Abstract

PLTMGC is a program package for solving nonlinear elliptic systems that have explicit dependence on a scalar parameter. In addition to being able to compute solutions for fixed parameter values, it can be used to solve the linear eigenvalue problem, trace solution branches, locate singular points (simple turning points and bifurcation points) and switch branch at simple bifurcation points. A multi-grid continuation approach is employed in which a continuation procedure is used to follow the solution curve on the coarsest grid and a multi-grid algorithm is used to refine the solution at selected points using an adaptive mesh refinement strategy. Some numerical examples illustrating the performance of the package are given.

Keywords: *Continuation methods, multi-grid algorithms, nonlinear elliptic systems, eigenvalue problems, turning point, bifurcation, adaptive mesh refinement.*

PLTMGC: A Multi-Grid Continuation Program for Parameterized Nonlinear Elliptic Systems

Randolph E. Bank[1]
Tony F. Chan[2]

Research Report YALEU/DCS/RR-261
December, 1983.

[1]Dept. of Math., U.C. San Diego, La Jolla, Calif.

This author's work was supported by the Office of Naval Research under contract N00014-82-K-0197.

[2]Dept. of Computer Science, Yale Univ., Box 2158, Yale Station, New Haven, CT 06520.

This author's work was supported in part by the Dept. of Energy under grant DE-AC02-81ER10996 and by the Army Research Office under grant DAAG-83-0177.

1. Introduction

PLTMGC is a program package for solving two dimensional nonlinear elliptic boundary value problems of the form:

$$\begin{aligned} -\nabla d\nabla u + f(x,y,u,\nabla u,\lambda) &= 0 \qquad \text{in } \Omega, \\ u &= g_1(x,y) \qquad \text{on } \partial\Omega_1, \\ (d\nabla u)\,.\,n &= g_2(x,y,u,\lambda) \quad \text{on } \partial\Omega - \partial\Omega_1. \end{aligned} \tag{1}$$

The package is specifically designed to take into account the explicit dependence of the system (1) on the scalar parameter λ. In this paper, we shall denote the system (1) by

$$G(u,\lambda) = 0, \tag{2}$$

where $u \in B$ (a real Banach space), $\lambda \in R$, and G is a continuously differentiable operator mapping B x R into B.

There are at least three situations in which it is important to consider the parameter λ explicitly. The first case is when the system (1) is a mathematical model for a physical problem, in which u may represent a physical field variable (e.g. flow field, structural displacement) and λ may be related to a physical parameter (e.g. Reynold's number, load on a structure) and one is interested in the dependence of u on λ. The second class of problems are linear and nonlinear eigenvalue problems [3] which can be cast in the form of (1). The third situation arises in the application of homotopy continuation methods for obtaining good initial guesses to highly nonlinear problems [24]. For example, consider the following nonlinear elliptic system:

$$Lu + N(u) = 0 \ , \tag{3}$$

where L is a linear elliptic operator and N is a highly nonlinear operator. Simple Picard-type iteration or even direct application of Newton's method to (3) may fail unless an extremely good initial guess is available. Assuming that the problem $Lu = 0$ is easier to solve, one may consider introducing an artifical parameter λ in (3) to obtain the following modified system:

$$G(u, \lambda) \equiv Lu + \lambda N(u) = 0$$

and slowly increasing λ from 0 to 1, using an old solution as initial guess for the next.

Thus, solving the system (2) may consist of determining the dependence of the solution $u(\lambda)$ on the parameter λ, i.e. in tracing the solution branches $[u(\lambda), \lambda]$ of (2), in addition to determining the solution u at some given value of λ or $||u||$ (target points). (Throughout this paper, $||u||$ denotes the usual L_2 norm of u, namely, $\int_\Omega u^2\, dx\, dy$.) When the operator G is nonlinear in u and λ, this can be accomplished numerically by applying an approximate Newton method to (2)

for a fixed value of λ, which makes use of the Jacobian matrix $G_u(u,\lambda)$. However, the solution branches often display very interesting and complicated behavior, among which are existence of multiple solutions and singular points (where $G_u(u,\lambda)$ is singular) known as *turning points* (where the solution branch bends back on itself) and *bifurcation points* (where two or more solution branches cross.) Straightforward application of Newton's method to (2) encounters difficulties near these singular points. To overcome these difficulties, a path following continuation method [2, 29, 34, 39] is usually employed. These continuation methods are designed to trace past turning points and can be used to switch branches at bifurcation points.

PLTMGC is a derivative of the program package PLTMG [7] which solves nonlinear elliptic systems (with no parameter dependence) of a similar form to (1). It employs a finite element discretization of (1) based on C^0 piecewise linear triangular finite elements and uses a multi-grid approach which obtains the solution on the finest grid by iterating through a hierarchy of coarser grids. The grids are obtained by adaptive mesh refinement techniques which adaptively refine the mesh only in regions where the solution error is large. In order to use PLTMGC, the user has to supply a coarse grid on which the continuation procedures are applied. At selected points, the multi-grid procedures are called on to adaptively refine the grid and obtain the solution on the fine grids.

PLTMGC differs from other continuation program packages [31, 41, 43] in that we treat directly partial differential equations and exploit the inherent structures of the PDE in the continuation procedures through the use of multi-grid techniques. We feel that this is crucial for the efficiency of the overall method. In some sense, PLTMGC represents our attempt to cohesively integrate algorithms and ideas from several divergent sources. While many of the algorithms in PLTMGC are taken directly from the existing literature, we have also developed many new ones which are especially suited to PLTMGC and which contribute to the efficiency and robustness of the package. Among these are: a new pseudo-arclength parameterization that facilitates computing target values in λ and $||u||$, a new robust predictor, multi-grid deflation techniques for ensuring numerical stability near singular points and a new algorithm for locating simple turning points and bifurcation points. Our goal in this paper is to present only an overview of PLTMGC which summarizes our (current) view as to how these various ideas *should* be synthesized. To remain true to this goal (and to keep the paper of manageable size) we have purposely omitted many technical details which are obviously very important to a successful implementation. Where possible, we provide references to more complete explanations.

The remainder of the manuscript is organized as follows. In Section 2, we summarize continuation algorithms, multi-level algorithms and adaptive mesh refinement techniques. In Section 3, we describe how these ideas are merged in PLTMGC. Section 4 contains some numerical examples, illustrating the procedures of Section 3. Section 5 contains some concluding remarks and possible extensions.

2. Continuation and Multi-Level Algorithms

In this section, we first review the major components of a basic continuation procedure. Next we summarize those aspects of the multi-level iterative method which are most important to the continuation process. Since much of the analysis of these methods appears elsewhere, (see for example [9, 13, 15, 25, 27, 29, 38]), our exposition is mainly descriptive. This section is divided into four subsections; in Section 2.1, we review continuation techniques and in Section 2.2, we describe a Newton-multi-level procedure in the context of nested iteration [13, 25]. In Section 2.3, we describe the j-level iterative method used to solve the sparse linear systems which arise [8, 11]. Finally, in Section 2.4, we describe the use of local adaptive mesh refinement.

2.1. Continuation Algorithms

In this section, we review the essential features of path-following continuation methods [20, 24, 29, 40]. The key idea is to parameterize the solutions $[u(\sigma),\lambda(\sigma)]$ in terms of a new parameter σ that approximates the arclength parameter s, instead of parameterizing $u(\lambda)$ in terms of the natural parameter λ. This is usually achieved by augmenting the equation (2) by an auxillary equation that approximates the arclength condition:

$$||\dot{u}(s)||^2 + |\dot{\lambda}(s)|^2 = 1 , \tag{4}$$

to give an inflated system with unknowns $u(\sigma)$ *and* $\lambda(\sigma)$:

$$\begin{aligned} G(u(\sigma),\lambda(\sigma)) &= 0 , \\ N(u(\sigma),\lambda(\sigma),\sigma) &= 0 . \end{aligned} \tag{5}$$

Instead of solving for $u(\lambda)$ for a given value of λ, we solve for $u(\sigma)$ and $\lambda(\sigma)$ for a given value of σ. The auxillary function N is constructed so that the target solution (u, λ) is a regular solution of (5) [29], even at simple turning points. Thus, a regular nonlinear iterative method can be applied to solve this inflated system.

Most continuation methods are of the predictor-corrector type, usually with a predictor along the tangent :

$$(u_p, \lambda_p) = (u_0+\alpha_p \dot{u}_0, \lambda_0+\beta_p \dot{\lambda}_0) \tag{6}$$

with the predictor step (α_p, β_p), and where the unit tangent $(\dot{u}_0,\dot{\lambda}_0)$ at a known solution (u_0,λ_0) is defined by:

$$G_u(u_0,\lambda_0)\dot{u}_0 + G_\lambda(u_0,\lambda_0)\dot{\lambda}_0 = 0$$
$$||\dot{u}_0||^2 + |\dot{\lambda}_0|^2 = 1 . \tag{7}$$

The predictor (u_p, λ_p) is then used as an initial guess for a nonlinear iteration used to solve the system (5). The predictor step (α_p, β_p) is usually chosen adaptively to ensure convergence of the corrector. If the predictor fails to achieve convergence in the corrector, then the parameterization N and σ are adjusted (e.g. the continuation step can be reduced) and the process repeated.

In addition, certain points on the solution curve demand special attention. For example, one may want to compute the solution at selected values of λ or $||u||$ (target points). Moreover, one may want to be able to detect singular points and locate them accurately. In case a bifurcation is detected, one may also want to switch to the bifurcating branch.

We summarize the essential features in the following general algorithm for one step of the continuation procedure:

Continuation Procedure

1. Compute the unit tangent $[\dot{u}_0,\dot{\lambda}_0]$ at $[u_0,\lambda_0]$ by (7).
2. Determine the local parameterization N and σ.
3. If close to a target point, adjust N and σ so that $[u(\sigma),\lambda(\sigma)]$ will be the target point.
4. Compute a predicted solution $[u_p,\lambda_p]$.
5. Use $[u_p,\lambda_p]$ as initial guess in a nonlinear iteration for solving the system (5) to obtain $[u(\sigma),\lambda(\sigma)]$. If the iteration fails to converge, then adjust N and σ and go to step 4.
6. If a limit point is detected, compute it accurately if desired.
7. If a bifurcation point is detected, compute it accurately and switch branch if desired.

2.2. Nested Iteration

For concreteness, we consider the nonlinear elliptic equation:

$$\mathcal{L}(u) \equiv - \nabla.d\nabla u + f(u, \nabla u, \lambda) = 0 \quad \text{in } \Omega ,$$
$$u = 0 \qquad \text{on } \partial\Omega, \tag{8}$$

where, for simplicity, Ω is a closed, polygonal region in R^2. We assume $d > 0$ in $\bar{\Omega}$ and d and f are smooth functions of their arguments. We will assume for the moment that $\lambda \in R$ is fixed, and that for this value of λ, (8) is well-posed, and has one or more isolated solutions.

A weak form of (8) is: Find $u \in H_0^1(\Omega)$ such that

$$a(u,v) = 0 \qquad \forall\, v \in H_0^1(\Omega), \tag{9}$$

where

$$a(u,v) = \int_\Omega \{ d\nabla u\nabla v + f(u, \nabla u, \lambda)v \}\, dx\, dy. \tag{10}$$

$H^1(\Omega)$ will denote the usual Sobolev space equipped with the norm

$$\|u\|_1^2 = \int_\Omega \{ |\nabla u|^2 + u^2 \}\, dx\, dy$$

and

$$H_0^1(\Omega) = \{ v \in H^1(\Omega) \mid v|_{\partial\Omega} = 0 \}.$$

For $u \in H_0^1(\Omega)$, we define the form b(u;v,w), linear in its last two arguments, by

$$b(u;v,w) = \int_\Omega \{ d\nabla v\nabla w + e.\nabla v\, w + c\, vw \}\, dx\, dy$$

where (11)

$$e_i = \partial f/\partial u_{x_i}\,(u,\nabla u,\lambda) ,$$
$$c = \partial f/\partial u\,(u,\nabla u,\lambda).$$

We assume that for u sufficiently close to an isolated solution of (9), the linear elliptic problem:

Find $z \in H_0^1(\Omega)$ such that

$$b(u;z,v) = g(v) \qquad \forall\, v \in H_0^1(\Omega) \tag{12}$$

has a unique solution, where g is a smooth linear functional [9].

Let $\mathcal{T}_1$ be a quasi uniform shape regular triangulation of Ω. Let $M_1 \subset H_0^1$ be the N_1-dimensional space of C^0 piecewise linear polynomials associated with $\mathcal{T}_1$. We inductively define a sequence of triangulations $\{\mathcal{T}_j\}$ with $\mathcal{T}_j$ being a refinement of $\mathcal{T}_{j-1}$ (though not necesarily a uniform refinement), and corresponding N_j-dimensional spaces $M_j \subset H_0^1$. We assume $N_j \geq \beta N_{j-1}$ for $\beta > 2$, $j > 1$; $\beta \approx 4$ is typical for elliptic equations in R^2. Because $\mathcal{T}_j$ is a refinement of $\mathcal{T}_{j-1}$, M_{j-1} will be a subspace of M_j, $j > 1$.

The discrete form of (9) is: Find $u_j \in M_j$ such that

$$a(u_j,v) = 0 \quad \forall\, v \in M_j. \tag{13}$$

We assume that for any isolated solution u of (9) there is a corresponding sequence of isolated solutions $\{u_j\}$ of (13), and that these solutions converge to u at a specific rate, i.e.

$$\|u - u_j\|_1 \leq c\, N_j^{-q}, \tag{14}$$

where $c = c(u, \mathcal{T}_1)$ is a constant and $q > 0$.

These discrete problems can be solved by approximate Newton methods [13, 21]. Given $u_j^{(0)} \in M_j$, we compute a sequence $u_j^{(k)} \in M_j$, using

$$\begin{aligned} b_j(x^{(k)}, v) &= - a(u_j^{(k)}, v) \quad \forall\, v \in M_j, \\ u_j^{(k+1)} &= u_j^{(k)} + t^{(k)}\, x^{(k)}. \end{aligned} \tag{15}$$

The case $b_j(.,.) = b(u_j^{(k)};.,.)$, $t^{(k)} = 1$ is Newton's method. Equations (15) describe an approximate Newton method because the bilinear form $b_j(.,.)$ is chosen to approximate the Jacobian (11). For example, b_j might correspond to the approximate solution of the Newton equations using an iterative method as an inner iteration [13]. The damping parameters $t^{(k)} \in (0,1]$ can be chosen to guarantee convergence [12, 21].

The nested iteration strategy (also called full multi-grid by Brandt [15]) involves sequentially solving the problems (13) for $j = 1,2,....$ using the approximate solution of the (j-1)-st problem as the initial guess for the j-th.

Algorithm NI:

1. Solve (13) for $j = 1$ using S_1 iterations of (15) and an appropriate initial guess $u_j^{(0)}$.
2. Solve (13) for $j > 1$ using S_j iterations of (15) and initial guess $u_j^{(0)} = u_{j-1}^{(S_{j-1})}$.

Under appropriate hypothesis [13] one can show Algorithm NI to be an extremely effective procedure for solving (13). For each problem after the first, one must reduce the initial error by only a fixed amount $\epsilon < \beta^{-q}$ independent of j in order for the computed solutions $u_j^{(S_j)}$ to converge to the true solution u of (9) at the same rate as the u_j (see (14)). Because of this modest required error reduction, one can often show $S_j = 1$ for j sufficiently large is adequate [13]. Because the N_j increases geometrically, asymptotically the cost of solving the nonlinear problem at level j is only a small multiple (usually less than 2) of the cost of approximately solving one set of linear equations of the form (15) for the grid $\mathcal{T}_j$. If this set of equations can be solved in $O(N_j)$ operations (as it frequently can using the j-level iterative method), then the overall complexity

will be of optimal order $O(N_j)$ [13].

2.3. The j-level Iterative Method

The j-level iterative method is designed to solve linear systems of the form:

Find $z \in M_j$ such that

$$b(u;z,v) = g(v) \quad \forall\, v \in M_j. \tag{16}$$

Here $g: M_j \rightarrow R$ is a linear functional. In the special case $u = u_j^{(k)}$, $g(v) = -a(u_j^{(k)}, v)$, (16) becomes the Newton equation for (13), and z can be identified as $x^{(k)}$.

For $j = 1$, equations (16) are solved directly, e.g., by sparse Gaussian elimination. For $j > 1$, a single iteration of the j-level scheme takes an initial guess $z^{(0)} \in M_j$ to a final guess $z^{(m+1)} \in M_j$. The procedure is defined recursively as follows:

For $1 \leq k \leq m$,

$$z^{(k)} = S(z^{(k-1)}), \text{ where S is a smoothing operator.} \tag{17}$$

Set $\bar{g}(v) = g(v) - b(u; z^{(m)}, v)$ and solve the coarse grid problem: find $\delta \in M_{j-1}$ such that

$$b(u; \delta, v) = \bar{g}(v) \quad \forall\, v \in M_{j-1}, \tag{18}$$

using 2 iterations of the j-1 level scheme and initial guess zero for $j-1 > 1$ or directly if $j-1 = 1$. In either case we obtain an (approximate) solution $\bar{\delta} \in M_{j-1}$. Finally, we set

$$z^{(m+1)} = z^{(m)} + \bar{\delta}. \tag{19}$$

The operation S corresponds to the application of some iterative method (e.g. symmetric Gauss-Seidel, damped Jacobi) to the set of equations (16). Typically, m is quite small (≤ 4). Equations (18) and (19) are now commonly called a "coarse grid correction"; it is easy to see that $\delta \in M_{j-1}$ is a Galerkin approximation to the error $z - z^{(m)} \in M_j$. From (16 and (18) we have

$$b(u; z-z^{(m)}, v) = \bar{g}(v) \quad \forall\, v \in M_{j-1}.$$

Under suitable hypothesis on the triangulations $\mathcal{T}_j$, subspaces M_j (and their bases), and the smoothing iteration, one can prove that the work required to reduce the error by any fixed amount is $O(N_j)$, leading to the overall optimality of the j-level iterative method when used in conjunction with nested iteration.

2.4. Adaptive Mesh Refinement

In the use of nested iteration and the j-level iteration the triangulation $\mathcal{T}_{k+1}$ is not required until the approximate solution $u_k^{(s_k)}$ has been computed. This suggests the possibility of using the computed solution $u_k^{(s_k)}$ in the computation of $\mathcal{T}_{k+1}$ by a local adaptive mesh refinement algorithm [4, 5, 6, 7, 15, 44]. The basic goal is to construct a triangulation in which the error is (approximately) uniformly distributed among the elements by refining those elements in which the local error is largest. Suppose element $\bar{t} \in \mathcal{T}_k$ has the largest error among the elements of $\mathcal{T}_k$. Further, suppose that the errors in elements obtained by the refinement of $\bar{t}$ would have local errors of size ϵ. Then a reasonable refinement criteria is to refine those elements in $\mathcal{T}_k$ whose error satisfies $||e||_t > \epsilon$. This will hopefully result in a new mesh in which all elements have errors bounded by ϵ, and will tend to equi-distribute the error among the elements. The threshold value ϵ can be extrapolated using the calculated elementwise error estimates from the previous two levels. Since only a few elements may have errors larger than ϵ, the procedure may have to be iterated several times in order to compute a triangulation $\mathcal{T}_{k+1}$ whose corresponding subspace M_k has $N_{k+1} \approx \beta N_k$. The scheme used in PLTMGC is described in detail in [7, 14].

3. Implementation in PLTMGC

In this section, we summarize our strategies for combining the methods of Sections 2 into an efficient and robust procedure for solving (1). There are several possible interpretations of what constitutes the solution of (1). From our present perspective, we desire a procedure which can yield a qualitatively accurate description of all the solution curves, with approximate locations of limit points and bifurcation points. At a few selected points, perhaps only one, we would like to compute an accurate approximation of the solution u.

Thus our overall strategy involves two main components:

1. The use of the continuation process of Section 2.1 on the coarsest mesh $\mathcal{T}_1$ only.
2. The use of the multi-level procedure of Section 2.2 - 2.4 to produce accurate solutions at a few selected points.

This view is in contrast to that of some other investigators, e.g. Mansfield [32], who suggest schemes for carrying out the continuation process on the finest grid. When a quantitatively accurate description of the solution curves is desired, such schemes would seem more natural than ours because they make use of previously generated solutions on the finer meshes to generate

initial guesses for the solutions on the finer meshes in the multi-level algorithm, as suggested by Hackbusch [27] for example. However, we feel that neither the convergence nor the efficiency of the multi-level algorithm is critically sensitive to the accuracy of these initial guesses, because initial guesses interpolated from coarser grids are accurate enough to insure rapid convergence. Also, because of our use of adaptive mesh refinement, in general our procedure will not generate the same sequence of meshes at all points on the solution curves. Finally, following the solution curves on all the finer meshes is far more costly, perhaps by orders of magnitude if several levels are involved. Thus, following solution curves on the coarsest mesh only is advantageous when the main goal of the computation is to compute a particular solution for a particular value of λ or $||u||$ and if only a general qualitative behaviour of the solution curve is sufficient. This necessarily implies that the coarsest grid has to be fine enough to represent the qualitative behavior of the solution curves of interest. Brandt [15] has suggested using the frozen-tau technique to obtain fine grid accuracy on the coarse grids.

With respect to coarse grid continuation, there are three main tasks which should be handled by a general solver:

1. Continuation from a starting value to a target value.
2. Location of limit points and bifurcation points.
3. Branch switching at bifurcation points.

With respect to the adaptive refinement at a selected point, two cases must be handled:

1. Refinement away from singular points.
2. Refinement near a singular point.

3.1. Computation of Tangent Field

A major component of a continuation method is the computation of the unit tangent $[\dot{u}(s),\dot{\lambda}(s)]$ to the solution curve at a point $[u,\lambda]$ on the solution curve, which can be computed relatively inexpensively from its definition by solving only *one* linear system with G_u:

$$\begin{aligned} G_u\, z &= -\, G_\lambda . \\ \dot{\lambda}^2 &= 1 \,/\, (\|z\|^2 + 1). \\ \dot{u} &= \dot{\lambda}\, z. \end{aligned} \tag{20}$$

Equation (20) determines $[\dot{u},\dot{\lambda}]$ up to a directional orientation, which can be fixed by some convention. For example, the strategy we employ is to require that the new tangent forms an

acute angle with the previous tangent to insure that we are "following" the solution curve.

Although the above procedure is not mathematical well-defined when G_u is exactly singular, in practice it behaves like inverse iteration and produces the right results.

3.2. Local Parameterizations

The most popular choices for the local parameterization N are usually some approximations to the arclength condition (4) to ensure that at least locally the solution curve is "followed". A few typical N's that have been used in the literature are:

1. $N_1(u,\lambda,\sigma) \equiv \dot{u}_0^T(u - u_0) + \dot{\lambda}_0(\lambda - \lambda_0) - \sigma$ [29].
2. $N_2(u,\lambda,\sigma) \equiv e_i^T(y - y_0) - \sigma$, where $y = (u,\lambda)^T$, e_i is the i-th unit vector and the index i is chosen so that the Jacobian of (5) is as well-conditioned as possible ([1, 31, 39]).
3. $N_3(u,\lambda,\sigma) \equiv ||u - u_0||^2 + |\lambda - \lambda_0|^2 - \sigma^2$ [29].
4. $N_\theta(u,\lambda,\sigma,\theta) = \theta\, \dot{u}_0^T(u - u_0) + (2\text{-}\theta)\, \dot{\lambda}_0\, (\lambda - \lambda_0) - \sigma = 0$, for $0 \leq \theta \leq 2$ [29].
5. $N_r(u,\lambda,\sigma,\theta) = \theta\, \dot{r}_0\, (r - r_0) + (2\text{-}\theta)\, \dot{\lambda}_0\, (\lambda - \lambda_0) - \sigma$, where $r \equiv ||u||$.

N_1 is the usual psuedo arclength parameterization and defines a hyperplane perpendicular to the tangent and at a distance σ along the tangent from (u_0,λ_0). N_2 corresponds to parameter switching. N_3 is similar to N_1 except a sphere is used instead of a hyperplane. N_θ is a modification of N_1. Choosing θ not equal to 1 corresponds to rotating the tangent. This extra degree of freedom sometimes allows an increase in the continuation step near points on the solution curve with high curvature. Note that $\theta = 0$ can be used to compute target points for given values of λ. N_r seems to be a new parameterization and is the one used in PLTMGC. It corresponds to N_θ in the space $(||u||, \lambda)$. The motivation behind this is that for many PDE-type problems, there are really only two parameters that affect the continuation procedure: some measure of u and λ. Note that with $\theta = 0$ and $\theta = 2$ both target points in λ and $||u||$ can be computed. Thus, this parameterization can be viewed as a hybrid between N_1 and N_2 adapted for PDE type problems. Note that since $\dot{r}\, r = \dot{u}^T u$, N_r is ill-defined when $\dot{u}_0^T u_0 = \dot{\lambda}_0 = 0$. In such situations, which can arise at isolated points (e.g. symmetry breaking bifurcation points), we switch (temporarily) to N_θ.

3.3. Target Points

In PLTMGC, the user can specify target values for both $||u||$ and λ. Depending on these values, θ and σ are determined for N_r. If only a value for λ, say λ_t, is given, then θ is set to 0 and σ is set to $2\,\dot{\lambda}_0\,(\lambda_t - \lambda_0)$ so that $N_r = 0$ corresponds exactly to specifying $\lambda = \lambda_t$ (assuming $\dot{\lambda}_0 \neq 0$). Similarly, if only $||u|| = r_t$ is specified, then θ is set to 2 and σ set to $2\,\dot{r}_0\,(r_t - r_0)$. If both target values are specified, then PLTMGC determines heuristically which is easier to reach. It does this by using an approximate local model of the solution curve obtained by interpolating the two most current solutions and their tangents. Points on this approximate solution curve which achieve either of the specified target values are computed and the one which is "closest" in arclength to (u_0, λ_0) determines which target value is to be attempted.

3.4. The Predictor

Once θ and σ are determined, the predictor (u_p,λ_p) of (6) is computed by finding scalars (α_p, β_p) which satisfy

$$N_r(u_p, \lambda_p, \sigma, \theta) = 0 \tag{21}$$

$$R(u_p, \lambda_p) = 0 \tag{22}$$

where $R(u,\lambda) = u^T G(u,\lambda) \,/\, u^T u$. Our use of (21) is motivated by the psuedo-arclength method, in which one chooses a predictor with $\alpha_p = \beta_p$ and (21) is satisfied [29]. Our use of the Rayleigh quotient (22) was motivated by the linear eigenvalue problem, and by the success of a continuation procedure of Mittelmann and Weber [35] for a certain class of nonlinear elliptic equations. The predictor equations are solved by several iterations of an approximate Newton method. It is easy to insure that (21) is always exactly satisfied after each iteration so that even if this iteration fails to converge (which is rare), the resulting predictor still satisfies one of the two corrector equations.

Our view of (21) - (22) again reflects our view that PDEs of the form (1) are (locally) very low dimensional problems. In some respects, this is analogous to the reduced basis approach of [23, 36, 37] and an approach of Jarausch and Mackens [28]. We also remark that it is usually worthwhile, from a computational point of view, to compute a more accurate predictor, which in this case involves solving two scalar equations, rather than use more corrector iterations, which involve solving much larger sets of equations. Such an investment can also allow much larger steps, if the goal of the continuation procedure is merely to reach a target point.

This new predictor algorithm seems to be very robust and allows rather large continuation steps to be taken (see Section 4).

3.5. The Corrector Iteration

The main task for the corrector iteration is to solve for the solution of the coupled nonlinear system (5) starting with an initial guess provided by the predictor. Since the solution we sought is a regular solution of (5), in principle any regular nonlinear iterative method (e.g. Newton's method, quasi-Newton methods, etc.) can be applied. However, for large and sparse problems, such as discretizations of partial differential equations that we treat here, it is important for efficiency reasons to exploit the structures in G. For a survey of methods used in the corrector iteration, the reader is referred to [16].

Most correctors used in continuation methods, especially for large and sparse systems, are based on Newton's method [16]. In PLTMGC, an approximate Newton method similar to the iteration (15) is used. The damping step $t^{(k)}$ is chosen to force the norm of the residual to decrease. At each corrector iteration, a linear system has to be solved for ∂u and $\partial \lambda$ involving some approximations to the Jacobian of (5). This often constitutes the major part of the overall computational cost. All the linear systems that arise in the corrector iterations are of the form:

$$\begin{bmatrix} A & b \\ c^T & d \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} f \\ g \end{bmatrix} \tag{23}$$

where (A, b, c, d) are approximations to (G_u, G_λ, N_u, N_λ) respectively. For solving (23), we use the following block-elimination algorithm :

Algorithm BE: [29]

1. Solve $A v = b$,
   $A w = f$.
2. Compute $y = (g - c^T w) / (d - c^T v)$.
3. Compute $x = w - y v$.

Note that only a solver for A is needed. On the coarsest grid, a direct solver is used. On the

finer meshes, one can use the j-level multi-grid method described in Section 2.3. However, as we have shown in [16, 19], Algorithm BE may be numerically unstable when A is nearly singular, as is the case near singular points of the solution curves. Moreover, the near-singularity of A may also retard the convergence of the multi-level algorithm [8, 20].

In [19], we proposed using implicit deflation techniques developed in [17, 42] to compute numerically stable representations for the solutions v and w. These deflation techniques can be viewed as working in subspaces orthogonal to approximate null vectors ϕ and ψ of A and are implicit in the sense that they only involve solving systems with A. These deflated decompositions [17] of v and w lead to a numerically stable variant of the BE algorithm. (We assume a one-dimensional null space, although our remarks could be modified to deal with the more general situation.)

Algorithm DBE: Deflated Block-Elimination Algorithm [19].

1. Compute an approximate normalized left singular vector ψ of A.
2. Compute $\phi = \delta\, A^{-1}\psi$, where $\delta = 1 \,/\, \|A^{-1}\psi\|$.
3. Compute $c_b = (\psi^T b)$ and $c_f = (\psi^T f)$.
4. Solve $Av_d = b - c_b\, \psi$ for v_d. (v is represented as: $v = v_d + (c_b/\delta)\phi$)
5. Solve $Aw_d = f - c_f\, \psi$ for w_d. (w is represented as: $w = w_d + (c_f/\delta)\phi$)
6. Compute $h_1 = g - c^T w_d$, $h_2 = d - c^T v_d$, $h_3 = h_1 c_b - h_2 c_f$, $h_4 = (c^T\phi)\, c_f - \delta h_1$, $D = (c^T\phi)\, c_b - \delta h_2$.
7. Compute $y = h_4 \,/\, D$ and $x = w_d + (h_3\phi - h_4 v_d) \,/\, D$.

Note that only two solves with A is need, exactly the same as in Algorithm BE. The only overhead involved for performing the deflation is the computation of approximate left and right singular vectors of A which can be accomplished with one or two backsolves with A [16, 17]. Algorithm DBE can be proven to be numerically stable [19] independent of the singularity of A.

To apply the deflation techniques on the fine grids, we need to compute the deflated solutions v_d and w_d using a multi-level algorithm. To do this, we have developed a multi-level deflation technique. Let ϕ_j (ψ_j) be the left (right) singular functions corresponding to the near zero

singular value of b_j. When solving for each of the vectors v_d and w_d, all residuals are systematically purged of their ψ_j components. One saves the decomposition coefficients c_b and c_f on the finest grid only. This procedure can be proven to be convergent and numerically stable independent of the singularity of A [10, 26]. This strategy is similar in spirit, though not in detail, to that of Chan and Keller [20]. We note that the singular functions can be computed with low cost by using a multi-level inverse iteration algorithm [9, 20, 26] and can be reused for several continuation steps. Because of their robustness and negligible overhead, these deflation algorithms are used at all continuation steps and on all grids, without necessitating a check on the singularity of A.

3.6. Failure Control for the Corrector Iteration

An essential part of the corrector algorithm is the failure control. Our philosophy is that when the corrector iteration is having convergence difficulty, we abort the iteration and change the parameterization. In our implementation, we declare that the corrector iteration has failed if either it took too many iterations without convergence or if it took too many damping steps in attempting to force descent of the residuals.

The corrector iteration usually fails only near points on the solution curve with high curvature. When this happens, it means that either the continuation step σ is too large or that no solution to (5) exists for the current values of θ and σ. For example, if a λ target value is specified near a turning point, with a corresponding value of $\theta = 0$, a solution may not exist. When this occurs, PLTMGC does two things: it reduces the value of σ (e.g. halves it) and it uses a value for θ ($\approx$ 1) forcing N_r to follow the solution curve more closely and deferring the attempt to hit the specified target value on the current step. This process is repeated until convergence is finally achieved which is guaranteed for small enough σ.

3.7. Singular Points

In many applications, in addition to tracing the solution branches, one is also interested in locating the singular points themselves, because they are often related to the stability of the solution. Due to their special physical significance, many algorithms have been proposed for determining these singular points accurately [18, 29, 33]. In this paper, we shall only deal with the determination of simple turning points and simple bifurcation points which can be characterized as points on the solution curve where

$$G_u \text{ is singular, with a one dimensional null space,} \tag{24}$$
$$G_u \text{ is boundedly invertible on its range,}$$

$$\text{and} \quad G_\lambda \notin \text{Range}(G_u) \text{ for a simple turning point,} \tag{25}$$
$$\text{and} \quad G_\lambda \in \text{Range}(G_u) \text{ for a simple bifurcation point.}$$

There are two basic issues: detection and accurate location. It can be shown that the determinant of G_u changes sign across simple turning points and simple bifurcation points and therefore it can be used as a detection device on the coarsest grid. In addition, across a turning point, $\dot{\lambda}$ also changes sign and this serves to differentiate between the two types of singular points. For accurate location, there are primarily two classes of methods. The first consists of local iterative algorithms based on an inflated system consisting of $G(u, \lambda) = 0$ augmented by a characterization similar to (24), constructed so that the singular point is a unique and isolated solution of the inflated system. The other class of algorithms consists of methods based on a path tracing continuation method by successively using it to compute points on the solution curve that approach the singular point. We use the second approach in PLTMGC, primarily because many parts of the procedure are already present in the continuation program.

For turning points the method that we use is similar to one proposed by Chan [18] and Keller [30]. It is based on applying the scalar secant method (combined with bisection to ensure convergence) to $\dot{\lambda}(\sigma) = 0$ to determine the correct value of σ to use in N_r so that the continuation procedure will hit the turning point.

For bifurcation points, we use a secant method (again combined with bisection to ensure convergence) on the characterization $\delta(\sigma) = 0$, where δ is the approximation to the smallest singular value computed in step (2) of Algorithm DBE. Since the singular value δ does not change sign across a singular point, we make a slight modification in order to speed convergence. We scale the normalized singular functions ψ and ϕ so that $\psi^T\phi$ has the same sign on both sides of the singular point. Then we define δ to be $\psi^T A\phi$ which has the same magnitude as the smallest singular value of A but does change sign across the singular point. This method seems to be new and is very robust and efficient.

3.8. Branch Switching at Bifurcation Points

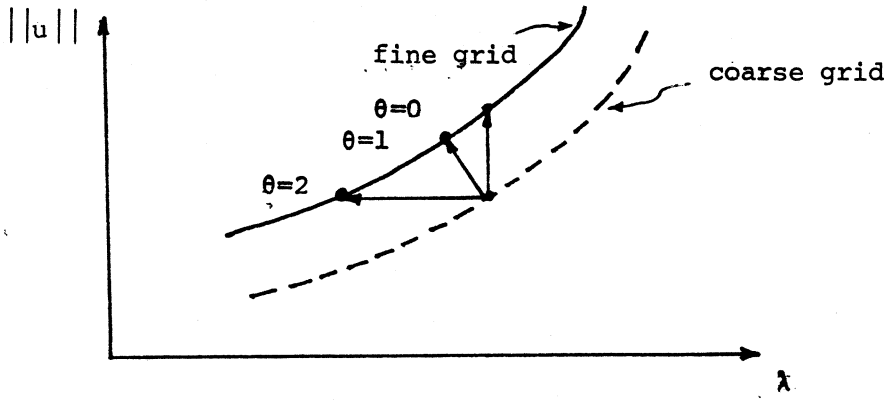
For a survey of branch switching methods, we refer the reader to Keller [29]. The method that we use is exactly Method I in [29]. Basically, the tangent $(\dot{u}, \dot{\lambda})$ corresponding to the bifurcated branch is computed using the approximate singular functions ψ and ϕ at the bifurcation point and finite difference approximations to the second derivatives of G. This new tangent is then used in the basic continuation procedure to march along the bifurcated branch.

3.9. Refinement Strategies

The algorithms presented so far allow the solution curves to be followed on the coarsest grid. However, at occasional points on the solution curve, we may want to compute a solution with higher accuracy than that provided by the discretization on the coarsest grid. PLTMGC allows three options for refinement onto finer grids, corresponding to different choices of θ in the local parameterization N_r. For all cases, the step length parameter σ is set to zero, and the standard multi-level procedure with adaptive local mesh refinement is used. We have found it adequate to use values of $\dot{r}$, $\dot{\lambda}$, u_0 and λ_0 corresponding to the level 1 solution in the parameterization N_r for all refined levels, although one could certainly update these values as the refinement proceeds.

First, one can choose to refine with a fixed target value λ_t by setting $\theta = 0$. Alternatively, one can use $\theta = 2$ to obtain solutions on all levels at a fixed target value for $||u||$. Third, one can refine on the perpendicular hyperplane passing through the current coarse grid solution by choosing $\theta = 1$. Typical situations are illustrated in Figure 3-1.

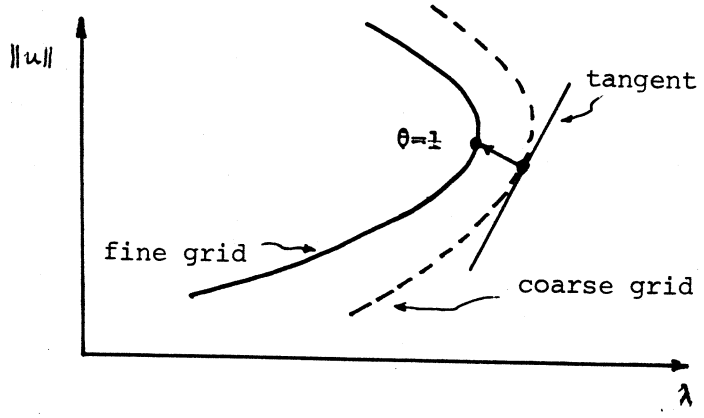
Figure 3-1: Refinement away from singular points.



In general, the appropriate choice of θ depends on the particular applications and requires user input. Away from singular points, all three strategies can usually be used. Near singular points, however, the situation is quite different. For example, although a solution on the coarsest grid

may exist for a particular value λ_t, solutions on the refined grids may not. Here an appropriate strategy is to take $\theta = 1$ (or $\theta = 2$.) This is illustrated in Figure 3-2. Another example is the linear eigenvalue problem, where there are vertical solution branches in the ($||u||$, λ) projection and one should refine with $\theta = 2$ or $\theta = 1$ but not $\theta = 0$ (see Section 4).

Figure 3-2: Refinement near singular points.

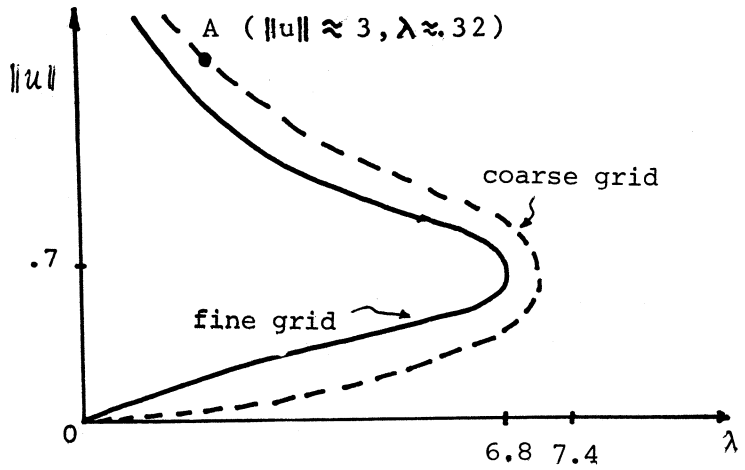


4. Examples of Usage

In this section, we shall present a few sample runs with PLTMGC. The goal is to demonstrate the capabilities of PLTMGC rather than to solve complicated problems.

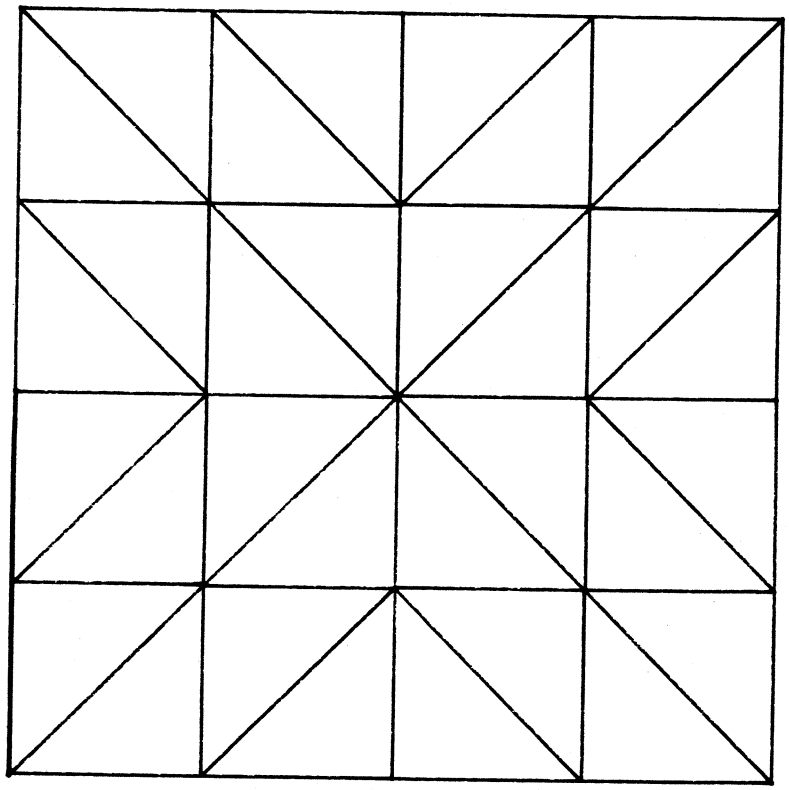
The first two runs are for the standard model problem (Bratu's Problem) $\Delta u + \lambda e^u = 0$ with the Dirichlet boundary condition $u = 0$ on a unit square in R^2.

Figure 4-1: Solution branches of Bratu's Problem: $\Delta u = \lambda e^u$.



This problem has a simple turning point at $\lambda \approx 6.8$ and $||u|| \approx 0.7$ (see Figure 4-1). The level 1 (coarsest) grid for multigrid is shown in Figure 4-2 and has 25 vertices, only 9 of which are unknowns. The first run is for demonstrating the λ-target and refinement capabilities of

Figure 4-2: Coarsest Grid for Bratu's Problem: $\Delta u = \lambda e^{u}$.

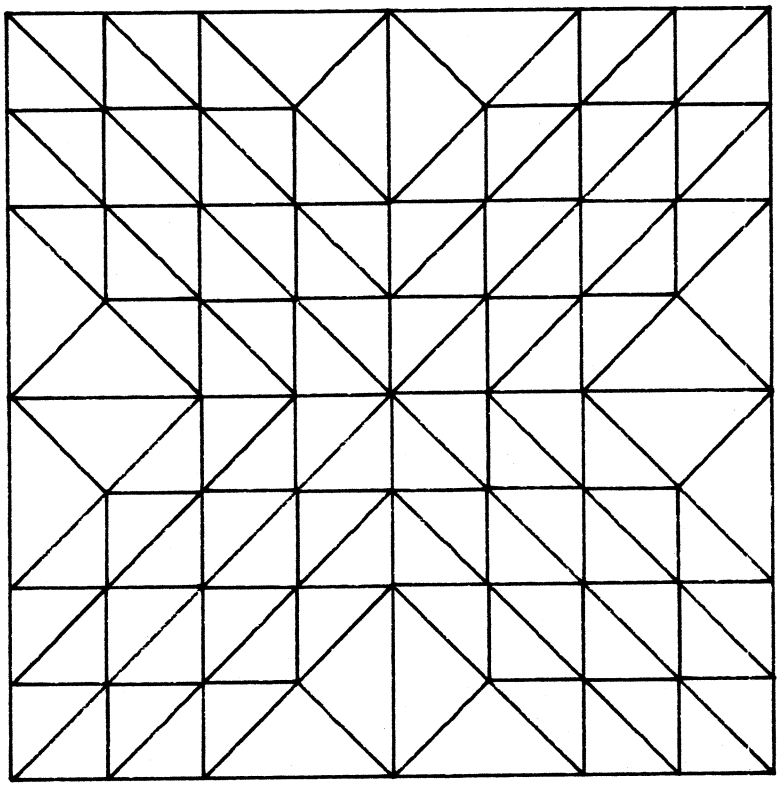


Number of
vertices = 25

Figure 4-3: Fine Grids for Bratu's Problem: $\Delta u = \lambda e^u$.

LEVEL 2

Number of
vertices = 69



LEVEL 3

Number of
vertices = 205

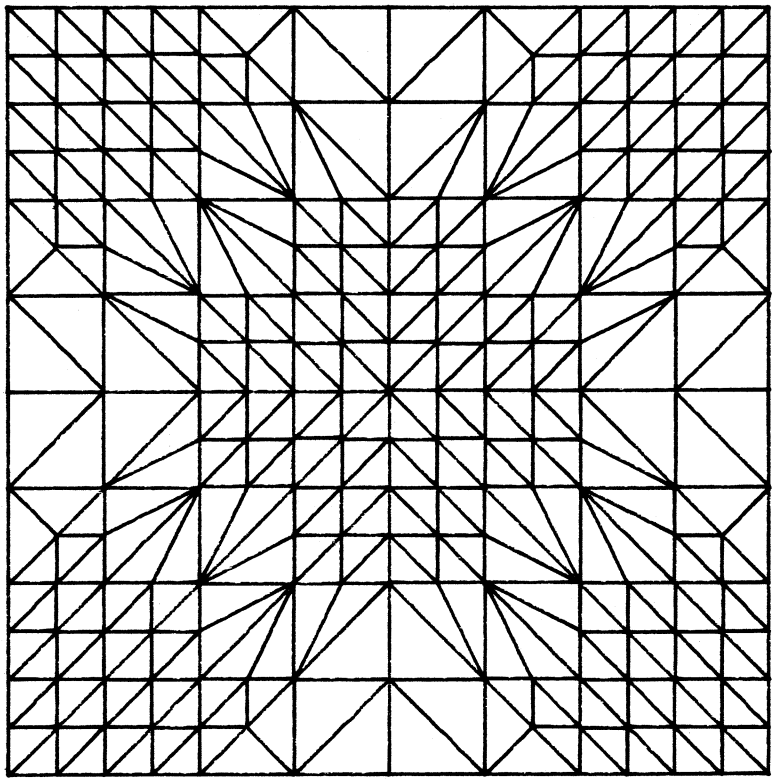


Table 4-1: Bratu's Problem from $\lambda = 0$ to $\lambda = 6$ with refinement.

Legend

```
L      : level number of grid,
NI     : number of corrector iterations,
DIGITS : number of correct digits in the solution.
```

Continue to target point with $\lambda = 6$:

L	NI	λ	$\|\|u\|\|$	$\dot{\lambda}$	$\dot{r}$	det(G_u)	δ
1	1	0.000E+00	0.000E+00	0.999E+00	0.000E+00	0.411E 9	0.117E+01
1	4	0.600E+01	0.358E+00	0.994E+00	0.111E+00	0.504E 8	0.630E+00

Refine to level 3 using $\theta = 2$ (constant $\|\|u\|\|$:)

2	3	0.567E+01	0.358E+00	0.993E+00	0.118E+00	0.595E 8	0.167E+00
3	3	0.557E+01	0.358E+00	0.993E+00	0.119E+00	0.626E 8	0.513E-01

Information about accuracy of the solutions:

NORM	SOLN	ERROR	DIGITS
H1	0.165E+01	0.176E+00	0.971
L2	0.357E+00	0.605E-02	1.771
MAX	0.682E+00	0.139E-01	1.691

Refine using $\theta = 0$ (constant λ:)

L	NI	λ	$\|\|u\|\|$	$\dot{\lambda}$	$\dot{r}$	det(G_u)	δ
2	3	0.600E+01	0.400E+00	0.990E+00	0.139E+00	0.478E 8	0.148E+00
3	2	0.600E+01	0.416E+00	0.988E+00	0.152E+00	0.468E 8	0.433E-01

Refine using $\theta = 1$ (perpendicular hyperplane:)

2	3	0.600E+01	0.399E+00	0.990E+00	0.139E+00	0.480E 8	0.149E+00
3	2	0.599E+01	0.415E+00	0.989E+00	0.151E+00	0.470E 8	0.434E-01

PLTMGC. Starting from the trivial solution ($u = 0$, $\lambda = 0$), PLTMGC managed to get to the target point $\lambda = 6$ in one step with 4 corrector iterations (see Table 4-1). Next we ask PLTMGC to refine the solution to 3 multigrid levels with $\theta = 2$ (i.e. constant $||u||$ level). The two extra grids are generated adaptively using the mesh refinement strategies discussed in Section 2.4 and are shown in Figure 4-3. As part of the adaptive mesh refinement procedure, the accuracy of the solutions is also generated. To demonstrate the different refinement possibilities, refinements using $\theta = 0$ (i.e. constant λ) and $\theta = 1$ (i.e. perpendicular hyperplane) are also generated from the same coarse grid solution. Notice that the number of corrector iterations on each level is quite small - no more than 3 iterations are needed.

The next run (see Table 4-2) demonstrates the $||u||$ target and turning point capabilities. Starting from the $\lambda = 6$ solution generated from the first run, a target value of $||u|| = 3$ is specified. Even though the target point is beyond the turning point on the upper branch and quite far away from it (point A in Figure 4-1), convergence is achieved in only one step with 4 corrector iterations. This robustness is primarily due to the extremely good predictor. Moreover, the turning point is detected and located accurately by the secant method on $\dot{\lambda}$ as described in Section 3.7. Notice that convergence is superlinear as the turning point is approached.

The third run (see Table 4-3) demonstrates the location of simple bifurcation points and branch switching. The problem is the linear eigenvalue problem: $-\Delta u = \lambda u$ with the Dirichlet boundary condition $u = 0$ on the unit square in R^2. The continuous problem has bifurcation points at the eigenvalues of the Laplacian $-\Delta$, i.e. $(n\pi)^2 + (m\pi)^2$ where n and m run through the natural numbers. Thus the first simple bifurcation is near $\lambda = 2\pi^2$ whereas the next simple bifurcation is near $\lambda = 8\pi^2$. The second eigenvalue of $-\Delta$ near $5\pi^2$ is a multiple one and does not correspond to a simple bifurcation. We use PLTMGC to locate the second simple bifurcation point. The coarsest grid used is illustrated in Figure 4-4 which is fine enough to resolve this eigenvalue. Starting from the trivial solution ($u = 0$, $\lambda = 0$), the target value of $\lambda = 25$ is reached in one step. Notice that the sign of the determinant has changed because we passed the first bifurcation. Next we continue to the target value of $\lambda = 80$. This is past the multiple eigenvalue near $\lambda = 5\pi^2$ and notice that the determinant does not change sign. At this point the bifurcation detection mechanism is switched on and the target value of $\lambda = 100$ is specified, which is beyond the second simple bifurcation point. Since the determinant changes sign this time, the bifurcation is detected and the secant method for δ described in Section 3.7 finds the bifurcation point in 4 steps. Then we switch branches and continue to the target value of $||u|| =$

Table 4-2: Bratu's problem near turning point.

From $\lambda = 6$ *to target value* $||u|| = 9$*:*

L	NI	λ	$\|\|u\|\|$	$\dot{\lambda}$	$\dot{r}$	$\det(G_u)$		δ
1	4	0.600E+01	0.358E+00	0.994E+00	0.111E+00	0.504E	8	0.630E+00
1	4	0.319E+00	0.300E+01	-0.550E+00	0.832E+00	-0.688E	9	0.246E+01

Secant method for turning point:

1	4	0.199E+01	0.206E+01	-0.959E+00	0.281E+00	-0.136E	9	0.213E+01
1	4	0.596E+01	0.121E+01	-0.978E+00	0.205E+00	-0.195E	8	-0.118E+01
1	3	0.737E+01	0.783E+00	-0.744E+00	0.667E+00	-0.381E	7	-0.140E+00
1	3	0.722E+01	0.570E+00	0.944E+00	0.330E+00	0.115E	8	0.267E+00
1	3	0.740E+01	0.689E+00	0.381E+00	0.924E+00	0.148E	7	0.454E-01
1	3	0.738E+01	0.770E+00	-0.675E+00	0.736E+00	-0.315E	7	-0.113E+00
1	3	0.741E+01	0.718E+00	-0.947E-01	0.994E+00	-0.336E	6	-0.109E-01
1	3	0.741E+01	0.710E+00	0.481E-01	0.998E+00	0.171E	6	0.546E-02
1	3	0.741E+01	0.713E+00	-0.412E-03	0.999E+00	-0.145E	4	-0.469E-04
1	3	0.741E+01	0.713E+00	0.979E-06	0.999E+00	0.796E	1	0.112E-06

Table 4-3: Linear eigenvalue problem: $-\Delta u = \lambda u$.

Continue from $\lambda = 0$ to $\lambda = 25$ then to $\lambda = 80$:

L	NI	λ	$\|\|u\|\|$	$\dot{\lambda}$	$\dot{r}$	$\det(G_u)$	δ
1	1	0.000E+00	0.000E+00	0.100E+01	0.000E+00	0.339E 35	0.304E+00
1	1	0.250E+02	0.000E+00	0.100E+01	0.000E+00	-0.206E 32	-0.711E-01
1	1	0.800E+02	0.000E+00	0.100E+01	0.000E+00	-0.329E 27	0.468E+00

Continue to $\lambda = 100$, check sign of determinant:

1	1	0.100E+03	0.000E+00	0.100E+01	0.000E+00	0.231E 25	0.174E+00

Determinant changes sign, find bifurcation point by secant method on δ:

1	1	0.900E+02	0.000E+00	0.100E+01	0.000E+00	-0.599E 25	0.188E-01
1	1	0.950E+02	0.000E+00	0.100E+01	0.000E+00	0.423E 25	-0.453E-01
1	1	0.915E+02	0.000E+00	0.100E+01	0.000E+00	-0.119E 21	0.884E-06
1	1	0.915E+02	0.000E+00	0.100E+01	0.000E+00	0.124E 22	-0.591E-05

Switch branch:

1	0	0.915E+02	0.000E+00	0.000E+00	0.100E+01	0.124E 22	-0.591E-05

Continue to target value $||u|| = 1$:

1	1	0.915E+02	0.100E+01	0.348E-04	0.100E+01	-0.126E 21	0.446E-06

Refine to 3 multigrid levels with $\theta = 2$ (constant $||u||$) :

2	3	0.813E+02	0.100E+01	0.196E-03	0.100E+01	-0.234E 27	0.402E-06
3	3	0.797E+02	0.100E+01	0.248E-03	0.100E+01	-0.354E 27	0.263E-06

Figure 4-4: Coarsest Grid for $-\Delta u = \lambda u$

Number of
vertices = 81

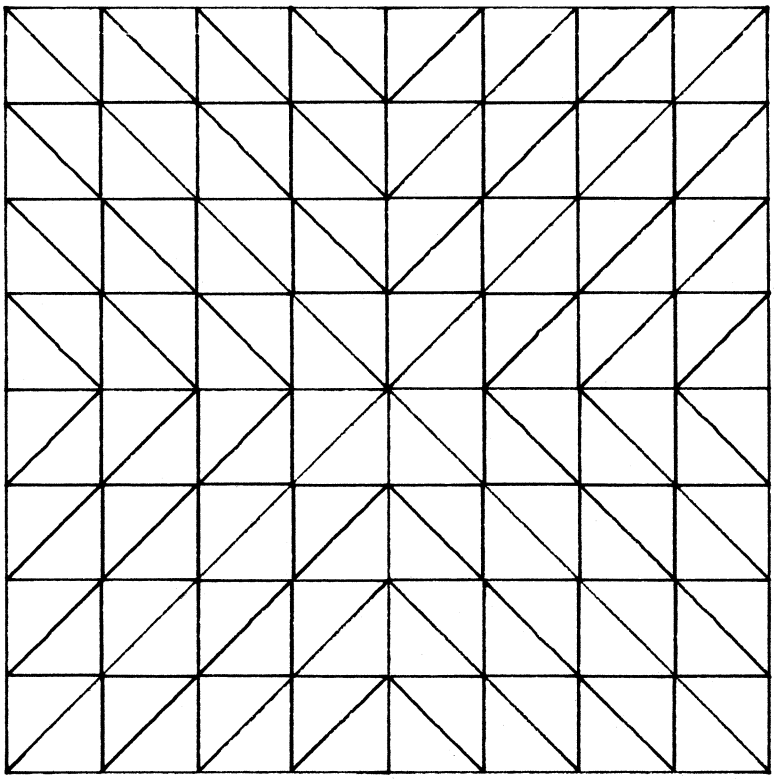
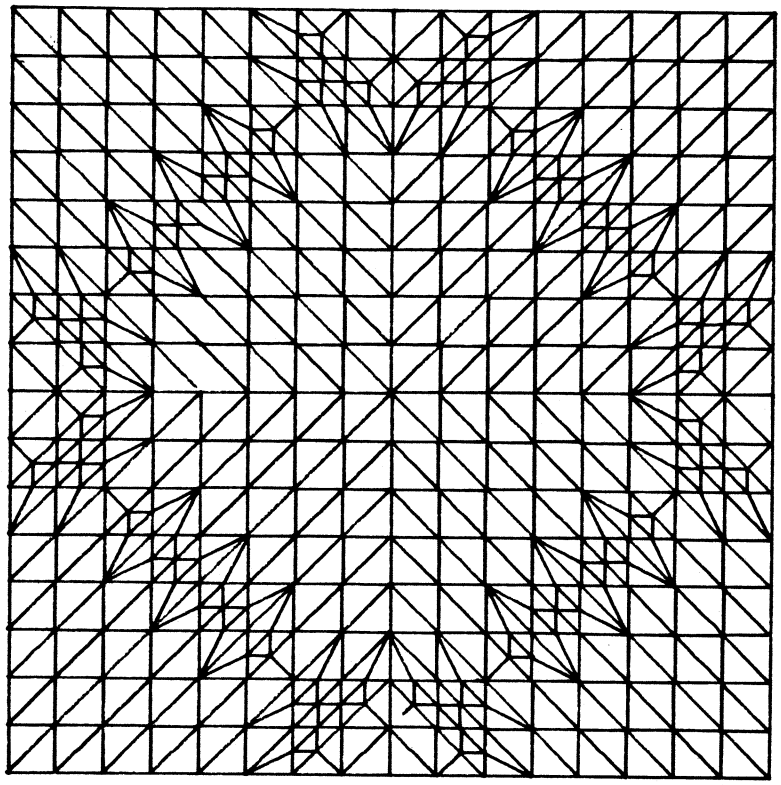


Figure 4-5: Fine Grids for $-\Delta u = \lambda u$

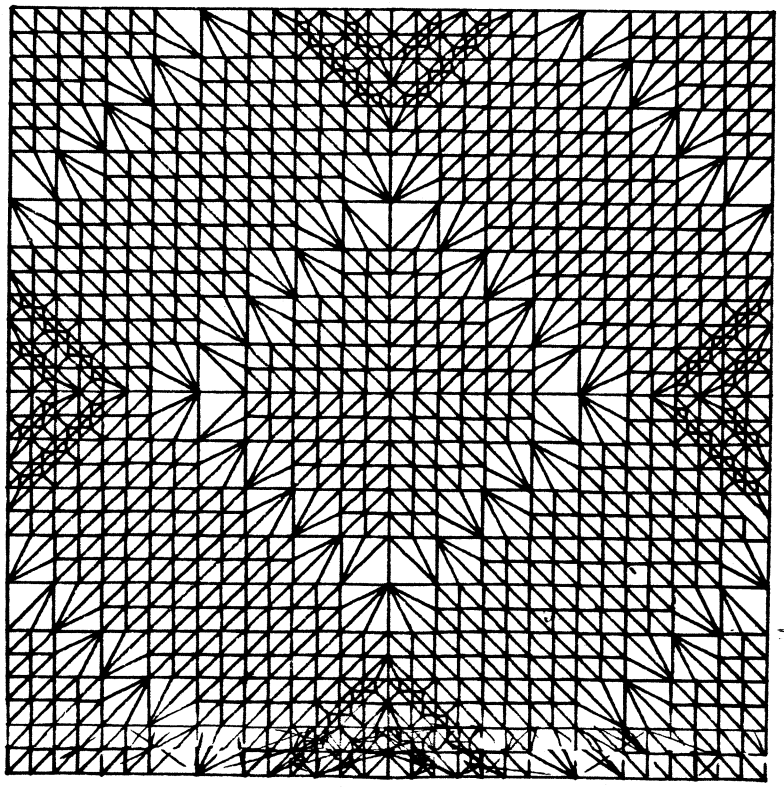
LEVEL 2

Number of
vertices = 425



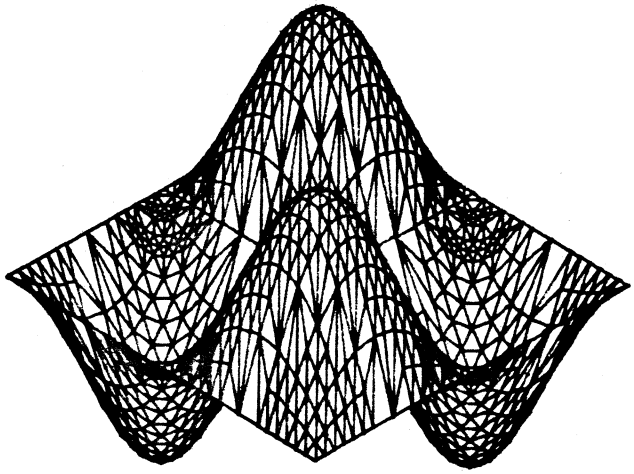
LEVEL 3

Number of
vertices = 1105



1. The solution u here is the normalized eigenfunction on the coarsest grid. To get better accuracy, we refine to 3 multigrid levels using $\theta = 2$ (constant ||u||). The grids generated are given in Figure 4-5. Notice that the eigenvalue $\lambda = 79.7$ on the finest grid is a rather good approximation to the true value of $\lambda = 8\pi^2$.

Figure 4-6: 4th eigenfunction of $-\Delta u = \lambda u$.



A plot of the eigenfunction is given in Figure 4-6.

A final example is for the problem:

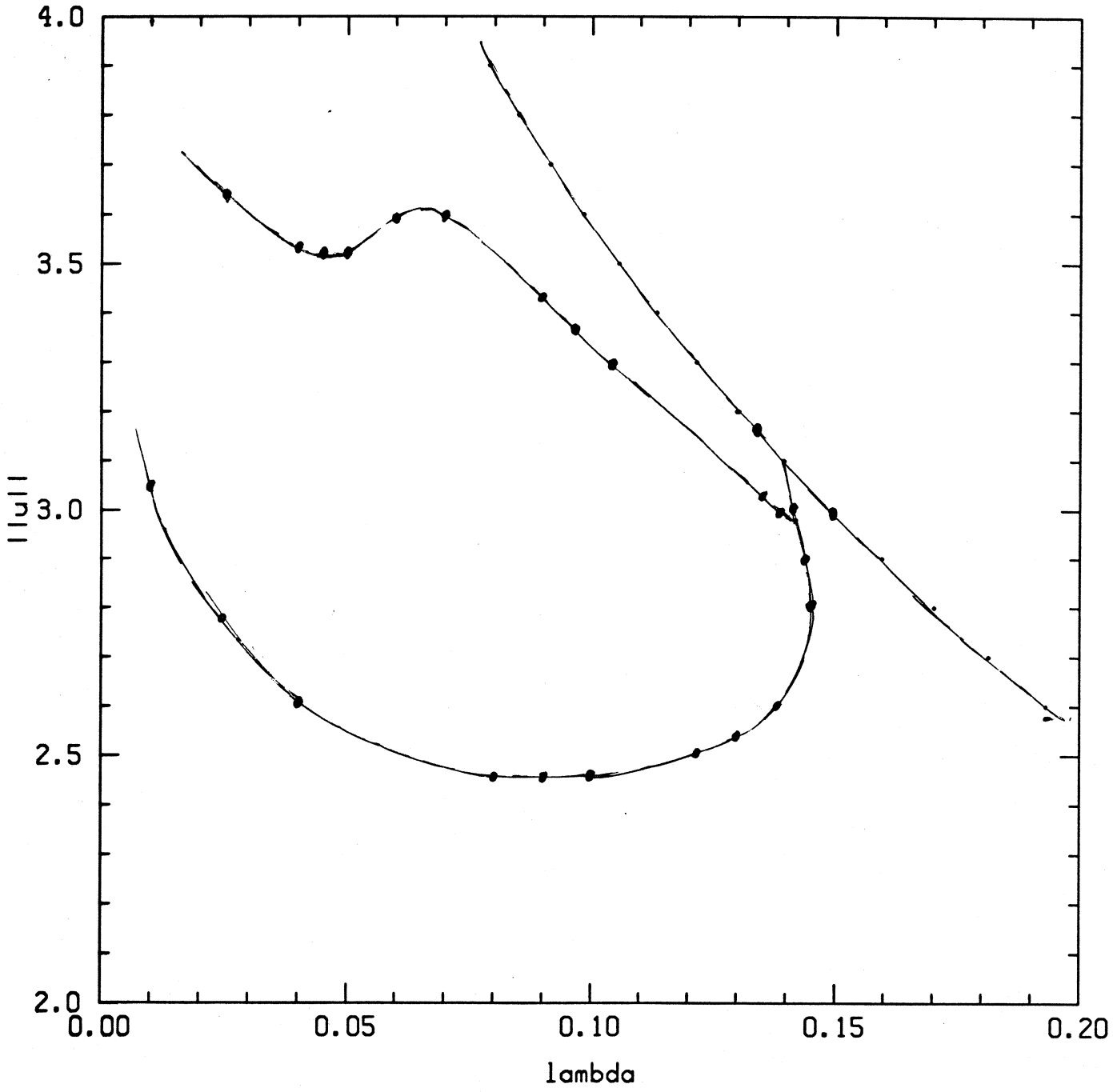
$$-\Delta u + 10\,(u - \lambda e^u) = 0$$

on the unit square in R^2 with homogeneous Neumann boundary conditions. This problem has more complicated behaviour than the other examples and illustrates the branch-switching capabilities of PLTMGC. There is a main branch of constant solutions satisfying the scalar equation

$$u = \lambda e^u.$$

On the upper part of this main branch, there is a symmetry breaking bifurcation point. Moreover, on this secondary branch there is another symmetry breaking bifurcation very close to the first bifurcation point. This part of the bifurcation diagram is shown in Figure 4-7 for the 5 by 5 grid shown in Figure 4-2 where the continuation points used by PLTMGC are also shown.

Figure 4-7: Bifurcation Diagram for $-\Delta u + 10(u - \lambda e^{u}) = 0$



5. Extensions

Although the current version of PLTMGC is quite complete in terms of its capabilities, we feel there are still some extensions that would enhance the usability of the package.

First, it is possible that one may want to locate singular points accurately on the finer meshes. This would appear to require a nontrivial computation on each level, and could involve several strategies, e.g. varying θ in N_r in an attempt to "hit" the limit point, or perhaps the application of the bisection/secant method to $\dot{\lambda}(\sigma) = 0$ or $\delta(\sigma) = 0$ directly on the finer meshes. This extension should be straight-forward.

In our experience, the overall efficiency and robustness of the continuation procedure depends critically on the predictor. In fact, it is a little dissatisfying that the current predictor, while surprisingly efficient and robust, is rather ad hoc. Our previous attempts to provide a predictor/step picker with more theoretically justification [22] did not seem to perform as well for PDE type problems.

Lastly, PLTMGC can only handle simple singular points. Extensions to higher order singularities like multiple bifurcation, cusps and following paths of singular points seem to be natural.

References

[1] J.P. Abbott. An Efficient Algorithm for the Determination of Certain Bifurcation Points. *Journal of Computational and Applied Mathematics* 4 :19-27, 1978.

[2] E. Allgower and K. Georg. Simplicial and Continuation Methods for Approximating Fixed Points and Solutions to Systems of Equations. *SIAM Review* 22(1):28-85, 1980.

[3] H. Amann. Fixed Point Equations and Nonlinear Eigenvalue Problems in Ordered Banach Spaces. *SIAM Review* 18:620-709, 1976.

[4] I. Babuska and W.C. Rheinboldt. A Posteriori Error Estimates for the Finite Element Method. *Int. J. Numer. Meth. Eng.* 12:1597-1615, 1975.

[5] I. Babuska and W.C. Rheinboldt. Error Estimates for Adaptive Finite Element Computations. *Siam J. Numer. Anal.* 15:736-754, 1978.

[6] I. Babuska and W.C. Rheinboldt. Reliable Error Estimation and Mesh Adaptation for the Finite Element Method. In J.T. Oden, Editor, *Computational Methods in Nonlinear Mechanics*, North Holland, New York, 1980 , pp. 67-108.

[7] R.E. Bank. *PLTMG User's Guide, June, 1981 version.* Technical Report, Dept. of Mathematics, University of Calif. at San Diego, 1982.

[8] R.E. Bank. A Comparison of Two Multi-level Iterative Methods for Nonsymmetric and Indefinite Elliptic Finite Element Equations. *Siam J. Numer. Anal.* 18:724-743, 1981.

[9] R.E. Bank. Analysis of a Multilevel Inverse Iteration Procedure for Eigenvalue Problems. *SIAM J. Numer. Anal.* 19:886-898, 1982.

[10] R E. Bank and T. F. Chan. *Multi-Grid Deflation.* 1984. In preparation.

[11] R.E. Bank and T.F. Dupont. An Optimal Order Process for Solving Finite Element Equations. *Math. Comp.* 36:35-51, 1981.

[12] R.E. Bank and D.J. Rose. Global Approximate Newton Methods. *Numer. Math.* 37:279-295, 1981.

[13] R.E. Bank and D.J. Rose. Analysis of a Multilevel Iterative Method for Nonlinear Finite Element Equations. *Math. Comp.* 39:453-465, 1982.

[14] R.E. Bank and A. Weiser. *A Posteriori Error Estimates in the Finite Element Method.* 1983. Submitted to Math. Comp.

[15] A. Brandt. Multi-level Adaptive Solution to Boundary Value Problems. *Math. Comp.* 31:333-390, 1977.

[16] T.F. Chan. Techniques for Large Sparse Systems Arising from Continuation Methods. In T. Kupper, H. Mittelmann and H. Weber, Editors, *Numerical Methods for Bifurcation Problems*, International Series of Numerical Math., Vol. 70, Birkhauser Verlag, Basel, 1984 , pp. 116-128.

[17] T.F. Chan. Deflated Decomposition of Solutions of Nearly Singular Systems. *Siam J. Numer. Anal., 1984* 21(4) August 1984.

[18] T.F. Chan. Newton-Like Pseudo-Arclength Methods for Computing Simple Turning Points. *Siam J. Sci. Stat. Comp.* 5 (1) March 1984.

[19] T.F. Chan. Deflation Techniques and Block-Elimination Algorithms for Solving Bordered Singular Systems. *Siam J. Sci. Stat. Comp.* 5 (1) March 1984.

[20] T.F. Chan and H.B. Keller. Arclength Continuation and Multi-Grid Techniques for Nonlinear Eigenvalue Problems. *SIAM J. Sci. Stat. Comp.* 3(2):173-194, June 1982.

[21] R.S. Dembo, S. Eisenstat and T. Steihaug. Inexact Newton Methods. *SIAM J. Numer. Anal.* 18(2):400-408, 1982.

[22] C. Den Heijer and W.C. Rheinboldt. On Steplength Algorithms for a Class of Continuation Methods. *SIAM J. Numer. Anal.* 18:925-947, 1981.

[23] J.P. Fink and W.C. Rheinboldt. *On the Error Behavior of the Reduced Basis Technique for Nonlinear Finite Element Approximations.* Technical Report ICMA-82-37, Institute for Computational Mathematics and Applications, Dept. of Math. and Stat., Univ. of Pittsburgh, 1982.

[24] C.B. Garcia and W.I. Zangwill. *Pathways to Solutions, Fixed Points and Equilibria.* Prentice-Hall, Englewood Cliffs, N.J., 1981.

[25] W. Hackbusch. On the Fast Solution of Nonlinear Eliptic Equations. *Numer. Math.* 32:83-95, 1979.

[26] W. Hackbusch. On the Computation of Approximate Eigenvalues and Eigenfunctions of Elliptic Operators By Means of a Multi-Grid Method. *SIAM J. Numer. Anal.* 16:201-215, 1979.

[27] W. Hackbusch. Multi-Grid Solution of Continuation Problems. In R. Ansorge, T. Meis and W. Torning, Editors, *Iterative Solution of Nonlinear Systems*, Springer-Verlag, Berlin, 1982 .

[28] H. Jarausch and W. Mackens. *Computing solution Branches by Use of a Condensed Newton - Supported Picard Iteration Scheme.* Technical Report, Institute for Geometrie und Praktische Mathematik, Rheinisch-Westfalische Technische Hochschule, Aachen, 1983.

[29] H.B. Keller. Numerical Solution of Bifurcation and Nonlinear Eigenvalue Problems. In P. Rabinowitz, Editor, *Applications of Bifurcation Theory*, Academic Press, New York, 1977, pp. 359-384.

[30] H.B. Keller. Global Homotopies and Newton Methods. In Carl de Boor and Gene Golub, Editor, *Recent Advances in Numerical Analysis*, Academic Press, New York, 1978, pp. 73-94.

[31] M. Kubicek. Dependence of Solution of Nonlinear Systems on a Parameter. *ACM-TOMS* 2:98-107, 1976.

[32] Lois Mansfield. On the Solution of Nonlinear Finite Element Systems. *Siam J. Numer. Anal.* 17:752-765, 1980.

[33] R.G. Melhem and W.C. Rheinboldt. A Comparison of Methods for Determining Turning Points of Nonlinear Equations. *Computing* 29:201-226, 1982.

[34] H.D. Mittelmann and H. Weber. Numerical Methods for Bifurcation Problems - A Survey and Classification. In H.D. Mittelmann and H. Weber, Editors, *Bifurcation Problems and their Numerical Solution*, Birkhauser, Dortmund, 1980 , pp. 1-45.

[35] H.D. Mittelmann and H. Weber. *Multi-Grid Solution of Bifurcation Problems.* Technical Report 65, Abteiling Math., Univ. Dortmund, 1983. To appear in Siam J. Sci. Stat. Comp.

[36] A.K. Noor. Recent Advances in Reduction Methods for Nonlinear Problems. *Computers and Structures* 13:31-44, 1981.

[37] A.K. Noor and G.M. Peters. Reduced Basis Technique for Nonlinear Analysis of Structures. *AIAA Journal* 18:455-462, 1980.

[38] W.C. Rheinboldt. Numerical Methods for a Class of Finite Dimensional Bifurcation Problems. *SIAM J. of Numer. Anal.* 15(1):1-11, 1978.

[39] W.C. Rheinboldt. Solution Fields of Nonlinear Equations and Continuation Methods. *SIAM J. Numer. Anal.* 17:221-237, 1980.

[40] W.C. Rheinboldt. Numerical Analysis of Continuation Methods for Nonlinear Structural Problems. *Computers and Structures* 13:103-113, 1981.

[41] W.C. Rheinboldt and J.V.Burkardt. A Locally Parameterized Continuation Process. *ACM Trans. Math. Soft.* 9(2):215-235, June 1983.

[42] G.W. Stewart. On the Implicit Deflation of Nearly Singular Systems of Linear Equations. *SIAM J. Sci. Stat. Comp.* 2(2):136-140, 1981.

[43] L.T. Watson and D. Fenner. Chow-Yorke Algorithm for Fixed Points or Zeros of C^2 Maps. *ACM Trans. on Math. Software* 6:252-259, 1980.

[44] A. Weiser. *Local-Mesh, Local-Order, Adaptive Finite Element Methods with A Posteriori Error Estimates for Elliptic Partial Differential Equations.* Ph.D. Thesis, Yale University, 1982. Techreport #213.