UNIVERSITY OF COPENHAGEN
FACULTY OF SCIENCE

NIELS BOHR INSTITUTE

MSc in Physics

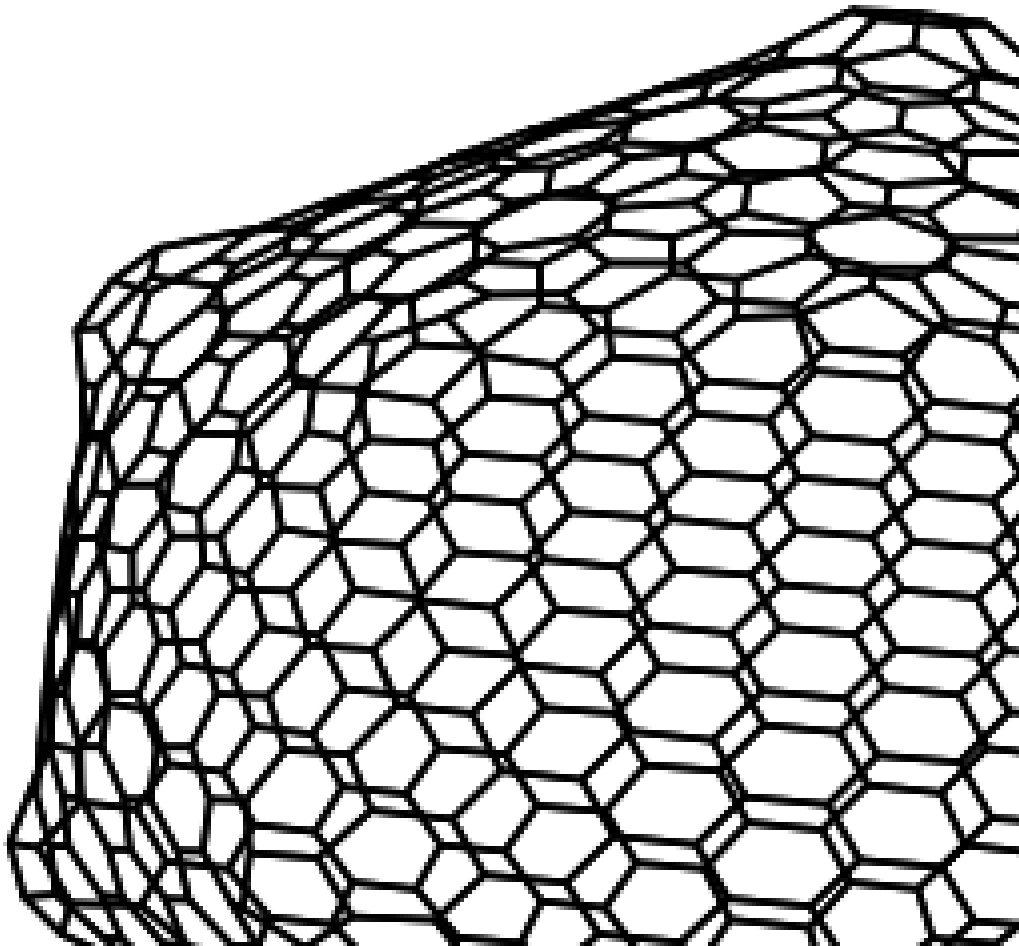
Molecular Shapes of Fullerenes

A robust force field method for fullerenes and polyhedral molecules

Buster N. Pedersen

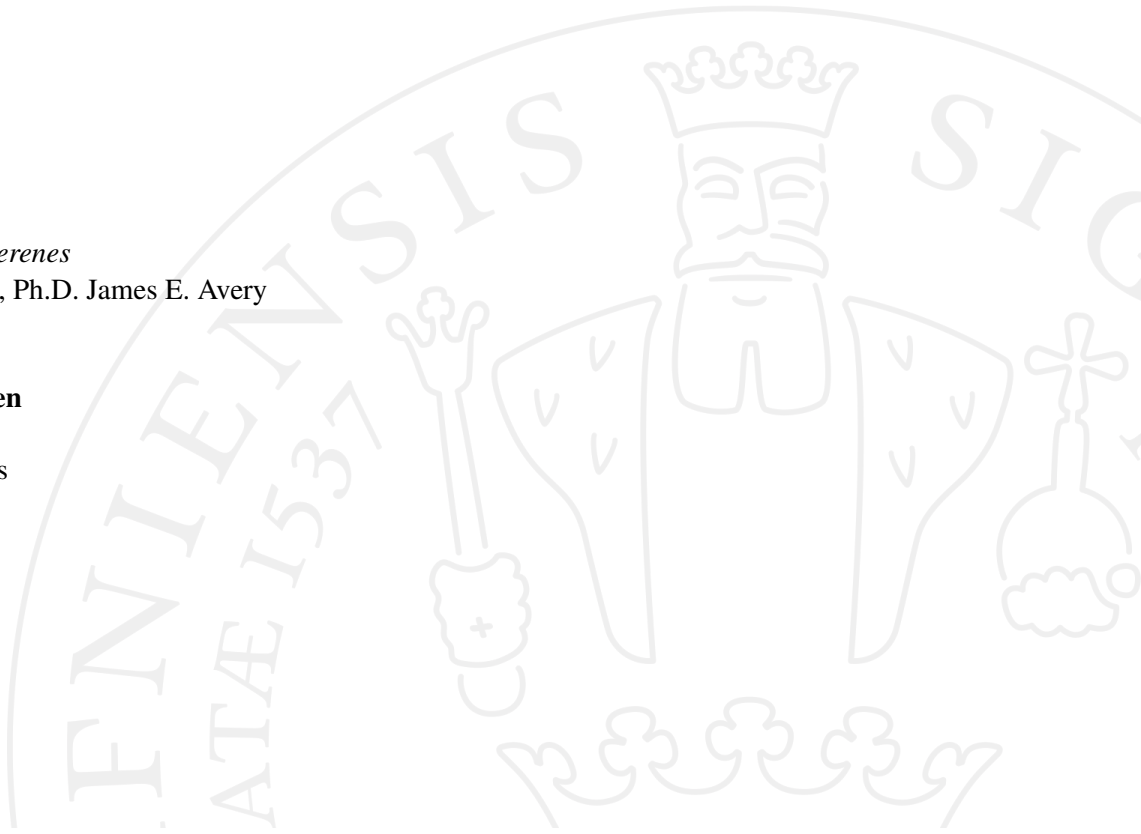
Supervised by Assoc. Prof., Ph.D. James E. Avery

Handed in: September 29, 2020



Buster N. Pedersen
Molecular Shapes of Fullerenes
Supervisors: Assoc. Prof., Ph.D. James E. Avery

University of Copenhagen
Niels Bohr Institute
Masters Degree in Physics
Blegdamsvej 17
2100 Copenhagen N



ABSTRACT

Fullerenes are a class of polyhedral carbon molecules with surfaces consisting only of pentagonal and hexagonal carbon rings. Theoretically there exists infinitely many stable fullerenes, of which only a handful have been created and analysed. The chemical and electronic properties of fullerenes are of major interest to the scientific community and their applications have played essential roles in revolutionary materials, medical drug delivery systems, and electron acceptors in solar cells. Fullerenes comes in many shapes and sizes with the number of isomers growing as $\mathcal{O}(N^9)$ with N being the number of carbon atoms. It is desirable to search through these enormous fullerene isomer spaces for fullerenes with specific physical properties and shapes. However this computational task proves impractical for ab inito quantum methods due to long computational time and thus, faster methods are needed.

This thesis offers an improvement to an already existing software which uses geometry force field methods in order to effeciently determine optimal molecular shapes for fullerenes. Furthermore a vectorized mathematical notation is derived, making the calculations and computations easily interpretable, in terms of geometry operations, and verifiability, as the mathematical calculations are directly translated into simple vectorized code. The code is easy to extend with additional physics and offers an good foundation for optimal paralellization. Harmonic force constants are obtained for the C_{60}-I_h Bucky ball through sub-Hessian matrix projection of multiple DFT calculations, namely **b3lyp**, **b3lyp+gd3**, **cam-b3lyp** and **m062x**.

With this new corrected fullerene geometry force field, implemented and used in this thesis, an assortment of fullerene structures have been computed and compared to ab inito quantum optimized geometries, of which all have been in good agreement.

Contents

List of Figures

List of Tables

BIBLIOGRAPHY

[1] H. W. Kroto, J. R. Heath, S. C. O'Brien, R. F. Curl, and R. E. Smalley, "C 60 : Buckminsterfullerene," *Nature*, vol. 318, pp. 162–163, Nov. 1985. Number: 6042 Publisher: Nature Publishing Group. (Cited on pages 1, 2, and 5.)

[2] J. E. Avery, "Wave equations without coordinates I: fullerenes," *Rendiconti Lincei. Scienze Fisiche e Naturali*, vol. 29, pp. 609–621, Sept. 2018. (Cited on page 9.)

[3] P. Schwerdtfeger, L. N. Wirz, and J. Avery, "The topology of fullerenes," *WIREs Computational Molecular Science*, vol. 5, no. 1, pp. 96–145, 2015. _eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1002/wcms.1207. (Cited on pages 1, 5, 9, 11, and 12.)

[4] L. T. Scott, M. M. Boorum, B. J. McMahon, S. Hagen, J. Mack, J. Blank, H. Wegner, and A. d. Meijere, "A Rational Chemical Synthesis of C60," *Science*, vol. 295, pp. 1500–1503, Feb. 2002. Publisher: American Association for the Advancement of Science Section: Report. (Cited on page 1.)

[5] "House of Graphs - Fullerenes." (Cited on page 2.)

[6] Z. Wu, D. A. Jelski, and T. F. George, "Vibrational motions of buckminsterfullerene," *Chemical Physics Letters*, vol. 137, pp. 291–294, June 1987. (Cited on pages 2 and 16.)

[7] L. N. Wirz, *Graph theoretic and electronic properties of fullerenes, &, Biasing molecular modelling simulations with experimental residual dipolar couplings : a thesis presented in partial fulfilment of the requirements for the degree of Doctor of Philosophy at Massey University, Albany, New Zealand*. Thesis, Massey University, 2015. Accepted: 2016-06-13T22:53:51Z Journal Abbreviation: Biasing molecular modelling simulations with experimental residual dipolar couplings. (Cited on pages 2, 14, and 23.)

[8] P. Schwerdtfeger, L. Wirz, and J. Avery, "Program Fullerene: A software package for constructing and analyzing structures of regular fullerenes," *Journal of Computational Chemistry*, vol. 34, no. 17, pp. 1508–1526, 2013. _eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1002/jcc.23278. (Cited on pages 2, 14, and 23.)

[9] E. Ōsawa, H. W. Kroto, P. W. Fowler, and E. Wasserman, "The evolution of the football structure for the C60 molecule: a retrospective," in *The Fullerenes: New Horizons for the Chemistry, Physics and Astrophysics of Carbon* (D. R. M. Walton and H. W. Kroto, eds.), pp. 1–8, Cambridge: Cambridge University Press, 1993. (Cited on page 5.)

[10] A. H. Francis, "An Atlas of Fullerenes By P. W. Fowler (University of Exeter), D. E. Manolopoulos (University of Nottingham). Oxford: New York. 1995. viii + 392 pp. $98.00. ISBN 0-19-855787-6.," *Journal of the American Chemical Society*, vol. 118, pp. 5161–5161, Jan. 1996. Publisher: American Chemical Society. (Cited on pages 6 and 18.)

[11] F. Jensen, *Introduction to computational chemistry*. 2017. OCLC: 957680749. (Cited on page 27.)

[12] "Advanced computer architecture: parallelism, scalability, programmability," *Choice Reviews Online*, vol. 31, no. 03, p. 31. (Cited on page 29.)

[13] "Computer Organization and Design MIPS Edition - 5th Edition."

[14] "William Stallings - Computer Organization and Architecture Designing for Performance (8th Edition).pdf." (Cited on page 29.)

[15] J. M. Seminario, "Calculation of intramolecular force fields from second-derivative tensors," *International Journal of Quantum Chemistry*, vol. 60, no. 7, pp. 1271–1277, 1996. _eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1002/%28SICI%291097-461X%281996%2960%3A7%3C1271%3A%3AAID-QUA8%3E3.0.CO%3B2-W. (Cited on page 31.)

[16] "Gaussian.com | Expanding the limits of computational chemistry." (Cited on pages 33 and 34.)

[17] W. H. Press, W. H. e. a. Press, S. A. Teukolsky, W. T. Vetterling, W. T. Vetterling, B. P. Flannery, and B. P. Flannery, *Numerical Recipes in C++: The Art of Scientific Computing*. Cambridge University Press, Feb. 2002. Google-Books-ID: mSLhDt_XIUQC. (Cited on page 35.)

[18] M. T. Heath, *Scientific Computing: An Introductory Survey, Revised Second Edition*. Philadelphia: SIAM - Society for Industrial and Applied Mathematics, second edition ed., Nov. 2018.

[19] J. R. Shewchuk, "An Introduction to the Conjugate Gradient Method Without the Agonizing Pain," tech. rep., Carnegie Mellon University, Pittsburgh, PA, USA, 1994. (Cited on page 35.)

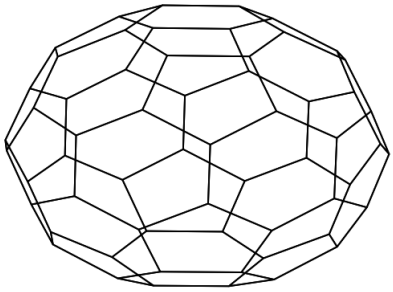
[20] G. M. Amdahl, "Validity of the single processor approach to achieving large scale computing capabilities," in *Proceedings of the April 18-20, 1967, spring joint computer conference*, AFIPS '67 (Spring), (New York, NY, USA), pp. 483–485, Association for Computing Machinery, Apr. 1967.

[21] A. E. A. Allen, M. C. Payne, and D. J. Cole, "Harmonic Force Constants for Molecular Mechanics Force Fields via Hessian Matrix Projection," *Journal of Chemical Theory and Computation*, vol. 14, pp. 274–281, Jan. 2018. (Cited on page 52.)

[22] U. Dinur and A. T. Hagler, "New Approaches to Empirical Force Fields," in *Reviews in Computational Chemistry*, pp. 99–164, John Wiley & Sons, Ltd, 2007. _eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1002/9780470125793.ch4. (Cited on page 53.)

[23] B. K. Ganser, S. Li, V. Y. Klishko, J. T. Finch, and W. I. Sundquist, "Assembly and Analysis of Conical Models for the HIV-1 Core," *Science*, vol. 283, pp. 80–83, Jan. 1999. Publisher: American Association for the Advancement of Science Section: Report. (Cited on page 56.)

[24] S. Mattei, B. Glass, W. J. H. Hagen, H.-G. Kräusslich, and J. A. G. Briggs, "The structure and flexibility of conical HIV-1 capsids determined within intact virions," *Science*, vol. 354, pp. 1434–1437, Dec. 2016. Publisher: American Association for the Advancement of Science Section: Report. (Cited on page 56.)



"Human beings always do the most intelligent thing...
after they have tried every stupid alternative and none of them have worked"
- Richard Buckminster Fuller

1

INTRODUCTION

Fullerenes are hollow, three-dimensional molecules formed entirely by carbon atoms. The first fullerene discovered was the football shaped C_{60}-I_h isocahedral symmetric fullerene by Kroto et al. in 1985[1] and it consists of 60 carbon atoms in 20 hexagonal and 12 pentagonal faces. At the point of their discovery the geometrical structures of fullerenes were already well admired by *Richard Buckminister Fuller* a famous arcitect, who ended up being the name giver to the family of molecular structures, as his geodesic dome building are graph duals to fullerenes. Fullerenes come in many shapes and with an array of possible applications e.g. biomedical research through hydrogen storage to solar cell development. The geometric properties of fullerenes, with only pentagonal and hexagonal faces, as well as their distinctive chemical bond structure makes them well suited for systematic analysis based on computational methods and graph theory.

Theoretically there exists infinitely many fullerenes with the number of isomers growing as $O(N^9)$[3] however only a few have been produced, like the most spherical isomers of C_{70} and C_{78} which have been realized and studied. However, present techniques of fullerene productions are exceptionally limited and rely upon procedures like laser ablation, resistive heating of graphite, and a bit of luck. Since these methods can produce large quantities of small fullerenes but are incapable of producing new specific fullerenes advancement is limited. In 2002 Scott et al.[4] proposed a solution to the problem and introduced a new method of fullerene production. A rational chemical synthesis path for the C_{60}-I_h fullerene from an auto-assembling planar precursor molecule. The precursor molecule consisted of all 60 carbon atoms and 75 of the 90 carbon bonds present in the C_{60}-I_h. Furthermore chlorine was placed at specific positions and when exposed to at flash vacuum pyrolysis, the precursor molecule would auto assemple. The method has proven groundbreaking and has lead to selective production of other small fullerenes, such as the C_{78} and C_{84} fullerene.

However calculating the positions of the chlorine atoms is still a trial and error process and further work needs to be done.

Another challenge presented by the fullerenes is the enormous isomer space. With isomers growing as $\mathcal{O}(N^9)$ one could spend an eternity listing everyone. There exist 1.812 C_{60} fullerene molecules, 1.674.171 C_{120} fullerenes and 132.247.999.328 C_{400} fullerene isomers and of all those fullerene isomers (1, 10.774 and 42.432.922.921, respectively)[5] exists which uphold the Isolated pentagon rule (IPR)[1] and thus are thermodynamically stable. The handful of fullerenes that have already been synthesized have a huge impact in general science and are used in many fields. Knowing this, it is desirable to navigate these huge isomer spaces in search of new possible fullerenes with desired properties. Unfortunatly, accurate quantum chemical calculations are not feasible for thousand upon thousands of possible fullerene molecules, since the cumbersome quantum calculations take too long. A faster method for assessing fullerenes is therefore needed.
The Born-Oppenheimer approximation (BO) is based on the large difference in nucleus and electron masses. The mass of the nucleus surpass the mass of the electron by several orders of magnitudes and given the same kinetic energy, the movement of a single electron is much more rapid than of a nucleus. For carbon, the size of the nucleus is roughly $\sim 1/33.000$ the size of the atom radius, so we can treat the nucleus being treated as a point particle in the electron potential, which is assumed to instantaneously readjust itself as we moce the nuclei. The BO approximation thus lets us treat the nuclei as classical particles moving around each other in an ever changing potential created by the electrons and nuclei. In molecular geometry force field optimization we also start with the BO approximation, however the complicated nucelus-electron potential is replaced with a much simpler potential. This serves the purpose of avoiding complicated ab inito quantum chemistry calculations at each iteration and speeding the computational process up by several orders of magnitude. However some accuracy is lost in the process, but the physical energy and geometry close to the optimum can often be captured by the second order Taylor expansion, if we can calculate a good approximation to the Hessian matrix.
Through the neat graph properties of fullerenes, the *Alexandrov's uniqueness theorem* implies that it is possible to find good fullerene geometries. Previous work done by Wu et al.[6] and the work from Wirz et al.[7, 8] demonstrates that fullerene geometries, which are in good agreement with ab inito quantum mechanical calculations. can be obtained through geometry force field methods.

This thesis investigate the possibilities of improving an already exisisting force field from Wirz et al. The force field implemented by Wirz et al. works well for small fullerenes, when it works. However errors will occur, which results in fullerenes not optimizing correctly, with the cases growing propor-

tional to N_{atoms}. Furthermore the code is written in *Fortran* with thousands of lines of code, with no existing basic documentation of the implemented derivatives or calculations, which makes the force field hard to debug, extend, and understand. Also, the original author of the force field has left academia. Thus, in order to create a force field which is, more robust, easier to extend with new physics, and more straightforward to understand, this thesis have had to start from the ground up. New force field gradients have been derived in a simple and vectorized notation, with accessible documentation. The vectorized notation allows for easy verification of derivatives and makes parallelization possible.
Furthermore, force field parameters have been calculated through Hessian matrix projections of multiple density functional theory (DFT) calculations of the C_{60}-I_h molecule. The results of the expanded force field method were compared to ab inito quantum mechanical DFT calculations with and without the new force field parameters and the possibilities of adding new geometric terms to the force field are discussed.

1.1 BUILDING AN FULLERENE INTUITION

This thesis is part of a larger collaboration exploring the mathematics, physics, and chemistry of polyhedral molecules in order to obtain as much information about fullerenes as possible. The project is led by James E. Avery at the University of Copenhagen, (UofC) (web-page) in collaboration with Peter Schwerdtfeger at Massey University, New Zealand (web-page). It is currently funded through Avery's VILLUM Experiment project 00023321 "Folding Carbon: A Calculus for Molecular Origami".

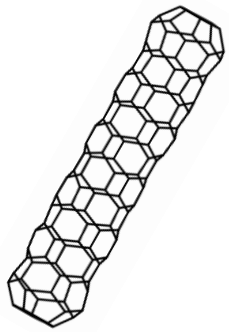
The overall aim is to search through the vast fullerene isomer spaces and procure fullerenes that have desired physical properties, however we face the problem of ab inito quantum methods being too slow for the task. Therefore it is desirable to create fast methods that can search for fullerenes with specific physical properties.
What the collaboration aims to do is to obtain a broad spectrum of physical, chemical, graph-theoretical, geometrical and mathematical properties for fullerenes and polyhedral-molecules. Many properties require good geometries to obtain, and so it is essential to have rapid, robust methods that produce fullerene geometries that are as physically correct as possible, which is what the work of this thesis aims to accomplish.

1.2 SOFTWARE

The software developed in this thesis can be found on https://github.com/Buster220992/Thesis-Molecular-Shapes-of-Fullerenes. The project contains:

- Methods to calculate molecular energies and gradients
- Methods to optimize and find optimal fullerene geometries
- Methods to automatically extract information from *Gaussian16* ".log" files
- Scripts to analyse and visualize results
- Scripts to calculate fullerene force constanst from the Seminario method

Wherever appropriate, the methods will be incorporated into a central code base for the "Folding Carbon project" (web-page). As this project is a work in progress, the content is most likely to be changed in the future.



2

FULLERENES, TOPOLOGY AND A BIT OF GRAPH THEORY

2.1 FULLERENES

Carbon is the fourth most abundant element in the universe making it among the most alluring atoms to work with. Solid carbon comes in several naturally occurring allotropes depending on the type of chemical bond. Two common allotropes are diamond and graphene. In diamond the carbon atoms form sp^3 orbital hybrids, which means that each carbon atom forms bonds to the four nearest neighbours in a tetrahedral shape. In graphene the bonds are sp^2 hybridised, resulting in the atoms only forming bonds to the nearest three. Another sp^2 allotrope are the fullerenes.

Carbon C_{60}-I_h was theoretically proposed by Õsawa in 1970 [9] but it was not until 1985 that Kroto et al. experimentally discovered the football shaped fullerene through laser vaporisation of graphene in a helium environment[1]. The name *Buckminsterfullerene* was chosen for C_{60}-I_h after the American architect *Richard Buckminister Fuller*, as his geodesic dome building are graph duals to fullerenes. Today the C_{60}-I_h is commonly referred to as the *Bucky ball*. Eventually the name fullerenes was chosen for the whole familly of shapes.

Fullerenes are cage-like, hollow molecules consisting only of 3-connected sp^2-hybridised carbon atoms. There are infinitely many posible fullerenes with the number of isomers growing as $O(N^9)$[3] for N carbon atoms. There exist 1812 C_{60} isomers and 132.247.999.328 C_{400} isomers, all of which have distinct chemical and physical properties. Fullerenes consists of exactly 12 pentagons and F_6 hexagons giving the fullerenes a pseudo-spherical symmetry. The formula for the number of atoms is C_{20+2F_6} for $(F_6 \geqslant 0 \wedge F_6 \neq 1)$ making C_{20} (*Dodecahedron*) the smallest fullerene, only consisting of pentagons. C_{60}-I_h is the minimal arrangement such that no two pentagons

share an edge which gives thermodynamic stability to the fullerene thereby fulfilling the Isolated Pentagon Rule[10]. Furthermore the positions of the pentagons determine the shape of the fullerene, which the following sections will shed some light upon.

2.2 TOPOLOGY

Every day we encounter different objects which all have boundaries described by different surfaces. One can think of a surface as the outer shell of a shape, e.g. the crust of the earth describes a spherical shape or the frosting covering a donut make a torus. Ideally the outer shells would have zero thickness, which would make for a sad amount of frosting on our donut.
Gauss curvature ($\mathcal{K}$) is an intrinsic property defined on all points of a surface. It describes how the surface bends and curves and whether or not it is a closed surface. The local curvature can mathematically be described as a product of the maximal and minimal curvatures in any direction through a point, See Figure 1.

$$\mathcal{K} = k_1 k_2 \tag{1}$$

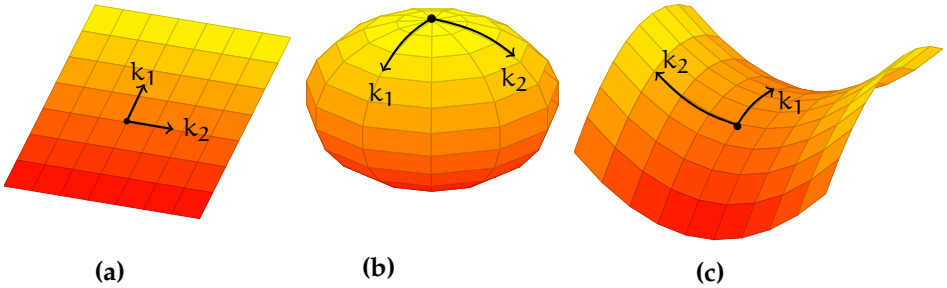


Figure 1: Examples of *Gauss* curvature around a point on a surface, (**a**) zero, (**b**) positive and (**c**) negative. k_1 and k_2 are the directions of most curvature.

Another way to think of Gauss curvature is through the Bertrand-Diquet-Puiseux theorem. It states that the Gauss curvature at a point can be geometrically interpreted as the difference between 2π and the angle of an infinitesimal circle drawn around the point. A surface such as Figure 1 **(a)** is flat everywhere and has zero Gauss curvature, since the surface lies in the same plane as the circle. If you placed the same infinitesmal circle on a point on the sphere in Figure 1 **(b)** only the point would lie in the plane of the circle, and the rest would go out of the plane, thus giving it positive curvature. In the case of the saddlepoint in Figure 1 **(c)** it would both go in and out of the plane which consequently gives negative curvature.

The total Gauss curvature dictate the *topology* of a surface through the *Gauss-Bonnet* theorem:

$$\int_S d\mathbf{s}\mathcal{K}(\mathbf{s}) = 4\pi(1-g) \tag{2}$$

Where $\mathbf{s}$ parameterizes the oriented surface S, $\mathcal{K}$ is the Gauss curvature and g is the genus of the surface (the number of holes). Again think back to the donut which has genus $g = 1$ or a salt pretzel which, in most cases, has genus $g = 3$. A surface can only be closed when the total Gauss curvature is a multiple of 4π, like the sphere in Figure 1 **(b)**.

2.3 GEOMETRY OF FULLERENES

Fullerenes are closed hollow surfaces with non-negative Gauss curvature everywhere. This is due to the fact that it only consists of exactly 12 pentagons and F_6 hexagons. To better understand this we start by considering graphenes honeycomb structure and its dual graph representation in Figure 2 **(a)**. Here the green dots outline the hexagons in graphene and the orange dots are the dual graph representation where each orange dot represent a hexagon face. A gray circle is placed around the center of the dual graph, illustrating the plane of an infinitesmal small circle in analogy to the Bertrand-Diquet-Puiseux theorem, showing that the hexagon planes have Gauss curvature zero everywhere. In Figure 2 **(b)** a triangular wedge of angle $\frac{2\pi}{6}$ is removed between atom $\{i, j\}$ and merged together in Figure 2 **(c)**. The two points $\{i, j\}$ are merged together at i' forming a cone with $\frac{2\pi}{6}$ Gauss curvature at the apex of the cone or the center point which still lies in the plane of the circle. In Figure 2 **(d)** the dual graph is shown with the matching faces illustrating the induced curvature around the pentagon.

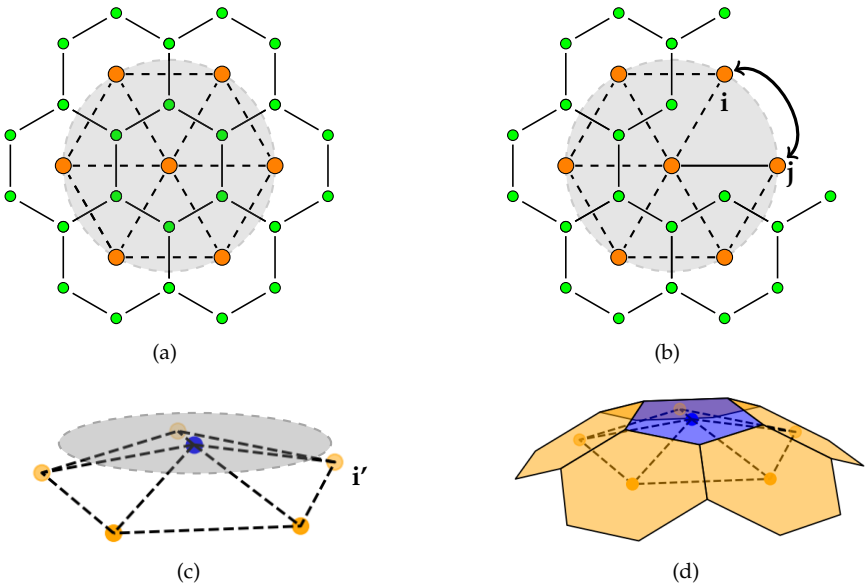
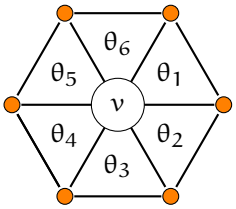


Figure 2: Illustration of induced Gauss curvature of pentagons. **(a)** graphene honeycomb lattice (green) with dual graph representation (orange). **(b)** a wedge of $\frac{2\pi}{6}$ is removed between $\{i, j\}$, **(c)** the two points $\{i, j\}$ are merged together into a new point i' inducing the curvature around the pentagon, **(d)** illustration of the faces on top of the dual graph to better understand the induced curvature.

A pentagon induces a Gauss curvature of $\frac{2\pi}{6}$ so in order to reach the 4π Gauss curvature of a sphere, 12 pentagons are needed. This can also be understood through the discrete form of the Gauss-Bonnet:

$$\sum_{v=1}^{N} \mathcal{K}_v = 4\pi(1-g) \tag{3}$$

In this setting, all curvature $\mathcal{K}_v$ is concentrated at the vertices which are the blue and orange dots of the dual graph in Figure 2 (c). The curvature around a point can easily be understood in the triangulated dual graph representation, why the curvature is the difference between 2π and the angle of a circle around the point:



$$\mathcal{K}_v = 2\pi - \sum_{i=1} \theta_i \tag{4}$$

To find the number of pentagon faces F_5 needed to close to a fullerene, we then use eq. 4 in eq. 3:

$$\sum_{v=1}^{N} (2\pi - \sum_{i=1} \theta_i) = 4\pi(1-g) \tag{5}$$

We then further divide the left hand side into two sums of each face type and remember that fullerenes have $g = 0$:

$$\sum_{n=1}^{F_5}(2\pi - \sum_{i=1}^{5}\theta_i) + \sum_{m=1}^{F_6}(2\pi - \sum_{j=1}^{6}\theta_j) = 4\pi \tag{6}$$

We set $\theta_i = \theta_j = \frac{2\pi}{6}$, and write the sums over $\{F_5, F_6\}$ out as a multiplicity of F_5 and F_6:

$$\begin{aligned} F_5(2\pi - \sum_{i=1}^{5}\frac{2\pi}{6}) + F_6 \underbrace{(2\pi - \sum_{j=1}^{6}\frac{2\pi}{6})}_{=0} &= 4\pi \\ 2\pi F_5(1 - \frac{5}{6}) &= 4\pi \\ \frac{F_5}{6} &= 2 \\ F_5 &= 12 \end{aligned} \tag{7}$$

Had we instead removed two triangular wedges from the hexagon, we would end up with a square having Gauss curvature $\frac{2\pi}{3}$, requiring only 6 to close the surface. Alternatively had we introduced another wedge it would result in a heptagon with negative $\frac{2\pi}{6}$ curvature, which would mean that another pentagon, 13 in all, would be needed to close the surface. Because fullerenes always have exactly 12 pentagons of curvature $\frac{2\pi}{6}$, with the remaining surface of Gauss curvature zero, they are not spherical but can at most achieve icosahedral symmetry[1]. For more information on fullerene topology see [2, 3].

2.4 GRAPH THEORY

Carbon has four valence electrons and therefore prefers to be part of a four covalent chemical bond system in order to fulfill the octet rule. In Fullerens, due to the sp^2 orbital hybritisation, the carbon atoms form bonds with the three closest neighbours, with two single bonds and one double bond, however we will not distinguish between single and double bonds.
Mathematically this can be characterized by a graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ with vertices $\mathcal{V}$ being a set of carbon atoms connected by pairwise unordered edges $\mathcal{E} \subseteq \{(x, y) \mid (x, y) \in \mathcal{V}^2 \wedge x \neq y\}$. We can think of the vertices $\mathcal{V}$ being the atoms and the chemical bonds would then be the edges $\mathcal{E}$.
A favorable way to represent the graph connectivity or the bonds between atoms and their neighbours is through an *Adjacency matrix* A_{ij}. The Adjacency matrix is a symmetric matrix where $A_{ij} = 1$ if vertex i and j are connected and $A_{ij} = 0$ if they are not. Since every vertex only has three

1 Actually both the C20 (Dodecahedron) and C_{60}-I_h (Buckyball) are spherical, in the sense that all their carbon atoms lie on the surface of a sphere.

edges it is often easier to have this information represented in a sparse matrix representation of size $N \times 3$ instead of an $N \times N$ matrix.

$$\underbrace{\begin{pmatrix}
0 & 1 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 \\
1 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\
0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\
0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\
0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 \\
1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\
1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0
\end{pmatrix}}_{\text{Adjacency matrix}}
\qquad
\underbrace{\begin{pmatrix}
2 & 6 & 7 \\
3 & 1 & 8 \\
4 & 2 & 9 \\
5 & 3 & 10 \\
6 & 4 & 11 \\
1 & 5 & 12 \\
1 & .. & .. \\
2 & .. & .. \\
3 & .. & .. \\
4 & .. & .. \\
5 & .. & .. \\
6 & .. & ..
\end{pmatrix}}_{\text{Sparse Adjacency matrix}}$$

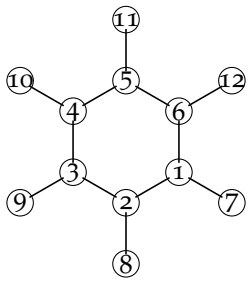


Figure 3: Simple connectivity graph. With Adjacency matrix and Sparse Adjacency matrix.

Both matrices represent the same connectivity information of the simple graph shown in Figure 3, it is therefore often preferable to work with the sparse matrix notation.

Knowing that each carbon atom have three neighbours, or each vertex is connected to three other vertices we call it a *cubic graph*, because a minimum of three edges have to be removed to form two sub graphs, it is also *three-connected*. If a graph can be mapped to a two dimensional space $\mathcal{V} \to \mathbb{R}^2$ such that vertices are assigned $[x, y]$ coordinates and can be connected by non-crossing straight lines (the edges), the graph is said to be *planar*. Fáry's theorm then affirms that the graph can be mapped onto a sphere also with non-crossing straight lines forming a polyhedral shape. However a planar graph in itself does not have a unique set of coordinates and the vertices can in a sense just be moved around.

For *three-connected planar graphs* there exists a set well defined faces $\mathcal{F}$, which allows us to represent the graph by the face information also $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \mathcal{F})$. Additionally through Steinitz's theorem the planar graph can be mapped to

the sphere as a convex polyhedra invoking adherent toplogical properties within our graph, thus making it a *Polyhedral graph*.

From the face information a dual graph $\mathcal{G}^*$ can be constructed and each vertex in $\mathcal{G}^*$ then corresponds to a face of the graph $\mathcal{G}$ and an edge in $\mathcal{G}^*$ is then joining two neighboring faces from $\mathcal{G}$. In Figure 2 (**a**) a dual graph of equilateral triangles are shown.

Knowing that fullerenes are *three-connected planar graphs* with well defined faces consisting of pentagons and hexagons that can be represented as equilateral triangles through the dual graph representation, we can obtain a rigid metric describing its surface. This essentially means that the edges of our dual graph have a well defined length or that the pentagons and hexagons are regular and also have well defined sizes. Thus by having a rigid convex polyhedra metric we can apply *Alexandrov's uniqueness* Theorem:

Alexandrov's Uniqueness Theorem 1 *Let M be a convex polyhedral metric on the sphere. Then there exists a convex polyhedron* $P \subset R^3$ *such that the boundary of* P *is isometric to M. Moreover,* P *is unique up to rigid motions.*

Which signifies that there exists one and only one embedding in 3D space, where all of our well defined properties of the graph, e.g. the edges and faces are preserved without the surface tearing or crossing and all lengths are preserved. For a more broad description of graph theory in regards to fullerenes, the reader is refered to[3].

In context to the work of this thesis it means that for each fullerene graph, of which there are infinitely many, there exists a unique geometry that preserves all of our molecular graph properties. The next section will describe how a planar graph can be obtained from the graph information, and how the 2D planar graph is embedded onto a sphere giving us an initial geometry.

2.5 PLANAR GRAPHS AND INITIAL GEOMETRIES

Planar embeddings can be drawn on a paper and give us a way to visualize the framework of the graph. Ideally we want the drawing to be simple with no crossing edges and informative.
There exists a variety of different methods for generating the planar embedding e.g. Schlegel projection and Cone-projection[3] where the 3d geometry prior is needed. A point p is placed slightly above the 3d geometry and a plane is situated below. Lines are then drawn from p through each vertex on the geometry intersecting the plane, which then yield the planar embedding as show in Figure 4 (**a**). However the Schlegel projection does not always yield correct results and will sometimes get crossing edges for non-spherical polyhedra. In the case of the Cone-projection the points are first projected upon an enveloping cone of the fullerene and later down onto the plane, the

metod is depicted in Figure 4 (**b**), the Cone-projection is often more robust with non-spherical polyhedra. Nevertheless both methods need the fullerene 3D geometry prior and even then they will not always yield correct answers. A third method *Tutte*-embedding[3], is guaranteed to always produce a planar graph for 3-connected planar graphs and no 3D geomtry is needed prior to the projection.

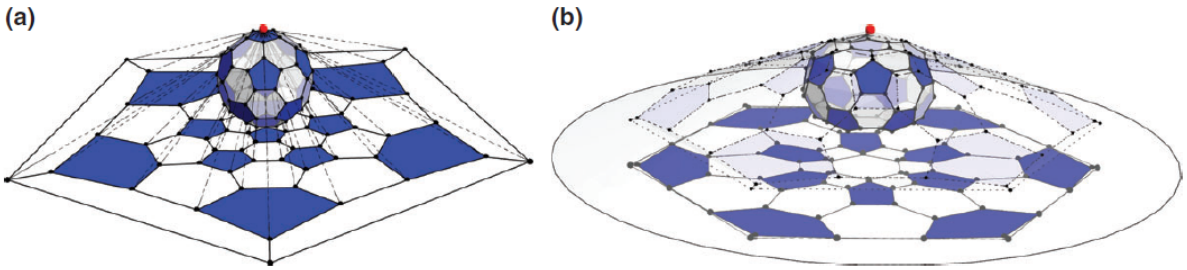


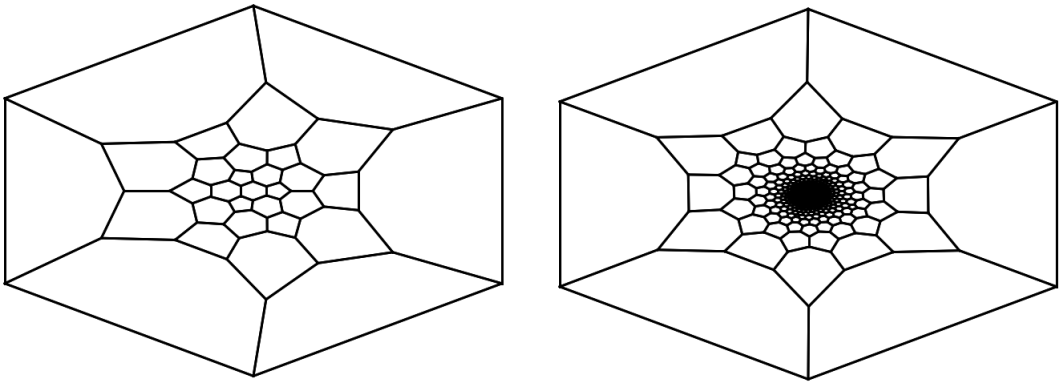
Figure 4: Figure of planar embedding of C_{60}-I_h. (**a**) Schlegel projection, (**b**) Cone projection. This Figure is borrowed from the article[3] with permission.

2.5.1 *Tutte-embedding*

In graph drawing, the Tutte-embedding of a simple 3-connected planar graph is a crossing-free straight-line embedding, where the outer face is a convex polygon and each interior vertex is the solution of a sparse linear system:

$$\sum_j \mathbf{T}_{ij}\mathbf{x}_j = \mathbf{y}_j \tag{8}$$

The interior vertices are then the barycenter solutions of the respective three neigbouring points of a vertex, which in the case of fullerenes the solution to $\mathbf{T}_{ij} = \delta_{ij} - \mathbf{A}_{ij}/3$, where $\mathbf{A}_{ij}$ is adjacency matrix and $y_{ij} = 0$ for every non-fixed vertex, so every vertex but the outer convex polygon which is fixed. However even though Tutte embedding always yields a planar graph of non-crossing edges, the solution to the sparse linear system will result in exponentially crowding of the vertices for larger system, as depicted in Figure 5 (**b**), making the planar graph hard to interpret.

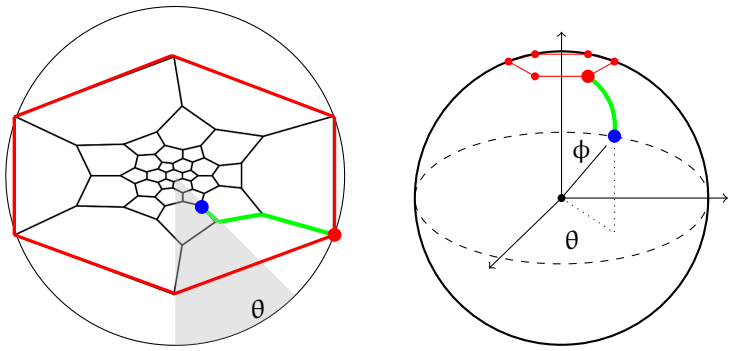


(a) (b)

Figure 5: Tutte embedding of (**a**) C_{60}-I_h and (**b**) C_{960}Ih, with exponential crowding of the faces around the center.

2.5.2 *Generating Initial 3d Geometries*

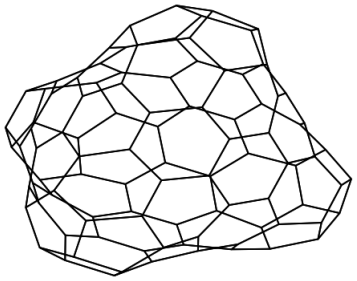
Having obtained a planar graph through the Tutte-embedding it is now possible to further embed it onto a sphere and acuire an initial geometry for a force field optimization.
From Alexandrov's theorem we know that an ideal convex geometry exists where the faces are preserved or the dual graph is made purely from equilateral triangles and can theoretically be derived from the graph. However it is not yet known how to determine the ideal 3d structure as a conformal, isometric embedding of the surface into the space. Nevertheless it is possible to use force field optimization from a crude initial geometry to obtain good results, close to the ideal molecular structure.



(a) (b)

Figure 6: Illustration of how the Tutte embedding is embedded onto the sphere. (**a**) The planar Tutte embedding with the point (red) on the enclosing pentagon and a point (blue) being a vertex in the graph. The line (green) shows the shortest path between the two points, where all edges have a fixed length and θ (grey area) shows the angle between a direction of reference and the edge (blue). (**b**) The enclosing pentagon is placed at the top of the sphere and the information from (**a**) is translated into the position of the (blue) vertex on the sphere.

The method used in the *Fullerene* program[7, 8] for obtaining initial geometries is a sort of gift wrapping of the Tutte-embedding down over the sphere. For each vertex v, we let the topological distance (minimum number of edges) from the outer face be d_v, and its coordinates in the Tutte embedding be $\mathbf{x}_v$. We define the maximum depth as D, which in the case of the Tutte embedding of C_{60}-I_h in Figure 5 (**a**) is $D = 7$. The barycenter of the embedding is then $\mathbf{x}_c$, which we will place at the pole of the sphere. We can then assign an angle $\phi(v) = \frac{(d_v+1/2)\pi}{D+1}$, which will place the vertex layers equidistantly along the ϕ-angle accordingly to the distance d_v from the outer face. The second angle $\theta(v)$ is the angle of $\mathbf{x}_v$ around $\mathbf{x}_c$. In Figure 6 an illustration of how the vertices are assigned $v \mapsto (\theta(v), \phi(v))$ is shown.
The Tutte-embedding on the sphere yields an initial geometry with non-crossing edges. Through force field optimization of the initial geometry it is then possible to obtain good 3d structures close to the physical molecules, which the next section will explain.



3

FULLERENE FORCE FIELDS

The forces acting on a physical system is the negative gradient to the systems potential energy. Because of this the force fields are defined as the potential energy of a system, where the forces are the negative gradient of the potential energy of the system. The force field optimization then aims to minimize this potential energy.
In the case of geometry optimization, it is favourable to express the potential energy as a harmonic approximation around the optimal geometry. This servers the purpose of obtaining a convex energy expression with a singular well defined minimum, which can be found deterministically contrary to more complicated physical expressions such as Lennard-Jones and Morse potentials, which can have messy singularities. Also, in the optimal energy the harmonic potential is correct up to second order of the Taylor expanssion. The potential energy at a minimum $\mathbf{X}_0$ can be approximated through the Taylor expansion:

$$E(\mathbf{X}_0 + \Delta\mathbf{X}) = E(\mathbf{X}_0) + \Delta\mathbf{X}^T \nabla E(\mathbf{X}_0) + \frac{1}{2}\Delta\mathbf{X}^T \mathbf{H} \Delta\mathbf{X} + \mathcal{O}(||\Delta\mathbf{X}||^3) \tag{9}$$

With $\mathbf{H}$ being the Hessian containing the second derivatives. We can ignore $\Delta\mathbf{X}^T \nabla E(\mathbf{X}_0) = 0$ since it will be zero at the minimum.
This leaves the energy at the minimum $E(\mathbf{X}_0)$, the distortive term $\frac{1}{2}\Delta\mathbf{X}^T \mathbf{H} \Delta\mathbf{X}$ and higher Taylor orders. The expression for the energy at the minimum then takes the form:

$$E(\mathbf{X}_0 + \Delta\mathbf{X}) = E(\mathbf{X}_0) + \frac{1}{2}\Delta\mathbf{X}^T \mathbf{H} \Delta\mathbf{X} + \mathcal{O}(||\Delta\mathbf{X}||^3) \tag{10}$$

This approximation is correct close to the minimum. However working with molecules, the cartesian coordinates $\mathbf{X} = \{x, y, z\}$ of atoms are not intuitive. In order to better understand the geometry the spatial coordiantes of $\mathbf{X}$ is expressed through intrinsic properties of inter-nuclear relations between atoms e.g. bond distance between atoms, bond-bond angles, and

dihedral angles between neighbouring faces. In the case of bond distance between carbon atoms, the potential is described by the distance R and is then optimized towards an equlibrium length instead of some specific cartesian coordinates $[x_a, y_a, z_a, x_b, y_b, z_b]$. The change of parameterization also servers the purpose of creating a general expression for all sorts of fullerene geometries. A Taylor expanssion in the cartesian coordiantes at the minimum would give an expression for a specific fullerene with $N \times 3$ different force constants restricted to the specific coordinates, and it would not be interchangeable with other fullerenes. Therefore an intuitive regular parameterization which can be described by few intutive parameters e.g. carbon atom bond lengths or the molecule bond angles, which we have a good chemical understanding of, are needed.
If we were to perturb our coordinates $\mathbf{X}$ in a single direction x_i, the right handside of Eq. 10 would take the form:

$$E(\mathbf{X}_0 + x_i) = E(\mathbf{X}_0) + \frac{1}{2}\frac{\partial^2 E(\mathbf{X}_0)}{\partial x_i^2}(x_i - x_i^0)^2 \tag{11}$$

This expression, which only consists of a single variable (x_i) is frequently refered to as *Hooke's law* or *Hooke's potential*, typically with $\frac{\partial^2 E(\mathbf{X}_0)}{\partial x_i^2} = k$ describing a vibrational frequency constant.
The intrinsic properties can be formulated in the same manner, since bond distance between two atoms also is a single variable. It is then possible to construct a potential for our force field, consisting of multiple *Hooke's potential* terms of each intrinsic property. The next section will describe the inter-nuclear properties, and how previous force fields have been constructed to describe the geometry of fullerenes.

3.1 FULLERENE FORCE FIELD

The first force field constructed for fullerenes was by Wu et al.[6] in 1987. It was made specifically for the C_{60}-I_h and used harmonic force terms in the form:

$$\begin{aligned} E_{Wu} = &\frac{k_p}{2}\sum_{i_p}^{p-e}(R_{i_p} - R_p)^2 + \frac{k_h}{2}\sum_{i_h}^{h-e}(R_{i_h} - R_h)^2 + \\ &\frac{f_p}{2}\sum_{j_p}^{60}(\theta_{i_p} - \theta_p)^2 + \frac{f_h}{2}\sum_{j_h}^{3N-60}(\theta_{i_h} - \theta_h)^2 \end{aligned} \tag{12}$$

Where $p-e$ and $p-h$ describes the pentagon and hexagon edges. In this simple force field only the bond distances and internal face angles of pentagons and hexagons were used. The C-C bond force constants $k_p = 11$ and $k_h = 10$ are given in (105 dyn cm) and $R_p = 1.54$Å and $R_h = 1.41$Å are the corresponding pentagon and hexagon bond distances. k_p and k_h

are the force constants for the two different internal modes in a pentagon and hexagon, both in units of ($10^5 \frac{dyn}{cm}$) with the corresponding bond angles $\theta_p = 108°$ and $\theta_h = 120°$. Since the first fullerene force field by Wu et. al. was made, a variety of changes and updates have been made, namely introducing dihedral angles and new harmonic force constants. The potential energy expression, which this thesis will focus on, is the one described in Wirz et al., the force field includes four dihedral angles with the center atom being connected to (0, 1, 2 or 3 pentagons) as shown in Figure 7, three types of bond distances e.g. bonds between (0, 1, or 2 pentagons) and two angle bending terms. The Energy expression is given by:

$$\begin{aligned} E_{Wirz} = \quad & \frac{f_{pp}}{2} \sum_{i_{pp}}^{pp-e} (R_{i_{pp}} - R_{pp})^2 \frac{f_{ph}}{2} \sum_{i_{ph}}^{ph-e} (R_{i_{ph}} - R_{ph})^2 \\ & + \frac{f_{hh}}{2} \sum_{i_{hh}}^{hh-e} (R_{i_{hh}} - R_{hh})^2 + \frac{f_p}{2} \sum_{i_p}^{60} (\theta_{i_p} - \theta_p)^2 \\ & + \frac{f_h}{2} \sum_{i_h}^{3N-60} (\theta_{i_h} - \theta_h)^2 + \frac{f_{ppp}}{2} \sum_{i_{ppp}}^{ppp-v} (\beta_{i_{ppp}} - \beta_{ppp})^2 \\ & + \frac{f_{pph}}{2} \sum_{i_{pph}}^{pph-v} (\beta_{i_{pph}} - \beta_{pph})^2 + \frac{f_{phh}}{2} \sum_{i_{phh}}^{phh-v} (\beta_{i_{phh}} - \beta_{phh})^2 \\ & + \frac{f_{hhh}}{2} \sum_{i_{hhh}}^{hhh-v} (\beta_{i_{hhh}} - \beta_{hhh})^2 \end{aligned} \tag{13}$$

Where pp-e, hp-e and hh-e denote the number of edges adjacent to (2, 1 or 0 pentagons respectively). ppp-v, hpp-v, hhp-v and hhh-v are the number of vertices centered in (3, 2, 1 or 0 pentagon arrangements respectively). The force field parameters are set as follows in Table 1:

R_{pp}	R_{ph}	R_{hh}	θ_p	θ_h	β_{ppp}	β_{pph}	β_{phh}	β_{hhh}
1.479 Å	1.458 Å	1.401 Å	108°	120°	37.377°	(29.202°, 35.285°)	(18.774°, 23.943°)	0°

Table 1: Table of force field parameters in E_{Wirz}. In the dihedral angle β_{pph}, β_{phh} the first value (29.202°, 19.774°) refer to the angle of the pentagon faces onto the base plane of atom b-c-d illustrated in Figure 7. The second values (35.285°, 23.943°) is that of the hexagons faces onto the base plane.

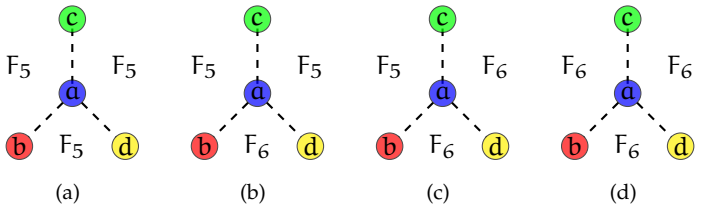


Figure 7: Figure of different dihedral settings. The force-field parameters are, starting from b, in clockwise order: **(a)** $(37.377°, 37.377°, 37.377°)$, **(b)** $(29.202°, 29.202°, 35.285°)$, **(c)** $(19.774°, 19.774°, 23.943°)$, **(d)** $(0°, 0°, 0°)$

The force constants in the *Fullerene* program have been obtained through a least squares fit to B3LYP frequencies of the vibrational spectra of $C_{50} - C_1(193)$, C_{60}-I_h and $C_{70} - D_{5h}(8149)$ (numbering of the isomers stems from lexicographically ordered face-spiral pentagon indices, see [10]). The harmonic force constants are shown in Table 2.

Force constants [N/m]

f_{pp}	f_{ph}	f_{hh}	f_{p}	f_{h}	f_{ppp}	f_{pph}	f_{phh}	f_{hhh}
260	390	450	100	100	35	65	85	270

Table 2: Table of force constants in E_{EHFF}, obtained through least squares fit to B3LYP frequencies of the vibrational spectra of $C_{50} - C_1$, C_{60}-I_h and $C_{70} - D_{5h}$.

It is possible to divide Equation 13 into three general harmonic force terms:

$$E_{Wirz} = E_{str} + E_{ang} + E_{dih} \tag{14}$$

E_{str} is the bond stretching energy, E_{ang} is the bond-bond angular bending energy, and E_{dih} is the neighbouring face dihedral energy.
In order to use a geometry force field method we need to find the gradient of each term in Eq. 14.

3.2 CALCULATING AND SIMPLIFYING DERIVATIVES

The main focus governing this thesis have been simplifying and making an already exsisting code more user-friendly, transparent, correct, and to create a foundation from which parallization could easily be implemented.

The force field implementation in the *Fullerene* program is written in *Fortran* with thousands upon thousands lines of code which are hard to navigate and maintain. Therefore it has been preferable to start over and redo a lot of the exsisting work. The calculations and derivatives of this thesis have been done using vector notation instead of single coordinates (x, y, z) representation, which makes the code easier to maintain, expand, trouble shoot, and implement parallisation, which will be discussed further in section 3.4.
Furthermore features like distance and angles is easily understood through

vector norms and dot products of vectors. Additionally a correction to the calculated dihedral gradient has been implemented, which gives an correct description of the energy gradient of Eq.13.

In the following sections the gradient derivation of each force term in Eq. 14 are presented, for a more meticulous derivation of each gradient term the reader is refered to Appendix A.
Since the backbone of this thesis has been to implement and write force field derivatives, into readable and intuitive *Python* code, code snippets from the software will be presented together with the respective calculated derivatives.

3.2.1 *Bond stretching*

The bond stretching energy E_{str} is the energy required to compress and stretch two bonded atoms **a** and **b** away from their chemical equilibrium in the direction of $\mathbf{ab}$. We can describe the system using *Hooke's potential* as an approximation around equilibrium.

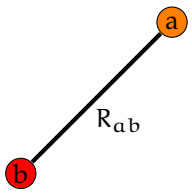


Figure 8: Illustration of two bonded carbon atoms (**a**,**b**).

$$E_{str} = \frac{k_{str}}{2}(R_{ab} - R_0)^2 \tag{15}$$

where k_{str} is the bond constant, R_{ab} the distance between **a** and **b** and R_0 is the distance between **a** and **b** at chemical equilibrium. In order to find the optimal distance, we calculate the gradient $\nabla_{\mathbf{a}} E_{str}$ with respect to the movement of **a**

$$\nabla_{\mathbf{a}} E_{str} = \frac{k_{str}}{2}\nabla_{\mathbf{a}}(\|\mathbf{ab}\| - R_0)^2$$

Here we have written R_{ab} as $\|\mathbf{ab}\| = \sqrt{(b_x - a_x)^2 + (b_y - a_y)^2 + (b_z - a_z)^2}$.
We then take the derivative of the function and get:

$$\nabla_{\mathbf{a}} E_{str} = k_{str}(\|\mathbf{ab}\| - R_0)\nabla_{\mathbf{a}}\|\mathbf{ab}\| \tag{16}$$

The gradient of the vector norm $\|\mathbf{ab}\|$ is the negative unitvector $\widehat{\mathbf{ab}}$ of the direction of $\mathbf{ab}$ which leaves us with the following expression as the gradient contribution from the stretching energy

$$\nabla_{\mathbf{a}} E_{str} = -k_{str}\left(\|\mathbf{ab}\| - R_0\right)\widehat{\mathbf{ab}} \tag{17}$$

With code implementation in the form

```
d_R = -np.sum( (fc * (R_ab - R0))[...,NA] * ab_hat , axis=1)
```

Where the shapes of each object is

fc	$(N \times neighbours)$	$(N_{atoms} \times 3)$
R_ab	$(N_{atoms} \times neighbours)$	$(N_{atoms} \times 3)$
R0	$(N_{atoms} \times neighbours)$	$(N_{atoms} \times 3)$
ab_hat	$(N_{atoms} \times neighbours \times (x,y,z))$	$(N_{atoms} \times 3 \times 3)$

Here fc is the force constants, R0 are the lengths at chemical equilibrium, both calculated respectively to the atom-neighbour adjacent faces. R_ab consists of the calculated bond distances, and ab_hat contains the the unit vectors pointing from the center atoms to the neighbours. Because d_R is a general bond stretching gradient description, it can be performed on all bonds. Thus, the bond strethcing gradient is computed simultaneously for all bonds, where the [...,NA], is creating a newaxis which can properly scale the each neighbouring gradient contribution. Lastly the neighbouring contributions are summed over the neighbour axis, giving a bond gradient expression at each atom with the $(N_{atoms} \times (x,y,z))$.

3.2.2 *Angular bending*

The angular energy E_{ang} is the energy required to bend a three atom system **b-a-c** away from its equilibrium angle, see Figure 9. Again the harmonic approximation is used around the equilibrium and yields the following expression for the angular contribution:

$$E_{bend}(\theta) = \frac{k_{ang}}{2}(\cos(\theta_a) - \cos(\theta_0))^2 \quad (18)$$

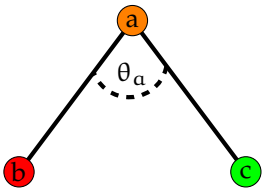


Figure 9: Illustration of angle θ between three cabon atoms (**a,b,c**).

where k_{ang} is the spring constant, $\cos(\theta_a)$ is the dot product[1] between the vectors $\widehat{\mathbf{ab}}$ and $\widehat{\mathbf{ac}}$ and $\cos(\theta_0)$ is the cosine of the equilibrium angle. We calculate the gradient $\nabla_{\mathbf{a}} E_{ang}$ with respect to the movement of **a**:

$$\nabla_{\mathbf{a}}\cos(\theta_a) = \frac{k_{ang}}{2}\nabla_{\mathbf{a}}(\underbrace{\widehat{\mathbf{ab}} \cdot \widehat{\mathbf{ac}}}_{\cos(\theta_a)} - \cos(\theta_0))^2 \quad (19)$$

We then use the Product rule:

$$\nabla_{\mathbf{a}}\cos(\theta_a) = k_{ang}(\cos(\theta_a) - \cos(\theta_0))(\nabla_{\mathbf{a}}\widehat{\mathbf{ab}}) \cdot \widehat{\mathbf{ac}} + \widehat{\mathbf{ab}} \cdot (\nabla_{\mathbf{a}}\widehat{\mathbf{ac}}) \quad (20)$$

1 This calculation is only precise up to the sign, meaning we will not know whether or not we are above or bellow π, but since our pentagon and hexagon faces have optimal energy in the convex case we can assume, given small enough iteration steps, that our cosine should be between $[0, \pi]$.

We now have two gradient terms $\nabla_{\mathbf{a}}\widehat{\mathbf{ab}}$ and $\nabla_{\mathbf{a}}\widehat{\mathbf{ac}}$. Here we use the vector Quotient rule (QR) and get the following expression.

$$\nabla_{\mathbf{a}}\cos(\theta_a) = k_{ang}(\cos(\theta_a) - \cos(\theta_0)) \left(\frac{1}{\|\mathbf{ab}\|}(\widehat{\mathbf{ab}}\cos(\theta_a) - \widehat{\mathbf{ac}}) + \frac{1}{\|\mathbf{ac}\|}(\widehat{\mathbf{ac}}\cos(\theta_a) - \widehat{\mathbf{ab}})\right) \tag{21}$$

With the corresponding code

```
grad  = (cos_a[...,NA] * ab_hat - ac_hat)/ R_ab +
        (cos_a[...,NA] * ac_hat - ab_hat)/ R_ac

d_ang = np.sum( (fc * (cos_a - ang0))[...,NA] * grad, axis=1);
```

In order to get ac_hat the ab_hat array is rolled $\texttt{roll}([N_{atoms} \times (\widehat{\mathbf{ab}}, \widehat{\mathbf{ac}}, \widehat{\mathbf{ad}})]) \rightarrow [N_{atoms} \times (\widehat{\mathbf{ac}}, \widehat{\mathbf{ad}}, \widehat{\mathbf{ab}})]$. Which makes it possible to use simple vector operations.

3.2.3 *Dihedral angle*

The dihedral energy E_{dih} is the energy required to distort a center atom **a** of a 4 atom system **a-b-c-d**, away from the optimal distance from the **b-c-d** plane, see Figure 10. The dihedral energy is as follows:

$$E_{dih} = \frac{k_{dih}}{2}(\cos(\beta) - \cos(\beta_0))^2 \tag{22}$$

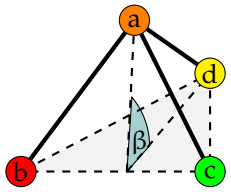


Figure 10: Illustration of dihedral angle β of a four carbon atom system (a,b,c,d).

where k_{dih} is the spring constant, cos(β) is the angle of the (**a**, **b**, and **c**) plane down onto a base plane formed by (**b**, **c**, and **d**) and β_0 is the angle at equilibrium. In order to find the optimal distance, we calculate the gradient $\nabla_{\mathbf{a}}E_{dih}$ with respect to the movement of **a**:

$$\nabla_{\mathbf{a}}E_{dih} = \frac{k_{dih}}{2}\nabla_{\mathbf{a}}(\underbrace{\hat{\mathbf{n}}_{abc}\cdot\hat{\mathbf{n}}_{bcd}}_{\cos(\beta)} - \cos(\beta_0))^2 \tag{23}$$

Once again we use the Product rule (PR)

$$\nabla_{\mathbf{a}}E_{dih} = k_{dih}(\cos(\beta) - \cos(\beta_0))(\nabla_{\mathbf{a}}\hat{\mathbf{n}}_{abc})\cdot\hat{\mathbf{n}}_{bcd} + \hat{\mathbf{n}}_{abc}\cdot(\nabla_{\mathbf{a}}\hat{\mathbf{n}}_{bcd}) \tag{24}$$

From this equation we see that the $\hat{\mathbf{n}}_{bcd} = \frac{\widehat{\mathbf{cb}}\times\widehat{\mathbf{cd}}}{\sqrt{1-(\widehat{\mathbf{cb}}\cdot\widehat{\mathbf{cd}})^2}}$ vector is independent of **a**, which mean that the second term $\hat{\mathbf{n}}_{abc}\cdot(\nabla_{\mathbf{a}}\hat{\mathbf{n}}_{bcd}) = 0$. We then

write out $\hat{\mathbf{n}}_{abc} = \frac{\widehat{\mathbf{ba}} \times \widehat{\mathbf{bc}}}{\sqrt{1-(\widehat{\mathbf{ba}} \cdot \widehat{\mathbf{bc}})^2}}$ and use a combination of the quotient rule and the product rule with respect to cross products.

$$\begin{aligned}
\nabla_{\mathbf{a}} E_{dih} &= k_{dih}(\cos(\beta) - \cos(\beta_0)) \\
&\left(\frac{(\nabla_{\mathbf{a}}(\widehat{\mathbf{ba}} \times \widehat{\mathbf{bc}}))\sqrt{1-(\widehat{\mathbf{ba}} \cdot \widehat{\mathbf{bc}})^2} + (\nabla_{\mathbf{a}}\sqrt{1-(\widehat{\mathbf{ba}} \cdot \widehat{\mathbf{bc}})^2})(\widehat{\mathbf{ba}} \times \widehat{\mathbf{bc}})}{(\sqrt{1-(\widehat{\mathbf{ba}} \cdot \widehat{\mathbf{bc}})^2})^2} \right) \cdot \hat{\mathbf{n}}_{bcd} \\
&= k_{dih}(\cos(\beta) - \cos(\beta_0)) \\
&\left(\frac{\overbrace{(\nabla_{\mathbf{a}}(\widehat{\mathbf{ba}} \times \widehat{\mathbf{bc}}))}^{F} \sin(\theta_{abc}) + \overbrace{(\nabla_{\mathbf{a}}\sqrt{1-(\widehat{\mathbf{ba}} \cdot \widehat{\mathbf{bc}})^2})}^{G}(\hat{\mathbf{n}}_{abc}\sin(\theta_{abc}))}{\sin(\theta_{abc})^2} \right) \cdot \hat{\mathbf{n}}_{bcd}
\end{aligned} \tag{25}$$

In order to calculate the derivative we find the gradient of F and G and add them back into Equation 25.

$$\begin{aligned}
F &= \nabla_{\mathbf{a}}(\widehat{\mathbf{ba}} \times \widehat{\mathbf{bc}}) \qquad \text{(product rule (PR))} \\
&= (\nabla_{\mathbf{a}}\widehat{\mathbf{ba}}) \times \widehat{\mathbf{bc}} + \underbrace{\widehat{\mathbf{ba}} \times (\nabla_{\mathbf{a}}\widehat{\mathbf{bc}})}_{=0} \qquad (\widehat{\mathbf{bc}} \text{ independent of } \mathbf{a}) \\
&= \left(\frac{1}{\|\mathbf{ba}\|}(\mathbf{I} \times \widehat{\mathbf{bc}} - \widehat{\mathbf{ba}} \otimes \Big(\underbrace{\widehat{\mathbf{ba}} \times \widehat{\mathbf{bc}}}_{\hat{\mathbf{n}}_{abc}\sin(\theta_{abc})} \Big)\right.
\end{aligned}$$

$$\begin{aligned}
G &= \sqrt{1-(\widehat{\mathbf{ba}} \cdot \widehat{\mathbf{bc}})^2} \\
&= \frac{1}{2\sin(\theta_{abc})}(\nabla_{\mathbf{a}})(1-(\widehat{\mathbf{ba}} \cdot \widehat{\mathbf{bc}})^2) \\
&= -\cot(\theta_{abc})\nabla_{\mathbf{a}}(\widehat{\mathbf{ba}} \cdot \widehat{\mathbf{bc}}) \qquad \text{(product rule (PR))} \\
&= -\cot(\theta_{abc})\left((\nabla_{\mathbf{a}}\widehat{\mathbf{ba}}) \cdot \widehat{\mathbf{bc}} + \underbrace{\widehat{\mathbf{ba}} \cdot (\nabla_{\mathbf{a}}\widehat{\mathbf{bc}})}_{=0}\right) \qquad (\widehat{\mathbf{bc}} \text{ independent of } \mathbf{a}) \\
&= \frac{-\cot(\theta_{abc})}{\|\mathbf{ba}\|}\left(\widehat{\mathbf{bc}} - \widehat{\mathbf{ba}}\cos(\theta_{abc})\right)
\end{aligned}$$

We insert F and G back into Equation 25 and dot with $\hat{\mathbf{n}}_{bcd}$:

$$\begin{aligned}
\nabla_{\mathbf{a}} E_{dih} &= k_{dih}(\cos(\beta) - \cos(\beta_0)) \\
&\left(\underbrace{\frac{\widehat{\mathbf{bc}} \times \hat{\mathbf{n}}_{bcd}}{\|\mathbf{ba}\|\sin(\theta_{abc})} - \frac{\widehat{\mathbf{ba}}\cos(\beta)}{\|\mathbf{ba}\|}}_{F} + \underbrace{\frac{\cot(\theta_{abc})\cos(\beta)}{\|\mathbf{ba}\|\sin(\theta_{abc})}(\widehat{\mathbf{bc}} - \widehat{\mathbf{ba}}\cos(\theta_{abc}))}_{G} \right)
\end{aligned} \tag{26}$$

With code implementation in the form

```
F     = np.cross(bc_hat, nbcd)/(Rba * sin_b) - (ba_hat *cos_beta[...,NA])/Rba
G     = (cot_b * cos_beta[...,NA]/(sin_b * Rba)) * (bc_hat - ba_hat * cos_b)
grad  = F + G

d_dih = np.sum((fc*(cos_beta-dih0)[...,NA]) * grad, axis=1)
```

These are the gradient contributions of the inner parts, as shown in Figure 11, which are already implemented in the *Fullerene* program. Furthermore the *Fullerene* force field by Wirz et al.[7, 8] also have outer angles, which will be described in the next section. However a thing which the Wirz et al. force field fail to include are the outer dihedral terms.
By expanding the gradient to also take outer dihedral contributions into consideration we obtain a gradient which correctly describe the energy function. So far Wirz et al. force field have still gotten results close to ab inito methods, but hopefully by introducing outer dihedral gradient terms in our force field optimization we can obtain even better results. The derivatives of the outer parts are introduced in the next section.

3.2.4 *Outer contributions*

What the outer parts have in common is that the movement of **a** now happens at the edge of the connections. As a result we cannot necessarily use the same force field parameters for the outer dihedrals. The outer angles are still within the same planes as those circumscribing **a**, however for the dihedrals a new face is introduced. The arrangement of faces governing the equilibrium parameters are then computed with respect to the faces around **b**, **c** and **d**.

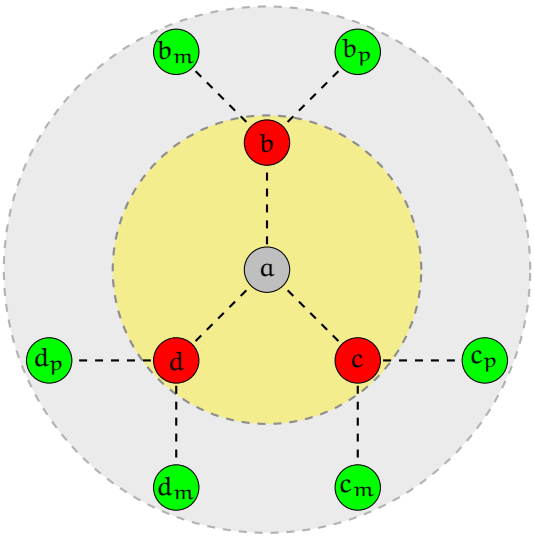


Figure 11: Figure of the connectivity around the atom **a**. The yellow circle is the inner connectivity. The gray circle is the outer connectivity.

3.2.5 *Outer angle*

One of the outer contributions is the *outer angle*, which are shown in Figure 12. When moving **a** the angles $(\theta_{bab_p}, \theta_{bab_m})$ changes. In order to get a more correct gradient describing the movement of **a** we need to include $\nabla_a \cos(\theta_{bab_p})$ and $\nabla_a \cos(\theta_{bab_p})$.

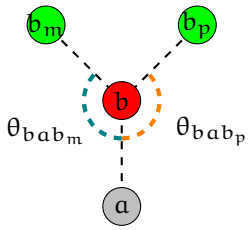


Figure 12: Illustration of outer angles of a 4 atoms system (a,b,bp,bc).

$$\nabla_a \cos(\theta_{bab_p}) = \nabla_a(\widehat{ba} \cdot \widehat{bb}_p) \tag{27}$$

We start by using the product rule

$$\begin{aligned} \nabla_a \cos(\theta_{bab_p}) =& (\nabla_a \widehat{ba}) \cdot \widehat{bb}_p + \underbrace{\widehat{ba} \cdot (\nabla_a \widehat{bb}_p)}_{=0} \\ =& \frac{1}{\|ba\|}(\widehat{bb}_p - \widehat{ba}\cos(\theta_{bab_p})) \end{aligned} \tag{28}$$

The same approach is used on $\nabla_a \cos(\theta_{bab_p})$:

$$\nabla_a \cos(\theta_{bab_m}) = \nabla_a(\widehat{ba} \cdot \widehat{bb}_m) \tag{29}$$

Using the product rule we get

$$\begin{aligned} \nabla_a \cos(\theta_{bab_m}) =& (\nabla_a \widehat{ba}) \cdot \widehat{bb}_m + \underbrace{\widehat{ba} \cdot (\nabla_a \widehat{bb}_m)}_{=0} \\ =& \frac{1}{\|ba\|}(\widehat{bb}_m - \widehat{ba}\cos(\theta_{bab_m})) \end{aligned} \tag{30}$$

Both Equation 28 and 30 are multiplied with their respective energy term $k_{bp_{ang}}(\cos(\theta_{bab_p}) - \cos(\theta_{bp_0}))$ or $k_{bm_{ang}}(\cos(\theta_{bab_m}) - \cos(\theta_{bm_0}))$.

```
grad      = (b_out_hat - ab_hat) * (cos_a/R_ab)[...,NA]
d_out_ang = np.sum( (fc * ( cos_a - ang0))[...,NA] * grad, axis=1)
```

Here b_out_hat can both take the value of $\widehat{bb}_p$ or $\widehat{bb}_m$. The reader is referred to Appendix A for a thorough calculation of the outer angles.

3.2.6 *Outer dihedral angle*

When perturbing **a** in the outer dihedrals it will change three of the dihedral angles at each neighbour. Therefore it is necessary to include the gradient contribution from each neighbouring outer dihedral into the general dihedral gradient. In Figure 13 the outer dihedrals are shown around the neighbours **b**. Since **a** occurs differently in dihedral face (bam, bmp and bpa) it also changes each dihedral angle differently. It is necessary to calculate each

gradient separately. In order to get a more correct gradient, we need to include the terms:

$$\left(\nabla_a(\hat{n}_{bam} \cdot \hat{n}_{amp}) \,,\quad \nabla_a(\hat{n}_{bmp} \cdot \hat{n}_{mpa}) \,,\quad \nabla_a(\hat{n}_{bpa} \cdot \hat{n}_{pam}) \right)$$

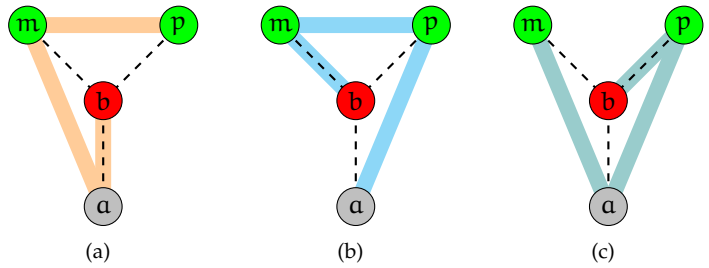


Figure 13: The three different outer dihedral planes. (**a**) the bam, (**b**) the bmp, and (**c**) the bpa.

The mathematical approach is similar to that of the inner dihedrals. detailed calculations can be found in Appendix A, the results are as follows:

$$\begin{aligned}\nabla_a(\hat{n}_{bab_m} \cdot \hat{n}_{ab_mb_p}) = & \underbrace{\frac{\widehat{ab}\cos(\beta)}{\|ab\|} - \frac{\widehat{am} \times \hat{n}_{amp}}{\|ab\|\sin\theta_a} + \frac{\widehat{am}\cos(\beta)}{\|am\|} - \frac{\hat{n}_{amp} \times \widehat{ab}}{\|am\|\sin(\theta_a)}}_{F} \\ & + \underbrace{\frac{\cot(\theta_a)\cos(\beta)}{\sin(\theta_a)}\left(\frac{\widehat{ab}\cos(\theta_a)}{\|ab\|} - \frac{\widehat{am}}{\|ab\|} + \frac{\widehat{am}\cos(\theta_a)}{\|am\|} - \frac{\widehat{ab}}{\|am\|}\right)}_{G} \\ & + \underbrace{\frac{\widehat{mp} \times \hat{n}_{bam}}{\|ma\|\sin(\theta_m)} - \frac{\widehat{ma}\cos(\beta)}{\|ma\|} + \frac{\cot(\theta_m)\cos(\beta)}{\|ma\|\sin(\theta_m)}(\widehat{mp} - \widehat{ma}\cos(\theta_m))}_{P}\end{aligned} \tag{31}$$

```
F   = (ab_hat*cos_beta[...,NA] / Rab) - (np.cross(am_hat,namp_hat)/(Rab*sin_a)) +
      (am_hat*cos_beta[...,NA]/Ram) - (np.cross(namp_hat,ab_hat)/(Ram*sin_a))

G   = (cot_a*cos_beta[...,NA]/sin_a)*( (ab_hat*cos_a/Rab) - (am_hat/Rab) +
      (am_hat*cos_a/Ram) - (ab_hat/Ram) )

P   = (np.cross(mp_hat,nbam_hat)/(Rma*sin_m)) -
      (ma_hat*cos_beta[...,NA]/Rma) +
      (cot_m*cos_beta/sin_m)*(  (mp_hat/Rma) - (ma_hat*cos_m/Rma) )

grad  = F + G + P
d_dih_a = np.sum(( fc* (cos_beta-dih0_a))[...,NA] * grad,axis=1)
```

$$\nabla_a(\hat{\mathbf{n}}_{bb_mb_p} \cdot \hat{\mathbf{n}}_{ab_mb_p}) = \frac{\hat{\mathbf{n}}_{bmp} \times \widehat{\mathbf{pm}}}{\|\mathbf{pa}\|\sin(\theta_p)} - \frac{\widehat{\mathbf{pa}}\cos(\beta)}{\|\mathbf{pa}\|} + \frac{\cot(\theta_p)\cos(\beta)}{\|\mathbf{pa}\|\sin(\sin(\theta_p))}(\widehat{\mathbf{pm}} - \widehat{\mathbf{pa}}\cos(\theta_p)) \tag{32}$$

```
grad  = (np.cross(nbmp_hat,pm_hat)/(Rpa*sin_p)) - (pa_hat*cos_beta[...,NA]/Rpa) +
        (cot_p*cos_beta[...,NA]/(sin_p*Rpa))*(pm_hat - pa_hat*cos_p)

d_dih_m = np.sum(( fc * (cos_beta-dih0_m))[...,NA] * grad,axis=1)
```

$$\begin{aligned}\nabla_a(\hat{\mathbf{n}}_{bb_pa} \cdot \hat{\mathbf{n}}_{ab_mb_p}) = &\underbrace{\frac{\hat{\mathbf{n}}_{pam} \times \widehat{\mathbf{pb}}}{\|\mathbf{pa}\|\sin(\theta_p)} - \frac{\widehat{\mathbf{pa}}\cos(\beta)}{\|\mathbf{pa}\|} + \frac{\cot(\theta_p)\cos(\beta)}{\|\mathbf{pa}\|\sin(\theta_p)}(\widehat{\mathbf{pb}} - \widehat{\mathbf{pa}}\cos(\theta_p))}_{F} \\ &+ \underbrace{\frac{\widehat{\mathbf{ap}}\cos(\beta)}{\|\mathbf{ap}\|} - \frac{\widehat{\mathbf{am}} \times \hat{\mathbf{n}}_{bpa}}{\|\mathbf{ap}\|\sin(\theta_a)} + \frac{\widehat{\mathbf{am}}\cos(\beta)}{\|\mathbf{am}\|} - \frac{\hat{\mathbf{n}}_{bpa} \times \widehat{\mathbf{ap}}}{\|\mathbf{am}\|\sin(\theta_a)}}_{G} \\ &+ \underbrace{\frac{\cot(\theta_a)\cos(\beta)}{\sin(\theta_a)}(\frac{\widehat{\mathbf{ap}}\cos(\theta_a)}{\|\mathbf{ap}\|} - \frac{\widehat{\mathbf{am}}}{\|\mathbf{ap}\|} + \frac{\widehat{\mathbf{am}}\cos(\theta_a)}{\|\mathbf{am}\|} - \frac{\widehat{\mathbf{ap}}}{\|\mathbf{am}\|})}_{P}\end{aligned} \tag{33}$$

```
F = (np.cross(npam_hat,pb_hat)/(Rpa*sin_p)) - (pa_hat*cos_beta[...,NA]/Rpa) +
    (cot_p*cos_beta[...,NA]/(Rpa*sin_p))*(pb_hat - pa_hat*cos_p)

G = (ap_hat * cos_beta[...,NA] / Rap) - (np.cross(am_hat,nbpa_hat)/(Rap*sin_a)) +
    (am_hat*cos_beta[...,NA]/Ram) - (np.cross(nbpa_hat,ap_hat)/(Ram*sin_a))

P = (cot_a*cos_beta[...,NA]/sin_a) * ((ap_hat*cos_a/Rap) - (am_hat/Rap) +
    (am_hat*cos_a/Ram) - (ap_hat/Ram))

grad = F + G + P

d_dih_p = np.sum((fc*(cos_beta-dih0_p))[...,NA] * grad,axis=1)
```

Again each part is multiplied with $k(\cos(\beta) - \cos(\beta_{out}))$ with the parameters defined accordingly to the surrounding faces.

3.3 PHYSICAL PROPERTIES AND SECOND DERIVATIVES

With the long-term desire to navigate the huge fullerene isomer space on the hunt for fullerenes with specific energy properties, higher order derivatives of the geometry needs to be calculated.
This section is an display of what might be possible given a fully calculated second derivative of the fullerene geometry. From the book [11] a comprehensive list is displayed of energy properties which can be derived from intrinsic molecule properties such as

- External electric field (**E**)
- External magnetic field (**B**)
- Electric field (**F**)
- Change in the nuclear geometry (**R**)

From surface density functional theory calculations it would be possible to obtain molecular properties such as (**E**,**B** and **F**).
Through the relation:

$$\text{Property} \propto \frac{\partial^{n_F+n_B+n_R}\mathbf{E}}{\partial\mathbf{F}^{n_F}\partial\mathbf{B}^{n_B}\partial\mathbf{R}^{n_R}} \tag{34}$$

Different properties with can be calculated from derivatives of a certain order of the intrinsic molecular properties.

To name a few geometry dependent:

n_F	n_B	n_R	**Properties**
0	0	1	*Molecular (nuclear) gradient*
0	0	2	*Harmonic vibrational frequencies*
1	0	1	*Infrared absorbtion*
2	0	1	*Raman intensity*
1	1	1	*Vibrational circular dichroism*
1	0	2	*Infrared intensities for overtone and combination bands*

These are but a few energy properties which can be derived from the fullerene given a good understanding and a accurate description of the intrinsic molecular properties.

3.4 CPU AND GPU PARALLELIZATION

Since all calculations within the force field implemented in this thesis are entirely vectorized and data-parallel, they can be done in parallel. In Figure 14, the fullerene isomer space with a single fullerene isomer and a single carbon atom with the respective neighbours and next neighbours, is visulaized. The Figure attempts to visualize the advantageous data-parallel structure, where each colored outline square shows an independent data structure. Each fullerene isomer in the isomer space have exactly the same regular basis ($N_{atoms} \times 3$), with 3 being the number of neighbours. The gradient opperations presented in thesis produces results in the same regular basis, and thus all gradient array-opperations mentioned in the previous section3.2 can be done simultaneously on each fullerene isomer (orange square) in the Fullerene isomers (see Figure 14), but also on each carbon atom (blue square) in the fullerene at each time-step. Having these conditions for our computations present an ideal foundation for optimal parallisation for CPUs but also GPUs. This section serves the purpose of explaining the advantageous structure of fullerenes and how parallelization can be implemented. Unfortunatly no parallelization have been implemented in the force field of this thesis and it remains a project for future work.

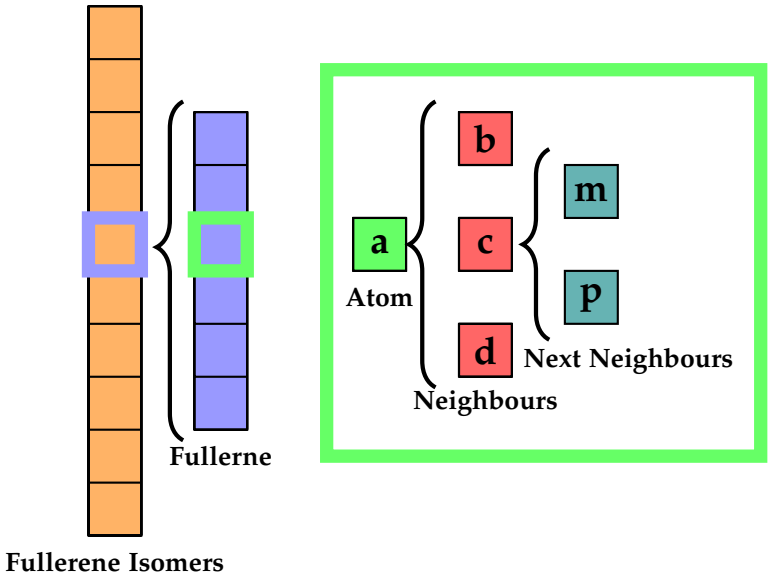


Figure 14: Visualisation of the fullerene problem. From left to right: Each square in the (orange) column represent a specific fullerene in the isomer space. The (blue) column represent a specific fullerene and within the column, each square represent a carbon atom (green). Three neighbours (red) are assigned to each carbon atom and two outer neighbours (teal) to each of the neighbours.

Before we get into detail about why the force field implemented in this thesis offers an optimal foundation for parallel computing, a small introduction to CPUs, GPUs and general parallelization should be mentioned.

Commonly three different classes of hardware are referred to when working with parallelization:

- Single Instruction Single Data (SISD)
- Multiple Instructions Multiple Data (MIMD)
- Single Instructions Multiple Data (SIMD)

CPUs have effecient computer cores, which in general administer the calculations done on a computer. A CPU core constists of many SISD processors, which when working together are good at dealing with MIMD problems, where multiple different operations are performed on multiple data points, and SIMD problems, where the same operation on multiple data points. CPUs are really good at MIMD problems where a number of independent programs which performs different calculations are present. A CPU core then administers a program and works independently of the other CPU cores. This is not feasible for GPUs which only work with SIMD problems[2]. This is due to GPU cores having to perform the same operation in each cores at the same time-step, which is commonly refered to as running in *lock-step*.
Also, due to something called *cache-prediction*, which guesses what operation should be performed next, CPUs use ~ 90% of the power on guessing the next task, in order to perform optimally. However it may not always guess correctly, in which cases can result in the program slowing down substantially and the cache needs to be refilled with new information. Cache prediction is also something that is impossible to avoid in CPUs, as it is part of the CPU core architecture. For GPUs only a fraction is spent on cache prediction and it is easier to exploit. Another notable difference between CPUs and GPUs are the number of cores available and the number of instructions performed each second (commonly refered to as clock cycles). In modern laptops there are usually a single CPU with 2-4 cores and a single GPU with 4000 cores. A standard CPU runs on ~ 4GHz whereas a GPU runs at ~ 1.5GHz which results in the CPU being able to do 16 Giga instructions and a GPU would do 6 Tera instructions. Again this is but a light introduction to parallelization for more information the reader is refered to [12–14].

Because all fullerene structures in the isomer space have exactly the same shape ($N_{atoms} \times 3$) and are independent of each other, not much *cache-prediction* is needed. Furthermore, all the fullerene structures are identical rectangular arrays, and each gradient-opperation e.g. (`d_R`, `d_ang`, `d_dih`, etc.) produces result in the same regular ($N_{atoms} \times 3$) basis and can be performed as data-parallel flat arithmetic array opperations on the whole isomer space ($N_{isomers} \times N_{atoms} \times 3$). Thus, the force field implemented in this thesis offers a structure which is ideal for GPU parallelization. The force field calculations (namely the gradients) allows each GPU core to

2 MIMD problems can be "faked" in GPUs if branching is implemented. Since GPUs work in lock-step a big portion of the GPU might be inactive if working on SIMD problems.

perform independent arithmethic data opperations on each fullerene in the isomer space (or each atom in a fullerene) in lock-step, it has the possibility to be parallelized and scaled up to millions upon millions of GPU cores, which each perform independent operations on different isomers in the huge fullerene isomer space. The force field of this thesis has the possibility to offer maximum GPU performance which makes parallelization essential for tackling the enormous possible fullerene isomer spaces.

3.5 HARMONIC FORCE CONSTANTS

As previously explained, the force field parameterisation is essentially a Taylor expansion of the energy in an internal coordinate system. These coordinates are the bond lengths, angles and the dihedral angles. The Taylor expansion is performed at the optimal geometry, which represents a minimum in our potential energy. Therefore, the first order terms disappear and the second order term yield an accurate description of the energy at minimum. If the system is perturbed too far away from the minimum, the Taylor expansion to second order will no longer be adequate to describe the energy anymore.

$$E(\mathbf{X}_0 + \Delta\mathbf{X}) = E(\mathbf{X}_0) + \frac{1}{2}\Delta\mathbf{X}^T\mathbf{H}\Delta\mathbf{X} + \mathcal{O}(||\Delta\mathbf{X}||^3)$$

Keeping in mind that the geometry and energy is only valid near the minimum it is therefore essential to get as accurate an description of the second order as possible, in order to appropriately resemble real physical energies in our force field optimization at the minimum.
Since the energy function is constructed of multiple indivual distortion terms, it is important to properly describe each second derivative or force constant in analogue to *Hooke's potential* effeciently. In order to obtain the force constants for the individual internal harmonic contributions, the Hessian matrix of the molecule has to be calculated with an acceptable method. The Hessian matrix consists of all second order derivative combinations of the $3N_{atoms}$ nuclear cartesian coordinates.

$$\mathbf{H} = \begin{pmatrix} \frac{\partial^2 E}{\partial^2 x_1} & \frac{\partial^2 E}{\partial x_1 \partial x_2} & \cdots & \frac{\partial^2 E}{\partial x_1 \partial x_n} \\ \frac{\partial^2 E}{\partial x_2 \partial x_1} & \frac{\partial^2 E}{\partial^2 x_2} & \cdots & \frac{\partial^2 E}{\partial x_2 \partial x_n} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial^2 E}{\partial x_n \partial x_1} & \frac{\partial^2 E}{\partial x_n \partial x_2} & \cdots & \frac{\partial^2 E}{\partial^2 x_n} \end{pmatrix}$$

The contribution $\frac{\partial^2 E}{\partial x_1 \partial x_2}$ is then the change of energy of x_2 when x_1 is perturbated. The change of energy at the minimum as a function of x_1 corresponds to the force acting on the system. Hence, in the case of a perturbation in x_1, this gradient describes how the molecule would react in order to minimize the energy after the change. However, physically this is only true close to

the optimal geometry and the optimization from an initial geometry to the optimal geometry is not physically correct, nevertheless we do only care about the physics at the minimum.

Through the *Seminario* method[15] it is possible to translate the sub-Hessian of a system into accurate forces constants of the inter-nuclei properties (bonds, angles, dihedral) by using sub-Hessian Matrix Projection.

The Seminario method uses the sub-Hessian of two atoms **a** and **b** in a chemical bond:

$$\mathbf{H}_{ab} = \begin{pmatrix} \frac{\partial^2}{\partial x_a \partial x_b} & \frac{\partial^2}{\partial x_a \partial y_b} & \frac{\partial^2}{\partial x_a \partial z_b} \\ \frac{\partial^2}{\partial y_a \partial x_b} & \frac{\partial^2}{\partial y_a \partial y_b} & \frac{\partial^2}{\partial y_a \partial z_b} \\ \frac{\partial^2}{\partial z_a \partial x_b} & \frac{\partial^2}{\partial z_a \partial y_b} & \frac{\partial^2}{\partial z_a \partial z_b} \end{pmatrix} V \tag{35}$$

$$\mathbf{V}_{ab} = \frac{1}{2}\mathbf{H}_{ab}(\|\mathbf{ab}\| - R_{ab})^2 \tag{36}$$

$$k_{ab}^{tot} = -\frac{\partial^2}{\partial \mathbf{a} \partial \mathbf{b}}\mathbf{V}_{ab} \tag{37}$$

The eigenvalues (λ^{ab}) and eigenvectors ($\mathbf{v}^{ab}$) of $\mathbf{H}_{ab}$ are the magnitude and direction of the forces acting on the **ab** system when displacing atom **a**. In order to obtain the sub-Hessian $\mathbf{H}_{ab}$ numerous energy calculations are made, which contain all forces acting on the **ab** system when **a** is perturbed, see Equation 37. Thus, the $\mathbf{H}_{ab}$ cannot merely be interpreted as the bond stretching force constants, because a displacement of atom **a** will also change all angles, dihedral angles and many more inter-nuclei properties in which **a** and **b** are involved. In order to obtain the force contribution in the direction of the **ab**-bond a Hessian matrix projection onto the **ab**-unit vector $\widehat{\mathbf{ab}}$ is then applied:

$$k_{ab} = \sum_{i=1}^{3} \lambda_i^{ab} \left| \widehat{\mathbf{ab}} \cdot \hat{\mathbf{v}}_i^{ab} \right| \tag{38}$$

By projecting all eigenvectors of the sub-hessian $\mathbf{H}_{ab}$ onto the **ab**-bond direction with the corresponding eigenvalue it is possible to obtain the restoring force constant between **a** and **b**.

In order to obtain the angular forces between **a**,**b** and **c** a similar procedure is used. Instead of projecting the eigenvectors onto the bond directions we project it onto the perpendicular direction of the bond directions within the plane, see Figure 15, since the angular force will act in these directions.

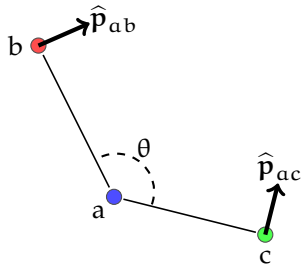


Figure 15: Illustration of Seminario method for angular force constant calculation.

The perpendicular vectors ($\widehat{\mathbf{p}}_{ab}, \widehat{\mathbf{p}}_{ac}$) are projected onto the two sub-Hessian matrices H_{ab} and H_{ac}. The combined angular force constant is then obtained through a two springs connected in series calculation:

$$\frac{1}{k_\theta} = \frac{1}{\|\mathbf{ab}\|^2 \sum_{i=1}^{3} \lambda_i^{ab} \left|\widehat{\mathbf{ab}} \cdot \widehat{\mathbf{v}}_i^{ab}\right|} + \frac{1}{\|\mathbf{ac}\|^2 \sum_{j=1}^{3} \lambda_j^{ac} \left|\widehat{\mathbf{ac}} \cdot \widehat{\mathbf{v}}_j^{ac}\right|} \tag{39}$$

Here each contribution is scaled with the squared length of the bonds ($\|\mathbf{ab}\|, \|\mathbf{ac}\|$).

The dihedral force constants are treated similar to the angles but the sub-Hessian is projected onto the normals of the plane ($\widehat{\mathbf{n}}_{abc}$,$\widehat{\mathbf{n}}_{bcd}$) as shown in Figure 16.

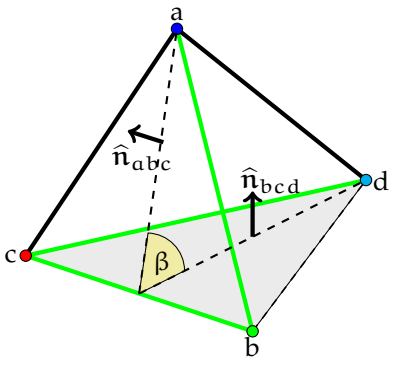


Figure 16: Illustration of Seminario method for dihedral force constant calculation.

The atoms (**b,c,d**) are then connected to a central atom **a** and the dihedral out of plane angle β is then the angle between plane **abc** and **bcd**. The movement of **a** then feels a restoring force from the **ab** bond, and **cd** bond[3].

3 In reality there is no bond, however internally all atoms in a molecule exert force on eachother

The dihedral force is then the sub-Hessians ($\mathbf{H}_{ab}, \mathbf{H}_{cd}$) matrix projection onto plane normals.

$$\frac{1}{k_\beta} = \frac{1}{\|\mathbf{ab}\|^2 \|\widehat{\mathbf{cb}} \times \widehat{\mathbf{ab}}\|^2 \sum_{i=1}^{3} \lambda_i^{ab} \left| \hat{\mathbf{n}}_{abc} \cdot \nu_i^{ab} \right|} + \frac{1}{\|\mathbf{cd}\|^2 \|\widehat{\mathbf{bc}} \times \widehat{\mathbf{cd}}\|^2 \sum_{i=1}^{3} \lambda_i^{dc} \left| \hat{\mathbf{n}}_{bcd} \cdot \nu_i^{dc} \right|} \tag{40}$$

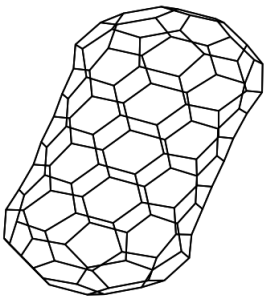
Due to fullerens only being made from pentagons and hexagons, it is possible to categorizes the different force constants into classes dependent on the type of faces that are present. The Seminario method provides a systematic approach to calculating the force constants directly from the cartesian Hessian matrix and the connectivity information. The harmonic force constats can then be calculated and compared by obtaining the Hessians through different ab inito quantum chemistry methods, in this thesis density functional theory (DFT) have been used to obtain the Hessians describing C_{60}-I_h.
A mathematical definition of DFT is beyond the scope of this thesis and the reader is refered to *Gaussian16*[16]. The program *Gaussian16* has been used to compute the carteesian Hessian matrices and a variety of different DFT functionals has been used in the process.

3.6 GAUSSIAN PROGRAM AND FUNCTIONALS

In order to examine the performace of our force field method and compute cartesian Hessian matrices a computational chemistry software package has been used. In 1970 John Pope and his research group at *Carnegie Mellon University* published the computational chemistry software package *Gaussian 70*, which since then has been continiously updated, with the lastest version being *Gaussian16*[16]. The program offers an assortment of different density functional theory methods.

In this thesis two different DFT methods **b3lyp** and **m062x** have been used to compute quantum optimized Hessians and geometries. **b3lyp** which stands for "Becke, 3-parameter, Lee–Yang–Parr" is one of the most used functional in computational chemistry. The functional is relatively cheap to compute and it is possible to do multiple calculations in a short time. **b3lyp** is also relatively simple compared to other functional models as it only incoorporate Local Density Approximation (LDA) which only uses close nuclei interaction, therefore two augmented **b3lyp** methods have also been applied. The Grimme's dispersion **gd3** which applies nuclei dispersion to the functional and the **cam-b3lyp** functional have been used, which is a more precise scheme handling the non-Coulomb part of the exchange functionals. The exchange functionals typically dies off too rapidly and becomes very inaccurate at large distances, making them unsuitable for modeling processes such as electron excitations to high orbitals. The **m062x** functional developed by Truhlar's research group at the University of Minnesota. The **m062x** functional has a strong focus on non-local exchange with parameters that are optimised for non-metal elements of the periodic table. Although the **m062x** functional does not include any dispersion terms, it was optimised for results including dispersion, which means that its parameters can be assumed to have a small contribution towards the dispersion corrections.

A lot can be said about DFT and each functional within the program, but it is beyond the scope of the thesis and the reader is refered to [https://gaussian.com/dft/] for a more thorough description of the *Gaussian16* functionals.



4

CONJUGATED GRADIENT METHOD

In order to obtain good fullerene molecular structures an acceptable optimization method is needed. There exists a large assortment of optimization tools of which two methods, Steepest Descend and Conjugated Gradients have been choosen[17–19]. What is general for these methods (and other methods) is that they aim to solve a linear or non-linear problem. In the linear case the problem takes the form

$$\mathbf{A}\mathbf{x} = \mathbf{b} \tag{41}$$

Where $\mathbf{A}$ is a known, square, symmetric, positive-definite/indefinite matrix, $\mathbf{b}$ is a known vector, and $\mathbf{x}$ is an unknown vector. Problems in the form of Eq. 41 emerge in many important settings, such as solving partial differential equations, circuit analysis, and structual analysis.
Steepest Descend is a simpler version of Conjugated Gradients and will in the following section be explained through Conjugated Gradients Method.

4.1 CONJUGATE GRADIENT METHOD

Conjugate Gradients Methods is among the most popular iterative tools for solving huge systems of linear or non-linear equations. Conjugate Gradients Methods (hencefort CGM) uses gradients and Gram-Schmidt conjugated residuals to find new search directions[19]. The gradient residuals have the neat property of being orthogonal to the previous search direction, so it is guaranteed to generate new linear independent search directions, unless the residual is 0 in which case the problem is solved. In Algorithm 1 a pseudo code of a linear CGM has been outlined.

Algorithm 1 Conjugate Gradient Method

1: $\mathbf{d}_0 = \mathbf{r}_0 = -f'(\mathbf{x}_0)$
2: **while** $|\mathbf{r}_i| > \epsilon$ **do**
3: $\quad \alpha_i = \frac{\mathbf{r}_i^T \mathbf{r}_i}{\mathbf{d}_i^T \mathbf{A} \mathbf{d}_i}$ {Find step size}
4: $\quad \mathbf{x}_{(i+1)} = \mathbf{x}_i + \alpha_i \mathbf{d}_i$ {Update solution}
5: $\quad \mathbf{r}_{(i+1)} = -f'(\mathbf{x}_{(i+1)})$ {Compute new gradient}
6: $\quad \beta_{(i+1)} = \frac{\mathbf{r}_{(i+1)}^T \mathbf{r}_{(i+1)}}{\mathbf{r}_i^T \mathbf{r}_i}$
7: $\quad \mathbf{d}_{(i+1)} = \mathbf{r}_{(i+1)} + \beta_{(i+1)} \mathbf{d}_i$ {Modify gradient}
8: **end while**

The CGM method can be used to find the minimum of any function $f(x)$, provided that it is continious and a gradient $f'(x)$ can be computed. In order to make Algorithm 1 work for non-linear systems a few changes are needed. First, the step-size α cannot be computed analytically and must be approximated through a line search. Second, there exsists various expressions for calcualting β in the Gram-Schmidt conjugated residuals, and it is uncertain which is the best. Two examples are Fleetcher-Reeves formula, which is the default method in the linear CGM case, and Polak-Ribiére formula:

$$\beta_{(i+1)}^{FR} = \frac{\mathbf{r}_{(i+1)}^T \mathbf{r}_{(i+1)}}{\mathbf{r}_i^T \mathbf{r}_i} \tag{42}$$

$$\beta_{(i+1)}^{PR} = \frac{\mathbf{r}_{(i+1)}^T (\mathbf{r}_{(i+1)} - \mathbf{r}_i)}{\mathbf{r}_i^T \mathbf{r}_i} \tag{43}$$

The Fletcher-Reeves method in Eq. 42 converges, if started sufficiently close to the solution. Polak-Ribiére in Eq. 43 can in some cases converge faster for nonlinear systems but may in rare cases cycle infinitely and not converge, this can be avoided and convergens can be ensured by choosing $\beta = max(\beta^{PR}, 0)$. In the case of $\beta = 0$ previous search directions are ignored, and CGM is started over. In the case of Steepest Descend, β is always zero and the search direction will then consistently be orthogonal to the previous search direction.
The more our nonlinear problem differ from a quadratic function, the more quickly the search directions lose conjugacy. Another problem is that a nonlinear functions may have many local minima. CGM is not guaranteed to converge to the global minimum, and may not even find a local minimum if the problem has no lower bound. However luckily for us, the potential described by the force field is a convex harmonic potential, thus only a global minimum exsists.

In order to navigating huge solution spaces in search of the lowest energy, it is desirable to take appropriate step sizes. Big steps might overshoot the minimum and we might end up in some other local extrema and too small

steps are time expensive. Hence it is preferable to find the perfect step size every time, which can be ensured with a good line search method.

4.2 LINE SEARCH

There exists numerous methods for finding steps sizes (or the minimum of a function), but what is general is that in order to find the correct step size α in a given direction $d = f'(X)$ we start out by defining a bracket $[a, b]$ enclosing a minimum. The optimal step size is then the minimum in the direction of the gradient within the bracket, see Figure 17 and Figure 18.

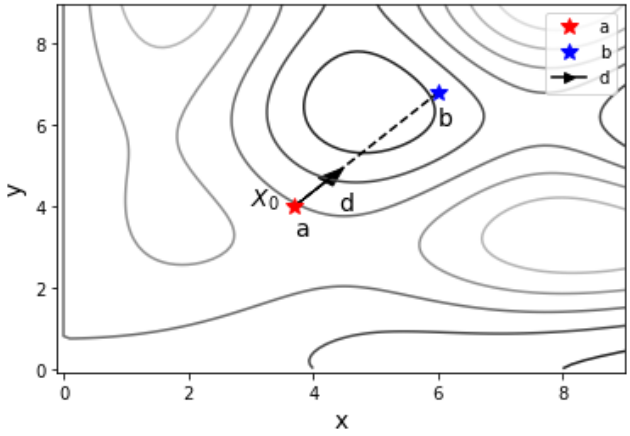


Figure 17: Line search in a solution space, **d** is the gradient and [a,b] brackets an interval of an minimum in the direction of **d** in the solution space.

4.2.1 *Bisection*

Bisection is a robust method for finding roots to continuously differentiable functions in a given interval $[a, b]$. The points $[a, b]$ must bracket the minimum, meaning they must be on opposite sites of a unimodal extremum, see Figure 18.

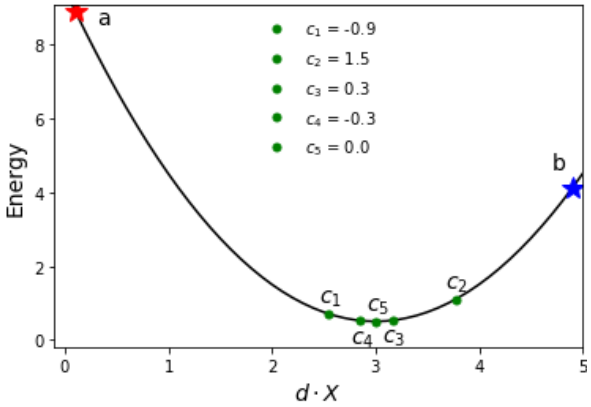


Figure 18: Bisection method in the direction of the gradient of the solution space in Figure 17. [a,b] are the intial brackets and c_{1-5} are the different step sizes with c_5 being the root.

At each iteration the interval is bisected by calculating the midpoint $c = \frac{a+b}{2}$ and the gradient $f'(c)$ is computed. From here there are 3 outcomes: $f'(c) = 0$ which means c is the root, $\text{sign}(f'(a)) \neq \text{sign}(f'(c))$ which means $[a, c]$ now brackets the root, or $\text{sign}(f'(b)) \neq \text{sign}(f'(c))$ which means $[c, b]$ brackets the root. This continues until the minimum is found where the gradient at the minimum is orthogonal, or bellow some threshold, to the search direction or we reach a maximum number iterations. In any case the step size c is returned. The method is guaranteed to converge if the previous criterias are met. The absolute error is halved at each step, producing one bit of the solution per iterration, thus the bisection method converges in linear time, and will converge after $\lceil -\log_2(\frac{b-a}{TOL}) \rceil$ steps.
The Bisection method is outlined in Algorithm 2.

Algorithm 2 Bisection Method

```
Input: X_0, f', a, b, Nmax, TOL
d = −f'(X_0)
while i <= Nmax;  i++ do
    c = (a + b)/2
    if |f'(X_0 + c * d)| <= TOL then
        return c
    else if f'(X_0 + c * d) < 0 then
        a = c
    else
        b = c
    end if
end while
return c
```

Bisection is a reliable method which ensures convergence if previously mentioned criterias are met. Nevertheless it is perhaps the slowest available method.

Another simple line search, very similar to bisection, would be *golden section search*. Where $[a, b]$ brackets a unimodal minimum and $x_1, x_2 \in [a, b]$ with $x_1 < x_2$. Comparing the function values $f(x_1)$ and $f(x_2)$ allows us to discard a subinterval ($(x_2, b] \text{or} [a, x_1)$), knowing that the minimum lies in the remaining subinterval. In order to pick relative positions for x_1 and x_2 within a current interval to τ and $1 - \tau$, where $\tau = 0.618$. With this choice, no matter which subinterval is retained, its length will be τ relative to the previous interval, and the new points (x_1, x_2) will be positioned τ and $1 - \tau$ relative to the new interval. The algorithm is outlined in Algorithm 3. It also converges in linear time similar to bisection, but it will converge in $\lceil -\log_\tau(\frac{b-a}{tol}) \rceil$. Furthermore since Bisection is a method for finding $f = 0$, we have to use the gradient function but golden section search does not have to find a zero, as it only looks for a minimum. Thus, the energy function can be evaluated instead of the gradient, which is computationally easier to calculate. However for this thesis the Bisection method have been chosen, as it was easy to implement and conceptionally straightforward.

Algorithm 3 Golden section search

```
τ = 0.618
x_1 = a + (1 − τ)(b − a)
f_1 = f(x_1)
x_2 = a + τ(b − a)
f_2 = f(x_2)
while (b − a) > tol; do
  if (f_1 > f_2); then
    a = x_1
    x_1 = x_2
    f_1 = f_2
    x_2 = a + τ(b − a)
    f_2 = f(x_2)
  else
    b = x_2
    x_2 = x_1
    f_2 = f_1
    x_1 = a + (1 − τ)(b − a)
    f_1 = f(x_1)
  end if
end while
```

4.2.2 *Faster methods*

There exists an abundant selection of line search algorithms which performs much faster than bisection or golden section seearch. Algorithms such as *Inverse quadratic interpolation* or *Secant method* both add a higher level of complexity in the calculation of the step size α and will often converge much faster than bisection.

In inverse quadratic interpolation three points (u, v and w) and matching function values (f_u, f_v and f_w) are needed. Here v is an initial guess of the minimum within the backet [u,w] and often the bisection method is used for the first step. The minimum of the inverse quadratic interpolation is given by

$$q = \frac{f_v f_w}{(f_u - f_v)(f_u - f_w)} u + \frac{f_u f_w}{(f_v - f_u)(f_v - f_w)} v + \frac{f_u f_v}{(f_w - f_u)(f_w - f_v)} w \tag{44}$$

The method only requires one new function calculation at each iteration and if started close enough to a minimum the method will converge with a rate $r \simeq 1.839$, making it much faster than bisection.

In secant method the finite difference approximation is used:

$$f'(x_k) \simeq \frac{f(x_k) - f(x_{k-1})}{x_k - x_{k-1}} \tag{45}$$

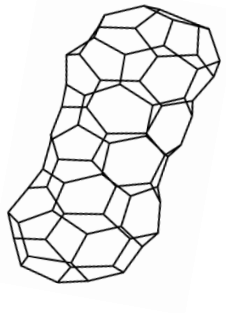
The secant method then approaches the function f by the secant line through the two previous iterations, taking the $f = 0$ of the resulting linear function to be the next step. The algorithm is outlined bellow

Algorithm 4 Secant Method

Input: $X_0, X_1 f, f', TOL$
$X_0, X_1 =$ *initial guesses*
while $f'(X) > TOL; \quad k = 0, 1, 2, ...$ **do**
$X_{k+1} = X_k - f(X_k)(X_k - X_{k-1})/(f(X_k - f(X_{k-1})))$
end while
return X

Similar to Inverse quadratic interpolation, secant method only requires one new function evaluation at each iteration and converges with a superlinear rate of $r \simeq 1.618$ given that it is started close enough to the solution.

A fourth method is *brent's* method, which is the method used in the *Fullerene* program. It combines secant, inverse quadratic interpolation, and bisection in order to ensure a fast method with guarenteed convergence.
In practise this is one of the default methods when working with numerical optimization as it will converge in superlinear time. It would have been preferable to use as the line search method. However bisection was choosen, as it ensures convergence and is much simple than brent's method. The algorithms time performance will be discussed in section 5.4.



5

RESULTS AND DISCUSSION

5.1 GRADIENT CORRECTION

As previously mentioned the force field optimizer implented in the *Fullerene* program by Wirz et al. (refered to as Wirz in the results) required a dihedral gradient term which took outer dihedrals into account. In Figure 19 three force field optimzers are presented. The first one used by Wu et al. (refered to as Wu in the results), the Wirz method, and the augmented Wirz et al. force field implemented in this thesis, which takes outer dihedrals into account (refered to as Wirz2 in the results). In order to decide which gradient best locates the minimum energy, a line search in the direction of each methods gradient is performed. The *Fullerene* program's optimized geometry for C_{60}-I_h is used as an intial geometry in order to compare the performance of each method. The gradients are compared to the numerical calculated gradient.

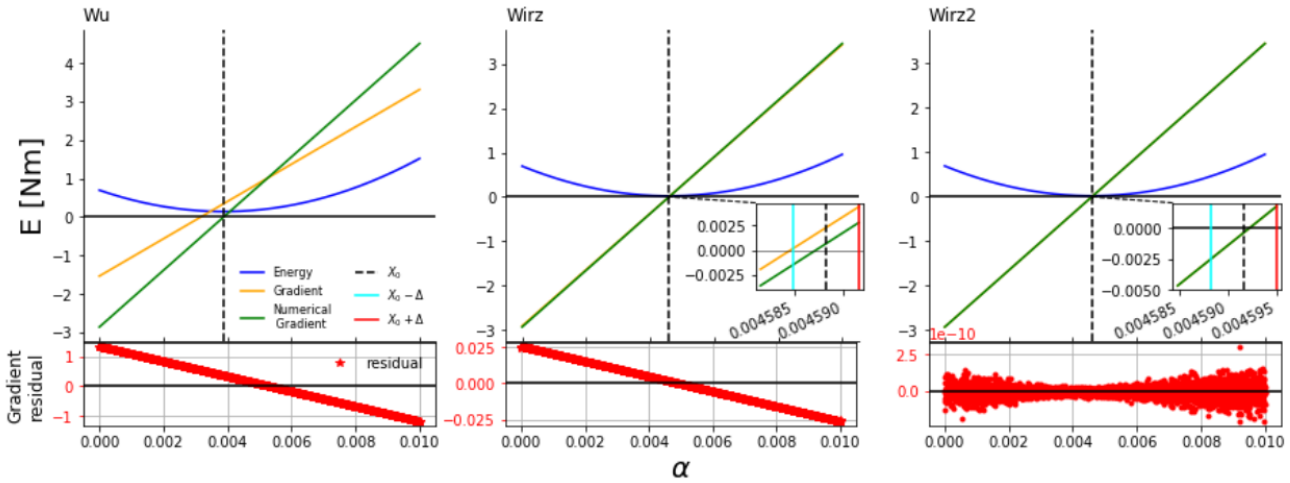


Figure 19: Calculated gradient versus numerical gradient for the Wu, Wirz, and Wirz2 method. The line search has been performed from an geometry provided by the *Fullerene* program and a minimum α is bracket between $[0, 0.01]$.

In the first plot a noticeable difference between the Wu gradient and the numerical gradient can be seen. In the second plot with the introduction of dihedral angles, Wirz manages to come much closer to the minimum but still is not within one nummerical step Δ of the numerical gradient. However by adding the expanded correct dihedral gradient we mange to perform just as well as the nummerical gradient, leaving the gradient residual as nummerical errors. In the next section the force field optimization methods on C_{60}-I_h are compared to a DFT optimized geometry of C_{60}-I_h.

5.2 GEOMETRY FORCE FIELD OPTIMIZATION

For the Conjugated gradient in the geometry force field a tolerance $\epsilon = 10^{-7}\,N$ has been used. For C_{60}-I_h a $N_{max} = 600$ have been chosen and for $C_{72}nt$ a $N_{max} = 720$ have been chosen. The Polak-Ribiére formula in Eq. 43 has been choosen for obtaining β. For the bisection method, when calculating steps size α, a convergence tolerance of $TOL = 10^{-10}\,N$ has been used. For performance comparison DFT optimized geometries from the *Gaussian16* program have been calculated. The optimal geometry of the *fullerene* program have been used as initial inputs for the DFT calculations. The **m062x** functional has been calculated for C_{60}-I_h. For the $C_{72}nt$ only the **b3lyp** was able to compile without crashing, which is not exellent, however the **b3lyp** is still more accurate compared to an energy force field, thus it will suffice.

5.2.1 *Optimized Geometry of C_{60}-I_h*

For the geometry optimization of the C_{60}-I_h fullerene the force constants of Table 2 has been used. Furthermore the energy function in Eq. 13 have been used to evaluate the calculated geometry energy at each iteration, this does not affect the optimization in any way and only serves the purpose of getting an intuition of the performance of the Wu, Wirz, and Wirz2 force fields. The Wu method converges at a higher energy than the Wirz and Wirz2 methods, which both, due to them having dihedral contributions, converges closer to a optimal minimum. In Figure 21 the different optimal C_{60}-I_h geometries are shown together with the starting geometry and the **m062x** optimized geometry. The different force fields performance are compared to the **m062x** optimized geometry of C_{60}-I_h and the root mean squared error of each parameter (R, angle, and dihedral) is shown in Table 3.

Method	Bond distance R [Å]	Internal Angle [°]	Dihedral Angle [°]
Wu	0.0017939	0.0016744	0.000137
Wirz	0.00036372	$2.7984979\ 10^{-10}$	$1.169802\ 10^{-7}$
Wirz2	0.00036371	$2.7984977\ 10^{-10}$	$1.169802\ 10^{-7}$

Table 3: The root mean square error of inter-nuclear parameter from the optimized geometries compared to the **m062x** method.

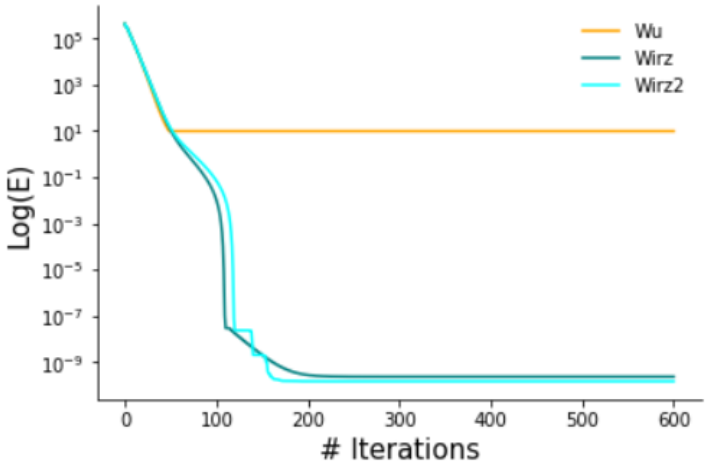


Figure 20: Force field performance of Wu, Wirz, Wirz2 of the geometry of C_{60}-I_h. The maximum number of iterations $N_{max} = 600$ and the $\epsilon = 10^{-7}$ N.

All the methods are in good agreement with the **m062x** optimized geometry and varies very little. Both Wirz and Wirz2 perform better than the Wu method which is an indication that the dihedral angles are an important feature when characterizing the geometry of fullerenes. In the next section the methods will be compared to a more complex structure, which is the thin nanotube of $C_{72}nt$ where all the curvature (the pentagons) are situated in each end.

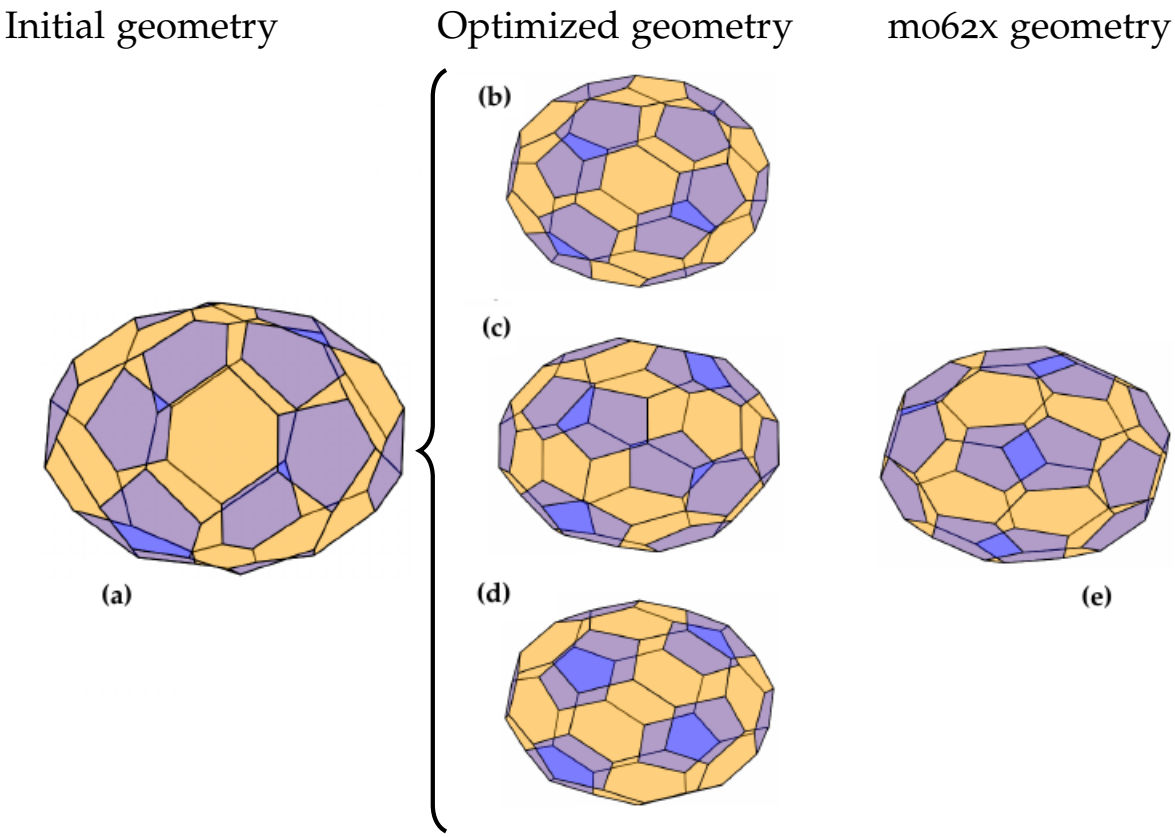


Figure 21: C_{60}-I_h geometries. **(a)** Tutte embedding C_{60}-I_h used as initial geometry **(b)** Wu optimized geometry **(c)** Wirz optimized geometry **(d)** Wirz2 optimized geometry **(e)** *Gaussian 16* **m062x** optimized geometry. The pentagons are colored blue and the hexagons are colored orange.

5.2.2 *Geometry Force Field Optimization of* $C_{72}nt$

In this section we compare the force field methods optimized geometries of the $C_{72}nt$, which is a more complex fullerene. The RMS-error of the inter-nuclear parameters compared to the **b3lyp** optimized geometry[1] is shown in Table 4. The energy at each iteration is shown in Figure 22 and the initial geometry and **b3lyp** geometry is shown together with the different force field optimized geometries in Figure 23.

Method	Bond distance R [Å]	Internal Angle [°]	Dihedral Angle [°]
Wu	0.31006	0.44618	0.01188
Wirz	0.01135	0.00330	0.00123
Wirz2	0.00525	0.00253	0.00114

Table 4: The RMS-error of inter-nuclear parameter of $C_{72}nt$ from the optimized geometries compared to the **b3lyp** geometry.

1 Ideally we would have wanted the **m062x** optimized geomtry of $C_{72}nt$. However time did not permit this as each *Gaussian16* DFT calculation could take weeks and in some cases would crash, leaving us with only the **b3lyp** results.

Given that the Wu method was initially developed for the C_{60}-I_h fullerene it comes as no surprise when it does not perform adequately for the nano tube, as shown in Figure 22 and Figure 23 **(b)**. Focusing on the Wirz and Wirz2 methods we see in Figure 22 that both performs better than the Wu method, but Wirz2 finds a lower minimum which is further emphasized from the results in Table 4. This is due to the correct dihedral gradient in the optimization.

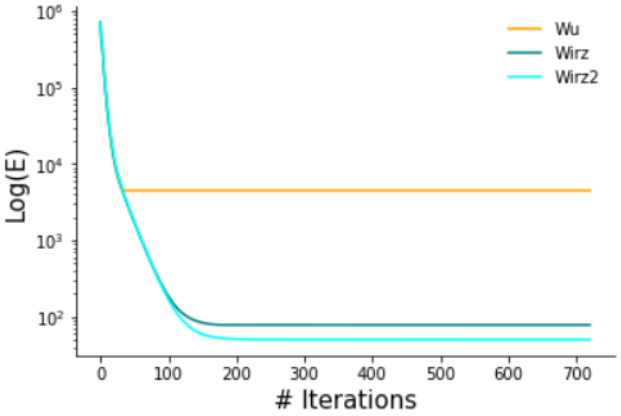


Figure 22: Force field performance of Wu, Wirz, Wirz2 of the geometry of C_{72}nt. The maximum number of iterations $N_{max} = 720$ and the $\epsilon = 10^{-7}$ N.

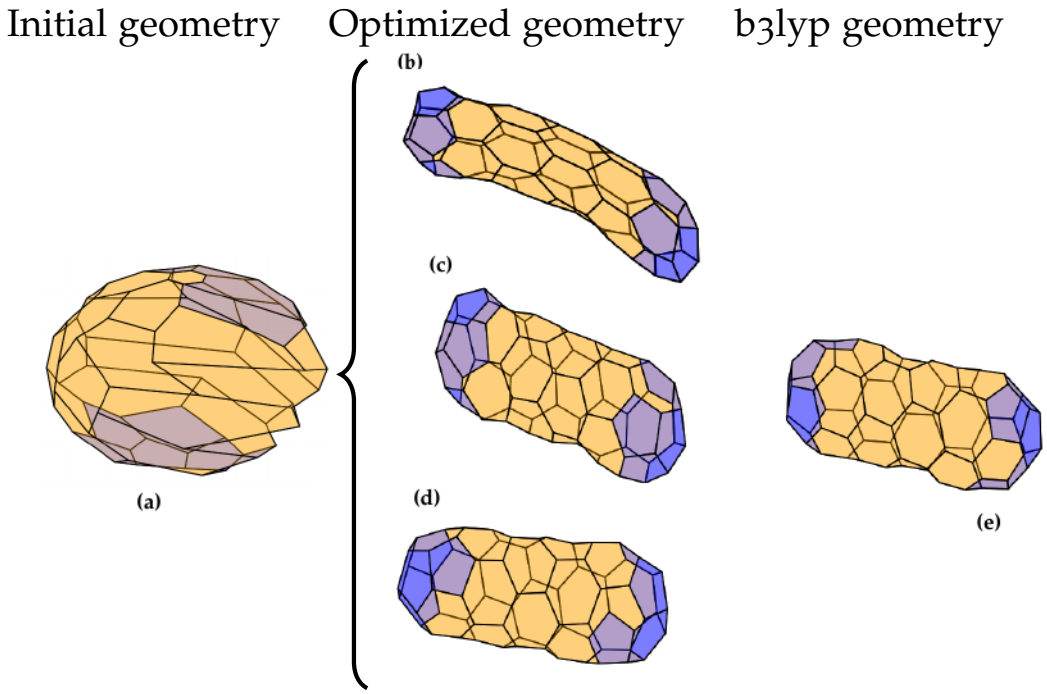


Figure 23: Force field optimized geometries of C_{72}nt. **(a)** Tutte embedding C_{72}nt used as initial geometry **(b)** Wu optimized geometry **(c)** Wirz optimized Geometry **(d)** Wirz2 optimized Geometry **(e)** *Gaussian 16* **b3lyp** optimized geometry.

5.3 WHEN THE WIRZ METHOD FAILS

In this section three fullerene structures where the Wirz method fails to accurately optimize the geometry will be presented together with the corrected Wirz2 optimization. A difficult problem of automated geometry optimization is to detect when something goes only slightly wrong. In the upper row of Figure 24 three fullerenes which have not been optimized correctly by the Wirz et al. force field are shown. Here conected faces are inverted, twisted, and pointing out of the fullerens, which is not physically feasible at a chemical equilibrium. In the bottom row the same fullerenes have been optimized using the Wirz2 force field implemented in this thesis. Here the optimal geometries are correcly calculated which is due to the outer dihedral parts. In Appendix B more examples of known C_{60} and C_{80} fullerenes can be seen where the Wirz et al. force field fails to correctly optimze the geometries, together with the correctly Wirz2 optimized geometries.

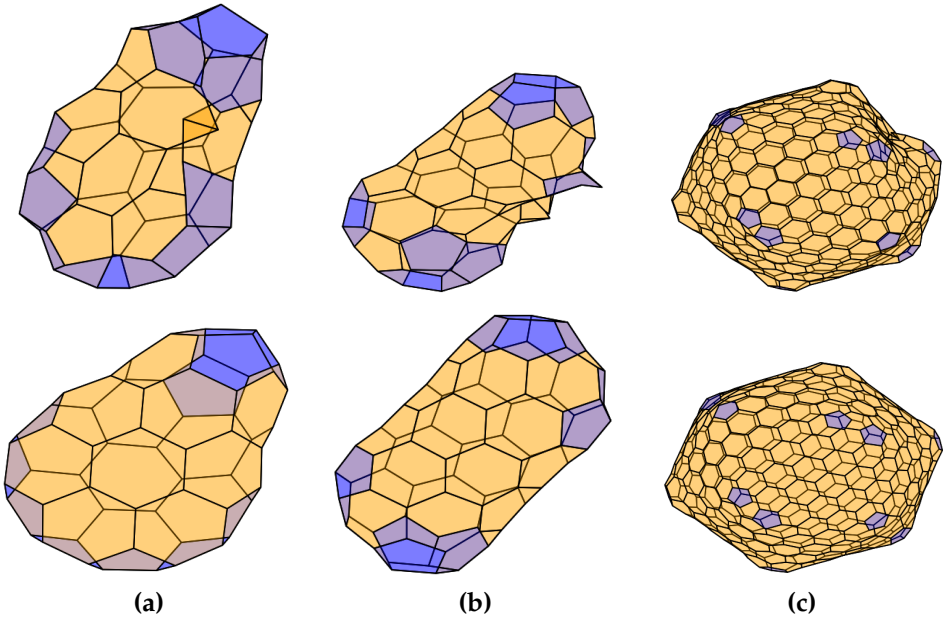


Figure 24: Comparisson of failed Wirz optimized fullerene geometries (first row) and Wirz2 optimized fullerene geometries. **(a)** C60-[1,2,3,4,7,19,23,26,27,30,31,32] , **(b)** C80-[1,2,3,5,10,18,29,30,38,39,41,42], **(c)** C540Ih.

The force field implemented in this thesis does not solve the problem of detecting when fullerenes optimize to something wrong but in the known cases it seems to perform better on the fullerenes where the Wirz et al. force field fails. However, this is no guarantee that the force field implemented in this thesis is incapable of making mistakes. More samples should therefore be obtained and tested. Furthermore a carefull outlier detection scheme should be implemented.

5.4 BENCHMARKING

Since all the code of this thesis is written in *Python*, the time performance comes nowhere near that of the *Fullerene* program's *Fortran* implementation. The performance of the force field implemented implemented in this thesis can be seen in Figure 25 and 26. The benchmarking has been done on fullerenes with isocahedral symmetry ranging from the smallest fullerene C_{20}-I_h dodecahedron to a C_{1040}-I_h fullerene.

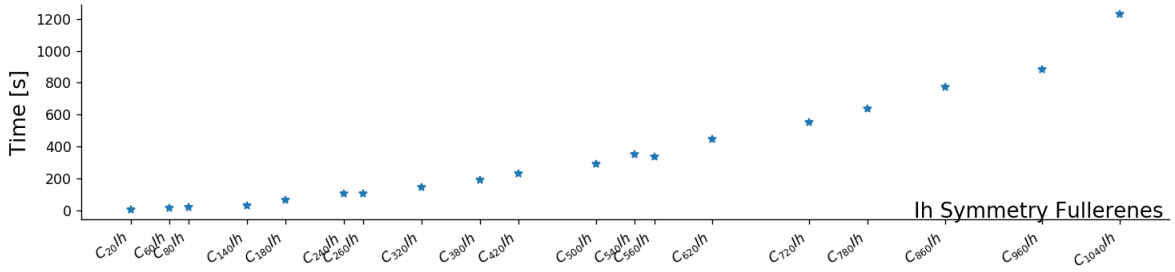


Figure 25: Time benchmark plot of gradient convergence time of isocahedral symmetric fullerenes sized C_{20}-I_h - C_{1040}-I_h.

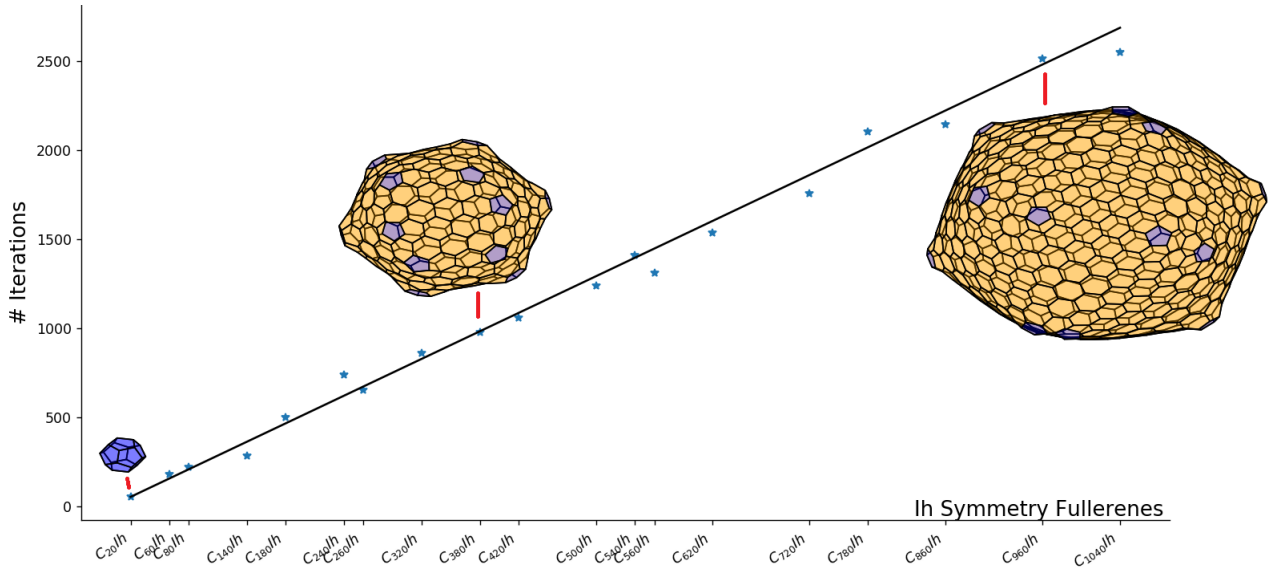


Figure 26: Benchmark plot of iterations needed for convergence with respect to isocahral symmetric fullerene size. A straight line with slope $a = 2.5989$ has been fitted with the data, illustrating the linear progression. The fullerene force field seems to need approximately $N = 2.6N_{atoms}$ iterations in order for the gradient to converge bellow the tolerance $\epsilon = 10^{-7}$ N.

The number of arithmetic operations needed when calculating either the energy or the gradient is $\mathcal{O}(N)$ operations, where $N = N_{atoms}$. In the linesearch $\lceil -\log 2(\frac{b-a}{tol}) \rceil$ iterations are needed, which is independent of N, and thus $\mathcal{O}(1)$ operations are needed. The number of iterations for the CGM follows a linear slope, as shown in Figure 26, making it $\mathcal{O}(N)$. This results in the overall time complexity being $\mathcal{O}(N) + \mathcal{O}(1) + \mathcal{O}(N) = \mathcal{O}(N^2)$.

A downfall and a strength of the force field performance also lies in the choice of line search method. If we wanted to optimize the fullerene isomers sequentially, which has been the case of this thesis, it would be optimal to choose a faster line search method, such as brent's method. However in the case of writing optimal code for GPUs a bottleneck is often the *test*-statements evaluated during the optimization (namely *if* and *while* statements). Since all the GPU cores execute the same arithmetic operations at each time-step, but on different data, the *test*-statements possesses a challenge, if the isomers converge at different time-steps, which is not optimal for lock-step operations. Using the bisection method we know that convergence is guaranteed after $\lceil \log_2(\frac{b-a}{TOL}) \rceil$ steps, which can be calculated prior to running the line search. Furthermore from Figure 26 we see that approximately $N = 2.6N_{atoms}$ iterations are needed in the CGM in order to converge[2]. Thus, two *test*-statements, which checks for convergence, can be removed and the GPU cores can run a fixed number of steps for the line search and CGM and avoid any communication or thread synchronization, which often are problems that makes parallel code run slow.
A speed up of $N_{isomers}$ should be achievable on independent MIMD CPUs, where the calculations are independent, such that one instruction is performed for each calculation. However due to the code and problem being optimal for GPUs, the whole problem can be treated as one huge SIMD calculation, and a speed up of $N_{isomers}N_{atoms}$ could be possible, but ultimately it could scale with the number of GPU cores available.

5.5 SEMINARIO FORCE CONSTANTS FOR C_{60}-I_h FULLERENE

In this section multiple DFT functionals from the *Gaussian16* program have been used in order to obtain Cartessian Hessians needed for the Seminario sub-Hessian projection for getting the C_{60}-I_h fullerene force field constants. A combined mean for each force constant can be seen in Table 5. The methods used for calculating the Hessians are:

- **b3lyp**
- **b3lyp + gd3**
- **cam-b3lyp**
- **m062x**

Force Parameters (N/m)				
k_{ph}	k_{hh}	k_p	k_h	k_{phh}
353.377	518.992	207.924	216.787	3.772

Table 5: Table of C_{60}-I_h Seminario force constants. A combined mean of the results of $(b3lyp, b3lyp + gd3, cam - b3lyp, m062x)$.

2 This is also the case in the optimization of $C_{72}nt$ in Figure 22.

For comparison the results of the **m062x** have been used, as they have shown prominent results in previous related work performed within the fullerene project group. The results of the force constants are shown in Figure 27 together with the geometry with bonds colored correspondingly to their bond stretching force constant. The results of the other DFT functionals can be seen in Appendix C.

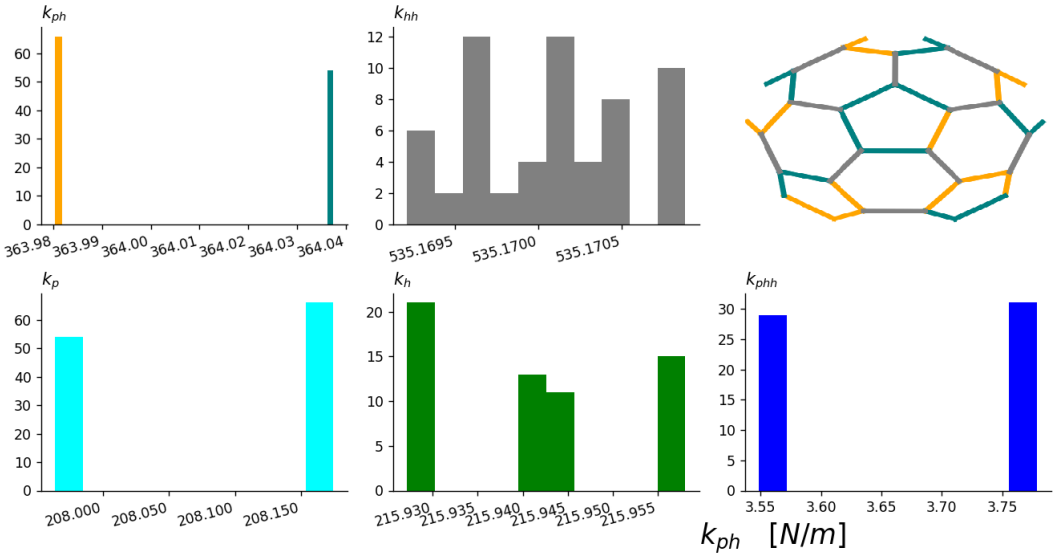


Figure 27: Distribution of force parameters of the **m062x** method. The upper most right figure is the **m062x** geometry with the two k_{ph} results (orange, teal) respectively and k_{hh} (gray).

What is immediatly notable, and present in the results of each DFT functionals, are the two peaks present in the results which are pentagon dependent (k_{ph}, k_p and k_{phh}). Since the carbon atoms in fullerenes are sp-2 hybridized, the two distinct peaks could be the result of the *Gaussian16* program introducing double bonds between neighbouring atoms. However we would then expect a single double bond (teal) to be present at each carbon atom, which as shown on the geometry in Figure 27 is not the case. A more thorough investigation of this is needed in order to say anything concrete. Since double bonds have not been implemented in the force field, a mean of the values will be used.

The force parameters have been used in the geometry optimization in Figure 28. However due to high symmetry only some force constants can be obtained, which results in C_{60}-I_h being the only fullerene we can optimize using only these parameters. In order to calculate the remaining fullerene force constants a number of DFT calculations of different shaped fullerenes are therefore needed.

In section 5.8 a discussion of the legitimacy of the Seminario method and wether or not an improvement for calculating fullerene force parameters should be implemented.

5.6 OPTIMIZATION USING SEMINARIO FORCE CONSTANTS

In this section the force constants from Table 5 have been used in the geometry force field optimization of the C_{60}-I_h molecule. A max number of iterations $N_{max} = 600$ have been used with a gradient tolerance $\epsilon = 10^{-7}$ N.

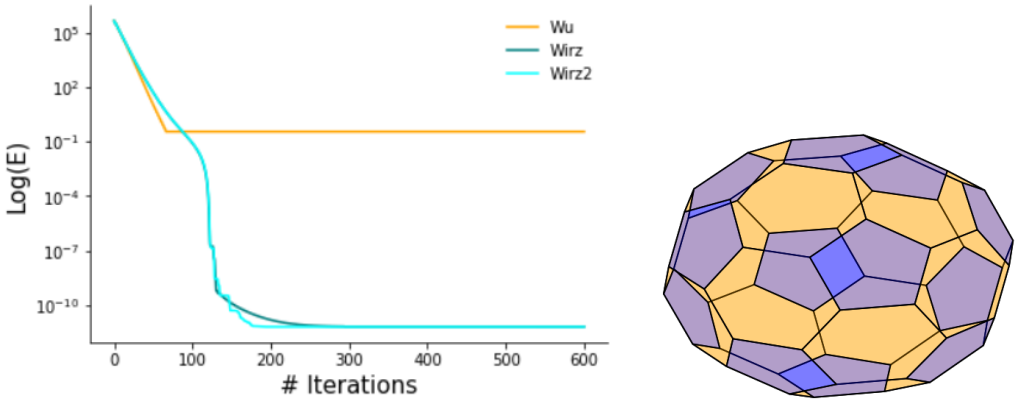


Figure 28: (**a**) Force field performance of Wu, Wirz, and Wirz2 on the geometry of C_{60}-I_h with Seminario force constants. The maximum number of iterations $N_{max} = 600$ and the $\epsilon = 10^{-7}$ N. (**b**) Wirz2 optimized geometry.

Method	Bond distance R [Å]	Internal Angle [°]	Dihedral Angle [°]
Wu	0.000310	$1.356638\ 10^{-5}$	$3.376614\ 10^{-6}$
Wirz	0.000150	$9.089976\ 10^{-9}$	$1.275802\ 10^{-8}$
Wirz2	0.000150	$9.089974\ 10^{-9}$	$1.275799\ 10^{-8}$

Table 6: The RMS-error of C_{60}-I_h inter-nuclear parameters from the Seminario parameter force field optimized geometries compared to the **m062x** geometry.

It is no surprise that the RMS-errors in Table 6 are better than the ones in Table 3 as the force constants have been tailored to specifically describe the C_{60}-I_h fullerene. Furthermore the Wirz method manages to perform almost as well as the Wirz2 method with the new force constants.
Due to the high symmetry of the isocahedral fullerene C_{60}-I_h we could only obtain a few force constants. In order to get the remaining force constants, which then can describe all fullerenes, more DFT calculations of different shaped fullerenes are needed. Furthermore DFT calculations of different sized fullerenes would also be interesting to see if that have an influence on the harmonic force constants.

5.7 AUGMENTING THE FORCE FIELD

When optimizing the geometry of fullerenes the approach of this thesis has been to describe the geometry from three intrinsic properties namely the bond distance, the bond-bond angular bending, and the three connected face dihedral angles. Still is it possible to describe the fullerene geometry more accuratly if the force field is augmented and additional intrinsic geometry properties are added. Another geometric feature of fullerenes could be the pentagon and hexagon face flatness.

From chemistry we know that molecules that are aromatic tends to be very stable. An aromatic molecule (or conjugated molecule) is a molecule where the electrons from the π-orbitals are shared between all the atoms. Benzene is one of these stable aromatic molecules which shares the π-orbital electrons. In Figure 29 we see the classical representation of the benzene ring, with carbon atoms (gray) where the black connections represent the σ-orbital bonds (the strong bonds) and the red bonds represent the π-orbital bonds (weaker bonds). Since the electrons from the π-orbitals move freely in the benzen ring we essentially have a superposition of double bonds which makes a very rigid molecule that prefer to be planar. One can vizualise the π-orbital electrons moving around both above and bellow the benzen ring and sort of pushing it from both sides, consequently creating this face flatness.

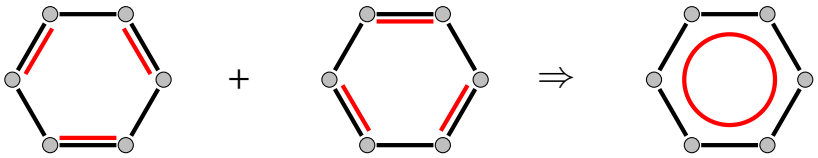


Figure 29: Aromatic molecule benzen. Carbon atoms (gray) with σ-bonds (black) and free π-orbital bonds (red).

While some fullerenes are spherical aromatic (or close), we can assume that the pentagons and hexagons too are aromatic (or close), thus face flatness could present a great addition to the force field. This is further backed up from the mean distance to the plane of each atom compared to the least squared plane of the respective faces of the **m062x** optimized geometry of C_{60}-I_h in Figure 30. Here we see that the pentagons are perfectly planar and that the hexagons vary only with approximately 10^{-6}Å.

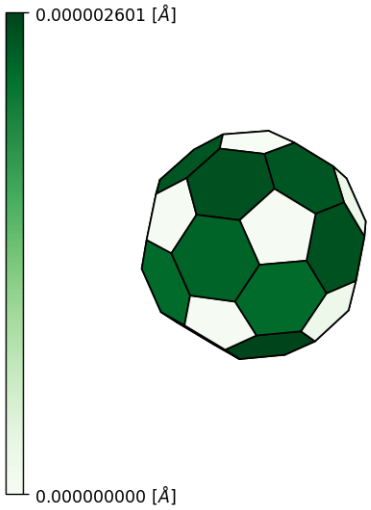


Figure 30: Plot of mean distance to the least squared face plane of the m062x optimized geometry for C_{60}-I_h.

5.8 VALIDITY OF A MOLECULAR FORCE FIELD

From the results in Figure 28 and Table 6, where the calculated Seminario force constants was used, we can conclude that we must be reasonably close to the minimum given the parameter RMS-error. However considering that the Seminario method is for obtaining force constants for general chemical structures, the question can be raised whether there exist methods more suitable for fullerene structures. For the angular force constants Allen et al.[21] offers a slightly different approach, which takes into account the coupling of multiple angles applied to the same atoms. This has been shown to improve vibrational frecuenzy results in the cases, where a single atom is bonded to a carbon ring. However in the case of fullerenes no single atom is connected to a ring. The atom is fully surrounded and therefore part of a much larger spring system. Meaning that in order to properly capture the effects of pertubating **a** in a fullerene, as shown in Figure 31, a much larger coupled force constant should be computed. The same goes for stretching forces and dihedral forces.

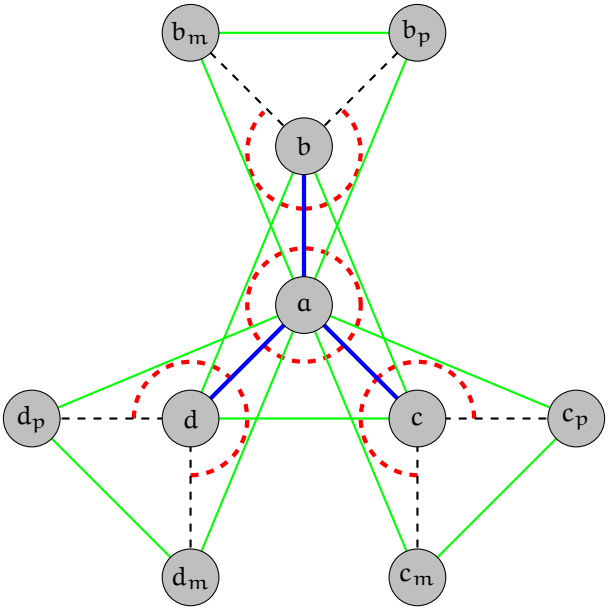
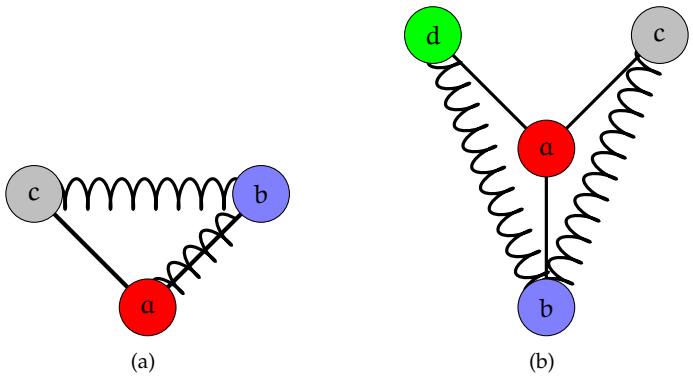


Figure 31: Illustration of parameters affected by the perturbation of a center atom **a**.

Since Allen et al. does not offer any correction to bond or dihedral forces, the classisc Seminario method has been used which still produces force constants that help opbtain fullerene geometries that are in good agreement with ab inito quantum mechanical geometries.

5.8.1 *Cross-terms*

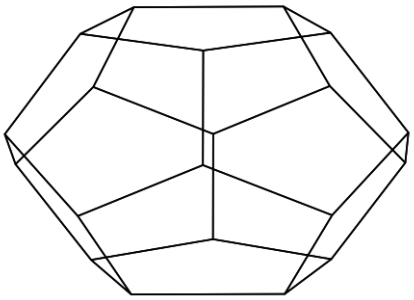
Suppose we wanted to calculate the contribution of bond stretching energy of a specific bond in the fullerene. Because one bond length cannot be changed without changing other inter-nuclear properties e.g. opposite dihedral angle and dihedrals, dependent on that specific bond, the energy would not be an accurate description. The question, *"how are we to isolate one contribution to the energy from the other"* is a problem that is common in many force field methods and is commonly ignored. Force field methods does not claim to be exact and are merely a tool for fast calculations given simple assumptions from chemical intuition. However, one cannot escape the fact that, unfortunately, in most cases the individual contributions to the energy expression are linearly related, and thus the magnitudes of the force field parameters are not necessarily meaningful. The article by Dinur et al.[22] argues that in order to get a more descriptive energy function close to the equilibrium, cross terms should be implemented e.g. bond-bond cross term, bond-angle cross term, or angle-angle cross term. In Figure 32 a visualization of the dependent cross terms of bond-angle and angle-angle is shown.



(a) (b)

Figure 32: Illustration of Cross terms. (a) bond-angle cross term, (**b**) angle-angle cross term.

Furthermore nonbonded interactions could be applied as well, such as Coulomb's law, electrostatic terms reflecting charge-charge interactions or, a Lennard-Jones term describing nonbonded exchange repulsion within the fullerene. However the approach of the force field for both Wirz et al. and the optimized version Wirz2 have neglected these terms, as the simplified energy expression in Eq. 14 encapsulates the essential geometric entities of fullerenes. Also the implementation of nonbond terms such as Coulomb's law would be time costly and would add an factor of N^2 calculations at each iteration. Thus the cross terms and nonbond calculations have been neglected.



6

CONCLUSSION AND OUTLOOK

In this thesis, a force field method for optimizing fullerene geometries was developed, with documentation of the implemented energy and gradient expressions and optimization tools.
The method provides fullerene geometries, which are in good agreement with ab inito quantum optimized geometries. A software package[Molecular-Shapes-of-Fullerenes] is provided, which generates optimized geometries given an intial geometry with relevant graph information e.g. connectivity and face orientation. Furthermore the software can visualize the results in an intuitve and interactive way. The software can compute stretching, angular bending, and dihedral harmonic force constants given a *Gaussian16* ".log" file containing a cartesian Hessian matrix.
This software should be considered in the larger context of the Folding Carbon Project.

The calculated force field energies and gradients have been written in a simple vectorized notation, making it more interpretable, in terms of geometry operations, and verifiability, as the mathematical calculations are directly translated into simple vectorized code. The code is easy to extend with additional physics. Additionally the vectorized gradient-operation produces results in the same regular ($N_{atoms} \times 3$) basis as the fullerene isomers, and can be performed as data-parallel flat arithmetic array opperations on the whole isomer space ($N_{isomers} \times N_{atoms} \times 3$) making it ideal for independent lock-step operations and optimal GPU parallelization, and possibly scalable with the number of GPU cores available.

The dihedral gradient correction to the Wirz et al. force field, has given a correct physical description of the gradient of Eq. 14 and has in the case of C_{60}-I_h and C_{72}nt proven to give results that are closer to ab intio quantum optimized geometries than that of Wirz et al.
A correction of the dihedral gradient have also further made it possible to obtain fullerene geometries, where the Wirz et al. force field would otherwise

fail. However a carefull outlier detection scheme should be implemented in order to detect such problems within the force field of this thesis.
A possible solution to detecting geometry outliers could be, to check how much the inter-nuclei parameters e.g. bond distance, bond-bond bending and dihedral angles varied from the optimal defined values. Furthermore the optimized geometry should be checked if it has converged to a minimum (gradient equal zero, or the Hessian being positive definit, if second derivatives was implemented into the force field). Checking if the optimized geometry is convex or if the face plane normals formes an ordered, smooth vector field pointing away from the fullerene center. These are all possible candidates, which could be implemented in order to detect if the geometries optimize correctly.

From the various *Gaussian16* DFT optimized geometries of C_{60}-I_h, harmonic force constants have been calculated, which applied in the force field optimization produced results that are in good agreement with the ab inito optimized geometries. However more work on this matter needs to be done, since all harmonic force constants, which describes fullerenes, could not be obtained from the C_{60}-I_h fullerene. Furthermore calculations of different fullerenes shapes and sizes should be performed in order to give more general harmonic force constants which can be applied for all fullerenes, or perhaps applied to specific shapes.

A discussion of fullerene face flatness in section 5.7 has opened the possibily of augmenting the Wirz et al. force field model by adding an energy term with respect to the face flatness. The **m062x** results of the C_{60}-I_h shows that the fullerene faces are notably flat but a more extensive investigation is needed.

As this work have created optimal conditions for parallelization it would be of interest to take this study further and actually implement it. This would truely advance the possibilities of searching through the enormous fullerene isomer spaces in search of desired geometries and physical properties. A fast automated fullerene geometry optimization tool together with the additional work being done in the Folding Carbon Project could enable researchers and engineers to automatically find fullerenes that match their demand and systematically produce them from scratch. Given the predicted applications of fullerene molecules, this could mean an huge step in nanotechnology.

Lastly a topic which is not mentioned in the thesis, but which in honesty truely inspired the choice of direction regarding this thesis, namely differential geometry and geometry optimization for fullerenes, was the application to medical science.
In 1999, Ganser et al.[23] discovered the HIV-1 virus capsid to have fullerene structure. A later study by Mattei et al.[24] performed high-resolution cryo-

tomography on 539 live HIV vira, showing that HIV capsids form a variety of fullerene nanocones. In particular, they propose specific C_{384} and C_{438} structures to match two of the tomograms. The project (CARMA/HIV-search/) seek to examine if other fullerenes might be better suited to describe the geometry of HIV vira. There are $\sim 2.3 \times 10^{13}$ fullerene structures between C_{200} and C_{500} which is the size range within which HIV capsids lie. The first step would be to go through this enormous isomer space and extract all the nanocones, which can be potential HIV structures.
Hopefully the work done in this thesis can help future projects scratch the surface of this enormous and interesting problem.

Appendices

A

EXTENDED CALCULATIONS OF DERIVATIVES

CHEAT SHEET

Quotion Rule (QR) Vec/Scal

$$\nabla_{\mathbf{a}}\Big(\frac{\mathbf{g}(\mathbf{a})}{h(\mathbf{a})}\Big) = \frac{(\nabla_{\mathbf{a}} \otimes \mathbf{g}(\mathbf{a}))h(\mathbf{a}) - (\nabla_{\mathbf{a}} h(\mathbf{a})) \otimes \mathbf{g}(\mathbf{a})}{h(\mathbf{a})^2} \tag{QR}$$

Product Rule (PR)

$$\nabla_{\mathbf{a}}(\mathbf{ab} \cdot \mathbf{ac}) = (\nabla_{\mathbf{a}}\mathbf{ab}) \cdot \mathbf{ac} + \mathbf{ab} \cdot (\nabla_{\mathbf{a}}\mathbf{ac}) \tag{PR1}$$

$$\nabla_{\mathbf{a}}(\mathbf{ab} \times \mathbf{ac}) = (\nabla_{\mathbf{a}}\mathbf{ab}) \times \mathbf{ac} + \mathbf{ab} \times (\nabla_{\mathbf{a}}\mathbf{ac}) \tag{PR2}$$

Vector properties:

$$\mathbf{a} \cdot (\mathbf{b} \times \mathbf{c}) = \mathbf{b} \cdot (\mathbf{c} \times \mathbf{a}) = \mathbf{c} \cdot (\mathbf{a} \times \mathbf{b}) = -\mathbf{a} \cdot (\mathbf{c} \times \mathbf{b}) = -\mathbf{b} \cdot (\mathbf{a} \times \mathbf{c}) = -\mathbf{c} \cdot (\mathbf{b} \times \mathbf{a}) \tag{VP}$$

Commutative dot product:

$$\mathbf{a} \cdot (\mathbf{b} \times \mathbf{c}) = (\mathbf{a} \times \mathbf{b}) \cdot \mathbf{c} \tag{Cdp}$$

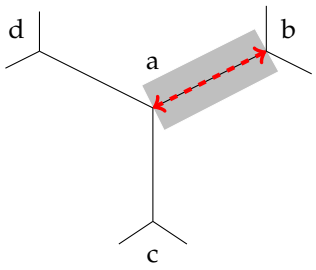
Combined Rules (CR)

$$\nabla_{\mathbf{a}}\widehat{\mathbf{ab}} = \frac{1}{\|ab\|}(\widehat{\mathbf{ab}} \otimes \widehat{\mathbf{ab}} - \mathbf{I}) \tag{CR1}$$

$$\nabla_{\mathbf{b}}\widehat{\mathbf{ab}} = \frac{1}{\|ab\|}(\mathbf{I} - \widehat{\mathbf{ab}} \otimes \widehat{\mathbf{ab}}) \tag{CR2}$$

$$(\nabla_{\mathbf{a}}\widehat{\mathbf{ab}}) \cdot \widehat{\mathbf{ac}} = \frac{1}{\|ab\|}(\widehat{\mathbf{ab}} \otimes (\widehat{\mathbf{ab}} \cdot \widehat{\mathbf{ac}}) - \widehat{\mathbf{ac}}) = \frac{1}{\|ab\|}(\widehat{\mathbf{ab}}\cos(\theta) - \widehat{\mathbf{ac}}) \tag{CR3}$$

A.1 LENGTH CALCULATIONS



Vector $\mathbf{ab} = \{x_b - x_a\,,\, y_b - y_a\,,\, z_b - z_a\}$

$$\|\mathbf{ab}\| = \sqrt{(x_b - x_a)^2 + (y_b - y_a)^2 + (z_b - z_a)^2}$$

$$\frac{\partial}{\partial x_a}(\|\mathbf{ab}\| - R_0)^2 =$$

$$2(\|\mathbf{ab}\| - R_0)\frac{\partial}{\partial x_a}\|\mathbf{ab}\| =$$

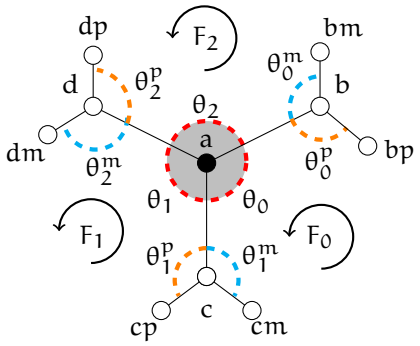
$$2(\|\mathbf{ab}\| - R_0)\frac{\partial}{\partial x_a}\sqrt{(x_b - x_a)^2 + (y_b - y_a)^2 + (z_b - z_a)^2} =$$

$$-2(\|\mathbf{ab}\| - R_0)\frac{x_b - x_a}{\sqrt{(x_b - x_a)^2 + (y_b - y_a)^2 + (z_b - z_a)^2}}$$

This expression can be generalized to:

$$\frac{\partial}{\partial \mathbf{a}}(\|\mathbf{ab}\| - R_0)^2 = -2(\|\mathbf{ab}\| - R_0)\frac{\mathbf{ab}}{\|\mathbf{ab}\|} = \underline{-2(\|\mathbf{ab}\| - R_0)\widehat{\mathbf{ab}}}$$

A.2 CORNER ANGLE CALCULATIONS



Vectors $\{\mathbf{ab}, \mathbf{ac}\}$ and their unit vectors $\{\widehat{\mathbf{ab}} = \frac{\mathbf{ab}}{||ab||}, \widehat{\mathbf{ac}} = \frac{\mathbf{ac}}{||ac||}\}$

Find the gradient of Inner angles θ

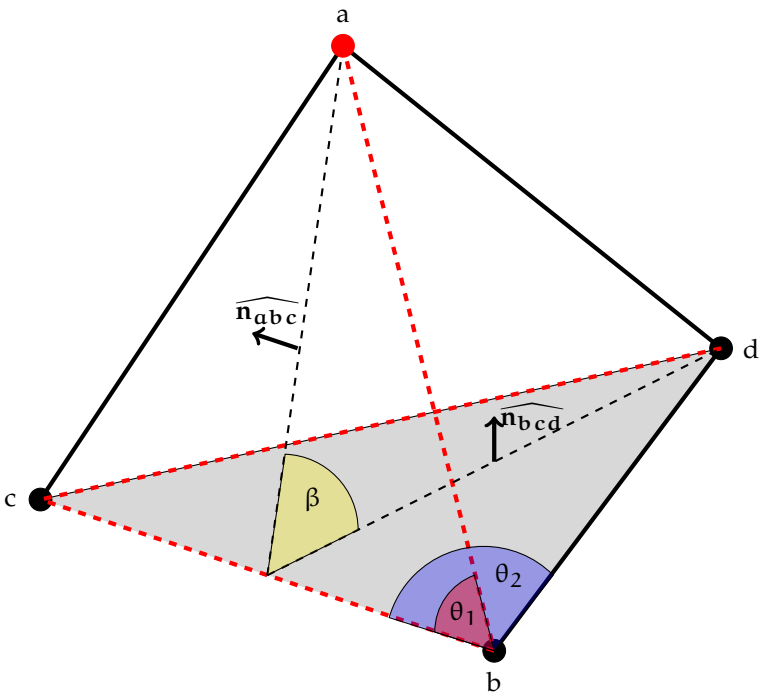
$$\begin{aligned}
\nabla_a(\widehat{\mathbf{ab}} \cdot \widehat{\mathbf{ac}}) &\overset{PR1}{=} (\nabla_a \widehat{\mathbf{ab}}) \cdot \widehat{\mathbf{ac}} + \widehat{\mathbf{ab}} \cdot (\nabla_a \widehat{\mathbf{ac}}) \\
&\overset{CR1}{=} \frac{1}{||ab||}(\widehat{ab}) \otimes \widehat{ab} - \mathbf{I}) \cdot \widehat{\mathbf{ac}} - \frac{1}{||ac||}(\widehat{ac}) \otimes \widehat{ac} - \mathbf{I}) \cdot \widehat{\mathbf{ab}} \\
&= \boxed{\frac{1}{||ab||}(\widehat{\mathbf{ab}} \cos(\theta) - \widehat{\mathbf{ac}}) + \frac{1}{||ac||}(\widehat{\mathbf{ac}} \cos(\theta) - \widehat{\mathbf{ab}})}
\end{aligned}$$

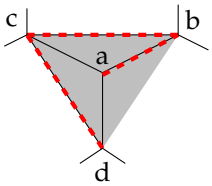
Find the gradient of Outer angles θ^m and θ^p

$$\begin{aligned}
\nabla_a(\widehat{\mathbf{ab}} \cdot \widehat{\mathbf{bp}}) &\overset{PR1}{=} (\nabla_a \widehat{\mathbf{ab}}) \cdot \widehat{\mathbf{bp}} + \widehat{\mathbf{ab}} \cdot \cancelto{0}{(\nabla_a \widehat{\mathbf{bp}})} \\
&\overset{CR1}{=} \frac{1}{||ab||}(\widehat{ab}) \otimes \widehat{ab} - \mathbf{I}) \cdot \widehat{\mathbf{bp}} \\
&= \boxed{\frac{1}{||ab||}(\widehat{\mathbf{ab}} \cos(\theta_0^p) - \widehat{\mathbf{bp}})}
\end{aligned}$$

$$\begin{aligned}
\nabla_a(\widehat{\mathbf{ab}} \cdot \widehat{\mathbf{bm}}) &\overset{PR1}{=} (\nabla_a \widehat{\mathbf{ab}}) \cdot \widehat{\mathbf{bm}} + \widehat{\mathbf{ab}} \cdot \cancelto{0}{(\nabla_a \widehat{\mathbf{bm}})} \\
&\overset{CR1}{=} \frac{1}{||ab||}(\widehat{ab}) \otimes \widehat{ab} - \mathbf{I}) \cdot \widehat{\mathbf{bm}} \\
&= \boxed{\frac{1}{||ab||}(\widehat{\mathbf{ab}} \cos(\theta_0^m) - \widehat{\mathbf{bm}})}
\end{aligned}$$

A.3 DIHEDRAL CALCULATIONS





$$\cos(\beta) = \widehat{\mathbf{n}}_{abc} \cdot \widehat{\mathbf{n}}_{bcd} = \left(\frac{\widehat{\mathbf{ab}} \times \widehat{\mathbf{cb}}}{\sin(\theta_1)}\right) \cdot \left(\frac{\widehat{\mathbf{db}} \times \widehat{\mathbf{cb}}}{\sin(\theta_2)}\right) \tag{46}$$

Find the gradient: $\nabla_{\mathbf{a}} \cos(\beta)$

$$\begin{aligned}
\nabla_{\mathbf{a}} \cos(\beta) &= \nabla_{\mathbf{a}} \left(\widehat{\mathbf{n}}_{abc} \cdot \widehat{\mathbf{n}}_{bcd} \right) && \text{PR1} \\
&= (\nabla_{\mathbf{a}} \widehat{\mathbf{n}}_{abc}) \cdot \widehat{\mathbf{n}}_{bcd} + \widehat{\mathbf{n}}_{abc} \cdot \cancelto{0}{(\nabla_{\mathbf{a}} \widehat{\mathbf{n}}_{bcd})} \\
&= (\nabla_{\mathbf{a}} \left(\frac{\widehat{\mathbf{ab}} \times \widehat{\mathbf{cb}}}{\sin(\theta_1)} \right)) \cdot \widehat{\mathbf{n}}_{bcd} && \text{QR} \\
\mathbf{g}(\mathbf{a}) &= \widehat{\mathbf{ab}} \times \widehat{\mathbf{cb}} \quad , \quad h(\mathbf{a}) = \sin(\theta_1) = \sqrt{1 - \cos^2(\theta_1)}
\end{aligned}$$

$$\begin{aligned}
\nabla_{\mathbf{a}} \otimes \mathbf{g}(\mathbf{a}) &= \nabla_{\mathbf{a}} (\widehat{\mathbf{ab}} \times \widehat{\mathbf{cb}}) && \text{(PR2)} \\
&= (\nabla_{\mathbf{a}} \widehat{\mathbf{ab}}) \times \widehat{\mathbf{cb}} + \widehat{\mathbf{ab}} \times \cancelto{0}{(\nabla_{\mathbf{a}} \widehat{\mathbf{cb}})} \\
&= \frac{1}{\|ab\|} (\mathbf{I} - \widehat{\mathbf{ab}} \otimes \widehat{\mathbf{ab}}) \times \widehat{\mathbf{cb}}
\end{aligned}$$

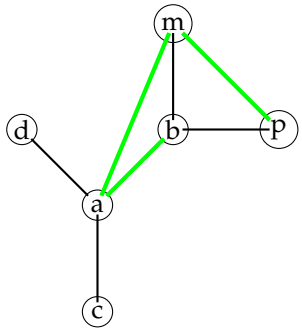
$$\begin{aligned}\nabla_{\mathbf{a}} h(\mathbf{a}) &= \nabla_{\mathbf{a}} \sqrt{1-(\widehat{\mathbf{ab}}\cdot\widehat{\mathbf{cb}})} \\ &= \frac{\cos(\theta_1)}{\sin(\theta_1)} \nabla_{\mathbf{a}} (\widehat{\mathbf{ab}}\cdot\widehat{\mathbf{cb}}) \qquad \text{(PR1)} + \text{(CR1)} \\ &= \frac{\cos(\theta_1)}{\|ab\|\sin(\theta_1)} (\mathbf{I} - \widehat{\mathbf{ab}}\otimes\widehat{\mathbf{ab}})\cdot\widehat{\mathbf{cb}}\end{aligned}$$

Combine and dot with $\hat{\mathbf{n}}_{bcd}$

$$\boxed{\nabla_{\mathbf{a}}\cos(\beta) = \frac{-1}{\|ab\|}\left(\widehat{\mathbf{ab}}\cos(\beta) + \frac{\widehat{\mathbf{cb}}\times\hat{\mathbf{n}}_{bcd}}{\sin(\theta_1)} - \frac{\cot(\theta_1)\cos(\beta)}{\sin(\theta_1)}(\widehat{\mathbf{ab}}\cos(\theta_1) - \widehat{\mathbf{cb}})\right)}$$

Find the gradient of Outer dihedral a

$$\begin{aligned}\hat{\mathbf{n}}_{bam} &= \frac{\widehat{\mathbf{ab}}\times\widehat{\mathbf{am}}}{\sqrt{1-(\widehat{\mathbf{ab}}\cdot\widehat{\mathbf{am}})^2}} \\ \hat{\mathbf{n}}_{amp} &= \frac{\widehat{\mathbf{ma}}\times\widehat{\mathbf{mp}}}{\sqrt{1-(\widehat{\mathbf{ma}}\cdot\widehat{\mathbf{mp}})^2}} \\ \cos(\beta) &= \hat{\mathbf{n}}_{bam}\cdot\hat{\mathbf{n}}_{amp} \\ \nabla_{\mathbf{a}}\cos(\beta) &= \underbrace{(\nabla_{\mathbf{a}}\hat{\mathbf{n}}_{bam})\cdot\hat{\mathbf{n}}_{amp}}_{part_1} + \underbrace{\hat{\mathbf{n}}_{bam}\cdot(\nabla_{\mathbf{a}}\hat{\mathbf{n}}_{amp})}_{part_2}\end{aligned}$$



$part_1$:

$$(\nabla_{\mathbf{a}}\left(\frac{\widehat{\mathbf{ab}}\times\widehat{\mathbf{am}}}{\sqrt{1-(\widehat{\mathbf{ab}}\cdot\widehat{\mathbf{am}})^2}}\right))\cdot\hat{\mathbf{n}}_{amp}$$

Using the Quotient Rule we set:

$$\begin{aligned}g(\mathbf{a}) &= \widehat{\mathbf{ab}}\times\widehat{\mathbf{am}} \\ h(\mathbf{a}) &= \sqrt{1-(\widehat{\mathbf{ab}}\cdot\widehat{\mathbf{am}})^2}\end{aligned}$$

$$\begin{aligned}\nabla_{\mathbf{a}} g(\mathbf{a}) &= (\nabla_{\mathbf{a}}\widehat{\mathbf{ab}})\times\widehat{\mathbf{am}} + \widehat{\mathbf{ab}}\times(\nabla_{\mathbf{a}}\widehat{\mathbf{am}}) \\ &= \frac{1}{\|\mathbf{ab}\|}\left(\widehat{\mathbf{ab}}\otimes(\widehat{\mathbf{ab}}\times\widehat{\mathbf{am}}) - \mathbf{I}\times\widehat{\mathbf{am}}\right) + \frac{1}{\|\mathbf{am}\|}\left(\widehat{\mathbf{am}}\otimes(\widehat{\mathbf{ab}}\times\widehat{\mathbf{am}}) - \widehat{\mathbf{ab}}\times\mathbf{I}\right) \\ &= \frac{1}{\|\mathbf{ab}\|}\left(\widehat{\mathbf{ab}}\otimes\hat{\mathbf{n}}_{bam}\sin(\theta_a) - \mathbf{I}\times\widehat{\mathbf{am}}\right) + \frac{1}{\|\mathbf{am}\|}\left(\widehat{\mathbf{am}}\otimes\hat{\mathbf{n}}_{bam}\sin(\theta_a) - \widehat{\mathbf{ab}}\times\mathbf{I}\right)\end{aligned}$$

$$\begin{aligned}\nabla_a h(a) &= \nabla_a \sqrt{1-(\widehat{ab}\cdot\widehat{am})^2} \\ &= \frac{1}{2sin(\theta_a)}\nabla_a(1-(\widehat{ab}\cdot\widehat{am})^2) \\ &= -cot(\theta_a)\nabla_a(\widehat{ab}\cdot\widehat{am}) \\ &= -cot(\theta_a)\left(\frac{1}{\|ab\|}(\widehat{ab}cos(\theta_a)-\widehat{am}) + \frac{1}{\|am\|}(\widehat{am}cos(\theta_a)-\widehat{ab})\right)\end{aligned}$$

$$(\nabla_a\left(\frac{\widehat{ab}\times\widehat{am}}{\sqrt{1-(\widehat{ab}\cdot\widehat{am})^2}}\right))\cdot\hat{n}_{amp} = \frac{\widehat{ab}cos(\beta)}{\|ab\|} - \frac{\widehat{am}\times\hat{n}_{amp}}{\|am\|sin\theta_a} + \frac{\widehat{am}cos(\beta)}{\|am\|} - \frac{\hat{n}_{amp}\times\widehat{ab}}{\|am\|sin(\theta_a)} + \frac{cot(\theta_a cos(\beta))}{sin(\theta_a)}\left(\frac{\widehat{ab}cos(\theta_a)}{\|ab\|} - \frac{\widehat{am}}{\|ab\|} + \frac{\widehat{am}cos(\theta_a)}{\|am\|} - \frac{\widehat{ab}}{\|am\|}\right)$$

$part_2$:

$$\hat{n}_{amp}\cdot(\nabla_a\left(\frac{\widehat{ma}\times\widehat{mp}}{\sqrt{1-(\widehat{ma}\cdot\widehat{mp})^2}}\right))$$

Using the Quotient Rule we set:

$$g(a) = \widehat{ma}\times\widehat{mp}$$
$$h(a) = \sqrt{1-(\widehat{ma}\cdot\widehat{mp})^2}$$

$$\begin{aligned}\nabla_a g(a) &= (\nabla_a\widehat{ma})\times\widehat{mp} + \widehat{ma}\times\cancelto{0}{(\nabla_a\widehat{mp})} \\ &= \frac{1}{\|ma\|}(I\times\widehat{mp} - \widehat{ma}\otimes\hat{n}_{amp}sin(\theta_m))\end{aligned}$$

$$\begin{aligned}\nabla_a h(a) &= \nabla_a\sqrt{1-(\widehat{ma}\cdot\widehat{mp})^2} \\ &= -cot(\theta_m)\nabla_a(\widehat{ma}\cdot\widehat{mp}) \\ &= -\frac{cot(\theta_m)}{\|ma\|}(\widehat{mp}-\widehat{ma}cos(\theta_m))\end{aligned}$$

$$\hat{n}_{amp}\cdot(\nabla_a\left(\frac{\widehat{ma}\times\widehat{mp}}{\sqrt{1-(\widehat{ma}\cdot\widehat{mp})^2}}\right)) = \frac{\widehat{mp}\times\hat{n}_{amp}}{\|ma\|sin(\theta_m)} - \frac{\widehat{ma}cos(\beta)}{\|ma\|} + \frac{cot(\theta_m)cos(\beta)}{\|ma\|sin(\theta_m)}(\widehat{mp}-\widehat{ma}cos(\theta_m))$$

$part_1 + part_2$

$$\begin{aligned}\nabla_a \cos(\beta) = &\frac{\widehat{\mathbf{ab}}\cos(\beta)}{\|\mathbf{ab}\|} - \frac{\widehat{\mathbf{am}} \times \hat{\mathbf{n}}_{amp}}{\|\mathbf{am}\|\sin\theta_a} + \frac{\widehat{\mathbf{am}}\cos(\beta)}{\|\mathbf{am}\|} - \frac{\hat{\mathbf{n}}_{amp} \times \widehat{\mathbf{ab}}}{\|\mathbf{am}\|\sin(\theta_a)} + \\ &\frac{\cot(\theta_a\cos(\beta))}{\sin(\theta_a)}\left(\frac{\widehat{\mathbf{ab}}\cos(\theta_a)}{\|\mathbf{ab}\|} - \frac{\widehat{\mathbf{am}}}{\|\mathbf{ab}\|} + \frac{\widehat{\mathbf{am}}\cos(\theta_a)}{\|\mathbf{am}\|} - \frac{\widehat{\mathbf{ab}}}{\|\mathbf{am}\|}\right) + \\ &\frac{\widehat{\mathbf{mp}} \times \hat{\mathbf{n}}_{amp}}{\|\mathbf{ma}\|\sin(\theta_m)} - \frac{\widehat{\mathbf{ma}}\cos(\beta)}{\|\mathbf{ma}\|} + \\ &\frac{\cot(\theta_m)\cos(\beta)}{\|\mathbf{ma}\|\sin(\theta_m)}(\widehat{\mathbf{mp}} - \widehat{\mathbf{ma}}\cos(\theta_m))\end{aligned} \tag{47}$$

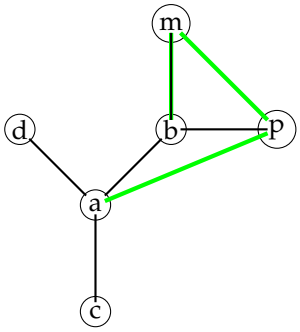
Find the gradient of Outer dihedral m

$$\hat{n}_{bmp} = \frac{\widehat{mb} \times \widehat{mp}}{\sqrt{1-(\widehat{mb}\cdot\widehat{mp})^2}}$$

$$\hat{n}_{mpa} = \frac{\widehat{pm} \times \widehat{pa}}{\sqrt{1-(\widehat{pm}\cdot\widehat{pa})^2}}$$

$$\cos(\beta) = \hat{n}_{bmp}\cdot\hat{n}_{mpa}$$

$$\nabla_a \cos(\beta) = \underbrace{\cancelto{0}{(\nabla_a \hat{n}_{bmp})\cdot\hat{n}_{mpa}}}_{part_1} + \underbrace{\hat{n}_{bmp}\cdot(\nabla_a \hat{n}_{mpa})}_{part_2}$$



$part_2$

$$\hat{n}_{bmp}\cdot(\nabla_a \frac{\widehat{pm} \times \widehat{pa}}{\sqrt{1-(\widehat{pm}\cdot\widehat{pa})^2}})$$

Using the Quotient Rule we set:

$$g(a) = \widehat{pm} \times \widehat{pa}$$

$$h(a) = \sqrt{1-(\widehat{pm}\cdot\widehat{pa})^2}$$

$$\begin{aligned}\nabla_a g(a) &= \cancelto{0}{(\nabla_a \widehat{pm})} \times \widehat{pa} + \widehat{pm} \times (\nabla_a \widehat{pa}) \\ &= \frac{1}{\|pa\|}(\widehat{pm} \times I - \widehat{pa} \otimes \hat{n}_{mpa}\sin(\theta_p))\end{aligned}$$

$$\begin{aligned}\nabla_a h(a) &= \nabla_a \sqrt{1-(\widehat{pm}\cdot\widehat{pa})^2} \\ &= -\cot(\theta_p)\nabla_a(\widehat{pm}\cdot\widehat{pa}) \\ &= -\frac{-\cot(\theta_p)}{\|pa\|}(\widehat{pm} - \widehat{pa}\cos(\theta_p))\end{aligned}$$

$$\begin{aligned}\nabla_a \cos(\beta) =& \frac{\hat{n}_{bmp} \times \widehat{pm}}{\|pa\|\sin(\theta_p)} - \frac{\widehat{pa}\cos(\beta)}{\|pa\|} + \\ & \frac{\cot(\theta_p)\cos(\beta)}{\|pa\|\sin(\sin(\theta_p))}(\widehat{pm} - \widehat{pa}\cos(\theta_p))\end{aligned} \tag{48}$$

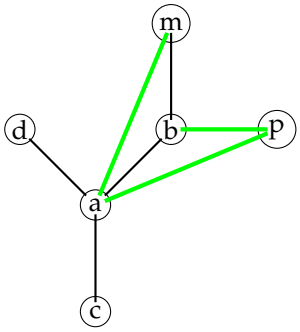
Find the gradient of Outer dihedral p

$$\hat{\mathbf{n}}_{bpa} = \frac{\widehat{\mathbf{pb}} \times \widehat{\mathbf{pa}}}{\sqrt{1-(\widehat{\mathbf{pb}} \cdot \widehat{\mathbf{pa}})^2}}$$

$$\hat{\mathbf{n}}_{pam} = \frac{\widehat{\mathbf{ap}} \times \widehat{\mathbf{am}}}{\sqrt{1-(\widehat{\mathbf{ap}} \cdot \widehat{\mathbf{am}})^2}}$$

$$\cos(\beta) = \hat{\mathbf{n}}_{bpa} \cdot \hat{\mathbf{n}}_{pam}$$

$$\nabla_{\mathbf{a}} \cos(\beta) = \underbrace{(\nabla_{\mathbf{a}} \hat{\mathbf{n}}_{bpa}) \cdot \hat{\mathbf{n}}_{pam}}_{\text{part}_1} + \underbrace{\hat{\mathbf{n}}_{bpa} \cdot (\nabla_{\mathbf{a}} \hat{\mathbf{n}}_{pam})}_{\text{part}_2}$$



part_1

$$\nabla_{\mathbf{a}} \left(\frac{\widehat{\mathbf{pb}} \times \widehat{\mathbf{pa}}}{\sqrt{1-(\widehat{\mathbf{pb}} \cdot \widehat{\mathbf{pa}})^2}} \right) \cdot \hat{\mathbf{n}}_{pam}$$

Using the Quotient Rule we set:

$$g(\mathbf{a}) = \widehat{\mathbf{pb}} \times \widehat{\mathbf{pa}}$$

$$h(\mathbf{a}) = \sqrt{1-(\widehat{\mathbf{pb}} \cdot \widehat{\mathbf{pa}})^2}$$

$$\nabla_{\mathbf{a}} g(\mathbf{a}) = \cancelto{0}{(\nabla_{\mathbf{a}} \widehat{\mathbf{pb}}) \times \widehat{\mathbf{pa}}} + \widehat{\mathbf{pb}} \times (\nabla_{\mathbf{a}} \widehat{\mathbf{pa}})$$

$$= \frac{1}{\|\mathbf{pa}\|} (\widehat{\mathbf{pb}} \times \mathbf{I} - \widehat{\mathbf{pa}} \otimes \hat{\mathbf{n}}_{bpa} \sin(\theta_p))$$

$$\nabla_{\mathbf{a}} h(\mathbf{a}) = \nabla_{\mathbf{a}} \sqrt{1-(\widehat{\mathbf{pb}} \cdot \widehat{\mathbf{pa}})^2}$$

$$= -\cot(\theta_p) \nabla_{\mathbf{a}} (\widehat{\mathbf{pb}} \cdot \widehat{\mathbf{pa}})$$

$$= \frac{-\cot(\theta_p)}{\|\mathbf{pa}\|} (\widehat{\mathbf{pb}} - \widehat{\mathbf{pa}} \cos(\theta_p))$$

$$\nabla_{\mathbf{a}} \left(\frac{\widehat{\mathbf{pb}} \times \widehat{\mathbf{pa}}}{\sqrt{1-(\widehat{\mathbf{pb}} \cdot \widehat{\mathbf{pa}})^2}} \right) \cdot \hat{\mathbf{n}}_{pam} = \frac{\hat{\mathbf{n}}_{pam} \times \widehat{\mathbf{pb}}}{\|\mathbf{pa}\| \sin(\theta_p)} - \frac{\widehat{\mathbf{pa}} \cos(\beta)}{\|\mathbf{pa}\|}$$

$$+ \frac{\cot(\theta_p)\cos(\beta)}{\|\mathbf{pa}\| \sin(\theta_p)} (\widehat{\mathbf{pb}} - \widehat{\mathbf{pa}} \cos(\theta_p))$$

$part_2$

$$\nabla_a \hat{n}_{bpa} \cdot (\nabla_a \frac{\widehat{ap} \times \widehat{am}}{\sqrt{1-(\widehat{ap} \cdot \widehat{am})^2}})$$

Using the Quotient Rule we set:

$$g(a) = \widehat{ap} \times \widehat{am}$$
$$h(a) = \sqrt{1-(\widehat{ap} \cdot \widehat{am})^2}$$

$$\begin{aligned}\nabla_a g(a) &= \nabla_a(\widehat{ap} \times \widehat{am}) \\ &= (\nabla_a \widehat{ap}) \times \widehat{am} + \widehat{ap} \times (\nabla_a \widehat{am}) \\ &= \frac{1}{\|ap\|}(\widehat{ap} \otimes \hat{n}_{pam} sin(\theta_a) - \mathbf{I} \times \widehat{am}) + \frac{1}{\|am\|}(\widehat{am} \otimes \hat{n}_{pam} sin(\theta_a) - \widehat{ap} \times \mathbf{I})\end{aligned}$$

$$\begin{aligned}\nabla_a h(a) &= \nabla_a \sqrt{1-(\widehat{ap} \cdot \widehat{am})^2} \\ &= -cot(\theta_a)\nabla_a(\widehat{ap} \cdot \widehat{am}) \\ &= -cot(\theta_a)\left(\frac{1}{\|ap\|}(\widehat{ap} cos(\theta_a) - \widehat{am}) + \frac{1}{\|am\|}(\widehat{am} cos(\theta_a) - \widehat{ap})\right)\end{aligned}$$

$$\begin{aligned}\nabla_a \hat{n}_{bpa} \cdot (\nabla_a \frac{\widehat{ap} \times \widehat{am}}{\sqrt{1-(\widehat{ap} \cdot \widehat{am})^2}}) = &\frac{\widehat{ap} cos(\beta)}{\|ap\|} - \frac{\widehat{am} \times \hat{n}_{bpa}}{\|ap\| sin(\theta_a)} + \\ &\frac{\widehat{am} cos(\beta)}{\|am\|} - \frac{\hat{n}_{bpa} \times \widehat{ap}}{\|am\| sin(\theta_a)} + \\ &\frac{cot(\theta_a) cos(\beta)}{sin(\theta_a)}(\frac{\widehat{ap} cos(\theta_a)}{\|ap\|} - \frac{\widehat{am}}{\|ap\|} + \frac{\widehat{am} cos(\theta_a)}{\|am\|} - \frac{\widehat{ap}}{\|am\|})\end{aligned}$$

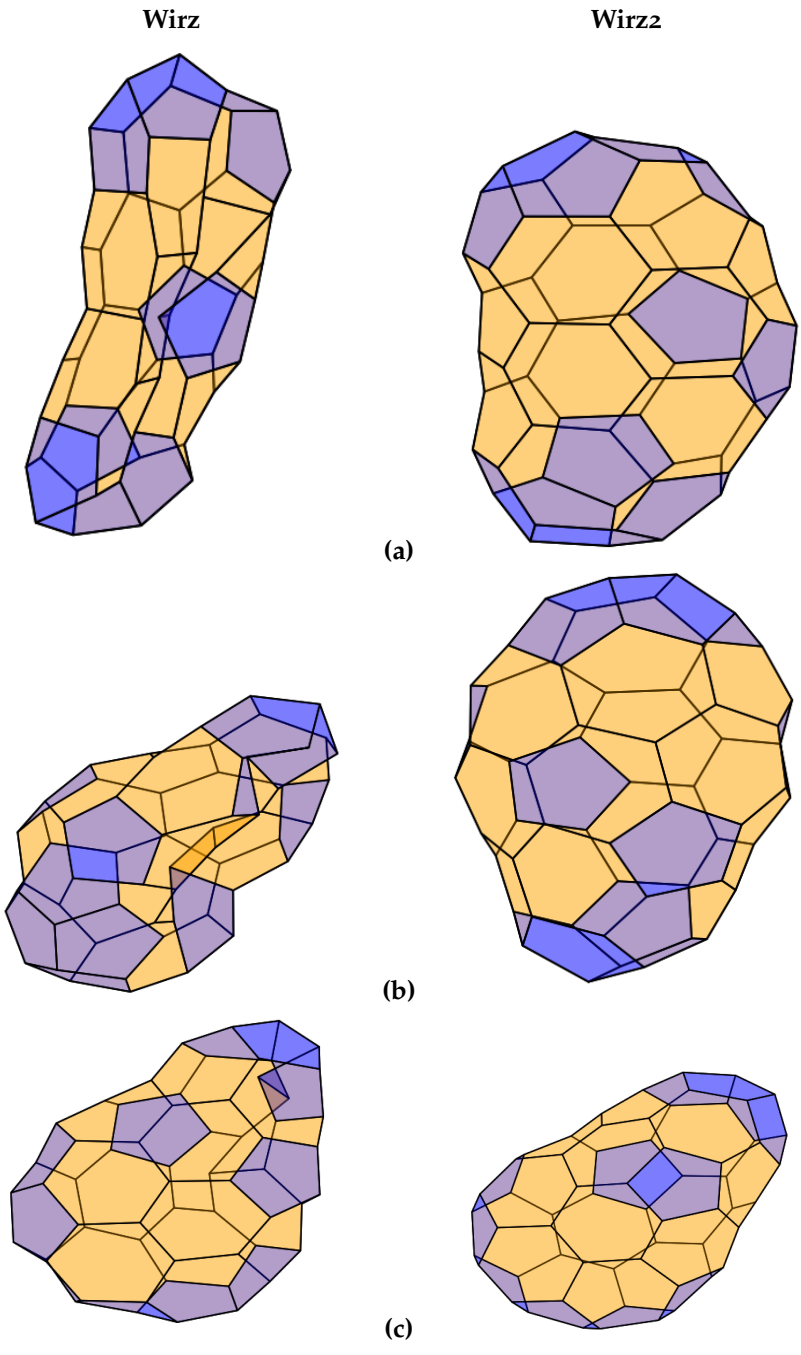
$$\begin{aligned}\nabla_a cos(\beta) = &\frac{\hat{n}_{pam} \times \widehat{pb}}{\|pa\| sin(\theta_p)} - \frac{\widehat{pa} cos(\beta)}{\|pa\|} + \frac{cot(\theta_p) cos(\beta)}{\|pa\| sin(\theta_p)}(\widehat{pb} - \widehat{pa} cos(\theta_p)) \\ &\frac{\widehat{ap} cos(\beta)}{\|ap\|} - \frac{\widehat{am} \times \hat{n}_{bpa}}{\|ap\| sin(\theta_a)} + \frac{\widehat{am} cos(\beta)}{\|am\|} - \frac{\hat{n}_{bpa} \times \widehat{ap}}{\|am\| sin(\theta_a)} + \\ &\frac{cot(\theta_a) cos(\beta)}{sin(\theta_a)}(\frac{\widehat{ap} cos(\theta_a)}{\|ap\|} - \frac{\widehat{am}}{\|ap\|} + \frac{\widehat{am} cos(\theta_a)}{\|am\|} - \frac{\widehat{ap}}{\|am\|})\end{aligned}$$

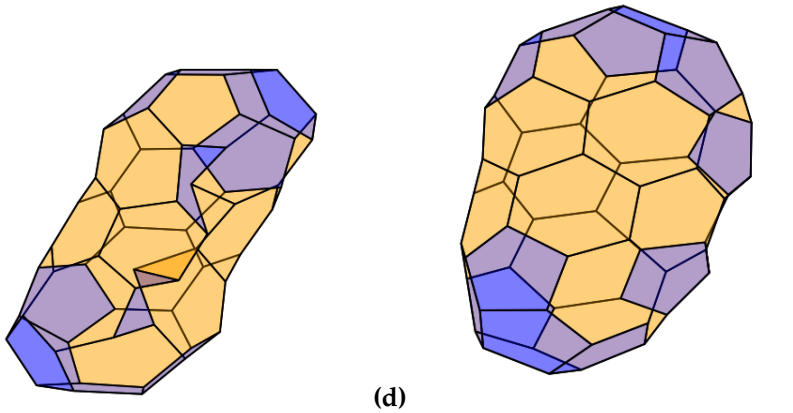
(49)

B

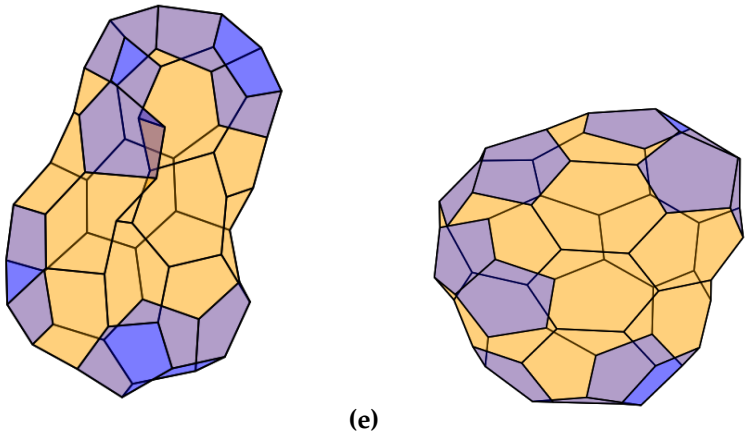
WIRZ INACCURACIES

B.1 FAILED C60 FULLERENES

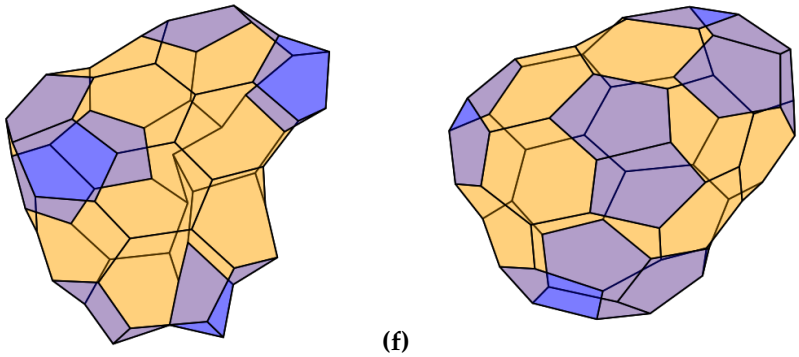




(d)



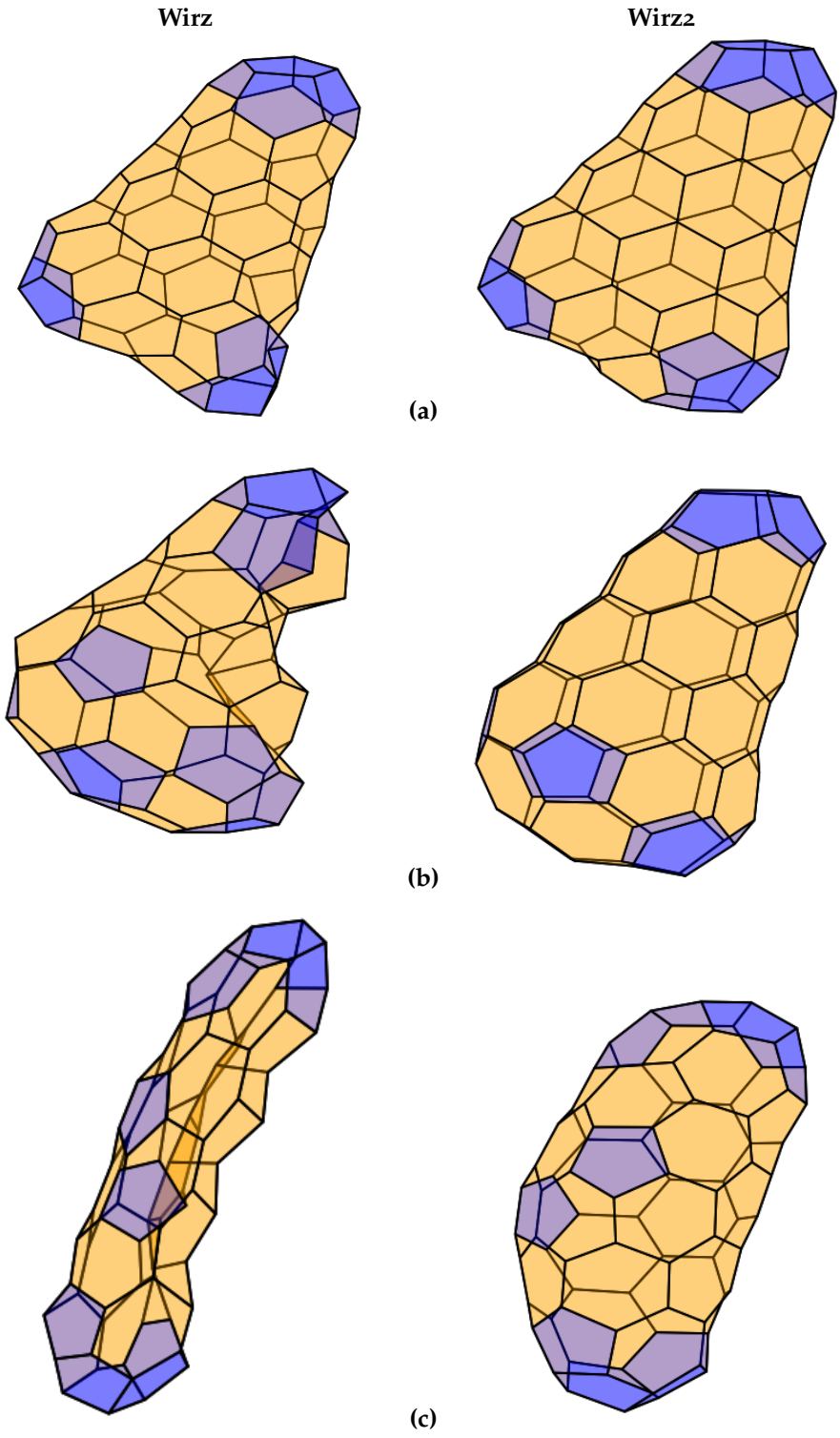
(e)

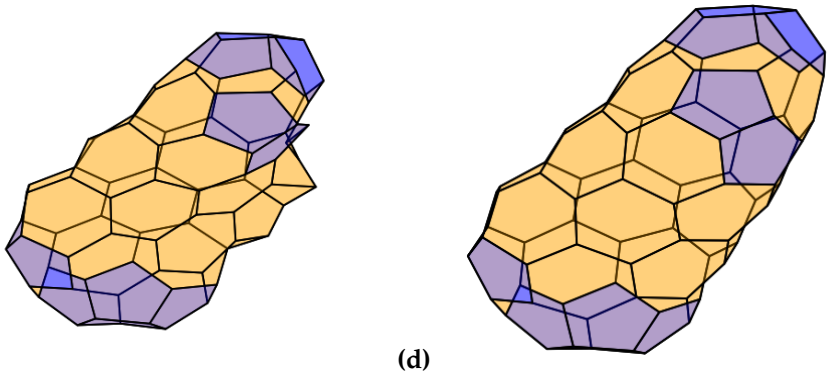


(f)

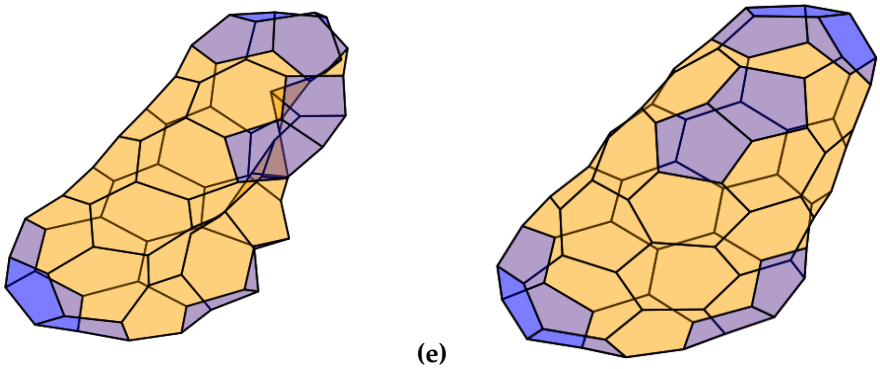
Figure 33: Comparisson of failed C60 Wirz optimized fullerene geometries (column 1) together with the Wirz2 optimized geometries(column 2). **(a)** C60-[1,2,3,4,7,20,22,25,26,27,30,31], **(b)** C60-[1,2,3,4,11,15,20,24,28,29,31,32], **(c)** C60-[1,2,3,4,11,15,21,25,28,29,31,32], **(d)** C60-[1,2,3,5,7,10,23,26,28,30,31,32], **(e)** C60-[1,2,3,5,10,18,22,23,27,28,30,31], **(f)** C60-[1,2,3,5,18,19,21,24,25,29,31,32].

B.2 FAILED C80 FULLERENES

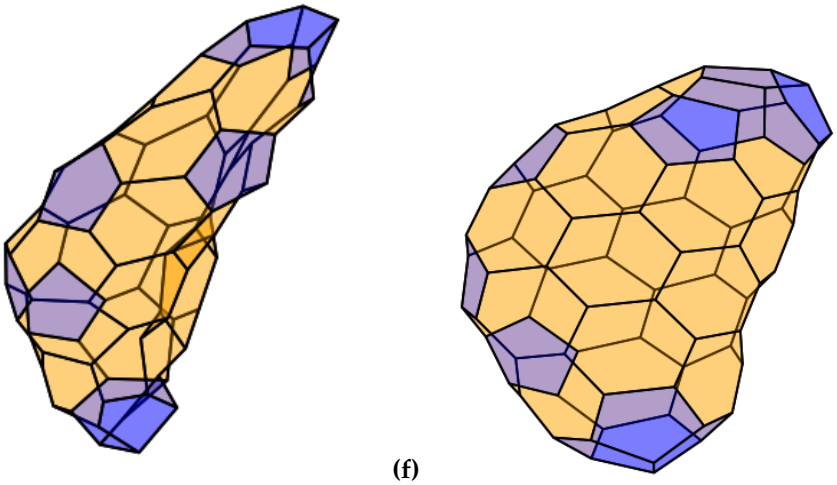




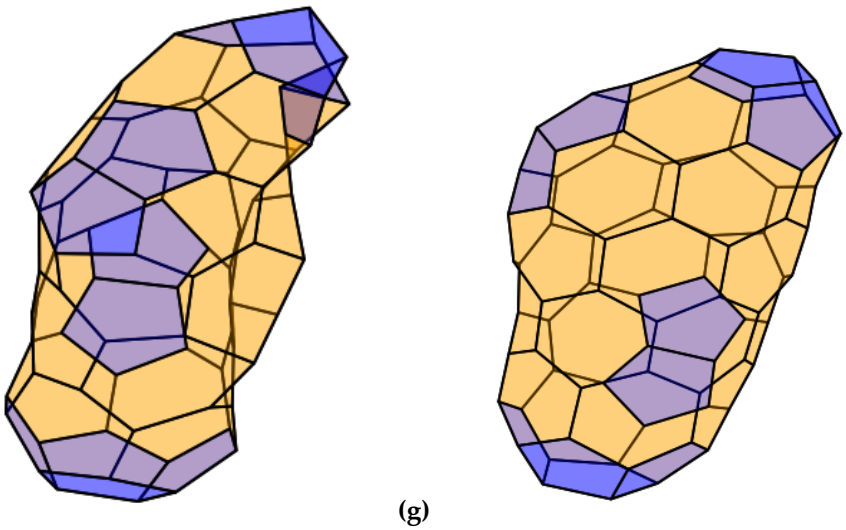
(d)



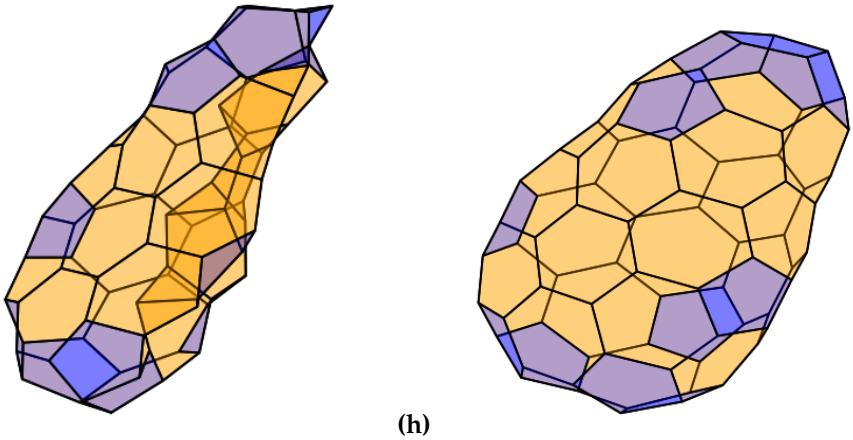
(e)



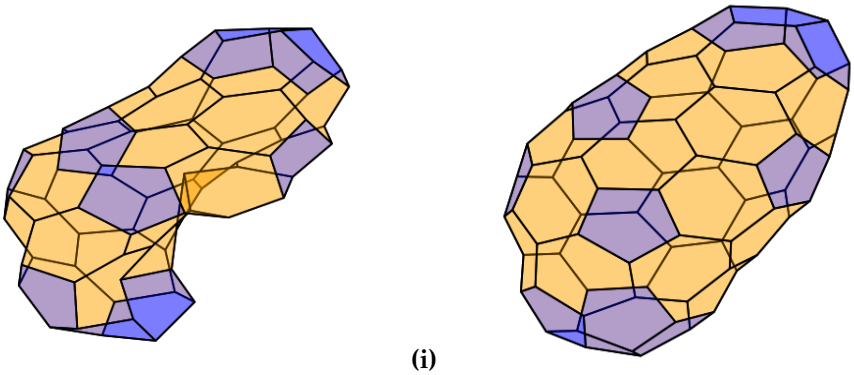
(f)



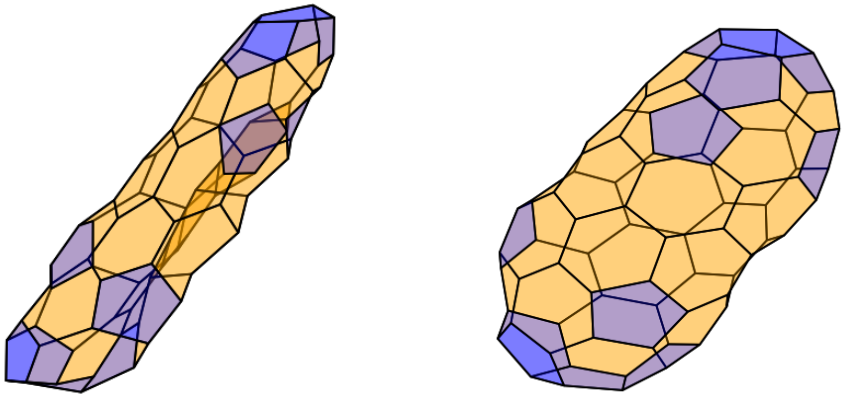
(g)



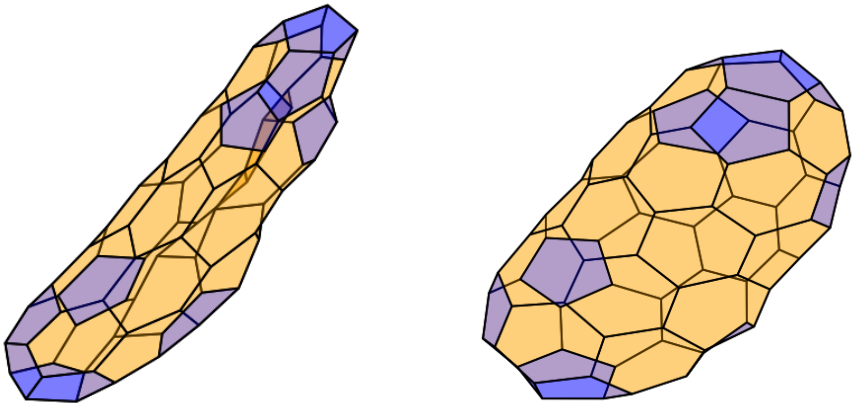
(h)



(i)



(j)



(k)

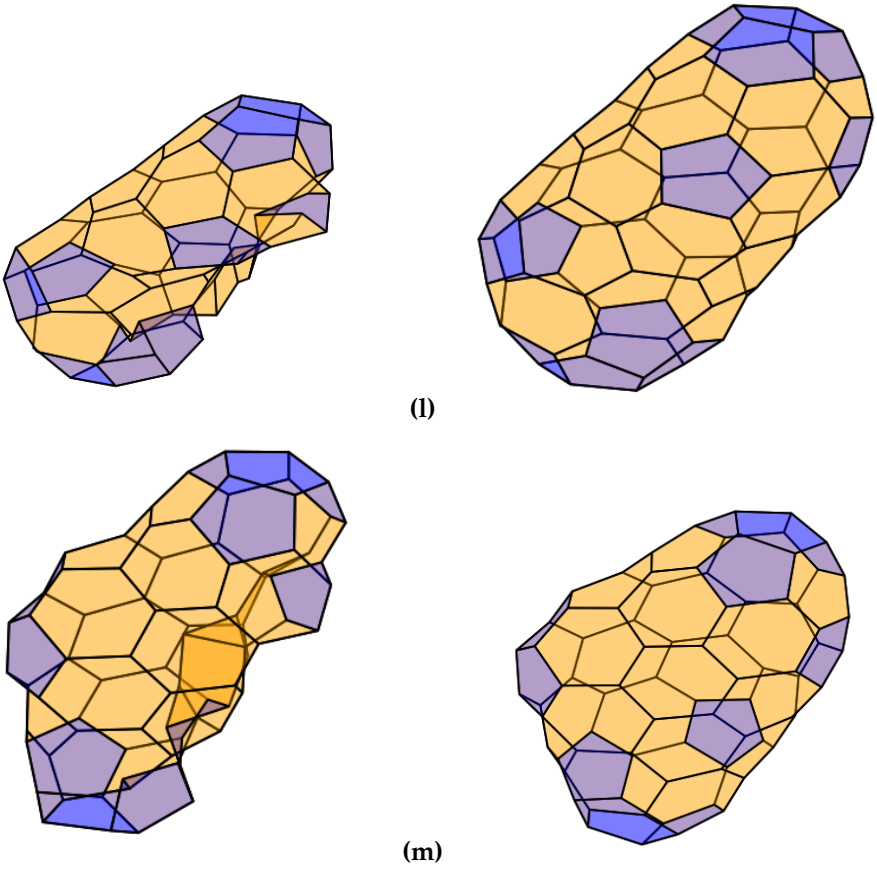


Figure 34: Comparisson of failed C80 Wirz optimized fullerene geometries (column 1) together with the Wirz2 optimized geometries(column 2). **(a)** C80-[1,2,3,4,5,27,28,35,38,39,40,42], **(b)** C80-[1,2,3,4,5,27,29,34,36,38,39,40], **(c)** C80-[1,2,3,4,7,19,29,34,38,39,40,42], **(d)** C80-[1,2,3,4,11,18,29,36,38,39,40,42], **(e)** C80-[1,2,3,4,11,19,30,33,37,38,40,42], **(f)** C80-[1,2,3,4,11,20,29,31,39,40,41,42], **(g)** C80-[1,2,3,4,12,20,31,36,38,40,41,42], **(h)** C80-[1,2,3,4,12,22,27,33,36,38,39,41], **(i)** C80-[1,2,3,4,14,18,28,30,39,40,41,42], **(j)** C80-[1,2,3,5,10,15,29,34,37,39,41,42], **(k)** C80-[1,2,3,5,10,15,30,33,36,40,41,42], **(l)** C80-[1,2,3,5,10,19,29,31,37,39,40,42], **(m)** C80-[1,2,3,5,10,21,29,33,38,40,41,42].

C

GAUSSIAN RESULTS OF C60Ih

C.1 PH BOND STRECHING FORCE CONSTANTS

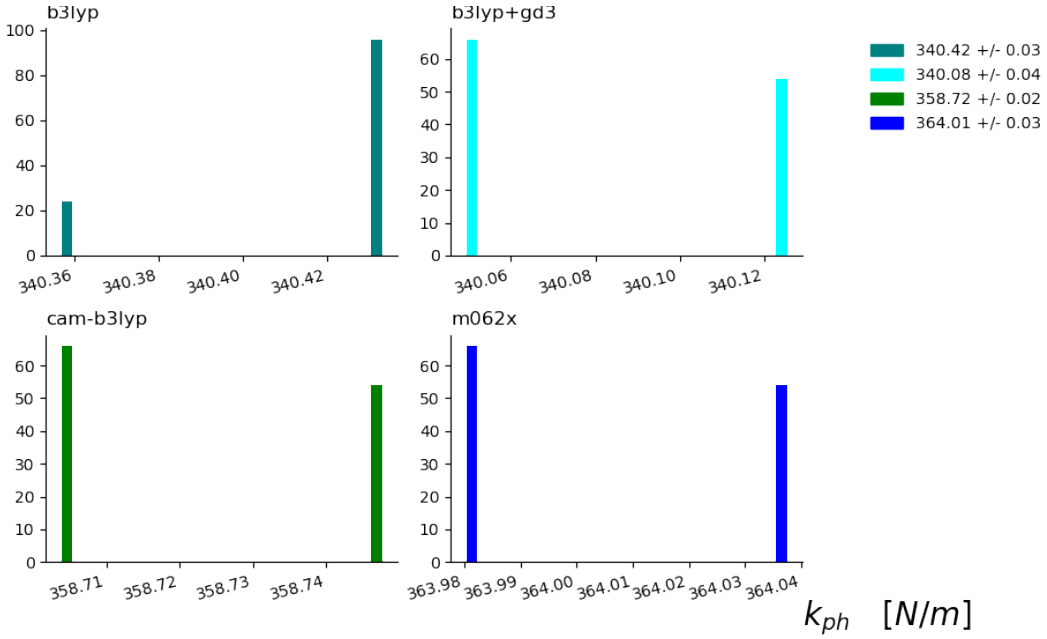


Figure 35: Histograms of k_{ph} force constants from various DFT methods.

C.2 HH BOND STRECHING FORCE CONSTANTS

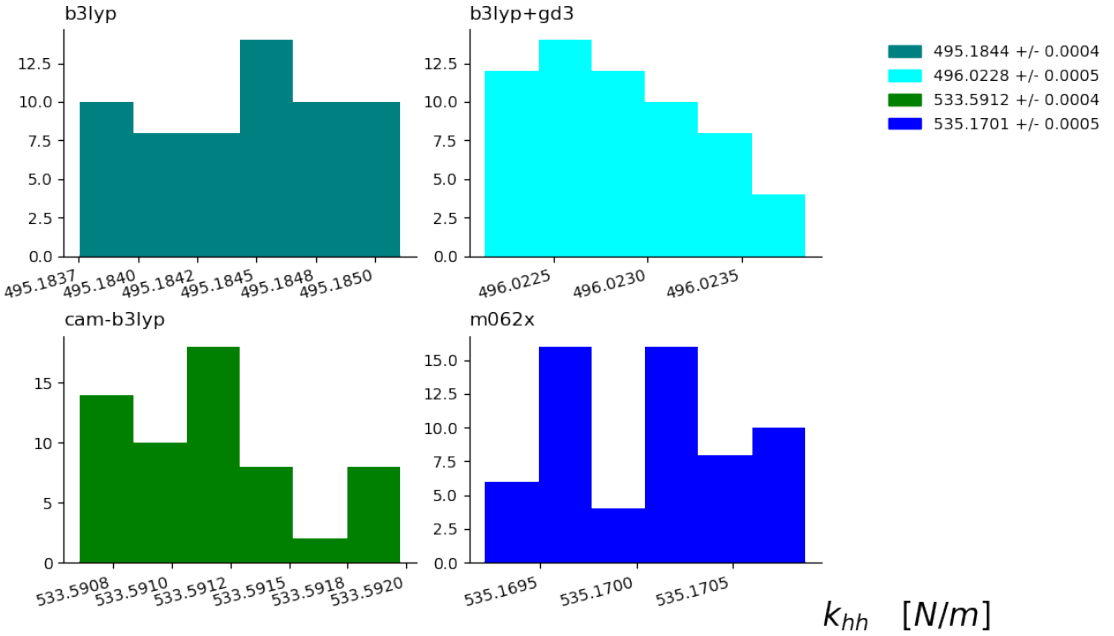


Figure 36: Histograms of k_{hh} force constants from various DFT methods.

C.3 P ANGLE BENDING FORCE CONSTANTS

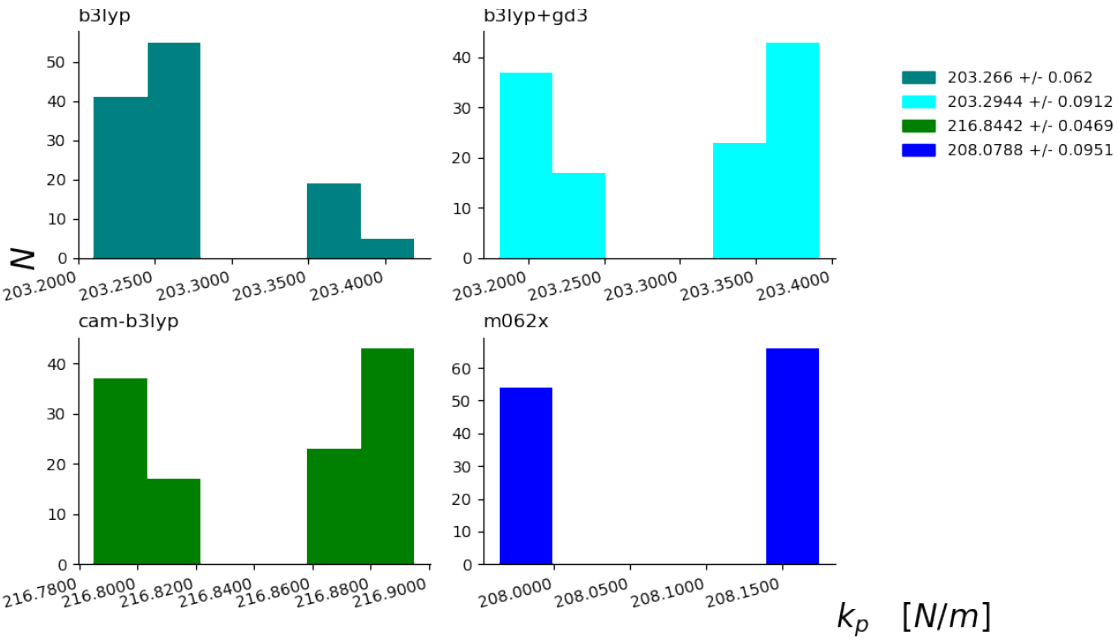


Figure 37: Histograms of k_p force constants from various DFT methods.

C.4 H ANGLE BENDING FORCE CONSTANTS

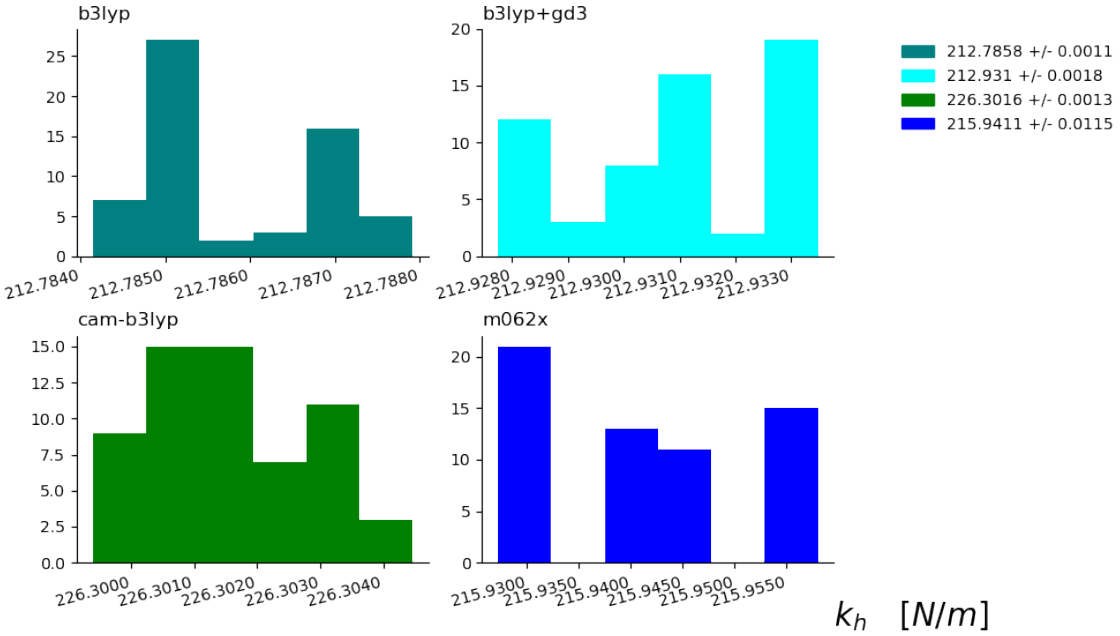


Figure 38: Histograms of k_p force constants from various DFT methods.

C.5 PHH DIHEDRAL BENDING FORCE CONSTANTS

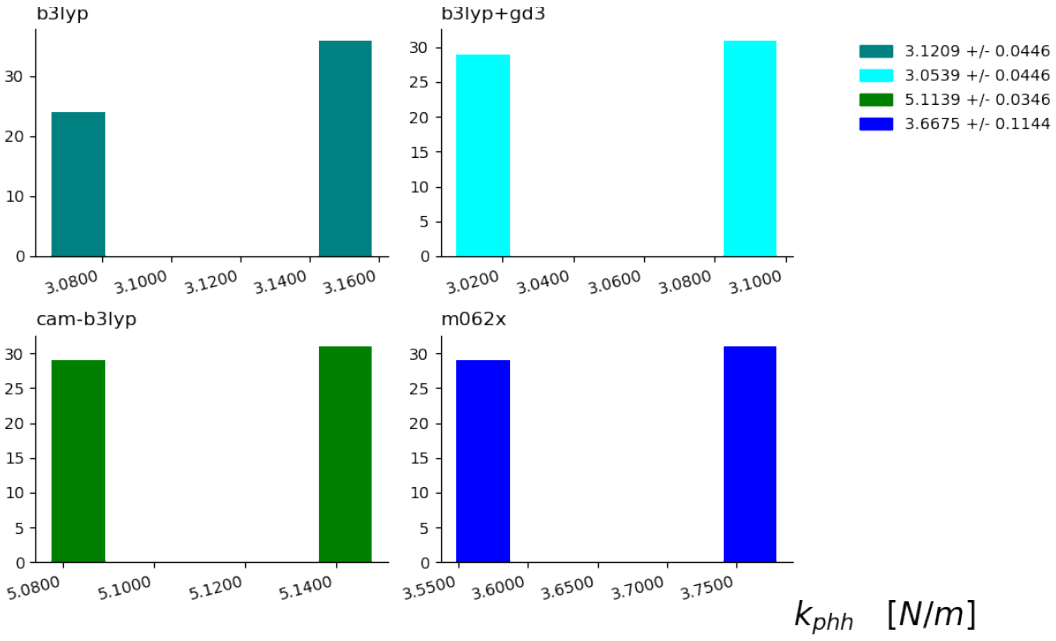


Figure 39: Histograms of k_p force constants from various DFT methods.

ACKNOWLEDGEMENTS

I want to thank James Avery for his perservance and patience when confronted with my endless stream of fullerene questions. It has been a interesting journey through physics, geometry, chemistry and computer science with you at the steering wheel.
I would like to thank Kurt Nielsen for his introduction and insight into the *Gaussian16* program.
I also want to thank my collegues at eScience for their enjoyable company, good memories, and interesting discussions.
Lastly, I would like to thank all of my friends and especially those whom have proofread my thesis, teaching me a lot about correctly spelling words and English punctuation.