Nano-κ : a Python code for the multiscale modelling of the thermal conductivity

Bruno Hartmann da Silva

THESIS

by Bruno HARTMANN DA SILVA

submitted as requirement for the degree of

Doctor of Philosophy (PhD)

in Mechanics and Energetics

awarded by

Doctoral School n°608

Sciences et Ingénierie des Molécules, des Produits, des Procédés et de l'Énergie (SIMPPÉ)

Nano-κ: a Python code for the multiscale modelling of the thermal conductivity

defended on December 5th 2023
before the committee composed by

Laurent CHAPUT	Professor, Université de Lorraine	Supervisor
David LACROIX	Professor, Université de Lorraine	Co-supervisor
Jérôme SAINT-MARTIN	Professor, ENS Paris-Saclay	President, Rapporteur
Anne TANGUY	Professor, INSA Lyon	Rapporteure
Mouna EL HAFI	Professor, IMT Mines Albi	Examinator
Francis H. R. França	Professor, UFRGS - Federal University of Rio Grande do Sul	Examinator
Nicolas STEIN	Professor, Université de Lorraine	Invited

Laboratoire Énergies & Mécanique Théorique et Appliquée - UMR 7563
Université de Lorraine
2 Avenue de la Forêt de Haye - 54505 - Vandoeuvre-lès-Nancy Cedex, France

All you really need to know for the moment is that the universe is a lot more complicated than you might think, even if you start from a position of thinking it's pretty damn complicated in the first place.

Douglas Adams
(1952 - 2001)

Nano-κ: a Python code for multiscale modelling of the thermal conductivity

B. H. da Silva

Abstract

Electronic devices are present in almost every aspect of modern society and their optimisation and control is of paramount importance in the development of new technologies. In addition, environmental concerns about their energy efficiency and lifetime require the testing of alternatives that minimise human impact on nature. One of the most common materials used in electronic nanodevices is semiconductors, such as silicon (Si) and germanium (Ge). In this context, there is a strong motivation to study phonons, quanta of crystal lattice vibration, which are the main carriers of thermal energy in semiconductors. At the macroscale, material properties such as thermal conductivity are usually considered to be independent of boundary conditions. This is not the case at the nanoscale, where each vibrational mode of the material can behave differently due to the geometric configuration. This requires a more detailed calculation to understand how geometric parameters affect the ability of the nanodevice to conduct heat. Understanding heat conduction at the nanoscale is important to avoid overheating the system and to understand how temperature affects its electrical performance. Computational tools could efficiently provide great insights to understand these effects. In fact, several works have already used numerical calculations to understand the thermal behaviour of nanodevices, but usually with in-house codes that are not open to the community. In this context, this thesis presents Nano-κ, a Python code to solve the Boltzmann transport equation (BTE) in nanodevices using the Monte Carlo method with *ab initio* data as input. First, the theory behind phonon transport and its computational implementation in Nano-κ is discussed. Then, a sensitivity analysis is performed to verify the effect of the main simulation parameters on the estimated thermal conductivity. The thermal conductivity calculated by Nano-κ is then compared with results from the literature in several thin film and nanowire settings, which in general show good agreement. In addition, an arbitrary geometry is simulated in two different cases, demonstrating Nano-κ's flexibility and consistency in providing good estimates of heat transfer in nanodevices. The thesis concludes by suggesting possible avenues for improvement in future work.

Keywords: phonons, nanodevices, heat transfer, thermal conductivity, numerical simulation, semiconductors.

Nano-κ: un code Python pour la modélisation multi-échelle de la conductivité thermique

B. H. da Silva

Résumé

Les appareils électroniques sont présents dans presque tous les aspects de la société moderne et leur optimisation et leur contrôle sont d'une importance capitale pour le développement de nouvelles technologies. En outre, les préoccupations environnementales relatives à leur efficacité énergétique et à leur durée de vie nécessitent de tester des alternatives qui minimisent l'impact de l'homme sur la nature. Les semi-conducteurs, tels que le silicium (Si) et le germanium (Ge), sont l'un des matériaux les plus couramment utilisés dans les nanodispositifs électroniques. Dans ce contexte, l'étude des phonons, quanta de vibration du réseau cristallin, qui sont les principaux vecteurs de l'énergie thermique dans les semi-conducteurs, suscite une forte motivation. À l'échelle macroscopique, les propriétés des matériaux telles que la conductivité thermique sont généralement considérées comme indépendantes des conditions de bord. Ce n'est pas le cas à l'échelle nanométrique, où chaque mode de vibration du matériau peut se comporter différemment en raison de la configuration géométrique. Cela nécessite un calcul plus détaillé pour comprendre comment les paramètres géométriques affectent la capacité du nanodispositif à conduire la chaleur. Il est important de comprendre la conduction de la chaleur à l'échelle nanométrique pour éviter la surchauffe du système et pour comprendre comment la température affecte ses performances électriques. Les outils informatiques pourraient fournir des informations précieuses pour comprendre ces effets. En fait, plusieurs travaux ont déjà utilisé des calculs numériques pour comprendre le comportement thermique des nanodispositifs, mais généralement avec des codes internes qui ne sont pas ouverts à la communauté. Dans ce contexte, cette thèse présente Nano-κ, un code Python pour résoudre l'équation de transport de Boltzmann (BTE) dans les nanodispositifs en utilisant la méthode Monte Carlo avec des données *ab initio* en entrée. Tout d'abord, la théorie du transport des phonons et sa mise en œuvre dans Nano-κ sont discutées. Ensuite, une analyse de sensibilité est réalisée pour vérifier l'effet des principaux paramètres de simulation sur la conductivité thermique estimée. La conductivité thermique calculée par Nano-κ est ensuite comparée aux résultats de la littérature dans plusieurs contextes de couches minces et de nanofils, qui montrent en général une bonne concordance. En outre, une géométrie arbitraire est simulée dans deux cas différents, démontrant la flexibilité et la cohérence d'Nano-κ pour fournir de bonnes estimations du transfert de chaleur dans les nanodispositifs. La thèse conclut en suggérant des pistes d'amélioration possibles pour les travaux futurs.

Mots-clés: phonons, nanodispositifs, transfert de chaleur, conductivité thermique, simulation numérique, semi-conducteurs.

Acknowledgements

Finishing this PhD is the end of a cycle that began back in 2010 when I started studying mechanical engineering at UFRGS, maybe even before. When deciding my next steps in high-school, I remember not even considering going to university, let alone becoming a doctor. I have no doubt that this path was the result of thousands of small nudges by school teachers, university professors, colleagues, mentors, family and friends, that allowed me to come this far. Nevertheless, there were several people who made these last three years much brighter, and for that I would like to express my gratitude:

First of all, to my supervisor, Prof. Laurent Chaput, and co-supervisor, Prof. David Lacroix, who trusted on my work and guided me to make the best out of it. Your words of appreciation at the end of the defense meant a lot to me.

To the National Research Agency for the financial support and to the EXPLOR centre hosted by the Université de Lorraine for the high performance computing resources, mostly represented by Mauricio. These results were only possible with your help to run Anaconda on the cluster.

To the jury, Prof. Jérôme Saint-Martin, Prof. Anne Tanguy, Prof. Mouna El Hafi and Prof. Nicolas Stein, for taking the time to read and evaluate this thesis.

To Prof. Francis H. R. França, which was present in my very first day of Mechanical Engineering at UFRGS, for accepting our invitation to be a member of the jury. I would not be here if it were not for your recommendation and trust.

To my wife, Bianca, who accepted to cruise the ocean and come to another continent with me during a global pandemic. Who supported me, shook me up when needed, and listened to all my venting and ranting. I love you.

To my parents, Edgar and Claudia, which endured over 3 years of my transatlantic absence and for their support despite all the distance. I would not be here if they had not shown me how education could change my life. I love you.

To my in-laws, Sergio and Irineia, who I know are also suffering with our decision to come to Europe. Thank you for welcoming me into your family since the beginning, and for supporting us during this challenge.

To Arthur and Elaine, who were almost like older siblings in my first days in France and even some weeks before I arrived; to Anna, Ítalo, Júlia and Mateus for all the help with the Brazilian buffet after the defense; to all my other Brazilian friends in Nancy or not that made me feel a bit closer to home every time we

talked: Arlindo, Bez, Guilherme, Gustavo, João, Kevin, Rafael, Rodrigo, Zucco and Ajuntamento Campeiro. Thank you all for the immense support, even when you didn't know you were giving it.

To Giuseppe, who hurried his own work so he could help me with the streaming of the defense to family and friends; to Léa for the motivating words before the big day; to Lucas for always being available to give professional advice; to Juan and Natalia, for being there even after moving to the other side of the France; to Raj and Hassan, for the chats while we all prepared our defenses. Thank you all for always being such good friends.

To all members of the nano group: Liudmyla, Lesia, Nataliya and Jenna, who had to share the office with me and hear me beating my fingers while thinking, swearing at the computer and making other annoying noises; Gilles and Mykola for the several pieces of advice; Viktor and Marcel, who I sometimes bothered in their office looking for help or just to chat.

Thank you all those not yet mentioned who helped me to finally fix my accordion: Adrien, Guillaume, Nadia, Thomas, William and any other who may have contributed in anonymity. I don't know how I survived for three years without it, but thanks to you it will finally be fixed.

To all other friends and colleagues at LEMTA, which are way too many to name, but were all important in one way or another. For their company during lunch, chats by the coffee machine and for the career advice. These three years would surely be very empty if it wasn't for all of you.

Contents

Obs.: L'Introduction (pages 9 à 17) et la Conclusion (pages 121 à 126) ont aussi des versions rédigées en français, ainsi comme le Résumé.

Obs.: The Introduction (pages 9 to 17) and Conclusion (pages 121 to 126) are also available in French, as is the Abstract.

List of Figures

List of Algorithms

List of abbreviations

1D	One dimension
2D	Two dimensions
3D	Three dimensions
AMM	Accoustic mismatch model
BC	Boundary condition
BTE	Boltzmann transport equation
BZ	Brillouin zone
CPU	Central processing unit
DFT	Density functional theory
DMM	Diffuse mismatch model
FBZ	First Brillouin zone
FCC	Face centered cubic
LD	Lattice dynamics
MC	Monte Carlo
MD	Molecular dynamics
OOP	Object-oriented programming
PnC	Phononic crystal
RTE	Radiative tranport equation

List of symbols

Occasionally the same symbol is used in different contexts. Auxiliary variables, quantities that are not important for the discussion as a whole or that occur only once are enunciated in the text only and are not present in the list. Alternative uses in specific sections are signaled.

Latin symbols

A	Amplitude / Surface area
a	Distance between atoms at equilibrium (Sec 2.1.2)
a_1, a_2, a_3, a_4	Barycentric coordinates
a, $a^\dagger$	Annihilation and creation operators (Sec. 2.1.2: Quantum mechanics)
$\mathbf{a}$, $\mathbf{b}$, $\mathbf{c}$	Real lattice basis vectors
$\mathbf{a}^*$, $\mathbf{b}^*$, $\mathbf{c}^*$	Reciprocal lattice basis vectors
$\mathbf{B}$	$\mathbf{q}$ to $\mathbf{k}$ transformation matrix
$\mathbf{b}$	Basis vector (Sec. 3.2.1)
C	Specific heat
$crit$	Stop criterion
$\mathbf{c}$	Centroid
E	Energy / energy exchange
e	Energy density
$\mathbf{e}$	Mode eigenvector
f	Ordinal frequency / function
$\mathbf{G}$	Reciprocal lattice vectors
$\mathcal{H}$	Hamiltonian
h	Film height
$\hbar$	Planck's reduced constant
j	Branch index
$\mathbf{j}$	Flux of particles
k	Wavevector in 1D (Sec. 2.1.2)
k_b	Boltzmann's constant
$\mathbf{k}$	Wavevector in 3D
$\mathfrak{K}$	Plane constant
$\mathcal{L}$	Lagrangian
m	Atom mass
N	Number of primitive cells / wavevectors in the FBZ
N_x	Number of x
n	Occupation number
n^0	Bose-Einstein occupation number

$\mathbf{n}$	Surface normal vector
$\mathbf{n}_{i,j}$	Connection vector between subvolumes i and j
$\mathbf{o}_{i,j}$	Midpoint between subvolumes i and j
P_x	Probability of choosing x
p	Specularity parameter
$p(x)$	Probability distribution
p_{fi}	Index of the i^{th} vertex in face f
$\mathbf{p}$	Momentum (Sec. 2.1.2: Quantum mechanics) / Vertex vector (Sec. 3.2.1) / BC reference points (Sec. 4.5)
q	Q-point index
$\mathbf{q}$	Normalised wavevector (q-point)
R	Rate of injection of particles
$\mathbf{R}_n$	Lattice vector to the n^{th} unit cell
r	1D Position
$\mathbf{r}$	3D Position / reference point
S	Geometry's surface
T	Thermodynamic temperature
T^*	Pseudo-temperature
$\mathcal{T}$	Kinetic energy
t	Time
Δt	Timestep size
u_n	Displacement from equilibrium of atom n
V	Volume (crystal, geometry) / Potential energy
V_0	Volume of the primitive cell
$\mathbf{v}$	Group velocity
$\mathbf{x}$	Domain vector
x_i	i^{th} component of $\mathbf{x}$
x_i, y_i, z_i	Cartesian coordinates of vertex/point i
X, Y, Z	Fixed Cartesian coordinates (Sec. 4.5)
x, y, z	Moving Cartesian coordinates (Sec. 4.5)
Z	Partition function (Sec. 2.1.2: Quantum mechanics)

Greek symbols

Γ	Imaginary part of self energy
γ	Random number uniformly distributed in [0, 1)
$\delta_{ii'}$	Kronecker delta
ϵ	Convergence error
$\mathbf{\Phi}$	Heat flux
φ	Atomic interaction energy function
ϕ	Wave phase
η	Surface roughness
κ	Thermal conductivity
Λ	Phonon mean free path
λ	Integrating factor
μ	Mean quantity
Ω	Function domain
ω	Angular frequency
ρ	Density of particles per volume

σ	Standard deviation
τ	Mode relaxation time
θ	Incidence angle

Subscripts and superscripts

0 (subscript)	Initial / primitive cell
0 (superscript)	Bose-Einstein / equilibrium
a	Atoms in the primitive cell
bt	Tetrahedron base
c	Connection / collision
$cold$	Cold / lowest temperature
$diff$	Diffuse reflections
e	Edge
F	Facet
f	Face (triangle)
hot	Hot / highest temperature
i	Particle i
iso	Isotope
k	Iteration
l	Unit cell index
$\mathbf{k}$	Reciprocal space
$\mathbf{k}j$	Mode
max	Maximum
N	Normal process
n	Atom index
o	Origin
p	Particles
pr	Particles crossing reservoir surface
ps	Particles in subvolume
r	Reservoir
$\mathbf{r}$	Real space
ref	Reference
s	Subvolume / Taylor expansion term index / Travelled path
$spec$	Specular reflections
t	Tetrahedron
U	*Umklapp* process
v	Vertex
α	Cartesian coordinate index
τ	Atom index in the primitive cell

Others

inf	Infinity
$f(x)$	f is function of x
Re	Real part
sign	Sign function
$\overline{x}$	Average of x
Δx	Difference in x / interval of x
δx	Deviation of x from equilibrium
∇x	Gradient of x
$\nabla \cdot x$	Divergent of x
$\|x\|$	Absolute value of x
$\|\mathbf{x}\|$	Length of $\mathbf{x}$
$\dot{x}$	Derivative of x in time

Chapter 1

Introduction

One of the most limited resources in technological development is space, whatever the context. Complex systems often require miniaturisation of their components to fit easily where they are needed. Alternatively, by reducing the size of the components that make up a system, its performance can be increased for the same overall size. In electronics, this trade-off between size and function can be translated into the number of components an electronic circuit can have. Among them, perhaps, the most important of these components is the transistor.

The transistor, although it has an old history, was notably developed at Bell Labs from the 1930s, leading to the 1956 Nobel Prize in Physics being awarded to Bardeen, Brattain and Shockley [1]. It is an electronic switch that acts as a logic gate, which can be turned on (state 1) and off (state 0) by applying a difference in electrical potential across one of its parts, aptly called a "gate". It is the combination of millions of transistors and their binary coding that allows modern digital computers to perform calculations. Their operating principle relies on the properties of semiconductors, solid materials that lie between conductors and insulators in terms of their ability to hold electrical current: their atoms are neither as good at conducting heat and electricity as metals such as aluminium (Al), gold (Au) or copper (Cu), nor as bad as insulators such as polymers or glass. Examples of semiconducting materials are silicon (Si) and germanium (Ge). The properties of these materials can be modified by adding atoms of other elements, such as phosphorus (P) or boron (B), which allow an electric current to flow under certain conditions. This process is called doping, and it is what makes it possible to control the current in transistors.

The smaller a transistor, the more of it can be set in a microprocessor. This translates directly into the computing power of the computer's CPU (central processing unit). Gordon E. Moore, co-founder of Intel, famously stated Moore's Law in 1965, predicting that the number of transistors in a microprocessor would double every two years or so [2]. This has held true for more than 50 years, directly translating into computing power [3]. However, there have been concerns that the validity of Moore's Law may be coming to an end [4] as transistor sizes approach the atomic scale. By 2022, transistor density exceeded 130 million per square millimetre, or 130 per square micrometre [5]. This is equivalent to a transistor occupying the average area of a square with a side of about 90 nm. For comparison, the unit cell of cubic silicon, containing 8 Si atoms, has a dimension of about 0.55 nm [6], making the edge of a transistor about 1300 Si atoms in length. Eventually, a wall

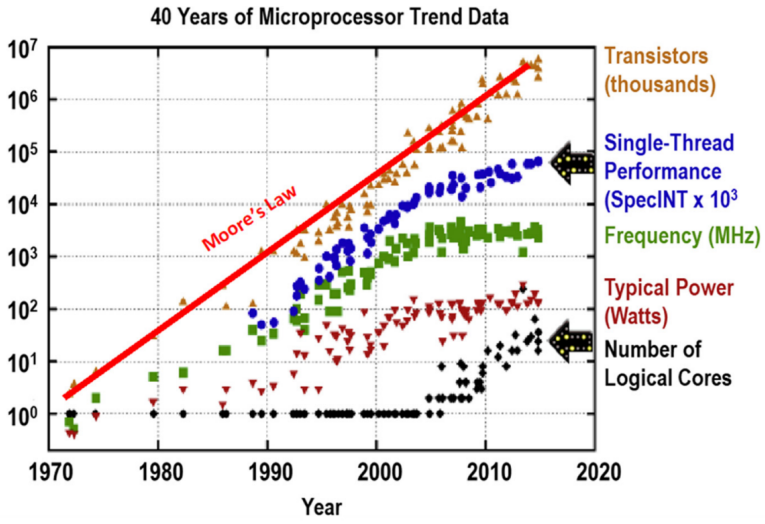


Figure 1.1: Evolution of computational processing capacity in different metrics. Adapted from Gargini, Balestra, and Hayashi [7].

of progress will be reached and new technological alternatives will have to emerge to keep increasing computing power. And since Moore's Law is an exponential law, this date should be close.

Whatever new technologies emerge, energy efficiency remains a major issue. Personal computers, mobile phones, smart watches, televisions... electronics permeate every aspect of our modern lives. Their energy efficiency and reliability can therefore have a major impact on greenhouse gas emissions in a digitalised economy. Despite existing evidence of "rebound effects" that cause global energy consumption to increase as the efficiency of devices improves [8], it is undeniable that the drive for lower energy consumption is a fundamental goal. The heat dissipated by electronic devices can damage their components, which would only reduce their lifespan and create more environmental problems from manufacturing, transport and recycling. Thermoelectric control of these systems is therefore a major concern, as is understanding the nanocomponents that can provide this control.

The importance of research on nanostructures, such as thin films and nanowires, is reflected in the number of publications per year on the subject. Between 1990 and 2010, the relative increase in the number of publications per year in this field was higher than the increase in the number of publications in general, reaching the same relative exponential growth rate as the total number of publications in the last decade [9].

The mathematical model for heat transfer at the nanoscale must differ from that used at the macroscale. While at the macroscale, heat conduction can be modelled using Fourier's law, with the thermal conductivity being defined as a global material property, this law is no longer valid when the analysis reaches the scale of tens of micrometres. At this scale, the individual modes of vibration of the atomic structure must be taken into consideration. The behaviour of these modes can be described by a gas of quantised particles, progressing as wave packets through the material and conveying energy through their oscillations. These entities are known as phonons (from the Greek word φωνή for "voice", alluding to sound waves) and serve as the primary carriers for transferring thermal energy within semiconductors [10, 11]. The

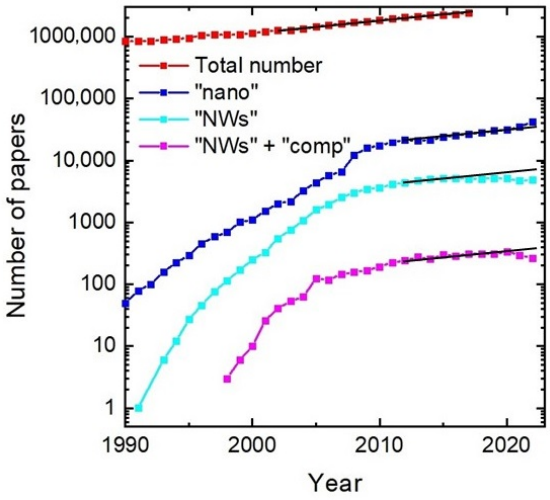


Figure 1.2: Number of publications per year when searching for "nano", "nanowires" and "nanowires + composition" on Scopus in comparison to the total number of published papers [9].

applicability of the phonon gas model has even been studied on amorphous solids and liquids [12, 13], despite lacking the periodic structure that allows phonons to be defined. Instead of Fourier's law, the Boltzmann Transport Equation (BTE) is the equation that models how energy is transported by phonons. The solution of the BTE for each mode depends on the estimation of local temperature, which in turn is dependent on the solution of the BTE for every mode. In consequence, there exists the necessity of solving every BTE simultaneously.

The majority of analysed nanocomponents comprise thin films and nanowires, with considerations given to various parameters such as size (film thickness, wire diameter), surface roughness, temperature, and composition. Some of these materials consist of alloys (composed of a mixture of multiple elements) or superlattices (ordered layers of different elements) [14]. Other nanocomponents can be fabricated with the objective of modifying the material properties by manipulating the geometry and producing pores, holes, protuberances, gratings and even entire wire networks. Thermal cloaking is one potential application of these materials [16, 20, 21], whereby a specific region of the component is protected from external heat flux. Another example involves the use of porous materials [22, 23] that can be fabricated to decrease thermal conductivity by limiting the phonon mean free path. When the porosity is arranged in a predictable geometric manner, the nanocomponents can be classified as phononic crystals (PnC), resulting in phonon interference and resonance effects [24, 25]. The usage of nanoengineering also involves the fusion of phonon transport with radiative heat transfer, seen in applications like solar cells and radiative cooling [26, 27]. Figure 1.3 shows some examples of nanoengineered components.

The manipulation of thermoelectric effects can also be studied. The three main phenomena studied in thermoelectrics are the Peltier, Seebeck and Thomson effects. They describe how an electrical current can induce a temperature gradient and vice-versa. These effects can be used for instance to harvest residual thermal energy for electricity generation [28–30]. The efficiency of this conversion is typically evaluated by examining the thermoelectric figure of merit ZT, which varies inversely with

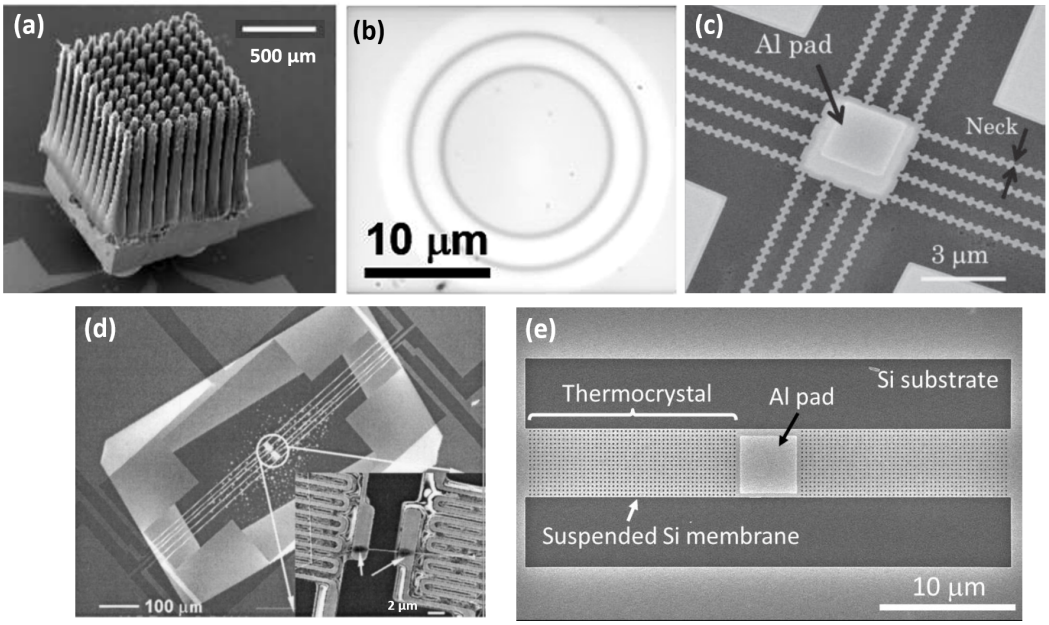


Figure 1.3: Examples of nanostructures. (a) Nanocolumns used for cooling nanodevices. Adapted from Kordás et al. [15]. (b) Nanostructure used from thermal cloaking. Adapted from Choe et al. [16]. (c) Fishbone PnC. Adapted from Maire and Nomura [17]. (d) Simple nanowire. Adapted from Li et al. [18]. (e) PnC composed of aligned circular pores. Adapted from [19].

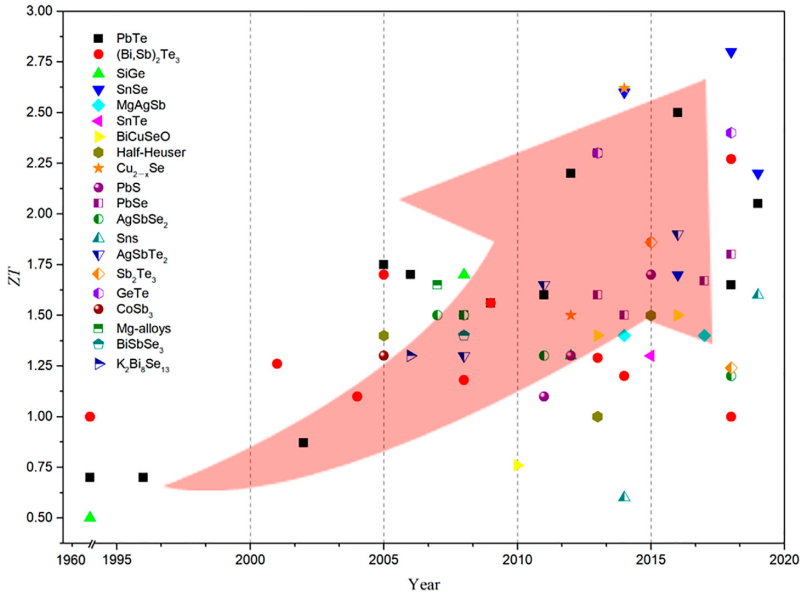


Figure 1.4: Evolution of the thermoelectric figure of merit ZT for several materials along the years. Original image by Sun et al. [30].

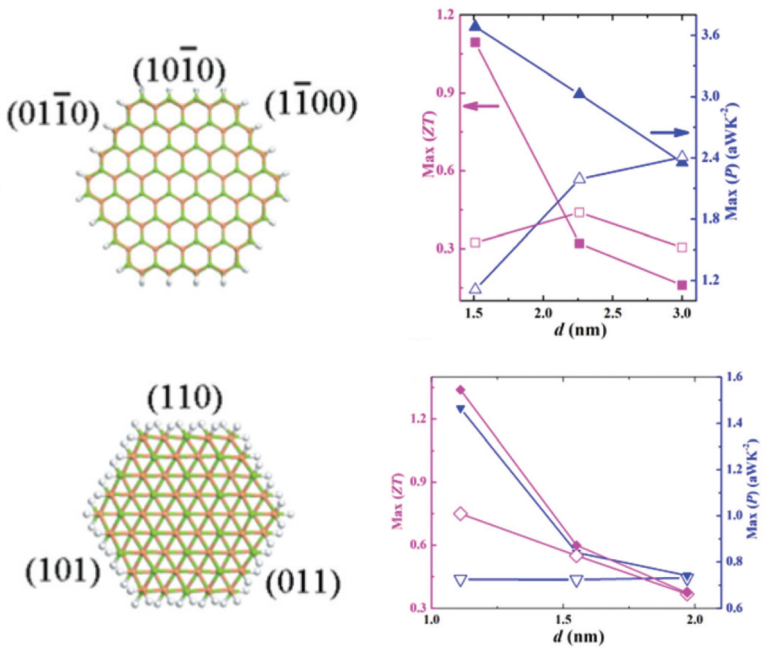


Figure 1.5: Thermoelectric figure of merit ZT as function of nanowire diameter in doped GaAs nanowires. The filled and empty plot markers refer to p and n type nanowires, respectively. Values calculated using DFT and non-equilibirum Green's function methods. Green, orange and white atoms represent Ga, As and H, respectively. Top: wurtzite stacking; bottom: zinc blende stacking. Adapted from Zou et al. [31].

thermal conductivity. Maintaining minimal heat flux in high-temperature gradients is therefore advantageous in this situation. Alternatively, an electric current could be used to induce a temperature gradient, helping to cool nanocomponents that generate excessive heat.

Experimental methods are frequently employed to estimate the thermal characteristics of nanocomponents, although fabricating and testing nanostructured materials can prove to be challenging. Moreover, testing nanocomponents in varying conditions (such as low/high temperature or pressure) may give rise to additional barriers. One solution is to investigate alternative means by computationally modelling the materials. Traditionally, material thermal properties have been calculated using molecular dynamics (MD) [32–35] for bulk, nanocomponents, and interfaces. However, MD poses a significant challenge due to its high computational cost when operating on larger components and its reliance on empirical functions for interatomic potentials. More recently, advances in *ab initio* modelling [36] have improved calculations by employing density functional theory (DFT) to solve interatomic forces from quantum mechanics. This approach gives a better understanding of the contribution of each vibrational frequency to the bulk crystal's properties.

At macroscale, the thermal conductivity is usually treated as independent of its geometry. Understanding the scattering of phonons at boundaries, however, is essential for accurate modelling of thermal transport in nanoscale semiconductor materials. The interaction of phonons with interfaces, surfaces, and defects significantly influences heat conduction in these systems [37]. The specularity parameter has been largely used to approximate the phonon reflection on boundaries as func-

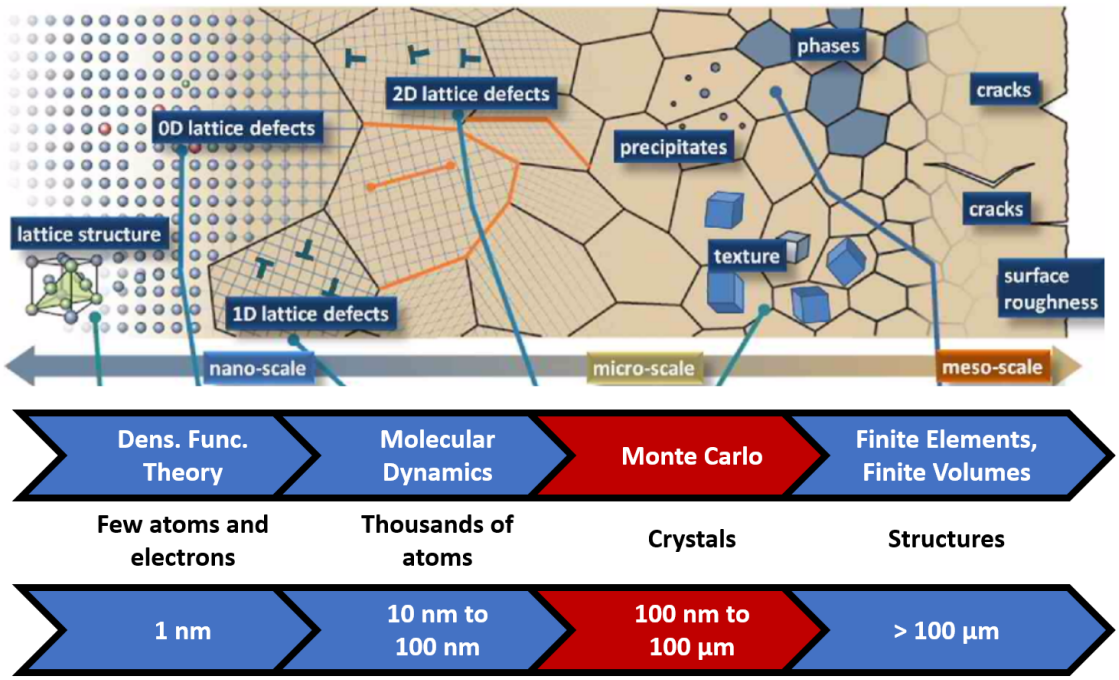


Figure 1.6: Numerical simulation methods applied to heat conduction problems in different size scales. Adapted from Lacroix [48] and Roters et al. [49].

tion of the surface roughness [10, 38]. The acoustic mismatch and diffuse mismatch models (AMM and DMM, respectively) and their variations have been used to estimate phonon transmission between different materials [39–42]. Additionally, various computational methods have been employed to understand boundary scattering, including molecular dynamics (MD) and lattice dynamics (LD) simulations and the S-matrix method [43–47]. These models provide valuable insights into the mechanisms governing phonon reflection and transmission at interfaces, shedding light on the intricate processes that control thermal resistance at boundaries.

While deterministic approaches can be applied to compute lattice vibration problems [50], significant advances have been made in phonon transport via the Monte Carlo (MC) method [51–53], which gives a statistical solution to this problem. Since its inception by Metropolis and Ulam [54], the Monte Carlo method has been developed into several variations, each with specific advantages and disadvantages for particular cases. Often used in other heat transfer domains, like radiative heat transfer, [55], the significant number of BTEs to solve presents phonon transport as an ideal case for implementing various MC techniques already investigated in other areas. When the MC method is applied to nanoscale heat transfer, phonons are represented by point particles, which carry energy through the domain, interact with the nanocomponent's boundaries and with other phonons, while collectively giving rise to temperature, heat flux and thermal conductivity measures. Typically, only the results of these MC simulations (typically performed with in-house code) are published, but not the code itself, which hinders the repeatability of the calculations.

Despite the recent availability of some BTE solver packages online [56, 57], there are still opportunities to provide tools with alternative formulations and goals. Carrete et al. [56], for example, brings almaBTE [58], a software package with several

tools for estimating thermal conductivity. The software is an improvement of the previous ShengBTE [59, 60], which calculated the bulk properties of the crystal using DFT and the effective thermal conductivity for nanowires assuming diffuse reflections. AlmaBTE improves on this by including, in addition to the previous features, specific models for in-plane and cross-plane conductivity in thin films, a Monte Carlo solver of the steady-state BTE in 1D, an analytical 1D solver of the transient BTE for heat pulses, and superlattice models. In films, the effective thermal conductivity is calculated by applying a "suppression function" S calculated for each mode, depending on its mean free path and the specularity of the wall. The contribution of each mode to the heat flux is then added, weighted by its S value. The transient analytical 1D solver focuses on solving the BTE for heat pulses, which are often used in experimental methods. Finally, the 1D MC solver is used to calculate the 1D heat transfer in layered materials, approximating the phonon properties at the mean temperature. The diffuse mismatch model (DMM) is used for the interfaces.

Although it provides several useful tools for phonon transport simulation, alma BTE has some limitations. The first obvious drawback is the lack of generality for 3D problems, since all problems other than the pure bulk crystal are solved in one dimension. For the thin film calculations, the suppression function S seems to assume that the planes of the rough surfaces of the film are symmetry planes of the crystal. In other words, it assumes that each mode has another mode which is the perfect reflection of the first. This assumption may not always be true, depending on the orientation of the crystal in relation to the surfaces. It also assumes that the specularity is the same for all modes, which has been shown theoretically not to be the case by Ziman [10] and Soffer [38]. In the MC solver, the assumption that the phonon properties vary little with temperature is only valid for high temperatures, making almaBTE unsuitable for low temperature applications. In fact, some of these concerns were treated in a recent expansion called BTE-Barna [61], which made a correction to the relaxation time approximation and expanded the calculation to two dimensions.

Another recently released phonon transport simulation package is P-TRANS [62], by Shao, Hori, and Shiomi [57]. Its main focus is on calculating the effects of porosity, inclusions and grains on thermal conductivity. P-TRANS uses the Monte Carlo method to sample the mean free path of phonons in arbitrary 3D geometries in isothermal settings. The material properties are derived from *ab initio* data using an isotropic approximation during the MC simulation. The transmissivity of the nanocomponent is then estimated from the effective mean free path. The software provides an efficient way of estimating how topological features can affect heat transfer, but its isothermal constraint and isotropic simplification can hide some of the nuances in specific cases. It is also unable to provide an estimate of temperature and heat flux distribution.

Given the exposed context, this thesis presents Nano-κ (pronounced "Nano-kappa"), a Python program designed to simulate phonon transport in nanostructures. Building on previous work by LEMTA's nanoscale heat transfer research group and collaborators [22, 63–65], Nano-κ aims to statistically estimate the solution of the Boltzmann Transport Equation (BTE) in any given semiconducting nanocomponent, and thus its thermal properties, such as thermal conductivity, temperature and heat flux distribution. For this, it employs input material data

obtained through *ab-initio* calculations using a unique methodology distinct from other available packages.

Chapter 2 presents the theoretical basis about phonons: what they are, how they are mathematically defined, how their properties are calculated and how they transport energy across the material.

Chapter 3 explains the methodological approach used to solve the Boltzmann Transport Equation. The first section brings an introduction about the Monte Carlo method and its applications on heat transfer, while the second part explains how these techniques were applied to the phonon transport problem.

Chapter 4 exposes how the techniques shown in the previous chapter were algorithmically applied in the code, with an example calculation being explained step by step, from the definition of the geometry, passing by the setting of boundary conditions, loading of material data, and analysis of the output files.

Chapter 5 brings an analysis of how the simulation parameters can affect the results of the simulation, using a reference case as a starting point.

Chapter 6 brings the results of several standard study cases and their comparison with experimental data available in the literature. The analysed cases were:

- The cross-plane conduction of Si and Ge thin films for different temperatures and thicknesses;
- The in-plane thermal conductivity for thin-films with different film heights;
- The thermal conductivity of nanowires for different wall roughness and diameters.
- Finally, a non-standard geometry is simulated with two different settings of boundary conditions to exemplify the capabilities of Nano-κ to predict its thermal behaviour.

Chapter 7 makes final remarks on the presented work, discussing what was achieved, how it could be improved and how it can contribute to future studies.

Introduction (français)

L'espace est l'une des ressources les plus limitées dans le développement technologique, quel que soit le contexte. Les systèmes complexes nécessitent souvent une miniaturisation de leurs composants afin de pouvoir les installer facilement là où ils sont nécessaires. Par ailleurs, en réduisant la taille des composants d'un système, il est possible d'en augmenter les performances à taille égale. En électronique, ce compromis entre taille et fonction peut se traduire par le nombre de composants qu'un circuit électronique peut avoir. Parmi ces composants, le plus important est peut-être le transistor.

Le transistor, bien qu'il ait une histoire ancienne, a été notamment développé aux Bell Labs à partir des années 1930, ce qui a conduit à l'attribution du prix Nobel de physique 1956 à Bardeen, Brattain et Shockley [1]. Il s'agit d'un commutateur électronique qui agit comme une porte logique, qui peut être activée (état 1) et désactivée (état 0) en appliquant une différence de potentiel électrique à l'une de ses parties, appelée "grille". C'est la combinaison de millions de transistors et de leur codage binaire qui permet aux ordinateurs numériques modernes d'effectuer des calculs. Leur principe de fonctionnement repose sur les propriétés des semi-conducteurs, des matériaux solides qui se situent entre les conducteurs et les isolants en termes de capacité à retenir le courant électrique : leurs atomes ne sont ni aussi bons conducteurs de chaleur et d'électricité que les métaux tels que l'aluminium (Al), l'or (Au) ou le cuivre (Cu), ni aussi mauvais que les isolants tels que les polymères ou le verre. Le silicium (Si) et le germanium (Ge) sont des exemples de matériaux semi-conducteurs. Les propriétés de ces matériaux peuvent être modifiées par l'ajout d'atomes d'autres éléments, tels que le phosphore (P) ou le bore (B), qui permettent le passage d'un courant électrique dans certaines conditions. Ce processus, appelé dopage, permet de contrôler le courant dans les transistors.

Plus un transistor est petit, plus il est possible d'en placer dans un microprocesseur. Cela se traduit directement par la puissance de calcul de l'unité centrale de l'ordinateur. Gordon E. Moore, cofondateur d'Intel, a énoncé la célèbre loi de Moore en 1965, prédisant que le nombre de transistors dans un microprocesseur doublerait tous les deux ans environ [2]. Cela s'est vérifié pendant plus de 50 ans, se traduisant directement en puissance de calcul [3]. Toutefois, certains craignent que la validité de la loi de Moore ne prenne fin [4] à mesure que la taille des transistors se rapproche de l'échelle atomique. En 2022, la densité des transistors a dépassé les 130 millions par millimètre carré, soit 130 par micromètre carré [5]. Cela équivaut à un transistor occupant la surface moyenne d'un carré d'environ 90 nm de côté. À titre de comparaison, la cellule unitaire du silicium cubique, contenant 8 atomes de Si, a une dimension d'environ 0.55 nm [6], ce qui fait que le bord d'un transistor a une longueur d'environ 1300 atomes de Si. Un jour ou l'autre, un mur de progrès sera atteint et de nouvelles alternatives technologiques devront émerger pour con-

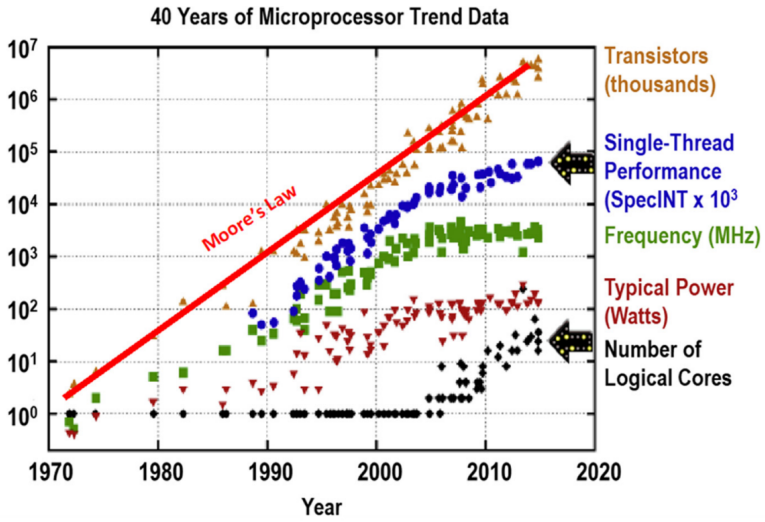


Figure 1.1: Évolution de la capacité de traitement informatique selon différentes mesures. Adapté de Gargini, Balestra, and Hayashi [7].

tinuer à augmenter la puissance de calcul. Et comme la loi de Moore est une loi exponentielle, cette date devrait être proche.

Quelles que soient les nouvelles technologies qui émergent, l'efficacité énergétique reste un problème majeur. Ordinateurs personnels, téléphones portables, montres intelligentes, téléviseurs... l'électronique imprègne tous les aspects de notre vie moderne. Leur efficacité énergétique et leur fiabilité peuvent donc avoir un impact majeur sur les émissions de gaz à effet de serre dans une économie numérisée. Malgré les preuves existantes des "effets de rebond" qui font que la consommation mondiale d'énergie augmente à mesure que l'efficacité des appareils s'améliore [8], il est indéniable que la recherche d'une moindre consommation d'énergie est un objectif fondamental. La chaleur dissipée par les appareils électroniques peut endommager leurs composants, ce qui ne ferait que réduire leur durée de vie et créerait davantage de problèmes environnementaux liés à la fabrication, au transport et au recyclage. Le contrôle thermoélectrique de ces systèmes est donc une préoccupation majeure, tout comme la compréhension des nanocomposants qui peuvent assurer ce contrôle.

L'importance de la recherche sur les nanostructures, telles que les couches minces et les nanofils, se reflète dans le nombre de publications annuelles sur le sujet. Entre 1990 et 2010, l'augmentation relative du nombre de publications par an dans ce domaine a été supérieure à l'augmentation du nombre de publications en général, atteignant le même taux de croissance exponentiel relatif que le nombre total de publications au cours de la dernière décennie [9].

Le modèle mathématique du transfert de chaleur à l'échelle nanométrique doit être différent de celui utilisé à l'échelle macroscopique. Alors qu'à l'échelle macroscopique, la conduction de la chaleur peut être modélisée à l'aide de la loi de Fourier, la conductivité thermique étant définie comme une propriété globale du matériau, cette loi n'est plus valable lorsque l'analyse atteint l'échelle des dizaines de micromètres. A cette échelle, les modes de vibration individuels de la structure atomique doivent être pris en considération. Le comportement de ces modes peut être décrit par un gaz de particules quantifiées, se déplaçant comme des paquets d'ondes à travers le matériau et transportant de l'énergie par leurs oscillations. Ces en-

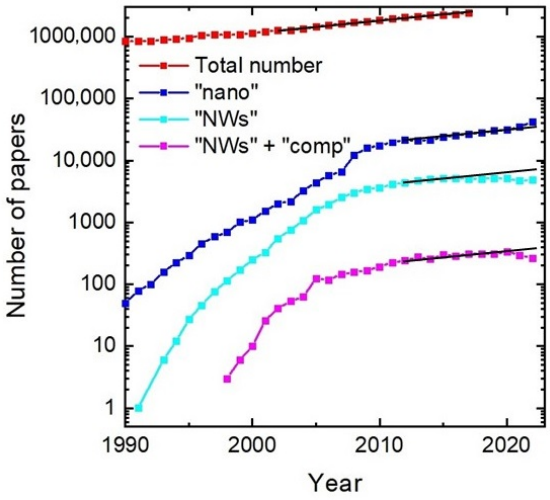


Figure 1.2: Nombre de publications par an lors de la recherche de “nano”, “nanowires” et “nanowires + composition” sur Scopus par rapport au nombre total d’articles publiés [9].

tités sont connues sous le nom de phonons (du mot grec φωνή pour ”voix”, faisant allusion aux ondes sonores) et servent de porteurs primaires pour le transfert de l’énergie thermique dans les semi-conducteurs [10, 11]. L’applicabilité du modèle du gaz de phonon a même été étudiée sur des solides et des liquides amorphes [12, 13], malgré l’absence de la structure périodique qui permet de définir les phonons. Au lieu de la loi de Fourier, l’équation de transport de Boltzmann (BTE) est l’équation qui modélise la façon dont l’énergie est transportée par les phonons. La solution de l’ETB pour chaque mode dépend de l’estimation de la température locale, qui dépend à son tour de la solution de l’ETB pour chaque mode. Par conséquent, il est nécessaire de résoudre chaque BTE simultanément.

La majorité des nanocomposants analysés sont des films minces et des nanofils, en tenant compte de divers paramètres tels que la taille (épaisseur du film, diamètre du fil), la rugosité de la surface, la température et la composition. Certains de ces matériaux sont des alliages (composés d’un mélange de plusieurs éléments) ou des super-réseaux (couches ordonnées de différents éléments). D’autres nanocomposants peuvent être fabriqués dans le but de modifier les propriétés du matériau en manipulant la géométrie et en produisant des pores, des trous, des protubérances, des grilles et même des réseaux de fils entiers. L’occultation thermique est une application potentielle de ces matériaux [16, 20, 21], par laquelle une région spécifique du composant est protégée du flux de chaleur externe. Un autre exemple concerne l’utilisation de matériaux poreux [22, 23] qui peuvent être fabriqués pour réduire la conductivité thermique en limitant le libre parcours moyen des phonons. Lorsque la porosité est disposée d’une manière géométrique prévisible, les nanocomposants peuvent être classés comme des cristaux phononiques (PnC), ce qui entraîne des effets d’interférence et de résonance des phonons [24, 25]. L’utilisation de la nano-ingénierie implique également la fusion du transport des phonons avec le transfert de chaleur radiatif, que l’on retrouve dans des applications telles que les cellules solaires et le refroidissement radiatif [26, 27]. La figure 1.3 présente quelques exemples de composants issus de la nanotechnologie.

La manipulation des effets thermoélectriques peut également être étudiée. Les

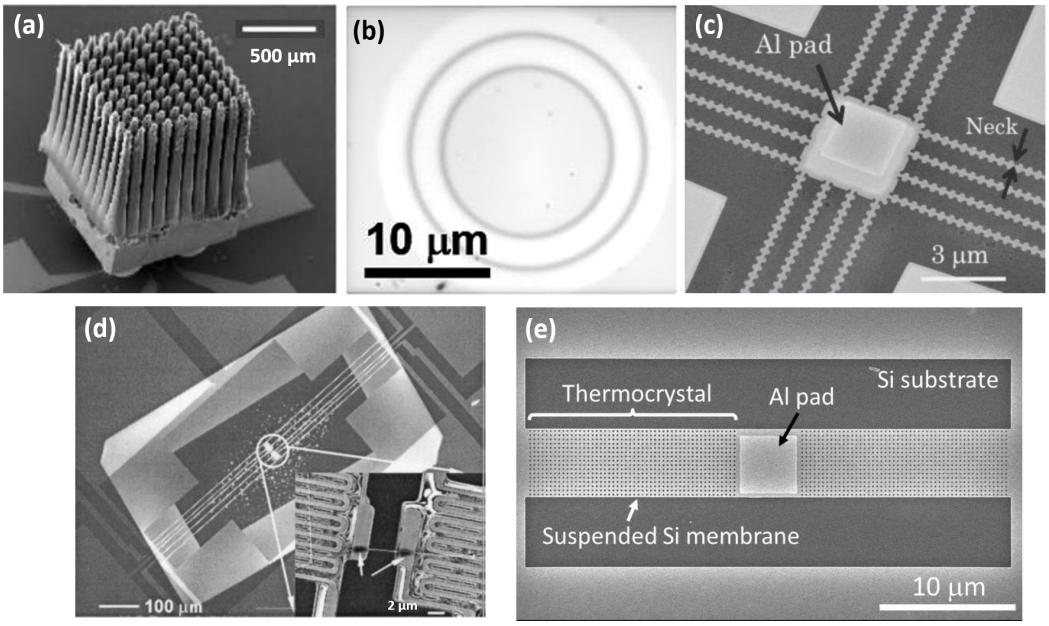


Figure 1.3: Exemples de nanostructures. (a) Nanocolonnes utilisées pour refroidir les nanodispositifs. Adapté de Kordás et al. [15]. (b) Nanostructure utilisée pour l'occultation thermique. Adapté de Choe et al. [16]. (c) PnC en arête de poisson. Adapté de Maire and Nomura [17]. (d) Nanofil simple. Adapté de Li et al. [18]. (e) PnC composé de pores circulaires alignés. Adapté de [19].

trois principaux phénomènes étudiés en thermoélectricité sont les effets Peltier, Seebeck et Thomson. Ils décrivent comment un courant électrique peut induire un gradient de température et vice-versa. Ces effets peuvent être utilisés, par exemple, pour récupérer l'énergie thermique résiduelle afin de produire de l'électricité [28–30].

L'efficacité de cette conversion est généralement évaluée en examinant le facteur de mérite thermoélectrique ZT, qui varie inversement à la conductivité thermique. Le maintien d'un flux thermique minimal dans des gradients de température élevés est donc avantageux dans cette situation. Par ailleurs, un courant électrique pourrait être utilisé pour induire un gradient de température, ce qui contribuerait à refroidir les nanocomposants qui génèrent une chaleur excessive.

Les méthodes expérimentales sont fréquemment utilisées pour estimer les caractéristiques thermiques des nanocomposants, bien que la fabrication et l'essai de matériaux nanostructurés puissent s'avérer difficiles. En outre, le fait de tester les nanocomposants dans des conditions variables (telles qu'une température ou une pression basse/élevée) peut créer des obstacles supplémentaires. Une solution consiste à rechercher des moyens alternatifs en modélisant les matériaux par le calcul. Traditionnellement, les propriétés thermiques des matériaux ont été calculées à l'aide de la dynamique moléculaire (MD) [32–35] pour la masse, les nanocomposants et les interfaces. Cependant, la MD pose un défi important en raison de son coût de calcul élevé lorsqu'elle opère sur des composants plus importants et de sa dépendance à l'égard des fonctions empiriques pour les potentiels interatomiques. Plus récemment, les progrès de la modélisation *ab initio* [36] ont amélioré les calculs en employant la théorie de la fonctionnelle de la densité (DFT) pour résoudre les forces interatomiques à partir de la mécanique quantique. Cette approche permet

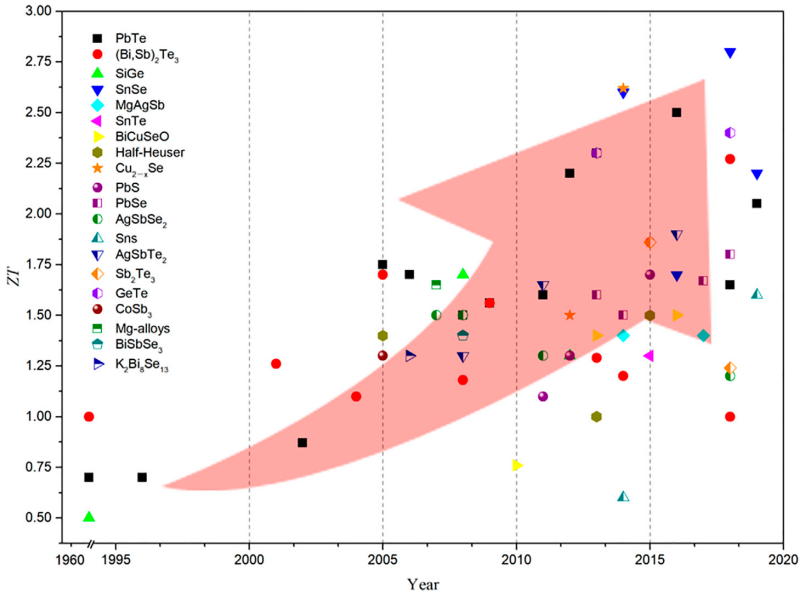


Figure 1.4: Évolution du facteur de mérite thermoélectrique ZT pour plusieurs matériaux au cours des années. Image originale par Sun et al. [30].

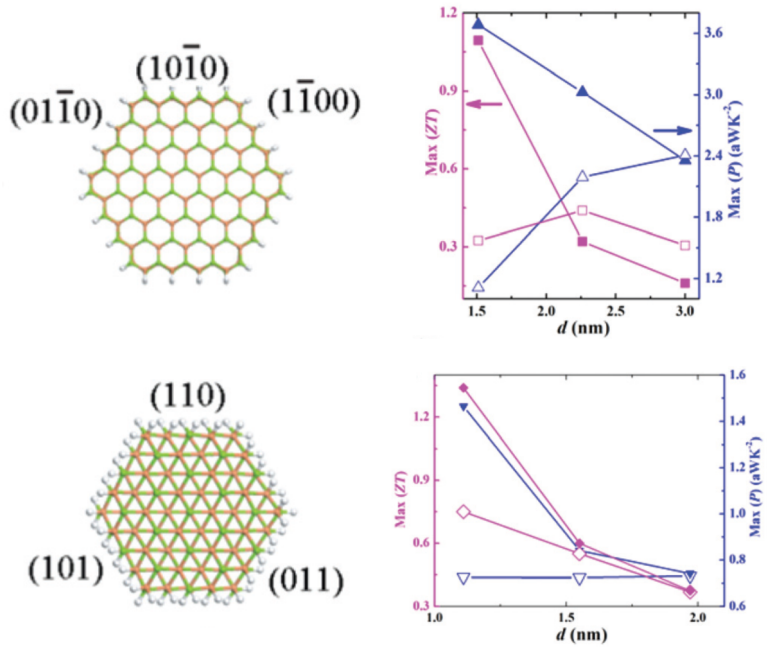


Figure 1.5: Figure de mérite thermoélectrique ZT en fonction du diamètre des nanofils de GaAs dopés. Les marqueurs de tracé remplis et vides se réfèrent aux nanofils de type p et n, respectivement. Valeurs calculées à l'aide des méthodes DFT et de la fonction de Green sans équilibre. Les atomes verts, orange et blancs représentent respectivement Ga, As et H. En haut : empilement wurtzite ; en bas : empilement zinc blende. Adapté de Zou et al. [31].

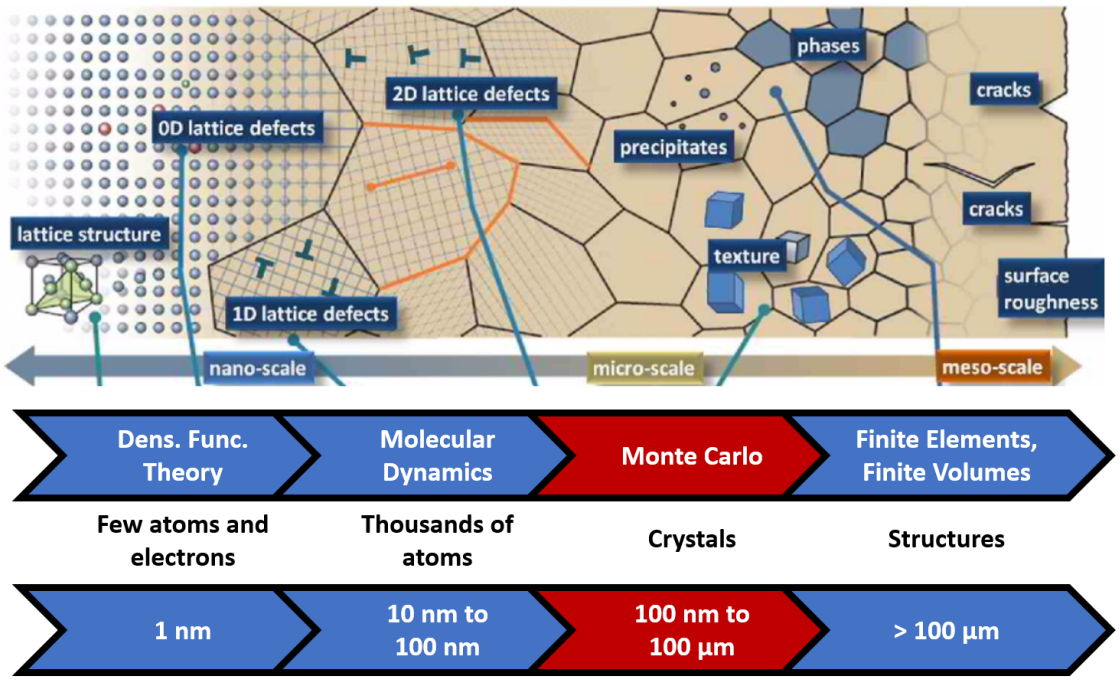


Figure 1.6: Méthodes de simulation numérique appliquées aux problèmes de conduction thermique à différentes échelles de taille. Adapté de Lacroix [48] et Roters et al. [49].

de mieux comprendre la contribution de chaque fréquence vibratoire aux propriétés du cristal.

À l'échelle macroscopique, la conductivité thermique est généralement considérée comme indépendante de la géométrie. Il est toutefois essentiel de comprendre la diffusion des phonons aux frontières pour modéliser avec précision le transport thermique dans les matériaux semi-conducteurs à l'échelle nanométrique. L'interaction des phonons avec les interfaces, les surfaces et les défauts influence considérablement la conduction de la chaleur dans ces systèmes [37]. Le paramètre de spécularité a été largement utilisé pour approximer la réflexion des phonons sur les limites en fonction de la rugosité de la surface [10, 38]. Les modèles de désadaptation acoustique et de désadaptation diffuse (AMM et DMM, respectivement) et leurs variations ont été utilisés pour estimer la transmission des phonons entre différents matériaux [39–42]. En outre, diverses méthodes de calcul ont été employées pour comprendre la diffusion aux limites, notamment les simulations de dynamique moléculaire (MD) et de dynamique des réseaux (LD) et la méthode de la matrice S [43–47]. Ces modèles fournissent des informations précieuses sur les mécanismes régissant la réflexion et la transmission des phonons aux interfaces, mettant en lumière les processus complexes qui contrôlent la résistance thermique aux frontières.

Alors que des approches déterministes peuvent être appliquées pour calculer les problèmes de vibration du réseau [50], des progrès significatifs ont été réalisés dans le transport des phonons via la méthode de Monte Carlo (MC) [51–53], qui donne une solution statistique à ce problème. Depuis sa création par Metropolis and Ulam [54], la méthode de Monte Carlo a été développée en plusieurs variantes, chacune présentant des avantages et des inconvénients spécifiques pour des cas particuliers. Souvent utilisé dans d'autres domaines de transfert de chaleur, comme le transfert

de chaleur radiatif, [55], le nombre important de BTE à résoudre présente le transport de phonon comme un cas idéal pour mettre en œuvre diverses techniques de MC déjà étudiées dans d'autres domaines. Lorsque la méthode MC est appliquée au transfert de chaleur à l'échelle nanométrique, les phonons sont représentés par des particules ponctuelles qui transportent l'énergie à travers le domaine, interagissent avec les limites du nanocomposant et avec d'autres phonons, tout en donnant lieu collectivement à des mesures de la température, du flux de chaleur et de la conductivité thermique. En général, seuls les résultats de ces simulations MC (généralement réalisées avec un code interne) sont publiés, mais pas le code lui-même, ce qui entrave la reproductibilité des calculs.

Malgré la disponibilité récente de certains solveurs de BTE en ligne [56, 57], il existe encore des possibilités de fournir des outils avec des formulations et des objectifs alternatifs. [56], par exemple, présente almaBTE [58], un logiciel comprenant plusieurs outils pour l'estimation de la conductivité thermique. Le logiciel est une amélioration du précédent ShengBTE [59, 60], qui calculait les propriétés globales du cristal à l'aide de la DFT et la conductivité thermique effective pour les nanofils en supposant des réflexions diffuses. AlmaBTE l'améliore en incluant, en plus des caractéristiques précédentes, des modèles spécifiques pour la conductivité dans le plan et dans le plan transversal dans les films minces, un solveur Monte Carlo du BTE à l'état stable en 1D, un solveur analytique 1D du BTE transitoire pour les impulsions de chaleur, et des modèles de super-réseaux. Dans les films, la conductivité thermique effective est calculée en appliquant une "fonction de suppression" S calculée pour chaque mode, en fonction de son libre parcours moyen et de la spécularité de la paroi. La contribution de chaque mode au flux de chaleur est ensuite ajoutée, pondérée par sa valeur S. Le solveur analytique 1D transitoire se concentre sur la résolution du BTE pour les impulsions de chaleur, qui sont souvent utilisées dans les méthodes expérimentales. Enfin, le solveur MC 1D est utilisé pour calculer le transfert de chaleur 1D dans les matériaux en couches, en approximant les propriétés des phonons à la température moyenne. Le modèle de désadaptation diffuse (DMM) est utilisé pour les interfaces.

Bien qu'il fournisse plusieurs outils utiles pour la simulation du transport des phonons, alma BTE présente certaines limites. Le premier inconvénient évident est le manque de généralité pour les problèmes 3D, puisque tous les problèmes autres que le cristal pur en vrac sont résolus en une seule dimension. Pour les calculs de couches minces, la fonction de suppression S semble supposer que les plans des surfaces rugueuses de la couche sont des plans de symétrie du cristal. En d'autres termes, elle suppose que chaque mode a un autre mode qui est la réflexion parfaite du premier. Cette hypothèse n'est pas toujours vraie, en fonction de l'orientation du cristal par rapport aux surfaces. Elle suppose également que la spécularité est la même pour tous les modes, ce qui a été démontré théoriquement comme n'étant pas le cas par Ziman [10] et Soffer [38]. Dans le solveur MC, l'hypothèse selon laquelle les propriétés des phonons varient peu avec la température n'est valable que pour les températures élevées, ce qui rend l'almaBTE inadapté aux applications à basse température. En fait, certains de ces problèmes ont été traités dans une extension récente appelée BTE-Barna [61], qui a apporté une correction à l'approximation du temps de relaxation et a étendu le calcul à deux dimensions.

P-TRANS [62], de Shao, Hori, and Shiomi [57], est un autre logiciel de simulation de transport de phonons récemment publié. Il se concentre principalement sur

le calcul des effets de la porosité, des inclusions et des grains sur la conductivité thermique. P-TRANS utilise la méthode Monte Carlo pour échantillonner le libre parcours moyen des phonons dans des géométries 3D arbitraires dans des conditions isothermes. Les propriétés des matériaux sont dérivées des données *ab initio* en utilisant une approximation isotrope pendant la simulation MC. La transmissivité du nanocomposant est ensuite estimée à partir du libre parcours moyen effectif. Le logiciel fournit un moyen efficace d'estimer comment les caractéristiques topologiques peuvent affecter le transfert de chaleur, mais sa contrainte isotherme et sa simplification isotrope peuvent masquer certaines nuances dans des cas spécifiques. Il n'est pas non plus en mesure de fournir une estimation de la distribution de la température et du flux de chaleur.

Compte tenu du contexte exposé, cette thèse présente Nano-κ (prononcé "Nano-kappa"), un programme Python conçu pour simuler le transport des phonons dans les nanostructures. S'appuyant sur les travaux antérieurs du groupe de recherche sur le transfert de chaleur à l'échelle nanométrique du LEMTA et de ses collaborateurs [22, 63–65], Nano-κ vise à estimer statistiquement la solution de l'équation de transport de Boltzmann (BTE) dans n'importe quel nanocomposant semi-conducteur donné, et donc ses propriétés thermiques, telles que la conductivité thermique, la température et la distribution du flux de chaleur. Pour ce faire, il utilise des données d'entrée sur les matériaux obtenues par des calculs *ab-initio* à l'aide d'une méthodologie unique distincte des autres progiciels disponibles.

Le Chapitre 2 présente la base théorique des phonons : ce qu'ils sont, comment ils sont mathématiquement définis, comment leurs propriétés sont calculées et comment ils transportent l'énergie à travers le matériau.

Le Chapitre 3 explique l'approche méthodologique utilisée pour résoudre l'équation de transport de Boltzmann. La première section présente la méthode de Monte Carlo et ses applications au transfert de chaleur, tandis que la deuxième partie explique comment ces techniques ont été appliquées au problème du transport des phonons.

Le Chapitre 4 expose comment les techniques présentées dans le chapitre précédent ont été appliquées algorithmiquement dans le code, avec un exemple de calcul expliqué étape par étape, depuis la définition de la géométrie, en passant par la définition des conditions aux limites, le chargement des données matérielles, et l'analyse des fichiers de sortie.

Le Chapitre 5 présente une analyse de la manière dont les paramètres de simulation peuvent affecter les résultats de la simulation, en utilisant un cas de référence comme point de départ.

Le Chapitre 6 présente les résultats de plusieurs cas d'étude standard et leur comparaison avec les données expérimentales disponibles dans la littérature. Les cas analysés sont les suivants

- La conduction dans le plan transversal des films minces de Si et de Ge pour différentes températures et épaisseurs ;
- La conductivité thermique dans le plan interne pour les films minces avec différentes hauteurs de film ;
- La conductivité thermique des nanofils pour différentes rugosités de paroi et différents diamètres.

- Enfin, une géométrie non standard est simulée avec deux paramètres différents de conditions aux limites pour illustrer les capacités de Nano-κ à prédire son comportement thermique.

Le Chapitre 7 présente les remarques finales sur le travail présenté, en discutant de ce qui a été réalisé, de la manière dont il pourrait être amélioré et de la manière dont il pourrait contribuer à de futures études.

Chapter 2

Theoretical basis

This chapter introduces the theoretical basis necessary to understand Nano-κ's object of study. First, the fundamentals of lattice dynamics will be explained, defining phonons and their related quantities, including velocity, relaxation time, energy density, heat flux and thermal conductivity. Additionally, the equation that describes the transport of phonons, the Boltzmann Transport Equation (BTE), will be presented.

2.1 Phonons

2.1.1 Crystal structure

Any solid material is composed of atoms tightly linked by atomic bonds. The geometrical arrangement of these atoms depends on their properties (electronic structure, mass, etc.) and the environmental conditions to which they were submitted (temperature, pressure, rapid cooling or heating, for instance). When the atoms are ordered such that a repeating pattern arises, as in a lattice, the solid is called a crystal. In this case, a single unit cell (the repeated unit) and the lattice vectors (the directions to which the unit cell is repeated) are sufficient to describe the structure of the whole solid. When a solid has no clear repeating structure at all, it is called amorphous (e.g. glass and polymers).

Unit cells can be defined in many ways, as long as they can reconstruct the crystal. There are however special cases of unit cells, called primitive cells, which are the smallest possible unit cells. This will be used in the following derivations.

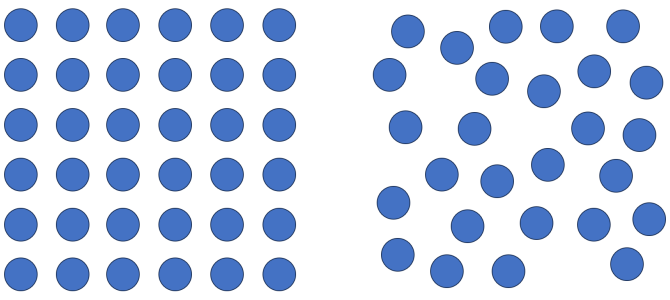


Figure 2.1: 2D-Schematic examples of a crystal and an amorphous solid.

Every crystalline structure has symmetries, but the number of possible symme-

tries (and hence of crystal types) are limited. Figure 2.2 shows 2-dimensional and 3-dimensional examples of a cubic and a hexagonal lattices. It is also shown their lattice vectors **a**, **b** and **c**, and a possible unit cell in red. The other types of Bravais lattices (the possible structures that can generate a crystal) are variations of the parallelepipedic case, named according to the relations between edges and angles (tetragonal, orthorhombic, monoclinic, etc.). Sometimes the structure can contain an atom in its core (being called "body centered") or in the middle of its faces (and called "face centered"). All of them have a corresponding primitive cell with a parallelepipedic form.

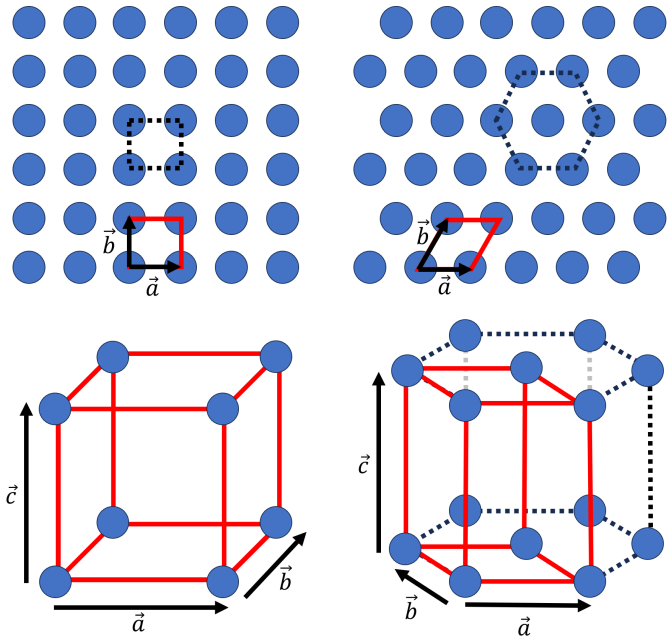


Figure 2.2: Examples of 2D (top) and 3D (bottom) Bravais lattices and their lattice vectors. The primitive cell of each is highlighted in red. Cubic lattices on the left and hexagonal lattices on the right.

The lattice does not have to be composed of a single atom. Multiple atoms, of the same or different elements, can be contained in a unit cell. The periodic configuration means that, if an atom is located at position $\mathbf{r}_0$ in the unit cell closest to the origin, an equivalent atom will be located at any position $\mathbf{r}_n$ in unit cell n, defined in three dimensions as:

$$\mathbf{r}_n = \mathbf{r}_0 + \alpha_n \mathbf{a} + \beta_n \mathbf{b} + \gamma_n \mathbf{c} \tag{2.1}$$

or, more simply:

$$\mathbf{r}_n = \mathbf{r}_0 + \mathbf{R}_n \tag{2.2}$$

where **a**, **b** and **c** are the lattice vectors in real space, and α_n, β_n and γ_n are integers that translate the position from unit cell 0 to unit cell n, such that:

$$\mathbf{R}_n = \alpha_n \mathbf{a} + \beta_n \mathbf{b} + \gamma_n \mathbf{c} \tag{2.3}$$

The term "real space" was used because a lattice has also a representation in reciprocal space, obtained from Fourier transform. The reciprocal space is generated

by the reciprocal space basis vectors $\mathbf{a}^*$, $\mathbf{b}^*$ and $\mathbf{c}^*$ defined as:

$$\mathbf{a}^* = 2\pi \frac{\mathbf{b} \times \mathbf{c}}{\mathbf{a} \cdot (\mathbf{b} \times \mathbf{c})} \tag{2.4}$$

$$\mathbf{b}^* = 2\pi \frac{\mathbf{c} \times \mathbf{a}}{\mathbf{b} \cdot (\mathbf{c} \times \mathbf{a})} \tag{2.5}$$

$$\mathbf{c}^* = 2\pi \frac{\mathbf{a} \times \mathbf{b}}{\mathbf{c} \cdot (\mathbf{a} \times \mathbf{b})} \tag{2.6}$$

Reciprocal space lattice vectors are defined by

$$\mathbf{G} = \alpha_n^* \mathbf{a}^* + \beta_n^* \mathbf{b}^* + \gamma_n^* \mathbf{c}^* \tag{2.7}$$

with α_n^*, β_n^* and γ_n^* integers. The absolute value of the denominator in the definition of the reciprocal space basis vectors is equal to the volume of the unit cell, V_0. The vectors that populate the reciprocal space, real linear combination of $\mathbf{a}^*$, $\mathbf{b}^*$ and $\mathbf{c}^*$, are called "wavevectors", and represented by the symbol $\mathbf{k}$. The wavevectors have units of length^{-1}. We will often consider functions which are periodic in reciprocal space. In this case it is enough to know this function in the parallelepiped build from $\mathbf{a}^*$, $\mathbf{b}^*$ and $\mathbf{c}^*$. Usually, another cell with the same volume $(2\pi)^3/V_0$ is however preferred. It is name Brillouin zone, and is defined as the smallest cell containing the origin whose surface planes are bisecting the reciprocal lattice vectors [10]. An example for an FCC crystal is shown in Fig. 2.3.

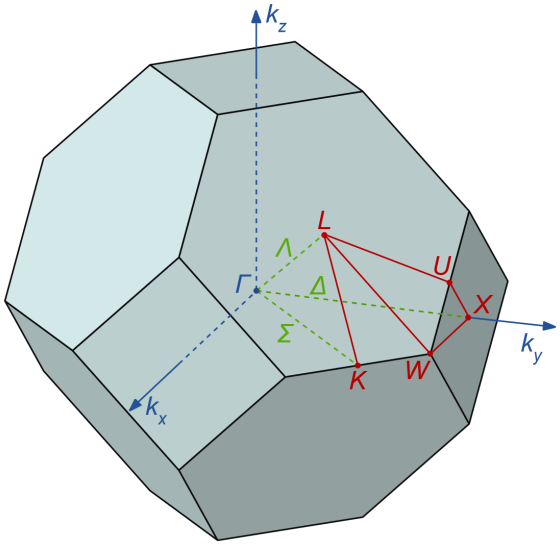


Figure 2.3: First Brillouin zone for a FCC crystal, with critical points and paths marked.

2.1.2 Lattice waves

Even so the atoms in a crystal are rigorously organised in a lattice, it does not mean that they stay still. The simplest case used to illustrate this is a chain of equivalent atoms (therefore referred as "monoatomic") aligned in one dimension and separated by a mean distance a, as shown in Fig. 2.4. The unit cell in this case contains one single atom, and has a length a as well. To ensure that there is no boundary effects,

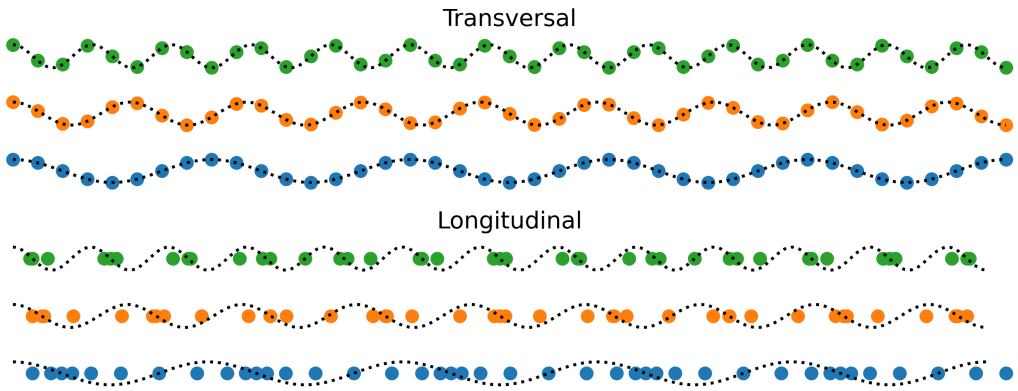


Figure 2.4: Examples of transversal and longitudinal waves in a monoatomic chain.

the Born-von Kármán boundary condition is imposed, linking both extremities of the chain together. The Newton's equations for one atom in the chain is:

$$m\frac{d^2u_n}{dt^2} = -\frac{dE}{du_n} \tag{2.8}$$

where m is the atom's mass, u_n is it's displacement from equilibrium, $r_n = na + u_n$, t is time, and E is the energy of the chain of atoms. If $\phi(r)$ is the energy between two atoms at a distance r, considering only nearest neighbor interactions, $E = \sum_{n=1}^{N}\phi(r_{n+1} - r_n)$. Therefore, when all the N atoms are at equilibrium $E = N\phi(a)$. Allowing the atoms to have a displacement u_n from there equilibrium position, we can write

$$E = N\varphi(a) + \sum_{n=1}^{N}\sum_{s=1}^{\infty}\frac{1}{s!}\frac{d^s\varphi(r)}{dr^s}\Big|_{r=a}(u_{n+1} - u_n)^s \tag{2.9}$$

The first term is zero because the forces on atoms are zero at equilibrium. The harmonic approximation is applied when we assume that the shape of the potential is parabolic, which truncates all terms with $s > 2$. In this approximation:

$$E = N\varphi(a) + \frac{J}{2}\sum_{n=1}^{N}(u_{n+1} - u_n)^2 \tag{2.10}$$

with $J = \frac{d^2\varphi(r)}{dr^2}\Big|_{r=a}$. Applying on Eq. 2.8:

$$m\frac{d^2u_n}{dt^2} = -J[2u_n - u_{n-1} - u_{n+1}] \tag{2.11}$$

We can think of the problem now as if the atoms were connected by identical springs of elastic constant J. To solve the above equations, we assume harmonic solutions,

$$u_n(t) = Ae^{i(kna-\omega t)} \tag{2.12}$$

Substituting in Eq. 2.11, we see it can only be a solution if

$$\omega^2 = \frac{4J}{m}\sin^2\left(\frac{ka}{2}\right). \tag{2.13}$$

The positive root gives

$$\omega_k = \sqrt{\frac{4J}{m}} \left| \sin\left(\frac{ka}{2}\right) \right|. \tag{2.14}$$

The solution method therefore evidence ω as a function of the wavevector k. The function relating the frequency to the wavevector is known as "dispersion relation".

The possible values of k can be obtained by analysing the periodicity of the solutions in the n^{th} and $(n+N)^{\text{th}}$ atoms, which are of course the same:

$$e^{i[k(n+N)a-\omega t]} = e^{i[kna-\omega t]} \tag{2.15}$$

$$e^{ikNa} = 1 = e^{i2\pi p} \tag{2.16}$$

where $p = 1, 2, 3....$ Therefore:

$$k_p = \frac{2\pi p}{Na}. \tag{2.17}$$

In the dispersion relation, the term $|sin(ka/2)|$ implies a periodic dependency of ω with k, more precisely restarting every time $ka/2 = n\pi$, or $k = n2\pi/a$. Moreover, we see from Eq. 2.12 that the displacement in unchanged if k is changed to $k+n2\pi/a$. It means that more than one wavevector can describe the same atomic displacements, as shown in Fig. 2.5. We usually chose the values of k such that $-\pi/a < k \leq \pi/a$. This region of k values is the one dimensional equivalent of the Brillouin zone in Fig. 2.6. Using the condition Eq. 2.17, we obtain that there is N possible values in this interval. This equals the number of primitive unit cells in the chain. These modes of vibrations are called phonons, from the Greek word φωνή for "voice" and in analogy to "photon" for quanta of light.

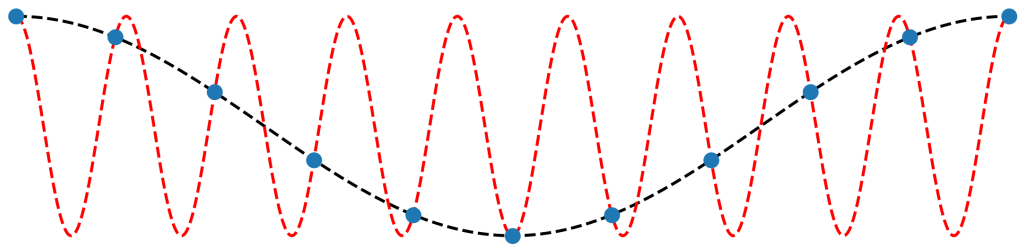


Figure 2.5: Two waves with different wavevectors that can describe the same configuration of the monoatomic chain. Only the one in black is in the first Brillouin zone.

In the limit where the chain becomes infinitely long, all wavevectors would be possible and the dispersion relation becomes virtually continuous. Figure 2.6 shows the possible wavevectors for a chain with 5, 11 and 23 atoms, as well as the continuous solution.

Giving one more step towards complexity, a second type of atom, different from the first, can be added to the chain. Now the chain alternates between atom A and atom B, and the primitive unit cell contains two atoms. When the same solution method than before is applied, the dispersion relation has two solutions, called "branches" or "polarisations", and each value of k has now two associated ω. The number of branches is given by the number of atoms in the primitive unit cell of the crystal, N_a. In 1D therefore are N_a branches, and $3N_a$ in 3D. To consider the different branches, one more index is needed to label the dispersion relation $\omega = f(\mathbf{k}, j)$. We usually write $\omega_{\mathbf{k}j}$.

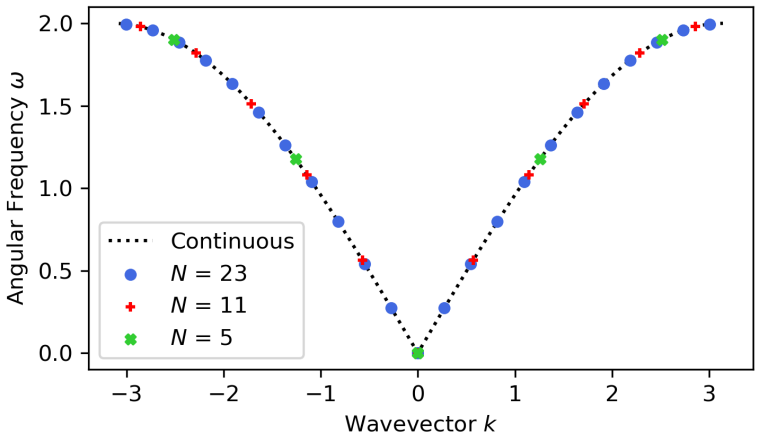


Figure 2.6: Plot of the dispersion relation for a monoatomic chain. m, J and a are equal to 1.

2.1.3 General Definition

Lattice Dynamics

In the previous section the concept of phonon was introduced using a one dimensional toy model. In this section the general theory is reviewed. The main objectives are to derive the phonon frequencies for a generic crystal in 3D space, and to find the mathematical expression for the thermodynamic energy of the crystal.

Let us consider a crystal, where the atoms positions are $\mathbf{r}_{l\tau\alpha}$, where l is the unit cell index, τ the index of the atoms within the unit cell, and α the Cartesian coordinates (x, y or z). We write the atomic positions as $\mathbf{r}_{l\tau\alpha}(t) = \mathbf{r}^0_{l\tau\alpha} + \delta\mathbf{r}_{l\tau\alpha}(t)$, where $\mathbf{r}^0_{l\tau\alpha}$ are the equilibrium positions. Then if E is the interaction energy in between the atoms, we obtain for the Lagrangian of the system

$$\mathcal{L} = \mathcal{T} - \mathcal{V} \tag{2.18}$$

$$= \frac{1}{2}\sum_{l\tau\alpha} m_\tau \delta\dot{\mathbf{r}}^2_{l\tau\alpha} - \frac{1}{2}\sum_{l\tau\alpha}\sum_{l'\tau'\alpha'} \delta\mathbf{r}_{l\tau\alpha}\frac{\partial^2 E}{\partial\mathbf{r}_{l\tau\alpha}\partial\mathbf{r}_{l'\tau'\alpha'}}\delta\mathbf{r}_{l'\tau'\alpha'} \tag{2.19}$$

$$= \mathcal{L}(\{\delta\mathbf{r}_{l\tau\alpha}\},\{\delta\dot{\mathbf{r}}_{l\tau\alpha}\}). \tag{2.20}$$

The Lagrange equations are

$$\frac{d}{dt}\frac{\partial\mathcal{L}}{\partial\delta\dot{\mathbf{r}}_{l\tau\alpha}} - \frac{\partial\mathcal{L}}{\partial\delta\mathbf{r}_{l\tau\alpha}} = 0 \iff m_\tau\delta\ddot{\mathbf{r}}_{l\tau\alpha} = -\sum_{l'\tau'\alpha'}\frac{\partial^2 E}{\partial\mathbf{r}_{l\tau\alpha}\partial\mathbf{r}_{l'\tau'\alpha'}}\delta\mathbf{r}_{l'\tau'\alpha'} \tag{2.21}$$

$$\iff m_\tau\delta\ddot{\mathbf{r}}_{l\tau\alpha} = -\sum_{l'\tau'\alpha'}\varphi_{l\tau\alpha,l'\tau'\alpha'}\delta\mathbf{r}_{l'\tau'\alpha'}. \tag{2.22}$$

To solve this equation, we look for harmonic solutions fulfilling Bloch theorem,

$$\delta\mathbf{r}_{l\tau\alpha} = \delta\mathbf{r}_{\tau\alpha}e^{-i(\omega t - \mathbf{k}\cdot\mathbf{r}_l)}, \tag{2.23}$$

whose real part will fulfil the above differential equation.

Substituting in the previous equation we obtain

$$\omega^2(\sqrt{m_\tau}\delta\mathbf{r}_{\tau\alpha}) = \sum_{\tau'\alpha'}\left(\sum_{l'}\frac{\varphi_{l\tau\alpha,l'\tau'\alpha'}}{\sqrt{m_\tau m_{\tau'}}}e^{i\mathbf{k}\cdot(\mathbf{r}_{l'}-\mathbf{r}_l)}\right)(\sqrt{m_{\tau'}}\delta\mathbf{r}_{\tau'\alpha'}) \tag{2.24}$$

or

$$\omega^2 \mathbf{u}_{\tau\alpha} = \sum_{\tau'\alpha'} D_{\tau\alpha,\tau'\alpha'}(\mathbf{k}) \mathbf{u}_{\tau'\alpha'} \tag{2.25}$$

with

$$\mathbf{u}_{\tau\alpha} \equiv \sqrt{m_\tau}\delta\mathbf{r}_{\tau\alpha} \tag{2.26}$$

$$D_{\tau\alpha,\tau'\alpha'}(\mathbf{k}) = \sum_{l'} \frac{\varphi_{l\tau\alpha,l'\tau'\alpha'}}{\sqrt{m_\tau m_{\tau'}}} e^{i\mathbf{k}\cdot(\mathbf{r}_{l'}-\mathbf{r}_l)}. \tag{2.27}$$

Equation 2.25 is an eigenvalue equation at each $\mathbf{k}$. We denote $\omega^2_{\mathbf{k}j}$, $j = 1, \ldots, 3N_a$ its eigenvalues and $\mathbf{e}^{\mathbf{k}j}_{\tau\alpha}$ its orthonormal eigenvectors,

$$\sum_{\tau\alpha} \mathbf{e}^{\mathbf{k}j'*}_{\tau\alpha} \mathbf{e}^{\mathbf{k}j}_{\tau\alpha} = \delta_{jj'}. \tag{2.28}$$

We have therefore obtained that a general solution to Eq. 2.22 can be written as a superposition

$$\delta\mathbf{r}_{l\tau\alpha}(t) = \frac{1}{\sqrt{N}} \sum_{\mathbf{k}j} \frac{\mathbf{e}^{\mathbf{k}j}_{\tau\alpha}}{\sqrt{m_\tau}} e^{i\mathbf{k}\cdot\mathbf{r}_l} Q_{\mathbf{k}j}(t), \tag{2.29}$$

$$Q_{\mathbf{k}j}(t) = \frac{1}{\sqrt{N}} \sum_{l\tau\alpha} \sqrt{m_\tau} \mathbf{e}^{\mathbf{k}j*}_{\tau\alpha} e^{-i\mathbf{k}\cdot\mathbf{r}_l} \delta\mathbf{r}_{l\tau\alpha}(t) \tag{2.30}$$

where $Q_{\mathbf{k}j}(t)$ is $e^{-i\omega t}$ with $\omega = \pm\omega_{\mathbf{k}j}$. In fact substituting this result into 2.22, and applying $\frac{1}{N}\sum_{l\tau\alpha} \mathbf{e}^{\mathbf{k}'j'*}_{\tau\alpha} e^{-i\mathbf{k}'\cdot\mathbf{r}_l}$ on both sides off the equation, we found that $Q_{\mathbf{k}j}(t)$ is solution of the equation

$$\ddot{Q}_{\mathbf{k}j} + \omega^2_{\mathbf{k}j} Q_{\mathbf{k}j} = 0, \tag{2.31}$$

the differential equation for an harmonic oscillator.

Because $\delta\mathbf{r}_{l\tau\alpha}$ is real, from Eq. 2.29, we obtain, using $\mathbf{e}^{\mathbf{k}j*}_{\tau\alpha} = \mathbf{e}^{-\mathbf{k}j}_{\tau\alpha}$,

$$Q^*_{-\mathbf{k}j} = Q_{\mathbf{k}j}. \tag{2.32}$$

We rewrite the kinetic and potential energies using $Q_{\mathbf{k}j}$.

$$\mathcal{T} = \frac{1}{2} \sum_{l\tau\alpha} m_\tau \delta\dot{\mathbf{r}}^2_{l\tau\alpha} \tag{2.33}$$

$$= \frac{1}{2}\frac{1}{N} \sum_{l\tau\alpha} \sum_{\mathbf{k}j} \sum_{\mathbf{k}'j'} m_\tau \frac{\mathbf{e}^{\mathbf{k}j}_{\tau\alpha}}{\sqrt{m_\tau}} e^{i\mathbf{k}\cdot\mathbf{r}_l} \dot{Q}_{\mathbf{k}j} \frac{\mathbf{e}^{\mathbf{k}'j'}_{\tau\alpha}}{\sqrt{m_\tau}} e^{i\mathbf{k}'\cdot\mathbf{r}_l} \dot{Q}_{\mathbf{k}'j'} \tag{2.34}$$

$$= \frac{1}{2} \sum_{\tau\alpha} \sum_{\mathbf{k}j} \sum_{\mathbf{k}'j'} \mathbf{e}^{\mathbf{k}j}_{\tau\alpha} \mathbf{e}^{\mathbf{k}'j'}_{\tau\alpha} \left(\frac{1}{N} \sum_l e^{i(\mathbf{k}+\mathbf{k}')\cdot\mathbf{r}_l} \right) \dot{Q}_{\mathbf{k}j} \dot{Q}_{\mathbf{k}'j'} \tag{2.35}$$

$$= \frac{1}{2} \sum_{\tau\alpha} \sum_{\mathbf{k}j} \sum_{\mathbf{k}'j'} \mathbf{e}^{\mathbf{k}j}_{\tau\alpha} \mathbf{e}^{\mathbf{k}'j'*}_{\tau\alpha} \left(\frac{1}{N} \sum_l e^{i(\mathbf{k}-\mathbf{k}')\cdot\mathbf{r}_l} \right) \dot{Q}_{\mathbf{k}j} \dot{Q}^*_{\mathbf{k}'j'} \tag{2.36}$$

$$= \frac{1}{2} \sum_{\mathbf{k}j} \dot{Q}_{\mathbf{k}j} \dot{Q}^*_{\mathbf{k}j} \tag{2.37}$$

and

$$\mathcal{V} = \frac{1}{2}\sum_{l\tau\alpha}\sum_{l'\tau'\alpha'}\delta\mathbf{r}_{l\tau\alpha}\phi_{l\tau\alpha,l'\tau'\alpha'}\delta\mathbf{r}_{l'\tau'\alpha'} \tag{2.38}$$

$$= \frac{1}{2}\frac{1}{N}\sum_{l\tau\alpha}\sum_{l'\tau'\alpha'}\sum_{\mathbf{k}j}\sum_{\mathbf{k}'j'}\frac{\mathbf{e}^{\mathbf{k}j}_{\tau\alpha}}{\sqrt{m_\tau}}e^{i\mathbf{k}\cdot\mathbf{r}_l}Q_{\mathbf{k}j}\varphi_{l\tau\alpha,l'\tau'\alpha'}\frac{\mathbf{e}^{\mathbf{k}'j'}_{\tau'\alpha'}}{\sqrt{m_{\tau'}}}e^{i\mathbf{k}'\cdot\mathbf{r}_{l'}}Q_{\mathbf{k}'j'} \tag{2.39}$$

$$= \frac{1}{2}\frac{1}{N}\sum_{l\tau\alpha}\sum_{\tau'\alpha'}\sum_{\mathbf{k}j}\sum_{\mathbf{k}'j'}\frac{\mathbf{e}^{\mathbf{k}j}_{\tau\alpha}}{\sqrt{m_\tau}}e^{i(\mathbf{k}+\mathbf{k}')\cdot\mathbf{r}_l}\left(\sum_{l'}\varphi_{l\tau\alpha,l'\tau'\alpha'}e^{i\mathbf{k}'\cdot(\mathbf{r}_{l'}-\mathbf{r}_l)}\right)\frac{\mathbf{e}^{\mathbf{k}'j'}_{\tau'\alpha'}}{\sqrt{m_{\tau'}}}Q_{\mathbf{k}j}Q_{\mathbf{k}'j'} \tag{2.40}$$

$$= \frac{1}{2}\sum_{\tau\alpha}\sum_{\tau'\alpha'}\sum_{\mathbf{k}j}\sum_{\mathbf{k}'j'}\left(\frac{1}{N}\sum_{l}e^{i(\mathbf{k}+\mathbf{k}')\cdot\mathbf{r}_l}\right)\left(\sum_{l'}\frac{\varphi_{0\tau\alpha,l'\tau'\alpha'}}{\sqrt{m_\tau m_{\tau'}}}e^{i\mathbf{k}'\cdot\mathbf{r}_{l'}}\right)\mathbf{e}^{\mathbf{k}'j'}_{\tau'\alpha'}\mathbf{e}^{\mathbf{k}j}_{\tau\alpha}Q_{\mathbf{k}j}Q_{\mathbf{k}'j'} \tag{2.41}$$

$$= \frac{1}{2}\sum_{\tau\alpha}\sum_{\mathbf{k}j}\sum_{\mathbf{k}'j'}\left(\frac{1}{N}\sum_{l}e^{i(\mathbf{k}+\mathbf{k}')\cdot\mathbf{r}_l}\right)\omega^2_{\mathbf{k}'j'}\mathbf{e}^{\mathbf{k}'j'}_{\tau\alpha}\mathbf{e}^{\mathbf{k}j}_{\tau\alpha}Q_{\mathbf{k}j}Q_{\mathbf{k}'j'} \tag{2.42}$$

$$= \frac{1}{2}\sum_{\tau\alpha}\sum_{\mathbf{k}j}\sum_{\mathbf{k}'j'}\left(\frac{1}{N}\sum_{l}e^{i(\mathbf{k}-\mathbf{k}')\cdot\mathbf{r}_l}\right)\omega^2_{\mathbf{k}'j'}\mathbf{e}^{\mathbf{k}'j'*}_{\tau\alpha}\mathbf{e}^{\mathbf{k}j}_{\tau\alpha}Q_{\mathbf{k}j}Q^*_{\mathbf{k}'j'} \tag{2.43}$$

$$= \frac{1}{2}\sum_{\mathbf{k}j}\omega^2_{\mathbf{k}j}Q_{\mathbf{k}j}Q^*_{\mathbf{k}j} \tag{2.44}$$

For the Lagrangian we obtain

$$\mathcal{L} = \mathcal{T} - \mathcal{V} = \frac{1}{2}\sum_{\mathbf{k}j}\dot{Q}_{\mathbf{k}j}\dot{Q}^*_{\mathbf{k}j} - \omega^2_{\mathbf{k}j}Q_{\mathbf{k}j}Q^*_{\mathbf{k}j}. \tag{2.45}$$

The independent variables are now $Q_{\mathbf{k}j}$ and $Q^*_{\mathbf{k}j}$. The momentum canonically conjugate to $Q^*_{\mathbf{k}j}$ is

$$P_{\mathbf{k}j} = \frac{\partial\mathcal{L}}{\partial\dot{Q}^*_{\mathbf{k}j}} = \dot{Q}_{\mathbf{k}j} \tag{2.46}$$

$$= \frac{1}{\sqrt{N}}\sum_{l\tau\alpha}\sqrt{m_\tau}\mathbf{e}^{\mathbf{k}j*}_{\tau\alpha}e^{-i\mathbf{k}\cdot\mathbf{r}_l}\delta\dot{\mathbf{r}}_{l\tau\alpha}(t) \tag{2.47}$$

$$= \frac{1}{\sqrt{N}}\sum_{l\tau\alpha}\frac{\mathbf{e}^{\mathbf{k}j*}_{\tau\alpha}}{\sqrt{m_\tau}}e^{-i\mathbf{k}\cdot\mathbf{r}_l}\mathbf{p}_{l\tau\alpha}(t) \tag{2.48}$$

and the momentum canonically conjugate to $Q_{\mathbf{k}j}$ is

$$P^*_{\mathbf{k}j} = \frac{\partial\mathcal{L}}{\partial\dot{Q}_{\mathbf{k}j}} = \dot{Q}^*_{\mathbf{k}j} \tag{2.49}$$

$$= \frac{1}{\sqrt{N}}\sum_{l\tau\alpha}\frac{\mathbf{e}^{\mathbf{k}j}_{\tau\alpha}}{\sqrt{m_\tau}}e^{i\mathbf{k}\cdot\mathbf{r}_l}\mathbf{p}_{l\tau\alpha}(t) \tag{2.50}$$

with

$$Q^*_{\mathbf{k}j}(t) = \frac{1}{\sqrt{N}}\sum_{l\tau\alpha}\sqrt{m_\tau}\mathbf{e}^{\mathbf{k}j}_{\tau\alpha}e^{i\mathbf{k}\cdot\mathbf{r}_l}\delta\mathbf{r}_{l\tau\alpha}(t) \tag{2.51}$$

Therefore we obtain for the Hamiltonian

$$\mathcal{H} = \mathcal{T} + \mathcal{V} = \frac{1}{2}\sum_{\mathbf{k}j} P_{\mathbf{k}j}P^*_{\mathbf{k}j} + \omega^2_{\mathbf{k}j}Q_{\mathbf{k}j}Q^*_{\mathbf{k}j}. \tag{2.52}$$

The differential equation for the harmonic oscillators, Eq. 2.31 can now be solved using for initial conditions

$$Q_{\mathbf{k}j}(t=0) = Q_{\mathbf{k}j}(0), \tag{2.53}$$
$$\dot{Q}_{\mathbf{k}j}(t=0) = P_{\mathbf{k}j}(t=0) = P_{\mathbf{k}j}(0). \tag{2.54}$$

We have

$$Q_{\mathbf{k}j}(t) = A_{\mathbf{k}j}e^{-i\omega_{\mathbf{k}j}t} + B_{\mathbf{k}j}e^{i\omega_{\mathbf{k}j}t} \tag{2.55}$$

with

$$Q_{\mathbf{k}j}(0) = A_{\mathbf{k}j} + B_{\mathbf{k}j} \tag{2.56}$$
$$P_{\mathbf{k}j}(0) = -i\omega_{\mathbf{k}j}A_{\mathbf{k}j} + i\omega_{\mathbf{k}j}B_{\mathbf{k}j} \tag{2.57}$$

and therefore

$$Q_{\mathbf{k}j}(t) = \frac{1}{2}\left[Q_{\mathbf{k}j}(0) - \frac{1}{i\omega_{\mathbf{k}j}}P_{\mathbf{k}j}(0)\right]e^{-i\omega_{\mathbf{k}j}t} + \frac{1}{2}\left[Q_{\mathbf{k}j}(0) + \frac{1}{i\omega_{\mathbf{k}j}}P_{\mathbf{k}j}(0)\right]e^{i\omega_{\mathbf{k}j}t} \tag{2.58}$$

Quantum mechanics

The transition to quantum mechanics is done promoting the position and momentum, $\mathbf{r}_{l\tau\alpha}$ and $\mathbf{p}_{l\tau\alpha} = m_\tau\dot{\mathbf{r}}_{l\tau\alpha}$, to operators fulfilling the commutation relations

$$[\mathbf{r}_{l\tau\alpha}, \mathbf{p}_{l'\tau'\alpha'}] = [\delta\mathbf{r}_{l\tau\alpha}, \mathbf{p}_{l'\tau'\alpha'}] = i\hbar\delta_{ll'}\delta_{\tau\tau'}\delta_{\alpha\alpha'}. \tag{2.59}$$

The operators associated to $Q_{\mathbf{k}j}$, $P_{\mathbf{k}j}$, $Q^*_{\mathbf{k}j}$, $P^*_{\mathbf{k}j}$ are defined by the expansions in Eqs. 2.30, 2.48, 2.51 and 2.50. Because $\mathbf{r}_{l\tau\alpha}$ and $\mathbf{p}_{l\tau\alpha}$ are hermitian operators, it is clear that $Q^*_{\mathbf{k}j} = Q^\dagger_{\mathbf{k}j}$ and $P^*_{\mathbf{k}j} = P^\dagger_{\mathbf{k}j}$. Using those expansions we also obtain the commutations between conjugate variables,

$$[Q^*_{\mathbf{k}j}, P_{\mathbf{k}'j'}] = i\hbar\Delta(\mathbf{k}-\mathbf{k}')\delta_{jj'} \tag{2.60}$$
$$[Q_{\mathbf{k}j}, P^*_{\mathbf{k}'j'}] = i\hbar\Delta(\mathbf{k}-\mathbf{k}')\delta_{jj'} \tag{2.61}$$

where $\Delta(\mathbf{k}-\mathbf{k}')$ is 1 if $\mathbf{k}-\mathbf{k}'$ is a reciprocal lattice vector, 0 otherwise.

It is useful to write $Q_{\mathbf{k}j}$ and $P_{\mathbf{k}j}$ as

$$Q_{\mathbf{k}j} = \sqrt{\frac{\hbar}{2\omega_{\mathbf{k}j}}}(a_{\mathbf{k}j} + a^\dagger_{-\mathbf{k}j}) \tag{2.62}$$

$$P_{\mathbf{k}j} = -i\sqrt{\frac{\hbar\omega_{\mathbf{k}j}}{2}}(a_{\mathbf{k}j} - a^\dagger_{-\mathbf{k}j}) \tag{2.63}$$

with

$$a_{\mathbf{k}j} = \sqrt{\frac{\omega_{\mathbf{k}j}}{2\hbar}}\left(Q_{\mathbf{k}j} + \frac{i}{\omega_{\mathbf{k}j}}P_{\mathbf{k}j}\right) \tag{2.64}$$

$$a^\dagger_{-\mathbf{k}j} = \sqrt{\frac{\omega_{\mathbf{k}j}}{2\hbar}}\left(Q_{\mathbf{k}j} - \frac{i}{\omega_{\mathbf{k}j}}P_{\mathbf{k}j}\right) \tag{2.65}$$

From the the definitions of the Q and P operators we have $Q_{\mathbf{k}j} = Q^{\dagger}_{-\mathbf{k}j}$ and $P_{\mathbf{k}j} = P^{\dagger}_{-\mathbf{k}j}$, and this property is automatically fulfilled by this representation. Also we have

$$[a_{\mathbf{k}j}, a^{\dagger}_{\mathbf{k}'j'}] = \sqrt{\frac{\omega_{\mathbf{k}j}}{2\hbar}}\sqrt{\frac{\omega_{\mathbf{k}'j'}}{2\hbar}}[Q_{\mathbf{k}j} + \frac{i}{\omega_{\mathbf{k}j}}P_{\mathbf{k}j}, Q_{-\mathbf{k}'j'}, -\frac{i}{\omega_{-\mathbf{k}'j'}}P_{-\mathbf{k}'j'}] \tag{2.66}$$

$$= \sqrt{\frac{\omega_{\mathbf{k}j}}{2\hbar}}\sqrt{\frac{\omega_{\mathbf{k}'j'}}{2\hbar}}\Big([Q_{\mathbf{k}j} - \frac{i}{\omega_{\mathbf{k}'j'}}P^{\dagger}_{\mathbf{k}'j'}] + [\frac{i}{\omega_{\mathbf{k}j}}P_{\mathbf{k}j}, Q^{\dagger}_{\mathbf{k}'j'}]\Big) \tag{2.67}$$

$$= \Delta(\mathbf{k} - \mathbf{k}')\delta_{jj'}. \tag{2.68}$$

The Hamiltonian can be written in term of those operators. We have

$$P_{\mathbf{k}j}P^{*}_{\mathbf{k}'j'} = \frac{\hbar}{2}\sqrt{\omega_{\mathbf{k}j}\omega_{\mathbf{k}'j'}}(a_{\mathbf{k}j}a^{\dagger}_{\mathbf{k}'j'} - a_{\mathbf{k}j}a_{-\mathbf{k}'j'} - a^{\dagger}_{-\mathbf{k}j}a^{\dagger}_{\mathbf{k}'j'} + a^{\dagger}_{-\mathbf{k}j}a_{-\mathbf{k}'j'}) \tag{2.69}$$

$$\omega_{\mathbf{k}j}\omega_{\mathbf{k}'j'}Q_{\mathbf{k}j}Q^{*}_{\mathbf{k}'j'} = \frac{\hbar}{2}\sqrt{\omega_{\mathbf{k}j}\omega_{\mathbf{k}'j'}}(a_{\mathbf{k}j}a^{\dagger}_{\mathbf{k}'j'} + a_{\mathbf{k}j}a_{-\mathbf{k}'j'} + a^{\dagger}_{-\mathbf{k}j}a^{\dagger}_{\mathbf{k}'j'} + a^{\dagger}_{-\mathbf{k}j}a_{-\mathbf{k}'j'}) \tag{2.70}$$

Rewriting the potential energy Eq. 2.43 in a more symmetric form, $\omega^2_{\mathbf{k}'j'} \to \omega_{\mathbf{k}j}\omega_{\mathbf{k}'j'}$, we obtain

$$\mathcal{H} = \mathcal{T} - \mathcal{V} \tag{2.71}$$

$$= \frac{1}{2}\sum_{\tau\alpha}\sum_{\mathbf{k}j}\sum_{\mathbf{k}'j'}\mathbf{e}^{\mathbf{k}j}_{\tau\alpha}\mathbf{e}^{\mathbf{k}'j'*}_{\tau\alpha}\Big(\frac{1}{N}\sum_{l}e^{i(\mathbf{k}-\mathbf{k}')\cdot\mathbf{r}_l}\Big)\hbar\sqrt{\omega_{\mathbf{k}j}\omega_{\mathbf{k}'j'}}(a_{\mathbf{k}j}a^{\dagger}_{\mathbf{k}'j'} + a^{\dagger}_{-\mathbf{k}j}a_{-\mathbf{k}'j'}) \tag{2.72}$$

If we use

$$\frac{1}{N}\sum_{l}e^{i(\mathbf{k}-\mathbf{k}')\cdot\mathbf{r}_l} = \delta_{\mathbf{k},\mathbf{k}'} \tag{2.73}$$

then

$$\mathcal{H} = \frac{1}{2}\sum_{\mathbf{k}j}\hbar\omega_{\mathbf{k}j}(a_{\mathbf{k}j}a^{\dagger}_{\mathbf{k}j} + a^{\dagger}_{-\mathbf{k}j}a_{-\mathbf{k}j}) \tag{2.74}$$

$$= \frac{1}{2}\sum_{\mathbf{k}j}\hbar\omega_{\mathbf{k}j}\Big[a^{\dagger}_{\mathbf{k}j}a_{\mathbf{k}j} + a_{\mathbf{k}j}a^{\dagger}_{\mathbf{k}j}\Big] \tag{2.75}$$

$$= \sum_{\mathbf{k}j}\hbar\omega_{\mathbf{k}j}\Big[a^{\dagger}_{\mathbf{k}j}a_{\mathbf{k}j} + \frac{1}{2}\Big] \tag{2.76}$$

which is the familiar result.

The Hamiltonian we obtained is the quantum mechanical operator associated to the energy. To obtain the thermodynamic energy E, we must compute the sum of eigenvalues weighted by the (normalised) Boltzmann factor.

It is well known that the eigenvalue of $a^{\dagger}_{\mathbf{k}j}a_{\mathbf{k}j}$, with $a^{\dagger}_{\mathbf{k}j}$ and $a_{\mathbf{k}j}$ fulfilling the commutation relations 2.68, are the positive integers [66], $n_{\mathbf{k}j} = 0, 1, 2, \dots$. The eigenvalues of the Hamiltonian are therefore

$$E_{\{n_{\mathbf{k}j}\}} = \sum_{\mathbf{k}j}\hbar\omega_{\mathbf{k}j}\Big[n_{\mathbf{k}j} + \frac{1}{2}\Big], \tag{2.77}$$

where $\{n_{\mathbf{k}j}\}$ represent all possible configuration of occupations $n_{\mathbf{k}j}$ of the modes $\mathbf{k}j$. The thermodynamic energy is therefore

$$E = \frac{\sum_{\{n_{\mathbf{k}j}\}} E_{\{n_{\mathbf{k}j}\}} \exp\Big(- \beta E_{\{n_{\mathbf{k}j}\}}\Big)}{\sum_{\{n_{\mathbf{k}j}\}} \exp\Big(- \beta E_{\{n_{\mathbf{k}j}\}}\Big)} \tag{2.78}$$

with $\beta = 1/k_b T$. We rewrite the energy as

$$E = -\frac{\partial \ln Z}{\partial \beta}, \tag{2.79}$$

with the partition function

$$Z = \sum_{\{n_{\mathbf{k}j}\}} \exp\Big(- \beta E_{\{n_{\mathbf{k}j}\}}\Big). \tag{2.80}$$

Fortunately Z is easily computed, what will allow to obtain the energy. We have

$$Z = \sum_{n_{\mathbf{k}_1 j_1}=0}^{+\infty} \cdots \sum_{n_{\mathbf{k}_N j_N}=0}^{+\infty} \exp\Big(- \beta \sum_{\mathbf{k}j} \hbar\omega_{\mathbf{k}j}\Big[n_{\mathbf{k}j} + \frac{1}{2}\Big]\Big) \tag{2.81}$$

$$= \prod_{\mathbf{k}j} e^{-\beta\frac{\hbar\omega_{\mathbf{k}j}}{2}} \sum_{n_{\mathbf{k}j}=0}^{+\infty} \Big(e^{-\beta\hbar\omega_{\mathbf{k}j}}\Big)^{n_{\mathbf{k}j}} \tag{2.82}$$

$$= \prod_{\mathbf{k}j} \frac{1}{2\sinh\beta\frac{\hbar\omega_{\mathbf{k}j}}{2}}. \tag{2.83}$$

This gives

$$E = -\frac{\partial \ln Z}{\partial \beta} \tag{2.84}$$

$$= \sum_{\mathbf{k}j} \hbar\omega_{\mathbf{k}j}\Big(\frac{1}{e^{\beta\hbar\omega_{\mathbf{k}j}} - 1} + \frac{1}{2}\Big) \tag{2.85}$$

$$= \sum_{\mathbf{k}j} \hbar\omega_{\mathbf{k}j}\Big(n^0_{\mathbf{k}j} + \frac{1}{2}\Big) \tag{2.86}$$

with

$$n^0_{\mathbf{k}j} = \frac{1}{\exp[\beta\hbar\omega_{\mathbf{k}j}] - 1} \tag{2.87}$$

the Bose-Einstein distribution function. In the following we will rather use the energy density which is

$$e = \frac{E}{V} = \frac{1}{NV_0} \sum_{\mathbf{k}j} \hbar\omega_{\mathbf{k}j}\Big(n^0_{\mathbf{k}j} + \frac{1}{2}\Big) \tag{2.88}$$

where N is the number of wavevectors in the FBZ and V_0 is the volume of the unit cell. The factor NV_0 is therefore the total volume of the solid.

2.1.4 Energy transport by phonons

In the previous section the Hamiltonian of the system has been written as a sum of independent harmonic oscillators. The total eigenstates of the system are therefore a product of the eigenstates of each of those oscillators. However, when the system is subject to an external field, such as a temperature gradient, the wavefunction of the system is not an eigenstate, but a linear combination of them.

In the realm of semi-classical physics, as for example with the Boltzmann equation to be discussed in sec. 2.3, a particle representation is needed. Therefore it is customary to assume that the wavefunction of the system is made of wavepacket which are well localised in reciprocal space as well as in direct space. The localization in reciprocal space allows to continue labeling the states using wavevectors, and the localization in direction space make the link with the particle concept of classical physics.

One way to build such wavepacket is to write the atomic displacements from the linear combination of the eigenmodes. Using Eq. 2.29 with Eq. 2.55 and noticing that $A^*_{-\mathbf{k}j} = B_{\mathbf{k}j}$, we obtain

$$\delta\mathbf{r}_{l\tau\alpha}(t) = 2\mathrm{Re}\left(\frac{1}{\sqrt{N}} \sum_{\mathbf{k}j} \frac{\mathbf{e}^{\mathbf{k}j}_{\tau\alpha}}{\sqrt{m_\tau}} e^{i(\mathbf{k}\cdot\mathbf{r}_l - \omega_{\mathbf{k}j}t)} A_{\mathbf{k}j} \right) \tag{2.89}$$

$$= \mathrm{Re}\left(\sum_j \int \frac{d^3\mathbf{k}}{(2\pi)^3} \mathbf{e}^{\mathbf{k}j}_{\tau\alpha} e^{i(\mathbf{k}\cdot\mathbf{r}_l - \omega_{\mathbf{k}j}t)} a_{\mathbf{k}j} \right) \tag{2.90}$$

with $a_{\mathbf{k}j} = 2\ V_0 A_{\mathbf{k}j}/\sqrt{N/m_\tau}$.

In a wavepacket the coefficients $a_{\mathbf{k}j}$ are chosen to be non zero around a wavevector $\mathbf{k}_0$ and for a band index j_0, but also such that the atomic displacements $\delta\mathbf{r}_{l\tau\alpha}(t)$ are localised in real space. The structure of such wavepacket can be better understood using the stationary phase approximation. The displacement are written considering only the band j_0,

$$\delta\mathbf{r}_{l\tau\alpha}(t) = \mathrm{Re}\left(\int \frac{d^3\mathbf{k}}{(2\pi)^3} |\mathbf{e}^{\mathbf{k}j_0}_{\tau\alpha}||a_{\mathbf{k}j_0}| e^{i(\mathbf{k}\cdot\mathbf{r}_l - \omega_{\mathbf{k}j_0}t + \phi^{\mathbf{k}j_0}_{\tau\alpha})} \right) \tag{2.91}$$

where $\phi^{\mathbf{k}j_0}_{\tau\alpha}$ is the phase of $\mathbf{e}^{\mathbf{k}j_0}_{\tau\alpha} a_{\mathbf{k}j_0}$.

In the above integral, if the phase vary significantly the exponential will oscillate, and the integral will average to zero. Therefore the dominant contribution to $\delta\mathbf{r}_{l\tau\alpha}(t)$ will come from the neighborhood of the points $\mathbf{k}_0$ where the phase is stationary. If we consider only one such point, this condition is written as

$$\nabla_{\mathbf{k}} \big(\mathbf{k}\cdot\mathbf{r}_l - \omega_{\mathbf{k}j}t + \phi^{\mathbf{k}j}_{\tau\alpha} \big) \Big|_{\mathbf{k}=\mathbf{k}_0} = 0, \tag{2.92}$$

or

$$\mathbf{r}_l = \nabla_{\mathbf{k}}\omega_{\mathbf{k}_0 j_0} t - \nabla_{\mathbf{k}}\phi^{\mathbf{k}_0 j_0}_{\tau\alpha} = \mathbf{v}_{\mathbf{k}_0 j_0} t + \mathbf{r}_l(t=0) \tag{2.93}$$

where

$$\mathbf{v}_{\mathbf{k}j} = \nabla_{\mathbf{k}}\omega_{\mathbf{k}j} \tag{2.94}$$

is the group velocity of state $\mathbf{k}j$. We will see below the origin of this name.

The previous results shows which wavevector $\mathbf{k}_0$ contribute most to the displacement $\delta\mathbf{r}_{l\tau\alpha}(t)$. The argument can of course be reserved, and for a given wavevector $\mathbf{k}$ the previous results shows which atom will obtain most of the contribution.

Let us now assume that $|a_{\mathbf{k}j}|$ has a strong maximum around $\mathbf{k}_0$. Then the frequency can be expanded around this wavevector,

$$\omega_{\mathbf{k}j_0} \approx \omega_{\mathbf{k}_0 j_0} + (\mathbf{k} - \mathbf{k_0}) \cdot \mathbf{v}_{\mathbf{k}_0 j_0}. \tag{2.95}$$

This gives

$$\begin{aligned}
\delta\mathbf{r}_{l\tau\alpha}(t) &\approx \mathrm{Re}\bigg(\int \frac{d^3\mathbf{k}}{(2\pi)^3} \mathbf{e}_{\tau\alpha}^{\mathbf{k}j_0} a_{\mathbf{k}j_0} e^{i(\mathbf{k}\cdot\mathbf{r}_l - \omega_{\mathbf{k}_0 j_0} t - (\mathbf{k}-\mathbf{k_0})\cdot\mathbf{v}_{\mathbf{k}_0 j_0} t)} \bigg) && (2.96)\\
&\approx \mathrm{Re}\bigg(\mathbf{e}_{\tau\alpha}^{\mathbf{k}_0 j_0} e^{i(\mathbf{k}_0\cdot\mathbf{r}_l - \omega_{\mathbf{k}_0 j_0} t)} \int \frac{d^3\mathbf{k}}{(2\pi)^3} a_{\mathbf{k}j_0} e^{i(\mathbf{k}-\mathbf{k}_0)\cdot(\mathbf{r}_l - \mathbf{v}_{\mathbf{k}_0 j_0} t)} \bigg) && (2.97)\\
&\equiv \mathrm{Re}\big(\mathbf{e}_{\tau\alpha}^{\mathbf{k}_0 j_0} e^{i(\mathbf{k}_0\cdot\mathbf{r}_l - \omega_{\mathbf{k}_0 j_0} t)} F_{\mathbf{k}_0 j_0}(\mathbf{r}_l - \mathbf{v}_{\mathbf{k}_0 j_0} t) \big). && (2.98)
\end{aligned}$$

The displacement field therefore appears the eigen displacement of the mode $\mathbf{k}_0 j_0$, time an envelop which propagate at the *group* velocity $\mathbf{v}_{\mathbf{k}_0 j_0}$.

Using this result, we are able to obtain the energy flux through the crystal. To have a non zero net flux, the system must be out of thermodynamic equilibrium. In Eq. 2.86 the energy was obtained assuming a thermodynamic equilibrium. For a system out of equilibrium it can be written the same way

$$e = \frac{1}{NV_0} \sum_{\mathbf{k}j} \hbar\omega_{\mathbf{k}j} \left[n_{\mathbf{k}j} + \frac{1}{2} \right], \tag{2.99}$$

but using an arbitrary $n_{\mathbf{k}j}$ to be determined latter from the Boltzmann equation. n drops its 0 superscript to indicate that it is out of equilibrium. We have obtained that the phonons wavepacket carry their energy with velocity $\mathbf{v}$. The total heat flux $\mathbf{\Phi}$ is therefore

$$\begin{aligned}
\mathbf{\Phi} &= \frac{1}{NV_0} \sum_{\mathbf{k}j} \hbar\omega_{\mathbf{k}j} \left[n_{\mathbf{k}j} + \frac{1}{2} \right] \mathbf{v}_{\mathbf{k}j} && (2.100)\\
&= \frac{1}{NV_0} \sum_{\mathbf{k}j} \hbar\omega_{\mathbf{k}j} n_{\mathbf{k}j} \mathbf{v}_{\mathbf{k}j} && (2.101)
\end{aligned}$$

2.1.5 Scattering

When phonons travel, they do not travel forever. There are interactions between modes that redistribute the energy such that the number of phonons tends to the Bose-Einstein distribution, naturally taking the system to equilibrium. These interactions are called phonon-phonon scattering, due to anharmonicities in the potential field around an atom. Quantum mechanically, the phonon-phonon scattering can be interpreted as the annihilation and creation of phonons. In this work we only considered interactions involving three phonons. Higher order interactions surely exist, but are usually considered negligible.

In Section 2.1.2, the harmonic approximation was used to simplify the problem and bring it closer to an harmonic oscillator, which is well known and easier to solve.

The potential forces generated by an atom, however, do not obey the parabolic shape.

The three phonon processes are represented by the cubic term of the Hamiltonian, four phonon processes by the quartic, and so on. The three phonon interaction has the condition [10]:

$$\mathbf{k} + \mathbf{k}' + \mathbf{k}'' = \mathbf{G} \tag{2.102}$$

This relation can describe the generation of one new phonon with wavevector $-\mathbf{k}''$ due to the collision of two phonons $\mathbf{k}$ and $\mathbf{k}'$, or the scattering of a single phonon $\mathbf{k}$ that generates other two, $-\mathbf{k}'$ and $-\mathbf{k}''$. Let's discuss the first case. When $\mathbf{G} = 0$, the new phonon $-\mathbf{k}''$ has its wavevector inside the FBZ, the scattering process is called a "normal process" or N-process. When $\mathbf{G} \neq 0$, one of the wavevectors would fall outside the FBZ and, because of the periodicity of the dispersion relation, are "brought back" to the FBZ, being "translated" by $\mathbf{G}$. These are called "*Umklapp* processes" ("to flip over" in German), or U-processes. Figure 2.7 illustrates both of them.

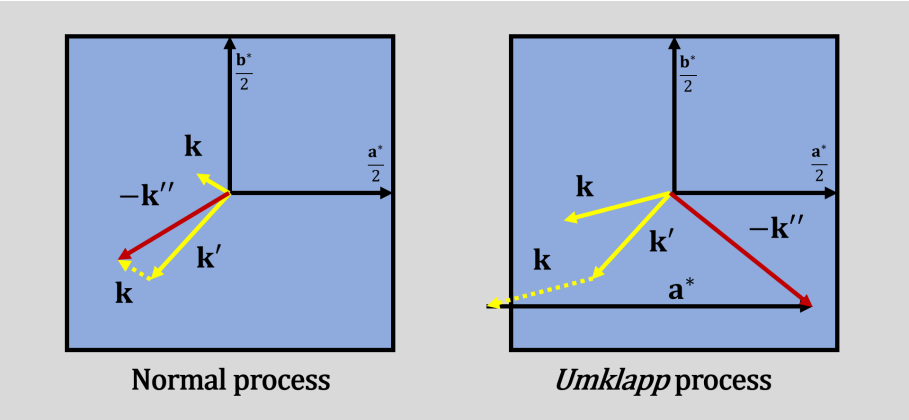


Figure 2.7: Schematics of a N-process (left) and an U-process (right) where $\mathbf{k}+\mathbf{k}'+\mathbf{k}''=\mathbf{G}$. The blue box represents the first Brillouin zone.

A parameter that is usually used to model this behaviour is the relaxation time τ. It represents the average time that a phonon quasiparticle "survives" before scattering. This time is different for each mode and varies with temperature, hence $\tau = f(\mathbf{k}, j, T)$. Alternatively, the mean free path $\Lambda = \|\mathbf{v}\|\tau$ is also used. At high temperatures, atoms vibrate more, and phonons cannot travel as long without scattering; at low temperatures, the reduced vibrations of the lattice allow waves to travel much farther without being scattered.

The use of τ or Λ is often employed for the estimation of the thermal conductvity in nanocomponents, considering not only the phonon-phonon scattering, but also the boundary scattering. The thermal conductivity, assumed to be isotropic, can be estimated by [10]:

$$\kappa = \frac{1}{3} C \overline{\|\mathbf{v}\|} \Lambda \tag{2.103}$$

where C is the total specific heat and with:

$$\Lambda = \overline{\|\mathbf{v}\|\tau} \tag{2.104}$$

The overall relaxation time is usually calculated by the Mathiessen rule, from the relaxation times for normal processes τ_N, for *Umklapp* processes τ_U, and for other

processes that may affect the transport, τ_i:

$$\frac{1}{\tau} = \frac{1}{\tau_N} + \frac{1}{\tau_U} + \frac{1}{\tau_i} \tag{2.105}$$

In the case of nanowires, for example, the estimation by phonon-phonon scattering may be enough for ambient temperatures and higher. For low temperatures, the relaxation time for phonon-phonon scattering alone does not give an accurate measurement, since Λ would tend to infinity as T approaches zero. This is surely not true, since the trajectory of the phonons are often interrupted by the encounter of the boundary of the solid. In a rough nanowire with diameter D, this causes $\Lambda \to D$ if there is not other scattering mechanism, and influences κ directly.

The nature of the surface is also of importance. Phonons with longer wavelengths tend to ignore small scale imperfections on the boundary while being reflected, while shorter wavelength phonons can be highly deviated from their initial trajectory. Ziman [10] brought an analysis that was initially deduced for long wavelength phonons with normal incidence and later generalised by Soffer [38] for oblique incidences. In this model, the specularity parameter p is defined as:

$$p = \exp[-(2\pi\|\mathbf{k}\|\eta\cos\theta)^2] \tag{2.106}$$

where η is the roughness of the surface (to be defined below). The incidence angle θ is the angle between the phonon velocity and the normal of the surface $\mathbf{n}$, oriented outwards by convention:

$$\cos\theta = \frac{\mathbf{v}\cdot\mathbf{n}}{\|\mathbf{v}\|} \tag{2.107}$$

The surface is assumed to be Gaussian, i.e. points on the surface are randomly deviated from the mean plane according to a normal distribution. The roughness η is thence the standard deviation of this distribution.

The specularity parameter can be interpreted as the probability of a phonon wavepacket of mode $(\mathbf{k}j)$ undergoing a specular (mirror-like) reflection to mode $(\mathbf{k}'j')$, keeping the same frequency and velocity parallel to the surface. Classically, it is the portion of the wavepacket's energy that follows Snell's law of reflection, it is:

$$\omega' = \omega \tag{2.108a}$$

$$\mathbf{v}' = \mathbf{v} - 2\mathbf{n}(\mathbf{v}\cdot\mathbf{n}) \tag{2.108b}$$

The rest of the wavepacket's energy are dispersed to other modes. Figure 2.8 shows an illustration of this distribution of energy. The higher the roughness of the surface, the lower is the specularity.

2.2 Ab-initio phonon data

Given the theory about lattice vibrations, in principle it would be possible to calculate the phonon modes by simulating the atomic structure of a crystal. It is possible indeed, using atomic properties to calculate the dispersion relation and relaxation times, what is called *ab-initio* simulations.

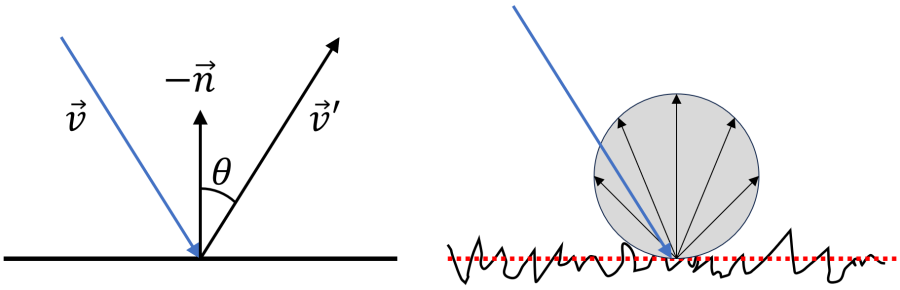


Figure 2.8: Illustration of a specular (left) and a diffuse (right) reflection.

The phonon frequencies and group velocities, $\omega_{\mathbf{k}j}$ and $\mathbf{v}_{\mathbf{k}j}$, can be computed from lattice dynamics [67], knowing the harmonic part of the interaction potential between atoms as shown in Eqs. 2.27 and 2.94. The phonon lifetime $\tau_{\mathbf{k}j}$ is calculated with the many-body perturbation theory [68], knowing the anharmonic part of the interaction potential. The harmonic and anharmonic part of the interaction potential are represented by harmonic and anharmonic force constants, which are the second and third derivatives of the interaction energy with respect to the atomic displacements. This interaction energy can be computed quantum mechanically within the framework of Density Functional Theory (DFT), and the derivatives can performed numerically from finite displacements, by moving the atoms around their equilibrium positions. This strategy has recently been implemented within the Phono3py computer code [36]. For a given bulk material, the results of such calculations are stored in a HDF5 file, which contains the $\mathbf{v}$, ω and $\tau(T)$ values for each combination of the band index j, and wavevector $\mathbf{k}$ within the irreducible part of the first Brillouin zone [69].

By having the values of $\mathbf{k}$, ω, $\mathbf{v}$ and τ for a large enough number of modes in the FBZ, a calculation can be performed to approximate how energy is carried by phonons in a given material.

2.3 The Boltzmann Transport Equation

The energy density and the heat flux of a system out of equilibrium, Eq. 2.87 and Eq. 2.101 have been expressed using the number of phonon $n_{\mathbf{k}j}$ in mode $\mathbf{k}j$. $n_{\mathbf{k}j}$ can be obtained solving the Boltzmann equation,

$$\frac{\partial n_{\mathbf{k}j}}{\partial t} + \mathbf{v}_{\mathbf{k}j} \cdot \nabla_{\mathbf{r}} n_{\mathbf{k}j} = \left.\frac{\partial n_{\mathbf{k}j}}{\partial t}\right|_{scat}. \tag{2.109}$$

As said previously, $n_{\mathbf{k}j} = n_{\mathbf{k}j}(\mathbf{r}, t)$ is the average number of phonons in mode $(\mathbf{k}j)$. The number of phonons, at time t, in mode $(\mathbf{k}j)$, in a small volume $d^3\mathbf{r}$ around point $\mathbf{r}$, with wavevector in the range $d^3\mathbf{k}$ around $\mathbf{k}$ is $n_{\mathbf{k}j} d^3\mathbf{r} d^3\mathbf{k}/(2\pi)^3$. ω is the angular frequency, and the sub-index *scat* refers to the variation of n due to the scattering of the phonons in mode $(\mathbf{k}j)$ with other phonons (phonon-phonon interactions), or due to other scattering mechanisms. The symbol $\nabla_{\mathbf{r}}$ is the gradient in the real space.

The first term in Eq. 2.109 refers to the local variation of the number of phonons in time. The second term treats the variation of the number of phonons due to their movement through space. The term on the right hand side of the BTE refers to

the variation due to collisions between phonons. According to the relaxation time approximation [64], the phonon-phonon scattering can be modelled as,

$$\left.\frac{\partial n_{\mathbf{k}j}}{\partial t}\right|_{scat} = \frac{n^0_{\mathbf{k}j}(T) - n_{\mathbf{k}j}}{\tau_{\mathbf{k}j}(T)}. \tag{2.110}$$

where τ is the relaxation time of the mode and n^0 is the number of phonons of that mode at equilibrium, both of them function of local temperature T.

Chapter 3

Methodology

This chapter focuses on the methodology employed by Nano-κ for modelling phonon transport. Firstly, an introduction to the Monte Carlo method is presented, elucidating its origins and typical case applications in heat transfer. The subsequent section elaborates on how Nano-κ utilises the theoretical foundation discussed in Chapter 2 and the Monte Carlo techniques illustrated hereafter to calculate the thermal conductivity in nanostructures.

3.1 The Monte Carlo method

Usual approaches for the simulation of heat transfer at macroscale are Finite Volumes Method and Finite Elements Method. While it is possible to solve the BTE in nanoscale settings using these methods, usually they are computationally expensive enough such that they remain impractical: because the coefficients of the BTE are different for each phonon mode, the equation needs to be solved for each separate mode in the crystal. Moreover, they need to be solved at the same time, since the relaxation time τ is a function of temperature T, and T can only be estimated considering the whole phonon population. In a large crystal, the number of modes would be virtually infinite, allowing the FBZ to be considered continuous. This would, of course, be computationally impossible to solve. Alternatively, one can solve the BTE for a finite number of modes sampled from the FBZ, while still giving a result that is accurate enough for its purpose. Another method that can be employed on this problem by trading determinism for randomness is the Monte Carlo method (MC).

The MC method was developed by Metropolis and Ulam [54] in 1949 specially for nuclear physics problems, and named in reference to Monte Carlo, Monaco, famous for its casino. The idea behind MC is to have an statistical approximation of a phenomenon instead of a deterministic result. This is particularly useful for problems where analytical solutions may be too cumbersome to solve. To do this, the method uses some random aspect in the calculation, generating random samples of the studied phenomenon, with the general picture arising as a result of a large number of samples. In fact, the randomness of the MC method can actually be an advantage, since deterministic methods do not offer a notion of uncertainty in their results.

3.1.1 Monte Carlo use example: integration of a function

The Monte Carlo method is often used to integrate a D-dimensional curve $y = f(\mathbf{x}) = f(x_1, x_2, ..., x_n)$ in a domain Ω that is computationally difficult to solve when D is large. This is called Monte Carlo integration, as described by Sobol [70]. The statistical problem is then to discover the integral I:

$$I = \int_\Omega f(\mathbf{x})\ d\mathbf{x} \tag{3.1}$$

$$= \int_\Omega \frac{f(\mathbf{x})}{p(\mathbf{x})} p(\mathbf{x})\ d\mathbf{x} \tag{3.2}$$

$$= \int_\Omega \eta(\mathbf{x}) p(\mathbf{x})\ d\mathbf{x} \tag{3.3}$$

$$= \overline{\eta} = \frac{1}{N} \sum_{i=1}^{N} \eta(\mathbf{x}_i) \tag{3.4}$$

where $p(\mathbf{x})$ is a probability distribution from which $\mathbf{x}$ is drawn in each of the trials, N is the number of trials and $\eta(\mathbf{x}) = f(\mathbf{x})/p(\mathbf{x})$. The probability density function $p(\mathbf{x})$ should be positive in the integration interval and:

$$\int_\Omega p(\mathbf{x})\ d\mathbf{x} = 1 \tag{3.5}$$

To draw $\mathbf{x}$, first a random variable $\gamma = [\gamma_1, \gamma_2, ..., \gamma_D]$ is defined, uniformly distributed between 0 and 1, allowing to numerically draw pseudorandom numbers. This variable can then be related to $\mathbf{x}$ by solving the integral:

$$\int_0^{\mathbf{x}} p(\mathbf{x})\ d\mathbf{x} = \gamma \tag{3.6}$$

To actually solve the integral, the following steps are done:

1. Draw N values for γ_i;
2. Calculate $\mathbf{x}_i(\gamma_i)$ with the relation calculated by Eq. 3.6;
3. Calculate $\eta_i = f(\mathbf{x}_i)/p(\mathbf{x}_i)$ for each $\mathbf{x}_i$;
4. Calculate $\overline{\eta}$.

One frequent unidimensional toy example is the calculation of π by integrating the curve $f(x) = \sqrt{1 - x^2}$ between 0 and 1. Knowing that $I = \pi R^2/4 = \pi/4$, the value of π can be estimated. Using an uniform probability distribution $p(x) = 1$ for a simple study, $x = \gamma$ and $\eta = f(x)$. The calculation then becomes:

$$\pi \approx \frac{4}{N} \sum_{i=1}^{N} \sqrt{1 - x_i^2} \tag{3.7}$$

Figure 3.1 shows the results for 100 trials, with $N = 10^2$, 10^3 and 10^4. As N increases, the standard deviation of I decreases, as expected. Nevertheless, the mean value is already close from the true π. Precision is then a matter of choice by the analyst running the calculation.

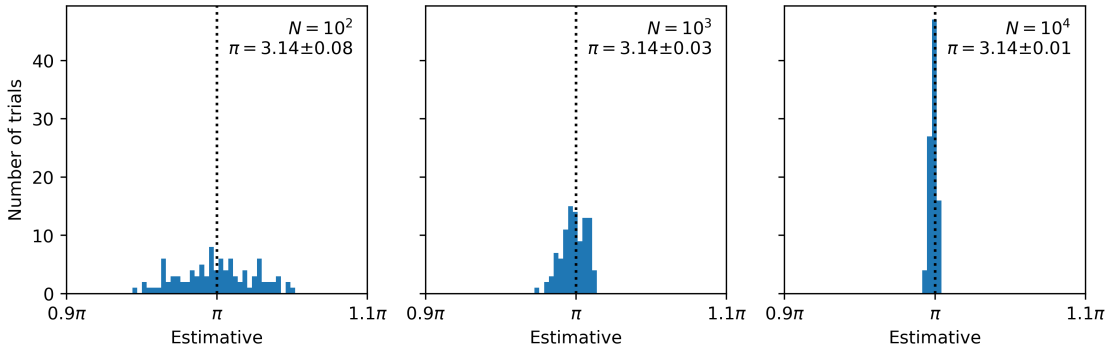


Figure 3.1: Monte Carlo integration for estimation of π, with 100 tries for each number of samples.

Another tactics used is the "hit and miss", as called by Sobol [70]. It consists of drawing the curve in the Cartesian plane, and sampling x and y uniformly in the desired interval (in this case [0,1] for both). The function to be integrated is then:

$$f(x, y) = \begin{cases} 1 \text{ if } x^2 + y^2 \leq 1 \\ 0 \text{ if } x^2 + y^2 > 1 \end{cases} \tag{3.8}$$

Just like before, the value of π is estimated by:

$$\pi \approx \frac{4}{N} \sum_{i=1}^{N} f(x_i, y_i) \tag{3.9}$$

Figure 3.2 shows the results with different values of N.

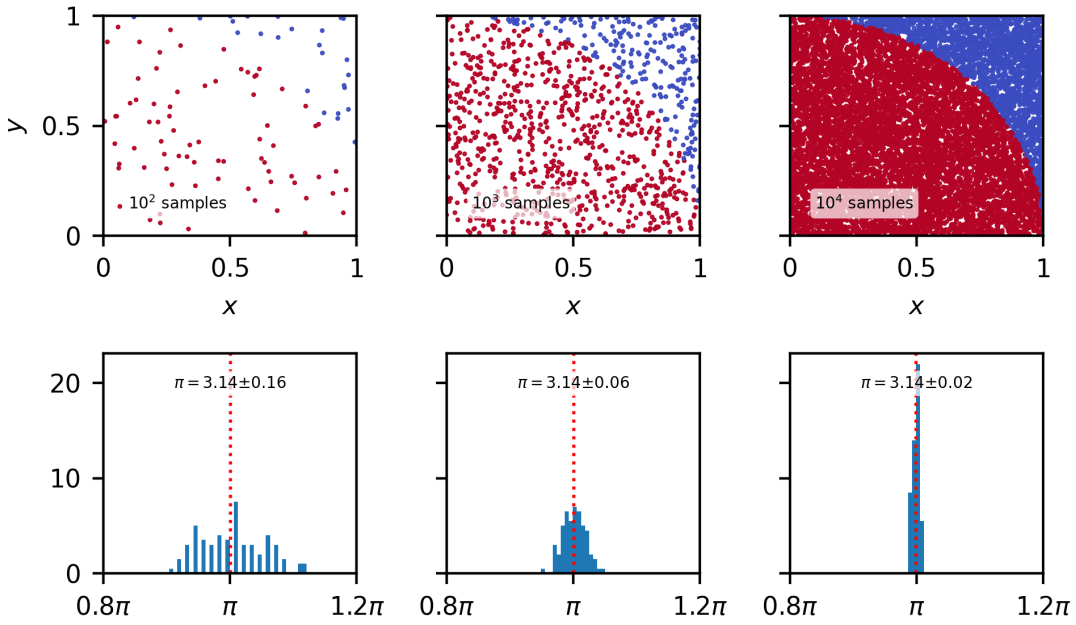


Figure 3.2: Monte Carlo integration for estimation of π by the "hit and miss" variant, with 100 tries for each N.

These are some of the simplest cases. More sophisticated methods can be used, as Sobol [70] shows, such as importance sampling (where a more appropriate $p(x)$ is used), or stratified sampling (integrating by regions to approximate the integral piece-wise and avoid oversampling regions with small significance). Other variation is the quasi-MC method, where random sampling is substituted by quasirandom sequences, such as Sobol sequence, to improve the uniformity of the sampling. All these techniques help to reduce the variance of the result, improving stability and reducing the number of samples necessary. It goes without saying that they can considerably improve the computational efficiency of the calculation when appropriately used.

After so many developments in the method during over 70 years, the name Monte Carlo does not refer anymore to a single algorithm, but to a whole family of statistical techniques. For most cases in engineering these stochastic approaches work well enough, including the field of heat transfer.

3.1.2 The MC method in heat transfer

The field of study of heat transfer has been a fertile ground for the application of the Monte Carlo method. Despite rarely being used to solve problems in conduction [71–73] and convection (where usually deterministic solutions are applied), radiative heat transfer has been using Monte Carlo for years. Rather than using a mathematically rigorous definition such as for the MC integration, the application of the MC method comes from a phenomenological approach, as discussed by Howell et al. [74]. Much like the original application of MC, where Metropolis and Ulam [54] used to model the random behaviour of neutrons during supercritical reactions [74], radiative heat transfer deals with particles, but with photons instead of neutrons. The particularities of the application are at first of less importance, since both rely on the idea of modelling a great number of their respective particles in order to observe the behaviour of the entire system.

In order of complexity, the MC method is used in radiative heat transfer to solve:

- **View factors**: the amount of radiative energy that leaves on geometry and arrives to another can be easy to calculate for simple geometries in particular cases. For complex settings, however, this can become difficult to do analytically. The MC method in this case samples energy bundles on each surface and shoots them in every direction. The view factor is calculated by counting how many of these bundles arrived on the other surface:

$$F_{1\to 2} = \frac{N_{1\to 2}}{N_1} \tag{3.10}$$

  This approach applies to black surfaces, where every bundle is absorbed and the emitted radiation obeys Planck's distribution for all wavelengths in all directions. The heat transfer can then be estimated by integrating the emissive powers considering Planck's distribution.

- **Heat transfer between surfaces**: when the emissivity varies with position, angle and/or wavelength, the probability distribution needs to be modified to account for these properties. The energy then cannot be calculated prior from Planck's distribution, but needs to be calculated from the simulation itself.

Usually this is done by associating an energy value to each emitted bundle, that will be subtracted from the surface. Then the bundles that arrive may be absorbed or not, and the balance between the energy that left and arrive modifies the temperature of the surface. Iteratively the system will reach a balance, and the final temperatures and heat fluxes will be achieved.

- **Participating media**: When the radiation travels through vacuum, it is not affected, and the bundle's energy arrives to the other side of the enclosure in its entirety. When the enclosure contains a gas or a particulate that can absorb or scatter photons, however, the decay in energy needs to be modelled. This can be done by considering the coefficients κ and σ, representing absorption and scattering respectively, in the radiative transfer equations. Sometimes both effects can be contracted into one single extinction coefficient β.

The original approach applied to these problems is to consider each particle as a bundle with fixed energy. The particle carries its initial energy for its whole path, and is extinct when it reaches S or meets a surface before, transferring its energy to the surface or to the medium. Their distance of travel S is limited by an exponential probability distribution, such that:

$$S = -\frac{1}{\beta}\ln(\gamma) \tag{3.11}$$

This is here to introduce the approach that Nano-κ uses. The extinction coefficient β here is an analogous to the relaxation time τ, but applied in space instead of time. Some works use the probabilistic approach to calculate S, setting a path length for the particle to be extinct at the end. Nano-κ uses the energy partitioning method explained here to reduce variance, which varies the energy of the particle along its path. When β is highly variable through space (for example due to temperature gradients or difference in compositions of the medium), it may be hard to sample S precisely. The null-collision method [75] can deal with it by adding a new type scattering event where the photon is "scattered forward", with no change to its frequency or trajectory. This forward scattering is considered together with the normal scattering (when photons change their direction) and absorption scattering (when their energy is absorbed by the medium) for the calculations of the probability of each event.

Instead of considering the path length a random variable, the variance of the answer can be reduced by using the energy partitioning method. In the previous case, the particle would travel through the domain carrying its initial energy e_0 until it is absorbed. Its energy would then be converted in a local variation of temperature in the medium, which in turn modifies its properties. The energy partitioning method changes this idea by replacing the complete absorption of the photon by the attenuation of its energy e along the path. The energy of the photon is modified according to:

$$e = e_0 \exp[-\beta(\mathbf{r}, t)(\mathbf{r}(t) - \mathbf{r}_0)] \tag{3.12}$$

where the extinction coefficient β is then a function of the medium, varying in space and time.

Another way to reduce variance is to divide the radiative intensity into two parts: a fixed part I_{ref}, arbitrarily chosen to be used as a reference; and a deviational part δI, which changes with the calculation. The resultant intensity is then:

$$I = I_{ref} + \delta I \tag{3.13}$$

There are several other techniques that allow reducing sample size without compromising the calculation [74]. The improvements on computational resources also permit to run much larger and more complex simulations that would be almost infeasible a decade before. All these allow the Monte Carlo method to be a reliable and direct way to solve problems that can be cumbersome with other methods.

3.2 Nano-κ

Just like with radiation transfer, the Monte Carlo method can also be used to solve problems related to phonon transport. Compared to what was discussed about its application on photons, the Monte Carlo modelling of phonon transport has particularities that are similar to some of the most complicated cases with photons, comprising several directional and spectral considerations. Indeed, the radiative transfer equation (RTE) has similarities with the Boltzmann Transport Equation (BTE), allowing the application of several of the techniques used to solve the RTE [76].

In this section the physical and mathematical basis of Nano-κ will be explained, with the algorithmic part of the code itself being exposed in the next chapter. To use Nano-κ, two main pieces of information are required as input: (1) a geometry that describes the surface of the analysed nanodevice and the boundary conditions imposed on it; and (2) the material data describing the available phonon modes. Using these inputs, the Monte Carlo method can be applied to solve the BTE. This section aims to answer the following questions:

- How is the geometry to be simulated mathematically defined, and how to calculate all of its relevant properties?
- What does the concept particle in Nano-κ represents?
- How does Nano-κ ensure the mathematical stability of the calculation?
- What are the steps in solving the BTE?
- How to estimate the thermal conductivity from the calculated data?

3.2.1 Geometry

Mesh definition

Nano-κ works with geometries constructed as triangular meshes, defined by vertices (points with coordinates in 3D space) and triangular faces (composed by any three vertices) that describe the surface of the volume to be simulated. The declaration of vertices can be done as a $N_v \times 3$ matrix such that:

$$\begin{array}{ccc} \text{Vertex} & \text{Vertex index} & \text{Vertex } x,\ y,\ z \\ \mathbf{p}_1 & 1 & \begin{bmatrix} x_1 & y_1 & z_1 \end{bmatrix} \\ \mathbf{p}_2 & 2 & \begin{bmatrix} x_2 & y_2 & z_2 \end{bmatrix} \\ \mathbf{p}_3 & 3 & \begin{bmatrix} x_3 & y_3 & z_3 \end{bmatrix} \\ \vdots & \vdots & \begin{bmatrix} \vdots & \vdots & \vdots \end{bmatrix} \\ \mathbf{p}_v & v & \begin{bmatrix} x_v & y_v & z_v \end{bmatrix} \\ \vdots & \vdots & \begin{bmatrix} \vdots & \vdots & \vdots \end{bmatrix} \\ \mathbf{p}_{N_v} & N_v & \begin{bmatrix} x_{N_v} & y_{N_v} & z_{N_v} \end{bmatrix} \end{array}$$

where $\mathbf{p}_v$[1] is the v^{th} vertex in the matrix, and N_v is the total number of vertices. Similarly, the faces are declared by a $N_f \times 3$ matrix:

$$\begin{array}{ccc} \text{Face} & \text{Face index} & \text{Vertex indices} \\ f_1 & 1 & \begin{bmatrix} p_{11} & p_{12} & p_{13} \end{bmatrix} \\ f_2 & 2 & \begin{bmatrix} p_{21} & p_{22} & p_{23} \end{bmatrix} \\ f_3 & 3 & \begin{bmatrix} p_{31} & p_{32} & p_{33} \end{bmatrix} \\ \vdots & \vdots & \begin{bmatrix} \vdots & \vdots & \vdots \end{bmatrix} \\ f_f & f & \begin{bmatrix} p_{f1} & p_{f2} & p_{f3} \end{bmatrix} \\ \vdots & \vdots & \begin{bmatrix} \vdots & \vdots & \vdots \end{bmatrix} \\ f_{N_f} & N_f & \begin{bmatrix} p_{N_f1} & p_{N_f2} & p_{N_f3} \end{bmatrix} \end{array}$$

where each line represents a triangular face, referencing its three vertices, and N_f is the total number of triangles in the mesh. The reference to the vertices is done by their indices. A face f_f of index f can be declared for example as $[p_{f1}, p_{f2}, p_{f3}] = [3, 5, 2]$, being therefore defined by the vertices $\mathbf{p}_{f1} = \mathbf{p}_3$, $\mathbf{p}_{f2} = \mathbf{p}_5$ and $\mathbf{p}_{f3} = \mathbf{p}_2$. From these two information, all else can be calculated and checked to ensure the stability of the simulation.

Watertightness and face normals

To be sure that there will not be phonons leaking out of the domain, the mesh needs to be "watertight". When it is, every edge between two vertices is connected to two triangles or more[2]. If there is one that is connected to only one triangle, it means that there is a hole in the mesh and the simulation cannot take place in that geometry.

The normals of each face (i.e. each triangle) can be calculated as:

$$\mathbf{n}_f = \frac{(\mathbf{p}_{f2} - \mathbf{p}_{f1}) \times (\mathbf{p}_{f3} - \mathbf{p}_{f1})}{\|(\mathbf{p}_{f2} - \mathbf{p}_{f1}) \times (\mathbf{p}_{f3} - \mathbf{p}_{f1})\|} \tag{3.14}$$

and its area as:

$$A_f = \frac{\|(\mathbf{p}_{f2} - \mathbf{p}_{f1}) \times (\mathbf{p}_{f3} - \mathbf{p}_{f1})\|}{2} \tag{3.15}$$

[1]The symbol $\mathbf{p}$ is used for the vertices instead of $\mathbf{v}$ to avoid confusion with phonon group velocity.

[2]Edges that are part of three or more triangles can be connected to internal interfaces dividing the geometry into regions. Nano-κ does not support interfaces yet, but already has a draft on how to detect them.

Their centroid can also be calculated as:

$$\mathbf{c}_f = \frac{\mathbf{p}_{f1} + \mathbf{p}_{f2} + \mathbf{p}_{f3}}{3} \tag{3.16}$$

The normals are defined outwards by convention. It is needed therefore to check the proper order of vertices to be taken as $\mathbf{p}_{f1}$, $\mathbf{p}_{f2}$ and $\mathbf{p}_{f3}$. It can be done by cycling through the vertices of each face in counterclockwise order as viewed from the outside, starting from any vertex. When an edge divides two faces, each triangle will go along the edge in a different direction. Figure 3.3 illustrates the analysis. This helps to check if all the normals are consistent in the geometry.

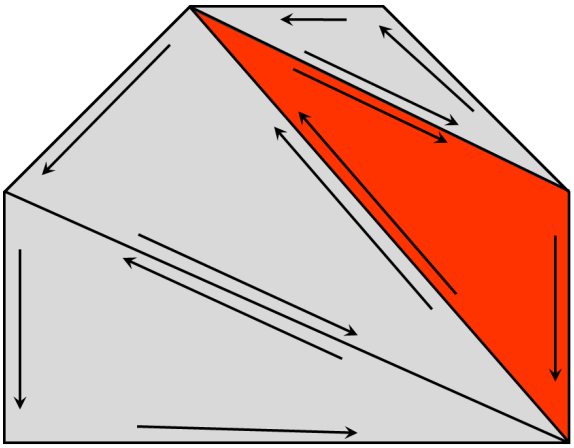


Figure 3.3: Check of consistent normals in part of a triangular mesh. The red triangle has a vertex order that leads to inconsistent normals.

Facets and ray tracing

Ray tracing, in the context of Nano-κ, refers to the identification of the intersection between the path of a particle and the surface of the geometry. It is at this intersection point that the phonon is reflected or absorbed. The ray tracing is done by extending the future trajectory of a particle in position $\mathbf{r}$ with velocity $\mathbf{v}$. Given a plane with normal $\mathbf{n}$, the point of collision $\mathbf{r}_c$ between particle and plane can be calculated as:

$$\mathbf{r}_c = \mathbf{r} + \frac{(\mathbf{r}_o - \mathbf{r}) \cdot \mathbf{n}}{(\mathbf{v} \cdot \mathbf{n})} \mathbf{v} \tag{3.17}$$

where $\mathbf{r}_o$ is an arbitrary point on the plane chosen as its origin. The number of timesteps until collision can also be calculated as:

$$\begin{aligned} N_c &= \frac{\mathbf{r}_c - \mathbf{r}}{\mathbf{v}\Delta t} \\ &= \frac{(\mathbf{r}_o - \mathbf{r}) \cdot \mathbf{n}}{(\mathbf{v} \cdot \mathbf{n})\Delta t} \end{aligned} \tag{3.18}$$

where Δt is the size of the timestep.

The most straightforward way to execute this detection is to calculate $\mathbf{r}_c$ for every triangle of the mesh. However, it is not rare for several triangles of the mesh to be located on the same plane. When several adjacent faces are coplanar, they form a "facet". For example, the mesh of a cube has 6 facets, composed of 2 triangular

faces each. The normal of all faces that compose the facet is the same, and the total area of the facet is the sum of the triangles that form it. The definition of facet can accelerate the calculation of ray tracing used for collision detection by reducing the number of line-plane intersection checks.

The reduction of checks from faces to facets happen in this first check of N_c, reducing the number of planes to calculate. When $N_c < 0$ a collision between the particle and the facet in question is impossible, since it will be travelling in the other direction. When $N_c \geq 0$, a collision of the particle on that plane is possible, but not necessarily will happen. This is because a triangle of the geometry may not be positioned where $\mathbf{r}_c$ is. Figure 3.4 shows an scheme.

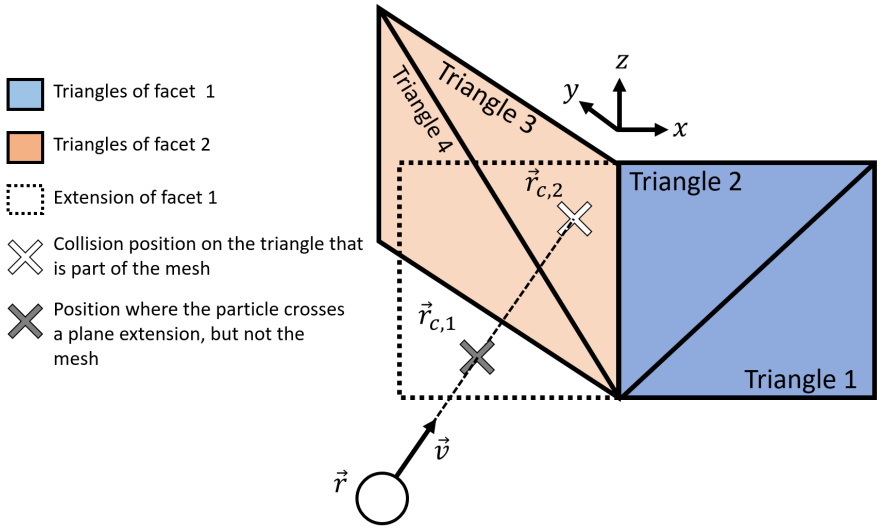


Figure 3.4: Example of ray tracing procedure to find the true collision point $\mathbf{r}_c$ between particle and mesh.

To be sure which plane is the true colliding one, the possible $\mathbf{r}_c$ are checked in increasing order for every plane with $N_c \geq 0$. Then the barycentric coordinates are used, having to be calculated for every triangle in the facet. Depending on the case, checking first if $\mathbf{r}_c$ is contained in the bounding box of the triangle can exclude some unnecessary calculations.

In barycentric coordinates, any point on the plane is calculated as a linear combination of the vertices of a triangle, $\mathbf{r}_c = a_1\mathbf{p}_1 + a_2\mathbf{p}_2 + a_3\mathbf{p}_3$. The points that are located inside the triangle will have all of its barycentric coordinates a_1, a_2 and a_3 between 0 and 1. The basis vectors can be constructed as:

$$\begin{aligned} \mathbf{b}_1 &= \mathbf{p}_2 - \mathbf{p}_1 \\ \mathbf{b}_2 &= \mathbf{p}_3 - \mathbf{p}_1 \\ \mathbf{b}_3 &= \mathbf{b}_1 \times \mathbf{b}_2 \end{aligned} \tag{3.19}$$

The transformation that gives these new coordinates is:

$$\mathbf{B}^{-1}\mathbf{r}_c = \begin{bmatrix} b_{1x} & b_{2x} & b_{3x} \\ b_{1y} & b_{2y} & b_{3y} \\ b_{1z} & b_{2z} & b_{3z} \end{bmatrix}^{-1} \begin{bmatrix} r_{cx} \\ r_{cy} \\ r_{cz} \end{bmatrix} = \begin{bmatrix} a_2 \\ a_3 \\ 0 \end{bmatrix} \tag{3.20}$$

The third component in the vector on the right side is known to be zero since $\mathbf{b}_3$ points parallel to $\mathbf{n}$ and $\mathbf{r}_c$ is on the plane. The a_1 coordinate calculated as

$a_1 = 1 - a_2 - a_3$ because:

$$\begin{aligned}\mathbf{r}_c &= \mathbf{p}_1 + a_2\mathbf{b}_1 + a_3\mathbf{b}_2 \\ &= \mathbf{p}_1 + a_2(\mathbf{p}_2 - \mathbf{p}_1) + a_3(\mathbf{p}_3 - \mathbf{p}_1) \\ &= (1 - a_2 - a_3)\mathbf{p}_1 + a_2\mathbf{p}_2 + a_3\mathbf{p}_3\end{aligned} \tag{3.21}$$

If any of them is out of the interval [0, 1], the point is outside of the triangle and the collision does not happen.

Barycentric coordinates can also be used to sample on the surface of the geometry. To generate a point on the surface of the mesh, first a triangle needs to be selected. Since an uniform sample is desired, the number of points on each face need to be proportional to its area. Therefore a face f has to be chosen with probability:

$$P_f = \frac{A_f}{A} = \frac{A_f}{\sum_{f=1}^{N_f} A_f} \tag{3.22}$$

where A is the total surface area of the mesh. Then the barycentric coordinates can be drawn using two uniformly random numbers γ [77]:

$$a_1 = 1 - \sqrt{\gamma_1} \tag{3.23a}$$
$$a_2 = (1 - \gamma_2)\sqrt{\gamma_1} \tag{3.23b}$$
$$a_3 = \gamma_2 * \sqrt{\gamma_1} \tag{3.23c}$$

and the point $\mathbf{r}$ on the triangle can be defined as:

$$\mathbf{r} = a_1\mathbf{p}_{f1} + a_2\mathbf{p}_{f2} + a_3\mathbf{p}_{f3} \tag{3.24}$$

This surface sampling can also be done on a single facet, by considering only the triangles that compose it. This is used in Nano-κ when simulating phonons coming from the external thermal reservoirs.

Volume triangulation

In order to estimate the volume of the geometry, the mesh can be triangulated in 3D, dividing the volume into tetrahedra. If needed, extra points can be created to ensure a proper triangulation. The tetrahedra are then used to sample the particles in the domain. When a particle is sampled, it has a probability of being generated in a tetrahedon t composed of points $\mathbf{p}_{t1}$, $\mathbf{p}_{t2}$, $\mathbf{p}_{t3}$ and $\mathbf{p}_{t4}$ equal to:

$$P_t = \frac{V_t}{V} = \frac{A_{bt}|\mathbf{n}_{bt} \cdot (\mathbf{p}_{t4} - \mathbf{p}_{t1})|}{\sum_{t=1}^{N_t} V_t} \tag{3.25}$$

$$A_{bt} = \frac{\|(\mathbf{p}_{t2} - \mathbf{p}_{t1}) \times (\mathbf{p}_{t3} - \mathbf{p}_{t1})\|}{2} \tag{3.26}$$

$$\mathbf{n}_{bt} = \frac{(\mathbf{p}_{t2} - \mathbf{p}_{t1}) \times (\mathbf{p}_{t3} - \mathbf{p}_{t1})}{\|(\mathbf{p}_{t2} - \mathbf{p}_{t1}) \times (\mathbf{p}_{t3} - \mathbf{p}_{t1})\|} \tag{3.27}$$

where V_t is the volume of the tetrahedron and V is the total volume of the geometry, A_{bt} is the surface area of the base of the tetrahedron (the triangle with three of its four vertices) and $\mathbf{n}_{bt}$ is the normal of that base. After picking the tetrahedron

where to sample, the actual point $\mathbf{r}$ is generated by randomly sampling based on barycentric coordinates $[a_1, a_2, a_3, a_4]$ [78]:

$$\mathbf{r} = a_1\mathbf{p}_{t1} + a_2\mathbf{p}_{t2} + a_3\mathbf{p}_{t3} + a_4\mathbf{p}_{t4} \tag{3.28}$$

$$a_i = \frac{-\ln(\gamma_i)}{\sum_{j=1}^{4} -\ln(\gamma_j)} \tag{3.29}$$

where γ_i is a random number between 0 and 1. This ensures an uniform sampling on the domain without rejection. The tetrahedrons also allow to rapidly check whether the particle is inside or outside the geometry by doing the inverse process (converting $\mathbf{r}$ to barycentric coordinates with a change of basis much like done with surface sampling.

3.2.2 Material Data

As commented in Section 2.2, the material data necessary for the Monte Carlo calculation can be obtained from *ab-initio* simulations. These are not performed by Nano-κ but rather by Phono3py [36] and used as input to Nano-κ.

The material data contains the necessary information about each vibrational mode. This information corresponds originally only to the modes found in the irreducible Brillouin zone (IBZ). These are then expanded using reflection and rotation operations that define the symmetries of the crystal, finally reconstructing the full Brillouin zone (FBZ). The data in the FBZ can then be used in the Monte Carlo simulation. It comprises, for each mode:

- The reduced coordinates $\mathbf{q}$ of the wavevectors (q-points);
- The ordinal frequencies f;
- The group velocities $\mathbf{v}$;
- The imaginary part of the self energy $\Gamma(T)$;

Each mode has two indices assigned to them, q for the q-point index and j for the branch index. The reduced coordinates $\mathbf{q}$ can be used together with the reciprocal lattice vectors, organised as columns in a matrix $\mathbf{B}$, to reconstruct the wavevectors for each mode, $\mathbf{k} = \mathbf{B} \cdot \mathbf{q}$. Consequently, every other property in the data is then associated to a wavevector $\mathbf{k}$ and a branch j.

The angular frequencies can be calculated as $\omega_{\mathbf{k}j} = 2\pi f_{\mathbf{k}j}$. The group velocities $\mathbf{v}_{\mathbf{k}j}$ do not need any treatment. The last relevant property to be calculated is the relaxation time $\tau_{\mathbf{k}j}(T)$, which controls the attenuation on the number of phonons due to collisions between modes (Eq. 2.110). These are calculated from $\Gamma_{\mathbf{k}j}(T)$ as:

$$\tau_{\mathbf{k}j}(T) = \frac{1}{4\pi\Gamma_{\mathbf{k}j}(T)} \tag{3.30}$$

From these, other relevant quantities can be calculated during the simulation. To cite a couple of examples, the number of phonons at equilbrium $n^0_{\mathbf{k}j}(T)$ at temperature T (Eq. 2.87) uses $\omega_{\mathbf{k}j}$, and the specularity parameter p (Eq. 2.106) uses $\mathbf{k}$ and $\mathbf{v}_{\mathbf{k}j}$ together with geometric parameters of the facet on which the phonon is being reflected.

The precision of the Monte Carlo calculation is associated with the number of modes in the data. The more modes there are available, more precise the description of the FBZ will be, and consequently the statistical estimation of the BTE solution. The discretisation of the FBZ is set on Phono3py when calculating the data. A too fine discretisation of the FBZ, however, may not be effective. This is because the random fluctuations in the quantities calculated by the Monte Carlo simulation (temperature, heat flux, etc.) are dependent on the number of particles being used, which in turn affects computation time. An optimum point may exist considering all these factors, being left for the user to decide on a case by case basis.

3.2.3 Particles

With the geometry and the material data set, the volume can now be populated with phonons. In order to understand the method used in Nano-κ, it is important to define "particle" first, as it is used in the code. Each particle is a punctual entity (with no radius or volume) that represents a wavepacket of a different mode $\mathbf{k}j$. Each particle has therefore properties linked to that mode, which may change or not over time:

- Immutable properties are those that do not change unless a boundary scattering event forces the particle to change its mode. These are the mode indices q and j, wavevector $\mathbf{k}$, frequency ω and velocity $\mathbf{v}$. The *only* event type that changes particles' modes is their reflection on a rough boundary, since the particles cannot keep the same $\mathbf{v}$, and thus the same $\mathbf{k}j$.

- Mutable properties change every timestep according to their immutable properties and their surrounding environment. For example, their position $\mathbf{r}$ is updated according to $\mathbf{v}$, the temperature T that they are submitted to changes with $\mathbf{r}$, their relaxation time τ is function of T, their occupation number n is affected by ω, T and τ, etc.

During the simulation, they will travel through space being generated, extinct or reflected on the geometry's surfaces. In a given region of space, there may be several particles with the same $\mathbf{k}j$, but coming from different origins (e.g. one may be coming from a reservoir with imposed T, while the other may be the result of a reflection on a facet). Since Nano-κ uses the energy partition method commented in Sec. 3.1.2, and because the paths of these particles are different, each of them may represent different numbers of phonons n. The expected number of the phonons of mode $\mathbf{k}j$ in each region of space is obtained by averaging the occupation number n of the particles that represent them. In order to do that, the number of particles that represent each mode is kept approximately constant throughout the simulation and uniformly distributed on the domain.

3.2.4 Reservoirs and injection of particles

Setting a fixed temperature can be modelled by an external thermal reservoir in contact with the desired boundary. To ensure that the reservoir is at a fixed temperature, it can be considered very large. Phonons that leave the geometry through the boundary with the reservoir have their energy completely absorbed. Since it is

very large, the absorbed phonon does not have any effect on the temperature of the reservoir.

Since thermodynamic temperature T is defined at equilibrium, and that the size of the reservoir ensures that it is overall at equilibrium, the phonons that populate the reservoir are distributed according to the Bose-Einstein distribution, Eq. 2.87 . Therefore, the number of phonons $N_{\mathbf{k}j}$ of a given mode $\mathbf{k}j$ that enter the domain in an interval Δt is proportional to $n^0_{\mathbf{k}j}$ and are independent of the phonons inside the domain [51]:

$$N_{\mathbf{k}j} = n^0_{\mathbf{k}j} A(-\mathbf{v}_{\mathbf{k}j} \cdot \mathbf{n})\Delta t \tag{3.31}$$

where A is the contact surface area with the reservoir, and $\mathbf{n}$ is the normal of the surface, pointing outwards from the geometry. We can say therefore that the surface in contact with the reservoirs emits and absorbs phonons as a black body.

The number of particles is kept constant in the geometry by calculating the rate at which the particles of each mode cross the reservoir's facet assuming they are uniformly distributed in the geometry.

The density of particles per volume in the domain is calculated by:

$$\rho = \frac{N_p}{V} \tag{3.32}$$

where N_p is the number of particles and V is the volume of the geometry. We can define the flux of particles of mode $\mathbf{k}j$ in any point of the geometry as $\mathbf{j}_{\mathbf{k}j} = (\rho/3N_aN)\mathbf{v}_{\mathbf{k}j}$, where N is the number of wavevectors in the FBZ and N_a is the number of atoms in an unit cell.

In order to keep the number of particles of mode $\mathbf{k}j$ constant, $\nabla \cdot \mathbf{j}_{\mathbf{k}j} = 0$ on the volume. Indeed, since the velocity of each mode is constant in absolute value and direction, $\nabla \cdot \mathbf{j}_{\mathbf{k}j} = 0$ everywhere and independently of the particle density ρ, which should also be constant. If we apply the divergence theorem on the geometry, we have:

$$\iiint_V \frac{\rho}{3N_aN}(\nabla \cdot \mathbf{v}_{\mathbf{k}j})\, dV = \iint_S \frac{\rho}{3N_aN}(\mathbf{v}_{\mathbf{k}j} \cdot \mathbf{n})\, dS \tag{3.33}$$

$$0 = \sum_F \mathbf{v}_{\mathbf{k}j} \cdot \mathbf{n}_F \tag{3.34}$$

where the sum over F refers to all the facets of the geometry. When $\mathbf{v}_{\mathbf{k}j} \cdot \mathbf{n}_F > 0$, the particles are reaching a boundary and are destroyed; when $\mathbf{v}_{\mathbf{k}j} \cdot \mathbf{n}_F < 0$, they are created from reflections of other particles or are injected from reservoirs. In a closed geometry, this will always be zero. In other words, every particle created on a boundary of the geometry will be destroyed on other boundary at rates proportional to $\mathbf{v} \cdot \mathbf{n}$. The number of particles created or destroyed on a facet of area A in a time interval Δt is calculated by integrating the flux over the surface and over time:

$$R_{\mathbf{k}j} = \int_A \int_0^{\Delta t} \frac{\rho}{3N_aN}\mathbf{v}_{\mathbf{k}j}\, dt\, dA \tag{3.35}$$

$$= \frac{\rho A \Delta t}{3N_aN}\mathbf{v}_{\mathbf{k}j} \cdot \mathbf{n} \tag{3.36}$$

If $R_{\mathbf{k}j} > 0$, $\mathbf{v}_{\mathbf{k}j} \cdot \mathbf{n} > 0$, and the particles are hitting the facet from inside the domain. If $R_{\mathbf{k}j} < 0$, the particles are coming from outside and entering the domain.

The rate of particles of mode $\mathbf{k}j$ being injected from a reservoir with normal $\mathbf{n}$ in a timestep Δt is therefore $\max(-R_{\mathbf{k}j}, 0)$. The particles can sampled by using Equations 3.22,3.23 and 3.24, considering only the triangles of the reservoir's facet on Eq. 3.22.

3.2.5 Drift and boundary scattering

After being injected, the position $\mathbf{r}$ of a particle is modified every timestep by its velocity:

$$\mathbf{r}^k = \mathbf{r}^{k-1} + \mathbf{v}_{\mathbf{k}j}\Delta t \tag{3.37}$$

where the superscripts $k-1$ and k refer to the previous and the current timestep.

Eventually, the particle will find a boundary of the geometry. The position of the collision with the boundary $\mathbf{r}_c$ can be calculated by the ray tracing procedure discussed previously (Eqs. 3.17, 3.18, 3.19, 3.20 and 3.21). When the particle finds a boundary, it can hit either a reservoir or a rough surface. In the first case it is completely absorbed. In the second case, it will undergo a reflection.

For the reflection of wavepackets on rough walls, there are several possible models. Firstly it is important to note that, in our modelling formalism, any wall is an interface between two solid media. They can be made of different materials, or even made of the same material but with different orientations. For interfaces between two crystals, there are two prominent models of transmissivity of materials in the literature [39]: the acoustic mismatch model (AMM) assumes a perfectly smooth surface, such that the reflections and transmissions are specular and elastic following Snell-Descartes formalism; the diffuse mismatch model (DMM) on the other hand, considers a highly rough interface, and phonon transmission is treated such that the phonon "forgets" from which side came from, and is re-emitted with another random mode in either side of the interface. There are several variations of these models, but with the same basic characteristics.

The modelling of the reflection only (instead of transmission) is just a matter of considering the interface as adiabatic, and the whole energy is contained inside the domain. From the two perspectives given by the AMM and DMM models, two conclusions can be done: for highly smooth surfaces, phonons behave in principle as plane waves, reflecting specularly according to Snell's law; while for highly rough surfaces, phonons are "reset", having their energy absorbed and re-emitted much like with a blackbody.

The approach taken in this thesis is to use the specularity parameter defined in Eq. 2.106 to estimate which part of the energy is reflected specularly and which part becomes diffuse. If a phonon mode that reflects on a given surface has a calculated $p = 0.8$, it means that 20% of its energy is diffused to all modes, while 80% is reflected with exactly the same energy and wavelength it originally had. Figure 3.5 shows a scheme of the model.

The particle however can only undergo one type of reflection at a time in Nano-κ. To know which one, the specularity p is used as a probability of a specular reflection, which can only happen if Eq. 2.108 is satisfied. If the conditions in Eq. 2.108 are not met, the specularity for that mode, on that facet, is forced to be 0. This means that since that mode does not have a perfectly specular correspondence, it cannot be specularly reflected and every reflection it suffers on that facet is diffuse. When the reflection is specular, the mode of the particle $\mathbf{k}j$ is changed to the new mode $\mathbf{k}'j'$

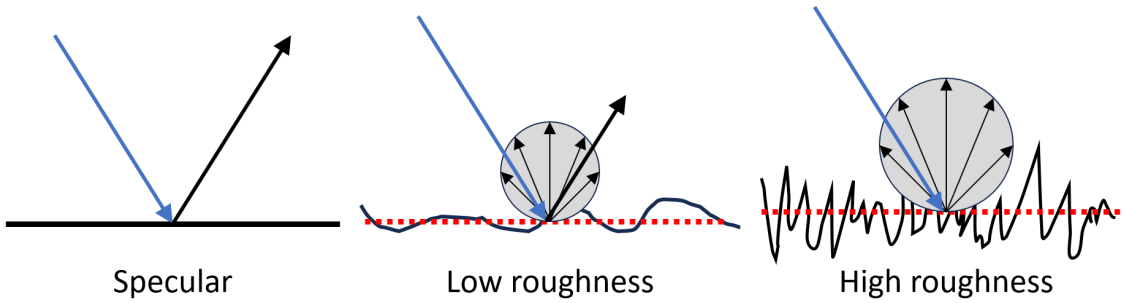


Figure 3.5: Representation of a reflection for a completely smooth surface, a surface with low roughness and a highly rough one.

that satisfies the conditions, and it continues its trajectory with the same energy as before and with a new reflected velocity. We can say that modes $\mathbf{k}j$ and $\mathbf{k}'j'$ are "specularly related" for this specific facet.

When it is diffuse, however, the particle is randomly assigned a new mode according to a probability $P_{\mathbf{k}'j'}$. For modes with $\mathbf{v}_{\mathbf{k}'j'} \cdot \mathbf{n} \geq 0$, $P_{\mathbf{k}'j'}$ is 0 since they cannot leave the surface. When $\mathbf{v}_{\mathbf{k}'j'} \cdot \mathbf{n} < 0$, $P_{\mathbf{k}'j'} > 0$. In order to keep the number of particles per mode constant, the probability $P_{\mathbf{k}'j'}$ has to complement the number of times that a particle $\mathbf{k}'j'$ is generated by specular reflections, so that $R_{\mathbf{k}'j'}$ is kept as calculated in Sec. 3.2.4. The reasoning is as follows.

Let's consider a particle of mode $\mathbf{k}j$ that is specularly related to the mode $\mathbf{k}'j'$. Every specular reflection of $\mathbf{k}j$ will *necessarily* generate a particle $\mathbf{k}'j'$. The total rate of creation of mode $\mathbf{k}'j'$ on that facet can be divided in specular and diffuse parts:

$$R_{\mathbf{k}'j'} = R^{spec}_{\mathbf{k}'j'} + R^{diff}_{\mathbf{k}'j'} \tag{3.38}$$

$$= -R_{\mathbf{k}j} p_{\mathbf{k}j} + R^{diff}_{\mathbf{k}'j'} \tag{3.39}$$

where $p_{\mathbf{k}j}$ is the specularity of mode $\mathbf{k}j$ on that facet. It follows that:

$$R^{diff}_{\mathbf{k}'j'} = R_{\mathbf{k}'j'} + R_{\mathbf{k}j} p_{\mathbf{k}j} \tag{3.40}$$

$$= \frac{\rho A \Delta t}{3 N_a N} \big(\mathbf{v}_{\mathbf{k}'j'} \cdot \mathbf{n}\big) + \frac{\rho A \Delta t}{3 N_a N} \big(\mathbf{v}_{\mathbf{k}j} \cdot \mathbf{n}\big) p_{\mathbf{k}j} \tag{3.41}$$

$$= \frac{\rho A \Delta t}{3 N_a N} \big[\big(\mathbf{v}_{\mathbf{k}'j'} \cdot \mathbf{n}\big) + \big(\mathbf{v}_{\mathbf{k}j} \cdot \mathbf{n}\big) p_{\mathbf{k}j}\big] \tag{3.42}$$

From Eq. 2.108 we have that $\mathbf{v}_{\mathbf{k}j} \cdot \mathbf{n} = -\mathbf{v}_{\mathbf{k}'j'} \cdot \mathbf{n}$. Thus:

$$R^{diff}_{\mathbf{k}'j'} = \frac{\rho A \Delta t}{3 N_a N} \big[\big(\mathbf{v}_{\mathbf{k}'j'} \cdot \mathbf{n}\big) - \big(\mathbf{v}_{\mathbf{k}'j'} \cdot \mathbf{n}\big) p_{\mathbf{k}j}\big] \tag{3.43}$$

$$= \frac{\rho A \Delta t}{3 N_a N} (\mathbf{v}_{\mathbf{k}'j'} \cdot \mathbf{n})(1 - p_{\mathbf{k}j}) \tag{3.44}$$

Having $R^{diff}_{\mathbf{k}'j'}$ for every possible mode, we can calculate a probability to pick each of them that is proportional to their rate:

$$P_{\mathbf{k}'j'} = \frac{R^{diff}_{\mathbf{k}'j'}}{\sum_{\mathbf{k}'j'} R^{diff}_{\mathbf{k}'j'}} \tag{3.45}$$

$$= \frac{(1 - p_{\mathbf{k}j}) \mathbf{v}_{\mathbf{k}'j'} \cdot \mathbf{n}}{\sum_{\mathbf{k}'j'} (1 - p_{\mathbf{k}j}) \mathbf{v}_{\mathbf{k}'j'} \cdot \mathbf{n}} \tag{3.46}$$

where mode $\mathbf{k}j$ is the one that is specularly correspondent to mode $\mathbf{k}'j'$ on that facet. For modes $\mathbf{k}'j'$ without specularly related modes $\mathbf{k}j$, $p_{\mathbf{k}j} = 0$. Only modes with $\mathbf{v}_{\mathbf{k}j} \cdot \mathbf{n} < 0$ are considered. Its occupation is then assigned n^0 for its new mode frequency at local T.

3.2.6 Energy and local temperature

As mentioned, Nano-κ uses the energy partition variation of Monte Carlo. Every particle that comes from a reservoir carries into the geometry a number of phonons equal to $n^0_{\mathbf{k}j}$ at the imposed temperature (Eq. 2.86). As the particle travels through the domain its number of phonons n is modified by its environment, either by scattering or by boundary reflections.

In order to estimate the local energy in each part of the domain, the geometry is subdivided into subvolumes. Only the particles inside the subvolume are considered to calculate its temperature. These subvolumes are defined with the help of reference points $\mathbf{r}_s$, which are points in the 3D space generated according to user parameters. A particle is considered to be in the subvolumes s when it is positioned closer to $\mathbf{r}_s$ than to any other reference point.

Temperature is a quantity only defined at equilibrium i.e. when phonons are distributed according Eq. 2.86, where the energy density e is calculated as function of temperature. Nevertheless, in systems out of equilibrium an equivalent temperature (sometimes called pseudo-temperature) T^* can be estimated by inverting this function, so that $T^* = f(e)$.

The energy density e of a subvolume is calculated by:

$$e = \frac{1}{V_0 N} \frac{3N_a N}{N_{ps}} \sum_{i=1}^{N_{ps}} \hbar\omega_i \left[n_i + \frac{1}{2} \right] \tag{3.47}$$

where N_{ps} refers to the number of particles in the subvolume. The coefficient $3N_aN/N_{ps}$ is added to the equation to scale the result according to the number of particles. If N_{ps} is, for example, twice the number of modes in the data (or $3N_aN/N_{ps} = 2$), the summation over N_{ps} results in twice the expected e, and needs to be corrected accordingly.

In order to reduce variance, a reference temperature T_{ref} is used in the calculation of energy density. This technique is derived from the application of reference and deviational parts to the calculation, as commented in Section 3.1.2. Equation 3.47 is then modified as:

$$e = e_{ref} + \delta e \tag{3.48}$$

$$= \frac{1}{V_0 N} \sum_{\mathbf{k}j} \hbar\omega_{\mathbf{k}j} \left[n^0_{\mathbf{k}j}(T_{ref}) + \frac{1}{2} \right] + \frac{1}{V_0 N} \frac{3N_a N}{N_{ps}} \sum_{i=1}^{N_{ps}} \hbar\omega_i \delta n_i \tag{3.49}$$

where $\delta n_i = n_i - n^0(\omega_i, T_{ref})$ for particle i of mode $\mathbf{k}j$. The reference T_{ref} can be any temperature that the user desires, and does not have to be the same for every subvolume neither kept constant in time. Since the values of n carried by each particle are absolute, T_{ref} and e_{ref} can be updated as necessary. The closest to the local temperature T_{ref} is, the smaller is the variance in the calculation [52, 79]. If T_{ref} is estimated to be T, and the calculation of energy density yields $\delta e \neq 0$, the

system is out of steady state and e_{ref} will be modified at the next iteration. Steady state is reached when $\delta e \approx 0$ for a long period of time, and e_{ref} is not significantly modified. Surely, the effect of this stability changes with the number of particles: the larger is the population, the more accurate is the prediction.

3.2.7 Phonon-phonon scattering

The relaxation time approximation, described in Section 2.3, is used to model phonon-phonon scattering. With the local temperature estimation, the phonon lifetime can be interpolated for each particle. The occupation number n for each particle is then modified by using an exponential time scheme.

As shown in [64], along the characteristic lines $\mathbf{r}(t) = \mathbf{r}(t_0) + \mathbf{v}_{\mathbf{k}j}(t - t_0)$ the Boltzmann equation can be written as

$$\frac{dn_{\mathbf{k}j}(t)}{dt} + \frac{1}{\tau_{\mathbf{k}j}(t)} n_{\mathbf{k}j}(t) = \frac{n^0_{\mathbf{k}j}(t)}{\tau_{\mathbf{k}j}(t)}. \tag{3.50}$$

In the above equation the time dependence of each factor has been made explicit. In the Boltzmann equation, the phonon population is $n_{\mathbf{k}j}(\mathbf{r}, t)$, but along the characteristic lines it becomes $n_{\mathbf{k}j}(\mathbf{r}(t), t) \equiv n_{\mathbf{k}j}(t)$. The time dependence of the Bose-Einstein distribution and lifetime are obtained indirectly through the temperature, $n^0_{\mathbf{k}j}(T(\mathbf{r}(t))) \equiv n^0_{\mathbf{k}j}(t)$ and $\tau_{\mathbf{k}j}(T(\mathbf{r}(t))) \equiv \tau_{\mathbf{k}j}(t)$.

To solve Eq. 3.50 numerically, the time derivative is usually approximated using first order finite differences considering a small time step Δt after a current time t_0, such as we get

$$\frac{n_{\mathbf{k}j}(t_0 + \Delta t) - n_{\mathbf{k}j}(t_0)}{\delta t} + \frac{1}{\tau_{\mathbf{k}j}(t_0)} n_{\mathbf{k}j}(t_0) = \frac{n^0_{\mathbf{k}j}(t_0)}{\tau_{\mathbf{k}j}(t_0)}, \tag{3.51}$$

or

$$n_{\mathbf{k}j}(t_0 + \Delta t) = n_{\mathbf{k}j}(t_0) + \frac{\Delta t}{\tau_{\mathbf{k}j}(t_0)} (n^0_{\mathbf{k}j}(t_0) - n_{\mathbf{k}j}(t_0)). \tag{3.52}$$

This equation allows evaluating the phonon population at the next time step. However it assumes that $n^0_{\mathbf{k}j}$ and $\tau_{\mathbf{k}j}$ have little variation in one timestep, virtually assuming a constant temperature inside this interval. This method works well if $\Delta t \ll \tau_{\mathbf{k}j}$, but can lead to instability problems as $\Delta t \to 2\tau_{\mathbf{k}j}$, diverging the calculations beyond this limit. It clearly imposes a cap on the timestep that can be chosen by the user even if they wish to do quick approximate calculations and are not concerned with high precision.

The change of the phonon population in a time step δt can also be obtained after having integrated the Boltzmann equation. Indeed, Eq. 3.50 can be solved using the integrating factor

$$\lambda_{\mathbf{k}j}(t) = e^{\int^t \frac{ds}{\tau_{\mathbf{k}j}(s)}} \Longrightarrow \frac{d\lambda_{\mathbf{k}j}(t)}{dt} = \frac{\lambda_{\mathbf{k}j}(t)}{\tau_{\mathbf{k}j}(t)}. \tag{3.53}$$

We obtain

$$\lambda_{\mathbf{k}j}(t) \frac{dn_{\mathbf{k}j}(t)}{dt} + \frac{\lambda_{\mathbf{k}j}(t)}{\tau_{\mathbf{k}j}(t)} n_{\mathbf{k}j}(t) = \lambda_{\mathbf{k}j}(t) \frac{n^0_{\mathbf{k}j}(t)}{\tau_{\mathbf{k}j}(t)} \tag{3.54}$$

or, integrating between t_0 and $t_0 + \Delta t$,

$$n_{\mathbf{k}j}(t_0 + \Delta t) = \frac{\lambda_{\mathbf{k}j}(t_0)}{\lambda_{\mathbf{k}j}(t_0 + \Delta t)} n_{\mathbf{k}j}(t_0) + \int_{t_0}^{t_0+\Delta t} \frac{\lambda_{\mathbf{k}j}(t)}{\lambda_{\mathbf{k}j}(t_0 + \Delta t)} \frac{n^0_{\mathbf{k}j}(t)}{\tau_{\mathbf{k}j}(t)}. \tag{3.55}$$

To obtain Eq. 3.52, the lifetime and the Bose-Einstein distribution were assumed to be constant in the interval between t_0 and $t_0 + \Delta t$. If we use the same approximation here, then $\lambda_{\mathbf{k}j}(t) = e^{\frac{t-t_0}{\tau_{\mathbf{k}j}(t_0)}}$ and we obtain

$$n_{\mathbf{k}j}(t_0 + \Delta t) = n_{\mathbf{k}j}(t_0) e^{-\frac{\Delta t}{\tau_{\mathbf{k}j}(t_0)}} + n^0_{\mathbf{k}j}(t_0)(1 - e^{-\frac{\Delta t}{\tau_{\mathbf{k}j}(t_0)}}). \tag{3.56}$$

In the limit of small time step, Eq. 3.52 is recovered from Eq. 3.56 at linear order in Δt. Eq. 3.56 is the only discretization obtained from the integrating factor method we have tested, but Eq. 3.55 allows, in principle, to obtain better ones.

Figure 3.6 shows the comparison between the schemes given by Eq. 3.52 and Eq. 3.56. In order to compare them, an arbitrary linear profile of n^0 was chosen. This profile can approximate, in fairly high temperatures, the n^0 profile a particle would encounter in a linearly decaying temperature field. The relaxation time τ is considered to be constant in each plot. The initial occupation of the particle, $n_i = n(t = 0)$, was arbitrarily set to 2. For a linear profile $n^0 = at + b$, the analytical solution described by the dashed line is $n = a(t - \tau) + b + (n_i + a\tau - b)\exp(-t/\tau)$. In the long time limit, therefore, the occupation of the particle approximates $n = a(t - \tau) + b$, or the same linear profile at equilibrium shifted to the right by τ. To calculate this evolution, each plot applies the linear and the exponential schemes with different values of τ on each column, and different values of dt in each row. It is clear that, for small values of dt/τ, both methods are almost equivalent, becoming equal as $dt/\tau \to 0$. When $dt/\tau > 1$, instabilities start to arise with the linear scheme, and from $dt/\tau \geq 2$, they are unsustainable, leading to divergence. The exponential scheme, on the contrary, keeps itself stable despite the increase of dt/τ, and follows closely the analytic solution. The smaller is dt, the closer are both schemes. A smaller timestep also helps to stabilise the calculation, but increases the time needed. The effects of both schemes on the thermal conductivity will be discussed in Chapter 5.

3.2.8 Heat flux and thermal conductvitity

Thermal conductivity

The thermal conductivity κ can be estimated globally, for the total geometry, or locally in each space region, by applying Fourier's law in the direction of the temperature gradient of interest.

The global thermal conductivity can be estimated for fairly simple geometries, such as films or wires, since they have a defined temperature gradient, and hence a direction in which κ can be analysed. The total temperature gradient is imposed by the temperature difference between reservoirs, and the global heat flux is calculated considering all particles in the geometry,

$$\mathbf{\Phi}_{global} = \frac{1}{V_0 N} \frac{3N_a N}{N_p} \sum_{i=1}^{N_p} \hbar\omega_i \delta n_i \mathbf{v}_i \tag{3.57}$$

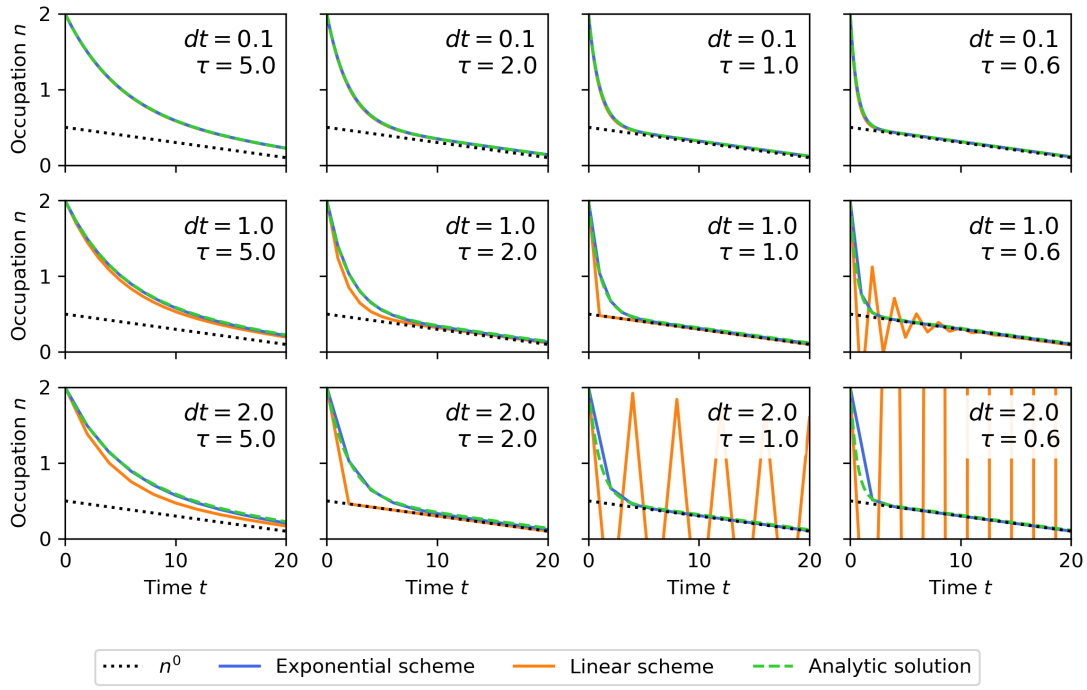


Figure 3.6: Comparison of the linear scheme used in previous works with the exponential scheme used in the present study.

where N_p is the instantaneous number of particles in the domain. This gives

$$\kappa_{global} = -\frac{\mathbf{r}_2 - \mathbf{r}_1}{T_2 - T_1} \cdot \mathbf{\Phi}_{global} \tag{3.58}$$

where $\mathbf{r}_r$ and T_r are the position and temperature of the reservoir number r. The local heat flux in the subvolume s is defined similarly by considering only the particles in the subvolume:

$$\mathbf{\Phi}_s = \frac{1}{V_0 N} \frac{3N_a N}{N_{ps}} \sum_{i=1}^{N_{ps}} \hbar\omega_i \delta n_i \mathbf{v}_i \tag{3.59}$$

where N_{ps} is the number of particles in the subvolume s.

A local thermal conductivity can then be calculated in two ways. First, in simple geometries simulating heat flux almost exclusively in one dimension, then κ_s, the thermal conductivity in subvolume s can be estimated by considering the local heat flux in the subvolume s and estimating the local ∇T from finite differences between the adjacent subvolumes, denoted as $s-1$ and $s+1$. This gives

$$\kappa_s = -\frac{\mathbf{r}_{s+1} - \mathbf{r}_{s-1}}{T_{s+1} - T_{s-1}} \cdot \mathbf{\Phi}_\mathbf{s} \tag{3.60}$$

However, if the subvolumes are defined in such a way that the problem becomes two or three-dimensional, then a local thermal conductivity is calculated in between each two subvolumes, denoted as s_1 and s_2, by taking the average heat flux and the temperature gradient between them. This gives

$$\kappa_{c12} = -\frac{\mathbf{r}_{s_2} - \mathbf{r}_{s_1}}{T_{s_2} - T_{s_1}} \cdot \frac{\mathbf{\Phi}_{s_2} + \mathbf{\Phi}_{s_1}}{2} \tag{3.61}$$

and it is supposed that $\kappa_{c_{12}}$ is the thermal conductivity along the connection between s_1 and s_2.

Chapter 4

Algorithm

This chapter aims to describe how Nano-ᴋ calculates each operation in the simulation. First the programming language choice and the structure of the code will be explained. Then, each class will be detailed, together with the operations that are performed. For that, a set of parameters is used as example, including an explanation of every parameter in the simulation, and output files.

4.1 Programming language

Writing a software from scratch while still aiming for great flexibility is difficult to plan for. One can never know for sure what types of problems will be found in the way, and the starting programming language should reflect this uncertainty. It should be general purpose, widely used so that there is enough public knowledge available to be consulted, and have available tools that could be helpful to solve possible problems.

For these reasons, Nano-ᴋ was chosen to be written in Python [80, 81]. Python is a high-level, general purpose, interpreted language that has been widely used for several years [82]. It is open source, which allows freedom of development and of intellectual property. Because of it, Python has a wide community that develops their own tools, often offering them back to other developers as open source libraries.

The main setback with Python regarding its application in computational physics is the fact that it is interpreted. In programming, an interpreter translates the typed code to machine code line by line, and stops the execution when it finds an error. It is mainly good for development of small codes that execute operations only once. This gives it a wide disadvantage in comparison to compiled languages, such as C or FORTRAN. In opposition to an interpreter, a compiler reads the code once and stops immediately if the code has any errors. During compilation, the compiler applies optimisations at low level and outputs an executable with the whole code translated to machine code. This makes compiled languages more efficient for speed and repetition.

Nevertheless, there are ways to bypass these restrictions. Some libraries such as SciPy [83] and Numpy [84] offer optimisations by pre-compiling their functions in C, C++ or FORTRAN, while still offering an interface to use in Python programs. This allows to remove loops in the code and to perform operations on vectors and matrices in a more efficient way. In the context of Nano-κ, this also shows that eventual performance bottlenecks can be optimised by compiled language without

compromising its general structure.

4.1.1 Object-oriented programming and terminology

In order to make the text comprehensible for scientists that are not specialised in programming, some explanation on the terminology used in the next sections may be important. More detailed discussion about the topic can be found in Phillips [85] and Lutz [86].

Nano-κ is loosely written in the object-oriented programming (OOP) paradigm. This paradigm defines *classes*, which are dedicated types of *objects* with their own functions and *attributes*. An attribute is a value that characterises an object of that class. Several different objects of the same class can be created, each with their own individualised attribute values. When an object of a given class is created, usually the word *instantiated* is used, since the object is an *instance* of its class. The operations that an object can execute are defined by internal functions called *methods*. A particular method is called every time an object is instantiated. This method is called the *constructor*, and is responsible to initialise the basic attributes of the object.

An important chracteristic of the OOP paradigm is "inheritance". A class can have subclasses that derive some of its attributes and methods. Inheritance can happen in several levels of specialisation, avoiding unnecessary multiple declarations of the same variables. One of the classical examples of inheritance in OOP is the definition of a `Pet` class, and two other classes `Cat` and `Dog`, both inheriting from `Pet`. For example, the `Pet` class may have the attribute `Pet.colour`. Every object of `Dog` and `Cat` class will inherit this attribute as well. Nevertheless, each instance of the same class can have different individual values for the same attribute. If two cats are instantiated in the code, `cat1` and `cat2`, they can be declared such that `cat1.colour = 'black'` and `cat1.colour = 'grey'`.

Other trait is encapsulation. By defining operations as methods of a class, the programmer can control which level of access other objects in the code will have to a given operation. For example, the `Cat` class can have the method `Cat.meow()` and the `Dog` class the method `Dog.bark()`. A cat (better, an instance of `Cat`) in the code cannot bark, since the access to the dedicated method is only available to dogs.

The OOP paradigm also has at its core the idea of polymorphism, i.e. different objects of the same class can have different attribute values. One example of Polymorphism is the differences between `Cat` and `Dog`, while both are `Pet`. They can have the same methods as `Pet`, while being slightly changed to better re-purpose them for their specific contexts.

The OOP gives a great flexibility for adapting and reusing the code in different contexts, which allows for easier improvements in the long term.

4.2 Code structure, inputs and outputs

The code organises the relevant information into six classes for better compartmentalisation: Constants, Mesh, Geometry, Phonon, Population and Visualisation. Each class deals with a scope, and mostly serve as auxiliaries to the Population

class, where the simulation itself takes place. Figure 4.1 shows the created objects and their relations inside the code.

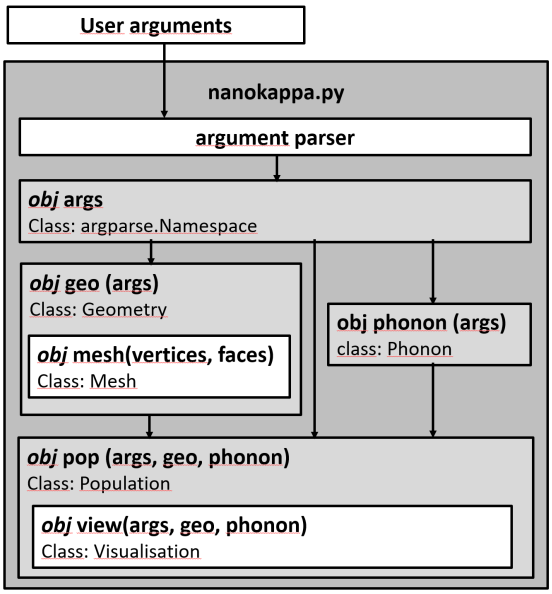


Figure 4.1: Code structure with relationships between objects. Internal boxes mean objects are instantiated inside another object, as its attribute.

- The `Constants` class stores all relevant physical constants and conversion factors to be inherited by the requiring classes so that these values are always available;
- The `Mesh` class stores the information about a single three-dimensional triangular mesh. It is instantiated as an attribute of `Geometry`.;
- The `Geometry` class loads, processes and manages all information about the relationships among the triangular mesh, the boundary conditions and subvolumes;
- The `Phonon` class loads and processes material data from the *ab-initio* simulations and stores the relevant methods to calculate energy density, relaxation times and interpolate temperature from energy. Each `Phonon` object represents a different material. Inherits the physical constants from `Constants`;
- The `Population` class is where the simulation itself takes place, using Phonon and Geometry objects as inputs along with additional parameters. It needs therefore one single Geometry object and one Phonon object Inherits the physical constants from `Constants`;
- The `Visualisation` class takes the information from the convergence files and `Population` object and translates them into figures to be analysed by the user as the simulation runs. A `Visualisation` object is created inside the `Population` object, and the `Geometry` and `Phonon` objects are passed on by `Population` to `Visualisation`. Inherits the physical constants from `Constants`.

The main script, `nanokappa.py` takes the input parameters and parses them with the `argparse` standard Python module. The parameters are stored in an `argparse.Namespace` object that is passed as argument to the other classes. The code then generates objects of the `Geometry`, `Phonon` and `Population` classes, and iterates the calculation until one of the termination conditions has been met. The parameters can be passed directly on command line or through the use of a file. In the latter case, Nano-κ should be executed with:

```
$ python nanokappa.py --from_file <file_name>
```

where `--from_file` can be substituted by its short form `-ff`. Every parameter have its long and short forms. These can be consulted in Appendix A or in the online documentation [87].

The following example case will be the calculation of heat transfer in a thin film, in the in-plane direction. The parameters, that will be used and explained in the next sections are the following:

```
# geometrical parameters
--geometry       box
--dimensions     2e4 2e4 1e3
--subvolumes     slice 20 0
--bound_pos      relative 0 0.5 0.5 1 0.5 0.5 0.5 0.5 0 0.5 0.5 1
--bound_cond     T T R R P
--connect_pos    absolute 1e4 0 5e2 1e4 2e4 5e2
--bound_values   302 298 10 10

# material data
--mat_folder     D:/LEMTA/Code/Materials/Si/
--hdf_file       kappa-m313131.hdf5
--poscar_file    POSCAR

# calculation parameters
--temp_dist      linear
--temp_interp    linear
--particles      total 1e6
--timestep       1
--iterations     10000
--conv_crit      0.01 5
--n_mean         100
--max_sim_time   0-03:00:00

# outputs
--results_folder test_results/example_inplane
--theme          white
--colormap       jet
--fig_plot       e
```

The following parameters are omitted since they take default values or no value at all:

```
--isotope_scat
--scale          1 1 1
--geo_rotation
```

```
--mat_rotation
--part_dist      random_subvol
--subvol_temp
--output         file
```

There are different types of outputs that Nano-κ generates. The first are text files, each describing a particular set of data (particles, subvolumes or evolution over time, for example). These are produced in the `Population` class. The other type are figures, that are produced by different classes depending on the context. Each of them will be detailed in the dedicated section of the responsible class.

4.3 Constants

The `Constants` class stores the values of every useful physical constants and conversion factors. Its declaration is composed only of a constructor that requires no arguments, defining the values of the constants. They are derived from `scipy.constants` library and converted to the adequate units. These are:

- Reduced Planck's constant: $\hbar = 6.582\,119\,569 \times 10^{-4}\,\mathrm{eV \cdot ps}$
- Boltzmann constant: $k_b = 8.617\,333\,262 \times 10^{-5}\,\mathrm{eV/K}$
- Electron volt to Joule: $1\,\mathrm{eV} = 1.602\,176\,634 \times 10^{-19}\,\mathrm{J}$
- Angstrom to metre: $1\,\mathrm{Å} = 1 \times 10^{-10}\,\mathrm{m}$
- Picosecond to second: $1\,\mathrm{ps} = 1 \times 10^{-12}\,\mathrm{s}$
- Heat flux conversion factor: $1\,\mathrm{eV/(psÅ^2)} = 1.602\,176\,634 \times 10^{13}\,\mathrm{W/m^2}$
- $\pi = 3.141592653589793$

The classes that deal constantly with converting units are `Phonon`, `Population` and `Visualisation`. `Geometry` and `Mesh` don't need it since the coordinates in space only acquire physical meaning during the simulation. The preparation of the geometry itself is independent of unit.

4.4 Mesh

Nano-κ simulates geometries that are constructed as triangular meshes. These geometries are defined by two entities: a set of vertices in 3D space and a list of faces referencing the vertices that compose them. The class that interprets this information and derives all other necessary attributes is the `Mesh` class. The inputs that will be given to the `Mesh` class are passed by the `Geometry` class, that will be detailed in the next section.

The `Mesh` class takes as inputs the vertices and the faces of the mesh. A set of N_v vertices are passed as a 2D array of dimensions $N_v \times 3$, containing the coordinates of the vertices with one column for each dimension x, y and z. Similarly, all N_f faces of the mesh are described by a 2D array with dimensions $N_f \times 3$. The face array

contains integers, which are the indices of the vertex that compose each face. They do not have to be in any particular order. Internally, these two arrays are saved as $Mesh.vertices$ and $Mesh.faces$.

A tolerance is first defined as 10^{-10} units. This tolerance is very small, since the standard unit length of Nano-κ is angstroms. Nevertheless it is still large enough to avoid numerical errors. The mesh is processed by the `Mesh.update_mesh_properties` method, described in Algorithm 1.

Algorithm 1 Process or update mesh

```
function MESH.UPDATE_MESH_PROPERTIES(self, remove_unref = True, triangulate_volume = True)
    Save number of vertices;
    Save self.bounds and self.extents;
    if remove_unref then
        self.remove_unref_vertices(update = False)
    end if
    self.get_faces_properties
    self.get_edges
    self.get_face_adjacency
    self.get_facets_properties
    self.get_interfaces
    self.check_winding
    if triangulate_volume then
        self.get_volume_properties
    end if
end function
```

The `Mesh.bounds` attribute defines the bounding box of the mesh (the smallest box that contains all vertices). The `Mesh.extents` attribute is the length of the bounding box in each direction. The `Mesh.remove_unref_vertices` method is the responsible to clean the vertices list of the unused points. They are removed by checking the vertex indices that are not present in `Mesh.faces`.

The call to `Mesh.update_mesh_properties` makes sure that everything will be appropriately adjusted after the removal. During initialisation, this call is not executed (`update = False`), since the mesh will already be processed next.

The next properties to be calculated are those related to faces. These are face normals, areas, centroids, etc. This is done as in Algorithm 2.

Every face data is stored in Numpy arrays so that they can be referenced by the face index. For example, for every array, the properties of face 2 will be accessed with the index 2 in the appropriate array dimension. This allows to pre-compute every necessary property and to apply vectorized operations, avoiding loops.

The edges are then analysed. Edges are the lines connecting pairs of vertices. Every triangle is composed of 3 edges. Their information is stored in:

- `Mesh.edges`: an array of dimensions $N_e \times 2$ with the indices of the connected vertices, sorted by row and column;
- `Mesh.face_edges`: an array of dimensions $N_f \times 3$ with the indices of the edges that outline each face;

Algorithm 2 Calculate faces properties

function MESH.GET_FACES_PROPERTIES(`self`)
  `self.n_of_faces` ← rows in `self.faces`
  for every face **do**
    Calculate face basis
    Calculate face normal (Eq. 3.14)
    Store matrix for change of basis to face
    Store face origin
    Calculate face area (Eq. 3.15)
    Calculate face centroid (Eq. 3.16)
    Calculate face plane constant $\mathfrak{K} = -\mathbf{n} \cdot \mathbf{r}_o$
    Calculate face bounding box
  end for
  Calculate total surface area
end function

- `Mesh.edges_faces`: a list with N_e elements, each element being an array with the indices of the faces that are outlined by each edge.

With the information on edges, it is possible to get which faces neighbour each other. Two faces are neighbours when they share the same edge (and consequently two of their vertices). Knowing the connection between faces, the facets can be identified. For each pair of adjacent faces, their normals are checked. If they are equal (considering a numerical tolerance), they are in the same facet, and are registered as coplanar. Then, the pairs of coplanar and adjacent faces are grouped in a list, where each element contains the indices of the faces that compose each facet. Other properties of the facets are also stored:

- `Mesh.facets_normal`: the normal of each facet;
- `Mesh.facets_centroid`: the centroid of each facet, as a mean of the centroids of the triangles that compose it, weighted by area;
- `Mesh.facets_boundary`: the edges in the facet that are located at the boundaries;
- `Mesh.facets_area`: the area of the facets, calculated by summing the area of the triangles that compose it.

Lastly, the volume of the mesh is calculated. To do it the mesh is divided into tetrahedrons with SciPy's Delaunay triangulation function [88]. Because SciPy's procedure does not deal with concave surfaces, auxiliary points, disposed in a 3D grid, are generated. The edges are also subdivided to help guiding the triangulation algorithm. These points help to guide the triangulation and allow for tetrahedrons outside the geometry to be identified and deleted.

Besides calculating the total volume of the mesh V, the tetrahedrical discretisation is also used to sample particles in the volume, and to check whether particles are inside or outside the domain. The volumetric sampling process was already described in Sec. 3.2.1, and uses Eqs. 3.25, 3.28 and 3.29. Particles that are inside the domain have only one tetrahedron in which their barycentric coordinates are

between 0 and 1. Instead of calculating the barycentric coordinates of every particle for every tetrahedron, it is faster to first check whether they are inside the bounding box of any of them. If so, only then the barycentric coordinates $[a_1, a_2, a_3, a_4]$ are calculated only for the relevant tetrahedrons for each particle. The matrices used for this conversion and the bounding boxes for each tetrahedron are pre-computed at initialisation. Figure 4.2 shows a 2D scheme of the particle detection steps.

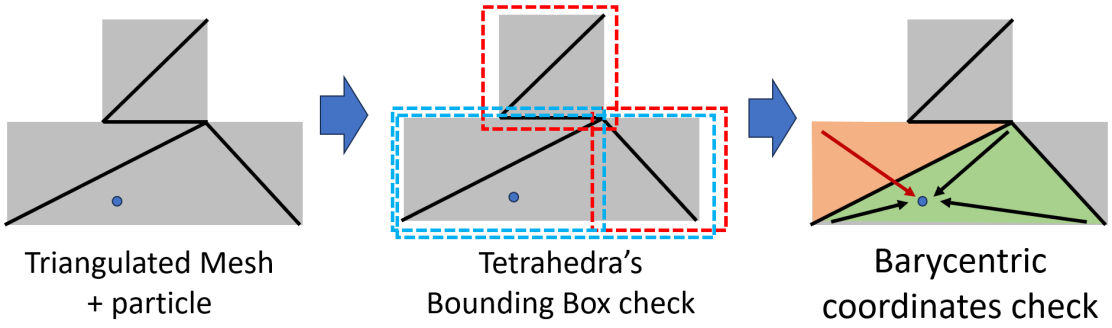


Figure 4.2: Volumetric particle detection.

During the simulation, the detection of collision between particles and the facets of the mesh is performed inside the `Mesh` object. The procedure is shown in Algorithm 3.

Algorithm 3 Find collision between particle and mesh

function MESH.FIND_BOUNDARY(`self`, position $\mathbf{r}$, velocity $\mathbf{v}$)
  Find the parameter t so that $\mathbf{n} \cdot (\mathbf{r} + t\mathbf{v} - \mathbf{r}_o) = 0$ for every facet;
  for each facet F where $t \geq 0$, in crescent order **do**
    Calculate the barycentric coordinates of the collision point $\mathbf{r}_c = \mathbf{r} + t\mathbf{v}$ for every triangle in F;
    if any triangle gives barycentric coordinates between 0 and 1 **then**
      return $\mathbf{r}_c$, t, F
    end if
  end for
  return inf, inf, -1
end function

Whenever the code has to plot the geometry, the `Mesh` class has a dedicated method to do it automatically. It takes the edges on the boundaries of the facets, and plots them in 3D. Figure 4.3 shows an example. This is used when plotting boundary conditions and the subvolume distribution, for example. Some of these plots are shown in the next section.

4.5 Geometry

4.5.1 Initialisation

The `Geometry` class is a shell class that creates and stores the `Mesh` object, as well as relates the mesh with boundary conditions and subvolumes.

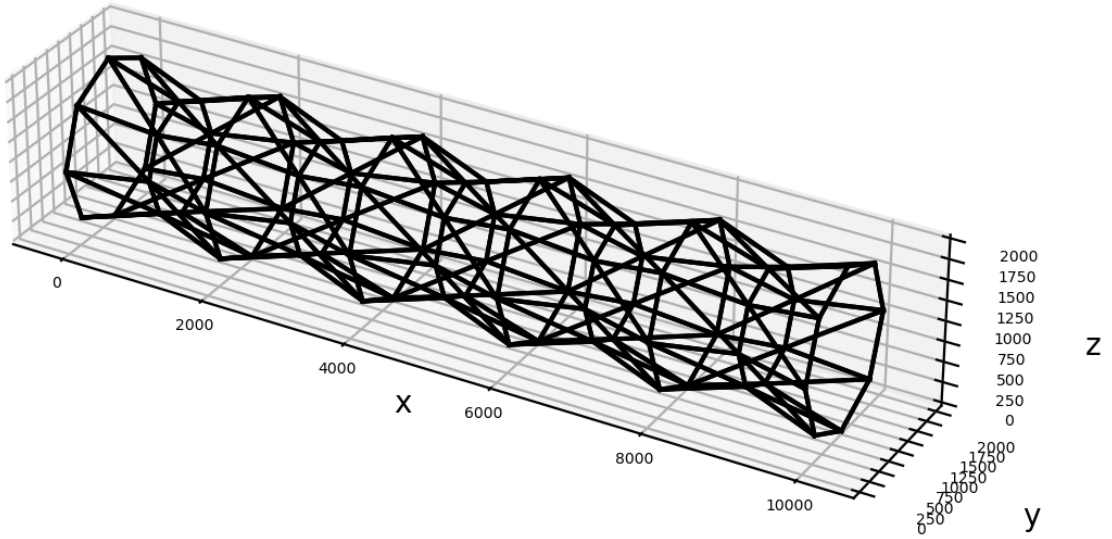


Figure 4.3: Example of plotted mesh.

The first step is to load the geometry and store the `Mesh` object as an attribute. This is done based on user inputs. In the example case, the geometrical parameters that are being applied are:

```
--geometry       box
--dimensions     2e4 2e4 1e3
--bound_pos      relative 0 0.5 0.5 1 0.5 0.5 0.5 0.5 0 0.5 0.5 1
--bound_cond     T T R R P
--connect_pos    absolute 1e4 0 5e2 1e4 2e4 5e2
--bound_values   302 298 10 10
--subvolumes     slice 20 0
```

The `--geometry` parameter sets the type of geometry being worked with. Some standard geometries are supported by Nano-κ (see Appendix B). These have their vertices and faces internally pre-defined, calculated according to the values passed to `--dimensions`. The unit of length used by Nano-κ is angstroms. In the example, the geometry is a parallepipedic box, measuring $[2 \times 10^4$ Å, 2×10^4 Å, 1×10^3 Å] (or [2 µm, 2 µm, 100 nm]. Figure 4.4 shows an scheme. The name of a STL file can also be passed to `--geometry`, and will be loaded and processed accordingly. Figure 4.5 show some examples of geometries and the parameters used to generate them.

After the mesh processed, it is transformed. If the user wishes, they can rescale or rotate the mesh by using the `--scale` and `--geo_rotation` parameters. The first parameter is defined by the rescaling factors in x, y and z directions, respectively (e.g. `--scale 1 2 1` to stretch the geometry to double its size in the y direction). The rotation is declared by passing the desired Euler angles in degrees and the order of the axis to rotate. Up to three consecutive rotations are supported. The rotations can be extrinsic, by using the fixed coordinate system (X, Y, Z) of the tridimensional space where the geometry is generated; or intrinsic, by considering the coordinate system (x, y, z) that moves with the geometry. Initially, while no rotation takes place, both have their unit vectors pointing at the same directions, $x = X,\ y = Y,\ z = Z$.

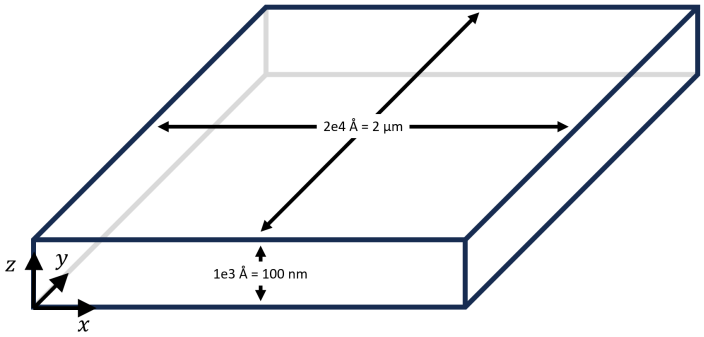


Figure 4.4: Geometry used as example.

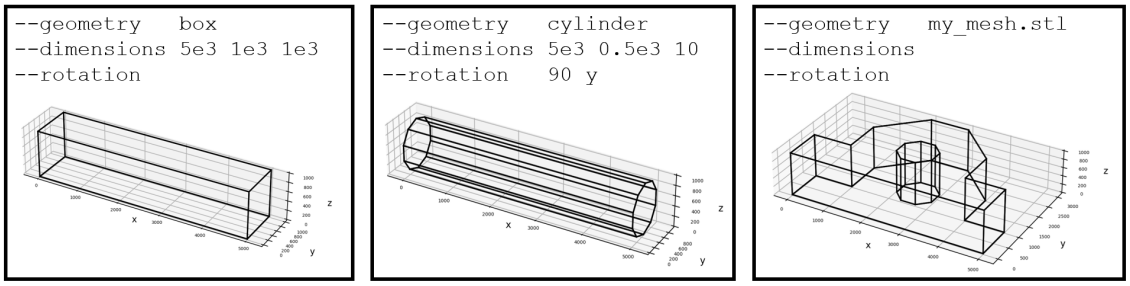


Figure 4.5: Examples of Nano-κ standard geometries and the parameters that generated them.

Figures 4.6 and 4.7 show an example of two sets of rotations with the same angle values, one extrinsic and other intrinsic. Figure 4.6 shows an scheme of how the geometry's coordinate system moves with each rotation, while Fig. 4.7 compares the final results on a quasicylindrical mesh. The quasi-cylinder is generated upright, with its quasi-circular bases at the bottom and top. Therefore, its intrinsic z axis goes from base to base. The extrinsic rotation uses the parameter values `--geo_-rotation 60 45 YZ`. This sequence rotates the geometry 60 degrees in Y, then rotates it again 45 degrees in Z. The intrinsic rotation used the parameters `--geo_-rotation 60 45 yz`, therefore using the same angles but in the intrinsic (moving) coordinate system. Since the intrinsic z axis was rotated together with the geometry, the 45 degrees rotation in z keeps the cylinder in place, rotating it only around its symmetry axis.

It is important to note that the rescaling is always applied *before* the rotation, and that the mesh is always translated to ensure that all vertices have positive coordinates.

4.5.2 Boundary conditions

There are four parameters that are used to declare the boundary conditions (BC):

- `--bound_cond`: the types of boundary conditions. `T` for imposed temperature, `R` for rough walls, or `P` for periodic (to be explained later);
- `--bound_values`: values of each temperature and roughness;
- `--bound_pos`: reference points in the tridimensional space that are used to identify the closest facet to each of them, onto which the declared boundary

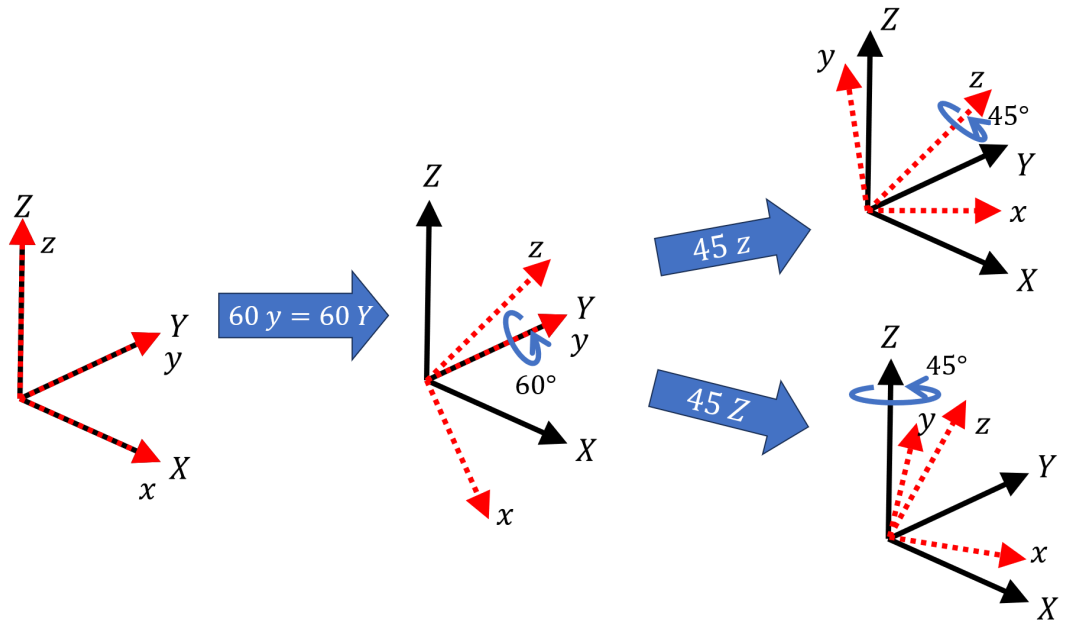
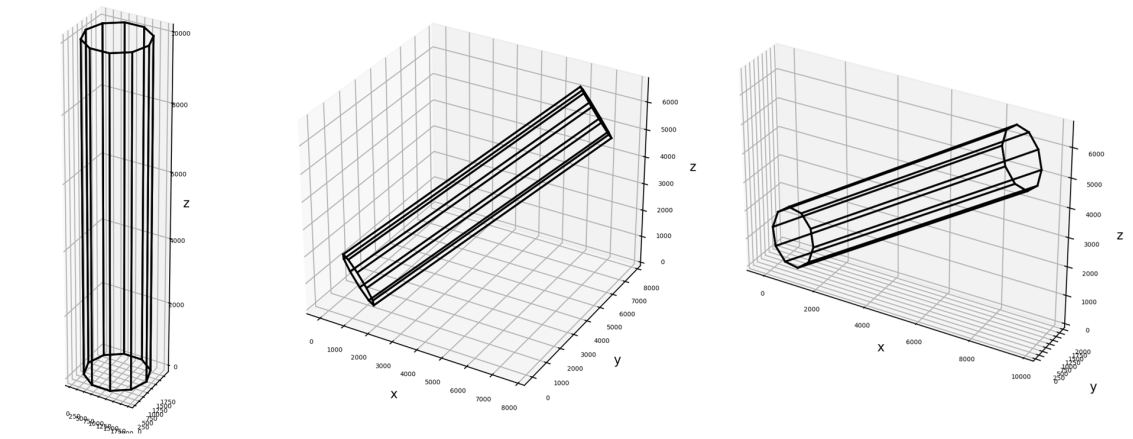


Figure 4.6: Rotations of the geometry coordinate system (x, y, z) in relation to the fixed coordinate system (X, Y, Z) using the same angle values for intrinsic and extrinsic rotations.



```
--geometry cylinder
--geo_rot
```

Initial

```
--geometry cylinder
--geo_rot  60 45 YZ
```

Extrinsic

```
--geometry cylinder
--geo_rot  60 45 yz
```

Intrinsic

Figure 4.7: Comparison between final geometry positions after an extrinsic and an intrinsic rotation for the same angles.

conditions will then be applied;

- `--connect_pos`: in the case of periodic BC, which facets are connected to which ones. These are also declared by defining points in space, and recovering the closest facet to each of them.

The declaration of all boundary conditions have to be consistent in their order of declaration. The points declared in `--bound_pos` and `--connect_pos` are first defined with the help of a keyword `absolute` or `relative`. The former (`absolute`) tells that the coordinates that will be read have absolute values; the latter (`relative`), that the coordinates are relative to the bounding box of the geometry, and have to be rescaled to obtain the absolute values. The reference points and boundary conditions used in the example are shown in Fig. 4.8.

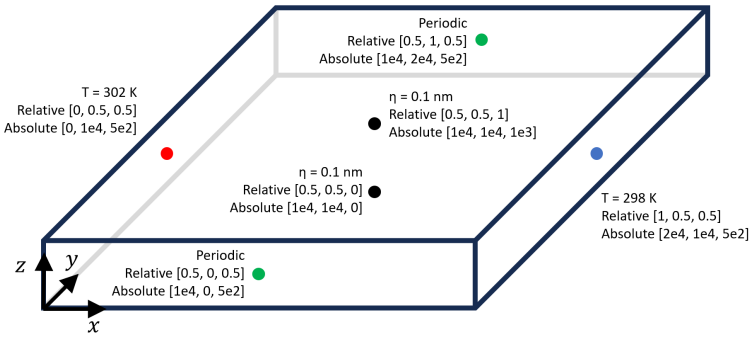


Figure 4.8: Schematics of the setting of boundary conditions for estimation of in-plane thermal conductivity in a thin film.

In the example, the first boundary condition is an imposed temperature of 302 K at the facet closest to the relative position $[x, y, z] = [0, 0.5, 0.5]$ (red point in Fig. 4.8). Since the bounding box of the geometry in this case has the exact dimensions of the created box, the absolute coordinates of the point are $[0 \cdot 10^4, 0.5 \cdot 10^4, 0.5 \cdot 10^3] = [0, 5 \times 10^3, 5 \times 10^2]$, which is on the center of the facet located at $x = 0$. The second one is another imposed temperature, this time 298 K, at relative position $[1, 0.5, 0.5]$. Therefore, the heat transfer will happen in the x direction, since the temperature difference is imposed between the extremities in x. The next two boundary conditions are rough walls, imposed on the bottom and top facets (relative coordinates $[0.5, 0.5, 0]$ and $[0.5, 0.5, 1]$).

The last boundary condition, `P`, indicates a periodic BC. This can be used when dealing with geometries that are periodically repeated. When that is the case, the period (i.e. each repeated unit of the geometry) has to have the same boundary conditions imposed. This periodicity can be applied in any direction. The restrictions for it to be valid are:

- The normals of the connected facets must be parallel and pointing in opposite directions;

- The boundaries of the facets (the edges that delimit the facet) must be equal.

This restriction on the normals come from the fact that dispersion relations are highly dependent on the orientation of the crystal. Even if the material is the same on both sides of the connection, connected facets that are not on parallel planes

will be oriented differently in relation to the crystal, much like an interface between grain boundaries. If this was the case, a transmissivity calculation would have to be considered, which is a feature currently not supported in Nano-κ. Figure 4.9 shows examples of valid periodic geometries in two dimensions.

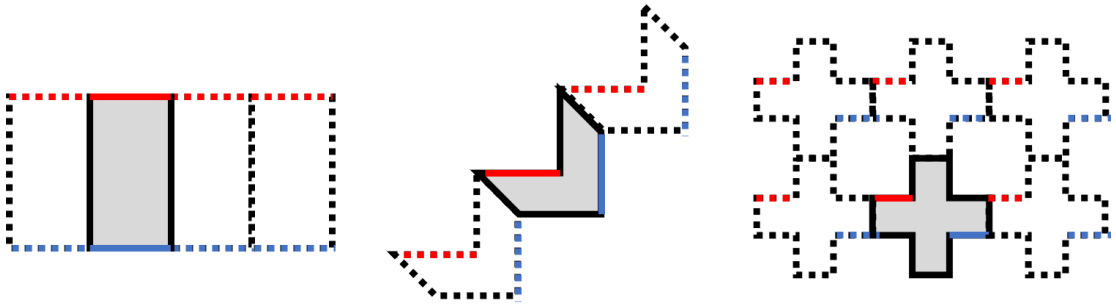


Figure 4.9: Examples of periodic geometries along one or two directions. Nano-κ accepts periodicity in all 3 directions. The red and blue lines represent the imposition of hot and cold temperatures, respectively. All the remaining non-periodic faces should have an associated roughness as BC.

When a particle meets a facet with periodic BC, it does not change mode as it would on reflections, nor it is absorbed as it would by reservoirs. In this case the particle is simply translated to the connected facet. The details of the algorithms that deal with BCs will be given in Section 4.7.

Coming back to our example simulation, the two facets that are interpreted by the code to have `P` as boundary condition are the ones on both extremities in y direction (see Fig. 4.8). Nevertheless, the code has to be informed that they are connected. This is passed to the `--connect_pos` parameter. Connected facets are declared in pairs. If $\mathbf{p}_1$, $\mathbf{p}_2$, $\mathbf{p}_3$ ... $\mathbf{p}_n$ are the points passed to `--connect_pos` and F_1, F_2, F_3 ... F_n their correspondent facets to be found by the code, F_1 is connected to F_2, F_3 to F_4 and F_{n-1} to F_n. Since each facet can only be connected to another one, n must be an even number.

With these parameters, all the facets have a proper boundary condition. A visualisation of the whole setting can be seen in Fig. 4.10. Figure 4.11 shows the output plot from Nano-κ.

4.5.3 Subvolumes

The next step is to divide the geometry in subvolumes so it is possible to calculate local quantities, such as energy density, temperature and heat flux. Their distribution is done by placing reference points in different positions, and associating each of these points with a different subvolume. The particle is said to be contained in a given subvolume s when it is closer to the reference point positioned at $\mathbf{r}_s$ than to any other reference point, creating a Voronoi decomposition of the geometry. Nano-κ allows three ways in which subvolumes can be distributed: `slice`, `grid` and `voronoi`.

When problems have one major direction of heat transfer, such as in films or wires, the geometry can be *sliced*. This means that the reference points are created along a line crossing the center of the bounding box, in the desired direction. They

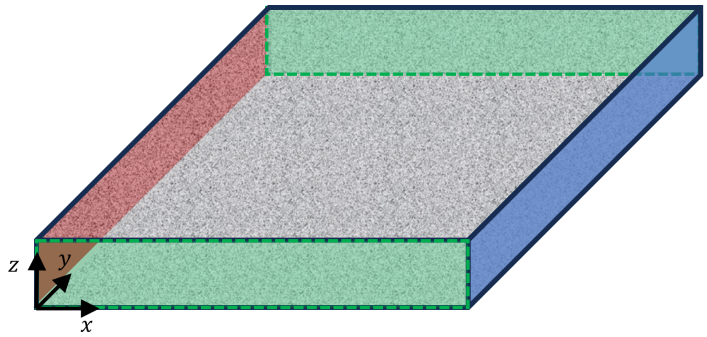


Figure 4.10: Illustration of the example problem with all boundary conditions set. Hot and cold temperatures are imposed on the red and blue facets respectively. The facets shaded green have periodic boundary conditions. The grey facets have an associated roughness.

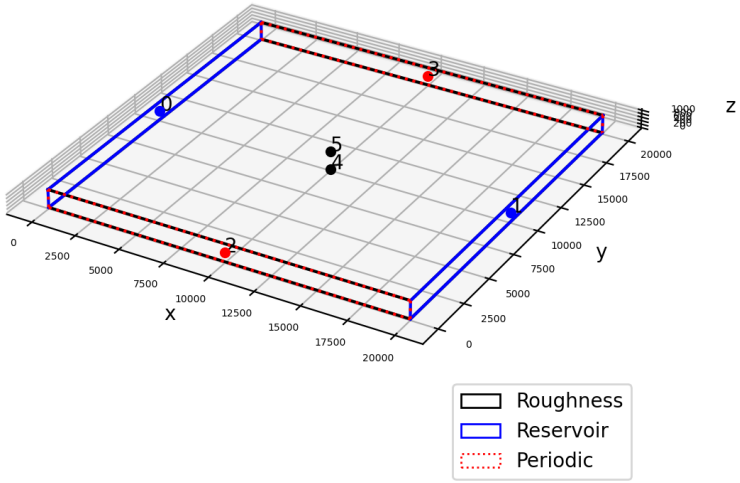


Figure 4.11: Nano-κ plot of the boundary conditions.

are positioned equally spaced, so every slice has the same thickness. This is the one used in the example, and Figure 4.12 shows the reference points generated with the parameters. To use this type of subvolume, the `--subvolumes` parameter is declared as follows:

```
--subvolumes slice <number> <axis>
```

The argument `<number>` refers to the number of desired subvolumes and `<axis>` is the desired direction (0 for x, 1 for y and 2 for z). In the example `<number>`= 20 and `<axis>`= 0, distributing the reference points along x.

When geometries are more complex, with holes and corners, or when a significant portion of the heat transfer can happen in more than one direction, the `grid` keyword can be used. In this case, the reference points are generated by slicing the bounding box in all 3 directions. The parameters to be informed are then the desired number of subvolumes in each of the three directions:

```
--subvolumes grid <number x> <number y> <number z>
```

The code initially creates all points, a total of $N_x \times N_y \times N_z$. However only those that are contained inside the mesh are kept. This exclusion of points outside

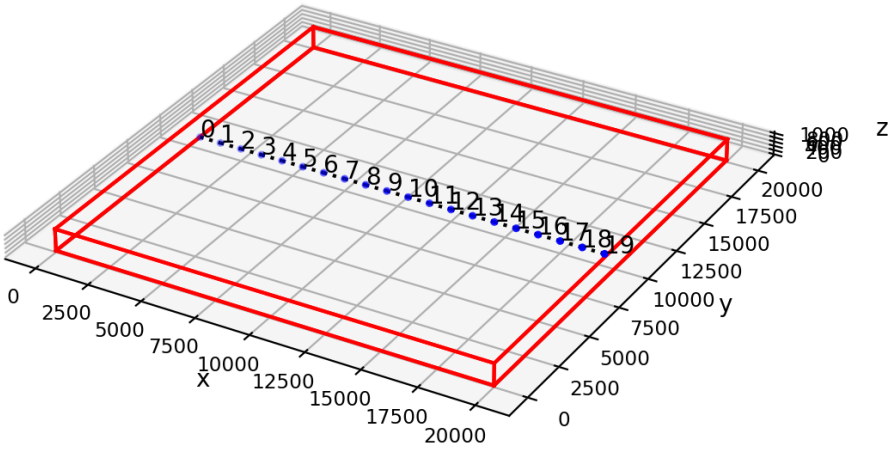


Figure 4.12: Example geometry with slice subvolumes.

the mesh is not done for slice geometries, since slices span on the two directions perpendicular to the direction of slice (y and z if slicing is applied on x, for instance), and there will be particles in that slice even if the reference point is not in the mesh. Here, if a particular region of interest is not properly divided by a subvolume, the user can change the parameters such that it is as they want. Figure 4.13 show an example of a geometry with a `grid` subvolume distribution.

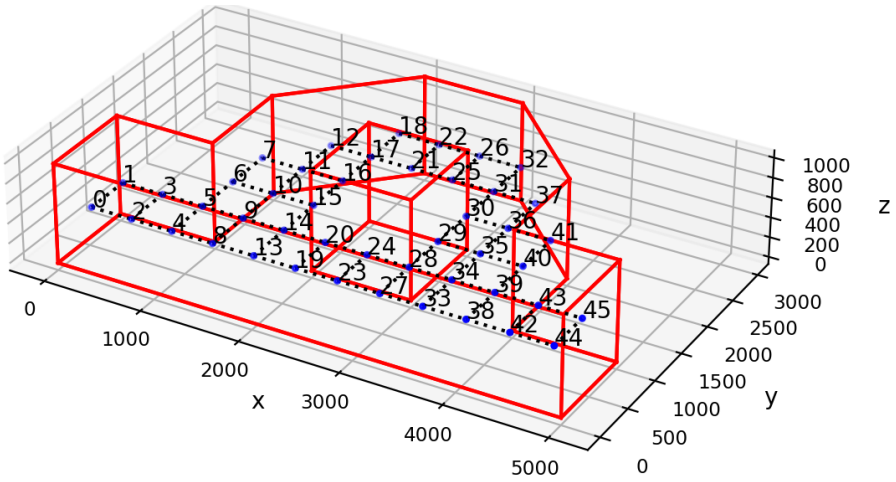


Figure 4.13: Examples of a geometry with grid subvolumes. Parameters: `--subvolumes grid 12 6 1`.

Lastly, when none of the previous cases apply, the user can generate an unstructured subvolume distribution with the keyword `voronoi`. In this case, the only value to be declared is the number of subvolumes N_s. In this case, the initial reference points are randomly generated in the domain. The algorithm then uses sample points, which are uniformly distributed in the geometry, to assess the subvolume distribution. This assessment is done by classifying every sample to their closest reference point. The reference points are then moved to the mean position of their respective samples ("center of mass" of the subvolume they are representing). This is done iteratively until all the N_s reference points stop moving. Algorithm 4 shows the procedure used to distribute the reference points, and Figure 4.14 shows an ex-

ample of a geometry with Voronoi subvolumes. The declaration is done as:

```
--subvolumes voronoi <number>
```

Algorithm 4 Voronoi subvolume distribution

function VORONOI_DISTRIBUTION(N_s, $crit$, N_p^0, $N_p^{\max}$)
  $N_p \leftarrow N_p^0$
  Generate N_p point samples, with positions $\mathbf{r}_p$
  Generate N_s reference points, with positions $\mathbf{r}_s$
  while True **do**
    Classify each $\mathbf{r}_p$ to the closest $\mathbf{r}_s^k$
    if Any $\mathbf{r}_s^k$ does not have any correspondent $\mathbf{r}_p$ **then**
      Re-generate those reference points randomly
    end if
    Calculate $\mathbf{r}_s^{k+1}$ as the mean position of the correspondent $\mathbf{r}_p$
    $\epsilon \leftarrow \max(\|\mathbf{r}_s^{k+1} - \mathbf{r}_s^k\|)$
    if $\epsilon < crit$ **then**
      if $N_p == N_p^{\max}$ **then**
        return $\mathbf{r}_s$
      else
        $N_p \leftarrow \min(2N_p, N_p^{\max})$
      end if
    end if
  end while
end function

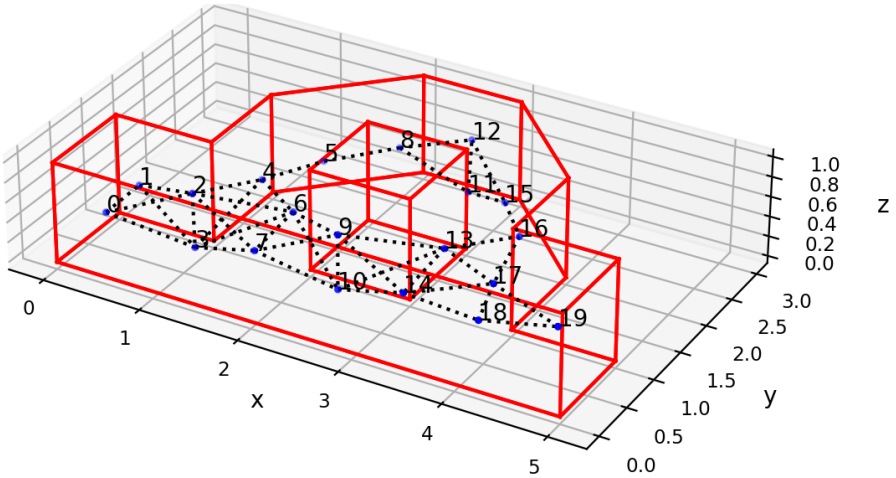


Figure 4.14: Examples of a geometry with Voronoi subvolumes. Parameters: `--subvolumes voronoi 20`.

After the reference points for each subvolumes are set, it is necessary to generate the connections between subvolumes. They allow for local estimation of temperature gradient, heat flux and thermal conductivity. For the slice case, it is simply a matter of connecting the each subvolume to the next (subvolume 1 to 2, 2 to 3, 3 to 4, and so on).

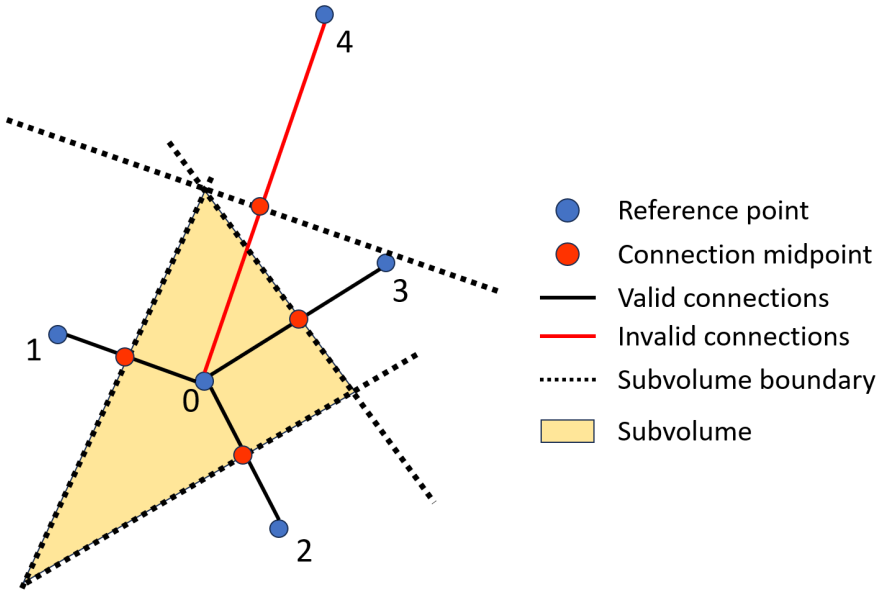


Figure 4.15: Schematics in 2D of the identification of valid subvolume connections. Connection with subvolume 4 is invalid because its midpoint is behind the boundary plane with subvolume 3.

For grid and Voronoi subvolumes, the procedure is more complex. First, connections between every pair of subvolumes are considered. The only connections that are kept are those on which heat can be transfered directly between subvolumes in a straight path, beginning in one reference point and ending in the other. The first selection is to remove every connection that crosses a facet. The second, to remove all those that cross a subvolume outside of the pair in the connection. In order to do this, the boundary planes between subvolumes are calculated.

A 2D scheme of this analysis is shown in Fig. 4.15. For example, let's analyse the connection $(0, 4)$ between subvolume 0, with reference point $\mathbf{r}_0$, and subvolume 4 with reference point $\mathbf{r}_4$. In this example, subvolume 0 has already been confirmed to be connected to other three subvolumes, 1, 2 and 3. The plane that defines the boundary between subvolumes 0 and 4 can be generated by using the vector of their connection as the plane normal, $\mathbf{n}_{0,4} = \mathbf{r}_4 - \mathbf{r}_0$, and the midpoint as the plane origin, $\mathbf{o}_{0,4} = (\mathbf{r}_0 + \mathbf{r}_4)/2$. This connection would only be confirmed as valid if, in the direction of $\mathbf{n}_{0,4}$, its midpoint $\mathbf{o}_{0,4}$ were located closer to $\mathbf{r}_0$ than the boundary plane of any other confirmed connection for subvolume 0. If not, subvolumes 0 and 4 may still share a boundary, but the path would have to be curved in order to avoid crossing a third subvolume (in Fig. 4.15, subvolume 3).

All possible connections for one subvolume are checked before going to the next one. Algorithm 5 shows the pseudocode of the procedure. Examples of connections are marked in dotted dark lines in Figures 4.12, 4.13 and 4.14.

4.6 Phonon

The `Phonon` class is responsible for storing the material properties and perform calculations of energy and temperature. When a `Phonon` object is instantiated, it first loads the POSCAR and hdf5 files that contain the necessary material data [89].

Algorithm 5 Subvolume connection identification procedure for grid and Voronoi subvolumes

Generate a list of every possible *pair* of subvolumes.
for each *pair* **do**
  if *path* between reference points in *pair* crosses a boundary **then**
    Remove *pair* from list.
  end if
end for
Sort *pair* list by order of proximity (*path* length).
Initialise an empty list of $confirmed$ pairs.
for each *pair* **do**
  if there are any $confirmed$ containing subvolumes in *pair* **then**
    if midpoint of the evaluated *path* is beyond the midplane of a confirmed connection **then**
      Remove pair.
    else
      Add *pair* to $confirmed$ list.
    end if
  else
    Add *pair* to $confirmed$ list.
  end if
end for
Remove any subvolumes that are not part of any pair.

These files are generated from Phono3py calculations [36, 69]. Their declaration is done with the following parameters:

```
--mat_folder  <folder_path>
--poscar_file <poscar_file>
--hdf_file    <hdf5_file>
```

In the example case, the data in the mentioned files refer to cubic silicon, containing $31 \times 31 \times 31 - 29,791$ wavevectors and 6 branches. There are therefore $31 \times 31 \times 31 \times 6 = 178,746$ modes to be simulated in total.

The code will search for both the POSCAR and hdf5 files inside the folder passed to `--mat_folder`. The quantities loaded in the data are [69]:

- Q-points $\mathbf{q}$, an array with the reduced wavevectors from which the wavevectors $\mathbf{k} = \mathbf{B} \cdot \mathbf{q}$ can be calculated, with $\mathbf{B}$ being the matrix containing basis vectors of the reciprocal lattice as columns, retrieved from the POSCAR file;
- Ordinal frequencies $f_{\mathbf{k}j}$, from which angular frequencies $\omega_{\mathbf{k}j} = 2\pi f_{\mathbf{k}j}$ are calculated;
- Group velocities $\mathbf{v}_{\mathbf{k}j}$;
- Imaginary part of self energy $\Gamma_{\mathbf{k}j}(T)$, from which relaxation times are calculated as:

$$\tau_{\mathbf{k}j}(T) = \frac{1}{4\pi\Gamma_{\mathbf{k}j}(T)} \tag{4.1}$$

- An array of temperatures at which the relaxation times were calculated.

- Optionally, if the material is an alloy, the values of $\Gamma_{\mathbf{k}j,iso}$ for the isotope scattering, so that $\Gamma_{\mathbf{k}j,total}(T) = \Gamma_{\mathbf{k}j}(T) + \Gamma_{\mathbf{k}j,iso}$.

In order to reduce file sizes, those data are only stored in the hdf5 file for the irreducible Brillouin zone. These quantities can be expanded to the full Brillouin zone, by applying the symmetry operations that reconstruct the FBZ.

With the FBZ completely reconstructed, every wavevector is checked to ensure its absolute value is the smallest possible. If not, $\mathbf{k}$ is translated by integer multiples of the reciprocal lattice vectors until it is as close to the origin as possible. This influences the calculations of specularity p, which depends on the absolute value of $\mathbf{k}$. Since the dispersion relation is periodic, the values of $\omega_{\mathbf{k}j}$, $\mathbf{v}_{\mathbf{k}j}$ and $\tau_{\mathbf{k}j}(T)$ are the same. The norms $\|\mathbf{k}\|$ are also pre-calculated and stored for later use.

The relaxation times $\tau_{\mathbf{k}j}(T)$ are calculated (Eq. 4.1), and are linearly interpolated so that $\tau = f(q, j, T)$. The interpolator is defined with SciPy's `LinearNDInterpolator` class [90]. The interpolated function therefore takes as input the wavevector index q, the branch index j and the temperature T, finding the appropriate value of τ from the vector (q, j, T). Since q and j are always integers, they will always fall exactly on an interpolation point in these two dimensions. On T axis, however, the value will be interpolated to better approximate the temperature the particle is submitted to, between the calculated temperatures.

Lastly, the temperature function is then interpolated as $T = f(e)$. For this, the energy density of the crystal is calculated with Eq. 2.88 for several temperatures inside the interval of the material data, every 0.1 K, obtaining a good enough discretisation for the interpolation function. This interpolation is defined by using SciPy's `interp1d` class [91]. This will be used to estimate the local temperature in the simulation.

Since modes have directional properties ($\mathbf{v}_{\mathbf{k}j}$ and $\mathbf{k}$), the `--mat_rotation` parameter is available if the user wants to simulate cases with different orientations of the crystal. The values are declared in the same manner as for `--geo_rotation`.

4.7 Population

The `Population` class is where the actual simulation takes place. It takes as arguments the simulation parameters, the `Geometry` object and the `Phonon` object. The relevant parameters in the example problem are the following:

```
--temp_dist    linear
--temp_interp  linear
--particles    total 1e6
--timestep     1
--iterations   10000
--conv_crit    0.01 5
--n_mean       100
--max_sim_time 0-03:00:00
```

4.7.1 Initialisation

First all the relevant simulation parameters are stored as attributes: number of particles, number of subvolumes, temperature distribution, etc. Some of the attributes of `Geometry` are also saved as `Population` attributes so the `Geometry` object as a whole does not have to be unnecessarily passed as argument to its methods. These are mainly attributes relating to boundary conditions, since the interaction between the particles and the geometry is handled by `Population`.

The first methods called by the constructor are the ones relating to reflections on rough walls. The `Population.calculate_fbz_specularity` method calculates the specularity p for each mode in every rough wall (Eq. 2.106). Then another method, `Population.find_specular_correspondences` checks if the conditions for specular reflections (Equation 2.108) are met for every mode in every rough facet. If they are not, p is set to zero. The valid specular relations between incoming modes (q, j) and outgoing modes (q', j') are then used to generate a nearest value interpolation function, $(q', j') = f(\mathbf{n}, q, j)$. The use of this interpolator accelerates the simulation, avoiding specular reflection calculations to be performed every time a specular event takes place. The third method is `Population.diffuse_scat_probability`, which calculates the probabilities of modes being chosen as a result of a diffuse reflection (Eq. 3.46), and generates the other arrays that will be used to draw the new mode from.

Next, the reservoirs are initialised as follows, if there are any. The rate of injection of particles is calculated for every mode in each reservoir (Eq. 3.36). Energy exchange through the facets of the reservoirs are stored in arrays, which are initialised as zeroes. Moreover, an "injection counter" for each mode is randomly initialised between 0 and 1. As discussed before in Sec. 3.2.4, if the calculated rate of injection for a given mode is, for instance, 2.3 particles/timestep, it means that every timestep 2 particles of that mode are necessarily injected, and a third has 30% of chance of being added. During the simulation, this counter helps to takes care of this fractional part. Every timestep the fractional part (0.3 in this case) is added to the counter. When the counter goes above 1, an extra particle is added besides the integer part (2, in the given example). The counter is then reduced by 1, to keep it always below the unit.

Then the particles are generated inside the domain. They are uniformly distributed in the geometry (Eq. 3.25,3.28 and 3.29), and modes are randomly assigned to each of them. The probability of picking any given mode is the same, ensuring that there is on average the same number of particles representing each mode. The number of particles can be declared in the parameter `--particles` together with one of the following keywords:

- `--particles total <value>`: the total number of particles in the domain;
- `--particles pmps <value>`: acronym for "per mode, per subvolume". The total number of particles would be then $N_p =$ `<value>` $\times N_s 3 N_a N$. It gives a notion of how good the representation of all modes is. The number of particles in each subvolume still varies with their volume (larger subvolumes have more particles);
- `--particles pv <value>`: number of particles per unit volume. It can be used to ensure the same density of particles when simulating several cases

with different domain sizes (such as films with different thicknesses). Since the volume is calculated in cubic angstrom ($Å^3$), usually this number is very small. For example, a 5 µm long wire with a square cross section of 100 nm sides has a total volume of $V = 5 \times 10^{10}$ $Å^3$. If $N_p = 10^6$ particles are simulated, `<value>` will be $N_p/V = 2 \times 10^{-5}$.

The three options to declare the number of particles (`total`, `pmps`, `pv`) are equivalent. They do not influence at all the calculation, being simply different alternatives for the users to choose as they prefer. It can be said that:

$$\texttt{<total>} = (\texttt{<pv>})V = (\texttt{<pmps>})(3N_aN)N_s \tag{4.2}$$

where `<total>`, `<pv>` and `<pmps>` are the `<values>` given in each option.

The user can set the initial temperature distribution within the studied structure with the parameter `--temp_dist`, declaring it in the following ways:

- `--temp_dist cold`: the lowest imposed temperature on the reservoirs;
- `--temp_dist hot`: the highest imposed temperature on the reservoirs;
- `--temp_dist mean`: the mean of all reservoir temperatures;
- `--temp_dist linear`: a linear interpolation of the temperatures on the reservoirs;
- `--temp_dist random`: random temperature values, between the highest and lowest imposed temperatures;
- `--temp_dist custom`: values given by the user.

All particles, the phonon carriers, are initialised with their Bose-Einstein occupation (Eq. 2.87) at the temperature of the subvolume. If `--temp_dist custom` is used, the values of T to be applied have to be passed to the parameter `--subvol_temp` (for "subvolumes' temperatures"), with the order of subvolume index. Subvolumes are ordered according to the coordinates of their reference points, first in x, then in y and lastly in z.

Alternatively, the user can use the final state of a previous simulation as the initial state of another, by using the parameter `--part_dist` (for "particle distribution"). One of the output files that Nano-κ saves is a text file containing the position, mode and occupation of every particle in the system, called `particle_data.txt`. If one version of this file is passed to `--part_dist`, the particles of the new simulation will be generated exactly as described, and the temperature of each subvolume will be estimated from their energy. When using this feature, it is important to be sure that the number of particles set in `--particles` is the same for the previous and for the new calculation, otherwise the density of particles in the domain will change.

After the domain is populated by particles, the path they will drift on is traced until their first collision with the mesh is identified (using Alg. 3). The facet on which they will collide and the number of timesteps until collision N_c is saved. These collision trackers avoid the necessity of calculating the next collision every timestep. Since the paths of the particles are linear, the collision position can be calculated

in advance, as well as the time it will take for the collision to happen. It becomes then just a matter of tracking the event with the counter N_c, by subtracting 1 every iteration, until it reaches 0.

Finally the convergence files are initialised and an instance of `Visualisation` class is created to handle most of the plots in the simulation.

4.7.2 The `run_timestep` method

After the initialisation, the simulation is run in a loop by repeatedly calling the `Population.run_timestep` method. As its name says, the method performs the calculation for one timestep (i.e. one iteration), and the loop is called until one of the stop conditions are met. These conditions are defined by the user with the following parameters:

- `--iterations <value>`: the maximum number of times (iterations) the `Population.run_timestep` method is executed;
- `--max_sim_time <D-HH:MM:SS>`: the maximum computation time. If the time is passed as zero (`0-00:00:00`), no time limit is imposed;
- `--conv_crit <value_1> <value_2>`: the convergence criterion used to check whether the system arrived to steady state. A residue ϵ is calculated every 100 iterations. The condition is met when $\epsilon <$ `<value_1>` for `<value_2>` consecutive checks.

The residue ϵ is calculated based on average values. Every 100 iterations, averages over time are calculated for subvolume temperatures T_s, heat fluxes $\mathbf{\Phi}_s$, thermal conductivity κ_s and energy exchange with reservoirs e_r. The data used to calculate the mean is the one registered on the `convergence.txt` output file. One datapoint (saved as a line in `convergence.txt`) is added every 10 iterations. The number of datapoints considered in the calculations of the average can be defined by the user with:

```
--n_mean <value>
```

With the means calculated, ϵ is calculated as:

$$\epsilon = \max\left(\left|\frac{\mu_i^k - \mu_i^{k-100}}{\mu_i^{k-100}}\right|\right) \tag{4.3}$$

where μ_i refers to any mean quantity of those previously referred (T_s, $\mathbf{\Phi}_s$, κ_s and E_r) and k is the current iteration. Since the simulation is random in nature, the minimum possible residue that can be achieved is dependent on simulation parameters. Number of particles, number of subvolumes and material properties affect ϵ directly, and should be taken into consideration by the user.

The number of iterations and the size of the timestep (in picoseconds) are declared as:

```
--iterations <value>
--timestep   <value>
```

In the example, the maximum number of iterations is 10000, with 1 ps timesteps. The maximum simulation time is 3 hours[1], and the convergence criterion is 1% maximum variation in 5 consecutive checks. The means are calculated over the last 100 datapoints, or in nanoseconds, over

$$100\frac{\text{datapoints}}{\text{check}} \times 10\frac{\text{iterations}}{\text{datapoint}} \times 1\frac{\text{ps}}{\text{iteration}} = 1\frac{\text{ns}}{\text{check}}$$

Saving information

Inside `Population.run_timestep`, the first operations are to write convergence information, update plots and check whether there are no particles leaking from the domain (which can happen sometimes due to small numerical errors). If untreated, these leaking particles alone can crash the calculation. In case of particle leakage, they are picked and randomly re-positioned inside the geometry. This is very rare, and by having a large enough number of particles in the domain its effect becomes negligible. These operations are done every 100 iterations.

Drifting particles

Then the particles are drifted through the domain. Their position is changed according to their velocity, and the number of timesteps to collision , `Population.n_timesteps`, is reduced by 1. Particles with corresponding `n_timesteps`≤ 0 will be treated in this iteration according to the boundary condition of the collision facet, as explained in the next steps.

Addition of particles from reservoirs

The wavepackets coming from the reservoir are then injected. In previous works [63, 65], the particles were randomly created inside an external volume that represented the reservoir. They were then drifted and some would go inside the domain. Inevitably, some of the particles would not enter the geometry, due to a combination of their position and velocity. Moreover, the generalisation for arbitrary geometries can make it difficult sometimes to define this external volume. Aiming for a more efficient way, Nano-κ uses the concept schematised in Figure 4.16.

In Nano-κ, particles are generated randomly and uniformly on the facets of each reservoir (yellow circles in Fig. 4.16). The injection counter is updated and extra particles are generated if necessary. In order to account for different initial positions at the start of the timestep (shown in Fig. 4.16 as dotted circles), the particles have a random portion of Δt assigned to them. For example, one may have drifted $0.8\Delta t$ while still inside the reservoir and the remaining $0.2\Delta t$ inside the domain. Their drift is calculated only for the portion inside the geometry, beginning at the randomly sampled point on the facet. This system gives the same result as the simple cases traditionally simulated (nanowires and thin films) and allows for flexibility in non-standard geometries.

From the predicted path and final position, the first collision is identified (Eqs. 3.17 and 3.18). Finally, the occupation of every injected particle is calculated from

[1]To offer a notion of time performance: the computation of the example case was executed in a Dell Inspiron 15-7580 laptop computer, with 16GB of RAM and Intel Corei7-8565U 1.80GHz CPU, running Windows 11 operating system.

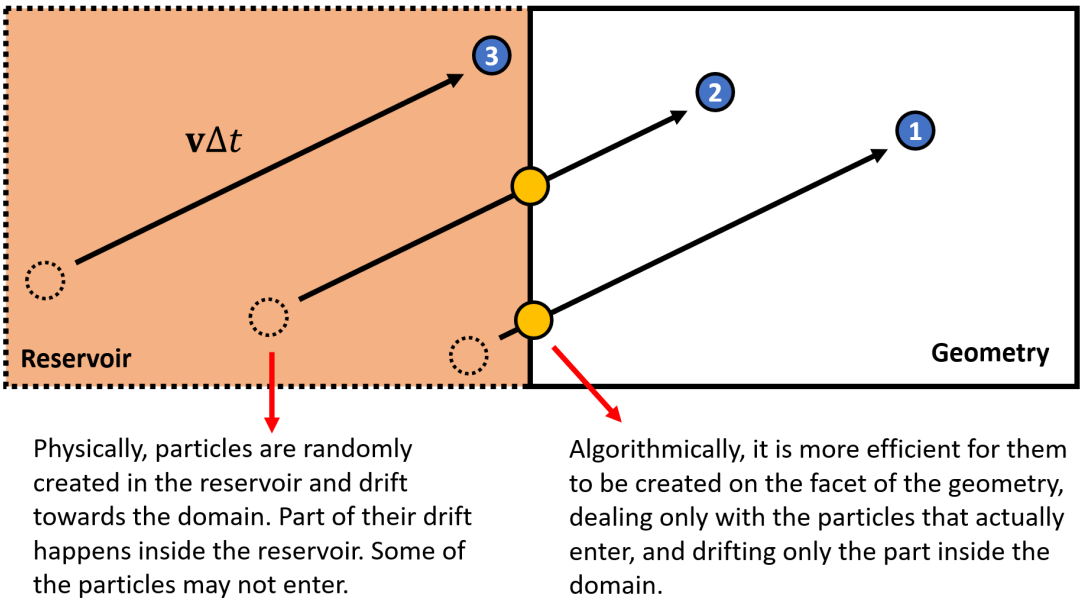


Figure 4.16: Schematics of the injection of particles from the reservoirs into the domain.

the Bose-Einstein distribution (Eq. 2.87) at the reservoir's temperature, and their information is added to the general population.

Boundary scattering

The next step is the boundary scattering. The particles that are colliding with a boundary are identified when the counter of timesteps to collision goes bellow zero. The boundary condition on the collision facet is identified, and their behaviour is treated accordingly:

- First, the particles that are colliding with a reservoir are deleted from the population. Being "deleted" here means that their properties are removed from all arrays (position, mode properties, collision trackers, occupation, energies and temperatures). The energy of the particles that are leaving to the reservoir is subtracted from the energy balance on the respective facet.
- Second, the particles hitting faces with periodic boundary conditions (if any) are treated. Their displacement has to be first reset to their collision position and then translated to the connected facet. There, the particle drifts during the remaining timestep interval. Then, the next boundary collision is detected and a new counter of timesteps to the next collision is set. All of their properties (frequency, velocity, occupation) are kept the same.
- Lastly, the reflections on rough walls (if any) are treated. Just like in the periodic case, the particles are brought back to their collision position on the facet. Then the specularity parameter of each of these reflections is read. A random number γ between 0 and 1 for each reflection and compared to p.
  - If $\gamma < p$, the reflection is specular, and the specular reflection interpolator function is used to find the new modes. Their occupations and frequencies

are kept the same (hence their energy), with their velocities changed to that of their new modes.

- If $\gamma \geq p$, the reflection is diffuse. In this case, the probabilities calculated at initialisation with Eq. 3.46 for that facet are used to draw a new mode. The frequencies and velocities of each particle are changed to those of the new modes. The temperature at that position is estimated from the current T distribution, and the occupations of the particles are changed to the Bose-Einstein occupation of their modes (Eq. 2.87).

- Finally the particles drift the remaining time of the timestep after collision, and update their counter of timesteps to the next collision.

These three procedures are enclosed in a loop, which runs until all particles have a number of timesteps to collision larger than zero. Some particles can be scattered several times in a single timestep, depending on the conditions of the boundary scattering (collision position, velocity of the new mode, geometry, etc.). Figure 4.17 shows a flowchart of the algorithm, considering the relevant parts of the main loop and the boundary scattering loop.

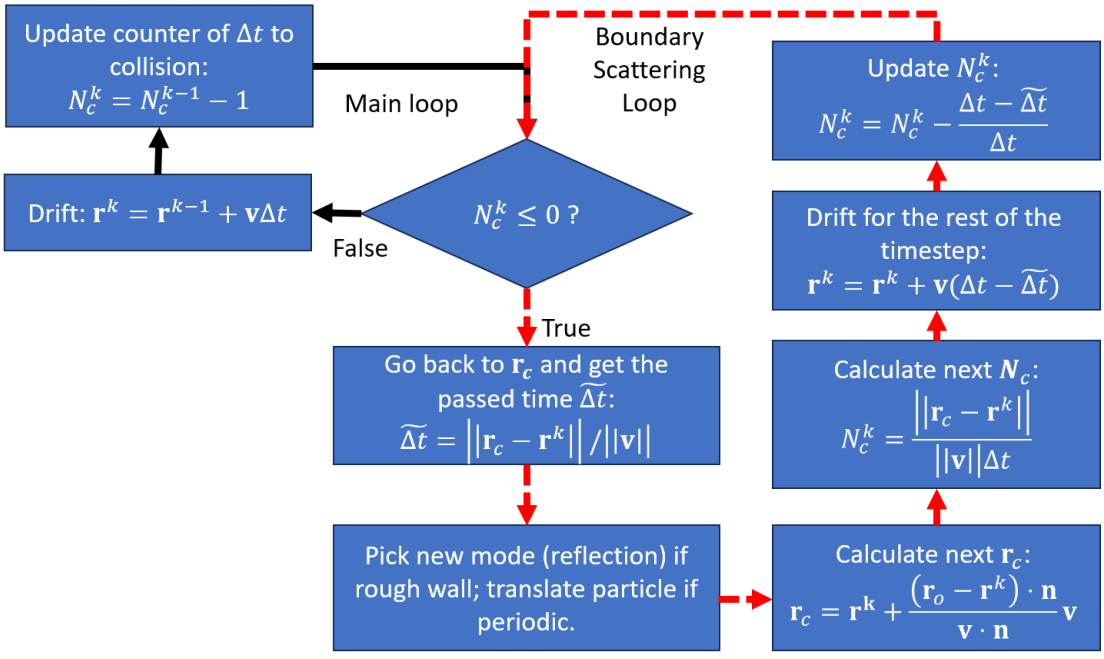


Figure 4.17: Algorithm of particle drift and boundary scattering.

Energy density and temperature

After boundary scattering, the temperatures T_s in each subvolume s are estimated. For this, the energy density e_s is calculated for each subvolume considering the particles inside it (Eq. 3.49). They are then converted to temperature by the Phonon object, using the inverse interpolator function $T^* = f(e)$ defined at initialisation, such that $T_s = f(e_s)$.

Aiming for a variance reduction in the calculation, previous works [63–65] used a fixed reference temperature T_{ref} for all particles, usually set as just a little lower

than that of the coldest reservoir. In comparison to calculations when $T_{ref} = 0\,\mathrm{K}$, this technique greatly improves the results through the reduction of uncertainties. Nevertheless, there is still room for improvement: the cold regions of the geometry will have smaller variance than hot regions, since their local temperatures are closer to T_{ref}. Nano-κ improves this technique by having a reference temperature *distribution* over the domain. The very temperature T_s calculated in each subvolume s can be used as a local reference. For example, if in the next iteration the particles in s carry more phonons than previously, the deviational energy density δe will be positive, and T_s will increase. In steady state, δe will tend to zero, keeping T_s unchanged. This reduces even more the need for a very large number of particles, consequently improving computational performance.

Phonon-phonon scattering

In order to calculate the relaxation time of each particle, their individual temperatures have to be also assigned. These temperatures are considered as the same temperature as the one of the subvolume that contains the particles, or be interpolated between subvolumes. This is chosen by the user, and declared with the help of the `--temp_interp` parameter as:

- `--temp_interp nearest` to take the value of the subvolume;
- `--temp_interp linear` to interpolate the value between subvolumes (currently only available for slice subvolumes).

With this temperature estimation, the relaxation time $\tau_{\mathbf{k}j}(T)$ of each particle can be found with the relaxation time interpolator $\tau = f(q, j, T)$, also defined in `Phonon`. With τ, the change in occupation due to the phonon-phonon scattering can be calculated.

Conclusion

With this, the iteration is concluded. Information are recorded on files if necessary. Particles drift one timestep again, and new ones are injected by the reservoirs. Those that encounter a boundary are reflected or absorbed, temperatures are updated and phonon-phonon scattering takes place. Figure 4.18 shows a flowchart of the general algorithm with the main operations.

In order to store the results the `Population` class creates several text files:

- `arguments.txt`: this file registers all the parameters used in the simulation. An equal simulation can be run by passing this file as input;
- `convergence.txt`: registers the evolution of the simulation over time. Each column is a different quantity, identified on the first line of the table. Each line contains the data for a different timestep. One line is added every 10 timesteps;
- `particle_data.txt`: the position, mode, and occupation of every particle in the simulation. Each line contains information of a single particle. This is saved every 100 timesteps;

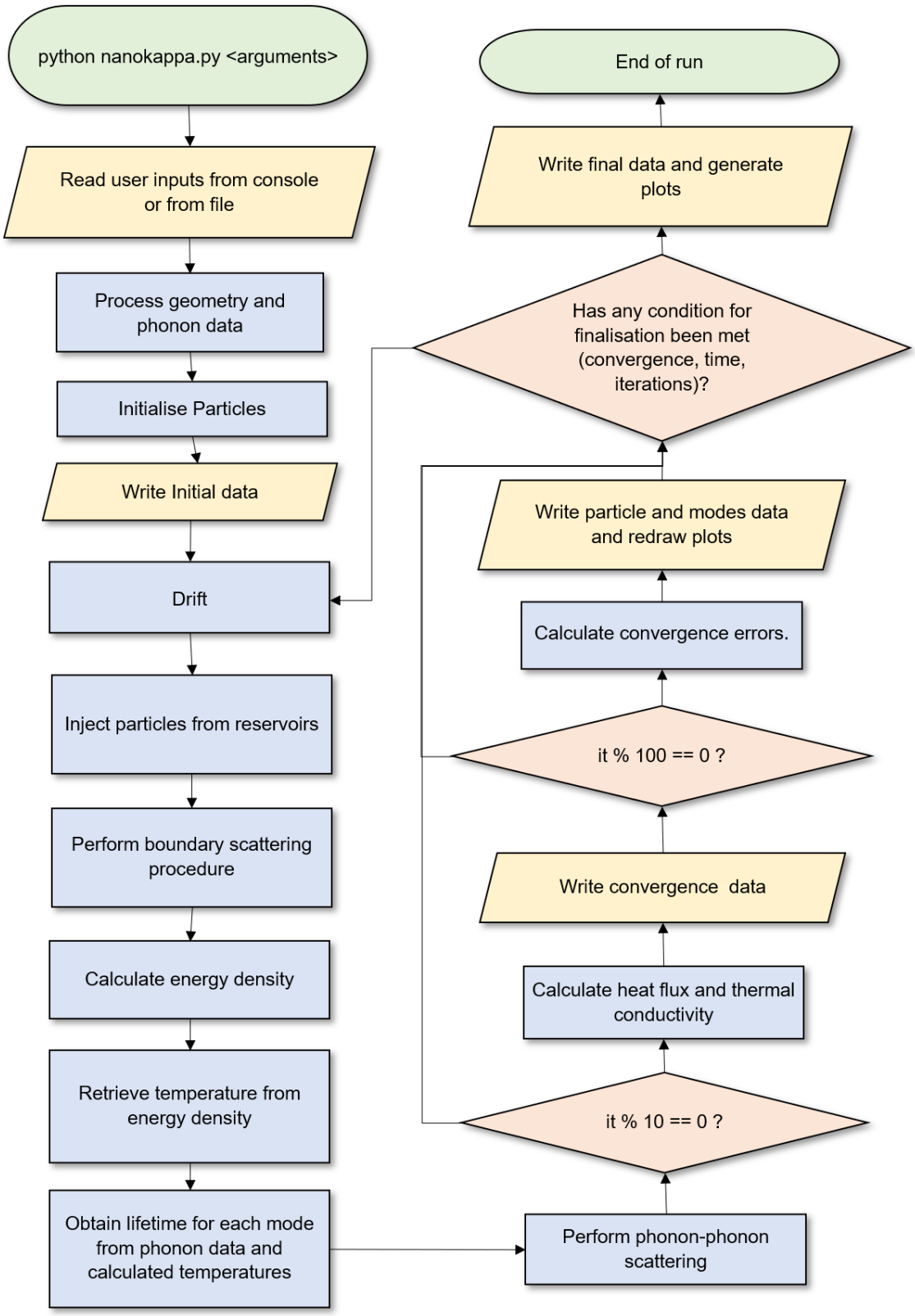


Figure 4.18: Flowchart showing the main operations performed in Nano-κ.

- `subvolumes.txt`: the data regarding each subvolume. It contains index, position, volume, temperature (mean μ and standard deviation σ), heat flux (μ

and σ) and thermal conductivity (μ and σ, for sliced geometries only);

- `specular_correspondences.txt`: saves the correspondences between incoming and outgoing modes for specular reflections for every rough wall in the geometry. The first three columns is the normal of the facet. The next two columns describe the incoming mode $\mathbf{k}j$ with the indices of $\mathbf{k}$ and j in the material data. The last two columns do the same for the outgoing mode $\mathbf{k}'j'$;
- `residue.txt`: residues are the values calculated to detect if the system reached steady state. They are checked every 100 iterations. Their values are stored in this file, where each column is a different quantity (subvolumes' temperatures, subvolumes' heat fluxes, reservoirs' energy balance and subvolumes' thermal conductivity);
- `subvol_connections.txt`: when the geometry is not sliced, the thermal conductivity is calculated for each connection between subvolumes. This file stores information on the connections: subvolumes indices, the components of the connection vector $\mathbf{n}_c = \mathbf{r}_{s2} - \mathbf{r}_{s1}$, ΔT (μ and σ), $\mathbf{\Phi} \cdot \mathbf{n}_c$ (μ and σ) and κ_c (μ and σ).

4.8 Visualisation

One type of output that Nano-κ produces are plots. Most of the plots of the particles are drawn by the `Visualisation` class. An instance of this class is created inside `Population` and receives information from it so it can produce the graphics.

The constructor of the `Visualisation` class receives as inputs the simulation parameters, a `Geometry` object and a `Phonon` object. Optionally, it can receive a `Population` object, avoiding reading information from a txt file, but directly from the calculation. They are converted as attributes to be easily accessible, and the standard filenames are also saved as attributes (`convergence.txt`, `particle_data.txt`, `modes_data.txt` and `subvol_data.txt`). The available styles are also initialised, changing the color palette with which the plots are generated. Currently, three themes are available (`white`, `light` and `dark`), and can be chosen by declaring:

```
--theme <name>
```

The plots that will be used as example and interpreted hereafter are the ones generated with the example parameters.

4.8.1 Convergence plots

The most important plots are the convergence plots. They show the evolution over time of several quantities, as well as their mean and standard deviation values calculated with the number of datapoints given in `--n_mean`.

Number of particles

One of the first plots that can be inspected to check the quality of the simulation is the one that shows the number of particles over time. Ideally, the number of particles in each subvolume N_{ps} should be kept the same. It is inevitable in a Monte

Carlo simulation however that some variation appears. As long as N_{ps} is stable and more or less proportional to the volume of the subvolume, the simulation is correct. Figure 4.19 shows the plot for the example case.

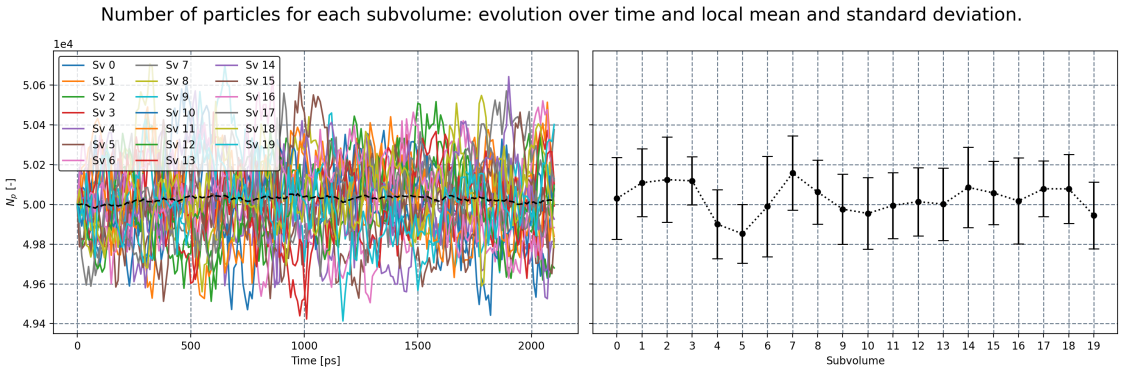


Figure 4.19: Convergence plot of number of particles in subvolume for the example case.

Presently, the choices of geometry and subvolume distribution (i.e. thin film with slice partition) causes all subvolumes to have the same volume. Consequently, every subvolume will have more or less the same N_{ps}. Indeed, in the left plot it can be seen that all lines are mixed up together, with their mean around 5×10^4 particles throughout the simulation. Moreover, the profile in the plot on the right shows a pretty even distribution throughout the whole domain.

Energy balance

Other convergence plots that can be inspected to check the quality of the simulation are the energy and flux balance with the reservoirs, E_r and $\mathbf{\Phi}_r$. The energy E_r and the flux $\mathbf{\Phi}_{res}$ are estimations calculated from the N_{pr} particles that cross the boundaries with the thermostats in 10 timesteps (it means in between two datapoints):

$$E_r = \frac{3N_a N}{N_{pr}} \sum_{i=1}^{N_{pr}} \hbar\omega_i [n_i - n_i^0(T_r)] \ \mathrm{sign}(-\mathbf{v}_i \cdot \mathbf{n}) \tag{4.4}$$

and

$$\mathbf{\Phi}_r = \frac{1}{NV_0} \frac{3N_a N}{N_{pr}} \sum_{i=1}^{N_{pr}} \hbar\omega_i [n_i - n_i^0(T_r)] \mathbf{v}_i \tag{4.5}$$

Figure 4.20 shows those quantities, for the example case.

The first obvious observation is that the energy balance plot on the left is symmetric. This is a direct consequence of the problem configuration: both reservoirs have the same area, and the temperature distribution is also symmetric. Because of this, the energy that enters on one side is basically the same value that leaves on the other side at all times. This can be confirmed by the dark line showing the balance always near 0 eV. If the initial temperature distribution was different (`--temp_dist cold` for example), it would not be true, but the balance would still tend to zero as the system approached steady state.

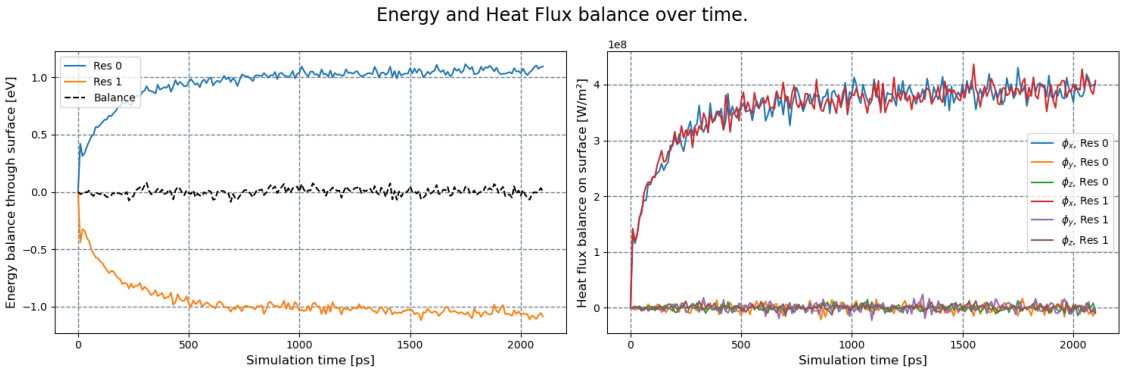


Figure 4.20: Convergence plot of energy balance and heat flux in the reservoirs for the example case.

On the right, the heat flux Φ_x in each reservoir rises together. Of course, the reasons for this are the same as before, with the difference that heat flux is directional. Since heat enters on one side and leaves on the other extremity, the vector points in the same direction, and hence the components of both are the same. The Φ_y and Φ_z components are always near zero, showing that the sampling of modes coming from the reservoirs as well as the modes that come from the geometry are evenly represented at all times.

Temperature and energy density

The distribution of temperature and energy density can be analysed together, since the relation between them is almost linear for fairly high temperatures [2]. Figures 4.21 and 4.22 show the similarities between both.

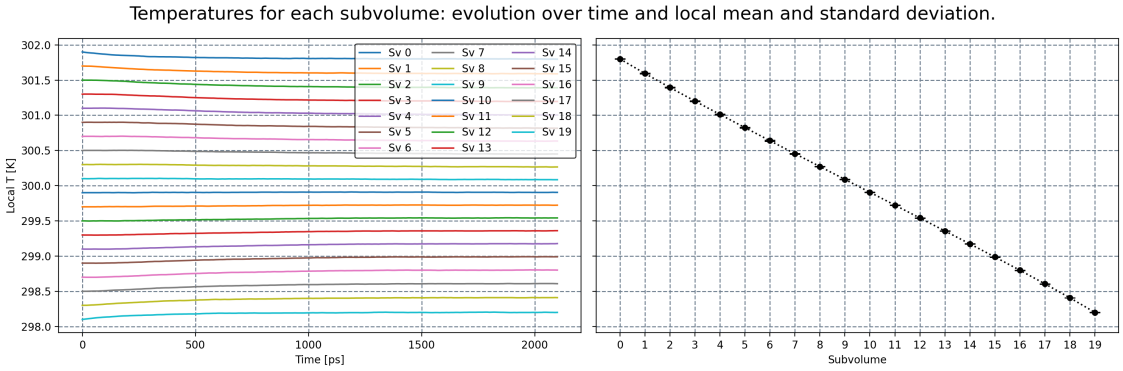


Figure 4.21: Convergence plot of subvolume temperature for the example case.

The initial distribution of temperature is linear, equivalent to what would be expected by solving the heat equation in macroscale. This is called the "Fourier regime". As time passes, the temperature on both extremities move away from their initial temperature. This deviation can also be seen in a slight bend of the temperature profile at the extremities, in the plot on the right. As it will be seen

[2]The term $n^0_{\mathbf{k}j} + (1/2)$ in Equation 2.88 tends to $y = k_b T/(\hbar\omega)$ as T increases. Therefore, the energy of each mode tends to $k_b T$ for any frequency, and the energy density becomes $e = (3N_a/V_0)k_b T$.

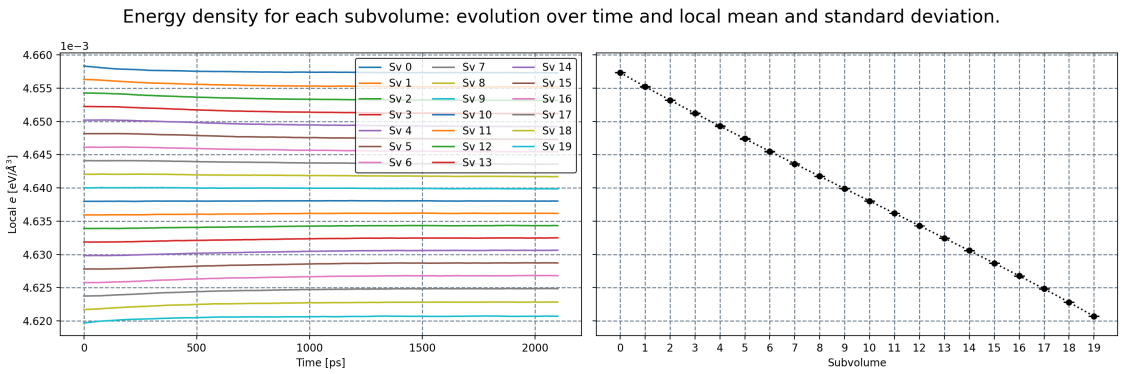


Figure 4.22: Convergence plot of subvolume energy density for the example case.

in the result section, it is an effect of the ballistic behaviour of phonons with long relaxation times. This bending of the temperature profile is more apparent in settings with low roughness, small length scales and lower temperatures.

Subvolume heat flux

The evolution of heat flux in each subvolume over time is shown in Fig. 4.23:

First of all, it can be seen the stability of the y and z heat flux through the simulation. The main component of the heat flux, Φ_x, is shown to have its mean value rapidly increasing and stabilising near its final value. The profile, however changes with time. Initially, the heat flux is larger at the centr of the domain. As time passes, the temperature profile is bent by ballistic effects, and the heat flux on the extremities increases. The end result is a heat flux profile that is not constant, but bulges at the center. This results from the redistribution of energy from phonon-phonon scattering, which mathematically is a source term in the Boltzmann Transport Equation of each mode. The closer a system is to the Fourier regime, the more uniform is the heat flux throughout the domain.

Thermal conductivity

Despite the curves of temperature profile and heat flux being related, their relation is not direct. This can be seen by analysing the thermal conductivity in each subvolume, κ_s. Figure 4.24 shows an evolution for κ_s over time and the total thermal conductivity in the domain:

The figure shows that the thermal conductivity is very stable through the simulation, both in total and local values. The profile is very constant in the middle, where T is closer to linear and $\mathbf{\Phi}$ is closer to constant. As we go from the center to the borders, κ_s decreases slightly, (the small increase at both ends due to a extrapolation of the temperature gradient near the borders). This tendency of decay on both extremities is also an effect of ballisticity: the wavepackets coming from the reservoirs take some time to distribute their energy to other modes and reach the local phonon distribution. Whenever wavepackets have this excess of energy, the so called "ballistic transport" happens. In other words, κ_s is reduced at the borders because ballistic transport is *not* heat conduction, but more similar to radiative heat transfer. Therefore, while $\mathbf{\Phi}_s$ may be higher, κ_s is reduced.

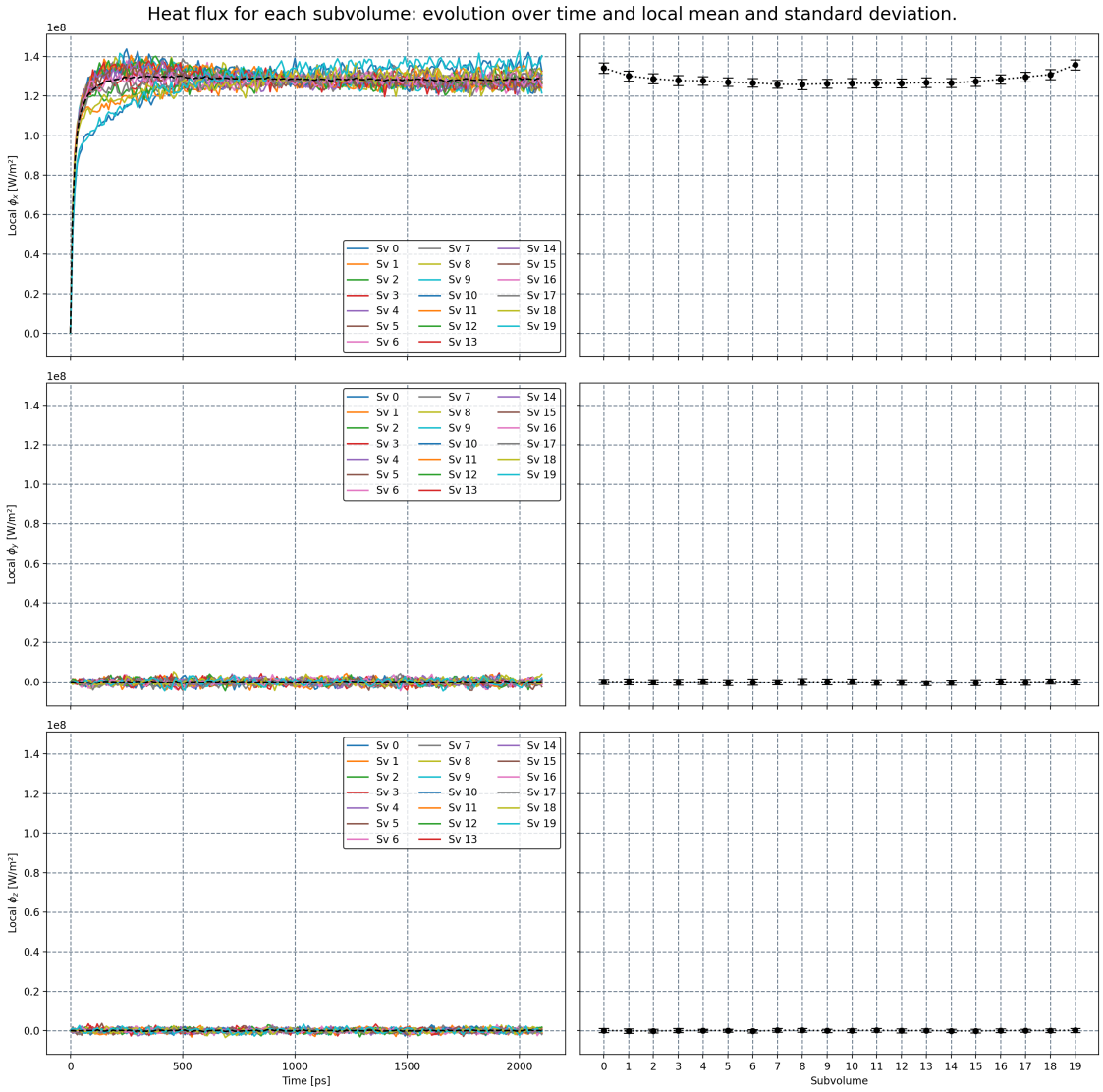


Figure 4.23: Convergence plot of subvolume heat flux in x, y and z directions for the example case.

At the bottom of this figure, the total thermal conductivity κ is shown, together with the rolling mean μ and standard deviation σ. Here, κ closely follows the behaviour of the mean Φ_x shown in Fig. 4.23. There can be seen a break on the behaviour of μ and σ around 1000 ps, when the windows inside of which the mean is calculated do not consider $\kappa(t = 0) = 0$ anymore. From there, μ closely agrees with the instantaneous κ, drastically reducing σ. At the end, κ stabilises around $62.12\,\mathrm{W/m \cdot K}$, with an standard deviation of $0.16\,\mathrm{W/m \cdot K}$ (recalling that the mean and standard deviations calculated for the last 100 datapoints, or 1 ns, as declared to the `n_mean` parameter).

Convergence analysis

One interesting thing to note is that the simulation ran for 2100 iterations, less than the maximum specified of 10000. Also it finished before the time limit. Clearly, the first condition that was met was the low residue, declared as 1% for 5 consecutive

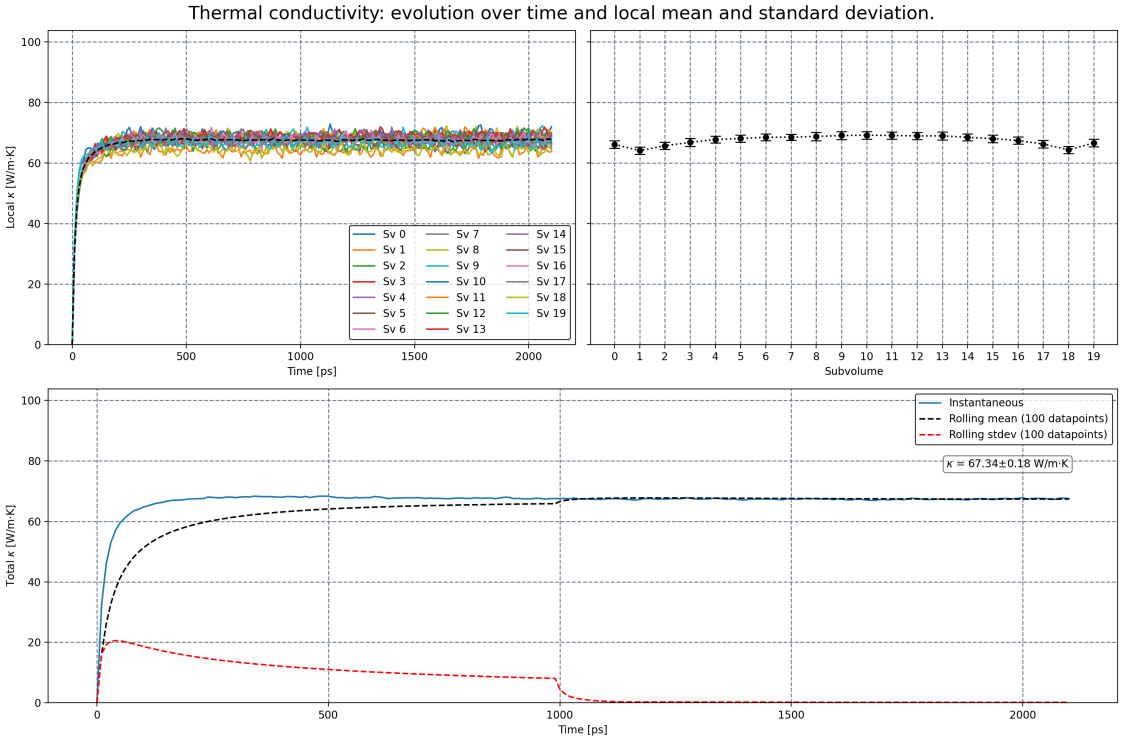


Figure 4.24: Convergence plot of subvolume thermal conductivity for the example case.

checks (a total of 500 ps), for all checked quantities. Indeed, all quantities seem to be stable at steady state or very close to approaching it. This shows an appropriate choice of number of particles, which kept the quantities fairly stable over time, with low standard deviations.

Usually, the quantities that have large variations, and consequently higher residues, are $\mathbf{\Phi}$ and the energy balance on the reservoirs. The former because the addition of velocity in the flux equation increases the noise; the latter because the calculation is done with fewer particles than over a subvolume. In general, T_s and κ_s are very stable for simple cases (films and wires). Temperature becomes unstable for small number of particles, depending on the material properties. Thermal conductivity in particular can cause problems with convergence when the temperature of adjacent subvolumes are close to each other. Small variations in these temperatures can cause large variations on κ since $\kappa \propto \Delta T^{-1}$. In this case, the convergence criterion may not be achieved, and a reasonable number of iterations becomes the dominant stop criterion to consider.

Given that, the number of iterations necessary for the system to achieve steady state varies case by case. Larger geometries need more time to stabilise the system. Velocities and relaxation times also influence the duration of the simulation. Since such a study was not performed yet due to lack of variety in *ab-initio* material data, it is in the end a decision of the user to verify these conditions for their particular problem.

4.8.2 Scatter plots

Nano-κ can also generate scatter plots to visualise the particles in the geometry, and its properties. One or more properties can be passed to the `--plot_fig` parameter:

- `sv`: plots particles colored according to their subvolume index;
- `T`: plots particles colored according to their temperature;
- `e`: plots particles colored according to their deviation of energy in relation to the mean temperature of subvolumes;
- `omega`: plots particles colored according to their angular frequency.

Subvolumes

This scatter plot can be used to visualise the distribution of subvolumes in the domain. In Figure 4.25 it can be seen that the reference points used to classify the particles in subvolumes indeed generate equally spaced slices.

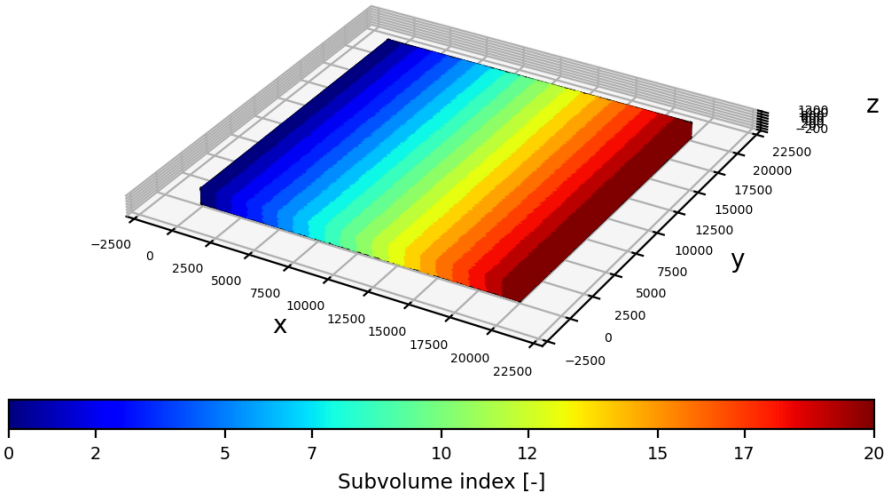


Figure 4.25: Scatter plot of particles coloured according to their subvolume index for the example case.

Temperature

Figure 4.26 shows the same scatter plot but this time coloured by particle temperature. Because of the linear interpolation specified by `--temp_interp linear`, the colours show a gradient in the x direction, and the borders between subvolumes fade.

Energy deviation

Perhaps the most important scatter plot is the energy deviation. The color in this case is defined by the equation:

$$\delta E_i = \hbar\omega[n_i - n_i^0(\overline{T_s})] = \hbar\omega_i \delta n_i \tag{4.6}$$

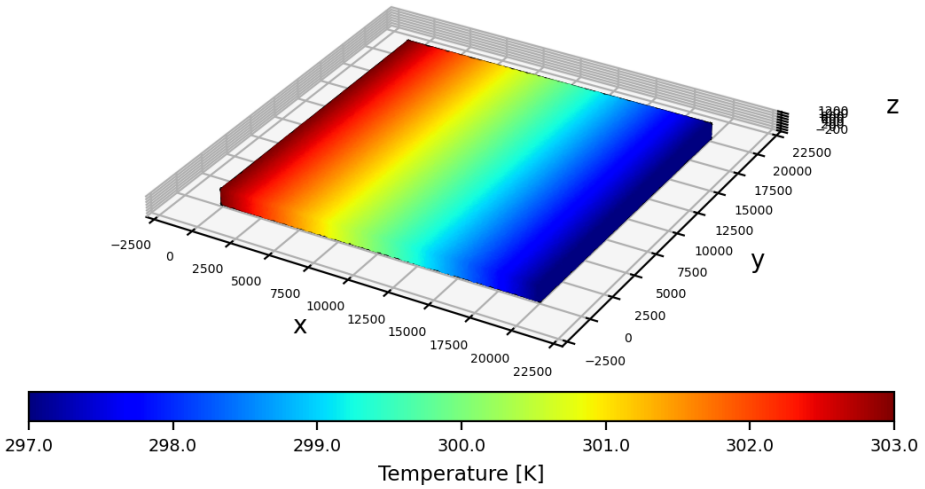


Figure 4.26: Scatter plot of particles coloured according to their temperature for the example case.

where the subindex i refers to the particle, with n_i^0 being calculated with Eq. 2.87 for the mode of the particle i. The reference temperature is a simple average of the subvolume temperatures, $\overline{T_s} = \sum T_s/N_s$. In the plot, the particles still have more or less the same colouring as in Fig. 4.26. The difference is that some particles in blue (low δn) or in red (high δn) can be seen cruising the domain far from the extremities. These particles are representing modes with long relaxation times, which are injected from the reservoirs and travel through the geometry with little influence from scattering. These are the phonons responsible for the ballistic effects previously mentioned.

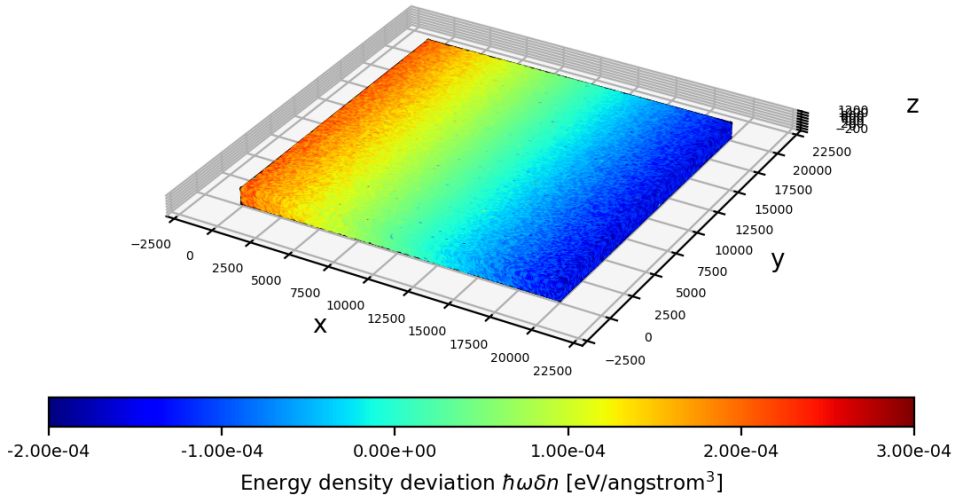


Figure 4.27: Scatter plot of particles coloured according to their energy deviation for the example case.

Angular frequency

Another scatter plot that is generated is the one coloured by angular frequency ω_i, shown in Figure 4.28. It shows an uniform distribution of frequencies in the domain,

with no clusters of particles of same ω_i. This shows the equal representation of modes everywhere in the geometry.

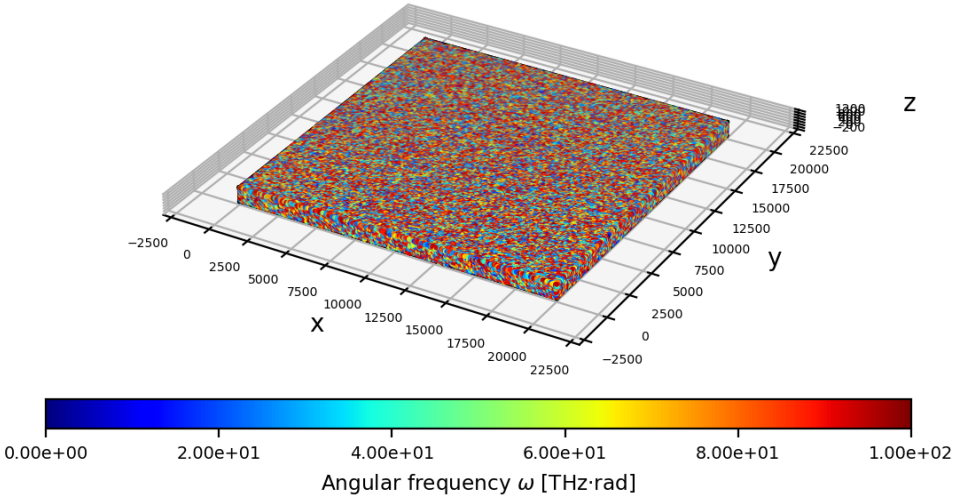


Figure 4.28: Scatter plot of particles coloured according to their angular frequency for the example case.

4.8.3 Connection thermal conductivity

Besides these, other relevant figures are also generated. They illustrate how thermal conductivity behaves in the subvolume connections.

Connection thermal conductivity

In Figure 4.29, the connections between subvolumes are plotted as lines, coloured by their value of thermal conductivity (Eq. 3.61). In this case, with the sliced geometry, it is intuitive that the κ_c data has more or less the same behaviour we have found for κ_s in Fig. 4.24. The lower thermal conductivity on the extremities, resulting from the ballistic phonons, appears as shades of cyan, while the red center shows the higher κ_s of those connections. For grid or Voronoi subvolumes, however, the convergence plot for κ_c will not be so easily read, and the geometrical visualisation of κ_c can give the user more comprehensible information.

Dependency of thermal conductivity on frequency.

One plot that may offer more insight about how the heat transfer happens is the analysis of thermal conductivity in relation to phonon frequency. Figure 4.30 shows the contribution of each frequency to the thermal conductivity for each connection. The interval $[0, \omega_{max}]$ is divided into 100 equal intervals $\Delta\omega$, and the heat flux $\Phi_x(\Delta\omega)$ considering only the particles in each frequency interval is calculated. The thermal conductivity $\kappa_c(\Delta\omega)$ is then calculated by dividing $\Phi_x(\Delta\omega)$ by the local temperature gradient.

There are two prominent peaks in the distribution, one around 25 radTHz and other around 75 radTHz. Both are located in regions where modes have longer relaxation times, as it can be seen in Figure 4.31. The first peak shows the high

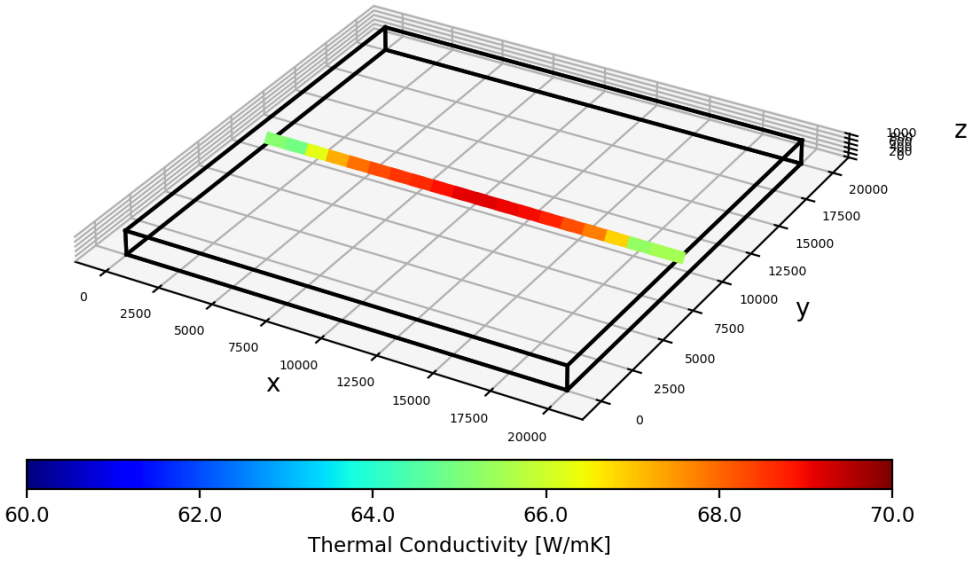


Figure 4.29: Line plot of subvolumes connections coloured by connection thermal conductivity κ_c for the example case.

contribution of acoustic modes, which have low frequencies but the highest velocities, and therefore a high heat flux. In contrast, the peak in higher frequencies is caused by optical modes, which have lower velocities, showing that the their contribution to κ is not negligible. Clearly, a combination of high velocity and long relaxation times is needed to improve thermal conductivity.

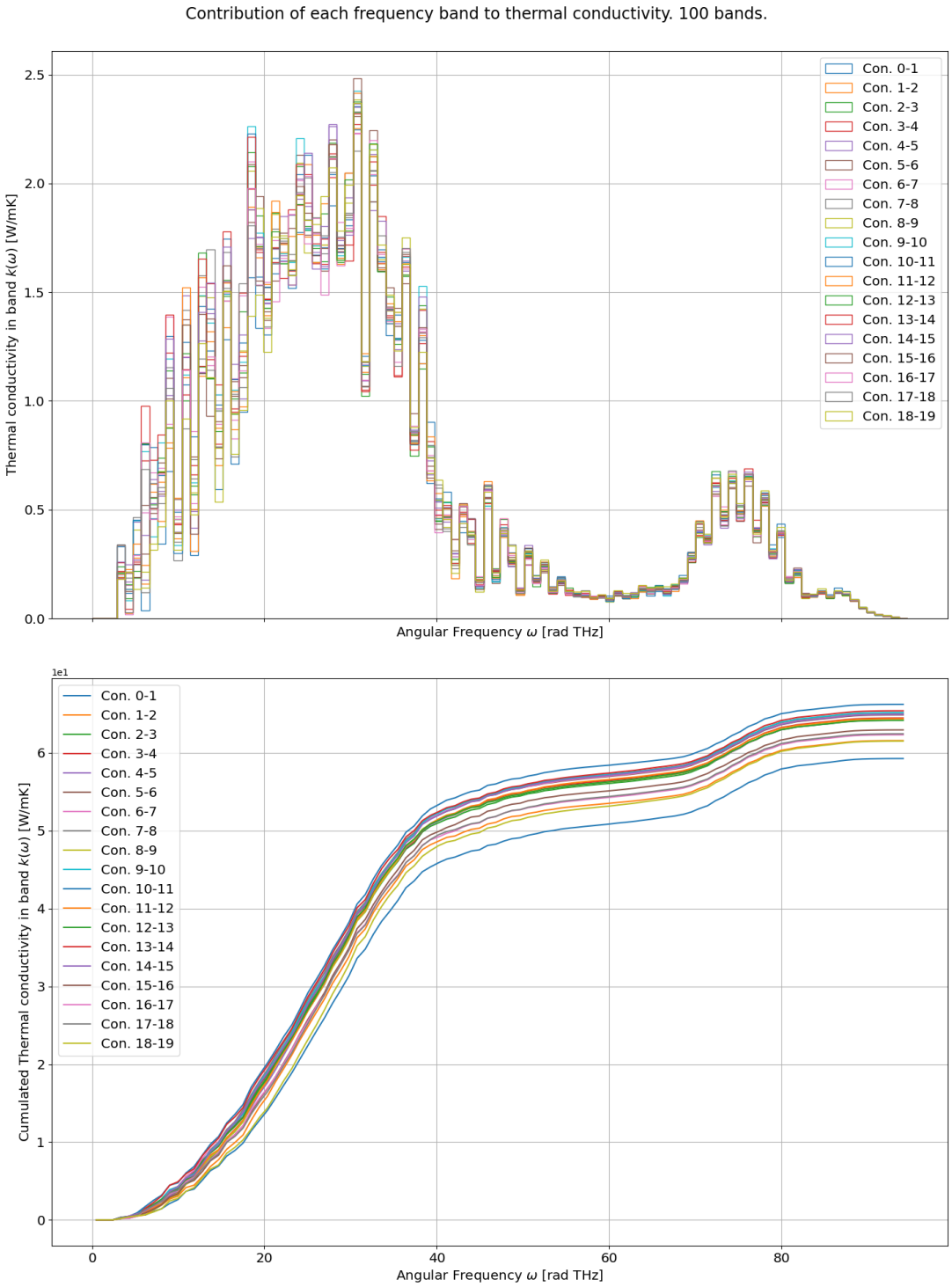


Figure 4.30: Plots showing the contribution of each frequency band $\Delta\omega$ to the connection thermal conductivity κ_c for the example case. Top: value per frequency band $\Delta\omega$; Bottom: cumulated values.

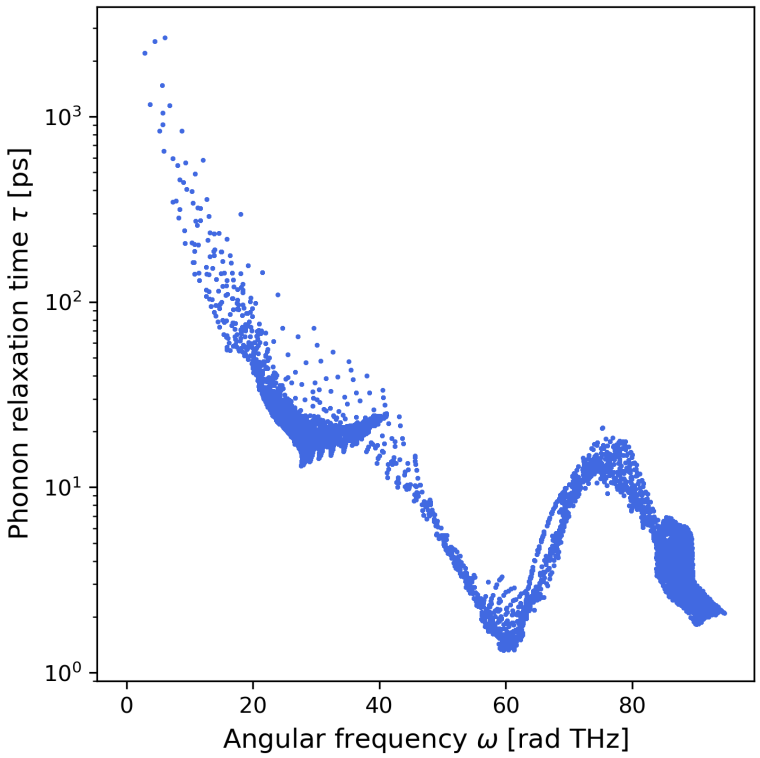


Figure 4.31: Mode relaxation time τ at $T = 300\,\mathrm{K}$ for the Silicon data used in the example simulation.

Chapter 5

Sensitivity analysis

In this chapter, a set of simple cases was run to study the sensitivity of the algorithm by varying:

- number of particles,
- the number of subvolumes,
- the timestep value,

from a reference calculation. Those directly affect the uncertainty and speed of the calculation. Indeed, the more particles a simulation uses, the more accurate is the result and the longer it takes to run the simulation. It remains a user decision to balance between expected accuracy and simulation duration.

5.1 Reference case

The chosen reference case was set as a 2 µm thick silicon thin film, submitted to a temperature gradient $\Delta T = 4\,\mathrm{K}$, where $T_{hot} = 302\,\mathrm{K}$ and $T_{cold} = 298\,\mathrm{K}$. The heat transfer happens in the cross-plane direction, in this case along x direction. The domain was sliced into 20 subvolumes and populated with 10^6 particles in total. The phonon properties computed from DFT simulations were considered in a Brillouin zone discretized into $31 \times 31 \times 31$ wavevectors. Considering all 6 branches, acoustic and optical, this totalizes 178,746 modes. The timestep was set to 1 ps, and the number of timesteps was 10,000, simulating the system for 10 ns. The initial temperature profile was set as varying linearly from T_{hot} to T_{cold}. The temperature for each particle was estimated by linear interpolation between subvolumes. No time limit or maximum error criterion was imposed. Calculation ends after the $10{,}000^{th}$ timestep.

The domain was built as a box with 20.000 Å sides, applying the temperature gradient in the x direction. Periodic boundary conditions were applied in the y and z directions. Despite it being possible to use shorter lengths in y and z directions to simulate a thin film, a larger domain decreases the number of boundary scattering calculations for the same end result, which decreases simulation time.

The parameters to reproduce this calculation are:

```
--mat_folder        D:/LEMTA/Code/Materials/Si/
--hdf_file          kappa-m313131.hdf5
```

```
--poscar_file       POSCAR
--geometry          box
--dimensions        2e4 2e4 2e4
--subvolumes        slice 20 0
--bound_pos         relative 0 0.5 0.5 1 0.5 0.5
--bound_cond        T T P
--connect_pos       relative 0.5 0 0.5 0.5 1 0.5 0.5 0.5 0 0.5 0.5 1
--bound_values      302 298
--temp_dist         linear
--temp_interp       linear
--particles         total 1e6
--timestep          1
--iterations        10000
--results_folder    test_results/ref_sensitivity
--conv_crit         0 5
--theme             white
--colormap          jet
--n_mean            20
--max_sim_time      0-00:00:00
```

Figures 5.1, 5.2 and 5.3 show respectively the temperature, heat flux and thermal conductivity evolution in each subvolume. The mean values and standard deviations were computed over the last 20 convergence data-points, or 200 ps.

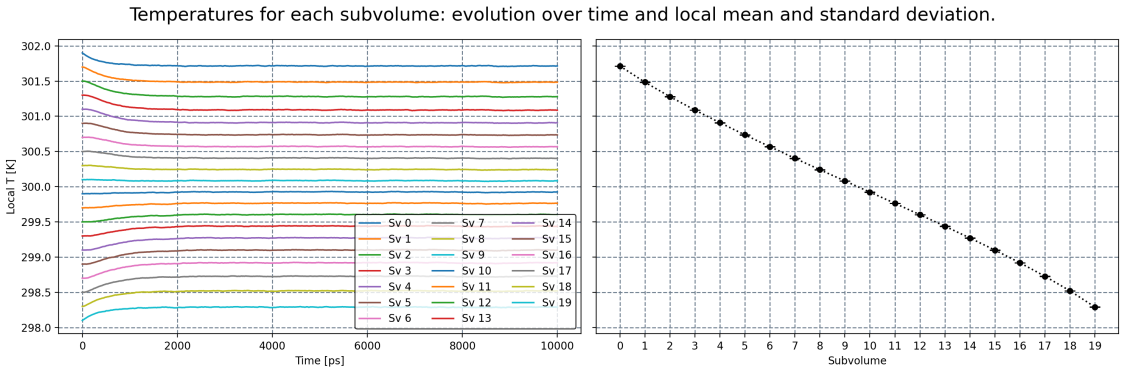


Figure 5.1: Evolution of the temperature in the reference case. Left: Time evolution of temperature in each subvolume.

Considering Fig. 5.1, just as the in-plane case of the previous chapter, the temperature profile slightly bends away from the linear profile predicted by the Fourier regime at macroscale. As already discussed, this curvature is due to the ballistic transport of the phonons coming from the reservoirs. In comparison to the previous case, the deviation of the temperature profile from Fourier's regime here is much more pronounced than in the in plane configuration. The rough walls present in the in-plane conduction study generated wavepackets with $n^0_{\mathbf{k}j}(T)$ phonons as results of diffuse reflections. These walls are not present in the cross-plane direction (here "periodic" BC are considered), allowing phonons to travel farther, increasing the curvature of the T profile. The heat flux Φ_x in Fig. 5.2 shows the same behaviour as in the in-plane case, with higher values due to the reduced thermal resistance. Moreover, the difference between values in the middle of the sample and its extremities is also much more pronounced, consequence of the difference in T profile.

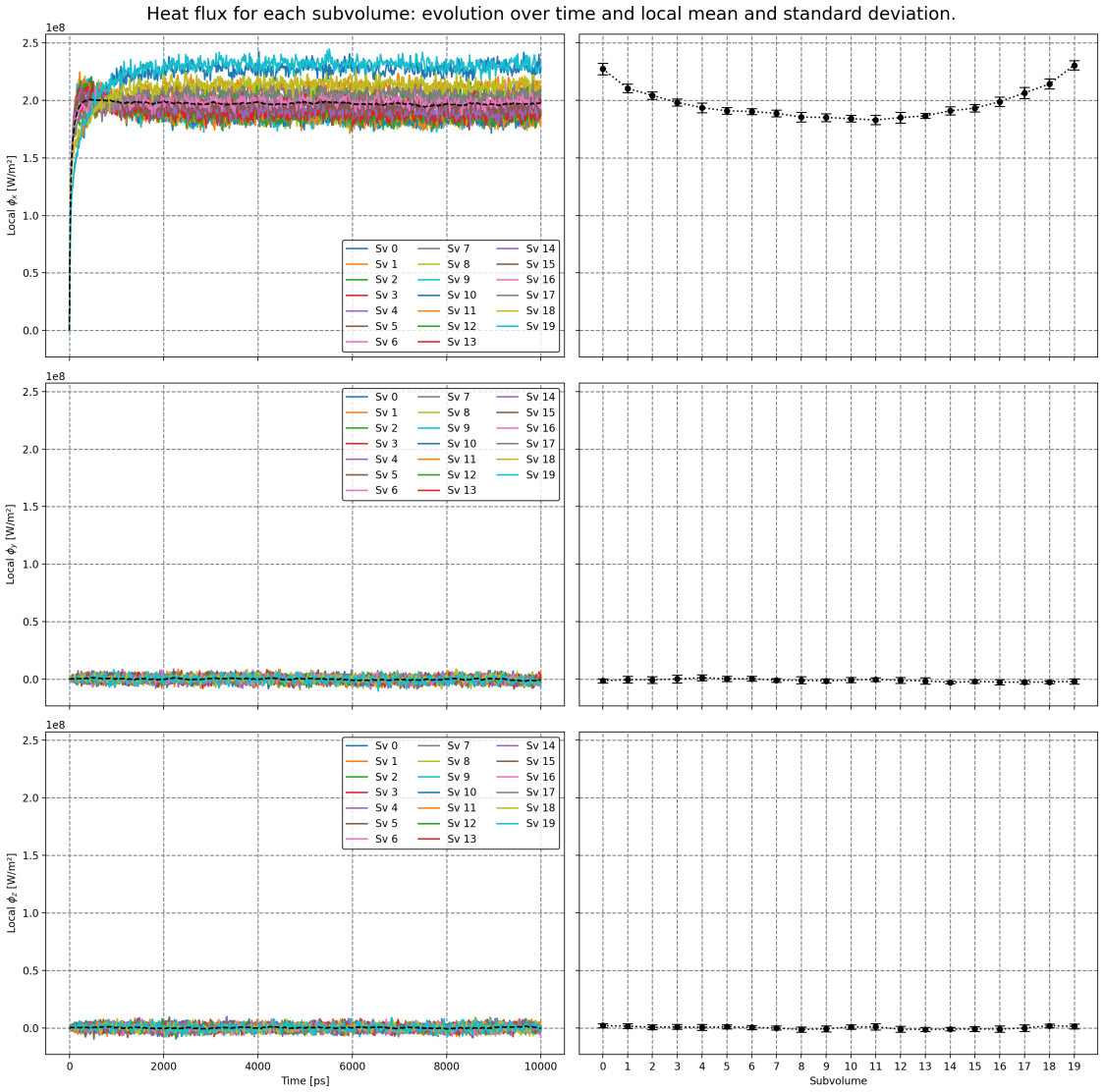


Figure 5.2: Evolution of the heat flux in x, y and z directions in the reference case. Left: Time evolution of Φ_i in each subvolume. Right: Final profile with mean and standard deviations.

Lastly, as a consequence of these differences, the thermal conductivity in Figure 5.3 is around 50% higher than in the previous case, 103.58 ± 0.18 W/m·K in total, and also with a larger difference between center and border of the sample.

5.2 Number of subvolumes

The first sensitivity study was done by changing the number of subvolumes while keeping the number of particles constant at 10^6. Because the number of subvolumes affects the smoothness of ∇T, each number of subvolumes was simulated with `linear` and `nearest` value interpolation. In the `linear` case, temperature profile in the system linearly varies from the center of a cell to the center of the following cell, allowing to perform Bose-Einstein and lifetime calculations, that both depends on T, at a temperature close to the expected one. In the `nearest` case, the tempera-

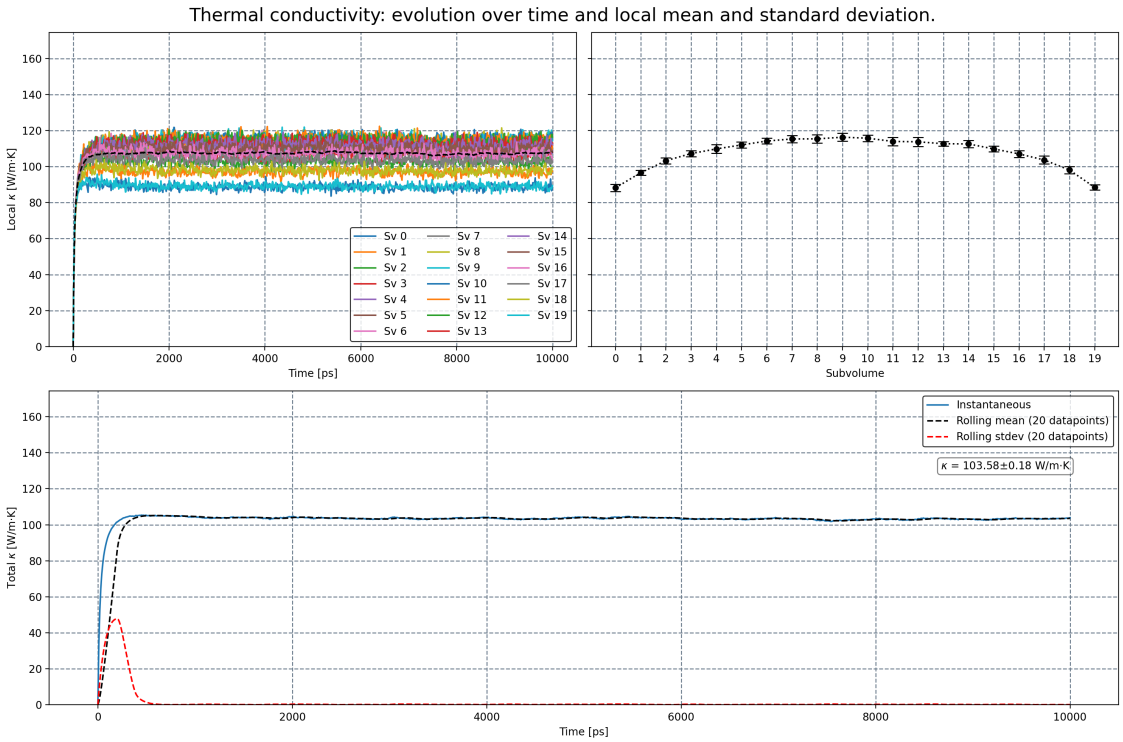


Figure 5.3: Convergence of thermal conductivity for each subvolume (top) and globally (bottom).

ture profile in the system is "step-like", keeping constant T value on the whole cell. Obviously, increasing the number of subvolumes makes the difference between linear and steps profiles vanishing. The results of system thermal conductivity are shown in Fig. 5.4.

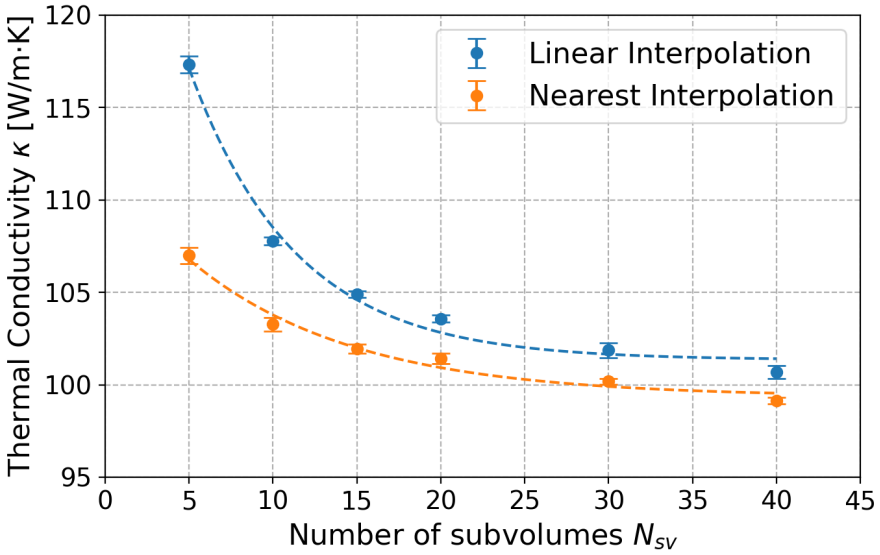


Figure 5.4: Variation of thermal conductivity as function of number of subvolumes.

The calculated system κ starts very high, and it decreases as the number of subvolumes is increased and stabilizes around $100\,\mathrm{W/m \cdot K}$. This value agrees with

the literature, as it will be shown in Chapter 6. The overestimation of κ in the cases with fewer subvolumes occurs because, with longer slices, the estimation of temperature is calculated in subvolumes that are too large as compared to the local equilibrium assumption. This phenomenon mostly affects the system borders where ∇T is steeper, since it groups regions with high and low ∇T, changing the perception of the temperature distribution. Smaller subvolumes better refine the T distribution, allowing these changes in gradient to be seen and to use them in the calculation. The difference between linear and nearest value interpolation is due to what each particle "locally sees" when scattering. In the nearest value interpolation, the temperature is kept constant inside a subvolume, and particles with short mean free path can easily reach equilibrium state. This does not happen in the linear interpolation, since the particle is always kept at a distance from the $n^0(T)$ every-time it drifts, even when it does not change subvolumes. This increases the obtained value of $\mathbf{\Phi}$ and, naturally, κ. However, as the number of subvolumes increase, the two methods get closer to each other. The uncertainties remain stable around $0.2\,\mathrm{W/m \cdot K}$, indicating that the number of subvolumes affects more the mean value than the standard deviation of the result.

5.3 Number of particles

Figure 5.5 shows the results of the sensitivity study with respect to the number of particles used in the simulation. Two groups of simulations were run: one with

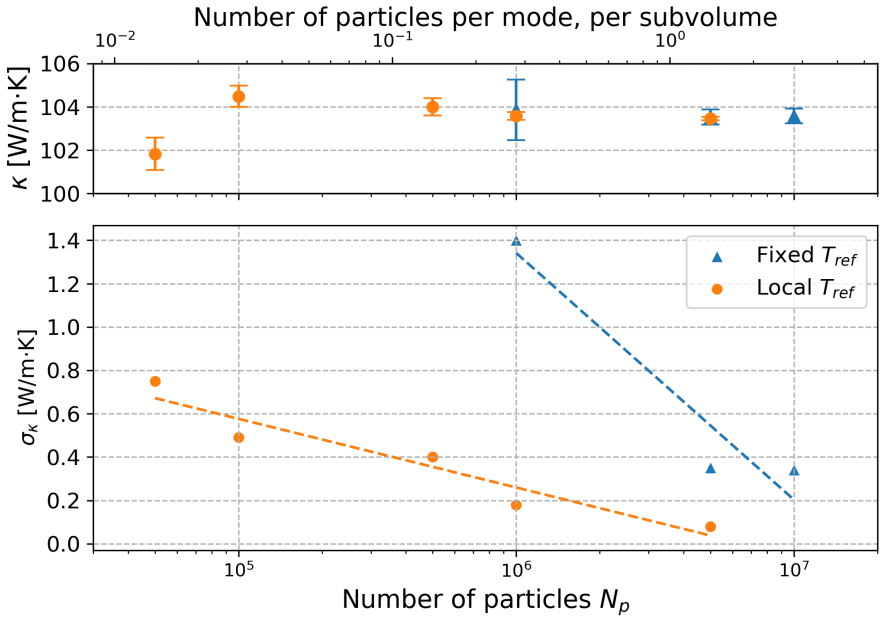


Figure 5.5: Variation of thermal conductivity as function of number of particles. In the detail, the variation of the uncertainty with N_p.

local T_{ref}, that changes from cell to cell (which is the default value in Nano-κ) and another with $T_{ref} = 300\,\mathrm{K}$ (fixed, as used in previous works [63–65]). There is no noticeable effect regarding the mean κ value calculated using each method. The calculated κ is found to be stable for all simulations, around $103\,\mathrm{W/mK}$.

The major effect can be seen in the standard deviation variation. The more particles are simulated, the more precise is the estimation of κ, both locally in each subvolume and globally in the entire domain. For large N_p, the time spent in each iteration increases proportionally, being at the end for the user to decide the acceptable σ_κ. Regardless, the calculations with local T_{ref} has consistently less noise than the simulations with fixed reference, with an approximate σ_κ being obtained with 10 to 15 times less particles (note the log x axis).

5.4 Timestep duration

The timestep duration was also varied, with the total simulated time being kept the same, at 10 ns. Both time discretization schemes (linear and exponential) exposed in Section 3.2.7 were compared. The results are shown in Fig. 5.6. In this case,

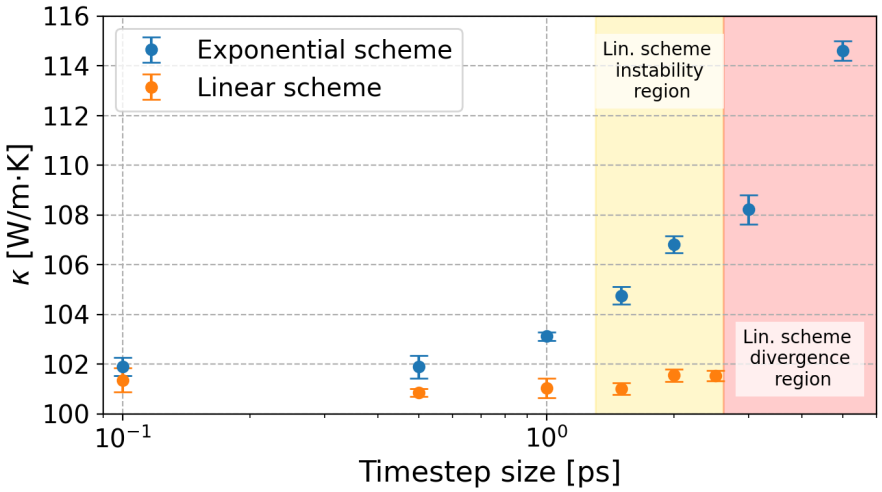


Figure 5.6: Variation of thermal conductivity as function of timestep for each time discretisation scheme. The instability regions for the linear scheme are highlighted.

the number of data-points used for the average and standard deviations evaluations must be changed such as to consider the same time interval of 200 ps. By doing this, with the exponential scheme, we can see an upward trend as timestep size increases. This increase is caused by the longer travelled distance of the phonons in a single timestep, carrying their energy farther without being affected by scattering, which in turn increases $\mathbf{\Phi}$ and consequently κ. A more correct estimation of interaction between phonons is achieved by smaller timesteps. Here, below 1 picosecond, thermal conductivity does not change much. Yet, the simulation time needed to achieve steady state is inversely proportional to Δt which can be an issue in the case of large systems. To tackle this issue, further numerical developments such as parallelization are needed.

The results with the linear scheme are also shown in Fig. 5.6. In the latter case, the largest timestep that could be applied was 2.5 ps. Above, the simulations crashed due to the instabilities shown in Section 3.2.7. These failures in calculations are due to the modes with shorter lifetimes that are the ones that become unstable first, even if they weakly contribute to the overall thermal conductivity. On the other hand, the modes with longer relaxation times are not affected as much by

the application of the linear scheme, and their contribution to κ is kept stable. To conclude on this issue, for materials with short lifetimes, the maximum Δt needs to be reduced accordingly to ensure convergence and reliability. For the studied cases, a good estimate is to have Δt in the order of the shortest τ in the data.

5.5 Initial conditions

It is also of interest to see whether the initial conditions have any influence on the final quantities. The main assumption done when starting a simulation is the initial temperature distribution. For example, the simulation can be started with a linear temperature variation between hot and cold baths, or the whole domain can be set at T_{cold} as it is done hereafter. Figure 5.7 shows the convergence of κ_s and κ for an initially constant profile as such. In the latter simulation, with

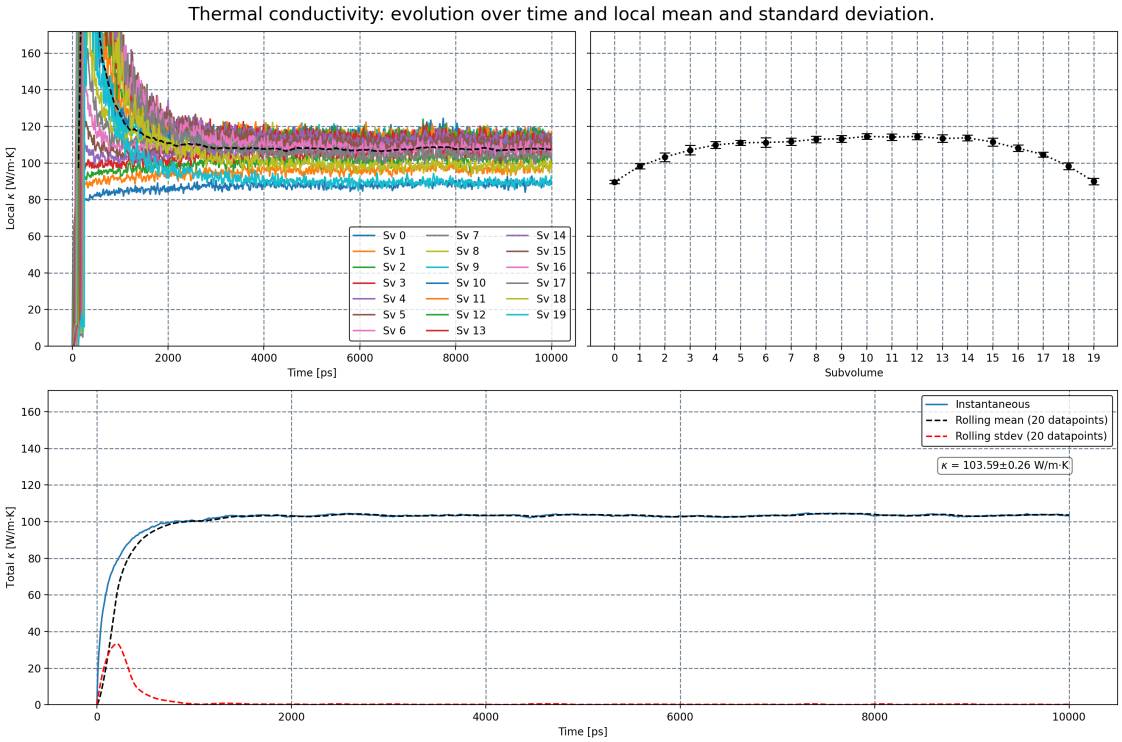


Figure 5.7: Convergence of local and global thermal conductivity with a constant cold initial T profile.

initially constant T profile, the temperatures of neighboring subvolumes are very close at the beginning and, thus, the thermal gradient is small. As a consequence, as shown in the top left plot, any small variation of ∇T induces strong fluctuation of the local thermal conductivity until the body is sufficiently heated and energy starts to find its way towards the colder side of the thin film. Also for this reason, the last subvolume's thermal conductivity stabilises much later than the first ones. This differs from the behaviour of the base case, where all thermal conductivities rise together as the heat flux increases. Comparing the plots at the bottom of each figure (Figs. 5.3 and 5.7), it can be seen also that the total κ approaches more quickly its final value with a linear initial temperature profile condition. Since κ is

calculated based on fixed ΔT, it depends only on $\mathbf{\Phi}$, which in turn is much faster adjusted because of the previously established temperature gradient.

The delayed convergence can be observed as well in the energy and heat flux net balance. Figure 5.8 shows the energy and flux balance for the constant cold case, while Fig. 5.9 shows them for the linear case.

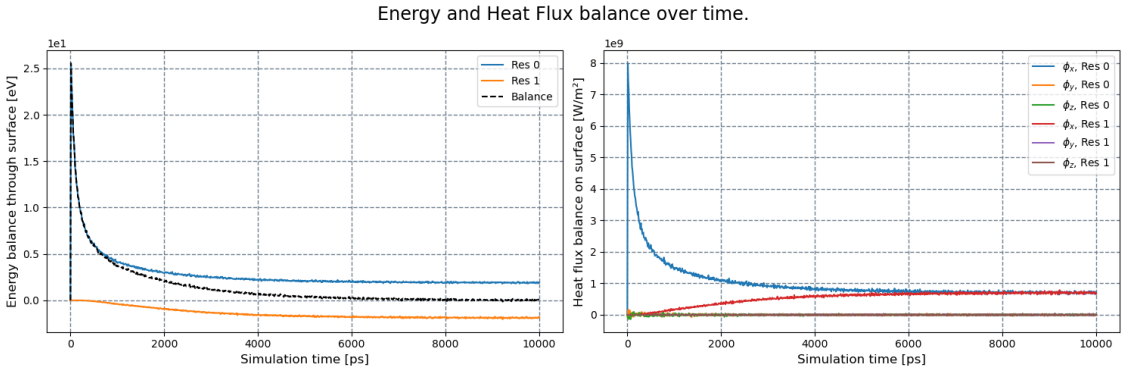


Figure 5.8: Convergence of energy balance with a constant cold initial T profile.

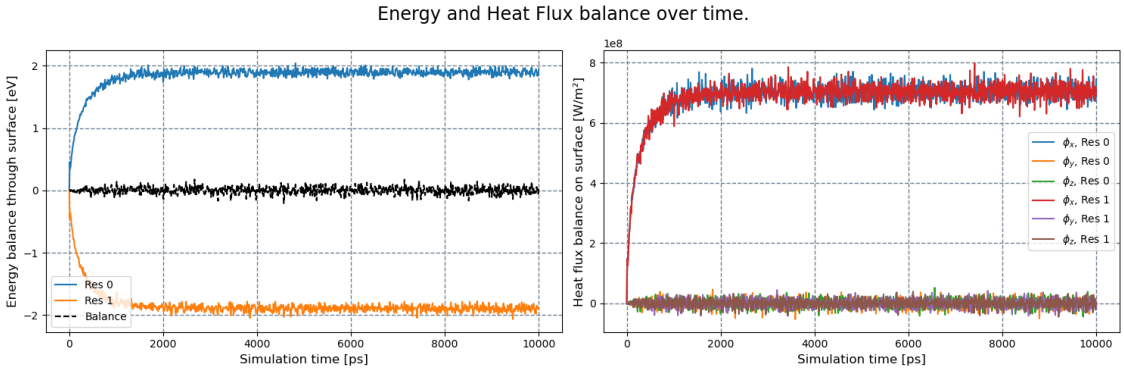


Figure 5.9: Convergence of energy balance with a linear initial T profile.

In contrast with the data shown in Fig. 5.9, the balance of energy starts high, as the only net energy exchange with the reservoir happens at the heated side. As time passes and high energy phonons start to reach the opposite side, the balance approaches zero and stabilises, but only near the 10 ns limit. In other words, based on the convergence of κ and energy balance alone, the initial linear profile can offer convergence, in a limited number of timesteps, that is at least half of that needed by the initial cold profile.

Another initial condition that can be set is the constant mean temperature, in this case 300 K. The behaviour is similar to the constant cold T case, but the convergence is as faster as the linear case. This happens because the system is already heated with an overall energy close to the final one, being only necessary to adjust the phonon distribution to the heat flux.

5.6 Final observations

Given these preliminary observations, some estimations for simulation parameters can be done for the tested material data, and will be applied to the study cases in

Chapter 6:

- The number of subvolumes was varied by keeping each subvolume 500 Å long;
- The density of particles per length of film or wire was kept the same by fixing 10^6 particles for every 2 µm;
- The initial temperature profile was set as linear;
- The timestep was 1 ps;
- The average values were taking with 100 timesteps, or 100 ps.

Dimensional parameters such as film thickness or wire diameter were changed according to the available experimental data.

Chapter 6

Study cases

The previous chapter was dedicated to exploring the effects different Nano-κ parameters have on the final results and on the evolution of the simulation. The present chapter will use the parameter estimations of Chapter 5 to run several studies, balancing precision and simulation time. The final results will then be analysed and compared to experimental data from the literature. Two final studies, with a relatively more complex geometry,are provided to demonstrate the capabilities of Nano-κ beyond those of the commonly published thin films or nanowires simulations.

6.1 Thin films - cross-plane conduction

Several cases of silicon and germanium thin films were simulated to show the effects of temperature and thickness in thermal conductivity. Figure 6.1 shows the results for both analyses.

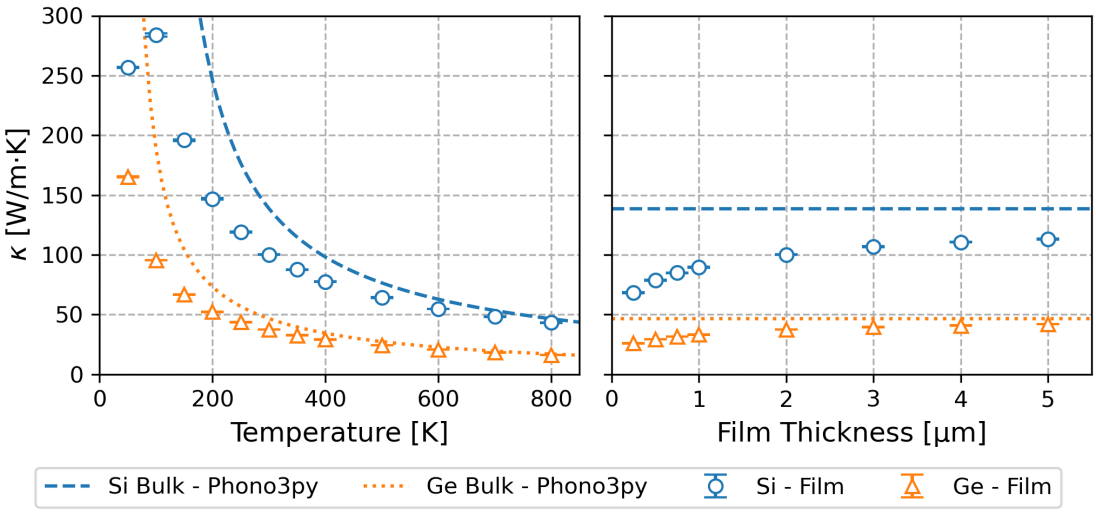


Figure 6.1: Thermal conductivity of 2 µm thin film as a function of temperature (left); Thermal conductivity at 300 K of thin film as a function of film thickness variation (right), for both Si and Ge.

In figure 6.1 (left) the expected thermal conductivity variations are recovered for both Si and Ge. In the present case, 2 µm thin film, thermal conductivities

remain smaller than in the bulk counterparts for temperature bellow 500 K. Above that temperature, our simulation results are in good agreement with the bulk values calculated by Phono3Py. This behaviour is consistent with the fact that phonon transport is dominated by phonon-phonon scattering at high temperatures and the mean free path of the energy carriers becomes small. Bellow 500 K size effects matter as the distribution of phonon mean free path can be larger than the system size. For the coldest temperatures, the relaxation time of phonons is greatly increased. If the distance between thermal reservoirs is short enough, there will be a point where the ballistic transfer between reservoirs will become more important and thermal conductivity will decrease. This point happens when a considerable part of the phonons have their mean free path in the same order of the film thickness. This is visible below 100 K for the Si film. For germanium, however, this limit is below the studied interval as Ge phonon lifetime is globally smaller than for Si.

In figure 6.1 (right) the thermal conductivity is plotted versus film thickness at room temperature. As the thickness increases, the closer to the Fourier regime the heat transfer is. With less ballistic effects, κ gradually approaches its bulk value (138.7 $\mathrm{W/m \cdot K}$ for Si and 46.6 $\mathrm{W/m \cdot K}$ for Ge, extracted from DFT-based calculations with our dispersion and lifetimes).

In addition to this general information, one interesting aspect to observe is the contribution of each frequency to the total heat flux. Figure 6.2 shows the sum of $\hbar\omega n\mathbf{v}$ for each frequency interval. Presently the full phonon spectrum of silicon (ω_{max} = 94.6 THz rad) is discretised over 100 frequency intervals of equal width. The curve is smoothed out to facilitate analysis. In the latter figure, each column corresponds to a different film thickness (500, 1000 and 2000 nm). Top row shows the distribution for the first subvolume, near the hot reservoir, while the bottom row shows the distribution for one subvolume near the middle of the film. In those calculations, the number of subvolumes (10, 20 and 40 respectively) is adjusted to keep same slice length.

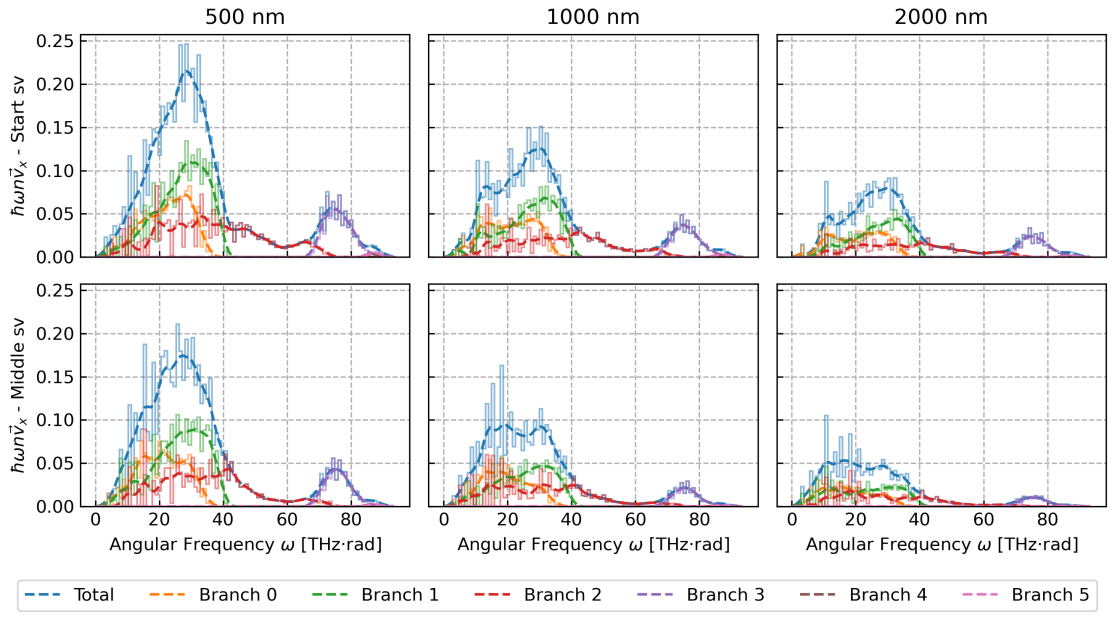


Figure 6.2: Contribution of frequency to the total heat flux for three different film thickness.

First, it can be noticed a reduction of the heat flux peaks, for all polarization branches, from left to right, as the system length increases. This reflects the fact that keeping the same hot and cold BCs the amount of energy flowing in the system per unit time naturally decreases with a increase of length. Second, we can see a major contribution of the three acoustic mode polarisations (branches 0, 1 and 2 on sub-figures) as compared to the optical branches depicted by the second peak above $60\,\mathrm{THz}\cdot\mathrm{rad}$. However contribution of the latter is not negligible and represents nearly 14% of the cumulated heat flux in the first subvolume, and 8 to 11% in the middle section of the film. Third, comparing the top and bottom rows, it can also be seen how the phonon-phonon interactions change the shape of the distribution along the same film as energy gets redistributed among modes. In particular we can observe a decrease of large frequency acoustic mode peaks (between 30 and $40\,\mathrm{THz}\cdot\mathrm{rad}$) toward low frequencies, corresponding to the scattering of phonons close to the edge of the FBZ (Umklapp processes).

The same analysis can be done for temperature variation. Figure 6.3 shows the distributions for calculations with average temeperatures of 100 K, 300 K and 500 K, and the same $\Delta T = 4\,\mathrm{K}$. For low temperatures, longer relaxation times allow higher

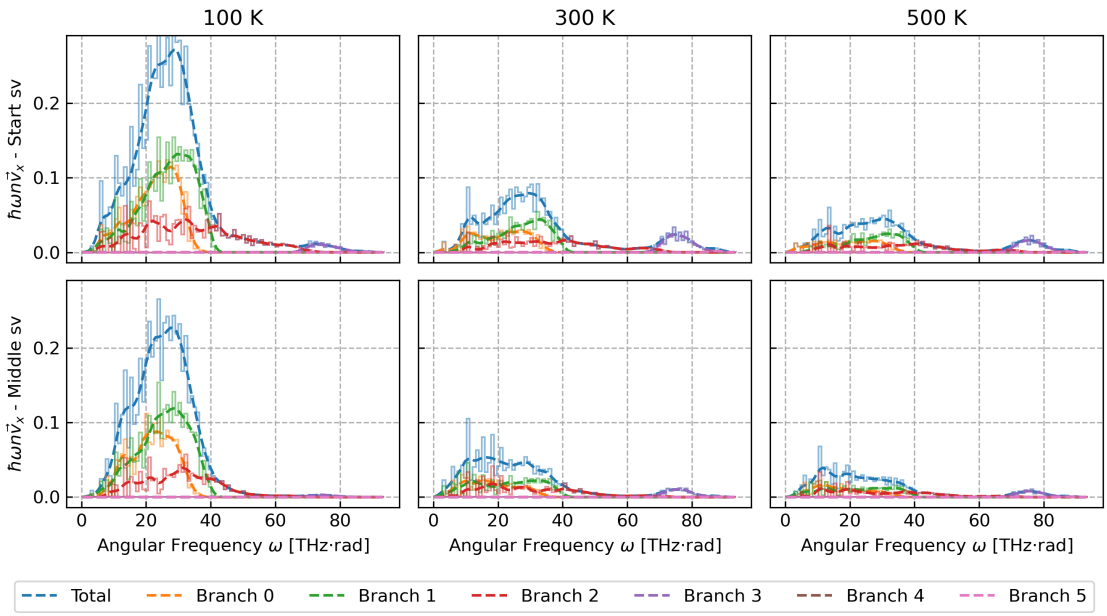


Figure 6.3: Contribution of frequency in the total heat flux for three different temperatures.

heat flux and acoustic modes are dominant. As temperature increases, the total flux is reduced and optical modes have more importance in the energy transport in agreement with Bose-Einstein distribution. This is also consistent with the thermal conductivity reduction as T increases.

The results were also compared with experimental data by Asheghi et al. [92] for temperature variation and Scott et al. [93] for film thickness variation, both for Si. The comparison is shown in Figures 6.4 and 6.5, respectively. The temperature variation study was performed in films which were 3 µm thick, while the thickness variation study was done with fixed average temperature of 300 K and $\Delta T = 4\,\mathrm{K}$.

The thermal conductivity of the film was calculated in two ways: first, considering the ΔT between reservoirs, secondly considering an "internal" ΔT related to

local temperature variation between subvolumes of the domain. At low temperatures or thin thickness, however, when there is a high ballistic effect on phonon transport, the actual temperature gradient inside the film is drastically reduced, which affects the estimation of κ. For this reason, the conductivity calculated considering the internal temperature gradient was also added to the plot. The results

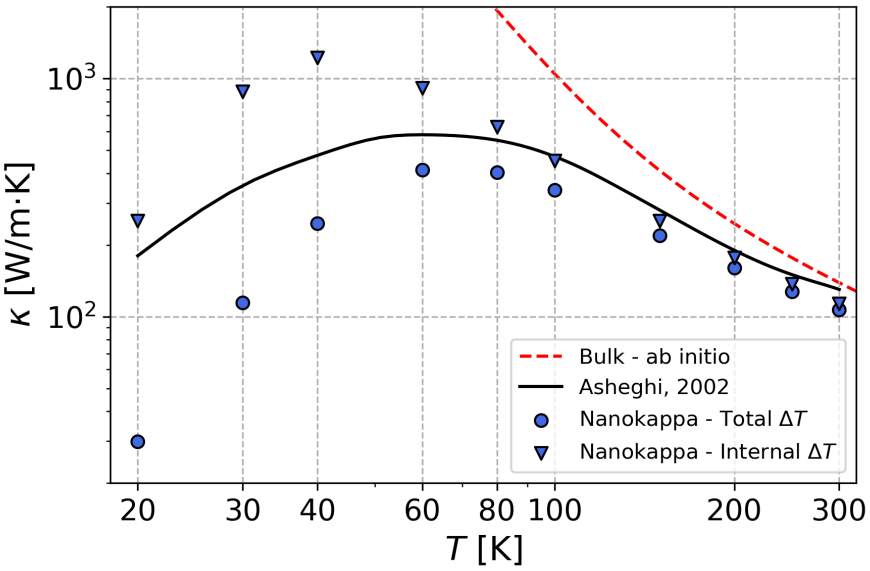


Figure 6.4: Simulation results for film cross-plane conductivity as function of temperature compared with experimental data referenced by Asheghi et al. [92]. Film thickness is 3 µm.

show good agreement with experimental data. For low temperatures and thinner films, as expected, there were higher deviations between both, total and internal, calculated κ values. Nevertheless, the estimates from simulations still work well as an upper and lower bounds for average values in low temperatures.

6.2 Thin films - in-plane conduction

In order to investigate the in-plane thermal conductivity, the thickness of the film was varied. The rough facets were considered to have a mean roughness of $\eta = 1$ nm. The varying thickness is applied between the rough facets. The results for κ are shown in Fig. 6.6.

As we have a very thick film and start to thin it, the thermal conductivity shows little variation, with a value close to the cross-plane conductivity (here, thickness along z varies, while it remains constant and equal to 2 µm along x and y directions). As the rough facets are brought closer and closer, more particles scatter on them and are diffusely reflected, decreasing heat flux and therefore thermal conductivity.

Some simulations were also executed to compare results with experimental data from Ju and Goodson [94], Liu and Asheghi [95], Hao et al. [96], Aubain and Bandaru [97, 98], Chávez-Ángel et al. [99], and Cuffe et al. [100] and Bosseboeuf et al. [101]. For this, the thickness was varied from 20 nm to 1 µm. The roughness was set to very high ($\eta = 10$ nm) to ensure high diffusivity. Results are shown in Figure 6.7.

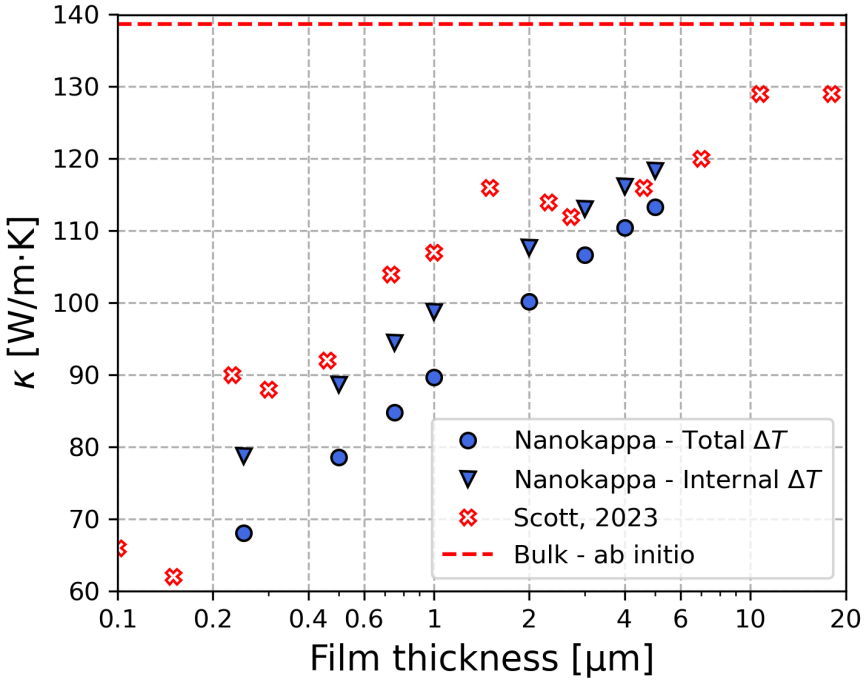


Figure 6.5: Simulation results for film cross-plane conductivity as function of film thickness compared with experimental data referenced by Scott et al. [93]. Average $T = 300\,\mathrm{K}$; $\Delta T = 4\,\mathrm{K}$.

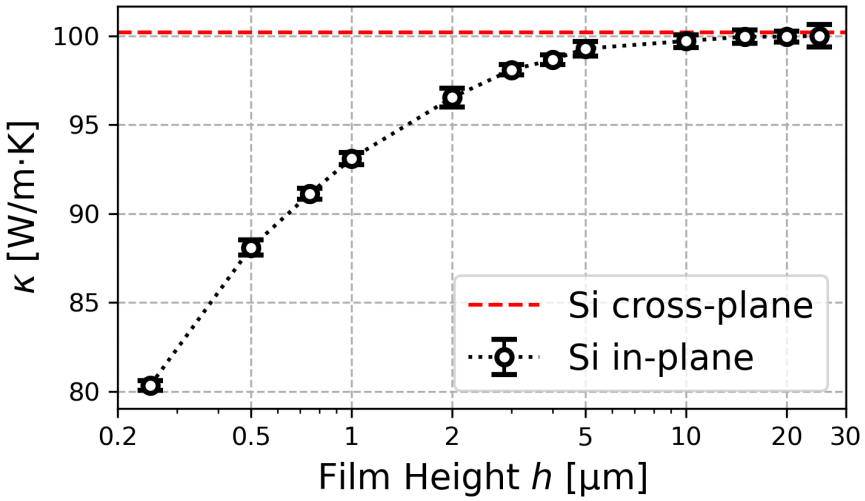


Figure 6.6: In-plane thermal conductivity of Si thin films as a function of film height and comparison with calculated cross-plane result with similar simulation parameters; length between thermostats is 2 µm; thermostats at 302 K and 298 K.

The experimental data was gathered near 300 K, with the Nano-κ calculations being performed at the same average temperature. The results show good agreement with experimental data in the entire treated range. The variation shown by the experimental data is due to different conditions and methods of measurement, as well as film growth techniques. Nevertheless, the presented trend is the same, with higher conductivity for thicker films. Due to the high roughness set as boundary

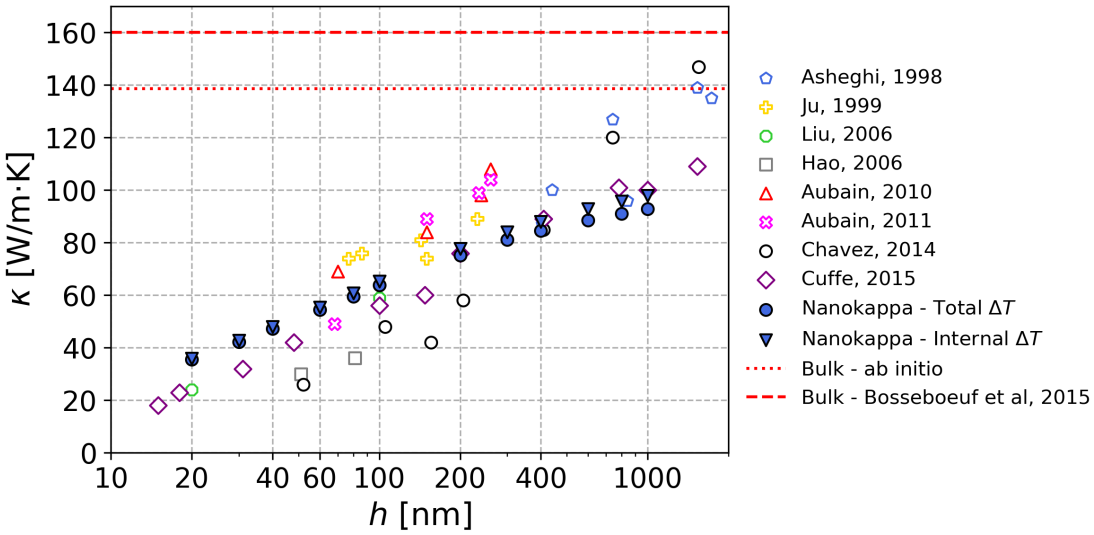


Figure 6.7: Simulation results for film in-plane conductivity compared with experimental data [94–101].

condition, the κ calculated with the total ΔT is very close to the the one calculated with internal ΔT, acknowledging weak effects related to ballistic phonons.

6.3 Nanowires

Investigating nanowires was a first attempt to increase complexity of the system geometry. For the analysed nanowires, two different shapes were tested. A first set was considered with a square cross-section and a second one had a discretised quasi-circular cross-section (polygonal cross-section with 20 sides). Both shapes were defined in order to have the same cross-sectional area. For both of them, the surface roughness of the walls were varied, and κ was compared. Figure 6.8 shows the obtained values.

As expected, in the square wire increasing wall roughness reduces the thermal conductivity. Yet, what is more interesting is the fact that the quasi-circular wire already starts with a small thermal conductivity. This is due to the polygonal side orientation that does not allow specular reflections to all modes as the crystal lattice is no longer aligned with those facets like in the square section case. This considerably increases diffuseness behaviour even without any rough walls. The curves for thermal conductivity of both shapes asymptotically approach nearly the same value, around $49\,\mathrm{W/m \cdot K}$. The latter can be considered to be the same after 2 nm roughness. No difference in temperature profile and heat flux was also observed between the two shapes when roughness was higher than this threshold.

The comparison between simulations and experimental measurements is shown in Fig. 6.9. For this, the diameter of the nanowire was varied between 22 nm and 300 nm. The simulated wire had a quasi-cylindrical cross section with 20 sides, was 2 µm long and highly rough walls. The imposed $\Delta T = 4\,\mathrm{K}$, with mean temperature of 300 K. The plots show good agreement with the experimental data, especially for large diameters. At small diameters, the conduction is slightly overestimated. Similarly to the analysis of thin films, the experimental inputs used for comparison

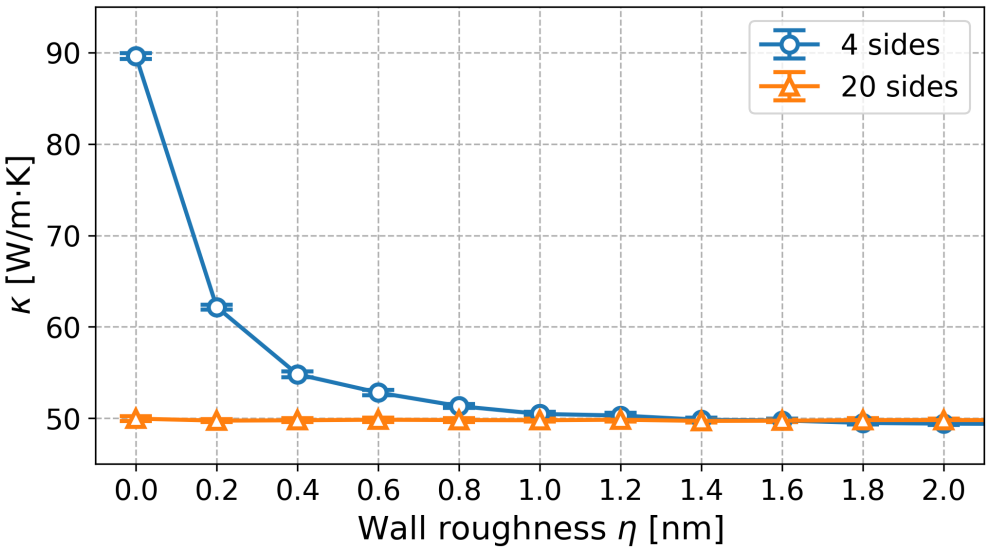


Figure 6.8: Comparison between thermal conductivity values calculated by Nano-κ for nanowires with square (4 sides) and polygonal (20 sides) cross-sections; length between thermostats at 302 K and 298 K is 2 µm.

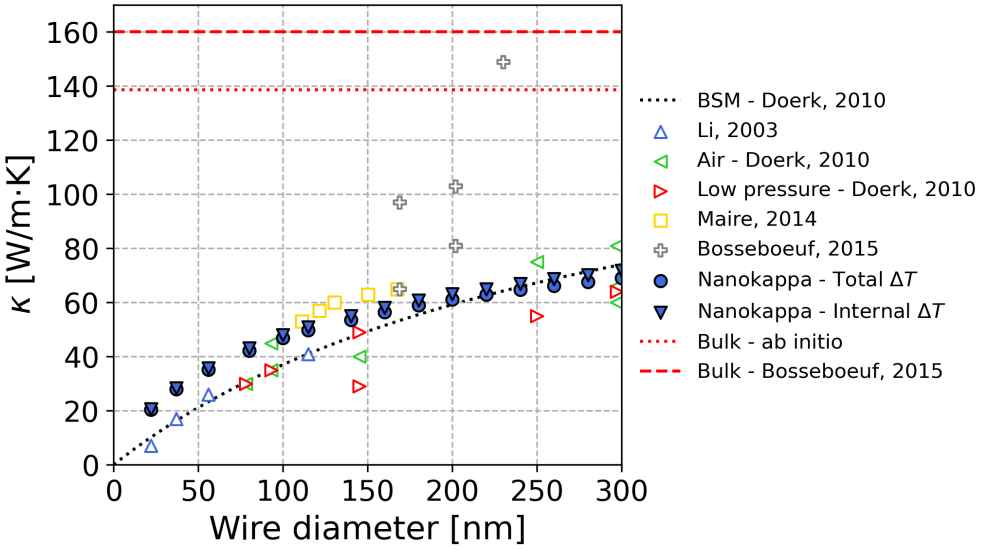


Figure 6.9: Simulation results for nanowire conductivity compared with experimental data referenced by Maire and Nomura [17], Li et al. [18], Bosseboeuf et al. [101], and Doerk, Carraro, and Maboudian [102].

relate to data-points obtained with different methods, wires with slightly different configurations (grown in different directions, poly or monocrystalline, etc.) and sometimes with missing information or error bars. For instance, the length of the nanowires reported in the experiments varied from 1.2 µm [102] to 15 µm [17]. When reported, the measurements were done at near room temperature (around 300 K). Nevertheless Nano-κ was able to give a good estimate of the thermal conductivity in the entire diameter range.

6.4 Complex geometry

To show the capabilities of Nano-κ, a mesh more complex than the previous, shown in Fig. 6.10 was simulated. The top and bottom facets were set as periodic, and temperatures were imposed on three facets, shown in blue. Facet 0 emits phonons at 302 K while facet 4 is at 298 K. The third reservoir at facet 8 had its temperature set in two different cases: a cold one (298 K) and hot one (302 K). The remaining facets were considered rough with $\eta = 0$ nm, making all reflections specular. The bounding box of the geometry measures 500 nm × 300 nm × 100 nm. For the subvolumes, a 18×9×1 grid layout was used, keeping only the subvolumes contained inside the geometry. For the grid layout, the conductivity is calculated locally for each connection between subvolumes, hence no global κ can be calculated.

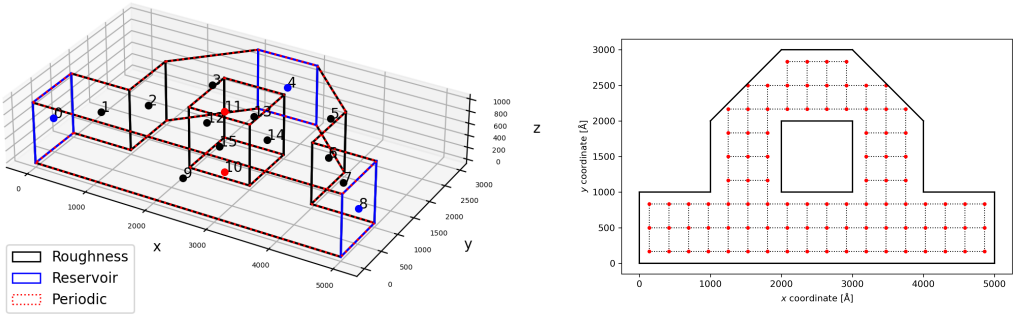


Figure 6.10: Complex geometry simulated. Left: Type of boundary conditions imposed on each facet. Right: subvolumes' reference points and connections.

Figure 6.11 shows the results after the system reaches steady state. Each plot is colored according to a different quantity. The arrows show the magnitude and direction of the heat flux. On the left column, all plots refer to the cold case (two cold BCs, one hot BC), while the plots on the right column refer to the hot case (one cold BC, two hot BCs).

The first plot is the temperature distribution. Because these simulations did not use interpolation of temperature, the borders of the subvolume can be seen clearly. The cold case has a fast decay of temperature on the left side, until it reaches around 300 K where the flux branches divide in two. Most of the heat flux goes straight to the right side, given that that is the path with less resistance. In the hot case, the temperature follows the symmetry of the domain and the boundary conditions, with the flux being redirected from the x direction to the y direction, leaving at the cold reservoir at the top.

A more refined view can be seen in the second plot, where it is shown the distribution of particles coloured according to their energy deviation from 300 K, $\delta e_{\mathbf{k}j} = \hbar\omega_{\mathbf{k}j}[n_{\mathbf{k}j} - n^0_{\mathbf{k}j}(300)]$. It is possible to see the modes responsible for ballistic effects travelling through the straight portion of the geometry as isolated blue dots close to the hot side or red dots near the cold side.

To analyse the heat flux, the two last figures show particles colored according to their heat flux in x and y directions respectively, for each of the simulated cases. To better see this on the plot, the energy flux of each particle was calculated as a deviation to the local temperature T_{local}: $\mathbf{\Phi} = \hbar\omega[n - n^0(T_{local})]\mathbf{v}$. Moreover, every particle with a calculated $|\mathbf{\Phi}_x|$ or $|\mathbf{\Phi}_y|$ of less than 2.5×10^{-3} TeV/Å^2·s was filtered out of the plots, so only the most significant particles, that carry energy are shown.

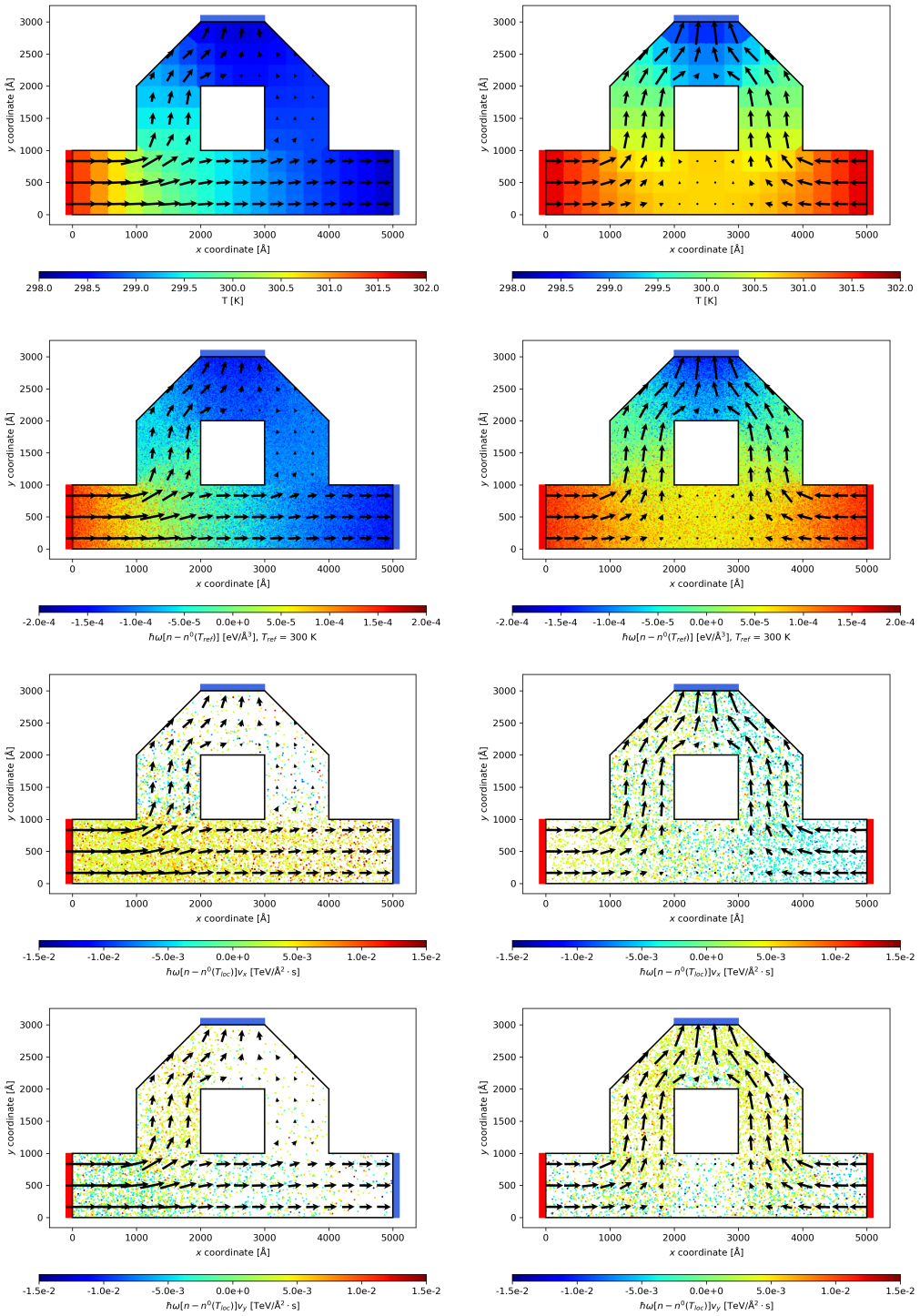


Figure 6.11: Color scatter plots for the two cases with complex geometry. In the left, the temperature of the reservoir at the right extremity is 298 K; on the right, it is 302 K. The arrows represent the heat flux magnitude and direction for each subvolume. The hot (red) and cold (blue) reservoirs' locations are shown as attached rectangles. From top to bottom: subvolume temperature, particle energy deviation ($\hbar\omega\delta n$) particle heat flux ($\hbar\omega\delta n\mathbf{v}$) in x and y directions.

This basically results in filtering the optical modes, the ones with shorter lifetimes and small group velocities, that are often close to local equilibrium, and thus do not contribute much to $\mathbf{\Phi}$.

In the x direction, we can see the amount of particles that have a positive flux (in the $+x$ direction) for the cold case, concentrated in the bottom straight path. Outside of this path, there are also particles with significant $\mathbf{\Phi}_x$, but in equal amount for $-x$ and $+x$, and thus the arrows indicate that there is basically no flux horizontally. For the hot case, the flux is again symmetric, with the left side showing shades of red and yellow ($\mathbf{\Phi}_x > 0$) and the right side exhibiting shades of blue ($\mathbf{\Phi}_x < 0$). Below the square hole, particles in both senses meet and balance out with a local flux equal zero. Similarly, in the y direction, regions with balanced blue and yellow colored particles have near zero $\mathbf{\Phi}_y$. It is also interesting to note in the hot case the behaviour of $\mathbf{\Phi}_y$ in the upper region of the geometry, between the square hole and the cold reservoir. In the vicinity of the hole, the heat flux is almost zero due to the symmetry and the proximity to the wall, creating a shaded region that is almost unaffected by the hot phonons coming from its sides. We can also see the increase in the y component of the heat flux as we approach the cold reservoir, where the geometry gradually reduces its cross-sectional area, concentrating the energy flux. Such calculations and results visualization clearly point the ability of Nano-κ to handle complex geometries and extract relevant and specific outputs that help to understand thermal transport in complex nanodevices.

Chapter 7

Conclusion

This thesis presented Nano-κ, a Python program to solve the Boltzmann Transport equation using the Monte Carlo method and *ab initio* material data as inputs.

7.1 Synthesis

The Introduction highlights the significance of investigating heat transfer in nanostructures. The inescapable importance of electronics in the contemporary world necessitates precise control over them. The evolutionary journey of the transistor is presented as an example, reflecting the computational might that has propelled human progress. The context of phonon research was introduced, highlighting its particularities in relation to macroscale heat transfer. Examples of nanoengineered structures, including phononic crystals (PnC), superlattices, and porous materials, were provided to showcase their ability to achieve effects such as thermal cloaking and frequency filtering. The applications of these structures in thermoelectrical engineering were also discussed. The most often used methods were mentioned (molecular dynamics, lattice dynamics, ab initio), and the importance of the use of the Monte Carlo method was introduced.

Chapter 2 examines phonon theory, presenting the concept of an atomic lattice and distinguishing between crystalline and amorphous materials. The text then delves into unit cells, reciprocal space, reciprocal lattice and the first Brillouin zone (FBZ) to explain how phonon wavevectors are established. The following section employs a classic unidimensional toy model of the monoatomic chain to derive the equations describing lattice waves and ultimately, the vibrational modes of the crystal. The dynamical equations were solved through the use of the harmonic approximation, followed by an explanation of the periodicity of dispersion relations. This approach was then employed to develop a generic, 3-dimensional formulation for an arbitrary crystal, accompanied by its quantum interpretation that defines phonons as quanta of lattice vibrations. Ultimately, the derivation leads to the Bose-Einstein distribution (Eq. 2.87) and the expression of energy density in solids (Eq. 2.86). The definition of wave packets was introduced by implementing the stationary phase approximation, which allowed the definition of the phonon group velocity and the subsequent calculation of the heat flux in solids. The scattering processes were described as higher order terms in the Taylor expansion of the interatomic potential, which are not considered by the harmonic approximation, and the normal and *Umklapp* processes were defined. In regards to boundary scattering, the definition of

specularity was presented, and specular and diffuse reflections were defined. Next, the method for calculating *ab-initio* phonon data and the properties included in the material data were briefly explained. Subsequently, the Boltzmann Transport Equation (BTE) along with the relaxation time approximation were described.

Chapter 3 outlines the methodological foundation of computation in Nano-κ. Initially, the Monte Carlo (MC) approach is presented, highlighting its suitability for phonon transport analysis. The MC method is exemplified by multidimensional function integration, with its mathematical formulation presented by Sobol [70]. The text delves into the MC method's context in heat transfer. It offers insightful examples of techniques employed in cases ranging from the simplest to the more complex ones, primarily in radiative heat transfer. A few of the techniques discussed include the energy partitioning method and the separation of energy into fixed and deviational quantities for variance reduction, both of which were implemented by Nano-κ. The text then presents a mathematical explanation of the calculations undertaken by Nano-κ, beginning with the definition of the geometry. The geometry is described as a watertight triangular mesh defined by vertices (points in 3D space) and a list of triangles (faces) connecting three vertices each. The calculations for face normals, areas and centroids are formulated, and the concept of facets is introduced, along with the computation of line-triangle intersections for boundary detection. The volume's triangulation was explained, defining equations for area and volumetric sampling.

Chapter 3 continues with the definition of "particles" as treated by Nano-κ. Fixed temperatures are imposed on the boundaries in the form of contact of the solid with external thermal baths, emitting phonons as black bodies. The methodology for defining the injection rate of particles for each mode into the domain utilizes the divergence theorem to preserve the conservation of particle numbers. Then the drift of the particles in the domain and their boundary scattering is detailed, using the specularity parameter as a probability to have a specular reflection, and the methodologies to choose the outgoing mode for specular and diffuse reflections are exposed. Next, the text detailed the calculation of the energy density and its conversion into a corresponding temperature value. The relaxation time approximation is applied, and an exponential scheme is utilized in lieu of the linear approach previously employed [64, 65] to enhance accuracy and allow for larger timesteps. Finally, the equations used to estimate heat flux and thermal conductivity are presented.

Chapter 4 presents the algorithmic implementation of Nano-κ, using Python as the selected programming language. This decision is supported by the language's flexibility for a first general implementation, despite its drawbacks in terms of computational speed. Object-oriented programming (OOP) terminology and advantages are also introduced for clarity in the following sections. The structure of Nano-κ is demonstrated by describing its constituent classes and how they facilitate information flow. The chapter employs a sample calculation - an evaluation of in-plane thermal conductivity in a thin film - to illustrate each simulation parameter. Subsequently, each class (`Constants`, `Mesh`, `Geometry`, `Phonon`, `Population` and `Visualisation`) is explicated. The values stocked in the `Constants` class are shown. The `Mesh` and `Geometry` classes are explained in detail, including instructions on how to define a triangular mesh, set boundary conditions, and divide the domain into subvolumes. The methodology for calculating material properties in the `Phonon` class is outlined. The `Population` class is defined, incorporating all other classes

in a simulation. The text demonstrates the process of particle initialisation, their injection, the algorithm of boundary scattering and convergence detection. Additionally, every generated output file is declared. Finally, the Visualisation class is discussed, listing each plot produced in the example calculation and the associated input parameters.

7.2 Simulation results

Chapter 5 was dedicated to the analysis of the simulation parameters in terms of their influence on the final result. The simple case of cross-plane heat conduction in a 2 µm thin film was chosen for this analysis. Three quantitative parameters were varied: the number of subvolumes, the number of particles and the time step duration. An additional analysis was performed considering different initial temperature distributions.

The number of subvolumes was found to have a significant effect on the calculated thermal conductivity by changing the local temperature estimate. More subvolumes meant a more accurate estimate of the temperature distribution, which in turn made the estimates of phonon scattering and heat flux more accurate. Two types of temperature interpolation (linear and nearest) were also tested, with the nearest interpolation giving the smaller values of κ. The difference between the two methods tended to disappear as the number of subvolumes increased. There was little or no effect on the simulation time.

The number of particles did not appear to affect the average value of κ, but only its uncertainty. In this case, for comparison, two types of reference temperature T_{ref} (used to calculate the fixed part of the phonon occupation, $n^0_{\mathbf{k}j}(T_{ref})$) were tested, one constant for the whole domain and another using the local temperature. The analysis showed that using the local T_{ref} was able to reduce the number of particles by a factor of 10 to 15, without compromising the mean or standard deviation values for κ. As the simulation time is roughly proportional to the number of particles, this greatly improved the efficiency of the simulation.

The last quantitative parameter tested was the timestep size Δt. In this case, it is expected that smaller timesteps will give better estimates of κ. Indeed, this was shown by the results. As Δt was reduced, the value of κ was also reduced until it stabilised around $102\,\mathrm{W/m \cdot K}$. The values of κ using the exponential scheme were higher as Δt increased, showing that larger time steps cause an overestimation of κ. This is expected because with larger time steps the particles carry their energy further before scattering, increasing the heat flux. The exponential scheme also showed larger values overall than the linear scheme. This is because the extrapolation of the local derivative applied by the linear scheme overestimates the scattering, which is also the case for the instabilities that occur at larger time steps (see Fig. 3.6). This overestimation gives a false sense of precision because the κ values are unchanged, but the distribution of the number of phonons is damaged.

A final sensitivity analysis was performed by starting the calculation with a uniform temperature distribution at 298 K. The results showed that the simulation took longer to converge as the system had to be heated before convergence was achieved. As the linear profile caused the system to be close to the expected energy, the simulation did not require as many iterations as the constant cold case. A similar effect can be achieved by starting the system with a constant T distribution at the

mean temperature, implying that the important factor is the total energy of the system, not its distribution.

Chapter 6 presented the comparison of the results with experimental data from the literature. The first case analysed was the cross-plane thermal conductivity of thin films of different thicknesses exposed to different mean temperatures for the same ΔT between hot and cold baths. The results showed that Nano-κ can accurately predict the variation of κ as T changes, and show how the true values differ from the expected values calculated for the bulk material. It also showed how the direct travel of phonons from one reservoir to the other, without much scattering along the path, in thin films causes κ to decrease as the film becomes thinner, and to approach the expected bulk value as the film becomes thicker.

The results for thin films in the cross-plane direction were compared with experimental measurements and showed good agreement with the data. In this analysis κ was calculated considering two different ΔT: one calculated from the temperature difference between the thermostats, the other from the temperature difference between the first and last subvolumes. This difference in calculation showed a significant deviation in the results, with the second method being closer to experimental measurements. This implies that the majority of the used experimental methods measure local values of κ rather than the conductivity of the nanodevice as a whole when subjected to an external ΔT. Nevertheless, both methods could provide useful insights depending on the application.

The in-plane conductivity of thin films was then analysed. The reduction of the film height was found to decrease κ for the same roughness. As the height increases, the diffusivity of the rough walls is less effective and the global κ approaches the value found for the cross-plane analysis. The results for the in-plane κ were also compared with experimental results. The simulation was in good agreement with measurements for high roughness boundary conditions. In this case the different methods for ΔT showed less difference in κ due to the low specularity of the reflections.

Nanowires measuring 2 µm were tested for different shapes and wall roughness. The comparison of the two shapes tested (quasicylinders with 4 and 20 sides) showed how the orientation of the walls relative to the crystal can influence the specularity of the reflections, according to the boundary scattering model implemented in Nano-κ. The square wire allowed much more specular reflections than the 20-sided one, because its walls were oriented in symmetry planes of the silicon crystal, allowing all planes to have specular correspondences available for multiple modes. This caused κ to be larger for the square wire than for the quasicylindrical one for the same cross-sectional area. However, at high roughness settings, both wires had the same value of κ as almost all reflections were diffuse in both cases. Wire simulations were performed for nanowires of different diameters and high roughness. Again, Nano-κ showed good agreement with experimental data from several works.

Finally, a relatively complex geometry was tested to demonstrate the capabilities of Nano-κwith arbitrary geometries. Two boundary conditions were tested and the distribution of temperature and heat flux was discussed. The results showed Nano-κ's ease of handling non-convex geometries containing corners and holes, simultaneously subjected to several different boundary conditions.

In summary, the first iteration of Nano-κ can be considered successful in estimating thermal conductivity when compared to experimental data in the literature.

7.3 Future developments

Despite the good results shown by Nano-κ, there is still a lot of room for improvement, both in theory and in software development.

On the theoretical side, perhaps the most important issue is the boundary scattering model. Boundary scattering is a notoriously difficult problem to solve, and the first model implemented in Nano-κ is quite simple. There are several reasons why the current model is incomplete:

- The model was originally developed by Ziman [10] for frontally incident long-wavelength acoustic waves, for which the relation $\mathbf{v}_{\mathbf{k}j} \approx \omega/\mathbf{k}$ was acceptable. Then Soffer [38] updated the model for oblique angles of incidence, still with the long wavelength assumption. The generalisation of the model to the full Brillouin zone is not formal and should be used with caution;

- There is no middle ground between specular and diffuse reflections, only a binary choice between the two;

- The effect of the correlation length in the Gaussian surface is not taken into account;

- Because the diffusely reflected particles are re-emitted at local temperature, energy is conserved on average as long as the temperature estimate near the wall is refined enough. This works well for wires and thin films, but for more complex geometries there may be deviations that affect the global energy balance. In addition, the model does not allow the energy exchange with the medium to be modelled.

Given these limitations, one implementation that could aid this research would be the ability to select an external boundary scattering model, allowing researchers to develop and investigate their own reflection models in Nano-κ.

Transmission between two materials is also of great importance for future implementations. The modelling of interfaces can provide insight into phonon transmission and help to optimise metamaterials.

On the computational side, the priority is to improve speed. Python is known for its flexibility, but not for its performance. The variance reduction techniques used in Nano-κ allowed simulations with fewer particles without compromising the results. Parallelisation can certainly help with this problem. Another technique is to use a progressive time step size, starting the simulation with a larger Δt to approximate the steady state value, and ending with a smaller Δt to properly refine the results. Coding performance bottlenecks (mainly the boundary scattering part) in compiled languages such as C or C++ could certainly help to improve the computation time.

Another planned addition is the implementation of built-in optimisation algorithms to perform geometric and material optimisation of nanocomponents. The use of stochastic methods, such as genetic algorithms and particle swarm optimisation, could help researchers find the perfect device. This could also generate a database of tested nanodevices with data on different modes under different conditions, allowing the use of machine learning algorithms that could be used to approximate the steady-state result much faster than a full simulation, leaving Nano-κ only to refine the phonon distribution at the end.

A graphical user interface would also be welcome to make it easier to visualise geometry, set boundary conditions and preview and edit plots.

In terms of experiments, the next simulation data to be analysed should include porous nanodevices and phononic crystals, both of which have been studied in the literature, although less so than thin films and nanowires.

Furthermore, a larger set of materials is planned to be simulated in order to build up a database and to properly understand how material properties (relaxation times, group velocities, number of modes in the *ab initio* data...) influence the choice of simulation parameters. However, this is again dependent on improvements in computing speed to allow such a large dataset to be generated in a reasonable time.

Conclusion (français)

Cette thèse a présenté Nano-κ, un programme Python pour résoudre l'équation de transport de Boltzmann en utilisant la méthode de Monte Carlo et *ab initio* les données matérielles comme entrées.

7.1 Synthèse

L'Introduction souligne l'importance de l'étude du transfert de chaleur dans les nanostructures. L'importance inéluctable de l'électronique dans le monde contemporain nécessite un contrôle précis. L'évolution du transistor est présentée comme un exemple, reflétant la puissance de calcul qui a propulsé le progrès humain. Le contexte de la recherche sur les phonons a été introduit, en soulignant ses particularités par rapport au transfert de chaleur à grande échelle. Des exemples de structures issues de la nanotechnologie, notamment des cristaux phononiques (PnC), des super-réseaux et des matériaux poreux, ont été présentés pour illustrer leur capacité à produire des effets tels que l'occultation thermique et le filtrage des fréquences. Les applications de ces structures dans l'ingénierie thermoélectrique ont également été discutées. Les méthodes les plus souvent utilisées ont été mentionnées (dynamique moléculaire, dynamique des réseaux, ab initio), et l'importance de l'utilisation de la méthode de Monte Carlo a été introduite.

Le chapitre 2 examine la théorie des phonons, en présentant le concept de réseau atomique et en faisant la distinction entre les matériaux cristallins et amorphes. Le texte traite ensuite des cellules unitaires, de l'espace réciproque, du réseau réciproque et de la première zone de Brillouin (FBZ) pour expliquer comment les vecteurs d'ondes des phonons sont établis. La section suivante utilise un modèle jouet unidimensionnel classique de la chaîne monoatomique pour dériver les équations décrivant les ondes de réseau et, en fin de compte, les modes vibrationnels du cristal. Les équations dynamiques ont été résolues par l'utilisation de l'approximation harmonique, suivie d'une explication de la périodicité des relations de dispersion. Cette approche a ensuite été utilisée pour développer une formulation générique tridimensionnelle pour un cristal arbitraire, accompagnée de son interprétation quantique qui définit les phonons comme des quanta de vibrations du réseau. Finalement, la dérivation conduit à la distribution de Bose-Einstein (Eq. 2.87) et à l'expression de la densité d'énergie dans les solides (Eq. 2.86). La définition des paquets d'ondes a été introduite en mettant en œuvre l'approximation de la phase stationnaire, ce qui a permis de définir la vitesse de groupe des phonons et de calculer ensuite le flux de chaleur dans les solides. Les processus de diffusion ont été décrits comme des termes d'ordre supérieur dans l'expansion de Taylor du potentiel interatomique, qui ne sont pas pris en compte par l'approximation harmonique, et les processus

normal et *Umklapp* ont été définis. En ce qui concerne la diffusion aux limites, la définition de la spécularité a été présentée, et les réflexions spéculaires et diffuses ont été définies. Ensuite, la méthode de calcul des données phononiques *ab-initio* et les propriétés incluses dans les données matérielles ont été brièvement expliquées. L'équation de transport de Boltzmann (BTE) et l'approximation du temps de relaxation ont ensuite été décrites.

Le chapitre 3 décrit les fondements méthodologiques du calcul dans Nano-κ. Dans un premier temps, l'approche de Monte Carlo (MC) est présentée, en soulignant sa pertinence pour l'analyse du transport des phonons. La méthode MC est illustrée par l'intégration de fonctions multidimensionnelles, dont la formulation mathématique est présentée dans Sobol [70]. Le texte se penche sur le contexte de la méthode MC dans le transfert de chaleur. Il offre des exemples éclairants de techniques employées dans des cas allant des plus simples aux plus complexes, principalement dans le transfert de chaleur radiatif. Parmi les techniques abordées figurent la méthode de partitionnement de l'énergie et la séparation de l'énergie en quantités fixes et déviantes pour la réduction de la variance, toutes deux mises en œuvre par Nano-κ. Le texte présente ensuite une explication mathématique des calculs effectués par Nano-κ, en commençant par la définition de la géométrie. La géométrie est décrite comme un maillage triangulaire étanche défini par des sommets (points dans l'espace 3D) et une liste de triangles (faces) reliant trois sommets chacun. Les calculs des normales des faces, des aires et des centroïdes sont formulés, et le concept de facettes est introduit, ainsi que le calcul des intersections ligne-triangle pour la détection des limites. La triangulation du volume a été expliquée, définissant les équations pour l'aire et l'échantillonnage volumétrique.

Le chapitre 3 se poursuit avec la définition des "particules" telle qu'elle est traitée par Nano-κ. Des températures fixes sont imposées aux limites sous la forme d'un contact du solide avec des bains thermiques externes, émettant des phonons en tant que corps noirs. La méthodologie pour définir le taux d'injection de particules pour chaque mode dans le domaine utilise le théorème de divergence pour préserver la conservation du nombre de particules. Ensuite, la dérive des particules dans le domaine et leur diffusion à la frontière sont détaillées, en utilisant le paramètre de spécularité comme probabilité d'avoir une réflexion spéculaire, et les méthodologies pour choisir le mode sortant pour les réflexions spéculaires et diffuses sont exposées. Ensuite, le texte détaille le calcul de la densité d'énergie et sa conversion en une valeur de température correspondante. L'approximation du temps de relaxation est appliquée, et un schéma exponentiel est utilisé à la place de l'approche linéaire précédemment employée [64, 65] pour améliorer la précision et permettre des pas de temps plus importants. Enfin, les équations utilisées pour estimer le flux de chaleur et la conductivité thermique sont présentées.

Le chapitre 4 présente la mise en œuvre algorithmique de Nano-κ, en utilisant Python comme langage de programmation sélectionné. Cette décision est justifiée par la flexibilité du langage pour une première implémentation générale, malgré ses inconvénients en termes de vitesse de calcul. La terminologie et les avantages de la programmation orientée objet (POO) sont également présentés pour plus de clarté dans les sections suivantes. La structure de Nano-κ est démontrée en décrivant ses classes constitutives et la manière dont elles facilitent le flux d'informations. Le chapitre utilise un exemple de calcul - une évaluation de la conductivité thermique dans le plan d'un film mince - pour illustrer chaque paramètre de simulation.

Ensuite, chaque classe (`Constantes`, `Maille`, `Géométrie`, `Phonon`, `Population` et `Visualisation`) est expliquée. Les valeurs stockées dans la classe `Constants` sont indiquées. Les classes `Maillage` et `Géométrie` sont expliquées en détail, avec des instructions sur la manière de définir un maillage triangulaire, de fixer des conditions limites et de diviser le domaine en sous-volumes. La méthodologie de calcul des propriétés des matériaux dans la classe `Phonon` est décrite. La classe `Population` est définie, incorporant toutes les autres classes dans une simulation. Le texte démontre le processus d'initialisation des particules, leur injection, l'algorithme de diffusion des frontières et la détection de la convergence. En outre, chaque fichier de sortie généré est déclaré. Enfin, la classe Visualisation est abordée, avec la liste de chaque graphique produit dans l'exemple de calcul et les paramètres d'entrée associés.

7.2 Résultats de la simulation

Le chapitre 5 a été consacré à l'analyse des paramètres de simulation en termes d'influence sur le résultat final. Le cas simple de la conduction thermique dans le plan transversal d'un film mince de 2 µm a été choisi pour cette analyse. Trois paramètres quantitatifs ont été modifiés : le nombre de sous-volumes, le nombre de particules et la durée du pas de temps. Une analyse supplémentaire a été réalisée en tenant compte de différentes distributions initiales de la température.

Le nombre de sous-volumes s'est avéré avoir un effet significatif sur la conductivité thermique calculée en modifiant l'estimation de la température locale. Un plus grand nombre de sous-volumes signifie une estimation plus précise de la distribution de la température, ce qui rend les estimations de la diffusion des phonons et du flux de chaleur plus précises. Deux types d'interpolation de la température (linéaire et la plus proche) ont également été testés, l'interpolation la plus proche donnant les plus petites valeurs de κ. La différence entre les deux méthodes tend à disparaître lorsque le nombre de sous-volumes augmente. L'effet sur le temps de simulation est faible ou nul.

Le nombre de particules ne semble pas affecter la valeur moyenne de κ, mais seulement son incertitude. Dans ce cas, pour comparaison, deux types de température de référence T_{ref} (utilisée pour calculer la partie fixe de l'occupation des phonons, $n^0_{\mathbf{k}j}(T_{ref})$) ont été testés, l'un constant pour l'ensemble du domaine et l'autre utilisant la température locale. L'analyse a montré que l'utilisation de T_{ref} local permettait de réduire le nombre de particules d'un facteur de 10 à 15, sans compromettre les valeurs de l'écart moyen ou de l'écart type pour κ. Le temps de simulation étant grossièrement proportionnel au nombre de particules, l'efficacité de la simulation s'en est trouvée grandement améliorée.

Le dernier paramètre quantitatif testé est la taille du pas de temps Δt. Dans ce cas, on s'attend à ce que des pas de temps plus petits donnent de meilleures estimations de κ. C'est en effet ce que montrent les résultats. Au fur et à mesure que *Deltat* était réduit, la valeur de *Kappa* était également réduite jusqu'à ce qu'elle se stabilise autour de $102\,\mathrm{W/m \cdot K}$. Les valeurs de κ utilisant le schéma exponentiel étaient plus élevées à mesure que Δt augmentait, ce qui montre que des pas de temps plus importants entraînent une surestimation de κ. Cela s'explique par le fait qu'avec des pas de temps plus importants, les particules transportent leur énergie plus loin avant de se disperser, ce qui augmente le flux de chaleur. Le schéma exponentiel a également montré des valeurs globales plus importantes que le schéma

linéaire. Cela s'explique par le fait que l'extrapolation de la dérivée locale appliquée par le schéma linéaire surestime la diffusion, ce qui est également le cas pour les instabilités qui se produisent à des pas de temps plus importants (voir Fig. 3.6). Cette surestimation donne une fausse impression de précision car les valeurs de κ sont inchangées, mais la distribution du nombre de phonons est endommagée.

Une dernière analyse de sensibilité a été réalisée en démarrant le calcul avec une distribution uniforme de la température à 298 K. Les résultats ont montré que la simulation mettait plus de temps à converger car le système devait être chauffé avant d'atteindre la convergence. Comme le profil linéaire a permis au système d'être proche de l'énergie attendue, la simulation n'a pas nécessité autant d'itérations que dans le cas du froid constant. Un effet similaire peut être obtenu en démarrant le système avec une distribution T constante à la température moyenne, ce qui implique que le facteur important est l'énergie totale du système, et non sa distribution.

Le chapitre 6 présente la comparaison des résultats avec les données expérimentales de la littérature. Le premier cas analysé était la conductivité thermique dans le plan transversal de films minces de différentes épaisseurs exposés à différentes températures moyennes pour le même ΔT entre les bains chauds et froids. Les résultats ont montré que Nano-κ peut prédire avec précision la variation de κ au fur et à mesure que T change, et montrer comment les valeurs réelles diffèrent des valeurs attendues calculées pour le matériau en vrac. Elle a également montré comment le déplacement direct des phonons d'un réservoir à l'autre, sans grande diffusion le long du chemin, dans les films minces, entraîne une diminution de κ au fur et à mesure que le film s'amincit et se rapproche de la valeur globale attendue au fur et à mesure que le film s'épaissit.

Les résultats pour les films minces dans la direction du plan transversal ont été comparés aux mesures expérimentales et ont montré une bonne concordance avec les données. Dans cette analyse, κ a été calculé en tenant compte de deux ΔT différents : l'un calculé à partir de la différence de température entre les thermostats, l'autre à partir de la différence de température entre le premier et le dernier sous-volume. Cette différence de calcul a révélé un écart significatif dans les résultats, la seconde méthode étant plus proche des mesures expérimentales. Cela implique que la majorité des méthodes expérimentales utilisées mesurent des valeurs locales de κ plutôt que la conductivité du nanodispositif dans son ensemble lorsqu'il est soumis à un ΔT externe. Néanmoins, les deux méthodes peuvent fournir des informations utiles en fonction de l'application.

La conductivité dans le plan des films minces a ensuite été analysée. On a constaté que la réduction de la hauteur du film diminuait κ pour la même rugosité. Au fur et à mesure que la hauteur augmente, la diffusivité des parois rugueuses est moins efficace et la valeur globale de κ se rapproche de la valeur trouvée pour l'analyse dans le plan transversal. Les résultats pour la κ dans le plan ont également été comparés aux résultats expérimentaux. La simulation est en bon accord avec les mesures pour les conditions limites à forte rugosité. Dans ce cas, les différentes méthodes pour ΔT ont montré moins de différence dans κ en raison de la faible spécularité des réflexions.

Des nanofils mesurant 2 µm ont été testés pour différentes formes et rugosités de paroi. La comparaison des deux formes testées (quasi-cylindres à 4 et 20 côtés) a montré comment l'orientation des parois par rapport au cristal peut influencer la spécularité des réflexions, selon le modèle de diffusion limite mis en œuvre dans

Nano-κ. Le fil carré a permis beaucoup plus de réflexions spéculaires que le fil à 20 côtés, parce que ses parois étaient orientées dans les plans de symétrie du cristal de silicium, permettant à tous les plans d'avoir des correspondances spéculaires disponibles pour plusieurs modes. C'est pourquoi κ est plus important pour le fil carré que pour le fil quasicylindrique pour la même surface de section transversale. Cependant, à des niveaux de rugosité élevés, les deux fils avaient la même valeur de κ car presque toutes les réflexions étaient diffuses dans les deux cas. Des simulations de fils ont été réalisées pour des nanofils de différents diamètres et de rugosité élevée. Là encore, Nano-κ a montré une bonne concordance avec les données expérimentales de plusieurs travaux.

Enfin, une géométrie relativement complexe a été testée pour démontrer les capacités de Nano-κavec des géométries arbitraires. Deux conditions aux limites ont été testées et la distribution de la température et du flux de chaleur a été discutée. Les résultats ont montré la facilité de Nano-κà traiter des géométries non convexes contenant des coins et des trous, soumises simultanément à plusieurs conditions limites différentes.

En résumé, la première itération de Nano-κ peut être considérée comme un succès dans l'estimation de la conductivité thermique par rapport aux données expérimentales de la littérature.

7.3 Développements futurs

Malgré les bons résultats obtenus par Nano-κ , il y a encore beaucoup de place pour l'amélioration, à la fois dans la théorie et dans le développement du logiciel.

Sur le plan théorique, le problème le plus important est peut-être le modèle de diffusion à la frontière. La diffusion à la frontière est un problème notoirement difficile à résoudre, et le premier modèle implémenté dans Nano-κ est assez simple. Le modèle actuel est incomplet pour plusieurs raisons :

- Le modèle a été développé à l'origine par Ziman [10] pour les ondes acoustiques de grande longueur d'onde incidentes frontalement, pour lesquelles la relation $\mathbf{v}_{\mathbf{k}j} \approx \omega/\mathbf{k}$ était acceptable. Ensuite, Soffer [38] a mis à jour le modèle pour les angles d'incidence obliques, toujours avec l'hypothèse de la grande longueur d'onde. La généralisation du modèle à l'ensemble de la zone de Brillouin n'est pas formelle et doit être utilisée avec prudence ;

- Il n'y a pas de juste milieu entre les réflexions spéculaires et diffuses, il n'y a qu'un choix binaire entre les deux ;

- L'effet de la longueur de corrélation dans la surface gaussienne n'est pas pris en compte ;

- Comme les particules réfléchies de manière diffuse sont réémises à la température locale, l'énergie est conservée en moyenne tant que l'estimation de la température près de la paroi est suffisamment précise. Cela fonctionne bien pour les fils et les films minces, mais pour des géométries plus complexes, il peut y avoir des déviations qui affectent l'équilibre énergétique global. En outre, le modèle ne permet pas de modéliser l'échange d'énergie avec le milieu.

Compte tenu de ces limitations, une mise en œuvre qui pourrait aider cette recherche serait la possibilité de sélectionner un modèle de diffusion à la limite externe, permettant aux chercheurs de développer et d'étudier leurs propres modèles de réflexion dans Nano-κ.

La transmission entre deux matériaux est également d'une grande importance pour les futures applications. La modélisation des interfaces peut fournir des informations sur la transmission des phonons et aider à optimiser les métamatériaux.

En ce qui concerne le calcul, la priorité est d'améliorer la vitesse. Python est connu pour sa flexibilité, mais pas pour ses performances. Les techniques de réduction de la variance utilisées dans Nano-κ ont permis des simulations avec moins de particules sans compromettre les résultats. La parallélisation peut certainement aider à résoudre ce problème. Une autre technique consiste à utiliser un pas de temps progressif, en commençant la simulation avec un Δt plus grand pour approcher la valeur de l'état stable, et en terminant avec un Δt plus petit pour affiner correctement les résultats. Le codage des goulets d'étranglement en matière de performances (principalement la partie de diffusion à la frontière) dans des langages compilés tels que C ou C++ pourrait certainement contribuer à améliorer le temps de calcul.

Un autre ajout prévu est la mise en œuvre d'algorithmes d'optimisation intégrés pour effectuer l'optimisation géométrique et matérielle des nanocomposants. L'utilisation de méthodes stochastiques, telles que les algorithmes génétiques et l'optimisation par essaims de particules, pourrait aider les chercheurs à trouver le dispositif parfait. Cela pourrait également générer une base de données de nanodispositifs testés avec des données sur différents modes dans différentes conditions, permettant l'utilisation d'algorithmes d'apprentissage automatique qui pourraient être utilisés pour approximer le résultat de l'état stable beaucoup plus rapidement qu'une simulation complète, laissant Nano-κ seulement pour affiner la distribution des phonons à la fin.

Une interface utilisateur graphique serait également la bienvenue pour faciliter la visualisation de la géométrie, la définition des conditions limites et la prévisualisation et l'édition des tracés.

En termes d'expériences, les prochaines données de simulation à analyser devraient inclure les nanodispositifs poreux et les cristaux phononiques, qui ont tous deux été étudiés dans la littérature, bien que moins que les films minces et les nanofils.

En outre, il est prévu de simuler un plus grand nombre de matériaux afin de constituer une base de données et de bien comprendre comment les propriétés des matériaux (temps de relaxation, vitesses de groupe, nombre de modes dans les données *ab initio*...) influencent le choix des paramètres de simulation. Cependant, cela dépend encore une fois des améliorations de la vitesse de calcul pour permettre la génération d'un ensemble de données aussi important dans un temps raisonnable.

Appendix A

Simulation parameters

This Appendix presents a description of the simulation parameters in Nano-κ. The following table contains:

- The long name of the parameter, to be used in the parameter declaration as `--long_name <values>`;
- The short name of the parameter, to be used in the parameter declaration as `-sn <values>`;
- A brief description of what the parameter controls;
- One or more example of the parameter declaration.;

The table is organised in categories, grouping the parameters more or less according to their function. The last group are parameters that are present in the code, but are used for development only, and should not be employed by the final user of Nano-κ. More details and examples can be consulted in the online documentation [87].

Parameter	Short name	Description	Examples
System parameters			
`--from_file`	`-ff`	Text file containing simulation parameters.	`-ff parameters.txt`
`--results_folder`	`-rf`	Folder where to save the results. When the path is relative, it is considered as starting in the working directory.	`-rf my_sim/results/`
Geometrical parameters			
`--geometry`	`-g`	Shape to be simulated. See Appendix B for more details.	`-g box`
`--dimensions`	`-d`	The dimensions of the shape to be simulated. See Appendix B for more details.	`-d 1e3 1e3 1e3`
`--scale`	`-s`	Scaling factors of the geometry, in x, y and z.	`-s 1 2 3.4`
`--geo_rotation`	`-gr`	Rotation angles (in degrees) and axes to rotate the geometry, as passed to SciPy's from_Euler function [103].	`-gr 90 45 xy`
Material parameters			
`--mat_folder`	`-mf`	The folder in which the material files are contained.	`-mf my_folder`
`--hdf_file`	`-hf`	hdf5 file containing material data from Phono3py [89].	`-hf data.hdf5`
`--poscar_file`	`-pf`	POSCAR file containing material data used by Phono3py [89].	`-pf POSCAR`
`--mat_rotation`	`-mr`	Rotation angles (in degrees) and axes to rotate the crystal, as passed to SciPy's from_Euler function [103].	`-mr 90 45 xy`
`--isotope_scat`	`-is`	Which materials should consider the isotope scattering. The respective hdf5 file should have an "gamma_isotope" field.	`-is 0`
Boundary condition parameters			
`--bound_cond`	`-bc`	The boundary conditions to be set. If the number of boundary conditions is more than the declared facets, the last BC is assigned to all non-referenced facets. `T` for imposed temperature, `R` for imposed roughness, `P` for periodicity.	`-bc T T P P R`
`--bound_values`	`-bv`	The values for each `T` and `R` BC, in the same order. Temperature in Kelvin, roughness in Angstrom.	`-bv 302 298 10`

`--bound_pos`	`-bp`	The position where to set the BCs, in the same order. The BC is applied to the closest facet of the geometry in that position. Declared as `<keyword>` x_1 y_1 z_1 x_2 y_2 z_2...x_n y_n z_n. If `<keyword>` is `absolute`, the coordinates will be read as absolute; if `relative`, the coordinates will be read as relative to the geometry's bounding box.	`-bp relative 0 0.5 0.5 1 0.5 0.5 0.5 0.5 0 0.5 0.5 1`
`--connect_pos`	`-cp`	The position of the connected facets. The facets are identified as the closest to that position. Declared the same way as `-bp`. The connections are set in pairs, so facet 1 and 2 are connected, 3 and 4, and so on.	`-cp relative 0.5 0 0.5 0.5 1 0.5`
Monte Carlo parameters			
`--particles`	`-p`	Number of particles in the domain. Declared as `total` for the total number, `pmps` for particles per mode per subvolume, or `pv` for particles per cubic angstrom.	`-p total 1e6`
`--timestep`	`-ts`	Size of the timestep for each iteration, in picoseconds.	`-ts 1`
`--subvolumes`	`-sv`	Type and number of subvolumes to divide the geometry. Can be `slice` followed by the number of subvolumes and the slicing axis (0 for x, 1 for y, 2 for z); `grid` followed by the number of subvolumes in x, y and z directions; or `voronoi`, with the number of subvolumes	`-sv slice 10 0` `-sv grid 10 2 3` `-sv voronoi 10`
`--temp_dist`	`-td`	Temperature distribution at the start. Choices: `cold`, `hot`, `linear`, `random`, or `custom`.	`-td linear`
`--temp_interp`	`-ti`	Interpolation of temperature to be used between subvolumes. Choices: `linear` for slice subvolumes, `nearest` for sliced, grid and voronoi subvolumes.	`-td linear`
`--subvol_temp`	`-st`	Temperature of each subvolume when `-td` is set to `custom`.	`-st 302 301 300 299 298`
Convergence parameters			
`--iterations`	`-i`	Maximum number of iterations to run.	`-i 10000`
`--max_sim_time`	`-mt`	Maximum computational time. The simulation is finished if the calculation runs for this long. Declared as `D-HH:MM:SS`. Not imposed if declared as `0-00:00:00`.	`-mt 2-05:30:00`

`--conv_crit`	`-cc`	Convergence criterion for steady state. The first value is the maximum relative variation between checks; the second value is the number of consecutive checks the error should be below in order to consider the calculation reached steady state. Convergence check is done every 100 iterations.	`-cc 1e-3 10`
`--n_mean`	`-nm`	Number of datapoints used to calculate the rolling mean in the time data. One datapoint is saved every 10 iterations.	`-nm 100`
Plotting parameters			
`--theme`	`-th`	Colour theme to be used in the plots. Choices: `white`, `light` or `dark`.	`-th white`
`--colormap`	`-cm`	Name of the Matplotlib colormap [104] to be used in the scatter plots.	`-cm jet`
`--fig_plot`	`-fp`	Quantities to be visualised in scatter plots. Choices: `T`, `omega`, `n`, `e`, `sv`.	`-fp e T`
Debugging parameters (for Nano-κ developers only)			
`--energy_normal`	`-en`	How to calculate the average energy of the particles. The default value is `mean`, arithmetic mean independent of the number of particles. Another option is `fixed`, based on the ideal number of particles each subvolume should have.	`-en mean`
`--reservoir_gen`	`-rg`	How to generate the particles coming from the reservoirs. The default value is `constant`, with a counter that controls deterministically when to inject an extra particle. Other options are `fixed_rate`, when the extra particle is injected at random, and `one_to_one`, when particles are injected only when a particle from the geometry enters the reservoir, in a one-to-one correspondence.	`-gn constant`
`--reference_temp`	`-rt`	Reference temperature used to calculate the fixed part of the energy. The default value is `local`, the local subvolume temperature. Another option is to set a fixed value for the whole domain, in Kelvin.	`-rt local`
`--part_dist`	`-pd`	The distribution to initialise the particles. The default is `random_subvol`, initialised in each subvolume in proportion of their volume. They can also be initialised as `center_subvol` (all positioned at their reference points), `random_domain` or `center_domain` (the same as the other two, but for the whole domain).	`-pd random_subvol`

`--empty_subvols`	`-es`	Which subvolumes to be initialised without any particles.	`-es 0 3 5`
`--bound_scat`	`-bs`	The specular boundary scattering model to be used. The default value is `v`, restricting reflection on velocity only. Another option is `k`, first restricting the reflection on wavevector, then checking for correspondence in velocity.	`-bs v`
`--rt_plot`	`-rp`	Plot updated in real time on a plot window. Significantly slows down the simulation. Choices: `T`, `omega`, `n`, `e`, `sv`.	`-rp e`
`--output`	`-op`	What type of output to use: `screen` to print on the terminal window, or `file` to save in an output.txt file.	`-op screen`

Appendix B

Standard geometries

This Appendix presents a description of the standard geometries that can be used in Nano-κ, without necessity of external STL files. A graphic representation of the parameters is given, an example on how to declare them and the geometry plot of the result.

Shape name:	Box
Description:	A parallepipedic box.
Valid keywords:	`cuboid`, `box`
Declaration:	`-g box -d <dx> <dy> <dz>`
Example:	`-g box -d 1e4 2e3 1e3`

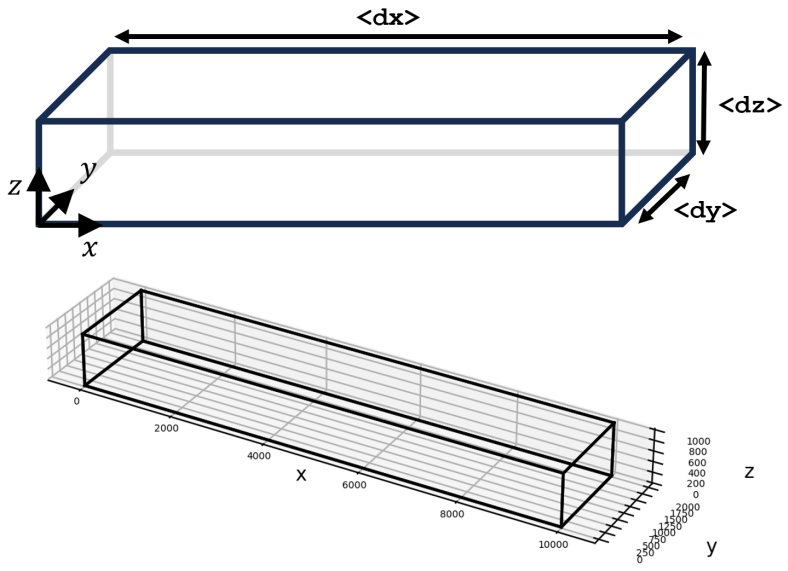


Figure B.1: Dimensions of the box geometry with example mesh plot.

Shape name: Cylinder
Description: A quasi-cylinder of radius `<R>`, created from the extrusion in the z direction of a regular polygon with `<N>` sides.
Valid keywords: `cylinder`, `rod`, `bar`
Declaration: `-g cylinder -d <L> <R> <N>`
Example: `-g cylinder -d 1e4 1e3 6`

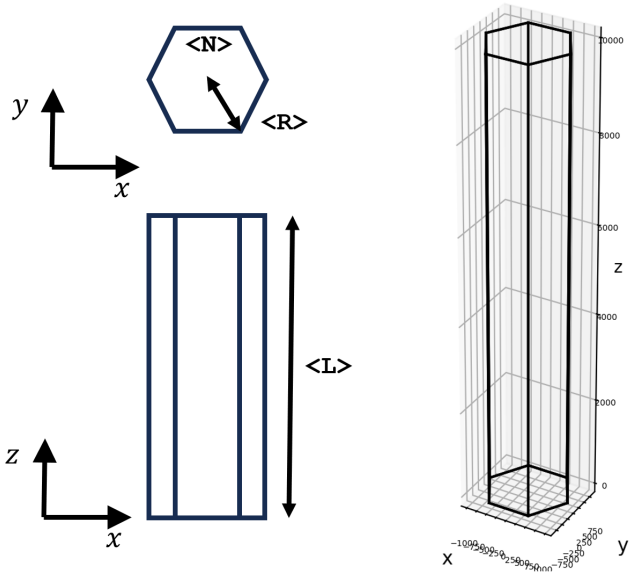


Figure B.2: Dimensions of the cylinder geometry with example mesh plot.

Shape name: Zigzag wire
Description: A corrugated quasi-cylinder with `<N>` sides and radius `<R>`, with constant cross section, with its central axis zigzagging between its original axis and another axis dislocated by `<dx>` and `<dy>`, in `<n>` sections of length `<L>`.
Valid keywords: `zigzag`
Declaration: `-g zigzag -d <L> <R> <dx> <dy> <N> <n>`
Example: `-g cylinder -d 2e2 1e2 50 50 6 4`

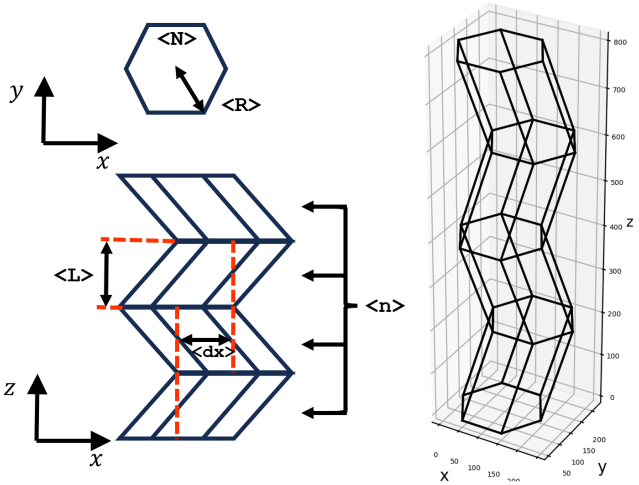


Figure B.3: Dimensions of the zigzag geometry with example mesh plot.

Shape name: Corrugated wire
Description: A corrugated quasi-cylinder with `<N>` sides and variating cross-section, with larger radius `<R>` and smaller radius `r`, divided in `<n>` sections of length `<L>`.
Valid keywords: `corrugated`
Declaration: `-g corrugated -d <L> <R> <r> <N> <n>`
Example: `-g corrugated -d 2e2 1e2 50 6 4`

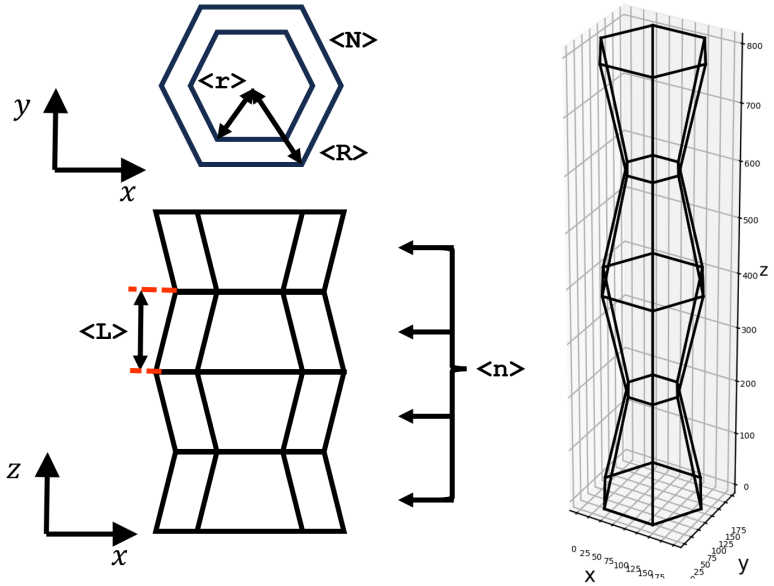


Figure B.4: Dimensions of the corrugated geometry with example mesh plot.

Shape name: Castled wire
Description: A corrugated quasi-cylinder with `<N>` sides, divided in `<n>` alternating sections of lengths `<L>` and `<l>` and radius `<R>` and `<r>`. If `<s>` is 0, the first section is the smaller one. If `<s>` is one, the larger one.
Valid keywords: `corrugated`
Declaration: `-g corrugated -d <L> <l> <R> <r> <N> <n> <s>`
Example: `-g corrugated -d 1e2 2e2 2e2 1e2 6 4 1`

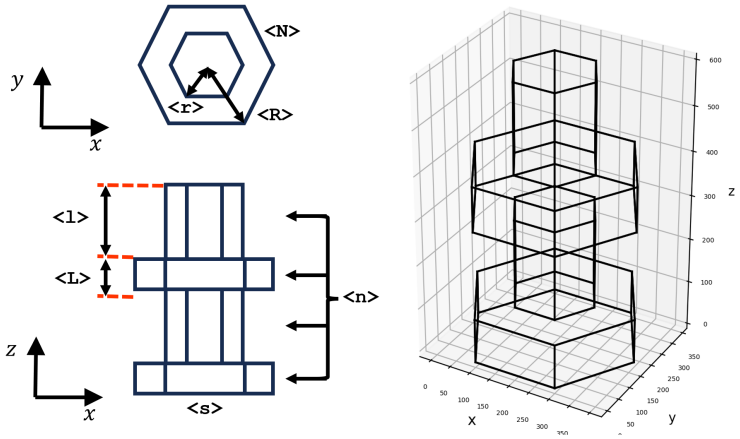


Figure B.5: Dimensions of the castle geometry with example mesh plot.

Shape name:	Starred wire
Description:	A star shaped wire with `<N>` points, large radius `<R>` and small radius `<r>`, with length `<L>`.
Valid keywords:	`star`
Declaration:	`-g star -d <L> <R> <r> <N>`
Example:	`-g corrugated -d 1e3 2e2 1e2 6`

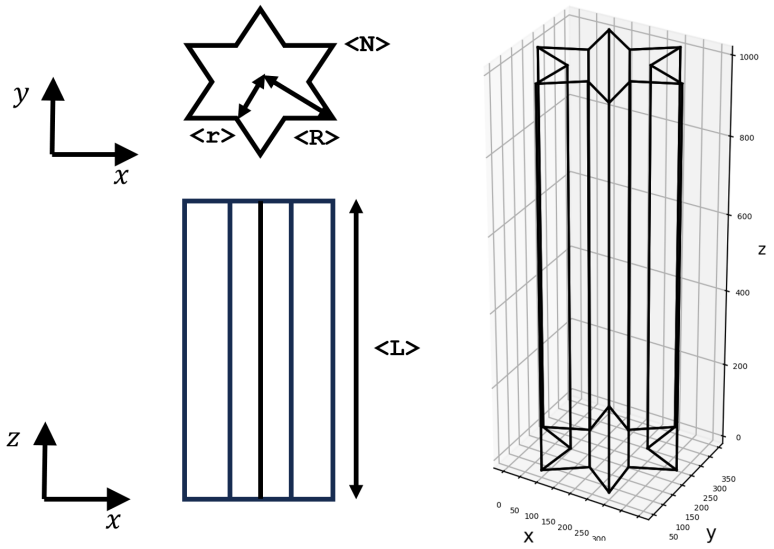


Figure B.6: Dimensions of the star geometry with example mesh plot.

Bibliography

[1] American Physics Society. *This Month in Physics History: November 17 - December 23, 1947: Invention of the First Transistor*. https://www.aps.org/publications/apsnews/200011/history.cfm. Nov. 2000.

[2] Gordon E. Moore. "Cramming more components onto integrated circuits, Reprinted from Electronics, volume 38, number 8, April 19, 1965, pp.114 ff." In: *IEEE Solid-State Circuits Society Newsletter* 11.3 (2006), pp. 33–35. DOI: 10.1109/N-SSC.2006.4785860.

[3] Our World in Data - University of Oxford. *What is Moore's Law - Our World in Data*. https://ourworldindata.org/moores-law. 2023.

[4] Hassan N. Khan, David A. Hounshell, and Erica R. H. Fuchs. "Science and research policy at the end of Moore's law". In: *Nature Electronics* 1.1 (Jan. 2018), pp. 14–21. DOI: 10.1038/s41928-017-0005-9. URL: https://doi.org/10.1038/s41928-017-0005-9.

[5] Samuel K. Moore and David Schneider. *The State of the Transistor in 3 Charts*. https://spectrum.ieee.org/transistor-density. Nov. 2022.

[6] Hossein Honarvar, Lina Yang, and Mahmoud I. Hussein. "Thermal transport size effects in silicon membranes featuring nanopillars as local resonators". In: *Applied Physics Letters* 108.26 (June 2016), p. 263101. ISSN: 0003-6951. DOI: 10.1063/1.4954739. eprint: https://pubs.aip.org/aip/apl/article-pdf/doi/10.1063/1.4954739/14481299/263101_1_online.pdf. URL: https://doi.org/10.1063/1.4954739.

[7] Paolo Gargini, Francis Balestra, and Yoshihiro Hayashi. "Roadmapping of Nanoelectronics for the New Electronics Industry". In: *Applied Sciences* 12.1 (2022). ISSN: 2076-3417. DOI: 10.3390/app12010308. URL: https://www.mdpi.com/2076-3417/12/1/308.

[8] Steffen Lange, Johanna Pohl, and Tilman Santarius. "Digitalization and energy consumption. Does ICT reduce energy demand?" In: *Ecological Economics* 176 (2020), p. 106760. ISSN: 0921-8009. DOI: https://doi.org/10.1016/j.ecolecon.2020.106760. URL: https://www.sciencedirect.com/science/article/pii/S0921800919320622.

[9] Egor D. Leshchenko and Vladimir G. Dubrovskii. "An Overview of Modeling Approaches for Compositional Control in IIIndash;V Ternary Nanowires". In: *Nanomaterials* 13.10 (2023). ISSN: 2079-4991. DOI: 10.3390/nano13101659. URL: https://www.mdpi.com/2079-4991/13/10/1659.

[10] J.M. Ziman. *Electrons and Phonons: The Theory of Transport Phenomena in Solids*. International series of monographs on physics. OUP Oxford, 1960. ISBN: 9780198507796.

[11] John S Reid. "Phonon gas". In: *Physics Education* 11.5 (July 1976), p. 348. DOI: 10.1088/0031-9120/11/5/007. URL: https://dx.doi.org/10.1088/0031-9120/11/5/007.

[12] Wei Lv and Asegun Henry. "Examining the Validity of the Phonon Gas Model in Amorphous Materials". In: *Scientific Reports* 6.1 (Dec. 2016). DOI: 10.1038/srep37675. URL: https://doi.org/10.1038/srep37675.

[13] Andrew Z. Zhao et al. "Phonon gas model for thermal conductivity of dense, strongly interacting liquids". In: *Journal of Applied Physics* 129.23 (June 2021), p. 235101. ISSN: 0021-8979. DOI: 10.1063/5.0040734. eprint: https://pubs.aip.org/aip/jap/article-pdf/doi/10.1063/5.0040734/15264628/235101_1_online.pdf. URL: https://doi.org/10.1063/5.0040734.

[14] Richa Saini, Vinod Ashokan, and B.D. Indu. "Phonon conduction in superlattices". In: *Superlattices and Microstructures* 82 (2015), pp. 574–583. ISSN: 0749-6036. DOI: https://doi.org/10.1016/j.spmi.2015.03.013. URL: https://www.sciencedirect.com/science/article/pii/S0749603615001391.

[15] K. Kordás et al. "Chip cooling with integrated carbon nanotube microfin architectures". In: *Applied Physics Letters* 90.12 (Mar. 2007), p. 123105. ISSN: 0003-6951. DOI: 10.1063/1.2714281. eprint: https://pubs.aip.org/aip/apl/article-pdf/doi/10.1063/1.2714281/13549901/123105_1_online.pdf. URL: https://doi.org/10.1063/1.2714281.

[16] Hwan Sung Choe et al. "Ion Write Microthermotics: Programing Thermal Metamaterials at the Microscale". In: *Nano Letters* 19.6 (2019). PMID: 31059272, pp. 3830–3837. DOI: 10.1021/acs.nanolett.9b00984. eprint: https://doi.org/10.1021/acs.nanolett.9b00984. URL: https://doi.org/10.1021/acs.nanolett.9b00984.

[17] Jeremie Maire and Masahiro Nomura. "Reduced thermal conductivities of Si one-dimensional periodic structure and nanowire". In: *Japanese Journal of Applied Physics* 53.6S (May 2014), 06JE09. DOI: 10.7567/JJAP.53.06JE09. URL: https://dx.doi.org/10.7567/JJAP.53.06JE09.

[18] Deyu Li et al. "Thermal conductivity of individual silicon nanowires". In: *Applied Physics Letters* 83.14 (2003), pp. 2934–2936. DOI: 10.1063/1.1616981. eprint: https://doi.org/10.1063/1.1616981. URL: https://doi.org/10.1063/1.1616981.

[19] Jérémie Maire. "Thermal phonon transport in silicon nanostructures". Theses. Ecole Centrale de Lyon, Dec. 2015. URL: https://theses.hal.science/tel-01374868.

[20] Yue Xiao, Qiyu Chen, and Qing Hao. "Inverse thermal design of nanoporous thin films for thermal cloaking". In: *Materials Today Physics* 21 (2021), p. 100477. ISSN: 2542-5293. DOI: https://doi.org/10.1016/j.mtphys.2021.100477. URL: https://www.sciencedirect.com/science/article/pii/S2542529321001383.

[21] Jian Zhang et al. "Mechanism analysis of double-layer nanoscale thermal cloak by silicon film". In: *Colloids and Surfaces A: Physicochemical and Engineering Aspects* 634 (2022), p. 128022. ISSN: 0927-7757. DOI: https://doi.org/10.1016/j.colsurfa.2021.128022. URL: https://www.sciencedirect.com/science/article/pii/S0927775721018914.

[22] V. Jean et al. "Monte Carlo simulations of phonon transport in nanoporous silicon and germanium". In: *Journal of Applied Physics* 115.2 (Jan. 2014), p. 024304. ISSN: 0021-8979. DOI: 10.1063/1.4861410. eprint: https://pubs.aip.org/aip/jap/article-pdf/doi/10.1063/1.4861410/13516505/024304_1_online.pdf. URL: https://doi.org/10.1063/1.4861410.

[23] D. Lacroix et al. "Thermal properties of nanoporous materials, large scale modelling with the use of Monte Carlo phonon transport autocorrelation". In: *Journal of Applied Physics* 134.2 (July 2023), p. 025101. ISSN: 0021-8979. DOI: 10.1063/5.0155582. eprint: https://pubs.aip.org/aip/jap/article-pdf/doi/10.1063/5.0155582/18042938/025101_1_5.0155582.pdf. URL: https://doi.org/10.1063/5.0155582.

[24] Thomas Vasileiadis et al. "Progress and perspectives on phononic crystals". In: *Journal of Applied Physics* 129.16 (Apr. 2021), p. 160901. ISSN: 0021-8979. DOI: 10.1063/5.0042337. eprint: https://pubs.aip.org/aip/jap/article-pdf/doi/10.1063/5.0042337/13337374/160901_1_online.pdf. URL: https://doi.org/10.1063/5.0042337.

[25] Zhongwei Zhang et al. "Coherent thermal transport in nano-phononic crystals: An overview". In: *APL Materials* 9.8 (Aug. 2021), p. 081102. ISSN: 2166-532X. DOI: 10.1063/5.0059024. eprint: https://pubs.aip.org/aip/apm/article-pdf/doi/10.1063/5.0059024/14565354/081102_1_online.pdf. URL: https://doi.org/10.1063/5.0059024.

[26] Yanpei Tian et al. "A Review of Tunable Wavelength Selectivity of Metamaterials in Near-Field and Far-Field Radiative Thermal Transport". In: *Materials* 11.5 (2018). ISSN: 1996-1944. DOI: 10.3390/ma11050862. URL: https://www.mdpi.com/1996-1944/11/5/862.

[27] Jatin J. Patil et al. "Failing Forward: Stability of Transparent Electrodes Based on Metal Nanowire Networks". In: *Advanced Materials* 33.5 (2021), p. 2004356. DOI: https://doi.org/10.1002/adma.202004356. eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1002/adma.202004356. URL: https://onlinelibrary.wiley.com/doi/abs/10.1002/adma.202004356.

[28] Woochul Kim. "Strategies for engineering phonon transport in thermoelectrics". In: *J. Mater. Chem. C* 3 (40 2015), pp. 10336–10348. DOI: 10.1039/C5TC01670C. URL: http://dx.doi.org/10.1039/C5TC01670C.

[29] Zhiwei Chen, Xinyue Zhang, and Yanzhong Pei. "Manipulation of Phonon Transport in Thermoelectrics". In: *Advanced Materials* 30.17 (2018), p. 1705617. DOI: https://doi.org/10.1002/adma.201705617. eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1002/adma.201705617. URL: https://onlinelibrary.wiley.com/doi/abs/10.1002/adma.201705617.

[30] Yan Sun et al. "Strategies to Improve the Thermoelectric Figure of Merit in Thermoelectric Functional Materials". In: *Frontiers in Chemistry* 10 (2022). ISSN: 2296-2646. DOI: 10.3389/fchem.2022.865281. URL: https://www.frontiersin.org/articles/10.3389/fchem.2022.865281.

[31] Xiaolong Zou et al. "Enhanced thermoelectric figure of merit in thin GaAs nanowires". In: *Nanoscale* 7 (19 2015), pp. 8776–8781. DOI: 10.1039/C5NR01892G. URL: http://dx.doi.org/10.1039/C5NR01892G.

[32] Xin Qian, Jiawei Zhou, and Gang Chen. "Phonon-engineered extreme thermal conductivity materials". In: *Nature Materials* 20.9 (Mar. 2021), pp. 1188–1202. DOI: 10.1038/s41563-021-00918-3. URL: https://doi.org/10.1038/s41563-021-00918-3.

[33] Leila Momenzadeh, Irina V. Belova, and Graeme E. Murch. "Prediction of the lattice thermal conductivity of zircon and the cubic and monoclinic phases of zirconia by molecular dynamics simulation". In: *Computational Materials Science* 176 (2020), p. 109522. ISSN: 0927-0256. DOI: https://doi.org/10.1016/j.commatsci.2020.109522. URL: https://www.sciencedirect.com/science/article/pii/S0927025620300136.

[34] Liping Shi et al. "Molecular dynamics simulation of phonon thermal transport in nanotwinned diamond with a new optimized Tersoff potential". In: *Phys. Chem. Chem. Phys.* 23 (14 2021), pp. 8336–8343. DOI: 10.1039/D1CP00399B. URL: http://dx.doi.org/10.1039/D1CP00399B.

[35] Seonghoon Jin, Young June Park, and Hong Shick Min. "A three-dimensional simulation of quantum transport in silicon nanowire transistor in the presence of electron-phonon interactions". In: *Journal of Applied Physics* 99.12 (June 2006), p. 123719. ISSN: 0021-8979. DOI: 10.1063/1.2206885. eprint: https://pubs.aip.org/aip/jap/article-pdf/doi/10.1063/1.2206885/14975451/123719_1_online.pdf. URL: https://doi.org/10.1063/1.2206885.

[36] Atsushi Togo, Laurent Chaput, and Isao Tanaka. "Distributions of phonon lifetimes in Brillouin zones". In: *Phys. Rev. B* 91 (9 Mar. 2015), p. 094306. DOI: 10.1103/PhysRevB.91.094306.

[37] Penghong Ci et al. "Giant Isotope Effect of Thermal Conductivity in Silicon Nanowires". In: *Phys. Rev. Lett.* 128 (8 Feb. 2022), p. 085901. DOI: 10.1103/PhysRevLett.128.085901. URL: https://link.aps.org/doi/10.1103/PhysRevLett.128.085901.

[38] Stephen B. Soffer. "Statistical Model for the Size Effect in Electrical Conduction". In: *Journal of Applied Physics* 38.4 (Mar. 1967), pp. 1710–1715. ISSN: 0021-8979. DOI: 10.1063/1.1709746. eprint: https://pubs.aip.org/aip/jap/article-pdf/38/4/1710/7939574/1710_1_online.pdf. URL: https://doi.org/10.1063/1.1709746.

[39] E. T. Swartz and R. O. Pohl. "Thermal boundary resistance". In: *Rev. Mod. Phys.* 61 (3 July 1989), pp. 605–668. DOI: 10.1103/RevModPhys.61.605. URL: https://link.aps.org/doi/10.1103/RevModPhys.61.605.

[40] Ravi S. Prasher and Patrick E. Phelan. "A Scattering-Mediated Acoustic Mismatch Model for the Prediction of Thermal Boundary Resistance ". In: *Journal of Heat Transfer* 123.1 (July 2000), pp. 105–112. ISSN: 0022-1481. DOI: 10.1115/1.1338138. eprint: https://asmedigitalcollection.asme.org/heattransfer/article-pdf/123/1/105/5485465/105_1.pdf. URL: https://doi.org/10.1115/1.1338138.

[41] M. Kazan. "Interpolation Between the Acoustic Mismatch Model and the Diffuse Mismatch Model for the Interface Thermal Conductance: Application to InN/GaN Superlattice". In: *Journal of Heat Transfer* 133.11 (Sept. 2011), p. 112401. ISSN: 0022-1481. DOI: 10.1115/1.4004341. eprint: https://asmedigitalcollection.asme.org/heattransfer/article-pdf/133/11/112401/5583609/112401_1.pdf. URL: https://doi.org/10.1115/1.4004341.

[42] Patrick E. Hopkins, John C. Duda, and Pamela M. Norris. "Anharmonic Phonon Interactions at Interfaces and Contributions to Thermal Boundary Conductance". In: *Journal of Heat Transfer* 133.6 (Mar. 2011), p. 062401. ISSN: 0022-1481. DOI: 10.1115/1.4003549. eprint: https://asmedigitalcollection.asme.org/heattransfer/article-pdf/133/6/062401/5793586/062401_1.pdf. URL: https://doi.org/10.1115/1.4003549.

[43] Sebastian G. Volz and Gang Chen. "Molecular dynamics simulation of thermal conductivity of silicon nanowires". In: *Applied Physics Letters* 75.14 (Oct. 1999), pp. 2056–2058. ISSN: 0003-6951. DOI: 10.1063/1.124914. eprint: https://pubs.aip.org/aip/apl/article-pdf/75/14/2056/7812711/2056_1_online.pdf. URL: https://doi.org/10.1063/1.124914.

[44] Javier V. Goicochea, Bruno Michel, and Cristina Amon. "Molecular dynamics simulations of oblique phonon scattering at semiconductor interfaces". In: *2010 3rd International Conference on Thermal Issues in Emerging Technologies Theory and Applications*. 2010, pp. 111–116. DOI: 10.1109/THETA.2010.5766386.

[45] H. Zhao and J. B. Freund. "Lattice-dynamical calculation of phonon scattering at ideal Si–Ge interfaces". In: *Journal of Applied Physics* 97.2 (Dec. 2004), p. 024903. ISSN: 0021-8979. DOI: 10.1063/1.1835565. eprint: https://pubs.aip.org/aip/jap/article-pdf/doi/10.1063/1.1835565/14682553/024903_1_online.pdf. URL: https://doi.org/10.1063/1.1835565.

[46] A. Alkurdi, S. Pailhès, and S. Merabia. "Critical angle for interfacial phonon scattering: Results from ab initio lattice dynamics calculations". In: *Applied Physics Letters* 111.9 (Aug. 2017), p. 093101. ISSN: 0003-6951. DOI: 10.1063/1.4997912. eprint: https://pubs.aip.org/aip/apl/article-pdf/doi/10.1063/1.4997912/13141668/093101_1_online.pdf. URL: https://doi.org/10.1063/1.4997912.

[47] Zhun-Yong Ong. "Specular transmission and diffuse reflection in phonon scattering at grain boundary". In: *Europhysics Letters* 133.6 (May 2021), p. 66002. DOI: 10.1209/0295-5075/133/66002. URL: https://dx.doi.org/10.1209/0295-5075/133/66002.

[48] David Lacroix. "Monte Carlo modeling of phonon transport in nanostructures". In: *146th Annual meeting & exhibition of the Minerals, Metals and Materials Society (TMS 2017)*. San Diego, CA, United States, Feb. 2017. URL: https://hal.science/hal-01578804.

[49] F. Roters et al. "Monte Carlo modeling of phonon transport in nanostructures". In: *DFG Winterschool*. Aachen, Germany, Feb. 2017. URL: https://www.dierk-raabe.com/app/download/5808235233/D+Raabe+multiscale+materials+simulation+Winter+school+2017+German+Research+Foundation+SPP+Aachen+Germany+02.pdf.

[50] Shuai Wang, LeiYang Zhao, and Yan Liu. "A concurrent multiscale method based on smoothed molecular dynamics for large-scale parallel computation at finite temperature". In: *Computer Methods in Applied Mechanics and Engineering* 406 (2023), p. 115898. ISSN: 0045-7825. DOI: https://doi.org/10.1016/j.cma.2023.115898. URL: https://www.sciencedirect.com/science/article/pii/S004578252300021X.

[51] Sandip Mazumder and Arunava Majumdar. "Monte Carlo Study of Phonon Transport in Solid Thin Films Including Dispersion and Polarization ". In: *Journal of Heat Transfer* 123.4 (Jan. 2001), pp. 749–759. ISSN: 0022-1481. DOI: 10.1115/1.1377018. eprint: https://asmedigitalcollection.asme.org/heattransfer/article-pdf/123/4/749/5912163/749_1.pdf. URL: https://doi.org/10.1115/1.1377018.

[52] Jean-Philippe M Péraud and Nicolas G Hadjiconstantinou. "An alternative approach to efficient simulation of micro/nanoscale phonon transport". In: *Applied Physics Letters* 101.15 (2012).

[53] N. D. Le et al. "Study of phonon transport across Si/Ge interfaces using Full-Band phonon Monte Carlo simulation". In: *Journal of Computational Electronics* 21.4 (May 2022), pp. 744–755. DOI: 10.1007/s10825-022-01885-x. URL: https://doi.org/10.1007/s10825-022-01885-x.

[54] Nicholas Metropolis and S. Ulam. "The Monte Carlo Method". In: *Journal of the American Statistical Association* 44.247 (1949), pp. 335–341. ISSN: 01621459. URL: http://www.jstor.org/stable/2280232 (visited on 07/19/2023).

[55] John R. Howell and Kyle J. Daun. "The Past and Future of the Monte Carlo Method in Thermal Radiation Transfer". In: *Journal of Heat Transfer* 143.10 (Sept. 2021), p. 100801. ISSN: 0022-1481. DOI: 10.1115/1.4050719. eprint: https://asmedigitalcollection.asme.org/heattransfer/article-pdf/143/10/100801/6754264/ht_143_10_100801.pdf. URL: https://doi.org/10.1115/1.4050719.

[56] Jesús Carrete et al. "almaBTE : A solver of the space–time dependent Boltzmann transport equation for phonons in structured materials". In: *Computer Physics Communications* 220 (2017), pp. 351–362. ISSN: 0010-4655. DOI: https://doi.org/10.1016/j.cpc.2017.06.023. URL: https://www.sciencedirect.com/science/article/pii/S0010465517302059.

[57] Cheng Shao, Takuma Hori, and Junichiro Shiomi. "P-TRANS: A Monte Carlo ray-tracing software to simulate phonon transport in arbitrary nanostructures". In: *Computer Physics Communications* 276 (2022), p. 108361. ISSN: 0010-4655. DOI: https://doi.org/10.1016/j.cpc.2022.108361. URL: https://www.sciencedirect.com/science/article/pii/S0010465522000807.

[58] Jesús Carrete et al. *almaBTE - First-principles thermal transport simulation*. Version 1.3. Accessed on 24/09/2023. URL: https://almabte.bitbucket.io/.

[59] Wu Li et al. "ShengBTE: a solver of the Boltzmann transport equation for phonons". In: *Comp. Phys. Commun.* 185 (2014), pp. 1747–1758. DOI: 10.1016/j.cpc.2014.02.015.

[60] Wu Li et al. *ShengBTE*. Version 1.2.0. Accessed on 24/09/2023. URL: https://www.shengbte.org/home.

[61] Martí Raya-Moreno, Xavier Cartoixà, and Jesús Carrete. "BTE-Barna: An extension of almaBTE for thermal simulation of devices based on 2D materials". In: *Computer Physics Communications* 281 (2022), p. 108504. ISSN: 0010-4655. DOI: https://doi.org/10.1016/j.cpc.2022.108504. URL: https://www.sciencedirect.com/science/article/pii/S0010465522002235.

[62] Cheng Shao, Takuma Hori, and Junichiro Shiomi. *P-TRANS documentation*. Version 1.0.0. Accessed on 24/09/2023. URL: https://www.phonon.t.u-tokyo.ac.jp/p-trans/.

[63] David Lacroix, Karl Joulain, and Denis Lemonnier. "Monte Carlo transient phonon transport in silicon and germanium at nanoscales". In: *Phys. Rev. B* 72 (6 Aug. 2005), p. 064305. DOI: 10.1103/PhysRevB.72.064305. URL: https://link.aps.org/doi/10.1103/PhysRevB.72.064305.

[64] Laurent Chaput et al. "Ab initio based calculations of the thermal conductivity at the micron scale". In: *Applied Physics Letters* 112.3 (2018), p. 033104. DOI: 10.1063/1.5010959. URL: https://doi.org/10.1063/1.5010959.

[65] Brice Davier et al. "Heat transfer in rough nanofilms and nanowires using Full Band Ab Initio Monte Carlo simulation". In: *Journal of Physics: Condensed Matter* 30.49 (Oct. 2018), p. 495902. DOI: 10.1088/1361-648X/aaea4f. URL: https://hal.archives-ouvertes.fr/hal-01906247.

[66] B.H. Bransden. *Quantum Mechanics*. Pearson Education, 2000. ISBN: 9788131708392. URL: https://books.google.fr/books?id=JBT8h%5C_FUbdEC.

[67] A.A. Maradudin, E.W. Montroll, and G.H. Weiss. *Theory of Lattice Dynamics in the Harmonic Approximation*. Solid state physics : advances in research and applications. Academic Press, 1963. ISBN: 9780126077834.

[68] A. A. Maradudin and A. E. Fein. "Scattering of Neutrons by an Anharmonic Crystal". In: *Phys. Rev.* 128 (6 Dec. 1962), pp. 2589–2608. DOI: 10.1103/PhysRev.128.2589. URL: https://link.aps.org/doi/10.1103/PhysRev.128.2589.

[69] Atsushi Togo et al. "Implementation strategies in phonopy and phono3py". In: *J. Phys. Condens. Matter* 35.35 (2023), p. 353001. DOI: 10.1088/1361-648X/acd831.

[70] Ilya M Sobol. *A primer for the Monte Carlo method*. en. Boca Raton, FL: CRC Press, May 1994.

[71] Vincent Gonneau. "Modélisation du transfert thermique par marcheurs browniens dans des milieux hétérogènes". 2020UPAST022. PhD thesis. 2020. URL: http://www.theses.fr/2020UPAST022/document.

[72] Yuankun Zhang et al. "Investigation on heat conduction of multi-sized sintered-ceramic powders based on Monte-Carlo method". In: *International Journal of Heat and Mass Transfer* 201 (2023), p. 123631. ISSN: 0017-9310. DOI: https://doi.org/10.1016/j.ijheatmasstransfer.2022.123631. URL: https://www.sciencedirect.com/science/article/pii/S0017931022011000.

[73] S. Talebi, K. Gharehbash, and H.R. Jalali. "Study on random walk and its application to solution of heat conduction equation by Monte Carlo method". In: *Progress in Nuclear Energy* 96 (2017), pp. 18–35. ISSN: 0149-1970. DOI: https://doi.org/10.1016/j.pnucene.2016.12.004. URL: https://www.sciencedirect.com/science/article/pii/S014919701630275X.

[74] John R Howell et al. *Thermal radiation heat transfer*. en. 7th ed. London, England: CRC Press, Dec. 2021. ISBN: 9780367347079.

[75] M. Galtier et al. "Integral formulation of null-collision Monte Carlo algorithms". In: *Journal of Quantitative Spectroscopy and Radiative Transfer* 125 (2013), pp. 57–68. ISSN: 0022-4073. DOI: https://doi.org/10.1016/j.jqsrt.2013.04.001. URL: https://www.sciencedirect.com/science/article/pii/S0022407313001350.

[76] Jose Ordonez-Miranda et al. "Analytical description of the radiative-conductive heat transfer in a gray medium contained between two diffuse parallel plates". In: *Applied Mathematical Modelling* 56 (2018), pp. 51–64. ISSN: 0307-904X. DOI: https://doi.org/10.1016/j.apm.2017.11.033. URL: https://www.sciencedirect.com/science/article/pii/S0307904X17307242.

[77] John Burkardt. *Computational Geometry Lab: MONTE CARLO ON TRIANGLES*. Accessed on 25/07/2023. Aug. 2018. URL: https://people.math.sc.edu/Burkardt/classes/cg_2007/cg_lab_monte_carlo_triangles.pdf.

[78] John Burkardt. *Computational Geometry Lab: MONTE CARLO ON TETRAHEDRONS*. Accessed on 25/07/2023. Dec. 2010. URL: http://people.sc.fsu.edu/~jburkardt/presentations/cg_lab_monte_carlo_tetrahedrons.pdf.

[79] Jean-Philippe M. Péraud and Nicolas G. Hadjiconstantinou. "Efficient simulation of multidimensional phonon transport using energy-based variance-reduced Monte Carlo formulations". In: *Phys. Rev. B* 84 (20 Nov. 2011), p. 205331. DOI: 10.1103/PhysRevB.84.205331. URL: https://link.aps.org/doi/10.1103/PhysRevB.84.205331.

[80] Guido van Rossum and Jelke de Boer. "Interactively testing remote servers using the Python programming language". In: *CWI Quarterly* 4.4 (Dec. 1991), pp. 283–304.

[81] Guido Van Rossum and Jelke de Boer. *Python documentation*. Version 3.9. Accessed on 28/07/2023. URL: https://docs.python.org/3.9/.

[82] Oscar Castro et al. *Landscape of High-performance Python to Develop Data Science and Machine Learning Applications*. 2023. arXiv: 2302.03307 [cs.DC].

[83] Pauli Virtanen et al. "SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python". In: *Nature Methods* 17 (2020), pp. 261–272. DOI: 10.1038/s41592-019-0686-2.

[84] Charles R. Harris et al. "Array programming with NumPy". In: *Nature* 585.7825 (Sept. 2020), pp. 357–362. DOI: 10.1038/s41586-020-2649-2. URL: https://doi.org/10.1038/s41586-020-2649-2.

[85] Dusty Phillips. *Python 3 object oriented programming*. Birmingham, England: Packt Publishing, Apr. 2023.

[86] Mark Lutz. *Learning python*. 5th ed. O'Reilly Media, June 2013.

[87] Bruno H. Silva. *Nano-κ*. https://github.com/brunohs1993/Nanokappa/. 2023.

[88] SciPy. *scipy.spatial.Delaunay - SciPy v1.11.2 Manual*. https://docs.scipy.org/doc/scipy/reference/generated/scipy.spatial.Delaunay.html. 2023.

[89] Atsushi Togo. *How to read the results stored in hdf5 files - Phono3py v2.7.0*. https://phonopy.github.io/phono3py/hdf5_howto.html. 2023.

[90] SciPy. *scipy.interpolate.LinearNDInterpolator - SciPy v1.11.2 Manual*. https://docs.scipy.org/doc/scipy/reference/generated/scipy.interpolate.LinearNDInterpolator.html. 2023.

[91] SciPy. *scipy.interpolate.interp1d - SciPy v1.11.2 Manual*. https://docs.scipy.org/doc/scipy/reference/generated/scipy.interpolate.interp1d.html. 2023.

[92] M. Asheghi et al. "Thermal conduction in doped single-crystal silicon films". In: *Journal of Applied Physics* 91.8 (Mar. 2002), pp. 5079–5088. ISSN: 0021-8979. DOI: 10.1063/1.1458057. eprint: https://pubs.aip.org/aip/jap/article-pdf/91/8/5079/10619777/5079_1_online.pdf. URL: https://doi.org/10.1063/1.1458057.

[93] Ethan A. Scott et al. "Simultaneous thickness and thermal conductivity measurements of thinned silicon from 100 nm to 17 µm". In: *Applied Physics Letters* 118.20 (May 2021), p. 202108. ISSN: 0003-6951. DOI: 10.1063/5.0050888. eprint: https://pubs.aip.org/aip/apl/article-pdf/doi/10.1063/5.0050888/13791862/202108_1_online.pdf. URL: https://doi.org/10.1063/5.0050888.

[94] Y. S. Ju and K. E. Goodson. "Phonon scattering in silicon films with thickness of order 100 nm". In: *Applied Physics Letters* 74.20 (May 1999), pp. 3005–3007. ISSN: 0003-6951. DOI: 10.1063/1.123994. eprint: https://pubs.aip.org/aip/apl/article-pdf/74/20/3005/7810349/3005_1_online.pdf. URL: https://doi.org/10.1063/1.123994.

[95] Wenjun Liu and Mehdi Asheghi. "Thermal Conductivity Measurements of Ultra-Thin Single Crystal Silicon Layers". In: *Journal of Heat Transfer* 128.1 (June 2005), pp. 75–83. ISSN: 0022-1481. DOI: 10.1115/1.2130403. eprint: https://asmedigitalcollection.asme.org/heattransfer/article-pdf/128/1/75/5641338/75_1.pdf. URL: https://doi.org/10.1115/1.2130403.

[96] Z. Hao et al. In: *8th International Conference on Solid-State and Integrated Circuit Technology ICSICT'06 (IEEE, Piscataway, 2006)* (2006), pp. 2196–2198.

[97] Max S. Aubain and Prabhakar R. Bandaru. "Determination of diminished thermal conductivity in silicon thin films using scanning thermoreflectance thermometry". In: *Applied Physics Letters* 97.25 (Dec. 2010), p. 253102. ISSN: 0003-6951. DOI: 10.1063/1.3527966. eprint: https://pubs.aip.org/aip/apl/article-pdf/doi/10.1063/1.3527966/13619580/253102_1_online.pdf. URL: https://doi.org/10.1063/1.3527966.

[98] Max S. Aubain and Prabhakar R. Bandaru. "In-plane thermal conductivity determination through thermoreflectance analysis and measurements". In: *Journal of Applied Physics* 110.8 (Oct. 2011), p. 084313. ISSN: 0021-8979. DOI: 10.1063/1.3647318. eprint: https://pubs.aip.org/aip/jap/article-pdf/doi/10.1063/1.3647318/15083664/084313_1_online.pdf. URL: https://doi.org/10.1063/1.3647318.

[99] E. Chávez-Ángel et al. "Reduction of the thermal conductivity in free-standing silicon nano-membranes investigated by non-invasive Raman thermometry". In: *APL Materials* 2.1 (Jan. 2014), p. 012113. ISSN: 2166-532X. DOI: 10.1063/1.4861796. eprint: https://pubs.aip.org/aip/apm/article-pdf/doi/10.1063/1.4861796/14556247/012113_1_online.pdf. URL: https://doi.org/10.1063/1.4861796.

[100] John Cuffe et al. "Reconstructing phonon mean-free-path contributions to thermal conductivity using nanoscale membranes". In: *Phys. Rev. B* 91 (24 June 2015), p. 245423. DOI: 10.1103/PhysRevB.91.245423. URL: https://link.aps.org/doi/10.1103/PhysRevB.91.245423.

[101] Alain Bosseboeuf et al. “Thermal and electromechanical characterization of top-down fabricated p-type silicon nanowires*”. In: *Advances in Natural Sciences: Nanoscience and Nanotechnology* 6.2 (Feb. 2015), p. 025001. DOI: 10.1088/2043-6262/6/2/025001. URL: https://dx.doi.org/10.1088/2043-6262/6/2/025001.

[102] Gregory S. Doerk, Carlo Carraro, and Roya Maboudian. “Single Nanowire Thermal Conductivity Measurements by Raman Thermography”. In: *ACS Nano* 4.8 (2010). PMID: 20731463, pp. 4908–4914. DOI: 10.1021/nn1012429. eprint: https://doi.org/10.1021/nn1012429. URL: https://doi.org/10.1021/nn1012429.

[103] SciPy. *scipy.spatial.transform.Rotation.from_euler - SciPy v1.11.2 Manual*. https://docs.scipy.org/doc/scipy/reference/generated/scipy.spatial.transform.Rotation.from_euler.html. 2023.

[104] Matplotlib. *Colormap reference - Matplotlib 3.7.2 documentation*. https://matplotlib.org/stable/gallery/color/colormap_reference.html. 2023.