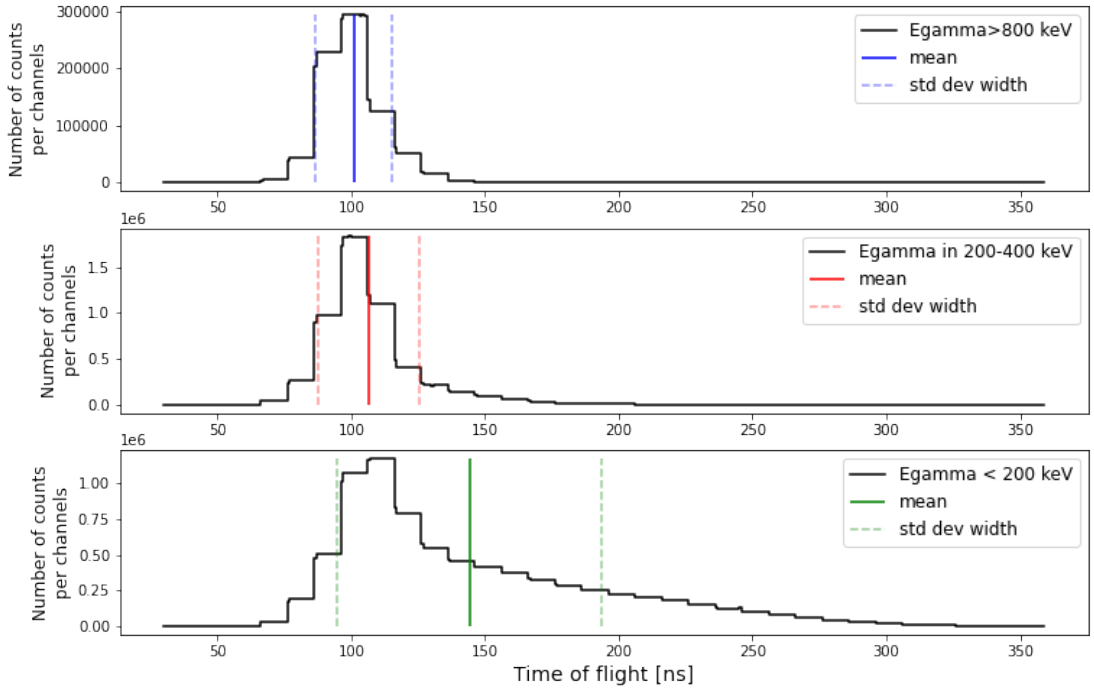


Figure 29: Time of flight projection of the 2D histogram for the γ *flash* in the detector *Rouge*, for three γ energy selections. The highest energies ($E_\gamma > 800$ keV, top) is the reference for unaffected signals. The projection at medium (200 keV $< E_\gamma <$ 400 keV, middle) and low ($E_\gamma < 200$ keV, bottom) show the effect of the loss of resolution. For each γ energy slice, the vertical full line indicates the mean value of the distribution, and the two dashed lines are the ± 1 standard deviation around the mean value.

Listing 4: Example of time shift file for the detector *Rouge*.

```
# Limit at high energy (800-1000keV) : 100.9120271042965 12.534132461296549
# Egamma low, Egamma high, Mean, shift, stddev
20 30 150.45398635057387 49.54195924627737 26.69601551556666
30 40 142.42414186141073 41.51211475711423 31.417816247914903
40 50 139.86647960366275 38.954452499366255 32.38838794135733
50 60 150.3958819370077 49.48385483271119 31.06875061423977
60 70 143.05243684082006 42.140409736523566 31.726058844043436
70 80 134.33240498528772 33.420377880991225 31.19029790962566
80 90 131.753422051708 30.841394947411487 30.772166135207254
90 100 129.7734483037287 28.861421199432215 30.40420622746303
```

From this file, and for each detector, the user extract *by hand* the time shift to be noted in the *configuration file*.

Germanium detectors time of flight and neutron energy windows

For each transition, a list of incident neutron energy windows are given by the user in the *configuration file*. *The Germanium detectors* are all located at the same distance from the neutron production source, but the time resolution, depending on the γ ray energy introduce a specific shift, meaning that a given neutron energy E_n will correspond to a given time of flight ToF_1 in one detector, and another ToF_2 in another Germanium detector.

Therefore, for each transition and each detector, a list of time of flight limits is generated from the incident neutron energy limits in the *configuration file*, taking into account the detector specific *time_shift*, as well as, for Monte Carlo processing, a `time_jitter` ($\pm 10\ ns$).

The result is stored in a *yaml* formatted file, with the original energy limits stored along side. This will allow a simple Time of flight $\Longleftrightarrow$ Neutron Energy correspondence and from this point on, rather than computing one from the other, we will refer to the file as a *look-up table*.

The *doit file* `do_ge_timelimits.py` contains one single task to create the Time of flight $\Longleftrightarrow$ Neutron Energy table.

γ yield extraction

The extraction of the number of γ rays detected in each Germanium detector and for the time of flight windows corresponding to the neutron energy ranges given in the *configuration file* is done via nested loops for each detector, for each γ transition and for each time of flight window.

For a given detector, transition and time of flight window, the *clean* 2D γ energy vs. time-of-flight is projected on the γ energy axis, according to the projections parameters given in the *configuration file*. The resulting 1D histogram is then fitted (using information from the *configuration file*) and a file with the γ peak of interest position, width and integral (as well as associated *intrinsic* uncertainties resulting from the fit procedure) is produced. The fitted function includes a linear background and, if applicable, surrounding peaks (defined in the `parasits` key of the configuration). A dedicated Python script, that automatically adapt to the number of peaks, has been written (adaptive_fit.py). It relies on the `scipy` package to perform the function minimization (see later *Fit improvements*). The *intrinsic uncertainty* returned by the fit includes the *statistical* uncertainty (i.e. $\propto \sqrt{N}$), as well as fit uncertainty (from background subtraction, disentangling with neighboring peaks, ...). Figure 30 shows an example of fit performed by the script, with Listing 5 the *yaml* output produced by the script.

Listing 5: Output file of the fitting script for the fit shown in Figure 30.

```
integral: 8201.854933241437
integral_u: 104.45584034680603
```

(continues on next page)

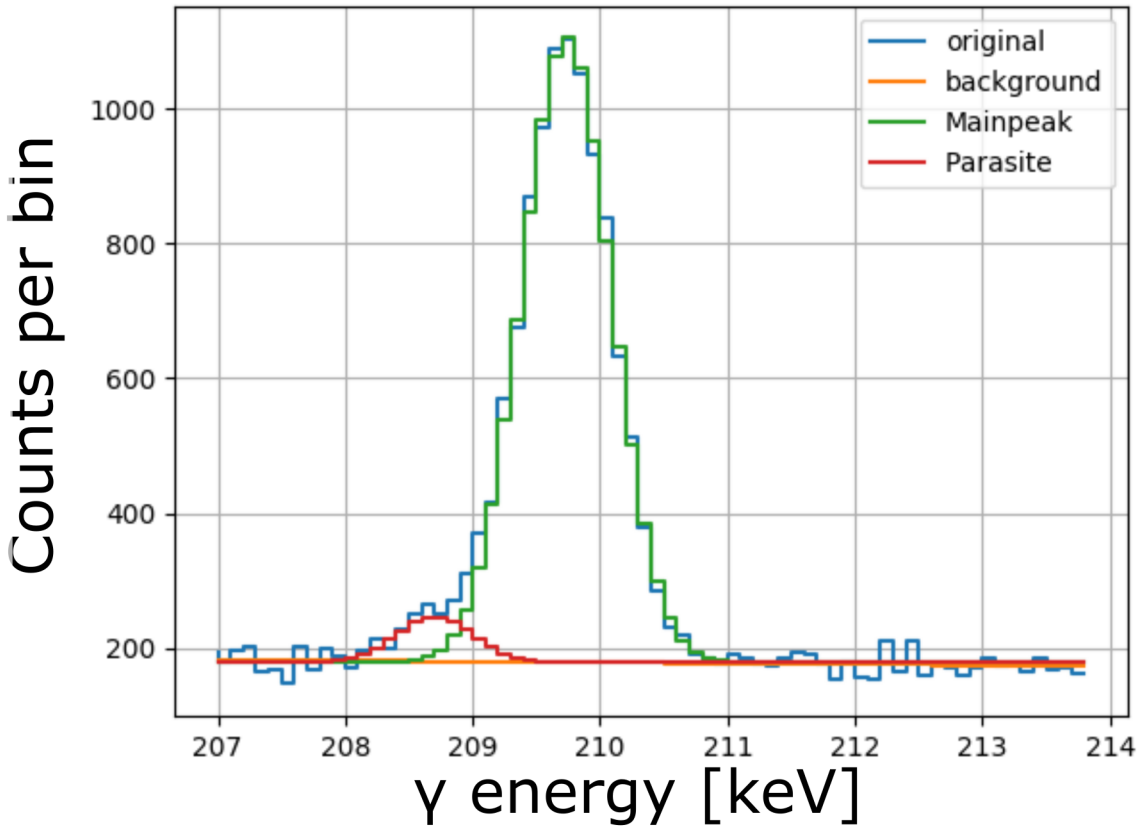


Figure 30: Example of fit on a γ spectra. γ projection for all events in the detector *vert*, in the time-of-flight range 1326 to 1481 nanoseconds (2.5 to 2 MeV) for the γ ray at 209.8 keV in the inelastic reaction on ^{183}W (as shown in Listing 3). Listing 5 displays the corresponding output file.

(continued from previous page)

```
mean: 209.78674323619109
mean_u: 0.004643289187903902
stdev: 0.35333200675810095
stdev_u: 0.004924373154590253
parasites:
   - 208.5
```

In fact, first, a fit on the *whole* time of flight range (i.e. from the highest to lowest neutron energy listed in the *configuration file*) is done. The resulting γ peak position and width are used as starting values for the fits on the time-of-flight windows projections. The integral over the whole time of flight range is used for *consistency check*.

The fit output files are collected into a file that will contain the γ peak integral as a function of the time-of-flight window (similarly, a file with the intrinsic uncertainty as a function of the time of flight window is produced).

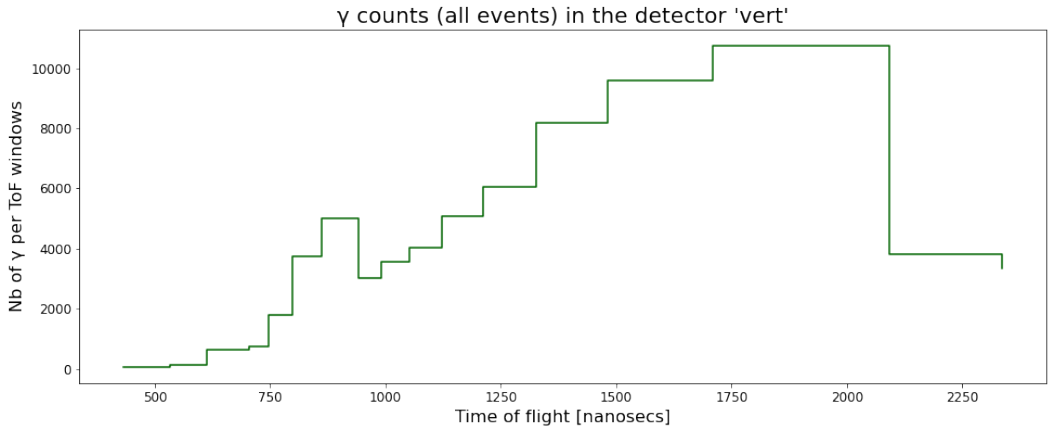


Figure 31: Histogram of the γ ray counts after fitting, as a function of time-of-flight windows, for the detector *vert*. The value shows in Figure 30 and Listing 5 is part of this histogram.

The process of γ rays fitting is described in the flowchart of Figure 32.

The *task file* `do_extract_transitions_gnum.py` organizes the fits, collecting of values, ... (among others tasks).

γ yield consistency check

In order to check the consistency of the number extracted from individual time of flight windows (I_{win}), compared to the whole range (I_{wr}), we sum the integrals from the time-of-flight window fit results, and compare this number to the integral obtained from the fit over the whole range. This is done by computing a kind of χ^2, taking the uncertainties ($u_{I_{\text{wr}}}$ and $u_{I_{\text{win}}}$) into account.

$$\chi^2 = \frac{\left(I_{wr} - \sum_{\text{tof windows}} I_{\text{win}}\right)^2}{u_{I^2_{wr}} + \sum_{\text{tof windows}} u^2_{I_{\text{win}}}}$$

If the comparison indicates that the numbers are compatible ($\chi^2 < 1$), then, nothing is done. If, on the other end, the differences are too large, we compute a scaling factor μ that will be applied to the uncertainties on the individual

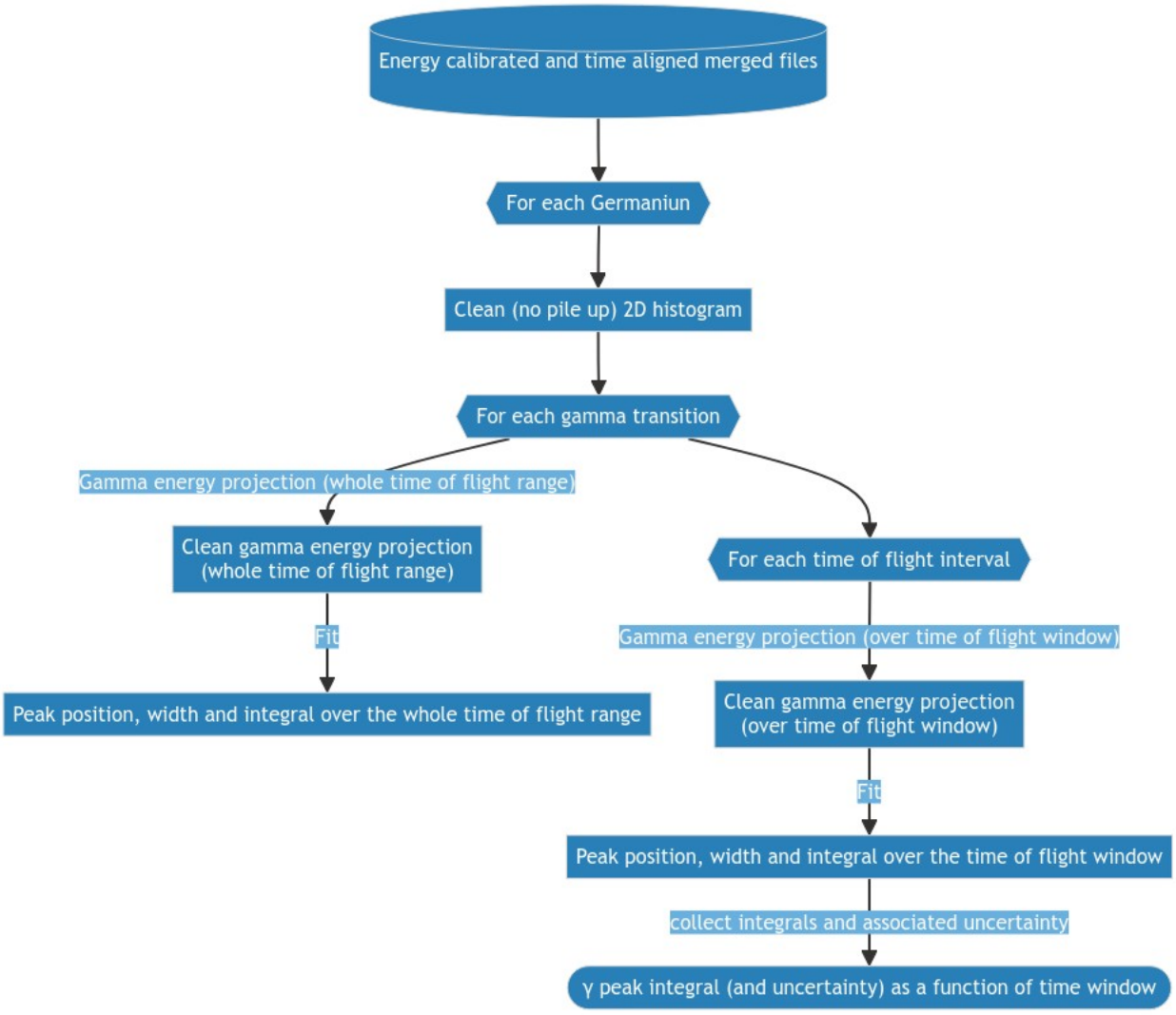


Figure 32: Flowchart of the γ fits.

time-of-flight windows.

$$\mu^2 = \frac{\left(I_{wr} - \sum_{\text{tof windows}} I_{\text{win}}\right)^2 - u_{I^2_{wr}}}{\sum_{\text{tof windows}} u_{I^2_{wr}}}$$

In other words, the uncertainties are scaled up so that the resulting χ^2 is brought down to 1.

The consistency check is performed in `do_extract_transitions_gnum.py`.

γ yield pile up correction

As noted *earlier*, the pile-up of germanium detectors is dependent on the time interval. That is because events from high energy neutrons (short time of flight) are closer to the γ flash and more likely to be counted as pile-up than events with long time of flight (low neutron energy).

For each transition, and for each detector, the pile-up time profile obtained before is averaged over the time-of-flight windows defined in the *configuration file*. The γ count and associated uncertainty extracted from the projection with fits *earlier* are corrected for the pile up according to the averaged value for the time of flight windows. Figure 33 shows the procedure.

The pile up correction is done by `do_extract_transitions_gnum.py`.

γ yield efficiency correction

Following the pile up correction, the γ counts are corrected for the *Germanium detectors efficiency*. The two steps are interchangeable. Figure 34 shows the efficiency correction workflow, that follows the same idea as *pile up correction*, simplified by the fact that the γ detection efficiency is independent on the time of flight.

The efficiency correction is done by `do_extract_transitions_gnum.py`.

Putting it all together: computing the cross sections

The final stage of the analysis is to combine γ counts and neutron flux together to produce the partial, then angle integrate, cross sections.

The number of incoming neutrons is integrated over the neutron energy windows, according to the ranges defined in the *configuration file*. The γ yield obtained earlier is first converted from ToF to neutron energy, using the correspondence table established earlier. It is then divided, energy window per energy window, by the corresponding neutron count. After that, the values are divided by the surface density of the target nuclei in the target material (in atoms per barn). The result is an angular cross section $\frac{d\sigma}{d\Omega}$. The processing to get this cross section is explained in Figure 35.

Finally, the angular cross sections (we use the name "*angular cross section*" to designate the differential $\frac{d\sigma}{d\Omega}$ cross section) are combined pair-wise to obtain the angle integrated one (see *Angle integration of the partial cross sections*). A *yaml* file (`gauss_method.yaml`) lists the pair of Ge detectors to consider and the associated coefficients. This final stage is just a linear combination, so very simple and straight forward. Figure 36 shows the procedure. The intrinsic uncertainties are processed along side the cross sections.

The summation, as well as associated tasks, is done in the `do_make_transitions_xs.py` *task file*. It relies on the *Python* native functions `map` and `lambda`, in order to keep a high flexibility in the number of detectors to combine, and the coefficients used.

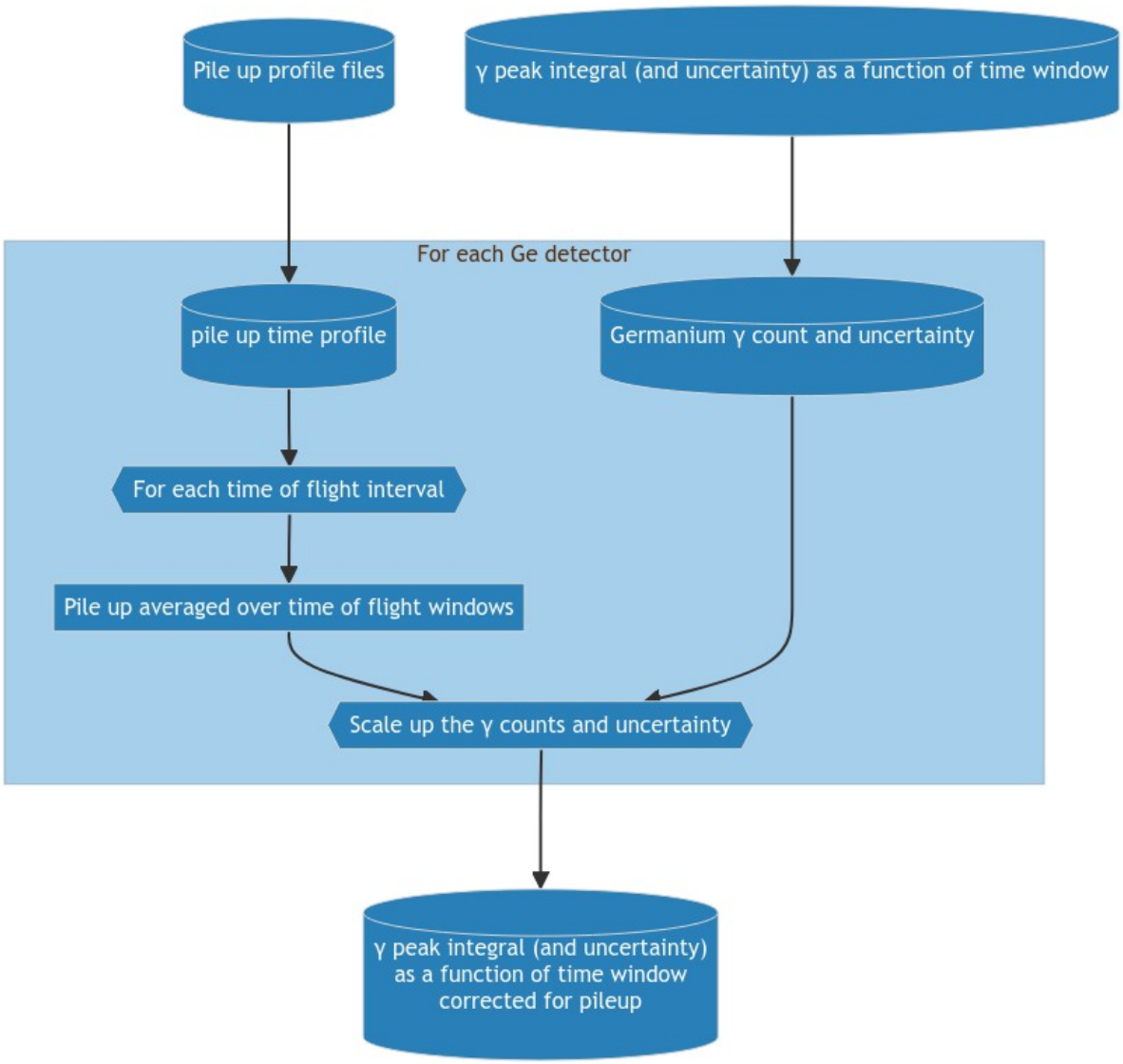


Figure 33: Flowchart of the γ count pile up correction.

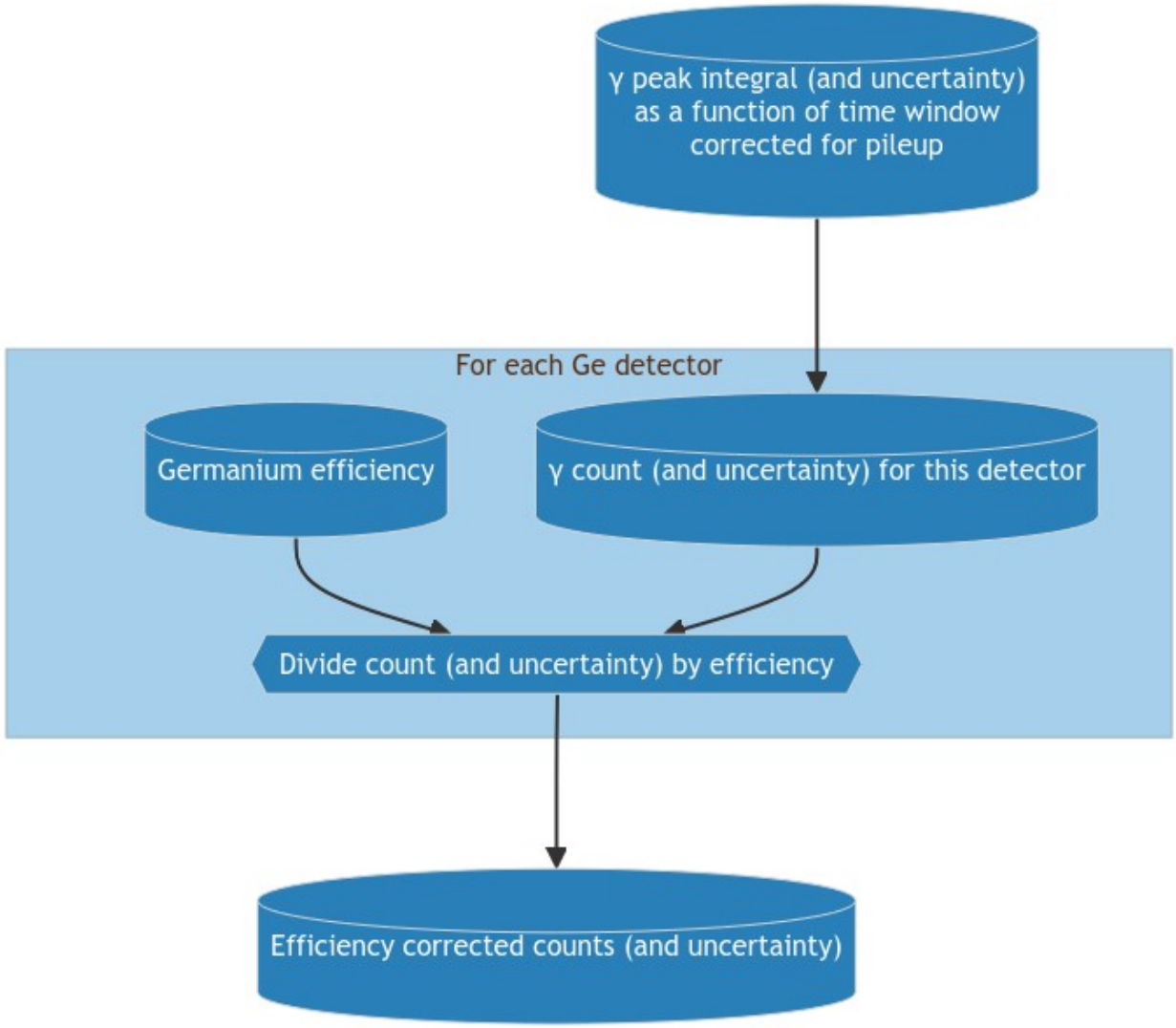


Figure 34: Flowchart of the γ count efficiency correction.

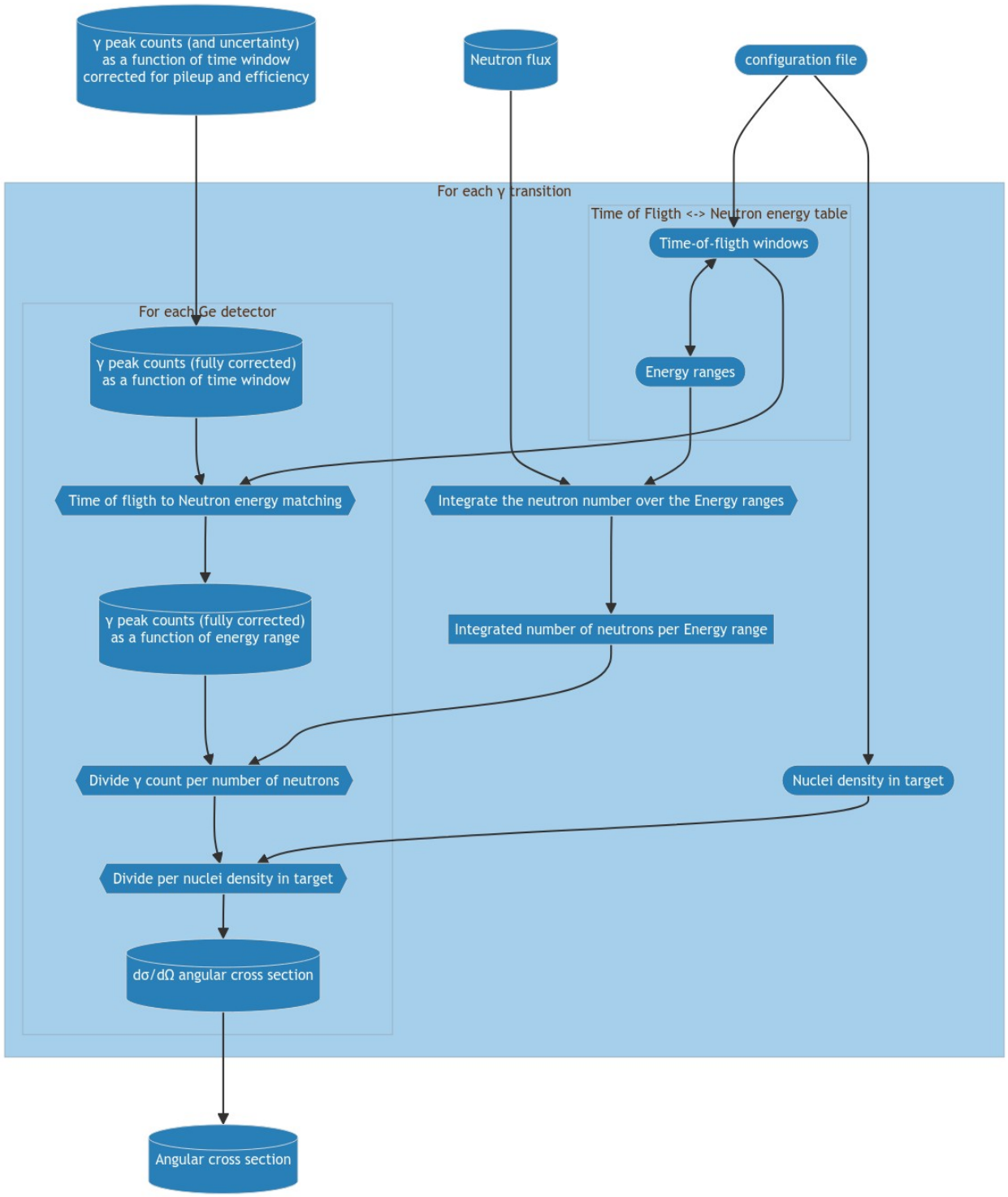


Figure 35: Flowchart of the angular cross section computation.

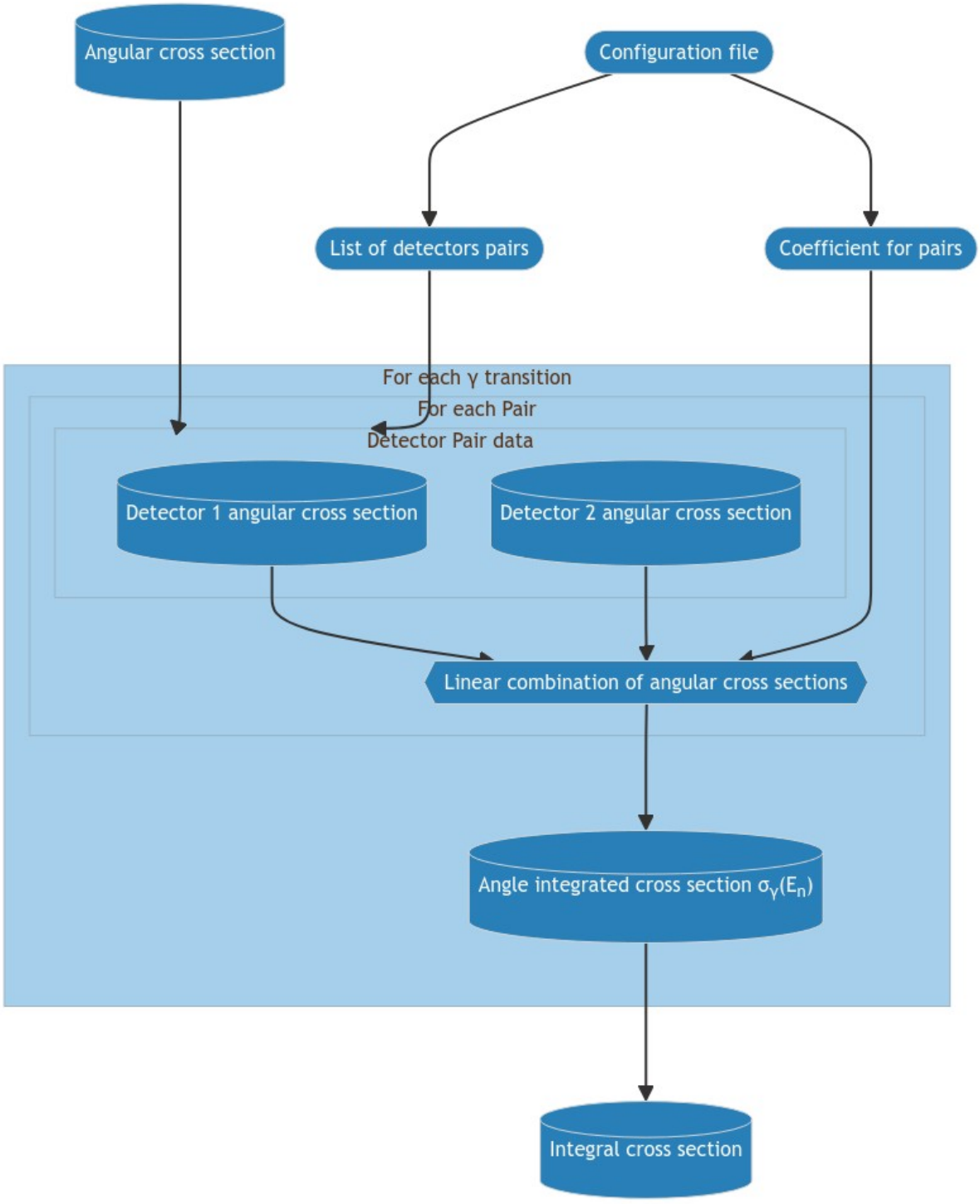


Figure 36: Flowchart of the angle integrated cross section computation.

Integration in a Monte Carlo Loop

The workflow described above is integrated into a Monte Carlo. As described in *Full Monte Carlo data analysis to produce uncertainties and covariances*, some parts are kept out of the MC iterations, as they do not depend on parameters. Figure 37 shows the workflow of the Monte Carlo processing.

Stages out of MC iterations

The raw `.evt` to histogram, as well as time and energy calibration, pile up calculation, general cuts and projections are excluded from the Monte Carlo loops, as they do not call for analysis parameters that are varied in the MC process.

MC iteration setup

The first step in a Monte Carlo iteration is to identify the iteration with a unique *name*. For this, we create a short "uuid"[8] of 6 alpha numerical characters (the `uuid` will be used to store the iteration results in a directory, so it must be a valid directory name). The identifier `000000` is reserved for *central* calculations, i.e. when the analysis parameters are **not** varied randomly and stay at their nominal value.

Once the iteration `uuid` is created and stored in a file, the analysis parameters are randomly drawn according to their distribution law. A *yaml* (`etc/mc_source_files.yaml`) file lists the configuration files where parameters to process are located, and a subdirectory contains the reference files. An automatic Python function goes through the configuration files and generate a varied version, recognizing the parameters to process in the keys: any variables `name` that has a matching `u_{name}` (or `pm_{name}`) is varied following a Normal distribution (or flat distribution) centered on the value given in `name` and of standard deviation `u_{name}` (or $\pm$ `pm_{name}`). In the output file, the nominal value is stored (for *debugging* purposes) as `src_{name}`. The `uuid` and date of the iteration are also recorded in the files.

The varied parameters are :

- The Fission chambers' *distance of Flight* and *efficiencies*.
- The Germanium detectors' *distance of flight* and *time-of-flight to neutron-energy* "*jitter*".
- The *target nuclei density*.
- The *neutron loss between the fission chambers and the target*.
- (The Ge detectors' efficiency is not varied at the *start of iteration* stage, but *later* because the efficiency is computed for each transition and the result is not stored in the *etc* directory. *This could change in the future*.)

A small *clean up* operation is done in the transitions' directories to prevent the accumulation of intermediate files between iterations. It is important, for debugging purpose, that only the files corresponding to the latest calculation are present in the directories.

The `doit` tasks for a new iteration is configured to force running it every time the file is called, so that we are sure a new `uuid` and new variable values are obtained when the new iteration step is processed. By passing the argument `iter=center` when calling the task file, the nominal values of parameters is restored and the `uuid` set to `000000`.

The file `do_start_new_iter.py` is the `doit` input file for setting up a new MC iteration.

[8] Universally unique identifier - Wikipedia

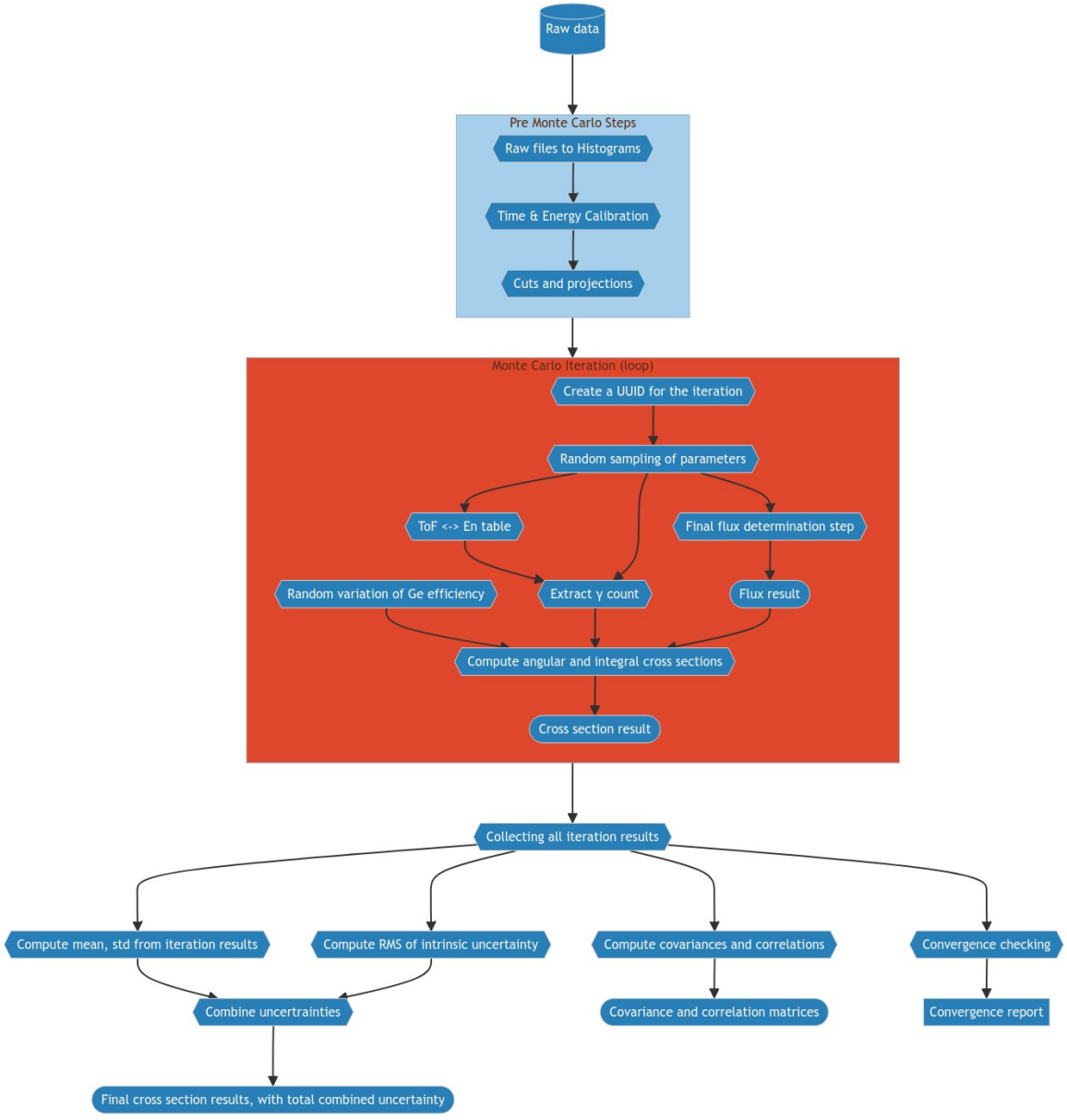


Figure 37: Flowchart of the *Monte Carlo organization* of the analysis code. The Monte Carlo Loop is highlighted in red.

Tasks in MC iterations

The tasks included in the Monte Carlo loop are the one that call the varied parameters.

For the flux, the *last step in flux determination* is included in the loop. The conversion from **time of flight to energy** is done, followed by the interpolation and smoothing to get a spectrum *per keV*. Then the values are divided by the standard $\sigma_{^{235}U(n,f)}$ (from ENDF/B-VIII.0[Page 45, 5 and Page 45, 4]). Finally, the scaling factors for pile-up correction, **efficiency**, **neutron loss** in the air and nuclei density in the FC is applied.

This produces two `flux.txt` files (one for each of the two fission chamber deposits).

For the γ count processing, the MC loop includes :

- Getting the Germanium detectors efficiencies for each transition (*Germanium detectors efficiencies*).
- Computing the time of flight *vs.* neutron energy matching tables for each transition (*Germanium detectors time of flight and neutron energy windows*).
- Extracting all the γ count from the spectra (because of the time *jitter*, the time-of-flight projection range is different at each MC iteration) (*gamma yield extraction*).
- Computing the $(\mathrm{n},\ x\mathrm{n}\ \gamma)$ angular and integral cross sections. (*Putting it all together: computing the cross sections*)

Finally, a dedicated task (`do_collect_iter_results.py`) collects all the outputs (flux, cross sections, as well as parameters collected for convergence checking) into a directory labeled by the iteration `uuid`.

MC final step

Once the *instructed number of iterations have run*, the post-loop stages of the Monte Carlo procedure can start.

Convergence check

The *convergence* is checked, by looking at different quantities.

- First, the sampling of the parameters is checked. Indeed, we need to have enough iterations to reproduce the target distribution of each analysis parameter. To this end, we collect the values of each parameter and extract from it the central value and standard deviation of the samples. It is compared to the target distribution by computing the *overlap* value of the two distributions. The computation is done using the `overlap` method of the `statistics.NormalDist` module in the python standard library. We expect a high overlap value ($\approx$ 90 %). Figure 38 illustrate how the checking of the sampled parameters is done. The convergence of the sampled parameters is a required minimum to consider the rest of the convergence criteria.
- The convergence of the mean values is checked by following a cross validation method inspired from machine learning. The principle is simple: for each point to check, the set of MC iteration values is divided in two subsets, the mean value is computed for each subset and the difference of the mean value between the two subsets is compared to the standard deviation of the whole set. Figure 39 illustrate the method. The underlying logic is that the result of two subsets differ from each other only a fraction ($\approx$ 10 %) of the whole set standard deviation, then we can be confident that any new iteration will not impact the mean value for the whole set in a significant way. In fact, this method will push toward more iterations than what's needed, and ensure we don't under sample. The splitting in subsets is done randomly to avoid any bias, and the validation done several times (as of now, 16). Since we have many points to check (each transition produces several cross section values at different neutron energy), we compute, for each iteration, an average cross validation score (and the related standard deviation). This is a variation on the *step-to-step* convergence criteria, testing for the stability of the *mean* value.
- The convergence of the parametric uncertainty follows the same cross validation method (it is actually done in the same script). The whole set is split in two, then the standard deviation of both subsets compared to the one of

the whole set. If the standard deviations are close together (ratio subset / whole set around 1), then we can infer that any additional iteration will not change the obtained value. Figure 40 illustrate the method. Like the mean, this is done several times on randomly split subsets to compute an average validation score. It is also a variation on the *step-to-step* convergence criteria.

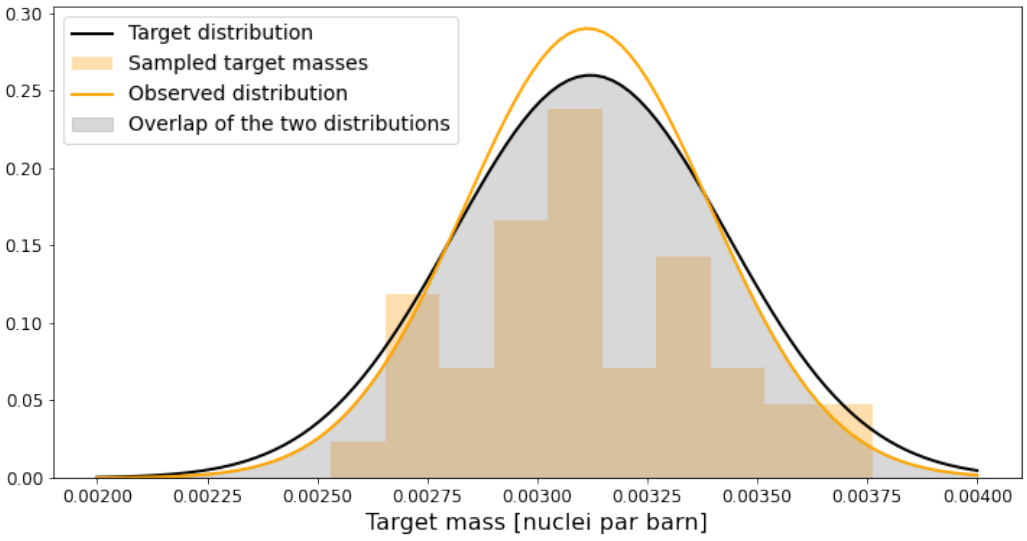


Figure 38: Example of checking the convergence of sampled parameters, here with the ^{183}W mass in the target (in fact, the nuclei surface density, in nuclei per barn). The *target* distribution (normalized to 1) is represented in full black line. The sampled values are the sandy histogram (with a normalization to 1 too), and the orange full line is the normalized Gaussian distribution associated with these samples (using the mean and standard deviation). The gray area is the overlap between the two distribution (here, the overlap value is 94.6 %)

It is left to the appreciation of the user to look at the convergence scores and decide if more iterations are needed. The way the analysis code is designed, it is possible to run a new batch of iterations without losing the results from the previous ones, so going from, for example, 30 to 35 iterations only "costs" 5.

The convergence check is done in the `doit` file `do_MC_convergence_report.py`.

Computing the central value, standard deviation and covariance

The final step of computing the central value, standard deviation, covariance and correlation matrix, is very simple as it requires just to collect the results of the Monte Carlo iteration and process them using `numpy` to produce the final results.

The mean value and associated standard deviation are computed using a dedicated script (`iters2mean_n_std.py`) that is mostly busy with managing options (output file name, log scale in plots) and plotting the result (using `pyplot`). The core of the code is just 6 lines[9]: Listing 6.

[9] Which can probably be even more condensed and optimized.

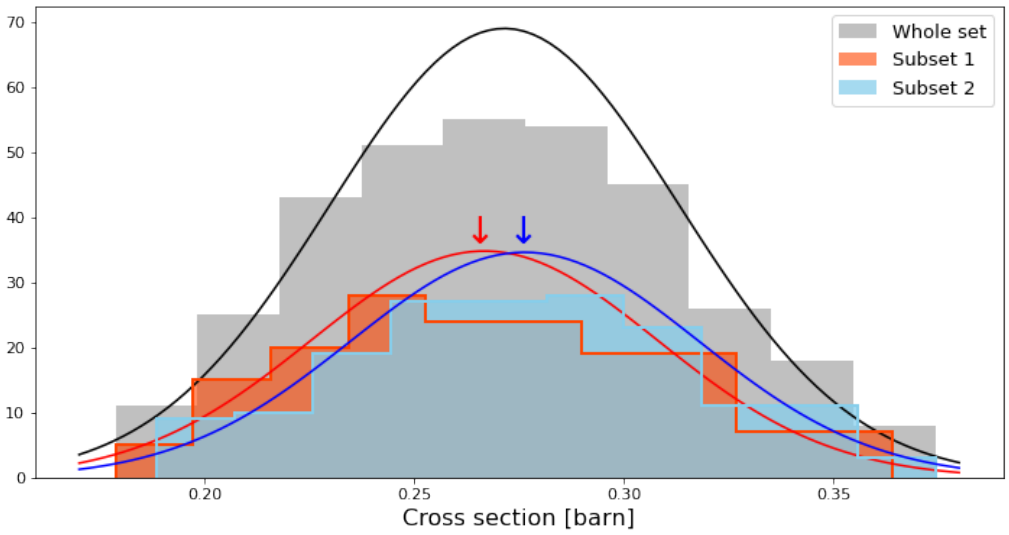


Figure 39: Example of checking the convergence of mean values, here with the cross section of the $(\mathrm{n},\ 2\mathrm{n}\,\gamma)$ channel, for the 351 keV γ ray, in the 9 to 10 MeV energy window. The *whole set* is represented in with a gray histogram of the 336 realizations, the corresponding fitted Gaussian distribution is in full black line. Two randomized half subsets are represented the same way (in red/orange and blue). The downward arrows represent the position of the mean values of each subset. The difference in x-position between the two subset mean values, is clearly much less than the standard deviation of the whole set. (For this transition, we compute in average that the difference between the means of subsets is around 7 % of the standard deviation of the whole set.)

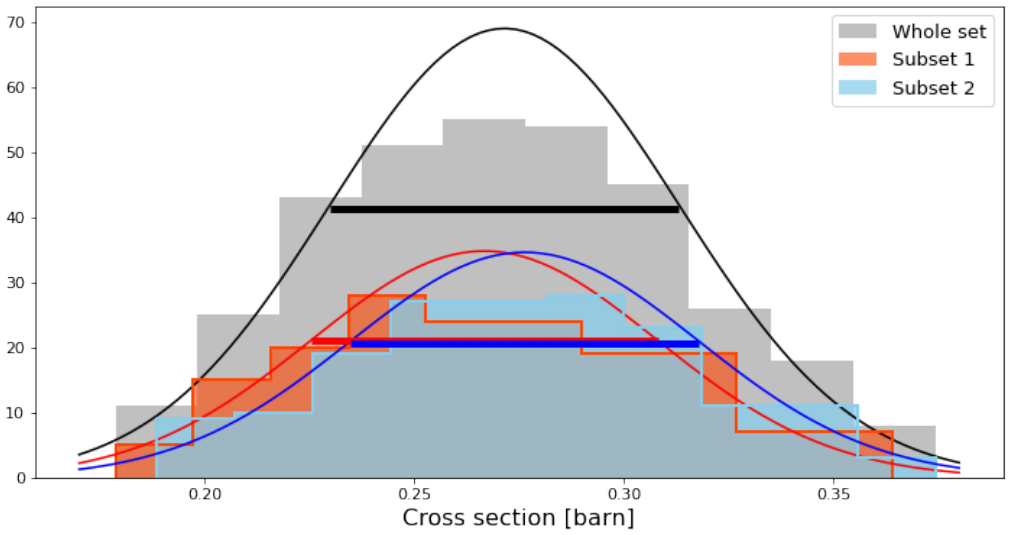


Figure 40: Example of checking the convergence of standard deviation values, here with the cross section of the $(\mathrm{n},\ 2\mathrm{n}\,\gamma)$ channel, for the 351 keV γ ray, in the 9 to 10 MeV energy window. The *whole set* is represented in with a gray histogram of the 336 realizations, the corresponding fitted Gaussian distribution is in full black line. The horizontal full black line, at 60 % of the maximum of the distribution, spans the standard deviation of the Gaussian. Two randomized half subsets are represented the same way (in red/orange and blue). The two *one standard deviation* lines for the subsets are just slightly shorter than the one for the whole set, indicating a good convergence. (For this transitions, we compute an average ratio between the subset's standard deviation and the whole set one of 0.997 (0.028).)

Listing 6: Key lines of Python code doing the mean and standard value calculations.

```
for this_file in iter_results_files:
    x, dx, y = np.loadtxt(this_file, unpack=True)
    data_cols.append(y)
 the_data = np.column_stack(data_cols)
 the_mean = the_data.mean(axis=1)
 the_std = the_data.std(axis=1)
```

The computation of the associated covariance and correlation matrices is basically the same, using the `cov` and `corrcoef` functions of `numpy`. Because the flux files contain a very large amount of points (about 27000), the data needs to be reduced before the covariance and correlations are computed (otherwise the amount of memory need for the processing is way too high). This is done by keeping only one out of 100 points (reduction of the neutron energy steps from 1 per keV to 1 per 100 keV).

The intrinsic uncertainty is processed separately, with the *root mean square* of the series of Monte Carlo iteration values computed as the combined one (`iters_err_rms.py`). The intrinsic and parametric uncertainties are combined by doing a square sum ($\sigma^2_{\text{combined}} = \sigma^2_{\text{intrinsic}} + \sigma^2_{\text{parametric}}$).

The file `do_MC_calcmean.py` is the `doit` task file that perform the final operations.

CHAPTER

FIVE

RESULTS AND DISCUSSIONS

Warning: A warning: Preliminary results

Only preliminary results can be shown in this manuscript. Indeed, with a focus on the method and the code, in the first round of analysis and writing this manuscritp, tricky cases were put aside for *later*, including input event files requiring increased inspection to correctly calibrate, transitions with very low statistics or high contamination from other γ rays, ...

Actually, the preliminary nature of the results, is a good opportunity to discuss the merits and disadvantages of the method and possible *improvements*.

With more work and fine tuning, some other γ rays can be exploited (at least one inelastic and two $(\mathrm{n},\ 2\mathrm{n})$ transitions). Present results can also be improved (as you will see *later*).

However, overall, if the current results had to be published as final today, I would not be embarrassed by then and stand by their value (even if I know they can be slightly better).

5.1 Results

I will present cross section results for 6 $(\mathrm{n},\ \mathrm{n}')$ and $(\mathrm{n},\ 2\mathrm{n})$ transitions, extracted using the code *described previously*. The transitions can be seen in blue in the level scheme in Figure 4.

The experimental values are compared to *Talys* calculations . (The *level scheme used by Talys is shown in the appendix*) We compare to *default* computations using version 1.96 (2021), P. Romain's own calculations[1], and M. Dupuis's calculations[2].

The values will also be compared to very preliminary results obtained from the data recorded with *Grapheme* and a natural tungsten target, analyzed by Pol Scholtes in 2011[3]. He extracted two $(\mathrm{n},\ \mathrm{n}'\ \gamma)$ cross sections related to reaction on ^{183}W.

At the time, the natural tungsten target mass was underestimated. Thanks to comparisons with the data from isotopically enriched targets[4], the natural target was re-characterized at *JRC-Geel* and the mass was reevaluated to be 6.35 % more than first considered, leading to a corresponding scaling down of the cross sections extracted from the natural tungsten dataset. This scaling down as been applied to the P. Scholtes' values shown here.

Note: The data from P. Scholtes was extracted from his report's figures using WebPlotDigitizer online, because the values (whether they were stored initially in `root` files or excel spreadsheet) could not be found again in the archives.

[1] Done by Pascal Romain (CEA/DAM/DIF) in 2016, private communication.
[2] Done by Marc Dupuis (CEA/DAM/DIF) in 2024, private communication.
[3] "Mesure des sections efficaces des réactions $(\mathrm{n},\ x\mathrm{n}\ \gamma)$ sur les isotopes ^{186}W, ^{186}W et ^{186}W" Pol Scholtes. Internship report (2011).
[4] "Improving the accuracy of 182,184,186W(n, n'γ) cross sections calculations" G. Henning, M. Dupuis, M. Kerveno, R. Capote, Ph. Dessagne, S. Hilaire, T. Kawano, P. Romain, C. de Saint Jean, P. Scholtes, P. Tamagno. In preparation for Physical Review C. (2024)

This is a good example of bad practice in file tracking and data management that has to be avoided[5].

The uncertainties associated with the cross section values extracted from the figures use a *flat* relative uncertainty value of 20 %, extracted from the graph for a few points and generalized to all. There is no uncertainty on the neutron energy part.

Finally, the values are also compared for 4 transitions (two $(\mathrm{n},\ 2\mathrm{n}\,\gamma)$ and two $(\mathrm{n},\ \mathrm{n}'\,\gamma)$) to values obtained by using the *old full Monte Carlo Code* on the same data.

The raw values of the experimental points are given in the *appendix*.

5.1.1 Convergence

The results presented come from 42 iterations, yielding 84 flux (since there are *two fission chambers*, that we chose to combined together) and 336 $(\mathrm{n},\ x\mathrm{n}\,\gamma)$ files for each studied transitions (*four detector pair combinations, two fission chambers*, all combined together to give one *mean* value). The *convergence of the result is tested* first.

The convergence of **sampled parameters** varies, with overlaps of target distributions and observed ones going between 0.85 and 0.988. We note that the lowest values are for distribution that are either very narrow (like the distance of flights, which is very well known), for which even a tiny shift in central value will lead to a significant drop in distribution overlap, or very broad (like the U_3O_8 fission chamber efficiency, with a 5 % relative uncertainty) for which one sample *too far* can lead to a difference in distribution width.

Right away, we can say that an increase in the number of iterations might lead to increased convergence of the sampled parameters, however, that does not mean the overall result will be affected.

The cross validation convergence for **the flux** shows that the mean value is well defined, with difference between subsets of iterations at $16\ (9)\ \%$ of the computed parametric uncertainty. The uncertainty from subsets of iterations are within $99\ (6)\ \%$ of the computed standard deviation for the whole set. Figure 41 shows the cross validation plots of the standard deviation.

It is clear, from observing the *cross validation graphs*, that there are two main contributors to the flux central values. Indeed, some symmetrical structures around the 1 line are observed. We attribute this to differences between the U_3O_8 and the UF_4 fission chambers. In Figure 42 the ratio of the flux from each fission chamber to the *final full MC analysis flux* clearly shows a difference, with an average $-5.6\ \%$ for the UF_4 and $+5.8\ \%$ for the U_3O_8. The curves even invert around 20 MeV and above 25 MeV, which will be reflected in the *correlation matrix*. The sources of these differences are not obvious. It could be linked to poor homogeneity of the deposit (as mentioned in[1]), or because of more parasitic reactions (like $(\mathrm{n},\ \alpha)$) on Oxygen compared to Fluorine[2].

In previous analysis, only the UF_4, with its higher detection efficiency, was considered. Here, we chose to combine the two, as a test of the method and a way to maximize the use of the recorded data. It might be relevant in the finalized analysis to discard the U_3O_8 fission chamber data, as a way to decrease the parametric uncertainty.

Note: It brings forward a fundamental question: "Can we ignore part of the data because it makes our result *'worse'* (whatever we mean by that)?" If the two fission chambers give us slightly different flux profile, this should be reflected in the uncertainty of our computed flux.

Finally, for the $(\mathrm{n},\ x\mathrm{n}\,\gamma)$ **cross sections**, the mean value cross validation yields a ratio of the difference between subsets to the total parametric uncertainty of 0.08 to 0.11 with standard deviation 0.02 to 0.05. The parametric uncertainty cross validation is steady at 0.995 with standard deviation from 0.02 to 0.04. This shows a good convergence of the

[5] I'm not blaming anyone, it was not possible at the time to anticipate that it would be so long before someone looked into the data again. Also, the technical solution to centralized data did not exists at the time.

[1] Jean-Claude Thiry. Measurement of $(\mathrm{n},\ x\mathrm{n}\,\gamma)$ reaction cross sections of interest for the Generation IV reactors. Université de Strasbourg, 2010. <http://www.theses.fr/2010STRA6144>

[2] Perhaps I should look into that.

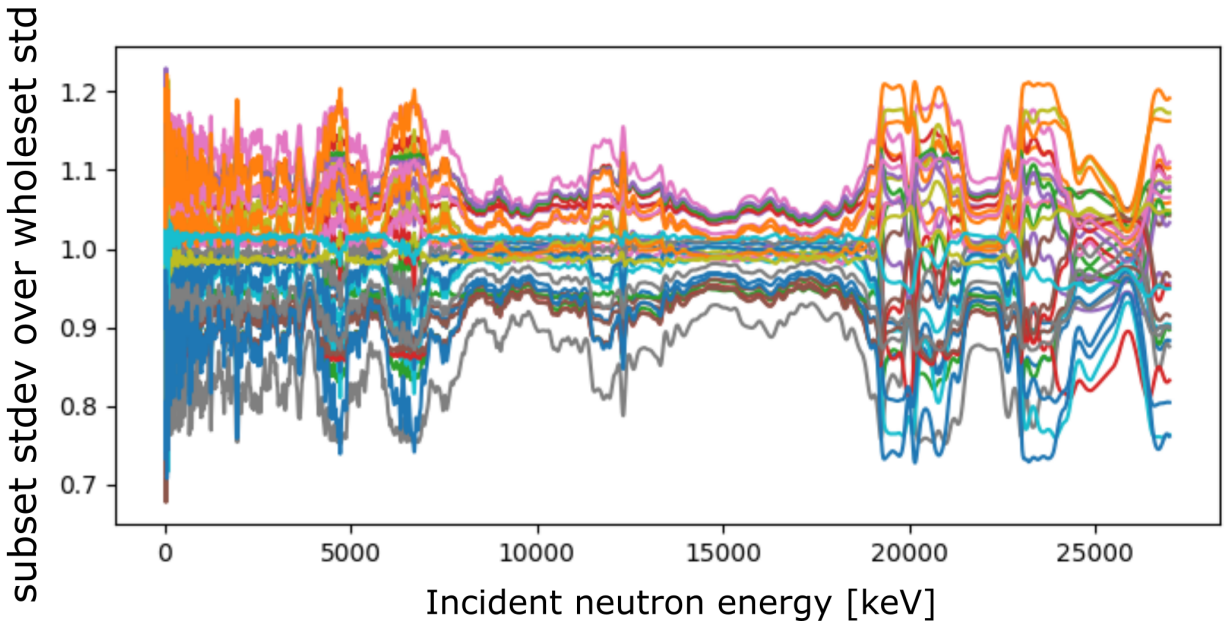


Figure 41: Cross validation of the standard deviation, i.e. ratio of the subsets' standard deviation over the whole set's one.

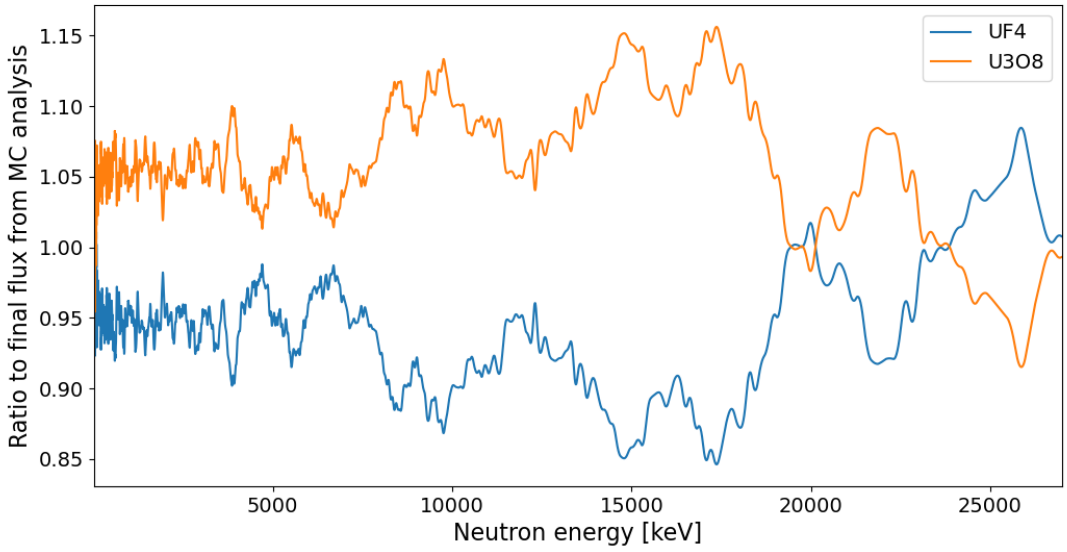


Figure 42: Ratio of the fluxes calculated from the UF_4 and the U_3O_8 with the *central values parameters* (`uuid == 000000`) to the *final full MC analysis flux*. The differences between the two fluxes appear clearly.

cross sections, even when using the two fission chambers (the effect of using only one or the other *fission chamber* might be small compared to other parameters like *Ge detectors efficiencies* or the sample mass).

As an example, figures Figure 43 and Figure 44 show the cross validations plots for the 259 keV transition. The *report* from cross validation is also given in Listing 7.

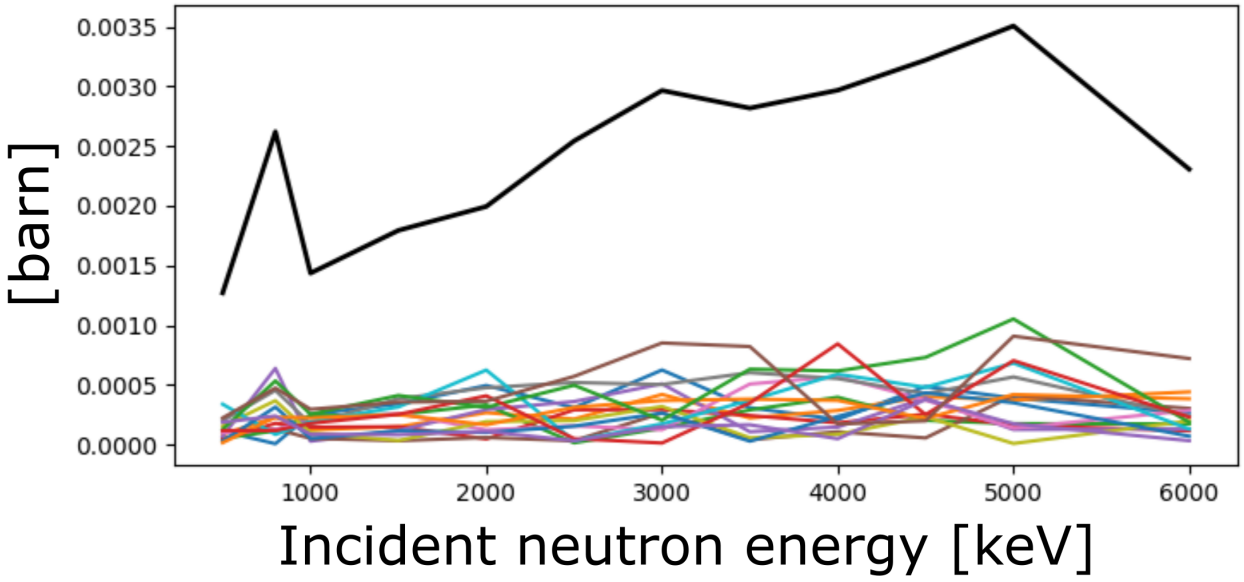


Figure 43: Cross validation of the cross-section mean value for the 259 keV γ transition. The differences between two mean values computed on subsets of the data are plotted (in color) against the standard deviation from the whole set (black line).

Listing 7: `inel259_cv.report.yaml`: Cross validation report for the 259 keV γ transition.

```
number of source files: 336
ratio of CV mean diffs to std:
   mean: 0.11030893039251516
   std: 0.04399769928959092
ratio of CV stds to ref std:
   mean: 0.995587275073077
   std: 0.021496687426985076
```

Overall, this indicates a reasonable level of convergence for the result we will discuss now.

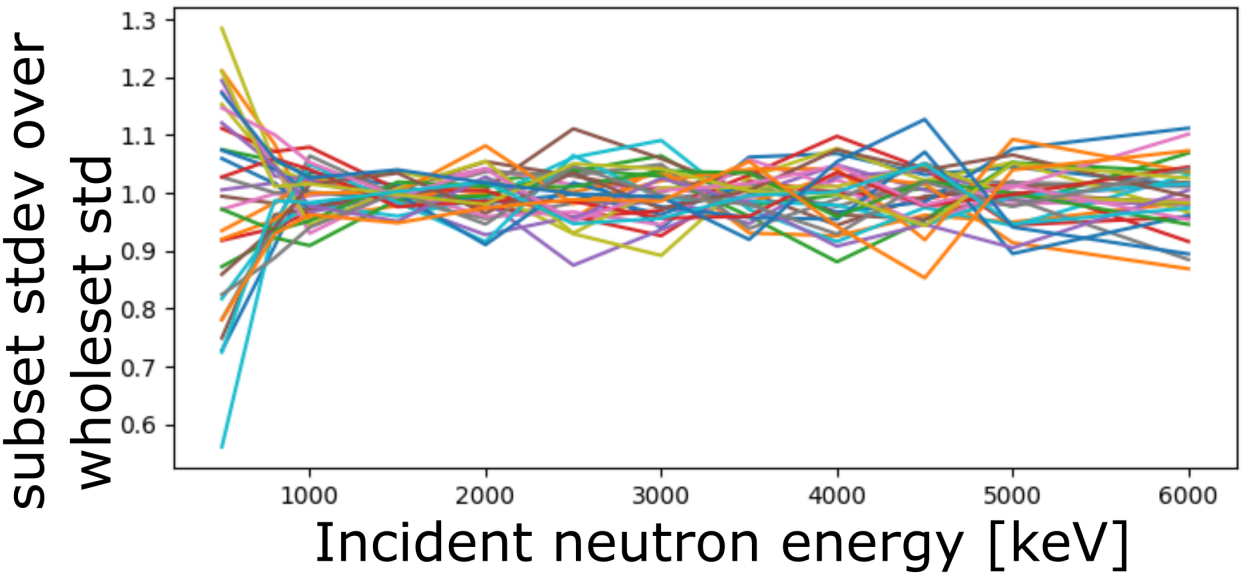


Figure 44: Cross validation of the standard deviation for the 259 keV γ transition. Ratio of the subsets' standard deviation over the whole set's one.

5.1.2 Flux

The *flux* (We use the term *flux loosely*. We actually compute the number of neutrons going through the target) is not, a priori, an output of our analysis. It is used in the computation of the cross sections, and a new flux (actually, two, since there are two fission chambers) is recalculated in each Monte-Carlo iteration, in order to account for FC related analysis parameters (efficiencies, mass of Uranium in the deposit, ...). However, for consistency checking, as well as possible usage in the cross section evaluation, the flux is computed at the end of the Monte Carlo iterations.

Figure 45 shows the neutron "flux". The flux is compared to neutron count from the same data set using the previous analysis code. As the old analysis code does not keep intermediate results, we are only able to extract the neutron number integrated over the neutron energy windows for *one* Monte Carlo iteration and *one* γ transition. It is also compared to one from a previous experiment using a ^{184}W target, scaled down to the effective beam time of the data set.

Due to the high efficiency of the fission chambers ($\geq$ 75 %), the intrinsic (i.e. *statistical*) uncertainty have not been considered, as they would be negligible compared to the parametric ones.

The correlation matrix (see Figure 46) for the flux is mostly positive, which is not surprising, as the parameters in play are mostly scaling factors. The negative correlations regions are at neutron energy ranges where there is a significant difference between the U_3O_8 and the UF_4 fission chambers (see Figure 41 and Figure 42). Using only one fission chamber in the analysis (see *later*) should resolve this and produce a *flatter* correlation matrix (as well as lower uncertainties).

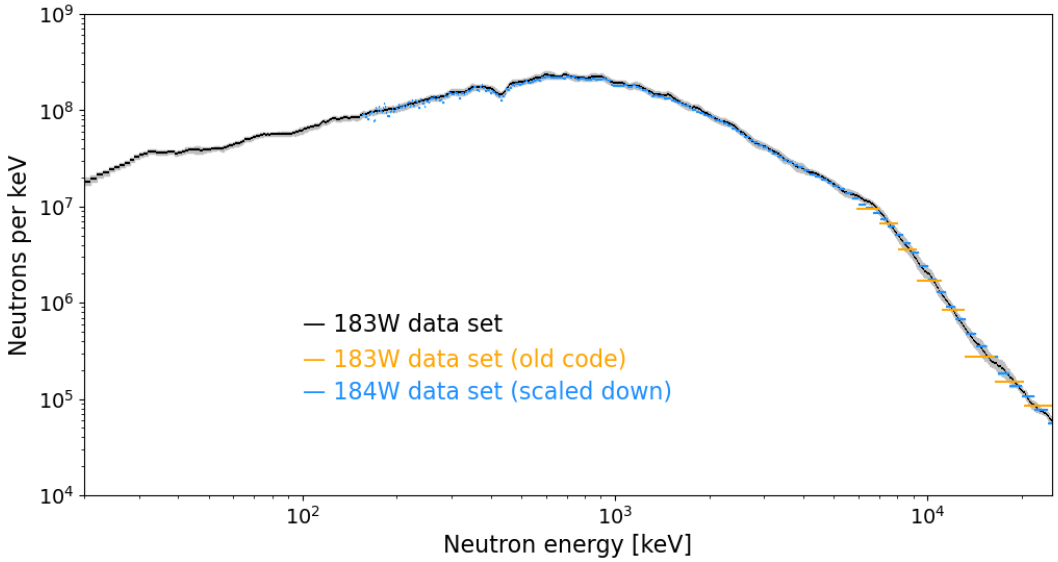


Figure 45: Number of neutrons per keV going through the Grapheme target during the data recording for the ^{183}W target. The central (mean) values are represented with the black line, the gray area delineates the $\pm$ 1 standard deviation range. The orange horizontal lines are the values computed by the previous analysis code for the same dataset (see *text* for details). The blue line is the value from the ^{184}W data set, scaled to the ^{183}W data set recording time.

5.1.3 Experimental (n, n' γ) and (n, 2n γ) cross sections

Check of angular cross section

As a diagnostic, we also take a look at the angular cross sections. Because of the *symmetry of the setup*, angular cross sections for the detectors *Bleu* and *Vert* (both at 150° with respect to the beam axis) should be compatible. The same apply for the detectors *Gris* and *Rouge* (at 110° with respect to the beam axis).

Figure 47 shows the angular cross sections for the *313.0 keV transition* in the $(\mathrm{n},\ \mathrm{n}')$ reaction on ^{183}W.

In the Figure 47, we see that the values for the detectors *Bleu* and *Vert* at 150° match very well together. The agreement between the values from detectors *Gris* and *Rouge* is not as good, but as it is the result from a Monte Carlo iteration, it is possible that the efficiencies sampled for this set of value creates the amplitude spread between the two detectors. Additionally, if the parametric uncertainties were to be taken into account to draw the error bars, the two data sets would probably be declared *compatible*.

As the angular cross sections are a product of the data analysis, it is absolutely possible to produce them as the *final* results (along with their covariance and correlation matrices, ...) rather than the angle integrated one as we will do in the rest of the manuscript.

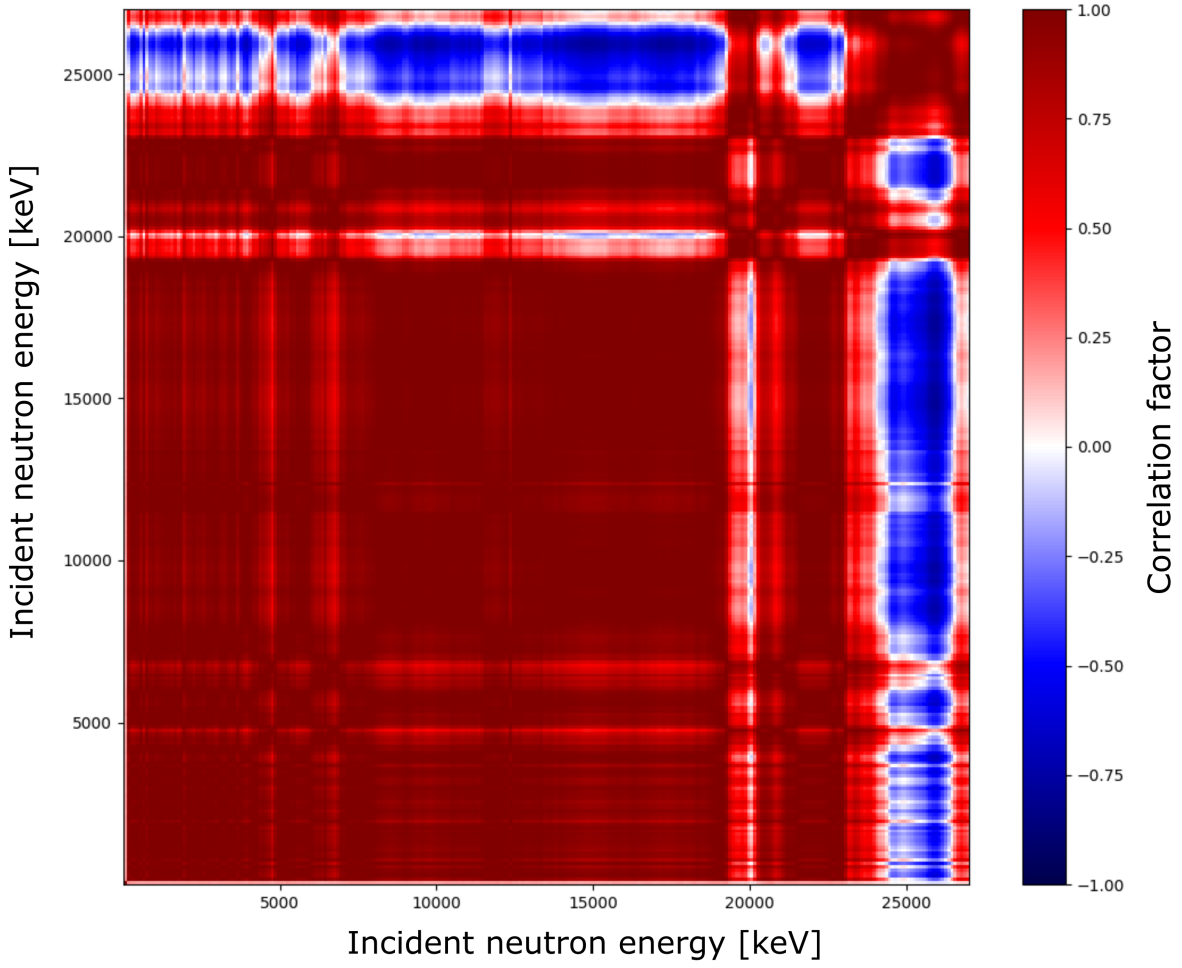


Figure 46: Correlation matrix for the neutron flux. The negative correlation areas are at the same energies as the discrepancies between fission chambers (see *in text*).

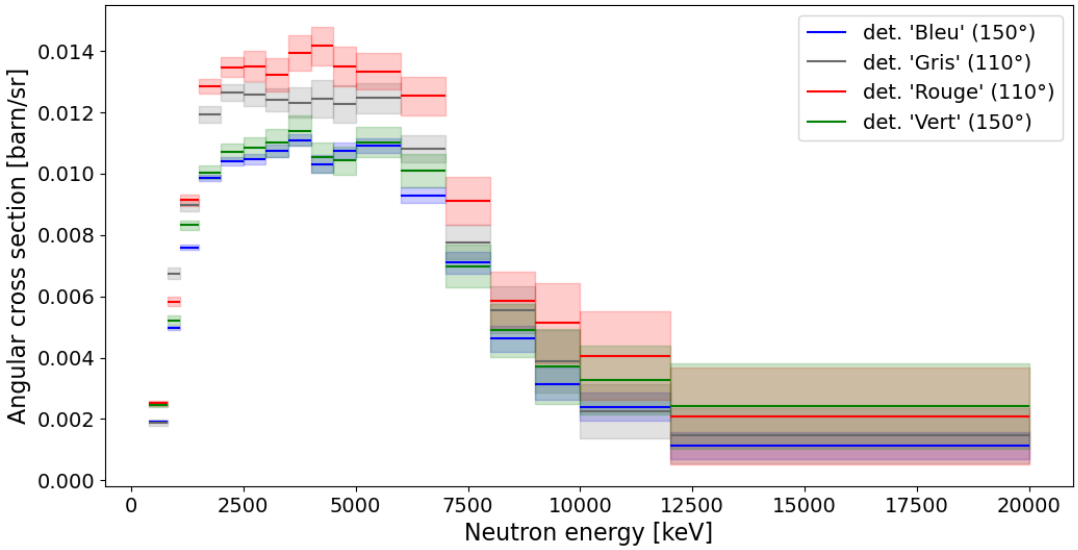


Figure 47: Angular cross section for the four detectors, for the 313.0 keV γ ray in the $(\mathrm{n},\ \mathrm{n}')$ reaction on ^{183}W. This is the result for one MC iteration. The cross section value is the full line, with the transparent color filling around the **intrinsic** uncertainty for this iteration. There is no parametric uncertainty (flux, efficiency, ...) represented here.

Inelastic reactions results

The choice of transitions to look at have been guided by the strength of the γ ray peak in the spectra, and by the *level scheme* of the ^{183}W nucleus to avoid an isomer.

209 keV

The 209.9 keV transitions in ^{183}W corresponds to the *decay* from the 308.9 keV $\frac{9}{2}^-$ level to the 99.1 keV $\frac{5}{2}^-$ (L06L02 in *Talys nomenclature*). We note that the high resolution of the planar detectors allow the separation of this γ peak from the nearby 208.8 keV decaying from the $\frac{3}{2}^-$ level down to the ground state.

Figure 48 shows the extracted cross section. The cross section central values are represented by the black full line. They are given over the ranges of neutron energy, with uncertainties on the neutron energy (from *Ge detectors timing uncertainty*) represented by the lines extending to the left and right of the ranges. The cross section uncertainties are represented in dashed lines for the *parametric* uncertainties (from the random sampling of the analysis parameters), dotted line for the *intrinsic* uncertainties (from the peak fits in the spectra) and gray box for the total combined uncertainty.

We notice in Figure 48 that both *analysis codes* give similar amplitude in cross section. A kind of *shift* can be seen in the shape of the two results, indicative of a difference in treating the time shift (in particular, there could be a double correction in the *old code*). The uncertainties are lower with the old code, this is probably a combination of two factors: using only the UF_4 fission chamber (reduction in parametric uncertainty), and better fit performances with the *Root* based spectrum analysis (reduction in intrinsic uncertainty).

Talys calculations give similar cross section profile, with the one by P. Romain being slightly larger in amplitude, in particular at higher neutron energy. The differences in amplitude with the experimental points is reminiscent of what

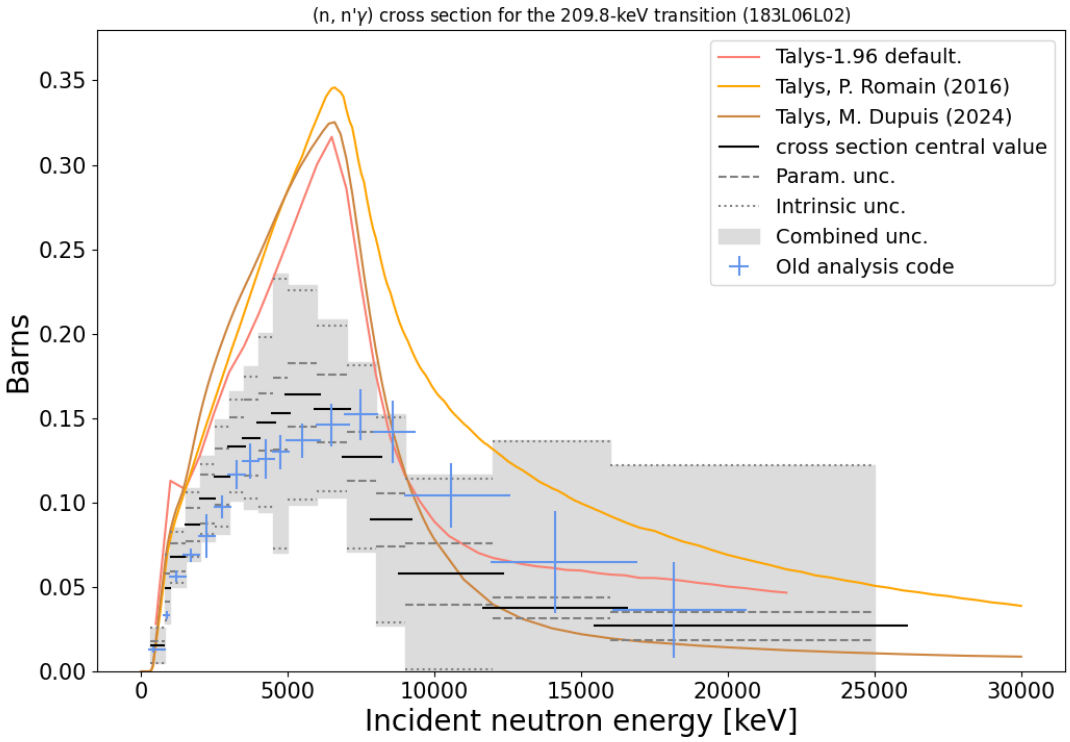


Figure 48: Experimental cross section for the 209.9 keV γ ray in the $(\mathrm{n},\ \mathrm{n}')$ reaction on ^{183}W. See text for full description. The values extracted from the same dataset using the *previous analysis Code* are plotted with the blue points with x and y error bars. The values calculated by *Talys* (either with default input, from P. Romain, or from M. Dupuis) are in full line. See raw values in Listing 14.

has been seen in other nuclei[1,2, and3].

The *correlation matrix* (Figure 49) is over all positive, with a correlation between points, in average, in the range 0.25 to 0.5. This is not surprising, as most of the parameters applied to the data during analysis to produce the cross section are *scaling* factors (efficiency, number of atoms in the target, ...). Using the *two fission chambers* might also contribute to the strong positive correlation, with the computed cross section being *moved* up or down depending on the FC used. In that case, the low correlation factor at high neutron energy ($E_n \approx 20$ to 25 MeV) can be related to *the negative correlation for the flux at these energies* being averaged out by integrating the number of neutron over the energy window.

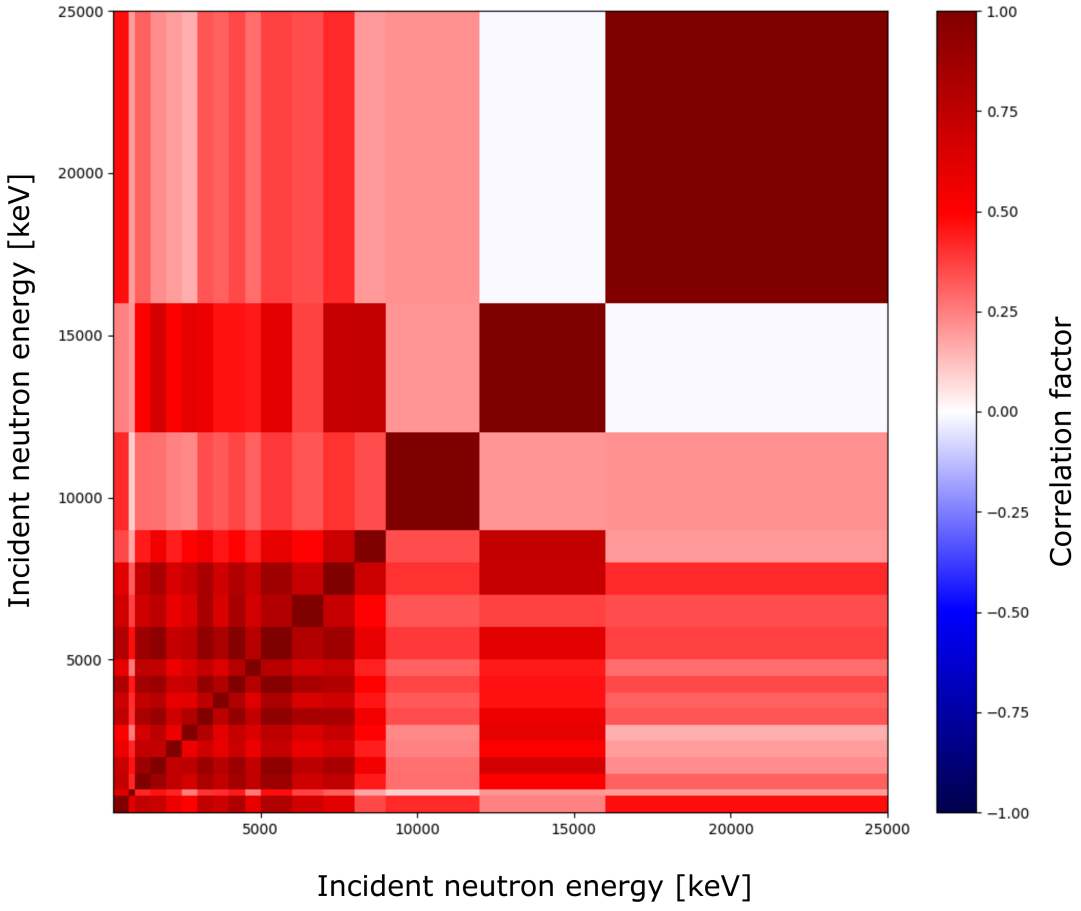


Figure 49: Correlation matrix for the 209.9 keV γ ray in the (n, n') reaction on ^{183}W. The correlation matrix is computed only for the parametric uncertainty.

[1] "Measurement of ^{238}U $(n, n' \gamma)$ cross section data and their impact on reaction models." M. Kerveno, M. Dupuis, A. Bacquias, F. Belloni, D. Bernard, *et al.* Physical Review C, 2021, 104 (4), pp.044605. ⟨10.1103/PhysRevC.104.044605⟩.

[2] "MEASUREMENT OF ^{182,184,186}W $(n, xn \gamma)$ CROSS SECTIONS AND WHAT WE CAN LEARN FROM IT." G. Henning, Antoine Bacquias, Catalin Borcea, Mariam Boromiza, Roberto Capote, *et al.* EPJ Web of Conferences, 2021, 247, pp.09003. ⟨10.1051/epjconf/202124709003⟩.

[3] "Improving the accuracy of 182,184,186W(n, n'γ) cross sections calculations" G. Henning, M. Dupuis, M. Kerveno, R. Capote, Ph. Dessagne, S. Hilaire, T. Kawano, P. Romain, C. de Saint Jean, P. Scholtes, P. Tamagno. In preparation for Physical Review C. (2024)

259 keV

The 259.5 keV transitions in ^{183}W corresponds to the *decay* from the 551.2 keV $\frac{9}{2}^-$ level to the 291.7 keV $\frac{5}{2}^-$ (L13L05 in *Talys nomenclature*). Figure 50 shows the extracted cross section.

For this transition, there is no other data to compare with. The extracted cross section stops at 10 MeV, as further up in energy, there is a contamination from a 260 keV γ ray from the ^{182}W (weakly populated in $(n,\ 2n)$ reaction – see *Extension of 183W analysis*).

As previously, the *Talys* calculations give larger cross sections than the experimental values, in a way that has already been seen in other isotopes.

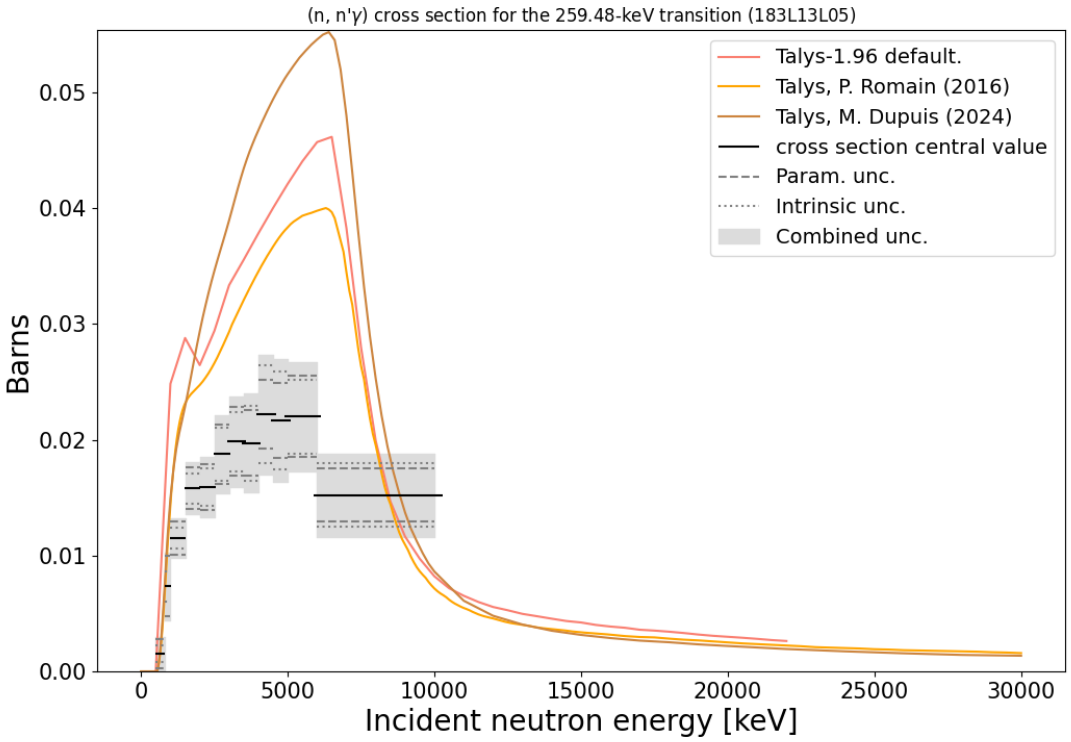


Figure 50: Experimental cross section for the 259.5 keV γ ray in the $(n,\ n')$ reaction on ^{183}W. See text and Figure 48 for full description. See raw values in Listing 15.

Just like *the previous studied transition*, the correlation matrix (Figure 51) is positive all over, indicating a strong correlation, mainly due to the target mass and efficiencies factors, common to all neutron energy windows.

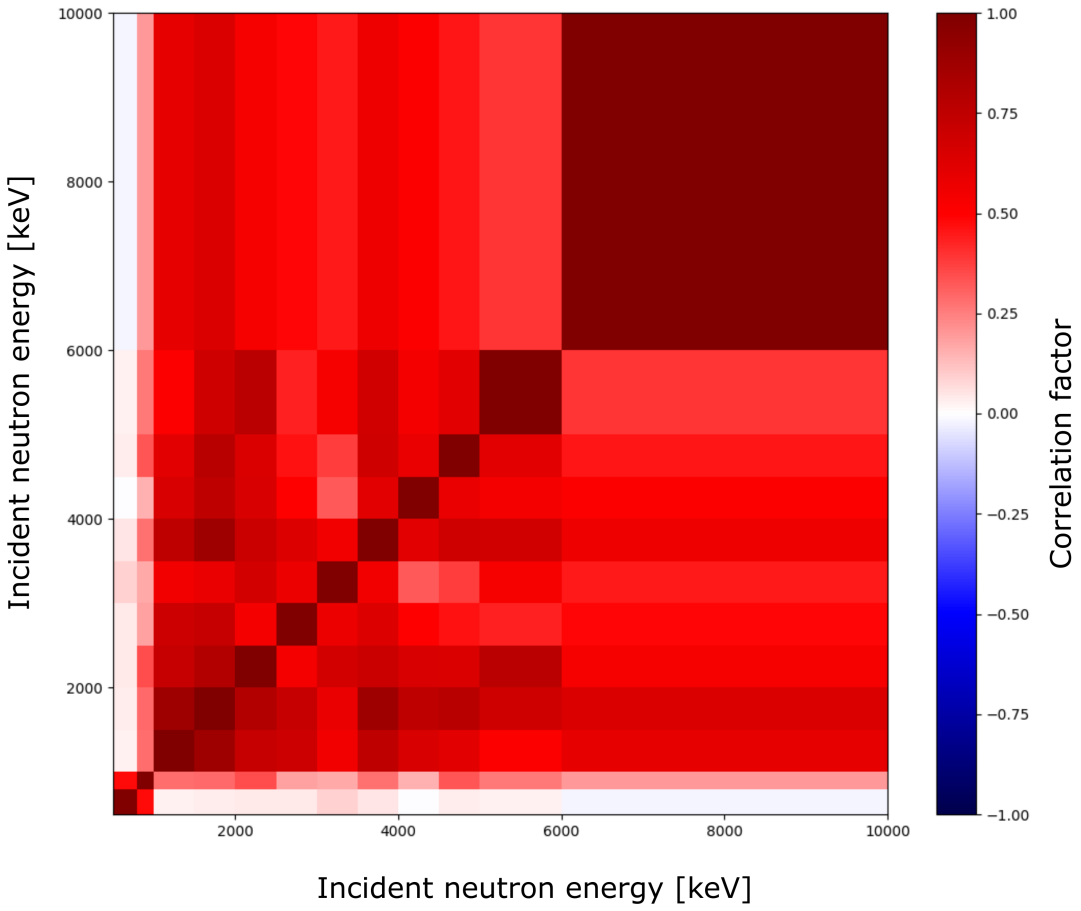


Figure 51: Correlation matrix for the 259.5 keV γ ray in the $(\mathrm{n},\ \mathrm{n}')$ reaction on ^{183}W. The correlation matrix is computed only for the parametric uncertainty.

268 keV

The 268.1 keV transition in ^{183}W corresponds to the *decay* from the 475.1 keV $\frac{11}{2}^-$ level to the 207.0 keV $\frac{7}{2}^-$ (L10L03 in *Talys nomenclature*).

In Figure 52, it is compared to values from P. Scholtes, as well as cross section extracted from the same data set using the old analysis code. The values from the previous analysis code are compatible with the ones extracted with the new code. The main differences come from a different choice of neutron energy ranges. The Scholtes' values are slightly higher in amplitude than ours.

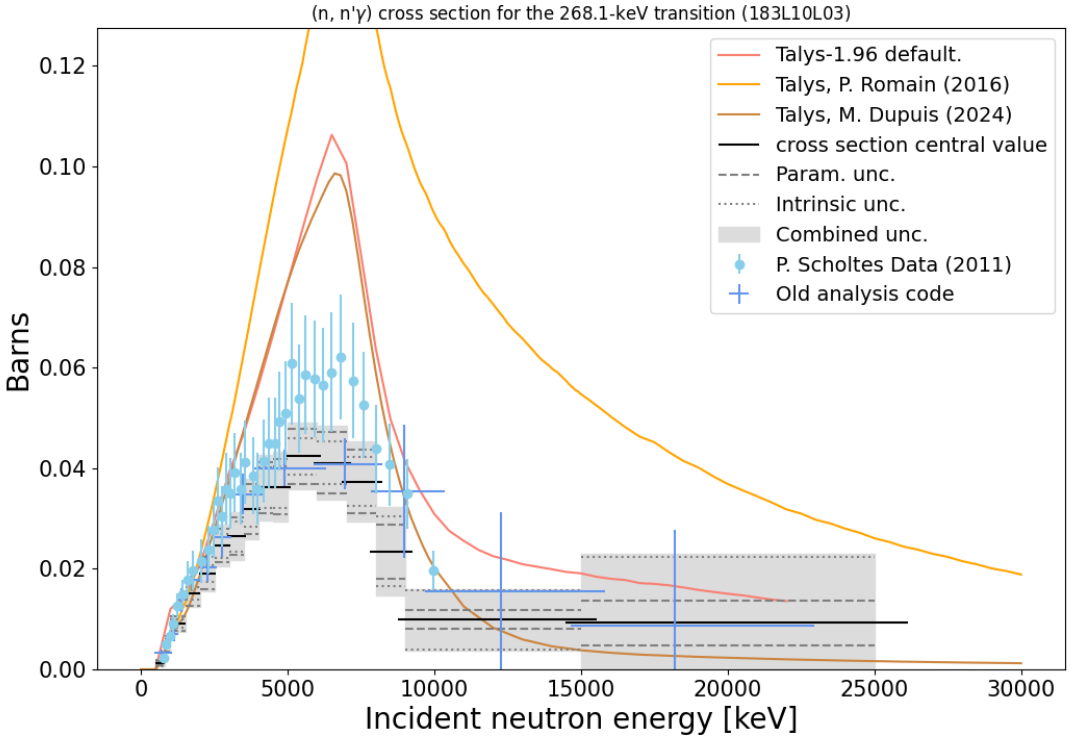


Figure 52: Experimental cross section for the 268.1 keV γ ray in the $(\mathrm{n},\ \mathrm{n}')$ reaction on ^{183}W. See text and Figure 48 for full description. See raw values in Listing 16.

Just like previously, the correlation matrix (Figure 53) is mostly positive all over. The exception is a slightly negative (-0.228) correlation between the lowest neutron energy window (500 keV $< E_\mathrm{n} <$ 800 keV) and the highest energy one (15 MeV $< E_\mathrm{n} <$ 25 MeV). This might be related to the time uncertainty, which results in event distribution being *moved* between high and low neutron energies. Indeed, the energy span is important for this transition, and the effect would be enhanced (compared to transitions with large neutron energy span, where such negative correlation is not observed) by the low cross section value in the low neutron energy window (0.00125 barn) with large parametric uncertainty (0.00036 barn, or 28.8 % relative). The *neutron flux* also presents a negative correlation between the low neutron energy and the 25 MeV region, that could contribute to the observed values in the correlation matrix for the γ ray.

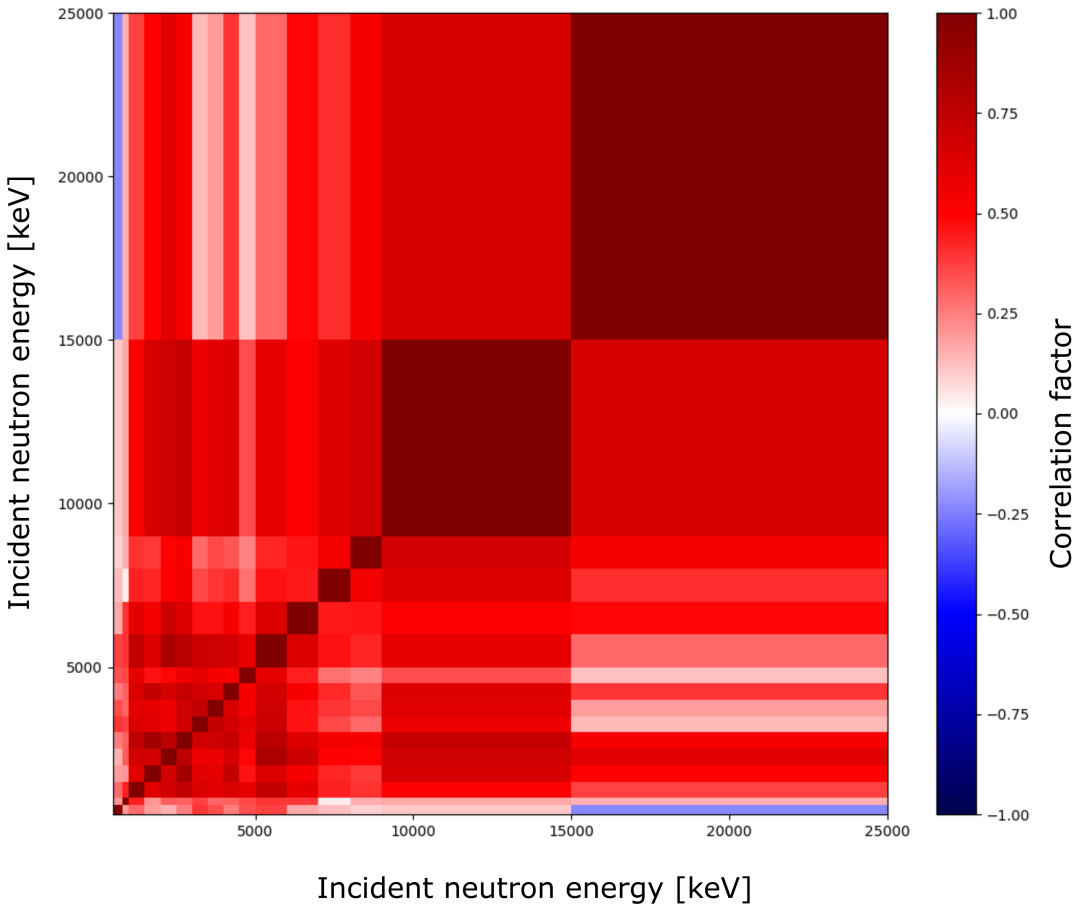


Figure 53: Correlation matrix the the 268.1 keV γ ray in the $(\mathrm{n},\ \mathrm{n}')$ reaction on ^{183}W. The correlation matrix is computed only for the parametric uncertainty.

291 keV

The 291.7 keV transition in ^{183}W corresponds to the *decay* from the 291.7 keV $\frac{5}{2}^-$ level to the ground state ($\frac{1}{2}^-$) (L05L00 in *Talys nomenclature*). Figure 54 shows the cross section. The experimental values are compatible with the Talys ones, thanks to large intrinsic uncertainties.

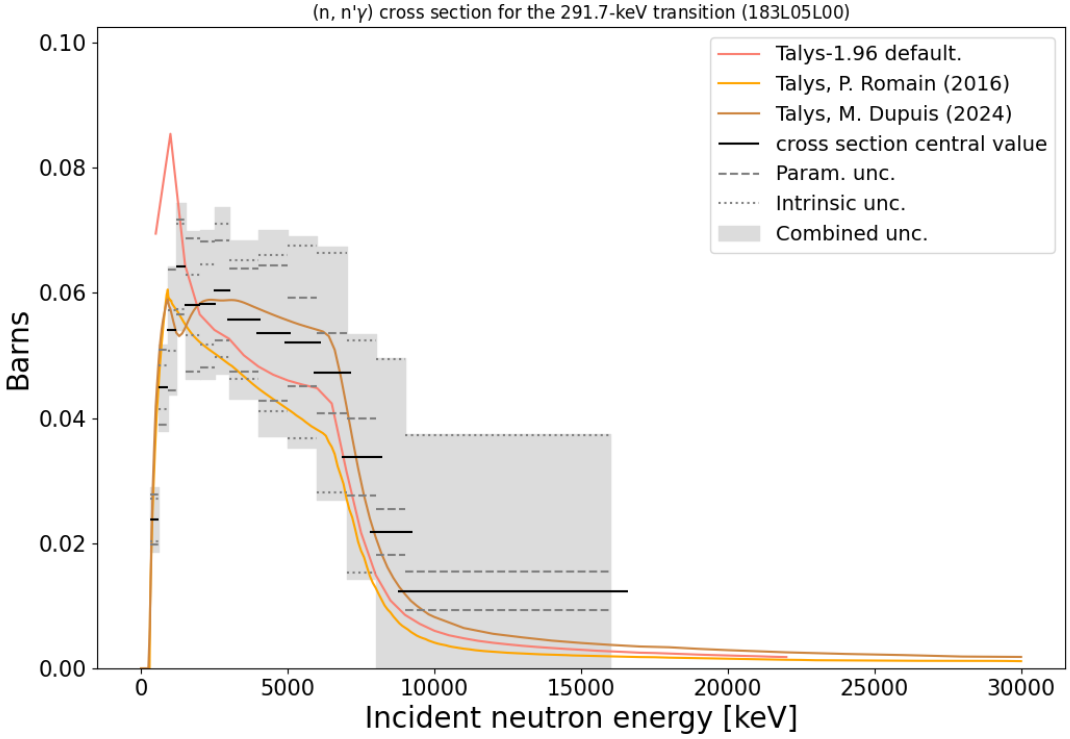


Figure 54: Experimental cross section for the 291.7 keV γ ray in the $(\mathrm{n},\ \mathrm{n}')$ reaction on ^{183}W. See text and Figure 48 for full description. See raw values in Listing 17.

As in other transitions, the correlation matrix (Figure 55) is overall positive. The two lower correlation areas, around 4 and 8 MeV, could be linked to the same *low correlation factors seen in the flux at these energies*.

313 keV

The 313.0 keV transition in ^{183}W corresponds to the *decay* from the 412.1 keV $\frac{7}{2}^-$ level to the 99.1 keV $\frac{5}{2}^-$ (L08L02 in *Talys nomenclature*).

It shows (in Figure 56) reasonable uncertainties (thanks to larger cross section than the previous ones). The cross section amplitude is larger than Talys predictions, again, not in a way that is so different from seen before.

As previously, the correlation matrix (Figure 57) is overall positive.

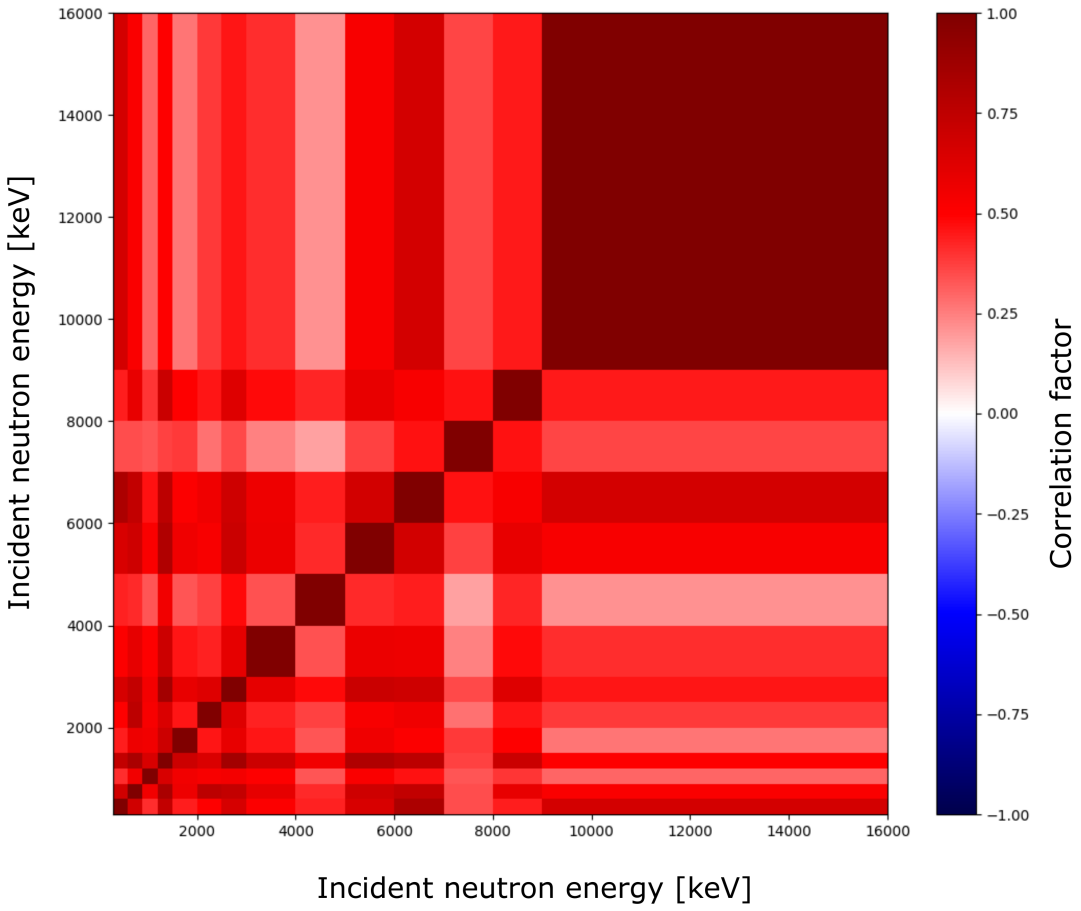


Figure 55: Correlation matrix for the 291.7 keV γ ray in the $(\mathrm{n},\ \mathrm{n}')$ reaction on ^{183}W. The correlation matrix is computed only for the parametric uncertainty.

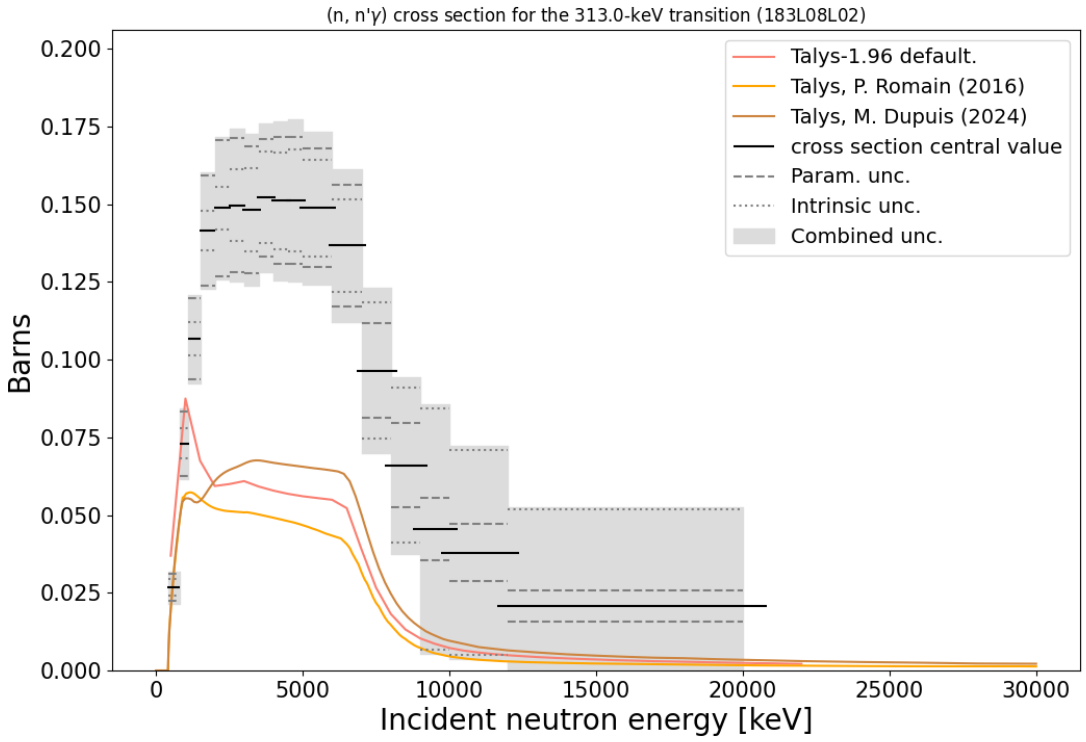


Figure 56: Experimental cross section for the 313.0 keV γ ray in the $(\mathrm{n},\ \mathrm{n}')$ reaction on ^{183}W. See text and Figure 48 for full description. See raw values in Listing 18.

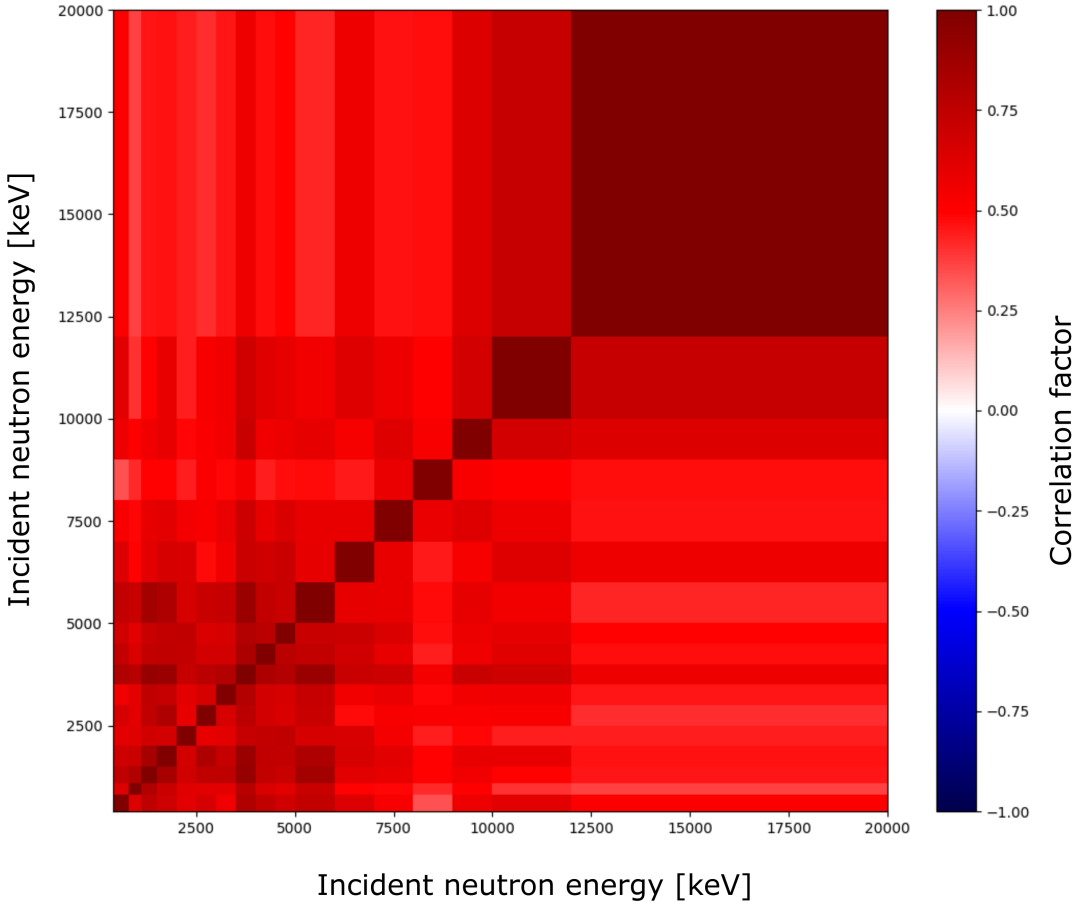


Figure 57: Correlation matrix for the 313.0 keV γ ray in the $(\mathrm{n},\ \mathrm{n}')$ reaction on ^{183}W. The correlation matrix is computed only for the parametric uncertainty.

354 keV

The 353.9 keV transition in ^{183}W corresponds to the *decay* from the 453.1 keV $\frac{7}{2}^-$ level to the 99.1 keV $\frac{5}{2}^-$ (L09L02 in *Talys nomenclature*).

Figure 58 show the extracted cross section. The quality of the obtained values is poor above 4 MeV. We notice that the central values are compatible with Talys predictions and just lower than the Scholtes values (as was also the case in the *The 268.1 keV transition*).

The parametric uncertainties have reasonable values. It is the intrinsic uncertainty that pushes the total combined uncertainty to the highest values (up to 31865 barns in the 4 to 6 MeV window !!!). Clearly, these are not correct, and the issue lies, it seems, with the fit procedure, as revealed in Figure 59 (see *Fit improvements*).

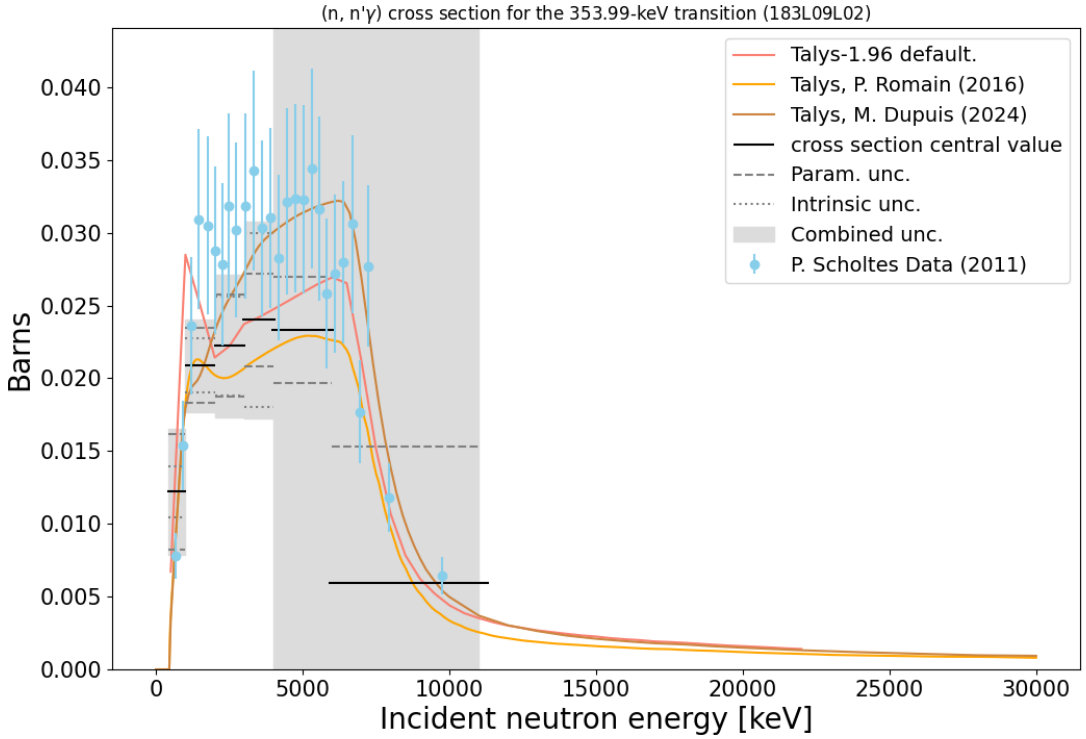


Figure 58: Experimental cross section for the 353.9 keV γ ray in the $(n,\ n')$ reaction on ^{183}W. See text and Figure 48 for full description. See raw values in Listing 19.

Listing 8: Extracted peak values from the fit represented in Figure 59. The keys ending in _u are the uncertainties associated with the corresponding variables.

```
integral: 1108.3233832129263
integral_u: 6739722924.904172
mean: 352.75049647945286
mean_u: 298366477.9849391
stdev: 252.8659491602733
stdev_u: 516019109.0222086
```

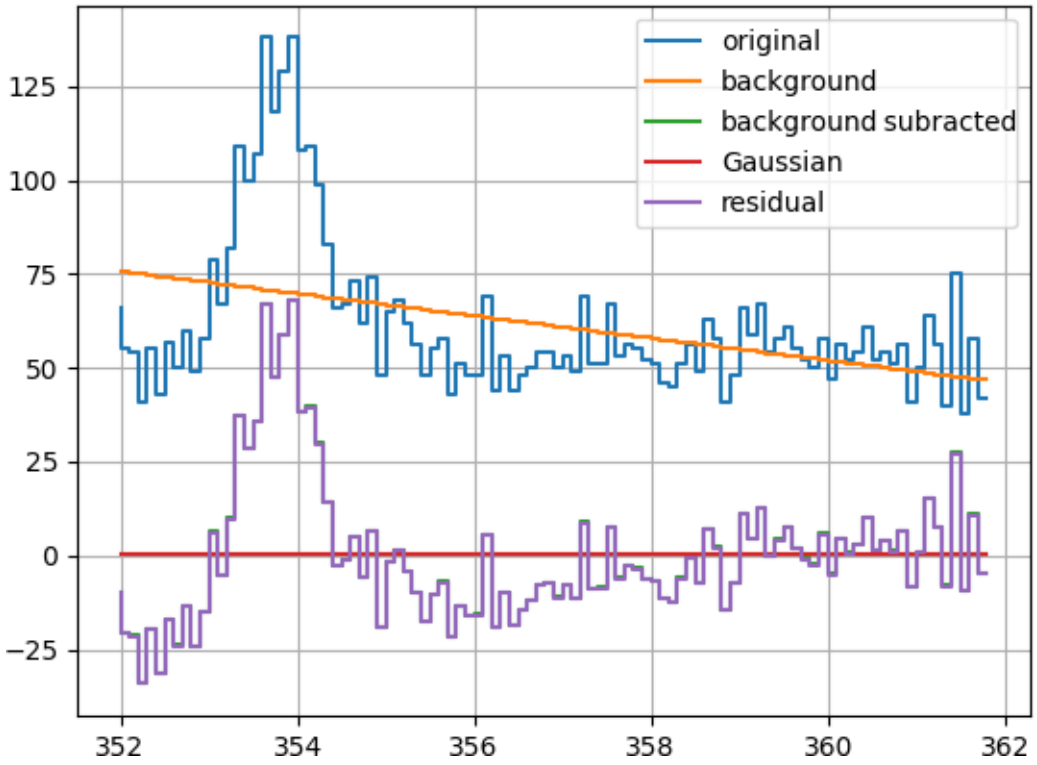


Figure 59: Plot of the histogram projected for incident neutron between 4 Mev and 6 MeV for the *vert* detector. The lines represent the fitted background and peak. The fit clearly failed here. The output of the fitting script is shown in Listing 8.

Looking into the projection and fit on the γ peak in the histogram (Figure 59), we clearly see that the fit failed, for unclear reasons, since human can identify the peak easily in the *spectrum*. The fitting procedure is clearly the culprit here. However, the fit did finish "properly", as indicated by the content of the fit result file, shown in Listing 8. The fit finished, with a reasonable integral value, but the peak width and uncertainties are all wrong (the mean value is also off by more than 1 keV). The wrong fit results points to needed improvement in the *first guess* values and parameters constrains given to the fitting procedure. More investigation is needed, and fixing the issues will help reduce uncertainties for all transitions. (see *Fit improvements*).

Because of the fitting issues, the interpretation of the correlation matrix (Figure 57) should be done carefully. The last energy window in particular (6 MeV $< E_{\mathrm{n}} <$ 11 MeV) is obviously decorrelated because of random, failed, fits. No more conclusion should be drawn before the fit has been fixed.

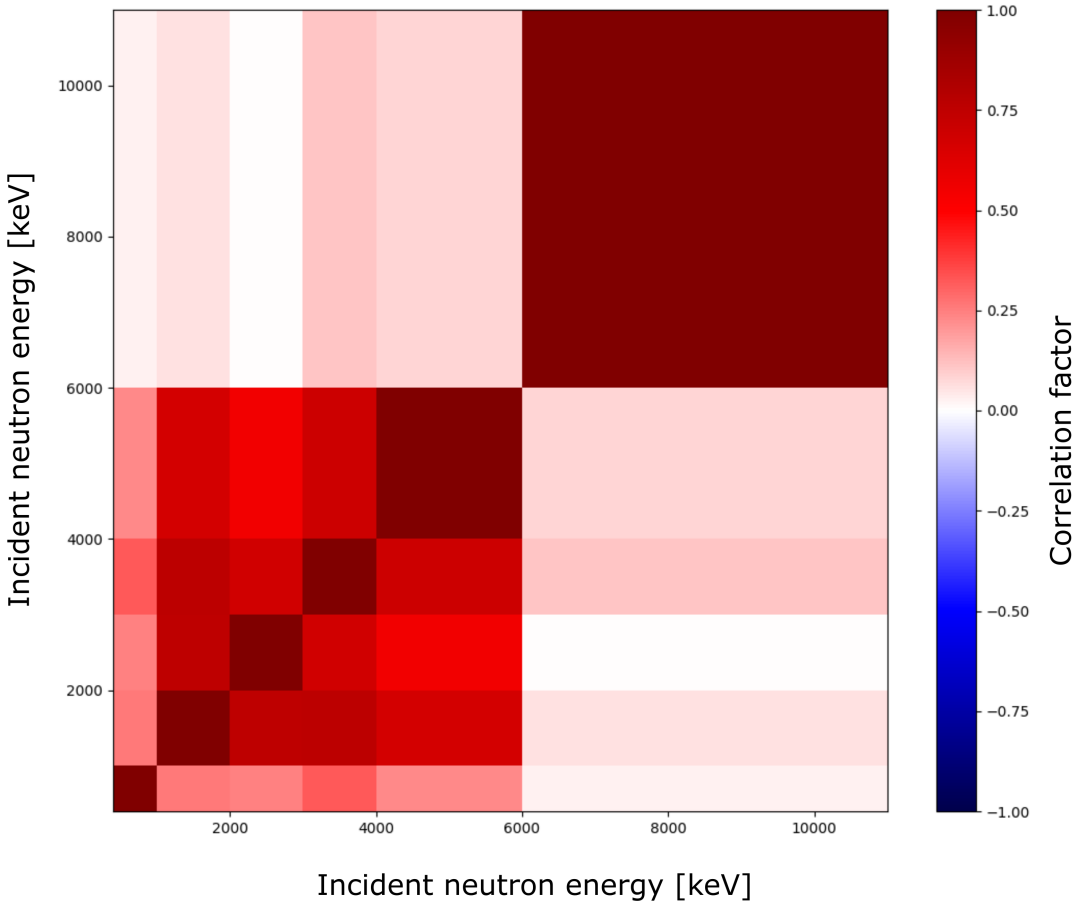


Figure 60: Correlation matrix for the 353.9 keV γ ray in the $(\mathrm{n},\ \mathrm{n}')$ reaction on ^{183}W. The correlation matrix is computed only for the parametric uncertainty.

Update

See in *Appendix* for updated results on this transition, following *Fit improvements* and *Fit script improvement*.

(n, 2n) results

229 keV

The 229.3 keV transition in ^{182}W, from the $(n,\ 2n)$ reaction on ^{183}W, corresponds to the decay from the 329.4 keV 4^+ level to the 100.1 keV 2^+ (L02L01 in *Talys nomenclature*).

The result (Figure 61) shows the experimental data below the Talys predictions. The data extracted from the Grapheme dataset using the previous analysis code show a significant difference from our result, which may be a combination of difference in fit and overcorrection of the time shift for low energy γ rays in the old version of the analysis. The *Talys* calculations by P. Romain[1] are quite different, and more compatible with the experimental results, from the default ones and the M. Dupuis's one[2].

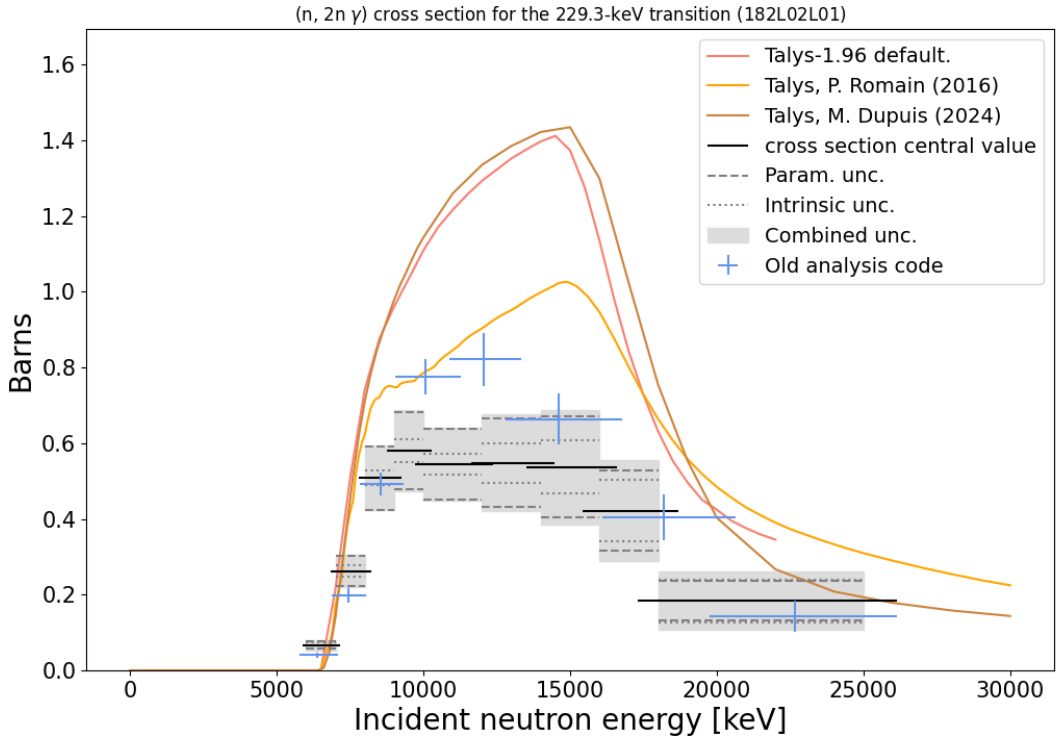


Figure 61: Experimental cross section for the 229 keV γ ray in the $(n,\ 2n)$ reaction on ^{183}W (transition in the ^{182}W isotope.). See text and Figure 48 for full description. See raw values in Listing 20.

The correlation matrix (Figure 62) is, as for the *inelastic results*, overall positive.

[1] Done by Pascal Romain (CEA/DAM/DIF) in 2016, private communication.
[2] Done by Marc Dupuis (CEA/DAM/DIF) in 2024, private communication.

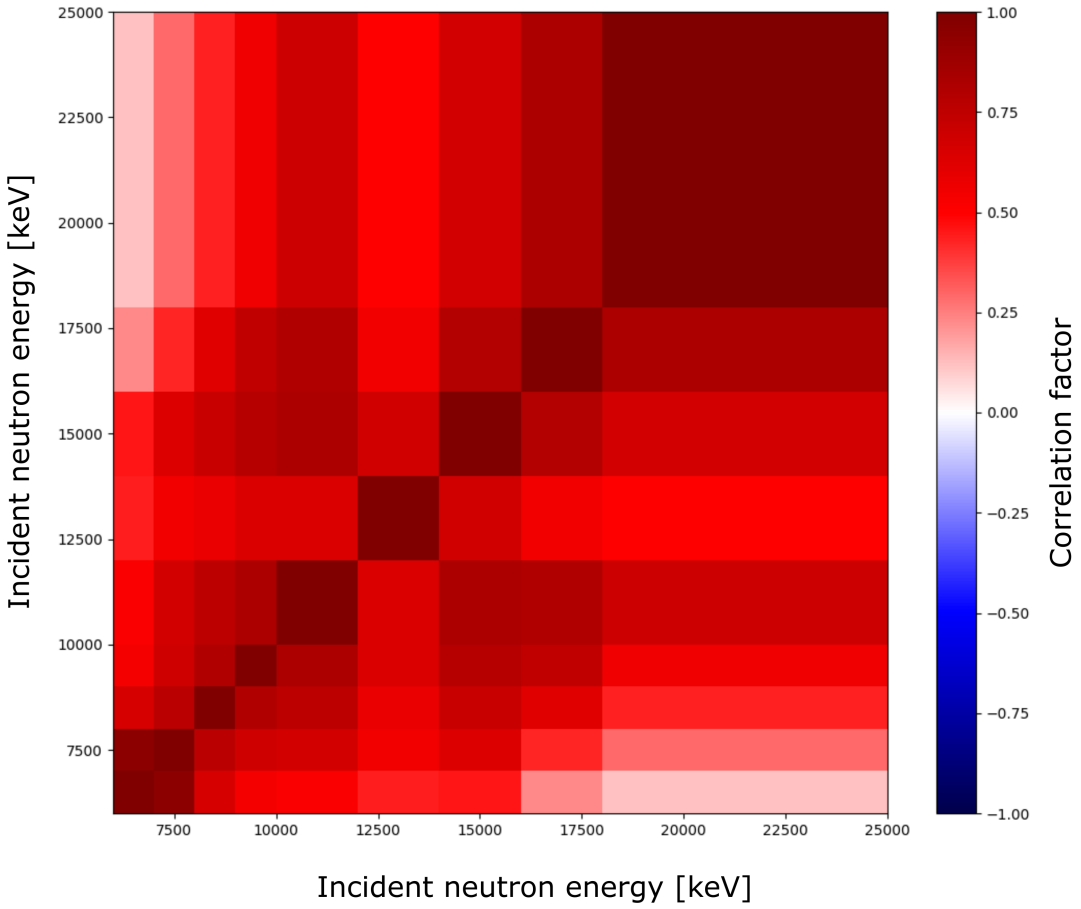


Figure 62: Correlation matrix for the 229 keV γ ray in the $(\mathrm{n},\ 2\mathrm{n})$ reaction on ^{183}W.

351 keV

The 351.0 keV transition in ^{182}W, from the $(\mathrm{n},\ 2\mathrm{n})$ reaction on ^{183}W, corresponds to the decay from the 680.44 keV 6^+ level to the 329.4 keV 4^+ (L03L02 in *Talys nomenclature*).

The result (Figure 61), like the *previous* $(\mathrm{n},\ 2\mathrm{n}\ \gamma)$ transition, shows the experimental data below the predictions, with the three calculations closer together. The data extracted with the previous analysis code show a small difference from the new result, hinting, again, at overcorrection of the time shift.

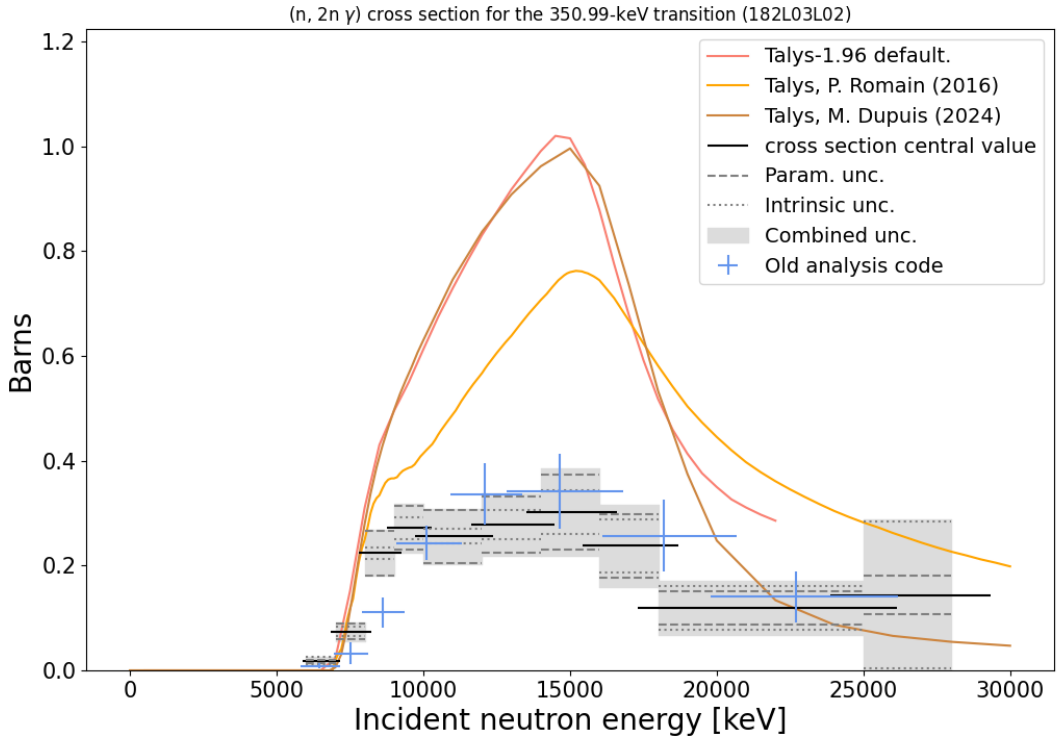


Figure 63: Experimental cross section for the 351.0 keV γ ray in the $(\mathrm{n},\ 2\mathrm{n})$ reaction on ^{183}W (transition in the ^{182}W isotope.). See text and Figure 48 for full description. See raw values in Listing 21.

Just as previously, the correlation matrix (Figure 64) is mostly positive.

5.2 Discussion

5.2.1 Critic of results

The first positive element from the new analysis implementation that should be mentioned is that having all the intermediate steps spectra, values, *etc.* saved is a great help for debugging and understanding results (as could be seen in the *354 keV* case for example). That's how we can tell that, apart from the few elements mentioned before, the results presented are a correct reflection of the data set.

The (overall small) differences with the results extracted from the same dataset using the previous analysis code support the idea that an over correction (or at least incorrect correction) of the time shifts is applied. This is expected, in regard

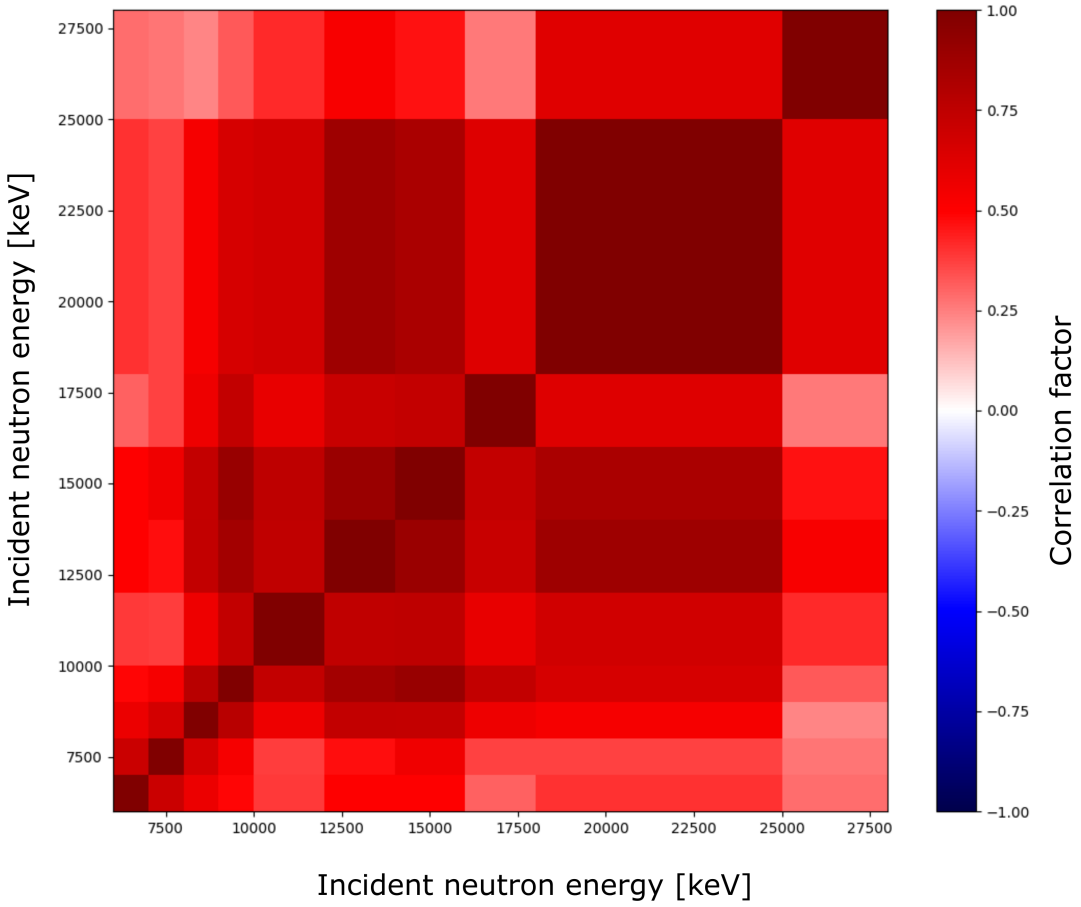


Figure 64: Correlation matrix for the 351.0 keV γ ray in the $(\mathrm{n},\ 2\mathrm{n})$ reaction on ^{183}W.

to the number of lines in the *previous analysis code* (1140 lines of code in the main **function**, calling many outside dependence). It would take a long time to go through the original code in order to find the proper way to input correctly the time shift parameters for a new dataset.

The compatibility of experimental values with *Talys* is limited, but not particularly worse than for other nuclei we studied already[1,3, and2]. For comparison, Figure 65 shows an example of results on even-even isotopes, compared to predictions from reactions codes. The agreement between experimental values from different analysis (*previous code* and *P. Scholtes' values*) gives us added confidence in the results.

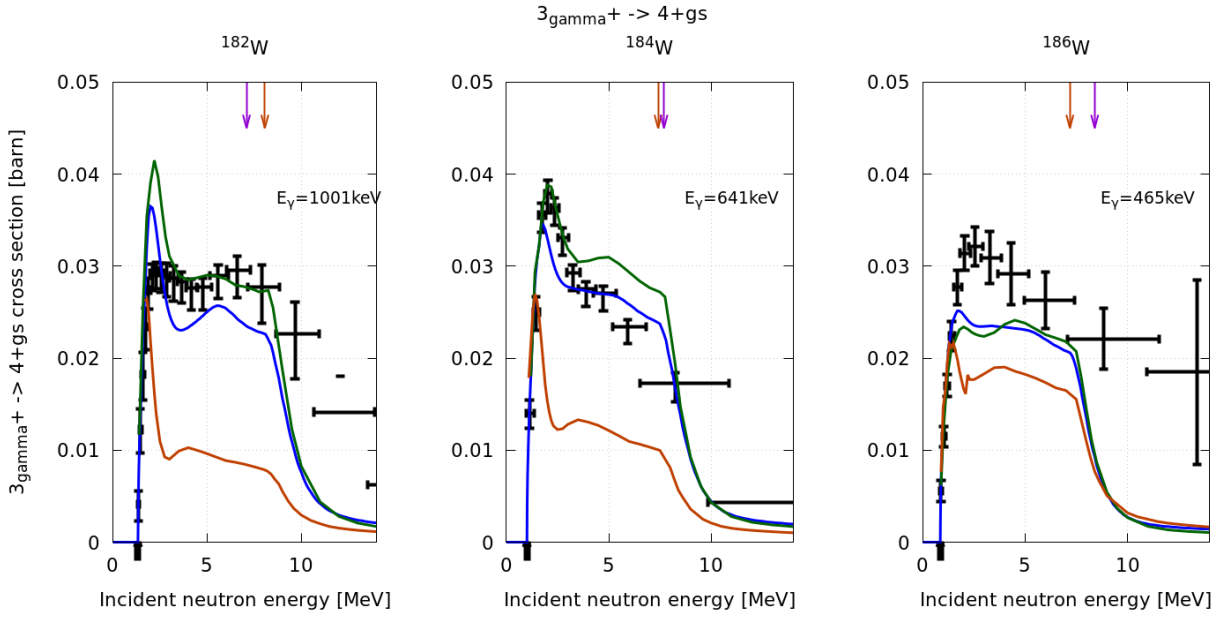


Figure 65: (n, n′ γ) cross sections measured with *Grapheme* for the isotopes ^{182}W (left), ^{184}W (center), and ^{186}W (right), for the transitions from the 3^+ state in the γ band (built on the 3^+ state) to the 4^+ in the ground state rotational band. The data from isotopic target is represented with black crosses. It is compared to *Talys*-1.73 (blue line), *Empire* (orange line) and *CoH3* (green line) calculations. The excitation energy of the decaying level is marked with the black up-pointing arrow on the bottom axis. The neutron separation energy S_n and proton separation energy S_p are marked with the orange and purple down-pointing arrows at the top, respectively. From[1 and2].

The correlation matrices are overall positive (almost *flat*) with an average correlation factor typically 0.5 (with a standard deviation of 0.25). This is expected as the target mass and HPGe detection efficiencies are the main source of parametric uncertainties, which apply to all the neutron energy windows in the same way and thus creating high correlation. The effect of the *two fission chambers* might be *enhancing* the *positive correlations*.

When compared to previous correlation matrices, either from *another Monte Carlo analysis* (Figure 66) or from a

[1] "MEASUREMENT OF ^{182,184,186} W (n, n′ γ) CROSS SECTIONS AND WHAT WE CAN LEARN FROM IT." G. Henning, Antoine Bacquias, Catalin Borcea, Mariam Boromiza, Roberto Capote, et al.. EPJ Web of Conferences, 2021, 247, pp.09003. https://doi.org/10.1051/epjconf/202124709003

[3] Greg Henning, Maëlle Kerveno, Philippe Dessagne, François Claeys, Nicolas Dari Bako, et al.. Using the Monte-Carlo method to analyze experimental data and produce uncertainties and covariances. 15th International Conference on Nuclear Data for Science and Technology (ND2022), Jul 2022, Online Conference, United States. pp.01045, ⟨10.1051/epjconf/202328401045⟩.

[2] "Improving the accuracy of 182,184,186W(n, n'γ) cross sections calculations" G. Henning, M. Dupuis, M. Kerveno, R. Capote, Ph. Dessagne, S. Hilaire, T. Kawano, P. Romain, C. de Saint Jean, P. Scholtes, P. Tamagno. In preparation for Physical Review C. (2024)

semi-Monte Carlo approach[4,5, and6] (Figure 67), our results are not too different, with mostly positive corrections. The negative correlations observed with other analysis maybe explained by the use of only the UF_4 fission chamber, reducing the global effect of the fission chambers and leaving others (such as time uncertainty) to have a greater impact on the correlation factors. Still, it is important to remember that the correlation matrix reflects how the data was analyzed (see *earlier*). Therefore, it is normal that different analysis methods will yield different matrices, even if the results (central values and uncertainties) are very compatible.

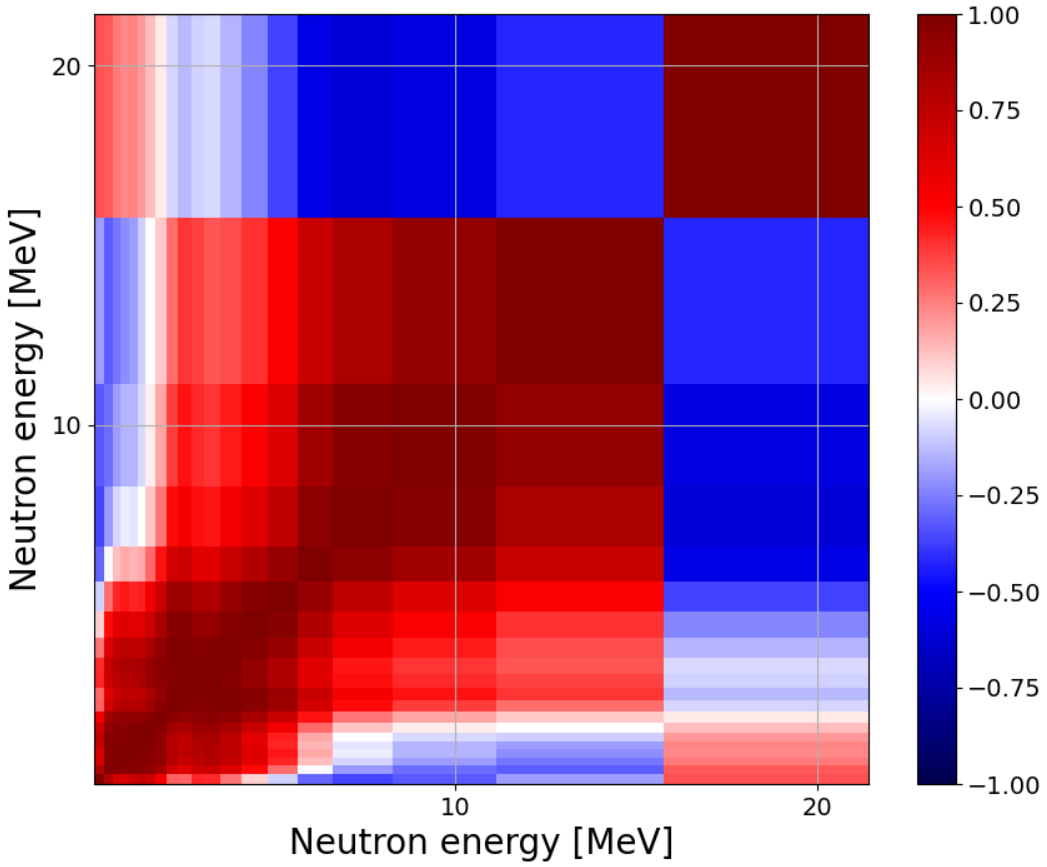


Figure 66: Correlation matrix for the 111 keV transitions in ^{184}W, obtained using the *previous full Monte Carlo analysis code*. This analysis used only the UF_4 fission chamber for the flux determination. We also noticed a larger sensitivity to the *time jitter*, leading to the positive/negative structure at the lowest and highest neutron energy windows[Page 94, 3,Page 94, 1, andPage 94, 2].

Of course, as we saw earlier, there are still elements to improve, mostly on the fitting procedure. It is on top of the list of *future improvements*.

[4] Greg Henning, Francois Claeys, Nicolas Dari Bako, Philippe Dessagne, Maelle Kerveno. "Producing uncertainties and covariance matrix from intermediate data using a Monte-Carlo method."" 6th International Workshop On Nuclear Data Evaluation for Reactor applications (WONDER), Jun 2023, Aix-en-Provence, France. hal-04126156

[5] "Producing uncertainties and covariance matrix from intermediate data using a Monte-Carlo method." Greg Henning, François Claeys, Nicolas Dari Bako, Philippe Dessagne and Maëlle Kerveno. EPJ Web of Conf., 294 (2024) 05002 https://doi.org/10.1051/epjconf/202429405002

[6] Francois Claeys, "Mesure, modélisation et évaluation de sections efficaces à seuil (n, xn γ) d'intérêt pour les applications de l'énergie nucléaire". Thèse de doctorat, 2023. Chapter 2.4 "Resimulateur" https://theses.fr/2023STRAE027

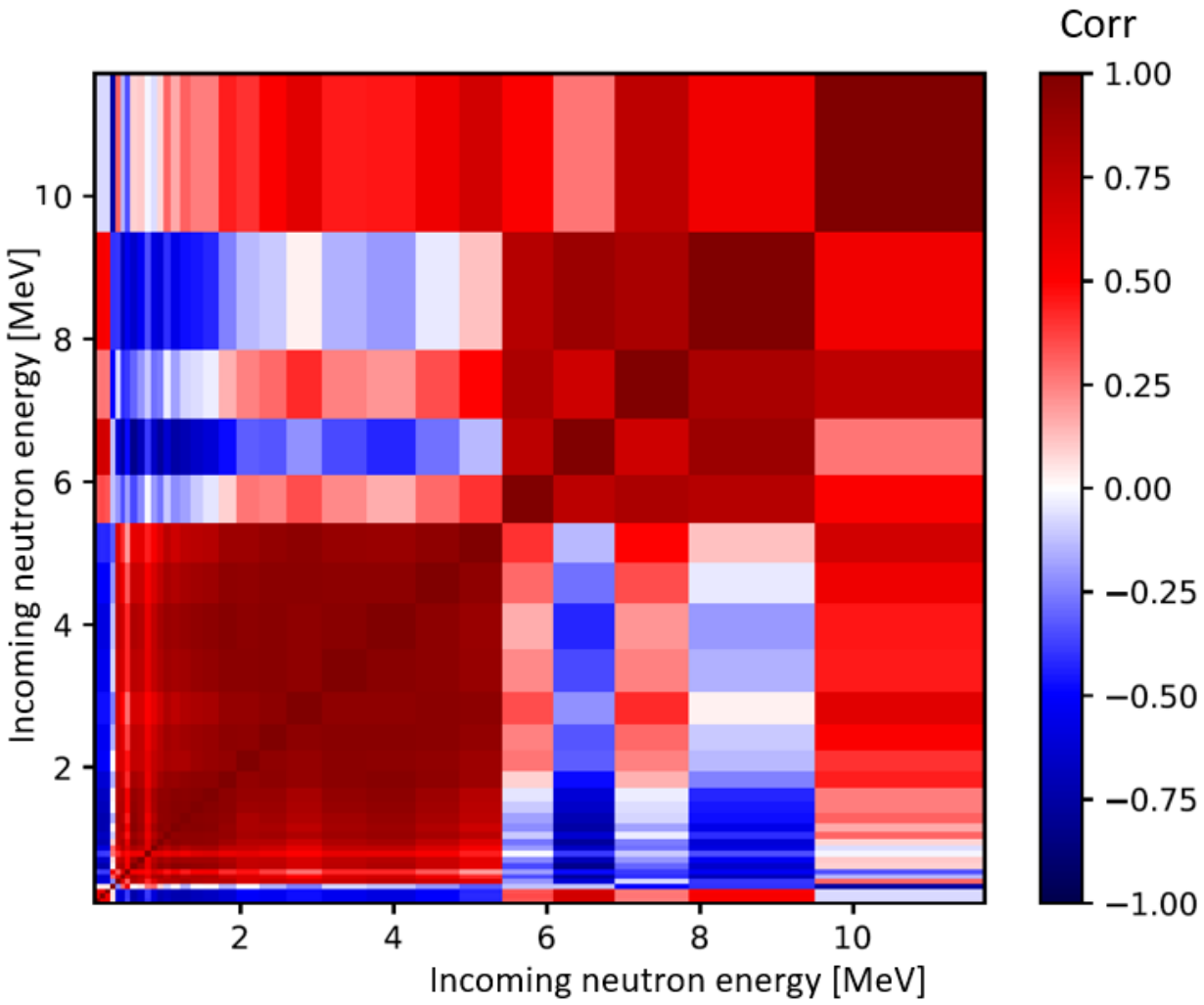


Figure 67: Correlation matrix for the 153 keV transitions in ^{238}U, obtained using a semi Monte Carlo method[4,Page 95, 5, andPage 95, 6]. The positive/negative structure between 6 and 10 MeV are attributed to existing structures in the source data. In this analysis method, there is no intrinsic uncertainty on the number of γ rays, and all the uncertainties are purely parametric.

5.2.2 SWOT analysis

To conclude on the interest of the new analysis code, we draw a Strength-Weakness-Opportunity-Threat analysis:

Strengths
: The code is a *Full Monte Carlo analysis* of the data, including sources of uncertainties that could be difficult to include in an analytical version (namely, uncertainties on parameters impacting the time of flight / neutron energy). By *saving all the intermediate files*, the debugging or study of *suspicious results* is made much simpler, with the ability of *traveling* back from a result file to the different previous steps leading to it.

Weaknesses
: The processing is currently very slow (about 2 hours per one full Monte Carlo iteration[7]). The fitting procedure has been shown to be *fragile*, with some *unexpected failure* to produce a meaningful value at times.

Opportunities
: *New data* from *Grapheme*, as well as data that will be recorded at *NFS*, will benefit greatly from this *complete* and flexible analysis code.

Threats
: There is a dual risk in the future for such a code. Either it is made so specific that any new case (in terms of number of detectors, ...) requires to completely rewrite and add new modules. Or it could become too *heavy*, with so many options that most of the user's time is taken building the *pipeline*. What we want to avoid is to have a *repeat* of the *previous code's fate*, and have a code so specific to a dataset that it cannot be reused for new data.

The *following part* will discuss how we can improve the code, in answers to *weaknesses*, while being careful not to fall in the *threats* pitfalls.

5.2.3 Perspectives

There are several perspectives for improvement on different plans: the inner working of the code, the way the users interact with the code, and the physics that can be treated by the code.

Fit improvements

The top priority is to improve the *fit procedure*.

Currently, the fit is performed using the `curve_fit` method of the `scipy` library for python[8] [,9]. This method uses a least-square minimization of the formula used to describe the spectrum shape (linear background and Gaussian peaks). The fit uses the *Trust Region Reflective* fitting method, (the default for a minimization with bounds), and the use of this default method or the other offered by the `scipy` library (e.g. *dogleg algorithm*) was not investigated. The *Root code* is relying on the *log likelihood* fitting method, more adapted to the adjustment on count histograms.

One way of improvement for the fit would be to implement a *log likelihood* fit. This can be done in `scipy` by subclassing the `rv_discrete` base class in order to describe the spectrum shape. This method, requires a significant amount of work and testing before being validated.

Quicker to implement, one can test the *dogleg* fitting algorithm of `curve_fit` or, with slightly more work, the different fitting methods allowed by *scipy*'s `optimize.minimize` method.

[7] This time is mostly devoted to file reading and writing, as well as dependency computation (see *Technical code improvements*). Each iteration generates hundreds of files.

[8] Pauli Virtanen, Ralf Gommers, Travis E. Oliphant, Matt Haberland, Tyler Reddy, David Cournapeau, Evgeni Burovski, Pearu Peterson, Warren Weckesser, Jonathan Bright, Stéfan J. van der Walt, Matthew Brett, Joshua Wilson, K. Jarrod Millman, Nikolay Mayorov, Andrew R. J. Nelson, Eric Jones, Robert Kern, Eric Larson, CJ Carey, İlhan Polat, Yu Feng, Eric W. Moore, Jake VanderPlas, Denis Laxalde, Josef Perktold, Robert Cimrman, Ian Henriksen, E.A. Quintero, Charles R Harris, Anne M. Archibald, Antônio H. Ribeiro, Fabian Pedregosa, Paul van Mulbregt, and SciPy 1.0 Contributors. (2020) SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python. Nature Methods, 17(3), 261-272.

[9] SciPy - Fundamental algorithms for scientific computing in Python

An additional work on the starting values given to the fitting procedure, and the bounds for the parameters, might also improve the results. In particular, the *guess* for the background level could be improved.

Finally, the current fitting script has been written to be completely flexible and accommodate any number of *parasites*. To simplify the code and reduce the number of parameters, one could write *fall back* scripts that will perform the fit on more specific function in usual cases (particularly when there is only one parasite). (That goes against the principle that *special cases aren't special enough to break the rules*, but on the other hand, *practicality beats purity*.)

Update

See in *Appendix* for updated information on the fit improvements.

User improvements

For the general users, the current code should be usable *out of the box*, but to accommodate other experiments, some modifications will be helpful.

The definition of the analysis *pipeline* should be made easier, by providing dedicated scripts to set up the different analysis steps. A `yaml` file, following the gitlab's CI/CD model, could be a way to implement this. However, that will require a significant amount of work for a small gain. A series of `bash` scripts to configure and run the pipeline is easier to envision and will do the job nicely enough. Additionally, a tool in python (class, decorator, ...) to simplify the file and task dependency management could be created, so that the creation of new tasks is easy, without needing to understand the details of `doit`. In particular, a task factory could be created for simple / usual tasks (2D histogram projection, histogram scaling, ...).

On the Monte Carlo side, it would be good to have the possibility to run the iterations *side-by-side*. This will be possible by adding a *setup* step at the start, when a temporary working directory will be created, with links to the source files of interest. The iteration will run entirely with the files in this directory, allowing several jobs to run in parallel, each in their own working environment. Combine with the *job dependency* option in `sbatch` to automatically run the post-loops stage, the user could start a full Monte Carlo analysis in one command and get a result in a couple of hours (compared to a couple of days at the moment).

Finally, to make the porting of the framework easier, a *Docker or Singularity* image could be created, that would include the dependencies for the code to run (the tasks file would not be included in the image).

Technical code improvements

A few technical improvements, that will only impact the inner working of the analysis without changing the way it is used (and the way it produces results) are foreseen:

- Put the *context* information (namely, detector names, paths, ...) in a *yaml* file imported as constants in a python module imported by the *doit* tasks scripts, instead of being individually treated at the top of each file. (See *context module in task files* for update.)
- Using this context module, we can rewrite the tasks code, making sure there are no hard-coded values.
- Move the HPGe detector *efficiency sampling* to the *new MC iteration* stage, rather than somewhere in the iteration (see *MC iteration setup*), for consistency. And ensure a proper *central* value sampling of the efficiencies when the iteration `uuid` is `000000` (see *MC iteration setup*).
- The number of parameters files can be reduced. It should be checked that each parameter values (detector distance of flight, efficiency, ...) are written only once in one parameter file.
- The tasks functions still written in the tasks files should be moved at least in an outside module, or (better) in scripts that can be called directly with the command line. (See *context module in task files* for update.)

- The python code formatter "black" has been used in the late part of the project to unify the format of the code. This leads to typically longer files, but generally increased readability. In the future, `black` could be used as a default step in validation of the python script to ensure consistent formatting, as well as making debugging easier.
- The *doit* task dependencies are, at the moment, computed using a file signature that takes time to calculate. Since there are many files involved in the analysis pipeline, a significant amount of time is dedicated to computing the file's `md5` signature (since we run in a Monte Carlo loop, most of the files need to be updated anyway, so this strong requirement on dependency calculation is an overkill). Switching to `timestamp` (a simple matter with `doit`) will save time and make the analysis run much faster. (See *Other updates* for update.)
- *Last but not least*, the current *pyhisto* implementation of histogram has the advantage of having no outside dependency. But being 100 % *Python*, it's execution (when performing projections, cuts, ...) is slow. To gain speed and memory usage, we could rely on the C++ library Boost that implements histograms. The scikit-hep/hist module already implements the `boost::histogram` into python, so we just have to wrap around it with a class that reads and write the `pyhisto` format and interface, so that it can be smoothly interchanged with the current pure python module.

These improvements should allow a faster processing over all and make easier the reuse of the analysis code on different platforms.

Addition to the code

To make the code more versatile, and in order to be able to use it for *other data sets*, two additions have to be made:

First, set up a file reader and processors for data recorded using the *Faster* acquisition system. A code to convert `.fast` files into 1D and 2D histograms with the same format already exists. One just needs to write the `doit` python scripts to run the conversion and make sure the outputs match the histogram required by the existing scripts in the analysis pipeline.

Some specific scripts and tasks may need to be written to accommodate the data recorded with the 36-pixels HPGe[10] included in *Grapheme* since 2014, but overall, it will fit in the present framework easily. This will allow the use of the analysis software for analysis of the ^{239}Pu data recorded at *JRC-Geel*, or future experiments at *NFS*, without any major hurdles.

The **second** addition is to set up the possibility to use the `resimu`[Page 95, 4, Page 95, 5, and Page 95, 6] stages once the number of neutrons and γs have been extracted from the data, to compute the cross sections (see *Full Monte Carlo data analysis to produce uncertainties and covariances*). This would allow the generation of uncertainties and covariance and correlation matrices, with a faster processing time, since the Monte-Carlo part would be only done on the last stage. (See *Use of intermediate data semi-Monte Carlo code* on that topic.)

Extension of ^{183}W analysis

Independently of the code improvement, more transitions could be added to the analysis.

A bit more raw input data could be taken in consideration. Indeed, for the development of the code, some data (representing an additional 10 to 15 % of the current size) were put aside, because, having low statistics, it required more work on Ge detectors energy calibration. By including it in the analysis, it will push up the statistics in the spectra and make the analysis more code reliable.

Using only one fission chamber in the flux determination (the UF_4 one), will also significantly improve the accuracy of the flux and reduced the parametric uncertainty (at the cost of requiring more iterations).

We already identified a few transitions that could be extracted, although with less statistics, so a finer work on fit parameters and a limited number of neutron energy ranges can be included in the analysis.

[10] Greg Henning, A. Bacquias, P. Dessagne, M. Kerveno, G. Rudolf, et al. GRAPhEME: a setup to measure (n, xn γ) reaction cross sections. 4th International Conference on Advancements in Nuclear Instrumentation Measurement Methods and their Applications (ANIMMA), Apr 2015, Lisbon, Portugal. pp.1-9, ⟨10.1109/ANIMMA.2015.7465505⟩.

In particular, for the $(\mathrm{n},\ 2\mathrm{n})$ channel, the 100.1 keV from the ^{182}W first excited state (2^+) to the ground state (*L01L00*) should be extracted. It will be partially contaminated with the 99.1 keV transition in ^{183}W, decaying from a $\frac{5}{2}^-$ state to the ground state, *which is fed by the isomeric state of this isotope*, making the contamination more tricky to handle (neutron energy selection might not be enough to cut out the $(\mathrm{n},\ \mathrm{n}'\ \gamma)$ contribution). The 260 keV γ ray mentioned *earlier* could also be investigated. The 1121 keV $(\mathrm{n},\ 2\mathrm{n}\,\gamma)$, from the 2^+ level at 1121.4 keV to the ground state in ^{182}W, is also a transition of interest in the $(\mathrm{n},\ 2\mathrm{n})$ channel.

In the $(\mathrm{n},\ \mathrm{n}')$ channel, the 84 keV transition from the $\frac{5}{2}^-$ level at 291.7 keV to the $\frac{7}{2}^-$ at 207 keV (L05L03) could also be looked at.

With these additional cross section extracted from the data, and in combination with $(\mathrm{n},\ \mathrm{n}'\ \gamma)$ and $(\mathrm{n},\ 2\mathrm{n}\ \gamma)$ cross section from the isotopically enriched ^{182}W and ^{184}W targets[Page 94, 2 and11] we will have a strong dataset to constrain the models[12,13].

Future uses

The code will be used in *future analysis* (whether it is the analysis of ^{239}Pu data, for data recorded at *NFS*). It offers full transparency on how the uncertainties are computed, which is *needed for meaningful evaluation* of experimental data. By *design*, it is intended to be flexible and reusable with different data (in terms of raw data format, number of detectors, …) so it should become a standard in our data analysis.

The use of the code, once its abilities are extended (as discussed in *Perspectives*), will streamline the analysis process and may greatly shorten the time needed to go from raw data to final cross-section result.

[11] HENNING, Greg, 2024, "Experimental (n, n' γ) cross sections for isotopes 182,184 and 186W", https://doi.org/10.57745/JRCNEJ, Recherche Data Gouv, V1

[12] "How to produce accurate inelastic cross sections from an indirect measurement method?" Maëlle Kerveno, Greg Henning, Catalin Borcea, Philippe Dessagne, Marc Dupuis, et al. EPJ N - Nuclear Sciences & Technologies, 2018, 4, pp.23. ⟨10.1051/epjn/2018020⟩. https://doi.org/10.1051/epjn/2018020

[13] "From γ emissions to $(\mathrm{n},\ x\mathrm{n})$ cross sections of interest: The role of GAINS and GRAPhEME in nuclear reaction modeling." M. Kerveno, A. Bacquias, C. Borcea, Ph. Dessagne, G. Henning, et al.. European Physical Journal A, 2015, 51 (12), pp.167. https://doi.org/10.1140/epja/i2015-15167-y

CHAPTER

SIX

OPEN SCIENCE

Open Science is a general term that encompasses all the practices, tools, and resources that support the transparent, collaborative, and reproducible conduct of research. This includes *open access* to publications and *data*, the use of *open source software*, and the development of open standards. It aims at increasing the visibility and accessibility of scientific research, foster greater collaboration among researchers, and improve the quality and impact of science.

As describe *before*, making experimental results (including uncertainties and covariances), along with a clear explanation of the analysis process, is a *key element to allow a proper inclusion of results in evaluations*. Following the methods and principles of Open Science will help to ensure that the results are published and fully available to all, along with a proper documentation to make sense of them.

We can describe Open Science by three majors *pillars* : *Open Access*, *Open Data* and *Open Source*.

Tip: I recommend the Moocs "La science ouverte", "Reproducible research: methodological principles for transparent science", and "Reproducible Research II: Practices and tools for managing computations and data" if you want to learn more on the topic. The Université de Strasbourg - Science Ouverte bibliography list on Zotero, curated by N. Cobolet is also a great source of information and references.

6.1 Open Access

Open Access (OA) means free access to research results, mostly in the form of articles. That mainly means that once a paper is published, it can be accessed without a paywall. But it goes beyond the financial aspect. *Open* refers to the ability (or not) to reuse and redistribute the content (figures, tables, ...) of a document. In that sense, the licensing on Open Access documents is less restricting than non-open ones.

6.1.1 Open Access Publishing Models

Open Access publishing can follow different paths and models. We list here the main ones, some variations can exist between listings[1] [and2].

Gold Open Access
: *Gold OA* refers to a publisher that produce only Open Access documents, with license similar to *Creative Commons*. However, such publisher usually charges for publication, directly to the authors, or their institutions. *Gold* publications are usually less frequent, or specific to a given subject (for example, conference proceedings).

Hybrid
: The *hybrid* model is a special case of *Gold OA* where the journal offers the option to the authors to pay to make their paper Open Access. Not all the articles of the publications are open. In this category, one can also shelf

[1] Author Services: Open Access Publishing Models
[2] Wikipedia, Open access

the cases when the journal chooses some articles of interest to make them Openly accessible, as a way to attract readership or promote the journal. The author may not have a say in the process (on the positive side, there is no fee to pay either).

Diamond or Platinum Open Access
This OA type is financially free for both publication and viewing, and is supported by other means. It can be that the publisher is subsidized by an institution or industry, or it can have a *freemium* economic model where one can pay or subscribe for additional features or services.

Green Open Access
The *green OA* model refers to an independent platform where authors can publish their paper in Open Access (self-archiving). Such platforms are usually maintained by institutions or government (in France: *HAL*). Free for the author, *Green OA* platforms usually don't offer any peer review of the content (but review of the *metadata* is possible). *Green* platforms are the right place to publish *postprints* (in which case, the paper has been through peer review). The *preprint* of a paper can also be published in Green Open Access, in order to get a fast publication of results.

Pirate Access
As in many fields where digital content is not free, illegal platforms have appeared to make access to restricted (paywalled) resources available to anyone. These platforms are not *publishing* platforms, they just distribute article that have been accessed in an unauthorized way. Not all papers are accessible via this channel, there is no guarantee that the content is the correct one, and it is definitely an illegal practice. However, the growth of these platforms participates in the current questioning of the commercial subscription model[3].

6.1.2 Legal and funding requirements

Access to some funding grants (such as EU or *ANR* projects) will usually come attached with a certain level of Open Access requirement (funds for the payment of Open Access fee to the publisher might be included in the grant).

Depending on the country where the author operates, they may have additional constrains and options.

In France, since the "Loi pour une république numérique"[4] the authors of public funded research can publish their *postprints* in open access after a maximum embargo period of six months (for fundamental sciences, one year for human and social science fields).

Protecting authorship's rights

In order to protect the authors' rights and go beyond the *publishing-the-postprint-in-open-access-after-a-maximum-embargo-of-6-months* possibility, a new approached is being proposed: the **Rights Retention Strategy** (RRS)[5].

The cOAlition S, a European consortium of funding and research agencies, is pushing RRS has a way for the authors to keep complete control of their publishing rights on the postprints. By asserting their rights on the manuscript from the very start of the paper submission process, the RRS allows the author to publish the postprint as soon as the paper is accepted, since the publications rights on the author's version have not been ceded to the journal.

Guidance on how to implement the Rights Retention Strategy can be found in the guide Implementing the rights retention strategy for scientific publications from Ouvrir la Science (the French comity for open science).

Note: On a smaller scale, the article's authors can first publish their figures on Open Access platforms (such as figshare) and included them in their manuscript with the proper references. That way, the figures are excluded from the publisher rights on the paper content[6].

---

[3] "The Library of Alexandra", Radiolab episode of April 7, 2023
[4] LOI n° 2016-1321 du 7 octobre 2016 pour une République numérique
[5] Rights Retention Strategy
[6] Retaining copyright for figures in academic publications to allow easy citation and reuse

6.1.3 Benefits and disadvantages of Open Access

There are clear benefits to publishing in Open Access: visibility of the research, ease of sharing the results, … Additionally, it frees the authors from the whim of the editor: if a journal wants to change its editorial line and stop publish results on a particular subfield (for example, as a way to re-center its publication on another field and attract subscriptions), authors of the said subfield may have to publish in other journals with less impact, or change the focus of their publication. Thanks to Open Access platforms, in particular in the *Green* model, authors may still publish on their own fields, without having to accommodate commercial editors' preferences. Finally, publication online marks authorship. By putting online your idea, method, results, it is now out in the world as *yours* and should not be reused by others without proper citation.

As for the disadvantages… Peer-review of papers is usually associated with some kind of fee (whether paid by the authors, the institution, the conference organizers, or a subscription model – sometimes a combination of these), and publishing directly in Open Access may *skip* the review step, making it less appealing for researchers. Still, self-published documents almost certainly have been through some kind of reviewed process before being put online (not counting the case of *postprints* deposited on a Green OA platform), but not in a *blind*, anonymous way that ensure the quality of research (however, we should note that the majority of research are honest and do their best to produce scientific results of quality anyway[7]).

As of today, papers that are not in Open Access (either their editor version or the *postprint*) are not counted in evaluating a *CNRS* researcher (Here, we note that when the *CNRS* researcher is not a primary coauthor of a publication, they may not have the ability to ensure a publication is in OA form). Finally, a significant hurdle to overcome for Open Access is the general resistance to change.

6.1.4 Conclusion on Open Access

Even thought there are still some hurdles, publishing articles in Open Access is easy enough today for publicly funded research in France: by publishing of the *postprint* after six months on the *HAL* platform, the threshold for *Green* Open Access is low and can be met by the vast majority of publications in *CNRS*.

Additionally, we should get use to publishing in Open Access other documents (working papers, notes, seminar slides, …), as those are still valuable contribution to the scientific discussion (and institutions should include these in researcher's performance evaluations).

6.2 Open Data

Open data refers to the practice of making research data freely available to others to use and reuse, with minimal restrictions. In the context of academic research, open data is becoming increasingly important as a way to promote transparency, collaboration, and reproducibility in the scientific process. By sharing their data, researchers can increase the visibility and impact of their work, facilitate collaborations, and build trust in the scientific method. Regarding this later point, Open Data can enable other researchers to replicate and build upon previous studies, leading to a more robust knowledge. Unfortunately, the production of Open Data sets is not currently included in the *metrics* used for researchers evaluation in *CNRS*.

[7] Also, websites like eBay or Uber have managed the issue of online reputation, so online scientific publication should be able to do so too.

6.2.1 The bare necessities

The minimum first step should be to publish the data related to all graphs included in a paper. In other words, if you include a figure in an article, the data points, or histograms data, … should be made available for anybody to re-plot the figure by themselves, or include your points in their studies.

Many publishers actually offer this option[1], and referees are more and more likely to request access to such material during the review process anyway.

We note that the data published as *Supplemental material* will only be *as accessible* as the paper itself (i.e. under the same *license*), and may not have a specific *DOI* (for example, the journal *Physical Review C.* only provides a Supplemental Material URL for the supplemental material), and it will just be associated with the article, not exist by itself.

6.2.2 True Open Data: the Fair principles

A true Open Data publication follows the *Fair* principles. *Fair* stands for **F**indability, **A**ccessibility, **I**nteroperability, and **R**euse[2] [and3]. Each of this item is important to ensure that the data can be found and use properly.

Findability
: The data should be easily found. This means steps have to be taken so that it can be indexed and searched correctly. This includes attribution of a *DOI*, as well as the inclusion of descriptive and relevant *metadata*.

Accessibility
: Once identified, the data should be accessible via a communication standard. If some data are small enough to be simply downloaded from a website or equivalent service, this might not be the best options for very large datasets. In the later cases, specific solutions should be implemented.

  The access to the data may require identification (the list of person accessing the data is logged) as well as registration (not everyone has access to the data).

Interoperability
: To be used, the data is most likely to need to be interpreted by a specific software, or be compared to other data set. For this purpose, the *metadata* should clearly indicate the compatible softwares, the data format, … so that it can be read correctly again in the future.

Reusability
: For reuse, the data should be attributed a specific copyright *license* that indicate who can use it, for what purpose, and if it can be redistributed as-is or in a modified way. Again, it is in the *metadata* that all this information is to be indicated.

Note: As for many things, *Fair* is more an ideal to aim for rather than a strict *pass* or *fail* stamp. Just because one can't check all the marks does not means the achievable steps should not be undertaken.

To help the researcher with Open Data publication, some website and services are available to deposit the data, such as *Zenodo* (an open registry developed by the European OpenAIRE program and operated by Cern) or *Recherche Data Gouv* (a French government backed open science repository). These repositories make the process of depositing data easier and ensure the compliance with Fair principles.

Note: Although the general goal is to have truly *Open* data sets, a published data set may still require some sort of registration, identification, or authorization before access, while still being considered *open* and following the *Fair*

[1] For example, "Supplemental Material" on Physical Review C

[2] Wilkinson, M., Dumontier, M., Aalbersberg, I. et al. The FAIR Guiding Principles for scientific data management and stewardship. Sci Data 3, 160018 (2016).

[3] Fair Principles

principles.

6.2.3 Metadata, metadata, metadata

The three key components of a good Open Data deposit are: the presence of *metadata*, extended *metadata* and accurate *metadata*. Indeed, having a binary data file is good, but if you don't know what's in it or how to read it, it's just a useless chunk of bits[4].

The term *metadata* means "all the information that is necessary to identify and make sense of Open Data files". This will most likely include the author's info, the description of the data set, ... but also reference and documentation on the format of the files.

Depending on the platform where the Open Data set is published, the required information in the metadata will vary. Below, are a few metadata fields that are most likely required in an Open Data deposit:

Data information
: A simple title and description should be given. Optionally, keywords, Subject headings and any other categorizing scheme should be used to ensure proper indexing.

Authorship
: The authors of the data, as well as eventual special funding sources, ... should be clearly identified. If possible, and in addition to name and affiliation, the author's identification numbers on platforms such as Orcid or HAL should be included.

License
: The copyright *license* under which the data is published has to be indicated.

Description
: Finally, the full description of the data (what format is it, how can it be read, how was it obtained, ...) should be given, so that anybody can access it and use it correctly.

Guidelines and *standards* for writing metadata exist[5] [and6]. The file format of the metadata (when you create it yourself, some deposit platform will help you by providing a webpage form to fill) is usually *XML*, *Json* or *yaml* (which are almost interchangeable).

The documentation of the *data format* is very important, it is the absolute requirements to be able to read the data again later. In the appendix *Data format*, we discuss the constraints and choices related to formatting data.

When many files are concerned, automatic gathering of metadata is possible with scripts. I developed one that collect file properties (type, size, checksum, ...) and output them in *yaml* format in order to prepare the *metadata.yaml* files in my repositories "Experimental (n, n' gamma) cross sections for isotopes 182,184 and 186W"[7] and Experimental gamma-ray data recorded with a LaBr3 and digital acquisition.[8]. The script can be found in a dedicated git repository: "file_metadata"[9].

[4] Here I use `bits` in the sense `0` and `1`.

[5] DataCite Metadata Schema

[6] Guide de saisie des métadonnées de citation

[7] HENNING, Greg, 2024, "Experimental (n, n' γ) cross sections for isotopes 182,184 and 186W", https://doi.org/10.57745/JRCNEJ, Recherche Data Gouv, V1

[8] HENNING, Greg, 2023, "Experimental γ-ray data recorded with a LaBr3 and digital acquisition.", https://doi.org/10.57745/D20CFB, Recherche Data Gouv, V1

[9] Henning G., `file_metadata` https://git.unistra.fr/hdr-ghenning/extras/file_metadata

Case Study: Metadata for this manuscript

The file metadata.xml (Listing 9) contains metadata in the **DataCite** format[Page 105, 5] about this document. It is shown in Listing 9.

Listing 9: Example of metadata file for this manuscript, formated in *XML*, using the DataCite scheme[Page 105, 5].

```
<?xml version="1.0" encoding="UTF-8"?>
<resource xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns="http://datacite.
↪org/schema/kernel-4" xsi:schemaLocation="http://datacite.org/schema/kernel-4 https://
↪schema.datacite.org/meta/kernel-4.4/metadata.xsd">

  <identifier identifierType="other">HDR_ghenning_2024</identifier>
  <titles>
    <title xml:lang="en">
      Applying Random Sampling methods to data analysis for uncertainty production, with␣
↪an Open source and Open science outlook.
    </title>
  </titles>

  <creators>
    <creator>
      <creatorName nameType="Personal">Henning, Greg</creatorName>
      <givenName>Greg</givenName>
      <familyName>Henning</familyName>
      <nameIdentifier nameIdentifierScheme="orcid" schemeURI="https://orcid.org/">
        https://orcid.org/0000-0003-3678-8728
      </nameIdentifier>
      <affiliation affiliationIdentifier="https://ror.org/01g3mb532"␣
↪affiliationIdentifierScheme="ror" SchemeURI="https://ror.org/">
        Institut Pluridisciplinaire Hubert Curien
      </affiliation>
      <affiliation affiliationIdentifier="https://ror.org/00pg6eq24"␣
↪affiliationIdentifierScheme="ror" SchemeURI="https://ror.org/">
        Université de Strasbourg
      </affiliation>
    </creator>
  </creators>

  <descriptions>
    <description xml:lang="en" descriptionType="Abstract">
    This HDR manuscript discusses the significance of nuclear data for energy␣
↪applications, with and emphasis on the need for improved accuracy in reaction modeling.
    It focuses on inelastic neutron scattering reactions and presents findings from␣
↪experimental studies using the Grapheme setup at the Gelina facility.
    Specifically, it presents measurements on 183W, with the end goal of constrainning␣
↪reaction models.
    Thies study uses a full Monte Carlo analysis approach to produce uncertainties and␣
↪correlation matrices, aiming for comprehensive documentation.
    Embracing Open Science principles, the manuscript details the current research␣
↪practice standards for better publication of research products.
    </description>
```

(continues on next page)

(continued from previous page)

```
  </descriptions>

  <Date dateType="Created">2024-06-27</Date>
  <Date dateType="Submitted">2024-07-01</Date>
  <publisher xml:lang="en">Université de Strasbourg</publisher>
  <publicationYear>2024</publicationYear>

  <subjects>
    <subject xml:lang="en" subjectScheme="PhySH - Physics Subject Headings" schemeURI=
↪"https://physh.org/" valueURI="https://physh.org/concepts/ef098eda-722a-4e25-8996-
↪aa19116b725a">
      Inelastic scattering reactions
    </subject>
    <subject xml:lang="en" subjectScheme="PhySH - Physics Subject Headings" schemeURI=
↪"https://physh.org/" valueURI="https://physh.org/concepts/624a78a4-af11-4f01-8bb8-
↪5a0a81905d3d">
      Nucleon induced nuclear reactions
    </subject>
    <subject xml:lang="en" subjectScheme="PhySH - Physics Subject Headings" schemeURI=
↪"https://physh.org/" valueURI="https://physh.org/concepts/14fede99-f5aa-4e3e-94be-
↪167116d8c322">
      Neutron physics
    </subject>
    <subject xml:lang="en" subjectScheme="PhySH - Physics Subject Headings" schemeURI=
↪"https://physh.org/" valueURI="https://physh.org/concepts/cd1858f1-89e4-4b0e-864a-
↪b5a4a73de5a8">
      Nuclear data analysis &amp;amp; compilation
    </subject>
    <subject xml:lang="en" subjectScheme="PhySH - Physics Subject Headings" schemeURI=
↪"https://physh.org/" valueURI="https://physh.org/concepts/eb9bd2e1-eedd-4bd0-997d-
↪58b44ffa3ebb">
      Monte Carlo methods
    </subject>
    <subject xml:lang="en" subjectScheme="LCSH - Library of Congress Subject Headings"␣
↪schemeURI="https://id.loc.gov/authorities/subjects.html" valueURI="http://id.loc.gov/
↪authorities/subjects/sh85139563">
      Uncertainty
    </subject>
    <subject xml:lang="en" subjectScheme="LCSH - Library of Congress Subject Headings"␣
↪schemeURI="https://id.loc.gov/authorities/subjects.html" valueURI="http://id.loc.gov/
↪authorities/subjects/sh2012002918">
      Measurement uncertainty (Statistics)
    </subject>
    <subject xml:lang="en" subjectScheme="LCSH - Library of Congress Subject Headings"␣
↪schemeURI="https://id.loc.gov/authorities/subjects.html" valueURI="http://id.loc.gov/
↪authorities/subjects/sh85004781">
      Analysis of covariance
    </subject>
  </subjects>

  <Contributor contributorType="ProjectLeader">
    <contributorName nameType="Personal ">Henning, Greg</contributorName>
```

(continues on next page)

(continued from previous page)

```
    <givenName>Greg</givenName>
    <familyName>Henning</familyName>
    <nameIdentifier nameIdentifierScheme="orcid" schemeURI="https://orcid.org/">
      https://orcid.org/0000-0003-3678-8728
    </nameIdentifier>
    <affiliation affiliationIdentifier="https://ror.org/01g3mb532"␣
↪affiliationIdentifierScheme="ror" SchemeURI="https://ror.org/">
      Institut Pluridisciplinaire Hubert Curien
    </affiliation>
  </Contributor>

  <language>en</language>
  <resourceType resourceTypeGeneral="Dissertation">
     Habiliatation à Diriger les Recherches / Manuscript
  </resourceType>
  <Format>application/pdf</Format>
  <Format>text/html</Format>
  <Format>text/x-rst</Format>
  <Rights rightsURI="https://creativecommons.org/licenses/by/4.0" rightsIdentifier="CC␣
↪BY 4.0">
    Creative Commons Attribution 4.0
  </Rights>
  <version>1.0</version>
</resource>
```

Here, one can see that the subjects of the document refers to two different *schemes* (the Physics Subject Headings, and Library of Congress Subject Headings) as the *PhySH* do not include subject heading for uncertainties or covariance.

Case Study: Test data recorded with an $LaBr_3$ detector

As a case study, I offer the deposit of a very simple dataset of a ^{152}Eu source γ ray spectrum recorded with an $LaBr_3$ detector connected to a *digital acquisition*, as shown in Figure 68.

The data[Page 105, 8], recorded when testing and characterizing the detector, were deposited on *Recherche Data Gouv* with the specific goal of being an example of Open Data publication. The data itself is very simple: one binary file where the raw data is recorded by the acquisition, and one additionally file of *metadata* created by the acquisition (this metadata is related only to the recorded data).

The deposit contains eleven files in total. Indeed, in addition to the raw data, documentation of the detector is given, so that any user can be sure of what has been used to record the data. A description of the conditions of recording are also given (schematic of the geometry, photographs). A *README.md* file (written in *markdown*) describe the files and the data (how it was obtained, how it can be read, ...). An example of the processed data is also given.

Finally, a metadata file is provided. In *yaml* format, it lists the deposit content, describes the data and the format of each file. For the type of file, the MIME Media type scheme was used. The important file (the actual data) has a `checksum` field that allows users to make sure the data they download is the actual file from the deposit.

Once deposited on the platform and validated by a curator, the dataset can be viewed, downloaded and cited with its own *DOI*[Page 105, 8].

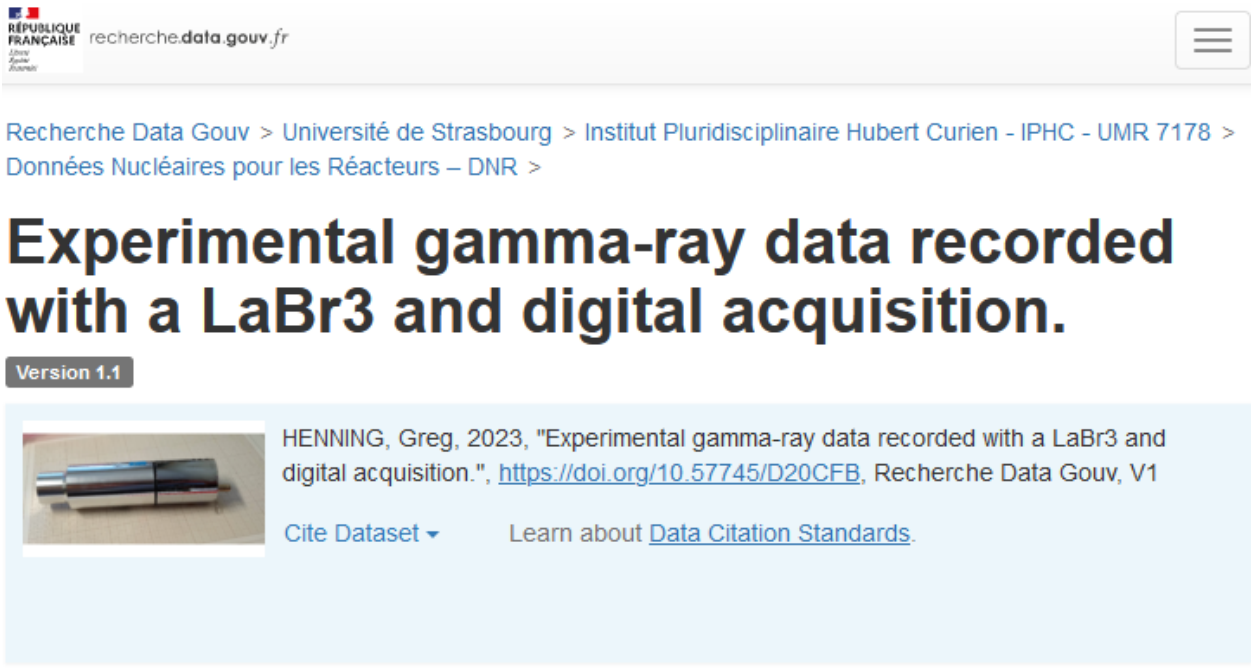


Figure 68: Screenshot of the *Recherche Data Gouv* web page for "Experimental gamma-ray data recorded with a LaBr3 and digital acquisition."[Page 105, 8]. The *DOI URL* https://doi.org/10.57745/D20CFB refers directly to the webpage.

6.2.4 DMP (i.e.) What we should have started with

DMP stands for *Data Management Plan*. As the name hints, it is the plan to manage the data. Ideally, the DMP is prepared in advance of data recording. Its goal is to cover the whole life cycle of the data: from its production to its long term storage, sharing and eventual permanent deletion. (Although the *DMP* is supposed to be written at the start of a data recording *cycle*, we mentioned it only at the end of this Section because to fully understand what is inside the management plan, we needed first to expose the different aspects of Open Data.)

The DMP will help identify who produce the data, where it will be stored, how it will be organized, for how long, who is going to have access (and how) as well as how/where/if the data will be made available as Open Data.

Writing the *DMP* is a great way to iron out many details that will be included in the *metadata* and foresee possible challenges when sharing large amount of data between different institutions. It is also a key element in maintaining *research continuity* i.e. resilience to changes in team composition.

You are not left by yourself to write your *DMP*, websites such as dmp.opidor.fr can help you create (with the help of collaborators) the *DMP* and maintain it.

Note: The *DMP* is not a fixed, *once-written-never-touched-again* file, but a *living* document that may evolve with the project. Of course, the more in-advanced prepared it is, the better, but there's no shame in going back and changing some aspects of it.

Case Study: DMP for data recorded at NFS

As a case study, I present a *DMP* for *experimental data recorded at NFS*: DMP du projet "(n, n'g) measurements at NFS" (an account is needed to access online).

6.3 Open Source

In computer programming, the principles of Open Source have been around for a long time[1]. In some ways, *Open Science* is just an extension of the concept to all research products (papers, data, ...). In the context of physics research, Open Source means providing full access to the code of the softwares used for data analysis.

It's important to realize that the analysis code is as much part of the scientific process leading to the results than any theory, experimental setup, parameters, ... However, many researchers (at least in the nuclear physics community) are self-taught in writing analysis code, and spend a significant amount of time creating their own, very application specific, softwares[2]. Having codes published in Open Source is a way to ensure code quality, strengthen the software capabilities, and allow researchers to reuse part of the code written by others, in order to spend less time and energy on re-creating software parts. More importantly, publishing analysis code is a way to demonstrate technique (and get ownership over it, *just as mentioned before*), expose your *skills* (this is important for PhD students and postdocs) and building trust in your results.

Speaking for myself, if I receive, as a referee, a paper to review that claims to use a novel or original analysis code, I will tend to ask to see it, at least to validate that it is written in a way that avoid mistakes, check that it does what it's supposed to do, ...

[1] What is the history of Open Source Software?

[2] Wilson G, Aruliah DA, Brown CT, Chue Hong NP, Davis M, et al. (2014) Best Practices for Scientific Computing. PLOS Biology 12(1): e1001745.

6.3.1 Writing Open Source code

One major thing to remember when writing Open Source code is that it's very likely that someone else will run your code[3]. This will have consequences on how the code is written, documented and packaged[Page 110, 2].

Particularly, when writing a code that has a vocation to be published in Open Source, one has to be careful of :

- Using a versioning, sharing, ... system such as *git* to keep track of the code, as well as provide a remote hosting and distribution page via a gitlab or github instance[4].
- Separating external libraries and dependencies (some of which may not be Open, or distributed under the same license), while giving clear indication on how to install them for correct functioning of your code.
- Writing the code in readable way (no funny variables names like `blabla`) and commenting, so people can understand its inner working and might even change it[5].
- Organize the directory containing the code, with parameter files, test scripts, ... in separate folders[6].
- **Test!** Before publication, and also provide a test procedure to ensure the software works as intended once installed on a different computer.
- Provide information to install, test and run the code.

Many classes (online or otherwise) can be found to learn the best practices in programming (for example[7]).

6.3.2 Source code publication

Different options exist for publishing source code:

General repositories that accept codes
: General repositories, such as *HAL* or *Zenodo* can receive softwares and store them. As such repositories are maintained by institutions, they are the best options for publishing your Open Source code. However, they are not the most ideal for retrieving and working with the code for future users.

Git repositories
: *Git* repositories are by far one of the best option for people to explore your code, download it and run it on their own. There are several institutions supported servers (gitlab.in2p3.fr, git.unistra.fr) and some private / commercial platforms (the most famous being Github.com).

Package manager repositories
: For programming languages such as *Python*, `go`[8], `rust`[9], `node.js`[10], ... public repositories exist that allow a speedy and clean installation of a library, with all the necessary checks for dependencies. The version of the code uploaded to these platform is usually a specially packaged one, which may not include all the *metadata* files, test information, ...

Thankfully, all these solutions are generally compatible between them. One can develop a `python` package and host the code on `gitlab.in2p3.fr`, publish the code on *HAL* and make it available via `Pypi.org`[11]. (That's what I have done for example for `faster` *A library to read Faster files with python*, published via HAL with the code hosted on the IN2P3 gitlab and available on pypi.org).

Following best practices from *Software Development methods* is a very good way to have a code good and ready for publication.

[3] Duh...
[4] Introduction to git (João Pedro Athayde Marcondes de André)
[5] Resources to write clear codes (**warning**: some links in that list may be dead now).
[6] Organizing directories for a computer project
[7] Software Carpentry Lessons
[8] The Go programming language
[9] Cargo, the Rust package manager
[10] Node.js, a free, open-source, cross-platform JavaScript runtime environment.
[11] The Python Package Index

Virtual Environment and Containers

To help the code run smoothly on different computers, platforms, ... it is better to rely on standard libraries, virtual environment and/or containers to embed the software in a controlled environment that is frozen in place and will stay in running state.

Cern Linux images
Cern provides a series of standard linux distributions on a dedicated webpage[12], that can be used as bases for your software development and distributions.

Python Virtual environment
Python provides a `venv` module to create a *virtual environment*, which, in concordance with *pip* will make it possible to have a standard execution setup for a python code. Python packaged version such as Anaconda[13] extends the concept of virtual environment further, but is less universal than the standard `venv`.

Containers
Finally, running "containers" with *Docker* or Singularity are the best way to provide a controlled environment that can be ported almost anywhere. (The Cern linux distributions mentioned earlier also come in the container format). For example, the current document is compiled with *Sphinx* using a *Docker* image. This ensures that it is compiled exactly the same way on a different computer[14]. The `Open Container Initiative`[15] format aims at making the container content available in a standard and open format. *Docker* can export images built into OCI tar balls. There is overall a great deal of compatibility between *Docker* and other containers runners like Singularity/Apptainer.

6.4 Distribution licenses

A document or dataset published *openly* may still come with some requirements and limits on what can be done with it. Theses are detailed in the *license* used to publish it.

Depending on the type of document, and the restrictions one wants to apply to its future (re)uses, several licenses are available.

6.4.1 Creative Commons

The *Creative Commons* licenses are the most widely used across different fields and locations. They come in different *flavors*:

CC0
The **0** version of the Creative Commons license completely opens the publication to reuse, including modifications, re-sharing, ... Publishing anything under CC0 basically puts it in the public domain.

CC-BY
With the **BY** mention, the publication can be used openly, but it is required that the original authors be mentioned. This license is the one that fit the best the *traditional* scientific practices, and can be the *default* choice when in doubt.

CC-SA
With **SA**, the open publication can be reused and reshared, be must be *shared alike*, i.e. under the same CC-SA license or an equivalent one.

[12] Linux @ CERN https://linux.web.cern.ch/
[13] Anaconda https://www.anaconda.com/
[14] This is not an exaggeration: I wrote and compiled this manuscript on two different machines.
[15] The Open Container Initiative

CC-NC
: **NC** stands for non commercial. It means any reuse or reshare is authorized as long as it is not for commercial use.

The **BY**, **SA** and **NC** can be combined and form, for example, the CC-BY-SA or CC-BY-NC licenses.

More information on https://creativecommons.org/

6.4.2 CeCIll

The CeCIll license is intended for open software publication. The name stands for "**CE**A **C**NRS **I**NRIA **l**ogiciel **l**ibre". Its current version is 2.1 and can be found in detail here: https://cecill.info/licences/Licence_CeCILL_V2.1-fr.txt.

6.4.3 European Union Public License (EUPL)

The EUPL is a European Public License, compatible with CeCILL and CC-BY-SA, intended to be a standard at the European Union level. It's current version is 1.2 and the text can be found on https://joinup.ec.europa.eu/collection/eupl/eupl-text-eupl-12

6.4.4 LICENCE OUVERTE etaLab

Finally, the *licence ouverte etaLab* is a general purpose Open License, compatible with CC-BY. Just like the EUPL, it's a license created to be fully in accordance with the French law and stay compatible with other licenses. The current version (2.0) can be found here : https://raw.githubusercontent.com/DISIC/politique-de-contribution-open-source/master/LICENSE.pdf

It is the default version for datasets published on *Recherche Data Gouv*.

6.4.5 The issue of Open Access licenses for nuclear data

There are some sticking points with distributing nuclear data for application:

- The scientist producing the data want their work to be cited, leaning toward **-BY**.
- The nuclear data will ultimately be used by industrial, so **-NC** is not a valid license variant.
- A specific license for nuclear data could include and exclusion for military uses, while making possible for the exact same data to be transferred to the relevant authorities for military purpose as long as it is done outside the Open Access license.

CHAPTER

SEVEN

MANAGEMENT METHODS FOR SOFTWARE PROGRAMMING AND DOING RESEARCH

This section will present some methods, mostly coming from computer programming, but also more general ones, that can be applied in the context of research[1].

Science often uses methods and procedures for data analysis or publications, but the daily aspect of conducting research are usually up to the people. The methods presented in this chapter can, I believe, be applied, or at least adapted, to some aspects of research, whether it is to conduct collaboration work, or ensure the proper follow-up and mentoring of students.

The use of these methods and tools can make conducting a project with an *Open Science* outlook easier, and ensure the reproducibility and continuity of research[2].

At the very least, you've been warned that this how I *function* generally.

7.1 Workflow cycle inspired from Integrated Safety Management

The *Integrated Safety Management* (ISM)[1] cycle is a process to ensure the best safety precaution when performing a work. One should obviously do their best to follow such safety oriented method, in addition, the cyclic workflow can be adapted to other activities.

The basic of ISM is to follow the steps:

```
Plan,
Execute,
Feedback and improve,
Repeat.
```

This is clearly the guideline to follow when developing a new setup (or analysis method), preparing a publication or presentation (in that case, the first *Execution* would be a rehearsal, or a review by collaborators), run an experiment (the first loops would be to test until confidence in the method and setup).

In fact, many scientists already follow (without realizing it) this cycle when preparing experiments, or conducting analysis.

[1] Not all management methods from outside the research community should be adapted. For example, the *Six Sigma* techniques, with a focus on minimizing variability, would be a disaster if applied to a research project.

[2] "*Research continuity*" is not just about keeping your analysis files organized, but following procedures, standards , and best practices, to ensure that all the information on a research project is kept. This is particularly important when short term collaborators are involved (i.e. students, postdocs, …) so that others can *pick up* their work where they left it when they move on to other positions/labs/… That also applies when your computer crashes, or your thumbdrive with a very important presentation is lost: what have you done to ensure that the work itself is resilient when the physical support fails.

[1] Integrated Safety Management (ISM)

In the context of this manuscript, we are at the junction of the *Execute* and *Feedback* steps on the analysis code: It has been executed once, the first *results* and *critics* are in, and we are planning *improvements*.

7.2 Gamification: Seeing projects as an adventure

Gamification is *the process of adding games or gamelike elements to something (such as a task) to encourage participation*[1]. In other words, it is about turning tasks into a game. Obviously, not all tasks are as pleasant as playing a good game, but elements of gameplay can be imported into a broader project in order to make the progress easier to track.

A good element to import from games like 'Dungeons & Dragons'[2] or 'The Legend of Zelda'[3] is the *Quest Log*. In these game, the heroes receive "quests" to achieve in order to unlock treasures, skills or access to new areas. The games usually have a "main" quest (kill the dragon, save the kingdom[4]), alongside many smaller "side quests" (find a lost chicken, carry a message, ...).

The main quest usually starts with a distant and broad objective (defeat the monster), that will, with the progress in the adventure, refine into smaller tasks (get the magic swords, ...). Side quests will be a way for the hero to acquire new skills and tools that will help them for the main quest (without being strictly necessary).

This is basically the same as running an experiment, or finishing a PhD. For a thesis, the main quest objective would be "Defend your thesis", which refines into "Perform experiment", "Analyze the data", "Compare to theory" and of course, the dreaded "Write the manuscript" quest.

Side quests will include "exchange with an expert in the field", "learn how to use a simulation framework", ... *rewards* will be "adding a new skill in their portfolio", "collecting a new badge", "attending a conference", "publication of a paper", ...

At a time when more and more adults play video games[5], turning some elements of a project progression into a "quest" is not a particularly difficult and have the benefits of using a familiar setting and way of measuring progress. In particular, that can be a very helpful way to present the general aspect of a thesis progression to a new student.

In fact, one does not need to use *gamelike* words, and many researchers have probably already used some kind of *gamification* when supervising students (for example "*If you finish the analysis by the end of next month, you can submit an abstract to the* `-- insert the name of prestigious conference here --`").

It's up to the manager / supervisor to find the correct goals and rewards to motivate their teammates, and the proper challenges to help them *level up*.

7.3 The Pomodoro method

The *Pomodoro* method[1] is a management method developed by F. Cirillo. It aims at working on single tasks for a limited amount of time (typically by "chunks" of 20 to 25 minutes) before moving to the next task.

The core of the method is to use a timer (such as a tomato shape kitchen timer, *Pomodoro* in Italian, hence the name) to clearly mark the work periods.

If this method is not adapted to some aspects of an experimental nuclear physicist work (data analysis usually requires a deep dive for long hours into code and data), it can be helpful when it comes to administrative tasks, for example. When small, low-value tasks are pilling up, using the pomodoro method is a good way to start checking items off your to-do list.

[1] Merriam-Webster dictionary "gamification"
[2] Dungeons & Dragons (Wikipedia)
[3] The Legend of Zelda (Wikipedia)
[4] Or the cheerleader
[5] Les français et le jeu vidéo
[1] The Pomodoro Technique

Obviously, the kitchen timer (whether in the shape of a tomato or an egg) is not a usual laboratory equipment. But many tools exist to implement your own pomodoro sessions. Microsoft Windows's Clock application has a *focus sessions* tool that automatically countdown work and break sessions. Website like pomofocus.io offer the same kind of countdowns.

7.4 Lean software programming

The Lean software programming is a general method of software development that started in the early 2000s[1]. Primarily adapted from industrial process, it relies on several general principles:

```
Eliminate waste
Amplify learning
Decide as late as possible
Deliver as fast as possible
Empower the team
Build integrity in
Optimize the whole
```

The lean philosophy grew and inspired the **Agile** software development practices. Agile development started with a Manifesto listing the values and principles to follow :

```
Individuals and interactions over processes and tools
Working software over comprehensive documentation
Customer collaboration over contract negotiation
Responding to change over following a plan
```

Additional principles were listed alongside, of which I'd like to highlight the followings (that apply best to scientific research work):

- Projects are built around motivated individuals, who should be trusted.
- Continuous attention to technical excellence and good design.
- Simplicity is essential.
- Regularly, the team reflects on how to become more effective, and adjusts accordingly.

The original goals of the Agile method is to give the "power" back to developers (people doing the work) and let them self-organize, the management being there to give general direction, and lift obstacles. I believe many physicists will find that this goal will resonate with their own aspirations.

7.4.1 Practical implementation

From the general principles of the Lean and Agile manifestos, methods and process have been derived and implemented in software development. Obviously, scientific research is not exactly like developing a software in a commercial context. However, many methods can be used and adapted in our research projects. The general idea is to avoid *micromanaging* the tasks (and the team members), and having supervisors *enabling* the work done by the team, which is generally what a Ph.D advisor should do with their student(s).

The first practical idea is to break down large tasks and goals into smaller items (just like a *main quest* is divided in *intermediate goals*), organize them by dependencies and priorities, and select the ones to work on *first* for the coming time period (see below *Sprint*). The individuals in the team then choose the tasks they will work on (and/or their shift

[1] Mary Poppendieck; Tom Poppendieck (2003). Lean Software Development: An Agile Toolkit. Addison-Wesley Professional. pp. 13–15. ISBN 978-0-321-15078-3.

schedule), and the team self-organizes to complete the list of tasks in the given time frame, after which the next set of goals will be determined.

The list of tasks to achieve is called *backlog* in the Agile methods. Important goals in the *backlog* could be related to *milestones* or *deliverables* in scientific projects. (The *backlog* is where the *quests* are listed.)

Note: During the writing of this manuscript, as well as the development of the related code, a *backlog*-like to-do list has been used.

The *unit* of time over which the work is done is call a **sprint** and may last from a few days to a few weeks. The set of goals to achieve during the sprint is *fixed*, and the team should focus only on these during that time. Hence, at the end of the sprint, a significant step forward in the project should have been reached.

During the sprint, the team organization revolves around the **daily** meeting[3] [and4] (sometime also called *stand-up*), which is intended to be limited in length (around 15 minutes). The *daily* should be an opportunity for all members to states what they did previously, what they plan to do next and what issue they face.

Important: The Sprint / daily organization is well adapted to experiments (limited time, a certain number of tasks, ...) or mentoring a student (when specific goals can be set over a *sprint* time of a week or two, with daily meetings to assess the progress).

In Agile development, the productions of the team are called *artifacts*, which relates well to what the scientific projects call *deliverable*. *Deliverables* are usually less numerous than Agile *artifacts*, but internally, teams should hold themselves to produce more deliverables (in particular in the context of *Open Science* where more than just peer-reviewed articles should be published: data and code should also be counted as *deliverables*).

7.5 The Zen of Python

A key list of principles to follow, especially when writing *(open source) code*, but not only, are the *Zen of Python*, that were listed in the *Python* Enhancement Proposal 20[1] and shown in Listing 10.

Listing 10: The *Zen of Python* by Tim Peters, a series of 20 aphorisms, only 19 of which have been written down[1].

```
Beautiful is better than ugly.
Explicit is better than implicit.
Simple is better than complex.
Complex is better than complicated.
Flat is better than nested.
Sparse is better than dense.
Readability counts.
Special cases aren't special enough to break the rules.
Although practicality beats purity.
Errors should never pass silently.
Unless explicitly silenced.
In the face of ambiguity, refuse the temptation to guess.
There should be one-- and preferably only one --obvious way to do it.
Although that way may not be obvious at first unless you're Dutch.
```

(continues on next page)

[3] Daily Stand-up: What, How, Who and When.
[4] Adopter le stand-up meeting : pourquoi & comment ?
[1] PEP 20 – The Zen of Python

(continued from previous page)

```
Now is better than never.
Although never is often better than *right* now.
If the implementation is hard to explain, it's a bad idea.
If the implementation is easy to explain, it may be a good idea.
Namespaces are one honking great idea -- let's do more of those!
```

Although written for Python code writing (in particular, number 19), the **Zen of Python** can be applied in all programming settings, and in general when conducting research. Whether it is when *choosing a data and metadata format*, *writing a DMP*, or writing an *analysis code*, following the principles is a great way to keep things organized, simple, ...

7.6 Note-taking

Note-taking is at the core of scientific work. Whether it is to put on paper what is heard during a conference or seminar, or to write new ideas from a collaboration meeting, having notes at the end is a *must*.

Warning: *Notes* are not *minutes*.

Notes are personal, usually brief and focused on key points, ideas, or actions. They are written to help to remember what was discussed and / or to share with colleagues.

Minutes are official records of what was discussed, decided, and agreed upon during a meeting or conference. They are usually taken by a designated person, and meant to be a permanent and definitive record of the meeting. They typically include a summary of the meeting's agenda, attendees, the list of actions or decisions taken. As an official document, they might be shared between the participants and corrected or modified before being validated in their final version.

Your notes should, as much as possible, reply to the 5W and 1H questions[1]: *Who*, *What*, *When*, *Where*, *Why* and *How*.

7.6.1 On paper

Taking notes on paper feels like an easy and straight forward task. But to better exploit the notes afterward, specific organization of the page can be used. One of them is the Cornell method[2]. The Cornell page organization reserved specific areas of the page for a summary, conclusions, and questions that may come after the meeting has finished.

A benefit of handwritten notes is that it is very easy to includes schematics, graphs, drawings in notes. However, handwritten notes, even when scanned and stored on a computer, can not be searched, unless some post-processing has been done.

As an example, Figure 69 shows a scanned page of notes taken about this section.

[1] The Five Ws checklist
[2] The Cornell method

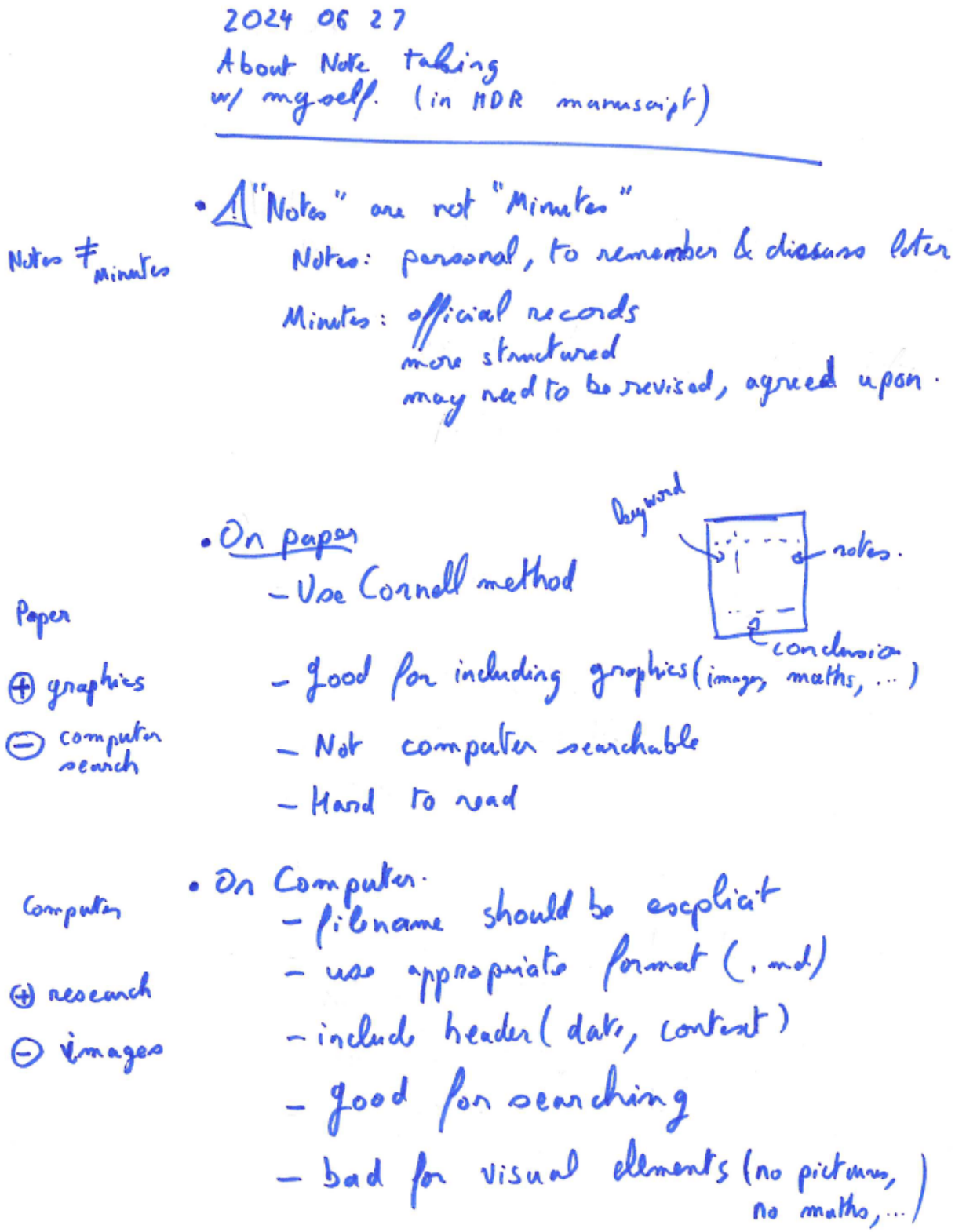


Figure 69: Handwritten notes about this section. The same content, in computer taken format, is shown in Listing 11.

7.6.2 On a computer

Since many works are done with a computer, using it for note-taking is something that will happen often. The benefit of computer notes, is that the files can be searched, organized afterward. On the other hand, inclusion of graphics, schematics, … maybe tricky.

When taking notes on a computer, one should always think about sorting, searching and finding content. File name should be explicit, include date, be in a proper folder, … The use of an appropriate format helps. *Markdown* is an obvious choice (a combination of nested bullet list and highlights is a good start for note-taking).

Tip: Syntax for *Mermaid* diagrams can be a good basis for certain note-taking (mind map, tasks sequence, …)

In your note files, include a *header* (or front matter block), to be able to remember the file content and context later.

As an example, the markdown file in Listing 11 would represent notes taken about this section.

Listing 11: Example of notes taken in Markdown. The same content, in handwritten format, is shown in Figure 69.

```
---
- date: 2024-06-27
- topic: Meeting about note-taking
- author: Myself
---

- `/!\` Notes are not minutes
    - Notes: personal, to remember and discuss with others.
    - Minutes: official record, more structured, may need to be revised/corrected before␣
↪final version.

- **on paper**:
    - use [Cornell method](https://medium.goodnotes.com/study-with-ease-the-best-way-to-
↪take-notes-2749a3e8297b)
    - Good when including schematics, graphs, math formulas, ...
    - Not computer searchable

- **on computer**:
    - file name should be explicit
    - appropriate format (like `.md`)
    - include header (date, context, ...)
    - good for searching, bad for visual elements (no images, bad at math).

<!-- end of file -->
```

Tip: Special characters like check marks, warning, … easy to represent with emoji can be written in plain text like `./` (for checkmark), `[ ]` and `[x]` (git flavored markdown style) for a checkbox, or `/!\` for a warning sign. These substitutions are recommended in plain text files, so they can be read without special rendering (in particular in terminals).

Tip: Adding an `end of file` comment at the end of a file (not just notes actually, any file) is a great way to ensures that the file is not truncated.

CHAPTER

EIGHT

CONCLUSIONS

We discussed here the *context* of nuclear data for applications, in particular for the production of energy. To overcome today's challenges (safety, sustainability, waste management), new design, or fuel cycle are under study. The development of these new concepts that will use different isotopes, or faster neutrons, relies essentially on numerical simulations. These simulations use, as inputs, evaluated databases that list nuclear reaction cross section, angular distribution, particle spectra, ... The *evaluations* are built on experimental data and theoretical models. However, for the isotopes and neutron energy ranges consider for next generation of reactors, the current state of the evaluations is *not enough to ensure an accurate simulation of reactors parameters*. This is because the models and experimental data *still present large uncertainties for these isotopes* and neutron energy. The necessary improvement of nuclear data evaluation, in order to meet the required accuracy in reactor simulations, will therefore be achieved by a combination of new, precise, experimental measurements, and improvement of reaction models (through theoretical refinement, and/or by constraining the model's parameters with experiments).

The *DNR* group at *IPHC* has decided to focus on *inelastic neutron scattering*. These reactions are important in a reactor, as they modify the neutron energy, and change the neutron population and create new isotopes (in the case of $(\mathrm{n},\ x\mathrm{n})$ with $x \leq 2$). The impact of these reactions on reactor parameters (power, reactivity, ...) has been demonstrated in *sensitivity studies*. The reaction models describing the inelastic scattering can benefit from new, precise measurements, as *only a few experimental data is currently available* to adjust the models.

With the *Grapheme* setup, installed at the *Gelina* facility in *JRC-Geel*, The $(\mathrm{n},\ \mathrm{n}')$ and $(\mathrm{n},\ 2\mathrm{n})$ reactions were studied on the isotopes of $^{\mathrm{nat}}$Zr, $^{\mathrm{nat,182,183,184,186}}$W, ^{233,235,238}U, and ^{232}Th ; with other isotopes foresee in the near future. The experimental $(\mathrm{n},\ x\mathrm{n}\ \gamma)$ cross section obtained with *Grapheme* are compared to model predictions and used to improve the reaction codes and input parameters (level density, γ strength functions, ...).

The *tungsten isotopes* are a good *playing field* for measurements and confrontation with models. The $(\mathrm{n},\ \mathrm{n}'\ \gamma)$ cross section for *even-even isotopes* have already been investigated. Adding measurements for ^{183}W is a nice complement to the set, as it will allow the study of the unpaired nucleon, as well as *thread* the values from ^{182} to ^{184}W. The *current knowledge* of $(\mathrm{n},\ x\mathrm{n})$ reaction on this isotope is limited, and our experimental values will fill a gap in the sparse data on $(\mathrm{n},\ \mathrm{n}')$ reaction and provide a valuable data set to constrain the models and parameters for the reactions code.

A very important thing when producing the $(\mathrm{n},\ x\mathrm{n}\ \gamma)$ cross sections is to produce precise results (ideally, the relative uncertainty should be around 5 % or less). The *uncertainties* associated with the cross sections should be fully documented (what is their sources, components, shape, ...) and ideally come with *covariance/correlation matrices*.

There are several ways to produce uncertainties and covariance matrices. One is relying on the *Monte Carlo* (i.e. random sampling method). With *a full Monte Carlo* (MC) analysis, the analysis parameters are sampled randomly according to their given distribution, and the analysis is performed many times, with the results being collected. At the end of the iterations, a central value (*mean*) and parametric uncertainty (*standard deviation*) are computed to produce the result. The covariance can easily be obtained from the collection of iteration results.

A *new analysis code* for *full MC analysis* has been developed, and applied to the ^{183}W *data set* recorded with *Grapheme*. The objective of this code is to rely on a *pipeline* of stages and *tasks* in order to automatize the analysis while reducing reprocessing files that did not change. The analysis is written so that it can be easily debugged, and deployed on different platforms. Finally, the ultimate goal is to have a flexible code base, that will be adapted to other data sets and *experiments in the future* (one thing that was not possible with a previous Monte Carlo code). As it is written now,

the code can easily be adapted to any number of detectors, at different angles (not just the 110° and 150° combination used currently).

The *detailed workflow* of the analysis has been presented, in order to completely explain how the values and their uncertainties are obtained.

The preliminary *results* show compatibility with *previous extraction from the same data set*, and a limited agreement with models, as had been seen in previous studies (*even-even W isotopes*, ^{238}U). For the first time, the correlation matrices have been obtained directly from the full Monte Carlo analysis for all studied γ rays. The correlation factors show, as expected, the strong correlation between points, driven by the target mass and HPGe detection efficiencies parameters. It could be enlightening to plot the correlation across all transitions together, rather than transition by transition.

Although in the current format only the angle integrated $(\mathrm{n},\ x\mathrm{n}\ \gamma)$ cross sections were produced as results of the full MC analysis, it is absolutely possible to extract other quantities from the analysis: angular cross section, correlation matrices between transitions (rather than correlation strictly within a transition).

The current way of presenting the results, with *side-by-side* experimental measurements plotted against model calculations, is efficiency in its simplicity, but somehow *dry* and lacks an *overview* aspect. Indeed, even in the data from one transition, there are different components to the cross section (direct peak, feeding from the continuum, ...) and a global comparison may not be the best option: what if the shapes of the measured values is well reproduced by the models, but not the amplitude? What if the amplitude match, but the shape is overall different? Following the latest work of M. Dupuis, one could look at how each transition constrains the different model parameters.

The code can still be *improved* in order to refine the results (so far preliminary) and being able to deploy it for other data sets. A list of *potential improvements* has been established, so that clear deadlines and delivery dates can be set for the improvements. It is already foreseen that the analysis framework will be used (at least in part) for the analysis of ^{239}Pu data recorded with *Grapheme* (the data set will include more HPGe detectors than in the ^{183}W one, with among the extra channels, a 36-pixels detector), as well as experimental data that will be recorded at *NFS*.

The ^{183}W *results* will complete the even-even isotopes ones. We expect a quick release of ^{184}W $(\mathrm{n},\ 2\mathrm{n}\ \gamma)$ cross section measurements after that. The whole experimental data will be published rapidly and made available to the public, as well as described in a *data paper*. The theoretical interpretation of the trans-isotopes $^{182,\ 183,\ \text{and}\ 184}$W $(\mathrm{n},\ \mathrm{n}'\ \gamma)$ and $(\mathrm{n},\ 2\mathrm{n}\ \gamma)$ will follow in a scientific article.

To properly take into account the experimental data in the evaluation, *the most complete description possible of the uncertainties and the way they were estimated is needed*. That is why detailing the way the data was analyzed, which parameters were sampled and how, what convergence criteria was used, ... is very important. The best *framework* to publish the details of an experimental analysis is given by *Open Science*. Through *Open Access*, the paper discussing the methods and results can be made widely available. *Open Data* is how the results files, but also, possibly, intermediate analysis files (like angular cross sections, flux, ...) can be distributed, so that some kind of data reanalysis, or complementary work can be performed. And, with publication of the analysis code in *Open Source*, the whole process of analysis can be studied, redone (if new information come to light that warrant a reanalysis) or simply used in other settings. In particular in the case of uncertainty and covariance, where the process that leads to the final uncertainties is important, having the complete analysis code to *play* with is a significant step in helping the final user go back to the source of the final result (literally).

Therefore, not only for ^{183}W results, but also in *the future* (in particular with ^{239}Pu data, or experiments at *NFS*), the code will be *our* primary tool for data analysis. And the philosophy behind it (detail description of analysis, *Open Source* code, ...) *our* driving principles.

Working with an *Open Science* outlook, in particular in the context of collegial research (involving researchers, students, interns, ...) requires, to be easy and successful, following a few *procedures*. Some of these procedures have been presented here. They, of course, should not be followed strictly. But they present a framework, a pattern, for the research supervisor to conduct research with their team. Combined with the principles of *Open Science*, these *time* and *project management* tips are also really helpful to maintain research continuity, *i.e.* ensuring the continuation of the research work when members of the team move to other labs or field, a student finishes his PhD., a hard drive crashes, ...

This manuscript and the work associated, following the *Open Science* principles, is available in its entirety in *Open Access*: the text, its sources (text in *reStructure Text*, figures data and generation scripts) the compilation script to generate the *HTML* pages or *PDF*, some associated documents (an example of *DMP*) or scripts.

TL;DR[1]

- There is a need for improved nuclear data evaluations for the development of nuclear applications.
- These improvements require new measurements and better theoretical descriptions by models.
- The group *DNR* at *IPHC* focuses its work on $(\mathrm{n},\ x\mathrm{n})$ reaction, with the measurements of $(\mathrm{n},\ x\mathrm{n}\,\gamma)$ cross sections.
- In these document, I present results for ^{183}W, following *past results on even-even isotopes*.
- A *full Monte Carlo analysis code* was developed to analyze the data taken with the *Grapheme* setup at *Gelina*.
- The analysis code is *extensively detailed*, in order to explain how the data is produced, *allowing for meaningful evaluation*.
- Embracing the principles of *Open Science*, the code will be distributed in *Open Source*, and the results made available as *Open Data*. Such openness is the best way to provide the transparency needed for proper exploitation of our results.
- This *Open Science* approach, as well as *methods* to help conform to it, will be our guiding principle in the *future*.

[1] "Too long; didn't read", a.k.a The take away message.

CHAPTER

NINE

APPENDIX

9.1 Experimental (n, n' γ) and (n, 2n γ) cross sections

Here we give the raw values from the analysis.

9.1.1 Talys structures

For reference, and to clarify the labelling of the transitions, here are the level scheme considered by *Talys* in the calculations:

^{182}W

Listing 12: 10 levels in the ^{182}W level scheme used by *Talys* (excerpt from the Talys output).

```
Discrete levels of Z= 74 N=108 (182W )

Number  Energy Spin Parity  Branching Ratio (%) Lifetime(sec) Assignment        ENSDF

  0     0.0000  0.0   +                                                          0+
  1     0.1001  2.0   +                                                          2+
                            --->  0  100.0000
  2     0.3294  4.0   +                                                          4+
                            --->  1  100.0000
  3     0.6804  6.0   +                                                          6+
                            --->  2  100.0000
  4     1.1358  0.0   +                                                          0+
                            --->  1  100.0000
  5     1.1443  8.0   +                                                          8+
                            --->  3  100.0000
  6     1.2214  2.0   +                                                          2+
                            --->  2    0.0921
                            --->  1   56.3688
                            --->  0   43.5391
  7     1.2574  2.0   +                                                          2+
                            --->  4    0.2463
                            --->  2   22.1486
                            --->  1   23.0585
                            --->  0   54.5466
```

(continues on next page)

(continued from previous page)

```
  8     1.2891  2.0   -                                                    2-
                           --->  7    3.1799
                           --->  6   71.4681
                           --->  2    0.4897
                           --->  1   22.9394
                           --->  0    1.9229
  9     1.3311  3.0   +                                                    3+
                           --->  2   15.2400
                           --->  1   84.7600
 10     1.3738  3.0   -                                                    3-
                           --->  9    1.3960
                           --->  8   69.5906
                           --->  7    1.6850
                           --->  6   23.9202
                           --->  2    0.7282
                           --->  1    2.0030
                           --->  0    0.6769
```

^{183}W

Listing 13: 15 first levels in the ^{183}W level scheme used by *Talys* (excerpt from the Talys output).

```
Discrete levels of Z= 74 N=109 (183W )

Number  Energy Spin Parity  Branching Ratio (%) Lifetime(sec) Assignment        ENSDF

  0     0.0000  0.5   -                                                      1/2-
  1     0.0465  1.5   -                                                      3/2-
                           --->  0  100.0000
  2     0.0991  2.5   -                                                      5/2-
                           --->  1   54.4400
                           --->  0   45.5600
  3     0.2070  3.5   -                                                      7/2-
                           --->  2   91.4500
                           --->  1    8.5500
  4     0.2088  1.5   -                                                      3/2-
                           --->  2   19.9000
                           --->  1   73.4800
                           --->  0    6.6200
  5     0.2917  2.5   -                                                      5/2-
                           --->  4   17.7804
                           --->  3   56.8111
                           --->  2    2.6731
                           --->  1    2.7451
                           --->  0   19.9904
  6     0.3089  4.5   -                                                      9/2-
                           --->  3   23.2200
                           --->  2   76.7800
  7     0.3608  5.5   +                        5.200E+00                    11/2+
                           --->  3  100.0000
```

(continues on next page)

(continued from previous page)

```
  8     0.4121  3.5   -                                                    7/2-
                          --->  6   17.9696
                          --->  5    2.6559
                          --->  4    6.0519
                          --->  3   17.6296
                          --->  2   48.9890
                          --->  1    6.7039
  9     0.4531  3.5   -                                                    7/2-
                          --->  8    6.3370
                          --->  6    7.1260
                          --->  5   21.3000
                          --->  4   10.5500
                          --->  3   40.4500
                          --->  2   13.6500
                          --->  1    0.5870
 10     0.4754  5.5   -                                                   11/2-
                          --->  6   58.1000
                          --->  3   41.9000
 11     0.4870  6.5   +                                                  (13/2)+
                          ---> 10  100.0000                 B
 12     0.5330  1.5   +                                     J           (1/2,3/2)
                          --->  2   50.0000                 B
                          --->  0   50.0000                 B
 13     0.5511  4.5   -                                                  (9/2)-
                          --->  5   49.4200
                          --->  3   27.2200
                          --->  2   23.3600
 14     0.5953  4.5   -                                                  (9/2)-
                          --->  9   98.7261
                          --->  6    1.2739
 15     0.6228  4.5   +                                                    9/2+
                          --->  7  100.0000
```

9.1.2 Inelastic reactions

Following are the results of the transitions studied in the *inelastic channel*.

209 keV

Listing 14: Raw result file for the experimental cross section for the 209.9 keV γ ray in the $(n,\ n')$ reaction on ^{183}W. As shown in Figure 48.

```
#
# title: 183W(n, n' gamma[209.8 keV - 183L06L02])
# date: '2024-03-18'
# author: Greg Henning
# reaction:
#   proj: N
#   target: 74-W-183
#   code: 74-W-183(N, INL)74-W-183,PAR,SIG,G
```

(continues on next page)

(continued from previous page)

```
# transition:
#   emitter: 183W
#   gamma energy: 209.8
#   energy unit: keV
#   label: 183L06L02
#   threshold energy: 308.0
#   icc: 2.620E-01
#   initial level:
#     excitation energy: 308.9
#     spin parity: 9/2 -
#   final level:
#     excitation energy: 99.1
#     spin parity: 5/2 -
# columns:
# - title: Neutron energy window, middle of range
#   unit: keV
# - title: gamma cross section
#   unit: barns
# - title: neutron energy window low range
#   unit: keV
# - title: Uncertainty on neutron energy window low range
#   unit: keV
# - title: neutron energy window high range
#   unit: keV
# - title: Uncertainty on neutron energy window high range
#   unit: keV
# - title: cross section parameter uncertainty
#   unit: barns
# - title: cross section intrinsic uncertainty
#   unit: barns
# - title: cross section total combined uncertainty
#   unit: barns
# number of lines: 17
#
      550.00      0.01553       300.00          1.5       800.00          6.5      0.
↪00232      0.01031      0.01057
      900.00      0.04959       800.00          6.5      1000.00          9.1      0.
↪00833      0.01968      0.02137
     1250.00      0.06753      1000.00          9.1      1500.00         16.8      0.
↪00832      0.01533      0.01744
     1750.00      0.08699      1500.00         16.8      2000.00         25.8      0.
↪00963      0.01902      0.02132
     2250.00      0.10214      2000.00         25.8      2500.00         36.1      0.
↪01455      0.02068      0.02529
     2750.00      0.11510      2500.00         36.1      3000.00         47.4      0.
↪01664      0.02963      0.03398
     3250.00      0.13339      3000.00         47.4      3500.00         59.8      0.
↪01698      0.02752      0.03234
     3750.00      0.13841      3500.00         59.8      4000.00         73.0      0.
↪02237      0.03592      0.04232
     4250.00      0.14762      4000.00         73.0      4500.00         87.1      0.
↪01694      0.05019      0.05297
```

(continues on next page)

(continued from previous page)

```
     4750.00      0.15275      4500.00         87.1      5000.00        102.1      0.
↪02132      0.07984      0.08264
     5500.00      0.16376      5000.00        102.1      6000.00        134.2      0.
↪01904      0.06222      0.06507
     6500.00      0.15560      6000.00        134.2      7000.00        169.1      0.
↪02016      0.04888      0.05287
     7500.00      0.12720      7000.00        169.1      8000.00        206.6      0.
↪01439      0.05432      0.05619
     8500.00      0.08978      8000.00        206.6      9000.00        246.5      0.
↪01597      0.06070      0.06277
    10500.00      0.05771      9000.00        246.5     12000.00        379.5      0.
↪01807      0.05623      0.05906
    14000.00      0.03762     12000.00        379.5     16000.00        584.2      0.
↪00613      0.09883      0.09902
    20500.00      0.02702     16000.00        584.2     25000.00       1141.1      0.
↪00826      0.09485      0.09521
# end of file: True
```

259 keV

Listing 15: Raw result file for the experimental cross section for the 259.5 keV γ ray in the $(n,\ n')$ reaction on ^{183}W. As shown in Figure 50.

```
#
# title: 183W(n, n' gamma[259.48 keV - 183L13L05])
# date: '2024-03-18'
# author: Greg Henning
# reaction:
#   proj: N
#   target: 74-W-183
#   code: 74-W-183(N, INL)74-W-183,PAR,SIG,G
# transition:
#   emitter: 183W
#   gamma energy: 259.48
#   energy unit: keV
#   label: 183L13L05
#   threshold energy: 551.2
#   icc: 1.325E-01
#   initial level:
#     excitation energy: 551.2
#     spin parity: 9/2 -
#   final level:
#     excitation energy: 291.7
#     spin parity: 5/2 -
# columns:
# - title: Neutron energy window, middle of range
#   unit: keV
# - title: gamma cross section
#   unit: barns
# - title: neutron energy window low range
#   unit: keV
```

(continues on next page)

(continued from previous page)

```
# - title: Uncertainty on neutron energy window low range
#   unit: keV
# - title: neutron energy window high range
#   unit: keV
# - title: Uncertainty on neutron energy window high range
#   unit: keV
# - title: cross section parameter uncertainty
#   unit: barns
# - title: cross section intrinsic uncertainty
#   unit: barns
# - title: cross section total combined uncertainty
#   unit: barns
# number of lines: 12
#
      650.00      0.00153       500.00          3.2       800.00          6.5      0.
↪00127      0.00076      0.00148
      900.00      0.00735       800.00          6.5      1000.00          9.1      0.
↪00262      0.00131      0.00293
     1250.00      0.01151      1000.00          9.1      1500.00         16.8      0.
↪00144      0.00090      0.00170
     1750.00      0.01581      1500.00         16.8      2000.00         25.8      0.
↪00179      0.00131      0.00222
     2250.00      0.01592      2000.00         25.8      2500.00         36.1      0.
↪00199      0.00162      0.00257
     2750.00      0.01875      2500.00         36.1      3000.00         47.4      0.
↪00254      0.00228      0.00341
     3250.00      0.01985      3000.00         47.4      3500.00         59.8      0.
↪00297      0.00257      0.00393
     3750.00      0.01972      3500.00         59.8      4000.00         73.0      0.
↪00282      0.00324      0.00430
     4250.00      0.02219      4000.00         73.0      4500.00         87.1      0.
↪00297      0.00425      0.00518
     4750.00      0.02168      4500.00         87.1      5000.00        102.1      0.
↪00322      0.00423      0.00532
     5500.00      0.02200      5000.00        102.1      6000.00        134.2      0.
↪00351      0.00318      0.00474
     8000.00      0.01523      6000.00        134.2     10000.00        288.7      0.
↪00231      0.00276      0.00360
# end of file: True
```

268 keV

Listing 16: Raw result file for the experimental cross section for the 268.1 keV γ ray in the (n, n') reaction on ^{183}W. As shown in Figure 52.

```
#
# title: 183W(n, n' gamma[268.1 keV - 183L10L03])
# date: '2024-03-18'
# author: Greg Henning
# reaction:
#   proj: N
```

(continues on next page)

(continued from previous page)

```
#   target: 74-W-183
#   code: 74-W-183(N, INL)74-W-183,PAR,SIG,G
# transition:
#   emitter: 183W
#   gamma energy: 268.1
#   energy unit: keV
#   label: 183L10L03
#   threshold energy: 475.0
#   icc: 1.197E-01
#   initial level:
#     excitation energy: 475.1
#     spin parity: 11/2 -
#   final level:
#     excitation energy: 207.0
#     spin parity: 7/2 -
# columns:
# - title: Neutron energy window, middle of range
#   unit: keV
# - title: gamma cross section
#   unit: barns
# - title: neutron energy window low range
#   unit: keV
# - title: Uncertainty on neutron energy window low range
#   unit: keV
# - title: neutron energy window high range
#   unit: keV
# - title: Uncertainty on neutron energy window high range
#   unit: keV
# - title: cross section parameter uncertainty
#   unit: barns
# - title: cross section intrinsic uncertainty
#   unit: barns
# - title: cross section total combined uncertainty
#   unit: barns
# number of lines: 16
#
      650.00      0.00125       500.00          3.2       800.00          6.5      0.
↪00036      0.00053      0.00064
      900.00      0.00504       800.00          6.5      1000.00          9.1      0.
↪00153      0.00095      0.00180
     1250.00      0.00909      1000.00          9.1      1500.00         16.8      0.
↪00137      0.00083      0.00160
     1750.00      0.01506      1500.00         16.8      2000.00         25.8      0.
↪00246      0.00120      0.00274
     2250.00      0.01914      2000.00         25.8      2500.00         36.1      0.
↪00326      0.00166      0.00366
     2750.00      0.02457      2500.00         36.1      3000.00         47.4      0.
↪00334      0.00238      0.00410
     3250.00      0.02644      3000.00         47.4      3500.00         59.8      0.
↪00369      0.00296      0.00473
     3750.00      0.03188      3500.00         59.8      4000.00         73.0      0.
↪00496      0.00349      0.00606
```

(continues on next page)

(continued from previous page)

```
     4250.00      0.03614      4000.00         73.0      4500.00         87.1      0.
↪00512      0.00408      0.00655
     4750.00      0.03627      4500.00         87.1      5000.00        102.1      0.
↪00553      0.00412      0.00690
     5500.00      0.04236      5000.00        102.1      6000.00        134.2      0.
↪00553      0.00364      0.00662
     6500.00      0.04105      6000.00        134.2      7000.00        169.1      0.
↪00606      0.00418      0.00736
     7500.00      0.03734      7000.00        169.1      8000.00        206.6      0.
↪00635      0.00481      0.00797
     8500.00      0.02342      8000.00        206.6      9000.00        246.5      0.
↪00540      0.00691      0.00877
    12000.00      0.00987      9000.00        246.5     15000.00        530.3      0.
↪00187      0.00587      0.00616
    20000.00      0.00923     15000.00        530.3     25000.00       1141.1      0.
↪00453      0.01305      0.01381
# end of file: True
```

291 keV

Listing 17: Raw result file for the experimental cross section for the 291.7 keV γ ray in the (n, n') reaction on ^{183}W. As shown in Figure 54.

```
#
# title: 183W(n, n' gamma[291.7 keV - 183L05L00])
# date: '2024-03-18'
# author: Greg Henning
# reaction:
#   proj: N
#   target: 74-W-183
#   code: 74-W-183(N, INL)74-W-183,PAR,SIG,G
# transition:
#   emitter: 183W
#   gamma energy: 291.7
#   energy unit: keV
#   label: 183L05L00
#   threshold energy: 291.7
#   icc: 9.240E-02
#   initial level:
#     excitation energy: 291.7
#     spin parity: 5/2 -
#   final level:
#     excitation energy: 0.0
#     spin parity: 1/2 -
# columns:
# - title: Neutron energy window, middle of range
#   unit: keV
# - title: gamma cross section
#   unit: barns
# - title: neutron energy window low range
#   unit: keV
```

(continues on next page)

(continued from previous page)

```
# - title: Uncertainty on neutron energy window low range
#   unit: keV
# - title: neutron energy window high range
#   unit: keV
# - title: Uncertainty on neutron energy window high range
#   unit: keV
# - title: cross section parameter uncertainty
#   unit: barns
# - title: cross section intrinsic uncertainty
#   unit: barns
# - title: cross section total combined uncertainty
#   unit: barns
# number of lines: 14
#
      450.00      0.02377       300.00          1.5       600.00          4.2      0.
↪00394      0.00340      0.00520
      750.00      0.04488       600.00          4.2       900.00          7.8      0.
↪00598      0.00348      0.00692
     1050.00      0.05402       900.00          7.8      1200.00         12.0      0.
↪00966      0.00328      0.01020
     1350.00      0.06419      1200.00         12.0      1500.00         16.8      0.
↪00754      0.00682      0.01017
     1750.00      0.05808      1500.00         16.8      2000.00         25.8      0.
↪01072      0.00481      0.01175
     2250.00      0.05819      2000.00         25.8      2500.00         36.1      0.
↪01005      0.00644      0.01194
     2750.00      0.06040      2500.00         36.1      3000.00         47.4      0.
↪00800      0.01065      0.01332
     3500.00      0.05571      3000.00         47.4      4000.00         73.0      0.
↪00826      0.00953      0.01261
     4500.00      0.05360      4000.00         73.0      5000.00        102.1      0.
↪01082      0.01246      0.01650
     5500.00      0.05216      5000.00        102.1      6000.00        134.2      0.
↪00702      0.01540      0.01692
     6500.00      0.04723      6000.00        134.2      7000.00        169.1      0.
↪00641      0.01916      0.02020
     7500.00      0.03382      7000.00        169.1      8000.00        206.6      0.
↪00619      0.01854      0.01955
     8500.00      0.02175      8000.00        206.6      9000.00        246.5      0.
↪00365      0.02760      0.02784
    12500.00      0.01234      9000.00        246.5     16000.00        584.2      0.
↪00306      0.02497      0.02516
# end of file: True
```

313 keV

Listing 18: Raw result file for the experimental cross section for the 313.0 keV γ ray in the $(n,\ n')$ reaction on ^{183}W. As shown in Figure 56.

```
#
# title: 183W(n, n' gamma[313.0 keV - 183L08L02])
```

(continues on next page)

(continued from previous page)

```
# date: '2024-03-18'
# author: Greg Henning
# reaction:
#   proj: N
#   target: 74-W-183
#   code: 74-W-183(N, INL)74-W-183,PAR,SIG,G
# transition:
#   emitter: 183W
#   gamma energy: 313.0
#   energy unit: keV
#   label: 183L08L02
#   threshold energy: 412.1
#   icc: 1.900E-01
#   initial level:
#     excitation energy: 412.1
#     spin parity: 7/2 -
#   final level:
#     excitation energy: 99.1
#     spin parity: 5/2 -
# columns:
# - title: Neutron energy window, middle of range
#   unit: keV
# - title: gamma cross section
#   unit: barns
# - title: neutron energy window low range
#   unit: keV
# - title: Uncertainty on neutron energy window low range
#   unit: keV
# - title: neutron energy window high range
#   unit: keV
# - title: Uncertainty on neutron energy window high range
#   unit: keV
# - title: cross section parameter uncertainty
#   unit: barns
# - title: cross section intrinsic uncertainty
#   unit: barns
# - title: cross section total combined uncertainty
#   unit: barns
# number of lines: 17
#
      600.00      0.02672       400.00          2.3       800.00          6.5      0.
↪00442      0.00269      0.00517
      950.00      0.07293       800.00          6.5      1100.00         10.5      0.
↪01030      0.00485      0.01138
     1300.00      0.10667      1100.00         10.5      1500.00         16.8      0.
↪01313      0.00535      0.01418
     1750.00      0.14149      1500.00         16.8      2000.00         25.8      0.
↪01758      0.00621      0.01864
     2250.00      0.14873      2000.00         25.8      2500.00         36.1      0.
↪02193      0.00674      0.02294
     2750.00      0.14967      2500.00         36.1      3000.00         47.4      0.
↪02165      0.01148      0.02451
```

(continues on next page)

(continued from previous page)

```
     3250.00      0.14811      3000.00         47.4      3500.00         59.8      0.
↪02049      0.01342      0.02449
     3750.00      0.15218      3500.00         59.8      4000.00         73.0      0.
↪01890      0.01466      0.02392
     4250.00      0.15111      4000.00         73.0      4500.00         87.1      0.
↪02046      0.01548      0.02566
     4750.00      0.15122      4500.00         87.1      5000.00        102.1      0.
↪02050      0.01627      0.02617
     5500.00      0.14876      5000.00        102.1      6000.00        134.2      0.
↪01913      0.01559      0.02468
     6500.00      0.13667      6000.00        134.2      7000.00        169.1      0.
↪01960      0.01484      0.02458
     7500.00      0.09646      7000.00        169.1      8000.00        206.6      0.
↪01511      0.02185      0.02657
     8500.00      0.06595      8000.00        206.6      9000.00        246.5      0.
↪01354      0.02488      0.02833
     9500.00      0.04555      9000.00        246.5     10000.00        288.7      0.
↪01003      0.03880      0.04008
    11000.00      0.03796     10000.00        288.7     12000.00        379.5      0.
↪00933      0.03293      0.03423
    16000.00      0.02075     12000.00        379.5     20000.00        816.5      0.
↪00506      0.03126      0.03167
# end of file: True
```

354 keV

Listing 19: Raw result file for the experimental cross section for the 353.9 keV γ ray in the $(n,\ n')$ reaction on ^{183}W. As shown in Figure 58.

```
#
# title: 183W(n, n' gamma[353.99 keV - 183L09L02])
# date: '2024-03-18'
# author: Greg Henning
# reaction:
#   proj: N
#   target: 74-W-183
#   code: 74-W-183(N, INL)74-W-183,PAR,SIG,G
# transition:
#   emitter: 183W
#   gamma energy: 353.99
#   energy unit: keV
#   label: 183L09L02
#   threshold energy: 453.1
#   icc: 1.373E-01
#   initial level:
#     excitation energy: 453.1
#     spin parity: 7/2 -
#   final level:
#     excitation energy: 99.1
#     spin parity: 5/2 -
# columns:
```

(continues on next page)

(continued from previous page)

```
# - title: Neutron energy window, middle of range
#   unit: keV
# - title: gamma cross section
#   unit: barns
# - title: neutron energy window low range
#   unit: keV
# - title: Uncertainty on neutron energy window low range
#   unit: keV
# - title: neutron energy window high range
#   unit: keV
# - title: Uncertainty on neutron energy window high range
#   unit: keV
# - title: cross section parameter uncertainty
#   unit: barns
# - title: cross section intrinsic uncertainty
#   unit: barns
# - title: cross section total combined uncertainty
#   unit: barns
# number of lines: 6
#
      700.00      0.01220       400.00          2.3      1000.00          9.1      0.
↪00395      0.00172      0.00431
     1500.00      0.02088      1000.00          9.1      2000.00         25.8      0.
↪00259      0.00184      0.00318
     2500.00      0.02225      2000.00         25.8      3000.00         47.4      0.
↪00353      0.00340      0.00490
     3500.00      0.02401      3000.00         47.4      4000.00         73.0      0.
↪00317      0.00596      0.00675
     5000.00      0.02334      4000.00         73.0      6000.00        134.2      0.
↪00365  31865.90662  31865.90662
     8500.00      0.00594      6000.00        134.2     11000.00        333.0      0.
↪00935  11139.71201  11139.71201
# end of file: True
```

9.1.3 (n, 2n) results

Then, we present the results of the transitions studied in the $(n,\ 2n)$ channel (in section *(n, 2n) results*).

229 keV

Listing 20: Raw result file for the experimental cross section for the 229 keV γ ray in the $(\mathrm{n},\ 2\mathrm{n})$ reaction on ^{183}W (transition in the ^{182}W isotope.). As shown in Figure 61.

```
#
# title: 183W(n, n' gamma[229.3 keV - 182L02L01])
# date: '2024-03-18'
# author: Greg Henning
# reaction:
#   proj: N
```

(continues on next page)

(continued from previous page)

```
#   target: 74-W-183
#   code: 74-W-183(N, 2N)74-W-182,PAR,SIG,G
# transition:
#   emitter: 182W
#   gamma energy: 229.3
#   energy unit: keV
#   label: 182L02L01
#   threshold energy: 6519.0
#   icc: 1.960E-01
#   initial level:
#     excitation energy: 329.4
#     spin parity: 4 +
#   final level:
#     excitation energy: 100.1
#     spin parity: 2 +
# columns:
# - title: Neutron energy window, middle of range
#   unit: keV
# - title: gamma cross section
#   unit: barns
# - title: neutron energy window low range
#   unit: keV
# - title: Uncertainty on neutron energy window low range
#   unit: keV
# - title: neutron energy window high range
#   unit: keV
# - title: Uncertainty on neutron energy window high range
#   unit: keV
# - title: cross section parameter uncertainty
#   unit: barns
# - title: cross section intrinsic uncertainty
#   unit: barns
# - title: cross section total combined uncertainty
#   unit: barns
# number of lines: 9
#
     6500.00      0.06613      6000.00        134.2      7000.00        169.1      0.
↪00951      0.00910      0.01316
     7500.00      0.26245      7000.00        169.1      8000.00        206.6      0.
↪03897      0.01471      0.04165
     8500.00      0.50857      8000.00        206.6      9000.00        246.5      0.
↪08370      0.02025      0.08611
     9500.00      0.58062      9000.00        246.5     10000.00        288.7      0.
↪10228      0.03025      0.10666
    11000.00      0.54425     10000.00        288.7     12000.00        379.5      0.
↪09333      0.02709      0.09718
    13000.00      0.54797     12000.00        379.5     14000.00        478.2      0.
↪11671      0.05290      0.12814
    15000.00      0.53738     14000.00        478.2     16000.00        584.2      0.
↪13378      0.07098      0.15144
    17000.00      0.42136     16000.00        584.2     18000.00        697.1      0.
↪10629      0.08091      0.13358
```

(continues on next page)

(continued from previous page)

```
    21500.00      0.18390     18000.00        697.1     25000.00       1141.1      0.
↪05323      0.05675      0.07781
# end of file: True
```

351 keV

Listing 21: Raw result file for the experimental cross section for the 351.0 keV γ ray in the $(n,\ 2n)$ reaction on ^{183}W (transition in the ^{182}W isotope.). As shown in Figure 63.

```
#
# title: 183W(n, n' gamma[350.99 keV - 182L03L02])
# date: '2024-03-18'
# author: Greg Henning
# reaction:
#   proj: N
#   target: 74-W-183
#   code: 74-W-183(N, 2N)74-W-182,PAR,SIG,G
# transition:
#   emitter: 182W
#   gamma energy: 350.99
#   energy unit: keV
#   label: 182L03L02
#   threshold energy: 6870
#   icc: 5.380E-02
#   initial level:
#     excitation energy: 680.4
#     spin parity: 6 +
#   final level:
#     excitation energy: 329.4
#     spin parity: 4 +
# columns:
# - title: Neutron energy window, middle of range
#   unit: keV
# - title: gamma cross section
#   unit: barns
# - title: neutron energy window low range
#   unit: keV
# - title: Uncertainty on neutron energy window low range
#   unit: keV
# - title: neutron energy window high range
#   unit: keV
# - title: Uncertainty on neutron energy window high range
#   unit: keV
# - title: cross section parameter uncertainty
#   unit: barns
# - title: cross section intrinsic uncertainty
#   unit: barns
# - title: cross section total combined uncertainty
#   unit: barns
# number of lines: 10
```

(continues on next page)

(continued from previous page)

```
#
     6500.00      0.01732      6000.00        134.2      7000.00        169.1      0.
↪00333      0.00760      0.00830
     7500.00      0.07365      7000.00        169.1      8000.00        206.6      0.
↪01483      0.00903      0.01736
     8500.00      0.22388      8000.00        206.6      9000.00        246.5      0.
↪04237      0.01154      0.04391
     9500.00      0.27151      9000.00        246.5     10000.00        288.7      0.
↪04171      0.02036      0.04641
    11000.00      0.25586     10000.00        288.7     12000.00        379.5      0.
↪05109      0.01378      0.05292
    13000.00      0.27826     12000.00        379.5     14000.00        478.2      0.
↪05303      0.02790      0.05992
    15000.00      0.30218     14000.00        478.2     16000.00        584.2      0.
↪07153      0.04176      0.08283
    17000.00      0.23749     16000.00        584.2     18000.00        697.1      0.
↪06014      0.05102      0.07887
    21500.00      0.11883     18000.00        697.1     25000.00       1141.1      0.
↪03126      0.04149      0.05195
    26500.00      0.14363     25000.00       1141.1     28000.00       1352.5      0.
↪03652      0.14004      0.14472
# end of file: True
```

9.2 `--help`

This section will describe the main softwares I rely upon for my work and give a few indication on how to use them

9.2.1 Markdown

Markdown is a formatting language created by John Gruber and Aaron Swartz in 2004 as an easy to write and use formatting option. Where *HTML* or Latex need explicit tags to indicate the formatting (such as `<strong></strong>` or `\textbf{}` for bold text), Markdown uses minimalist indicators (** around bold text). That way, the markdown text is still very readable even without being rendered as a *webpage* or *PDF*. In particular, many modern text editors offer syntax highlighting for markdown files, allowing an easy editing of documents.

Markdown files have an extension `.md` or `.mdown` and *MIME type* `text/markdown`.

Note: Variants of markdown exist, such as the git-flavored markdown (`gfm`) used on git repository pages. Variant include specific style and methods such as citations, task list, ...

Tools such as Pandoc can be used to convert markdown files into latex, *PDF* or word documents for larger distributions.

Tip: Visit https://www.markdownguide.org/ for a full documentation of markdown format.

9.2.2 reStructure Text

Another minimal markup language is reStructure Text. It is very similar to Markdown, but includes additional commands (called *directives*) to structure the document (such as *include*). Closer to Latex than *Markdown*, it is more adapted to the production of large documents, inclusion of images, ...

reStructure Text files have an extension `.rst` and *MIME type* `text/x-rst`.

Tip: Visit https://docutils.sourceforge.io/rst.html for a full documentation of reStructure Text format

Note: The current document is written in reStructure Text.

9.2.3 Pandoc

Pandoc is a universal document converter. It *translates* documents from *Markdown*, Latex, ... to other formats (either ligth markup style, or heavier ones such as MS Word or *HTML*).

In particular, it can translate from *Markdown* to Latex and from there be compiled into a *PDF* file (a *PDF* file can also be obtained via the *Markdown* → Word → *PDF* path).

Pandoc includes modules to process math equations and citations.

Typical Pandoc call looks like :

```
$ pandoc --standalone --table-of-contents \
      --biblatex \
      --to=latex --output=output.tex \
      --from=gfm input.md
```

Tip: Visit https://pandoc.org/ for a full documentation on `pandoc`.

9.2.4 Sphinx

Like pandoc, Sphinx converts *Markdown* and *reStructure Text* documents into rendered documents like *HTML* pages or *PDF* using Latex.

Tip: Visit https://www.sphinx-doc.org for a full documentation on Sphinx.

Note: The current document is converted into *HTML* and *PDF* using Sphinx.

Sphinx configuration

Producing this document uses Sphinx in conjunction with specific extensions (in particular for styling). The sphinx configuration can be found in the `conf.py` file. The extensions and environment to run sphinx and produce this document from source is available via the dedicated `Dockerfile` on git.unistra.fr.

9.2.5 Mermaid

Mermaid is a javascript packages that creates diagrams from definition using a simple syntax. The syntax can also be used in more general context to *take notes*, ...

Note: Flow charts in the *Implementation* section are made with Mermaid.

Tip: Visit https://mermaid-js.github.io/mermaid/ for documentation, and https://mermaid.live/ for live trying Mermaid.

9.2.6 Yaml

Yaml (stands for Yaml Ain't Markup Language) is a meta data language. It's goal is to represent organized information (i.e. programming variables) in a human friendly way.

Yaml documents can be converted in to *Json*, *XML*, or Python's dictionaries, ...

A typical YAML document looks like this:

```yaml
FC_DOF:
    U3O8: 27.3981
    UF4: 27.285
u_FC_DOF:
    U3O8: 0.005
    UF4: 0.005

SPEED_OF_LIGHT: 0.299792458 # meters per nanoseconds
TNT_TS_STEP: 10. # nanoseconds per timestamp
```

and will translate into the following json structure:

```json
{
  "FC_DOF": {
    "U3O8": 27.3981,
    "UF4": 27.285
  },
  "u_FC_DOF": {
    "U3O8": 0.005,
    "UF4": 0.005
  },
  "SPEED_OF_LIGHT": 0.299792458,
  "TNT_TS_STEP": 10
}
```

Yaml files are easy to read directly in a text editor and can be imported into a software, which makes them particularly useful to store software configurations.

Yaml files have an extension `.yml` or `.yaml` and *MIME type* `application/yaml`.

In the context of Open Science, Yaml is also a great language to write *metadata* .

Tip: Visit https://yaml.org/ for a full documentation of the yaml format.

9.2.7 git

`git` is version control system designed for tracking changes in source code during software development. It is very widely used in managing projects, and can be adapted to small or large teams, big or tiny projects.

The main feature of `git` is keeping a complete history of the project, including all changes made to the code. The user can then create a flagged version of their files at any point in time, making it easier to go back to a previous version if (when) mistakes are made.

In addition, online services such as github, or gitlab instances provide remote servers on which to send and store the git repository, making it easier to share and keeping it for later use.

Tip: Visit https://git-scm.com/ for documentation on `git`. A good way to learn the versioning system is also to refer to the software carpentry class on the topic .

Note: The current manuscript sources, public pages, scripts, as well as supporting softwares have been written using `git` for version control as well as progress saving. Their public versions are published and hosted on the Université de Strasbourg `gitlab` instance: https://git.unistra.fr/hdr-ghenning

9.2.8 Python

Python is a high-level, interpreted programming language known for its simplicity and readability. Created by Guido van Rossum in 1991, it has since become one of the most popular programming languages worldwide, used in domains such as web development, data analysis, artificial intelligence, scientific computing, and more.

Key features of Python include its clear and concise syntax, which emphasizes readability and reduces the cost of program maintenance. Python uses indentation to denote blocks of code, promoting code consistency and readability among developers.

Python files have an extension `.py` and *MIME type* `application/x-python-code` or `text/x-python`.

Tip: Visit https://www.python.org/ for all things related to Python.

Python comes with an extensive standard library, providing a wide range of extra packages to perform specific tasks such as database access, data manipulation, … Additionally, third-party libraries can *easily be installed* to further extends the language's capabilities with libraries like NumPy, Pandas, Matplotlib and many others.

As an interpreted language, it suffers from slow execution time (compared to C++ for example) and high memory usage. However, these drawbacks are balanced by the high portability of the code that can be executed **without any** modification on Linux, Windows or MacOs plateform.

Danger: Although still in use for many projects, version 2.7 of python is now deprecated. And should **not** be used.

Python gave us the *Zen of python* that are fantastic guiding principle for programming.

pip

`pip` is a python tool to install external *modules* (*modules* are libraries one can load and used to extend python's ability). It takes care of modules' dependencies and versions.

It is recommended to publish python modules on the Pypi website so that they can be installed by the simple command :

```
$ pip install `module-name`
```

Tip: Visit https://pip.pypa.io/ for the `pip` documentation and https://pypi.org/ to browse the available modules.

Note: My own published modules can be found on https://pypi.org/user/ghenning/ .

Virtual environment

A virtual environment in Python is a directory that contains a specific reference to the Python interpreter, and, more importantly, a set of installed packages.

It makes it possible to have an isolated environment for Python projects, ensuring that dependencies are managed separately and avoiding conflicts between different projects that may share the same name, or require different versions of the same package. The packages loaded into a virtual environment are usually limited in number, it makes the dependency management easier than on a system-wide installation where many unrelated packages may be installed.

Virtual environments also allow developers to replicate a specific environment in which to develop and test the code in different computers and between different collaborators.

There are different ways to create virtual environments, depending on the version of python installed and general framework used. For example conda provides package dependence and environment management on a very detail level, but comes with a large codebase (it is a framework that encapsulates python within an extended environment). The virtualenv tool is closer to the *bare* python, managing not only packages but also python version within the virtual environment.

The simplest solution, and the one I personally prefer because it does not require additional packages, is the built-in `venv` module from the python standard library. Even with limited capabilities compared to more advanced solutions, it is always available and does the job.

Jupyter notebooks

Jupyter notebooks, as well as the Jupyter-lab environment is a full python *IDE* that runs as a server and can be accessed via the web-browser.

This offer several advantages: first, the server can run on a specific machine (for example, at the *in2p3* computation center) and be accessed from a desktop or laptop. Second, the using tabs, one can easily organize the workspace and switch between files, terminals and file display.

The Notebooks, especially, mix computing cells (containing for example *Python* code) with rendered *Markdown* text. The final document is therefore directly readable and shows the full process of analysis directly.

Notebooks files are formerly *json* files, with an extension `.ipynb` and *MIME type* `application/x-ipynb+json`.

Note: Most of the development of the *data analysis in this manuscript* has been done within a jupyter-lab environment.

Tip: Visit https://jupyter.org/ to discover Jupyter and Jupyter Lab.

9.2.9 Emacs

Emacs is an historical text editor for Linux. Now available on all platforms. It runs either in the terminal or with a GTK visual interface. It provides syntax highlighting for many languages (including *Python*, *Markdown*, *Yaml*, ...)

Emacs uses many keyboard combination shortcuts to find, modify, save, ... and once the mains ones are remembered, it can be a very powerful tool for programming. The Emacs' keybindings are so specific and well known by their users that they are a frequent possible setting in *IDEs*.

Emacs archenemy is vim, with its own keyboard shortcuts. The Emacs *vs.* Vim conflict has been raging for years, with no signs of ever ending[1].

Tip: Visit https://www.gnu.org/software/emacs/ to get `Emacs`.

9.2.10 Visual Studio code

Visual Studio Code (also known as VS Code) is a modern source code editor by Microsoft (launched in 2015). Although a Microsoft product, it has been developed on other platforms and can be run on Windows, macOS, and Linux, as well as online (when connected to a github account for example) providing a consistent development experience across different operating systems.

It is a highly customizable *IDE*, supporting syntax highlighting for an extensive list of programming languages. It can also load extensions that augment its capabilities. For example, one can preview *Markdown* or *reStructure Text* documents, open and run *jupyter notebooks*, perform *git* operations, ... from within VS Code. The shortcuts can also be redefined, to fit, for example, the *Emacs* key bindings.

Note: This manuscript has been written within VS code.

Tip: Visit https://code.visualstudio.com/ to see how you can get VS Code on your computer.

9.2.11 Docker

Docker is a containerization platform that allows users to package applications and their dependencies into isolated environments called containers. These containers can run on any operating system without modification, providing consistency across different environments.

This solution ensures the consistency of an environment for research projects and softwares (*See more about that in the Open Source section* and[2]). T he Docker image packages the relevant code along with all necessary dependencies, ensuring that the software will run in a consistent way regardless of where it is executed.

Furthermore, Docker images can be exported and shared, making it easier to share the content.

Along with Docker, other containerization softwares exists, such as Singularity/Apptainer or Podman, among many others.

[1] As you can guess from the presence of emacs in this list, I am in the Emacs team.
[2] Greg Henning. Using containers for data analysis. 2020. <hal-02553519>

Note: The *Sphinx* environnement (version, extension, ...) used to compile the reStructure sources files into *HTML* and *PDF* is contained in a docker image.

On Windows systems, Docker uses the Windows Subsystem for Linux to run the containers.

Tip: Visit https://www.docker.com/ to install Docker and https://hub.docker.com/ to browse the library of publicly available images.

9.2.12 Talys

Talys is a nuclear reaction code for the prediction of reaction cross sections and related quantities[3]. It can handle reactions induced by neutrons, photons, protons, deuterons, and other light ions, on target nuclei from mass 12 to the heaviest tabulated isotopes, and for incident energies from the eV to hundreds of MeV range.

For the beginner, *Talys* can be used with minimal inputs. Indeed, it just needs four keywords to run properly: the `projectile` and its `energy`, and the `mass` of the target `element`. However, for more detailed works, the number of keywords that allow the Talys calculation to be tweaked is huge. An isotope level scheme can be changed, one level or one transitions at a time. Isomer life time can be modified. Closer to the theoretical modelling, the level density, optical model potential, fission barrier, preequilibrium, and so on, parameters can be changed.

The output of *Talys* can also be extensive: cross section for many different output channels including partial transitions or level production.

Note: Since version 2.0, Talys can be tried live, online.

Tip: Visit https://nds.iaea.org/talys/ to download and install Talys.

9.3 File format for $(\mathrm{n},\ x\mathrm{n}\ \gamma)$ cross section data

A key point of the *FAIR* principles of *Interoperability* and *Reusability* is to have the data easily readable. To that purpose, the data format should be in a well defined format and, if possible, the software to read and interpret it should be available too. All this is stated in the *metadata* along with the data, but there's still a requirement to have a well defined format.

Some data sets have a predefined format that can't be changed (for example, raw data directly from the acquisition system), some others are totally up to the user (histograms data, final values, ...). A wise choice in file data is needed.

In the context of nuclear data, several data format exist already.

ENDF-6
: The historical evaluated nuclear data format (*ENDF*) has been around since 1960 (with the latest update in 2011), and bears the stigma of history. Files are divided into material (*MAT*), data type (*MF*), reaction (*MT*) sections. The format uses fixed columns and other methods from *old time* when storage space was scarce and data was stored on tapes. It is particularly **not** human-readable. An excerpt from the ^{183}W section of the *JEFF* 3.3 library related to $(\mathrm{n},\ \mathrm{n}')$ reaction is given in Listing 22, and as you can see, it's not easy to understand what's going on...

[3] TALYS: modeling of nuclear reactions, Arjan Koning, Stephane Hilaire, Stephane Goriely. Eur. Phys. J. A 59 (6) 131 (2023) DOI: 10.1140/epja/s10050-023-01034-3

Listing 22: Part of the ^{183}W entry in *JEFF*-3.3, in *ENDF*-6 format. Note the use of, for example, `5.0+4` to mean `5.0E+04`, a space saving format.

```
 7.418300+4 1.813786+2          1          0          2          17434 1451    1
 0.000000+0 0.000000+0          0          0          0          67434 1451    2
 1.000000+0 1.500000+8          3          0         10          37434 1451    3
 0.000000+0 0.000000+0          0          0        375        1567434 1451    4
 74-W -183  FZK/IRS,+ EVAL-OCT13 P.Pereslavtsev et al., +         7434 1451    5
  PE03,PE04           DIST-DEC17 REV3-DEC17            20171231   7434 1451    6
---JEFF-3.3           MATERIAL 7434                                7434 1451    7
-----INCIDENT NEUTRON DATA                                        7434 1451    8
------ENDF-6 FORMAT                                              7434 1451    9
...
 MF=3  MT=4  TOTAL INELASTIC CROSS SECTION                        7434 1451  156
                                                                  7434 1451  157
   Equal to the sum of the inelastic level excitation and         7434 1451  158
 continuum cross sections.                                        7434 1451  159
...
------------------------ C O N T E N T ---------------------------- 7434 1451  379
 7.418300+4 1.813786+2          0          0          0           07434 3  4    1
 0.000000+0-4.648300+4          0          0          1        1007434 3  4    2
        100          2                                           7434 3  4    3
 4.673928+4 0.000000+0 5.000000+4 8.365700-2 6.000000+4 2.245610-17434 3  4    4
 7.000000+4 3.113560-1 8.000000+4 3.734510-1 9.000000+4 4.205670-17434 3  4    5
 9.962525+4 4.461144-1 1.000000+5 4.481090-1 1.200000+5 5.047930-17434 3  4    6
 1.400000+5 5.293290-1 1.500000+5 5.417281-1 1.600000+5 5.563270-17434 3  4    7
 1.800000+5 5.795610-1 2.000000+5 5.965609-1 2.081000+5 6.012058-17434 3  4    8
 2.081533+5 6.012374-1 2.099572+5 6.036410-1 2.102000+5 6.049178-17434 3  4    9
 2.500000+5 1.070499+0 2.933324+5 1.318352+0 2.936000+5 1.321558+07434 3  4   10
 3.000000+5 1.398313+0 3.106493+5 1.463571+0 3.107000+5 1.463884+07434 3  4   11
 3.111993+5 1.466763+0 3.112000+5 1.466767+0 4.000000+5 1.691896+07434 3  4   12
 4.143000+5 1.719826+0 4.143670+5 1.719945+0 4.555000+5 1.816094+07434 3  4   13
 4.555689+5 1.816244+0 4.896850+5 1.919803+0 4.897000+5 1.919849+07434 3  4   14
 5.000000+5 1.949524+0 5.541384+5 2.045104+0 5.571000+5 2.050429+07434 3  4   15
 5.986223+5 2.106552+0 6.000000+5 2.108461+0 6.033000+5 2.112859+07434 3  4   16
 6.234000+5 2.138962+0 6.261935+5 2.142214+0 7.000000+5 2.237573+07434 3  4   17
 7.439994+5 2.289433+0 7.441000+5 2.289683+0 7.500000+5 2.303002+07434 3  4   18
 8.000000+5 2.388849+0 9.000000+5 2.510261+0 1.000000+6 2.614783+07434 3  4   19
 1.100000+6 2.727683+0 1.200000+6 2.797268+0 1.300000+6 2.867075+07434 3  4   20
 1.400000+6 2.895882+0 1.500000+6 2.937643+0 1.600000+6 2.970448+07434 3  4   21
 1.700000+6 2.985648+0 1.800000+6 3.000151+0 1.900000+6 2.998522+07434 3  4   22
 2.000000+6 3.001463+0 2.100000+6 3.019932+0 2.200000+6 3.021913+07434 3  4   23
 2.300000+6 3.021152+0 2.400000+6 3.014486+0 2.500000+6 3.008775+07434 3  4   24
 2.600000+6 2.993317+0 2.800000+6 2.964127+0 3.000000+6 2.937316+07434 3  4   25
 3.200000+6 2.917485+0 3.400000+6 2.898280+0 3.500000+6 2.888906+07434 3  4   26
 3.600000+6 2.884546+0 3.800000+6 2.876070+0 4.000000+6 2.867934+07434 3  4   27
 4.200000+6 2.847658+0 4.400000+6 2.827546+0 4.500000+6 2.817547+07434 3  4   28
 4.600000+6 2.810430+0 4.800000+6 2.796244+0 5.000000+6 2.782134+07434 3  4   29
 5.500000+6 2.736055+0 6.000000+6 2.691355+0 6.500000+6 2.608172+07434 3  4   30
 7.000000+6 2.407241+0 7.500000+6 2.035581+0 8.000000+6 1.775451+07434 3  4   31
 8.500000+6 1.504183+0 9.000000+6 1.243975+0 1.000000+7 8.460758-17434 3  4   32
 1.100000+7 6.561284-1 1.200000+7 5.358298-1 1.300000+7 4.556767-17434 3  4   33
```

(continues on next page)

(continued from previous page)

```
 1.400000+7 4.047817-1 1.450000+7 3.836709-1 1.500000+7 3.625604-17434 3  4    34
 1.600000+7 3.338595-1 1.700000+7 3.121871-1 1.800000+7 2.900821-17434 3  4    35
 1.900000+7 2.678049-1 2.000000+7 2.537605-1 2.000001+7 0.000000+07434 3  4    36
 1.500000+8 0.000000+0                                          7434 3  4    37
 0.000000+0 0.000000+0          0          0          0          07434 3  099999
```

GNDS

To overcome the shortcoming of *ENDF*-6, a working group of the Working Party on International Nuclear Data Evaluation Cooperation was formed in 2012, to propose a new, more modern format, that would allow more flexibility and has fewer limitations. The Generalized Nuclear Database Structure (*GNDS*), released in 2020[1], is before all a structure that can be implemented into *XML*, *Json*, ...

The *GNDS* formatted version of the data presented in Listing 22 in *ENDF* format is given in Listing 23. It is somewhat more readable.

Listing 23: Same data as in Listing 22, but formatted in *GNDS*.

```
<reactionSuite projectile="n" target="W183" format="gnd version 1.2" temperature="0␣
↪K">
  <styles>
    <style name="evaluated" version="3.3.1" library="JEFF"/>
  </styles>
  <documentations>
  </documentations>
  <summedReaction label="30" name="(z,n)" Q="-46483 eV" date="2013-10-01" ENDF_MT="4
↪">
    <crossSection nativeData="linear">
      <linear xData="XYs" length="100" accuracy="0.001">
      <axes>
        <axis index="0" label="energy_in" unit="eV" interpolation="linear,linear"␣
↪frame="lab"/>
        <axis index="1" label="crossSection" unit="b" frame="lab"/>
      </axes>
      <data>
            46739.28 0 5e4 0.083657 6e4 0.224561 7e4 0.311356 8e4 0.373451 9e4 0.
↪420567 99625.25 0.4461144
            1e5 0.448109 1.2e5 0.504793 1.4e5 0.529329 1.5e5 0.5417281 1.6e5 0.
↪556327 1.8e5 0.579561 2e5
            0.5965609 208100 0.6012058 208153.3 0.6012374 209957.2 0.603641 210200␣
↪0.6049178 2.5e5 1.070499
            293332.4 1.318352 293600 1.321558 3e5 1.398313 310649.3 1.463571 310700␣
↪1.463884 311199.3 1.466763
            311200 1.466767 4e5 1.691896 414300 1.719826 414367 1.719945 455500 1.
↪816094 455568.9 1.816244
            489685 1.919803 489700 1.919849 5e5 1.949524 554138.4 2.045104 557100 2.
↪050429 598622.3 2.106552
            6e5 2.108461 603300 2.112859 623400 2.138962 626193.5 2.142214 7e5 2.
↪237573 743999.4 2.289433
            744100 2.289683 7.5e5 2.303002 8e5 2.388849 9e5 2.510261 1e6 2.614783 1.
↪1e6 2.727683 1.2e6 2.797268
            1.3e6 2.867075 1.4e6 2.895882 1.5e6 2.937643 1.6e6 2.970448 1.7e6 2.
```

(continues on next page)

[1] Specifications for the Generalised Nuclear Database Structure (GNDS)

(continued from previous page)

```
↪985648 1.8e6 3.000151 1.9e6
           2.998522 2e6 3.001463 2.1e6 3.019932 2.2e6 3.021913 2.3e6 3.021152 2.
↪4e6 3.014486 2.5e6 3.008775
           2.6e6 2.993317 2.8e6 2.964127 3e6 2.937316 3.2e6 2.917485 3.4e6 2.89828␣
↪3.5e6 2.888906 3.6e6
           2.884546 3.8e6 2.87607 4e6 2.867934 4.2e6 2.847658 4.4e6 2.827546 4.5e6␣
↪2.817547 4.6e6 2.81043
           4.8e6 2.796244 5e6 2.782134 5.5e6 2.736055 6e6 2.691355 6.5e6 2.608172␣
↪7e6 2.407241 7.5e6 2.035581
           8e6 1.775451 8.5e6 1.504183 9e6 1.243975 1e7 0.8460758 1.1e7 0.6561284␣
↪1.2e7 0.5358298 1.3e7
           0.4556767 1.4e7 0.4047817 1.45e7 0.3836709 1.5e7 0.3625604 1.6e7 0.
↪3338595 1.7e7 0.3121871 1.8e7
           0.2900821 1.9e7 0.2678049 2e7 0.2537605 20000010 0 1.5e8 0
      </data>
    </linear>
   </crossSection>
  </summedReaction>
</reactionSuite>
```

However, the *GNDS* format has one major issue: its versatility. It is very hard to write a full file or just one data set. There are many options, types, … It can be cumbersome to write/read, in particular for a physicist that just wants to format its final data into a standardized format for publication.

Exfor

Exfor has a format for experimental nuclear data sets. Like *ENDF*, it relies on fixed length format.

Listing 24 is an example of the *Exfor* entry for refrence[2] and[3] (See *Population of an isomer in (\text{n},~\text{n}') reactions*).

Listing 24: Part of the *Exfor* entry `21818.007`[Page 150, 2].

```
ENTRY            21818   20180311   20190222   20190220       2270
SUBENT        21818001   20180311   20190222   20190220       2270
BIB                 14         39
INSTITUTE  (2GERKIG) Experimental site,R.P.,H.-U.F.
           (2GERHAM) B.A.
REFERENCE  (C,82ANTWER,,859,1982)
AUTHOR     (B.Anders,R.Pepelnik,H.-U.Fanger)
TITLE       Application of a novel 14 MeV neutron activation
            analysis system for cross section measurements
            with short-lived nuclides
FACILITY   (NGEN,2GERKIG) 14 MeV neutron generator KORONA at
                          GKSS Geesthacht.
INC-SOURCE (D-T) No details given.
INC-SPECT  (EN-RSV-HW) Neutron flux more that 3 x 10**10 n/cm2/s
METHOD     .Activation. The gamma line of the decay of the
            isomeric state is measured.
DETECTOR   (GELI) 18% Ge(Li) detector at 26.5 mm from the sample.
```

(continues on next page)

[2] http://www-nds.iaea.org/EXFOR/21818.007

[3] Anders, B., Pepelnik, R., Fanger, HU. (1983). Application of a Novel 14-Mev Neutron Activation Analysis System for Cross-Section Measurements with Short-Lived Nuclides. In: Böckhoff, K.H. (eds) Nuclear Data for Science and Technology. Springer, Dordrecht. https://doi.org/10.1007/978-94-009-7099-1_193

(continued from previous page)

```
           (LONGC) Long counter to monitor the neutron
            source strength of about 3.E+12 n/sec.
MONITOR    (23-V-51(N,P)22-TI-51,,SIG)  - value is given in
           82ANTWER,,859 with comment that it is measured
           previously at KORONA relative to 27Al(n,alpha) and
           27Al(n,p) reaction. Similar value is given in 21999.014
CORRECTION .Coincidence summing up of gamma-lines,
           .absorption of the gamma lines.
ERR-ANALYS (ERR-T) Total error given. It is composed as follows
            * Gamma intensity errors.
           (ERR-1,0.3,3.) the error of the peak area estimated by
            the peak analysis program                    0.3-3.0%
           (ERR-2,1.5,2.5) Summing correction errors,    1.5-2.5%
           (ERR-3) Absorption correction errors.              5%
           (ERR-4) Flux and gamma-ray efficiency calibration
                                                  estimated to 3%
            * Errors due to half-lives, mass determinations
              and isotopic abundances  are negligible.
STATUS     (TABLE) Data taken from 82ANTWER,,859,1982.
HISTORY    (19830120C) G.C.
           (20180311A) SD:Updated to new date formats,lower case.
           Comma in main ref. was added. ERR-ANALYS update. Data
           were update according to presentation in the Table.
           Two Subents (013,014) were added. REACTION code
           Subent 012 corrected. EN-RSL -> EN-RSL-HW.
ENDBIB              39
COMMON               6          3
EN         EN-RSL-HW  MONIT      MONIT-ERR  ERR-3      ERR-4
MEV        MEV        MB         MB         PER-CENT   PER-CENT
    14.6      0.2       28.0        1.5          5.          3.
ENDCOMMON            3
ENDSUBENT           46
SUBENT        21818007   20180311   20190222   20190220       2270
BIB                  4          5
REACTION   (74-W-183(N,INL)74-W-183-M,,SIG)
SAMPLE     (74-W-183,ENR=0.774) .W sample 77.4% enriched in W-183.
DECAY-DATA (74-W-183-M,5.3SEC,DG,108.,0.18,DG,160.,0.049,
            DG,210.,0.009)
STATUS     (TABLE) Data taken from table 1 of 82ANTWER,,859,1982.
ENDBIB               5
NOCOMMON             0          0
DATA                 2          1
DATA       ERR-T
MB         MB
       127.        14.
ENDDATA              3
ENDSUBENT           13
ENDENTRY             2
```

These formats suffer from their age (for *ENDF* and *Exfor*) or there versatility (for *GNDS*). Old formats are hard to read and write, the new *GNDS* has so many options, types, ... that writing one little data set into the format is a challenge. In other words, GNDS is a good exchange format (it can easily be read by a software and the data can be transformed into other format). But we have to remember that *Readability counts*, therefore publishing our data, even a very little dataset, should be done with caution and using a format that is easy to write (for the producer) and to read and import into a database.

The good news, is that *Exfor* allows the export of data set in *json* format. Listing 25 is the *Exfor* data set used as example *earlier*, exported in *json*.

Listing 25: Same data as in Listing 24 but formatted in *json*.

```
{
    "format":"X4JSON.0.1.2"
    ,"now":"2023-04-26T14:02:12.000Z"
    ,"program":"EXFOR converter, by V.Zerkin, IAEA-NDS, 2007-2019 (ver.2020-05-11)"
    ,"input": {  "files": [ { "name":"X4R8350_tdat.x4"  ,"format":"EXFOR"  ,"created":
↪"2023-04-26T14:02:12.000Z"  } ] }
    ,"output": { "files": [ { "name":"X4R8350_tdat.x4.x4t"  ,"format":"JSON"  ,"outBib":
↪"0"  ,"created":"2023-04-26T14:02:12.000Z"  } ] }
    ,"datasets": [

    {"id":"21818007"
     ,"subent": {"id":"21818007", "updated":"20180311" }
     ,"entry": {"id":"21818", "updated":"20180311" }
     ,"author1":"B.Anders+"
     ,"year":"1982"
     ,"ref1":{ "code":"C,82ANTWER,,859,1982" ,"std":"C,82ANTWER,,859,1982"
                ,"exp":"Conf: Conf.on Nucl.Data for Sci.and Technol.,Antwerp 1982, p.859␣
↪(1982)"
              }
     ,"reaction": {
        "code": "74-W-183(N,INL)74-W-183-M,,SIG"
       ,"C4Reaction":"(N,INL),SIG"
       ,"Proj":"N" ,"ProjZA":"1"
       ,"Target":"74-W-183" ,"TargZA":"74183"
       ,"ProdMeta":"M"
       ,"MF": 3 ,"MT": 4
       ,"ReactionType":"CS"
       ,"Quantity":"Cross section"
       ,"IndVarFamCode":"0 2"
       ,"ExpectedUnits":"B"
     }
     ,"datasect": {
       "xVariables": 1
       ,"formula":"Y = Y(X1)"
       ,"col": 10
       ,"row": 1
     }
     ,"headers": [

        { "ii":1 ,"varType":"Data" ,"varVar":"variable"
       ,"nval":1   ,"minval":"127."   ,"maxval":"127."
       ,"varSorted":"0.1" ,"what":"Y.Value"
```

(continues on next page)

(continued from previous page)


```
       ,"dataType":"21"   ,"numVar":"0.1"
       ,"x4Header":"DATA" ,"SAN":"007" ,"expansion":"Data: data"
       ,"x4Units":"MB" ,"hlpUnits":"millibarns"
       ,"BasicUnits":"B" ,"ConvFactor":"0.001"
       }

      , { "ii":2 ,"varType":"Error" ,"varVar":"variable"
       ,"nval":1    ,"minval":"14."    ,"maxval":"14."
       ,"varSorted":"0.911" ,"what":"Y.Err+-"
       ,"dataType":"21"   ,"numVar":"0.911"
       ,"x4Header":"ERR-T" ,"SAN":"007" ,"expansion":"Uncertainty: +-error"
       ,"x4Units":"MB" ,"hlpUnits":"millibarns"
       ,"BasicUnits":"B" ,"ConvFactor":"0.001"
       }

      , { "ii":3 ,"varType":"Error" ,"varVar":"constant"
       ,"nval":1    ,"minval":"0.3"    ,"maxval":"0.3"
       ,"varSorted":"0.955" ,"what":"Y.pErr+-"
       ,"dataType":"21"   ,"numVar":"0.955"
       ,"x4Header":"ERR-1" ,"SAN":"007" ,"expansion":"Uncertainty: +-partial error"
       ,"x4Units":"PER-CENT" ,"hlpUnits":"per-cent"
       }

      , { "ii":4 ,"varType":"Error" ,"varVar":"constant"
       ,"nval":1    ,"minval":"1.5"    ,"maxval":"1.5"
       ,"varSorted":"0.955" ,"what":"Y.pErr+-"
       ,"dataType":"21"   ,"numVar":"0.955"
       ,"x4Header":"ERR-2" ,"SAN":"007" ,"expansion":"Uncertainty: +-partial error"
       ,"x4Units":"PER-CENT" ,"hlpUnits":"per-cent"
       }

      , { "ii":5 ,"varType":"Error" ,"varVar":"constant"
       ,"nval":1    ,"minval":"5."    ,"maxval":"5."
       ,"varSorted":"0.955" ,"what":"Y.pErr+-"
       ,"dataType":"21"   ,"numVar":"0.955"
       ,"x4Header":"ERR-3" ,"SAN":"001" ,"expansion":"Uncertainty: +-partial error"
       ,"x4Units":"PER-CENT" ,"hlpUnits":"per-cent"
       }

      , { "ii":6 ,"varType":"Error" ,"varVar":"constant"
       ,"nval":1    ,"minval":"3."    ,"maxval":"3."
       ,"varSorted":"0.955" ,"what":"Y.pErr+-"
       ,"dataType":"21"   ,"numVar":"0.955"
       ,"x4Header":"ERR-4" ,"SAN":"001" ,"expansion":"Uncertainty: +-partial error"
       ,"x4Units":"PER-CENT" ,"hlpUnits":"per-cent"
       }

      , { "ii":7 ,"varType":"Error" ,"varVar":"constant"
       ,"nval":1    ,"minval":"1.5"    ,"maxval":"1.5"
       ,"varSorted":"0.955" ,"what":"Y.pErr+-"
       ,"dataType":"21"   ,"numVar":"0.955"
       ,"x4Header":"MONIT-ERR" ,"SAN":"001" ,"expansion":"Uncertainty: +-partial error"
```


(continues on next page)

(continued from previous page)


```
      ,"x4Units":"MB" ,"hlpUnits":"millibarns"
      ,"BasicUnits":"B" ,"ConvFactor":"0.001"
      }

     , { "ii":8 ,"varType":"Data" ,"varVar":"constant"
      ,"nval":1   ,"minval":"14.6"   ,"maxval":"14.6"
      ,"varSorted":"1.1" ,"what":"X1.Value"
      ,"dataType":"41"  ,"numVar":"2.1"
      ,"x4Header":"EN" ,"SAN":"001" ,"expansion":"Incident energy: energy"
      ,"x4Units":"MEV" ,"hlpUnits":"MeV"
      ,"BasicUnits":"EV" ,"ConvFactor":"1.e+06"
      }

     , { "ii":9 ,"varType":"Error" ,"varVar":"constant"
      ,"nval":1   ,"minval":"0.2"   ,"maxval":"0.2"
      ,"varSorted":"1.933" ,"what":"X1.hRes+-"
      ,"dataType":"41"  ,"numVar":"2.933"
      ,"x4Header":"EN-RSL-HW" ,"SAN":"001" ,"expansion":"Uncertainty: +-half resolution"
      ,"x4Units":"MEV" ,"hlpUnits":"MeV"
      ,"BasicUnits":"EV" ,"ConvFactor":"1.e+06"
      }

     , { "ii":10 ,"varType":"Data" ,"varVar":"constant"
      ,"nval":1   ,"minval":"28."   ,"maxval":"28."
      ,"varSorted":"" ,"what":"ZZZ"
      ,"dataType":"11"  ,"numVar":"ZZZ"
      ,"x4Header":"MONIT" ,"SAN":"001" ,"expansion":"Assumed values: monitor"
      ,"x4Units":"MB" ,"hlpUnits":"millibarns"
      ,"BasicUnits":"B" ,"ConvFactor":"0.001"
      }
     ]
    ,"data": [
       [127.0,14.0,0.3,1.5,5.0,3.0,1.5,14.6,0.2,28.0]
     ]
   }
  ]
  }
```


The *json* format is not the easiet to read, but can be completely reformated in *yaml*, which is very human friendly, as seen in Listing 26

Listing 26: Same data as in Listing 25 but formatted in *Yaml*.


```
---
format: X4JSON.0.1.2
now: '2023-04-26T14:02:12.000Z'
program: EXFOR converter, by V.Zerkin, IAEA-NDS, 2007-2019 (ver.2020-05-11)
input:
  files:
  - name: X4R8350_tdat.x4
    format: EXFOR
    created: '2023-04-26T14:02:12.000Z'
```


(continues on next page)

(continued from previous page)


```
output:
  files:
  - name: X4R8350_tdat.x4.x4t
    format: JSON
    outBib: '0'
    created: '2023-04-26T14:02:12.000Z'
datasets:
- id: '21818007'
  subent:
    id: '21818007'
    updated: '20180311'
  entry:
    id: '21818'
    updated: '20180311'
  author1: B.Anders+
  year: '1982'
  ref1:
    code: C,82ANTWER,,859,1982
    std: C,82ANTWER,,859,1982
    exp: 'Conf: Conf.on Nucl.Data for Sci.and Technol.,Antwerp 1982, p.859 (1982)'
  reaction:
    code: 74-W-183(N,INL)74-W-183-M,,SIG
    C4Reaction: "(N,INL),SIG"
    Proj: N
    ProjZA: '1'
    Target: 74-W-183
    TargZA: '74183'
    ProdMeta: M
    MF: 3
    MT: 4
    ReactionType: CS
    Quantity: Cross section
    IndVarFamCode: 0 2
    ExpectedUnits: B
  datasect:
    xVariables: 1
    formula: Y = Y(X1)
    col: 10
    row: 1
  headers:
  - ii: 1
    varType: Data
    varVar: variable
    nval: 1
    minval: '127.'
    maxval: '127.'
    varSorted: '0.1'
    what: Y.Value
    dataType: '21'
    numVar: '0.1'
    x4Header: DATA
    SAN: '007'
```


(continues on next page)

(continued from previous page)

```
    expansion: 'Data: data'
    x4Units: MB
    hlpUnits: millibarns
    BasicUnits: B
    ConvFactor: '0.001'
  - ii: 2
    varType: Error
    varVar: variable
    nval: 1
    minval: '14.'
    maxval: '14.'
    varSorted: '0.911'
    what: Y.Err+-
    dataType: '21'
    numVar: '0.911'
    x4Header: ERR-T
    SAN: '007'
    expansion: 'Uncertainty: +-error'
    x4Units: MB
    hlpUnits: millibarns
    BasicUnits: B
    ConvFactor: '0.001'
  - ii: 3
    varType: Error
    varVar: constant
    nval: 1
    minval: '0.3'
    maxval: '0.3'
    varSorted: '0.955'
    what: Y.pErr+-
    dataType: '21'
    numVar: '0.955'
    x4Header: ERR-1
    SAN: '007'
    expansion: 'Uncertainty: +-partial error'
    x4Units: PER-CENT
    hlpUnits: per-cent
  - ii: 4
    varType: Error
    varVar: constant
    nval: 1
    minval: '1.5'
    maxval: '1.5'
    varSorted: '0.955'
    what: Y.pErr+-
    dataType: '21'
    numVar: '0.955'
    x4Header: ERR-2
    SAN: '007'
    expansion: 'Uncertainty: +-partial error'
    x4Units: PER-CENT
    hlpUnits: per-cent
```

(continues on next page)

(continued from previous page)

```
- ii: 5
  varType: Error
  varVar: constant
  nval: 1
  minval: '5.'
  maxval: '5.'
  varSorted: '0.955'
  what: Y.pErr+-
  dataType: '21'
  numVar: '0.955'
  x4Header: ERR-3
  SAN: '001'
  expansion: 'Uncertainty: +-partial error'
  x4Units: PER-CENT
  hlpUnits: per-cent
- ii: 6
  varType: Error
  varVar: constant
  nval: 1
  minval: '3.'
  maxval: '3.'
  varSorted: '0.955'
  what: Y.pErr+-
  dataType: '21'
  numVar: '0.955'
  x4Header: ERR-4
  SAN: '001'
  expansion: 'Uncertainty: +-partial error'
  x4Units: PER-CENT
  hlpUnits: per-cent
- ii: 7
  varType: Error
  varVar: constant
  nval: 1
  minval: '1.5'
  maxval: '1.5'
  varSorted: '0.955'
  what: Y.pErr+-
  dataType: '21'
  numVar: '0.955'
  x4Header: MONIT-ERR
  SAN: '001'
  expansion: 'Uncertainty: +-partial error'
  x4Units: MB
  hlpUnits: millibarns
  BasicUnits: B
  ConvFactor: '0.001'
- ii: 8
  varType: Data
  varVar: constant
  nval: 1
  minval: '14.6'
```

(continues on next page)

(continued from previous page)

```
    maxval: '14.6'
    varSorted: '1.1'
    what: X1.Value
    dataType: '41'
    numVar: '2.1'
    x4Header: EN
    SAN: '001'
    expansion: 'Incident energy: energy'
    x4Units: MEV
    hlpUnits: MeV
    BasicUnits: EV
    ConvFactor: '1.e+06'
  - ii: 9
    varType: Error
    varVar: constant
    nval: 1
    minval: '0.2'
    maxval: '0.2'
    varSorted: '1.933'
    what: X1.hRes+-
    dataType: '41'
    numVar: '2.933'
    x4Header: EN-RSL-HW
    SAN: '001'
    expansion: 'Uncertainty: +-half resolution'
    x4Units: MEV
    hlpUnits: MeV
    BasicUnits: EV
    ConvFactor: '1.e+06'
  - ii: 10
    varType: Data
    varVar: constant
    nval: 1
    minval: '28.'
    maxval: '28.'
    varSorted: ''
    what: ZZZ
    dataType: '11'
    numVar: ZZZ
    x4Header: MONIT
    SAN: '001'
    expansion: 'Assumed values: monitor'
    x4Units: MB
    hlpUnits: millibarns
    BasicUnits: B
    ConvFactor: '0.001'
  data:
  - - 127
    - 14
    - 0.3
    - 1.5
    - 5
```

(continues on next page)

(continued from previous page)

```
  - 3
  - 1.5
  - 14.6
  - 0.2
  - 28
```

Even in *yaml*, the data from *Exfor* suffers from its origin (the *old Exfor format*). But we can get inspired by it to include *metadata* in our data set and facilitate their import into databases.

An example of *yaml* formatted metadata for one cross section file is below (other examples are given in *Experimental (n, n' \gamma) and (n, 2n \gamma) cross sections*):

Listing 27: *Yaml* formatted metadata from a given *cross section file from our analysis*.

```
title: 183W(n, n' gamma[209.8 keV - 183L06L02])
date: '2024-03-18'
author: Greg Henning
reaction:
  proj: N
  target: 74-W-183
  code: 74-W-183(N, INL)74-W-183,PAR,SIG,G
transition:
  emitter: 183W
  gamma energy: 209.8
  energy unit: keV
  label: 183L06L02
  threshold energy: 308.0
  icc: 2.620E-01
  initial level:
    excitation energy: 308.9
    spin parity: 9/2 -
  final level:
    excitation energy: 99.1
    spin parity: 5/2 -
columns:
- title: Neutron energy window, middle of range
  unit: keV
- title: gamma cross section
  unit: barns
- title: neutron energy window low range
  unit: keV
- title: Uncertainty on neutron energy window low range
  unit: keV
- title: neutron energy window high range
  unit: keV
- title: Uncertainty on neutron energy window high range
  unit: keV
- title: cross section parameter uncertainty
  unit: barns
- title: cross section intrinsic uncertainty
  unit: barns
- title: cross section total combined uncertainty
```

(continues on next page)

(continued from previous page)

```
  unit: barns
number of lines: 17

end of file: True
```

The choice of this format and content of meta data has been made for even-even Tungsten results[4] [and5] and the results presented in the current manuscript.

9.3.1 Summary on data format

As used in tungsten data sets, the combination of *yaml* and text files is a simple, easy to read, choice. That works well for small data sets (*json* would be a good format choice for publication too).

For larger data sets (histograms, matrices, ...) the HDF5[6] format would a good choice. HDF is a hierarchical format, and the data is stored in binary format. Therefore, it's not directly readable by *humans*. However, it is a standard format that can be read by many software libraries, making it a good option.

The Tabular Data Package format[7] can also be used. It combines comma separated values (*CSV*) with added *json* metadata. Its specifications even include some general metadata associated with the dataset.

9.4 Metadata standards

When writing metadata, one should use some predefined standards to ensure that specific information can be read correctly, depending on the information to record.

Dates
: For dates, one should use the ISO-8601 string format *YYYY-MM-DD* or *YYYYMMDD*. Additionally, this date format has the interest, when use at the start of a file's name, that when sorted *by name*, the sorted file list will be also sorted by date (older first).

File type
: For file type and format, the MIME scheme is a great choice. It will not usually include science specific types (such as `.root` or `.fast`), but text encoded, images, *PDFs*:, and generic binary will be covered.

Language
: When indicating in which language is a document, following the ISO 639-1 scheme is a good option. It is simple and one will most generally use `en` (English).

Places
: The website GeoName offers references to many places, with permalink to them (for example, Strasbourg) or can reference coordinates (48.584/7.746). This is a nice way to reference a specific place on a map.

Laboratories and institutions
: The Research Organization Registry provides `ROR ID`, *DOI*-like references for laboratories, universities, ... (for example: IPHC).

Researchers
: For researcher, the Open Researcher and Contributor ID (Orcid) is the standard way to identify a researcher across platforms (example: Greg Henning).

[4] "Improving the accuracy of 182,184,186W(n, n'γ) cross sections calculations" G. Henning, M. Dupuis, M. Kerveno, R. Capote, Ph. Dessagne, S. Hilaire, T. Kawano, P. Romain, C. de Saint Jean, P. Scholtes, P. Tamagno. In preparation for Physical Review C. (2024)

[5] HENNING, Greg, 2024, "Experimental (n, n' gamma) cross sections for isotopes 182,184 and 186W", https://doi.org/10.57745/JRCNEJ, Recherche Data Gouv, V1

[6] The HDF5 Library and File Format

[7] Tabular Data Package

Research Topics
: For research topics, one can refer to the old style Physics and Astronomy Classification Scheme (Pacs) numbers , but these should be replaced by the new Physics Subject Headings (PhySH). For more general topics, the Library of Congress Subject Headings (LCSH) provides more subjects. Wikidata or the Bibliothèque nationale de France provide subject ID schemes, but the list of headings is not as easy to search.

Researcher contributions
: To indicate the roles and contributions of collaborators in a project, several schemes exist. The DataCite scheme includes, for the `Contributor` fields, several `contributorType` values, with an emphasis on *project* management. An alternative scheme, that is aimed at being used in publications (either of paper or data), is the Contributor Roles Taxonomy (**Cr**edi**t**), with 14 *roles* for collaborators.

9.5 Updates

This section contains the updated information on some elements mentioned in the main text, that were updated between the manuscript submission to the jury and the defense of the HDR.

It seems appropriate to put the updates in a separate section, in order to, first, maintain the cohesion of the main text, and second, show the progression of the work.

9.5.1 Exfor entry `41245.022`

The questions asked in the section "*Inelastic reactions results*" about the *Exfor* entry `41245.022` may find answers thanks to finding the original paper[2] by collaborators[3] and[4]. Although in Russian, the result referred to in the *Exfor* entry can be found in Table 1 in the article. In that table, the result is shown as pertaining to 4 transitions in different isotopes of tungsten, see Figure 70.

0,198	^{186}W		
0,207	^{183}W	0,21±0,03	2523±730
0,215	^{184}W		
0,215	^{186}W		

Figure 70: Excerpt from the Table 1 in[Page 161, 2].

The quoted cross section may therefore be interpreted as a weighted sum of $(\mathrm{n},\ \mathrm{n}'\ \gamma)$ reactions in a natural tungsten sample.

To ensure that, a future work will soon be undertaken in order to verify that hypothesis thanks to *Talys* calculations, and, eventually, the experimental data gathered with *Grapheme*.

[2] A.I.Lashuk, I.P.Sadokhin. Vop.At.Nauki i Tekhn.,Ser.Yaderno-Reak.Konstanty, Issue.2, p.59 (1996), Russia.
[3] Emmeric Dupont (CEA Irfu, Université Paris-Saclay, F-91191 Gif-sur-Yvette, France), private communication.
[4] Naohiko Otsuka (NDS, International Atomic Energy Agency), private communication.

9.5.2 Fit script improvement

The *354 keV* results pointed to deficiencies in the fitting scripts, that are discussed in the main text (*Fit improvements perspectives*).

A little, but efficient, work has been done on the fitting procedure. This update is twofold:

Background guess

The initial guess for the background value was initially done by picking up the value of the first (leftmost) bin in the histogram to fit. This is method is simple, but can be very off in cases where the background is very noisy. The improvement has been made by using the average value of the histogram bins, which is more adapted to cases where the peaks of interest are seating on top of an important background.

The code in Listing 28 shows the change related to the background guess in the fit script.

Listing 28: `diff` display of the change in the `adaptive_fit.py` script in order to improve the background guess.

```
@@ -74,14 +93,19 @@ def adaptive_fit(
#   .. Background is either linear or constant
+    bg_guess = sum(the_histo)/(1.+ len(the_histo))
+    # print(f"# {bg_guess=}")
     if bg_type == "lin":
         functions_to_fit.append(lambda x, B, SL: B + SL * (x - xpeak))
-        variables_guess.extend([the_histo[0].count, 0.0])
+        variables_guess.extend([
+            bg_guess,
+            0.0
+        ])
```

Fall back to single peak fit

In the case of single peak fit (as is the case for the *354 keV transition*), the original fit method was (and still is) falling back on the `pyhisto.tools.GaussRealFit` method that is integrated with the `pyhisto` package.

The `GaussRealFit` method relied on statistical methods (computing averages, standard deviation, ...) to extract first guesses from the histogram to fit. This very simple guessing strategy can be wrong in cases where the histogram is very noisy, or any situation where the mean value, or standard deviation of the peak of interest are very different from the ones computed by simple statistics.

In order to maintain the transmission of value guesses from the *whole range* fit (see *in the analysis flowcharts*), the `pyhisto` module was updated to version *0.1.2* where the `GaussRealFit` can now take as argument external value of mean and standard deviation guesses. Listing Listing 29 shows the update in that code. The corresponding call in `adaptive_fit.py` is updated, as shown in Listing 30.

Listing 29: Changes in the `GaussRealFit` method in the `pyhisto` package, in order to take as call argument initial guesses for the mean and standard deviation values.

```
@@ -43,7 +43,9
class GaussRealFit:
    '''From an histogram, extract Gaussian 'fitted' parameters '''
    def __init__(self,
-                h: Histogram):
+                h: Histogram,
+                _xguess: float = None,
```

(continues on next page)

(continued from previous page)

```
+                 _stdevguess: float = None):
        '''computes sum, average x, stddev, max, bin'''
        self.h1 = h
        _simplefit = GaussFit(self.h1)
@@ -56,13 +586,13
        first_guess = [_simplefit.integral,
-                       _simplefit.mean,
-                       _simplefit.stdev,
-                       self.h1[0].count,
+                       _xguess or _simplefit.mean,
+                       _stdevguess or _simplefit.stdev,
+                       sum(self.h1)/(1. + len(self.h1)), # background guess
                       0.0]
```

Listing 30: Changes in the `adaptive_fit.py` scripts to pass initial guesses for the mean and standard deviation values to the `GaussRealFit` fall back method.

```
@@ -53,7 +68,11 @@ def adaptive_fit(
    # if no parasites,
    if parasites is None or len(parasites) == 0:
        # fall back to realfit
-        the_fit = GaussRealFit(the_histo)
+        the_fit = GaussRealFit(
+            the_histo,
+            _xguess=xpeak,
+            _stdevguess=guessstdev,
+        )
```

Consequences of Fit improvement

As a consequence of the fitting procedure improvement, the 354 keV transition in the inelastic scattering channel now returns much more reasonable values, as shown in Figure 71.

9.5.3 `context` module in task files

A `context` module was added at the top of tasks files. The goal of the the `context` module is to simplify the imports in tasks files and avoid copy-pasted loading of detectors list.

As an example, Listing 31 shows a `diff` of the top of the do_start_new_iter.py file. In the listing, it is clear that the use of the `context` module greatly improves the readability and clarity of the code.

Listing 31: Edits on the `do_start_new_iter.py` file that implement the `context` module. The `from tasks import fn_start_new_iter as fns` lines implements functions of interest for the tasks on the same model than the `context` module (but specific to the tasks file).

```
from doit.tools import create_folder
+from context import the_transitions, Ge_detectors, iter_type
+from tasks import fn_start_new_iter as fns
-from doit.action import CmdAction
```

(continues on next page)

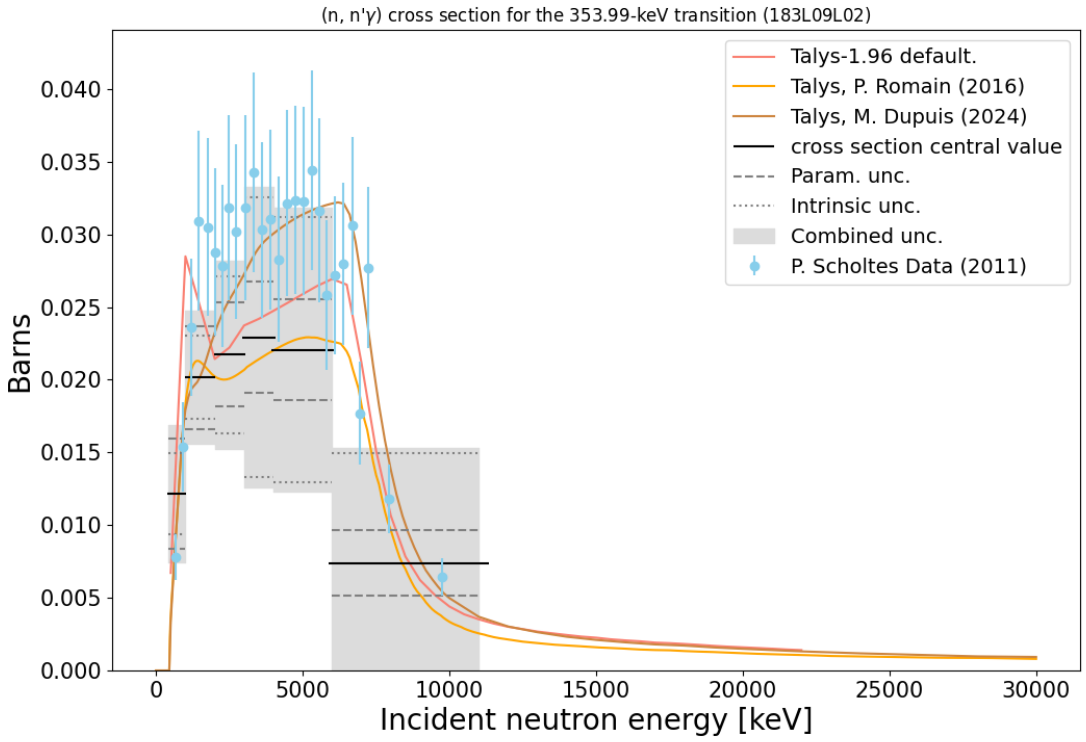


Figure 71: Updated experimental cross section for the 353.9 keV γ ray in the $(\mathrm{n},\ \mathrm{n}')$ reaction on ^{183}W. See Figure 58 for full description and the original (faulty) results.

(continued from previous page)

```
-from doit import get_var
-
-import yaml
-import pathlib
-from itertools import product, chain
-import uuid
-import datetime
-
-from random import gauss as rand_normal
-from random import uniform as rand_flat
-
-# Load mapping and transitions data
-the_mapping = yaml.load(open("etc/mapping.yaml"), Loader=yaml.Loader)
-detectors = the_mapping["detectors"]
-FC_detectors = list(filter(lambda x: x.startswith("U"), detectors))
-Ge_detectors = list(filter(lambda x: not x.startswith("U"), detectors))
-the_transitions = yaml.load(open("etc/transitions_to_look_at.yaml"), Loader=yaml.Loader)
-
-
-# ID of the iteration (load if exists)
-UUID = "000000"
-if pathlib.Path("etc/uuid.txt").exists():
-    UUID = open("etc/uuid.txt", "r").read().strip()
-
-# Determine if it's a Monte Carlo iteration or not
-iter_type = "mc" if get_var("iter", "mc") == "mc" else "center
```

9.5.4 Use of *intermediate data* semi-Monte Carlo code

As explained in *Full Monte Carlo data analysis to produce uncertainties and covariances*, a partial Monte Carlo code (called `resimu`), running on intermediate analysis results, has been developed with F. Claeys during his thesis, in order to apply the Monte Carlo method on previous results, mainly to produce correlation matrices[5,6 and 7].

This other code has a much reduced Monte Carlo loop, that runs in seconds. It has been developed by following some principles of open source: using `git`, relying on configuration files and testing, ... in order to be usable with any data set from *Grapheme*. And indeed, it has been used so far on ^{238}U, ^{233}U, and ^{232}Th data[Page 165, 6].

We can now add ^{183}W to the list. The *plugging in* of `resimu` in the analysis pipeline was very easy, as all the files needed for input already exist (γ count files for each HPGe detector, neutron flux integrated over energy / time-of-flight windows, efficiencies, ...). It was just a matter of gathering the input data, doing a little reformating (order of the data columns in the file, ...) and writing the configuration file (which mostly lists the input files), all of these can be fully automatize.

The result, at the moment only available for the *268 keV transition*) is compared to the one from the full Monte Carlo code in Figure 72. The updated figure shows a very high degree of compatibility between the full Monte Carlo and the semi Monte Carlo code.

[5] Greg Henning, Maëlle Kerveno, Philippe Dessagne, François Claeys, Nicolas Dari Bako, et al.. Using the Monte-Carlo method to analyze experimental data and produce uncertainties and covariances. 15th International Conference on Nuclear Data for Science and Technology (ND2022), Jul 2022, Online Conference, United States. pp.01045, ⟨10.1051/epjconf/202328401045⟩.

[6] "Producing uncertainties and covariance matrix from intermediate data using a Monte-Carlo method." Greg Henning, François Claeys, Nicolas Dari Bako, Philippe Dessagne and Maëlle Kerveno. EPJ Web of Conf., 294 (2024) 05002 https://doi.org/10.1051/epjconf/202429405002

[7] Francois Claeys, "Mesure, modélisation et évaluation de sections efficaces à seuil (n, xn γ) d'intérêt pour les applications de l'énergie nucléaire". Thèse de doctorat, 2023. Chapter 2.4 "Resimulateur" https://theses.fr/2023STRAE027

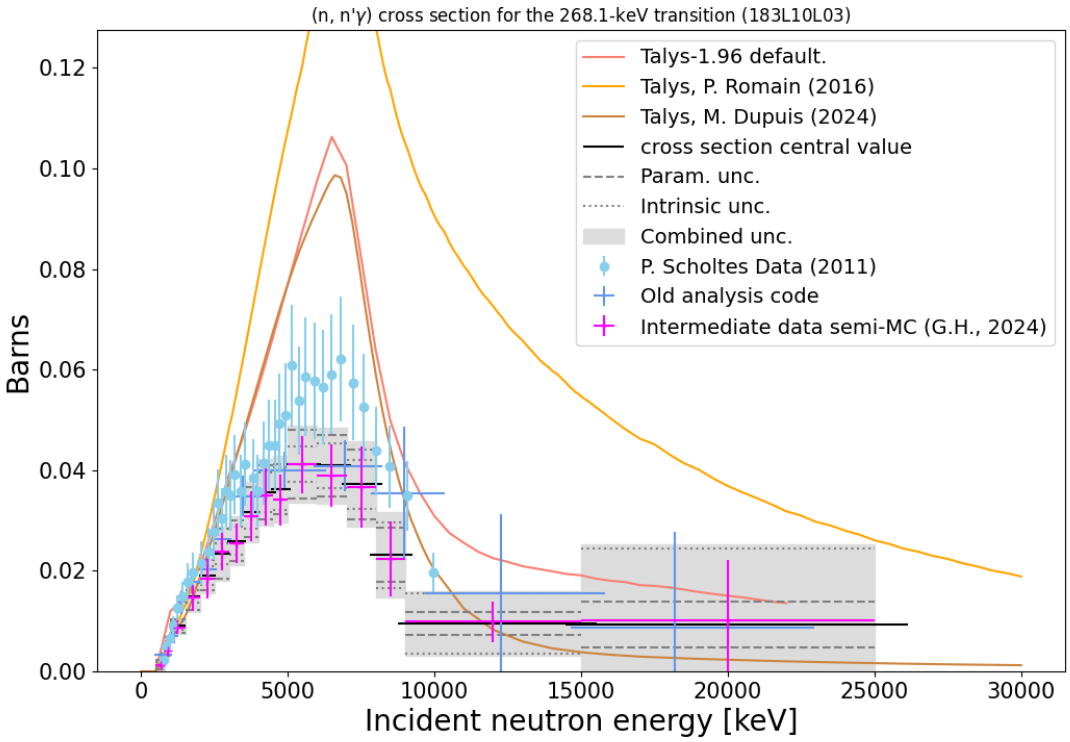


Figure 72: Updated experimental cross section for the 268.1 keV γ ray in the $(\mathrm{n},\ \mathrm{n}')$ reaction on ^{183}W. See Figure 52 for full description.

Figure 73 compares the correlation matrices obtained with both method. Strong similarities can be observed, in particular, the negative correlation at the lowest energies. Still, some differences can be noticed, but this is to be expected, as the correlation matrix strongly depends on the way the results where computed, and the two methods (full or partial MC) have some significant differences in input parameters (in particular on the γ counts and the treatment of their uncertainty and normalization).

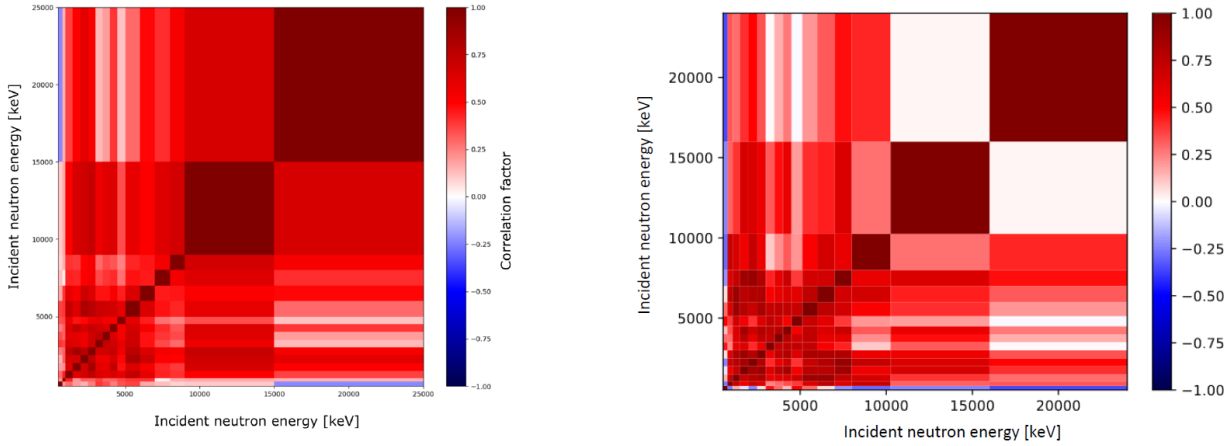


Figure 73: Correlation matrix the the 268.1 keV γ ray in the $(\mathrm{n},\ \mathrm{n}')$ reaction on ^{183}W, from the full Monte Carlo code on the left (Figure 53), and the semi-Monte Carlo analysis on the right.

The ease of implementing the semi-MC procedure, along the high compatibility of the results is encouraging us to make use of this speedy method of analysis in the future, at least in addition and as a way to check the full Monte Carlo procedure.

9.5.5 Other updates

Other updates and changes have been implemented:

Minimization method in fitting procedures
: Changing from the default method `trf` to `dogbox` in the fitting procedure did not significantly improve the speed of the script, or accuracy of the result.

Using last modification *timestamp* instead of file `md5` checksum in *job dependencies*.
: This change did not impact significantly the iteration running time

9.6 Glossary

ANR
: "Agence nationale de la recherche" (anr.fr). A French institution tasked with funding scientific research, mostly through project calls.

CI/CD
: Continuous Integration and Continuous Delivery: software development process aimed at streamlining and accelerating a software development lifecycle. A key component of CI/CD methodology is the use of **pipeline** for compilation, running tests, ... The pipeline philosophy has been used in implementing the new analysis code described in *Implementation of the Monte Carlo analysis*.

CNRS
: Centre national de la recherche scientifique. The French state research organisation.

CoH3
An "optical model and Hauser-Feshbach theory code, which focuses on the nuclear reaction calculations in the keV to tens of MeV region"[1].

Creative Commons
Organization that published a set of Open Access *licenses* (see *Creative Commons*) with different range of reusability.

CSV
The "Comma-separated values" (CSV) format stores tabular data in plain text, using commas to separate values, and newlines to separate records.

CSV description on Wikipedia

CSV files have an extension `.csv` and *MIME type* `text/csv`.

Delco
"Détecteur d'électron de conversion". A conversion electron detector for in beam $(\mathrm{n},\ x\mathrm{n}\ e^{-}_{\mathrm{conv.}})$ cross section studies, developed by the *DNR* group.

Read reference[2] for more information.

DMP
The *Data Management Plan* is a document that describe what data will be collected during an experiment, how, how it will be stored, used, ... The aim is to describe the whole life cycle of the data.

See *DMP (i.e.) What we should have started with*.

DNR
Données Nucléaires pour les Réacteurs The group in IPHC studying $(\mathrm{n},\ x\mathrm{n})$ reactions.

See *Nuclear Data study program at IPHC*.

DOI
A **Digital Object Identifier** is a unique and persistent identifier for a digital document (whether an article, a thesis, a data set ...). *Metadata* are associated to the `DOI`.

Getting a `DOI` for a document is not automatic. For example, documents deposited on *HAL* don't get a `DOI` (they get assigned an *HAL-Id* : `hal-nnnnnn`). University library that publish thesis and HDR manuscripts do not give them a *DOI* either (in general). To get a `DOI` for this manuscript, it was first deposited on zenodo, then the DOI can be referred to in other platforms.

A `DOI` is *persistent*, meaning that if the original repository changes name, is closed, ... the DOI does not change, even if the document is hosted somewhere else. This allows a document to be available from different places while still being clearly identified as the same across all platforms.

Empire
The Empire Nuclear Reaction Code is (according to the website) an easy-to-use tool for the basic research and evaluation of nuclear data. It is a modular system for advanced modeling of nuclear reactions using various theoretical models.

See reference[3] for more information.

[1] Kawano, T. (2021). CoH3: The Coupled-Channels and Hauser-Feshbach Code. In: Escher, J., et al. Compound-Nuclear Reactions. Springer Proceedings in Physics, vol 254. Springer, Cham. https://doi.org/10.1007/978-3-030-58082-7_3

[2] Markus Nyman, Thomas Adam, Catalin Borcea, Marian Boromiza, Philippe Dessagne, *et al.* "New equipment for neutron scattering cross-section measurements at GELINA." ND 2019: International Conference on Nuclear Data for Science and Technology, 2019, Pékin, China. pp.17003, ⟨10.1051/epjconf/202023917003⟩.

[3] M. Herman, R. Capote, B.V. Carlson, P. Oblozinsky, M. Sin, A. Trkov, H. Wienke, V. Zerkin, "EMPIRE: Nuclear Reaction Model Code System for Data Evaluation", Nucl. Data Sheets, 108 (2007) 2655-2. https://doi.org/10.1016/j.nds.2007.11.003

ENDF
"Evaluated Nuclear Data Files", a *file format* for nuclear data evaluation (in particular the ENDF/B series[4 and5]).

The website *ENDF* allow research into evaluations, just as *Exfor* is for research into experimental results.

ENDF database
The "Evaluated Nuclear Data File" database, hosted by the *IAEA* allow research of evaluations values for *any experimental data for a given reaction on a specified isotope*.

Access at: https://www-nds.iaea.org/exfor/endf.htm

Its sister service *Exfor* is dedicated to experimental results.

Exfor
"Experimental Nuclear Reaction Data". A database hosted by the *IAEA*'s Nuclear Data Service where one can search for *any experimental data for a given reaction on a specified isotope*.

Access at: https://www-nds.iaea.org/exfor/

Exfor sister service is *ENDF*, a website to search into evaluations.

Fair
The *Fair* principles are at the core of *Open Science* and stands for Findability, Accessibility, Interoperability, and Reuse.

Faster
Fast Acquisition System for nuclEar Research, https://faster.in2p3.fr/. The acquisition system currently used in *our* experiments.

gamma flash
Strong flash of γ rays coming from the neutron production source, in particular at *Gelina*. At *NFS*, the γ flash is much weaker.

In our data, its presence allows an accurate *time calibration* of the events, as well as a reference for the correction of γ ray energy dependent *time shifts* in the detectors. See also Figure 13 for an illustration.

Ganil
Grand accelerateur National d'ions lourds. https://www.ganil-spiral2.eu/en/ In Caen, France.

Geant4
A "toolkit for the simulation of the passage of particles through matter. Its areas of application include high energy, nuclear and accelerator physics, as well as studies in medical and space science."

See more information on the website.

Gelina
The *JRC*'s Neutron Time-of-Flight Facility

GNDS
"Generalised Nuclear Database Structure", a *file format* for nuclear data evaluation, intended to replace *ENDF*.

Specifications of the format are available on the *NEA* website.

Grapheme
GeRmanium array for Actinides PrEcise MEasurements Experimental setup of the *DNR* group, housed at the *Gelina* facility.

See Figure 3 in *Nuclear Data study program at IPHC* and *The Grapheme setup*.

[4] ENDF/B-VIII.0 Evaluated Nuclear Data Library https://www.nndc.bnl.gov/endf-b8.0/

[5] ENDF/B-VIII.0: The 8th Major Release of the Nuclear Reaction Data Library with CIELO-project Cross Sections, New Standards and Thermal Scattering Data, D.A. Brown, M.B. Chadwick, R. Capote, A.C. Kahler, A. Trkov, M.W. Herman, A.A. Sonzogni, et al. Nuclear Data Sheets, 148 (2018) 1-142 https://doi.org/10.1016/j.nds.2018.02.001.

HAL
"*Hyper article en ligne*" French platform for publication in Open Access https://hal.science/. However, HAL does not provide a *DOI* to documents deposited on it.

HTML
HyperText Markup Language (HTML) is the standard markup language for documents designed to be displayed in a web browser.

HTML files have an extension `.html` or `.htm` and *MIME type* `text/html`.

IAEA
The "International Atomic Energy Agency" is an intergovernmental organization that seeks to promote cooperation in the nuclear field.

IDE
Integrated Development Environment. A text editor that comes with code styling, debugging abilities, ...

VS Code and *Jupyter Lab* are two example of IDEs.

Ifin-HH
Horia Hulubei National Institute for R&D in Physics and Nuclear Engineering, Bucarest, Romania. https://www.nipne.ro/

IN2P3
Institut national de physique nucléaire et de physique des particules, now *aka* "CNRS Nucléaire & Particules"

The insitute of *CNRS* overseeing the study of nuclear and particule physics.

IPHC
Institut pluridisciplinaire Hubert Curien. *CNRS* and *Université de Strasbourg Laboratory*.

Jeff
Joint Evaluated Fission and Fusion (Jeff) Nuclear Data Library. An evaluation from the *NEA*.

More information is available on the project webpage https://www.oecd-nea.org/dbdata/jeff.

JRC-Geel
Joint Research Center, Geel, Belgium. one of the six scientific sites of the European Commission's Joint Research Centre

Json
"Json (JavaScript Object Notation) is a lightweight data-interchange format. It is easy for humans to read and write." (https://www.json.org)

The Json format is interchangeable with *Yaml*, and most of the time, equivalent to a *Python* dictionary.

Json files have an extension `.json` and *MIME type* `application/json`.

License
A legal text that defines the condition of use and reuse of a document, file, data set, article, software, ... (See *Distribution licenses*.)

MCNP
"The MCNP®, Monte Carlo N-Particle, code can be used for general-purpose transport of many particles including neutrons, photons, electrons, ions, and many other elementary particles, up to 1 TeV/nucleon."

More information online.

metadata
Metadata are data associated with a document. It usually includes the name and affiliation of the authors, a description of the content, ...

(See *Metadata, metadata, metadata*).

NEA
Nuclear Energy Agency, an intergovernmental agency of the OECD focused on nuclear safety, technology, and science.

NFS
"Neutron For science". A neutron beam facility at *Ganil*[6].

PDF
The Portable Document Format (PDF), is a file format developed by Adobe in 1992 to produce documents that may include formatted text and images, in a manner independent of application software, or operating systems.

PDF files have an extension `.pdf` and *MIME type* `application/pdf`.

postprint
Also called *Author Accepted Manuscript* (AAM). It is the final *author* version of their paper (i.e. the one accepted by the journal, but processed by the author). It does not include the editor formatting, logo, ... This is the version that can be published in *Open Access* (after an embargo period) for public funded research in France.

preprint
The *preprint* is a version of an article manuscript before peer-review. It can be published in *Open Access* on some platforms (such as *HAL* or arxiv) before being submitted to a journal.

Recherche Data Gouv
Recherche Data Gouv (https://recherche.data.gouv.fr) is a French Government funded repository to deposit *Open Data*.

Some data from the *DNR* group (see *Case Study: Test data recorded with an LaBr3 detector*) have already been deposited on the platform: https://entrepot.recherche.data.gouv.fr/dataverse/dnr.

Root
Root is an open-source data analysis framework developed by CERN that is widely used in nuclear physics and high energy particle physics.

Root files have an extension `.root`. No official *MIME type* is currently recognized, and the `.root` files are usually identified as the default `application/octet-stream`. Types `application/x-root` or `application/x+cern-root` have been proposed, but are not registered as of today.

Talys
Talys is a nuclear reaction code for the prediction of reaction cross sections and related quantities. (See *Talys*.)

Unistra
Université de Strasbourg. University of Strasbourg.

URL
A Uniform Resource Locator (URL), also known as a web address on the Web, is a standardized string of characters specifying the *location* of a *resource* (file, service, ...) on a network and how to access it.

XML
The "Extensible Markup Language" (XML) is a markup language and file format for storing organized data. It can be used to store *metadata*, in particular, when using the DataCite Metadata scheme, as seen in *Case Study: Metadata for this manuscript*.

Full documentation is available at https://www.w3.org/XML/

XML files have an extension `.xml` and *MIME type* `text/xml` or `application/xml`.

Zenodo
Zenodo (https://zenodo.org/) is a general-purpose open repository developed under the European OpenAIRE program and operated by CERN.

[6] "The neutrons for science facility at SPIRAL-2" X. Ledoux, et al. EPJ Web Conf., 146 (2017) 03003 DOI: https://doi.org/10.1051/epjconf/201714603003

Code and Data availability Statement

The analysis code described in this manuscript is available publicly, under the CeCILL-C FREE SOFTWARE LICENSE AGREEMENT, on the University of Strasbourg gitlab.

Other scripts and softwares of interest can be found in the extra section of the manuscript repository, and are distributed openly (check repository for specific license information).

Some data used in the analysis is made available in the analysis repository.

The results from the analysis presented here will be made available publicly on the *Recherche Data Gouv* platform, alongside previous datasets from the same setup.

Acknowledgments

The work presented in this document was supported in parts by the CNRS multipartner NEEDS program, PACEN/GEDEPEON, and by the European Commission within the Sixth Framework Program through I3-EFNUDAT (EURATOM contract n° 036434) and NUDAME (Contract FP6-516487), within the Seventh Framework Program through EUFRAT (EURATOM contract n° FP7-211499), through ANDES (EURATOM contract n° FP7-249671), from the Euratom research and training program 2014-2018 under grant agreement n° 847594 (ARIEL) and under grant agreement n° 847552 (SANDA).

You made it to the end. Congratulations!