SIAM J. SCI. STAT. COMPUT.
Vol. 7, No. 4, October 1986

A MULTIGRID CONTINUATION METHOD FOR ELLIPTIC PROBLEMS WITH FOLDS*

JOHN H. BOLSTAD† AND HERBERT B. KELLER‡

Abstract. We introduce a new multigrid continuation method for computing solutions of nonlinear elliptic eigenvalue problems which contain limit points (also called turning points or folds). Our method combines the frozen tau technique of Brandt with pseudo-arc length continuation and correction of the parameter on the coarsest grid. This produces considerable storage savings over direct continuation methods, as well as better initial coarse grid approximations, and avoids complicated algorithms for determining the parameter on finer grids. We provide numerical results for second, fourth and sixth order approximations to the two-parameter, two-dimensional stationary reaction-diffusion problem:

$$\Delta u + \lambda \exp(u/(1 + \alpha u)) = 0.$$

For the higher order interpolations we use bicubic and biquintic splines. The convergence rate is observed to be independent of the occurrence of limit points.

Key words. multigrid, arc-length continuation, nonlinear elliptic eigenvalue problems, limit points, folds, frozen tau method, tau extrapolation, deferred correction, defect correction

AMS(MOS) subject classifications. 65N20, 65N25

1. Introduction. Many problems of computational interest can be viewed in the general form:

$$G(u, \boldsymbol{\lambda}) = 0, \tag{1.1}$$

where X is some Banach space, $u \in X$, $\boldsymbol{\lambda} \in R^{\nu}$, and $G: X \times R^{\nu} \to X$. Of course u represents a "solution" field (e.g., flow field, displacements, etc.), and $\boldsymbol{\lambda}$ is a real vector of physical parameters (e.g., Reynolds number, load, etc.). It is required to find the solution for some $\boldsymbol{\lambda}$-intervals (or some $\boldsymbol{\lambda}$-arcs), that is, a path (or manifold) of solutions: $(u(\boldsymbol{\lambda}), \boldsymbol{\lambda})$. We consider problems of the form (1.1) which are nonlinear elliptic eigenvalue problems.

A common feature on solution paths of such problems is the frequent occurrence of *limit points* (also called *turning points* or *folds*). Figure 1 illustrates two limit points in the $(\lambda, \|u\|_{\infty})$ plane for two different families of numerical solutions. The limit points are at $\lambda = \lambda_f$ and λ_c. A *limit point* $P^0 = (u^0, \lambda^0)$ is defined as a solution of (1.1) for which the Fréchet derivative G_u^0 of G with respect to u evaluated at P^0 satisfies:

a) G_u^0 is singular;

b) $G_\lambda^0 \notin R(G_u^0)$.

The limit point or fold is said to be *simple* if in addition:

c) $\dim N(G_u^0) = \operatorname{codim} R(G_u^0) = 1$.

Here N and R denote null space and range, respectively. In the rest of this paper, when we refer to limit points, we shall mean simple limit points.

Pseudo-arc-length continuation methods (Keller [19], [20]) have been used successfully for computing limit points and bifurcation points of nonlinear elliptic eigen-

* Received by the editors February 8, 1984, and in revised form June 13, 1985. This work was typeset at the Lawrence Livermore National Laboratory using a *troff* program running under UNIX. The final copy was produced on July 23, 1985. It was supported by the U.S. Department of Energy under contract EY-76-S-03-070 and by the U.S. Army Research Office under contract DAAG-29-78-C-0011.

† Present address: Lawrence Livermore National Laboratory, Box 808 L-16, Livermore, California 94550.

‡ Applied Mathematics 217-50, California Institute of Technology, Pasadena, California 91125.

value problems (e.g., Meyer-Spasche and Keller [24], Schreiber and Keller [32]). However, the size of problems that can be solved by these methods is limited principally by the storage required for the Jacobian G_u, as well as the time required for the direct factorization of these Jacobians. For these reasons it is natural to consider multigrid methods, which, for a grid with N^2 mesh points, require $O(N^2)$ storage and approximately $O(N^2)$ arithmetic operations to solve to the accuracy of the truncation error.

Straightforward implementations of multigrid methods work well for nonlinear elliptic eigenvalue problems, but they fail near singular points. One type of difficulty near a limit point has been noted by several workers (e.g., Chan and Keller [9], Meis, Lehmann and Michael [23]). Assume that the solution branches on coarse and fine grids look like those in Fig. 1, say Γ_1 and Γ_M, respectively. Suppose we use "natural continuation" with λ as parameter on the coarse grid, and use the full approximation scheme (FAS) full multigrid algorithm while keeping λ fixed. Assume we start at λ_1 on the coarse grid, and wish to compute the fine grid solution at the limit point λ_f.

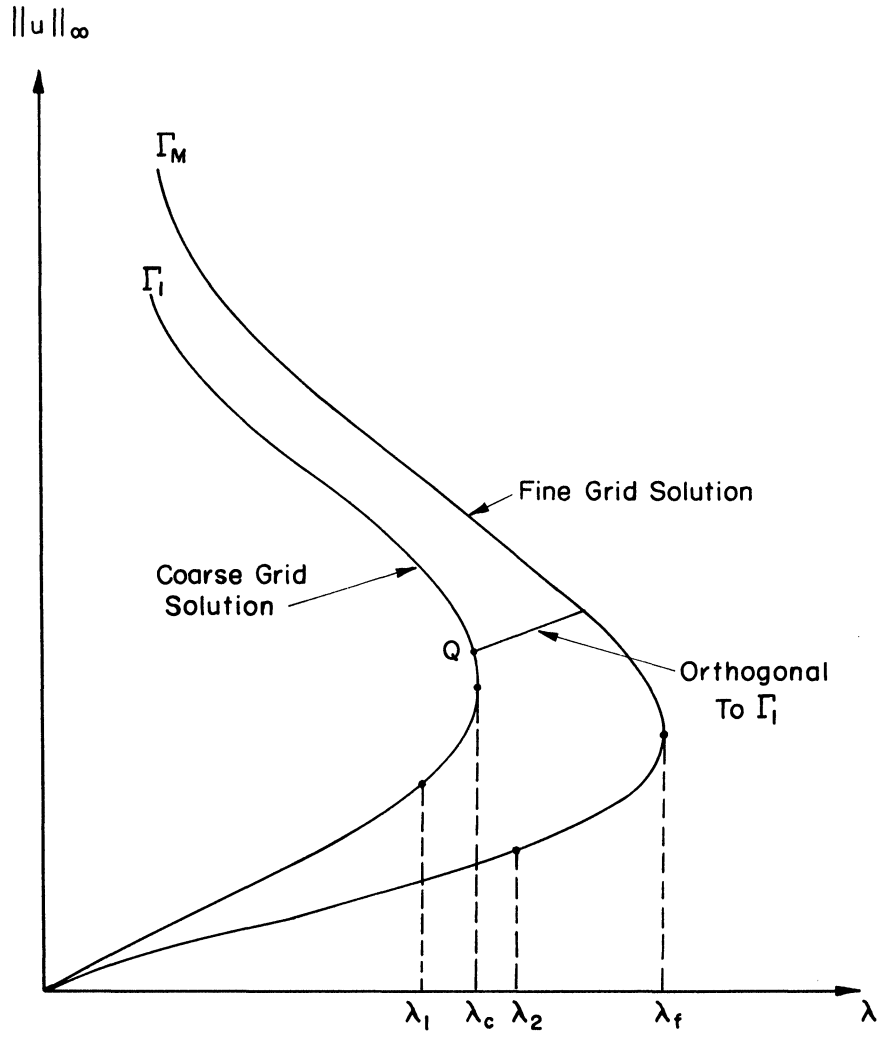

FIG. 1. *Limit points for different grids.*

As we approach the coarse grid limit point λ_c, there is a drastic slowing in the rate of convergence of the multigrid method, followed by divergence at λ_c. The divergence occurs first on the coarsest grid. (Although Chan and Keller [9] used the Cycle C "Correction Scheme" multigrid algorithm, we found the same results when we used the FAS full accommodative multigrid algorithm.) For points $\lambda_2 > \lambda_c$ there is no coarse grid solution, so we will not be able to approach λ_f without some remedy, such as deleting coarse grid(s).

Another (less well-known) type of difficulty occurs when the coarse and fine grid solution paths in Fig. 1 are interchanged. If we approach the fine grid limit point λ_f in the direction of increasing λ, the multigrid method converges, but the solution is not very accurate. However, if we do not know the location of λ_f in advance, we will try to compute solutions for λ_2 with $\lambda_f < \lambda_2 < \lambda_c$. For these λ_2 no fine grid solution exists. In practice, for λ_2 not "sufficiently close" to λ_f, the residuals of the fine grid solution cannot be reduced below a certain level, i.e., we encounter divergence on the fine grid.

Both configurations occur in practice, even using different approximations on the same problem. See §6, Table 1.

One remedy for the first type of difficulty was suggested by Chan and Keller [9]. They approximated the eigenfunction of the discrete differential operator that was causing the divergence, and then altered the interpolation operator and skipped a grid to prevent divergence.

All other methods do not fix the parameter λ during the multigrid iterations, i.e., they use different parameters λ^k on different grid levels k. One class of methods has been suggested by several authors, e.g., Meis, Lehmann and Michael [23], Stüben and Trottenberg [35]. Consider the model problem (1.2). In addition to the usual difference approximations, they impose an additional constraint on the finest grid, say:

$$u^M(P) \equiv u^M(1/2, 1/2) \equiv \|u^M\|_\infty = \text{constant},$$

where u at the center point is specified but λ^M remains unknown. This amounts to switching the role of unknown $u^M(P)$ and parameter λ^M. This parameter switching is one of the earliest techniques for treating limit points for single grid methods (see, e.g., Abbott [1]).

More specifically, Stüben and Trottenberg suggest modifying the multigrid smoothing step at one level as follows:

(a) Apply one (nonlinear) smoothing (relaxation) step to the actual approximation u^k, including the prescribed value at P.

(b) Multiply the relaxed approximation by a factor such that the constraint is satisfied afterwards.

(c) Compute a new value of λ^k for this level by using the difference equations together with a one-dimensional rootfinder (e.g., using a suitable average over all the equations).

Of course, this procedure will produce different λ^k at different grid levels if the constraint has the same value on all levels. At convergence, however, the parameters λ^k will be (approximately) the same if the FAS multigrid algorithm is used.

An objection to this procedure is, of course, its lack of generality. In more complicated problems (e.g., Lentini and Keller [22]) there is no clear "natural" parameter which can be used to eliminate limit points. Rheinboldt [29] circumvents this problem for single grid algorithms by treating the parameter as an additional unknown, and at each step appropriately selecting one of the unknowns as the continuation parameter.

A second class of methods has been proposed by Hackbusch [17], and by Bank and Chan [2]. Assume we are given the solution structure of Fig. 1. Having found a coarse grid solution point $Q \equiv (u, \lambda) \in R^{J+1}$ on Γ_1, these workers suggest finding a corresponding fine grid point on the line orthogonal (in the space R^{J+1}) to the coarse grid curve passing through Q. The projection of this orthogonal onto the $\lambda - \|u\|_\infty$ plane is shown in Fig. 1. Furthermore, Bank and Chan use additional diagonal shifts of Jacobians to assure that all matrices have the same number of negative eigenvalues. A third method, the generalized inverse iteration method, has been proposed in Mittlemann [25], and Mittlemann and Weber [26]. Their method determines the parameter λ^k on each grid as a generalized Rayleigh quotient.

Instead of using different parameter values on different grids, our method changes the structure of the solutions on different grids, by using the frozen tau technique of Brandt [7], [8]. In Fig. 1, our method in effect "moves" the coarse grid curve so that locally it is very close to the fine grid curve. Then the parameters λ needed on different grids are very near to each other, so we may use the same λ on all grids, and change λ (using pseudo-arc-length continuation) only on the coarsest grid. The multigrid and continuation methods interact through the full approximation scheme (eq. (3.3)). A special procedure (eq. (4.7)) produces accurate initial approximations.

Since we use the frozen tau method, the coarse grid equations produce approximations which, in general, more closely approximate the fine grid solutions than the solutions of the unmodified coarse grid equations. This is an advantage when computing problems with multiple solutions. Together with Mittlemann's method, our method is *robust* in the sense that the same algorithm can be used for both regular points and limit points, without significant loss of efficiency. That is, we need not detect the presence of limit points and try to switch algorithms. Our method requires a Jacobian only on the coarsest grid, so it is well-suited to large problems, especially those obtained from discretizing coupled systems of partial differential equations on fine grids [24], [31]. Finally, in our numerical results we observed that the rate of convergence (i.e., the rate of decrease of residual norms) did not degrade near limit points.

Throughout this paper we consider the following model problem, which will be used to illustrate the methods and to furnish numerical examples.

A two-dimensional stationary reaction-diffusion problem is formulated as:

$$Lu(\lambda, \alpha) \equiv \Delta u + \lambda \exp(u/(1 + \alpha u)) = 0 \quad \text{in } \Omega, \tag{1.2a}$$

$$u = 0 \quad \text{on } \partial\Omega. \tag{1.2b}$$

We take $\Omega \equiv$ the unit square: $[0, 1]\times[0, 1]$. In the notation of (1.1), $\mathbf{\lambda} = (\lambda, \alpha)$.

The solution of this problem has interesting geometric features (see Fig. 2). There exists a critical value of α, say α^*, for which: i) $u(\cdot, \lambda, \alpha)$ has two simple (quadratic) limit points in λ for fixed $\alpha < \alpha^*$; ii) $u(\cdot, \lambda, \alpha)$ has no limit points for fixed $\alpha > \alpha^*$; iii) $u(\cdot, \lambda, \alpha^*)$ has one (cubic) limit point. In case i), in a sufficiently small neighborhood of each limit point P_i, the solution curve lies on one side of the (vertical) tangent line at P_i, while in case iii) the solution curve lies on both sides of the tangent line at the limit point. See Spence and Werner [34] for a discussion of methods to compute α^*.

In § 2 of this paper we review arc-length continuation methods. Section 3 summarizes the accommodative FAS full multigrid algorithm for a fixed parameter value. In §4 we review the frozen tau algorithm and then describe our method. We also describe implementation details such as the use of high order splines for interpolation. An outline of a convergence proof, which is based on methods of Hackbusch [16], [17] is given in § 5. In § 6 we give numerical results for second, fourth and sixth order approximations to our model problem (1.2).

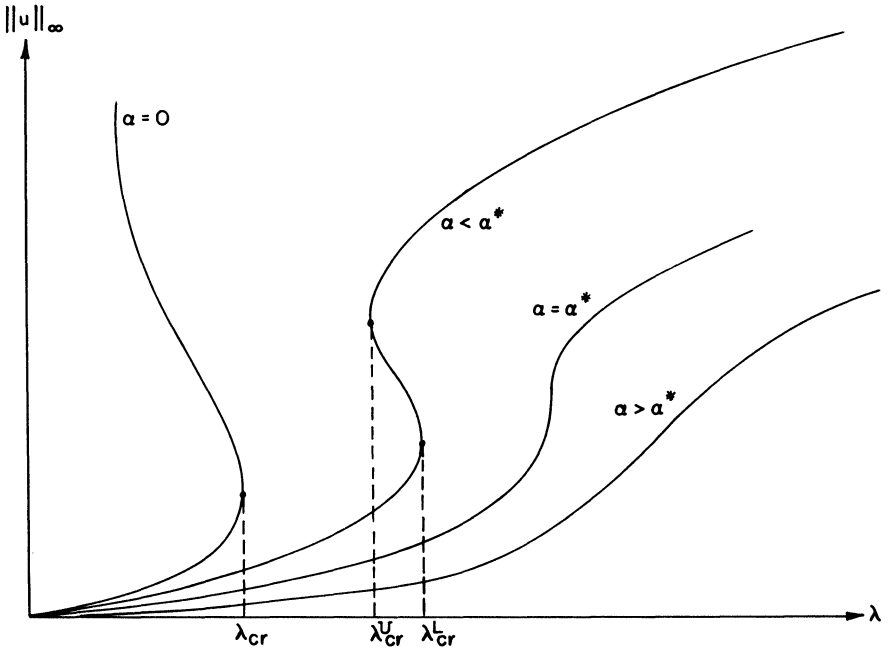

FIG. 2. *Solution curves of reaction-diffusion equation* (1.2) *for various values of parameter* α.

2. Continuation methods. In this section we briefly review continuation methods for computing a family or path of solutions of (1.1), without using multigrid methods.

2.1. Newton's method. Given a value of λ and an initial guess u^0 of the solution $u(\lambda)$, we perform the following steps repeatedly until $\|\delta u^i\| < \varepsilon$ is satisfied:

$$G_u(u^i, \lambda)\delta u^i = -G(u^i, \lambda), \tag{2.1}$$

$$u^{i+1} = u^i + \delta u^i. \tag{2.2}$$

This procedure will generally converge quadratically when it does converge. However, as is well known, it can fail to converge when the initial guess is not sufficiently close to a solution, or if a solution does not exist for the given λ value.

2.2. Natural continuation. To overcome the former difficulty, we can start at a known solution (u_0, λ_0) on the solution path and use it as an initial guess for a Newton-type iteration to find the solution for a neighboring point on the solution path with λ close to λ_0. The procedure is then repeated. We can improve on this by computing the derivative u_λ at a known solution and use it to get a better initial guess for the next value of λ in a predictor-corrector fashion. We call this a *natural* continuation procedure because it corresponds to parametrizing the solution path by λ, the naturally occurring parameter. A specific form of this is the well-known:

Euler-Newton continuation procedure. Given a known solution (u_0, λ_0), compute the solutions at nearby values of λ as follows:

1. First compute the derivative u_λ at (u_0, λ_0) from

$$G_u u_\lambda = -G_\lambda.$$

2. Perform an Euler predictor step:

$$u^0 = u_0 + u_\lambda(\lambda - \lambda_0).$$

3. Use u^0 as initial guess in Newton's method. Repeat

$$G_u(u^{i+1} - u^i) = -G(u^i, \lambda),$$

until convergence.

4. Use $(u(\lambda), \lambda)$ as the new (u_0, λ_0) and go to Step 1.

Unfortunately, this procedure needs some modification in order to handle general nonlinear systems because of the possibility of nonunique solutions. The nonuniqueness usually manifests itself in the form of existence of "singular" points where the Jacobian, G_u, is singular. The most common (i.e., generic) singular point is a simple limit point. (Another type of singular point, a bifurcation point, at which $G_\lambda \in R(G_u)$, will not be considered in this paper.) A natural continuation procedure will encounter two difficulties at limit points. First, since G_u is singular at these points, Newton's method (unaltered) will at best be linearly convergent, making it much more costly to compute the solution. Second, and more serious, the Euler-Newton procedure may lead us to a value of λ (such as λ_2 for the coarse grid in Fig. 1) for which no nearby solution exists, and the iterations will generally fail to converge.

2.3. Pseudo-arc-length continuation. In the pseudo-arc-length approach (Keller [19]), these difficulties are overcome by not parametrizing the solution u by λ. Instead, we parametrize the solution branches using a pseudo-arc-length parameter s, and specify how far along the current solution branch we want to try to march in s. This requires us to add an "arc-length" equation to our system of equations.

To be more specific, we let s be the arc-length-like parameter, and treat $u(s)$ and $\lambda(s)$ as functions of s. We then replace the Euler-Newton continuation procedure by the following:

Pseudo-arc-length Euler-Newton continuation procedure [19]: Assume given a solution $(u(s_0), \lambda(s_0))$.

1. Compute a tangent

$$(\dot{u}_0, \dot{\lambda}_0) \equiv (\dot{u}(s_0), \dot{\lambda}(s_0))$$

to the solution branch (where the dots denote differentiation with respect to s) satisfying:

$$G_u\dot{u}_0 + \dot{\lambda}_0 G_\lambda = 0, \tag{2.3}$$

$$\|\dot{u}\|^2 + |\dot{\lambda}|^2 - 1 = 0. \tag{2.4}$$

Equation (2.3) is obtained by differentiating (1.1) with respect to s, and (2.4) is an arc-length condition. The norm $\|\cdot\|$ is a discrete vector norm which approximates the continuous L_2 norm for integrals. For example, in m dimensions,

$$\|u\|^2 = h^m \sum_i u_i^2,$$

where u_i are the vector components. The equations (2.3)-(2.4) are easily solved [20] as follows.

First solve the system of linear equations

$$G_u\phi = -G_\lambda; \tag{2.5a}$$

then determine $\dot{u}(s_0)$ and $\dot{\lambda}(s_0)$ by

$$\dot{\lambda}_0 = \pm(1 + \|\phi\|^2)^{-1/2}, \tag{2.5b}$$

$$\dot{u}_0 = \dot{\lambda}_0\phi. \tag{2.5c}$$

Here the sign in (2.5b) determines the direction in which we traverse the solution path. If two solutions (u_{-1}, λ_{-1}) and (u_0, λ_0) have been computed, then we choose the sign such that:

$$\langle \dot{u}_0, u_0 - u_{-1} \rangle + \dot{\lambda}_0(\lambda_0 - \lambda_{-1}) > 0,$$

and the continuation proceeds in the direction from (u_{-1}, λ_{-1}) to (u_0, λ_0). Here $\langle\ ,\ \rangle$ is an inner product which induces the norm in (2.4).

2. Select a step size $s - s_0$. (See § 4.5.) Then take an Euler step of length $s - s_0$ along the tangent:

$$u^0 = u_0 + (s - s_0)\dot{u}_0, \tag{2.6a}$$

$$\lambda^0 = \lambda_0 + (s - s_0)\dot{\lambda}_0. \tag{2.6b}$$

In general (u^0, λ^0) will not be a solution of (1.1).

3. We now use the *pseudo-arc-length condition* to return to the solution branch. We require our new solution point to satisfy the equations

$$G(u(s), \lambda(s)) = 0, \tag{2.7}$$

$$N(u(s), \lambda(s), s) = 0, \tag{2.8a}$$

where

$$N(u(s), \lambda(s), s) \equiv \langle \dot{u}_0, (u(s) - u(s_0)) \rangle + \dot{\lambda}_0(\lambda(s) - \lambda(s_0)) - (s - s_0) = 0. \tag{2.8b}$$

Equation (2.8) is a linearization of the arc-length condition (2.4) and is used because it contains $(u(s), \lambda(s))$ and not $(\dot{u}(s), \dot{\lambda}(s))$. Equation (2.8) forces the new solution to lie on a hyperplane perpendicular to the tangent vector to the solution curve at s_0, and at a distance $|s - s_0|$ from it.[1] We solve the coupled system (2.7)-(2.8) for $u(s)$ and $\lambda(s)$ by using Newton's method with initial guess (u^0, λ^0). This requires solving the following system at each iteration:

$$A\begin{bmatrix} \delta u^i \\ \delta \lambda^i \end{bmatrix} \equiv \begin{bmatrix} G_u & G_\lambda \\ N_u & N_\lambda \end{bmatrix}\begin{bmatrix} \delta u^i \\ \delta \lambda^i \end{bmatrix} = -\begin{bmatrix} G \\ N \end{bmatrix}. \tag{2.9}$$

The quantities G, N and their derivatives are all evaluated at $(u^i(s), \lambda^i(s))$.

It can be shown [19] that at simple limit points, the linear system in (2.9) is nonsingular, and so Newton's method for the coupled system (2.7)-(2.8) is well-defined. Hence simple limit points present no computational problems and even quadratic convergence is achievable.

In order to solve the linear system in (2.9) by direct or iterative methods, several approaches are possible. One way is to perform Gaussian elimination on the inflated matrix A, with some form of pivoting to ensure stability. But this approach completely ignores the sparse structure which is usually found in the Jacobian G_u arising from discretizations of nonlinear elliptic eigenvalue problems. In order to take advantage of the structure in the Jacobian, Keller [19], [21] used the following bordering algorithm:

Solve

$$G_u y = G_\lambda \tag{2.10}$$

[1] Another choice is: $N = \|u - u_0\|^2 + |\lambda - \lambda_0|^2 - |s - s_0| = 0$. This forces the solution to lie on a sphere of radius $|s - s_0|$ about the previous solution.

and

$$G_u z = -G. \tag{2.11}$$

Set

$$\delta\lambda = (N + N_u^T z)/(N_\lambda - N_u^T y), \tag{2.12}$$

$$\delta u = z + \delta\lambda y. \tag{2.13}$$

Now only systems with the coefficient matrix G_u have to be solved, so structures in G_u can be exploited. Moreover, only one factorization of G_u is needed. It has been shown (Szeto [34]), and observed in practice, that even when G_u becomes singular, this bordering algorithm produces iterates that converge quadratically at simple limit points. Some loss in accuracy (cancellation errors in (2.13)) are to be expected, however [21]. We discuss this further in § 4.5.

But, as mentioned before, a major disadvantage of these continuation methods for many problems is the need to store and factor large Jacobians, G_u. For that reason we consider multigrid methods.

3. Multigrid methods. In this section we shall give a brief description of the multigrid method we use for a fixed value of the parameter vector λ. It is assumed (in this section only) that (u, λ) is not a limit point (or bifurcation point). In the next section we shall give modifications necessary for continuation. For a survey and more complete description of multigrid methods, see Brandt [5], Brandt and Dinar [6], Brandt [8], Stüben and Trottenberg [35], and other papers in the latter volume. We use *accommodative, full* multigrid with the *full approximation scheme* (FAS). Unlike Chan and Keller [9] we do not use the "Cycle C" algorithm.

We consider an elliptic partial differential equation defined on a region Ω with boundary $\partial\Omega$:

$$\begin{aligned} LU &= F \quad \text{on } \Omega, \\ U &= 0 \quad \text{on } \partial\Omega. \end{aligned} \tag{3.1}$$

(If U or F depend on a parameter λ, fix the parameter.) Then we construct a hierarchy of grids $\Omega^1, \Omega^2, \cdots, \Omega^M$, all approximating the domain Ω, with corresponding mesh sizes $h_1 > h_2 > \cdots > h_M$. The discrete approximation on grid Ω^k is written as

$$\begin{aligned} L^k U^k(x) &= F^k \quad \text{on } \Omega^k, \\ U^k &= 0 \quad \text{on } \partial\Omega^k, \end{aligned} \tag{3.2}$$

where $\partial\Omega^k$ approximates the boundary $\partial\Omega$. We wish to solve this discrete problem on the finest grid, Ω^M.

Figure 3, adapted from [6], gives a flow chart of the FAS full multigrid (FMG) algorithm. In the FAS method, each U^k is an approximation to the exact solution U. The FAS method (as opposed to the correction scheme) is particularly suited to the solution of nonlinear problems. When properly employed, it eliminates the need for large Jacobians.

The FMG method can be divided into two phases: the "initialization" phase, in which we start with a solution obtained in some manner on the coarsest grid Ω^1, and "bootstrap" our way up to a first approximation u^M, on the finest grid Ω^M. (For continuation problems it is important to have good first approximations on Ω^M.) The second phase improves the first approximation on grid Ω^M. This phase is common to all multigrid algorithms. In the flow chart, these phases are combined by replacing the

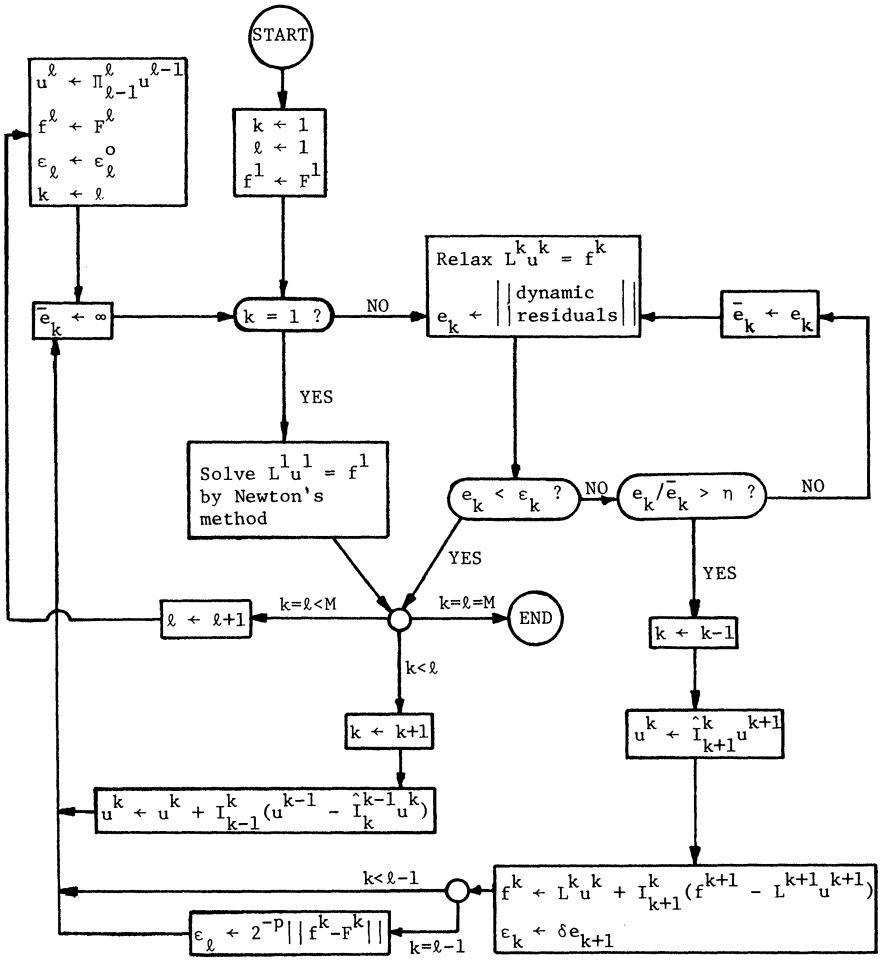

FIG. 3. *Accommodative FAS full multigrid algorithm. The notation is explained in the text.*

maximum level M with a level l, which denotes the highest grid level seen up to this time. The index l increases from 1 to M in phase one, and is fixed at M in phase two.

It is somewhat more natural to view the FMG method recursively; such a formulation is given in Hackbusch [16].

We now explain the quantities in the flow chart. The current grid level is denoted by k. In the FAS multigrid method, equations (3.2) are solved only at the highest current level l. The current approximation to U^l at this level is denoted by u^l. In the process, eq. (3.2) on coarser grids ($k < l$) are modified by changing their right-hand sides. The modified right-hand sides of the kth level equations are given by

$$f^k = L^k(\hat{I}^k_{k+1}u^{k+1}) + I^k_{k+1}(f^{k+1} - L^{k+1}u^{k+1}), \tag{3.3}$$

and depend on u^{k+1}, the current approximation on the next finer level. The solutions to these modified equations are denoted by $\tilde{U}^k$, and their computed approximations by u^k. Thus, the FAS multigrid method attempts to solve the equations

$$L^k\tilde{U}^k = f^k \tag{3.4}$$

at each level $k < l$, not (3.2).

The order of accuracy of the difference approximation is p. The norm of the dynamic residual on level k is

$$e_k = \|f^k - L^k u^k\|,$$

and its tolerance is ε_k. The interpolation operator I_{k-1}^k interpolates *corrections* from a "coarse" grid to the next finer grid. The interpolation operator Π_{l-1}^l interpolates *solutions* from a coarse grid to the next finer grid; it is used only in phase one, and frequently must be of higher order than I_{k-1}^k. The operators I_{k+1}^k and $\hat{I}_{k+1}^k$ are projection operators from a "fine" grid to the next coarser grid. The former projects residuals and the latter projects approximate solutions.

The parameter η controls switching to a coarser grid. When the norm e_k of the residual for this iteration of the relaxation process on grid k exceeds η times the corresponding residual norm $\bar{e}_k$ of the preceding iteration (i.e., the convergence rate becomes too slow), we switch to grid $k-1$. The parameter δ controls the accuracy with which we solve the equations on grid $k-1$ before using the result to correct the solution on grid k. The use of η and δ characterizes an *accommodative* multigrid algorithm, where the number of relaxation sweeps on a grid is determined dynamically by the progress of the iterations. This is in contrast to *fixed* algorithms, which perform a fixed number of relaxation sweeps on grid k before going to grid $k-1$, and then a fixed number of sweeps after returning to grid k. Accommodative methods are more robust, especially for nonlinear problems.

On the coarsest grid we do not use a relaxation method. On the upper branches of multiple solutions (Figs. 1 and 2) the matrix G_u becomes indefinite. This leads to divergence on the coarsest grid if a standard relaxation method is used ([9], [35]). Hence we use the full Newton method on the coarse grid. In § 4.2 we will see further reasons for using Newton's method here.

We defer until § 4.5 further details of our multigrid implementation.

4. Description of the algorithm. In this section we describe our algorithm. This description will involve Brandt's frozen tau method, and some modifications to the methods described in §2 and 3. Our goals in designing a continuation method are to do as much as possible on the coarse grid, and to avoid the need for large order Jacobians.

4.1. The frozen tau technique. We first describe the "dual view" (Brandt [8], Stüben and Trottenberg [35]) of the multigrid algorithm, and the relative truncation error. The local truncation error of the kth level approximation is

$$\tau^k \equiv L^k(\hat{I}^k U) - F^k = L^k(\hat{I}^k U) - I^k(LU),$$

where $\hat{I}^k$ projects continuous solutions onto the kth grid, and I^k (possibly different from $\hat{I}^k$) does the same for right-hand sides. This is obtained by substituting the solution U of the differential equation into the difference equations (3.2).

To estimate τ^k, we replace U by the "converged" approximate solution u^m, $k < m \leqq M$, and replace the projections I^k and $\hat{I}^k$ by I_m^k and $\hat{I}_m^k$, respectively. We then obtain the (m, k)-*relative truncation error*, or *fine-to-coarse defect correction*

$$\tau_m^k = L^k(\hat{I}_m^k u^m) - I_m^k(L^m u^m), \tag{4.1}$$

where

$$I_m^k = I_{k+1}^k I_{k+2}^{k+1} \cdots I_m^{m-1},$$

and similarly for $\hat{I}_m^k$. Using (3.4) with $k = m$, this can be rewritten

$$L^k(\hat{I}_m^k u^m) = I_m^k f^m + \tau_m^k. \tag{4.2}$$

The relative truncation error is related to the local truncation error by

$$\tau_m^k \approx \tau^k - \tau^m, \qquad 1 \leqq k < m \leqq M,$$

where $\approx$ indicates equality of the leading terms in an asymptotic expansion in powers of h_k. (Here we have assumed that the global error can be expanded in a power series in h_k whose coefficients are independent of k.)

This leads to the "dual" interpretation of the multigrid method. Instead of regarding the coarse grid as a device for accelerating convergence of the fine grid equations, we can view the fine grid as a device for calculating the correction τ_m^k to the coarse grid equations. In other words, if grid k is a coarsening of grid M, and if $\hat{I}_M^k$ is the straight injection operator, then τ_M^k is that quantity which has to be added to a (modified) right-hand side, $I_M^k f^M = I_M^k F^M$, to obtain values of the fine grid solution u^M by solving the coarse grid equations. That is, at convergence, the coarse grid solutions are simply projections of the finest grid solution. This would not be true in general, if one solved problems (3.2) independently, without FAS.

We now present a modified version of Brandt's [7], [8] frozen tau technique for continuation. Suppose we have computed a multigrid solution for parameter λ_0 by the method of § 3, and we wish to compute an FAS full multigrid solution at the "nearby" parameter λ_1. Assume we have an initial approximation to the solution on the coarsest grid at λ_1. We can make the coarse grid solution equation "appear" like the (as yet unknown) fine grid solution at λ_1 as follows.

We first determine the relative truncation errors $\tau_M^k(\lambda_0)$, $k = 1, 2, \cdots, M-1$. If $F^k \equiv 0$, as in our model problem (1.2), τ_M^k is, at convergence, simply the modified right-hand side f^k. Otherwise, from (3.3) and induction, τ_M^k is given by

$$\tau_M^k = f^k - I_M^k f^M. \tag{4.3}$$

Before the computation even begins at λ_1, we add $\tau_M^k(\lambda_0)$ to the right sides of the λ_1 equations, for $k = 1, 2, \cdots, M-1$. If λ_1 is close to λ_0, this makes the coarse grid equations at λ_1 locally look like the fine grid "corrected" equations (4.2) with $m = M$. The effect in Fig. 1 is approximately and locally to shift the coarsest grid curve (and all "intermediate" grid curves) close to the fine grid curve. Away from the point where we are computing, the curves corresponding to different grids are not necessarily close to each other.

Of course, this procedure initially ignores the error in $\tau_M^k(\lambda_1)$ due to the change in λ. However, the error so produced will not depend on the high-frequency components, but only on the changes in those components from their values at λ_0. Normally these changes are small compared with the components themselves [7], [8]. Furthermore, the $\tau_M^k(\lambda_0)$ terms are implicitly improved by the multigrid iterations at λ_1, as will be explained in §4.3. More accuracy could be obtained initially by extrapolation in λ using, say, $\tau_M^k(\lambda_0)$ and $\tau_M^k(\lambda_0 - \delta\lambda)$ to better approximate $\tau_M^k(\lambda_1)$.

4.2. Combining multigrid and continuation methods. Despite this improvement, it is clear that we still must vary λ_1 during the multigrid iteration. As we observed in § 1, many workers use different values of λ on different grids. But our shifting of the coarse grid curves allows us to use a very simple algorithm for changing λ_1. We propose correcting λ_1 during the multigrid iteration *only on the coarsest grid.*

We need a method to produce a first approximation u^1 on the coarsest grid at λ_1, and values u^1 and λ_1 for the coarsest-grid correction at λ_1. The pseudo-arc-length continuation method of § 2 will supply both, while being insensitive to limit points. This method interacts with the FAS-FMG algorithm in a natural manner, through the right-hand sides of the coarsest grid equations.

We will now change our notation slightly, and refer to the old and new parameters as $\lambda(s_0)$ and $\lambda(s_1)$, not λ_0 and λ_1. The old parameter $\lambda(s_0)$ is fixed. We allow $\lambda(s_1)$ to vary, but it will be the same on all s_1-grids.

4.3. Algorithm for continuing through limit points. We now summarize the steps of our multigrid continuation algorithm. Then we will explain each of these steps. This algorithm will follow solution curves as they pass through limit points, but in general none of the computed solution points will coincide with any limit point. We will show how to accurately locate limit points in the next subsection.

1. Assume given an initial parameter value λ_0, and an initial coarse grid approximation u^1 to a solution of (1.1) which is not "too close" to a limit point. Let $s_0 = 0$ and define $\lambda(s_0) \equiv \lambda_0$, $u^1(s_0) \equiv u^1$.

2. Use Newton's method (2.1)-(2.2) to obtain an improved solution on the coarsest grid. Note λ_0 is kept fixed.

3. Perform the accommodative FAS full multigrid algorithm, as described in § 3 and Fig. 3. Use a method such as Gauss-Seidel-Newton to relax (smooth) on grids $2, 3, \cdots, M$, and use Newton's method (2.1)-(2.2) to solve on the coarsest grid. The parameter λ_0 is still fixed.

4. When the multigrid algorithm has converged, project the solution from the finest grid to all coarser grids,

$$u^k(s_0) = \hat{I}^k_{k+1} u^{k+1}(s_0), \quad k = M-1, M-2, \cdots, 1. \tag{4.4}$$

5. If $F^k \neq 0$, determine the relative truncation errors $\tau^k_M(s_0)$. First compute

$$\tau^k_{k+1} = f^k - I^k_{k+1} f^{k+1}, \quad k = 1, 2, \cdots, M-1,$$

and overwrite on f^k. Then compute

$$\tau^k_M = \tau^k_{k+1} + I^k_{k+1} \tau^{k+1}_M, \quad k = M-2, M-3, \cdots, 1,$$

and overwrite on f^k.

6. Choose a step length Δs. (See §4.5.) Let $s_1 = s_0 + \Delta s$.

7. Apply the frozen tau technique by adding $\tau^k_M(s_0)$ to the right sides $f^k(s_1) \equiv F^k(s_1)$ of the grid equations (3.4) at levels $k = 1, 2, \ldots, M-1$.

8. Perform the Euler step (2.5)-(2.6) to obtain a first approximation $(u^1(s_1), \lambda(s_1))$ on the coarsest grid at s_1.

9. Use the pseudo-arc-length Newton method (2.10)-(2.13) to improve this approximation on the coarsest grid. The "right-hand side" term $F^1(s_1)$ in

$$G(u^1(s_1), \lambda(s_1)) \equiv L^1(s_1) u^1(s_1) - F^1(s_1), \tag{4.5}$$

which appears in (2.9) or (2.11), was replaced in step 7 by

$$f^1(s_1) \equiv F^1(s_1) + \tau^1_M(s_0). \tag{4.6}$$

10. Perform the multigrid iteration to obtain $\lambda(s_1)$ and $u^M(s_1)$. This is the same as step 3, except: (a) The right sides F^k will have already been modified by step 7; (b) Initial approximations u^k on grids $2, 3, \cdots, M$ are obtained by a method to be described; (c) We use the pseudo-arc-length Newton method (2.10)-(2.13) to correct

on the coarsest grid. The right side $F^1(s_1)$ will have been replaced by the usual FAS right-hand side in (3.3). We obtain not only a new $u^1(s_1)$, but also a new $\lambda(s_1)$.

This procedure yields the solution for two parameter values s_0, s_1. To compute the solution at further values $s_2, s_3, \cdots$, repeat steps 4 to 10, but replace s_i by s_{i+1}. In practice we reuse the storage for s_{i-1} when we start computing at s_{i+1}. Thus only two sets of grids are needed, each set consisting of values of u and f on all levels. Hence the total storage is approximately eight times the number of grid points in Ω^M, times the number of differential equations, plus a small amount for one coarsest-grid Jacobian.

We will now expand on some of these steps.

In steps 2 and 3 we fix λ because we do not have past information to enable us to use pseudo-arc-length continuation. Step 4, projection onto coarser grids, is not strictly necessary, since before it is applied, the FAS method ensures that $u^1(s_0)$ is approximately equal to the right-hand side of (4.4).

Step 5 can be omitted when there are no inhomogeneous terms in the differential equations (e.g., problem (1.2)). Then step 7 can be slightly simplified by *copying* $\tau_M^k(s_0)$ to the right sides of the s_1-grids rather than adding.

For the Euler step 8 and the Newton steps 2 and 9 we must calculate the Jacobian G_u, but this is on the coarsest grid, so there is no storage problem.

Just as in the ordinary pseudo-arc-length method, the Newton step 9 brings us back to the solution curve. Since we have added $\tau_M^1(s_0)$ to the right side of the coarse grid equation, we are brought back (approximately) to the finest grid curve, rather than the coarse grid curve at s_1.

During the multigrid iterations at the new parameter s_1, we return to the coarsest grid for corrections. Just before doing so, we replace the right-hand-side term $f^1(s_1)$ of (4.6) by a new $f^1(s_1)$, as prescribed by the FAS-multigrid method in (3.3). We then use pseudo-arc-length continuation (2.10)–(2.13) to obtain a new $\lambda(s_1)$ and $u^1(s_1)$. The new $\lambda(s_1)$ is used on all finer grid levels until the next coarsest-grid correction.

At first glance it appears that λ_1, since it is changed only on the coarsest grid, incorporates information only from the coarsest grid and the frozen tau term $\tau_M^1(s_0)$. But as the full multigrid iteration at s_1 progresses, λ_1 actually incorporates information about all other grids and all other terms $\tau_M^k(s_0)$, $1 < k < M - 1$, as well. For example, when the finest grid seen so far at s_1 is 2 (i.e., $l = 2$ in Fig. 3), the coarsest-grid correction involves $f^2(s_1)$, by (3.3). But $f^2(s_1)$ has incorporated the term $\tau_M^2(s_0)$. Thus the new λ_1 will also incorporate this term. Meanwhile, the $\tau_M^k(s_0)$ are in effect being replaced by "new" $\tau_M^k(s_1)$. When the iteration at s_1 returns to the coarsest grid for the first time in phase 2 (i.e., when $k = 1$ for the first time after $l = M$ in Fig. 3), the $\tau_M^k(s_0)$ in effect will have been completely replaced by new and better $\tau_M^k(s_1)$. This implicit updating of the frozen tau terms is a consequence of eq. (3.3) and step 7 of our algorithm as stated. No further computations beyond steps 1–10 are required. The frozen tau terms are *explicitly* formed only in Step 5.

To obtain initial approximations $u^k(s_1)$ for grids other than the coarsest, Brandt [8] and Hackbusch [17] suggest starting with the old (s_0) solution at the same level, and correcting it by the difference between the old and new solutions at the next coarser level:

$$u^k(s_1) = u^k(s_0) + \Pi_{k-1}^k(u^{k-1}(s_1) - u^{k-1}(s_0)), \tag{4.7}$$

where Π is the high-order interpolation described in § 3.2. We found that this works well, and it makes possible the error analysis in § 5.

4.4. Locating limit points. The algorithm of the last subsection gave a method for continuing through limit points. In this section we describe how to accurately locate limit points.

If the Jacobian G_u is available on the finest grid, then Keller [20] suggests locating zeros of $\dot{\lambda}^M(s)$. (Here the superscript M means that the the derivative is computed on the finest grid.) If we are performing pseudo-arc-length continuation on a single grid (i.e., $M = 1$), then this derivative is free, since we solve equations (2.5) (with superscripts M added) anyway. But if $M > 1$ then $\dot{\lambda}^M(s)$ requires the Jacobian *on the finest grid*, which we wish to avoid.

A first approach is to use our algorithm of the preceding subsection, and compute the root of $\dot{\lambda}^1(s)$ on the coarse grid where this derivative is computed anyway. Unfortunately, this does not work very well. Although this derivative does become small in the neighborhood of a limit point $P = (u^M(s^*), \lambda(s^*))$, it has so far been too inaccurate in locating P to many decimal places. For example, we used the method of Keller [20] for problem (1.2) with $\alpha = 0$ on a single grid Ω^M, $M = 1$. We located a turning point P by finding s^* for which $\dot{\lambda}^M(s^*) \approx 10^{-12}$. We then repeated the calculation with multigrid (using a method about to be described), in which the finest grid Ω^M had the same mesh size as before. We found a limit point $\tilde{P}$, which was extremely close to P (i.e., the parameters λ agreed to more than 12 digits). But at $\tilde{P}$ we had only $\dot{\lambda}^1(s) \approx 10^{-3}$.

The next approach is to try to imitate the frozen tau method. To do this we would need to replace

$$G(u^M(s_0), \lambda(s_0)) = 0$$

by

$$G(u^1(s_0), \lambda(s_0)) + \tau_M^1(u^M(s_0), \lambda(s_0)) = 0. \tag{4.8}$$

Recall that we obtain (2.3) by differentiating (1.1) with respect to s. If we differentiate (4.8) in the same way, we obtain derivatives of τ with respect to λ and u. The former could be approximated by difference quotients, but it is difficult to find the Jacobian $(\tau_M^1)_u$ without knowing the explicit form of τ_M^1.

Our approach, then is to use a derivative-free method. The algorithm proceeds in two parts. We first apply the method of the preceding section to obtain points P_1, P_2 lying on a solution curve and on either side of the limit point. Then we restart at one of the points, say $P_1 = (u^M(s_1), \lambda(s_1))$, and use $\Delta s \equiv s - s_1$ as the independent variable in a one-dimensional derivative-free optimizer to find an extremum of $\lambda(s)$. Each "outer" iteration of the optimizer requires a complete multigrid solution $(u^M(s), \lambda(s))$ on the finest grid ("inner iteration") with Δs chosen by the optimizer.

We chose the optimizer FMIN, written by R. Brent and appearing in Forsythe, Malcolm and Moler [14]. FMIN uses a combination of golden section search and successive parabolic interpolation. Typically about twelve outer iterations are required to locate the limit point to machine precision.

The only disadvantage of this method (compared with that of Keller [20]) is that it cannot locate limit points which are also inflection points (e.g., on the curve in Fig. 2 for $\alpha = \alpha^*$). However, such points are very rare.

4.5. Implementation details. In this subsection we illustrate our algorithm by applying it to the model reaction-diffusion problem (1.2). In contrast to the description of the previous subsections, most of these implementation details (difference approxi-

mations, smoothing algorithm, etc.) are problem-dependent. They are included to fully characterize the numerical results of § 6.

For simplicity, we consider a square region. This is not, however, a restriction on our method. Removing this restriction would only change some of the implementation details, especially the interpolations. Similarly, our method is not restricted to Dirichlet conditions on $\partial\Omega$.

Let $N_1 > 0$ be the number of coarse grid intervals, and $h_1 = 1/N_1$ be the mesh size on the coarsest grid. The finer mesh sizes are given by letting $N_k = 2N_{k-1}$, and $h_k = h_{k-1}/2$, $k = 2, 3, \cdots, M$. Grid Ω^k is then defined as

$$\Omega^k = \{(x_i^k, y_j^k): x_i^k = ih_k, y_j^k = jh_k, i, j = 0, 1, \cdots, N_k\}, \tag{4.9}$$

for $k = 1, 2, \cdots, M$. We let U_{ij}^k be an approximation to the exact solution $U(x_i^k, y_j^k)$, and u_{ij}^k be the approximation to U_{ij}^k computed by the FAS method. We shall omit the superscript k on U, x, and y whenever possible. For our Dirichlet problem, the difference equations, the right-hand sides $F^k \equiv 0$ and f^k, and the residuals $f^k - L^k u^k$ are defined only on interior grid points, i.e., those for which neither i nor j is 0 or N_k. Naturally, we set the discrete solution to zero at the boundary points.

We use several difference approximations. The second-order approximation to (1.2) is obtained by replacing the Laplacian Δ by the usual five-point approximation Δ_h^5:

$$\Delta_h^5 U_{ij} + \lambda \exp(U_{ij}/(1 + \alpha U_{ij})) = 0. \tag{4.10}$$

Since this approximation is relatively inaccurate, we also consider two fourth order approximations, both obtained from Collatz's Mehrstellen Verfahren. The local truncation error for the nine-point approximation to the Laplacian is

$$\Delta_h^9 U - (\Delta U) = h^2 \Delta(\Delta U)/12 + O(h^4). \tag{4.11}$$

We use the differential equation (1.2) to replace (ΔU) on both left and right sides of (4.11). Then we replace the remaining Δ operator on the right side by the 5-point operator Δ_h^5, to obtain the $O(h^4)$ accurate approximation:

$$\Delta_h^9 U_{ij} + \lambda(I + (h^2/12)\Delta_h^5) \exp(U_{ij}/(1 + \alpha U_{ij})) = 0. \tag{4.12}$$

The second fourth order approximation proceeds as before, up to the last step. But we do not replace the outer Δ operator on the right side of (4.11) by Δ_h^5. Instead we analytically differentiate. In addition to the obvious terms Δu, we obtain terms in U_x^2 and U_y^2. We now replace ΔU by the five-point operator, and U_x and U_y by centered differences $D_{0x}U$ and $D_{0y}U$, respectively. The result is another $O(h^4)$ approximation:

$$\Delta_h^9 U + \lambda \exp(U/\beta)[1 + h^2(\beta^2 \Delta_h^5 U + (1 - 2\alpha\beta)(D_{0x}^2 U + D_{0y}^2 U))/12\beta^4] = 0. \tag{4.13}$$

Here we have defined $\beta = 1 + \alpha U$; and we have omitted subscripts i, j and superscript k on U. This more complicated approximation is less accurate in practice than (4.12), although both are fourth order.

We can obtain a sixth-order accurate approximation to (1.2) by applying *tau extrapolation* [6] to approximation (4.11). We will discuss this in the next subsection.

We now discuss the smoothing (relaxation) method. On the kth grid Ω^k we must solve $J = (N_k - 1)^2$ nonlinear equations, say:

$$g_j(u_1, u_2, \cdots, u_J) = 0, \qquad j = 1, 2, \cdots, J, \tag{4.14}$$

in J unknowns.

On all but the coarsest grid, we use a method called Gauss-Seidel-Newton by Ortega and Rheinboldt [28]. In the jth equation (4.14) we fix all but the jth variable u_j, and apply Newton's method to that equation. In the process, we must replace the "old" iterates $u_j^{(i)}$ by new ones $u_j^{(i+1)}$ in all equations (4.14) as soon as they are available. Were it not for this updating, this method would be equivalent to replacing the full Jacobian for the system (4.14) by its diagonal.

More specifically, the new approximation $u_j^{(i+1)}$ to the jth unknown is given by

$$u_j^{(i+1)} = u_j^{(i)} - g_j(u^{j,i})/(\partial g_j/\partial u_j)(u^{j,i}),$$

where we have set

$$u^{j,i} = (u_1^{(i+1)}, u_2^{(i+1)}, \cdots, u_{j-1}^{(i+1)}, u_j^{(i)}, \cdots, u_J^{(i)}).$$

Together with the full approximation scheme, this method avoids using large Jacobians. Of course, this smoothing algorithm does not work in all problems, see [8].

We used "checkerboard" (or red/black) ordering of the unknowns for relaxation. For the second order approximation, we first relax all the points (x_i, y_j) with $i+j$ even, then all the points with $i+j$ odd. Foerster, Stüben and Trottenberg [13] have shown that this method speeds up the rate of convergence by a factor of two for Poisson's equation with Gauss-Seidel smoothing, compared with lexicographic ordering.

For the nine-point approximations, we replaced the two colors (red and black) with four colors [15], [35]. Consider a "fundamental square" with corners $P_{ij} \equiv (x_i, y_j)$, i or $j = 1$ or 2. Each corner is given a different color. Then the colors are extended to the whole grid by periodicity: P_{ij} has the same color as P_{lm} if and only if $i \equiv l \bmod 2$ and $j \equiv m \bmod 2$. For one iteration we relax all points with the color of P_{11}, then P_{22}, P_{12}, and P_{21}. Computations by Stüben and Trottenberg [35] have shown that, for Poisson's equation, this ordering produces better smoothing properties than any other commonly used smoothing method for the fourth order Mehrstellen Verfahren approximation.

For the projection operator $\hat{I}_{k+1}^k$ on approximate solutions we use simple injection:

$$u_{ij}^k \leftarrow \hat{I}_{k+1}^k u_{2i,2j}^{k+1} \equiv u_{2i,2j}^{k+1}.$$

For the projection operator I_{k+1}^k on residuals we use full weighting, given in stencil form as

$$I_{k+1}^k \equiv \frac{1}{16}\begin{pmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{pmatrix}_{k+1}. \tag{4.15}$$

For the second order approximation, it is sufficient to use bilinear interpolation I_{k-1}^k for corrections. This is easy to program. For the interpolation operator Π_{l-1}^l of solutions used in phase 1, it is necessary to use bicubic interpolation. We also used bicubic interpolation for both interpolation operators in conjunction with our fourth-order approximations, as did Schaffer [30]. Our sixth-order approximation requires biquintic interpolation for Π.

When the region Ω is, in some coordinate system, equal to the cross product of intervals, we can easily implement these high-order interpolations. We used de Boor's [4] program SPLI2D, which computes the tensor product of one-dimensional splines, using not-a-knot end conditions. It is sufficient to use SPLI2D with bicubic and biquintic interpolation. We placed the knots at the points x_i or y_j. However, bicubic (biquintic) splines require the number of intervals N_1 on the coarsest grid to be at least 3 (5).

Since we wished to use coarse grids that were as small as possible, we also used SPLI2D with biquadratic and biquartic interpolation for the cases $N_1 = 2$ and $N_1 = 4$, respectively. This requires us to place the knots midway between the points x_i or y_j, as well as at the endpoints 0 or 1. Further details are in de Boor [4].

For our second and fourth order approximations we chose $N_1 = 4$, but $N_1 = 3$ also proved acceptable in our computations. Some workers (e.g., [35]) have used $N_1 = 2$ (one interior grid point) on Poisson's equation. We found that this produced very sensitive behavior, even divergence, near limit points of problem (1.2). For our sixth order approximation (to be described shortly) we found that using $N_1 < 5$ caused a slowdown in the rate of convergence, since the interpolation is not sufficiently accurate on the coarsest grid. Therefore we used $N_1 = 8$ for comparison with other results.

The spline interpolations are relatively expensive compared to the cost of the relaxation sweeps. But the cost of the former is still linear in the number of grid points, since SPLI2D exploits the band structure of one-dimensional spline interpolation. Fortunately the number of relaxation sweeps is about three times the number of interpolations. The second order method requires only $M - 1$ bicubic interpolations. The rest are bilinear.

In the use of Newton's method for continuation, it is customary (e.g., [24], [32]) to economize in the computation of the Jacobian G_u. This is done by not always recomputing it after computing a new iterate $u^{(i)}$. In contrast, our computation of the Jacobian on the coarsest grid is so cheap that we recompute it after every iteration. For the convergence tolerance ε_1 on the coarsest grid, we solve to nearly machine accuracy, say

$$\|u_{i+1}^1 - u_i^1\| + |\lambda_{i+1} - \lambda_i| < 15\epsilon(\|u_i^1\| + |\lambda_i|),$$

where i denotes the iteration number, ε is machine epsilon (the smallest positive number for which $1 \oplus \varepsilon \neq 1$), and $\|\cdot\|$ is the Euclidean norm.

The storage required for the Jacobian is small. Typically $N_1 = 4$, so we need store only a 9 by 9 Jacobian. We used the banded solver DGBCO, DGBSL (with pivoting) in LINPACK [12] for our computations.

Our step-size control is based on the convergence behavior of Newton's method on the coarsest grid. (See Rheinboldt [29] for another algorithm.) We choose an initial step size $s_1 - s_0$ based on our knowledge of the problem. To determine all other step sizes $s_i - s_{i-1}$, $i > 1$, we first take a trial Euler step from s_{i-1} to s_i, using as trial step size the old step size $s_{i-1} - s_{i-2}$. We then count the number of (pseudo-arc) Newton iterations performed on the coarsest grid until the first switch to the next finer grid. If the residual norms of the iterates do not decrease, or if more than six iterations are required, we multiply the trial step size by one-third and restart at s_{i-1}. If five or six iterations are performed, we accept the trial step. If three or fewer (resp. four) iterations are required, we complete this step, and set the trial steplength $s_{i+1} - s_i$ to two (resp. 1.5) times our currrent steplength. (These figures for the number of Newton iterates depend slightly on the machine precision.)

This strategy can also be used for single-grid methods. Its advantage here is the low cost of the coarse grid operations, which simultaneously approximate the fine grid equations because of the frozen tau term. This step control is not used when locating limit points, as in the previous subsection, since the root-finder supplies the step control.

For the parameters η and δ which control switching between grids we used $\delta = 0.3$, $\eta = 0.25$.

An objection is raised by T. F. Chan [10] to our use of the bordering algorithm (2.10)-(2.13). Keller [21] shows that, in the absence of roundoff error, this procedure is valid in the neighborhood of limit points. Chan asks if this is also true in the presence of roundoff error. To resolve this question, R. Schreiber [33] suggested replacing the bordering algorithm (on the coarsest grid) by a full matrix solver for the system (2.9). This solver takes no advantage of the the structure of the matrix A in (2.9), but this imposes only a small storage penalty. We obtained exactly the same results as before.

4.6. Tau extrapolation. We now explain how tau extrapolation [6] is implemented to increase the order of accuracy of the approximation to problem (1.2) from four to six. We do not implement this exactly as in [6], and we do not claim that our method is optimal.

We retain phase one of the FMG algorithm (before the finest grid is reached) unaltered; that is, we do not apply tau extrapolation in phase one. (The frozen tau technique has in effect already produced an extrapolation here.) Upon reaching phase two, we smooth on the finest grid as before, until the convergence slows. This is the last time smoothing is done on the finest grid. Throughout we use sixth order interpolations I_k^{k+1}, so the order of the interpolation error is the same as the order of the truncation error.

When the algorithm switches from the finest grid M to grid $M-1$, we form the right-hand side f^{M-1} as in (3.3). Then we modify it by forming

$$f_{\text{new}}^{M-1} \leftarrow (1-2^{-p})^{-1}[f^{M-1} - I_M^{M-1}F^M] + I_M^{M-1}F^M, \tag{4.16}$$

with $p=4$. (Note $F^M = 0$ for problem (1.2)). The expression in square brackets is, by (4.3), the relative truncation error τ_M^{M-1}. Multiplying it by the factor in parenthesis produces the local truncation error τ^{M-1}, to leading order terms. Therefore, solving $L^{M-1}u^{M-1} = f_{\text{new}}^{M-1}$ produces an $O(h^6)$ approximation on grid $M-1$. The right-hand sides of all coarser grids $M-2, M-3, \cdots, 2, 1$ are modified as in (3.3) but not as in (4.16). Finally, our sixth order interpolation of the correction $u^M - u^{M-1}$ produces a sixth order approximation on grid M.

During the rest of phase two of the multigrid algorithm, we do not smooth on the finest grid. That would only force the finest grid solution to satisfy the original (fourth order) equations.

Of course, tau extrapolation is the same as applying deferred correction once. However, we need not explicitly compute the leading terms of the local truncation error, nor form large order Jacobians.

After completing this paper we learned of the work of Schaffer [31], which extends the tau extrapolation method as follows. The procedure given above requires a minimum of two grids and applies deferred correction once. Schaffer extends this to *iterated* deferred corrections. For example, to attain $O(h^8)$ accuracy with an $O(h^4)$ basic scheme requires a minimum of three grids. Clearly this method will require less work to obtain the same $O(h^8)$ accuracy than Richardson extrapolation. Schaffer gives numerical results only for linear problems, but we believe that his method could be extended to our problem if more accurate interpolations are used. As he points out, the second-order-accurate projection operator (4.15) still suffices, due to a fortunate cancellation of errors.

5. Convergence. In this section we outline a local convergence proof for our algorithm. Further details will appear elsewhere. Our analysis is based on the techniques

of Hackbusch [16], [17]. We assume the reader is familiar with these papers, and use the notation in them with only slight changes.

Throughout we assume that all functions are as smooth as necessary. We shall also ignore round-off errors since they are dominated by truncation errors.

Let us first recall the linear theory for a fixed parameter λ. Hackbusch shows that the interpolation, projection and smoothing operators satisfy a "smoothing property" and an "approximation property". The analysis proceeds stepwise from two levels to full multigrid.

The two-level multigrid method has iteration matrix

$$M_k^{k-1} = S_k^{\nu_2}(I_k - I_{k-1}^k L_{k-1}^{-1} I_k^{k-1} L_k) S_k^{\nu_1},$$

where I_k is the identity matrix on level k and S_k is the smoothing iteration matrix on level k. Under appropriate assumptions he then shows that the norm of the two-level matrix is less than one. The iteration matrix M_M^1 for the "cycle C" multigrid algorithm (which starts with an approximation on the finest level) is then found to be a perturbation of the two-grid iteration matrix. By recursion Hackbusch shows that this matrix has norm less than one. It is characteristic of the multigrid method that this bound is independent of the grid level k. Finally, an inductive proof can be given to show that the full multigrid method will produce approximations u^M on the finest level whose "iteration error" (the difference between u^M and the solution U^M of the discretized equations) is bounded by the local truncation error.

Hackbusch provides a convergence proof in the nonlinear case (with fixed λ), but not for the FMG full approximation scheme. Instead he proves convergence for what he calls the "multigrid method of the second kind". To do this, he proves a contraction property which corresponds to the boundedness of the norm of M_M^1 by a quantity less than one in the linear case.

Let $\phi_k(u_j^k, f^k)$ be a nonlinear iteration function (the multigrid algorithm) on level k which produces a new iterate u_{j+1}^k from old iterates u_j^k and right hand side f^k. For u_j^k in a sufficiently small neighborhood of the solution U^k of the difference equation (3.2), he requires

$$\|u_{j+1}^k - U^k\| \leqq \rho \|u_j^k - U^k\|, \tag{5.1}$$

with $\rho < 1$, $k = 1, 2, \cdots, M$. Again ρ is independent of the level k.

The contraction property (5.1) is appropriate for "phase one" of the full multigrid algorithm (§ 3) when a first approximation on the finest level is being obtained. After that, for the full approximation scheme a more appropriate contraction property is

$$\|u_{j+1}^k - I_M^k(U^M)\| \leqq \rho' \|u_j^k - I_M^k(U^M)\|,$$

where $0 < \rho' < 1$. For $k = M$ this is identical to (5.1); for other k it can be deduced from (5.1). From this we can show the convergence of the FAS multigrid method for fixed λ, provided the initial guesses are sufficiently close to the solution U^M.

Now we discuss the differences between our approach to the continuation problem and that of Hackbusch. First, we parametrize our solutions in terms of pseudo-arc-length instead of the natural parameter λ. That is, given differential equation (1.1), we parametrize a solution branch as in § 4.3 by choosing pseudo-arc-length points $s_0, s_1, s_2, \cdots$, and letting the exact solution path

$$(U(s), \Lambda(s)), \tag{5.2}$$

and approximate solutions

$$(U^k(s), \lambda(s)), \quad k = 1, 2, \cdots, M, \tag{5.3}$$

depend on the pseudo-arc-length parameter s. Assume that the solution path (5.2) encounters no bifurcation points and only simple limit points, and that the starting point s_0 is such that path (5.2) at s_0 is not "too close" to a limit point. Then the results of Keller [18] and Decker and Keller [11] show that the Jacobian G_u at s_0, and the augmented Jacobian matrices (2.9) at $s = s_1, s_2, \cdots$, are nonsingular in suitable neighborhoods of the exact solution (5.2). Hence our method will not have any difficulties at limit points.

Our analysis proceeds in three steps. We examine the multigrid convergence at s_0, where λ_0 is fixed. Then we examine the errors in the initial approximations at s_1. Finally we obtain bounds on the errors in the multigrid method at s_1, where λ varies. For all other intervals $[s_i, s_{i+1}]$ we repeat the last two steps.

The proof of convergence at s_0 was already mentioned above in the FAS modification of Hackbusch's proof. The result is that

$$\|u^M(s_0) - U^M(s_0)\| \leqq C_1 h_M^p,$$

where C_1 is a constant independent of the the level k and of the mesh spacing, and p is the order of accuracy of the difference approximation.

For step two we estimate the error in the initial approximations at s_1 . Hackbusch [17] analyzed both the starting procedure (4.7) and the frozen tau method when λ is the parameter. Since he did not combine the two methods, he concluded that the frozen tau method was unsatisfactory since it did not produce sufficiently accurate initial approximations. By combining his results, and the results of Decker and Keller [11] for Newton's method, we can show that the initial approximations $(u^k(s_{i+1}), \lambda(s_{i+1}))$ have truncation error $O[(s_{i+1} - s_i)h_k^p]$.

For step three we must analyze the multigrid convergence at s_1, when both u and λ vary. Let the composite vector $\tilde{u}^k$ be the vector u^k with λ appended. For our composite vector we can prove a contraction property by induction on grid levels. For $k = 1$ this is certainly true since we use only Newton's method. The most difficult case is $k = 2$. All other levels introduce no more difficulties than the case of fixed λ. This is because the coarse grid correction for two levels $k-1$ and k $(k \geqq 3)$ involves no change in λ. The result is that

$$\|\tilde{u}^M(s_1) - \tilde{U}^M(s_1)\| \leqq C_2(s_1 - s_0)h_M^p,$$

where again C_2 is independent of the level and mesh spacing.

The main difficulty in these arguments is assuring that the result of each step of the algorithm lies in a sufficiently small neighborhood for the iteration of the next step to be well defined. Hackbusch has proven many of these results.

6. Numerical results. In this section we present results of our computations of the limit point locations for two different parameter values α in problem (1.2). We have also used this method in more realistic problems, e.g., the Taylor vortex problem [3]. The smoothing algorithm we used for that problem is alternating zebra [35], a variant of alternating line (or block) Gauss-Seidel-Newton [28, p. 225, eq. (39)].

We first computed the location of the limit point for $\alpha = 0$ (see Fig. 2). This is a good test problem because the very accurate results of Meis, Lehmann and Michael [23] can be used for comparison. Meis, Lehmann and Michael computed solution

points using the method described in §1, with second order approximation (4.10). They obtained three points on the solution curve near the limit point, then fitted a parabola to these points to find the location of the limit point. These calculations were done on each of a sequence of grids whose smallest mesh size was $h = 1/4, 1/8, \cdots, 1/128$. Repeated Richardson extrapolation (up to $O(h^6)$) was then used to better approximate the location of the limit point of problem (1.2) with $\alpha = 0$. The location thus obtained is:

$$\lambda_{cr} = 6.808124, \qquad \|u(\lambda_{cr})\|_\infty \equiv \|u_{cr}\|_\infty = 1.39166.$$

The maximum of u always occurs at the center $(x, y) = (1/2, 1/2)$ of Ω.

We used our methods on essentially the same grids: the second-order scheme (4.10), the fourth order schemes (4.12) and (4.13), and a sixth order-scheme obtained by applying tau extrapolation to (4.12). We show the results in Table 1. The number of intervals on the finest mesh is N_M; hence the smallest mesh size is $h_M = 1/N_M$. For each method, the values labelled ∞ were obtained by repeated Richardson extrapolation to $O(h^8)$. Thus, the extrapolated values for the second (respectively fourth, sixth)-order method were obtained by three (resp. two, one) Richardson extrapolations from the four (resp. three, two) values immediately above. Each column was calculated independently of the others, so these results serve as a severe check on each other. To our knowledge, the existence of asymptotic expansions for the global error (i.e., the validity of Richardson extrapolation) for this problem has not been proved, but these computational results provide almost certain evidence that one exists for each scheme.

From Table 1 we observe, as expected, that the sixth-order scheme is much more accurate than the fourth-order or the second-order schemes. More striking is the relationship between limit point locations on different grids. Our results show that the limit points on coarser grids may lie on either side of the limit point of the finest grid, depending on the approximation chosen. In fact, the value of λ_{cr} at the limit point increases monotonically (with decreasing grid size) for schemes (4.10) and (4.12), and decreases for the others. The maximum solution value $\|u_{cr}\|_\infty$ increases for all approximations except the fourth-order scheme (4.13). Clearly, our method does not depend on the orientation of limit points on different grids, except for starting points. Thus, if we use the second-order scheme with $M = 2$, $h_1 = 1/16$, and $h_2 = 1/32$, then we must not use $\lambda = 6.804$, for example, as a starting point.

We give the results for $\|u\|_\infty$ to fewer digits than those for λ, because of the geometry of the limit point. Locally, the solution curve in the $\lambda - \|u\|_\infty$ plane is a quadratic in the neighborhood of a limit point (u^*, λ^*), that is, $\|u - u^*\| = O(|\lambda - \lambda^*|^{1/2})$ (see, e.g., Moore and Spence [27]). Thus λ^* can be known (at best) to the machine precision, but $\|u^*\|_\infty$ can be known only to the square root of the machine precision. Our computer (VAX 11/750) has about 17 decimal digits of precision. Thus we seem to obtain 7 correct decimal digits of $\|u_{cr}\|_\infty$ and about 10 correct digits of λ_{cr}.

The second order results and the extrapolated results agree with those in [23] to the seven and six digits they gave for λ_{cr} and $\|u_{cr}\|_\infty$, respectively. The fourth order results (4.12) for $N = 16$ agree with those of [27] to the 11 digits they gave for λ. For $\|u_{cr}\|_\infty$ our eight-digit results agree with the first eight of their eleven digits.

Tables 2 and 3 show results similar to those of Table 1, but for the case $\alpha = 0.2$ in problem (1.2). Here there are two limit points. For the upper limit point we can partially compare our results with those of Mittlemann [25] for $N_M = 32$. He used second-order approximation (4.10) with his generalized inverse iteration multigrid method [25], [26], and gave results to four digits. Rather than locate the upper limit point, he showed successive values of the parameter λ on the solution path near this

TABLE 1
Location of limit point of (1.2) *with* $\alpha = 0$.

		Order and Scheme			
	N_M	2 (4.10)	4 (4.12)	4 (4.13)	6 [extrap. (4.12)]
λ_{cr}	16	6.80217409563	6.80808657467	6.80830691585	6.80812517811
	32	6.80665272920	6.80812207169	6.80813582072	6.80812443571
	64	6.80775749456	6.80812427588	6.80812513486	6.80812442280
	128	6.80803275282	6.80812441342	6.80812446710	6.80812442259
	∞	6.80812442263	6.80812442259	6.80812442258	6.80812442259
$\|u\|_\infty$	16	1.3888573	1.3916567	1.3917381	1.3916593
	32	1.3909601	1.3916609	1.3916661	1.3916612
	64	1.3914859	1.3916612	1.3916615	1.3916612
	128	1.3916174	1.3916612	1.3916612	1.3916612
	∞	1.3916612	1.3916612	1.3916612	1.3916612

TABLE 2
Location of lower limit point of (1.2) *with* $\alpha = 0.2$.

		Order		
	N_M	2	4 (4.12)	6
λ^l_{cr}	16	9.12131236518	9.13630924484	9.13638435052
	32	9.13263701343	9.13637838604	9.13638298751
	64	9.13544784102	9.13638268079	9.13638296698
	128	9.13614927051	9.13638294880	9.13638296666
	∞	9.13638296666	9.13638296667	9.13638296666
$\|u\|_\infty$	16	2.8756967	2.8857321	2.8858002
	32	2.8832818	2.8857962	2.8858004
	64	2.8851712	2.8858001	2.8858004
	128	2.8856430	2.8858003	2.8858004
	∞	2.8858003	2.8858003	2.8858004

TABLE 3
Location of upper limit point of (1.2) *with* $\alpha = 0.2$.

		Order		
	N_M	2	4 (4.12)	6
λ^u_{cr}	16	7.08025536111	7.10152536891	7.10195697086
	32	7.09656018055	7.10187720721	7.10190065819
	64	7.10056845538	7.10189761734	7.10189897803
	128	7.10156658312	7.10189886674	7.10189894998
	∞	7.10189894893	7.10189894958	7.10189894953
$\|u\|_\infty$	16	18.207497	18.195894	18.192661
	32	18.195850	18.192933	18.193740
	64	18.193507	18.192778	18.192767
	128	18.192951	18.192768	18.192768
	∞	18.192768	18.192767	18.192768

point. These values of λ, in increasing order of $\|u\|_\infty$, are 7.103, 7.097, 7.096, 7.098, 7.104. Our value of λ^u_{cr} for $N_M = 32$ in the first column of Table 3 agrees well with these. Unfortunately, he did not supply values of $\|u\|_\infty$.

Our starting guess for the solution values in Tables 1 and 2 was zero. For Table 3 our starting guess was $u = 12$ in the interior of the coarse grid, and $u = 0$ on the boundary. Of course, we obtained the same results by starting on the lower branch and continuing to the upper limit point.

In all cases we iterate the multigrid algorithm until two successive solutions are sufficiently close:

$$\|u_{i+1}^M - u_i^M\| < \varepsilon \|u_i^M\|.$$

Here the subscript denotes the iteration number and ϵ is a small number related to the machine precision. Normally we would iterate only until the norm of the residual is less than the norm of the relative truncation error τ_M^{M-1}. But we wished to ensure that the "convergence error" was much less than the truncation error so that we could perform Richardson extrapolation.

The work required to reduce the l_2 norm of the residuals by a factor of 10^{-12} was never more than 35 work units, and usually about 25 units (after the first step s_0). A work unit (following Brandt) is the work required to relax all the grid points once on the finest grid. It does not include the work required for interpolations or projections. The work required to reduce the residuals to the level of the truncation error was 4–10 work units after the first step, depending on the length of the step $s_i - s_{i-1}$. Computing one solution of problem (1.2) on a 129 by 129 grid ($N_M = 128$) took approximately 20 minutes on a DEC VAX 11/750. This reduced the l_2 norm of the residuals by a factor of 10^{-12}. (This machine is about 70% of the speed of a VAX 11/780.) We know from § 2 that the pseudo-arc-length Newton method eliminates slow convergence or divergence near limit points when used with a "single grid" method. Similarly, it is experimentally observed that our method eliminates convergence difficulties with the multigrid method at simple limit points. This is confirmed by the experimental observation that *the multigrid convergence rate is the same near limit points as away from them.* An exception to this observation occurs at the starting point s_0, where we cannot use the pseudo-arc-length procedure during the multigrid iteration. If this point is too close to the limit point, the rate of convergence will of course be slow, but *only* at s_0.

Acknowledgments. We wish to thank Achi Brandt for technical discussions. He discovered this method before us but has not published it. We are also grateful for the use of computer time on the Fluid Dynamics VAX and the Applied Math-IBM 4341 at Caltech. The former is supported by the Office of Naval Research, and the latter is supported by the IBM Corporation. Finally, we thank the referees for carefully reading the manuscript.

REFERENCES

[1] J. P. ABBOTT, *An efficient algorithm for the determination of certain bifurcation points*, J. Comput. Appl. Math., 4 (1978), pp. 19–27.

[2] R. E. BANK AND T. F. CHAN, *PLTMGC: A multigrid continuation package for solving parametrized nonlinear elliptic systems*, Report 261, Dept. of Computer Science, Yale University, New Haven, CT, 1983.

[3] J. H. BOLSTAD AND H. B. KELLER, *Computation of anomalous modes in the Taylor experiment*, to appear.

[4] C. DE BOOR, *A practical guide to splines*, Springer Verlag, New York, 1978.

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

[6] A. BRANDT AND N. DINAR, *Multigrid solutions to elliptic flow problems*, in Numerical Methods for Partial Differential Equations, S. Parter, ed., Academic Press, New York, 1979, pp. 53–147.

[7] A. BRANDT, *Multigrid solvers on parallel computers*, in Elliptic Problem Solvers, M. Schultz, ed., Academic Press, New York, 1981, pp. 39–83.

[8] A. BRANDT, *Guide to multigrid development*, in Multigrid Methods, Lecture Notes in Mathematics 960, W. Hackbusch and U. Trottenberg, eds., Springer Verlag, New York, 1982, pp. 220-312.

[9] T. F. CHAN AND H. B. KELLER, *Arc-length continuation and multigrid techniques for nonlinear elliptic eigenvalue problems*, this Journal, 3 (1980), pp. 173-194.

[10] T. F. CHAN, personal communication.

[11] D. W. DECKER AND H. B. KELLER, *Path following near bifurcation*, Comm. Pure Appl. Math., 34 (1981), pp. 149-175.

[12] J. J. DONGARRA, J. R. BUNCH, C. B. MOLER AND G. W. STEWART, LINPACK *Users' Guide*, Society for Industrial and Applied Mathematics, Philadelphia, 1979.

[13] H. FOERSTER, K. STUEBEN AND U. TROTTENBERG, *Non-standard multigrid techniques using checkered relaxation and intermediate grids*, in Elliptic Problem Solvers, M. Schultz, ed., Academic Press, New York, 1981, pp. 285-300.

[14] G. FORSYTHE, M. MALCOLM AND C. MOLER, *Computer Methods for Mathematical Computations*, Prentice-Hall, Englewood Cliffs, NJ, 1977.

[15] W. HACKBUSCH, *On the multigrid method applied to difference equations*, Computing, 20 (1978), pp. 291-306.

[16] W. HACKBUSCH, *Multigrid convergence theory*, in Multigrid Methods, Lecture Notes in Mathematics 960, W. Hackbusch and U. Trottenberg, eds., Springer Verlag, New York, 1982, pp. 177-219.

[17] W. HACKBUSCH, *Multigrid solution of continuation problems*, in Iterative Lösung Nichtlinearer Gleichungssysteme, Lecture Notes in Mathematics 953, R. Ansorge, T. Meis and W. Törnig, eds., Springer Verlag, New York, 1982.

[18] H. B. KELLER, *Approximation methods for nonlinear problems with application to two-point boundary value problems*, Math. Comp., 29 (1975), pp. 464-474.

[19] H. B. KELLER, *Numerical solutions of bifurcation and nonlinear eigenvalue problems*, in Applications of Bifurcation Theory, P. Rabinowitz, ed., Academic Press, New York, 1977, pp. 359-384.

[20] H. B. KELLER, *Global homotopies and Newton methods*, in Recent Advances in Numerical Analysis, C. de Boor and G. Golub, eds., Academic Press, New York, 1978, pp. 73-94.

[21] H. B. KELLER, *Practical procedures in path following near limit points*, in Computing Methods in Applied Sciences and Engineering, R. Glowinski and J. Lions, eds., North-Holland, New York, 1982.

[22] M. LENTINI AND H. B. KELLER, *The von Karman swirling flows*, SIAM J. Appl. Math., 38 (1980), pp. 52-64.

[23] T. MEIS, H. LEHMANN AND H. MICHAEL, *Application of the multigrid method to a nonlinear indefinite problem*, in Multigrid Methods, Lecture Notes in Mathematics 960, W. Hackbusch and U. Trottenberg, eds., Springer Verlag, New York, 1982, pp. 545-557.

[24] R. MEYER-SPASCHE AND H. B. KELLER, *Computations of the axisymmetric flow between rotating cylinders*, J. Comp. Phys., 35 (1980), pp. 100-109.

[25] H. D. MITTLEMANN, *Multigrid methods for simple bifurcation problems*, in Multigrid Methods, Lecture Notes in Mathematics 960, W. Hackbusch and U. Trottenberg, eds., Springer Verlag, New York, 1982, pp. 558-575.

[26] H. D. MITTLEMANN AND H. WEBER, *Multigrid solution of bifurcation problems*, International Multigrid Conference, Copper Mountain, CO, 1983.

[27] G. MOORE AND A. SPENCE, *The calculation of turning points of nonlinear equations*, SIAM J. Numer. Anal., 17 (1980), pp. 567-576.

[28] J. ORTEGA AND W. RHEINBOLDT, *Iterative Solution of Nonlinear Equations in Several Variables*, Academic Press, New York, 1970.

[29] W. RHEINBOLDT, *Solution fields of nonlinear equations and continuation methods*, SIAM J. Numer. Anal., 17 (1980), pp. 221-237.

[30] S. SCHAFFER, *High order multigrid methods to solve the Poisson equation*, in Multigrid Methods, H. Lomax, ed., NASA Conference Publication 2202, 1982, pp. 275-284.

[31] S. SCHAFFER, *Higher order multigrid methods*, Math. Comp., 43 (1984), pp. 89-115.

[32] R. SCHREIBER AND H. B. KELLER, *Driven cavity flows by efficient numerical techniques*, J. Comp. Phys., 49 (1983), pp. 310-333.

[33] R. SCHREIBER, personal communication.

[34] A. SPENCE AND B. WERNER, *Non-simple turning points and cusps*, IMA J. Numer. Anal., 2 (1982), pp. 413-427.

[35] K. STUEBEN AND U. TROTTENBERG, *Multigrid methods: Fundamental algorithms, model problem analysis and applications*, in Multigrid Methods, Lecture Notes in Mathematics 960, W. Hackbusch and U. Trottenberg, eds., Springer Verlag, New York, 1982, pp. 1-176.

[36] R. K. H. SZETO, *The flow between rotating coaxial disks*, Ph.D. Thesis, California Institute of Technology, Pasadena, CA, 1978.