Journal of Quantitative Spectroscopy & Radiative Transfer 328 (2024) 109163

Contents lists available at ScienceDirect

Journal of Quantitative Spectroscopy and Radiative Transfer

journal homepage: www.elsevier.com/locate/jqsrt

DSGF solver for near-field radiative heat transfer: User guide

Lívia M. Corrêa [a], Lindsay P. Walter [a], Jan L. Čas [a], Mathieu Francoeur [a,b,*]

[a] Department of Mechanical Engineering, The University of Utah, Salt Lake City, 84112, UT, USA
[b] Department of Mechanical Engineering, McGill University, Montréal, H3A 0C3, QC, Canada

ARTICLE INFO

Keywords:
Near-field radiative heat transfer
Fluctuational electrodynamics
Discrete system Green's function method
DSGF solver

ABSTRACT

The discrete system Green's function (DSGF) method is a fluctuational electrodynamics-based numerical framework for predicting near-field radiative heat transfer (NFRHT) between three-dimensional thermal sources of arbitrary number, shape, size, and material. In the DSGF method, thermal sources are discretized into cubic subvolumes along a cubic lattice, and the system Green's functions between all subvolumes are obtained by solving a system of linear equations. From the system Green's functions, quantities of interest in heat transfer such as the power dissipated and the thermal conductance are calculated. The objective of this paper is to provide a user guide of the DSGF solver publicly available on GitHub. The basics of the DSGF method are first reviewed, followed by a detailed description of the DSGF solver implemented in MATLAB and C. The C implementation is parallelized and includes an iterative procedure which is not available in the MATLAB version. Example problems of NFRHT between two dipoles, two spheres, two cubes, and two membranes that can be used for verification are provided.

1. Introduction

Near-field radiative heat transfer (NFRHT) is a regime in which at least one of the system characteristic lengths, namely the size of the thermal sources or their separation distances, is smaller than the thermal photon wavelength as given by Wien's law [1]. Planck's theory is not applicable in the near-field regime because the wave characteristics of thermal radiation must be taken into account. These wave characteristics include interference of propagating waves and the tunneling of evanescent waves that result in radiative heat transfer exceeding the blackbody limit [2–8]. The heat transfer enhancement provided by NFRHT can potentially be exploited in various technologies, such as thermophotovoltaics [9–12], localized radiative cooling [13], and thermal rectification [14–17].

NFRHT is modeled using fluctuational electrodynamics, a formalism in which fluctuating current densities are added in Maxwell's equations to account for thermal emission [18]. Most NFRHT problems are solved using the volume integral equation technique combined with analytical expressions for the system Green's function relating the monochromatic electromagnetic fields between a source and an observation point. This approach provides exact results and can be applied to systems in which the boundaries are geometrically simple, such as layered media [19], point dipoles [20–23], and spheres [24,25].

A few discretization-based methods have been developed for modeling NFRHT in complex geometries. In the finite-difference frequency-domain [26,27] and the finite-difference time-domain [28–31] methods, the differential form of Maxwell's equations is discretized. The boundary element method based on a fluctuating-surface current formulation in which the surface integral form of Maxwell's equations is discretized has been proposed [32,33] and implemented in an open-source solver called SCUFF-EM [34]. This solver, however, can only handle homogeneous thermal sources. Finally, approaches based on discretizing the volume integral form of Maxwell's equations have been developed, and include the thermal discrete dipole approximation [35–42], the method of moments based on a fluctuating volume–current formulation [43] implemented in the BUFF-EM solver [44], and the discrete system Green's function (DSGF) method [42,45,46]. Conversely to SCUFF-EM, the BUFF-EM solver can handle inhomogeneous thermal sources. In addition to these discretization-based frameworks, the semi-analytical rigorous coupled wave analysis combined with the scattering matrix method implemented in the MESH solver [47] enables NFRHT simulations between plane layers decorated by structures exhibiting one- and two-dimensional periodicity.

Despite the recent efforts in developing computational methods, NFRHT simulations in complex geometries remain challenging. The transition of NFRHT from laboratory-scale concepts to actual energy conversion and thermal management devices will require a variety

* Corresponding author at: Department of Mechanical Engineering, The University of Utah, Salt Lake City, 84112, UT, USA.
E-mail address: mfrancoeur@mech.utah.edu (M. Francoeur).

https://doi.org/10.1016/j.jqsrt.2024.109163
Received 7 May 2024; Received in revised form 1 August 2024; Accepted 20 August 2024
Available online 22 August 2024
0022-4073/© 2024 Elsevier Ltd. All rights are reserved, including those for text and data mining, AI training, and similar technologies.

of accurate and efficient modeling tools. This paper aims to contribute to computational NFRHT by providing a user guide of the DSGF solver available in both MATLAB and C on GitHub [48]. In the DSGF method, thermal sources are discretized into cubic subvolumes along a cubic lattice, and the system Green's functions are calculated numerically between all subvolumes by solving a system of linear equations. Quantities of interest in heat transfer such as the power dissipated and the thermal conductance are calculated in the post-processing phase using the system Green's functions. The DSGF method has been verified against exact results based on the analytical solution for chains of two and three silicon dioxide (SiO_2) spheres [45]. In addition, the DSGF method has been validated with experimental data of NFRHT between two coplanar subwavelength-thick silicon carbide (SiC) membranes [49]. The DSGF solver also enables far-field radiative heat transfer simulations, since fluctuational electrodynamics converges to Planck's theory of thermal radiation in the limit that all system characteristic lengths are larger than the thermal photon wavelength.

Limitations associated with the current implementation of the DSGF solver should be mentioned. First, this version predicts NFRHT between two groups of thermal sources maintained at two distinct temperatures. Yet, it is possible to utilize the system Green's functions calculated in the solver to, for example, predict NFRHT between N thermal sources maintained at N distinct temperatures, which would require a few modifications by the user. Second, simulations with the MATLAB and C codes are limited to ∼20,000 subvolumes for 256 GB of memory (RAM) and ∼40,000 subvolumes for 1 TB of RAM when solving the system of equations by direct inversion. This limit in the C code is extended to ∼29,000 subvolumes for 256 GB of RAM and ∼55,000 subvolumes for 1 TB of RAM when solving the system of equations iteratively.

The rest of the paper is organized as follows. The basics of the DSGF method are reviewed in Section 2. The installation and initialization of the DSGF codes written in MATLAB and C are discussed in Section 3. The DSGF solver and the required user-specified inputs are described in detail in Section 4, along with the solution of the system of linear equations. In Section 5, two tutorials for simulating NFRHT between two cubes of SiO_2 and NFRHT between two membranes of SiC are described. Additional example problems of NFRHT between two dipoles, two spheres, and two cubes are discussed. Concluding remarks are provided in Section 6.

2. Overview of the DSGF method

The DSGF method described in Ref. [45] is a framework based on fluctuational electrodynamics enabling NFRHT simulations between finite, three-dimensional thermal sources of arbitrary number, shape, size, and material embedded within a lossless background reference medium. The thermal sources occupy a total volume V_{therm} and are isotropic, linear, nonmagnetic, and in local thermodynamic equilibrium. The background reference medium occupies a volume V_{ref} and is homogeneous, linear, isotropic, nonmagnetic, nonabsorbing, and therefore nonemitting. The fields are time-harmonic ($e^{-i\omega t}$), and the electromagnetic responses of the thermal sources and the background reference medium are respectively described by a complex-valued dielectric function $\varepsilon(\mathbf{r},\omega)$ and a real-valued dielectric function $\varepsilon_{\text{ref}}(\omega)$.

In the absence of external electromagnetic fields, the following expression for the electric field at location $\mathbf{r}$ and frequency ω is derived from the stochastic Maxwell equations of fluctuational electrodynamics using the volume integral equation technique:

$$\mathbf{E}(\mathbf{r},\omega) = i\omega\mu_0 \int_{V_{\text{therm}}} \bar{\bar{\mathbf{G}}}(\mathbf{r},\mathbf{r}',\omega)\mathbf{J}^{(\text{fl})}(\mathbf{r}',\omega)\mathrm{d}^3\mathbf{r}', \quad \mathbf{r}\in\Re^3, \tag{1}$$

where μ_0 is the vacuum permeability. The fluctuating current density, $\mathbf{J}^{(\text{fl})}$, is correlated to the local temperature of a thermal source via the fluctuation–dissipation theorem [18]:

$$\left\langle \mathbf{J}^{(\text{fl})}(\mathbf{r},\omega)\mathbf{J}^{(\text{fl})^{\dagger}}(\mathbf{r}',\omega')\right\rangle = 4\pi\omega\varepsilon_0 \mathrm{Im}[\varepsilon(\mathbf{r},\omega)]\Theta(\omega,T)\delta(\mathbf{r}-\mathbf{r}')\delta(\omega-\omega')\bar{\bar{\mathbf{I}}}, \tag{2}$$

where ε_0 is the vacuum permittivity, $\langle\rangle$ represents the ensemble average, † is the conjugate transpose, δ is the Dirac function, and $\bar{\bar{\mathbf{I}}}$ is the unit dyadic. The mean energy of an electromagnetic state is given by

$$\Theta(\omega,T) = \frac{\hbar\omega}{e^{\frac{\hbar\omega}{k_B T}} - 1}, \tag{3}$$

where $\hbar$ is the reduced Planck constant and k_B is the Boltzmann constant.

The system Green's function in Eq. (1), $\bar{\bar{\mathbf{G}}}(\mathbf{r},\mathbf{r}',\omega)$, relates the monochromatic electric field observed at $\mathbf{r}$ to a point source excitation at $\mathbf{r}'$ and therefore depends on the boundary conditions of the system. The system Green's function is the only unknown in Eq. (1), and it is the variable that is calculated in the DSGF method.

The unknown system Green's function may be written in terms of the known free-space Green's function, $\bar{\bar{\mathbf{G}}}^0(\mathbf{r},\mathbf{r}',\omega)$, as follows:

$$\bar{\bar{\mathbf{G}}}^0(\mathbf{r},\mathbf{r}',\omega) = \bar{\bar{\mathbf{G}}}(\mathbf{r},\mathbf{r}',\omega) - k_0^2 \int_{V_{\text{therm}}} \bar{\bar{\mathbf{G}}}^0(\mathbf{r},\mathbf{r}'',\omega)\varepsilon_r(\mathbf{r}'',\omega)\bar{\bar{\mathbf{G}}}(\mathbf{r}'',\mathbf{r}',\omega)\mathrm{d}^3\mathbf{r}'', \tag{4}$$

where $\varepsilon_r(\mathbf{r},\omega) = \varepsilon(\mathbf{r},\omega) - \varepsilon_{\text{ref}}(\omega)$ is the relative dielectric function of the thermal sources with respect to the background reference medium. The variable $k_0 = \omega\sqrt{\mu_0\varepsilon_0}$ is the magnitude of the vacuum wavevector. The free-space Green's function relates the monochromatic electric field observed at $\mathbf{r}$ to a point source excitation at $\mathbf{r}'$ in the lossless background reference medium [50].

The self-consistent Eq. (4) for the system Green's function is solved numerically by discretizing the bulk thermal sources of total volume V_{therm} into N cubic subvolumes along a cubic lattice. The subvolume size, ΔV_i, must be smaller than all characteristic lengths, namely the vacuum and material wavelengths, and the gap spacing between the thermal sources. In each subvolume, the temperature, dielectric function, electric field, and the free-space and system Green's functions are assumed to be uniform. The discretized Eq. (4) results in the following system of linear equations:

$$\left\{\begin{bmatrix} \bar{\bar{\mathbf{I}}} & 0 & 0 \\ 0 & \ddots & 0 \\ 0 & 0 & \bar{\bar{\mathbf{I}}} \end{bmatrix} - k_0^2 \begin{bmatrix} \bar{\bar{\mathbf{G}}}^0_{11} & \dots & \bar{\bar{\mathbf{G}}}^0_{1N} \\ \vdots & \ddots & \vdots \\ \bar{\bar{\mathbf{G}}}^0_{N1} & \dots & \bar{\bar{\mathbf{G}}}^0_{NN} \end{bmatrix} \begin{bmatrix} \alpha_1^{(0)} & 0 & 0 \\ 0 & \ddots & 0 \\ 0 & 0 & \alpha_N^{(0)} \end{bmatrix}\right\} \begin{bmatrix} \bar{\bar{\mathbf{G}}}_{11} & \dots & \bar{\bar{\mathbf{G}}}_{1N} \\ \vdots & \ddots & \vdots \\ \bar{\bar{\mathbf{G}}}_{N1} & \dots & \bar{\bar{\mathbf{G}}}_{NN} \end{bmatrix} = \begin{bmatrix} \bar{\bar{\mathbf{G}}}^0_{11} & \dots & \bar{\bar{\mathbf{G}}}^0_{1N} \\ \vdots & \ddots & \vdots \\ \bar{\bar{\mathbf{G}}}^0_{N1} & \dots & \bar{\bar{\mathbf{G}}}^0_{NN} \end{bmatrix}, \tag{5}$$

where $\alpha_i^{(0)} = \Delta V_i \varepsilon_r(\mathbf{r}_i,\omega)$ is the bare polarizability of subvolume i. The system of equations (Eq. (5)) can be represented as $\bar{\bar{\mathbf{A}}}\bar{\bar{\mathbf{G}}} = \bar{\bar{\mathbf{G}}}^0$, where $\bar{\bar{\mathbf{A}}}$ is the interaction matrix containing the terms inside the curly brackets. Each submatrix $\bar{\bar{\mathbf{G}}}^0_{ij}$, $\bar{\bar{\mathbf{G}}}_{ij}$, and $\bar{\bar{\mathbf{A}}}_{ij}$ is a 3×3 matrix, where the subscripts i and j refer to the interacting subvolumes. Therefore, the full matrices $\bar{\bar{\mathbf{A}}}$, $\bar{\bar{\mathbf{G}}}$, and $\bar{\bar{\mathbf{G}}}^0$ contain $3N\times3N$ terms.

The system of linear equations (Eq. (5)) is solved for each frequency ω of the thermal spectrum. Once the system Green's functions are known for all frequencies and between all subvolumes, quantities of interest can be computed. The net total power dissipated (units of W) in a subvolume ΔV_i is calculated as:

$$Q_{t,\Delta V_i} = \frac{1}{2\pi}\int_0^{\infty}\left\{\sum_{\substack{j=1\\ j\neq i}}^{N}\left[\Theta(\omega,T_j) - \Theta(\omega,T_i)\right]\mathcal{T}_{ij}(\omega)\right\}d\omega. \tag{6}$$

In Eq. (6), $\mathcal{T}_{ij}(\omega)$ is the spectral transmission coefficient between subvolumes i and j calculated from the system Green's functions as follows:

$$\mathcal{T}_{ij}(\omega) = 4k_0^4\Delta V_i\Delta V_j \mathrm{Im}[\varepsilon(\mathbf{r}_i,\omega)]\mathrm{Im}[\varepsilon(\mathbf{r}_j,\omega)]\mathrm{Tr}[\bar{\bar{\mathbf{G}}}(\mathbf{r}_i,\mathbf{r}_j,\omega)\bar{\bar{\mathbf{G}}}^{\dagger}(\mathbf{r}_i,\mathbf{r}_j,\omega)], \tag{7}$$

where Tr represents the trace. The net total power dissipated within bulk thermal sources of volume V_A is obtained by summing up Eq. (6) over all subvolumes contained in V_A.

The total thermal conductance (units of W/K) between two distinct groups of thermal sources of volumes V_A and V_B maintained by a temperature difference $\delta T \to 0$ is calculated as:

$$G_{t,AB} = \frac{1}{2\pi}\int_0^\infty \left[\frac{\partial\Theta(\omega,T')}{\partial T}\right]_{T'=T} \sum_{i\in V_A}\sum_{j\in V_B} \mathcal{T}_{ij}(\omega)d\omega. \tag{8}$$

The net spectral power dissipated and the spectral thermal conductance are respectively obtained by evaluating Eqs. (6) and (8) without the integration over the angular frequency ω.

3. Code installation and initialization

DSGF codes written in MATLAB and C are available on GitHub at https://GitHub.com/Discrete-System-Greens-Function. The specificities of each implementation are discussed hereafter.

3.1. MATLAB implementation

The MATLAB implementation of the DSGF solver is the simplest to use. The only file that requires modifications before performing simulations is `DSGF_user_inputs.m`. Details about the user-specified inputs are provided in Section 4. Once the inputs are specified, simulations are launched by clicking on the `Run` button in the MATLAB GUI interface. In the MATLAB implementation, the system of linear equations (Eq. (5)) is solved strictly via direct inversion, which requires substantial memory for a large number of subvolumes, as discussed in Section 4.2.3. Once the simulations are completed, the results can be accessed in the folder named with the date and time when the computations were initiated.

3.2. C implementation

The C implementation of the DSGF solver is parallelized, thus enabling faster computations than with the MATLAB code. As in the MATLAB code, the system of linear equations (Eq. (5)) can be solved by direct inversion. In addition, the system of linear equations can be solved via an iterative procedure [51,52] that reduces memory usage. The C code necessitates installation; the instructions for installing the required packages and solvers are provided on GitHub: https://GitHub.com/Discrete-System-Greens-Function/DSGF_c/blob/main/DSGF_C_Installation_Instructions_Windows_Linux.pdf.

After installation, the user must first compile the solver by typing `\.run_dsgf` in the terminal. When this step is completed, the message “compilation complete” appears in the terminal and the user can start performing simulations. Then, the user can modify the input parameters and the desired outputs using the files located in the `user_inputs` folder (see Fig. 1 and further discussion in Section 4). After modifying the parameters, the user should type `\.DSGF` to initiate the simulations. Once the simulations are completed, the results can be accessed in the folder `results`, where the data are ordered as a function of the thermal source geometry, number of subvolumes, and date and time when the simulations were initiated. The solver does not need to be compiled again prior to performing additional simulations.

4. Description of the DSGF solver

This section describes the key building blocks of the DSGF solver, namely the user-specified inputs and the solution of the system of linear equations. Calculations with the DSGF solver follow a simple forward pathway summarized in Fig. 2. Inputs are first specified by the user. These inputs include the spatial and spectral discretizations, the material selection (i.e., dielectric function of the thermal sources), and the temperatures. For each frequency, the free-space Green’s function matrix $\bar{\bar{\mathbf{G}}}^0$ and the interaction matrix $\bar{\bar{\mathbf{A}}}$ are populated from the inputs. Then, the system Green’s functions are calculated by solving the system of linear equations (Eq. (5)). The desired spectral and total outputs are computed and stored in the post-processing stage.

4.1. User-specified inputs

4.1.1. Spatial discretization

Thermal sources are discretized into cubic subvolumes along a cubic lattice (see Fig. 3). The center point $\mathbf{r}_i$ and volume ΔV_i of each subvolume i must be specified. Two types of spatial discretization can be used in the DSGF solver: sample and user defined. The `sample` discretization refers to pre-defined spatial discretization files provided with the DSGF solver. Spatial discretization files are available for spheres (1 to 382,336 uniform subvolumes) and for cubes (1 to 3375 uniform subvolumes). Dipoles can also be simulated by assigning one subvolume per thermal source. When opting for `sample` spatial discretization, the user needs to modify the following inputs in the DSGF solver:

1. Select the geometry of each thermal source (sphere or cube);
2. Select the number of subvolumes per thermal source based on the available options in `Library/Discretizations/sphere` for spheres and in `Library/Discretizations/cube` for cubes;
3. Define the characteristic length (L_{char}) of the thermal sources (radius for spheres, side length for cubes);
4. Define the edge-to-edge gap spacing (d) between the thermal sources.

The solver automatically scales the discretization to the defined L_{char}, calculates the total volume of the thermal sources, and provides a uniform ΔV_i discretization in each thermal source.

For spatial discretizations generated by the user, `user_defined` must be selected. In that case, the spatial discretization must contain the complete information about the system, namely all discretized thermal sources with their actual dimensions and gap spacings. Two `.txt` files need to be inserted by the user and imported by the solver. The first file provides the center points $\mathbf{r}_i$ for each subvolume, whereas the second file provides the ΔV_i values for each subvolume. The user should copy these `.txt` files in `Library/Discretizations/User_defined` and type the file names in the appropriate input file (`DSGF_user_inputs.m` in MATLAB and `user_defined.txt` in C). Because ΔV_i is imported into the solver, subvolumes in a given thermal source may be of different sizes. Nonuniform spatial discretization enables increased accuracy without increasing excessively the computational resources. For example, in Fig. 3(c), smaller subvolumes in the cubes are defined near the gap d where $\Delta V_i < d$ must be satisfied.

A MATLAB script is available for discretizing systems of two cuboids of the same dimensions separated by a gap d. The script is located at: https://GitHub.com/Discrete-System-Greens-Function/DSGF_utilities/tree/main/matlab_scripts/Spatial_discretization/cuboid_uniform_nonuniform_meshing. To generate user defined discretizations with this MATLAB script, `cuboid_discretization_SCRIPT.m` must be edited as follows:

1. Provide a description of the system in the name of the `.txt` files;
2. Define the edge-to-edge gap spacing d;
3. Define the cuboid dimensions;
4. Define the discretization type as uniform, nonuniform with two different subvolume sizes, or nonuniform with three different subvolume sizes;
   For uniform discretization, the user must:
   (a) Specify `refinements` as 0;
   (b) Specify `n_0` as the number of subvolumes along the z-axis, which is perpendicular to the gap spacing (see Fig. 4(a) for axis orientation).

   For nonuniform discretization with two different subvolume sizes along the x-axis in a given thermal source (see Fig. 4(a)), the user must:

1. `user_inputs/control.txt`:
   - Spatial discretization type: `sample` or `user_defined`.
   - Number of subvolumes in thermal source A.
   - Number of subvolumes in thermal source B.
   - Material: `SiO2`, `SiC` or `Si3N4`.
   - Dielectric function of lossless background reference medium.
   - Number of frequencies.
   - Spectral discretization: file name with its extension.
   - Solution of system of equations: direct inversion (`D`) or iterative solver (`I`).
   - Multithread (parallel calculations): yes (`Y`) or no (`N`).
   - Saving options: yes (`Y`) or no (`N`).
2. `user_inputs/geometry/sample.txt`:
   - Geometry of thermal source A: `sphere` or `cube`.
   - Characteristic length `L_char` of thermal source A (m).
   - Geometry of thermal source B: `sphere` or `cube`.
   - Characteristic length `L_char` of thermal source B (m).
   - Edge-to-edge gap spacing `d` between thermal sources (m).
   - Temperature of thermal source A (K).
   - Temperature of thermal source B (K).
3. `user_inputs/geometry/user_defined.txt`
   - Edge-to-edge gap spacing `d` between thermal sources (m).
   - File name of the user defined discretization.
   - Temperature of thermal source A (K).
   - Temperature of thermal source B (K).
4. `user_inputs/T_cond.txt`:
   - Conductance temperatures (K).

Fig. 1. List of four files located in the `user_inputs` folder to be modified prior to launching a simulation with the C code. All user-specified inputs are discussed in Section 4. Files 1, 2, and 4 should be modified for a simulation with `sample` spatial discretization (spheres or cubes), whereas files 1, 3, and 4 should be modified for a simulation with `user_defined` spatial discretization.

(a) Specify `refinements` as 1;
(b) Specify `n_0` as the number of subvolumes along the z-axis for the least refined mesh (i.e., largest subvolume size);
(c) Specify `Lx_ref_1` as the length along the x-axis where the discretization is refined;
(d) Specify `n_1` as the number of subvolumes along the z-axis for the most refined mesh (i.e., smallest subvolume size);

For nonuniform discretization with three different subvolume sizes along the x-axis in a given thermal source (see Fig. 4(b)), the user must:

(a) Specify `refinements` as 2;
(b) Specify `n_0` as the number of subvolumes along the z-axis for the least refined mesh (i.e., largest subvolume size);
(c) Specify `Lx_ref_1` as the length along the x-axis where the discretization is firstly refined;
(d) Specify `n_1` as the number of subvolumes along the z-axis for the intermediate refined mesh;
(e) Specify `Lx_ref_2` as the length along the x-axis where the discretization is further refined;
(f) Specify `n_2` as the number of subvolumes along the z-axis for the most refined mesh (i.e., smallest subvolume size);

5. Specify `save_txt` and `save_fig` as 1 to save the `.txt` files and save the discretization figures;
6. Select 1 (0) in `show_axes` and `show_titles` to display (hide) the axes and titles in the figures.

After running the script, the discretized thermal sources and the location of all subvolume center points are provided. The user must copy the two `.txt` files located in the `Discretizations` folder in the DSGF solver at `Library/Discretizations/user_defined` prior to running simulations.

Note that any cubic lattice mesh generators can be used to discretize thermal sources. For example, DDSCAT Convert [53] was used to discretize irregularly shaped particles in Ref. [45]. When using DDSCAT Convert, each thermal source is discretized separately and the subvolume center points are not scaled to a specific characteristic length. A MATLAB script available at https://GitHub.com/Discrete-System-Greens-Function/DSGF_utilities/tree/main/matlab_scripts/Spatial_discretization/user_defined can be used to transfer the mesh generator

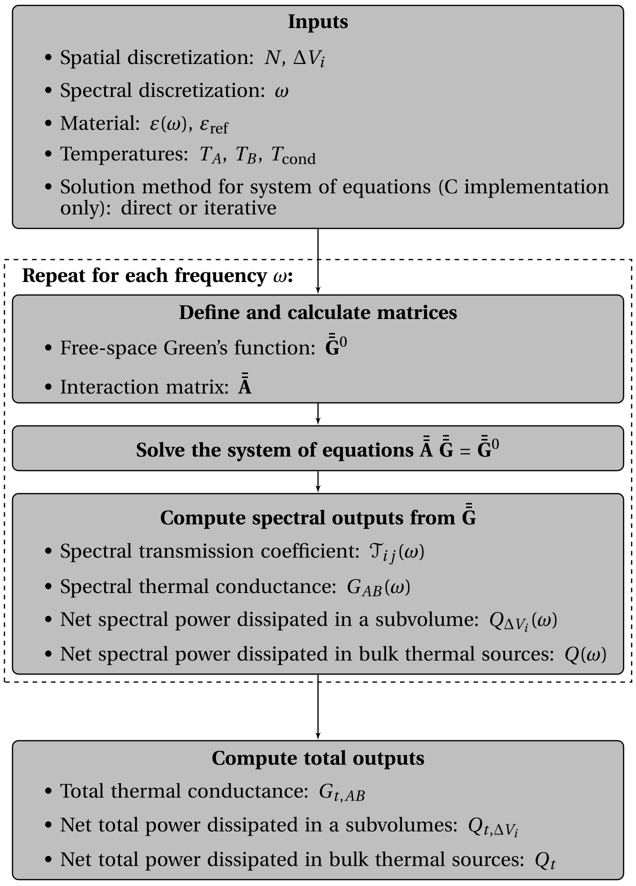
Inputs
• Spatial discretization: N, ΔV_i
• Spectral discretization: ω
• Material: $\varepsilon(\omega)$, $\varepsilon_{\mathrm{ref}}$
• Temperatures: T_A, T_B, T_{cond}
• Solution method for system of equations (C implementation only): direct or iterative
Repeat for each frequency ω:
Define and calculate matrices
• Free-space Green's function: $\bar{\bar{\mathbf{G}}}^0$
• Interaction matrix: $\bar{\bar{\mathbf{A}}}$
Solve the system of equations $\bar{\bar{\mathbf{A}}}\,\bar{\bar{\mathbf{G}}} = \bar{\bar{\mathbf{G}}}^0$
Compute spectral outputs from $\bar{\bar{\mathbf{G}}}$
• Spectral transmission coefficient: $\mathcal{T}_{ij}(\omega)$
• Spectral thermal conductance: $G_{AB}(\omega)$
• Net spectral power dissipated in a subvolume: $Q_{\Delta V_i}(\omega)$
• Net spectral power dissipated in bulk thermal sources: $Q(\omega)$
Compute total outputs
• Total thermal conductance: $G_{t,AB}$
• Net total power dissipated in a subvolumes: $Q_{t,\Delta V_i}$
• Net total power dissipated in bulk thermal sources: Q_t

Fig. 2. Flowchart of a DSGF simulation.

(e.g., DDSCAT Convert) outputs in two `.txt` files containing the subvolume center points $\mathbf{r}_i$ for each subvolume and the ΔV_i, scaled to the actual system dimensions, needed for `user_defined` discretization. Prior to running this script, the discretization file(s) from the mesh generator must be copied into the `single_sources` folder. In the file `user_defined_SCRIPT.m`, the user should:

1. Specify in `discretization` the name of each thermal source discretization file;
2. Specify `L_char` as the characteristic length L_{char} for each thermal source;
3. Specify `d` as the gap spacing;
4. Specify the `origin` of each thermal source;
5. Specify `save_txt` and `save_fig` as 1 to save the `.txt` files and save the discretization figures;
6. Select `1 (0)` in `show_axes` and `show_titles` to display (hide) the axes and titles in the figures.

After running the MATLAB script, the user needs to copy the two `.txt` files located in the `Discretizations` folder, paste them in the DSGF solver at `Library/Discretizations/User_defined`, and type the file names in the appropriate input file (`DSGF_user_inputs.m` in MATLAB and `user_defined.txt` in C). The MATLAB script can be used in combination with any cubic lattice mesh generators that output files in the same format as DDSCAT Convert.

To ensure accurate results, the user should perform a convergence analysis for every modeled system. In this convergence analysis, the number of subvolumes N should be increased until stable results are obtained [42]. A comprehensive convergence analysis that is applicable to the DSGF method is available in Ref. [36].

4.1.2. *Spectral discretization*

The system Green's functions are calculated for each angular frequency ω. The frequency range and frequency discretization (uniform or nonuniform) must be specified by the user.

The angular frequencies (units of rad/s) are imported from a spectral discretization file stored at `Library/Spectral_discretizations/`, where spectral discretization examples are available. For the MATLAB code, the user must type the file name with its extension (e.g., `.csv`) in `DSGF_user_inputs.m`. For the C code, the user needs to type the number of frequencies and the imported file name with its extension in the `control.txt` file.

MATLAB scripts for generating uniform (`uniform_spectra.m`) and nonuniform (`nonuniform_spectra.m`) spectral discretization are available in the folder `Library/Spectral_discretizations/` and at https://GitHub.com/Discrete-System-Greens-Function/DSGF_utilities/tree/main/matlab_scripts/spectral_discretization. For both uniform and nonuniform discretization files, the user should uncomment the desired material, and modify the range and the number of frequencies prior to running the script. Once completed, a `.csv` file is generated in the folder `Spectral_discretizations` and the dielectric function is displayed as a function of the angular frequency. The file name combines the selected material, the number of frequencies, the spectral discretization type (uniform or nonuniform), and the spectral range with its original units (e.g., `SiO2_100_uniform_5_25_um.csv`). The data file must be copied in the DSGF solver in the `Library/Spectral_discretizations` folder prior to running simulations. Note that any data file can be used as long as the frequencies are in units of rad/s and are listed in a single column.

4.1.3. *Material selection*

With the $e^{-i\omega t}$ convention, the dielectric function of a subvolume is defined as $\varepsilon(\omega) = \varepsilon'(\omega) + i\varepsilon''(\omega)$. The dielectric functions of SiC, SiO_2, and silicon nitride (Si_3N_4) are currently available in the DSGF solver. The default dielectric function of the lossless background reference medium is $\varepsilon_{\mathrm{ref}} = 1$ (i.e., vacuum), and it can be changed under the restriction that it must be real-valued.

4.1.4. *Temperature selection*

Two types of temperatures must be specified. The first type corresponds to the temperatures of the two distinct groups of thermal sources, T_A and T_B. These temperatures are used to calculate the net power dissipated. The second type corresponds to the temperatures at which the thermal conductance is calculated between two groups of thermal sources, T_{cond}. The user can define multiple T_{cond} values in a single simulation. All temperatures are defined in units of kelvin.

4.2. *Solution of the system of linear equations*

The system of linear equations (Eq. (5)) can be solved by direct inversion in both codes, and via an iterative procedure in the C implementation only. The user must select the solution method prior to running simulations with the C code.

4.2.1. *Direct inversion*

Direct inversion is the most straightforward solution approach. For each frequency, the system Green's functions are obtained by directly inverting the interaction matrix: $\bar{\bar{\mathbf{G}}} = \bar{\bar{\mathbf{A}}}^{-1}\bar{\bar{\mathbf{G}}}^0$.

4.2.2. *Iterative procedure*

The iterative procedure is based on the exact scheme proposed by Martin et al. [51,52]. In this paradigm, the system Green's function between subvolumes i and j is calculated iteratively by considering the electromagnetic contribution of each subvolume successively, from $m = 1$ to N. For each step m in the iterative procedure, a small 3×3 system of linear equations is solved. The iterative solver thus reduces

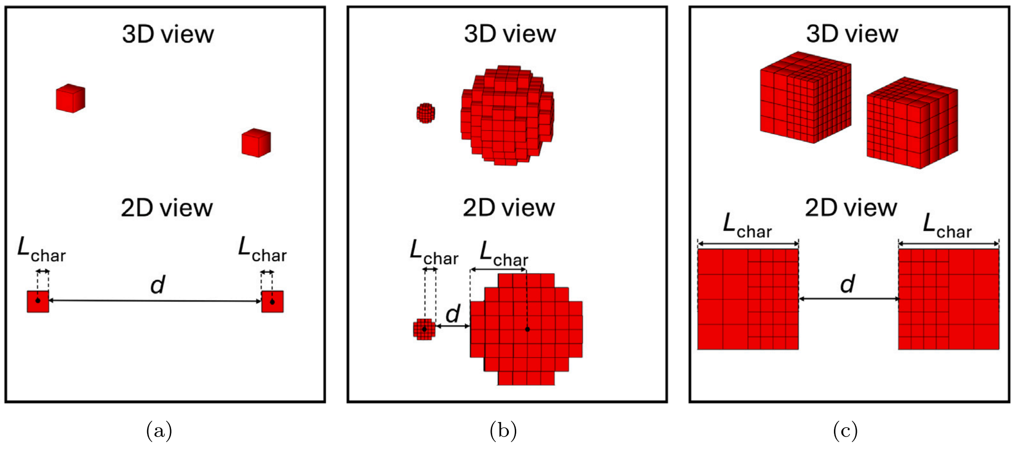

Fig. 3. Spatial discretization using cubic subvolumes along a cubic lattice for different geometries: (a) two `sample` dipoles (one subvolume per thermal source), (b) two `sample` spheres (136 and 280 uniform subvolumes for the left and right spheres, respectively), and (c) two `user_defined` cubes (288 nonuniform subvolumes per cube). The thermal sources are separated by an edge-to-edge gap d. The radius is the characteristic length (L_{char}) for dipoles and spheres, whereas the side length is the L_{char} for cubes.

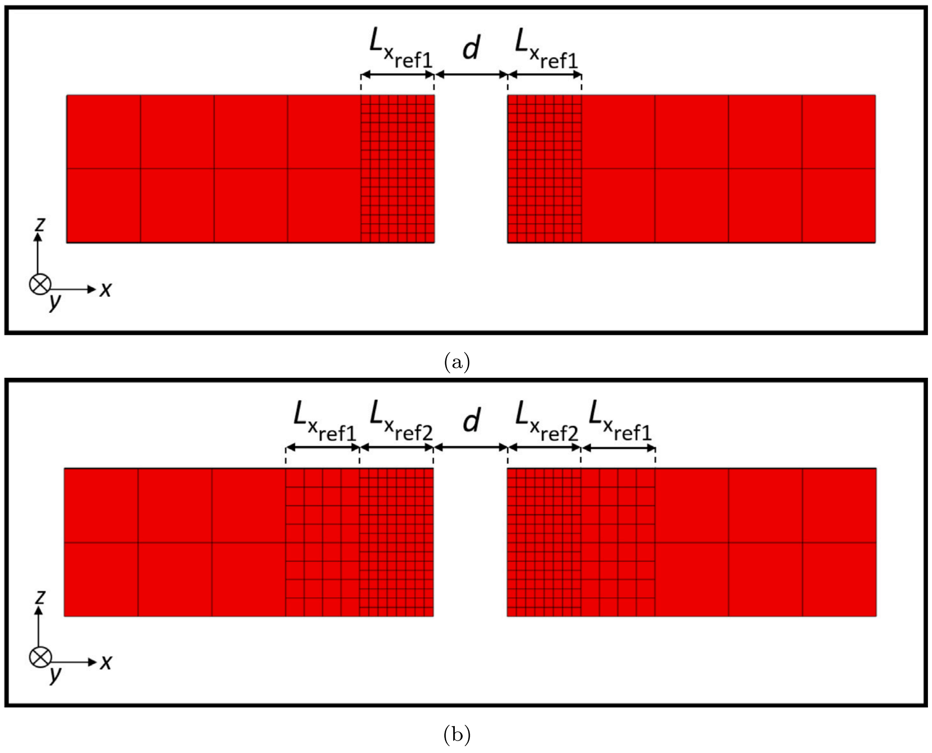

Fig. 4. Example of nonuniform spatial discretizations. (a) Nonuniform discretization with two different subvolume sizes. The least refined mesh has two subvolumes along the z-axis (`n_0 = 2`) and the most refined mesh is characterized by 16 subvolumes along the z-axis (`n_1 = 16`). (b) Nonuniform discretization with three different subvolume sizes. The least refined mesh has two subvolumes along the z-axis (`n_0 = 2`), the intermediate refined mesh has 8 subvolumes along the z-axis (`n_1 = 8`) and the most refined mesh is characterized by 16 subvolumes along the z-axis (`n_2 = 16`).

the memory usage at the expense of longer computational time, as detailed next.

4.2.3. Comparison between direct inversion and iterative procedure in the C code

The iterative procedure significantly reduces the memory required for solving the system of linear equations (Eq. (5)) when compared with direct inversion (Fig. 5(a)). For example, a simulation with $N = 11{,}232$ subvolumes requires 76 GB of RAM using direct inversion. This number drops to 39 GB when the iterative procedure is employed. The memory usage does not vary with the number of frequencies because the system Green's functions are calculated independently at each frequency. The reduced memory usage from the iterative procedure comes at the expense of longer computational time (Fig. 5(b)). For instance, a one-frequency simulation with $N = 11{,}232$ subvolumes takes ~31 min in MATLAB, ~26 min using direct inversion in C, and ~7 h with the iterative procedure in C, when using 16 cores and 256 GB of RAM.

When the available computational resources can handle the memory requirements, direct inversion is recommended. Note that Fig. 5(a) can be used to determine the maximum number of subvolumes that can

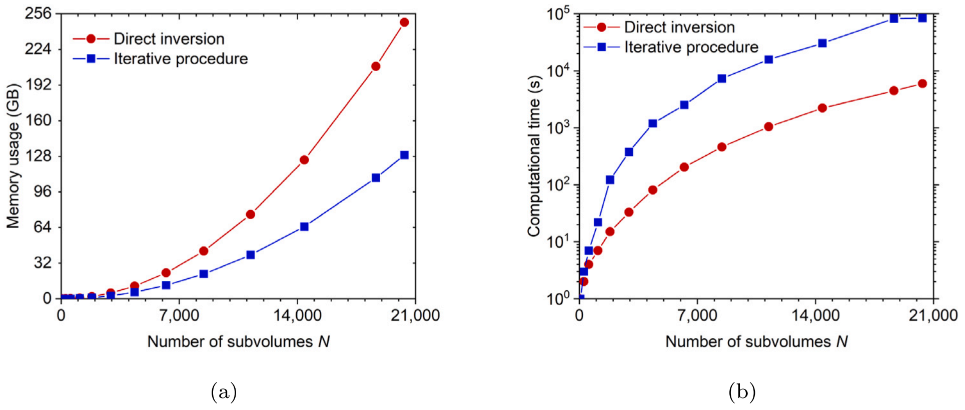

Fig. 5. Sample DSGF simulations with the C code for one frequency: (a) memory usage (GB) and (b) computational time (s) as a function of the number of subvolumes N for direct inversion (red) and the iterative procedure (blue). The results were generated using 64 cores and 256 GB of RAM.

be handled with direct inversion for a specific amount of memory. When the number of subvolumes is too large to be handled with direct inversion, a message appears in the terminal and the iterative procedure must be used.

5. Examples

The results presented in this section were generated using a desktop computer with 16 GB of RAM.

5.1. Tutorial 1: NFRHT between two cubes of SiO_2

NFRHT between two cubes made of SiO_2 embedded in vacuum is considered. Both cubes are characterized by side lengths of 500 nm and are separated by a gap of 100 nm. The spectral discretization file for these simulations is already available. For completeness, the user can also generate the spectral discretization file by editing `uniform_spectra.m` indicated in Section 4.1.2, uncommenting the block of code “SiO2 in terms of wavelength”, running the script, and copying the file in the `library/spectral_discretizations` folder.

To perform this simulation with the MATLAB code, the user must edit `DSGF_user_inputs.m` and define the following inputs:

1. Write a description of the simulation such as `‘‘2_cubes_SiO2_Lchar_500nm_d_100nm_N_2000’’`;
2. Specify `spatial_discretization_type` as `sample`;
3. Specify `discretization` as `{Discretization.cube_1000, Discretization.cube_1000}`;
4. Specify `L_char` as `[500e-9, 500e-9]`;
5. Specify `d` as `100e-9`;
6. Specify `material` as `Material.SiO2`;
7. Specify `epsilon_ref` as `1`;
8. Specify `spectral_discretization` as `‘‘SiO2_100_uniform_5_25_um.csv’’`;
9. Specify `T` as `[300, 800]` and `T_cond` as `[200, 250, 300, 350, 400]`;
10. Specify `output.save_workspace` as `true` to save the results in a MATLAB workspace file;
11. Specify `output.save_fig` as `true` to save the figures;
12. Specify the figure format with `output.figure_format`. We recommend `FigureFormat.fig` because the figure size can be adjusted after the simulation is completed. The options `FigureFormat.png` and `FigureFormat.jpg` are also possible;
13. Specify `conductance`, `power_dissipated_subvol`, and `power_dissipated_bulk` as `true`. These outputs are discussed in Section 2;
14. Specify `output.heatmap_sliced` as `false`. This option enables visualization of the internal power dissipated in thermal sources sliced in two parts;
15. Specify `DSGF_matrix` and `transmission_coefficient_matrix` as `false`. These outputs correspond to Eqs. (5) and (7), respectively.

Once the inputs are defined, press the `Run` button located at the top of the MATLAB editor. When the simulation is initiated, a results folder named with the date and time from when the simulation started is created. Once the simulation is completed, the results will be available in a MATLAB workspace data file inside the results folder.

The steps to run the same simulation with the C code are:

1. Go to `control.txt` and edit the file (Steps 2 to 18);
2. Specify `spatial_discretization_type` as `sample`;
3. Specify `number_subvolumes_per_object_1` as `1000`;
4. Specify `number_subvolumes_per_object_2` as `1000`;
5. Specify `material` as `SiO2` and `epsilon_ref` as `1`;
6. Specify `number_omega` as `100`;
7. Specify `spectral_discretization` as `SiO2_100_uniform_5_25_um.csv`;
8. For `solution`, select between `D` for direct inversion or `I` for iterative solution;
9. Specify `multithread` as `Y`;
10. Specify `single_spectrum_analysis` as `N`;
11. Specify `save_spectral_conductance` as `Y`;
12. Specify `save_total_conductance` as `Y`;
13. Specify `save_power_dissipated_spectral_subvolumes` as `Y`;
14. Specify `save_power_dissipated_total_subvolumes` as `Y`;
15. Specify `save_power_dissipated_spectral_bulk` as `Y`;
16. Specify `save_power_dissipated_total_bulk` as `Y`;
17. Specify `save_power_density_total_subvolumes` as `Y`;
18. Specify `save_spectral_transmissivity` as `N`;
19. Go to `sample.txt` and edit the file (Steps 20 to 23);
20. Specify `geometry_1` as `cube` and `L_char_1` as `500.e-9`;
21. Specify `geometry_2` as `cube` and `L_char_2` as `500.e-9`;
22. Specify `d` as `100.e-9`;
23. Specify `T_A` as `300` and `T_B` as `800`;
24. Go to `T_cond.txt` and specify temperatures of 200, 250, 300, 350, 400;
25. Go to the terminal and type `\.DSGF`.

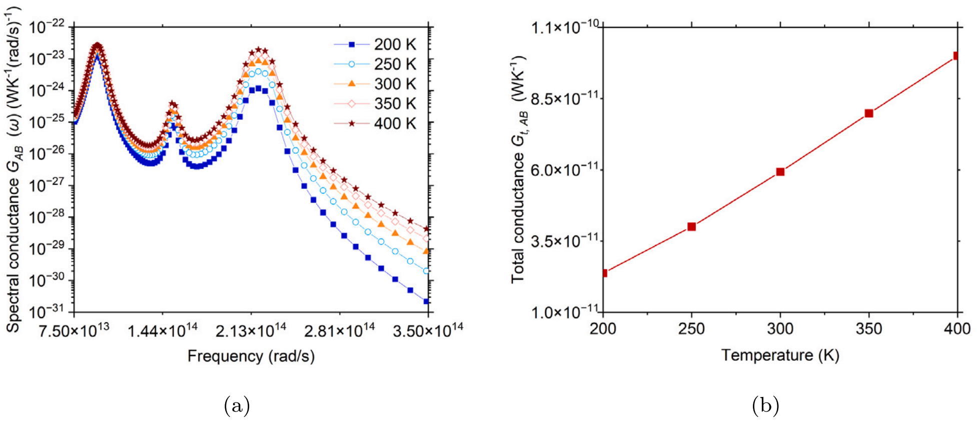

Fig. 6. (a) Spectral and (b) total conductances for two SiO_2 cubes characterized by side lengths of 500 nm and separated by a gap of 100 nm. The conductance is reported for temperatures of 200 K, 250 K, 300 K, 350 K, and 400 K.

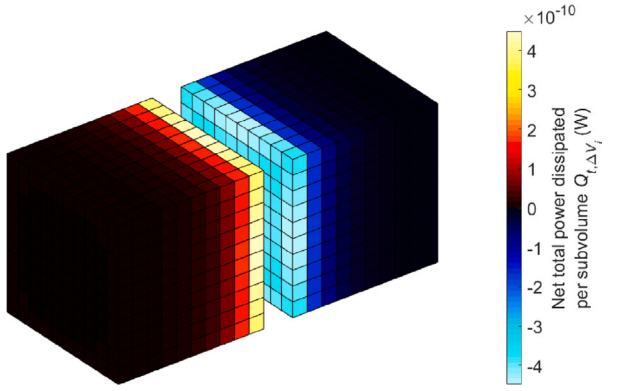

Fig. 7. Spatial distribution of net total power dissipated for two SiO_2 cubes characterized by side lengths of 500 nm and separated by a gap of 100 nm. Negative values represent heat loss, and positive values represent heat gain. The low-temperature cube (red color scheme since thermal energy is gained) is at 300 K, whereas the high-temperature cube (blue color scheme since thermal energy is lost) is at 800 K.

Table 1
Temperature-dependent total thermal conductance for two membranes of SiC.

Temperature (K)	$G_{t,AB}$ (W/K)
200	7.5245×10^{-12}
250	1.7183×10^{-11}
300	2.8210×10^{-11}
350	3.8839×10^{-11}
400	4.8273×10^{-11}

For the C code, the results folder is named with the type of simulation (i.e., sample or user defined), number of subvolumes, material, gap spacing, and date.

The spectral and total thermal conductances for the two-cube problem are shown in Fig. 6. For example, the total conductance at 300 K is 5.9286×10^{-11} W/K, and the net total power dissipated for temperatures of 300 K and 800 K is $\pm 7.1671 \times 10^{-8}$ W. Note that the negative sign applies to the high-temperature cube losing heat, and vice-versa for the low-temperature cube (see Fig. 7).

5.2. Tutorial 2: NFRHT between two membranes of SiC

NFRHT between two membranes made of SiC embedded in vacuum is considered (see Fig. 4 for system geometry). Both membranes are characterized by a thickness of 100 nm, widths and lengths of 1 μm and are separated by a gap of 100 nm. This simulation relies on a user defined spatial discretization type and a nonuniform spectral discretization. The necessary spatial and spectral discretization files are already available, but instructions to generate these files are provided hereafter for completeness.

As indicated in Section 4.1.1, `cuboid_discretization_SCRIPT.m` must be edited in order to generate the spatial discretization:

1. Specify `description` as `‘‘2_membranes’’`;
2. Specify `d` as `100e-9`;
3. Specify `Lx` as `1000e-9`;
4. Specify `Ly` as `1000e-9`;
5. Specify `Lz` as `100e-9`;
6. Specify `refinement` as `0`;
7. Specify `n0` as `2`;
8. After running the script, the two `.txt` files located in the `Discretizations` folder in the DSGF solver must be copied at `Library/Discretizations/user_defined`.

As indicated in Section 4.1.2, `nonuniform_spectra.m` must be edited by the user in order to generate the spectral discretization, and the block of code “SiC in terms of angular frequency for two membranes” must be uncommented. After running the script, the `.csv` file located in the `Spectral_discretizations` folder must be copied in the DSGF solver at `Library/Spectral_discretizations`.

To perform the DSGF simulation with the MATLAB code, the user must edit `DSGF_user_inputs.m` and define the following inputs:

1. Write a description of the simulation such as `‘‘2_membranes_SiC_Lx_1um_Ly_1um_Lz_100nm_d_100nm_N_1600’’`;
2. Specify `spatial_discretization_type` as `user_defined`;
3. Specify `discretization` as `‘‘2_membranes_Lx1000nm_Ly1000nm_Lz100nm_d100nm_N1600_discretization’’`;
4. Specify `delta_V` as `‘‘2_membranes_Lx1000nm_Ly1000nm_Lz100nm_d100nm_N1600_delta_V_vector’’`;
5. Specify `material` as `Material.SiC`;
6. Specify `epsilon_ref` as `1`;
7. Specify `spectral_discretization` as `‘‘SiC_100_nonuniform_140_190_Trad_s.csv’’`;
8. Specify `T` as `[400, 300]` and `T_cond` as `[200, 250, 300, 350, 400]`;
9. Specify `output.save_workspace` as `true`;
10. Specify `output.save_fig` as `true`;
11. Specify `output.figure_format` as `FigureFormat.fig`;
12. Specify `conductance`, `power_dissipated_subvol`, and `power_dissipated_bulk` as `true`;

Table 2
Total thermal conductance at 300 K and net total power dissipated for three NFRHT examples.

Outputs	Two dipoles of SiC	Two spheres of SiO_2	Two cubes of Si_3N_4
$G_{t,AB}$ at 300 K (W/K)	1.7898×10^{-16}	3.4408×10^{-13}	3.8794×10^{-13}
Q_t (W)	$\pm 2.5199 \times 10^{-14}$	$\pm 2.2220 \times 10^{-10}$	$\pm 5.5891 \times 10^{-11}$

13. Specify `output.heatmap_sliced` as `false`;
14. Specify `DSGF_matrix` and `transmission_coefficient_matrix` as `false`;

The steps to run the same simulation with the C code are:

1. Go to `control.txt` and edit the file (Steps 2 to 18);
2. Specify `spatial_discretization_type` as `user_defined`;
3. Specify `number_subvolumes_per_object_1` as 800;
4. Specify `number_subvolumes_per_object_2` as 800;
5. Specify `material` as `SiC` and `epsilon_ref` as 1;
6. Specify `number_omega` as `100`;
7. Specify `spectral_discretization` as `SiC_100_nonuniform_140_190_Trad_s.csv`;
8. For `solution`, select between `D` for direct inversion or `I` for iterative solution;
9. Specify `multithread` as `Y`;
10. Specify `single_spectrum_analysis` as `N`;
11. Specify `save_spectral_conductance` as `Y`;
12. Specify `save_total_conductance` as `Y`;
13. Specify `save_power_dissipated_spectral_subvolumes` as `Y`;
14. Specify `save_power_dissipated_total_subvolumes` as `Y`;
15. Specify `save_power_dissipated_spectral_bulk` as `Y`;
16. Specify `save_power_dissipated_total_bulk` as `Y`;
17. Specify `save_power_density_total_subvolumes` as `Y`;
18. Specify `save_spectral_transmissivity` as `N`;
19. Go to `user_defined.txt` and edit the file (Steps 20 to 22);
20. Specify `d` as `100.e-9`;
21. Specify `file_name` as `2_membranes_Lx1000nm_Ly1000nm_Lz100nm_d_100nm_N1600`;
22. Specify `T_A` as `400`, and `T_B` as `300`;
23. Go to `T_cond.txt` and specify temperatures of `200, 250, 300, 350, 400`;
24. Go to the terminal and type `\.DSGF`.

In this example, the net total power dissipated in bulk thermal sources at temperatures of 300 K and 400 K is $\pm 3.8636 \times 10^{-9}$ W. The temperature-dependent total conductance is provided in Table 1.

5.3. Additional examples

Table 2 provides the total thermal conductance at 300 K and the net total power dissipated for three different examples.

The first example involves two dipoles made of SiC maintained at $T_A = 300$ K and $T_B = 400$ K. Both dipoles are characterized by radii of 10 nm and are separated by $d = 100$ nm; this configuration is shown in Fig. 3(a). The spectrum is uniformly discretized into 100 frequencies within the spectral band of from 140 to 190 Trad/s.

Second, NFRHT between two spheres made of SiO_2 maintained at $T_A = 600$ K and $T_B = 250$ K is considered. Sphere A is characterized by a radius of 50 nm and is discretized into 136 subvolumes. Sphere B has a radius of 250 nm and is discretized into 280 subvolumes. The spheres are separated by $d = 150$ nm (see Fig. 3(b)). For SiO_2, the angular frequencies are defined based on uniformly discretizing the spectral band of from 5 to 25 μm into 100 wavelengths.

The last example is a user defined simulation with nonuniform spatial discretization involving two cubes of Si_3N_4 maintained at $T_A = 400$ K and $T_B = 300$ K (see Fig. 3(c)). Each cube is characterized by a side length of 500 nm, discretized into 288 subvolumes, and separated by $d = 500$ nm. Two different subvolume sizes are defined in each cube: `n_0 = 4`, `Lx_ref_1 = 250e-9`, and `n_1 = 8` (see Section 4.1.1). The two spatial discretization files are located in `Library/Discretizations/User_defined` under the name `2_cubes_Lx500nm_Ly500nm_Lz500nm_d500nm_N576`. The spectral band of from 20 to 300 Trad/s is uniformly discretized into 100 frequencies.

Note that in addition to the problems provided in Section 5, the solver can also be tested for the case of two SiO_2 spheres described in Ref. [45], where DSGF results are compared against exact results based on an analytical solution.

6. Conclusions

This paper described MATLAB and C implementations of the DSGF solver for NFRHT publicly available on GitHub. In theory, the DSGF method can simulate an arbitrary number of thermal sources of arbitrary shape, size, and material. In practice, the number of thermal sources and their size are limited by memory requirements. The memory usage in the C implementation of the DSGF solver is mitigated when using the iterative procedure, but this comes at the expense of significantly longer computational time. The DSGF solver implemented in MATLAB and C are open-source codes, and users are highly encouraged to upgrade these codes. Since the key bottleneck preventing large-scale simulations of NFRHT with the DSGF solver is the solution of the system of linear equations, a useful upgrade would be to implement a solution procedure mitigating the memory usage without sacrificing too much on the computational time.

CRediT authorship contribution statement

Lívia M. Corrêa: Writing – original draft, Methodology, Investigation, Formal analysis, Conceptualization. **Lindsay P. Walter:** Writing – review & editing, Software, Methodology, Conceptualization. **Jan L. Čas:** Software, Methodology, Formal analysis. **Mathieu Francoeur:** Writing – review & editing, Supervision, Resources, Project administration, Funding acquisition, Data curation, Conceptualization.

Declaration of competing interest

The authors declare the following financial interests/personal relationships which may be considered as potential competing interests: The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

Codes and data are available on GitHub.

Acknowledgments

This work was supported by the National Science Foundation (Grant No. CBET-1952210). L.M.C. acknowledges the American Association of University Women (AAUW) for the 2022–2023 international PhD fellowship. L.P.W. acknowledges that this material is based upon work supported by the National Science Foundation Graduate Research Fellowship under Grant No. DGE-1747505. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the National Science Foundation. M.F. acknowledges the Natural Sciences and Engineering Research Council of Canada (funding reference number RGPIN-2023-03513). The resources from the Center for High Performance Computing at The University of Utah and the support of Dr. Martin Cuma are gratefully acknowledged.

References

[1] Francoeur M. Near-field thermal radiation. In: Handbook of thermal science and engineering. Springer; 2017, p. 1–43.
[2] Polder D, Van Hove M. Theory of radiative heat transfer between closely spaced bodies. Phys Rev B 1971;4(10):3303.
[3] Loomis JJ, Maris HJ. Theory of heat transfer by evanescent electromagnetic waves. Phys Rev B 1994;50(24):18517.
[4] Mulet J-P, Joulain K, Carminati R, Greffet J-J. Enhanced radiative heat transfer at nanometric distances. Microsc Thermophys Eng 2002;6(3):209–22.
[5] Rousseau E, Siria A, Jourdan G, Volz S, Comin F, Chevrier J, Greffet J-J. Radiative heat transfer at the nanoscale. Nat Photonics 2009;3:514–7.
[6] St-Gelais R, Zhu L, Fan S, Lipson M. Near-field radiative heat transfer between parallel structures in the deep subwavelength regime. Nature Nanotechnology 2016;11(6):515–9.
[7] Ghashami M, Geng H, Kim T, Iacopino N, Cho SK, Park K. Precision measurement of phonon-polaritonic near-field energy transfer between macroscale planar structures under large thermal gradients. Phys Rev Lett 2018;120:175901.
[8] DeSutter J, Tang L, Francoeur M. A near-field radiative heat transfer device. Nature Nanotechnology 2019;14:751–5.
[9] Fiorino A, Zhu L, Thompson D, Mittapally R, Reddy P, Meyhofer E. Nanogap near-field thermophotovoltaics. Nature Nanotechnology 2018;13(9):806–11.
[10] Inoue T, Koyama T, Kang DD, Ikeda K, Asano T, Noda S. One-chip near-field thermophotovoltaic device integrating a thin-film thermal emitter and photovoltaic cell. Nano Lett 2019;19(6):3948–52.
[11] Mittapally R, Lee B, Zhu L, Reihani A, Lim JW, Fan D, Forrest SR, Reddy P, Meyhofer E. Near-field thermophotovoltaics for efficient heat to electricity conversion at high power density. Nature Commun 2021;12(1):4364.
[12] Lucchesi C, Cakiroglu D, Perez J-P, Taliercio T, Tournié E, Chapuis P-O, Vaillon R. Near-field thermophotovoltaic conversion with high electrical power density and cell efficiency above 14%. Nano Lett 2021;21(11):4524–9.
[13] Guha B, Otey C, Poitras CB, Fan S, Lipson M. Near-field radiative cooling of nanostructures. Nano Lett 2012;12(9):4546–50.
[14] Otey CR, Lau WT, Fan S, et al. Thermal rectification through vacuum. Phys Rev Lett 2010;104(15):154301.
[15] Basu S, Francoeur M. Near-field radiative transfer based thermal rectification using doped silicon. Appl Phys Lett 2011;98(11):113106.
[16] Wang L, Zhang Z. Thermal rectification enabled by near-field radiative heat transfer between intrinsic silicon and a dissimilar material. Nanoscale Microscale Thermophys Eng 2013;17(4):337–48.
[17] Fiorino A, Thompson D, Zhu L, Mittapally R, Biehs S-A, Bezencenet O, El-Bondry N, Bansropun S, Ben-Abdallah P, Meyhofer E, et al. A thermal diode based on nanoscale thermal radiation. ACS Nano 2018;12(6):5774–9.
[18] Rytov SM, Kravtsov YA, Tatarskii VI. Principles of statistical radiophysics. 4. Wave propagation through random media. Springer-Verlag; 1989.
[19] Francoeur M, Mengüç MP, Vaillon R. Solution of near-field thermal radiation in one-dimensional layered media using dyadic Green's functions and the scattering matrix method. J Quant Spectrosc Radiat Transfer 2009;110(18):2002–18.
[20] Ben-Abdallah P, Biehs S-A, Joulain K. Many-body radiative heat transfer theory. Phys Rev Lett 2011;107(11):114301.
[21] Tervo E, Francoeur M, Cola B, Zhang Z. Thermal radiation in systems of many dipoles. Phys Rev B 2019;100(20):205422.
[22] Biehs S-A, Messina R, Venkataram PS, Rodriguez AW, Cuevas JC, Ben-Abdallah P. Near-field radiative heat transfer in many-body systems. Rev Modern Phys 2021;93(2):025009.
[23] Walter LP, Francoeur M. Generalized many-body approach for near-field radiative heat transfer between nonspherical particles. 2024, arXiv:2403.07114.
[24] Narayanaswamy A, Chen G. Thermal near-field radiative transfer between two spheres. Phys Rev B 2008;77(7):075125.
[25] Czapla B, Narayanaswamy A. Thermal radiative energy exchange between a closely-spaced linear chain of spheres and its environment. J Quant Spectrosc Radiat Transfer 2019;227:4–11.
[26] Wen S-B. Direct numerical simulation of near field thermal radiation based on Wiener chaos expansion of thermal fluctuating current. J Heat Transfer 2010;132(7):072704.
[27] Jin W, Molesky S, Lin Z, Rodriguez AW. Material scaling and frequency-selective enhancement of near-field radiative heat transfer for lossy metals in two dimensions via inverse design. Phys Rev B 2019;99(4):041403.
[28] Rodriguez AW, Ilic O, Bermel P, Celanovic I, Joannopoulos JD, Soljačić M, Johnson SG. Frequency-selective near-field radiative heat transfer between photonic crystal slabs: a computational approach for arbitrary geometries and materials. Phys Rev Lett 2011;107(11):114302.
[29] Liu B, Shen S. Broadband near-field radiative thermal emitter/absorber based on hyperbolic metamaterials: Direct numerical simulation by the Wiener chaos expansion method. Phys Rev B 2013;87(11):115403.
[30] Didari A, Mengüç MP. Analysis of near-field radiation transfer within nano-gaps using FDTD method. J Quant Spectrosc Radiat Transfer 2014;146:214–26.
[31] Didari A, Mengüç MP. A design tool for direct and non-stochastic calculations of near-field radiative transfer in complex structures: The NF-RT-FDTD algorithm. J Quant Spectrosc Radiat Transfer 2017;197:95–105.
[32] Rodriguez AW, Reid MH, Johnson SG. Fluctuating-surface-current formulation of radiative heat transfer for arbitrary geometries. Phys Rev B 2012;86(22):220302.
[33] Rodriguez AW, Reid MH, Johnson SG. Fluctuating-surface-current formulation of radiative heat transfer: theory and applications. Phys Rev B 2013;88(5):054305.
[34] SCUFF-EM. 2024, http://homerreid.github.io/scuff-em-documentation/. [Accessed 07 March 2024].
[35] Edalatpour S, Francoeur M. The Thermal Discrete Dipole Approximation (T-DDA) for near-field radiative heat transfer simulations in three-dimensional arbitrary geometries. J Quant Spectrosc Radiat Transfer 2014;133:364–73.
[36] Edalatpour S, Čuma M, Trueax T, Backman R, Francoeur M. Convergence analysis of the thermal discrete dipole approximation. Phys Rev E 2015;91(6):063307.
[37] Edalatpour S, DeSutter J, Francoeur M. Near-field thermal electromagnetic transport: an overview. J Quant Spectrosc Radiat Transfer 2016;178:14–21.
[38] Edalatpour S, Francoeur M. Near-field radiative heat transfer between arbitrarily shaped objects and a surface. Phys Rev B 2016;94(4):045406.
[39] Ekeroth RA, García-Martín A, Cuevas JC. Thermal discrete dipole approximation for the description of thermal emission and radiative heat transfer of magneto-optical systems. Phys Rev B 2017;95(23):235428.
[40] Edalatpour S. Near-field thermal emission by periodic arrays. Phys Rev E 2019;99(6):063308.
[41] Edalatpour S, Hatamipour V, Francoeur M. Spectral redshift of the thermal near field scattered by a probe. Phys Rev B 2019;99(16):165401.
[42] Walter LP, Tervo EJ, Francoeur M. The thermal discrete dipole approximation and the discrete system Green's function methods for computational near-field radiative heat transfer. In: Light, plasmonics and particles. Elsevier; 2023, p. 223–47.
[43] Polimeridis AG, Reid MH, Jin W, Johnson SG, White JK, Rodriguez AW. Fluctuating volume-current formulation of electromagnetic fluctuations in inhomogeneous media: Incandescence and luminescence in arbitrary geometries. Phys Rev B 2015;92(13):134202.
[44] BUFF-EM. 2024, https://homerreid.github.io/buff-em-documentation/. [Accessed 07 March 2024].
[45] Walter LP, Tervo EJ, Francoeur M. Near-field radiative heat transfer between irregularly shaped dielectric particles modeled with the discrete system Green's function method. Phys Rev B 2022;106(19):195417.
[46] Walter LP, Francoeur M. Orientation effects on near-field radiative heat transfer between complex-shaped dielectric particles. Appl Phys Lett 2022;121(18):182206.
[47] Chen K, Zhao B, Fan S. MESH: A free electromagnetic solver for far-field and near-field radiative heat transfer for layered periodic structures. Comput Phys Comm 2018;231:163–72.
[48] DSGF solver. 2024, https://github.com/Discrete-System-Greens-Function. [Accessed 07 March 2024].
[49] Tang L, Corrêa LM, Francoeur M, Dames C. Corner-and edge-mode enhancement of near-field radiative heat transfer. Nature 2024;629:67–73.
[50] Novotny L, Hecht B. Principles of nano-optics. Cambridge University Press; 2012.
[51] Martin OJ, Dereux A, Girard C. Iterative scheme for computing exactly the total field propagating in dielectric structures of arbitrary shape. J Opt Soc Am 1994;11(3):1073–80.
[52] Martin OJ, Girard C, Dereux A. Generalized field propagator for electromagnetic scattering and light confinement. Phys Rev Lett 1995;74(4):526.
[53] DDSCAT convert. 2024, https://nanohub.org/resources/ddaconvert/about. [Accessed 25 June 2024].