algorithms
MDPI

Article

Offset-Assisted Factored Solution of Nonlinear Systems

José M. Ruiz-Oltra, Catalina Gómez-Quiles * and Antonio Gómez-Expósito

Received: 31 October 2015; Accepted: 14 December 2015; Published: 23 December 2015
Academic Editors: Alicia Cordero, Juan R. Torregrosa and Francisco I. Chicharro

Department of Electrical Engineering, University of Seville, Camino de los Descubrimientos s/n, 41092 Sevilla, Spain; E-Mails: josruiolt@alum.us.es (J.M.R.-O.); age@us.es (A.G.-E.)
* Correspondence: catalinagq@us.es; Tel.: +34-954481273; Fax: +34-954487283

Abstract: This paper presents an improvement to the recently-introduced factored method for the solution of nonlinear equations. The basic idea consists of transforming the original system by adding an offset to all unknowns. When searching for real solutions, a real offset prevents the intermediate values of unknowns from becoming complex. Reciprocally, when searching for complex solutions, a complex offset is advisable to allow the iterative process to quickly abandon the real domain. Several examples are used to illustrate the performance of the proposed algorithm, when compared to Newton's method.

Keywords: nonlinear systems; factored solution method; Newton's method

1. Introduction

Numerically finding the solutions of nonlinear equation systems constitutes both a critical and challenging task in many engineering and scientific disciplines [1]. According to John Rice, who coined the term mathematical software in 1969, "solving systems of nonlinear equations is perhaps the most difficult problem in all of numerical computations" [2].

Among the plethora of procedures developed over the last three centuries for the solution of nonlinear systems [3,4], Newton's method stands out as the best general-purpose iterative algorithm, trading off simplicity and reliability, particularly when reasonable initial guesses can be made [5,6].

Even though most existing methods for solving nonlinear equations preserve the structure of the original system throughout the solution process, it has long been known that better convergence rates or wider basins of attractions can be achieved by previously rearranging nonlinear systems. According to [7] (pp. 174–176), "the general idea is that a global nonlinear transformation may create an algebraically equivalent system on which Newton's method does better because the new system is more linear. Unfortunately, there is no general way to apply this idea; its application will be problem-specific".

Recently, the idea of transforming nonlinear systems to enhance the solution process has been systematically exploited by the so-called factored solution approach [8]. It consists of unfolding the nonlinear system by preliminarily identifying all distinct nonlinear terms, each deemed a new variable. This leads to an augmented system, composed of two sets of linear equations along with a one-to-one nonlinear mapping with the explicit inverse. The resulting procedure involves three steps at each iteration: (1) solution of the least-distance linear problem; (2) nonlinear trivial transformation; (3) computation of a Newton-like step.

The factored procedure has been successfully applied to large-scale power systems problems [9], generally showing wider basins of attractions than Newton's method while preserving its quadratic convergence rates near the solution. Moreover, in large-scale engineering applications, the factored method has proven to be capable of reliably converging to complex solutions when real solutions do not exist [10].

In the presence of nonlinear terms comprising products of variables, very commonly found in practice, the factored approach has to resort to the *log* please check the use of italics for the term throughout version of the involved variables, so as to transform the original system into a suitable canonical form [8]. For systems of equations with positive real solutions, this procedure performs satisfactorily. However, when searching for negative or complex solutions, the use of *log* functions sometimes lead to oscillatory or unexpected behavior.

This article proposes a straightforward modification to the factored solution procedure allowing negative real or complex solutions to be safely handled. The idea is to transform the original system by simply adding an offset to those variables involved in log expressions. A large enough positive real offset prevents intermediate values from becoming negative throughout the solution process, significantly enhancing the capability of the factored approach to reach negative real solutions. When needed, convergence to complex solutions can be also assured, or at least enhanced, by adding an imaginary component to the offset. Several benchmark examples are used to illustrate the proposed methodology and compare its performance to that of Newton's method.

2. Review of the Factored Solution Method

The aim of this section is to briefly review the factored solution approach recently introduced in [8]. Consider a general nonlinear system, compactly written as follows:

$$h(x) = p \tag{1}$$

where $p \in \mathbb{R}^n$ is a given parameter vector and $x \in \mathbb{R}^n$ is the unknown vector.

Newton's method approaches a solution from an initial guess, x_0, by successively solving:

$$H_k \Delta x_k = \Delta p_k \tag{2}$$

where subindex k denotes the iteration counter, $\Delta x_k = x_{k+1} - x_k$, H_k is the Jacobian of $h(\cdot)$, computed at the current point x_k, and $\Delta p_k = p - h(x_k)$ is the mismatch vector.

The factored solution method assumes that suitable auxiliary vectors, y and $u \in \mathbb{R}^m$, with $m > n$, can be introduced so that the original Equation system(1) can be unfolded into three simpler problems:

$$\begin{aligned} Ey &= p && (3)\\ u &= f(y) && (4)\\ Cx &= u && (5) \end{aligned}$$

where E and C are rectangular matrices of sizes $n \times m$ and $m \times n$, respectively, and vector $f(\cdot)$ comprises a set of one-to-one nonlinear functions, also known as a diagonal nonlinear mapping [11],

$$u_i = f_i(y_i) \quad ; \quad i = 1, \ldots m \tag{6}$$

each with the closed-form inverse,

$$y_i = f_i^{-1}(u_i) \Rightarrow y = f^{-1}(u) \tag{7}$$

By eliminating vector u the above system can also be written in more compact form:

$$\begin{aligned} Ey &= p && (8)\\ Cx &= f(y) && (9) \end{aligned}$$

It is worth noting that Equation (8) is a linear underdetermined system, while Equation (9) is an overdetermined one. Among the infinite solutions to Equation (8), only those exactly satisfying Equation (9) constitute solutions to the original nonlinear Equation system (1). As explained in [8], many systems can be found in practice to which such a factorization can be applied, either directly or after trivial algebraic manipulations transforming the original system into suitable canonical forms.

The augmented Equation systems (3)–(5) can be efficiently solved as follows (the reader is referred to [8], where the solution procedure below is developed from scratch as an optimization problem):

- Step 0: Given an initial guess, x_0, set $y_k = f^{-1}(Cx_0)$.
- Step 1:
  1.1. Compute λ by solving the linear system,
  $$(EE^T)\lambda = p - Ey_k \tag{10}$$
  1.2. Obtain $\tilde{y}$ from,
  $$\tilde{y} = y_k + E^T\lambda \tag{11}$$
- Step 2: Perform the one-to-one nonlinear transformation, $\tilde{u} = f(\tilde{y})$, and obtain the (trivial) inverse of the diagonal Jacobian matrix, $\tilde{F}^{-1}$.
- Step 3:
  3.1. Obtain x_{k+1} by solving,
  $$\underbrace{E\tilde{F}^{-1}C}_{\tilde{H}}\, x_{k+1} = E\tilde{F}^{-1}\tilde{u} \tag{12}$$
  3.2. Update $y_{k+1} = f^{-1}(Cx_{k+1})$.
  3.3. Convergence check: if $||p - Ey_{k+1}||_\infty < \varepsilon$, then stop; else, return to Step 1.

Step 1 provides a new vector $\tilde{y}$ that, satisfying Equation (8), minimizes the distance to the previous value y_k. The computational cost of this step is very small provided the Cholesky factors of EE^T were saved during the first iteration, and the mismatch vector, $p - Ey_k$, is taken from the previous execution of Step 3.3. It is worth noting that the Jacobian matrix $\tilde{H}$ in Step 3.1 has the same structure as the one involved in Newton's method, Equation (2), but it is computed at a different point.

Tests performed so far with the factored method on large realistic problems [9] have shown that each iteration of the above factored procedure is generally faster than that of Newton's method. Moreover, as shown in [8], the factored method exhibits a quadratic convergence rate in the neighborhood of a solution and wider basins of attractions. An outstanding additional feature of the factored method is that, unlike Newton's method, it can converge to complex solutions when the initial point is not in the basin of attraction of a real solution or simply when real solutions do not exist at all [10].

3. Limitations of the Factored Method When Real Negative or Complex Solutions Are Involved

When the nonlinear system to be solved contains terms of the form $x_i^a x_j^b$, the application of the factored method requires that the original set of variables be replaced by its *log* counterpart [8],

$$\alpha_i = \ln x_i \qquad i = 1, \ldots, n$$

Then, each distinct term $x_i^a x_j^b$ gives rise to a new auxiliary variable y_{ij},

$$y_{ij} = x_i^a x_j^b = \exp\left(a\alpha_i + b\alpha_j\right)$$

In this particular case, the corresponding component of the nonlinear relationship $u = f(y)$ reduces to,

$$u_{ij} = \ln y_{ij} \quad \Rightarrow \quad y_{ij} = \exp u_{ij}$$

which leads to a linear relationship, $u = C\alpha$, as required by the last step,

$$u_{ij} = a\alpha_i + b\alpha_j$$

In those cases in which all solutions of the nonlinear system in hand are real and positive, the use of the *log* function does not usually pose any problem. However, in the presence of negative or complex solutions, it has been found experimentally that the performance of the factored method is not always as good, leading in many cases to oscillatory or erratic behavior. In what follows, the nature of this drawback is analyzed, and ways of circumventing it when searching for negative and complex solutions are separately discussed.

3.1. Negative Real Solutions

Let us assume that, as usually happens in engineering and science, one is mostly interested in finding the real solutions of nonlinear systems. Typically, for a particular application, it is known in advance whether the solutions will be positive or negative, according to the nature of the involved variables (e.g., in power engineering, voltage and current magnitudes are positive, by definition). As stated above, when those real solutions are positive, the factored method performs well. However, in the presence of negative solutions, some components of the auxiliary vector y will eventually become negative throughout the iterative process, driving the respective components of vector u to the complex domain (more specifically, to be of the form $u_k = \ln|y_k| + \pi i$). In turn, when solving Equation (12) to obtain the *log* variables α (linear combinations of u variables), it may happen that one or several components α_k become also complex, with an angle other than π, which means that the original variable x_k would be complex, rather than negative real. In those cases, the iterative process may remain oscillating or be diverted away from the basin of attraction of the intended real solution, eventually reaching one of the unsought complex solutions.

This potential risk is easily illustrated with the help of the following simple example.

example 1. Consider the solution of the system,

$$\begin{aligned} p_1 &= x_1x_2 + x_2 \\ p_2 &= x_2^2 + 2x_1 \end{aligned} \tag{13}$$

For $p_1 = -10$ and $p_2 = 19$, there are two real solutions, $x = [3, -5]$ and $x = [9, -1]$, both of them containing a negative component. The two constant matrices involved in the factored procedure are in this case:

$$E = \begin{pmatrix} 0 & 1 & 1 & 0 \\ 2 & 0 & 0 & 1 \end{pmatrix} \quad ; \quad C = \begin{pmatrix} 1 & 0 \\ 0 & 1 \\ 1 & 1 \\ 0 & 2 \end{pmatrix}$$

and the nonlinear transform, $u = f(y)$,

$$u = \ln \begin{pmatrix} y_1 \\ y_2 \\ y_3 \\ y_4 \end{pmatrix} \quad \text{with} \quad y = \begin{pmatrix} x_1 \\ x_2 \\ x_1x_2 \\ x_2^2 \end{pmatrix}$$

Starting from $x^{(0)} = [0, 0]$, the first iteration provides the following results:

$$y^{(1)} = \begin{pmatrix} 7.6 \\ -5 \\ -5 \\ 3.8 \end{pmatrix} \quad ; \quad u^{(1)} = \begin{pmatrix} 2.0281 \\ 1.6094 + \pi i \\ 1.6094 + \pi i \\ 1.3350 \end{pmatrix} \quad ; \quad x^{(1)} = \begin{pmatrix} -3.9872 - 6.9061i \\ -0.8853 - 1.5334i \end{pmatrix}$$

As is clearly seen, the two negative components of y translate into complex components in the solution x, which are obtained as the exponential of linear combinations of u components. These complex components of x propagate through the iterative process until a steady-state cyclic pattern settles down or a complex solution is eventually reached (it is very unlikely that vector x becomes again real during the transient period).

The simplest way of preventing this risk (*i.e.*, the factored method being unable to smoothly converge to negative real solutions) is by adding a sufficiently large positive offset, m, to all variables in x, or at least to those components in x that are clear candidates to become negative, so that the resulting transformed problem has only positive real solutions. This idea is applied below to the previous example.

example 2. Introducing new variables, $x_{o1} = x_1 + m$ and $x_{o2} = x_2 + m$, the original Equation system (13) becomes:

$$\begin{aligned} p_1 - m^2 + m &= x_{o1}x_{o2} - mx_{o1} + (1-m)x_{o2} \\ p_2 - m^2 + 2m &= x_{o2}^2 + 2x_{o1} - 2mx_{o2} \end{aligned} \tag{14}$$

and the new matrix E,

$$E = \begin{pmatrix} -m & 1-m & 1 & 0 \\ 2 & -2m & 0 & 1 \end{pmatrix}$$

Starting again from $x^{(0)} = [0, 0]$, the first iteration provides this time the following results:

$$y^{(1)} = \begin{pmatrix} 7.1429 \\ 0.0476 \\ 2.3333 \\ 4.9048 \end{pmatrix} \quad ; \quad u^{(1)} = \begin{pmatrix} 1.9661 \\ -3.0445 \\ 0.8473 \\ 1.5902 \end{pmatrix} \quad ; \quad x^{(1)} = \begin{pmatrix} 7.5497 \\ -0.4475 \end{pmatrix}$$

After just four iterations, the process converges to,

$$x_o^{(4)} = \begin{pmatrix} 11.0000 \\ 1.0000 \end{pmatrix} \quad ; \quad x^{(4)} = \begin{pmatrix} 9.0000 \\ -1.0000 \end{pmatrix}$$

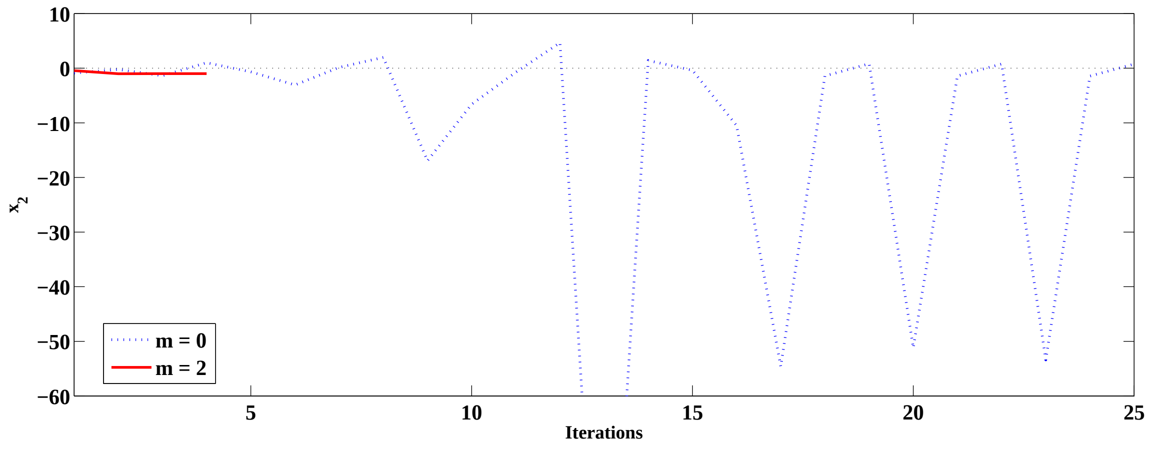

Figure 1. Evolution of x_2 as iterations progress for the simple system of Examples 1 ($m = 0$) and 2 ($m = 2$).

Figure 1 represents the evolution of x_2 along the iterative process for $m = 0$ (real part only) and $m = 2$. In the first case ('vanilla' factored solution), the iterative process oscillates in the complex domain after nearly 20 iterations, whereas it converges in just four iterations to the correct real solution with $m = 2$.

3.2. Complex Solutions

If the starting point does not lie in the basin of attraction of a real solution, or simply when real solutions do not exist at all, the factored iterative procedure is able to converge to a complex solution. However, as it is never known in advance whether the starting point is in the basin of a positive, negative or complex solution, it is advisable to always add a sufficiently large positive offset m, as discussed in the previous section. The problem of adding such a positive offset is that, when starting from a real starting point, the iterative process will not abandon the real domain unless a negative component arises by chance in y, which may not happen at all. This is a consequence of the log of a positive real number being also real. Fortunately, this problem is easily solved by adding an imaginary component to the offset m, as illustrated in the following example.

example 3. Consider again the simple system of the previous examples, but this time with $p_1 = 2$ and $p_2 = 0$, which has the real solution $x = [-2, -2]$ and the complex conjugate solution $x = [\pm i, 1 \pm i]$. Starting from $x^{(0)} = [0, 0]$, with $m = 2$, the values of x_1 and x_2 show initially a sustained oscillatory behavior (see Figure 2) until x_1 becomes negative at Iteration 9. Afterwards, the imaginary component of x_1 is non-null, and the iterative process manages to converge after 20 iterations to one of the complex conjugate solutions. For other starting points, the oscillations may persist much longer or indefinitely. However, with $m = 2 + i$, the factored method quickly converges in six iterations, as shown in Figure 2.

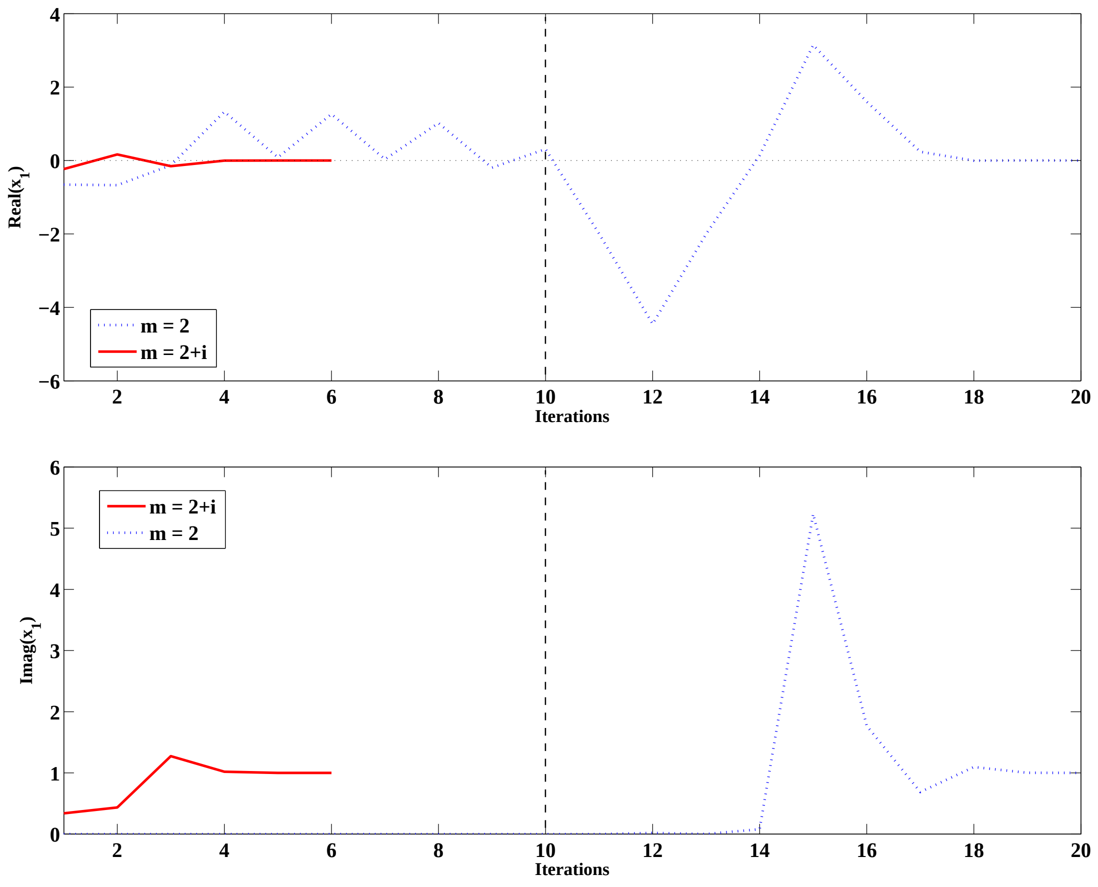

Figure 2. Evolution of real (x_1) and imag (x_1) as iterations progress for the simple system of Example 3 with $m = 2$ and $m = 2 + i$.

In summary, a positive real offset m is needed to prevent the iterative process from converging to complex solutions when searching for real negative solutions. Reciprocally, an imaginary component should be added to m so that complex solutions more easily pop up when real solutions do not exist or are of no interest.

4. Generalized Offset-Assisted Factored Solution

As discussed above, the idea is to prevent the appearance of negative variables by simply adding a sufficiently large scalar m to all components of the unknown vector, or at least those which are suspected to eventually become negative throughout the solution process, as follows:

$$x_o = x + m \begin{bmatrix} 1 \\ 1 \\ \vdots \\ 1 \end{bmatrix} \tag{15}$$

Then, the original Equation system (1) is expressed in terms of the new set of positive variables,

$$h_o(x_o) = p_o \tag{16}$$

In turn, the resulting system can be factored in the same way as the original one, leading to the transformed system:

$$\begin{aligned} E_o y &= p_o && (17)\\ u &= f_o(y) && (18)\\ C x_o &= u && (19) \end{aligned}$$

which can be solved by applying the three-step procedure reviewed in Section 2 without facing the risk of negative variables arising.

Finally, the original variables x can be easily recovered by removing the offset m.

5. Case Studies

Two case studies will be used to compare the performance of the offset-assisted factored method with that of the standard Newton's method.

5.1. Fourth-Order Polynomial System

Consider the following 2×2 nonlinear system,

$$\begin{aligned} p_1 &= x_1 x_2^2 \\ p_2 &= 2x_1^2 x_2 - x_1^2 \end{aligned} \tag{20}$$

which is equivalent to two fourth-order polynomials in x_1 and x_2. For $p_1 = -4$ and $p_2 = 4$, it is satisfied by the solutions shown in Table 1, two of them real with a negative component.

Table 1. Solutions of the 4th-order system for $p_1 = -4$ and $p_2 = 4$.

x_1	x_2	Color
-2	1	🟦
-3.8581	0.6344	🟦
$0.2624 \pm 0.7889i$	$-1.8172 \pm 1.733i$	🟨
-	-	

Figure 3 shows the basins of attraction (upper) and the number of iterations (lower) corresponding to Newton's method. The colors corresponding to each solution are in accordance with those of Table 1, where the white color means that no convergence is achieved.

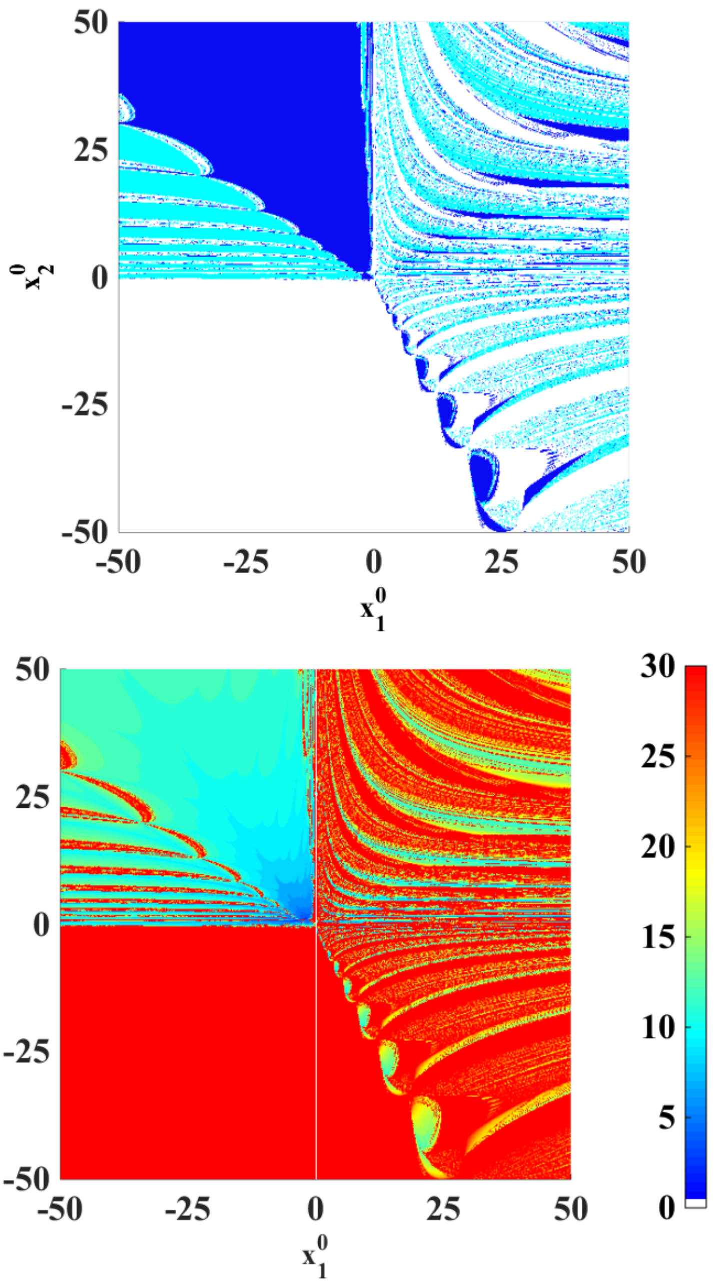

Figure 3. Basins of attraction (**upper**) and number of iterations (**lower**) for the fourth-order system when Newton's method is applied.

As can be noticed by the white color in the upper diagram (and the red color in the right one), when both components in x_0 are negative, convergence is never achieved. Moreover, many starting points in the first and fourth quadrant also lead to divergence. As expected, Newton's method is not capable of reaching complex solutions (yellow color missing in the upper diagram).

Figure 4 is the counterpart of Figure 3 for the factored method with a real offset ($m = 10$). As can be seen, in the neighborhood of the two real solutions (central area in the diagrams), the behavior of the factored method is similar to that of Newton's method. However, the total surface of the white areas in the upper diagram (non-convergent cases) is much smaller than that of Figure 3, indicating lower risk of divergence when using the factored method.

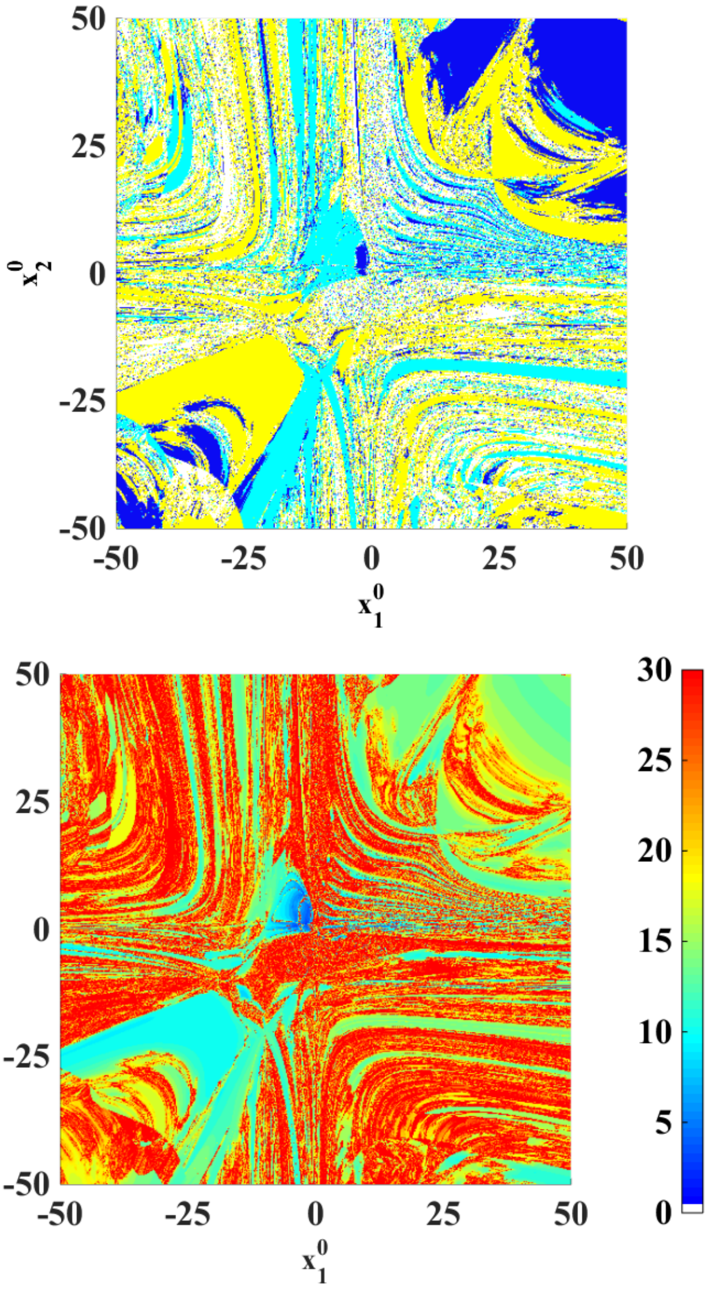

Figure 4. Basins of attraction (**upper**) and number of iterations (**lower**) for the fourth-order system when the factored method with $m = 10$ is applied.

Finally, the results obtained with the factored method with $m = 10 + 5i$ are shown in Figure 5. Compared to Figure 4, it is clear that white areas in the upper diagram have virtually faded away (no risk of divergence at all), while convergence to real solutions in their neighborhood (central area in light and dark blue) is preserved. Large areas in yellow are also apparent in the upper diagram, indicating that the addition of an imaginary component to m clearly favors the trend to reach complex solutions.

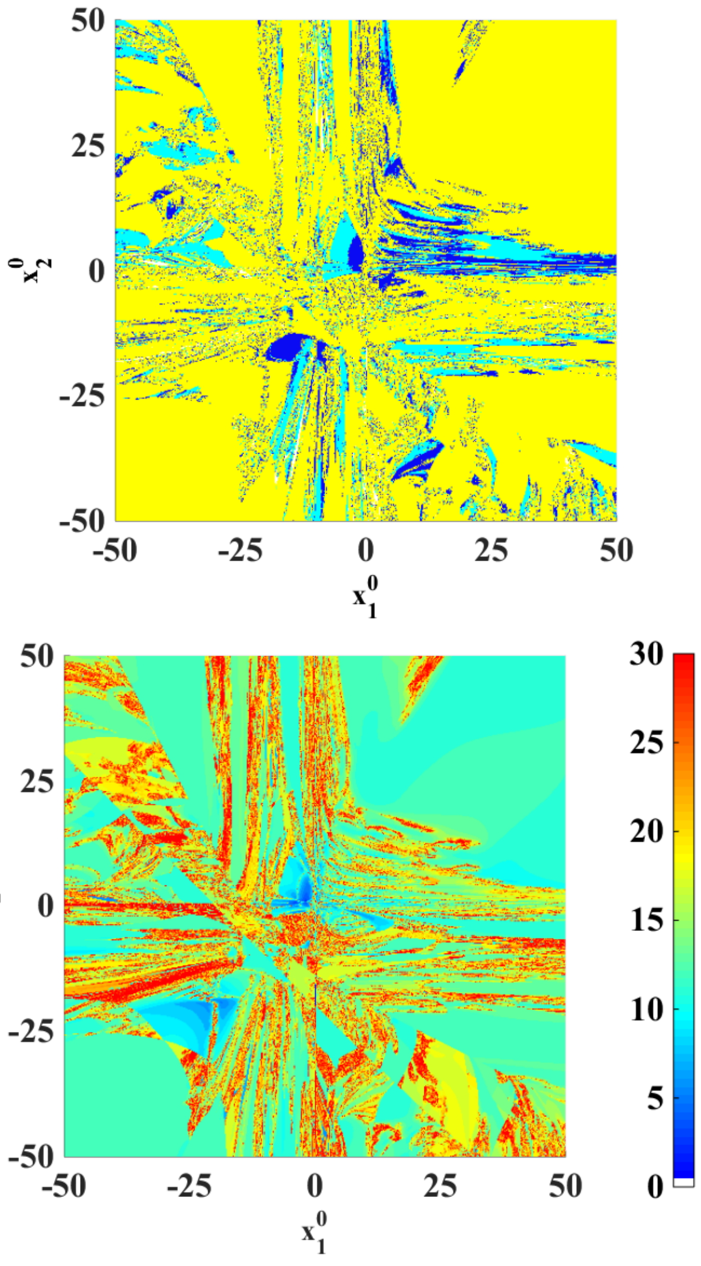

Figure 5. Basins of attraction (**upper**) and number of iterations (**lower**) for the fourth-order system when the factored method with $m = 10 + 5i$ is applied.

5.2. Kelley's Synthetic System

The next system studied, originally posed by Kelley [12], is as follows:

$$\begin{aligned} 2 &= x_1^2 + x_2^2 \\ 2 &= e^{x_1 - 1} + x_2^2 \end{aligned} \tag{21}$$

This system is symmetric under the transformation $x_2 \to -x_2$ and has the solutions provided in Table 2.

Figure 6 shows the basins of attraction (upper) and number of iterations (lower) corresponding to Newton's method. The colors corresponding to each solution are in accordance with those of Table 2, where the white color means that no convergence is achieved. In this apparently simple case, the basins of attraction are perfect rectangles, and convergence is not possible in most of the right half plane. Since the Jacobian is singular along the line $x_2 = 0$, Newton's method gets in trouble also in that region.

Table 2. Solutions for Kelley's system.

x_1	x_2	**Color**
1	-1	🟦
1	1	🟦
-0.4777	-1.3311	🟩
-0.4777	1.3311	🟩
3.5129	$-3.2156i$	🟨
-	-	

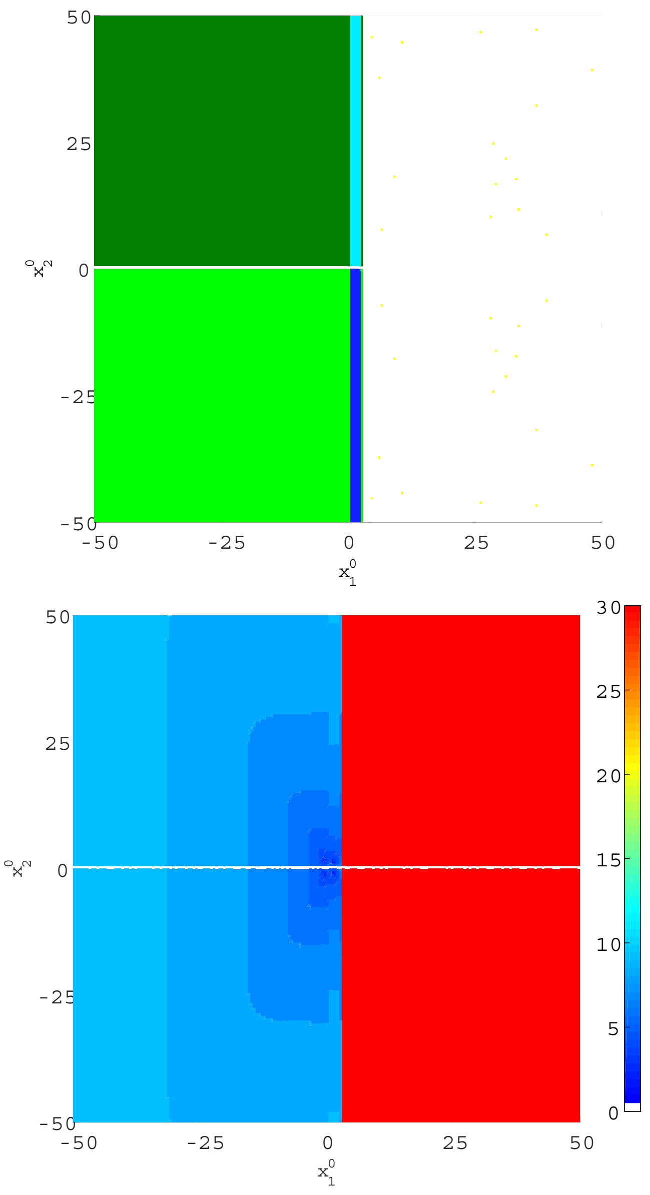

Figure 6. Basins of attraction (**upper**) and number of iterations (**lower**) for Kelley's system when Newton's method is applied.

Figure 7 is the counterpart of Figure 6 for the factored method with a real offset ($m = 2$). As can be seen, the white (red) color in the upper (lower) diagram has vanished, which means that convergence is achieved nearly globally. The symmetry around x_2 is still apparent.

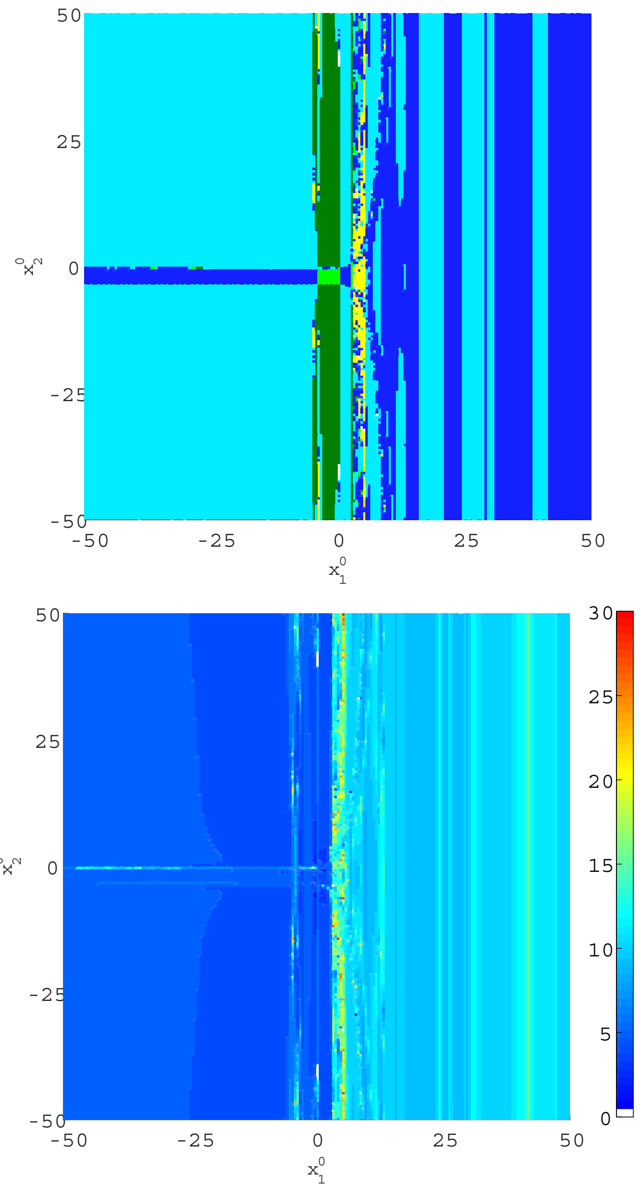

Figure 7. Basins of attraction (**upper**) and number of iterations (**lower**) for Kelley's system when the factored method with $m = 2$ is applied.

Finally, the results obtained with the factored method with $m = 10 + 0.5i$ are shown in Figure 8. Compared to Figure 7, a larger and more homogenous yellow area is visible in the central part of the left diagram, indicating the trend for the factored method to reach complex solutions when m has an imaginary component, as explained in Section 3.2.

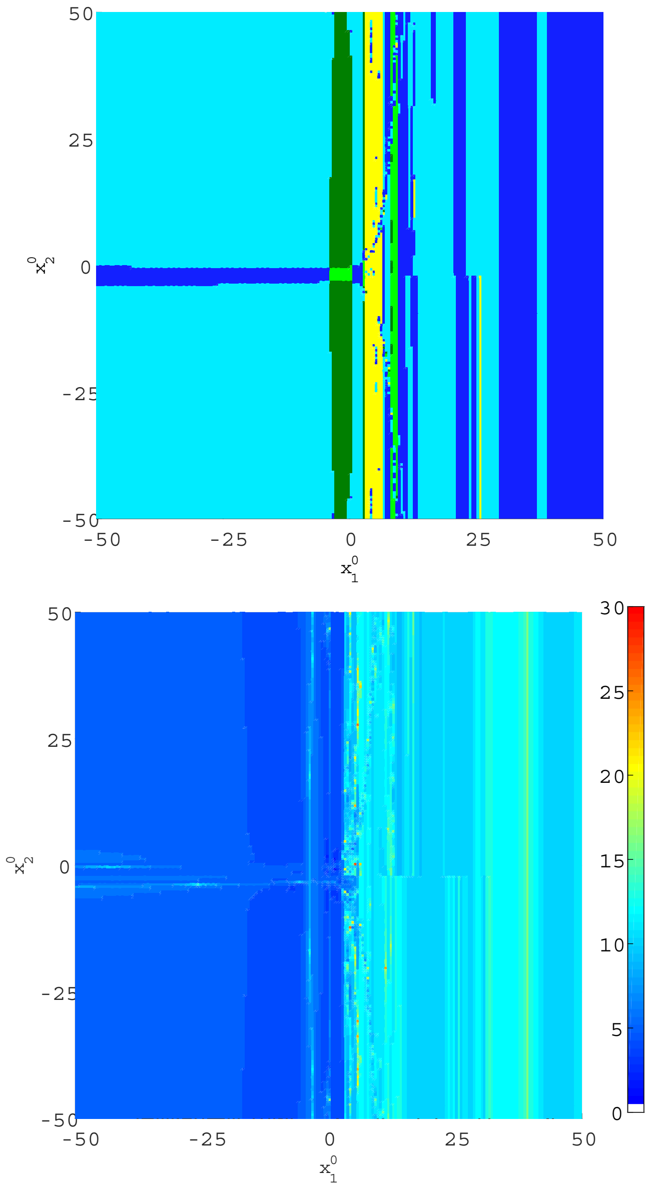

Figure 8. Basins of attraction (**upper**) and number of iterations (**lower**) for Kelley's system when the factored method with $m = 2 + 0.5i$ is applied.

6. Conclusions

This paper has addressed the difficulties faced by the factored solution method when solving nonlinear systems with negative or complex solutions. It has been shown that by adding a sufficiently large real offset, both positive and negative real solutions can be safely reached. Moreover, if no real solutions exist or the user is rather interested in finding complex solutions, it is advisable to add an imaginary component to the offset. Some examples have been worked out showing that, while locally preserving the quadratic convergence rate of Newton's method, the global behavior (basins of attractions) of the factored procedure can be tuned to a large extent with the help of the proposed offset, increasing the chances for real or complex solutions to be selectively reached.

Acknowledgments: This work has been financially supported by the Spanish government (Ministry of Economy and Competitiveness) under Grant ENE2013-48428-C2.

Author Contributions: J.M. Ruiz-Oltra coded several versions of the proposed algorithm when applied to the load flow in rectangular coordinates, prepared the large-scale test cases, obtained numerical and graphical results

and wrote parts of the paper. C. Gomez-Quiles discovered the problem arising with the factored solution method when solving cases with negative variables, helped the student develop an efficient MATLAB code, designed the experiments and wrote parts of the paper. A. Gomez-Exposito conceived of the idea of adding an offset to solve the problem and wrote parts of the paper.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Kuo, M. Solution of nonlinear equations. *IEEE Trans. Comput.* **1968**, *17*, 897–898.
2. Rice, J. *Numerical Methods, Software, and Analysis*; Academic Press: New York, NY, USA, 1993.
3. Ortega, J.M.; Rheinboldt, W.C. *Iterative Solution of Nonlinear Equations in Several Variables*; Academic Press: New York, NY, USA, 1970 (also published by SIAM, Philadelphia, 2000).
4. Kelley, C.T. *Iterative Methods for Linear and Nonlinear Equations*; SIAM: Philadelphia, PA, USA, 1995.
5. Deuflhard, P. *Newton Methods for Nonlinear Problems*; Springer-Verlag: Heidelberg, Germany, 2004.
6. Kelley, C.T. *Solving Nonlinear Equations with Newton's Method, Fundamentals of Algorithms*; SIAM: Philadelphia, PA, USA, 2003.
7. Judd, K.L. *Numerical Methods in Economics*; MIT Press: Boston, MA, USA, 1998.
8. Gómez-Expósito, A. Factored Solution of Nonlinear Equation Systems. *Proc. R. Soc. A* **2014**, doi:10.1098/rspa.2014.0236.
9. Gómez-Expósito, A.; Gómez-Quiles, C. Factorized Load Flow. *IEEE Trans. Power Syst.* **2013**, *28*, 4607–4614.
10. Gómez-Expósito, A.; Gómez-Quiles, C.; Vargas, W. Factored Solution of Infeasible Load Flow Cases. In Proceedings of the Power Systems Computation Conference (PSCC), Wroclaw, Poland, 18–22 August 2014.
11. Sandberg, I.W.; Willson, A.N. Existence and Uniqueness of Solutions for the Equations of Nonlinear DC Networks. *SIAM J. Appl. Math.* **1972**, *22*, 173–186.
12. Rheinboldt, W. Some Nonlinear Test Problems. Available online: http://folk.uib.no/ssu029/Pdf_file/Testproblems/testprobRheinboldt03.pdf (accessed on 22 December 2015).