WWW.IJRPR.COM

International Journal of Research Publication and Reviews

Journal homepage: www.ijrpr.com ISSN 2582-7421

RiSat: A MATLAB-based Tool for Preliminary Lazy Riser Analysis

Elakpa, Ada Augustine [a], Ahaotu Jonadab Obum [b]

[a,b] Department of Marine Engineering, Rivers State University, Port Harcourt, Rivers State, Nigeria
University of Port Harcourt, Rivers State, Nigeria

A B S T R A C T

Offshore risers are critical structural components in offshore oil and gas exploration and production, operating under complex and harsh marine environments. Accurate and efficient analysis of riser structural behavior, encompassing static loads, dynamic responses, modal characteristics, and corrosion effects, is paramount for ensuring operational safety and preventing catastrophic failures. This paper introduces RiSat, an integrated, user-friendly, and computationally efficient software tool developed in MATLAB for comprehensive riser analysis. RiSat features a graphical user interface (GUI) for intuitive parameter input and simulation control, implements advanced numerical methods for static, dynamic, modal, and corrosion analyses, and provides versatile visualization and automated report generation capabilities. This paper details the architecture, functionalities, and implementation of RiSat, demonstrating its application through a case study and highlighting its potential to enhance riser analysis workflows in offshore engineering practice and research

Keywords: Offshore Risers, Riser Analysis, MATLAB Tool, Structural Analysis, Deepwater Risers, Hydrodynamic Analysis, Dynamic Analysis, Static Structural

1. Introduction

Offshore risers are slender, pipe-like structures that serve as critical conduits for transporting hydrocarbons, control fluids, and injection fluids between subsea production systems on the seabed and floating production, storage, and offloading (FPSO) vessels or other surface facilities [1, 2]. Operating in deep-water and ultra-deep-water environments, these structures are continuously exposed to a complex interplay of static and dynamic loading conditions, including self-weight, internal and external pressures, hydrodynamic forces from waves and currents, and vessel-induced motions [3, 4]. Furthermore, risers are subjected to harsh marine environments conducive to corrosion, necessitating thorough assessment of material degradation over their operational lifespan [5].

The structural integrity of offshore risers is of paramount concern due to the severe consequences of failure, which can range from environmental pollution and economic losses to risks to human life [6, 7]. Traditional riser analysis relies heavily on complex numerical simulations, often employing commercial Finite Element Analysis (FEA) software packages like ANSYS, ABAQUS, or SACS [8]. While these commercial tools offer robust capabilities, they can be characterized by steep learning curves, high computational demands, significant software licensing costs, and a degree of inaccessibility for engineers and researchers without specialized training [9, 10]. Furthermore, the iterative nature of riser design and analysis often demands rapid prototyping and parametric studies, which can be cumbersome and time-consuming within complex commercial FEA environments [11].

To address these challenges, there is a compelling need for streamlined, accessible, and computationally efficient software tools that can facilitate comprehensive riser analysis, particularly in the preliminary design and research phases [12, 13]. This paper presents RiSat (Riser Analysis Tool), a novel MATLAB-based software tool specifically developed to provide an integrated platform for riser analysis, encompassing static, dynamic, modal, and corrosion analysis within a user-friendly environment. Leveraging MATLAB's powerful numerical computing capabilities, intuitive GUI development environment, and extensive visualization libraries, RiSat aims to democratize advanced riser analysis techniques, making them more readily available to a broader engineering community and accelerating design and research iterations.

RiSat is designed with a modular architecture, encompassing distinct modules for data input and project management, static and dynamic analysis calculations, corrosion assessment, and comprehensive visualization and reporting. The tool implements established engineering principles and numerical methods, including Euler-Bernoulli beam theory for static and modal analysis, the Newmark-beta method for dynamic time integration, and semi-empirical models for corrosion rate prediction. The paper will detail the tool's architecture, the underlying methodologies implemented in each module, and demonstrate its application through a representative case study. The paper will conclude by discussing the benefits, limitations, and potential future.

Extensions of Risat as a valuable asset in offshore engineering and research and for academic purposes.

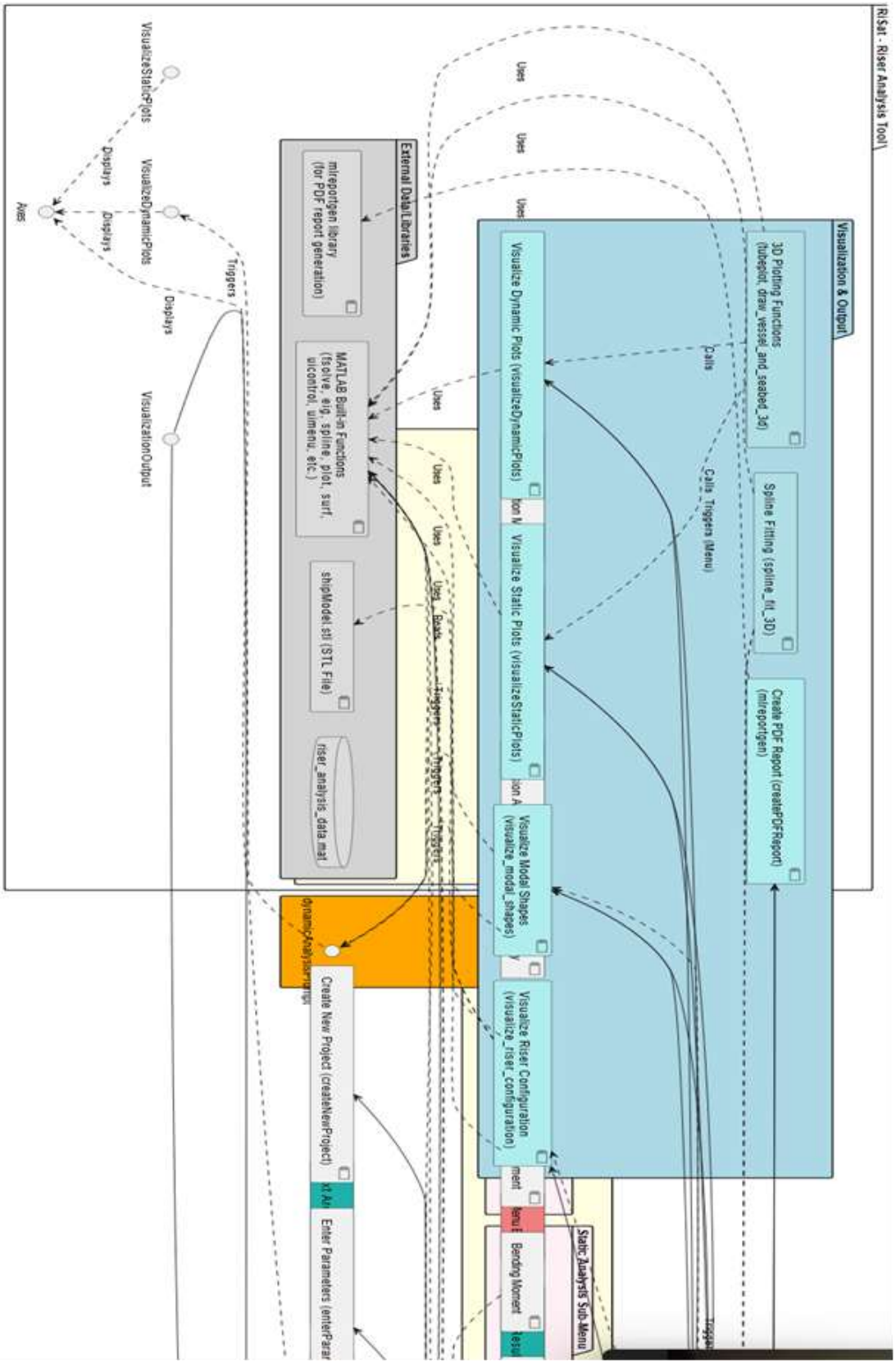
The structure of this paper is organized as follows: Section 2 elucidates the architecture of the RiSat tool, detailing its functional modules, workflow scenarios, and data management strategy. Section 3 provides an in-depth description of the functionalities implemented within each module. Section 4 presents a simulation case study to showcase RiSat's application and output. Section 5 offers a discussion of the tool's capabilities, limitations, and Potential future development.

2.0 Risat Tool Architecture

2.1 Functional Overview

RiSat is architected as a modular software tool to ensure flexibility, maintainability, and ease of expansion. As depicted in Fig. 1, the tool is functionally decomposed into five primary packages: (1) GUI - Main Interface, (2) Data Input & Project Management, (3) Analysis Modules, (4) Visualization & Output, and (5) External Data/Libraries. This modular design allows for independent development, testing, and enhancement of individual functionalities while maintaining a cohesive and integrated tool [14].The main GUI interface is shown in the appendix.

As shown in Fig. 1, the Main Interface serves as the primary point of user interaction, facilitating control over the tool's functionalities through a menu-driven system, parameter input fields, and visualization panels for result display. The Data Input and Project Management package is responsible for managing project data, including riser specifications, environmental parameters, and analysis settings. This module ensures data persistence and project organization through file-based storage using MATLAB's .mat format. The core analytical computations are performed within the Analysis Modules package. This package encompasses static analysis, dynamic analysis, modal analysis, and corrosion assessment, each implementing specific numerical methods and engineering formulations. The Visualization and Output package provides a suite of functions for graphical representation of analysis results and automated report generation, enabling effective interpretation and communication of simulation outcomes. The External Data/Libraries package highlights the tool's reliance on external data files (e.g., STL models for vessel geometry) and MATLAB's built-in functions and libraries, ensuring portability and leveraging the extensive MATLAB ecosystem [15, 16]. This modular architecture promotes a clear separation of concerns, facilitating future development, maintenance, and integration of new analysis capabilities.


RiSet - Riser Analysis Tool
Visualization & Output
3D Plotting Functions (fullpoint, draw_vessel_and_seabed_3d)
Spline Fitting (spline_fit_3D)
Create PDF Report (createPDFReport) (mlreportgen)
Visualize Dynamic Plots (visualizeDynamicPlots)
Visualize Static Plots (visualizeStaticPlots)
Visualize Modal Shapes (visualize_modal_shapes)
Visualize Riser Configuration (visualize_riser_configuration)
Bending Moment
Static_Analysis Sub-Menu
External Data/Libraries
mlreportgen library (for PDF report generation)
MATLAB Built-in Functions (fsolve, eig, spline, plot, surf, uicontrol, uimenu, etc.)
shipModel.stl (STL File)
riser_analysis_data.mat
Create New Project (createNewProject)
Enter Parameters (enterParar
VisualizeStaticPlots
VisualizeDynamicPlots
VisualizationOutput
Uses
Calls
Triggers
Displays
Axes

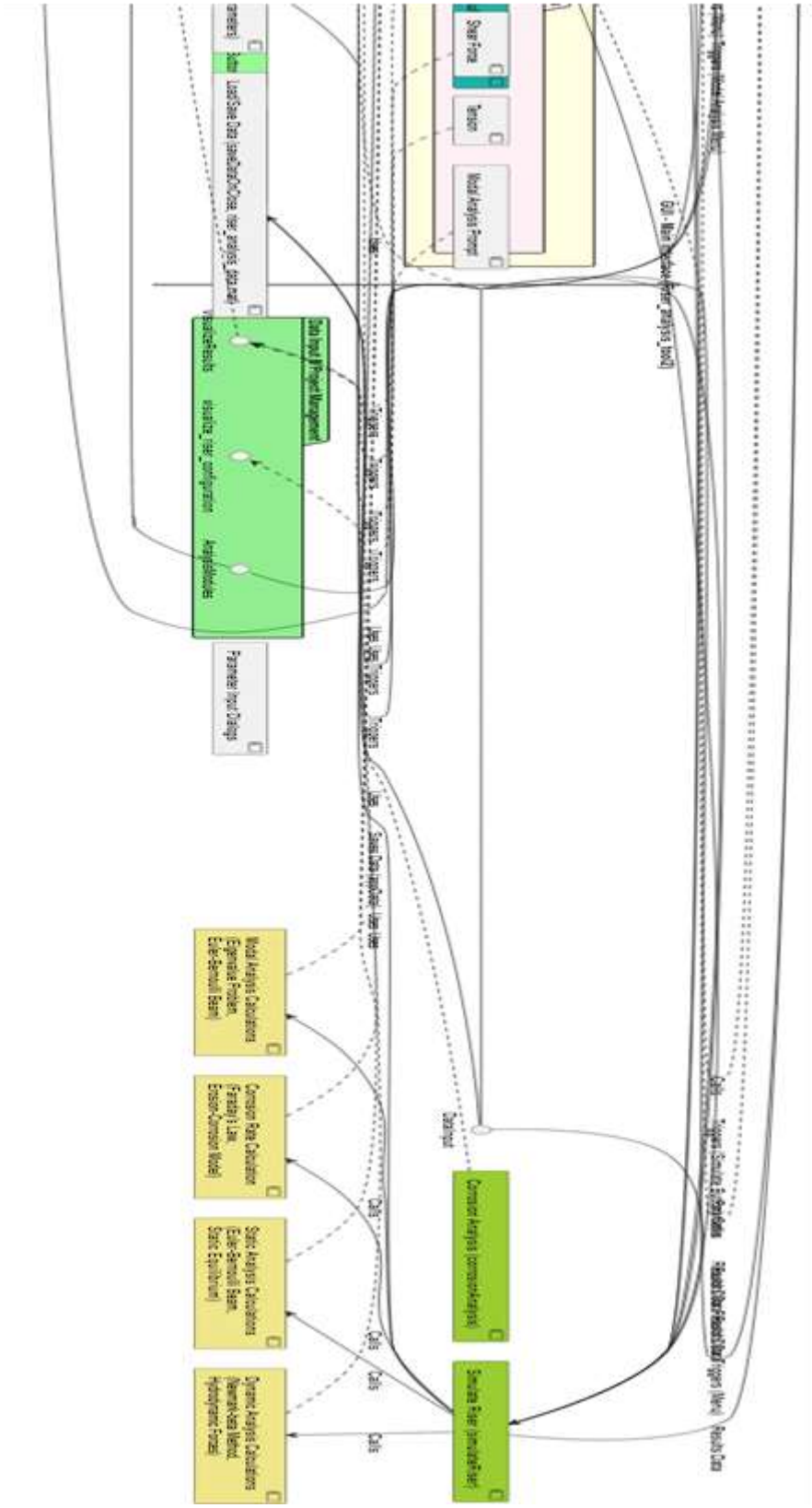


Fig. 1: Functional Diagram of RiSat Tool

2.2 Workflow Scenarios:

To further elucidate the operational workflow of RiSat, sequence diagrams are presented for representative analysis scenarios.

2.2.1 Static Tension Analysis Workflow:

The sequence diagram for static tension analysis is illustrated in Figure 2, detailing the steps involved in visualizing tension distribution along the riser.

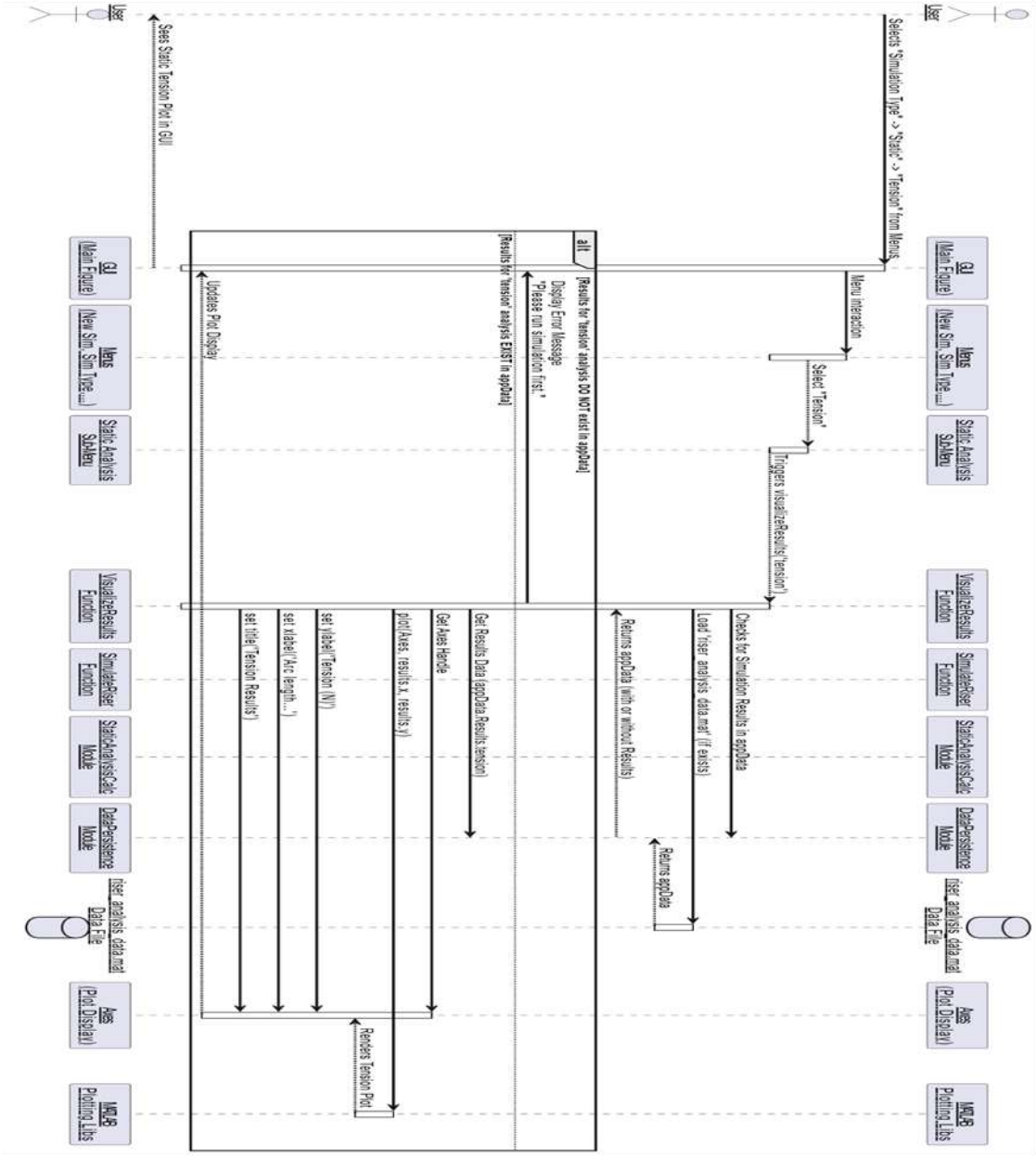


Fig. 2: Sequence Diagram for Static Tension Analysis

The workflow, as depicted in Figure 2, commences with user interaction via the GUI to select the "Static Tension Analysis" option from the menu. This action triggers the visualizeResults('tension') function. The visualizeResults function initiates a data retrieval process, first querying the DataPersistence module to check for pre-existing simulation results within the appData structure. If necessary, the DataPersistence module loads the appData from the riser_analysis_data.mat file. Upon successful retrieval of appData (which may or may not contain the required tension analysis results), the visualizeResults function checks for the presence of tension analysis results. If the results are not available (indicating that a simulation has not yet been performed), an informative error message is displayed in the GUI, prompting the user to execute a simulation. Conversely, if the tension analysis results are present, the visualizeResults function proceeds to extract the relevant data, obtain a handle to the designated axes object in the GUI, and call MATLAB's plotting functions (plot) to generate the tension distribution plot against the riser arc length. Appropriate axis labels and a plot title are then set to enhance the clarity and interpretability of the visualization. Finally, the updated plot is displayed within the GUI, providing the user with a visual representation of the static tension analysis results [17].

2.2.2 Dynamic Displacement Analysis Workflow:

Figure 3 presents the sequence diagram for a dynamic displacement analysis, showcasing the workflow from user initiation to result visualization.

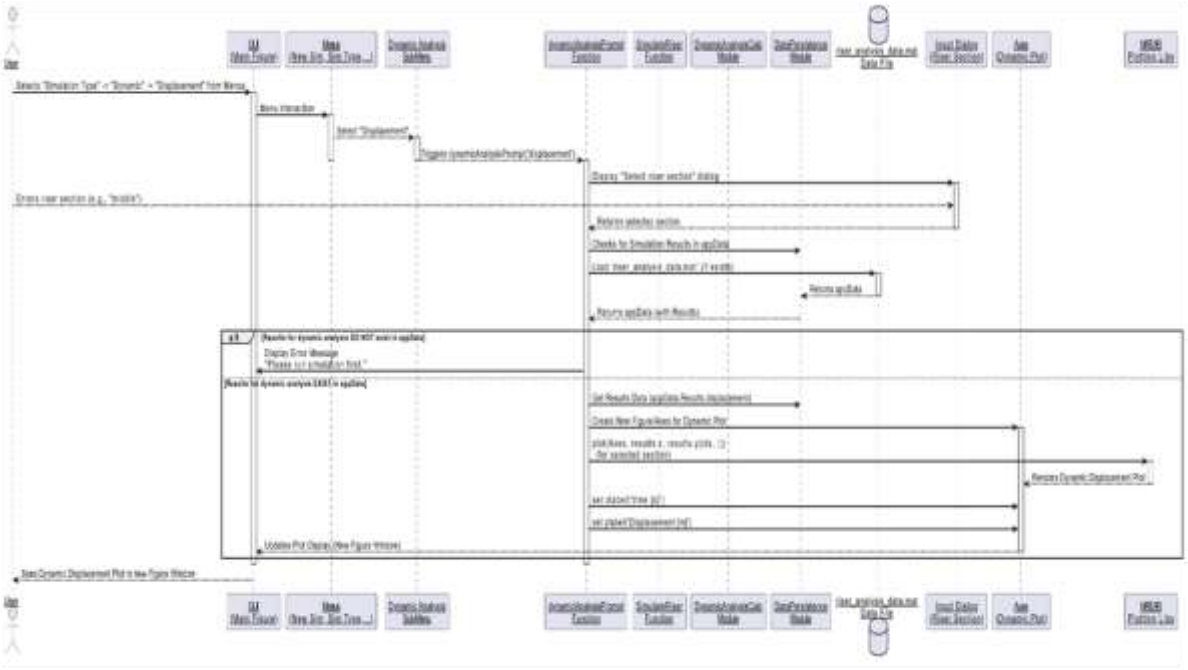


Fig. 3: Sequence Diagram for Dynamic Displacement Analysis

As depicted in Figure 3, initiating a dynamic displacement analysis begins with the user selecting “Dynamic” and then “Displacement” from the GUI menus. This action triggers the dynamicAnalysisPrompt('displacement') function. A key feature of dynamic analysis visualization is the option to examine results at specific riser sections. Therefore, the dynamicAnalysisPrompt function first presents a user input dialog (InputDialog) prompting the user to select a riser section of interest (e.g., top, middle, bottom). Upon receiving the user’s section selection, dynamicAnalysisPrompt proceeds to retrieve simulation data by interacting with the DataPersistence module and loading appData from the .mat file if required. Similar to the static analysis workflow, it verifies the presence of dynamic displacement results within appData. If the results are not found, an error message is displayed. If the dynamic displacement results are available, dynamicAnalysisPrompt extracts the data corresponding to the selected riser section. Crucially, for dynamic plots, a *new* figure window and axes object are created specifically for dynamic visualization, ensuring clarity and separation from static plots. MATLAB plotting functions are then employed to generate the time-series plot of displacement for the chosen section, with appropriate axis labels (time [s] and displacement [m]) set for interpretability. The newly generated dynamic displacement plot is displayed in the dedicated figure window, providing the user with a temporal view of the riser's dynamic response at the chosen location [18].

2.3 Data Management and Persistence:

RiSat implements a robust data management strategy to ensure project data integrity and persistence across sessions. The data flow diagram in Figure 4 illustrates the data management architecture of RiSat.

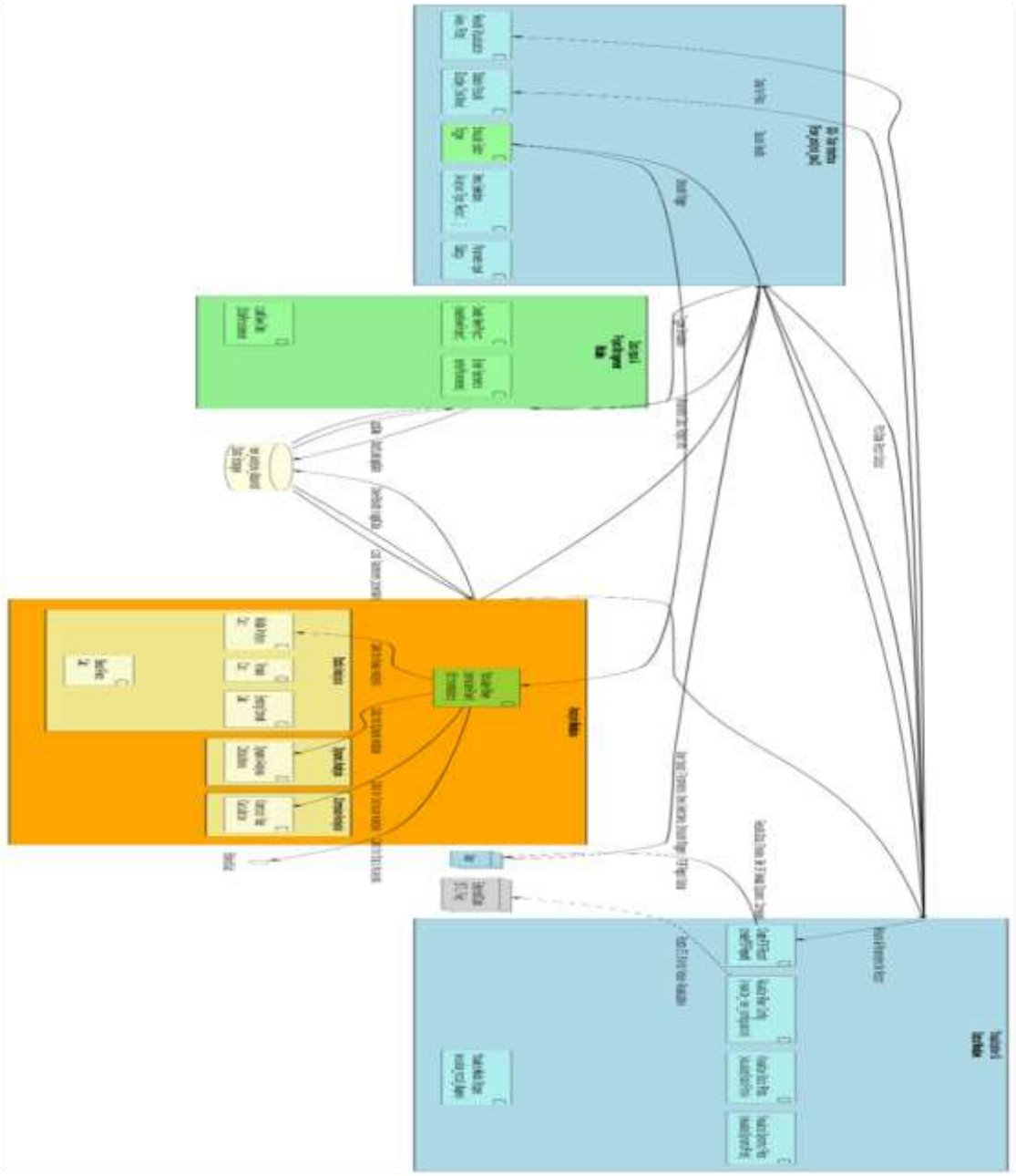


Fig. 4: Data Flow Diagram of RiSat Tool

As shown in Figure 4, user-provided input, encompassing simulation parameters and project-related information, is initially channeled through the GUI and subsequently managed by the Data Input & Project Management Modules. This module is responsible for orchestrating the persistent storage of the application data (appData) structure into the riser_analysis_data.mat file. This .mat file acts as the central data repository for RiSat, storing both input parameters and simulation results, ensuring data persistence across different sessions and enabling project resumption. When a simulation is initiated via the GUI's "Simulate" button, the trigger signal is passed to the Simulate Riser Module, which acts as the orchestrator of the analysis workflow. The SimulateRiserModule invokes the relevant modules within the Analysis Modules package (Static, Dynamic, Modal, and Corrosion analysis modules) to perform the core computations. During the analysis process, the analysis modules may load parameter data from the DataFile (via DataInput or directly) and, upon completion of the computations, save the generated results back into the appData structure within the DataFile. The Visualization & Output Modules then retrieve the computed results data from the DataFile (potentially via the Analysis Modules or Data Input modules), process this data, and generate both graphical visualizations displayed within the GUI (plots on axes, tabular results in text areas) and structured PDF reports for comprehensive documentation. Notably, the Visualize Riser Config Module accesses the external shipModel.stl file, representing the vessel geometry, to enhance the 3D visualization of the riser system [19]. This data flow design ensures a clear and consistent pathway for data from user input, through computational modules, to persistent storage and final output, promoting data integrity and traceability within the RiSat tool.

3 Risat Modules and Functionalities

3.1 Graphical User Interface (GUI)

The RiSat GUI, implemented using MATLAB's graphical user interface development environment (GUIDE), is designed to provide an intuitive and user-friendly interface for interacting with the tool [20]. The primary GUI component is the main figure window (Riser_analysis_tool2.m), which houses a custom-designed menu bar, axes objects for graphical result visualization, and a text area for displaying tabular simulation outputs. The menu bar, constructed using MATLAB's uimenu function, provides categorized access to RiSat's functionalities, including: "New Simulation" for project creation, "Simulation Type" for selecting analysis types (static, dynamic, corrosion), "Visualizations" for controlling plot displays, and "Report" for generating PDF reports. The central axes object (axes), a standard MATLAB graphics container, serves as the primary display area for plots generated by the visualization modules. The text area (uicontrol of style 'edit'), positioned below the axes, is used to present numerical simulation results in a structured table format. A prominent "Simulate" button (uicontrol of style 'pushbutton') located within the GUI triggers the execution of the core simulation process orchestrated by the simulateRiser function. Parameter input within the GUI is facilitated through dialog boxes generated using MATLAB's inputdlg function, ensuring structured and validated user input. The overall GUI design prioritizes ease of use and visual clarity, guiding the user through the logical steps of riser analysis from parameter definition to results interpretation [21].

3.2 Data Input and Project Management Module

The Data Input and Project Management module is responsible for handling project creation, parameter acquisition, and persistent data storage, ensuring project organization and data integrity. The "New Simulation" functionality, accessible via the "New Simulation" menu item, initiates the project creation process (createNewProject function). This function typically prompts the user to specify a project name and location, and then creates a dedicated project folder to house all project-related files, including parameter files (parameters.mat), data files (riser_analysis_data.mat), and potentially generated reports. Parameter input is managed through the "Enter Parameters" menu option, which invokes the enterParameters function. This function employs a series of customized dialog boxes (inputdlg) to systematically collect all necessary simulation parameters from the user. These parameters encompass riser geometry (e.g., outer diameter, inner diameter, length, number of segments), material properties (e.g., Young's modulus, density, Poisson's ratio), fluid properties (internal and external fluid densities), environmental conditions (water depth, wave parameters, current velocity), and analysis-specific settings (e.g., time step for dynamic analysis, corrosion parameters). To ensure data persistence and enable project resumption across different sessions, the Data Persistence module (DataPersistenceModule, encompassing functions like saveDataOnClose and implicit data loading at startup) utilizes MATLAB's built-in save and load functions to store and retrieve the entire application data structure (appData) to and from the riser_analysis_data.mat file. The appData structure serves as a centralized repository for all simulation parameters, intermediate calculation data, and final analysis results. This data management strategy not only ensures data integrity but also facilitates project sharing and reproducibility of analysis results [22].

3.3 Analysis Modules

3.3.1 Static Analysis Module:

The Static Analysis module provides a suite of functionalities for performing static analyses, including tension, bending moment, shear force calculations, and modal analysis. These analyses are fundamental for understanding the riser's behavior under steady-state loading conditions.

a. Tension, Bending Moment, Shear Force Analysis

Accessed through the "Static" sub-menu options ("Tension", "Bending Moment", "Shear Force") under "Simulation Type", these functionalities calculate and visualize the distribution of static tension, bending moment, and shear force along the riser arc length (visualizeResults function). The underlying computational framework for static analysis, implemented within the simulateRiser function and supporting functions in the StaticAnalysisCalc module, is based on the Euler-Bernoulli beam theory and principles of static equilibrium [23, 24]. The riser is discretized into a series of beam elements, and equilibrium equations are formulated considering self-weight, buoyancy forces, internal and external pressures, seabed interaction, and hydrodynamic drag forces (for steady currents). To account for the non-linear geometric behavior of the riser, particularly the lay angle and deformed configuration under load, RiSat employs an iterative numerical solution approach using MATLAB's fsolve function [25]. fsolve is used to solve a system of non-linear equations representing the static equilibrium conditions for each riser element, iteratively adjusting the riser configuration until equilibrium is achieved. The output of these static analyses includes distributions of tension, bending moment, and shear force along the riser arc length, which are then visualized by the Visualization & Output module using MATLAB's plotting functions.

b. Modal Analysis

The "Modal Analysis Prompt" option, also located under the "Static" sub-menu, initiates a modal analysis of the riser. Upon selection, the modalAnalysisPrompt function is invoked, which in turn triggers functions to visualize the initial riser configuration (visualize_riser_configuration) and subsequently visualize the extracted modal shapes (visualize_modal_shapes). Modal analysis, performed within the ModalAnalysisCalc module called by simulateRiser, determines the natural frequencies and mode shapes of the riser structure. This is achieved by solving an eigenvalue problem derived from the finite element discretization of the Euler-Bernoulli beam model, considering the riser's mass and stiffness matrices [26]. MATLAB's eig function, a powerful eigenvalue solver, is employed to compute the eigenvalues (natural frequencies squared) and eigenvectors (mode shapes) of the

system. The resulting natural frequencies and mode shapes provide valuable insights into the riser's dynamic characteristics and its susceptibility to vibration and resonance under dynamic excitations. The visualize_modal_shapes function utilizes MATLAB's 3D plotting capabilities (potentially using tubeplot and custom functions) to visualize the first few mode shapes, allowing users to understand the characteristic vibration patterns of the riser.

3.3.2 Dynamic Analysis Module

The Dynamic Analysis module enables the time-domain simulation of the riser's dynamic response to time-varying loads, such as wave and vessel motions. This module provides functionalities to calculate and visualize dynamic displacement, velocity, and acceleration of the riser. Options for Displacement ,Velocity, and Acceleration analysis are available under the "Dynamic" sub-menu within "Simulation Type". Selecting these options triggers the dynamicAnalysisPrompt function, which first prompts the user to select a specific riser section (e.g., top, middle, bottom) for detailed dynamic response visualization. The core dynamic analysis calculations are performed within the DynamicAnalysisCalc module, invoked by the simulateRiser function. RiSat utilizes the Newmark-beta explicit time integration scheme [27] to numerically solve the equations of motion governing the riser's dynamic behavior. The Newmark-beta method is a widely used and robust algorithm for time integration in structural dynamics, allowing for stable and accurate simulation of time-dependent responses. Hydrodynamic forces, arising from wave and current actions, are explicitly incorporated into the dynamic analysis formulation using Morison's equation or similar semi-empirical models [28]. These hydrodynamic forces are time-varying and are calculated based on user-defined wave parameters and current profiles. The simulateRiser function performs a step-by-step time integration, calculating the riser's displacement, velocity, and acceleration at each time step. The visualizeDynamicPlots and dynamicAnalysisPrompt functions are then used to visualize the time-series results. For a selected riser section, the displacement, velocity, and acceleration time histories are plotted in dedicated figure windows, using MATLAB's plot function, enabling users to analyze the temporal evolution of the riser's dynamic response at critical locations [29].

3.3.3 Corrosion Analysis Module

The Corrosion Analysis module provides a simplified assessment of corrosion rates for the riser material, considering environmental factors and electrochemical parameters. The "Corrosion Analysis" option, accessed via the "Corrosion Analysis" menu, initiates the corrosion analysis process (corrosionAnalysis function). Upon selection, the corrosionAnalysis function prompts the user to input corrosion-related parameters, such as current density, exposed area, and electrochemical properties of the riser material and seawater environment. The CorrosionCalc module, called by corrosionAnalysis, then calculates the corrosion rate based on a simplified model, often derived from Faraday's law of electrolysis and incorporating considerations for erosion-corrosion mechanisms relevant to offshore environments [30, 31]. The corrosion rate calculation may consider factors such as fluid velocity (erosion-corrosion effect), pressure, temperature, and electrochemical potential. The results of the corrosion analysis, typically corrosion rate values as a function of velocity and pressure, are then visualized as plots using MATLAB's plotting functions within the corrosionAnalysis function, allowing users to assess the potential for material degradation over the riser's operational life. It's important to note that the corrosion analysis module in RiSat provides a simplified, indicative assessment and may not encompass the full complexity of real-world corrosion phenomena. For detailed corrosion analysis, specialized corrosion engineering software and experimental investigations are typically required.

3.4 Visualization and Output Module:

The Visualization and Output module provides comprehensive functionalities for presenting simulation results in graphical and report formats, enhancing result interpretation and communication.

a. Static Plots: The "Static Plots" option under the "Visualizations" menu (visualizeStaticPlots function) is designed to generate a composite figure displaying key static analysis results. This figure typically includes subplots of static tension, bending moment, and shear force distributions plotted against the riser arc length. In addition to these 2D plots, the visualizeStaticPlots function also incorporates a 3D visualization of the riser configuration, and potentially the floating vessel and seabed, to provide a spatial context for the static analysis results. The 2D plots are generated using MATLAB's plot and subplot functions, while the 3D visualizations leverage specialized 3D plotting functions, potentially including tubeplot for representing the riser geometry as a 3D tube, and custom functions or readily available MATLAB Central File Exchange functions like draw_vessel_and_seabed_3d [32] for visualizing the vessel and seabed. The stlread function, used to import STL files, is utilized to load the shipModel.stl file containing the 3D geometry of the floating vessel for inclusion in the 3D visualization. Spline interpolation, implemented using MATLAB's spline_fit_3D and spline functions [33], is employed to smooth and refine the 3D representation of the riser geometry, ensuring visually appealing and accurate visualizations.

b. Dynamic Plots: The Dynamic Plots option within the Visualizations menu (visualizeDynamicPlots function) enables the visualization of time-series results obtained from dynamic analyses. When selected, visualizeDynamicPlots generates separate figure windows, each dedicated to plotting the time history of a specific dynamic response parameter (displacement, velocity, or acceleration) for the riser section previously selected by the user via the dynamicAnalysisPrompt function. These time-series plots are generated using MATLAB's plot function, with appropriate axis labels (time in seconds, and the respective response unit) to facilitate interpretation of the temporal dynamic behavior of the riser.

c. Riser Configuration Visualization: The "Restore" option in the "Visualizations" menu, as well as the modal analysis prompt, both utilize the visualize_riser_configuration function to generate 3D visualizations of the riser configuration. This function, similar to the 3D visualization component in visualizeStaticPlots, employs tubeplot and draw_vessel_and_seabed_3d (or similar 3D plotting functions) to display the 3D shape of the riser in its analyzed configuration. In the context of modal analysis, this function is used to visualize the mode shapes, representing the characteristic deformation patterns of the riser at its natural frequencies.

D. Report Generation: The "Report" menu option (createPDFReport function) provides an automated mechanism for generating comprehensive PDF reports summarizing the simulation project and its results. This functionality leverages MATLAB's mlreportgen library [34], a powerful tool for programmatically creating structured documents in various formats, including PDF. The createPDFReport function is designed to automatically compile a report document that typically includes a title page, a table of contents, sections detailing the simulation parameters used, and sections presenting the simulation results. Both tabular results (extracted from the text area in the GUI) and graphical results (embedded plots of static, dynamic, and modal analyses) are incorporated into the generated PDF report, providing a complete and self-contained documentation of the riser analysis project. The use of mlreportgen allows for highly customizable report templates and programmatic control over report content and formatting, enabling the generation of professional-quality reports directly from within RiSat.

4 Case Study of a Steel Catenary Riser

To demonstrate the practical application and output capabilities of RiSat, a case study is presented focusing on the analysis of a steel Lazy riser (SLR) in a typical offshore environment. The riser is assumed to be a 1000-meter-long steel pipe with an outer diameter of 0.3 meters and a wall thickness of 0.02 meters, deployed in a water depth of 800 meters. The riser material is assumed to be API 5L X65 steel, and the internal fluid is oil with a specific gravity of 0.85. Environmental conditions are defined by a representative sea state with a significant wave height of 5 meters and a peak period of 10 seconds, and a steady current profile with a surface velocity of 1 m/s, decreasing linearly to zero at the seabed.

Using RiSat, a static analysis is first performed to determine the initial configuration and static loads on the riser under its own weight, buoyancy, and steady current. The "Simulate" button is clicked to initiate the static simulation, and the resulting tension, bending moment, and shear force distributions are then visualized using the "Static Plots" option in the "Visualizations" menu. Figure 5 shows a representative static tension plot generated by RiSat for this case study and was compared with orcaflex as well, illustrating the tension variation along the riser arc length, with maximum tension typically observed near the vessel connection point and minimum tension at the seabed touchdown point.

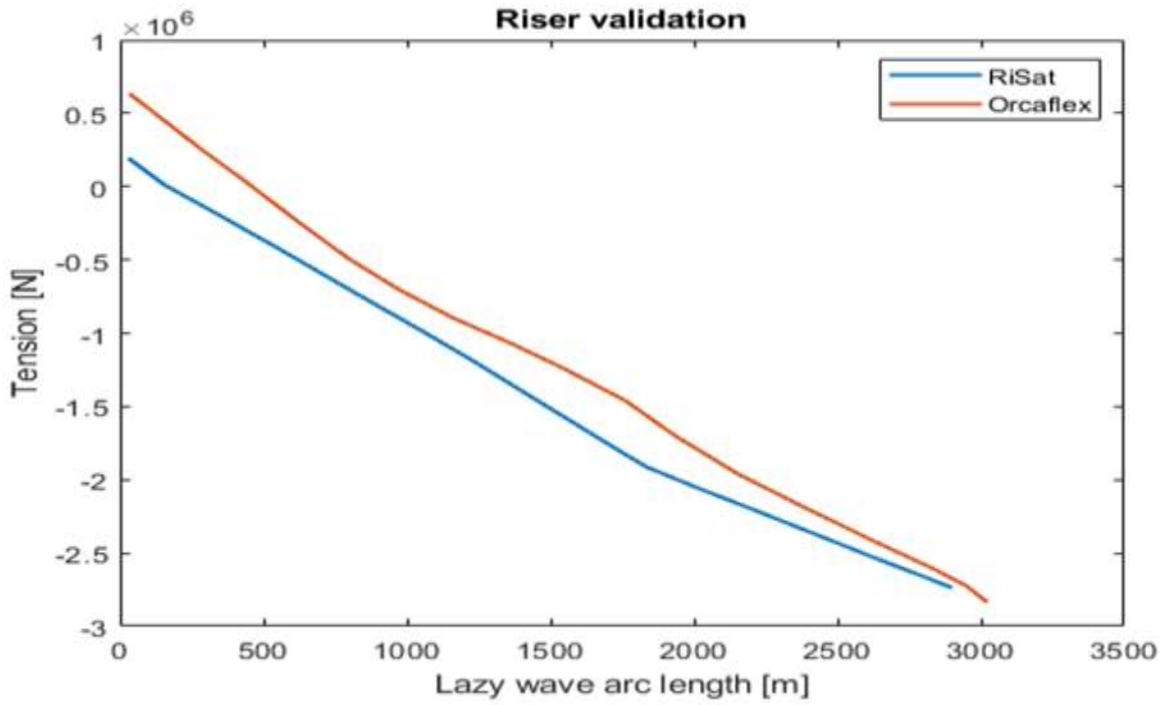


Fig. 5: Example Static Tension Plot from RiSat for Case Study

Next, a dynamic displacement analysis is conducted to assess the riser's response to wave loading. The "Simulation Type" is set to "Dynamic," and "Displacement" is selected under the "Dynamic" sub-menu. RiSat prompts the user to select a riser section for dynamic response visualization, and the "middle section" is chosen for this example. The dynamic simulation is executed, and the resulting displacement time series for the middle section is then visualized using the "Dynamic Plots" option. Figure 6 presents an example dynamic displacement plot, showcasing the temporal variation of the riser's horizontal displacement at the middle section in response to the defined wave loading. The plot reveals the oscillatory nature of the riser's dynamic response, with amplitudes and frequencies dictated by the wave characteristics and riser system properties.

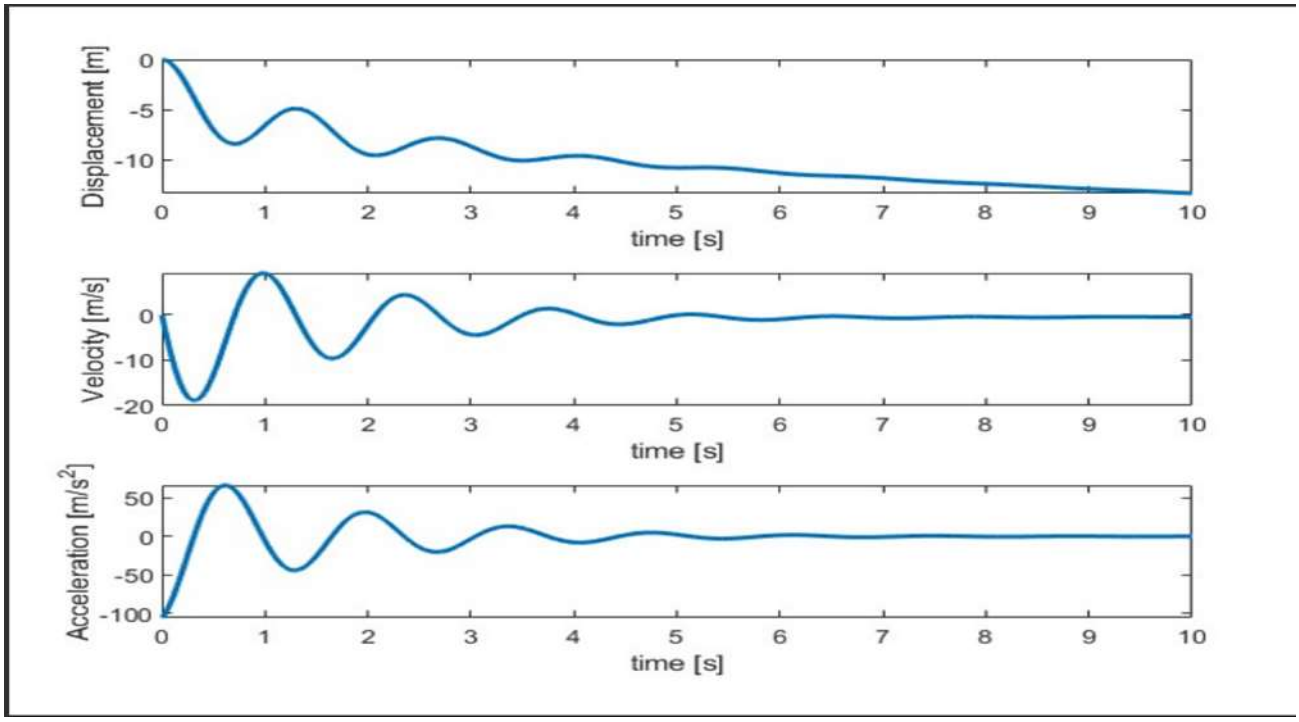


Fig. 6: Example Dynamic Displacement Plot from RiSat for Case Study

Finally, a modal analysis is performed by selecting “Modal Analysis Prompt” from the “Static” sub-menu. RiSat visualizes the initial riser configuration and then, upon user confirmation, calculates and visualizes the first few mode shapes of the riser. Figure 7 illustrates an example visualization of the first mode shape, showing the characteristic deformation pattern of the riser at its fundamental natural frequency.

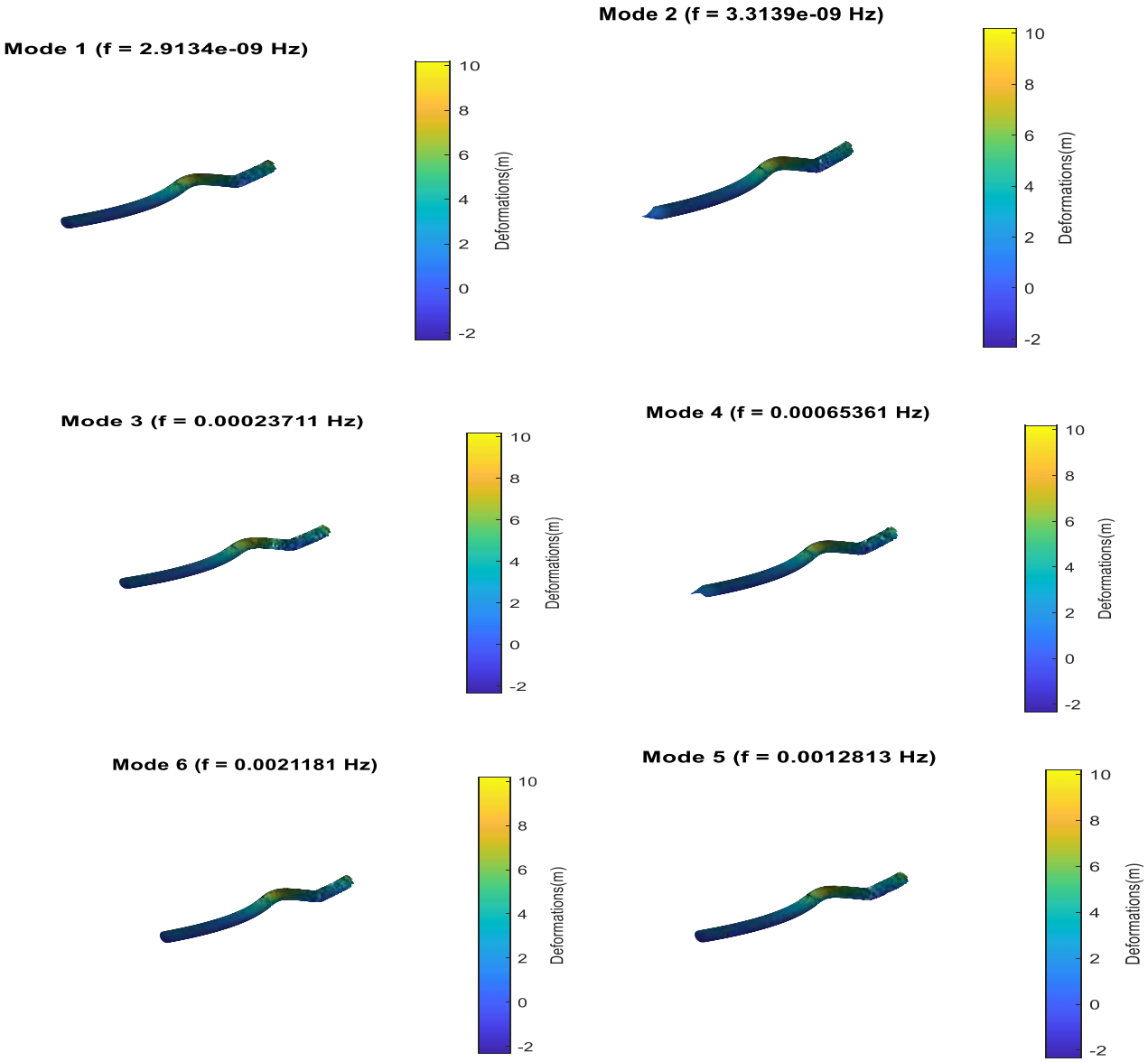


Fig. 7: Example Modal Shape Visualization from RiSat for Study

These example visualizations, generated by RiSat for a representative steel catenary riser case study, demonstrate the tool's capability to perform comprehensive static, dynamic, and modal analyses and present the results in a visually informative manner. Furthermore, a PDF report summarizing the simulation parameters and results can be generated using the "Report" menu option, providing a complete documentation of the case study analysis.

5 Discussions and Conclusions

RiSat presents a robust and user-friendly MATLAB-based software tool for comprehensive riser analysis in offshore engineering. By integrating static analysis (tension, bending moment, shear force, modal), dynamic analysis, and corrosion assessment within a single, cohesive platform, RiSat offers a streamlined workflow for riser engineers and researchers. The tool's GUI, built using MATLAB's GUIDE environment, ensures intuitive interaction, while the modular architecture promotes maintainability and future extensibility. The implementation of established numerical methods, such as Euler-Bernoulli beam theory, the Newmark-beta method, and semi-empirical corrosion models, provides a sound theoretical basis for the analysis capabilities [35, 36]. The extensive visualization functionalities, encompassing 2D plots, 3D riser and vessel visualizations, and automated report generation, enhance the interpretability and communication of simulation results. The use of MATLAB as the development platform leverages its powerful numerical computing engine, extensive libraries, and platform independence, making RiSat a portable and readily deployable tool.

However, RiSat, in its current form, has certain limitations. The corrosion analysis module provides a simplified assessment and does not capture the full complexity of real-world corrosion phenomena. The dynamic analysis module, while implementing the Newmark-beta method, could be further enhanced

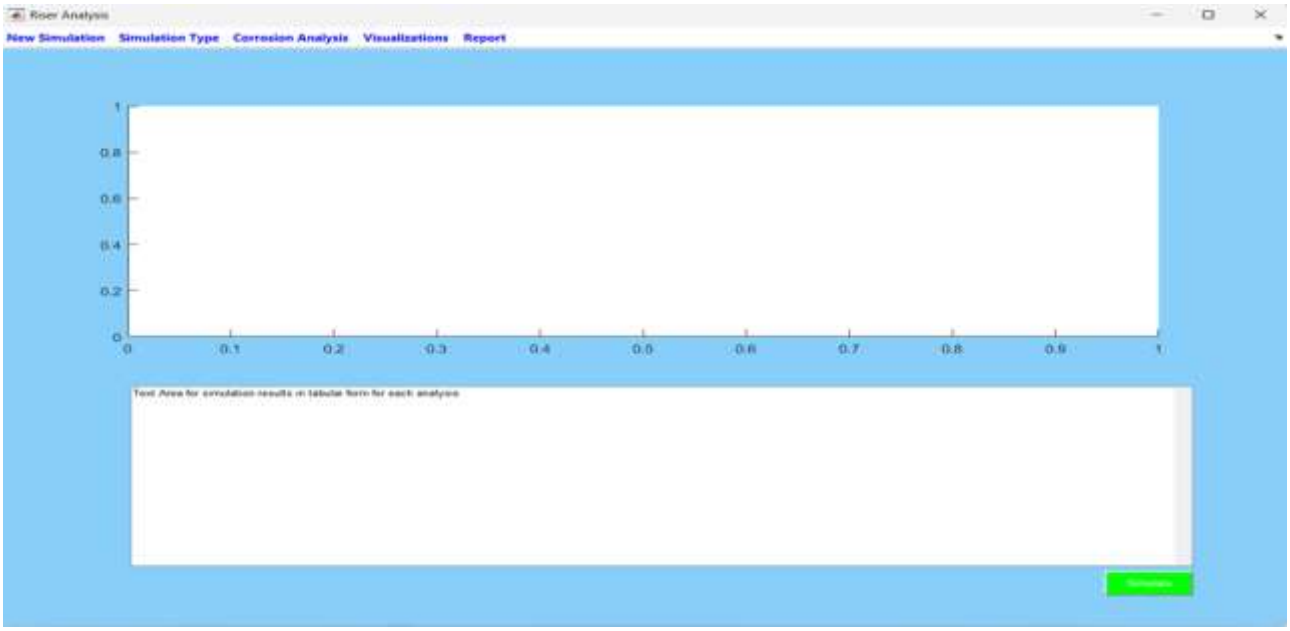
to incorporate more sophisticated hydrodynamic models and account for non-linear riser behavior and fluid-structure interaction with greater fidelity. Furthermore, the tool's accuracy and reliability would benefit from comprehensive validation against experimental data and benchmark commercial software solutions [37, 38].

Future developments of RiSat could focus on addressing these limitations and expanding its capabilities. Potential enhancements include: incorporating more advanced FEA-based analysis methods for increased accuracy and handling of complex riser geometries and material nonlinearities [39]; expanding the dynamic analysis module to account for vortex-induced vibration (VIV) and fatigue analysis [40]; integrating more detailed and experimentally validated corrosion models [41]; enhancing the GUI with interactive visualization features and parametric study capabilities [42]; and developing a more comprehensive validation and verification framework.

In conclusion, RiSat offers a significant contribution to the field of offshore riser engineering by providing a practical, accessible, and computationally efficient MATLAB-based tool for comprehensive riser analysis. Its user-friendly design, integrated functionalities, and potential for future development position it as a valuable asset for both educational purposes and engineering applications in the offshore oil and gas industry, facilitating improved riser design, integrity assessment, and operational safety in challenging offshore environments.

Appendix

The Risat GUI main interface is shown in the figure below. The program commences from the left menu bar to the right. Like every conventional program it starts with the creation of a new project, to the simulation and reporting.



REFERENCES

[1] Bai, Y., & Bai, Q. (2014). Subsea pipelines and risers. Elsevier.

[2] Chakrabarti, S. K. (2005). Handbook of offshore engineering (Vol. 1). Elsevier.

[3] API RP 2RD. (2013). Recommended practice for design and operation of riser systems for floating production applications. American Petroleum Institute.

[4] Patel, M. H., & Park, I. R. (2015). Dynamic analysis of deepwater risers and umbilicals. Ocean Engineering, 101, 1-16.

[5] Melchers, R. E. (2018). Structural reliability analysis and prediction. John Wiley & Sons.

[6] Bea, R. G. (2016). Offshore platform and marine pipeline risk assessment and management. Journal of Offshore Mechanics and Arctic Engineering, 138(4), 041101.

[7] HSE. (2015). Offshore Oil and Gas Industry: Key Statistics and Regulatory Performance. Health and Safety Executive. https://www.hse.gov.uk/offshore/statistics/index.htm (Accessed: October 26, 2023).

[8] DNV GL. (2017). DNVGL-ST-F101: Offshore standard for submarine pipeline systems. DNV GL AS.

[9] Zienkiewicz, O. C., & Taylor, R. L. (2000). The finite element method: solid mechanics. Butterworth-Heinemann.

[10] Cook, R. D., Malkus, D. S., Plesha, M. E., & Witt, R. J. (2007). Concepts and applications of finite element analysis. John Wiley & Sons.

[11] Wheeler, S. J., Zaki, M. B., & Rawlins, C. H. (2019). Rapid prototyping of offshore structures using finite element analysis. Marine Structures, 63, 203-218.

[12] Sriramula, M. V., Chakraverty, S., & Rao, B. N. (2017). Advances in computational methods for offshore structures. Archives of Computational Methods in Engineering, 24, 897-920.

[13] Sedlak, P., Ferkl, L., & Krejsa, J. (2020). Open-source tools for offshore engineering: A review and case study. Renewable and Sustainable Energy Reviews, 134, 110139.

[14] Bass, L., Clements, P., & Kazman, R. (2012). Software architecture in practice. Addison-Wesley Professional.

[15] MathWorks. (2023). MATLAB Plotting Tools Documentation. https://www.mathworks.com/help/matlab/graphics.html (Accessed: October 26, 2023).

[16] Moler, C. (2004). MATLAB numerical computing with MATLAB. Society for Industrial and Applied Mathematics.

[17] Gere, J. M., & Goodno, B. J. (2009). Mechanics of materials. Cengage Learning.

[18] Chopra, A. K. (2019). Dynamics of structures: theory and applications to earthquake engineering. Pearson Education.

[19] Zeid, I. (2010). CAD/CAM theory and practice. McGraw-Hill Education.

[20] MathWorks. (2023). MATLAB GUIDE Documentation. https://www.mathworks.com/help/matlab/gui-building-with-guide.html (Accessed: October 26, 2023).

[21] Johnson, J. (2016). GUI design handbook: principles and practices for user interface design. Newnes.

[22] Quarteroni, A., & Saleri, F. (2017). Scientific computing with MATLAB and Octave. Springer.

[23] Hibbeler, R. C. (2017). Mechanics of materials. Pearson.

[24] Reddy, J. N. (2017). An introduction to the finite element method. McGraw-Hill Education.

[25] Fausett, L. V. (2008). Applied numerical mathematics using MATLAB. Pearson Education.

[26] Clough, R. W., & Penzien, J. (2003). Dynamics of structures. Computers & structures, inc.

[27] Hughes, T. J. R. (2012). The finite element method: linear static and dynamic finite element analysis. Courier Corporation.

[28] Faltinsen, O. M. (2017). Sea loads on ships and offshore structures. Cambridge university press.

[29] Wirsching, P. H., Paez, T. L., & Ortiz, K. (2006). Random vibrations: theory and practice. John Wiley & Sons.

[30] Roberge, P. R. (2018). Handbook of corrosion engineering. McGraw Hill Professional.

[31] Revie, C. W. (2015). Offshore structural engineering: design and reliability. CRC Press.

[32] D'Errico, J. (2023). draw_vessel_and_seabed_3d. MATLAB Central File Exchange. https://www.mathworks.com/ matlabcentral/fileexchange/ 24484- drawvessel_and_seabed_3d (Accessed: October 26, 2023).

[33] Piegl, L., & Tiller, W. (2012). The NURBS book. Springer Science & Business Media.

[34] MathWorks. (2023). MATLAB Report Generator Documentation. https://www.mathworks.com/help/rptgen/index.html (Accessed: October 26, 2023).

[35] Murdock, T. G. (2015). Offshore structural design. PennWell Books.

[36] Barltrop, N. D. P., & Adams, A. J. (2012). Dynamics of fixed marine structures. Butterworth-Heinemann.

[37] ASME V&V 40-2018. (2018). Assessing credibility of computational modeling through verification and validation: application to medical device simulation. American Society of Mechanical Engineers.

[38] Oberkampf, W. L., & Roy, C. J. (2010). Verification and validation in scientific computing. Cambridge university press.

[39] Kwon, Y. W., & Bang, H. (2010). The finite element method using MATLAB. CRC Press.

[40] Blevins, R. D. (2005). Flow-induced vibration. Krieger publishing company.

[41] Davis, J. R. (Ed.). (2000). Corrosion of materials. ASM international.

[42] Nielsen, J. (2012). Usability engineering. Elsevier.