Summer 2016

Efficient high-order integral equation methods for the heat equation

Shaobo Wang
New Jersey Institute of Technology

ABSTRACT

EFFICIENT HIGH-ORDER INTEGRAL EQUATION METHODS FOR THE HEAT EQUATION

by
Shaobo Wang

Efficient high-order integral equation methods have been developed for solving the boundary value problems of the heat equation with complex geometries in two and three dimensions. First of all, the classical heat potential theory is applied to convert such problems to Volterra integral equations of the second kind via the heat layer potentials. Some advantages of the integral formulation as compared with standard finite difference and finite element methods include reduction of the dimension of the problem by one, high order accuracy, unconditional stability, insensitivity to different geometries, and elimination of truncating the computational domain and the need of artificial boundary conditions for exterior problems. However, the heat layer potentials contains convolution integrals in both space and time whose direct evaluation requires $O(N_S^2 N_T^2)$ work and $O(N_S N_T)$ storage, where N_S is the total number of discretization points in the spatial boundary and N_T is the total number of time steps. This is excessively expensive even for problems of modest size, especially for three-dimensional problems.

In order to evaluate the heat layer potentials accurately and efficiently, they are split into two parts - the local part which contains the temporal integration from $t-\delta$ to t and the history part which contains the temporal integration from 0 to $t-\delta$. For the local part, Product integration is applied on the temporal integral to convert it to a sum of several spatial convolution integrals where the so-called local kernels have logarithmic singularity in two dimensions and $\frac{1}{r}$ singularity in three dimensions. These weakly singular integrals are discretized via high-order quadratures and the

resulting discrete summations can then be evaluated via fast algorithms such as the fast multipole method and its descendants.

For the history part, efficient separated sum-of-exponentials approximations can be constructed for the heat kernel in any dimension. Specifically, in one space dimension, the heat kernel admits an approximation involving a number of terms that is of the order $O(\log(\frac{T}{\delta})(\log(\frac{1}{\epsilon}) + \log\log(\frac{T}{\delta})))$ for any $x \in \mathbb{R}$ and $\delta \leq t \leq T$, where ϵ is the desired precision. In all higher dimensions, the corresponding heat kernel admits an approximation involving only $O(\log^2(\frac{T}{\delta}))$ terms for fixed accuracy ϵ. These approximations can be used to accelerate the evaluation of the history part of the heat layer potentials for *stationary* geometries.

For two-dimensional problems with complex *stationary* geometries, the sum-of-exponentials approximation is used for the heat kernel and all local and history kernels are compressed only once. The resulting algorithm is very efficient with quasilinear complexity in both space and time for both interior and exterior problems. For two-dimensional problems with complex *moving* geometries, the spectral Fourier approximation is applied for the heat kernel and NUFFT is used to speed up the evaluation of the history part of the heat potentials. The complexity of the algorithm is again quasilinear in both space and time, albeit only for the interior problem. For three-dimensional problems, the sum-of-exponentials approximation is applied to speed up the evaluation of the history part. The singular surface integrals in the local kernels are treated with a spectrally accurate integrator. The algorithm is applicable for both interior and exterior problems and has quasilinear complexity with respect to the temporal variable. All these algorithms can be parallelized in a straightforward manner and their performance is demonstrated with extensive numerical experiments.

EFFICIENT HIGH-ORDER INTEGRAL EQUATION METHODS FOR THE HEAT EQUATION

by
Shaobo Wang

A Dissertation
Submitted to the Faculty of
New Jersey Institute of Technology and
Rutgers, The State University of New Jersey – Newark
in Partial Fulfillment of the Requirements for the Degree of
Doctor of Philosophy in Mathematical Sciences

Department of Mathematical Sciences
Department of Mathematics and Computer Science, Rutgers-Newark

August 2016

APPROVAL PAGE

EFFICIENT HIGH-ORDER INTEGRAL EQUATION METHODS FOR THE HEAT EQUATION

Shaobo Wang

Dr. Shidong Jiang, Dissertation Advisor Date
Associate Professor of Mathematical Sciences, New Jersey Institute of Technology

Dr. Shahriar Afkhami, Committee Member Date
Associate Professor of Mathematical Sciences, New Jersey Institute of Technology

Dr. Michael Siegel, Committee Member Date
Professor of Mathematical Sciences, New Jersey Institute of Technology

Dr. Shravan Veerapaneni, Committee Member Date
Assistant Professor of Mathematics, University of Michigan

Dr. Yuan-nan Young, Committee Member Date
Associate Professor of Mathematical Sciences, New Jersey Institute of Technology

BIOGRAPHICAL SKETCH

Author: Shaobo Wang

Degree: Doctor of Philosophy

Date: August 2016

Undergraduate and Graduate Education:

- Doctor of Philosophy in Mathematical Sciences,
  New Jersey Institute of Technology, Newark, NJ, 2016
- Bachelor of Science in Mathematics and Applied Mathematics
  Nanjing University, Nanjing, China, 2010

Major: Mathematical Sciences

Presentations and Publications:

S. Jiang, L. Greengard, and S. Wang, "Efficient sum-of-exponentials approximations for the heat kernel and their applications," *Adv. in Comput. Math.*, 41:529-551, 2015.

Dedicated to my parents, wife and newborn daughter.

ACKNOWLEDGMENT

First and foremost, I would like to express my deep thanks to my advisor, Professor Shidong Jiang for his immense support and encouragement over the past years. His advice on both research and my career has been invaluable. My dissertation would not be possible without his guidance.

I would also like to thank Professor Shahriar Afkhami, Professor Michael Siegel, Professor Shravan Veerapaneni, and Professor Yuan-nan Young for serving on my thesis committee.

Special thanks to Dr. Zydrunas Gimbutas at National Institute of Standard and Technology for many helpful suggestions and for generously sharing the code for computing singular integrals on smooth surfaces, and Professor Michael O'Neil at Courant Institute of New York University for providing me a computer account so that I can run large-scale simulations. Also thanks to the National Science Foundation for its partial support to my PhD study through my advisor's grant.

Finally, I would like to express my gratitude to my parents for their tremendous support and endless love throughout my life. I am also very grateful to my beloved wife and newborn daughter for all the joy they bring.

TABLE OF CONTENTS

TABLE OF CONTENTS
(Continued)

Chapter **Page**

LIST OF TABLES

Table **Page**

LIST OF FIGURES

Figure **Page**

LIST OF FIGURES (Continued)

LIST OF FIGURES
(Continued)

Figure **Page**

LIST OF FIGURES
(Continued)

Figure **Page**

CHAPTER 1

INTRODUCTION

In this dissertation, we consider the boundary value problems of heat equation in two and three dimensions:

$$\begin{cases} U_t(x,t) = \Delta U(x,t), \quad (x,t) \in \prod_{\tau=0}^{T} \Omega(\tau), \\ U(x,0) = 0, \quad x \in \Omega_0, \\ U(x,t) = f(x,t), \quad (x,t) \in \prod_{\tau=0}^{T} \Gamma(\tau), \end{cases} \tag{1.1}$$

where $x \in \mathbb{R}^d$ $(d = 2,3)$, $\Omega(\tau) \in \mathbb{R}^d$ is a domain at $t = \tau$ which could be multiply connected and/or unbounded, and $\Gamma(\tau)$ is its boundary. Although we will focus on the Dirichlet boundary conditions for the most part of the dissertation, the Neumann and Robin conditions can be dealt with in an almost identical manner. Moreover, it is straightforward to treat the nonzero initial data and the inhomogeneous term in the equation in our framework. We refer the reader to Chapter 3 and references therein for a detailed discussion about this.

In order to solve (1.1) numerically, we first apply standard heat potential theory to convert it to a Volterra integral equation of the second kind using the heat double layer potentials. As compared with the commonly used finite difference and finite element methods, integral formulation have a number of advantages for solving this type of problems. First, the unknown density is only on the space-time boundary $\prod_{\tau=0}^{T} \Gamma(\tau)$. This reduces the dimension of the problem by one and thus the total number of unknowns by a large extent. Second, it is easier to design high-order discretization scheme for the boundary and the unknowns on the boundary rather than the whole volume and the unknowns in the whole volume, especially in the case of complex geometries. In two dimensions, one only needs to discretize the boundary curves instead of the 2D domain; while in three dimensions, one only needs to discretize the boundary surfaces instead of the 3D domain. Third, it is easier to design high-order marching scheme in time using integral formulation in the case of moving geometry. Fourth, the integral formulation leads to a well-conditioned linear system which requires a constant number of iterations to solve and the associated marching scheme is unconditionally stable, while finite difference/finite element methods either requires certain restriction on the Courant number for explicit schemes or needs good preconditioners for solving the linear system for implicit schemes. Finally, for exterior problems, there is no need to design artificial boundary conditions to truncate the computational domain when the integral formulation is used.

However, the heat layer potentials contain the convolution integrals in both space and time. A straightforward way to evaluate heat potentials at a sequence of time steps $t_n = n\Delta t$, for $n = 0, \ldots, N_T$, clearly requires an amount of work of the order $O(N_T^2 N_S^2)$, where N_S denotes the number of points in the discretization of the spatial boundary. While direct discretization methods have been used in the absence of fast algorithms, it is difficult to argue that integral equation methods would be methods of choice for large-scale simulation. In the last two decades, a variety of schemes have been developed to overcome this obstacle. The scheme of [31] used discrete Fourier methods to represent the history part, while [10] replaced the Fourier representation with a regular (physical space) grid on which to update the history part of the heat layer potential. In [28], the problem of exterior heat flow was considered using the continuous Fourier transform in the spatial variables. Sethian and Strain [58] and Ibanez and Power [37] developed variants of the fast algorithm of [31] in the analyses of solidification, melting and crystal growth. More recently, Tausch [62] developed an interesting space-time “fast-multipole-like” method that also overcomes the cost of history-dependence (although it involves a hierarchical decomposition of the entire space-time domain and thus has a very large prefactor in its computational complexity).

A somewhat different approach to overcoming history-dependence is based on using the Laplace transform in the time variable, leading to what are sometimes called ”Laplace transform boundary element methods”. That is the approach we consider here, for the following reasons:

1. The Fourier methods of [28, 31] assume that the spatial domain of interest is finite (even when considering exterior problems). Both the computation of the Fourier modes and the evaluation of the solution at large distances can involve highly oscillatory integrals.
2. The required number of Fourier modes in [28, 31] is $O\left(\left(\frac{a}{\sqrt{\delta}}\right)^d\right)$, where a is a bound on the extent of the domain in each spatial direction. This makes the method inefficient for small δ.

Using the Laplace transform avoids both of these difficulties and leads to a sum-of-exponentials approximation of the heat kernel that is asymptotically optimal (although in the end, hybrid schemes may yield better constants).

Sum-of-exponentials approximations of convolution kernels have many applications in scientific computing. They permit, for example, the construction of diagonal forms for translation operators. We refer the reader to [17, 67] for their use in the elliptic case in accelerating fast multipole methods. They also permit the development of highly efficient nonreflecting boundary conditions for the wave and the Schrödinger equations [2, 3, 34, 38, 39, 48].

Function approximation using sums of exponentials is a highly nonlinear problem, so that the numerical construction of such approximations is nontrivial. In a series of papers, Beylkin and Monzón [5, 6, 7] carried out a detailed investigation and developed efficient and robust algorithms when given function values on a fixed interval. In some cases, the function of interest can be represented as a parameterized integral with exponential functions in the integrand, in which case generalized Gaussian quadrature methods [13, 49] can also be used. In other cases, however, the function being approximated, say $f(t)$ is only accessible as the inverse Laplace transform of an explicitly computable function, say $\hat{f}(s)$. If $\hat{f}(s)$ is sectorial (i.e., holomorphic on the complement of some acute sector containing the negative real axis for $s \in \mathbb{C}$), then the truncated trapezoidal or midpoint rule can be used in conjunction with carefully chosen contour integrals, leading to efficient and accurate sum-of-exponentials approximations. López-Fernández, Palencia, and Schädle, for example, have made effective use of various hyperbolic contours [46, 47]. On the other hand, if the Laplace transform $\hat{f}(s)$ does not have such well-defined properties, one may instead try to find a sum-of-poles approximation in the s-domain, from which a sum-of-exponentials approximation for $f(t)$ can be obtained by inverting the sum-of-poles approximation analytically (see, for example, [66]).

In this dissertation, we construct efficient *separated* sum-of-exponentials (SOE) approximations for the free-space heat kernel. In particular, we show that the one-dimensional heat kernel $\frac{1}{\sqrt{4\pi t}}e^{-|x|^2/(4t)}$ admits an approximation of the form

$$\frac{1}{\sqrt{4\pi t}}e^{-|x|^2/(4t)} \approx \sum_{i=1}^{N_1} w_i e^{-s_i t} e^{\sqrt{s_i}|x|}$$

for any $t \in [\delta, T]$ and $x \in \mathbb{R}$, where $N_1 = O(\log(\frac{T}{\delta})(\log(\frac{1}{\epsilon}) + \log\log(\frac{T}{\delta})))$ where ϵ is the desired precision. In the d-dimensional case ($d > 1$), the heat kernel $\frac{1}{(4\pi t)^{d/2}}e^{-|x|^2/(4t)}$ admits an approximation of the form

$$\frac{1}{(4\pi t)^{d/2}}e^{-|x|^2/(4t)} \approx \sum_{j=1}^{N_2} \tilde{w}_j e^{-\lambda_j t} \sum_{i=1}^{N_1} w_i e^{-s_i t} e^{\sqrt{s_i}|x|} \tag{1.2}$$

for all $t \in [\delta, T]$ and $x \in \mathbb{R}^d$, where $N_1 = O(\log(\frac{T}{\delta})(\log(\frac{1}{\epsilon}) + \log\log(\frac{T}{\delta})))$ and $N_2 = O\left(\log\left(\frac{1}{\epsilon}\right) \cdot \left(\log\frac{T}{\delta} + \log\log\frac{1}{\epsilon}\right)\right)$. Both our construction and proof draw on earlier work, especially [7, 47].

The basic idea is to use the sum-of-exponentials approximation for the computation of the history part, since the parameter δ separates the time integration variable τ from the current time t so that $t - \tau \in [\delta, T]$. Since the temporal dependence for each term in equation (1.2) involves a simple exponential, the convolution in time can be easily computed using standard recurrence relations, as in [28, 31]. Furthermore, the convolutions in space can be evaluated by a variety of fast algorithms, such as variants of the fast multipole method. These issues are discussed in Chapter 5. In the local part, the variants of fast multipole methods are again employed to evaluate the convolution in space.

Although the Fourier method suffers from the aforementioned shortcomings, it has the advantage that the Fourier approximation of the heat kernel is completely separable in the temporal variable and spatial variable, and also in the source points and target points. This makes it very suitable for dealing with moving geometries. Hence, as a complementary algorithm of the SOE approximation method, we have also implemented the spectral Fourier approximation for the heat kernel and applied it to solve the boundary value problems of the heat equation for complex moving geometries in two dimensions.

To summarize, we have developed the following three algorithms for solving the boundary value problems of the heat equation.

1. A quasilinear complexity algorithm in both space and time for solving the boundary value problems of the heat equation with *stationary* complex geometries in two dimensions using sum-of-exponentials approximations.

2. A quasilinear complexity algorithm in both space and time for solving the boundary value problems of the heat equation with *moving* complex geometries in two dimensions using spectral Fourier approximation of the heat kernel and nonuniform FFT.

3. A high-order algorithm with quasilinear complexity in time for solving the boundary value problems of the heat equation with *stationary* complex geometries in three dimensions using sum-of-exponentials approximations.

The rest of the dissertation is organized as follows. In Chapter 2, we review several numerical and analytical apparatus that are needed for our algorithms. In Chapter 3, we collect some well-known results on the potential theory for the heat equation. In Chapter 4, we present theoretical results about the sum-of-exponentials approximation for the heat kernels in all dimensions. In Chapter 5, we discuss in detail various algorithms that we have developed for solving the boundary value problems of the heat equation in both two and three dimensions. The performance of these algorithms is demonstrated via extensive numerical experiments in Chapter 6. Finally, we conclude the dissertation with further discussion and future research directions.

CHAPTER 2

PRELIMINARIES

In this chapter, we collect several numerical tools to be used subsequently.

2.1 Recursive Skeletonization Factorization

Consider the boundary integral equation

$$-\frac{1}{2}\sigma(x) + K(\sigma)(x) = f(x), \tag{2.1}$$

which is commonly seen in physics and engineering when applying classical potential theory. The equation can be discretized into a linear system

$$Ax = b\,, \tag{2.2}$$

where $A \in C^{NXN}$, $x \in C^{NX1}$ and $b \in C^{NX1}$. Direct matrix-vector product of the above system clearly requires $O(N^2)$, which become impractical when N is large. There have been many fast algorithms in recent thirty years that speed up either the matrix-vector product or solving the above linear system. Here we will only discuss the fast multipole method (FMM) and its descendants. The original FMM was first developed by Greengard and Rokhlin [29] to evaluate the sum of the form

$$f_i = \sum_{j=1, j\neq i}^{N} \ln|x_i - x_j|\, q_j, \quad x_i \in \mathbb{R}^2, \quad i = 1, \cdots, N\,, \tag{2.3}$$

with $O(N)$ computational cost. It was then extended in [17, 30] to speed up the Coulomb interactions in three dimensions. Since then there have been many generalizations of the FMM which is more or less "kernel-independent" in the sense that it is capable of speeding up the evaluation of the matrix-vector product as long as the matrix satisfies certain hierarchical low rank structure. In these kernel-independent FMMs, the analytical multipole and local expansions are replaced by the singular value decomposition [23], the equivalent proxy charges [68], the interpolative decomposition [52], etc. Recently, there have been rapid developments in the so-called fast direct solvers (FDS) which construct an efficient representation of the matrix inverse and solve the linear system efficiently (see, for example, [12, 22, 26, 35, 42, 42, 50, 51, 53]). The most recent paper by Ho and Ying [36] presents two fast algorithms - recursive skeletonization factorization (`rskelf`) and hierarchical interpolative factorization (`hifie`). Both algorithms factorize the matrix and its inverse into a product of sparse low-rank matrices and have optimal complexity for intrinsically 1D problems. The difference between `rskelf` and `hifie` is that `hifie` will introduce extra compression steps to achieve better complexity for higher dimensional problems.

For the boundary value problems of the heat equation in two dimensions, we will need to solve a linear system whose unknowns are on the boundary curves if we use a high-order implicit scheme to discretize the boundary integral equation. In this case, the intrinsic geometry is 1D for most practical problems and `rskelf` seems to be the method of choice for matrix (and its inverse) compression. It is especially advantageous for *stationary* geometries since one only needs to compress the matrix once and the *apply* time is often at least ten times faster than the FMM.

We now present a short discussion about `rskelf` in [36]. First, we introduce interpolative decomposition (ID) [16], which is different from the known singular value decomposition and QR decomposition in a sense that we can use it to develop a multi-level compression for a matrix in a hierarchical way.

Lemma 2.1. *For any $m \times n$ matrix A with rank k, there exists an approximation $A \approx SP$, where $S \in C^{m\times k}$ is called the skeleton matrix, whose columns are a subset of the columns of A, and $P \in C^{k\times n}$ is called the projection matrix, a subset of whose* *columns are the $k \times k$ identity matrix. Furthermore, $||P||$ is small and $||A - SP|| \sim \sigma_{k+1}$, where σ_{k+1} is the $(k+1)st$ greatest singular value of A.*

Lemma 2.1 shows ID compresses the column space of A. Similarly the row space of A can be compressed by applying ID to A^T. The compression requires $O(kmn)$ in [16], while a more recent technology based on random sampling [45, 65] needs a work of $O(mn \log k + k^2 n)$.

Lemma 2.2. *Assume that p, q, r, k are ordered sets of indices. Suppose*

$$A = \begin{bmatrix} A_{rr} & A_{rk} & A_{rq} \\ A_{kr} & A_{kk} & A_{kq} \\ A_{qr} & A_{qk} & A_{qq} \end{bmatrix},$$

with the submatrices

$$\begin{bmatrix} A_{qr} & A_{qk} \end{bmatrix}$$

and

$$\begin{bmatrix} A_{rq} \\ A_{kq} \end{bmatrix}$$

low rank. Suppose there exists a matrix $P_t \in C^{|k| \times |r|}$ such that $A_{qr} = A_{qk} P_t$. Let

$$Q_t = \begin{bmatrix} I & & \\ -P_t & I & \\ & & I \end{bmatrix}.$$

Then

$$Q_t^* A Q_t = \begin{bmatrix} B_{rr} & B_{rk} & \\ B_{kr} & A_{kk} & A_{kq} \\ & A_{qk} & A_{qq} \end{bmatrix}, \tag{2.4}$$

where

$$B_{rr} = A_{rr} - P_t^* A_{kr} - A_{rk} P_t + P_t^* A_{kk} P_t \,,$$

$$B_{rk} = A_{rk} - P_t^* A_{kk} \,,$$

$$B_{kr} = A_{kr} - A_{kk} P_t .$$

In addition, there exists unit triangular matrices L_r and U_r such that $B_{rr} = L_r D_r U_r$. Then

$$\mathcal{L}(A) = R_r^* Q_t^* A Q_t S_r = \begin{bmatrix} D_r & & \\ & B_{kk} & A_{kq} \\ & A_{qk} & A_{qq} \end{bmatrix}, \qquad (2.5)$$

where

$$R_r^* = \begin{bmatrix} I & & \\ -B_{kr} U_r^{-1} D_r^{-1} & I & \\ & & I \end{bmatrix} \begin{bmatrix} L_r^{-1} & & \\ & I & \\ & & I \end{bmatrix},$$

$$S_r = \begin{bmatrix} U_r^{-1} & & \\ & I & \\ & & I \end{bmatrix} \begin{bmatrix} I & -D_r^{-1} L_r^{-1} B_{rk} & \\ & I & \\ & & I \end{bmatrix},$$

and

$$B_{kk} = A_{kk} - B_{kr} B_{rr}^{-1} B_{rk} .$$

Lemma 2.2 uses ID to sparsify A with only the skeletons k left, leading to further compression by efficient sparse techniques. Moreover, for a collection C of disjoint index sets, the compression leads to

$$\mathcal{L}(A) \approx U^* A V, \qquad U = \prod_{t \in C} Q_t R_r, \quad V = \prod_{t \in C} Q_t S_r .$$

With Lemma 2.2, the main algorithm can be constructed. Suppose that the integral equation (2.1) defined in region $\Omega = (0,1)^2$ is discretized into an $N \times N$ matrix. Consider an L-level compression scheme where at each level l, the region is partitioned into Voronoi cells [4] $m(j_1 - 1, j_1) \times m(j_2 - 1, j_2)$ with width $m = \frac{N}{2^{L-l}}$ about the centers $m(j_1 - \frac{1}{2}, j_2 - \frac{1}{2})$ for $1 \leq j_1, j_2 \leq m$. Then the full compression is as in Algorithm 2.1.

Algorithm 2.1 Recursive Skeletonization Factorization

```
for $l = 0$ to $h - 1$ do
    Compress the $A_l$ such that $A_{l+1} = \mathcal{L}(A) \approx U_l^* A_l V_l$
end for
```

After the multi-level compression, it results in

$$D \approx U_{L-1}^* \cdots U_0^* A V_0 \cdots V_{L-1},$$

with unit triangular matrices U_l and V_l. Then A can be represented by

$$A \approx U_0^{-*} \cdots U_{L-1}^{-*} D V_{L-1}^{-1} \cdots V_0^{-1}. \tag{2.6}$$

Note the inverse matrix is obtained by simply negating the off-diagonal entries from each unit triangular matrix. Therefore, we can apply the compression (2.6) for efficient matrix-vector product.

Remark 2.1. *The cost of the above compression is dominated by applying ID to submatrices, which need to access all rows or columns during the skeletonization. The process can be accelerated by introducing a proxy surface for each ID and making local compression only as in [16, 19, 22, 26, 35, 51, 52, 57, 68].*

Remark 2.2. *For intrinsically 2D problems, the compression cost of* `rskelf` *becomes* $O(N^{\frac{3}{2}})$. *In this case, we could switch to* `hifie` *in [36] to obtain better complexity.*

2.2 Spectral Fourier Approximation for the Heat Kernel in Free Space

Spectral Fourier approximation for the heat kernels has been studied in detail in [28]. For the 1D heat kernel $G_1(x,t)$, the starting point is the following well-known Fourier representation:

$$G_1(x,t) = \frac{e^{-x^2/4t}}{\sqrt{4\pi t}} = \frac{1}{2\pi}\int_{-\infty}^{\infty} e^{-k^2 t} e^{ikx} dk. \tag{2.7}$$

The explicit construction of the spectral Fourier approximation for G_1 in [28] first truncates the above infinite integral to a finite interval $[-p,p]$, then divides this finite interval dyadically with smaller and smaller subintervals towards the origin, and approximates the integral on each subinterval by the Gaussian-Legendre quadrature with a constant number of nodes. In summary, in order to approximate G_1 by a discrete Fourier series for all $t \geq \delta > 0$ and $|x| \leq R$ within an absolute error ϵ, i.e.,

$$\left| G_1(x,t) - \sum_{i=1}^{N_F} w_i e^{-k_i^2 t} e^{ik_i x} \right| \leq \epsilon, \quad |x| \leq R, \quad t \geq \delta, \tag{2.8}$$

the number of Fourier nodes needed N_F has to be of the following order:

$$N_F = O\left(\log\left(\frac{1}{\epsilon}\right) \left(\log\left(\frac{1}{\epsilon\sqrt{\delta}}\right)\right)^{1/2} \frac{R}{\sqrt{\delta}} \right). \tag{2.9}$$

The spectral Fourier approximation for the heat kernel G_d in higher dimensions is obtained via the tensor product in [28].

2.3 Nonuniform Fast Fourier Transform

The ordinary FFT computes the discrete Fourier transform (DFT) and its inverse:

$$\begin{aligned} F(k) &= \sum_{j=0}^{N-1} f(j) e^{-2\pi i k j/N}, \qquad k = 0, \cdots, N-1, \\ f(j) &= \frac{1}{N}\sum_{k=0}^{N-1} F(k) e^{2\pi i k j/N}, \quad j = 0, \cdots, N-1 \end{aligned} \tag{2.10}$$

in $O(N \log N)$ operations by exploiting the algebraic structure of the DFT matrix. The points $x_j = 2\pi j/N$ and the frequencies k, however must be equispaced in both the physical and Fourier domains (see, for example, [18]).

The purpose of the NUFFT is to remove this restriction, while maintaining a computational complexity of $O(N \log N)$, where N denotes the total number of points in both the physical and Fourier domains. Dutt and Rokhlin were the first to construct an algorithm of this type, with full control of precision [21], although heuristic versions had been used earlier. There are, by now, many variants of the NUFFT. All of these algorithms rely on interpolation coupled with a judicious use of the FFT on an oversampled grid. Here, we will use the convention in [27], which describes a simple framework for the NUFFTs using Gaussian kernels for interpolation.

The type-1 NUFFT evaluates sums of the form

$$f(\mathbf{x}) = \frac{1}{N} \sum_{n=0}^{N-1} F_n e^{\pm i \mathbf{k}_n \cdot \mathbf{x}}, \tag{2.11}$$

for "targets" $\mathbf{x}$ on a regular grid in $\mathbb{R}^d$, given function values F_n prescribed at arbitrary locations $\mathbf{k}_n$ in the dual space. Here, N denotes the total number of source points.

The type-2 NUFFT evaluates sums of the form

$$F(\mathbf{k}_n) = \sum_{j_1=-M_1/2}^{M_1/2-1} \cdots \sum_{j_d=-M_d/2}^{M_d/2-1} f(\mathbf{x}_j) e^{\pm i \mathbf{k}_n \cdot \mathbf{x}_j}, \tag{2.12}$$

where the "targets" $\mathbf{k}_n$ are irregularly located points in $\mathbb{R}^d$, given the function values $f(\mathbf{x}_j)$ on a regular grid in the dual space.

Finally, the type-3 NUFFT evaluates sums of the form

$$f(\mathbf{x}_j) = \frac{1}{N} \sum_{n=0}^{N-1} F_n e^{\pm i \mathbf{k}_n \cdot \mathbf{x}_j}, \tag{2.13}$$

where both the "targets" $\mathbf{x}_j$ and the sources $\mathbf{k}_n$ are irregularly located in their corresponding domain.

All these NUFFTs based on the algorithms in [21, 27, 43] consist of three steps: interpolation (or convolution) via the Gaussians to spread each irregular grid to the nearest p grid points, FFT on an oversampled grid points, pointwise multiplication which performs deconvolution. Hence, the computational cost of NUFFTs is $O(p^d N_I + M \log M)$, where d is the dimension of the underlying space, N_I is the number of irregular grid points, and M is the number of regular grid points. In practice, with the oversampling factor equal to 2, $p = 24$ yields about 12 digits of accuracy, and $p = 12$ yields about 6 digits of accuracy.

CHAPTER 3

POTENTIAL THEORY FOR THE HEAT EQUATION

In this chapter, we collect some standard theoretical results on the potential theory for the heat equation, which can be found in [11, 55, 54, 56, 15, 20, 32]. We will denote Ω as a bounded domain in R^d, and its boundary ∂D as Γ. Assume that functions on Γ are at lease twice differentiable. We will use $\bar{\Omega}$ to denote $\Omega \cup \Gamma$. Denote Ω_T as the space-time domain $\Omega \times [0, T]$ and Γ_T as the space-time boundary $\Gamma \times [0, T]$. We will use n_y to represent the unit outward normal to Γ at the point y.

3.1 Fundamental Solution of the Heat Equation

Consider the initial value problem for the heat equation

$$\begin{aligned} U_t(x,t) - \Delta U(x,t) &= 0, \qquad x \in R^d, \quad t > 0 \\ U(x,0) &= U_0(x), \qquad x \in R^d. \end{aligned} \tag{3.1}$$

The solution to the problem (3.1) can be represented as

$$U(x,t) = \int_{R^d} G(x-y,t) U_0(y) dy \,, \tag{3.2}$$

in terms of the fundamental solution $G(x,t)$, where

$$G(x,t) = \frac{1}{(4\pi t)^{d/2}} e^{-\frac{|x|^2}{4t}} \,.$$

Lemma 3.1. *Suppose that $x, y \in R^d$ and $t > \tau > t_0$. Then the fundamental solution satisfies*

$$\begin{cases} \partial_t G(x-y, t-\tau) - \Delta G(x-y, t-\tau) = 0, \\ \int_{R^d} G(x-y, t-\tau) dy = 1, \\ \lim\limits_{t \to 0^+} G(x,t) = \delta(x), \end{cases} \tag{3.3}$$

where $\delta(x)$ is the Dirac delta function.

We now consider the inhomogeneous initial value problem

$$\begin{aligned} U_t(x,t) - \Delta U(x,t) = h(x,t), \qquad x \in R^d, \quad t > 0 \\ U(x,0) = U_0(x), \qquad x \in R^d\,. \end{aligned} \tag{3.4}$$

The solution can be expressed as

$$U(x,t) = \int_{R^d} G(x-y,t)U_0(y)dy + \int_0^t \int_{R^d} G(x-y,t-\tau)h(y,\tau)dyd\tau\,. \tag{3.5}$$

Theorem 3.1. *Suppose in the system* (3.4)*, $U_0(x)$ is continuous in R^d and bounded such that $|U_0(x)| \leq M$ for all x. Assume that $h(x,t)$ is continuous and $|h(x,t)| \leq N$ for $0 \leq t \leq T$ and $x \in R^d$. In addition, $h_x(x,t)$ is also bounded. Then equation* (3.5) *is the unique continuous solution to the system* (3.4)*. Furthermore, U also satisfies the following inequality:*

$$|U(x,t)| \leq M + NT, \qquad x \in R^d \quad \text{and} \quad 0 \leq t \leq T.$$

3.2 Heat Potentials and their Properties

The solution to the initial-boundary value problem of the heat equation can be represented by an initial potential, a volume potential, a single layer potential and a double layer potential defined as follows.

Definition 3.1. *Suppose U_0 is a function on Ω. Then the initial potential is defined by the following formula*

$$I(U_0)(x,t) = \int_\Omega G(x-y,t)U_0(y)dy. \tag{3.6}$$

Definition 3.2. *Suppose h is a function on Ω_T. Then the volume potential is defined by the following formula*

$$V(h)(x,t) = \int_0^t \int_\Omega G(x-y,t-\tau)h(y,\tau)dyd\tau. \tag{3.7}$$

Definition 3.3. *Suppose σ is a function on Γ_T. Then the single layer potential is defined by the following formula*

$$S(\sigma)(x,t) = \int_0^t \int_\Gamma G(x-y, t-\tau)\sigma(y,\tau) ds_y d\tau. \tag{3.8}$$

The double layer potential is defined by the following formula

$$D(\sigma)(x,t) = \int_0^t \int_\Gamma \frac{\partial}{\partial n_y} G(x-y, t-\tau)\sigma(y,\tau) ds_y d\tau\,. \tag{3.9}$$

Theorem 3.2. *The single layer potential satisfies the following jump relation*

$$\lim_{z\to x} \frac{\partial S}{\partial n_x}(\sigma)(x,t) = \pm\frac{1}{2}\sigma(x,t) + \int_0^t \int_\Gamma \frac{\partial}{\partial n_x} G(x-y, t-\tau)\sigma(y,\tau) ds_y d\tau\,. \tag{3.10}$$

The plus sign applies if z approaches x from within the domain Ω while the minus sign applies if z approaches x from outside Ω. The double layer potential satisfies the following jump relation

$$\lim_{z\to x} D(\sigma)(z,t) = \pm\frac{1}{2}\sigma(x,t) + D(\sigma)(x,t). \tag{3.11}$$

where the plus sign applies for the limit from exterior and the minus sign applies for the limit from interior.

3.3 The Dirichlet Problem

For the Dirichlet problem

$$\begin{cases} U_t - \Delta U = h(x,t), & x\in\Omega,\ t>0 \\ U(x,0) = U_0, & x\in\Omega \\ U(x,t) = f(x,t), & x\in\Gamma\,. \end{cases} \tag{3.12}$$

it is standard to represent U using the double layer heat potential, the volume potential and the initial potential:

$$U(x,t) = D(\sigma)(x,t) + I(U_0)(x,t) + V(h)(x,t), \tag{3.13}$$

where σ is an unknown density function.

From Section 3.1, the representation (3.13) clearly satisfies the first two equations of (3.12). The remaining is to satisfy the boundary condition by solving the unknown density σ. Imposing the boundary condition and using Theorem 3.2 leads to the Volterra integral equation of the second kind for the interior Dirichlet problem:

$$-\frac{1}{2}\sigma(x,t) + D(\sigma)(x,t) = f(x,t) - V(h)(x,t) - I(U_0)(x,t)\,, \tag{3.14}$$

and the integral equation for the exterior Dirichlet problem:

$$\frac{1}{2}\sigma(x,t) + D(\sigma)(x,t) = f(x,t) - V(h)(x,t) - I(U_0)(x,t)\,. \tag{3.15}$$

3.4 The Neumann Problem

For the Neumann problem

$$\begin{cases} U_t - \Delta U = h(x,t), & x \in \Omega,\, t > 0 \\ U(x,0) = U_0, & x \in \Omega \\ \frac{\partial}{\partial n} U(x,t) = f(x,t), & x \in \Gamma\,. \end{cases} \tag{3.16}$$

U is represented by using the single layer heat potential:

$$U(x,t) = S(\sigma)(x,t) + V(h)(x,t) + I(U_0)(x,t). \tag{3.17}$$

The representation (3.17) clearly satisfies the first two equations of (3.16) from Section 3.1. To satisfy the boundary condition, apply the boundary condition and Theorem 3.2, leading to the integral equations

$$\frac{1}{2}\sigma(x,t) + \int_0^t \int_\Gamma \frac{\partial}{\partial n_x} G(x-y, t-\tau)\sigma(y,\tau)ds_y d\tau = f(x,t) - V(h)(x,t) - I(U_0)(x,t) \tag{3.18}$$

and

$$-\frac{1}{2}\sigma(x,t)+\int_0^t\int_\Gamma \frac{\partial}{\partial n_x}G(x-y,t-\tau)\sigma(y,\tau)ds_y d\tau = f(x,t)-V(h)(x,t)-I(U_0)(x,t) \tag{3.19}$$

for the interior Neumann problem and the exterior Neumann problem, respectively.

CHAPTER 4

SUM-OF-EXPONENTIALS APPROXIMATION FOR THE HEAT KERNEL

We develop the sum-of-exponentials approximation for the heat kernel in this chapter by inverting the Laplace transform $\hat{G}(x,s)$ of the heat kernel $G(x,t) = \frac{1}{\sqrt{4\pi t}}e^{-\frac{|x|^2}{4t}}$ numerically through truncated trapezoidal or midpoint rule applied on contour integrals on the condition that $\hat{G}(x,s)$ is sectorial. We also show that power functions $\frac{1}{t^{\beta/2}}, \beta = 1,2,3,4$ can be approximated by sum-of-exponentials. Furthermore, we also develop sum-of-exponentials approximations for heat kernels in higher dimensions with the approximations for the one-dimensional heat kernel and power functions.

4.1 Analytical Apparatus

The following lemma provides an error estimate for the sum-of-exponentials approximation obtained by the truncated trapezoidal rule discretization of a certain contour integral in the Laplace domain.

Lemma 4.1. [Adapted from [47].] *Suppose that $U(z)$ is holomorphic on $W = \mathbb{C} \setminus (-\infty, 0]$ and satisfies the estimate*

$$\|U(z)\| \leq \frac{1}{2|z|^{1/2}} \tag{4.1}$$

for $z \in W$. Suppose that $u(t) = \frac{1}{2\pi i}\int_\Gamma e^{tz}U(z)dz$ is the inverse Laplace transform of U. Suppose further that α and β satisfy the condition $0 < \alpha - \beta < \alpha + \beta < \frac{\pi}{2}$, $0 < \theta < 1$, and that Γ is chosen to be the left branch of the hyperbola

$$\Gamma = \{\lambda T(x) : x \in \mathbb{R}\}, \tag{4.2}$$

where $T(x) = (1 - \sin(\alpha + ix))$. Finally, suppose that $u_n(t)$ is the approximation to $u(t)$ given by the formula

$$u_n(t) = -\frac{h\lambda}{2\pi i} \sum_{k=-n}^{n} U(\lambda T(kh)) T'(kh) e^{\lambda T(kh)t}. \tag{4.3}$$

Then the choice of parameters

$$h = \frac{a(\theta)}{n}, \tag{4.4}$$

$$\lambda = \frac{2\pi\beta n(1-\theta)}{Ta(\theta)}, \tag{4.5}$$

$$a(\theta) = \operatorname{arccosh}\left(\frac{2T}{\delta(1-\theta)\sin\alpha}\right) \tag{4.6}$$

leads to the uniform estimate on $\delta \leq t \leq T$,

$$\|u(t) - u_n(t)\| \leq \frac{1}{\sqrt{t}} \phi(\alpha, \beta) \cdot L(\lambda\delta \sin(\alpha - \beta)/2) \cdot e^{-\frac{2\pi\theta\beta n}{a(\theta)}}, \tag{4.7}$$

where

$$\phi(\alpha, \beta) = \frac{2}{\pi} \sqrt{\frac{1 + \sin(\alpha + \beta)}{1 - \sin(\alpha + \beta)}} (e \sin(\alpha - \beta))^{1/2}, \tag{4.8}$$

$$L(x) = 1 + |\ln(1 - e^{-x})|. \tag{4.9}$$

Proof. Choosing $s = \frac{1}{2}$, $\mu = \frac{1}{2}$ in the first equation of Remark 2 of [47], we have the estimate

$$\|u(t) - u_n(t)\| \leq \frac{1}{2\sqrt{t}} \phi(\alpha, \beta) \cdot L(\lambda t \sin(\alpha - \beta)/2) \cdot e^{\lambda t} \left(\frac{1}{e^{2\pi\beta/h} - 1} + \frac{1}{e^{\lambda t \sin\alpha \cosh(nh)/2}}\right). \tag{4.10}$$

With the choice of parameters given by equations (4.4)-(4.6), it is easy to see that

$$\begin{aligned} e^{\lambda t} &= e^{\frac{2\pi\beta n(1-\theta)t}{Ta(\theta)}} \le e^{\frac{2\pi\beta n(1-\theta)}{a(\theta)}}, \qquad \delta \le t \le T, && (4.11)\\ \frac{1}{e^{2\pi\beta}h - 1} &\approxeq e^{-2\pi\beta/h} = e^{-\frac{2\pi\beta n}{a(\theta)}}, && (4.12)\\ \frac{1}{e^{\lambda t \sin\alpha \cosh(nh)}/2} &= e^{-\frac{2\pi\beta n t}{a(\theta)\delta}} \le e^{-\frac{2\pi\beta n}{a(\theta)}}, \qquad \delta \le t \le T, && (4.13) \end{aligned}$$

Finally, it is easy to see that $L(x)$ is decreasing in x and thus

$$L(\lambda t \sin(\alpha-\beta)/2) \le L(\lambda\delta \sin(\alpha-\beta)/2), \qquad \delta \le t \le T. \tag{4.14}$$

Substituting equations (4.11)-(4.14) into equation (4.10), we obtain equation (4.7). □

Remark 4.1. *The parameters α, β, and θ are available for optimization. For our problem, i.e., the sum-of-exponentials approximation of the 1D heat kernel, we choose $\alpha = 0.8$, $\beta = 0.7$ in equation* (4.8)*. We have also tested various values of θ in* $(0, 1)$*. Numerical experiments indicate that the number of nodes needed for a prescribed accuracy is relatively insensitive when θ is in the range* $[0.85, 0.95]$*.*

Combining Lemma 4.1 and Remark 4.1, we have the following Corollary.

Corollary 4.1. *Suppose that $0 < \epsilon < 0.1$ is the prescribed relative error and that $T \ge 1000\delta > 0$. Then under the conditions of Lemma 4.1, the following estimate holds*

$$\|u(t) - u_n(t)\| \le \frac{1}{\sqrt{t}} \cdot \epsilon, \qquad \delta \le t \le T, \tag{4.15}$$

if the number of exponentials n satisfies the following estimate:

$$n = O\left(\left(\log\left(\frac{1}{\epsilon}\right) + \log\log\left(\frac{T}{\delta}\right)\right)\log\left(\frac{T}{\delta}\right)\right). \tag{4.16}$$

Proof. We choose the parameters α, β, and θ as in Remark 4.1. The factor $\phi(\alpha, \beta)$ in equation (4.7) is just a fixed constant independent of n, T, and δ. For $T \ge 1000\delta > 0$, the parameter $a(\theta)$ defined in equation (4.6) satisfies the estimate

$$a(\theta) = O(\log \frac{T}{\delta}). \tag{4.17}$$

Moreover, the function L defined in equation (4.9) is decreasing for $x > 0$, $L(x) \approx |\ln x|$ as $x \to 0^+$, and $L(x) \to 1$ as $x \to +\infty$. Combining this observation with the assumption equation (4.16), we observe that the factor $L(\lambda\delta\sin(\alpha-\beta))$ in equation (4.7) satisfies the estimate

$$L(\lambda\delta\sin(\alpha-\beta)) \leq C \log\left(\frac{T}{\delta}\right). \tag{4.18}$$

Substituting equation (4.17) and equation (4.18) into equation (4.7), we obtain

$$\|u(t) - u_n(t)\| \leq \frac{C_1}{\sqrt{t}} \cdot \log\left(\frac{T}{\delta}\right) \cdot e^{-C_2 n/\log\left(\frac{T}{\delta}\right)}. \tag{4.19}$$

It is then easy to see that equation (4.15) follows if n satisfies equation (4.16). □

Remark 4.2. *For most practical cases, the* $\log\log\left(\frac{T}{\delta}\right)$ *factor is much smaller than the* $\log\left(\frac{1}{\epsilon}\right)$ *factor. Thus for a fixed precision* ϵ*, we will simply say that* $n = O\left(\log\left(\frac{T}{\delta}\right)\right)$.

Remark 4.3. *The hyperbolic contour is chosen in such a way that the horizontal strip* $D_\beta = \{z \in \mathbb{C} : |Imz| \leq \beta\}$ *is transformed into a region bounded by the left branches of two hyperbolas defined as in equation* (4.2)*, but with* x *replaced by* $x \pm i\beta$*. The reason that such contour is chosen is the (well-known) fact that the trapezoidal rule converges exponentially fast for functions holomorphic on a horizontal strip containing the real axis (see, for example, [59, 60]). We have actually used the midpoint rule to eliminate the occurrence of a node directly on the real axis. There is almost no difference in terms of accuracy, but this allows us to assume that all nodes lie in the upper half plane in actual computation.*

Remark 4.4. *It is likely that other contours would yield similar results. Trefethen* et al. *[63] have analyzed this issue with great care and presented a detailed comparison* *of various contours (hyperbolic, parabolic, and Talbot contours [61]) for inverting sectorial Laplace transforms, though they are mainly concerned with the efficiency of various contours for a fixed time t.*

4.2 Sum-of-exponentials Approximation for Power Functions

In this section, we show how to construct the efficient sum-of-exponentials approximation for the power function $\frac{1}{t^2}$ using the techniques in [38] for building the sum-of-exponentials approximation for $\frac{1}{\sqrt{t}}$. Then it would be straightforward to extend the techniques for constructing the sum-of-exponentials approximations for $\frac{1}{t}$ and $\frac{1}{t^{\frac{3}{2}}}$. First, we have the following observation

$$\frac{1}{t^2} = \int_0^\infty s e^{-st}\, ds\,. \tag{4.20}$$

Lemma 4.2. *For $t \geq \delta \geq 0$,*

$$\left| \int_p^\infty s e^{-st}\, ds \right| \leq \epsilon \frac{1}{t^2} \tag{4.21}$$

where $\epsilon = (p\delta + 1)e^{-p\delta}$.

Proof. Let $u = st$,

$$\left| \int_p^\infty s e^{-st}\, ds \right| = \left| \int_{pt}^\infty u e^{-u}\, du \right| \frac{1}{t^2} < \left| \int_{p\delta}^\infty u e^{-u}\, du \right| \frac{1}{t^2} < (p\delta + 1)e^{-p\delta}\frac{1}{t^2} \tag{4.22}$$

□

The error approaches to zero as p grows to infinity. Then we will show dyadic intervals (as in [28]) using Gauss-Legendre quadrature can be applied to discretize the integral on the interval $[0, p]$.

Lemma 4.3. *Consider a dyadic interval $[a, b] = [2^j, 2^{j+1}]$ and let $s_1, ..., s_n$ and $w_1, ..., w_n$ be the nodes and weights for n-point Gauss-Legendre quadrature on the* *interval. Then*

$$\left|\int_a^b se^{-st}\,ds - \sum_{k=1}^{n} e^{-s_k t} s_k w_k\right| < \sqrt{\pi}2^{-4n+5}n^{\frac{3}{2}}\frac{1}{t^2}. \tag{4.23}$$

Proof. From the n-point Gauss-Legendre quadrature, we have

$$\left|\int_a^b se^{-st}\,ds - \sum_{k=1}^{n} e^{-s_k t} s_k w_k\right| = \frac{(b-a)^{2n+1}}{2n+1}\frac{(n!)^4}{[(2n)!]^3}\left|D_\xi^{2n}\xi e^{-\xi t}\right|, \quad a < \xi < b, \tag{4.24}$$

where D_ξ denotes the partial derivative with respect to ξ.

Based on the equation

$$\left|D_\xi^{2n}\xi e^{-\xi t}\right| = \left|(-t)^{2n}\xi e^{-\xi t} + 2n(-t)^{2n-1}e^{-\xi t}\right| \; < 2a(t)^{2n}e^{-at} + 2n(-t)^{2n-1}e^{-at} \tag{4.25}$$

we obtain

$$\left|\int_a^b se^{-st}\,ds - \sum_{k=1}^{n} e^{-s_k t} s_k w_k\right| < \frac{2}{2n+1}\frac{(n!)^4}{[(2n)!]^3}[(at)^{2n+2}e^{-at} + n(at)^{2n+1}e^{-at}]\,\frac{1}{t^2}. \tag{4.26}$$

Using the Stirling's approximation

$$\sqrt{2\pi}n^{n+1/2}e^{-n} < n! < 2\sqrt{\pi}n^{n+1/2}e^{n}, \tag{4.27}$$

and the following equations

$$\max_{x>0} x^n e^{-x} = (n)^n e^{-n}, \quad n \in R^+\,, \tag{4.28}$$

$$\left(1+\frac{1}{n}\right)^n < e\,, \tag{4.29}$$

it is easy to derive equation (4.23). □

Lemma 4.4. *For* $0 < t \leq T$,

$$\left|\int_0^a se^{-st}\,ds - \sum_{k=1}^{n} e^{-s_k t} s_k w_k\right| < 2\sqrt{\pi}n^{\frac{3}{2}}(\frac{aT}{n})^{2n+1}(\frac{e}{8})^{2n}\frac{1}{t^2}\,. \tag{4.30}$$

Proof. The proof is very similar to the preceding lemma. First, we get

$$\left|\int_0^a se^{-st}\,ds - \sum_{k=1}^{n} e^{-s_k t} s_k w_k\right| < \frac{1}{2n+1}\frac{(n!)^4}{[(2n)!]^3}[(at)^{2n+2} + 2n(at)^{2n+1}]\,\frac{1}{t^2}. \qquad (4.31)$$

instead of equation (4.26). Then the result follows after applying equation (4.27), equation (4.28) and equation (4.29) to equation (4.30). □

Combining the preceding three lemmas, it is straightforward to come up with the sum-of-exponentials approximation for $\frac{1}{t^2}$.

Theorem 4.1. *Let $\epsilon > 0$ be the desired precision, let $0 < \delta \le t \le T$, let $n_o = O(\log(\frac{1}{\epsilon}))$, let $N_L = O(\log(\frac{n_o}{T}))$, and let $N_M = O(\log\log(\frac{1}{\epsilon}) + \log(\frac{1}{\delta}))$. Further, let $s_{0,1}, ..., s_{0,n_0}$ and $w_{0,1}, ..., w_{0,n_o}$ be the nodes and weights for the n_o-point Gauss-Legendre quadrature on the interval $[0, 2^{N_L}]$, and let $s_{j,1}, ..., s_{j,n}$ and $w_{j,1}, ..., w_{j,n}$ be the nodes and weights for the n-point Gauss-Legendre quadrature on the interval $[2^j, 2^{j+1}]$, where $n = O(\log(\frac{1}{\epsilon}))$. Then,*

$$\left|\frac{1}{t^2} - \left(\sum_{k=1}^{n_o} e^{-s_{o,k}t} s_{o,k} w_{o,k} + \sum_{j=N_L}^{N_M}\sum_{k=1}^{n} e^{-s_{j,k}t} s_{j,k} w_{j,k}\right)\right| < 3\epsilon\frac{1}{t^2} \qquad (4.32)$$

Remark 4.5. *The theorem above shows that we only need $O\left(\log(\frac{1}{\epsilon})\left(\log\log(\frac{1}{\epsilon}) + \log(\frac{T}{\delta})\right)\right)$ to approximate $\frac{1}{t^2}$ for $0 < \delta \le t \le T$.*

Remark 4.6. *The sum-of-exponentials approximation for $\frac{1}{\sqrt{t}}$ can be found in [38]. Moreover, similar sum-of-exponentials approximations can be constructed for $\frac{1}{t}$ and $\frac{1}{t^{\frac{3}{2}}}$ using the same techniques for $\frac{1}{t^2}$ based on the observations*

$$\frac{1}{t} = \int_0^\infty e^{-st}\,ds\,, \qquad (4.33)$$

and

$$\frac{1}{t^{\frac{3}{2}}} = \frac{4}{\sqrt{\pi}}\int_0^\infty s^2 e^{-s^2 t}\,ds\,. \qquad (4.34)$$

Then it would be sufficient for the heat kernel in one, two, and three dimensions.

Remark 4.7. *In [8], the error of the approximation of $1/x$ by exponential sums is studied in detailed on both finite and infinite intervals. In [9], Braess and Hackbusch extend their analysis to the general power function $1/x^{\alpha}$, $\alpha > 0$, obtaining sharp error estimates for the absolute error.*

4.3 Sum-of-exponentials Approximation of the Heat Kernel

In this section, we first develop a separated sum-of-exponentials approximation for the one-dimensional heat kernel. We then extend the approximation to arbitrary space dimensions, and to directional derivatives of the heat kernel, such as the kernel of the double layer potential $D(\sigma)$.

The one-dimensional result is summarized by the following theorem.

Theorem 4.2. *Let $G(x,t) = \frac{1}{\sqrt{4\pi t}} e^{-\frac{|x|^2}{4t}}$ denote the one-dimensional heat kernel and let its Laplace transform be denoted by $\hat{G}(x,s) = \int_0^\infty e^{-st} G(x,t) dt$. Then, for $s > 0$ and $x \in \mathbb{R}$,*

$$\hat{G}(x,s) = \frac{1}{2\sqrt{s}} e^{-\sqrt{s}|x|}. \tag{4.35}$$

Furthermore, let $0.1 > \epsilon > 0$ be fixed accuracy and $T \geq 1000\delta > 0$. Then there exists a sum-of-exponentials approximation

$$G_A(x,t) = \sum_{k=-n}^{n} w_k e^{s_k t} e^{-\sqrt{s_k}|x|} \tag{4.36}$$

such that

$$|G(x,t) - G_A(x,t)| < \frac{1}{\sqrt{t}} \cdot \epsilon \tag{4.37}$$

for $x \in \mathbb{R}$ and $t \in [\delta, T]$, with

$$n = \mathcal{O}\left(\log(\frac{T}{\delta}) \left(\log(\frac{1}{\epsilon}) + \log\log\left(\frac{T}{\delta}\right) \right) \right). \tag{4.38}$$

Proof. The formula (4.35) is well-known and can be derived from standard Laplace transform tables. The natural extension of $\hat{G}(x,s)$ to the complex s-plane has an obvious branch point at $s=0$, and a branch cut along the negative real axis assuming the restriction $-\pi < s \leq \pi$. Thus, $\hat{G}(x,s)$ is holomorphic on $W = \mathbb{C} \setminus (-\infty, 0]$ and satisfies the estimate

$$\|\hat{G}(x,s)\| \leq \frac{1}{2|s|^{1/2}} \tag{4.39}$$

for $s \in W$ and all $x \in \mathbb{R}$. As a result, $G(x,t)$ can be represented by the inverse Laplace transform:

$$\begin{aligned} G(x,t) &= \frac{1}{2\pi i} \int_\Gamma e^{st} \hat{G}(x,s) ds \\ &= \frac{1}{4\pi i} \int_\Gamma \frac{1}{\sqrt{s}} e^{st-\sqrt{s}|x|} ds, \end{aligned} \tag{4.40}$$

where Γ is a simple contour lying in $W = \mathbb{C} \setminus (-\infty, 0]$, and parametrizable by a regular mapping $S : (-\infty, +\infty) \rightarrow \mathbb{C}$ such that

$$\lim_{x\rightarrow\pm\infty} \mathrm{Im} S(x) = \pm\infty \qquad \text{and} \qquad \lim_{x\rightarrow\pm\infty} \frac{\mathrm{Re}\,(S(x))}{|x|} < 0.$$

The last condition implies that

$$\mathrm{Re}(z) \leq -b|z|, \qquad \text{as } z \rightarrow \infty, z \in \Gamma,$$

for some $b > 0$. This forces the integral equation (4.40) to be absolutely convergent. It is easy to see that the integral is independent of the choice of Γ.

The main result equation (4.36) is now a direct consequence of Lemma 4.1 and Corollary 4.1 since $\hat{G}$ satisfies the condition equation (4.39). In particular, we have

$$G_A(x,t) = \sum_{k=-n}^{n} w_k e^{s_k t} e^{-\sqrt{s_k}|x|}, \tag{4.41}$$

where

$$s_k = \lambda(1 - \sin(\alpha + ikh)), \tag{4.42}$$

$$w_k = -\frac{h}{4\pi i \sqrt{s_k}} s_k', \tag{4.43}$$

with $s_k' = -\lambda i \cos(\alpha + ikh)$, and the parameters h, λ specified in equation (4.4) and equation (4.5), respectively. □

In Table 4.1, we list the number of exponentials needed to approximate the 1D heat kernel for $x \in \mathbb{R}$ over three different time intervals $[\delta, T]$: $I_1 = [10^{-3}, 1]$, $I_2 = [10^{-3}, 10^3]$, $I_3 = [10^{-5}, 10^4]$, which correspond roughly to 10^3, 10^6, and 10^9 time steps, respectively. The first column lists the maximum error of the approximation computed over a 50×1000 grid (x_j, t_k) where $x_0 = 0$, $x_j = 2^{-16+j}$ for $j = 1, \cdots, 49$, and the t_k are 1000 samples on $[\delta, T]$ chosen to be equispaced on a logarithmic scale. The node locations are plotted in Fig. 4.1.

Table 4.1 Number of Exponentials Needed to Approximate the 1D Heat Kernel for $x \in \mathbb{R}$ over Different Time Intervals: $I_1 = [10^{-3}, 1]$, $I_2 = [10^{-3}, 10^3]$, $I_3 = [10^{-5}, 10^4]$.

ϵ	I_1	I_2	I_3
10^{-3}	15	23	32
10^{-6}	31	50	68
10^{-9}	47	77	105

Remark 4.8. *To obtain Table 4.1, we set* $\alpha = 0.8$, $\beta = 0.7$, $\theta = 0.9$ *for* I_1 *and* $\theta = 0.95$ *for* I_2 *and* I_3.

Remark 4.9. *The coefficients* w_i *are not positive (but complex) and thus stability is an issue. It is difficult to bound the expression*

$$\frac{\sum |w_i \exp(s_i t)|}{|\sum w_i \exp(s_i t)|}$$

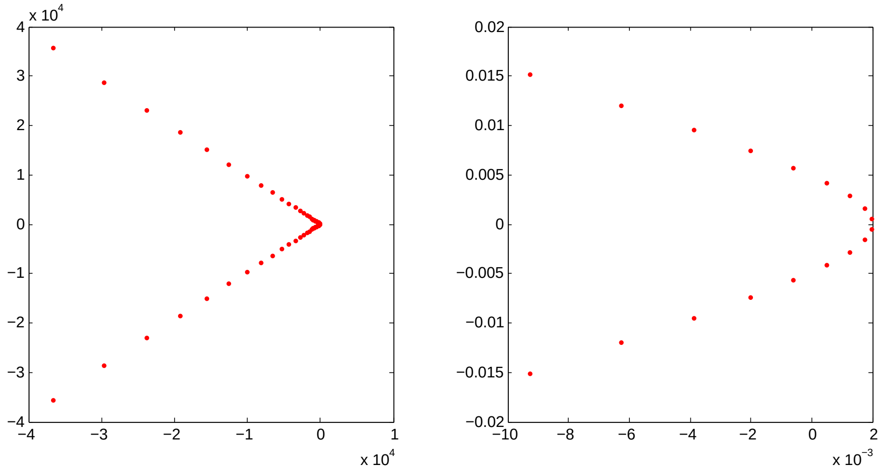

Figure 4.1 The location of the exponential nodes s_k ($k = \pm 1, \cdots, \pm n$ for $n = 77$) used in equation (4.41) to approximate the 1D heat kernel for $t \in [10^{-3}, 10^3]$ with 9-digit accuracy. The left figure shows all nodes, while the right figure is a close-up of the nodes near the origin. All nodes lie on the left branch of the hyperbola specified in equation (4.2).

analytically for all time. However, we have checked the value of the expression numerically for all cases presented in Table 4.1, and found that it is roughly 1.08. *This suggests that the sum-of-exponentials approximation is, indeed, well-conditioned.*

Remark 4.10. *If we make a further change of variable* $z = \sqrt{s}$ *in equation* (4.40), *we obtain*

$$G(x,t) = \frac{1}{2\pi i} \int_{\Gamma'} e^{z^2 t - z|x|} dz, \tag{4.44}$$

where Γ' *is any contour lying in the sector* $\{\frac{\pi}{4} < |\arg z| \leq \frac{\pi}{2}\}$ *of the complex plane. In particular, if* Γ' *is chosen to be the imaginary axis, then we essentially recover the Fourier integral representation of the heat kernel (see, for example, [31]).*

Finally, it is worth repeating that the number of terms required in the sum-of-exponentials approximation does not depend on the spatial extent of the problem.

4.3.1 Heat Kernels in Higher Dimensions

Suppose now that we are interested in heat flow in $\mathbb{R}^d$, for $d \geq 2$, where the heat kernel is

$$G_d(x,t) = \frac{1}{(4\pi t)^{d/2}} e^{-\frac{|x|^2}{4t}}.$$

The following theorem describes an efficient sum-of-exponentials representation.

Theorem 4.3. *For any $0.1 > \epsilon > 0$ and $T \geq 1000\delta > 0$, the heat kernel $G_d(x,t)$ admits the following approximation:*

$$\tilde{G}_d(x,t) = \sum_{j=1}^{N_2} \tilde{w}_j e^{-\lambda_j t} \sum_{k=-N_1}^{N_1} w_k e^{s_k t} e^{-\sqrt{s_k}|x|} \tag{4.45}$$

such that

$$|G_d(x,t) - \tilde{G}_d(x,t)| < \frac{1}{t^{d/2}} \cdot \epsilon \tag{4.46}$$

for any $x \in \mathbb{R}^d$, $t \in [\delta, T]$ with N_1 specified in equation (4.38) and N_2 specified in Remark 4.5.

Proof. We first rewrite the d dimensional heat kernel as a product of two functions:

$$G_d(x,t) = G_1(x,t) \cdot F(t), \tag{4.47}$$

where

$$G_1(x,t) = \frac{1}{(4\pi t)^{1/2}} e^{-\frac{|x|^2}{4t}}, \qquad F(t) = \frac{1}{(4\pi t)^{(d-1)/2}}. \tag{4.48}$$

Similarly, we rewrite $\tilde{G}_d$ as follows:

$$\tilde{G}_d(x,t) = S_1(x,t) \cdot S_2(t), \tag{4.49}$$

where

$$S_1(x,t) = \sum_{k=-N_1}^{N_1} w_k e^{s_k t} e^{-\sqrt{s_k}|x|}, \qquad S_2(t) = \sum_{j=1}^{N_2} \tilde{w}_j e^{-\lambda_j t}. \tag{4.50}$$

Using the triangle inequality, we have

$$
\begin{aligned}
|G_d(x,t) - \tilde{G}_d(x,t)| &\leq |G_1 \cdot F - G_1 \cdot S_2| + |G_1 \cdot S_2 - S_1 \cdot S_2| \\
&\leq |G_1| \cdot |F - S_2| + (1+\epsilon)|F| \cdot |G_1 - S_1| \qquad (4.51) \\
&\leq \frac{3}{t^{d/2}} \cdot \epsilon,
\end{aligned}
$$

and the result follows. □

4.3.2 Double Layer Heat Potential

Since we often rely on the double layer potential in integral equation methods, it is worth writing down a sum-of-exponentials approximation for this case as well. We denote the kernel of the double layer heat potential by

$$D(x,y;t) = \frac{\partial G(x-y,t)}{\partial n_y} = \frac{(x-y) \cdot n_y}{(4\pi t)^{d/2} 2t} e^{-\frac{|x-y|^2}{4t}}.$$

Theorem 4.4. *Let $0.1 > \epsilon > 0$ be fixed accuracy, $R > 1$ and $T \geq 1000\delta > 0$. Then there exists a sum-of-exponentials approximation*

$$D_A(x,y;t) = \sum_{j=1}^{N_2} \tilde{w}_j e^{-\lambda_j t} \sum_{k=-N_1}^{N_1} w_k e^{s_k t} e^{-\sqrt{s_k}|x-y|} (x-y) \cdot n_y \qquad (4.52)$$

such that

$$|D(x,y;t) - D_A(x,y;t)| < \frac{1}{t^{(d+2)/2}} \cdot \epsilon \qquad (4.53)$$

for $|x-y| \leq R$ and $t \in [\delta, T]$. Here N_1 and N_2 are as follows:

$$N_1 = \mathcal{O}\left(\log(\frac{T}{\delta})\left(\log(\frac{1}{\epsilon}) + \log\log\left(\frac{T}{\delta}\right) + \log R\right)\right) \qquad (4.54)$$

and

$$N_2 = O\left((\log\left(\frac{1}{\epsilon}\right) + \log R) \cdot \left(\log\frac{T}{\delta} + \log\log\frac{1}{\epsilon} + \log R\right)\right). \qquad (4.55)$$

Proof. We first introduce $\hat{\epsilon} = \epsilon/R$. Then by Theorem 1, there exist $N_1 = \mathcal{O}\left(\log(\frac{T}{\delta})\log(\frac{1}{\hat{\epsilon}})\right) = \mathcal{O}\left(\log(\frac{T}{\delta})\left(\log(\frac{1}{\epsilon}) + \log\log\left(\frac{T}{\delta}\right) + \log R\right)\right)$, coefficients w_k, and nodes s_k $(k = 1, \cdots, N_1)$ such that

$$|G_1(x,t) - \sum_{k=-N_1}^{N_1} w_k e^{s_k t} e^{-\sqrt{s_k}|x|}| < \frac{1}{\sqrt{t}} \cdot \hat{\epsilon}, \tag{4.56}$$

for $x \in \mathbb{R}$ and $t \in [\delta, T]$.

Changing x to $x-y$ and multiplying both sides of equation (4.56) by $(x-y)\cdot n_y$, we obtain

$$\begin{aligned} &\left| \frac{(x-y)\cdot n_y}{(4\pi t)^{1/2}} e^{-\frac{|x-y|^2}{4t}} - \sum_{k=-N_1}^{N_1} w_k e^{s_k t} e^{-\sqrt{s_k}|x-y|} \cdot ((x-y)\cdot n_y) \right| \\ &\leq \frac{|(x-y)\cdot n_y|}{(4\pi t)^{1/2}} \cdot \hat{\epsilon} \\ &\leq \frac{R}{(4\pi t)^{1/2}} \cdot \frac{\epsilon}{R} \\ &\leq \frac{1}{(4\pi t)^{1/2}} \cdot \epsilon \end{aligned} \tag{4.57}$$

for all $|x-y| \leq R$ and $t \in [\delta, T]$.

Similarly, by Lemma 2, we have

$$\left| \frac{1}{t^{(d+1)/2}} - \sum_{i=1}^{N_2} \tilde{w}_i e^{-\lambda_i t} \right| \leq \frac{1}{t^{(d+1)/2}} \cdot \epsilon/R, \qquad \delta \leq t \leq T, \tag{4.58}$$

where N_2 is given by equation (4.55).

The result is then obtained by an argument almost identical to that in the proof of Theorem 4.3. □

Remark 4.11. *It is worth noting that the theorems above provide what are, in essence, relative error estimates. Our numerical experiments also indicate that the* $\log R$ *dependence in* N_1 *and* N_2 *and the restriction on* $x-y$ *are somewhat artificial since* $D(x,y;t)$ *and* $D_A(x,y;t)$ *are exponentially small for large* $(x-y)$.

CHAPTER 5

NUMERICAL ALGORITHMS

In this chapter, we present fast algorithms for the exterior and interior Dirichlet problems governed by the heat equation (3.1). From Chapter 3, we need to solve the density function σ in the integral equation

$$\frac{1}{2}\sigma(x,t) + D(\sigma)(x,t) = f(x,t) \tag{5.1}$$

for the exterior problem and the integral equation

$$-\frac{1}{2}\sigma(x,t) + D(\sigma)(x,t) = f(x,t) \tag{5.2}$$

for the interior problem.

5.1 Split of the Layer Potentials

It is convenient both analytically and numerically to decompose the double layer potential into two pieces: a *history* part D_H and a *local* part D_L. Let δ be a small positive parameter, we write

$$D(\sigma)(x,t) = D_H(\sigma)(x,t) + D_L(\sigma)(x,t),$$

where

$$D_H(\sigma)(x,t) = \int_0^{t-\delta}\int_\Gamma \frac{\partial}{\partial n_y} G(x-y,t-\tau)\sigma(y,\tau)ds_y d\tau \tag{5.3}$$

and

$$D_L(\sigma)(x,t) = \int_{t-\delta}^{t}\int_\Gamma \frac{\partial}{\partial n_y} G(x-y,t-\tau)\sigma(y,\tau)ds_y d\tau\,. \tag{5.4}$$

The equation (5.1) can be written in the following form

$$\frac{1}{2}\sigma(x,t) + D_L(\sigma)(x,t) = f(x,t) - D_H(\sigma)(x,t). \tag{5.5}$$

5.2 Efficient 2D Heat Solver with Stationary Geometry Using SOE

5.2.1 Expressions of the Local Kernels

In order to evaluate the local part accurately, we switch the order of integration and carry out the integration in time semi-analytically using product integration (see, for example, [40, 44]). Since the density σ is smooth in time, we expand it on $[t-\delta, t]$ for each y in the form

$$\sigma(y,\tau) = \sigma_0(y) + (t-\tau)\sigma_1(y) + \cdots + \frac{(t-\tau)^{k-1}}{(k-1)!}\sigma_{k-1}(y) + O((t-\tau)^k).$$

The functions $\sigma_0(y), \ldots, \sigma_{k-1}(y)$ are obtained from the function values $\sigma(y, t-j\Delta t)$ for $j = 0, \cdots, k-1$ via standard polynomial interpolation. In other words, we have

$$\sigma(y,\tau) = [1 \quad \frac{t-\tau}{\Delta t} \quad \ldots \quad \frac{(t-\tau)^{k-1}}{\Delta t^{k-1}}]\, M_k \begin{bmatrix} \sigma(y,t) \\ \sigma(y,t-\Delta t) \\ \ldots \\ \sigma(y, t-(k-1)\Delta t) \end{bmatrix} + O((t-\tau)^k), \tag{5.6}$$

where M_k is the coefficient matrices given by the formulas

$$M_2 = \begin{bmatrix} 1 & 0 \\ -1 & 1 \end{bmatrix}, \tag{5.7}$$

$$M_3 = \begin{bmatrix} 1 & 0 & 0 \\ -\frac{3}{2} & 2 & -\frac{1}{2} \\ \frac{1}{2} & -1 & \frac{1}{2} \end{bmatrix}, \tag{5.8}$$

$$M_4 = \begin{bmatrix} 1 & 0 & 0 & 0 \\ -\frac{11}{6} & 3 & -\frac{3}{2} & \frac{1}{3} \\ 1 & -\frac{5}{2} & 2 & -\frac{1}{2} \\ -\frac{1}{6} & \frac{1}{2} & -\frac{1}{2} & \frac{1}{6} \end{bmatrix}. \tag{5.9}$$

Substituting the above expression into equation (5.4) and changing the order of integration in time and space, we obtain

$$D_L(\sigma)(x,t) = \int_\Gamma [D_{L_0} \quad D_{L_1} \quad ... \quad D_{L_{k-1}}]\, M_k \begin{bmatrix} \sigma(y,t) \\ \sigma(y,t-\Delta t) \\ ... \\ \sigma(y,t-(k-1)\Delta t) \end{bmatrix} ds_y \tag{5.10}$$

where the local kernels D_{L_k} are given by the formulas

$$\begin{aligned} D_{L_k}(x,y) &= \int_{t-\delta}^{t} \frac{\partial}{\partial n_y} G_d(x-y,t-\tau)(t-\tau)^k\, d\tau \\ &= \frac{1}{\Delta t^k} \int_{t-\delta}^{t} \frac{(x-y)\,\cdot\, n_y}{2^{d+1}\pi^{\frac{d}{2}}(t-\tau)^{\frac{d}{2}+1-k}} e^{-\frac{|x-y|^2}{4(t-\tau)}}\, d\tau. \end{aligned} \tag{5.11}$$

In two dimensions, we have the following explicit expressions for the local kernels

$$D_{L_k}(x,y) = \begin{cases} \frac{(x-y)\cdot n_y}{2\pi|x-y|^2}\, e^{-\rho}\,, & k=0, \\ \frac{(x-y)\cdot n_y}{8\pi\Delta t}\, Ei(\rho)\,, & k=1, \\ \frac{(x-y)\cdot n_y \delta}{8\pi\Delta t^2}\left(e^{-\rho} - \rho Ei(\rho)\right), & k=2, \\ \frac{(x-y)\cdot n_y \delta^2}{16\pi\Delta t^3}\left((1-\rho)e^{-\rho} + \rho^2 Ei(\rho)\right), & k=3, \end{cases} \tag{5.12}$$

where $\rho = \frac{|x-y|^2}{4\delta}$ and Ei is the exponential integral function given by

$$Ei(x) = \int_x^\infty \frac{e^{-t}}{t}\, dt.$$

5.2.2 Discretization and Compression of the Spatial Integrals in the Local Part

The local kernels $D_{L_k}(x, y)$ are logarithmically singular at the diagonal. There are many high-order quadratures for discretizing such weakly singular integrals. Here we use the Alpert quadrature [1] to discretize the spatial integrals involving these local kernels. We would also like to avoid the $O(N_S^2)$ work that would be required by direct evaluation of the matrix-vector product. A large number of fast algorithms are now available to reduce the cost of this step to $O(N_S)$ or $O(N_S \log N_S)$. These include fast multipole methods, kernel-independent fast multipole methods [68], HSS and H-matrix methods [14, 33], and HBS or recursive skeletonization methods [35, 36, 22]. We use recursive skeletonization factorization developed in [36]. That is, each of the operators $D_{L,j}(x_m, y_n)$ $(j = 1, \cdots, k-1)$ will be compressed once, with subsequent applications of the operator computed in optimal complexity with much smaller prefactor than the FMMs.

`Rskelf` requires the access of matrix entries and submatrix A_{ts}, where t and s are both set of indices. Although we could apply `rskelf` directly on the matrix A resulted from the discretization of the spatial integrals using the Alpert quadrature, the prefactore in the compression stage would be large since the Alpert quadrature contains nonequispaced nodes near the diagonal. In order to reduce the prefactor in the compression phase, we split the matrix A into three parts:

$$A = A_{eq} + A_{alp} - A_{adj}, \tag{5.13}$$

where

- A_{eq}: the matrix obtained by using the equispaced trapezoidal rule to discretize the spatial integral with diagonal entries set to 0. Only A_{eq} is compressed by `rskelf` and the matrix entries of A_{eq} can be evaluated efficiently. The matrix compression and apply costs are both $O(N_S)$ with the apply step much faster than the compression step (very often 100 times faster).

- A_{alp}: the matrix which contains the effect of the non-equispaced points in the Alpert quadrature. This matrix is not sparse since we use global spectral interpolation to interpolate the function values at those nonequispaced points to equispaced points. However, since there are only 30 nonequispaced points near the diagonal and one could apply NUFFT to speed up the global interpolation. The cost of the matrix-vector product $A_{alp}\sigma$ is $O(30N_S\log(N_S))$.
- A_{adj}: a banded matrix which contains the effect of the adjacent 18 equispaced points near the diagonal from the trapezoidal rule. We need to subtract them since they are not in the Alpert quadrature. Evaluating $A_{adj}\sigma$ costs $O(18N_S)$.

Remark 5.1. *In the case of stationary boundary, one could also compress the matrix inverse directly using the aforementioned fast direct solvers. This will lead to an optimal algorithm for long-time simulations. When the number of time step is not too large, our method is more efficient since (a) the compression cost is reduced by a large factor; (b) the apply cost is much smaller than the compression cost; and (c) it only takes about* 6 *iterations for GMRES to converge to* 12*-digit accuracy due to the second kind Volterra structure of the integral equation.*

5.2.3 Evaluation of the History Part

For the history part, approximating the kernel by $D_A(x, y; t-\tau)$ (4.52) and substituting equation (4.52) into equation (5.3), we obtain

$$
\begin{aligned}
D_H(\sigma)(x,t) \approx \int_0^{t-\delta} & \int_\Gamma \sum_{j=1}^{N_2} \tilde{w}_j e^{-\lambda_j(t-\tau)} \\
& \sum_{k=-N_1}^{N_1} w_k e^{s_k(t-\tau)} e^{-\sqrt{s_k}|x-y|}[(x-y)\cdot n_y]\sigma(y,\tau)ds_y d\tau \\
& = \sum_{j=1}^{N_2} \tilde{w}_j \sum_{k=-N_1}^{N_1} w_k H_{j,k}(x,t),
\end{aligned}
\tag{5.14}
$$

where each history mode $H_{j,k}$ is given by the formula

$$
H_{j,k}(x,t) = \int_0^{t-\delta} e^{(-\lambda_j+s_k)(t-\tau)} V_k(x,\tau)d\tau, \tag{5.15}
$$

with V_k given by the formula

$$
V_k(x,\tau) = \int_\Gamma e^{-\sqrt{s_k}|x-y|}[(x-y)\cdot n_y]\sigma(y,\tau)ds_y. \tag{5.16}
$$

Here, we have interchanged the order of summation and integration.

For each fixed τ, $V_k(x,\tau)$ can be discretized using the trapezoidal rule to achieve spectral accuracy. This is because although each integral is not smooth, the kernel of the whole history part is smooth since the heat kernel is smooth for $\tau \in [0, t-\delta]$. The resulting discrete summation can again be computed via `rskelf` The computational cost for this step is $\mathcal{O}(N_S)$ for each k. Once the V_k have been evaluated, each history mode $H_{j,k}$ can be computed recursively, as in [31, 40]:

$$
H_{j,k}(x,t+\Delta t) = e^{(-\lambda_j+s_k)\Delta t} H_{j,k}(x,t) + \int_{t-\delta}^{t+\Delta t-\delta} e^{(-\lambda_j+s_k)(t+\Delta t-\tau)} V_k(x,\tau)d\tau. \tag{5.17}
$$

Similar to local kernels, V_k can be expressed as

$$
V_k(x,\tau) = [1 \quad \frac{t-\delta-\tau}{\Delta t} \quad ... \quad \frac{(t-\delta-\tau)^{k-1}}{\Delta t^{k-1}}] \, M_k \begin{bmatrix} V_k(x,t-\delta) \\ V_k(x,t-\delta-\Delta t) \\ ... \\ V_k(x,t-\delta-(k-1)\Delta t) \end{bmatrix} + O(\Delta t^k). \tag{5.18}
$$

Then

$$\int_{t-\delta}^{t+\Delta t-\delta} e^{(-\lambda_j+s_k)(t+\Delta t-\tau)} V_k(x,\tau) d\tau =$$

$$[D_{H_0} \quad D_{H_1} \quad ... \quad D_{H_{k-1}}]\, M_k \begin{bmatrix} V_k(x,t-\delta) \\ V_k(x,t-\delta-\Delta t) \\ ... \\ V_k(x,t-\delta-(k-1)\Delta t) \end{bmatrix} + O(\Delta t^k), \tag{5.19}$$

where $D_{H_j}(j=0,...,k-1)$ are given by the formulas

$$\begin{aligned} D_{H_j} &= \int_{t-\delta}^{t+\Delta t-\delta} e^{(-\lambda_j+s_k)(t+\Delta t-\tau)} \frac{(t-\delta-\tau)^j}{\Delta t^j}\, d\tau \\ &= (k-1)^{j+1}\Delta t \begin{cases} \frac{1-e^{-q}}{q}\,, & j=0 \\ \frac{1-e^{-q}-qe^{-q}}{q^2}\,, & j=1 \\ \frac{2-2e^{-q}-2qe^{-q}-q^2e^{-q}}{q^3}\,, & j=2 \\ \frac{6-6e^{-q}-6qe^{-q}-3q^2e^{-q}-q^3e^{-q}}{q^4}\,, & j=3 \end{cases} \end{aligned} \tag{5.20}$$

where

$$q = (k-1)(\lambda_j - s_k)\Delta t.$$

The equation (5.20) suffers from catastrophic cancellation errors when q is small. Therefore, for $q < 10^{-3}$ we need to evaluate it with the Taylor expansions

$$D_{H_j} = (k-1)^{j+1}\Delta t \begin{cases} 1 - \frac{1}{2}q + \frac{1}{6}q^2 - \frac{1}{24}q^3 + \frac{1}{120}q^4\,, & j=0 \\ \frac{1}{2} - \frac{1}{3}q + \frac{1}{8}q^2 - \frac{1}{30}q^3 + \frac{1}{144}q^4\,, & j=1 \\ \frac{1}{3} - \frac{1}{4}q + \frac{1}{10}q^2 - \frac{1}{36}q^3 + \frac{1}{168}q^4\,, & j=2 \\ \frac{1}{4} - \frac{1}{5}q + \frac{1}{12}q^2 - \frac{1}{42}q^3 + \frac{1}{192}q^4\,, & j=3 \end{cases} \tag{5.21}$$

Equivalently, each history mode $H_{j,k}$ can be seen to satisfy a simple linear ODE. The point is that each history mode can be computed in $\mathcal{O}(1)$ operations at each time step for each x. Since both N_1 and N_2 are $\mathcal{O}(\log(T/\delta)) = \mathcal{O}(\log(N_T))$ with N_T the total number of time steps, the net computational cost for the evaluation of the history part at each time step is $\mathcal{O}(N_S \log N_T + N_S \log^2 N_T)$, with storage requirements of the order $\mathcal{O}(N_S \log^2 N_T)$. The computational cost for the entire simulation is

$$\mathcal{O}(N_S N_T \log N_T + N_S N_T \log^2 N_T)\,.$$

The algorithm is embarrassingly parallel in that the computation of each history mode is independent. Furthermore, the hierarchical fast algorithms used for each $V_k(x,\tau)$ are themselves amenable to parallelization, and there is already a substantial body of research and software devoted to that task for large-scale problems.

5.2.4 The Full Algorithm

In this section, we present the full algorithm in details for solving two-dimensional heat equation with stationary boundary using the sum-of-exponentials approximation.

First of all, we show the total number of local kernels required for the 4th order scheme. During the time matching in the simulation, suppose we are at $t = 4\,dt$ as in Figure 5.1. Then we are able to evaluate the Fourier mode (5.15) in the history part with previous three steps by the equation (5.17). The local part consists of one time step and $\delta = dt$ at this step. When the time proceeds to $t = 5\,dt$ as in Figure 5.2, we treat the previous two steps as the local part and $\delta = 2\,dt$. The history mode at this step is evaluated by

$$H_{j,k}(x, t+\Delta t) = e^{(-\lambda_j + s_k)\Delta t} H_{j,k}(x,t)\,. \tag{5.22}$$

Similarly at $t = 6\,dt$ as in Figure 5.3, we treat the previous three steps as the local part and $\delta = 3\,dt$ and the history mode is computed by the equation (5.22). Then at the next time step $t = 7\,dt$, we evaluate the Fourier mode by the equation (5.15) again and $\delta = dt$.

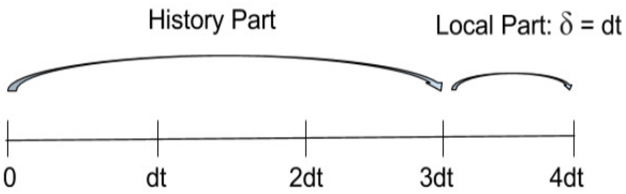

Figure 5.1 Time matching at $t = 4\,dt$.

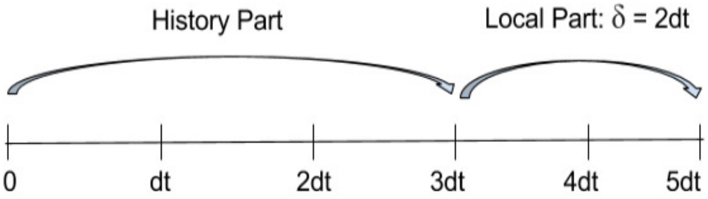

Figure 5.2 Time matching at $t = 5\,dt$.

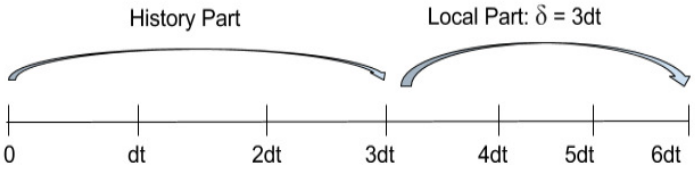

Figure 5.3 Time matching at $t = 6\,dt$.

As a consequence, we have to evaluate local kernels with different δ ($\delta = 1,\ 2,\ 3.$) in the 4th order scheme. In each δ, we need to compute four kernels as in equation (5.12). Furthermore, the 4th order simulation requires the 2nd order scheme to initialize the first few steps, where we need to evaluate two additional kernels in the 2nd order scheme. Hence, we need to evaluate 14 local kernels in the simulation of the 4th order scheme in total.

As the boundary is stationary, we can precompute all local kernels. The local kernel is split into three part D_{eq}, D_{alp}, and D_{adj} as in the equation (5.13). D_{eq} of all local kernels will be compressed using `rskelf`. Algorithm 5.1 lists the steps of evaluating the local part $D_L(\sigma)$.

Algorithm 5.1 Evaluation of the Local Part

```
Require: Suppose $\sigma$ is the density function and the local kernel $D_L$ is split into three
    parts: $D_{eq}$, $D_{alp}$, and $D_{adj}$. Evaluate $D_L(\sigma)$.
Compress $D_{eq}$ and evaluate $D_{eq}\sigma$ via rskelf.
Evaluate $D_{alp}\sigma$ via 1D NUFFT.
Evaluate $D_{adj}\sigma$ directly.
$D_L\sigma = D_{eq}\sigma + D_{alp}\sigma - D_{adj}\sigma$.
```

The history kernels are approximated by the sum-of-exponentials and the spatial integral is discretized by the trapezoidal rule. The resulting summation can be accelerated using `rskelf` and its application. The integral with respect to time can be represented by history modes (5.15) and evaluated through the recurrence relation (5.17) in O(1) operations at each time step as described in Subsection 5.2.3. Algorithm 5.2 shows how to evaluate the history part $D_H(\sigma)$.

Denote the density σ at time step j as σ_j. We give the integral equation for the interior Dirichlet problem at the time step j during the time matching for the 4th order scheme:

$$-\frac{1}{2}\sigma_j + D_L(\sigma_j) = b_j - D_H - D_L(\sigma_{j-1}) - D_L(\sigma_{j-2}) - D_L(\sigma_{j-3}). \tag{5.23}$$

Algorithm 5.2 Evaluation of the History Part for 2D Heat Solver Using SOE Approximation

```
Require: Suppose $\sigma$ is the density function. Assume that we are at $j$th step in a $k$th orde scheme. Evaluate $D_H(\sigma)$.
if $mod(j-2,\, k-1) == 0$ then
    Evaluate the spatial integral (5.16) with rskelf.
    Evaluate the recurrence relation (5.17) for the history mode.
else
    Evaluate the history mode with the equation (5.22).
end if
Compute the equation (5.14) directly and obtain $D_H(\sigma)$.
```

In Algorithm 5.3, the computation cost for the compression of local kernels and history kernels is

$$\mathcal{O}(14N_S + N_S \log N_T)\,.$$

The time complexity for solving the density σ is

$$\mathcal{O}(3N_SN_T + (c+1)N_T(N_S + 30N_S \log 30N_S + 18N_S) + N_SN_T \log N_T + N_SN_T \log^2 N_T)\,,$$

where c is the average iterations of GMRES. Evaluating the solution U costs

$$\mathcal{O}(N_T(N_S + N_{targ})(\log N_T + \log^2 N_T) + (N_S + N_{targ})(30 \log 30N_S))\,,$$

where N_{targ} is the total number of target points.

Remark 5.2. *For high-order schemes, the first few steps of σ would be initialized with smaller time step in the* 2*nd order scheme.*

Algorithm 5.3 2D Heat Solver Using SOE Approximation

```
Require: Given the total discretization points $N_S$, source points $src$, target points $targ$, total number of time steps $N_T$, time step $dt$ in a $k$th order scheme. Solve the density $\sigma$ at all time steps and evaluate the solution $U$ to the heat equation at the final time for all target points.
Separate all local kernels into three parts $D_{eq}$, $D_{alp}$, and $D_{adj}$. Compress $D_{eq}$ of all local kernels and all history kernels approximated by the SOE approximation using rskelf.
for $j = k$ to $N_T + 1$ do                                        ▷ Solving the density $\sigma$
    Compute the right hand side of the linear system $b$.
    if $j > k$ then
        Evaluate the history part $D_H$ using Algorithm 5.2.
    end if
    Evaluate the local part $D_L(\sigma)$ using Algorithm 5.1.
    Use GMRES to solve the linear system (5.23) and obtain $\sigma_j$.
end for
Evaluate the solution $U$ at the final time for all target points.
```

5.3 Efficient 2D Heat Solver with Moving Geometry Using NUFFT

5.3.1 Accurate Evaluation of the Local Part

We follow the treatment in [44] and use the double layer potential to illustrate the idea. When the boundary is function of time, i.e., $\Gamma = \Gamma(\tau)$, the kernel of the double layer potential has the following explicit expression:

$$\frac{\partial G_2(\mathbf{x}(t), \mathbf{y}(\tau); t-\tau)}{\partial \mathbf{n}_{\mathbf{y}(\tau)}} = \frac{e^{-\frac{\|\mathbf{x}(t)-\mathbf{y}(\tau)\|^2}{4(t-\tau)}}}{8\pi(t-\tau)^2}[(\mathbf{x}(t)-\mathbf{y}(\tau))\cdot \mathbf{n}_{\mathbf{y}(\tau)}]. \tag{5.24}$$

The local part of the double layer potential is given by the formula

$$\begin{aligned} D_L[\sigma](\mathbf{x}, t) &= \int_{t-\delta}^{t} \int_{\Gamma(\tau)} \frac{\partial G_2(\mathbf{x}(t), \mathbf{y}(\tau); t-\tau)}{\partial \mathbf{n}_{\mathbf{y}(\tau)}} \sigma(\mathbf{y}(\tau), \tau) ds_{\mathbf{y}(\tau)} d\tau \\ &= \int_{t-\delta}^{t} \int_{\Gamma(\tau)} \frac{e^{-\frac{\|\mathbf{x}(t)-\mathbf{y}(\tau)\|^2}{4(t-\tau)}}}{8\pi(t-\tau)^2}[(\mathbf{x}(t)-\mathbf{y}(\tau))\cdot \mathbf{n}_{\mathbf{y}(\tau)}]\sigma(\mathbf{y}(\tau), \tau) ds_{\mathbf{y}(\tau)} d\tau, \end{aligned} \tag{5.25}$$

where one has to carry out the integration in space first since the boundary is now a function of time as well. We now switch the order of integration by using the boundary at the current time $\Gamma(t)$ for spatial integration. That is,

$$\begin{aligned} D_L[\sigma](\mathbf{x}, t) &= \int_{t-\delta}^{t} \int_{\Gamma(\tau)} \frac{\partial G_2(\mathbf{x}(t), \mathbf{y}(\tau); t-\tau)}{\partial \mathbf{n}_{\mathbf{y}(\tau)}} \sigma(\mathbf{y}(\tau), \tau) ds_{\mathbf{y}(\tau)} d\tau \\ &= \int_{\Gamma(t)} \int_{t-\delta}^{t} \frac{\partial G_2(\mathbf{x}(t), \mathbf{y}(\tau); t-\tau)}{\partial \mathbf{n}_{\mathbf{y}(\tau)}} \sigma(\mathbf{y}(\tau), \tau) \frac{ds_{\mathbf{y}(\tau)}}{ds_{\mathbf{y}(t)}} d\tau ds_{\mathbf{y}(t)} \\ &= \int_{\Gamma(t)} \int_{t-\delta}^{t} \frac{e^{-\frac{\|\mathbf{x}(t)-\mathbf{y}(\tau)\|^2}{4(t-\tau)}}}{8\pi(t-\tau)^2}[(\mathbf{x}(t)-\mathbf{y}(\tau))\cdot \mathbf{n}_{\mathbf{y}(\tau)}]\sigma(\mathbf{y}(\tau), \tau) \frac{ds_{\mathbf{y}(\tau)}}{ds_{\mathbf{y}(t)}} d\tau ds_{\mathbf{y}(t)}. \end{aligned} \tag{5.26}$$

In order to carry out the product integration in time as in the stationary case, we decompose the exponential function as follows:

$$e^{-\frac{\|\mathbf{x}(t)-\mathbf{y}(\tau)\|^2}{4(t-\tau)}} = e^{-\frac{\|\mathbf{x}(t)-\mathbf{y}(t)\|^2}{4(t-\tau)}} \cdot e^{-\frac{\|\mathbf{y}(t)-\mathbf{y}(\tau)\|^2}{4(t-\tau)}} \cdot e^{-\frac{(\mathbf{x}(t)-\mathbf{y}(t))\cdot(\mathbf{y}(t)-\mathbf{y}(\tau))}{2(t-\tau)}}. \tag{5.27}$$

The first term on the right hand side of (5.27) is the same as the stationary case, while the second and third terms both contain the factor $\frac{(\mathbf{y}(t)-\mathbf{y}(\tau))}{(t-\tau)}$, which is a smooth function of τ when the boundary undergoes a smooth motion. Let

$$\mu(\tau) = e^{-\frac{\|\mathbf{y}(t)-\mathbf{y}(\tau)\|^2}{4(t-\tau)}} \cdot e^{-\frac{(\mathbf{x}(t)-\mathbf{y}(t))\cdot(\mathbf{y}(t)-\mathbf{y}(\tau))}{2(t-\tau)}} [(\mathbf{x}(t)-\mathbf{y}(\tau))\cdot \mathbf{n}_{\mathbf{y}(\tau)}]\sigma(\mathbf{y}(\tau),\tau)\frac{ds_{\mathbf{y}(\tau)}}{ds_{\mathbf{y}(t)}}. \quad (5.28)$$

Then (5.26) can be written as follows:

$$D_L[\sigma](\mathbf{x},t) = \int_{\Gamma(t)}\int_{t-\delta}^{t} \frac{e^{-\frac{\|\mathbf{x}(t)-\mathbf{y}(t)\|^2}{4(t-\tau)}}}{8\pi(t-\tau)^2}\mu(\tau)d\tau ds_{\mathbf{y}(t)}. \quad (5.29)$$

We may now proceed as in the stationary case. That is, approximate $\mu(\tau)$ by a polynomial of τ, convert the integration in time to a sum of products of local kernels D_i and $\mu(t-i\Delta t)$, and then use proper quadrature to discretize the spatial integrals.

Remark 5.3. *One will need the value of $\mu(t)$, which may be obtained by replacing $\frac{(\mathbf{y}(t)-\mathbf{y}(\tau))}{(t-\tau)}$ with its limiting value $\mathbf{y}'(t)$ as $\tau \to t$.*

5.3.2 Evaluation of the History Part

For the history part, we use the spectral Fourier approximation of the heat kernel. Unlike [28] in which a tensor product is applied to obtain the spectral Fourier approximation of the heat kernel in higher dimensions, we first write the Fourier representation of $G_2(\mathbf{x},t)$ in polar coordinates:

$$\begin{aligned} G_2(\mathbf{x},t) &= \frac{1}{4\pi^2}\int_{-\infty}^{\infty}\int_{-\infty}^{\infty} e^{-(k_1^2+k_2^2)t}e^{i(k_1x_1+k_2x_2)}dk_1dk_2 \\ &= \frac{1}{4\pi^2}\int_0^{2\pi}\int_0^{\infty} e^{-k^2t}e^{ik(x_1\cos(\theta)+x_2\sin(\theta))}kdkd\theta. \end{aligned} \quad (5.30)$$

We then use the generalized Gaussian quadrature to construct an optimal quadrature along the radial direction, and the trapezoidal rule discretize the integration along the azimuthal direction which achieves spectral accuracy for smooth periodic integrals. Altogether, we need 21600 Fourier modes to achieve 13-digit accuracy for $t \geq \delta = 10^3$ and $\|\mathbf{x}\| \leq R = 1$. If the tensor product in [28] were used, one would need about 700 for each direction in the Fourier domain and 700^2 (i.e., about half million) Fourier nodes to approximate G_2 in the same region. In any case, we have the following spectral Fourier representation for G_2:

$$G_2(\mathbf{x}, t) \approx \frac{1}{4\pi^2} \sum_{j=1}^{N_F} w_j e^{-\|\boldsymbol{\xi}_j\|^2 t} e^{i\boldsymbol{\xi}_j \cdot \mathbf{x}}. \tag{5.31}$$

The Fourier representation of the double layer kernel can be obtained by replacing t, $\mathbf{x}$ with $t-\tau$, $\mathbf{x}-\mathbf{y}$, respectively, and then differentiating the resulting expression with respect to $\mathbf{y}$. We have

$$\frac{\partial G_2(\mathbf{x}(t), \mathbf{y}(\tau); t-\tau)}{\partial \mathbf{n}_{\mathbf{y}(\tau)}} \approx -\frac{i}{4\pi^2} \sum_{j=1}^{N_F} w_j e^{-\|\boldsymbol{\xi}_j\|^2 (t-\tau)} e^{i\boldsymbol{\xi}_j \cdot (\mathbf{x}-\mathbf{y})} \left(\boldsymbol{\xi}_j \cdot \mathbf{n}_{\mathbf{y}(\tau)}\right). \tag{5.32}$$

Substituting the above approximation into the history part of the double layer potential, we obtain

$$\begin{aligned} D_H[\sigma](\mathbf{x}, t) &= \int_0^{t-\delta} \int_{\Gamma(\tau)} \frac{\partial G_2(\mathbf{x}(t), \mathbf{y}(\tau); t-\tau)}{\partial \mathbf{n}_{\mathbf{y}(\tau)}} \sigma(\mathbf{y}(\tau), \tau) ds_{\mathbf{y}(\tau)} d\tau \\ &\approx -\frac{i}{4\pi^2} \sum_{j=1}^{N_F} \int_0^{t-\delta} \int_{\Gamma(\tau)} w_j e^{-\|\boldsymbol{\xi}_j\|^2 (t-\tau)} e^{i\boldsymbol{\xi}_j \cdot (\mathbf{x}-\mathbf{y})} \left(\boldsymbol{\xi}_j \cdot \mathbf{n}_{\mathbf{y}(\tau)}\right) \sigma(\mathbf{y}(\tau), \tau) ds_{\mathbf{y}(\tau)} d\tau \\ &= -\frac{i}{4\pi^2} \sum_{j=1}^{N_F} w_j e^{i\boldsymbol{\xi}_j \cdot \mathbf{x}(t)} \int_0^{t-\delta} e^{-\|\boldsymbol{\xi}_j\|^2 (t-\tau)} H_j(\tau) d\tau, \end{aligned} \tag{5.33}$$

where H_j is defined by the formula

$$H_j(\tau) = \int_{\Gamma(\tau)} e^{-i\boldsymbol{\xi}_j \cdot \mathbf{y}(\tau)} \left(\boldsymbol{\xi}_j \cdot \mathbf{n}_{\mathbf{y}(\tau)}\right) \sigma(\mathbf{y}(\tau), \tau) ds_{\mathbf{y}(\tau)}, \quad j = 1, \ldots, N_F. \tag{5.34}$$

The integrals in (5.34) can be discretized via the trapezoidal rule with spectral accuracy. After that, all H_j ($j = 1, \ldots, N_F$) can be evaluated via type-3 NUFFT with $O((N_S + N_F)\log(N_S + N_F))$ cost. The temporal integral can be evaluated via standard recurrence relation as in the stationary case with $O(N_F)$ cost for each time step. Finally, we may apply type-3 NUFFT again to evaluate the summation in (5.33) with $O((N_S + N_F)\log(N_S + N_F))$ cost. The algorithm is spectrally accurate and has $O(N_T(N_S + N_F)\log(N_S + N_F))$ complexity for the whole simulation.

5.3.3 The Full Algorithm

The algorithm is very similar to Algorithm 5.3 in Section 5.2 except that the history part is approximated by the spectral Fourier approximation and accelerated using NUFFT, and the local kernels need to be compressed at every time step. Local kernels are still split into three parts D_{eq}, D_{alp}, D_{adj} and they will be applied the density σ separately during the time matching as in Algorithm 5.3.

Algorithm 5.4 2D Heat Solver Using Spectral Fourier Approximation

```
Require: Given the total discretization points $N_S$, source points $src$, target points
    $targ$, total number of time steps $N_T$, time step $dt$ in a $k$th order scheme. Solve
    the density $\sigma$ at all time steps and evaluate the solution $U$ to the heat equation
    at the final time for all target points.
for $j = k$ to $N_T + 1$ do                                    ▷ Solving the density $\sigma$
    Compute the right hand side of the linear system $b$.
    if $j > k$ then
        Evaluate the history part $D_H$ using NUFFT as in Subsection 5.3.2.
    end if
    Split the local kernels described in Subsection 5.3.1 into three parts
    $D_{eq}$, $D_{alp}$, $D_{adj}$. Compress $D_{eq}$ of all local kernels using rskelf.
    Evaluate the local part $D_L(\sigma)$ using Algorithm 5.1.
    Use GMRES to solve the linear system (5.23) and obtain $\sigma_j$.
end for
Evaluate the solution $U$ at the final time for all target points.
```

In Algorithm 5.4, the computation cost for solving the density σ is

$$\mathcal{O}(4N_S N_T + (c+1)N_T(4N_S + 30N_S \log 30N_S + 18N_S) + N_T(N_S + N_F)\log(N_S + N_F)),$$

where c is the average iterations of GMRES. The time complexity for evaluating the solution U is

$$\mathcal{O}(N_T(N_S+N_F+N_{targ})\log(N_S+N_F+N_{targ})+N_{targ}+N_S+30N_S\log 30N_S+18N_S)\,,$$

where N_{targ} is the total number of target points.

5.4 High-order 3D Heat Solver with Stationary Geometry Using SOE Approximations

5.4.1 High-order Discretization of the Singular Integrals in the Local Part

For three dimensions, the local kernels in (5.11) become

$$D_{L_k} = \begin{cases} \frac{(x-y)\cdot n_y}{4\pi^{\frac{3}{2}}\delta^{\frac{1}{2}}|x-y|^2}e^{-\lambda^2} + \frac{(x-y)\cdot n_y}{4\pi|x-y|^3}[1-erf(\lambda)]\,, & k=0, \\ \frac{1}{\Delta t}\frac{(x-y)\cdot n_y}{8\pi|x-y|}[1-erf(\lambda)]\,, & k=1, \\ \frac{1}{\Delta t^2}\Big\{\frac{(x-y)\cdot n_y\sqrt{\delta}}{8\pi^{\frac{3}{2}}}e^{-\lambda^2} - \frac{(x-y)\cdot n_y|x-y|}{16\pi}[1-erf(\lambda)]\Big\}\,, & k=2, \\ \frac{1}{\Delta t^3}\Big\{\frac{(x-y)\cdot n_y\delta^{\frac{3}{2}}}{24\pi^{\frac{3}{2}}}e^{-\lambda^2} - \frac{(x-y)\cdot n_y\delta^{\frac{1}{2}}|x-y|^2)}{48\pi^{\frac{3}{2}}}e^{-\lambda^2} + & k=3, \\ \frac{(x-y)\cdot n_y|x-y|^3)}{96\pi}e^{-\lambda^2}[1-erf(\lambda)]\Big\} & \end{cases} \tag{5.35}$$

where $\lambda = \frac{|x-y|}{2\sqrt{\delta}}$ and $erf(x)$ is the error function given by

$$erf(x) = \frac{2}{\sqrt{\pi}}\int_0^x e^{-t^2}\,dt.$$

The error function $erf(x)$ is an entire function and the Taylor expansion of $erf(x)$ is

$$erf(x) = \frac{2}{\pi}\sum_{n=0}^{\infty}\frac{(-1)^n x^{2n+1}}{n!(2n+1)} \quad = \frac{2}{\pi}(x - \frac{x^3}{3} + \frac{x^5}{10} - \frac{x^7}{42} + \cdots)\,. \tag{5.36}$$

In addition, it is well known that

$$(x-y)\cdot n_y = O(|x-y|^2)\,. \tag{5.37}$$

From equations (5.36) and (5.37), we can see the local kernels have the $\frac{1}{r}$ singularity along the diagonal when $x = y$ and the local part cannot be evaluated using regular quadrature for discretizing the boundary surface.

Without loss of generality, the spatial integral of the local kernel can be expressed as

$$\int_\Gamma \frac{1}{|x-y|} f(y) + g(y) ds_y \,, \tag{5.38}$$

where $f(y)$ and $g(y)$ are smooth over the boundary surface. Suppose the boundary is an unit sphere and is parametrized by spherical coordinates (θ, ϕ). Assume that $x = (0,0,1)$ locates at the north pole of the spherical coordinates and $y = (\sin\theta\cos\phi, \sin\theta\sin\phi, \cos\theta)$. Then

$$|x-y| = \sqrt{(1-\cos\theta)^2 + \sin^2\theta} = \sqrt{2(1-\cos\theta)} \tag{5.39}$$

and

$$\begin{aligned} \int_\Gamma \frac{1}{|x-y|} f(y) ds_y &= \int_0^{2\pi} \int_0^\pi \frac{1}{\sqrt{2(1-\cos\theta)}} f(\theta,\phi) \sin\theta d\theta d\phi \\ &= \int_0^{2\pi} \int_{-1}^1 \frac{1}{\sqrt{2(1-z)}} f(\cos^{-1} z, \phi) dz d\phi \,, \end{aligned} \tag{5.40}$$

where $z = \cos\theta$. Similarly, we get

$$\int_\Gamma g(y) ds_y = \int_0^{2\pi} \int_{-1}^1 g(\cos^{-1} z, \phi) dz d\phi \,. \tag{5.41}$$

Since ϕ can be discretized accurately with equispaced nodes $\{\phi_k\}$ in $(0, 2\pi)$ with

$$\phi_k = \frac{2\pi k}{2p+2} \,, \tag{5.42}$$

we would focus on the integral with respect to z and simply the equation (5.38) as

$$\int_{-1}^1 \frac{1}{\sqrt{1-z}} p^{(1)}(z) + p^{(2)}(z) dz \,, \tag{5.43}$$

where $p^{(1)}(z)$ and $p^{(2)}(z)$ are both polynomials. Let $s = \sqrt{\frac{(1-z)}{2}}$, the above equation (5.43) becomes

$$\int_0^1 q^{(1)}(s) + sq^{(2)}(s)ds\,, \tag{5.44}$$

where $q^{(1)}(s)$ and $q^{(2)}(s)$ are both polynomials. Note that the integral (5.44) does not have singularities and can be discretized using the classical Gauss-Legendre quadrature. Thus, the quadrature for θ in the original spatial integral (5.38) can be obtained by mapping back the variables with the quadrature nodes and weights for the integral (5.44). For the boundary surface that is smooth and topologically equivalent to a sphere, we can apply the similar techniques as in [25]. Furthermore, if the target point does not lie on the north pole, we may rotate the quadrature grid using the algorithm in [24]. The quadrature is spectrally accurate and the computation cost for the local kernels at each time step is $\mathcal{O}(p^4 \log p)$, which leads to $\mathcal{O}(N_T p^4 \log p)$ work for the entire simulation.

Remark 5.4. *The above algorithm is suitable for the case where there are many small objects and each small object may be discretized by a small number* p *so that the overall cost is* $O(N_{obj} p^4 \log p)$ *as in, say, [64]. When there are few objects which need large* p *for each object, one may switch to other high-order discretization schemes that are more amenable to fast algorithms. For instance, the FMM accelerated QBX (Quadrature by Expansion) scheme in [41] can be used to discretize the spatial integral and accelerate the evaluation of the resulting discrete summation. The algorithm will be high-order and of optimal complexity.*

5.4.2 Evaluation of the History Part

The heat kernels are approximated again by sum-of-exponentials. As discussed in the 2D case, even though each history kernel is not smooth, what we need is the whole history part whose kernel is actually smooth. Thus we may use standard quadratures for smooth integrals to discretize each spatial integral in the history part. There is no need to use the expensive integrator as in the local part. The total computational cost is $O(N_T N_S N_H)$, where N_T is the total number of time steps, N_S is the total number of discretization points in space, and N_H is the number of history modes. The whole algorithm is very similar to the 2D algorithm and thus we omit the details here.

5.4.3 The Full Algorithm

The algorithm is similar to Algorithm 5.3 where we approximate the history part by sum-of-exponentials and evaluate it using the high-order quadrature. However, the local part is treated very differently since we need to rotate the spherical grid for targeting points that do not locate at the north pole to evaluate the local part with the quadrature described in Subsection 5.4.1.

Algorithm 5.5 High-order 3D Heat Solver Using SOE

```
Require: Given the degree of the spherical harmonic expansion $p$, source points $src$,
    target points $targ$, total number of time steps $N_T$, time step $dt$ in a $k$th order
    scheme. Solve the density $\sigma$ at all time steps and evaluate the solution $U$ to the
    heat equation at the final time for all target points.
for $j = k$ to $N_T + 1$ do                                    ▷ Solving the density $\sigma$
    Compute the right hand side of the linear system $b$.
    if $j > k$ then
        Evaluate the history part $D_H$ similar to Algorithm 5.2.
    end if
    Evaluate the local part $D_L(\sigma)$ as in Subsection 5.4.1.
    Use GMRES to solve the linear system (5.23) and obtain $\sigma_j$.
end for
Evaluate the solution $U$ at the final time for all target points.
```

In Algorithm 5.5, the time complexity for solving the density σ is

$$\mathcal{O}(N_T N_{obj} p^4 \log p + N_T N_S N_H)\,,$$

where N_{obj} is the total number of objects, $O(p^2)$ is the number of discretization points for the boundary of each object, $N_S = N_{obj}p^2$ is the number of discretization points in space, N_T is the total number of time steps.

CHAPTER 6

NUMERICAL RESULTS

In this chapter, we illustrate the performance of the three algorithms outlined in the preceding chapter via extensive numerical examples. We have implemented 2nd, 3rd, and 4th order schemes in time (defined by the number of terms taken in the local Taylor expansion of σ in Chapter 5), with numerical experiments carried out using the fourth order version with OpenMP.

The following notations are used in the examples shown subsequently:

- Δt: time step;
- N_T: total number of time steps;
- N_S: total number of discretization points in space;
- N_F: total number of Fourier modes;
- N_θ: total number of discretization points in the θ direction;
- N_ϕ: total number of discretization points in the ϕ direction;
- E: relative L^2 error of the numerical solution at the final time;
- $\bar{T}_f^L$: average factorization time of all local kernels in seconds;
- $\bar{T}_a^L$: average application time of all compressed local kernels in seconds;
- $\bar{T}_D^L$: average direct computation time of all local kernels in seconds;
- $\bar{T}_F^L$: average computation time with fast algorithms of all local kernels in seconds;
- $\bar{T}_f^H$: average factorization time of all history kernels in seconds;

- $\bar{T}_a^H$: average application time of all compressed history kernels in seconds;
- $\bar{T}_D^H$: average direct computation time of all history kernels in seconds;
- $\bar{T}_F^H$: average computation time with fast algorithms of all history kernels in seconds;
- T_D^H: total evaluation time for the history part directly in seconds;
- T: total simulation time in seconds;
- T_m: total matching time in seconds;
- T_f: total factorization time in seconds.
- T_{fft}^H: total evaluation time for the history part using NUFFT in seconds.

It is worth noting that direct computations in the examples below involve constructing $N_S \times N_S$ matrices explicitly, which is extremely slow during the simulations when N_S is very large. The available computer memory might also be insufficient for very large N_S. Therefore, we only build $20 \times N_S$ matrices and record the time t_o. And the direct computation time for $N_S \times N_S$ matrices is estimated by $t_o * N_S/20$.

6.1 2D Heat Solver Using SOE Approximations

Example 1: Exterior Dirichlet Problem with the Boundary Consisting of a 32-gram.

We consider the exterior Dirichlet problem with the boundary consisting of a 32-gram shown in Figure 6.1. Here the boundary curve is roughly of size $R = 4$ and the center locates at $(0, 0)$.

We generate boundary data by placing heat sources *inside* the boundary and test the accuracy of our numerical solution by comparing it with the analytical solution at 20 target points on a circle of radius 6.

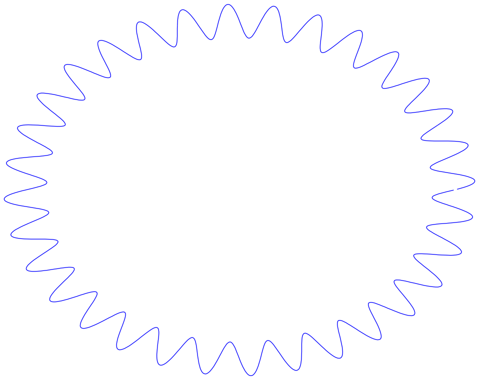

Figure 6.1 32-gram boundary curve for Example 1.

We first check the spatial accuracy of our algorithm. The table on the left panel of Figure 6.2 lists the relative L^2 error versus the total number of discretization points N_S in space; and the figure on the right panel plots out these data points. Here $N_T = 80$ and $dt = 0.0125$ are fixed throughout the simulation and the error due to temporal discretization is negligible. We observe that the algorithm has very high order convergence rate in space.

N_S	125	250	500	1000
E	3.37e-2	1.35e-4	4.02e-6	2.48e-9

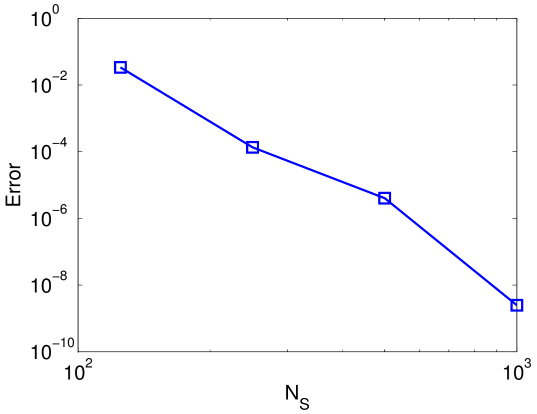

Figure 6.2 Relative L^2 errors versus N_S for Example 1. Here $N_T = 80$ and $dt = 0.0125$ are fixed. The figure on the right uses logarithmic scale for both axes.

Next, we check the convergence order in time. The table on the left panel of Figure 6.3 lists the time step dt, the relative L^2 error E, and the ratio r of the relative errors of two consecutive runs. For all these simulations, N_S is set to 6250 so that the error due to spatial discretization is negligible. The figure on the right panel plots out E versus dt and a curve with the ideal fourth order scheme. We observe the convergence rate is roughly consistent with the fourth order accuracy (Slightly better because of the smoothing behavior of the heat equation).

dt	E	r
2.00e-1	1.07e-2	
1.00e-1	6.65e-4	16.1
5.00e-2	1.76e-5	37.5
2.50e-2	2.73e-7	64.7
1.25e-2	2.47e-9	110

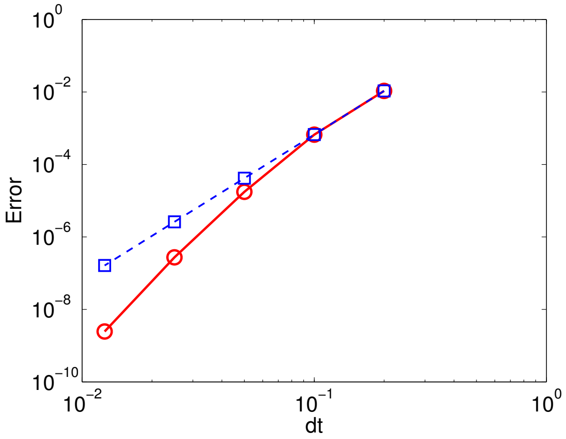

Figure 6.3 Convergence order in time for Example 1. The figure on the right is log-log plot for dt and E. The red solid line is the numerical results and the blue dashed line represents the relation of dt and E in an ideal 4th order scheme.

We now check the complexity of our algorithm with respect to the spatial variable. Here we fix $N_T = 80$ and $dt = 0.0125$. In Table 6.1, the first column lists the total number of discretization points in space; the second column lists the average factorization time for the local kernels; the third column lists the average apply time for the local kernels; the fourth column lists the average time for the matrix-vector product of local kernels by direct computation; the fifth, sixth, and seventh columns list the corresponding values for the history kernels; the eighth column lists the total computational time of our algorithm; and the last column lists the ratio of T of two consecutive runs.

Table 6.1 Timing Results with Respect to the Spatial Variable

N_S	$\bar{T}_f^L$	$\bar{T}_a^L$	$\bar{T}_D^L$	$\bar{T}_f^H$	$\bar{T}_a^H$	$\bar{T}_D^H$	T	r
6250	2.10	4.11e-3	1.32	3.36	1.15e-2	1.54	67.0	
12500	3.05	6.73e-3	5.31	4.03	1.69e-2	6.19	120	1.79
25000	4.32	1.61e-2	23.1	4.78	2.73e-2	24.7	208	1.73
50000	6.55	3.69e-2	106	5.95	6.92e-2	144	390	1.87
100000	10.8	8.29e-2	558	8.11	1.36e-1	581	804	2.06

The figure on the left panel of Figure 6.4 shows the total computational time T versus N_S, while that on the right panel shows the factorization time and direct computational time for both local and history kernels.

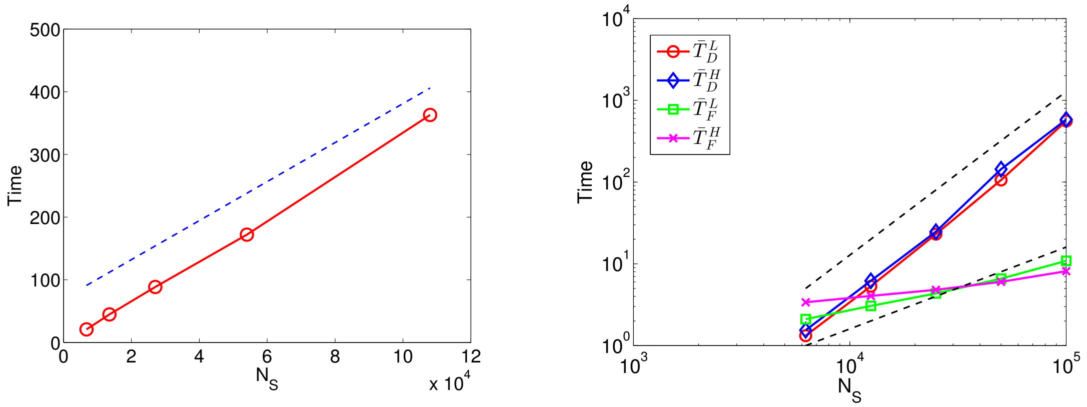

Figure 6.4 Timing results with respect to the spatial variable for Example 1. In the left figure, the red solid line is numerical results of N_S and T. The blue dashed line represents $O(N_S)$ scaling. The right figure shows the log-log plot of N_S and $\bar{T}_D^L$, $\bar{T}_D^H$, $\bar{T}_F^L$ and $\bar{T}_F^H$, including reference scalings (black dashed lines) of $O(N^2)$ and $O(N)$ (from top to bottom).

Both Table 6.1 and Figure 6.4 demonstrate clearly that while the direct calculation clearly exhibits quadratic complexity, our algorithm has linear complexity.

Finally, we examine the complexity of our algorithm with respect to the temporal variable. The table on the left panel of Figure 6.5 lists the time step size

dt, the total number of time steps N_T, and the total matching time T_m with fixed N_S, while the figure on the right panel shows the total factorization time, the total marching time versus dt. Since the geometry is stationary, the factorization needs to be carried out once and its cost is thus independent of N_T. On the other hand, we observe that the matching time is almost linear with respect to time steps.

dt	N_T	T_m
1.00e-1	16	9.72
5.00e-2	26	14.6
2.50e-2	46	24.5
1.25e-2	86	43.8
6.25e-3	166	82.4

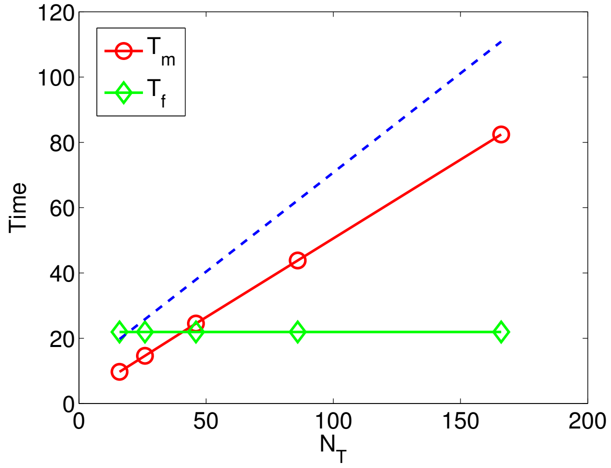

Figure 6.5 Timing results with respect to the temporal variable for Example 1. $N_S = 6250$ in all simulations. The total factorization time $T_f = 21.9$ is the same for all different time steps. In the right figure, the red line represents the total matching time; the green line represents the total factorization time; the blue dashed line represents $O(N)$ scaling.

Example 2: Exterior Dirichlet Problem with the Boundary Consisting of Nine Circles.

We consider the exterior Dirichlet problem with the boundary consisting of nine circles shown in Figure 6.6. Here the radius of each circle is 2 and the centers of the circles locate at

$$(-6,6),\ (0,6),\ (6,6),\ (-6,0),\ (0,0),\ (6,0),\ (-6,-6),\ (0,-6),\ (6,-6).$$

We generate boundary data by placing heat sources *inside* the nine circles and test the accuracy of our numerical solution by comparing it with the analytical solution

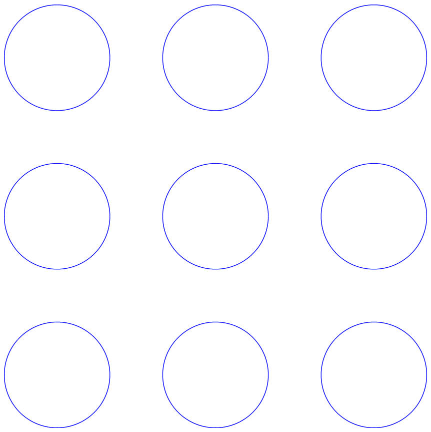

Figure 6.6 Nine-circle boundary curve for Example 2.

at 20 target points on a circle of radius 10. Moreover, refer to Example 1 for all notations and explanations for the figures and tables below.

From Figure 6.7, we observe that the algorithm has very high order convergence rate in space for Example 2. Figure 6.8 shows the convergence order is very close to the theoretical value for Example 2. Both Table 6.2 and Figure 6.10 demonstrate clearly that while the direct calculation clearly exhibits quadratic complexity, our algorithm has linear complexity for Example 2. Finally, from Figure 6.10, we observe that the matching time is almost linear with respect to time steps for Example 2.

Table 6.2 Timing Results with Respect to the Spatial Variable

N_S	$\bar{T}_f^L$	$\bar{T}_a^L$	$\bar{T}_D^L$	$\bar{T}_f^H$	$\bar{T}_a^H$	$\bar{T}_D^H$	T	r
6750	0.69	2.63e-3	1.37	1.01	4.67e-3	1.96	21.0	
13500	1.31	6.48e-3	5.32	1.36	8.37e-3	8.24	44.8	2.03
27000	2.66	2.05e-2	26.8	1.97	1.71e-2	31.2	88.6	1.97
54000	5.04	2.75e-2	99.5	3.11	3.03e-2	140	174	1.96
108000	9.71	5.05e-2	433	5.33	6.11e-2	544	363	2.08

N_S	576	1152	2304	4608
E	1.20e-1	2.95e-3	1.63e-5	6.59e-9

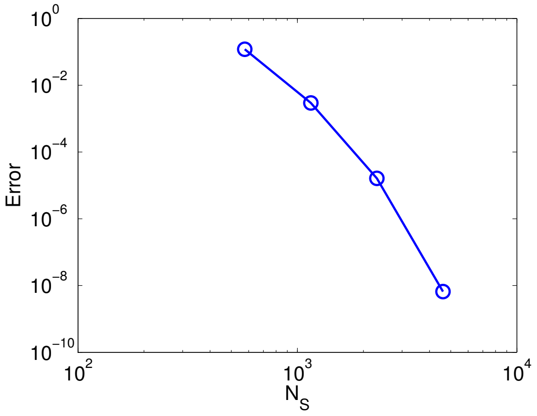

Figure 6.7 Relative L^2 errors versus N_S for Example 2. Here $N_T = 80$ and $dt = 0.0125$ are fixed. The figure on the right uses logarithmic scale for both axes.

dt	E	r
2.00e-1	7.13e-4	
1.00e-1	1.72e-4	4.13
5.00e-2	1.13e-5	15.1
2.50e-2	3.11e-7	36.5
1.25e-2	8.86e-9	35.1

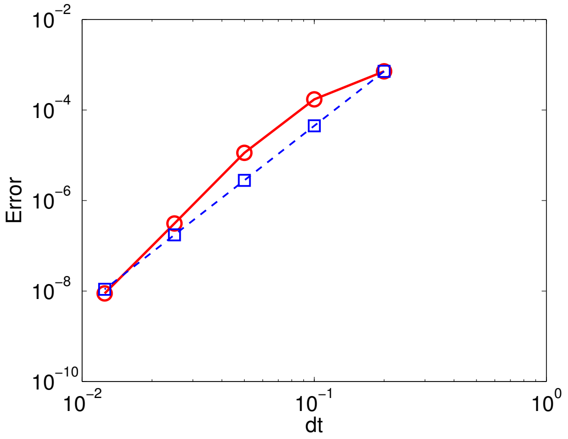

Figure 6.8 Convergence order in time for Example 2. The figure on the right is log-log plot for dt and E. $N_S = 54000$ is fixed. The red solid line is the numerical results and the blue dashed line represents the relation of dt and E in an ideal 4th order scheme.

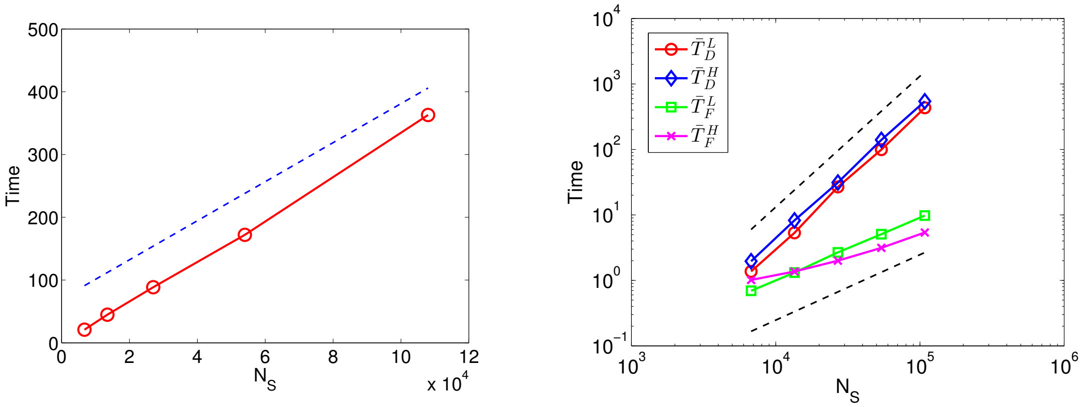

Figure 6.9 Timing results with respect to the spatial variable for Example 2. In the left figure, the red solid line is numerical results of N_S and T. The blue dashed line represents $O(N_S)$ scaling. The right figure shows the log-log plot of N_S and $\bar{T}_D^L$, $\bar{T}_D^H$, $\bar{T}_F^L$ and $\bar{T}_F^H$, including reference scalings (black dashed lines) of $O(N^2)$ and $O(N)$ (from top to bottom).

dt	N_T	T_m
1.00e-1	16	28.1
5.00e-2	26	44.6
2.50e-2	46	73.1
1.25e-2	86	137
6.25e-3	166	279

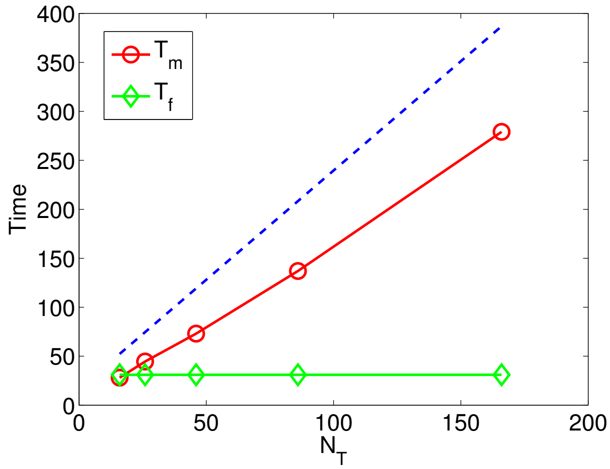

Figure 6.10 Timing results with respect to the temporal variable for Example 2. $N_S = 54000$ in all simulations. The total factorization time $T_f = 31$ is the same for all different time steps. In the right figure, the red line represents the total matching time; the green line represents the total factorization time; the blue dashed line represents $O(N)$ scaling.

Example 3: Interior Dirichlet Problem with the Boundary Enclosed by One Outer Circle and Sixteen Inner Hexagrams.

We consider the interior Dirichlet problem with the boundary enclosed by one outer circle and sixteen inner hexagrams shown in Figure 6.11. Here the radius of the outer circle is 5.5 and the center of the circle locates at $(0, 0)$. Each hexagram is roughly of the size $R = 0.5$. The centers of the hexagrams are

$$(-3, 3),\ (-1.5, 3),\ (1.5, 3),\ (3, 3),\ (-3, 1.5),\ (-1.5, 1.5),\ (1.5, 1.5),\ (3, 1.5),$$

$$(-3, -1.5),\ (-1.5, -1.5),\ (1.5, -1.5),\ (3, -1.5),\ (-3, -3),\ (-1.5, -3),\ (1.5, -3),\ (3, -3).$$

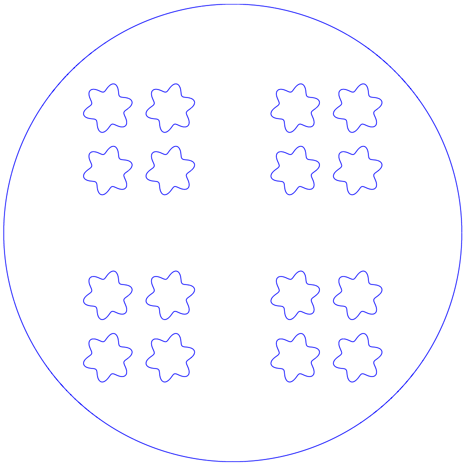

Figure 6.11 Boundary enclosed by one circle and sixteen hexagrams for Example 3.

We generate boundary data by placing heat sources *outside* the boundary region on a circle of radius 7.5 and test the accuracy of our numerical solution by comparing it with the analytical solution at 30 target points. The first ten target points are on a circle of radius 0.2 with center $(0, 4)$. The second ten target points are on a circle of radius 0.2 with center $(0, 0)$. The last ten target points are on a circle of radius 0.2

with center $(0, -4)$. Finally, refer to Example 1 for all notations and explanations for the figures and tables below.

From Figure 6.12, we observe that the algorithm has very high order convergence rate in space for Example 3. Figure 6.13 shows that although the ratio r fluctuates somewhat widely, the overall convergence order is actually close to the theoretical value (Slightly better because of the smoothing behavior of the heat equation) for Example 3. Both Table 6.3 and Figure 6.14 demonstrate clearly that while the direct calculation clearly exhibits quadratic complexity, our algorithm has linear complexity. Finally, from Figure 6.15, we observe that the matching time is almost linear with respect to time steps for Example 3.

N_S	850	1700	2550	3400
E	4.52e-3	1.28e-5	6.96e-7	7.87e-9

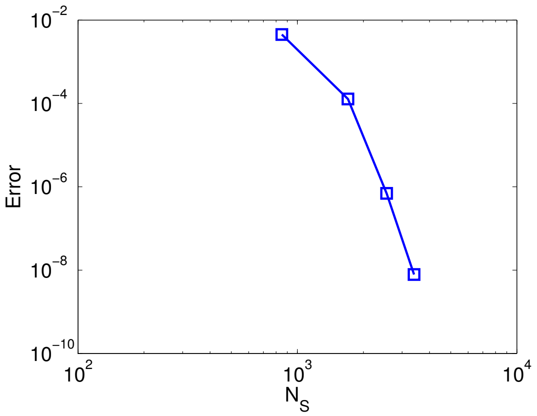

Figure 6.12 Relative L^2 errors versus N_S for Example 3. Here $N_T = 80$ and $dt = 0.0125$ are fixed. The figure on the right uses logarithmic scale for both axes.

Example 4: Interior Dirichlet Problem with the Boundary Enclosed by One Outer Circle and Sixty-four Inner Ellipses.

We consider the interior Dirichlet problem with the boundary enclosed by one outer circle and sixty-four inner ellipses shown in Figure 6.16. Here the radius of the circle is 6.5 and the center of the circle locates at $(0, 0)$. The major axis a and minor axis b of each ellipse are 0.3 and 0.2, respectively. The centers (c_1, c_2) of the ellipses are all permutations chosen from $\pm 4, \pm 3, \pm 2, \pm 1$.

dt	E	r
2.00e-1	6.93e-3	
1.00e-1	4.69e-4	14.7
5.00e-2	4.11e-5	11.4
2.50e-2	1.49e-7	274
1.25e-2	6.52e-9	22.9

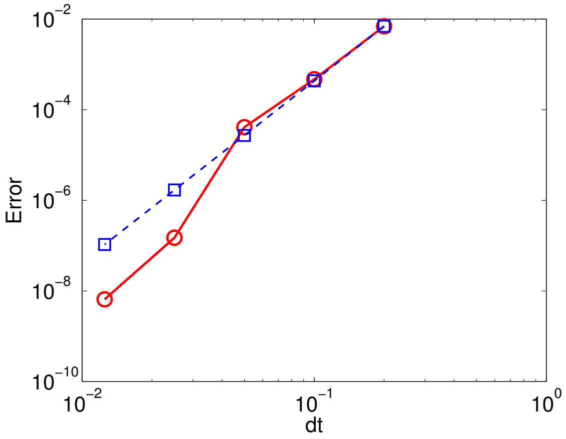

Figure 6.13 Convergence order in time for Example 3. The figure on the right is log-log plot for dt and E. $N_S = 12750$ is fixed. The red solid line is the numerical results and the blue dashed line represents the relation of dt and E in an ideal 4th order scheme.

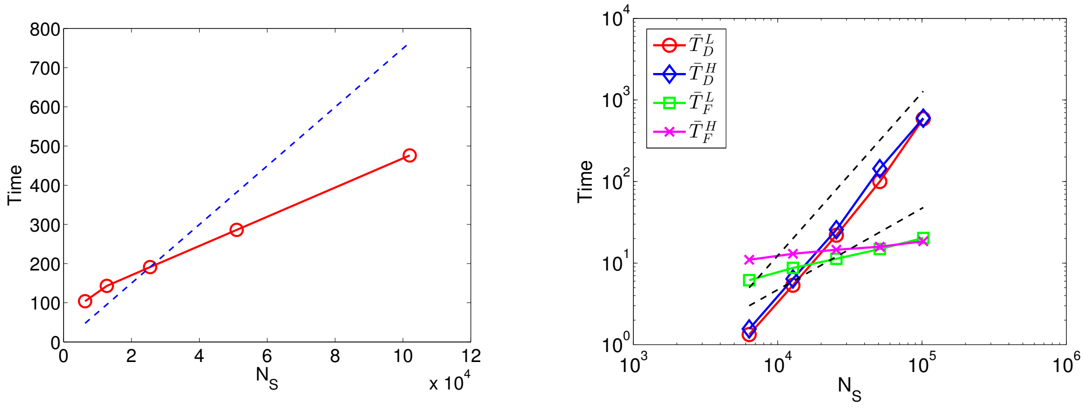

Figure 6.14 Timing results with respect to the spatial variable for Example 3. In the left figure, the red solid line is numerical results of N_S and T. The blue dashed line represents $O(N_S)$ scaling. The right figure shows the log-log plot of N_S and $\bar{T}_D^L$, $\bar{T}_D^H$, $\bar{T}_F^L$ and $\bar{T}_F^H$, including reference scalings (black dashed lines) of $O(N^2)$ and $O(N)$ (from top to bottom).

dt	N_T	T_m
1.00e-1	16	11.8
5.00e-2	26	17.4
2.50e-2	46	29.1
1.25e-2	86	49.3
6.25e-3	166	84.1

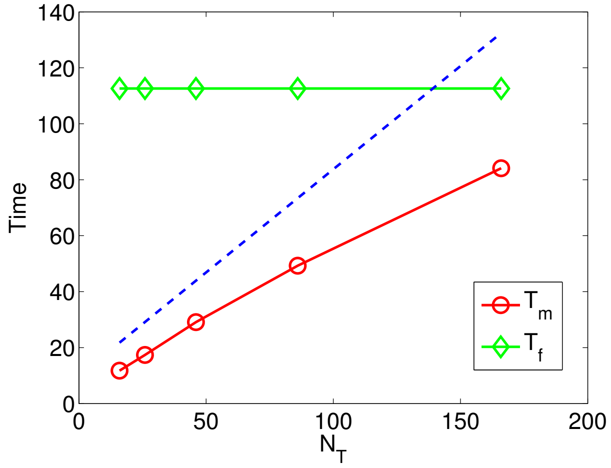

Figure 6.15 Timing results with respect to the temporal variable for Example 3. $N_S = 12750$ in all simulations. The total factorization time $T_f = 106.4$ is the same for all different time steps. In the right figure, the red line represents the total matching time; the green line represents the total factorization time; the blue dashed line represents $O(N)$ scaling.

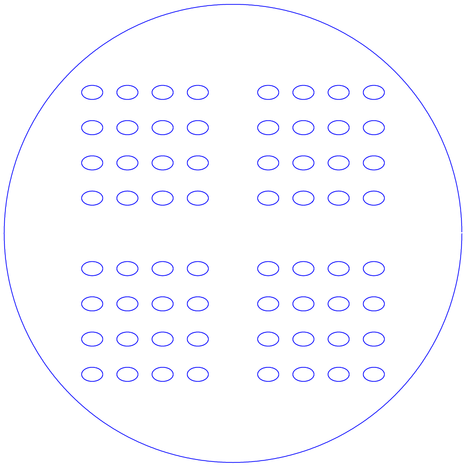

Figure 6.16 Boundary enclosed by one outer circle and sixty-four inner ellipses for Example 4.

Table 6.3 Timing Results with Respect to the Spatial Variable

N_S	$\bar{T}_f^L$	$\bar{T}_a^L$	$\bar{T}_D^L$	$\bar{T}_f^H$	$\bar{T}_a^H$	$\bar{T}_D^H$	T	r
6350	6.17	7.07e-3	1.33	11.0	1.43e-2	1.56	104	
12750	8.72	1.19e-2	5.35	13.0	2.06e-2	6.41	143	1.37
25500	11.3	2.37e-2	22.0	14.6	2.88e-2	25.7	191	1.33
51000	14.9	6.57e-2	100	15.9	4.90e-2	144	286	1.49
102000	20.3	7.65e-2	589	18.5	9.64e-2	599	476	1.66

We generate boundary data by placing heat sources *outside* the boundary region on a circle of radius 8.5 and test the accuracy of our numerical solution by comparing it with the analytical solution at 30 target points. The first ten target points are on a circle of radius 0.2 with center $(0, 5)$. The second ten target points are on a circle of radius 0.2 with center $(0, 0)$. The last ten target points are on a circle of radius 0.2 with center $(0, -5)$. Finally, refer to Example 1 for all notations and explanations for the figures and tables below.

From Figure 6.17, we observe that the algorithm has very high order convergence rate in space for Example 4. Figure 6.18 shows the overall convergence order is actually close to the theoretical value for Example 4. Both Table 6.4 and Figure 6.19 demonstrate clearly that while the direct calculation clearly exhibits quadratic complexity, our algorithm has linear complexity for Example 4. Finally, from Figure 6.20, we observe that the matching time is almost linear with respect to time steps for Example 4.

6.2 2D Heat Solver Using the Spectral Fourier Approximation

Example 5: Interior Dirichlet Problem with the Boundary Enclosed by One Outer Circle and One Inner Moving Decagram.

N_S	4500	9100	18200
E	2.02e-3	1.87e-5	6.52e-9

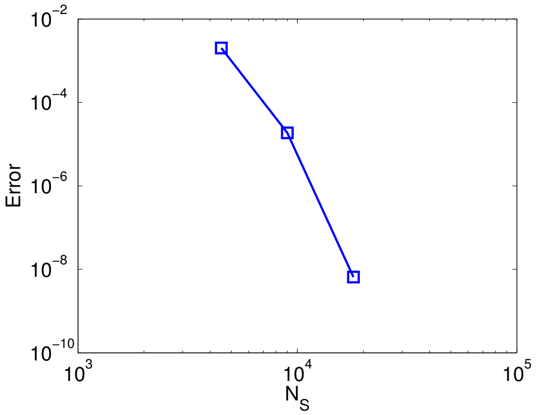

Figure 6.17 Relative L^2 errors versus N_S for Example 4. Here $N_T = 80$ and $dt = 0.0125$ are fixed. The figure on the right uses logarithmic scale for both axes.

dt	E	r
2.00e-1	6.58e-3	
1.00e-1	3.77e-4	17.4
5.00e-2	3.24e-5	11.6
2.50e-2	8.19e-8	396
1.25e-2	6.61e-9	12.37

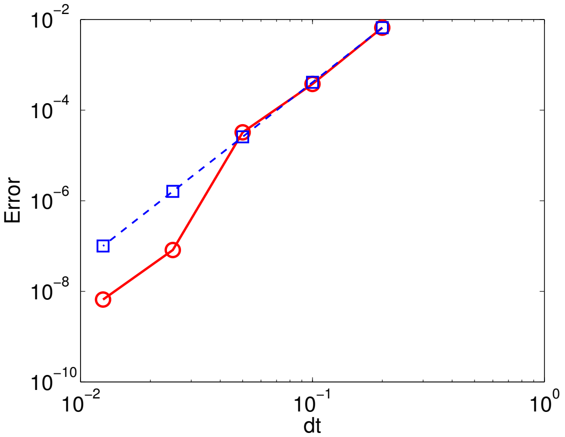

Figure 6.18 Convergence order in time for Example 4. The figure on the right is log-log plot for dt and E. $N_S = 26000$ is fixed. The red solid line is the numerical results and the blue dashed line represents the relation of dt and E in an ideal 4th order scheme.

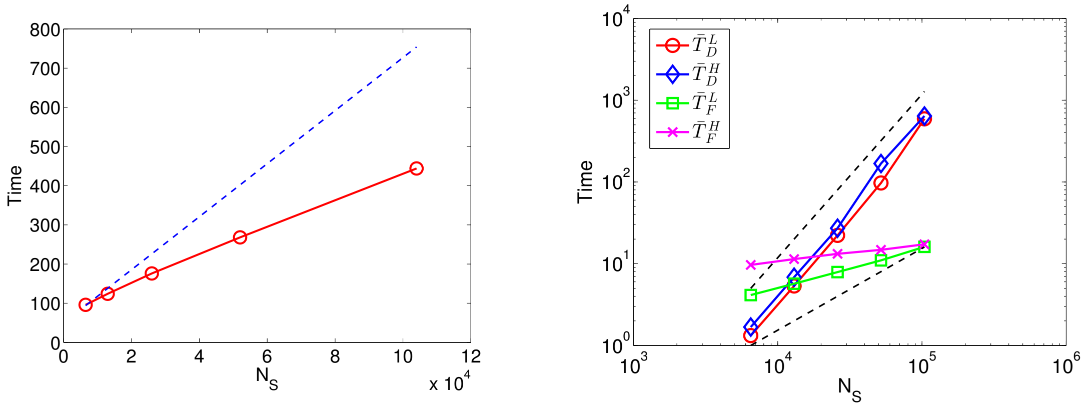

Figure 6.19 Timing results with respect to the spatial variable for Example 4. In the left figure, the red solid line is numerical results of N_S and T. The blue dashed line represents $O(N_S)$ scaling. The right figure shows the log-log plot of N_S and $\bar{T}_D^L$, $\bar{T}_D^H$, $\bar{T}_F^L$ and $\bar{T}_F^H$, including reference scalings (black dashed lines) of $O(N^2)$ and $O(N)$ (from top to bottom).

dt	N_T	T_m
1.00e-1	16	17.8
5.00e-2	26	26.1
2.50e-2	46	43.6
1.25e-2	86	74.8
6.25e-3	166	141

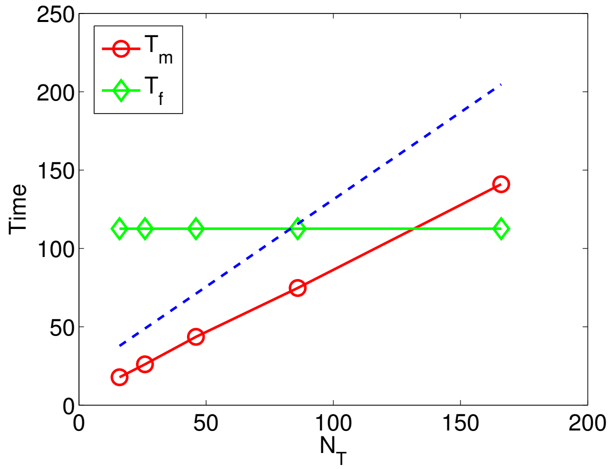

Figure 6.20 Timing results with respect to the temporal variable for Example 4. $N_S = 26000$ in all simulations. The total factorization time $T_f = 112.6$ is the same for all different time steps. In the right figure, the red line represents the total matching time; the green line represents the total factorization time; the blue dashed line represents $O(N)$ scaling.

Table 6.4 Timing Results with Respect to the Spatial Variable

N_S	$\bar{T}_f^L$	$\bar{T}_a^L$	$\bar{T}_D^L$	$\bar{T}_f^H$	$\bar{T}_a^H$	$\bar{T}_D^H$	T	r
6500	4.13	5.52e-3	1.32	9.66	1.39e-2	1.68	95.8	
13000	5.69	7.92e-3	5.34	11.4	1.82e-2	6.84	124	1.29
26000	7.90	2.50e-2	22.2	13.2	2.61e-2	27.2	176	1.41
52000	11.0	5.64e-2	97.2	14.8	5.21e-2	168	268	1.52
104000	16.1	6.87e-2	592	17.3	9.08e-2	641	444	1.65

We consider the interior Dirichlet problem with the boundary enclosed by one outer circle and one inner moving decagram shown in Figure 6.21. Here, the radius of the outer circle is 0.7 and the centers of the circle and decagram are both at $(0, 0)$ at the initial time. The size of the inner decagram is roughly 0.1 and it moves towards the positive y-axis with the velocity of 0.01. At the final time, the center of the decagram is at $(0, 0.1)$.

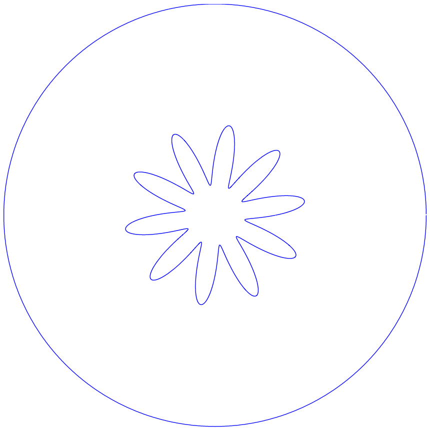

Figure 6.21 Boundary enclosed by one outer circle and one inner moving decagram for Example 5.

We generate boundary data by placing heat sources *outside* the boundary region on a circle of radius 1 and test the accuracy of our numerical solution by comparing it with the analytical solution at 20 target points on a moving circle inside the boundary region. In addition, refer to Example 1 for all notations and explanations for the figures and tables below.

From Figure 6.22, we observe that the algorithm has very high order convergence rate in space for Example 5. Figure 6.23 shows the overall convergence order is actually close to the theoretical value for Example 5.

N_S	200	400	800
E	5.09e-3	2.37e-5	9.51e-10

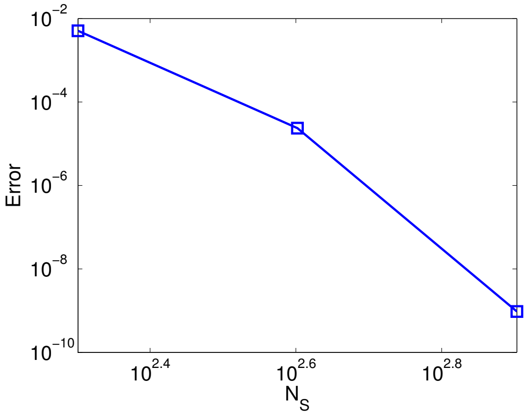

Figure 6.22 Relative L^2 errors versus N_S for Example 5. Here $N_T = 80$ and $dt = 0.0125$ are fixed. The figure on the right uses logarithmic scale for both axes.

dt	E	r
2.00e-1	1.93e-3	
1.00e-1	7.54e-5	25.5
5.00e-2	5.21e-7	144
2.50e-2	3.88e-8	13.4
1.25e-2	1.69e-9	22.9

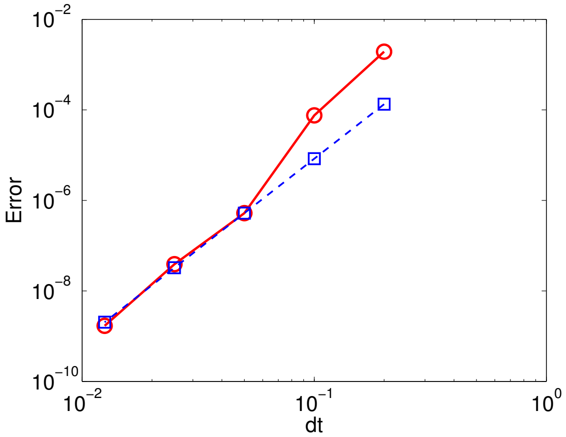

Figure 6.23 Convergence order in time for Example 5. The figure on the right is log-log plot for dt and E. $N_S = 20000$ is fixed. The red solid line is the numerical results and the blue dashed line represents the relation of dt and E in an ideal 4th order scheme.

In Table 6.5, the fifth column lists the time for evaluating the history part using NUFFT and the sixth column lists the time for evaluating the history part directly. The meanings of all other columns refer to Example 1. The figure on the left panel of Figure 6.24 shows the total computational time T versus N_S, while that on the right panel shows the factorization time and direct computational time for the local kernel and the time for evaluating the history part with and without NUFFT. Note that the complexity of evaluating the history part without NUFFT is $O(N_F N_S)$, which is linear in terms of N_S with fixed N_F. With NUFFT, the computation cost for the history part is $O((N_F + N_S)\log(N_S + N_F))$. Moreover, $N_F = 21600$ in the numerical example and the computation involving Fourier modes would dominate for evaluating the history part using NUFFT when N_S is much smaller than N_F. Both Table 6.5 and Figure 6.24 demonstrate clearly that while the direct calculation of local kernels clearly exhibits quadratic complexity, our algorithm has linear complexity. They also show NUFFT has significantly reduced the evaluation time for the history part.

Table 6.5 Timing Results with Respect to the Spatial Variable

N_S	$\bar{T}^L_f$	$\bar{T}^L_a$	$\bar{T}^L_D$	T^H_{fft}	T^H_D	T	r
2500	1.30	3.16e-3	0.33	0.32	11.8	179.6	
5000	1.92	4.99e-3	1.30	0.35	23.4	268.5	1.49
10000	4.39	8.99e-3	6.71	0.40	47.4	413.7	1.54
20000	5.14	2.26e-2	24.7	0.51	93.9	659.6	1.59
40000	5.84	3.46e-2	96.8	0.73	195	1109	1.68

Finally, from Figure 6.25, we observe that the computational time is almost linear with respect to time steps for Example 5.

Example 6: Interior Dirichlet Problem with the Boundary Enclosed by One Outer Circle and Sixteen Inner Expanding Circles.

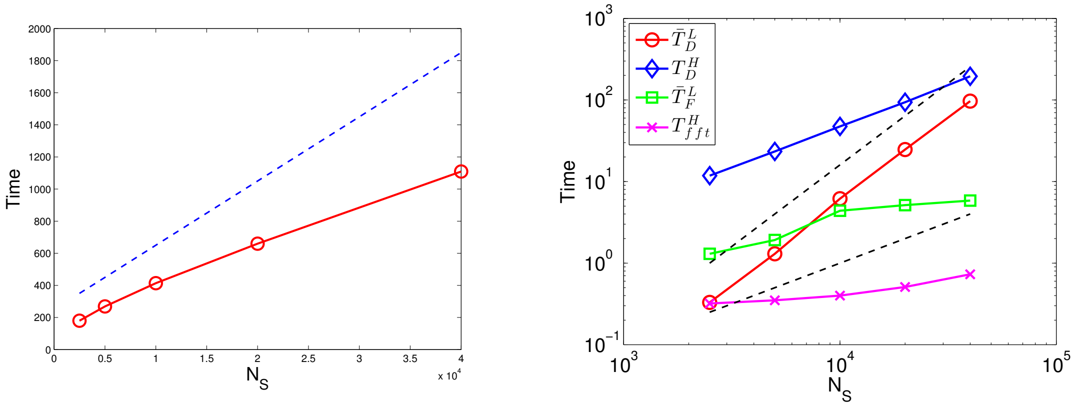

Figure 6.24 Timing results with respect to the spatial variable for Example 5. In the left figure, the red solid line is numerical results of N_S and T. The blue dashed line represents $O(N_S)$ scaling. The right figure shows the log-log plot of N_S and $\bar{T}_D^L$, T_D^H, $\bar{T}_F^L$ and T_{fft}^H, including reference scalings (black dashed lines) of $O(N^2)$ and $O(N)$ (from top to bottom).

dt	N_T	T
1.00e-1	16	25.2
5.00e-2	26	42.4
2.50e-2	46	80.3
1.25e-2	86	153
6.25e-3	166	308

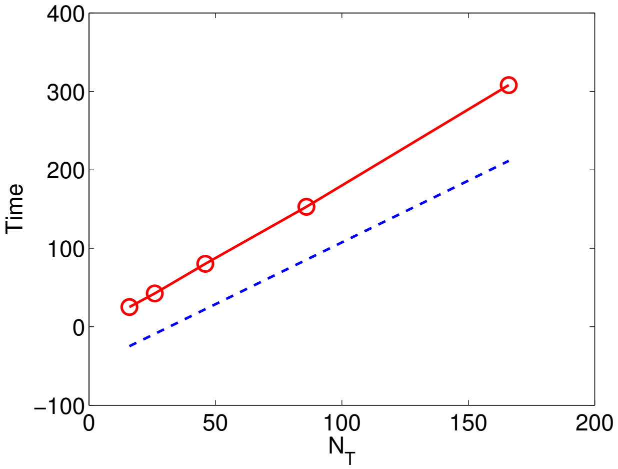

Figure 6.25 Timing results with respect to the temporal variable for Example 5. $N_S = 2000$ in all simulations. In the right figure, the red line represents the total computational time; the blue dashed line represents $O(N)$ scaling.

We consider the interior Dirichlet problem with the boundary enclosed by one outer circle and sixteen inner expanding circles shown in Figure 6.26. Here, the radius of the outer circle is 0.5 and the center of the outer circle locates at $(0, 0)$. The radius of each inner circle is 0.01 at the initial time and the radius is expanding at the speed of 0.001. At the final time, the radius of each inner circle is 0.02. The centers of the inner circles are

$$(-0.25, 0.25),\ (-0.125, 0.25),\ (0.125, 0.25),\ (0.25, 0.25),$$

$$(-0.25, 0.125),\ (-0.125, 0.125),\ (0.125, 0.125),\ (0.25, 0.125),$$

$$(-0.25, -0.125),\ (-0.125, -0.125),\ (0.125, -0.125),\ (0.25, -0.125),$$

$$(-0.25, -0.25),\ (-0.125, -0.25),\ (0.125, -0.25),\ (0.25, -0.25).$$

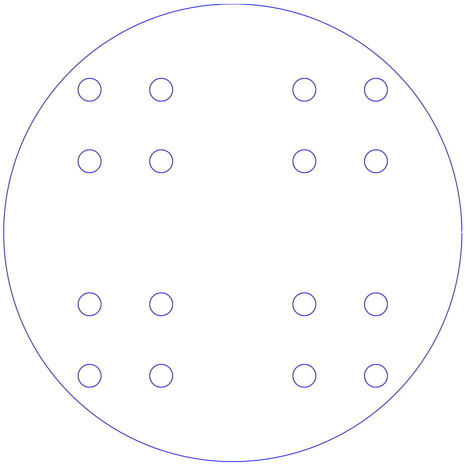

Figure 6.26 Boundary enclosed by one circle and sixteen inner expanding circles for Example 6.

We generate boundary data by placing heat sources *outside* the boundary region on a circle of radius 1 and test the accuracy of our numerical solution by comparing it with the analytical solution at 20 target points on an expanding circle of radius 0.02 with center at the origin and expanding velocity of 0.001. In addition, refer to Example 1 and Example 5 for all notations and explanations for the figures and tables below.

From Figure 6.27, we observe that the algorithm has very high order convergence rate in space for Example 6. Figure 6.28 shows the overall convergence order is actually close to the theoretical value for Example 6. Both Table 6.6 and Figure 6.29 demonstrate clearly that while the direct calculation of local kernels clearly exhibits quadratic complexity, our algorithm has linear complexity for Example 6. They also show NUFFT has significantly reduced the evaluation time for the history part. Finally, from Figure 6.30, we observe that the computational time is almost linear with respect to time steps for Example 6.

N_S	204	408	816
E	7.22e-5	1.82e-6	8.37e-11

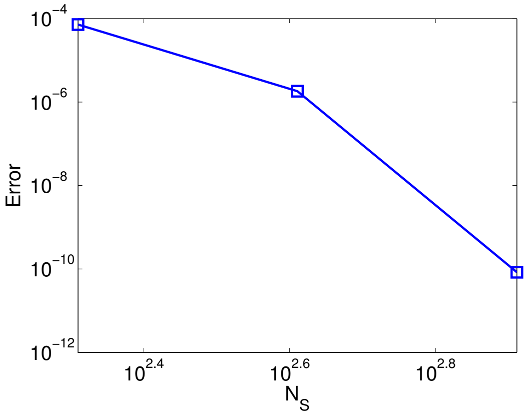

Figure 6.27 Relative L^2 errors versus N_S for Example 6. Here $N_T = 80$ and $dt = 0.0125$ are fixed. The figure on the right uses logarithmic scale for both axes.

Example 7: Interior Dirichlet Problem with the Boundary Enclosed by One Outer Circle and Thirty-six Inner Shrinking Circles.

We consider the interior Dirichlet problem with the boundary enclosed by one outer circle and thirty-six inner shrinking circles shown in Figure 6.31. Here, the radius of the outer circle is 0.5 and the center of the outer circle locates at $(0, 0)$. The

dt	E	r
2.00e-1	9.73e-4	
1.00e-1	1.89e-5	51.4
5.00e-2	4.28e-7	44.1
2.50e-2	2.88e-8	14.8
1.25e-2	5.26e-10	54.7

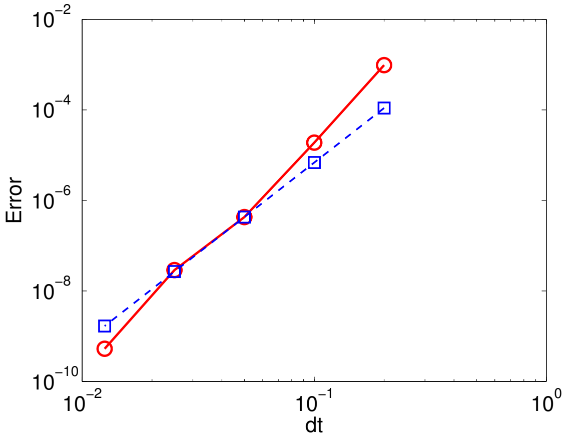

Figure 6.28 Convergence order in time for Example 6. The figure on the right is log-log plot for dt and E. $N_S = 6800$ is fixed. The red solid line is the numerical results and the blue dashed line represents the relation of dt and E in an ideal 4th order scheme.

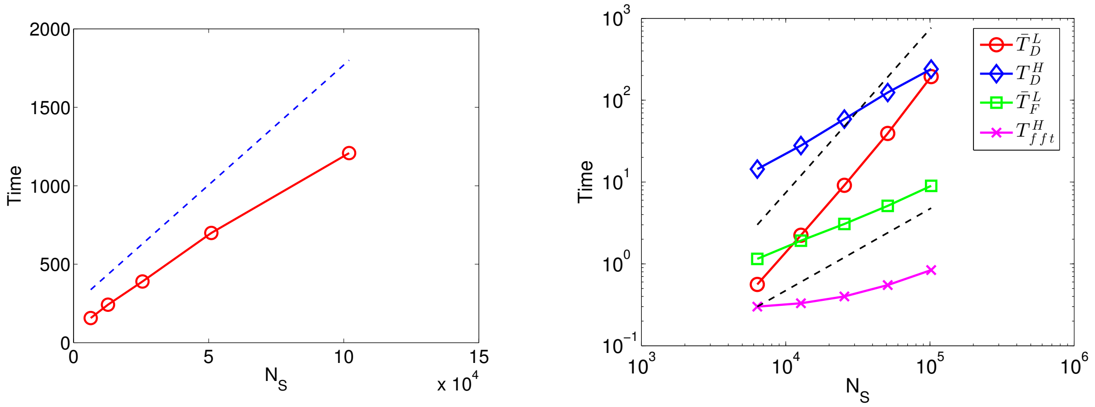

Figure 6.29 Timing results with respect to the spatial variable for Example 6. In the left figure, the red solid line is numerical results of N_S and T. The blue dashed line represents $O(N_S)$ scaling. The right figure shows the log-log plot of N_S and $\bar{T}_D^L$, T_D^H, $\bar{T}_F^L$ and T_{fft}^H, including reference scalings (black dashed lines) of $O(N^2)$ and $O(N)$ (from top to bottom).

Table 6.6 Timing Results with Respect to the Spatial Variable

N_S	$\bar{T}_f^L$	$\bar{T}_a^L$	$\bar{T}_D^L$	T_{fft}^H	T_D^H	T	r
3400	1.15	1.87e-3	0.56	0.30	14.4	157.2	
6800	1.92	3.27e-3	2.24	0.33	27.9	242.3	1.54
13600	3.07	5.93e-3	9.12	0.40	58.7	390.3	1.61
27200	5.12	1.13e-2	39.2	0.55	124	699.0	1.79
54400	8.98	2.74e-2	194	0.84	240	1208	1.72

dt	N_T	T
1.00e-1	16	42.3
5.00e-2	26	72.0
2.50e-2	46	131
1.25e-2	86	247
6.25e-3	166	482

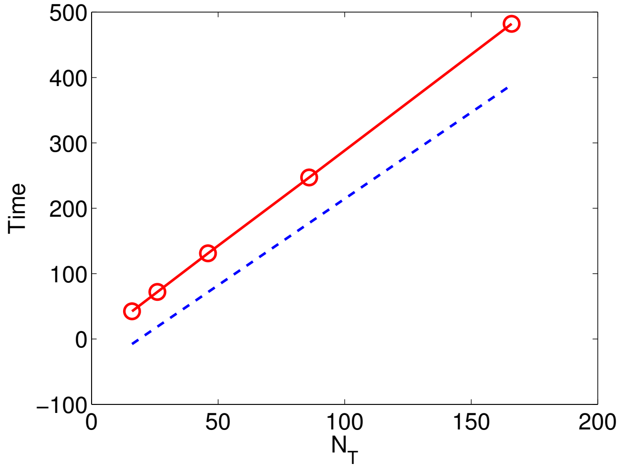

Figure 6.30 Timing results with respect to the temporal variable for Example 6. $N_S = 6800$ in all simulations. In the right figure, the red line represents the total computational time; the blue dashed line represents $O(N)$ scaling.

radius of each inner circle is 0.03 at the initial time and the radius is shrinking at the speed of 0.001. At the final time, the radius of each inner circle is 0.02. The centers of the inner circles are all permutations chosen from $\pm 0.3, \pm 0.2, \pm 0.1$.

We generate boundary data by placing heat sources *outside* the boundary region on a circle of radius 1 and test the accuracy of our numerical solution by comparing it with the analytical solution at 20 target points on a shrinking circle of radius 0.03

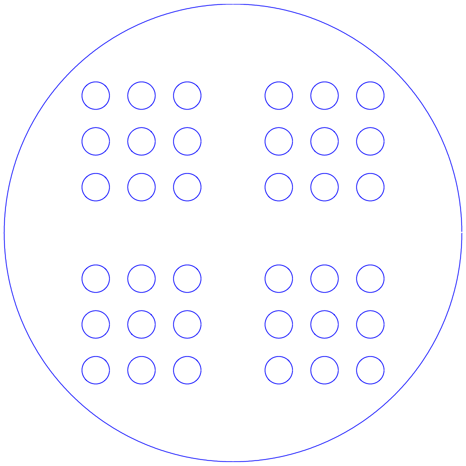

Figure 6.31 Boundary enclosed by one outer circle and thirty-six inner shrinking circles for Example 7.

with center at the origin and shrinking velocity of 0.001. In addition, refer to Example 1 and Example 5 for all notations and explanations for the figures and tables below.

From Figure 6.32, we observe that the algorithm has very high order convergence rate in space for Example 7. Figure 6.33 shows the overall convergence order is actually close to the theoretical value for Example 7. Both Table 6.7 and Figure 6.34 demonstrate clearly that while the direct calculation of local kernels clearly exhibits quadratic complexity, our algorithm has linear complexity for Example 7. They also show NUFFT has significantly reduced the evaluation time for the history part. Finally, from Figure 6.35, we observe that the computational time is almost linear with respect to time steps for Example 7.

Example 8: Interior Dirichlet Problem with the Boundary Enclosed by One Outer Circle and Sixty-four Inner Oscillating Circles.

We consider the interior Dirichlet problem with the boundary enclosed by one outer circle and sixty-four inner oscillating circles shown in Figure 6.36. Here, the radius of the outer circle is 0.5 and the center of the outer circle locates at $(0, 0)$. The radius of each inner circle is 0.01. The centers of the inner circle are all permutations

N_S	370	740	1480	2960
E	1.22e-5	4.38e-7	4.88e-8	7.45e-10

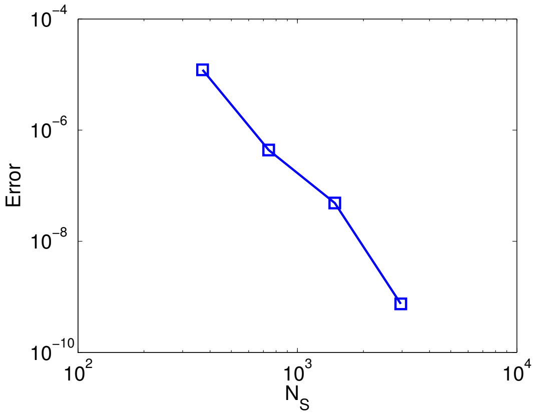

Figure 6.32 Relative L^2 errors versus N_S for Example 7. Here $N_T = 80$ and $dt = 0.0125$ are fixed. The figure on the right uses logarithmic scale for both axes.

dt	E	r
2.00e-1	2.36e-4	
1.00e-1	5.35e-6	44.1
5.00e-2	2.29e-7	23.3
2.50e-2	1.49e-8	15.3
1.25e-2	3.80e-10	39.1

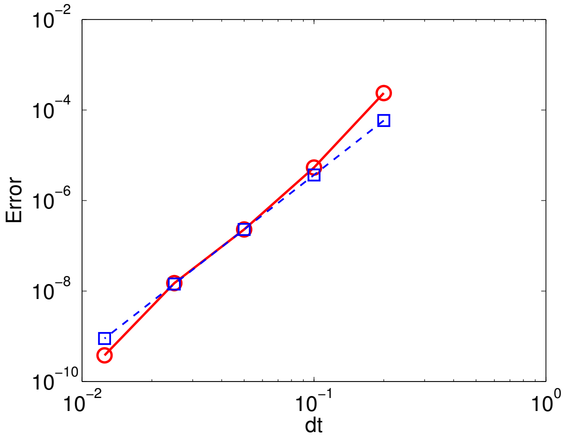

Figure 6.33 Convergence order in time for Example 7. The figure on the right is log-log plot for dt and E. $N_S = 7400$ is fixed. The red solid line is the numerical results and the blue dashed line represents the relation of dt and E in an ideal 4th order scheme.

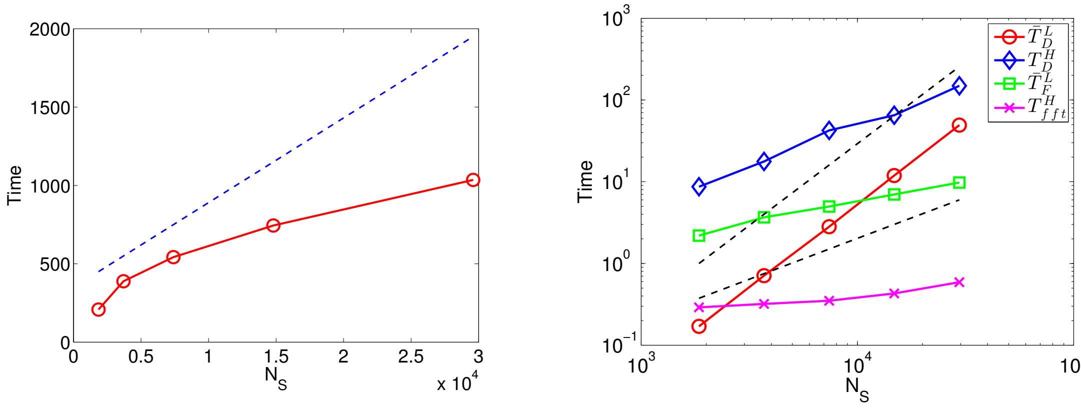

Figure 6.34 Timing results with respect to the spatial variable for Example 7. In the left figure, the red solid line is numerical results of N_S and T. The blue dashed line represents $O(N_S)$ scaling. The right figure shows the log-log plot of N_S and $\bar{T}_D^L$, T_D^H, $\bar{T}_F^L$ and T_{fft}^H, including reference scalings (black dashed lines) of $O(N^2)$ and $O(N)$ (from top to bottom).

dt	N_T	T
1.00e-1	16	81.4
5.00e-2	26	128
2.50e-2	46	281
1.25e-2	86	535
6.25e-3	166	941

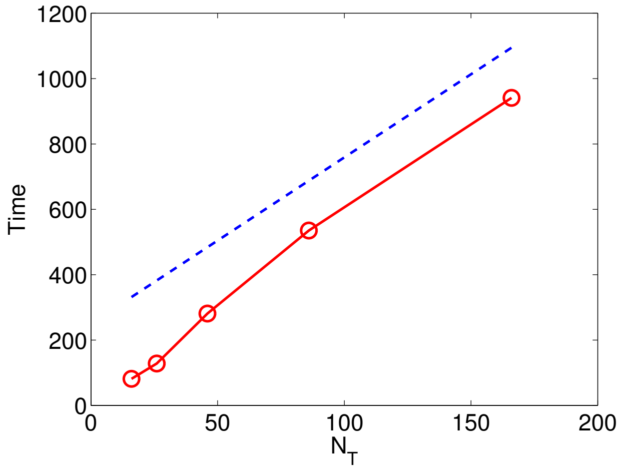

Figure 6.35 Timing results with respect to the temporal variable for Example 7. $N_S = 7400$ in all simulations. In the right figure, the red line represents the total computational time; the blue dashed line represents $O(N)$ scaling.

Table 6.7 Timing Results with Respect to the Spatial Variable

N_S	$\bar{T}^L_f$	$\bar{T}^L_a$	$\bar{T}^L_D$	T^H_{fft}	T^H_D	T	r
1850	2.19	4.16e-3	0.17	0.29	8.69	208.1	
3700	3.67	8.16e-3	0.71	0.32	17.7	388.8	1.86
7400	4.97	9.99e-3	2.82	0.35	42.5	542.5	1.39
14800	6.98	1.86e-2	11.9	0.43	65.1	744.4	1.37
29600	9.78	3.09e-2	49.2	0.59	149	1035	1.39

chosen from $\pm 0.25, \pm 0.185, \pm 0.125, \pm 0.0625$ at the initial time and then changed to $(c_1 + v * \cos(\tau),\ c_2 + v * \sin(\tau))$, where $\tau \in (0, 10]$ is the matching time, $v = 0.001$ and $(c_1,\ c_2)$ is the center of each inner circle at the initial time.

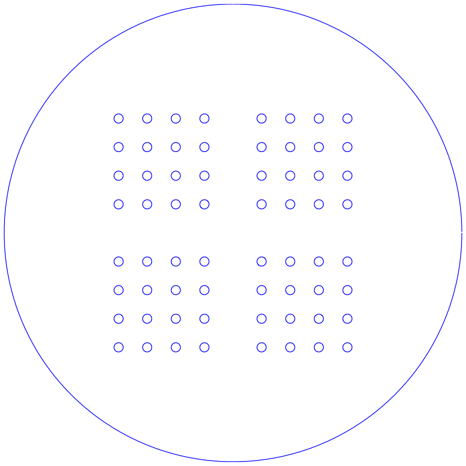

Figure 6.36 Boundary enclosed by one outer circle and sixty-four inner oscillating circles for Example 8.

We generate boundary data by placing heat sources *outside* the boundary region on a circle of radius 1 and test the accuracy of our numerical solution by comparing it with the analytical solution at 20 target points on an oscillating circle of radius 0.03.

In addition, refer to Example 1 and Example 5 for all notations and explanations for the figures and tables below.

From Figure 6.37, we observe that the algorithm has very high order convergence rate in space for Example 8. Figure 6.38 shows the overall convergence order is actually close to the theoretical value for Example 8. Both Table 6.8 and Figure 6.39 demonstrate clearly that while the direct calculation of local kernels clearly exhibits quadratic complexity, our algorithm has linear complexity for Example 8. They also show NUFFT has significantly reduced the evaluation time for the history part. Finally, from Figure 6.40, we observe that the computational time is almost linear with respect to time steps for Example 8.

N_S	780	1300	1820	2340
E	1.37e-6	4.14e-8	5.39e-9	2.80e-10

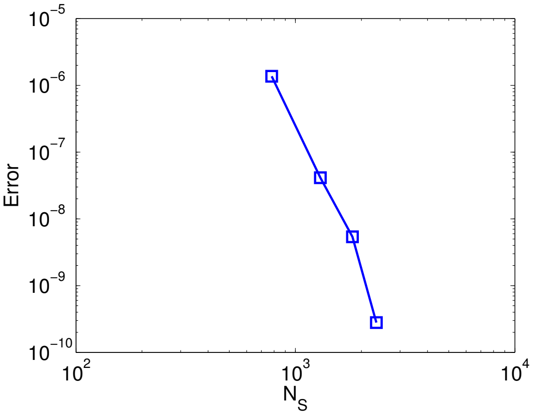

Figure 6.37 Relative L^2 errors versus N_S for Example 8. Here $N_T = 80$ and $dt = 0.0125$ are fixed. The figure on the right uses logarithmic scale for both axes.

6.3 3D Heat Solver Using SOE Approximations

Example 9: Interior Dirichlet Problem with the W-shape Boundary

We consider the interior Dirichlet problem with the w-shape boundary shown in Figure 6.41. Here the boundary surface is roughly of size $R = 1$.

We generate boundary data by placing heat sources *outside* the boundary surface and test the accuracy of our numerical solution by comparing it with the analytical

dt	E	r
2.00e-1	1.58e-4	
1.00e-1	3.68e-6	42.9
5.00e-2	6.83e-8	53.9
2.50e-2	7.76e-9	8.79
1.25e-2	1.29e-9	6.01

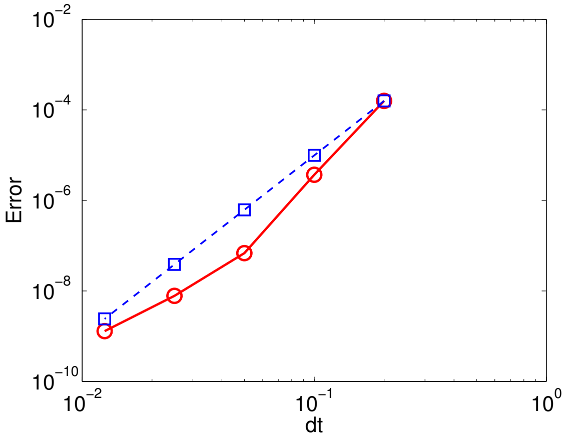

Figure 6.38 Convergence order in time for Example 8. The figure on the right is log-log plot for dt and E. $N_S = 6500$ is fixed. The red solid line is the numerical results and the blue dashed line represents the relation of dt and E in an ideal 4th order scheme.

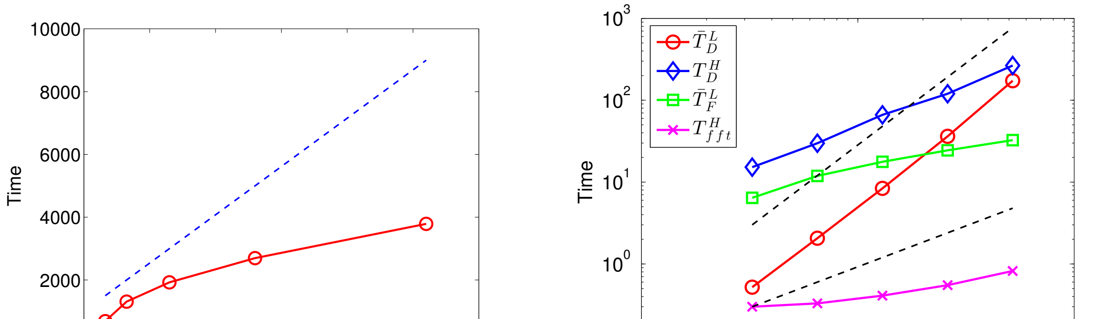

Figure 6.39 Timing results with respect to the spatial variable for Example 8. In the left figure, the red solid line is numerical results of N_S and T. The blue dashed line represents $O(N_S)$ scaling. The right figure shows the log-log plot of N_S and $\bar{T}_D^L$, T_D^H, $\bar{T}_F^L$ and T_{fft}^H, including reference scalings (black dashed lines) of $O(N^2)$ and $O(N)$ (from top to bottom).

dt	N_T	T
1.00e-1	16	211.6
5.00e-2	26	365.6
2.50e-2	46	681.0
1.25e-2	86	1307
6.25e-3	166	2622

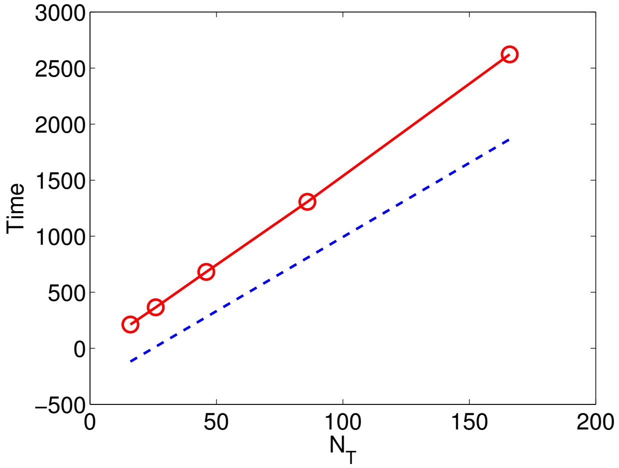

Figure 6.40 Timing results with respect to the temporal variable for Example 8. $N_S = 7400$ in all simulations. In the right figure, the red line represents the total computational time; the blue dashed line represents $O(N)$ scaling.

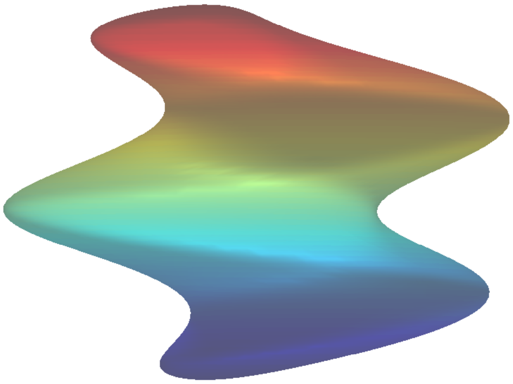

Figure 6.41 W-shape boundary surface for Example 9.

Table 6.8 Timing Results with Respect to the Spatial Variable

N_S	$\bar{T}_f^L$	$\bar{T}_a^L$	$\bar{T}_D^L$	T_{fft}^H	T_D^H	T	r
3250	6.42	9.16e-3	0.52	0.30	15.2	688.7	
6500	11.9	1.66e-2	2.06	0.33	29.8	1310	1.90
13000	17.7	3.39e-2	8.39	0.41	66.2	1925	1.46
26000	24.4	4.96e-2	36.2	0.55	120	2696	1.40
52000	32.5	7.64e-2	173	0.82	265	3788	1.40

solution at 25 target points inside the boundary surface. In addition, refer to Example 1 for all notations and explanations for the figures and tables below.

Figure 6.42 shows the algorithm has very high order convergence rate in space for Example 9. From Figure 6.43, we observe the convergence rate is roughly consistent with fourth order accuracy for Example 9. Figure 6.44 clearly demonstrates the matching time is almost linear with respect to time steps for Example 9.

N_θ	N_ϕ	N_S	E
12	24	288	7.39e-2
48	96	4608	7.08e-4
84	180	15120	5.38e-6
120	240	28800	5.47e-8

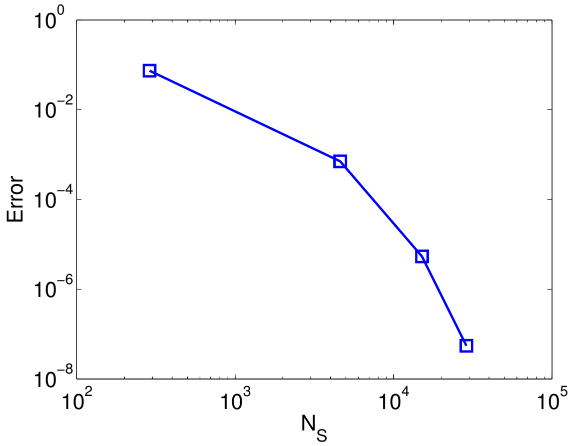

Figure 6.42 Relative L^2 errors versus N_S for Example 9. Here $N_T = 160$ and $dt = 0.00625$ are fixed. The figure on the right uses logarithmic scale for both axes.

Example 10: Interior Dirichlet Problem with the Bowl Boundary

dt	E	r
1.00e-1	1.18e-4	
5.00e-2	2.62e-6	45.2
2.50e-2	2.28e-7	11.4
1.25e-2	5.28e-9	43.1

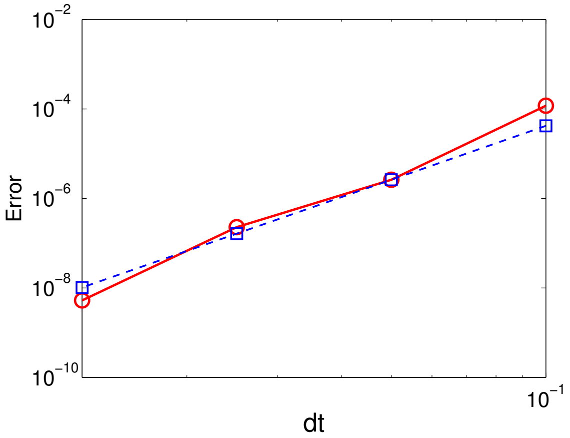

Figure 6.43 Convergence order in time for Example 9. The figure on the right is log-log plot for dt and E. $N_S = 35640$ is fixed. The red solid line is the numerical results and the blue dashed line represents the relation of dt and E in an ideal 4th order scheme.

dt	N_T	T_m
1.00e-1	10	1512
5.00e-2	20	2782
2.50e-2	40	5488
1.25e-2	80	10577
6.25e-3	160	20252

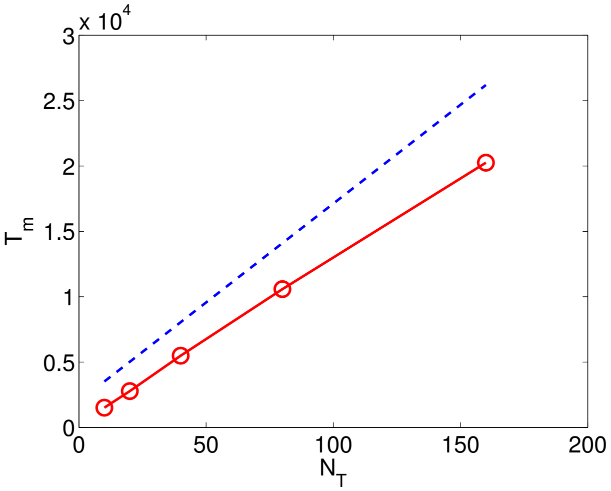

Figure 6.44 Timing results with respect to the temporal variable for Example 9. $N_S = 23328$ ($N_\theta = 108$ and $N_\phi = 216$) in all simulations. In the right figure, the red line represents the total matching time and the blue dashed line represents $O(N)$ scaling.

We consider the interior Dirichlet problem with the bowl boundary shown in Figure 6.45. Here the boundary surface is roughly of size $R = 1$.

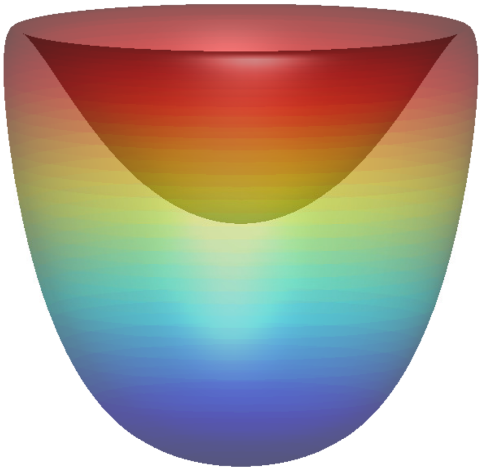

Figure 6.45 Bowl boundary surface for Example 10.

We generate boundary data by placing heat sources *outside* the boundary surface and test the accuracy of our numerical solution by comparing it with the analytical solution at 16 target points inside the boundary surface. In addition, refer to Example 1 for all notations and explanations for the figures and tables below.

From Figure 6.46, we observe that the algorithm has very high order convergence rate in space for Example 10. Figure 6.47 shows the convergence rate is roughly consistent with the theoretical value for Example 10. Finally, Figure 6.48 clearly demonstrates the matching time is almost linear with respect to time steps for Example 10.

Example 11: Interior Dirichlet Problem with the Starfish Boundary

We consider the interior Dirichlet problem with the starfish boundary shown in Figure 6.49. Here the boundary surface is roughly of size $R = 2$.

N_θ	N_ϕ	N_S	E
12	24	288	8.06e-3
36	72	2592	5.43e-5
72	144	10368	3.15e-7
108	216	23328	5.33e-9

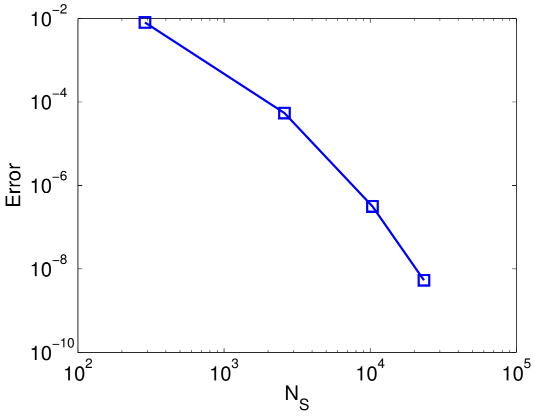

Figure 6.46 Relative L^2 errors versus N_S for Example 10. Here $N_T = 160$ and $dt = 0.00625$ are fixed. The figure on the right uses logarithmic scale for both axes.

dt	E	r
1.00e-1	1.06e-4	
5.00e-2	2.63e-6	40.1
2.50e-2	2.21e-7	11.9
1.25e-2	7.28e-9	30.3
6.25e-3	8.62e-10	8.44

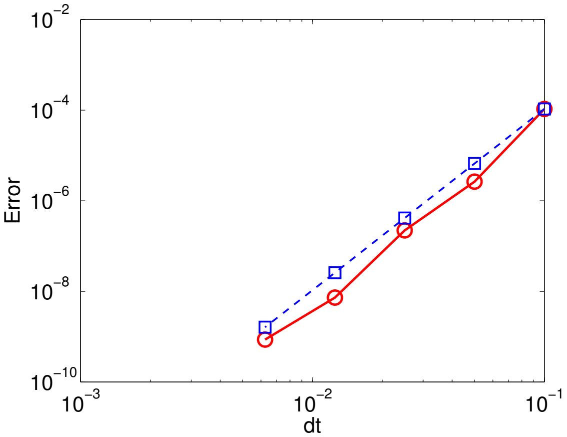

Figure 6.47 Convergence order in time for Example 10. The figure on the right is log-log plot for dt and E. $N_S = 40320$ is fixed. The red solid line is the numerical results and the blue dashed line represents the relation of dt and E in an ideal 4th order scheme.

dt	N_T	T_m
1.00e-1	10	555
5.00e-2	20	1018
2.50e-2	40	1979
1.25e-2	80	3642
6.25e-3	160	6997

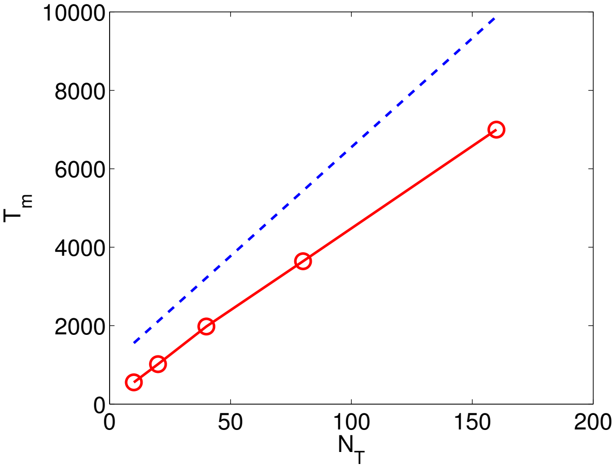

Figure 6.48 Timing results with respect to the temporal variable for Example 10. $N_S = 15120$ ($N_\theta = 84$ and $N_\phi = 180$) in all simulations. In the right figure, the red line represents the total matching time and the blue dashed line represents $O(N)$ scaling.

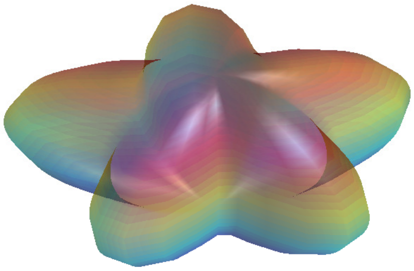

Figure 6.49 Starfish boundary surface for Example 11.

We generate boundary data by placing heat sources *outside* the boundary surface and test the accuracy of our numerical solution by comparing it with the analytical solution at 16 target points inside the boundary surface. In addition, refer to Example 1 for all notations and explanations for the figures and tables below.

From Figure 6.50, we observe that the algorithm has very high order convergence rate in space for Example 11. Figure 6.51 shows the convergence rate is roughly consistent with the theoretical value for Example 11. Finally, Figure 6.52 clearly demonstrates the matching time is almost linear with respect to time steps for Example 11.

N_θ	N_ϕ	N_S	E
12	24	288	5.54e-3
36	72	2592	1.34e-4
60	120	7200	1.36e-6
84	180	15120	2.47e-9

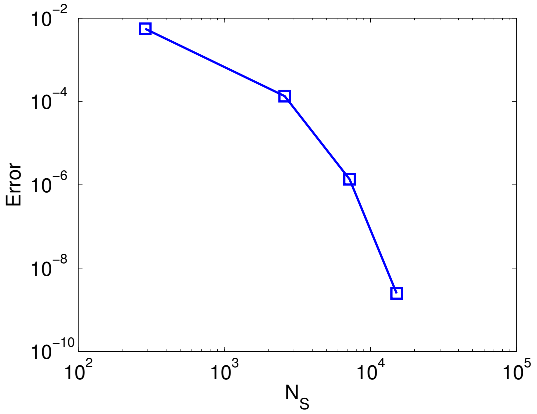

Figure 6.50 Relative L^2 errors versus N_S for Example 11. Here $N_T = 160$ and $dt = 0.00625$ are fixed. The figure on the right uses logarithmic scale for both axes.

Example 12: Exterior Dirichlet Problem with the Four-sphere Boundary

We consider the exterior Dirichlet problem with the boundary consisting of four spheres shown in Figure 6.53. Here the radius of each sphere is 1.2. The centers of the spheres are

$$(-3,-3,3),\ (-3,3,3),\ (-3,-3,-3),\ (3,3,-3).$$

dt	E	r
1.00e-1	9.20e-4	
5.00e-2	1.53e-4	5.99
2.50e-2	1.98e-5	7.73
1.25e-2	7.68e-7	25.8
6.25e-3	2.47e-9	310

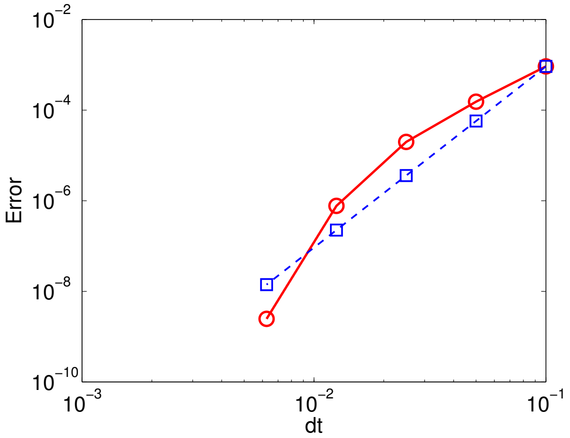

Figure 6.51 Convergence order in time for Example 11. The figure on the right is log-log plot for dt and E. $N_S = 15120$ is fixed. The red solid line is the numerical results and the blue dashed line represents the relation of dt and E in an ideal 4th order scheme.

dt	N_T	T_m
1.00e-1	10	520
5.00e-2	20	992
2.50e-2	40	1859
1.25e-2	80	3553
6.25e-3	160	7146

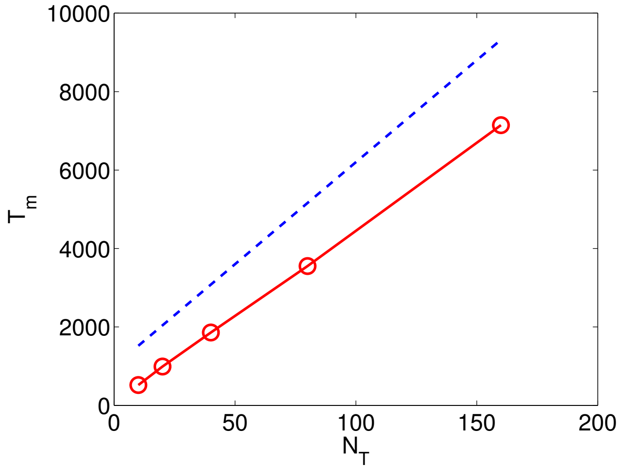

Figure 6.52 Timing results with respect to the temporal variable for Example 11. $N_S = 15120$ ($N_\theta = 84$ and $N_\phi = 180$) in all simulations. In the right figure, the red line represents the total matching time and the blue dashed line represents $O(N)$ scaling.

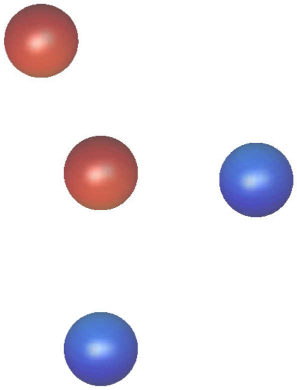

Figure 6.53 Four-sphere boundary surface for Example 12.

We generate boundary data by placing heat sources *inside* spheres and test the accuracy of our numerical solution by comparing it with the analytical solution at 25 target points outside the boundary surface. In addition, refer to Example 1 for all notations and explanations for the figures and tables below.

From Figure 6.54, we observe that the algorithm has very high order convergence rate in space for Example 12. Figure 6.55 shows the convergence rate is roughly consistent with theoretical value for Example 12. Finally, Figure 6.56 clearly demonstrates the matching time is almost linear with respect to time steps for Example 12.

Example 13: Exterior Dirichlet Problem with the Four-ellipsoid Boundary

We consider the exterior Dirichlet problem with the boundary consisting of four ellipsoids shown in Figure 6.57. The lengths of the three simi-principle axes a, b, c are $1.1, 1.0, 0.9$, respectively. The centers of the ellipsoids are

$$(-3.5, -3.5, 3.5),\ (-3.5, 3.5, 3.5),\ (-3.5, -3.5, -3.5),\ (3.5, 3.5, -3.5).$$

N_θ	N_ϕ	N_S	E
12	24	1152	2.62e-2
24	48	4608	3.88e-4
36	72	10368	7.67e-7
50	100	20000	5.07e-9

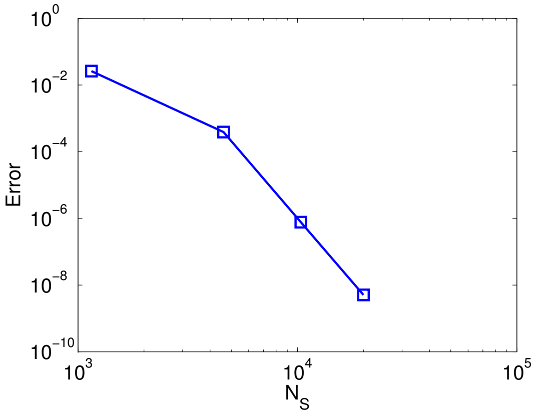

Figure 6.54 Relative L^2 errors versus N_S for Example 12. Here $N_T = 320$ and $dt = 0.003125$ are fixed. $N_S = 4 \cdot N_\theta \cdot N_\phi$ as there are four spheres. The figure on the right uses logarithmic scale for both axes.

dt	E	r
1.00e-1	1.58e-2	
5.00e-2	4.49e-4	35.3
2.50e-2	1.94e-4	2.30
1.25e-2	2.00e-5	9.73
6.25e-3	5.54e-7	36.1
3.125e-3	5.07e-9	109

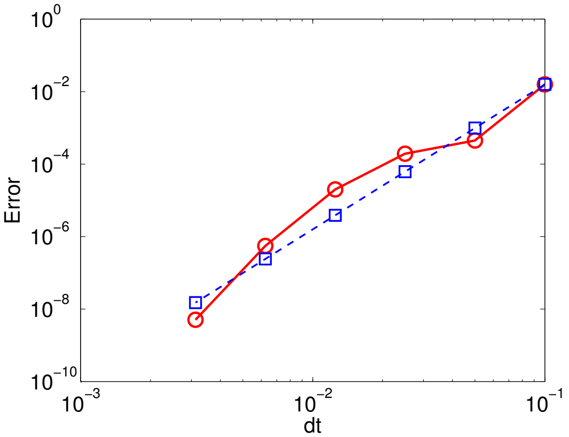

Figure 6.55 Convergence order in time for Example 12. The figure on the right is log-log plot for dt and E. $N_S = 20000$ is fixed. The red solid line is the numerical results and the blue dashed line represents the relation of dt and E in an ideal 4th order scheme.

dt	N_T	T_m
1.00e-1	10	738
5.00e-2	20	1379
2.50e-2	40	2740
1.25e-2	80	5142
6.25e-3	160	9596

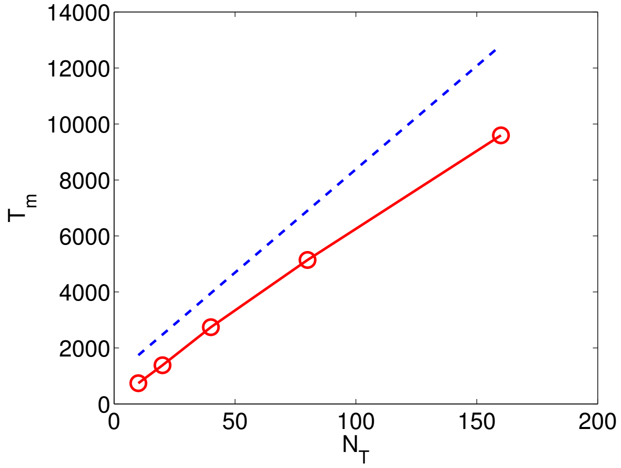

Figure 6.56 Timing results with respect to the temporal variable for Example 12. $N_S = 20000$ in all simulations. In the right figure, the red line represents the total matching time and the blue dashed line represents $O(N)$ scaling.

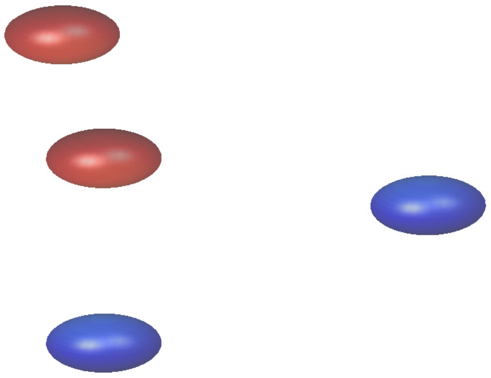

Figure 6.57 Four-ellipsoid boundary surface for Example 13.

We generate boundary data by placing heat sources *inside* ellipsoids and test the accuracy of our numerical solution by comparing it with the analytical solution at 25 target points outside the boundary surface. In addition, refer to Example 1 for all notations and explanations for the figures and tables below.

From Figure 6.58, we observe that the algorithm has very high order convergence rate in space for Example 13. Figure 6.59 shows the convergence rate is roughly consistent with the theoretical value for Example 13. Finally, Figure 6.60 clearly demonstrates the matching time is almost linear with respect to time steps for Example 13.

N_θ	N_ϕ	N_S	E
12	24	1152	9.87e-3
24	48	4608	2.72e-5
36	72	10368	2.10e-8
48	90	17280	7.02e-9

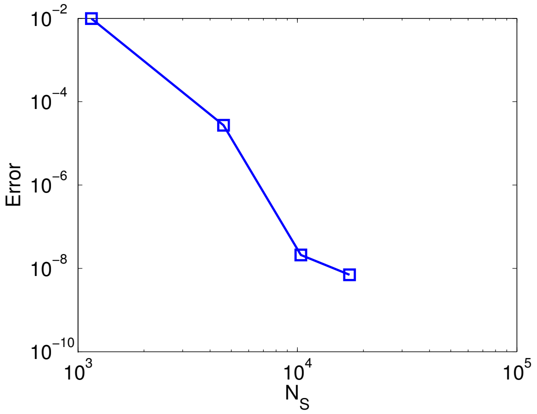

Figure 6.58 Relative L^2 errors versus N_S for Example 13. Here $N_T = 320$ and $dt = 0.003125$ are fixed. $N_S = 4 \cdot N_\theta \cdot N_\phi$ as there are four ellipsoids. The figure on the right uses logarithmic scale for both axes.

dt	E	r
1.00e-1	1.15e-2	
5.00e-2	1.59e-3	7.24
2.50e-2	5.57e-4	2.86
1.25e-2	3.08e-5	18.0
6.25e-3	1.06e-6	29.0
3.125e-3	7.02e-9	151

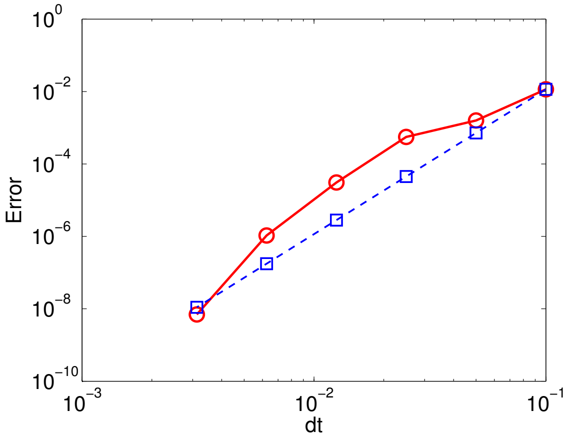

Figure 6.59 Convergence order in time for Example 13. The figure on the right is log-log plot for dt and E. $N_S = 17280$ is fixed. The red solid line is the numerical results and the blue dashed line represents the relation of dt and E in an ideal 4th order scheme.

dt	N_T	T_m
1.00e-1	10	982
5.00e-2	20	1541
2.50e-2	40	2737
1.25e-2	80	4662
6.25e-3	160	8598

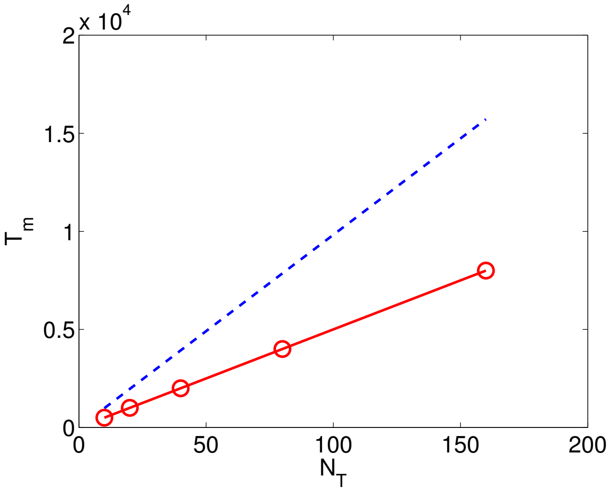

Figure 6.60 Timing results with respect to the temporal variable for Example 13. $N_S = 17280$ in all simulations. In the right figure, the red line represents the total matching time and the blue dashed line represents $O(N)$ scaling.

CHAPTER 7

CONCLUSIONS AND FURTHER DISCUSSIONS

In this dissertation, we have constructed an efficient *separated* sum-of-exponentials approximation for the heat kernel in arbitrary dimensions. The approximation relies on the SOE approximation for the 1D heat kernel and power functions. We have developed fast and high-order numerical algorithms for solving the boundary value problems of the heat equation in both two and three dimensions. For two-dimensional problems with *stationary* complex geometries, we use the SOE approximation for the heat kernel and compress all local and history kernels only once. The resulting algorithm is very efficient with quasilinear complexity and is capable of dealing with both the interior and the exterior problems. For two-dimensional problems with *moving* complex geometries, we apply the spectral Fourier approximation for the heat kernel and NUFFT to speed up the evaluation of the history part of the heat potentials. The algorithm does require the compression of the local kernels at each time step due to the facts that the boundary is moving and the local kernels change correspondingly. The algorithm also has the restrictions that the physical domain shall not be very large and the time step size shall not be very small and is only applicable for interior problems. Nevertheless, the overall complexity of the algorithm is quasilinear in both space and time. Furthermore, the algorithm achieves high order for *moving* complex geometries. We expect that the algorithm will have some applications in applied physics and engineering such as dislocation dynamics in materials science.

For three-dimensional problems, we have used the SOE approximations to speed up the evaluation of the history part. The integrals involving the local kernels are treated via the spectrally accurate integrator. The algorithm is applicable for both interior and exterior problems and has quasilinear complexity with respect to the temporal variable. The algorithm is expected to have applications on the diffusion process of nanoparticle assembly where there are a large number of particles with the surface of each nanoparticle requiring very small number of discretization points. Moreover, it is more or less straightforward to use some other high order integrator for the spatial integrals and apply fast algorithms to compress or speed up the evaluation of the corresponding discrete summation in the spatial variable. One candidate is the FMM accelerated QBX - "Quadrature by Expansion" scheme. By this, we will have near optimal algorithm in both space and time.

In this dissertation, we have dealt only with the boundary value problems of the heat equation. The SOE approximation can obviously be applied to speed up the evaluation of the history part of the heat volume potential. When combined with the Fast Gauss Transform, we will be able to build an efficient and high-order solver for solving the general initial-boundary value problem for the inhomogeneous heat equation in both two and three dimensions. We are currently working on these projects and will report our findings on a later date.

APPENDIX A

SUM-OF-EXPONENTIALS WEIGHTS AND NODES FOR THE 1D HEAT KERNEL

We give the weights and nodes for the sum-of-exponentials approximation for the 1D heat kernel $G_1 = e^{-x^2/4t}/\sqrt{4\pi t}$. $G_1 \approx \sum_{k=N_1}^{N_1} w_k e^{s_k t} e^{-\sqrt{s_k} x}$. Here $N_1 = 47$ and only the exponentials in the lower half of the complex plane are listed. The relative error is 10^{-9} for $t \in [10^{-3}, 1]$. In Table A.1, the first column lists the values of s_k and the second column lists the values of w_k.

Table A.1 Weights and Nodes for the Sum-of-exponentials Approximation for the 1D Heat Kernel

s_k	w_k
+5.2543538566883130D-001 -1.5355921434143971D-001i	3.3406450069243053D-002 + 7.8694879334856565D-004i
+4.5124072851051300D-001 -4.6901795847816991D-001i	3.3858524697168449D-002 + 2.3714958079918535D-003i
+2.9882165376338349D-001 -8.0995063945421741D-001i	3.4768791677853524D-002 + 3.9881352204446229D-003i
+5.9899770775166200D-002 -1.1948744535190035D+000i	3.6149569249415578D-002 + 5.6587442927134319D-003i
-2.7850156846250274D-001 -1.6446959027588257D+000i	3.8019542860491549D-002 + 7.4059306342027370D-003i
-7.3476207454759612D-001 -2.1838462972755117D+000i	4.0404018032102237D-002 + 9.2533381404359474D-003i
-1.3336627843342230D+000 -2.8416087014987785D+000i	4.3335262806011098D-002 + 1.1225966955375540D-002i
-2.1077320020767232D+000 -3.6537083955326728D+000i	4.6852944413778977D-002 + 1.3350511787250059D-002i
-3.0990120206246341D+000 -4.6642532348208814D+000i	5.1004666075762921D-002 + 1.5655723156165313D-002i
-4.3613425792163030D+000 -5.9281292955658031D+000i	5.5846611194320855D-002 + 1.8172796462103385D-002i
-5.9632850801149733D+000 -7.5139819214310553D+000i	6.1444303658802629D-002 + 2.0935794138393353D-002i
-7.9918463880908517D+000 -9.5079440821622647D+000i	6.7873494551196165D-002 + 2.3982106603468702D-002i
-1.0557204464776218D+001 -1.2018314543775274D+001i	7.5221187251822183D-002 + 2.7352958248766927D-002i
-1.3798692502929331D+001 -1.5181439937409575D+001i	8.3586814817376690D-002 + 3.1093965310080390D-002i
-1.7892366578989638D+001 -1.9169120201715121D+001i	9.3083585564254306D-002 + 3.5255753171782450D-002i
-2.3060567848491313D+001 -2.4197939613161662D+001i	1.0384001506633338D-001 + 3.9894641457634002D-002i
-2.9583998639207351D+001 -3.0541030203791603D+001i	1.1600166529906714D-001 + 4.5073406179197625D-002i
-3.7816968335105827D+001 -3.8542906477049684D+001i	1.2973311446494412D-001 + 5.0862129255676444D-002i
-4.8206637106159420D+001 -4.8638177144812161D+001i	1.4522018415709023D-001 + 5.7339146901350516D-002i
-6.1317302675453163D+001 -6.1375150182749685D+001i	1.6267245400022495D-001 + 6.4592109714717016D-002i
-7.7861049219288773D+001 -7.7445613273673700D+001i	1.8232609779849215D-001 + 7.2719168815048321D-002i
-9.8736423044743205D+001 -9.7722407114526959D+001i	2.0444707957048894D-001 + 8.1830304077826238D-002i
-1.2507723565031891D+002 -1.2330683231921709D+002i	2.2933475272201531D-001 + 9.2048812443472808D-002i
-1.5831414482743193D+002 -1.5558846474502963D+002i	2.5732591106254793D-001 + 1.0351297643999625D-001i
-2.0025235847869459D+002 -1.9632062801341692D+002i	2.8879934648604044D-001 + 1.1637793549892739D-001i
-2.5316968150772374D+002 -2.4771562239065099D+002i	3.2418097499311860D-001 + 1.3081778538823258D-001i
-3.1994023103558078D+002 -3.1256488222792592D+002i	3.6394960042284941D-001 + 1.4702793417289575D-001i
-4.0419053933177224D+002 -3.9439058811408586D+002i	4.0864339389209720D-001 + 1.6522774658533385D-001i
-5.1049652293304160D+002 -4.9763696830499521D+002i	4.5886717662584403D-001 + 1.8566351259073638D-001i
-6.4463201600277023D+002 -6.2791167965116256D+002i	5.1530060473377282D-001 + 2.0861178031960287D-001i
-8.1388236661047688D+002 -7.9229037823057672D+002i	5.7870736669405309D-001 + 2.3438309847056926D-001i
-1.0274401283961181D+003 -9.9970102193834418D+002i	6.4994551800979161D-001 + 2.6332621882780033D-001i
-1.2969043389592882D+003 -1.2614087777944171D+003i	7.2997909289244700D-001 + 2.9583281576374076D-001i
-1.6369105024555524D+003 -1.5916278709100643D+003i	8.1989115010796476D-001 + 3.3234278659416305D-001i
-2.0659254928655710D+003 -2.0082936066805410D+003i	9.2089842952925771D-001 + 3.7335020451285178D-001i
-2.6072505517969930D+003 -2.5340364973160094D+003i	1.0343678177356390D+000 + 4.1941000466434852D-001i
-3.2902868569904354D+003 -3.1974114007865637D+003i	1.1618348454808407D+000 + 4.7114549383439813D-001i
-4.1521323987955984D+003 -4.0344484308358847D+003i	1.3050244673476217D+000 + 5.2925678538340226D-001i
-5.2395968963294799D+003 -5.0906098731829061D+003i	1.4658744047907917D+000 + 5.9453027356908161D-001i
-6.6117441901645352D+003 -6.4232593945873459D+003i	1.6465613684596623D+000 + 6.6784927547013706D-001i
-8.3431001974123810D+003 -8.1047776557969619D+003i	1.8495305146549577D+000 + 7.5020598452333265D-001i
-1.0527700663992231D+004 -1.0226493547754852D+004i	2.0775285345412993D+000 + 8.4271489743589800D-001i
-1.3284198561072539D+004 -1.2903644569677639D+004i	2.3336408238968738D+000 + 9.4662789617368170D-001i
-1.6762308525499571D+004 -1.6281635763758291D+004i	2.6213332364019597D+000 + 1.0633511891228977D+000i
-2.1150938363365654D+004 -2.0543937150208803D+004i	2.9444989854949983D+000 + 1.1944643406826623D+000i
-2.6688449265921696D+004 -2.5922048598801452D+004i	3.3075113294981566D+000 + 1.3417416468073389D+000i
-3.3675602004538763D+004 -3.2708073362471001D+004i	3.7152827529767691D+000 + 1.5071761457643933D+000i

APPENDIX B

SUM-OF-EXPONENTIALS WEIGHTS AND NODES FOR THE POWER FUNCTION $\frac{1}{T^{3/2}}$

We give the weights and nodes for the sum-of-exponential approximation for the power function $\frac{1}{t^{3/2}}$. $\frac{1}{t^{3/2}} \approx \sum_{k=1}^{N} w_k e^{s_k t}$. Here $N = 22$. The relative error is 10^{-9} for $t \in [10^{-3}, 1]$. In Table B.1, the first column lists the values of s_k and the second column lists the values of w_k.

Table B.1 Weights and Nodes for the Sum-of-exponentials Approximation for the Power Function $\frac{1}{t^{3/2}}$

s_k	w_k
-7.2906159549928551D-001	1.4185610815528382D+000
-3.0048118783719833D+000	6.1168393596966855D+000
-7.1264765899586058D+000	1.5711472927135111D+001
-1.3711947176886415D+001	3.4005632951162106D+001
-2.3888158541731844D+001	6.9073954353680932D+001
-3.9552395196025927D+001	1.3693587825572760D+002
-6.3705812577837150D+001	2.6803903557553036D+002
-1.0092380540293159D+002	5.1918775664556608D+002
-1.5806801843263241D+002	9.9539106627427793D+002
-2.4535153453758187D+002	1.8891680379043360D+003
-3.7789560121839855D+002	3.5504717885763807D+003
-5.7797736857567725D+002	6.6101535364473611D+003
-8.7825970527795960D+002	1.2196709015044245D+004
-1.3264302774624159D+003	2.2315687874782194D+004
-1.9919031777954062D+003	4.0515943857625483D+004
-2.9756729573498451D+003	7.3077757126301352D+004
-4.4253961362173968D+003	1.3121449467706907D+005
-6.5602925574528726D+003	2.3547202449444451D+005
-9.7174072351500563D+003	4.2563843229579012D+005
-1.4450262425240786D+004	7.8639206472664094D+005
-2.1750443538417538D+004	1.5127774335123657D+006
-3.3020803685636522D+004	2.6486986425576657D+006

APPENDIX C

SUM-OF-EXPONENTIALS WEIGHTS AND NODES FOR THE POWER FUNCTION $\frac{1}{T^2}$

We give the weights and nodes for the sum-of-exponentials approximation for the power function $\frac{1}{t^2}$. $\frac{1}{t^2} \approx \sum_{k=1}^{N} w_k e^{s_k t}$. Here $N = 44$. The relative error is 10^{-13} for $t \in [10^{-3}, 1]$. In Table C.1, he first column lists the values of s_k and the second column lists the values of w_k.

Table C.1 Weights and Nodes for Sum-of-exponentials Approximation for the Power Function $\frac{1}{t^2}$

s_k	w_k
0.49816097852842284D+00	0.41792329598969274D+00
0.16806708242710908D+01	0.25719203841494926D+01
0.35700113350143514D+01	0.80519010424116875D+01
0.62085182228220503D+01	0.18821621650994373D+02
0.96562475448359066D+01	0.37436959522664253D+02
0.13991619685598669D+02	0.67296131023505851D+02
0.19309793473336615D+02	0.11285259165079646D+03
0.25715313814472559D+02	0.17957517576121185D+03
0.33303378796019885D+02	0.27315383201452903D+03
0.42121854797631435D+02	0.39703175011879631D+03
0.52106071142751006D+02	0.54707777032875265D+03
0.62984741623253626D+02	0.70304122728005905D+03
0.74174333691048986D+02	0.82007948871433530D+03
0.84713745902110603D+02	0.83065511203111600D+03
0.93325793081081102D+02	0.67093386655665483D+03
0.98679955988816687D+02	0.33100272862403295D+03
0.10199025144253839D+03	0.52225373626928115D+03
0.11060150326402565D+03	0.13473849887749343D+04
0.12657537623995276D+03	0.25200712706100153D+04
0.15078496058776398D+03	0.43323174394854796D+04
0.18457111644199449D+03	0.72265575565852496D+04
0.22981916123749966D+03	0.11895748493955531D+05
0.28905196478122900D+03	0.19438296284932647D+05
0.36555007134723945D+03	0.31599396998625918D+05
0.46350713949880992D+03	0.51148389314505665D+05
0.58822867300627013D+03	0.82468749135566322D+05
0.74638314253071064D+03	0.13247930955776412D+06
0.94631702749702401D+03	0.21207060224239426D+06
0.11984490699616354D+04	0.33834255087831256D+06
0.15157646626332562D+04	0.53809202744656312D+06
0.19144400634407839D+04	0.85326116343166889D+06
0.24146405441750503D+04	0.13494938898270358D+07
0.30415614681794937D+04	0.21297038566261446D+07
0.38268260827182398D+04	0.33559325502733039D+07
0.48104374194145830D+04	0.52854357377097402D+07
0.60436434932699267D+04	0.83324354655784611D+07
0.75934011812903973D+04	0.13179211949340181D+08
0.95498164632420521D+04	0.20989730727789059D+08
0.12039510889823023D+05	0.33857166768530093D+08
0.15251770298672704D+05	0.55848629821905211D+08
0.19495260483786209D+05	0.95807184891746387D+08
0.25338803210820472D+05	0.17639430143426201D+09
0.34030470435410687D+05	0.37157326630489612D+09
0.48885715098457906D+05	0.97155900985648072D+09

BIBLIOGRAPHY

[1] B. K. Alpert. Hybrid Gauss-trapezoidal quadrature rules. *SIAM J. Sci. Comput.*, 20(5):1551–1584, 1999.

[2] B. K. Alpert, L. Greengard, and T. Hagstrom. Rapid evaluation of nonreflecting boundary kernels for time-domain wave propagation. *SIAM J. Numer. Anal.*, 37(4):1138–1164, 2000.

[3] B. K. Alpert, L. Greengard, and T. Hagstrom. Nonreflecting boundary conditions for the time-dependent wave equation. *J. Comput. Phys.*, 180(1):270–296, 2002.

[4] F. Aurenhammer. Voronoi diagrams–a survey of a fundamental geometric data structure. *ACM Comput. Surv.*, 23(3):345–405, 1991.

[5] G. Beylkin and L. Monzón. On generalized Gaussian quadrature for exponentials and their applications. *Appl. Comput. Harmon. Anal.*, 12:332–373, 2002.

[6] G. Beylkin and L. Monzón. On approximation of functions by exponential sums. *Appl. Comput. Harmon. Anal.*, 19:17–48, 2005.

[7] G. Beylkin and L. Monzón. Approximation by exponential sums revisited. *Appl. Comput. Harmon. Anal.*, 28:131–140, 2010.

[8] D. Braess and W. Hackbusch. Approximation of $1/x$ by exponential sums in $[1, \infty)$. *IMA J. Numer. Anal.*, 25(4):685–697, 2005.

[9] D. Braess and W. Hackbusch. *Multiscale, nonlinear and adaptive approximation.* Springer-Verlag, Berlin, Germany, 2009.

[10] K. Brattkus and D. I. Meiron. Numerical simulations of unsteady crystal growth. *SIAM J. Appl. Math.*, 52:1303–1320, 1992.

[11] C. A. Brebbia, editor. *Topics in Boundary Element Research*, volume 1. Springer-Verlag, Berlin, Germany, 1981.

[12] J. Bremer. A fast direct solver for the integral equations of scattering theory on planar curves with corners. *J. Comput. Phys.*, 231(4):1879–1899, 2012.

[13] J. Bremer, Z. Gimbutas, and V. Rokhlin. A nonlinear optimization procedure for generalized Gaussian quadratures. *SIAM J. Sci. Comput.*, 32(4):1761–1788, 2010.

[14] S. Chandrasekaran, P. Dewilde, W. Lyons M. Gu, and T. Pals. A fast solver for HSS representations via sparse matrices. *SIAM J. Matrix Anal. Appl.*, 29:67–81, 2006.

[15] R. Chapko and R. Kress. Rothe's method for the heat equation and boundary integral equations. *J. Integral Equations Appl.*, 9(1):47–69, 1997.

[16] H. Cheng, Z. Gimbutas, P. G. Martinsson, and V. Rokhlin. On the compression of low rank matrices. *SIAM J. Sci. Comput.*, 26:1389–1404, 2005.

[17] H. Cheng, L. Greengard, and V. Rokhlin. A fast adaptive multipole algorithm in three dimensions. *J. Comput. Phys.*, 155(2):468–498, 1999.

[18] J. W. Cooley and J. W. Tukey. An algorithm for the machine calculation of complex Fourier series. *Math. Comput.*, 19:297–301, 1965.

[19] E. Corona, P. G. Martinsson, and D. Zorin. An $O(N)$ direct solver for integral equations on the plane. *Appl. Comput. Harmon. Anal.*, 38(2):284–317, 2015.

[20] M. Costabel. *Time-dependent problems with the boundary integral equation method.* John Wiley, 2004.

[21] A. Dutt and V. Rokhlin. Fast Fourier transforms for nonequispaced data. *SIAM J. Sci. Comput.*, 14:1368–1393, 1993.

[22] A. Gillman, P. Young, and P. G. Martinsson. A direct solver with $O(N)$ complexity for integral equations on one-dimensional domains. *Front. Math. China*, 7(2):217–247, 2012.

[23] Z. Gimbutas and V. Rokhlin. A generalized fast multipole method for nonoscillatory kernels. *SIAM J. Sci. Comput.*, 24:796–817, 2003.

[24] Z. Gimbutas and S. Veerapaneni. A fast algorithm for spherical grid rotations and its application to singular quadrature. *SIAM J. Sci. Comput.*, 35(6):2738–2751, 2013.

[25] I. G. Graham and I. H. Sloan. Fully discrete spectral boundary integral methods for Helmholtz problems on smooth closed surfaces in $\mathbb{R}^3$. *Numer. Math.*, 92:289–323, 2002.

[26] L. Greengard, D. Gueyffier, P. G. Martinsson, and V. Rokhlin. Fast direct solvers for integral equations in complex three-dimensional domains. *Acta. Numer.*, 18:243–275, 2009.

[27] L. Greengard and J.Y. Lee. Accelerating the nonuniform fast Fourier transform. *SIAM Rev.*, 46:443–454, 2004.

[28] L. Greengard and P. Lin. Spectral approximation of the free-space heat kernel. *Appl. Comput. Harmon. Anal.*, 9:83–97, 2000.

[29] L. Greengard and V. Rokhlin. A fast algorithm for particle simulations. *J. Comp. Phys.*, 73(2):325–348, 1987.

[30] L. Greengard and V. Rokhlin. A new version of the fast multipole method for the Laplace equation in three dimensions. *Acta. Numer.*, 6:229–270, 1997.

[31] L. Greengard and J. Strain. A fast algorithm for the evaluation of heat potentials. *Comm. Pure Appl. Math.*, 43:949–963, 1990.

[32] R. B. Guenther and J. W. Lee. *Partial Differential Equations of Mathematical Physics and Integral Equations*. Prentice-Hall, Englewood Cliffs, New Jersey, 1988.

[33] W. Hackbusch and S. Börm. Data-sparse approximation by adaptive H2-matrices. *Computing*, 69(1):1–35, 2002.

[34] T. Hagstrom. Radiation boundary conditions for the numerical simulation of waves. *Acta Numerica*, 8:47–106, 1999.

[35] K. L. Ho and L. Greengard. A fast direct solver for structured linear systems by recursive skeletonization. *SIAM J. Sci. Comput.*, 34(5):A2507–A2532, 2012.

[36] K. L. Ho and L. Ying. Hierarchical interpolative factorization for elliptic operators: integral equations. *Comm. Pure Appl. Math.*, 69(7):1314–1353, 2016.

[37] M. T. Ibanez and H. Power. An efficient direct BEM numerical scheme for phase change problems using Fourier series. *Comput. Methods Appl. Mech. Engrg.*, 191:2371–2402, 2002.

[38] S. Jiang and L. Greengard. Fast evaluation of nonreflecting boundary conditions for the Schrödinger equation in one dimension. *Comput. Math. Appl.*, 47(6-7):955–966, 2004.

[39] S. Jiang and L. Greengard. Efficient representation of nonreflecting boundary conditions for the time-dependent Schrödinger equation in two dimensions. *Comm. Pure Appl. Math.*, 61(2):261–288, 2008.

[40] S. Jiang, S. Veerapaneni, and L. Greengard. Integral equation methods for unsteady Stokes flow in two dimensions. *SIAM J. Sci. Comput.*, 34(4):A2197–A2219, 2012.

[41] A. Kloeckner, A. Barnett, L. Greengard, and M. O'Neil. Quadrature by expansion: A new method for the evaluation of layer potentials. *J. Comput. Phys*, 252:332–349, 2013.

[42] W. Y. Kong, J. Bremer, and V. Rokhlin. An adaptive fast direct solver for boundary integral equations in two dimensions. *Appl. Comput. Harmon. Anal.*, 31(3):346–369, 2011.

[43] J.Y. Lee and L. Greengard. The type 3 nonuniform FFT and its applications. *J. Comput. Phys.*, 206:1–5, 2005.

[44] J. Li and L. Greengard. High order accurate methods for the evaluation of layer heat potentials. *SIAM J. Sci. Comput.*, 31:3847–3860, 2009.

[45] E. Liberty, F. Woolfe, P.-G. Martinsson, V. Rokhlin, and M. Tygert. Randomized algorithms for the low-rank approximation of matrices. *Proc. Natl. Acad. Sci.*, 104:20167–20172, 2007.

[46] M. López-Fernández and C. Palencia. On the numerical inversion of the Laplace transform of certain holomorphic mappings. *Appl. Numer. Math.*, 53:289–303, 2004.

[47] M. López-Fernández, C. Palencia, and A. Schädle. A spectral order method for inverting sectorial Laplace transforms. *SIAM J. Numer. Anal.*, 44(3):1332–1350, 2006.

[48] C. Lubich and A. Schädle. Fast convolution for nonreflecting boundary conditions. *SIAM J. Sci. Comput.*, 24:161–182, 2002.

[49] J. Ma, V. Rokhlin, and S. Wandzura. Generalized Gaussian quadrature rules for systems of arbitrary functions. *SIAM J. Numer. Anal.*, 33(3):971–996, 1996.

[50] P. G. Martinsson. A fast direct solver for a class of elliptic partial differential equations. *J. Sci. Comput.*, 38(3):316–330, 2009.

[51] P. G. Martinsson and V. Rokhlin. A fast direct solver for boundary integral equations in two dimensions. *J. Comput. Phys.*, 205(1):1–23, 2005.

[52] P. G. Martinsson and V. Rokhlin. An accelerated kernel-independent fast multipole method in one dimension. *SIAM J. Sci. Comput.*, 29(3):1160–1178, 2007.

[53] P. G. Martinsson and V. Rokhlin. A fast direct solver for scattering problems involving elongated structures. *J. Comput. Phys.*, 221(1):288–302, 2007.

[54] E. A. McIntyre. Boundary integral solutions of the heat equation. *Math. Comp.*, 46(173):71–79, 1986.

[55] L. Morino and R. Piva, editors. *Boundary Integral Methods: Theory and Applications.* Springer-Verlag, Berlin, Germany, 1990.

[56] P. J. Noon. *The single layer heat potential and Galerkin boundary element methods for the heat equation.* PhD thesis, The University of Maryland, 1988.

[57] X. M. Pan, J. G. Wei, Z. Peng, and X. Q. Sheng. A fast algorithm for multiscale electromagnetic problems using interpolative decomposition and multilevel fast multipole algorithm. *Radio Sci.*, 47, 2012.

[58] J. A. Sethian and J. Strain. Crystal growth and dendritic solidification. *J. Comput. Phys.*, 98:231–253, 1992.

[59] F. Stenger. Approximation via Whitaker's Cardinal functions. *J. Approx. Theory*, 17:222–240, 1976.

[60] F. Stenger. Numerical methods based on Whitaker Cardinal, or sinc functions. *SIAM Rev.*, 23:165–224, 1981.

[61] A. Talbot. The accurate numerical inversion of Laplace transforms. *J. Inst. Math. Appl*, 23:97–120, 1979.

[62] J. Tausch. A fast method for solving the heat equation by layer potentials. *J. Comput. Phys*, 224(2):956–969, 2007.

[63] L. N. Trefethen, J. A. C. Weideman, and T. Schmelzer. Talbot quadratures and rational approximations. *BIT Numer. Math.*, 46:653–670, 2006.

[64] S. K. Veerapaneni, A. Rahimian, G. Biros, and D. Zorin. Fast algorithm for simulating vesicle flows in three dimensions. *J. Comput. Phys.*, 230:5610–5634, 2011.

[65] F. Woolfe, E. Liberty, V. Rokhlin, and M. Tygert. A fast randomized algorithm for the approximation of matrices. *Appl. Comput. Harmon. Anal.*, 25:335–366, 2008.

[66] K. Xu and S. Jiang. A bootstrap method for sum-of-poles approximations. *J. Sci. Comput.*, 55:16–39, 2013.

[67] N. Yarvin and V. Rokhlin. An improved fast multipole algorithm for potential fields on the line. *SIAM J. Numer. Anal.*, 36(2):629–666, 1999.

[68] L. Ying, G. Biros, and D. Zorin. A kernel-independent adaptive fast multipole algorithm in two and three dimensions. *J. Comput. Phys.*, 196:591–626, 2004.