

## NONLOCAL AUTOMATED SENSITIVITY ANALYSIS

R. KALABA<sup>1,2</sup> and L. TEFATSION<sup>3†</sup>

Departments of <sup>1</sup>Electrical Engineering, <sup>2</sup>Biomedical Engineering and <sup>3</sup>Economics (Modelling Research Group MC-0152), University of Southern California, Los Angeles, CA 90089, U.S.A.

(Received 28 July 1989; in revised form 17 August 1989)

**Abstract**—Previously a complete ordinary differential equation (ODE) system was developed for tracking the solution  $\mathbf{x}(\alpha)$  of a parameterized system of nonlinear equations  $\mathbf{0} = \Psi(\mathbf{x}, \alpha)$  over an  $\alpha$ -interval  $[\alpha^0, \alpha^1]$ . This paper develops a two-phase complex homotopy continuation method for obtaining the required initial conditions at  $\alpha^0$ . An initial "short" artificial continuation is followed by a continuation which essentially proceeds through the physically meaningful function  $\Psi$ , which can ameliorate the problem of artificially induced singularities. Also, the path in the complex plane followed by the continuation parameter in each phase evolves sequentially in an attempt to keep the path both short (minimal number of integration steps) and numerically stable (avoidance of singular points). A FORTRAN program *Nasa* has been developed for solving the complete ODE system, starting with a two-phase initialization. The program incorporates a fast and efficient procedure for the automatic evaluation of partial derivatives. Numerical experiments are reported which illustrate the program's effectiveness.

### 1. INTRODUCTION

In Ref. [1] an analytically complete system of ordinary differential equations (ODEs) was developed for finding a solution vector  $\mathbf{x}(\alpha)$  for a parameterized system of nonlinear equations  $\mathbf{0} = \Psi(\mathbf{x}, \alpha)$ . The well-known ODE  $d\mathbf{x}(\alpha)/d\alpha = -(\Psi_{\mathbf{x}})^{-1}\Psi_{\alpha}$  was supplemented with ODEs for the adjoint  $A(\alpha)$  and the determinant  $\delta(\alpha)$  of the Jacobian matrix  $\Psi_{\mathbf{x}}(\mathbf{x}(\alpha), \alpha)$ . This differential system, in principle, permits *nonlocal sensitivity analysis*, i.e. the tracking of the solution vector  $\mathbf{x}(\alpha)$  and the sensitivity vector  $d\mathbf{x}(\alpha)/d\alpha$ , together with the adjoint  $A(\alpha)$  and the determinant  $\delta(\alpha)$ , over any closed  $\alpha$ -interval  $[\alpha^0, \alpha^1]$  where the determinant  $\delta(\alpha)$  remains nonzero.

A FORTRAN program was used successfully in Ref. [1] to solve this differential system for several illustrative examples. Subsequently, the program was modified to incorporate a fast and efficient algorithm *Feed* for the automatic evaluation of higher-order partial derivatives [2]. Nevertheless, to make the program a truly practical tool, a simple and reliable procedure was still needed for obtaining the initial conditions,  $\mathbf{x}(\alpha^0)$ ,  $A(\alpha^0)$  and  $\delta(\alpha^0)$ , required by the differential system at the initial parameter point  $\alpha^0$ .

This paper develops a simple but effective homotopy continuation method for obtaining these initial conditions which differs in two principal respects from standard homotopy continuation methods.‡

First, the proposed continuation essentially proceeds through the function  $\Psi$  rather than through an artificial construct. Letting  $F(\mathbf{x}) \equiv \Psi(\mathbf{x}, \alpha^0)$ , a standard continuation method for finding a solution vector  $\mathbf{x}^0 = \mathbf{x}(\alpha^0)$  to the system of equations  $\mathbf{0} = F(\mathbf{x})$  would generally involve solving a system of equations of the form

$$\mathbf{0} = \mu \cdot F(\mathbf{x}) + [1 - \mu] \cdot [\mathbf{x} - \mathbf{c}], \quad (1)$$

for  $\mathbf{x}$  as a function of  $\mu$  as  $\mu$  varies from 0 to 1. In contrast, the basic continuation developed in this paper solves the system of equations

$$\mathbf{0} = [F(\mathbf{x}) - F(\mathbf{c})] + \beta \cdot F(\mathbf{c}), \quad (2)$$

for  $\mathbf{x}$  as a function of  $\beta$  as  $\beta$  varies from 0 to 1.

†To whom all correspondence should be addressed.

‡Although homotopy continuation methods have a long history in mathematics, it is only in recent years, with the advent of high-speed computers, that they have become a practical tool for solving systems of nonlinear equations. Homotopy continuation methods involve the imbedding of a function  $F(\mathbf{x})$  into a parameterized family of functions  $F(\mathbf{x}, \mu)$  such that  $F(\mathbf{x}, 0)$  has a simple known structure, whereas  $F(\mathbf{x}, 1)$  coincides with  $F(\mathbf{x})$ . Given certain regularity conditions, a solution for  $\mathbf{0} = F(\mathbf{x})$  can in principle be found by following the curve of solutions  $\mathbf{x}(\mu)$  to  $\mathbf{0} = F(\mathbf{x}, \mu)$ , as  $\mu$  varies from 0 to 1. See Ref. [3] for a survey of some of this work. For brevity, the qualifier "homotopy" will hereafter be suppressed.

As is clarified below, a potential advantage of the  $\beta$ -continuation (2) over the more standard  $\mu$ -continuation (1) is that  $\mathbf{x}$  only appears in (2) as an argument of  $F$ . Consequently, the Jacobian matrix for the r.h.s. of equation (2) coincides with the Jacobian matrix  $F_x(\mathbf{x})$  for the original system function  $F$ , which can ameliorate the problem of artificially induced singularities. The complete differential system developed in Ref. [1] is used to solve equation (2) over the  $\beta$ -path from 0 to 1. In principle, the solution at  $\beta = 1$  yields the required initial conditions  $\mathbf{x}(\alpha^0)$ ,  $A(\alpha^0)$  and  $\delta(\alpha^0)$ .

The initial conditions needed to solve equation (2) at  $\beta = 0$  are the solution vector  $\mathbf{x} = \mathbf{c}$ , together with evaluations for the adjoint and determinant of the Jacobian matrix  $F_x(\mathbf{c})$ . The vector  $\mathbf{c}$  in equation (2) might be chosen to be a point where these evaluations are easily determined. Alternatively, these evaluations can be obtained by means of an initial  $\theta$ -continuation of the form

$$\mathbf{0} = \theta \cdot [F(\mathbf{x}) - F(\mathbf{c})] + [1 - \theta] \cdot J \cdot [\mathbf{x} - \mathbf{c}], \quad (3)$$

for  $\theta$  varying from 0 to 1, also solvable by the complete differential system developed in Ref. [1]. In equation (3),  $J$  is an initial guess for the Jacobian matrix  $F_x(\mathbf{c})$ ; for example,  $J$  could be the identity matrix  $I$ . Note that, in contrast to equation (1), the solution vector  $\mathbf{x}(\theta)$  for the  $\theta$ -continuation (3) stays constant at  $\mathbf{c}$ ; and the components of the Jacobian matrix for equation (3) exhibit constant rates of change with respect to  $\theta$  which are smaller in magnitude the better is the initial guess  $J$ .

A second departure from more standard continuation methods is that the continuation parameters  $\beta$  and  $\theta$  in equations (2) and (3) are to follow *sequentially* determined paths in the complex plane. Briefly, each parameter moves in the complex plane from  $0 + 0i$  to  $1 + 0i$  through a spider-web grid centered at  $1 + 0i$ . The path through the grid is determined step by step in an attempt to satisfy two potentially conflicting criteria: (1) keep the path short (minimize the number of integration steps); and (2) keep the path numerically stable (avoid singular points).

The complete differential system for solving a system of parameterized equations over a parameter interval is reviewed in Section 2. The incorporation of automatic derivative evaluation is discussed in Section 3. Section 4 presents the two-phase complex continuation method for obtaining all required initial conditions. Illustrative simulation experiments are presented in Section 5, using a FORTRAN program *Nasa* (Nonlocal automated sensitivity analysis) which incorporates both automatic differentiation and the two-phase initialization procedure. Concluding comments are given in Section 6. The logical progression of the *Nasa* program is discussed in the Appendix.

## 2. NONLOCAL SOLUTION OF PARAMETERIZED EQUATIONS

Sensitivity studies in many fields typically reduce to determining the response of a vector  $\mathbf{x}^0 = (x_1^0, \dots, x_n^0)$  to changes in a scalar  $\alpha^0$ , where  $\mathbf{x}^0$  and  $\alpha^0$  are required to satisfy an  $n$ -dimensional system of nonlinear equations of the form

$$\mathbf{0} = \Psi(\mathbf{x}, \alpha) \equiv \begin{bmatrix} \Psi^1(\mathbf{x}, \alpha) \\ \vdots \\ \Psi^n(\mathbf{x}, \alpha) \end{bmatrix}. \quad (4)$$

Given standard regularity conditions, the implicit function theorem guarantees the existence of a continuously differentiable function  $\mathbf{x}(\alpha)$  taking some open neighborhood  $N(\alpha^0)$  of  $\alpha^0$  into  $R^n$  such that

$$\mathbf{0} = \Psi(\mathbf{x}(\alpha), \alpha), \quad \alpha \in N(\alpha^0). \quad (5)$$

From equation (5) one obtains the fundamental equation of sensitivity analysis:

$$d\mathbf{x}(\alpha)/d\alpha = -\Psi_x(\mathbf{x}(\alpha), \alpha)^{-1} \Psi_\alpha(\mathbf{x}(\alpha), \alpha), \quad \alpha \in N(\alpha^0). \quad (6)$$

As it stands, equation (6) is an analytically incomplete system of ODEs. That is, a closed-form representation for the Jacobian inverse  $J(\alpha)^{-1} \equiv \Psi_x(\mathbf{x}(\alpha), \alpha)^{-1}$  as a function of  $\alpha$  is often not obtainable for  $n \geq 3$ . Thus, the integration of equation (6) from initial conditions would typically require the supplementary algebraic determination of the Jacobian inverse  $J(\alpha)^{-1}$  at each step in the integration process. Perhaps for this reason, many analysts interpret system (6) as a purely algebraic relation of the form  $\mathbf{z} = -A^{-1}\mathbf{b}$  which can be used, qualitatively, to sign or partially sign the components of  $d\mathbf{x}(\alpha)/d\alpha$  at a particular point  $\alpha^0$ . Numerical integration using specific functional forms is generally not attempted.

In Ref. [1] the differential system (6) is extended by the incorporation of ODEs for the Jacobian inverse. More precisely, letting  $A(\alpha)$  and  $\delta(\alpha)$  denote the adjoint and the determinant of the Jacobian matrix  $J(\alpha)$ , and recalling that the inverse of any nonsingular matrix can be represented as the ratio of its adjoint to its determinant, the following differential system is validated for  $\mathbf{x}(\alpha)$ ,  $A(\alpha)$  and  $\delta(\alpha)$ :

$$d\mathbf{x}(\alpha)/d\alpha = -A(\alpha)\Psi_x(\mathbf{x}(\alpha), \alpha)/\delta(\alpha), \quad (7a)$$

$$dA(\alpha)/d\alpha = [A(\alpha) \text{trace}(A(\alpha)B(\alpha)) - A(\alpha)B(\alpha)A(\alpha)]/\delta(\alpha) \quad (7b)$$

and

$$d\delta(\alpha)/d\alpha = \text{trace}(A(\alpha)B(\alpha)), \quad (7c)$$

where

$$B(\alpha) \equiv dJ(\alpha)/d\alpha \quad (7d)$$

is expressible as a known function of  $\mathbf{x}(\alpha)$ ,  $A(\alpha)$ ,  $\delta(\alpha)$  and  $\alpha$ . Initial conditions for equations (7a-c) must be provided at a parameter point  $\alpha^0$  by specifying values for  $\mathbf{x}(\alpha^0)$ ,  $A(\alpha^0)$  and  $\delta(\alpha^0)$  satisfying

$$\Psi(\mathbf{x}(\alpha^0), \alpha^0) = \mathbf{0}, \quad (7e)$$

$$A(\alpha^0) = \text{adj}(J(\alpha^0)) \quad (7f)$$

and

$$\delta(\alpha^0) = \det(J(\alpha^0)) \neq 0. \quad (7g)$$

The system of equations (7) provides an analytically complete system of ODEs for tracking the solution vector  $\mathbf{x}(\alpha)$  and the sensitivity vector  $d\mathbf{x}(\alpha)/d\alpha$ , together with the adjoint  $A(\alpha)$  and determinant  $\delta(\alpha)$  of the Jacobian matrix  $J(\alpha)$ , over any  $\alpha$ -interval  $[\alpha^0, \alpha^1]$  where the determinant remains nonzero. The feasibility of carrying out nonlocal sensitivity analysis is thus enhanced.

In Ref. [1] the differential system (7) is solved for various illustrative examples using a FORTRAN program incorporating a fourth-order Adams–Moulton integration method with a Runge–Kutta start. High numerical accuracy is obtained, even near critical points  $\alpha$  where the determinant  $\delta(\alpha)$  becomes zero. Two difficulties nevertheless arose concerning the routine application of this program in more realistic problem contexts.

First, it quickly became apparent that recourse had to be made to an automatic method of evaluating needed partial derivatives if a practical numerical tool was to be obtained. The components of the matrix  $B(\alpha) \equiv dJ(\alpha)/d\alpha$  appearing in equations (7b,c), involve the second-order partial derivatives of  $\Psi$ , and  $\Psi$  in turn could be defined in terms of the partial derivatives of some still more basic function (e.g. a criterion function for an optimization problem). Second, a reliable procedure was needed for generating the required initial conditions (7e–g) at the initial parameter point  $\alpha^0$ . Ideally, this procedure should be analytical rather than algebraic—in conformity with the guiding principle underlying the development of system (7)—and “nonlocal” in the sense that a good initial guess  $\mathbf{c}$  for  $\mathbf{x}(\alpha^0)$  is not an essential requirement.

A FORTRAN program *Nasa* for solving the differential system (7) has now been developed which incorporates a fast and efficient algorithm *Feed* for automatic differentiation and a two-phase complex continuation method for obtaining all required initial conditions. These two program features are discussed in turn in the following two sections.

### 3. INCORPORATION OF AUTOMATIC DIFFERENTIATION

In considering how best to incorporate automatic differentiation, use was first made of a method developed by Wengert [4] for sequentially evaluating higher-order partial derivatives.† Wengert's key idea was to decompose the evaluation of complicated functions of many variables into a sequence of simpler evaluations of special functions of one or two variables, referred to below as a "Wengert list". Total differentials of the special functions could be automatically evaluated along with the special function values. Partial derivatives could then be recovered from the total differentials by solving certain associated sets of linear algebraic equations.

Although programs were successfully written for implementing Wengert's method for first- and second-order partial derivatives, two important problems were noted. First, Wengert's method requires the repeated evaluation of certain identical functional forms as each individual partial derivative is separately recovered from a total differential, resulting in significant computational inefficiency. Second, Wengert's method requires the formation and solution of a distinct set of linear algebraic equations for each successive higher-order partial differentiation, and it does not seem possible to provide a systematic rule for how this is to be done.

An algorithm *Feed* (Fast efficient evaluation of derivatives) was then developed for the systematic exact evaluation of higher-order partial derivatives that overcomes both of these problems [2]. *Feed* retains Wengert's key idea of sequential function evaluation, but total differentials and linear algebraic equations play no role. An additional advantage of *Feed* is that memory and arithmetic requirements can be determined prior to any calculations.

As a simple illustration of *Feed*, consider the function  $F: R^2_{++} \rightarrow R$ , defined by

$$z = F(x, y) \equiv x + \log(xy). \quad (8)$$

Suppose one wishes to evaluate the function value  $z$ , the first-order partial derivatives  $z_x$  and  $z_y$ , and the second-order partial derivative  $z_{xx}$  at a given domain point  $(x, y)$ . Consider Table 1. The first column of Table 1 constitutes the Wengert list for  $F$ ; it sequentially evaluates the function value  $z = x + \log(xy)$  at the given domain point  $(x, y)$ . The second, third and fourth entries in each row give the indicated derivative evaluations of the first entry in the row, using only algebraic operations. The first two rows initialize the algorithm, one row being required for each function variable. The only input required for the first two rows is the domain point  $(x, y)$ . Each subsequent row outputs a one-dimensional array of the form  $(p, p_x, p_y, p_{xx})$ , using the arrays obtained from previous row calculations as inputs. The final row yields the desired evaluations  $(z, z_x, z_y, z_{xx})$ . These evaluations are exact up to round-off error.

As explained in Ref. [2], the rows in Table 1 can be numerically implemented by means of FORTRAN calculus subroutines called in order by a subroutine FUN, as is indicated schematically below:

```

SUBROUTINE FUN(X,Y,Z)
DIMENSION A(4),B(4),C(4),D(4),Z(4)
CALL VEC(1,X,A)
CALL VEC(2,Y,B)
CALL MULT(A,B,C)
CALL LOGG(C,D)
CALL ADD(A,D,Z)
RETURN
END

```

(9)

Given any domain point  $(X, Y)$ , FUN obtains the value and partial derivatives of the function  $z = x + \log(xy)$  at  $(X, Y)$  and returns these evaluations in the array Z.

Table 1. An illustrative application of the *Feed* algorithm

| Wengert list  | $\partial/\partial x$ | $\partial/\partial y$ | $\partial^2/\partial x^2$                |
|---------------|-----------------------|-----------------------|--|
| $a = x$       | $a_x = 1$             | $a_y = 0$             | $a_{xx} = 0$                             |
| $b = y$       | $b_x = 0$             | $b_y = 1$             | $b_{xx} = 0$                             |
| $c = ab$      | $c_x = a_x b + ab_x$  | $c_y = a_y b + ab_y$  | $c_{xx} = a_{xx} b + 2a_x b_x + ab_{xx}$ |
| $d = \log(c)$ | $d_x = c^{-1} c_x$    | $d_y = c^{-1} c_y$    | $d_{xx} = -c^{-2} c_x^2 + c^{-1} c_{xx}$ |
| $z = a + d$   | $z_x = a_x + d_x$     | $z_y = a_y + d_y$     | $z_{xx} = a_{xx} + d_{xx}$               |

†A similar method has been developed by Rall [5].

In principle, the *Feed* algorithm can be used to evaluate the value and partial derivatives through order  $k$  of any real-valued multivariable function which can be sequentially evaluated in a finite number of steps by means of the two-variable functions

$$w = u + v, \quad w = u - v, \quad w = uv, \quad w = u/v \quad \text{and} \quad w = u^v \quad (10a)$$

and arbitrary, nonlinear, one-variable,  $k$ th-order differentiable functions such as

$$\cos(u), \quad \sin(u), \quad \exp(u), \quad c^u, \quad \log(u) \quad \text{and} \quad au^b + c, \quad (10b)$$

for arbitrary constants  $a$ ,  $b$  and  $c$ . Systematic rules for constructing general  $k$ th-order calculus subroutines for special functions such as functions (10) are presented in Ref. [2].

The incorporation of *Feed* into a FORTRAN program for solving the complete differential system (7) is detailed in Ref. [6]. Additional applications and theoretical extensions of *Feed*, e.g. the tracking of eigenvalues and eigenvectors for parameterized matrices, are discussed in Ref. [7].

#### 4. OBTAINING INITIAL CONDITIONS VIA TWO-PHASE COMPLEX CONTINUATION

Recall from Section 2 that the initial conditions (7e-g) needed to integrate the complete differential system (7) from a given initial parameter point  $\alpha^0$  consist of a solution vector  $\mathbf{x}(\alpha^0)$  to  $\Psi(\mathbf{x}, \alpha^0) = \mathbf{0}$ , together with evaluations for the adjoint  $A(\alpha^0)$  and determinant  $\delta(\alpha^0)$  of the Jacobian matrix  $\Psi_{\mathbf{x}}(\mathbf{x}(\alpha^0), \alpha^0)$ .

In Ref. [6] it was suggested that a continuation method might be used to simplify the process of obtaining these required initial conditions. Specifically, letting  $F(\mathbf{x}) \equiv \Psi(\mathbf{x}, \alpha^0)$ , it was suggested that the modified system of equations

$$\mathbf{0} = F^{\wedge}(\mathbf{x}, \mu) \equiv \mu \cdot F(\mathbf{x}) + [1 - \mu] \cdot [\mathbf{x} - \mathbf{c}] \quad (11)$$

could be solved for  $\mathbf{x}$  as a function of  $\mu$  over the  $\mu$ -interval  $[0, 1]$  by numerically integrating an associated set of ODEs of the form (7).

At  $\mu = 0$ , system (11) takes the simple form

$$\mathbf{0} = F^{\wedge}(\mathbf{x}, 0) \equiv I \cdot [\mathbf{x} - \mathbf{c}]. \quad (12)$$

The initial conditions required at  $\mu = 0$  are  $\mathbf{x}^{\wedge}(0) = \mathbf{c}$ ,  $A^{\wedge}(0) = I$  and  $\delta^{\wedge}(0) = 1$ . For each  $\mu > 0$ , one obtains a solution vector  $\mathbf{x}^{\wedge}(\mu)$  for system (11), together with evaluations of the adjoint  $A^{\wedge}(\mu)$  and the determinant  $\delta^{\wedge}(\mu)$  of the Jacobian matrix  $F_{\mathbf{x}}^{\wedge}(\mathbf{x}^{\wedge}(\mu), \mu)$ . At  $\mu = 1$ , system (11) coincides with the original system of interest.

$$\mathbf{0} = F^{\wedge}(\mathbf{x}, 1) \equiv F(\mathbf{x}). \quad (13)$$

Recalling that  $F(\mathbf{x}) \equiv \Psi(\mathbf{x}, \alpha^0)$ , it follows from system (13) that  $\mathbf{x}^{\wedge}(1)$ ,  $A^{\wedge}(1)$  and  $\delta^{\wedge}(1)$  in principle provide the required initial conditions (7e-g) for the original differential system (7).

Subsequent numerical experimentation has shown that the suggested use of the continuation (11) to obtain the initial conditions (7e-g) was overly optimistic. The principal difficulty with system (11) is that the artificial component  $[1 - \mu] \cdot [\mathbf{x} - \mathbf{c}]$  involves the vector  $\mathbf{x}$ . Consequently, the Jacobian matrix

$$F_{\mathbf{x}}^{\wedge}(\mathbf{x}(\mu), \mu) = \mu \cdot F_{\mathbf{x}}(\mathbf{x}(\mu)) + [1 - \mu] \cdot I \quad (14)$$

is an artificial construct which changes in potentially complicated ways along the  $\mu$ -integration path. In particular, the artificial component can induce singularities in  $F_{\mathbf{x}}^{\wedge}(\mathbf{x}(\mu), \mu)$  even when the Jacobian matrix  $F_{\mathbf{x}}(\mathbf{x}(\mu))$  for the original system is well-behaved. In simulation experiments, these singular points were avoided by letting  $\mu$  take a prespecified U-shaped detour into the complex plane; but other singularities or regions of near-singularity were sometimes encountered along the complex  $\mu$ -path which destroyed or significantly reduced integration accuracy.

These difficulties suggested that it would be desirable to have the continuation proceed through the original system function  $F$  rather than through an artificial construct. First, in many applications the function  $F$  represents a physical process; and the singular points for such functions

tend to be fairly well-behaved (e.g. isolated), or at least fairly well understood. In contrast, as seen with system (11), even the simplest artificially constructed continuation can have singularities or regions of near-singularity which are difficult to determine in advance. Second, having the integration path proceed through  $F$  rather than through an artificial construct has the important advantage that potentially useful information about  $F$  is obtained at each point along the path rather than only at the endpoint of the path.

While certainly not the only way to proceed, one continuation which satisfies these requirements is the following simple translation of  $F$ :

$$\mathbf{0} = F^{**}(\mathbf{x}, \beta) \equiv [F(\mathbf{x}) - F(\mathbf{c})] + \beta \cdot F(\mathbf{c}). \quad (15)$$

As  $\beta$  varies from 0 to 1, the system of equations (15) varies from the translated form  $\mathbf{0} = [F(\mathbf{x}) - F(\mathbf{c})]$  to the system of interest  $\mathbf{0} = F(\mathbf{x})$ . However, the Jacobian matrix for system (15) is not an artificial construct; at each point  $\beta$ ,  $F_x^{**}(\mathbf{x}, \beta)$  coincides with the basic Jacobian matrix  $F_x(\mathbf{x})$ .

A differential system analogous to system (7) can be used to solve system (15) over the interval  $\beta = 0$  to  $\beta = 1$ . The initial conditions needed at  $\beta = 0$  are the solution vector  $\mathbf{x} = \mathbf{c}$ , together with evaluations for the adjoint and the determinant of the Jacobian matrix  $F_x(\mathbf{c})$ . The vector  $\mathbf{c}$  might be chosen to facilitate some standard algebraic method for obtaining these evaluations. Alternatively, these evaluations can be obtained analytically by solving the continuation

$$\mathbf{0} = F^*(\mathbf{x}, \theta) \equiv \theta \cdot [F(\mathbf{x}) - F(\mathbf{c})] + [1 - \theta] \cdot J \cdot [\mathbf{x} - \mathbf{c}] \quad (16)$$

over the interval  $\theta = 0$  to  $\theta = 1$  by means of an associated differential system of the form (7).

In system (16),  $\mathbf{c}$  denotes an initial guess for a solution vector  $\mathbf{x}(\alpha^0)$  to  $F(\mathbf{x}) = 0$  and  $J$  denotes an initial guess for the Jacobian matrix  $F_x(\mathbf{c})$ . For example,  $J$  could be the identity matrix  $I$ . The expression  $J \cdot [\mathbf{x} - \mathbf{c}]$  thus constitutes a linear approximation for  $[F(\mathbf{x}) - F(\mathbf{c})]$ , expanded around  $\mathbf{c}$ . Unlike system (11), the solution vector  $\mathbf{x}(\theta)$  for system (16) takes the constant value  $\mathbf{c}$  for all  $\theta$ , and the Jacobian matrix for system (16) exhibits a constant rate of change with respect to  $\theta$ . As  $\theta$  varies from 0 to 1, system (16) varies from the linear approximation system  $\mathbf{0} = J \cdot [\mathbf{x} - \mathbf{c}]$  to the system  $\mathbf{0} = [F(\mathbf{x}) - F(\mathbf{c})]$ . Thus, system (16) at  $\theta = 1$  coincides with system (15) at  $\beta = 0$ ; and the solution for system (16) at  $\theta = 1$  provides the required initial conditions for the solution of system (15) at  $\beta = 0$ .

Two questions need to be addressed in more detail. First, do the continuations (15) and (16) generate the required initial conditions (7e–g), as claimed? Second, how are integration paths from 0 to 1 to be determined for the continuation parameters  $\beta$  and  $\theta$ ?

For simplicity, the initial condition question is considered in Subsection 4.1 without specifying the exact paths followed by the continuation parameters. Subsection 4.2 takes up the sequential determination of these paths in the complex plane.

#### 4.1. Initial conditions via two-phase continuation

In this subsection it is assumed without further comment that the continuations (15) and (16) are each solved by means of an associated system of ODEs of the form (7) over an integration path from 0 to 1 for the continuation parameter. Consequently, at each parameter point one obtains a solution vector together with evaluations for the adjoint and determinant of the system Jacobian matrix.

At  $\theta = 0$ , the system of equations (16) for the  $\theta$ -continuation phase reduces to

$$\mathbf{0} = F^*(\mathbf{x}, 0) = J \cdot [\mathbf{x} - \mathbf{c}]. \quad (17)$$

If, for simplicity,  $J$  is specified to be the identity matrix  $I$ , the initial conditions required at  $\theta = 0$  are  $\mathbf{x}^*(0) = \mathbf{c}$ ,  $A^*(0) = I$  and  $\delta^*(0) = 1$ .

At each subsequent  $\theta$  point, one obtains the (constant) solution vector  $\mathbf{x}^*(\theta) = \mathbf{c}$  for system (16), together with evaluations for the adjoint  $A^*(\theta)$  and the determinant  $\delta^*(\theta)$  of the Jacobian matrix:

$$J^*(\theta) \equiv F_x^*(\mathbf{x}(\theta), \theta) = \theta \cdot F_x(\mathbf{c}) + [1 - \theta] \cdot J. \quad (18)$$

The components of the Jacobian matrix  $J^*(\theta)$  have constant rates of change with respect to  $\theta$ , i.e.

$$dJ^*(\theta)/d\theta = [F_x(\mathbf{c}) - J]; \quad (19)$$

and these rates of change are reduced in magnitude to the extent that  $J$  is a good initial guess for  $F_x(\mathbf{c})$ .† At the initial point  $\theta = 0$ ,  $J^*(\theta) = J$ . At the terminal point  $\theta = 1$ ,  $J^*(\theta)$  coincides with  $F_x(\mathbf{c})$ . Consequently, the evaluations for the adjoint  $A^*(1)$  and determinant  $\delta^*(1)$  of the Jacobian matrix  $J^*(1)$  also yield evaluations for the adjoint and determinant of the Jacobian matrix  $F_x(\mathbf{c})$ .

At  $\beta = 0$ , the system of equations (15) for the  $\beta$ -continuation phase reduces to

$$\mathbf{0} = F^{**}(\mathbf{x}, 0) = [F(\mathbf{x}) - F(\mathbf{c})]. \quad (20)$$

The initial conditions required at  $\beta = 0$  are provided by the terminal conditions from the preceding continuation phase: namely,  $\mathbf{x}^{**}(0) = \mathbf{x}^*(1) = \mathbf{c}$ ,  $A^{**}(0) = A^*(1) = \text{adjoint of } F_x(\mathbf{c})$  and  $\delta^{**}(0) = \delta^*(1) = \text{determinant of } F_x(\mathbf{c})$ .

At each subsequent  $\beta$ -point, the Jacobian matrix for system (15) coincides with the Jacobian matrix for the original function of interest  $F(\mathbf{x})$  evaluated at  $\mathbf{x} = \mathbf{x}^{**}(\beta)$ ; i.e.

$$J^{**}(\beta) \equiv F_x^{**}(\mathbf{x}^{**}(\beta), \beta) = F_x(\mathbf{x}^{**}(\beta)). \quad (21)$$

Moreover, at  $\beta = 1$ , system (15) reduces to the original system of equations:

$$\mathbf{0} = F^{**}(\mathbf{x}, 1) = F(\mathbf{x}). \quad (22)$$

At  $\beta = 1$  one obtains a solution vector  $\mathbf{x}^{**}(1)$ , together with evaluations for the adjoint  $A^{**}(1)$  and determinant  $\delta^{**}(1)$  of the Jacobian matrix  $J^{**}(1)$ . Recalling that  $F(\mathbf{x}) \equiv \Psi(\mathbf{x}, \alpha^0)$ , it follows from equations (21) and (22) that  $\mathbf{x}^{**}(1)$ ,  $A^{**}(1)$  and  $\delta^{**}(1)$  yield evaluations for  $\mathbf{x}(\alpha^0)$ ,  $A(\alpha^0)$  and  $\delta(\alpha^0)$ , the initial conditions (7e–g) required for the differential system (7).

In practice, round-off and truncation errors may affect the accuracy of the evaluations obtained for  $\mathbf{x}(\alpha^0)$ ,  $A(\alpha^0)$  and  $\delta(\alpha^0)$  by means of the  $\beta$ - and  $\theta$ -continuations, especially when the percentage error in the initial guess  $\mathbf{c}$  is large, say 1000%. As reported in Section 5, using these evaluations as the initial conditions for a second iteration of the  $\beta$ - and  $\theta$ -continuations has in some cases resulted in significantly improved calculations.

#### 4.2. Sequential determination of complex integration paths for $\beta$ and $\theta$

Singularity problems are potentially ameliorated by the  $\beta$ - and  $\theta$ -continuations (15) and (16), but they are not eliminated. Since locating singular points or regions of near-singularity prior to actual integration is generally not practical, the problem of determining integration paths for  $\beta$  and  $\theta$  was considered as a sequential multicriteria optimization problem with two potentially conflicting criteria: namely, (1) keep the paths short (minimal number of integration steps); and (2) keep the paths numerically stable (avoidance of singular points).

Consideration of these two criteria resulted in the development of an algorithm for the step-by-step evolution of integration paths for  $\beta$  and  $\theta$  in the complex plane. As a natural consequence, all variables in the algorithm are considered to be complex-valued.‡

This subsection describes the basic properties of the algorithm. For expositional clarity, a number of technical considerations are omitted. A more detailed discussion of the algorithm and its computer implementation is given in the Appendix; see also Ref. [8]. Since the algorithm is similarly applied in both the  $\beta$ - and  $\theta$ -continuation phases, the symbol  $\lambda$  is used below to denote either of the continuation parameters  $\beta$  or  $\theta$ .

A basic assumption maintained throughout this subsection is that at least one path exists from  $\lambda = 0 + 0i$  to  $\lambda = 1 + 0i$  along which the absolute value of the system determinant  $\delta(\lambda)$  is uniformly bounded away from zero.§ However, no such path is known *a priori*. The problem is then to

†As reported in Section 5, setting  $J = I$  has in fact worked very well in simulation experiments run to date.

‡The use of complex-valued variables is potentially beneficial, regardless of the method proposed for solving  $\mathbf{0} = F(\mathbf{x})$ . If the components of  $F$  involve exponentials or logarithms, as is often the case, then difficulties can arise during the course of the solution procedure in the form of negative radicands or negative arguments for logarithms. The use of complex-valued variables eliminates these difficulties.

§The determinant  $\delta(\lambda)$  denotes  $\delta^*(\theta)$  when  $\lambda = \theta$  and  $\delta^{**}(\beta)$  when  $\lambda = \beta$ .

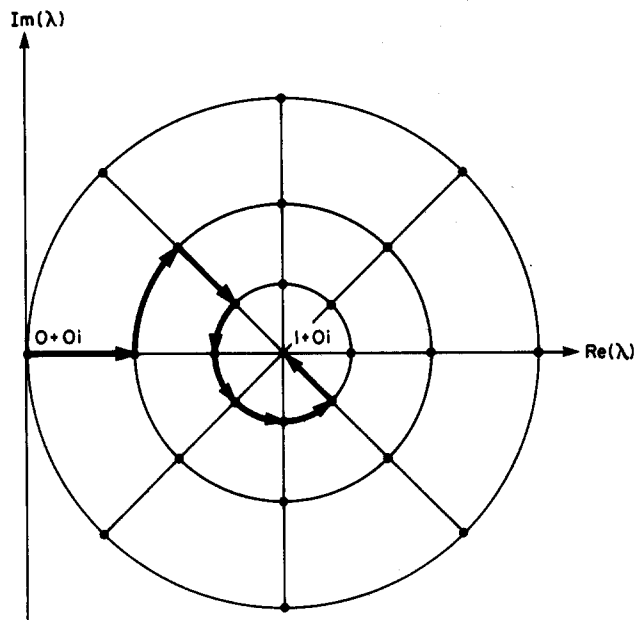


Fig. 1. Schematic illustration of a sequentially determined integration path for the continuation parameter  $\lambda$  through a complex spider-web grid.

determine, on a step-by-step basis, an actual path of integration from  $\lambda = 0 + 0i$  to  $\lambda = 1 + 0i$  in approximate agreement with the shortness and stability criteria.

For simplicity, this problem is addressed in two stages. First, on what type of grid is  $\lambda$  going to be allowed to move? Second, how is the actual path taken by  $\lambda$  through the grid to be decided?

First consider the grid. If the integration path is to be kept short, then it should be geometrically possible to proceed in a direct line to the desired endpoint  $1 + 0i$  from any given current point on the grid. If the integration path is to be numerically stable, singular points must of course be avoided; but the geometry of the grid should permit this avoidance to be carried out efficiently with respect to the shortness criterion. In particular, it should be geometrically possible to step away from a singular point without increasing the distance to the endpoint  $1 + 0i$ . This in turn suggests that the grid mesh should be denser in a neighborhood of the endpoint  $1 + 0i$  in order to permit intricate paths to evolve in this neighborhood without increasing the distance from  $1 + 0i$ .

One simple grid specification which satisfies these geometric requirements is the spider-web grid depicted in Fig. 1. This spider-web grid consists of a nested family of concentric circles ("rims") in the complex plane with common center  $1 + 0i$ , and with a number of equally-spaced rays ("spokes") branching out from this common center. Points on the grid are defined by the intersections of spokes and rims. Starting from any current point on the grid, it is geometrically possible for the continuation parameter  $\lambda$  to proceed in a direct line to the endpoint  $1 + 0i$  by stepping inward along the current spoke. On the other hand, a singular point along a current spoke can be avoided by taking a suitable number of rim-steps before again attempting an inward spoke-step. Rim-steps do not increase the distance to the endpoint  $1 + 0i$ . Finally, the grid mesh along rims automatically becomes finer in a neighborhood of  $1 + 0i$ .

One possible path for the continuation parameter  $\lambda$  through the spider-web grid is depicted in Fig. 1. How is the exact path taken by  $\lambda$  to be decided? The basic steps of the algorithm developed for the step-by-step determination of this path take the following form.

Suppose a minimum tolerance TOL has been set for the absolute value  $|\delta(\lambda)|$  of the determinant of the Jacobian matrix. Starting at any current point  $\lambda$  on the grid, an inward step  $\Delta\lambda$  along the current spoke is considered. If this spoke-step passes the tolerance test, i.e. if  $|\delta(\lambda + \Delta\lambda)| \geq \text{TOL}$ , then the spoke-step is actually taken. Another inward step  $\Delta\lambda$  along the current spoke is then considered. If this process continues without ever encountering a tolerance test failure, then eventually the desired endpoint  $\lambda = 1 + 0i$  is attained by successive inward steps along the current spoke.

On the other hand, if some considered spoke-step fails the tolerance test, additional tolerance tests are performed to see if a rim-step to a new spoke is possible at the current tolerance level. If so, the rim-step is taken; and an attempt is then made to have the continuation parameter attain the endpoint  $1 + 0i$  by successive inward steps along the new spoke, in the manner described above. If not, the minimum tolerance TOL is adjusted downward until either an inward spoke-step or a rim-step away from the current point does pass the tolerance test. This step is then taken, and again an attempt is made to have the continuation parameter attain the endpoint  $1 + 0i$  by successive inward steps along the new spoke. If a complete revolution around the current rim is made without finding a tolerable inward spoke-step, then TOL is adjusted downward until an inward spoke-step from some point along the current rim does pass the tolerance test.† This spoke-step is then taken, and an attempt is made to continue stepping inward along this spoke until the endpoint  $1 + 0i$  is attained.

In principle, this sequential procedure should allow the continuation parameter  $\lambda$  to make its way through the spider-web grid to the desired endpoint  $1 + 0i$  along a path which is both reasonably short and reasonably distant from singular points. The increasing fineness of the grid mesh along rims in a neighborhood of the endpoint  $1 + 0i$  should permit  $\lambda$  to find its way to the endpoint  $1 + 0i$  even when the only tolerable path into  $1 + 0i$  is a narrow curvy ridge. The possible reduction in the minimum tolerance TOL along the integration path also plays a potentially important role in permitting  $\lambda$  to reach  $1 + 0i$ . For example, TOL may have to be substantially reduced over the final portion of the integration path if the endpoint  $1 + 0i$  is surrounded by a region in which the determinant  $\delta(\lambda)$  is nearly zero. An example of this kind is given in the next section.

## 5. NUMERICAL EXPERIMENTS

A single-precision FORTRAN program *Nasa* has been written to numerically solve the complete differential system (7)—see the Appendix.‡ *Nasa* incorporates both the automatic differentiation procedure *Feed* outlined in Section 3 and the two-phase initialization procedure described in Section 4. This section reports on several numerical experiments undertaken with *Nasa* to test out the two-phase initialization procedure. Typical CPU running times on an IBM 3090 were in the order of 1–10 s.

The basic problem is to find a solution vector  $\mathbf{x}(\alpha^0)$  for a system of equations  $\mathbf{0} = \Psi(\mathbf{x}, \alpha^0)$ , together with evaluations  $A(\alpha^0)$  and  $\delta(\alpha^0)$  for the adjoint and determinant of the Jacobian matrix  $\Psi_{\mathbf{x}}(\mathbf{x}(\alpha^0), \alpha^0)$ , assuming  $\Psi(\cdot)$  is continuously differentiable and at least one solution vector  $\mathbf{x}(\alpha^0)$  exists for which  $\delta(\alpha^0)$  is nonzero. In contrast to successive approximation methods, the two-phase initialization procedure for finding such a solution vector is exact. That is, if the integrations along the closed  $\theta$ - and  $\beta$ -curves from  $0 + 0i$  to  $1 + 0i$  could be carried out in an exact manner, with all singular points successfully avoided, then in theory the evaluation  $\mathbf{x}^{**}(1)$  found at  $\beta = 1 + 0i$  should yield a solution vector  $\mathbf{x}(\alpha^0)$  for the original system of equations.

In practice, round-off and truncation errors degrade the accuracy of the evaluation  $\mathbf{x}^{**}(1)$ , especially when the initial guess  $\mathbf{c}$  for the solution vector is highly inaccurate. Specifying a smaller integration step-size or an increased word length are two obvious ways in which one could attempt to increase numerical accuracy. Another possibility, however, is to turn the two-phase initialization procedure into a successive approximation method by iterating the successive  $\theta$ - and  $\beta$ -continuations. The latter alternative has worked well in numerical experiments conducted to date.

For illustration, first consider the following variant of a single-equation problem investigated in Ref. [9]:

$$\mathbf{0} = \Psi(\mathbf{X}, \alpha^0) \equiv \mathbf{x} - 1 + \log(\alpha^0) + \log(\mathbf{x}), \quad (23a)$$

†Alternatively, for analytical functions  $\Psi$  one could use Cauchy residue formulas to obtain the desired evaluations at the center point  $\lambda = 1 + 0i$  by appropriate integrations around the circular closed curve defined by the current rim.

‡For the reasons explained in Subsection 4.2, all variables in *Nasa* are complex-valued. The computer software available to us prohibited the use of double-precision with complex-variable calculations.

Table 2. Numerical solution of example (23)

| Starting value $c$ | No. of $\theta$ - $\beta$ -iterations | Complex $\theta$ - $\beta$ -paths? (No. of steps) | Numerical solution                |                                    |                                     |
|--------------------|---------------------------------------|---|-----------------------------------|------------------------------------|-------------------------------------|
|                    |                                       |   | $x$                               | $\delta$                           | $\Psi(x, \alpha^0)$                 |
| $0.40 + 0i$        | 1                                     | No<br>(400)                                       | 0.80789E + 00                     | 0.22377E + 01                      | -0.24795E - 04                      |
| $0.40 + 0i$        | 2                                     | No<br>(800)                                       | 0.80788E + 00                     | 0.22377E + 01                      | -0.37551E - 05                      |
| $100 + 0i$         | 1                                     | No<br>(400)                                       | 0.80137E + 00                     | 0.22483E + 01                      | -0.14694E - 01                      |
| $100 + 0i$         | 2                                     | No<br>(800)                                       | 0.80787E + 00                     | 0.22377E + 01                      | -0.11906E - 04                      |
| $10 + 10i$         | 1                                     | No<br>(400)                                       | 0.80751E + 00                     | 0.22382E + 01                      | -0.84686E - 03                      |
| $10 + 10i$         | 2                                     | No<br>(800)                                       | 0.80788E + 00                     | 0.22377E + 01                      | -0.84629E - 05                      |
| $-1 + 0i$          | 1                                     | Yes<br>(1638)                                     | (0.53587E + 01,<br>0.90648E + 01) | (0.47044E + 01,<br>-0.90757E + 01) | (-0.35990E + 01,<br>0.11170E + 02)  |
| $-1 + 0i$          | 2                                     | Yes<br>(2238)                                     | (0.80788E + 00,<br>-0.6109E - 04) | (0.22378E + 00,<br>0.11628E - 03)  | (-0.19074E - 05,<br>-0.01478E - 03) |

with

$$\alpha^0 = 1.5. \tag{23b}$$

The unique solution for problem (23) is  $x(\alpha^0) = 0.807878\dots$ , with  $\delta(\alpha^0) = 2.23781\dots$

Various numerical experiments were run for problem (23) with different starting values  $c$ , and with either one or two iterations of the successive  $\theta$ - and  $\beta$ -continuations.† As indicated in Table 2, at most, two iterations resulted in convergence to approximate values for  $x(\alpha^0)$  and  $\delta(\alpha^0)$  which were accurate to about four or five decimal places, even for highly inaccurate starting values  $c$ . For these values,  $\Psi$  itself was reduced to about  $10^{-5}$ .

What kinds of integration paths evolved for  $\theta$  and  $\beta$  in each case? As detailed in Section 4, the continuation parameters  $\theta$  and  $\beta$  in each iteration moved from  $0 + 0i$  to  $1 + 0i$  through a spider-web grid in the complex plane along integration paths which were sequentially determined in an attempt to keep the paths both short and numerically stable. For each different starting value  $c$  in Table 2, the integration path which evolved for  $\beta$  in each iteration consisted of 200 integration steps of size 0.005 from  $0 + 0i$  to  $1 + 0i$  along the real axis; deviations into the complex plane to avoid singular points or regions of near-singularity did not turn out to be necessary. The same was true for the integration paths for  $\theta$ , with one exception: namely, the rather whimsical starting value  $c = -1.0 + 0i$ .

For the latter case, the first  $\theta$ - $\beta$ -iteration resulted in the evolution of an intricate path for the continuation parameter  $\theta$  in a small neighborhood of the desired endpoint  $\theta = 1 + 0i$ . Specifically,  $\theta$  took 180 integration steps of size 0.005 to reach  $\theta = 0.9 + 0i$  along the real line. However, a total of 1438  $1^\circ$  rim-steps and 20 spoke-steps of size 0.005 were then taken to reach  $\theta = 1 + 0i$ ; and the tolerance level TOL along this path was sequentially reduced from  $0.007E + 00$  at  $\theta = 0.9 + 0i$  to  $0.389E - 03$  at  $\theta = 1 + 0i$ . The succeeding  $\beta$ -path was entirely real, but the values obtained at the end of the first  $\theta$ - $\beta$ -iteration were far from satisfactory. Despite all this, the second  $\theta$ - $\beta$ -iteration took only 400 integration steps of size 0.005, with real  $\theta$ - and  $\beta$ -paths, to converge to a value for  $x$  which was accurate to five decimal places.

A more challenging two-equation system will now be considered. Consider the problem of numerically solving

$$0 = \Psi(x, \alpha^0), \quad \alpha^0 = 0.5, \tag{24a}$$

where the function  $\Psi = (\Psi^1, \Psi^2)$  taking  $R^2_{++} \times R$  into  $R^2$  is defined by

$$\Psi^1(x_1, x_2, \alpha) \equiv (1/2) \cdot (x_1)^{-1/2} \cdot (x_2)^{1/3} - \alpha \tag{24b}$$

†The following common program specifications were used for all the numerical experiments reported in this section (see the Appendix for detailed explanations):  $J = I$ ;  $VSS = 0.005$ ;  $NSPS = 200$ ;  $MAXRS = 360$ ;  $TOL = 0.007$ ;  $NR = 1$ ; and  $RED = 0.90$ .

and

$$\Psi^2(\mathbf{x}_1, \mathbf{x}_2, \alpha) \equiv (1/3) \cdot (\mathbf{x}_1)^{1/2} \cdot (\mathbf{x}_2)^{-2/3} - (1/3). \tag{24c}$$

The unique solution for problem (24) is  $\mathbf{x}(\alpha^0) = (1, 1)$ , with  $\delta(\alpha^0) = 0.02777 \dots$

Problem (24) was considered in Ref. [10], where it arose as the first-order necessary conditions for the maximization of firm profits with a ‘‘Cobb–Douglas’’ production function. Attempts to solve problem (24) by means of a simple  $\mu$ -continuation of the form (11) with the continuation parameter  $\mu$  following a real path from 0 to 1 failed.

It turns out that the Jacobian matrix for the  $\mu$ -continuation (11) for problem (24) has a singular point along the real interval  $\mu = 0$  to  $\mu = 1$ , so that an entirely real path for  $\mu$  is not feasible. Attempts to side-step the singular point by having  $\mu$  take a prespecified U-shaped detour into the complex plane were only moderately successful. Regions of near-singularity were encountered along the path which reduced integration accuracy, especially for starting vectors  $\mathbf{c}$  which were an appreciable distance away from the exact solution vector  $(1 + 0i, 1 + 0i)$ .

When *Nasa* was used for the numerical solution of problem (24) for a variety of starting vectors,  $\mathbf{c}$ , it was found that at most two iterations of the successive  $\theta$ - and  $\beta$ -continuations typically resulted in approximate values for  $\mathbf{x}(\alpha^0)$  and  $\delta(\alpha^0)$  which were accurate to about four or five decimal places. For these values, the real and imaginary parts of the component functions of  $\Psi$  were reduced to about  $10^{-5}$  and  $10^{-6}$ , respectively. Some of these numerical experiments are summarized in Table 3.

In each case depicted in Table 3, the integration paths which evolved for  $\beta$  in the first and second  $\theta$ - $\beta$ -iterations, and for  $\theta$  in the second  $\theta$ - $\beta$ -iteration, consisted of 200 steps of size 0.005 from  $0 + 0i$  to  $1 + 0i$  along the real axis. In contrast, in the first  $\theta$ - $\beta$ -iteration for each case, the integration path which evolved for  $\theta$  became complex and intricate in a small neighborhood of the endpoint  $1 + 0i$ .

For example, in the first iteration of the experiment with starting vector  $\mathbf{c} = (15 + 0i, 5 + 0i)$ , the continuation parameter  $\theta$  first took 180 integration steps of size 0.005 along the real axis at the tolerance level  $TOL = 0.007E + 00$ . A tolerance test failure then occurred when a further inward step along the real-axis ‘‘spoke’’ was considered. The continuation parameter  $\theta$  subsequently had to take 48  $1^\circ$  rim-steps in the counterclockwise direction before it found another spoke along which it could continue stepping inward in the direction of the desired endpoint  $1 + 0i$  at the current minimum tolerance level  $TOL = 0.007E + 00$ . An additional 1925 integration steps, and a reduction in  $TOL$  from  $0.007E + 00$  down to  $0.247E - 03$ , were ultimately needed to achieve  $1 + 0i$  through the spider-web grid.

In summary, allowing for the round-off and truncation errors which inevitably arise from the use of single-precision calculations, the results in Tables 2 and 3 suggest that the  $\theta$ - $\beta$ -continuation

Table 3. Numerical solution of example (24)

| $\mathbf{c}_1$<br>$\mathbf{c}_2$ | No. of $\theta$ - $\beta$ -iterations | Complex $\theta$ - $\beta$ -paths? (No. of steps) | Numerical solution               |                              |  |
|----------------------------------|---------------------------------------|---|----------------------------------|------------------------------|--|
|                                  |                                       |   | $\mathbf{x}_1$<br>$\mathbf{x}_2$ | $\delta$                     | $\Psi^1(\mathbf{x}, \alpha^0)$<br>$\Psi^2(\mathbf{x}, \alpha^0)$ |
| $1.2 + 0i$                       | 1                                     | Yes   | $(0.10E + 01, + 0.79E - 05)$     | $(0.28E - 01, + 0.13E - 05)$ | $(0.39E - 05, - 0.75E - 06)$                                     |
| $1.1 + 0i$                       |                                       | (459)   | $(0.10E + 01, + 0.73E - 05)$     |                              | $(0.25E - 05, - 0.32E - 06)$                                     |
| $6.0 + 0i$                       | 1                                     | Yes   | $(0.10E + 01, - 0.12E - 02)$     | $(0.27E - 01, + 0.63E - 04)$ | $(- 0.15E - 02, 0.91E - 04)$                                     |
| $5.0 + 0i$                       |                                       | (1224)  | $(0.10E + 01, - 0.12E - 02)$     |                              | $(- 0.12E - 02, 0.68E - 04)$                                     |
| $6.5 + 0i$                       | 2                                     | Yes   | $(0.10E + 01, - 0.51E - 06)$     | $(0.28E - 01, + 0.84E - 06)$ | $(0.86E - 05, + 0.34E - 07)$                                     |
| $5.0 + 0i$                       |                                       | (1624)  | $(0.10E + 01, + 0.56E - 06)$     |                              | $(0.36E - 05, + 0.40E - 07)$                                     |
| $10 + 0i$                        | 1                                     | Yes   | $(0.11E + 01, - 0.52E - 03)$     | $(0.23E - 01, + 0.23E - 04)$ | $(- 0.58E - 02, 0.37E - 04)$                                     |
| $9 + 0i$                         |                                       | (2192)  | $(0.11E + 01, - 0.54E - 03)$     |                              | $(- 0.51E - 02, 0.30E - 04)$                                     |
| $10 + 0i$                        | 2                                     | Yes   | $(0.10E + 01, - 0.28E - 06)$     | $(0.28E - 01, - 0.54E - 07)$ | $(0.67E - 05, 0.20E - 07)$                                       |
| $9 + 0i$                         |                                       | (2592)  | $(0.10E + 01, - 0.30E - 06)$     |                              | $(0.44E - 05, 0.20E - 07)$                                       |
| $15 + 0i$                        | 1                                     | Yes   | $(0.11E + 01, + 0.58E - 04)$     | $(0.22E - 01, + 0.10E - 06)$ | $(- 0.12E - 01, - 0.29E - 05)$                                   |
| $5 + 0i$                         |                                       | (2153)  | $(0.11E + 01, + 0.66E - 04)$     |                              | $(- 0.33E - 02, - 0.46E - 05)$                                   |
| $15 + 0i$                        | 2                                     | Yes   | $(0.10E + 01, - 0.27E - 07)$     | $(0.28E - 01, - 0.71E - 08)$ | $(0.62E - 05, 0.20E - 08)$                                       |
| $5 + 0i$                         |                                       | (2753)  | $(0.10E + 01, - 0.29E - 07)$     |                              | $(0.50E - 05, 0.19E - 08)$                                       |
| $15 + 0i$                        | 1                                     | Yes   | $(0.12E + 01, - 0.32E - 02)$     | $(0.17E - 01, + 0.79E - 04)$ | $(- 0.15E - 01, 0.16E - 03)$                                     |
| $15 + 0i$                        |                                       | (3129)  | $(0.13E + 01, - 0.35E - 02)$     |                              | $(- 0.15E - 01, 0.19E - 03)$                                     |
| $15 + 0i$                        | 2                                     | Yes   | $(0.10E + 01, + 0.46E - 04)$     | $(0.28E - 01, - 0.10E - 03)$ | $(0.29E - 05, - 0.99E - 05)$                                     |
| $15 + 0i$                        |                                       | (3732)  | $(0.10E + 01, + 0.99E - 05)$     |                              | $(0.38E - 05, + 0.55E - 05)$                                     |

procedure incorporated into *Nasa* provides a potentially useful tool for obtaining at least preliminary estimates for the initial conditions required by the complete differential system (7).

## 6. DISCUSSION

This paper has considered the problem of finding a solution vector  $\mathbf{x}(\alpha)$  for a parameterized system of equations  $\mathbf{0} = \Psi(\mathbf{x}, \alpha)$ , together with evaluations for the adjoint and determinant of the Jacobian matrix  $\Psi_{\mathbf{x}}(\mathbf{x}(\alpha), \alpha)$ , at an initial parameter point  $\alpha = \alpha^0$ . Many procedures for solving nonlinear equations already exist which could have been applied to this initialization problem. However, our objective was to see whether this problem could be effectively solved by making use of the system of ODEs previously developed in Ref. [1] for tracking the solution vector  $\mathbf{x}(\alpha)$  over  $\alpha$ -intervals. If so, the entire nonlocal sensitivity analysis could be carried out by making repeated use of the same basic computational steps.

In keeping with this objective, a two-phase continuation has been developed for obtaining all the required initial conditions. As clarified in previous sections, the two-phase continuation potentially ameliorates certain difficulties which can arise with single-phase continuations, such as continuation (11), that proceed entirely through an artificial construct. The complex integration path for the continuation parameter in each phase is determined sequentially rather than in advance in an attempt to keep the path both short and numerically stable.

A single-precision FORTRAN program *Nasa* with automatic derivative evaluation has been developed for solving the differential system in Ref. [1], starting with a two-phase initialization. Various numerical experiments have been conducted with *Nasa* to test the program. In these experiments the evaluations obtained for the initial conditions have been accurate to about four or five decimal places, even for highly inaccurate initial guesses  $\mathbf{c}$  for the solution vector  $\mathbf{x}(\alpha^0)$ , and despite the fact that no attempts were made to optimize the program specifications for the sequential determination of complex integration paths for the continuation parameters.

Future studies will undertake a more systematic experimental and theoretical investigation of the two-phase initialization procedure incorporated in *Nasa*. More generally, the capabilities of the program as a whole for carrying our nonlocal automated sensitivity analysis will be examined.

A number of applications are also planned, particularly in the area of "applied general equilibrium". The latter area is a subdiscipline of economics which has attracted increasing attention over the past 10 years [11–13]. An applied general equilibrium researcher typically attempts to "calibrate" a theoretical model of an economic process to a benchmark data set; but various key parameter values (e.g. elasticities) must often be incorporated from other studies in order to complete the calibration. At times there may be limited, contradictory, or even no information concerning these key parameter values, and the researcher must resort to "best guess" evaluations. An important question concerns the sensitivity of the model solution to changes in these evaluations.

## REFERENCES

1. R. Kalaba and L. Tefatsion, Complete comparative static differential equations. *J. nonlinear Analysis: TMA* **5**, 821–833 (1981).
2. R. Kalaba, L. Tefatsion and J.-L. Wang, A finite algorithm for the exact evaluation of higher-order partial derivatives of functions of many variables. *J. math. Analysis Applic.* **12**, 181–191 (1983).
3. E. Allgower and K. Georg, Simplicial and continuation methods for approximating fixed points and solutions to systems of equations. *Siam Rev.* **22**, 28–85 (1980).
4. R. Wengert, A simple automatic derivative evaluation program. *Commun. ACM* **7**, 463–464 (1964).
5. L. B. Rall, *Automatic Differentiation: Techniques and Applications*. Springer, New York (1981).
6. R. Kalaba, L. Tefatsion and J.-L. Wang, Local and nonlocal comparative static analysis of economic systems. *Appl. Math. Comput.* **9**, 227–234 (1981).
7. R. Kalaba and L. Tefatsion, Nonlocal sensitivity analysis, automatic derivative evaluation, and sequential nonlinear estimation. *Comput. Statist. Data Analysis* **4**, 79–91 (1986).
8. R. Kalaba and L. Tefatsion, Nonlocal automated sensitivity analysis. Modelling Research Group Working Paper No. 8911, Dept of Economics, Univ. of Southern California, Los Angeles, Calif. (July 1989).
9. A. Khilnani and E. Tse, A fixed point algorithm with economic applications. *J. econ. Dynam. Control* **9**, 127–137 (1985).
10. J.-L. Wang, The computation and the sensitivity analysis of economic equilibrium. Dissertation, Dept of Economics, Univ. of Southern California, Los Angeles, Calif. (July 1984).

11. S. Smale, Global analysis and economics. In *Handbook of Mathematical Economics*, Vol. 1, Chap. 8 (Eds K. J. Arrow and M. D. Intriligator), pp. 331–370. North-Holland, New York (1981).
12. F. Kydland and E. Prescott, Time to build and aggregate fluctuations. *Econometrica* **50**, 1345–1370 (1982).
13. A. Mansur and J. Whalley, Numerical specification of applied general equilibrium models. In *Applied General Equilibrium Analysis*, Chap. 3 (Eds H. E. Scarf and J. B. Shoven), pp. 69–137. Cambridge Univ. Press, New York (1984).

## APPENDIX

### *A FORTRAN Program for Nonlocal Automated Sensitivity Analysis*

A FORTRAN program *Nasa* (Nonlocal automated sensitivity analysis) with automatic derivative evaluation has been developed for solving the complete ODE system (7) starting with a two-phase initialization.† The logical progression of the program statements is outlined below.

*Nasa* consists of six program segments: (I) The MAIN program, together with a subroutine PSIFUN, which both require user inputs; (II) a subroutine OUTPUT which provides output for the  $\theta$ -,  $\beta$ - and  $\alpha$ -phases of the sensitivity analysis; (III) two subroutines FUN and INIT which direct the  $\theta$ - and  $\beta$ -continuation phases; (IV) three subroutines DAUX, CINT1 and CINT2 which oversee all necessary derivative evaluations and carry out all the required integrations; (V) a block of 16 calculus subroutines which constitute a *Feed* library for automatic evaluation of derivatives; and (VI) a block of six subroutines for carrying out various matrix operations.

Detailed explanations for the user inputs required by MAIN are provided in comment statements. Briefly, the user must specify: the number  $N$  of component functions of  $\Psi(x, \alpha)$ ; an initial value  $\alpha^0$  for the parameter  $\alpha$ ; an integration step-size  $H$  for  $\alpha$ ; the number NSP of  $\alpha$ -integration steps to be taken; the number NAP of  $\alpha$ -integration steps between output; an initial guess  $c$  for the solution vector  $x(\alpha^0)$ , and the maximum number MAXINT of integration steps to be permitted overall during the program execution (a safety stop device). He must also specify the number NIT of  $\theta$ - $\beta$ -iterations to be undertaken to improve on the initial guess  $c$  prior to the  $\alpha$ -integration. [For simplicity, the initial guess  $J$  for the Jacobian matrix  $\Psi_x(c, \alpha^0)$  is automatically specified to be the identity matrix  $I$ .]

Finally, MAIN requires the user to specify certain parameters which guide the sequential determination of integration paths for  $\theta$  and  $\beta$  in the complex plane: namely, an integration step-size VSS for stepping along spokes; the total number NSPS of spoke-steps to be taken (with  $VSS \cdot NSPS = 1$ ); the number NP of spoke-steps to be taken between output; the maximum number MAXRS of  $1^\circ$  rim-steps which can be taken along any one rim; an initial minimum tolerance level (TOL) for the absolute value of the determinant of the system Jacobian in the first  $\theta$ -continuation phase; the number NR of  $1^\circ$  rim-steps to be attempted before another tolerance test after a spoke-step tolerance test failure; and a percentage reduction factor RED for reducing TOL when no tolerable move from the current point can be found at the current tolerance level.

Detailed explanations for the user inputs required by the subroutine PSIFUN are also provided in comment statements. Making use of program segment V—the library of *Feed* calculus subroutines—the user must implement a “Wengert list” for each of the  $N$  component functions of  $\Psi$  by specifying a list of call statements to the appropriate calculus subroutines. [See Section 3 for an illustration; the implementation needed for function (8) is carried out by the sequence of call statements in subroutine (9).]

A list of program statements for *Nasa* is provided in Ref. [8, Appendix B]. This list is dimensioned for functions  $\Psi$  with  $N \leq 6$  component functions, and is specifically programmed for the numerical solution of the first case reported in Table 3:  $c = (1.2 + 0i, 1.1 + 0i)$  and  $NIT \equiv \text{No. of } \theta\text{-}\beta\text{-iterations} = 1$ . [See the footnote on p. 62 for additional program specifications.] To reprogram *Nasa* for another application, the user would need to make appropriate changes in MAIN and the subroutine PSIFUN (i.e. in program segment I), as described above. Aside from possible changes in dimension statements, the remaining program segments II–VI are not problem-specific and do not require user inputs.

---

†A copy of the program *Nasa* will be provided by the second author upon request; please send a blank  $5\frac{1}{4}$ " diskette. A listing of the program statements can also be found in Ref. [8].