Scalable Computing: Practice and Experience
Volume 10, Number 4, pp. 347–361. http://www.scpe.org
ISSN 1895-1767

CONDITIONING AND HYBRID MESH SELECTION ALGORITHMS FOR TWO-POINT BOUNDARY VALUE PROBLEMS*

JEFF R. CASH† AND FRANCESCA MAZZIA‡

Abstract. Boundary value problems for ordinary differential equations (BVODES) occur in a great many practical situations and they are generally much harder to solve than initial value problems. Traditionally codes for BVODES did not take into account the conditioning of the problem and it was generally assumed that the problem being solved was well conditioned so that small local errors gave rise to correspondingly small global errors. Recently a new generation of codes which take account of conditioning has been developed. However most of these codes are based on a rather ad hoc approach with the need to choose several heuristics without any real guidance on how these choices can be made. In this paper we identify clearly which heuristics need to be chosen and we discuss different choices of monitor functions that are used in our codes. This has the important effect of unifying the various approaches that have recently been proposed. This in turn allows us, in the present paper, to introduce a new technique for computing the conditioning which is ideally suited to BVODES.

1. Introduction. The task of solving systems of nonlinear two-point boundary value problems numerically has, for a long time, received a great deal of attention. Boundary value problems for ordinary differential equations (BVODES) occur in a great many practical situations and they are generally much harder to solve than initial value problems. In particular the numerical solution of singular perturbation problems can be especially difficult because such equations can have solutions with very narrow regions of rapid variation typified by boundary layers, shocks and interior layers.

Traditionally codes for BVODES did not take into account the conditioning of the problem and it was generally assumed that the problem being solved was well conditioned so that small local errors gave rise to correspondingly small global errors. However, in an important paper by Shampine and Muir [28], the need to consider the conditioning of the problem being solved was clearly demonstrated by means of a numerical example and this brought into focus some important earlier work by Brugnano and Trigiante [5, 6] and by Mazzia and Trigiante [27] who derived codes with a monitor function depending on both conditioning and error. Subsequently a new generation of codes which take account of conditioning has been developed. However most of these codes are based on a rather ad hoc approach with the need to choose several heuristics without any real guidance on how these choices can be made. In this paper we identify clearly which heuristics need to be chosen and we discuss different choices of monitor functions that are used in our codes. This has the important effect of unifying the various approaches that have recently been proposed. This in turn allows us, in the present paper, to introduce a new technique for computing the conditioning parameters which is ideally suited to BVODES. Finally we describe some codes which implement these new ideas and we give some numerical results obtained from using these codes.

We will only consider in this paper singular perturbation boundary value problems, although the algorithms developed are designed for general first order boundary value problems. We will not be concerned in this paper with shooting methods but will instead confine our attention to so called finite difference or boundary value methods. However a lot has been done on estimating the conditioning of shooting methods and the interested reader is referred to [1, 20, 21, 19, 13].

In section 2 we give a review of the algorithms based on conditioning for singularly perturbed boundary value problems. In section 3 we present the conditioning parameters for the continuous problem and in section 4 the corresponding parameters for the discrete one. In section 5 we analyze the hybrid mesh selection strategies based on conditioning and the way they have been unified and updated. In section 6 we present the results of some numerical experiments to analyze the behavior of the codes with the new condition estimator.

2. Singular Perturbation Problems.

2.1. Second Order Scalar Problems.

The first attempt to use conditioning in algorithms for the solution of singularly perturbed BVODES was made in [24, 8] by Mazzia and Trigiante. The algorithm presented in [24, 8] was designed for the following general class of scalar linear problems:

*Work developed within the project "Numerical methods and software for differential equations"
†Department of Mathematics, Imperial College, South Kensington, London SW7, England
‡Dipartimento di Matematica, Università di Bari, Via Orabona 4, I-70125 Bari, Italy

$$\begin{aligned} &\epsilon y'' + p(x)y' + q(x)y = f(x) \\ &y(a) = y_a, \;\; y(b) = y_b, \;\; a \le x \le b, \end{aligned} \tag{2.1}$$

where ϵ is a positive parameter which is small compared with $b-a$, $q(x)$ and $f(x)$ are continuous functions and $p(x)$ is differentiable.

If we discretize (2.1) using simple three point finite difference formulae, the discrete problem is a tri-diagonal system of algebraic equations that can be solved to give an approximation to the solution of (2.1). In matrix form this system is

$$\mathbf{T}\boldsymbol{y} = f \tag{2.2}$$

where $y_0, y_1, \ldots, y_n$ is the numerical solution computed on the mesh $\pi = \{x_0, x_1, \ldots, x_n\}$, $\boldsymbol{y} = (y_1, y_2, \ldots, y_n)^T$ and

$$\mathbf{T} = \begin{pmatrix} 1 & \tau_1 & & & \\ \sigma_1 & 1 & \tau_2 & & \\ & \sigma_2 & 1 & \ddots & \\ & & \ddots & \ddots & \tau_{n-1} \\ & & & \sigma_{n-1} & 1 \end{pmatrix}. \tag{2.3}$$

The matrix T depends on π and the algorithm presented in [24, 8] uses sufficient conditions for the well conditioning of tridiagonal matrices derived in [3] to compute the mesh. This allows us to derive, for example, the conditions that a constant stepsize, h, should satisfy in order to have T well conditioned. In general we require $h \approx O(\sqrt{\epsilon})$. However in the much more important case where we solve (2.1) using a variable meshsize the mesh must be choosen in order to have the $n \times n$ system (2.2) such that $n \ll \dfrac{1}{\epsilon}$.

In [24, 8] Mazzia and Trigiante, using only information related to the well conditioning of the tridiagonal matrix T, derive second order methods for the solution of (2.1) where the grid is chosen so that

1. The user requested accuracy is achieved.
2. The inverse coefficient matrix T^{-1} exists and its condition number is either independent of n (well conditioned) or grows as n or n^2 (weakly well conditioned).

The way that these two conditions are satisfied is to choose h_i in order to have (σ_i, τ_i) that satisfy the conditions for the (weakly) well-conditioning of T. The algorithm simplifies considerably if σ_i and τ_i have constant sign. We note that in a classical approach, where ϵ is not small, it would be normal to choose σ_i and τ_i so that the matrix T is diagonally dominant. However in this case h is of the order ϵ and n is of order $\frac{1}{\epsilon}$ and this is unacceptable when ϵ is very small. An algorithm that chooses τ_i and σ_i so that 1) and 2) are satisfied is described in [8] and some numerical results presented for singular perturbation problems show the power of the method.

2.2. First Order Systems of Singular Perturbation Problems. Although the algorithm described in the previous section represented a major step forward in the solution of singular perturbation problems it has the disadvantage that it is only applicable for linear, scalar, second order systems, it is of low order and it is not very easy to apply for variable stepsizes. However the strategy of using conditioning in choosing the mesh size is an important one and in the following sections we will discuss how to deal with nonlinear systems of BVODES.

In what follows we will be concerned with the nonlinear system of singular perturbation problems

$$D(\epsilon)y'(x) = f(x, y), \;\; a \le x \le b, \quad g(y(a), y(b)) = 0, \tag{2.4}$$

where $D(\epsilon)$ is a diagonal matrix whose elements depend linearly on ϵ. Usually $D(\epsilon) = diag(1, \ldots, 1, \epsilon)$, and $y, f,\ g \in R^m$. As in the previous section we will be particularly interested in the case where $0 < \epsilon \ll 1$. An efficient numerical method for the solution of (2.4) needs to be able to use non-uniform grids so that many

grid points are clustered in regions of rapid variation of the solution and relatively few grid points are placed in regions where the solution is smooth. Most codes attempt to control the error in the solution either by estimating the discrete error at the mesh points [7] or else by controlling a residual [28]. The important point to realise is that in both cases the codes attempt to control the local error on the assumption that the problem is well conditioned so that if the local error is small then the global error will also be small. However in the case where the problem is ill-conditioned a standard backward error analysis shows that it is possible for an accepted solution to have a small local error but to have an unacceptably large global error. There is the additional problem that if a monitor function ([1], p. 363) takes no account of conditioning of a problem then the grid choosing algorithm may become very inefficient. This is manifested by a sort of cycling where points are added into the grid on conditioning considerations and are then removed in the next remeshing due to accuracy considerations. To illustrate these ideas we present a test problem which we solve by using the code TWPBVPC [11] both with the standard mesh selection strategy based on the estimation of the local error and with a mesh selection strategy which considers both conditioning and local error estimation. We note that to change from the code TWPBVPC, which takes account of conditioning, to TWPBVP, which uses a conventional mesh choosing strategy, we need to change just one input parameter of the code TWPBVPC.

Example 1. We examine the following problem [11] :

$$\begin{aligned} &\epsilon y'' + xy' = -\epsilon\pi^2\cos(\pi x) - \pi x\sin(\pi x), \\ &y(-1) = -2, y(1) = 0, \end{aligned} \tag{2.5}$$

whose exact solution is $\cos(\pi x) + \mathrm{erf}(x/\sqrt{2\epsilon})/\mathrm{erf}(1/\sqrt{2\epsilon})$, solved using the code TWPBVP when conditioning is not taken into account. The problem is rewritten as a first order system with two components $(y, y')^T$ and the input parameters are $\epsilon = 10^{-7}$ and $tol(y) = 10^{-8}$. This means that we are seeking an approximation to the solution with an error $0.5 \cdot 10^{-8}$ or less in the y component. The code gives a solution using a final mesh of 4192 points with a maximum error of $0.17 \cdot 10^{-8}$. The mesh sizes used in the intermediate steps of the computation are: 16, 31, 61, 121, 241, 481, 530, 1059, 1108, 2215, 2242, 4483, 8965, 19912, 4192. It seems that the code finds the problem very difficult to solve and it needs 19912 mesh points to get some worthwhile information about the solution. This is mainly due to the fact that the mesh selection algorithm adds extra points in the wrong place. If we use the same code with the mesh selection based on conditioning, and we call this TWPBVPC, the solution is obtained using 368 mesh points with maximum error of $0.85 \cdot 10^{-8}$. The meshes used are of sizes: 16, 31, 58, 85, 169, 368. The information given by the conditioning parameters allows the code to put the mesh points in the correct place and makes it considerably more efficient for the solution of this problem.

The first papers to consider this problem of estimating the conditioning constants in a serious way were by Brugnano and Trigiante [4, 5] and by Mazzia and Trigiante [27]. Their basic approach is to identify two constants which characterise the conditioning of the continuous problem. Having done this the monitor function used in the mesh choosing algorithm is based on the relative size of these two parameters as well as on a local error estimate. The basic aim of the algorithm described in [27] is to choose the mesh so that the discrete and continuous problems have conditioning parameters which have the same order of magnitude. Perhaps the first really powerful code that used a monitor function based on accuracy and conditioning was the code TOM [27]. Numerical results presented in [27] show the excellent performance of this code compared with the case where conditioning is not considered. Using ideas explained in [27] the two deferred correction codes TWPBVP and TWPBVPL were modified to include conditioning in their monitor function and the much improved performance of the modified codes TWPBVPC and TWPBVPLC can be seen from the results on the authors' web page [11].

3. Conditioning parameters. To introduce the conditioning parameters which will be used in the mesh selection strategy of our codes, let us consider for simplicity the following linear boundary value problem:

$$\frac{dy}{dx} = A(x)y(x) + q(x), \quad a \le x \le b, \quad B_a y(a) + B_b y(b) = \beta,\ \beta \in R^m \tag{3.1}$$

whose solution is given by

$$y(x) = Y(x)Q^{-1}\beta + \int_a^b G(x,t)q(t)dt. \tag{3.2}$$

Here $Y(x)$ is a fundamental solution, $Q = B_a Y(a) + B_b Y(b)$ is non singular and $G(x,t)$ is the Greens' function. Using the ∞-norm we can compute the conditioning parameter by considering a perturbed equation:

$$\frac{du}{dx} = A(x)u + q(x) + \delta(x),\ a \le x \le b, \quad B_a u(a) + B_b u(b) = \beta + \delta\beta.$$

Here $\delta(x)$ and $\delta\beta$ are small perturbations of the data. The difference between the two solutions satisfies:

$$||u(x) - y(x)|| \le ||Y(x)Q^{-1}\delta\beta|| + ||\int_a^b G(x,t)\delta(t)dt||. \tag{3.3}$$

After some algebraic manipulation we obtain:

$$\max_{a\le x\le b} ||u(x) - y(x)|| \le \kappa_1 \|\delta\beta\| + \kappa_2 \max_{a\le x\le b} ||\delta(x)||,$$

and

$$\max_{a\le x\le b} ||u(x) - y(x)|| \le \kappa \max(\|\delta\beta\|, \max_{a\le x\le b} ||\delta(x)||),$$

where

$$\kappa_1 = \max_{a\le x\le b} ||Y(x)Q^{-1}||, \qquad \kappa_2 = \sup_x \int_a^b ||G(x,t)||dt,$$

and

$$\kappa = \max_{a\le x\le b} (||Y(x)Q^{-1}|| + \int_a^b ||G(x,t)||dt).$$

Following the same procedure as above and using the 1-norm we obtain

$$\frac{1}{b-a}\int_a^b ||u(x) - y(x)||dx \le \gamma_1 \|\delta\beta\| + \gamma_2 \max_{a\le x\le b} ||\delta(x)||,$$

and

$$\frac{1}{b-a}\int_a^b ||u(x) - y(x)||dx \le \gamma \max(\|\delta\beta\|, \max_{a\le x\le b} ||\delta(x)||),$$

where

$$\gamma_1 = \frac{1}{b-a}\int_a^b ||Y(x)Q^{-1}||dx, \qquad \gamma_2 = \frac{1}{b-a}\int_a^b\int_a^b ||G(x,t)||dtdx,$$

and

$$\gamma = \frac{1}{b-a}\int_a^b (||Y(x)Q^{-1}|| + \int_a^b ||G(x,t)||dt)dx.$$

For many problems of interest the relative sizes of the two parameters κ and γ tell us about the conditioning of the continuous problem. However in the discrete case the parameter γ is very difficult to compute. In general, in a problem where the correct dichotomy is present only κ_1 and γ_1 are of interest. In [4] a problem with κ_1 large and γ_1 small was called stiff, the stiffness ratio being $\dfrac{\kappa_1}{\gamma_1}$. This definition, although very often giving the correct information about the stiffness, may fail for non scalar problems, as pointed out in [6]. In [18] Iavernaro, Mazzia and Trigiante give a slightly different definition of stiffness. The conditioning parameters presented in [18], called $\kappa_{1,c}([a,b])$ and $\gamma_{1,c}([a,b])$, only depend on the perturbations of the boundary conditions, and are defined by supposing that $\delta(x) \equiv 0$ in (3.3). The way in which these parameters are defined is as follows:

$$\begin{aligned}
\kappa_{1,c}([a,b],\delta\beta) &= \frac{\max_{a\le x\le b}\|u(x)-y(x)\|}{\|\delta\beta\|}, \qquad & \kappa_{1,c}([a,b]) &= \max_{\delta\beta}\kappa_{1,c}([a,b],\delta\beta),\\
\gamma_{1,c}([a,b],\delta\beta) &= \frac{\int_a^b\|u(x)-y(x)\|dx}{(b-a)\|\delta\beta\|}, & \gamma_{1,c}([a,b]) &= \max_{\delta\beta}\gamma_{1,c}([a,b],\delta\beta).
\end{aligned} \tag{3.4}$$

Upper bounds on $\kappa_{1,c}([a,b]$ and $\gamma_{1,c}([a,b])$ can be obtained in terms of the parameters previously introduced, and it can be shown that

$$\kappa_{1,c}([a,b]) \le \kappa_1, \qquad \gamma_{1,c}([a,b]) \le \gamma_1. \tag{3.5}$$

This definition of $\kappa_{1,c}([a,b],\delta\beta)$ and $\gamma_{1,c}([a,b],\delta\beta)$ allows us to define the stiffness ratio which is defined as

$$\sigma_c([a,b]) = \max_{\delta\beta}\frac{\kappa_{1,c}([a,b],\delta\beta)}{\gamma_{1,c}([a,b],\delta\beta)}.$$

With these parameters it is possible to give the definitions of well conditioned, stiff and ill conditioned problems, see [18, 23] for details:

Well conditioned: κ, κ_1, γ_1 and $\sigma_c([a,b])$ are of moderate size;
Stiff: $\sigma_c([a,b]) \gg 1$;
Ill conditioned: $\kappa \gg 1$ and $\gamma \gg 1$.

If $\sigma_c([a,b])$ is large we are dealing with problems possessing different time scales for which the growth or decay rates of some fundamental solution modes are very rapid compared to others. Many singularly perturbed BVODES have $\sigma_c([a,b])$ large. Our aim is to choose the mesh so that the continuous and discrete problems have similar conditioning parameters. This leads us to investigate the conditioning of the discrete problem and this we do in the next section.

4. Conditioning parameters for the discrete problems. In order to use the conditioning parameters in a mesh selection strategy we first need to compute a discrete approximation to them which can be used in our numerical method. If in our algorithm we use a Newton iteration scheme to solve the nonlinear algebraic equations we need to solve a linear system of algebraic equations of the form $My = b$ for each iteration. The matrix M depends on the numerical scheme and on the stepsize used. We use a grid $\pi = \{x_0, x_1, \ldots, x_n\}$ with grid spacing $h_i = x_i - x_{i-1}$, $i = 1, \ldots, n$ on which to solve the problem. The block matrix M, of size $(n+1)m \times (n+1)m$, is set up so that the boundary conditions appear only in the first row block of b. Ascher, Mattheij and Russell [1] prove that for one-step schemes $\|M^{-1}\|_\infty \approx \kappa$ and this is a fundamentally important result. For the computation of $\|M^{-1}\|$ we have used up to now the algorithm presented by Higham in [15], which computes an approximation to the 1-norm of a matrix. This algorithm has now been optimized in order to use the information that we already know to compute the estimation of κ_1 and γ_1. Using the definition given in (3.4) it is possible to define the values of $\kappa_{1,d}(\pi,\delta\beta)$, $\gamma_{1,d}(\pi,\delta\beta)$ as:

$$\begin{aligned}
\kappa_{1,d}(\pi,\delta\beta) &= \tfrac{1}{\|\delta\beta\|}\max_{0\le i\le n}\|y_i\|, & \kappa_{1,d}(\pi) &= \max_{\delta\beta}\kappa_{1,d}(\pi,\delta\beta),\\
\gamma_{1,d}(\pi,\delta\beta) &= \tfrac{1}{(b-a)\|\delta\beta\|}\sum_{i=1}^n h_i\max(\|y_i\|,\|y_{i-1}\|), & \gamma_{1,d}(\pi) &= \max_{\delta\beta}\gamma_d(\pi,\delta\beta),
\end{aligned} \tag{4.1}$$

where $y_i, i = 0, 1, \ldots, n$ is the solution of the discrete problem having $\delta\eta$ as boundary conditions and $\kappa_{1,d}(\pi)$ and $\gamma_{1,d}(\pi)$ are the discrete approximations of κ_1 and γ_1 respectively. The *discrete stiffness ratio* is:

$$\sigma_d(\pi) = \max_{\delta\beta}\frac{\kappa_{1,d}(\pi,\delta\beta)}{\gamma_{1,d}(\pi,\delta\beta)}.$$

Since in the numerical codes we need to use easy to compute parameters, we define upper and lower bounds for them using the information given by the matrix M^{-1}. We define the matrices $G = M^{-1}$ and Ω having elements $\Omega_{ij} = ||G_{ij}||_\infty$ of size $(n+1)\times(n+1)$. Here G_{ij} is the i,jth block element of size m appearing in G.

The discrete approximations to κ_1 and γ_1 satisfy the following upper bounds that correspond to the discrete parameters computed in [27, 5, 6, 8, 9]:

$$\kappa_{1,d}(\pi) \leq \max_i \Omega_{i0}, \qquad \gamma_{1,d}(\pi) \leq (\sum_{i=1}^{N} h_i \max(\Omega_{i-1,0}, \Omega_{i0}))/(b-a). \tag{4.2}$$

In the following we approximate the discrete conditioning parameters by their upper bounds. Moreover, taking into account the relation between the matrix G and the Green's function, a discrete approximation of κ_2 is given by $\|g^r_{m+1,(n+1)m}\|_1$ (we consider only the components from $m+1$ to $(n+1)m$), where g^r is the row of the matrix G such that $\|g^r\|_1 = \|G\|_\infty$. Since the first block column of the matrix G_{*0} is a discrete approximation of $Y(t)Q^{-1}$ we have that the *i-th* column of G_{*0} is an approximation of the solution of the continuous problem (3.3) when $\delta(x) = 0$ and $\delta\beta = e_i$, with e_i being the column i of the identity matrix of size m. We are able now to define, denoting by $g^{(i)}$ the *i-th* column of G_{*0}, and by $g^{(i)}_j, 0 \leq j \leq N$ its blocks of size m,

$$\kappa_{1,d}(\pi, e_i) = \|g^{(i)}\|_\infty,$$

and

$$\gamma_{1,d}(\pi, e_i) = \frac{1}{b-a} \sum_{j=1}^{n} (x_j - x_{j-1}) \max(\|g^{(i)}_{j-1}\|_\infty, \|g^{(i)}_j\|_\infty).$$

A lower bound for $\sigma_d(\pi)$ is computed as

$$\sigma_d(\pi) \geq \max_{e_i, 1 \leq i \leq m} \frac{\kappa_{1,d}(\pi, e_i)}{\gamma_{1,d}(\pi, e_i)}. \tag{4.3}$$

We note that the new value of $\sigma_d(\pi)$ differs from the one computed in [6] because, instead of considering the ratio $\frac{\kappa_{1,d}(\pi)}{\gamma_{1,d}(\pi)}$ involving the norm of G_{i0}, we consider the columns of G_{*0} separately. This allows us to retain information that would be lost by using directly the norm of each block.

By construction $\sigma_d(\pi)$ is a discrete approximation of the following continuous lower bound of $\sigma_c([a,b])$:

$$\sigma_c([a,b]) \geq \max_{e_i, 1 \leq i \leq m} \frac{\kappa_{1,c}([a,b], e_i)}{\gamma_{1,c}([a,b], e_i)}.$$

In the following we approximate the discrete value of $\sigma_d(\pi)$ by the lower bound given in (4.3). One of the important advantages of computing this approximation of the discrete conditioning parameters is that it is very inexpensive to implement since it requires only one block column of G to be computed. In what follows we describe how to use this information to compute an estimate of $\|G\|_\infty$ which allows us to obtain a discrete approximation of κ. In this paper we describe the algorithm only for the one step formulae (implemented in TWPBVPLC and TWPBVPC). In what follows we give in detail the algorithm implemented by Higham in [15] for the estimation of the one norm of a matrix. This algorithm is a variant of the original algorithm derived by Hager in [14]. We apply it to the transpose matrix, to compute an approximation of the infinity norm. This algorithm, given $G \in \mathbb{R}^{N \times N}$, computes $\kappa_d(\pi)) \leq \|G\|_\infty$ and $\|Gx\|_\infty = \kappa_d(\pi)\|x\|_\infty$

The flow chart (matlab-like) for doing this is given below (see [16]):

function $\kappa_d(\pi)$ = *KappaHigham(G,N)*
  $\zeta = N^{-1} ones(N,1)$
  $g^r = G^T \zeta$
  $\kappa_d = \|g^r\|_1$
  $\xi = sign(g^r)$
  $z = G\xi$
  $nstep = 2$
  while nstep <= 5
    $\zeta = e_j$, *where* $|z_j| = \|z\|_\infty$ *(smallest j)*

$g^r = G^T\zeta$
$\kappa_d^n = \|g^r\|_1$
if $sign(g^r) = \xi$ *or* $\kappa_d^n(\pi) < \kappa_d(\pi)$, *goto (*), end*
$\kappa_d = \kappa_d^n$
$\xi = sign(g^r)$
$z = G\xi$
$nstep = nstep + 1$
if $|z_j| = \|z\|_\infty$, *break, end*
end
(*) $z_i = (-1)^{i+1}(1 + (i-1)/(N-1)), i = 1 : N$
$z = G^T z$
if $2\|z\|_1/(3N) > \kappa_d(\pi)$ *then*
$g^r = z$
$\kappa_d(\pi) = 2\|z\|_1/(3N)$
end

In the following we present a variant of the Higham algorithm that uses the information given by the conditioning parameters. Since to compute $\kappa_{1,d}(\pi)$ we need the infinity norm of the vector Ω_{*0}, which depends on the first m columns of G, we could easily compute the index j_k in which Ω_{*0} reaches its maximum, that is $\Omega_{j_k 0} = \kappa_{1,d}(\pi) = \max_i \Omega_{i0}$, and the index k such that $\|(G_{j_k,0})_{k*}\|_\infty = \Omega_{j_k 0}$. We call g^r the row of index $i_k = (j_k - 1)m + k$ of the matrix M^{-1}, ($g^r = G^T e_{i_k}$). In the case of separated boundary conditions, and it is these boundary conditions that are allowed in the codes, if the problem has an exponential dichotomy and it is well conditioned, we have that κ_1 gives all the required information about the conditioning, and the Green's function in (3.2) decays exponentially with $|x - t|$ (see [1] p.126). Since for stiff problems Ω_{*0} reaches its maximum in one isolated element we conclude that the 1-norm of g^r is a good approximation to $\|G\|_\infty$. In this case $\kappa_{1,d}(\pi) = \|g^r_{1:m}\|_1$ by construction and $\kappa_{2,d}(\pi) = \|g^r_{m+1:(n+1)m}\|_1$ is an approximation to κ_2. However for non stiff problems Ω_{*0} could have many elements equal to the maximum, and if $\kappa_{2,d}(\pi)$ is of the same order as $\kappa_{1,d}$ its contribution to $\kappa_d(\pi)$ is important. We have modified the Higham algorithm in order to use this information given by the stiffness parameters. The new algorithm is:

function $[\kappa_d(\pi), \kappa_{2,d}(\pi)] =$ *KappaStiffBvp(G,N,m,*i_k*,*$\kappa_{1,d}(\pi)$*,* $\sigma_d(\pi)$*)*
$nstep = 1$
if $\sigma_d(\pi) < 10$
$\zeta = e_{i_k}$
else
$\zeta = ones(N, 1)/(N + 11)$, $\zeta_{i_k} = 11/(N + 11)$
end
$g^r = G^T\zeta$
$\kappa_d(\pi) = \|g^r\|_1$
$\kappa_{2,d}(\pi) = \|g^r_{m+1:N}\|_1$
while $(\sigma_d(\pi) < 10 \mid \kappa_{2,d}(\pi) > \kappa_{1,d}/10)$ & $nstep \le 3$
$\xi = sign(g^r)$
$z = G\xi$
if $\|z\|_\infty \le z^T\zeta$
break
end
$\zeta = e_j$, *where* $|z_j| = \|z\|_\infty$ *(smallest j)*
$g^r = G^T\zeta$
$\kappa_d^n(\pi) = \|g^r\|_1$
if $\kappa_d^n(\pi) < \kappa_d(\pi)$, *break, end*
$\kappa_d(\pi) = \kappa_d^n(\pi)$
$\kappa_{2,d}(\pi) = \|g^r_{m+1:N}\|_1$
$nstep = nstep + 1$
end

We note that a similar modification could be applied to the block algorithm to estimate the one norm presented in [17], we have however explained only the algorithm which is used in the current version of the codes.

Since the matrix that has already been factorized in the codes is of the form $\tilde{M} = DM$, where D is a diagonal matrix with blocks $D_0 = I$, $D_i = h_i I, i = 1, \ldots, N$, where h_i are the gridsizes used (for simplicity we suppose that the boundary conditions are in the first block row), we need to compute an approximation of $\|M^{-1}\|_\infty$, knowing the factorization of $\tilde{M}$. To do this we apply the algorithm previously described for the computation of $\|(\tilde{M}^{-1}D)\|_\infty$.

We have updated the codes TWPBVPC and TWPBVPLC in order to compute $\sigma_d(\pi)$ using the approximation in (4.3), and $\kappa_d(\pi)$ and $\kappa_{2,d}(\pi)$ using the new algorithm *KappaStiffBvp* and these are available on the authors' web page.

5. Hybrid mesh selection strategies based on conditioning. At present there exist just a few codes that implement hybrid mesh selection strategies, and these include TOM, TWPBVPC and TWPBVPLC. We have structured our approach to mesh selection so that the strategies described in this paper are common to all three codes being considered with the only things changing being the values of several variables which need to be chosen heuristically. The common approach is to choose the mesh in order to have a discrete problem with conditioning parameters similar to those of the continuous problem. That is, we need to choose the mesh so that the discrete monitor function used by all three codes, when only the conditioning is taken into account, is:

$$\psi(x_i) = |\Omega_{i0} - \Omega_{i-1,0}| + \alpha \tag{5.1}$$

where $\alpha = \frac{p}{(1-p)(b-a)} \sum_{i=1}^{N} |\Omega_{i0} - \Omega_{i-1,0}|$.

5.1. Mesh selection for Deferred Correction Formulae. The code TWPBVPL is a deferred correction code based on Lobatto IIIa formulae of order 4,6 and 8 [12]. This code is an extension of the one presented in [2], and this is considerably more robust than TWPBVPC, especially for singularly perturbed boundary value problems. For this reason these Lobatto deferred correction algorithms have also been implemented in the code ACDC [10] in a continuation framework, in order to deal with extremely stiff problems. The deferred correction scheme implemented in TWPBVPL has many similarities with that used in TWPBVPC, and as a result the hybrid mesh selection strategy derived in TWPBVPC has been implemented for TWPBVPLC using a similar procedure. The two hybrid strategies have been described in detail in [8, 12]. However since the underlying numerical schemes used in TWPBVPC and TWPBVPLC are very different it is important to set up some empirical parameters in order to have an efficient mesh selection for each code. In what follows we will describe some of these parameters which have been chosen as a result of extensive numerical experimentation. In particular, we report the empirical parameters that have been set up when the codes have been updated using $\sigma_d(\pi)$ in (4.3) for the definition of stiffness and computing $\kappa_d(\pi)$ using the algorithm *KappaStiffBvp* presented in the previous section.

In particular the monitor function is as defined by (5.1), but the parameter $p/(1-p) = 10^{-5}$ is used for Lobatto and $p/(1-p) = 0.08$ is used for TWPBVPC. The three parameters $\kappa_d(\pi)$ $\kappa_{1,d}(\pi)$ and $\gamma_{1,d}(\pi)$ are considered as having become stabilised using the same criterion as was defined in [8] that is if they change by less than 5 % from one mesh to another. A problem is considered stiff if

$$\sigma_d(\pi) > 10. \tag{5.2}$$

This criterion is different from the one presented in the previous version of the code TWPBVPLC, where a problem was considered stiff if $\kappa_1(\pi)/\gamma_1(\pi) > 100$. However the new definition of the stiffness parameter $\sigma_d(\pi)$ in (4.3) allows us to use the same criterion (5.2) for both the codes.

The algorithm for adding and removing points, for stiff problems, is based on the following two quantities associated with the monitor function ψ:

$$r_1 = \max_{i=1,\ldots,n} (\psi(x_i)h_i),$$

and

$$r_2 = \sum_{i=1}^{n} (\psi(x_i)h_i)/N.$$

Here h_i refers to the mesh spacing on the current mesh and n refers to the number of points in the mesh. We decided to add additional mesh points when $\psi(x_i)h_i$ is sufficiently large and in our Lobatto code we have taken this as being when it is greater than $\max(0.65r_1, r_2)$. For TWPBVPC we add additional mesh points when $\psi(x_i)h_i$ is greater than $max(0.5r_1, r_2)$. Note that these two parameters differ because usually the Lobatto schemes give us more reliable information concerning the regions where it is appropriate for points to be added. The number of mesh points to be added depends on the number of intervals in which this relation is satisfied. If the problem is stiff and the conditioning parameters are not stabilized the mesh control procedure puts points in the region of rapid variation of the monitor function and also makes sure that not too many points are added at any stage. If the conditioning parameters are stabilized, we are usually working with a good mesh and, if the error is small, we add points using the estimated local error. The technique for removing points also takes into account the monitor function and we remove points only when $\psi(x_i)h_i$ is less than $10^{-5}r_2$. This parameter has been changed for TWPBVPLC in order to make it more robust for non linear problems.

For nonlinear problems the strategy used in TWPBVPLC and in TWPBVPC remain the same that is, if the Newton iteration scheme does not converge, the partially converged solution is considered stiff if $\sigma_d(\pi) > 10$ in TWPBVPLC whereas in TWPBVPC 10 is replaced by 5.

5.2. Mesh selection for boundary value methods. The code TOM, based on boundary value methods of even order from 2 to 10, is different from the deferred correction codes in two main respects. The first is in the solution of non linear problems, where it uses a quasi linearization procedure that allows the use of the conditioning parameters for each linear problem arising during the quasi linearization. The second is the use of the monitor function that allows us to both move the mesh points and add and remove them. Nevertheless, we tried to keep most of the empirical parameters similar to the codes based on deferred corrections. In particular, the three parameters $\kappa_d(\pi)$ $\kappa_{1,d}(\pi)$ and $\gamma_{1,d}(\pi)$ are considered as having become stabilised using the same criterion as for the deferred correction codes, that is if they change by less than 5% from one mesh to another. A problem is considered stiff when $\sigma_d(\pi) > 100$ and this is different from the criterion (5.2) used in the deferred correction codes and is mainly due to the different numerical schemes used. We set $p/(1-p) = 0.08$ which is exactly the same as was used for TWPBVPC. We decided to add additional mesh points when $\psi(x_i)h_i$ is sufficiently large and in TOM we take this as being when it is greater than $\max(0.65r_1, r_2)$, which again is exactly the same as in the Lobatto code. In TOM we remove points, in the first quasilinearization step, when $\psi(x_i)h_i$ is less than $10^{-3}r_2$ or $10^{-8}r_2$, depending on the order of the method used and on the estimated error in the boundary points. For the subsequent quasilinearization steps we use $10^{-3}r_2$. We note that the quasilinearization algorithm implemented in TOM makes the behavior of this code different from the codes using a damped Newton technique. The reader is referred to [23] for more details concerning the solution of nonlinear problems.

5.3. Estimation of the error. We note that the hybrid mesh selection strategy asks for the conditioning parameters to be stabilized before we are able to use the error or the defect in the mesh selection. This is equivalent to asking that the fundamental matrix $Y(x)$ in (3.2) is well approximated by the numerical method before selecting the mesh. In this case the error estimate would be of interest and the size of the stability constant $\kappa_d(\pi)$ gives us information about the reliability of this estimate. In codes based on deferred correction the error for the order 4 method is estimated using the local truncation error, while for the order 6, and the order 8 methods, the error is estimated using the difference between the computed solution of order 4 and 6, or 6 and 8, respectively. Since the error is usually computed using the relative difference between the solution computed by methods of different orders, it is a good estimate of the global error. The same is true for the code TOM, where the error is estimated by computing an approximate solution of a higher order method using one step of a deferred correction procedure. However, the conditioning parameters give us more information about the error estimation. In fact, if the codes do not give a solution because the input error tolerances are too stringent, the user could change the input parameters accordingly. The codes only give a warning about this, because the input error tolerances are usually different for each component of the solution and cannot be changed automatically.

Additional important information related to the conditioning parameters is whether they are stabilized or not. In particular if $\kappa_d(\pi)$ is not stabilized the exit flag of the codes is -1 and not 0, because we can not assume that the numerical solution is reliable and we may be basing our strategy on non-converged solutions.

6. Numerical Results. In this section we present some numerical results to justify the changes made in the codes TWPBVPLC and TWPBVPC. We do not report on numerical experience with the code TOM

because the Fortran version is not yet available. Numerical results using TOM can be found in the following papers [23, 22, 25, 26]. In the following, to simplify the notation, we denote the discrete conditioning parameters by $\kappa(\pi), \kappa_2(\pi), \kappa_1(\pi), \gamma_1(\pi), \sigma(\pi)$. We have run the codes on a set of 33 singular perturbation boundary value problems (see [11] for a description of the test problems), the first set of numerical results is intended to show the reliability of the infinity norm estimator used in the codes. In Figure 6.1 we report, for each test problem, the mean value of the ratio $\kappa(\pi)/\kappa_H(\pi)$ where $\kappa_H(\pi)$ is the value computed using the Higham algorithm in [15]. The mean value has been computed running the problems for different values of ϵ and for different values of the input tolerances. A value smaller than unity means that the Higham algorithm is on average more accurate; a value higher than unity means that the new estimator is better. The number of computed matrices is 19195, having size ranging from 9 to 34976. The mean value over all the experiments for both codes is 1.139. This value means that the new estimator gives, on average, better results. In Figure 6.1 we give a diagram showing the mean value of the ratio $\kappa(\pi)/\kappa_H(\pi)$ for the 33 problems and for the two codes. We note that for problems 9 and 16 the new estimator computes a better lower bound. For the other problems the results are similar. However it is interesting to see that the minimum value of the mean ratio is 0.8 (computed when solving problem 27 with $\epsilon = 10^{-5}$ and $tol = 10^{-6}$ using TWPBVPC), the maximum value of the mean ratio is 84.3 (computed when solving problem 9 with $\epsilon = 10^{-6}$ and $tol = 10^{-4}$ using TWPBVPC).

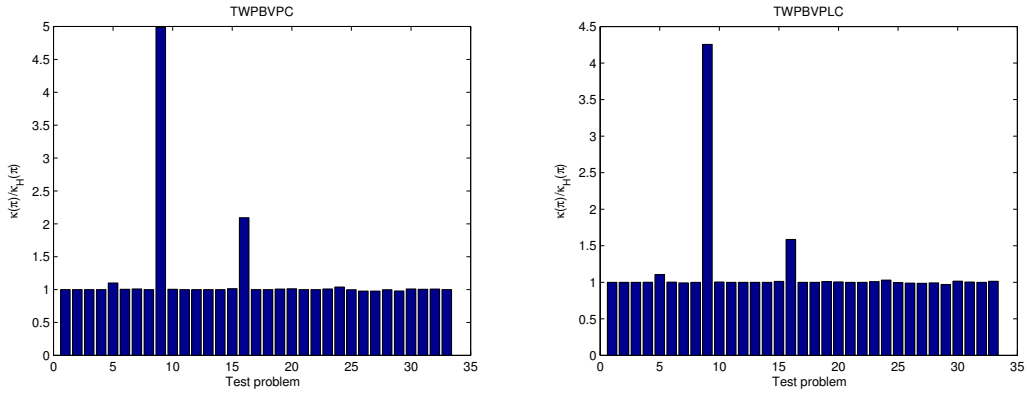

FIG. 6.1. *Mean value of the ratio $\kappa(\pi)/\kappa_h(\pi)$ for each test problem solved with different values of ϵ and different values of tol.*

In Figure 6.2 we report, for the problems 1 and 15, the ratio $\kappa_1(\pi)/\gamma_1(\pi)$ compared with the new value of $\sigma(\pi)$. We note that for these problems $\sigma(\pi)$ is a more reliable estimate of the stiffness of the problem. In general, however, the difference between the two stiffness estimators is minimal.

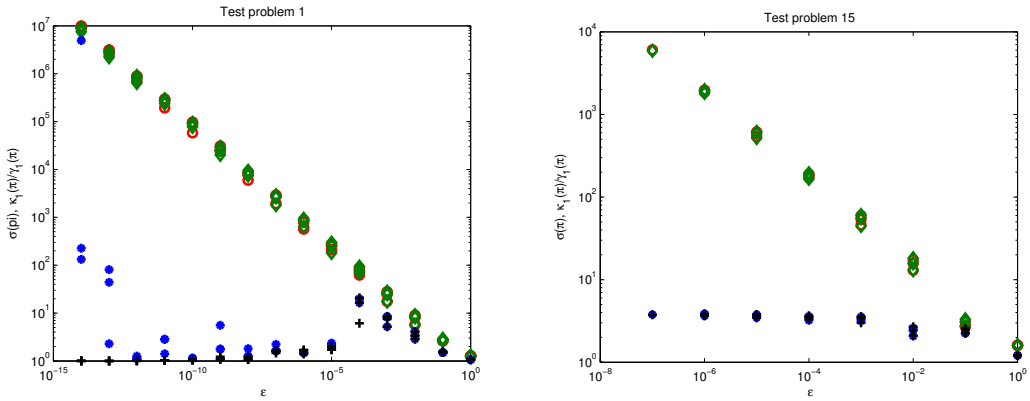

FIG. 6.2. *Circle (TWPBVPC) and diamond (TWPBVPLC) are the values of $\sigma(\pi)$ computed with different values of the input tolerances, asterisks (TWPBVPC) and plus (TWPBVPLC) are the corresponding values of $\kappa_1(\pi)/\gamma_1(\pi)$.*

The behavior of the codes using the conditioning parameters is more or less the same as for the previous versions. We refer to [8, 12], where many experiments are reported, for a comparison to the standard mesh

selection strategy with the one using conditioning. Nevertheless there are some improvements on certain difficult problems that are explained below.

We note that when $\kappa_2(\pi)$ is very small the standard mesh selection strategy and the one based on conditioning gives very similar results, and in many cases the standard mesh selection works better. This is explained by the fact that, since $\kappa_2(\pi)$ is small, the local truncation error is a good approximation of the global error. In the following we report only the results for three problems, the first two are problems for which $\kappa_2(\pi)$ is of the same size as $\kappa_1(\pi)$, the third problem is one that has 2, 1 or 0 solutions depending on the value of a certain parameter appearing in the differential equation. For all the problems we set $tol(ncomp_1) = tol(ncomp_2) = tol$ as input. The number of the problem is the same as reported in [11].

Problem 6

$$\epsilon y'' + xy' = -\epsilon\pi^2\cos(\pi x) - \pi x\sin(\pi x), \qquad y(-1) = -2,\ y(1) = 0.$$

This problem is a linear singularly perturbed problem. It has been chosen because, for $0 < \epsilon \ll 1$, the solution has a turning point at $x = 0$. The exact solution is $y(x) = \cos(\pi x) + \exp((x-1)/\sqrt{\epsilon}) + \exp(-(x+1)/\sqrt{\epsilon})$.

Problem 19

$$\epsilon y'' + \exp(y)y' - \frac{\pi}{2}\sin(\frac{\pi x}{2})\exp(2y) = 0, \quad y(0) = 0, y(1) = 0.$$

This problem has a boundary layer at $x = 0$. We use as initial guess 0 for y and y'.

Problem 34

$$y'' + \lambda exp(y) = 0, \quad y(0) = y(1) = 0.$$

This example is Bratu's problem, which is considered by Shampine and Muir [28]. Setting $\lambda^* = 3.51383\ldots$ it is known that if $0 \le \lambda < \lambda^*$ the problem has two solutions, for $\lambda = \lambda^*$ it has one solution and for $\lambda > \lambda^*$ there is no solution. We use as initial guess 0 for y and y'.

6.1. Description of the results. In Figure 6.3 we report, for problems 6 and 19, the maximum mesh used by the two codes with and without using the conditioning parameters in the mesh selection. For simplicity we add in brackets to the name of the code the term ON or OFF that indicates if the conditioning parameters have been used or not in the mesh selection (TWPBVPC(ON) means that the conditioning has been used) . The problems were solved for different values of ϵ and $tol = 10^{-4}, 10^{-6}, 10^{-8}$. In the diagrams, circles and diamonds are related to TWPBVPC(ON) and TWPBVPLC(ON) respectively, asterisks and plus are related to TWPBVPC(OFF) and TWPBVPLC(OFF) respectively (if the code was not able to find a solution the corresponding symbol is not reported). We note that for these problems the conditioning parameters usually allow us to obtain the solution with a smaller number of mesh points than when conditioning is not used. For both problems TWPBVPC is able to obtain the solution when the other algorithm fails; TWPBVPLC has no problem with the linear example, but for the nonlinear example the use of the conditioning allows us to compute the solution for smaller values of ϵ.

Figure 6.4 plots the condition numbers $\kappa(\pi)$ and $\kappa_1(\pi)$ for a range of values of ϵ, and Figure 6.5 plots the value of $\sigma(\pi)$ and the ratio $\kappa_1(\pi)/\gamma_1(\pi)$. For these problems we note that $\kappa_1(\pi)/\gamma_1(\pi) \approx \sigma(\pi)$, and the condition number $\kappa(\pi) \approx 2\kappa_1(\pi)$ grows like $\sqrt{(}1/\epsilon)$ for problem 6 and like $1/\epsilon$ for problem 19. The stiffness ratio $\sigma(\pi)$ has the same behavior (see 6.5).

The third example is taken from [28] where the authors solved the problem by using the MATLAB code BVP4C with $\lambda = 3.45$ for which there are two solutions. For this problem a solution was computed in a perfectly satisfactory way and the condition number was estimated to be 3400. The same procedure was carried out again with $\lambda = 3.55$ (for which there is no solution) and the MATLAB code produced what looked like a perfectly satisfactory solution with a conditioning constant estimated to be 10^6. This very big conditioning constant warns that the 'solution' may have no correct digits and this is indeed the case. Further experiments are reported in [28] where it is emphasised that we need to look at the conditioning of a problem before accepting a solution. The approach taken by Shampine and Muir is to warn the user of a very large condition number if it exists. We note that the condition number computed in [28] is the infinity norm of a scaled matrix W_1GW_2 where the diagonal scaling matrices are related to the scaling factors used in the code to compute the residual. We compute, instead, a condition number related to the problem and some important information computed by the code is whether the estimates of the conditioning parameters stabilised or not. In Table 6.1 we report

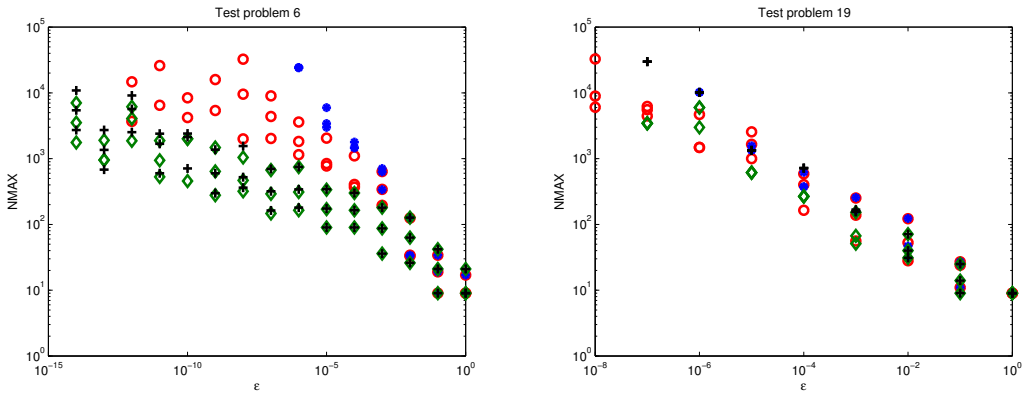

FIG. 6.3. *Circle (TWPBVPC(ON)), asterisks (TWPBVPC(OFF)), diamond (TWPBVPLC(ON)) and plus (TWPBVPLC(OFF))) are the values of maximum mesh used by the codes with different values of tol and of* ϵ.

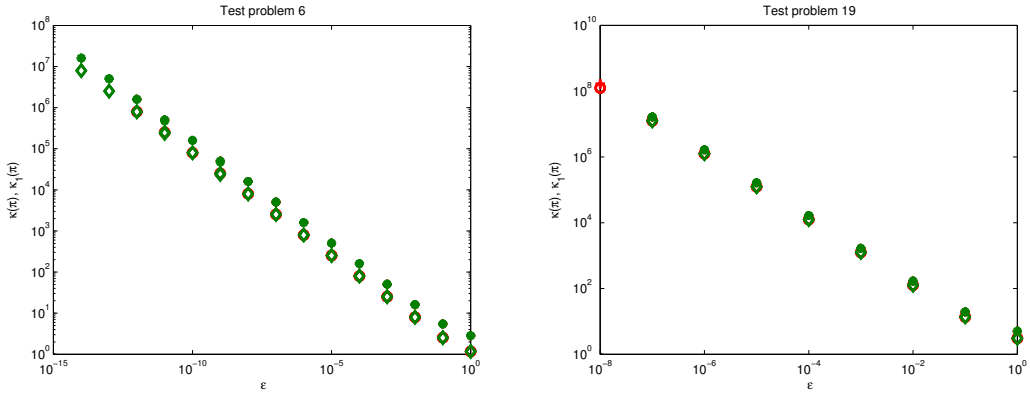

FIG. 6.4. *Circle (TWPBVPC) and diamond (TWPBVPLC) are the values of* $\kappa_1(\pi)$, *asterisks (TWPBVPC) and plus (TWPBVPLC) are the values of* $\kappa(\pi)$.

the results obtained by the codes TWPBVPC and TWPBVPLC solving the problem with different values of λ and $tol = 10^{-3}$, $tol = 10^{-6}$ and $tol = 10^{-9}$, using an initial mesh with $N_S + 1$ mesh points. To be sure that the conditioning parameters obtained are stabilized we checked the value by running the code using a mesh doubled with respect to the final one and where we do not perform any grid refinement (the final mesh has $N_F + 1$ points). In Table 6.1 we set the value of $STAB_C$ equal to T if for the doubled mesh the value of $\kappa(\pi)$ differs by less than 5 % from the value computed with the mesh $N_F + 1$, $STAB_C$ is set to F otherwise.

Only for $\lambda = 3.5, 3.51, 3.513$ do the codes give an output flag with a value of 0 and $STAB_C$= T, for all the other values we have that the flag is -1. But in no cases do we accept a computed 'solution' when none exist. If we start the code with a starting mesh with a larger value of N_S, we see that for both the codes the value of $STAB_C$ is T when $\lambda <= 3.51383$. For $\lambda > 3.51383$, however, the number of mesh points in the final mesh can be very high compared with the one used for $\lambda \leq 3.51383$. In particular TWPBVPC fails to give a solution for $\lambda \geq 3.513832$, TWPBVPLC fails for $\lambda \geq 3.513833$, and for the solution given with $\lambda = 3.513832$ the $STAB_C$ flag is always F.

7. Conclusion. In this paper we have been concerned with the numerical solution of singularly perturbed boundary value problems using variable grid integration methods. We have established a fundamental approach to choosing our meshes, defining sequences of meshes so that the continuous and discrete problems have the same conditioning. We do this by developing a monitor function which depends both on local accuracy and conditioning. We have presented new techniques to compute the conditioning parameters and we have emphasised that the approach adopted is very similar for all three codes: TOM, TWPBVPC and TWPBVPLC. The main difference is in how we choose the different values of a few heuristic parameters and we have explained in

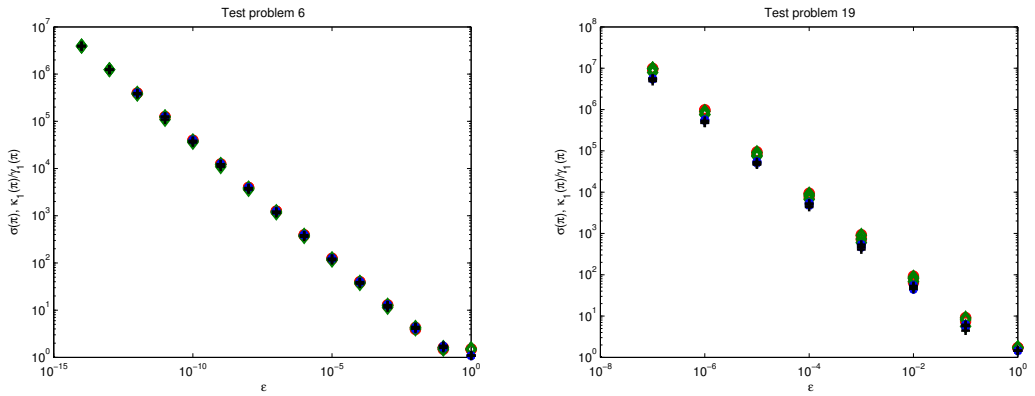

FIG. 6.5. *Circle (TWPBVPC) and diamond (TWPBVPLC) are the values of $\sigma(\pi)$, asterisks (TWPBVPC) and plus (TWPBVPLC) are the corresponding values of $\kappa_1(\pi)/\gamma_1(\pi)$.*

some detail how this is done. Our codes can be considerably more efficient than other codes, due to the fact that we pay attention to the conditioning of the problem, and we have never accepted a 'solution' when none exists. However, for some problems, our conditioning parameters may converge rather slowly and we intend to examine this in detail in a future paper.

REFERENCES

[1] U. M. Ascher, R. M. M. Mattheij and R. D. Russell. *Numerical solution of boundary value problems for ordinary differential equations*, Classics in Applied Mathematics **13**. Society for Industrial and Applied Mathematics (SIAM), Philadelphia, PA, 1995. Corrected reprint of the 1988 original.

[2] Z. Bashir-Ali, J. R. Cash and H. H. M. Silva. Lobatto deferred correction for stiff two-point boundary value problems. *Comput. Math. Appl.*, **36**, no. 10-12, 59–69, 1998. Advances in difference equations, II.

[3] L. Brugnano and D. Trigiante. Tridiagonal matrices: invertibility and conditioning. *Linear Algebra Appl.*, **166**, 131–150, 1992.

[4] L. Brugnano and D. Trigiante. On the characterization of stiffness for ODEs. *Dynam. Contin. Discrete Impuls. Systems*, **2**, no. 3, 317–335, 1996.

[5] L. Brugnano and D. Trigiante. A new mesh selection strategy for ODEs. *Appl. Numer. Math.*, **24**, no. 1, 1–21, 1997.

[6] L. Brugnano and D. Trigiante. *Solving Differential Problems by Multistep Initial and Boundary Value Methods*. Gordon & Breach,Amsterdam, 1998.

[7] J. R. Cash. On the numerical integration of nonlinear two-point boundary value problems using iterated deferred corrections. II. The development and analysis of highly stable deferred correction formulae. *SIAM J. Numer. Anal.*, **25**, no. 4, 862–882, 1988.

[8] J. R. Cash and F. Mazzia. A new mesh selection algorithm, based on conditioning, for two-point boundary value codes. *J. Comput. Appl. Math.*, **184**, no. 2, 362–381, 2005.

[9] J. R. Cash, F. Mazzia, N. Sumarti and D. Trigiante. The role of conditioning in mesh selection algorithms for first order systems of linear two point boundary value problems. *J. Comput. Appl. Math.*, **185**, no. 2, 212–224, 2006.

[10] J. R. Cash, G. Moore and R. Wright. An automatic continuation strategy for the solution of singularly perturbed nonlinear boundary value problems. *ACM Trans. Math. Software*, **27**, no. 2, 245–266, 2001.

[11] J. R. Cash and F. Mazzia. Algorithms for the solution of two-point boundary value problems. `http://www.ma.ic.ac.uk/~jcash/BVP_software/twpbvp.php`.

[12] J. R. Cash and F. Mazzia. Hybrid mesh selection algorithms based on conditioning for two-point boundary value problems. *JNAIAM J. Numer. Anal. Ind. Appl. Math.*, **1**, no. 1, 81–90, 2006.

[13] C. de Boor and H.-O. Kreiss. On the condition of the linear systems associated with discretized BVPs of ODEs. *SIAM J. Numer. Anal.*, **23**, no. 5, 936–939, 1986.

[14] W. W. Hager. Condition estimates. *SIAM J. Sci. Statist. Comput.*, **5**, no. 2, 311–316, 1984.

[15] N. J. Higham. FORTRAN codes for estimating the one-norm of a real or complex matrix, with applications to condition estimation. *ACM Trans. Math. Software*, **14**, no. 4, 381–396 (1989), 1988.

[16] N. J. Higham. *Accuracy and stability of numerical algorithms*. Society for Industrial and Applied Mathematics (SIAM), Philadelphia, PA, 1996.

[17] N. J. Higham and F. Tisseur. A block algorithm for matrix 1-norm estimation, with an application to 1-norm pseudospectra. *SIAM J. Matrix Anal. Appl.*, **21**, no. 4, 1185–1201 (electronic), 2000.

[18] F. Iavernaro, F. Mazzia and D. Trigiante. Stability and conditioning in numerical analysis. *JNAIAM J. Numer. Anal. Ind. Appl. Math.*, **1**, no. 1, 91–112, 2006.

[19] P. Kunkel and V. Mehrmann. *Differential-algebraic equations,Analysis and numerical solution*. EMS Textbooks in Mathematics. European Mathematical Society (EMS), Zürich, 2006.

TABLE 6.1

Conditioning parameters, flags, number of mesh points in the initial constant mesh and number of mesh points in the final mesh for Problem 34

				TWPBVPC(ON)					
λ	tol	$\kappa(\pi)$	$\kappa_1(\pi)$	$\gamma_1(\pi)$	$\sigma(\pi)$	FL	N_S+1	$STAB_C$	N+1
3.5	10^{-3}	0.534E+02	0.366E+02	0.289E+02	0.130E+01	0	10	T	10
	10^{-6}	0.537E+02	0.368E+02	0.277E+02	0.136E+01	0	10	T	18
	10^{-9}	0.538E+02	0.368E+02	0.275E+02	0.137E+01	0	10	T	21
3.51	10^{-3}	0.102E+03	0.701E+02	0.556E+02	0.128E+01	0	10	T	10
	10^{-6}	0.104E+03	0.712E+02	0.539E+02	0.134E+01	0	10	T	17
	10^{-9}	0.104E+03	0.712E+02	0.535E+02	0.135E+01	0	10	T	21
3.513	10^{-3}	0.209E+03	0.144E+03	0.114E+03	0.126E+01	0	10	F	10
	10^{-6}	0.224E+03	0.154E+03	0.117E+03	0.132E+01	-1	10	T	19
	10^{-9}	0.224E+03	0.154E+03	0.115E+03	0.134E+01	-1	10	T	24
3.5138	10^{-3}	0.105E+04	0.720E+03	0.549E+03	0.131E+01	-1	10	F	19
	10^{-6}	0.105E+04	0.720E+03	0.549E+03	0.131E+01	-1	10	F	19
	10^{-9}	0.105E+04	0.720E+03	0.549E+03	0.131E+01	-1	10	F	19
	10^{-9}	0.117E+04	0.801E+03	0.593E+03	0.135E+01	0	40	T	40
3.51383	10^{-3}	0.214E+04	0.147E+04	0.112E+04	0.131E+01	-1	10	F	19
	10^{-6}	0.537E+04	0.369E+04	0.274E+04	0.135E+01	-1	10	F	37
	10^{-9}	0.537E+04	0.369E+04	0.274E+04	0.135E+01	-1	10	F	37
	10^{-9}	0.721E+04	0.495E+04	0.363E+04	0.136E+01	0	80	T	80
3.513831	10^{-3}	0.225E+04	0.155E+04	0.118E+04	0.131E+01	-1	10	F	19
	10^{-6}	0.120E+05	0.823E+04	0.612E+04	0.135E+01	-1	10	F	37
	10^{-9}	0.120E+05	0.823E+04	0.612E+04	0.135E+01	-1	10	F	37
	10^{-9}	0.773E+04	0.531E+04	0.385E+04	0.138E+01	-1	80	T	10113
3.513832	10^{-3}	*							
	10^{-6}	*							
	10^{-9}	*							
				TWPBVPLC(ON)					
3.5	10^{-3}	0.534D+02	0.366D+02	0.289D+02	0.130D+01	0	10	T	10
	10^{-6}	0.534D+02	0.366D+02	0.289D+02	0.130D+01	0	10	T	10
	10^{-9}	0.538D+02	0.368D+02	0.276D+02	0.137D+01	0	10	T	21
3.51	10^{-3}	0.102D+03	0.701D+02	0.556D+02	0.128D+01	0	10	T	10
	10^{-6}	0.102D+03	0.701D+02	0.556D+02	0.128D+01	0	10	T	10
	10^{-9}	0.104D+03	0.712D+02	0.535D+02	0.135D+01	0	10	T	22
3.513	10^{-3}	0.209D+03	0.144D+03	0.114D+03	0.126D+01	0	10	F	10
	10^{-6}	0.209D+03	0.144D+03	0.114D+03	0.126D+01	0	10	F	10
	10^{-9}	0.154D+03	0.116D+03	0.224D+03	0.133D+01	-1	10	T	21
3.5138	10^{-3}	0.102D+04	0.700D+03	0.533D+03	0.131D+01	-1	10	F	19
	10^{-6}	0.102D+04	0.700D+03	0.533D+03	0.131D+01	-1	10	F	19
	10^{-9}	0.102D+04	0.700D+03	0.533D+03	0.131D+01	-1	10	F	19
	10^{-9}	0.117E+04	0.801E+03	0.593E+03	0.135E+01	0	40	T	40
3.51383	10^{-3}	0.214D+04	0.147D+04	0.112D+04	0.131D+01	-1	10	F	19
	10^{-6}	0.537D+04	0.369D+04	0.274D+04	0.135D+01	-1	10	T	37
	10^{-9}	0.537D+04	0.369D+04	0.274D+04	0.135D+01	-1	10	T	37
3.513831	10^{-3}	0.225D+04	0.155D+04	0.118D+04	0.131D+01	-1	10	F	19
	10^{-6}	0.354D+04	0.244D+04	0.176D+04	0.138D+01	-1	10	T	9217
	10^{-9}	0.354D+04	0.244D+04	0.176D+04	0.138D+01	-1	10	T	9217
3.513832	10^{-3}	0.238D+04	0.164D+04	0.125D+04	0.131D+01	-1	10	F	19
	10^{-6}	0.146D+05	0.101D+05	0.729D+04	0.138D+01	-1	10	F	9217
	10^{-9}	0.146D+05	0.101D+05	0.729D+04	0.138D+01	-1	10	F	9217
	10^{-9}	0.146D+05	0.101D+05	0.729D+04	0.138D+01	-1	20	F	9729
	10^{-9}	*	*	*	*	-1	40	*	*
3.513833	10^{-3}	*							
	10^{-6}	*							
	10^{-9}	*							

[20] R. M. M. Mattheij and G. W. M. Staarink. An efficient algorithm for solving general linear two-point BVP. *SIAM J. Sci. Statist. Comput.*, **5**, no. 4, 745–763, 1984.

[21] R. M. M. Mattheij and G. W. M. Staarink. On optimal shooting intervals. *Math. Comp.*, **42**, no. 165, 25–40, 1984.

[22] F. Mazzia, A. Sestini and D. Trigiante. The continous extension of the B-spline linear multistep metods for BVPs on non-uniform meshes. *Appl. Numer. Math.*, **59**, no. 3-4, 723–738, 2009.

[23] F. Mazzia and D. Trigiante. Efficient strategies for solving nonlinear problems in bvps codes. *Nonlinear Studies*. In press.

[24] F. Mazzia and D. Trigiante. Numerical solution of singular perturbation problems. *Calcolo*, **30**, no. 4, 355–369, 1993.

[25] F. Mazzia, A. Sestini and D. Trigiante. B-spline linear multistep methods and their continuous extensions. *SIAM J. Numer. Anal.*, **44**, no. 5, 1954–1973 (electronic), 2006.
[26] F. Mazzia, A. Sestini and D. Trigiante. BS linear multistep methods on non-uniform meshes. *JNAIAM J. Numer. Anal. Ind. Appl. Math.*, **1**, no. 1, 131–144, 2006.
[27] F. Mazzia and D. Trigiante. A hybrid mesh selection strategy based on conditioning for boundary value ODE problems. *Numer. Algorithms*, **36**, no. 2, 169–187, 2004.
[28] L. F. Shampine and P. H. Muir. Estimating conditioning of BVPs for ODEs. *Math. Comput. Modelling*, **40**, no. 11-12, 1309–1321, 2004.

Edited by: Pierluigi Amodio and Luigi Brugnano
Received: April 9, 2009
Accepted: October 7, 2009