

Video Article

Scalable Nanohelices for Predictive Studies and Enhanced 3D Visualization

Kwyn A. Meagher¹, Benjamin N. Doblack¹, Mercedes Ramirez², Lilian P. Davila¹

¹Materials Science and Engineering, School of Engineering, University of California Merced

²Computer Science and Engineering, School of Engineering, University of California Merced

Correspondence to: Lilian P. Davila at ldavila@ucmerced.edu

URL: <https://www.jove.com/video/51372>

DOI: [doi:10.3791/51372](https://doi.org/10.3791/51372)

Keywords: Physics, Issue 93, Helical atomistic models; open-source coding; graphical user interface; visualization software; molecular dynamics simulations; graphical processing unit accelerated simulations.

Date Published: 11/12/2014

Citation: Meagher, K.A., Doblack, B.N., Ramirez, M., Davila, L.P. Scalable Nanohelices for Predictive Studies and Enhanced 3D Visualization. *J. Vis. Exp.* (93), e51372, doi:10.3791/51372 (2014).

Abstract

Spring-like materials are ubiquitous in nature and of interest in nanotechnology for energy harvesting, hydrogen storage, and biological sensing applications. For predictive simulations, it has become increasingly important to be able to model the structure of nanohelices accurately. To study the effect of local structure on the properties of these complex geometries one must develop realistic models. To date, software packages are rather limited in creating atomistic helical models. This work focuses on producing atomistic models of silica glass (SiO₂) nanoribbons and nanosprings for molecular dynamics (MD) simulations. Using an MD model of “bulk” silica glass, two computational procedures to precisely create the shape of nanoribbons and nanosprings are presented. The first method employs the AWK programming language and open-source software to effectively carve various shapes of silica nanoribbons from the initial bulk model, using desired dimensions and parametric equations to define a helix. With this method, accurate atomistic silica nanoribbons can be generated for a range of pitch values and dimensions. The second method involves a more robust code which allows flexibility in modeling nanohelical structures. This approach utilizes a C++ code particularly written to implement pre-screening methods as well as the mathematical equations for a helix, resulting in greater precision and efficiency when creating nanospring models. Using these codes, well-defined and scalable nanoribbons and nanosprings suited for atomistic simulations can be effectively created. An added value in both open-source codes is that they can be adapted to reproduce different helical structures, independent of material. In addition, a MATLAB graphical user interface (GUI) is used to enhance learning through visualization and interaction for a general user with the atomistic helical structures. One application of these methods is the recent study of nanohelices via MD simulations for mechanical energy harvesting purposes.

Video Link

The video component of this article can be found at <https://www.jove.com/video/51372/>

Introduction

Helical nanostructures are typically produced in the laboratory using chemical vapor deposition techniques^{1,2}, while new approaches have been reported in the literature³. In particular nanosprings and nanoribbons have been studied because of their distinct properties and promising applications in sensors, optics, and electromechanical and fluidic devices⁴⁻⁷. Synthesis methods have been reported to produce silica (SiO₂) nanoribbons, making these structures potential building block units for hierarchical systems. Novel synthesis of 3D silica nanosprings has expanded their applications to chemiresistors when coated with ZnO⁸ or nanoparticles for diagnostic applications⁹⁻¹⁰.

Experimental studies on the mechanical properties of silica nanosprings and nanoribbons are scarce, primarily due to current limitations in manipulation and testing methods and equipment. Investigations into the nanomechanics of nanostructures and nanosprings have been reported using theory and simulations¹¹⁻¹⁴. Some simulations¹³ have focused on nanomechanical behavior of amorphous nanosprings because they can explore regimes not fully accessible through experimentation. Atomistic studies of metallic nanosprings have been reported in literature to investigate the size dependence of elastic properties¹⁵, and more recently the nanomechanics of helical crystalline silica nanostructures¹⁴.

Experimental testing of nanospring structures has also been performed in different materials such as helical carbon nanostructures and carbon nanocoils¹⁶⁻¹⁷. Despite the knowledge gathered thus far, a more complete understanding of the mechanical properties of these novel nanostructures is needed for future nanodevice fabrication efforts.

As MD studies of silica glass (non-crystalline silica) nanohelices are still quite limited, the atomistic modeling of such structures requires the creation of customized codes. No other alternative methods of creating silica glass helical MD models have been identified thus far upon recent literature search. In this work, a bottom-up approach to the atomistic modeling of helical silica glass nanostructures including nanosprings and nanoribbons is pursued for future large-scale MD nanomechanical simulations. The general approach involves the creation of an MD “bulk” silica glass model as reported previously¹⁸, and carving out various helical nanostructures from this “bulk” sample via two robust and adaptable computer codes developed for this purpose. Both computational procedures offer a distinct way to create nanoribbon and nanospring models with great efficiency and atomistic detail; these structures are suitable for large-scale atomistic simulations. In addition, a customized graphical user interface is used to facilitate creation and visualization of the helical structures.

The structure of the “bulk” silica glass model is initially created at room temperature. Large-scale MD simulations are conducted for this purpose using the Garofalini interatomic potential similar to prior studies¹⁸, which is relatively efficient computationally and appropriate for large systems. The initial “bulk” silica glass structure consists of a cubical model ($14.3 \times 14.3 \times 14.3 \text{ nm}^3$) which contains 192,000 atoms. The “bulk” silica glass model is equilibrated at 300 K for 0.5 nsec to obtain the initial state using periodic boundary conditions.

Two computational procedures are designed and utilized to create atomistic silica nanoribbon and nanospring models. The first method involves carving out silica nanoribbons from the “bulk” structure using the parametric equations that define a helix, and its geometry (pitch, radius of helix, and wire radius). This procedure includes using the AWK programming language, the LINUX operating system, and open-source visualization software¹⁹. The general iterative procedure to create atomistic models of nanoribbons involves: (1) selecting an atom in the “bulk” silica glass model, (2) calculating the distance from the selected atom to a point in space on a pre-defined helical function, (3) comparing this distance to the radius of the desired nanoribbon, and (4) discarding or keeping the atom in an output data model. A detailed step-by-step description for this method is included in the **Scalable Open-Source Codes Supplemental Material**. With this method, several silica nanoribbons were created using different pitch, radius of helix and nanoribbon radius values, which were measured subsequently for accuracy against the desired dimensional values with molecular analysis and visualization software¹⁹⁻²⁰. Atomistic models of silica nanoribbons were generated with functional geometries (high values of pitch and low values of nanoribbon radius). Some artifacts, consisting of atoms excluded in error, leading to a less smooth nanoribbon surface, were observed at exceedingly high nanoribbon radius values and extremely low pitch values. Similar methods have been used in the process of creating silica nanowires²¹⁻²³.

The second method presented here includes carving out silica nanosprings from the “bulk” silica structure by implementing pre-screening methods to increase efficiency in addition to the mathematical equations for a helix. This procedure required creating a more robust C++ code to allow greater flexibility in modeling these helical nanostructures. The iterative method to create atomistic models of nanosprings includes: (1) discarding all atoms guaranteed to fall outside the helical path, (2) deterministically selecting a point on the helical path, (3) comparing all atoms within a specific distance to this selected point, and (4) discarding or storing each atom in an output data model. A step-by-step description for this method is also included in the **Scalable Open-Source Codes Supplemental Material**. With this method, several silica nanospring models were obtained with varied dimensions (wire radius, radius of helix, and pitch of nanospring) as shown in **Figure 1**. Highly precise silica nanospring models were obtained efficiently with this method, with no evidence of artifacts found at extreme (low and high) pitch values for the nanospring. The creation and use of the graphical user interface for this method is described in the **Protocol** section.

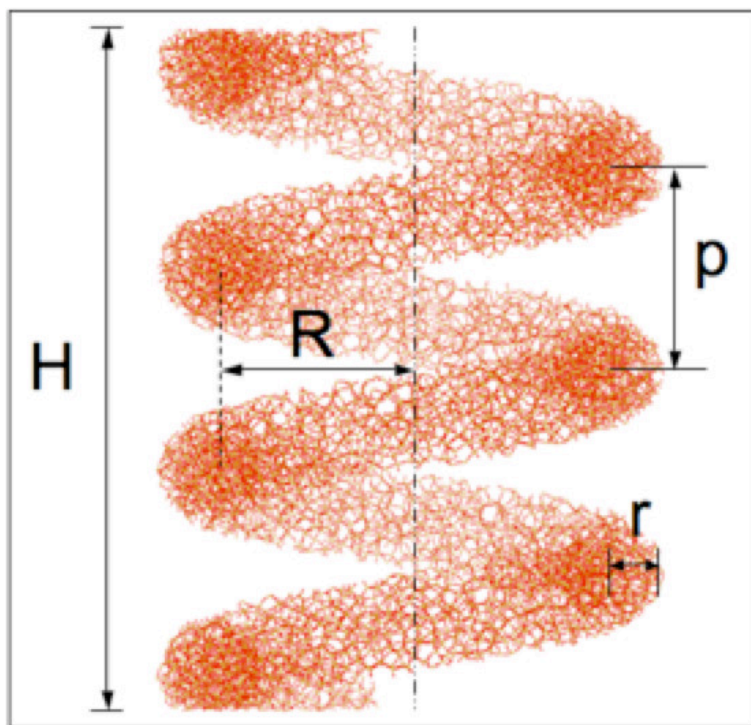


Figure 1: A general helical structure showing characteristic dimensions, where r , R and p represent the wire radius, radius of helix, and pitch respectively. H denotes the total height of the helical structure²³.

This protocol describes how to prepare the NanospringCarver files, running MATLAB²⁴ on a LINUX²⁵ PC, and use a graphical user interface to prepare atomistic nanospring models. These previously unavailable models serve as the basis for novel molecular dynamics (MD) simulations²³ toward materials innovation research.

The general step-by-step procedure to create atomistic nanospring models involves using the following elements: (a) NanospringCarver (v. 0.5 beta) code (open-source in C++ language), (b) bulk silica glass model (input file), (c) MATLAB GUI interface and related files, and (d) MATLAB software (version 7) using a local license on a LINUX PC. Items (a)-(c) above (NanospringCarver code, silica glass model, MATLAB GUI files) are free to download online²⁶. MATLAB (Matrix Laboratory) is a high-level language for numerical computation, visualization, and application development from MathWorks²⁴, which is mostly used for data visualization and analysis, image processing, and computational biology.

Protocol

1. Preparing NanospringCarver Files and Starting MATLAB on a LINUX PC

The following steps are designed for a general user to make use of the files provided online²⁶.

1. Unpack the **nanosprings.tar.gz** file archive into the "Home" or another preferred directory.
 1. Download the **nanosprings.tar.gz** file archive from the web repository²⁶.
 2. Locate the downloaded archive and move it to a preferred working directory entitled "Documents/Nanosprings".
 3. Right-click **nanosprings.tar.gz** and select "extract here" from the right-click context menu.

2. Verify that all of the required files are present in the current directory. A list of those files and their purpose follows:

Makefile – manually managed compile file for *nanosprings.cpp* and *Point.cpp*

Nanosprings.fig – MATLAB GUI internals

Nanosprings.m – MATLAB GUI code

Point.cpp – *Point* (atom) class definition

Point.h – *Point* (atom) class header

carve – stand-alone nanosprings executable

example.par – example parameter file

glasscube.inp – glasscube data file

nanosprings.cpp – main nanosprings code

nanosprings_diagram.jpg – example nanospring for display

nanospringsmex.cpp – MATLAB-integrated nanosprings.cpp

nanospringsmex.mexglx – MATLAB-integrated nanosprings executable

Note: The user will need to create the "**nanospringsmex.mexglx**" executable file for the particular Linux machine being used (32-bit version in this example). If this has not yet been done, verify access to the MATLAB "mex" compiler by typing on the command line "**which mex**" and verifying the existence of the program. Also verify access to the MATLAB program by typing on the command line "**which matlab**". Using a command line to type "**mex nanospringsmex.cpp Point.cpp**" will create the "**nanospringsmex.mexglx**" executable MATLAB-integrated NanospringCarver file, as shown in the instructions below. Though not required for the GUI interface, if desired a stand-alone version of the NanospringCarver program can be created by typing "**make**" on a command line. This will compile the **nanosprings.cpp** and **Point.cpp** program elements together to create the "**carve**" executable file. In this tutorial, the "**glasscube.inp**" file contains position information for 192,000 silicon and oxygen atoms representing a silica glass model, with each line containing an atom ID, atom type, and x, y, z coordinates for the atom. The first line of the file is the total atom count (192,000). The atomic coordinates in this file are relative values, which if multiplied by 0.716 would represent nanometer distances.

3. On the desktop, open a terminal window. On many LINUX versions accomplish this by simultaneously pressing the "Ctrl", "Alt" and "T" keys.
4. Change the directory to the folder into which the nanosprings project files were extracted by typing:
cd Documents/Nanosprings/
5. Next, run the command to compile the binary for the system by typing:
mex nanospringsmex.cpp Point.cpp
6. Next initiate MATLAB by typing **matlab** on the command line

2. Modifying and Using a Graphical User Interface (GUI) to the NanospringCarver Program

Follow the steps below using the files provided online²⁶.

1. Open the GUIDE in MATLAB by clicking the GUIDE icon, on the top left toolbar area (**Figure 2**), to display a new window with the GUIDE quick start (**Figure 3**).

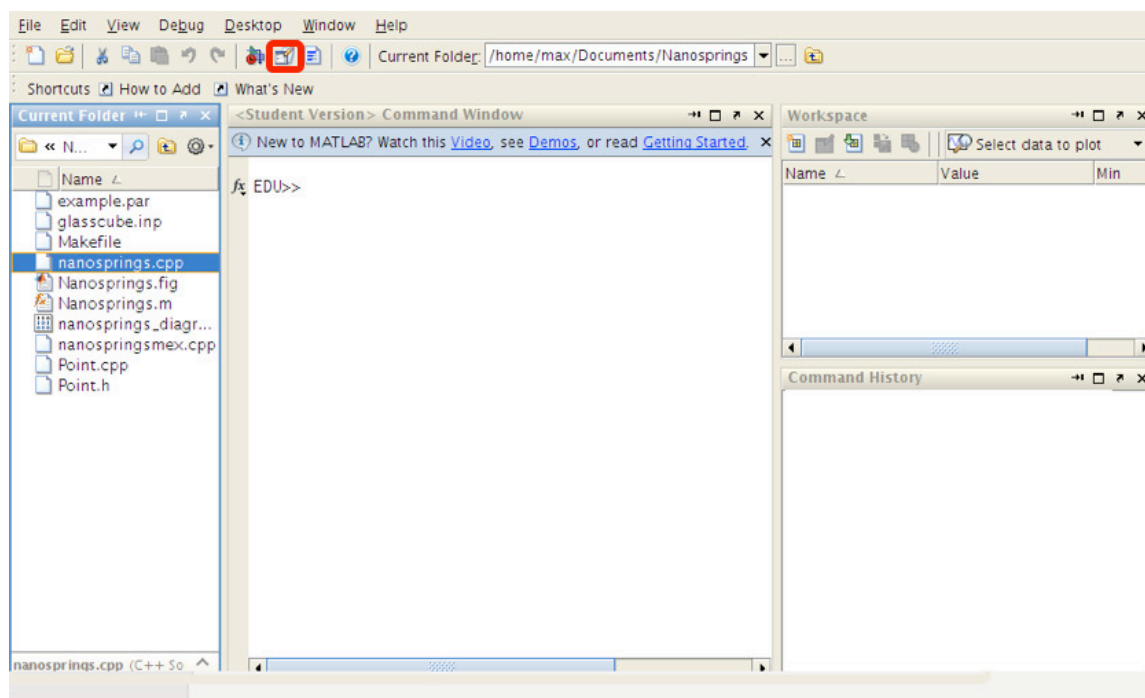


Figure 2: MATLAB user interface showing how to open MATLAB GUIDE.

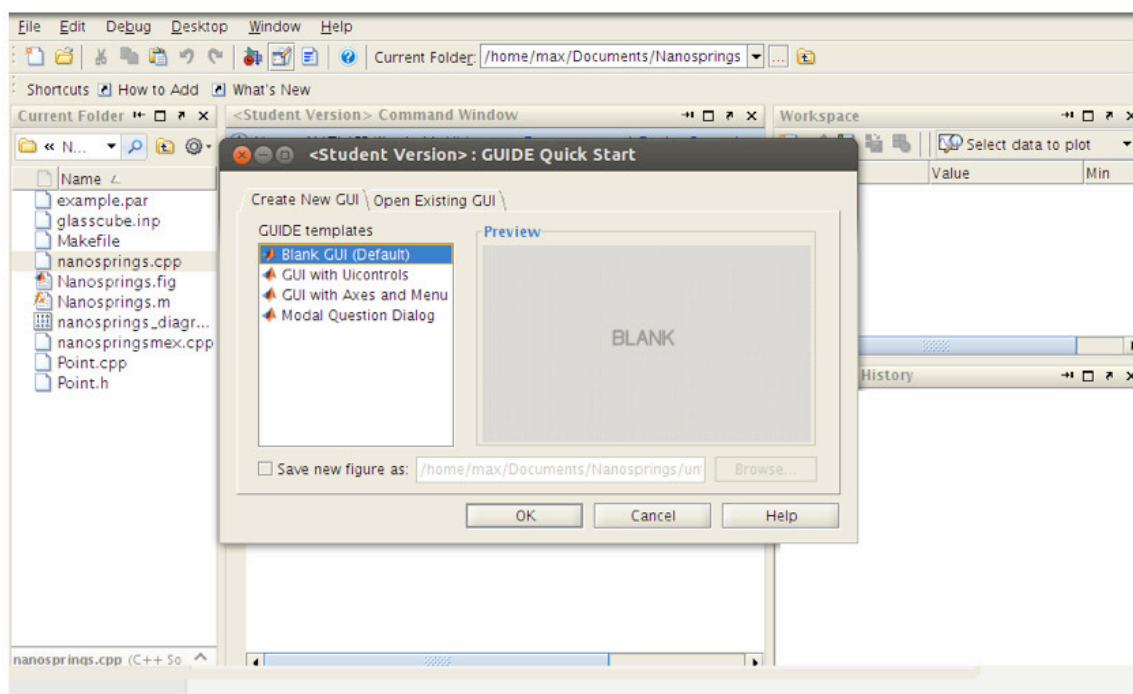


Figure 3: MATLAB GUIDE interface initializing.

2. Use the "Open Existing GUI" tab (Figure 4) to modify an existing figure. Click on the "Browse" button to search for the existing GUI figure to be modified. After selecting the figure file (**Nanosprings.fig**, see blue box), click on "Open" on both windows to display a new window with the GUI figure. Locate the buttons available to be utilized for GUI creation on the left panel (Figure 5).

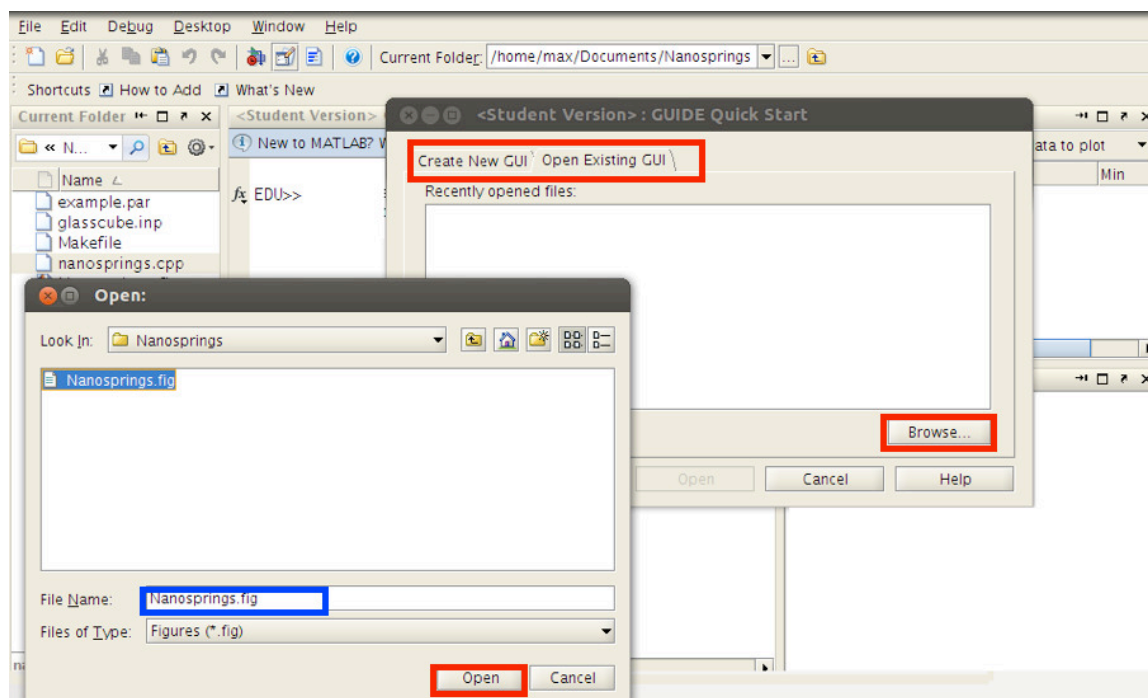


Figure 4: MATLAB GUIDE interface showing how to open an existing GUI figure file.

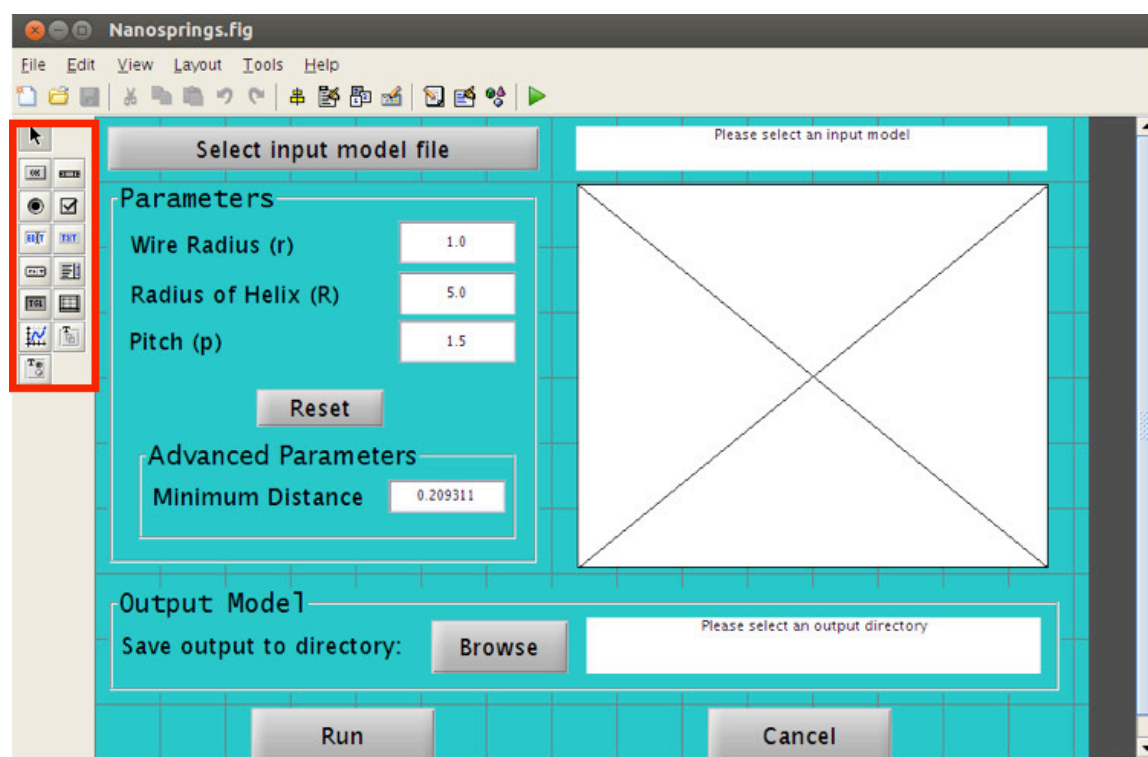


Figure 5: MATLAB GUIDE interface showing tools for modifying an existing GUI figure.

3. In order to run the GUI, click on "Run" under the "Tools" menu. Then, click "Yes" when a pop-up window prompts whether to save the figure before running. A new window displays the modified GUI.
4. If necessary, create another GUI for a different specific material using this GUI as an example.
5. To set up the example run, first click on the "Select input model file" button at the top of the GUI and navigate to the "**glasscube.inp**" file. Select this file and click "Open" to close the browsing window. The selected input file and path to it should now appear in the GUI window to the right of the "Selected input model file" button (**Figure 6**).

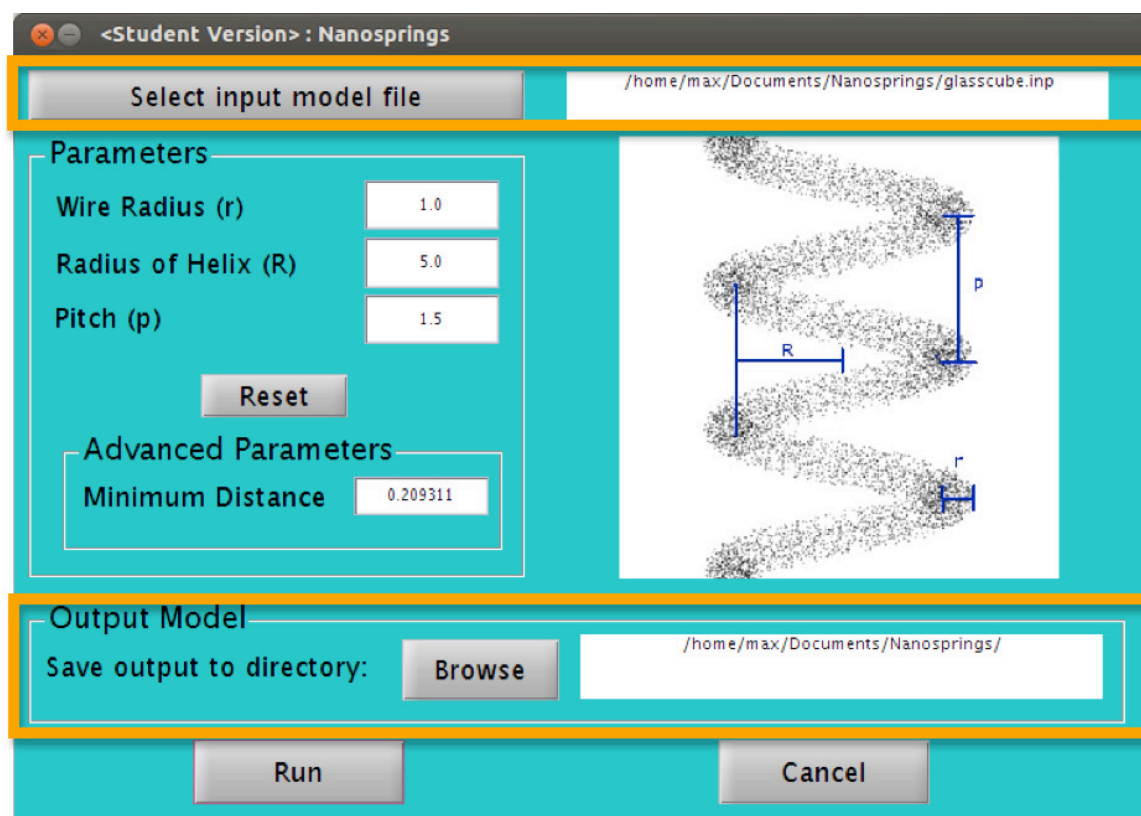
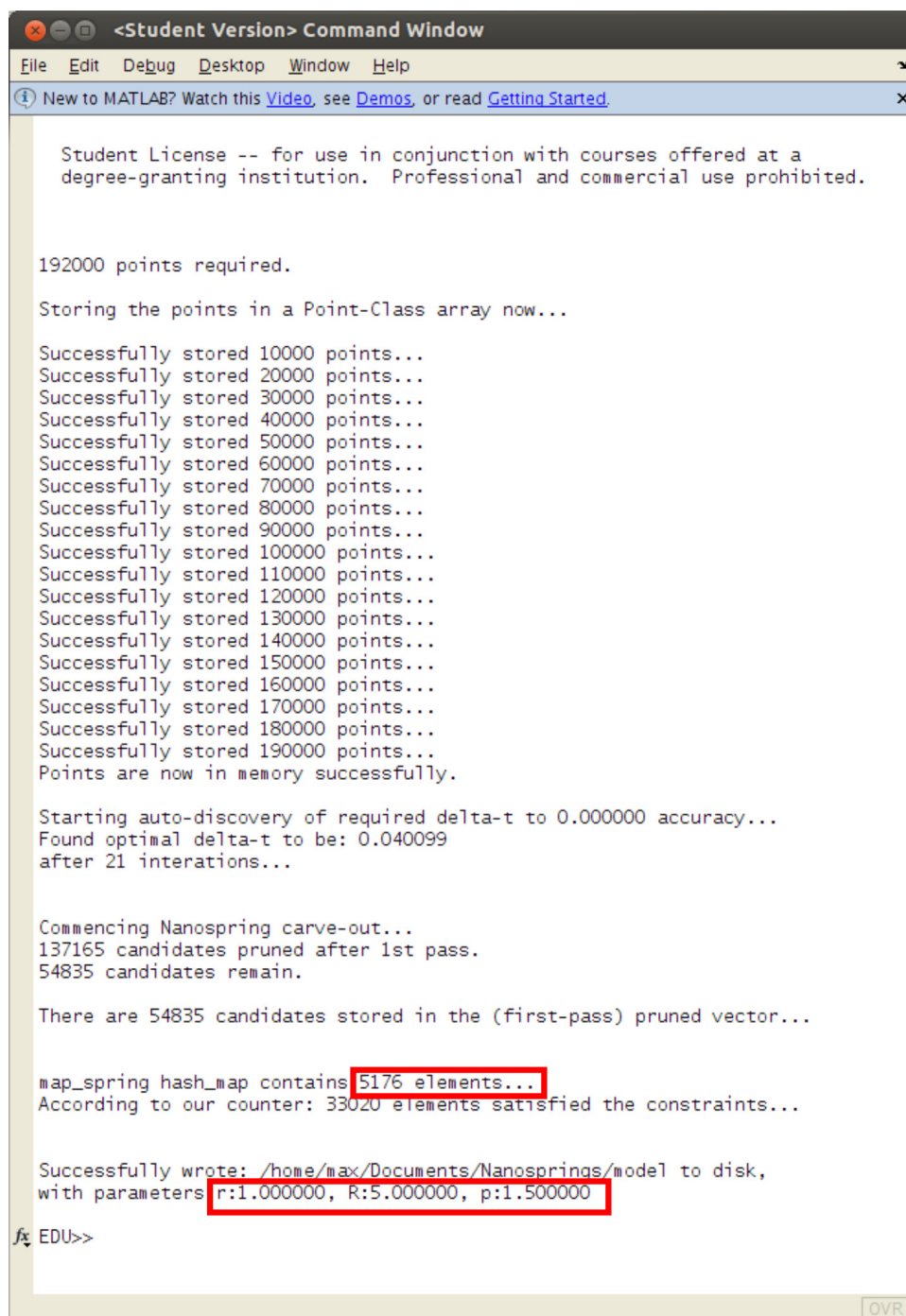


Figure 6: Screenshot of using GUI to create an example silica nanospring model.

6. Next, use the "Browse" button in the "Output Model" section to browse for and select the directory to save the output model into. Ensure that the output directory is actively selected in order for the run to be enabled, even though there is an output directory already listed in the accompanying window to the right of this button.
 Note: The "Advance Parameters Minimum Distance" value listed (0.209311 in **Figure 6**) was computed specifically for the "glasscube.inp" input file provided in this example, and should be left as is. This value may be computed as necessary on first use of a different input file by entering a value of "0" in this location before running the model. In this example, all parameter values are in relative units to match the input atomic coordinate system. If multiplied by 0.716 the parameter values would represent nanometer distances.
7. Run the example using the given spring parameters of $r = 1.0$, $R = 5.0$, $p = 1.5$, and $d = .209311$ by pressing the GUI "Run" button. View feedback from the run in the MATLAB Command window (**Figure 7**). In the feedback, check that the spring parameters are confirmed, that the input data file is read successfully, and the results stored in the output file named "model" are described.



```

<Student Version> Command Window
File Edit Debug Desktop Window Help
New to MATLAB? Watch this Video, see Demos, or read Getting Started.

Student License -- for use in conjunction with courses offered at a
degree-granting institution. Professional and commercial use prohibited.

192000 points required.

Storing the points in a Point-Class array now...

Successfully stored 10000 points...
Successfully stored 20000 points...
Successfully stored 30000 points...
Successfully stored 40000 points...
Successfully stored 50000 points...
Successfully stored 60000 points...
Successfully stored 70000 points...
Successfully stored 80000 points...
Successfully stored 90000 points...
Successfully stored 100000 points...
Successfully stored 110000 points...
Successfully stored 120000 points...
Successfully stored 130000 points...
Successfully stored 140000 points...
Successfully stored 150000 points...
Successfully stored 160000 points...
Successfully stored 170000 points...
Successfully stored 180000 points...
Successfully stored 190000 points...
Points are now in memory successfully.

Starting auto-discovery of required delta-t to 0.000000 accuracy...
Found optimal delta-t to be: 0.040099
after 21 iterations...

Commencing Nanospring carve-out...
137165 candidates pruned after 1st pass.
54835 candidates remain.

There are 54835 candidates stored in the (first-pass) pruned vector...

map_spring hash_map contains 5176 elements...
According to our counter: 33020 elements satisfied the constraints...

Successfully wrote: /home/max/Documents/Nanosprings/model to disk,
with parameters r:1.000000, R:5.000000, p:1.500000

fx EDU>>
  
```

Figure 7: MATLAB Command window feedback from GUI-based Nanosprings run.

Note: In the above example, the file “model” contains 5,176 atoms comprising the desired spring, one per line, with the first line giving the total number of atoms in the file. Each line defining an atom includes the atom ID, atom type, and x, y, z coordinates of that atom.

- Once the GUI interface is finalized, perform successive runs by right-clicking on “Nanosprings.m” in the MATLAB “Current Folder” window, and selecting “Run” to bring up the GUI interface directly.
Note: Various references are listed²⁷⁻³¹ for additional information on MATLAB GUIDE and the basic GUIDE interface.

3. Verifying NanospringCarver Results in an Open-source Visualizer¹⁹

The following steps are designed for a general user to visualize and verify the output spring models created by NanospringCarver.

- Use the NanospringCarver MATLAB GUI as described above to generate files for input into the visualization program¹⁹. When running the visualization program, use the “point coordinate file” input option, distinguish atom types by color, and select an axis grid border for the field.

2. Measure distances in the spring models and make a record of them.
3. Compare measured data against desired spring dimensions and verify spring model accuracy.

4. Using NanospringCarver Results in MD Tensile Simulations of Nanosprings

The following steps are summarized for a general user to use the spring models created by NanospringCarver as input to a conventional open-source MD code³².

1. Download the latest version of the open-source MD program LAMMPS. Refer to the associated online documentation for manuals and examples.
2. Determine the dimensions of the desired nanospring model in order to prepare the appropriate initial bulk silica glass model, as reported before¹⁸.
3. Create the desired nanospring model using the NanospringCarver MATLAB GUI (see Section 2 above).
4. Perform tensile simulations on the desired nanospring, by stretching the model axially^{11,13,23}. Produce a representative video of nanospring model being stretched (see **Figure 8**, below, and **Animated Figure 1**) for visualization and analysis. Scientific results regarding the stress-strain behavior and stiffness of several nanospring models under tension have been reported elsewhere²³.

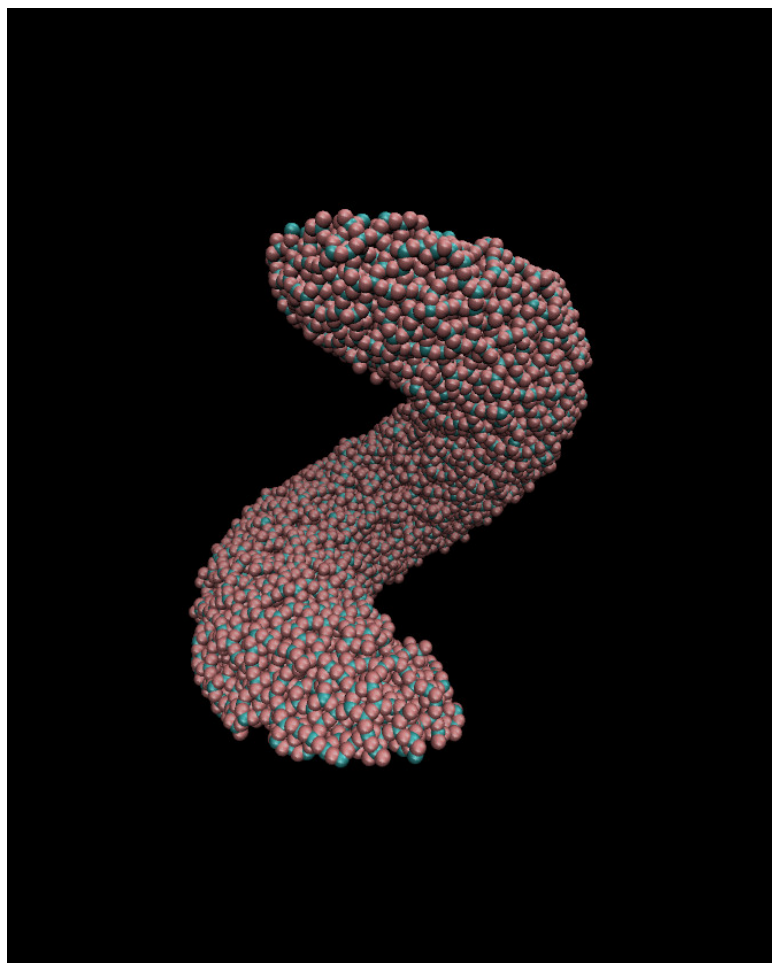


Figure 8: Screenshot of a silica nanospring during tensile simulation (see also **Animated Figure 1**).

Representative Results

The atomistic nanoribbon models created with the first computational procedure (nanoribbons code) and their associated dimensions are shown in **Figure 9**. The resulting nanospring models using the second computational procedure (nanosprings code) and associated dimensions are shown in **Figure 10**.

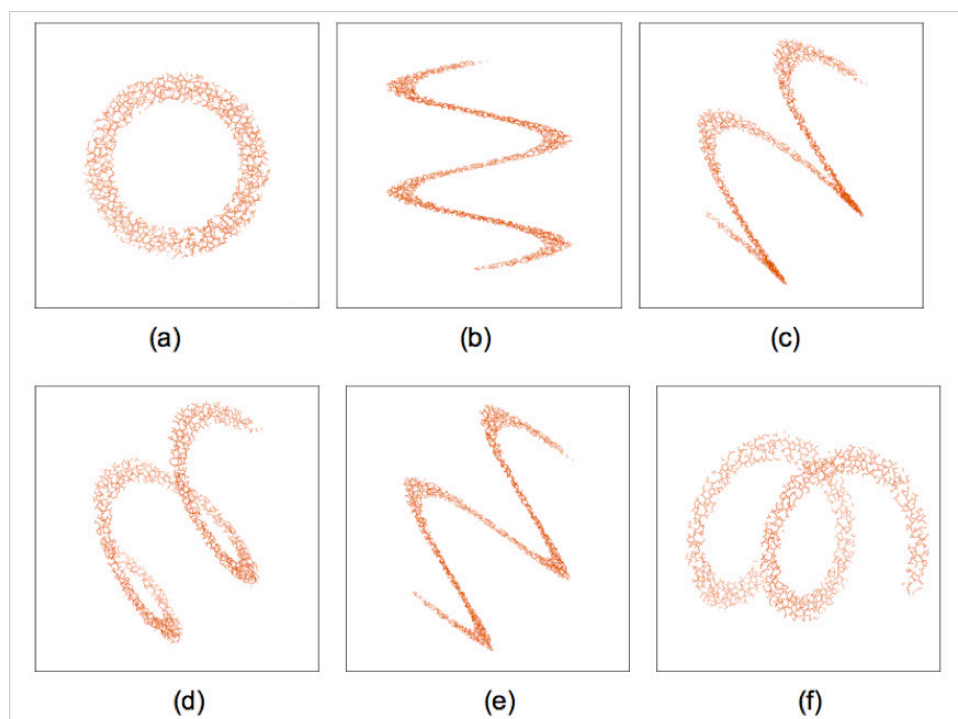


Figure 9. Atomistic model of a silica nanoribbon with desired dimensions: r (nanoribbon radius) = 1.07 nm, R (radius of helix) = 5.37 nm, and p (pitch) = 7.16 nm. Snapshots illustrate distinct views of the nanostructure: (a) top view, (b) lateral view, (c) lateral view with additional rotation, and (d)-(f) diagonal views. The SiO_2 nanoribbon model contains 3,354 atoms. The total ribbon height H is 14.1 nm^{23} .

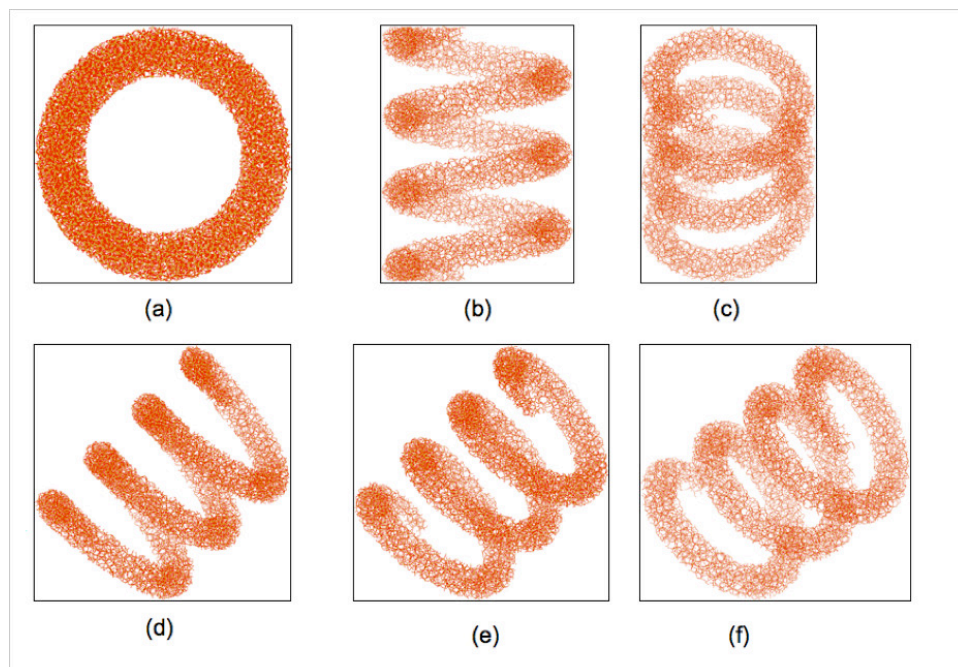


Figure 10. Atomistic model of a silica nanospring with specified dimensions: r (wire radius) = 1.07 nm, R (radius of helix) = 4.29 nm, and p (pitch) = 4.29 nm. Snapshots show different views of the nanospring model: (a) top view, (b) lateral view, (c) lateral view with additional forward rotation, and (d)-(f) diagonal views. The SiO_2 nanospring model consists of 21,246 atoms. The total spring height H is 14.32 nm^{23} .

The range of nanoribbon and nanospring dimensions generated with both codes was ample ($r < 3.75 \text{ nm}$, $R < 9 \text{ nm}$, and $p < 12.57 \text{ nm}$). Each of the above methods offers a unique way to create silica nanosprings and nanoribbons suitable for atomistic simulations. Both methods are flexible and can be adapted to produce different helical structures independent of the material, which makes them highly useful and versatile.

Animated Figure 1. Silica nanospring during tensile simulation.

Discussion

Modification of the original approach to create nanohelical structures led to the development of two distinct codes to allow creation of both nanoribbons and nanosprings from an initial bulk silica glass MD model. The verification of the silica nanoribbon and nanospring models was pursued using different software packages¹⁹⁻²⁰, which confirmed their dimensional accuracy within the measurement capability of the programs.

Comparison between nanosprings and nanoribbons was also performed by overlaying the models from different sides and angles, which resulted in additional geometry verification. Both computational methods developed in this project created helical nanostructures in a distinct way, with added value due to their scalability for use with any bulk material model size and potential use in modeling nanohelical structures from other materials. The resultant models presented here showed there are no detectable artifacts (atoms missing from the desired nanohelical structure) generated using either method. In addition, the computational methods developed in this work are flexible for creating right-handed or left-handed helical nanostructures, simply by inverting the order of the sine and cosine functions defining the helix. Future applications of this method will include scaling to larger helical structures allowing extended parameter variation, and exploration of use with different initial materials.

Limitations of this method include dimensional restrictions on the created nanohelices depending on the initial bulk silica model used, which can involve significant computing resources as the model size increases. As currently implemented, the nanoribbon or nanospring height will extend to the size of the original bulk model. The first computational method generates accurate nanoribbon models for a range of parameters when the pitch value is greater than 7.16 nm and the radius of the helical wire is greater than 10% of the shortest dimension of the "bulk" silica glass structure. The second computational method generates accurate nanospring models without parameter limitation. This is particularly important for conducting MD simulations where readily available atomistic nanostructural models are needed to investigate different size conditions.

A critical step in the protocol would be to verify on first use of a particular initial MD bulk material model that the minimum distance between the closest two atoms in the model has been determined and input correctly with the dimensional parameters. Additionally, care should be taken to ensure that requested helical dimensions do not exceed the bulk material model dimensions.

Technological advances have facilitated the creation and characterization of complex helical nanostructures such as oxide nanoribbons and nanosprings in the laboratory. These nanoscale structures have unique properties that require thorough investigation in order to realize their full potential for various applications. MD studies of the mechanical behavior of these helical structures require flexible codes which can easily and precisely create helical nanostructures, and subsequently make use of appropriate interatomic potentials and methods for predictive simulations. To fulfill this first requirement, accurate structural modeling codes were developed which will be used for large-scale MD compression simulations and experimental validation.

This method of creating MD silica glass (non-crystalline) nanohelical models is significant, as similar codes not readily available and other alternative approaches have been focused on crystalline nanostructures. This modeling effort has been expanded, with the resultant nanostructures used in MD simulation studies, which have led to a thesis focused on the elastic response of silica glass nanohelices under tensile loads²³. Time-efficient simulation of nanostructures is a challenging problem, however new programming techniques and atomistic models are especially becoming important for predictive studies. This modeling technique is rapidly gaining interest and quickly becoming an efficient method for models which require high performance computing. Future academic efforts will likely include the adaptation of these codes for training computational researchers and in classroom exercises. Performing MD simulations to study the response of helical structures to different loading conditions is certainly feasible with these robust atomistic models. The success of future manufacturing using these nanostructures as building blocks will depend on understanding of their structure and properties, with implications on nanomanipulation and self-assembly processes. This work is a step toward understanding the mechanical behavior of such nanostructures using large-scale MD simulations, which can be potentially useful for designing nanodevices for a large number of applications.

Disclosures

The authors declare that they have no competing financial interests.

Acknowledgements

The authors want to thank Tim Allis at UC Merced for his assistance in this project. The NSF-COINS program at UCM supported (KAM) in an early part of this work. An NSF-BRIGE award supported co-authors (BND and KAM), providing funds for this work and travel expenses to conferences.

The research group wishes to acknowledge primarily the National Science Foundation for funding this work via a BRIGE award. This material is based upon work supported by the National Science Foundation under Grant No. 1032653.

References

1. Gao, P.X., *et al.* Conversion of zinc oxide nanobelts into superlattice-structured nanohelices. *Science*. **309** (5741), 1700-1704, doi:10.1126/science.1116495, (2005).
2. McIlroy, D.N., Zhang, D., Kranov, Y., Norton, M.G. Nanosprings. *Appl. Phys. Lett.* **79** (10), 1540-1542, doi:10.1063/1.1400079, (2001).
3. He, Y., *et al.* Multilayered Si/Ni nanosprings and their magnetic properties. *Small*. **3** (1), 153-160, doi:10.1002/sml.200600375, (2007).
4. Cammarata, R.C., Sieradzki, K. Surface and interface stresses. *Annu. Rev. Mater. Sci.* **24** (1), 215-234, doi:10.1146/annurev.ms.24.080194.001243, (1994).
5. Becker, N., *et al.* Molecular nanosprings in spider capture-silk threads. *Nat. Mater.* **2** (4), 278-283, doi:10.1038/nmat858, (2003).

6. Singh, J.P., Liu, D.-L., Ye, D.-X., Picu, R.C., Lu, T.-M., Wang, G.-C. Metal-coated Si springs: nanoelectromechanical actuators. *Appl. Phys. Lett.* **84** (18), 3657-3659, doi:10.1063/1.1738935, (2004).
7. Kim, K.J., Park, K., Lee, J., Zhang, Z.M., King, W.P. Nanotopographical imaging using a heated atomic force microscope cantilever probe. *Sens. Actuators A-Phys.* **136** (1), 95-103, doi:10.1016/j.sna.2006.10.052, (2007).
8. Dobrokhotov, V., *et al.* ZnO coated nanospring-based chemiresistors. *J. Appl. Phys.* **111** (4), 044311-044318, doi:10.1063/1.3686212, (2012).
9. Sai, V.V.R., *et al.* *Nanotechnology 2010: Bio Sensors, Instruments, Medical, Environment and Energy*. Chapter 1: Bio Sensors, Diagnostics & Imaging, 19-22 (2010).
10. Sai, V.V.R. *et al.* Silica nanosprings coated with noble metal nanoparticles: highly active SERS substrates. *J. Phys. Chem. C.* **115** (2), 453-459, doi:10.1021/jp109586f, (2010).
11. da Fonseca, A.F., Galvão, D.S. Mechanical properties of nanosprings. *Phys. Rev. Lett.* **92** (17), 175502-175505, doi:10.1103/PhysRevLett.92.175502, (2004).
12. Zhang, G., Zhao, Y. Mechanical characteristics of nanoscale springs. *J. Appl. Phys.* **95** (1), 267-271, doi:10.1063/1.1630699, (2004).
13. da Fonseca, A.F., Malta, C.P., Galvão, D.S. Mechanical properties of amorphous nanosprings. *Nanotechnology.* **17** (22), 5620-5626, doi:10.1088/0957-4484/17/22/015, (2006).
14. Mohedas, I., Garcia, A.P., Buehler, M.J. Nanomechanics of biologically inspired helical silica nanostructures. *Proceedings of the Institution of Mechanical Engineers, Part N: J. Nanoengineering and Nanosystems.* **224** (3), 93-100, doi:10.1177/1740349911411234, (2010).
15. Chang, I.L., Yeh, M.-S. An atomistic study of nanosprings. *J. Appl. Phys.* **104** (2), 0243051- 0243056, doi:10.1063/1.2951478, (2008).
16. Poggi, M.A. *et al.* Measuring the compression of a carbon nanospring. *Nano Lett.* **4** (6), 1009-1016, doi:10.1021/nl0497023, (2004).
17. Chen, X. *et al.* Mechanics of a carbon nanocoil. *Nano Lett.* **3** (9), 1299-1304, doi:10.1021/nl034367o, (2003).
18. Dávila, L.P., *et al.* Transformations in the medium-range order of fused silica under high pressure. *Phys. Rev. Lett.* **91** (20), 2055011-2055014, doi:10.1103/PhysRevLett.91.205501, (2003).
19. Nick Gnedin's Ionization FRont Interactive Tool (IFRIT) v. 3.2.8 - A general purpose visualization software [Internet]. Chicago (IL): Nick Gnedin; [updated 2014 Mar 9; cited 2013 May 15]. Available from: <https://sites.google.com/site/ifrithome/> (2014).
20. Accelrys Inc. *Materials Studio Overview* [Internet]. [U.S.A.]: Accelrys; [cited 2013 May 15]. Available from: <http://accelrys.com/products/materials-studio> (2014).
21. Silva, E.C.C.M., Tong, L., Yip, S., Van Vliet, K.J. Size effects on the stiffness of silica nanowires. *Small.* **2** (2), 239-243, doi:10.1002/sml.200500311, (2006).
22. Dávila, L.P., Leppert, V.J., Bringa, E.M. The mechanical behavior and nanostructure of silica nanowires via simulations. *Scripta Mater.* **60** (10), 843-846, doi:10.1016/j.scriptamat.2008.12.057, (2009).
23. Doblack, B. N. *The structure and properties of silica glass nanostructures using novel computational systems [thesis]*. Merced (CA): University of California Merced; 55p., (2013).
24. MathWorks. *MATLAB Overview* [Internet]. [Massachusetts]: MathWorks; [cited 2013 May 30]. Available from: <http://www.mathworks.com/products/matlab/description1.html> (2014).
25. *Linux homepage* [Internet]. [U.S.A.]: Linux; [cited 2013 May 22]. Available from: <http://www.linux.org> (2014).
26. *Davila group website* [Internet]. Nanospring Models via MATLAB and NanospringCarver. Merced (CA): University of California Merced; [updated 2013 Aug 21, cited 2013 May 1]. Available from: <https://eng.ucmerced.edu/people/ldavila/home/nanospring-models-via-matlab-nanospringcarver-dissemination-of-research-results-and-products> (2014).
27. *Blinkdagger - An Engineering and MATLAB blog* [Internet]. MATLAB GUIDE Tutorial for Beginners. [place unknown]: Blinkdagger; [cited 2013 May 20]. Available from: <http://tinyurl.com/p2gqen6> (2014).
28. MathWorks. *Introduction to MATLAB GUIDE* [Internet]. [Massachusetts]: MathWorks; [cited 2013 May 30]. Available from: http://www.mathworks.com/help/matlab/creating_guis/about-the-simple-guide-gui-example.html (2014).
29. MathWorks. *Use and create MATLAB MEX-files* [Internet]. [Massachusetts]: MathWorks; [cited 2013 May 30]. Available from: <http://www.mathworks.com/help/matlab/call-mex-files-1.html> (2014).
30. MathWorks. *Lay out the simple GUI in MATLAB GUIDE* [Internet]. [Massachusetts]: MathWorks; [cited 2013 May 30]. Available from: http://www.mathworks.com/help/matlab/creating_guis/lay-out-the-simple-gui-in-guide.html (2014).
31. MathWorks. *Add components to the MATLAB GUIDE layout area* [Internet]. [Massachusetts]: MathWorks; [cited 2013 May 30]. Available from: http://www.mathworks.com/help/matlab/creating_guis/adding-components-to-the-gui.html (2014).
32. LAMMPS (Large-scale Atomic/Molecular Massively Parallel Simulator) molecular dynamics code [Internet]. Albuquerque (NM): Steve Plimpton; [updated 2014 Feb 1; cited 2013 May 30]. Available from: <http://lammps.sandia.gov> (2014).