Accelerating Stochastic Lightning Attachment Simulations for the Estimation of Lightning Incidence to Overhead Lines

Alexios I. Ioannidis, *Student Member, IEEE*, Zacharias G. Datsios, *Member, IEEE*, Apostolos K. Gerodimos, and Thomas E. Tsovilis, *Senior Member, IEEE*

***Abstract*—Lightning attachment can be modeled through a stochastic approach adopting a detailed representation of the lightning phenomenon. A fractal-based modeling technique can be used for this purpose, considering lightning discharge branched and tortuous behavior, as well as physical properties associated with downward and upward leaders' inception and propagation. However, fractal-based simulations require substantial computational resources, especially for the accurate calculation of the electric field at all points of the discretized simulation domain at each simulation step. Thus, the considerable computational cost inhibits the extensive application of stochastic simulations for estimating lightning incidence to common structures and power systems. This work investigates optimization techniques for fractal-based simulations regarding total simulation time; these are applicable to both high-performance computing and personal computers. The proposed techniques consist of a C-MATLAB integration methodology, as well as a multi-color ordering algorithm enabling parallel execution using CPU and GPU programming. Applications associated with lightning incidence to overhead transmission lines are presented. The total simulation time is substantially reduced with respect to the original code. A reduction of up to 98% is achieved, enhancing the applicability of stochastic modeling to lightning incidence estimation problems.**

***Index Terms*—Code optimization, fractals, lightning attachment model, lightning incidence, overhead lines (OHLs), simulation time.**

I. Introduction

LIGHTNING is a physical phenomenon that can cause severe damage to human life and infrastructure. Regarding power systems, lightning protection is of crucial importance for their reliability and continuous operation preventing unscheduled power supply interruptions and outages that can have a huge economic impact [1]; thus, this subject still attracts wide attention [2], [3], [4]. An accurate and comprehensive lightning protection study requires the use of a lightning attachment model [5], [6], [7] to assess lightning incidence. Such models are mainly categorized into the so-called electrogeometric models (EGM) and leader propagation models (LPM).

More recently, LPM- and fractal-based lightning attachment models [8], [9], [10], [11], [12], [13] have been developed using advanced simulation techniques and computational electromagnetic methods to enable a detailed representation of the lightning phenomenon in 3D domains. Thus, simplifications of the EGMs, which consider a geometric approach of the final step of lightning, may be overcome. However, the complexity of the natural processes involved and the huge simulation domains (3D complex geometries) usually lead to excessive simulation times. This may hinder the applicability of these models to practical engineering problems.

A few studies have been conducted proposing techniques to reduce the computational time needed for lightning attachment simulations. These focus mainly on adaptive strategies for appropriate mesh size selection [14], [15], "tricks" to accelerate the computational electromagnetics methods employed [16], or teaching learning-based techniques [17], [18]. However, a systematic application of stochastic models for lightning incidence estimation calls for further investigations.

This study deals with the optimization of stochastic lightning attachment simulations to reduce simulation time. A fractal-based model is investigated, considering the physical processes involved in lightning attachment, as well as the tortuous and branched leader propagation. However, the associated computational cost of such models is substantial as electric field calculation is required at all points of the discretized simulation domain at each simulation step. The investigated code optimization techniques comprise a C-MATLAB integration methodology, as well as a multi-color ordering algorithm, developed to enable parallel execution using central processing unit (CPU) and graphics processing unit (GPU) programming. The proposed techniques are not tied to the specific model and are applicable to both high-performance computing (HPC) servers and personal computers. They may also be utilized in various problems employing finite differences to solve partial differential equations

[1]

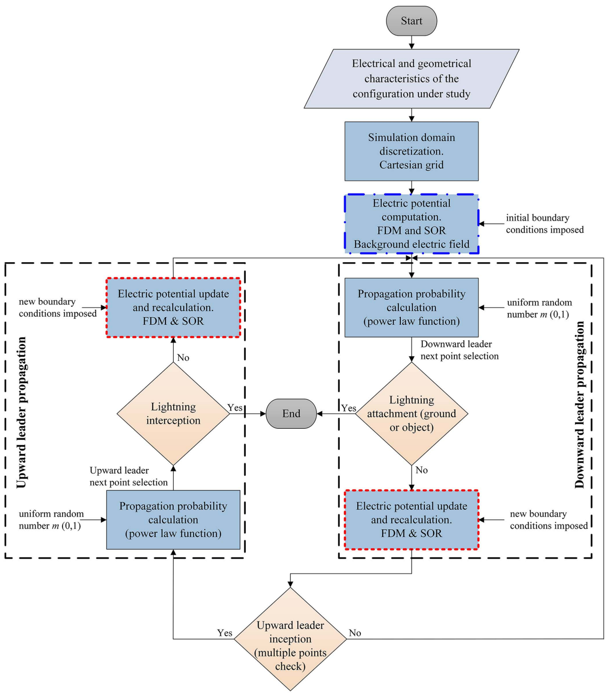
Manuscript received 27 March 2022; revised 23 September 2022; accepted 3 January 2023. Date of publication 3 February 2023; date of current version 13 June 2023. The research project was supported by the Hellenic Foundation for Research and Innovation (H.F.R.I.) under the "2nd Call for H.F.R.I. Research Projects to support Post-Doctoral Researchers" (Project Number: 367).[1] *(Corresponding author: Alexios I. Ioannidis.)*

The authors are with the High Voltage Laboratory, School of Electrical & Computer Engineering, Aristotle University of Thessaloniki, 541 24 Thessaloniki, Greece (e-mail: ialexios@ece.auth.gr; zdatsios@auth.gr; gerodiak@ece.auth.gr; tsovilis@auth.gr).

Color versions of one or more figures in this article are available at https://doi.org/10.1109/TEMC.2023.3234626.

Digital Object Identifier 10.1109/TEMC.2023.3234626

This work is licensed under a Creative Commons Attribution 4.0 License. For more information, see https://creativecommons.org/licenses/by/4.0/



Start
Electrical and geometrical characteristics of the configuration under study
Simulation domain discretization. Cartesian grid
Electric potential computation. FDM and SOR Background electric field
initial boundary conditions imposed
Upward leader propagation
new boundary conditions imposed
Electric potential update and recalculation. FDM & SOR
No
Lightning interception
Yes
End
Upward leader next point selection
uniform random number m (0,1)
Propagation probability calculation (power law function)
Downward leader propagation
Propagation probability calculation (power law function)
uniform random number m (0,1)
Downward leader next point selection
Yes
Lightning attachment (ground or object)
No
Electric potential update and recalculation. FDM & SOR
new boundary conditions imposed
Yes
Upward leader inception (multiple points check)
No

Fig. 1. Flowchart of the stochastic lightning attachment model. Dashed frames: Simulation steps, highlighted boxes: Electric potential computations.

[19], [20]. Applications associated with lightning incidence to a transmission tower, as well as to an HVAC OHL are presented. Results are discussed in terms of the decrease in total simulation time using the original code as a reference. A reduction of up to ∼98% is obtained, enhancing the applicability of stochastic lightning attachment modeling to practical problems.

II. Stochastic Lightning Attachment Model

A general description of the stochastic lightning attachment model employed in this work is presented in Section II-A. Electric potential computations performed initially, as well as at each simulation step are detailed in Section II-B. These dominate the total simulation time, as shown in Section II-C. Thus, they are the focal point of this work dealing with accelerating stochastic lightning attachment simulations.

A. General Description of the Model

The investigated stochastic lightning attachment model has been developed in MATLAB based on the dielectric breakdown model (DBM) [21] for electrical discharge propagation. The model considers the stochastic nature of lightning attachment taking into account the lightning discharge branched and tortuous behavior, as well as multiple competing upward leaders and the interaction among them and the downward leader. Results are both quantitatively and qualitatively similar to fractal structures and their main characteristic of self-similarity [22].

Fig. 1 shows a flowchart of the investigated model. The latter simulates the propagation of the downward-stepped leader and upward leaders (procedures enclosed in dashed frames in Fig. 1, one simulation step each) in a discretized simulation domain with a uniform mesh size. Initially, the background electric field is calculated by employing the finite difference method (FDM) and the successive over-relaxation (SOR) iterative method (see Section II-B), to solve Laplace equation, with appropriate boundary conditions imposed (blue dashed-dotted box, Fig. 1). Then, the downward leader starts propagating. As it approaches the ground the electric field in the vicinity of structures or facilities increases and multiple competing upward leaders may incept from different points according to an adopted inception criterion. The simulation is terminated with lightning striking the ground or with lightning interception between the downward and one of the upward leaders. This enables the determination of useful lightning attachment quantities as well as the point of impact. Fig. 2 shows typical simulation results in a discretized 3D space.

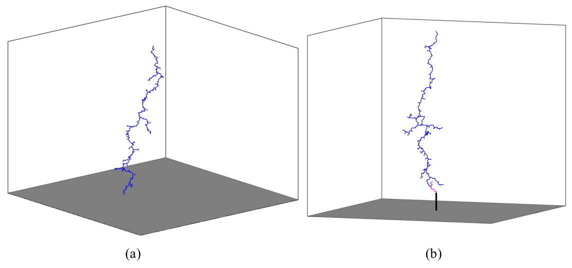


Fig. 2. Typical simulation results of the stochastic lightning attachment model in a 3D space. Lightning strike to the (a) ground and (b) lightning mast.

As shown in Fig. 1, propagation is simulated by iteratively adding domain points to the leaders. These points are selected at each simulation step associated with downward or upward leader propagation (dashed frames, Fig. 1) via propagation probability calculations considering the distribution of the electric field. Each added domain point to a leader obtains a new potential value. This affects the electric field distribution of the whole simulation domain. Hence, after the addition of each new leader point, the electric potential is recalculated at all domain points by employing the FDM and SOR methods (red dotted boxes, Fig. 1), as described in Section II-B. This dynamic recalculation, taking also into account the small mesh size and low converging tolerance required for accurate FDM computations, results in a substantial computational cost stressing the need for adopting code optimization techniques.

Note that different physical criteria [23], [24] (see Appendix A) can be integrated into the stochastic model for the inception and propagation of both downward and upward leaders considering lightning discharge physics, as well as published measurements and observations. However, if appropriately selected, these will not affect the reduction of the simulation time treated in this study.

B. Electric Potential Computation

The electric potential, both background and at each simulation step (highlighted boxes, Fig. 1), is computed by solving the Laplace equation using the FDM

$$\nabla^2 V = 0. \tag{1}$$

By discretizing the simulation domain employing a uniform Cartesian grid (coordinates: i, j), a difference scheme for solving

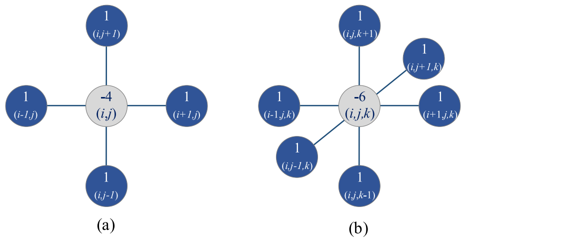


Fig. 3. (a) Five-point stencil (2D) and (b) seven-point stencil (3D) with the coefficient values used in approximating the partial differential equation with finite differences.

Laplace equation can be constructed. For a 2D domain, a second-order scheme can be created by discretizing (1) and replacing x and y derivatives with centered finite differences [25], [26]

$$\frac{1}{(\Delta x)^2}(V_{i-1,j} - 2V_{i,j} + V_{i+1,j}) + \frac{1}{(\Delta y)^2}(V_{i,j-1} - 2V_{i,j} + V_{i,j+1}) = 0, \tag{2}$$

which for $\Delta x = \Delta y$ (grid spacing) is equal to

$$(V_{i-1,j} + V_{i+1,j} + V_{i,j-1} + V_{i,j+1} - 4V_{i,j}) = 0. \tag{3}$$

This finite difference scheme can be represented by the five-point stencil for a 2D space [see Fig. 3(a)]. An equivalent analysis (see Appendix B) may lead to the seven-point stencil for a 3D domain [see Fig. 3(b)]

$$(V_{i-1,j,k} + V_{i+1,j,k} + V_{i,j-1,k} + V_{i,j+1,k} + V_{i,j,k-1} + V_{i,j,k+1} - 6V_{i,j,k}) = 0. \tag{4}$$

Iterative methods can be applied [25], [26] to solve (3) or (4); namely the Jacobi, the Gauss-Seidel, and the SOR methods. A predefined tolerance between successive iterations shall be used as a break criterion to terminate the selected iterative method (change between successive estimates at all points lower than the tolerance ε).

Among the three iterative methods, SOR converges faster. Hence, it is adopted in this work. For a 2D domain

$$V_{i,j}^{[n+1]} = (1-\omega)\cdot V_{i,j}^{[n]} + \omega \cdot V_{i,j}^{*} \tag{5a}$$

where

$$V_{i,j}^{*} = \frac{1}{4}\left(V_{i-1,j}^{[n+1]} + V_{i+1,j}^{[n]} + V_{i,j-1}^{[n+1]} + V_{i,j+1}^{[n]}\right). \tag{5b}$$

$V_{i,j}{}^{[n+1]}$ denotes the $(n+1)$th estimate of the solution at point (i, j) and ω is the over-relaxation parameter ($1 \leq \omega < 2$). The optimal choice for ω significantly affects the convergence; ω can be estimated by theoretical equations [25] but can also be obtained via a trial-and-error procedure. For a 3D simulation domain, (5) can be written using the seven-point stencil of Fig. 3(b) as

$$V_{i,j,k}^{[n+1]} = (1-\omega)\cdot V_{i,j,k}^{[n]} + \frac{\omega}{6}\cdot\left(V_{i+1,j,k}^{[n]} + V_{i-1,j,k}^{[n+1]} + V_{i,j+1,k}^{[n]} + V_{i,j-1,k}^{[n+1]} + V_{i,j,k+1}^{[n]} + V_{i,j,k-1}^{[n+1]}\right). \tag{6}$$

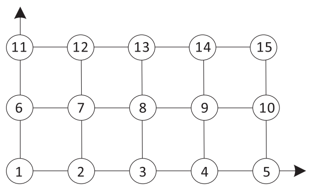


Fig. 4. Natural ordering for a 2D simulation domain; points are accessed from the left to the right and from the bottom to the top.

TABLE I
TYPICAL RESULTS OF SIMULATION TIME ANALYSIS IN MATLAB SOFTWARE

Function	Execution time (s)	Percentage over total simulation time (%)
Main function	4.15	0.04
Leader propagation	245.56	2.09
Computation of the initial background electric field	8,348.40	71.19
Computation of the electric potential at each simulation step	3,129.18	26.68
Total simulation time	11,727.29 (~3.26 h)	100

In the originally developed code for the stochastic lightning attachment model of Fig. 1, the electric potential at every point of the 2D [3D] discretized simulation domain is computed by solving (5) [(6)] iteratively, adopting the so-called “natural ordering” of the domain points (serial implementation). Actually, 2D domain points are accessed from the left to the right and from the bottom to the top, as shown in Fig. 4. For a 3D domain, natural ordering can be achieved by additionally accessing points from the bottom to the top surface.

It is noted that Dirichlet boundary conditions ($V =$ ct.) are chosen as initial boundary conditions for points of fixed electric potential throughout the whole simulation domain. Neumann boundary conditions ($\partial V/\partial x = 0$, $\partial V/\partial y = 0$) are used for the lateral surfaces of the simulation domain.

C. Time-Profile Analysis

The total simulation time of the original stochastic model described above and the execution time of each function of the model were estimated with the aid of the built-in time profiler of MATLAB software. Multiple simulations were performed for different domain discretizations (number of grid points). It was found that the functions implementing the FDM and SOR routines dominate the simulation time for all investigated cases. Their combined execution time percentage over total time was ~98%, as shown in Table I for a typical 3D domain case [total time ~3.26 hours, Fig. 2(a)]. Cases with bigger and more complex domains may comprise thousands of leader discharge points and a simulation can last up to a few days hindering the applicability to practical problems. Thus, this study is focused on accelerating the code of the FDM and SOR function both for the initial electric field calculation (blue dashed-dotted box, Fig. 1) and electric potential recalculation at every simulation step (red dotted boxes, Fig. 1).

```
/* MEX file syntax example */
#include "mex.h"

/* Input function */
void mexFunction(input, output arrays)
 {
  /* variable definition */
  FDM_SOR_3D()
  {
  /* Implementation of FDM & SOR iterative
  methods in C programming language */
  }
 }
```

Fig. 5. Code sample of the MEX function used to compile the C code from the main MATLAB function of the stochastic lightning attachment model.

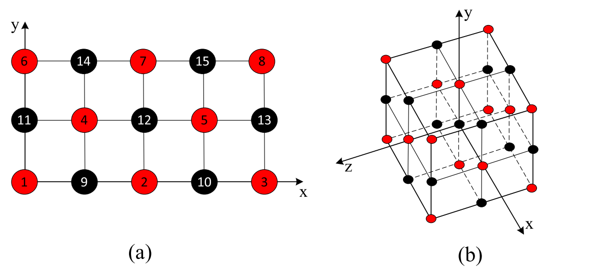


Fig. 6. Red-black ordering for (a) 2D and (b) 3D simulation domains. The update of red nodes depends only on black nodes and vice versa; thus, it can be done independently in parallel.

III. Code Optimization Techniques to Reduce Simulation Time

The code optimization techniques, employed to reduce the simulation time of the stochastic model, are described here. A combination of computational electromagnetics and programming techniques is used to optimize the SOR routine employed to solve the Laplace equation via the FDM; this routine governs the total simulation time (see Section II-C). A C-MATLAB integration methodology is presented (see Section III-A), as well as a multi-color ordering algorithm (see Section III-B), developed to enable parallel execution using CPU and GPU programming.

A. Serial Implementation – C-MATLAB Integration

The original serial implementation written in MATLAB (natural ordering, Section II-B) was optimized through a C-MATLAB integration methodology. In fact, the time-consuming FDM and SOR methods were implemented in the C programming language. Thus, a significant reduction of the total simulation time can be achieved, as C is a lower-level language and the memory access becomes significantly faster and its utilization more efficient. C routines were integrated into the original code via MEX file functions (code sample in Fig. 5). The latter are created in MATLAB for calling a C/C++ program or a Fortran subroutine [27]; MEX files define the required input and output arguments.

B. Parallel Implementation – Multi-Color Ordering Algorithm

Natural ordering comprises a sequential procedure that can be parallelized only if the Jacobi method is used. For the faster SOR method adopted in this work (see Section II-B), parallelization cannot be achieved as every processor would entail calculations dependent on other processors (updated potential values) at every iteration {[n] and [n+1] in (6)}.

In order to enable parallelism in the FDM and SOR methods, the so-called red-black (or checkerboard) ordering [28] is used as shown in Fig. 6 for 2D and 3D simulation domains. It is based on the fact that neighboring points, necessary in the calculations, of a red grid point are black points and vice versa, so their electric potential values can be computed independently in parallel. Thus, two passes over the grid are performed. In the first pass (iteration circle k), the potential values of all red (black)

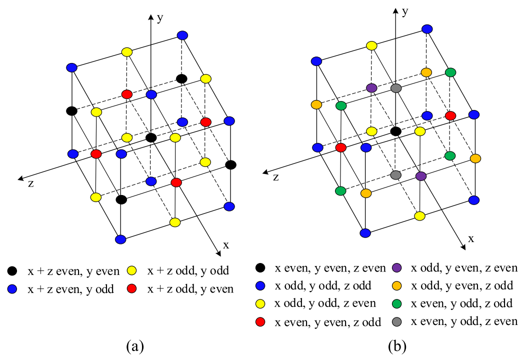


Fig. 7. Multi-color ordering of (a) four and (b) eight color groups for 3D simulation domains; parallelism can be applied for each color group.

grid points can be computed simultaneously in parallel. Then, these updated values are used in the second pass (iteration circle k+1) for the parallel computation of the electric potential of the black (red) grid points; thus, the procedure of red-black (and multi-color) ordering is based on parallel calculations for each color group and serial passes from a color group to the next. It is noted that the way different nodes of the same color are accessed is not important provided that the calculation of all red (black) points (first pass) is completed before the calculation of all black (red) points (second pass) is launched [28].

Additionally, multi-color ordering [29], [30] can be applied both to 2D and 3D domains; nodes represented by a color are able to be updated at the same time, i.e., in parallel. The larger number of colors used, the less parallel the algorithm can become; the classic form of the SOR iterative method with natural ordering (sequential procedure) comprises N colors, that is, equal to the total number of grid points. On the other hand, the larger the number of colors, the faster the convergence becomes, as every next color computation will use the updated values (iteration k+1) of the previous color; thus, a compromise between the two factors should be made to account for both parallelism and efficiency [31]. In this work, four-color ordering for 3D domains [see Fig. 7(a)] is adopted; this is a general compromise, considering the fact that the optimal number of colors is expected to depend on the combination of the application under study and the employed architecture. In addition, selected results obtained using two and eight colors [see Figs. 6(b) and 7(b), respectively] are presented. For the investigated red-black (two colors) and

```
#pragma omp parallel for num_threads(num) collapse(3)
private(index, input variables)
   /* k index */
   for (int k = 1; k < Nz; k = k + 2)
   {
     /* j index */
     for (int j = 1; j < Ny; j = j + 2)
     {
       /* i index */
       for (int i = 1; i < Nx; i = i + 2)
       {
            /* Implementation of FDM & SOR methods */
        }
     }
   }
```

Fig. 8. Code sample for the implementation of OpenMP enabling parallelism of the FDM and SOR routine; four colors.

```
#include "mex.h"
#include "cuda_runtime.h"

/* kernel definition */
__global__ void oddEvenKernel(parameters)
{
    /* indexing based on number of threads, thread
blocks, and grid size */
}

static void FDM_SOR_3D()
{
    /* Implementation of FDM & SOR methods */
}
```

Fig. 9. Code sample for the implementation of CUDA enabling parallelism of the FDM and SOR routine.

multi-color ordering techniques of Figs. 6 and 7, grid points are grouped in colors depending on their relative position; for each color parallelism can be applied.

1) CPU Programming – OpenMP: The red-black and multi-color ordering of Figs. 6 and 7 were implemented in FDM and SOR routines in the C code with the aid of a shared-memory Open Multi-Processing (OpenMP) [32], [33] technique typically used for loop-level parallelism. OpenMP is an application programming interface (API) that supports shared-memory multi-processing programming in C/C++ and Fortran. It is commonly employed in computationally "heavy" parts of a code for parallel execution with multiple cores. This model is used in shared-memory systems; thus, one computing node of the system is required ensuring that all available cores will have access to the same memory.

A code sample for the implementation of OpenMP is given in Fig. 8, illustrating the indexing employed to calculate the electric potentials for a specific grid color solved in parallel. Indexing can be adjusted accordingly based on Fig. 7 to account for the other colors. "#pragma omp parallel for" is used for the creation of *num* number of threads defined in "num_threads(*num*)" and "collapse(3)" denotes that the following three nested "for loops" should be conducted in parallel.

2) GPU Programming – CUDA: Appropriate kernel functions are created and memory is allocated in order to allow for compute unified device architecture (CUDA) programming. Then, the code is compiled for the generation of the necessary CUDA file required for execution. Depending on the available GPU and system characteristics, the number of threads, thread blocks, and grid size are specified so as to define the total number of threads per block and appropriate indexing. Code sample for the CUDA implementation is given in Fig. 9. This part of the code corresponds again to a specific grid color, which is computed in parallel. "__global__ void" is used to define the kernel with appropriate indexing based on the number of thread blocks, the number of threads per block, and grid size; *parameters* denote the input function parameters.

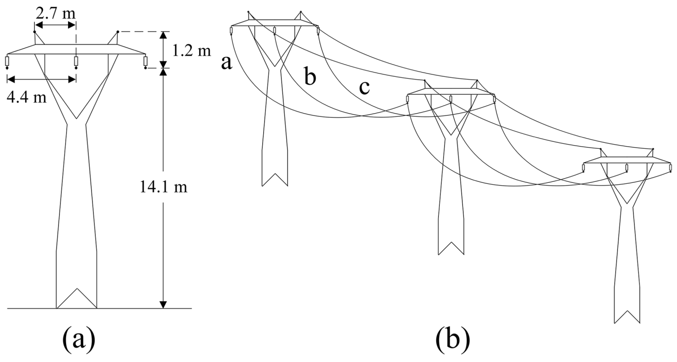


Fig. 10. (a) Typical 66 kV tower (not according to scale) and (b) OHL.

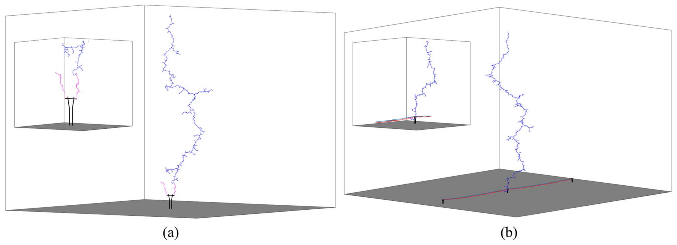


Fig. 11. Typical simulation results: Lightning strike to (a) a transmission tower and (b) an overhead transmission line; zoom-in shown in insets.

IV. Simulation Results

Applications associated with lightning incidence estimation are made for the transmission tower of Fig. 10(a), as well as the corresponding OHL of Fig. 10(b). Typical results of the stochastic lightning attachment model are depicted in Fig. 11. Total simulation time values are presented in this section and discussed in Section V. The performance of the optimized versions of the model (see Section III) is compared against the original model (see Section II). The simulated cases are

i) MATLAB original code of Section II;
ii) C-MATLAB integration methodology of Section III-A;
iii) OpenMP parallel programming of Section III-B-1 with 2 up to 32 CPU threads and four colors [see Fig. 7(a)];
iv) CUDA (GPU) programming of Section III-B-2 [four colors (Fig. 7(a))].

Only 3D simulation domains were considered, as these provide a more accurate representation of the actual geometry and the lightning phenomenon. Also, simulation times for 2D domains are inherently very short, in the order of some seconds or minutes at most, so acceleration is less significant.

The HPC infrastructure of the Aristotle University of Thessaloniki, Greece [34] was used for simulations. It consists of multiple computing nodes with 20 CPU cores per node (CPU model: Intel Xeon E5-2630 v4) and a cumulative RAM of 128GB per node. 2 Nvidia Tesla P100 supercomputing GPUs were employed for case (iv) (maximum number of threads per

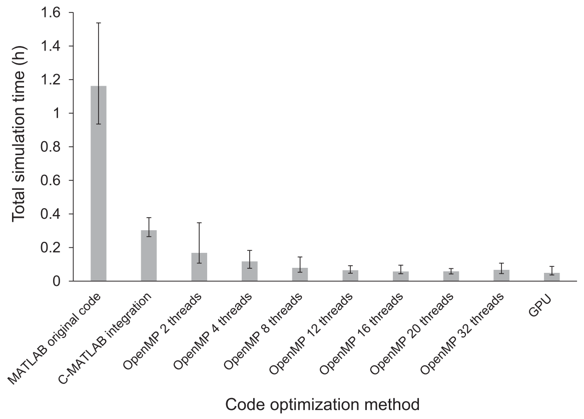


Fig. 12. Total simulation time in hours for the original model (Section II) and the optimization methods of Section III; tower of Fig. 10(a), reference domain.

block = 2048, maximum thread block dimension = 1024, and maximum grid dimension = 2^31-1). Simulations were also conducted using a high-end personal computer [workstation, Intel Core i9-12900K, (max frequency: 5.20 GHz), RAM: 128GB] to demonstrate the applicability of the proposed techniques to personal computers.

The tower of Fig. 10(a) is modeled as points of fixed potential considering actual dimensions together with the uniform mesh size used for discretization in the Cartesian coordinates system. The OHL of Fig. 10(b) is modeled using the catenary equation for a conductor between equal height supports to account for phase conductor and shield wire sag; the span length is 250 m. Dirichlet boundary conditions of $V = 0$ kV were selected for the towers and shield wires; for the phase conductors, an instance of the power frequency voltage was simulated ($V_a = 0$ kV, $V_b = -46.7$ kV, $V_c = +46.7$ kV). For both cases of Fig. 10, the dimensions of the 3D domains were selected appropriately considering the lightning peak current and tower geometry.

Multiple simulations were carried out to account for the stochastic behavior of the fractal-based model, as well as the different possible lightning termination points (earth, tower, phase conductor, or shield wire); cases with multiple upward leaders are generally associated with a higher total simulation time. In addition, simulations were also performed for a fixed random number generator (see Fig. 1). This yields a "deterministic" mode of the model, giving at each run a predefined lightning discharge path so as to facilitate comparisons.

A. 66 kV Transmission Tower

1) HPC Simulation Results: Fig. 12 shows the total simulation time results for the tower of Fig. 10(a); bars denote the minimum and maximum values observed. The mean values are summarized in Table II together with the percentage decrease of the time between the original MATLAB code and each optimization method. This table also includes simulation time results for a deterministic scenario; these fall into the statistical variation shown in Fig. 12.

TABLE II
TOTAL SIMULATION TIME FOR THE TOWER OF FIG. 10(a) – REFERENCE DOMAIN

Code		Mean total simulation time (h)	Percentage decrease (%)	Total simulation time (h)	Percentage decrease (%)
		Statistical results		Deterministic results	
MATLAB original		1.163	–	1.500	–
C-MATLAB integration		0.301	**73.9**	0.319	78.7
OpenMP (4 colors) number of threads	2	0.169	85.5	0.195	87.0
	4	0.118	89.9	0.105	93.0
	8	0.079	93.1	0.074	95.1
	12	0.065	94.4	0.058	96.1
	16	0.058	95.0	0.052	96.5
	20	0.059	95.0	0.053	96.5
	32	0.068	94.2	0.058	96.2
GPU		0.049	**95.8**	0.042	97.2

Grid size = 300x300x300 = 27,000,000 grid points (reference domain)

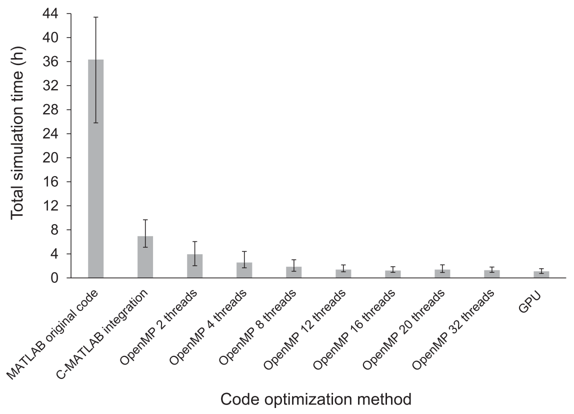


Fig. 13. Total simulation time in hours for the original model (Section II) and the optimization methods of Section III; tower of Fig. 10(a), extended domain.

The same investigation was performed for an extended 3D domain to examine the effect of simulation domain on the percentage decrease of total simulation time. This domain is double the size in each dimension as the initially investigated domain (reference domain); the latter is the smallest (minimum) domain for which the lateral boundaries do not affect the simulation results. It is noted that the extended domain is also appropriate for lightning incidence estimation and lightning performance assessment of the 66 kV single-circuit OHL [see Fig. 10(b)]. Results are presented in Fig. 13 per code optimization method and are also listed in Table III together with the percentage decrease and deterministic results.

From Figs. 12 and 13 and Tables II and III, it is evident that all optimization methods of Section III yield a significant reduction of the total simulation time. Among these methods, the lowest and highest time decrease correspond, respectively, to the C-MATLAB integration and GPU programming; the associated mean percentage decrease is ∼74% and ∼96% for the

TABLE III
TOTAL SIMULATION TIME FOR THE TOWER OF FIG. 10(a) – EXTENDED DOMAIN

Code		Mean total simulation time (h)	Percentage decrease (%)	Total simulation time (h)	Percentage decrease (%)
		Statistical results		Deterministic results	
MATLAB original		36.35	–	39.44	–
C-MATLAB integration		6.96	**80.8**	5.52	86.0
OpenMP (4 colors) number of threads	2	3.96	89.1	3.01	92.4
	4	2.57	92.9	1.92	95.1
	8	1.86	94.9	1.52	96.1
	12	1.40	96.2	1.52	96.1
	16	1.23	96.6	1.62	95.9
	20	1.39	96.2	1.52	96.1
	32	1.29	96.5	1.55	96.1
GPU		1.10	**97.0**	1.27	96.8
Grid size = 600x600x600 = 216,000,000 grid points (extended domain)					

TABLE IV
TOTAL SIMULATION TIME FOR THE TOWER OF FIG. 10(a) – WORKSTATION

Code		Total simulation time (h)	Percentage decrease (%)	Total simulation time (h)	Percentage decrease (%)
		Deterministic results Reference simulation domain		Deterministic results Extended simulation domain	
MATLAB original		0.460	–	9.40	–
C-MATLAB integration		0.139	69.8	2.92	68.9
OpenMP 2 - 32	2 colors	0.050	90	0.95	90
	4 colors	0.060	87	1.42	85
	8 colors	0.090	81	1.62	83

initial (reference) simulation domain. For the extended one, the corresponding values are ~81% and 97%. Also, it is important that the percentage decrease remains almost constant for the OpenMP method with 8 to 32 threads.

By comparing the results for the two domains (see Figs. 12 and 13, Tables II and III), it can be observed that for the extended domain, the mean simulation time is much longer (19–31 times), when considering a specific code optimization method and the original code. However, the notable percentage decrease in the simulation time obtained using the optimization methods is generally comparable for the two domains, though somewhat higher for the extended one, when examining the same method.

2) Workstation Simulation Results: Simulations on the workstation were performed solely for a fixed random number generator, since the obtained deterministic simulation time falls into the simulation time statistical variation, as shown in the previous section. Table IV lists the results for the cases of the reference and extended 3D simulation domains.

It is noted that GPU programming was not employed due to the absence of a suitable GPU. A significant percentage decrease

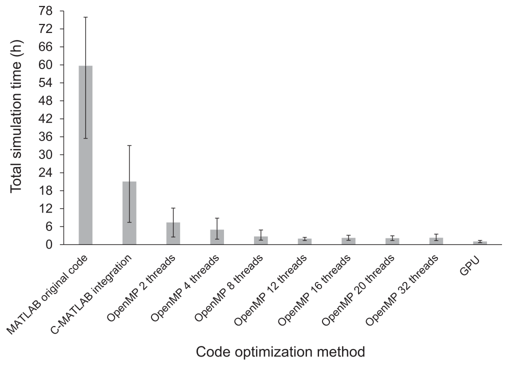


Fig. 14. Total simulation time in hours for the original model (Section II) and the optimization methods of Section III; OHL geometry of Fig. 10(b).

can be observed from Table IV, even though lower than the deterministic HPC simulations cases (Tables II and III). However, the time values obtained with the workstation for the original model of Section II are much lower than those of the HPC; the same also applies for the C-MATLAB integration methodology results. This is due to the faster CPU of this high-end workstation (see Section IV); however, the HPC enables the parallel execution of many simulations due to multiple computing nodes. Moreover, from Table IV it is evident that the shortest simulation times for parallel programming are obtained employing two colors [red-black ordering, Fig. 6(b)] and the longest simulation times employing eight colors [see Fig. 7(b)]. This was also found to apply for stochastic simulations.

B. 66 kV Overhead Transmission Line

1) HPC Simulation Results: Results regarding the total simulation time for the 66 kV OHL are presented in Fig. 14 with bars denoting the minimum and maximum values observed. The mean values together with the percentage decrease of the time between the original MATLAB code and each optimization method are listed in Table V; deterministic simulation results are also included.

It is evident that for this complex case, the simulation time increases considerably as compared to the tower case employing the same simulation domain (see Fig. 13 and Table III). This is due to the fact that in this case multiple upward leader inception may occur increasing the total simulation time. Nevertheless, the percentage decrease of the optimization methods is almost the same with the exception of the C-MATLAB integration methodology yielding a lower value (see Table V). Among optimization methods, the lowest and highest mean time decrease corresponds again to the C-MATLAB integration methodology (~65%) and GPU programming (98%), respectively.

2) Workstation Simulation Results: Deterministic simulation results obtained using the workstation are presented in Table VI. The simulation time values are significantly lower than the deterministic values of the HPC for the original MATLAB code and the C-MATLAB integration methodology (see Table V), in line with the results of Tables II–IV for the single tower geometry. The latter is also true for the percentage decrease of simulation time, which is lower than the relevant HPC value,

TABLE V
TOTAL SIMULATION TIME FOR THE OHL OF FIG. 10(b)

Code		Mean total simulation time (h)	Percentage decrease (%)	Total simulation time (h)	Percentage decrease (%)
		Statistical results		Deterministic results	
MATLAB original		59.69	–	45.64	–
C-MATLAB integration		21.06	**64.7**	18.61	59.2
OpenMP (4 colors) number of threads	2	7.39	87.6	7.19	84.2
	4	4.99	91.6	4.10	91.0
	8	2.73	95.4	3.62	92.1
	12	1.99	96.7	3.03	93.4
	16	2.25	96.2	2.05	95.5
	20	2.13	96.4	2.20	95.2
	32	2.27	96.2	2.26	95.0
GPU		1.06	**98.2**	0.92	98.0
Grid size = 600x600x600 = 216,000,000 grid points					

TABLE VI
TOTAL SIMULATION TIME FOR THE OHL OF FIG. 10(b) – WORKSTATION

Code		Total simulation time (h)	Percentage decrease (%)
		Deterministic results	
MATLAB original		20.13	–
C-MATLAB integration		9.72	51.7
OpenMP 2 - 32	2 colors	2.85	86
	4 colors	4.55	77
	8 colors	5.90	71

as observed by comparing Tables V and VI. In addition, as in Table IV, the shortest simulation times are obtained employing two colors [red-black ordering, Fig. 6(b)].

V. DISCUSSION

The percentage decrease of the total simulation time results presented in Section IV for the optimization methods of Section III is discussed here against the total time of the initial code. The statistical dispersion shown in Figs. 12–14 is attributed to

i) the stochastic character of the fractal-based lightning attachment model yielding different results per simulation;
ii) the different HPC nodes that were assigned for each simulation;
iii) their corresponding computational load at the time of execution.

As for (i), each simulation may involve the inception and competing propagation of multiple upward leaders; more leaders increase the simulation time. Thus, the termination point of lightning strike and the number of upward leaders involved, as well as the degree of branching and tortuosity of the simulated lightning discharges significantly affect the simulation time and contribute to the statistical dispersion observed.

For the OpenMP method, a general trend of decreasing statistical variation with increasing requested threads can be observed as an entire HPC node is usually occupied in the cases of a high number of threads. It is also noted that the deterministic results presented in Section IV are well within the corresponding statistical dispersion range of the stochastic simulations.

For the studied cases, that is, the tower (two simulation domains, Tables II and III) and the OHL (see Table V), the percentage decrease of the total simulation time is generally comparable when considering a specific optimization method (see Section III) and the statistical variation of the results. Nevertheless, a slightly higher decrease is observed for more complex cases. C-MATLAB integration methodology is an exception with a mean percentage decrease ranging from ~65% to 81%. In addition, from Tables II, III, and V, it can be seen that the percentage decrease remains almost constant (~94-96%) for a particular case of the OpenMP method with 8–32 threads. This can be attributed to integrated communication delays between HPC nodes that the simulation is assigned to, the requested number of threads, as well as on how the system handles the assignment of each part of the parallelized code ("for loops," Fig. 8) to the requested threads. For the workstation (see Tables IV and VI), the percentage decrease of the simulation time remains unaffected by the requested number of threads in the OpenMP method. This is due to the way the simulation is allocated to cores and threads by the operating system. Hence, modifications in core and thread handling may further reduce simulation time in specific systems.

The biggest decrease of simulation time was found for the GPU programming case (~96% to 98%, Tables II, III, V) due to the significantly larger number of cores available for parallel implementation (3584 CUDA cores for the case of the available GPU, Section IV). GPU technology can be considered as optimal for this work, as handling and scheduling of GPU cores are optimized for multiple computations. This method actually diminishes the mean total simulation time to 1 h for the investigated OHL (see Table V). Such a reduction is of great importance when considering that a thorough assessment of the lightning performance of OHLs requires multiple simulation runs per lightning peak current level, as well as simulations for different phase angles of the operating voltage. It is important that an acceptable reduction of the total simulation time cannot be achieved by modifying parameters of the original stochastic model, for example, the over-relaxation parameter, ω, in (5) and (6); in this work, the optimal value for the latter was selected (see Appendix A) prior to the application of the investigated optimization techniques.

As regards the effect of the number of colors in parallel programming (see Section III-B), from the results presented in Tables IV and VI it can be seen that the shortest simulation times correspond to two colors [red-black ordering, Fig. 6(b)] and the longest to eight colors [see Fig. 7(b)]. Nevertheless, the optimal number of colors is expected to depend on the combination of the application under study and the available hardware and may be selected using a trial-and-error procedure. As a general compromise to balance parallelism and efficiency, four colors can be used.

The optimization techniques proposed in this study are not tied to the specific stochastic lightning attachment model but can also be used in other models employing the FDM in both 2D and 3D geometries with the aid of HPC computing servers and personal computers. The proposed techniques are applicable to both single- and multi-element models [35], associated, respectively, with the addition of single and multiple discharge points at each simulation step. Additionally, to the best of the authors' knowledge, this is the first time that multi-color ordering techniques, enabling parallel programming, are used in a lightning attachment model. This would enhance applicability to lightning protection studies of complex geometries taking advantage of the rapid developments in computing resources [36] and the increasing use of HPCs in engineering applications.

These techniques can also be applied to fractal-based models simulating electric discharge propagation in solid, liquid, or gaseous dielectrics [37], as well as several engineering applications employing finite differences to solve partial differential equations, such as the heat diffusion, Laplace, or Poisson equations [19], [20].

VI. Conclusion

Optimization techniques of stochastic lightning attachment simulations have been investigated regarding total simulation time. The proposed techniques comprise a C-MATLAB integration methodology, as well as a multi-color ordering algorithm, developed to enable parallel programming with the aid of the API OpenMP (for CPUs) as well as CUDA (for GPUs). Applications on a transmission tower and an HVAC OHL are presented using both HPC and a high-end workstation.

The total simulation time is subject to statistical dispersion resulting from stochastic modeling. The obtained decrease in the mean simulation time depends to some extent on the complexity of the simulated problem, that is, on the investigated structure and the 3D domain dimensions. For parallel OpenMP (CPU) programming, it also depends on the way the operating system handles cores and threads, as after a certain point the overhead imposed by the system to handle and schedule the requested threads plays an important role.

Results on the HPC show that the total simulation time can be reduced with respect to the original MATLAB code from about 65%–81% with the use of the C-MATLAB integration methodology, by ∼95% with the use of OpenMP with multiple threads, and from ∼96% to 98% with CUDA programming. As an example, for the CUDA case, this corresponds to a reduction of the mean simulation time from ∼60 h to 1 h. Thus, the total simulation time and associated computational cost can be remarkably reduced making the applicability of stochastic modeling to the lightning performance assessment of large-scale engineering applications, such as OHLs, feasible within a reasonable total time. The proposed acceleration techniques are not tied to a specific lightning attachment model and/or computing infrastructure. Moreover, these techniques can be utilized in broader engineering problems employing finite differences to solve partial differential equations with computational models.

TABLE VII
Parameters Used in Stochastic Lightning Attachment Simulations of Section IV; Adopted From [23], [24]

Parameter	Value
Downward leader potential (MV) as a function of lightning peak current I (kA)	$3.6858+0.7282\cdot I-0.000818\cdot I^2$
Critical average electric field for discharge propagation (kV/m)	$E_{cr}=E_{cr0}\cdot\exp(-h/u)$ * E_{cr0}=1100 kV/m, u=10 km for the negative downward leader E_{cr0}=464 kV/m, u=7.5 km for positive upward leaders
Upward leader inception criterion	Petrov and Waters model [38]
Average discharge gradient (kV/m)	10
Velocity ratio	Dynamic relation [39]
Propagation parameter	8
Over-relaxation parameter in SOR	1.6

* h (km) is the distance of the leader tip from ground level.

TABLE VIII
Mean Total Simulation Time for Different Values of the Propagation Parameter

Propagation parameter, η	Mean total simulation time (h)
6	7.1
8	6.0
10	5.4
2	> 144
60	1.2

Appendix A

The physical criteria adopted in the stochastic lightning attachment simulations of Section IV are listed in Table VII.

The simulation steps depicted in Fig. 1 are iterated until lightning interception of the downward leader with the upward connecting leader; this interception is based on the attachment of the common streamer zones of the negative downward and positive upward leaders if the corresponding streamer propagation criterion is satisfied.

The propagation parameter value of 8 (see Table VII) used in simulations of Section IV was selected on the basis of lightning discharge fractal dimension calculations [40] considering also field observations. Indicative total simulation time results are presented in Table VIII for several propagation parameter values, including 6, 8, and 10, which yield fractal dimensions in accordance with field observations, as well as two extreme cases: 2 (bushed-type fractal pattern) and 60 (progression in a straight line). It can be seen that simulation time decreases as the propagation parameter increases. Nevertheless, meaningful values (6, 8, 10) yield generally comparable but yet excessive total simulation times. This stresses the need for applying optimization techniques to achieve acceptable simulation times.

As regards the selection of the over-relaxation parameter, ω, of SOR, typical results demonstrating the effect of ω on the total simulation time are presented in Fig. 15 for the 66 kV tower geometry (extended domain) for a fixed lightning path employing the C-MATLAB integration methodology at the workstation; qualitatively similar results were obtained for other

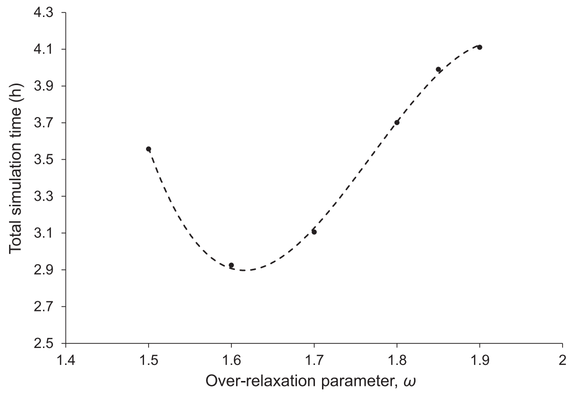


Fig. 15. Variation of the total simulation time with the over-relaxation parameter, ω; C-MATLAB integration methodology, tower geometry of Fig. 10(a), extended domain.

cases. Thus, the optimal value of 1.6 was adopted; this is also in line with literature studies [16].

Appendix B

This section presents the derivation of the second-order scheme for a 3D domain as described by (4) [seven-point stencil, Fig. 3(b)]. The Laplace (1) can be written on its partial derivative form as

$$\frac{\partial^2 V(x,y,z)}{\partial^2 x} + \frac{\partial^2 V(x,y,z)}{\partial^2 y} + \frac{\partial^2 V(x,y,z)}{\partial^2 z} = 0. \tag{B1}$$

By discretizing the 3D simulation domain employing a uniform Cartesian grid (point coordinates: *i*, *j*, *k*), a second-order difference scheme for solving the Laplace equation can be constructed. Employing centered difference approximations, the partial derivatives at the grid point (*i*, *j*, *k*) become

$$\frac{\partial^2 V(x,y,z)}{\partial^2 x} \approx \frac{(V_{i-1,j,k} - 2V_{i,j,k} + V_{i+1,j,k})}{(\Delta x)^2} \tag{B2}$$

$$\frac{\partial^2 V(x,y,z)}{\partial^2 y} \approx \frac{(V_{i,j-1,k} - 2V_{i,j,k} + V_{i,j+1,k})}{(\Delta y)^2} \tag{B3}$$

$$\frac{\partial^2 V(x,y,z)}{\partial^2 z} \approx \frac{(V_{i,j,k-1} - 2V_{i,j,k} + V_{i,j,k+1})}{(\Delta z)^2}. \tag{B4}$$

Substituting (B2)–(B4) to (B1)

$$\frac{(V_{i-1,j,k} - 2V_{i,j,k} + V_{i+1,j,k})}{(\Delta x)^2} + \frac{(V_{i,j-1,k} - 2V_{i,j,k} + V_{i,j+1,k})}{(\Delta y)^2} + \frac{(V_{i,j,k-1} - 2V_{i,j,k} + V_{i,j,k+1})}{(\Delta z)^2} = 0 \tag{B5}$$

which, for the case of a uniform Cartesian grid $\Delta x = \Delta y = \Delta z$, yields the seven-point stencil for a 3D domain, that is, (4)

$$(V_{i-1,j,k} + V_{i+1,j,k} + V_{i,j-1,k} + V_{i,j+1,k} + V_{i,j,k-1} + V_{i,j,k+1} - 6V_{i,j,k}) = 0. \tag{B6}$$

Acknowledgment

Results presented in this work have been produced using the Aristotle University of Thessaloniki High-Performance Computing Infrastructure; the authors would also like to acknowledge the support provided by Ms. A. Charalampidou, throughout the progress of this research work.

References

[1] M. Bruch, V. Münch, M. Aichinger, M. Kuhn, M. Weymann, and G. Schmid, "Power blackout risks: Risk management options," *CRO Forum Emerg. Risk Initiative*, The Netherlands, Nov. 2011, pp. 1–32.
[2] M. R. Alemi and K. Sheshyekani, "Wide-band modeling of tower-footing grounding systems for the evaluation of lightning performance of transmission lines," *IEEE Trans. Electromagn. Compat.*, vol. 57, no. 6, pp. 1627–1636, Dec. 2015.
[3] R. Shariatinasab, J. G. Safar, J. Gholinezhad, and J. He, "Analysis of lightning-related stress in transmission lines considering ionization and frequency-dependent properties of the soil in grounding systems," *IEEE Trans. Electromagn. Compat.*, vol. 62, no. 6, pp. 2849–2857, Dec. 2020.
[4] J. Bao, X. Wang, Y. Zheng, F. Zhang, X. Huang, and P. Sun, "Lightning performance evaluation of transmission line based on data-driven lightning identification, tracking, and analysis," *IEEE Trans. Electromagn. Compat.*, vol. 63, no. 1, pp. 160–171, Feb. 2021.
[5] F. A. M. Rizk, "Modeling of substation shielding against direct lightning strikes," *IEEE Trans. Electromagn. Compat.*, vol. 52, no. 3, pp. 664–675, Aug. 2010.
[6] P. N. Mikropoulos and T. E. Tsovilis, "Estimation of lightning incidence to telecommunication towers," *IEEE Trans. Electromagn. Compat.*, vol. 61, no. 6, pp. 1793–1802, Dec. 2019.
[7] P. N. Mikropoulos, J. He, and M. Bernardi, "Lightning attachment to overhead power lines," in *Lightning Interaction with Power Systems - Vol. 1: Fundamentals and Modelling*, A. Piantini, Ed. London, U.K.: Inst. Eng. Tech., Jan. 2020.
[8] A. Borghetti, F. Napolitano, C. A. Nucci, M. Paolone, and M. Bernardi, "Numerical solution of the leader progression model by means of the finite element method," in *Proc. 30th Int. Conf. Lightning Protection*, Cagliary, Italy, Sep. 2010, Art. no. 1498.
[9] W. Sima, Y. Li, V. A. Rakov, Q. Yang, T. Yuan, and M. Yang, "An analytical method for estimation of lightning performance of transmission lines based on a leader progression model," *IEEE Trans. Electromagn. Compat.*, vol. 56, no. 6, pp. 1530–1539, Dec. 2014.
[10] A. Rahiminejad and B. Vahidi, "LPM-based shielding performance analysis of high-voltage substations against direct lightning strokes," *IEEE Trans. Power Del.*, vol. 32, no. 5, pp. 2218–2227, Oct. 2017.
[11] S. Xie, F. D'Alessandro, and X. Zhao, "A three-dimensional downward leader model incorporating geometric and physical characteristics," *Electr. Power Syst. Res.*, vol. 163, pp. 10–17, Oct. 2018.
[12] A. Rahiminejad, B. Vahidi, and J. He, "A fractal-based stepped downward leader model including branched channel charge distribution and branch fading," *Electr. Power Syst. Res.*, vol. 176, Nov. 2019, Art. no. 105940.
[13] J. Guo, X. Zhang, B. Wang, X. Hao, S. Zheng, and Y.-Z. Xie, "A three-dimensional direct lightning strike model for lightning protection of the substation," *IET Gener. Transm. Distrib.*, vol. 15, no. 19, pp. 2760–2772, Oct. 2021.
[14] Q. Zhang and B. Zhou, "Identification of striking distance considering the velocity ratio and competition among the leaders," *J. Electrostat.*, vol. 72, no. 5, pp. 365–371, Oct. 2014.
[15] C. Zhuang, H. Liu, R. Zeng, and J. He, "Adaptive strategies in the leader propagation model for lightning shielding failure evaluation: Implementation and applications," *IEEE Trans. Magn.*, vol. 52, no. 3, Mar. 2016, Art. no. 9400604.
[16] J. Sañudo, J. B. Gómez, F. Castaño, and A. F. Pacheco, "Fractal dimension of lightning discharge," *Nonlin. Processes Geophys.*, vol. 2, pp. 101–106, Jan. 1995.
[17] F. Aslani, M. Yahyaabadi, and B. Vahidi, "An intelligent-reduced time method to analyze lightning performance of communication towers and validation using experimental tests," *Electr. Power Syst. Res.*, vol. 173, pp. 143–152, Aug. 2019.
[18] F. Aslani, M. Yahyaabadi, and B. Vahidi, "A new-intelligent method for evaluating the lightning protection system performance of complex and asymmetric structures," *Electr. Power Syst. Res.*, vol. 190, Jan. 2021, Art. no. 106843.

[19] S. C. Chapra, *Applied Numerical Methods with MATLAB for Engineers and Scientists*, 4th ed. New York, NY, USA: McGraw-Hill, 2018, pp. 658–673.

[20] T.-R. Hsu, “Applications of partial differential equations in mechanical engineering analysis,” in *Applied Engineering Analysis*. Hoboken, NJ, USA: Wiley, Apr. 2018.

[21] H. J. Wiesmann and H. R. Zeller, “A fractal model of dielectric breakdown and prebreakdown in solid dielectrics,” *J. Appl. Phys.*, vol. 60, no. 5, pp. 1770–1773, Sep. 1986.

[22] A. A. Tsonis and J. B. Elsner, “Fractal characterization and simulation of lightning,” *Contrib. Atmosph. Phys.*, vol. 60, no. 2, pp. 187–192, May 1987.

[23] A. I. Ioannidis and T. E. Tsovilis, “Shielding failure of high voltage substations: A fractal-based approach for negative and positive lightning,” *IEEE Trans. Ind. Appl.*, vol. 57, no. 3, pp. 2317–2325, May/Jun. 2021.

[24] Z. G. Datsios, A. I. Ioannidis, T. A. Papadopoulos, and T. E. Tsovilis, “A stochastic model for evaluating the lightning performance of a −400kV HVDC overhead line,” *IEEE Trans. Electromagn. Compat.*, vol. 63, no. 5, pp. 1433–1443, Oct. 2021.

[25] R. J. LeVeque, *Finite Difference Methods for Ordinary and Partial Differential Equations*. Philadelphia, PA, USA: Soc. Ind. Appl. Math., 2007, pp. 76–77.

[26] J. Li and Y.-T. Chen, *Computational Partial Differential Equations Using MATLAB*, 2nd ed. London, U.K.: Chapman and Hall, 2020, pp. 65–88.

[27] MEX File Functions. Accessed: Jan. 2023. [Online]. Available: https://www.mathworks.com/help/matlab/call-mex-file-functions.html

[28] D. J. Evans, “Parallel S.O.R. iterative methods,” *Parallel Comput.*, vol. 1, pp. 3–18, Mar. 1984.

[29] J. Kouatchou and J. Zhang, “Optimal injection operator and high order schemes for multigrid solution of 3D poisson equation,” *Intern. J. Comp. Math.*, vol. 76, no. 2, pp. 173–190, 2000.

[30] W. Auzinger and C. Fabianek, “Iterative solution of large linear systems arising in the 3-Dimensional modelling of an electric field in human thigh,” Tech. Rep., ANUM Preprint No. 12/04, TU Wien, 2004.

[31] “Parallel methods in numerical analysis,” Lecture Notes, CME342/AA220, Stanford University, Stanford, CA, USA, Apr. 2014.

[32] D. Unat, X. Cai, and S. B. Baden, “Mint: Realizing CUDA performance in 3D stencil methods with annotated C,” in *Proc. 25th Intern. Conf. Supercomput.*, Tucson, AZ, USA, May-Jun. 2011, pp. 214–224.

[33] C. L. Shah, D. Majumdar, and S. Sarkar, “Performance enhancement of an immersed boundary method based FSI solver using OpenMP,” in *Proc. 21st Annu. CFD Symp.*, Bangalore, India, Aug. 2019, pp. 1–6.

[34] AUTH HPC Infrastructure. Sep. 2022. Accessed: Jan. 2023. [Online]. Available: https://it.auth.gr/en/hpc-en/

[35] V. P. Charalambakos, D. P. Agoris, E. C. Pyrgioti, and C. P. Stamatelos, “Computer simulation of lightning strokes,” in *Proc. 28th Int. Conf. Lightning Protection*, Kanazawa, Japan, Sep. 2006, pp. 163–168.

[36] S. K. Moore, “Behind intel’s HPC chip that will pierce the exascale barrier,” *IEEE Spectr.*, Feb. 2022, Accessed: Jan. 2023. [Online]. Available: https://spectrum.ieee.org/intel-s-exascale-supercomputer-chip-is-a-master-class-in-3d-integration

[37] A. I. Ioannidis, N. C. Karanikiotis, P. N. Mikropoulos, P. K. Samaras, and T. E. Tsovilis, “Development of a fractal-based model for simulating streamer flashover of insulating surfaces,” in *Proc. IEEE Conf. Elect. Insul. Dielectric Phenomena*, Vancouver, B.C., Canada, Dec. 2021, pp. 663–666.

[38] N. I. Petrov and R. T. Waters, “Lightning to earthed structures: Striking distance variation with stroke polarity, structure geometry and altitude based on a theoretical approach,” *J. Electrostat.*, vol. 112, Jul. 2021, Art. no. 103599.

[39] A. I. Ioannidis and T. E. Tsovilis, “Stochastic modeling of shielding failures to substations: Implications for the energy absorption requirements of surge arresters,” in *Proc. IAS Annu. Meeting*, Detroit, MI, USA, Oct. 2022, pp. 1–16.

[40] A. I. Ioannidis, Z. G. Datsios, G. A. Tsaousakis, and T. E. Tsovilis, “Analysis of the fractal dimension of lightning discharges based on a stochastic lightning attachment simulation model,” in *Proc. 36th Int. Conf. Lightning Protection*, Cape Town, South Africa, Oct. 2022, pp. 36–41.

Alexios I. Ioannidis (Student Member, IEEE) was born in Ptolemaida, Greece, in 1993. He received the M.Eng. degree in electrical and computer engineering in 2017 from the Aristotle University of Thessaloniki (AUTh), Thessaloniki, Greece, where he is currently working toward the Ph.D. degree with the High Voltage Laboratory.

His research interests include computational electromagnetics for high-voltage engineering applications and lightning protection. Emphasis is given to the development of a stochastic lightning attachment model for assessing the lightning performance of power systems.

Zacharias G. Datsios (Member, IEEE) was born in Kozani, Greece, in 1986. He received the M.Eng. and Ph.D. degrees in electrical and computer engineering from the Aristotle University of Thessaloniki (AUTh), Thessaloniki, Greece, in 2010 and 2017, respectively.

He is currently a Research Associate with the High Voltage Laboratory, AUTh, and an Adjunct Lecturer in high-voltage engineering with the University of Western Macedonia, Kozani, Greece. His research interests include electrical discharges and breakdown, grounding systems, soil electrical properties and ionization, lightning protection, and insulation coordination for power systems.

Apostolos K. Gerodimos was born in Larisa, Greece, in 1994. He received the M.Eng. degree in electrical and computer engineering from the Aristotle University of Thessaloniki (AUTh), Thessaloniki, Greece, in 2021.

He has been a DevOps Engineer for companies that offer consulting services. His research interests include software engineering, and optimizing the design, development, and maintenance of software systems. In particular, he has explored algorithm optimization techniques to improve their efficiency, accuracy, and performance.

Thomas E. Tsovilis (Senior Member, IEEE) was born in Piraeus, Greece, in 1983. He received the M.Eng. and Ph.D. degrees in electrical and computer engineering from the Aristotle University of Thessaloniki (AUTh), Thessaloniki, Greece, in 2005 and 2010, respectively.

He held various managerial positions with the R&D Department, Raycap Corporation, leading innovation in surge protective devices. In 2018, he joined AUTh, where he is currently an Assistant Professor. His research interests include the broad area of high-voltage engineering with emphasis given to electrical discharges, lightning and surge protection, and insulation coordination for power systems. He has authored or coauthored more than 80 papers and is the inventor of eight granted U.S. patents.