UCLA

COMPUTATIONAL AND APPLIED MATHEMATICS

High Order Expanding Domain Methods for the Solution of Laplace's Equation In Infinite Domains

Christopher R. Anderson

April 2014

CAM Report 14-44

Department of Mathematics
University of California, Los Angeles
Los Angeles, CA. 90095-1555

Abstract

In this paper we describe a discrete Fourier transform based procedure to evaluate a solution of Laplace's equation in R^2 or R^3 at points in a rectangular computational region. The numerical procedure is a modification of an "expanding domain" type method where one obtains approximations of increasing accuracy by expanding the computational domain. The modification presented here is one that leads to approximations that converge with high order rates of convergence with respect to domain size. Spectrally accurate approximations are used to approximate differential operators and so the method possess very high rates of convergence with respect to mesh size as well. Computational results on both two and three dimensional test problems are presented that demonstrate the accuracy and computational efficiency of the procedure.

0.1 Introduction

In this paper we present a numerical technique for the evaluation of the solution of Laplace's equation

$$\Delta u = f \qquad x \in \mathrm{R}^{\mathrm{N}} \tag{1}$$

for $\mathrm{N} = 2, 3$ at the grid points associated with a uniform discretization of a rectangular subregion $\Omega_0 \subset \mathrm{R}^{\mathrm{N}}$ with edges of characteristic length L_0. It is assumed that f vanishes outside of Ω_0.

There are many families of methods in use for computing the solution of Laplace's equation in infinite domains. The general idea behind one family of these methods [8] [14] [11] [13] is to create the solution as a sum of two components, one component that is due to a solution of Poisson's equation in the computational domain with pre-specified boundary conditions (typically homogeneous Dirichlet or Neumann conditions), and the other component a function harmonic in the computational domain and constructed so that the sum of the two components is a good approximation to the values of the infinite domain solution within the computational domain. Another family of methods are based upon convolution with the Greens' functions [7], with higher order spatial accuracy obtained either by using specially designed convolution algorithms [12] or using a regularized free-space Greens' function [6]. All these methods have the desirable property that the domain used for the construction of these component solutions is of the same size (or double in the case of convolution based approaches), than the original domain size. Accuracy is improved by improving the order of accuracy of the approximations to the differential operators involved and by decreasing the mesh size used.

Another family of methods, sharing many of the features of the convolution method approach, are of the "expanding domain" type. In these methods one trades accuracy for algorithmic simplicity and computes the solution to (1) by modifying the right hand side to have zero average value, solves a periodic problem on an extended computational domain, and then adds to that solution a component to account for the modification of the right hand side. The size of the extended domain used is typically determined experimentally, e.g. it's chosen large enough so that the finite size errors are acceptably small. When the solution procedure on the extended domain is implemented using FFT's this method is computationally efficient, especially when the requisite transforms are computed using high performance FFT implementations [4]. Moreover, since the procedure is based upon Fourier transform techniques, it implicitly uses spectrally accurate discretization procedures, and thus for problems with smooth right hand sides the rate of convergence of the errors due to spatial discretization are not fixed, but only limited by the smoothness of the right hand side. This aspect of the method has the consequence that for smooth problems one can use coarse grids to obtain accurate solutions and thus the cost of using an expanding domain may be acceptable.

This simple expanding domain procedure is observed to converge to the approximate solution values as the domain size is increased with a rate that is $O(\eta^{-2})$ in two dimensions and $O(\eta^{-3})$ in three dimensions where $\eta = (L/L_0)$ is the non-dimensional domain expansion factor and L is a measure of the expanded domain size. While this might be considered "fast" since it's better than a linear convergence rate, the cost of the computational effort and memory requirements grow quadratically in two dimensions and cubically in three dimensions with respect to the expansion factor η. This growth in computational and memory

requirements restricts the size of the expanded domain that can be used, and hence restricts the accuracy that can be obtained. In this paper, we show how one can modify the simple expanded domain approach and create "higher order" expanding domain methods in which the finite domain size errors are reduced at a rate proportional to to $O(\eta^{-q})$ with $q > 2$. In the specific implementation described here the decay is observed to be $O(\eta^{-4})$ for two dimensional problems and $O(\eta^{-5})$ for three dimensional problems. For the problems which motivated this work, that of evaluating two-electron integrals arising in Ab-initio quantum chemistry computations, this proved sufficient. However, if one assumes sufficient differentiability of the right hand side and sufficient differentiability of the Fourier transform of the right hand side, there is nothing preventing one from obtaining arbitrarily high order rates of convergence provided one is willing to expend more computational effort.

The motivation for the modifications required to improve the simple expanding domain procedure arises from an understanding of the errors that are associated with the approximation of an inverse Fourier transform on an infinite domain by a discrete Fourier transform on an expanding finite domain. In the first section we present a general discussion of these errors and in the second section propose a Fourier transform solution procedure for (1) that is specifically constructed so that the errors introduced by it's discrete approximation on an expanding finite domain converge to zero with a high order rates of convergence. The proposed procedure shares ideas similar to those in [6]; both rely on numerical procedures for explicitly accommodating the singular nature of the integrand arising in a Fourier transform based procedure. The main difference in the methods is the specific technique by which this singularity is accommodated. In [6], the singularity in the integrand is removed by explicit regularization, whereas in the method proposed here the singular nature is avoided by subtracting off a component that leads to a singular integrand and then adding back in its contribution analytically. This different treatment of the singularity has implications for the amount and nature of computational work that needs to be expended to improve the accuracy of a given solution. The advantage of one technique over the other can be expected to be problem and implementation specific. For example, in the procedure presented here the mesh sized used resolve the solution in the computational domain is decoupled from the errors associated with finite domain size effects, and thus the procedure presented here is most efficient for problems with smooth right hand sides or for problems where modest sized systematic errors associated with finite size domain effects can be tolerated. On the other hand, in [6], the errors associated with resolving the solution and the finite domain size effects are implicitly intertwined and both reduced by refining the mesh size in the computational domain. This fact suggests that the procedure in [6] will likely be more efficient for problems with less smooth right hand sides.

The Fourier transform based solution procedure is described in general terms; a numerical method results when the procedure is discretized. We have developed two different discretizations and in this paper, we present the first version, a version that is a modification of the simple expanded domain procedure already in use. This procedure has the virtue of being easy to implement and works very well when modest finite size domain errors can be tolerated. The computational costs and memory requirements still grow approximately $O(\eta^2)$ in two dimensions and $O(\eta^3)$ in three dimensions, so this version will have limited utility for problems where the finite size effects must be reduced to very small values. In a companion paper [1], we describe a "recursive expanded domain" method, which is more algorithmically complex, but has computational costs and memory requirements that scale

much more favorably with η.

In last section we present results for both two and three dimensional problems that demonstrate the high order spatial accuracy that can be achieved as well as the high order convergence with respect to domain size.

0.2 Approximate Inverse Fourier Transforms

A key ingredient to the solution procedure for the two and three dimensional problems is the use of discrete Fourier transforms to approximate the inverse of a continuous Fourier transform. In this section we discuss the nature of such an approximation and bring attention to those aspects that contribute to the errors of such approximations. To keep the exposition simple we focus on one dimensional transforms, but the following observations and conclusions can equally be made about the errors associated with approximating continuous multi-dimensional inverse Fourier transforms.

Given a continuous Fourier transform $\hat{f}(\xi)$ with it's inverse Fourier transform defined by

$$f(x) = \frac{1}{2\pi} \int_{-\infty}^{\infty} \hat{f}(\xi) \mathrm{e}^{i\,x\,\xi}\, d\xi \tag{2}$$

consider the task of creating approximations to the inverse transform at a set of P points associated with a uniform discretization of an interval $[-L, L]$ with mesh size $h = \frac{2L}{P}$, e.g. values $f_j = f(x_j)$ where $x_j = -L + j\,h$, $j = 0 \ldots P-1$. One approximation that can be used is a Fourier series approximation of the form

$$f_j = \frac{1}{2L} \sum_{k=-[L/h]}^{k=[L/h]} \hat{f}(\frac{k\,\pi}{L}) \mathrm{e}^{i\,x_j\,\frac{k\,\pi}{L}} \tag{3}$$

Here $[L/h]$ refers to the integer part of the value. This approximation has the advantage that the required sums can be evaluated using a discrete Fast Fourier transform routine with appropriate scaling and coefficient re-arrangement.

The derivation of (3) as an approximation to (2) arises when one denotes $\delta\xi = \frac{\pi}{L}$ and considers the following sequence of equalities and approximations;

$$\begin{aligned}
f_j &= \frac{1}{2\pi} \int_{-\infty}^{\infty} \hat{f}(\xi) \mathrm{e}^{i\,x_j\,\xi}\, d\xi && (4)\\
&\approx \frac{1}{2\pi} \sum_{k=-\infty}^{k=\infty} \hat{f}(k\,\delta\xi) \mathrm{e}^{i\,x_j\,k\,\delta\xi} \delta\xi && (5)\\
&= \frac{1}{2L} \sum_{k=-\infty}^{k=\infty} \hat{f}(\frac{k\,\pi}{L}) \mathrm{e}^{i\,x_j\,\frac{k\,\pi}{L}} && (6)\\
&\approx \frac{1}{2L} \sum_{k=-[L/h]}^{k=[L/h]} \hat{f}(\frac{k\,\pi}{L}) \mathrm{e}^{i\,x_j\,\frac{k\,\pi}{L}} && (7)
\end{aligned}$$

From this sequence of approximations, one can infer two primary sources of error that are introduced when using (3) to approximate (2). At a specific point x_j, the first source of error is the approximation of a continuous integral (4) by an infinite discrete sum (5). This

approximate integration error is expected to be $O((\delta\xi)^q)$ for some $q > 1$, and thus, to reduce the size of the errors one reduces $\delta\xi$. Since $\delta\xi = \frac{\pi}{L}$, this implies increasing L. Utilizing asymptotic error estimates based upon the Euler-MacLauren summation formula [3], one finds that the rate at which the error decreases, q, is related to the differentiability of $\hat{f}(\xi)$, e.g. the smoothness of $\hat{f}(\xi)$. For example, assuming that $\hat{f}(\xi)$ decays exponentially and is $2m+2$ times continuously differentiable, then the error is $O(\delta\xi)^{2\,m+2}$. When the function is not sufficiently differentiable to allow one to make use of the standard asymptotic error expansions one can utilize other error expansions [10], but the conclusion is the same, the more differentiable the transform the faster the rate of convergence of the computed values with respect to domain size.

The second source of the error is introduced by limiting the infinite sum (6) to a finite sum (7). For a fixed L, the number of terms kept in the finite sum is inversely related to the mesh size h, so as the mesh size is decreased this error is decreased. The rate at which this error decreases is directly related to the decay of $\hat{f}(\xi)$ as $\xi \to \pm\infty$, e.g. the differentiability of the function f associated with $\hat{f}$.

In addition to these two primary sources of error, there is an important secondary source of error due to the magnitude of the derivatives of the integrand in (2). Specifically, the rth derivative of the integrand in (3) with respect to ξ has factors that depend on x^r, so that when x is near the perimeter of computational domain the integrand has large derivatives and consequently the error in the approximation can be expected to be larger. For this reason, when evaluating the errors in the approximation of an inverse transform, one is typically concerned with the convergence of the errors at a set of points restricted to a fixed sub-interval $\Omega_0 = [-L_0, L_0] \subseteq [-L, L]$ as $L \to \infty$.

Thus, to reduce the errors associated with an approximation of the inverse transform that is computed with a discrete inverse Fourier transform, one needs to refine the mesh size used in physical space and increase the domain size L. The rate at which the errors decrease with mesh size in physical space is related to the differentiability of the function $f(x)$, and, if one restricts the evaluation points to a fixed sub-domain as $L \to \infty$, the rate at which the errors decrease with increasing L is related the differentiability of it's Fourier transform, $\hat{f}(\xi)$.

A simple example that demonstrates these errors is the approximation of the inverse transform of $\hat{f}(\xi) = \pi\,\xi^2\,\mathrm{e}^{\,|\xi|}$ corresponding to $f(x) = \dfrac{2-6x^2}{(x^2+1)^3}$. In Figure 1 we show the convergence behavior of the error in $f(x)$ for points $x \in \Omega_0 = [-1, 1]$ and $x \in [-L, L]$ with respect to the mesh size and computational domain size. The mesh size used is given by $h = \dfrac{2}{M}$, where M is the number of grid panels in $[-1, 1]$ and the computational domain used to evaluate the approximate inverse transform is $[-L, L]$ with $L \geq 1$.

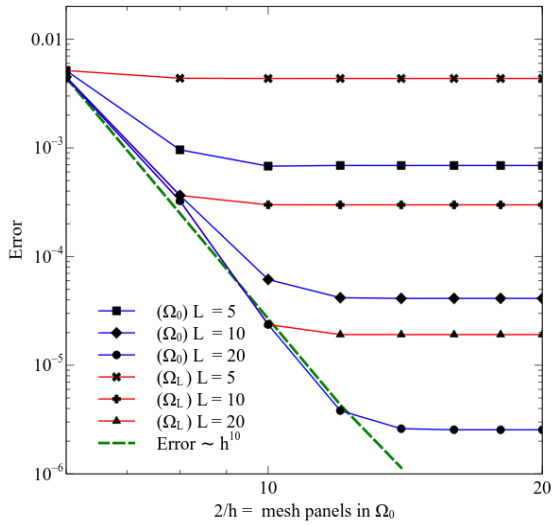


Figure 1: Maximal inverse transform errors for points in $\Omega_0 = [-1, 1]$ and points in $\Omega_L = [-L, L]$

.

Each of the curves in Figure 1 correspond to the error in $f(x)$ as a function of mesh size for values of $L = 5, 10, 20$. When the mesh width is large the error in the approximate inverse is dominated by the truncation of the infinite sum (6) to a finite sum (7). Since $f(x)$ is infinitely differentiable one expects rapid convergence with respect to the mesh width. This behavior is clearly observed in error curves towards the left side of Figure 1. As one decreases the mesh size one observes that the errors decay as h^{-10}. (Even though the function is infinitely differentiable, the growth in the size of the higher derivatives of the function ultimately limits the rate of convergence).

After convergence with respect to mesh size occurs, the dominant error becomes that of the domain size used to approximate the inverse transform. The behavior of this error is indicated by the behavior of the error curves towards the right hand side of Figure 1. If one uses the data at the finest mesh size and values $L = 10$ and $L = 20$, an estimate the rate of convergence of the error in Ω_0 is approximately $L^{-4.4}$. This rate is close to that predicted by a Trapezoidal method asymptotic error estimate of $O(L^{-4})$; an estimate that arises because the integrand and the derivative of the integrand in (2) both vanish at $\xi = 0$.

The non-uniformity of the errors in physical space are also evident. For a given domain size L, after convergence with respect to the mesh width one observes that the maximal error over the whole domain $[-L, L]$ is essentially an order of magnitude larger than the maximal error over the sub-interval $[-1, 1]$.

0.3 A High Order Expanding Domain Procedure

The procedure we propose for two and three dimensional problems are discretized versions of a Fourier transform based procedure for the constructing a solution to the continuous problem (1). With an understanding of the nature of the errors associated with approximations

of inverse transforms, the continuous procedure is designed so that when discrete approximations are used for the transforms, the errors associated with this discretization have the property that there is rapid convergence of the solution as the mesh size is decreased *and* rapid convergence as the domain size is increased.

Given a function $f(\vec{x})$ which vanishes for $\vec{x} \notin \Omega_0$, a Fourier transform based procedure for solving

$$\Delta u = f \qquad \vec{x} \in \mathrm{R}^N \tag{8}$$

consists of the following steps

(i) Create a modification function $\tilde{f}(\vec{x})$ so that the moments of $\vec{g}(\vec{x}) = f(\vec{x}) - \tilde{f}(\vec{x})$ vanish up to mth order;

$$\int_{\mathrm{R}^N} g(\vec{x})\, \vec{x}^{\,\alpha} = 0 \qquad |\alpha| = 0, 1, \ldots m \tag{9}$$

where $\alpha = \alpha_1 \alpha_2 \ldots \alpha_N$ is a multi-index, $|\alpha| = \alpha_1 + \alpha_2 + \cdots \alpha_N$, and $\vec{x}^{\,\alpha} = x_1^{\alpha_1} x_2^{\alpha_2} \cdots x_N^{\alpha_N}$.

(ii) Compute the solution to $\Delta v = g$ using Fourier transforms. Specifically, if $\hat{g}(\vec{\xi})$ is the Fourier transform of $\vec{g}(\vec{x})$, evaluate $v(\vec{x})$, the solution to $\Delta v = g$, using the inverse transform

$$v(\vec{x}) = \int \frac{\hat{g}(\vec{\xi})}{||\vec{\xi}||^2}\, \mathrm{e}^{i\vec{x}\cdot\vec{\xi}}\, d\vec{\xi} \tag{10}$$

(iii) Create the solution $u(\vec{x})$ to (8) by setting

$$u(\vec{x}) = v(\vec{x}) + w(\vec{x}) \tag{11}$$

for $\vec{x} \in \Omega_0$, where $w(\vec{x})$ is the solution to $\Delta w = \tilde{f}$ in R^N. It is assumed that the form of $\tilde{f}(\vec{x})$ is chosen so that an analytic representation of w is readily available.

The motivation for decomposing the task of constructing $u(\vec{x})$ into two components is so that the integrand in the inverse transform (10) required for Step (ii) is well behaved at $\vec{\xi} = 0$. The reason for reducing the singular behavior of the Fourier transform at the origin is so that when the Fourier transform solution of $\Delta v = g$ is approximated using a Fourier series solution over an expanding domain, the Fourier series solution will converge more rapidly as the domain size increases.

The reduction in the singular behavior of the transform at $\vec{\xi} = 0$ follows from the fact that

$$\left.\frac{\partial^\alpha \hat{g}}{\partial \xi^\alpha}\right|_{\vec{\xi}=0} = \int_{\mathrm{R}^N} g(\vec{x})\, \vec{x}^{\,\alpha} \tag{12}$$

so that the vanishing of the moments of $g(\vec{x})$ up to order m implies that all terms in the Taylor expansion of $\hat{g}(\vec{\xi})$ up to mth order vanish and therefore

$$\hat{g}(\vec{\xi}) = \sum_{|\alpha|>m} \frac{1}{\alpha!} \left.\frac{\partial^\alpha \hat{g}}{\partial \xi^\alpha}\right|_{\vec{\xi}=0} \vec{\xi}^{\,\alpha} \tag{13}$$

If the moment conditions in Step (i) hold, the integrand in (10) will be $O(||\vec{\xi}||^{m-1})$. This implies that at $\vec{\xi} = 0$ the integrand will be bounded if $m = 1$, and the integrand will vanish

along with it's first $m-2$ derivatives when $m \geq 2$. Therefore, the use of the modification function, $\tilde{f}$, leads to a reduction in the singular behavior of the integrand in the inverse Fourier transform (10).

In order that the procedure just described be viable as a basis for a computational method, it is necessary to identify a specific procedure for constructing the function $\tilde{f}(\vec{x})$ and the potential that it induces, $w(\vec{x})$. There are many ways to construct such a function, one choice that we found that works particularly well is to use a function of the general form

$$\tilde{f}(\tilde{x}) = \sum_{|\alpha|=0}^{m} \gamma_\alpha \frac{\partial^\alpha B_\delta(\vec{x})}{\partial x^\alpha} \tag{14}$$

Here $B_\delta(\vec{x})$ is a member of the family of polynomial mollifiers described in [2] of radius δ,

$$B_\delta(r) = \begin{cases} \frac{1}{\delta^N} \frac{\sigma_{q(N)}}{\omega_N} (1 - \left(\frac{r}{\delta}\right)^2)^q & r \leq \delta \\ \\ 0 & r > \delta \end{cases} \tag{15}$$

where $r = ||\vec{x}||$ and the radius of the mollifier δ is taken to be the minimum distance from the origin to the boundary of Ω_0. The value of q in the mollifier determines it's differentiability; $B_\delta(r)$ is $q-1$ times continuously differentiable. This fact implies that $\tilde{f}(\vec{x})$ will be $q-m-1$ times continuously differentiable. The coefficients $\sigma_{q(N)}$ and ω_N of the mollifiers appearing in (15) are determined by the condition that the mollifier have unit integral in R^N and are given in Appendix I.

The coefficients γ_α of $\tilde{f}(\vec{x})$ are determined so that the conditions in (9) are satisfied. Specifically, γ_α are determined by solving the linear system of equations

$$\sum_{|\alpha|=0}^{m} \left[\int_{\mathrm{R}^N} \frac{\partial^\alpha B_\delta(\vec{x})}{\partial x^\alpha} \vec{x}^\beta \right] \gamma_\alpha = \int_{\mathrm{R}^N} f(\vec{x})\, \vec{x}^\beta \qquad \text{for } |\beta| = 0 \ldots m \tag{16}$$

If the equations (16) are ordered by increasing $|\beta|$ and one applies integration by parts to the integrals on the left hand side of these equations, one finds that this system of equations is lower triangular with non-zero integer coefficients on the diagonal. The equations determining γ_α's are therefore non-singular and the evaluation of the solution to these equations is a matter of back-substitution. The non-zero coefficients on the left hand side of the system of equations (16) can be computed analytically.

With the coefficients γ_α determined, one must then have a means of evaluating the solution to $\Delta w = \tilde{f}$. The piecewise polynomial form of $B_\delta(\vec{x})$ leads to a solution of $\Delta W_\delta = B_\delta$ that has an analytic representation; $W_\delta(\vec{x})$ is a polynomial in r^2 for $r < \delta$ and for $r \geq \delta$ has the value $\frac{1}{2\pi}\log(r)$ in two dimensions and the value $-\frac{1}{4\pi r}$ in three dimensions. Since

$$\Delta \frac{\partial^\alpha W_\delta(\vec{x})}{\partial x^\alpha} = \frac{\partial^\alpha \Delta W_\delta(\vec{x})}{\partial x^\alpha} = \frac{\partial^\alpha B_\delta(\vec{x})}{\partial x^\alpha} \tag{17}$$

it follows that

$$w(\vec{x}) = \sum_{|\alpha|=0}^{m} \gamma_\alpha \frac{\partial^\alpha W_\delta(\vec{x})}{\partial x^\alpha} \tag{18}$$

and since $W_\delta(\vec{x})$ has an analytic representation, so does every term in the sum (18). The required formulas for W_δ are given in Appendix I.

0.4 Discretization

A computational method results when steps (i)-(iii) are implemented using discrete approximations. We describe the discretization procedure for two dimensional problems when the moment condition (9) is satisfied with $m = 2$. The discretization procedure for three dimensional problems or for larger or smaller values of m is similar.

We assume that the domain Ω_0 is the rectangular region centered at the origin $[-L_x^0, L_x^0] \times [-L_y^0, L_y^0]$. In order to utilize FFT's to carry out the approximate transforms, we use a uniform grid with grid points being those of an $M_x^0 \times M_y^0$ panel discretization of Ω_0. The mesh widths in each direction h_x and h_y are defined by $h_x = \dfrac{2L_x^0}{M_x^0}$ and $h_y = \dfrac{2L_y^0}{M_y^0}$.

Step (i) of the procedure requires the construction of the "moment matching function" $\tilde{f}$. To accomplish this, the right hand side of equations (16) must be approximated. These values are just the moments of f up to order m, and since the mesh is uniform and f vanishes outside of Ω_0, these moments can be evaluated with spectral accuracy using a standard Trapezoidal method approximation to the integral. After computation of these integrals, the solution of lower triangular system of equations (16) can be obtained by back substitution. In two dimensions, for a given choice of mollifier exponent q, if $\bar{f}_\alpha$ are the approximate values of the moments of f for $|\alpha| \leq 2$ and $\beta = \dfrac{\delta^2}{2(q+2)}$, then coefficients γ_α that satisfy (16) are given by

$$\gamma_{1\,1} = \bar{f}_{1\,1}$$

$$\gamma_{x\,1} = -\bar{f}_{x\,1} \qquad \gamma_{1\,y} = -\bar{f}_{1\,y}$$

$$\gamma_{x\,x} = \frac{(\bar{f}_{x\,x} - \beta\gamma_{1\,1})}{2} \qquad \gamma_{x\,y} = \bar{f}_{x\,y} \qquad \gamma_{y\,y} = \frac{(\bar{f}_{y\,y} - \beta\gamma_{1\,1})}{2}$$

Step (ii) requires the approximation of the coefficients of the forward transform of $g(\vec{x})$ and then the evaluation of the inverse transform (10). To approximate these transforms, the domain Ω_0 is expanded by a nominal expansion factor $\eta > 1$ in each direction to a domain Ω_L. This expansion is accomplished by adding panels of width h_x and h_y to both sides of the domain so that Ω_0 is approximately centered within Ω_L. In determining the expanded domain, one increases the size of Ω_L as needed so that an integral number of panels are added. Also, in order to improve the efficiency of discrete FFT routines one can add panels so that the total number in each direction is product of small primes. The expanded computational domain that results will thus be rectangular region approximately centered at the origin with grid points those of an $M_x \times M_y$ panel discretization and mesh widths h_x and h_y.

The function $\vec{g}(\vec{x}) = f(\vec{x}) - \tilde{f}(\vec{x})$ is then evaluated at all grid points in the domain Ω_L by evaluating $f(\vec{x}) - \tilde{f}(\vec{x})$ at points in Ω_0 and setting the values outside Ω_0 to zero. Delineating

the array of values of g at the grid points in the domain by $g(m,n)$ where $m = 0 \ldots M_x$ and $n = 0 \ldots M_y$, one then uses an FFT routine to construct an approximate forward transform of these values, e.g. $\hat{g}(k_1, k_2)$, defined by

$$\hat{g}(k_1,k_2) = \sum_{m=0}^{M_x-1} \sum_{n=0}^{M_y-1} \mathrm{e}^{-\frac{2\pi i k_1 m}{M_x}} \mathrm{e}^{-\frac{2\pi i k_2 n}{M_y}} \, g(m,n) \tag{19}$$

$$\begin{aligned} k_1 &= -[M_x/2] \ldots [(M_x-1)/2] \\ k_2 &= -[M_y/2] \ldots [(M_y-1)/2] \end{aligned}$$

where $[*]$ designates the integer part.

The array values comprising approximations to v defined by (10), $v(m,n)$, are obtained by using an inverse discrete Fourier transform to evaluate

$$v(m,n) = \sum_{k_1=-[M_x/2]}^{[(M_x-1)/2]} \sum_{k_2=-[M_y/2]}^{[(M_y-1)/2]} \mathrm{e}^{\frac{2\pi i k_1 m}{M_x}} \mathrm{e}^{\frac{2\pi i k_2 n}{M_y}} \frac{-\hat{g}(k_1,k_2)}{\left[\frac{4\pi^2 k_1^2}{D_x^2} + \frac{4\pi^2 {k_2}^2}{D_y^2}\right]} \tag{20}$$

$$\begin{aligned} m &= 0 \ldots M_x \\ n &= 0 \ldots M_y \end{aligned}$$

where $D_x = M_x h_x$ and $D_y = M_y h_y$. In these sums, the value of the summand at $(k_1, k_2) = 0$ is set to zero. We caution the reader that the evaluation of the sums (19) and (20) using standard FFT routines likely requires both scaling and coefficient re-arrangement. The coefficient re-arrangement is necessary because of our use of both positive and negative values for k_1 and k_2.

In Step (iii), one obtains the final result by combining the values of $v(m,n)$ with the values of $w(m,n) = w(\vec{x}(m,n))$ where w is given by (18) for each grid point $\vec{x}(m,n) \in \Omega_0$.

In three dimensional discretization procedure is completely analogous to the two dimensional discretization procedure. However, in Step (i), the size of the system of equations that must be solved to construct $\tilde{f}$ is larger because there are more moments of a given order to match. If one restricts oneself to just matching moments up to second order, then for a given choice of mollifier exponent q, if $\bar{f}_\alpha$ are the approximate values of the moments of f for $|\alpha| \leq 2$ and $\beta = \dfrac{\delta^2}{2(q+2)+1}$, then coefficients γ_α that satisfy (16) are given by

$$\gamma_{1\,1} = \bar{f}_{1\,1}$$

$$\gamma_{x\,1} = -\bar{f}_{x\,1} \quad \gamma_{1\,y} = -\bar{f}_{1\,y} \quad \gamma_{1\,z} = -\bar{f}_{1\,z}$$

$$\gamma_{x\,x} = \frac{(\bar{f}_{x\,x} - \beta\gamma_{1\,1})}{2} \quad \gamma_{y\,y} = \frac{(\bar{f}_{y\,y} - \beta\gamma_{1\,1})}{2} \quad \gamma_{z\,z} = \frac{(\bar{f}_{z\,z} - \beta\gamma_{1\,1})}{2}$$

$$\gamma_{x\,y} = \bar{f}_{x\,y} \qquad \gamma_{x\,z} = \bar{f}_{x\,z} \quad \gamma_{y\,z} = \bar{f}_{y\,z}$$

The main source of error in this discrete approximation is that associated with approximating continuous transforms by discrete transforms. The error in the forward transform is solely due to the use of a finite size mesh width since g has support contained within Ω_0. However, as discussed in Section 1, the error in the inverse transform is due to the use of a finite mesh width *and* the error due to the use of a finite sized domain.

The error in both the forward and inverse transforms associated with finite mesh width size is due to the limited range of wavenumbers at which the Fourier transform values can be approximated when using discrete values on a grid. The rate of convergence of these errors to zero is not a fixed property of the discretization but dictated by the differentiability of f and $\tilde{f}$, e.g. it is a "spectrally accurate" discretization. However, one typically fixes the differentiability of $\tilde{f}$ by selecting the mollifier exponent q, thus leading to a discretization error contribution to the final solution that has a fixed rate of convergence. Assuming the differentiability if f is greater than that of $\tilde{f}$, the expected size of this component of the error will be $O(h^{(q-m)+1})$, where h is a measure of the mesh width. The computational results given in the next section will demonstrate the effect that the differentiability of $\tilde{f}$ has on the accuracy of the computed values and, in particular, show that in computations with $m = 2$, that these errors do behave as $O(h^{q-1})$.

The behavior of the errors associated with the use of a domain of finite size for the inverse transform approximation are dictated by the rate of convergence of a Trapezoidal approximation to the integral (10). Since mesh width in this approximation, $\delta\xi$, is $O(\frac{1}{\eta L_0})$ where L_0 is the minimum of L_x^0 and L_y^0, the errors associated with the use of a finite size domain can be expected to behave as $O(\eta^{-p})$ or $O(\eta^{-p}\log(\eta))$ where p is determined by the rate of convergence of the integral approximation and η is the nominal domain expansion factor. A general a-priori estimation of the rate of convergence is difficult because the asymptotic error expansions for the Trapezoidal method applied to integrals of the form (10) requires having specific knowledge about the functional form of the integrand at $\vec{\xi} = 0$. However, what can be inferred from an asymptotic error expansion of the Trapezoidal method applied to each quadrant separately [9], is that one expects higher rates of convergence when more terms in the local Taylor series expansion at $\vec{\xi} = 0$ vanish. As will be demonstrated by the numerical results in the next section, we find that the rate of convergence with respect to the expansion factor η is $O(\eta^{-(m+2)})$ in two dimensions, and $O(\eta^{-(m+3)})$ in three dimensions when modification functions $\tilde{f}$ are used that match match the moments of f to order m

As for the computational work associated with a discretization, if K is the total number of grid points in Ω_0 (e.g. $K = M_x^0 \times M_y^0$ and $K = M_x^0 \times M_y^0 \times M_z^0$ in two and three dimensions respectively), then for an expansion factor of η, the computational work in Step (i) and Step (iii) is $O(K)$ and is independent of η. The computational work associated with Step (ii) will be $O(\eta^N K \log(K)) + O(\eta^N \log(\eta^N) K)$. For modest values of η, this growth in computational work can be acceptable, especially when the discrete Fourier transform is computed using very efficient FFT's. The amount of memory required also grows as η^N, which in the case of three dimensional problems and larger values of η may be unacceptable. For such cases, an alternate but more complicated procedure for carrying out Step (ii) described in [1] can be used. This latter procedure has a memory scaling that is just linear in η and requires computational work that grows as $\eta^{(N-1)}$.

0.5 Computational Results

The test problem consisted of determining the values of the solution to

$$\Delta u = f \quad \vec{x} \in \mathrm{R}^N \tag{21}$$

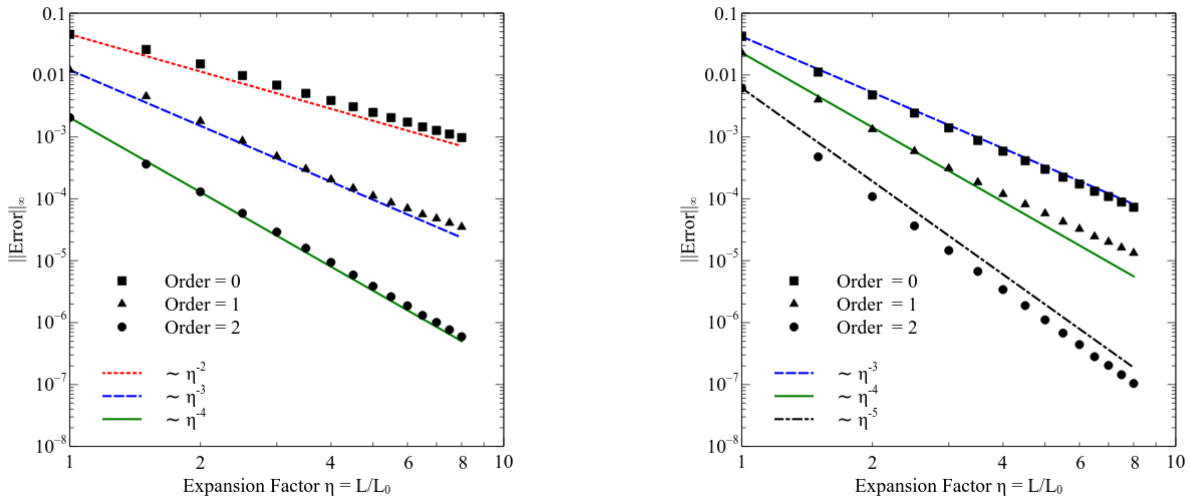
in the region $\vec{x} \in \Omega_0 = [-L_0, L_0]^N$ with $L_0 = 1$ and $N = 2, 3$. The computational grid used in Ω_0 was taken to be a uniform grid with M panels in each direction. The function f was chosen to be a linear combination of two mollifiers of the form (15) with width $\delta = 0.6$, specifically

$$f(\vec{x}) = B_{\delta=.6}(\vec{x} - \vec{x}_A) + B_{\delta=.6}(\vec{x} - \vec{x}_B) \tag{22}$$

where $\vec{x}_A = (.11, .22, (.33))$ and $\vec{x}_B = (-.33, -.22, -(.11))$. A mollifier exponent $q = 9$ was specified. This choice of exponent leads to a potential u that is 10 times continuously differentiable. An exact solution can be evaluated analytically using the formulas given in Appendix I.

In the construction of the moment matching function $\tilde{f}$ in Step (i), the location and width of the mollifiers must be chosen. It is advantageous to use mollifiers with as large a width as possible so $\tilde{f}$ and it's transform can be accurately represented with a coarse mesh. For all of the test computations the moment matching function was constructed using mollifiers and their derivatives centered at the origin and of width $\delta = 0.9$.

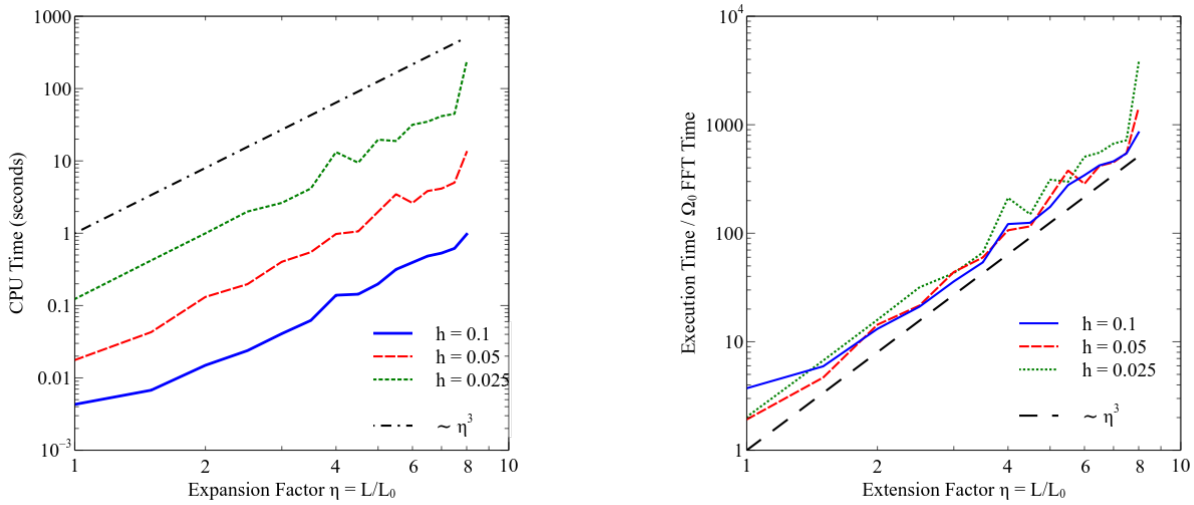
The discrete Fourier transform computations were carried out using FFTW3 routines [4] [5].



(a) (b)

Figure 2: The behavior of the error in $\Omega_0 = [-L, L]^N$ as the domain is expanded for different orders, $m = 0, 1, 2$, of the moment matching modification. A mesh width of $\frac{1}{40}$ was used in each direction. (a) N= 2 (b) N = 3.

The first set of computational results concerns the behavior of the error in the potential as the domain used in Step (ii), $[-L, L]^N$, is increased in size (e.g. as $\eta = \frac{L}{L_0}$ is increased). Of particular interest is the dependence of the rate of convergence on the maximal order of the moments matched in Step (i). In this computation, the number of panels M used in each direction of Ω_0 was fixed at 80, e.g. the mesh width was fixed at $\frac{1}{40}$. This mesh width was sufficiently small to insure that the solution values were essentially converged with respect to mesh size. A value of $q = 7$ was used as the mollifier exponent for the construction of the moment correction function. In Figure 2 the relative errors in the potential evaluated in the maximum norm are presented for values of $\eta = 1 \ldots 8$ for two and three dimensional computations. The results clearly demonstrate a well defined rate of convergence, a rate that increases by one with each increase in the maximal order of the moments used in Step (i). In particular, the observed rate of convergence is $\eta^{-(m+2)}$ in two dimensions and $\eta^{-(m+3)}$ in three dimensions.



(a) (b)

Figure 3: Three dimensional test problem computational time. Time in CPU seconds (a) and Time in FFT units (b). Ω_0 FFT Time is the time for one forward and one inverse discrete Fourier Transform applied to the values in Ω_0.

In Figure 3, we show the computational time for the three dimensional test problem as the domain expanded for different mesh sizes. Second order moment matching, $m = 2$, was used and the exponent of the mollifier used in the moment matching function was taken to be $q = 7$. The results are given in units of CPU seconds (a) and in FFT units (b). The CPU seconds are those of a desktop machine with an AMD FX-8120 eight-core processor with multi-threading execution obtained using OpenMP. The FFT unit of time is the time required for one forward and one inverse transform of the data values in Ω_0. We give the results in both of these units, as the time in seconds gives one an idea about

the computational time required when using a currently available high performance desktop machine, and the reported time in FFT units allows one to estimate the time the procedure would take for other types of computational hardware.

If K is the total number of grid points in Ω_0, then as η is increased, the computational work is dominated by the FFT computation over the expanded domain and so formally scales as $\eta^N K \log(\eta^N K)$. The computational timings are in approximate agreement with this scaling. From the data in Figure 3(b) when $\eta = 1$, one can deduce that the additional cost of implementing Step (i) and Step (iii), e.g. forming the moment correction function $\tilde{f}$ and then adding in the potential it induces. Specifically, one finds that computational time is only a small multiple of the time of a single forward and inverse FFT over Ω_0. Since the computational cost of Steps (i) and (iii) does not grow with η one can conclude that if one is already using an expanding domain procedure, very little additional computational work need be performed in order to create a method that has higher order rates of convergence with respect to domain size.

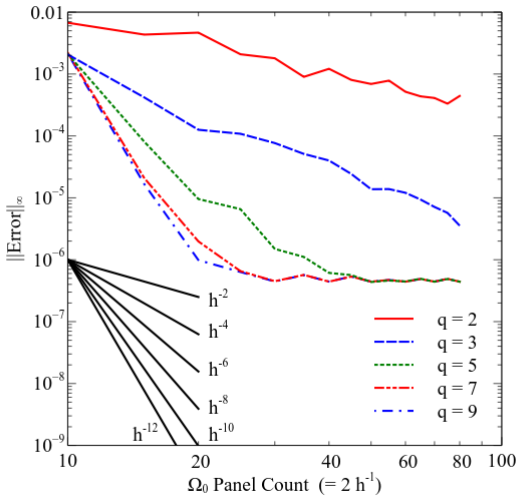


Figure 4: Maximal relative potential error in $\Omega_0 = [-1, 1]^3$ for different computational mesh sizes $h = \frac{2}{\Omega_0 \text{ panel count}}$.

The last set of computational results concerns the behavior of the error in the potential with respect to the use of a finite mesh size, e.g. the discretization error. If the moment matching function $\tilde{f}$ were infinitely differentiable, the error component associated with the mesh size would decrease at a rate dependent only on the smoothness of the right hand side (e.g. spectral convergence). However, the moment matching function proposed here has a limited differentiability, and this limits the rate of convergence which can be obtained. In Figure 4 we show the behavior of the errors in the potential of the three dimensional test problem with respect to decreasing mesh size for several values of the mollifier exponent q. Second order moment matching $m = 2$ and an expanded domain size corresponding to $\eta = 6$ were used. For this value η, the finite domain size contribution to the total error is $O(10^{-7})$, so that over the range of mesh widths where the mesh width error dominates, the results

clearly indicate a rapid convergence with respect to mesh size. The rate of convergence increases as the differentiability of the moment matching function increases. In fact for $q = 9$, the rate is approximately $O(h^{-12})$. For many problems, as well as the particular test problem considered here, one finds little difference between the convergence for rates when $q \geq 7$.

In consideration of the timing results presented in Figure 3, one might be concerned about the utility of a method for solving the three dimensional Poisson's equation that requires tens of seconds for a single computation to obtain an accuracy of $O(10^{-7})$ (e.g. 3(a) with $h = 0.025$ and $\eta = 8$). However, for smooth problems, the use of spectral approximation leads to such a rapid rate of convergence that the mesh size required to resolve the solution need not be particularly small and thus one can avoid such large execution times. For example, in our test problem, Figure 4 indicates that convergence with respect to mesh size occurs near the coarsest mesh width $h = 0.1$. The solution times obtained with the largest domain $\eta = 8$ is less than a second. To obtain an error of $O(10^{-5})$, one can use a domain with $\eta = 4$ and the solution time is just tenths of a second. If one needs to have solution values on a finer grid, one can use spectral interpolation over the original domain.

0.6 Conclusion

In the simple expanding domain procedure for computing approximate solutions of Laplace's equation in infinite domains, one computes the solution by modifying the right hand side to have zero average value, solves a periodic problem on an extended computational domain, and then adds to that solution a component to account for the modification of the right hand side. The accuracy of the approximation is increased by both refining the computational mesh and expanding the size of the computational domain. In this paper we have described an improved version this simple procedure, a procedure that ostensibly consists of adding a specially constructed modification function to the right hand side before one solves the periodic problem. Motivated by an understanding that the rate of convergence of the expanding domain procedure with respect to the domain size is dictated by the rate of convergence of a discrete approximation to an inverse Fourier transform, the modification function used to alter the right hand side is a "moment matching function" and one that leads to an integrand of the inverse transform that is less singular at the origin. The consequence of this modification is a more rapid rate of reduction of the finite domain size errors as the domain size is increased. Our computational results demonstrate that if one uses a modification function that matches the moments of the right hand side to second order the rate of convergence with respect to domain size is $O(\eta^{-4})$ for two dimensional problems and $O(\eta^{-5})$ for three dimensional problems where η is the non-dimensional expansion factor.

The particular type of moment matching functions proposed here have the virtue that both their values and the values of their associated potential can be efficiently evaluated. As demonstrated by the computational results, the additional computational cost to create and utilize this modification function is but a small factor times the cost of an FFT over the original domain. In addition, the proposed functions have a parameterized degree of differentiability so that when the procedure is applied to problems with smooth right hand side the high accuracy obtainable with spectral differential operator approximation is not adversely effected.

The computational work scales formally as $O(\eta^N K \log(\eta^N K))$ where K is the total number of grid points in the non-expanded domain and N is the dimension. With the use of high performance FFT's to carry out the required transforms, the procedure is relatively easy to implement, and is highly efficient for modest values of η. Since the underlying approximation of the differential operators has spectral accuracy, the proposed procedure is ideally suited for use on problems with smooth right hand sides. For problems that require very small finite domain size errors, the computational cost and memory requirements associated with larger values of η can become significant, and one should consider using the "recursive expanded domain" method [1], a method which is more algorithmically complex, but has computational costs and memory requirements that scale much more favorably with η.

Lastly, the method that is proposed here is one to compute a potential in an infinite domain; the method readily extends to the computation of any derivatives of the potential. For example, to evaluate the derivative with respect to x, one just multiplies the integrand in (20) by an extra factor of $\frac{2\pi k_1}{L_x}$ before applying the approximate inverse transform and then adds in the x-derivative of the potential induced by the moment matching function. The required derivative of the potential induced by the moment matching function can be evaluated analytically.

0.7 Appendix I

0.7.1 Normalization factors

For mollifiers of the form

$$B(r) = \begin{cases} \dfrac{\sigma_{q(N)}}{\omega_N}(1 - r^2)^q & r \leq 1 \\ \\ 0 & r > 1 \end{cases} \tag{23}$$

Table 1 gives the scaling factors $\sigma_{q(N)}$ for $q = 1 \ldots 9$ so that $\int_{\mathrm{R}^N} B(r)\, dr = 1$ for $N = 2, 3$ and $\omega_2 = 2\pi$ and $\omega_3 = 4\pi$.

	q = 1	q = 2	q = 3	q = 4	q = 5	q = 6	q = 7	q = 8	q = 9
$\sigma_{q(2)}$	4	6	8	10	12	14	16	18	20
$\sigma_{q(3)}$	$\frac{15}{2}$	$\frac{105}{8}$	$\frac{315}{16}$	$\frac{3465}{128}$	$\frac{9009}{256}$	$\frac{45045}{1024}$	$\frac{109395}{2048}$	$\frac{2078505}{32768}$	$\frac{4849845}{65536}$

Table 1: Mollifier normalization factors

0.7.2 Moments

The 0th order moment of all members of the family (23) is 1, all first order moments vanish due to radial symmetry, but not all second order moments vanish. The moments with respect to x_i^2 are non-zero and one finds that in two dimensions for a mollifier with exponent q and width δ,

$$\int_{\mathrm{R}^2} B_\delta(\vec{r})\, {x_i}^2 = \frac{\delta^2}{2(q+2)} \qquad i = 1, 2$$

and for three dimensions, one finds

$$\int_{\mathrm{R}^3} B_\delta(\vec{r})\, {x_i}^2 = \frac{\delta^2}{2(q+2)+1} \qquad i = 1, 2, 3$$

0.7.3 Potentials

As discussed in [2], the functions $v_2(r)$ and $v_3(r)$ defined by

$$v_2(r) \;=\; c_{0(2)} + \sum_{j=1}^{q+1} \frac{(-1)^{(j-1)}}{(2\,j)^2} \binom{q}{j-1} r^{\,2j} \tag{24}$$

$$v_3(r) \;=\; c_{0(3)} + \sum_{j=1}^{q+1} \frac{(-1)^{(j-1)}}{(2\,j)(2\,j+1)} \binom{q}{j-1} r^{\,2j} \tag{25}$$

are solutions of

$$\Delta v_N \;=\; (1-r^2)^{\,q} \;=\; \sum_{k=0}^{q} (-1)^k \binom{M}{k} r^{\,2k}$$

for $r \leq 1$ for two and three dimensions respectively. It is convenient to choose $c_{0(N)}$ so that that $v_N(r)$ vanish at $r = 1$. Values of $c_{0(N)}$ are given in Table 2. The potential associated with solutions of $\Delta W_\delta = B_\delta(r)$, where $W_\delta(r)$ is given by (23), are obtained by the appropriate scaling;

$$W_\delta(r) = \begin{cases} \dfrac{\sigma_{M(2)}}{2\pi}\, v_2(\dfrac{r}{\delta}) + \dfrac{\log(\delta)}{2\pi} & r \leq \delta \\ \\ \dfrac{\log(r)}{2\pi} & r > \delta \end{cases}$$

in two dimensions, and in three dimensions by

$$W_\delta(r) = \begin{cases} \dfrac{1}{\delta}\left(\dfrac{\sigma_{M(3)}}{4\pi}\, v_3(\dfrac{r}{\delta}) \;-\; \dfrac{1}{4\pi}\right) & r \leq \delta \\ \\ -\dfrac{1}{4\pi r} & r > \delta \end{cases}$$

In all of the above formulas $\sigma_{M(2)}$ and $\sigma_{M(3)}$ are the normalization factor for $B(r)$ whose values are given in Table 1.

	M = 1	M = 2	M = 3	M = 4	M = 5	M = 6	M = 7	M = 8	M = 9
$c_{0(2)}$	$-\frac{3}{16}$	$-\frac{11}{72}$	$-\frac{25}{192}$	$-\frac{137}{1200}$	$-\frac{49}{480}$	$-\frac{363}{3920}$	$-\frac{761}{8960}$	$-\frac{7129}{90720}$	$-\frac{7381}{100800}$
$c_{0(3)}$	$-\frac{7}{60}$	$-\frac{19}{210}$	$-\frac{187}{2520}$	$-\frac{437}{6930}$	$-\frac{1979}{36036}$	$-\frac{4387}{90090}$	$-\frac{76627}{1750320}$	$-\frac{165409}{4157010}$	$-\frac{141565}{3879876}$

Table 2: Potential constant factors $c_{0(N)}$ so that $v_N(1) = 0$, $N = 2, 3$.

Bibliography

[1] Christopher R. Anderson. A Recursive Expanding Domain Method for the Solution of Laplace's Equation In Infinite Domains. Technical Report CAM-14-45, Department of Mathematics, UCLA, Los Angeles, California, May 2014.

[2] Christopher R. Anderson. Compact Polynomial Mollifiers For Poisson's Equation. Technical Report CAM-14-43, Department of Mathematics, UCLA, Los Angeles, California, May 2014.

[3] K.E. Atkinson. *An Introduction To Numerical Analysis*. Wiley, 2nd edition, 1978.

[4] Matteo Frigo and Steven G. Johnson. The Design and Implementation of FFTW3. *Proceedings of the IEEE*, 93(2):216–231, 2005. Special issue on "Program Generation, Optimization, and Platform Adaptation".

[5] Matteo Frigo and Steven G. Johnson. FFTW 3.3.4. http://www.fftw.org/, 2014.

[6] Mads Mølholm Hejlesen, Johannes Tophøj Rasmussen, Philippe Chatelain, and Jens Honoré Walther. A high order solver for the unbounded Poisson equation. *Journal of Computational Physics*, 252(0):458–467, 2013.

[7] Roger W Hockney and James W Eastwood. *Computer simulation using particles*. CRC Press, 1988.

[8] R.A James. The solution of poisson's equation for isolated source distributions. *Journal of Computational Physics*, 25(2):71–93, 1977.

[9] JN Lyness. Applications of extrapolation techniques to multidimensional quadrature of some integrand functions with a singularity. *Journal of Computational Physics*, 20(3):346–364, 1976.

[10] JN Lyness and BW Ninham. Numerical quadrature and asymptotic expansions. *Math. comp*, 21(98):162–178, 1967.

[11] P. McCorquodale, P. Colella, G.T. Balls, and S.B. Baden. A scalable parallel Poisson solver in three dimensions with infinite-domain boundary conditions. *International Conference Workshops on Parallel Processing, ICPP 2005 Workshops*, pages 163–172, 2005.

[12] Ji Qiang. A high-order fast method for computing convolution integral with smooth kernel. *Computer Physics Communications*, 181(2):313–316, 2010.

[13] D.B. Serafini, P. McCorquodale, and P. Colella. Advanced 3D Poisson solvers and particle-in-cell methods for accelerator modeling. *Journal of Physics: Conference Series*, 16(1):481–485, 2005. cited By (since 1996)2.

[14] Z Jane Wang. Efficient implementation of the exact numerical far field boundary condition for Poisson equation on an infinite domain. *Journal of Computational Physics*, 153(2):666–670, 1999.