Technical Notes

Sixth-Order Accurate Direct Solver for the Poisson and Helmholtz Equations

Patrick J. Roache*

*Ecodynamics Research Associates, Inc.,
Albuquerque, N. Mex.*

Introduction

THE Poisson and Helmholtz equations are some of the most common and important equations of mathematical physics. Their solution is required in subsonic aerodynamics problems, steady-state heat conduction problems, and electric field calculations, as well as in other areas. While iterative methods are easy to code and flexible, they are notoriously slow in converging for large meshes. Beginning with Hockney's paper[1] in 1965, direct (noniterative) methods were frequently used because of their great speed advantage. A complete review of direct methods would require a large review paper, since this field has become very active in recent years. Almost all the methods published to date are limited to second-order accuracy, i.e., the truncation error is $O(h^2)$ where h is the mesh spacing. An exception is Ref. 2, which presents a method that is $O(h^4)$, but is limited to the constant-coefficient equation in Cartesian coordinates.

In the present paper, a direct solver is described for a class of separable elliptic equations in $r\theta$ coordinates that is $O(h^6)$ accurate provided that the problem is periodic in θ. In r, the Hermite 6 discretization of Rubin and Khosla[3] is used, giving $O(\Delta r^6)$ accuracy at interior points. The basic equation used at boundary points is $O(\Delta r^4)$ but is corrected in a single-deferred correction step to $O(h^6)$ accuracy.

The Method

The following elliptic equation is solved directly, with periodic boundary conditions in θ and combinations of Dirichlet (function) and Neumann (gradient) conditions in r. Subscripts indicate partial derivatives.

$$a(r)\psi_{rr} + b(r)\psi_r + c(r)\psi + \psi_{\theta\theta} = d(r,\theta) \quad (1)$$

With the proper choice of a-d, this equation can represent the Poisson or Helmholtz equation in Cartesian, polar, elliptic, biaxial, or parabolic cylinder coordinates, including an arbitrary coordinate transformation in r. The only requirement that will be imposed by the use of the Fast Fourier Transform (FFT) in θ is that the coefficients be separable and the θ term be simply $\psi_{\theta\theta}$. (An arbitrary constant can be absorbed by scaling, of course.) Other coordinates may not meet these requirements; e.g., spherical, prolate spheroidal, oblate spheroidal, toroidal, bipolar, and paraboloidal.[4] An important system which does meet these requirements is the conformal coordinate system for Joukowski airfoils used by Mehta[5] to approximate real airfoils.

The discrete Fourier transform is applied to Eq. (1) using the FFT algorithm. This transforms the variable $\psi(r,\theta)$ to $\hat{\psi}(r,k)$ where k is the wave number in the θ direction. As in Ref. 1, this gives

$$a(r)\hat{\psi}_{rr} + b(r)\hat{\psi}_r + \{c(r) - k^2/4\}\hat{\psi} = \hat{d}(r,k) \quad (2)$$

where $\hat{d}(r,k)$ is the discrete Fourier transform of $d(r,\theta)$. Equation (2) is now a decoupled system of ordinary differential equations in r (with k as a parameter) that is solved by Hermite 6 discretization.[3] A final backward FFT then gives ψ.

In the notation of Ref. 3, Eq. (2) is rewritten with $u \equiv \hat{\psi}$, $m \equiv u_r$, and $M \equiv u_{rr}$, as

$$AM + Bm + Cu = D \quad (3)$$

where A-D are obvious shorter notations for the coefficients in Eq. (2). The Hermite 6 discretization provides two more equations, as follows. The index j is the index for the discrete node points in r, equally spaced h apart.

$$M_{j+1} - 8M_j + M_{j-1} = -(24/h^2)(u_{j+1} - 2u_j + u_{j-1})$$
$$+ (9/h)(m_{j+1} - m_{j-1}) \quad (4)$$

$$7m_{j+1} + 16m_j + 7m_{j-1} = (15/h)(u_{j+1} - u_{j-1})$$
$$+ h(M_{j+1} - M_{j-1}) \quad (5)$$

Equations (3-5) are three equations in the three unknowns u,m,M. Since Eq. (3) applies at all points, we can write it at index values j and $j \pm 1$ and use it to eliminate all M from Eqs. (4) and (5), giving

$$-\left(\frac{-9}{h} + \frac{B_{j-1}}{A_{j-1}}\right)m_{j-1} + \left(8\frac{B_j}{A_j}\right)m_j - \left(\frac{9}{h} + \frac{B_{j+1}}{A_{j+1}}\right)m_{j+1}$$

$$-\left(\frac{C_{j-1}}{A_{j-1}} - \frac{24}{h^2}\right)u_{j-1} + \left(8\frac{C_j}{A_j} - \frac{48}{h^2}\right)u_j - \left(\frac{C_{j+1}}{A_{j+1}} - \frac{24}{h^2}\right)u_{j+1}$$

$$= -\frac{D_{j-1}}{A_{j-1}} + 8\frac{D_j}{A_j} - \frac{D_{j+1}}{A_{j+1}} \quad (6)$$

$$-\left(h\frac{B_{j-1}}{A_{j-1}} - 7\right)m_{j-1} + 16m_j - \left(-h\frac{B_{j+1}}{A_{j+1}} - 7\right)m_{j+1}$$

$$-\left(h\frac{C_{j-1}}{A_{j-1}} - \frac{15}{h}\right)u_{j-1} - \left(-h\frac{C_{j+1}}{A_{j+1}} + \frac{15}{h}\right)u_{j+1}$$

$$= -h\frac{D_{j-1}}{A_{j-1}} + h\frac{D_{j+1}}{A_{j+1}} \quad (7)$$

If boundary values of u and m are known, Eqs. (6) and (7) constitute a 2×2 block-tridiagonal system of linear equations. However, the boundary values of both u and m cannot be given, since that would overspecify the continuum problem. The difficulty in most high-order accuracy methods is to find a consistently high-order boundary formulation. If conventional one-sided differences are used to evaluate m at boundaries, the bandwidth of the system is greatly increased,

leading to inefficiency and increasing the accumulation of roundoff error. Instead, we utilize Eq. (19b) of Ref. 3, written at the left-hand boundary $j=1$ as:

$$u_2 - u_1 - (h/2)(m_2 + m_1) + (h^2/12)(M_2 - M_1) = G_1 \quad (8)$$

where

$$G_1 = \frac{h^5}{720}(u^v)_1 + \frac{h^6}{1440}(u^{vi})_1 \quad (9)$$

where (u^v) and (u^{vi}) are the fifth and sixth derivatives in r. Again using Eq. (3) to eliminate M from Eq. (8), we obtain

$$\left(\frac{h^2}{12}\frac{B_1}{A_1} - \frac{h}{2}\right)m_1 - \left(\frac{h}{2} + \frac{h^2}{12}\frac{B_2}{A_2}\right)m_2 + \left(\frac{h^2}{12}\frac{C_1}{A_1} - 1\right)u_1$$

$$- \left(\frac{h^2}{12}\frac{C_2}{A_2} - 1\right)u_2 = \frac{h^2}{12}\left(\frac{D_1}{A_1} - \frac{D_2}{A_2}\right) + G_1 \quad (10)$$

The analogous equation at the right boundary $j=JL$ is:

$$\left(\frac{h}{2} - \frac{h^2}{12}\frac{B_{JL-1}}{A_{JL-1}}\right)m_{JL-1} + \left(\frac{h}{2} + \frac{h^2}{12}\frac{B_{JL}}{A_{JL}}\right)m_{JL}$$

$$- \left(\frac{h^2}{12}\frac{C_{JL-1}}{A_{JL-1}} - 1\right)u_{JL-1} + \left(\frac{h^2}{12}\frac{C_{JL}}{A_{JL}} - 1\right)u_{JL}$$

$$= \frac{h^2}{12}\left(\frac{D_{JL}}{A_{JL}} - \frac{D_{JL-1}}{A_{JL-1}}\right) + G_{JL} \quad (11)$$

where

$$G_{JL} = -\frac{h^5}{720}(u^v)_{JL} + \frac{h^6}{1440}(u^{vi})_{JL} \quad (12)$$

If G_1 and G_{JL} are known, the system of Eqs. (6, 7, 10, and 11) constitute a 2×2 block-tridiagonal system which is an $O(h^6)$ approximation to Eq. (2). Since G_1 and G_{JL} are actually unknown, we obtain the solution by deferred corrections as follows.

Using the bitridiagonal algorithm of von Rosenberg[6] or another banded matrix solver, we obtain a first solution to the system with $G_1 = G_{JL} = 0$. From Eqs. (8) and (9), it is clear that this gives an initial solution u^0 which is formally $O(h^4)$ accurate. Based on u^0, we evaluate G_1 and G_{JL} using conventional one-sided 7-point differences[7] for u^v and u^{vi} as follows:

$$h^5(u^v)_1 = (-21 u_1 + 120 u_2 - 285 u_3 + 360 u_4 - 255 u_5$$

$$+ 96 u_6 - 15 u_7)/6 \quad (13a)$$

$$h^5(u^v)_{JL} = (15 u_{JL-6} - 96 u_{JL-5} + 255 u_{JL-4} - 360 u_{JL-3}$$

$$+ 285 u_{JL-2} - 120 u_{JL-1} + 21 u_{JL})/6 \quad (13b)$$

$$h^6(u^{vi})_1 = u_1 - 6 u_2 + 15 u_3 - 20 u_4 + 15 u_5 - 6 u_6 + u_7 \quad (13c)$$

$$h^6(u^{vi})_{JL} = u_{JL-6} - 6 u_{JL-5} + 15 u_{JL-4} - 20 u_{JL-3}$$

$$+ 15 u_{JL-2} - 6 u_{JL-1} + u_{JL} \quad (13d)$$

These values are used in Eqs. (10) and (11) and the solution is repeated. The process can be continued until convergence. However, since u^v from Eqs. (13a and b) are already $O(h^2)$ accurate and u^{vi} from Eqs. (13c and d) are already $O(h)$ accurate, it is clear from Eqs. (9-12) that the first corrected solution to u is already $O(h^6)$ accurate, so the method with a single correction can be termed "direct."

Although the requirement for a bitridiagonal solution is more expensive of computer time than the scalar tridiagonal solution required for $O(h^2)$ accuracy, and although the present method requires a correction step, the operation count is the same *order* as Hockney's method.[1] That is, the operation count for the FFTs with N nodes in θ is $O(N \ell n N)$ if $N-1=2^p$, and the two-dimensional solution for $N \times N$ nodes is then obtained in $O(N^2 \ell n N)$ operations. It should be noted that the deferred correction step is done within a solution for a Fourier component, so that the FFT operations are not repeated for the corrections. Also, most of the coefficients of the 2×2 system need not be re-evaluated for the correction step(s).

If more than one deferred correction step is used, roundoff error may cause the iterations to diverge unless the corrections are underrelaxed. For one correction, we use

$$G_1^1, G_{JL}^1 = \text{function } (u^0) \quad (14a)$$

but for more correction steps, we use

$$G_1^n, G_{JL}^n = \text{function } (\tfrac{1}{2}u^{n-1} + \tfrac{1}{2}u^{n-2}) \quad (14b)$$

Numerical Verification

The code incorporating this method has been tested on a variety of two-dimensional, as well as one-dimensional, problems, and is designed to handle arbitrary user-specified coefficients a-d in Eq. (1). Representative results are presented here for the simple equation:

$$\psi_{rr} + \psi_{\theta\theta} = d, \ 0 \le r \le 2\pi, 0 \le \theta \le 2\pi \quad (15)$$

for two cases,

$$d_a = -\tfrac{1}{2}\sin(r)\sin(\theta) \quad (16a)$$

which gives the solution $\psi(r,\theta) = \sin(r)\sin(\theta)$, and

$$d_b = -\tfrac{1}{2}\cos(r)\cos(\theta) \quad (16b)$$

which gives the solution $\psi(r,\theta) = \cos(r)\cos(\theta)$. These two test cases give different behavior because the solution to Eq. (16a) gives $u^{vi} = 0$ at boundaries, while Eq. (16b) gives $u^v = u^{vii} = 0$ at boundaries. These are useful for isolating the effect of the deferred corrections for G_1 and G_{JL}.

The numerical tests were performed on a time-shared CDC 6600 NOS system. The computer has 14^+ significant decimal figures of word length. With only mesh sizes of the form $N-1 = 2^p$ considered, the largest problem which fits into the 34,752 word central memory allocation is a 65×65 node problem.

The truncation-error convergence for the test cases is shown in Table 1. For both test problems, Eqs. (16a) and (16b), the maximum error in the solution E is presented for cases of no deferred correction ($Lc=0$), one deferred correction ($Lc=1$) and $Lc=10$, for meshes ranging from 9×9 to 65×65 ($p=3$-6). The maximum error E is seen to decrease very rapidly, as expected. A more precise evaluation is obtained by inspection of the tabulated values of E/h^4, E/h^5, etc. In the idealized cases of no roundoff error and $h \to 0$, the value of E/h^s will become constant for an s-order accurate method as h decreases.

Part a of Table 1 shows the performance for the uncorrected method. Although formally only $O(h^4)$ accurate, it is seen that the actual performance is closer to $O(h^5)$, since the value of E/h^5 increases only from 3.04×10^{-2} to 3.86×10^{-2} as p varies from 4 to 6, for the more difficult test case Eq. (16a). Part b of Table 1 shows the performance for the method with a single deferred correction. Formally, the method is $O(h^6)$ accurate, and this is indicated by the values of E/h^6, which for the worse case Eq. (16a) are nearly constant, increasing only 21% as p increases from 4 to 6. However, the

Table 1 Performance of the direct $0(h^6)$ elliptic solver

Part	Eq.	p	Mesh, $N \times N$	Lc	E	E/h^4	E/h^5	E/h^6	CPU time s	CPU time/N^2, ms
a	(16a)	3	9×9	0	0.637-03	0.167-02	0.213-02		0.037	0.457
		4	17×17	0	0.284-04	0.119-02	0.304-02		0.135	0.467
		5	33×33	0	0.104-05	0.702-03	0.357-02		0.519	0.477
		6	65×65	0	0.352-07	0.379-03	0.386-02		2.063	0.488
	(16b)	3	9×9	0	0.283-03	0.743-03	0.947-03	0.121-02	same as Eq. (16a)	
		4	17×17	0	0.581-05	0.244-03	0.622-03	0.159-02	same as Eq. (16a)	
		5	33×33	0	0.104-06	0.702-04	0.356-03	0.182-02	same as Eq. (16a)	
		6	65×65	0	0.174-08	0.188-04	0.191-03	0.195-02	same as Eq. (16a)	
b	(16a)	3	9×9	1	0.188-03			0.799-03	0.050	0.617
		4	17×17	1	0.858-05			0.234-02	0.182	0.630
		5	33×33	1	0.147-06			0.257-02	0.694	0.637
		6	65×65	1	0.253-08			0.282-02	2.737	0.648
	(16b)	3	9×9	1	0.953-03			0.406-02	same as Eq. (16a)	
		4	17×17	1	0.938-05			0.256-02	same as Eq. (16a)	
		3	33×33	1	0.518-07			0.904-03	same as Eq. (16a)	
		6	65×65	1	0.280-09			0.312-03	same as Eq. (16a)	
c	(16a)	3	9×9	10	0.205-03			0.873-03	0.174	2.148
		4	17×17	10	0.699-05			0.190-02	0.609	2.107
		5	33×33	10	0.950-07			0.166-02	2.276	2.090
		6	65×65	10	0.874-09			0.976-03	8.805	2.084
	(16b)	3	9×9	10	0.897-03			0.382-02	same as Eq. (16a)	
		4	17×17	10	0.872-05			0.238-02	same as Eq. (16a)	
		5	33×33	10	0.450-07			0.785-03	same as Eq. (16a)	
		6	65×65	10	0.191-09			0.213-03	same as Eq. (16a)	

Note: Lc = number of deferred corrections; E = error of the solution; h = mesh spacing in r and θ, $= 2\pi/(N-1)$. CPU times are for a FORTRAN IV program run on a time-shared CDC 6600 NOS system with Level 2 optimization. The notation 0.637-03 means 0.637×10^{-3}, etc.

single deferred correction is not as accurate as a fully iterated solution, such as that shown in part c of Table 1 for ten corrections. This shows E/h^6 decreasing significantly for both test cases, indicating convergence at better than $0(h^6)$.

Also shown in Table 1 are timing tests, obtained on a CDC 6600 NOS system using FORTRAN IV with Level 2 optimization. The CPU time shown is the average of at least three runs, and does not include I/0. An optimal method will have an operation count for an $N \times N$ problem proportional to N^2, so that the value of (CPU time)/N^2 would be a constant. The present method has a suboptimal operation count proportional to $N^2 \ell nN$, so we would expect (CPU time)/N^2 to increase with N. However, there are evidently enough overhead operations in the program to mask this behavior over the range of N tested, since (CPU time)/N^2 increases only slightly with N in Table 1. (If we consider only the unknowns at interior nodes, (CPU time)/$(N-2)^2$ actually decreases.) The time difference between the $0(h^4)$ solution with no correction and the $0(h^6)$ solution with one correction is only about 35% for $N=9$ and decreases slightly for increasing N.

Discussion

A direct method for obtaining $0(h^6)$ accurate solutions of a class of separable elliptic equations with periodic boundary conditions in one direction has been presented and verified. The method as presently coded uses the constant-mesh-spacing version of Hermite 6 discretization of Rubin and Khosla.[3] However, it should be noted that the Hermite 6 equations as given in Ref. 3 are actually more general, allowing for variable h with $0(h^5)$ accuracy. Shear layer resolution can be achieved in the present code, since the form of Eq. (1) allows for arbitrary coordinate stretching in r; however, the variable-h equations would be useful for self-adaptive mesh solutions. Another advantage of the compact formulas such as Hermite 6 is that both the function and its first derivative (u and m in Eqs. 3-11) appear explicitly. It is thus very easy to set accurate boundary conditions on the gradient, which is more difficult in conventional high-order differencing. Further, the numerical oscillations associated with high-cell Reynolds numbers are mitigated by the use of compact forms.[3] The advantages of high-order methods are

well known, provided that high accuracy is indeed required for the problem at hand. Good candidates for sixth-order methods are problems which involve fluid stability calculations. The direct solver described previously is currently being used in studies of time-dependent vortex shedding from oscillating cylinders.

On a less optimistic consideration, we note: 1) it will be difficult to actually realize $0(h^6)$ accuracy in real aerodynamics problems with ambiguities in far-field computational boundary conditions; and 2) high formal accuracy does not solve resolution problems associated with the thin shear layers of high Reynolds number problems (e.g., see Ref. 8).

References

[1] Hockney, R.W., "A Fast Direct Solution of Poisson's Equation Using Fourier Analysis," *Journal of the Association for Computing Machinery,* Vol. 12, Jan. 1965, pp. 95-113.

[2] Shaanan, S. and Ferziger, J.H., *A Direct Method of Solution of Poisson Equation Accurate to Fourth Order,* Rept. TF-3, Thermosciences Division, Dept. of Mechanical Engineering, Stanford University, Calif., Aug. 1974.

[3] Rubin, S.G. and Khosla, P.K., "Polynomial Interpolation Methods for Viscous Flow Calculations," *Journal of Computational Physics,* Vol. 24, No. 3, July 1977, pp. 217-244.

[4] Schelkunoff, S.A, *Applied Mathematics for Engineers and Scientists,* 2nd ed., D. Van Nostrand, Princeton, N.J., 1965, Chap. VIII.

[5] Mehta, U.B., "Dynamic Stall of an Oscillating Airfoil," Paper 23, AGARD Fluid Dynamics Panel Symposium on Unsteady Aerodynamics, Ottawa, Sept. 1977.

[6] von Rosenberg, D.U., *Methods for the Numerical Solution of Partial Differential Equations,* American Elsevier, N.Y., 1969, Appendix C.

[7] Southwell, R.V., *Relaxation Methods in Theoretical Physics,* Oxford at the Clarendon Press, Oxford, England, 1946, Table V.

[8] Roache, P.J., "A Pseudo-Spectral FFT Technique for Non-Periodic Problems," *Journal of Computational Physics,* Vol. 27, No. 2, May 1978, pp. 204-220.