On the evaluation of layer potentials close to their sources ⋆

Johan Helsing * and Rikard Ojala

Numerical Analysis, Centre for Mathematical Sciences,
Lund University, Box 118, SE-221 00 LUND, Sweden

Abstract

When solving elliptic boundary value problems using integral equation methods one may need to evaluate potentials represented by a convolution of discretized layer density sources against a kernel. Standard quadrature accelerated with a fast hierarchical method for potential field evaluation gives accurate results far away from the sources. Close to the sources this is not so. Cancellation and nearly singular kernels may cause serious degradation. This paper presents a new scheme based on a mix of composite polynomial quadrature, layer density interpolation, kernel approximation, rational quadrature, high polynomial order corrected interpolation and differentiation, temporary panel mergers and splits, and a particular implementation of the GMRES solver. Criteria for which mix is fastest and most accurate in various situations are also supplied. The paper focuses on the solution of the Dirichlet problem for Laplace's equation in the plane. In a series of examples we demonstrate the efficiency of the new scheme for interior domains and domains exterior to up to 2000 close-to-touching contours. Densities are computed and potentials are evaluated, rapidly and accurate to almost machine precision, at points that lie arbitrarily close to the boundaries.

Key words: Numerical integration, potential theory, fast solvers, singular integrals, multiply connected domains, integral equations, Dirichlet–Neumann map
1991 MSC: 65R20, 77C05

⋆ This work was supported by the Swedish Research Science Council under contract 621-2004-3672.

* Corresponding author.
Email addresses: helsing@maths.lth.se (Johan Helsing), rikardo@maths.lth.se (Rikard Ojala).

1 Introduction

The accurate and fast numerical evaluation of layer potentials close to their sources is of importance when solving elliptic boundary value problems using integral equation methods. Assume that D is a domain in the plane with boundary L. Let z be a point in D, let τ be a point on L, and assume that the solution $U(z)$ to a given boundary value problem in D is represented as

$$U(z) = \int_L \mu(\tau) k(\tau, z)\, \mathrm{d}\sigma_\tau\,, \quad z \in D\,, \tag{1}$$

where $\mathrm{d}\sigma$ denotes an element of arc length along L and $\mu(\tau)$ is a density which can be obtained from an integral equation of Fredholm's second kind

$$\mu(z) + \int_L \mu(\tau) k(\tau, z)\, \mathrm{d}\sigma_\tau = f(z)\,, \quad z \in L\,. \tag{2}$$

Here $f(z)$ represents boundary conditions on L. If D should be simply connected and $k(\tau, z)$ is the normal derivative of the free-space Green's function for the Laplacian with respect to τ, then (1) is the double layer potential and (2) is the classic integral equation for the interior Dirichlet problem. Numerous similar, but more involved, recent examples exist with applications to electrostatics [15,25], elasticity [7,9,10,22], Stokes flow [4,14], and materials science at large [6,32].

The kernel $k(\tau, z)$ of (1,2) often contains a denominator, loosely speaking, of the type "$\tau - z$". When z is close to τ this could cause numerical difficulties. To see this, consider the evaluation of (1). If z is close to L, the kernel $k(\tau, z)$ may undergo rapid changes as τ passes by on L. This behavior cannot be resolved by standard quadrature. Furthermore, the accuracy in $k(\tau, z)$ could be destroyed due to cancellation and so could the accuracy of $U(z)$. Now consider the solution of (2). Things typically go well when z is close to τ in space and in arc length and when L is smooth. The singularity in $k(\tau, z)$ may, for example, vanish. But if z is close to τ in space while distant in arc length, such as when L consists of closely spaced disjoint contours or in some other way falls back on itself, or if L has corners, the problems are similar as for (1). Actually, the problem with z being close to τ could be harder when (2) is to be solved than when (1) is to be evaluated. Knowledge of $\mu(\tau)$ helps a lot. The process of evaluating (1) amounts merely to a matrix-vector multiplication. In (2) a linear system has to be solved. This problem is more sensitive with respect to details in the quadrature.

Finding highly accurate and fast numerical methods for the situations described in the previous paragraph is an active research topic. A simple method for the evaluation of (1) close to L is interpolation between boundary values and values which can be obtained accurately using standard quadrature [37].

More elaborate methods suggested for both (1) and (2) rely on interpolation of $\mu(\tau)$ and intense resolution of $k(\tau, z)$, possibly in combination with singularity subtraction techniques [1,19]. Such schemes can be made high order accurate and, if adaptive, moderately fast but do not overcome the cancellation problem. Combinations of singularity subtraction, smoothing, and various types of error correction, but without use of special quadrature points or overresolution, have also been suggested [3,26,27], as have schemes where variable transformations help to simplify the integrand [11,24]. Other interesting research directions involve generalized Gaussian quadrature [36] and interpolation of $\mu(\tau)$ only in combination with analytical integration [28,33] which is related to rational quadrature [9,12,29,35]. Here some implementations, involving the homotopy methods and various recurrence relations, could be costly and may require extended precision arithmetic for stability. A complicating circumstance, pointed out in Section 3 of [9], is the inherent conflict associated with solving (2) and evaluating (1) using the same set of discretization points – points τ which are optimal for (2) may not be good for certain z in (1).

There seem to be three major choices of basic quadrature techniques for (1,2): Fourier methods, the composite trapezoidal rule, and composite Gaussian quadrature. Fourier methods [5,7] are efficient for circles. The trapezoidal rule [1,3,4,6,14,15,17,20,25,26,32] is spectrally accurate for smooth and for Cauchy-singular integrands (alternate point rule) on closed contours. It allows for accurate differentiation via the fast Fourier transform. Drawbacks are that it is clumsy when it comes to adaptive mesh refinement and local modifications, and that the spectral accuracy may be lost on open boundaries. Composite n-point Gaussian quadrature [9,19,22] is about as accurate as the trapezoidal rule for $n = 16$ and smooth integrands on closed contours, it can be modified locally with greater ease, it lends itself better to adaptive mesh refinement, it works well for open boundaries and also for surfaces in higher dimensions. Disadvantages include lower order accuracy for Cauchy-singular kernels and for differentiation, unless global methods are used. In this paper, aiming at the treatment of difficult setups, we shall work with composite Gaussian quadrature.

Our paper differs from most previous work in that we, on one hand, are more specialized. For brevity, we chiefly discuss the double layer potential with application to the Dirichlet problem in the plane. On the other hand, we go further to adapt efficiently to a wide range of situations. We present quadrature formulas and implementations along with criteria for their use. Some are variants of standard ideas. Some are new and, maybe, daring, and have bearing on the solution to several elliptic problems with piece-wise constant coefficients arising in materials science. In passing we give a more efficient formulation for mode II of Mikhlin's method for the exterior problem [17,20], and a new twist to the GMRES method with Gram-Schmidt orthogonalization which for systems that stem from the discretization of Fredholm second kind integral

equations achieves stability properties on par with, or better than, those of more expensive implementations relying on Householder reflections [34]. The organization is as follows: Section 2 presents integral equations. Section 3 describes an elegant *globally compensated* quadrature for simple potential evaluation at arbitrary points z when L consists of closed contours. Section 4 defines three more general quadratures for the double layer potential, all based on Gauss-Legendre nodes. One of these, the *special* quadrature (34), is new and fundamental for high accuracy close to L. It contains interpolations of the kernel and the density of different orders. Section 5 deals with the efficient computation of quadrature weights. A central relation is (54), which expresses the special quadrature as an easily computed and numerically stable rational quadrature. Section 6 gives a criterion for which z are to be considered as lying close to L. The criterion contains a basic integral, called p_1, which also lies at the heart of the special quadrature. Its evaluation is discussed in Section 7. Section 8 is on GMRES and Section 9 discusses a competitive alternative to Nyström interpolation. Large-scale numerical examples are saved for Section 10.

2 Two boundary value problems

We shall solve two particular problems involving Laplace's equation in the plane: the Dirichlet problem on a simply connected interior domain and the Dirichlet problem on a multiply connected exterior domain. To simplify the transition between real and complex notation we shall make no distinction between points in the real plane $\mathbb{R}^2$ and points in the complex plane $\mathbb{C}$. Points in $\mathbb{C}$ will be denoted z and τ.

The interior problem reads: find $U(z)$ such that

$$\Delta U(z) = 0\,, \quad z \in D\,, \tag{3}$$

$$\lim_{D \ni z \to \tau} U(z) = f(\tau)\,, \quad \tau \in L\,, \tag{4}$$

where D is the interior domain, L is its boundary with positive orientation, and $f(z)$ is the prescribed boundary value at L. The classic Fredholm integral equation formulation for this problem can be derived using the double layer potential representation which in complex variables reads

$$U(z) = \frac{1}{2\pi} \int_L \mu(\tau) \Im \left\{ \frac{\mathrm{d}\tau}{\tau - z} \right\}\,, \quad z \in D\,, \tag{5}$$

where $\mu(z)$ is a real valued density. The integral equation itself is

$$\mu(z) + \frac{1}{\pi} \int_L \mu(\tau) \Im \left\{ \frac{\mathrm{d}\tau}{\tau - z} \right\} = 2f(z)\,, \quad z \in L\,. \tag{6}$$

Discretization of (6) with a Nyström scheme yields a linear system of equations

$$(I + K)\,\boldsymbol{\mu} = 2\mathbf{f}\,, \tag{7}$$

where I is the identity matrix and K is a discretization of the kernel in (6).

The exterior problem reads: find $U(z)$ such that

$$\Delta U(z) = 0\,, \quad z \in D\,, \tag{8}$$

$$\lim_{D \ni z \to \tau} U(z) = f(\tau)\,, \quad \tau \in L\,, \tag{9}$$

$$|U(z)| \leq \lambda\,, \quad z \in D\,, \tag{10}$$

where λ is a real number that bounds the solution, D is a domain exterior to M closed contours L_k, $k = 1, \ldots, M$, each of which has positive orientation, and L is the union of all L_k. For this problem we shall use the modified Mikhlin representation

$$U(z) = \frac{1}{2\pi} \int_L \mu(\tau) \Im \left\{ \frac{\mathrm{d}\tau}{\tau - z} \right\} + c_0 + \sum_{k=1}^{M} a_k \log|z - z_k|\,, \quad z \in D\,, \tag{11}$$

where z_k is a point placed inside L_k, $\mu(z)$ is a real valued density, and c_0 and the a_k are real constants determined by

$$\mu(z) - \frac{1}{\pi} \int_L \mu(\tau) \Im \left\{ \frac{\mathrm{d}\tau}{\tau - z} \right\} - 2c_0 - 2 \sum_{k=1}^{M} a_k \log|z - z_k| = -2f(z)\,, \tag{12}$$

$$\sum_{k=1}^{M} a_k = 0\,, \tag{13}$$

$$\frac{1}{2|L_k|} \int_{L_k} \mu(\tau)\,\mathrm{d}\sigma = 0\,, \quad k = 1, \ldots, M\,. \tag{14}$$

Discretization of (12,13,14) with a Nyström scheme yields a linear system of equations on block form

$$\begin{bmatrix} I - K & \tilde{B} \\ \tilde{C} & \tilde{D} \end{bmatrix} \begin{bmatrix} \boldsymbol{\mu} \\ \begin{bmatrix} c_0 \\ \mathbf{a} \end{bmatrix} \end{bmatrix} = \begin{bmatrix} -2\mathbf{f} \\ 0 \end{bmatrix}\,, \tag{15}$$

where K is the same type of matrix as in (7).

The system (15) has a condition number which, among other things, depends on the number M of contours and their shapes and relative distances. Assuming that the shapes and relative distances are not significantly changed as M increases, then the condition number grows linearly with M. Traditionally, the system is solved with left preconditioning [17], called Mode II in [20],

whose purpose is to alleviate this ill-conditioning due to large M. Here we shall introduce right preconditioning for the same purpose

$$\left(\begin{bmatrix} I & 0 \\ 0 & I \end{bmatrix} - \begin{bmatrix} K & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} I & \tilde{B} \\ \tilde{C} & \tilde{D} \end{bmatrix}^{-1} \right) \boldsymbol{\omega} = \begin{bmatrix} -2\mathbf{f} \\ 0 \end{bmatrix}, \tag{16}$$

The solution to (15) can the be recovered via

$$\begin{bmatrix} \boldsymbol{\mu} \\ \begin{bmatrix} c_0 \\ \mathbf{a} \end{bmatrix} \end{bmatrix} = \begin{bmatrix} I & \tilde{B} \\ \tilde{C} & \tilde{D} \end{bmatrix}^{-1} \boldsymbol{\omega}\,. \tag{17}$$

Ill-conditioning due to groups of boundaries lying close to each other is harder to cope with. A promising approach, developed in a different setting, can be found in [5].

An interesting feature with (16) is that the last $M+1$ entries of $\boldsymbol{\omega}$ can be solved for analytically. They are zero. Introducing $\tilde{\boldsymbol{\omega}}$ for the vector $\boldsymbol{\omega}$ with the last $M+1$ entries removed, we can rewrite (16) as

$$\left(I - K\left(I + \tilde{B}(\tilde{D} - \tilde{C}\tilde{B})^{-1}\tilde{C}\right)\right)\tilde{\boldsymbol{\omega}} = -2\mathbf{f}\,. \tag{18}$$

The equation (18) appears to be new. It is simpler than the corresponding equation obtained by traditional left preconditioning since it has fewer unknowns. It may also be more accurate since it focuses solely on enforcing the boundary conditions at L – not a mix of boundary conditions at L and somewhat arbitrarily modeled boundary conditions at infinity.

In Section 10 we shall solve (7) and (18) numerically and evaluate $U(z)$ of (5) and (11) for a few setups and a large number of points z. As for the Nyström scheme we shall use composite Gauss-Legendre quadrature as basic tool. In situations like those mentioned in the Introduction, however, we need an improved quadrature. Its construction will be the topic of the next four sections.

3 A globally compensated quadrature

We start out with a quadrature that is narrow in scope, but remarkably simple and also accurate and extremely robust under certain conditions. Assume that the situation (the regularity of the contour, the grading of the mesh, the regularity of the data etc.) is such that we have solved the interior problem (7) and obtained a discretized approximation of $\mu(\tau)$ to high accuracy (in the

sense that composite n-point polynomial interpolation based on the discrete values of this approximation interpolate $\mu(\tau)$ accurately) and that we now want to evaluate $U(z)$ from (5) for z close to L. Assume, further, that we can accurately convert the solution $\mu(\tau)$ into the boundary values $F^+(\tau)$ of a function $F(z)$, analytic in D, and such that

$$\Re\left\{F^+(\tau)\right\} \equiv \lim_{D\ni z\to\tau} \Re\left\{F(z)\right\} = f(\tau)\,, \quad \tau \in L\,. \tag{19}$$

Then

$$U(z) = \Re\left\{F(z)\right\} = \Re\left\{\frac{1}{2\pi i}\int_L \frac{F^+(\tau)\,\mathrm{d}\tau}{\tau - z}\right\}, \quad z \in D\,. \tag{20}$$

This trivial formula may not look like much of an improvement. The integrand in (20) will suffer from the same difficulties with kernel resolution and cancellation as the integrand in (5). But if we instead, following Ioakimidis, Papadakis, and Perdios [21], consider the identity

$$\frac{1}{2\pi i}\int_L \frac{(F^+(\tau) - F(z))\,\mathrm{d}\tau}{\tau - z} = 0\,, \quad z \in D\,, \tag{21}$$

we have a smooth integrand. Let t_k and w_k, $k = 1, \ldots, N$, be all the nodes and weights of a composite n-point Gaussian quadrature for functions on L integrated with respect to a parameter t. In particular, let the boundary be parameterized as $\tau(t)$ and let D_t^1 be an operator that differentiates with respect to t. Introduce $\tau_k = \tau(t_k)$, and $\tau_k' = (D_t^1\tau)(t_k)$. Then (21) can be written

$$\sum_{k=1}^{N} \frac{(F^+(\tau_k) - F(z))\tau_k' w_k}{\tau_k - z} = 0\,, \quad z \in D\,. \tag{22}$$

The identity (22) should hold to about the same accuracy as that with which $F^+(\tau)$ can be integrated on L with the chosen nodes and weights. Since $F(z)$ is the only unknown in (22), we can use (22) to solve accurately for $F(z)$ and insert this value into the first equality of (20). An explicit formula for the evaluation of $U(z)$ is then

$$U(z) = \Re\left\{\frac{\sum_{k=1}^{N} \frac{F^+(\tau_k)\tau_k' w_k}{\tau_k - z}}{\sum_{k=1}^{N} \frac{\tau_k' w_k}{\tau_k - z}}\right\}, \quad z \in D\,. \tag{23}$$

One could say that the denominator in this formula compensates for the error in the numerator. The error compensation includes quadrature error as well as cancellation error. As an illustration of the latter, consider the limit process $D \ni z \to \tau_j$. The numerator and the denominator of (23) will both be dominated by the jth term in the sums. The floating point representation of the small quantity $\tau_j - z$ will contain an increasingly large relative error, but the actual number $\mathrm{fl}(\tau_j - z)$ is identical in the numerator and in the denominator. Therefore $U(z) \to \Re\left\{F^+(\tau_j)\right\}$ as $z \to \tau_j$ and $U(z)$ can be accurate up to the

point where $\mathrm{fl}(\tau_j - z) = 0$ and the process breaks down. Apparently this is a breakdown of a benign sort. One can set $U(z) = \Re\{F^+(\tau_j)\} = f(\tau_j)$ when z and τ_j coincide in finite precision arithmetic.

It remains to accurately convert $\mu(\tau)$ into $F^+(\tau)$. We could take

$$F^+(z) = f(z) + i\Im\left\{\frac{1}{2\pi i}\int_L \frac{\mu(\tau)\,\mathrm{d}\tau}{\tau - z}\right\}, \quad z \in L\,. \tag{24}$$

Let $\mu(t) = \mu(\tau(t))$, $\mu_k = \mu(t_k)$, and $z = \tau_j$. Then

$$\frac{1}{2\pi i}\int_L \frac{\mu(\tau)\,\mathrm{d}\tau}{\tau - \tau_j} \approx \frac{\mu_j}{2} + \frac{1}{2\pi i}\sum_{\substack{k=1\\k\neq j}}^{N} \frac{(\mu_k - \mu_j)\tau_k' w_k}{\tau_k - \tau_j} + \frac{1}{2\pi i} w_j (D_t^1 \mu)(t_j)\,, \tag{25}$$

and we can find $F^+(\tau(t))$ at all nodes t_k. The operator D_t^1 can act analytically on $\tau(t)$ but has to be implemented numerically, for example via n-point polynomial interpolation, when acting on $\mu(t)$.

This *globally compensated* quadrature can also be applied to the exterior problem. Assume that the situation is such that we have solved (18) and, via (17), obtained a discretized approximation of $\mu(\tau)$ to high accuracy and that we now want to evaluate $U(z)$ from (11) for z close to L. Only the first term on the right hand side of (11) poses difficulties. Let us call it $U_1(z)$. Proceeding as above we can define

$$F^-(z) = -\frac{\mu(z)}{2} + \frac{1}{2\pi i}\int_L \frac{\mu(\tau)\,\mathrm{d}\tau}{\tau - z}\,, \quad z \in L\,, \tag{26}$$

and compute $U_1(z)$ from

$$U_1(z) = \Re\left\{\frac{\sum_{k=1}^N \frac{F^-(\tau_k)\tau_k' w_k}{\tau_k - z}}{-2\pi i + \sum_{k=1}^N \frac{\tau_k' w_k}{\tau_k - z}}\right\}, \quad z \in D\,. \tag{27}$$

The globally compensated quadrature, as described so far and within the composite Gaussian setting, may have a minor disadvantage in terms of economy of discretization points when compared with the interpolatory quadratures described in the next section. The reason is the term $(D_t^1\mu)(t_j)$ in (25), which, if computed via n-point polynomial interpolation, for simple problems may reduce the global order of the accuracy. The globally compensated quadrature can, however, be made competitive also in this respect with the use of local high polynomial order corrections, see the end of Section 9, or with the use of a more expensive global formula derived from (6)

$$(D_t^1\mu)(t_j) = 2(D_t^1 f)(t_j) - \frac{1}{\pi}\int_L \mu(\tau)\Im\left\{\frac{\tau_j'\,\mathrm{d}\tau}{(\tau - \tau_j)^2}\right\}, \tag{28}$$

which has a continuous kernel and retains the order of accuracy.

We remark that $F^+(z)$ of (24) could be cast in the same form as $F^-(z)$ of (26), but not vice versa. The advantage of the form (24) over the form (26) is that (24) contains cleaner data. The free term $f(z)$ in (24) is the given boundary value, while the free term $-\mu(z)/2$ in (26) is a computed solution, possibly polluted with error.

4 Interpolatory quadrature and approximation

Let $g(t)$ be a function of the real variable $t \in [-1, 1]$. Let $s(t)$ be a continuous differentiable injective map from $t \in [-1, 1]$ into the complex plane. Let $\mathbb{P}_n$ denote the set of polynomials of degree at most n. Let t_{nk} and w_{nk}, $k = 1, \ldots, n$, denote the nodes and weights associated with n-point Gauss-Legendre quadrature. Let $P_n[g(t)](s) \in \mathbb{P}_{n-1}$ denote the polynomial in $s(t)$ interpolating $g(t)$ at t_{nk}, $k = 1, \ldots, n$.

Often one wants to make an accurate numerical approximation to an integral of $g(t)$ over $t \in [-1, 1]$. Gauss-Legendre quadrature

$$\int_{-1}^{1} g(t)\,\mathrm{d}t \approx \sum_{k=1}^{n} g(t_{nk}) w_{nk} = \int_{-1}^{1} P_n[g(t)](t)\,\mathrm{d}t\,, \tag{29}$$

will be then be exact for any $g(t) \in \mathbb{P}_{2n-1}$.

Now we want to make accurate numerical approximations to integrals

$$I_i(z) = \int_{l_i} \mu(\tau) \Im \left\{ \frac{\mathrm{d}\tau}{\tau - z} \right\}, \tag{30}$$

where l_i is a segment, called panel, of some larger smooth contour L in the complex plane, arising in the context of composite quadrature. For simplicity the panel l_i is assumed to start at $\tau = -1$ and end at $\tau = 1$. The density $\mu(\tau)$ is a smooth real function of the complex variable τ on L. The point z could be anywhere in the plane. Actually, we could be more general and remove $\Im\{\cdot\}$ in (30) to get a complex cauchy-singular kernel. All quadratures that we are about to construct apply equally well in this case. The smoothness of the double layer kernel, for τ and z both on l_i, has no relevance for z away from L. Furthermore, with minor modifications one could let $\mu(\tau)$ be a complex function, possibly the limit on l_i of an analytic function or the product of a real function and a complex function such as the outward unit normal on l_i. One such generalization is tested in the final numerical example of Section 10.2 in connection with equation (71). But since this paper is chiefly

about the double layer potential, as for now we stick to real $\mu(\tau)$ and (30) as it stands.

Assume that there exists a parameterization $\tau(t)$ which traverses l_i for $t \in [-1, 1]$ and let $\mu(t) = \mu(\tau(t))$. Introducing

$$k(t, z) = \Im \left\{ \frac{\tau'(t)}{\tau(t) - z} \right\}, \tag{31}$$

and with t being the variable of integration we can approximate (30) as

$$I_i(z) \approx \sum_{k=1}^{n} \mu(t_{nk}) k(t_{nk}, z) w_{nk} = \int_{-1}^{1} P_n[\mu(t)k(t, z)](t) \, \mathrm{d}t \,. \tag{32}$$

This will be referred to as *straight-up* quadrature and is the preferred choice when $\mu(t)$ is a more rapidly changing function of t than is $k(t, z)$. This holds, for example, when z is far away from l_i, but also when $z \in L$ and z is on l_i or close to l_i in space as well as in arc length.

When z is moderately close to l_i, but distant in arc length if $z \in L$, one could expect $\mu(t)$ and $k(t, z)$ to be equally rapidly changing functions of t. Given $\mu(t_{nk})$ one can improve on (32) as follows: approximate $\mu(t)$ with a polynomial in $\mathbb{P}_{n-1}$, multiply with $k(t, z)$, use m-point Gauss-Legendre quadrature with $m \geq n$ for the integral. In other words, we keep the nodes t_{nk} but find new weights $\hat{w}_{nk}$, depending on l_i and z, so that

$$I_i(z) \approx \sum_{k=1}^{n} \mu(t_{nk}) \hat{w}_{nk} = \int_{-1}^{1} P_m[P_n[\mu(t)](t)k(t, z)](t) \, \mathrm{d}t \,. \tag{33}$$

This will be referred to as *extended* quadrature.

When z is very close to l_i, but distant in arc length if $z \in L$, one could expect $k(t, z)$ to be a much more rapidly changing function of t than $\mu(t)$. Then polynomial quadrature is not a good idea. Given $\mu(t_{nk})$ a better option is to find new weights $\tilde{w}_{nk}$, depending on l_i and z, so that

$$I_i(z) \approx \sum_{k=1}^{n} \mu(t_{nk}) \tilde{w}_{nk} = \int_{l_i} \Im \left\{ \frac{P_m[P_n[\mu(t)](t)](\tau) \, \mathrm{d}\tau}{\tau - z} \right\}. \tag{34}$$

This will be referred to as *special* quadrature and could be seen as a quadrature for (30) where $\mu(t)$ is approximated with a polynomial in t and where $k(t, z)$ also is approximated as to facilitate analytical evaluation for l_i with non-zero curvature. The higher the value of m, the better the integrand captures $P_n[\mu(t)](t)$. Should the interpolation $P_m[\cdot](\tau)$ not be included, the integral in (34) has to be evaluated numerically. This is costly.

Finding the weights $\tilde{w}_{nk}$ is straight-forward in theory. One simply requires that the equality in (34) holds for a set of $\mu(t)$ forming an independent basis for

$\mathbb{P}_{n-1}$, for example monomials or Legendre or Chebyshev polynomials. Different choices of basis functions involve different number of floating point operations (FLOPs) and different condition numbers of intermediary matrices. But also, given a basis, the order in which operations are carried out plays a major role for accuracy and for speed. We shall choose monomials, which seems to be the fastest route. We counterbalance the effect of ill-conditioning, used as a major argument against monomials in Section 2 of Ref. [35], by being careful about the order in which matrices act. In particular, we avoid the explicit construction of inverses of Vandermonde matrices and mixed Vandermonde-Cauchy matrices. We shall let Vandermonde matrices act as system matrices only in linear systems whose right hand sides have substantial components in directions corresponding to the first few left singular vectors. For such systems, linear solvers which are backward stable in practice produce solutions with very small residuals. This is most often sufficient for high accuracy in contexts involving interpolation of smooth functions determined by their values at Legendre nodes. See, further, Appendix A.

The density $\mu(t)$ in (33,34) is assumed known at the points t_{nk} on the panel l_i, interpolated by $P_n[\mu(t)](t)$, and evaluated at the points t_{mk}. Polynomial interpolation at t_{nk} cannot produce a better approximation to $\mu(t)$ than $P_n[\mu(t)](t)$. But if information on $\mu(t)$ from neighboring panels is incorporated in the quadrature rule for l_i, then $P_n[\mu(t)](t)$ at t_{mk} could be replaced with more accurate values. An extreme variant of this, when $\mu(t)$ is the solution to a Fredholm integral equation, is Nyström interpolation. This option, used by Atkinson [1] for the evaluation of (5), involves global information on $\mu(t)$ and has the theoretical advantage that it carries the accuracy of $\mu(t_{nk})$ over to $\mu(t_{mk})$. Unfortunately, Nyström interpolation is hard to apply in the process of solving integral equations. It could suffer from cancellation and in our tests it does not quite live up to expectations. Section 9 presents a related idea, which often works better in practice. We also remark that for the special case of $m \equiv n = 2$, the quadrature (34) resembles the interpolatory quadrature suggested by McKenney [28].

5 Computing weights and sums

This section compares different implementations of the quadratures (33,34) for $I_i(z)$. The comparisons contain crude complexity estimates. For simplicity we do not distinguish between real and complex FLOPs and we tacitly assume that n and m are some medium sized numbers differing with a factor of approximately two. The $O(\cdot)$ symbol indicates how the number of FLOPs scales with n or m or combinations thereof to leading order including a constant of proportionality. In the numerical examples of Section 10 we choose $n = 16$ and $m = 32$. We first turn our attention to some useful definitions.

Let $\mathbf{p} \in \mathbb{C}^m$ denote the column vector with entries given by

$$p_j = \int_{l_i} \frac{\tau^{j-1} \mathrm{d}\tau}{\tau - z}\,, \quad j = 1, \ldots, m\,. \tag{35}$$

Once p_1 has been computed, the other entries of $\mathbf{p}$ can be computed by a simple recurrence relation costing, on average, 3/2 FLOPs per entry: Let $\mathbf{c} \in \mathbb{R}^m$ denote the column vector with entries given by

$$c_j = \frac{1 - (-1)^j}{j}\,, \quad j = 1, \ldots, m\,. \tag{36}$$

Then

$$p_{j+1} = z p_j + c_j\,, \quad j = 1, \ldots, m-1\,. \tag{37}$$

Let $\tau_{nk} = \tau(t_{nk})$, $\tau_{mk} = \tau(t_{mk})$, $\tau'_{nk} = (D_t^1 \tau)(t_{nk})$, and $\tau'_{mk} = (D_t^1 \tau)(t_{mk})$. Let $V \in \mathbb{R}^{n \times n}$, $W \in \mathbb{R}^{m \times n}$, and $C \in \mathbb{C}^{m \times m}$ denote the Vandermonde matrices with entries given by

$$V_{kj} = t_{nk}^{j-1}\,, \quad k, j = 1, \ldots, n\,, \tag{38}$$

$$W_{kj} = t_{mk}^{j-1}\,, \quad k = 1, \ldots, m\,, \quad j = 1, \ldots, n\,, \tag{39}$$

$$C_{kj} = \tau_{mk}^{j-1}\,, \quad k, j = 1, \ldots, m\,. \tag{40}$$

Let $\mathbf{a} \in \mathbb{C}^n$ and $\hat{\mathbf{a}} \in \mathbb{C}^m$ denote the column vectors with entries given by

$$a_k = \frac{\tau'_{nk} w_{nk}}{\tau_{nk} - z} \quad k = 1, \ldots, n\,, \tag{41}$$

$$\hat{a}_k = \frac{\tau'_{mk} w_{mk}}{\tau_{mk} - z} \quad k = 1, \ldots, m\,. \tag{42}$$

Let $\boldsymbol{\mu}_i$, $\tilde{\mathbf{w}}$, and $\hat{\mathbf{w}}$ denote the column vectors in $\mathbb{R}^n$ with entries given by $\mu(t_{nk})$, $\tilde{w}_{nk}$, and $\hat{w}_{nk}$, $k = 1, \ldots, n$. The extended quadrature (33) can now be written

$$I_i(z) \approx \hat{\mathbf{w}}^{\mathrm{T}} \boldsymbol{\mu}_i = \Im \left\{ \hat{\mathbf{a}}^{\mathrm{T}} \right\} W V^{-1} \boldsymbol{\mu}_i\,, \tag{43}$$

where $\hat{\mathbf{w}}^{\mathrm{T}}$ is the transpose of $\hat{\mathbf{w}}$. The special quadrature (34) can be written

$$I_i(z) \approx \tilde{\mathbf{w}}^{\mathrm{T}} \boldsymbol{\mu}_i = \Im \left\{ \mathbf{p}^{\mathrm{T}} C^{-1} \right\} W V^{-1} \boldsymbol{\mu}_i\,. \tag{44}$$

We remark that $m \times m$ Vandermonde systems can be solved in $O(5m^2/2)$ FLOPs using methods that rely on Newton interpolation [13]. The well conditioned matrix WV^{-1} can be precomputed to full accuracy once and for all. For $n = 16$ and $m = 32$ its condition number is 1.63. When the matrix WV^{-1} acts on a vector to the right it performs polynomial interpolation. This

costs $O(2mn)$ FLOPs in a straight-forward implementation, but asymptotically faster algorithms certainly do exist [8]. We settle for precomputation in quadruple precision (even though it is not really needed, see Appendix A) followed by rounding to double precision and a simple speedup, applicable for even n and m based on the observation that only half of the entries of WV^{-1} are distinct. It is possible to factorize

$$WV^{-1} = \begin{bmatrix} I_{m/2} & I_{m/2} \\ J_{m/2} & -J_{m/2} \end{bmatrix} \begin{bmatrix} (WV^{-1})_1 & 0 \\ 0 & (WV^{-1})_2 \end{bmatrix} \begin{bmatrix} I_{n/2} & J_{n/2} \\ I_{n/2} & -J_{n/2} \end{bmatrix} . \tag{45}$$

Here $I_{m/2}$ and $I_{n/2}$ are $m/2 \times m/2$ and $n/2 \times n/2$ identity matrices and $J_{m/2}$ and $J_{n/2}$ are anti-diagonal identity matrices of the same dimensions. The $m/2 \times n/2$ matrices $(WV^{-1})_1$ and $(WV^{-1})_2$ contain the precomputed entries. Equation (45) leads to a cost of $mn + n$ FLOPs for application of the matrix WV^{-1} to a column vector to the right and $mn + m$ FLOPs for application to a row vector on the left.

We wish to stress at this point that even though we are to use the symbols C^{-1}, V^{-1}, and $C^{-\mathrm{T}}$ frequently, nowhere are the matrices C and V intended to be inverted numerically as to produce explicit inverse matrices. The matrix WV^{-1} is precomputed once and for all and applied to vectors according to the previous paragraph. The action of C^{-1} on a vector of function values $\hat{\boldsymbol{\mu}}$ is achieved via the Björk-Pereyra algorithm which should give monomial coefficients yielding accurate interpolation up to $m \approx 40$, see Appendix A and Section 5.2. The action of $C^{-\mathrm{T}}$ on a vector of integrated rational functions $\mathbf{p}$ is achieved via a mix of a reduced Björk-Pereyra algorithm and analytical methods within a certain approximation, see Section 5.3, and should give quadrature weights yielding accurate integration up to $m \approx 40$ provided $\mathbf{p}$ is accurate.

5.2 Several points z

Assume that we need accurate approximations to $I_i(z)$ for several points z close to l_i but not on L. Then we can use (43) for those z that are moderately close and (44) for those z that are very close. An interesting aspect of the rightmost expressions of (43) and (44) is that only the vectors $\hat{\mathbf{a}}$ and $\mathbf{p}$ contain information about z. We can therefore first precompute

$$\hat{\boldsymbol{\mu}} = (WV^{-1})\boldsymbol{\mu}_i \,, \tag{46}$$

and

$$\mathbf{y} = C^{-1}\hat{\boldsymbol{\mu}} \,. \tag{47}$$

Then only the vector $\hat{\mathbf{a}}$ or $\mathbf{p}$ needs to be computed for each z.

The precomputation of $\hat{\boldsymbol{\mu}}$ costs $O(mn)$ FLOPs and $\mathbf{y}$ costs $O(5m^2/2)$ FLOPs once $\hat{\boldsymbol{\mu}}$ is available. Including the computation of $\hat{\mathbf{a}}$ and $\mathbf{p}$, the sums

$$\hat{\mathbf{w}}^{\mathrm{T}}\boldsymbol{\mu}_i = \Im\left\{\hat{\mathbf{a}}^{\mathrm{T}}\right\}\hat{\boldsymbol{\mu}}\,, \tag{48}$$

and

$$\tilde{\mathbf{w}}^{\mathrm{T}}\boldsymbol{\mu}_i = \Im\left\{\mathbf{p}^{\mathrm{T}}\mathbf{y}\right\}\,, \tag{49}$$

both cost approximately $O(4m)$ FLOPs per z, once $\hat{\boldsymbol{\mu}}$ or $\mathbf{y}$ and p_1 are available. The vector of monomial coefficients $\mathbf{y}$ may be inaccurate, but the right hand side of (49) will be accurate if the integrated rational functions in $\mathbf{p}$ are accurate since the coefficients in $\mathbf{y}$ represent $\boldsymbol{\mu}_i$ well for interpolation purposes and since integration is a smoothing process.

5.3 Several densities $\boldsymbol{\mu}_i$ and points z

When solving a discretized integral equation iteratively, new accurate approximations to $I_i(z)$ are needed at each iteration. That is, one may need $I_i(z)$ for several densities $\boldsymbol{\mu}_i$ as well as for several points $z \in L$ close to l_i in space but distant in arc length. An efficient implementation of (43) and (44) could then be to precompute the weights

$$\hat{\mathbf{w}}^{\mathrm{T}} = \Im\left\{\hat{\mathbf{a}}^{\mathrm{T}}\right\}(WV^{-1})\,, \tag{50}$$

or

$$\tilde{\mathbf{w}}^{\mathrm{T}} = \Im\left\{\left(C^{-\mathrm{T}}\mathbf{p}\right)^{\mathrm{T}}\right\}(WV^{-1})\,, \tag{51}$$

for each z and store them. Forming $\hat{\mathbf{w}}^{\mathrm{T}}\boldsymbol{\mu}_i$ or $\tilde{\mathbf{w}}^{\mathrm{T}}\boldsymbol{\mu}_i$ only costs $O(2n)$ FLOPs per z and iteration. The cost of computing $\hat{\mathbf{w}}$, given $\hat{\mathbf{a}}$, is $O(mn)$ FLOPs per z. The cost of computing $\tilde{\mathbf{w}}$, given $\mathbf{p}$, seems at first glance to be $O(mn + 5m^2/2)$ FLOPs per z.

Interestingly, the cost of computing $C^{-\mathrm{T}}\mathbf{p}$ in (51) can be reduced from $O(5m^2/2)$ per z in the estimate above to $O(7m)$, or less, once p_1 and some other quantities are available. To see this, assume that the two vectors $\mathbf{u}, \mathbf{v} \in \mathbb{C}^m$ depending solely on l_i,

$$\mathbf{u} = C^{-\mathrm{T}}\mathbf{c}\,, \quad \mathbf{v} = C^{-\mathrm{T}}\mathbf{e}_m\,, \tag{52}$$

are precomputed. Here $\mathbf{e}_m \in \mathbb{C}^m$ denotes a vector where all entries are zero except for the last entry which can be an arbitrary non-zero number, for example unity or $\tau_{mm} - \tau_{m1}$, and $\mathbf{c}$ is as in (36). Let $\tilde{\mathbf{a}}, \tilde{\mathbf{d}} \in \mathbb{C}^m$ be given by

$$\tilde{a}_k = \frac{u_k}{\tau_{mk} - z}\,, \quad \tilde{d}_k = \frac{v_k}{\tau_{mk} - z}\,, \quad k = 1, \ldots, m\,. \tag{53}$$

Using rational quadrature based on partial fraction expansions [9,35] as a starting point one can, with help of the Sherman-Morrison formula, show the

remarkably simple relation

$$C^{-\mathrm{T}}\mathbf{p} = \tilde{\mathbf{a}} - \alpha\tilde{\mathbf{d}}\,, \tag{54}$$

where

$$\alpha = \frac{\mathtt{sum}(\tilde{\mathbf{a}}) - p_1}{\mathtt{sum}(\tilde{\mathbf{d}})}\,. \tag{55}$$

Here $\mathtt{sum}(\tilde{\mathbf{a}})$ means summation of the entries of $\tilde{\mathbf{a}}$. The construction of $C^{-\mathrm{T}}\mathbf{p}$ via (53,54,55) only costs $O(7m)$ FLOPs per z. In addition, it has excellent numerical stability. See Section 6.

The precomputation of $\tilde{\mathbf{a}}$ and $\tilde{\mathbf{d}}$ of (53) can be done cheaply. The cost of forming $\mathbf{v}$ in (52) is $O(3m^2/2)$ FLOPs. If the segment l_i can be well approximated with a low degree polynomial in t, which often is the case for discretized and well resolved integral equations, then $\tilde{\mathbf{a}} \approx \hat{\mathbf{a}}$ holds to high accuracy, omitting the need to precompute $\tilde{\mathbf{a}}$ via (52,53) altogether. Furthermore, in our process of determining whether extended quadrature $\hat{\mathbf{w}}^{\mathrm{T}}\boldsymbol{\mu}_i$ is accurate or if special quadrature $\tilde{\mathbf{w}}^{\mathrm{T}}\boldsymbol{\mu}_i$ should be activated, the quantities $\hat{\mathbf{a}}$ and $\mathtt{sum}(\hat{\mathbf{a}}) - p_1$ need to be computed anyhow, see Section 6. The construction of $C^{-\mathrm{T}}\mathbf{p}$, from this point of view, only costs $4m$ FLOPs per z and the cost of computing $\tilde{\mathbf{w}}$ is $O(mn)$ FLOPs per z.

Should it happen that a point z lies very close to a point, τ_{mk} say, the k:th entry of vectors $\tilde{\mathbf{a}}$ and $\tilde{\mathbf{d}}$ may be inaccurate due to cancellation, while α still is good, and we need to modify the k:th entry of (54) according to the following

$$\left(C^{-\mathrm{T}}\mathbf{p}\right)_k = -\mathtt{sum}(\tilde{\mathbf{a}}_{0k}) + \mathtt{sum}(\tilde{\mathbf{d}}_{0k})\alpha + p_1\,, \tag{56}$$

where $\tilde{\mathbf{a}}_{0k}$ and $\tilde{\mathbf{d}}_{0k}$ are the $\tilde{\mathbf{a}}$ and $\tilde{\mathbf{d}}$ vectors with the k:th entries set to zero. This situation seldom occurs, so the cost of doing the modification is negligible. But the cost of checking if z is close to τ_{mk} is $O(6m)$ FLOPs. Our criterion for when a point z lies very close to a point τ_{mk} is that the distance should be less than $1/1000$ of the panel length.

5.4 Transformations

When discretizing boundary integral equations using composite quadrature one often uses a global parameterization $\tau(t)$ where t is a global parameter. The canonical Gauss-Legendre nodes t_{nk} and t_{mk} therefore have to be translated and scaled so that they fit each actual panel l_i. An actual panel l_i will then, in general, not start at $\tau(-1) = -1$ and end at $\tau(1) = 1$, as assumed up until now. Since the kernel of (30) is invariant under rotations, scalings, and translations of the plane and since $\tau'(t)\,\mathrm{d}t$ is invariant under translations and scalings of t, it is simple to modify our three quadratures to account for this.

One can use actual τ_{nk}, z, and τ'_{nk} in the straight-up quadrature if the weights w_{nk} are multiplied with the ratio between the length of the actual range of the parameter t on l_i and the length of the canonical range $[-1, 1]$. The extended quadrature is modified in a similar fashion. Actual τ_{mk}, z, and τ'_{mk} can be used if the entries of $\hat{\mathbf{a}}$ are scaled. In the special quadrature and when computing $\mathbf{p}$ and the action of C^{-1} and $C^{-\mathrm{T}}$, we do indeed transform the plane for each panel so that l_i starts at $\tau = -1$ and ends at $\tau = 1$. The transformation $h(z)$ from the actual plane to the transformed plane is

$$h(z) = \frac{2z - (\tau_{\mathrm{e}} + \tau_{\mathrm{s}})}{(\tau_{\mathrm{e}} - \tau_{\mathrm{s}})}\,, \tag{57}$$

where τ_{s} and τ_{e} denote the points where panel l_i starts and ends in the actual plane. The transformation $h(z)$ prevents the condition number of C from becoming unnecessarily high. The vectors $\tilde{\mathbf{a}}$ and $\tilde{\mathbf{d}}$ of (53) have the same invariance properties as the kernel of (30). Since the numerators are computed in the transformed plane the same must hold for the denominators. But in practice we not have to worry about transformations here: $\tilde{\mathbf{a}}$ is approximated with $\hat{\mathbf{a}}$ and $\tilde{\mathbf{d}}$ is not unique anyhow – it can only be determined up to multiplication with an arbitrary complex number and we can use the actual τ_{mk} and z in its denominator.

5.5 Panel mergers and splits

Should it happen that the point z lies very close to either of the tips of the panel l_i, the computation of p_1 will suffer from cancellation. Although a rare event, this may destroy the accuracy in the special quadrature for $I_i(z)$. Actually, the cancellation problem will occur for two panels – if z is close to a tip of one panel it is also close to a tip of a neighboring panel. The following procedure takes care of the situation for both involved panels in the context of (47,49) assuming that $\mu(\tau)$ is smooth: Merge the panels. Divide the merged panel into three temporary panels of equal length and distribute temporary points τ_{mk}. Interpolate the two different $\boldsymbol{\mu}_i$ corresponding to the original panels at the points τ_{mk} on each temporary panel in the style of (46,47). This gives three vectors $\mathbf{y}$. Compute the three vectors $\mathbf{p}$ corresponding to the temporary panels. Compute (49) for the two original panels as a sum of three expressions $\Im\left\{\mathbf{p}^{\mathrm{T}}\mathbf{y}\right\}$. Our criterion for when a point z lies very close to a panel tip is that the distance should be less than 1/1000 of the panel length.

We remark that McKenney [28] also uses merger of neighboring panels when evaluating Cauchy integrals at points z close to panels tips.

6 When is a point z close to a panel l_i?

Assume that L is centered at the origin. Let *tol* be a tolerance indicative of the relative accuracy we seek in our overall computation. Our criterion for when straight-up quadrature is sufficiently accurate for $I_i(z)$ of (30) is

$$|\mathtt{sum}(\mathbf{a}) - p_1| < \max\left(tol, \frac{|L|\epsilon_{\text{mach}}}{|l_i|}\right), \tag{58}$$

where $|l_i|$ is the length of the panel l_i, $|L|$ is the size of the geometry, for example the diameter of the smallest circle enclosing L, and ϵ_{mach} is machine epsilon. Our criterion for when extended quadrature is sufficiently accurate is (58) with $\mathbf{a}$ replaced by $\hat{\mathbf{a}}$.

The criterion (58) appears to be efficient in tests. Since p_1 is of order unity, the left hand side of (58) can be interpreted as the relative error produced by straight-up quadrature for a modification of (30) with complex kernel and unit $\boldsymbol{\mu}$. Admittedly, a constant density is simpler to integrate than a general density, but the complex kernel is more difficult to integrate than its imaginary part. Furthermore, if $\mu(t)$ is expanded in Legendre polynomials, low order polynomials should have larger coefficients than high order polynomials and accurate integration of the former is of greater importance. In integrals such as (5), contributions $I_i(z)$ from a large number of panels are summed up. If z lies very close to a panel l_i, the contribution $I_i(z)$ from that particular panel could be as large as that from all other panels together. Therefore it makes sense to relate the accuracy in the overall computation to the accuracy of the contribution from a panel that is particularly close to a particular z.

It will be expensive to check (58) for all combinations of points z and panels l_i. We speed up the process by performing a heuristic screening of the point-panel pairs, based on the experimental observation that straight-up quadrature at a point z with respect to the panel l_i is accurate enough when z lies more than a distance $|l_i|$ away from the mid-point of l_i. We first divide the computational domain into a square grid. The size of the squares is the mean panel length. Each panel l_i then belongs to every square such that a portion of it is at a distance of less than or equal to $|l_i|$ from the mid-point of the panel. Now, for each point z we determine in which square it is situated and only check for closeness against the panels belonging to that particular square.

It is important to know what happens if extended or special quadrature is activated by mistake, that is, when straight-up quadrature gives better results. If extended quadrature is activated by mistake, the consequences are barely noticeable. If the special quadrature of Section 5.3 is activated for z far away from l_i, it may happen that the denominator in (55) becomes exactly zero. Apart from this, it is robust when used to evaluate an integral with a well

resolved density just once. A few digits, at most, seem to be lost in inappropriate contexts. The opposite holds for the special quadrature of Section 5.2. It is sensitive to mistakes as it stands. If z is far away from l_i and m is large, then roundoff error in p_1 amplifies greatly in the forward recurrence (37) and $\mathbf{p}$ becomes completely wrong. But for $|z| < 1$ in the transformed plane (57), and for $1 \leq |z| < 2$ and m not too big, the situation is under control. Actually, for $|z| > 1.2$ and $m = 32$, a more accurate $\mathbf{p}$ results from running the recurrence (37) backwards, with p_m computed using extended quadrature in (35). We shall do this.

We remark that McKenney [28] determines the need for modified quadrature from a geometric criterion coupled to the internal structure of the fast multipole method [16], while Atkinson [1] uses repeated refinement doublings in combination with an *aposteriori* error estimate.

7 The computation of p_1

The quantity p_1 of (35) is needed in several situations such as in (55) and (58). It is also needed to initiate the forward recurrence (37) for $\mathbf{p}$ which is subsequently used in (49). In the transformed plane we have

$$p_1 = \int_{-1}^{1} \frac{d\tau}{\tau - z}\,. \tag{59}$$

The path of integration should follow the transformed panel l_i, but it can be deformed without consequences as long as the singularity at z is not passed. We integrate along the real axis and check if the singularity is located between the panel and the real axis. If so, the residue $2\pi i$ is added or subtracted to the value of the integral depending on whether the singularity is above or below the real axis. We get

$$p_1 = \begin{cases} \log(1-z) - \log(-1-z) \pm 2\pi i, & \text{if } z \text{ is between panel and real axis,} \\ \log(1-z) - \log(-1-z), & \text{if it is not.} \end{cases} \tag{60}$$

In our numerical examples, and when doing plots of solutions, we assume that the point z is always in the computational domain D. This assumption simplifies the process of determining whether or not the singularity is located between the panel and the real axis and automatically gives the correct sign of the residue – positive for the interior problem and negative for the exterior problem. In the vast majority of conceivable setups for both problems, provided that the boundaries are properly resolved, the following two requirements are enough to determine if the singularity is located between the panel and the real axis:

- z is below the real axis for interior domains or above for exterior domains,
- z is inside the smallest rectangle containing l_i.

For convex boundaries, this is sufficient regardless of the boundary resolution. For more complex boundaries, the need for proper resolution increases. But the boundary needs to be well resolved anyway to get acceptable overall accuracy. The calculation of p_1 does not require over-resolution of the boundary in any of our numerical tests.

To obtain an equispaced grid of points z with the property $z \in D$, we simply "draw" the boundaries in a matrix of appropriate size and fill the computational domain using a flood fill-type routine. Every filled element of the matrix then corresponds to a point $z \in D$.

8 Avoiding low-threshold stagnation in the GMRES

The generalized minimal residual algorithm (GMRES) is an iterative solver for linear systems $A\mathbf{x} = \mathbf{b}$ [30]. At iteration step s an approximation $\tilde{\mathbf{x}}_s$ is found in the Krylov subspace $\mathcal{K}_s = \text{span}\,\{\mathbf{b}, A\mathbf{b}, \ldots, A^{s-1}\mathbf{b}\}$ which minimizes the residual $||\mathbf{r}_s||_2 = ||\mathbf{b} - A\tilde{\mathbf{x}}_s||_2$. In practice this is done via the Arnoldi process which performs partial reduction of A to Hessenberg form and produces matrices $\tilde{H}_s$ and Q_s such that $AQ_s = Q_{s+1}\tilde{H}_s$. The columns of Q_s, denoted $\mathbf{q}_i$, $i = 1, \ldots, s$, is an ortonormal basis for $\mathcal{K}_s$. The approximation $\tilde{\mathbf{x}}_s$ is represented as $\tilde{\mathbf{x}}_s = Q_s\mathbf{y}_s$ where the vector $\mathbf{y}_s$ is the solution to a $(s+1) \times s$ least squares problem with the upper Hessenberg matrix $\tilde{H}_s$ as system matrix. The solution $\mathbf{y}_s$ can be produced cheaply using QR factorization by Givens rotations. From a computational viewpoint one can see the Arnoldi process as QR factorization performed using column-oriented modified Gram-Schmidt on a matrix with columns given by $\{\mathbf{b}, A\mathbf{q}_1, \ldots, A\mathbf{q}_{s-1}\}$.

A number of implementations for GMRES exist. Gram-Schmidt can be replaced with Householder in the Arnoldi process [34]. Various strategies for stagnation control and restart are available. Linear systems arising from the discretization of integral equations of Fredholm's second kind have $A = I + K_{\text{c}}$, where K_{c} is a matrix which is small in the sense that the spectrum of K_{c} is clustered around the origin. This is good and makes GMRES converge fast. Since no restart is necessary we shall use a simple version of GMRES which chiefly deviates from the classic description [2] in only one respect: we shall feed the Arnoldi process with a matrix whose columns are given by $\{\mathbf{b}, K_{\text{c}}\mathbf{q}_1, \ldots, K_{\text{c}}\mathbf{q}_{s-1}\}$. Clearly, this gives the same Q_s, mathematically, as the standard approach. But the accuracy often is better since for the new column vector that is introduced at step s we avoid a contribution q_{s-1} that in its entirety lies in $\mathcal{K}_{s-1}$. Besides, this modification saves some computational work.

As to compensate for using $K_{\rm c}$ rather than A, unity must be added to the diagonal entries of $\tilde{H}_s$.

The relative residual $||\mathbf{r}_s||_2/||\mathbf{b}||_2$ at step s can be expressed and computed in several mathematically equivalent ways. We choose the cheapest expression which is a product of the entries of the Givens rotations. As it turns out, the combination of this expression and the Krylov subspace generation based on $K_{\rm c}$ allows us to use a stopping criterion threshold in the relative residual that can be almost arbitrarily low without leading to stagnation. In practice, this may be the most important new feature of our GMRES implementation.

9 High polynomial order corrections

Decompose $\mu(t) = P_n[\mu(t)](t) + \mu_{\rm r}(t)$ on panel l_i and assume that there exists a function $h(t) = P_n[h(t)](t) + h_{\rm r}(t)$ such that $\mu_{\rm r}(t) \approx h_{\rm r}(t)$. In other words, if $\mu(t)$ and $h(t)$ are expanded in Legendre polynomials their high order coefficients will be similar. Assume that $\mu(t)$ is known at the nodes t_{nk} while $h(t)$ is known both at the nodes t_{nk} and the nodes t_{mk} and that $\mathbf{h}_n \in \mathbb{R}^n$ and $\mathbf{h}_m \in \mathbb{R}^m$ are vectors with entries given by $h(t_{nk})$, $k = 1, \ldots, n$ and $h(t_{mk})$, $k = 1, \ldots, m$. Then, rather than interpolating $\mu(t)$ at t_{mk} via $(WV^{-1})\boldsymbol{\mu}_i$ one can use the corrected formula $(WV^{-1})(\boldsymbol{\mu}_i - \mathbf{h}_n) + \mathbf{h}_m$ and modify the special quadrature of (44) to

$$I_i(z) \approx \Im\left\{\mathbf{p}^{\rm T}C^{-1}\right\}(WV^{-1})(\boldsymbol{\mu}_i - \mathbf{h}_n) + \Im\left\{\mathbf{p}^{\rm T}C^{-1}\right\}\mathbf{h}_m\,. \tag{61}$$

This modification should give improved accuracy for short-ranged interaction since high order polynomial components of $\mu(t)$, not captured by the non-modified quadrature are cancelled by similar components in $h(t)$, and then corrected by a more accurate quadrature for $h(t)$.

We shall use the corrected quadrature (61) when evaluating (5) after solving (7) for the interior problem. The function $h(t)$ here corresponds to $2f(z(t))$. Since the integral operator of (6) is compact the difference $\mu(t) - 2f(t)$ should have a particularly small high order polynomial content.

The usefulness of (61) is further illustrated in Figure 1, which compares polynomial interpolation via $(WV^{-1})\boldsymbol{\mu}_i$ with the high polynomial order corrected interpolation and also with Nyström interpolation. The setting is (6) with L parameterized as in (63) and $f(z) = \Re\{1/(z - 1 - i)\}$. The boundary is discretized using 40 panels of equal parameter length and with $n = 16$ and $m = 32$. The density $\mu(t)$ is first solved via (7) at the 640 discretization points t_{nk}. Actually, $\mu(t)$ is partly overresolved in the sense that some panels are shorter than what is needed for full pointwise convergence of $\mu(t)$. The com-

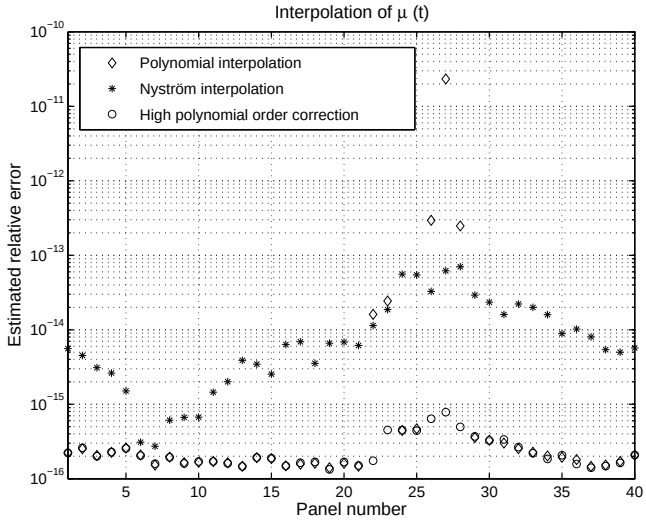

Fig. 1. Accuracy for various interpolations of the solution $\boldsymbol{\mu}$ to (7) for an interior problem. The boundary is divided into 40 panels with 16 discretization points each.

puted $\boldsymbol{\mu}$ from (7) is interpolated to the 1280 points t_{mk} and compared with a reference solution obtained by solving (7) again. This time on a grid consisting of the points t_{mk}. The estimated relative error is computed pointwise in max-norm and averaged over each panel. Three observations can be made in Figure 1: Polynomial interpolation via $(WV^{-1})\boldsymbol{\mu}_i$ is accurate where $\mu(t)$ is overresolved but not so accurate where $\mu(t)$ is just resolved. Nyström interpolation falls short of its theoretical promises with a factor of about 100. The high polynomial order corrected interpolation is the most accurate.

High polynomial order corrections could also have large impact in the context of differentiation, assuming that $h(t)$ rather than being known at extra nodes t_{mk} can be differentiated analytically at the nodes t_{nk}. For the numerical evaluation of $(D_t^1\mu)(t_j)$ in (25) one can use the corrected expression

$$(D_t^1(\mu - h))(t_j) + (D_t^1 h)(t_j)\,, \tag{62}$$

where the first term is computed numerically and the second term analytically. In the interior problem $h(t)$ corresponds to $2f(z(t))$ and in the exterior problem $h(t)$ corresponds to $-2f(z(t)) + 2\sum a_k \log|z(t) - z_k|$.

10 Numerical Examples

The techniques proposed will now be demonstrated numerically. Most of the programming is done in MATLAB 7.3, but time critical parts, such as the fast multipole method (FMM), are written in C, compiled with MATLAB's in-built compiler, and interfaced with the MATLAB environment using MEX.

Our computer is an Intel Core2 6400 at 2.13 GHz with 4 gigabytes of installed memory. The FMM is used extensively. Our implementation follows Ref. [16] with precision $\epsilon = 10^{-13}$. The multiplication by K in (7) and (18) is done by the FMM as is the evaluation of the discretized integrals in (5) and (11). The effect of the FMM is that of using the straight-up quadrature of (32) everywhere. Corrections due to point-panel pairs requiring other quadrature are applied subsequently. The FMM is also used when evaluating the logarithmic terms of (11) and (18).

Except for when explicitly stated otherwise we choose a mesh that is sufficiently refined with respect to the regularity of the boundary and the boundary data so that the unknown density, should it be known, could be accurately interpolated with composite 16-point polynomial interpolation at the discretization points.

10.1 The interior Dirichlet problem

We compute the solution to an interior Dirichlet problem. The boundary has parameterization

$$z(t) = (1 + 0.3\cos 5t)e^{it}\,, \quad -\pi \le t \le \pi\,. \tag{63}$$

We wish to determine the error in the numerical solution accurately and choose boundary conditions compatible with a reference solution $U_{\text{ref}}(z)$, known on closed form. We take

$$U_{\text{ref}}(z) = \Re\left\{\sum_{k=1}^{3} \frac{1}{z - z_k}\right\}\,, \quad z \in D\,, \tag{64}$$

where $z_1 = 1.5 + 1.5i$, $z_2 = -0.25 + 1.5i$, and $z_3 = -0.5 - 1.5i$ are sources outside of D. The discretization is carried out using 35 panels of equal length in the parameter t. This gives 560 discretization points over the boundary. When computing $\boldsymbol{\mu}$ of (7) we use GMRES without restart as described in Section 8 and the FMM. This scheme, in our implementation, is somewhat more accurate than MATLAB's backslash operator and also faster already at this system size. The precise breakeven size depends, of course, on many implementational details. The stopping criterion threshold of 10^{-16} in the relative residual is reached after 16 iterations.

We now evaluate $U(z)$ of (5) on a grid with 483519 equispaced points z in D. For comparison we first use only straight-up quadrature. Figure 2 depicts the relative pointwise error

$$e(z) = \frac{|U_{\text{ref}}(z) - U(z)|}{||U_{\text{ref}}(z)||_\infty}\,, \tag{65}$$

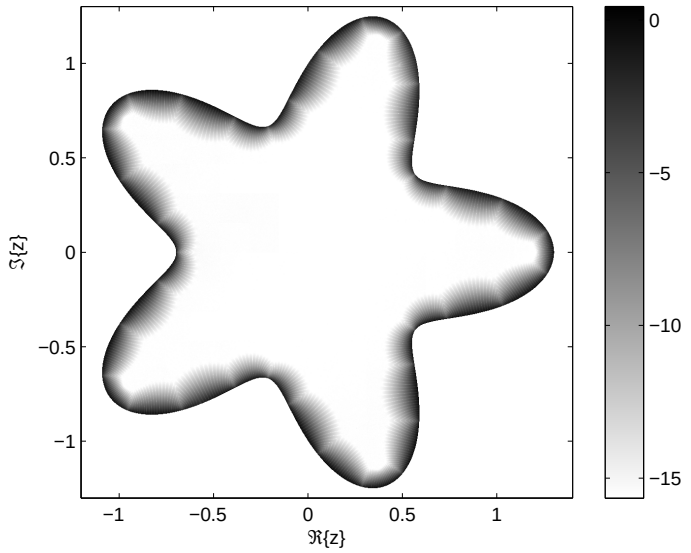

Fig. 2. The 10-logarithm of the pointwise error $e(z)$ of (65) when solving the interior Dirichlet problem with boundary conditions from (64) and using only straight-up quadrature for evaluating $U(z)$ of (5). There are 560 discretization points on the boundary and $U(z)$ is evaluated at 483519 points. The maximum value of $e(z)$ is 2.78.

where $||U_{\text{ref}}(z)||_\infty$ is the maximum absolute value of $U_{\text{ref}}(z)$ over all grid points. Figure 2 shows that the quality of the solution deteriorates rapidly as z approaches the boundary, as stated in the second paragraph of the introduction. The maximum relative pointwise error is 2.78. The relative Euclidean error is $7.6 \cdot 10^{-2}$.

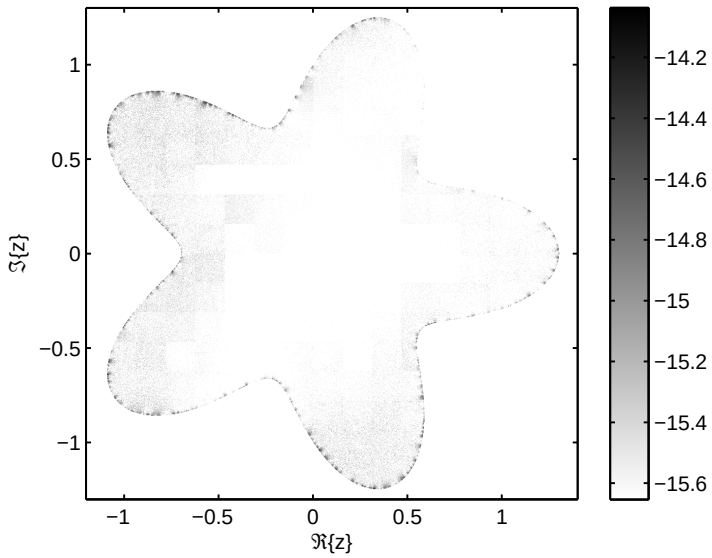

Fig. 3. Same as in Figure 2, but globally compensated quadrature is used. The maximum value of $e(z)$ is $9.2 \cdot 10^{-15}$.

The result of a similar evaluation using the globally compensated quadrature of Section 3 with high order polynomial corrections as at the end of Section 9 is shown in Figure 3. The difference is enormous. Compare with Figure 2 and note the very different scales in the colorbar. The maximum relative pointwise error here is only $9.2 \cdot 10^{-15}$. The relative Euclidean error is down to $5.6 \cdot 10^{-16}$. One could almost say that there is no error.

As for timings, we present results for a sparser grid with 19015 equispaced points in D. The linear 560×560 system (7) is solved for $\boldsymbol{\mu}$ in 0.19 seconds. Evaluation of $U(z)$ of (5) with straight-up quadrature also takes 0.19 seconds. Evaluation of $U(z)$ of (5) with globally compensated quadrature takes 0.36 seconds. The time for generating the grid points themselves not included.

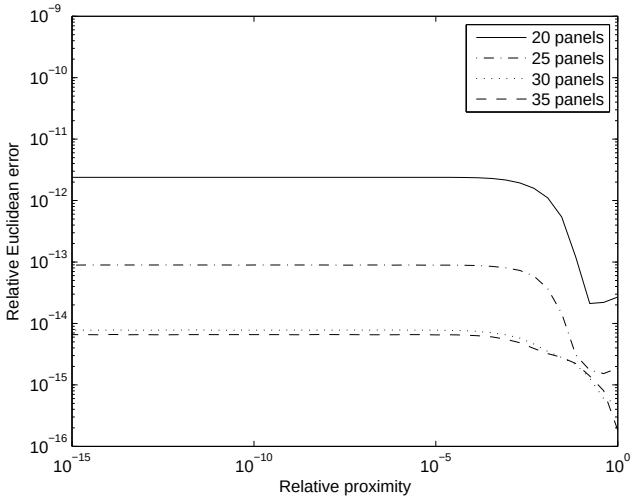

Fig. 4. The relative Euclidean error in $U(z)$ of (5) as a function of the relative proximity of z to the boundary (63) and of the numbers of panels.

The 483519 gridpoint problem is not really challenging as far as extreme closeness is concerned. The smallest distance from a point z to the boundary is on the order of 10^{-3}. As a harder test, we calculate the solution at points z_p of a scaled-down version of the discretized boundary

$$z_p = (1 - r)z \,, \tag{66}$$

where z are 1000 points of (63) equispaced in parameter and where the relative proximity to the boundary r is varied in the interval $10^{-15} \leq r \leq 10^0$. This test will take us right up to the numerical limit of closeness. We use globally compensated quadrature. Figure 4 shows a relative Euclidean error that, for 30 panels or more, is bounded by $40\epsilon_{\text{mach}}$. This seems to be an improvement in several ways over, for example, Table 1 of Ref. [1] where r varies in the interval $10^{-2} \leq r \leq 10^0$ and D is an ellipse.

All tests above for the evaluation of $U(z)$ of (5) have also been carried out using

the quadratures of Section 5.2 with panel mergers and splits, high polynomial order corrections, the parameter *tol* of Section 6 set to ϵ_{mach}, and backward recursion for $\mathbf{p}$ whenever $|z| > 1.2$ in the transformed plane. We show no plots. The results are similar to, or only marginally worse than, those obtained with globally compensated quadrature. With 36 panels, corresponding to 576 discretization points on the boundary, for the 483519 gridpoint problem we get a maximum relative pointwise error of $5.6 \cdot 10^{-14}$ and a relative Euclidean error of $1.4 \cdot 10^{-15}$. The evaluation of $U(z)$ for the 19015 gridpoint problem takes 0.39 seconds. But again, the globally compensated quadrature is specially crafted for the Dirichlet problem on closed contours, while the quadratures of Section 5.2 are more generally applicable. In conclusion, our techniques are robust and capable of computing the solution to very high accuracy in the entirety of the domain using relatively few discretization points. These favorable traits also come at a reasonable price, speed-wise.

10.2 The exterior Dirichlet problem

The exterior Dirichlet problem will be solved on two different types of domains. We begin with a domain D exterior to a collection of M circles with centers in a unit cell and choose boundary conditions compatible with the reference solution

$$U_{\text{ref}}(z) = c + \sum_{k=1}^{M} d_k \log(|z - s_k|^2)\,, \quad z \in D\,, \tag{67}$$

where s_k is a point within each circle. We take $c = 1$ and

$$d_k = 2(k-1)/(M-1) - 1\,. \tag{68}$$

The condition number of this type of problem seems to increase as the relative distance between neighboring boundaries gets smaller [23]. It is then of interest to compare the performance of our methods on setups where the distances between the circles are somehow controlled. The relative closeness of two circles can be expressed in terms of a closeness factor

$$f_{\text{cl}} = \frac{2\pi r_{\text{l}}}{|z_{\text{l}} - z_{\text{s}}| - r_{\text{l}} - r_{\text{s}}}\,, \tag{69}$$

where r_{l}, z_{l}, r_{s}, and z_{s} are the radii and centra of the larger and smaller circles, respectively. Setting an upper limit, f_{clup}, to f_{cl} prohibits circles getting too close, and in particular small circles getting close to large ones. The higher the value of f_{clup}, the lesser these restrictions become.

We let $M \in [10, 2000]$ and $f_{\text{clup}} \in [10, 10000]$. Different setups are created as follows: For a given f_{clup}, start with $M = 2000$ and choose a number β. Then for each $k = 1, \ldots, M$, pick a random radius r_k in the interval $[\beta, 30\beta]$

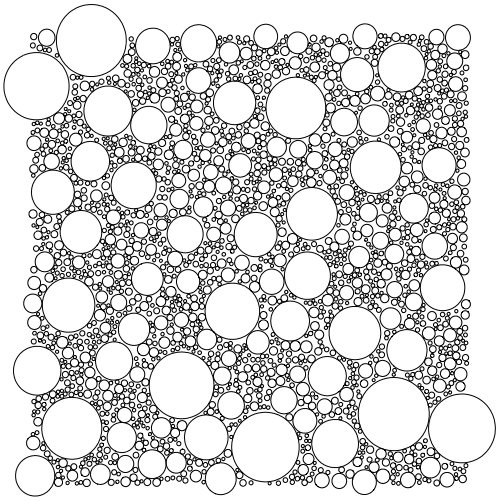

Fig. 5. A domain exterior to $M = 2000$ circles with centers in a unit cell and with closeness factor upper limit $f_{\text{clup}} = 10000$.

and a random position z_k in the unit cell, place the circle centered at z_k if allowed by f_{clup}, otherwise pick a new radius and a new position and repeat until the circle fits and choose s_k randomly in a smaller circle, centered at z_k and with radius $0.4r_k$. Once all circles are placed, the values d_k are assigned to these randomly. The number β is chosen high, but with the constraint that M circles can be placed with reasonable ease. This results in area fractions of circles in the unit cell varying from around 35% for $f_{\text{clup}} = 10$ to 80% for $f_{\text{clup}} = 10000$. Geometries with $M < 2000$ are created as magnified parts of the $M = 2000$ geometry. The values d_k are again assigned randomly. Figure 5 shows the most challenging geometry.

The numerical solution $U(z)$ is calculated via (11,17,18) at those points on a 1200×1200 Cartesian grid, covering the unit cell, that belong to D. For the integral equation, we use 16 panels corresponding to 256 discretization points on each circle and activate the quadratures of Section 5.3 whenever their corresponding criteria are met according to Section 6 with $tol = \epsilon_{\text{mach}}$. The quadratures of Section 5.2 are used for (11). Results for a large number of M and different f_{clup} are shown in Figures 6 and 7. The error does not depend much on f_{clup}. The accuracy in Figure 6 is equally high irrespective of whether the circles are close or far apart from each other. The error seems to depend only on M and grows linearly with the relative error in the positions of the discretization points due to finite precision arithmetic. Figure 7 shows that, for a given f_{clup}, the number of GMRES iterations required to reach a relative residual of 10^{-16} converges with increasing M. There is also some indication that, for a given M, the number of iterations converges with increasing f_{clup}.

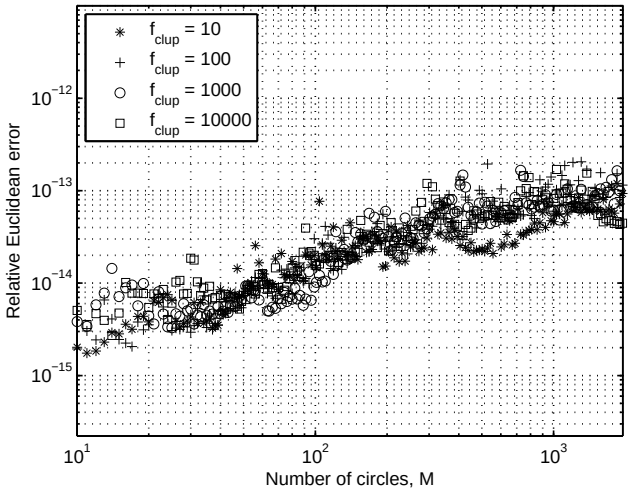

Fig. 6. The relative Euclidean error in the numerical solution $U(z)$ to the exterior problem (8,9,10) as a function of the number of circles M. The error does not depend on the closeness of the circles. Its growth seems to be quite close to the ideal $O(M^{0.5})$. There are 256 discretization points on each circle.

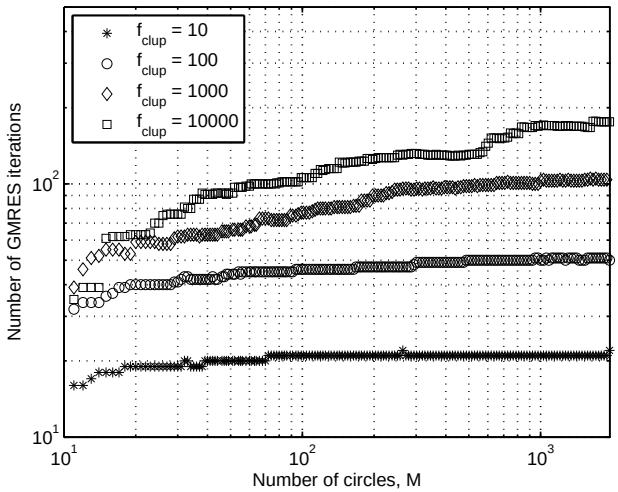

Fig. 7. Number of GMRES iterations needed to solve the system (18), with a stopping criterion threshold in the relative residual of 10^{-16}, as a function of the number of circles M and for various closeness factor upper limits f_{clup}.

In terms of speed and memory requirements our methods are cheap indeed. To illustrate this, we time the solution for a setup with $M = f_{\text{clup}} = 1000$. The number of unknowns in the linear system (18) is 256000. A total of 622640 corrections to the entries of the system matrix are required due to activated extended or special quadrature. Actually, the matrix entries are doubly corrected: first the correct entries are computed and then the erroneous values that would result from straight-up quadrature are subtracted. In this way the output from the FMM only has to be corrected once in the iterative solver. It

takes 45 seconds to locate and to precompute the doubly corrected entries with the methods of Sections 5.3 and 6. Their storage requires about 5 megabytes of memory. GMRES needs 87 iterations to reach the stopping criterion threshold of 10^{-16} in the relative residual, and requires 2 megabytes of memory for each new Krylov subspace vector. The total memory usage of GMRES comes to around 174 megabytes, so the storage of precomputed entries is only 3% of this. Speed-wise, the total time for solving the integral equation is 1388 seconds, about 3% of which, thus, is spent on precomputing corrected entries and 1% is spent on applying corrected entries in the GMRES solver.

Finally, we shall solve a problem which does not have a closed form solution. As for Dirichlet data (9), we choose the Gibbs-Thompson boundary conditions

$$f(\tau) = \kappa(\tau)\,, \quad \tau \in L\,, \tag{70}$$

where $\kappa(\tau)$ denotes curvature. Rather than evaluating the solution $U(z)$ we shall compute its normal derivative at L, the Dirichlet–Neumann map

$$n_z \cdot \nabla U(z) = \Im\left\{ \frac{n_z}{2\pi i} \int_L \frac{\bar{n}_\tau \frac{\mathrm{d}\mu(\tau)}{\mathrm{d}\sigma_\tau}\,\mathrm{d}\tau}{\tau - z} \right\} + \sum_{k=1}^{M} a_k \Re\left\{ \frac{n_z}{z - z_k} \right\}, \quad z \in L\,, \tag{71}$$

where $n_z = (n_x, n_y) = n_x + i n_y$ is the outward unit normal of L at z and $\bar{n}_z$ is its complex conjugate.

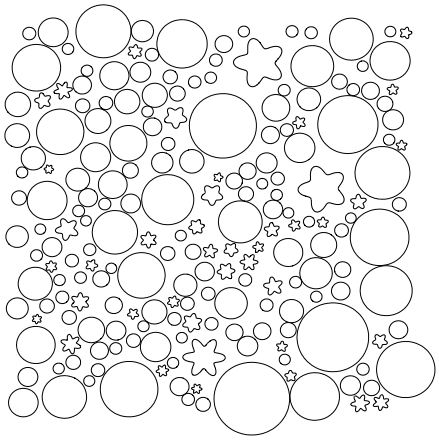

Fig. 8. The geometry used when computing the Dirichlet-Neumann map (71).There are 200 boundaries, a fifth of which are starfish-looking and parameterized as modifications of (63). The rest are circles.

We use a geometry exterior to 200 boundaries, about a fifth of which are parameterized as slight modifications of (63) and the rest are circles. See Figure 8. Computing the Dirichlet-Neumann map over this geometry is a rather hard problem. The normal derivative of $U(z)$ at L oscillates rapidly due to

the varying curvature of the boundaries, and also because small circles lie close to larger ones. Ideally, adaptive mesh refinement in combination with some preconditioning should be used. Anyhow, we stick to a uniform mesh and increase resolution of the boundaries to 64 panels corresponding to 1024 discretization points on each circle and 320 panels corresponding to 5120 discretization points on each boundary of the modified (63). Using a reference solution computed on a mesh with twice this resolution, we achieve an estimated relative Euclidean error of $2 \cdot 10^{-10}$.

A few closing words about robustness are in order. We have tried to challenge our quadrature formulas by a series of additional experiment on problems with smooth boundaries and boundary data; using panels of wildly different sizes, odd refinements and boundaries that fall back onto and almost touch themselves. Still, we have not succeeded in inducing failures. Provided one follows the guidelines given throughout the paper, the computed solution should be about as accurate as allowed by the prescribed tolerance and the conditioning of the underlying mathematical problem in all but very exotic circumstances.

11 Discussion

Elliptic boundary value problems are often well-conditioned in a mathematical sense. It should be possible to solve them extremely rapidly and with a minimal loss of precision, at least on smooth domains in the plane. A number of generally efficient boundary integral solvers have been developed over the years [37]. Still, in our opinion, there is room for improvement when it comes to finding the best integral equation for a given problem, finding strategies for mesh refinement, and treating points and boundaries that lie close to each other. This paper is chiefly on the last topic.

A special situation is when a double layer density can be accurately converted into the boundary value of an analytic function on a closed contour. Section 3, here, suggests a globally compensated quadrature for the evaluation of the potential in the entire computational domain. Considering its efficiency and striking simplicity one would think that it should already have appeared in the literature on solving Laplace's equation. Perhaps it has? But we have only been able to find the barely cited paper [21], containing the basic idea on the evaluation of analytic functions by Cauchy's theorem.

The bulk of the paper is on more general and almost equally efficient quadratures for double layer potentials which should be applicable not only to the Dirichlet problem for Laplace's equation but also to problems in Stokes flow and elasticity where similar double layers are used [4,9,14,22]. Of particular interests here are applications to large-scale problems for composite materials

and fracture mechanics. Inclusions that lie close to each other or have corners, adjacent grains that meet at triple-junctions [31], and growing cracks with kinks [10] are common features which, in theory, can be resolved with established integral equation methods, but where problems related to that the boundary lies close to itself destroy the performance so much that these methods are seldom used but for rather small problems. The techniques presented in this paper can not alone solve all the problems associated with such geometries efficiently, but they could be key ingredients in successful and more complex algorithms and in this way help to change this situation.

We also give a new twist to the GMRES method, a competitive alternative to Nyström interpolation, and an efficient formulation of Mikhlin's method in mode II. These improvements help in getting the most out of the calculations.

Appendix

A High-degree polynomial interpolation

We briefly review the mechanisms of polynomial interpolation in order to explain why high-degree interpolation of sufficiently smooth functions on the interval $[-1, 1]$ can give accurate results once the underlying function is resolved. At least up to degree 40 in double precision arithmetic.

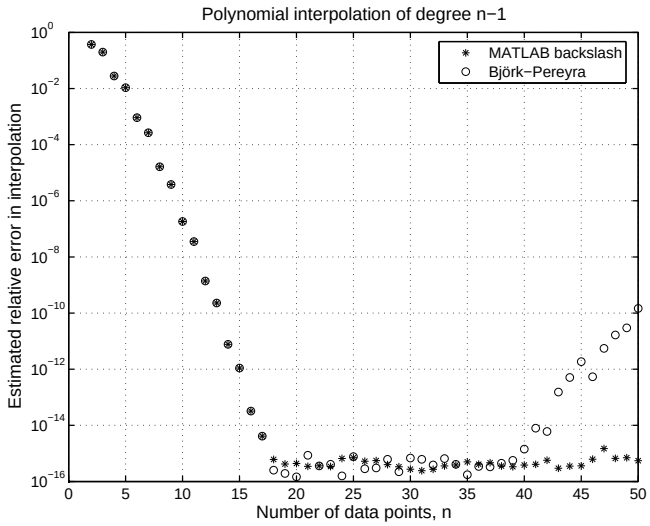

Fig. A.1. Polynomial interpolation of degree $n-1$ of the function $f(x) = \cos(2(x+1))$ in the interval $x \in [-1, 1]$. The n data points are bunched near the ends of the interval. The interpolating polynomial is evaluated at 1000 equispaced points and the relative Euclidean error is computed. Polynomial coefficients are computed by solving a Vandermonde system by MATLAB's backslash operator and by the Björk-Pereyra method.

High-degree polynomial interpolation involves solving an ill-conditioned Vandermonde system for unknown coefficients. The condition number of the Vandermonde matrix grows exponentially with the number of data points n. The error in the computed coefficients of the interpolating polynomial will grow in a similar fashion. Still, the actual interpolation can be very accurate, at least if the data points are bunched near the ends of the interval (e.g. Chebyshev points or Legendre points), if the underlying function is sufficiently smooth, and if a backward stable linear solver is used. This is well known, see, for example, Chapters 2 and 7 of Ref [18] and in particular pages 62-63, 315, and 325. The key observation is that even though the solution to the linear system is inaccurate, the relative residual is small. A small residual means that the numerical interpolation error, that is, the difference between the interpolating polynomial with computed coefficients and the interpolating polynomial with exact coefficients, is a polynomial of degree $n-1$ that practically vanishes at n points. Using an argument based on Lagrange basis functions one can show that the numerical interpolation error then is very small everywhere in the interval.

As an illustration we interpolate the function $f(x) = \cos(2(x+1))$ in the interval $x \in [-1, 1]$ and let the x-values of the n data points be taken as the nodes of the Legendre polynomial of order n. The condition number of the Vandermonde system starts at unity for $n = 1$ and reaches $O(10^{16})$ for $n = 43$. Figure A.1 depicts the relative error of interpolating functions with coefficients computed using MATLAB's backslash operator ($O(2n^3/3)$ FLOPs) and with coefficients computed using the Björk-Pereyra method **Algorithm 4.6.1** in Ref. [13] ($O(5n^2/2)$ FLOPs). The interpolating polynomials are evaluated at 1000 equidistant points in the interval and compared with true values of $f(x)$. The relative error is in Euclidean norm. Figure A.1 shows that the function $f(x)$ is resolved with $n = 18$ data points and that both MATLAB's backslash and the Björk-Pereyra method produce excellent results up to at least $n \approx 40$ data points.

References

[1] K. Atkinson and Y. Jeon, Algorithm 788: Automatic boundary integral equation program for the planar Laplace equation, ACM Trans. Math. Softw., **24**(4), 395–417 (1998), doi:10.1145/293686.293692

[2] R. Barrett, M. Berry, T. F. Chan, J. Demmel, J. Donato, J. Dongarra, V. Eijkhout, R. Pozo, C. Romine, and H. Van der Vorst: *Templates for the Solution of Linear Systems: Building Blocks for Iterative Methods*, 2nd edition, SIAM, (Philadelphia, PA, 1994).

[3] J.T. Beale and M.-C. Lai, A Method for Computing Nearly Singular

Integrals, SIAM J. Numer. Anal., **38**(6), 1902–1925 (2001), doi:10.1137/S0036142999362845

[4] G. Biros, L. Ying, and D. Zorin, A fast solver for the Stokes equations with distributed forces in complex geometries, J. Comput. Phys. **193**(1), 317–348 (2004), doi:10.1016/j.jcp.2003.08.011

[5] H. Cheng and L. Greengard, On the numerical evaluation of electrostatic fields in dense random dispersions of cylinders, J. Comput. Phys. **136**(2), 629–639 (1997), doi:10.1006/jcph.1997.5787

[6] V. Cristini, J. Lowengrub, and Q. Nie, Nonlinear simulation of tumor growth, J. Math. Biol. **46**(3), 191–224 (2003), 10.1007/s00285-002-0174-6

[7] A. Dejoie, S.G. Mogilevskaya, and S.L. Crouch, A boundary integral method for multiple circular holes in an elastic half-plane, Engng Anal. Boundary Elem. **30**(6), 450–464 (2006), doi:10.1016/j.enganabound.2005.12.005

[8] A. Dutt, M. Gu, and V. Rokhlin, Fast Algorithms for Polynomial Interpolation, Integration, and Differentiation, SIAM J. Numer. Anal., **33**(5), 1689–1711 (1996), doi:10.1137/0733082

[9] J. Englund, A Nyström scheme with rational quadrature applied to edge crack problems, Comm. Num. Meth. Engng, **23**(10), 945–960 (2007) doi: 10.1002/cnm.937

[10] J. Englund, A higher order scheme for two-dimensional quasi-static crack growth simulations, Comput. Methods Appl. Mech. Engrg., **196**(21-24), 2527–2538 (2007), doi:10.1016/j.cma.2007.01.007

[11] L. Farina, Evaluation of single layer potentials over curved surfaces, SIAM J. Sci. Comp., **23**(1), 81–91 (2001), doi:10.1137/S1064827599363393

[12] W. Gautschi, The use of rational functions in numerical quadrature, J. Comput. Appl. Math. **133**(1-2), 111–126 (2001), doi:10.1016/S0377-0427(00)00637-3

[13] G.H. Golub and C.F Van Loan, Matrix Computations – 2nd ed., The John Hopkins University Press, Baltimore, 1989.

[14] L. Greengard and M.C. Kropinski, Integral equation methods for Stokes flow in doubly-periodic domains, J. Eng. Math. **48**(2), 157–170 (2004), doi:10.1023/B:ENGI.0000011923.59797.92.

[15] L. Greengard and J.-Y. Lee, Electrostatics and heat conduction in high contrast composite materials, J. Comput. Phys. **211**(1), 64–76 (2006), doi:10.1016/j.jcp.2005.05.004

[16] L. Greengard and V. Rokhlin, A fast algorithm for particle simulations, J. Comput. Phys. **73**(2), 325–348 (1987), doi:10.1016/0021-9991(87)90140-9

[17] A. Greenbaum, L. Greengard, and G.B. McFadden, Laplace's equation and the Dirichlet–Neumann map in multiply connected domains, J. Comput. Phys. **105**(2), 267–278 (1993), doi:10.1006/jcph.1993.1073

[18] M.T. Heath, Scientific Computing: An introductory survey – 2nd ed., McGraw-Hill, New York, 2002.

[19] J. Helsing, Thin bridges in isotropic electrostatics, J. Comput. Phys. **127**(1), 142–151 (1996), doi:10.1006/jcph.1996.0164

[20] J. Helsing and E. Wadbro, Laplace's equation and the Dirichlet-Neumann map: a new mode for Mikhlin's method, J. Comput. Phys. **202**(2), 391–410 (2005), doi:10.1016/j.jcp.2004.06.024

[21] N.I. Ioakimidis, K.E. Papadakis, and E.A. Perdios, Numerical evaluation of analytic functions by Cauchy's theorem, BIT Numer. Math., **31**(2), 276–285 (1991), doi:10.1007/BF01931287

[22] A. Jonsson, Discrete dislocation dynamics by an O(N) algorithm, Comput. Mater. Sci., **27**(3), 271–288 (2003), doi:10.1016/S0927-0256(02)00363-4

[23] H.-J. Jou, P.H. Leo, and J.S. Lowengrub, Microstructural evolution in inhomogeneous elastic media, J. Comput. Phys. **131**(1), 109–148 (1997), doi:10.1006/jcph.1996.5581.

[24] M.A. Khayat and D.R. Wilton, Numerical evaluation of singular and near-singular potential integrals, IEEE Trans. Antennas Propag., **53**(10), 3180–3190 (2005), doi:10.1109/TAP.2005.856342

[25] P.G. Martinsson, Fast evaluation of electro-static interactions in multi-phase dielectric media, J. Comput. Phys **211**(1), 289–299 (2006), doi: 10.1016/j.jcp.2005.05.018

[26] A. Mayo, Fast high-order accurate solution of Laplace's equation on irregular regions, SIAM J. Sci. Stat. Comp. **6**(1), 144–157 (1985), doi:10.1137/0906012

[27] A. Mayo and A. Greenbaum, Fourth order accurate evaluation of integrals in potential theory on exterior 3D regions J. Comput. Phys **220**(2), 900–914 (2007), doi:10.1016/j.jcp.2006.05.042

[28] A. McKenney, An adaptation of the fast multipole method for evaluating layer potentials in two dimensions, Computers Math. Applic., **31**(1), 33–57 (1996), doi:10.1016/0898-1221(95)00181-W

[29] G. Monegato and A. Palamara Orsi, Product formulas for Fredholm integral equations. In *Numerical Integration III, International Series of Numerical Mathematics* vol. **85**, H. Braß H. and G. Hämmerlin (eds), Birkhäuser, (Basel, 1988), pp. 140–156

[30] Y. Saad and M.H. Schultz, GMRES: A generalized minimal residual algorithm for solving nonsymmetric linear systems, SIAM J. Sci. Stat. Comp. **7**(3), 856–869 (1986), doi: 10.1137/0907058

[31] J.A Sethian and J. Wilkening, A numerical model of stress driven grain boundary diffusion, J. Comput. Phys. **193**(1), 275–305 (2004), doi:10.1016/j.jcp.2003.08.015

[32] K. Thornton, N. Akaiwa, and P.W. Voorhees, Large-scale simulations of Ostwald ripening in elastically stressed solids: I. Development of microstructure, Acta Mater., **52**(5), 1353–1364 (2004), doi:10.1016/j.actamat.2003.11.037

[33] G. Tsamasphyros and E.E. Theotokoglou, A quadrature formula for integrals with nearby singularities, Int. J. Numer. Meth. Engng **67**(8), 1082–1093 (2006), doi:10.1002/nme.1649

[34] H.F. Walker, Implementation of the GMRES method using Householder transformations, SIAM J. Sci. Stat. Comp. **9**(1), 152–163 (1988), doi:10.1137/0909010

[35] J.A.C. Weideman and D.P. Laurie, Quadrature rules based on partial fraction expansions, Numer. Algo. **24**(1-2), 159–178 (2000), doi:10.1023/A:1019145327098

[36] N. Yarvin and V. Rokhlin, Generalized Gaussian quadratures and singular value decompositions of integral operators, SIAM J. Sci. Comp. **20**(2), 699–718 (1998), doi: 10.1137/S1064827596310779

[37] L. Ying, G. Biros, and D. Zorin, A high-order 3D boundary integral equation solver for elliptic PDEs in smooth domains, J. Comput. Phys. **219**(1), 247–275 (2006), doi:10.1016/j.jcp.2006.03.021