elPaSo-Core Manual

Institute for Acoustics, Technische Universität Braunschweig

2023

Contents

1 Introduction

1.1 About elPaSo

The research code "Elementary Parallel Solver (elPaSo)" is the in-house vibroacoustic simulation tool constantly developed over 25 years at TU Braunschweig, presently extended and maintained by the Institute for Acoustics (InA), TU Braunschweig.

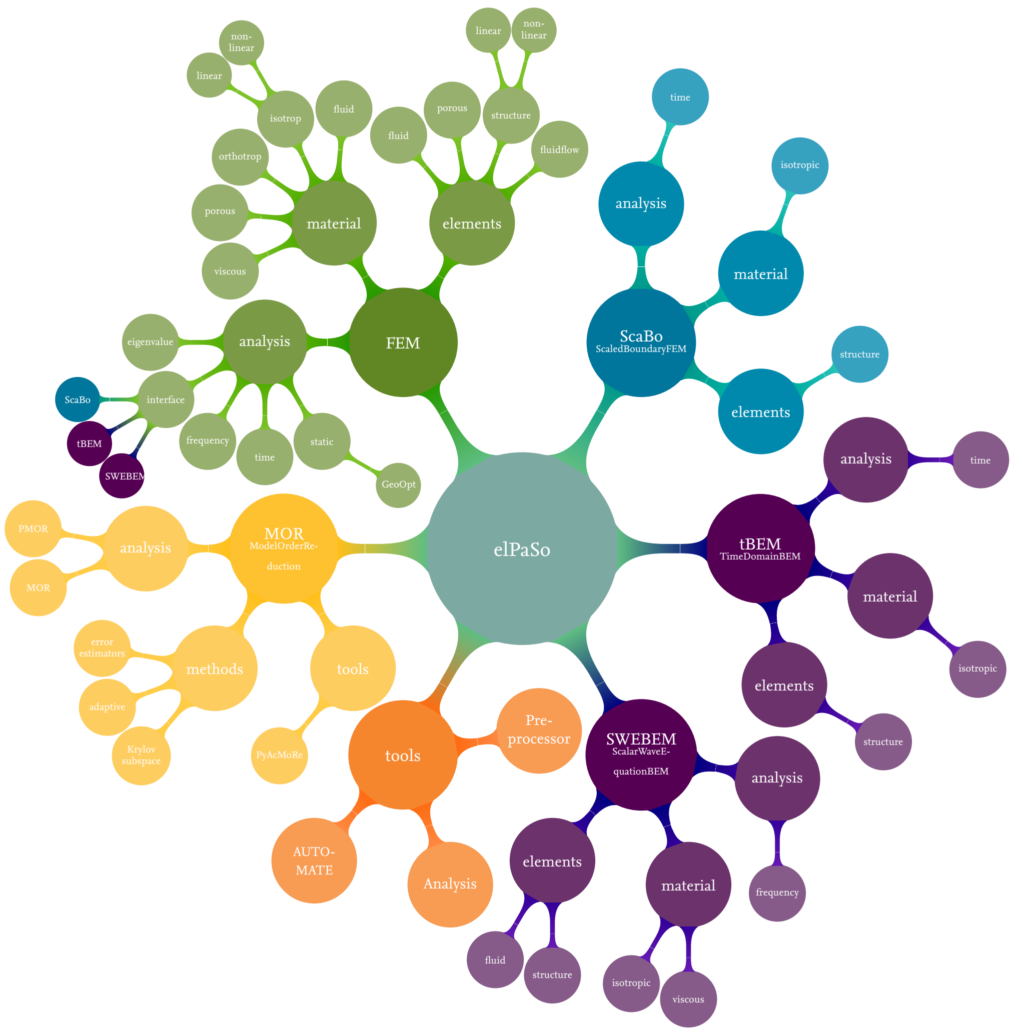

Figure 1.1: elPaSo bubble diagram

The tool is used extensively for research and teaching for many years. The whole elPaSo project includes the core software written in C++ and other assisting in-house tools written in python. elPaSo offers a wide range of features (see Features and Capability) and facilitates efficient computations in HPC clusters to support parallel computing of large-scale high-fidelity vibroacoustic models. Moreover, the project is built on a flexible and modular framework to support new research features and potential collaborations.

1.2 elPaSo Modules

elPaSo implementations are provided to users and developers in two different modules: (1) `elPaSo Core (eCore)` module and (2) `elPaSo Research (eResearch)` module; described as follows.

1.2.1 elPaSo Core Module

The eCore module comprises stable functionalities of elPaSo and is available to the public as open-source. The eCore project can be used as a standalone vibroacoustic simulation tool or/and as a FEM library which can be incorporated into other projects. The main components included in eCore module is discussed in Chapter 2.

1.2.2 elPaSo Research Module

The eResearch module extends eCore module with advanced features and capabilities. The project is currently closed-source. If you are interested in using or benefitting eResearch functionalities, you are warmly welcome to contact us. A sneak peek into functionalities offered by both the eResearch module is illustrated in Chapter 2.

1.3 Features and Capabilities

1.3.1 Vibroacoustic solver for FEM, BEM and SBFEM

The capabilities of the tool include acoustic and structural analysis using the popular numerical methods of FEM, BEM and SBFEM supporting various complex material and element types. The solver is suitable for acoustic and structural analysis for static analysis, modal analysis, time domain analysis and frequency domain analysis. Various applied fields with elPaSo are building acoustics, geometric optimizations, wave propagation, soil-structure interactions and fluid-structure interaction.

1.3.2 Uncertainty quantification

elPaSo enables non-intrusive parametric uncertainty quantification (UQ). This capability allows the use of elPaSo as a black-box to obtain the system response associated with each realisation of random vector and UQ is performed as an extension of the deterministic analysis of the model. Also, the possibility to parallelize the elPaSo code in this approach provides an added advantage. These characteristics make the elPaSo solver very attractive for parametric uncertainty quantification in complex models and industrial applications.

1.3.3 Model order reduction

elPaSo provides methods to perform model order reduction in both frequency and parameter domain. As a result, computations can be performed faster in a reduced space without compromising on accuracy. The methods are constantly developed and improved for handling large-scale vibroacoustics models.

1.3.4 Code sustainability

elPaSo ensures code sustainability developed as a part of the DFG project SURESOFT. This includes continuous integration, software testing, containerization and bug reporting. Software-testing concepts in the development workflow incorporating unit-, integration- and performance testing are also adapted within the code. Best practices to facilitate detailed code documentation, both technical and usage, is underway.

1.3.5 Efficient deployment

In order take advantage of the available computational resources, parallel computing has been enabled with MPI and OMP parallelization. elPaSo uses state-of-the-art mathematical routines provided from Intel Math Kernel Library for basic LAPACK and BLAS operations, and PETSc, SLEPc, ARPACK for advanced scientific computations. A wide of direct, iterative and eigen solvers assists the solution process for various analysis types which includes MUMPS, PARDISO and GMRES. The tool is successfully deployed in the TU Braunschweig Phoenix cluster and is used since many years for large-scale computations.

1.3.6 Supporting tools and interfaces

The supporting tools of elPaSo includes sub-projects supporting pre and post processing of simulation data, visualization and advanced computational wrappings. The tools offered are formulated in the following table.

Table 1.1: Overview of supporting tools and interfaces

elPaSo Pre-processor Tool	Provides modelling features of material modelling, loading types, boundary conditions, conforming and non-conforming coupling interfaces, and many more.
elPaSo Post-processor/Analysis Tool	Provides a range of post-processing routines to compute the interested acoustic quantity and routines for visualizing the obtained results. The tool associates with methods to visualize measurement data and routines to perform data fitting.
elPaSo Py-AcMoRe **(Python module for faster training and evaluation of vibroacoustics systems using model order reduction)**	Provides model order reduction functionalities for faster computations in frequency and parametric domain.

In addition to the above-mentioned in-house tools, elPaSo is open to interface with external softwares. Currently, the code interfaces to commercial softwares of Coreform Cubit and ABAQUS.

1.4 Key publications

[1] Christopher Blech, Christina K. Appel, Roland Ewert, Jan W. Delfs, and Sabine C. Langer. “Numerical prediction of passenger cabin noise due to jet noise by an ultra–high–bypass ratio engine”. In: *Journal of Sound and Vibration* 464 (2020), p. 114960. ISSN: 0022-460X. DOI: https://doi.org/10.1016/j.jsv.2019.114960. URL: https://www.sciencedirect.com/science/article/pii/S0022460X1930522X.

[2] Marco Schauer, Jose E Roman, Enrique S Quintana-Orti, and Sabine Langer. "Parallel computation of 3-D soil-structure interaction in time domain with a coupled FEM/SBFEM approach". In: *Journal of Scientific Computing* 52.2 (2012), pp. 446–467.

[3] Harikrishnan K. Sreekumar, Christopher Blech, and Sabine C. Langer. "Large-scale vibroacoustic simulations using parallel direct solvers for high-performance clusters". In: *Proceedings of the DAGA Conference*. Vienna, Austria, 2021.

[4] Harikrishnan K. Sreekumar, Christopher Blech, and Sabine C. Langer. "Sustainable development and deployment of an acoustic FEM research software". In: *Proceedings of the DAGA Conference (abstract submitted)*. Hamburg, Germany, 2023.

2 Components in elPaSo Core

In this chapter, the main components constituiting the eCore project is described.

2.1 Finite element method

2.1.1 Analysis

Table 2.1: Analysis types in elPaSo

Functionality	Identifier	elPaSo Core	elPaSo Research
Basic frequency domain steady state analysis	frequency-basic	✓	
Advanced frequency domain steady state analysis	frequency		✓
Time domain analysis	time		✓
Static analysis	static		✓
Geometrical optimization	geoopt		✓
Eigenvalue analysis	eigen		✓
Model order reduction	mor-offline		✓

2.1.2 Elements

Identifier	elPaSo Core	elPaSo Research
Structure | Beam elements		
BeamBernoulli	✓	
BeamTimoshenko	✓	
BeamBernoulli10	✓	
BeamTimoshenko10	✓	
BeamBernoulli12	✓	
BeamTimoshenko12	✓	
BeamBernoulli10nl		✓
BeamTimoshenko10nl		✓
Structure | Brick elements		
Brick8	✓	
Brick20		✓
Brick27		✓
Structure | Plate elements		
Kirch4	✓	
DSG3		✓
DSG4	✓	
DSG9		✓

<table>
<tr><td>DSG9pre</td><td></td><td>✓</td></tr>
<tr><td colspan="3">Structure | Disc elements</td></tr>
<tr><td>Disc3</td><td></td><td>✓</td></tr>
<tr><td>Disc4</td><td>✓</td><td></td></tr>
<tr><td>Disc9</td><td></td><td>✓</td></tr>
<tr><td>Disc9s</td><td></td><td>✓</td></tr>
<tr><td>DiscDr4</td><td></td><td>✓</td></tr>
<tr><td colspan="3">Structure | Shell elements</td></tr>
<tr><td>PlShell3</td><td></td><td>✓</td></tr>
<tr><td>PlShell4</td><td></td><td>✓</td></tr>
<tr><td>PlShell9</td><td></td><td>✓</td></tr>
<tr><td>PlShell9pre</td><td></td><td>✓</td></tr>
<tr><td>PlShellDr4</td><td></td><td>✓</td></tr>
<tr><td colspan="3">Structure | Tetrahedron elements</td></tr>
<tr><td>Tetra4</td><td></td><td>✓</td></tr>
<tr><td>Tetra10</td><td></td><td>✓</td></tr>
<tr><td>Tetra4L</td><td></td><td>✓</td></tr>
<tr><td>Tetra10L</td><td></td><td>✓</td></tr>
<tr><td>Tetra16</td><td></td><td>✓</td></tr>
<tr><td>Tetra16L</td><td></td><td>✓</td></tr>
<tr><td colspan="3">Structure | Spring elements</td></tr>
<tr><td>Spring</td><td>✓</td><td></td></tr>
<tr><td>Springz</td><td>✓</td><td></td></tr>
<tr><td>SpringBC</td><td>✓</td><td></td></tr>
<tr><td>SpringBCx</td><td>✓</td><td></td></tr>
<tr><td>SpringBCy</td><td>✓</td><td></td></tr>
<tr><td>SpringBCz</td><td>✓</td><td></td></tr>
<tr><td>SpringBCrx</td><td>✓</td><td></td></tr>
<tr><td>SpringBCry</td><td>✓</td><td></td></tr>
<tr><td>SpringBCrz</td><td>✓</td><td></td></tr>
<tr><td colspan="3">Structure | Point mass</td></tr>
<tr><td>Pointmass</td><td>✓</td><td></td></tr>
<tr><td colspan="3">Structure | Cable elements</td></tr>
<tr><td>Cable2D</td><td></td><td>✓</td></tr>
<tr><td>Cable3D</td><td></td><td>✓</td></tr>
<tr><td colspan="3">Structure | Porous elements</td></tr>
<tr><td>PoroPlateQ2P1</td><td></td><td>✓</td></tr>
<tr><td>PoroPlateQ2P2</td><td></td><td>✓</td></tr>
<tr><td>PoroPlateQ2P3</td><td></td><td>✓</td></tr>
<tr><td>PoroPlateQ3P1</td><td></td><td>✓</td></tr>
<tr><td>PoroPlateQ3P3</td><td></td><td>✓</td></tr>
<tr><td>DiscPQ2P1</td><td></td><td>✓</td></tr>
<tr><td>ShellQ2P1</td><td></td><td>✓</td></tr>
<tr><td>PoroPlateKienzler4</td><td></td><td>✓</td></tr>
<tr><td>PoroDiscKienzler4</td><td></td><td>✓</td></tr>
<tr><td>PoroShellKienzler4</td><td></td><td>✓</td></tr>
<tr><td>Poro3dUP8</td><td></td><td>✓</td></tr>
</table>

Poro3dUP27		✓
Poro3dUU8		✓
Fluid | Linear elements		
Fluid4		✓
Fluid8	✓	
Fluid27		✓
Fluid2d4		✓
Fluid2d9		✓
Fluid | Flow elements		
FF4		✓
FF9		✓

Table 2.2: Element types in elPaSo

2.1.3 Materials

Elpaso is capable of implementing different material models for acoustical simulations. For example: orthotropic, isotropic, fluid, spring, etc. Each material has its own behavior and characteristics, each of them being described within the syntax against the respective property values. See Table 2.3 for an overview of material types offered in eCore.

Functionality	Identifier	elPaSo Core	elPaSo Research
Structure | Isotropic materials			
Linear elastic isotropic	STR_LIN_ELA_ISO_DIR	✓	
Linear viscoelastic isotropic	STR_LIN_VIS_ISO_DIR	✓	
Non-linear elastic isotropic	STR_NL_ELA_ISO_DIR		✓
Linear viscoelastic isotropic implementing Cremer/Heckl constrained layer damping (CLD) model	STR_LIN_VIS_ISO_CLDCH		✓
Linear viscoelastic isotropic implementing Ross/Kerwin/Ungar (RKU) constrained layer damping (CLD) model	STR_LIN_VIS_ISO_CLDRKU		✓
Linear viscoelastic isotropic implementing RKU CLD and laminate theory	STR_LIN_VIS_ISO_CLDRKU_-LAMWA		✓
Structure | Orthotropic materials			
Linear elastic orthotropic	STR_LIN_ELA_ORT_DIR		✓
Linear viscoelastic orthotropic	STR_LIN_VIS_ORT_DIR		✓
Linear viscoelastic orthotropic laminate	STR_LIN_VIS_ORT_LAM_NOPRE		✓
Linear viscoelastic orthotropic laminate prestressed	STR_LIN_VIS_ORT_LAM		✓
Structure | Spring			
Linear spring orthotropic	STR_LIN_SPR_ORT_DIR	✓	
Structure | Mass			
Point mass	STR_LIN_POINTMASS	✓	
Fluid | Acoustic			

Linear acoustic fluid undamped	AF_LIN_UAF_ISO_DIR	✓	
Linear acoustic fluid damped	AF_LIN_EQF_ISO_DIR		✓
Linear acoustic fluid lossy	AF_LIN_VIS_LF_DIR		✓
Linear acoustic fluid to treat porous media	AF_LIN_VIS_ISO_POROUS		✓
Linear acoustic cloaking fluid	AF_LIN_CLK_AISO_DIR		✓
Structure | Poroelastic materials			
Linear poroelastic	STR_LIN_ELA_POR_DIR		✓
Linear poroelastic for plate elements based on Biot's theory	STR_LIN_ELA_POR_PLATE		✓
Linear poroelastic for foams	STR_LIN_ELA_POR_FOAM		✓
Linear poroelastic 3D elements based on Biot's theory	STR_LIN_ELA_POR_3D		✓
Linear poroelastic 3D elements with loss	STR_LIN_VIS_POR_3D		✓
Linear poroelastic 3D elements with loss and compression modulus	STR_LIN_VIS_POR_3DCOMP		✓
Linear poroelastic 3D elements trans	STR_LIN_ELA_POR_3DTRANS		✓
Structure | Elasticplatic materials			
Non-linear elasticplastic cap model	STR_NL_ELPLA_ISO_CM		✓
Non-linear elasticplastic cap model with outer adjustment	STR_NL_ELPLA_ISO_CMO		✓
Non-linear elasticplastic Drucker Prager parameters	STR_NL_ELPLA_ISO_DP		✓
Non-linear elasticplastic Drucker Prager parameters ideal plastic	STR_NL_ELPLA_ISO_DPIP		✓
Non-linear elasticplastic Drucker Prager parameters with outer adjustment	STR_NL_ELPLA_ISO_DPO		✓
Non-linear elasticplastic Drucker Prager parameters with outer adjustment ideal plastic	STR_NL_ELPLA_ISO_DPOIP		✓

Table 2.3: Material types in elPaSo

2.1.4 Loads

Table 2.4: Load types in elPaSo

Functionality	Identifier	elPaSo Core	elPaSo Research
	Nodal loads		
Force/moment on a node	point_force	✓	
Force/moment on a node - time dependent	structuretime		✓
Sound source at a fluid node	fluid	✓	
	Element loads		
Frequency dependent element loads	plane_wave		✓
Frequency dependent element loads TBL	turbulent_boundary_layer		✓
Normal velocity vn or flux q acting normal to surface of fluid element	v_n		✓

2.1.5 Boundary conditions

Table 2.5: Boundary condition types in elPaSo

Functionality	Identifier	elPaSo Core	elPaSo Research
Structural nodal constraints	structural	✓	
Acoustic nodal constraints	acoustic	✓	
Acoustic nodal constraints, oblique incidence	fluidoblique		✓

2.1.6 Coupling interfaces

Table 2.6: Coupling interface types in elPaSo

Identifier	elPaSo Core	elPaSo Research
Conforming interface elements		
InterfaceBeam	✓	
InterfaceFFPoro3dUP		✓
InterfaceFFShell		✓
InterfaceKirchhoff	✓	
InterfaceMindlin	✓	
InterfaceMindlinEquiporo		✓
InterfaceMindlinPoro3dUP		✓
InterfacePlaneShellPoro2dUP		✓
MultiPointConstraint		✓
Non-conforming interface elements		
NCInterfaceMindlin		✓
NCInterfaceMindlinEquiporo		✓

2.2 Input file definition

Input to elPaSo is currently limited to tailor-made HDF5 file format produced by the elPaSo Preprocessing tool. Support for other common file formats is in development. However, you can develop a new parser to support your input file by adapting from the parser interface class in cFemParserInterface.h. Contact us if you require support.

2.3 Output file formats

Various output file formats are support for outputting simulation results. Currently eCore supports HDF5, VTK and STP.

3 Using elPaSo Core

As already mentioned in the introduction, elPaSo Core is can be used in two ways according to your purpose:

1. **eCore Executable**: The executable can be used to solve vibroacoustic problems. Guideline to compile an elpaso executable is outlined in Section 3.1.
2. **eCore Library**: By generating eCore library, you can link eCore as a third party library in your project and use the FEM functionalies from eCore.

3.1 elPaSo Core as a vibroacoustic solver

elPaSo can be with GNU compilers and Intel compilers (recommended). This section mainly focus on elPaSo preparation using a basic GNU compiler.

3.1.1 Executable types

1. **elpaso**: Executable compiled with real data-types and hence computations without complex domain (mainly used for time-domain computations).
2. **elpasoC**: Executable compiled with complex data type.
3. **elpasoT**: Executable running all the unit-tests for eCore using GoogleTest.

3.1.2 Building executable

1. Download and run the docker image containing the basic environment required for building elPaSo. This contains the necessary third part libraries and compiler.
   - `sudo docker pull <link to the image will follow soon after intial release>`
   - `sudo docker run -it <image name>`
   - Now you are inside the container with all necessary elPaSo dependencies.
2. Checkout the current version of the elPaSo Core project
   - `git clone https://git.rz.tu-bs.de/akustik/elPaSo-Core`
   - `cd elPaSo-Core`
3. Configure a cmake configuration file for your system
   - `make cmake-gen`
   - This creates a config.cmake file with your hostname in the "cmake" folder. The file contains the default configuration. You need not change anything here unless required.
4. Build elPaSo Core executable

a) With GNU compiler
   - `mkdir project`
   - `make elpaso-gnu`
   - `make elpaso-build`
   - When sucessfull, executables (elpaso, elpasoC and elpasoT) are generated in ./bin folder.
b) With Intel compiler
   - Install intel-parallel-studio and configure the ./cmake/$HOSTNAME.config.cmake with the respective directory where the installation is done with INTEL_DIR and the respective version number to INTELMPI_VERSION.
   - `mkdir project`
   - `make elpaso-intel`
   - `make elpaso-build`
   - Executables (elpaso, elpasoC and elpasoT) are generated in ./bin folder.

3.1.3 Running executable

elPaSo can be started by direct use of the compiled files elpaso and elpasoC. The input file is needed in the folder where the calculation is started. A simple serial run with standard settings can be made by the command:

```
elpasoC -c -inp myInputFile.hdf5
```

A check of the input file is started by the command:

```
elpasoC -check -inp myInputFile.hdf5
```

To run problems with the executable elpaso having real-only petsc datatype, follow the same commands with the respective executable name:

```
elpaso -c -inp myInputFile.hdf5
```

Unit-tests can be performed by running the test executable:

```
elpasoT
```

3.1.4 Running executable in parallel using MPI

In order to execute elpaso in parallel may some additional MPI commands have to be used. These commands are depending on the computer used, e.g.,

```
mpirun -np 4 elpasoC -c -inp myInputFile.hdf5
```

in order to run the program using 4 processes. `mpirun` is the instance supplied from an MPI vendor. For GNU compiler build, we use OpenMPI and for INTEL compiler build, we use Intel MPI.

3.2 elPaSo Core as a FEM library

3.2.1 Building library

1. Download and run the docker image containing the basic environment required for building elPaSo. This contains the necessary third part libraries and compiler.
   - sudo docker pull <link to the image will follow soon after intial release>
   - sudo docker run -it <image name>
   - Now you are inside the container with all necessary elPaSo dependencies.
2. Checkout the current version of the elPaSo Core project
   - `git clone https://git.rz.tu-bs.de/akustik/elPaSo-Core`
   - `cd elPaSo-Core`
3. Configure a cmake configuration file for your system
   - `make cmake-gen`
   - This creates a config.cmake file with your hostname in the "cmake" folder. The file contains the default configuration.
   - To generate the eCore library, active the library generation option in ./cmake/$HOSTNAME.config.cmake by modifying: `OPTION(GEN_DLIB "elPaSo DYNAMIC LIB" ON)`
4. Build elPaSo Core library
   a) With GNU compiler
      - `mkdir project`
      - `make elpaso-gnu`
      - `make elpasolib-build`
      - When sucessfull, elpasoCore-gnu folder contains the necessary entities for linking eCore.

   b) With Intel compiler
      - Install intel-parallel-studio and configure the ./cmake/$HOSTNAME.config.cmake with the respective directory where the installation is done with INTEL_DIR and the respective version number to INTELMPI_VERSION.
      - `mkdir project`
      - `make elpaso-intel`
      - `make elpasolib-build`
      - When sucessfull, elpasoCore-intel folder contains the necessary entities for linking eCore.

3.2.2 Linking library

Follow the steps below, to link eCore library with CMake. In your FindELPASOCORE.cmake:

1. Include the path ./elpasoCore-<COMPILER>/include to your cmake project for link to header files
2. Link the eCore dynamic library:

- For real-valued build, link `./elpasoCore-<COMPILER>/lib/libelpasoCore-intel-cxx-o.so`
- For complex-valued build, link `./elpasoCore-<COMPILER>/lib/libelpasoCore-intel-cxx-complex.so`