Visual Analytics to Support Atomistic Simulations Design

Daniel Felipe Mejia Padilla
Purdue University

VISUAL ANALYTICS TO SUPPORT ATOMISTIC SIMULATIONS DESIGN

A Dissertation

Submitted to the Faculty

of

Purdue University

by

Daniel Felipe Mejia Padilla

In Partial Fulfillment of the

Requirements for the Degree

of

Doctor of Philosophy

August 2018

Purdue University

West Lafayette, Indiana

THE PURDUE UNIVERSITY GRADUATE SCHOOL

STATEMENT OF DISSERTATION APPROVAL

To the memory of my father.

ACKNOWLEDGMENTS

I would like to express my gratitude to my advisor Gerhard Klimeck, and the core NEMO5 developers Jim Fonseca, Michael Povolotsky, and Tillman Kubis. I wish to thank my wife and my family for their unconditional love and support.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

ABSTRACT

Daniel Mejia Ph.D., Purdue University, August 2018. Visual Analytics to Support Atomistic Simulations Design. Major Professor: Gerhard Klimeck.

Nowadays, complex simulations of a variety of processes are extensively used in academia and industry. Particularly in academia, powerful scientific software tools are constantly developed to simulate complex systems; for instance, simulations of quantum transport using the non-equilibrium greens Function formalism. The potential impact of these scientific tools in industry is huge, but it is hindered by the lack of usability of the software by those who are not deeply familiar with it. Visual analytics is a new field that has shown the positive impact of interactive visualizations in software usability and the cognitive process of the user. This research investigates whether the implementation of interactive visual aids also improves the usability and the cognitive processes of research codes users, particularly those used for simulation design. To accomplish this goal, this study defines a framework for simulation design in scientific research, identifies the stages in which visual aids can be implemented to increase usability, and implements an interactive visualization system (NemoViz). NEMO5, a tool for designing atomistic simulation, is used as a case study to measure the effectiveness, efficiency, and user satisfaction of the use of visual aids in scientific simulation design. The results from this research provide a framework of reference for development of user-friendly simulation design tools, and will shed light on strategies that scientific developers might implement to broaden the impact of their simulation codes.

CHAPTER 1. INTRODUCTION

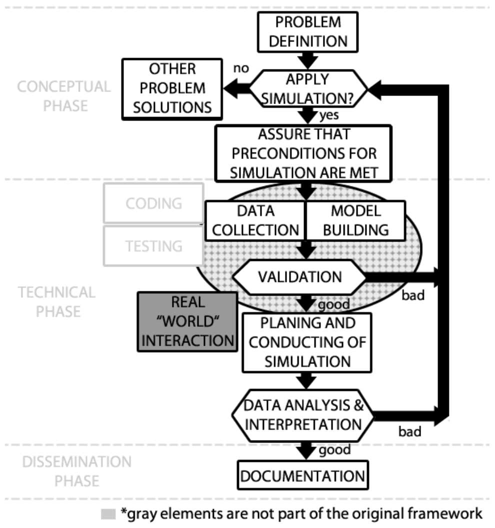
Nowadays, complex simulations of a variety of processes are extensively used in academia and industry. For instance, since the 1960s, the drift diffusion model and the understanding of electron transport in transistors have driven work in computational electronics at the mesoscopic scale [1]. Most recently, other models, such as non-equilibrium greens function (NEGF) formalism for simulating quantum transport, have been incorporated into codes to simulate phenomena at the nanoscopic scale. However, these codes demand considerable user experience in semiconductor physics to operate them properly [1]. This additional knowledge includes new layers of information in the cognitive process of the user [2]; for example, crystallographic information and other material science concepts. Additionally, the customization of each simulation creates cognitive challenges associated with the need to control parameters and the generation of process definitions and simulation models, among others [2] [3]. Designers analyze parameters and their relations, create simulation models, and evaluate results, among other cognitive challenges. This work focuses on addressing the challenges of scientific simulation design by studying the impact of visualizations on the cognitive process of simulation design. The cognitive process required for simulation design can be improved by resources such as tutorials, manuals, visual aids, web resources, and others. The main advantage of manuals and tutorials is that the user learns by example, which reduces the learning curve of the simulation tools and allows users to efficiently achieve results [4]. The downside of manuals and tutorials is that they restrict, or bias, the users thinking, limiting the capacity of the simulation tool [5] [6]. Visual aids improve the cognitive process by including external cognition as part of the process [7]. Properly designed, interactive visual aids allow the user to rapidly understand the simulation tool and get results, and also creates the flexibility to explore different aspects of the simulation. These advantages no longer hold true when the visual aid is poorly designed. Visualizations are usually part of a graphical user interface (GUI). Several studies have shown that the use of visual representations and GUIs contribute to a more efficient transfer of knowledge by reducing the computer users cognitive load [8]. Currently, almost all software applications include a GUI; in some cases, there is no distinction between the program and the GUI. The software industry complements their developments by adding visual tools and programmatic interfaces. GUI design has become a main part in the software development life cycle. However, this prevalence of GUIs does not hold true in the case of scientific computing. Scientific computing typically deals with complex problems and extensive computational solutions. Developing a final user interface seems to be the least of scientific software developers concerns. The interaction between the software and the final user is determined by the users ability to customize the execution of the program by modifying a configuration file. Scientific software tools are focused on solving the scientific problem, and they assume the user is knowledgeable both in the field of study and about the tool itself. Non-user-friendly software is not easily adopted by other scientists or successfully transferred and used in industrial applications. Unfortunately, this is the case for many powerful simulation codes that never leave university computers because they are difficult for non-experts to use. Most scientific software developers are not aware of this limitation. To enhance the impact of research codes, particularly in the field of modeling and simulation, there is a need to standardize modeling and simulation frameworks and tools. This will encourage the scientific community to create user-friendly codes This research investigates how to better design effective, user-friendly simulations in scientific research. Specifically, this research proposes the integration of visual analytics to enhance the usability of scientific codes by non-experts in other scientific fields and industry. Visual analytics is a new field of information visualization that focuses on analytical reasoning facilitated by interactive visual interfaces [9].

This thesis is organized as follows: Chapter 2 describes different simulation frameworks or models, and proposes a new framework based on key aspects of previously published frameworks. Chapter 3 describes the implementation of an interactive visualization system called NemoViz, which supports the framework proposed in this research. Chapter 4 presents some of the high-quality images obtained using NemoViz. Chapter 5 describes some of the additional contributions that were made to the Nemo 5 code along with the development of NemoViz. Chapter 6 describes the formal evaluation performed to evaluate NemoViz. Finally; Chapter 7 presents general conclusions and recommendations for future work.

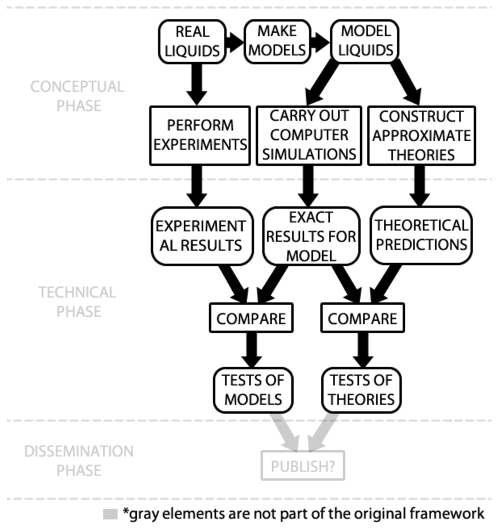
Additionally, appendices present the validation tutorials, input files used during the validation, all data obtained from the evaluation, and the paper presented in the 2015 IEEE International Conference on e-Science describing the interactive analytic system used to measure Nanohub impact. nanoHUB.org is a cyberinfrastructure where researchers, and educators collaborate, share resources, and simulate real nanotechnology problems.

CHAPTER 2. SIMULATION FRAMEWORK

2.1 Introduction



(a) Kruger simulation workflow.



(b) Allen et al. model.

Fig. 2.1. Simulation design frameworks.

Three main frameworks, illustrated in Figures 2.1 and 2.2, describe the simulation process. The first formal documented definition of a simulation workflow model was proposed by Krüger et al. in the 1970s [10]. Krüger proposed that the simulation workflow begins with the problem definition and undergoes the stages of data collection and model building, validation, data analysis and interpretation, and finally, documentation (2.1(a)). His model introduced the term *real world interaction*, which encompasses data collection, model building, and validation as the core of the process. While his model is mostly linear, it is the first model to define the design simulation loop. However, according to his model, the only two ways to trigger the loop is through a bad validation of the model or a bad validation of the results. In the 1990s, Allen et al. proposed a simulation workflow designed for scientific simulations, specifically for fluid simulations [11]. In his model, Allen highlights the importance of conducting experiments in the real world and compares the results of the experiments with the simulation results using theoretical models (Figure 1b). He clarifies the definition of *real world interaction* as interactions between user and data that arise from theoretical models or experimental results [11]. An interesting feature of this model is that data acquisition is part of its process This is not always necessary, since existing data can be used as validation sets. In this case, experimental data generation or collection is not part of the simulation process.

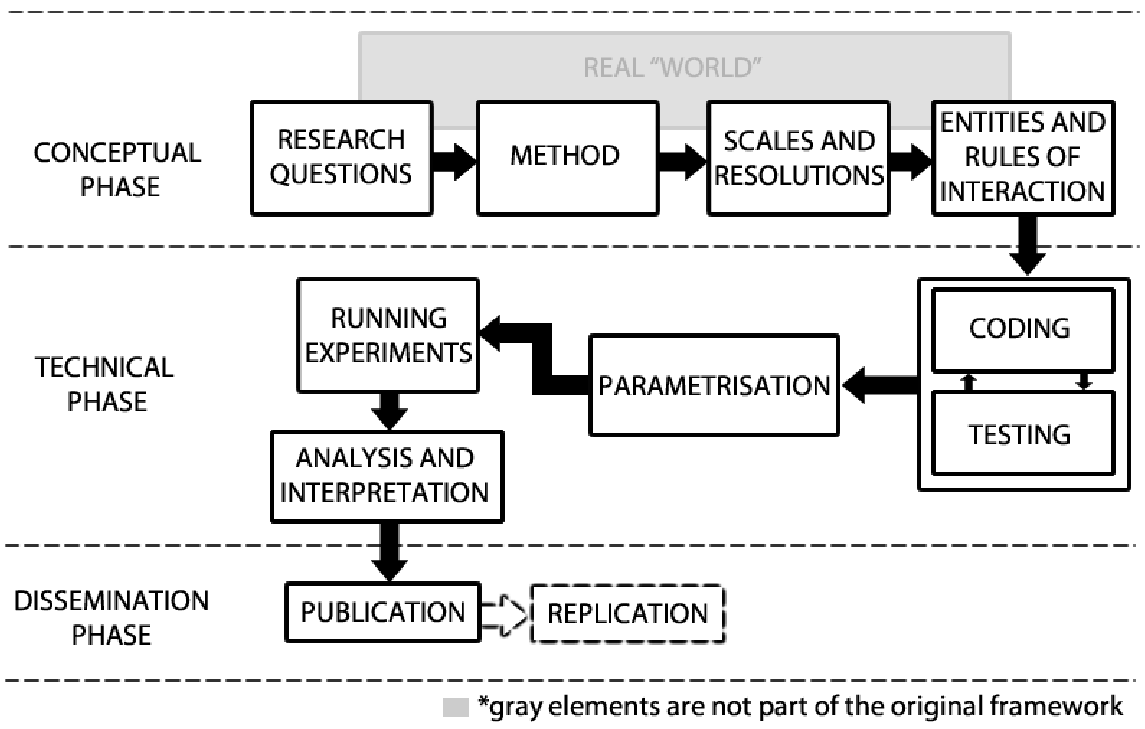


Fig. 2.2. Romanowska process for building and evaluating simulation models

Most recently, Romanowska [12] proposed another framework for the simulation process (see Figure 2.2). His model is unique in its flexibility navigating from one stage of the process to another in no particular order. This model also includes two additional stages: coding and testing, and result replication.

Combined, all of the above frameworks present the simulation process steps. However, when looking at each framework individually, one or more steps are missing. As the simulation process in scientific research becomes more and more complex, there is a need to integrate these frameworks into a single, more detailed model that describes all possible steps and their interactions.

2.2 Combined Model

The combined framework proposed in this study is mainly based on work by Romanowska [12]. It includes elements from the models proposed by Kruger [10] and Allen et al. [11], and explicitly defines the possible interactions between stages. This last feature is not present in any of the models proposed by other authors to date. Figure 2.3 presents the schematic of the proposed framework. Each box represents a stage in the simulation design process. The dotted boxes note input information, but they are not part of the simulation design process itself. Possible relations between stages are shown using lines that connect the stages.

The process begins with the *conceptual phase*. This phase consists of one stage. the concept stage. Some authors refer to this stage as the identification of the research question or the definition of the problem. During this stage, a simulation designer defines the studys premises and assumptions, and tries to align some of those concepts with a relevant question and possible solution. Usually, these premises and assumptions come from previous knowledge of theoretical models or data reported by experiments in scientific research; unfortunately, the designer is biased about the expected results based on the subjective nature of human cognition and the nature of human cognition. After the concept is clearly defined, the technical phase follows. The first step in this phase is the *design model* stage. In this stage the designer represents the model in some technical language, from mathematical frameworks to specific

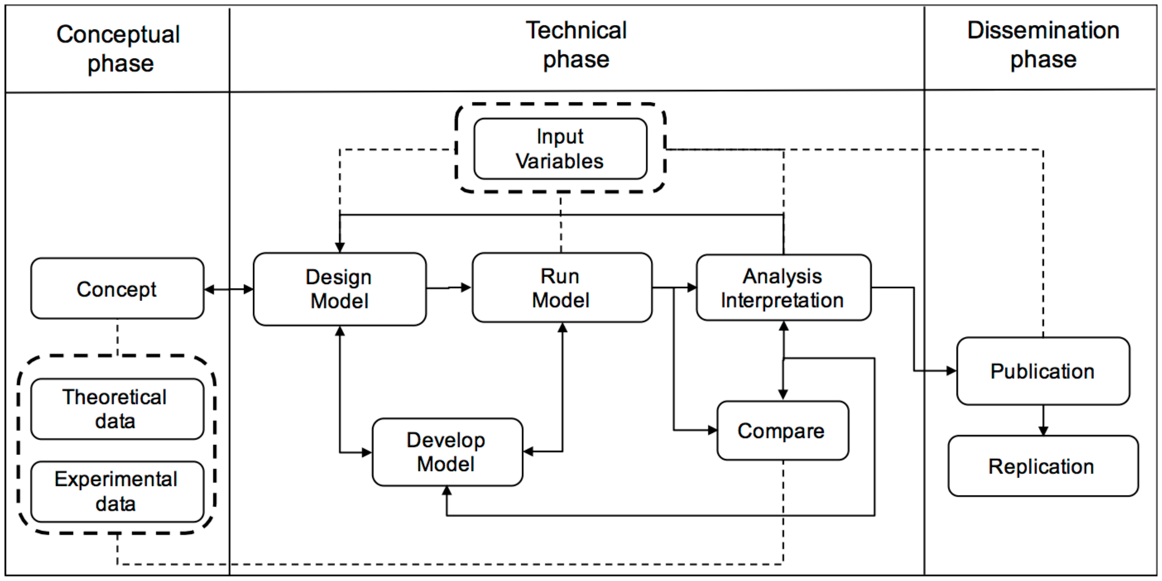


Fig. 2.3. Schematic of the proposed framework for the simulation design process

simulator inputs. Unless there is a completely new model, designers prefer to model their concepts in well-defined research codes or a commercial simulator. This requires modification to a relatively high number of parameters to represent the model. All languages have their restrictions, and those restrictions can force the designer to prefer a particular language to represent the model, choose a new simulator, develop a new simulator, or re-evaluate the concept altogether. If the simulator allows for the functionality desired by the designer, the natural process is to advance to the *run model* stage. If not, new code should be developed. This is the case for many research tools that are considered incomplete; new functionality needs to be implemented or requested. In the *develop model* stage, the designer must advance through a traditional software development life-cycle: analysis, design, coding, and testing. This process has been widely studied. Multiple frameworks have been proposed and they are beyond the scope of this work. However, in the last decade, extreme programing has become the standard in academia and industry due to its flexibility. Since high performance computing (HPC) is strongly connected to simulation, some extra profiling tools are required along with development. After the *design model* stage is the *run model* stage. This stage seems trivial, but it is not. Nowadays, simulators claim to be HPC efficient; therefore, the model should run in highly parallel systems and run in heterogeneous systems that include accelerator cards, such as those installed in the two main supercomputers Titan (USA) and Tianhe2 (China). The former uses graphical processor units (GPUs), and the latter uses an Intel many integrated core architecture (Intel MIC). The complexity of the execution requires some expertise. Extra parameterization of variables can be used to run the model with specific rules; for instance, the number of cores per node or multi-threading. The execution will produce result data with which to compare the model with real world data. This is the *compare* stage. Data from experiments or theoretical models represent specific observables that can differ from the observables simulated by the model. Some post-processing is required to transform data into a similar shape, and the statistical model can be applied to make conclusions. In the *analysis and interpretation* stage, results from the previous stage are examined, hypotheses are created, and new ideas are generated. These ideas are materialized into new concepts, new simulation designs, and conclusions. Conclusions provide the input for the next phase. During the *dissemination* phase, researchers share the new knowledge they have acquired, as they would following any scientific research process. In general, the dissemination of knowledge is done by publishing new findings in journals or in conference proceedings. However, it is difficult to publish all parameters used by the simulation or the exact code used in the simulation, and the reproducibility of results can, therefore, be compromised. The process of reproducibility can be addressed by publishing a snapshot of the simulation code in a specialized cyber-infrastructure. The benefit of this model is its clear definition of steps and their connections; this allows for a clear definition and measurement of the impact of the visualization tools.

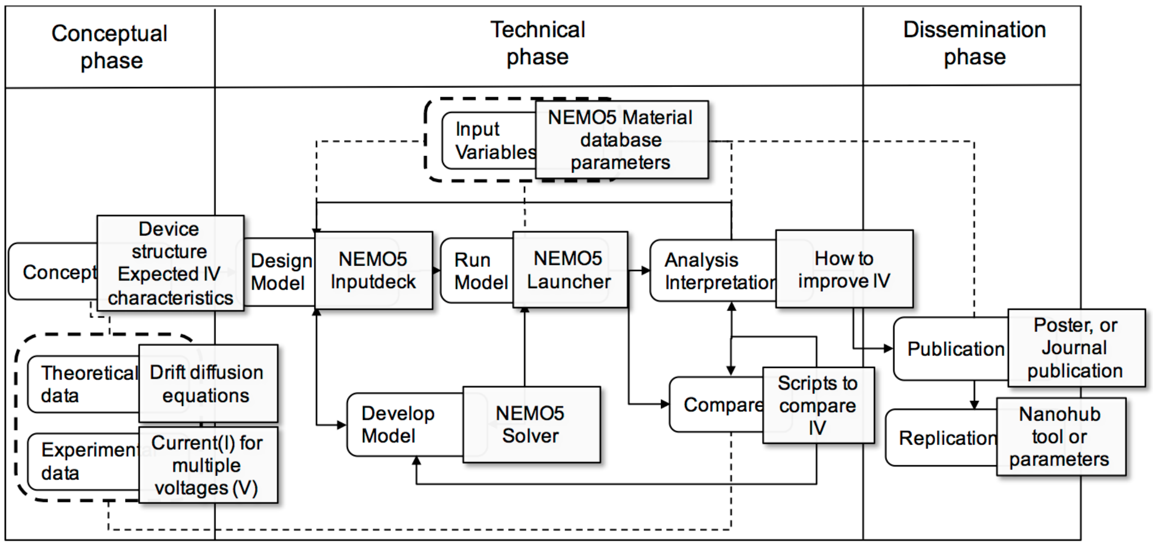


Fig. 2.4. Mapping of the proposed simulation design process using NEMO5

2.3 Case study

The proposed framework describes the process of designing simulations in the field of nanoelectronics using NEMO5 as the simulation engine. Figure 2.4 presents an example of how the framework can be applied to a NEMO5 simulation. The conceptual phase in NEMO5 is mainly defined by the users. For instance, the user may want to test different device designs and their IV characteristics to select the best model. The initial design might be based on experimental results and/or theoretical data from drift diffusion equations.

The *technical phase* incorporates the input variables from the user, which NEMO5 pulls from the material database. The *design model* stage in NEMO5 is defined by the input deck. For instance, the input deck defines the devices crystallographic information, geometrical structure, physical model (QTBM or RGF), and the restrictions of the physical model (boundary conditions for Poisson's equation). The *develop model* stage in NEMO5 consists of the implementation of a solver in C++ or Python. The *run model* stage consists of the execution of the input deck. This is usually done with the NemoLauncher. The other two stages of the *technical* phase, the *comparison* stage and the *analysis and interpretation* stage, are not covered by NEMO5. Usually, during the *comparison* stage, NEMO5 users create their own visualization scripts; for example, to compare IV curves. In the *analysis and interpretation* stage, users decide which device parameters to change in order to improve the device characteristics; for instance, to improve the relation current versus its voltage. These decisions might lead to new simulations. The *dissemination* phase is not covered by NEMO5 either. In this phase, the user presents the NEMO5 simulation results in journals, at scientific conferences, and other venues (the *publication* stage). Finally, the user publishes the final simulation parameters to allow reproducibility; on a few occasions, the simulation is published as a tool in Nanohub to ensure reproducibility of results by other researchers.

2.4 Conclusions

This chapter proposed a combined model that describes the simulation process and delineates the main steps faced by simulation users. This framework could help to describe the cognitive impact of visualizations in the process of designing a simulation, particularly the impact of visualization on Nemo5's users.

CHAPTER 3. INTERACTIVE VISUALIZATION SYSTEM

In this research, an interactive visualization system was designed involving the complete simulation process described on Figure 2.3. The main functional requirements for each phase are

- Conceptual phase: users visualize data from external sources, learn about possible physical models, and perform parameter exploration on those models.
- Technical Phase: users import simulation inputs, visualize spatial information of loaded inputs, visualize no-spatial information and its relations with other inputs, visualize simulation results, and visualize useful information to debug new models.
- Dissemination phase: users export inputs as tools and graphical material that support results' reproducibility.

The following sections, discuss how these requirements were fulfilled.

3.1 Requirements

3.1.1 Conceptual phase

During this process, users upload raw data into the system, and data is processed and transformed into visual representations such as heatmaps or line plots. Users can visualize available physical models in the system, and access documentation for each specific model. The documentation contains descriptive information about assumptions, formulas, restrictions, and possible parameters of the physical model. Users can also explore databases containing material properties and change values.

3.1.2 Technical phase

Given a specific physical model and restrictions that represent external results, users include simulation inputs into the system, and this information is processed to extract spatial and abstract components of the simulation as well as the relations between these components. Spatial components are visualized as three-dimensional models and abstract components are listed with their relations. Users can filter visible components and reconfigure each component. When users define a specific experiment, the system runs the experiment and results are displayed as visual representations. Users can then interpret the results and perform comparisons.

3.1.3 Dissemination phase

After a set of experiments has been conducted and new insights found, researchers define the inputs that describe the new model, parameterize them, and export the model as a tool. This tool is capable runoff running the model and visualizing the output results.

3.2 System Implementation

NemoViz was developed as a modular visualization environment system. The system was developed using HTML5, Webgl, Javascript, Python, and C++, and uses Nemo5 as a library. Users can inspect Nemo5 input decks, explore three-dimensional models, visualize simulation results, generate reproducibility tools, and export three-dimensional models to generate high-quality images. All these components were designed to support users during the entire process of designing a new simulation. The following sections will introduce the actual user interfaces and examples of how users can interact with a Nemo5 simulation.

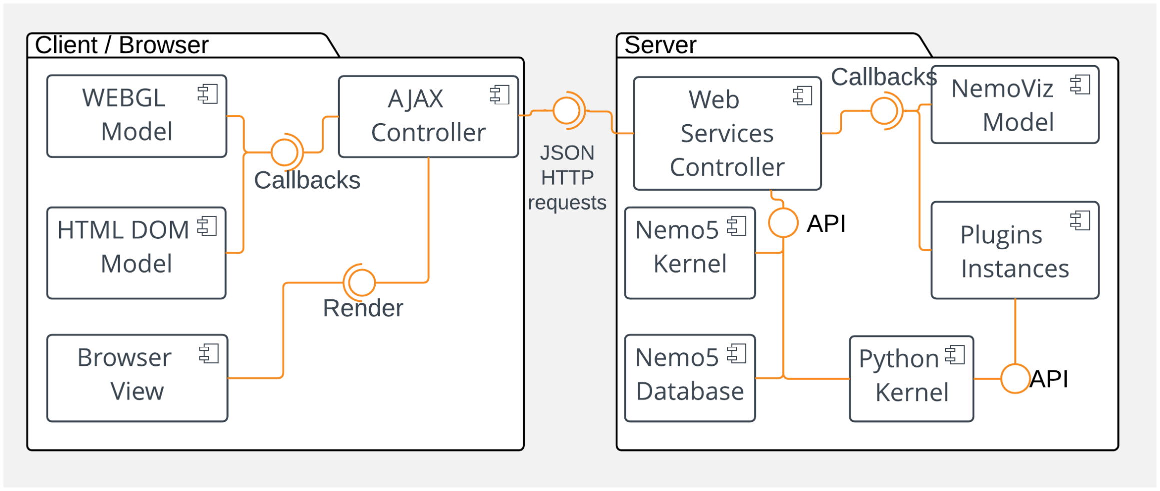


Fig. 3.1. System Architecture: NemoViz is implemented using modelviewcontroller (MVC), and client-server architectures

3.2.1 System structure

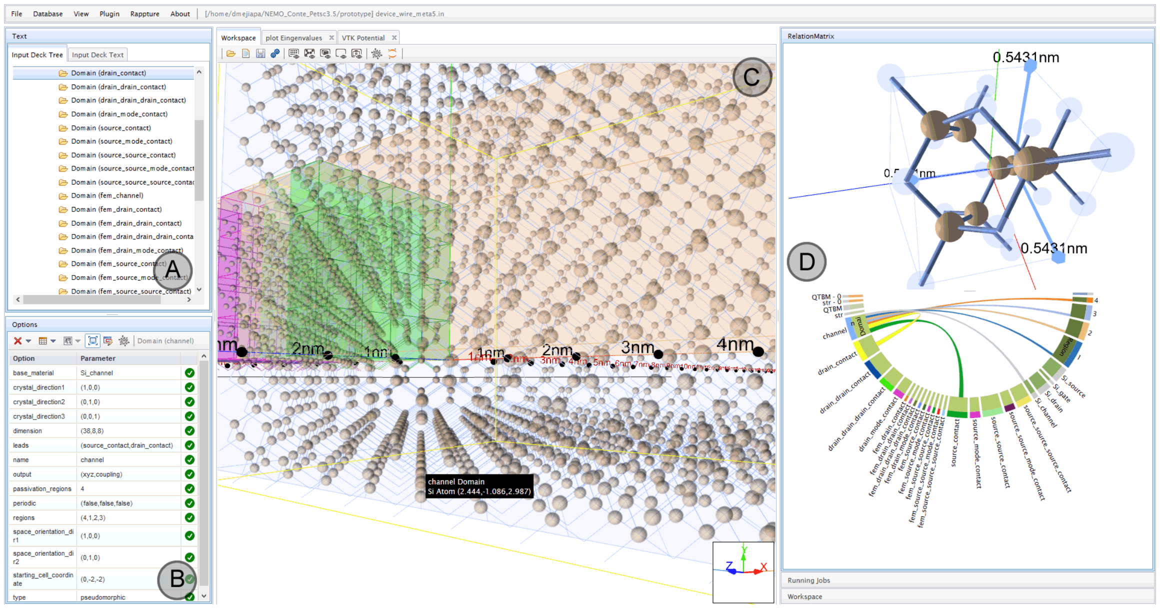
Figure 3.1 describes the architecture defined to develop NemoViz. In order to be extensible for developing new visualizations, it is based on modelviewcontroller (MVC), and client-server architectures. The client-side is implemented with JavaScript, control elements are based on the Dojo toolkit library [13], Webgl visualizations use Three.js as the core [14], and the D3 [15] library supports all other kinds of visualizations. The server-side is implemented with C++ supported by the Boost library [16], and visualization plug-ins are implemented in Python, using well known visualization libraries.

All user interactions with the client are captured as Javascript events via a web browser, and they are passed to the *asynchronous JavaScript and XML (AJAX) controller*. This controller dispatches events in the Webgl model, and triggers refreshes of views that need to be changed. The AJAX controller can also demand information from web services as JSON requests. All client requests are captured in the server by the web services (WS) controller, The WS controller is in charge of four main tasks. The first task is to call a Nemo5 kernel and execute small calculations using Nemo5's API. These calculations are triggered by the NemoViz model, sent to the WS controller and, when Nemo5's results are ready, passed back to the model. Then, the controller receives changes required by the views in the client and notifies the AJAX controller. The second task is similar to the previous task, but instead of calling Nemo5 calculations it executes Nemo5 database queries and processes information related to Nemo5 parameters. The third task is to communicate to the model any interaction that users have with visual models. For example, to hide layers of information, or include advance calculation of atoms positions. The last task is to execute Python code from a specific plug-in, which captures HTML representations of a visualization, compresses the text, and sends it back to the client by notifying the AJAX controller.

3.2.2 User Interface

NemoViz is based on Nemo5's input structure of blocks of properties. A Nemo5 simulation is described as a text file (input deck). An input deck is written in a C-like format (similar to a STRUCT statement). Table B contains a simple example, and it shows how an input deck is divided into groups, identified by keywords at the beginning of each curly bracket; henceforth these groups will be called *blocks*. NemoViz was designed to show multiple visualizations of *blocks* and relations found in an input deck provided as input. Figure 3.2. shows an example of an input deck loaded in NemoViz.

NemoViz layout consists of four main visual containers. These containers are synchronized accordingly to each user action, particularly enabling additional visualizations when needed (e.g., if a domain is selected, the visualization details of the crystal must be displayed). Description of the four containers are presented below.



Domain (drain_contact)
Domain (drain_drain_contact)
Domain (drain_drain_drain_contact)
Domain (drain_mode_contact)
Domain (source_contact)
Domain (source_mode_contact)
Domain (source_source_contact)
Domain (source_source_mode_contact
Domain (source_source_source_contac
Domain (fem_channel)
Domain (fem_drain_contact)
Domain (fem_drain_drain_contact)
Domain (fem_drain_drain_drain_conta
Domain (fem_drain_mode_contact)
Domain (fem_source_contact)
Domain (fem_source_mode_contact)
Domain (fem_source_source_contact)

Option	Parameter	
base_material	Si_channel	✓
crystal_direction1	(1,0,0)	✓
crystal_direction2	(0,1,0)	✓
crystal_direction3	(0,0,1)	✓
dimension	(38,8,8)	✓
leads	(source_contact,drain_contact)	✓
name	channel	✓
output	(xyz,coupling)	✓
passivation_regions	4	✓
periodic	(false,false,false)	✓
regions	(4,1,2,3)	✓
space_orientation_dir1	(1,0,0)	✓
space_orientation_dir2	(0,1,0)	✓
starting_cell_coordinate	(0,-2,-2)	✓
type	pseudomorphic	✓

channel Domain
Si Atom (2.444,-1.086,2.987)

Fig. 3.2. NemoViz server client layout consists of four main visualizations containers: (1) Outline view: A hierarchical data structure, (2) Properties view: Editable options of a selected block, (3) Main view: A three-dimensional representation of the model defined in an input deck, and (4) Relations view: An abstract representation of a selected block and its relationship with other blocks.

Outline view

The **outline** view visualization represents hierarchical data and defines a simulation. It allows viewing the input deck in two different ways: a hierarchical tree structure or a plain text editor. A hierarchical tree structure is a natural way to represent the input deck's *blocks* (see the top left figure of Figure 3.2). The user can collapse and expand a *block* by clicking on the folder icon with its name. Users can change from a tree-like view to a plain text editor by changing to ”input deck text” mode.

Properties view

The *properties* visualization represents the state of a block. Each *block* represents a set of parameters that configure part of the simulation. When a user selects a *block* in the tree view or on any other visualization, the table on the *properties* view appears and is filled with the name-value pairs that are defined in that *block* (see the bottom left of Figure 3.2). Users can edit values in the table, and automatically see the changes visualized in the other visualization containers, particularly in the *main* view and the *relations* view. Also, the system alerts users about errors detected in the parameters of a particular block by showing a tooltip. The *properties* view also includes a toolbar where users can hide or show the visual representations of the selected *block* and find possible parameters and documentation for the block and each parameter.

Main view

This container is a set of visualizations comprising the main visualization, called *workspace*, and instances of post-processing visualization plug-ins. Each plug-in instance is loaded in a different container (tab), and has a unique name that is assigned when the plug-in is executed the first time. Tabs can be renamed and closed. All vi-

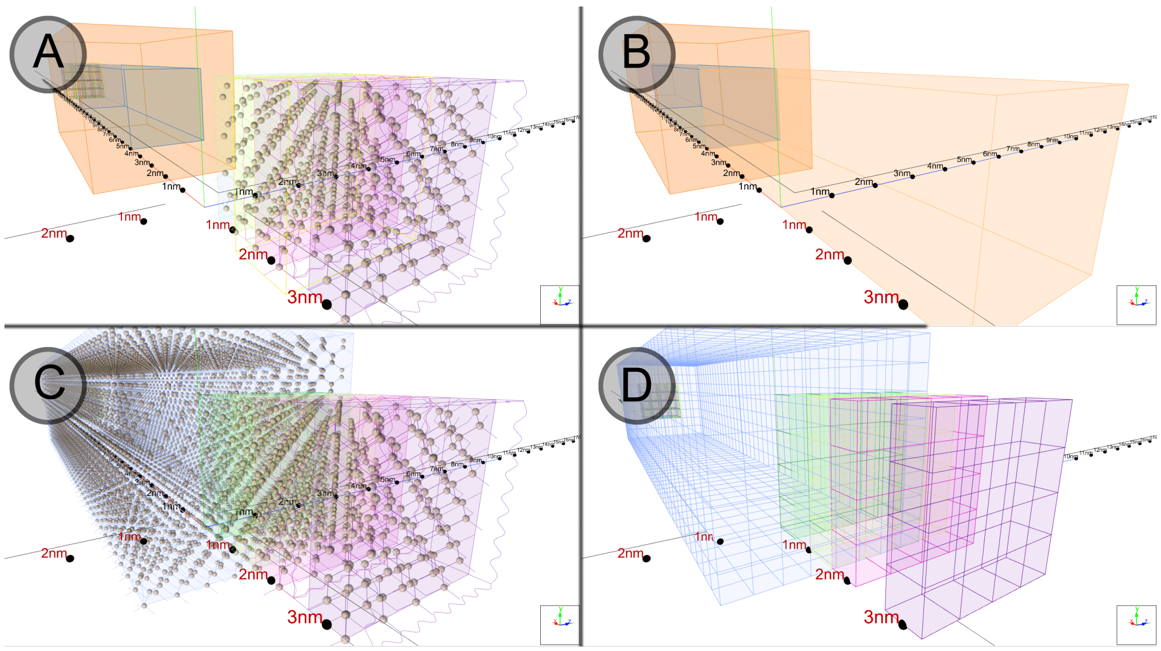


Fig. 3.3. Main view: Blocks with a spatial representation are visualized in this view and users can filter the types of models that are overlapped: A) different block types enabled for different parts of the device, B) only geometrical representations are visualized, C) only atomistic representations are visualized, and D) only meshes are visualized

sualizations can be accessed by clicking on the tab identifier name, and the *workspace* visualization cannot be closed.

TThe workspace visualization consists of three-dimensional models that represent different aspects of the simulation. Any block that contains information about a spatial representation (e.g., regions, domains, boundary conditions, finite element domains, etc.), are included, as three-dimensional models in the main view, as is depicted in Figure 3.3. All models have an opacity level to enable users to visually detect overlapped elements. Traditional zoom/pan interaction techniques are enabled so the user can visualize the models from different points of view, and information about elements in the model is visible as a tooltip when the mouse is close to that element.

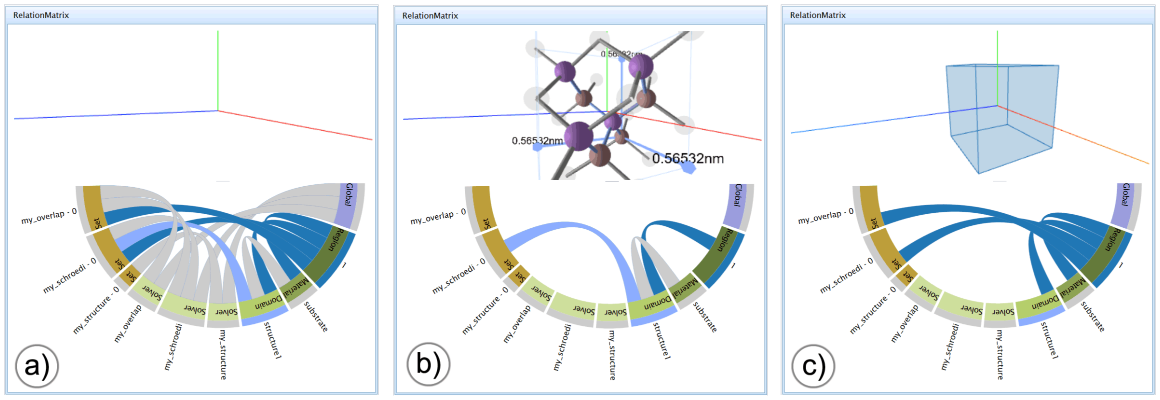


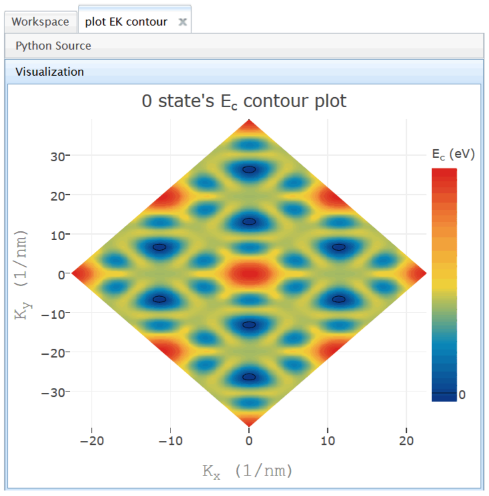
Fig. 3.4. NemoViz representations of an example of a Gallium arsenide (GaAs) cuboid: a) Relations view showing all identified relations between blocks. B) Relations view showing relations of a selected domain. c) Relations view showing relations of a selected region

Relations view

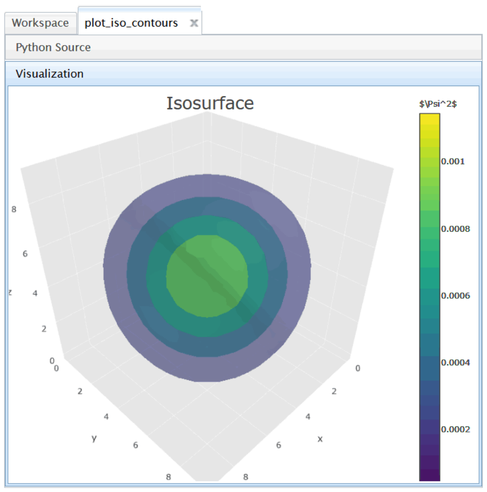
The *relations* visualization represents a block of the simulation and its relations. This visualization is a mix between a bottom quadrant chord diagram, as shown in Figure 3.4, and a three-dimensional canvas. The chord diagram is represented only in the bottom quadrant, and it shows all detected relations between blocks in an input deck, but it also highlights connections from a particular block. The *relations* view also represents blocks as geometrical models; for instance, as a mesh or as an atomistic structure and its material information. The user is allowed to navigate different blocks in the inputdeck by clicking on the block's name in the chord diagram, and all other visualizations are synchronized accordingly.

3.2.3 NemoViz Plug-ins

Plug-ins are post-processing visualization scripts that not only represent output data from a Nemo5 simulation (e.g., I-V characteristics (current vs. voltage plot), error convergence plots, and spatially resolved density of states), but also external



(a) GaAs simulation: density of states mapping the Brillouin zone are represented as a heat map.



(b) Quantum-dot simulation: eigenfunctions of the ground state are represented as 3D contours.

Fig. 3.5. NemoViz Plugins Plotly

data such as: simulation memory consumption, tic-toc traces and object lifetimes. Plug-ins are classified by type; each type represents a wrapper around specific Python libraries. Currently, NemoViz supports five plug-in types based on popular visualization libraries: Bokeh, Plotly (Figure 3.5(a)), Paraview, X3Dom and HTML/D3. Each plug-in can be parameterized and configured, meta-variables allow users to change parameters directly from the NemoViz client and execute the same script with different inputs. Visualizations can go from simple line plots or histograms to heat maps and three-dimensional surfaces. They can also include different layers of information like heat maps and contour visualizations.

3.2.4 Dissemination Tools

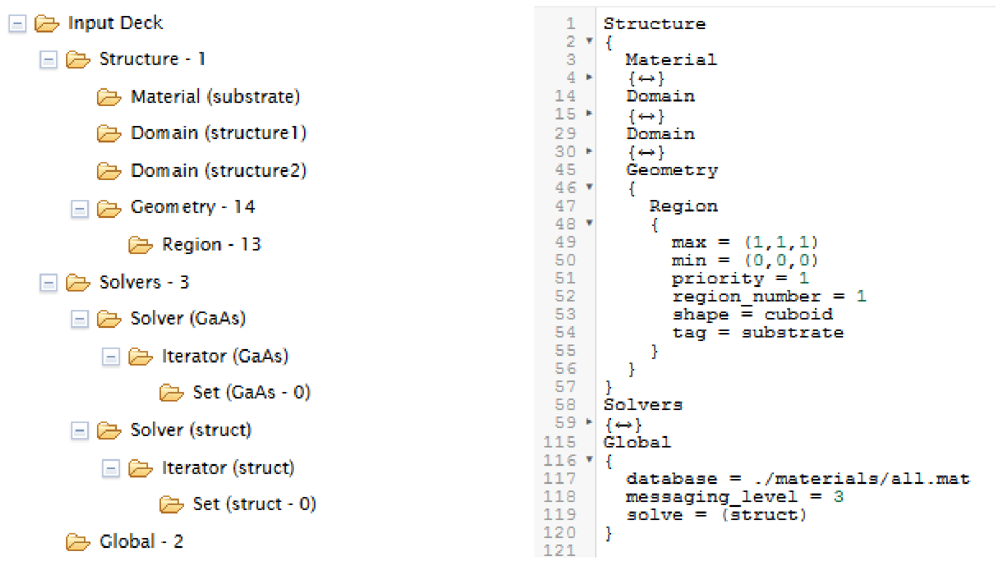
NemoViz includes two main components that can help researchers to disseminate their findings. The first component allows users to export three-dimensional models as a Threejs scene. Threejs describes all elements included in the scene as a JSON file including cameras, lights, and geometrical definitions. These files can be imported into Blender using a Blender add-on called NemoViz loader. Blender would allow researchers to create high-quality pictures of their models.

The second component is the reproducibility exporter. Researchers can define parameters in the input deck and export these parameters as a Rappture tool that can be published on nanohub.org. Rappture tools consist of an XML file that describes the graphical user interface, and a Python script. The Python script takes the parameters from the GUI, creates a valid Nemo5 input deck, and executes Nemo5. By default, it captures all the output from the simulation as a log file, but researchers can modify the Python script to include some visualization as part of the output.

The limited number of available visualizations is one of Rappture's weaknesses, and to include new visualizations would require heavy redevelopment of the Rappture infrastructure [17]. However, users have a second option with NemoViz: users can export an inputdeck as a Jupyter notebook. In this case, researchers have the option of not just predefined parameters, but the Jupyter notebook also includes instances of the visualization plug-ins. The exported Jupyter notebook translates all selected parameters as Jupyter widgets, and plug-ins codes as functions. The Jupyter notebook also includes button widgets that trigger the execution of those functions. Jupyter notebooks can be downloaded and published on nanohub.org as public tools (this option was recently added).

3.3 Defining a Simulation with Nemo5

NEMO5 is described as a text file, also called an input deck. It contains text written in a C-like format (similar to a STRUCT statement), with keywords at the beginning of each structure that define the part of the simulation each block represents; there are clear sections or definitions. A NEMO5 input deck starts with the Structure section, which contains information about structure and materials. The information about materials, atomic composition, and nonstandard material parameters is contained under the Materials nested definition. The Geometry definition specifies the geometric shape of individual regions, and Domain defines which regions are aggregated to a domain. Each simulation takes place in a certain Domain, and several simulations can be carried out, possibly by coupling. The Solvers definition sets the simulation types. Each simulation, given its own solver definition , has a set of options specific to its task. The Global section defines the location of the material parameters and which of the defined solvers are executed on the top level. Some general remarks on the input deck are as follows:



```
Input Deck
  Structure - 1
    Material (substrate)
    Domain (structure1)
    Domain (structure2)
    Geometry - 14
      Region - 13
  Solvers - 3
    Solver (GaAs)
      Iterator (GaAs)
        Set (GaAs - 0)
    Solver (struct)
      Iterator (struct)
        Set (struct - 0)
  Global - 2
```

(a) Tree view

```
Structure
{
  Material
  {↔}
  Domain
  {↔}
  Domain
  {↔}
  Geometry
  {
    Region
    {
      max = (1,1,1)
      min = (0,0,0)
      priority = 1
      region_number = 1
      shape = cuboid
      tag = substrate
    }
  }
}
Solvers
{↔}
Global
{
  database = ./materials/all.mat
  messaging_level = 3
  solve = (struct)
}
```

(b) Editor view

Fig. 3.6. Hierarchical Visualization of NEMO5 input deck

1. Comments are done in C++-style: // marks the remainder of a line as comment, and /*...*/ allows for multi-line comments.

2. Only ... are accepted as brackets. The opening bracket needs to be placed in the same line as the section name and the closing bracket on a separate line. There is no end line character.

3. Vectors are given as (a,b,c) or (1,2,3).

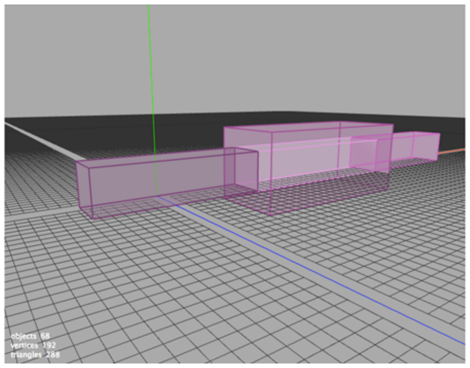
4. Vectors of vectors are given as [(a,b), (c,d)].
5. Misspelled parameters are simply ignored and do not create an error, unless the corrected parameter is mandatory in a simulation and missing from the input deck.
6. Spaces can either be spaces or tab stops.

3.3.1 Design Visualizations From Concept Stage To Design Model Stage In Nemo5

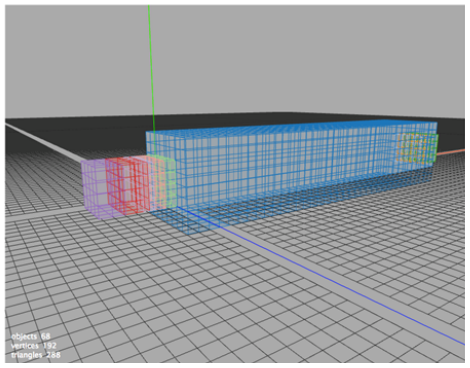
Given that there is an intrinsic hierarchy of input deck sections, a natural way to visually represent this hierarchy is a tree visualization: each node of the tree represents a section in the input deck, and each node can be expanded or collapsed to show children sections. Figure 3.6(a) shows the representation of the text from an input deck loaded into the GUI. However, there is a loss of context with this visualization, and this is the reason why some users prefer to visualize the full text. An alternative visualization shows the full text and enables visual encoding elements to fold and unfold blocks of text (see Figure 3.6(b)).

A simulation also describes elements with an intuitive three-dimensional representation. Definitions, such as finite element grids or geometrical descriptions, can be represented with three-dimensional objects, as shown in Figures 3.7(a) - 3.7(d). NEMO5s input deck also contains information, such as crystallographic definitions, that does not have a trivial representation in space. The process to understand or mentally visualize these elements is one of the most challenging cognitive processes in atomistic simulations. The cognitive process is boosted when non-trivial visualizations accompany reference visualizations as boxes or planes. The designer decides to enable or disable each layer of information to avoid occlusion.

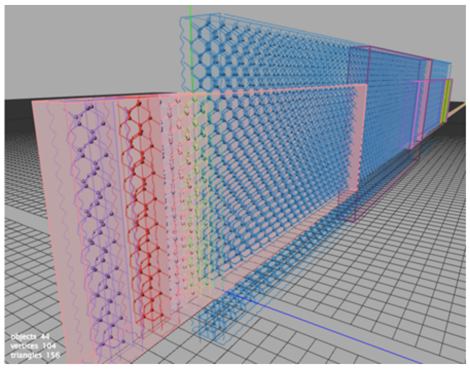
Given the complexity of crystal structure, a detailed visualization of the crystal unit cell was also designed. All atoms included in the unit cell are shown as solid spheres, with possible bonds as translucent discs and periodic atoms as translucent



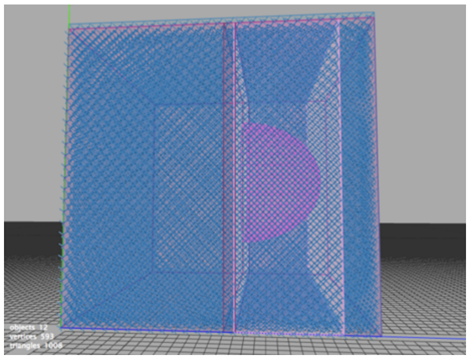
(a) Geometrical definitions of a nanowire.



(b) finite element meshes of a nanowire.

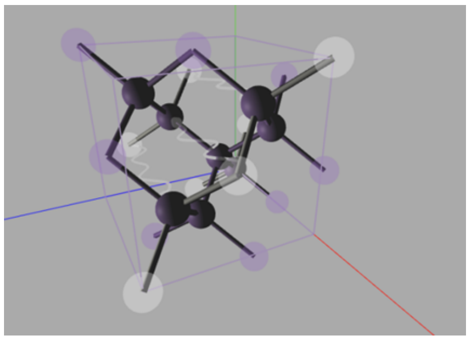


(c) Silicon ultra-thin body (UTB)

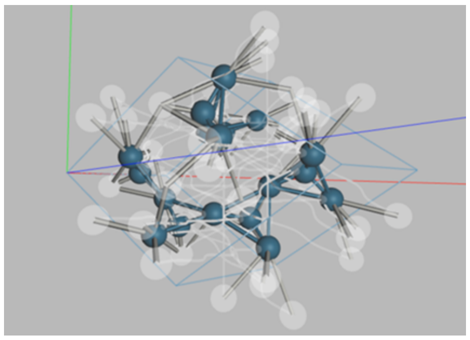


(d) Si dome-like quantum dot.

Fig. 3.7. Three-dimensional representations of visual elements included in an input deck



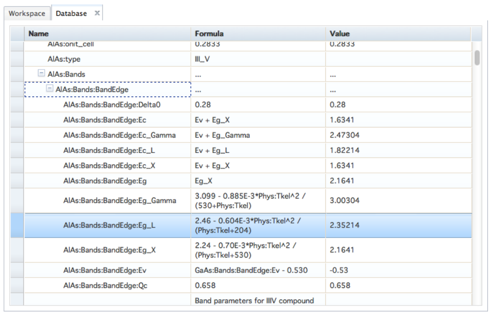
(a) GaAs crystal unit cell periodic in X direction



(b) MoS2 crystal unit cell with periodicity in all directions.

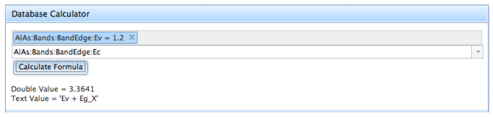
Fig. 3.8. Crystal unit cells representations

white discs . Also, curly arrows point to the periodic equivalent atom, as depicted in Figure 3.8.



Name	Formula	Value
AlAs:onit_cell	0.2833	0.2833
AlAs:type	III_V	
AlAs:Bands	...	...
AlAs:Bands:BandEdge	...	...
AlAs:Bands:BandEdge:Delta0	0.28	0.28
AlAs:Bands:BandEdge:Ec	Ev + Eg_X	1.6341
AlAs:Bands:BandEdge:Ec_Gamma	Ev + Eg_Gamma	2.47304
AlAs:Bands:BandEdge:Ec_L	Ev + Eg_L	1.82214
AlAs:Bands:BandEdge:Ec_X	Ev + Eg_X	1.6341
AlAs:Bands:BandEdge:Eg	Eg_X	2.1641
AlAs:Bands:BandEdge:Eg_Gamma	3.099 - 0.885E-3*Phys:Tkel^2 / (530+Phys:Tkel)	3.00304
AlAs:Bands:BandEdge:Eg_L	2.46 - 0.604E-3*Phys:Tkel^2 / (Phys:Tkel+204)	2.35214
AlAs:Bands:BandEdge:Eg_X	2.24 - 0.70E-3*Phys:Tkel^2 / (Phys:Tkel+530)	2.1641
AlAs:Bands:BandEdge:Ev	GaAs:Bands:BandEdge:Ev - 0.530	-0.53
AlAs:Bands:BandEdge:Qc	0.658	0.658
	Band parameters for IIIV compound	

(a) interactive table that displays parameter-defined values and the calculated values



AlAs:Bands:BandEdge:Ev = 1.2

AlAs:Bands:BandEdge:Ec

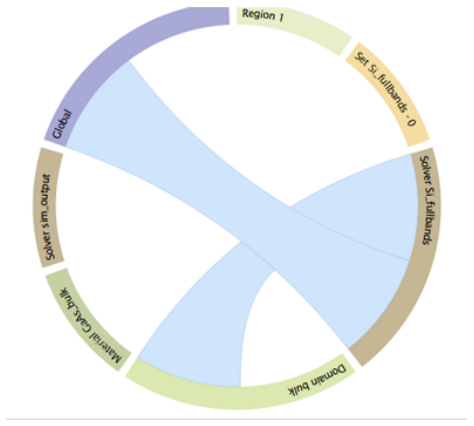
Double Value = 3.3641
Text Value = 'Ev + Eg_X'

(b) Database Calculator, that allows for parameter overloading and calculating parameter values

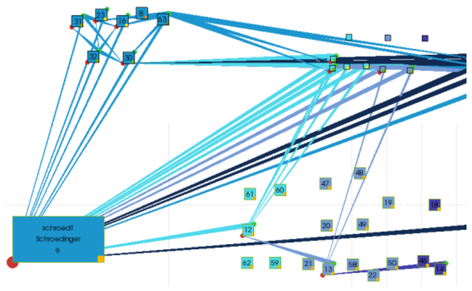
Fig. 3.9. Database browser visualization

Additionally, the designer must choose the set of parameters used by the simulation. This set is defined by a keyword, which is usually the last name of the main author. All parameter sets are defined in an additional text file in a format similar to the input deck. This file is called the *material database*, and its current size is around one megabyte. Parameters can be overloaded via the input deck or directly modified by the material database file. Parameters can be defined as values, strings, rules, or functions with other parameters as arguments. Figure 3.9 presents a table visualization designed to request data on demand.

Two visualizations were designed to represent relations between definitions. For instance, Figure 3.10(a) presents a chord diagram visualization to highlight the relations between a solver, its domain, and execution. Figure 3.10(b) presents a directed connected graph with context layout when a solver is selected.



(a) A chord diagram highlighting the relations between a solver, its domain and execution



(b) directed connected graph with context layout when a solver is selected

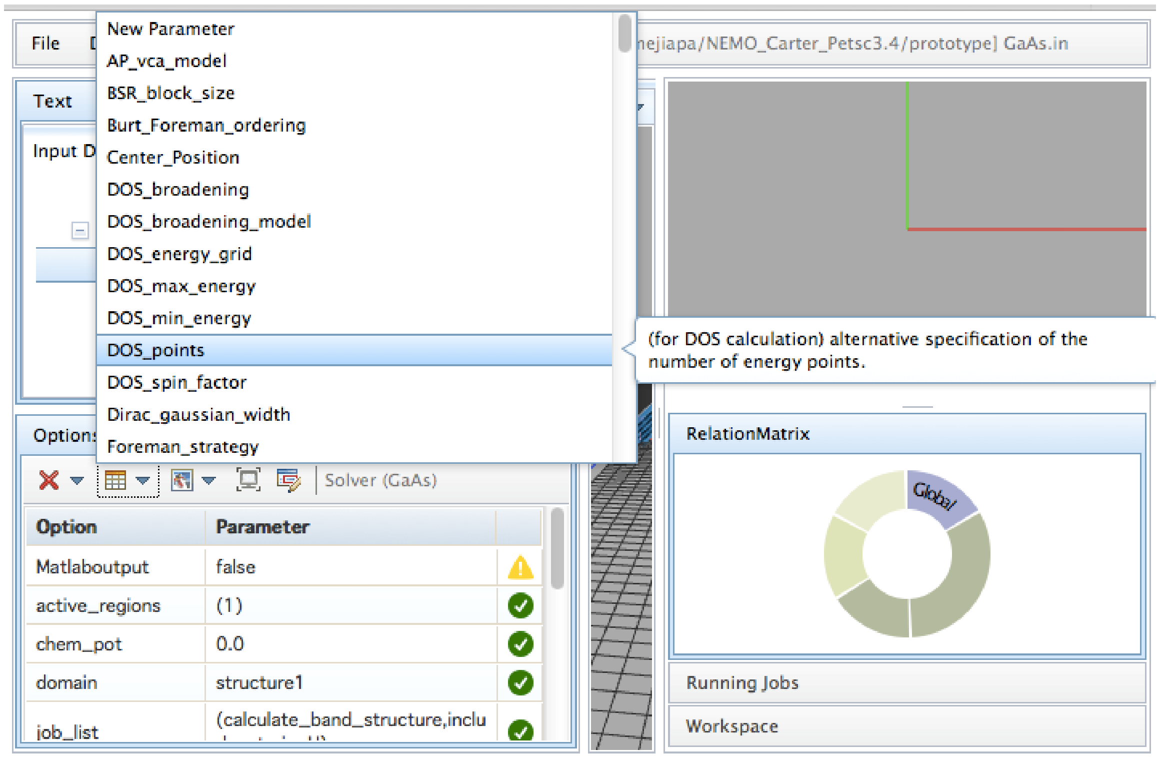
Fig. 3.10. A chord diagram examples

3.3.2 Design Visualizations From The Design Model Stage To The Develop Model Stage

NEMO5 implements multiple physical models; however, some of these models are developed using commercial libraries or require specific configurations. These models can be excluded when NEMO5 is compiled. Currently, designers do not know which models are available in a specific instance of NEMO5 based on the manual or code documentation. However, given that NEMO Server has access to the NEMO object factory, a list of solvers and their specific options are visualized, as illustrated in Figure 3.11.

If the available models do not support the designers concept, then new features must be implemented in the simulator. New features in NEMO5 could be implemented in two ways.

The first solution is to include new parameters in an existing solver, and include new function calls when these parameters are defined. Alternatively, a second solution is to create a completely new solver. Any new solver has to be registered in the NEMO5 object factory in order to be available from the input deck. In both



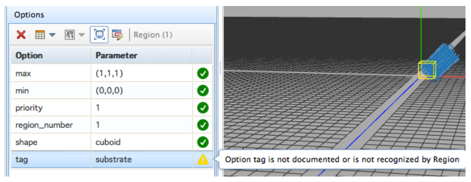
nejiapa/NEMO_Carter_Petsc3.4/prototype] GaAs.in

New Parameter
AP_vca_model
BSR_block_size
Burt_Foreman_ordering
Center_Position
DOS_broadening
DOS_broadening_model
DOS_energy_grid
DOS_max_energy
DOS_min_energy
DOS_points
DOS_spin_factor
Dirac_gaussian_width
Foreman_strategy

(for DOS calculation) alternative specification of the number of energy points.

Option	Parameter	
Matlaboutput	false	⚠
active_regions	(1)	✓
chem_pot	0.0	✓
domain	structure1	✓
job_list	(calculate_band_structure,inclu	✓

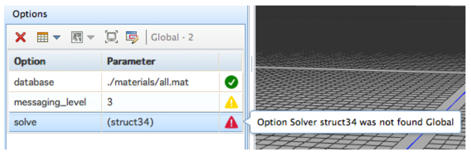
Fig. 3.11. Schroedinger solver visualization options and the description of its parameter DOS_points.



Option	Parameter	
max	(1,1,1)	✓
min	(0,0,0)	✓
priority	1	✓
region_number	1	✓
shape	cuboid	✓
tag	substrate	⚠

Option tag is not documented or is not recognized by Region

(a) Visualization when there is empty documentation



Option	Parameter	
database	./materials/all.mat	✓
messaging_level	3	⚠
solve	(struct34)	⚠

Option Solver struct34 was not found Global

(b) Visualization when there is missing documentation

Fig. 3.12. Properties visualizations and error handling

cases, NEMO5 forces the designer to document all new options and the description of the new functionality. Validation of this documentation is represented as visual aids, depicted in Figure 3.17. Color encoding is used to show the missing documentation. NEMO5 can be configured to not run if the documentation is incomplete. This

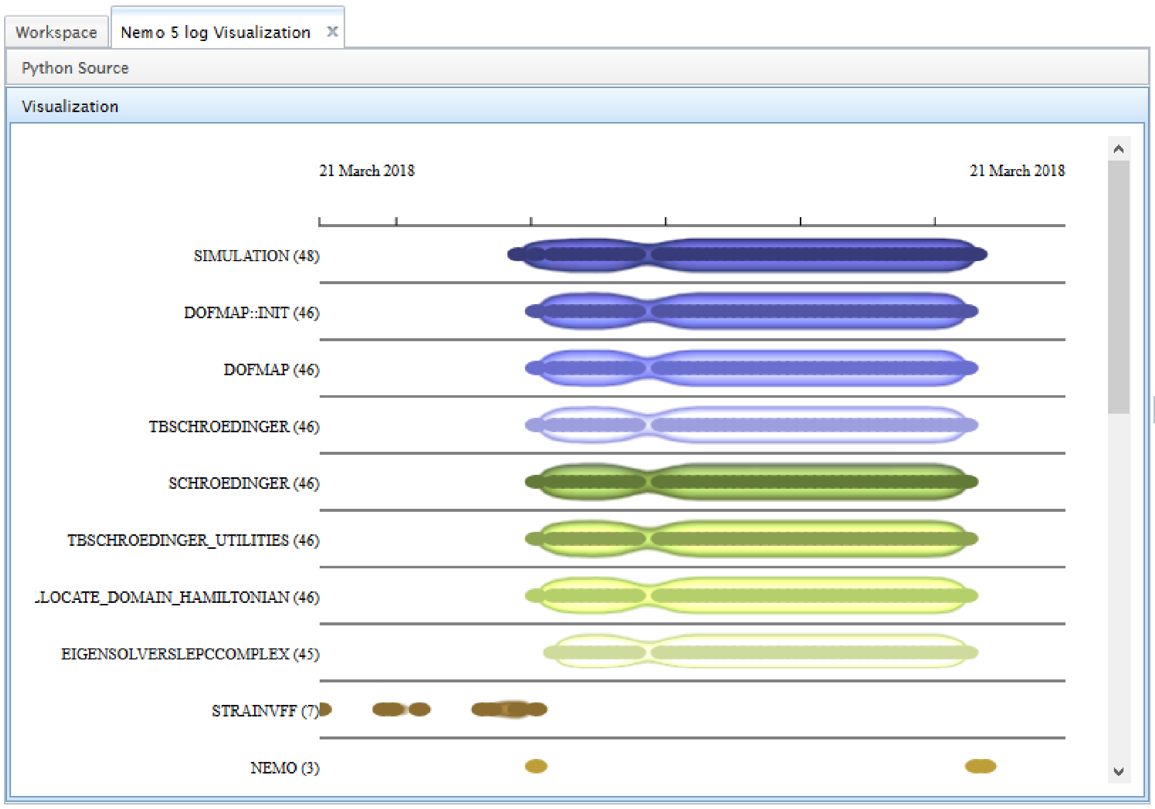


Fig. 3.13. Visualization of NEMO5 logs

visualization is supported by the fact that these solvers, and all NEMO5 objects in general, implement two interfaces: a documentation interface and a factorizable interface. Both interfaces are responsible for enforcing documentation in newly developed components and allowing tracing of logs.

When a simulation is running, the designer does not have any feedback on the status of the simulation. The only information that developers have is the log file. Log files are usually long text files with the simulators internal status messages. NEMO5 generates log files with different levels of detail. Browsing log files requires significant experience with NEMO5, and only some specific keywords can guide the search. Interactive visualizations can be used to guide this search or to display the

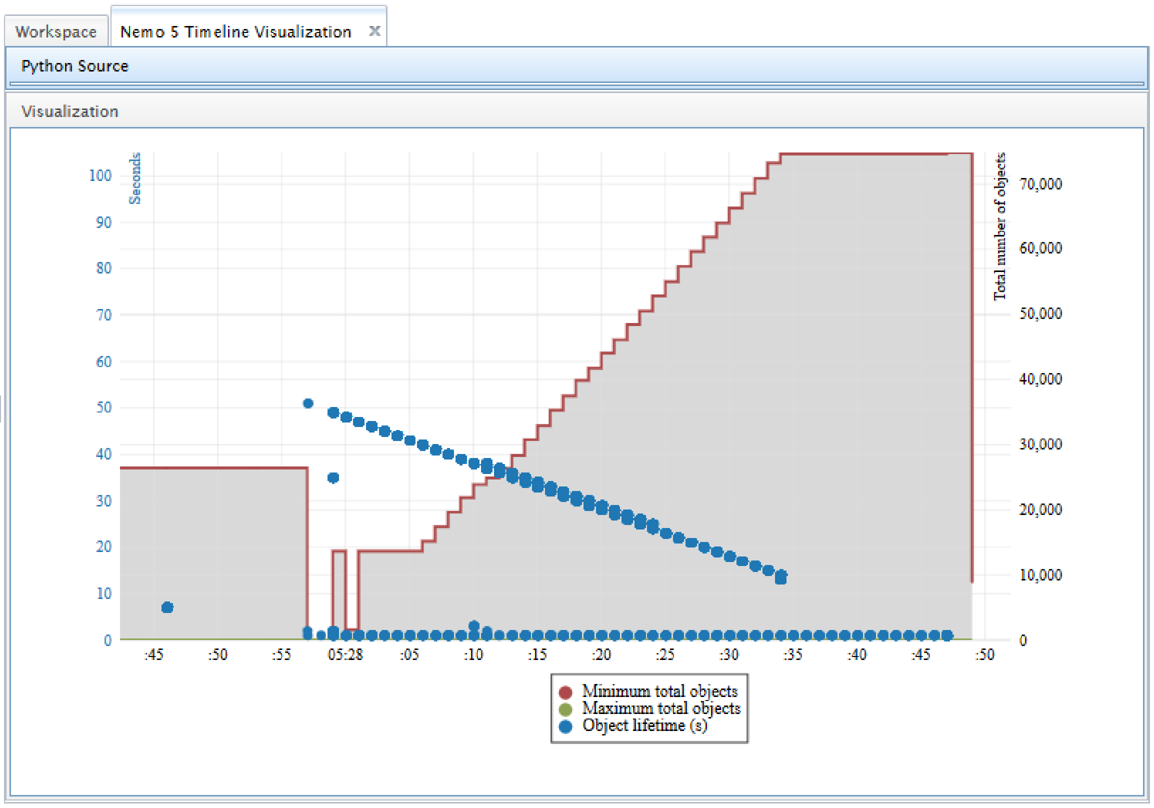
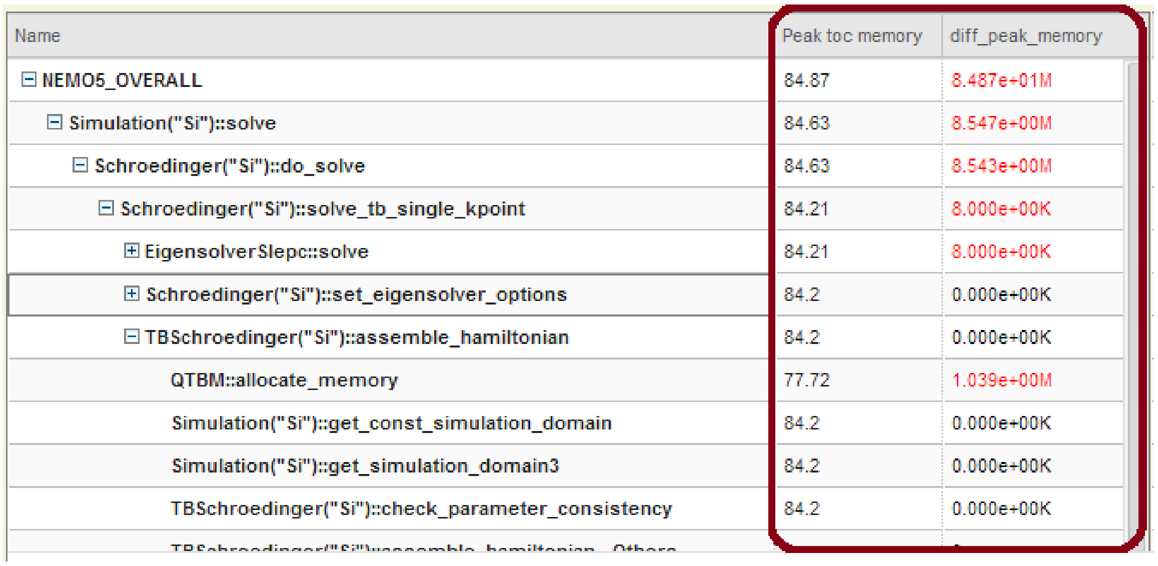


Fig. 3.14. Visualization of the time-life of NEMO5 objects

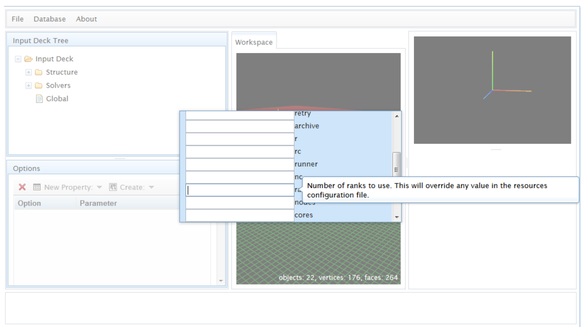
simulation status. Nemo5 logs can be visualized on NemoViz plug-ins, as illustrated in Figures 3.13 and 3.14

NEMO5 also includes an embedded tic-toc system that allows designers and developers to benchmark and profile new codes. The profiling information is dumped into an XML file that can be visualized as an interactive tree view, as illustrated in Figure 3.15. This visualization was implemented with help of Santiago Perez.



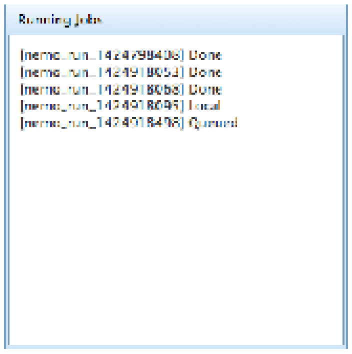
Name	Peak toc memory	diff_peak_memory
⊟ NEMO5_OVERALL	84.87	8.487e+01M
⊟ Simulation("Si")::solve	84.63	8.547e+00M
⊟ Schroedinger("Si")::do_solve	84.63	8.543e+00M
⊟ Schroedinger("Si")::solve_tb_single_kpoint	84.21	8.000e+00K
⊞ EigensolverSlepc::solve	84.21	8.000e+00K
⊞ Schroedinger("Si")::set_eigensolver_options	84.2	0.000e+00K
⊟ TBSchroedinger("Si")::assemble_hamiltonian	84.2	0.000e+00K
QTBM::allocate_memory	77.72	1.039e+00M
Simulation("Si")::get_const_simulation_domain	84.2	0.000e+00K
Simulation("Si")::get_simulation_domain3	84.2	0.000e+00K
TBSchroedinger("Si")::check_parameter_consistency	84.2	0.000e+00K
TBSchroedinger("Si")::assemble_hamiltonian - Others		

Fig. 3.15. Visualization of NEMO5 Tic-Toc output using the profiling view.



Number of ranks to use. This will override any value in the resources configuration file.

(a) Visualization of the command-line tool NEMO Launcher



[nemo_run_1424798408] Done
[nemo_run_1424918053] Done
[nemo_run_1424918068] Done
[nemo_run_1424918695] Local
[nemo_run_1424918498] Queued

(b) simulation execution queue

Fig. 3.16. Nemo5 Launcher visualization interface

3.3.3 Design Visualizations From The Development/Design Model Stage To The Run Model Stage

After the concept is completely translated into input deck language, NEMO can be executed via NEMO Launcher. The launcher tool configures NEMO5 to use specific hardware resources, such as accelerator cards, or to define restriction in the MPI execution. Nemo Launcher is a main component of the NEMO5 regression test system. However, the launcher is a command-line utility that is not part of the NEMO binary. A visualization that extracts the main components of the launcher and represents them as simulation parameters was designed and is depicted in Figure 3.16(a). All simulations executed via the launcher can be visualized as a list Figure 3.16(b)

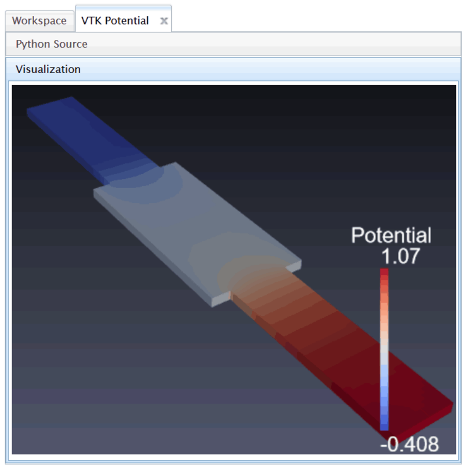
3.3.4 Design Visualizations From The Run Model Stage To The Compare Stage

After a successful simulation is executed, NEMO5 generates multiple text files that describe the physical model observables; however, these files come in different formats. Additionally, some of the results from NEMO5 might not be comparable to the experimental data results. Therefore, data may need to be transformed and unified. Simulation designers create scripts to transform the data and perform comparisons. Tools such as Matlab, PyLab, and R are widely used for data transformations as well as simple visualizations. Tools such as Paraview or Visit are used to visualize three-dimensional data.

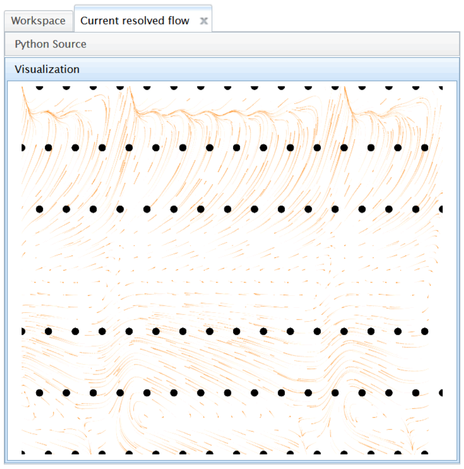
To support scripting flexibility, Nemo Server includes Nemo Server plug-ins. All scripts written in Python can be loaded as NEMO Server plug-ins and visualized at any time. Figure 3.5 presents examples of visualizations created with a Plotly plug-in, which allows designers to generate two-dimensional visualizations. Figure 3.17(a) presents visualizations created with a Paraview plugin. In fact, any Paraview state can be exported as a Python script and loaded directly as a plug-in.

3.3.5 Design Visualizations From The Run Model Stage To The Analysis Stage

After validation is performed in the compare stage, a designer enters the exploration phase. The designer seeks better insight into the model by changing some of

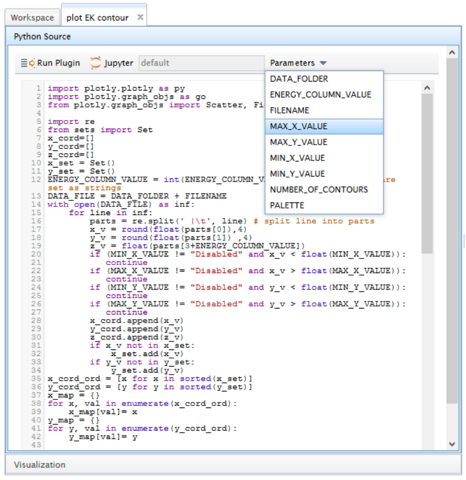


(a) Silicon Ultra-thin-body UTB simulation, final self-consistent potential represented as a mesh surface



(b) Transport simulation on MoS2 sheets, streamlines represent the current flow between atoms

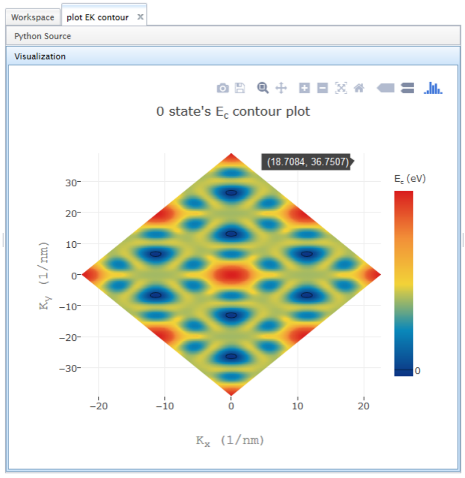
Fig. 3.17. Plug-ins visualizations



```
import plotly.plotly as py
import plotly.graph_objs as go
from plotly.graph_objs import Scatter, Fi

import re
from sets import Set
x_cord=[]
y_cord=[]
z_cord=[]
x_set = Set()
y_set = Set()
ENERGY_COLUMN_VALUE = int(ENERGY_COLUMN_V                      re
set as strings
DATA_FILE = DATA_FOLDER + FILENAME
with open(DATA_FILE) as inf:
    for line in inf:
        parts = re.split(' |\t', line) # split line into parts
        x_v = round(float(parts[0]),4)
        y_v = round(float(parts[1]) ,4)
        z_v = float(parts[3+ENERGY_COLUMN_VALUE])
        if (MIN_X_VALUE != "Disabled" and x_v < float(MIN_X_VALUE)):
           continue
        if (MAX_X_VALUE != "Disabled" and x_v > float(MAX_X_VALUE)):
           continue
        if (MIN_Y_VALUE != "Disabled" and y_v < float(MIN_Y_VALUE)):
           continue
        if (MAX_Y_VALUE != "Disabled" and y_v > float(MAX_Y_VALUE)):
           continue
        x_cord.append(x_v)
        y_cord.append(y_v)
        z_cord.append(z_v)
        if x_v not in x_set:
            x_set.add(x_v)
        if y_v not in y_set:
            y_set.add(y_v)
x_cord_ord = [x for x in sorted(x_set)]
y_cord_ord = [y for y in sorted(y_set)]
x_map = {}
for x, val in enumerate(x_cord_ord):
    x_map[val]= x
y_map = {}
for y, val in enumerate(y_cord_ord):
    y_map[val]= y
```

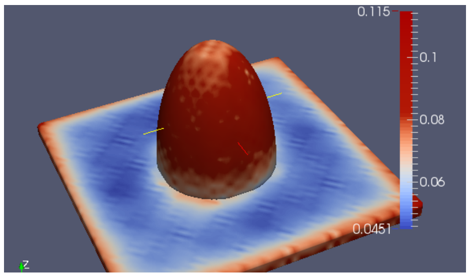
(a) Parameterization of an energy-momentum heatmap plug-in



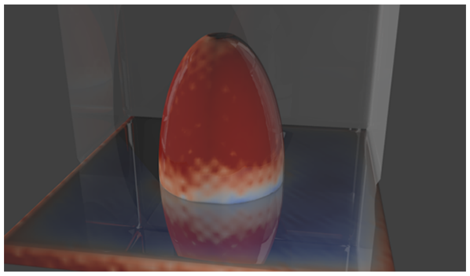
(b) Energy-momentum heatmap visualization

Fig. 3.18. Plug-in Parameterization

the parameters and running multiple simulation scenarios. Analysis is performed with new data results. The designer tries to find patterns that explain how the model addresses the research question defined in the conceptual stage. This cognitive process shares similar characteristics to the cognitive process in the compare stage. Visualizations described in the previous section also apply to this stage. However, given the different data result sets, the designer should parameterize any visualization plug-in. NemoViz allows plug-in to be parameterized as a Json file that can be uploaded to the server (see Figure 3.18).

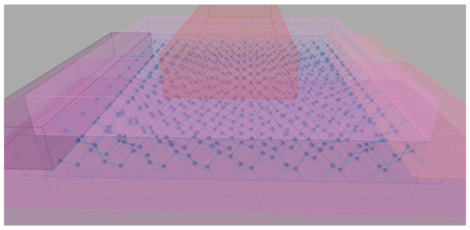


(a) Model exported from Paraview

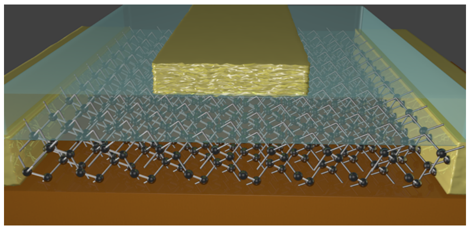


(b) Render obtained after post-processing on Blender

Fig. 3.19. Magnitude of the strain forces on the surface of a quantum dot's core



(a) Visualization of the main view in NemoViz



(b) Render obtained after post-processing on Blender

Fig. 3.20. Two-dimensional MoS2 sheet input deck representation

3.3.6 Design Visualizations From The Analysis Stage To Publication Stage

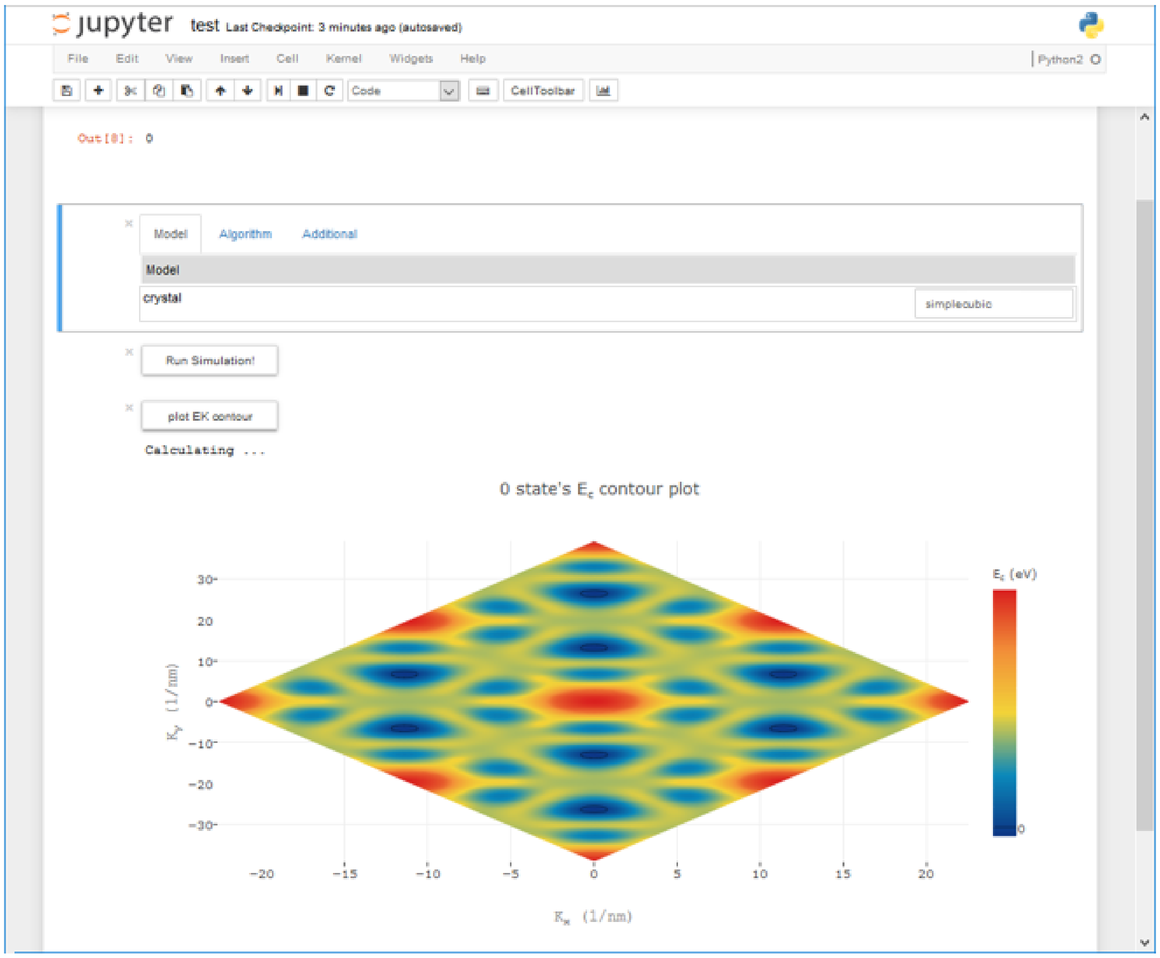
Results obtained during the compare and analysis stages can be used to create high-impact images that document and enrich the designers work. Blender is an open source tool that creates professional, three-dimensional models that can enhance the visualization generated with a Paraview plug-in or from the input deck visualization. NemoViz allows the export of Paraview plug-ins to Blender to create high-impact images Figure 3.19(a) presents the visualization of the Paraview plug-in of a quantum dot and Figure 3.19(b) presents the visualization of the model exported as VRML, imported and rendered with Blender.

NemoViz also includes another option to create high impact-images. The input deck visualization, instead of the plug-in, can be directly exported to Blender. Figure 3.20(a) depicts the visualization of the input deck in the Main view, and Figure 3.20(b) the result after applying material properties and textures in Blender to this model.

NemoViz dissemination tools help users to generate visualizations that support findings in the data. The next chapter describes some of the images generated from simulation results; they highlight important information and convey data from the simulation results.

3.3.7 Visualizations Design From The Publication Stage To Replication Stage

Experiment replication is one of the most important aspects of science, and it is included in the definition of the scientific method. If simulations are considered a scientific research method, replication must be included as part of the process. One way to replicate simulation is to publish tools on Nanohub. There are examples of how Nanohub tools successfully replicate data published in papers [18]. Since NEMO5 is already part of the Nanohub pool of libraries, designers could easily create a tool based on an input deck. NemoViz enables users to select experimentation parameters and generate a template Rappture tool or Python notebook based on the inputdeck. Plug-ins that generate example visualizations shown in this chapter can be included as part of the notebook (see Figure 3.21).



```
Out[8]: 0
```

Model

crystal: simplecubic

```
Calculating ...
```

Fig. 3.21. Jupyter notebook generated including parameterized plug-ins

CHAPTER 4. HIGH-QUALITY IMAGES

This chapter presents some of the visualizations generated with the help of NemoViz dissemination tools. All images were rendered using Blender software. The images shown in this chapter have been published supporting different research projects, and they highlight important information obtained from simulation research projects and convey data from final results.

4.1 Atomistic Structures

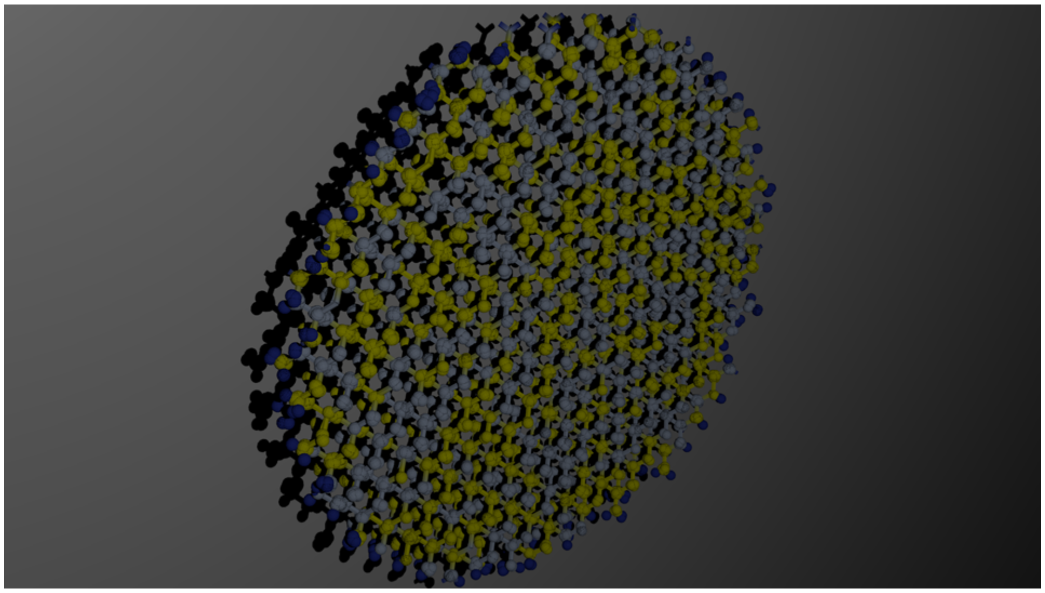


Fig. 4.1. Atoms in a Silicon-Germanium Disk

Atomistic simulations are based on the interaction between atoms. This interaction is heavily defined by atoms' positions and types. Visualizations of the atomistic structure being simulated help an audience to understand details of the final results.

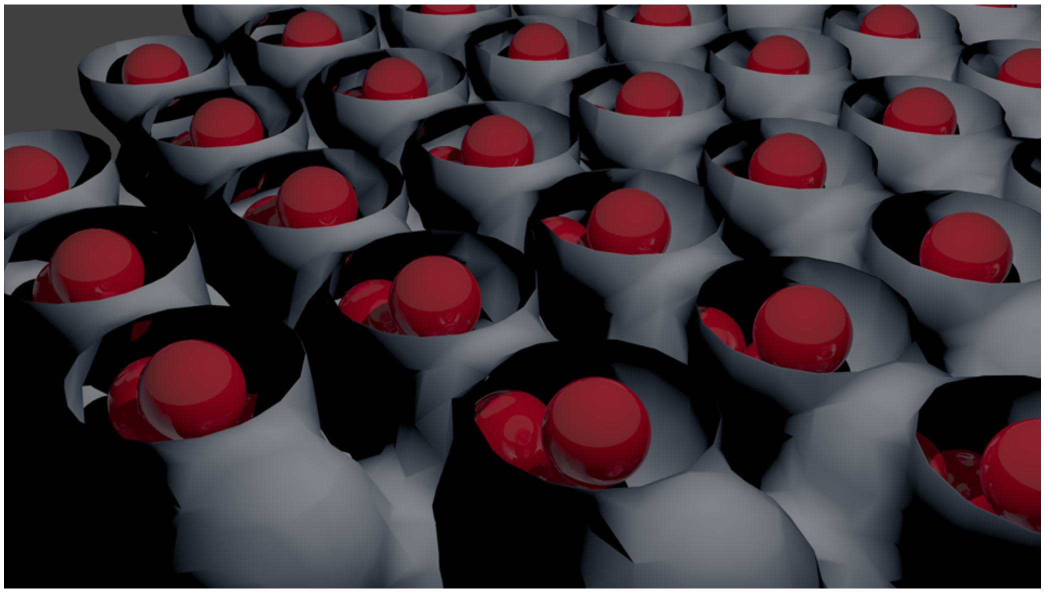


Fig. 4.2. Silicon atoms distributed in a diamond lattice

Some of the following figures highlight structure defects such as roughness, corrugation, and atomic disorder. For example, Figure 4.1 presents a representation of a disk of silicon-germanium (SiGe) after a strain model was applied to relax its atoms positions. In this picture, colors represent different atom types: hydrogen (blue), silicon (yellow) and germanium (light blue). Figure 4.2 represents the organization of the atoms in a pure material. Each atom (in red) is surrounded by a gray shell, representing the interaction field of each atom. Figure 4.5 shows multiple water (H2O) molecules between two graphene sheets (carbon atoms) as the result of a relaxation process in a molecular dynamics simulation.

Figures 4.3 and 4.4 represent the atomic-resolved SI-Ge alloy ultra-thin-body device with surface roughness. White spheres represent silicon atoms and green spheres germanium atoms. Shells' colors indicate source, drain, and channel regions. Green: source (doped region); red: channel; orange: drain (doped region); and gray/black: Oxide. Figures 4.6 and 4.7 show periodic surface roughness in a sheet of graphene.

In addition to the atomistic structure, some pictures include simulation results on top of the atomistic structure. Spatially resolved data such as density of states,

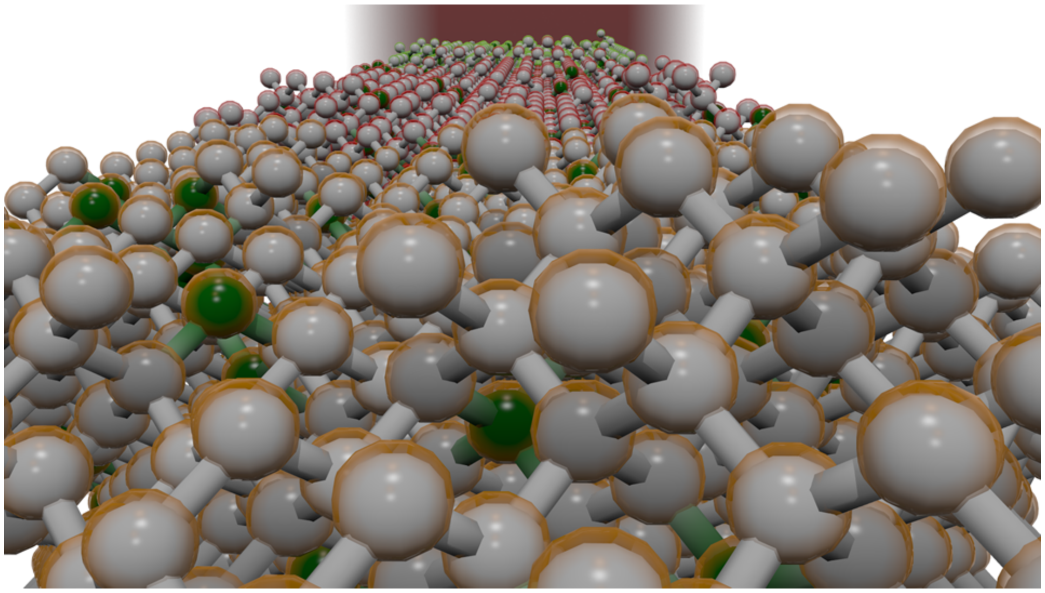


Fig. 4.3. Alloy structure of silicon and germanium atoms

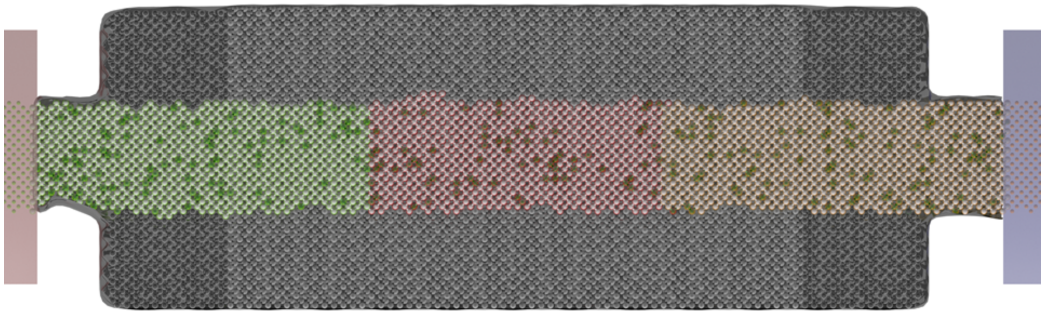


Fig. 4.4. silicon germanium atoms in an ultra-thin-body device

wave-functions, or electric potential energy applied to a simulation can be visualized as volumetric data. For example, Figure 4.8 compares the wave-functions intensity of two different energy level over a graphene sheet with a hole in the middle. Figure

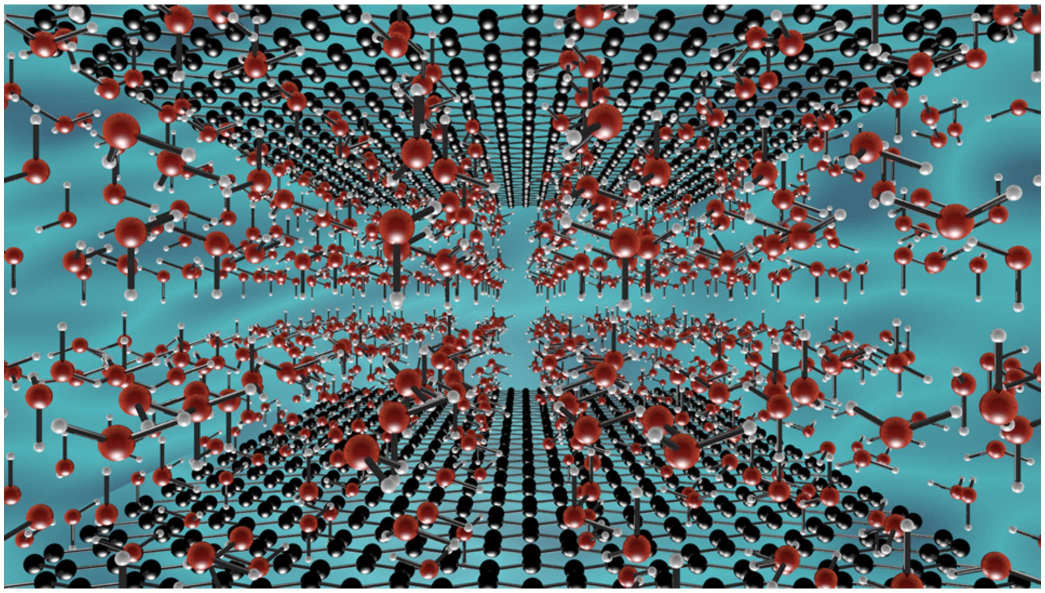


Fig. 4.5. Multiple Water (H20) molecules between two graphene sheets

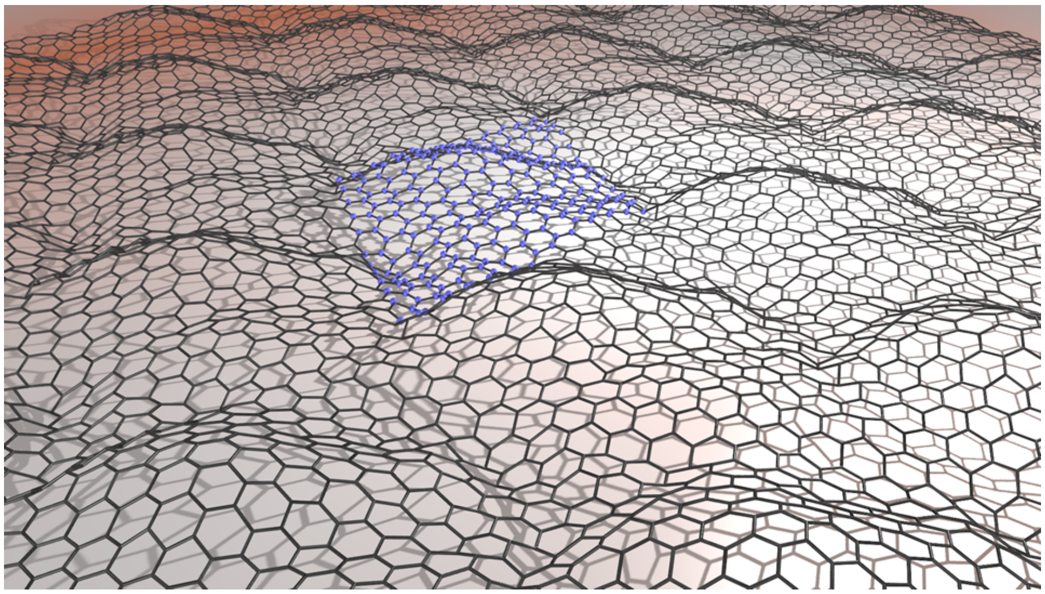


Fig. 4.6. Periodic surface Roughness

4.9 depicts the organization of the atoms in an ultra-thin body transistor and the amount of electric potential along the transistor.

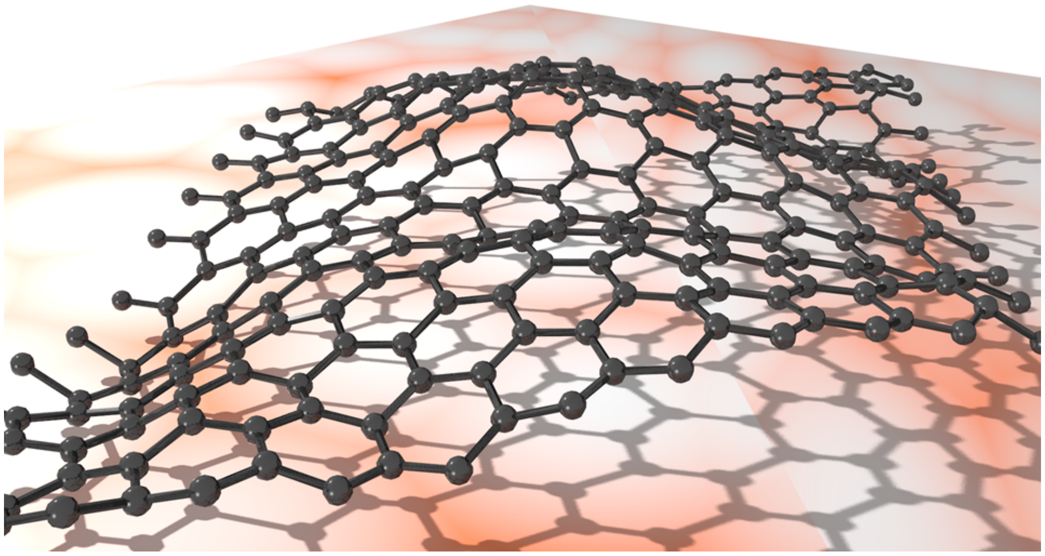


Fig. 4.7. Basic unit cell used to simulate periodic surface roughness

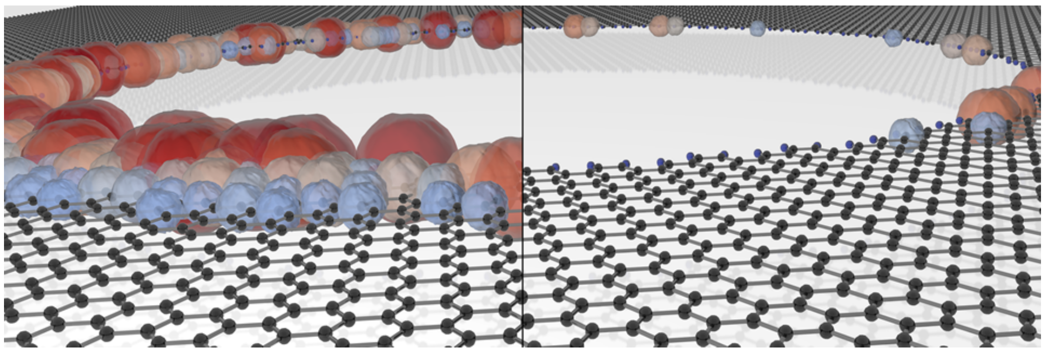


Fig. 4.8. Intensity of two energy level over a graphene sheet with a hole

4.2 Devices

In addition to the atomistic structure, some pictures describe well-known devices in the semiconductor industry, transistors in particular. In these cases, images not only represent devices from an atomistic point of view, but also contextualize the data. For example, Figure 4.10 represents an internal composition of an ultra-thin body transistor. Figure 4.11 includes the position of the atoms and their chemical

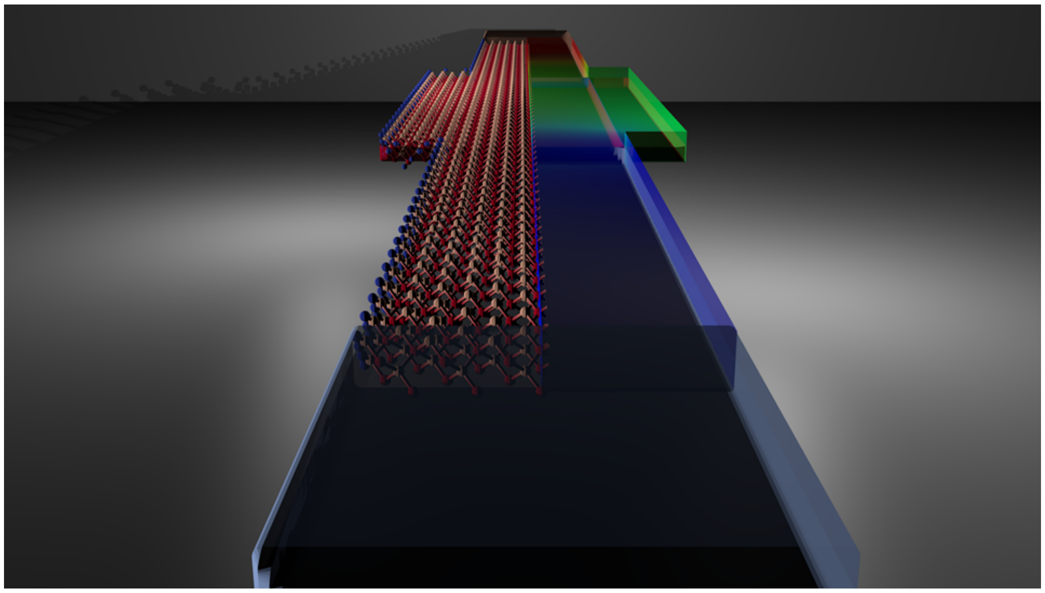


Fig. 4.9. Electric potential along an ultra-thin body transistor

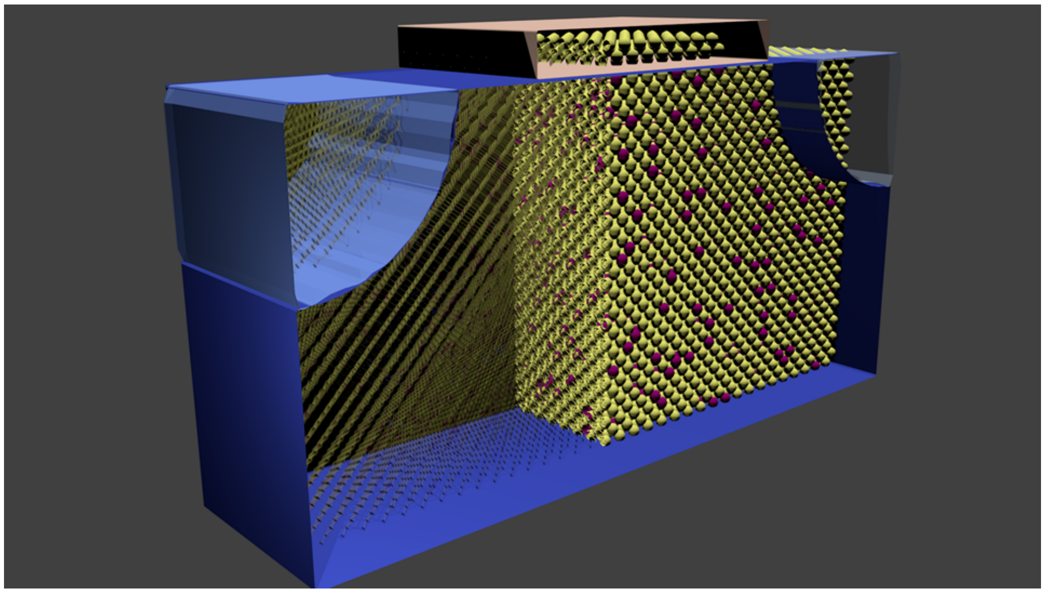


Fig. 4.10. Silicon and Germanium Alloy ultra-thin body transistor

bonds along with the amount of electric potential in the transistor. Fig 4.11 represents a cylindrical gallium arsenide nanowire showing the surfaces and the atoms.

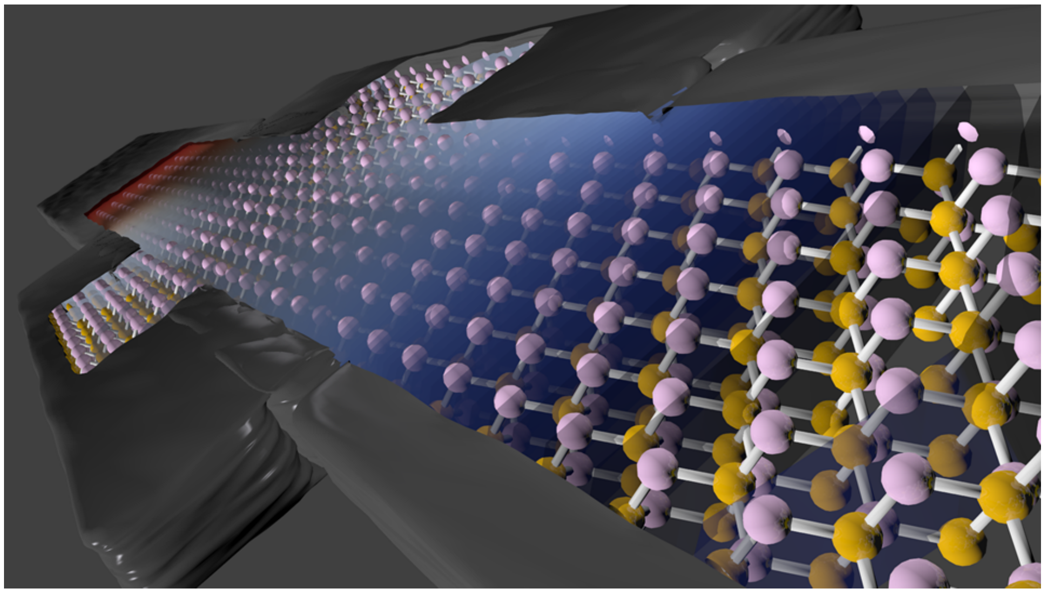


Fig. 4.11. Gallium Arsenide ultra-thin body transistor.

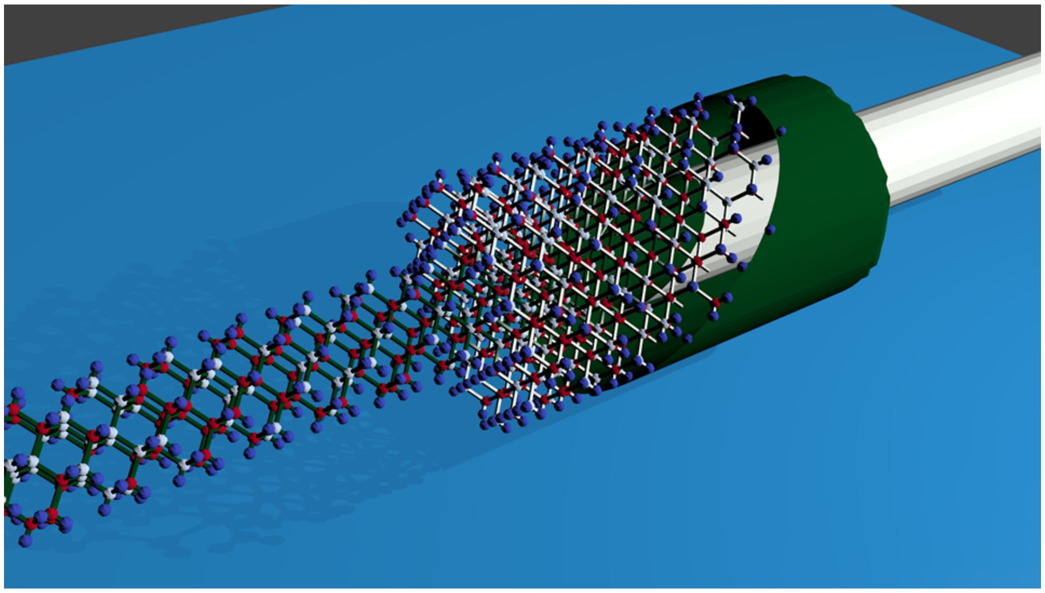


Fig. 4.12. Cylindrical Gallium Arsenide Nanowire transistor

In Addition to transistors, quantum dots are devices of great interest in the semiconductor industry. Quantum dots are nanoscale particles that behave similarly to an atom but can be created artificially. A quantum dot's core can have different shapes and composition. Both factors affect the energy levels inside a quantum dot. Fig 4.13 portrays the intensity of an energy level inside a quantum dot with a conic core (red). Fig 4.14 illustrate the directions of the strain forces on the surface of the quantum dot core. Fig 4.15 depicts the magnitude of the strain forces on the surface of the quantum dot core.

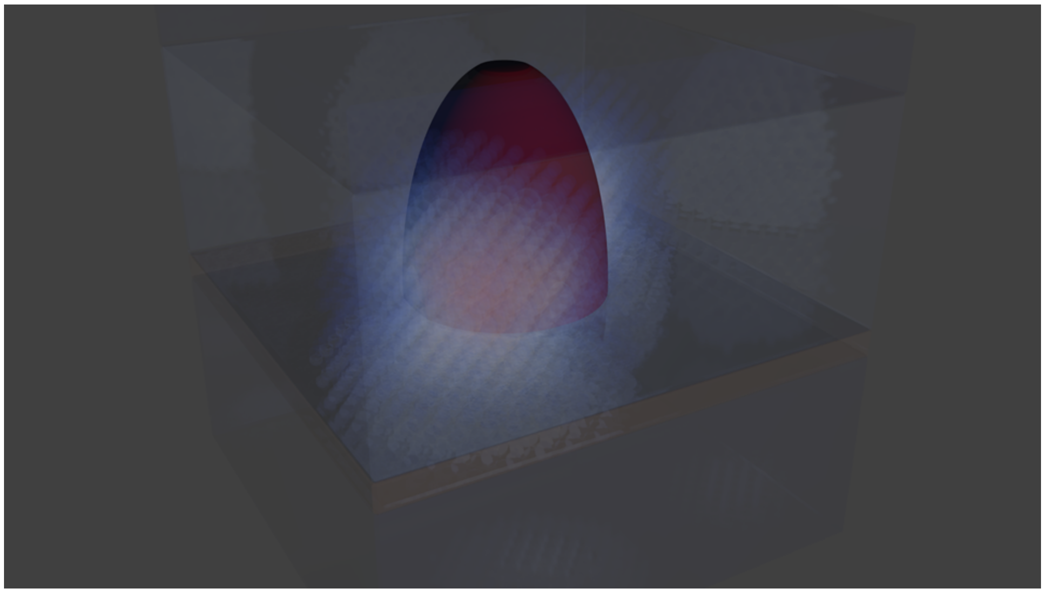


Fig. 4.13. Intensity of an energy level inside a quantum dot with a conic core

More experimental devices such as flying qubits were also simulated with Nemo5. Their structure was exported with NemoViz and rendered on Blender. Fig 4.16 shows multiple flying qubits, where their superposition is controlled by gates.

4.3 Images Dissemination

As mentioned before, all the previously mentioned figures supported research findings and were published along with the results. Figures 4.3 and 4.4 were published in [19] and presented at the Blue Waters Symposium [20]. Figures 4.13, 4.10, 4.15, 4.9, 4.14 and 4.11 were presented at the Blue Waters Symposium as well [21], [22].

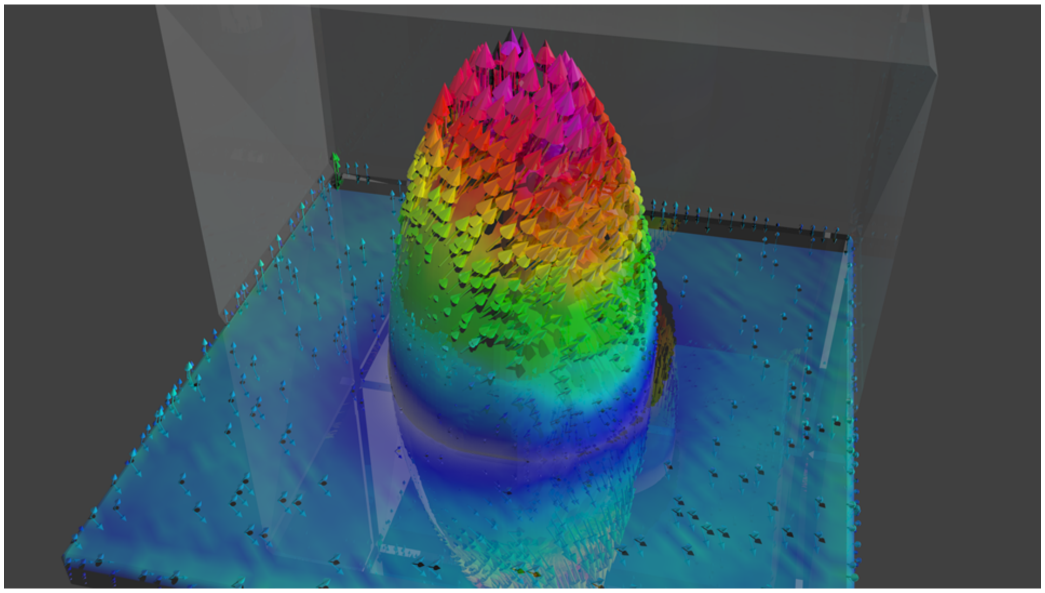


Fig. 4.14. Direction of strain forces on a quantum dot

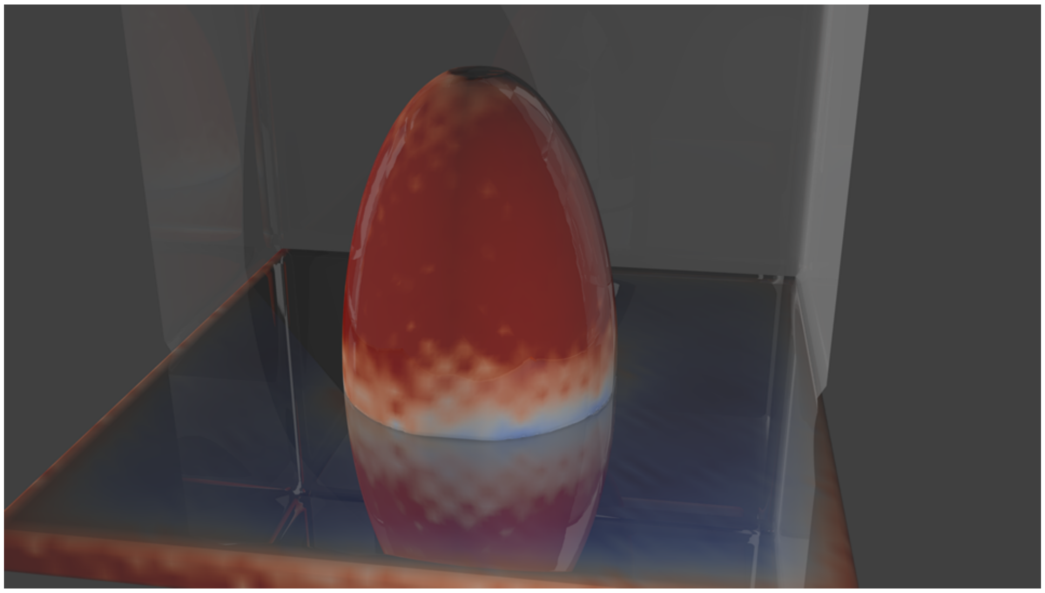


Fig. 4.15. Magnitude of strain forces on a quantum dot

Figures 4.1, 4.2, and 4.12 were included in promotional advertising for the Network for Computational Nanotechnology (NCN) and the iNemo Research Group at Purdue.

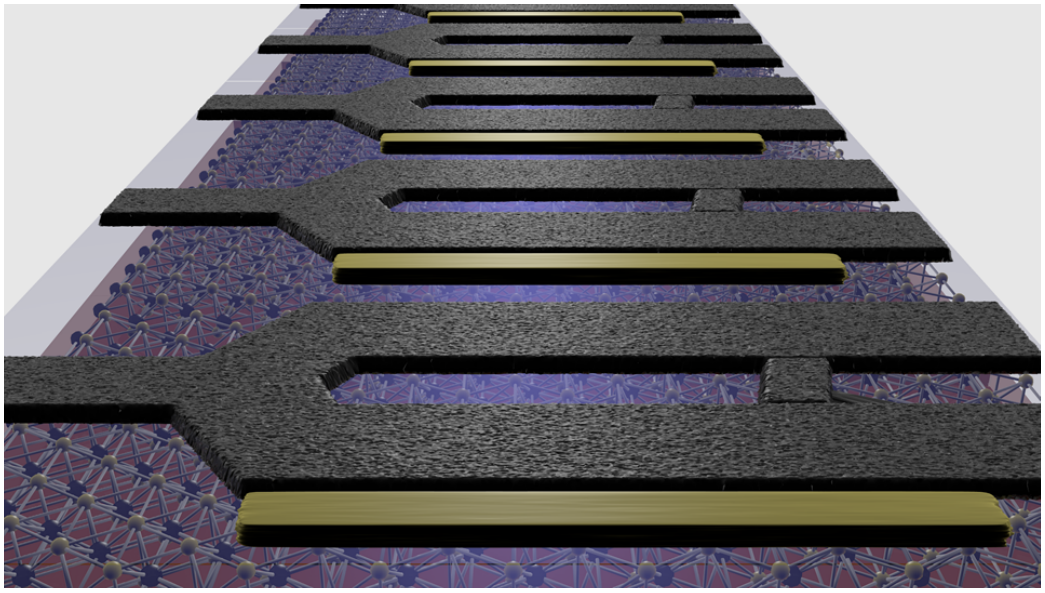


Fig. 4.16. Multiple Flying qubits

Figures 4.6, 4.7, 4.5, 4.2 and Fig. 4.16 were included in National Science Foundation (NSF) proposals.

Figures 4.9, 4.14, 4.13 and 4.13 were published in the Discovery NSF-supported magazine, [23], and has been used as reference in multiple publications [24], [25], [26], [23]. Similarly, Figure 4.11 was published in the Discovery NSF-supported magazine, and used in multiple publications [27], [24], [25], [26] and [28]

CHAPTER 5. NEMO5 CONTRIBUTIONS

In Addition to the NemoViz infrastructure, I have contributed different features into the NEMO5 code under the username *denphi*. I was the third contributor of the code based on lines of code up to December 2017 (see Figure 5.1).

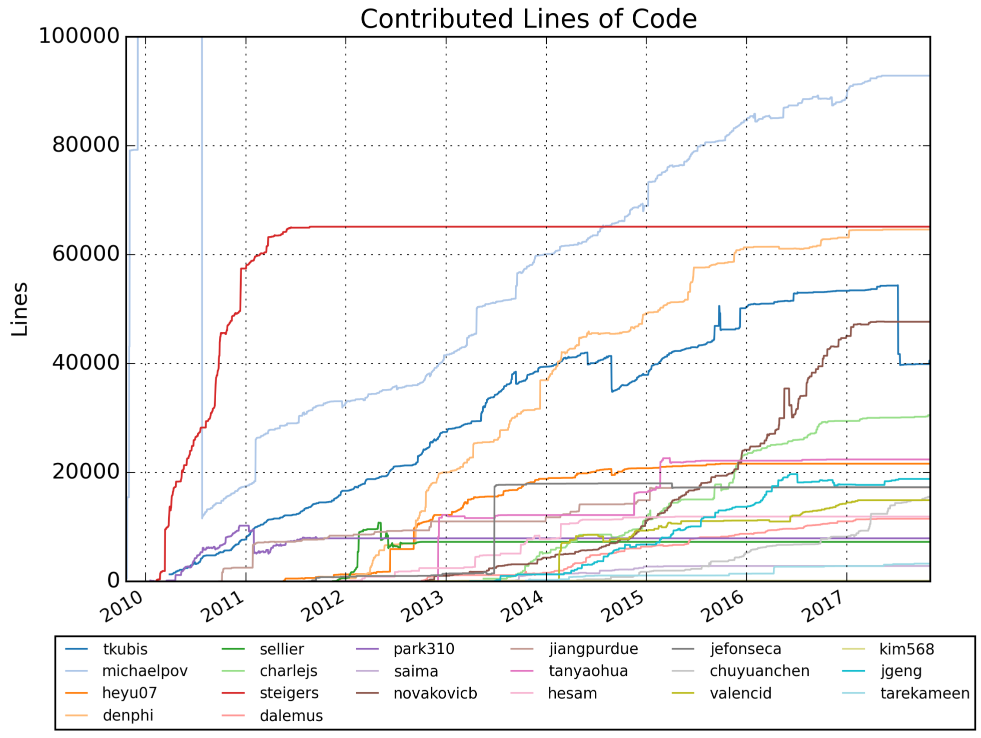


Fig. 5.1. Nemo5 lines of code by contributor

5.1 Contributions

Compilation: I re-factored all Makefile files and generalized the Makefile system. I could compile NEMO as fully functional on OSX, and partially functional on WINDOWS.

Code Optimizations: I re-factored the code that constructs the Hamiltonian matrices, and included a cache system to reuse Hamiltonian matrices when possible.

Database improvements: I included multiple features in the material database: database rules, database functional calls, database stack debugger, and database views.

Documentation: I developed input/output system documentation and improved the manual by including automatic documentation solvers. I implemented the NEMO5 command-line interface.

Python Interfaces: I developed multiple Python interfaces to NEMO5: Python input decks, Python templates (meta solvers), and Python solvers. I created multiple examples using these interfaces: Python RGF-Propagation, fitting code migrated to Python, and read-in potential (OMEN/Nemo3D/File).

Profiling: I developed an embedded profiling system, and its visualization (Santiago). I also developed the lifetime/timeline profiling system

Input / Output: I extended the input deck to support iterators, develop device templates, implement region surface solvers, and import shape regions (VTK).

Algorithms: I developed a new MPI parallelization scheme and implemented the interaction radius concept. I defined the cluster solvers, helped with coupling QTBM-Poisson, implemented the domain bisection algorithm, and developed a new adaptive grid implementation using adaptive mesh refinements.

Others: I defined the template factory implementation and included genetic algorithm libraries into Nemo5.

5.2 Infrastructure discussions

During the development of my contributions on the Nemo5 code, I was involved in multiple discussions to define guidelines to future Nemo developments. All guidelines were discussed as part of the software meetings hold by the Nanoelectronic Modeling Group at Purdue University. People involved in these discussions were: Santiago Perez-Rubiano, David Bermeo, James Charles, Daniel Mejia, Jim Fonseca, Tillmann Kubis, Michael Povolotskyi, and Gerhard Klimeck. Santiago Perez-Rubiano documented all the following guidelines and they are included as reference.

5.2.1 Guiding principles

The NEMO tool is a multi-physics, multi-scale, high performance computing, software for nanoelectronic devices simulation. Its target audience includes semiconductor industry R&D groups, nanoelectronics research groups and nanoelectronics students students. Its computational demands make it suitable for grid computing, HPC and even cloud computing, and so it must be able to deal with several restrictions, e.g. resources reliability issues, limited resources availability, etc ...

The software is developed by physics or electrical engineers with little or no software engineering background, however the developed software will evolve and has to cope with constantly changing requirements. Because of these the core functionality of the software should be really easy to reuse, hard to use in unintended ways e.g. up to the point to keeping the developers from compiling when some unintended uses are done. Perspectives

The NEMO tool needs to be seen from different perspectives in order to fully understand all its requirements. Some of the most important perspectives include:

1. Developers trying to write their models in NEMO5 by taking advantage of already available tools

2. Scientist trying to explore parameters, designs, etc

3. New users trying to understand the available features.

5.2.2 Software distribution requirements

- Should have a standard compilation process for both dynamic and static linking (some supercopmuters require static linking).
- Should be easy to disable all non-strictly necessary third-party libraries from the compilation process (e.g. VTK or libmesh).
- Should be compilable WITHOUT internet access!! (TianHe-2 and TMSC)
- Should be both an imperative (for experimentalist users?) and a declarative language (for programmers) that allow one to connect between different components..
- Should be compatible with Linux and Windows as much as possible (this will help to extend its user base).
- Should be distributed as web-services preferable under a Service-oriented architecture.

5.2.3 Testing framework

- Adding unit tests should be easy for developers, specially attaching files to a specific test for inputs should be easy.
- Tests should help on the search for scalability and so they need to measure time, memory and CPU consumption as we are interested in later assigning resources to different components based on their historic behavior.
- General tests should contain information regarding required execution resources
- Testing procedure (not just comparison of files in some cases) should be well defined

- Unit tests should be mandatory. If such a test is missing for a new method, the code should fail to compile.

5.2.4 Profiling and Logging framework

There should be well defined logging levels for final users (interested in measurements), profiling information, debugging information, etc. The time, and network location of every message must be recorded.

- Internal profiling (e.g. through a tic-toc system) should be available for developers to optimize their own code.
- Have the possibility to measure flops manually or automatically.
- Be able to classify tic-tocs in at least four categories : I/O, communication, computation, math operations.
- This should help to automatically detect load imbalance problems.
- Its output must be available through appropriate visualization tools which let the user:
- Filter/order sections of the code by their resources (time/memory/flops) consumption
- Compare two almost identical simulations that ran with different amount of resources. These would help to identify potential sources of scalability problems (functions that do not scale well).
- Plot the resources consumption across MPI ranks
- Explore information hierarchically
- Filter information by identifiers

5.2.5 Input/Output framework

- Theres a set of inputs received by any simulation. This options could be generated on the fly by the simulation.
- The output of the simulations should, in general, be done through a unified manager that decides where to store the output, otherwise every developer will store whatever and wherever they want.
- The process to generate plots from the output of NEMO6 should be standard and provided by a framework most of the time. This framework should be extensible enough to use plotting scripts from Matlab, Python, etc...

5.2.6 Options framework

- Should support options generated on the fly depending on the value of some other options.
- Should support the documentation of the options and the definition of its type and expected values (if available). The definition of a default value should be centralized (in the best case), in order for a user to know the default values before actually executing a simulation.
- There must be a way to know all the options set for a simulation, even the default ones
- Grouping options and lying out dependencies between them should be possible.
- Support inheritance of options
- Functionality should be dependent only on the presence or absence of an option.
- The way in which the options are parsed has to allow parameter exploration somehow, an example of this could be the way in which Makefiles allow you to define the value of a variable in the command line calls.

- There should be support for groups of options that depend on the value of another option.

5.2.7 Geometry specification framework

A region in space may have an arbitrary form, size and position. Regions may also have other algorithmic properties defined as well, like whether or not it is active, who is it distributed among MPI ranks, boundaries for simulation models. Regions should be represented as atomistic, continuous or Meshes. There should be mechanism to translate between different regions. Regions represent element of different layers of information, and each layers should share some common characteristics, there should be at least there supported layers: macro, micro, and atomistic.

Macro characteristics

- Atom's materials
- Form, size and position of chunks of materials
- Periodicity of chunks of materials
- Passivation on certain regions of the space
- Chemical coupling between different chunks of materials.
- It should be easy to import structures from other simulation software like VASP
- It should support different kinds of distribution among MPI ranks.
- It has to support regular/irregular/pseudoirregular structures (i.e. crystals, pseudo crystals and amorphous structures, created by various algorithms)
- It has to be easy to set up.

Micro characteristics

- Type of crystal to model a unit cell.

- Type of lattice to model a unit cell.
- There should be a possibility to have a non-integer amount of unit cells.

Atomistic characteristics

- Connection with other atoms
- Position of the atoms
- Get the N nearest neighbor atoms.

5.2.8 Simulation framework

Simulations should return data on demand. They only initialize and solve the problem when the data request method is called.

Main components on the tool should contain:

- Solvers: Basic computation unit. It should be stateless
- Methods: Group of solvers sharing a state.
- Modules: Group of methods and solvers, solving an specific computational problem.
- Meta-methods : Simulations with some undefined parameters that will be defined at runtime by other simulations at different stages (maybe when the simulation is initialized or maybe when it is running).

All components should shared local structures like:

- Domain atoms information
- DOF map
- There should be a way to differentiate between stateful and stateless simulations.

- Simulations may have geometry distribution or resources restrictions that should be explicitly stated somewhere
- There should be solver templates
- Methods need to be destroyed and recreated

5.2.9 Documentation framework

When writing an inputdeck one should be able to document details about:

- The structure that is being simulated, its geometry and its materials.
- The conceptual program flow.
- The expected outcome and output. Valid range for parameters..

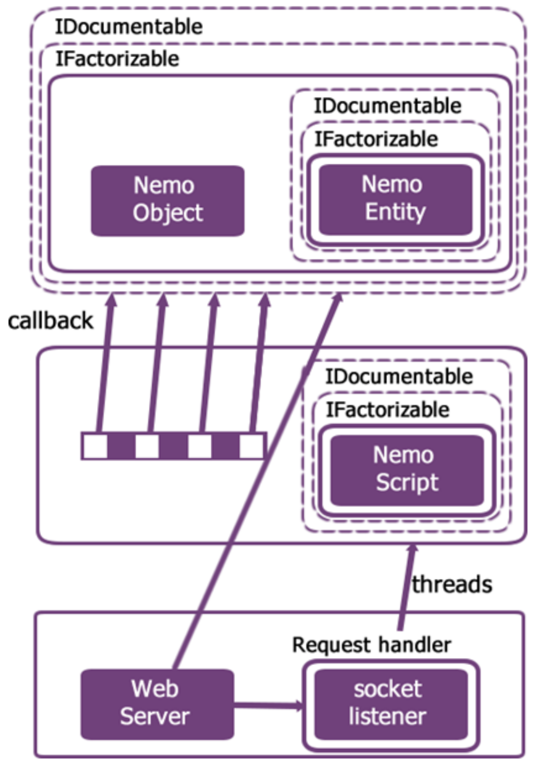


Fig. 5.2. NemoViz auto-documentation architecture

5.3 Nemo5 auto documentation

Figure 5.2 shows the architecture of Nemo5 auto-documentation. all classes that represent any entity on Nemo5 ecosystem should implement to basic interfaces: IDocumentable and IFactorizable.

5.3.1 Interface Factorizable

This interface requires any class to implement two basic method. First, get_factory_name method should return an unique name that is used as identifier by the factory. Second get_factory_aliases that return a list of identifier aliases or alternative names to be used by the factory.

All Nemo5 entities implement the Factorizable interface as shown in the table 5.1. Each class should define its class name and a parent class, and for each unique parent class, a new Factory constructor would be created using a singleton pattern. singletons are implemented using c++ templates and static variables. All objects created by the factory are automatically casted to the parent class as is shown in table 5.2.

5.3.2 Interface Documentable

All entities that are exposed to the final user have to implement the documentable interface, this interface requires classes to implement basic methods to document the options or parameters, developers have to document both inputs and outputs. Table 5.3 shows an example of basic documentation of input options of the Simulation class, the first call documents an option required to run the simulation (”name”), a second call documents an optional input (”domain”), and next calls defined the type of data that options need to have in order to have a proper behavior.

All documentation can be requested using methods of the interface, using the Nemo5 command line as shown in table 5.4, or using NemoViz.

Table 5.1.
Example implementation of factorizable interface, Dummy Module

```
...
class ModuleDummy :
/*extends*/public Module,
/*implements*/public NemoFactorizable<ModuleDummy,Simulation>
{
  public:
    virtual std::string get_factory_name ( void )
    { return "ModuleDummy"; }
    virtual void get_factory_aliases (std::set<std::string>&)
    { return }
    virtual ~ModuleDummy();
    virtual void do_solve();
...
}
```

Table 5.2.
NemoFactory instantiation example

```
...
Simulation* e;
e = NemoFactoryBase<Simulation >::new_instance("ModuleDummy");
...
```

5.4 Impact

Some publications that have used directly or indirectly some of my contributions to Nemo5 are:

Table 5.3.
Example implementation of factorizable interface, Dummy Module

```
...
class Simulation :
/*implements*/ public NemoDocumentableBase<Simulation>
{
...
  void set_input_options_map( void )
  {
  set_input_option_map("name",
    InputOptions::Req_Def( "A␣unique␣name␣tag␣..." ));
  set_input_option_map("domain",
    InputOptions::NonReq_Def( "", "Domain␣required␣..." ));
...
  set_input_option_property("name", "type",
    InputOptions::Type_Def(InputOptions::TYPE_STRING));
  set_input_option_property("domain", "type",
    InputOptions::Type_Def(InputOptions::TYPE_DOMAIN));
...
  }
}
```

1. KuangChung Wang, Teodor Stanev, Daniel Valencia, James Charles, Alex Henning, Vinod Sangwan, Aritra Lahiri, **Daniel Mejia**, Prasad Sarangapani, Michael Povolotskyi, A. Afzalian, Jesse Maassen, Gerhard Klimeck, Mark Hersam, Lincoln Lauhon, Nathaniel Stern, Tillmann Kubis, "Control of interlayer physics in 2H transition metal dichalcogenides" Journal of Applied Physics 122, 224302 (2017);

Nemo5 was used to explore different properties of TMD materials, quantum transport properties were simulated using the RGFModule solver. This solver encapsulates multiple solvers, dataflows and the RGF's algorithm. This is possible by the inheritance of the Module solver. Modules allow developers to create black boxes, that predefine a set of solvers and dataflows. In contrast, this information is usually read from an inputdeck. A Nemo5 module control the persistence of Nemo5 objects created on the module, can be created and destroyed at any point of the simulation, and can embed other modules.

2. Geng, Junzhe, Prasad Sarangapani, Erik Nelson, Ben Browne, Carl Wordelman, Tillmann Kubis, and Gerhard Klimeck. "NEMO5: realistic and efficient NEGF simulations of GaN light-emitting diodes." In Physics and Simulation of Optoelectronic Devices XXV, vol. 10098, p. 1009813. International Society for Optics and Photonics, 2017.

3. Long, Pengyu, Jun Z. Huang, Michael Povolotskyi, Devin Verreck, Gerhard Klimeck, and Mark JW Rodwell. "High-current InP-based triple heterojunction tunnel transistors." In Compound Semiconductor Week (CSW)[Includes 28th International Conference on Indium Phosphide & Related Materials (IPRM) & 43rd International Symposium on Compound Semiconductors (ISCS), 2016, pp. 1-2. IEEE, 2016.

4. Charles, James, Prasad Sarangapani, Roksana Golizadeh-Mojarad, Robert Andrawis, Daniel Lemus, Xinchen Guo, **Daniel Mejia**, Jim Fonseca, Michael Povolotskyi, Tillmann Kubis, Gerhard Klimeck, "Incoherent transport in NEMO5: realistic and efficient scattering on phonons" Journal of Computational Electronics, pp 17, 2016.

   This work describes some of the transport capabilities of Nemo5, in particular incoherent transport using the self-consistent Born approximation. This algorithm was implemented as a module in Nemo5 as described before for the RGFModule. This model suffers from convergence issues with scattering in the leads, and the solution in this case was to use a resonance mesh. ResonanceMesh solver in nemo5 resolves the meshes using adaptive mesh refinements. This algorithm not only gives a refined mesh close to the quantum resonances but also full control over the final number of points in the mesh.

5. Wang, Kuang-Chung, Daniel Valencia, James Charles, Yu He, Michael Povolotskyi, Gerhard Klimeck, Jesse Maassen, Mark Lundstrom, and Tillmann Kubis. "NEMO5: Predicting MoS 2 heterojunctions." In Simulation of Semiconductor Processes and Devices (SISPAD), 2016 International Conference on, pp. 221-224. IEEE, 2016.

6. Ankit Sharma, Ahmed Reza, Kaushik Roy, "Proposal of an Intrinsic-Source Broken-Gap Tunnel FET to Reduce Band-Tail Effects on Subthreshold Swing: A Simulation Study" IEEE Transactions on Electron Devices, Volume:6, Issue: 6, Page(s): 2597 - 2602, 2016

7. Ankit Sharma, Arun Akkala, Jaydeep , Kaushik Roy, "Source-Underlapped GaSbInAs TFETs With Applications to Gain Cell Embedded DRAMs" IEEE Transactions on Electron Devices, Volume:63, Issue: 6, Page(s): 2563 - 2569; doi:10.1109/TED.2016.2555627, 2016.

8. Ameen, Tarek A., Hesameddin Ilatikhameneh, Gerhard Klimeck, and Rajib Rahman. "Few-layer phosphorene: An ideal 2D material for tunnel transistors." Scientific reports 6 (2016): 28515., 2016

9. Chen, Fan W., Hesameddin Ilatikhameneh, Gerhard Klimeck, Zhihong Chen, and Rajib Rahman. "Configurable electrostatically doped high performance bilayer graphene tunnel FET." IEEE Journal of the Electron Devices Society 4, no. 3 (2016): 124-128.

10. Ilatikhameneh, Hesameddin, Gerhard Klimeck, and Rajib Rahman. "Can homojunction tunnel FETs scale below 10 nm?." IEEE Electron Device Letters 37, no. 1: 115-118, (2016)

11. Ilatikhameneh, Hesameddin, Yaohua Tan, Bozidar Novakovic, Gerhard Klimeck, Rajib Rahman, and Joerg Appenzeller. ”Tunnel field-effect transistors in 2-D transition metal dichalcogenide materials.” IEEE Journal on Exploratory Solid-State Computational Devices and Circuits 1: 12-18. 2015

12. Chen, Fan W., Michael Manfra, Gerhard Klimeck, and Tillmann Kubis. ”NEMO5: Why must we treat topological insulator nanowires atomically?.” In Proc. IWCE. 2015.

13. Ilatikhameneh, Hesameddin, Fan W. Chen, Rajib Rahman, and Gerhard Klimeck. ”Electrically doped 2D material tunnel transistor.” In Computational Electronics (IWCE), 2015 International Workshop on, pp. 1-3. IEEE, 2015.

14. Li, Wenjun, Saima Sharmin, Hesameddin Ilatikhameneh, Rajib Rahman, Yeqing Lu, Jingshan Wang, Xiaodong Yan et al. ”Polarization-engineered III-nitride heterojunction tunnel field-effect transistors.” IEEE Journal on Exploratory Solid-State Computational Devices and Circuits 1 (2015): 28-34.

Table 5.4.
NemoFactory instantiation example

```
$ ./bin/nemo -h Dummy1
Usage: nemo INPUTDECK_FILE [--profiling [PROFILING_TYPE...
NanoElectronics MOdeling Tool 5 (NEMO5), purdue univer...
Type nemo --version(-v) to see the program version and...
Type nemo --help(-h) [ENTITY_TYPE] to see entity options


-----------------------------------------------------------

-------------------------- SOLVER --------------------------
         Dummy1 : This simulation is a TEST simulation ...
[#]      (*) = required ,          (____)= default value
-----------------------------------------------------------

[1]      ( ) _disable_init (false) : flag command to ...
...
[14]     ( ) active_regions (()) : required regions ...
[15]     ( ) boundary_regions (()) : required regions ...
[16]     ( ) d () : Name of the simulation to be  ...
[17]     ( ) default_solver_mat_type (PetscMatrix ...
[18]     ( ) domain () : Domain required for this ...
[19]     (*) name : A unique name tag / identifier ...
[20]     ( ) output_file_suffix () : suffix to be  ...
[21]     ( ) surface_of_regions () : surface  ...
[22]     ( ) tic_toc_name ($(name)) : Prefix for ...
[23]     (*) type : A unique type tag / identifier ...
-----------------------------------------------------------
```

CHAPTER 6. EVALUATION AND DISCUSSION

Evaluation of new visualizations and visualization frameworks is one of the top challenges of the visual analytics field [29] [30], and there is an absence of well-defined methods of evaluation. Measuring accuracy, utility, and efficiency are the most accepted measurement methods [31]. Recently, several studies have been exploring user experience goals as part of the evaluation. However, user experience evaluations should be complemented with other standard measures in order to identify key elements of the visualization [32].

NemoViz was evaluated by two different methods: a user experience study defined as an informal evaluation test, and a quantitative study that measured efficiency and effectiveness. Both studies fell under the Purdue IRB Exemption (1706019282), and users taking part in these studies were part of the Nanoelectronic Modeling Group at Purdue University

6.1 Informal evaluation test

First, NemoViz was assessed through an informal evaluation. This evaluation was designed to ask users of NEMO5 with previous exposure to NemoViz about their experience with the tool. The design of the informal evaluation test followed the main recommendations of Davidson's methodology for designing evaluation tools (2005-2018) [33].

1. Defining evaluative dimensions: Establish key components to be assessed.
2. Determining dimensions' importance: Prioritize and justify the importance of each dimension,
3. Developing instruments: Define tools that could gather the data appropriately.

4. Constructing rubrics: Define rules and formulas to assign scores.
5. Measuring performance: applying the rubrics to the data we gathered.
6. Results and conclusions.

6.1.1 Defining evaluative dimensions

To evaluate NemoViz the informal evaluation covered four dimensions:

- Functionality: The tool allows users to explore text files based on visualizations.
- Content/Design: The tool captures atomistic representation of the model and summarizes relations between elements in the text files.
- Usability: The tool is easy to use.
- Efficiency: The tool helps users to save time.

The questions, listed in Table 6.1, were designed based on work of [34] and modified to be used for the evaluation of NemoViz.

6.1.2 Determining dimensions' importance

Table 6.1 shows the importance defined for each dimension. In this case, the most important dimensions were as follows. First, to know if interactive visualization helps users to save time when modifying input decks. Modifying and running input decks are the tasks most commonly performed by users of any simulator tool, and a single modification in the input parameters is equivalent to a new experiment. Second, to know if NemoViz allows users to explore input decks in a simplified way. The main design concept in the designing of NemoViz was to allow users to interactively highlight different elements of the input deck. Other dimensions are important as well, but they are not the main focus of the evaluation of the impact of NemoViz.

Table 6.1.
Evaluative Dimensions and Dimensional Evaluation Questions

Dimension	**Question**	**Weighting**
Functionality	NemoViz supports the user well in inspecting a Nemo5 input deck	Very important
Content/Design	NemoViz structures and summarizes the atomistic representation of a model defined in a Nemo5 input deck	Important
Content/Design	NemoViz structures and summarizes relations between different elements in a Nemo5 input deck	Important
Usability	NemoViz is easy to use and self-explanatory.	Important
Efficiency	NemoViz saves you time when modifying Nemo5 input decks	Very important

6.1.3 Developing instruments

The instrument consisted of five (5) questions designed to assess the usability of NemoViz, and NemoViz's ability to save users time, to structure and summarize spatial information and relations from an input deck, and to enable users to explore the input deck (Table 6.1). Information about the level of expertise of the user and optional feedback were included as well. The questionnaire was created with the Qualtrics survey tool, and it was sent as an on-line survey.

6.1.4 Constructing rubrics

The questions were scored using a Likert-type scale from 1 (*strongly agree*) to 5(*strongly disagree*).

6.1.5 Measuring performance

Table 6.2.
Scored value for each dimension. Functionality and Efficiency dimensions scored the lowest values (lower scores are better.)

Dimension	Score
Functionality	9
Content/Design 1	14
Content/Design 2	15
Usability	15
Efficiency	12

Nine (9) NemoViz users answered the on-line survey. Most of the participants (57%) were expert users of Nemo5, 29% were intermediate users, and 14% said they had just started to use Nemo5. The most important dimensions scored the lowest values (Table 6.2), supporting the hypotheses of the functionality and efficiency of the tool.

6.1.6 Results and Discussion

The results from the informal evaluation test showed that 100% of the participants (strongly agree + agree) agree that NemoViz provides support when exploring Nemo 5 input decks and captures the atomistic representation of the model defined in the NEMO5 input deck. The exploration of the input decks is directly related to the visualizations presented in the outline view of NemoViz. Some users also found the relation view useful for exploring relationships among blocks. The representation of the atomistic structure of the simulation is mostly shown in the main view of NemoViz. Some characteristics of the structure were also located in the property and relation views.

Table 6.3.
Percentages of participants that answer the questionnaire, listed from 1 (*strongly agree*) to 5 (*strongly disagree*)

Question	**1**	**2**	**3**	**4**	**5**
NemoViz supports the user well in inspecting a Nemo5 input deck	100%	0%	0%	0%	0%
NemoViz structures and summarizes the atomistic representation of a model defined in a Nemo5 input deck	43%	57%	0%	0%	0%
NemoViz structures and summarize relations between different elements in a Nemo5 input deck	43%	43%	14%	0%	0%
NemoViz is easy to use and self-explanatory	43%	43%	14%	0%	0%
NemoViz saves you time when modifying Nemo5 input decks	72%	14%	14%	0%	0%

Regarding the ability of NemoViz to summarize relations between input deck blocks, 86% of the participants perceived that NemoViz accurately represents these relations (Table 6.3). Only 14% of the participants responded that NemoViz does not add value to the structure and relations found in the text file of the input deck. Regarding usability, 86% of participants reported that NemoViz was easy to use and intuitive. We hypothesize that elements such as its simple design, extensive use of visualizations, real-time synchronization of the multiple views, and incorporation of widgets contribute to the usability. These elements were intentionally incorporated into the design of NemoViz for this purpose.

Finally, and most importantly, a very significant number of users (86%) reported that NemoViz helps them to save time when modifying the input decks. Modifying and running the input decks are the tasks most commonly performed by users of any simulator tool given that a single modification of the input parameters is equivalent to a new experiment. The other 14% reported spending the same amount of time modifying the NEMO5 input decks using NemoViz or using a text editor. It is worth noting that 100% of the expert users reported a decrease in the time spent modifying input decks when using NemoViz.

6.2 Identification of relevant errors, Previous User Experience test

The informal evaluation showed that the efficiency dimension scored highly. Moreover, optional feedback from the informal evaluation test highlighted the importance of NemoViz to detect errors when modifying an input deck. The most common problem that users faced when modifying an input deck is the detection of errors. In other words, users had to debug an input deck.

To determine NemoViz's efficiency in debugging input decks, the most relevant errors encountered by NEMO5 users were detected using an additional evaluation based on user experience.

6.2.1 Defining evaluative dimensions and importance

In this study three different dimensions were measured:

- Frequency: How often users observed the error
- Complexity: How complicated is the process to fix the error.
- Difficulty: How much time a user takes to fix the error.

The most common input-deck errors were classified as follows:

- domain sizes are not correctly defined (structure dimension), domain positions are not correctly defined (structure position),
- geometrical regions are not properly defined (spatial definitions)

- crystal orientation is not well defined (crystal information)
- missing connections between solvers (solution-solution relations)
- missing connections between domains (domain-domain relations)
- missing relations between domains and solvers (solution-structure relations)
- missing relations between regions and solvers (solution-spatial relations)
- inconsistency between the domains and geometrical regions (spatial-structure relations).

6.2.2 Developing Instrument and Rubrics

Table 6.4.
Tabulated results from user experience evaluation.

Dimension	**Frequency**				**Difficulty**				**Complex**		
	0	**1**	**2**	**3**	**1**	**2**	**3**	**4**	**1**	**2**	**3**
Structure Dimension	2	4	6	2	4	5	2	3	9	3	1
Structure Position	1	6	5	2	2	7	1	4	6	5	2
Spatial Definitions	2	3	7	2	3	3	5	3	7	4	2
Crystal information	4	7	3	0	4	2	5	3	6	7	1
Solution-Spatial Relations	3	6	3	2	7	4	2	1	11	2	1
Solution-Solution Relations	1	7	5	1	7	3	1	3	9	4	1
Structure-Structure Relations	1	4	5	4	3	4	5	2	10	2	2
Spatial-Structure Relations	3	5	3	3	2	4	6	2	8	5	0
Solution-Structure Relations	4	8	0	2	6	3	3	2	11	3	0

These previous experience measurements were designed as a survey to ask NEMO5 users about their own perception of debugging each type of error. The questionnaire was created following the same format as the first evaluation, it was implemented with Qualtrics survey tool, and it was sent as an on-line survey. Each question was scored based on Likert-type scales: How often do you face this error? from 0 (*never*) to 3 (*very often*); How difficult is the process of fixing this error? from 1 (*easy*) to 4 (*very difficult*); and how long does it usually take you to solve it? from 1 (*few minutes*) to 3 (*more than an hour*). Table 6.4 shows the tabulated data for each dimension

To compare between dimensions, scores were normalized and values between 0.0 and 1.0 were assigned.

6.2.3 Measuring Performance and Results

Table 6.5.
Scores (S) and Normalized scores (uS) for the three dimensions evaluated in the user experience evaluation.

Dimension	**Frequency**		**Difficulty**		**Complex**	
	S	**uS**	**S**	**uS**	**S**	**uS**
Structure dimension	22	0.69	32	0.64	22	0.18
Structure position	22	0.69	35	0.91	29	0.82
Spatial definition	23	0.77	36	1.00	27	0.64
Crystal Information	13	0.00	35	0.91	31	1.00
Solution-Spatial relations	18	0.38	25	0.00	21	0.09
Solution-Solution relations	20	0.54	28	0.27	25	0.45
Structure-Structure relations	26	1.00	34	0.82	24	0.36
Spatial-Structure relations	20	0.54	36	1.00	23	0.27
Solution-Structure relations	14	0.08	29	0.36	20	0.00

Fourteen (14) NEMO5 (not necessarily exposed to NemoViz) users answered this new survey following the same format as the first evaluation. These results indicate

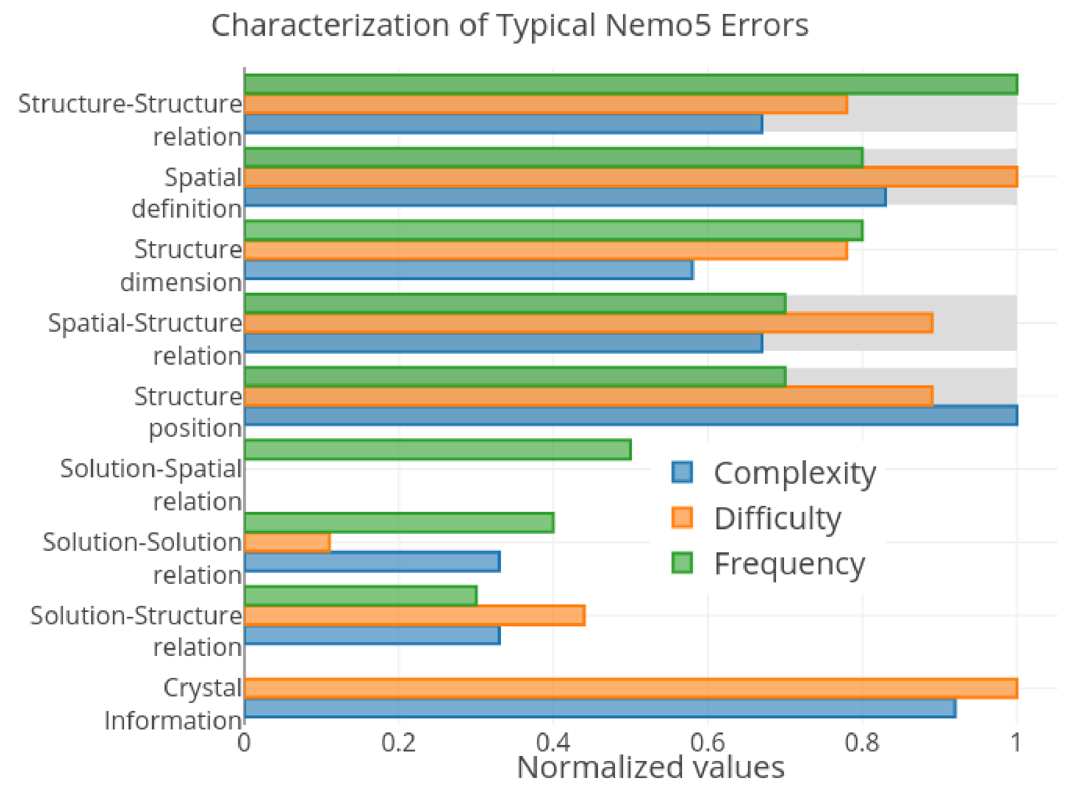


Fig. 6.1. Normalized scores: results are ordered by frequency and gray areas highlights questions that scored above 0.6 in all dimensions.

that the most frequent errors were related to the structure definitions and relations. The most complex and frequent errors were related to crystal information. However, these latter errors were far less common. The results also suggest that errors involving relations between regions and solutions were easily solvable.

6.2.4 Results and Discussion

Five (5) error types were classified as the most important errors Nemo5's users face when dealing with input-deck debugging (see Figure 6.1). Four (4) error types obtained a high score (0.6 or higher) on all test dimensions: structure-structure relations (StrStrRel), spatial definitions (SDef), spatial-structure relations (SStrRel), and structure position (StrPos). However, based on the assumption that the most frequent errors have a larger impact, the structure-dimension (StrDim) error is also classified as an important error to evaluate.

6.3 Quantitative Evaluation

NemoViz expands the user's cognitive process in different ways, and detecting the input deck errors can be defined as two cognitive processes: spatial representations that involve spatial cognition processes and well-defined mental models, as well as abstract representations that require new mental models to represent the relations. A measurement instrument was developed based on the error types previously identified.

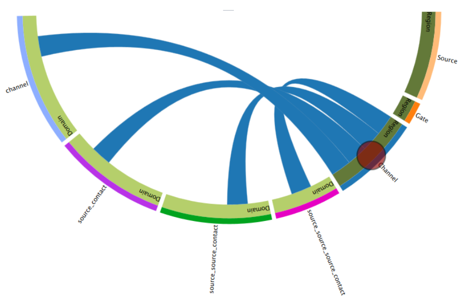
6.3.1 Measurement instrument

Table 6.6.
Mean, Standard Deviation, and number of questions included in the pilot run. (+) Original, (-) Removed, (*) Included

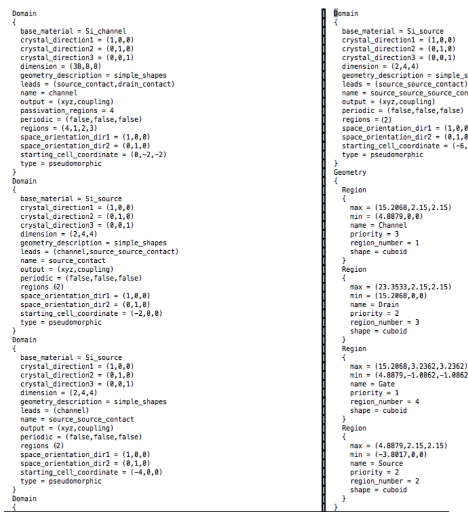
	Statistics		**Questions**		
Error type	**Mean**	**STD**	+	-	*
Structure-Structure Relations	56.96	10.11	8	2	6
Spatial Definition	49.78	3.76	8	4	4
Structure Dimension	25.09	6.11	8	2	6
Spatial-Structure Relations	103.66	25.83	8	2	6
Structure Position	35.11	9.91	8	2	6
Total			**40**	**14**	**26**

We validate the spatial representation errors by analyzing users' understanding of the main view, and the abstract representation by users' understanding of the relations view. We chose a widely used quantum transport calculation of a silicon nanowire (see Appendix B). For each error type, four (4) input decks were created, the corresponding errors introduced, and NemoViz visualization snapshots taken (see Figure 6.2). The survey was implemented with the Qualtrics survey tool assigned to a specific IP address, and a timer was added for each question.

Initially, two participants took the survey that included all forty (40) possible questions. input deck associated with questions for which their answering time fell more than three (3) standard deviations were removed from the pool of questions. Twenty-six (26) possible questions were included as part of the instrument 6.6 (see Table 6.6).



(a) NemoViz representation



```
Domain
{
  base_material = Si_channel
  crystal_direction1 = (1,0,0)
  crystal_direction2 = (0,1,0)
  crystal_direction3 = (0,0,1)
  dimension = (38,8,8)
  geometry_description = simple_shapes
  leads = (source_contact,drain_contact)
  name = channel
  output = (xyz,coupling)
  passivation_regions = 4
  periodic = (false,false,false)
  regions = (4,1,2,3)
  space_orientation_dir1 = (1,0,0)
  space_orientation_dir2 = (0,1,0)
  starting_cell_coordinate = (0,-2,-2)
  type = pseudomorphic
}
Domain
{
  base_material = Si_source
  crystal_direction1 = (1,0,0)
  crystal_direction2 = (0,1,0)
  crystal_direction3 = (0,0,1)
  dimension = (2,4,4)
  geometry_description = simple_shapes
  leads = (channel,source_source_contact)
  name = source_contact
  output = (xyz,coupling)
  periodic = (false,false,false)
  regions (2)
  space_orientation_dir1 = (1,0,0)
  space_orientation_dir2 = (0,1,0)
  starting_cell_coordinate = (-2,0,0)
  type = pseudomorphic
}
Domain
{
  base_material = Si_source
  crystal_direction1 = (1,0,0)
  crystal_direction2 = (0,1,0)
  crystal_direction3 = (0,0,1)
  dimension = (2,4,4)
  geometry_description = simple_shapes
  leads = (channel)
  name = source_source_contact
  output = (xyz,coupling)
  periodic = (false,false,false)
  regions (2)
  space_orientation_dir1 = (1,0,0)
  space_orientation_dir2 = (0,1,0)
  starting_cell_coordinate = (-4,0,0)
  type = pseudomorphic
}
Domain
{
```

```
Domain
{
  base_material = Si_source
  crystal_direction1 = (1,0,0)
  crystal_direction2 = (0,1,0)
  crystal_direction3 = (0,0,1)
  dimension = (2,4,4)
  geometry_description = simple_shapes
  leads = (source_source_contact)
  name = source_source_source_contact
  output = (xyz,coupling)
  periodic = (false,false,false)
  regions = (2)
  space_orientation_dir1 = (1,0,0)
  space_orientation_dir2 = (0,1,0)
  starting_cell_coordinate = (-6,0,0)
  type = pseudomorphic
}
Geometry
{
  Region
  {
    max = (15.2068,2.15,2.15)
    min = (4.8879,0,0)
    name = Channel
    priority = 3
    region_number = 1
    shape = cuboid
  }
  Region
  {
    max = (23.3533,2.15,2.15)
    min = (15.2068,0,0)
    name = Drain
    priority = 2
    region_number = 3
    shape = cuboid
  }
  Region
  {
    max = (15.2068,3.2362,3.2362)
    min = (4.8879,-1.0862,-1.0862)
    name = Gate
    priority = 1
    region_number = 4
    shape = cuboid
  }
  Region
  {
    max = (4.8879,2.15,2.15)
    min = (-3.8017,0,0)
    name = Source
    priority = 2
    region_number = 2
    shape = cuboid
  }
}
```

(b) Input deck segment

Fig. 6.2. Visualizations of failing input decks after introducing artificial errors of the abstract representations type

Users were asked to classify the correct type of error based on a visualization or a segment of an input deck. In order to avoid biased results, the users were asked to schedule an appointment in a controlled space with a prepared desktop to answer the survey. Additionally, before the survey was taken, users had to take a small tutorial on how NemoViz represents input decks.

The tutorial consists of a small description of the error types Nemo5's users face while debugging input decks (see Appendix A). Participants had to read the entire

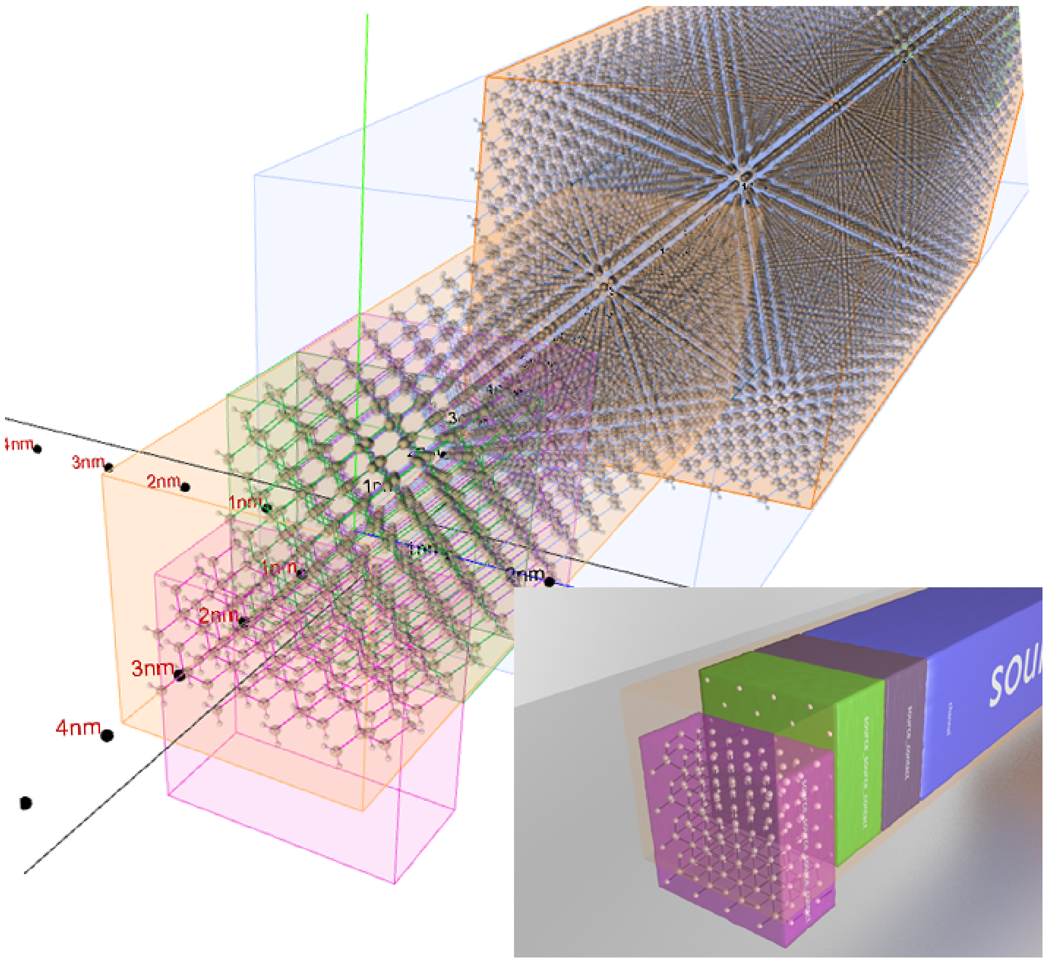


Fig. 6.3. Example of the Tutorial's feedback after each test answer

Table 6.7.
Domain Position error explanation

Domain position: The position of a domain is not aligned with its adjacent domains. In this case, the source_source_source_contact domain (pink box) is not aligned with the other domains included in the source region, the pink box is moved down. Notice that this visualization only shows half of the atoms in the source_source_source_contact domain (pink box). This is because only the atoms that are part of a region AND a domain are shown in the intersection view. Since the domain is misplaced, the atoms that are not part of a region are not shown. This does not mean that the size (number of atoms) of the domain is incorrect. The missing atoms are just not shown due to the misplacement of the domain. This simulation would simulate a domain with half of its atoms missing.

text and answer a set of training questions (a question for each error type). For each of the training questions, participants got feedback independently about the correctness of their answer showing a description of the problem (Table 6.7) and an illustration of the three-dimensional representation (see Figure 6.3).

After the tutorial section, users were exposed to three questions for each error type: two visualizations and one text segment. A total of fifteen questions were grouped into two sections: Questions involving spatial cognition processes (9 questions) and questions involving new mental models to represent relations (6 questions). The order of appearance of the questions was random.

6.3.2 Measuring Performance

The time taken by each respondent to answer each survey question and the number of correct answers were measured. The effectiveness of users in detecting errors using either the visualization or the text was calculated as the percentage of correct answers (accurately identified errors) over the total number of questions. The effectiveness using the visualization or text input decks was also compared. The efficiency of the visualization and text input deck were similarly calculated as the time it took the user to correctly identify an error.

Twelve (12) participants answered the survey, with an average effectiveness of 87%, an average time of 49.23 seconds for each answer, and an average time of 50.10 minutes to complete the tutorial and answer all the questions (see Appendix C).

6.3.3 Results and Discussion

In general, the users improved their effectiveness of error detection by 7% and were able to detect errors twice as fast using NemoViz compared to simply analyzing the text input decks. The left-hand box plot of Figure 6.4 shows the increased average accuracy. The standard deviation shrank dramatically to less than 10% when using NemoViz. The right-hand box plot shows how the average time users took to detect errors was cut in half when interactive visualizations were used. The standard deviation was reduced even further.

We also analyzed data about the users' expertise and the question type. The graph in the bottom right quadrant of Figure 6.5 illustrates that non-expert users improved their detection times threefold when using NemoViz. Expert users showed a twofold improved detection time when using NemoViz (bottom left quadrant of Figure 6.5). Detection times for the structure-structure relations and structure dimension questions did not show any dramatic improvement when using visualizations. However, spatial-structure relations experienced extraordinary improvements in time and efficiency (Top of Figure 6.5).

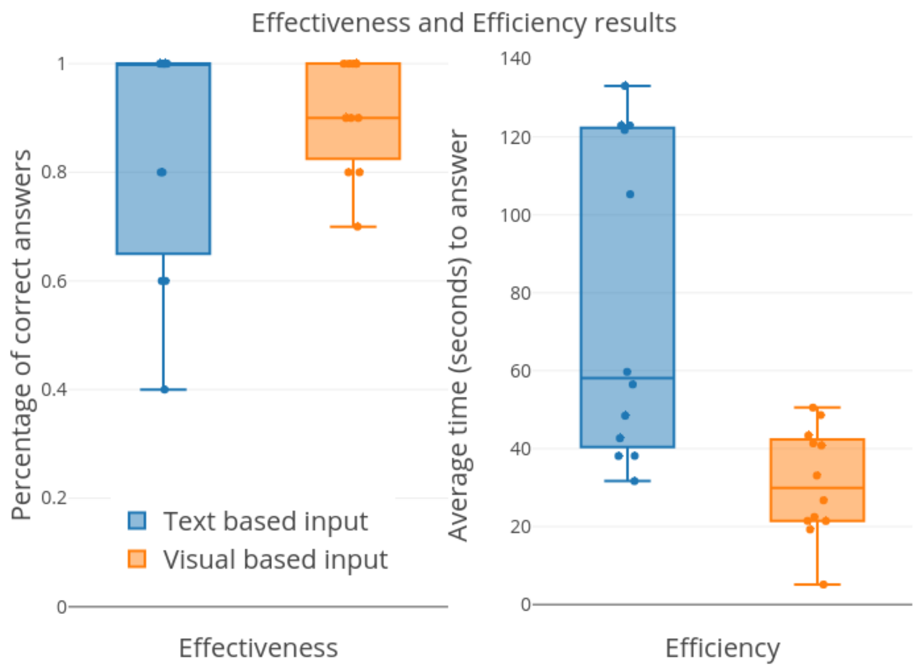


Fig. 6.4. Box plots showing users' effectiveness and efficiency averaged and quantiles. Blue boxes show visually based input results, orange text-based input results.

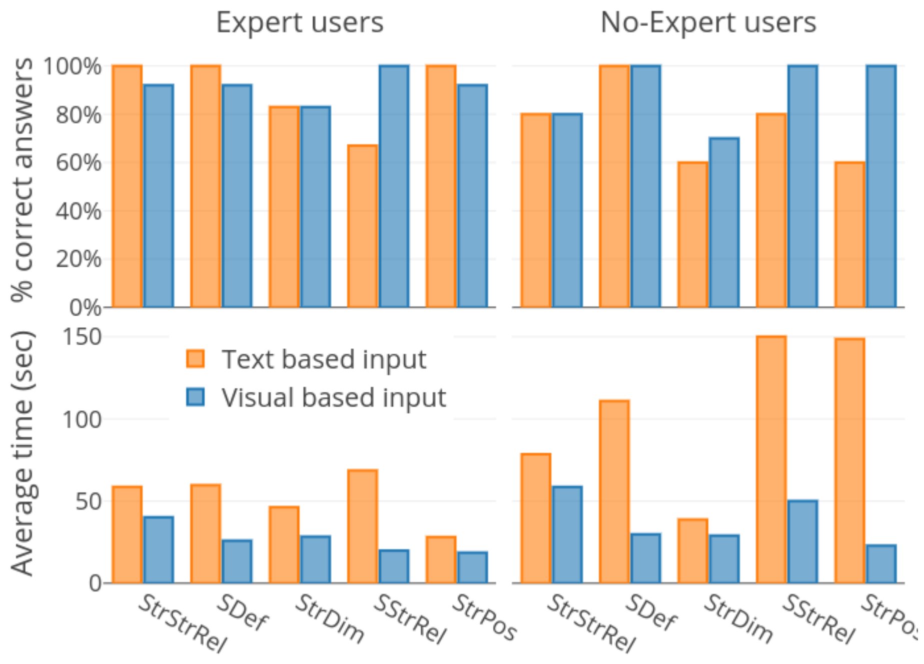


Fig. 6.5. Comparison of expert and non-expert users. Top: Average percentage of correct answers. Bottom: Average time to correctly detect errors

CHAPTER 7. SUMMARY AND OUTLOOK

The potential impact of research simulation codes is hindered by the lack of usability by those not already deeply familiar with them. This thesis describes how the introduction of interactive visualizations helped whit this problem. Visual analytics were included as a way to improve usability in a research code and decrease the learning curve of new users.

In the process, a framework was proposed to describe the complete simulation process and was illustrated with a case study. This framework is a combined model that describes the complete simulation and serves as a reference to identify key stages that would enhance usability and user experience. Nemo5 users' experience was described based on this framework, and the role of visualizations in this experience was illustrated with examples.

Visualizations are useful for each step in the simulation process, but also help users to transition from one step to another in the process. All examples described in this work consisted of visualizations that support Nemo5 users and help them to understand simulation inputs and outputs.

All visualizations in this work were implemented as part of an interactive visualization system called NemoViz. This work defines the requirements needed for any interactive system that aims to support atomistic simulations. NemoViz meets all these requirements mainly through the use of interactive visualizations and tools to disseminate simulation results.

NemoVizs impact was measured on the most important and common task Nemo5 users face: modifying an input deck. In general, users take a working and functional simulation input and modify some parameters to create a new configuration. This new input is equivalent to a completely new experiment, and small changes in parameters can have huge impacts in the simulated structure. In particular, inclusion of errors could make structures unreasonable. The detection of these errors involves a complex cognitive process.

Results suggests that NemoViz enhances the cognitive process during error detection as follows: 1) it improves user efficiency and effectiveness in debugging NEMO5 input decks; and 2) it accelerates the learning curve of novice users by enhancing their effectiveness to the level of expert users. Enhancing the cognitive process improves the usability of research codes, and the introduction of visual analytics as part of the design process highlights new ways to deliver research codes to final users.

This work describes some of the products created using NemoViz. These products were generated mainly via the dissemination tools. This work documents the impact they had in different research projects.

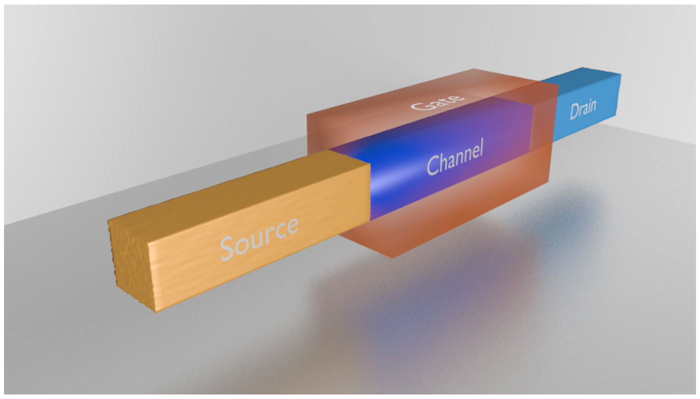
The infrastructure created along with this work defines an ideal workspace to continue investigating the impact of visualization on the simulation process.

APPENDIX A
QUANTITATIVE VALIDATION TUTORIAL

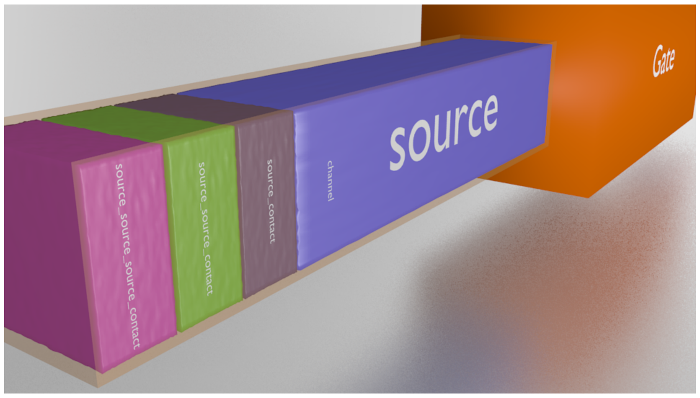
Spatial Representations

Thank you for agreeing to take part in this important survey for my research. This survey consists of a Nemo Server training session followed by a set of questions. It should take you less than 20 minutes to complete the training and the questions. Be assured that all answers you provide will be kept anonymously. As this survey is trying to keep track of the effectiveness, please be sure you remove all distractions that can affect your time of response.

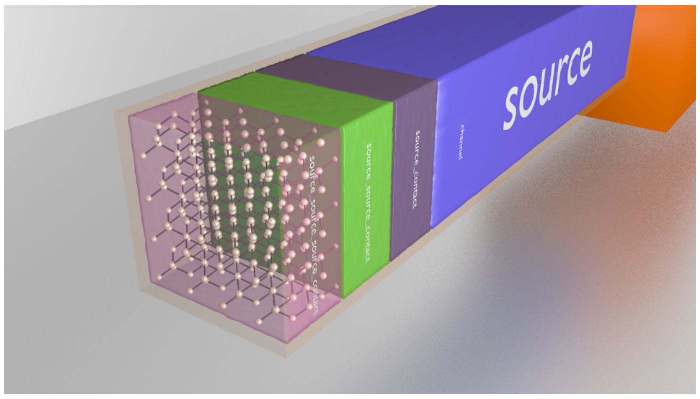
All questions in this survey are based on a Nemo5 input-deck that simulates quantum transport across a silicon nanowire as shown in the figure below. The nanowire is surrounded by a gate all-around, and it contains two contacts to source and drain regions.



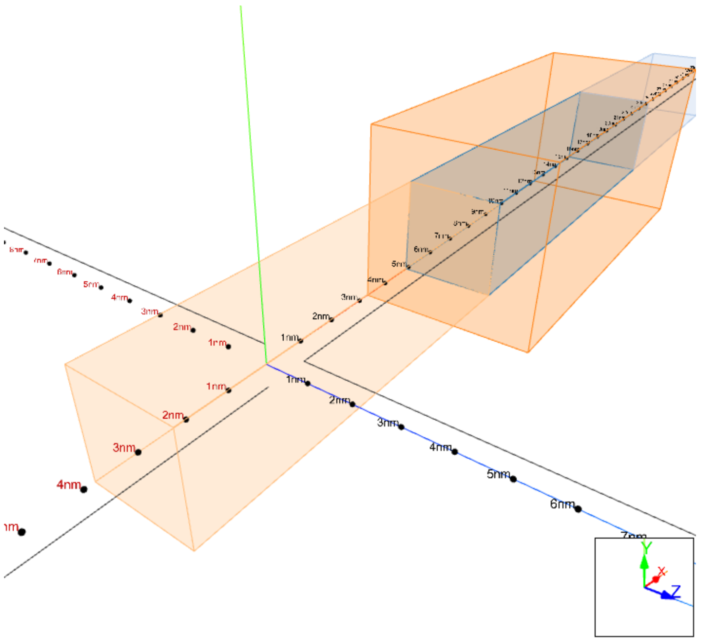
The transport simulation uses the Quantum Transport Boundary Method (QTBM), and its implementation in Nemo5 requires to explicitly define three additional adjacent domains for each contact as shown in the figure below. These domains have been called source_contact, source_source_contact, and source_source_source_contact. This naming scheme has become the ”standard” way to define the contacts in any Nemo5 Simulation.



Each domain in Nemo5 is represented as a block of atoms that is defined by the unit cell of a specific material. The intersection between Domains and Geometrical regions represents the spatial work-space to be used in the simulation as a representation of the device to be simulated (the ”real device”)

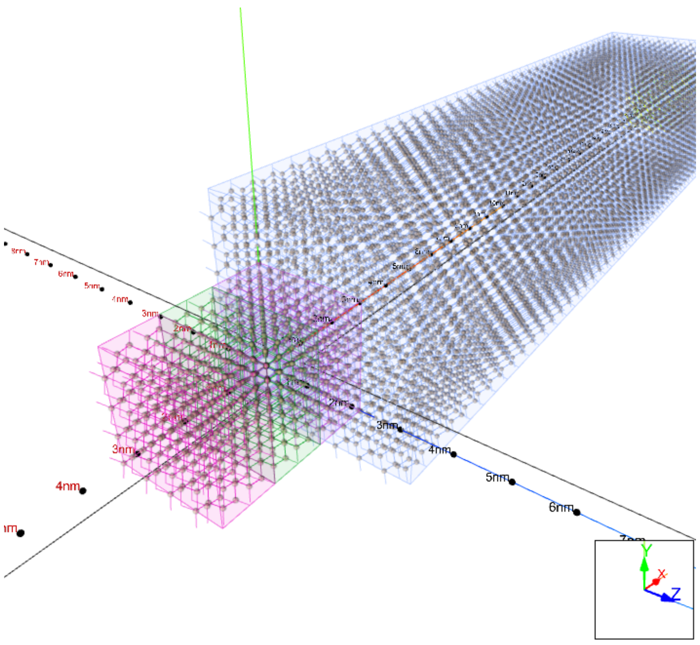


Nemo Server is a web-based tool for visualization of Nemo5 input-decks. With Nemo Server you will be able to see the 3D representations of the geometrical regions and the domain definitions from a Nemo5 input-deck. The figure below shows the view in Nemo Server of the geometrical regions from a Nemo5 input-deck that simulate quantum transport across a silicon nanowire. In this example, there are 4 geometrical regions represented as translucent boxes. Each box represents a region of the nanowire, as follows: source (light orange box), channel (blue box), drain (light blue box), and gate (orange box). Make sure that you are able to identify these 4 regions in the figure below.



Nemo Server can also show domains defined in a Nemo5 input-deck. Nemo Server shows the position of the atoms in each domain. The surrounding surface of the domain is represented by a translucent box. The figure below shows the domains for the simulation of quantum transport across a silicon nanowire. The figure below shows 4 domains: source_contact in purple, source_source_contact in green, source_source_source_contact in pink, and channel in blue. Make sure that you are able to identify all these domains in the figure below. Notice that this is the same device as in the previous example, but in this case, we are looking at the domains, not the regions

Finally, Nemo Server gives you the option to visualize parts of the device that are defined as a region AND as a domain. Some people call this the real device. You could also think about this visualization as an intersection between the domains and the regions (previous two figures). The figure below shows the visualization of the real device for a simulation of quantum transport across a silicon nanowire (domains

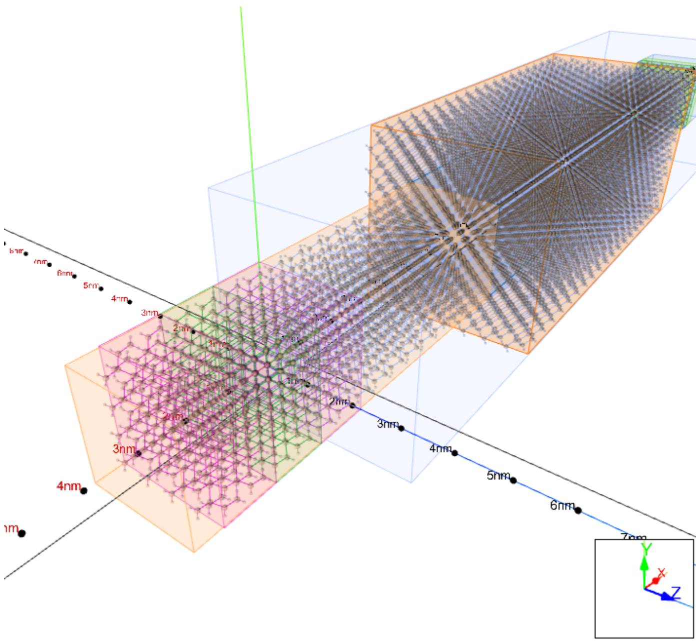


AND regions) using Nemo Server. Keep track of the colors of the boxes to identify the regions and the domains. For instance, the purple box represents the source_contact domain; the light orange box represents the source region. Take a moment to identify all regions and domains in the intersection view (figure below). You might notice that in the intersection view (domains AND regions) there are some atoms missing in the channel domain. This is because Nemo Server only visualizes the atoms that are going to be used in the Nemo5 simulation.

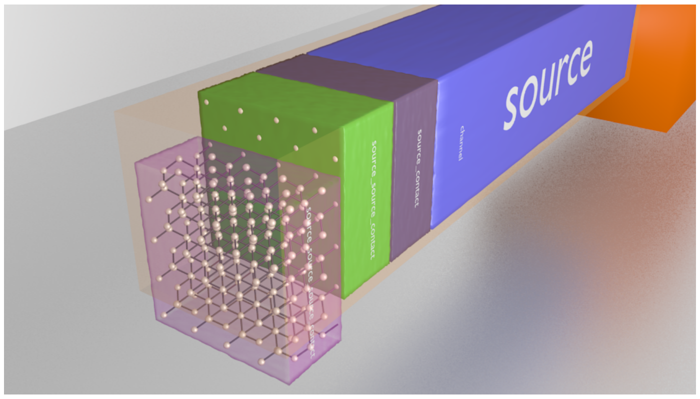
Now that you know how Nemo Server visualizes the devices from Nemo5 input-decks, lets talk about frequent errors in Nemo 5 input-decks, and then we will practice how to use Nemo Server to identify these errors.

Some of the most frequent errors that Nemo5 users face are SPATIAL DEFINITIONS. These errors refer to errors with domains positions, errors with domains size, and errors with geometrical shapes. Each of these errors would be illustrated below.

Domain position error: Number of atoms included in the domain box/unitcell is correct; however, the position of a domain is not aligned with its adjacent domains. The picture below shows an example of this type or error. In this example, there

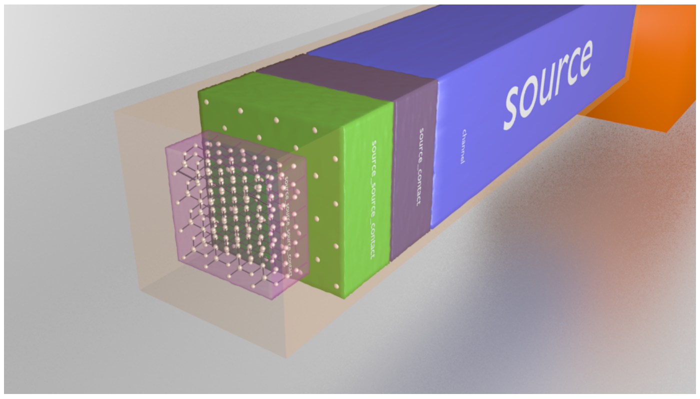


is a misalignment between the source_source_source_contact domain and its adjacent domain source_source_contact.

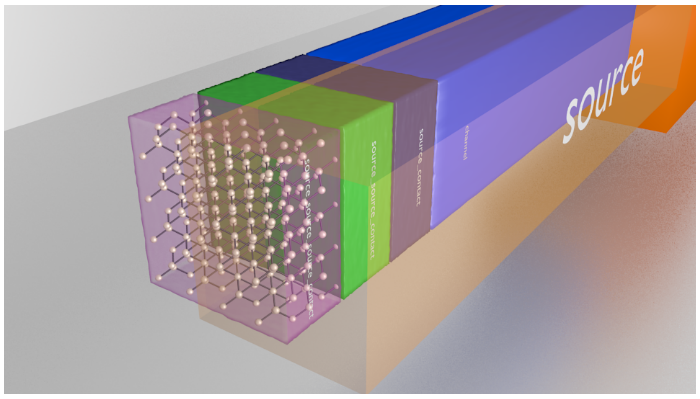


Domain size error: Number of atoms included in the domain box/unitcell is NOT correct. The picture below shows an example of this type or error. In this example, the source_source_source_contact domain has less number of atoms than its adjacent domain source_source_contact. As a result, it looks smaller than the adjacent domains. Notice that the position of the domain is correct, as it is centered in the source region.

Geometrical regions error: Geometrical regions are NOT aligned with domains; however, the number of atoms (size) and the position of the domains are correct.The



picture below shows an example of this type or error. In this example, all the domains are correctly defined, however, the source region is not aligned with the domains.

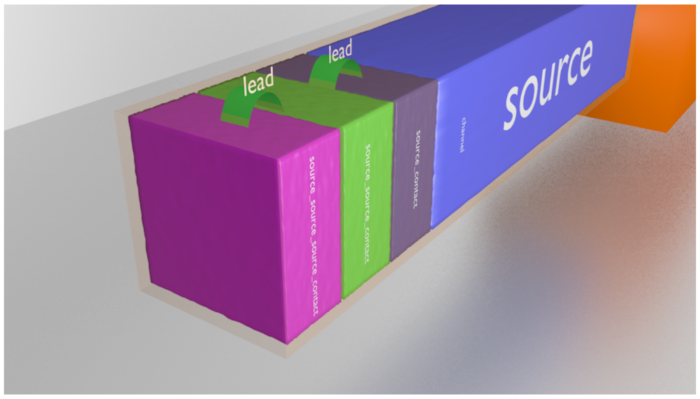


Errors due to SPATIAL DEFINITIONS can be detected if the user takes a careful look at Nemo Server's intersection view (regions AND domains). In the next pages, you will be presented with a series of training questions to practice how to identify errors in a Nemo5 input-deck by looking at visualizations generated by Nemo Server. Remember that all questions in this survey are based on a Nemo5 input-deck that simulates quantum transport across a silicon nanowire.

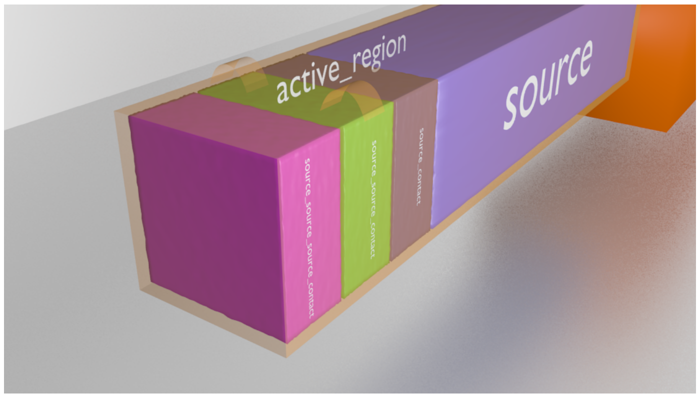
Abstract Representations

Another frequent set of errors that Nemo5 users face are SPATIAL RELATIONS. There are two main spatial relations mistakes that Nemo5 input-decks have: Domain-Domain and Domain-Region. Each of these errors is illustrated below.

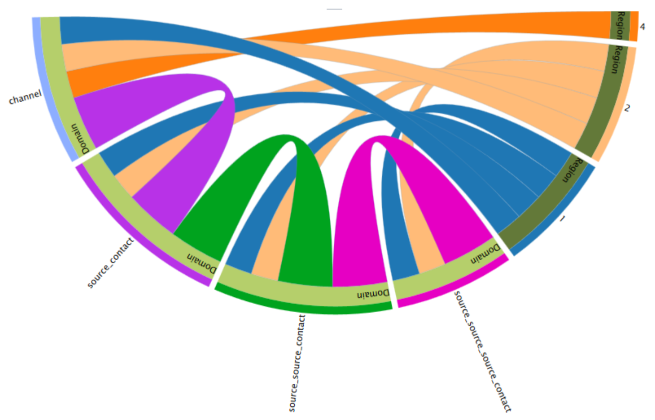
Domian-Domain: Domain is not connected with adjacent domains (leads). Each domain should be connected with its adjacent domains (also known as leads). if this relation is missing Nemo5 will complain and will have an unexpected behavior.



Domain-Region: Domain is not connected with correct regions: Domains should be connected with all regions it is contained, if this relation is missing Nemo5 will complain and will have an unexpected behavior

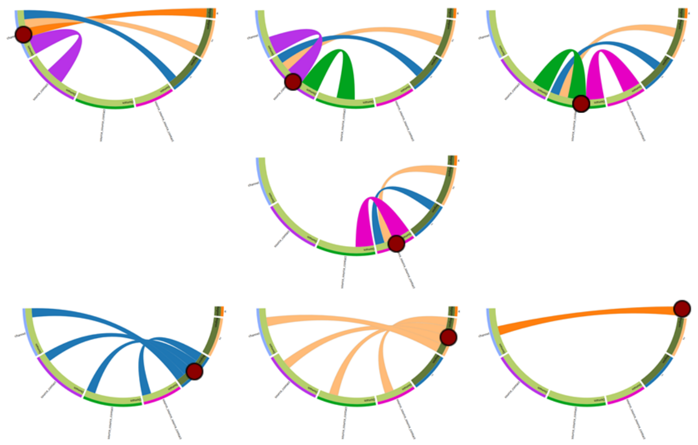


Nemo Server uses a chord diagram to visualize all the relations defined in a Nemo5 input-deck. The figure below shows all the relations defined in a Nemo5 input-deck for the simulation of a quantum transport across a silicon nanowire. Each block type (region or domain) has a color in the inner chord. For instance, domains are light green and regions are dark green. Each block in the input-deck has a specific color in the outer-chord . For example, the source_contact domain is purple and the channel region is blue. Notice that these colors correspond to the colors of the regions and domains in the 3D visualization. Nemo server shows the relations between blocks by connecting them with an arch. For instance, the pink arch in the figure below indicates that there is a relation between the source_source_source_contact and the source_source_contact.



The number of relations shown in the figure above could be overwhelming. This is why Nemo Server allows the user to only look at the relations of a single block when it is selected. The figure below shows multiple examples of the way that Nemo Server represent the relations, depending on the block that the user selects (the selected block is denoted by a red dot). Take a moment to understand the meaning of the representations in the figure below.

Nemo server visualization of relations is useful to detect SPATIAL RELATIONS errors, that is, Domain-Domain errors or Domain-Region errors. This kind of errors are easily identified because when a relation between blocks is missing in a Nemo 5 input-deck, the chord diagram generated by Nemo Server will be missing the arch that connect those two blocks.



APPENDIX B
NEMO5 GALLIUM ARSENIDE BAND-STRUCTURE CALCULATION INPUTDECK

```
Structure {
  Material {
    name = GaAs
    tag = substrate
    crystal_structure = zincblende
    regions = (1)
  }
  Domain {
    name = structure1
    type = pseudomorphic
    base_material = substrate
    dimension = (20,20,20)
    periodic = (true, true, true)
    regions = (1)
    crystal_direction1 = (1,0,0)
    crystal_direction2 = (0,1,0)
    crystal_direction3 = (0,0,1)
  }
  Geometry {
    Region {
      shape = cuboid
      region_number = 1
      min = (0,0,0)
      max = (5,5,5)
    }
  }
}
Solvers {
  solver {
    name = my_schroedi
    type = Schroedinger
    set {
      domain = structure1
      active_regions = (1)
      tb_basis = sp3d5sstar_SO
      job_list = (assemble_H, passivate_H, calculate_band_structure)
      output = (energies, eigenfunctions_VTK)
      charge_model = electron_hole
      automatic_threshold = true
      eigen_values_solver = krylovschur
      k_space_basis = cartesian
      k_points = [(0,0,0)]
    }
  }
  solver {
    name = my_overlap
    type = MatrixElements
    set {
      domain = structure1
      active_regions = (1)
      operator = overlap
      wf_simulation = my_schroedi
      output_file = matrix_elements
    }
  }
  solver {
    name = my_structure
    type = Structure
    set {
      domain = structure1
      active_atoms_only = true
    }
  }
}
Global {
   solve = (my_structure,my_schroedi,my_overlap)
   database = all.mat
}
```

APPENDIX C

EVALUATION MEASUREMENT TABLES

Table C.1.
Participants Expertise with Nemo5 and previous exposition to NemoViz

	Expertice	Nemoviz
U1	Expert	Yes
U2	Beginner	Yes
U3	Expert	Yes
U4	Intermediate	Yes
U5	Intermediate	No
U6	Beginner	No
U7	Expert	Yes
U8	Intermediate	Yes
U9	Expert	No
U10	Expert	Yes
U11	Expert	Yes
U12	Expert	Yes

Table C.2.
Number of correct answers for each user. Red values highlight values that were not taking into account due the empirical rule (z–score = $abs((x - \mu)/\sigma) > 3$). Q1: StrDim, Q2:StrPos, Q3:Sdef, Q4:SStrRel, Q5:StrStrRel

	User	Q1	Q2	Q3	Q4	Q5	Effective	Z–score	Total
Visual Input	U1	0	1	0	0	0	0.10	3.05	10
	U2	2	2	2	2	2	1.00	0.65	10
	U3	2	2	1	2	1	0.80	0.17	10
	U4	1	2	2	2	1	0.80	0.17	10
	U5	2	2	2	2	2	1.00	0.65	10
	U6	2	2	2	2	2	1.00	0.65	10
	U7	2	2	2	2	2	1.00	0.65	10
	U8	0	2	2	2	1	0.70	0.58	10
	U9	2	1	2	2	2	0.90	0.24	10
	U10	1	2	2	2	2	0.90	0.24	10
	U11	1	2	2	2	2	0.90	0.24	10
	U12	2	2	2	2	2	1.00	0.65	10
Textual Input	U1	0	1	1	1	1	0.80	0.17	5
	U2	1	1	1	1	1	1.00	0.85	5
	U3	0	1	1	0	1	0.60	1.18	5
	U4	0	0	1	1	0	0.40	2.20	5
	U5	1	1	1	1	1	1.00	0.85	5
	U6	0	1	1	1	1	0.80	0.17	5
	U7	1	1	1	1	1	1.00	0.85	5
	U8	1	0	1	0	1	0.60	1.18	5
	U9	1	1	1	1	1	1.00	0.85	5
	U10	1	1	1	1	1	1.00	0.85	5
	U11	1	1	1	1	1	1.00	0.85	5
	U12	1	1	1	0	1	0.80	0.17	5

Table C.3.
Times participants spend correctly answering a question. (G) Visual Based Input. (T) Textual Based Input

Time	**StrDim**		**StrPos**		**Sdef**		**SStrRel**		**StrStrRel**	
	G	T	G	T	G	T	G	T	G	T
T1	24.45	32.14	12.75	28.66	23.66	85.70	63.35	254.81	57.34	39.58
T2	14.40	55.11	12.93	121.74	18.63	51.48	39.39	181.31	28.27	94.48
T3	32.37	32.42	21.51	47.55	13.38	68.68	75.74	162.18	45.74	17.22
T4	19.40	38.23	18.12	23.87	66.25	73.00	11.27	74.32	13.05	55.15
T5	6.04	53.85	39.98	45.66	32.97	101.02	12.66	30.53	42.61	40.38
T6	3.99	24.21	39.61	128.01	71.32	67.23	16.79	125.95	30.19	14.86
T7	25.93	98.30	67.07	17.56	15.23	122.24	19.20	44.37	29.53	205.66
T8	37.27	13.31	24.76	27.23	64.64	205.28	21.87	38.24	25.05	28.94
T9	21.53		11.31	196.17	41.29	121.91	47.09	62.14	21.68	162.67
T10	8.68		31.41	51.93	6.48	62.43	12.08		6.16	85.69
T11	9.33		16.43		11.24	93.13	6.84		53.06	20.11
T12	40.90		4.61		9.07	43.28	138.46		19.58	
T13	59.87		7.50		7.63		28.94		79.94	
T14	79.94		6.65		10.59		70.65		37.15	
T15	82.37		22.13		63.24		27.10		149.12	
T16	15.12		8.60		32.86		12.19		158.56	
T17	4.69		23.53		11.68		21.95		117.79	
T18			37.03		40.25		42.83		21.30	
T19			16.47		12.92		20.92		40.74	
T20			19.93		23.02		31.39		4.18	
T21			17.86		5.96		13.75			
T22			4.57				3.85			
Mean	28.60	43.44	21.13	68.84	27.73	91.28	33.56	108.21	49.05	69.52
STD	23.94	24.51	14.44	56.38	21.39	42.00	30.28	73.29	43.19	60.20

Table C.4.
Z-score for each time participants spend correctly answering a question. Red values highlight outliers values removed during the analysis (based on the empirical rule)

Z–score	**StrDim**		**StrPos**		**Sdef**		**SStrRel**		**StrStrRel**	
	G	T	G	T	G	T	G	T	G	T
Z1	0.17	0.58	0.19	0.46	0.71	0.13	0.98	0.19	2.00	0.50
Z2	0.59	0.57	0.43	0.48	0.94	0.95	0.19	0.48	1.00	0.41
Z3	0.16	0.03	0.67	0.45	0.38	0.54	1.39	0.08	0.74	0.87
Z4	0.38	0.21	1.80	0.21	0.80	0.44	0.74	0.83	0.46	0.24
Z5	0.94	1.31	0.25	0.42	0.41	0.23	0.69	0.15	1.06	0.48
Z6	1.03	1.28	2.04	0.78	1.05	0.57	0.55	0.44	0.24	0.91
Z7	0.11	3.18	0.58	2.24	0.91	0.74	0.47	0.45	0.87	2.26
Z8	0.36	0.25	1.73	1.23	0.74	2.71	0.39	0.56	0.95	0.67
Z9	0.30	0.68	0.63		2.26	0.73	0.45	0.63	0.63	1.55
Z10	0.83	0.71	0.99		0.30	0.69	0.71	0.99		0.27
Z11	0.81	0.33	0.77			0.04	0.88	0.09		0.82
Z12	0.51	1.14	0.87			1.14	3.46	0.68		
Z13	1.31	0.94	0.94				0.15	0.72		
Z14	2.14	1.00	0.80				1.22	0.28		
Z15	2.25	0.07	1.66				0.21	2.32		
Z16	0.56	0.87	0.24				0.71	2.54		
Z17	1.00	0.17	0.75				0.38	1.59		
Z18		1.10	0.59				0.31	0.64		
Z19		0.32	0.69				0.42	0.19		
Z20		0.08	0.22				0.07	1.04		
Z21		0.23	1.02				0.65			
Z22		1.15					0.98			

APPENDIX D
INTERACTIVE ANALYTIC SYSTEMS FOR UNDERSTANDING THE SCHOLARLY IMPACT OF LARGE-SCALE E-SCIENCE CYBER ENVIRONMENTS

The following text in this chapter was published as a short paper in the proceedings of the ESCIENCE2015 (2015 IEEE International Conference on e-Science) [35], this paper describes the interactive analytic system used to measure Nanohub.org impact.

Introduction

The advent of Internet and Web 2.0 has given rise to the emergence and popularity of cyber-environments. A cyber-environment is defined as a collection of computational resources, data, visualization resources made available through an online portal [36], supported by underlying network, services, software, and hardware [37]. The academic use of cyber-environments helps disseminate educational tools, scientific workflow/simulations, academic publications, and other resources to benefit a much wider range of audience. For example, as a leading cyber-environment in nanotechnology, nanoHUB [38] has served about 310,000 users over the past 12 months with over 4,000 presentations, teaching materials, simulation tools, and other nanotechnology-related resources as of today [39]. Another renowned cyber-environment, PhET [40], is a virtual lab environment that offers over 75 million science simulation tools for researchers, educators, and students. Last but not least, Molecular Workbench [41] is an online platform that facilitates sharing of molecular dynamics simulations primarily for educational purposes [42]. As more cyber-environments produce significant impacts on their intended communities, it is critical to precisely measure and demonstrate their scholarly performance. There is a large body of literatures discussing what methods to use for evaluating research quality and how to apply these methods in a specific context. Among all these assessment methods, bibliometrics is the most widely used method, which evaluate research quality by quantitative analyses of scientific publications. Tremendous prior studies have used bibliometrics to demonstrate the scholarly impacts of theories [43], journals [44], [45], research areas [46] [47], and countries [48]. The long history and popularity of applying bibliometrics in research evaluation indicates the potential of using it to evaluate cyber-environment. Assessing a cyber-environment with bibliometric data, however, places additional challenges that are not commonly encountered in the aforementioned prior studies. First, analyses of bibliometric data often have a clearly defined data source to draw publications from and a programmatic sampling strategy to narrow down the dataset. For example, it is a common practice for a bibliometric study to analyze papers that contain specific keywords from certain journals over a period of time. On the contrary, the scholarly impact of a cyber-environment is demonstrated not only by academic publications produced by the core team members who develop the platform, but also by those published by users who utilize resources in the cyber-environment. For example, a researcher contributes a simulation tool to the cyber-environment and studies how the tool is adopted and used by other registered users. Nevertheless, these authors rarely report their academic work built upon the cyber-environment facilities and resources back to the online community. Also, users may publish their work in a diversity of journals and conferences and therefore publication venues cannot be determined easily. Finally, they may not cite the cyber-environment as a reference and instead, only mention it in the footnote or acknowledgement. All these factors make it extremely difficult to keep track of all the citations of a cyber-environment. Second, due to the uncertainty and diversity of publications, the data quality is radically compromised. The majority of bibliometric studies have bibliographic metadata drawn from only a few data sources. Therefore the data tend to be consistent and suffer less from problems like name ambiguity, missing data fields, and missing full text. Cyber-environment citations, however, are acquired from a large variety of publishers and indexing engines. The citation data collected need to be carefully handled to make sure that the data quality is appropriate for the analysis. Third, most bibliometric analysis results are published once every few years and the most recent publications cannot be factored in until the next analysis. While this may not sound like a problem in some scenarios, it is critical for cyber-environments to gain immediate feedbacks from the community users to improve the services. For example, if the results show that a particular simulation tool has gained a lot of attentions from scholars, this may be an indication that the tool is being frequently used and require much more computational resources. Last but not least, the diversity of cyber-environment audience implies the need of delivering the analysis results in multiple facets. The funding agency may be interested in different statistics from a hub user in the evaluation result. There is not a single template for reporting analysis results that can suit all the needs and the audience should be granted more freedom to navigate the results. This means, the traditional static representations need to be transformed into an interactive form. This paper presents our attempt to demonstrate the scholarly impact of a cyber-environment based on bibliometric data. In particular, we aim to address the four challenges above by answering the following questions: 1. How to define the appropriate sample scope and increase data quality in evaluating cyber-environments scholarly impact? 2. How to keep the evaluation results up-to-date and allow audience to interact with them? To answer the first question, we present our workflow and implementation of a web-based citation management system. The system facilitates metadata collection, data quality control, and metadata annotation in managing citations of a cyber-environment. For the second question, we construct a public data gateway with interactive visualizations and statistics to offer users the capability of interacting with the bibliometric data. To show the effectiveness of the system, we apply it to showcase nanoHUBs scholarly impact. To our best knowledge, the present study is the first attempt to characterize an ill-defined bibliometric dataset and represent it in an interactive and visual form. The system we propose in this study can be used to evaluate a broad range of other cyber-environments. It also has the potential in other contexts where sample scope cannot be easily defined.

Related Work

Research evaluation

Research evaluation can be performed via a number of different ways. Some organizations, such as the National Science Foundation in the US, depend on internal and external evaluators to conduct and report evaluations of funded projects [49]. These evaluators are usually domain experts that have sufficient knowledge and experience to perform a comprehensive review [50]. However, the evaluators are either permanently employed or contracted to perform the tasks and are often offered training programs to get prepared for the job. Therefore it is a costly approach and is most appropriate for comprehensive evaluations of a very limited number of objects infrequently. Also, there are debates questioning the independency, objectivity, knowledge, credibility, and ethic of evaluators [50]. Also, evaluators vary in their competencies and in many organizations there is no widely accepted taxonomy of what should be considered as essential evaluator competencies [51]. This may lead to inconsistent evaluation results produced by different evaluators, which make it difficult to compare results across programs/projects. Besides the human-based approach, scientists and government agencies have also sought for a more data-driven solution. The popularity of digital resources online has transformed academic publishing and made data-driven solutions technologically feasible [52]. Data often refer to the bibliographic data, which includes a wide range of formally written publications in academia such as journal papers, conference proceedings papers, books, grant proposals, and other communication medium. Bibliometrics is defined as the quantitative study of physical published units, or of bibliographic units, or of the surrogates for either [53]. Since it was first coined in 1969 [54] bibliometrics has sustained the mainstream in data-driven research evaluation. Therefore, we will discuss bibliometrics in greater depth in the next section. Another recent strand of data driven research evaluation is altmetrics, also known as Scientometrics 2.0 [55]. Rather than depending on academic articles, altmetrics focuses on evaluating academic work based on social media data, measured by web-based metrics such as number of social bookmarking [56], mention in microblogging platforms [57], and occurrences in social networking applications [55]. To date, altmetrics is still in its infancy and has not been widely adopted in research evaluation. Therefore, our approach selects bibliometrics as the primary toolkit for measuring research quality. Both bibliometrics and altmetrics are capable of analyzing large-scale quantitative data and require radically less human efforts than depending on evaluators.

Bibliometrics

Bibliometrics is a prevalent quantitative method used not only for assessing academic performance but also for demonstrating the evolution of a research community. Citation analysis and content analysis are the two most popular methods in bibliometrics. Citation analysis refers to the examination of the frequency, patterns, and graphs of citations in articles and books [58]. It is widely used to evaluate and compare journal impacts of a given research area [44], [46], [59], examine past governmental investments [60], [61], showcase institutes academic contributions to a field [62], [63], and compare geographical patterns in co-citation and co-authorship [64]. However, some scholars questioned [65], [66] and opposed [67] the use of citation analysis as a quality evaluation tool because it fails to take other factors into consideration and hence may yield misleading results. For instance, Sims and McGhee identified factors that might threaten the validity of the citation analysis: fields of study, inconsistencies in calculation due to manual annotation, inaccuracies in database processing, bias against non-English language journals, self-citation, and time taken to review manuscripts [59]. Content analysis, in the context of bibliometrics, aims to study publications in greater depth and more descriptively than citation counting. In combination with citation analysis, some scholars define content analysis as the study of contexts in which citations occur [68] (also known as citation function [69]). In such cases, content analysis supplements citation analysis with more contexts for citations other than simple counts. The core of content analysis studies focus on proposing classification schemes for different citation types such as affirmative and negational [70]. Rather than focusing on contextualizing citations, another line of research attempts to analyze attributes that are not immediately available from the basic metadata. Instead, it often requires efforts from domain experts or algorithms to annotate articles with supplemental information. The additional descriptions generated manually by human tend to be more insightful and purposeful and are used to classify literatures into categories. As a result, statistics are presented regarding the intra-categorical status and inter-categorical connections. For instance, scholars aim to understand a large body of literature by study type such as comparative study, descriptive study, and usability testing [71] and by topics [72]. The supplemental information can be also derived from full text by algorithms automatically. For example, the additional annotation extracted programmatically from full text can help characterize topical trends in a domain [73]. Some researchers combine these two methods to reveal the main theme in a research area by keyword co-occurrence [74] Regardless of whether it is a citation analysis or a content analysis, analysis results of bibliometric data are often represented as a formal document such as an evaluation report and an academic paper. In such cases, it is up to the authors what to report, how to report, and how often a report is published. It leaves little room for the audience to freely explore the bibliometric data. Also, this traditional publishing process inevitably incurs a lag in time, where the results written in the document are already obsolete when they are published. There is a need for transformative changes in how bibliometric analysis results are presented.

Methodology

Demonstrating the scholarly impact of cyber-environments has the difficulties of defining sample scopes, increasing data quality, keeping results up-to-date, and allowing user navigation. The first two difficulties can be overcome by a citation management system, which is a web-based platform we develop for managing a high-quality bibliographic database. The last two difficulties require development of a user interface for visually monitoring and interacting with the most up-to-date scholarly impact. In response, we design a workflow illustrated in Figure D.1 to tackle these problems. Figure D.1 presents the two main components in our design: citation management and visual analytics. In citation management, E-team refers to an editor team that works via the web interface on adding and managing new citations relevant to the specific cyber-environment. Part of the citation information comes directly from the data sources defined by the E-team, whereas supplemental information is added to each citation manually later. Any change made by the E-team will be saved into a bibliographic database. Those flagged as approved in the database are used to produce statistics and visualizations, which are available in the public view. Potential users such as project directors, evaluators, and hub users can interact with the presented data to explore them in different aspects.

Citation management

The citation management module aims to provide a set of services for importing and managing citation data with the E-team involved. For a new citation to become valid and complete, it must go through a sequence of processes: Bibliographic metadata acquisition, Full text download, Bibliographic metadata correction

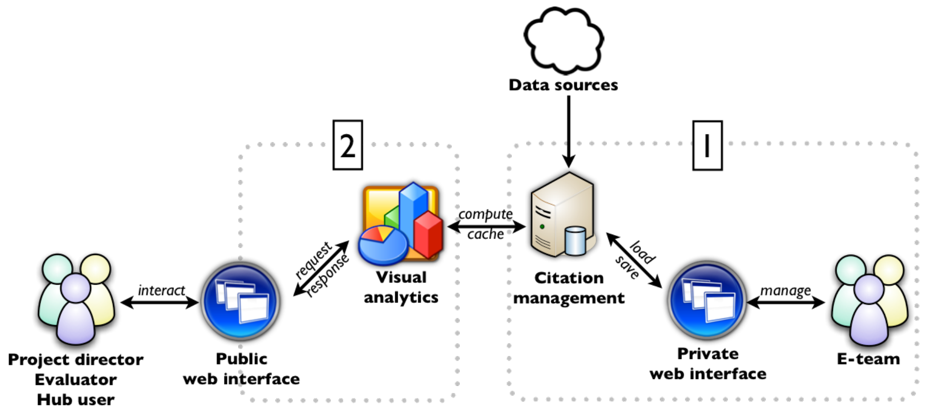
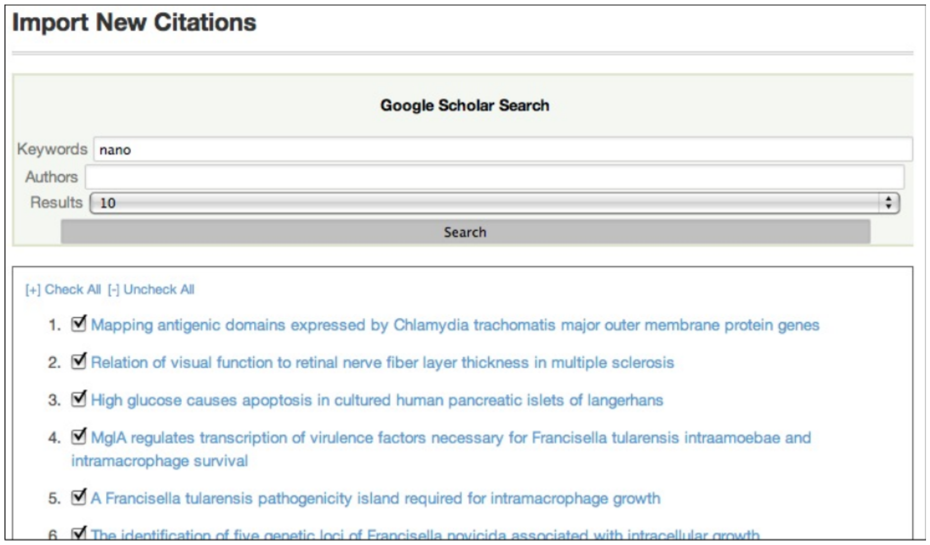


Fig. D.1. The general workflow of our system for demonstrating the scholarly impact of cyber-environments

and completion, Name disambiguation, Supplemental annotations, Final review and publications.

1. Bibliographic metadata acquisition: There are two major challenges in acquiring bibliographic metadata of publications that are relevant to the given cyber-environment. First, the new citations must be relevant to the cyber-environment and the definition of relevance may vary over time. This requires creation of a list of data sources and development of corresponding processors to detect and parse the new data. For instance, based on observations of where the past publications that cite the cyber-environment resources are archived, the acquisition step may rely on metadata processors to download and extract bibliographic data from Web of Science, Google Scholar, and EBSCOhost. Different sources vary in the way metadata are downloaded and in the format the data is presented. For example, indexing engines such as Microsoft Academic Search and Web of Science offers web API such as web services and JSON-RPC for querying their databases. Some provides the option of file download in BibTex, EndNote, and RIS formats. Others such as Google Scholar have no infrastructure to facilitate a batch download and therefore a webpage crawl needs to be developed to mine information from their sites. Also, administrators without any programming experience should be able to modify the sampling criteria to retrieve new bibliographic metadata. Figure D.2 shows an example of finding new citations using Google Scholar with the keyword nano.

   Second, a regular routine must be set up to update the citation database so as to keep it always up-to-date. The downloader and extractor mentioned above run as a daemon program to pull new citations from the specified data sources and insert them into the pending queue for further processing. During the data acquisition step, the E-team decides whether to exclude certain citations from the list based on the publication titles. The blacklisted record is saved to prevent the same item from appearing in the future. However, the title per se is sometimes insufficient for making a decision about the citations relevance. In



Import New Citations

Google Scholar Search

Keywords nano

Authors

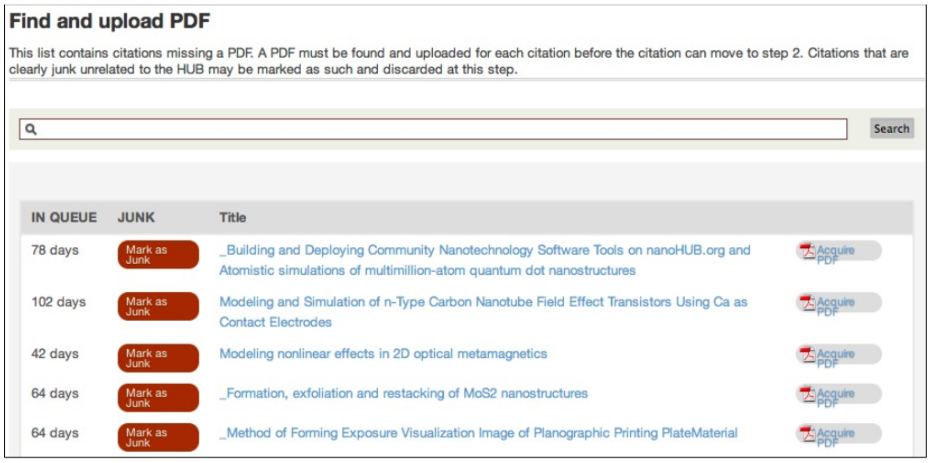
Results 10

Search

[+] Check All [-] Uncheck All

1. ☑ Mapping antigenic domains expressed by Chlamydia trachomatis major outer membrane protein genes
2. ☑ Relation of visual function to retinal nerve fiber layer thickness in multiple sclerosis
3. ☑ High glucose causes apoptosis in cultured human pancreatic islets of langerhans
4. ☑ MglA regulates transcription of virulence factors necessary for Francisella tularensis intraamoebae and intramacrophage survival
5. ☑ A Francisella tularensis pathogenicity island required for intramacrophage growth
6. ☑ The identification of five genetic loci of Francisella novicida associated with intracellular growth

Fig. D.2. The web interface for importing bibliographic metadata from Google Scholar



Find and upload PDF

This list contains citations missing a PDF. A PDF must be found and uploaded for each citation before the citation can move to step 2. Citations that are clearly junk unrelated to the HUB may be marked as such and discarded at this step.

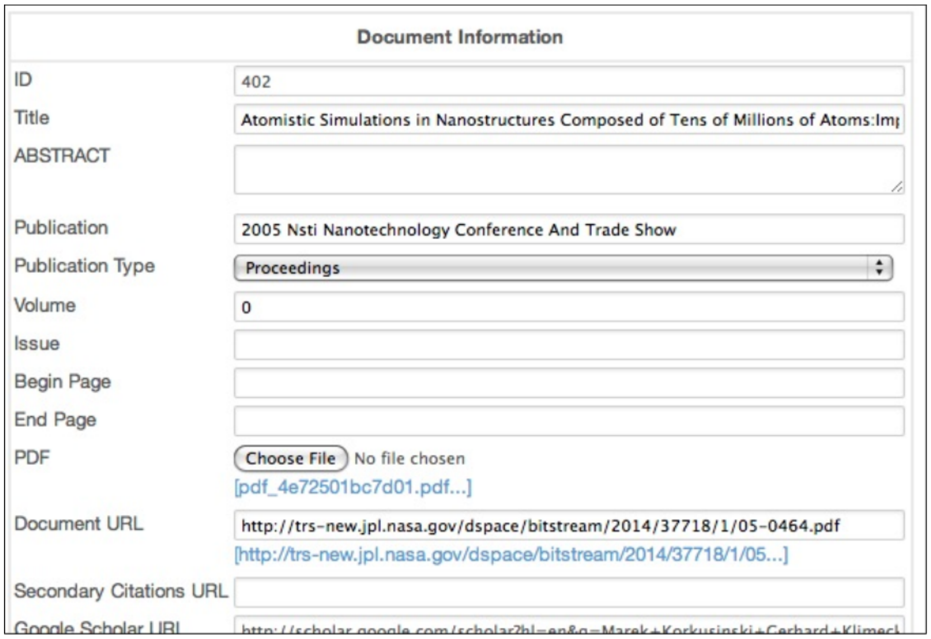
IN QUEUE	JUNK	Title	
78 days	Mark as Junk	_Building and Deploying Community Nanotechnology Software Tools on nanoHUB.org and Atomistic simulations of multimillion-atom quantum dot nanostructures	Acquire PDF
102 days	Mark as Junk	Modeling and Simulation of n-Type Carbon Nanotube Field Effect Transistors Using Ca as Contact Electrodes	Acquire PDF
42 days	Mark as Junk	Modeling nonlinear effects in 2D optical metamagnetics	Acquire PDF
64 days	Mark as Junk	_Formation, exfoliation and restacking of MoS2 nanostructures	Acquire PDF
64 days	Mark as Junk	_Method of Forming Exposure Visualization Image of Planographic Printing PlateMaterial	Acquire PDF

Fig. D.3. The web interface for importing bibliographic metadata from Google Scholar

the following steps, the E-team will be offered opportunities to remove irrelevant citations.

2. Full text download: Content analysis has to be performed over publication full text. When the data sources provide full text download, the metadata downloader mentioned earlier automatically retrieves the file and associates it with the corresponding metadata in the database. However, full text may sometimes be missing from the data sources. In such cases, the full text has to be downloaded by the E-team from other data sources that may not be defined before. It may also happen that the full text cannot be found anywhere online. Figure D.3 illustrates the webpage for associating full text with new citations. In our implementation, we place an indicator of how long an item remains in the pending queue. An item that remains unresolved for a long time usually implies the unavailability of full text.

3. Bibliographic metadata correction and completion: The bibliographic metadata acquired from the data sources in many cases contain incomplete and even incorrect data. For instance, a journal paper may miss the publication year or a book may have incorrect publisher information. Such errors significantly



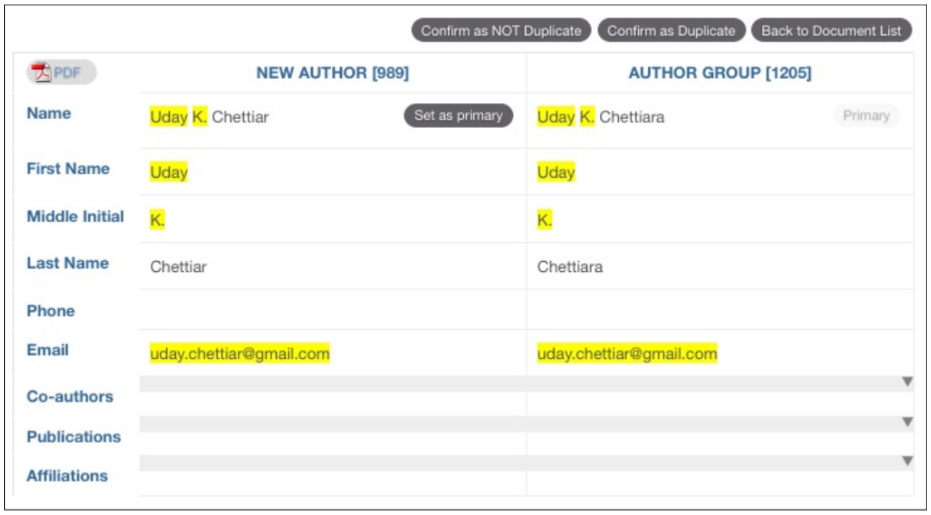
Document Information

ID	402
Title	Atomistic Simulations in Nanostructures Composed of Tens of Millions of Atoms:Imp
ABSTRACT	
Publication	2005 Nsti Nanotechnology Conference And Trade Show
Publication Type	Proceedings
Volume	0
Issue	
Begin Page	
End Page	
PDF	Choose File No file chosen [pdf_4e72501bc7d01.pdf...]
Document URL	http://trs-new.jpl.nasa.gov/dspace/bitstream/2014/37718/1/05-0464.pdf [http://trs-new.jpl.nasa.gov/dspace/bitstream/2014/37718/1/05...]
Secondary Citations URL	
Google Scholar URL	http://scholar.google.com/scholar?hl=en&q=Marek+Korkusinski+Gerhard+Klimeck

Fig. D.4. The web interface for correcting and completing citation metadata

threaten the validity of the analysis results. To solve this problem, we build a user interface to facilitate the E-team to input data for the missing fields and provide a link to the full text for reference, as shown in Figure D.4.

4. Name disambiguation: Name ambiguity problem is a common problem in bibliographic database [75]. It refers to cases where one individual is represented as various names and different individuals share the same name. Taking author names as an example, an author may publish papers under different name variations caused by first name abbreviation, middle initial omission, and even typos [76]. On the other hand, the same name may represent more than one scholar. Failure to identify such relationships incurs errors in studying author collaboration and also affects other author-related statistics. Existing approaches on author name disambiguation can be classified as either supervised or unsupervised solutions. Supervised approaches [77], [78] involve human in the decision-making process and are generally believed to produce higher-quality result. Unsupervised solutions [79] [80] depend entirely on algorithms to detect

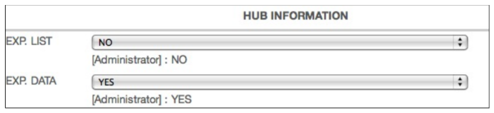


	NEW AUTHOR [989]	AUTHOR GROUP [1205]
Name	Uday K. Chettiar	Uday K. Chettiara
First Name	Uday	Uday
Middle Initial	K.	K.
Last Name	Chettiar	Chettiara
Phone		
Email	uday.chettiar@gmail.com	uday.chettiar@gmail.com
Co-authors		
Publications		
Affiliations		

Fig. D.5. The web interface for correcting and completing citation metadata

name duplicates automatically and are capable of handling large-scale data. Because the number of citations of cyber-environments is relatively small and data precision is more critical, we choose supervised solutions by detecting suspected duplicates and offering users visual aids in the disambiguation process. Figure D.5 shows a pair of suspected duplicate names with identical information highlighted. Similar to author name ambiguity, publications also undergo a similar problem. Due to the inconsistency of data fields between two publications, two actually identical documents may be mistakenly viewed as separate ones. We apply a similar approach to compute the similarity of two citations by their titles, abstracts, publication years, authors, and publication venues.

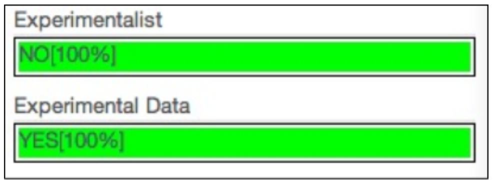
Supplemental annotations: The bibliographic metadata provide basic information of a publication such as title, abstract, author, keyword, publication venue, and publication year. However, content analysis often requires more insightful and elaborative data, which are most likely to be produced by human. Depending on the nature of the cyber-environment and the purpose of the research evaluation, additional annotations may include the study type, related cyber-environment resources, population studied, and sample size. However, the E-team may not agree on how to annotate a citation and therefore, we develop a voting mechanism and ensure that each citation has at least three editors annotations. Based on the aggregated results, the E-team leader decides the final annotation. Figure D.6 demonstrates the process of voting and the result.


HUB INFORMATION

EXP. LIST: NO
[Administrator] : NO

EXP. DATA: YES
[Administrator] : YES

(a)


Experimentalist
NO[100%]

Experimental Data
YES[100%]

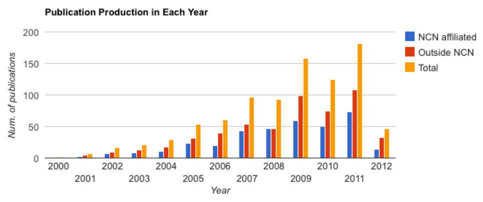
(b)

Fig. D.6. The web interface for (1) annotation and (2) showing the voting result.

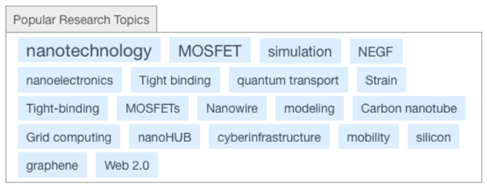
Final review: In the last stage of the citation management process, the E-team leader reviews the bibliographic metadata and supplemental annotations and approves the citation if all seems appropriate. If there is a problem, he/she can rewind the citation to an earlier stage or even drop it. Once a citation is finally approved, it enters the pool of published data, based on which analyses and visual representations are created.

Visual analytics

The goal of the visual analytics module is to present the analytics visually and allow any user to freely explore the up-to-date citation data. Traditionally, the scholarly impact is documented as reports and publications in which the authors can select what and how to report. Instead of restricting audience to what they can see, we offer many alternatives of looking at the bibliographic data and let audience decide what they prefer to view. To achieve this goal, we develop a web portal with various visualizations and statistics for citations approved in the citation management process.



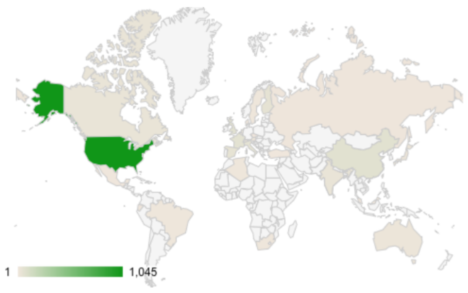
(a)



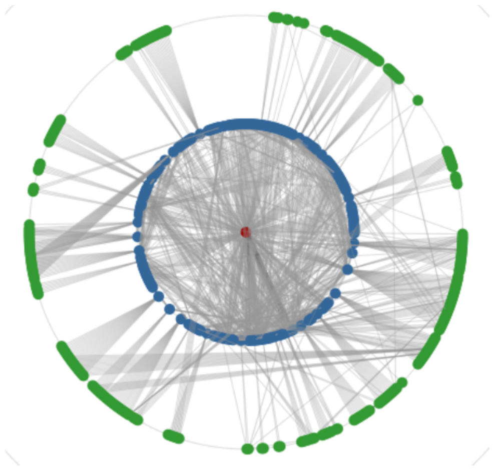
Popular Research Topics

nanotechnology MOSFET simulation NEGF
nanoelectronics Tight binding quantum transport Strain
Tight-binding MOSFETs Nanowire modeling Carbon nanotube
Grid computing nanoHUB cyberinfrastructure mobility silicon
graphene Web 2.0

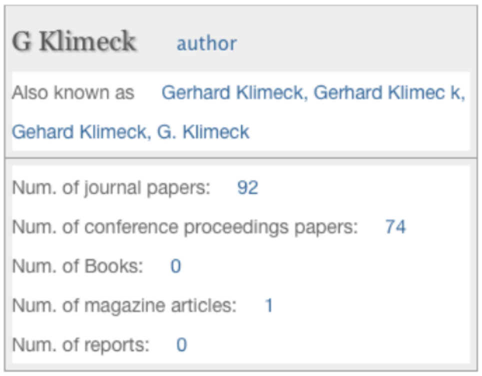
(b)



(c)



(d)



G Klimeck author

Also known as Gerhard Klimeck, Gerhard Klimec k, Gehard Klimeck, G. Klimeck

Num. of journal papers: 92

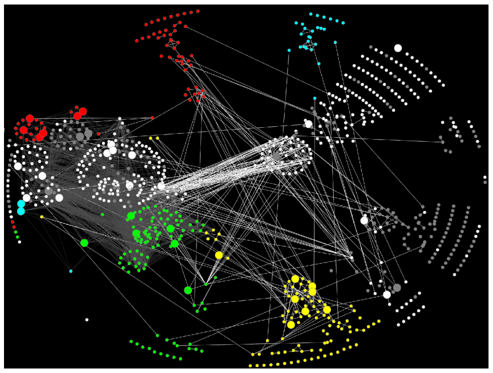
Num. of conference proceedings papers: 74

Num. of Books: 0

Num. of magazine articles: 1

Num. of reports: 0

(e)



(f)

Fig. D.7. Some of the visualizations and statistics available on the impact demonstration site.

Figure D.7 lists some of the visualizations and statistics available for the users to navigate. Figure D.7(a) presents the number of citations in each year; D.7(b) shows the main research topics among all the citations; D.7(c) demonstrates the geographical distribution of the authors; D.7(d) is the collaboration network with two degree of separations of a scholar; D.7(e) shows an authors academic profile; and D.7(f) illustrates how citations are linked by common authors. Some of the visualizations involve intensive computation and have to be executed in a distributed environment such as Condor [81] and Hadoop [82] clusters. The technical details of how to implement each visualization are beyond the scope of this paper.

Case Study: Nanohub

To demonstrate the usefulness of the workflow and implementations we propose, we select a cyber-environment called nanoHUB and study its scholarly impact. nanoHUB is a resource hub for nanotechnology education and research and aims to promote resource sharing and user collaboration. Over the past 12 months, it serves over 310,000 users worldwide who add a large number of new scientific resources. nanoHUB is selected in this study because it is a great example of cyber-environments in academia with a long history and significant impact on many research and education communities. Before our system was introduced, the nanoHUB editor team collected and filtered citations manually on a timely basis. The team then compiled a list of new citations in an Excel spreadsheet and sent it over to the database administrator, who inserted and maintained the citation database. When there was a demand for demonstrating nanoHUBs scholarly impact, the database administrator along with other visualization designers created statistics and graphs using software such as Pajek [83] and NetDraw [84]. It was not only a costly procedure but also led to many problems. For example, the editor team found it difficult to coordinate and sometimes ended up working on including the same citation multiple times on the spreadsheet. The database administrator encountered the problem of name disambiguation and spent a lot of time resolving them case by case. The statistics and visualization producers were tired of repeating similar tasks every time when the report was demanded. We deploy our system using the cyber-infrastructure provided by nanoHUB and treat citations in the legacy database as new citations. In our case study, the E-team is composed of one domain expert (team leader) and four undergraduate students with no domain knowledge. A 30-minute training session is provided to the E-team to learn the new system. From then on, the E-team works individually on the web-based interface to collect, correct, complete, and annotate bibliographic data of nanoHUB citations. The E-team leader reviews other team members efforts and the approved citations are presented as statistics and visualizations on the public site. The visual products are also included as part of the annual report submitted to the funding agency. By 3/27/2015, a total of 1,740 citations have been acquired from Google Scholar, out of which 281 are identified as irrelevant to nanoHUB or duplicate to existing citations. 1210 citations have been approved with complete metadata, full text, and annotations. The rest are to be processed in the citation management system. Among the published citations, 1760 author names are identified as ambiguous, out of a total of 4354 author names. The supplemental annotations for nanoHUB citations are: (1) Whether the research project is NCN-affiliated; (2) Whether the research study contains experimentalist/experimental data; (3) What tools on nanoHUB are cited; and (4) What type of study it belongs to.

Discussion

We select cyber-environments as a stereotype to show the effectiveness of the workflow we propose. However, our solution is highly flexible and configurable so that it can be applied to other similar scenarios. The intended use of our approach is to showcase from many different perspectives the scholar impact of a relatively small, ill-defined, constantly growing dataset where data precision and real-time updates are of high priority. For example, our solution can also be used to show the evolution of an emerging discipline that has no dedicated journals or conferences and is on the way of forming its unique knowledge body. It can also be used to characterize the impact of a renowned theory in multiple domains. Besides web-based solutions, there are other alternative platforms for visualizing bibliometric data in an interactive way. In general, standalone applications such as ClaiMapper [85] have the advantage over web applications in performance. However, we deploy the computationally intensive components to a distributed environment and show the cached content to the users. Therefore, the overall performance exceeds that of a standalone application in some cases. Also, web-base platforms require no software installation and are highly accessible from any computer with Internet connections. Therefore, we choose to develop a web-base data gateway for managing and presenting bibliometric data. One major limitation of our current design is the scalability problem in citation management. As discussed earlier, automating the citation management process can be achieved by adopting the unsupervised name disambiguation algorithms and overlooking missing data fields. However, automation compromises data quality and eventually affects the accuracy of analysis results. To sustain high data precision while lowering the cost of recruiting a dedicated team, we would like to explore in a future study the possibility of deploying our implementations on crowdsourcing platforms such as Amazon Mechanical Turk. Based on our observation of the E-team working on the nanoHUB dataset, we believe that it is feasible to crowdsource the tasks of downloading full text, correcting and completing metadata, and disambiguating names to ordinary novice users

REFERENCES

[1] M. Lundstrom, “Drift-diffusion and computational electronics - still going strong after 40 years!” in *2015 International Conference on Simulation of Semiconductor Processes and Devices (SISPAD)*, Sep. 2015, pp. 1–3.

[2] U. Wilensky and M. Resnick, “Thinking in Levels: A Dynamic Systems Approach to Making Sense of the World,” *Journal of Science Education and Technology*, vol. 8, no. 1, pp. 3–19, Mar. 1999. [Online]. Available: https://doi.org/10.1023/A:1009421303064

[3] C. E. Hmelo-Silver and R. Azevedo, “Understanding Complex Systems: Some Core Challenges,” *Journal of the Learning Sciences*, vol. 15, no. 1, pp. 53–61, Jan. 2006.

[4] R. K. Atkinson, S. J. Derry, A. Renkl, and D. Wortham, “Learning from Examples: Instructional Principles from the Worked Examples Research,” *Review of Educational Research*, vol. 70, no. 2, pp. 181–214, Jun. 2000. [Online]. Available: http://rer.sagepub.com/content/70/2/181

[5] J. Wiley, “Expertise as mental set: The effects of domain knowledge in creative problem solving,” *Memory & Cognition*, vol. 26, no. 4, pp. 716–730, Jul. 1998. [Online]. Available: http://link.springer.com/article/10.3758/BF03211392

[6] M. llinger, G. Jones, and G. Knoblich, “Investigating the Effect of Mental Set on Insight Problem Solving,” *Experimental Psychology*, vol. 55, no. 4, pp. 269–282, Jan. 2008. [Online]. Available: http://econtent.hogrefe.com/doi/abs/10.1027/1618-3169.55.4.269

[7] M. Scaife and Y. Rogers, “External cognition: how do graphical representations work?” *International Journal of Human-Computer Studies*, vol. 45, no. 2, pp. 185–213, Aug. 1996. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S1071581996900488

[8] W. O. Galitz, *The Essential Guide to User Interface Design: An Introduction to GUI Design Principles and Techniques.* John Wiley & Sons, Apr. 2007.

[9] J. J. Padilla, S. Y. Diallo, and A. Tolk, “Do We Need M&S Science?” *SCS M&S Magazine*, vol. 8, no. 2011, pp. 161–166, 2011. [Online]. Available: http://www.scs.org/magazines/2011-10/index_file/Files/Padilla-Diallo-Tolk.pdf

[10] S. Krger, *Simulation: Grundlagen, Techniken, Anwendungen.* Berlin ; New York: De Gruyter, Jan. 1975.

[11] M. P. Allen and D. J. Tildesley, *Computer Simulation of Liquids*, reprint edition ed. Clarendon Press, Jun. 1989.

[12] I. Romanowska, "So You Think You Can Model? A Guide to Building and Evaluating Archaeological Simulation Models of Dispersals," *Human Biology Open Access Pre-Prints*, Nov. 2015. [Online]. Available: http://digitalcommons.wayne.edu/humbiol_preprints/79

[13] "Dojo Toolkit." [Online]. Available: http://dojotoolkit.org/

[14] "three.js - Javascript 3d library." [Online]. Available: http://threejs.org/

[15] M. Bostock, "D3.js - Data-Driven Documents." [Online]. Available: https://d3js.org/

[16] B. Karlsson, *Beyond the C++ Standard Library: An Introduction to Boost*. Pearson Education, Aug. 2005, google-Books-ID: lFfuYJ0OeZkC.

[17] Lan Zhao, Carol X. Song, Rajesh Kalyanam, Larry Biehl, Robert Campbell, Leif Delgass, Derrick Kearney, Wei Wan, Jaewoo Shin, I Luk Kim, and Carolyn Ellis, "GABBs - Reusable Geospatial Data Analysis Building Blocks for Science Gateways," in *9th International Workshop on Science Gateways*, Jun. 2017.

[18] A. Strachan, "Reproducing DFT calculations of Al2o3/GaAs interface structure and Fermi level pinning," 2015.

[19] R. Andrawis, J. D. Bermeo, J. Charles, J. Fang, J. Fonseca, Y. He, G. Klimeck, Z. Jiang, T. Kubis, D. Mejia, D. Lemus, M. Povolotskyi, S. A. P. Rubiano, P. Sarangapani, and L. Zeng, "NEMO5: Achieving High-end Internode Communication for Performance Projection Beyond Moore's Law," *arXiv:1510.04686 [physics]*, Oct. 2015, arXiv: 1510.04686. [Online]. Available: http://arxiv.org/abs/1510.04686

[20] G. Klimeck, "Accelerating Nanoscale Transistor Innovation with Nemo5," *Blue Waters Highlight*, 2015.

[21] J. Fonseca, H. Sahasrabudhe, E. Wilson, S. Mehdi, and G. Klimeck, "NEMO5 NanoElectronics MOdeling," *Blue Waters Symposium*, 2014.

[22] J. Fonseca, "NEMO5 on Blue Waters - A Flexible Package for Nanoelectronics Modeling Problems," *Blue Waters Symposium*, 2016.

[23] A. Dubrow and N. Gaynor, "From massive supercomputers come tiniest transistors," *Discovery - National Science Foundation*, 2015. [Online]. Available: https://www.nsf.gov/discoveries/disc_summ.jsp?cntn_id=134313&WT.mc_id=USNSF_6

[24] "NSF-supported supercomputing resources enable research that would otherwise be impossible," 2015. [Online]. Available: http://primeurmagazine.com/weekly/AE-PR-12-15-112.html

[25] A. Dubrow, "10 Ways Advanced Computing Catalyzes Science," 2017.

[26] "Using supercomputers to design nanoelectronics components," 2015. [Online]. Available: https://www.nanowerk.com/nanotechnology-news/newsid%3D39269.php

[27] G. Klimeck, "Atomistic Modeling of Future Nanoscale Electronic Devices with Nemo5," *Blue Waters Symposium*, 2015.

[28] "   / Japanese scientists have developed innovative transistors," 2017. [Online]. Available: https://hightech.fm/2017/12/28/transistors

[29] J. Thomas and J. Kielman, "Challenges for Visual Analytics," *Information Visualization*, vol. 8, no. 4, pp. 309–314, Dec. 2009. [Online]. Available: http://ivi.sagepub.com/content/8/4/309

[30] J. Scholtz, "User-Centered Evaluation of Visual Analytics," *Synthesis Lectures on Visualization*, Oct. 2017.

[31] Y. Zhu, "Measuring effective data visualization," *Advances in visual computing*, pp. 652–661, 2007. [Online]. Available: http://www.springerlink.com/index/H474878RR850R168.pdf

[32] B. Saket, A. Endert, and J. Stasko, "Beyond Usability and Performance: A Review of User Experience-focused Evaluations in Visualization," in *Proceedings of the Sixth Workshop on Beyond Time and Errors on Novel Evaluation Methods for Visualization*, ser. BELIV '16. New York, NY, USA: ACM, 2016, pp. 133–142. [Online]. Available: http://doi.acm.org/10.1145/2993901.2993903

[33] J. Davidson, *Evaluation Methodology Basics The Nuts and Bolts of Sound Evaluation.* Sage, 2018. [Online]. Available: https://us.sagepub.com/en-us/nam/evaluation-methodology-basics/book226129

[34] F. Beck, S. Koch, and D. Weiskopf, "Visual Analysis and Dissemination of Scientific Literature Collections with SurVis," *IEEE Transactions on Visualization and Computer Graphics*, vol. 22, no. 1, pp. 180–189, Jan. 2016.

[35] K. Madhavan, D. F. Mejia, H. Xian, L. K. Zentner, V. A. Farnsworth, S. Samek, and G. Klimeck, "Interactive Analytic Systems for Understanding the Scholarly Impact of Large-Scale E-science Cyberenvironments," in *2015 IEEE 11th International Conference on e-Science*, Aug. 2015, pp. 288–291.

[36] K. Madhavan, "Cyber-environments as Platforms for Integrating Engineering Research and Education,," *Proceedings of the Research in Engineering Education Symposium*, 2008.

[37] T. Roberts, "Todays Cyber Environment: Where Does Software Fit?" *Defensive Cyber Secur. Policies Proced. 2*, vol. 12, no. 2, 2010.

[38] "nanoHUB.org - Resources: Tools: Resonant Tunneling Diode Simulation with NEGF: Session: 1303996 "Resonant Tunneling Diode Simulation with NEGF"." [Online]. Available: https://nanohub.org/tools/rtdnegf/session?sess=1303996

[39] G. Klimeck, G. B. A. III, K. P. C. Madhavan, N. Denny, M. G. Zentner, S. Shivarajapura, L. K. Zentner, and D. L. Beaudoin, "Social Networks of Researchers and Educators on nanoHUB.org," in *2011 11th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing*, May 2011, pp. 560–565.

[40] "PhET Interactive Simulations." [Online]. Available: https://phet.colorado.edu/

[41] "Molecular Workbench." [Online]. Available: http://mw.concord.org/modeler/

[42] R. F. Tinker and Q. Xie, "Applying Computational Science to Education: The Molecular Workbench Paradigm," *Computing in Science Engineering*, vol. 10, no. 5, pp. 24–27, Sep. 2008.

[43] M. G. Jones and L. Brader-Araje, "The Impact of Constructivism on Education: Language, Discourse, and Meaning," *Am. Commun. J.*, vol. 5, no. 3, pp. 1–10, 2002.

[44] E. Garfield, "Citation Analysis as a Tool in Journal Evaluation: Journals can be ranked by frequency and impact of citations for science policy studies," *Science*, vol. 178, no. 4060, pp. 471–479, Nov. 1972. [Online]. Available: http://science.sciencemag.org/content/178/4060/471

[45] Wankat Phillip C., "Analysis of the First Ten Years of the Journal of Engineering Education," *Journal of Engineering Education*, vol. 93, no. 1, pp. 13–21, Jan. 2013. [Online]. Available: https://onlinelibrary.wiley.com/doi/abs/10.1002/j.2168-9830.2004.tb00784.x

[46] L. D. Brown and J. C. Gardner, "Using Citation Analysis to Assess the Impact of Journals and Articles on Contemporary Accounting Research (CAR)," *Journal of Accounting Research*, vol. 23, no. 1, pp. 84–109, 1985. [Online]. Available: http://www.jstor.org/stable/2490908

[47] Q. Zhou and K. C. Tan, "A bibliographic analysis of the literature on new service development," in *2008 4th IEEE International Conference on Management of Innovation and Technology*, Sep. 2008, pp. 872–877.

[48] S. Maggi, J. L. Kelsey, J. Litvak, and S. P. Heyse, "Incidence of hip fractures in the elderly: A cross-national analysis," *Osteoporosis International*, vol. 1, no. 4, pp. 232–241, Sep. 1991. [Online]. Available: https://link.springer.com/article/10.1007/BF03187467

[49] E. R. House, C. Haug, and N. Norris, "Evaluation policies and issues in the National Science Foundation," *Boulder, CO Univ. Color. Sch. Educ.*, 1995.

[50] R. C. Sonnichsen, *Building evaluation capacity within organizations.* Transaction Publishers New Brunswick, NJ and London, 1999.

[51] G. Ghere, J. A. King, L. Stevahn, and J. Minnema, "A Professional Development Unit for Reflecting on Program Evaluator Competencies," *American Journal of Evaluation*, vol. 27, no. 1, pp. 108–123, Mar. 2006. [Online]. Available: https://doi.org/10.1177/1098214005284974

[52] M. Henderson, S. Shurville, and K. Fernstrom, "The quantitative crunch: The impact of bibliometric research quality assessment exercises on academic development at small conferences," *Campus-Wide Information Systems*, vol. 26, no. 3, pp. 149–167, Jun. 2009. [Online]. Available: http://www.emeraldinsight.com/doi/10.1108/10650740910967348

[53] R. Broadus, "Toward a definition of bibliometrics," *Scientometrics*, vol. 12, no. 5-6, pp. 373–379, Nov. 1987. [Online]. Available: https://akademiai.com/doi/abs/10.1007/BF02016680

[54] A. Pritchard, "Statistical bibliography or bibliometrics?" *J. Doc.*, no. 25, pp. 348–349, 1969.

[55] J. Priem and B. H. Hemminger, "Scientometrics 2.0: New metrics of scholarly impact on the social Web," *First Monday*, vol. 15, no. 7, Jul. 2010. [Online]. Available: http://journals.uic.edu/ojs/index.php/fm/article/view/2874

[56] S. Haustein and T. Siebenlist, "Applying social bookmarking data to evaluate journal usage," *Journal of Informetrics*, vol. 5, no. 3, pp. 446–457, Jul. 2011. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S1751157711000393

[57] K. Weller and C. Puschmann, "Twitter for Scientific Communication: How Can Citations/References be Identified and Measured," 2011.

[58] R. E. Rubin, *Foundations of Library and Information Science, 3rd Edition*. New York: Neal-Schuman Publishers, 2010, vol. 3, google-Books-ID: muk_DwAAQBAJ.

[59] J. L. Sims and C. N. J. McGhee, "Citation analysis and journal impact factors in ophthalmology and vision science journals," *Clinical & Experimental Ophthalmology*, vol. 31, no. 1, pp. 14–22, Feb. 2003.

[60] D. Campbell, M. Picard-Aitken, G. Ct, J. Caruso, R. Valentim, S. Edmonds, G. T. Williams, B. Macaluso, J.-P. Robitaille, N. Bastien, M.-C. Laframboise, L.-M. Lebeau, P. Mirabel, V. Larivire, and . Archambault, "Bibliometrics as a Performance Measurement Tool for Research Evaluation: The Case of Research Funded by the National Cancer Institute of Canada," *American Journal of Evaluation*, vol. 31, no. 1, pp. 66–83, Mar. 2010. [Online]. Available: https://doi.org/10.1177/1098214009354774

[61] D. Hicks, H. Tomizawa, Y. Saitoh, and S. Kobayashi, "Bibliometric techniques in the evaluation of federally funded research in the United States," *Research Evaluation*, vol. 13, no. 2, pp. 76–86, Aug. 2004. [Online]. Available: https://academic.oup.com/rev/article/13/2/76/1529274

[62] E. C. M. Noyons, H. F. Moed, and M. Luwel, "Combining mapping and citation analysis for evaluative bibliometric purposes: A bibliometric study," *Journal of the American Society for Information Science; New York*, vol. 50, no. 2, pp. 115–131, Feb. 1999. [Online]. Available: https://search.proquest.com/docview/231475449/abstract/B681478EDEE94BA5PQ/1

[63] E. Rahm and A. Thor, "Citation Analysis of Database Publications," *SIGMOD Rec.*, vol. 34, no. 4, pp. 48–53, Dec. 2005. [Online]. Available: http://doi.acm.org/10.1145/1107499.1107505

[64] B. sdiken and Y. Pasadeos, "Organizational Analysis in North America and Europe: A Comparison of Co-citation Networks," *Organization Studies*, vol. 16, no. 3, pp. 503–526, May 1995. [Online]. Available: https://doi.org/10.1177/017084069501600306

[65] E. Garfield, "Is citation analysis a legitimate evaluation tool?" *Scientometrics*, vol. 1, no. 4, pp. 359–375, May 1979. [Online]. Available: https://link.springer.com/article/10.1007/BF02019306

[66] R. Kostoff, "The use and misuse of citation analysis in research evaluation," *Scientometrics*, vol. 43, no. 1, pp. 27–43, Sep. 1998. [Online]. Available: https://akademiai.com/doi/abs/10.1007/BF02458392

[67] M. MacRoberts and B. MacRoberts, "Problems of citation analysis," *Scientometrics*, vol. 36, no. 3, pp. 435–444, Jul. 1996. [Online]. Available: https://akademiai.com/doi/abs/10.1007/BF02129604

[68] L. C. SMITH, “Citation analysis,” *Library Trends*, vol. 30, pp. 83–106, 1981. [Online]. Available: https://ci.nii.ac.jp/naid/10009572773/

[69] S. Teufel, A. Siddharthan, and D. Tidhar, “Automatic Classification of Citation Function,” ser. EMNLP ’06. Stroudsburg, PA, USA: Association for Computational Linguistics, 2006, pp. 103–110. [Online]. Available: http://dl.acm.org/citation.cfm?id=1610075.1610091

[70] D. E. Chubin and S. D. Moitra, “Content Analysis of References: Adjunct or Alternative to Citation Counting?” *Social Studies of Science*, vol. 5, no. 4, pp. 423–441, 1975. [Online]. Available: http://www.jstor.org/stable/284806

[71] D. Koufogiannakis, L. Slater, and E. Crumley, “A Content Analysis of Librarianship Research,” *Journal of Information Science*, vol. 30, no. 3, pp. 227–239, Jun. 2004. [Online]. Available: https://doi.org/10.1177/0165551504044668

[72] K. Jrvelin and P. Vakkari, “Evolution of library and information science, 19651985: Content analysis of journal articles,” *Journal of the Association for Information Science and Technology*, vol. 29, no. 1, pp. 423–441, 1975.

[73] X. Wang and A. McCallum, “Topics over Time: A non-Markov Continuous-time Model of Topical Trends,” ser. KDD ’06. New York, NY, USA: ACM, 2006, pp. 424–433. [Online]. Available: http://doi.acm.org/10.1145/1150402.1150450

[74] Y. Ding, G. G. Chowdhury, and S. Foo, “Bibliometric cartography of information retrieval research by using co-word analysis,” *Information Processing & Management*, vol. 37, no. 6, pp. 817–842, Nov. 2001. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0306457300000510

[75] B.-W. On, D. Lee, J. Kang, and P. Mitra, “Comparative Study of Name Disambiguation Problem Using a Scalable Blocking-based Framework,” ser. JCDL ’05. New York, NY, USA: ACM, 2005, pp. 344–353. [Online]. Available: http://doi.acm.org/10.1145/1065385.1065463

[76] Smalheiser Neil R. and Torvik Vetle I., “Author name disambiguation,” *Annual Review of Information Science and Technology*, vol. 43, no. 1, pp. 1–43, Feb. 2011. [Online]. Available: https://onlinelibrary.wiley.com/doi/abs/10.1002/aris.2009.1440430113

[77] H. Han, L. Giles, H. Zha, C. Li, and K. Tsioutsiouliklis, “Two supervised learning approaches for name disambiguation in author citations,” in *Proceedings of the 2004 Joint ACM/IEEE Conference on Digital Libraries, 2004.*, Jun. 2004, pp. 296–305.

[78] C. Niu, W. Li, and R. K. Srihari, “Weakly Supervised Learning for Cross-document Person Name Disambiguation Supported by Information Extraction,” ser. ACL ’04. Stroudsburg, PA, USA: Association for Computational Linguistics, 2004. [Online]. Available: https://doi.org/10.3115/1218955.1219031

[79] W. W. Cohen, P. Ravikumar, and S. Fienberg, “A Comparison of String Metrics for Matching Names and Records,” *International Conference on Knowledge Discovery and Data Mining (KDD) 09, Workshop on Data Cleaning, Record Linkage, and Object Consolidation*, 2003.

[80] Y. Song, J. Huang, I. G. Councill, J. Li, and C. L. Giles, "Efficient Topic-based Unsupervised Name Disambiguation," ser. JCDL '07. New York, NY, USA: ACM, 2007, pp. 342–351. [Online]. Available: http://doi.acm.org/10.1145/1255175.1255243

[81] D. Thain, T. Tannenbaum, and M. Livny, "Condor and the Grid," 2003, pp. 299–335.

[82] D. Borthakur, "The Hadoop Distributed File System: Architecture and Design," 2007.

[83] V. Batagelj and A. Mrvar, "Pajek-program for large network analysis," *Connections*, vol. 21, no. 2, pp. 47–57, 1998.

[84] S. P. Borgatti, "NetDraw: Graph visualization software," 2002.

[85] V. Uren, S. Buckingham Shum, M. Bachler, and G. Li, "Sensemaking tools for understanding research literatures: Design, implementation and user evaluation," *International Journal of Human-Computer Studies*, vol. 64, no. 5, pp. 420–445, May 2006. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S1071581905001709

VITA

Daniel Mejia is a Colombian who received his bachelor degree on Systems and Computer Engineering from the Universidad de los Andes at Bogot-Colombia on 2002, and his master degree on Engineering from the same University on 2004. In this year He started his PhD studies; being part of IMAGINE group; until 2005 when He joined the industry. During this period of time his research was focused on Virtual reality, Human-Machine interaction, and graphical programming.

From 2004 to 2009 he was a lecturer at Universidad de los Andes, and from 2005 to 2010 he worked as Information Technology Manager at Dayscript (Bogot, Colombia), during this period of time he was involved in the development of the CMS Dayware and different web portals (Golgolgol.net, Colfuturo, Fulbright Colombia, etc...)

On the summer of 2011, Daniel started his PhD at Purdue University and joined the Klimeck's research group. He is a former board officer in the Colombian Student Asociation at Purdue (CSAP) (2011-2013)

During his tenure in the Klimeck group he worked on the following research projects:

- **Nemo5 Development**
  - Compilation of Nemo5 on multiple architectures: Mac (OSX, macports), Windows (cygwin, real-petsc only) , linux-static (ubuntu, fedora), nanoHUB, and support on standard architectures: Linux, Purdue clusters, and Cray systems.
  - Speed optimization of Hamiltonian-constructor (code optimization + memory reusability).
  - Improving Nemo5 control files.
  - Extending Nemo5 capabilities using Python.

  - Time/Memory tracking features on Nemo5.
  - Online Tool to support creation of Nemo5 Input Decks.
  - Nemo5 Graphical User Interfaces Nemui and Nemo Server.
  - Nemo5 Documentation
  - wxVTK modification to support GTK3

- **NanoHUB**
  - Demonstrate nanoHUB Impact
  - Different contributions on nanoHUB
  - Summer School: Extending Nemo5 using Python scripting.