The Journal of Open Source Software

# GAMMA_FLOW: **G**uided **A**nalysis of **M**ulti-label spectra by **M**atrix factorization for **L**ightweight **O**perational **W**orkflows

**Viola Rädle** 1*¶, **Tilman Hartwig** 1*, **Benjamin Oesen**1, **Julius Vogt** 2, **Eike Gericke** 2, **Emily Alice Kröger** 2, and **Martin Baron** 2

**1** Application Lab for AI and Big Data, German Environmental Agency, Leipzig, Germany **2** Federal Office for Radiation Protection, Berlin, Germany ¶ Corresponding author * These authors contributed equally.

**DOI:** 10.xxxxxx/draft

**Software**

- Review
- Repository
- Archive

**Editor:** Open Journals

**Reviewers:**

- @openjournals

**Submitted:** 01 January 1970
**Published:** unpublished

## Summary

Radioactive sources can be identified by measuring their emitted radiation (X-rays and gamma 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 `[@Kulesza:2022]`. 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. - At last, 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

It 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.

# 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 `[@Lam:2011; @Naseredding:2023]`. In addition, the underlying mechanisms are usually not revealed and may act as a black-box system to the user `[@El Amri:2022]`. On top of that, a spectral comparison is typically only possible for spectra of pure substances `[@Cowger:2021]`. 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 `[@Ferreiro:2023; @Whiting:2022]` - mass spectrometry for protein identification in snake venom `[@Zelanis:2019; @Yasemin:2021]` - raman spectroscopy for analysis of complex pharmaceutical mixtures and detection of dilution products like lactose `[@Fu:2021]` - UV-Vis spectroscopy for detection of pesticides in surface waters `[@Guo:2020; @Qi:2024]` - stellar spectroscopy to infer the chemical composition of stars `[@Gray:2021]`

# Methodology and structure

This python package consists of three jupyter notebooks that are executed consecutively. In this section, their functionality and is outlined, with an emphasis on the mathematical struction of the model.

## Preprocessing and data exploration

The notebook `01_preprocessing.ipynb` synchronizes spectral data and provides a framework of visualizations for data exploration. All functions called in this notebook are found in `tools_preprocessing.py`.

During preprocessing, the following steps are performed:
- Spectral data files are converted from .xlsm/.spe data to .npy format and saved. - Spectra of different energy calibrations are rebinned to a standard energy calibration. - Spectral data are aggregated by label classes and detectors. Thus, it is possible to collect data from different files and formats. - Optional: The spectra per isotope are limited to a maximum number. - The preprocessed spectra are saved as .npy files.

Data exploration involves the following visualizations: - For each label class (e.g. for each isotope), the mean spectra are calculated detector-wise and compared quantitatively by the cosine similarity. - For each label class, example spectra are chosen randomly and plotted to provide an overview over the data. - The cosine similarity is calculated and visualized as a matrix for all label classes and detectors. This helps to assess whether the model can handle spectra from different detectors.

## Model training and testing

The notebook `02_model.ipynb` trains and tests a dimensionality reduction model that allows for denoising, classification and outlier detection of test spectra. All functions called in this notebook are found in `tools_model.py`.

The dimensionality reduction model presented in this paper comprises a matrix decomposition of spectral data. More precisely, the original spectra matrix $X$ is reconstructed by two low-rank matrices $S$ and $L$:

$$X \approx SL^T$$

with S: scores matrix (spectra in latent space) L: loadings matrix (transformation matrix or latent components)

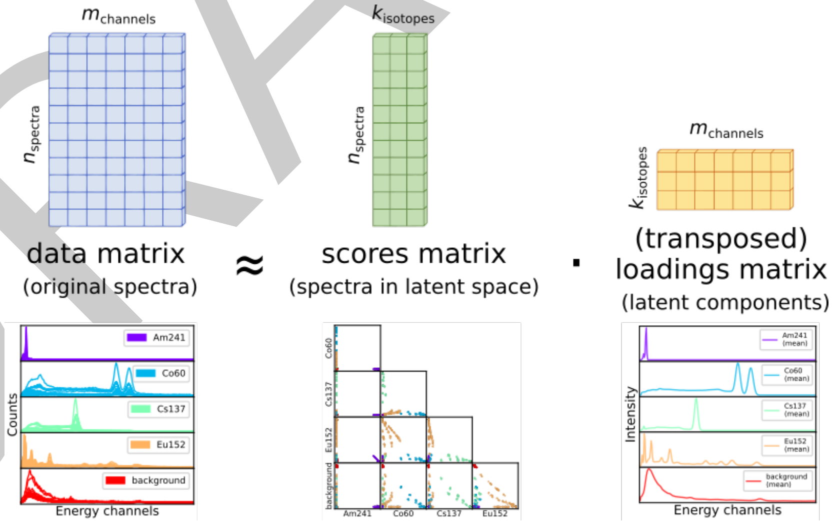

**Figure 1:** Matrix decomposition of spectral data.

As illustrated in Figure Figure 1, original spectral data can be compressed into $k_{\text{isotopes}}$ dimensions. To ensure a conclusive assignment of the latent space axes to the isotopes (i.e. one axis stands for of one isotope), the loadings matrix is predefined as the mean spectra of the $k_{\text{isotopes}}$ isotopes.

During model training, mean spectra for all isotopes are calculated. The scores are then derived by non-negative least squares fit of the original spectra to the loadings matrix. Thus, the components of the normalized scores vectors directly reveal the contributions of the individual isotopes. Denoised spectra, on the other hand, are computed by transforming the non-normalized scores back into spectral space (i.e. by multiplication of with the loadings matrix).

In mathematical terms, this model represents a 'supervised' approach to Non-negative Matrix Factorization (NMF) [@Shreeves:2020; @Bilton:2019]. While dimensionality reduction is conventionally an unsupervised task as it only considers data structure [@Olaya:2022], our approach integrates labels in model training. This leads to an interpretable latent space and obviates the need for an additional classification step. While other supervised NMF approaches incorporate classification loss in model training [@Leuschner:2019; @Lee:2010; @Bisot:2016], our model focuses on a comprehensible construction of the latent space.

The model is trained using spectral data from the specified detectors `dets_tr` and isotopes `isotopes_tr`. Subsequently, it is inferenced (i.e. scores are calculated) on three different test datasets: 1. validation data/holdout data from same detector as used in training (each spectrum including only one isotope or pure background) 2. test data from different detector (each spectrum including one isotope and background) 3. multi-label test data from different detector (each spectrum including multiple isotopes and background)

For all test datasets, spectra are classified and denoised. The results are visualized as - confusion matrix - misclassified spectra - denoised example spectrum - misclassification statistics - scores as scatter matrix - mean scores as bar plot This helps to assess model performance with respect to classification and denoising.

### Outlier analysis

The notebook `03_outlier.ipynb` provides an exploratory approach to outliers detection, i.e. to identify spectra from isotopes that were not used in model training. All functions called in this notebook are found in `tools_outlier.py`.

To simulate outlier spectra, a mock dataset is generated by training a model after removing one specific isotope. The trained model is then inferenced on spectra of this unknown isotope to investigate its behaviour with outliers. First, the resulting latent space distribution and further meta data are analyzed to distinguish known from unknown spectra. Using a decision tree, the most informative feature is identified. Next, a decision boundary is derived for this feature, by a) using the condition of the first split in the decision tree b) fitting a logistic regression (sigmoid function) to the data
c) setting a manual threshold by considering accuracy, precision and recall of outlier identification. The derived decision boundary can then be implemented in the measurement pipeline by the user.

## Acknowledgements

We gratefully acknowledge the support provided by the Federal Ministry for the Environment, Nature Conservation and Nuclear Safety (BMUV), whose funding has been instrumental in enabling us to achieve our research objectives and explore new directions. We also extend our appreciation to Martin Bussick in his function as the AI coordinator. Additionally, we thank the entire AI-Lab team for their support and inspiration, with special recognition to Ruth Brodte for guidance on legal and licensing matters.

## References