Cent. Eur. J. Math. • 10(1) • 2012 • 204-216
DOI: 10.2478/s11533-011-0103-2

VERSITA

Central European Journal of **Mathematics**

# A brief review of some application driven fast algorithms for elliptic partial differential equations

**Review Article**

Prabir Daripa[1]*

1 *Department of Mathematics, Texas A&M University, College Station, TX-77843, USA*

**Received 9 June 2011; accepted 26 August 2011**

**Abstract:** Some application driven fast algorithms developed by the author and his collaborators for elliptic partial differential equations are briefly reviewed here. Subsequent use of the ideas behind development of these algorithms for further development of other algorithms some of which are currently in progress is briefly mentioned. Serial and parallel implementation of these algorithms and their applications to some pure and applied problems are also briefly reviewed.

**MSC:** 35J05, 65D99, 65E05, 65Y20

**Keywords:** Fast algorithms • Numerical methods • Elliptic equations



## 1. Introduction

Computational resources such as fast algorithms, fast processors and multi-processor computers together provide an unprecedented opportunity to solve many large scale problems of fundamental, industrial, defense and security interests that were impossible previously. Moreover, each of these computational resources on its own has the potential to provide solutions at levels of accuracy and details that were hitherto unimaginable. Such solutions allow scientists to better understand many fundamental processes, engineers to better design many processes; e.g., design of aircrafts, cars, turbines, catheters, medical imaging tools, security related finger-print and face recognition tools, unidentified object detection tools for exploration, medical and security purposes. Realization of all these exciting possibilities requires research and training of personnel in many areas including applied and computational mathematics, fast methods, and scientific computation. Over last few decades of extensive dedicated research efforts in this direction by many have resulted in various methods and algorithms producing solutions of varying accuracy and details. However, overall

* *E-mail: daripa@math.tamu.edu*

principle of all these methods hinges on the following concept: discrete representation of a domain in which the solution is sought and some sort of discrete approximation of the differential or equivalent integral operators which involves approximate representation of solution of the partial differential equations. The systems of nonlinear discrete equations arising from this process are solved these days by a variety of direct and iterative methods whose choice is dictated by many factors including accuracy, resolution, computational complexity, type and number of processors, ease of implementation, and so on. Consideration of these and many other steps laid out in a logical fashion for the purpose of encapsulation into programs lead to algorithms. Implementation of these algorithms generate software tools of a wide ranging capabilities: from simple to modest to advanced.

The fast algorithms and associated modular serial and parallel software that we have developed so far can currently solve **separable** elliptic problems involving constant coefficient operators in two-dimension [3, 6, 7, 9, 10, 12, 13]. However, in problems where the constant coefficient or the inhomogeneous terms have highly varying local features or multi-scale features, high order accurate and modified versions of the fast algorithms that use non-uniform grids are needed. Such modifications are highly non-trivial because our existing algorithms are based on analysis on uniform grid. Extending the **analysis** behind our fast algorithm from uniform grid to non-uniform grid is a topic of current research. Also extending the analysis to the case of **non-separable** equations is a real challenge and success here will have impact in many areas.

Our analysis based fast algorithms described in the next section are different from many numerical algorithms that result from the discretization of integral representation of solutions of such elliptic partial differential equations (see [15–21]). Below, we briefly review some of our recent studies on the development of analysis based fast and accurate algorithms and their applications to some areas.

## 2. Application area driven need of an accurate algorithm

During the course of solving some applied problems, it is not so uncommon at times to realize that accurate, reliable and efficient algorithms are needed but not available. This creates the need for the development of application driven fast algorithms. Consider the application area of inverse design of airfoils (the cross section of the wing of an aircraft) in compressible flows from the pressure data on the as yet unknown airfoil surface as a function of the arclength parameter. This is a classic problem which comes up in aerospace industry. In Daripa [8] a new method, which is very accurate and very fast, for solving this problem was proposed for the first time. Some details on this follow.

The appropriate equations of fluid flow for this problem are given by the following nonlinear system of partial differential equations:

$$\nabla \cdot (\rho \vec{q}) = 0, \qquad \nabla \times \vec{q} = 0, \qquad p = C\rho^{\gamma}.$$

The variables $\vec{q}$, $\rho$ and $p$ are respectively velocity, density and pressure. However, this system is not suitable for inverse design and requires some ingenious techniques. In Daripa [8], it was shown that for design purpose a hierarchy of transformations of these equations involving different variables and domains is needed. This process transforms the above system into the following Beltrami equation in the unit disk in complex plane parameterized by $\sigma$.

$$\tau_{\bar{\sigma}} = \chi \tau_{\sigma}, \qquad \chi = \frac{\mu(\overline{\omega_{\sigma}})}{\omega_{\sigma}}.$$

This formulation turns out to be the most natural and suitable formulation for the inverse design of subcritical airfoils. Here the complex unknown $\tau$ is related to the speed and the flow angle, $\sigma$ is the complex plane and $\mu$ depends on Mach number and so on. In Daripa [8], a fast method for solving the above Beltrami equation was proposed by writing the solution in terms of Green's function of the operator. To better expose the challenges, consider the following simpler problem of solving $u_{\bar{z}} = f$, where $f$ satisfies a Hölder condition with exponent $\alpha$ in a disk $\Omega$: $|z| < 1$. It is well known, see [8], that a solution procedure for this equation involves Cauchy transform $T_1 f$ and Beurling transform $T_2 f$ which are respectively defined by

$$T_1 f(z) = -\frac{1}{\pi} \iint_{\Omega} \frac{f(\zeta)}{\zeta - z}\, d\xi\, d\eta, \qquad T_2 f(z) = -\frac{1}{\pi} \iint_{\Omega} \frac{f(\zeta)}{(\zeta - z)^2}\, d\xi\, d\eta, \tag{1}$$

where $\zeta = \xi + i\eta$. The above two integrals have singular kernels. Accurate and efficient evaluation of these integrals is at the heart of the problem of developing fast and accurate methods for solving the inverse problem. In [8], these integrals were evaluated by quadrature rules which were required in the design process.

# 3. Application driven birth of an accurate and fast algorithm

The recognition that the integral transforms $(1)_1$ and $(1)_2$, even though singular, are of convolution type and the functions in the kernel are periodic in the azimuthal direction allows analysis of these transforms using theory of single complex variable. At the end of this analysis, a nice beautiful easy-to-use accurate, fast, and inherently parallelizable algorithm for fast evaluation of the singular integrals arises. This analysis is briefly presented below; see Daripa [9].

## 3.1. Mathematical foundation of the algorithm

In this section we outline the theory needed to construct an efficient and accurate algorithm for evaluation of these singular integrals [10]. Below, we discuss the evaluation of Cauchy transform $T_1$ only because evaluation of the Beurling transform $T_2$ is similar and will not be discussed here. In the following we use the notations $\Omega_r : |\sigma| \le r < 1$, $\Omega_{\bar{r}} : \Omega \setminus \Omega_r$, and $\Omega_{ij} : r_i \le |\sigma| \le r_j$. The following theorem is crucial for the later development and evolution of the algorithm.

### Theorem 3.1.

*The particular solution of $u_{\bar{\sigma}} = f(\sigma)$ is the Cauchy transform $T_1 f(\sigma)$ of $f$. With $\sigma = re^{i\alpha}$, the Cauchy transform $T_1(\sigma)$ can be written as*

$$T_1 f(\sigma) = \sum_{n=-\infty}^{\infty} c_n(r) e^{in\alpha} \tag{2}$$

*where*

$$c_n(r) = \begin{cases} \dfrac{1}{\pi} \displaystyle\iint_{\Omega_r} \frac{f(\zeta)}{\zeta} \left(\frac{r}{\zeta}\right)^n d\xi\, d\eta & \text{if } n < 0, \\ -\dfrac{1}{\pi} \displaystyle\iint_{\Omega_{\bar{r}}} \frac{f(\zeta)}{\zeta} \left(\frac{r}{\zeta}\right)^n d\xi\, d\eta & \text{if } n \ge 0. \end{cases} \tag{3}$$

The proof of this theorem can be found in Daripa [10]. The fast algorithm makes use of the following corollaries of this theorem.

### Corollary 3.2.

*Suppose that $\zeta = \rho e^{i\theta}$ and $f(\zeta)$ has Fourier coefficients $f_n(\rho)$, then it easily follows that the coefficients $c_n(r)$ in (3) can be written as*

$$c_n(r) = \begin{cases} 2 \displaystyle\int_0^r f_{n+1}(\rho) \left(\frac{r}{\rho}\right)^n d\rho & \text{if } n < 0, \\ -2 \displaystyle\int_r^1 f_{n+1}(\rho) \left(\frac{r}{\rho}\right)^n d\rho & \text{if } n \ge 0. \end{cases} \tag{4}$$

### Corollary 3.3.

*It follows directly from (4) that $c_n(1) = 0$ for $n \ge 0$, $c_n(0) = 0$ for all $n \ne 0$. It also follows from the Fourier expansion of $f(\zeta)$ that $f_n(0) = 0$ for $n \ne 0$, and $f_0(0) = f(0)$.*

### Corollary 3.4.

*Let $r_j > r_i$. Define*

$$c_n^{ij} = 2 \int_{r_i}^{r_j} f_{n+1}(\rho) \left(\frac{R}{\rho}\right)^n d\rho, \tag{5}$$

*where*

$$R = \begin{cases} r_i & \text{if } n \geq 0, \\ r_j & \text{if } n < 0. \end{cases}$$

*After some algebraic manipulation it follows*

$$c_n(r_j) = \left(\frac{r_j}{r_i}\right)^n c_n(r_i) + c_n^{ji}, \qquad n < 0, \tag{6}$$

*and*

$$c_n(r_i) = \left(\frac{r_i}{r_j}\right)^n c_n(r_j) - c_n^{ij}, \qquad n \geq 0. \tag{7}$$

### Corollary 3.5.

*Let* $0 = r_1 < r_2 < r_3 < \cdots < r_M = 1$. *It follows from recursive applications of* (6) *and* (7) *and from using Corollary* 3.3 *that*

$$c_n(r_l) = \begin{cases} \displaystyle\sum_{i=2}^{l} \left(\frac{r_l}{r_i}\right)^n c_n^{i-1,i} & \text{for } n < 0 \text{ and } l = 2, \ldots, M, \\ \displaystyle -\sum_{i=l}^{M-1} \left(\frac{r_l}{r_i}\right)^n c_n^{i,i+1} & \text{for } n \geq 0 \text{ and } l = 1, \ldots, M-1. \end{cases}$$

## 3.2. The fast algorithm

We construct the fast algorithm based on the theory given above. The unit disk is discretized using $M \times N$ lattice points with $M$ equidistant points in the radial direction and $N$ equidistant points in the circular direction.

#### Initialization

Choose $M$ and $N$. Define $K = N/2$.

#### Step 1

For $l \in [1, M]$ and $n \in [-K+1, K]$, compute the Fourier coefficients $f_n(r_l)$ of $f(\zeta)$ from the known values of $f\left(\zeta = r_j e^{2\pi i k/N}\right)$, $j = 1, \ldots, M$, $k = 1, \ldots, N$.

#### Step 2

Compute $c_n^{i,i+1}$, $i \in [1, M-1]$, for $n \in [-K, K-1]$ using (5).

#### Step 3

Compute the Fourier coefficients $c_n(r_l)$ $n \in [-K, K-1]$, $l \in [1, M]$ using the relations (6) and (7).

#### Step 4

Finally compute $u^p\left(\sigma = r_j e^{2\pi i k/N}\right)$, $j \in [1, M]$, $k \in [1, N]$, using a truncated version of (2).

Figure 1 shows the grid and complexity of the algorithm. This problem is also important because of the following reasons.

$$f(r_j) \xrightarrow[\text{FFT}]{\mathcal{O}(MN\log_2 N)} f_n(r_j) \xrightarrow[C_n^{i,i+1}]{\mathcal{O}(MN)} C_n(r_j) \xrightarrow[\text{inverse FFT}]{\mathcal{O}(MN\log_2 N)} T_1 f(r_j)$$

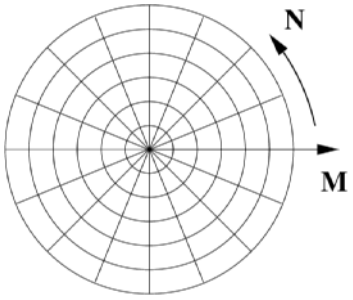


**Figure 1.** Grid for the fast algorithm

- Complexity: $\mathcal{O}(MN \log_2 N)$.
- $\mathcal{O}\left(N^2 \log_2 N\right)$ for $N^2$ points implies $\mathcal{O}(\log_2 N)$ per point.

We see that this algorithm is based on some recursive relations in Fourier space together with FFT (fast Fourier transform). The resulting method has theoretical computational complexity $\mathcal{O}\left(N^2 \log_2 N\right)$ or equivalently $\mathcal{O}(\log_2 N)$ per point which represents substantial savings in computational time when compared with quadrature rules. Furthermore, results are more accurate because the algorithm is based on exact analysis. Additionally, recursive nature of the evaluation of the coefficients above offers good parallelization opportunities and a lower computational complexity when compared with methods based on quadrature rules. In Borges and Daripa [6], we have developed a parallel version of the fast algorithm by redefining the inherently sequential recurrences present in the original sequential formulation. The parallel version utilizes only a linear neighbor-to-neighbor communication path which makes the algorithm very suitable for any distributed memory architecture. Numerical results and theoretical estimates reported there show good parallel scalability of the algorithm.

In Daripa and Mashat [12], the above fast algorithm has been extended for arbitrary operator $T_m$ ($m$ an integer) within the unit disk with possibility of applications to hyper-singular integrals. This operator is defined as

$$T_m f(\sigma) = -\frac{1}{\pi} \iint\limits_{B(0;1)} \frac{f(\zeta)}{(\zeta - \sigma)^m}\, d\xi\, d\eta, \qquad \zeta = \xi + i\eta,$$

where $f$ is a complex valued function of $\sigma$ defined on $B(0;1) = \{z : |z| < 1\}$, for a suitable positive integer $m$.

## 4. Application, extension, transformation and evolution of the fast algorithm

These fast and accurate algorithms for these singular integral transforms have been encapsulated into software by Daripa and some of his past collaborators. These algorithms and the associated software have been validated on many test problems details of which can be found in Daripa [9]. The software has the scope of applications to a variety of two-dimensional problems. We have used/applied the concepts and the algorithms and associated software to following problems: (i) solving the Beltrami equation and quasi-conformal mapping problems (see Section 4.1); (ii) developing parallel algorithms (see Section 4.2); (iii) developing similar serial and parallel fast algorithms for solving separable elliptic problems in $\mathbb{R}^2$ in regular domains (see Section 4.3); (iv) applying to solving bio-fluid mechanics problems (see Section 4.4); (v) developing fast algorithms for solving separable elliptic problems in irregular domains in $\mathbb{R}^2$ (see Section 5). Other investigators have applied these ideas and our algorithms to solving inverse conductivity problems, see [14, 23], and two-dimensional density estimation using smooth invertible transformation [1] in $\mathbb{R}^2$.

## 4.1. Solving the Beltrami equation and quasiconformal mapping

The fast algorithms and the software for solving the Beltrami equation within unit disk have applications in a variety of areas as it is. In addition to solving the problem of inverse airfoil design which motivated the development, the algorithms and software have been applied to quasi-conformal mappings [9, 10, 12, 13] of simply and doubly connected domains. In both of these numerical methods, use of our algorithms for fast evaluation of Cauchy and Beurling transforms is essential. An iterative algorithm has been developed in Daripa [10] and Daripa & Mashat [13] for simply-connected and doubly-connected domains respectively. In these papers, an iterative algorithm which uses the fast algorithms for singular integrals in each iteration has been presented. We do not have space in this paper to present this complicated algorithm. In Figures 2 and 3, some quasi-conformal grids generated this way for simply and doubly connected domains are shown respectively. These figures are taken from [10] and Daripa & Mashat [13].

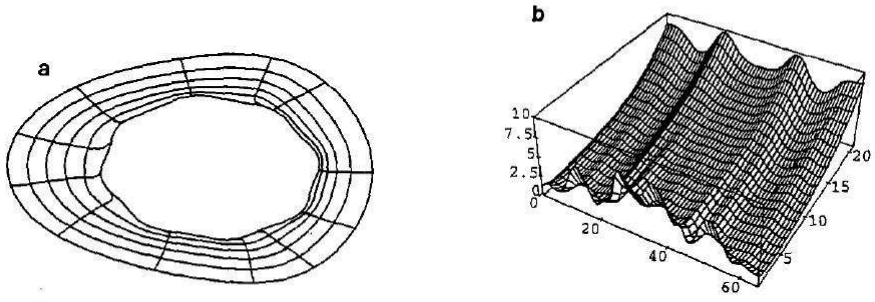


**Figure 2.** Quasi-conformal mapping of a simply connected domain: grids generated by quasi-conformal mapping around a perturbed circle; the figure on the right shows dilatation.

## 4.2. Parallel implementation

The algorithm presented in Section 4.1 above is not only fast but inherently parallel. The identification of intrinsic parallelism in the method leads to our choice for data partitioning [22]. Therefore, we have developed a fast parallel version of the algorithm of Section 4.2. The fast parallel algorithm employs two groups of Fourier transforms which can be evaluated independently for each fixed radius $r_l$. Consequently their computations can be performed in parallel. Since each FFT usually engages lengthy computations, the computational granularity of each processor will be large and therefore very well suited for MIMD architectures. Negative effects resulting from communication delays in a MIMD computer can be minimized by an efficient implementation. Mechanisms to reduce communication delays on message-passing architectures include evenly distributed load balancing between processors, overlapping of communication and computations, reduced message lengths, and reduced frequency in exchanging messages. Often the above mechanisms are conflicting and, in practice, a trade-off will define an efficient implementation. We have addressed these and many other issues in detail in Borges and Daripa [6]. We have analyzed the parallel algorithm also in detail there providing complexity of the stream-based parallel algorithm among many others. Below we show some self-explanatory figures that are discussed in detail in Borges and Daripa [6]. Figure 4 presents the message distribution and message passing strategy in the algorithm. Figure 5 summarizes the coordination scheme to minimize delays due to interprocessor communication. Figures 6(a) and 6(b) show the speedups of parallel implementation with various processors. For details, see Borges and Daripa [6].

## 4.3. Fast algorithms for separable elliptic equations in the real plane $\mathbb{R}^2$

Our discussion so far involved the fast algorithms for evaluation of singular integrals in the unit disk in the complex plane. This naturally involves theory of single complex variables in their derivation as well as in the choice of applied problems as we have seen above in the context of inverse design and quasi-conformal mapping. However, it turns out that an analogous analysis can be developed for fast evaluation of singular integrals in $\mathbb{R}^2$. Such singular integrals

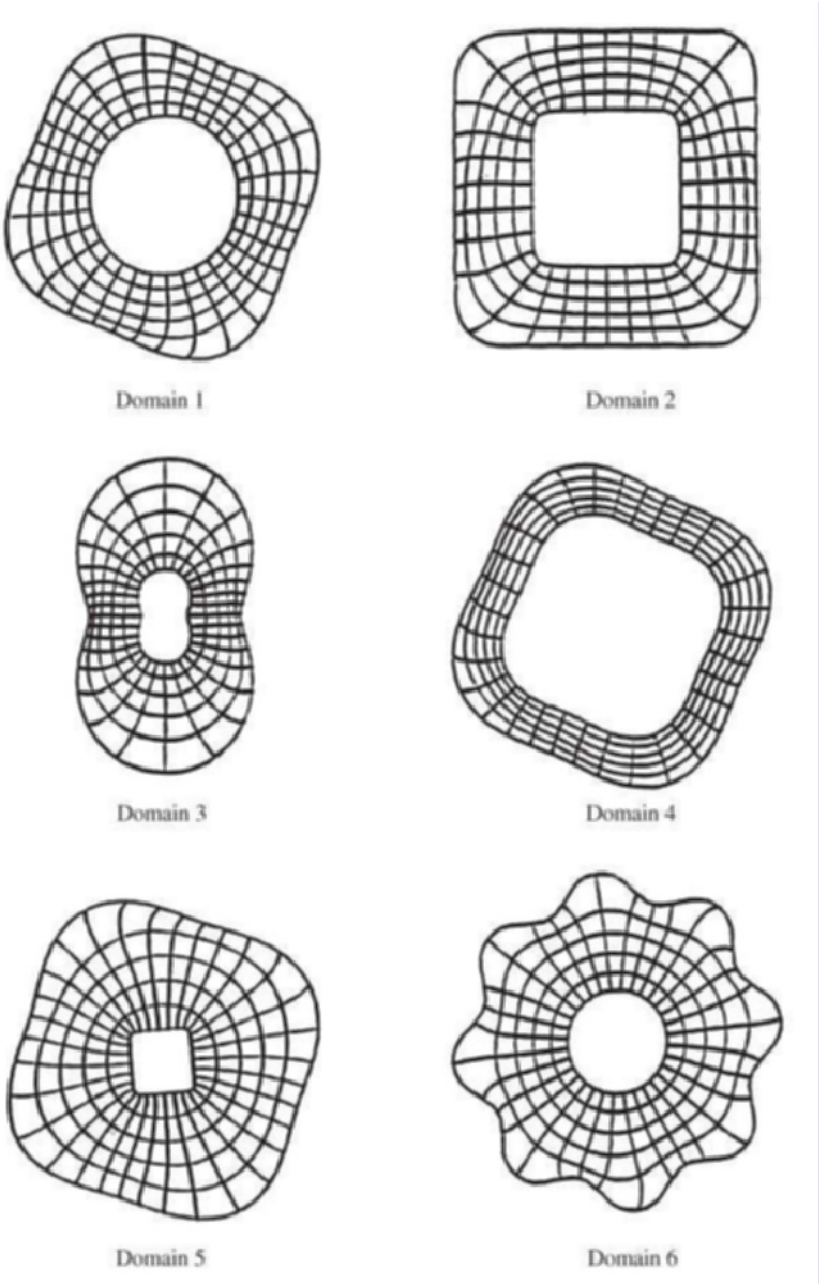


**Figure 3.** Quasi-conformal mapping of a doubly connected domain: grids generated by a quasi-conformal mapping of some doubly connected domains.

arise naturally in solving *separable* elliptic equations in $\mathbb{R}^2$ using Green's function approach. Therefore, similar fast algorithms have been developed for solving Dirichlet and Neumann problems involving the Poisson equation within disks and concentric annular regions; see Borges and Daripa [7]. We briefly outline below the algorithm from this paper.

Consider the Dirichlet problem for the Poisson equation

$$\begin{cases} \Delta u = f & \text{in } B, \\ u = g & \text{on } \partial B, \end{cases} \tag{8}$$

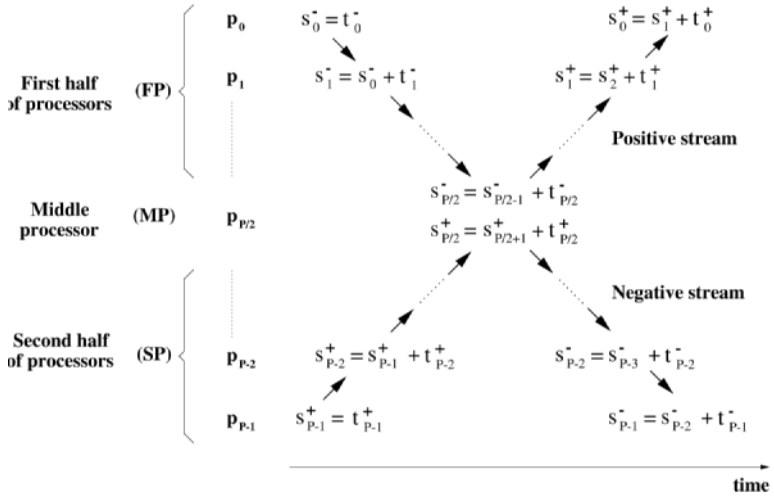


**Figure 4.** Message distribution in the algorithm. Two streams of neighbor-to-neighbor messages cross communication channels simultaneously.

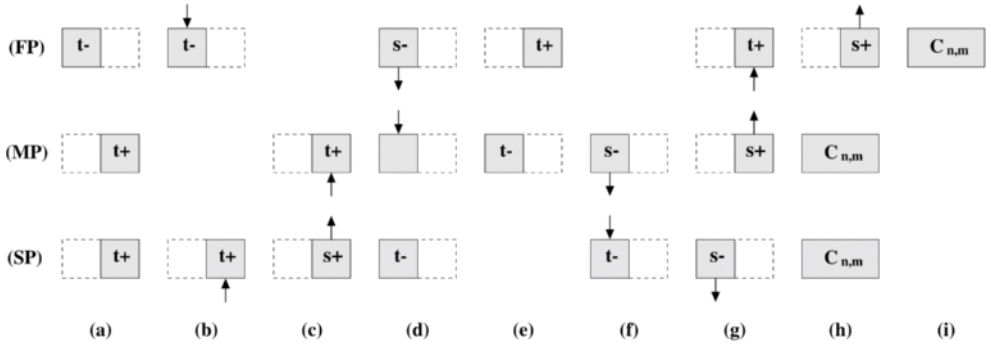


**Figure 5.** Coordination scheme to minimize delays due to interprocessor communication. The middle processor (MP) plays a key role to forward the positive stream to the first half of processors (FP) and to forward the negative stream to the second half of processors (SP).

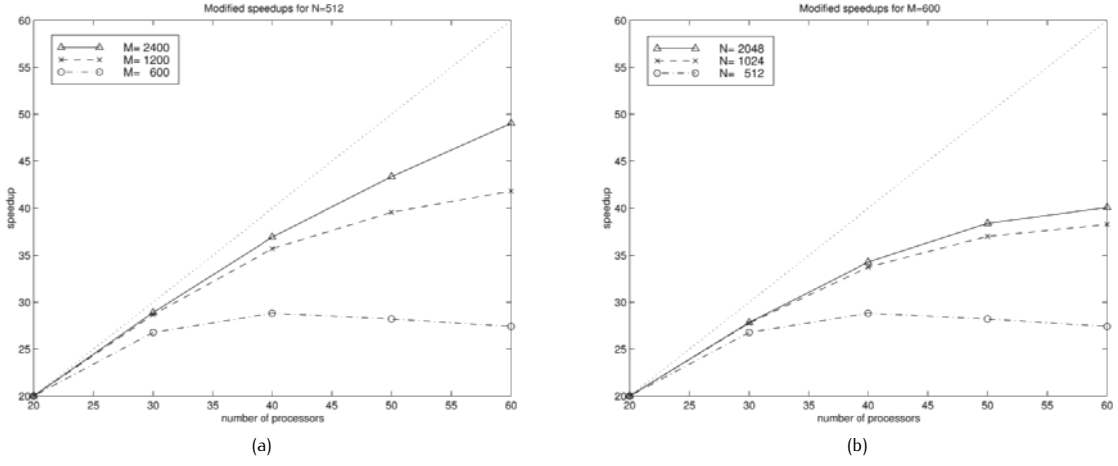


**Figure 6.** Modified speedups $S^{[20]}$ for 20, 30, 40, 50 and 60 processors: (a) for variable number of radial grid points $M = 600$, 1200 and 2400, with $N = 512$ fixed; (b) for variable number of angular grid points $N = 512$, 1024 and 2048, with $M = 600$ fixed.

where $B = B(0,R) = \{x \in \mathbb{R}^2 : |x| < R\}$. Specifically, let $v$ satisfy

$$\Delta v = f \qquad \text{in} \quad B, \tag{9}$$

and $w$ be the solution of the homogeneous problem

$$\begin{cases} \Delta w = 0 & \text{in } B, \\ w = g - v & \text{on } \partial B. \end{cases}$$

Thus, the solution of the Dirichlet problem (8) is given by

$$u = v + w.$$

A principal solution of (9) can be written as

$$v(x) = \int_B f(\eta)\, G(x,\eta)\, d\eta, \qquad x \in B, \tag{10}$$

where $G(x,\eta)$ is the free-space Green's function for the Laplacian given by

$$G(x,\eta) = \frac{1}{2\pi} \log |x - \eta|.$$

To derive a numerical method based on equation (10), the interior of the disk $B(0,R)$ is divided into a collection of annular regions. The use of quadrature rules to evaluate (10) incurs poor accuracy for the approximate solution. Moreover, the complexity of a quadrature method is $\mathcal{O}(N^4)$ for a $N^2$ net of grid points. For large problem sizes it represents prohibitive costs in computational time. Here we expand $v(\cdot)$ in terms of Fourier series by deriving radius dependent Fourier coefficients of $v(\cdot)$. These Fourier coefficients can be obtained by recursive relations which utilize only one-dimensional integrals in the radial direction. The fast algorithm is embedded in the following theorem:

**Theorem 4.1.**
*If $u(r,\alpha)$ is the solution of the Dirichlet problem (8) for $x = re^{i\alpha}$ and*

$$f(re^{i\alpha}) = \sum_{n=-\infty}^{\infty} f_n(r) e^{in\alpha},$$

*then the nth Fourier coefficient $u_n(r)$ of $u(r,\cdot)$ can be written as*

$$u_n(r) = v_n(r) + \left(\frac{r}{R}\right)^{|n|} (g_n - v_n(R)), \qquad 0 < r < R,$$

*where $g_n$ are the Fourier coefficients of $g$ on $\partial B$, and*

$$v_n(r) = \int_0^r p_n(r,\rho)\, d\rho + \int_r^R q_n(r,\rho)\, d\rho, \tag{11}$$

*with*

$$p_n(r,\rho) = \begin{cases} \rho f_0(\rho) \log r & \text{for } n = 0, \\ \dfrac{-\rho}{2|n|} \left(\dfrac{\rho}{r}\right)^{|n|} f_n(\rho) & \text{for } n \neq 0, \end{cases}$$

*and*

$$q_n(r,\rho) = \begin{cases} \rho f_0(\rho) \log \rho & \text{for } n = 0, \\ \dfrac{-\rho}{2|n|} \left(\dfrac{r}{\rho}\right)^{|n|} f_n(\rho) & \text{for } n \neq 0. \end{cases} \tag{12}$$

## Recursive relations of the algorithm

Despite the fact that the above theorem presents a mathematical foundation of the algorithm, an efficient implementation can be devised by making use of recursive relations to perform the integrations in (11). Consider the disk $\overline{B(0,R)}$ discretized by $N \times M$ lattice points with $N$ equidistant points in the angular direction and $M$ distinct points in the radial direction. Let $0 = r_1 < r_2 < \ldots < r_M = R$ be the radii defined on the discretization. Theorem 4.1 leads to the following corollaries:

### *Corollary 4.2.*

*It follows from* (11) *and* (12) *that* $v_n(0) = 0$ *for* $n \neq 0$.

### *Corollary 4.3.*

*Let* $0 = r_1 < r_2 < \ldots < r_M = R$, *and*

$$C_n^{i,j} = \int_{r_i}^{r_j} \frac{\rho}{2n} \left(\frac{r_j}{\rho}\right)^n f_n(\rho)\, d\rho, \qquad n < 0,$$
$$D_n^{i,j} = -\int_{r_i}^{r_j} \frac{\rho}{2n} \left(\frac{r_i}{\rho}\right)^n f_n(\rho)\, d\rho, \qquad n > 0.$$

*If* $r_j > r_i$, *we define*

$$\begin{cases} v_n^-(r_1) = 0 & \text{for } n < 0, \\ v_n^-(r_j) = \left(\dfrac{r_j}{r_i}\right)^n v_n^-(r_i) + C_n^{i,j} & \text{for } n < 0, \end{cases} \tag{13}$$

*and*

$$\begin{cases} v_n^+(r_M) = 0 & \text{for } n > 0, \\ v_n^+(r_i) = \left(\dfrac{r_i}{r_j}\right)^n v_n^+(r_j) + D_n^{i,j} & \text{for } n > 0, \end{cases} \tag{14}$$

*then for* $i = 1, \ldots, M$ *we have*

$$v_n(r_i) = \begin{cases} v_n^-(r_i) + \overline{v_{-n}^+(r_i)} & \text{if } n < 0, \\ v_n^+(r_i) + \overline{v_{-n}^-(r_i)} & \text{if } n > 0. \end{cases} \tag{15}$$

## Some of the features of the algorithm

- Our fast method presented above is a hybrid method: Fourier in one direction and recursive relations in the other direction. This feature leads to a natural parallel algorithm with favorable complexity.
- It is important to emphasize that $M$ distinct points $r_1, \ldots, r_M$ need not be equidistant. Therefore, the fast algorithm can be applied without any difficulty on a grid.
- Complexity of the algorithm scales almost linearly with the number of discretization points in the domain.
- Values of the Fourier coefficients $v_n(r_i)$ (see (15)) on any ring $r = r_i$ depend only on the values of $v_n(r_l)$, $r_l < r_i$, for $n < 0$ and only on the values of $v_n(r_l)$, $r_l > r_i$, for $n > 0$. Therefore, one has to compute only half the total number of Fourier coefficients on rings inside and outside of the ring $r = r_i$ to evaluate the values of all Fourier coefficients on the ring $r = r_i$. This constitutes considerable savings on computation when the interest is in finding solution of the problem on a ring or certain number of points on a ring. The asymptotic complexity in this case remains the same but the value of the constant hidden in the complexity estimate will be reduced by half.

#### *Paralellization*

The algorithm in Borges and Daripa [7] is also highly parallelizable and the implementation is virtually architecture-independent. Theoretical estimates show good parallel scalability of the algorithm and numerical results show the accuracy of the method for problems with sharp variations on the inhomogeneous term. In [7], these have been discussed in detail and many performance results for sequential and parallel implementations have been presented.

#### *Extension*

In Badea and Daripa [3], fast algorithms developed by Daripa et. al. [7, 10, 12, 13] have been extended to a larger class of elliptic problems in a variety of domains. In particular, analysis-based fast algorithms to solve inhomogeneous elliptic equations of three different types in three different two-dimensional domains have been derived. These three different domains are: (i) interior of a circle, (ii) exterior of a circle, and (iii) circular annulus. Dirichlet, Neumann and mixed boundary value problems have been treated for all these cases. Three different types of problems have been considered there: (i) the Poisson equation, (ii) the Helmholtz equation (oscillatory case), and (iii) the Helmholtz equation (monotone case). The Poisson equation was considered again because in [7] we had used only one type of domain, but in Badea and Daripa [3] other domain cases are also addressed. These algorithms are derived from an exact formula for the solution of a large class of elliptic equations (where the coefficients of the equation do not depend on the polar angle when written in polar coordinates) based on Fourier series expansion and a one-dimensional ordinary differential equation.

### 4.4. Application to bio-fluid dynamics

In Daripa and Dash [11], our fast algorithm discussed in Borges and Daripa [7] has been applied to solve the problem of pulsatile blood flow in an eccentric catheterized artery. There the *eccentric* annular domain in which the problem of pulsatile blood flow needs to be solved is first conformally mapped onto a *concentric* annular domain before our fast algorithm for the concentric annular domain is applied. The details can be found in [11] which is also available in my publications homepage: http://www.math.tamu.edu/~daripa/pubs/allpubs.

## 5. Fast algorithms for separable elliptic equations in irregular domains

So far, we have presented our previous works on fast algorithms and their applications in two-dimensional domains with circular boundaries. However, many domains that arise in applications are irregular. Therefore, we have developed fast algorithms for two-dimensional irregular domains using a combination of domain embedding and our fast algorithms for regular domains such as the ones discussed above. Details on these can be found in several of our recent papers [2, 4, 5].

In adapting our fast algorithm for regular domains with circular boundaries to problems in irregular domains using domain embedding, we make a careful decision whether disk-like domain is the best choice or not as an embedding domain. Of course, this depends on the shape of the boundary of the irregular domain. For example, when shape of the irregular boundary is closer to that of a circle than that of a rectangle, disk like domain is an optimal one. In this case, we make use of our fast algorithm and least squares method (distributed or boundary control) the following way to solve the problem in an irregular domain. Figure 7 below shows an irregular domain embedded inside a disk.

In the left hand side of Figure 7, the problem for the elliptic operator $L$ in an irregular domain $\omega$ with its boundary $\gamma$ is shown. The figure shows the embedding disk $\Omega$ with its circular boundary $\Gamma$. (For now, ignore the strip region between $\gamma$ and the dashed curve). The modified problem in the regular domain $\Omega$ is shown on the right hand side of the figure. In solving the modified problem, our fast algorithm has been used. Only one of the irregular domains in which we have applied our fast algorithms to solve the Poisson equation is shown in Figure 8. This figure shows a hexagon embedded in the disk and the part of the grid system we need for use of our algorithm within the disk. These and more results can be found in our publications [2, 4, 5].

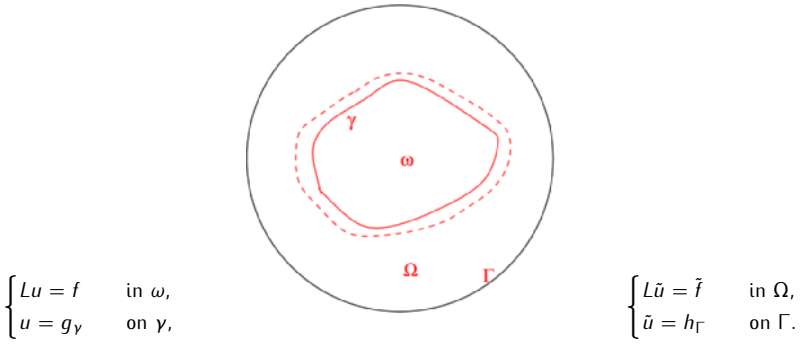


**Figure 7.** Domain embedding with a strip around the interior domain

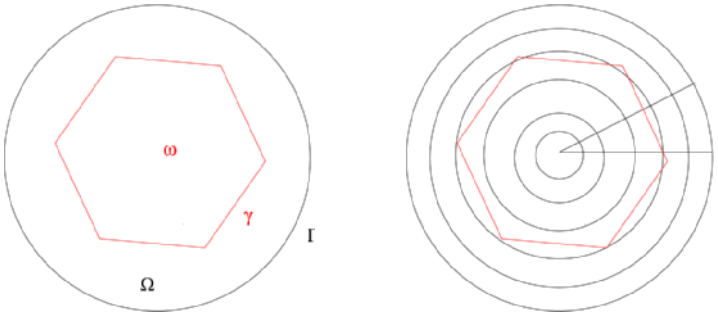


**Figure 8.** Illustration of grid, embedding and embedded domains

## 6. Conclusions

In this paper, we have reviewed some of our useful, accurate, parallelizable, fast algorithms for solving elliptic problems. We have also briefly presented some application areas where these algorithms have been applied in regular and irregular domains.

## References

[1] Anderes E., Coram M.A., Two-dimensional density estimation using smooth invertible transformation, J. Statist. Plann. Inference, 2011, 141(3), 1183–1193
[2] Badea L., Daripa P., On a boundary control approach to domain embedding method, SIAM J. Control Optim., 2001, 40(2), 421–449
[3] Badea L., Daripa P., A fast algorithm for two-dimensional elliptic problems, Numer. Algorithms, 2002, 30(3-4), 199–239
[4] Badea L., Daripa P., On a Fourier method of embedding domains using an optimal distributed control, Numer. Algorithms, 2003, 32(2-4), 261–273
[5] Badea L., Daripa P., A domain embedding method using the optimal distributed control and a fast algorithm, Numer. Algorithms, 2004, 36(2), 95–112

[6] Borges L., Daripa P., A parallel version of a fast algorithm for singular integral transforms, Numer. Algorithms, 2000, 23(1), 71–96
[7] Borges L., Daripa P., A fast parallel algorithm for the Poisson equation on a disk, J. Comput. Phys., 2001, 169(1), 151–192
[8] Daripa P., On applications of a complex variable method in compressible flows, J. Comput. Phys., 1990, 88(2), 337–361
[9] Daripa P., A fast algorithm to solve nonhomogeneous Cauchy–Riemann equations in the complex plane, SIAM J. Sci. Statist. Comput., 1992, 13(6), 1418–1432
[10] Daripa P., A fast algorithm to solve the Beltrami equation with applications to quasiconformal mappings, J. Comput. Phys., 1993, 106(2), 355–365
[11] Daripa P., Dash R.K., A numerical study of pulsatile blood flow in an eccentric catheterized artery using a fast algorithm, J. Engrg. Math., 2002, 42(1), 1–22
[12] Daripa P., Mashat D., Singular integral transforms and fast numerical algorithms, Numer. Algorithms, 1998, 18(2), 133–157
[13] Daripa P., Mashat D., An efficient and novel numerical method for quasiconformal mappings of doubly connected domains, Numer. Algorithms, 1998, 18(2), 159–178
[14] Du K., A simple numerical method for complex geometrical optics solutions to the conductivity equation, SIAM J. Sci. Comput., 2011, 33(1), 328–341
[15] Golberg M.A. (Ed.), Solution Methods for Integral Equations, Math. Concepts Methods Sci. Engrg., 18, Plenum Press, New York, 1978
[16] Golberg M.A. (Ed.), Numerical Solution of Integral Equations, Math. Concepts Methods Sci. Engrg., 42, Plenum Press, New York, 1990
[17] Greengard L., The Rapid Evaluation of Potential Fields in Particle Systems, ACM Disting. Diss., MIT Press, Cambridge, 1988
[18] Greengard L., Kropinski M.C., Mayo A., Integral equation methods for Stokes flow and isotropic elasticity in the plane, J. Comput. Phys., 1996, 125(2), 403–414
[19] Greengard L., Rokhlin V., A fast algorithm for particle simulations, J. Comput. Phys., 1987, 73(2), 325–348
[20] Greengard L., Rokhlin V., A new version of the fast multipole method for the Laplace equation in three dimensions, In: Acta Numer., 6, Cambridge University Press, Cambridge, 1997, 229–269
[21] Hackbusch W., Integral Equations, Internat. Ser. Numer. Math., 120, Birkhäuser, Basel, 1995
[22] Hwang K., Advanced Computer Architecture: Parallelism, Scalability, Programmability, McGraw–Hill, New York, 1993
[23] Uhlmann G., Electrical impedance tomography and Calderón's problem, Inverse Problems, 2009, 25(12), #123011