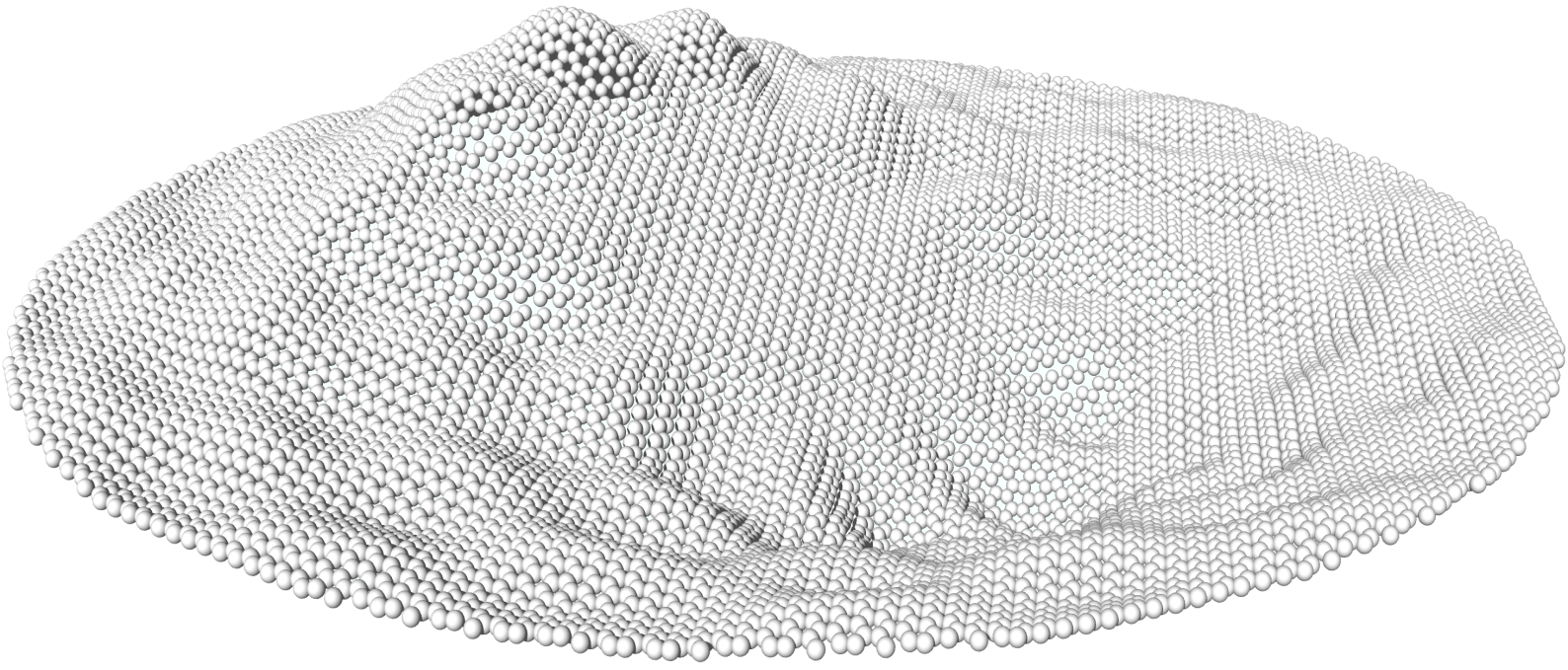
Department of Precision and Microsystems Engineering

Extracting elastic properties of graphene nanodrums using a multi-modal approach in Molecular Dynamics

Simon van Hemert

Report no : 2017.053
Coach : Dr. Farbod Alijani
Professor : Prof.dr. Peter G. Steeneken
Specialisation : Dynamics of Micro and Nanosystems
Type of report : MSc Thesis
Date : October 17, 2017



Extracting elastic properties of graphene nanodrums using a multi-modal approach in Molecular Dynamics

by

Simon van Hemert

to obtain the degree of Master of Science
at the Delft University of Technology,
to be defended publicly on Thursday November 9, 2017 at 9:30 AM.

Student number:	4149572	
Project duration:	Oktober 2016 – November 2017	
Thesis committee:	Prof. dr. P. G. Steeneken,	TU Delft, Chairman
	Dr. F. Alijani,	TU Delft, Supervisor
	Dr. C. Ayas,	TU Delft
	Dr. B. Arash,	TU Delft

An electronic version of this thesis is available at `http://repository.tudelft.nl/`.

TU Delft

Preface

Over a year of hard work is now concluded by writing these last words on this thesis. I am proud of the result, and hope I managed to contribute something small.

This would not have been possible without the help and supervision by Farbod. I also want to specially thank Banafsheh and Pierpaolo for their contribution, and Behrouz for his help in understanding Molecular Dynamics. Of course there are many others which contributed, too many to name here.

Then, I want thank my friends, family, flat mates and fellow students for all the good times, Elena, for making everything better, and finally, Lotta, for bringing me to the best of life.

Simon van Hemert
Delft, October 2017

Abstract

The field of nanotechnology has been quickly growing over the last few decades and many different functional Nano Electro Mechanical Systems(NEMS) can now be made. To further increase the possibilities and functionality of NEMS, new materials have to be characterized. Of the new materials which become available at the nanoscale, graphene is one of the most promising. This is mainly due to the combination of its extraordinary strength, electric and thermal conductivity, and low weight. In order to be able to use graphene's full potential in future applications, the elastic properties, such as the Young's modulus and the bending rigidity, have to be known.

These elastic properties have been obtained following different approaches. However, the obtained values are scattered. This scattering has a few reasons. Firstly, experimental research into graphene is difficult, due to the small scale, big influence of the environment, and the difficulty of fabricating well defined graphene membranes. Secondly, graphene is a purely two-dimensional material. Therefore, continuum theory is not easily applied. The bending rigidity for example is not related to thickness as it is in a continuum plate. Finally, graphene exhibits strong mode coupling and is always vibrating due to Brownian motion, the motion which results from the stochastic excitation due to temperature. Parameters extracted from static measures may thus not match the reality or experiments. In conclusion, the elastic properties have to be obtained from the dynamic response, following a multi-modal approach.

As graphene, only one atom thick, is close to the atomic scale, Molecular Dynamics simulations are used to investigate its behavior. To extract parameters, these simulations are compared to an analytical continuum model. In this way, the advantages of continuum mechanics and Molecular Dynamics simulations are combined with a dynamic, multi-modal approach to extract the bending rigidity and Young's modulus of graphene.

In the continuum model, the equations of motion of a circular graphene plate and membrane are derived from a Lagrangian approach. The governing equations are discretized using admissible functions that satisfy boundary conditions. Furthermore, the geometric nonlinearity is included, as graphene is so thin, and is thus easily driven into the nonlinear regime.

The basic principle of Molecular Dynamics simulations is to solve Newton's equation of motion for every single atom of a system. The force acting on the atoms is described by a potential field. The equations of motion are then integrated over time.

The mode coupling in graphene is shown to be so strong that the energy in all modes is equal after some time. Therefore, the only steady state attainable is the state in which the energy in all modes is equal, which corresponds to the Brownian motion. The eigenfrequencies are obtained from the time response of the atoms in the graphene membrane, excited by Brownian motion. The obtained eigenfrequencies are compared to the values obtained from the continuum model. From this comparison the bending rigidity of graphene is extracted. This is done following an optimization approach, which minimizes the difference between the eigenfrequencies obtained from the continuum and the Molecular Dynamics model. Including multiple modes is shown to be a necessity for reaching convergence. Furthermore, the Young's modulus is obtained. This is done by comparing the geometric nonlinear behavior of graphene obtained in Molecular Dynamics with a continuum prediction of this behavior.

The bending rigidity and the Young's modulus thus have been obtained, independently, from the dynamic response of graphene obtained in Molecular Dynamics.

Contents

1 Introduction

This chapter gives an introduction to the problem. The background and motivation will be explained based on literature. Finally, the research question will be formulated.

1.1. Problem statement

The field of nanotechnology has been quickly growing over the last few decades and many different functional devices can now be made. These Nano Electro Mechanical Systems are indispensable to our daily lives; they are used in mobile phones, printers, air-bags, sensors and many more devices. Now, the challenge is to further increase the possibilities and functionality. Therefore, new materials should be investigated, produced and characterized.

Of the new materials which become available at the nanoscale, graphene is one of the most promising. This is mainly due to the combination of its extraordinary strength, electric and thermal conductivity and low weight. Graphene is claimed to have a Young's modulus of 1 TPa, which makes it the strongest material in nature [14]. To illustrate the extreme strength versus weight, it was calculated that a graphene drum would only break under its own weight for a diameter of 13 520 km, comparable to the diameter of the earth [19]. Moreover, graphene has superior electrical conductivity which allows for electronic readout in case of sensing applications. Furthermore, graphene is a truly 2D crystal, which creates new possibilities and room for fundamental research. In order to be able to use the full potential of this material however, its behavior and properties have to be well understood and characterized.

The characterization of graphene is not fully completed yet, as the found parameters are scattered. This is mainly due to difficulties in using classical research methods for graphene. As graphene is only one atom thick, it is very close to the atomic scale. Therefore, continuum theories can not always describe the behavior of graphene well. For example the bending rigidity of graphene is not dependent on the thickness and stiffness, as it would be in continuum plate theory. Furthermore, experimental research on graphene is difficult due to the small scale. For example, the eigenfrequencies scale with the inverse square of the length, and can easily reach the GHz regime, which makes measuring these very difficult.

In conclusion, in order to move forward in the field of nanotechnology, further research into graphene is needed. This research should be focused on extracting material properties and characterizing the behavior of graphene.

1.2. Literature review

The use of new materials lead to the need for new modeling methods. Partly because the continuum assumption may not hold valid anymore when close to the atomic scale, but mainly because graphene is not as easily characterized as a continuum body. Many methods have already been developed and used over the past decade, and some of these will be shortly discussed here.

1.2.1. Experimental research of graphene

Since the first graphene sheets were produced by mechanical exfoliation in 2004 by Geim, Novoselov and coworkers [23], many experiments have been done. The elastic properties have been measured experimentally

by Lee et al., who reported the Young's modulus to be 1 TPa [14]. The thermal conductivity was measured to be approximately 5×10^3 WmK^{-1}, which is an extremely high value, by Balandin and co-workers [4]. Bunch et al. showed that a single layer graphene is in fact impermeable to standard gases, including Helium [6]. Lindahl et at obtained the bending rigidity of double-layer graphene from the snap-through behavior of buckled graphene membranes under the application of electrostatic pressure [15]. Davidovikj et al. looked into the dynamic behavior of graphene and visualized the motion of a graphene membrane, showing the mode shapes [10]. Brownian motion was used to calibrate the local displacement.

Indeed, many interesting findings have been done in experimental research. This field will always be indispensable in investigating new materials, such as graphene. However, there are three main problems with experimental research into graphene. Firstly, it can be hard to control the environment. A single dust particle on a graphene membrane for example, has a significant influence on the effective mass. Secondly, fabrication of well defined graphene membranes is still difficult, and is a topic of research on itself. Finally, when moving to smaller scales, measuring using conventional techniques becomes very difficult, as the eigenfrequencies enter the GHz regime. Therefore, much research in this field has been done on a more abstract level, including analytical models and atomistic simulations.

1.2.2. Continuum modeling of graphene

Continuum mechanics is the standard tool for investigating vibrations and dynamics on the macro-scale, but is also often used on the micro-scale, possibly with certain modifications. Eriksson et al. investigated the non linear response of a circular nanomechanical graphene resonator, using continuum theory for membranes [13]. Dolleman et al. used continuum mechanics to show the change in frequencies of a graphene nanodrum under external pressure [12]. They showed that as the membrane is stretched by the static deflection caused by the pressure, the stiffness increases. On a more fundamental level, Croy et al. used continuum mechanics to investigate the nature of damping in graphene [9]. They found that coupling between flexural modes and in-plane phonons leads to linear and non-linear damping of out-of-plane vibrations. Besides the experimental effort already named before, Lindahl et al. extensively used classical continuum mechanics to predict the snap through behavior and to extract the bending rigidity from the performed experiments [15]. Tapasztó et al. investigated the periodic rippling of suspended graphene membranes, and showed that the nanorippling mode violates the continuum model [29], hereby proving one of the limits of continuum modeling.

In conclusion, continuum modeling is an often used method for the analysis of graphene membranes, especially in combination with other methods, such as experiments or molecular simulations. The main strength of continuum mechanics in modeling graphene membranes is the ease-of-use, computational efficiency and ability to give insight in scaling effects and relations between variables. The main drawbacks however are the inaccuracy at smaller scales, where the continuum assumption does not hold valid anymore, and the difficulty of handling purely two-dimensional materials such as graphene, as the thickness can not be defined easily.

1.2.3. Molecular Dynamics modeling of graphene

Many researchers have investigated graphene by use of atomistic simulations, such as Molecular Dynamics. Static stress-strain tests have been performed in order to measure the Young's modulus and identify the failure mechanisms of a perfect graphene sheet by Marianetti and Yevick [20]. The effective bending stiffness has been investigated by Liu and Zhang, showing that the bending stiffness of graphene is related to the temperature [18]. Midtvedt et al. used Molecular Dynamics simulations to investigate the intrinsic mode coupling and thermalization of graphene membranes, and related these findings to nonlinear mode coupling in a system of Duffing oscillators [21]. Arash et al. use d Molecular Dynamics to investigate the application of graphene membranes for detection of noble gases via a vibration analysis [3]. Xu and Liao compared the deflection shapes of a graphene membrane under a transverse center load obtained using molecular dynamics, continuum theory, or finite element method [33]. The deflection shapes were found to differ strongly.

Molecular Dynamics modeling is a strong research method, and has been used often for research into graphene. The strength of Molecular Dynamics lies in its accuracy. With the correct potential field, complex systems can be accurately modeled. The fact that temperature is included in the simulations allows for investigating thermal effects such as Brownian motion[1] and thermal mode coupling. The main drawback is the computational cost, which restricts this method to smaller systems.

[1] Brownian motion in graphene is the out of plane motion resulting from the thermal vibrations of the atoms in the membrane

1.2.4. Other modeling methods for graphene

There is a wide range of modeling methods, of which the above two are at the extremes, continuum mechanics for the biggest scale and Molecular Dynamics at the atomic scale. Besides these two methods, there are many other modeling methods for graphene, of which some will be shortly discussed here.

A method which is working at an even smaller scale is the Density functional theory(DFT). This theory is based upon approximate descriptions of the electron densities, and is thus very close to quantum mechanics. Wei et al. used this method to find the bending rigidity of a graphene sheet from the energies of fullerenes and nanotubes [32]. Due to the high computational costs, this method is restricted to very small systems.

Nonlocal theories are also often suggested for small scale modeling of materials. These models aim to solve the scaling problem of continuum mechanics by applying non-local kernels for the state of stress. Ansari et al. estimated the non-local parameters for a graphene sheet [2]. The problem however is that the non-local parameter used in these models is dependent on many variables, such as the mode number, the temperature, and the boundary conditions. Therefore, these theories are generally not widely applicable.

Another group of methods is the Course-grained models. These models combine atomistic simulations with Finite element methods, in order to allow using atomistic based methods for larger systems. An example is the Quasi Continuum Theory, developed by E. B. Tadmor and coworkers in 1996 [28]. The basic idea is to combine atomistic simulations and continuum theory by choosing a set of atoms within a unit cell, which represent all atoms in that unit cell. The continuum body is than divided into a mesh, as one would when using normal FEM. Another example is the atomic-scale finite element method(aFEM), developed by Liu and coworkers [16]. Here, the unit cells are overlapping, which is used to include long body interactions. In these Coarse-grained modes, one looses accuracy for the gain of computational efficiency.

1.2.5. Spread in obtained elastic properties

The elastic properties of graphene obtained using different research methods are scattered. To illustrate this, Table 1.1 is presented. The obtained values indeed differ by up to 50 %. Obtained values for the Young's modulus are generally less spread, ranging from 1 TPa to 1.2 TPa, although here also much lower or higher values have been reported.

Table 1.1: Bending rigidity obtained using different research methods.

Bending rigidity	Method	Authors
1.52 eV	DFT	Sanchez-Portal et al. [27]
1.02 eV	Empirical potential	Tersoff [31]
1.2 eV	Experiments	Nicklow et al. [22]

1.2.6. Conclusion

The literature study has very briefly touched upon the different fields of research into graphene. The characterization of graphene is still not fully completed. Experimental research is not yet able to accurately characterize graphene at the nanometer scale, obtained values are scattered, due to the difficulties of investigating graphene experimentally. The continuum theory is not directly suitable for purely two-dimensional materials. Furthermore, mode coupling and thermalization are not easily studied in an analytical way. However, continuum mechanics can be very well used to investigate results obtained in experiments. Modeling techniques like quantum mechanical simulations and density functional theory are computationally too expensive to study a graphene membrane in its natural dynamical state. Material properties extracted from atomistic simulations are often based on static experiments.

On a more general level, it was found that graphene exhibits strong mode-coupling. Furthermore, due to its low out of plane stiffness, Brownian motion causes relatively strong vibrations. These two dynamic effects are generally not taken into account in research characterizing graphene.

1.3. Research Question

The literature study showed that graphene is an extremely promising and interesting material. However, in order to be able to use graphene's full potential in future applications, its material properties should be obtained. So far, the obtained values are scattered. Therefore, further research is needed. This research should be

focused on extracting elastic properties, such as the Young's modulus and the bending rigidity, and characterizing the behavior of graphene. Furthermore, as graphene is shown to exhibit strong mode coupling and is always vibrating due to Brownian motion, the parameters should be extracted from the dynamic response, including multiple modes.

A Molecular Dynamics study would be the best option, as it is combining the accuracy of atomic scale simulations with a reasonable computational efficiency. This means systems of realistic size can be studied using a very fundamental modeling method. In order to extract parameters, this model can be compared to an analytical continuum model.

In conclusion, the research question can be stated as follows:
Can the bending rigidity and Young's modulus of a circular graphene membrane be extracted from the vibration response obtained in Molecular Dynamics using a multi-modal approach?

2

Linear and nonlinear vibrations of graphene membranes based on Continuum Mechanics

In this chapter, the equations of motion of a circular graphene plate and membrane are derived using classical continuum theory. After the governing equations are obtained, the discretization of the equation using admissible functions that satisfy boundary conditions is discussed. Finally, the nonlinear vibration response is briefly touched upon.

This chapter is based on the book *Vibration of continuous systems* by Rao [25] and *Nonlinear vibrations and stability of shells and plates* by Amabili [1].

2.1. Equations of motion

As we consider a circular membrane, polar coordinates will be used. In general, the equations are a function of the radial in plane displacements, $u(r,\theta)$ and transverse, out of plane, displacements, $w(r,\theta)$. The tangential displacements $v(r,\theta)$ are neglected.

2.1.1. Lagrange equations

The Lagrange equations of motion are given by

$$\frac{\partial L}{\partial q_i} - \frac{d}{dt}\left(\frac{\partial L}{\partial \dot{q}_i}\right) = 0, \tag{2.1}$$

here L is the Lagrangian $L = T - U$, with T is the Kinetic energy and U is the potential energy, q_i are the generalized coordinates and $\frac{d}{dt}$ is the derivative with respect to time.

Now the kinetic and potential energy, T and U, will be derived in more detail. The kinetic energy is related to the mass and velocity of the structure:

$$T = \frac{1}{2}\rho h \int_0^{2\pi}\int_0^R (\dot{w}^2 + \dot{u}^2)\, r\, \mathrm{d}r\mathrm{d}\theta, \tag{2.2}$$

here ρ is the density and h the thickness, which is taken as the interlayer distance of multilayer graphene. $\dot{w}$ and $\dot{u}$ are the time derivatives of w and u.

The potential energy is related to the stretching and bending of the membrane, and can be written as:

$$\begin{aligned} U = \int_0^{2\pi}\int_0^R \Bigg[&\frac{Eh}{2(1-\nu^2)}\left(\epsilon_{rr}^2 + \epsilon_{\theta\theta}^2 + 2\nu\epsilon_{rr}\epsilon_{\theta\theta} + \frac{1-\nu}{2}\gamma_{r\theta}^2\right) \\ &+ \frac{1}{2}k\left(k_{rr}^2 + k_{\theta\theta}^2 + 2\nu k_{rr}k_{\theta\theta} + \frac{1-\nu}{2}k_{r\theta}^2\right)\Bigg]\, r\, \mathrm{d}r\mathrm{d}\theta, \end{aligned} \tag{2.3}$$

where E is the Young's modulus, ν the Poisson ratio and k the bending rigidity. The first part of the integral represents the energy related to strains, here $\epsilon_{\theta\theta}$, ϵ_{rr} and $\epsilon_{r\theta}$ are the normal and shear strains. The second part represents the energy related to the bending. Here k_{rr}, $k_{\theta\theta}$ and $k_{r\theta}$ give the changes in the curvature.

In plate theory, the bending rigidity k is related to the stiffness, poison ratio and thickness of the plate:

$$k = \frac{Eh^3}{12(1-\nu^2)}. \tag{2.4}$$

As the value of k scales with the third power of thickness, its value diminishes quickly with a decreasing thickness. Therefore, if a plate becomes very thin as compared to its length, the contribution of the stiffness due to bending can generally be neglected. In that case, the out of plane stiffness is dominated by the pre-tension applied to the thin plate. If one completely neglects the bending rigidity, a membrane model is obtained.

In case of graphene however, the bending stiffness is not directly related to the stiffness and thickness as described in Equation 2.4. In fact, the value predicted by Equation 2.4 is an order of magnitude higher than values obtained by phonon measurements [15]. A particular problem is that graphene does not have a clearly defined thickness. However, some bending stiffness is expected, resulting from the overlapping π-orbitals. To solve this problem, the bending stiffness can be implemented as a constant parameter, k.

If the Von Kármán hypothesis is used, the strains and changes in curvature of a thin plate can be written as:

$$\epsilon_{rr} = \frac{\partial u}{\partial r} + \frac{1}{2}\left(\frac{\partial w}{\partial r}\right)^2, \tag{2.5}$$

$$\epsilon_{\theta\theta} = \frac{\partial v}{r\partial\theta} + \frac{u}{r} + \frac{1}{2}\left(\frac{\partial w}{r\partial\theta}\right)^2, \tag{2.6}$$

$$\gamma_{r\theta} = \frac{\partial u}{r\partial\theta} + \frac{\partial v}{\partial r} - \frac{v}{r} + \frac{\partial w}{\partial r}\frac{\partial w}{r\partial\theta}, \tag{2.7}$$

$$k_{rr} = -\frac{\partial^2 w}{\partial r^2}, \tag{2.8}$$

$$k_{\theta\theta} = -\frac{\partial w}{r\partial r} - \frac{\partial^2 w}{r^2\partial\theta^2}, \tag{2.9}$$

$$k_{r\theta} = -2\left(\frac{\partial^2 w}{r\partial r\partial\theta} - \frac{\partial w}{r^2\partial\theta}\right). \tag{2.10}$$

Note that v is neglected in this approach. This assumption is strictly valid only for axisymmetric modes, but has a small influence on other modes.

2.1.2. Discretized equations

In order to solve the equations of motion, they have to be discretized. This means a displacement field as a function of a set of generalized coordinates q_i and the polar coordinates has to be chosen. In this way the equation of motion becomes a function of the generalized coordinates.

In case of a clamped circular membrane, the shape functions are related to each mode shape:

$$W_{mn}(r,\theta) = J_m(\omega_{mn}\frac{r}{R})\cos(m\theta), \tag{2.11}$$

where m is the number of nodal diameters, n is the number of nodal circles, and R the radius of the membrane. ω_{mn} is a known constant, and J_m is the Bessel function of the first kind. In case of a plate, the mode shapes are slightly different and can be described by

$$W_{mn}(r,\theta) = \left(I_m(\omega_{mn})J_m(\omega_{mn}\frac{r}{R}) - J_m(\omega_{mn})I_m(\omega_{mn}\frac{r}{R})\right)\cos(m\theta), \tag{2.12}$$

where I_m is the modified Bessel function of the first kind.

The full displacement field can now be approximated by the sum over the first n mode shapes and the corresponding generalized coordinates q_i;

$$w = \sum_{i=1}^{n} q_i W_i(r,\theta), \tag{2.13}$$

where W_i is chosen such that the dimensionless frequency ω_{mn} is always increasing.

The in plane displacement field can be approximated by a polynomial, given by:

$$u(r) = u_0 r + r(R-r)\sum_{i=1}^{n} q_i r^{i-1}, \tag{2.14}$$

here u_0 is the radial displacement due to the pre-tension n_0 as in $u_0 = \frac{n_0(1-\nu)}{Eh}$.

Note that in assuming shape functions for the out of plane displacement, one can considers either a plate or a membrane. In these two cases the corresponding mode shapes are well known. In the case of graphene however, neither the plate nor the membrane mode shapes are perfect, as graphene shows some bending rigidity, although much less than a plate.

Now the equations of motion can be discretized using Equation 2.13 and Equation 2.14. With this discretization, the equation of motion can be written as a function of generalized coordinates as:

$$\mathbf{M}\ddot{q} + \mathbf{K}q + \mathbf{K_3}q^3 = \mathbf{0}, \tag{2.15}$$

where $\mathbf{M}$ and $\mathbf{K}$ are the effective mass- and stiffness matrices respectively, describing the inertia and stiffness corresponding to each mode shape. $\mathbf{K_3}$ is the coefficient of cubic nonlinearity. q are the generalized coordinates, which are in fact the amplitudes of vibration of each mode shape.

The absence of a damping signifies that we are looking at an undamped system. Furthermore, the $\mathbf{0}$ term on the right hand side shows that there is no external force acting on the system, and we are thus looking at free vibrations. The cubic stiffness term $\mathbf{K_3}$ is resulting from von Karman geometric nonlinearities, introduced in Equation 2.5-2.7. This cubic stiffness term is of significant importance for graphene membranes. As graphene is only one atom thick, it can easily be driven into the nonlinear regime, where the amplitudes have a magnitude comparable to the thickness.

2.2. Linear and nonlinear eigenfrequencies

When neglecting the cubic stiffness $\mathbf{K_3}$, the system becomes linear. The linear eigenfrequencies can be directly determined from the eigenvalues of the matrix $\mathbf{M}^{-1}\mathbf{K}$ which can be obtained by solving:

$$\det\left(\mathbf{M}^{-1}\mathbf{K} - \mathbf{I}\omega^2\right) = \mathbf{0}. \tag{2.16}$$

The obtained linear eigenfrequencies were compared to a Finite Element Model, and the difference between the two was less than 0.5 % for any of the eigenfrequencies, for the full comparison see Appendix C. This also means that assuming the tangential displacement v to be negligible is valid.

The nonlinear eigenfrequencies are dependent on the amplitude of vibration, and can thus not be obtained as easily. In order to illustrate the meaning of this added cubic stiffness term and amplitude dependent eigenfrequency, we can look at a one-dimensional, normalized version of Equation 2.15:

$$\ddot{q} + \omega_0^2 q + \alpha q^3 = 0, \tag{2.17}$$

here ω_0^2 is defined as $\sqrt{k/m}$ and α is the coefficient of cubic nonlinearity. By assuming a solution of the form $q = Q\cos(\Omega t)$ equation Equation 2.17 becomes:

$$\left(-\Omega^2 + \omega_0^2 + \frac{3}{4}\alpha Q^2\right)\cos(\Omega t) + \frac{1}{4}\alpha Q^2\cos(3\Omega t) = 0. \tag{2.18}$$

Now, by balancing the coefficient of the left and right hand side of Equation 2.18 , one can show that:

$$\Omega = \sqrt{\omega_0^2 + \frac{3}{4}\alpha Q^2}. \tag{2.19}$$

This equation shows that the non-linear eigenfrequency Ω is indeed dependent on the amplitude of vibration Q, the linear eigenfrequency ω_0 and the coefficient α.

3

Molecular Dynamics

The basic principle of Molecular Dynamics(MD) simulations is to solve Newton's equation of motion for every single atom of a system. The force acting on the atoms is described by a potential field. This potential field consists of energies for covalent bond stretch, bending of bond angle, twist of dihedral angle, and non-bonded interactions such as Van der Waal's interaction and electrostatic interaction. The equations of motion are then integrated over time. As an illustration of the hexagonal structure of the atoms and to give an idea of the size of a 10 nm membrane, Figure 3.1 is presented. Note that the out of plane displacement is scaled.

In this chapter the Molecular Dynamics(MD) method will be explained. First, some important concepts in MD will be discussed. Then, the general set-up and contents of a MD simulation will be discussed.

This chapter is based on the book *Computational physics of carbon nanotubes* by Rafii-Tabar [24].

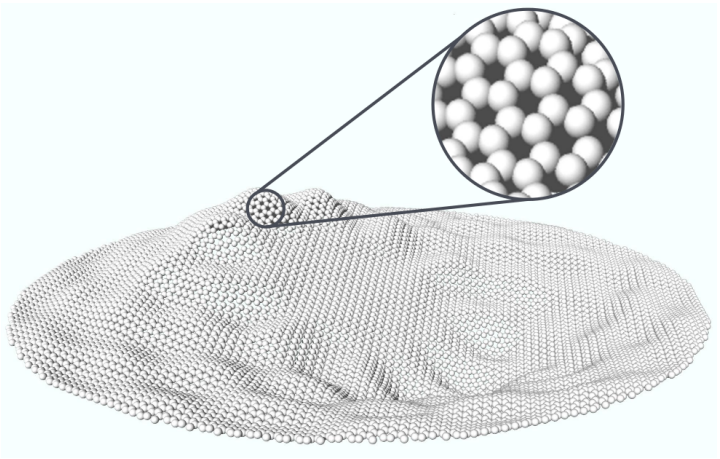


Figure 3.1: Vibrating graphene membrane of which part is enlarged to show the hexagonal structure.

3.1. Important concepts in Molecular Dynamics

In this section some important concepts in MD will be explained. Understanding of these concepts is needed to understand the underlying principles of the method. In section 3.2, these concepts will be placed in the framework of an MD simulation.

3.1.1. Potential functions

In order to compute the forces between atoms, a potential field must be defined. This interatomic potential describes the forces between atoms as a function of the atomic coordinates only. Potentials based on Quantum-mechanics or *ab initio* strategies do exist, but are too computationally expensive for systems bigger than a few hundred atoms. Therefore, interatomic potentials used in MD are essentially a simplified

model, usually based on Quantum simulations. These potential fields are thus designed to combine computational efficiency with accuracy. There are many different potential fields which can be used, but all fields can be expressed as a summation over two-body, three-body and possibly higher order body potential functions:

$$H_{\mathrm{I}} = \frac{1}{2!}\Sigma_i\Sigma_{j\neq i}V_2(\mathbf{r}_i,\mathbf{r}_j) + \frac{1}{3!}\Sigma_i\Sigma_{j\neq i}\Sigma_{k\neq i,j}V_3(\mathbf{r}_i,\mathbf{r}_j,\mathbf{r}_k) + ... \tag{3.1}$$

In this equation, H_{I} is the total potential energy, V_n are the n-body interatomic potential functions, as a function of atom coordinates $\mathbf{r}$. V_2 is thus the potential involving any pair of two atoms, such as covalent bond stretch. V_3 could include potential energies related to bending of a bond angle, as this involves 3 point interaction. Higher order interactions are usually negligible, and thus not included in the potential.

According to D.W. Brenner a potential field should posses the following four properties[5]:

1. **Flexibility**. A Potential Energy Function(PEF) must be sufficiently flexible that it accommodates a range as wide as possible for fitting data. For solid systems, the data might include crystalline lattice constants, cohesive energies, elastic properties, vacancy formation energies and surface energies.
2. **Accuracy**. A PEF should be able to accurately reproduce an appropriate fitting database.
3. **Transferability**. A PEF should be able to describe, at least qualitatively, if not with quantitative accuracy, structures that were not included in the fitting database.
4. **Computational efficiency**. Evaluation of the PEF should be relatively efficient.

The first two points are focused on the accuracy of the model and the extend to which it can describe a certain system. The third points states that the potential should be widely applicable. In this context, it is shown that analytic potential functions, based on quantum-mechanical principles, show better transferability than potentials based on experimental data, even though the latter usually includes more parameters. The last point, computational efficiency, is also of big importance, as it restricts the size of the problems one can study with a given computational power and time.

For modeling Carbon in diamond, graphite and nowadays graphene, the Tersoff potential is most widely used. Tersoff described this potential as early as 1988[30]. The parameters were chosen on basis of a fit of the cohesive energies, lattice constant and bulk modulus of diamond. Later the precise values were optimized, but the general form did not change since. In Figure 3.2 the bond energy and bondlenght are plotted versus the atomic coordination number, thus for different configurations of carbon polytypes. The atomic coordination number is the number of neighboring atoms of each atom. For example, the atomic coordination number is 3 for graphene and 4 for diamond. The squares are experimental values when available, or quantum mechanical calculations, the circles the result of the original Tersoff Potential.

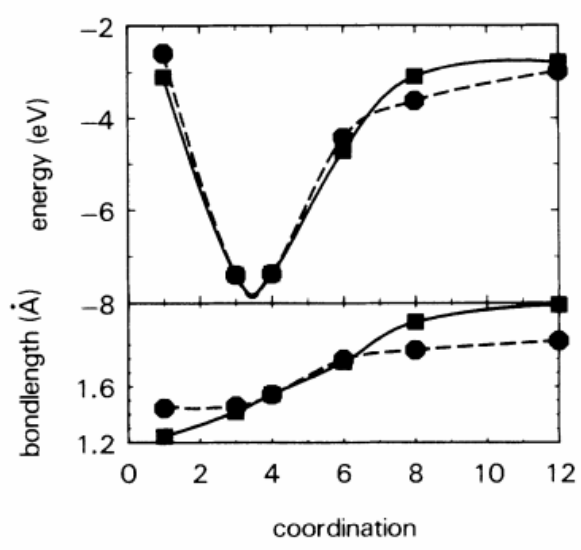


Figure 3.2: Energy and bondlenght vs atomic coordination number. Picture taken from [30]

3.1.2. Statistical-mechanical ensemble

An ensemble in this case is a collection of copies of a certain system. As many systems do not have a determined state, but are stochastic in nature, one can consider the given system along with an infinitely large number of fictional copies of it. These fictional copies are identically constraint systems independent of each other. Considering such an ensemble allows one to replace certain assumptions of the kinetic theory of gases by statements of statistical mechanics.

The subsystems of an ensemble are separated by boundaries. The boundary conditions basically define the nature of the ensemble. For example, one can imagine a perfectly isolated system, of which the number of atoms(N), volume(V) and Energy(E) are fixed. This system can be divided in infinitely many subsystems, which do allow for the exchange of N, V and E, but otherwise interact only weakly. The collection of these subsystems can be seen as the most general representation of the full system. In the same way other boundary conditions can be chosen. For example, one can consider a closed isothermal system, which exchanges energy with the surrounding environment in order to sustain the constant temperature(T).

This concept is of particular interest in MD as in these simulations time averages of thermodynamic state variables are computed, whereas in statistical mechanics ensemble-averages of these variables are computed. Obviously the time averages should be equal to the ensemble-averages. In order to achieve this, the time step should be small enough as to have minimal fluctuations over time.

3.1.3. Temperature at the atomic level

Temperature as known on the macro level is an overall quantity of the system. On the atomic scale however, temperature does not exist as such but is merely a measure of the movement of atoms around their average position. The velocity of an atom at a certain temperature does not have a fixed value and direction, but instead follows a probability distributions which is given by the Maxwell-Boltzmann distribution:

$$\rho(v_1) = \sqrt{\frac{m}{2\pi k_B T}} \exp\left(-\frac{m v_1^2}{2 k_B T}\right), \tag{3.2}$$

where $\rho(v_1)$ is the probability density for velocity in the x direction. m is the mass, T the temperature and k_B the Boltzmann constant. The same expression holds for the other directions. This means that in order to bring a system to a certain temperature, all atoms will have to be accelerated stochastically according to the Maxwell-Boltzmann distribution.

3.1.4. Relaxation of initial positions

In general one would like to investigate a material as it is found in reality. In giving the initial positions of atoms however, it is almost impossible to give the exact locations of all atoms. If a system starts a simulation in a state which is not the minimum potential state, it will start with a certain amount of potential energy which will be translated into kinetic energy. To avoid this, one can minimize the potential energy of the system. This relaxation is done by means of an optimization algorithm which aims to minimize the potential energy by changing the positions of atoms. Usually the convergence criteria are also related to the average total force between atoms. A converged minimization indicates that the system is at a (local) minimum potential energy state, as one would find it in reality at 0 K.

3.2. Model set-up

All MD programs have a similar structure. The main parts of this structure are summarized here, as well as how they are implemented in the model used in this investigation. For a full explanation of the used model, see Appendix A.

3.2.1. Initial conditions

The initial state of the system has to be provided. This means that the position and initial velocities of all atoms have to be defined. The type of atoms and their mass have to be defined as well.

In this case the considered geometry consist of a circular, flat, single layer graphene sheet. The atoms are ordered in a hexagonal grid with an interatomic distance of 1.42Å. The edges are fully clamped, which is achieved by restricting the translational degrees of freedom of three layers of atoms along the boundary. The radius of the considered drum is 10 nm. The boundaries of the simulation box are shrink-wrapped, which means they are not periodic, but can move such that atoms can not leave the simulation box.

Potentials
Next, a potential has to be chosen. The importance of an accurate potential is evident. Which potential is best for a certain case is dependent on the material, computational power available, needed accuracy and what sort of physical phenomenon is investigated, for example crack propagation, phase transition or mechanical vibrations.

In this research, the Tersoff BNC(Boron, Nitride, Carbon) potential is used. This potential contains the specific set of parameters suitable for studying these atoms and specifies the C-C interaction. This potential has been widely used for modeling Carbon in diamond, graphite and graphene.

3.2.2. Relaxation

After the potential is set, the system can be relaxed using a relaxation algorithm. This algorithm will minimize the potential energy of the system. Convergence criteria are based upon the change in potential energy between iterations as well as absolute value for the maximum force on an atom.

The minimization is done by the Polak-Ribiere version of the conjugate gradient algorithm. The used stopping criteria are 1×10^{-10} eV for the potential energy and 1×10^{-10} eVÅ^{-1} for the force. The potential energy per atom of the relaxed geometry equals -7.9613 eV. While relaxing the system, the out of plane coordinates are fixed, to prevent curling of the membrane.

3.2.3. Thermalization

Now the system can be thermalized. This is usually done in an NVT environment, such that the number of atoms, volume and temperature are kept constant. In order to keep the temperature constant, there is an energy flow from the environment into the system in the form of an acceleration on each atom. In order to reach a certain temperature, all atoms must be slowly actuated. After a certain number of time steps, the thermal energy of the system is evaluated and the velocities are rescaled accordingly. This is done by equating the kinetic energy of all particles to the thermal energy, according to:

$$\frac{1}{2}\sum m_i v_i^2 = \frac{3}{2} k_\mathrm{B} T, \tag{3.3}$$

where m_i and v_i are the mass and velocities of each particle, k_B is the Boltzmann constant and T is the desired temperature. After a certain time this should have converged to the Maxwell-Boltzmann distribution(3.2), leading to the desired overall temperature.

While graphene is being thermalized, the thermal energy, having a high frequency but low amplitude, will partly be moved to kinetic energy with a lower frequency and higher amplitude. When heated up, graphene will thus develop Brownian motion. Therefore, it is important to give the system enough time to find its steady state in order to ensure a correct temperature.

Note that there will always be some fluctuation in temperature, as this is still a macroscopic observation of a stochastic effect rather than a microscopic quantity. For bigger systems however the fluctuations should converge to 0.

Here, the thermostatting is done for 50000 time steps of 1 fs. This is long enough to ensure a stable, converged temperature. During thermalization, the boundaries of the membrane are fixed. This means the membrane will be tensioned as a result from the negative thermal expansion of graphene.

3.2.4. Vibration

Finally the dynamic response of the system can be studied. This can be done by applying an initial velocity, displacement or force. Now the system needs to be isolated in the sense that there should be no change in the number of atoms(N), Volume(V) and energy (E). This is thus called the NVE ensemble. A time integration scheme is used, and the system moves forward in time with small time increments. In this way many things can be studied, for example vibrations.

In this research, the velocity-Verlet integration algorithm is used, with a time step of 1 fs. The membrane can be excited by applying an initial velocity profile, or the Brownian motion can be studied, without any excitation other than the applied temperature. The simulation time may differ between different studies. The atom coordinates are saved at least once every 500 time steps, which corresponds to 20 points per vibration period of the fifth mode shape.

4

Results and Discussion

In the previous chapters the CM and MD methods have been explained. This chapter will go into the results obtained in MD, as well as compare the results from MD to the values expected from CM. The findings will be compared to what is known from literature.

4.1. Method

For this research, a graphene membrane of different radii has been investigated using the MD program called LAMMPS(Large-scale Atomic/Molecular Massively Parallel Simulator). Lammps is a classical molecular dynamics simulation code designed to run efficiently on parallel computers. It is an open-source code, and has been used in a wide range of research areas.

First, a small model (r = 1 nm) was made, in order to show the feasibility and value of this research. After this was done, a bigger model of 10 nm radius was investigated. This is the scale on which all results are obtained. As a final check different sizes were tested in order to check for size dependent behavior of the obtained parameters, as to be sure that the findings are valid on a larger scale.

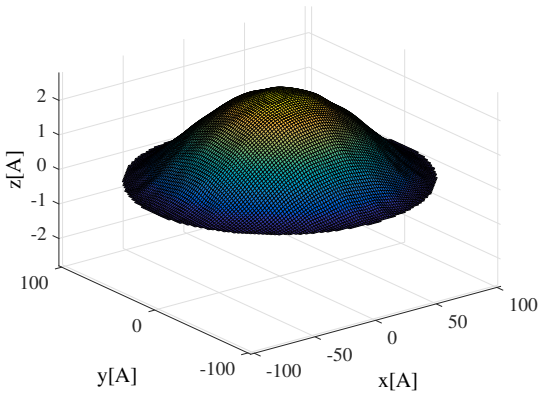
4.2. Free vibration response

In this section the free vibration response of the membrane is discussed. The membrane is actuated with an initial velocity, after which it is left to vibrate freely. From this free vibration, the mode shapes and eigenfrequencies can be extracted. As the research is focused on dynamical behavior, the mode shapes are of great importance. The mode shapes allow for a comparison with CM in terms of deflection shape. Furthermore, the frequencies corresponding to each mode are obtained, such that these can be compared to a CM model.

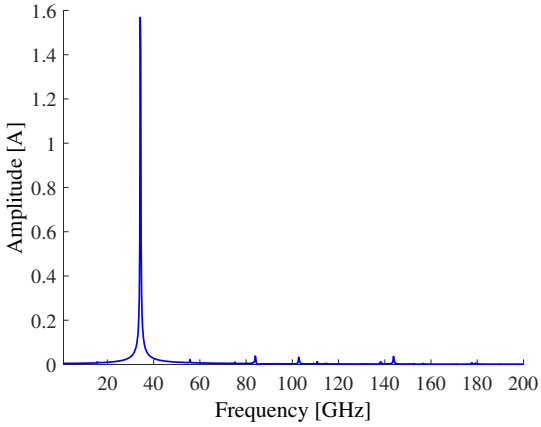
4.2.1. Mode shapes

First, the mode shapes and the corresponding natural frequencies of the first six modes of the membrane were extracted. This was done by actuating the membrane with an initial velocity profile with a maximum amplitude of 0.5 Åps^{-1} which resembled the mode shapes as known from a continuum membrane. After letting the system vibrate freely for 10 ns the mode shapes and eigenfrequencies were obtained. This was done at 5 K as to minimize the thermal noise. The resulting mode shapes as well as the frequency response are shown in Figure 4.1a-4.1i.

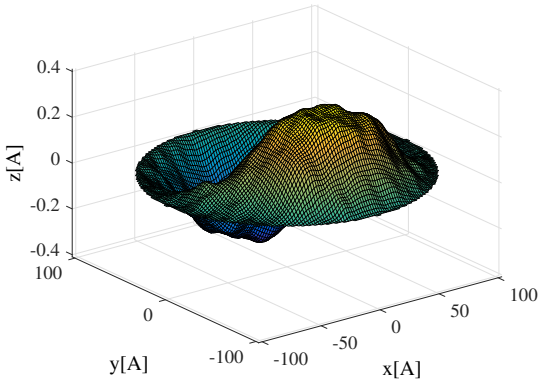
The first six mode shapes and eigenfrequencies can indeed be identified, although the third, fifth and sixth mode are clearly influenced by mode coupling. This mode coupling can be seen in the frequency response, where clear peaks indicating the other eigenfrequencies can be seen. This mode coupling was further investigated in section 4.3.


z[A]
2
1
0
-1
-2
100
0
-100
y[A]
-100
-50
0
50
100
x[A]

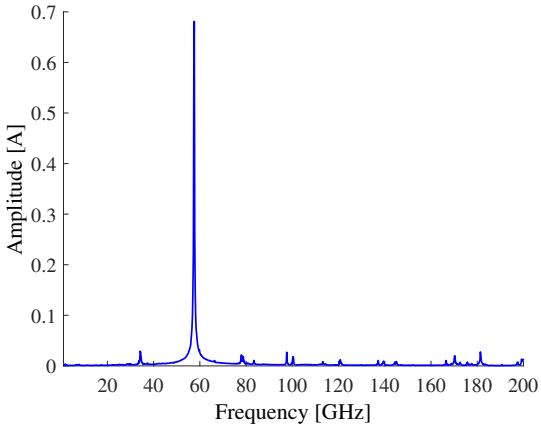
(a)


Amplitude [A]
1.6
1.4
1.2
1
0.8
0.6
0.4
0.2
0
20 40 60 80 100 120 140 160 180 200
Frequency [GHz]

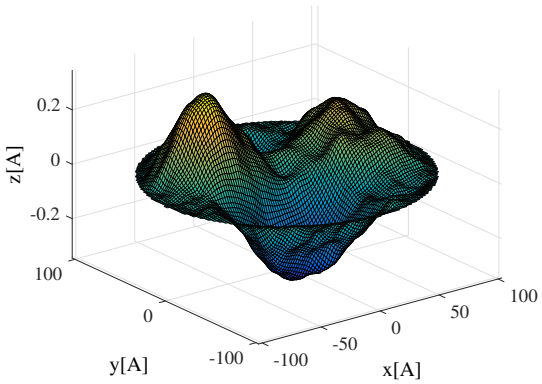
(b)


z[A]
0.4
0.2
0
-0.2
-0.4
100
0
-100
y[A]
-100
-50
0
50
100
x[A]

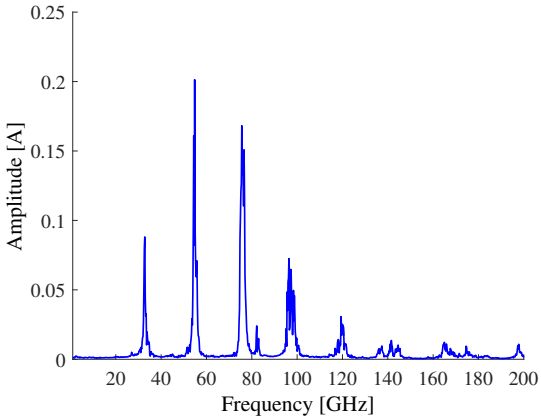
(c)


Amplitude [A]
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
20 40 60 80 100 120 140 160 180 200
Frequency [GHz]

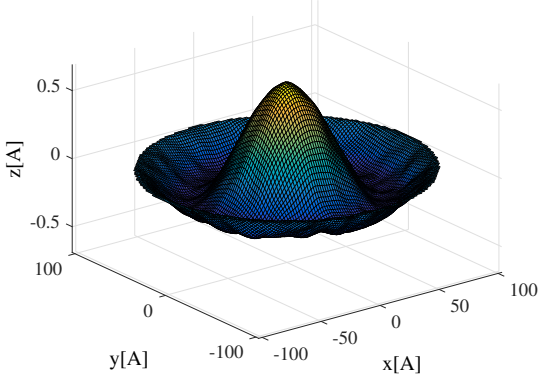
(d)


z[A]
0.2
0
-0.2
100
0
-100
y[A]
-100
-50
0
50
100
x[A]

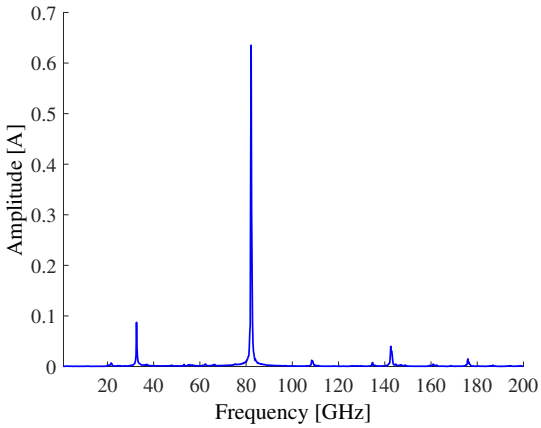
(e)


Amplitude [A]
0.25
0.2
0.15
0.1
0.05
0
20 40 60 80 100 120 140 160 180 200
Frequency [GHz]

(f)


z[A]
0.5
0
-0.5
100
0
-100
y[A]
-100
-50
0
50
100
x[A]

(g)


Amplitude [A]
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
20 40 60 80 100 120 140 160 180 200
Frequency [GHz]

(h)

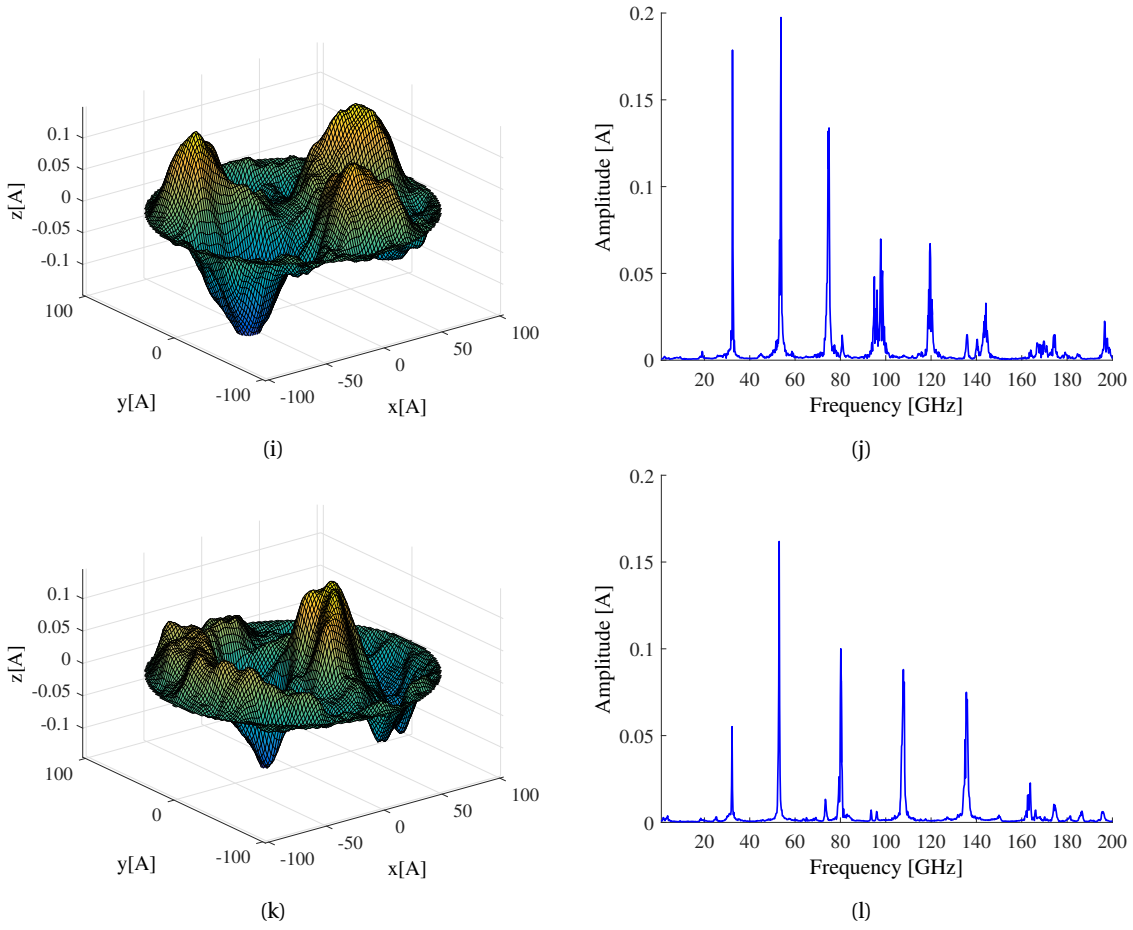


Figure 4.1: The left figures show the mode shape as obtained after 10 ns of free vibration at 5 K. The modes are actuated by an initial velocity profile of 0.5 Åps^{-1}. The right figures show the corresponding fast Fourier transform of the time response. (a) and (b) correspond to the first mode, (c) and (d) to the second mode, (e) and (f) to the third mode, (g) and (h) to the fourth mode, (i) and (j) to the fifth mode, and (k) and (l) to the sixth mode.

4.2.2. Comparison to Continuum Mechanics

Since graphene is only one atom thick, one would expect to find a mode shape very similar to a membrane mode in CM. Instead, the obtained mode shapes show some bending stiffness, which is best seen at the boundary. To illustrate this, cross sections of the first and second mode shapes are given in Figure 4.2a and 4.2b where they are compared to the CM membrane and plate modes.

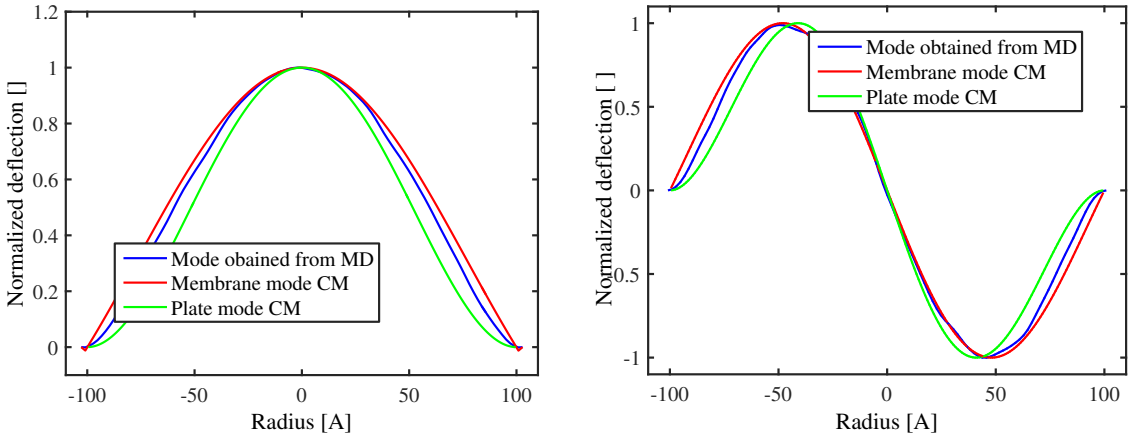


Figure 4.2: Comparing mode shapes obtained in MD with known mode shapes from CM for membranes and plates (a) first mode shape (b) second mode shape.

Indeed, the mode shapes obtained from MD are different from both the plate and membrane mode in CM. We can thus conclude that graphene shows some bending rigidity, although much less than a plate. This finding will be further investigated in section 4.5, where the bending rigidity, independent of Young's modulus or thickness, will be obtained.

Note that the mode shapes and eigenfrequencies are obtained by actuating the membrane with an initial velocity profile, corresponding to the mode shapes expected from membrane theory. The mode shapes obtained from MD however, differ slightly from these expected mode shapes. Therefore, the excitation is done using an incorrect velocity profile, and will also excite other modes. This may thus be part of the seen mode coupling. However, after 10 ns the eigenfrequencies do not change anymore over time, and therefore the obtained mode shapes are assumed to have reached a steady state, and are thus accurate.

4.2.3. Thermal noise

The thermal vibrations of atoms create an additional velocity. This induces mode coupling and thermal ripples. The thermal ripples, which will be further investigated and explained in section 4.4, can be seen as a noise term which affects the time and frequency response by exciting all modes. This can thus be used to extract the natural frequencies of the structure. The influence of the temperature on obtained mode shapes is visible in Figure 4.3 where the mode shapes obtained at 300 K are shown.

Indeed, the influence of thermal noise can clearly be seen from the obtained mode shapes. Part of this noise is the vibration of atoms and part of it is caused by the interference with other modes, which are excited and coupled through thermal energy.

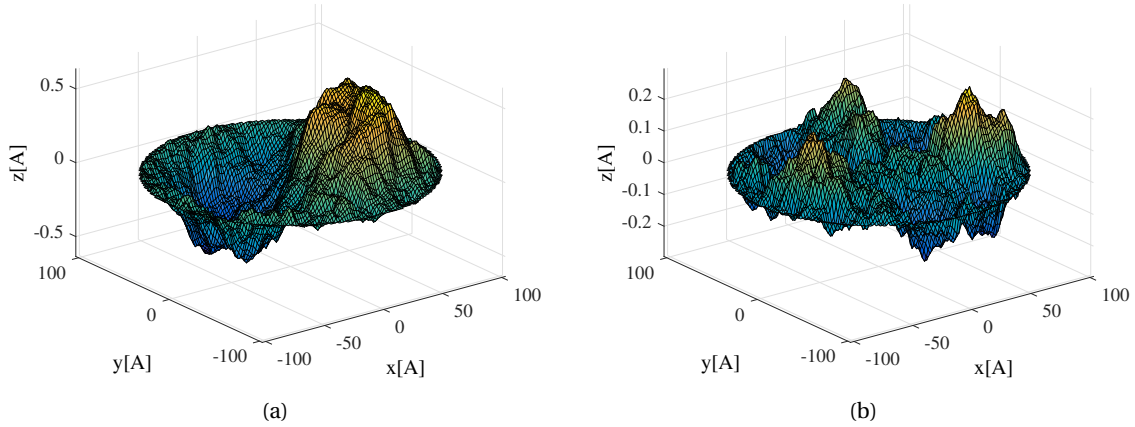


Figure 4.3: Influence of thermal noise on the mode shapes and eigenfrequencies obtained at 300 K (a) Second mode shape (b) Fifth mode shape.

4.3. Mode coupling and modal energy evolution

The effect of mode coupling was already seen in section 4.2. In Figure 4.1j it was shown that an excitation in the fifth mode leads to a vibration in all other modes. This effect is now studied in further detail.

4.3.1. Mode coupling in the frequency response

First the frequency content is investigated over time. This shows that all the modes couple quickly towards the first fundamental mode. The coupling between similar modes, such as axisymmetric modes, appears to be stronger than between different modes. For example, the coupling from first to fourth mode appears much stronger than the coupling between first and second mode. In Figure 4.4, the frequency is shown at different time instances. Here, the time response of the membrane is filtered to show only one mode. This is achieved by projecting the mode shape on the total displacement. The first mode is decreasing in amplitude during the simulation, whereas the other modes are steadily increasing. At the start of the simulation the first mode is so strong, that an harmonic can be seen at three times the first eigenfrequency.

4.3.2. Modal Energy evolution

Another way of looking at the mode coupling is by looking at the evolution of energy present in each mode over time. The energy per mode is obtained by combining the amplitude of vibration with the stiffness of that particular mode as in

$$E_i = \frac{1}{2} k_i A_i^2. \tag{4.1}$$

Here, k_i is the stiffness connected to mode i and A_i is the corresponding amplitude.

Now the energy in each mode is calculated over time after the first mode is actuated with a considerable initial velocity of 1 Åps^{-1}. The evolution of normalized energy per mode over time is shown in Figure 4.5.

Indeed, the normalized energy is decreasing in the first mode and increasing in all other modes. After 25 ns, the energy in all modes is comparable. In the last part of the simulation, the energy in all the modes is still fluctuating.

This actually means the only steady state attainable is the state of Brownian motion, where the energy in all modes is equal. Therefore, Brownian motion is the only good way of extracting eigenfrequencies.

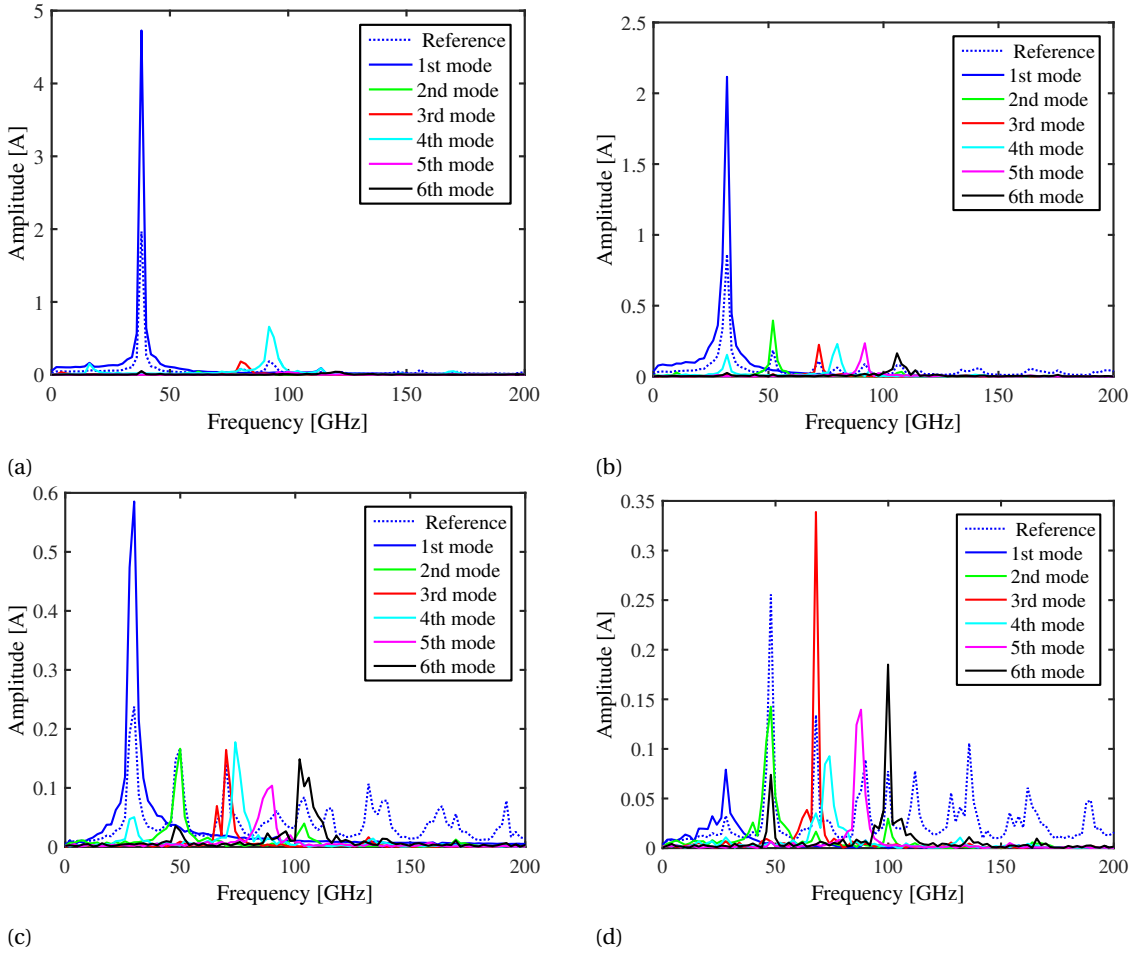


Figure 4.4: Filtered frequency response at different times, showing the FFT of the all atoms as well as of each mode separately. (a) Is taken at the start of the simulation, (b) at 10 ns,(c) at 25 ns,(d) at 45 ns.

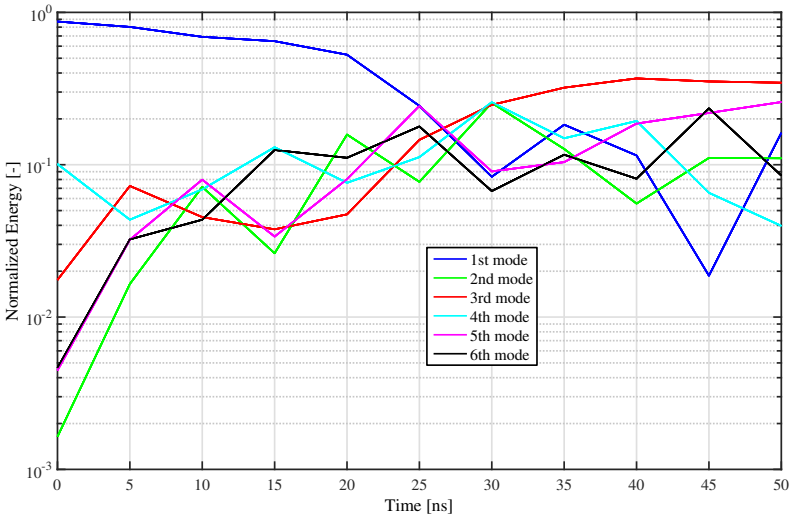


Figure 4.5: Energy evolution of the first six modes over 50 ns

4.4. Brownian motion

One way of comparing the results between MD and CM is by looking at the eigenfrequencies of the system. One can find the eigenfrequencies by actuating a certain mode, as discussed in section 4.2. The problem

however is that all modes are coupled through thermal mode coupling, as was discussed in section 4.3. This means that the steady state of the system is in fact the state in which the energy is divided over all modes. This state corresponds to the Brownian motion. This observation led to the idea to use the Brownian motion of the system to extract the eigenfrequencies. The main advantage is that reaching a steady state is now feasible. Furthermore, we do not have to impose anything on the system, it is simply a stochastic actuation.

4.4.1. Stochastic excitation

On the macro-scale stochastic excitation is a common method for investigating the dynamical behavior of systems. In this method the system is excited with a stochastic signal, for example white noise. This way all frequencies can be extracted. This does indeed correspond very closely to the thermalization, where all atoms are given a stochastic velocity according to the Maxwell-Boltzmann distribution (see Equation 3.2).

This method was used to extract the first six natural frequencies of the graphene membrane on different sizes, ranging from 1 nm to 20 nm as well as different temperatures. After thermalization in a NVT environment, the system was left to vibrate while conserving total Energy (NVE). Even though the actuation is fully stochastic, the system is not directly at steady state, as the energy has to be redistributed from a stochastic distribution to all the modes. Depending on size and temperature, the membrane is left to vibrate freely for a certain time before starting the measurement. The position (x,y,z) of a subset of all atoms is measured. From the out of plane time response of each atom the fast Fourier transform (FFT) is taken. The FFT's of all atoms are then averaged to find the amplitude per vibration frequency of the full system. Now one can easily identify at least the first six eigenfrequencies, though many more could be identified when increasing the simulation time.

4.4.2. Eigenfrequencies extracted from Brownian motion

The thermal ripples resulting from the Brownian motion are visualized in Figure 4.6.

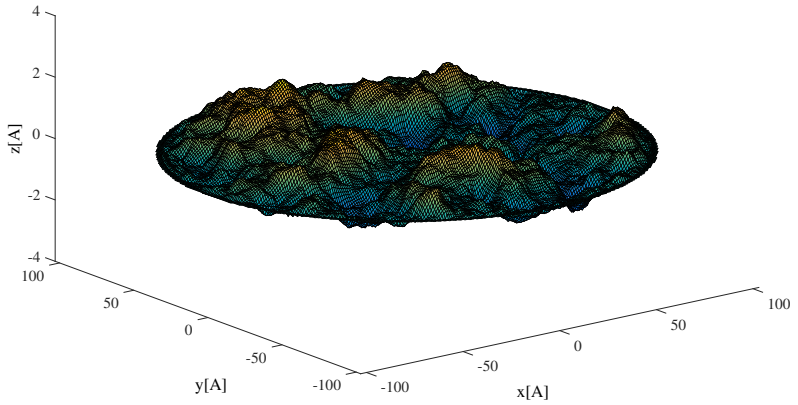


Figure 4.6: Snapshot of the membrane vibrating due to Brownian motion, obtained at 300 K for a 10 nm radius membrane

The Brownian motion results in a seemingly random vibration of the membrane. Furthermore, the amplitude of vibration at 300 K, about 2Å, is comparable to the thickness of the membrane, when considered as the interlayer distance of multilayer graphene (3.35Å). This is another problem of exciting a single mode in order to obtain the linear eigenfrequency, as the amplitude of vibration resulting from Brownian motion is already close to the nonlinear regime. The frequency response, shown in Figure 4.7, shows that the thermal ripples are in fact a superposition of all modes. Included in the figure are the obtained mode shapes corresponding to that eigenfrequency.

All eigenfrequencies can clearly be identified. If an eigenfrequency could not be identified with certainty, such as the fifth eigenfrequency in the presented figure, filtering the time response such that only one mode is left can solve this problem. The extracted frequencies compare well to the values given by the continuum model, as discussed in chapter 2. A comparison is made in Table 4.1.

The values obtained using the continuum model are obtained with the following parameters; thickness 3.35 Å, Poisson ratio is 0.16, density 2300 $\mathrm{kg\,m^{-3}}$, Young's modulus 1.04 TPa and pre-tension is 0.32 $\mathrm{N\,m^{-1}}$ as it is measured in MD at 300 K.

Note that the fifth eigenfrequency could not be identified at 5 K. This is because the temperature is so low that the thermal energy is very slowly redistributed. As the fifth mode is relatively stiff, the amplitude of

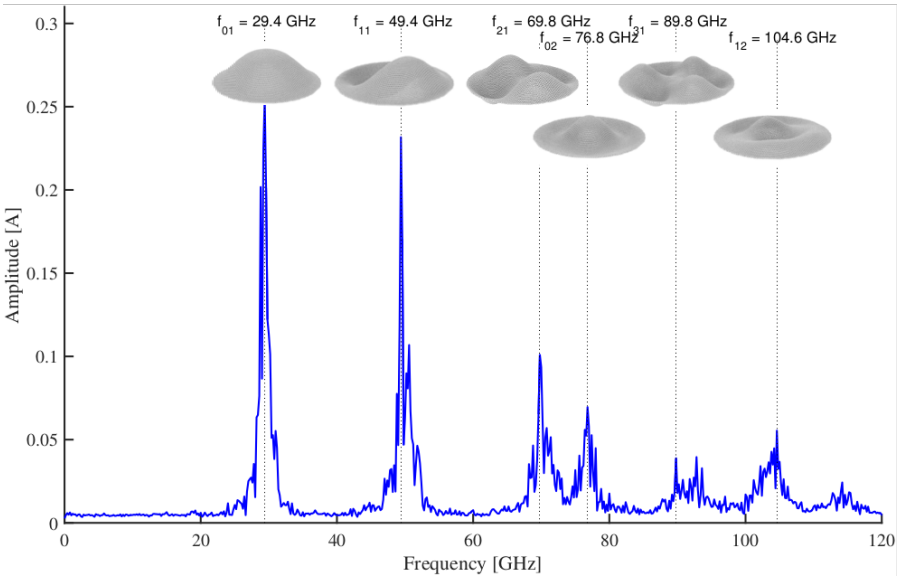


Figure 4.7: Frequency response obtained by Brownian motion. Eigenfrequencies are shown by a dotted vertical line, the corresponding mode shapes are shown in the pictures. Results are obtained at 300 K for a 10 nm radius membrane

Table 4.1: Comparing eigenfrequencies obtained from the Brownian motion with a continuum model

	Results Brownian motion				Results CM	
	5 K	100 K	200 K	300 K	Membrane	Plate
1st eigenfrequency	30.8 GHz	30.4 GHz	29.4 GHz	29.4 GHz	24.6 GHz	43.2 GHz
2nd eigenfrequency	50.8 GHz	50.4 GHz	49.4 GHz	49.4 GHz	39.2 GHz	82.0 GHz
3rd eigenfrequency	70.6 GHz	70.4 GHz	70.4 GHz	69.8 GHz	52.5 GHz	128 GHz
4th eigenfrequency	76.8 GHz	76.8 GHz	74.8 GHz	76.8 GHz	56.5 GHz	144 GHz
5th eigenfrequency	-GHz	90.4 GHz	89.0 GHz	89.8 GHz	65.2 GHz	183 GHz
6th eigenfrequency	103 GHz	103 GHz	102 GHz	105 GHz	71.8 GHz	215 GHz

vibration is low. Furthermore, the energy couples slower to the fifth mode than to other modes, as can be seen in Figure 4.5.

The values obtained seem to be independent of the temperature, which is very unexpected. As the pre-tension is solely coming from the thermal contraction of graphene, the pre-tension is linearly dependent on temperature. In Appendix B, subsection B.3.1 the measured stress on the boundary is shown as a function of temperature. Indeed, the stress is increasing linearly with temperature, which indicates an almost constant thermal expansion coefficient. In CM, the eigenfrequency of a membrane dependents on the square root of the pre-tension, in this case thus on the square root of the temperature. In fact, in membrane models in CM, the eigenfrequency is solely defined by the pre-tension. Therefore, in CM the obtained eigenfrequencies would be increasing from close to 0 at 5 K towards the reported value at 300 K.

A possible explanation could be that the pre-tension is so low that the stiffness is dominated by nonlinear stiffening. However, this would mean that the frequencies should be dependent on amplitude, which is not the case for such low amplitudes. In fact, the amplitude of vibration is also strongly dependent on temperature.

Another hypothesis is that in case of graphene, the temperature dependent stress on the boundary actually results from the vibrations of the membrane, rather than influencing these vibrations. The negative thermal expansion would then be caused by the out of plane vibrations of graphene by Brownian motion. This means the coupling is not from pre-strain to eigenfrequencies, but from eigenfrequencies to pre-strain. This again shows how graphene is not easily captured with the laws of continuum mechanics.

Furthermore, it can be seen that the eigenfrequencies obtained are of the same order of magnitude as the

values from CM. The values are higher than expected for a membrane, but lower than a plate. This corresponds well to the findings from section 4.2, where it was shown that the mode shapes of graphene obtained in MD show some bending stiffness, as opposed to membranes in CM.

4.5. Parameter extraction

Now the graphene membrane has been studied in depth, the next step is to see how well it compares to a continuum model and to extract parameters. As the scale we are looking at is so small, it is important to verify that the findings on this scale are representable for bigger sizes of membranes as well. Therefore, membranes from different radii, from 1 to 20 nm, are investigated.

4.5.1. Extracting the bending rigidity

The mode shapes as obtained in section 4.2 as well as the extracted eigenfrequencies indicate that graphene membranes have some bending rigidity. In order to find this bending rigidity, the first six eigenfrequencies as extracted using the Brownian motion are compared to the eigenfrequencies as obtained using a continuum model. The error, defined as:

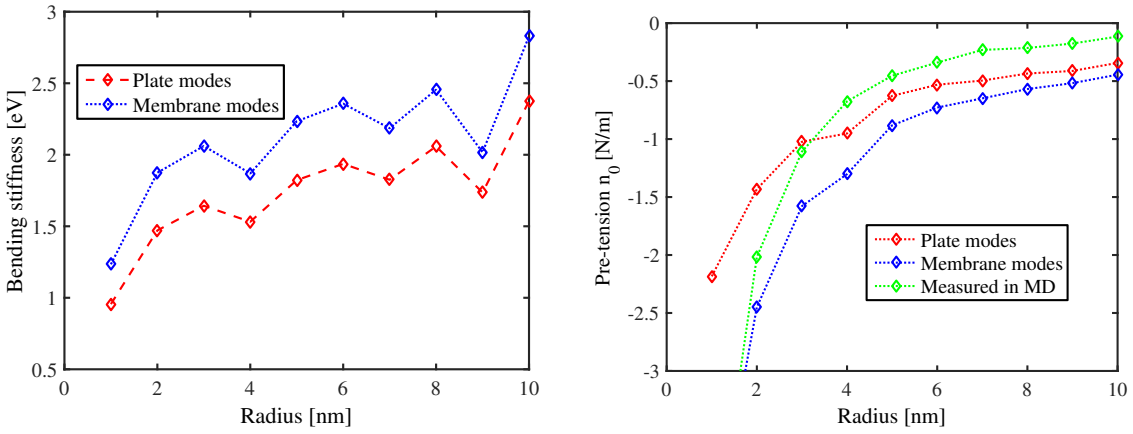
$$\mathrm{e} = \sqrt{\frac{\sum_{i=1}^{6}\left(\frac{\omega_i^{\mathrm{MD}}-\omega_i^{\mathrm{CM}}}{\omega_i^{\mathrm{CM}}}\right)^2}{6}}, \tag{4.2}$$

is then minimized using an optimization algorithm. Here, ω_i^{MD} and ω_i^{CM} are the MD and CM eigenfrequencies, respectively. There are two things which have to be fit; the ratio between higher eigenfrequencies and the first eigenfrequency and the absolute value of all eigenfrequencies. Therefore, there have to be two fitting parameters in order to allow for an accurate fit. These two fitting parameters would be the bending rigidity, k, expressed in eV and the pre-tension, $n0$, in $\mathrm{N\,m^{-1}}$. The eigenfrequencies following from the CM model are given by :

$$\omega_i = \sqrt{C_{i,1}k + C_{i,2}n_0 + C_{i,3}\sqrt{C_{i,4}k^2 + C_{i,5}kn_0 + C_{i,6}n_0^2}}. \tag{4.3}$$

The values for C_i are constants, dependent on radius, mode shape, Poisson ratio, thickness and mass density.

The obtained values for k and $n0$ are shown in Figure 4.8. The fit of the pre-tension is actually a measure



(a) (b)

Figure 4.8: (a) Obtained values for bending rigidity for radii from 1 to 10 nm at 300 K, comparing plate and membrane modes. (b) Obtained values for pre-tension, for radii from 1 to 10 nm, comparing plate and membrane modes as well as the values measured in Lammps.

of linear stiffness of the membrane, and would simply shift up and down the eigenfrequencies. The bending rigidity is a measure of how much the membrane behaves like a plate or a membrane, and influences the ratio between eigenfrequencies as well as the absolute values. As both values have a similar influence on the fit, the fitting becomes sensitive to small changes. This explains the noisy behavior which was seen even stronger for $r > 10$ nm (not shown here). In subsection B.3.2 the error value is plotted for different values of k and n0. The error is close to 1% for the fit values. The bending rigidity seems to converge to 1.8 eV when using plate

modes, and 2.4 eV when using membrane modes. The pre-tension converges to $-0.4\,\mathrm{N\,m^{-1}}$ for both plate and membrane modes.

To illustrate the importance of including multiple modes in a fit, Figure 4.9 is presented. As the higher modes are harder to obtain, the found values are less accurate. Therefore, a weighting function is used to weight the error on each eigenfrequency. These weights are a logarithmically spaced between 100 for the first mode and 10 for the tenth mode.

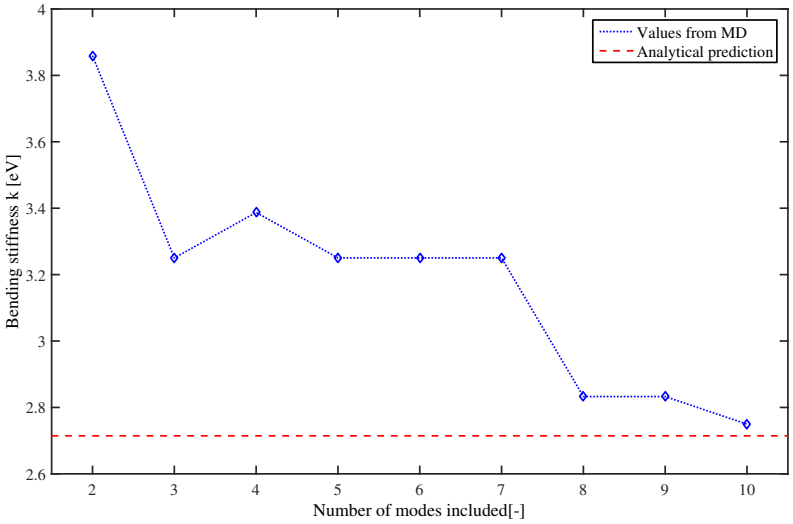


Figure 4.9: Obtained values for bending rigidity, when considering two up to ten eigenfrequencies, for a membrane of 10 nm radius at 300 K.

Indeed, the obtained value seems to converge when including multiple modes. Note that when one is using one mode only, one can only fit one parameter, and therefore this value is not included. Higher eigenfrequencies are harder to obtain, as the amplitude is decreasing. This is the main reason why higher order eigenfrequencies are not included for finding the bending rigidity. In Appendix B.3.2, the convergence with modes and the errors are further investigated.

The red dotted line gives the value predicted analytically by Rafael Roldán et al.[26]. In this paper, the dependency of the bending rigidity on the wave-vector and temperature was approximated. When the lower limit for all wave-vectors is chosen, the values correspond very closely, up to 5% at 300 K. The dependency on temperature is also corresponding fairly well, showing the same trend.

The obtained value of 1.8 eV for plate modes compares relatively well to the literature, although it is higher than the most reported value of 1.2 eV [18][32]. The main difference in research methods however, is that the bending rigidity is now obtained from a dynamical point of view, rather than a statical measurement.

Furthermore, including multiple eigenfrequencies as opposed to looking only at the fundamental resonance has shown to have an important influence on the derived value. This has never been done in literature, as to my best knowledge.

4.5.2. Size dependence bending rigidity

It was shown that the obtained values converge around 10 nm radius. Therefore, the derived values are valid for larger systems as well. For smaller radii, the obtained values can be interpolated, such that the size dependent parameters can be approximated.

4.5.3. Static stress-strain measure

In addition to the dynamical parameter extraction, a static stress-strain test is performed. The membrane is stretched radially and the resulting strain on the boundary is measured. The stress-strain curve is shown in Figure 4.10. The Young's modulus was found to be 1.03 TPa, which corresponds well to reported values.

The static stress-strain test has been reported several times[7][17], in particular by Marianetti and Yevick [20]. However, thus far, these tests were always performed on a rectangular configurations, which makes for an chiralty dependent value. In this case the membrane is symmetric, and the obtained value can thus be seen as an overall value. The derived Young's modulus is very close to reported values[8].

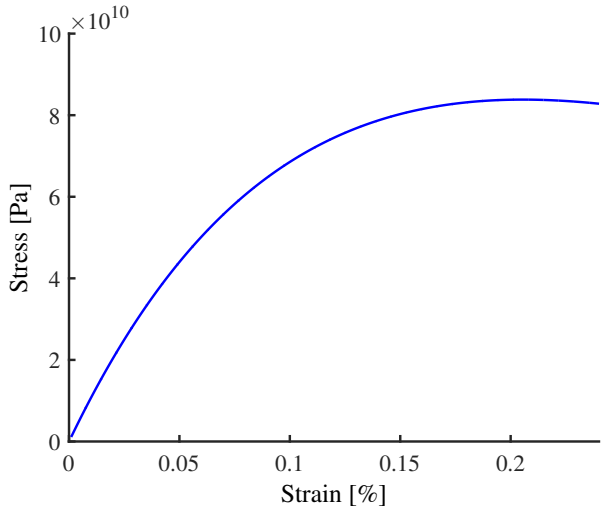


Figure 4.10: Stress-Strain curve obtained by static strain test. The obtained Young's modulus is 1.03 TPa

4.6. Nonlinear parameter fitting

Besides the linear eigenfrequencies and mode shapes, one can also look at the nonlinear response. As graphene is atomically thin, it is very easily driven into the nonlinear regime.

In order to obtain the nonlinear eigenfrequencies and backbone, the membrane was excited with a stepwise increasing initial velocity and the first eigenfrequency was measured together with the vibration amplitude. When combining these measured points, one can obtain a backbone curve. A continuum model was fit to the obtained curve in order to obtain the value for the Young's modulus.

As discussed in section 2.2, the equation of motion for a one dimensional geometric nonlinear system is given by:

$$\ddot{q} + \omega_0^2 q + \alpha q^3 = 0, \tag{4.4}$$

where α is the coefficient of nonlinearity. Now a curve can be made to fit the data. From the coefficient α the Young's modulus can be derived by the relation:

$$\alpha = C(\nu)\frac{Eh\pi}{R^2}, \tag{4.5}$$

where $C(\nu)$ is a dimensionless function which depends on the deformed shape of the membrane and the Poisson ratio [11]. The deformed shape can be approximated with a Bessel function, although in section 4.2 we have seen that this approximation is not perfect. h can be taken as the distance between different layers, which is 3.35 Å and R is the radius of the membrane. Thus, by fitting a curve which correctly describes the nonlinear behavior seen in the numerical experiments, one can find the value of the coefficient of nonlinearity and thereby extract the Young's modulus.

For a circular membrane of 10 nm radius at 300 K, the Young's modulus was obtained to be 1.08 TPa, which is very close to previously reported values, as well as the value obtained from a static stress-strain test. The measured points and fitted backbone are shown in Figure 4.11a and the made fit is shown in Figure 4.11b.

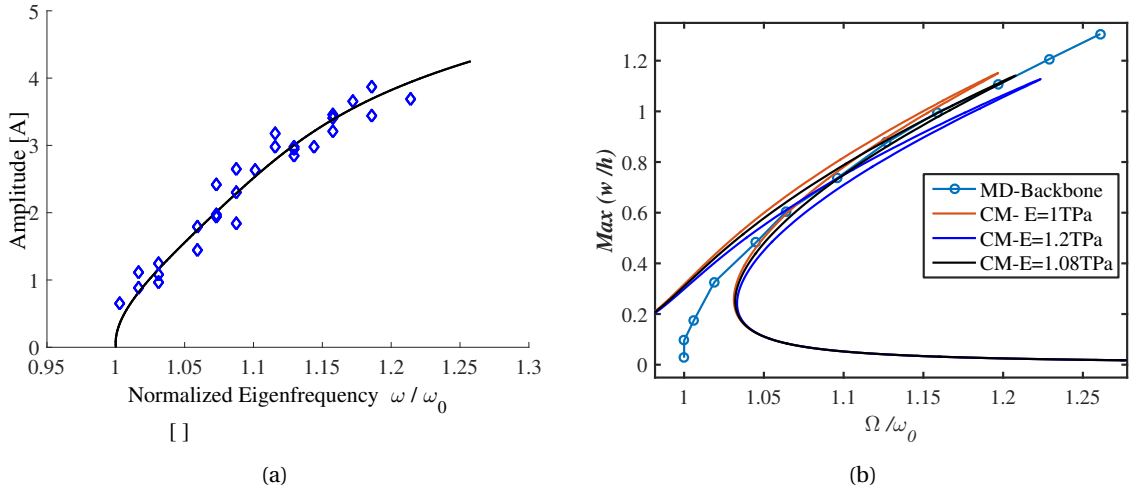


Figure 4.11: (a) measured points and fit backbone, (b) measured data together with the fit curve.

4.7. Conclusions

Now the conclusions drawn from the MD results as well as the comparison to CM and the parameter extraction are summarized.

The investigation into the free vibration response has yielded some important conclusions. Firstly, it has shown that the obtained mode shapes and eigenfrequencies appear in the same order, and that the spacing between eigenfrequencies in the frequency domain is comparable. This is needed in order to validate the comparison with a continuum model. Secondly, comparing the obtained mode shapes to the mode shapes from CM has shown that graphene indeed shows some bending rigidity, which is significant but less than expected from plate theory. Thirdly, an indication of the mode coupling was obtained.

The mode coupling, as was seen already in the free vibration response, was then investigated further. It was shown that all modes are coupled by thermal and nonlinear mode coupling. Therefore, it is impossible to look only at one mode in steady state. The mode shapes and frequency response obtained in free vibration were not obtained at steady state, but during a transition. Therefore, the only good way of finding all eigenfrequencies would be by looking at the Brownian motion, as the energy between all modes is evenly distributed in this case.

The eigenfrequencies were obtained from the Brownian motion of the graphene membrane. The eigenfrequencies were compared to the values expected from the CM model. This showed that eigenfrequencies, when normalized to the fundamental eigenfrequency, show higher ratios than those obtained for membranes. As the free vibration response has already showed that some bending rigidity is expected, the stiffening as compared to a membrane model is attributed to the existence of bending rigidity. Furthermore, it became apparent that the pre-strain on the membrane, resulting from thermal strain, does not influence the eigenfrequencies of the membrane.

Moreover, by utilizing the eigenfrequencies extracted from the Brownian motion, the bending rigidity of graphene was extracted and found to be 1.8 eV when using plate modes, and 2.4 eV when using membrane modes. This was done following an optimization approach, which minimizes the difference between the eigenfrequencies obtained from the CM model and the MD model. Including multiple modes was shown to be a necessity for reaching convergence. The obtained value was close to the values in literature, and corresponded very well to a theoretical approximation.

As a final step, the Young's modulus was obtained. This was done using a static stress-strain measure, which yielded a Young's modulus of 1.03 TPa. Furthermore, the Young's modulus was found by looking at the geometric nonlinear behavior. In this way, the Young's modulus was obtained as 1.08 TPa. These values are in very good agreement with present literature.

5
Conclusion

This chapter will conclude the performed research. The research question will be answered using the results discussed in the previous chapter. Then, recommendations for future research will be given.

5.1. Conclusion

The bending rigidity and Young's modulus of a circular graphene membrane have been extracted from the vibration response obtained in Molecular Dynamics, using a multi-modal approach. Besides this finding, this report also has contributed to fundamental understanding of the behavior of graphene.

5.1.1. Fundamental understanding

This research has made advances into better understanding graphene. The unique thermal behavior has been investigated. Thermal ripples were shown, the effect of thermal mode coupling was investigated and the connection between thermal expansion coefficient and dynamic behavior was touched upon. Furthermore, the mode coupling was investigated.

5.1.2. Characterizations

Some steps were taken into further characterizing graphene. In particular, the Young's modulus and bending rigidity were obtained, independently, from the dynamical behavior of graphene. The Young's modulus was extracted from the nonlinear behavior, by fitting a continuum model to the obtained backbone. The existence of a bending rigidity, was shown qualitatively by looking at the mode shapes. Furthermore, the value of this bending rigidity was obtained quantitatively by comparing the eigenfrequencies to an analytical model. An optimization algorithm was used, such that the first six frequencies were taken into account.

5.2. Recommendations

Few recommendations are made for future research. In general, graphene is still a rather unknown material, and much more research, both experimental and theoretical, is needed. Some directions are proposed here.

Firstly, to increase the accuracy of the found bending rigidity, the mode shapes found in MD could be used in the CM model. In this way, the energies would be calculated with respect to the real mode shapes, rather than assuming plate or membrane modes.

Secondly, it would be very interesting to further investigate the mode coupling. This fundamental property is strongly visible in graphene. Therefore, graphene offers a unique testing environment to investigate this effect. Besides the theoretical value, knowledge on this effect would be of great importance when designing graphene resonators.

Thirdly, the temperature dependence of the obtained parameters was not studied in sufficient depth. As the effect of temperature on the behavior of graphene is strong, it is expected that many of its properties will be temperature dependent. Therefore, this should be investigated further.

Fourthly, the relation between thermal expansion and the thermal vibration behavior is not fully understood, and should be investigated further. The fact that the eigenfrequencies seen in Brownian motion seem to be independent on pre-strain resulting from thermal expansion, is unique and interesting.

Finally, the obtained data can be used to fit a non-local parameter which would describe the behavior of a circular graphene membrane at very small sizes. This could be of use when designing extremely small graphene devices.

Bibliography

[1] Marco Amabili. *Nonlinear vibrations and stability of shells and plates*. Cambridge University Press, 2008. ISBN 1139469029.

[2] R. Ansari, S. Sahmani, and B. Arash. Nonlocal plate model for free vibrations of single-layered graphene sheets. *Physics Letters A*, 375(1):53–62, 2010. ISSN 03759601. doi: 10.1016/j.physleta.2010.10.028.

[3] Behrouz Arash, Quan Wang, and Wen Hui Duan. Detection of gas atoms via vibration of graphenes. *Physics Letters A*, 375(24):2411–2415, 2011. ISSN 03759601. doi: 10.1016/j.physleta.2011.05.009.

[4] Alexander A Balandin, Suchismita Ghosh, Wenzhong Bao, Irene Calizo, Desalegne Teweldebrhan, Feng Miao, and Chun Ning Lau. Superior thermal conductivity of single-layer graphene. *Nano letters*, 8(3): 902–907, 2008. ISSN 1530-6984.

[5] DW Brenner. The art and science of an analytic potential. *physica status solidi(b)*, 217(1):23–40, 2000. ISSN 0370-1972.

[6] J Scott Bunch, Scott S Verbridge, Jonathan S Alden, Arend M Van Der Zande, Jeevak M Parpia, Harold G Craighead, and Paul L McEuen. Impermeable atomic membranes from graphene sheets. *Nano letters*, 8(8):2458–2462, 2008. ISSN 1530-6984.

[7] E. Cadelano, P. L. Palla, S. Giordano, and L. Colombo. Nonlinear elasticity of monolayer graphene. *Phys Rev Lett*, 102(23):235502, 2009. ISSN 0031-9007 (Print) 0031-9007 (Linking). doi: 10.1103/PhysRevLett.102.235502. URL https://www.ncbi.nlm.nih.gov/pubmed/19658947.

[8] A. H. Castro Neto, F. Guinea, N. M. R. Peres, K. S. Novoselov, and A. K. Geim. The electronic properties of graphene. *Reviews of Modern Physics*, 81(1):109–162, 2009. ISSN 0034-6861 1539-0756. doi: 10.1103/RevModPhys.81.109.

[9] Alexander Croy, Daniel Midtvedt, Andreas Isacsson, and Jari M. Kinaret. Nonlinear damping in graphene resonators. *Physical Review B*, 86(23), 2012. ISSN 1098-0121 1550-235X. doi: 10.1103/PhysRevB.86.235435.

[10] D. Davidovikj, J. J. Slim, S. J. Cartamil-Bueno, H. S. van der Zant, P. G. Steeneken, and W. J. Venstra. Visualizing the motion of graphene nanodrums. *Nano Lett*, 16(4):2768–73, 2016. ISSN 1530-6992 (Electronic) 1530-6984 (Linking). doi: 10.1021/acs.nanolett.6b00477. URL https://www.ncbi.nlm.nih.gov/pubmed/26954525.

[11] Dejan Davidovikj, Farbod Alijani, Santiago J Cartamil-Bueno, Herre SJ van der Zant, Marco Amabili, and Peter G Steeneken. Young's modulus of 2d materials extracted from their nonlinear dynamic response. *arXiv preprint arXiv:1704.05433*, 2017.

[12] R. J. Dolleman, D. Davidovikj, S. J. Cartamil-Bueno, H. S. van der Zant, and P. G. Steeneken. Graphene squeeze-film pressure sensors. *Nano Lett*, 16(1):568–71, 2016. ISSN 1530-6992 (Electronic) 1530-6984 (Linking). doi: 10.1021/acs.nanolett.5b04251. URL https://www.ncbi.nlm.nih.gov/pubmed/26695136.

[13] A. M. Eriksson, D. Midtvedt, A. Croy, and A. Isacsson. Frequency tuning, nonlinearities and mode coupling in circular mechanical graphene resonators. *Nanotechnology*, 24(39):395702, 2013. ISSN 1361-6528 (Electronic) 0957-4484 (Linking). doi: 10.1088/0957-4484/24/39/395702. URL https://www.ncbi.nlm.nih.gov/pubmed/24008430.

[14] Changgu Lee, Xiaoding Wei, Jeffrey W Kysar, and James Hone. Measurement of the elastic properties and intrinsic strength of monolayer graphene. *science*, 321(5887):385–388, 2008. ISSN 0036-8075.

[15] N. Lindahl, D. Midtvedt, J. Svensson, O. A. Nerushev, N. Lindvall, A. Isacsson, and E. E. Campbell. Determination of the bending rigidity of graphene via electrostatic actuation of buckled membranes. *Nano Lett*, 12(7):3526–31, 2012. ISSN 1530-6992 (Electronic) 1530-6984 (Linking). doi: 10.1021/nl301080v. URL https://www.ncbi.nlm.nih.gov/pubmed/22708530.

[16] B. Liu, Y. Huang, H. Jiang, S. Qu, and K. C. Hwang. The atomic-scale finite element method. *Computer Methods in Applied Mechanics and Engineering*, 193(17-20):1849–1864, 2004. ISSN 00457825. doi: 10.1016/j.cma.2003.12.037.

[17] Fang Liu, Pingbing Ming, and Ju Li. Ab initiocalculation of ideal strength and phonon instability of graphene under tension. *Physical Review B*, 76(6), 2007. ISSN 1098-0121 1550-235X. doi: 10.1103/PhysRevB.76.064120.

[18] P. Liu and Y. W. Zhang. Temperature-dependent bending rigidity of graphene. *Applied Physics Letters*, 94(23):231912, 2009. ISSN 0003-6951 1077-3118. doi: 10.1063/1.3155197.

[19] JH Los, A Fasolino, and MI Katsnelson. Mechanics of thermally fluctuating membranes. *npj 2D Materials and Applications*, 1(1):9, 2017. ISSN 2397-7132.

[20] C. A. Marianetti and H. G. Yevick. Failure mechanisms of graphene under tension. *Phys Rev Lett*, 105 (24):245502, 2010. ISSN 1079-7114 (Electronic) 0031-9007 (Linking). doi: 10.1103/PhysRevLett.105.245502. URL https://www.ncbi.nlm.nih.gov/pubmed/21231533.

[21] D. Midtvedt, A. Croy, A. Isacsson, Z. Qi, and H. S. Park. Fermi-pasta-ulam physics with nanomechanical graphene resonators: intrinsic relaxation and thermalization from flexural mode coupling. *Phys Rev Lett*, 112(14):145503, 2014. ISSN 1079-7114 (Electronic) 0031-9007 (Linking). doi: 10.1103/PhysRevLett.112.145503. URL https://www.ncbi.nlm.nih.gov/pubmed/24765986.

[22] R. Nicklow, N. Wakabayashi, and H. G. Smith. Lattice dynamics of pyrolytic graphite. *Physical Review B*, 5(12):4951–4962, 1972. URL https://link.aps.org/doi/10.1103/PhysRevB.5.4951.

[23] Kostya S Novoselov, Andre K Geim, Sergei V Morozov, D Jiang, Y Zhang, Sergey V Dubonos, Irina V Grigorieva, and Alexandr A Firsov. Electric field effect in atomically thin carbon films. *science*, 306(5696): 666–669, 2004. ISSN 0036-8075.

[24] Hashem Rafii-Tabar. *Computational physics of carbon nanotubes*. Cambridge University Press, 2008. ISBN 0521853001.

[25] Singiresu S Rao. *Vibration of continuous systems*. John Wiley & Sons, 2007. ISBN 0471771716.

[26] Rafael Roldán, Annalisa Fasolino, Kostyantyn V. Zakharchenko, and Mikhail I. Katsnelson. Suppression of anharmonicities in crystalline membranes by external strain. *Physical Review B*, 83(17), 2011. ISSN 1098-0121 1550-235X. doi: 10.1103/PhysRevB.83.174104.

[27] Daniel Sánchez-Portal, Emilio Artacho, José M. Soler, Angel Rubio, and Pablo Ordejón. Ab initio. *Physical Review B*, 59(19):12678–12688, 1999. URL https://link.aps.org/doi/10.1103/PhysRevB.59.12678.

[28] E. B. Tadmor, M. Ortiz, and R. Phillips. Quasicontinuum analysis of defects in solids. *Philosophical Magazine A*, 73(6):1529–1563, 1996. ISSN 0141-8610 1460-6992. doi: 10.1080/01418619608243000.

[29] Levente Tapasztó, Traian Dumitrica, Sung J Kim, Péter Nemes-Incze, Chanyong Hwang, and László P Biró. Breakdown of continuum mechanics for nanometer-wavelength rippling of graphene. *arXiv preprint arXiv:1210.6812*, 2012.

[30] J. Tersoff. Empirical interatomic potential for carbon, with application to amorphous carbon. *Phys Rev Lett*, 61(25):2879–2882, 1988. ISSN 1079-7114 (Electronic) 0031-9007 (Linking). doi: 10.1103/PhysRevLett.61.2879. URL https://www.ncbi.nlm.nih.gov/pubmed/10039251.

[31] J. Tersoff. Energies of fullerenes. *Physical Review B*, 46(23):15546–15549, 1992. ISSN 0163-1829 1095-3795. doi: 10.1103/PhysRevB.46.15546.

[32] Y. Wei, B. Wang, J. Wu, R. Yang, and M. L. Dunn. Bending rigidity and gaussian bending stiffness of single-layered graphene. *Nano Lett*, 13(1):26–30, 2013. ISSN 1530-6992 (Electronic) 1530-6984 (Linking). doi: 10.1021/nl303168w. URL https://www.ncbi.nlm.nih.gov/pubmed/23214980.

[33] Xiaojing Xu and Kin Liao. Molecular and continuum mechanics modeling of graphene deformation. *Materials Physics and Mechanics*, 4:148–151, 2001. ISSN 1605-2730.

A

Lammps Manual

In this manual, a short explanation will be given, in order to allow future generations to redo the work done. In general it will be split in two parts, first the programming using Lammps will be discussed followed by the post-processing in Matlab, needed to obtain the results.

A.1. Running Lammps

An introduction in to running Lammps will be given here. For a more general explanation on the method, the reader is referred to chapter 3.

A.1.1. Introduction

This section is intended to give new Lammps users a quick introduction to the program. In general, the Lammps manual, at http://lammps.sandia.gov/doc/Manual.html , should answer most questions, though it takes some time to get used to how this manual is set up. I have been free in quoting from the official manual. The general setup of this section will follow the setup of a Lammps program. Some examples will be presented, in general the file used at this moment is used as an example. All commands are made bold, and all input for commands italic.

In general, in order to run Lammps, you need a Lammps program file, which will be discuss here. You would run it using command prompt or run it on the server. The general output can be written in a log.lammps file, here also errors etcetera will show up. Then, the desired output can be saved to text files or be extracted from the log file. All files involved will be discussed here, with a focus on the command file.

There will be three main directions of research being:

1. Studying a ring-down setup, excited by an initial velocity.
2. Study on the Brownian motion, which is not excited besides with the thermal energy.
3. Study on a static Stress-Strain test.

With which one should be able to reproduce all the results reported here.

One of the most prominent problems in running MD simulations is how to handle the big amount of data. A system will easily contain 1×10^4 atoms with 3 degrees of freedom over 1×10^6 time steps, leading to approximately 1×10^{11} data points, only considering position. This means one can not always investigate everything as easily, simply because the computational power needed to post-process and save the data is too high.

A.1.2. Download and run

Lammps can be simply downloaded from the website. How to install is different per operating systems, but it is all clearly explained on the website. In order to run, simply go to the right folder in the command prompt and type lmp_serial < <filename>. It depends on how and where you want to run your simulation, it may be a little bit annoying to get it to run, just check the manual and Google. It's always different, depending on the operating system etcetera.

A.1.3. Files

Lammps can be simply downloaded from the website. In general there are about 6 files involved in running Lammps:

1. **Lammps command file**, a file where you write your program. My file will be explained here, as an example.
2. **Server submission file**, short file to submit the program to the cluster.
3. **Input files**, files defining atom coordinates an possibly other variables, such as initial velocity. Pay attention to the layout!
4. **Log file**, output file, written by Lammps. Showing general progress and "intensive" results, such as temperature.
5. **.stdout file**, more extended output file, also including time steps and sometimes more errors.
6. **Output files**, files you ask Lammps to write for you, containing "extensive" results, per atom values, such as position or velocity. For example containing atom coordinates over time, etc.

All of these files will be discussed in the following sections.

A.1.4. Lammps command file

The command file is basically your program. It is an input file, containing all information and commands. What it contains will be discussed here, in order of appearance.

Initialization The initialization would generally look like this:

```
################## SET-UP ##################

units           metal
dimension       3
boundary        s s s                   # s= shrink wrapped, alternative is f, p is periodic which seems to be needed for NPT calc
newton          on

atom_style      atomic
atom_modify     map array
neighbor        2 nsq                   # to deside when to 'cut' a bond
neigh_modify    delay 10                # delay in steps before to cut a bond
```

Figure A.1: Initialization part, containing set up information

The # denotes a comment, anything written after on the same line is not read by Lammps.

The first thing to specify is which **units** you want to use. I choose $metal$, which means the units are:
mass = grams/mole
distance = Angstroms
time = picoseconds
energy = eV
etcetera. Another option would be to use SI units, the command would then be **units** *si*
For all possible options, or an overview of the used units, see the manual. In general, I would say using $metal$ is convenient.

The command **dimension** simply states the dimension of the studied system, in this case 3 (xyz). The other option would be 2, for purely 2d systems.

Next, **boundary**. This determines how the boundaries of the simulation box are defined. It basically states what would happen if an atom reaches the edge of the simulation box. In this case, I used s, which is non-periodic shrink wrapped. This means the boundary does not allow any atoms to pass, but instead increases with them, as to still include all atoms. This option is chosen for all three directions(xyz). Another option would be to use a periodic boundary condition (p), which means that if atoms move across the boundary on the right side, they will come in with the same velocity on the left side. To fix the boundary and simply delete atoms which cross, choose f.

The **newton** command turns on or off newtons 3rd law. The only influence is possibly a slight decrease in simulation time, as the force in case of a pairwise interaction, the force is only computed once. This does come at the cost extra communication, so weather it really saves time depends. It should give the same results in the end.

Atom_ style command determines which data as saved and calculated for each atom and interaction. The choice of style affects what quantities are stored by each atom, what quantities are communicated between processors to enable forces to be computed, and what quantities are listed in the input files, provided later. One could use a very general option, such as *full* but this would slow down the simulation. For all options, see the manual. *Atomic* should be sufficient in general.

atom_ modify is an extra option to the **atom_ style** command. In general, a _modify command is an extension to the more general command, and can be used at different parts of the simulation. In this case it is used to define how Lammps should assign ID's to atoms and create a lookup table for these ID's. It would usually not be needed, and probably increases simulation time, but it is useful or needed when assigning attributes to atoms in a later stage of the simulation. In this case for example, I assign a velocity to each atom separately, after the relaxation and thermalization of the system. Therefore, I need to specify.

Neighbour defines when to cut a bond. The length specified is the length after the cutoff-distance as specified in the used potential. This thus means that the neighbor list will be longer, as there are more bonds than strictly necessary when just looking at the potential. The advantage is however, that if the two atoms come closer again, the bond is still in the list, which speeds up the computational time.

Neigh_modify modifies the **Neighbor** command by applying a time delay as well as the already defined spatial delay, before removing bond from the neighbor list.

Initial conditions Next, the initial conditions can be applied, some variables can be defined and groups can be defined.

```
###---------------- DEFINING VARIABLES

variable        T0              equal   300                     #K
variable        dt              equal   0.001                   #picosec time increment 0.002
variable        d_T             equal   (20*${dt})              #sec time delay for thermostat temp delay time

variable        t0              equal   5000000                 # start simulation (timesteps)
variable        trun            equal   ${t0}                   # runtime simulation
variable        ttherm          equal   (${t0}/10)              # Time for thermalization
variable        T               equal   (${t0}/10000)           # period of measuring
variable        Ttherm          equal   (${t0}/10000)           # Period of thermo output

###---------------- LOAD ATOMS FROM FILE

read_data       Data_mem/data_r_100_bc_3_modes.mem                              # read atom position and mass

# Groups for measuring eigenfrequencies
group           mode1_4_group           type 2                  # Group to measure 1,4 mode
group           mode2_3_5_group         type 3                  # Group to measure 2,3,5 mode
group           mode6_group             type 4                  # Group to measure 6 mode

group           measure_group           union   mode1_4_group mode2_3_5_group mode6_group

# Groups for BC
group           bc_group                type 5                  # Group to attacht bc
group           non_bc_group            subtract        all     bc_group

region          tophalf_region          block           INF     INF     0       INF     INF     INF
region          bottomhalf_region       block           INF     INF     INF     0       INF     INF

group           tophalf_group           region  tophalf_region
group           bottomhalf_group        region  bottomhalf_region

group           bc_top_group            intersect tophalf_group         bc_group        # Tophalf of the BC
group           bc_bottom_group         intersect bottomhalf_group      bc_group        # Bottomhalf of the BC
```

Figure A.2: Initial conditions and variables

Variable, a variable can be defined as a constant value, or a function of constant values or other variables. When calling upon a variable, the syntax is ${variable}. A Variable can also be read from a file, in that case there are multiple options on how to this input file is set-up and read, for this, see the official manual.

textbfRead_data is the command used to read data needed to run a simulation. In this case the position of all atoms is read from a file. The layout for this file is rather strict, see the manual. The header of mine is shown in Figure A.3:

```
# position data to read, made in matlab

 13026  atoms
 05     atom types

 -105.00 105.00 xlo xhi
 -105.00 105.00 ylo yhi
 -3.00  3.00   zlo zhi

 Masses

 1 12.011000  # C
 2 12.011000  # C
 3 12.011000  # C
 4 12.011000  # C
 5 12.011000  # C

 Atoms

     1       2   0.00000000   1.42000000   2.50000000
     2       2  -1.22975607   0.71000000   2.50000000
     3       2  -1.22975607  -0.71000000   2.50000000
     4       2   0.00000000  -1.42000000   2.50000000
```

Figure A.3: Header of atom file

The total number of atoms needs to be specified, as well the the number of atom types. In my case I defined 5 atom types, which are all carbon. This is simply an easier way to import groups. The columns under atoms are; ID, type, x, y, z position. The position of all atoms can be made for example with Matlab, which can also write a file like this.

Group defines a group of atoms. Groups can be overlapping. A group can be defined on base of a geometric region, atom type, or by ID. I would say its often easiest to define a type in the same file where you specify the atom positions and than use this type to create a group in Lammps.

The log file shows how many atoms are in every group and more of such information. For my case, see Figure A.4.

```
###---------------- LOAD ATOMS FROM FILE

read_data       Data_mem/data_r_100_bc_3_modes.mem                          # read atom position and mass
  orthogonal box = (-105 -105 -3) to (105 105 3)
  2 by 4 by 1 MPI processor grid
  reading atoms ...
  13026 atoms

variable        vz0             atomfile                Data_mem/velocity_mode_6_v1_r_100.mem           # 1 A/ps max in 1st mode shape

group           mode1_4_group           type 2                  # Group to measure 1st mode
504 atoms in group mode1_4_group
group           mode2_3_5_group         type 3                  # Group to measure 1st mode
516 atoms in group mode2_3_5_group
group           mode6_group             type 4                  # Group to measure 1st mode
516 atoms in group mode6_group

group           measure_group           union   mode1_4_group mode2_3_5_group mode6_group
1536 atoms in group measure_group

group           bc_group                type 5                  # Group to attacht bc
1362 atoms in group bc_group
group           non_bc_group            subtract        all     bc_group
11664 atoms in group non_bc_group
```

Figure A.4: Log file showing groups and loaded atoms

The initial conditions, variables and groups will be the same regardless of the topic of research. The one main difference is that in case of **Brownian motion** the time for thermalization can be much longer than in case of **Ring-down**. In fact, if the thermalization time is too long in case of a Ring-down, the Brownian motion will be so far developed that the amplitude of Brownian motion will be rather high, and all modes will be present in the vibration response. In case of a **Static** test, there is no thermalization at all.

Potential A potential is easily applied. The choice of the right potential may be a little more complicated however. Some potentials come with installing Lammps, these can be found in the *potentials* folder in the installed Lammps folder. Many other potentials can be downloaded from the Internet.

```
###---------------- POTENTIALS TO BE APPLIED

pair_style      tersoff                                 # Potential for graphene
pair_coeff       * * ./BNC.tersoff C C C C C            # Number of C dependent on number of atom types
```

Figure A.5: Applying an atomic potential energy function

Pair_style defines the style of the potential. It defines the general formula used to calculate the forces in all bonds. The specific coefficients of the used formula are defined in the **pair_coeff** command. Note that here you have to add the number of types defined in your simulation. In my case I have 5 types, which are all Carbon. Therefore, I added $C\ C\ C\ C\ C$ after the command. If one would have only 2 types, Nitrogen and Carbon, it would be $N\ C$ instead.

Relaxation After importing the atom positions, the system needs to be relaxed, as to make sure it is in its minimal potential state. Otherwise it would start vibrating later around this state. Finding the positions of all atoms is done by optimizing towards a minimal force and lowest energy The relaxation can be done as:

```
################## Relaxation of the system ##################

fix             fix_all         all             setforce        NULL    NULL    0.0

#----------- Dump
#dump           relax all xyz 100 res/pos/relax_bc_3_N_1_r_100.txt                          # all pos appended
#dump_modify    relax append yes

thermo          100
thermo_style    custom step temp etotal pe ke

minimize        1.0e-10 1.0e-10 10000 10000

unfix           fix_all
#undump         relax
```

Figure A.6: Relaxation of the system

Fix fixes *all* atoms to move in y direction. This is done by setting the force in y to 0, just before solving Newton's law. Therefore, there will be no acceleration and thus no movement in y. This is done here because I want to study a flat membrane. If one would not restrict anything, one may just end up with a carbon-nanotube.

Dump the command is actually commented here, because I m not interested in this data right now. **Dump** is one of the ways to save data from Lammps. It can be used to take snapshot pictures or movies, though I would recommend doing this sort of post-processing yourself or with something like Ovito, which can be downloaded for free. You can also save text files with information. In this case, I called the dump *relax*, which is just a name. The *zyx* position of *All* atoms is saved every *100* time steps, in a text file in a folder.

dump_modify modifies the dump command with also specifying that the file should be appended. This means that every time step is written under the previous one, in the same file. This is in my opinion much nicer than creating a separate file for each time step.

Thermo command is used before every run you make. It defines how often you want overall properties, such as temperature, to be saved. In this case every *100* time steps.

Thermo_style states which information you want to be saved. In this case I also want to see the step, temperature and kinetic an potential energy.

Minimize actually does the minimization of kinetic energy and forces. The first two numbers are related to the stopping tolerance for energy and force respectively, the second two are related to the maximum number of iterations. I just chose a very small and high number. In my experience, it does not really matter, the Relaxation is rather fast, and after say 200 time steps an equilibrium is usually found.

unfix undoes the fix done above. I like to unfix after each step, even if you may need the same fix in the next part. It just keeps everything structured.

Undump undoes the dump command. Always good to remember, otherwise you end up with a huge text file, easily up to some gigabytes.

The output, which is given in Lammps log file for this relaxation is:

```
WARNING: Resetting reneighboring criteria during minimization (../min.cpp:168)
Neighbor list info ...
  1 neighbor list requests
  update every 1 steps, delay 0 steps, check yes
  max neighbors/atom: 2000, page size: 100000
  master list distance cutoff = 4.1
  ghost atom cutoff = 4.1
Memory usage per processor = 0.492744 Mbytes
Step Temp TotEng PotEng KinEng
       0            0   -103336.6   -103336.6            0
     100            0   -103702.87  -103702.87           0
     200            0   -103703.12  -103703.12           0
     211            0   -103703.12  -103703.12           0
Loop time of 1.00852 on 8 procs for 211 steps with 13026 atoms

100.0% CPU use with 8 MPI tasks x no OpenMP threads

Minimization stats:
  Stopping criterion = energy tolerance
  Energy initial, next-to-last, final =
        -103336.596882     -103703.118259     -103703.118269
  Force two-norm initial, final = 152.957 0.0314433
  Force max component initial, final = 8.15894 0.0029581
  Final line search alpha, max atom move = 1 0.0029581
  Iterations, force evaluations = 211 421
```

Figure A.7: Part of the log file showing the relaxation

The minimal state is found after 211 steps. The output is written every 100 steps, as defined in the **thermo** and **thermo_style** commands. The objective values for force and kinetic energy are also given.

It may be convenient to visualize this data with for example Matlab. As an example the relaxation of a 10 nm is shown in Figure A.8. The potential energy is decreasing from 1.0335 to 1.037 eV. This is a decrease of about 0.3%, which may seem small, but can be quite significant in an undamped system.

In general however, you are not really interested in how and how fast it converges, this is only a very small part of the computational effort. Therefore, its best just to set convergence criteria which are sure to be low enough.

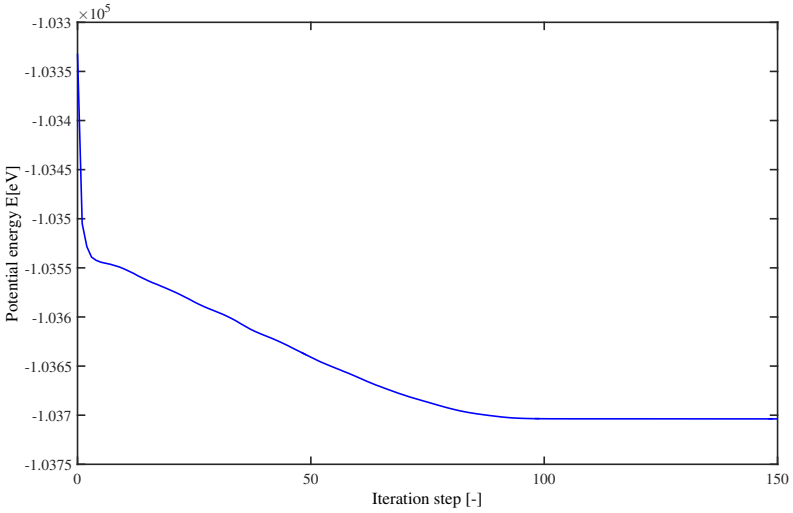


Figure A.8: Relaxation of the system; Potential energy as a function of time steps(iterations)

Thermalization Next step is to thermalize the system. All atoms must be slowly accelerated in order to create an overall velocity distribution which corresponds to the Maxwell-Boltzmann distribution; given by:

$$\rho(v_1) = \sqrt{\frac{m}{2\pi k_B T}} \exp\left(-\frac{m v_1^2}{2 k_B T}\right). \tag{A.1}$$

In order to reach this equilibrium all atoms are slowly accelerated and the velocities are rescaled after every time step. This rescaling is done according to:

$$\frac{1}{2}\sum m_i v_i^2 = \frac{3}{2} K_B T. \tag{A.2}$$

Lammps does all the complicated things for you. The only important thing really is to fix the right atoms and to set the right environment. This would be NVT, which means the number of atoms, volume and temperature are constant. This means there can be an exchange of temperature with the environment, which allows for the rescaling of velocities. In my case see Figure A.9.

```
################## THERMODYNAMICS ##################

###---------------- equilibrium by NVT

#----------- computations
compute         newtemp all temp                                # calculate temp at each timestep
fix             fix_bc_top              bc_top_group            setforce        0.0     0.0     0.0
fix             fix_bc_bottom           bc_bottom_group         setforce        0.0     0.0     0.0
fix             fix_nvt                 all                     nvt temp        ${T0}   ${T0}   ${d_T}

#----------- Dump
thermo_style    custom step temp etotal press ke
thermo_modify   norm no

#----------- run

timestep        ${dt}
thermo          ${Ttherm}

reset_timestep  0
run             ${ttherm}

unfix           fix_bc_top
unfix           fix_bc_bottom
unfix           fix_nvt
```

Figure A.9: Thermalization

Compute Define a computation that will be performed on a group of atoms. Quantities calculated by a compute are instantaneous values, meaning they are calculated from information about atoms on the current time step or iteration. Defining a compute does not perform a computation. Instead computes are invoked by other Lammps commands as needed, for example in this thermalization. So, one needs to ask Lammps specifically to compute temperature as this is needed in this case.

Fix, again fix some atoms of the simulation. In this case, first I fix the boundary. If the boundary would not be fixed in the thermalization, but it would be later in a movement analysis, you would force the system

to cool down. Now, the fixed atoms have 0 temperature, as they do not move. This is also the state they will have in later simulations. Next the NVT is fixed. This is the environment discussed before. I set the desired temperature to be equal to the defined variable $T0$. The last parameter, d_T is a value for the damping in the system. The system will have have the exact right temperature, and to avoid a huge overshoot, the control needs to be damped.

Timestep defines the time step, which is in this case already defined in the variable. The choice you make here can have a great influence on computational time and convergence of the simulation. If you do not specify a value yourself, Lammps will just take some default value.

Reset_timestep simply resets the time step counter. Just for convenience, I reseted to 0 here.

Run defines for how many time steps the simulation will run. Note that the simulated time is thus dependent on the time step size. How long one wants to run the thermalization depends, its smart to simply check convergence.

In this case, the log file showed:

```
################## THERMODYNAMICS ##################

###---------------- equilibrium by NVT

#----------- computations
compute         newtemp all temp                                # calculate temp at each timestep
fix             fix_bc_top              bc_top_group            setforce        0.0     0.0     0.0
fix             fix_bc_bottom           bc_bottom_group         setforce        0.0     0.0     0.0
fix             fix_nvt                 all                     nvt temp        ${T0}   ${T0}   ${d_T}

#----------- Dump
thermo_style    custom step temp etotal press ke
thermo_modify   norm no

#----------- run

timestep        ${dt}
thermo          ${ttherm}

reset_timestep  0
run             ${ttherm}

unfix           fix_bc_top
unfix           fix_bc_bottom
unfix           fix_nvt
```

Figure A.10: Log file showing Thermalization

Again, one can plot this data in Matlab, to have a better view on how the temperature develops over time. An example is given in Figure A.11 .

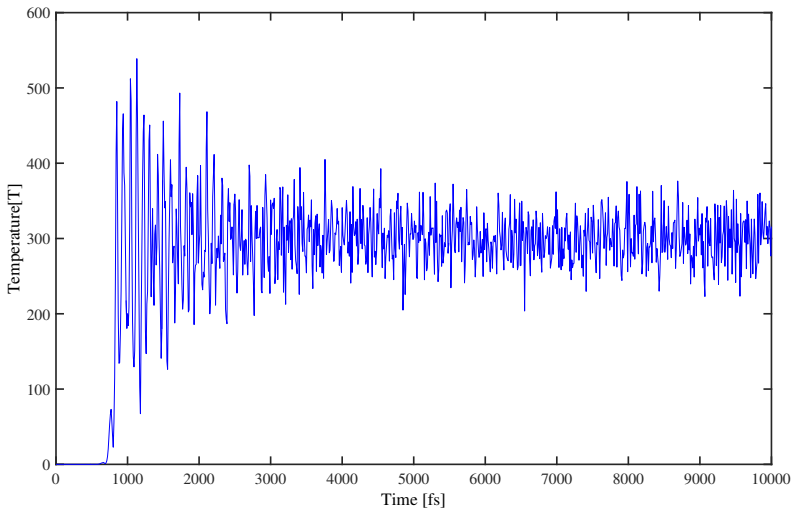


Figure A.11: Thermalization of the system. Temperature versus time.

In the start the temperature is rising only very slowly, until it suddenly increases quickly. Next a region of high overshoot follows. Finally, the temperature converges more or less. It will never fully converge, as it is a statistical thing, and with a limited number of atoms it is hard to keep a constant value. Furthermore, as the system is vibrating, there is always a coupling between thermal and kinetic energy, which will always make for some fluctuations. It may be interesting to compare different values for the thermal damping value d_T and see which gives the best results.

The temperature was probably quite fine after 20000 time steps already, but it does not take that much computational time, so it is not worth the effort of always checking instead of just let it go some longer instead.

Note that in case of a **Static** test, no Thermalization should be done. In case of a **Ring-down**, the thermalization time should be limited, in order to avoid a fully developed Brownian motion.

Movement Now the system can be actuated and left to vibrate. Now, we want to isolate the system from the environment, basically not allow any energy or mass transfer. Therefore we create a NVE environment, not allowing atoms, volume or energy to change. The simulation is run for some time steps and the results are saved. My file is shown in Figure A.12.

```
################## MOVEMENT ##################

#----------- fixes
fix             fix_bc          bc_group                setforce        0.0     0.0     0.0
fix             fix_nve         non_bc_group            nve

#----------- set velocity
velocity        all     set     NULL NULL v_vz0         sum     yes             # set initial speed

#----------- Save results
dump            positionsdump non_bc_group xyz 500      res/pos/pos_T0_300_bc_3_N_500_100000_mode_6_v0_1_appended_all_0.txt                     # all pos ap
dump_modify     positionsdump append yes
dump            measure measure_group xyz 1000          res/pos/pos_T0_300_bc_3_N_1000_5000000_mode_6_v0_1_appended_0.txt                       # all pos ap
dump_modify     measure append yes

#----------- run
thermo          10000
timestep        ${dt}

reset_timestep  0
run             100000

undump          positionsdump

timestep        ${dt}
thermo          10000
run             4900000

undump          positionsdump
```

Figure A.12: Program file of the movement

Again the boundary is fixed in all directions. Next the NVE fix is done, on all atoms except the boundary. This is because the boundary does not have any energy anyway, and including it in the fix causes problems.

Velocity sets a velocity to a set of atoms. In this case, *all* atoms gain a velocity in z direction. The velocity is saved in the variable $v_v z0$ which was read from a file earlier. The *sum* option signifies that the velocity is summed with the already present velocities, coming from the Brownian motion.

Next the positions of some groups of atoms is saved. Some information is used to make a video, and some for the actual post-processing of the data. Therefore, I want some things to be saved more often or for a longer timespan than others. To achieve this, simply run the simulation for a while, delete a dump with the **undump** command, and run for some time longer. This is also a good way to cut up the simulation in parts. One could save results after 0, 5 and 10 ns, and take only 0,1 ns for post-processing. In general, you have to be smart in which information you need and want to save. Lammps can easily provide you with more data than you could save anywhere.

The log file is showed in Figure A.13.

```
Step Temp TotEng Press f_fix_bc_top[1] f_fix_bc_top[2] f_fix_bc_top[3]
       0            0   -84820.523     112.7749 -4.8810678e-12  0.016393036    3.9416067
    1000    315.80931   -84227.891   -31500.857 -1.8933477e-10    -41.25612   -21.287701
    2000    314.02094   -83995.849   -18925.122 -8.1591622e-11    -85.031105   -30.836824
    3000    300.54583    -83812.88   -17652.011 7.5154034e-09   -92.006105   -69.116723
    4000     289.0762   -84312.697   -14712.117 -9.1110763e-10    -29.540898   -20.071171
    5000    306.35974   -84255.319   -17711.967 3.0654803e-09   -59.331302   -31.515388
    6000    288.73939   -84081.619   -18432.957 1.2361486e-08   -37.346365    20.675111
    7000    316.76869    -83575.68   -21353.352 -5.0122075e-09    -40.061832  -0.28157362
    8000    289.42745   -84195.782   -14482.716 -6.2534014e-09    -94.167701    20.974961
    9000    295.09752   -84304.951   -10610.201 5.4733214e-09   -46.063743    6.0532457
   10000    297.09513   -84144.055   -17754.154 -1.5864146e-08    -47.26317   -19.589328
   11000    302.46939   -83635.378   -19628.156 1.3428604e-08   -153.94015   -105.21137
   12000    310.69651   -84118.011   -16099.509 7.9589304e-09   -10.705183   -4.0202111
   13000    299.43626   -84258.312   -21446.384 -1.9424107e-10    -56.918285    -14.88911
   14000    299.05914   -84235.138   -11524.455 -1.5715194e-08    -48.575528    3.0519695
   15000    307.73942    -83772.88   -20212.883 1.3526055e-08   -58.813804    32.464836
   16000    281.88418   -84078.051   -13464.341 -9.8261168e-09    -45.368226    29.613254
```

Figure A.13: Log file of the movement

In case of **Brownian motion** there will be no velocity given to the system, the movement resulting from thermal vibrations is all that is needed. In case of **Static** test, there will be no velocity imposed, but instead a displacement will be imposed on the boundary. This command would read as:

```
################## Strain and force measurement ##################

###---------------- Strain the system
fix             fix_strain      all     deform  1       x       erate   ${dr}   y       erate   ${dr}
fix             fix_bc_bottom           bc_bottom_group         setforce        0.0     0.0     0.0
fix             fix_bc_top_right        bc_top_right_group      setforce        0.0     0.0     0.0
fix             fix_bc_top_left         bc_top_left_group       setforce        0.0     0.0     0.0

thermo_style    custom step etotal press f_fix_bc_top_right[*] xhi
thermo_modify   norm no format 4 %20.15g format 5 %20.15g format 7 %20.15g

#----------- run

timestep        ${dt}
thermo          1

run             30000

unfix           fix_strain
unfix           fix_bc_bottom
unfix           fix_bc_top_right
unfix           fix_bc_top_left
```

Figure A.14: Part of the command file which takes care of the static strain

Here, the **fix** *deform* command applies the engineering strain to the system. Every so many steps, the system is strained by a certain percentage. Also note how the *format* of output is changed by the **thermo_modify** command, as to ensure a good precision of numbers.

A.1.5. Server submission file

When running on the server, you have to write a small script to tell the server how many nodes and processors you need. An example is given in Figure A.15.

```
#PBS -l nodes=1:ppn=8
cd $PBS_O_WORKDIR

echo Running on host `hostname`
echo Time is `date`
echo Directory is `pwd`
echo This jobs runs on the following processors:
echo `cat $PBS_NODEFILE`

if [ -e log.lammps ];
        then
                #The last dot is the separator
                LAST=`ls -td log.lammps.* | head -n 1 | awk -F. '{print $NF}'`
                if [ "$LAST" ==  "" ];
                        then
                                cp log.lammps log.lammps.0
                        else
                                NEXT=`echo "$LAST + 1" | bc -l`
                                cp log.lammps log.lammps.$NEXT
                        fi
                fi

##!IMPORTANT
module load intel/2013sp1
module load mpi/openmpi-1.8.8-intel
module load cuda/7.0
mpirun /home/svanhemert/bin/lammps17nov < in.graphene_mem_cluster_5_big > lammps_r100_mode_6_v1_T300.stdout
```

Figure A.15: Example of a submission file for running Lammps on the server

In fact, only the first and last lines are of importance. The first line specifies how many nodes and processors per node are needed. How many nodes and cores you need and can use is a matter of how much of the server you can use, how much time you can wait and mostly, the size of your system. I mostly used 8 cores on one node, for a system containing 13 000 atoms. The center part just specifies that if a standard output file, created by Lammps, already exist, it should be renamed, such that you can run different tests at the same time, and the output will be nicely saved in different .log files. The last part, under Important is showing which specific modules are most suitable for the programming of Lammps. Once chosen, you never have to change this. Than the last line is actually running the Command file, which will be discussed later. Mpirun is the run command, followed by the direction to the Lammps executable. < indicates which command file should be run and > indicates the path to the output file, created by the server. This output file contains information very similar to the .log file created by Lammps, but also contains extra information on errors and furthermore, is updated every time step rather than after each **run** command.

A.1.6. Used constants

The used constants are summarized in Table A.1. Note that true values are of course also dependent on membrane size, temperature, total simulation time and many more parameters. If one is interested in bigger membranes, it may be needed to set the step size to 2 fs in order to reduce computational time.

A.1.7. Handling data

How to save results to text files was already discussed in the previous section. However, I want to make some extra clarifications and remarks to this subject. First of all, always try to keep an eye on how much data

Table A.1: Used parameters for step size, minimization tolerances, temperature damping and thermalization time. Values may differ depending on size, temperature and total simulation time.

	Static	Brownian	Ring-Down
Step size	1 fs	1 fs	1 fs
Minimization stopping tolerance energy	1×10^{-10}	1×10^{-10}	1×10^{-10}
Minimization stopping tolerance force	1×10^{-10} eVÅ^{-1}	1×10^{-10} eVÅ^{-1}	1×10^{-10} eVÅ^{-1}
Temperature damping parameter	20 fs	20 fs	20 fs
Thermalization time	0 ps	500 ps	20 ps

you really need and are saving. The amount of data saved is easily reaching gigabytes, which makes it hard to handle the data. It will be impossible to simple open the text file, and moving it from the server to the desktop can take a while. If you don't need a set of data anymore, it is generally a good idea to compress the files, as they can easily be reduced to 30% of the original size by just zipping. Secondly, keep in mind that only per-atom data can be saved to text files using **dump**. For over-all values such as the force on the boundary, temperature or kinetic energy, it is easiest to write the output to the log file by the **thermo** command. Than, in order to allow for post processing, the data should be copied from the .log or .stdout file to a new text file. Furthermore, it makes the post processing much easier if you save the data with a sensible name, such that the name can tell you all you need to know about the content, and it is easy to run a loop over several sets of data using Matlab. Think of something like $atom\_positions < radius >< temperature >< measuringperiod >< totaltime >< etc >$.

A.2. Post processing using Matlab

All the post-processing is done using Matlab. The used code is provided digitally. Here, the general overview of the program will be given.

A.2.1. Reading data

First the data has to be read from a text file. It is a good idea to save the text files in such a way that the name can be build up from different variables, such as temperature. Reading data is best done using the **textscan** command. This is much faster than using **dlmread**, as it does not open the full file and save everything at once, but reads it line by line. Here you can read the x,y,z position of the measured atoms over time.

A.2.2. Thermalization, relaxation and over-all values

For checking the Thermalization, relaxation or static stress strain, a slightly different approach is taken. The data is now read from a slightly different text file, which is a hand-made copy of part of the .log file. The format of the data is depending on the **thermo_modify** settings, used in Lammps. In this case you might as well use **dlmread** because the size of the data is small anyway.

A.2.3. Time response

The time response is actually already obtained by simply loading the data from the file. You can choose to show the average out of plane displacement, which will not show any contribution from the asymmetric modes(2,3,5,6), as these have an equal contribution in positive and negative direction. Another option is to visualize the vibration of the membrane in three dimension. For that purpose, a grid of equally spaced points has to be created. Now the values for z can be interpolated such that they fit on the grid. This will result in a vector containing equally spaced points in x and y and a matrix containing the corresponding out of plane positions z.

The mode shapes can be obtained by taking snapshots of the time response at intervals equal to the period of vibration of the particular mode shape. After averaging over a sufficiently large number of snapshots, the mode shape can be obtained. Note that degenerate modes can rotate over time. This means every snapshot has to be rotated, as to overlap with the other snapshots. This can be done by finding the angular position of the highest peak, and rotate this such that it overlaps with one of the angular positions of the maximum in the reference mode shape.

A.2.4. Frequency response

The frequency response can be obtained by taking the Fast Fourier transform(fft) of the time response of every single atom. All these fft's can be summed and divided over the number, as to average. This will give the average frequency response. In fact, it will give the average amplitude per frequency. This information can also be used to calculate the energy in each mode, by combining the amplitude of vibration with the stiffness of that mode.

B

Molecular Dynamics results

Here, all valuable results obtained in Lammps will be discussed and shown. First, a 1 nm radius model was made, as a small scale test. From this model, some important conclusions were obtained, which were implemented and used on the bigger scale model. Most results were obtained for a 10 nm membrane. Finally, some results were obtained by comparing different radii, from 1 to 20 nm.

B.1. Membrane of 1 nm radius

First a small model was made as an initial try to get all things running, without spending to much time on computations. The eigenfrequencies were indeed found either by actuating in a certain mode or by looking at the Brownian response. This program showed that the concept was feasible. In this chapter first the geometry will be shortly discussed. Next the actuation method is explained. Finally some important conclusions and results are discussed.

Some fundamental problems were discovered. Finding these on a smaller scale made it possible to partly solve these on the bigger scale. The findings will be shortly summarized here, with an emphasis on the lessons learned for the bigger model.

B.1.1. Geometry

This model had a 1 nm radius, resulting in 234 atoms. The used grid is shown in Figure B.1. Here the crosses are fixed atoms at the boundary.

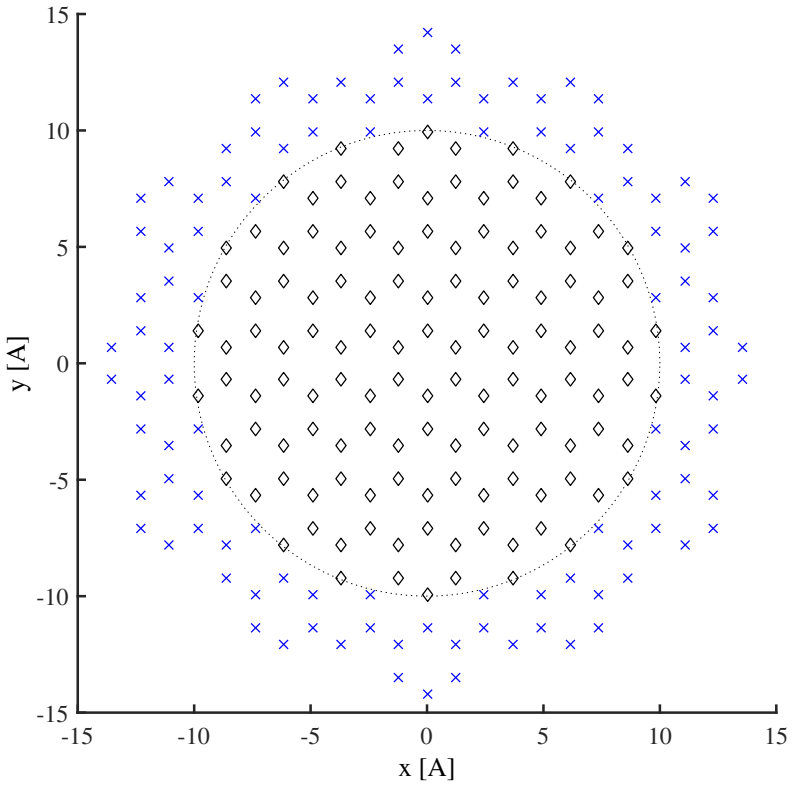


Figure B.1: Small model geometry, showing atom positions and boundary.

B.1.2. Excitation

One of the fundamental problems was how to actuate the system best. One option would be to give an initial displacement. The main problem here is that one needs to know the exact position of all displaced atoms, which would also move a little bit in the in plane direction. As the minimization done in Lammps would simply move the atoms back to the original plane when not restricted in z direction. If restricted in z direction, the boundary would move a little bit in, which would result in a buckled configuration. Therefore, giving an initial velocity was preferred. This is done by giving every atom a velocity in out of plane direction, with magnitude corresponding to the mode-shape. As an example, the velocity profile used for the second mode is shown in Figure B.2.

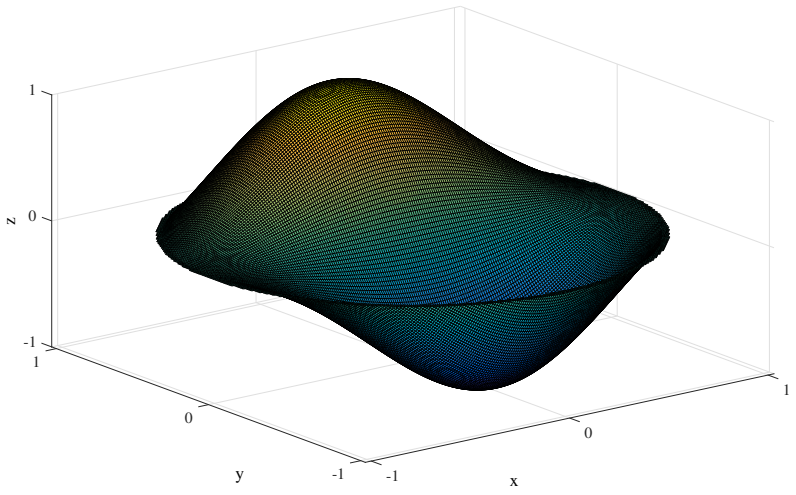


Figure B.2: Initial velocity profile used to excite the second mode

B.1.3. Convergence

Convergence was reached after 2.5 ns. The rate at which the eigenfrequencies converge to a stable value depends strongly on the amount of energy in the system. For higher temperatures or higher initial velocities, the time to convergence goes down. The system will always redistribute the available energy over all modes. At higher temperature this goes faster due to the increased thermal mode coupling. At higher initial velocities this is enhanced by nonlinear mode coupling. This also means that if one is interested in a single mode, waiting for a longer time only decreases the part of total energy present in that certain mode. Furthermore, several different states can be found which are present for a certain timespan. This can be seen through sharp changes in the time and frequency response. What these states exactly mean is not really clear yet. The best solution is probably to average over a longer timespan to be sure to include at least different states of which the biggest part should be the normal steady state. Or maybe, when running for a sufficiently long time will result in a converged steady state.

B.1.4. Influence of temperature

Temperature has only a small influence on eigenfrequency. This is unexpected in the sense that temperature actually gives the pre-tension in the membrane, which is expected to have a strong influence on the eigenfrequencies. Therefore, the eigenfrequencies are dominated by the non-linear hardening of the system rather than the linear stiffness resulting from in plane pre-tension. The frequency response for different temperatures is shown in Figure B.3. Indeed, the eigenfrequency does not change much with temperature. It can be seen that the amplitude of the super harmonic increases with increasing temperature. This is due to the extra energy in the system which increases the maximum amplitude and thereby the nonlinearity.

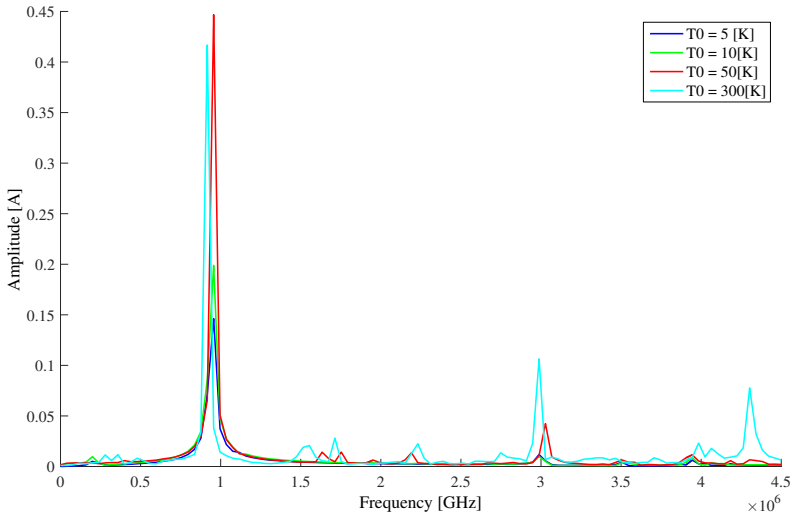


Figure B.3: Frequency response for different temperatures when exciting the first mode

Thermal noise does have a very strong influence on the time response. Only after measuring over a longer timespan and averaging over all atoms a clear time and frequency response can be obtained. As en example the time-response of one atom in the system is shown in Figure B.4a and Figure B.4b comparing between 5 and 300 K after 10 ns simulation time. Indeed, at 300 K the thermal noise is significant.

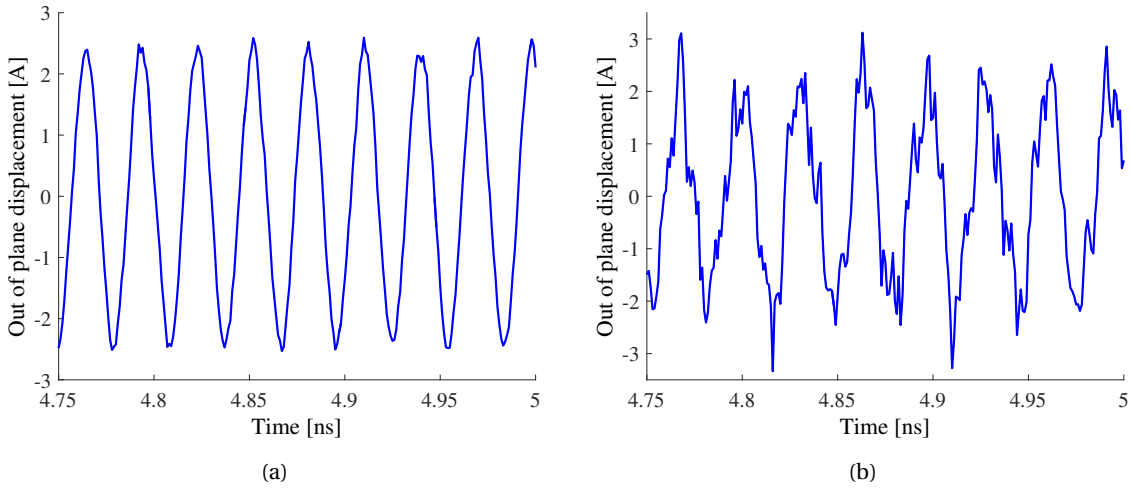


Figure B.4: Comparing the time response for (a)5 K and (b)300 K

B.1.5. Influence of initial velocity

A non-linear response is almost directly seen. Therefore, a low actuation velocity would be preferred. The problem is however, that if there is little energy in the actuated mode, it will be very quickly dispersed among all other modes. Besides that, the amplitude of thermal noise is quickly comparable to the amplitude of the actuated mode. In Figure B.5 the frequency response is shown for an initial velocity which is increased from 1 to 10 Å ps^{-1} at 5 K. Indeed the eigenfrequency is increasing with increasing amplitude. A super harmonic can also be seen. This means that one the bigger scale different actuation velocities should be compared, as to ensure the best signal to noise ratio. This will not be easy, as a higher initial velocity does increase the energy in one mode in comparison to all other modes, but as well increases non-linear mode coupling and can easily show very strong non-linear behavior. A very low initial velocity would quickly turn into a state where the energy in all modes is comparable, such as seen in Brownian motion.

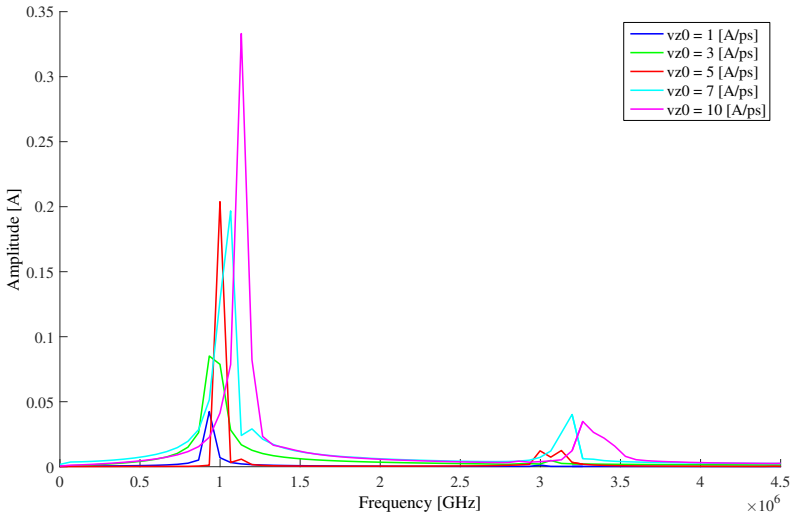


Figure B.5: Frequency response for different initial velocities, showing an increasing nonlinear behavior.

B.1.6. Study on Brownian motion

Furthermore it was shown that Brownian motion can give the frequencies of all modes. This response also converges very quickly. Using the Brownian motion would be preferable over exciting each mode separately, because you do not need to impose any starting condition, and the response is stable over time.

B.2. Membrane of 10 nm radius

After the small model was done, I moved to a bigger scale. The dimensions were set to 10 nm radius, with 3 (radial) layers of atoms added as the fixed boundary. This results in a geometry very similar to what was shown for the small model, now including 13026 atoms.

B.2.1. Convergence

The first criterion for convergence is that the amplitude of vibration does not change over time. As the system is not damped, the amplitude should reach a fixed value after some time. The amplitude of vibration for the first mode over time is shown in Figure B.6.

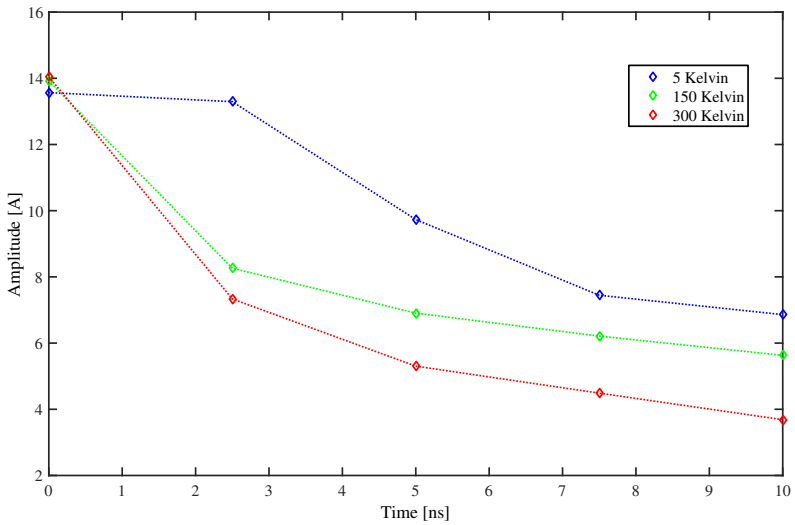


Figure B.6: Amplitude over time, comparing 5, 150 and 300 K. Ring-down, after giving an initial velocity profile of 5 Å ps^{-1}

The amplitude of vibration is clearly decreasing over time, and has not converged in the shown time. Furthermore, the temperature seems to have a strong influence on the speed with which the amplitude is decreasing. With a higher temperature the decrease seems to be much faster as compared to the lower temperature.

B.2.2. Mode Coupling

Here, the mode coupling is further investigated by looking at the change of the frequency content over time. Therefore, the logarithmic amplitude is shown at 3 parts of the total simulation time. The results are shown in Figure B.7.

As already seen in Figure B.6, the temperature indeed has an enormous influence on the convergence rate of the frequency response. Furthermore, the added stochastic excitation is clearly seen in the first part of the frequency response, where all modes are already present. It can be seen that the coupling from first to second mode is much smaller than from first to fourth mode.

In addition to the energy per mode, as shown in the main part, the total energy over time can also be calculated, by summation of the energy in the first six modes.

The total energy in the first six modes also decreases steadily over time. Part of this energy is fed into other modes, another part may be converted to thermal energy, and finally, part of this energy may be able to leave the NVE ensemble, through the small allowed fluctuations through the boundary. Note however that the amount of energy is low, which means the influence on temperature will be very small.

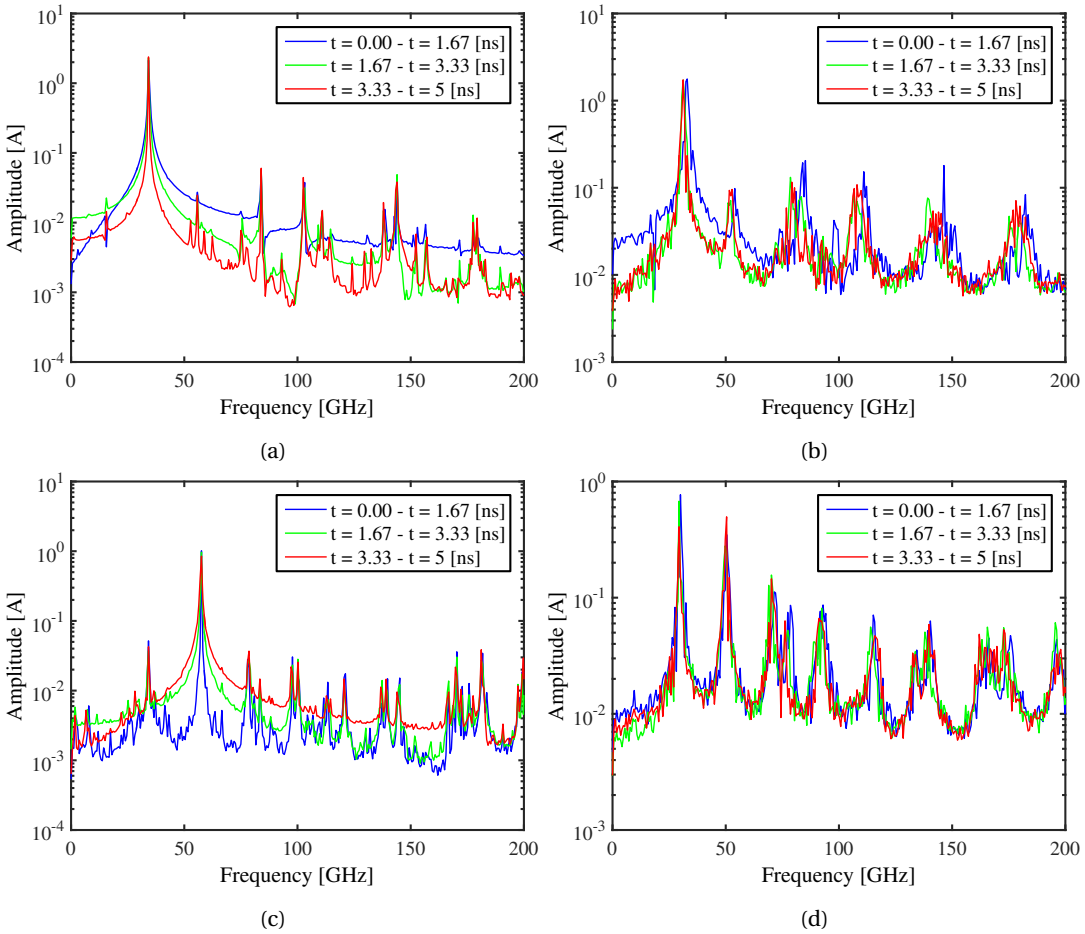


Figure B.7: Frequency response when actuation a mode with 0.5 Å ps^{-1}, comparing first, second and third part of a 5 ns simulation time, at 5 and 300 K. (a) First mode at 5 K , (b) First mode at 300 K , (c) Second mode at 5 K , (d) Second mode at 300 K.

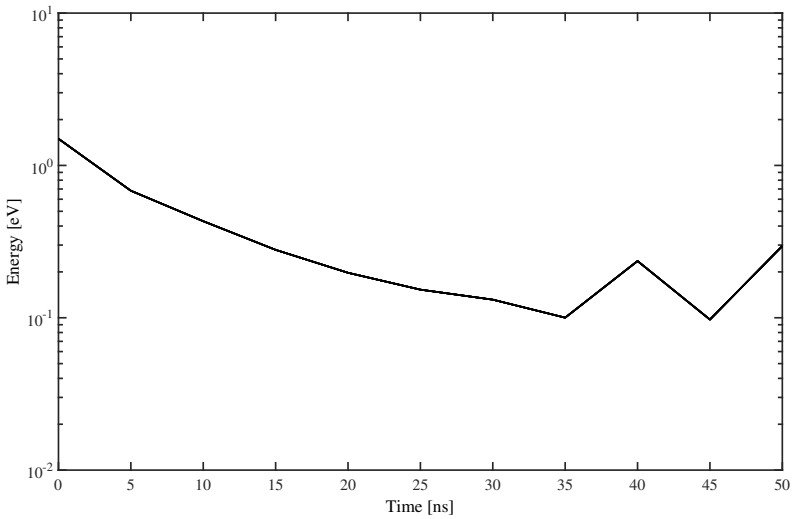


Figure B.8: Total energy in the first six modes over 50 ns

B.2.3. Exciting all modes simultaneously

As to excite all modes at once, the membrane was excited with a shape corresponding to the sum of normalized profiles. In this way all modes are excited equally strong at the same time. The initial velocity is thus

given as

$$vz_0 = \frac{\sum_{i=1}^{6} \boldsymbol{\phi}_i}{\max(\sum_{i=1}^{6} \boldsymbol{\phi}_i)} \tag{B.1}$$

where $\boldsymbol{\phi}_i$ is the normalized membrane mode i which results from the modified Bessel function. After summation the velocities are scaled to the preferred maximum value. This profile is shown in Figure B.9.

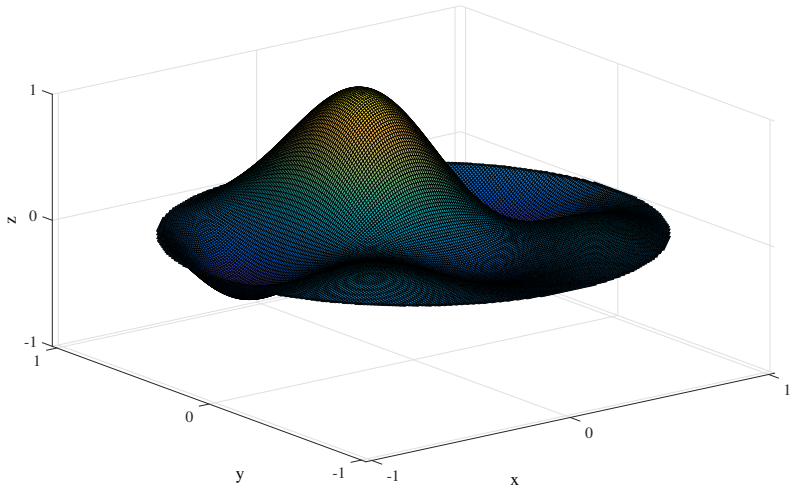
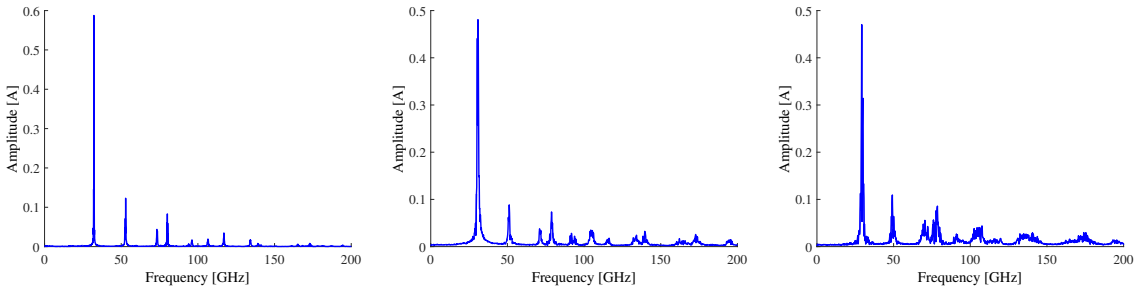


Figure B.9: Initial velocity profile which includes a weighted sum of all modes

Actuation with this initial velocities indeed showed all eigenfrequencies. The results are shown in Figure B.10a to Figure B.10c:



(a) (b) (c)

Figure B.10: Frequency response after actuating all modes simultaneously at (a) 5 K, (b) 150 K and (c) 300 K

The found eigenfrequencies are summarized in Table B.1. The eigenfrequencies decrease by 1-5% with increasing temperature. This is not directly as expected, as the increase in temperature actually increases the pre-strain in the membrane, which would increase the eigenfrequency. This decrease could result from exactly this increase in pre-strain, as this decreases the non-linearity of the membrane. As the membrane is very little tensioned, the vibration could show geometric hardening. This hardening is decreased with increasing pre-strain. Another explanation could be that the simulation is not fully converged at lower temperature. Therefore, the frequency is still higher due to the high initial velocity.

Table B.1: Eigenfrequencies as found by actuating all modes simultaneously

	5 K	150 K	300 K
1st eigenfrequency	32.19 GHz	30.79 GHz	29.19 GHz
2nd eigenfrequency	52.89 GHz	51.19 GHz	49.19 GHz
3rd eigenfrequency	73.29 GHz	70.99 GHz	70.39 GHz
4th eigenfrequency	80.08 GHz	78.78 GHz	78.58 GHz
5th eigenfrequency	96.18 GHz	91.78 GHz	90.98 GHz
6th eigenfrequency	106.6 GHz	104.4 GHz	102.6 GHz

B.2.4. Brownian motion

In the previous case the vibration had to be forced on the system by applying an initial velocity with a certain distribution. This may not be the natural way of vibration of the membrane. Therefore the system was excited randomly, by simply applying a temperature. A temperature on this scale is basically a random velocity in all 3 directions, according to the Maxwell-Boltzmann distribution, as discussed in subsection 3.1.3. The random velocity had to be applied in steps, as to be sure that the temperature would converge to the required value, according to $\frac{1}{2}\sum m_i v_i^2 = \frac{3}{2}K_B T$. This does indeed correspond to the Brownian motion. The found frequency response for 5 , 100 ,200 and 300 K is shown in Figure B.11a to Figure B.11d.

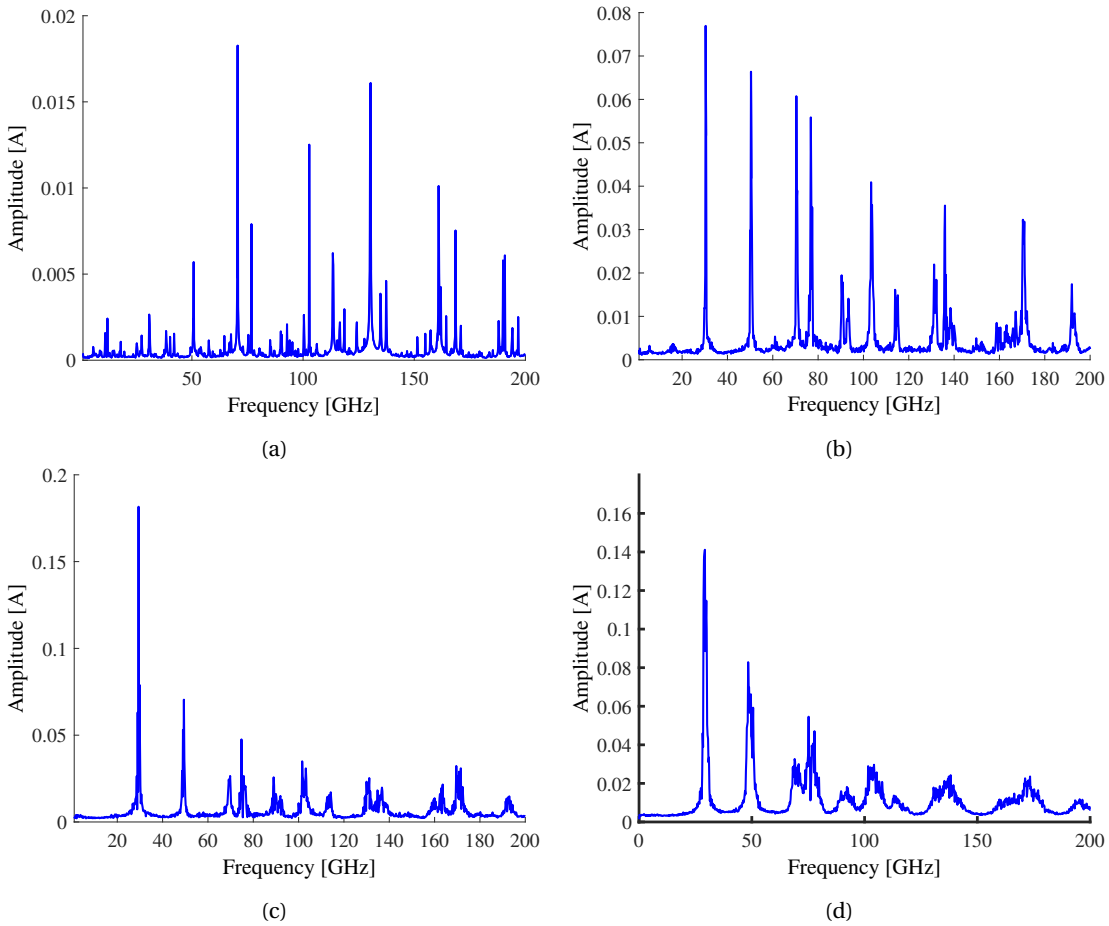


Figure B.11: Frequency response as obtained by Brownian motion for different temperatures. (a) T = 5 K, (b) T = 100 K, (c) T = 200 K, (d) T = 300 K.

It seems that at higher temperatures, more time is needed in order to ensure steady state. This becomes apparent when looking at the time response. The average out of plane motion of a subset of atoms is shown

in Figure B.12.

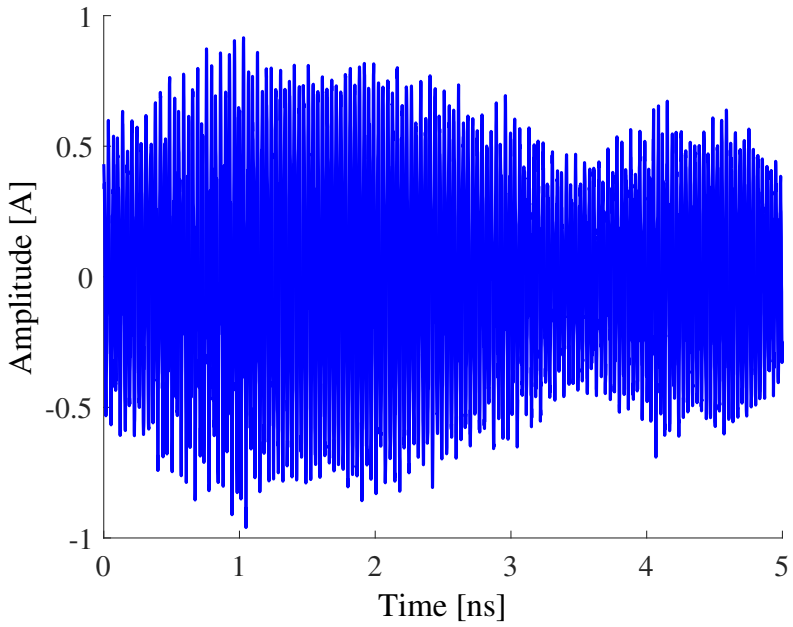


Figure B.12: Average time response when looking at Brownian motion, at 300 K from t0 = 5 ns to t = 10 ns.

Indeed, the time response does not show a converged solution, apparently energy is still being moved between modes.

As a final comparison to CM, the ratios between the higher order frequencies and the fundamental frequencies are calculated and compared. This ratio is defined as:

$$\omega_i = \frac{\omega_i}{\omega_0}. \tag{B.2}$$

Now the ratio can be compared to the known ratio's, for both plate and membranes. This is shown in Figure B.13.

Indeed, the normalized eigenfrequencies are different from either membrane or plate models. However, the ratio's compare much better to the membrane than plate model, as expected.

Thermal ripples The thermal ripples where already shown in the main part. Here, the deflection of the membrane is shown at 5 and 300 K, as to give an indication of the influence of temperature.

Note how the z-axis is scaled differently, the amplitude at 300 K is much higher. This amplitude is indeed comparable to the thickness of the membrane, which is 3.35 Å when considered as the interlayer distance. This again shows the difficulty of actuating the membrane with a significant energy as compared to thermal vibration, without going into the non-linear regime.

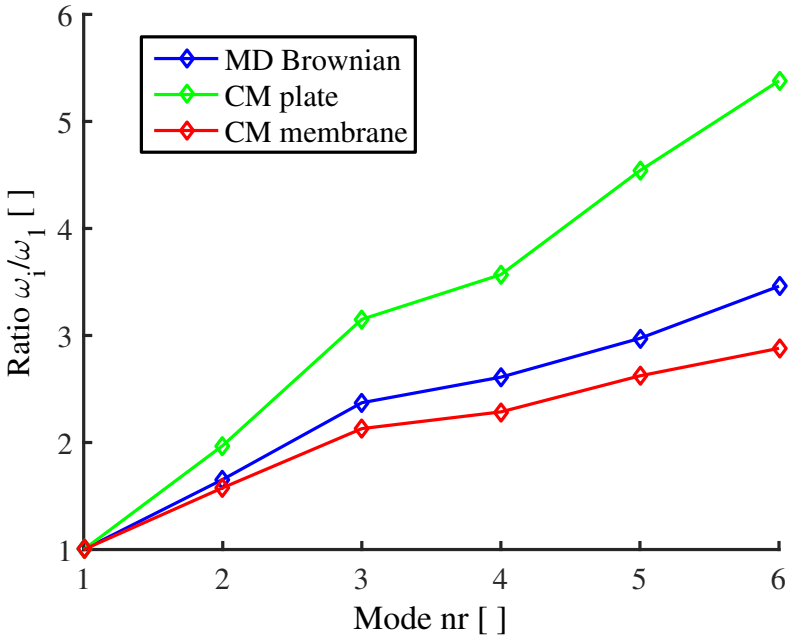


Figure B.13: Eigenfrequencies normalized with respect to the fundamental eigenfrequency. Comparing MD with CM plate and CM membrane.

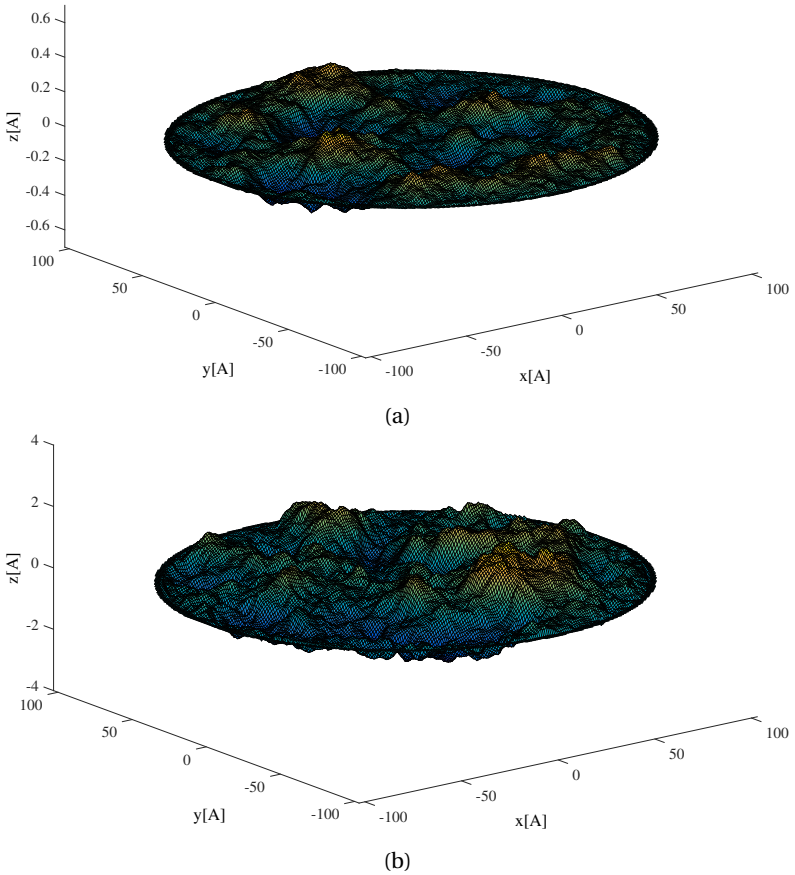
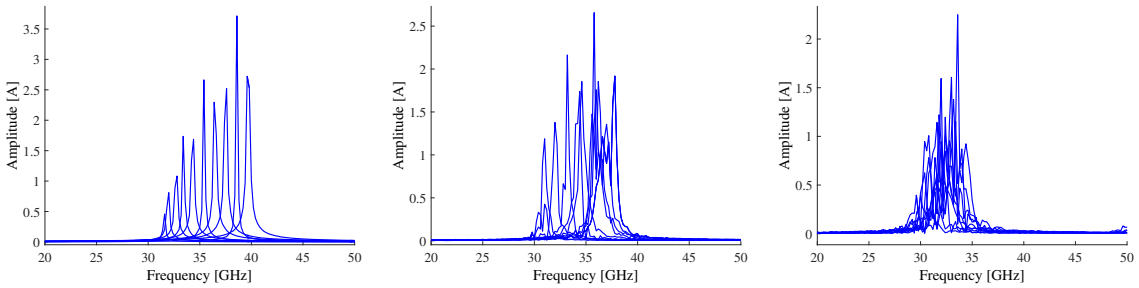


Figure B.14: Thermal ripples seen in Brownian motion at (a) 5 K and (b) 300 K .

B.2.5. Nonlinear free vibrations

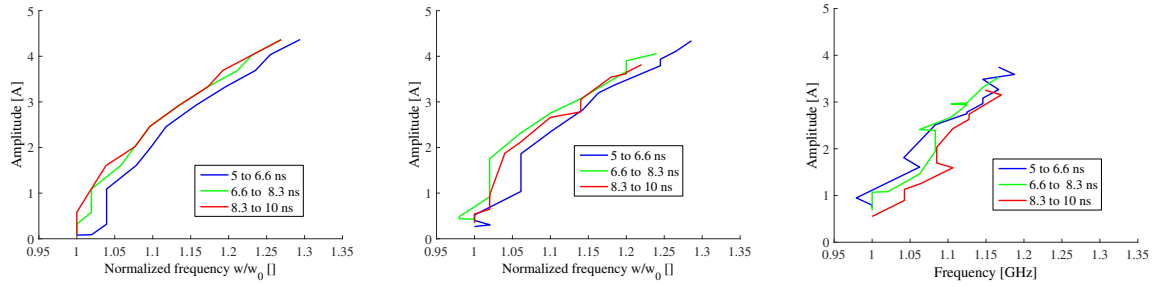
The next step would be to excite every mode independently. This would give more information on each eigenfrequency and allow us to identify the mode shapes. The big question however would be to at what initial velocity to actuate in order to minimize the non-linearity while keeping the signal to (thermal)noise ratio acceptable. Besides this, the previous results have shown that nonlinearities are playing a big role in the vibration response of this graphene membrane. Therefore, the frequency response for the first mode has been investigated for a range of actuation velocities. The frequency response is shown in Figure B.15a to Figure B.15c.



(a) (b) (c)

Figure B.15: First mode excited with an initial velocity from v0 = 0.1 Å ps^{-1} at (a) T = 5 K, (b) T = 150 K, (c) T = 300 K.

To get a clear relation between vibration amplitude and shift in eigenfrequency, the average maximal amplitude is plotted versus the normalized frequency shift in figures Figure B.16a to Figure B.16c.



(a) (b) (c)

Figure B.16: Maximum average amplitude versus normalized eigenfrequency at 3 time intervals for (a) T = 5 K, (b) T = 150 K, (c) T = 300 K.

At 5 K the first timespan has a little offset towards higher eigenfrequencies. This could signify that steady state may not have been reached yet. It is a fact that at such low temperatures, steady state is not as easily reached, because the thermal coupling is so slow. At higher temperatures, the backbones are not as clear, as the time response is not stable over time. Due to thermal mode coupling, different states are excited.

Now the backbones for the 3 different temperatures are compared in Figure B.17. As expected, the trend is very similar. For higher temperatures, the figures become more "noisy" as a result of the effect discussed before.

The problem of noise at higher temperatures was solved by dividing the total simulation time in smaller parts, such that the change of eigenfrequency within one part changes as little as possible, while maintaining a minimal resolution in the FFT. All points obtained this way are then fitted to, such that a single backbone is created, which best represents all the measurement points.

The last plot presented shows the decrease in amplitude over time, as given in Figure B.18.

The amplitude is steadily decreasing over time, and seems to have reached the steady state amplitude after 10 ns.

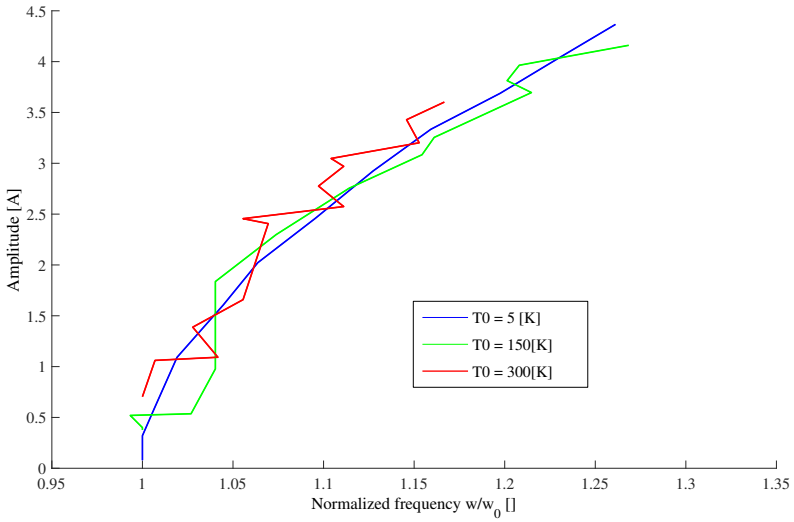


Figure B.17: Maximum average amplitude versus normalized frequency, comparing different temperatures

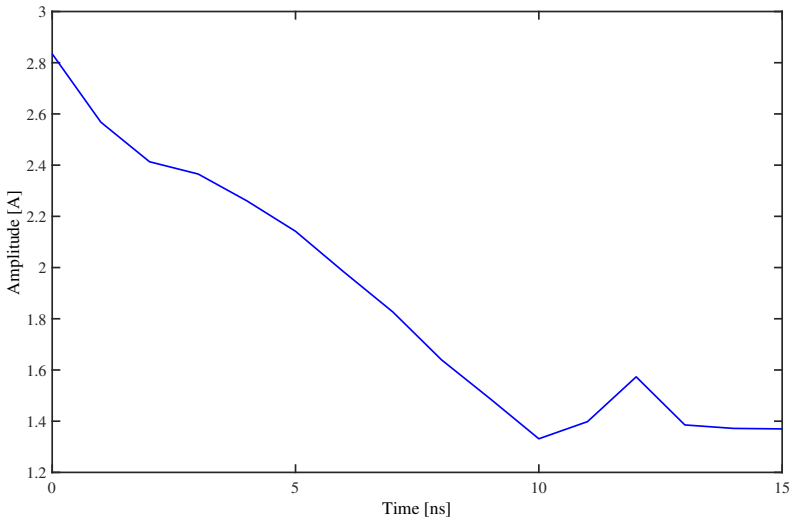


Figure B.18: Maximum average amplitude over time, at 300 K after exciting the membrane with an initial velocity profile of 1 $Åps^{-1}$

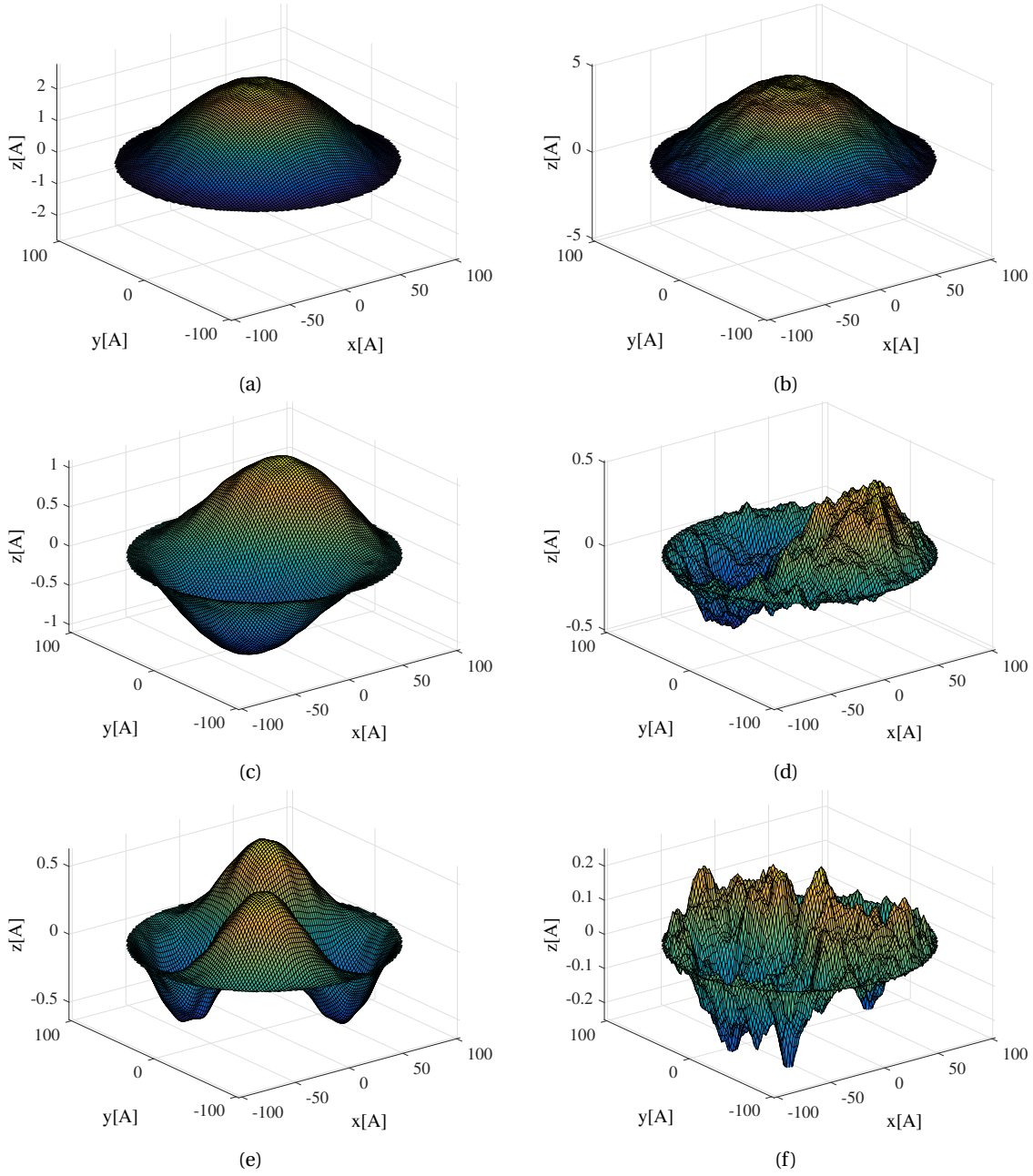
B.2.6. Free vibration response

3D mode-shapes Finally every mode was excited separately in order to check which eigenfrequency corresponds to which mode number and shape. This was done at 5, 150 and 300 K for an initial velocity of $0.5\,Åps^{-1}$. The first 6 modes have been investigated this way. In Figure B.19a to Figure B.19l the found mode-shapes are presented, at 5 and 300 K. These are the modes as found by averaging each maximal amplitude snapshot in the first 0.25 ns. Note that if one would look just at a single frame or a movie of vibration, the mode-shapes are usually hardly distinguishable. Only after averaging over a number of vibrations a good result can be found.

3D mode-shapes; Temperature influence If the mode-shapes are found at the end of the simulation, the results are much more influenced by thermal and nonlinear mode coupling. To illustrate this, the first and second mode are presented in Figure B.20a to Figure B.20d. These are the modes as found at the end of the simulation time, after 10 ns, at 5 and 300 K.

Indeed, with a higher temperature there is more noise at the end of the simulation. In general the mode-shapes are much less clear at the end of the simulation due to mode-coupling.

In order to obtain more accurate and less noisy mode shapes in this manner, one would have to take a longer simulation time in consideration, as to average out thermal noise. The problem however, is that the data needed to do this quickly becomes too much, as saving the position of 13000 atoms over say 10000



time steps is too much data to save and handle. Another method would be to obtain the mode shapes in a different manner, for example by filtering the time signal such that only a certain frequency range is left to pass through.

2D mode-shapes; Comparing with CM Now the mode-shapes are compared to the mode-shapes found in continuum mechanics. This is done by comparing the amplitude on a line through the membrane. The results are shown in Figure B.21a to Figure B.21l. Some modes could not be found well, mostly because they are simply not actually appearing strong enough anymore, as all the energy has been fed into other modes.

On the edges, a clear discrepancy with the continuum modes can be seen. This is due to the bending stiffness which graphene shows. A continuum membrane does not have any bending stiffness as the thickness is negligible. At higher temperatures the modes become less clear as well. This is due to the thermal noise and the increased mode-coupling. The fourth mode for example is not really visible at all, the first mode seems to be much stronger there. Interestingly the 6th mode can be found very clearly, where the lower order modes are much more disturbed by noise.

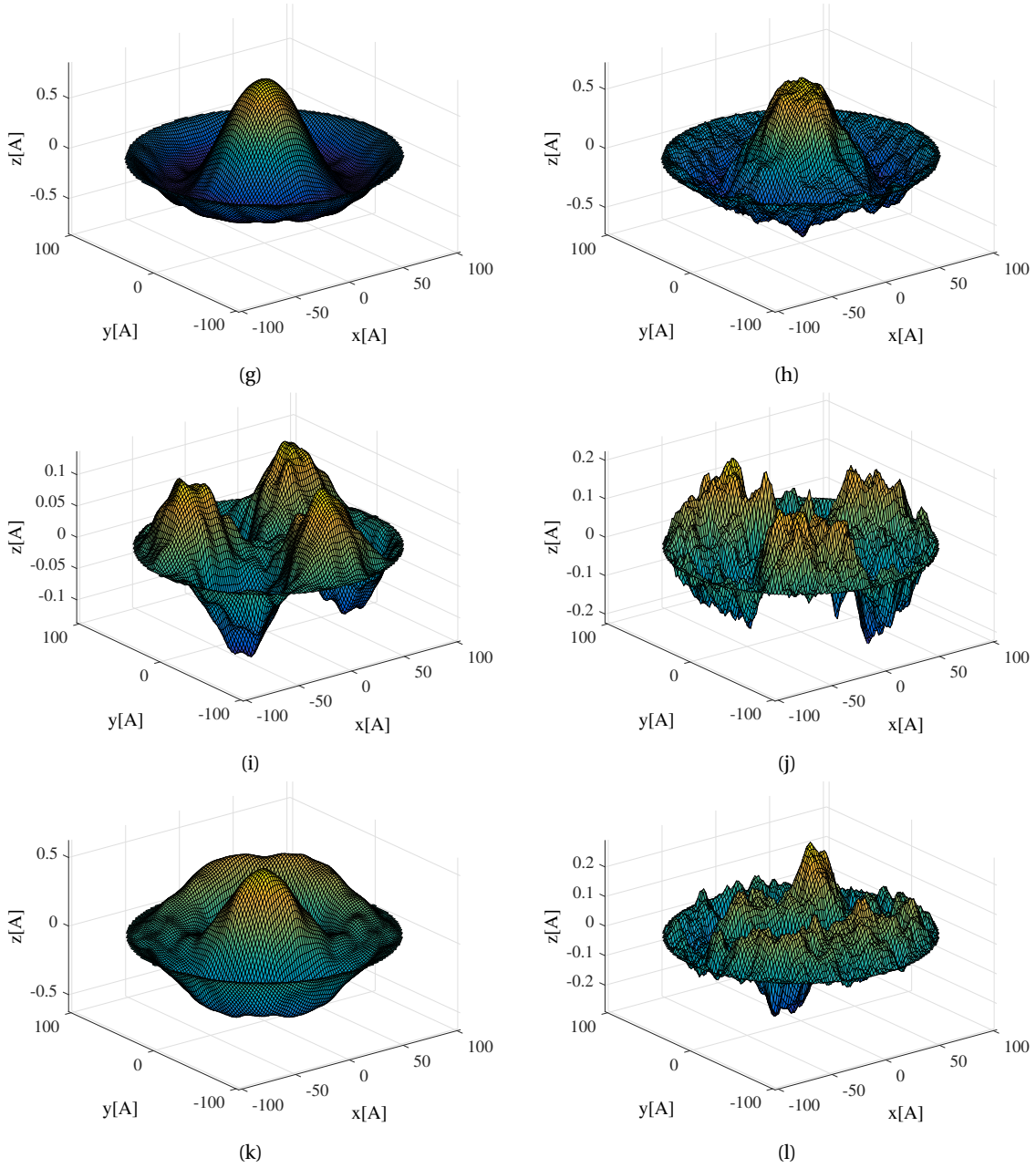


Figure B.19: Modeshapes of the first six modes, found at the start of the simulation, after exciting in that specific mode. Comparing the shape found at 5 K and 300 K.

Note how the fourth mode at the end of the simulation has already coupled all energy back to the first mode and is hardly visible. The difference between the modes found at the end of the simulation and the beginning is clear in terms of noise and overlapping modes.

Frequency response per mode In Figure B.22 the frequency response is shown, for the first six mode shapes, again comparing 5, 150 and 300 K.

Note that the coupling from axisymmetric(1,4) to non-axisymmetric modes appears to be very weak. Between axisymmetric modes the coupling is very strong, when exciting the 4th mode at 300 K, the first mode quickly becomes much more prominent. The coupling from non-axisymmetric modes to axisymmetric modes seems to be much faster, the first mode is almost always clearly visible. Furthermore, the coupling from 6th to first mode seems to be stronger than from second, third of fifth to first mode. This findings correspond quit well to what we found in comparing the modeshapes.

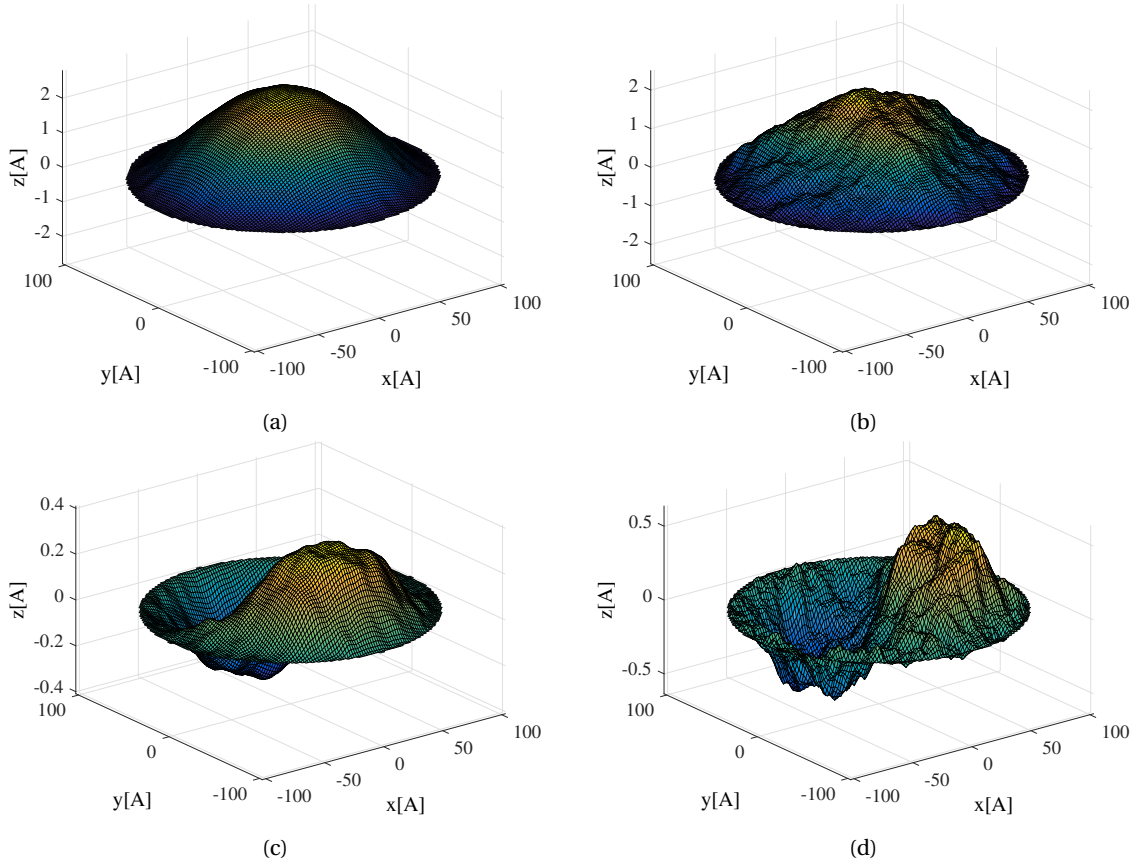
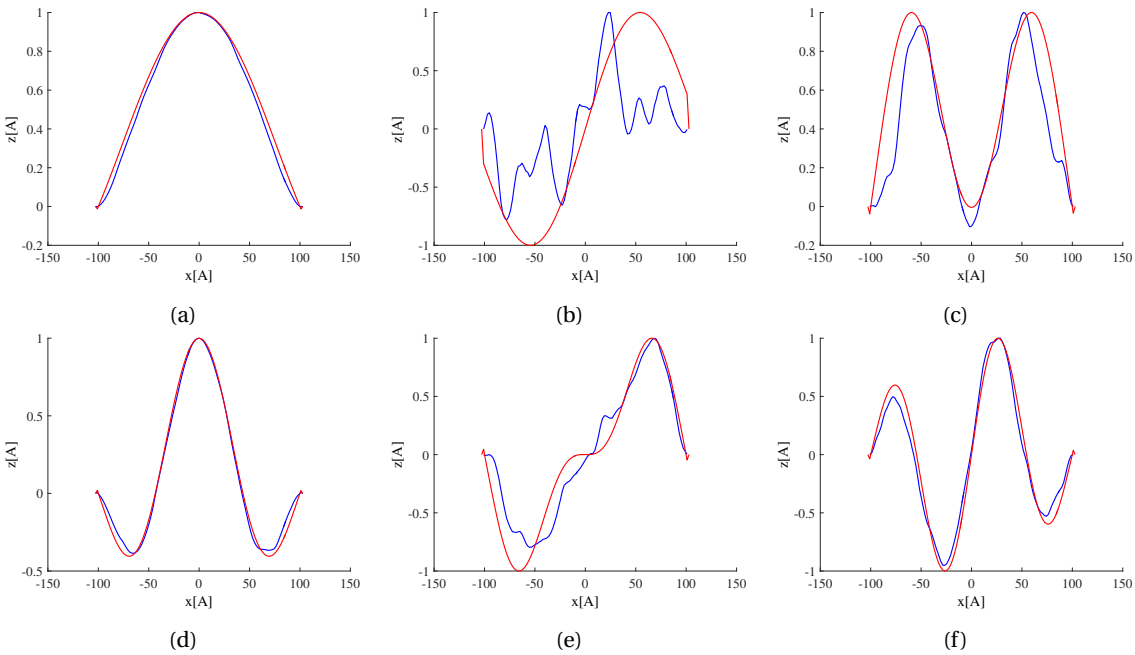


Figure B.20: Comparing the first and second mode shape as found after 10 ns of the simulation time, for (a; (c) 5 K and (b; (d) 300 K



Increasing the temperature always increases the coupling between modes, and excites all other modes through Brownian motion. At 150 K the amplitude of the first mode is always higher as compared to 5 K, whereas at 300 K the amplitude of the first mode is lower again, and the amplitude of all non-excited higher modes is often higher. Furthermore, at 300 K the frequency response is showing relatively much noise. This

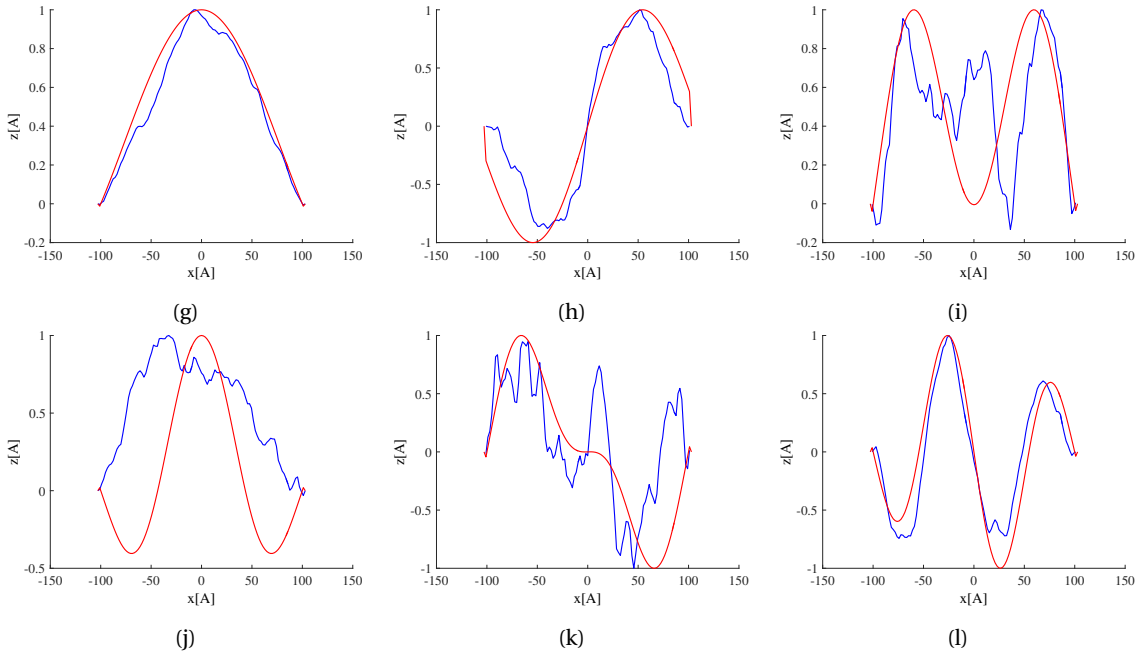


Figure B.21: Comparing the two dimensional mode shape obtained in MD (blue) with a CM membrane mode shape(red), all mode shapes are obtained at the end of the simulation, (a to (f at 5 K and (g to (l at 300 K.

is due to changes in state, steady state has not been reached, and multiple states pass during the timespan. As the frequency response is different per state, the total frequency response shows different peaks, seen as noise. See also Figure B.12, where the time response is shown, showing different stages when looking at the amplitude.

The Eigenfrequencies found through exciting each mode separately are summarized in Table B.2.

Table B.2: Eigenfrequencies as found by actuating every mode at 0.5 Å ps^{-1}.

	5 K	150 K	300 K
1st eigenfrequency	34.19 GHz	33.79 GHz	30.99 GHz
2nd eigenfrequency	57.59 GHz	51.79 GHz	50.39 GHz
3rd eigenfrequency	75.58 GHz	70.39 GHz	69.79 GHz
4th eigenfrequency	82.18 GHz	77.98 GHz	77.58 GHz
5th eigenfrequency	97.78 GHz	90.58 GHz	92.58 GHz
6th eigenfrequency	107.8 GHz	103.6 GHz	102.8 GHz

These eigenfrequencies are indeed very close to the values found using Brownian motion or when exciting all modes simultaneously. As compared to the results from Brownian motion, the found values are up to 14% higher for the excited case. This is attributed to a non-linear stiffening, resulting from a high amplitude of vibration. The error is quickly decreasing with increasing temperature, which is the result of the increased thermal mode coupling. Furthermore, the error is decreasing with an increasing frequency. This results from an increasing mode-coupling, due to thermal and non-linear mode coupling.

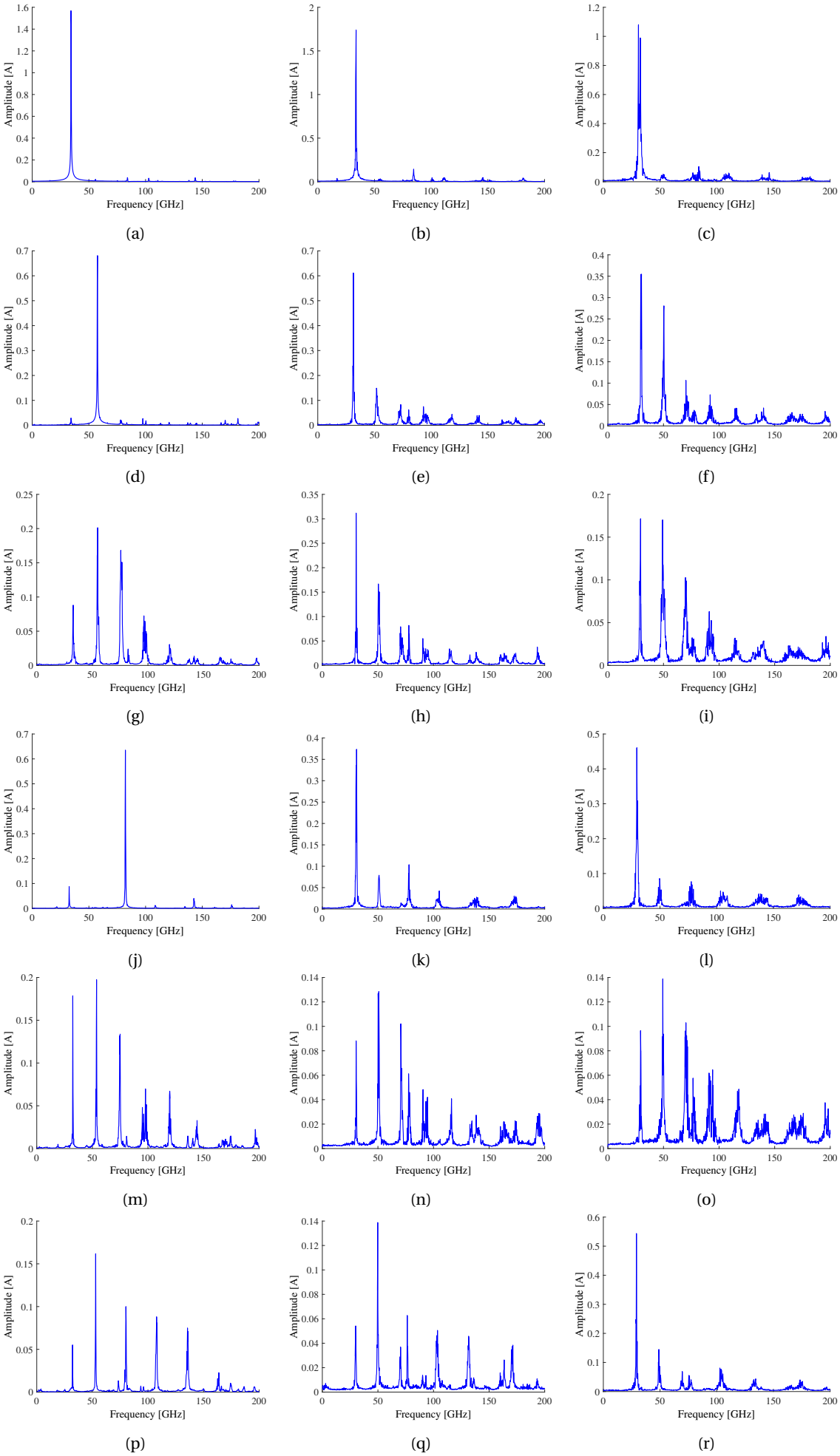


Figure B.22: Frequency response when exciting one mode, with an initial velocity of 0.5 Å ps^{-1}. Comparing 5, 150 and 300 K. (a to (c show the first mode at 5, 150 and 300 K respectively, (d to (f show the second mode at 5, 150 and 300 K respectively, etcetera.

B.3. Extracting parameters

As a final result, parameters could be extracted from the system. This is done by relating the measurements to known relations, as in the case of thermal expansion and Young's modulus, or by minimizing the difference between known values from CM and measured values, and hereby fitting parameters.

B.3.1. Thermal expansion and Young's modulus from boundary force

In order to allow for comparison with a CM model, the pre-strain on the membrane needs to be known. Therefore the Force on the boundary was measured. From this force the pre-strain along the boundary can be extracted. This is done for different temperatures, the resulting stress at the boundary as a function of temperature is shown in figure Figure B.23.

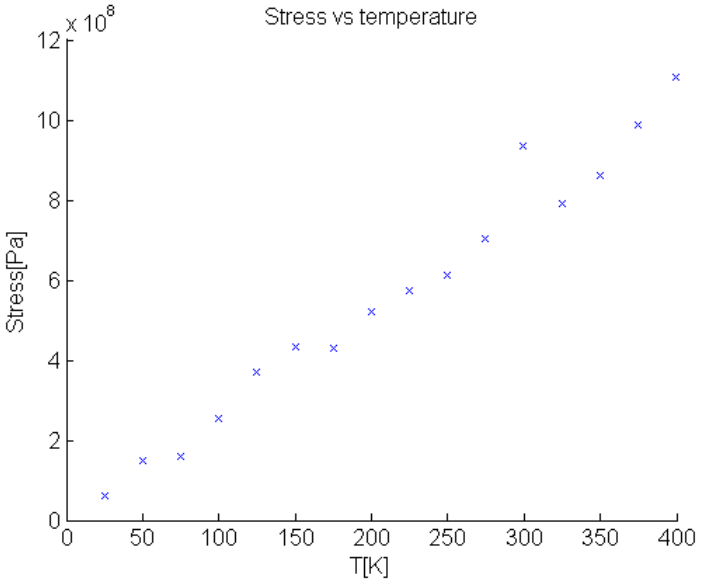
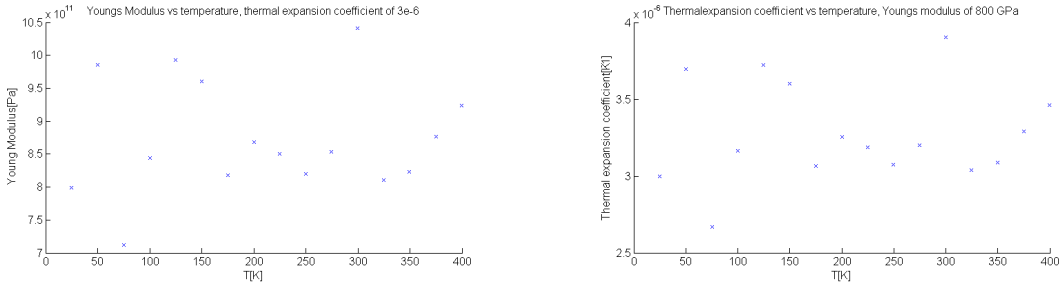


Figure B.23: Stress at the boundary for an increasing temperature

Indeed, the stress is linearly increasing with temperature, which is expected as graphene has a negative thermal expansion coefficient. This also shows that the dependency of thermal expansion coefficient on temperature is not strong.

Now one can assume a constant thermal expansion coefficient, and find the Young's modulus. This was indeed done. In Figure B.24a and Figure B.24b the Young's modulus and thermal expansion coefficient as function of temperature are shown.



(a) (b)

Figure B.24: The Young's modulus and thermal expansion coefficient, found from the stress at the boundary as a function of Young's modulus, temperature en thermal expansion. (a) Young's modulus versus Temperature, assuming a constant thermal expansion coefficient (b) Thermal expansion coefficient versus Temperature, assuming a constant Young's modulus

Note that the thermal expansion coefficient is in fact negative, in contrast to the data in the figure. Indeed the Young's modulus is found around 1 TPa.

As a final check, the Young's modulus can also be extracted from a static stress-strain test. This is done before thermalization, which means the membrane is still at zero Kelvin, and no thermal effects are present. The membrane is stretched radially, by applying a radial displacement every so many time steps. From the reaction force on the boundary and the applied strain, the Young's modulus can be extracted. The stress-strain curve is shown in Figure B.25a.

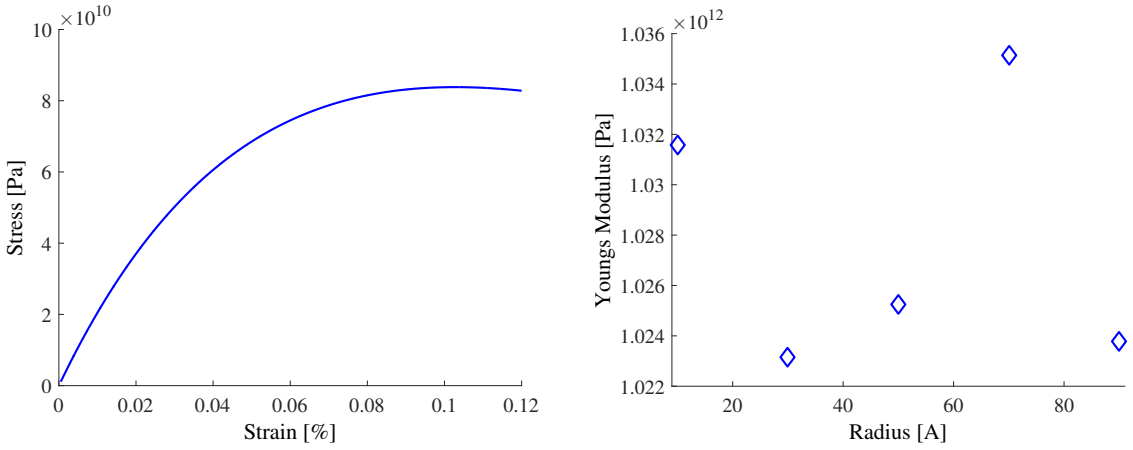


Figure B.25: (a) shows the Stress-Strain curve, obtained by radially stretching the membrane (b) shows the found Young's modulus for different radii.

Now the Young's modulus was obtained by looking at the change of Stress per strain, in the first region of strain. The Young's modulus was obtained for different radii of the membrane, as to be sure that the found value would not be size-dependent. The found Young's modulus for a few radii is shown in Figure B.25b. The found values are indeed very close to the most reported value of 1 TPa. Furthermore, they seem to be independent of radius. The small spread in values is expected to result from numerical errors in the simulation and post processing.

In subsection B.2.5 the influence of temperature and simulation time on the backbone was already discussed. It was found that the influence of temperature was negligible, although a higher temperature makes obtaining a backbone more difficult. As the simulation is changing over time, the time span considered should not be too big, as it would otherwise contain a too big spread in vibration amplitude and eigenfrequency. The backbone at 300 K was thus obtained by dividing the total simulation time in 3 equal parts. Each part now gives a data point, and all points are fit with a 4th order polynomial.

This backbone was then compared to backbones obtained with a numerical continuation technique, called Auto. The Young's modulus was found to be 1.08 TPa, which is very close to previously reported values, as well as the value obtained from a static stress-strain test. The measured points and fitted backbone are shown in Figure B.26a and the made fit is shown in Figure B.26b.

B.3.2. Bending rigidity and pre-strain from matching eigenfrequencies

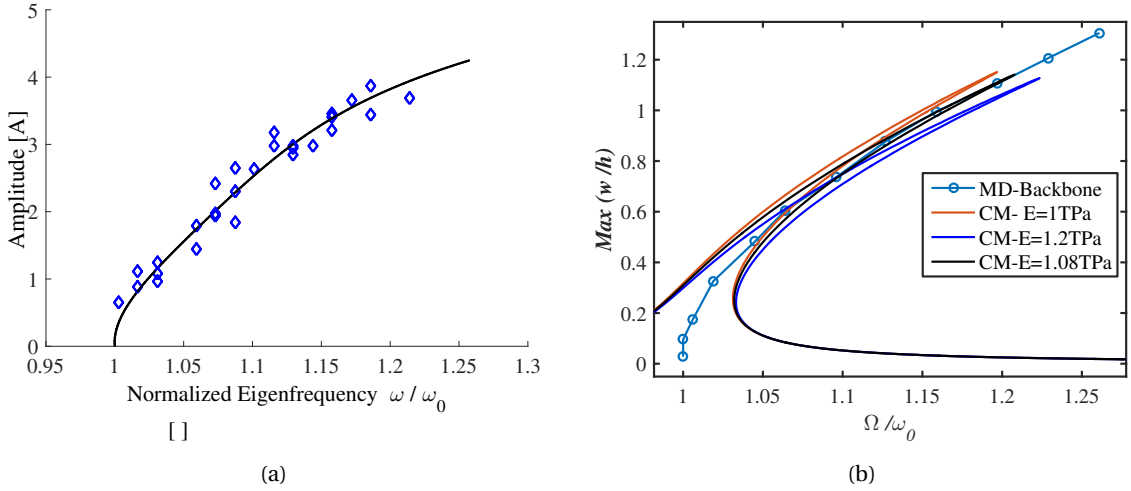
Now the Young's modulus is extracted, the next interesting parameter would be the bending rigidity. The bending rigidity is a parameter describing the out of plane stiffness k in eV.

Fit on k First the pre-strain was taken either as a constant, or as a known value as measured in MD. The resulting values for k and the corresponding RMS error are shown in Figure B.27.

The minimal error which can be obtained is around 10 %, which is too high. This can be explained because two parameters are fit, the absolute values and the ratios between eigenfrequencies. Therefore, it is not possible to fit with only one parameter.

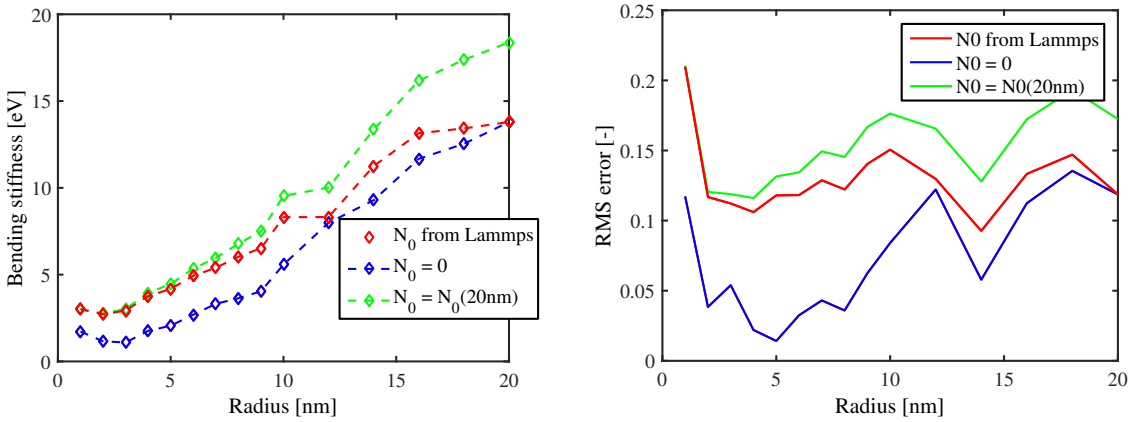
To solve this problem, another parameter needs to be add in order to make the fit more realistic and decrease the error. The pre-strain would be the most obvious choice and is also added in the optimization.

Fit on k and n0 Now n0 is added as a variable to the optimization, we are able to fit both the ratio and the absolute values of the eigenfrequencies. The resulting k and n_0 are shown in Figure B.28.



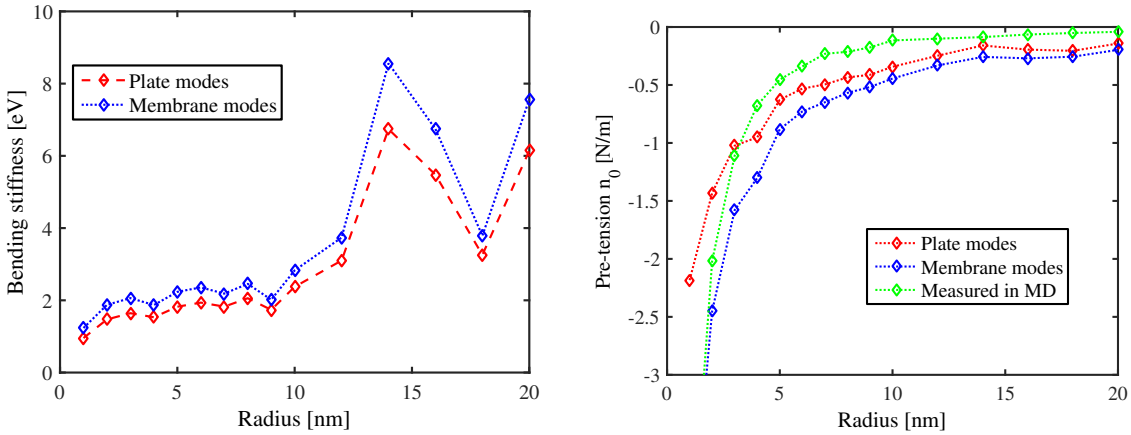
(a) (b)

Figure B.26: (a) measured points and fit backbone, (b) measured data together with the fit curve.



(a) (b)

Figure B.27: (**??**) Found values for k for different radii, taking the pre-strain n_0 as 0, as measured in MD or as the value measured for the biggest size. (b) Corresponding error versus radius.



(a) (b)

Figure B.28: Found values for k and n_0 for different radii, when optimizing for both bending rigidity and pre-strain.

Now indeed the values seem to have converged at 10 nm, although afterwards the fit values become very scattered. This is attributed to the faulty measurements in MD, as the size increases, the time to converges

also increases, as well as the computational cost. Therefore, it is harder to assure a converged solution. The found pre-strain is slightly higher than measured in MD and seems to converge as well. The error is shown in Figure B.29.

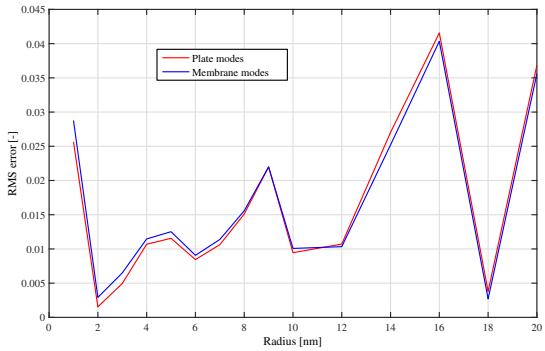


Figure B.29: Root mean square error for different radii, when fitting both bending rigidity and pre-strain using either plate or membrane mode shapes.

The found error is indeed much lower than in case of only fitting to k.

Temperature dependence The influence of temperature on the found bending rigidity was also investigated. The same procedure as described before was followed for a range of temperatures, at 10 nm. The found bending rigidity is compared to values as predicted by the analytical model as described in a paper by Rafael et al.[26]. Here, the bending rigidity of graphene is approximated as:

$$\kappa_R(q,T) = \kappa_0 + k_B T A \left(\frac{q_0}{q}\right)^{\eta}. \tag{B.3}$$

Here, $k_0 = 1\,\text{eV}$, k_B is the Boltzmann constant, T the temperature, q the wave vector of a flexural phonon and $\eta = 0.85$ is a characteristic exponent. q_0 is defined as:

$$q_0 = 2\pi\sqrt{\frac{Y}{\kappa}}, \tag{B.4}$$

with Y is the 2D Young's modulus.

And A is a temperature dependent coefficient, defined as $A = 5.9T^{\eta/2-1}$. Now the only unknown remaining is the wave vector q. One can define a value for q related to the membrane size as $q = \frac{2\pi}{\sqrt{A}}$, with A is the area.

Now the value for k can be approximated. The values found by optimization are compared to the calculated values in page 63.

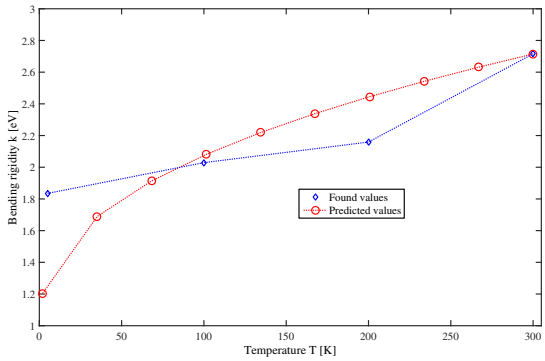


Figure B.30: Found values for bending stiffness as compared to predicted values

Interestingly the value at 300 K exactly overlaps. The values for lower temperatures are further off, although the trend is similar. Note however, that the lower temperatures were not obtained with as much care as the higher temperatures, as this is just a side-track of the main research. In conclusion this has shown that the found values are close to what is expected from theoretical predictions.

Discussion Now the performed optimization will be shortly discussed. Firs of all, it can be seen that the found bending rigidity is changing strongly with radius after 12 nm. This is partly contributed to the measurement being inaccurate due to the difficulties which arise when measuring such large membranes. In order to further investigate this, the error, as defined in **??**, is plotted as a function of both k and n_0, for different radii. This gives insight in how sensitive the found values are to slight changes in k and n_0. The surface plots for 6 radii are given in Figure B.31.

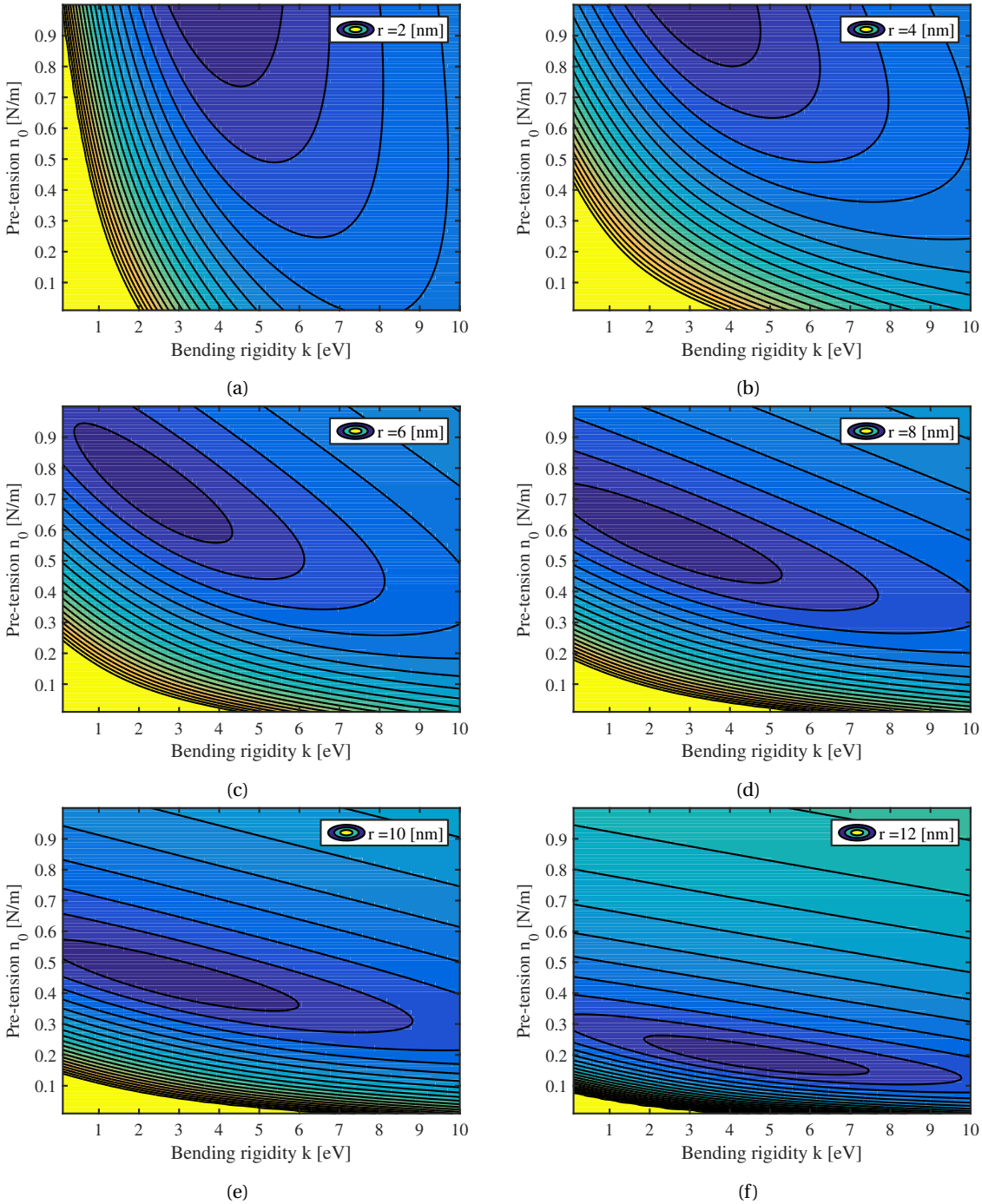


Figure B.31: Surface plots showing the error for different values of k and n_0. Each color indicates an area of 5% error. (a) 2 nm, (b) 4 nm, (c) 6 nm, (d) 8 nm, (e) 10 nm, (f) 12 nm.

The contour plots show that for bigger radii the area of errors smaller than 5 % becomes larger and spreads out such that the same magnitude of error can be obtained for a large range of values for k. This means the

result is very sensitive to small changes in the given parameters. In fact, this also means that the bending rigidity has a smaller effect on eigenfrequencies on a bigger scale than on a smaller scale.

To solve this problem, more modes can be added. Adding more information would reduce the influence of measurement errors. In general, the first five eigenfrequencies were used. The convergence over the first five eigenfrequencies is shown for the same 6 radii in Figure B.32.

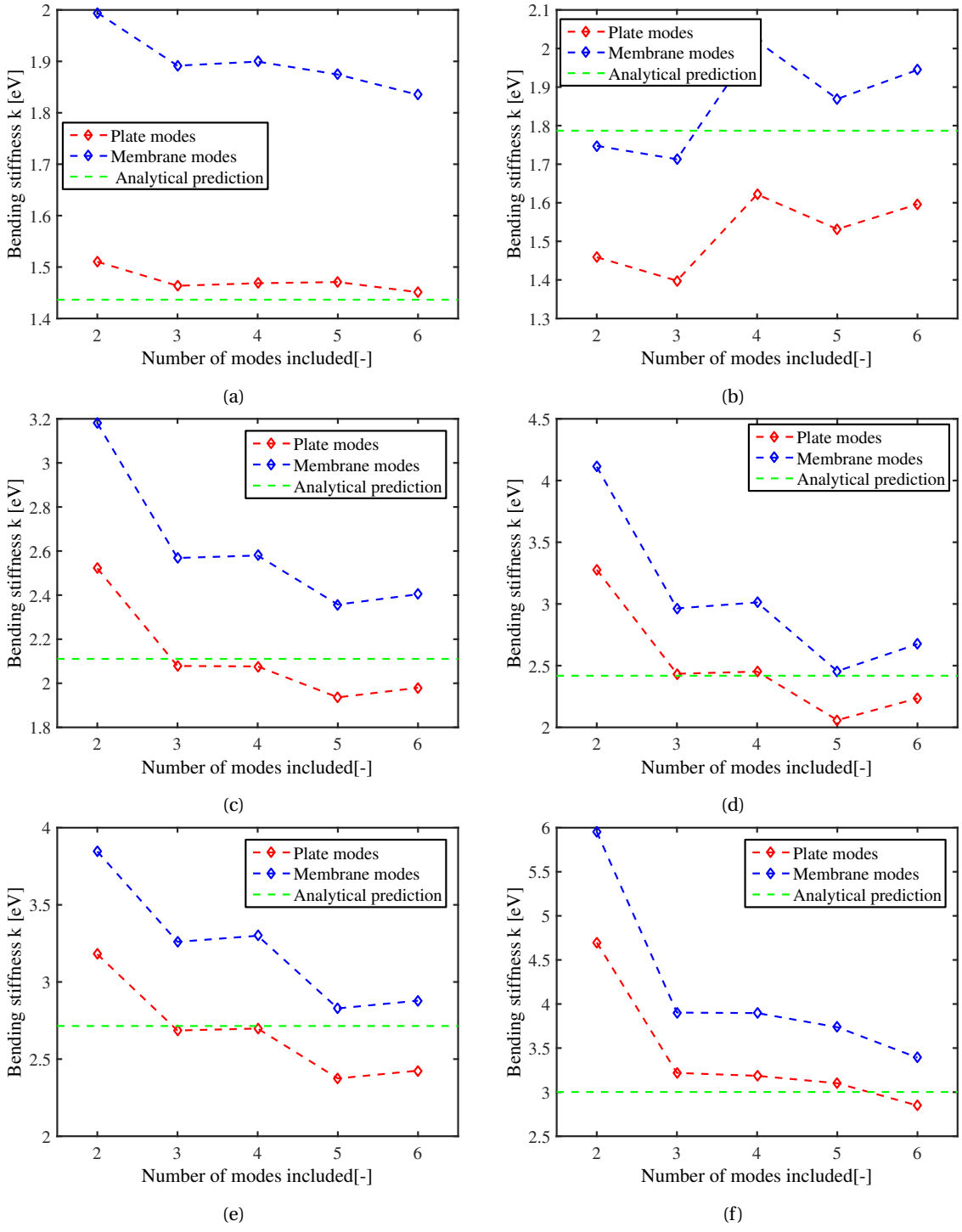


Figure B.32: Convergence of bending rigidity when including multiple modes for different radii. The red dotted line gives the value as predicted by Rafael Roldán et al.[26]. (a) 2 nm, (b) 4 nm, (c) 6 nm, (d) 8 nm, (e) 10 nm, (f) 12 nm.

It seems as if adding more modes generally makes the solution converge, although in many cases convergence has not been reached. A possible solution could be to add more modes, such that all solutions

converge. However, in order for the frequency response to be clear enough to distinguish more modes with certainty, the simulation needs to run for much longer.

C

Validation of the continuum model by Finite Element Simulations

In this appendix, the continuum model used will be validated by comparing the found eigenfrequencies with a finite element model. The Finite element model is made in Comsol.

The used parameters are listed in Table C.1. Note that in case of a plate, the pre-tension is set to 0.

Table C.1: Parameters used in the continuum and finite element models.

Parameter	Value	Units
Young's modulus	1.04	TPa
Poison ratio	0.16	–
Pre-tension	0.318 25	$\mathrm{N\,m^{-1}}$
Radius	10	nm
Density	2300	$\mathrm{kg\,m^{-3}}$
Thickness	3.35	Å

Now the obtained values are given in Table C.2. Here, the error between the models is also given.

Table C.2: Obtained eigenfrequencies and the error between those found in CM and FEM

	Plate			Membrane		
Eigenfrequency	CM	FEM	Error	CM	FEM	error
First	33.862 GHz	33.860 GHz	0.0059 %	24.598 GHz	24.598 GHz	0 %
Second	70.497 GHz	70.363 GHz	0.19 %	39.193 GHz	39.193 GHz	0 %
Third	115.66 GHz	115.26 GHz	0.35 %	52.530 GHz	52.530 GHz	0 %
Fourth	131.83 GHz	131.40 GHz	0.33 %	56.461 GHz	56.463 GHz	0.0035 %
Fifth	169.19 GHz	168.36 GHz	0.49 %	65.237 GHz	65.260 GHz	0.035 %

As can be seen, the error is always well below 1 %. The error in the membrane mode is almost non existents. The error in the plate model is mostly attributed to the mesh in the FEM model, rather than an error in the CM model. This could have been solved easily by adapting a finer mesh, but the main goal here is to verify the continuum model. This goal is indeed reached.