Rapid Elliptic Solvers

(the final instalment?)

by

Edward Detyna

Numerical Analysis Report 6/80

Invited paper. The Institute of Mathematics and its Applications

Conference on "Sparse Matrices and their uses".

Reading, 9-11 July, 1980.

The University of Reading,
Department of Mathematics,
Whiteknights,
Reading. RG6 2AX.

Introduction

The aim of this paper is to provide a review of the Rapid Elliptic Solvers (RES) for the solution of elliptic partial differential equations (e.p.d.e.) in one, two or three dimensions. [We shall use the expression "e.p.d.e." for one dimensional equations even though it is not correct.] The review is neither complete nor comprehensive but, hopefully, indicates the developments in this area in the last 15 years.

First of all the term "rapid" in RES requires a definition. With the aid of the finite element method (FEM) or finite difference method (FDM), the e.p.d.e. can be approximated by a set of linear equations $Ax = b$, where A is an $N \times N$ matrix (usually very sparse) and x and b are vectors length N of unknowns and RHS's respectively. To classify as a RES the algorithm must be able to provide the solution $x = A^{-1}b$ in αN floating point operations* (f.p.o.) <u>and</u> use no more than N to $2N$ memory storage locations, where α is a constant (or slowly varying function of N) of the order of 10 . This last requirement excludes various sparse matrix techniques which often result in algorithms with $\alpha \sim N^{\frac{1}{2}}$ [1], [2], [3] .

It is clear that with such stringent conditions imposed on the performance of the algorithms not all e.p.d.e.'s on arbitrary domains can be solved with RES. In particular, there is no RES capable of solving general non-separable e.p.d.e. in 2 or 3 dimensions. What is more the author feels that no such algorithm is possible.

The most general e.p.d.e. for which RES exists, at least on special domains, is the general separable e.p.d.e.:

* the term f.p.o. means here a single floating point operation of the type +, -, *, / .

$$\sum_{\alpha=1}^{d} \left(a_\alpha \partial_\alpha^2 + b_\alpha \partial_\alpha + c_\alpha\right) u = f \, , \tag{1.1}$$

where a_α, b_α and c_α are arbitrary functions of one variable only $\left(a_1(x^1), a_2(x^2)\right.$ and so on$\left.\right)$ $\partial_\alpha \equiv \partial/\partial x^\alpha$ and d is the number of dimensions. The a_α's have to be all of the same sign for equation (1.1) to be elliptic. For certain boundary conditions (b.c.) the function f on the RHS may have to satisfy compatibility conditions. The equation (1.1) is to be satisfied on domain D with appropriate b.c. on ∂D (Dirichlet, Neuman, periodic or mixture of the three).

The equation (1.1) can be symmetrized by transforming out the terms $b_\alpha \partial_\alpha u$ with $u' = ug$, where $g = \prod_{a=1}^{d} \exp(\frac{1}{2} \int b_\alpha/a_\alpha \, dx^\alpha)$. Therefore without a loss of generality we shall assume $b_\alpha = 0$.

In the case of isotropic diffusion, $a_1(x^1) = a_2(x^2) = a_3(x^3) = \text{const}$, we have a separable Helmholtz equation:

$$[\nabla^2 + \sum_{\alpha=}^{d} c_\alpha(x^\alpha)] \, u = f \, . \tag{1.2}$$

Finally, when all the coefficients are constant we have constant coefficient Helmholtz equation

$$(\nabla^2 + c) \, u = f \tag{1.3}$$

and if $c = 0$ it becomes the Poisson equation. In the following sections we shall discuss methods of solving these equations depending on the dimension d of the domain. We shall see that for $d = 1$ there is a number of RES's for the most general equation (1.1) while for $d = 2$ and 3 some of these methods are no longer RES even for the simplest Poisson equation.

2. One dimensional equation.

The equation (1.1) in one dimension may be reduced to

$$\left(d^2/dx^2 + c(x)\right) u(x) = f(x) \tag{2.1}$$

on a domain $0 \le x \le 1$ with b.c. at $x = 0$ and $x = 1$. Discretizing it on a uniform mesh with h^2 accuracy gives

$$U_{i-1} + C_i U_i + U_{i+1} = F_i , \tag{2.2}$$

where $C_i = -2 + h^2 c(ih)$, $F_i = h^2 f(ih)$, $U_i = U(ih)$, $h = 1/n$ and $i = 2, 3, \ldots, n-1$. For $i = 0$ and $i = n$ the equation (2.2) has to be suplemented by the b.c. We shall consider homogeneous Dirichlet b.c. $U_o = U_n = 0$ unless otherwise stated.

The equation (2.2) is the simplest approximation to (2.1), we could use higher order formulae which would result in, say, 5-diagonal matrix in equation (2.2). This does not increase the work for some algorithms as will be indicated later.

We shall now review methods of solving equation (2.2) bearing in mind possible applicability to higher dimensions.

2.1 Fast Fourier Transform (FFT).

The functions $\sin(\pi ki/n)$ are the eigenfunctions of equation (2.2) provided that the coefficient C_i is a constant $C_i = C$. Taking FFT of equation (2.2) gives $\hat{U}_k = G_k \hat{F}_k$, where $G_k = (C + 2\cos \pi k/n)^{-1}$ is the influence function. We note that the higher order schemes would merely change this function without increasing the op. count.

Originally the FFT algorithm existed for $n = 2^{\ell}$ only [4] but later [5] the FFT was extended to any value of n. The op. count of FFT is $\alpha n \log_2 n$, where α in the latest methods is about 2-2.5. Hence the algorithm has the op. count of $1 + 5 \log_2 n$ f.p.o. per mesh point. It is stable for all values of C provided that the influence function G_n is not singular.

2.2 LU decomposition.

If we write equation (2.2) in the matrix form $Au = F$, the matrix A can be decomposed into $A = LU$, where

$$L = \begin{bmatrix} \lambda_1^{-1} & & & \\ 1 & \lambda_2^{-1} & & \\ & \ddots & \ddots & \\ & & 1 & \lambda_n^{-1} \end{bmatrix}, \qquad U = \begin{bmatrix} 1 & \lambda_1 & & \\ & 1 & \lambda_2 & \\ & & \ddots & \ddots \\ & & & 1 \end{bmatrix}, \qquad (2.3)$$

where $\lambda_1 = 1/C_1$, $\lambda_i = 1/(C_i - \lambda_{i-1})$ for $i = 2, 3, \ldots, n$.

The solution is then obtained in two stages

1) $v = L^{-1} F \; \left(v_1 = F_1 ,\;\; v_i = (F_i - v_{i-1})\lambda_i \quad \text{for} \quad i = 2, 3, \ldots n\right)$

and then

2) $u = U^{-1} v \; (u_n = v_n ,\; u_i = v_i - \lambda_i u_{i+1} \quad \text{for} \quad i = n-1, n-2, \ldots, 1)$.

The op. count is $4n$ plus $2n$ for precalculation of λ's regardless of whether C_i is constant or not.

There is one exception [6] to this - if we retain the Dirichlet b.c. at $x = 1$ but impose mixed b.c. at $x = 0$, namely $\mu u_{-1} + u_o = 0$, where $\mu = \frac{C}{2} + \left(\frac{C^2}{4} - 1\right)^{\frac{1}{2}}$ and $C = C_i$ is a constant, then the matrix A takes takes the form

$$A = \begin{bmatrix} \mu & 1 & & & & \\ 1 & C & 1 & & & \\ & 1 & C & 1 & & \\ & & \cdot & \cdot & \cdot & \\ & & & \cdot & \cdot & \cdot \\ & & & & 1 & C \end{bmatrix}$$

which has a particularly simple LU decomposition (2.3) with all $\lambda_i = \mu^{-1}$. We note that for the Poisson equation $C = -2$, hence $\mu = -1$ and the b.c. at $x = 0$ is simply the Neuman condition, but for the Helmholtz equation $C \neq -2$, the mixed b.c. is less obvious. The memory required in this case is one location per mesh point whereas in the general case an additional storage is needed for λ's.

2.3 Cyclic reductions (CR).

We shall consider equation (2.2) with $C_i = C$ although the CR algorithm for variable C_i is only slightly more complicated. By taking an appropriate linear combination of 3 consecutive equations ($i-1$, i and $i+1$) we get

$$U_{i-2} + C^{(1)} U_i + U_{i+2} = F_i^{(1)} \equiv F_{i-1} - C F_i + F_{i+1}, \qquad (2.4)$$

where $C^{(1)} = 2 - C^2$. Thus only U_i's at every second point are connected. This process can be repeated $\ell-1$ times for $n = 2\ell$ resulting in a single equation, connecting boundary values U_o and U_n with the centre point $U_{n/2}$, which can be solved. Then using in the reverse order the reduced equations the remaining U's can be calculated. We note that for Poisson equation, $C = -2$, hence $C^{(\ell)} = -2$ for all levels ℓ . On the other hand, the most unstable harmonic will grow by a factor of 4 at each level of reduced densities $F^{(\ell)}$. Therefore for $C = -2$ the algorithm is unstable with error ϵ growing as $\epsilon \sim 4^{\ell} = n^2$. For $C \neq -2$, the fastest harmonic grows in the reduced density as $\prod_{p=0}^{\ell-1} (2 + |C^{(p)}|)$, whereas the solution is found by dividing it by $C^{(\ell)}$. Since for $C << -2$ the constants $C^{(\ell)} \approx C^{2^{\ell}}$ the algorithm is stable but for $|C + 2| << 1$ the error growth may be appreciable and needs evaluating especially for large n .

The CR method has been extended [7] to cope with every value of n .

The op. count is $6n$ with additional storage of $\log_2 n$ for $C^{(\ell)}$'s .

In the variable coefficient case the op. count grows to $18n$ with additional storage of $2n$.

2.4 Marching algorithms.

Marching methods are based on the fact that if solution is known at two consecutive points then equation (2.2) may be used as a recursive formula

$$U_{i+1} = F_i - C_i U_i - U_{i-1} \quad . \tag{2.5}$$

Assuming a value for U_1 with given U_o we can calculate $u^*_2, u^*_3, \ldots, u^*_n$. The calculated u^*_n differs from the given boundary value U_n but this difference can be used to find the correct solution U_1 and then repeat the march to find the corrected solution. In order to study the stability let us solve equation (2.5) for $C_i = C$ and $F_i = 0$:

$$U_i = A\mu_+^{\ i} + B\mu_-^{\ i} \quad , \tag{2.6}$$

where $\mu_\pm = -\frac{C}{2} \pm \sqrt{\frac{C^2}{4} - 1}$ and A and B are constants. For $C < 0$, μ_+ is the larger root, hence the error grows as $\mu_+^{\ i}$. After marching over p points we loose D decimal places of accuracy where $D = p \log_{10} \mu_+$. With the exception of the Poisson equation, where $C = -2$ and $\mu_+ = \mu_- = 1$ and the march is stable, the procedure for $C \neq -2$ is exponentially unstable. One strategy (Lorenz march [8]) is to do partial marches of length $p = D/\log_{10} \mu_+$, where D is the number of digits we are prepared to loose, over n/p separate regions. The resulting n/p equations are then solved to get correct starting values. This could be an efficient method providied that μ_+ is not too large.

The op. count is $6n + O(D^{-1})$.

2.5 Conclusions.

It seems that in the one dimensional case all the methods are rapid. The various properties of these methods are gathered together in Table I and it is clear that LU decomposition is by far the best method.

TABLE I

One dimensional problem.

Method	Equation	Op. Count in f.p.o.	Additional Storage	Stability
FFT**	Const. Coeff. Helmholtz	$n(1 + 5 \log_2 n)$	n	stable
LU decomp.	General*	$4n + 2n$	n	stable
CR	Poisson	$6n$	-	unstable
	Helmholtz			
	a) const.	$6n$	$\log_2 n$	stable
	b) variable	$18n$	$2n$	stable
March	Poisson	$6n$	-	stable
	Helmholtz*	$6n + O(D^{-1})$	$O(D^{-1})$	unstable

* see text

** can be used for higher order discretization methods with no additional cost.

3. Two-dimensional equation.

The RES for two-dimensional equation (1.1) exist only for regular domains either rectangle $(x^1 = x \; , \; x^2 = y)$ or circle $(x^1 = r \; , \; x^2 = \phi)$. The arbitrarily shaped boundary can be solved by embedding the region of solution in a regular domain and applying RES to the regular domain. The required b.c. are imposed in the interior of the domain by a correction technique known as Capacity Matrix method. This method requires numerical inversion of $m \times m$ matrix C, where m is the number of internal boundary points [9, 10]. Therefore for all but almost regular boundaries (small m) it cannot be classified as RES.

Since equations described on a circle and a rectangle are similar we shall continue this section considering rectangular b.c. only.

Discretizing equation (1.1) on a rectangular mesh one obtains the following interaction molecule at the mesh point (i, j)

$$\left\{\begin{array}{ccc} & p_j & \\ q_i & r_i + s_j & q_i \\ & p_j & \end{array}\right\} U_{ij} = F_{ij} \; . \qquad (3.1)$$

The above is the shorthand for the equation.

$$q_i(U_{i-1,j} + U_{i+1,j}) + p_j \, (U_{i,j-1} + U_{i,j+1}) + (r_i + s_j)U_{ij} = F_{ij} \; ,$$

where $q_i = a_1(ih)$, $p_j = a_2(jh)$, $r_i = c_1(ih)h^2 - 2q_i$,

$s_j = C_2(jh)h^2 - 2p_j$, $F_{ij} = h^2 f(ih, jh)$; $i,j = 1, 2, \ldots, n-1$.

We have chosen equal spacing on the square domain since the extensions to the rectangle or different spacings in x- and y-directions are trivial.

3.1 Fourier Transform.

The Fourier analysis can be applied in, say, x direction, only if $a_1(x)$ and $c_1(x)$ are constants which implies $q_i = 1$, $r_i = -2$. In that case the Fourier transform in i-variable results in a set of n one-dimensional equations for each harmonic k:

$$p_j \hat{U}_{k,j-1} + (s_j - 2)\hat{U}_{k,j} + p_j \hat{U}_{k,j+1} = \hat{F}_{k,j} \qquad (3.2)$$

which can be solved by any method of the previous section. Assuming the op. count for solving (3.2) to be $6n$ the total is $(\cdot 5\ n \log_2 n + 6\ n)n$. Since this algorithm is a "product" of two one-dimensional algorithms all the remarks of the previous section apply.

3.2 LU decomposition.

Unlike the one-dimensional case, where tri-diagonal matrix A factorized into bi-diagonals L and U, the five diagonal matrix A describing equation (3.1) does not factorize into tri-diagonal L and U. This can only be done approximately [11] $LU = A + E$, where E represents the two additional diagonals of "in-fill". Hence the equation $Au = F$ can be solved iteratively $LUu^{(\ell+1)} = F + Eu^{(\ell)}$. The efficiency of this algorithm depends on choosing such LU decomposition as to minimize E. This is not RES algorithm and since it does not depend strongly on structure of matrix A it is best used for general e.p.d.e's.

3.3 Cyclic Reductions.

3.3.1 Global Cyclic Reductions.

The equation (3.1) may be written in quisi-one-dimensional form

$$Q_i \underline{v}_{i-1} + C_i \underline{v}_i + Q_i \underline{v}_{i+1} = \underline{g}_i \qquad (3.3)$$

where Q_i and C_i are matrices $Q_i = I\,q_i$, $C_i = r_i I + T$, T is tri-diagonal $[p_j, s_j, p_j]$ and $\underline{v}_i$ and $\underline{\underline{g}}_i$ are n-vectors made up of U_{ij} and F_{ij} . Let us first study a constant coefficient Helmholtz equation ($Q_i = I$, $C_i = C$, constant diagonal). In this case we can carry on as in section 2.3 cyclicly reducing equation (3.3). Of course, this time the C's at higher levels $C^{(1)}$, $C^{(2)}$, ... are matrices, therefore the solution at level ℓ requires inverting $C^{(\ell)}$ matrix which is $(1 + 2^{\ell+1})$-diagonal. It was shown [12] that $C^{(\ell)}$ can be factorized into tri-diagonal matrices and solved as one-dimensional cases. Because the ratio of the maximum to minimum eigenvalues of matrix $C^{(\ell)}$ grows exponentially with ℓ the algorithm is unstable. Buneman [13] stabilized it by rephrasing the method of calculating reduced densities on the RHS of equation (3.3). The resulting algorithm is stable with op. count $n^2(6 \log_2 n + \alpha)$ where $\alpha = 1$ or 3.5 depending on whether an additional memory of $\frac{1}{2} n^2$ is used or not.

The algorithm has been extended to cope with mesh sizes of arbitrary n [7] .

The general equation (3.3) with variable coefficients has also been solved [14] but the factorization of $C^{(\ell)}$ matrices is no longer a simple process and requires numerical calculation of coefficients of the tri-diagonal matrices. This can be done in the preprocessing stage and then the solution is found as in constant coefficient case. The op. count for the solution phase alone is $(32 \log_2 n - 88)n^2$ while the preprocessing phase takes usually twice as long.

3.3.1.1 FACR algorithm.

The equation (3.3) and the reduced equation have the same form, therefore after r reductions the $n/2^r$ equations can be Fourier analized [15] as in section 3.2, provided that the coefficients are constant. Since the FFT algorithm has large op. count for large n while GCR is efficient at low levels ℓ of cyclic reductions one can find the optimal ℓ algorithm, the so called FACR(ℓ). Its optimal op. count is $4n^2 \log_2(\log_2 n)$, that is to say 8-12 operations per mesh point. In practice this optimal value is never reached and the realistic op. count is 20-24 operations per mesh point [16] . This algorithm is the most efficient combination of FFT, CR and LU decomposition.

3.3.2 Point Cyclic Reductions.

The two-dimensional e.p.d.e. was descretized to obtain equation (3.1) using one h^2-accurate formula, but for the constant coefficient Helmholtz equation there is another h^2-accurate formula.

Let us write both interaction molecules in this case

$$\left\{\begin{array}{ccc} & 1 & \\ 1 & C^{+} & 1 \\ & 1 & \end{array}\right\} \qquad \left\{\begin{array}{ccc} 1 & & 1 \\ & C^{\times} & \\ 1 & & 1 \end{array}\right\} \qquad (3.4)$$

where C^{+} and $C^{\times}$ are constants. In order to simplify notation let us introduce S^{+} and $S^{\times}$ operators

$$S^{+} = \left\{\begin{array}{ccc} & 1 & \\ 1 & & 1 \\ & 1 & \end{array}\right\} \text{ and } S^{\times} = \left\{\begin{array}{ccc} 1 & & 1 \\ & & \\ 1 & & 1 \end{array}\right\} .$$

We can now write the two types of the descretized equations as

$$(S^{+} + C^{+})u_{ij} = F_{ij}^{+} \quad \text{and} \quad (S^{\times} + C^{\times})u_{ij} = F_{ij}^{\times} \ , \qquad (3.5)$$

where F^{+} and $F^{\times}$ are chosen in such a way as to minimise the difference between the two solutions of equations (3.5). If we note [17] that $(S^{+})^2 = S^{2^+} + 2S^{\times} + 4$ and $(S^{\times})^2 = S^{2^\times} + 2S^{2^+} + 4$, where S^{2^+} and $S^{2^\times}$ stand for interaction molecules connecting points on the mesh twice removed from the centre, the two-dimensional cyclic reductions are evident. The first reduced densities will be

$$F^{2^+} = (S^{+} - C^{+})F^{+} - 2F^{\times} \ ,$$

$$F^{2^\times} = (S^{\times} - C^{\times})F^{\times} - 2F^{2^+} \qquad (3.6)$$

and the higher level reductions follow from the form of the reduced equations

$$(S^{2^+} + C^{2^+})u = F^{2^+} \ ,$$

$$(S^{2^\times} + C^{2^\times})u = F^{2^\times} \ ,$$

where $C^{2^+} = 4 - 2C^{\times} - (C^{+})^2$, $C^{2^\times} = 4 - 2C^{2^+} - (C^{\times})^2$.

This is an "n^2-algorithm" and the op. count depends on the choice of F_{ij}^{+} and $F_{ij}^{\times}$. If we choose the standard formulae: the "5-point" and "rotated 5-point", i.e. $F_{ij}^{+} = F_{ij}$ and $F_{ij}^{\times} = (1 + \frac{1}{4}S^{+})F_{ij}$ the op. count is $9.5\ n^2$.

The stability can be studied by noting that in the Poisson case, $C^{+} = C^{\times} = -4$, the fastest growing harmonic in the reduced densities (3.6) grows by a factor of 16 at each level of PCR. Therefore after

$\ell-1$ point reductions, where $\ell = \log_2 n$ the error is $\epsilon \sim (n^2)^2$. This result is identical to that for one-dimensional CR where error growth was proportional to the square of the number of mesh points. But the algorithm is stabilized for $C^+ < -4$ since then the solution grows faster than the reduced densities. The same applies to the error due to the difference in the two descretizations, equation (3.4).

3.4 Marching algorithms.

The two dimensional marching method is identical in principle to the one described in 2.4, the difference being in that we now march over whole lines:

$$u_{i+1,j} = \left(F_{ij} - P_j(u_{i,j+1} + u_{i,j-1}) - (r_i + s_j)u_{ij}\right)/q_j - u_{i-1,j} \cdot$$

This also means that the instability is growing faster. For the Poisson equation the error now grows as $\epsilon \sim (3 + \sqrt{8})^p \cong 5.8^p$ for p lines marched. In practice, it means that we loose D digits of accuracy for p lines marched, where $D = 0.77p$. If we are prepared to loose D digits in the result we have to do $1.3n/D$ partial marches and solve the equations over $1.3n/D$ lines to get corrected starting values. Therefore the op. count is of the order of $(10 + \alpha/D)n^2$ and $(14 + \alpha/D)n^2$ for constant and variable coefficient cases respectively, where α depends on the method of solution for the correct starting values for partial marches. If we use FFT for the constant coefficient case then $\alpha \cong 10 + 4 \log_2 n$. In the variable coefficient case the solution has to be obtained iteratively, which means $\alpha \sim n$ in practice.

In any event this algorithm is only suitable for the computers with very long word length since only then we can afford a loss of reasonably large number D of decimal digits of accuracy.

3.5 Conclusions.

There is only one RES capable of solving general separable e.p.d.e in two dimensions on a rectangle [14] . On the other hand there are several algorithms for the constant coefficient Helmholtz equation. The best two seem to be FACR and PCR. Both of them are in practice "n^2 -algorithms", the former being twice as slow as the latter but also far more accurate.

TABLE II

Two dimensional problem.

Method	Equation	Op. count	Additional storage	Stability
FFT + LU	Const. coeff. in one variable	$(5 \log_2 n + 6)n^2$	n	stable
GCR	general separable const. coeff.	$(32 \log_2 n - 88)n^2$ $(6 \log_2 n + 3.5)n^2$	$3\ n^2$ -	stable stable
FACR	const. coeff.	$(20\text{-}24)n^2$	n	stable
PCR	const. coeff. Helmholtz Poisson	$9.5\ n^2$ $9.5\ n^2$	$\frac{1}{8}\ n^2$ $\frac{1}{8}\ n^2$	stable unstable
March*	general const. coeff.	$(14 + \alpha/D)n^2$ $(10 + \alpha/D)n^2$	- $O(D^{-1})$	unstable unstable

* see text

4. Three dimensional equation.

Descretising equation (1.1) on a regular grid with uniform spacing gives an equation which written in "two-dimensional form" is

$$\left\{\begin{array}{ccc} & p_j & \\ q_i, & r_i + s_j + t_k, & q_i \\ & p_j & \end{array}\right\} U_{ijk} + z_k(U_{i,j,k-1} + U_{i,j,k+1}) = F_{ijk},$$

where the interaction molecule acts on the first pair of indices i and j only. The notation here is the same as in equation (3.1) and $z_k = a_3(kh)$, $t_k = c_3(kh)h^2 - 2z_k$.

There does not seem to be a RES algorithm capable of solving this general equation but if the coefficients are independent of one variable, say k, $z_k = 1$, $t_k = -2$, then we can Fourier analyse it in that coordinate using FFT. This reduces the problem to n two-dimensional equations each of which can be solved as in section 3.

In particular, we note that in the constant coefficient case the PCR method is stable for both Helmholtz and Poisson equation since in the later case the FT in the z-direction results in 2-D Helmholtz equations. This should then give an efficient Poisson solver with op. count $(9.5 + 5 \log_2 n)n^3$.

On the other hand the marching algorithm is even more unstable loosing one decimal digit of accuracy per plane marched. (The error grows as 9.9^p, where p is the number of planes.)

To conclude it is instructive to compare the op. counts per mesh point for the best RES algorithms in different dimensions for the constant coefficient

Helmholtz equation. In other words, we would like to compare the op. counts per mesh point for the solution of $Au = f$ equation, where A is the finite difference matrix approximating e.p.d.e. To make the comparison meaningful we should calculate the op. count of the discretization, or in other words, the op. count of the product $A.u$ and then take the ratio of the two.

number of dimensions	discretization f.p.o./mesh point	solution f.p.o./mesh point	ratio
1	3	6 (LU, CR)	2
2	5	9.5(PCR)	1.9
		~ 20 (FACR)	4
3	7	~ 30 (FFT + PCR)	~ 4.3
		~ 40 (FFT + FACR)	~ 5.7

We are comparing algorithms dealing with problems of different complexity so we should account for that somehow.

It is difficult to imagine a one-dimensional RES with lower op. count than the known ones hence we take the ratio of 2 as the limit. This limit has been reached also for the two-dimensional problems but it would appear that it is still to be found for the three-dimensional equation.

One hopes that the future 3-D RES would have an op. count of the order of $15n^3$ or twice as good as the best existing ones. It will, most likely, be based on the cyclic reductions.

5. Conclusions.

In this very short review of RES algorithms only a few most important methods used for solving e.p.d.e's have been outlined. Their importance is far greater than it would seem at first. It is true they only solve simple "Poisson-like" equations on a rectangle with uniform meshes, but many other methods for solving general equations on arbitrary domains use RES iteratively hence their efficiency depends on the performance of the simple RES. Just to mention one example: the whole rich field of the molecular dynamics simulations was severely restricted until an efficient RES algorithm tailor-made for that purpose was developed, the so called P^3M method due to Hockney and Eastwood.

References.

[1] George, J.A. Nested dissection of regular finite element mesh, SIAM J. Numer. Anal., 10 (1973) p. 345-363.

[2] George, J.A. and Liu, J.W.H. An automatic nested dissection algorithm for irregular finite element problems, SIAM J. Numer. Anal., 15 (1978) p. 1053-1069.

[3] Duff, I.S. and Steward, G.W. (ed.) Sparse Matrix Proceedings, SIAM Press (1978).

[4] Cooley, J.W. and Tukey, J.W. An algorithm for the machine calculation of complex Fourier series, Math. Comput., (1965) p. 297-301.

[5] Singleton, R.C. An algorithm for computing the mixed radix fast Fourier transform, IEEE Trans. Audio and Electroacoustics, AU-17, (1969) p.93-103.

[6] O'leary, D.P. and Widlund, O. Capacitance matrix methods for the Helmholtz equation on general three-dimensional regions, Math. Comp. 33, (1979) p. 849-879.

[7] Sweet, R.A. A generalized cyclic-reduction algorithm, SIAM J. Numer. Anal. 11, (1974) p. 506-520.

[8] Lorenz, E.N. A rapid procedure for inverting del-square with certain computers, Monthly Weather Review 104, (1976) p. 961-966.

[9] Hockney, R.W. Formation and stability of virtual electrodes in a cylinder, J. Appl. Phys. 39, (1968) p. 4166-4170.

[10] Proskurowski, W. and Widlund, O. On the numerical solution of Helmholtz's equation by the capacity matrix method, Math. Comput. 30, (1976) p. 433-468.

[11] Stone, H.L. Iterative solution of implicit approximations of multidimensional partial differential equations, SIAM J. Numer. Anal. 5, (1968) p. 530-558.

[12] Buzbee, B.L., Golub, G.H. and Nielson, C.W. On direct methods for solving Poisson's equations, SIAM J. Numer. Anal. 7, (1970) p.627-656.

[13] Buneman, O. A compact non-iterative Poisson-solver, Report 294, Stanford University Institute for Plasma Research, Stanford, California.

[14] Swarztrauber, P.N. A direct method for the discrete solution of separable elliptic equations, SIAM J. Numer. Anal. 11, (1974) p. 1136-1150.

[15] Hockney, R.W. A fast direct solution of Poisson's equation using Fourier analysis, J. Assoc. Computing Machinery 12, (1965) p. 95-113.

[16] Temperton, C. On the FACR(ℓ) algorithm for the discrete Poisson equation, Report 14, European Centre for Medium Range Weather Forecasts, Bracknell, (1977).

[17] Detyna, E. Point cyclic reductions for elliptic boundary value problems. I. The constant coefficient case, J. Comput. Phys. 33, (1979) p. 204-216.

[18] Eastwood, J.W. In : Computational Methods in Classical and Quantum Physics, ed. M.B. Hooper, p. 206-228, Advance Publications Ltd. (1976).