gamma_flow: **G**uided **A**nalysis of **M**ulti-label spectra by **Ma**trix **F**actorization for **L**ightweight **O**perational **W**orkflows

Viola Rädle[a,*], Tilman Hartwig[a], Benjamin Oesen[a], Emily Alice Kröger[b], Julius Vogt[b], Eike Gericke[b], Martin Baron[b]

[a] *Application Lab for AI and Big Data, German Environmental Agency, Leipzig, Germany*
[b] *Federal Office for Radiation Protection, Berlin, Germany*

Abstract

gamma_flow is an open-source Python package for real-time analysis of spectral data. It supports classification, denoising, decomposition, and outlier detection of both single- and multi-component spectra. Instead of relying on large, computationally intensive models, it employs a novel supervised approach to non-negative matrix factorization (NMF) for dimensionality reduction. This ensures a fast, efficient, and adaptable analysis while reducing computational costs. **gamma_flow** achieves classification accuracies above 90% and enables reliable automated spectral interpretation. Originally developed for gamma-ray spectra, it is applicable to any type of one-dimensional spectral data. As an open and flexible alternative to proprietary software, it supports various applications in research and industry.

Keywords: Python, Gamma spectroscopy, Non-negative Matrix Factorization, Classification, Denoising, Spectral Deconvolution

Metadata

The ancillary data table 1 is required for the sub-version of the codebase. Please replace the italicized text in the right column with the correct information about your current code and leave the left column untouched.
Optionally, you can provide information about the current executable software version filling in the left column of Table 2. Please leave the first column as it is. FRAGE: Welche Tabelle sollen wir ausfüllen?

*Corresponding author: raedle.htwk@web.de

Nr.	Code metadata description	Metadata
TO DO C1	Current code version	For example v42
C2	Permanent link to code/repository used for this code version	`https://gitlab.opencode.de/uba-ki-lab/gamma_flow`
C3	Permanent link to Reproducible Capsule	TO DO For example: `https://codeocean.com/capsule/0270963/tree/v1`
C4	Legal Code License	BSD 3-Clause "New" or "Revised" License
C5	Code versioning system used	git
C6	Software code languages, tools, and services used	Python
C7	Compilation requirements, operating environments & dependencies	TO DO. Jupyter?
C8	If available Link to developer documentation/manual	`https://gitlab.opencode.de/uba-ki-lab/gamma_flow/-/blob/main/README.md?ref_type=heads`
C9	Support email for questions	raedle.htwk@web.de

Table 1: Code metadata (mandatory)

1. Motivation and significance

In this section, we want you to introduce the scientific background and the motivation for developing the software.

- *Explain why the software is important and describe the exact (scientific) problem(s) it solves.*
- *Indicate in what way the software has contributed (or will contribute in the future) to the process of scientific discovery; if available, please cite a research paper using the software.*
- *Provide a description of the experimental setting. (How does the user use the software?)*
- *Introduce related work in literature (cite or list algorithms used, other software etc.).*

übernommen aus JOSS-Summary (part 1) Most radioactive sources can be identified by measuring their emitted radiation (X-rays and gamma

Nr.	(Executable) software metadata description	Please fill in this column
S1	Current software version	For example 1.1, 2.4 etc.
S2	Permanent link to executables of this version	For example: `https://github.com/combogenomics/DuctApe/releases/tag/DuctApe-0.16.4`
S3	Permanent link to Reproducible Capsule	
S4	Legal Software License	List one of the approved licenses
S5	Computing platforms/Operating Systems	For example Android, BSD, iOS, Linux, OS X, Microsoft Windows, Unix-like , IBM z/OS, distributed/web based etc.
S6	Installation requirements & dependencies	
S7	If available, link to user manual - if formally published include a reference to the publication in the reference list	For example: `http://mozart.github.io/documentation/`
S8	Support email for questions	

Table 2: Software metadata (optional)

rays), and visualizing them as a spectrum. In nuclear security applications, the resulting gamma spectra have to be analyzed in real-time as immediate reaction and decision making may be required. However, the manual recognition of isotopes present in a spectrum constitutes a strenous, error-prone task that depends upon expert knowledge. Hence, this raises the need for algorithms assisting in the initial categorization and recognizability of measured gamma spectra. The delineated use case brings along several requirements:

- As mobile, room temperature detectors are often deployed in nuclear security applications, the produced spectra typically exhibit a rather low energy resolution. In addition, a high temporal resolution is required (usually around one spectrum per second), leading to a low acquisition time and a low signal-to-noise ratio. Hence, the model must be robust and be able to handle noisy data.

- For some radioactive sources, acquisition of training spectra may be challenging. Instead, spectra of those isotopes are simulated using Monte Carlo N-Particle (MCNP) code [1]. In this process, energy

deposition in a detector material is simulated, yielding spectra that can be used for model training. However, simulated spectra and measured spectra from real-world sources may differ, which may be a constraint for model performance. On this account, preliminary data exploration is crucial to assess the similarity of spectral data from different detectors and to evaluate potential data limitations.

- Lastly, not only the correct classification of single-label test spectra (stemming from one isotope) is necessary, but also the decomposition of linear combinations of various isotopes (multi-label spectra). Hence, classification approaches like k-nearest-neighbours that solely depend on the similarity between training and test spectra are not applicable.

This paper presents **gamma_flow**, a python package that includes the

- classification of test spectra to predict their constituents
- denoising of test spectra for better recognizability
- outlier detection to evaluate the model's applicability to test spectra

To Do: kurz und schwammig beschreiben, was noch toll ist.

2. Software description

Describe the software. Provide enough detail to help the reader understand its impact. **gamma_flow** is based on a dimensionality reduction model that constitutes a novel, supervised approach to non-negative matrix factorization (NMF). More explicitly, the spectral data matrix is decomposed into the product of two low-rank matrices denoted as the scores (spectral data in latent space) and the loadings (transformation matrix or latent components). The loadings matrix is predefined and consists of the mean spectra of the training isotopes. Hence, by design, the scores axes correspond to the share of an isotope in a spectrum, resulting in an interpretable latent space. As a result, the classification of a test spectrum can be read directly from its (normalized) scores. In particular, shares of individual isotopes in a multi-label spectrum can be identified. This leads to an explainable quantitative prediction of the spectral constituents. The scores can be transformed back into spectral space by applying the inverse model. This inverse transformation rids the test spectrum of noise and results in a smooth, easily recognizable denoised spectrum. If a test spectrum of an isotope is unknown to the model (i.e. this isotope was not included in model training), it can still be projected into latent space. However, when the latent space information (scores) are

decompressed, the resulting denoised spectrum does not resemble the original spectrum any more. Some original features may not be captured while new peaks may have been fabricated. This can be quantified by calculating the cosine similarity between the original and the denoised spectrum, which can serve as an indicator of a test spectrum to be an outlier.

2.1. Software architecture

Give a short overview of the overall software architecture; provide a pictorial overview where possible; for example, an image showing the components. If necessary, provide implementation details. Soll hier ein Schaubild erstellt werden von jupyter-files und auf welche python-files sie zugreifen?

2.2. Software functionalities

Present the major functionalities of the software. Einteilung dieses Abschnitts:

2.2.1. Preprocessing of the spectra
2.2.2. Model training (inkl. NMF)
2.2.3. Model application: Classification and Decomposition
2.2.4. Model application: Denoising
2.2.5. Model application: Outlier detection
Bzw. hier einfügen: alles aus JOSS - Methodology and structure

2.3. Sample code snippets analysis (optional)

Würde ich eher weglassen? Ggf. Das Modell vorstellen, aber eher nicht

3. Illustrative examples

Provide at least one illustrative example to demonstrate the major functions of your software/code.
***Optional**: you may include one explanatory video or screencast that will appear next to your article, in the right hand side panel. Please upload any video as a single supplementary file with your article. Only one MP4 formatted, with 150MB maximum size, video is possible per article. Recommended video dimensions are 640 x 480 at a maximum of 30 frames / second. Prior to submission please test and validate your .mp4 file at* `http://elsevier-apps.sciverse.com/GadgetVideoPodcastPlayerWeb/verification` *. This tool will display your video exactly in the same way as it will appear on ScienceDirect.*

Plots:

- ggf. Loadings

- ggf. Scores (different detector, single-label class.)
- Confusion matrix (different detector, single-label class.)
- Denoised spectrum
- Outlier: Feature importance (decision tree)

Problem: Diese Figures wollten wir eigentlich erst im wiss. Paper bringen! Alternative: Nur graphical abstract? Wirkt auf mich zu wenig für diese Section...

4. Impact

This is the main section of the article and reviewers will weight it appropriately. Please indicate:

- *Any new research questions that can be pursued as a result of your software.*
- *In what way, and to what extent, your software improves the pursuit of existing research questions.*
- *Any ways in which your software has changed the daily practice of its users.*
- *How widespread the use of the software is within and outside the intended user group (downloads, number of users if your software is a service, citable publications, etc.).*
- *How the software is being used in commercial settings and/or how it has led to the creation of spin-off companies.*

Please note that points 1 and 2 are best demonstrated by references to citable publications.
übernommen aus JOSS-Statement of need: In many research fields, spectral measurements help to assess material properties. In this context, an area of interest for many researchers is the classification (automated labelling) of the measured spectra. Proprietary spectral analysis software, however, are often limited in their functionality and adaptability [2, 3]. In addition, the underlying mechanisms are usually not revealed and may act as a black-box system to the user [4]. On top of that, a spectral comparison is typically only possible for spectra of pure substances [5]. However, there may be a need to decompound multi-label spectra (linear combinations of different substances)

and identify their constituents.
gamma_flow is a Python package that can assist researchers in the classification, denoising and outlier detection of spectra. It includes data preprocessing, data exploration, model training and testing as well as an exploratory section on outlier detection. Making use of matrix decomposition methods, the designed model is lean and performant. Training and inference do not require special hardware or extensive computational power. This allows real-time application on ordinary laboratory computers and easy implementation into the measurement routine. The provided example dataset contains gamma spectra of several measured and simulated isotopes as well as pure background spectra. While this package was developed in need of an analysis tool for gamma spectra, it is suitable for any one-dimensional spectra. Examplary applications encompass

- **Infrared spectroscopy** for the assessment of the polymer composition of microplastics in water [6, 7]
- **mass spectrometry** for protein identification in snake venom [8, 9]
- **Raman spectroscopy** for analysis of complex pharmaceutical mixtures and detection of dilution products like lactose [10]
- **UV-Vis spectroscopy** for detection of pesticides in surface waters [11, 12]
- **stellar spectroscopy** to infer the chemical composition of stars [13]

5. Conclusions

... to do

Acknowledgements

Optional. You can use this section to acknowledge colleagues who don't qualify as a co-author but helped you in some way. to do (s. Joss)

References

[1] J. A. Kulesza, T. R. Adams, J. C. Armstrong, S. R. Bolding, F. B. Brown, J. S. Bull, T. P. Burke, A. R. Clark, R. A. A. Forster III, J. F. Giron, T. S. Grieve, C. J. Josey, R. L. Martz, G. W. McKinney, E. J. Pearson, M. E. Rising, C. J. C. Solomon Jr., S. Swaminarayan,

T. J. Trahan, S. C. Wilson, A. J. Zukaitis, MCNP® Code Version 6.3.0 Theory & User Manual, Tech. Rep. LA-UR-22-30006, Los Alamos National Laboratory (LANL), Los Alamos, NM (United States) (Sep. 2022). doi:10.2172/1889957.

[2] H. Lam, Building and Searching Tandem Mass Spectral Libraries for Peptide Identification, Molecular & Cellular Proteomics 10 (12) (Dec. 2011). doi:10.1074/mcp.R111.008565.

[3] J. Nasereddin, M. Shakib, Ira: A free and open-source Fourier transform infrared (FTIR) data analysis widget for pharmaceutical applications, Analytical Letters 56 (16) (2023) 2637–2648. doi:10.1080/00032719.2023.2180516.

[4] L. El Amri, A. Chetaine, H. Amsil, B. El Mokhtari, H. Bounouira, A. Didi, A. Benchrif, K. Laraki, H. Marah, New open-source software for gamma-ray spectra analysis, Applied Radiation and Isotopes 185 (2022) 110227. doi:10.1016/j.apradiso.2022.110227.

[5] W. Cowger, Z. Steinmetz, A. Gray, K. Munno, J. Lynch, H. Hapich, S. Primpke, H. De Frond, C. Rochman, O. Herodotou, Microplastic Spectral Classification Needs an Open Source Community: Open Specy to the Rescue!, Analytical Chemistry 93 (21) (2021) 7543–7548. doi:10.1021/acs.analchem.1c00123.

[6] B. Ferreiro, R. Leardi, E. Farinini, J. M. Andrade, Supervised classification combined with genetic algorithm variable selection for a fast identification of polymeric microdebris using infrared reflectance, Marine Pollution Bulletin 195 (2023) 115540. doi:10.1016/j.marpolbul.2023.115540.

[7] Q. T. Whiting, K. F. O'Connor, P. M. Potter, S. R. Al-Abed, A high-throughput, automated technique for microplastics detection, quantification, and characterization in surface waters using laser direct infrared spectroscopy, Analytical and Bioanalytical Chemistry 414 (29) (2022) 8353–8364. doi:10.1007/s00216-022-04371-2.

[8] A. Zelanis, D. A. Silva, E. S. Kitano, T. Liberato, I. Fukushima, S. M. T. Serrano, A. K. Tashima, A first step towards building spectral libraries as complementary tools for snake venom proteome/peptidome studies, Comparative Biochemistry and Physiology Part D: Genomics and Proteomics 31 (2019) 100599. doi:10.1016/j.cbd.2019.100599.

[9] N. Ç. Yasemin, A. Izzet, K. M. Ali, K. Selçuk, S. Bekir, Identification of Snake Venoms According to their Protein Content Using the MALDI-TOF-MS Method, Analytical Chemistry Letters 11 (2) (2021) 153–167. doi:10.1080/22297928.2021.1894974.

[10] X. Fu, L.-m. Zhong, Y.-b. Cao, H. Chen, F. Lu, Quantitative analysis of excipient dominated drug formulations by Raman spectroscopy combined with deep learning, Analytical Methods 13 (1) (2021) 64–68. doi:10.1039/D0AY01874K.

[11] Y. Guo, C. Liu, R. Ye, Q. Duan, Advances on Water Quality Detection by UV-Vis Spectroscopy, Applied Sciences 10 (19) (2020) 6874. doi:10.3390/app10196874.

[12] X. Qi, Y. Lian, L. Xie, Y. Wang, Z. Lu, Water quality detection based on UV-Vis and NIR spectroscopy: A review, Applied Spectroscopy Reviews 59 (8) (2024) 1036–1060. doi:10.1080/05704928.2023.2294458.

[13] D. F. Gray, The Observation and Analysis of Stellar Photospheres, https://www.cambridge.org/highereducation/books/the-observation-and-analysis-of-stellar-photospheres/67B340445C56F4421BCBA0AFFAAFDEE0 (Dec. 2021). doi:10.1017/9781009082136.

References

[1] Use this style of ordering. References in-text should also use a similar style.

If the software repository you used supplied a DOI or another Persistent IDentifier (PID), please add a reference for your software here. For more guidance on software citation, please see our guide for authors or this article on the essentials of software citation by FORCE 11, of which Elsevier is a member.

Reminder: Before you submit, please delete all the instructions in this document, including this paragraph. Thank you!