Abstract

We present a new algorithm for solving general semilinear, elliptic partial differential equations. The algorithm is based on Newton's Method but uses an approximate iterative method to solve the linear systems that arise at each step of Newton's Method. We show that the algorithm can maintain the quadratic convergence of Newton's Method and that it may be substantially faster than other available methods for semilinear or nonlinear partial differential equations.

The Application of Sparse Matrix Methods to the Numerical Solution of Nonlinear Elliptic Partial Differential Equations

S. C. Eisenstat, M. H. Schultz, and A. H. Sherman

Research Report #27

This research was supported in part by the Office of Naval Research, N0014-67-A-0097-0016.

1. The Model Problem

Let D be a bounded region of the plane, and let ∂D denote the boundary of D. We are interested in solving problems of the form

$$-\Delta u = f(u) \qquad \text{on } D \tag{1.1a}$$

$$\text{with } u = 0 \qquad \text{on } \partial D \tag{1.1b}$$

where f and D are such that

$$f_u(u) \leq \lambda < \wedge$$

(where $\wedge$ is the fundamental eigenvalue of $-\Delta$ on D), so that u(x,y) exists and is unique. Under suitable conditions, our results apply to more general semilinear, self-adjoint elliptic boundary value problems in two and three dimensions and to certain nonlinear problems.

To obtain a numerical solution to (1.1), we must reduce the continuous, infinite dimensional problem to a discrete, finite dimensional one. Several techniques are available to do this, and we consider the use of finite difference and finite element approximations.

For the finite difference approximation, we replace the differential operator of (1.1) by a five-point difference approximation on a square grid D_h with boundary ∂D_h. Here h is the vertical or horizontal distance between two adjacent grid points. Letting U_{ij} be the approximation to u(ih,jh), we see that we may derive a problem given by the equations

$$U_{i-1,j} + U_{i,j-1} + U_{i,j+1} + U_{i+1,j} - 4U_{ij} = h^2 f(U_{ij}) \tag{1.2a}$$

$$\text{for } (ih,jh) \in D_h$$

$$\text{and } U_{ij} = 0 \qquad (1.2b)$$

$$\text{for } (ih,jh) \in \partial D_h$$

which is equivalent to (1.1). More concisely, we may rewrite (1.2) as an N by N system of nonlinear equations

$$A(\underset{\sim}{U}) = \underset{\sim}{F}(\underset{\sim}{U}) \qquad (1.3)$$

where

$$\underset{\sim}{U} = \{U_{ij} : (ih,jh) \in D_h\}$$

$$\underset{\sim}{F}(\underset{\sim}{U}) = \{h^2 f(U_{ij}) : (ih,jh) \in D_h\}$$

N is the number of grid points in D_h, and A is obtained from the finite difference equations (1.2). The system (1.3) will have a unique solution $\underset{\sim}{U}$ and U_{ij} will be a close approximation to u(ih,jh) under the same conditions as we imposed on the system (1.1) earlier.

To solve (1.3) for $\underset{\sim}{U}$, we use Newton's Method. Letting $\underset{\sim}{U}^{(0)}$ be a given initial guess, we obtain a sequence of successive approximations $\{\underset{\sim}{U}^{(K)}\}$ to $\underset{\sim}{U}$

$$\underset{\sim}{U}^{(K+1)} = \underset{\sim}{U}^{(K)} - J(\underset{\sim}{U}^{(K)})^{-1}[A\underset{\sim}{U}^{(K)} - \underset{\sim}{F}(\underset{\sim}{U}^{(K)}] \qquad (1.4)$$

where $J(\underset{\sim}{U}^{(K)})$ is given by

$$J(\underset{\sim}{U}^{(K)}) = A - \frac{\partial \underset{\sim}{F}}{\partial \underset{\sim}{U}}(\underset{\sim}{U}^{(K)}) \qquad (1.5)$$

Under the assumptions given above, $J(\underset{\sim}{U}^{(K)})$ will be symmetric, positive definite, and sparse (i.e. the number of nonzeroes per row is bounded independent of h). As we shall see later, it is these properties which enable our algorithm to be efficient.

We now state the following result concerning the convergence of Newton's Method, cf. [8].

Theorem 1.1: Newton's Method is quadratically convergent:

$$\|\underset{\sim}{U}^{(K+1)} - \underset{\sim}{U}\| \leq c\|\underset{\sim}{U}^{(K)} - \underset{\sim}{U}\|^2$$

Thus convergence to the solution of the discrete problem will be quite rapid, requiring only $O(\log \log N)$ iterations to reduce the error by a factor of h^2.

As an alternative to this finite difference approximation, we now consider finite element methods for approximating the solution of (1.1). If we let $\frac{\partial \phi}{\partial u} = f(u)$, then (1.1) is equivalent to minimizing the functional $G[u]$ given by

$$G[u] = \int_D \{(\frac{\partial u}{\partial x})^2 + (\frac{\partial u}{\partial y})^2 - 2\phi(u)\}\, dx\, dy \tag{1.6}$$

over all functions $u \in W^{1,2}(D)$ satisfying $u = 0$ on ∂D.

Again we must reduce the infinite dimensional continuous problem (1.6) to a finite dimensional discrete one. To do this we introduce S_N, a finite dimensional subspace of $W^{1,2}(D)$ with a linearly independent set of basis functions $\{\psi_i\}_{i=1}^N$. Then we approximate u by

$$\sum_{i=1}^{N} \beta_i \psi_i(x,y)$$

with the coefficients β_i chosen so as to minimize

$$G[\sum_{i=1}^{N} \beta_i \psi_i] = \int_D \{(\frac{\partial}{\partial x} \sum_{i=1}^{N} \beta_i \psi_i)^2 + (\frac{\partial}{\partial y} \sum_{i=1}^{N} \beta_i \psi_i)^2 \tag{1.7}$$

$$-2\phi(\sum_{i=1}^{N} \beta_i \psi_i)\}\, dx\, dy$$

Differentiating, we wish to choose the β_i so that

$$\int_D \psi_i \{-\Delta(\sum_{i=1}^{N} \beta_i \psi_i) - f(\sum_{i=1}^{N} \beta_i \psi_i)\}\, dx\, dy = 0 \qquad (1.8)$$

for $1 \leq i \leq N$

Rewriting this is a more convenient form, we wish to solve the system of nonlinear equations given by

$$A\underset{\sim}{U} = \underset{\sim}{F}(\underset{\sim}{U}) \qquad (1.9)$$

where $\underset{\sim}{U} = \{\beta_i\}$,

$$A = \{a_{ij}\} = \{\int_D [\frac{\partial \psi_i}{\partial x} \frac{\partial \psi_j}{\partial x} + \frac{\partial \psi_i}{\partial y} \frac{\partial \psi_j}{\partial y}]\, dx\, dy\}$$

$$\text{and } \underset{\sim}{F}(\underset{\sim}{U}) = \{\int_D \psi_i f(\sum_{j=1}^{N} \beta_j \psi_j)\}$$

As before, we use Newton's Method to solve (1.9). In this case, we are given an initial guess $\underset{\sim}{U}^{(0)}$ and obtain a sequence of successive approximations to $\underset{\sim}{U}$

$$\underset{\sim}{U}^{(K+1)} = \underset{\sim}{U}^{(K)} - J(\underset{\sim}{U}^{(K)})^{-1}[A\underset{\sim}{U}^{(K)} - \underset{\sim}{F}(\underset{\sim}{U}^{(K)})] \qquad (1.10)$$

where

$$J(\underset{\sim}{U}^{(K)}) = A - \frac{\partial \underset{\sim}{F}}{\partial \underset{\sim}{U}}(\underset{\sim}{U}^{(K)}) = A - \{\int_D \psi_i \psi_j \frac{\partial f}{\partial \underset{\sim}{U}}(\sum_{\ell=1}^{N} \beta_\ell \psi_\ell)\} \qquad (1.11)$$

We will assume that $\{\psi_i(x,y)\}_{i=1}^{N}$ is a local basis (i.e. for each i, the support of ψ_i intersects with the support of a bounded number of ψ_j's). Then $J(\underset{\sim}{U})$ will be sparse, symmetric, and positive definite, and the system (1.9) will have the same properties as the system (1.3).

We have seen how we may use Newton's Method with either finite difference or finite element approximations. In practice, we rewrite the iteration equations (1.4) or (1.10) for Newton's Method so as to obtain

the sequence of successive approximations to $\underset{\sim}{U}$ by solving

$$J(\underset{\sim}{U}^{(K)})\underset{\sim}{\delta}^{(K)} = A\underset{\sim}{U}^{(K)} - \underset{\sim}{F}(\underset{\sim}{U}^{(K)}) \tag{1.12}$$

and setting

$$\underset{\sim}{U}^{(K+1)} = \underset{\sim}{U}^{(K)} - \underset{\sim}{\delta}^{(K)} \tag{1.13}$$

Our problem is then reduced to the solution of a sequence of related, sparse, symmetric, and positive definite systems of linear equations.

2. The Solution of Sparse, Symmetric, Positive Definite Systems of Linear Equations

In this section we consider methods for the solution of an N by N sparse, symmetric, positive definite system of linear equations

$$A\underset{\sim}{z} = \underset{\sim}{b} \tag{2.1}$$

As we saw in Section 1, such systems arise in the use of Newton's Method in the solution of a model nonlinear problem. We may solve (2.1) using either a direct decomposition method or one of several iterative methods, and we will discuss both classes of methods here.

The direct method that we consider is the Choleski decomposition, a symmetric variant of standard Gaussian elimination. To solve (2.1) we first decompose A into the product LL^T, with L a lower triangular matrix. Then we successively solve $L\underset{\sim}{w} = \underset{\sim}{b}$ followed by $L^T\underset{\sim}{z} = \underset{\sim}{w}$ to obtain the desired solution $\underset{\sim}{z}$.

It is well known that this direct method may be equally well applied to PAP^T instead of A for any permutation matrix P [8], and we will use this fact to reorder the variables and equations of the system (2.1) so as to reduce the storage and/or time required for its solution.

In general, the use of the Choleski decomposition to solve an N by N symmetric system of linear equations requires $O(N^2)$ storage and $O(N^3)$ time. However, since our model problem is sparse, we may use any of several techniques to reduce these requirements. All of the direct methods which we will describe are sensitive to the ordering of the variables

and equations. It is of great importance to choose a permutation matrix P so that the reordered matrix PAP^T can be decomposed in as little time and with as little storage as possible. It has been shown that our model problem on a square five-point finite difference mesh requires at least O(N log N) storage and at least $O(N^{3/2})$ time [7]. Moreover, using techniques which we describe later and an ordering due to Birkhoff and George [2], we can actually achieve these lower bounds.

The simplest methods which take advantage of the sparsity in A are the band methods. We define the bandwidth b(A) of A by

$$b(A) = \max \{|i-j|: a_{ij} \neq 0\} \tag{2.2}$$

It is easily verified that if $|i-j| > b(A)$, then $a_{ij} = \ell_{ij} = 0$, so that we need only store and operate on those elements which fall in the band of A (i.e. those elements a_{ij} such that $|i-j| \leq b(A)$). For our model problem the use of a band method with the natural (row by row) variable ordering will reduce the storage and time to $O(N^{3/2})$ and $O(N^2)$, respectively [8].

With more effort we can do even better than this. Band methods assume that all the matrix elements are nonzero within the band. For many variable orderings, though, this is not the case, and we can obtain further reductions in space and time by keeping more careful track of the positions of the nonzeroes in A and L. To do this we store the nonzero matrix elements row by row, keeping track of the number of nonzero elements in each row and the column index of each element. Since column accesses are quite costly, we perform all the eliminations in each row at

the same time and insert new nonzeroes where and when they occur during the decomposition process. Using such techniques has the advantage that we can fully utilize our knowledge about the zero structures of A and L in order to reduce the costs of solving (2.1). However, the data structures (linked lists) and programming needed to implement the scheme may be quite complex, the storage requirements are variable so that storage management may be difficult, and the technique may not adapt well to paged, virtual storage systems.

To avoid the difficulties just mentioned, we use an improvement, due to Chang [3], in which we preprocess the system (2.1) in order to determine the exact locations of the nonzeroes in L before we actually do the decomposition. This symbolic factorization, though quite complex in itself, need be done only once for a sequence of problems (2.1) in which A has a fixed zero structure (even if the actual values of the elements of A do change). Thus for an application like Newton's Method, the cost of the symbolic factorization may be spread over the solution to a number of systems (2.1). For each system to be solved, we need perform only a numerical factorization, a process which becomes quite efficient after the symbolic factorization has determined the exact zero structure of L. When this technique is used with the nested dissection variable ordering due to Birkhoff and George [2], the Choleski decomposition requires only $O(N \log N)$ storage and $O(N^{3/2})$ time.

The alternative to using a direct method for the solution of (2.1) would be to use one of the many iterative methods [13] which are avail-

able. These have been developed mainly to solve sparse linear systems like our model problem, and they work very well for regular problems on simple areas of the plane. The main advantages of the iterative methods are in the storage requirements (usually O(N)), in the fact that some of them may be faster than direct methods, and in their ability to make use of a good guess at the solution. This last point is especially applicable to our model nonlinear problem, for the solutions to the sequence of linear problems arising from Newton's Method are converging quadratically towards the solution of the nonlinear problem.

However, the good points of the iterative methods may often be outweighed by their negative features. Most important, it is difficult to know when to stop iterating, since a posteriori error bounds are usually not conveniently available. Also, several of the methods have parameters which must be accurately estimated in order for the methods to behave well. Finally, not all of the methods can be extended easily to non-rectangular or irregular domains, so that there are many problems for which the iterative methods may not help at all.

In Table 2.1 (cf. [6]) we summarize the storage and time requirements of several direct and iterative methods. One should be careful about making direct comparisons, however, since the choice of the best method for a problem may depend on the problem size, on the context of the linear system in a larger numerical method, and, especially, on the coefficients of the terms in the expressions for the storage and time requirements.

Table 2.1

Method	Storage	Time
Choleski Decomposition	$O(N^2)$	$O(N^3)$
Band Choleski Decomposition	$O(N^{3/2})$	$O(N^2)$
Sparse Choleski Decomposition	$O(N \log N)$	$O(N^{3/2})$
Gauss-Seidel Iteration	$O(N)$	$O(N^2)$
SOR Iteration (Optimal ω)	$O(N)$	$O(N^{3/2} \log N)$
SSOR Iteration	$O(N)$	$O(N^{5/4})$
ADI Iteration (Optimal α)	$O(N)$	$O(N \log^2 N)$
SIP Iteration	$O(N)$	$O(N \log^2 N)$

3. The Newton-Sparse-Richardson Method

We now return to the model nonlinear problem presented in Section 1 and discuss the numerical solution of the sequence of linear problems generated by Newton's Method. We consider direct decomposition methods and certain variations on them that we will mention later. Although we restrict our analysis to the model problem, it is clear that what we will say is applicable to any numerical method which requires the solution of a sequence of related systems of linear equations.

The most obvious method for solving the sequence of linear systems is to solve each system using the sparse Choleski decomposition presented in Section 2. We shall call this method Newton-Sparse (NS), and it is clear that it has all of the convergence properties of Newton's Method. The advantages of NS are that there are no parameters to estimate and that it is essentially independent of the domain and the source of the linear systems. However, it ignores the relationships which exist between the solutions of successive problems, and its requirements for $O(N \log N)$ storage and $O(N^{3/2} \log \log N)$ time very likely make it inferior to variations of Newton's Method using SSOR, ADI, or SIP (cf. [8]) to solve the linear systems occurring at each step (although this may depend heavily on implementation efficiency).

In an effort to avoid these difficulties, we will present a modification to the straightforward NS method based on the ideas embodied in the Strongly Implicit Procedures (SIP) of Stone [11], Diamond [4], and

others. The basic strategy for the solution of each linear system in the sequence is to use the solution to a different, but related, system in conjunction with an iteration procedure to obtain the solution to the system of interest. In particular, suppose that we wish to solve (2.1) and that for some matrix B (depending on A) we can quickly solve (say in O(N) time) the system

$$(A+B)\underset{\sim}{w} = \underset{\sim}{d} \tag{3.1}$$

We may then obtain the solution of (2.1) by using a Richardson-D'Jakonov iteration (cf. [5, 9, 13]) to generate a sequence of vectors $\{\underset{\sim}{z}^{(K)}\}$ which converge linearly to the desired $\underset{\sim}{z}$. For a given starting guess $\underset{\sim}{z}^{(0)}$, the successive iterates are obtained by solving the system

$$(A+B)\underset{\sim}{z}^{(K+1)} = (A+B)\underset{\sim}{z}^{(K)} + \gamma(A\underset{\sim}{z}^{(K)} - \underset{\sim}{b}) \tag{3.2}$$

or, equivalently, by first solving the system

$$(A+B)\underset{\sim}{\delta}^{(K)} = A\underset{\sim}{z}^{(K)} - \underset{\sim}{b} \tag{3.3}$$

and then setting $\underset{\sim}{z}^{(K+1)} = \underset{\sim}{z}^{(K)} + \gamma\underset{\sim}{\delta}^{(K)}$. The cost of each step of the iteration will be O(N), and the iteration will converge to the solution of (2.1) if $\rho \equiv \|I + \gamma(A+B)^{-1} A\| < 1$. To maximize the rate of convergence, we must choose γ to minimize ρ. If all the eigenvalues of $(A+B)^{-1} A$ lie in the interval $[\alpha,\beta]$, then we will choose $\gamma = 2/(\alpha+\beta)$ and obtain $\rho = (\beta-\alpha)/(\alpha+\beta)$. If A and B are such that α and β are independent of N, then the iteration will converge at a rate independent of the mesh size. Hence, the solution of (2.1) with the above version of SIP would require O(log N) Richardson iterations or O(N log N) total time in order to reduce the initial error by a factor of 1/N. Further, it has been shown

by several authors that these requirements may be reduced by using Tchebychev acceleration to choose a sequence of parameters $\{\gamma_K\}$ instead of the single parameter γ [4].

The main problem with SIP is that it has so far defied analysis even for our model problem, and it is not known whether there exists a matrix B for which α and β are independent of N. We now examine a method similar to SIP which does lead to a rate of convergence independent of N. SIP chooses a matrix B so that A+B has a sparse, symmetric decomposition of an especially nice form. What we propose is to choose B so that A+B is a matrix for which we already have a decomposition, even though the decomposition will have O(N log N), rather than O(N) nonzero elements. That is, in order to guarantee a rapid rate of convergence independent of the mesh size, we will sacrifice a factor of log N in the work per iteration. More explicitly, we use a direct decomposition method to decompose $J(\underset{\sim}{U}^{(0)})$ into a product LL^T in the first step of Newton's Method. Then at successive steps of Newton's Method, we define $B = J(\underset{\sim}{U}^{(0)}) - J(\underset{\sim}{U}^{(K)})$ and use the Richardson-D'Jakonov iteration given above to solve (1.12). We shall call this method Newton-Sparse-Richardson (NSR).

There are two main theoretical questions which arise concerning the use of NSR. First, is it possible to maintain the quadratic convergence of the Newton iteration by using an inner Richardson-D'Jakonov iteration scheme at each step? And if so, how many Richardson-D'Jakonov iterations are required to do it? In answer to these questions, we give the following two theorems, which will be proved elsewhere [10].

Theorem 3.1: If Newton's Method is quadratically convergent for a given semilinear problem (1.1), and if we define $\underset{\sim}{U}^{(K+1)}$ to be the 2^K-th Richardson-D'Jakonov iterate in the K-th step of NSR, then NSR is quadratically convergent independent of N:

$$\|\underset{\sim}{U}^{(K+1)} - \underset{\sim}{U}\| \leq c\|\underset{\sim}{U}^{(K)} - \underset{\sim}{U}\|^2, \quad c < 1.$$

Theorem 3.2: If Newton's Method is quadratically convergent for a given semilinear problem (1.1), and if we define $\underset{\sim}{U}^{(K+1)}$ to be the <u>first</u> Richardson-D'Jakonov iterate in the K-th step of NSR, then NSR is linearly convergent independent of N:

$$\|\underset{\sim}{U}^{(K+1)} - \underset{\sim}{U}\| \leq c\|\underset{\sim}{U}^{(K)} - \underset{\sim}{U}\|, \quad c < 1.$$

Having determined that NSR will actually converge whenever Newton's method converges, we now wish to know how much effort is required to solve a semilinear problem using NSR. Rather surprisingly, we have the following theorem which shows that if we neglect preprocessing and the cost of the sparse LL^T factorization at the first step, then the NSR methods described in Theorems 3.1 and 3.2 asymptotically require the same amount of time!

Theorem 3.3: Assume that Newton's Method converges quadratically for a given semilinear problem (1.1). Then the NSR methods described in Theorems 3.1 and 3.2 will reduce the initial error $\|\underset{\sim}{U}^{(0)} - \underset{\sim}{U}\|$ by a factor of $1/N$ in $O(N \log^2 N)$ iteration time, if we ignore preprocessing and the cost of the LL^T factorization at the first step.

Proof: Each Richardson-D'Jakonov (RD) iteration requires $O(N \log N)$ time. If we take 2^K RD iterations at the K-th step of NSR, Theorem 3.1 tells us that we can achieve quadratic convergence, so that we need only log log N NSR steps to reduce the initial error by 1/N. The total iteration time required for NSR is then

$$O\left(\sum_{K=1}^{\log \log N} 2^K N \log N\right) = O(N \log^2 N)$$

On the other hand, if we use only one RD iteration at each step of NSR, we will need log N steps to reduce the initial error by 1/N. Since each step of NSR will require $O(N \log N)$ time, we see again that the total time for NSR is $O(N \log^2 N)$.

If we consider the sparse factorization at the first NSR step to be a form of preprocessing, then NSR requires only $O(N \log^2 N)$ time as against the $O(N^{3/2} \log \log N)$ time required for the NS method that we discussed above. Other advantages of the NSR method are that it is independent of both the problem domain and the method used to generate the nonlinear system of equations and that it takes advantage of good guesses to the solution at each iteration step. The only serious disadvantages of NSR are its requirements for $O(N \log N)$ storage locations and $O(N^{3/2})$ preprocessing time.

4. Numerical Experiments

In this section, we shall illustrate our general method by solving the equation

$$(su_x)_x + (tu_y)_y = f(x,y,u)$$

over the domain

$$D = (0,1) \times (0,1) - [\tfrac{3}{8},\tfrac{5}{8}] \times [\tfrac{3}{8},\tfrac{5}{8}]$$

the unit square with a hole cut out. In particular, we consider two problems given by Bartels and Daniel [1]:

$$\{(1 + x^2 + y^2)u_x\}_x + \{(1 + e^x + e^y)u_y\}_y = g(x,y)e^u \qquad (4.1a)$$

in D

$$u(x,y) = x^2 + y^2 \qquad (4.1b)$$

on ∂D

and

$$\{(1 + (x-y)^2)u_x\}_x + \{(10 + e^{xy})u_y\}_y = g(x,y)(1 + u)^3 \qquad (4.2a)$$

in D

$$u(x,y) = x^2 + y^2 \qquad (4.2b)$$

on ∂D

where in each case g(x,y) is chosen to make the unique exact solution $u(x,y) = x^2 + y^2$. For all tests, the initial approximation was chosen to be zero inside the region and to satisfy the boundary conditions. The results are summarized in Table 4.1. By way of comparison, we also give in Table 4.2 the corresponding times for the Nonlinear Conjugate Gradient method (NCG) proposed by Bartels and Daniel [1]. All times listed are in

seconds and reflect execution times on a CDC-6600. We note that asymptotically NCG requires $O(N)$ storage, $O(N^{3/2} \log N)$ preprocessing time, and $O(N \log^2 N)$ iteration time, whereas NSR requires $O(N \log N)$ storage, $O(N^{3/2})$ preprocessing time, and $O(N \log^2 N)$ iteration time.

The results are clear: even including domain-dependent preprocessing time (ordering and symbolic factorization), which accounts for roughly half the total execution time, NSR reduces the nonlinear residual by a factor of 10^{-11} in significantly less time than NCG requires to reduce the error by a factor of 10^{-3}. There is little to choose between the two variants of NSR. Other experiments indicate that letting the number of Richardson iterations vary between the two extremes gives better results.

Table 4.1

Newton-Sparse-Richardson Method

	Problem (4.1)				Problem (4.2)			
	h = 1/16		h = 1/32		h = 1/16		h = 1/32	
Domain-dependent preprocessing time	.496	.474	3.901	3.786	.495	.511	3.902	3.908
Problem-dependent preprocessing time	.170	.167	1.211	1.173	.160	.165	1.164	1.162
Number of Richardson iterations per K-th Newton iteration	2^K	1	2^K	1	2^K	1	2^K	1
Richardson iteration time	.032	.032	.169	.165	.032	.034	.170	.174
Number of Newton iterations to reduce nonlinear residual by a factor of 10^{-11}	4	8	4	8	4	12	4	11
Nonlinear residual time	.055	.053	.227	.219	.044	.045	.178	.175
Total iteration time	.669	.650	3.274	2.918	.629	.918	3.086	3.604
Total time	1.335	1.291	8.386	7.877	1.284	1.594	8.152	8.674

Table 4.2

Nonlinear Conjugate Gradient Method

	Problem (4.1)		Problem (4.2)	
	h = 1/16	h = 1/32	h = 1/16	h = 1/32
Preprocessing time	.394	2.90	.398	2.82
Number of conjugate gradient iterations to reduce error by factor h^2	6	8	9	12
Time per iteration	.347	1.20	.287	.943
Total iteration time	2.08	9.63	2.58	11.3
Total time	2.57	12.9	3.05	14.4

5. Conclusion

In this paper we have presented a means for improving the efficiency of Newton's Method as used in the numerical solution of semilinear self-adjoint, elliptic partial differential equations. However, our technique of using an SIP-like iteration to solve the linear systems that arise in each step of Newton's Method is not limited to just this one application. Under suitable conditions, the method may be equally useful in the numerical solution of more general nonlinear and time dependent problems. In fact, the technique may prove fruitful whenever the numerical solution of any problem requires the solution of a sequence of related systems of linear equations; the only restriction is that the operators in successive systems be close enough so that the Richardson-D'Jakonov iteration converges quickly.

In theory we have seen that our method is quite efficient. The numerical examples show the practical value of the method, both as a tool for enhancing the efficiency of Newton's Method and as a very efficient means of solving certain semilinear problems for which other methods have been successfully used in the past. We are currently examining the applicability of the method to the numerical solution of more general nonlinear problems, and we fully expect that it will be able to compete quite successfully with methods which are currently in use.

References

[1] R. Bartels and J. W. Daniel. A conjugate gradient approach to nonlinear elliptic boundary value problems in irregular regions. Report #CNA 63, Center for Numerical Analysis, The University of Texas at Austin, 1973.

[2] G. Birkhoff and D. J. Rose. Elimination by nested dissection. Complexity of Sequential and Parallel Numerical Algorithms, edited by J. F. Traub, Academic Press, New York, 1973.

[3] A. Chang. Application of sparse matrix methods in electric power system analysis. Sparse Matrix Proceedings, edited by R. A. Willoughby, IBM Research Report #RA1, Yorktown Heights, New York, 1968.

[4] M. A. Diamond. An Economical Algorithm for the Solution of Finite Difference Equations, PhD dissertation, Department of Computer Science, University of Illinois, 1971.

[5] E. G. D'Jakonov. On certain iterative methods for solving nonlinear difference equations. Proceedings of the Conference on the Numerical Solution of Differential Equations, (Scotland, June 1969), Springer-Verlag, Heidelberg, 1969.

[6] F. W. Dorr. The direct solution of the discrete Poisson equation on a rectangle. SIAM Review 12:248-263, 1970.

[7] A. J. Hoffman, M. S. Martin, and D. J. Rose. Complexity bounds for regular finite difference and finite element grids. SIAM Journal on Numerical Analysis 10:364-369, 1973.

[8] J. M. Ortega and W. C. Rheinboldt. Iterative Solution of Nonlinear Equations in Several Variables, Academic Press, New York, 1970.

[9] L. F. Richardson. The approximate arithmetical solution by finite differences of physical problems involving differential equations with an application to the stresses in a masonry dam. Philosophical Transactions of the Royal Society, London, Series A(210):307-357, 1910.

[10] A. H. Sherman. PhD dissertation, Department of Computer Science, Yale University. To appear.

[11] H. L. Stone. Iterative solution of implicit approximations of

multidimensional partial differential equations. SIAM Journal on Numerical Analysis 10:530-558, 1968.

[12] J. H. Wilkinson. The Algebraic Eigenvalue Problem, Clarendon Press, London, 1965.

[13] D. M. Young. Iterative Solution of Large Linear Systems, Academic Press, New York, 1971.