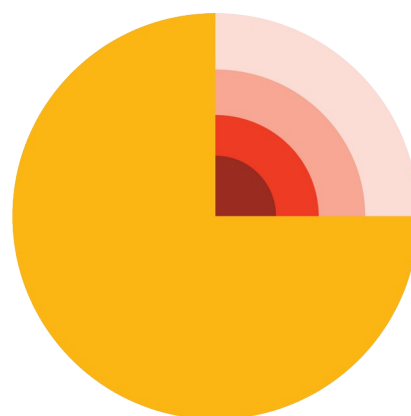




ChEESE

Deliverable D4.1 Initial report on workflow setups and workflow auditing



Funded by the European Union. This work has received funding from the European High Performance Computing Joint Undertaking (JU) and Spain, Italy, Iceland, Germany, Norway, France, Finland and Croatia under grant agreement No 101093038.



ChEESE-2P Deliverable

D4.1 Initial report on workflow setups and workflow auditing

Document Information

Contract Number	101093038
Project Website	https://cheese2.eu
Contractual Deadline	30/06/2024 (M18)
Activity	WP4
Dissemination Level	PUBLIC
Nature	R

Abstract

This deliverable introduces the methodology to collect and present details for the simulation workflows in ChEESE-2P. We report the details for workflows around pilot demonstrators and derived simulation cases available up to the current point. Further, initial results from workflow analysis and optimization activities are presented.

I. Copyright

Copyright Members of ChEESE collaboration, 2023-2026.

II. Delivery Slip

Role	Name	Partner/Activity
Author	Christoph Niethammer	HLRS
Contributor 1	Bin Song	HLRS
Contributor 2	PD/SC leads	ALL
Reviewer 1	Erwan Raffin	BA
Reviewer 2	Louise Cordrie	INGV

III. Version control

Version	Date	Description of change	Author/Partner
V1	11/06/2024	Initial draft	Christoph Niethammer
V2	21/06/2024	Second draft	Christoph Niethammer
V3	26/06/2024	Revised version	Christoph Niethammer
V4	28/06/2024	Final revised version	ALL

IV. List of Acronyms

Acronym	Description
SC	Simulation Case
PD	Pilot Demonstrator
JIT compiler	Just In Time compiler
MPI	Message Passing Interface
UC	Urgent Computing
WP	Work Package

Table of Contents

Introduction.....	5
Overall status of workflows.....	5
Relationship with other WPs.....	6
Characterisation of workflows.....	8
Notation.....	8
Workflow descriptions.....	10
PD1 (SC1.1, SC1.2, SC1.3).....	10
PD2 (SC2.1).....	11
PD4 (SC4.1).....	11
PD5 (SC5.1, SC5.2).....	12
PD6 (SC6.1, SC6.2).....	15
PD9 (SC9.1, SC9.2).....	16
Initial Workflow analysis and improvements.....	18
Initial workflow analysis and benchmarking for SC6.1.....	18
Workflow optimizations for SC6.1.....	19
1) Optimization of ASCtoSTL.py.....	19
2) Optimization of snappyHexMesh.....	20
Summary and general findings around workflow activities in WP4.....	23
References.....	24

Introduction

The Pilot Demonstrators (PD) and related Simulation Cases (SC) in ChEESE-2P couple multiple highly-scalable codes together with pre- and postprocessing tools to complex simulation workflows. Therefore, targeting exascale, not only the optimization of the codes is important but also the workflows and all components within them have to be prepared and adapted accordingly.

In the workflows, datasets have to be transformed and processed often multiple times between the various workflow components to obtain the intended final results. Hence, the handling of input and output datasets with scripts, tools, and infrastructure is an essential part. At exascale this part of the ecosystem now faces challenges, e.g., either because of the sheer size of a few large or by the large amount of smaller data sets to be processed. Therefore, beside the scalability of the main codes, also the surrounding ecosystem has to be addressed.

One of the objectives of work package 4 (WP4) is to address these challenges at the simulation workflow level that go beyond the main application codes. Specifically, task T4.1 “Workflow implementation and optimization” will implement and optimize simulation workflows as a whole. Therefore, workflow audits will be performed to identified bottlenecks. Those bottlenecks will then be analysed in more detail to address them with respective optimizations.

The workflows in ChEESE-2P are closely related to the simulation cases (SC). The SCs are developed in the context of WP6 which started in M12. The SCs are now defined and reported in deliverable D6.1 at the same time as this document. However, up to now not all SCs are at the state that they can already provide a setup or an in-depth workflow description at a level enabling the auditing as targeted by T4.1.

The purpose of this document is therefore to present the initial information about planned workflow setups and show initial workflow auditing results as well as first workflow optimisations for some of them.

Overall status of workflows

As of now, all PDs/SCs have drafted their workflows. These drafts are included in D6.1 and describe at a very high level the general procedures, what should be simulated as well as the expected outputs. The corresponding workflows for all the PDs and SCs are then detailed as a next step. Out of the 9 PDs and corresponding SCs, 6 detailed their workflows up to the date of this deliverable. One workflow is already

implemented and could be analysed as well as optimized within the reporting period. An overview of the workflows related to all the PDs and SCs is shown in Table 1. We distinguish here between *capability* (referring to a view large monolithic simulation runs) and *capacity* (referring to loosely to tightly coupled aggregates, or ensemble runs). Workflows can also fall into the *Urgent Computing (UC)* category, requiring them to run in emergency contexts with a target on the reduction of time-to-solution.

No	Name	category	UC	HPC systems	status				
					drafted	details	implemented	analyzed	optimized
PD1	Extreme-scale modeling of seismic hazards				x	x			
	SC1.1 Anthropogenic noise and seismicity, wave propagation at higher frequencies in Rade d'Hyères	capability		Leonardo	x				
	SC1.2 Next generation of earthquake simulators for the Hellenic arc and Iceland	capacity		Leonardo, LUMI	x				
	SC1.3 Physics-based PSHA and urgent UQ for Iceland	capacity	x	MN5	x	x			
PD2	High-resolution seismic inversion and tomography				x	x			
	SC2.1 Seismic tomography of the Adriatic-Ionian region	capability		Leonardo	x				
PD3					x				
	SC3.1 The Global Tsunami Hazard Map	capacity		Leonardo	x				
PD4	Complex multi-source tsunami modeling				x	x			
	SC4.1 Multi-source local high-resolution scenario with emphasis on the 2022 Tonga case	capcability	x	Leonardo	x				
PD5	Ensemble-based volcanic dispersal across multiple scales				x	x			
	SC5.1 Towards the European tephra hazard map	capacity		MN5	x	x			
	SC5.2 Volcanic dispersal at local scale: a reanalysis of the 2021 La Palma case	capacity	x	MN5	x	x			
PD6					x	x			
	SC6.1 Urgent high-resolution, 3D multiphase flow simulation of phreatic eruptions at Vulcano	capability	x	LUMI	x	x	x	x	x
	SC6.2 Long-term probabilistic hazard maps for phreatic eruptions at Vulcano island	capacity		LUMI	x	x			
PD7					x				
	SC7.1 MHD turbulence closure models in planetary cores using AI	capacity		MN5, LUMI, VEGA	x				
PD8					x				
	SC8.1 Magmatic systems and induced landslides: the Etna Case	capability		LUMI	x				
	SC8.2 3D evolution of tectonic deformation as driver of seismic hazard in the Hellenic arc	capacity		LUMI	x				
PD9	Glacial outburst floods				x	x			
	SC9.1 Eastern Skaftá cauldron GLOF simulation, Iceland	capacity		LUMI	x				
	SC9.2 Calving glacier hazard in Greenland	capacity		LUMI	x				

Table 1: Overview of PD and SC workflow activities

Relationship with other WPs

The details of the Simulation Cases are part of WP6. For their detailed definitions we refer here to the related deliverable D6.1 (Simulation Cases and trials definition, delivered at M18).

The analysis and optimizations of the components related to the main simulation codes is performed in WP2. An initial analysis of these codes was performed in D2.1 (First code audit report, delivered at M13).

Characterisation of workflows

Notation

Early on in the project, Pilot Demonstrator (PD) and Simulation Case (SC) owners were asked to describe the expected workflows in detail. This is a crucial step for the workflow implementation and workflow analysis. Of interest are here especially the various components, i.e., codes and scripts that are involved, the inter-component-dependencies arising from their input and output data, as well as the systems where the various components will be executed on, e.g., workstations or EuroHPC systems. Therefore, a schematic notation to describe the workflows with all the relevant information was developed.

Figure 1 shows the template example that was provided to the PD and SC owners to describe their simulation workflows. In Figure 1

- blue boxes represent components, i.e., codes or scripts that process data and contain information about
 - the programming language the code or script is written in, e.g., python, Julia, C++, Fortran, Rust, ...
 - the parallel programming model the code or script uses, e.g., mpi4py, OpenMP, MPI, ...
 - the system on which the code or script should run on, e.g., workstation, EuroHPC system, ...
- yellow boxes represent input/output files or databases and contain information about
 - the approximate size
 - for files the file type or data format, e.g., HDF5, json, csv, binary data, ...
 - for databases the database type, e.g., MySQL, ElasticSearch, ...
- grey boxes represent manual actions from users and contain a short description of the action, e.g., “check data for convergence”
- red boxes were asked to be added and describe already known bottlenecks in the workflow.

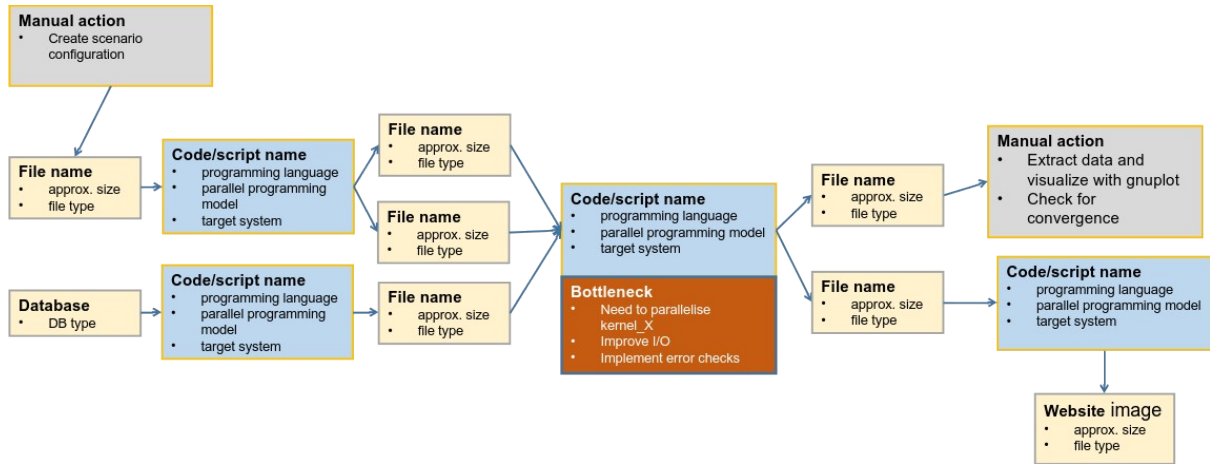


Figure 1: Example showcasing the ChEESE-2P workflow notation

Workflow descriptions

In the following workflow descriptions for some of the ChEESE-2P PDs and SCs are presented. They differ in the level of detail depending on the current state of their implementation. Some of them may just outline the general workflow structure, others have all components already implemented and set up. The detailed descriptions, scientific methods, and use cases behind the various SCs can be found in deliverable D6.1.

PD1 (SC1.1, SC1.2, SC1.3)

PD1 “Extreme-scale modeling of seismic hazards” derives the three following simulation cases:

- SC1.1 “Anthropogenic noise and seismicity, wave propagation at higher frequencies in Rade d'Hyères”
- SC1.2 “Next generation of earthquake simulators for the Hellenic arc and Iceland”
- SC1.3 “Physics-based PSHA and urgent UQ for Iceland”. The workflows are based around the main codes SeisSol, ExaHyPE, SPECFEM3D and Tandem.

Figure 2 shows the related workflow around SPECFEM3D. As can be seen the workflow includes extremely large input datasets up to 1 TB. The handling of these files in the pre- and postprocessing will especially be important for SC1.3, which targets UC services.

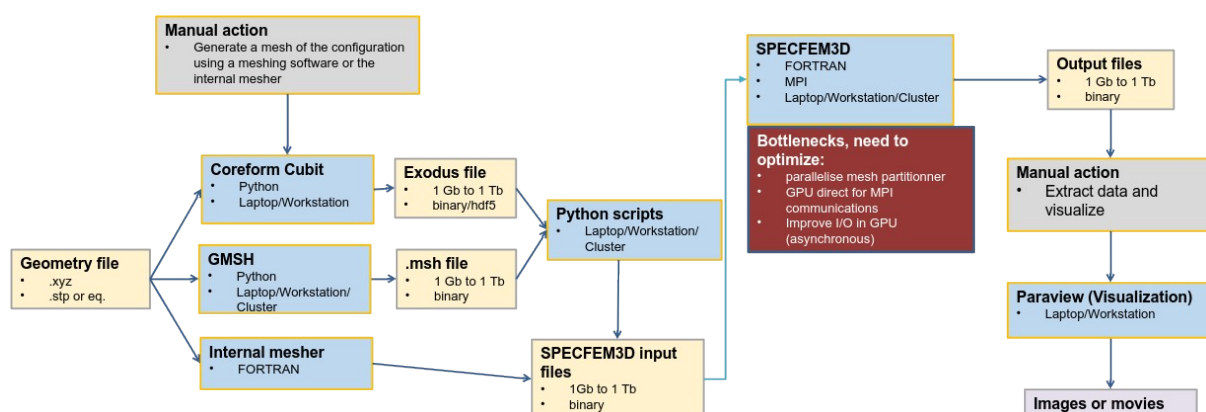


Figure 2: PD1 workflow using SPECFEM3D

The respective workflow for SC1.3 using SeisSol is already more detailed and can be seen in Figure 3.

The details for SC1.1 and SC1.2 are under progress.



Figure 3: SC1.3 workflow using SeisSol

PD2 (SC2.1)

PD2 “High-resolution seismic inversion and tomography” and the derived simulation case SC2.1 “Seismic tomography of the Adriatic-Ionian region” are built around the SPECFEM3D code and focuses on improvements in seismic imaging. The workflow shall enable improved precision capabilities. The workflow is outlined in Figure 4 and comes only with a few pre- and postprocessing components. However, the main challenge of this workflow will arise from the size of the input and output data. As file format, the well-established HDF5 file format is intended here.

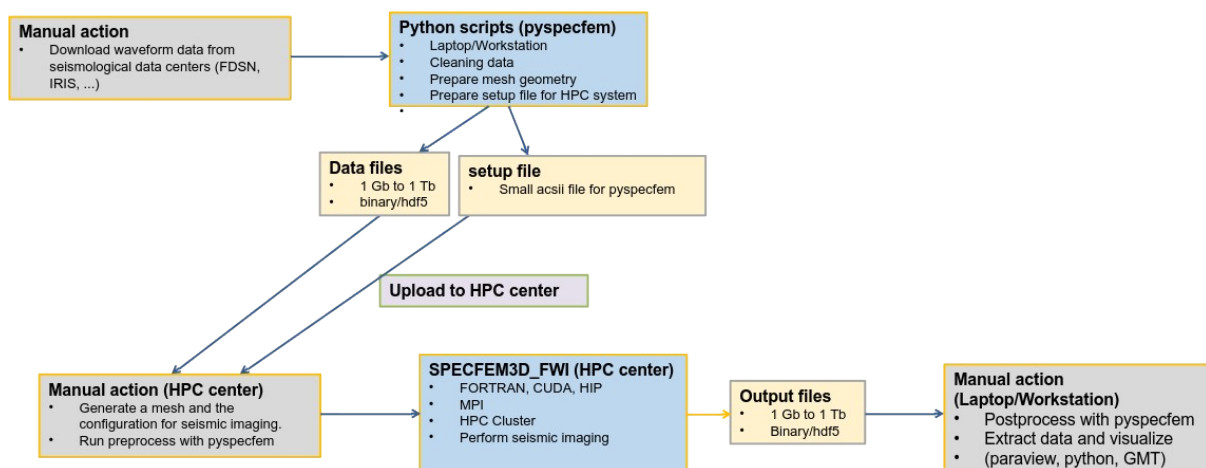


Figure 4: PD2 workflow

PD4 (SC4.1)

PD4 “Complex multi-source tsunami modeling” represents a pilot demonstrator that aims to connect different physics of tsunami generation

into a joint representation. The workflow will focus on interoperability between different flagship codes. The codes will be executed in separate runs, providing outputs that allow coupling different physics. An initial sketch of the workflow is shown in Figure 5.

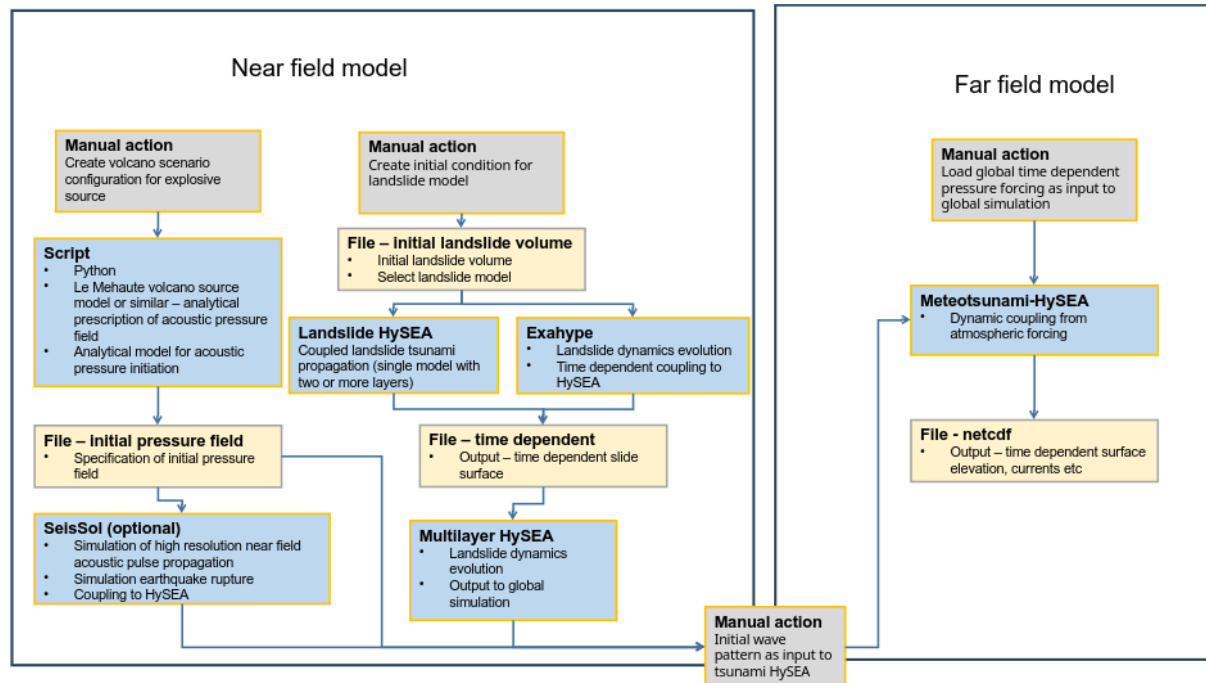


Figure 5: PD4 workflow sketch

PD5 (SC5.1, SC5.2)

PD5 “Ensemble-based volcanic dispersal across multiple scales” intends to implement an automated forecast system for atmospheric dispersal and ground deposition of volcanic tephra (fallout). This volcanic hazard affects civil aviation and local communities settled around the volcano. The workflow configures and runs an ensemble of FALL3D model realizations, potentially assimilating in the model different sources of data (ground-based or satellite observations), in order to have probabilistic and/or deterministic forecasts for the next 48h time window.

The overall workflow for PD5 is shown in Figure 6. The workflow includes two sub workflows at the beginning to prepare input data: one responsible for preparing meteorological input data (METEO) and one to set up the emission source parameters (ESP). Those are then used as input data in the main simulation performed with FALL3D. The details of the preprocessing sub workflows are shown in Figure 7.

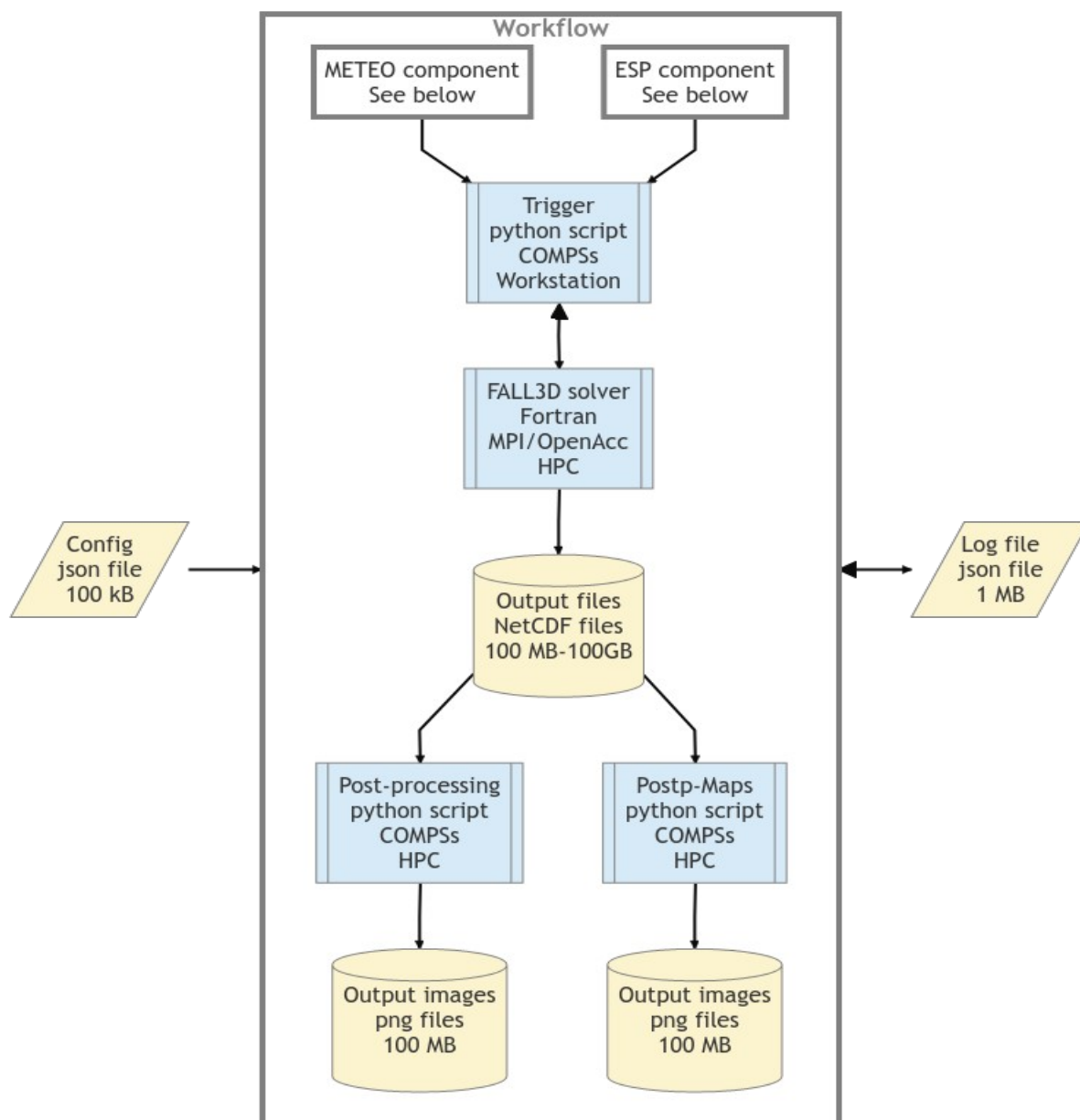


Figure 6: PD5 workflow overview

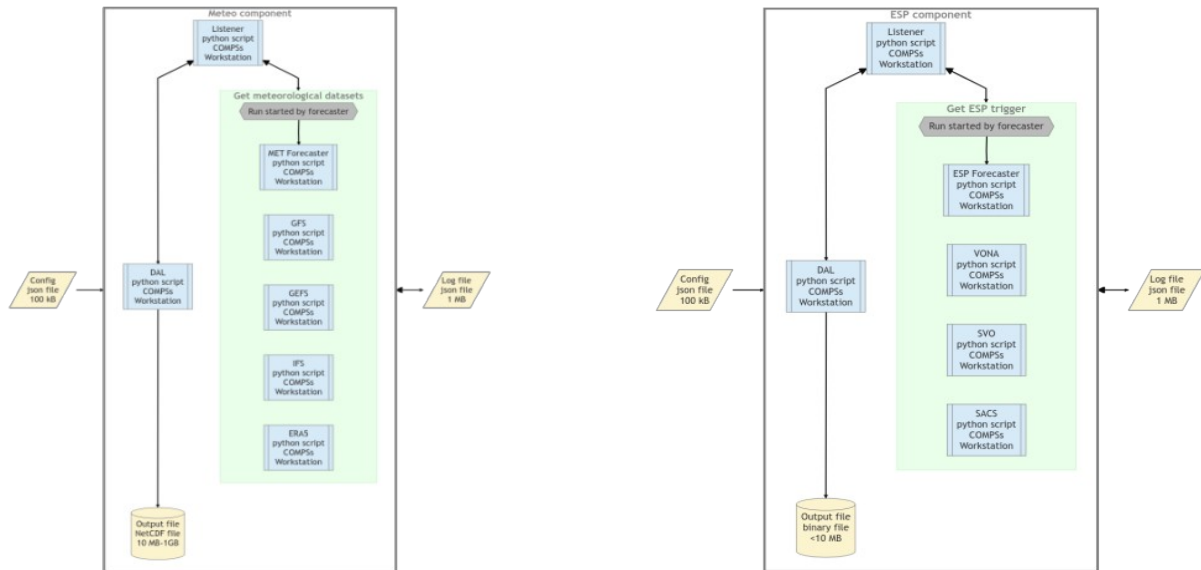


Figure 7: PD5 preprocessing subworkflows

From PD5 the simulation case SC5.1 “Towards the European tephra hazard map” is derived. The details of the related workflow around FALL3D are shown in Figure 8. One challenge here will be the management of multiple parallel simulation runs and their corresponding in- and output-data.

A sketch for the workflow of SC5.2 is shown in Figure 9.

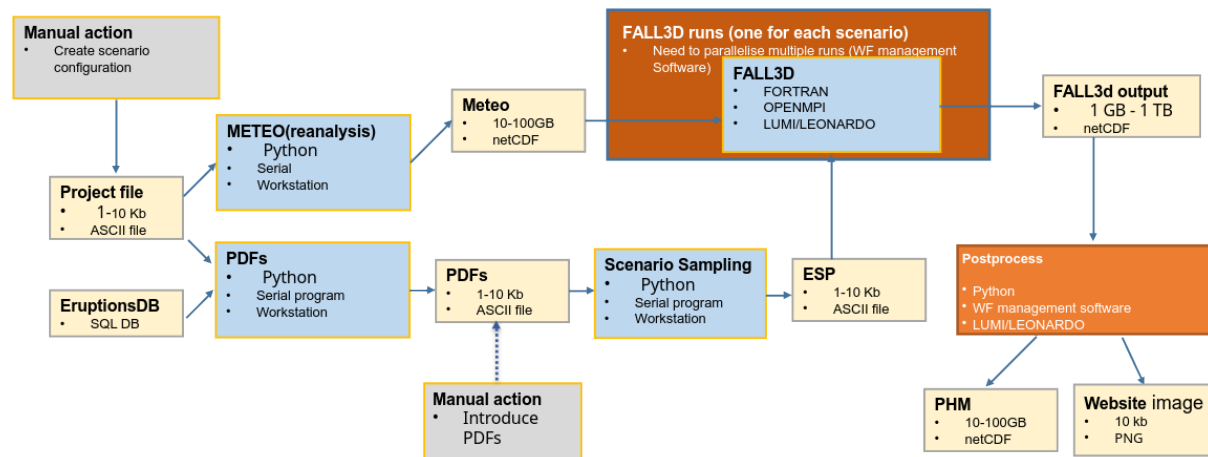


Figure 8: PD5 SC5.1 workflow with FALL3D

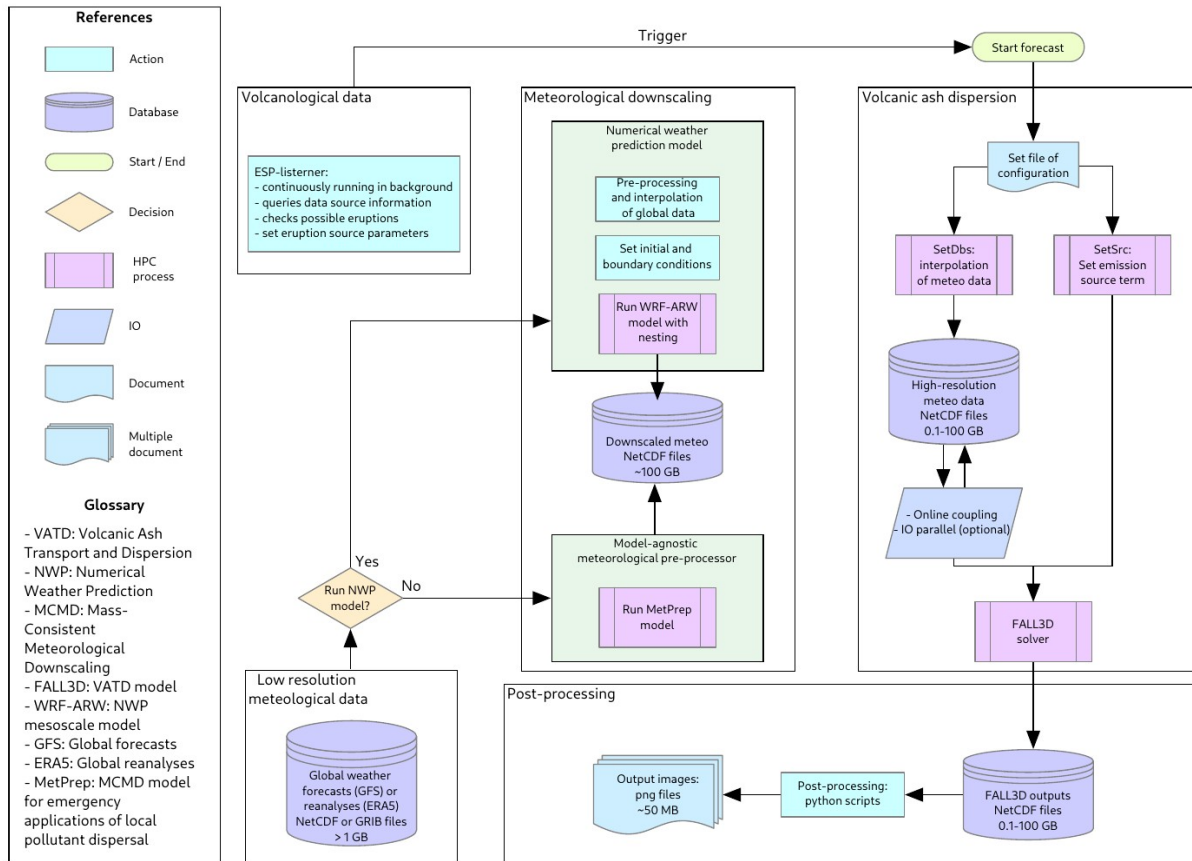


Figure 9: PD5 SC5.2 workflow sketch

PD6 (SC6.1, SC6.2)

PD6 “Multiphase 3D volcanic explosion modeling” and the derived simulation cases

- SC6.1 “Urgent high-resolution, 3D multiphase flow simulation of phreatic eruptions at Vulcano”
- SC6.2 “Long-term probabilistic hazard maps for phreatic eruptions at Vulcano island” are built around the OpenPDAC code.

They share a common workflow, which will either be used as an urgent computing workflow in case of SC6.1 and as a capacity workflow in the context of SC6.2.

Figure 10 shows the details of the workflow for SC6.1. The workflow is already implemented as bash and python scripts. As can be seen the workflow includes many pre- and postprocessing components. At the beginning the mesh generation requires several steps with intermediately sized input files. In the postprocessing a larger amount of output files from OpenPDAC have to be processed which are stored in several folders. The workflow was implemented within the reporting period and some initial

test runs with the LUMI and HAWK supercomputer systems were performed.

The SC6.1 workflow is planned to be adapted for the specifics of the capacity use case in SC6.2 later on.

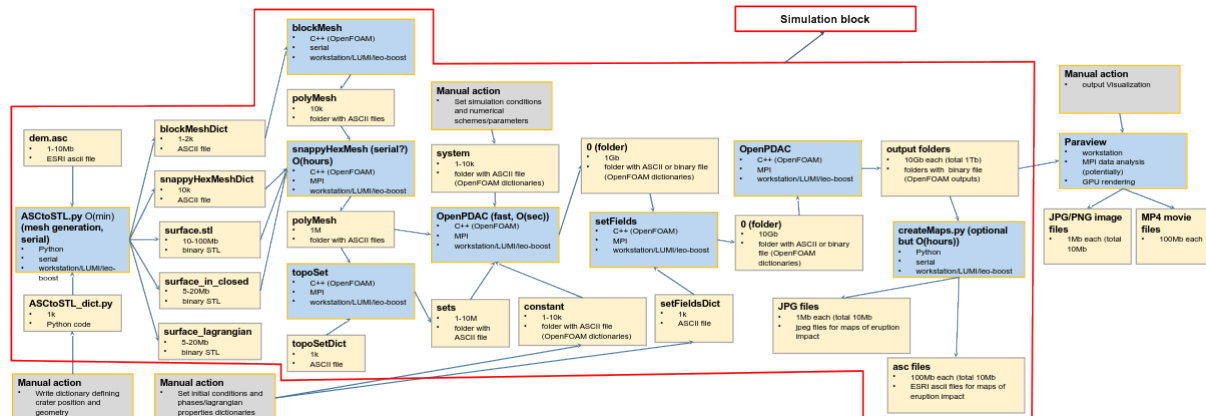


Figure 10: SC6.1 workflow details

PD9 (SC9.1, SC9.2)

The planned workflow for PD9 “Glacial outburst floods” and the derived simulation cases

- SC9.1 “Eastern Skaftá cauldron GLOF simulation, Iceland”
- SC9.2 “Calving glacier hazard in Greenland” will use pre- and postprocessing components around the Elmer/ICE code, which is used for the main simulation.

The studies around this workflow are intended to be several independent-single workflow-runs. Figure 11 shows the details of the related workflow.

As can be seen, the pre- and postprocessing components will have to handle several large input files – with an expected size of 10GB or more. Therefore, the challenges in this workflow will mainly arise from the large data set sizes.

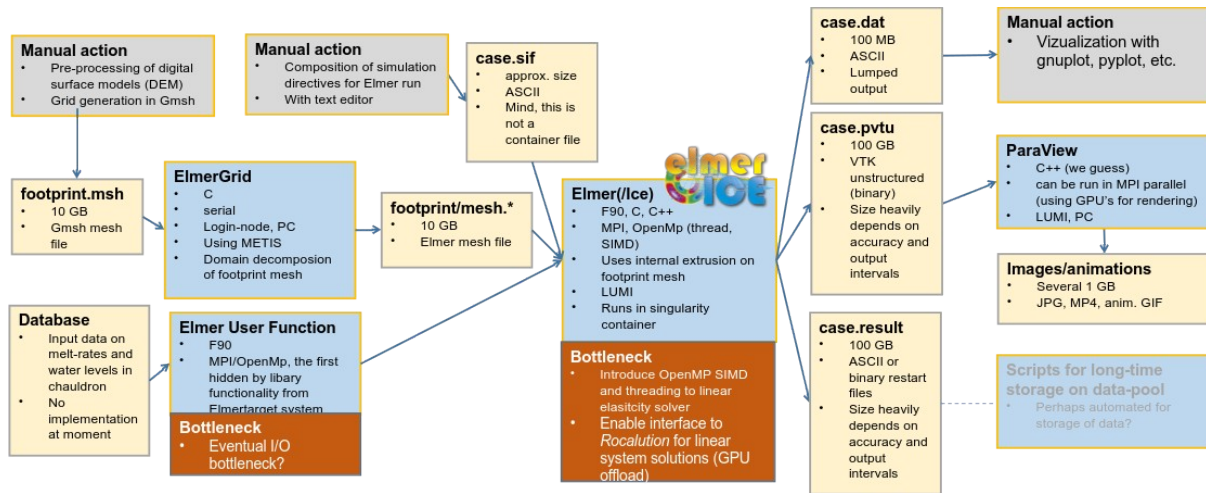


Figure 11: PD9 Workflow

Initial Workflow analysis and improvements

Initial workflow analysis and benchmarking for SC6.1

The first workflow that was set up in the reporting period was related to SC 6.1 (OpenPDAC). The details of the workflow are shown in Figure 10. The workflow is used for 3D multiphase flow simulations of phreatic volcanic eruptions and is authored by Tomaso Esposti Ongaro, Mattia de' Michieli Vitturi, and Federica Pardini.

We have benchmarked the initial workflow implementation with a scaled down dataset on the Hawk HPC system at the High Performance Computing Center Stuttgart. The workflow was run on a single AMD EPYC 7742 node with 128 cores. Performance data for all the workflow components were collected with respect to their execution times. Results are presented in Figure 12. Two components of the workflow stand out with respect to their execution time beside the main code (OpenPDAC): The ASCtoSTL.py script and the snappyHexMesh program that is part of the OpenFOAM package.

Beside the execution time, the high number of files produced by the OpenFOAM-based OpenPDAC application was identified as a potential issue for scaled up versions of this workflow.

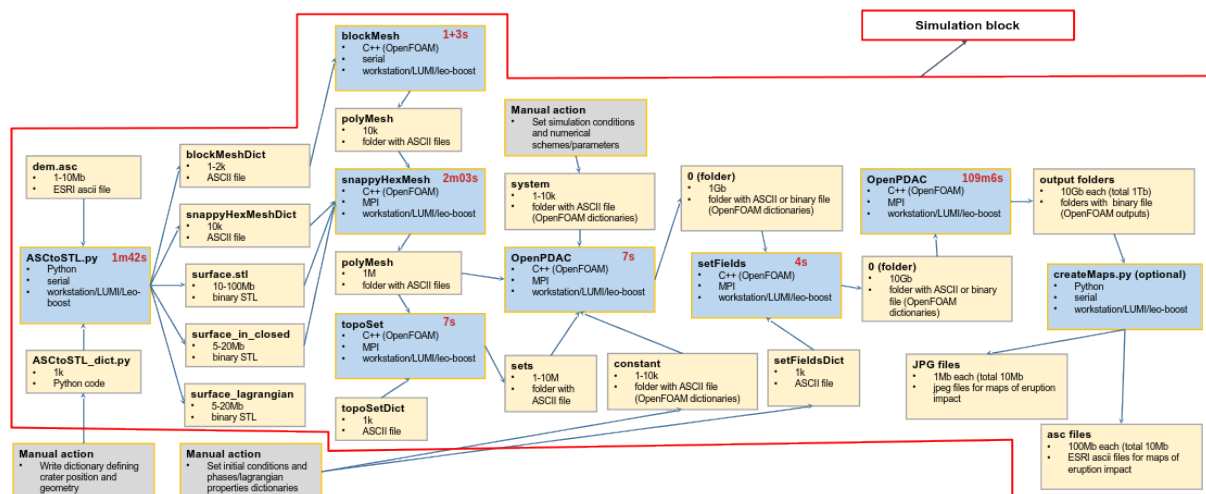


Figure 12: Results from benchmarking the SC6.1 workflow listing the time spent for each component in the crucial simulation block (blue boxes).

Workflow optimizations for SC6.1

Based on the analysis results for the SC6.1 workflow, the identified main bottleneck components in the preprocessing phase were analysed in more detail, and suggestions for improvements were made.

1) Optimization of ASCtoSTL.py

The Python script ASCtoSTL.py is a serial code that generates input files in the stl file format [1]. These stl files contain the mesh grids used in the simulations in later steps. We have performed an in-depth analysis of the script to identify bottlenecks in the code. Around 40% of the time is spent on the creation of Delaunay objects using the SciPy module. As this is a well-known scientific package – and is not part of the ChEESE-2P project – we do not work on this part. Instead, we focused on the other parts of the script which are developed within ChEESE-2P itself.

For these parts, we achieve an overall 26% speedup. The details of the optimizations leading to this improvement are described in the following.

Performance Gain using Numba

Several time-consuming operations that were repeatedly called in the main function have been identified and replaced by the functions `verticesCopy`, `vertices2Copy` and `verticesTBCopy` to improve the maintainability of the code.

In a next step, we addressed the performance of these functions by utilizing Numba [2], an open-source JIT compiler that can translate Python code into fast machine code. To optimize the code with Numba, the data types of some variables in these “vertices”-functions were replaced with Numba data types so that the functions can be compiled by the Numba compiler. These changes are the primary contributors to the 26% speedup.

Another optimization performed throughout the script was the rewriting of classical for loops using Python’s list comprehension.

Bugfixes

A bug related to deleting items from a list was identified during the optimization work. The original code could produce erroneous results in rare cases, particularly when a high number of items has to be removed, which could especially become an issue when scaling up the workflow. The issue has been fixed as part of the optimization work.

User Input Management with Spock Module

The user input format has been changed from a Python module to a YAML-format-based input file. For the handling of the YAML file input the Spock framework is now used [3]. A Spock class object is created by reading the YAML input file once. The input no longer needs to be imported multiple times in various locations of the script and checked repeatedly. The Spock class serves as a centralized location for the management of user inputs. It enforces default values, types of variables, their boundaries and relationships. This change helps the developers to enforce various input requirements and users to understand these requirements.

2) Optimization of snappyHexMesh

The snappyHexMesh program is a tool to generate mesh grids for the workflow. It is a part of the openFOAM package and already supports parallel execution with MPI. Although there are no plans to work on the snappyHexMesh code within ChEESE-2, we are interested in understanding the limitations of the program for its application in the ChEESE-2P workflows and in providing valuable feedback to the original developer community as well as related projects like exaFOAM.

Analysis of snappyHexMesh

We analysed the snappyHexMesh application with the SC6.1 input. Therefore, we created a trace based on the benchmark setup to understand its behaviour. To generate a trace of the snappyHexMesh execution we used the Extrae library from BSC. We executed the snappyHexMesh command using 32 processes instead of 128 in order to limit the size of the trace file. This approach is practical because issues found at a smaller scale are likely to be even worse at larger scale later on. The trace file was then analysed with Paraver.

Figure 13 shows the resulting trace. The overall program ran for about two minutes. The trace shows unsatisfactory communication efficiency. Especially in the first minute, the MPI_Probe calls performed by the main MPI process block all other MPI processes. In the second minute of the execution there is a significant number of calls of MPI_Allreduce and MPI_Alltotal.

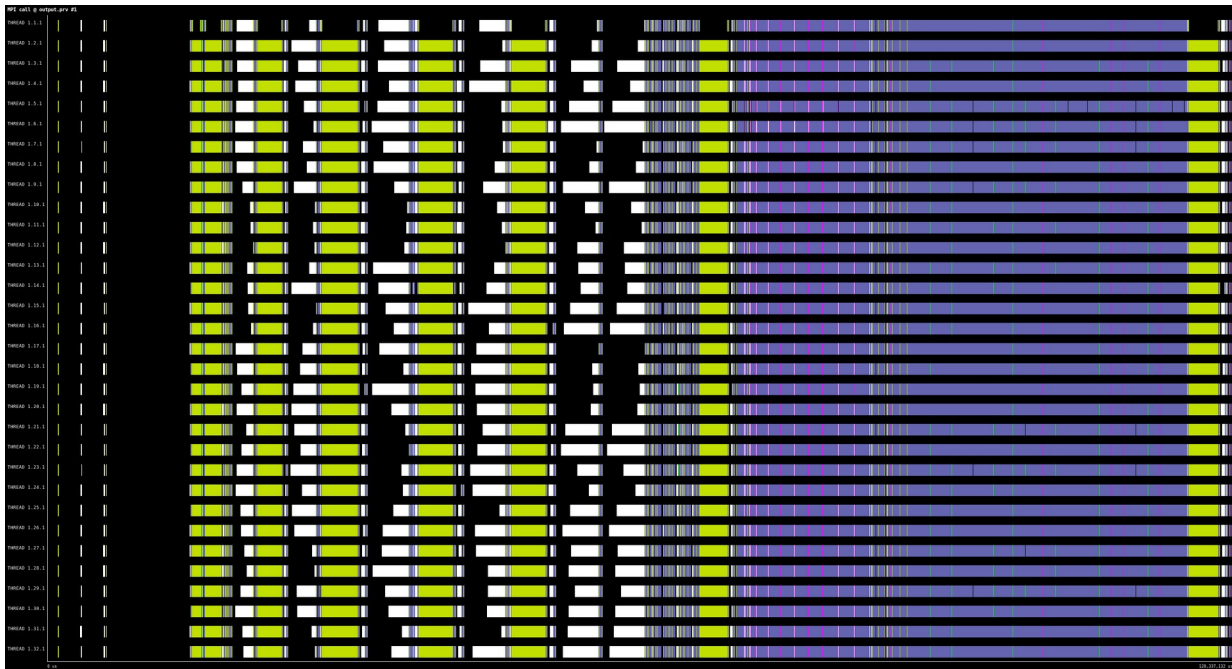


Figure 13: Trace of the snappyHexMesh program used in the SC6.1 workflow. The colors represent MPI communication, black computations.

Testing distributedTriSurfaceMesh instead of triSurfaceMesh

The initial phase of snappyHexMesh involves reading previously generated stl files. As we see from our analysis, in the current workflow setup, all MPI processes read the complete stl files. To improve the scalability, we suggest switching from triSurfaceMesh to distributedTriSurfaceMesh for processing these stl files. We tried this approach in the benchmark setup, however, as shown in Figure 3, it did not improve the performance – likely due to the relatively small size of the stl files used in the benchmark example.

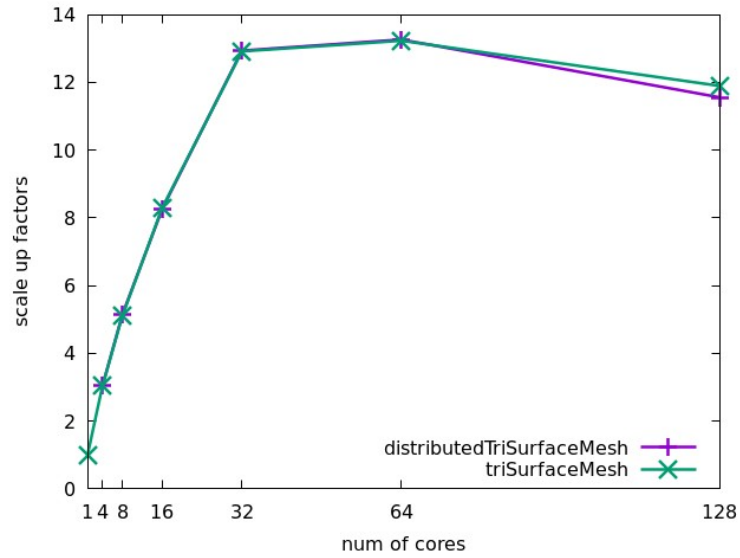


Figure 14: Performance comparison of regular and distributed stl input for parallel snappyHexMesh execution.

Summary and general findings around workflow activities in WP4

From the 9 PDs already 6 PDs detailed workflows and respective workflow descriptions are available. Among them, the PD6 and the related SC6.1 workflow was chosen for analysis and optimization activities as it could already provide a full setup and implementation. The analysis showed potential for improvements in the preprocessing part. Here components developed within ChEESE-2P could be optimized already.

In general, the components around the main codes developed within ChEESE-2P are mostly implemented using python. The scripts rely mainly on common python packages, e.g., NumPY or SciPy, which provide optimized implementations and may even support parallel execution.

Beside this, a wide variety of third-party tools, e.g., mesh generators, are used. Those are outside the scope of ChEESE-2P. However, basic analysis can be performed for them and may lead to recommendations for their usage, e.g., tuning parameters for I/O. If possible, we try to reach out to the respective developer communities, HPC system hosting sites, and related projects, providing feedback about our use cases.

For future work we will continue the process of detailing and analysing the workflows. Here we are also interested in identifying common issues among the workflows and potentially coming up with solutions and best practices applicable to multiple workflows, e.g., related to I/O and data file formats.

References

- [1] [https://en.wikipedia.org/wiki/STL_\(file_format\)](https://en.wikipedia.org/wiki/STL_(file_format))
- [2] <https://numba.pydata.org/>
- [3] <https://github.com/fidelity/spock>