D4.1 Requirements on the eFlows4HPC software stack from Pillar I and evaluation metrics

Version 1.2

Documentation Information

Contract Number	9555558
Project Website	www.eFlows4HPC.eu
Contractual Deadline	30.06.2021
Dissemination Level	PU
Nature	R
Author	Riccardo Rossi (CIMNE)
Contributors	Jorge Ejarque (BSC), Rosa Badia (BSC), Gianluigi Rozza (SISSA), Giovanni Stabile (SISSA), Guglielmo Scovazzi (CIMNE)
Reviewer	Mario Ricchiuto (INRIA)
Keywords	ROM, Digital Twin, Manufacturing

This project has received funding from the European High-Performance Computing Joint Undertaking (JU) under grant agreement No 955558. The JU receives support from the European Union's Horizon 2020 research and innovation programme and Spain, Germany, France, Italy, Poland, Switzerland, Norway.

Change Log

Version	Description Change
V0.1	Preliminary version, brainstorm format
V0.2	Reworking the deliverable based on proposed template, and after iteration with the BSC team
V1.0	Converged proposal
V1.1	Rewriting the deliverable to fit the presentation format employed in the other parallel deliverables
V1.2	Clarification and addition of introductory references on the basic theory + reformatting. Sparse correction as required in the review process

Table of Contents

1. Executive Summary

The overall goal of Pillar I is to provide an integrated workflow enabling the development of Reduced Order Models (ROM) from their inception to their deployment. The underlying objective is to enable the effective usage of large scale HPC hardware in speeding up the generation of reduced order models and to enable the solution of problems larger than it was possible prior to the implementation of the new capabilities.

Section 3 details the subdivision of the workflow in successive phases as well as the requirements of each of the steps and its relation with each other. Section 5 formalizes the requirements and constraints for the Pillar I developments. Section 6 identifies the target metrics to be employed in monitoring the progress of the Pillar developments during the project.

2. Introduction

The use of modern simulation techniques makes possible the simulation of complex engineering problems. This capability comes however at the cost of high computational requirements, often incompatible with deployment in compute constraint environments.

ROM models provide a possible solution to this limitation. The essential idea is that the accurate models, known as “Full Order Models” (FOM) can be used as a source of data to be harvested in search of common patterns. Once those patterns are identified similar problems can be solved at a much reduced computational cost.

The subject of current deliverable is to describe the phase needed for such learning, and to provide an estimation of the requirements needed by each of the steps. The deliverable attempts also to provide a clear definition of the expected data size and of the persistency requirements for the data being generated.

3. General description of workflow and its components

WP4 aims at providing a flexible workflow for the construction of reduced order models to be used in defining Digital Twins for manufacturing applications.

The essential idea is to start by defining a detailed model, named “Full Order Model” (FOM). The FOM is applied to the solution of multiple scenarios and the results are stored in a database. The results are harvested to find common patterns in the solution, defining a “reduced basis” which is used to construct Reduced Order Models (ROM) able to perform approximate predictions at a much reduced computational cost. A refinement of the model leading to the so called “hyper-reduced” model is then performed to further reduce the computational cost. Finally, the results of the hyper-reduced model are compared to the results of FOM, to decide if the procedure needs to be iterated or if the model can be deployed to the final consumer. This idea is shown in the Figure.

The reader is referred to [1,2,3,4] for an introductory discussion on the general algorithms at the base of projection based reduction approach which will be targeted by Pillar I.

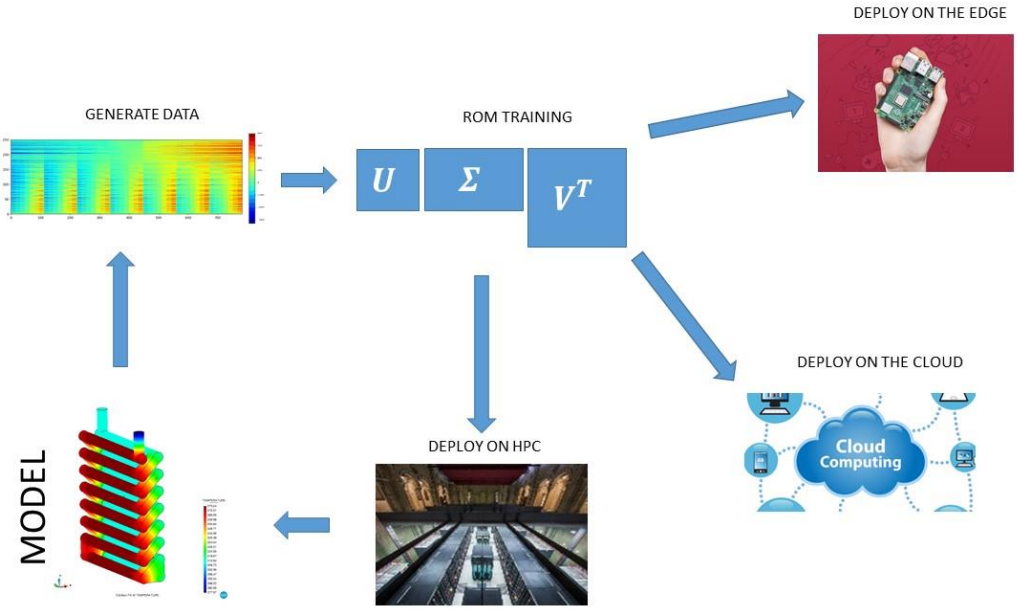

Figure 1. Model Reduction workflow

In order to design the workflow integration, we identify 5 clear substeps:

1. Generation of input data - Here a “training campaign” is defined and a set of simulations is performed in order to provide the data on which the training of the reduced order models will be based. Such data can be either used “on the flight” (one could recover the same data as needed by rerunning the testcase) or stored to disk to avoid the need for recomputing the data.
2. Data extraction phase - This phase analyses the output of the first phase in search of common patterns to be used in the construction of a ROM basis which will form the basis for the Projection-based ROM to be constructed. The most relevant kernel to be used in this step will be the “Singular Value Decomposition” although other alternatives based on the use of clustering techniques or on autoencoders may be evaluated. The output of such a step is thus either a “linear” reduced order basis (which would take the form of a matrix of doubles) or a cluster of solutions (multiple matrices) or potentially a trained neural network (storing the trained autoencoder). Such data allow constructing a ROM at anytime
3. Hyperreduction Step - This step builds on the ROM computed as output of the previous step and constructs new auxiliary data (by rerunning the training simulations basing on the ROM model). Such data will be employed to detect an optimal integration basis thus permitting the construction of “Hyperreduced models”. The possibility of generating and using on the flight those data will be evaluated with the aim of minimizing intermediated disk storage
4. Validation Step - the hyperreduced model is evaluated by assessing its performance against the data generated in the step 1. New, not previously seen simulations (either newly run at this stage or stored in step1 but not used) need to be used for this step. Depending on the results of this step the workflow may need to go back to step1 and improve the training set. Such a decision can either happen automatically or be user driven.

5. Deployment of the trained (and validated) Hyperreduced-ROM. Deployment may happen in the cloud to ease the usage outside of the compute environment or on small form factor devices

For each of those phases we detail the requirements as well as the IO required and the link to the other phases.

4. Innovative components of the Pillar I workflow

ROM training, also known as "offline phase" of model reduction, is very intensive from the computational point of view. At the same time, the memory requirements involved in the Data Analytics phases (as we shall see later, phases 2 and 3) may be very demanding depending on the amount of data to be analysed. This poses a practical challenge for the practical application of the technology to relevant test cases.

The goal of the project is to enable taking advantage of next generation HPC in solving this challenge. This will be accomplished by employing state-of-the-art workflow management systems in reducing the computational bottlenecks involved through the Pillar I workflow. The expected outcome is on one side to enlarge the problem size that can be practically tackled, and on the other hand to reduce the "time to solution" needed to arrive to the desired result.

5. Requirements & Constraints

The table that follows attempts to describe the computational requirements of the different step that compose the model reduction workflow, as described in Section 3. This includes both the technologies to be used in each phase, the input and output dependencies and an estimation of the expected runtime on the target hardware.

Table 1: Computational requirements of the different phases

Building Block	Name	Included actions	Input/Output data structure	HPDA/ML	Deployment	Time scale	Description
1	Generation of input data (Phase 1)	Execute the training campaign and make the data ready for successive actions	**Input** data set consists of a model definition completed by several setups describing the simulation scenario to be considered. The **output** consists logically of a "snapshot matrix" storing the desired results of the simulation. Each column of the matrix will represent the flattened output of a simulation step. This implies that for a transient	NO	HPC	hours	Input model must be prepared for execution and made available locally in the HPC machine.

			problem the columns will represent "frames" of the solution. Results from multiple scenarios will be logically concatenated within the same snapshots matrix. Such logical snapshot matrix may be divided in multiple archives to be merged during the second step.				
2	Data Extraction Phase (Phase 2)	Analyze training data to identify a reduced basis	**Input**: data sets consists of a subset of the output of Phase 1 (some data will be set a part for the validation phase) **Output**: ROM that consists of either a smaller matrix of doubles or a ML model (clustering/neural network)	Requires (distributed) SVD to analyse the snapshot matrix obtained from Phase 1. This is one of the most computationally intensive steps since such a matrix will be very large Eventually employ clustering algorithms/Neural Networks for the construction of a non-linear basis (will also require a distributed version)	HPC	Minutes/hours	The goal of this step is to identify the reduced basis to be used in the construction of the reduced order model and in the following steps
3	Hyperreduction step (Phase 3)	Determine optimized integration rule	**Input**: consists of the output of Phase 1 **Output**: Hyper-reduced Model is a list of indices and of weights which, together with the original solver, defines the ROM model.	Requires distributed SVD as a first step before an optimization loop. May optionally need Clustering algorithms/Neural Networks tools if those were employed in phase 2	HPC	Minutes/hours	The objective of this step is to identify an optimized integration rule which allows reducing the overall computational cost
	Validation Phase (Phase 4)	Determine if the hyperreduced model is "good enough"	**Input**: data sets consists of a subset of the output of Phase 1 (The part not used in phase, 2 and 3) as well of the outputs of phase 2 and 3 **Output** is "go"/"no go" (may eventually suggest to go back to phase 1 or phase 2)	Might make use of Clustering algorithms/Neural Networks (only inference). Essentially we need to compare the output of the ROM model to data that we didn't see before, to see how accurate our predictions are. Model Evaluation based on accuracy	HPC	Minutes	Assess if the model can deliver on the target accuracy when targeting data it was not trained against. An unsuccessful output mandates to go back to Phase 2 (or even 1 if more data are needed)
	Deployement phase	Distribute the executable and the trained model so that it can be executed	Input: outputs of Phase2 and Phase3. Model used in Phase1. Parameters to be used in the simulation	Requires the simulation code as a basis. May need ANN inference module in case a nonlinear basis is selected in Phase1	HPC/cloud	Seconds/minutes	Distribution of the model so that it can be run as needed. May run on the same

		where needed					hardware as where the Phases 1-4 are performed or may be run on a different target (for example in the cloud)

5.1. Programming languages

The Pillar I workflow will require the use of development tools as described in the following table.

Table 2: Development tools needed for the development

Programming language	Versions/distributions	Role in the Pillar I workflow
sh	bash, csh/tcsh, ksh	Scripting at all stages of workflow execution
Python	Conda or pip installation with possibility to install user software	Scripting, data processing and visualization, glue language between different components.
C++	Can be gcc or commercial versions	Compilation of model code and analysis tools

5.2. Specialized software dependencies

A number of libraries should be available in the system in order to facilitate the deployment of the Pillar I software stack. The list is contained in the table below.

Table 3: Main libraries needed for the deployment of the Pillar software stack

Library	Versions/distributions	Role in the Pillar I workflow
MPI	Commercial versions or OpenMPI.	Parallelization. The workflow should allow model compilation with different versions of the MPI libraries, e.g. IntelMPI, OpenMPI, if the ESM has this option.
OpenMP	Commercial versions or free.	Parallelization. Support for OpenMP parallelisation is not as common in ESM components as MPI, but still can be found quite often.
BLAS/LAPACK	Commercial or free versions.	linear algebra libraries.
HDF5	High performance library to write binary data to disk.	HDF5 is used as option to write intermediate data to disk so that it can be easily retrieved, parsed and analysed when needed

5.3. Infrastructural requirements

The workflow should provide information to workflow components on what infrastructure is available on the machine, how to use it, and check if there are enough resources that can be allocated to execute a particular configuration of the workflow. The most important resources are:

- **Availability of computational resources**. The number of cores used in Pillar I depends on the size of the task and can range from just a few cores for test configurations to thousands of cores, when very high spatial resolution, or a very wide training set is to be used.
- **File system**. The Pillar I model simulations produces a large amount of intermediate (and temporary) data. Such data is to be written on disk (or such behaviour emulated) so that locality and persistency of data becomes very important

5.4. Workflow deployment /execution requirements

The solvers and reduction models to be developed in Pillar I are expected to be portable, it should hence be possible to compile or retrieve the required model on any target platform. Specific requirements to be fulfilled by the workflow are:

- Workflow programming and management have to allow re-start the ROM computation according to validation results.
- Workflow management is also required through the phases to coordinate the execution of the different computing units

5.5. Data Requirements

The complete workflow involves a number of different steps and the storage of various intermediate data.

The simulation starts with an initial mesh and configuration files, with a memory occupation of the order of “few” GB. Training data are then generated in Phase1 and consumed in the following table. The size of such data can be estimated to be in the order of 100GB, corresponding to a typical matrix size of the order of 10^6 by 5000 (specific cases may however exceed such sizes). In phase 3 an even larger data set is constructed, however the constructed data is consumed immediately and not persisted for the following phases.

The storage of ML models, to be used when exploring nonlinear reduction, will also require the storage of structured data, a task for which native data formats will be employed whenever possible. The table below resumes the Production and consumption patterns to be used in the workflow.

Table 4: Data Requirements for Pillar 1

Data Item	Data Flow	Persistency	Data Type	Consumption Pattern	Notes
Simulations Definitions	Source	persistent	json	Read in Phase 1,3,4	This will depend on the actual solver being used
Mesh	Source	persistent	Solver specific format/ HDF5 (if possible)	Read in Phase 1,3,4	This will depend on the actual solver being used
Snapshot Matrix	Intermediate data	persistent	Matrix format	1-N (Created in 1 and consumed in 2,3 and 4)	The matrix format to be employed will depend on the implementation being employed for the distributed storage
Reduced Model	Intermediate data	persistent	Matrix format	1-1 (Created in 2 and used in 3)	
Intermediate matrices for Hyper-reduced Modes	Intermediate data	Not persistent	Matrix format	Created and consumed in phase 3	
Hyper-reduced Model	Workflows Result	persistent	Native ML format o r HDF5based	Validated in Phase 4 and deployed in Phase 5	

5.6. Summary of requirements for WP4

The following table summarizes the essential features required for the successful development of the Pillar Software stack, providing also an estimation of their relative importance.

Table 5: Summary of features required for Pillar 1

ID	Name	Description	Priority
P1-1	Distributed SVD	Requires an optimized distributed execution of the Singular Value Decomposition (SVD) algorithm to analyze large scale matrices which can exceed the memory of different computing nodes of a cluster.	Must
P1-2	Storing of hyper-reduced model	Requires storing and transferring the meshes and the trained ML model needed to reconstruct the hyper-reduced model, together with the solver executable needed to run it.	Must
P1-3	ANN model	Require Artificial Neural Networks (ANN) (probably convolutional) to train autoencoders. This may provide an attractive option to improve the reduction ratio of the reduced model. Here both training data and the output to be used in the inference step need to be saved.	May
P1-4	Clustering model	Clustering algorithms as an option to improve the reduction ratio. Here both training data and the output to be used in the inference step need to be saved.	Should
P1-5	Persistent storage	Requires persistent storage for data to be consumed between the steps	May
P1-6	Restart	Workflow programming and management have to allow re-start the Reduced-Order-Model (ROM) computation according to validation results.	Should
P1-7	Workflow orchestration	Workflow management is also required through the phases to coordinate the execution of the different computing steps.	Must
P1-8	ML inference	Simulation code requires access to the ML trained model.	May
P1-9	Hyper Reduced Model Deployment	Once the hyper reduced model is computed it may be deployed in a computing infrastructure,(such as a Cloud) to be accessible and reusable by final users. This requires to deployment the model together with software and data needed to run a complete hyper-reduced model from scratch.	May

6. Metrics

The purpose of this section is to identify the target metrics to be used in assessing the improvements of the software stack as well as of the underlying theory.

Workflow developments will be based on python scripts based on PyCOMPSs as well as on developments on the solver side will. Minimal, automated, test cases should be provided to verify the implementation within the workflow. Improvements in the runtimes for selected testcase should be monitored during the project. The selected metrics is thus the time needed to train a Reduced Order Model starting from a Full Order Model and a predefined training campaign.

A second possible metrics is the “size” of problem that can be reduced. To this end we shall consider that the model reduction phases are both limited by the amount of available memory and by the computational cost involved. The developments shall progressively relief such practical limitations by allowing the workflow to effectively leverage large distributed systems through the entire workflow.

A third metrics, will be to keep track of the “reduction ratio” between the number of degrees of freedom (dofs) in the Full Order and Reduced Order model when clustering or autoencoders are used as alternative to linear projection algorithms.

Data transfer should be limited to input testcases and training output. In particular the hyper-reduced model will contain only the minimal information needed to restart the model at any place.

A possible metrics will be also to monitor the distributed SVD speedup as well as the largest achievable problem size.

A summary of the proposed metrics is given in the table below.

Table 6: Collection of Target Metrics to be used for Pillar I

Acronym	Name	Definition (how it is measured)
ROMtime	Time to compute a Reduced Order Model starting from a Full Order Model and a predefined training campaign (WP4)	Measure progress in performance with respect to a baseline once fixed the training conditions
ROMsize	"size" of the problem that can be reduced	Keep track of the largest problem to which the workflow can be successfully applied. This can include either the reduction of larger models or the inclusion of more training scenarios
RedR	Reduction ratio	Given a target level of accuracy, number of degrees of freedom in the ROM vs number of degrees of freedom in the FOM
SVDS	Speedup of Distributed SVD	Speedup of SVD extraction for a given problemsize
SVDL	Largest possible SVD	Largest problem which could be analyzed by SVD

7. Conclusion

The workflow is divided in a number of interdependent phases. The main requirements are sketched and the relation between the workflow phases is identified. The data flow is also identified and tentatively quantified.

8. Acronyms and Abbreviations

- HPC – High Performance Computing
- KPI – Key Performance Indicator
- WP – Work Package
- ML – Machine Learning
- DA – Data Analytics
- SVD – Singular Value Decomposition
- ROM – Reduced Order Model
- FOM – Full Order Model

9. References

[1] M. Hinze and S. Volkwein, Proper orthogonal decomposition surrogate models for nonlinear dynamical systems: Error estimates and suboptimal control, in Dimension Reduction of Large-Scale Systems, Lect. Notes Comput. Sci. Eng. 45, Springer, Berlin, 2005, pp. 261–306.

[2] K. Kunisch and S. Volkwein, Control of the Burgers equation by a reduced-order approach using proper orthogonal decomposition, J. Optim. Theory Appl., 102 (1999), pp. 345–371.

[3] W. R. Graham, J. Peraire, and K. Y. Tang, Optimal control of vortex shedding using low- order models. Part I—Open-loop model development, Internat. J. Numer. Methods Engrg., 44 (1999), pp. 945–972.

[4] S. S. Ravindran, A reduced-order approach for optimal control of fluids using proper orthogonal decomposition, Internat. J. Numer. Methods Fluids, 34 (2000), pp. 425–448.