RESEARCH METHODS VOLCANICA

PetThermoTools: a fast, flexible, and accessible Python3 package for performing thermodynamic calculations

Matthew Gleeson *[α], Penny E. Wieser[α], Paula M. Antoshechkina[β], and Nicolas Riel[γ]

[α] Earth and Planetary Science, UC Berkeley, CA, USA.
[β] Division of Geological and Planetary Sciences, Caltech, Pasadena, CA, USA.
[γ] Institute of Geosciences, Johannes Gutenberg-University, Mainz, Germany.

Submitted: 2025-12-23 | Accepted: 2026-05-14 | Published: 2026-05-18

Editor: C. Montagna | Typesetter: J. Farquharson

Abstract

We present PetThermoTools: an open-source Python3 tool for performing thermodynamic simulations of mantle and magmatic processes. Thermodynamic modeling forms a central component to many research projects in igneous petrology. However, few studies utilize the full potential of these methods due to the steep learning curve associated with existing code or text-based interfaces (e.g. ENKI/ThermoEngine, alphaMELTS, Theriak–Domino) and the limited model flexibility provided by spreadsheet and GUI based packages (e.g. MELTS for Excel). We designed PetThermoTools to bridge this divide, providing logical, easy-to-use functions with well-documented examples for novice users, while retaining the potential for customization and model automation desired by more experienced users. PetThermoTools also provides the opportunity to directly compare different thermodynamic models and approaches, by integrating with both the alphaMELTS and MAGEMin packages. Furthermore, PetThermoTools model outputs seamlessly integrate with other Python-based packages created for petrological research (e.g. PySulfSat), and provide speed and stability advantages over existing free MELTS-based software owing to the use of parallel processing routines.

Keywords: Thermodynamic modeling; MELTS; Volcanology; Open-Source; Python.

1 Introduction

Simulating mantle and magmatic processes is key for advancing our understanding of volcanic systems. These models provide constraints on the nature and location of magma storage [Gualda and Ghiorso 2014; Gleeson et al. 2017], the chemical and physical properties of magma before eruption [Blundy et al. 2006; Tramontano et al. 2017], and the characteristics of mantle source regions (i.e. temperature, lithology, and composition [Lambart 2017; Brunelli et al. 2018; Brown et al. 2020; Gleeson and Gibson 2021; Matthews et al. 2021; 2022]; helping to interpret the available petrological, geochemical, and geophysical data. Petrological modeling can be subdivided into two main approaches: empirical parameterizations of experimental data [Ariskin et al. 1993; Danyushevsky and Plechov 2011; Ariskin et al. 2018]; and thermodynamic models, where the equilibrium state of the system is determined by minimizing thermodynamic potentials [Ghiorso and Sack 1995; Ghiorso et al. 2002; Gualda et al. 2012; Holland et al. 2018; Riel et al. 2022].

Empirical parameterizations are typically faster than thermodynamic approaches and less prone to crashes and/or model failures. However, they are usually calibrated on a narrow range of compositions and are designed for a singular purpose (e.g. mantle melting or fractional crystallization [Matthews et al. 2022]), which can limit their wider applicability. By contrast, thermodynamic models are often calibrated on large and compositionally diverse datasets that allow them to be applied (with caution) to a range of magma compositions from various tectonic settings [Ghiorso and Sack 1995; Wieser et al. 2022a]. Furthermore, while empirical parameterizations might track a small number of key variables (e.g. phase proportions and/or chemistry), thermodynamic simulations are able to trace the distribution of energy, mass, and chemical components within igneous systems. This enables the creation of more realistic models, including energy-constrained assimilation and melt-mush reaction [Heinonen et al. 2019; Boulanger and France 2023; Gleeson et al. 2023]. Machine learning models may offer a promising compromise, retaining the wide applicability of thermodynamic models while substantially improving computational efficiency [Candioti et al. 2025].

Despite their potential, application of thermodynamic models to pressing research questions in volcanology and igneous petrology has been restricted by the time-consuming nature of these calculations, the high level of coding experience required to perform more complex simulations and integrate results with other calculations/models, as well as the challenges associated with automating calculations due to the frequency of software crashes. In addition, many thermodynamic software tools operate as 'black boxes', with most users unable to access, or interpret the underlying code and minimization routines. To address these issues, we developed PetThermoTools, an open-source Python package designed to facilitate thermodynamic simulations of mantle and magmatic systems. For novice users with limited coding experience, PetThermoTools provides logical, easy-to-use functions and example notebooks for common thermodynamic calculations (e.g. crystallization models). The simple code layout and logical output structures support integration of thermodynamic model results with other Python packages developed for petrological research (e.g. PySulfSat [Wieser and Gleeson 2023]). Furthermore, the use of CPU-based parallelization isolates each thermodynamic calculation in a separate process, preventing a single failure from corrupting the entire calculation, a key limitation of previous software options. These improvements allow users to integrate PetThermoTools di-

* gleesonm@berkeley.edu

rectly into their Python workflow. As a result, we believe that PetThermoTools has the potential to become a critical component of petrological research and education, providing greater access to thermodynamic modeling for all researchers, regardless of prior coding experience.

2 Software tools for thermodynamic modeling

Since the release of the initial MELTS model in 1995 the "MELTS" family of thermodynamic models has emerged as the most influential group of models in igneous petrology. Currently, the term "MELTS" can be used to refer to four different thermodynamic models—pMELTS [Ghiorso et al. 2002]; rhyolite-MELTS v1.0.2 [Gualda et al. 2012]; rhyolite-MELTS v1.1.0; and rhyolite-MELTS v1.2.0 [Ghiorso and Gualda 2015]—and at least six different software options that are used to perform the calculations. These software tools can broadly be split into two groups: (i) GUI and Spreadsheet based options that provide users with highly accessible tools, optimized for specific calculations but with limited potential for automation or more complex calculations (e.g. easyMelts, MELTS for Excel [Gualda and Ghiorso 2015]); and (ii) code and text-based interfaces that allow more complex simulations, integration of thermodynamic approaches with other research tools, and an increased potential for model automation (e.g. alphaMELTS [Antoshechkina and Ghiorso 2018]; ENKI/ThermoEngine [Thermoengine 2022]). However, the steep learning curve associated with this second group of software options discourages novice users and limits applications to undergraduate and postgraduate education.

While the "MELTS" thermodynamic models are the most widely used in the igneous petrology and volcanology literature, alternative models are available. Most notably, work by Tim Holland, Roger Powell, and co-workers has resulted in the development of several thermodynamic models applicable to igneous systems [Jennings and Holland 2015; Holland et al. 2018; Riel et al. 2022; Weller et al. 2024; Green et al. 2025]. These models have traditionally been accessed through the software options THERMOCALC, Perple_X, and Theriax_Domino, which are each associated with their own strengths and weaknesses [Connolly 2009; De Capitani and Petrakakis 2010]. In 2022, however, a Gibbs Free Energy minimization software called MAGEMin was released, providing easy access to these models through Julia wrappers (MAGEMin_C) and a Julia-based app (MAGEMinApp [Riel et al. 2022]). MAGEMin provides advantages in the speed of calculations (relative to Perple_X etc.) and the app allows novice users to perform simple phase diagram, crystallization, and melting calculations. However, a feature that is currently absent from all available software options is the ability to easily switch between and compare the "MELTS" and "H&P" group of models. Consequently, comparison of the two groups of thermodynamic models has remained limited in the published literature [Jennings and Holland 2015; Hernández-Uribe et al. 2022; Otto et al. 2023; Wieser et al. 2025].

3 PetThermoTools

To address the limitations of the currently available software options, we developed PetThermoTools as an easy-to-use and expandable package for thermodynamic calculations. We chose to write this package in Python as it is one of the most widely used coding platforms in petrology, with a series of other Python-based packages recently developed for petrological research (e.g. VESIcal [Iacovino et al. 2021; Wieser et al. 2022a]; pyMelt [Matthews et al. 2022]; Thermobar [Wieser et al. 2022b]; pySulfSat [Wieser and Gleeson 2023]; meltPT [McNab and Ball 2023]; pyrolite [Williams et al. 2020]). Where possible, we designed PetThermoTools to easily interact with these other petrology-focused Python packages (see Section 4.3), allowing thermodynamic modeling to become an integrated component of the Python workflow of researchers. Our aim was to produce a software package in which users with little to no coding experience can run simple MELTS models, but with the flexibility and customization required for advanced users to perform more complex calculations and/or contribute to new functionality. In addition, by integrating PetThermoTools with the core functions of both alphaMELTS for Python and MAGEMin, it is now possible for researchers to critically examine and compare different thermodynamic models in an efficient and easy manner (e.g. Wieser et al. 2025).

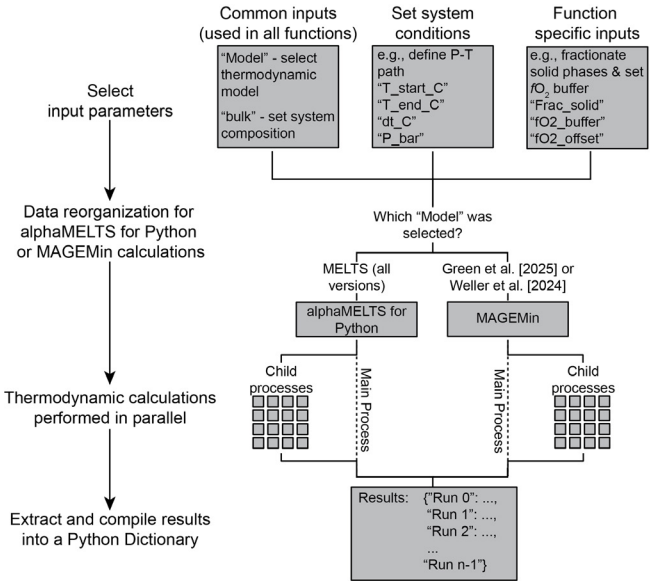


Figure 1: Schematic flow chart demonstrating the structure of PetThermoTools. User input parameters are re-organized into a structure that allows calculations to be performed in alphaMELTS for Python [Antoshechkina and Ghiorso 2018] or MAGEMin [Riel et al. 2022], depending on the model choice. Calculations are then performed in parallel using the Python multiprocessing library or Julia Distributed module and results are returned in a Python Dictionary.

3.1 Installation

Detailed installation instructions, including differences for MacOS, Linux, and Windows users, can be found on the PetThermoTools ReadTheDocs page*. This page will be con-

*https://PetThermoTools.readthedocs.io/en/latest/

tinuously updated as the code develops. A brief summary of the installation processes is included here:

1. Users must first ensure that they have an up-to-date version of Python installed on their machine (version 3.10 or later as of April 2026). Installation through Anaconda is recommended for users new to Python programming.

2. PetThermoTools can then be installed through PyPI via the command prompt (Windows) or Terminal (MacOS, Linux):

```
pip install petthermotools
```

Or directly from a Jupyter Notebook:

```
%pip install petthermotools
```

Once PetThermoTools is installed, it can be imported into the user's Python environment. This is done using the following code where any combination of letters can be used to represent PetThermoTools (here we use *ptt*):

```
import petthermotools as ptt
```

3. **Only required for MELTS calculations**. Simulations using the pMELTS or rhyolite-MELTS (v1.0.2, v1.1.0, and v1.2.0) thermodynamic models require users to follow an installation notebook, provided through ReadtheDocs, that downloads the underlying alphaMELTS for Python package onto the user's local machine. This notebook walks the user through the steps required and provides an option to add the alphaMELTS for Python code to the user's Python path.

4. **Only required for "H&P" calculations**. To enable calculations in MAGEMin we provide a second installation notebook, also available on the PetThermoTools ReadtheDocs page. This notebook walks users through a simple process to install the Julia packages required for calculations in MAGEMin and uses PythonCall to establish a connection between Python and Julia. Critically, this notebook does not require the users to have Julia pre-installed, nor do they need to install Julia or any dependencies independently. All required installations are handled in the provided notebook and associated functions. Once installation is complete, users must run one additional line of code at the start of their notebook/script to activate the correct Julia environment and establish the Python–Julia connection. We recommend that in all cases where calculations are being performed in MAGEMin through PetThermoTools users start their notebook with:

```
import petthermotools as ptt
ptt.activate_petthermotools_env()
```

3.2 Code structure

PetThermoTools is structured so that little to no prior Python knowledge is required, ensuring that thermodynamic modeling is available to all users and can be used as a teaching tool. Extensive documentation is available for all functions with examples for the most common calculations available on ReadtheDocs. All functions contain a small number of key input variables (e.g. bulk composition and thermodynamic model choice), variables to set and control system conditions (i.e. temperature, pressure, entropy, volume and/or enthalpy), and additional variables that are specific to the chosen function (Figure 1). These additional variables provide greater flexibility for more complex calculations; for example, users can specify a crystallinity threshold (rather than temperature condition) as the end point of a crystallization calculation.

One of the key aspects of PetThermoTools is that input variables (with the exception of the "Model" key-word argument) are identical regardless of which thermodynamic model the user chooses. The input parameters provided by the user are reorganized within PetThermoTools so that this information can be fed directly into the key functions of alphaMELTS for Python or MAGEMin. This means that novice users have no need to directly interact with the underlying packages, and do not have to worry about reformatting their data or using the correct parameter names to match each model's unique syntax.

If the input parameters indicate more than one calculation is desired (e.g. the user provides an array of initial system H_2O contents), by default PetThermoTools will utilize Python's multiprocessing library (MELTS) or Julia's Distributed module (H&P) to run these calculations simultaneously in separate 'Child Processes'. The maximum number of parallel calculations that can be performed is automatically determined based on the specifications of the user's computer, with parallel processing providing both speed and stability advantages for users wishing to perform large numbers of calculations (see Section 4.2).

Outputs from thermodynamic calculations performed through PetThermoTools are stored as Pandas DataFrames nested within a Python Dictionary (Figure 2). These DataFrames record the system conditions (temperature pressure, enthalpy etc.), as well as the mass, volume and density of each phase that saturated in the model. In addition, DataFrames are created for the chemical composition (in weight percent), and physical and thermodynamic properties (e.g. mass, density, enthalpy, heat capacity) of phases that saturated in the calculations. The 'keys' used in the higher-level Dictionary to identify each phase are based on the naming convection used in alphaMELTS ('liquid1', 'clinopyroxene1', etc.). Furthermore, the units for all outputs, with the exception of chemical compositions, are indicated in the column headers of their respective DataFrames and in the Dictionary keys for DataFrames of specific parameters (e.g. 'rho_kg/m3'). This structure makes it easy to access any parameter; for example, the concentration of MgO in the melt phase during fractional crystallization—from either an alphaMELTS or MAGEMin calculation—can be examined via:

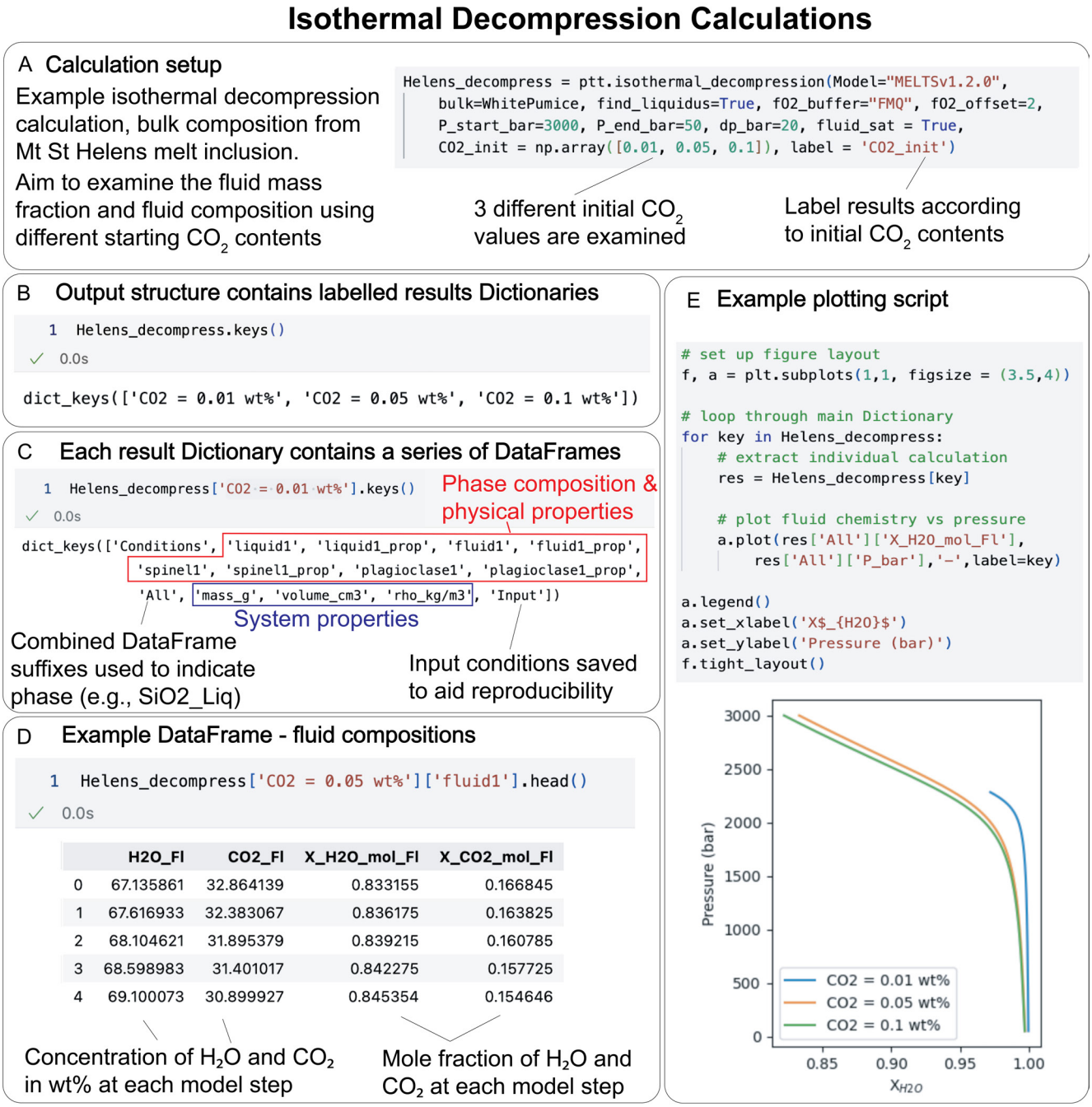


Figure 2: Outline of the PetThermoTools output structure demonstrated using an example of an isothermal decompression calculation. [A] The calculation is set up using the composition of a plagioclase hosted melt inclusion from the Mount St Helens white pumice [Blundy and Cashman 2005]). Three different initial CO_2 contents are tested and the `label` key-word argument is used to indicate that the output variables should be labeled by these initial CO_2 contents. [B] As multiple calculations are initiated the main output variable is a Python Dictionary containing sub-Dictionaries labeled by the initial CO_2 content of each calculation. [C] Each sub-Dictionary contains several Pandas DataFrames recording the system conditions and properties in addition to the composition and physical properties of each phase. [D] The '`fluid1`' key can be used to access the DataFrame containing the composition of the fluid phase. [E] Example plotting script revealing the different fluid compositions observed in the decompression models. A more complex version of this example, including both open and closed system degassing calculations, is available on ReadtheDocs.

```
results["liquid1"]["MgO_Liq"]
```

where `results` represents the Dictionary returned by the `PetThermoTools` calculation, with the chemical composition of the melt phase recorded in the DataFrame associated with the key `"liquid1"`. As can be seen in this example, we have applied a suffix to all output column headers to identify the phase being examined. We have used the naming convention for different phases from `Thermobar` to enable easier integration of thermodynamic model results with other `Python` packages (see Section 4.3 [Wieser et al. 2022b]). By storing these data as Pandas DataFrames nested within a `Python` Dictionary, it is trivial to convert the model outputs into an `Excel` file, where the "sheets" within the spreadsheet represent each DataFrame within the output dictionary. An example of how to do this conversion is available on ReadtheDocs. In cases where the input variables require that more than one thermodynamic calculation is performed—for example, an array of initial system CO_2 contents are passed to the `isothermal_decompression` function (Figure 2)—`PetThermoTools` will create a nested dictionary structure, with the dictionary of results for each individual calculation stored as items within a higher-level dictionary (Figure 2). The keys used to identify each individual calculation will, by default, be set to Run 0, Run 1, Run 2, etc. although the user can specify certain parameters—based on the chosen input arguments—to use as the keys if appropriate (e.g. CO_2 = 0.001 wt.%, CO_2 = 0.01 wt.%, CO_2 = 0.025 wt.%). For example, in crystallization or decompression calculations, users can access the mass of each phase for calculations with an initial starting CO_2 content of 0.025 wt.% via the following code:

```
results["CO2 = 0.025 wt%"]["mass_g"]
```

By creating this nested dictionary structure, we have made it simple for users to identify and isolate the outputs that they need.

3.3 Normalization

If a user provides a list of oxide values, representing the initial composition of their system, to `MAGEMin` or `alphaMELTS` for `Python` these values will be treated slightly differently in each package. `MAGEMin` normalizes the input composition to unity, including H_2O, and expresses all output compositions as either the mass fraction or mole fraction of each oxide. `alphaMELTS` for `Python`, however, treats the provided values as the mass, in grams, of each oxide at the start of the calculation. To account for this difference and aid in the comparison of the two model approaches, we normalize all compositions provided by the user to a total of 100 g, prior to any thermodynamic calculations. Normalization to 100 g (rather than to 100 wt.%) is chosen due to the way `alphaMELTS` for `Python` treats these input values and reports output values such as volume, enthalpy, entropy, etc. Notably, to ensure consistency between the `MAGEMin` and `alphaMELTS` for `Python` calculations, we have had to convert the `MAGEMin` outputs—which report values such as volume or mass as a percentage of the system at that model step—into the mass, volume, enthalpy etc. relative to the initial 100 g system. As a result, if a user wanted to assess the volume of a magmatic system with a set mass (potentially informed by the mass of erupted products), they would have to scale up the volume estimate from `PetThermoTools` by the ratio between their chosen mass value and the mass value returned by `PetThermoTools`.

How the initial compositions are normalized in `PetThermoTools` is dependent on which key-word arguments are used to define the system volatile contents. If the initial abundance of H_2O or CO_2 in the system is defined within the 'bulk' key-word argument (alongside all other oxides), `PetThermoTools` will rescale all oxides—including these volatile components—to a hydrous total of 100 g. If the initial H_2O and CO_2 contents are instead specified using the 'H2O_init' and/or 'CO2_init' keyword arguments, `PetThermoTools` will keep these values fixed, and re-adjust the other oxides so that the hydrous sum is 100 g. This means that calculations can be run at the volatile contents the user specified, rather than being scaled due to normalization of all components. For example, if a user provides a bulk composition with 10 wt.% H_2O and the non-volatile oxides sum to 100 wt.% (anhydrous normalized), all inputs will be divided by 110/100 = 1.1. This means that if the SiO_2 content was initially 50 wt.% the calculation will be initiated at 50/1.1 = 45.45 wt.%, and H_2O will be scaled down to 10/1.1 = 9.09 wt.%. In contrast, if the user specifies H_2O=10 wt.% as a keyword argument ('H2O_init'), all other oxides will be divided by a factor of 100/90 =1.1111, so SiO_2 will be scaled down to 45 wt.%, and the calculation will initiate with 10 wt.% H_2O.

3.4 Available functions

We are continually working to update and expand the functions available in `PetThermoTools`, and to keep these up to date with the latest versions of `alphaMELTS` for `Python`, `MAGEMin`, and all other underlying packages (e.g. `NumPy`). An up-to-date list of the available functions will be maintained on the `PetThermoTools` ReadtheDocs page*, along with extensive documentation and example notebooks. A summary of the currently available functions is provided in Figure 3, which also highlights what methods are available using the `MELTS` and/or H&P models. Figures 4, 5, 6, and 7 represent example calculations, with the code used to create these figures available via ReadtheDocs. Current functions can be split into several different groups (note that the items below do not represent the function names; please refer to the ReadtheDocs page for this information). Methods that are in development—i.e. not currently available in the latest release, v1.0.0—are indicated by '***':

1. Crystallization calculations

(a) `isobaric crystallization`: simulations of equilibrium or fractional crystallization at a constant pressure (Figure 4). Several additional options are provided for these calculations, such as the ability to remove any excess volatile phase

*https://PetThermoTools.readthedocs.io/en/latest/available_functions_pdf.html

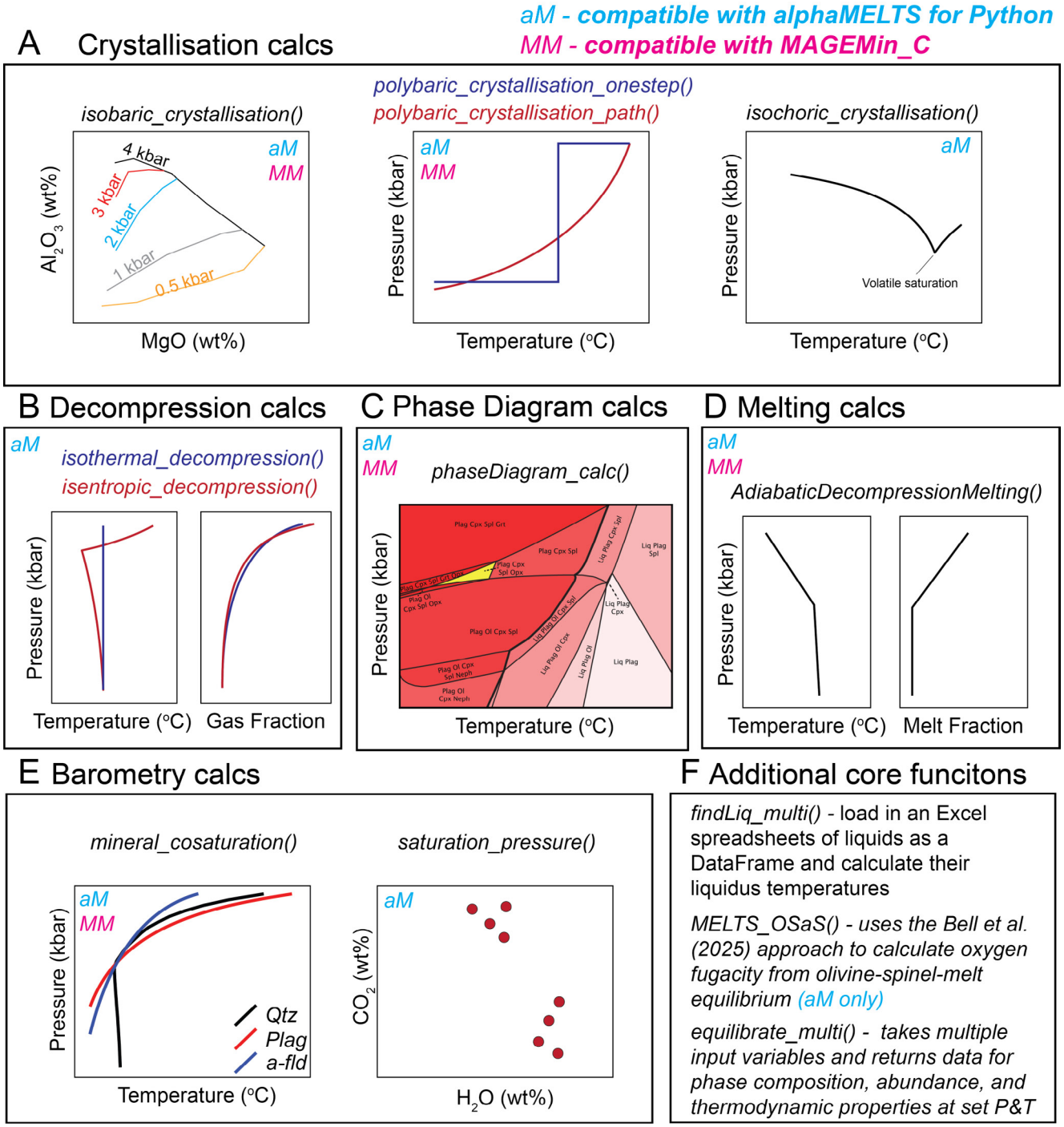


Figure 3: Outline of the functions currently available in `PetThermoTools`. These can be split into multiple groups: crystallization functions, decompression functions, phase diagram functions, melting functions, and barometry functions. We are aiming to grow and expand the range of functions available in `PetThermoTools`. For example, current work is focused on the development of an adiabatic decompression melting function for a multi-component mantle. Methods that require knowledge of the CO_2 content of the melt phase are only available for calculations using the `rhyolite-MELTS` v1.1.0 and v1.2.0 models.

from the system prior to initializing the crystallization path (i.e. starting calculations at the point of fluid saturation but with no excess fluid phase) or to set a crystallinity limit (rather than temperature) as the end point of the calculation. As with prior `MELTS` tools, certain phases can be suppressed, so a crystallization path excluding orthopyroxene, or including only olivine and spinel can be run.

(b) `polybaric crystallization`: the user can either define a full P-T path, or select a single temperature at which the system changes pressure.

(c) `isochoric crystallization`: the volume of the system is set from the initial calculation step (at the user-specified pressure), after which volume is held constant and pressure is treated as a dependent variable (i.e. a model assuming a

magma chamber has a fixed volume and undergoes no crustal relaxation).

(d) ***`enthalpy constrained crystallization`: rather than setting a temperature change for each model calculation the user can specify a change in enthalpy associated with each model step.

(e) ***`energy-constrained assimilation`: simulate assimilation (and melting) of solid and/or mushy material with the enthalpy of the total system used to constrain the new equilibrium state (e.g. Gleeson et al. 2023).

2. Decompression calculations

(a) `isothermal decompression`: magma ascent and decompression with temperature held constant (Figure 2). All decompression calculations in `PetThermoTools` track the evolution of the full liquid-vapor-solid system and, therefore, provide significant advantages of prior thermodynamic modeling approaches that only track the evolution of the liquid-vapor system (e.g. `VESIcal` [Iacovino et al. 2021]). Future work will focus on integrating decompression calculations with models of system viscosity that account for factors such as microlite nucleation and growth.

(b) `isentropic decompression`: similar ascent calculations but with the entropy of the system held constant (i.e. an adiabatic processes).

(c) ***`isenthalpic decompression`: the system enthalpy is held constant with temperature treated as a dependent variable.

3. Mineral co-saturation calculations

(a) `find pressure of cosaturation`: perform multiple isobaric crystallization calculations to determine the pressure of multiple phase saturation. This method is based on prior approaches to determining the pressure of plagioclase - alkali feldspar - quartz cosaturation [Gualda and Ghiorso 2014]), and orthopyroxene - clinopyroxene - plagioclase cosaturation [Harmon et al. 2018]), but is expanded to include any possible combination of two or three mineral phases (Figure 5). Unlike prior software options, it is also possible to perform these calculations with the H&P thermodynamic models.

(b) ***`find` H_2O`, pressure, and` fO_2 `of cosaturation`: for more complex scenarios the search for mineral cosaturation will be extended over multiple parameters. This will allow users to search for mineral cosaturation over any combination of H_2O, CO_2, pressure, and fO_2 values.

4. Melting Calculations

(a) `adiabatic decompression melting`: isentropic decompression of solid material triggering melting and magma generation (Figure 6). `PetThermoTools` is also designed to interact with—and call key functions and methods from—the `pyMelt` `Python` package [Matthews et al. 2022]). This allows easy comparison of, not only different thermodynamic models, but also thermodynamic and empirical approaches to simulating mantle melting (Figure 6).

(b) ***`isobaric melting`: melt generation at a constant pressure due to changes in temperature

(c) ***`multi-component melting`: an expansion of the adiabatic decompression melting function where the melting behaviour of multiple lithologies—that are thermally connected but chemically distinct—is simulated.

5. Phase Diagram calculations

(a) `Phase Diagram from P-T grid`: assign a pressure and temperature range to construct a phase diagram. `PetThermoTools` also provides functions for visualizing the phase diagrams, and methods for creating contour plots (e.g. of phase proportions) are available on ReadtheDocs (Figure 7).

(b) ***`Phase Diagram by reaction`: rather than calculating the full P-T grid, phase diagrams can be constructed by tracing the P-T conditions at which particular phases saturate within the system (or are removed from the stable assemblage). For example, this could be used to identify the garnet-out and spinel-in reactions for different mantle lithologies.

6. Volatile saturation calculations

(a) `Saturation pressure`: Given a melt composition and volatile content `PetThermoTools` can calculate the volatile saturation pressure. This can be achieved using either a user specified T or `PetThermoTools` can simultaneously solve for the liquidus temperature and saturation pressure.

(b) ***`Volatile solubility`: for a given melt composition and pressure calculate the maximum amount of H_2O or CO_2 that can dissolve in the melt phase. This function requires knowledge of either the H_2O or CO_2 content of the melt phase (if solving for the other component) or the composition of a co-existing fluid (X_{H_2O}).

7. Additional calculations

(a) `find liquidus`: search for the temperature where the first solid stabilizes.

(b) `equilibration`: calculate the equilibrium state of the system for a given pressure, temperature, system composition, and oxygen fugacity.

(c) `supplemental calculator`: estimate the thermodynamic properties of a phase for a given temperature, pressure, and composition. A previous online web-calculator for this had been extensively used for several petrological applications, such as estimating the activity of TiO_2 in magmatic liquids [Ghiorso and Evans 2008].

8. Additional Methods

(a) `MELTS Olivine-Spinel-a`$^{melt}_{SiO2}$ `(OSaS) oxybarometer`: recreation of the `MELTS`-based OSaS oxybarometer of Bell et al. [2025] including correction for a_{Fe3O4} in spinel, allowing users to calculate oxygen fugacity from the composition of co-existing olivine, spinel, and melt.

Isobaric Crystallization Calculations

A Calculation setup

Isobaric crystallization calculations at different pressures. Starting composition is taken from a melt inclusion collected from offshore Isla Fernandina in the Galápagos Archipelago.

```
Isobaric_Xtal_CO2 = ptt.isobaric_crystallisation(Model = "MELTSv1.2.0",
    bulk=starting_comp,find_liquidus=True,P_bar=np.array([500,1000,2000,4000]),
    T_end_C=1050,dt_C=2,fO2_buffer="FMQ",fO2_offset=-1.0,
    Frac_solid=True,Frac_fluid=True,H2O_init=0.4,CO2_init=0.5,label="P_bar")
```

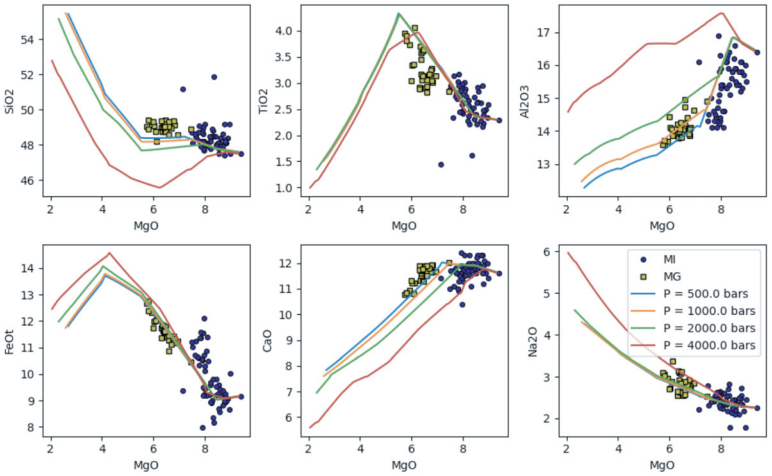
B Inbuilt functions for oxide-oxide plots

```
ptt.harker(Results = Isobaric_Xtal_CO2, data = {'MI': MI, 'MG': MG},
    d_color = ['b','y'], legend_loc = [1,2],
    y_axis = ['SiO2', 'TiO2', 'Al2O3', 'FeOt', 'CaO', 'Na2O'])
```

Specifies which graph (row,column) should contain the legend

By default MgO is shown on the x-axis with up to 7 other oxides displayed on the y-axis. Here 6 oxides are chosen.

The 'data' input allows users to plot their data alongside the models. Here we use a Dictionary structure to load two DataFrames to compare to the model results: MI - melt inclusion data, and MG - matrix glass data.



C Using pyrolite (Williams et al. 2020) to compare models and data on TAS diagrams

```
# ---- Create TAS diagram from pyrolite ----
from pyrolite.util.classification import TAS
cm = TAS()
f, a = plt.subplots(1,1, figsize=(4,3.5))
cm.add_to_axes(a,alpha=0.5,linewidth=0.5,
    zorder=-1,add_labels=False)

# ---- Loop through model results ----
for i in Isobaric_Xtal_CO2:
    df = Isobaric_Xtal_CO2[i]['All'].copy()
    df['Na2O+K2O']=df['Na2O_Liq']+df['K2O_Liq']
    a.plot(df['SiO2_Liq'],df['Na2O+K2O'], lw = 2,
        linestyle='-',alpha=1, label = i)

# ---- Plot glass data ----
a.plot(MI['SiO2'],MI['Na2O']+MI['K2O'],
    'ok', mfc = 'y', label="MI")
a.plot(MG['SiO2'],MG['Na2O']+MG['K2O'],
    'ok', mfc = 'b', label="MG")

a.set_xlim([40, 65])
a.set_ylim([1, 8])
a.legend()
```

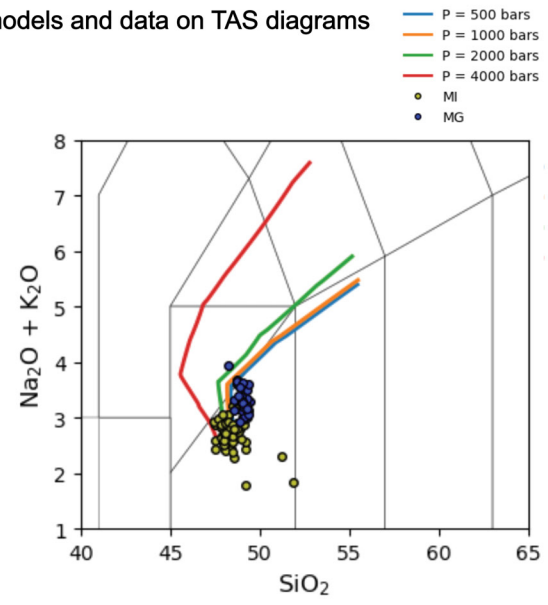


Figure 4: Isobaric crystallization calculations in PetThermoTools. [A] Isobaric crystallization calculations at 500, 1000, 2000, and 4000 bar are initiated from a single function in PetThermoTools, with the composition of an olivine hosted melt inclusion, collected from a submarine flow off the coast of Isla Fernandina, Galápagos, used as the starting composition. Melt inclusion data and matrix glass data from the submarine Isla Fernandina flows are taken from Koleszar et al. [2009] and Peterson et al. [2017]. [B] Using the in-built harker function we can inspect the composition of these models against the melt inclusion and matrix glass data. [C] Using pyrolite [Williams et al. 2020] we can plot the data and models on a Total Alkali versus Silica (TAS) diagram. A more detailed version of this example is available on ReadtheDocs.

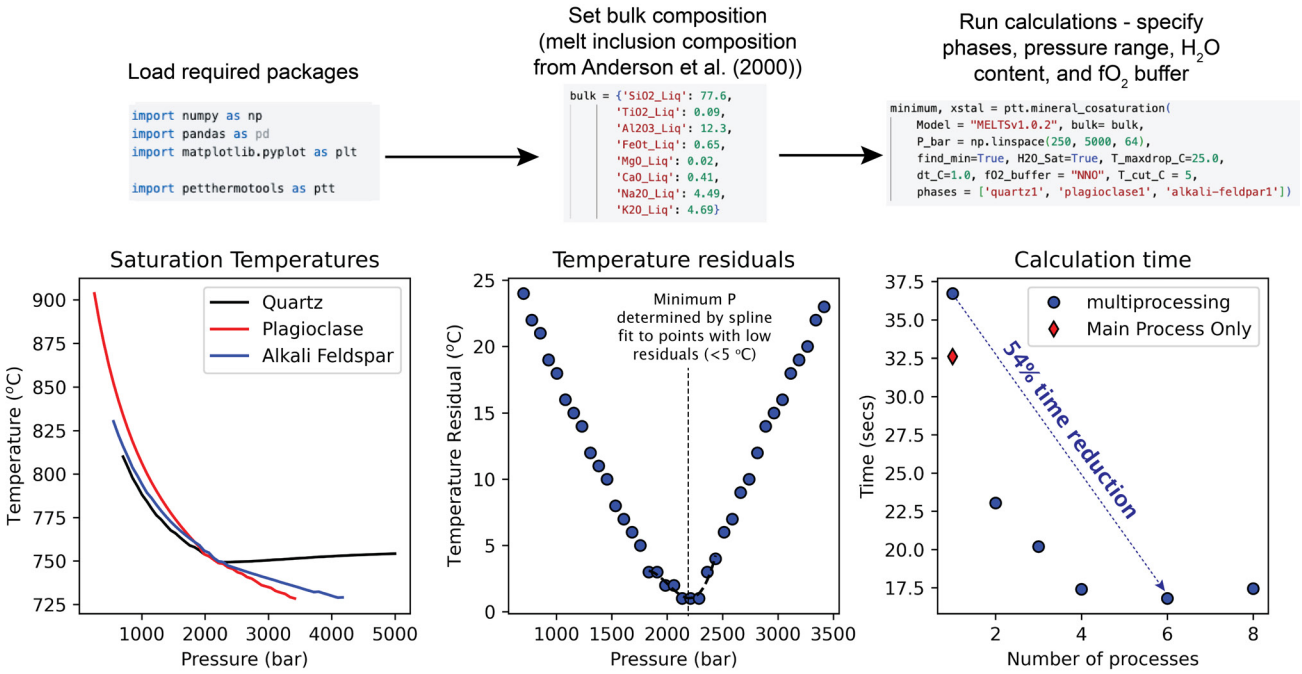


Figure 5: Outline of the code structure, bulk composition, function, and arguments used to perform barometry calculations in `PetThermoTools`. By varying the number of parallel processes we are also able to use these calculations to demonstrate the speed advantages gained by running calculations in parallel versus the time taken to run all calculations in the main Python process sequentially (red diamond). All calculations shown here were performed on a 2022 MacBook Pro with an Apple M2 chip (8 processors). A complete version of this example is available through ReadtheDocs.

We also provide several in-built plotting functions, allowing for easy visualization of model results. Several of these plotting methods are shown in the figures included in this paper with more examples (and descriptions of the options available) presented on ReadtheDocs (Figures 4, 6, 7).

4 Key improvements

During the conceptualization and development of `PetThermoTools` we identified three key targets for this resource that are crucial for expanding the application of thermodynamic modeling in petrological research and education: (i) integration of `PetThermoTools` with both `alphaMELTS` for `Python` and `MAGEMin` [Antoshechkina and Ghiorso 2018; Riel et al. 2022], allowing users to perform calculations with alternative thermodynamic models and facilitating easier comparison of different approaches; (ii) improving the stability of calculations, specifically developing better handling of calculation errors so that multiple calculations can be performed without failure (in prior software options, a single calculation failure would terminate a run and typically require a complete restart of the system/kernel); and (iii) integration with other `Python` packages developed for petrological research. These three features are key for many researchers using thermodynamic modeling, and will allow these models to be used directly within `Python` workflows.

4.1 Integration with alternative thermodynamic models

Comparisons of different thermodynamic models remain limited in the literature, largely due to the different software tools required to run the `MELTS` and `H&P` thermodynamic models [Jennings and Holland 2015; Hernández-Uribe et al. 2022; Wieser et al. 2025]. `PetThermoTools` was initially developed as an easy-to-access platform for thermodynamic calculations with the `MELTS` models. However, recognizing the potential impact of being able to run calculations with both the `MELTS` and `H&P` models within the same Jupyter Notebook, we established a method for initiating calculations in `Julia` through `MAGEMin_C` [Riel et al. 2022]. Currently, `PetThermoTools` only provides access to the Weller et al. [2024] and Green et al. [2025] thermodynamic models in `MAGEMin`. Set-up and installation of the `MAGEMin` code and a `Python`–`Julia` connection through `PetThermoTools` is simplified to a single notebook, with straightforward installation and update functions that ensure the underlying packages are compatible with the version of `PetThermoTools` being used. This minimizes the amount of prior coding experience required for new users.

The code structure of `PetThermoTools` allows users to switch between the `MELTS` and `H&P` thermodynamic models by changing the string variable passed to the `Model` key-word argument (Figures 6, 7). Consequently, we believe that this development will encourage future comparisons of different thermodynamic approaches, and potentially contribute to a systematic assessment of thermodynamic model performance across P-T-X space.

There are several functions in `PetThermoTools` that are only available if the calculations are performed through `alphaMELTS` for `Python`. This is not due to an issue in initiating calculations through MAGEMin, but refers to specific func-

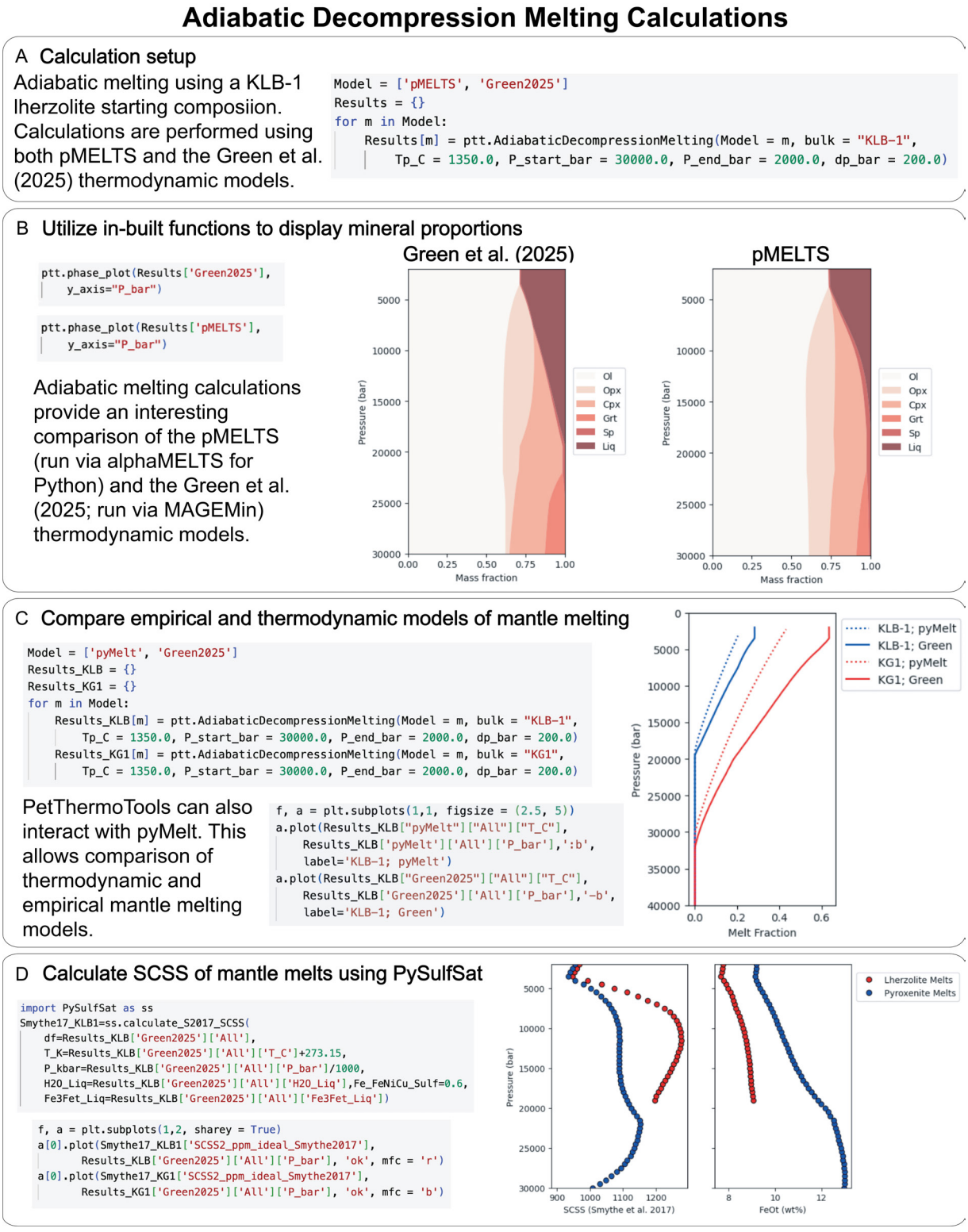


Figure 6: Adiabatic melting calculations in PetThermoTools. [A] PetThermoTools contains several 'saved' starting compositions, including the experimental KLB-1 lherzolite [Hirose and Kushiro 1993] and the KG1 pyroxenite [Kogiso et al. 1998]. Calculations here are performed using both pMELTS [Ghiorso et al. 2002] and the Green et al. [2025] thermodynamic models. [B] PetThermoTools provides a function for plotting up the phase assemblage of mantle melting calculations and this can be used to compare the results of the two thermodynamic models. [C] PetThermoTools can also call functions from the Python3 package pyMelt [Matthews et al. 2022], allowing comparison of thermodynamic and empirical parameterizations of mantle melting. [D] Easy integration of the PetThermoTools outputs with other Python packages allows the Sulfide Content at Sulfide Saturation (SCSS) to be calculated using PySulfSat [Wieser and Gleeson 2023].

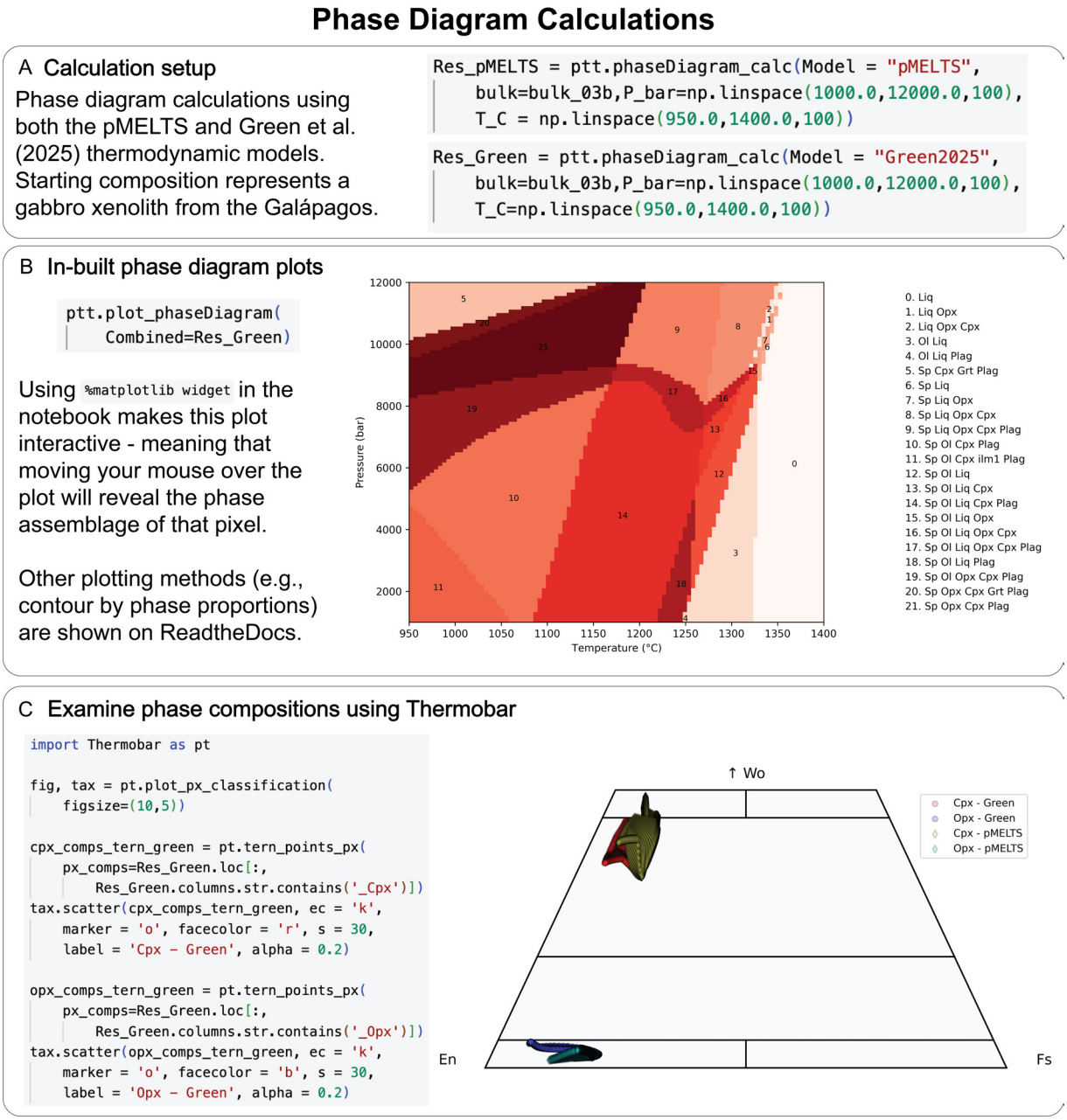


Figure 7: Phase diagram calculations using the composition of Galápagos gabbro xenolith 17MMSG03b from Gleeson et al. [2020] and Gleeson et al. [2025]. [A] Following the approach of Gleeson et al. [2025] we calculate the phase stability between 950 and 1400 °C and 1000 to 12000 bar. Calculations are performed using both the `pMELTS` [Ghiorso et al. 2002] and Green et al. [2025] thermodynamic models. [B] An in-built function can be used to visualize the phase diagrams. By using the `%matplotlib widget` in a Jupyter Notebook it is possible to make this plot interactive (so that the phase assemblage at any P-T point is shown as the user moves their mouse over the graph). [C] `PetThermoTools` outputs are designed to integrate seamlessly with `Thermobar` [Wieser et al. 2022b]. Here we use `Thermobar` to compare the clinopyroxene and orthopyroxene compositions predicted by the two different thermodynamic models using a pyroxene quadrilateral.

tions that require the presence of a H_2O-CO_2 mixed solubility model. A mixed H_2O-CO_2 solubility model is included in the latest versions of `rhyolite-MELTS` (v1.1.0 and v1.2.0) [Ghiorso and Gualda 2015], but the presence of CO_2 is not accounted for in the `H&P` thermodynamic models at present. Finally, we acknowledge that performing calculations in `MAGEMin` through `PetThermoTools` is associated with a decrease in computational efficiency (compared to running these calculations directly in `Julia`) owing to the time required to initialize and compile the `Julia` environment from a `Python` notebook.

4.2 Speed and stability improvements

One of the main weaknesses of thermodynamic modeling, specifically considering calculations performed with the `MELTS` family of thermodynamic models, is the frequency of model failures. In many `MELTS` software options, failures in model

convergence, that is, inability to find an equilibrium solution, can terminate the process and cause the software to crash. For example, automating 1000s of equilibrium crystallization calculations in MELTS for Excel is possible, but if the second calculation experiences an error or fails to converge on an equilibrium solution at any point of the calculation, the calculations will stop and no other calculations will be performed (Brad Pitcher personal communications [Gualda and Ghiorso 2015]). This reduces the ability for researchers to perform 100s to 1000s of models in a single run, limiting the application of Monte Carlo methods in thermodynamic modeling. Even some of the more advanced software options suffer from similar fragility to model failures. alphaMELTS for Python, for example, cannot re-initialize the libalphaMELTS C library after a failure has occurred. This means that if the calculation hits an error, the user would have to restart the Python kernel and reload all libraries and input parameters before restarting the models.

To address this, PetThermoTools utilizes Python's multiprocessing library to initiate and run MELTS calculations in separate 'child processes'. While MAGEMin does not suffer from the same limitations described above, we opt to initialize the processes and calculations in a similar way using the Julia Distributed module. Input data organization and compilation/formatting of model outputs takes place within the main Python and/or Julia process, but new child processes are initiated prior to any thermodynamic calculations (Figure 1). The key influence of this is that if a thermodynamic calculation failure occurs, PetThermoTools returns the model results up to the failure point, terminates the 'poisoned' process, and initiates a new child process to complete the remaining calculations. As a result, by using Python's multiprocessing library to initiate and run thermodynamic calculations in child processes, PetThermoTools protects the main process from model failures allowing large numbers of calculations to be performed without risk of software crashes.

It is important to note that initiating new child processes adds time to the calculations (likely 1–2 seconds depending on the computer). Consequently, if the user is only performing a single calculation (i.e. one fractional crystallization model), PetThermoTools provides the option to 'turn off' the use of child processes to minimize calculation times (this is not provided as the default option as calculation errors could restrict the potential for further MELTS calculations later in the Python script/Jupyter Notebook). This option is not available for instances involving multiple calculations, as parallel processing with the multiprocessing library can provide speed improvements for large numbers of calculations (in addition to the stability improvements outlined above). Specifically, when more than one model is initiated, PetThermoTools will create multiple child processes (up to the number of processors on the system's CPU, with additional restrictions built around the available system memory), distribute the calculations between these new processes, and perform the thermodynamic calculations in parallel. New processes are only initiated when model failures occur. This approach significantly reduces the total calculation time and maximizes the system CPU capabilities. In cases where the user wants to limit the number of processors used—e.g. so that their computer can be used for other applications while PetThermoTools runs in the background—they can manually select the extent of multiprocessing using the 'cores' key-word argument in all functions.

A simple demonstration of the speed advantages provided by PetThermoTools is seen when the 'mineral_cosaturation' function is used to assess the magma storage pressure of high-silica rhyolite following the method outlined by Gualda and Ghiorso [2014]. This method consists of running isobaric crystallization calculations at multiple pressures (64 different pressures in the example shown in Figure 5) to determine the pressure at which all specified phases are saturated at or close to the liquidus for the chosen melt composition. In Figure 5 we calculate the quartz - 2-feldspar co-saturation pressure for a melt inclusion from the Early Bishop Tuff [Anderson et al. 2000] and rerun the calculation multiple times steadily increasing the number of processes performed in parallel. Our results indicate that increasing the number of parallel processes to six (rather than running each model individually) can decrease computational time by ~54 %. The calculations shown in Figure 5 were performed using a 2022 MacBook Pro with the Apple M2 chip and eight logical processors. When eight calculations are performed in parallel a further 1–2 % speed advantage can be observed, but this is highly dependent on the number (and CPU cost) of other background processes running on the machine. Rerunning the calculations on a Lenovo P620 Workstation (with an Intel i9 12th generation chip with 32 logical processors) we found a maximum time reduction of ~57 %. However, we note that PetThermoTools is not built to prioritize the speed of calculations, but rather to protect the system from potential model failures and to provide access to thermodynamic modeling in Python for all users regardless of prior coding experience. Nevertheless, by simplifying the necessary set up procedure for thermodynamic modeling and allowing easy integration with other Python packages designed for petrological research (see Section 4.3), PetThermoTools will provide a significant overall speed benefit to most researchers using thermodynamic modeling as part of their workflow.

4.3 Integration with other petrological toolkits

Over the last few years there has been a rapid expansion in the number of Python-based, open-source packages designed to facilitate petrological research (e.g. VESIcal [Iacovino et al. 2021; Wieser et al. 2022a]; Thermobar [Wieser et al. 2022b]; pyMelt [Matthews et al. 2022]; PySulfSat [Wieser and Gleeson 2023]; Pyrolite [Williams et al. 2020]; and meltPT [McNab and Ball 2023]). There are hundreds of researchers utilizing these tools to examine, interpret, and visualize their data. Consequently, we designed PetThermoTools to integrate with as many of these different toolkits as possible, so that users can incorporate thermodynamic modeling with other aspects of their data analysis and visualization workflows. One of the key choices was to use the naming convention for different phases from Thermobar [Wieser et al. 2022b], specifically using the suffixes '_Liq', '_Cpx', '_Plag' etc. in the column headers for the output DataFrames. Consequently, results from PetThermoTools can be used in the Thermobar and

PySulfSat functions without any further data handling; i.e. output variables can be directly used as input parameters in these other packages.

Several examples of how PetThermoTools integrates with other Python packages are shown in figures here and on the associated ReadtheDocs site. First, we show how a few simple lines of code can be used to plot the result of crystallization models on a Total Alkali versus Silica (TAS) diagram (Figure 4). This utilizes the plotting capabilities of the package pyrolite [Williams et al. 2020]. In addition, we demonstrate how the outputs of PetThermoTools directly feed into the functions of Thermobar, allowing visualization of mineral composition on a pyroxene quadrilateral (Figure 7). We also use simple mantle melting calculations to demonstrate the comparison of different thermodynamic and empirical mantle melting models. This is achieved through the integration of PetThermoTools with some of the key methods in pyMelt [Matthews et al. 2022], allowing users to access the Matthews et al. [2021] parameterizations for melting of KLB-1 lherzolite and KG1 and G2 pyroxenites (Figure 6). Finally, we show how PetThermoTools outputs integrate with PySulfSat functions [Wieser and Gleeson 2023], facilitating easy calculation of the Sulfide Content at Sulfide Saturation (SCSS) from any PetThermoTools calculation (Figure 6).

5 PetThermoTools IN THE CLASSROOM

In addition to being a valuable tool for petrological research, PetThermoTools can be very useful in undergraduate and graduate level igneous petrology, volcanology, and geochemistry classes. Running simple thermodynamic calculations in class can give students first-hand experience investigating how the mantle melts, magmatic systems cool and crystallize, and degas upon eruption. The biggest barrier to the use of computer code in a classroom environment is often installation, given the range of computing literacy among students and operating systems available. At present, we have found the best solution is to get students to register for an account with the VICTOR server [Lev et al. 2025], which has alphaMELTS for Python pre-installed. On the PetThermoTools ReadTheDocs page, we provide two examples of class assignments/worksheets. The first introduces the concept of the liquidus, and how the liquidus temperature and phase change as a function of pressure, water content, and melt composition. This is integrated with Thermobar to allow students to investigate how these parameters influence the melt viscosity [Wieser et al. 2022b]. Then, after developing intuition for what P-T-X conditions cause certain phases to stabilize, students run fractional crystallization models at varying oxygen fugacities and water contents. This allows students to see first hand that calc-alkaline differentiation trends are favoured by higher water contents and oxygen fugacities, while dryer and more reducing conditions form tholeiitic differentiation paths. A second practical combines PetThermoTools and VESIcal [Iacovino et al. 2021], allowing students to investigate the controls on volatile solubility, and track volatile exsolution during ascent from the mantle (first boiling), differentiation in the crust (second boiling), and finally ascent towards the surface (fragmentation). We have found that allowing students to model these processes using PetThermoTools, with the ability to easily and quickly adjust different parameters, can enhance their understanding of processes operating within volcanic systems beyond what is possible from lectures alone.

6 FUTURE WORK

Moving forward, PetThermoTools will continue to develop and expand. Specifically, we will work with existing MELTS and MAGEMin users to ensure that all common thermodynamic calculations are available with clear documentation and example notebooks. We will solicit feedback from the thermodynamic modeling community (through workshops, conferences, and the alphaMELTS Discord server) to help direct further expansion of the PetThermoTools functionality. This will be achieved through the creation of new methods, such as the development of a multi-component mantle melting function, or via expansion and customization of existing methods; for example, the ability to calculate P-fO_2 or T-H_2O phase diagrams. We also plan to develop methods for performing trace element and stable isotope fractionation calculations alongside PetThermoTools models. Planned developments to PetThermoTools are listed above and represent some of the key methods that have been requested by existing PetThermoTools users. Furthermore, as PetThermoTools and the underlying alphaMELTS for Python and MAGEMin packages are open source, we encourage more advanced users to reach out and collaborate with the development team. In doing so, we hope that new methods using the underlying code packages can be incorporated into the framework of PetThermoTools and, therefore, become accessible to other users.

7 CONCLUSIONS

We expect that PetThermoTools will greatly benefit researchers and educators in igneous petrology, from undergraduate students to established professors. The code is designed to be accessible to all users, with a number of well-documented example Jupyter Notebooks available via ReadTheDocs. For more advanced users, the application of parallel processing workflows has proven successful in reducing the time required to perform thermodynamic calculations and minimizes the influence of calculation failures, increasing the potential of utilizing Monte Carlo methods alongside MELTS modeling and other applications. In addition, we have specifically designed the PetThermoTools model outputs so that they seamlessly integrate with other Python-based petrological tools (e.g. Thermobar [Wieser et al. 2022b]), allowing thermodynamic modeling to be directly incorporated into user workflows. Finally, PetThermoTools represents one of the first software packages to provide easy access to both the MELTS and H&P thermodynamic models, facilitating easier comparison of different thermodynamic approaches.

AUTHOR CONTRIBUTIONS

M.G. led the development of PetThermoTools and writing of this manuscript. P.W. provided extensive testing of functions

as they are developed and aided with distribution through PyPI and ReadTheDocs, including creating example workflows, YouTube videos and education materials. P.A. is the lead developer of `alphaMELTS` for `Python`, N.R. is the lead developer of `MAGEMin`, and both were key to their implementation in this package. All authors contributed to editing of this manuscript.

Acknowledgements

M.G., P.W., and P.A., acknowledge funding and support from the National Science Foundation (EAR-2514734 and EAR-2514733). PW acknowledges funding from a Heising Simons Faculty Fellowship. N.R. acknowledges the German Research Foundation (DFG) - project number #521637679.

Data Availability

All files are available on GitHub https://github.com/gleesonm1/PetThermoTools. Further documentation is available at ReadTheDocs (https://petthermotools.readthedocs.io/en/latest/index.html.

Copyright notice

References

Anderson, A. T., A. M. Davis, and F. Lu (2000). "Evolution of Bishop Tuff rhyolitic magma based on melt and magnetite inclusions and zoned phenocrysts". *Journal of Petrology* 41(3), pages 449–473. DOI: https://doi.org/10.1093/petrology/41.3.449.

Antoshechkina, P. M. and M. S. Ghiorso (2018). "MELTS for MATLAB: A new educational and research tool for computational thermodynamics". *AGU Fall Meeting Abstracts*. Volume 2018, ED44B–23.

Ariskin, A. A., M. Y. Frenkel, G. S. Barmina, and R. L. Nielsen (1993). "COMAGMAT: a Fortran program to model magma differentiation processes". *Computers & Geosciences* 19(8), pages 1155–1170. DOI: https://doi.org/10.1016/0098-3004(93)90020-6.

Ariskin, A. A., K. A. Bychkov, G. S. Nikolaev, and G. S. Barmina (2018). "The COMAGMAT-5: modeling the effect of Fe–Ni sulfide immiscibility in crystallizing magmas and cumulates". *Journal of petrology* 59(2), pages 283–298. DOI: https://doi.org/10.1093/petrology/egy026.

Bell, A. S., L. E. Waters, and M. Ghiorso (2025). "The olivine-spinel-aSiO2melt (OSaS) oxybarometer: A new method for evaluating magmatic oxygen fugacity in olivine-phyric basalts". *American Mineralogist* 110(1), pages 48–64. DOI: 10.2138/am-2023-9021.

Blundy, J. and K. Cashman (2005). "Rapid decompression-driven crystallization recorded by melt inclusions from Mount St. Helens volcano". *Geology* 33(10), pages 793–796. DOI: https://doi.org/10.1130/G21668.1.

Blundy, J., K. Cashman, and M. Humphreys (2006). "Magma heating by decompression-driven crystallization beneath andesite volcanoes". *Nature* 443(7107), pages 76–80. DOI: https://doi.org/10.1038/nature05100.

Boulanger, M. and L. France (2023). "Cumulate formation and melt extraction from mush-dominated magma reservoirs: the melt flush process exemplified at mid-ocean ridges". *Journal of Petrology* 64(2), egad005. DOI: https://doi.org/10.1093/petrology/egad005.

Brown, E. L., K. D. Petersen, and C. E. Lesher (2020). "Markov chain Monte Carlo inversion of mantle temperature and source composition, with application to Reykjanes Peninsula, Iceland". *Earth and Planetary Science Letters* 532, page 116007. DOI: https://doi.org/10.1016/j.epsl.2019.116007.

Brunelli, D., A. Cipriani, and E. Bonatti (2018). "Thermal effects of pyroxenites on mantle melting below mid-ocean ridges". *Nature Geoscience* 11(7), pages 520–525. DOI: https://doi.org/10.1038/s41561-018-0139-z.

Candioti, L. G., C. Nathwani, and C. Chelle-Michou (2025). "Validating thermodynamic models of arc-magma differentiation and training neural networks for rapid thermodynamic property inference". DOI: 10.22541/essoar.176169621.16239367/v1.

Connolly, J. (2009). "The geodynamic equation of state: what and how". *Geochemistry, geophysics, geosystems* 10(10). DOI: https://doi.org/10.1029/2009GC002540.

Danyushevsky, L. V. and P. Plechov (2011). "Petrolog3: Integrated software for modeling crystallization processes". *Geochemistry, Geophysics, Geosystems* 12(7). DOI: https://doi.org/10.1029/2011GC003516.

De Capitani, C. and K. Petrakakis (2010). "The computation of equilibrium assemblage diagrams with Theriak/Domino software". *American mineralogist* 95(7), pages 1006–1016. DOI: https://doi.org/10.2138/am.2010.3354.

Ghiorso, M. S. and B. W. Evans (2008). "Thermodynamics of rhombohedral oxide solid solutions and a revision of the Fe-Ti two-oxide geothermometer and oxygen-barometer". *American Journal of science* 308(9), pages 957–1039. DOI: https://doi.org/10.2475/09.2008.01.

Ghiorso, M. S. and G. A. Gualda (2015). "An H 2 O–CO 2 mixed fluid saturation model compatible with rhyolite-MELTS". *Contributions to Mineralogy and Petrology* 169, pages 1–30. DOI: https://doi.org/10.1007/s00410-015-1141-8.

Ghiorso, M. S., M. M. Hirschmann, P. W. Reiners, and V. C. Kress III (2002). "The pMELTS: A revision of MELTS for improved calculation of phase relations and major element partitioning related to partial melting of the mantle to 3 GPa". *Geochemistry, Geophysics, Geosystems* 3(5), pages 1–35. DOI: https://doi.org/10.1029/2001GC000217.

Ghiorso, M. S. and R. O. Sack (1995). "Chemical mass transfer in magmatic processes IV. A revised and internally consistent thermodynamic model for the interpolation and extrapolation of liquid-solid equilibria in magmatic systems at elevated temperatures and pressures". *Contributions to*

Mineralogy and Petrology 119, pages 197–212. DOI: https://doi.org/10.1007/BF00307281.

Gleeson, M., P. E. Wieser, C. L. DeVitre, S. C. Shi, M.-A. Millet, D. D. Muir, M. J. Stock, and J. Lissenberg (2025). "Persistent high-pressure magma storage beneath a near-ridge ocean island volcano (Isla Floreana, Galápagos)". *Journal of Petrology* 66(5), egaf031. DOI: https://doi.org/10.1093/petrology/egaf031.

Gleeson, M. L. and S. A. Gibson (2021). "Insights Into the Nature of Plume-Ridge Interaction and Outflux of H2O From the Galápagos Spreading Center". *Geochemistry, Geophysics, Geosystems* 22(11), e2020GC009560. DOI: https://doi.org/10.1029/2020GC009560.

Gleeson, M. L., S. A. Gibson, and M. J. Stock (2020). "Upper mantle mush zones beneath low melt flux ocean island volcanoes: insights from Isla Floreana, Galápagos". *Journal of Petrology* 61(11-12), egaa094. DOI: https://doi.org/10.1093/petrology/egaa094.

Gleeson, M. L., C. J. Lissenberg, and P. M. Antoshechkina (2023). "Porosity evolution of mafic crystal mush during reactive flow". *Nature Communications* 14(1), page 3088. DOI: https://doi.org/10.1038/s41467-023-38136-x.

Gleeson, M. L., M. J. Stock, D. M. Pyle, T. A. Mather, W. Hutchison, G. Yirgu, and J. Wade (2017). "Constraining magma storage conditions at a restless volcano in the Main Ethiopian Rift using phase equilibria models". *Journal of Volcanology and Geothermal Research* 337, pages 44–61. DOI: https://doi.org/10.1016/j.jvolgeores.2017.02.026.

Green, E. C., T. J. Holland, R. Powell, O. M. Weller, and N. Riel (2025). "Corrigendum to: Melting of Peridotites through to Granites: a Simple Thermodynamic Model in the System KNCFMASHTOCr, and, a Thermodynamic Model for the Subsolidus Evolution and Melting of Peridotite." *Journal of Petrology* 66(1), egae079. DOI: https://doi.org/10.1093/petrology/egae079.

Gualda, G. A. and M. S. Ghiorso (2014). "Phase-equilibrium geobarometers for silicic rocks based on rhyolite-MELTS. Part 1: Principles, procedures, and evaluation of the method". *Contributions to Mineralogy and Petrology* 168, pages 1–17. DOI: https://doi.org/10.1007/s00410-014-1033-3.

– (2015). "MELTS_Excel: A Microsoft Excel-based MELTS interface for research and teaching of magma properties and evolution". *Geochemistry, Geophysics, Geosystems* 16(1), pages 315–324. DOI: https://doi.org/10.1002/2014GC005545.

Gualda, G. A., M. S. Ghiorso, R. V. Lemons, and T. L. Carley (2012). "Rhyolite-MELTS: a modified calibration of MELTS optimized for silica-rich, fluid-bearing magmatic systems". *Journal of Petrology* 53(5), pages 875–890. DOI: https://doi.org/10.1093/petrology/egr080.

Harmon, L. J., J. Cowlyn, G. A. Gualda, and M. S. Ghiorso (2018). "Phase-equilibrium geobarometers for silicic rocks based on rhyolite-MELTS. Part 4: plagioclase, orthopyroxene, clinopyroxene, glass geobarometer, and application to Mt. Ruapehu, New Zealand". *Contributions to Mineralogy and Petrology* 173, pages 1–20. DOI: https://doi.org/10.1007/s00410-017-1428-z.

Heinonen, J. S., A. V. Luttinen, F. J. Spera, and W. A. Bohrson (2019). "Deep open storage and shallow closed transport system for a continental flood basalt sequence revealed with Magma Chamber Simulator". *Contributions to Mineralogy and Petrology* 174, pages 1–18. DOI: https://doi.org/10.1007/s00410-019-1624-0.

Hernández-Uribe, D., F. J. Spera, W. A. Bohrson, and J. S. Heinonen (2022). "A comparative study of two-phase equilibria modeling tools: MORB equilibrium states at variable pressure and H2O concentrations". *American Mineralogist* 107(9), pages 1789–1806. DOI: https://doi.org/10.2138/am-2022-8211.

Hirose, K. and I. Kushiro (1993). "Partial melting of dry peridotites at high pressures: determination of compositions of melts segregated from peridotite using aggregates of diamond". *Earth and Planetary Science Letters* 114(4), pages 477–489. DOI: https://doi.org/10.1016/0012-821X(93)90077-M.

Holland, T. J., E. C. Green, and R. Powell (2018). "Melting of peridotites through to granites: a simple thermodynamic model in the system KNCFMASHTOCr". *Journal of Petrology* 59(5), pages 881–900. DOI: https://doi.org/10.1093/petrology/egy048.

Iacovino, K., S. Matthews, P. E. Wieser, G. Moore, and F. Bégué (2021). "VESIcal Part I: An open-source thermodynamic model engine for mixed volatile (H2O-CO2) solubility in silicate melts". *Earth and Space Science* 8(11), e2020EA001584. DOI: https://notebooks.agu.org/articles/NN0001.

Jennings, E. S. and T. J. Holland (2015). "A simple thermodynamic model for melting of peridotite in the system NCFMASOCr". *Journal of Petrology* 56(5), pages 869–892. DOI: https://doi.org/10.1093/petrology/egv020.

Kogiso, T., K. Hirose, and E. Takahashi (1998). "Melting experiments on homogeneous mixtures of peridotite and basalt: application to the genesis of ocean island basalts". *Earth and Planetary Science Letters* 162(1-4), pages 45–61. DOI: https://doi.org/10.1016/S0012-821X(98)00156-3.

Koleszar, A., A. Saal, E. Hauri, A. Nagle, Y. Liang, and M. Kurz (2009). "The volatile contents of the Galapagos plume; evidence for H2O and F open system behavior in melt inclusions". *Earth and Planetary Science Letters* 287(3-4), pages 442–452. DOI: https://doi.org/10.1016/j.epsl.2009.08.029.

Lambart, S. (2017). "No direct contribution of recycled crust in Icelandic basalts". *Geochemical Perspectives Letters* 4, pages 7–12. DOI: http://dx.doi.org/10.7185/geochemlet.1728.

Lev, E., S. Krasnoff, S. Charbonnier, C. Connor, A. Patra, and A. Mullins (2025). "VICTOR—A new cyberinfrastructure for volcanology". *Volcanica* 8(2), pages 563–582. DOI: 10.30909/vol/ikoj4933.

Matthews, S., K. Wong, and M. Gleeson (2022). "pyMelt: An extensible Python engine for mantle melting calculations". *Volcanica* 5(2), pages 469–475. DOI: https://doi.org/10.30909/vol.05.02.469475.

Matthews, S., K. Wong, O. Shorttle, M. Edmonds, and J. Maclennan (2021). "Do olivine crystallization temperatures faithfully record mantle temperature variability?" *Geochemistry, Geophysics, Geosystems* 22(4), e2020GC009157. DOI: https://doi.org/10.1029/2020GC009157.

McNab, F. and P. Ball (2023). "meltPT: A Python package for basaltic whole-rock thermobarometric analysis with application to Hawai¯i". *Volcanica* 6(1), pages 63–76. DOI: https://doi.org/10.30909/vol.06.01.6376.

Otto, T., G. Stevens, M. J. Mayne, and J.-F. Moyen (2023). "Phase equilibrium modelling of partial melting in the upper mantle: A comparison between different modelling methodologies and experimental results". *Lithos* 444, page 107111. DOI: https://doi.org/10.1016/j.lithos.2023.107111.

Peterson, M., A. Saal, M. Kurz, E. Hauri, J. Blusztajn, K. Harpp, R. Werner, and D. Geist (2017). "Submarine basaltic glasses from the Galapagos Archipelago: determining the volatile budget of the mantle plume". *Journal of Petrology* 58(7), pages 1419–1450. DOI: https://doi.org/10.1093/petrology/egx059.

Riel, N., B. J. Kaus, E. Green, and N. Berlie (2022). "MAGEMin, an efficient Gibbs energy minimizer: application to igneous systems". *Geochemistry, Geophysics, Geosystems* 23(7), e2022GC010427. DOI: https://doi.org/10.1029/2022GC010427.

Thermoengine (2022). *ThermoEngine: Software for Model Building and Computational Thermodynamics Supporting Applications in the Earth Sciences*. Version 1.0.0. DOI: 10.5281/zenodo.6527840.

Tramontano, S., G. A. Gualda, and M. S. Ghiorso (2017). "Internal triggering of volcanic eruptions: tracking overpressure regimes for giant magma bodies". *Earth and Planetary Science Letters* 472, pages 142–151. DOI: https://doi.org/10.1016/j.epsl.2017.05.014.

Weller, O. M., T. J. Holland, C. R. Soderman, E. C. Green, R. Powell, C. D. Beard, and N. Riel (2024). "New thermodynamic models for anhydrous alkaline-silicate magmatic systems". *Journal of Petrology* 65(10), egae098. DOI: https://doi.org/10.1093/petrology/egae098.

Wieser, P. E. and M. Gleeson (2023). "PySulfSat: An open-source Python3 tool for modeling sulfide and sulfate saturation". *Volcanica* 6(1), pages 107–127. DOI: 10.30909/vol.06.01.107127.

Wieser, P. E., M. L. M. Gleeson, S. Matthews, C. DeVitre, and E. Gazel (2025). "Determining the pressure-temperature-composition (P-T-X) conditions of magma storage". Edited by A. Anbar and D. Weis, pages 83–151. DOI: https://doi.org/10.1016/B978-0-323-99762-1.00024-3.

Wieser, P. E., K. Iacovino, S. Matthews, G. Moore, and C. Allison (2022a). "VESIcal: 2. A Critical Approach to Volatile Solubility Modeling Using an Open-Source Python3 Engine". *Earth and Space Science* 9(2), e2021EA001932. DOI: https://doi.org/10.1029/2021EA001932.

Wieser, P. E., M. Petrelli, J. Lubbers, E. Wieser, S. Ozaydin, A. Kent, and C. Till (2022b). "Thermobar: An open-source Python3 tool for thermobarometry and hygrometry". *Volcanica* 5(2), pages 349–384. DOI: 10.30909/vol.05.02.349384.

Williams, M. J., L. Schoneveld, Y. Mao, J. Klump, J. Gosses, H. Dalton, A. Bath, and S. Barnes (2020). "pyrolite: Python for geochemistry". *Journal of Open Source Software* 5(50), page 2314. DOI: 10.21105/joss.02314.