Politecnico di Torino

Master Thesis

Porosity Optimization in Nanoporous materials via Machine Learning

Author:
Marco Diez

Supervisors:
Dr. Giuseppe Romano
Dr. Cristiano Fanelli
Prof. Matteo Fasano

Co-supervisor:
Prof. Pietro Asinari

A thesis submitted for the degree of Master of Science in Mechanical Engineering

in the

Department of Mechanical and Aerospace Engineering

This thesis is the result of study, data processing and research conducted by the author in a first 3-months stage abroad at the Massachusetts Institute of Technology (USA) in the Mechanical Engineering Department, and in a second stage at the Polytechnic University of Turin in the Mechanical Engineering Department.

POLITECNICO DI TORINO

Abstract

Department of Mechanical and Aerospace Engineering

Master of Science in Mechanical Engineering

Porosity Optimization in Nanoporous materials via Machine Learning

by Marco Diez

Nanomaterials have acquired a great importance in technological progress due to their tunable biological, chemical and physical properties such as electrical and thermal conductivity, with improved performance over their bulk counterparts. Materials at nanoscale have different properties than at macroscale and these properties can be managed and exploited in a wide variety of fields such as in information technologies, manufacturing processes, energy production and storage, medicine, sport, environmental applications and thermoelectric applications. In particular for the latter one, this work and the research done behind it are dedicated in terms of understanding and trying to find a method to improve the properties involved in the thermal transport at nanoscale. To this aim, this Thesis has been conducted by investigating the phonon thermal transport in nanostructured porous materials and by applying machine learning algorithms based on Bayesian theory to optimize the porosity of the nanostructure materials under study in order to achieve a specific target thermal conductivity value.

The work, after a brief introduction to nanotechnology and thermoelectric applications topic, is developed on 5 Chapters, starting with an introduction in *Chapter 1* about renewable energy current and near-future scenarios, to highlight the important role played by nanomaterials with their relative properties. *Chapter 2* describes the kinetic theory on which the used phonon thermal transport solver is based for the computation of the thermal conductivity, including the thermal equations and calculus methods adopted. *Chapter 3* detailes the OpenBTE heat transport solver used for all the simulations carried out in this work as a tool for achieving the goal of this Thesis. The work proceeds by illustrating the machine learning algorithms used in the optimization processes and the related theory behind them in *Chapter 4*. *Chapter 5* presents how the phonon transport simulations are carried out by applying machine learning together with the thermal transport solver OpenBTE to optimize the porosity in the material nanostructures and steer the effective thermal conductivity to the target value.

Acknowledgements

I am extremely grateful to my supervisors Dr. Giuseppe Romano and Dr. Cristiano Fanelli from the Massachusetts Institute of Technology. Their beautiful minds, knowledge, constant presence and interest made me possible to take always the proper path in order to obtain the final outcome of this work. The completion of this thesis would not have been possible without their support. I would like to extend my sincere thanks to Prof. Pietro Asinari and Prof. Matteo Fasano from the Polytechnic University of Turin for providing me the opportunity to work on this Thesis and for their support. Besides, I show my gratitude to them for their enormous willingness to help, insights, experience and advices that made this process easier and enjoyable.

My whole hearted gratitude to my family and friends for their constant support and motivation. In particular to my parents Davide and Speranza, my brother Mauro, my aunts Amelia and Lillina, my lifelong friend Nicola together with Moein, Luigi, Felix, Salvo and Angelo. They know that without their support I would not be here in this moment. This result is much of theirs as it is mine.

Contents

List of Figures

List of Tables

List of Source Codes

Introduction

As the devices and structures dimension has become smaller reaching the nanoscale, the physical laws and principles governing their operation change substantially.

One key component of nanotechnology is information processing, which includes data transmission, storage and processing. Since devices are becoming increasingly compact, heat generation density increases and thermal management is a major topic to focus on, for various industry sectors.

The ability of a material to conduct heat is given by the thermal conductivity. In information technology, in order to dissipate heat, such a quantity needs to be as large as possible. However, materials for thermal energy conversion must conduct heat poorly. In fact, a thermoelectric material must have low thermal conductivity and high-electrical conductivity.

Thermoelectric (TE) materials have several applications, ranging from space technologies, wearable devices and waste heat recovery. However, despite their potentials, TE materials still have low efficiency due to fact that, generally, good electrical conductors have also good thermal proprierties.

Nanostructures are a promising candidates for thermal energy conversion because they are able to decouple the eletrical and thermal properties. In fact, at the nanoscale the particle picture of heat, encoded by *phonons*, leads to a deviation of heat transport from *Fourier's law*. As this effect is less significant for electrons, at a given lengthscales, nanostructuring can lead to an incrase in the thermoelectric efficiency [1].

Recent studies have shown that the geometry of the material have a strong influence on the thermal conductivity [2]. However, due to the large number of possible configurations, engineering a nanostructure is actually challenging.

In this context the thesis has been conducted to investigate the thermal conductivity in nanoporous structures and to determine its minima by

[1] C. Vineis, A. Shakouri, A. Majumdar, and M. Kanatzidis, "Nanostructured thermoelectrics: Big efficiency gains from small features," *Current Opinion in Solid State and Materials Science*, vol. 22, pp. 3970-3980, 09 2010.

[2] G. Romano and C. J. Grossman, "Toward phonon-boundary engineering in nanoporous materials," *Applied Physics Letters*, vol. 105, no. 3, p. 033116, 2014.

applying machine learning algorithms based on Bayesian theory and to obtain the corresponding optimum porous material structure.

Chapter 1

Nanoporous Materials

1.1 Renewable energy

With the increasing global interest for sustainable energy, resource and environment protection, a constant research for new material technologies is an essential necessity. Design of innovative materials has proven to serve a great improvement in the last twenty years on the sustainable energy field supply and applications, which is fundamental to solve the shortage of resource problem and to lower the negative impact on the environment, such as the well known pollution problem.

Today, fossil fuels contribute for 90% of the world's energy consumption, and their use will peak around 2050. The extensive use of fossil fuels leads to increasingly serious environmental problems, with related climate changes, therefore the long-term availability limits of crude oil is pushing to develop a suite of sustainable energy sources and energy-storage materials with low environmental impact. The focus on the conversion of environmentally friendly energy sources leads to the development of several devices which can result in new materials for already developed devices. In particular, the study and synthesis of novel functional nanomaterials with well-controlled sizes, shapes, structures, and porosity are crucial for breakthroughs in sustainable energy technologies progress.

1.2 Nanotechnology

Nanotechnology deals with devices and materials with characteristic lengths in the range from 1 nm to 100 nm where quantum phenomena and size effects are involved. Nanoscience and nanotechnology have influenced many research areas, as shown by the crossing points in Figure 1.1 [3].

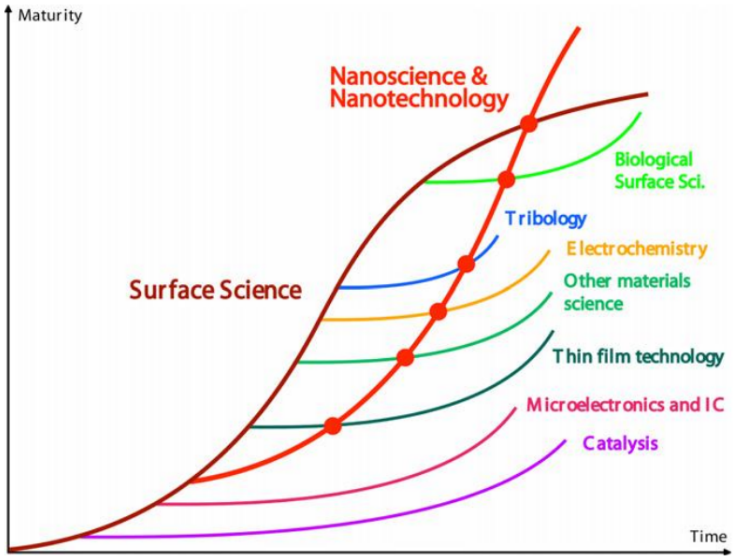

FIGURE 1.1: Nanotechnology curve development

1.2.1 Size matters on the nanoscale

It is widely known that the properties of matter change significantly when size shrinks from macroscale to nanoscale for which macroscopic descriptions of heat transfer become invalid. The size effect occurs when wavelength or mean-free-path (i.e. the average distance that a gas molecule travels between successive collisions) of a given particle (or quasi-particle) gets close to or even larger than the size of the system.

Size effects, which are well known for gases, can also be found for electrons and phonons, since both can be considered as gases existing within solids. When the phonons and electrons mean free paths become closer to or larger than a device's characteristic length, heat conduction in solids can diverge remarkably from the prediction of the diffusive law. Thermal conductivity of nanowires is no longer simply a material property but it depends also on the wire diameter or on the film thickness in case of thin films.

[3] M. Zach, "Nanoscience and nanotechnology for advanced energy system," *Current Opinion in Solid State and Materials Science*, vol.10, pp. 132-143, 2006.

The thermal conductivity curve of silicon nanowires as function of wire core-diameter is shown in Figure 1.2 [4].

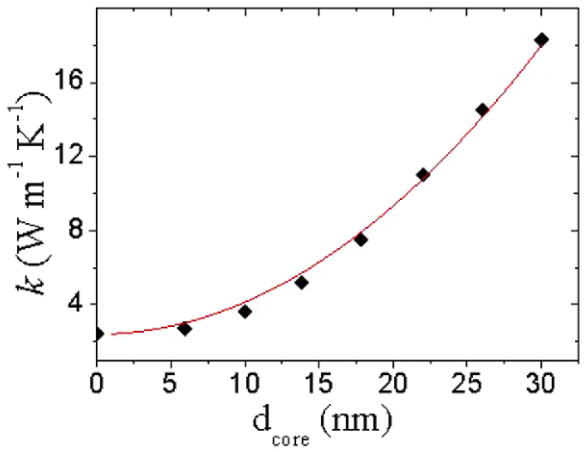

FIGURE 1.2: Room-temperature thermal conductivity data for silicon nanowires as a function of wire core-diameter

It is observed a remarkable reduction in thermal conductivity from its bulk value by reducing the nanowire cross-section. Figure 1.3 shows the thermal conductivity of silicon film layers near room temperature as a function of film thickness [5].

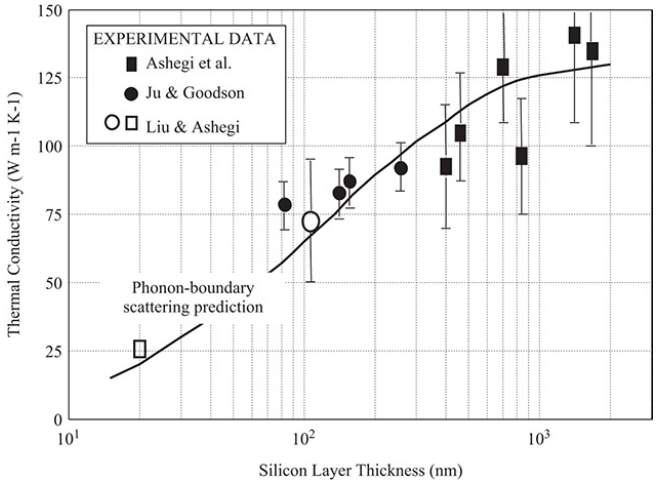

FIGURE 1.3: Room-temperature thermal conductivity data for silicon film layers as a function of film layer thickness

The experimental observations plotted in Fig. 1.3 highlight that the silicon film layers thermal conductivity reduce in value by decreasing the film thickness. In addition, the deviation of the thermal conductivity

[4]P. Ferrando Villalba, L. D'Ortenzi, G.G. Dalkiranis, E. Cara, A.F. Lopeandia, Ll Abad, R. Rurali, X. Cartoixa, N. De Leo, Z. Saghi, M. Jacob, N. Gambacorti, L. Boarino, and J. Rodriguez Viejo, "Impact of pore anisotropy on the thermal conductivity of porous Si nanowires," in *Scientific Reports*, 2018

[5]W. Liu and M. Asheghi, "Phonon-boundary scattering in ultrathin single-crystal silicon layers," *Applied Physics Letters*, vol. 84, no. 19, pp. 3819-3821, 2004.

with respect to the bulk value takes a substantial drop as the film thickness shrinks beyond 300 nm, corresponding to the order of magnitude of the phonon mean free path in silicon at room temperature. The Fourier heat conduction equation is not capable to describe the thickness dependency of thermal conductivity in silicon material. The prediction of phonon boundary scattering influence on thermal conductivity is feasible by applying the *Boltzmann Transport Equation* (BTE), with the result that it agrees remarkably well with the experimental data [6].

Heat carriers such as electrons and phonons are also material waves according to quantum mechanics. The size of the system can alter the wave characteristics and influence the energy transport such as creating new modes that do not exist in bulk material and forming standing waves. For instance, electrons in thin films can be approximated as standing waves inside a potential well of finite height as shown in Figure 1.4.

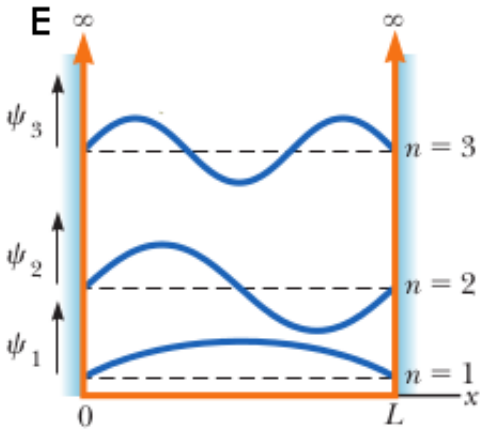

FIGURE 1.4: Standing waves in a quantum well

The condition for the formation of such standing waves is that the wavelength λ satisfies the following relation expressed by 1.1:

$$\frac{n\lambda}{2} = L \quad (n = 1, 2, 3, ...) \tag{1.1}$$

Where L is the width of the potential well.

[6] W. Liu and M. Asheghi, "Phonon-boundary scattering in ultrathin single-crystal silicon layers," *Appl. Phys. Lett.*, vol. 84, p.3819, 2004.

Given the electron wavelength, its momentum can be calculated according to the de Broglie relation as shown in 1.2 between wavelength λ and momentum p,

$$p = \frac{h}{\lambda} \tag{1.2}$$

Where h is the Planck constant ($h = 6.6 \cdot 10^{-34}$ Js). The energy of the electron is then defined as $E = \frac{p^2}{2m}$ and expressed in 1.3,

$$E_n = \frac{h^2}{8m}\left[\frac{n}{L}\right]^2 \quad (n = 1, 2, 3, ...) \tag{1.3}$$

For a free electron, $m = 9.1 \cdot 10^{-31}$ kg, and $L = 1\mu$m, $E_n = 5.9 \cdot 10^{-25} n^2$J, so that the energy separation between the $n = 1$ and $n = 2$ levels is $1.8 \cdot 10^{-24}$J. At room temperature, this energy separation is too small compared to the thermal fluctuation energy $\kappa_B T = 41.6 \cdot 10^{-22}$J to be distinguishable from thermal fluctuation. In addition, the electron mean free path at room temperature is usually much smaller than 1μm. Scattering of electrons destroys the condition for forming standing waves, making equation 1.1 inapplicable. However, as the film size is further reduced, say to $L \approx 100\mathring{A}$, scattering is negligible and energy quantization becomes observable in comparison with thermal fluctuation.

1.2.2 Time matters on the nanoscale

Alongside size effects, also transport at short time scales may differ significantly from that at longer time scales. This difference is due to the fact that classical laws are derived for time scales much longer than the time scales of microscopic processes. The average time interval τ between successive collisions of phonons can be determined as in 1.4

$$\tau = \frac{\Lambda}{v} \tag{1.4}$$

where Λ is the phonon mean free path and v the average phonon velocity. For numerous materials, τ is of the order 10^{-12} to 10^{-10} s, while a laser pulse can be as short as a few femtoseconds (1 fs $= 10^{-15}$ s). For processes shorter than the relaxation time, the classical Fourier diffusion law is not valid anymore, since the diffusion process is established by considering the multiple collisions of the heat carriers such that their motion can be considered as random. Besides the relaxation time, other time scales need to be considered, such as the time characterizing the energy exchange between electrons and phonons.

In Table 1.1 are summarized the basic characteristics of energy carriers and the relative statistics they obey.

	Molecules	Phonons	Free Electrons	Photons
Source	Atoms	Lattice	Free from nucleic bonding	Electron and atom motion
Propagation media	In vacuum or media	In media	In vacuum or media	In vacuum or media
Statistics	Boltzmann	Bose-Einstein	Fermi-Dirac	Bose-Einstein
Frequency or energy range	0-infinite	Debye cutoff	0-infinite	0-infinite
Velocity (m/s)	10^2	10^3	10^6	10^8

TABLE 1.1: Basic characteristics of energy carriers

The transport of energy carriers can be classified into different regimes depending on the characteristic length compared to the mean free path, the phase-breaking length and the coherence length as shown in Table 1.2:

<table>
<tr><td>Important Length Scales</td><td colspan="2">Regimes</td></tr>
<tr><td rowspan="2">Coherence length, l_c
Phase-breaking length, l_p</td><td>Wave regime</td><td>D <O(l_p)
D <O(l_c)</td></tr>
<tr><td>Transition regime</td><td>D $\sim$ O(l_p)
D $\sim$ O(l_c)</td></tr>
<tr><td rowspan="3">Men free path, Λ</td><td rowspan="3">Particle regime
D >O(l_p), D >O(l_c)</td><td>D <O(Λ)
ballistic</td></tr>
<tr><td>D $\sim$ O(Λ)
quasi-diffusive</td></tr>
<tr><td>D >O(Λ)
diffusive</td></tr>
<tr><td colspan="3">l_c for photon: μm - km
l_c for phonon: 10 $\mathring{A}$
l_c for electron: 100 $\mathring{A}$
$l_p \gtrsim \Lambda$</td></tr>
<tr><td colspan="3">Λ for photon: 100 $\mathring{A}$ - 1 km
Λ for phonon: 100 - 1000 $\mathring{A}$
Λ for electron: 100 - 1000 $\mathring{A}$</td></tr>
</table>

TABLE 1.2: Transport regimes of energy carriers

The different regimes are defined as follow:

- *Wave regime*, where the phase information of the energy carriers is considered and the transport is coherent;
- *Particle regime*, where phase information can be neglected and the transport is incoherent;
- *Transition regime*, in between the first two regimes, is the partially coherent transition regime from wave description to particle description.

The coherence length l_c measures the distance beyond which waves from the same source can be superimposed without considering the phase information. The phase-breaking l_p length is the distance needed to completely destroy the phase of the heat carriers for various collision processes such as phonon-phonon collision and phonon-electron collision, and it is usually of the same order of the mean free path. The overlapping length scale in Table 1.2 highlights the complexity in judging when to treat the heat carriers as waves and when to treat them as particles [7], which becomes clearer with the understanding of the wave and particle size effects described in the next chapter.

[7] G. Chen, *Nanoscale energy transport and conversion : a parallel treatment of electrons, molecules, phonons, and photons*. Oxford University Press, 2005.

Chapter 2

Kinetic Theory

The heat transport simulations of nanoporous materials realized in this work have been performed employing an innovative space-dependent Boltzmann transport equation solver for phonons implemented in the tool OpenBTE, which is treated and described more specifically in the next chapter. The kinetic theory on which OpenBTE is based on is detailed in the following paragraphs starting from the classical size-dependent effects and going through the equations used for solving the nanoscale heat transport along with the boundary conditions and the limits considered.

2.1 Phonon classical size effects

Size effects on phonon transport have been modeled through Casimir's pioneering work, which assumes that all phonons scatter diffusely at the boundaries. However, to actually design thermal transport in arbitrary shapes and structures, a more accurate model of heat transport is required. Many studies make use of the Boltzmann Transport Equation (BTE) to compute phonon transport. The simplest BTE model uses the phonon mean free path (MFP) and the gray medium approximation, that is, the MFP is independent of frequency. Even though the gray model has been helpful for understanding the thermal transport trends in many nanostructured materials, it has a poor predictive power, particularly for materials having wide MFP distributions [8]: for instance, in silicon, it has been predicted that the heat is carried by phonons having MFP spanning in between 100 nm to more than 1μm [9], and half of the heat is carried by phonons with the latter length. Recent studies have faced this challenge by including the whole phonon distribution by solving the

[8]K. Esfarjani, G. Chen, "Heat transport in silicon from first-principles calculations," *Phys. Rev. B*, vol. 84, p. 085204, 2011.

[9]G. Romano and C. J. Grossman, "Heat Conduction in Nanostructured Materials Predicted by Phonon Bulk Mean Free Path Distribution," *Journal of Heat Transfer*, vol. 137, 07 2015.

BTE frequency-dependent (FD-BTE) for the transient case and for the steady-state case [10, 11, 12].

The novel approach of the steady-state BTE is implemented in OpenBTE phonon transport solver, for computing the nanoscale heat transport with the advantage of using as input parameter only the bulk MFP distribution, that can be experimentally measured, however the FD-BTE accuracy is preserved. The method, namely MFP-BTE is coupled with the ballistic BTE and the Fourier model to include, respectively, the ballistic effect and the diffusive effect in a consistent way.

2.2 Frequency dependent BTE (FD - BTE)

Nanoscale heat transport is described as shown in 2.1 by the intensity of phonons I,

$$I(r, s, \omega, p) = \frac{v(\omega, p)\hbar\omega f(\omega, p)D(\omega, p)}{4\pi} \tag{2.1}$$

where:

- r is the spatial coordinate
- s is the direction of phonon transport within a unit solid angle
- ω and p are respectively the phonon frequency and polarization
- $v(\omega,p)$ is the velocity magnitude of the phonon group
- $\hbar$ is the reduced Planck's constant
- $f(\omega,p)$ is the nonequilibrium phonon distribution
- $D(\omega,p)$ is the states phonon density

The FD-BTE original formulation describes the intensity of phonon I under the relaxation time approximation as shown in 2.2 [13]:

$$\frac{1}{v}\frac{\partial I}{\partial t} + s \cdot \nabla I = \frac{I^0(T_L) - I}{v\tau} \tag{2.2}$$

[10] A. J. Minnich, G. Chen, S. Mannor, and B. Yilbas, "Quasiballistic heat transfer studied using the frequency-dependent boltzmann transport equation," *Phys. Rev. B*, vol. 84, p. 234207, 2011.

[11] T.-Y- Hsieh, H. Lin, T.-J. Hsieh, and J.-C. Huang, "Thermal conductivity modeling of periodic porous silicon with aligned cylindrical pores," *Journal of Applied Physics*, vol. 111, p. 124329, 2012.

[12] J. Loy, J. Murthy, and D. Singh, "A fast hybrid fourier-boltzmann transport equation solver for nongray phonon transport," *Journal of Heat Transfer*, vol 135, p. 011008, 2012.

[13] A. Majumdar, "Microscale heat conduction in dielectric thin films," *Journal of Heat Transfer*, vol. 115, pp. 7-16, 1993.

where τ is the scattering time and $I^0(T_L)$ is the isotropic intensity equal to:

$$I^0(T_L) = \frac{v(\omega,p)\hbar\omega f^0(\omega,T_L)D(\omega,p)}{4\pi} \tag{2.3}$$

with $f^0(\omega,T_L)$ being the parametrization by the Bose-Einstein distribution at a given local effective temperature $T_L(r)$ expressed by 2.4[14, 15]:

$$f^0(\omega,T_L) = \left[exp\left(\frac{\hbar\omega}{\kappa_B T_L}\right) - 1\right]^{-1} \tag{2.4}$$

To compute the phonon thermal conductivity, a temperature difference ΔT is applied across a simulation domain with length L and the thermal flux is calculated from the hot contact to the cold contact. The thermal flux *J(r)* is computed by 2.5,

$$J(r) = 4\pi \sum_p \int_0^{\omega_M^p} \langle I_s \rangle d\omega \tag{2.5}$$

where $\langle x \rangle = \frac{1}{4\pi}\int_{4\pi} x d\Omega$ represents the angular average over the solid angle 4π, and ω_M^p defines the frequency cut-off for a given polarization. Considering the steady-state BTE, $\frac{\partial I}{\partial t} \approx 0$, the term $I^0(r,s,\omega,p)$ can be computed by applying the continuity equation to the heat flux ($\nabla \cdot J(r) = 0$) to Equation 2.5, resulting in 2.6:

$$\sum_p \int_0^{\omega_M^p} \frac{I_0(T)}{v\tau} d\omega = \sum_p \int_0^{\omega_M^p} \frac{\langle I \rangle}{v\tau} d\omega \tag{2.6}$$

Both $I^0(\omega,p,T)$ and $I(\omega,p)$ are frequency dependent, hence the scattering times and the phonon dispersion curve knowledge are required.

[14] J. M. Ziman, *Electrons and Phonons: The Theory of Transport Phenomena in Solids*. OUP, Oxford, 2001

[15] N. Mingo, D. A. Broido, L. Lindsay, and W. Li, *Ab initio Thermal Transport*. Springer, New York, 2014

2.3 Mean free path dependent BTE (MFP - BTE)

The BTE version considered in this work is developed by starting from Equations 2.2, 2.5 and 2.6, requiring only the bulk phonon MFP distribution as input [8].

$$K(\Lambda) = -\frac{\Lambda}{3}\sum_{p} C_s v \left(\frac{d\Lambda}{d\omega}\right)^{-1} \tag{2.7}$$

In Equation 2.7, $C_s(\omega)$ is the spectral heat capacity (i.e., the product between the volumetric heat capacity and the phonon density of states) and $\Lambda(\omega, p) = v(\omega, p)\tau(\omega, p)$ is the MFP. If the applied temperature gradient is small enough, it is reasonable to assume that all the material properties are constant throughout the simulation domain, and the variable can be defined as $\tilde{T}$ computed by Equation 2.8,

$$\tilde{T} = \frac{T_{\omega,p} - T_0}{\Delta T} = 4\pi \frac{I - I_0(T_0)}{C_s v \Delta T} \tag{2.8}$$

which represents the effective temperature $T_{\omega,p}$ with departure from T_0 associated with a given polarization, phonon frequency, and direction normalized by ΔT. For simplicity, the simulations are performed by imposing $\Delta T = 1$ and $T_0 = 0$. Including the first-order Taylor expansion of $I^0(\omega, p, T_L)$ involved in Equation 2.6 it stands:

$$I^0(\omega, p, T_L) = I^0(\omega, p, T_0) + (\frac{1}{4\pi})C_s(\omega, p)v(\omega, p)(T_L - T_0) \tag{2.9}$$

It follows the 2.10:

$$\Lambda s \cdot \nabla\tilde{T} + \tilde{T} = \gamma \int_0^\infty \frac{K}{\Lambda'^2} \left\langle \tilde{T} \right\rangle d\Lambda' \tag{2.10}$$

where it is considered $I^0(T_0)$ to be isotropic and spatially independent, $\gamma = \left[\int_0^\infty \left(\frac{K}{\Lambda^2}\right) d\Lambda\right]^{-1}$ is a material property, which for silicon is equal to $\gamma_{Si} = 2.2739 \cdot 10^{-17} m^3 W^{-1} K$. The right-hand side of Equation 2.10 is equal to $\tilde{T_L}(r) = \left(\frac{T_L(r) - T_0}{\Delta T}\right)$, which is the normalized temperature. For Equation 2.10, only the bulk MFP distribution is required as input, so that we can refer to it as MFP dependent BTE (MFP - BTE).

[8] G. Romano and J. Grossman, "Heat conduction in nanostructured materials predicted by phonon bulk mean mean free path distribution," *Journal of Heat Transfer*, vol. 137, p. 071302, 2015.

2.4 Boundary conditions

The periodic boundary conditions for the frequency dependent (FD) BTE have to be applied from the equilibrium condition to the departure of the phonon intensity. Being P the periodicity vector, it stands the 2.11:

$$I(r+P,s,\omega,p) - I_0(r+P,\omega) = I(r,s,\omega,p) - I_0(r,\omega) \tag{2.11}$$

It is relevant to note that following the MFP - BTE approach it is sufficient to apply $\Delta\tilde{T} = 1$. In 2.12 is expressed the application of the partially diffusive boundary conditions on a surface, with normal n, by stating:

$$\tilde{T}(r,s) = (1-p)\frac{1}{\pi}\int_{s'\cdot n>0} \tilde{T}(r,s')s'\cdot n d\Omega + p\tilde{T}(r,s_i) \tag{2.12}$$

where:

- $s_i = s - 2|s\cdot n|n$ is the specular direction with normal n associated to the surface;
- p is the specularity parameter, that is function of the surface roughness [16].

In conclusion, by forcing $\tilde{T} = T_p$, an arbitrary temperature T_p can be considered in the boundary conditions.

2.5 Effective thermal conductivity

Considering the MFP - BTE approach, the thermal flux can be evaluated by 2.13:

$$J = 3\Delta T\int_0^\infty \left(\frac{K(\Lambda)}{\Lambda}\right)\langle\tilde{T}(\Lambda)\rangle s d\Lambda \tag{2.13}$$

Once Equation 2.10 converges, the phonon thermal conductivity can be computed by combining Equation 2.5 with Fourier's Law obtaining the following

$$\kappa_{eff} = \frac{3L}{A}\int_\Gamma\int_0^\infty \frac{K}{\Lambda}\langle\tilde{T}_s\cdot n\rangle d\Lambda dS \tag{2.14}$$

where Γ represents the cold contact or the hot contact, and A is its area. In Equation 2.14 is used $\langle I^0 s\rangle = 0$, because I^0 is considered isotropic.

[16] J. M. Ziman, *Electrons and Phonons: The THeory of Transport Phenomena in Solids*. OUP, Oxford, 2001

It is important to mark the fact that Equations 2.10 and 2.14 do not require the use of ΔT parameter. The phonon suppression function is defined as in 2.15:

$$S(\Lambda) = \frac{3L}{\Lambda A} \int_{\Gamma} \langle \tilde{T}_s \cdot n \rangle dS \tag{2.15}$$

which led to a straightforward formula for the phonon thermal conductivity written as 2.16:

$$\kappa_{eff} = \int_0^{\infty} K(\Lambda) S(\Lambda) d\Lambda \tag{2.16}$$

Denoting the MFP distribution in the nanostructures material as $K^{nano}(\Lambda)$, the phonon suppression fuction can be formally defined by 2.17:

$$S(\Lambda) = \frac{K^{nano}\Lambda}{K(\Lambda)} \tag{2.17}$$

2.6 Ballistic limit of the MFP - BTE

The ballistic limit of the MFP - BTE is derived based on a parameter called Knudsen number which is defined as the ratio of the mean free path to the characteristic length of the material as expressed by 2.18:

$$K_n = \frac{\Lambda}{L} \tag{2.18}$$

Phonons tend to travel mostly ballistically in nanostructures where the Knudsen number $K_n \gg 1$ and in particular their effective mean free paths get closer to the characteristic length of the material. With regards to this condition, it is demonstrated that the mode temperature distributions are mean free path independent. Taking this consideration, 2.10 turns into 2.19:

$$\Lambda s \cdot \nabla \tilde{T} + \tilde{T} = const \tag{2.19}$$

The above equation is defined as the ballistic BTE equation. The positive result is that 2.19 is computationally less expensive with respect to the MFP-BTE (2.10) since phonons with different mean free paths are decoupled from each other.

2.7 Diffusive limit of the MFP - BTE

In this section is described how the diffusive limit of the MFP - BTE can be realized. This is done by considering the first spherical expansion of $\tilde{S}(r,s)$. Assuming a small perturbation $\Phi(r)\cdot s$ in the s direction, the inclusion of the spherical expansion of $\tilde{T}(r,s) = \langle\tilde{T}(r,s)\rangle + \Phi(r)\cdot s$ into 2.10, results in 2.20,

$$\tilde{T} = \gamma\int_0^\infty \frac{K}{\Lambda'^2}\left\langle\tilde{T}\right\rangle d\Lambda' - \Lambda\nabla\cdot\langle s\tilde{T}\rangle \tag{2.20}$$

where is applied $\nabla\cdot\Phi \ll \Lambda\nabla\cdot\langle s\tilde{T}\rangle$. It is noticed that the heat flux correlated $J = -K3\Delta T\nabla\langle\tilde{T}\rangle$, is made diffusive after accomplishing the following mathematical steps:

- multiplying both sides of 2.20 by $(\frac{3\Delta TK}{\Lambda})s$
- integrating both sides of 2.20 over the solid angle

The continuity equation for the thermal flux is then derived and expressed by Equation 2.21.

$$\nabla\cdot J = \frac{3\Delta TK}{\Lambda^2}\left(\gamma\int_0^\infty \frac{K}{\Lambda'^2}\langle\tilde{T}\rangle d\Lambda' - \langle\tilde{T}\rangle\right) \tag{2.21}$$

Combining equation 2.20 and equation 2.21 the diffusive equation is finally obtained and formulated in 2.22.

$$\Lambda^2\nabla^2\langle\tilde{T}\rangle - \langle\tilde{T}\rangle = \gamma\int_0^\infty \frac{K}{\Lambda'^2}\langle\tilde{T}\rangle d\Lambda' \tag{2.22}$$

The energy balance between different phonon modes is embedded on the right-hand side which represents an effective heat source. Equation 2.22 is denoted as the modified Fourier Equation (MFE).

2.8 Fourier/BTE Coupling

It is necessary a discretization about the domain simulation into a mesh for solving Equation 2.10. The mesh should have a characteristic size at least small as the smallest mean free path. This lead an expensive computation calculation in materials with wide mean free path distributions. The answer for solving this numerical problem is found in setting a threshold in the Knudsen number K_{n_D}, below which phonons are solved making use of the MFE. Equation 2.22 appears less expensive from computational point of view with respect to the MFP-BTE when defined directly in term of $\langle \tilde{T} \rangle$, but with the approximation of neglecting scattering between material boundaries and phonons. The phonon modes are decoupled between each other just like in the ballistic case and Equation 2.19 is calculated only once.

The energy conservation between all phonons modes, related to each other by means of $T_L(r)$, is guaranted by applying an iterative method in solving the equations 2.10, 2.19 and 2.22. The iterations are performed starting with a first guess for $T_L(r)$ derived by Fourier's simulation. The conventional approach for selecting the transition points between different regimes is to start by considering a fairly good estimation for the Knudsen number $K_{n_D} = K_{n_B} = 1$. It follows the tuning of both parameters, K_{n_D} towards lower values and K_{n_B} towards higher value, until reaching the convergence in the heat flux [17].

[17] G. Romano and J. Grossman, "Heat conduction in nanostructured materials predicted by phonon bulk mean mean free path distribution," *Journal of Heat Transfer*, vol. 137, p. 071302, 2015.

Chapter 3

OpenBTE

OpenBTE is an open-source tool to compute heat transport in nanostructured materials with arbitrary shape and dimensions. The code takes into account the material's Brillouin zone and diffuse boundaries for solving the space-dependent BTE.

3.1 OpenBTE Overview

OpenBTE is interfaced with two first-principle solvers: ShengBTE and TDEP. ShengBTE [18] is a package for solving the BTE for phonons with the main purpose of computing the lattice contribution to the thermal conductivity of bulk crystalline solids. TDEP [19] is a package collection of tools for finite temperature lattice dynamics. The features included are temperature dependent phonon frequencies, anharmonic free energy and lattice thermal conductivity. The finite element mesh generator used for building up the mesh is Gmsh [20].

As shown in Figure 3.1, given the material properties and geometry parameters as input, BTE/Fourier solver implemented in OpenBTE uses the First-principles and the mesh generator to provide the following outputs:

- Temperature and Thermal Flux;
- Effective Thermal Conductivity;
- Phonon Suppression Function;
- Directional Phonon Mean Free Path Distribution .

[18] W. Li, J. C. Montana, N. Katcho, and N. Mingo, "ShengBTE: A solver of the Boltzmann transport equation for phonons," *Computer Physics Communications*, vol. 185, pp. 1747-1758, 2014.

[19] O. Hellman, "TDEP", https://ollehellman.github.io, 2018.

[20] C. Geuzaine and J.-F. Remacle, "Gmsh: A 3-d finite element mesh generator with built-in pre- and post-processing facilities," *International Journal for Numerical Methods in Engineering*, vol. 79, pp. 1309 - 1331, 09 2009.

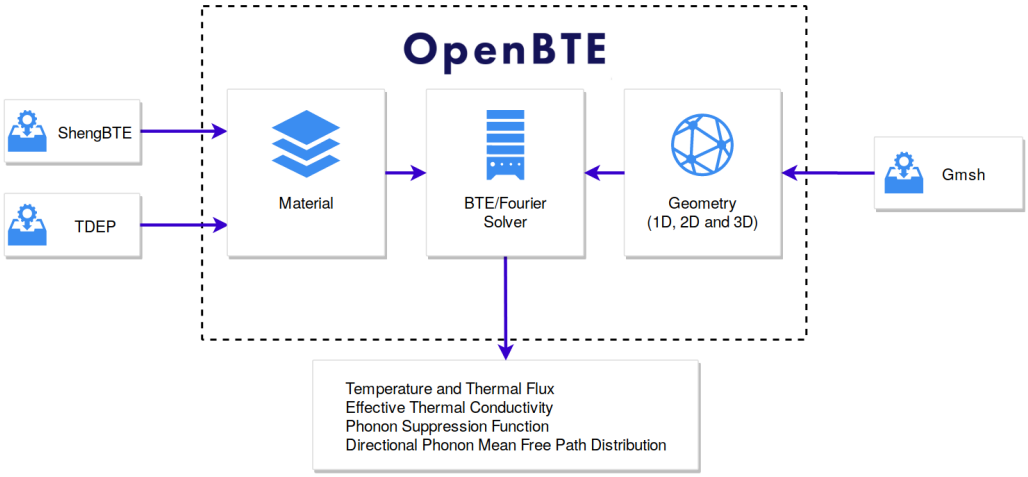

FIGURE 3.1: OpenBTE Structure

3.2 OpenBTE Model

OpenBTE main elements are the modules Material, Geometry, Solver and Plot.

3.2.1 Geometry

Within the Geometry block, flexible models are available to create porous geometries. It is possible to generate aligned configurations with square or circle lattice of length on x-, y-, z-direction, with the temperature gradient assumed to be applied along the x-direction. If the length along z-direction is not specified, the sample is considered as infinite along z and only a two-dimensional simulation is performed. The boundary conditions are assumed to be periodic along both x- and y-direction.

OpenBTE Geometry

```
openbte -g -type=porous/square\_lattice -shape=circle -porosity=0.25 -lx=10 -ly=10
```

SOURCE CODE 3.1: OpenBTE Geometry

The command in Source Code 3.1 generates the file geometry.hdf5. It is possible to set an arbitrary pore shape and configuration using the type "porous/custom" as shown in Source Code 3.2,

OpenBTE Arbitrary Pore Shape Geometry

```
echo -2.5 2.5 2.5 2.5 2.5 -2.5 -2.5 -2.5 >polygons.dat
openbte -g -type=porous/custom -polyfile polygons.dat -lx=10 -ly=10
```

SOURCE CODE 3.2: Arbitrary Pore Shape Geometry

and in that case the pores are given in a file with the following allowed format:

```
x0_a y0_a x1_a y1_a x2_a y2_a x3_a y3_a
x0_b y0_b x1_b y1_b x2_b y2_b x3_b y3_b
...
```

In the case a pore lies on the unit-cell boundary it will appear in multiple locations. Only the full description of the pore intersecting any of the boundary is needed and any periodic repetition will be performed internally.

Moreover, a bulk system can be created by using the type "bulk" as in Source Code 3.3.

OpenBTE Bulk System

```
openbte -g -type=bulk -lx=10 -ly=10 -step=1
```

Source Code 3.3: OpenBTE Bulk System

The geometry can be plotted by means of Source Code 3.4 with the output plot shown in Figure 3.2.

OpenBTE Geometry Plot

```
openbte -p -variable=map/geometry
```

Source Code 3.4: OpenBTE Geometry Plot

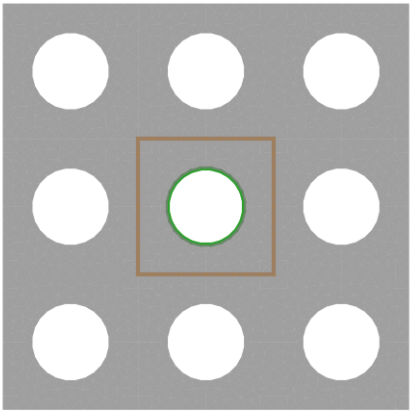

Figure 3.2: Geometry Plot

3.2.2 Material

Within the MFP - BTE, a material can be simply specified by the bulk MFP distribution. The model for a nongray material is *model=nongray* and the material file can be specified with *filename*. When the path is not specified, the file is taken from the examples provided with OpenBTE. The file format is the following:

```
mfp_1  kappa_1
mfp_2  kappa_2
...
```

The MFPs are given in meters. The BTE is solved in MFP as well as angular space. The grid is specified by the options *n_mfp* (mean free paths number), *n_theta* (azimuthal angle) and *n_phi* (polar angle: x-y plane) that must be given, as in Source Code 3.5.

OpenBTE Material

```
openbte -m -model=nongray -n\_mfp=30 -n\_phi=48 -n\_theta=16 -matfile=Si-300K.dat
```

Source Code 3.5: OpenBTE Material

With *"model=gray"* and *mfp* (in meters) it is possible to specify a single MFP to be used in the material, considering a gray material, as shown in Source Code 3.6.

OpenBTE Gray Material

```
openbte -m -model=gray -n\_phi=48 -n\_theta=16
```

Source Code 3.6: OpenBTE Gray Material

To convert data from ShengBTE to OpenBTE format the following command is required,

ShengBTE to OpenBTE Conversion

```
shengbte2openbte
```

Source Code 3.7: ShengBTE to OpenBTE Conversion

from the directory containing the file *BTE.cumulative_kappa_scalar*. The output file is named *mat.dat*.

3.2.3 Solver

The BTE solver in OpenBTE is iterative. The number of iterations is set with *max_bte_iter* = 10 as shown in Source Code 3.8. In case of zero iteration, the solver will simply perform a standard heat diffusion simulation.

OpenBTE Solver

```
openbte -s -multiscale -max\_bte\_iter = 10
```

Source Code 3.8: OpenBTE Solver

Once the simulation is finished, the file *solver.hdf5* is created.

3.2.4 Plot

Once the simulation is over it is possible to plot relevant results by means of the module *Plot*. The possible plots are:

- suppression: Phonon suppression function in the MFP space;
- map\fourier_flux|thermal_flux|fourier_temperature|bte_temperature: Map of a given variable. Currently it works only in 2D. In case of a flux, it is also possible to specify either the Cartesian axis or the magnitude;
- vtk: the *output.vtk* is created with all the relevant variables. It is recommended using Paraview. It is to be noted that for both *map* and *vtk* options, it is possible to repeat the unit-cell in x and y direction with *repeat_x* and *repeat_y*, respectively. OpenBTE handles the cell-to-node conversion internally, also including the periodicity.

The plot of heat flux magnitude is given by the Source Code 3.9 and shown in Figure 3.3.

OpenBTE Heat Flux Magnitude Plot

```
openbte -p -variable=map/bte\_flux/magnitude
```

Source Code 3.9: OpenBTE Heat Flux magnitude Plot

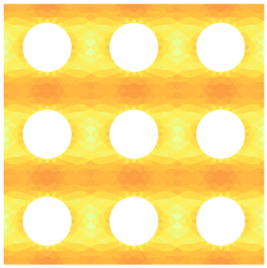

Figure 3.3: Heat Flux Magnitude Plot

The pseudotemperature is plotted by means of Source Code 3.10 and shown in Figure 3.4

OpenBTE Pseudotemperature Plot

```
openbte -p -variable=map/temperature\_bte
```

Source Code 3.10: OpenBTE Pseudotemperature Plot

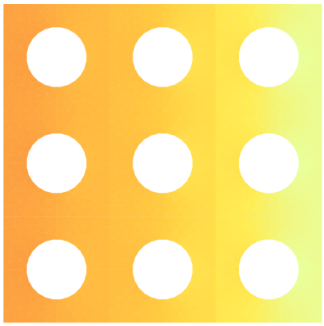

Figure 3.4: Pseudotemperature Plot

The mean free path (MFP) distribution in the porous material is plotted making use of Source Code 3.11 and shown in Figure 3.5 [21].

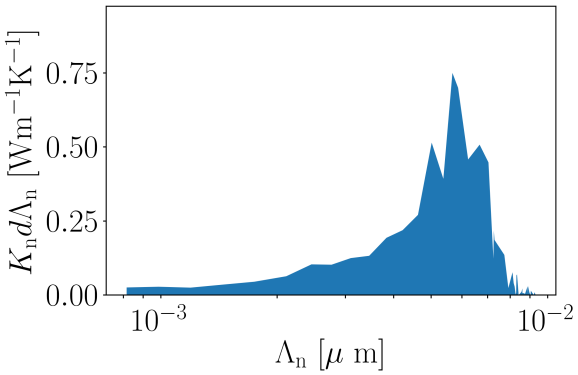

Figure 3.5: MFP Distribution Plot

OpenBTE MFP Distribution Plot

```
openbte -p -variable=distribution
```

Source Code 3.11: OpenBTE MFP Distribution Plot

[21] G. Romano, "OpenBTE: A Multiscale Solver for the Phonon Boltzmann Transport Equation," in *APS March Meeting Abstracts*, vol. 2019 of *APS Meeting Abstracts*, p. T70.321, Jan. 2019.

3.3 OpenBTE code optimization

The porosity optimization in the material nanostructures to steer the effective thermal conductivity, computed by the phonon transport solver OpenBTE, to a target value, is the main goal of this thesis. Given the input parameters, through the OpenBTE modules described in the previous sections, the thermal conductivity function κ has been calculated, for different sample porosity configurations. Input parameters are used for building up the sample model geometry and its configuration modeled as a grid. They are reported as follow:

- Material model;
- Mean free path length: mfp [nm];
- Mean free path number: n_{mfp};
- Sample length: L [nm];
- Sample slices number: N;
- Pores number: NP.

The optimization process is carried out following an inverse design model which consists of defining a target thermal conductivity value κ_0 and minimizing the deviation of the effective thermal conductivity κ, computed by OpenBTE solver, from the target one:

$$min|\kappa - \kappa_0| \tag{3.1}$$

The deviation $|\kappa - \kappa_0|$ is embedded in a function called *myfun* to let be minimized by the optimizator and coded as shown in Source Code 3.12.

Objective function

```
from openbte import Material,Solver,Geometry
Material(model='gray',mfp=[10.0],n_mfp = 40) # Material model
L = 10          # Sample length
N = 10          # Sample slices number
NP= 10          # Pores number within the sample
kappa0 = 0.3    # Target thermal conductivity
def myfun(x):
    grid = np.zeros((N,N))
    for n in range(int(len(x)/2)):
      i = x[n*2]
      j = x[n*2+1]
      grid[i,j] = 1
    geo = Geometry(model='porous/random_over_grid',
                   lx = L,ly = L, nx = N, ny = N, np = NP,
                   mesh = True, manual = True,
                   grid = grid,
                   step = L/10,
                   automatic_periodic = False)
    sol = Solver(max_bte_iter = 5,multiscale=False)
    kappa  = dd.io.load('solver.hdf5')['kappa'][-1]
    objective=abs(kappa-kappa0)      # Output to minimize
    return objective
```

Source Code 3.12: Objective function

Chapter 4

Bayesian Optimization of Machine Learning Models

4.1 Bayesian optimization

Bayes Theorem is the basis of Bayesian Optimization and a fundamental result of probability theory. The main features are recalled in what follows.

- Marginal Probability $P(A)$: the probability of an event irrespective of the outcome of other random variables;
- Joint Probability $P(A, B)$: the probability of two (or more) simultaneous events;
- Conditional Probability $P(A|B)$: the probability of one (or more) event given the occurrence of another event. It can be calculated using the joint probability:

$$P(A|B) = \frac{P(A, B)}{P(B)} \tag{4.1}$$

Another way to calculate the conditional probability is by using the other conditional probability. This approach is useful either when the joint probability is challenging to calculate or when the reverse conditional probability is available or easy to calculate. This alternative approach is referred to as Bayes Rule or Bayes Theorem and it is summarized in the following formula:

$$P(A|B) = \frac{P(B|A)P(A)}{P(B)} \tag{4.2}$$

The theorem puts the posterior probability $P(A|B)$ of a hypothesis as a product of the probability of the data given the hypothesis (likelihood) $P(B|A)$, multiplied by the probability of the hypothesis $P(A)$ (prior probability), divided by the probability of seeing the data $P(B)$ (evidence).

The unknown objective is considered as a random function (a stochastic process) on which we place a prior, capturing the beliefs about the function behaviour. The function evaluations are treated as data and used to update the prior to form the posterior distribution over the objective function.

The optimization problem corresponds to find the input value or the input set of values of an objective function that yields the lowest (or highest) output value. For simple functions in low dimensions, the optimum can be found by trying many input values and determine which is the best looking at the value of the objective function. For example a of input values can be created (Grid Search), alternatively input values can be picked randomly (Random Search). These algorithms might be suitable as long as the evaluations of the objective function are not computationally expensive.

For complex objective functions, a way to limit the number of calls of evaluation function is necessary. Random Search algorithm is actually more efficient than Grid Search algorithm for high dimensions problems, but it is still an uniformed method where the search does not use the previous results to guess the next input values to try.

The concept of using previous results for building a probabilistic model of the objective function, is implemented in Bayesan optimization, also called Sequential Model Based Optimization (SMBO), where the input parameters are mapped to a probability of loss. The probability model usually named *surrogate model*, has the facility to be easier to optimize compared to the actual objective function. The surrogate model in Bayesian methods differs depending on the criteria on which it is based, for choosing the next input values to evaluate, such as Lower Confidence Bound, Expected Improvement or Probability of Improvement.

Bayesian reasoning means updating a model based on new evidence, and after each evaluation, the surrogate model is calculated to the latest information of the actual objecive function. Bayesian Optimization methods differ also in how the surrogate function is built: common choices include Gradient Boosted Regression Trees, Gaussian Processes, Random Forest Regression.

4.2 Scikit-Optimize

In machine learning and deep learning there is an important distinction between *parameters* and *hyperparameters*. The first define the configuration variables that are internal to the model and the latter one define variables that are external to the model. Typically, hyperparameter values are assigned manually.

However, the architectural complexity of deep learning models involves a larger number of hyperparameters that need to be optimized in attempt to get the predictions previously made.

Manual search requires a strong experience, intuition and understanding of the model, and is typically an extremely time consuming task. In addition, the hyperparameter values may involve continuous values, which will have an indetermined number of possibilities. Therefore, hyperparameter optimization may appear discouraging. Luckily nowadays we can take advantage of several open source optimization libraries. One of such libraries, Scikit-Optimize [22], has been used in this work.

There are several different optimization algorithms that have been used extensively in recent years and most of them are implemented in the library used: Scikit-Optimize also includes Grid Search and Random Search to make a comparison. Grid Search consists in an exhaustive search of every combination of every setting of the parameters. This method allows to find the best settings in the discrete version of the space, but it is simply computationally expensive for large parameter spaces. Random Search, instead, is a method that samples from the full grid. Clearly it does not guarantee the same results as Grid Search, but it has been shown to be extremely effective [23] .

It has been extensively demonstrated that SMBO approaches like Bayesian Optimization are the most suitable in finding a global optimum of black-box functions, which can be also noisy and non-differentiable.

Scikit-Optimize, or Skopt, is indeed a user-friendly and efficient library used to minimize very expensive and noisy black-box functions. The library is built on top of the main libraries NumPy-SciPy and Scikit-Learn [24].
As mentioned in the previous section, Bayesian optimization process is based on finding a posterior distribution as the function to be optimized during the parameter optimization, then an acquisition function (e.g. *Expected Improvement*) is used to sample from the posterior, to find the next set of parameters to be explored.

[22] G. Louppe, "Bayesian optimization with Scikit-Optimize", *PyData Amsterdam*, 2017.

[23] J. Bergstra and Y. Bengio, "Random search for hyper-parameter optimization", *Journal of machine learning research*, vo. 13, pp. 281-305, Feb 2012

[24] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot and E. Duchesnay, "Scikit-learn: Machine learning in Python," *Journal of Machine Learning Research*, vol. 12, pp. 2825-2830, 2011.

Since Bayesian optimization determines the next points based on the available data, it is expected to carry out better configurations compared to parameter optimization approaches such as Random Search in a relatively fast fashion. In this work, Skopt has been applied and adapted to OpenBTE in all the simulations performed and described in next chapters, with the aim of minimizing the *objective function*, represented by the deviation from the *target thermal conductivity* κ_0 following the "inverse design model" [1].

The optimization problem we are interested in solving is the following:

$$x^* = arg\min_{\mathbf{x}} f(x) \tag{4.3}$$

The objective is to minimize a function *f(x)*, thus to find the value x at which the function is minimum, under the following constraints:

- f is a black box function, so that any close form of the function is unknown;
- f is very expensive to evaluate;
- only noisy observation of f may be available.

The porosity optimization problem in nanoporous materials deals with all these features, hence Bayesian Optimization is a proper approach for this problem.

In general, the first step of the optimization process is to start with a few observations of the function, i.e. few values corresponding to the red dots shown in Figure 4.1, where the red dashed line represents the true function that is unknown and the red band is the uncertainty.

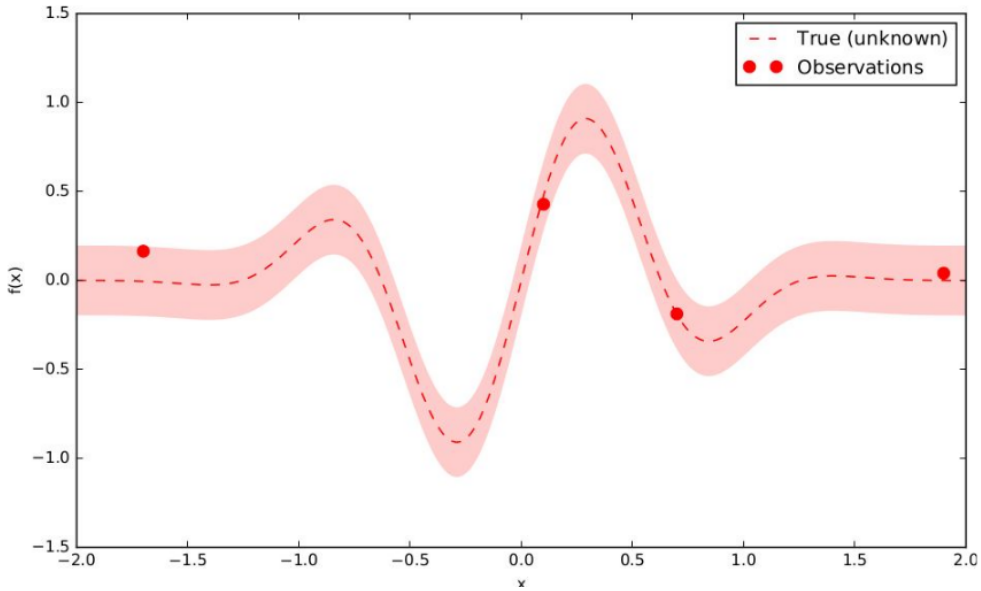

FIGURE 4.1: Step 1

[1] For the sake of clarity, in this work we are not tuning any hyperparameter. We are rather tuning the input parameters of the objective function.

The second step is to build a probabilistic model for the objective function *f*. For instance, a Gaussian Process regression is used to approximate the true unknown function by generating a family of functions as a distribution, shown by the green band in Figure 4.2, i.e., the uncertainty or the variance, of all the possible functions under the distribution.

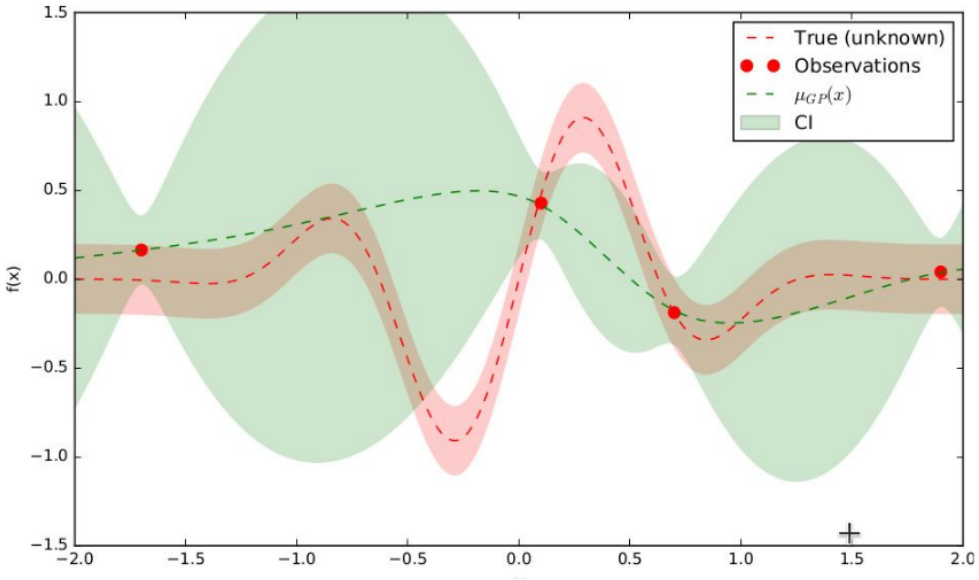

FIGURE 4.2: Step 2

The probabilistic model is therefore a distribution of functions and some properties can be evaluated, for example the mean of the distribution, represented by the green line in Figure 4.2. Once the probabilistic model is determined, a cheap utility function is considered, called *acquisition function*, that will guide the decision process of sampling the next point to evaluate. In particular, the utility function should be designed to allow the exploration of regions in the space where the probabilistic model is very uncertain: this process is called *exploration*. In regions where the uncertainty is large, some weight will be assigned in sampling those regions, but also a trade-off is necessary between *exploration* and *exploitation*, which means to sample also in regions where the minimum of the function is really located.

Once the utility function is determined, the gradient based method can be used as well to optimize the function.

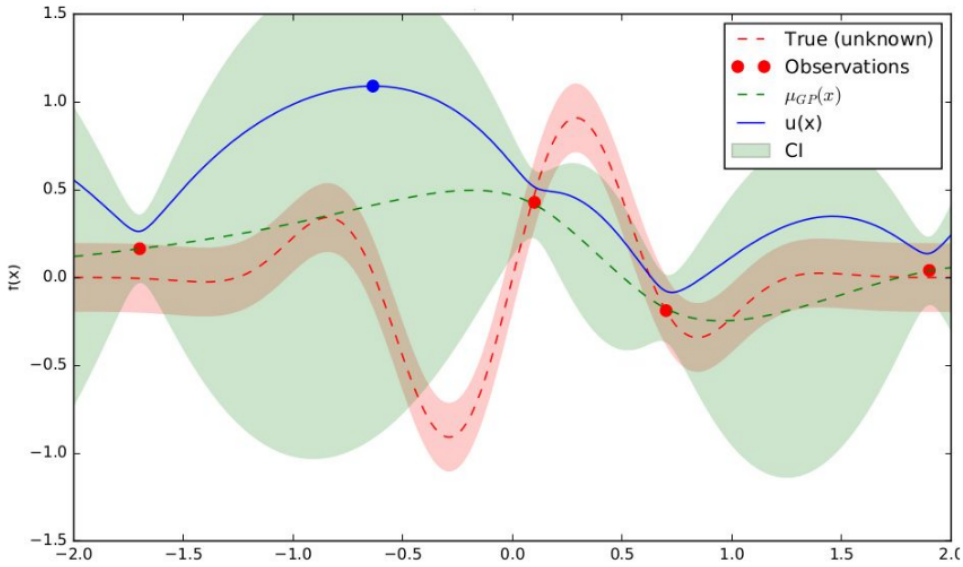

FIGURE 4.3: Step 3

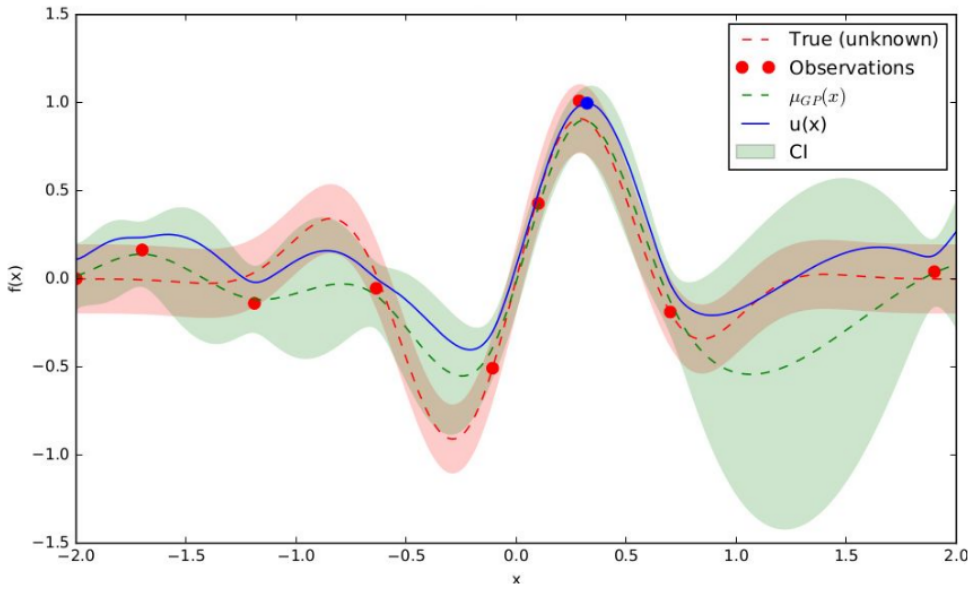

FIGURE 4.4: Step 4

As shown in the example in Figure 4.3, the utility function asks to evaluate the expected function (in blue mark), and the output will be the corresponding evaluation of the function represented by the red mark in Figure 4.4 at the same x-coordinates.

In an iteratively way it is therefore possible to evaluate further points, until reaching the point where a location very close to the minimum (or maximum) of the function is found.

Bayesian optimization loop For $t = 1 : T$

1. Given observations $(x_i, y_i = f(x_i))$ for $i = 1 : t$, build a probabilistic model for the objective f. Integrate out all possible true functions, using a surrogate model (e.g. Gaussian Process regression).
2. Optimize a cheap acquisition/utility function u based on the posterior distribution for sampling the next point.

$$x_{t+1} = arg\min_{\mathbf{x}} u(x) \tag{4.4}$$

   Exploit uncertainty to balance exploration against exploitation.
3. Sample the next observation y_{t+1} at x_{t+1}.

Acquisition functions

Acquisition functions *u(x)* specify which sample x should be tried next:

- Expected improvement (default):
  $-EI(x) = -\mathbb{E}[f(x) - f(x_t^+)]$
- Lower confidence bound:
  $LCB(x) = \mu_{GP}(x) + \kappa\sigma_{GP}(x)$
- Probability of improvement:
  $-PI(x) = -P(f(x) \geq f(x_t^+) + \kappa)$

Where x_t^+ is the best point observed so far. In most cases, acquisition functions provide knobs(e.g., k) for controlling the exploration-exploitation trade-off. Search in regions where $\mu_{GP}(x)$ is high (exploitation) and probe regions where uncertainty $\sigma_{GP}(x)$ is high (exploration).

In the Skopt library, the function f is assumed to take as input a 1D vector x represented as an array-like and to return a scalar *f(x)*. Then, the Bayesian optimization based on different surrogate moldels is implemented and carried out as follow:

Bayesian Optimization

```
from skopt import gp_minimize

res = gp_minimize(f,                  # function to minimize
                  [(-2.0, 2.0)],      # bounds on each dimension of x
                  acq_func="EI",      # acquisition function
                  n_calls=15,         # number of function evaluations
                  n_random_starts=5, # number of random initialization points
                  noise=0.1**2,       # noise level (optional)
                  random_state=1234) # the random seed
```

Source Code 4.1: Bayesian optimization

In Source Code 4.1, the optimization with one of the surrogate model implemented in Skopt and used in this work is shown, which is based

on Gaussian Process regression, called as *gp_minimize*. For analyzing the results of the optimization process, attributes of the *res* named tuple provide the following information:

- x [float]: location of the minimum;
- *fun* [float]: function value at the minimum;
- *models*: surrogate models used for each iteration;
- *x_iters* [array]: location of the function evaluation for each iteration;
- *func_vals* [array]: function value for each iteration;
- *space* [Space]: the optimization space;
- *specs* [dict]: parameters passed to the function.

These attributes have been used in the simulations of this work to visually analyze the results of the minimization, such as the very intuitive convergence trace shown in Figure 4.5, as output of Source Code 4.2 :

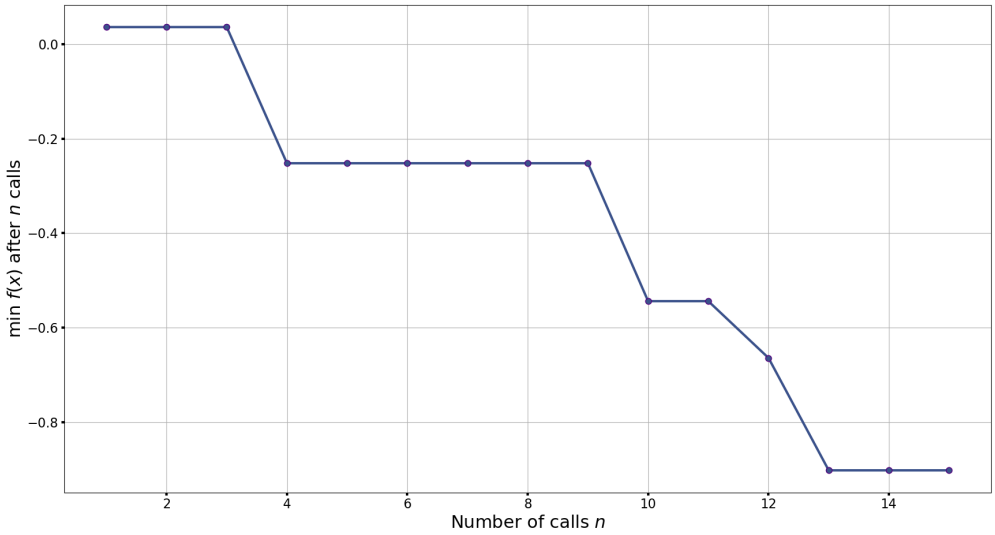

FIGURE 4.5: Convergence Plot

Convergence Plot

```
from skopt.plots import plot_convergence
plot_convergence(res);
```

SOURCE CODE 4.2: Convergence Plot

4.3 Surrogate models used for optimization

In numerous applications the objective function is expensive to evaluate. In these cases, the general approach is to create a simpler surrogate model of the objective function which is cheaper to evaluate and it will be used instead to solve the optimization problem. In addition, due to the high cost of evaluating the objective function, it is suggested to use an iterative approach. Iterative optimizers work by iteratively requesting evaluations of the function at a sequence of points in the domain $x_1, x_2, x_3, ... \in X$. Through these evaluations the optimizer is able to build up an image of the function. At any time, the iterative optimizer will be able to state its best approximation to the true value of the points in the domain. Bayesian Optimization uses a prior model on the space of possible target functions:

$$f = (f_1, f_2, ..., f_N) \tag{4.5}$$

At points:

$$X_N = (x_1, x_2, ..., x_N) \tag{4.6}$$

The surrogate model is trained using **N** known evaluations of the function f at any point X_N.

There are many approaches used for building up the surrogate model such as Neural Networks, Random Forest and Gaussian Processes.

The surrogate models used in this work for the optimization process are the following:

- Gaussian process;
- Gradient boosted regression trees;
- Tree based regression.

4.3.1 Random search

A Random Search algorithm may be called a Monte Carlo method or a stochastic algorithm. Random Search algorithms are useful for ill-structured global optimization problems, where the objective function is eventually nonconvex, nondifferentiable, and possibly discontinuous over a continuous, discrete, or mixed continuous-discrete domain. A global optimization problem with continuous variables may contain several local optima or stationary points. A problem with discrete variables, such as the case study object of this thesis, falls into the category of combinatorial optimization.

In contrast to deterministic methods, which typically guarantee asymptotic convergence to the optimum, Random Search algorithms ensure convergence in probability. The tradeoff is in terms of computational effort. These algorithms are used because they can provide a relatively good solution quickly and easily. Random Search methods have been shown to have a potential to solve large scale problems efficiently in a way that is not possible for deterministic algorithms [23].

Another advantage of Random Search methods is that they are relatively easy to implement on complex problems with black-box function evaluations, because they usually only rely on function evaluations, rather than gradient. They can be coded quickly, and applied to a broad class of global optimization problems. A disadvantage of these algorithms is that they are customized to each specific problem largely through trial and error.

Different classifications have been advocated for Random Search algorithms including the "two-phase method", where a global search phase and a local search phase are defined. The global phase is conceived as an exploration phase pointed at exploring the whole feasible region, while the local phase is pointed at exploiting local information, hence identified as an exploitation phase.

Another classification is based on grouping algorithms between instance-based and model-based, where instance-based methods address the formulation of new candidate points based on the current point or population of points. Model-based methods instead, rely on an explicit sampling distribution and update parameters of the probability distribution [25].

[23] J. Bergstra and Y. Bengio, "Random search for hyper-parameter optimization", *Journal of machine learning research*, vo. 13, pp. 281-305, Feb 2012

[25] Z. B. Zabinsky, *"Random Search Algorithms"*. American Cancer Society, 2011

The global optimization problem is defined as,

$$\min_{\mathbf{x}\in X} f(x) \tag{4.7}$$

where $\mathbf{x}$ is a vector of n decision variables in a n-dimensional feasible region X. The goal is to find a value for $\mathbf{x}$ that minimizes f. The global optimal solution to the global optimization problem is denoted by (x^*, y^*) where:

$$x^* = arg\min_{\mathbf{x}} f(x) \quad \text{and} \quad y^* = f(x^*) = \min_{\mathbf{x}} f(x) \tag{4.8}$$

Being the optimization problem of this work a discrete problem, and the feasible set X is nonempty and finite, a global solution exists. The existence of a unique minimum at x^* is not required. When multiple optimal minima exist, x^* is assumed to be an arbitrary fixed global minimum. A generic random search algorithm is characterized by a sequence of iterates X_k, on iteration $k = 0, 1, ..., N$, which may depend or not on previous points. The current iterate X_k represents a sequence of discrete values. The *Generic Random Search Algorithm* is articulated in the following steps:

1. Perform an algorithm initialization parameters, initial points $x_0 \subset X$ with iteration index $k = 0$.
2. Generate a sequence of candidate points $w_{k+1} \subset X$ according to a sampling distribution.
3. Update x_{k+1} points based on the candidate points w_{k+1} and the algorithm parameters. Strictly improving algorithm has a simple procedure of updating the current points only if the candidate points are improving:

$$x_{k+1} = \begin{cases} w_{k+1} & \text{if } f(w_{k+1}) < f(x_k) \\ x_k & otherwise \end{cases} \tag{4.9}$$

4. Stop the iteration process, otherwise update k and return to Step 1.

The Generic Random Search Algorithm is based on two procedures: the initialization in Step 2 that creates the candidate points and the update process in Step 3.

For the optimization of the objective function *myfun* with Skopt using Random Search, the user has to call the function *dummy_minimize*. An example of its application in the optimizing process is shown in Source Code 4.3.

Optimization using Random Search model

```
from skopt import dummy_minimize
from skopt.plots import plot_convergence
from skopt import expected_minimum
from skopt import callbacks
from skopt.callbacks import CheckpointSaver
input_list = [[0,N-1]]*2*NP       # Input vector dimension
res_dummy = dummy_minimize(myfun,             # Objective function to minimize
                           input_list,        # Bounds on each dimension
                           n_calls=100,       # Number of myfun evaluations
                           callback=[checkpoint_saver]) # Callbacks list
plot_convergence(res_dummy)       # Convergence plot
```

Source Code 4.3: Random search - dummy_minimize

where the parameters function are defined as follow:

- *myfun*: the function to minimize. It takes a single list of parameters and return the objective value;
- *input_list*: the list of search space dimensions. It is defined as (lower_bound, upper_bound);
- *n_calls*: the number of calls to *myfun* to find the minimum;
- *callback*: the result is called after each call to *myfun*.

4.3.2 Gaussian Processes

One of the approaches used for building up the surrogate model is the Gaussian Processes (GPs). GPs provide a rich and flexible class of non-parametric statistical models over function spaces with domains that can be continuous, discrete, mixed, or even hierarchical in nature. Furthermore, the GPs provides not just information about the expected value of *f*, but importantly also about the uncertainty around that value. The idea behind Gaussian Processes is to approximate the cost function for a set of observed values at some points X_k, assuming the function values following a multivariate Gaussian Process with a prior distribution.
The latter is given by Equation 4.10:

$$p(x;\mu,\Sigma) = \frac{1}{(2\pi)^{\frac{n}{2}}\,|\Sigma|^{\frac{1}{2}}} e^{(-\frac{1}{2}(x-\mu)^{\tau}\Sigma^{-1}(x-\mu))} \tag{4.10}$$

$|\Sigma|$ is the determinant of the covariance matrix Σ with coefficients expressed in terms of a correlation function (kernel) $K_{mn} = K(x_m, x_n, \theta)$, so that the covariance of the function values are given by GP kernel between the parameters. The hyperparameters θ of the kernel are calibrated according to a maximum likelihood principle. The covariance matrix is chosen to reflect a prior assumption of the function and therefore the choice of the kernel will have a significant impact on the correctness of the regression. Then a smart choice regarding the next parameter to evaluate can be made by the acquisition function which is much quicker to evaluate.

The total number of evaluations (*n_calls*) are performed in two ways: if x_0 is provided but not $f(x_0)$, the elements of x_0 are first evaluated, followed by a number of random starts evaluations (n_random_starts). Finally, (n_calls - len(x_0) - n_random_starts) evaluations are made guided by the surrogate model. If x_0 and $f(x_0)$ are both provided, then a number of random starts evaluations are first made, then (n_calls - n_random_starts) subsequent evaluations are made guided by the surrogate model.

Bayesian Optimization using Gaussian Processes is performed by incorporating a GPs prior model on the domain of possible target functions *f*. By updating the model, a function evaluation is reported, and a Bayesian optimization routine keeps a posterior model of the target function *f*. Through mathematical transformations and using the conditional probability rule it is possible to estimate the posterior distribution $P(x_k + 1)$ and express $f(x_k + 1)$ as a function of $f(x_k)$ and Σ with an uncertainty. This posterior model is the Gaussian Process surrogate model for the function *f*.

The pseudo code for a Bayesian Optimization routine with a GPs prior is:

Initialisation:

1. Place a GPs prior on f;
2. Observe f at n_0 points according to an initial space-filling experimental design;
3. Set n at n_0.

While $n \leq N$ *do*:

4. Update the posterior probability distribution on f using all available data;
5. Identify the maximizer x_n of the acquisition function over X, where the acquisition function is calculated using the current posterior distribution;
6. Observe $f_n = f(x_n)$;
7. Increment n.

End while
Return either the point evluated with the largest *f(x)* or the point with the largest posterior mean.

Assuming the noisy function example in Figure 4.6,

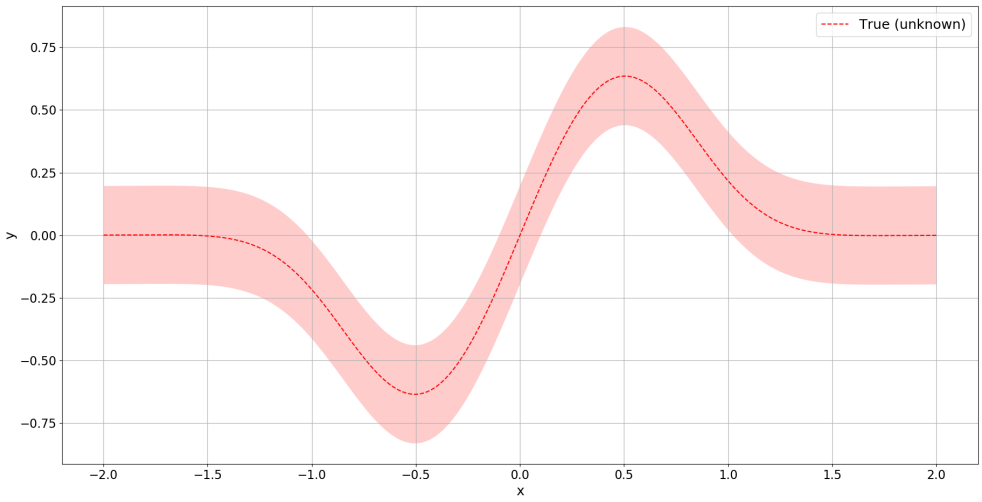

FIGURE 4.6: GP True function

Bayesian optimization based on gaussian process regression is carried out assuming a standard acquisition function used in the simulations: the *Expected Improvement Criterion (EI)*. It is considered for the improvement, for any given point in $x \in X$, the value of f at x over the best value of f

yet seen, given that the function f at x is indeed lowerer than the best value of f yet seen. When finding the maxima of f, EI can be written as:

$$EI(x) = E(max(f(x) - f^*, 0)) \tag{4.11}$$

The GP optimization process is implemented in Skopt optimizer and called by *gp_minimize* as shown in Source Code 4.5.

Bayesian optimization Gaussian Processes

```
from skopt import gp_minimize

res = gp_minimize(f,                  # the function to minimize
                  [(-2.0, 2.0)],      # the bounds on each dimension of x
                  acq_func="EI",      # the acquisition function
                  n_calls=15,         # the number of evaluations of f
                  n_random_starts=5,  # the number of random initialization points
                  noise=0.1**2,       # the noise level (optional)
                  random_state=1234)  # the random seed
```

Source Code 4.4: Gaussian Process - gpminimize

Figure 4.7 illustrates the evolution of the surrogate model and its first 5 interactions. The acquisition function improves its knowledge after each iteration of the underlying function that it is trying to minimize, following the 5 random points.

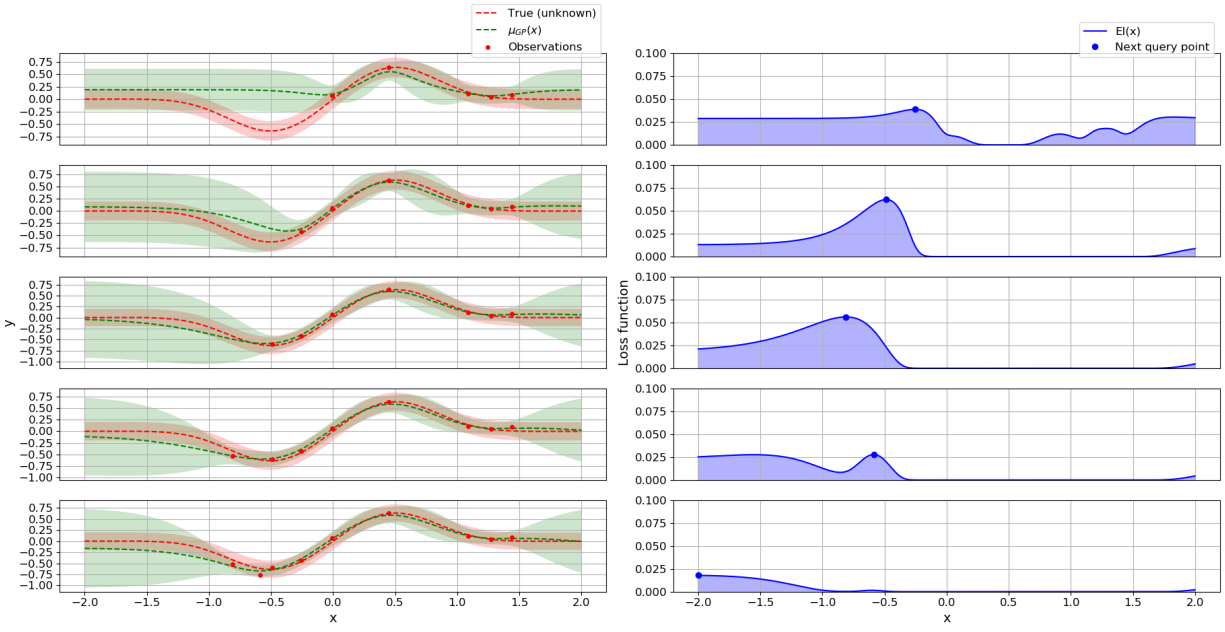

Figure 4.7: An illustration of the BO procedure over 5 iterations [Osborne, M.A., 2010]

The first column displays the following:

- The true function;
- The approximation of the true function by the GP model;
- The confidence of the GP about the true function.

In the second column are reported the acquisition function values after every fitting performed by the surrogate model. Depending on the minimizer used to minimize the acquisition function, potentially the global minimum will not be chosen, but a local minimum instead. In the neighborhood points of the points previously evaluated, the variance drop down to zero. Finally, by increasing the number of points evaluated, the GP model tends to improve the approximation of the true function. The final few points are clustered around the minimum because the GP does not gain anything more by further exploration, as shown in Figure 4.8.

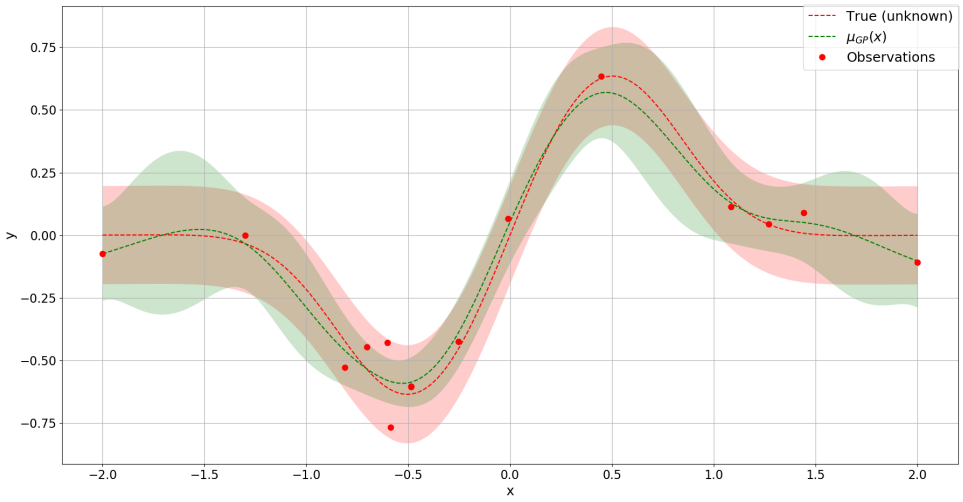

FIGURE 4.8: Gaussian Process - Optimization results

An example of its application in the optimizing process performed in this work is shown in Source Code 4.5.

Optimization using GPs model

```
from skopt import gp_minimize
from skopt.plots import plot_convergence
from skopt import expected_minimum
from skopt import callbacks
from skopt.callbacks import CheckpointSaver
input_list = [[0,N-1]]*2*NP     # Input vector dimension
res_gp = gp_minimize(myfun,             # Objective function to minimize
                     input_list,        # Bounds on each dimension
                     acq_func="EI",     # Acquisition function
                     n_calls=100,       # Number of myfun evaluations
                     callback=[checkpoint_saver]) # Callbacks list
plot_convergence(res_gp)        # Convergence plot
```

SOURCE CODE 4.5: Gaussian Process - gp_minimize

4.3.3 Ensemble methods

Ensemble methods combine the predictions of several base estimators built with a given learning algorithm in order to improve generalizability or robustness over a single estimator. Ensemble methods are usually distinguished in two families [26]:

- Averaging methods: where the driving principle is to build several estimators independently and then to average their predictions. On average, the combined estimator is usually better than any of the single base estimator because its variance is reduced. Examples are Bagging methods anf Forests of randomized trees.
- Boosting methods: where the base estimators are built sequentially and one tries to reduce the bias of the combined estimator. The motivation is to combine several weak models to produce a powerful ensamble. Examples are Gradient boosted regression trees and Random forest.

4.3.4 Gradient boosted regression trees

Gradient Tree Boosting or Gradient Boosted Regression Trees is a generalization of boosting to arbitrary differentiable loss functions. Gradient Boosting of Regression Trees (GBRT), was initially developed by Friedman (2001) [27], for estimating the car-following model, and it is actually used in a variety of areas including Web search ranking and ecology. GBRT is an accurate and effective off-the-shelf procedure that can be used for both regression and classification. The key goal in the algorithm is to, at each step, fit a regression tree to the difference between the observed response and the aggregated prediction of all learners grown previously. As indicated by the name of the algorithm, the decision trees algorithm with a fixed size is chosen as base learner, usually called weak learner.

In boosting methods algorithms, weak learners are trained iteratively and in a stage-wise fashion so as to find a model that reduces the bias and variance of prediction. In each iteration, a strength weight is assigned to the weak learner that implies its prediction error rate. Meanwhile, each training instance is reweighted by how incorrectly it was classified. The method of weighting training data and base learners distinguishes between boosting algorithms such as the adaptive boosting, and gradient boosting. The final model is the sum of the weighted weak learners. It is a surrogate model used to evaluate the expensive function.

[26] Scikit-learn developers, "Ensamble methods". https://scikit-learn.org/stable/modules/ensemble.html, 2019.

[27] J. H. Friedman, "Greedy function approximation: A gradient boosting machine," *The Annals of Statistics*, vol. 29, pp. 1189-1232, 10 2001.

The model is improved by sequentially evluating the expensive function at the next best point, finding the minimum of the cost function with as few evaluations as possible.

The advantages of GBRT are:

- Possibility to manage heterogeneous data;
- Robustness to anomalous values in output space;
- Predictive capacity.

The disadvantages of GBRT are:

- Scalability (due to the sequential nature of boosting it can hardly be parallelized);
- Experience required in tuning;
- Time consuming in training.

The module Skopt used in this work provides methods for both classification and regression via gradient boosted regression trees. The latter one is used for optimizing the objective function in the inverse design model.

GBRT considers additive models of the following form [26]:

$$f(x) = \sum_{k=1}^{K} \gamma_k h_k(x) \tag{4.12}$$

Where $h_k(x)$ are the basis functions which are usually called weak learners in the context of boosting. Gradient Tree Boosting uses decision trees of fixed size as weak learners. Decision trees have a number of abilities that make them valuable for boosting, namely the ability to handle data of mixed type and the ability to model complex functions.

Similar to other boosting algorithms, GBRT builds the additive model in a greedy fashion expressed by Equation 4.13,

$$f_k(x) = f_{k-1}(x) + \gamma_k h_k(x), \tag{4.13}$$

where the newly added tree h_k tries to minimize the loss L, given the previous ensemble f_{k-1}:

$$h_k = arg\,\min_{h} \sum_{i=1}^{n} L(y_i, f_{k-1}(x_i) + h(x_i)) \tag{4.14}$$

[26] Scikit-learn developers, "Ensamble methods". https://scikit-learn.org/stable/modules/ensemble.html, 2019.

The initial model f_0 is problem specific. For least-square regression is usually chosen the mean of the target values. GBRT attempts to solve this minimization problem numerically via steepest descent. The steepest descent direction is the negative gradient of the loss function evaluated at the current model F_{m-1} which can be calculated for any differentiable loss function:

$$f_k(x) = f_{k-1}(x) - \gamma_k \sum_{i=1}^{n} \nabla_f L(y_i, f_{k-1}(x_i)) \tag{4.15}$$

Where the step length γ_k is chosen using line search:

$$\gamma_k = arg\,\min_{\gamma} \sum_{i=1}^{n} L(y_i, f_{k-1}(x_i) - \gamma \frac{\partial L(y_i, f_{k-1}(x_i))}{\partial f_{k-1}(x_i)} \tag{4.16}$$

The algorithms for regression and classification only differ in the concrete loss function used.

For regression, the loss functions supported are the following:

- Least squares ('ls'): the natural choice for regression due to its superior computational properties. The initial model is given by the mean of the target values.
- Least absolute deviation ('lad'): a robust loss function for regression. The initial model is given by the median of the target values.
- Huber ('huber'): another robust loss function that combines least squares and least absolute deviation; it is used alpha to control the sensitivity with regards to outliers.
- Quantile ('quantile'): a loss function for quantile regression. The quantile is specified by alpha in between $0 < \alpha < 1$. This loss function can be used to create prediction intervals.

In the sequential optimization using GBRT, the total number of evaluations (n_calls) are performed in two ways: if x_0 is provided but not $f(x_0)$, the elements of x_0 are first evaluated, followed by a number of random starts evaluations (n_random_starts).

Finally, (n_calls - len(x_0) - n_random_starts) evaluations are made guided by the surrogate model. If x_0 and $f(x_0)$ are both provided, a number of random starts evaluations are first made, then (n_calls - n_random_starts) subsequent evaluations are made guided by the surrogate model.

An example of its application in the optimizing process performed in this work is shown in Source Code 4.6.

Optimization using Gradient boosted regression trees model

```
from skopt import gbrt_minimize
from skopt.plots import plot_convergence
from skopt import expected_minimum
from skopt import callbacks
from skopt.callbacks import CheckpointSaver
input_list = [[0,N-1]]*2*NP       # Input vector dimension
res_gbrt = gbrt_minimize(myfun,             # Objective function to minimize
                         input_list,        # Bounds on each dimension
                         acq_func="EI",     # Acquisition function
                         n_calls=100,       # Number of myfun evaluations
                         callback=[checkpoint_saver]) # Callbacks list
plot_convergence(res_gbrt)        # Convergence plot
```

Source Code 4.6: GBRT - gbrt_minimize

4.3.5 Forests of randomized trees

It is a sequential optimization using decision trees. The Skopt module includes two averaging algorithms based on randomized decision trees [28].

A tree based regression model is used to sequentially evaluate the expensive function at the next point. The base estimator is the regressor to use as surrogate model. It can be either:

- *Random Forest Trees* (RF);
- *Extremely Randomized Trees* (ET, as default in Scikit-Optimize).

Both algorithms are perturb-and-combine techniques [29] specifically designed for trees. Either of these regressors are tested for the optimization goal of this work.

In *Random Forest Trees*, each tree in the ensemble is built from a sample drawn with replacement (i.e., a bootstrap sample) from the training set. In addition, when splitting a node during the construction of the tree, the split that is chosen is no longer the best split among all features (or a random subset of features). Instead, the split that is picked is the best split among a random subset of the features. As a result of this randomness, the bias of the forest usually slightly increases (with respect

[28] Scikit-learn developers, "Forests of randomized trees". https://scikit-learn.org/stable/modules/ensemble.html#forests-of-randomized-trees, 2019.

[29] L. Breiman, "Arcing classifier (with discussion and a rejoinder by the author)," *The Annals of Statistics*, vol. 26, pp. 801-849, 06 1998.

to the bias of a single non-random tree) but, due to averaging, its variance also decreases, usually more than compensating for the increase in bias, hence yielding an overall better model.

An example of its application in the optimizing process done in this work is shown in Source Code 4.7.

Optimization using Random Forest Trees model

```
from skopt import forest_minimize
from skopt.plots import plot_convergence
from skopt import expected_minimum
from skopt import callbacks
from skopt.callbacks import CheckpointSaver
input_list = [[0,N-1]]*2*NP       # Input vector dimension
res_forestRF = forest_minimize(myfun,             # Objective function to minimize
                               input_list,          # Bounds on each dimension
                               base_estimator="RF", #Surrogate model regressor
                               acq_func="EI",       # Acquisition function
                               n_calls=100,         # Number of myfun evaluations
                               callback=[checkpoint_saver]) # Callbacks list
plot_convergence(res_forestRF)    # Convergence plot
```

Source Code 4.7: Random Forest Trees - forestRF_minimize

In *Extremely Randomized Trees*, randomness goes one step further in the way splits are computed. As in random forests, a random subset of candidate features is used, but instead of looking for the most discriminative thresholds, thresholds are drawn at random for each candidate feature and the best of these randomly-generated is picked as the splitting rule. This usually allows to reduce the variance of the model a bit more, at the expense of a slightly greater increase in bias.

An example of its application in the optimizing process realized in this work is shown in Source Code 4.8.

Optimization using Extremely Randomized Trees model

```
from skopt import forest_minimize
from skopt.plots import plot_convergence
from skopt import expected_minimum
from skopt import callbacks
from skopt.callbacks import CheckpointSaver
input_list = [[0,N-1]]*2*NP       # Input vector dimension
res_forestET = forest_minimize(myfun,             # Objective function to minimize
                               input_list,          # Bounds on each dimension
                               base_estimator="ET", #Surrogate model regressor
                               acq_func="EI",       # Acquisition function
                               n_calls=100,         # Number of myfun evaluations
                               callback=[checkpoint_saver]) # Callbacks list
plot_convergence(res_forestET)    # Convergence plot
```

Source Code 4.8: Extremely Randomized Trees - forestET_minimize

Chapter 5

Inverse Design Model

5.1 Phonon transport across nano-constriction

In this section is reported the analysis of phonon transport across nano-constriction performing digital simulations with the aid of OpenBTE solver. The study is concerned in particular with 2-D simulations phonon transport, in x-y dimensions, carried out on a gray material sample of square shape with dimensions equal to 40 nm along x-direction and 40 nm along y-direction, considering periodic boundary conditions applied. A nano-constriction exists within the sample, due to the presence of porosity, generated by a fixed number of pores placed on two opposite ends of the sample. Having set the necessary input parameters, the output produced by the simulation process consists in the ratio K between the BTE coefficient and the Fourier coefficient, which is the parameter that gives the information about the phonon effects in the sample. The aim is to obtain and observe how the phonon transport phenomenon behaves by reducing its transit through the nano-constriction, in the way of decreasing the distance d between the two pores batches but keeping constant the total porosity density, starting from the initial sample configuration shown in Figure 5.1.

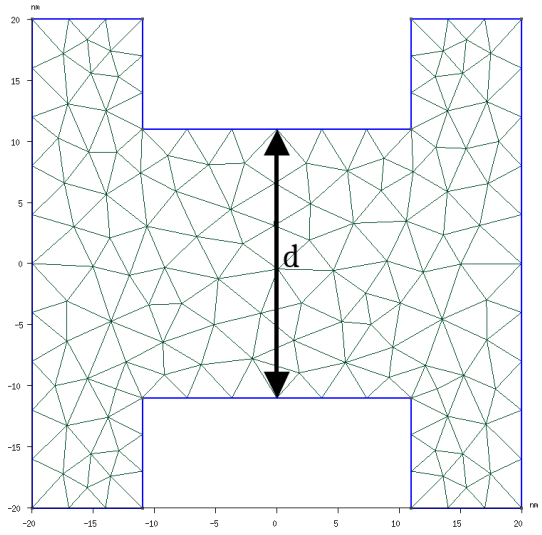

FIGURE 5.1: Nano-constriction initial configuration ($d = 22$ nm)

The material enclosed within the distance d is reduced until the trivial configuration of a pores junction is obtained leading to a material non-continuity. The unit cells that assemble the sample, given the input dimensions previousy mentioned, are 400 in number with a single cell measuring 4 square nanometer. The number of pores considered is equal to 99, representing the total porosity of the tested specimen leading to a porosiy density of about 0.25 (0.2475). In Figure 5.2 is shown the starting configuration analyzed, represented by the mesh of the sample (top-left), the thermal flux behaviour (top-right) and the resuling plot of the two merged together (bottom). In the starting configuration, the gap between the two pores batches is equal to $d = 22$ nm.

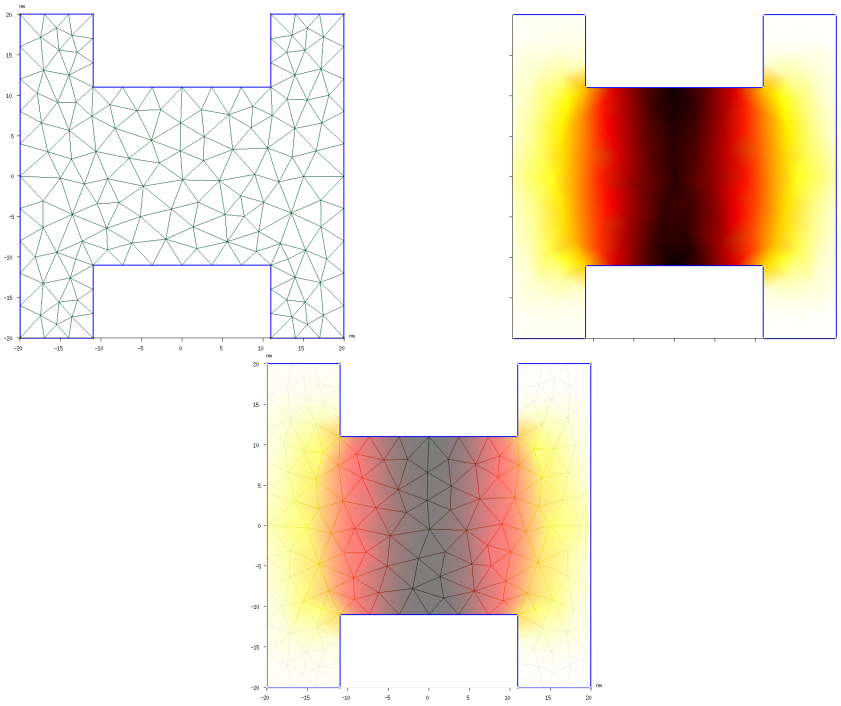

FIGURE 5.2: Mesh geometry and thermal flux plot ($d = 22$ nm)

Further simulations are performed by decreasing the starting distance d from 22 nm up to a material non-continuity, as listed in the following steps:

- $d = 22$ nm: starting configuration;
- $d = 18$ nm;
- $d = 14$ nm;
- $d = 10$ nm;
- $d = 6$ nm;
- $d = 2$ nm;
- $d = 0$ nm: materialy non-continuity configuration.

In Figure 5.3 some of the configurations tested are shown, represented by the thermal flux plots merged with the geometry meshes.

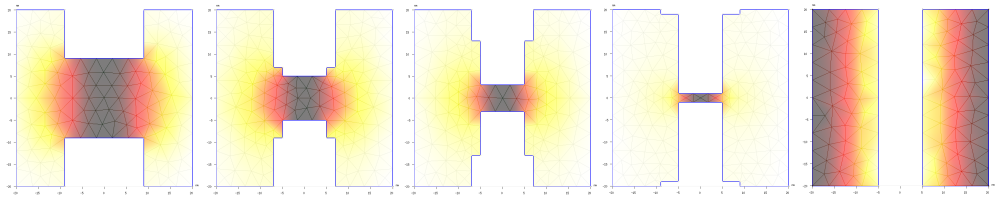

FIGURE 5.3: Thermal flux plots (from left, $d = 18$ nm, $d = 10$ nm, $d = 6$ nm, $d = 2$ nm, $d = 0$ nm)

It is expected that the reduction of the distance d leads to a decrease of the thermal ratio K. This is confirmed by the results of the simulations performed and summarized in Table 5.1.

Simulation step	Nano-constriction d [nm]	Thermal ratio K
1	22	0.5020
2	18	0.4455
3	14	0.3832
4	10	0.3159
5	6	0.2301
6	2	0.1162
7	0	0

TABLE 5.1: Thermal ratio as function of nano-constriction d

The results can be visualized in Figure 5.4, where is plotted the thermal ratio as a function of the nano-constriction d.

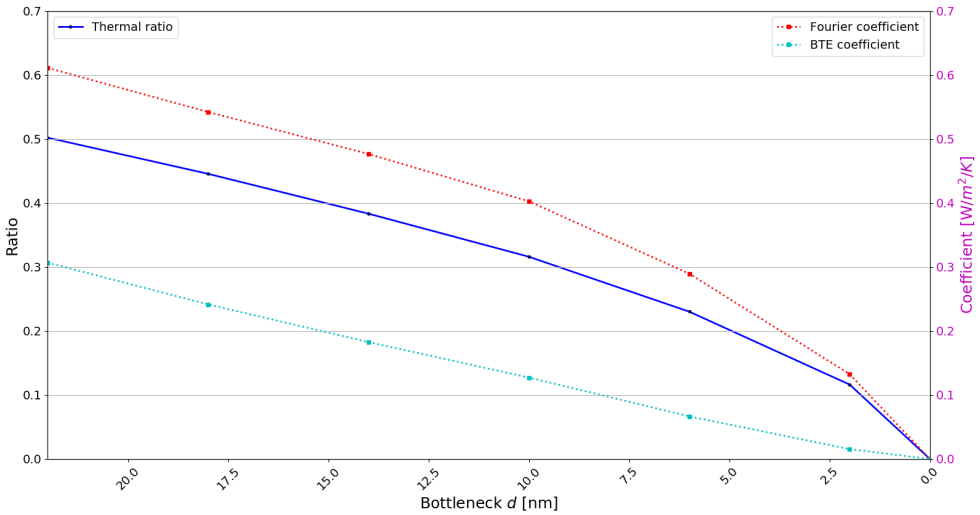

FIGURE 5.4: Thermal ratio as a function of nano-constriction d

It is observed a continuous reduction in the thermal ratio K from a value of 0.5020 related to the starting configuration up to a zero value associated to a material discontinuity in the sample, which represent the trivial condition.

In Figure 5.4, together with the thermal ratio are plotted the coefficients defining its value, namely BTE coefficient and Fourier coefficient; respectively the numerator and the denominator of the ratio. The plotted results of the nano-constriction simulations exhibit an evident different evolution for the two coefficients curves while decreasing the nano-constriction distance d. In this case it is possible to appreciate the difference between the two thermal regimes for which the BTE coefficient converges to zero faster than the Fourier coefficient and this is mainly due to the classical size effect described in Section 2.1.

The BTE coefficient curve in Figure 5.4 exhibits a linear decrease starting from a lower value with respect to the Fourier coefficient curve, until the null value is reached. The same linear decrease is shown by the Fourier coefficient curve but for low distance d values the slope of the curve increases, towards the non-continuity condition due to the generation of an infinite pores column.

In Table 5.2 are detailed the values assumed by the two coefficients depending on the nano-constriction distance d.

Simulation step	Nano-constriction d[nm]	BTE coefficient [W/m/K]	Fourier coefficient [W/m/K]
1	22	0.3067	0.6110
2	18	0.2415	0.5421
3	14	0.1826	0.4765
4	10	0.1272	0.4026
5	6	0.0666	0.2894
6	2	0.0154	0.1325

TABLE 5.2: BTE coefficient and Fourier coefficient - nano-constriction d

In Source Code 5.1 is reported the first step body code of the simulation carried out with $d = 22$ nm.

Nano-constriction simulation body code

```
#-----------------------INPUT PARAMETERS--------------------------#
L = 40
N= 20
NP = 99
#---------------------------BODY CODE-------------------------------#
def myfun(x):
    grid = np.zeros((N,N))

    for n in range(int(len(x)/2)):
      i = x[n*2]
      j = x[n*2+1]
      grid[i,j] = 1

    geo = Geometry(model='porous/random_over_grid',
                   lx = L,ly = L,
                   nx = N, ny = N,np=NP,
                   mesh = True,
                   manual = True,
                   grid = grid,
                   step = L/10,
                   automatic_periodic = True)

    sol = Solver(max_bte_iter = 5,multiscale=False)
    kappa  = dd.io.load('solver.hdf5')['kappa'][-1]

    return kappa
#-------------------------FIRST STEP------------------------------#
x1 = [5,0,6,0,7,0,8,0,9,0,10,0,11,0,12,0,13,0,14,0,15,0,\
      5,19,6,19,7,19,8,19,9,19,10,19,11,19,12,19,13,19,14,19,15,19,\
      5,1,6,1,7,1,8,1,9,1,10,1,11,1,12,1,13,1,14,1,15,1,\
      5,18,6,18,7,18,8,18,9,18,10,18,11,18,12,18,13,18,14,18,15,18,\
      5,2,6,2,7,2,8,2,9,2,10,2,11,2,12,2,13,2,14,2,15,2,\
      5,17,6,17,7,17,8,17,9,17,10,17,11,17,12,17,13,17,14,17,15,17,\
      5,3,6,3,7,3,8,3,9,3,10,3,11,3,12,3,13,3,14,3,15,3,\
      5,16,6,16,7,16,8,16,9,16,10,16,11,16,12,16,13,16,14,16,15,16,\
      5,4,6,4,7,4,8,4,9,4,10,4,11,4,12,4,13,4,14,4,15,4]
d1 = 22
y1 = myfun(x1)
```

Source Code 5.1: Nano-constriction simulation body code

5.2 Inverse Design Optimization

The inverse design model approach adopted is based on setting in advance for the simulation process a desired output target value, and by tuning the paramters involved in the computation, to obtain a resulting output as much closer to the target one. In particular, since the purpose of this work is investigating the phonon transport across porous nanostructures, the output of the simulations performed is the effective thermal conductivity κ computed by the use of OpenBTE solver and its deviation from thr target thermal conductivity defined as κ_0.

The thermal conductivity is influenced by the porosity density present in the material sample, and in particular by the shape of the pores, their dimensions and the positions they occupy within the material, together with the sample dimensions.

It is easily intuitive to expect an infinite number of possible combinations that can be found by manually varying each of these parameters resulting in infinite possible geometric configurations. For this reason it comes in handy the use of *Bayesian Optimization* algorithms for the simulation processes performed in this work.

In particular, following the inverse design approach, the objective function is defined as the module of the difference between the thermal conductivity κ and the target thermal conductivity κ_0. The minimization of this module is the goal behind this work, which means computing a thermal conductivity κ as much closer to the κ_0 value chosen as the desired one. In this way is possible to get the proper geometry configuration for every value of thermal conductivity given any input parameter such as the number of pores within the material, the shape of the pores and the sample dimensions.

The minimization process under the purpose of this Thesis is carried out by applying some Bayesian optimization algorithms, provided by *Scikit-optimize* library, to the thermal conductivity computation performed by OpenBTE, with $|\kappa - \kappa_0|$ representing the cost function.

The minimization of the cost function is identified as *optimization process*, which is formulated under the Equation 5.1.

$$f = min|\kappa - \kappa_0| \tag{5.1}$$

Three different screnarios, listed in Table 5.3, have been investigated starting from three different value of a target thermal conductivity κ_0 with a fixed porosity value equal to 0.1.

Scenario	κ_0	Porosity
1	0.3	0.1
2	0.2	0.1
3	0.1	0.1

TABLE 5.3: Inverse design model scenarios

The total porosity parameter defined as Φ is a function of the number of pores and the unit cells that assemble the material sample, as expressed by 5.2,

$$\Phi = \frac{np}{n_x \cdot n_y} \tag{5.2}$$

where, np is the pores number within the sample while n_x and n_y indicate respectively the unit cells in x- and y-direction.

The same input parameters are considered for every scenario, letting the opportunity of comparison between the three by only tuning the target thermal conductivity as reported in the last paragraph of this section. In this way is possible to analyze the tendence of the pores in positioning within the material for generating the proper configuration for that particular thermal conductivity.

The input parameters assumed for the simulations are the following:

- Mean free path: $mfp = [10.0]$;
- Mean free path number: $n\_mfp = 40$;
- Sample length in x-direction: $Lx = 10$ nm;
- Sample length in y-direction: $Ly = 10$ nm;
- Pores number: $NP = 10$;
- Unit cells in x-direction: $n_x = 10$;
- Unit cells in y-direction: $n_y = 10$.

The Bayesian optimization algorithms used in the simulation work are 5 different algorithms based on the 5 main functions available in the Scikit-Optimize library already described in the previous Chapter of this Thesis. They all have been applied to OpenBTE thermal conductivity computation to minimize the objective function $|\kappa - \kappa_0|$. In terms of number of iterations, a hundred of calls are performed for each one of

the five algorithms used for minimizing the cost function. The three scenarios are analyzed in detail in the next section.

To notice that, the pores combination selected by each algorithm is expressed by a series of couples of discrete values between 0 and 9 within square brackets, which identify the x- and y-coordinate of each pore within the sample.

A representative example is given in Table 5.4 where is given a pores combination between square brackets and evidenced the specific coordinates for each pore which all together generate the MIT Logo by means of 18 pores on a material sample as shown in Figure 5.5.

Porosity configuration	[1,3, 1,4, 1,5, 1,6, 2,5, 2,6, 3,3, 3,4, 3,5, 3,6, 5,3, 5,4, 5,6, 7,3, 7,4, 7,5, 7,6, 8,6]																	
Pore	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
Coordinates (x,y)	1,3	1,4	1,5	1,6	2,5	2,6	3,3	3,4	3,5	3,6	5,3	5,4	5,6	7,3	7,4	7,5	7,6	8,6

TABLE 5.4: Pores position combination Table

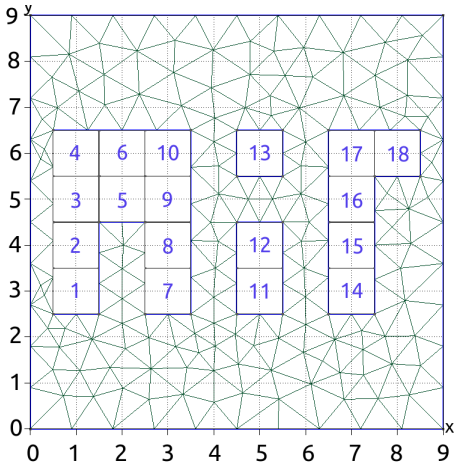

FIGURE 5.5: Pores position configuration example - MIT Logo

5.3 Pores configurations by tuning κ_0

5.3.1 Scenario 1 - $\kappa_0 = 0.3$

In the first scenario, the target thermal conductivity value κ_0 has been set to 0.3 $\frac{W}{m \cdot K}$, given the input parameters previously mentioned.

In Table 5.5 are summarized the relevant results generated by the optimization processes.

$\kappa_0 = 0.3$		
Surrogate model of the function	Optimum pores configuration	Optimum objective function
Gaussian Process 'gp_minimize'	[0,3, 4,0, 8,8, 6,0, 9,9, 9,9, 1,7, 0,0, 5,4, 3,0]	0.000207813
Random Search 'dummy_minimize'	[5,3, 5,8, 8,3, 9,6, 5,6, 2,1, 2,7, 7,6, 7,3, 1,2]	0.000437959
Decision Trees w/randomized regressor 'forestET_minimize'	[3,6, 9,6, 5,1, 8,7, 7,7, 3,9, 7,6, 1,5, 7,5, 0,8]	0.000854115
Decision Trees w/random forest regressor 'forestRF_minimize'	[9,2, 9,6, 5,6, 1,2, 8,9, 8,0, 4,6, 9,6, 8,0, 5,3]	0.00197337
Gradient Boosted Regression Trees 'gbrt_minimize'	[5,3, 9,6, 6,6, 3,2, 9,4, 3,0, 1,1, 6,0, 5,0, 1,1]	0.00223101

TABLE 5.5: Scenario 1 Results - $\kappa_0 = 0.3$

In the first column are listed the Scikit-optimize surrogate models used for the optimization of the objective function in order of best convergence. The second column evidences the pores positions within the sample, between the square brackets, defining its porosity configuration. The third column gives the corresponding optimum output value reached by the optimization process obtained after having performed 100 iterations for each of the model used.

In Figure 5.6 is shown the convergence plot of the objecive function minimization $|\kappa - \kappa_0|$ with a zoom of the latter calls region.

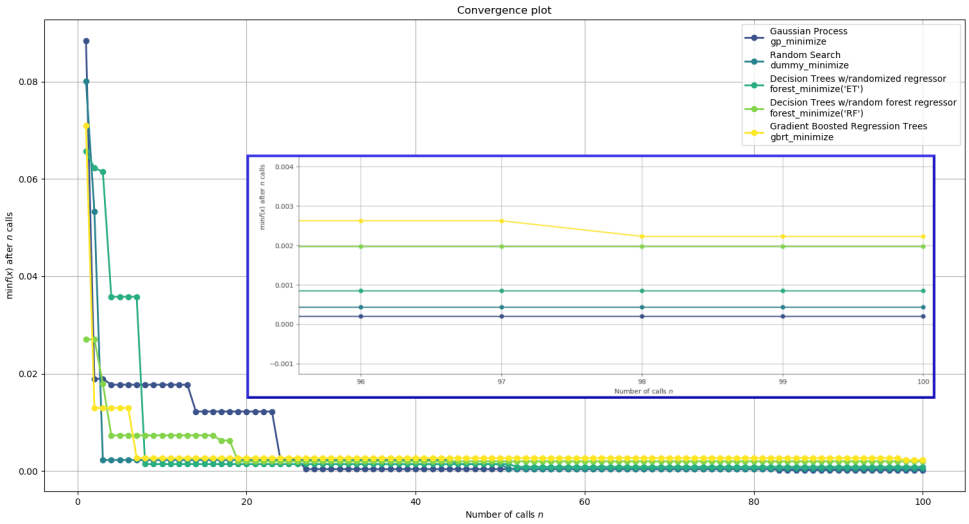

FIGURE 5.6: Objective function minimization for $\kappa_0 = 0.3$

It is observed that all the function models show a good convergence to the objective function, ranging between about 0.0022 and 0.0002, with the latter value reached by the Gaussian Process minimization model. This denotes that the sample configurations selected by the optimizer lead to have a thermal conductivity κ really close to the target one κ_0 equal to 0.3 $\frac{W}{m \cdot K}$, with the highest deviation being lower than 1% given by the Gradient Boosted Regression Trees. The algorithm that best converges the objecive function, achieving a thermal conductivity κ closer to the target one κ_0, is the one based on the Gaussian Process model. Moreover, all the algorithms manifest a remarkably fast and good convergence just after few calls, proving to be a powerful tool for the purpouse object of this Thesis.

In Figure 5.7 is possible to visualize the thermal fluxes with the geometry meshes merged together related to the optimized sample configurations generated by means of all the algorithms tested.

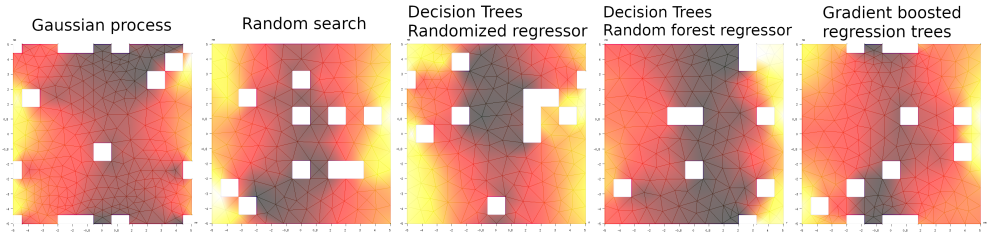

FIGURE 5.7: Geometry meshes and thermal flux merged plots for each function model tested with $\kappa_0 = 0.3$

It is observed that in general the pores tend to occupy disordered positions in the sample preferably along the periodic borders with a lesser tendency of positioning themselves around the geometric center. In all the cases studied the thermal flux cross the nanostructure along the x-direction.

5.3.2 Scenario 2 - $\kappa_0 = 0.2$

The second scenario analyzed consists in repeating all the simulations performed in the first scenario with the same input parameters and tuning only the target thermal conductivity κ_0 which is now set to the value of 0.2 $\frac{W}{m \cdot K}$. By this considerations, the direction of this work is the one of analyze how the phonon thermal transport varies by requiring to the optimizer to reach a lower thermal conductivity while maintaining a constant initial porosity densityl. In this way it can be examined which are the specific pores configurations that better try to restrict the phonon thermal transport across the nanostructure. The Table 5.6, as done for scenario 1, outlines the results of the optimization by applying the five Scikit-optimize algorithms.

$\kappa_0 = 0.2$		
Surrogate model of the function	Optimum pores configuration	Optimum objective function
Gaussian Process 'gp_minimize'	[0,7, 0,3, 9,3, 9,5, 1,9, 4,7, 0,1, 7,9, 0,9, 9,0]	0.000565067
Gradient Boosted Regression Trees 'gbrt_minimize'	[4,2, 0,0, 2,7, 0,3, 1,6, 3,3, 9,4, 9,6, 5,0, 0,2]	0.00163097
Decision Trees w/randomized regressor 'forestET_minimize'	[3,4, 3,0, 2,3, 7,9, 1,7, 0,1, 7,5, 7,3, 1,6, 0,8]	0.00163225
Decision Trees w/random forest regressor 'forestRF_minimize'	[3,8, 8,2, 9,4, 0,7, 9,1, 5,4, 3,0, 8,0, 1,6, 0,4]	0.00324764
Random Search 'dummy_minimize'	[9,7, 5,2, 9,4, 3,1, 8,9, 9,0, 9,5, 0,1, 2,2, 4,9]	0.00518854

TABLE 5.6: Scenario 2 Results - $\kappa_0 = 0.2$

The results in the table are listed, as for scenario 1, in order of which function model better converges the objective function.

Also in this case the algorithm which best converges to the target thermal conductivity is the one based on the Gaussian Process model. The optimization performed has dispensed a minimum objective function value for the different algorithms ranging between approximately 0.0052 and 0.0006, realizing a deviation from the target thermal conductivity of 2.6% and 0.3% respectively. In Figure 5.8 is shown the convergence plot of the objective function for $\kappa_0 = 0.2$ $\frac{W}{m \cdot K}$, with a zoom of the latter calls region.

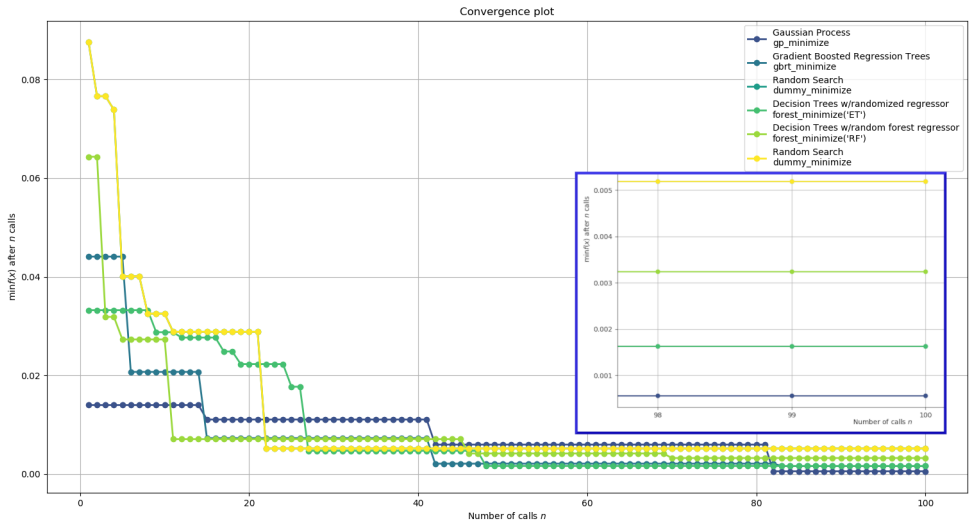

FIGURE 5.8: Objective function minimization for $\kappa_0 = 0.2$

It is evident a slowdown of the optimization process in converging to the target thermal conductivity, as shown by the behaviour of the curves plotted, that reach a minimum objective function value higher with respect to those observed in the previous scenario where $\kappa_0 = 0.3$ $\frac{W}{m \cdot K}$.

In Figure 5.9 is possible to appreciate the geometry meshes merged with the corresponding thermal flux for each one of the five pores combinations listed in Table 5.6.

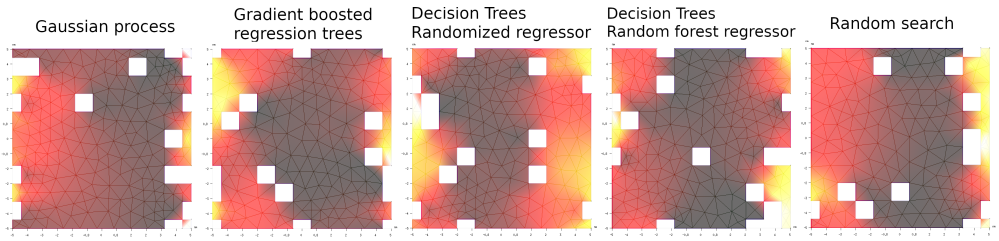

FIGURE 5.9: Geometry meshes and thermal flux plots for each function model performed with $\kappa_0 = 0.2$

It is observed that the pores tend to position themselves along the periodic borders away from the geometric center, exhibiting a tendence to group with eachother creating a sort of block to the phonon thermal transport due to the will of decrease the thermal conductivity from the value of 0.3 to the value of 0.2 $\frac{W}{m \cdot K}$ without increasing the porosity density. Regardless to the slowdown of the converging process with respect to the scenario 1 analyzed previously, the optimization is still producing optimum results with an error lower than 2.6%.

5.3.3 Scenario 3 - $\kappa_0 = 0.1$

In this section is analyzed the third and last scenario about the target thermal conducivity tuning while keeping constant the input parameters. In this case the target thermal conductivity κ_0 is set to 0.1 $\frac{W}{m \cdot K}$. The results of the optimization process are listed in Table 5.7.

$\kappa_0 = 0.1$		
Surrogate model of the function	Optimum pores configuration	Optimum objective function
Gaussian Process 'gp_minimize'	[9,5, 0,2, 9,7, 9,3, 9,9, 0,1, 1,9, 5,0, 0,0, 0,4]	0.00197179
Decision Trees w/randomized regressor 'forestET_minimize'	[7,5, 4,4, 9,0, 9,8, 0,4, 5,2, 9,7, 8,9, 8,6, 9,1]	0.0390348
Gradient Boosted Regression Trees 'gbrt_minimize'	[8,2, 9,2, 0,0, 0,5, 9,9, 3,9, 8,8, 8,6, 8,3, 9,7]	0.0528188
Decision Trees w/random forest regressor 'forestRF_minimize'	[7,6, 4,9, 3,2, 0,4, 1,3, 1,5, 6,5, 1,9, 4,1, 0,8]	0.0892143
Random Search 'dummy_minimize'	[1,2, 5,7, 3,8, 4,0, 3,9, 9,7, 6,2, 0,5, 1,3, 1,6]	0.0978718

TABLE 5.7: Scenario 3 Results - $\kappa_0 = 0.1$

Even in this scenario with $\kappa_0 = 0.1$ $\frac{W}{m \cdot K}$, as for the previous two scenarios, the model function based on Gaussian Process is confirmed to be the one that best minimizes the objective function $|\kappa - \kappa_0|$. The minimum objective function value ranges between around 0.0979 and 0.002 for the five algorithms used for the minimization. It is evident that these results lead to an higher deviation of the thermal conductivity κ from the desired

one κ_0, with respect to the previous two scenarios investigated. In particular, only the Gaussian Process surrogate model produce an acceptable convergence with a deviation from the target thermal conductivity κ_0 of about 2%. All the other algorithms instead, show a high deviation ranging between 30% to 90% from the desired κ_0 value, with the worst one being the Random Search algorithm.

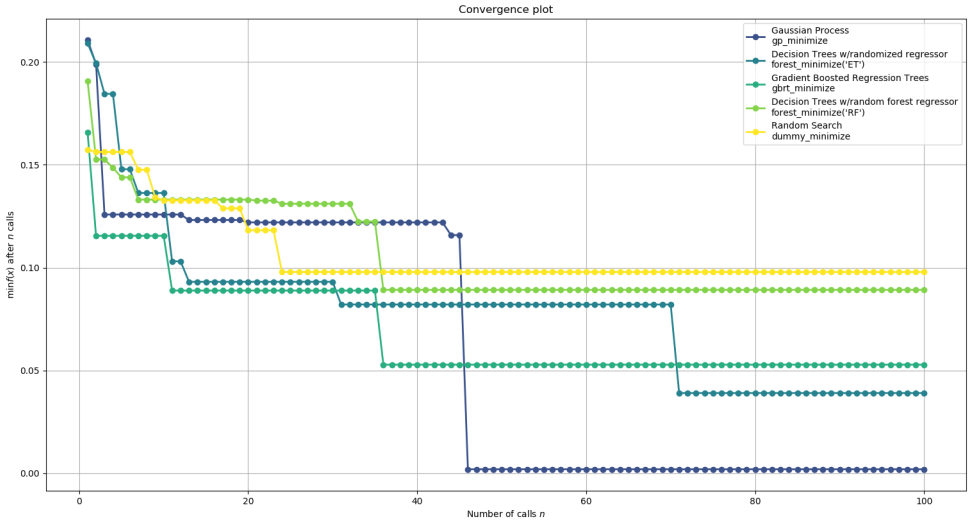

FIGURE 5.10: Objective function minimization for $\kappa_0 = 0.1$

In Figure 5.10 is shown the convergence plot for the different surrogae models used to minimize the objective function $|\kappa - \kappa_0|$.

In this scenario the slowdown and the deviations of the curves in the convergence of the objective function is more pronunced compared to the previous scenario (for which $\kappa_0 = 0.2\ \frac{W}{m \cdot K}$), with the exception of the model function based on Gaussian Process which still lead to a thermal conductivity value κ almost equal to the desired value of 0.1.

The geometry meshes merged together with the corresponding thermal flux results of the optimization process are pictured in Figure 5.11.

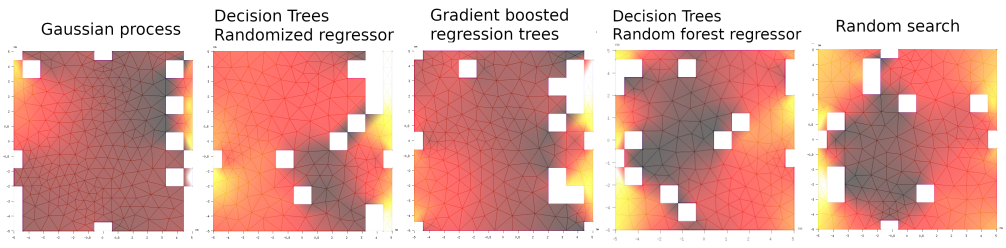

FIGURE 5.11: Geometry meshes and thermal flux plots for each function model performed with $\kappa_0 = 0.1$

It is relevant to remark the differences between the algorithms about the pores positions within the sample which can be easily visualized from the meshes in Figure 5.11. All the function models which lead a worst convergence, except to the Gaussian Process one, exhibit a disordered pores

configurations, however with the pores tendency to place in the neighborhood or along the periodic border. On the contrary, the function model based on Gaussian Process which gives the proper minimization results, shows an ordered pores configuration with most of the pores positioned along one border of the sample. This trend can be explained by the fact that since the target thermal conductivity κ_0 equal to 0.1 $\frac{W}{m \cdot K}$ is apparently low despite of an evidently low porosity density, therefore the pores align along the y-axis on the sample border to obstruct the phonon thermal transport that cross the material through the x-direction. In conclusion an ordered pores configuration is necessary to achieve a thermal conductivity value equal to 0.1 $\frac{W}{m \cdot K}$ with the porosity input parameter represented by 10 pores.

5.4 Scenarios comparison by tuning the porosity density

In this section is illustrated the work done referred to the comparison between the scenarios previously analyzed with the additional change of tuning the porosity density, which was previously kept fixed to the value of 0.1. It is indeed carried out a simulations process of the cases exposed in Table 5.8.

Case	κ_0 $[\frac{W}{m \cdot K}]$	Porosity density
1	0.1	0.1
		0.2
		0.3
2	0.2	0.1
		0.2
		0.3
3	0.3	0.1
		0.2
		0.3

TABLE 5.8: Scenarios comparison tuning the porosity density

The results for all thee three cases related to porosity density equal to 0.1 are already known because coincide with those reported in the previous section, so that the further phonon thermal transport simulations are carried out considering a porosity density of 0.2 and 0.3 for the 3 cases mentioned above.

The input parameters, are the same considered for the simulations performed in the three scenarios presented in the previous section with the exception of the number of pores which is here the tuned parameter. They are listed below:

- Mean free path: $mfp = [10.0]$;
- Mean free path number: $n_{mfp} = 40$;
- Sample length in x-direction: $Lx = 10$ nm;
- Sample length in y-direction: $Ly = 10$ nm;
- Unit cells in x-direction: $n_x = 10$;
- Unit cells in y-direction: $n_y = 10$;
- Pores number Case 1: $NP = 10$;
- Pores number Case 2: $NP = 20$;
- Pores number Case 3: $NP = 30$.

5.4.1 Tuning porosity density keeping $\kappa_0 = 0.1$ fixed

Here is described the first case, for which the target thermal conductivity κ_0 is set to be equal to 0.1 $\frac{W}{m \cdot K}$. This value is kept fixed for all the Scikit-optimize algorithm used, while is tuned the porosity density between the values of 0.1, 0.2, and 0.3. The number of iterations carried out for the minimization process with every function model is equal to 100. Below, are shown the results obtained.

Optimum cost function values for $\kappa_0 = 0.1$			
Surrogate model of the function	Porosity density		
	0.1	0.2	0.3
Random Search 'dummy_minimize'	0.097871	0.000322	0.000326
Gaussian Process 'gp_minimize'	**0.001971**	0.001025	**0.000051**
Gradient Boosted Regression Trees 'gbrt_minimize'	0.052818	0.000230	0.004709
Decision Trees w/randomized regressor 'forestET_minimize'	0.039034	0.000237	0.000270
Decision Trees w/random forest regressor 'forestRF_minimize'	0.089214	**0.000042**	0.002140

TABLE 5.9: Case 1 - Porosity comparison results for $\kappa_0 = 0.1$ (100 calls)

From Table 5.9 is possible to understand how the optimization performance varies depending on the value of porosity density with fixed κ_0 equal to 0.1 $\frac{W}{m \cdot K}$. In particular, the function model based on Gaussian Process is the one which exhibits in general the best results in term of minimizing the objective function $|\kappa - \kappa_0|$ producing the lowest output divergence for porosity level of 0.1 and 0.3. For a porosity value of 0.2, the Decision Trees with Random Forest Regressor (ForestRF) is the one which yields the best optimization result. Comparing the three porosity levels outputs for this Case 1 analyzed, it is noted a better convergence with the intermediate porosity density value, which leads to suppose that for the sample dimensions considered, the number of pores equal to 20 is the quantity that better manage the thermal conductivity $\kappa = 1 \frac{W}{m \cdot K}$.

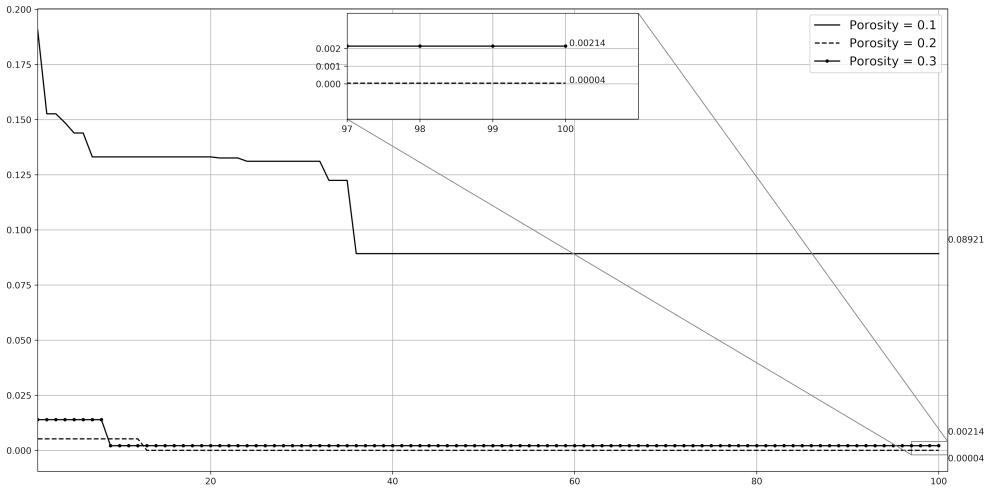

FIGURE 5.12: Decision trees w/random forest regressor plot convergence for different porosity densities with $\kappa_0 = 0.1$

In Figure 5.12 is plotted the objective function convergence plot for the different porosity densities related to ForestRF model function. It is evident a different converging trend between the lowest porosity density and the higher porosity density, with the first one showing an higher deviation. The reason already mentioned in the previous section, is in the complexity of obstruct the phonon thermal transport to produce a low κ value equal to 0.1 $\frac{W}{m \cdot K}$ by considering a relatively low pores number. In Figure 5.13 are shown the optimum geometry meshes merged with the thermal flux generated by the model function based on Decision trees w/random forest regressor, for different level of porosity, by keeping fixed $\kappa_0 = 0.1 \ \frac{W}{m \cdot K}$.

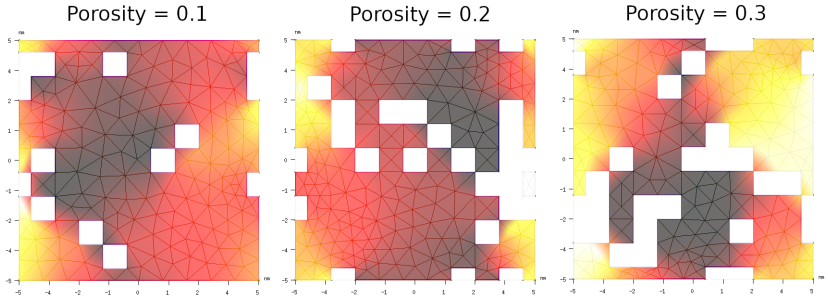

FIGURE 5.13: Geometry meshes and thermal flux obtained by Decision trees w/random forest regressor function for different porosity densities with fixed $\kappa_0 = 0.1$

With the lowest porosity density, on the left picture, the pores occupy mostly the positions along the border on the y-direction, in trying to contain the phonon thermal transport. By increasing the porosity level the pores tend to spread toward the geometric center of the sample, trying to line upe together to generate a sort of two channels for the thermal flux with the aim of managing the desired thermal conductivity κ_0. It is interesting to note that for the case of porosity density equal to 0.3, it is startig to appear a pores overlapping phenomenon, because probably all the 30 pores in the sample are excessive and would have lead to a relevant divergence from the target thermal conductivity, leading to lowering κ with respect to the target one.

5.4.2 Tuning porosity density keeping $\kappa_0 = 0.2$ fixed

Here is described the second case, for which the target thermal conductivity κ_0 is set to be equal to 0.2 $\frac{W}{m \cdot K}$. This value is kept fixed for all the Scikit-optimize algorithm used, while is tuned the porosity density between the values of 0.1, 0.2, and 0.3. The number of iterations carried out for the minimization process with every function model is equal to 100. Below, are shown the results obtained.

Optimum cost function values for $\kappa_0 = 0.2$			
Surrogate model of the function	Porosity density		
	0.1	0.2	0.3
Random Search 'dummy_minimize'	0.005188	0.008812	0.072236
Gaussian Process 'gp_minimize'	**0.000565**	**0.001365**	**0.000642**
Gradient Boosted Regression Trees 'gbrt_minimize'	0.001630	0.013436	0.087245
Decision Trees w/randomized regressor 'forestET_minimize'	0.001632	0.003629	0.104861
Decision Trees w/random forest regressor 'forestRF_minimize'	0.003247	0.007799	0.084249

TABLE 5.10: Case 2 - Porosity comparison results for $\kappa_0 = 0.2$ (100 calls)

From Table 5.10 it is observed the minimum objective function value $|\kappa - \kappa_0|$ to be dispensed by the Gaussian Process model based, for all the porosity levels tested considering a target thermal conductivity equal to 0.2 $\frac{W}{m \cdot K}$.

It is evident that with respect to the Case 1 where a lowest phonon thermal transport was requested, here the porosity densities that better lead to the desired κ_0 are those with lower pores number. This lies in the fact that the increasing of the target thermal conductivity value, involves a higher phonon thermal transport phenomenon and this is in contrast with the presence of high number of pores within the sample. It is important to note that the optimal value of 0.000642 produced by Gaussian Process model based with the corresponding porosity density of 0.3 it is not truthful, because as shown in the mesh configurations a strong overlapping phenomenon occurs, bringing the porosity to be reduced from 0.3 to 0.1. This demonstrate the initial hypothesis for which a porosity level lower than 0.3 is necessary to achieve a thermal conductivity $\kappa = 0.2$ $\frac{W}{m \cdot K}$.

In Figure 5.14 is plotted the objective function convergence plot for the different porosity densities related to GPs model function.

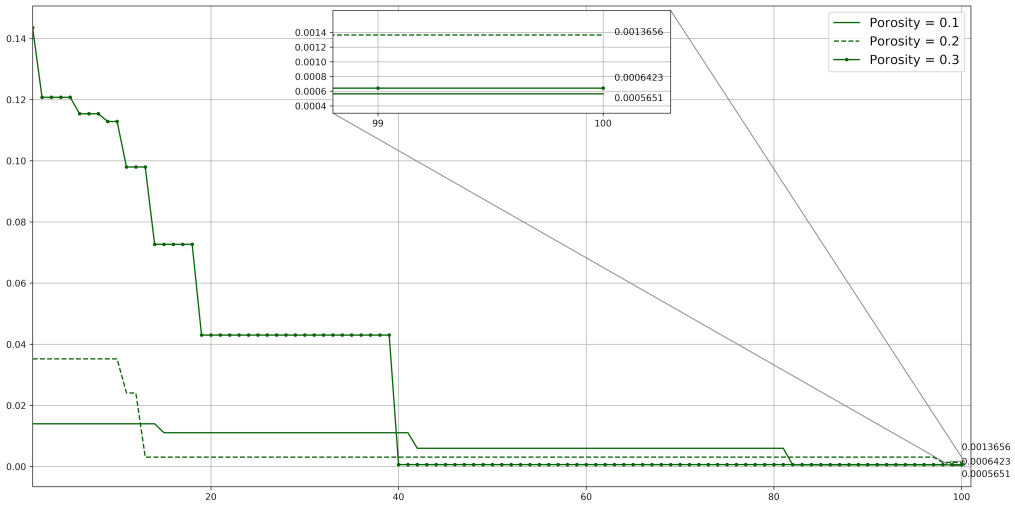

FIGURE 5.14: Gaussian Process plot convergence for different porosity densities with $\kappa_0 = 0.2$

Here it is observed a different converging trend between the highest porosity density and the lowest porosity densities, with the first one showing an higher deviation for the first 40 calls, due to what detailed previously in analyzing Table 5.10. The jumps visible on the porosity curve related to the highest porosity value, and the final convergence to the other two curves it is powered by the overlapping phenomenon. Each jump is a direct consequence of an increase in the overlapped pores number to finally end up to the amount of 15 pores (i.e. porosity value of 0.15) within the sample, starting from 30. The same phenomenon happens for the porosity level of 0.2 as perceived from the jump after around 10 calls, guiding an overlapping phenomenon although weaker than the one with the higher porosity density.

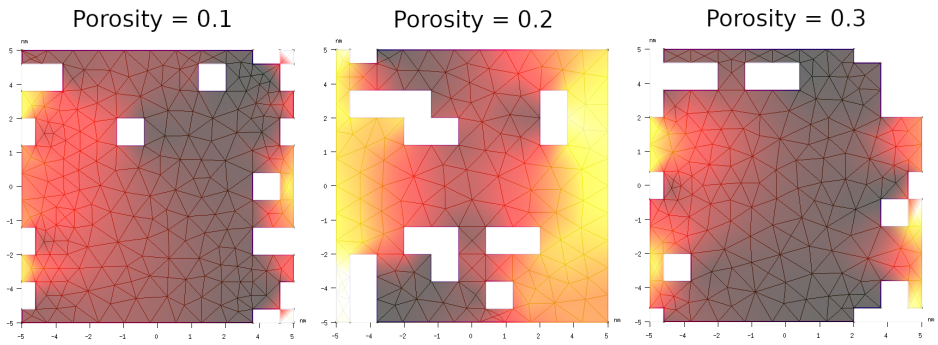

FIGURE 5.15: Geometry meshes and thermal flux obtained by Gaussian Process function for different porosity densities with fixed $\kappa = 0.2$

In Figure 5.15 are shown the optimum geometry meshes merged with the thermal flux generated by the model functiong based on Gaussian Process, for different porosity densities, by keeping fixed $\kappa_0 = 0.2$ $\frac{W}{m \cdot K}$.

It can be appreciated the overlapping pores phenomenon for the porosity density value of 0.3, and a less pronunced one for porosity density of 0.2, since it is expected a corresponding pores number equal to 30 and 20, when only 15 and 16 are respectively present.

With respect to the previous Case 1 studied for which $\kappa_0 = 0.1\ \frac{W}{m \cdot K}$ is considered, here the pores show a weaker tendency on canalizing the phonon thermal transport, with the optimizator orienting the pores to position along the border on y-direction without forming porosity continuity or in a sparse manner around the geometric center of the sample. The overlapping phenomenon holds within itself important information about the bounds of the porosity density value for reaching a determined thermal conductivity. In this Case 2 here analyzed, the results obtained suggest that given the input parameters assumed, it could be not a feasible task reaching a thermal conductivity $\kappa = 0.2$ with a porosity higher than $0.2\ \frac{W}{m \cdot K}$, so that the overlapping phenomenon advises a reduction of pores number to around 15 pores.

5.4.3 Tuning porosity density keeping $\kappa_0 = 0.3$ fixed

Here is described the Case 3, for which the target thermal conductivity κ_0 is set to be equal to $0.3\ \frac{W}{m \cdot K}$. This value is kept fixed for all the Scikit-optimize algorithm used, while is tuned the porosity density between the values of 0.1, 0.2, and 0.3. The number of iterations carried out for the minimization process with every function model is equal to 100. Below, are shown the results obtained.

Optimum cost function values for $\kappa_0 = 0.3$			
Surrogate model of the function	Porosity density		
	0.1	0.2	0.3
Random Search 'dummy_minimize'	0.000437	0.118357	0.181427
Gaussian Process 'gp_minimize'	**0.000207**	**0.000906**	**0.049376**
Gradient Boosted Regression Trees 'gbrt_minimize'	0.002231	0.090412	0.174201
Decision Trees w/randomized regressor 'forestET_minimize'	0.000854	0.074937	0.184580
Decision Trees w/random forest regressor 'forestRF_minimize'	0.001973	0.010945	0.180542

TABLE 5.11: Case 3 - Porosity comparison results for $\kappa_0 = 0.3$ (100 calls)

From Table 5.11 it is observed the minimum objective function value $|\kappa - \kappa_0|$ to be supplied by the Gaussian Process model based, for all the porosity densities tested. In general all the function models reach a better minimization of the objective function with the lowest porosity density of 0.1. However, the minimum value obtained by the GP model based do not reflect the real output because of the overlapping pores phenomenon,

as visualized in the previous Case 2, but in this Case all the simulations are affected. In Figure 5.16 is plotted the objective function convergence plot for the different porosity values related to GPs model function.

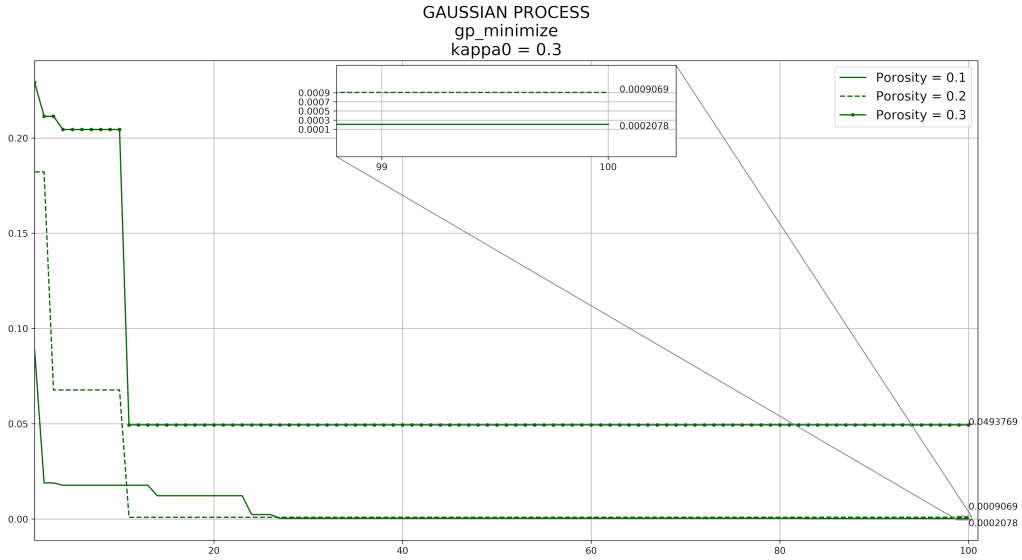

FIGURE 5.16: Gaussian Process plot convergence for different porosity densities with $\kappa_0 = 0.3$

All the curves exhibit jumps in the convergence between the very first to around the 10 calls, where the pores overlapping phenomenon appears. In Figure 5.17 are shown the optimum geometry meshes merged with the thermal flux generated by the model functiong based on Gaussian Process, for different porosity densities, by keeping fixed $\kappa_0 = 0.3$ $\frac{W}{m \cdot K}$.

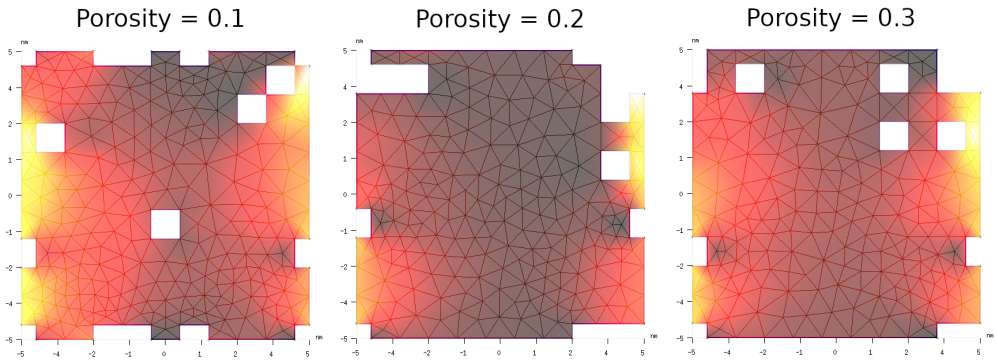

FIGURE 5.17: Geometry meshes and thermal flux obtained by Gaussian Process function for different porosity densities with fixed $\kappa_0 = 0.3$

Despite to the Case 2, here whatever the porosity value, a strong pores overlapping phenomenon is present, leading, in any case, to a reduction of pores number. It is relevant to observe that for the porosity density of 0.2 and 0.3, 10 pores are present within the sample when the expected pores number are respectively 20 and 30, therefore the related configurations undergo to a severe overlapping pores experience. This is due to the fact that, since the sample dimensions are constrained to a relatively small

size, the target thermal conductivity of κ_0 equal to 0.3 $\frac{W}{m \cdot K}$ is theoretically impossible to reach by considering more than 10 pores in the structure, otherwise leading to a significant divergence from the target thermal conductivity.

The result of the simulations performed emphisize the importance of balancing the dimensions of the sample, the number of pores, together with the target thermal conductivity κ_0 to be achieved. The simulations process adopted in this work for studying the inverse design model can be a powerful tool, once set the desiderable thermal property of a structure, for understandig in advance the proper number of pores to place within the material and the proper coordinates of them.

5.4.4 Pores overlapping visualization

For a better visualization of the resulting configurations, a python script for visualizing the pores overlapping has been created and implemented within this work. The bidimensional sample is placed in a tridimensional plane x-y-z assuming negligible the sample's thickness, where **x** and **y** are the *spatial sample coordinates* and **z** is the *overlapped pores coordinate*, indicating the number of pores located in every point of the sample. The previous optimum geometry meshes merged with the thermal flux obtained by the Gaussian Process surrogate model are shown by the plots in Figure 5.18.

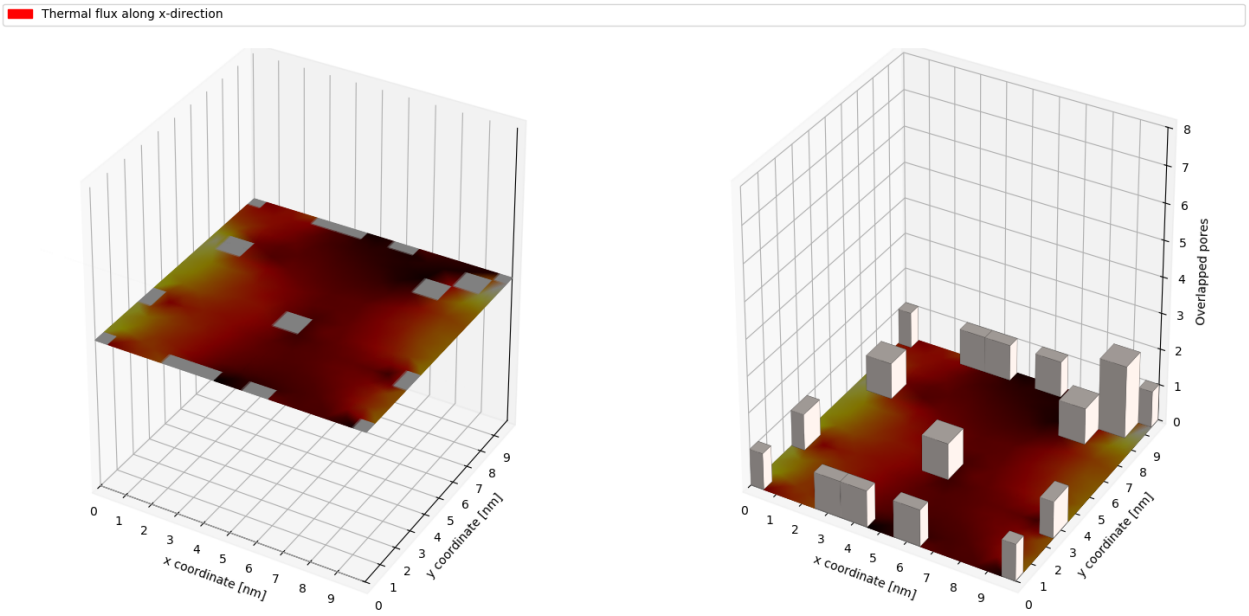

FIGURE 5.18: Pores overlapping with porosity density 0.1 and $\kappa_0 = 0.3$

Making use of this tool, the pores overlapping phenomenon is made easily evident and it is represented by the height of the 3-dimensional columns. The higher the column in a specific x,y coordinates, the more severe is the pores overlapping phenomenon experienced by that individual unit cell of the sample. This is more relevant in the case with porosity Φ equal to 0.2 and 0.3, as shown respectively in Figure 5.19 and 5.20.

In addition, it is supposed that if a unit cell is subjected to pores overlapping phenomenon more than other unit cells, it means that its position is more critical to the achievement of the target thermal conductivity value.

Following this procedure, it would be possible to understand what are the best pores coordinates in order of importance for placing the maximum number of pores within a nanomaterial sample. It is proposed to consider a sort of *overlapping index* that identifies the magnitude of pores overlapping phenomenon that a specific unit cell undergoes.

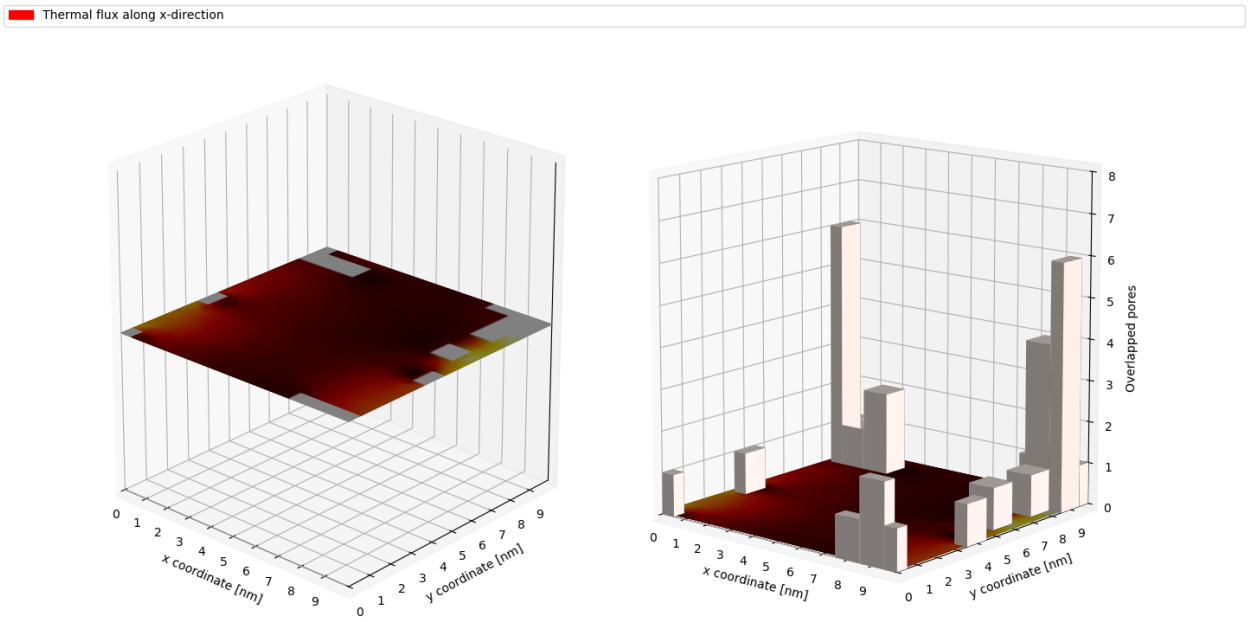

FIGURE 5.19: Pores visualization sample with porosity density 0.2 and $\kappa_0 = 0.3$

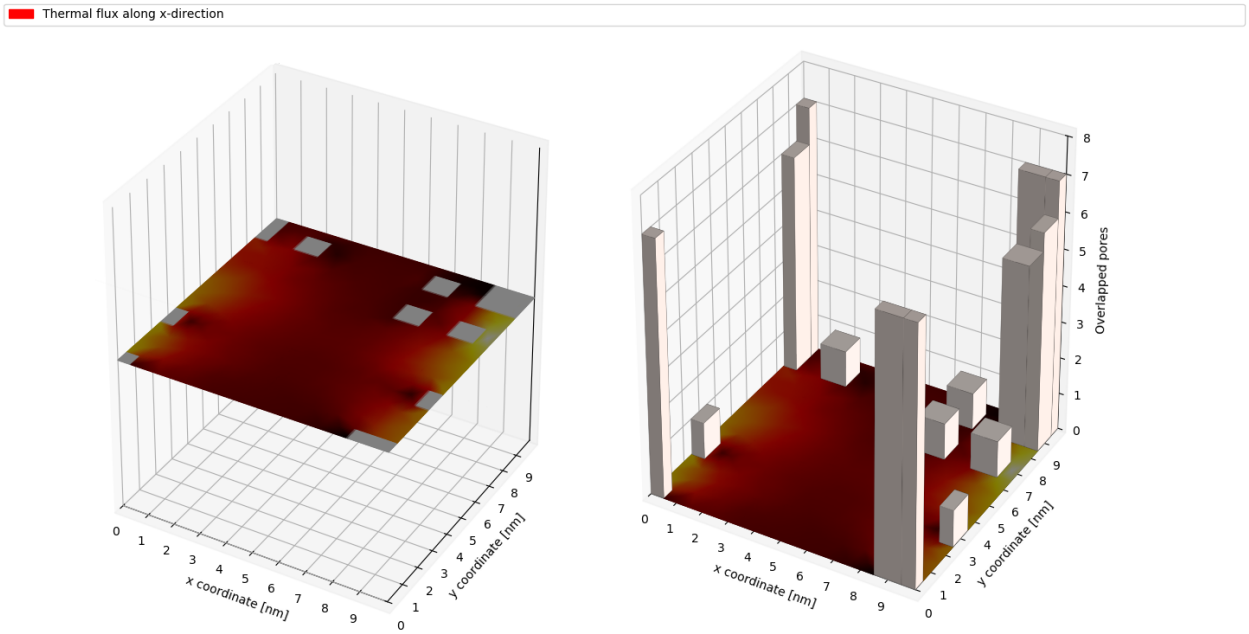

FIGURE 5.20: pores visualization sample with porosity density 0.3 and $\kappa_0 = 0.3$

5.5 Optimization with $\kappa_0 = 0$ tuning the porosity density

In this section is illustrated the work behind the simulations referred to the optimizing objective function $|\kappa - \kappa_0|$, imposing a target thermal conductivity $\kappa_0 = 0$ and tuning the porosity density value between 0.1, 0.2 and 0.3. It is indeed carried out a simulations process as for the cases exposed in Table 5.8. The simulations described in the following subsections, will incur sooner or later to a trivial solution, since a null phonon thermal transport is possible only if a material non-continuity is present within the sample.

5.5.1 Trivial solution with $\kappa_0 = 0$ and 0.1 porosity density

Table 5.12 outlines the results of the optimization by applying the five Scikit-optimize algorithms having set a null target thermal conductivity and a porosity density of 0.1.

$\kappa_0 = 0$		
Surrogate model of the function	Optimum pores configuration	Optimum objective function
Gaussian Process 'gp_minimize'	[9,1, 8,2, 9,3, 9,9, 9,9, 8,0, 5,6, 0,4, 9,0, 9,5]	0.173585
Decision Trees w/randomized regressor 'forestET_minimize'	[0,0, 1,5, 8,3, 0,4, 0,8, 2,1, 1,1, 9,9, 2,3, 3,2]	0.168029
Boosted Regression Trees 'gbrt_minimize'	[5,6, 6,5, 6,3, 9,0, 4,9, 7,7, 7,0, 5,8, 3,2, 6,1]	0.185042
Decision Trees w/random forest regressor 'forestRF_minimize'	[0,2, 9,5, 1,6, 1,9, 0,9, 0,1, 2,8, 1,7, 9,6, 1,5]	0.168607
Random Search 'dummy_minimize'	[4,6, 9,0, 8,9, 5,5, 7,6, 8,8, 9,1, 9,2, 5,5, 7,7]	0.186378

TABLE 5.12: Trivial solution - $\kappa_0 = 0$ and porosity density 0.1

From the resulting objective function values it can be noticed that any of them does not lead a good covergence to the target value κ_0. The deviation spans between the values of 0.168 to 0.186 which means that with the porosity density of 0.1 considered, it was not possible to reach a null κ_0 value. The lowest thermal conductivity value achieved for κ is equal to 0.168 $\frac{W}{m \cdot K}$ after performing 100 calls.

In Figure 5.21 is shown the convergence plot related to all the five Scikit-optimize algorithms used in the minimization of the objective function.

It is observed the significant general deviation from the zero target thermal conductivity for all the surrogate models testes. The deviation is the consequence of the low porosity level which does not allow a strong decrease of the phonon thermal transport oriented to the trivial condition of the null value, and this would also happen by increasing the number of calls even if it would lead to relative better converging curves.

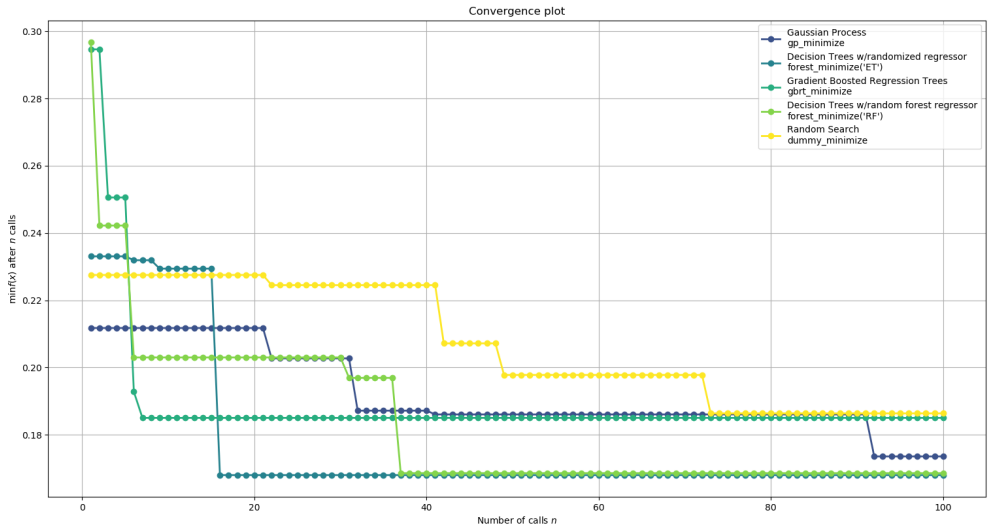

FIGURE 5.21: Objective function minimization for $\kappa_0 = 0$ and porosity density 0.1

In Figure 5.22 are shown the optimum geometry meshes merged with the thermal flux generated by the model function based on Gaussian Process, for porosity density of 0.1, by keeping fixed $\kappa_0 = 0$.

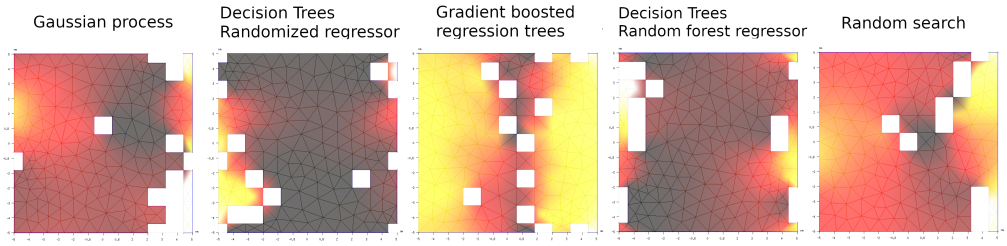

FIGURE 5.22: Geometry meshes and thermal flux plots for each function model performed with $\kappa_0 = 0$ and porosity density 0.1

From the pictured meshes it is easily visualized the tendency of he pores to align and form a line along the y-direction to try in this way to handle the obstruction of the phonon thermal transport.

5.5.2 Trivial solution with $\kappa_0 = 0$ and 0.2 porosity density

Table 5.13 outlines the results of the optimization by applying the five Scikit-optimize algorithms having set a null target thermal conductivity and a porosity density of 0.2.

$\kappa_0 = 0$		
Surrogate model of the function	Optimum pores configuration	Optimum objective function
Gaussian Process 'gp_minimize'	[1, 1, 8, 0, 6, 1, 9, 5, 4, 5, 7, 0, 8, 8, 9, 2, 8, 1, 6, 7, 2, 4, 1, 2, 9, 7, 3, 0, 8, 3, 9, 9, 5, 6, 9, 7, 8, 5, 4, 3]	0.062588
Decision Trees w/randomized regressor 'forestET_minimize'	[7, 8, 5, 3, 2, 8, 8, 3, 9, 9, 2, 6, 9, 1, 1, 9, 9, 2, 0, 0, 3, 0, 9, 6, 3, 7, 9, 4, 4, 3, 3, 0, 0, 9, 6, 6, 1, 7, 0, 9]	0.068837
Boosted Regression Trees 'gbrt_minimize'	[2, 0, 4, 0, 8, 5, 7, 1, 4, 3, 4, 0, 9, 7, 7, 0, 5, 7, 7, 2, 4, 6, 8, 1, 1, 3, 9, 6, 7, 4, 9, 2, 3, 9, 8, 3, 4, 8, 6, 2]	0.059284
Decision Trees w/random forest regressor 'forestRF_minimize'	[3, 3, 8, 8, 1, 8, 2, 5, 8, 6, 6, 5, 9, 4, 8, 1, 0, 9, 3, 6, 2, 1, 1, 2, 4, 5, 1, 5, 1, 4, 1, 1, 9, 9, 2, 7, 7, 7, 7, 3]	0.071300
Random Search 'dummy_minimize'	[6, 9, 7, 1, 2, 5, 2, 1, 4, 1, 5, 9, 0, 3, 0, 9, 4, 7, 0, 7, 3, 8, 3, 6, 3, 0, 8, 6, 0, 5, 9, 4, 8, 4, 4, 1, 1, 2, 6, 9]	0.042257

TABLE 5.13: Trivial solution - $\kappa_0 = 0$ and porosity density 0.2

The outcomes in terms of optimum objective function from Table 5.13 describe an improvement in reaching the null target thermal conductivity value by means of increasing the porosity density with respect to the previous case. In particular the lowest deviation has drop down from 0.168 to 0.042. In Figure 5.23 is shown the convergence plot related to all the five Scikit-optimize algorithms used in the minimization of the objective function.

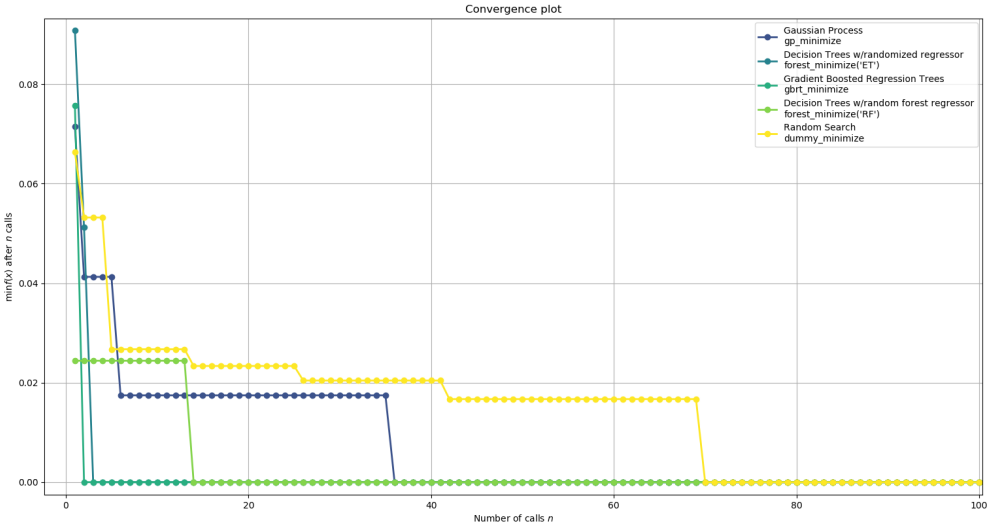

FIGURE 5.23: Objective function minimization for $\kappa_0 = 0$ and porosity density 0.2

The curves exhibit a better converging trend compared to those with lower porosity density, just after few calls performed, with the exception of Random Search based algorithm. However an important improvement is observed, the convergence is still not close to the ideal one for which

stands a null phonon thermal transport. In Figure 5.24 are shown the optimum geometry meshes merged with the thermal flux generated by the model function based on Gaussian Process, for porosity density of 0.2, by keeping fixed $\kappa_0 = 0$.

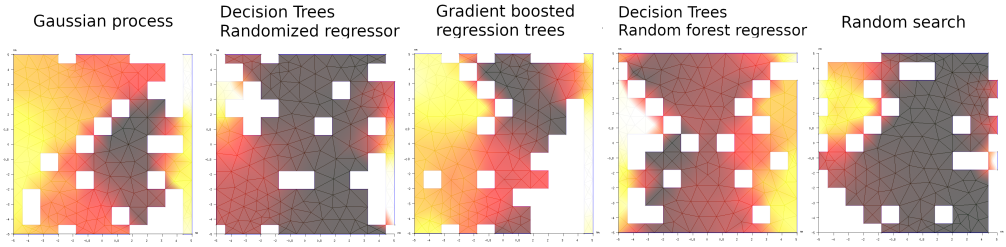

FIGURE 5.24: Geometry meshes and thermal flux plots for each function model performed with $\kappa_0 = 0$ and porosity density 0.2

The improvement in the minimizing the objective function $|\kappa - \kappa_0|$ is clearly detectable by the larger amount of pores within the sample, that to impede the phonon thermal transport across the material, are organized in a sort of arrow-like arrangement in most of the resulting meshes or by grouping together to form a large porosity block. In particular, the mesh generated by the ForestRF based model shows the pores placed to form two opposite arrows generating a x shape.

5.5.3 Trivial solution with $\kappa_0 = 0$ and 0.3 porosity density

Table 5.14 outlines the results of the optimization by applying the five Scikit-optimize algorithms having set a null target thermal conductivity and a porosity density value to 0.3.

$\kappa_0 = 0$		
Surrogate model of the function	Optimum pores configuration	Optimum objective function
Gaussian Process 'gp_minimize'	[3, 7, 0, 9, 4, 3, 0, 8, 9, 7, 0, 9, 0, 5, 9, 5, 0, 0, 0, 1, 9, 0, 0, 6, 2, 4, 1, 8, 5, 1, 4, 2, 0, 0, 0, 0, 1, 9, 8, 2, 0, 6, 9, 1, 0, 2, 2, 0, 9, 5, 4, 4, 1, 4, 0, 9, 0, 3, 2, 1]	0
Decision Trees w/randomized regressor 'forestET_minimize'	[4, 7, 5, 3, 7, 9, 1, 8, 1, 4, 9, 9, 6, 3, 0, 7, 2, 9, 4, 2, 0, 1, 8, 6, 0, 2, 9, 6, 0, 3, 9, 4, 5, 4, 7, 4, 5, 6, 3, 0, 8, 0, 5, 7, 8, 2, 9, 1, 7, 5, 5, 3, 8, 7, 7, 2, 4, 2, 3, 3]	0
Boosted Regression Trees 'gbrt_minimize'	[6, 8, 5, 0, 3, 9, 8, 1, 1, 7, 1, 8, 1, 0, 7, 8, 9, 7, 3, 4, 7, 2, 2, 7, 2, 4, 6, 9, 8, 5, 8, 3, 4, 6, 1, 7, 8, 4, 8, 6, 9, 2, 0, 6, 2, 7, 6, 1, 8, 8, 6, 0, 2, 5, 6, 3, 0, 7, 9, 0]	0
Decision Trees w/random forest regressor 'forestRF_minimize'	[6, 8, 5, 0, 3, 9, 8, 1, 1, 7, 1, 8, 1, 0, 7, 8, 9, 7, 3, 4, 7, 2, 2, 7, 2, 4, 6, 9, 8, 5, 8, 3, 4, 6, 1, 7, 8, 4, 8, 6, 9, 2, 0, 6, 2, 7, 6, 1, 8, 8, 6, 0, 2, 5, 6, 3, 0, 7, 9, 0]	0
Random Search 'dummy_minimize'	[9, 8, 0, 4, 7, 1, 1, 5, 0, 1, 1, 2, 0, 2, 3, 1, 1, 8, 9, 6, 4, 5, 7, 0, 8, 1, 9, 0, 9, 3, 9, 5, 6, 8, 9, 9, 9, 3, 0, 1, 9, 6, 0, 7, 6, 7, 3, 2, 0, 4, 5, 1, 9, 6, 6, 3, 8, 3, 1, 6]	0

TABLE 5.14: Trivial solution - $\kappa_0 = 0$ and porosity density 0.3

All the function models converge to the objective function reaching a thermal conductivity $\kappa = 0$ as shown in the optimum objective function column. The zero value has been reached because of the increse of the porosity density to 0.3, together with keeping the same starting geometry dimension and input paramenters. This led to a pores arrangement to generate a material non-continuity within the sample, which is indeed the trivial solution.

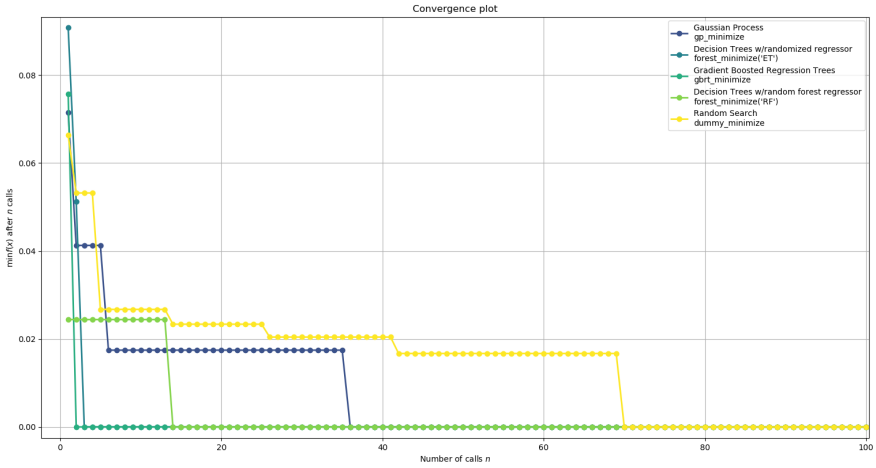

FIGURE 5.25: Objective function minimization for $\kappa_0 = 0$ and porosity density 0.3

In Figure 5.25 is shown the convergence plot related to all the five Scikit-optimize algorithms used in the minimization of the objective function.

All the function models, except for the Random Search and Gaussian Process based algorithms, show a convergence to the zero value just after few calls, but sooner or later any of them will converge to zero before the 100 calls. In Figure 5.24 are shown the optimum geometry meshes generated by the model function based on Gaussian Process, for porosity density of 0.3, by keeping fixed $\kappa_0 = 0$.

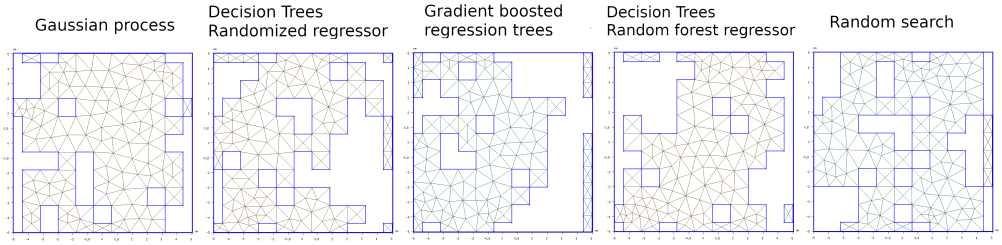

FIGURE 5.26: Geometry meshes for each function model performed with $\kappa_0 = 0$ and porosity density 0.3

As anticipated all the meshes involve a material discontinuity which implies the absence of the phonon thermal transport across the samples.

Chapter 6

Conclusions

In this Thesis we presented a procedure for porosity optimization of nanoporous material.

Part of the work has been spent in studying the heat transport of nanostrucures supporting a nano-constriction, providing the difference between the BTE and Fourier thermal regime. The BTE thermal coefficient is proved to decrease faster than the Fourier coefficient by shrinking the nano-constriction.

Afterward, the phonon transport solver OpenBTE has been interfaced with machine learning algorithms within a Bayesian framework. Following an inverse design approach, we determined nanoporous structured with a given thermal conductivity value and a desired porosity density within the material.

The optimization process has been performed considering three porosity densities experimentally realizable for different target thermal conductivity values. At the state of the art of this work, since no constraints have been imposed in the computation of the phonon thermal transport, a target thermal conductivity value is set as objective purpose. Possible constraints to consider in future progress of the research and to implement in the optimization process are the mechanical stability and electric conductivity. A remarkable feature has been analyzed when increasing the porosity density, defined as porosity overlapping, which is shown to be more evident at high thermal conductivity values. This phenomenon can be addressed in the optimization process to build a powerful tool, once set a desiderable thermal conductivity, to obtain in advance the maximum porosity density for a nanostructure that is the maximum pores number for which the pores overlapping vanishes.

In addition, we tested different Bayesian machine learning algorithms and compared their performances, once applied to the heat transport solver. It has been observed that for the optimization problem, among all the Bayesian algorithms, the one that exhibited in general for all the scenarios the lowest divergence from the target function is the one based on Gaussian processes.

The results obtained by all the simulations carried out in this work offer interesting insights for observing and analyzing the phonon transport and the nanoporous structures generated after the optimization process. Moreover, as highlighted at the end of the work, the approach used would be an innovative one, for several engineering applications, to provide the exact structure porosity density and the proper pores coordinates within the material.

Bibliography

[1] C. Vineis, A. Shakouri, A. Majumdar, and M. Kanatzidis, "Nanostructured thermoelectrics: Big efficiency gains from small features," *Current Opinion in Solid State and Materials Science*, vol. 22, pp. 3970–3980, 09 2010.

[2] G. Romano and C. J. Grossman, "Toward phonon-boundary engineering in nanoporous materials," *Applied Physics Letters*, vol. 105, no. 3, p. 033116, 2014.

[3] M. Zach, "Nanoscience and nanotechnology for advanced energy system," *Current Opinion in Solid State and Materials Science*, vol. 10, pp. 132–143, 2006.

[4] P. F. Villalba, L. D'Ortenzi, G. G. Dalkiranis, E. Cara, A. F. Lopeandia, L. Abad, R. Rurali, X. Cartoixa, N. D. Leo, Z. Saghi, M. Jacob, N. Gambacorti, L. Boarino, and J. R. Viejo, "Impact of pore anisotropy on the thermal conductivity of porous si nanowires," in *Scientific Reports*, 2018.

[5] W. Liu and M. Asheghi, "Phonon–boundary scattering in ultrathin single-crystal silicon layers," *Applied Physics Letters*, vol. 84, no. 19, pp. 3819–3821, 2004.

[6] W. Liu and M. Asheghi, "Phonon-boundary scattering in ultrathin single-crystal silicon layers," *Appl. Phys. Lett.*, vol. 84, p. 3819, 2004.

[7] G. Chen, *Nanoscale energy transport and conversion : a parallel treatment of electrons, molecules, phonons, and photons*. Oxford University Press, 2005.

[8] G. Romano and C. J. Grossman, "Heat conduction in nanostructured materials predicted by phonon bulk mean free path distribution," *Journal of Heat Transfer*, vol. 137, 07 2015.

[9] K. Esfarjani, G. Chen, and H. T. Stokes, "Heat transport in silicon from first-principles calculations," *Phys. Rev. B*, vol. 84, p. 085204, 2011.

[10] A. J. Minnich, G. Chen, S. Mannor, and B. Yilbas, "Quasiballistic heat transfer studied using the frequency-dependent boltzmann transport equation," *Phys. Rev. B*, vol. 84, p. 235207, 2011.

[11] T.-Y. Hsieh, H. Lin, T.-J. Hsieh, and J.-C. Huang, "Thermal conductivity modeling of periodic porous silicon with aligned cylindrical pores," *Journal of Applied Physics*, vol. 111, p. 124329, 2012.

[12] J. Loy, J. Murthy, and D. Singh, "A fast hybrid fourier–boltzmann transport equation solver for nongray phonon transport," *Journal of Heat Transfer*, vol. 135, p. 011008, 2012.

[13] A. Majumdar, "Microscale heat conduction in dielectric thin films," *Journal of Heat Transfer*, vol. 115, pp. 7–16, 1993.

[14] J. M. Ziman, *Electrons and Phonons: The Theory of Transport Phenomena in Solids*. OUP, Oxford, 2001.

[15] N. Mingo, D. A. Broido, L. Lindsay, and W. Li, *Ab initio Thermal Transport*. Springer, New York, 2014.

[16] J. M. Ziman, *Electrons and Phonons: The Theory of Transport Phenomena in Solids*. OUP, Oxford, 2001.

[17] G. Romano and J. Grossman, "Heat conduction in nanostructured materials predicted by phonon bulk mean free path distribution," *Journal of Heat Transfer*, vol. 137, p. 071302, 2015.

[18] W. Li, J. C. Montana, N. Katcho, and N. Mingo, "Shengbte: A solver of the boltzmann transport equation for phonons," *Computer Physics Communications*, vol. 185, pp. 1747–1758, 2014.

[19] O. Hellman, "Tdep." https://ollehellman.github.io, 2018.

[20] C. Geuzaine and J.-F. Remacle, "Gmsh: A 3-d finite element mesh generator with built-in pre- and post-processing facilities," *International Journal for Numerical Methods in Engineering*, vol. 79, pp. 1309 – 1331, 09 2009.

[21] G. Romano, "Openbte: A multiscale solver for the phonon boltzmann transport equation," in *APS March Meeting Abstracts*, vol. 2019 of *APS Meeting Abstracts*, p. T70.321, Jan. 2019.

[22] G. Louppe, "Bayesian optimization with scikit-optimize," *PyData Amsterdam*, 2017.

[23] J. Bergstra and Y. Bengio, "Random search for hyper-parameter optimization," *Journal of machine learning research*, vol. 13, pp. 281–305, Feb 2012.

[24] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay, "Scikit-learn: Machine learning in Python," *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.

[25] Z. B. Zabinsky, *Random Search Algorithms*. American Cancer Society, 2011.

[26] scikit-learn developers, "Ensemble methods." https://scikit-learn.org/stable/modules/ensemble.html, 2019.

[27] J. H. Friedman, "Greedy function approximation: A gradient boosting machine," *The Annals of Statistics*, vol. 29, pp. 1189–1232, 10 2001.

[28] scikit-learn developers, "Forests of randomized trees." https://scikit-learn.org/stable/modules/ensemble.html#forests-of-randomized-trees, 2019.

[29] L. Breiman, "Arcing classifier (with discussion and a rejoinder by the author)," *The Annals of Statistics*, vol. 26, pp. 801–849, 06 1998.