ELSEVIER

Computers and Chemical Engineering 26 (2002) 1449–1457

Computers & Chemical Engineering

www.elsevier.com/locate/compchemeng

Numerical solution of non-linear algebraic equations with discontinuities

Mordechai Shacham [a,*], Neima Brauner [b]

[a] *Department of Chemical Engineering, Ben Gurion University of the Negev, Beer-Sheva 84105, Israel*
[b] *School of Engineering, Tel-Aviv University, Tel-Aviv 69978, Israel*

Received 13 April 2001; received in revised form 1 May 2002; accepted 1 May 2002

Abstract

The solution of systems of non-linear algebraic equations with discontinuities in the solution search domain is considered. It is demonstrated that such problems are often very difficult to solve, even with the state of the art numerical solvers, and even when initial guesses close to the solutions are used. The application of constrained solution methods that do not require evaluation of function and derivative values outside of a predefined (feasible) subspace of the variables for solving such systems is considered. An algorithm is presented for identifying and handling of sub-expressions that introduce discontinuities. These are either removed by algebraic manipulations, or defined as boundaries of a feasible subspace. Using the proposed approach, it is demonstrated that a feasible solution for originally unsolvable problems can be found. © 2002 Elsevier Science Ltd. All rights reserved.

Keywords: Root finding; Constrained non-linear equations; Function discontinuity

1. Introduction

Many practical problems in chemical engineering give rise to the need to solve systems of non-linear algebraic equations (NLE) with discontinuities and/or regions where some of the functions are undefined. Such points and regions may often lie in the vicinity of the solution. Typical examples that involve NLE with discontinuities include: calculation of chemical equilibrium by minimization of Gibbs energy or using equilibrium constants, continuous operation of some reactors (such as continuous stirred tank reactors, CSTR), heat exchanger calculations (where logarithmic mean temperature difference is used), calculation of the parameters of activity coefficient equations (such as the Van Laar and Wilson equations), solving various equations of state for specific volume or compressibility factor and calculation of the minimum reflux ratio using the Underwood equations.

Many powerful algorithms and codes for solving NLE have been developed in recent years, including methods that claimed to be ‘globally convergent’ (for a recent review, see for example Wilhelm & Swaney, 1994). Unfortunately, for NLE with discontinuity in the vicinity of the solution, all of the methods may fail. Most of the ‘globally convergent’ methods assume continuity, which is obviously not valid for the case under consideration. Minimum search methods that use only function values may ‘overstep’ the solution if the search resolution is not sufficiently high or/and may converge to a local minimum instead of the true solution.

Often, it is possible to remove the discontinuities by algebraic manipulation of the equations (e.g. by multiplying equations by denominators that may obtain zero value during the solution process), but in many cases it is impossible to remove all the discontinuities.

Shacham and Shacham (1990) used symbolic analysis of the equations system to detect discontinuities along a one-dimensional ray, which is the progress direction determined by the solution method. The step length along this direction is restricted so as to avoid entering regions where discontinuity has been detected. This is a very effective technique for solving NLE with discontinuities. Unfortunately, it is limited to fairly simple functions, where the zeros of the sub-expressions that introduce the discontinuities can be obtained analytically.

* Corresponding author. Tel.: +972-8-646-1481; fax: +972-8-647-2916
E-mail address: shacham@bgumail.bgu.ac.il (M. Shacham).

The so-called 'constrained' NLE solution methods can serve as a basis for a general approach for solving NLE with discontinuities. These methods search for the solution in a predefined subspace of the variables and do not require function or derivative evaluations outside this subspace. Several algorithms are available for solving constrained NLE: the *CONLES* algorithm of Shacham (1986), the iterative linear programming (LP) methods of Bullard and Biegler (1991), Wilhelm and Swaney (1994) and the multidimensional bisection method of Gupta (1995). The *CONLES* algorithm will be discussed in some details in the next section. In the iterative LP technique, linear programs are solved, while combining local Jacobian and global bounding information, in order to generate search directions that satisfy region feasibility. In the multidimensional bisection method, a sequence of one-dimensional problems is solved using bisection, in order to ensure that the variables stay inside the feasible region.

Successful applications of the *CONLES* algorithm in solving chemical problems are widely documented (for example, Lorenzini, Bertrand & Villermaux, 1991; Sarkar & Gupta, 1992; von Bergen, von Bergen & Rogel, 1997). Three most recent applications of this method include simulation of polymer absorption at the solid liquid interface by a continuum model (Juvekar, Anoop, Puttanayek & Naik, 1999), simulation of micro-phase enhanced reactions (Hasnat & Roy, 1999) and modeling the acid separation behavior of weak base ion exchange resins (Bhandari, Yonemoto & Juvekar, 2000).

In this paper the *CONLES* algorithm is used as a basis for developing a general approach for solving NLE with discontinuities. In the next section, the principles of solution of constrained NLE will be briefly reviewed. Typical difficulties in solving NLE with discontinuities will be demonstrated using three examples in Section 3. The Section 4 presents a general approach for solving NLE with discontinuities and finally some conclusions are presented.

2. Basic concepts

Let us denote the n-dimensional Euclidean space as R^n and let f be a function with domain and range in R^n. Then, the problem of solving n non-linear equations in n variables can be stated in vector form as

$$\mathbf{f}(\mathbf{x}) = 0 \tag{1}$$

Any, or all of the variables may be subject to constraints of the type

$$\mathbf{x}_j \geq 0, \quad j = 1, 2\ldots, \mathbf{m} \tag{2}$$

It is important to point out from the outset the similarities and dissimilarities between constrained NLE and constrained non-linear optimization. In both cases only a solution that satisfies the constraints (lies in the feasible region) is acceptable and is of physical significance. But, unlike in constrained optimization, the constraints of NLE are expected to be inactive at the solution. This is because in Eq. (1) both **f** and **x** are in R^n, and any active bound results in an over-determined system.

The *CONLES* algorithm uses the step length restricted Newton–Raphson (NR) method for solving constrained NLE. Given the iterate $\mathbf{x}_k$, this method generates the next iterate by solving the linear system (3) for the correction vector $\Delta\mathbf{x}_k$.

$$\mathbf{f}'(\mathbf{x}_k)\Delta\mathbf{x}_k = -\mathrm{f}(\mathbf{x}_k) \tag{3}$$

where $\mathbf{f}'(\mathbf{x}_k)$ is the Jacobian matrix (evaluated at $\mathbf{x}_k$). The next iterate, $\mathbf{x}_{k+1}$ is obtained by:

$$\mathbf{x}_{k+1} = \mathbf{x}_k + \lambda_k\Delta\mathbf{x}_k \tag{4}$$

In the original NR method, $\lambda_k = 1$ is used. Here, its value is set so as to ensure that none of the constraints are violated at $\mathbf{x}_{k+1}$. Let j represent the indexes of all the variables for which constraints have been specified and $\mathbf{x}_{kj} + \Delta\mathbf{x}_{kj} < 0$, then:

$$\lambda_k = \min_j \left|\frac{\alpha x_{kj}}{\Delta x_{kj}}\right| \tag{5}$$

where α is a number smaller than (however close to) one. The constant α is used to ensure that a constrained variable would not attain a value of exactly zero during iterations (*CONLES* uses $\alpha = 0.99$ as a default value).

The NR method may fail to generate the right direction vector if a singular, or nearly singular Jacobian matrix is encountered. Should a singular Jacobian matrix, or excessively large $\Delta\mathbf{x}_k$ encountered (an indication for a near singular matrix) *CONLES* recalculates the correction vector using the Levenberg–Marquardt equation:

$$[\mu_k\mathbf{I} + \mathbf{f}'(\mathbf{x}_k)^{\mathrm{T}}\mathbf{f}'(\mathbf{x}_k)]\Delta\mathbf{x}_k = -\mathbf{f}'(\mathbf{x}_k)^{\mathrm{T}}(\mathbf{x}_k) \tag{6}$$

where **I** is the unity matrix and μ_k is a non-negative parameter. For $\mu_k = 0$, the length and the direction of the correction vector are the same as in the NR method, while for $\mu_k \to \infty$ the 'steepest descent' direction with infinitesimally small step length is obtained. In *CONLES*, the value $\mu_k = \max|\mathbf{f}'(\mathbf{x}_k)|$ is used.

Even the combined use of these two algorithms can not ensure convergence to the solution, unless the initial guess is close enough. If no convergence is obtained from a particular initial guess, or the convergence is to a local minimum, a provision should be provided for generating a better initial guess. *CONLES* uses a 'continuation' type method to generate a sequence of initial guesses. This sequence is generated by the solutions of the following equation system:

$$\mathbf{G}(\mathbf{x}, \theta) = \mathbf{f}(\mathbf{x}) - \theta \mathbf{f}(x_0) = 0, \quad \theta \in [1, 0] \tag{7}$$

where $\mathbf{x}_0$ is the first initial guess. The value of θ is changed from its initial value, $\theta = 1$ (where the solution Eq. (7) is known, $\mathbf{x}_0$) to its final value, $\theta = 0$ (the solution of the original NLE problem) in a sequence $\theta^{k+1} = \theta^k - \Delta\theta$. This is carried out in successive stages, using the solution of one stage as initial guess for the next stage. First the value of $\Delta\theta = 1$ (a single step from $\theta = 1$ to $\theta = 0$) is used. If the combined NR, Levenberg–Marquardt method fails to converge (function values are too large when the correction vector generated is very small) the value of $\Delta\theta$ is reduced and the problem is resolved from $\theta = 1$. This process continues until convergence, or until $\Delta\theta$ reaches a pre-specified minimal value, or the number of iterations reaches a pre-specified maximal value. Thus, the *CONLES* algorithm can find one solution in the feasible region if such a solution exists. Otherwise, if such solution does not exist or it has not been found, a 'no convergence' message is issued.

A detailed description and the FORTRAN code for the *CONLES* algorithm can be found in Shacham (1986). This algorithm is implemented in the POLYMATH 5.0[1] numerical computation package.

Several additional subroutines and programs for solving NLE have been used for the present study. These include the subroutines *mnewt* and *newt* from the 'Numerical Recipes' (Press, Teukolsky, Vetterling & Flannery, 1992) book, the *fsolve* and *fzero* algorithms of MATLAB[2] and the *FindRoot* algorithm of Mathematica[3]. The subroutine *mnewt* implements the classical NR method (Eqs. (3) and (4) with $\lambda_k = 1$). The *newt* algorithm employs the NR method with a line search. Using this method, λ_k is set so as to ensure that $\mathbf{f}(\mathbf{x}_{k+1})^T\mathbf{f}(\mathbf{x}_{k+1}) < \mathbf{f}(\mathbf{x}_k)^T\mathbf{f}(\mathbf{x}_k)$. This method is considered by Press et al. (1992) as a 'globally convergent' method. The *fzero* algorithm finds a zero of a function with one variable. It uses a combination of bisection, secant and inverse quadratic interpolation methods. The *fsolve* algorithm uses several non-linear least-squares techniques to solve system of NLE. It uses the trust region Newton's method for large-scale systems and the Gauss–Newton or Levenberg–Marquardt methods for medium scale problems. Further details on the *fsolve* and *fzero* algorithms can be found in the Optimization Toolbox User Guide (The MATHWORKS Inc., 2000). The *FindRoot* algorithm uses the NR method in cases where analytical expressions for the terms of the Jacobian can be symbolically derived. Otherwise, it uses a multidimensional secant method.

It should be noted that the use of various, widely used NLE solvers is done not for comparison purposes but to demonstrate that all the algorithms that rely on the assumption of function continuity in a predefined subspace of the variables may fail to solve problems that include such discontinuities. 'Robust' NLE solvers such as the ones presented by Bullard and Biegler (1991), Gupta (1995), Wilhelm and Swaney (1994) use algorithms that rely also on function continuity and do not claim to solve problems with discontinuities.

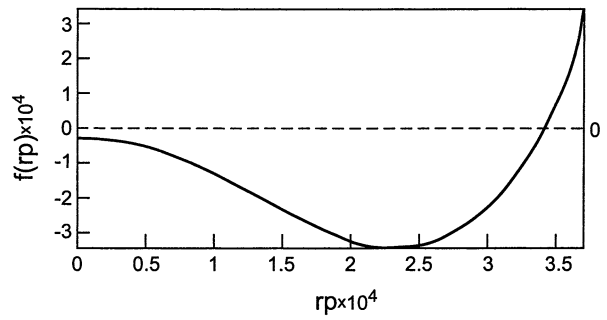

Fig. 1. Plot of $f(r_p)$ vs. r_p in the feasible region (example 1).

3. Motivating examples

3.1. Example 1. Global reaction rate in catalytic oxidation of hydrogen (Vasudevan, 2000)

This example concerns the calculation of the global reaction rate in a packed bed reactor, where catalytic oxidation of hydrogen is carried out. The details of the problem are presented by Smith (1981). Vasudevan (2000) suggested solving the following equation to obtain the global reaction rate r_p:

$$f(r_p) = r_p - 0.327(0.06 - 161 r_p)^{0.804} \times \exp\left\{\frac{5230}{[1.98(373 + 1.84 \times 10^6 r_p)]}\right\} = 0 \tag{8}$$

Physical considerations dictate that $r_p \geq 0$.

This equation cannot be solved with a numerical algorithm, unless the region where the function is defined is first identified. The function is undefined for $(0.06 - 161\ r_p) < 0$ (thus, $r_p > 0.0003727$) and there is a discontinuity for $(373 + 1.84 \times 10^6\ r_p) = 0$ (thus, $r_p = -0.0002027$). In Fig. 1, $\mathbf{f}(r_p)$ is plotted versus r_p in the region of $0 \leq r_p \leq 0.00037$. The root of the equation is at $r_p = 0.000340568862$. Thus, about 3% increase of r_p above the root value puts it in the region where the function is undefined. Within the feasible region,

[1] POLYMATH is copyrighted by M. Shacham, M.B. Cutlip and M. Elly (http://www.polymath-software.com).

[2] MATLAB is a trademark of The Math Works, Inc. (http://www.mathworks.com).

[3] Mathamatica is a trademark of Wolfram Research, Inc. (http://www.wolfram.com).

Table 1
Model equations for the CSTR (Fogler, 2000)—example 2

$f_1 = V - v_o(C_{AO} - C_A)/-r_A = 0$
$f_2 = V - v_o(C_{BO} - C_B)/-r_B = 0$
$f_3 = V - v_o C_C/r_C = 0$
$f_4 = V - v_o\ C_D/r_D = 0$
$f_5 = V - v_o\ C_E/r_E = 0$
$f_6 = 5000(350 - T) - 25(20 + 40)(T - 300) + V(\mathrm{SRH}) = 0$
Where
$r_A = -2k_{1B}C_AC_B$
$r_B = -(k_{1B}C_AC_B + 2k_{2C}C_CC_B^2)$
$r_C = 3k_{1B}C_AC_B - k_{2C}C_CC_B^2$
$r_D = -k_{3E}C_D + k_{2C}C_CC_B^2$
$r_E = k_{3E}C_D$
$k_{1B} = 0.4\exp[(20000/R)(1/300 - 1/T)]$
$k_{2C} = 10\exp[(5000/R)(1/310 - 1/T)]$
$k_{3E} = 10\exp[(10\,000/R)(1/320 - 1/T)]$
$\mathrm{SRH} = 2k_{1B}C_AC_B 20\,000 - 2k_{2C}C_CC_B^2 10\,000 + 5000k_{3E}C_D$
$R = 1.987$, $V = 500$, $v_o = 75/3.3$, $C_{AO} = 25/v_o$, and $C_{BO} = 50/v_o$

$f'(r_p) = 0$ at two locations: at $r_p = 0.0002338$ and at $r_p = 1.974 \times 10^{-6}$.

3.2. Example 2. Modeling of a CSTR for complex sequence of reactions (Fogler, 2000)

Fogler (2000) considered the steady state solution for a CSTR where a complex sequence of reactions: $2A + B \rightarrow 3C$; $C + 2B \rightarrow D$; and $D \rightarrow E$ is carried out. The reactor design equations, including the numerical values of some of the constants are shown in Table 1.

Physical considerations dictate that all the concentrations C_A, C_B, C_C, C_D, and C_E, as well as the temperature T are positive, the reaction rates r_A and r_B are negative, while r_C, r_D and r_E are positive.

All the algorithms mentioned in the Section 2 failed to converge to a solution of this system, irrespective of the initial guess and parameter settings used. Analysis of the system for discontinuities (see Section 4) reveals that it can be made solvable by eliminating the division by unknowns (multiplying f_1 by $(-r_A)$, f_2 by $(-r_B)$ f_3 by r_C, f_4 by r_D and f_5 by r_E). Using this formulation the solution shown in Table 2 was reached (with the *CONLES* algorithm) from the initial guess shown in the same table.

The reason for failure of all the algorithms tested to solve this system using the original problem formulation (Table 1) becomes obvious when one of the functions, f_4 is plotted versus the temperature in the vicinity of the solution (see Fig. 2). It can be seen that the function value changes very sharply over a very small temperature interval. At $T = 372.70$ K the function value is 350, at $T = 372.7646$ K the function value is practically zero and at $T = 372.79128$ K the function value is unbounded as the denominator in f_4 approaches zero. Thus, for a temperature difference smaller than 0.04 K (relative change = 0.01%), the function value goes from zero to infinity.

Regarding the results shown in Table 2, it is worth noting that variable values at the solution are reported

Table 2
Initial guess and solution for example 2

Variable	Initial guess		Solution	
	Value	**f(x)**	Value	**f(x)**
CA (f_1)	0.5	29108.87	2.6663269113340$E-03$	$-1.1724E-13$
CB (f_2)	0.01	14519.86	3.3464055791589$E-02$	$-9.2371E-14$
CC (f_3)	1	43656.84	8.3706595580096$E-01$	$-1.5277E-13$
CD (f_4)	0.0001	−16.9574	3.9669844981400$E-04$	$2.1302E-14$
CE (f_5)	1	−1.5817	8.0853785538223$E-01$	$-3.9080E-14$
T (f_6)	420	5.82E+08	3.7276458623092$E+02$	$-1.9791E-09$
K1B	5824.501		2.7950788123442$E+02$	
K2C	83.80888		3.9225986593124$E+01$	
K3E	422.9115		9.2643973568468$E+01$	
RA	−58.245		−4.9878803322212$E-02$	
RB	−29.1393		−9.8478906554928$E-02$	
RC	87.35913		3.8048452536407$E-02$	
RD	−0.03391		1.8031747719000$E-05$	
RE	0.042291		3.6751720699192$E-02$	
SRH	1164944		4.4593962100198$E+02$	
Vo	22.72727		22.72727273	
CAO	1.1		1.1	
R	1.987		1.987	
V	500		500	
CBO	2.2		2.2	

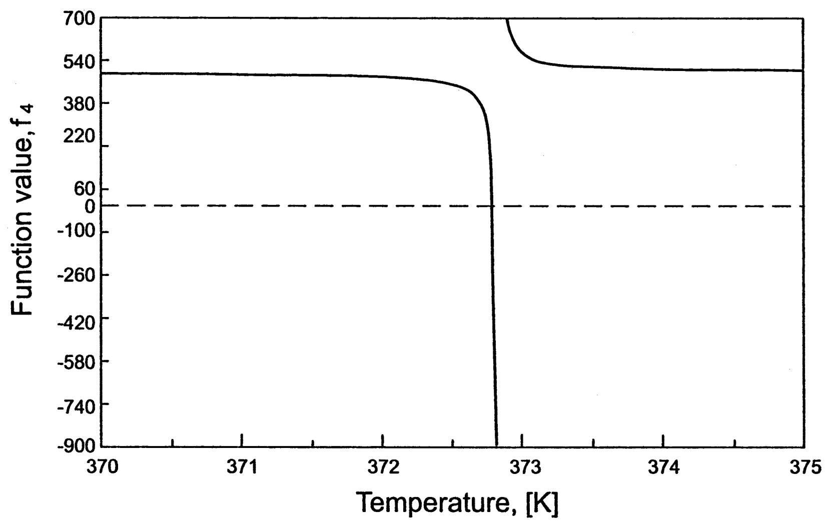

Fig. 2. Plot of f_4 vs. temperature in the vicinity of the solution (example 2).

with the precision used by the numerical solver (15 significant decimal digits for double precision). Function values at the solution are also reported. It was shown by Shacham et al. (2002) that reporting the results rounded to 'engineering' precision can be misleading as a local minimum may be interpreted as solution of the NLE system.

3.3. Example 3. Combustion of propane (Meintjes & Morgan, 1990)

This example concerns the combustion of propane in air according to the following equation:

$$C_3H_8 + \frac{R}{2}(O_2 + 4N_2) \rightarrow \text{products} \tag{9}$$

where R is a parameter expressing the relative amounts of air and propane. There are ten different products, the unknowns n_i ($i = 1-10$) represent the number of moles of product i formed per mole of propane consumed. The model equations are shown in Table 3. Physical considerations dictate that all the n_i are positive.

For $R = 10$ all the algorithms converged to the feasible solution shown in Table 4, from the initial guess shown in the same table. For $R = 5$ using the same initial guess, only *CONLES* converged to the feasible solution shown in Table 5. The *fsolve* converged to a local minimum in the infeasible region (see Table 5) and the rest of the programs stopped with an error message concerning an attempt to calculate a square root of a negative number. Comparing the results in Tables 4 and 5 shows that for $R = 5$, the values of some of the variables are much closer to the feasible region boundary than for $R = 10$ (see the values of n_8, n_9 and n_{10}, for example). Consequently the problem is much more difficult to solve.

4. Eliminating discontinuities from the feasible region

In the previous section it was demonstrated that the solution of NLE with discontinuities often couldnot be found with current state of the art numerical software, irrespective of the initial guess used. Thus, there is a need to modify those systems to make them solvable. The approach we suggest for making such NLE solvable is eliminating the discontinuities by algebraic manipulation of the equations, if possible. If this is impractical, subspaces of the original feasible space are defined, with the discontinuities located on the subspace's boundaries.

To carry out either type of modification, the sub-expressions that may introduce discontinuities have to be identified. This is carried out in a sequential manner, using the following algorithm for discontinuity removal and exclusion of regions where functions are undefined.

1. Select an equation from the set.

Table 3
Model equations for combustion of propane (Meintjes & Morgan, 1990) example 3

$f_1 = n_1 + n_4 - 3 = 0$
$f_2 = 2n_1 + n_2 + n_4 + n_7 + n_8 + n_9 + 2n_{10} - R = 0$
$f_3 = 2n_2 + 2n_5 + n_6 + n_7 - 8 = 0$
$f_4 = 2n_3 + 2n_9 - 4R = 0$
$f_5 = K_5 n_2 n_4 - n_1 n_5 = 0$
$f_6 = K_6 n_1^{1/2} n_4^{1/2} - n_1^{1/2} n_6 (p/n_T)^{1/2} = 0$
$f_7 = K_7 n_1^{1/2} n_2^{1/2} - n_4^{1/2} n_7 (p/n_T)^{1/2} = 0$
$f_8 = K_8 n_1 - n_4 n_8 (p/n_T) = 0$
$f_9 = K_9 n_1^{1/2} n_3^{1/2} - n_4 n_9 (p/n_T)^{1/2} = 0$
$f_{10} = K_{10} n_1^2 - n_4^2 n_{10} (p/n_T) = 0$
where
$n = \Sigma_{i=1}^{10} m$
and
$K_5 = 0.193$, $K_6 = 0.002597$, $K_7 = 0.003448$, $K_8 = 1.799 \times 10^{-5}$, $K_9 = 2.155 \times 10^{-4}$, $K_{10} = 3.846 \times 10^{-5}$ and $p = 40$

Table 4
Initial guess and solution for example 3 ($R = 10$)

Variable	Initial guess		Solution	
	Value	**f(x)**	Value	**f(x)**
N_1 (f_1)	1.5	−1	2.915725423895220	$-3.11E-15$
N_2 (f_2)	2	5.563	3.960942810808880	$-7.11E-15$
N_3 (f_3)	35	−3.855	19.986291646551500	$3.55E-15$
N_4 (f_4)	0.5	30.02	0.084274576104777	$-8.53E-14$
N_5 (f_5)	0.05	0.118	0.022095601769893	$1.94E-15$
N_6 (f_6)	0.005	−0.0032339	0.000722766590884	$3.61E-16$
n_7 (f_7)	0.04	−0.0209598	0.033200408251574	$1.16E-16$
n_8 (f_8)	0.003	−0.0013330	0.000421099693392	$-2.98E-17$
n_9 (f_9)	0.02	−0.0076095	0.027416706896918	$-3.25E-17$
n_{10} (f_{10})	5	−1.1332377	0.031146775227006	$-7.59E-19$
n_T	44.118		27.062238	

Table 5
CONLES and *fsolve* solutions for example 3 ($R = 5$)

Variable	*CONLES* solution		*Fsolve* solution	
	Value	**f(x)**	Value	**f(x)**
n_1 (f_1)	3.56128767073319E-01	0	2.9997	$2.34E-07$
n_2 (f_2)	1.64275227166320E+00	0	3.7881	$-9.75E-08$
n_3 (f_3)	9.99997007578516E+00	0	8.1253	$-9.06E-08$
n_4 (f_4)	2.64387123292668E+00	0	$2.77E-04$	$2.68E-08$
n_5 (f_5)	2.35376244201401E+00	0	$6.73E-05$	$-4.85E-08$
n_6 (f_6)	5.91308317420100E-03	$8.67E-19$	$2.95E-05$	$-3.59E-08$
n_7 (f_7)	1.05748947136700E-03	$-4.34E-19$	0.42362	$-3.65E-05$
n_8 (f_8)	1.03009357100000E-06	0	0.068001	$2.45E-06$
n_9 (f_9)	5.98484296890000E-05	$5.42E-20$	3.7495	$1.24E-04$
n_{10} (f_{10})	2.96634425000000E-07	$8.47E-22$	−4.5145	$3.47E-04$
n_T	17.00351653726560		14.64009	

2. Identify a sub-expression that may introduce discontinuity. The identification can be done by inspection (for small systems) or it can be automated. The proposed technique for automating this algorithm will be discussed in Section 5.
3. Sub-expressions that introduce discontinuity on the boundaries of the feasible region can be left unchanged. In such a case go to 5, otherwise proceed to 4.
4. Remove the discontinuity-causing sub-expression by suitable algebraic manipulation, or replace the sub-expression by an additional variable. The latter is then defined in an additional implicit equation as equal to the sub-expression it represents. A constraint on the new variable must also be added, so that the discontinuity lies on the boundary of the so-defined subspace.
5. Check the equation for an additional discontinuity causing sub-expressions. If any remains, go back to step 3, else proceed to 6.
6. Check for unexamined equations. If any left, go back to 1, else finish.
7. For the numerical solution stage an initial estimate that satisfies all the constraints must be selected.
8. Solve the resultant set of equations. Verify the results by introducing the solution to the original set of equations.

There are various algebraic manipulations and variable transformations that can remove discontinuities for particular types of equation sets (see for example Meintjes & Morgan, 1990). For the general case, however, the multiplication of the equations by denominators that include unknowns has proven to be the most effective approach for alleviating the solution of such systems. In case the discontinuities are handled by splitting the feasible space to subspaces, a solution should be sought in all of the subspaces (the initial guess should be set inside the particular subspace).

To this aim it can be beneficial to employ state of the art constrained homotopy continuation methods (Paloschi, 1995) for locating the roots inside a feasible subspace. Those methods have the advantage of determining all of the roots (or that there are no roots) in the subspace. In some cases physical considerations can be invoked to reduce the number of feasible subspaces.

It should be mentioned that in the case where discontinuity elimination is done by multiplication of the equations by several denominators, it might happen that the multiplication introduces spurious solutions, which are not solutions of the original set. While such a situation is rare (it requires all the denominators to have a root at the exact same point) it will be detected at the solution verification step.

The use of the algorithm for discontinuity removal will be demonstrated with reference to the three examples presented in the previous section.

4.1. Analysis of example 1

In this example there are two terms that introduce discontinuity and/or render the function undefined. The term (0.06–161 r_p) must be ≥ 0 and the zero value of the term $(373 + 1.84 \times 10^6\ r_p)$ should be avoided. Since physical considerations dictate that $r_p \geq 0$ the second term will never be zero in the feasible region, so it does not require any further consideration. The first term may become negative in the feasible region. This difficulty cannot be avoided by an algebraic manipulation. Accordingly a new variable, $x = (0.06–161\ r_p)$ is defined, and the following constrained system of equations is solved:

$$f_1 = r_p - 0.327x^{0.804} \exp\left\{-\frac{5230}{[1.987(373 + 1.84 \times 10^6 r_p)]}\right\} = 0$$

$$f_2 = x - (0.06 - 161r_p) = 0 \qquad (10)$$

subject to the constrains $r_p \geq 0$ and $x \geq 0$.

This formulation enables convergence to the solution using constrained algorithms, such as *CONLES*. For example, with the initial guess: $r_p = 0.1$ and $x = 0.5$ (satisfying the constraints but is very far from the solution, actually lying in the region where the original function is undefined) convergence to the solution in eight iterations is achieved. The unconstrained algorithms used in this study could not find the solution even when using this revised formulation.

4.2. Analysis of example 2

Here f_1 is undefined for $r_A = 0$, f_2 is undefined for $r_B = 0$, f_3 is undefined for $r_C = 0$, f_4 is undefined for $r_D = 0$, f_5 is undefined for $r_E = 0$ and k_{1B}, k_{2C} and k_{3E} are undefined for $T = 0$ (K). This last discontinuity lies on the boundary of the feasible region and does not require any further attention. The discontinuities involving r_A, r_B and r_E lie also on the boundary of the feasible region (as one, or more, of the concentrations involved, C_A, C_B, C_C and C_D, attains a zero value). However, r_C and r_D may become zero inside the feasible region for various combinations of temperature and concentration values.

In this example all the discontinuities (except the one involving $T = 0$) can be removed by algebraic manipulation, namely by multiplying f_1 by $(-r_A)$, f_2 by $(-r_B)$ f_3 by r_C, f_4 by r_D and f_5 by r_E. In addition to removing the discontinuities, such transformations also reduce the high level of non-linearity of the original set of equations in the vicinity of the solution (where some of the reaction rates, r_D in particular, are close to zero). Using this transformation, only the implicit equations of the set (Table 2) have to be modified as follows:

$$f_1 = V(-r_A) - v_o(C_{AO} - C_A) = 0$$

$$f_2 = V(-r_B) - v_o(C_{BO} - C_B) = 0$$

$$f_3 = Vr_C - v_oC_C = 0$$

$$f_4 = Vr_D - v_oC_D = 0$$

$$f_5 = Vr_E - v_oC_E = 0$$

$$f_6 = 5000(350 - T) - 25(20 + 40)(T - 300) + V(\mathrm{SRH}) = 0 \qquad (11a)$$

With this revised formulation, *CONLES* converges to the correct solution from the initial estimate shown in Table 1 in 11 iterations. The subroutine *mnewt* also converged, but the *fsolve* and *FindRoot* algorithms failed to converge from the initial estimate shown. They did converge using an initial estimate for the temperature that is closer to the solution. It should be noted that even in this revised form and starting close to the solution, non-constrained algorithms (such as *fsolve* and *FindRoot*) may converge to an infeasible solution. This may happen since the feasible solution is very close to an infeasible one, where $T = 371.422$, $C_A = -0.0028$ and $C_B = -0.0351$.

The other possible modification of the original formulation is to place the discontinuities caused by zero values of r_C and r_D on the boundary of a feasible subspace. This is accomplished by transforming the respective equations (as shown in Table 2) to implicit equations:

$$f_7 = r_C - (3k_{1B}C_AC_B - k_{2C}C_CC_B^2)$$

$$f_8 = r_D - (3k_{3B}C_D + k_{2C}C_CC_B^2) \qquad (11b)$$

with the following constraints: $C_A > 0$, $C_B > 0$, $C_C > 0$, $C_D > 0$, $C_E > 0$, $r_C > 0$ and $r_D > 0$. Note that the positive values of the concentrations ensure positive values of r_A, r_B and r_E. Thus, there is no need to specify constraints on these variables.

Using this formulation the system is still rather difficult to solve, but it is definitely solvable using a constrained algorithm. *CONLES*, for example, converged to the correct solution within 54 iterations using the following initial guess: $C_A = 0.005$, $C_B = 0.05$, $C_C = 1$, $C_D = 0.01$, $C_E = 1$, $T = 370$, $r_C = 0.04$ and $r_D = 0.00002$.

4.3. Analysis of example 3

In this example, only the square root terms and the division by n_T may cause discontinuity. However, since all the variables are constrained to have positive values, all the discontinuities lie on the boundaries of the feasible region, thus no modification of this equation set is required. Indeed, a constrained solution algorithm (*CONLES*) converged to the solution of this system without any difficulties.

Using the iterative linear programming method Bullard and Biegler (1991) solved a slightly different version of this problem. For $R = 40$ they reported two solutions. Those solutions were later identified by Shacham, Brauner and Cutlip (2002) as local minimum, while *CONLES* did identify the correct solution starting from the same initial estimates.

5. Automation of the discontinuity elimination process

The automation of the discontinuity elimination process can be based on the extension and improvement of the algorithm of Shacham and Shacham (1990). According to this algorithm every equation in the set is converted into a binary tree with an operator (such as '+' or 'function call') in the internal nodes and 'atomic' expressions (variable, or constant names and numerical constants) in the leaves. Different branches of the tree are connected with '+' or '−' operators, while sub-branches are connected with '*', '/' or '^' operators or 'function calls'.

After creating the tree, every branch is examined in turn for the existence of discontinuity causing sub-branches. These sub-branches are then removed or modified in the following steps.

1. Examine the sub-branch and the node it is connected to for the values of the sub-branch that may introduce discontinuity.
2. Determine whether such values can be obtained in the feasible subspace of the unknowns.
3. If both of the above conditions are satisfied, proceed to step 4, otherwise select a different sub-branch and return to 1.
4. If the discontinuity is caused by a sub-branch is positioned in the denominator (preceded by a '/' node) and it can attain a zero value in the feasible subspace, check whether multiplication of the whole equation by this sub-branch removes it from being a denominator. If so multiply the equation by the sub-branch and generate a new equation and a new tree which replace the old ones.
5. In any case where the discontinuity-causing sub-branch cannot be 'multiplied out', a new variable, which replaces the sub-branch, has to be defined. This new variable must be constrained to be non-zero, negative or positive (according to the operation or function call that involves the sub-expression).

The procedure of examination of the branches, addition of new variables to replace sub-branches and addition of constraints continues until it is ensured that none of the branches can become undefined in the feasible subspace.

Let us consider, for example, the following equation (Batch distillation at infinite reflux, Paterson, 1986):

$$f(x) = \frac{1}{63}\ln x + \frac{64}{63}\ln\frac{1}{1-x} + \ln(0.95 - x) - \ln 0.9 \tag{12}$$

This equation is separated into the following four branches:

$$\frac{1}{63}\ln x \tag{12a}$$

$$\frac{64}{63}\ln\frac{1}{1-x} \tag{12b}$$

$$\ln(0.95 - x) \tag{12c}$$

$$\ln 0.9 \tag{12d}$$

Examining the various branches of Eq. (12) yields the following results. In branch Eq. (12a) the sub-branch 'x' is connected to the node with 'function call' 'ln'. Consequently this branch is undefined for $x \le 0$. Thus the constraint $x > 0$ should be added to the equation. The sub-branch: $(1 - x)$ of branch Eq. (12b) is connected to a '/' operation, it can obtain zero value in the feasible region and it cannot be multiplied out. Consequently a new variable $y = 1 - x$ should be defined and it should be constrained: $y <> 0$. After this replacement is made and branch Eq. (12a) is reexamined, it is found that $1/y$ should be positive. This can be achieved by constraining y to be positive only. Branch Eq. (12c) can get undefined in the feasible region if $(0.95 - x) \le 0$. Thus a new variable $z = (0.95 - x)$ has to be defined and it should be constrained to positive values only. Branch Eq. (12d) is always defined. This concludes the analysis of Eq. (12). The resultant set of equations, which is well defined for the entire feasible subspace is the following:

$$f_1(x, y, z) = \frac{1}{63}\ln x + \frac{64}{63}\ln\frac{1}{y} + \ln(z) - \ln 0.9$$
$$f_2(x, y, z) = (1 - x) - y$$
$$f_3(x, y, z) = (0.95 - x) - z \tag{13}$$

with the constraints: $x > 0$; $y > 0$ and $z > 0$.

The example presented involves a single (original) equation, however, this procedure can be carried out similarly for systems of equations. Let us consider example 2, which involves modeling of the CSTR.

Analysis using the algorithm presented above reveals that there are five sub-branches (namely r_A, r_B, r_C, r_D and r_E) connected to '/' nodes which are not allowed to attain a zero value. Since all the concentration values are constrained to positive values, only r_C and r_D can attain such a value in the feasible region and they can be multiplied out. Thus, the automated procedure can be carried out for systems of equations, which involve several variables, and it results in the same equation set that was successfully solved in the previous section.

The differences between the proposed algorithm and the S&S algorithm are worth noting. While the S&S algorithm requires carrying out the analysis in every iteration of the numerical solution, here, the analysis and the derivation of the new set of equations is carried out only once, before the start of the numerical solution. Also, the S&S algorithm cannot deal with expressions where the zero of a sub-branch cannot be calculated analytically. In the herein proposed algorithm, new variables can be introduced and constraints can be set even if the zero of a sub-branch is not exactly known.

6. Conclusions

It has been demonstrated that NLE systems with discontinuities are often very difficult to solve, using state of the art numerical solvers, even when initial guesses close to the solution are used. This happens, in particular, when the discontinuities are located very close to the solution.

If all the discontinuities lie on the boundaries of the feasible space, constrained solution methods, such as the step length restricted NR method, the multidimensional bisection and the iterative linear programming algorithms, can convergence to a solution located within this space. This is because these constrained methods do not require calculation of function and derivative values outside the feasible space.

Discontinuities within the feasible subspace should be identified and either removed by algebraic manipulation of the equations, or put as boundaries of a feasible subspace. This is accomplished by introducing new variables, additional equations and constraints. Following the algorithm presented for carrying the above steps, it has been demonstrated that solutions can be found to originally unsolvable NLE systems by using a constrained (in some cases even unconstrained) NLE numerical solvers.

References

Bhandari, V. M., Yonemoto, T., & Juvekar, V. A. (2000). Investigating the differences in acid separation behavior on weak base ion exchange resins. *Chemical and Engineering Science*, *55*, 6197–6208.

Bullard, L. G., & Biegler, L. T. (1991). Iterative linear programming strategies for constrained simulation. *Computers and Chemial Engineering*, *15*(4), 239–254.

Fogler, H. S. (2000). Personal communications.

Gupta, Y. P. (1995). Bracketing method for on-line solution of low-dimensional nonlinear algebraic equations. *Industrial and Engineering Chemistry Research*, *34*, 536–544.

Hasnat, A., & Roy, S. (1999). Microphase-enhanced reactions: simultaneous effects of ion coupling and counterion binding. *Industrial and Engineering Chemistry Research*, *38*, 4571–4578.

Juvekar, V. A., Anoop, C. V., Puttanayek, S. K., & Naik, V. M. (1999). A continuum model for polymer adsorption at the solid–liquid interface. *Macromolecules*, *32*, 863–873.

Lorenzini, P., Bertrand, P., & Villermaux, J. (1991). Model for ethylene-alpha-olefin copolymerization. *Canadian Journal of Chemical Engineering*, *69*(3), 682–697.

Meintjes, K., & Morgan, A. P. (1990). Chemical equilibrium problems as numerical test problems. *ACM Transactions on Mathematical Software*, *16*(2), 143–151.

Paloschi, J. R. (1995). Bounded homotopies to solve systems of algebraic nonlinear equations. *Computers and Chemical Engineering*, *19*(12), 1243–1254.

Paterson, W. R. (1986). A new method for solving a class of nonlinear equations. *Chemical Engineering and Science*, *41*, 1935.

Press, W. H., Teukolsky, S. A., Vetterling, W. T., & Flannery, B. P. (1992). *Numerical recipes* (2nd ed.). Cambridge: Cambridge University Press.

Sarkar, P., & Gupta, S. K. (1992). Steady state simulation of continuous-flow stirred-tank slurry propylene polymerization reactors. *Polymer Engineering and Science*, *32*(11), 732–742.

Shacham, M. (1986). Numerical solution of constrained nonlinear equations. *International Journal of Numerical Methods in Engineering*, *23*, 1455–1481.

Shacham, O., & Shacham, M. (1990). Finding boundaries of the domain of definition for functions along a one dimensional ray. *ACM Transactions on Mathematical Software*, *16*(3), 258–268.

Shacham, M., Brauner, N., & Cutlip, M. B. (2002). A web-based library for testing performance of numerical software for solving nonlinear algebraic equations. *Computers and Chemical Engineering*, *26*(4-5), 547–554.

Smith, J. M. (1981). *Chemical engineering kinetics*. McGraw-Hill.

Vasudevan, P. T. (2000). Personal communications.

Von Bergen, R., von Bergen, Y., & Rogel, E. (1997). A lattice fluid approach to complex mixtures: natural gas and crude oil. *Fluid Phase Equilibrium*, *137*, 33–52.

Wilhelm, C. E., & Swaney, R. E. (1994). Robust solution of algebraic process modeling equations. *Computers and Chemical Engineering*, *18*(6), 511–531.