# MULTILEVEL EVALUATION OF INTEGRAL TRANSFORMS WITH ASYMPTOTICALLY SMOOTH KERNELS

A. BRANDT † AND C. H. VENNER ‡

**Abstract.** Multilevel algorithms developed for the fast evaluation of integral transforms and the solution of the corresponding integral and integro-differential equations rely on smoothness properties of the discrete kernel (matrix) and thereby on grid uniformity (see [6], [18]). However, in actual applications, e.g. in contact mechanics, in many cases a substantial increase of efficiency can be obtained using non-uniform grids, since the solution is smooth in large parts of the domain with large gradients that occur only locally.

In this paper a new algorithm is presented which relies on the smoothness of the continuum kernel only, independent of the grid configuration. This will facilitate the introduction of local refinements, wherever needed. Also, the evaluations will generally be faster; for a $d$ dimensional problem only $O(s^{d+1})$ operations per gridpoint are needed, if $s$ is the order of discretization. The algorithm is tested using a one dimensional model problem with logarithmic kernel. Results are presented using both a second and fourth order discretization. For testing purposes, and to compare with results presented in [6], uniform grids covering the entire domain were considered first.

**Key words.** multigrid, integral transform, singular smooth kernel, fast evaluation, local grid refinement

**AMS subject classification.** 65R10, 65R20, 65N55

## 1. Introduction.

The numerical evaluation of integral transforms (multi-integrals) of the type:

$$Gu(x) = \int_{\Omega} G(x,y)u(y)dy \tag{1}$$

is a common task in many fields in mathematics, physics, and engineering, including: integro-differential equations, (Fredholm) integral equations, elasticity problems, computer graphics (radiosity), electrostatics, astrophysics, and ab-initio Hartree-Fock chemistry calculations. The evaluation can be a task by itself or be needed in the process of (numerically) solving a (system of) integro-differential equation(s) in which case $u$ generally is the unknown function.

Discretizing (1) on a grid generally implies that, at the expense of a *discretization error*, the continuous transform is replaced by a matrix multiplication, or "multi-summation", i.e., the evaluation of the $n$-vector $\underline{G}u = \overline{G}\underline{u}$ given the $\bar{n} \times n$ (dense) matrix $\overline{G}$ and the $\bar{n}$-vector $\underline{u}$, a task well known also from problems with gravitating masses, vortex schemes, coulombic molecular forces, and other many-body long range interactions.

If the matrix $\bar{G}$ has arbitrary entries, each of them must enter the calculations and $\bar{n}n$ (often $n = \bar{n}$) operations must be used. In that case there can exist no way which is significantly faster than such a straigthforward multiplication. In many applications

† Weizmann Institute of Science, Department of Applied Mathematics and Computer Science, Rehovot, 76100, Israel. The work of this author was partially supported by the United States Air Force Office of Scientific Research, Grant F 49620-92-J-0439.

‡ University of Twente, Faculty of Mechanical Engineering, P.O. Box 217, 7500 AE Enschede, The Netherlands, and Weizmann Institute of Science, Department of Applied Mathematics and Computer Science, Rehovot, 76100, Israel, e-mail: c.h.venner@wb.utwente.nl and kees@wisdom.weizmann.ac.il. This author is a research fellow of the Royal Netherlands Academy of Arts and Sciences. His research is partially supported by the Feinberg Graduate School of the Weizmann Institute of Science.

of interest, however the "discrete kernel" $\bar{G}$ has certain special properties that can be used. In particular, in most physical problems $\bar{G}_{ij} = G(x_i, y_j)$ where $x_i \in I\!R^{\bar{d}}$ and $y_j \in I\!R^d$ (i.e. $x_i = (x_i^1, x_i^2, ..., x_i^{\bar{d}})$, $y_j = (y_j^1, y_j^2, ..., y_j^d)$ where $x_i^\alpha$ and $y_j^\beta$ are real numbers) and the kernel $G(x, y)$ has some smoothness properties; usually $\bar{d} = d$ and often $y_j = x_j$. These kernels for example arise in many-body interactions, where $y_j$ is the position of the particle with "charge" $u_j = u(y_j)$, and $Gu_i = Gu(x_i)$ is the total effect of all particles at point $x_i$. When originating from the discretization of (1), the points $x_i$ are usually the points of a grid, and so are the points $y_j$. However, unlike the cases referred to above, the discretization generally yields a mesh-size dependent discrete kernel $\bar{G}_{ij} \neq G(x_i, y_j)$; see e.g. [6].

Using specific properties of the matrix $\bar{G}$, a variety of approaches has been introduced to reduce the computational cost of the multi-summation $\bar{G}\underline{u}$ to below the $O(\bar{n}n)$ operations mentioned above, e.g. using far field expansions [16], [15], [13], hierarchical solvers in many body simulations [1], [3], [10], [12], and FFT based schemes [17] (for the solution of the corresponding integral equations). In the past decade wavelet techniques have become popular, e.g. see [5], [2], where the complexity reduction is obtained by representation on a suitable set of increasingly coarse base functions. Most of these techniques have restrictions, e.g. a limited accuracy, limitation to potential type kernels, or they require a significant amount of matrix manipulations to arrive at the sparse matrix which enables the fast evaluation.

A simple and general approach referred to as *multilevel matrix multiplication* or *multilevel multi-integration* was introduced in [6], [7]. This algorithm only relies on the smoothness properties of the matrix $\bar{G}$. For example in the case $\bar{n} = n$ the complexity of the evaluation up to accuracy $\epsilon$ was reduced by this approach to $O(n\frac{\log\frac{1}{\epsilon}}{\log\log\frac{1}{\epsilon}})$ for smooth kernels, to $O(n\ln(\frac{1}{\epsilon}))$ for asymptotically smooth kernels, and to $O((n\log n)(\log(\frac{1}{\epsilon})^d)$ if $d > 1$ and $O(n\ln(\frac{1}{\epsilon}))$ if $d = 1$ for oscillatory kernels. Moreover, when merged with a suitable distributed relaxation and the usual multigrid solution techniques, an associated integral or integro-differential equation (e.g., finding $u$ given $Gu$)), can be solved to an error below the discretization error in an amount of work that is only a fraction larger than the work involved in a single evaluation of the discrete transform; see e.g. [6],[18].

In this paper the subject of fast evaluation of integral transforms with asymptotically smooth kernels is revisited. In actual applications, e.g. in contact mechanics, in many cases the function $u(y)$ will be smooth in large parts of the domain, and large gradients will occur only locally. It is only in such regions that a fine grid is really needed. Whenever this is the case one can expect local grid refinement techniques (adaptive grids) to yield substantially reduced computing times without loss of accuracy. In addition, local grid refinements may be essential to maintain work-accuracy efficiency in the case that $u$ has some singularity. In principle the multilevel methodology allows such local grid refinements in a very natural way. For partial differential problems this is shown in e.g. [8], [4], and [9]. The aim of the present work is to develop such techniques for the fast evaluation of integral transforms of the above type, and for the solution of the corresponding integro-differential equation. However, first we restrict ourselves to the evaluation only.

In [6] the discrete transform (multisummation) was obtained by discretizing (1) on a uniform grid covering the entire domain. Indeed, if the kernel $G(x, y)$ is smooth, then with a suitable discretization on a uniform grid the discrete kernel $\bar{G}_{ij}$ will have the same smoothness properties, and the evaluation of $\bar{G}\underline{u}$ on a given grid can (recursively) be replaced by a restriction of $\underline{u}$ to a coarser grid, a multisummation

on this coarser grid, and interpolation to the finer grid (and a local correction if the kernel has a singularity). However, on a non-uniform grid, smoothness of $G(x,y)$ does not entail that $\bar{G}_{ij}$ is also smooth and therefore a new algorithm had to be developed.

The novelty of this algorithm is that instead of writing the discretized transform as a single discrete transform $\bar{G}\underline{u}$ it is written in a form containing several multi-summations $\bar{G}^l\underline{U}^l$, where for each $l$, $\bar{G}^l_{ij} = G^l(x_i, y_j)$, with $G^l(x,y)$ being the $l$ times integrated kernel $G(x,y)$. Each of these multisummations can subsequently be evaluated fast relying only on the smoothness of the *continuum* kernels $G^l(x,y)$, which trivially follows from the smoothness of the *continuum* kernel $G(x,y)$. Hence, the algorithm allows fast evaluation independent of grid uniformity. This will facilitate the introduction of local refinements, wherever needed.

In addition, due to the use of the integrated kernels, the evaluation can generally be faster and the algorithm allows a minimal (even zero) number of local corrections $m$ and a minimal order of transfer $p$ on the finest grid (where the bulk of the computational work is invested), and only gradually increasing $m$ and $p$ at coarser levels, reaching values $O(ln\frac{1}{h})$ at the coarsest evaluation level. The total work needed to evaluate the transform to the level of accuracy of the employed discretization adds up to $O(s^{d+1})$ operations per gridpoint, where $s$ is the order of discretization and $d$ the dimension of the problem.

The algorithm was tested for a one dimensional model problem with $G(x,y) = \ln|y-x|$. Detailed results will be presented using both a second and fourth order discretization. For testing purposes, and to compare with [6], uniform grids covering the entire domain were considered first. This leaves the actual application to locally refined grid structures for the next step.

**2. Discretization.** For simplicity below we will restrict the description to a one dimensional problem. The generalization to $d$ dimensional problems will be discussed in section 7.

The domain $\Omega$ is subdivided into intervals $[y_j, y_{j+1}]$. This will be called the *integration*-grid. Let $\tilde{u}^h_j(y)$ denote an interpolation polynomial of order $s$, i.e. of degree $s-1$ approximating $u(y)$ on $[y_j, y_{j+1}]$. This interpolation is done from a *data*-grid of points $\{z_j\}$ on which for each point $z_j$, $u^h_j = u(z_j)$ is given. In that case $G^h_j u^h(x)$ , the contribution of the interval $[y_j, y_{j+1}]$ to the discrete integral transform $G^h u^h(x)$, is defined by:

$$G^h_j u^h(x) = \int_{y_j}^{y_{j+1}} G(x,y)\tilde{u}^h_j(y)dy \tag{2}$$

Let $G^k(x,y)$ denote a "family" of kernels defined recursively:

$$G^0(x,y) = G(x,y)$$

$$G^l = \int_x^y G^{l-1}(x,\eta)d\eta \tag{3}$$

If $G(x,y)$ is asymptotically smooth, so are all the new kernels. By "asymptotically smooth" we mean that $G(x,y)$ is increasingly smooth with increasing $|y-x|$. This is explained in more detail in section 3.

In some cases the new kernels can be computed analytically. For example for the case $G(x,y) = \ln|y-x|$:

$$G^l(x,y) = \frac{1}{l!}(y-x)^l(\ln|y-x| - \sum_{j=1}^{l}\frac{1}{j}) \tag{4}$$

However, such closed-form expressions for $G^l$ are not essential.

Integrating (2) by parts $s$ times using (3) one obtains:

$$G_j^h u^h(x) = \sum_{l=1}^{s}(-1)^l \tilde{u}_j^{h,(l-1)}(y_j)G^l(x,y_j) - \sum_{l=1}^{s}(-1)^l \tilde{u}_j^{h,(l-1)}(y_{j+1})G^l(x,y_{j+1}), \tag{5}$$

where $\tilde{u}_j^{h,(l-1)}$ denotes the $l-1$ derivative of $\tilde{u}_j^h$. Subsequently $G^h u^h(x)$ approximating $Gu(x)$ is obtained by summing up over all integration intervals:

$$G^h u^h(x) = \sum_{j=0}^{n-1} G_j^h u^h(x)$$

$$= \sum_{l=1}^{s}(-1)^l \left[\tilde{u}_0^{h,(l-1)}(y_0)G^l(x,y_0) - \tilde{u}_{n-1}^{h,(l-1)}(y_n)G^l(x,y_n)\right]$$

$$+ \sum_{l=1}^{s}(-1)^l \sum_{j=1}^{n-1}\left[\tilde{u}_j^{h,(l-1)}(y_j) - \tilde{u}_{j-1}^{h,(l-1)}(y_j)\right] G^l(x,y_j). \tag{6}$$

Hence, the discrete integral transform is the sum of a series of "boundary terms" and $s$ discrete transforms:

$$G^h u^h(x) = B^h(x) + \sum_{l=1}^{s}(-1)^l S^{h,l}(x), \tag{7}$$

where:

$$B^h(x) = \sum_{l=1}^{s}(-1)^l \left[\tilde{u}_0^{h,(l-1)}(y_0)G^l(x,y_0) - \tilde{u}_{n-1}^{h,(l-1)}(y_n)G^l(x,y_n)\right],$$

and

$$S^{h,l}(x) = \sum_{j=1}^{n-1} G^l(x,y_j)U_j^{h,l} \tag{8}$$

with

$$U_j^{h,l} = \tilde{u}_j^{h,(l-1)}(y_j) - \tilde{u}_{j-1}^{h,(l-1)}(y_j). \tag{9}$$

The discretization error, i.e. the difference between (7) and (1), in the case of a uniform grid, per unit length is bounded by

$$|\tau^h| \leq (\gamma_1 h)^s ||u^{(s)}|| \; |G|, \tag{10}$$

where $h$ is the mesh size, $||u^{(s)}||$ the maximum of the $s$ derivative of $u$ in $\Omega$, and $|G|$ stands for the average of $|G(x,y)|$ over the integration domain.

Depending on the choice of the integration grid intervals relative to the datagrid, for some $l$ all the terms in $S^{h,l}(x)$ may vanish, namely if $U_j^{h,l} = 0$ for all $j$. This for example holds for $l = 1$ if the integration intervals coincide with the intervals of the datagrid ($y_j = z_j$). In the case of a uniform grid this also holds (except perhaps at some endpoints) for any $s$ even and $l$ odd (assuming integration intervals coinciding with data grid intervals, as is natural for $s$ even) or for any $s$ odd and $l$ even (assuming integration interval endpoints coincide with data-grid *midpoints* $y_j = (z_j + z_{j-1})/2$, as is natural for $s$ odd). Hence, the number of discrete transforms to be evaluated will usually be $s/2$ if $s$ is even, and $(s+1)/2$ if $s$ is odd. In the cases where (9) does not vanish:

$$|U_j^{h,l}| = 2(\gamma_2 h)^{s-l+1} |u^{(s)}(y_j)| + O(h^{s-l+2}), \tag{11}$$

with $\gamma_2 \approx 0.5$ for a uniform grid; see Appendix A.

At this point it should be noted that the discretization of the integral as used here is the same as used in e.g. [6]. However, in [6], the common approach of condensing (6) to the form

$$G^h u^h(x) = h^d \sum_j G^h(x, y_j) u_j^h, \tag{12}$$

is used. [6] then exploits the asymptotic smoothness of $G^h(x, y_j)$ to obtain a fast evaluation algorithm. Unfortunately, the asymptotic smoothness of $G^h(x,y)$ as a function of $y$ depends sensitively on the simultaneous uniformity of the data grid and the integration grid. The core of the new algorithm is that instead of rewriting (6) into such a single discrete transform, it is maintained in its form, and each of the discrete transforms $S^{h,l}(x)$ for which the aforementioned cancellations do not occur, is evaluated separately. As a result the fast evaluation will use only the asymptotic smoothness of the given *continuum* kernel $G(x,y)$, and the asymptotic smoothness of the continuum kernels $G^l(x,y)$ which trivially follows from it, *independently of grid uniformity*. This will facilitate the introduction of local refinements, wherever needed. Also, the evaluations will be *faster*, since we will be able to differentially use the smallness of $U_j^{h,l}$ for small $l$ (see (11)), and the reduced degree of singularity in $G^l$ for larger $l$ (see (4)).

**3. Kernel Softening.** We assume that the kernel $G(x,y)$ is *asymptotically smooth*. By this we mean that $G(x,y)$ is increasingly smooth for larger $|y-x|$, in such a way that for any given "allowed error" $\epsilon > 0$ there exist nonnegative integers $m$ and $p$ for which a "softened kernel" $G_H(x,y)$ can be defined at any "softening scale" $H > 0$, with the following two properties.

TABLE 1

*For the kernel* $G^2(x,y) = \frac{1}{2}(y-x)^2(\ln|y-x| - \frac{3}{2})$ *the* $p$*−order softened kernel* $G^2_H(x,y)$ *in the interval* $|y-x| \le mH$ *is given by* $\tilde{G}^2_H(x,y) = \frac{\xi^2}{2}\ln(mH) + (mH)^2\sum_{j=0}^{p-1} A_j(\frac{\xi}{mH})^{2j}$, *where* $\xi = |y-x|$. *In the table MAX denotes the maximum of* $|(mH)^p \tilde{G}^{2,(p)}_H(x,y)|$.

| p | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
|---|---|---|---|---|---|---|---|---|---|
| $A_0$ | $-\frac{1}{4}$ | $-\frac{1}{8}$ | $-\frac{1}{12}$ | $-\frac{1}{16}$ | $-\frac{1}{20}$ | $-\frac{1}{24}$ | $-\frac{1}{28}$ | $-\frac{1}{32}$ | $-\frac{1}{36}$ |
| $A_1$ | $-\frac{1}{2}$ | $-\frac{3}{4}$ | $-\frac{7}{8}$ | $-\frac{23}{24}$ | $-\frac{49}{48}$ | $-\frac{257}{240}$ | $-\frac{89}{80}$ | $-\frac{643}{560}$ | $-\frac{1321}{1120}$ |
| $A_2$ | | $\frac{1}{8}$ | $\frac{1}{4}$ | $\frac{3}{8}$ | $\frac{1}{2}$ | $\frac{5}{8}$ | $\frac{3}{4}$ | $\frac{7}{8}$ | 1 |
| $A_3$ | | | $-\frac{1}{24}$ | $-\frac{1}{8}$ | $-\frac{1}{4}$ | $-\frac{5}{12}$ | $-\frac{5}{8}$ | $-\frac{7}{8}$ | $-\frac{7}{6}$ |
| $A_4$ | | | | $\frac{1}{48}$ | $\frac{1}{12}$ | $\frac{5}{24}$ | $\frac{5}{12}$ | $\frac{35}{48}$ | $\frac{7}{6}$ |
| $A_5$ | | | | | $-\frac{1}{80}$ | $-\frac{1}{16}$ | $-\frac{3}{16}$ | $-\frac{7}{16}$ | $-\frac{7}{8}$ |
| $A_6$ | | | | | | $\frac{1}{120}$ | $\frac{1}{20}$ | $\frac{7}{40}$ | $\frac{7}{15}$ |
| $A_7$ | | | | | | | $-\frac{1}{168}$ | $-\frac{1}{24}$ | $-\frac{1}{6}$ |
| $A_8$ | | | | | | | | $\frac{1}{224}$ | $\frac{1}{28}$ |
| $A_9$ | | | | | | | | | $-\frac{1}{288}$ |
| *MAX* | 1 | 3 | 9 | 50 | 390 | 3864 | $4.7\ 10^4$ | $6.5\ 10^5$ | $1.03\ 10^7$ |

TABLE 2

*For the kernel* $G^4(x,y) = \frac{1}{24}(y-x)^4(\ln|y-x| - \frac{25}{12})$ *the* $p$*−order softened kernel* $G^4_H(x,y)$ *in the interval* $|y-x| \le mH$ *is given by* $\tilde{G}^4_H(x,y) = \frac{\xi^4}{24}\ln(mH) + (mH)^4\sum_{j=0}^{p-1} A_j(\frac{\xi}{mH})^{2j}$, *for* $p > 2$. *In the table MAX denotes the maximum of* $|(mH)^p \tilde{G}^{4,(p)}_H(x,y)|$.

| p | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
|---|---|---|---|---|---|---|---|---|---|
| $A_0$ | | $\frac{1}{96}$ | $\frac{1}{288}$ | $\frac{1}{576}$ | $\frac{1}{960}$ | $\frac{1}{1440}$ | $\frac{1}{2016}$ | $\frac{1}{2688}$ | $\frac{1}{3456}$ |
| $A_1$ | | $-\frac{1}{24}$ | $-\frac{1}{48}$ | $-\frac{1}{72}$ | $-\frac{1}{96}$ | $-\frac{1}{120}$ | $-\frac{1}{144}$ | $-\frac{1}{168}$ | $-\frac{1}{192}$ |
| $A_2$ | | $-\frac{1}{18}$ | $-\frac{11}{144}$ | $-\frac{25}{288}$ | $-\frac{3}{32}$ | $-\frac{19}{192}$ | $-\frac{33}{320}$ | $\frac{307}{2880}$ | $-\frac{2209}{20160}$ |
| $A_3$ | | | $\frac{1}{144}$ | $\frac{1}{72}$ | $\frac{1}{48}$ | $\frac{1}{36}$ | $\frac{5}{144}$ | $\frac{1}{24}$ | $\frac{7}{144}$ |
| $A_4$ | | | | $-\frac{1}{576}$ | $-\frac{1}{192}$ | $-\frac{1}{96}$ | $-\frac{5}{288}$ | $-\frac{5}{192}$ | $-\frac{7}{192}$ |
| $A_5$ | | | | | $\frac{1}{1440}$ | $\frac{1}{360}$ | $\frac{1}{144}$ | $\frac{1}{72}$ | $\frac{7}{288}$ |
| $A_6$ | | | | | | $-\frac{1}{2880}$ | $-\frac{1}{576}$ | $-\frac{1}{192}$ | $-\frac{7}{576}$ |
| $A_7$ | | | | | | | $\frac{1}{5040}$ | $\frac{1}{840}$ | $\frac{1}{240}$ |
| $A_8$ | | | | | | | | $-\frac{1}{8064}$ | $-\frac{1}{1152}$ |
| $A_9$ | | | | | | | | | $\frac{1}{12096}$ |
| *MAX* | | 4/3 | 11/6 | 5/3 | 15 | 126 | 1274 | $1.5\ 10^4$ | $2.1\ 10^5$ |

(i) *Locality*: $G_H(x,y) = G(x,y)$ for $|y-x| \geq mH$.

(ii) $G_H(x,y)$ is *suitably smooth* on the scale $H$. By this it is meant that, both as a function of $x$ for any fixed $y$, and as a function of $y$ for any fixed $x$, $G_H(x,y)$ can be approximated up to an error smaller than $\epsilon$ by a $p$-order interpolation from its values on any uniform grid with meshsize $H$ (or smaller). This translates into the requirement that

$$(\gamma_3 H)^p |G_H^{(p)}(x,y)| \leq O(\epsilon)$$

for any $(x,y)$, where $G^{(p)}(x,y)$ stands for any $p$-order derivative of $G_H$ with respect to either $x$ or $y$, and $\gamma_3$ is a constant depending on the interpolation geometry, $\gamma_3 = 1/2$ for the usual central interpolations.

With the exception of oscillatory kernels, treated in [7], most kernels arising in physics are smooth in a way that a "softening" satisfying (i) and (ii) can easily be provided with $m$ and $p$ rising only slowly for decreasing $\epsilon$. In particular, a convenient softening is obtained by defining

$$G_H(x,y) = \begin{cases} \tilde{G}_H(x,y) & \text{if } |y-x| \leq mH \\ G(x,y) & \text{otherwise,} \end{cases} \tag{13}$$

where $\tilde{G}_H(x,y)$ is a $(2p-1)$ degree polynomial of $(y-x)$ such that $G_H(x,y)$ is continuous and has $p-1$ continuous derivatives at $y-x = \pm mH$.

For the family of kernels $G^l(x,y)$ defined by (4), softened kernels exist with $m = O(\ln \frac{1}{\epsilon})$ and $p = O(\ln \frac{1}{\epsilon})$. For these kernels the polynomial $\tilde{G}_H^l$ can be written as:

$$\tilde{G}_H^l(x,y) = \frac{(y-x)^l}{l!} \ln(mH) + (mH)^l \sum_{k=0}^{p-1} A_k \left(\frac{y-x}{mH}\right)^{2k+odd} \tag{14}$$

with $odd = 0$ if $l$ is even, and $odd = 1$ if $l$ is odd. The coefficients $A_k$ are $H$-independent; for $l = 2$ and $l = 4$ and for $2 \leq p \leq 10$ they are given in Tables 1 and 2, together with the maximum of the $p$–order derivative of $\tilde{G}_H^l(x,y)$. An illustration of kernel softening appears in Figure 1.

Let $H$ denote the mesh size of any grid coarser than $h$ and let $\{Y_J\}$ be its grid-points. The softening property (ii) implies that the value of $G_H(x,y)$ can be interpolated from $G_H(x,Y_J)$ with only $O(\epsilon)$ error. In particular, for any $y_j$ on grid $h$ there are interpolation weights $w_{jJ}^{hH}$ such that:

$$G_H(x,y_j) = \sum_J w_{jJ}^{hH} G_H(x,Y_J) + O(\epsilon) \tag{15}$$

for all $x$. Notice that the summation actually only extends over just $p$ terms (e.g., the terms for which $|y_j - Y_J| < pH/2$, if even $p$ and central interpolation are used).

In the same way, if $\{x_i\}$ are the points of a grid $h$ and $\{X_I\}$ are the points of a grid $H$ coarser than $h$, there are interpolation weights $\bar{w}_{iI}^{hH}$ such that for all $y$:

$$G_H(x_i,y) = \sum_I \bar{w}_{iI}^{hH} G_H(X_I,y) + O(\epsilon). \tag{16}$$

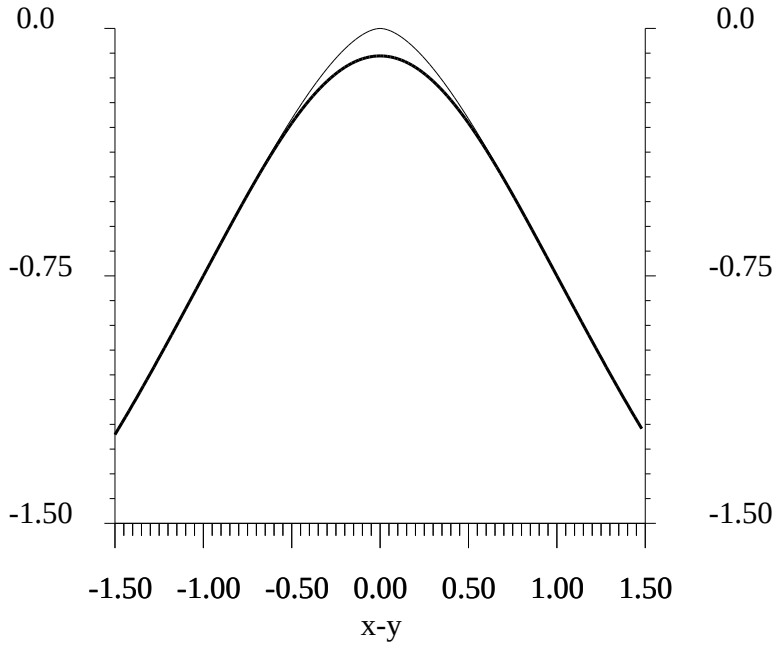


FIG. 1. *Illustration of kernel softening.* $G^2(x,y)$ *(thin line) and* $G^2_H(x,y)$ *(thick line) as a function of* $y-x$ *for* $mH=1$, $p=4$.

For smallest errors, these $p$—order interpolations should preferably be central. Near the boundaries non-central interpolations can be used such that all interpolation points are within the problem domain $\Omega$. However, usually $G(x,y)$ is well defined beyond the boundaries, and central interpolations can be used throughout.

**4. Fast evaluation of the discrete transforms.** Let $G^l_H(x,y)$ denote the softened kernel of $G^l(x,y)$ on the scale $H$. The integration grid points on this scale will be denoted with $\{Y_J\}$. Using softening requirement (I) the discrete transform $S^{h,l}(x)$ (8) can be written as:

$$S^{h,l}(x) = S^{h,l}_H(x) + M^{h,l}(x), \tag{17}$$

with

$$S^{h,l}_H(x) = \sum_{j=1}^{n-1} G^l_H(x,y_j) U^{h,l}_j \tag{18}$$

$$M^{h,l}(x) = \sum_{|y-x|\le mH} (G^l(x,y_j) - G^l_H(x,y_j)) U^{h,l}_j. \tag{19}$$

Assuming $m \ll n$, the main computational task of evaluating $S^{h,l}(x)$ now is the evaluation of the *softened-kernel transform* (18). From softening property (ii), i.e. equation (15), it follows that, neglecting $O(\epsilon)$ errors:

$$S_H^{h,l}(x) = \sum_{j=1}^{n-1} \sum_J w_{jJ}^{hH} G_H^l(x, Y_J)\, U_j^{h,l}. \tag{20}$$

Changing the order of summation gives

$$S_H^{h,l}(x) = \sum_{J=0}^{N} G_H^l(x, Y_J) U_J^{H,l}, \tag{21}$$

where

$$U_J^{H,l} = \sum_{j=0}^{n} w_{jJ}^{hH} U_j^{h,l}. \tag{22}$$

Notice that, for a given $J$, the summation over $j$ actually only involves the points $j$ for which $|y_j - y_J| < pH/2$, hence it is local. $U^{H,l}$ is simply the restriction of $U^{h,l}$ to the coarse (integration)grid $H$, a procedure referred to as *anterpolation* in [7], since it is the *adjoint* of *interpolation* (15).

Next, let $\{x_i\}$ denote the points of the evaluation grid $h$, i.e. the points in which the integral transform is to be computed, and let $\{X_I\}$ be the points of a similar grid but coarser, with a mesh size $H$. By the smoothness of $G_H^l(x, y)$ with respect to the $x$ variable, it follows from (16) that up to an $O(\epsilon)$ error, for each point $x_i$:

$$S_H^{h,l}(x_i) = \sum_I \bar{w}_{iI}^{hH} S_H^{H,l}(X_I), \tag{23}$$

where

$$S_H^{H,l}(X_I) = \sum_{J=0}^{N} G_H^l(X_I, Y_J) U_J^{H,l}. \tag{24}$$

Summarizing, the level-$h$ (grid $h$) multisummation task (17) to obtain $S^{h,l}(x_i)$ for all $x_i$ in the level-$h$ evaluation grid, is replaced by:

1. anterpolation from the level-$h$ integration grid points $y_j$ to the level- $H$ integration grid points $Y_J$, according to equation (22).

2. coarse grid multi-summation, i.e. (24).

3. interpolation of the result of this summation from the level-$H$ evaluation grid points $X_I$ to the level-$h$ evaluation grid points $x_i$, i.e. equation (23).

4. Addition of the local corrections $M^{h,l}(x_i)$ as defined by equation (19)

So far coarser grids (with respect to $x$ and $y$) with mesh size $H$ were assumed. However, $H$ can not be chosen arbitrarily. For example $H$ must be sufficiently close to $h$, to keep the evaluation of (19) inexpensive. In this respect $H = 2h$ is often convenient. However, if the number of nodes on grid $h$ is $n$, than the number of nodes

on grid $H$ will be roughly $N = n/2$, which may still be too large to directly calculate (24). In that case, with one minor modification, the procedure described above to obtain $S^{h,l}(x_i)$ can also be applied to calculate $S^{2h,l}_{2h}(X_I)$. Indeed, equation (24) can be written as

$$S^{2h,l}_{2h}(X_I) = S^{2h,l}_{4h}(X_I) + M^{2h,l}(X_I), \tag{25}$$

where

$$S^{2h,l}_{4h}(x_I) = \sum_{J=0}^{N} G^l_{4h}(X_I, Y_J) U^{2h,l}_J, \tag{26}$$

and $M^{2h,l}(X_I)$ is a grid $2h$ local correction:

$$M^{2h,l}(X_I) = \sum_{|X_I - Y_J| \le 4mh} (G^l_{2h}(X_I, Y_J) - G^l_{4h}(X_I, Y_J)) U^{2h,l}_J. \tag{27}$$

Subsequently, using the smoothness of $G^l_{4h}$, direct evaluation of $S^{2h,l}_{2h}(X_I)$ using (24) can be replaced by an anterpolation of $U^{2h,l}$ to grid $4h$, a multisummation on grid $4h$ yielding $S^{4h,l}_{4h}$, interpolation of the result from evaluation grid $4h$ to evaluation grid $2h$, and addition of the grid $2h$ correction $M^{2h,l}$ as defined by (27).

The above described procedure can be repeated recursively until a grid is reached at which direct summation can be done in at most $O(n)$ operations.

**5. Parameter optimization and control.** In the previous sections the basic elements of the algorithm were described. The remaining question is the selection of $m$ and $p$ on each of the grids ($2h$, $4h$, etc) that will be used in the process of evaluating each of the transforms $S^{h,l}$. Below the basic procedure is explained to obtain optimal values of $p$ and $m$, so as to minimize computational work under the constraint that the incremental evaluation error, is smaller than an estimate for the original fine-grid discretization error ($\epsilon = O(h^s)$). For the example problem $G = \log|y - x|$ this optimization is discussed in detail in appendix B.

For any $x$ the discretization error per unit length of integration, see §2, is bounded by:

$$|\tau^h| \le (\gamma_1 h)^s \|u^{(s)}\| \, |G|. \tag{28}$$

Due to (11) and (15), the error that results from replacing (18) by (20), i.e. the incremental evaluation error resulting from transferring evaluation from grid $H/2$ to grid $H$, per unit length, is given by:

$$|\epsilon^H| = \gamma_2 (\gamma_2 h)^{s-l} |G^l_H - I^{H/2}_H G^l_H| \, \|u^{(s)}\|, \tag{29}$$

where $G^l_H(x,y) - I^{H/2}_H G^l_H(x,y)$ stands for the order $p$ interpolation error made with the interpolation of $G^l_H$ from grid $H$ to a point $x$ or a point $y$ of grid $H/2$, and $|G^l_H - I^{H/2}_H G^l_H|$ is the average of its absolute value over the integration points $y$. Generally this error is bounded by

$$|G_H^l - I\!I_H^{H/2} G_H^l| < (\gamma_3 H)^p |G_H^{l(p)}|, \tag{30}$$

where $|G_H^{l(p)}|$ stands for the average of the absolute value of the $p^{th}$ derivative of $G_H^l$, and $\gamma_3$ depends on the geometry of the interpolation, i.e., $\gamma_3 = 1/2$ for central interpolation. Generally, by (28), (29) $|e^H| \leq |\tau^h|$ requires:

$$(\gamma_2)^{s-l+1} h^{-l} |G_H^l - I\!I_H^{H/2} G_H^l| \leq (\gamma_1)^s \ |G|. \tag{31}$$

Under this constraint we want to minimize the incremental work $W = O(p + 4m)$ related to transferring evaluation from grid $H/2$ to grid $H$. This work estimate was obtained as follows. Taking an operation to mean a combination of one multiplication and one addition, the number of operations in transferring $U^{H/2,l}$ to grid $H$, is $p/2$ per grid $H/2$ point (since $d = 1$, for half of the values the transfer is trivial). Similarly $p/2$ operations per grid $H/2$ point are used to interpolate $S_H^{H,l}$ from grid $H$ to grid $H/2$. Finally the corrections are added which involve summations over $4m$ grid $H/2$ points.

Due to the use of integrated kernels it will usually be possible to employ $m = 0$ and a certain minimal $p$ (derived from (31) and depending only on $l$) for several of the finest coarsening $H$ ( provided the basic meshsize $h$ is sufficiently small). At large $H$ (i.e., after several coarsening stages), $m$ and $p$ will start to rise, reaching finally the typical size (e.g. $O(\log \frac{1}{\epsilon})$) which in the method of [6] must be used at *all* coarsenings stages. The rates at which $p$ and $m$ increase need not actually be calculated very precisely: If they increase faster than the necessary rates, no substantial harm (i.e., increased work) is done, since most of the work is anyway still spent at the finest levels with $m = 0$ and a minimal $p$.

For the case $G = \ln|y - x|$ this optimization, see appendix B, yields that $p$ should be taken the maximum of the first non-negative integer satisfying

$$p \geq -0.83 \ln(g) + l + 1, \tag{32}$$

and $l + 1$, and $m$ the first integer satisfying

$$m \geq 1.23(p - l - 1), \tag{33}$$

where

$$g = \frac{\gamma_1^s}{\gamma_3^{l+1} \gamma_2^{s-l+1}} \frac{h^l}{H^{l+1}} |G|. \tag{34}$$

From (34) it can be seen that:

$$\ln(g) = c_l + l \ \ln(h) - (l + 1) \ln(H), \tag{35}$$

with $c_l$ some constant depending on the geometry of the interpolation, order of discretization, and $l$. From (32), (33), and (35) it follows that, for a given finest grid mesh size $h$, $m$ and $p$ will increase logarithmically with increasing coarse grid mesh

size. On the other hand, with decreasing $H$, both $m$ and $p$ decrease, and for $H \approx h$ they reach the limits $m = 0$ and $p \geq l + 1$.

The implications for the amount of work performed to obtain the discrete transform are investigated below. $p + 4m$ operations are performed per finest grid point in the anterpolation to grid $2h$, the interpolation from grid $2h$ and the grid $h$ correction. Adding to this the work performed on all the coarser grids, with mesh sizes $4h$, .. $2^t h$, then, assuming the actual multisummation is performed on a grid with $H = h^{1/2}$, the total work per fine grid point in evaluating $S^{h,l}$ will be:

$$W = 1 + \sum_{j=0}^{t} (p_j + 4m_j) 2^{-j}, \tag{36}$$

where $p_j$ and $m_j$ denote $p$ and $m$ from equations (32), and (33) using $H = 2^j h$ in (35). With decreasing $h$, on an increasing number grids: $m = 0$ and $p = p_{min} \geq l + 1$. As a result the work per grid point will tend to $2p_{min} + 1$.

**6. Numerical Results.** As a first model problem the following integral transform was taken:

$$Gu(x) = \int_{-1}^{1} \ln|y - x| u(y) dy \tag{37}$$

with $u(y) = (1 - y^2)$. For this example the integral transform $Gu(x)$ can be computed analytically, which enables a detailed check of the accuracy of the multilevel algorithm. The integration grid, data grid, and evaluation grid were chosen to be the same uniform grid with mesh size $h$, ($y_i = z_i = x_i$ for $0 \leq i \leq n$). In that case equation (6) for $s = 2$ (piecewise linear) yields:

$$G^h u^h(x_i) = B^h(x_i) + S^{h,2}(x_i), \tag{38}$$

where

$$S^{h,2}(x_i) = \sum_{j=1}^{n-1} G^2(x_i, y_j) U_j^{h,2} \tag{39}$$

with

$$U_j^{h,2} = \frac{1}{h}(u_{j-1}^h - 2u_j^h + u_{j+1}^h), \tag{40}$$

and

$$\begin{aligned} B^h(x_i) \quad = \quad & u_n^h G^1(x_i, y_n) - u_0^h G^1(x_i, y_0) + \\ & \frac{1}{h}(u_1^h - u_0^h) G^2(x_i, y_0) - \\ & \frac{1}{h}(u_n^h - u_{n-1}^h) G^2(x_i, y_n). \end{aligned} \tag{41}$$

$B^h(x_i)$ was computed straightforwardly. $S^{h,2}(x_i)$ for all $x_i$ was computed using the fast evaluation algorithm explained in section 4 on a sequence of grids with mesh sizes $h$, $2h$ etc. These grids will be referred to as levels and are numbered, starting with the coarsest grid that will be called level 1, the next finer grid being level 2, etc. For the present calculations the coarsest grid (level 1) had $8+1$ gridpoints (mesh size $1/4$), the second coarsest $16+1$, (mesh size $1/8$), etc. The finest level, with $n+1$ gridpoints, is denoted by $k$, thus $n = 2^{k+2}$. To monitor the accuracy of the fast evaluation vs. the discretization error we define the error $E_k^r$ as the average absolute difference between the analytical integral transform and the discrete integral transform obtained on level $k$ when the multisummation itself is carried out on level $r$, ($r \leq k$):

$$E_k^r = \frac{1}{n+1} \sum_{i=0}^{n} |(G^h u^h)^{k,r}(x_i) - Gu(x_i)|. \tag{42}$$

In particular $E_k^k$ is the $L_1$ norm of the discretization error on grid $k$. The minimum objective to be achieved is $E_k^r \approx E_k^k$ for $r \approx k/2$ such that the multisummation on level $r$ requires at most $O(n)$ operations.

As explained in section 5, the softening order $p$ and softening width $m$ depend on the mesh size. First $p'$ was computed using (32) with $l = 2$ and $c_l = 0$ in (35). Then $p$ was obtained from:

$$p = \max(round(p'), p_{min}) \tag{43}$$

and set to the nearest (larger) even value. Subsequently $m$ was obtained from:

$$m = \begin{cases} round(1.23(p' - l - 1)) & \text{if } p' \geq p_{min} \\ 0 & \text{otherwise.} \end{cases} \tag{44}$$

Table 3 and Table 4 give $E_k^r$ obtained in numerical tests with such parameters, for the case $s = 2$, i.e. $2^{nd}$ order discretization. Indeed, the leftmost column giving the discretization error $E_k^k$ confirms its second order. The results in the tables marked by asterisks denote the cases where gridlevel $r$, the summation grid, consists of $n^{1/2} + 1$ points. The tables clearly show that with the fast evaluation algorithm as presented, the integral transform can be computed executing the multisummation on a grid with $O(n^{1/2})$ points at negligable loss of accuracy.

To get a better insight into the error introduced by the fast evaluation at the various levels we have also monitored the *incremental* error defined as:

$$IE_k^r = \frac{1}{n+1} \sum_{i=0}^{n} |(G^h u^h)^{k,r}(x_i) - (G^h u^h)^{k,r+1}(x_i)|. \tag{45}$$

For a given $k, r$ this quantity can be explained as the additional error introduced by the coarsening step from $r+1$ to $r$. Table 5 and Table 6 display this quantity for the cases presented in Table 3 and Table 4. It shows that the evaluation errors are in most relevant cases ($k \leq r \leq k/2$), one or several orders of magnitude smaller than the discretization error.

TABLE 3
*Error* $\frac{1}{n+1}\sum_{i=0}^{i=n}|(G^h u^h)^{k,r}(x_i) - (Gu)(x_i)|$ *for the second order* ($s=2$) *model problem.*

| $k$ | $r=k$ | $r=k-1$ | $k-2$ | $k-3$ | $k-4$ | $k-5$ |
|---|---|---|---|---|---|---|
| 2 | $3.92\,10^{-3}$ | $3.76\,10^{-3}$ | | | | |
| 3 | $1.02\,10^{-3}$ | $9.62\,10^{-4}$ | $1.63\,10^{-3}$ | | | |
| 4 | $2.58\,10^{-4}$ | $2.49\,10^{-4}$ | $3.03\,10^{-4}$ | $*\,3.75\,10^{-4}$ | | |
| 5 | $6.51\,10^{-5}$ | $6.37\,10^{-5}$ | $5.45\,10^{-5}$ | $5.85\,10^{-5}$ | $8.58\,10^{-5}$ | |
| 6 | $1.63\,10^{-5}$ | $1.62\,10^{-5}$ | $1.48\,10^{-5}$ | $1.92\,10^{-5}$ | $*\,2.09\,10^{-5}$ | $2.34\,10^{-5}$ |
| 7 | $4.10\,10^{-6}$ | $4.07\,10^{-6}$ | $3.89\,10^{-6}$ | $4.26\,10^{-6}$ | $4.47\,10^{-6}$ | $4.77\,10^{-6}$ |
| 8 | $1.03\,10^{-6}$ | $1.02\,10^{-6}$ | $9.98\,10^{-7}$ | $8.13\,10^{-7}$ | $8.19\,10^{-7}$ | $*\,9.24\,10^{-7}$ |
| 9 | $2.56\,10^{-7}$ | $2.56\,10^{-7}$ | $2.53\,10^{-7}$ | $2.29\,10^{-7}$ | $2.62\,10^{-7}$ | $2.68\,10^{-7}$ |
| 10 | $6.41\,10^{-8}$ | $6.41\,10^{-8}$ | $6.37\,10^{-8}$ | $6.06\,10^{-8}$ | $6.37\,10^{-8}$ | $6.34\,10^{-8}$ |
| 11 | $1.60\,10^{-8}$ | $1.60\,10^{-8}$ | $1.60\,10^{-8}$ | $1.56\,10^{-8}$ | $1.25\,10^{-8}$ | $1.24\,10^{-8}$ |
| 12 | $\approx 4\quad 10^{-9}$ | | $4.00\,10^{-9}$ | $3.95\,10^{-9}$ | $3.56\,10^{-9}$ | $3.89\,10^{-9}$ |
| 13 | $\approx 1\quad 10^{-9}$ | | | $9.95\,10^{-10}$ | $9.46\,10^{-10}$ | $9.82\,10^{-10}$ |
| 14 | $\approx 2.5\,10^{-10}$ | | | | $2.43\,10^{-10}$ | $1.94\,10^{-10}$ |
| 15 | $\approx 6\quad 10^{-11}$ | | | | | $5.56\,10^{-11}$ |
| 16 | $\approx 1.5\,10^{-11}$ | | | | | |
| 17 | $\approx 4\quad 10^{-12}$ | | | | | |
| 18 | $\approx 1\quad 10^{-12}$ | | | | | |

TABLE 4
*Error* $\frac{1}{n+1}\sum_{i=0}^{i=n}|(G^h u^h)^{k,r}(x_i) - (Gu)(x_i)|$ *for the second order* ($s=2$) *model problem.*

| $k$ | $r=k$ | $r=k-6$ | $k-7$ | $k-8$ | $k-9$ | $k-10$ |
|---|---|---|---|---|---|---|
| 7 | $4.10\,10^{-6}$ | $5.36\,10^{-6}$ | | | | |
| 8 | $1.03\,10^{-6}$ | $1.08\,10^{-6}$ | $1.23\,10^{-6}$ | | | |
| 9 | $2.56\,10^{-7}$ | $2.78\,10^{-7}$ | $3.16\,10^{-7}$ | $3.51\,10^{-7}$ | | |
| 10 | $6.41\,10^{-8}$ | $*\,6.46\,10^{-8}$ | $6.68\,10^{-8}$ | $7.56\,10^{-8}$ | $9.21\,10^{-8}$ | |
| 11 | $1.60\,10^{-8}$ | $1.28\,10^{-8}$ | $1.37\,10^{-8}$ | $1.39\,10^{-8}$ | $1.48\,10^{-8}$ | $2.11\,10^{-8}$ |
| 12 | $\approx 4\quad 10^{-9}$ | $3.91\,10^{-9}$ | $*\,3.95\,10^{-9}$ | $4.15\,10^{-9}$ | $4.21\,10^{-9}$ | $4.42\,10^{-9}$ |
| 13 | $\approx 1\quad 10^{-9}$ | $9.63\,10^{-10}$ | $9.68\,10^{-10}$ | $9.77\,10^{-10}$ | $1.02\,10^{-9}$ | $1.08\,10^{-9}$ |
| 14 | $\approx 2.5\,10^{-10}$ | $1.91\,10^{-10}$ | $1.93\,10^{-10}$ | $*\,1.98\,10^{-10}$ | $1.98\,10^{-10}$ | $2.01\,10^{-10}$ |
| 15 | $\approx 6\quad 10^{-11}$ | $5.98\,10^{-11}$ | $5.98\,10^{-11}$ | $6.00\,10^{-11}$ | $6.09\,10^{-11}$ | $6.10\,10^{-11}$ |
| 16 | $\approx 1.5\,10^{-11}$ | $1.53\,10^{-11}$ | $1.49\,10^{-11}$ | $1.49\,10^{-11}$ | $*\,1.49\,10^{-11}$ | $1.52\,10^{-11}$ |
| 17 | $\approx 4\quad 10^{-12}$ | | $2.98\,10^{-12}$ | $2.98\,10^{-12}$ | $3.00\,10^{-12}$ | $3.00\,10^{-12}$ |
| 18 | $\approx 1\quad 10^{-12}$ | | | $9.28\,10^{-13}$ | $9.28\,10^{-13}$ | $*\,9.32\,10^{-13}$ |

As mentioned above, $m$ and $p$ depend on the mesh size. As an example Table 7 gives the values of $p$ and $m$ used on the different grids $k-1$, $k-2$, $k-3$ (or in terms of section 4 grids $2h$, $4h$, $8h$) used to obtain $S^{h,2}(x_i)$ as part of the computation of $(G^h u^h)^{k,r}(x_i)$, for $k=12$ as presented in Table 3 and Table 4. Indeed, on the finest grid, and several of the coarser grids no corrections are needed at all. For larger $k$ the number of such grids only increases. This is reflected in the amount of work needed to obtain the discrete transform, as is illustrated in Table 8 which shows the amount of work per finest grid point invested to obtain $S^{h,2}(x_i)$ for all $x_i$, as a function of the gridlevel $k$ and the summation grid $r$. Notice that the leftmost column, $k=r$ represents the amount of work per gridpoint if $S^{h,2}(x_i)$ is computed by a simple multi-summation on grid level $k$. As explained in section 5, asymptotically, i.e. for large enough $k$, the work per gridpoint should tend to $2p_{min}+1$. As $p_{min}=4$ the present results clearly satisfy this prediction. The total work per grid (h) point invested to obtain the discrete transform $G^h u^h$ itself can be obainted from Table 8 by adding the additional operations needed for the evaluation of $B^h(x_i)$.

TABLE 5

*Incremental error* $\frac{1}{n+1}\sum_{i=0}^{i=n}|(G^h u^h)^{k,r}(x_i) - (G^h u^h)^{k,r+1}(x_i)|$ *for the second order* ($s = 2$) *model problem*

| $k$ | $r = k-1$ | $k-2$ | $k-3$ | $k-4$ | $k-5$ |
|---|---|---|---|---|---|
| 2 | $4.64\ 10^{-4}$ | | | | |
| 3 | $6.97\ 10^{-5}$ | $6.67\ 10^{-4}$ | | | |
| 4 | $1.03\ 10^{-5}$ | $5.41\ 10^{-5}$ | $*\ 1.30\ 10^{-4}$ | | |
| 5 | $1.42\ 10^{-6}$ | $1.02\ 10^{-5}$ | $8.64\ 10^{-6}$ | $4.76\ 10^{-5}$ | |
| 6 | $1.88\ 10^{-7}$ | $1.42\ 10^{-6}$ | $4.44\ 10^{-6}$ | $*\ 3.07\ 10^{-6}$ | $7.14\ 10^{-6}$ |
| 7 | $2.44\ 10^{-8}$ | $1.88\ 10^{-7}$ | $3.72\ 10^{-7}$ | $5.93\ 10^{-7}$ | $7.69\ 10^{-7}$ |
| 8 | $3.10\ 10^{-9}$ | $2.43\ 10^{-8}$ | $1.88\ 10^{-7}$ | $4.33\ 10^{-8}$ | $*\ 1.97\ 10^{-7}$ |
| 9 | $3.92\ 10^{-10}$ | $3.10\ 10^{-9}$ | $2.43\ 10^{-8}$ | $3.31\ 10^{-8}$ | $1.24\ 10^{-8}$ |
| 10 | $4.93\ 10^{-11}$ | $3.92\ 10^{-10}$ | $3.10\ 10^{-9}$ | $3.17\ 10^{-9}$ | $3.47\ 10^{-9}$ |
| 11 | $6.18\ 10^{-12}$ | $4.93\ 10^{-11}$ | $3.92\ 10^{-10}$ | $3.10\ 10^{-9}$ | $3.13\ 10^{-10}$ |
| 12 | | $6.18\ 10^{-12}$ | $4.93\ 10^{-11}$ | $3.92\ 10^{-10}$ | $3.28\ 10^{-10}$ |
| 13 | | | $6.18\ 10^{-12}$ | $4.93\ 10^{-11}$ | $3.63\ 10^{-11}$ |
| 14 | | | | $6.18\ 10^{-12}$ | $4.92\ 10^{-11}$ |
| 15 | | | | | $6.18\ 10^{-12}$ |
| 16 | | | | | |
| 17 | | | | | |
| 18 | | | | | |

TABLE 6

*Incremental error* $\frac{1}{n+1}\sum_{i=0}^{i=n}|(G^h u^h)^{k,r}(x_i) - (G^h u^h)^{k,r+1}(x_i)|$ *for the second order* ($s = 2$) *model problem.*

| $k$ | $k-6$ | $k-7$ | $k-8$ | $k-9$ | $k-10$ |
|---|---|---|---|---|---|
| 7 | $1.86\ 10^{-6}$ | | | | |
| 8 | $4.49\ 10^{-7}$ | $3.80\ 10^{-7}$ | | | |
| 9 | $2.81\ 10^{-8}$ | $1.23\ 10^{-7}$ | $1.16\ 10^{-7}$ | | |
| 10 | $*\ 3.04\ 10^{-9}$ | $7.69\ 10^{-9}$ | $3.16\ 10^{-8}$ | $4.41\ 10^{-8}$ | |
| 11 | $8.10\ 10^{-10}$ | $1.77\ 10^{-9}$ | $2.04\ 10^{-9}$ | $3.92\ 10^{-9}$ | $1.40\ 10^{-8}$ |
| 12 | $5.31\ 10^{-11}$ | $*\ 1.10\ 10^{-10}$ | $4.81\ 10^{-10}$ | $5.44\ 10^{-10}$ | $8.81\ 10^{-10}$ |
| 13 | $3.17\ 10^{-11}$ | $1.22\ 10^{-11}$ | $3.01\ 10^{-11}$ | $1.29\ 10^{-10}$ | $2.69\ 10^{-10}$ |
| 14 | $3.49\ 10^{-12}$ | $3.76\ 10^{-12}$ | $*\ 7.00\ 10^{-12}$ | $8.00\ 10^{-12}$ | $1.68\ 10^{-11}$ |
| 15 | $4.22\ 10^{-12}$ | $2.82\ 10^{-13}$ | $4.45\ 10^{-13}$ | $1.88\ 10^{-12}$ | $2.13\ 10^{-12}$ |
| 16 | $5.02\ 10^{-13}$ | $4.07\ 10^{-13}$ | $5.36\ 10^{-14}$ | $*\ 1.20\ 10^{-13}$ | $5.06\ 10^{-13}$ |
| 17 | | $5.17\ 10^{-14}$ | $2.52\ 10^{-14}$ | $3.09\ 10^{-14}$ | $3.37\ 10^{-14}$ |
| 18 | | | $6.04\ 10^{-15}$ | $5.57\ 10^{-15}$ | $*\ 1.09\ 10^{-14}$ |

TABLE 7

$m$ *and* $p$ *on the different grid levels* $k-t$ *used in the calculation of* $(G^h u^h)^{k,r}(x)$ *as presented in Table 3 and Table 4* ($k = 12$, $1 \le t \le 10$).

| $k-t$ | $p$ | $m$ |
|---|---|---|
| 11 | 4 | 0 |
| 10 | 4 | 0 |
| 9 | 4 | 0 |
| 8 | 4 | 0 |
| 7 | 4 | 1 |
| 6 | 6 | 4 |
| 5 | 8 | 6 |
| 4 | 10 | 7 |
| 3 | 12 | 10 |
| 2 | 14 | 12 |

TABLE 8
*Work per gridpoint $W$ invested to obtain $S^{h,2}$ for the second order ($s = 2$) model problem.*

| $k$ | $r = k$ | $k-1$ | $k-2$ | $k-3$ | $k-4$ | $k-5$ | $k-6$ | $k-7$ | $k-8$ | $k-9$ | $k-10$ |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 2 | 17 | 9 | | | | | | | | | |
| 3 | 33 | 13 | 10 | | | | | | | | |
| 4 | 65 | 21 | 12 | * 13 | | | | | | | |
| 5 | 129 | 37 | 14 | 12 | 13 | | | | | | |
| 6 | 257 | 69 | 22 | 12 | * 12 | 13 | | | | | |
| 7 | 513 | 133 | 38 | 16 | 12 | 12 | 13 | | | | |
| 8 | 1025 | 261 | 70 | 23 | 13 | * 11 | 11 | 12 | | | |
| 9 | 2049 | 517 | 134 | 39 | 16 | 11 | 11 | 11 | 11 | | |
| 10 | 4097 | 1028 | 262 | 71 | 24 | 13 | * 11 | 11 | 11 | 11 | |
| 11 | 8193 | 2052 | 518 | 135 | 40 | 16 | 11 | 10 | 10 | 10 | 10 |
| 12 | 1.6 $10^4$ | | 1030 | 263 | 72 | 24 | 13 | * 10 | 10 | 10 | 10 |
| 13 | 3.2 $10^4$ | | | 519 | 136 | 41 | 17 | 11 | 10 | 9 | 10 |
| 14 | 6.4 $10^4$ | | | | 264 | 72 | 24 | 13 | * 10 | 9 | 9 |
| 15 | 1.3 $10^5$ | | | | | 136 | 40 | 16 | 11 | 9 | 9 |
| 16 | 2.6 $10^5$ | | | | | | 72 | 24 | 12 | * 10 | 9 |
| 17 | 5.2 $10^5$ | | | | | | | 40 | 16 | 10 | 9 |
| 18 | 1 $10^6$ | | | | | | | | 24 | 12 | * 9 |

As a second model problem we considered (37) with $u(y) = (1-y^4)$, using a $s = 4$ discretization. Also for this example the integral transform $Gu(x)$ can be computed analytically, which facilitates a detailed check of the accuracy of the developed fast evaluation algorithm. As for the previous example the integration, data, and evaluation grids are chosen to coincide. For each point $x_i$ equation (6) with $s = 4$ (piecewise cubic) yields:

$$G^h u^h(x_i) = B^h(x_i) + S^{h,2}(x_i) + S^{h,4}(x_i), \tag{46}$$

where:

$$\begin{aligned} B^h(x_i) \;=\; & -G^1(x_i,y_0)u_0^h + \\ & +G^1(x_i,y_n)u_n^h \\ & -G^2(x_i,y_0)(11u_0^h - 18u_1^h + 9u_2^h - 2u_3^h)/(6h) \\ & -G^2(x_i,y_n)(11u_n^h - 18u_{n-1}^h + 9u_{n-2}^h - 2u_{n-3}^h)/(6h) \\ & -G^3(x_i,y_0)(2u_0^h - 5u_1^h + 4u_2^h - u_3^h)/(h^2) \\ & +G^3(x_i,y_n)(2u_n^h - 5u_{n-1}^h + 4u_{n-2}^h - u_{n-3}^h)/(h^2) \\ & -G^4(x_i,y_0)(u_0^h - 3u_1^h + 3u_2^h - u_3^h)/(h^3) \\ & -G^4(x_i,y_n)(u_n^h - 3u_{n-1}^h + 3u_{n-2}^h - u_{n-3}^h)/(h^3), \end{aligned} \tag{47}$$

$$S^{h,2}(x_i) \;=\; \sum_{j=2}^{n-2} G^2(x_i,y_j)U_j^{h,2}, \tag{48}$$

$$S^{h,4}(x_i) \;=\; \sum_{j=2}^{n-2} G^4(x_i,y_j)U_j^{h,4}, \tag{49}$$

TABLE 9
*Error* $\frac{1}{n+1}\sum_{i=0}^{i=n}|(G^h u^h)^{k,r}(x_i) - (Gu)(x_i)|$ *for the fourth order* ($s = 4$) *model problem*

| $k$ | $r = k$ | $r = k-1$ | $k-2$ | $k-3$ | $k-4$ |
|---|---|---|---|---|---|
| 2 | $1.12\ 10^{-4}$ | $1.26\ 10^{-3}$ | | | |
| 3 | $7.96\ 10^{-6}$ | $1.03\ 10^{-5}$ | $1.59\ 10^{-5}$ | | |
| 4 | $5.33\ 10^{-7}$ | $6.00\ 10^{-7}$ | $7.12\ 10^{-7}$ | $*\ 9.41\ 10^{-7}$ | |
| 5 | $3.43\ 10^{-8}$ | $3.43\ 10^{-8}$ | $3.00\ 10^{-8}$ | $3.33\ 10^{-8}$ | $5.44\ 10^{-8}$ |
| 6 | $2.18\ 10^{-9}$ | $2.17\ 10^{-9}$ | $2.03\ 10^{-9}$ | $2.63\ 10^{-9}$ | $*\ 3.03\ 10^{-9}$ |
| 7 | $1.37\ 10^{-10}$ | $1.36\ 10^{-10}$ | $1.31\ 10^{-10}$ | $1.43\ 10^{-10}$ | $1.52\ 10^{-10}$ |
| 8 | $8.58\ 10^{-12}$ | $8.55\ 10^{-12}$ | $8.39\ 10^{-12}$ | $7.04\ 10^{-12}$ | $7.18\ 12^{-12}$ |
| 9 | $5.37\ 10^{-13}$ | $5.35\ 10^{-13}$ | $5.28\ 10^{-13}$ | $4.84\ 10^{-13}$ | $5.52\ 10^{-13}$ |
| 10 | $3.29\ 10^{-14}$ | $3.19\ 10^{-14}$ | $3.23\ 10^{-14}$ | $2.95\ 10^{-14}$ | $3.20\ 10^{-14}$ |
| 11 | $\approx 2\ 10^{-15}$ | | $4.18\ 10^{-15}$ | $3.59\ 10^{-15}$ | $3.05\ 10^{-15}$ |

TABLE 10
*Error* $\frac{1}{n+1}\sum_{i=0}^{i=n}|(G^h u^h)^{k,r}(x_i) - (Gu)(x_i)|$ *for the fourth order* ($s = 4$) *model problem.*

| $k$ | $r = k$ | $k-5$ | $k-6$ | $k-7$ |
|---|---|---|---|---|
| 6 | $2.18\ 10^{-9}$ | $5.60\ 10^{-9}$ | | |
| 7 | $1.37\ 10^{-10}$ | $1.75\ 10^{-10}$ | $2.04\ 10^{-10}$ | |
| 8 | $8.58\ 10^{-12}$ | $*\ 8.23\ 10^{-12}$ | $1.15\ 10^{-11}$ | $1.82\ 10^{-11}$ |
| 9 | $5.37\ 10^{-13}$ | $5.84\ 10^{-13}$ | $6.43\ 10^{-13}$ | $1.09\ 10^{-12}$ |
| 10 | $3.29\ 10^{-14}$ | $3.15\ 10^{-14}$ | $*\ 3.37\ 10^{-14}$ | $4.27\ 10^{-14}$ |
| 11 | $\approx 2\ 10^{-15}$ | $2.66\ 10^{-15}$ | $2.54\ 10^{-15}$ | $2.16\ 10^{-15}$ |

with:

$$U_j^{h,2} = \frac{1}{6h}(-u_{j-2}^h + 4u_{j-1}^h - 6u_j^h + 4u_{j+1}^h - u_{j+2}^h), \tag{50}$$

and

$$U_j^{h,4} = \frac{1}{h^3}(u_{j-2}^h - 4u_{j-1}^h + 6u_j^h - 4u_{j+1}^h + u_{j+2}^h). \tag{51}$$

Compared with the previous example to obtain the discrete integral transform $G^h u^h$, now *two* discrete transforms $S^{h,2}(x_i)$ and $S^{h,4}(x_i)$ have to be evaluated. To each of these transforms the algorithm as described in section 4 was applied separately. The values of $m$ and $p$ on the different coarser grids used in the evaluation process of $S^{h,2}$ were computed as described above for the $s = 2$ example. The values of $m$ and $p$ used in the evaluation process of $S^{h,4}(x_i)$ were obtained in the same way, but for this case $p_{min}$ was set to 6, the maximum $p$ was set to 16, and $c_4 = -2$ was used.

Table 9 through 14 show the results in the same way as for the second order example. In Table 9 and Table 10 $E_k^r$ is given as a function of $r$ and $k$. The left-most column of these tables, $E_k^k$, shows the anticipated $4^{th}$ order convergence of the discrete transform to the analytical transform. Each time the mesh size is halved the discretization error decreases by a factor 16. The results marked by an asterisk indicate the cases were the summation grid $r$ contains $n^{1/2} + 1$ points. As for the previous example for sufficiently dense fine grids the algorithm allows $E_k^r \approx E_k^k$ with level $O(n^{1/2})$ points on the summation level.

Table 11 and Table 12 give the incremental error $IE_k^r$ as defined by (45). These tables indicates that the selected $m$ and $p$ for relevant $r$ ensure that the evaluation error is smaller than the discretization error, although not by an order of magnitude

TABLE 11

*Incremental error* $\frac{1}{n+1}\sum_{i=0}^{i=n}|(G^h u^h)^{k,r}(x_i) - (G^h u^h)^{k,r+1}(x_i)|$ *for the fourth order* ($s = 4$) *model problem.*

| $k$ | $k-1$ | $k-2$ | $k-3$ | $k-4$ |
|---|---|---|---|---|
| 2 | $1.15\ 10^{-3}$ | | | |
| 3 | $2.68\ 10^{-6}$ | $5.72\ 10^{-6}$ | | |
| 4 | $7.60\ 10^{-8}$ | $1.15\ 10^{-7}$ | $*\ 4.78\ 10^{-7}$ | |
| 5 | $6.47\ 10^{-10}$ | $5.43\ 10^{-9}$ | $7.71\ 10^{-9}$ | $3.80\ 10^{-8}$ |
| 6 | $1.75\ 10^{-11}$ | $2.04\ 10^{-10}$ | $6.58\ 10^{-10}$ | $*\ 1.10\ 10^{-9}$ |
| 7 | $1.13\ 10^{-12}$ | $6.21\ 10^{-12}$ | $1.32\ 10^{-11}$ | $2.81\ 10^{-11}$ |
| 8 | $3.85\ 10^{-14}$ | $1.83\ 10^{-13}$ | $1.55\ 10^{-12}$ | $6.82\ 10^{-13}$ |
| 9 | $5.57\ 10^{-15}$ | $8.43\ 10^{-15}$ | $4.77\ 10^{-14}$ | $7.48\ 10^{-14}$ |
| 10 | $1.31\ 10^{-14}$ | $5.83\ 10^{-15}$ | $5.32\ 10^{-15}$ | $3.31\ 10^{-15}$ |
| 11 | | $1.31\ 10^{-14}$ | $5.34\ 10^{-15}$ | $4.43\ 10^{-15}$ |

TABLE 12

*Incremental error* $\frac{1}{n+1}\sum_{i=0}^{i=n}|(G^h u^h)^{k,r}(x_i) - (G^h u^h)^{k,r+1}(x_i)|$ *for the fourth order* ($s = 4$) *model problem.*

| $k$ | $k-5$ | $k-6$ | $k-7$ |
|---|---|---|---|
| 6 | $4.09\ 10^{-9}$ | | |
| 7 | $7.80\ 10^{-11}$ | $8.48\ 10^{-11}$ | |
| 8 | $*\ 2.65\ 10^{-12}$ | $1.34\ 10^{-11}$ | $1.18\ 10^{-11}$ |
| 9 | $8.94\ 10^{-14}$ | $4.02\ 10^{-13}$ | $7.62\ 10^{-13}$ |
| 10 | $3.66\ 10^{-15}$ | $*\ 7.48\ 10^{-15}$ | $2.54\ 10^{-14}$ |
| 11 | $3.03\ 10^{-15}$ | $2.95\ 10^{-15}$ | $2.14\ 10^{-15}$ |

TABLE 13

$m$ *and* $p$ *on the different grid levels* $k-t$ *used in the calculation of* $(G^h u^h)^{k,r}(x)$ *as presented in Table 9 and Table 10* ($k = 10$, $1 \le t \le 7$).

| | $S^{h,2}$ | | $S^{h,4}$ | |
|---|---|---|---|---|
| $k-t$ | $p$ | $m$ | $p$ | $m$ |
| 9 | 4 | 0 | 6 | 0 |
| 8 | 4 | 0 | 6 | 1 |
| 7 | 4 | 0 | 10 | 5 |
| 6 | 4 | 1 | 12 | 9 |
| 5 | 6 | 2 | 16 | 12 |
| 4 | 8 | 5 | 16 | 16 |
| 3 | 10 | 7 | 16 | 21 |

TABLE 14

*Work per gridpoint* $W$ *invested to obtain* $S^{h,2}$ *and* $S^{h,4}$ *for the fourth order* ($s = 4$) *model problem.*

| $k$ | $r=k$ | $r=k-1$ | $k-2$ | $k-3$ | $k-4$ | $k-5$ | $k-6$ | $k-7$ |
|---|---|---|---|---|---|---|---|---|
| 2 | 2x17 | 37 | | | | | | |
| 3 | 2x33 | 38 | 46 | | | | | |
| 4 | 2x65 | 54 | 50 | $*\ 59$ | | | | |
| 5 | 2x129 | 80 | 48 | 51 | 59 | | | |
| 6 | 2x257 | 144 | 61 | 49 | $*\ 53$ | 58 | | |
| 7 | 2x513 | 268 | 89 | 53 | 49 | 53 | 56 | |
| 8 | 2x 1025 | 524 | 149 | 62 | 46 | $*\ 46$ | 48 | 50 |
| 9 | 2x 2049 | 1036 | 277 | 94 | 53 | 46 | 47 | 48 |
| 10 | 2x 4097 | 2069 | 530 | 154 | 65 | 46 | $*\ 43$ | 43 |
| 11 | 2x 8193 | | 1042 | 281 | 94 | 50 | 41 | 40 |

as in many of the cases for the $s = 2$ model problem. As an illustration, Table 13 gives an example of the values of $m$ and $p$ used on the different grids in the evaluation of $S^{h,2}(x_i)$ and $S^{h,4}(x_i)$ as part of the computation of $(G^h u^h)^{k,r}$, for $k = 10$. For both discrete transforms the tendency towards $m = 0$ and $p = p_{min}$ on the finest grids (where most of the work is invested) is clearly shown.

Finally Table 14 shows the total number of operations per gridpoint used in the evaluation of the two discrete transforms. Based on work estimate given in section 5 one would expect the work per gridpoint to tend to $2(p^2_{min} + p^4_{min}) + 2$ if $p^2_{min}$ and $p^4_{min}$ denote the values of $p_{min}$ used in the evaluation of $S^{h,2}$ and $S^{h,4}$ respectively. For the present results this would give $2 \times (4 + 6) + 2 = 22$ operations per gridpoint. Table 14 shows that the actual work per gridpoint is about twice as large: Indeed, for such a one dimensional problem, the asymptotic behaviour would clearly show only on much finer grids; finer in fact than can be calculated with the usual computer precision.

**7. Higher Dimensions.** In section 2 for simplicity a one dimensional problem was addressed. Below the generalization of our approach to higher dimensions is briefly discussed. For simplicity we assume $\bar{d} = d$. Let $x$ and $y$ denote vectors $x = (x^1, x^2, ..., x^d)$ and $y = (y^1, y^2, ..., y^d)$. In the same way the subscript $j$ is now a vector $j = (j^1, j^2, .., j^d)$ and $y_j$ stands for the vector $y_j = (y^1_{j^1}, y^2_{j^2}, ...y^d_{j^d})$. Let $\mathbf{e}^\kappa$ denote the $\kappa^{th}$ unit vector, e.g. $e^1 = (1, 0, 0...., 0)$, $e^2 = (0, 1, 0...., 0)$, etc. for $1 \le \kappa \le d$. The $d$ dimensional domain is subdivided in integration subspaces $V_j = \{y \in I\!R^d | y^\kappa_{j^\kappa} \le y^\kappa \le y^\kappa_{j^\kappa+1}, 1 \le \kappa \le d\}$. On each of these integration subspaces $\tilde{u}_j(y)$ denotes an interpolation polynomial of order $s$ (degree $s-1$) approximating $u^h$. The contribution of this subspace to the discrete integral transform $G^h u^h$ is:

$$G^h_j u^h(x) = \int_{V_j} G(x, y)\tilde{u}(y)dy^1, ...., dy^d. \tag{52}$$

$G^l(x, y)$, with $l = (l^1, l^2, ...., l^d)$, again stands for a family of kernels defined by

$$G^0(x, y) = G(x, y)$$

$$G^l(x, y) = \int_{x^\kappa}^{y^\kappa} G^{l-e^\kappa}(x, y + (\eta - y^\kappa)\mathbf{e}^\kappa)d\eta, \tag{53}$$

for any $\kappa$ such that $l^\kappa > 0$. Integrating (52) by parts $s$ times with respect to each dimension yields:

$$G^h_j u^h(x) = \sum_{\nu^1,..,\nu^d=0}^{1} \quad \sum_{l^1,..,l^d=1}^{s} (-1)^{|l|+|\nu|} G^l(x, y_{j+\nu})\tilde{u}^{(l-1)}(y_{j+\nu}), \tag{54}$$

where $|l| = \sum_{\kappa=1}^d l^\kappa$ and $|\nu| = \sum_{\kappa=1}^d \nu^\kappa$. The discrete integral transform itself is obtained by summing up over all integration intervals:

$$G^h u^h(x) = \sum_j \sum_{\nu^1,..,\nu^d=0}^{1} \quad \sum_{l^1,..,l^d=1}^{s} (-1)^{|l|+|\nu|} G^l(x, y_{j+\nu})\tilde{u}^{(l-1)}(y_{j+\nu}). \tag{55}$$

TABLE 15

*Estimated Work per gridpoint W needed for the evaluation of all $(s/2)^d$ (s even) discrete transforms $S^{h,l}(x)$, given that $m = 0$ and $p^\kappa = l^\kappa + 2$ are used on the finest grid, and the interpolations and anterpolations, are done in the optimal way: one direction at a time, interpolation with highest order first, and anterpolation with lowest order first.*

| | $s$ | | | | | |
|---|---|---|---|---|---|---|
| $d$ | 2 | 4 | 6 | 8 | 10 | 12 |
| 1 | 8 | 20 | 36 | 56 | 80 | 108 |
| 2 | 8 | 39 | 103 | 211 | 373 | 601 |
| 3 | 8 | 75 | 293 | 793 | 1743 | 3348 |

As was done in §2 for $d = 1$, for a given $d$, equation (55) can be written as the sum of a set of boundary terms, and a series of discrete transforms. In general there will be $s^d$ of these transforms. However, depending on the specifics of the relative positions of the integration grid and the datagrid many cancellations may occur. For example for $s$ even and datagrid and integration grid uniform and coinciding (as is natural for $s$ even), all discrete transforms for which one of the $l^\kappa$ is odd cancel, except for contributions from the boundary. Similarly, if $s$ is odd and the datagrid points are integration-grid midpoints, as is natural for $s$ odd, all terms for which one of the $l^\kappa$ is even cancel. In these situations there remain only $(s/2)^d$ discrete transforms to be evaluated for $s$ even, and $((s+1)/2)^d$ for odd $s$.

A first estimate of the work per gridpoint to be invested for the evaluation of all discrete transforms can be obtained as follows. The anterpolation from and interpolation to the finest grid can be done one dimension at a time. If the order of tranfer $p$ is the same for all directions, the interpolation and anterpolation each require $(1-2^{-d})p$ operations per fine grid point. Assuming that for each discrete transform $S^{h,l}(x)$ we can use $m = 0$ and some $p = p^l_{min}$ on the finest grid, one obtains that for even $s$, $W \approx 2^{1-d}\bar{p}_{min}s^d$ operations per fine grid point are needed to evaluate all discrete transforms. $\bar{p}_{min}$ denotes the largest of all $p^l_{min}$. Since for $s$ even $\bar{p}_{min} = s + 2$ we obtain $W \approx 2^{1-d}(s+2)s^d$. In the same way for $s$ odd one obtains $W \approx 2^{1-d}(s+1)^{d+1}$. Hence, $W \approx 2^{1-d}s^{d+1}$ for sufficiently large $s$. However, the work needed is actually smaller. If the anterpolation and interpolation are indeed done one dimension at a time, different orders of transfer and softening can be used in the different dimensions, e.g. $p^\kappa$ for the order of transfer with respect to the $\kappa^{th}$ dimension. The optimal way then is to interpolate first with respect to the dimension for which $p^\kappa$ is largest, and last with respect to the dimension for which $p^\kappa$ is the lowest, and vice versa, to anterpolate with respect to the dimensions in order of ascending $p^\kappa$, i.e. treating the lowest order first. Organized in this way the fast evaluation of a discrete transform $S^{h,l}(x,y)$ with $l = (l^1, l^2, ..., l^d)$ can use $p^\kappa = l^\kappa + 2$ on the finest grid if $s$ is even. The estimated work per gridpoint needed to obtain all discrete transforms can be computed numerically, e.g. see Table 15 where this optimal work is given as a function of $s$ and $d$ for $s$ even.

To get the discrete integral transform itself we have to add the boundary terms. In $d$ dimensions these terms in principle are multisummations over domains of dimension $d-1$. For $d = 1$ their evaluation has little consequences for the work per gridpoint, but for general $d$ these evaluations should also be done using a multilevel approach. However, in many practical cases this can be avoided by adding external points with $u(y) = 0$ for $y \notin \Omega$. In that case (11) near the boundary may no longer hold and locally a larger $m$ and $p$ may be needed. However, the extra work involved will be small, and the algorithm in principle enables evaluation using a number of operations

TABLE 16
*Work per gridpoint in multilevel solver of equivalent differential problem* $W = 6sd$

| | $s$ | | | | | |
|---|---|---|---|---|---|---|
| $d$ | 2 | 4 | 6 | 8 | 10 | 12 |
| 1 | 12 | 24 | 36 | 48 | 60 | 72 |
| 2 | 24 | 48 | 72 | 96 | 120 | 144 |
| 3 | 36 | 72 | 108 | 144 | 180 | 216 |

per fine grid point that is only a little larger than the work needed for the evaluation of all discrete transforms $S^{h,l}(x)$; see Table 15.

**8. Comment on the harmonic kernels.** For some special kernels, the integral transform (1) is equivalent to a differential equation with special boundary conditions. The most common example is the harmonic kernel, given by:

$$G(x,y) = \begin{cases} |y-x|^{2-d} & \text{for } d \neq 2 \\ \log|y-x| & \text{d=2,} \end{cases} \tag{56}$$

where $d$ is the dimension and

$$|y-x| = \left[\sum_{\kappa=1}^{d}(x^\kappa - y^\kappa)^2\right]^{1/2}.$$

For this kernel, the evaluation of (1) is equivalent to solving the Poisson equation $\Delta U = C_d U$ with so-called "absorbing" boundary conditions. A multigrid solver of this problem, discretized to $s$ order accuracy, is estimated to cost roughly $W = 6sd$ operations per gridpoint; see Table 16. From comparing this table with Table 15 it appears that for small $s$ and $d$ the evaluation of the integral transform requires less work per gridpoint than solving the equivalent differential problem. However, for larger $s$ and $d$, solving the problem in its differential form should generally be preferred.

**9. Conclusion.** Initiated by the need for local grid refinement techniques in actual applications, a new algorithm has been developed for the fast evaluation of integral transforms with asymptotically smooth kernels. This new algorithm does not depend on the uniformity of the grid for creating a suitably smooth discrete kernel. Rather, it only relies on the asymptotic smoothness of the *continuum* kernel as it appears in the integral transform. Thereby it facilitates local grid refinements. Also, the evaluations will generally be faster; for a $d$ dimensional problem only $O(s^{d+1})$ operations per gridpoint are needed, if $s$ is the order of discretization, and $d$ the dimension of the problem. This is illustrated by the results obtained for a one dimensional model problem with logarithmic kernel.

**10. Acknowledgement.** The authors wish to thank Dr. V. Mikulinsky, for his research leading to the conclusion that a new algorithm was needed, and prof. H. Lee of Murray State University, Murray, Kentucky, U.S.A., for his contribution in the initial phase of the present research.

## REFERENCES


[1] A.W. Appel, *An efficient program for many-body simulation*, SIAM J. Sci. Stat. Comput., 6, (1985), pp. 85–103.

[2] B. Alpert, G. Beylkin, R. Coifman, and V. Rokhlin, *Wavelet-like bases for the fast solution of second-kind integral equations*, SIAM J. Sci. Comput., 14,1 (1993), pp. 159–184.

[3] J. Barnes and P. Hut, *A hierarchical O(n ln n) force calculation algorithm*, Nature, 324 (1986), pp. 446–449.

[4] D. Bai and A. Brandt, *Local mesh refinement multilevel techniques*, SIAM J. Sci. Stat. Comput., 8, 2 (1986), pp. 109–134.

[5] G. Beylkin, R. Coifman, and V. Rokhlin, *Fast wavelet transforms and numerical algorithms*, Comm. Pure Appl. Math., 44 (1991), pp. 141–183.

[6] A. Brandt, and A. A. Lubrecht, *Multilevel matrix multiplication and the fast solution of integral equations*, J. Comput. Phys., 90, 2, (1990), pp 348–370.

[7] A. Brandt, *Multilevel computation of integral transforms and particle interactions with oscillatory kernels*, Computer Physics Communications, 65 (1991), pp. 24–38.

[8] A. Brandt, *Guide to Multigrid develpment*, In: Multigrid Methods (Proc. Köln-Porz, 1981), edited by W. Hackbusch and U. Trottenberg, Lecture Notes in Math., 960, Springer Verlag, New York, 1982, pp. 220–312, also included in *GMD Studien No. 85, Gesellschaft für Mathematik und Datenverarbeitung MBH, Bonn, Germany* (1984).

[9] A. Brandt, *Multi-level adaptive solutions to boundary value problems*, Math. Comp., 31 (1977), pp. 333–390.

[10] J. Carrier, L. Greengard, and V. Rokhlin, *A fast adaptive multipole algorithm for particle simulations*, SIAM J. Sci. Stat. Comput., 9 (1988), pp. 669–686.

[11] G. Dahlquist and A. Björck, *Numerical Methods*, Prentice Hall, Englewood Cliffs, NJ, 1974.

[12] L. Greengard and V. Rokhlin, *A fast algorithm for particle simulations*, J. Comput. Phys., 73 (1987), pp. 325–348.

[13] L. Greengard, *Fast algorithms for classical physics*, Science, 265 (1994), pp. 909–914.

[14] Z. Kopal, *Numerical Analysis*, Chapman & Hall, London, 1961.

[15] Z.P. Nowak and W. Hackbusch, *On the complexity of the panel Method*, International conference on Modern Problems in Numerical Analysis, Moscow, September 1986.

[16] V. Rokhlin, *Rapid solution of integral equations of classical potential theory*, J. of Comp. Phys., 60 (1983), 187–207.

[17] L. Reichel, *A fast method for solving certain integral equations of the first kind with application to conformal mapping*, J. Comput. and Appl. Math., 14 (1986), pp. 125–142.

[18] C.H. Venner and A.A. Lubrecht, *Multilevel solvers for integral and integro-differential equations in contact mechanics and lubrication*, in: Multigrid Methods IV: Proc. 4rd European Multigrid Conference, Amsterdam 1993, Ed. P. Hemker and P. Wesseling, Birkhauser Verlag, Germany, 1994.

# Appendix A

Let $\{z_j\}$ be the points of a uniform datagrid with meshsize $h$, and $\{y_j\}$ the points of a uniform integration grid. Let $\tilde{u}_j^h$ denote the interpolation polynomial of degree $s-1$ in the interval $[y_j, y_{j+1}]$, where $y_j = z_j$ if $s$ is even, and $y_j = (z_j + z_{j-1})/2$ if $s$ odd. The interpolation is done from $s$ datagrid points $z_{j+k}$ with $s_1 \le k \le s_2$, where $s_1 = -s/2+1$ and $s_2 = s/2$ for $s$ even, and $s_1 = -s_2 = (1-s)/2$ for odd $s$. The objective is to show that in the cases where it doesn't vanish, i.e. for $s-l$ even,

$$|U_j^{h,l}| = |\tilde{u}_{j-1}^{h,(l-1)}(y_j) - \tilde{u}_j^{h,(l-1)}(y_j)| = 2(\gamma_2 h)^{s-l+1}|u^{(s)}(y_j)| + O(h^{s-l+2}), \tag{57}$$

with $\gamma_2 \approx 0.5$.
Below this is shown in three steps. First we prove

$$\tilde{u}_j^{h,(l-1)}(y_j) = u^{(l-1)}(y_j) + C_{s,l-1}h^{s-l+1}u^{(s)}(y_j) + O(h^{s-l+2}) \tag{58}$$

and

$$\tilde{u}_{j-1}^{h,(l-1)}(y_j) = u^{(l-1)}(y_j) - (-1)^{s-l}\, C_{s,l-1}h^{s-l+1}u^{(s)}(y_j) + O(h^{s-l+2}). \tag{59}$$

Finally we show that for practical values of $s$ and $l$

$$|C_{s,l-1}|^{\frac{1}{s-l+1}} \lessapprox 0.5. \tag{60}$$

LEMMA 1.

$$\tilde{u}_j^{(l-1)}(y_j) = u^{h,(l-1)}(y_j) + C_{s,l-1}h^{s-l+1}u^{(s)}(y_j) + O(h^{s-l+2}) \tag{61}$$

*Proof.*
From standard numerical analysis, e.g. [11, 14], it is well known that the interpolation error is given by:

$$\tilde{u}_j^h(y) - u(y) = \frac{h^s}{s!}u^{(s)}(\xi)\prod_{k=s_1}^{s_2}(\tilde{y}-k) = h^s\frac{u^{(s)}(\xi)}{s!}P(\tilde{y}) \tag{62}$$

where $\xi$ is a point in the interval $[z_{j+s_1}, z_{j+s_2}]$ and $\tilde{y}$ is a local normalized coordinate: $\tilde{y} = (y - z_j)/h$. From (62) it follows that

$$\tilde{u}_j^{h,(l-1)}(y) - u^{(l-1)}(y) = \frac{h^s}{s!}\frac{d^{l-1}}{dy^{l-1}}\left[u^{(s)}(\xi)P(\tilde{y})\right]$$

$$= \sum_{t=0}^{l-1}\frac{(l-1-t)!}{(s+l-1-t)!}\begin{pmatrix} l-1 \\ t \end{pmatrix}h^{s-t}P^{(t)}(\tilde{y})u^{(s+l-1-t)}(\xi_t) \tag{63}$$

where $\xi_t \in [z_{j+s_1}, z_{j+s_2}]$, see [14]. Therefore $|\xi_t - y| = O(h)$ and

$$\tilde{u}_j^{h,(l-1)}(y) - u^{(l-1)}(y) = \frac{h}{s!}^{s-l+1} P^{(l-1)}(\tilde{y}) u^{(s)}(y) + O(h^{s-l+2}). \tag{64}$$

Substitution of $y = y_j$ then proves (61) with

$$C_{s,l-1} = \frac{P^{(l-1)}(\tilde{y}_j)}{s!} \tag{65}$$

where $\tilde{y}_j = 0$ in the case of even $s$ and $\tilde{y}_j = -\frac{1}{2}$ for odd $s$.

LEMMA 2.

$$\tilde{u}_{j-1}^{h,(l-1)}(y_j) = u^{(l-1)}(y_j) - (-1)^{s-l} C_{s,l-1} h^{s-l+1} u^{(s)}(y_j) + O(h^{s-l+2}) \tag{66}$$

*Proof.*
From Lemma 1, by symmetry (even s) or antisymmetry (odd s) around $\tilde{y} = (z_{j+s_1} + z_{j+s_2})/2$.

Finally we have to show that for practical values of $s$ and $l$

$$|C_{s,l-1}|^{\frac{1}{s-l+1}} \lessapprox 0.5. \tag{67}$$

(In principle we had to show (67) only for those cases where $|U_j^{l,h}|$ does not vanish, i.e. for $s - l$ even.) Since $C_{s,l-1}$ can easily be computed, this is simply shown by a table: see Table 17. We remark that for $l \leq 2$ if $s$ is odd, and for $l \leq 3$ if $s$ is even, one can rigorously prove that $|C_{s,l-1}|^{\frac{1}{s-l+1}} \leq 0.5$. Furthermore, for any $s$, directly from the definition $|C_{s,l-1}|^{\frac{1}{s-l+1}} = 0.5$ for $l = s$. However, for other cases $|C_{s,l-1}|^{\frac{1}{s-l+1}}$ may somewhat exceed 0.5, as the table shows.

TABLE 17
$|C_{s,l-1}|^{\frac{1}{s-l+1}}$ *as a function of* $s$ *and* $l-1$.

| | $s$ | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| $l-1$ | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
| 0 | 0 | 0.40 | 0 | 0.41 | 0 | 0.42 | 0 | 0.43 | 0 |
| 1 | 0.50 | 0.20 | 0.44 | 0.26 | 0.44 | 0.30 | 0.45 | 0.32 | 0.45 |
| 2 | | 0.50 | 0.29 | 0.47 | 0.32 | 0.47 | 0.35 | 0.47 | 0.37 |
| 3 | | | 0.50 | 0.35 | 0.50 | 0.37 | 0.49 | 0.39 | 0.49 |
| 4 | | | | 0.50 | 0.41 | 0.53 | 0.41 | 0.52 | 0.42 |
| 5 | | | | | 0.50 | 0.46 | 0.55 | 0.45 | 0.54 |
| 6 | | | | | | 0.50 | 0.50 | 0.57 | 0.48 |
| 7 | | | | | | | 0.50 | 0.54 | 0.59 |
| 8 | | | | | | | | 0.50 | 0.58 |
| 9 | | | | | | | | | 0.50 |

# Appendix B

Below we derive the optimal $m$ and $p$ for the case of the logarithmic kernel $G = \ln|y-x|$ from the minimization the incremental work per gridpoint $p+4m$, under the constraint that the incremental evaluation error does not exceed a discretization error bound. In §2 the discretization error per unit length was given by:

$$|\tau^h| \leq (\gamma_1 h)^s ||u^{(s)}|| \; |G|, \tag{68}$$

with $h$ standing for the mesh size on the finest grid. The incremental evaluation error, i.e. the error introduced by transferring evaluation from grid $H/2$ to grid $H$, per unit length is given by:

$$|e^H| = \gamma_2 (\gamma_2 h)^{s-l} |G_H^l - I_H^{H/2} G_H^l| \; ||u^{(s)}||, \tag{69}$$

where $G_H^l(x,y) - I_H^{H/2} G_H^l(x,y)$ stands for the order-$p$ interpolation error made with the interpolation of $G_H^l$ from grid $H$ to a point $x$ or a point $y$, and $|G_H^l - I_H^{H/2} G_H^l|$ is the average of its absolute value over the integration points $y$ of the $H/2$ grid, for any point $x$. Generally this error is bounded by:

$$|G_H^l - I_H^{H/2} G_H^l| < (\gamma_3 H)^p |G_H^{l(p)}|, \tag{70}$$

where $|G_H^{l(p)}|$ is the average of the absolute value of the $p^{th}$ derivative of $G_H^l$. Generally, by (68) and (69), $|e^H| \leq |\tau^h|$ requires

$$(\gamma_2)^{s-l+1} h^{-l} |G_H^l - I_H^{H/2} G_H^l| \leq (\gamma_1)^s \; |G|. \tag{71}$$

First we investigate whether $m = 0$, i.e. no softening at all, can be achieved for sufficiently small $H$ and $h$. If $m = 0$ and $p = l$ then $G_H^{l(p)} = G^{l-p}$, and from (69) and (68) with $H = O(h)$ it is clear that $|\tau^h|$ and $|e^H|$ are both $O(h^s)|G|$. In particular, for $\gamma_1 \approx \gamma_2 \approx \gamma_3 \approx 0.5$ and $H = 2h$ one can then show that $|e^H| \approx 2^l |\tau^h|$. This implies that using grid $2h$ in the evaluation of the discrete transform $S^{h,l}$ already introduces an error of the same size as the error that would have been made if the entire integral transform were discretized on grid $2h$ to begin with. Obviously therefore $p = l$ must be rejected. Next consider any $p > l$. Due to the singularity in $|G^{l-p}(x,y)| \propto |y-x|^{l-p}$ at $y = x$, we have to distinguish two regions:

(i) $|y-x| < pH/2$: The singularity is in the interpolation interval, hence (70) is useless. However, due to the smallness of this region, its contribution to the integral will also be small. Indeed, $G^l = O(|y-x|^l \log|y-x|)$ plus a polynomial of degree $l$ in $(y-x)$. Consequently for $p > l$ and $|y-x| = O(h)$ we obtain $G^l - I_H^{H/2} G^l = O(h^l |\log(h)|)$. Since the size of the region is just $O(h)$, its entire contribution to the left-hand side of (71) is just $O(h|\log(h)|)$, while the right-hand side of (71) is $O(1)$.

(ii) $|y-x| \geq pH/2$: The singularity is outside the interpolation interval. In this case we can use (70), hence the requirement (71) will be satisfied if for every $x$ and $y$ in this region

$$(\gamma_2)^{s-l+1} h^{-l} (\gamma_3 H)^p |y-x|^{l-p} \leq (\gamma_1^s) \log|y-x|. \tag{72}$$

For $|y-x| = O(h)$ and $H = O(h)$ the left-hand side is $O(1)$ which, for sufficiently small $h$, is much smaller than $|\log(h)|$. For $|y-x| \gg h$, the left-hand side is $o(1)$ whereas the right-hand side is $O(1)$.
From (i) and (ii) it is anticipated that $m = 0$ and any $p \geq l+1$ can indeed be used on sufficiently fine grids.

Assuming $p \geq l+1$ to be satisfied, next consider $m > 0$, and an arbitrary $H > h$. We again distinguish two regions:

(i) $|y-x| \leq mH + pH/2$. In this region the evaluation error per unit length is bounded by (70), with $G_H^{l\,(p)}$ attaining its maximum value at $|y-x| = mH$. For the polynomial given by (14) this maximum is approximately

$$\tilde{G}_H^{l(p)}(mH) \approx f(p)(mH)^{l-p} \tag{73}$$

with

$$f(p) \approx (p-l)!\, 2^{p-l} \approx \left(\frac{2(p-l)}{e}\right)^{p-l}, \tag{74}$$

as can be seen from Tables 1 and 2. Substituting (70), (73) and (74) in (71), and taking into account that the size of the region is $O(mH)$ we obtain the requirement

$$(m/\gamma_3)^{l-p+1} f(p) \leq g \tag{75}$$

where

$$g = \frac{\gamma_1^s}{\gamma_3^{l+1}\gamma_2^{s-l+1}} \frac{h^l}{H^{l+1}} |G| \tag{76}$$

Using Euler-Lagrange optimization, assuming equality in (75), and making minor simplifications such as $\frac{p-l}{p-l-1} \approx 1$ one obtains that $p+4m$ is minimized when:

$$8\gamma_3 \epsilon^{1/\chi} = \chi \tag{77}$$

with

$$\chi = \frac{-(p-l-1)}{(p-l-1)+\ln(g)} \tag{78}$$

For $\gamma_3 = 1/2$, i.e. central interpolation operators, the solution of (77) is $\chi = 4.9$.
Summarizing we obtain that $p$ should be taken the maximum of the lowest non-negative integer satisfying:

$$p \geq -0.83\ln(g) + l + 1 \tag{79}$$

and $l+1$, and $m$ the first integer that satisfies:

$$m \geq 1.23(p-l-1) \tag{80}$$

(ii) $|y-x| \geq mH + pH/2$: Here, $G_H^l = G^l$ so again we need to satisfy (72). For $H = O(h)$ this is obtained by any $p \geq l+1$, as explained above. Generally for $p \geq l+1$, (72) will be satisfied for

$$|y-x| \geq \left(\frac{H^p}{h^l}\right)^{\frac{1}{p-l}}$$

.

For $|y-x| = O(1)$ and $H = O(h^{\frac{1}{2}})$ (note that no larger $H$ is needed in the algorithm) this is obtained by any $p > 2l$. For $|y-x| = O(H)$ one can see that (72) will hold except for a subregion of $\bar{m} = \left(\frac{H}{h}\right)^{\frac{l}{p-l}}$ points. With $H \leq O(h^{\frac{1}{2}})$ and $p$ sufficiently large (compared to $l$), in particular for the $p$ and $m$ as obtained from $(i)$ either $m+p/2 \geq \bar{m}$, or $\bar{m}-m-p/2$ is small, hence the contribution of this sub-region to the total integral can be shown to be small, at most requiring $p$ to be slightly increased beyond (79).

The approximations made above are only crude estimates. In fact, there is no need for very accurate approximations here, as practical limitations, such as $m$ and $p$ being integers, will prevent us from obtaining the exact optimum anyway. Besides, as explained in §5, the exact rates at which $p$ and $m$ increase can safely be *increased*, as most of the work is anyway spent at the finest levels (with $m = 0$ and any convenient $p \geq l+1$).