

# TriMEph User manual

Filip pomajbo

October 2023

## 1 Introduction

The *TriMEph* is an object-oriented software designed to compute the Mössbauer factor in dependence of temperature for materials using input parameters specified in a `phonopy.yaml` file. The Mössbauer effect plays a crucial role in understanding the interaction of gamma rays with atomic nuclei, and the Mössbauer factor is a key parameter in describing this interaction.

### Key Features:

1. **Object-Oriented Design:** The software is built using an object-oriented paradigm, enhancing modularity and code organization. Objects encapsulate data and methods, providing a scalable and maintainable structure.
2. **Input from `phonopy.yaml`:** The software takes input from a `phonopy.yaml` file, a standard format used in phonon calculations. This enables users to seamlessly integrate Mössbauer factor calculations into existing phonon analysis workflows.
3. **Mössbauer Factor Computation:** Utilizing algorithms specific to Mössbauer spectroscopy, the software performs calculations based on the provided crystal structure, atomic positions, and other relevant parameters.
4. **User-Friendly Interface:** The software is designed with a user-friendly interface, allowing researchers and scientists to easily incorporate Mössbauer factor calculations into their studies.
5. **Quasi-Harmonic/Harmonic Approximation** The calculation strength of this software is that it can both compute Harmonic and Quasi-harmonic approximations in calculating the Mössbauer temperature dependence. While the Harmonic approximation is useful the Quasi-harmonic approximation is much more accurate and produces better results.

### Applications:

- **Materials Science:** Gain insights into the behavior of materials at the atomic level, crucial for understanding magnetic and structural properties.
- **Research:** Facilitate Mössbauer spectroscopy studies by automating and streamlining the computation of Mössbauer factors.

The Mössbauer Factor Calculator is a powerful tool for researchers and scientists working in the field of materials science and nuclear physics, providing an efficient and accurate way to compute Mössbauer factors from `phonopy.yaml` input.

## 2 User interface

We begin by examining the initial four graphical windows located on the left side of the interface. These windows facilitate the uploading of files necessary for processing the Mössbauer factor and the mean square displacement in the  $x^2$ ,  $y^2$ , and  $z^2$  dimensions.

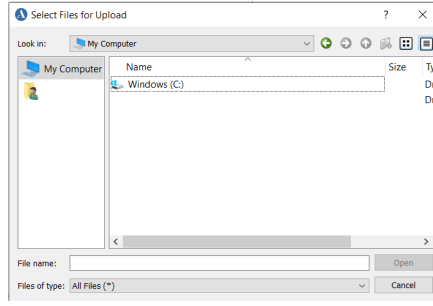


Figure 1: Uploading and saving window

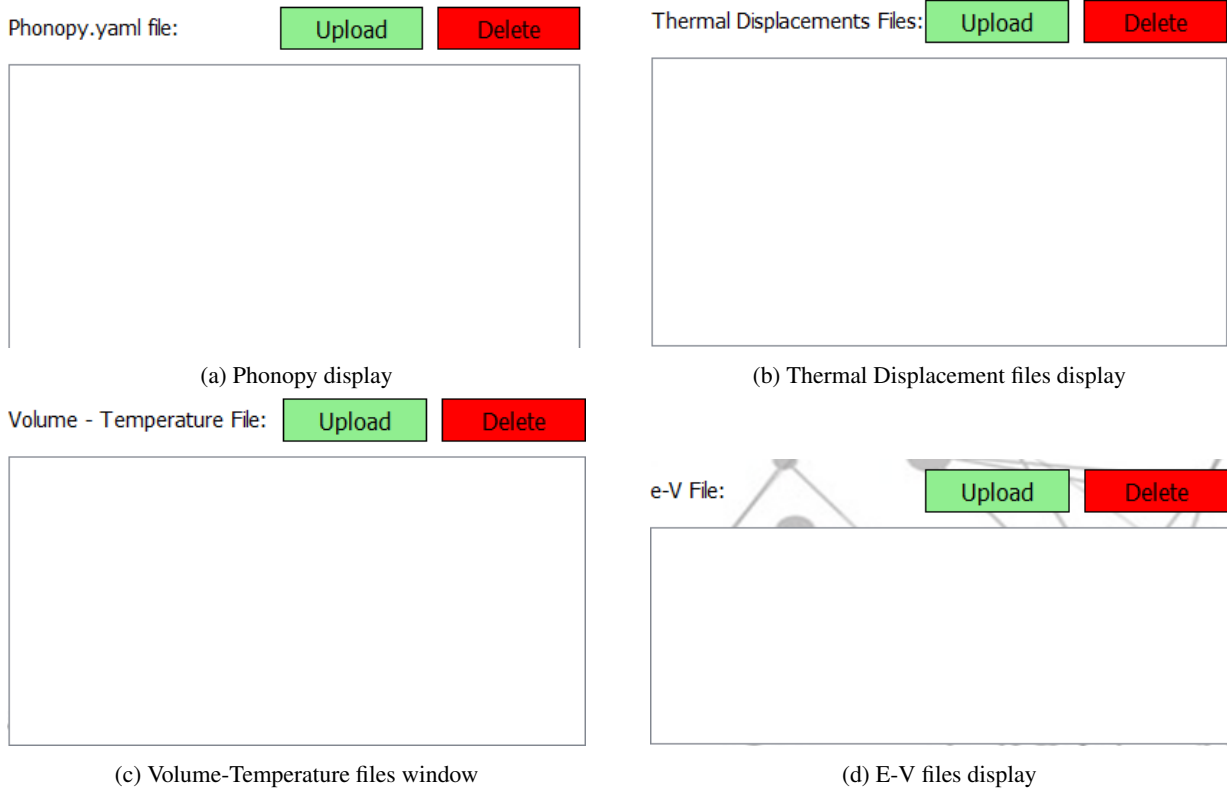


Figure 2: Graphic windows for files used in computations

## 2.1 Uploading and deleting files

The Phonopy upload button (see Figure 1) is used to upload your `phonopy.yaml` file. Note that this software is designed so that the data must be in the correct format, as detailed later in this document. When pressed, an Upload dialog window will appear, allowing you to choose your `phonopy.yaml` file and press open. This will load the file into the backend of this software for further processing and display the path to the file in the graphical window (see Figure 1).

Apply the same procedure for your e-V file and Volume-Temperature file.

As for your Thermal displacement files, this depends on the function of the software.

To delete these files, press the Delete button. This will automatically delete all files loaded in the corresponding graphic window. If there are multiple files, such as in the case of the Thermal displacement files, you can either delete all of them by pressing Delete, or select specific files. Selected files will change their color to red; clicking Delete will remove all selected files. To undo the selection, click the file again so that it is not red.

The final uploading function is the Input Experimental Data button. This button enables you to select a specific file in either the `.dat` or `.tex` format. The selected file will be used to input experimental data into the software. This function automatically loads the data into the backend, and once Graph is clicked after loading these data, it will display points on the created graphs from your data. The Clear Experimental Data button is used for clearing these data from your graph.

In Figure 3, you can see a graph with experimental data visualized.

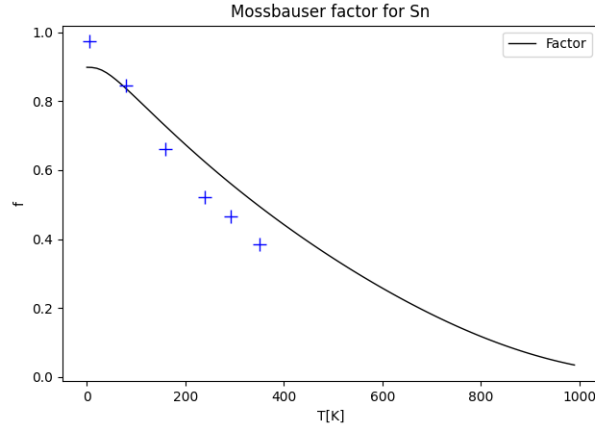


Figure 3: Graph with visualized experimental data

## 2.2 Graph

Once you have processed your data, you can display it on a graph by pressing the `Graph` button. This will automatically generate  $n$ -number of graphs, where  $n$  is the number of atoms in the data. These graphs can be displayed using the `Atom 1 Probability` combobox, where you can choose different types of atoms and mean square displacements.

If you load new data and click `Processing`, you will also need to click `Graph` to display the new data. This also applies to the `Save Graph` function.

### 2.2.1 Augmenting Graph

The graphs themselves are standard `matplotlib` graphs, so the augmentation is limited to the capabilities of the library. If you want to augment a specific graph, you need to select which variable you wish to change, starting with either the factor,  $x^2$ ,  $y^2$ , or  $z^2$ . This will determine which variable will change on the graphs. This will apply to all generated graphs. Once you make your selected augmentations, click `Graph` again, and your new graph will display on the graphic window. You can then save the graph as you see it.

The resolution combobox allows you to change the resolution of all graphs.

In Figure 4, you can see a generated graph with the following augmentations: Color: Green, Linestyle: Solid, Linewidth: 2, Markers: None, Resolution: 720x480.

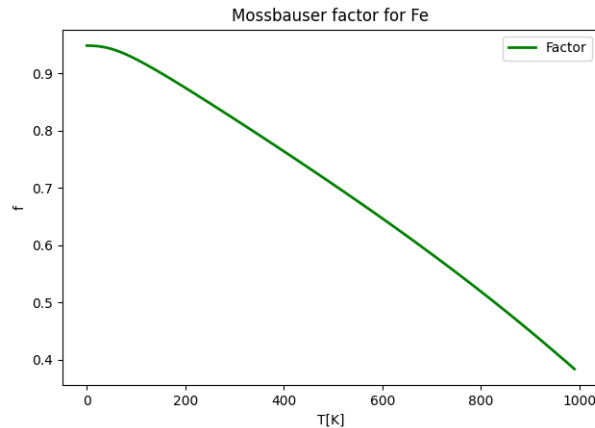


Figure 4: Generated graph

## 2.3 Saving Data

Once processed, you can save your data by pressing the `Save Tex` button. This will open a save dialog window where you can choose the location to save your data. The data will be saved in the format displayed in Figure 5. You can also save the plotted graph using the same method with the `Save Graph` button. Note that it will save the graph currently selected in the combobox.

Temperature [K]	MSD of x	MSD of y	MSD of z	Factor
141.2894185587	0.0035130253	0.0035130253	0.0041944985	0.8173409008
190.6537113207	0.0045654377	0.0045654377	0.0055446650	0.7675341108
235.7517632707	0.0055107243	0.0055107243	0.0065455595	0.7294224840
279.1578052473	0.0064694360	0.0064694360	0.0074354220	0.6949761765
320.6705485258	0.0074066777	0.0074066777	0.0085565716	0.6585308073
360.5589395952	0.0084596035	0.0084596035	0.0097278709	0.6212945715
398.9792037663	0.0095086931	0.0095086931	0.0107498185	0.5885183614
489.7668517954	0.0124656063	0.0124656063	0.0131539799	0.5114826144
573.5108099521	0.0144200170	0.0144200170	0.0163279431	0.4472459331
650.9143438572	0.0173358381	0.0173358381	0.0211002230	0.3657382428
722.4817268112	0.0189778001	0.0189778001	0.0246082575	0.3196245690
788.7075704599	0.0209212091	0.0209212091	0.0293484060	0.2683371890
849.6801365339	0.0228779532	0.0228779532	0.0324161612	0.2352756245
906.4498820617	0.0261967406	0.0261967406	0.0332990077	0.2107778409
958.7835593661	0.0290546243	0.0290546243	0.0404835243	0.1620660625

Figure 5: Text file with calculated data

## 3 Processing

### 3.1 Quasi-harmonic Approximation

To use this function correctly, you need to load all the respective displays with their corresponding files. The required files are: `phonopy.yaml`, Thermal displacement files, Volume-Temperature files, and e-V files. The respective formats for these files can be found in the section *Formats*.

### 3.2 Harmonic Approximation

For this function, you need only one `phonopy.yaml` file in the correct format and one Thermal displacement file. Then click **Processing** to complete the process.

### 3.3 Recoil Energy Constants Included

As you know, not all metals are Mössbauer active. To prevent the issue of calculating the Mössbauer factor for non-active Mössbauer metals, we have included some of the recoil energies, which are automatically selected based on your `phonopy.yaml` file in the backend of our software. These specific atoms are:

- $^{57}\text{Fe}$ :  $E_R = 1.95883310 \cdot 10^{-3} \text{ eV}$
- $^{119}\text{Sn}$ :  $E_R = 2.57423 \cdot 10^{-3} \text{ eV}$
- $^{129}\text{I}$ :  $E_R = 2.57423 \cdot 10^{-3} \text{ eV}$
- $^{121}\text{Sb}$ :  $E_R = 6.122 \cdot 10^{-3} \text{ eV}$
- $^{191}\text{Ir}$ :  $E_R = 1.9094 \cdot 10^{-2} \text{ eV}$

If there are other atoms included in your Phonopy simulation, the function that appends the recoil energy to the respective atom will return 0. But since this would return a factor of 1, which can not be for all temperatures our software uses a condition that will set factor to 0. In the future, we may add other isotopes to the backend of the software depending on user feedback.

## 4 File Formats

In this section you can find all the respective formats and descriptions of the files used by this software.

### 4.1 Phonopy.yaml Format

The format of the `Phonopy.yaml` file is illustrated in Figure 6. The software extracts critical information from this file, including the number of atoms, their names, and **their respective atomic numbers**. The `Phonopy.yaml` file is essential for all calculations; without it, the software will not function correctly.

```

phonopy:
  version: 2.6.1
  frequency_unit_conversion_factor: 15.633302
  symmetry_tolerance: 1.00000e-05
  configuration:
    atom_name: "Fe Co Ni"
    dim: "2 2 2"
    mp: "20 20 20"
    pdos: "1 2 3 4 5 6 7 8 9, 10 11 12 13 14 15 16 17 18, 19 20 21 22 23 24 25 26 27"
    fc_symmetry: ".TRUE."

physical_unit:
  atomic_mass: "AMU"
  length: "Angstrom"
  force_constants: "eV/Angstrom^2"

supercell_matrix:
- [ 2, 0, 0 ]
- [ 0, 2, 0 ]
- [ 0, 0, 2 ]

space_group:
  type: "P1"
  number: 1
  Hall_symbol: "P 1"

primitive_cell:
  lattice:
- [ 0.003254512089877, 5.305972571279662, 5.294783818069269 ] # a
- [ 5.306043931509095, -0.006567057763793, 5.304607227504651 ] # b
- [ 5.315487753641971, 5.325264245183370, -0.016039922332086 ] # c
  points:
- symbol: Fe # 1
  coordinates: [ 0.669392220535605, 0.336304132060926, 0.661888596743881 ]
  mass: 55.845000
- symbol: Fe # 2
  coordinates: [ 0.995583634031596, 0.673744751688022, 0.994428766691188 ]

```

Figure 6: Phonopy format for  $FeCoNi$

```

# Thermal displacements
natom: 4
freq_min: 0.000000
thermal_displacements:
- temperature: 0.000000
  displacements:
- [ 0.0026610, 0.0026610, 0.0026610 ] # atom 1
- [ 0.0026610, 0.0026610, 0.0026610 ] # atom 2
- [ 0.0020373, 0.0020373, 0.0020373 ] # atom 3
- [ 0.0015003, 0.0015003, 0.0015003 ] # atom 4

```

Figure 7: Thermal displacement format for  $FeSn_2$

## 4.2 Thermal Displacement Files

The standard format for Thermal displacement files is shown in Figure 7.

The distinction between the Harmonic and Quasi-Harmonic approximations lies not in the format of these files, but in their respective names.

### 4.2.1 Harmonic Approximation

When using the software for Harmonic approximation, you will need the standard Thermal displacement format as shown in Figure 7. No additional suffix is required beyond the ".yaml" extension.

### 4.2.2 Quasi-Harmonic Approximation

For the Quasi-Harmonic approximation, if you are using more than one Thermal displacement file, each file must have a specific suffix following ".yaml-". The suffix should be "+n", where  $n$  represents the order of the sorted Thermal displacement file and corresponds to the specific Energy-Volume relation. This is a crucial step and must be executed correctly for the software to function properly.

### 4.3 Energy-Volume Files

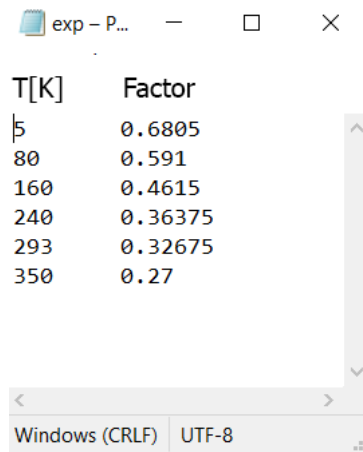
Energy-Volume files are required exclusively for the Quasi-Harmonic approximation. You will need a standard file in either ".dat" or ".txt" format. This file should list the energies and volumes corresponding to the indices in the Thermal displacement files. It is crucial that the number of rows in this file matches the number of Thermal displacement files that were uploaded. The format should be such that energies are listed first, followed by a tab, and then the corresponding volumes.

### 4.4 Volume-Temperature Files

Volume-Temperature files are also required solely for the Quasi-Harmonic approximation. The standard file formats are the same as those used for Energy-Volume files. In this case, the first column should contain temperatures, and the second column should list the corresponding volumes.

### 4.5 Experimental Data

To ensure that you load the correct version of experimental data files for displaying points on the graph, you need to normalize the experimental data so that the factor lies within the interval  $(0, 1)$  and corresponds to a temperature in Kelvin. The values must be formatted as shown in Figure 8.



T[K]	Factor
5	0.6805
80	0.591
160	0.4615
240	0.36375
293	0.32675
350	0.27

Figure 8: Text file format of experimental data

## 5 Flowchart

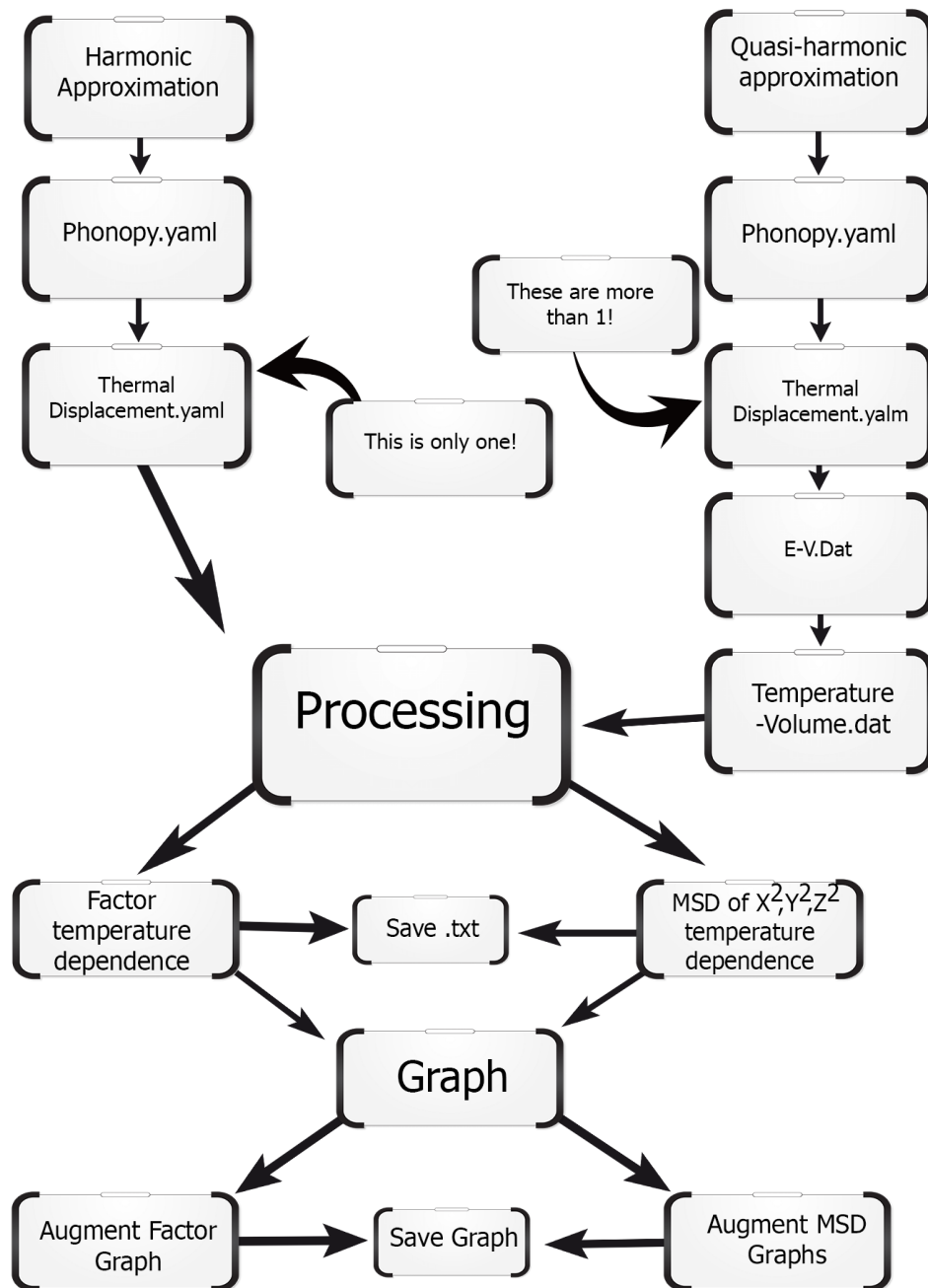


Figure 9: Flowchart

## 6 Conclusion and Outlook

In this manual we have presented *TriMEph*, an object-oriented tool for computing Mössbauer factors under both harmonic and quasi-harmonic approximations, directly from standard `phonopy.yaml` inputs. You learned how to:

- Upload and manage all required input files (`phonopy.yaml`, thermal-displacement, energy–volume, and volume–temperature data).
- Process your data using the built-in harmonic or quasi-harmonic routines.
- Visualize results interactively, overlay experimental points, and export both numerical tables and publication-quality graphs.
- Leverage automatic recoil-energy selection for common Mössbauer-active isotopes.

Looking ahead, planned enhancements include:

- Expanding the list of supported isotopes and automating user-defined recoil energies.
- Integrating advanced phonon-lifetime modelling for temperature-dependent line-broadening.
- Adding scripting hooks to enable batch processing and high-throughput studies.
- A plugin interface for custom plotting styles and additional data formats.

We hope *TriMEph* streamlines your Mössbauer spectroscopy workflows and invites your feedback. For bug reports, feature requests, or contributions, please visit our repository or contact the author.

**Happy computing!**