Insertion of Parametric Anchors to Facilitate Newton-Raphson Iterations near Pseudo Roots

Xiaojian Lin[a], Tienmo Shih[b,*]

[a]*Department of Computer Science, The University of Hong Kong, 999077, Hong Kong SAR, China*
[b]*Department of Mechanical Engineering, University of Maryland, College Park, MD 20742, USA*

Abstract

In a nonlinear system, when both function values $f_i(x_1, x_2, ...x_i,...,x_n)$ and corresponding Jacobian determinants approach zeros, their ratios may remain finite. Under this circumstance, iterations of the traditional Newton-Raphson method (NRM) tend to wander around local extrema, resulting in non-convergence (not necessarily divergence). Herein, we propose to parametrically modify the given nonlinear system primarily based on the concept of matrix diagonal dominance. In addition to faithfully following the linearization formula of first-order Taylor Series Expansion adopted by NRM, we manage to guide iterations to travel along diminishing-parameter paths that are established by roots of these modified systems. When the parametrized system eventually reverts to the original one, iterated solutions have already passed extrema and approached the desired root. Using four examples governed by scientific and engineering laws, we illustrate the strategy of the proposed algorithm and, in passing, introduce the benefit of finding complex roots. Hopefully, the proposed study will serve as a reference for the community that are interested in using NRM to solve scientific and engineering nonlinear systems.

Keyword: Parametric anchor (PA); Pseudo root (PR); Diagonal dominance; Complex root (CR); Newton-Raphson method (NRM); Local extrema;

1. Introduction

Other than the Newton-Raphson method (NRM), in which the linearization is performed according to first-order Taylor Series Expansion, numerous innovative techniques have also been developed over the past few decades. For example, domain decomposition enhances computational efficiency [1]; deep learning combines with orthogonal decomposition for

* Corresponding author. Tel.: +86-180-4616-3831; *E-mail address:* tienmoshih@gmail.com.

complicated PDEs [2]; radial basis networks provide advanced solutions for nonlinear PDEs [3]; dimensionality reduction facilitates probability density functions [4]; iterative methods bypass complex derivative calculations [5]; and intelligent algorithms excel in root identification [6]. Innovatively, these methods solve nonlinear systems without relying on the traditional Taylor series expansion.

Simultaneously, the NRM has been applied to analyses of numerical properties, such as accuracy, stability, robustness, and convergence. These studies include overcoming challenges in finding analytical solutions, applications in eigenvalue problems [7], rapid derivative-based convergence [8], dependence on the initial guess [9], broader convergence theory [10], and sensitivity to computational errors [11].

Progressively, various NRM-improved versions have been developed and applied to solving engineering problems. Convergence to exact solutions has been improved by Pho Kim-Hung using modified decomposition methods [12]. Tajima and Yamada tailored topology optimization to manufacturing constraints [13]. Machine learning for optimal quadrature in computational analysis has been harnessed by Teijeiro et al. [14]. Optimization algorithms for photovoltaic models are enhanced by Mohammed et al., boosting the LSHADE algorithm [15]. Kumar et al. have combined optimization algorithms with NRM for better parameter estimation in solar models [16]. Efficient methods for large deformations are implemented by Mohammed et al., enhancing finite element simulations [17]. Near-zero error rates under varying statistical conditions are achieved by Wang et al. with a modified NRM [18]. Finally, Salvador and Marsden's research indicates that, for polynomial problems, different iterative methods converge with varying rates of error, highlighting the progress in iterations related to NRM [19].

Herein, while focusing our efforts on solving scientific and engineering problems governed by both nonlinear algebraic equations and nonlinear partial differential equations, we propose an algorithm in which each nonlinear algebraic equation is inserted with a parametric anchor (PA). This anchoring effect tends to smooth curvatures of nonlinear equations and to allow iterated Jacobian matrix determinants to travel along an iterative route (different from the NRM iterative route), significantly reducing sign changes of f/f' values near extrema.

2. Examples

Totally, we have selected four examples to demonstrate merits of the proposed algorithm. When fabricating the first, second, and third examples, we ensure that at least one real-number root exists for the nonlinear system.

2.1. Governing equations of examples

In the first example, a seventh-degree one-variable polynomial equation,

$$f(x) = c_0 + c_1x + c_2x^2 + c_3x^3 + c_4x^4 + c_5x^5 + c_6x^6 + c_7x^7 = 0, \quad (1)$$

satisfies the condition, under which a root ($x = -2$) is flanked by two local extrema (see values of coefficients in Appendix A).

The second example is involved with chemical physics, and the Arrhenius equation related to the Boltzmann Distribution law is given as $k = Ae^{(-E/RT)}$. Here, k denotes the chemical-reaction rate; A the pre-exponential factor; E the activation energy; R the universal gas constant; and T the absolute temperature. A nonlinear system may be represented as

$$\mathrm{f}_1(x,y) = zt + c_0 + c_1x + c_2x^5y^6 + c_3x^3 + c_4e^{c_5t} = 0, \quad (2a)$$

and

$$\mathrm{f}_2(x,y) = -zt + d_0 + d_1y + d_2x^7y^4 + d_3y^3 + d_4e^{d_5z} = 0. \quad (2b)$$

For pre-conditioning, additional equations are introduced to ensure numerical stability by normalizing certain terms, specifically $\mathrm{f}_3(x,y) = xz - 1 = 0$ and $\mathrm{f}_4(x,y) = yt - 1 = 0$, where z and t serve as reciprocal terms of x and y, respectively. The known root is $x = 10$, $y = 10$ and $x = 10$, $y = -10$. For details regarding the linearized counterpart of Equations (2a & 2b), readers are encouraged to refer to Appendix B, where the corresponding coefficient matrices and elements are provided. Instead of using the negative value in the exponential term, which must lie in the range of $[exp\,(-\infty), exp\,(-0)]$ or $[0, 1]$, we are willing to face the challenge of a much wider range of $[1,\infty]$ or $[exp(+0), exp(large\ positive\ number)]$.

In the third example, Generative Adversarial Networks (GANs) consist of two adversarial neural networks, namely Generator G and a Discriminator, that compete against each other through a sophisticated nonlinear loss function. Also, being inherently nonlinear, the training process of GANs involves intricate interactions among parameters of G, D, latent variables, and iterative steps in the training process. Here, the dynamic interactions during GAN training can be represented by

$$f_1(x, y, z, t) = 22/x + 7/y + y + 3.5/z + 11/t + 7 = 0, \quad (3a)$$

$$f_2(x, y, z, t) = 0.1x^4 + 0.1e^{x/t} - 0.1e^{y/z} - 1.6t^4 = 0, \quad (3b)$$

$$f_3(x, y, z, t) = xy - 4zt + y^5 + 32z^5 = 0, \quad (3c)$$

$$f_4(x,y,z,t) = xz + yt + yy + 2yx/t + e^{-0.1yt} - 2131.3 = 0, \quad (3d)$$

where x denotes the parameter vector of the Generator, which includes weights and biases used to transform latent noise into synthetic data; y the parameter vector of the Discriminator,

which is optimized to distinguish between real and generated data; z the latent noise vector sampled from a prior distribution (e.g., Gaussian or uniform), which serves as input to the Generator; and t the iteration step in the training process, indicating the progression of parameter updates. The exponential decay term $e^{-0.1yt}$ indicates the diminishing influence of the Discriminator over time, whereas nonlinear interaction terms, which represent the evolving feedback between $G\ and\ D$ as training continues, gradually stabilize the system. The fabricated root is $x = -22, y = -7, z = 3.5$, and $t = 11$. See Appendix C for linearized coefficient-matrix elements, real roots, complex roots (CRs).

In the fourth example, the energy conservation of one-dimensional transient heat conduction is governed by

$$\frac{\partial T}{\partial t} = \frac{\partial}{\partial x}\left(\alpha(T)\frac{\partial T}{\partial x}\right). \tag{4}$$

Because the thermal diffusivity depends on the temperature, discretized equations become a nonlinear system. Subject to Dirichlet boundary conditions at $x = 0$ and Neumann boundary conditions at $x = L$, both corresponding discretized nonlinear equations ($\Delta x = 0.1L,\ x \in L$) and the coefficient matrix of linearized equations are given in Appendix D.

2.2. Purposes of choosing these examples

The first example (one-variable polynomial) is chosen to graphically describe the strategy and the procedure of PA algorithm, which constitutes the crux of this study. Next, in the second example (4-equation set), we examine occurrences of different pairs of positive and negative values of PAs (Table 1). Then, the third example (10-equation set) is selected to demonstrate the capability of PA when multi-variable high-nonlinearity problems are encountered. Due to the complexity of the problem, all four cases of convergence difficulties (shown in Fig. 1, to be further qualitatively explained in the next section) are encountered by NRM. Incidentally, this example is also related to the benefit of finding CRs. Finally, the 4th example (also 10 algebraic-equation set) is introduced to demonstrate that PA is also capable of handling discretized nonlinear partial differential equations.

3. Proposed parametric anchor (PA) algorithm

Generally, NRM iterations may encounter non-convergence difficulties when they approach local extrema. To face this challenge, we must first learn how to precisely classify and identify these difficulties. For clear illustrations, let us graphically examine simple cases of finding roots in a high-degree one-variable polynomial, namely $y = f(x)$ and $y = 0$ (i.e. x axis). In Fig. 1. (a), although a minimum is located closely to the x axis, the curve and the axis do not intersect. Herein, let us tentatively name this nonexistent root as pseudo root (PR). By contrast, in Fig. 1. (b), the curve and the axis are separated by a finite distance. In Fig. 1.

(c), they intersect at only one single point, suggesting that both the function and its derivative equal zero at this point. Finally, in Fig. 1. (d), the curve and the axis are almost parallel to each other locally.

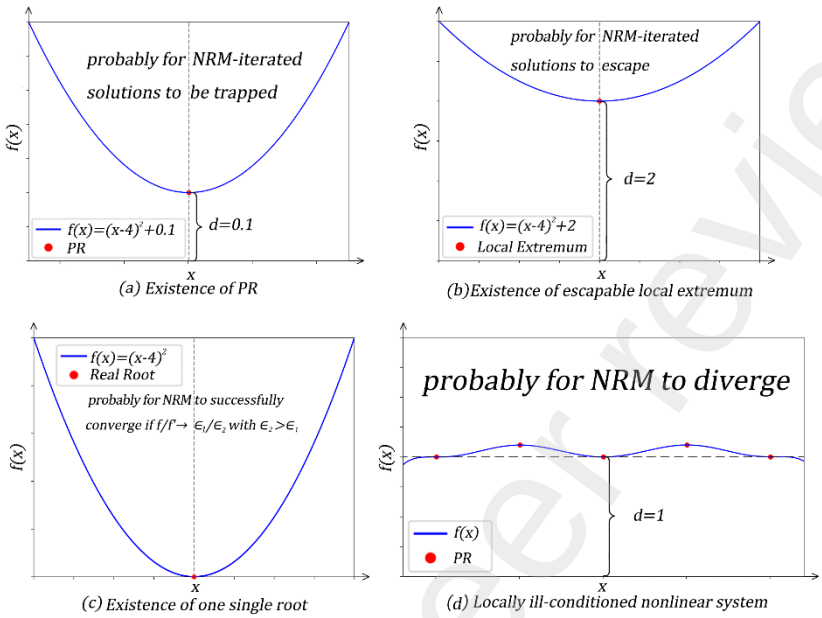

Fig. 1. Four scenarios that seem to trouble NRM. Our imagination should be able to extend to multi-variable (multi-dimension) systems.

3.1. Troubles caused by the existence of pseudo roots (PRs)

Based on our simulation experiences, among these four cases mentioned above, the most troublesome one belongs to Fig. 1. (a), which consequently has drawn most research attention to ours. It is well known that NRM iteration is based on the formula, namely $x = \bar{x} - \bar{f}/\bar{f'}$. Near a PR, both the numerator and the denominator approach zero, thus allowing the ratio, namely a small-valued numerator divided by a small-valued denominator (briefly written as ϵ/ϵ), to remain as a finite value. Near PR, if the initial guess is located rightward to PR, $\bar{f'}$ is positive and the new iterated value of x may decrease to reach leftward of PR. At this new location, $\bar{f'}$ clearly becomes negative, forcing the ratio to become negative and thus prompting the x value to increase. Consequently, the iterated value will fluctuate around PR, as if it is trapped by a barrier.

Differently, in the case of Fig. 1. (b), when the iterated solution approaches the local minimum, the ratio will become finite/small, i.e. a large finite number, and the iterated solution will escape from the trap, suggesting that NRM may be no longer seriously troubled in this case. In Fig. 1. (c), although the ratio appears ϵ/ϵ again, often a portion of the

numerator will cancel with that of the denominator (e.g. $x^3/3x^2 = x/3 \neq \infty.$). Finally, in the case of Fig. 1. (d), the number of unknowns will exceed that of equations, leading to an ill-conditioned problem. Such a scenario may remind us of a simple similar system consisting of two equations (e.g. $x + 2y = 3; 2000x + 4001y = 6001$). Under this circumstance, prior to seeking the solution, we should first pre-condition the system, and this treatment is unrelated to NRM or other nonlinear solvers.

3.2. Strategies of the proposed algorithm

Overall, we authors abide by the philosophy of DNS, a.k.a. Direct Numerical Simulation, in which Navier-Stokes equations of fluid mechanics are solved within the whole range of spatial and temporal domains without having to rely on turbulence modeling. By analogy, NRM serves as the most fundamental tool to handle nonlinear systems, whereas Navier-Stokes equations serve as the most fundamental tool to simulate computational fluid mechanics. In both realms, the prowess of computer-computing speeds belongs to preferable concern. In other words, we continue to faithfully follow the term-by-term linearization of NRM, i.e. Taylor-Series expansion in the first order, e.g. $x^2 \approx -\bar{x}^2 + 2\bar{x}x$, and do not modify NRM.

Instead, our strategy focuses on parametrically modifying the nonlinear system itself. When these parameters, which are imbedded in nonlinear equations, gradually diminish to zero, the parameterized system is recovered to the originally-given one. The purpose of this modification lies in that those existent lumps or parallelisms, shown in Fig. 1a-d, are eliminated during the beginning phase of iterations. Only when the iterated solution enters the vicinity of a root, do they reappear. By then, this reappearance does not matter any longer because our iterated solution has nearly reached its destination.

Realizing that procedures of modifying the given nonlinear system may abound (for example, a parameter ξ can be inserted in front of a likely trouble-making term e^{8x}, and then is allowed to diminish from unity to zero), we have aimed at choosing one that is capable of avoiding the scenario in which the Jacobian determinant approaches zero. In particular, the concept of matrix diagonal dominance constitutes the essential idea. Into each equation, we insert a PA, Mx, where M is a large-valued parameter, which will be properly estimated and gradually reduced to zero. For example, let us examine an extreme case, in which $f(x) = 0$ represents a complicatedly wiggling curve. If we transform this equation into $Mx + f(x) = 0$, or $x + \epsilon f(x) = 0$, the second term can almost be neglected. Hence, at the beginning of the iteration procedure, the originally-given equation is indeed smoothed. As the iteration proceeds, M gradually decreases, and the so-called linearized line is also gradually recovered to a wiggling curve.

3.3. NRM applied to example 1 for the purpose of comparison

In Fig. 2, we present a 7th-degree one-variable polynomial equation to demonstrate how

NRM tends to wander around the PR and fail to converge. In total, seven roots, 1 real root ($x = -2$) and 3 pairs of CRs ($x = -4.0073 \pm 0.1151i$, $x = 5.2807 \pm 0.7576i$, and $1.1476 \pm 0.1458i$) exist. At this moment, let us pay attention to only the real root, which is flanked by two PRs at $x = -4$ and $x = 1.1451$ (also local extrema). Regarding CRs and their relevance with this study will be briefly mentioned in Section 4.

Two initial guesses ($x = -5, 1.2$) are taken. It is well known that NRM iteration is based on the formula, namely $x = \bar{x} - \bar{f}/\bar{f'}$. Near a PR, both the numerator and the denominator approach zero, thus allowing the ratio ϵ/ϵ to remain as a finite value. Near $x = 1.1451$ (right PR), if the initial guess is located rightward relatively to the local minima, $\bar{f'}$ is positive and the new iterated value of x may decrease and reach leftward of the minima. At this new location, $\bar{f'}$ clearly becomes negative, forcing the ratio to become negative and thus prompting the x value to increase. In Fig. 2, root ($x = -2$) is not shown because it has failed to be reached by NRM iterations. Naturally, if we take an initial guess in the vicinity of the root, NRM iteration will converge.

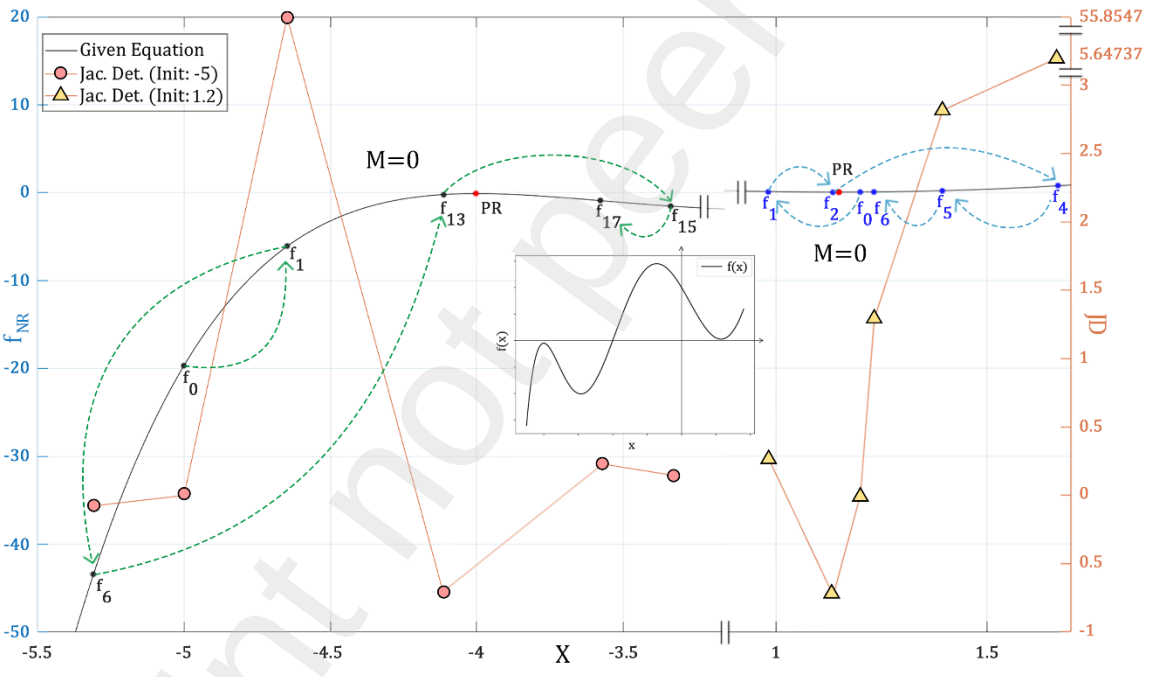

Fig. 2. Progressing route of NRM iterations for Eq. (1). Subscripts of function f denote i^{th} iterations; dash lines denote iteration routes; corresponding Jacobian determinants are quantified on the right JD axis. Note that the graph of Eq. (1), which clearly depicts the real root being flanked by two PRs, is shown in the central area.

3.4. PA algorithm applied to example 1

For fairness, the same initial guesses are taken when the PA algorithm is adopted. In Fig. 3. (a), in reality, a family of 24 parameterized (in M) curves should have been drawn. For clarity, however, only six of them are chosen and presented. Among these six, however, three of them almost coincide, in the vicinity of zero M, with one another.

At the beginning of the iteration procedure (subscripted with 0), x_0 value is guessed and M_0 value is estimated. When the iterated solution has converged, this converged solution will become the initial guess for the M_1 parameter curve. Notably, values of Jacobian Determinants (JD) have remained positive from approximately 1000 to 3 without changing signs, demonstrating a quite different behavior from that of NRM. Also, at the eighth parametric iteration (i.e. $M_8 = 5.05$), the iterated solution (i.e. $x_8 = -1.53$) has already safely passed the PR at $x = -4$.

In Fig. 3. (b), the initial guess, x_0, is taken at 1.2. We observe that values of JDs also remain positive throughout the entire iteration procedure, suggesting that the PA algorithm has managed to avoid the undesirable existence of PR by "linearizing" the nonlinear system.

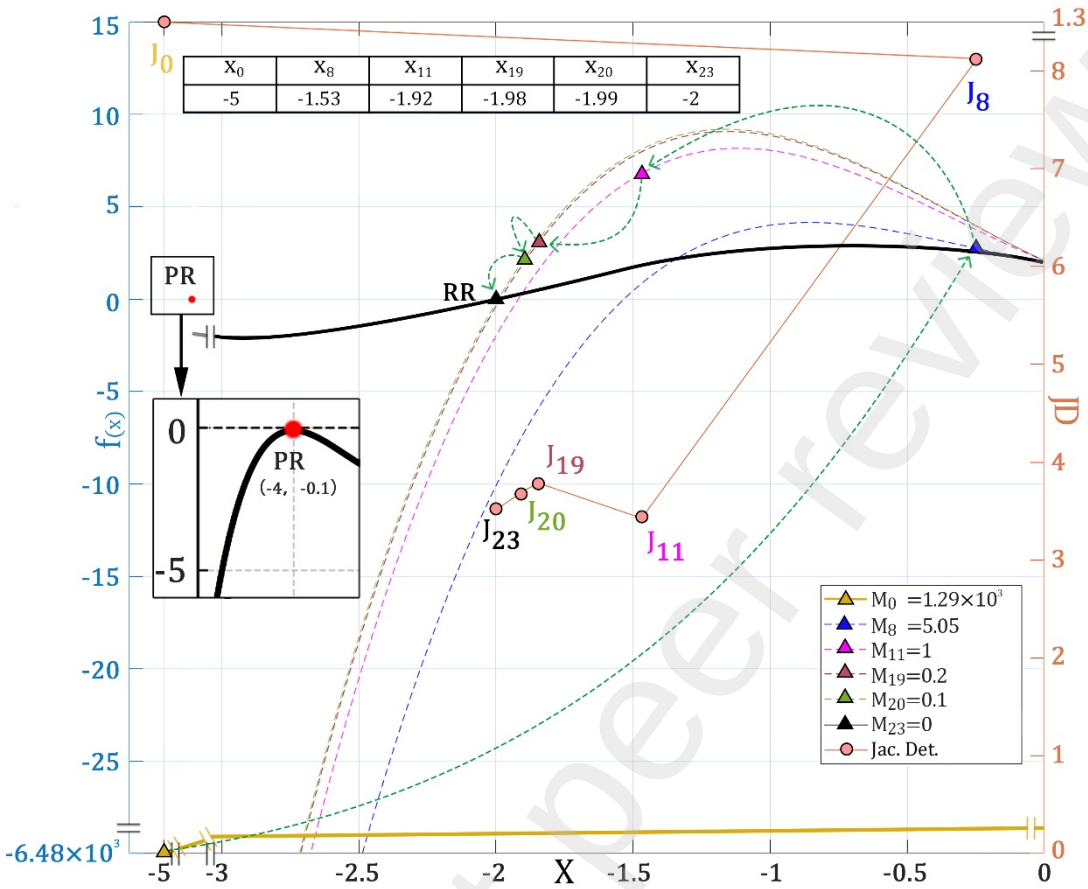

Fig. 3. (a) Relationship among four types of quantities, namely x, $f(x)$, M, JD, denoting variable, function, PA parameter, and Jacobian determinants. Due to the fact that magnitudes of these quantities differ among them, a few broken signs are drawn. The initial modified curve (M_0) is severely "linearized" to almost look like a straight line.

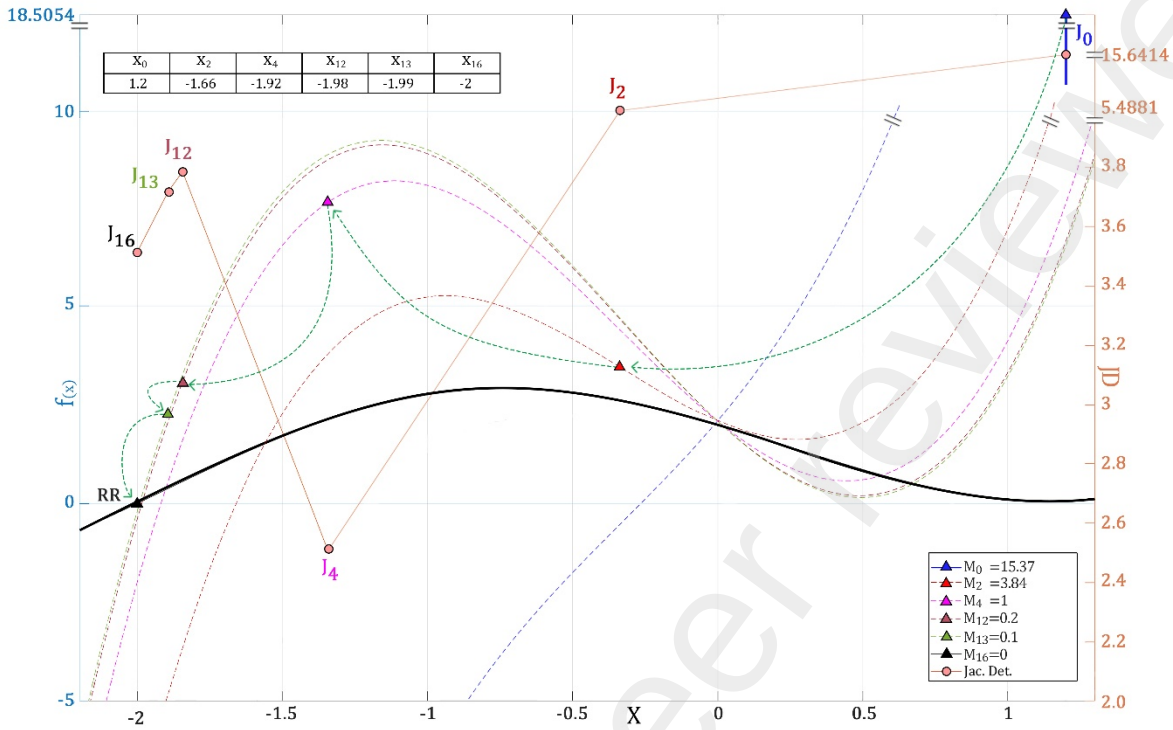

Fig. 3. (b) Relationship among four types of quantities, namely x, $f(x)$, M, JD, denoting variable, function, PA parameter, and Jacobian parameters respectively, appear. Due to the fact that the magnitudes of these quantities differ among them, a few broken signs are used.

3.5. PA algorithm applied to example 2

When the number of unknowns increases to 2, the convergence of the proposed PA algorithm is no longer unconditionally achieved if the value of PA is kept positive all the time. Incidentally, due to the need for pre-conditioning, the number of unknowns is actually 4. However, because those two additional equations appear merely for the purpose of computational convenience, they are not imbedded with PAs. Based on our simulation experiences and logical intuition, we decided to change the signs of PAs when iterations have failed to converge. Our logical intuition lies in that, in an NRM-linearized system of $a_{11}x + a_{12}y = b_1$ and $a_{21}x + a_{22}y = b_2$, clearly signs of a_{11} and a_{22}, in which PAs are imbedded, greatly influence structures and slopes of two linearized curves. Consequently, we adopt the sequence of $[M(+), N(+)]$, $[M(+), N(-)]$, $[M(-), N(+)]$,and $[M(-), N(-)]$, where M denotes the PA for x and N denotes the PA for y. Table 1 lists the simulation results. In comparison, NRM has failed to converge with same initial guesses in all four cases.

Table 1 Convergent results of example #2 & 3 with different combinations of PA signs. P and Q denote PAs for Eq. (3a) and Eq. (3b), respectively.

Trial	Initial Guess of Examples #2 & 3	PA (M, N, P, Q) Signs
Example #2		
#1	$(-0.5, -0.3)$	$(-,-)$
#2	$(0.45, 0.25)$	$(-,+)$
#3	$(-0.1, -0.05)$	$(+,-)$
Example #3		
#4	$(1, -16, -42, 2)$	$(-,-,+,-)$
#5	$(-2, 100, -4, 29)$	$(+,-,-,+)$
#6	$(80, -6, 23, -72)$	$(-,+,+,-)$
#7	$(13, 22, -10, 30)$	$(+,-,+,-)$

With the aid of the PA algorithm, the convergence is achieved for all initial guesses as long as signs of M_i and N_i are allowed to paired and changed (see trials #1, #2, and #3 in Table 1). Here, M_i denotes the PA for Eq. (2a); N_i for Eq. (2b); and the subscript denotes the iteration-sequence number. Expectedly, NRM encounters difficulties of convergence for most of the initial guesses, and results are omitted.

3.6. PA algorithm applied to example 3

Because of three reasons, the nonlinear system of Eq. (3) that contains 4 unknowns (or 10 unknowns if intermediate variables are also included) is presented. First, the convergence robustness of the proposed algorithm is demonstrated for a highly nonlinear system. Second, during our simulation research, we are pleasantly surprised to become aware that the finding of CRs often helps researchers discover the existence of real roots or PRs as well. For example, as aforementioned in example 1, two pairs of CRs do exist near those two PRs. In addition, Fig. 4 shows that numerous satellites of CRs develop a cluster that surrounds a real root. Third, for highly nonlinear systems, it is noted that finding CRs appears easier than finding real roots, partly because CRs must exist in pairs if they do exist.

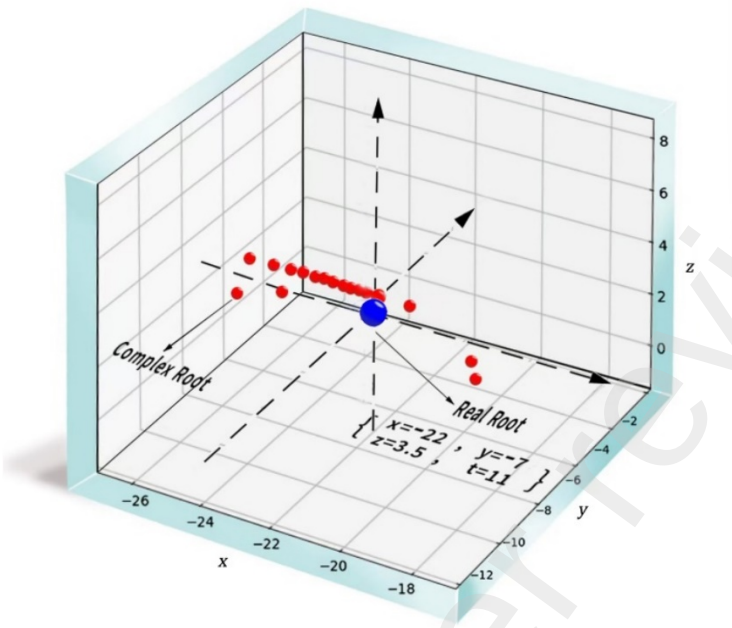

Fig. 4. A real root and CRs in the multi-dimensional solution space, highlighting the real root position at $x = -22$, $y = -7$, $z = 3.5$, and $t = 11$ and the distribution of CRs.

In Equations 3a-d that contain 4 variables, the multiplied product of all diagonal terms in the Jacobian matrix, Ω, can be written as

$$\Omega = \left(\partial f_1 / \partial x\right)\left(\partial f_2 / \partial y\right)\left(\partial f_3 / \partial z\right)\left(\partial f_4 / \partial t\right) = a_{11}a_{22}a_{33}a_{44}, \quad (5)$$

where a_{11} denotes the coefficient of x after the linearization among others and so do others similarly. When anchoring terms, namely $M_i x_i$, are inserted into all 4 equations, Ω becomes

$$\Omega = (M_1 + a_{11})(M_2 + a_{22})(M_3 + a_{33})(M_4 + a_{44}). \quad (6)$$

Due to the existence of PAs, Ω dominates all other products in the Jacobian determinant during most iterations. When PAs gradually diminish to zero, the given nonlinear system is recovered. At this time, because all iterated variables have already approached the root and are located far away from PAs, the Jacobian-matrix diagonal dominance is no longer required.

In Table 1, 4 sets of initial guesses (trials #4 - #7) and their successful corresponding sign groups of PAs, namely M, N, P, Q, are listed. With the aid of the PA algorithm, 16 real roots and approximately 1,000 CRs (~ 500 pairs) are found. In Appendix C2, only 50 pairs are

presented, with other 450 pairs available upon request.

3.7. PA algorithm applied to example 4

In example #4, we investigate a discretized partial differential equation that is governed by heat-conduction energy conservation with a temperature-dependent diffusivity. A heat source is intentionally embedded in the middle finite-difference element of a vertical slab so that the left-side heat conduction (the first derivative of the temperature) entering the element does not equal the right-side counterpart that exits from the element. In other words, this nonlinear system does not belong to C^1 space. Due to the aid of the PA algorithm, the solution converges and the global energy balance is safely checked at every time step (Appendix D).

4. Discussion

In this section, a few relevant issues are discussed below.

4.1. Potential benefit of finding complex roots (CRs)

It is well known that, in quantum mechanics, complex numbers are used in Schrodinger equation, which solves the uncertain position of a particle at a given time, and that, in Fourier transforms, complex numbers are used in understanding oscillation occurrence both in alternating currents and in signals modulated by electromagnetic waves. Furthermore, in finding the golden ratio, $\varphi = (1 \pm \sqrt{5})/2$, another golden ratio is found to equal -0.618. Although it is not a complex number nor a physically meaningful quantity, it may become magically useful in the future.

During the search for real roots for Eq. (3a-d), we have also obtained approximately 1000 CRs (~ 500 pairs). By using an extremely simple example, an offer of the reason why real roots may behave as centers of complex-root clusters is attempted below. Consider roots of $a x^2 + bx + c = 0$, given in the textbook of middle schools as $\left(-b \pm \sqrt{b^2 - 4ac}\right)/2$. If the quantity inside the square root is positive, two real roots exist; if it equals zero, one single real root exists; if it equals a small negative number, a pair of two CRs, named as a PA in the proposed study, exist; if it equals a finite or large negative number, a pair of regular CRs exist. Due to this relationship, a numerical scheme employing the proposed PA algorithm with the aid of finding CRs may secure high convergence robustness. Hopefully, this research task can be conducted in the future.

Finally, let us mention that, via the experience of solving the third example related to GANs, we have learned that, generally, the possibility of obtaining CRs exceeds that of obtaining real roots. Two reasons are briefly mentioned below. (1) Upon convergence, if the initial guess is complex numbers, then the converged solution can be either a real root or a CR. However, if the initial guess is real numbers, then the converged solution can only remain as a real root. (2) CRs, if existent, must exist in conjugate pairs because $(a + bi)(a - bi) =$

$a^2 + b^2$ appears as the only way to allow a portion of complex numbers to be transformed into real numbers. Hence, unless the original nonlinear system contains a complex number coefficient, otherwise it is nearly impossible for a single non-paired CR to exist.

4.2. Preconditioning

In examples that are involved with exponential terms or high-degreed polynomials, it is recommended that we adopt the preconditioning procedure before marching into iterations. Usually, in this preconditioning procedure, a new intermediate variable, T, is introduced to set equal to a ratio, such as f/g, so that $Tg = f$ and the denominator is eliminated. In addition, the magnitudes of all equations should be wisely and nearly unified.

4.3. Slightly nonlinear systems

Naturally, partial differential equations, which are subject to smooth boundary conditions and do not contain source or sink terms, lead to relatively smooth nonlinear algebraic equations that can be readily tackled by the traditional NRM.

4.4. Temperature distribution that does not belong to C^1 space

In example #4, we investigate a discretized partial differential equation that is governed by heat-conduction energy conservation with a temperature-dependent diffusivity. A heat source is intentionally embedded in the middle finite-difference element of a vertical slab so that the left-side heat conduction (proportional to the first derivative of the temperature) entering the element does not equal the right-side counterpart that exits from the element. Due to the aid of the PA algorithm, the solution converges and the global energy balance is safely checked at every time step.

5. Conclusion

Regarding remedying the shortcomings of NRM when PRs exist in nonlinear systems, relevant strategies have rarely been reported in the literature. Herein we insert PAs in all equations so that Jacobian-determinant values of parametrized systems mostly stay sufficiently distantly away from the neighborhood of PRs. Finally, we also report an additional potential of using PAs to find CRs, thus helping find real roots as well as possibly offering scientific intrigues in the future.

Declaration of competing interest

We authors declare that our work is absolutely unrelated to any financial, personal, or commercial interests.

Data availability

We have included our data on the appendix pages, which can be freely accessed.

Acknowledgments

We would like to express our sincere gratitude to Professor Zhong Chen, the Dean of the School of Electronic Science at Xiamen University, for his invaluable support and guidance throughout this project. His leadership has been crucial in advancing our research under the National Key R&D Program: **Development and Engineering of Core Magnetic Resonance Amplification Equipment** (Project Code: 2022YFF0700703) and the National Natural Science Foundation of China–Israel International Cooperation Project: **Ultra-High-Field Magnetic Resonance Imaging and Spectroscopy: Development of New Methods and Applications** (Project Code: 22161142004). These projects have provided significant resources and insights that greatly contributed to the success of our work. Finally, our thanks should be extended to Zhi Liu for his effort in finding several roots of example #3.

Appendix A. Coefficient of Equation 1.

Table A.1
Coefficient of equation 1.

Coefficient	Value
c_0	2.0000
c_1	−2.1750
c_2	−0.7060
c_3	0.9211
c_4	0.1326
c_5	−0.0696
c_6	−0.0046
c_7	0.0016

Appendix B. Coefficient and Coefficient Matrices of Equation 2a & 2b after linearization.

Table B.1
Coefficient of equation 2a & 2b.

Coefficient	Value

c_0	-3.0000×10^{11}
c_1	2.0000
c_2	3.0000
c_3	4.0000
c_4	8.0000
c_5	13.0000
d_0	-7.0000×10^{11}
d_1	6.0000
d_2	7.0000
d_3	8.0000
d_4	9.0000
d_5	12.0000

linearized coefficient-matrix elements, variable vector, and known-valued vector
$z = 1/x;\ t = 1/y;$ (intermediate quantities for convenience and preconditioning)
$a(1,1) = c_1 + 5c_2x^4\mathrm{y}^6 + 3c_3x^2,\ \ a(1,2) = 6c_2x^5\mathrm{y}^5,$
$a(1,3) = t, a(1,4) = c_5c_6e^{c_6\mathrm{t}} + z$
$a(2,1) = 7d_2x^6\mathrm{y}^4, a(2,2) = d_1 + 4d_2x^7\mathrm{y}^3 + 3d_3\mathrm{y}^2,$
$a(2,3) = d_5d_6e^{d_6z} - t, a(2,4) = -z$
$a(3,1) = z, a(3,3) = x, a(4,2) = t,$
$a(4,4) = y$ (Other unwritten elements all equal zero.)
$b(1) = -c_0 + 10c_2x^5\mathrm{y}^6 + 2c_3x^3 + c_5c_6te^{c_6\mathrm{t}} - c_5e^{c_6\mathrm{t}} + zt,$
$b(2) = -d_0 + 10d_2x^7\mathrm{y}^4 + 2d_3\mathrm{y}^3 + d_5d_6ze^{d_6z} - d_5e^{d_6z} - zt,$
$b(3) = xz + 1,$
$b(4) = \mathrm{y}t + 1.$

Appendix C. Coefficient Matrices, CRs, and Real Roots of the Nonlinear Dynamics System in GANs after Linearization.

linearized coefficient-matrix elements, variable vector, and known-valued vector
$A = x/t;\ B = y/z;\ H = e^{-0.1yt}$ (intermediate quantities for convenience and preconditioning)
$a(1,2) = 1, a(1,5) = 22, a(1,6) = 7, a(1,7) = 3.5,\ a(1,8) = 11,$
$a(2,1) = 0.4x^3, a(2,4) = -6.4t^3, a(2,9) = 0.1e^A, a(2,10) = -0.1e^B$
$a(3,1) = y, a(3,2) = x + 5y^4, a(3,3) = 160z^4 - 4t, a(3,4) = -4z$
$a(4,1) = z, a(4,2) = 2y + t + 2A - 0.1Ht,$
$\mathrm{a}(4,3) = x, a(4,4) = y - 0.1Hy, a(4,9) = 2y$
$a(5,1) = X, a(5,5) = x$
$a(6,2) = Y, a(6,6) = y$
$a(7,3) = Z, a(7,7) = z$
$a(8,4) = T, a(8,8) = t$

$a(9,1) = 1, a(9,4) = -A, a(9,9) = -t$
$a(10,2) = 1, a(10,3) = -B,$
$a(10,10) = -Z$ (Other unwritten elements all equal zero.)
$b(1) = -7,\ b(2) = 0.3x^4 - 4.8t^4 - 0.1e^A(1-A) + 0.1e^B(1-B)$
$b(3) = 4y^5 + 128z^5 + xy - 4zt,$
$b(4) = xz + yt + y^2 + 2Ay + 2131.3 - H(1 + 0.2yt)$
$b(5) = xX + 1,\ b(6) = yY + 1,\ b(7) = zZ + 1,\ b(8) = tT + 1,$
$b(9) = -tA,\ b(10) = -ZB.$

Table C.1
Real roots of the nonlinear dynamics system in GANs.

Real Root	x	y	z	t
1	-22.0000	-7.0000	3.5000	11.0000
2	-1643.8858	-7.4634	2.4509	-821.9429
3	-0.5933	-42.5977	21.2989	0.1513
4	0.3234	-45.0327	22.5163	-0.3668
5	0.8899	-47.2686	23.6343	0.7075
6	15.5354	-9.8317	4.9191	7.7679
7	293.4783	-7.0000	3.5000	-146.7391
8	-89.6691	-6.5315	3 .1994	-44.8346
9	-89.6691	-6.5315	3 .1994	-44.8346
10	-10.7129	-7.0000	3.5000	5.3565
11	4.6010	-16.5441	8.2722	8.2722
12	-16.1965	-7.0000	3.5000	8.0982
13	0.4761	-14.0599	7.0300	-0.2810
14	-2.0073	14.0696	-7.0347	-1.0881
15	-183.2368	-0.8548	1.7126	91.6184
16	176.9846	-0.8513	1.9177	88.4923

Table C.2
Summary version of complex roots in the nonlinear dynamic system for GANs.

Complex Root	x_r	x_i	y_r	y_i	z_r	z_i	t_r	t_i
1	-3827.0056	251.0418	-6.7040	-0.4930	3.5914	-1.9729	-125.5209	-1913.5028
2	-3759.9033	-630.6798	-5.4294	0.9647	-1.7093	4.0156	-315.3399	1879.9517
3	-3220.8331	221.2571	-6.7585	-0.5263	3.3703	-1.8638	-110.6285	-1610.4165
4	-3041.1003	-938.4795	-5.5371	1.6565	-0.5712	2.5264	469.2397	-1520.5502
5	-1788.3528	-596.5841	-5.3560	1.7036	-0.7870	2.3464	298.2920	-894.1764
...	...		...			...		...
100	-133.5313	779.4946	-1.2230	0.3623	-1.0623	-2.4679	66.7656	-389.7473
101	-133.0367	495.6223	-4.0944	0.7863	-2.4437	0.9133	66.5184	-247.8112

102	-131.0331	257.8225	-4.8984	1.8826	-1.1082	1.8168	65.5165	-128.9112
103	-130.5951	468.4095	-4.0640	0.8027	-2.4056	0.8944	65.2976	-234.2047
104	-129.0543	19.0924	-6.8699	0.2041	33.3968	-0.0581	9.5462	64.5272
...	...			...		...		...
200	-22.4820	-199.0779	-7.0000	0.0000	3.5000	0.0000	11.2410	99.5390
201	-22.2753	-144.8445	-7.0000	0.0000	3.5000	0.0000	11.1377	72.4222
202	-22.2155	-126.7546	-7.0000	0.0000	3.5000	0.0000	11.1077	63.3773
203	-22.1615	108.6590	-7.0000	0.0000	3.5000	0.0000	11.0807	54.3295
204	-22.1141	90.5582	-7.0000	0.0000	3.5000	0.0000	11.0570	-45.2791
...	...			...		...		...
300	15.5554	-35.9223	-7.4479	-1.0359	3.7240	0.4995	7.7777	-17.9612
301	15.5645	-17.9786	-8.2113	-1.4006	4.1074	0.6922	7.7822	-8.9893
302	16.4398	1.8560	-9.2069	1.0721	1.9300	4.2149	8.2201	0.9279
303	17.9113	-66.6654	-2.3564	-1.6802	-1.2152	0.1454	8.9556	-33.3327
304	18.4919	-17.4781	-7.7102	0.9617	1.6475	3.5191	8.7391	9.2459
...	...			...		...		...
400	169.7652	9.2402	-6.5965	1.2495	1.5961	2.9415	-4.6201	84.8826
401	173.2456	-987.4161	-6.3657	0.9584	1.6987	3.2913	86.6228	-493.7081
402	177.9505	17.7470	-7.1210	-0.1228	3.6115	0.0162	8.8735	-88.9753
403	184.1343	-896.9345	-5.1181	1.1746	-1.8411	2.7696	-92.0671	448.4672
404	186.4457	139.1357	-3.4434	1.7269	-1.4085	0.6355	69.5678	-93.2229
...	...			...		...		...
500	2131.3413	1247.5884	-6.4447	1.2305	1.3423	2.7921	-623.7942	1065.6706
501	2410.0826	-15.9027	-6.8494	-0.1219	4.0120	-0.4399	-7.9513	-1205.1413
502	2462.7787	-446.1380	-6.4238	-1.2150	1.3324	-2.8588	-223.0690	-1231.3893
503	2805.6744	-514.2570	-6.4166	-1.2219	1.3078	-2.8588	-257.1285	-1402.8372
504	2954.4204	-29.4986	-6.8207	-0.1322	4.1066	-0.5130	-14.7493	-1477.2102

Table C.3
First 50 rows of the full version of complex roots in the nonlinear dynamic system for GANs.

x_r	x_i	y_r	y_i	z_r	z_i	t_r	t_i
-3827.0056	251.0418	-6.7040	-0.4930	3.5914	-1.9729	-125.5209	-1913.5028
-3759.9033	-630.6798	-5.4294	0.9647	-1.7093	4.0156	-315.3399	1879.9517
-3220.8331	221.2571	-6.7585	-0.5263	3.3703	-1.8638	-110.6285	-1610.4165
-3041.1003	-938.4795	-5.5371	1.6565	-0.5712	2.5264	469.2397	-1520.5502
-1788.3528	-596.5841	-5.3560	1.7036	-0.7870	2.3464	298.2920	-894.1764
-1222.2883	163.3070	-6.7648	-1.0517	1.9802	-2.1896	-81.6535	-611.1442
-1213.5499	423.9038	-5.2266	-1.7086	-0.9270	-2.2380	211.9519	606.7750
-1035.9653	-350.6375	-5.1287	-1.9548	-0.8424	-1.9489	-175.3187	517.9827
-1029.5864	-2251.0746	-3.3494	-1.6128	-1.5494	-0.6206	-514.7932	-1125.5373
-940.1189	-10.7766	-2.4968	0.6096	-2.3045	0.1175	-30.6057	-10.8758

-939.7647	-73.2125	-3.0470	0.8199	-2.4161	0.0177	-21.9396	15.1886
-936.6847	34.0505	-2.7237	-0.6507	-2.3327	-0.0682	-33.8813	-2.8246
-797.2886	-4879.0064	-6.0549	-0.9473	1.0225	-4.2978	398.6443	2439.5032
-729.6153	-1724.9502	-6.3485	-2.5818	0.2099	-1.5649	364.8077	862.4751
-728.4713	-1514.9654	-3.2695	-1.6873	-1.4976	-0.5841	-364.2356	-757.4827
-717.4717	-4258.1220	-6.0656	-0.9744	1.0014	-4.1692	358.7358	2129.0610
-676.8906	-1393.0507	-3.2492	-1.7029	-1.4867	-0.5746	-338.4453	-696.5253
-653.6214	-264.7268	-3.9694	-0.7971	-2.3838	-0.8724	-132.3634	326.8107
-635.7368	221.2442	-2.8398	-1.4102	-1.5791	-0.3348	-317.8684	110.6221
-626.1629	352.8926	-1.4396	0.4284	-1.8685	-1.6721	-313.0814	176.4463
-600.1267	-80.1943	-6.5751	1.1610	1.7217	2.5158	-40.0971	300.0633
-597.8452	-3369.6202	-6.0877	-1.0214	0.9779	-3.9562	298.9226	1684.8101
-586.3370	-3287.0709	-6.0904	-1.0264	0.9764	-3.9342	293.1685	1643.5354
-585.9240	630.6092	-4.8837	-0.8198	-1.4921	-2.2795	-0.0606	23.3847
-540.4614	-108.7284	-3.6582	0.4693	-2.3635	0.4548	54.3642	-270.2307
-537.4068	116.8856	-5.1155	-1.4099	-1.4355	-2.6134	-58.4428	-268.7034
-536.8317	643.2465	-4.7819	-0.9612	-1.4854	-2.3065	-7.5542	27.3168
-531.4087	1059.1810	-3.1756	1.7519	-1.4524	0.5394	-265.7043	529.5905
-516.1709	198.8251	-4.9858	-1.6424	-1.1772	-2.0733	99.4125	258.0854
-515.4116	825.0197	-1.4018	0.6426	-1.2705	-1.7873	-20.4796	1.2485
-508.4903	-650.5166	-4.7244	1.0290	-1.4826	2.3188	-13.0946	-27.0394
-477.8444	661.4810	-4.6639	-1.0935	-1.4802	-2.3323	-18.3957	24.2273
-472.2952	-1067.6615	-5.7771	-2.3970	-0.1694	-1.7621	236.1476	533.8307
-466.5200	-2462.8934	-6.1268	-1.0828	0.9760	-3.6878	233.2600	1231.4467
-454.6220	99.8647	-6.4018	-1.0607	1.5456	-2.8959	49.9324	227.3110
-447.5814	677.5564	-4.6021	-1.1549	-1.4763	-2.3476	-21.8609	19.5002
-417.7535	-51.0111	-6.5090	1.1892	1.6639	2.6099	-25.5056	208.8768
-397.2948	-719.3361	-4.4763	1.2711	-1.4596	2.3788	-22.3586	-10.1608
-378.7044	80.1942	-5.0442	-1.4786	-1.4277	-2.4793	-40.0971	-189.3522
-377.3294	-727.3724	-3.0596	-1.8104	-1.4106	-0.4827	-188.6647	-363.6862
-367.1840	-1565.8932	-5.6935	-2.1583	-0.3054	-2.2392	-5.0432	-53.7176
-356.3739	249.9108	-4.7544	-1.3677	-1.6386	-2.1111	-178.1870	124.9554
-355.9206	-74.6593	-5.0318	1.4907	-1.4262	2.4592	-37.3297	177.9603
-355.1423	1765.7345	-6.1832	1.1386	1.0191	3.4328	177.5711	-882.8672
-344.0403	38.8429	-6.4785	-1.2043	1.6448	-2.6458	-19.4214	-172.0201
-343.7958	18.4709	-6.9702	0.1008	3.3817	0.0623	9.2354	171.8979
-341.0326	-80.3848	-6.3866	1.0476	1.5243	2.8868	40.1924	-170.5163
-340.0836	341.3255	-0.8791	-1.3331	-0.6825	0.4358	-170.6628	-170.0418
-339.3235	-544.2263	-1.3240	-0.3952	-1.5637	2.1503	-272.1131	169.6618

Appendix D. Coefficient, Discretized Nonlinear Equations and Coefficient Matrices of Ten-variable Heat Conduction Slab Problem after Linearization.

Table D.1
Coefficient of ten-variable heat conduction slab problem.

Coefficient	Value
ρ	8960
c_V	385
A	1
dx	0.0001
dt	0.0001
c_1	0.1467
k_i	$404.6675 - ((T_{i-0.5} + T_{i+0.5}) / 2) * c1;$

Discretized Nonlinear Equations

$\beta = \frac{\alpha_{\mathrm{dt}}}{\Delta x^2}$; $\theta = \frac{h_{\mathrm{dt}}}{\rho c_V dx}$; (intermediate quantities for convenience and preconditioning)

$$T_1 = 25 \tag{D.1}$$

$$T_i - Tp_i = \beta_{i+0.5}(T_{i+1} - T_i) - \beta_{i-0.5}(T_i - T_{i-1}), \text{ for } i \in \{2, 3, 4, 5\} \cup \{7, 8, 9, 10\} \tag{D.2}$$

$$T_6 - Tp_6 = 2.1452 \times 10^{-3} + \beta_{6.5}(T_7 - T_6) - \beta_{5.5}(T_6 - T_5) \tag{D.3}$$

$$T_{11} - Tp_{11} = 2\beta_{10.5}(T_{11} - T_{10}) - 2\theta(T_{11} - 25) \tag{D.4}$$

linearized coefficient-matrix elements, variable vector, and known-valued vector

$m = \frac{\rho \cdot cv \cdot dx^2}{dt}$; $n = h \cdot dx$; (intermediate quantities for convenience and preconditioning)

$a(1,1) = 1,$
$a(2,1) = -k_{1.5}, a(2,2) = \mathrm{m} + k_{1.5} + k_{2.5}, a(2,3) = -k_{2.5},$
$a(3,2) = -k_{2.5}, a(3,3) = \mathrm{m} + k_{2.5} + k_{3.5}, a(3,4) = -k_{3.5},$
$a(4,3) = -k_{3.5}, a(4,4) = \mathrm{m} + k_{3.5} + k_{4.5}, a(4,5) = -k_{4.5},$
$a(5,4) = -k_{4.5}, a(5,5) = \mathrm{m} + k_{4.5} + k_{5.5}, a(5,6) = -k_{5.5},$
$a(6,5) = -k_{5.5}, a(6,6) = \mathrm{m} + k_{5.5} + k_{6.5}, a(6,7) = -k_{6.5},$
$a(7,6) = -k_{6.5}, a(7,7) = \mathrm{m} + k_{6.5} + k_{7.5}, a(7,8) = -k_{7.5},$
$a(8,7) = -k_{7.5}, a(8,8) = \mathrm{m} + k_{7.5} + k_{8.5}, a(8,9) = -k_{8.5},$
$a(9,8) = -k_{8.5}, a(9,9) = \mathrm{m} + k_{8.5} + k_{9.5}, a(9,10) = -k_{9.5},$
$a(10,9) = -k_{9.5}, a(10,10) = \mathrm{m} + k_{9.5} + k_{10.5}, a(10,11) = -k_{10.5},$

$a(11,10) = -2k_{10.5}, a(11,11) = \mathbb{m} + 2k_{10.5} + 2n.$ (Other unwritten elements all equal zero.)
$b(1) = 25,\ b(2) = mTp(2),\ b(3) = mTp(3),$
$b(4) = mTp(4),\ b(5) = mTp(5) + 38400,\ b(6) = mTp(6),$
$b(7) = mTp(7),\ b(8) = mTp(8),\ b(9) = mTp(9),$
$b(10) = mTp(10),\ b(11) = mTp(11) + 50n.$

References

[1] S. Dong, Z. Li, Local extreme learning machines and domain decomposition for solving linear and nonlinear partial differential equations, Comput. Methods Appl. Mech. Eng. 387 (2021) 114129. https://doi.org/10.1016/j.cma.2021.114129.

[2] S. Fresca, A. Manzoni, POD-DL-ROM: Enhancing deep learning-based reduced order models for nonlinear parametrized PDEs by proper orthogonal decomposition, Comput. Methods Appl. Mech. Eng. 388 (2022) 114181. https://doi.org/10.1016/j.cma.2021.114181.

[3] J. Bai, G.R. Liu, A. Gupta, L. Alzubaidi, X.Q. Feng, Y. Gu, Physics-informed radial basis network (PIRBN): A local approximating neural network for solving nonlinear partial differential equations, Comput. Methods Appl. Mech. Eng. 415 (2023) 116290. https://doi.org/10.1016/j.cma.2023.116290.

[4] M.Z. Lyu, D.C. Feng, J.B. Chen, J. Li, A decoupled approach for determination of the joint probability density function of a high-dimensional nonlinear stochastic dynamical system via the probability density evolution method, Comput. Methods Appl. Mech. Eng. 418 (2024) 116443. https://doi.org/10.1016/j.cma.2023.116443.

[5] M. Rasheed, S. Shihab, A. Rashid, T. Rashid, S.H.A. Hamed, S. Saber Abdulwahhab, Some Step Methods Applied to Nonlinear Equation, J. Al-Qadisiyah Comput. Sci. Math. 13 (2) (2021) 69–77. https://doi.org/10.29304/jqcm.2021.13.2.800.

[6] W. Gong, Z. Liao, X. Mi, L. Wang, Y. Guo, Nonlinear equations solving with intelligent optimization algorithms: A survey, Complex Syst. Mod. Simul. 1 (1) (2021) 15–32. https://doi.org/10.23919/CSMS.2021.0002.

[7] C.J. Lin, R.C. Weng, S.S. Keerthi, Trust region Newton method for large-scale logistic regression, J. Mach. Learn. Res. 9 (4) (2008) 627–650. https://doi.org/10.1145/1273496.1273567.

[8] A. Chauhan, A study of modified Newton-Raphson method, J. Univ. Shanghai Sci. Technol. 23 (8) (2021) 129–134. https://jusst.org/wp-content/uploads/2021/08/A-Study-of-Modified-Newton-Raphson-Method.pdf

[9] F. Casella, B. Bachmann, On the choice of initial guesses for the Newton-Raphson algorithm, Appl. Math. Comput. 398 (2021) 125991. https://doi.org/10.1016/j.amc.2021.125991.

[10] R. Yuan, A. Lazaric, R.M. Gower, Sketched Newton-Raphson, SIAM J. Optim. 32 (3) (2022) 1555–1583. https://doi.org/10.1137/21M139788X.

[11] U. Ozdemir, Comparison of the Newton--Raphson Method and genetic algorithm solutions for nonlinear aircraft trim analysis, J. Aerosp. Eng. 237 (3) (2023) 725–740. https://doi.org/10.1177/09544100221107726.

[12] K.H. Pho, Improvements of the Newton--Raphson method, J. Comput. Appl. Math. 408 (2022) 114106. https://doi.org/10.1016/j.cam.2022.114106.

[13] M. Tajima, T. Yamada, Topology optimization with geometric constraints for additive manufacturing based on coupled fictitious physical model, Comput. Methods Appl. Mech. Eng. 417 (2023) 116415. https://doi.org/10.1016/j.cma.2023.116415.

[14] T. Teijeiro, J.M. Taylor, A. Hashemian, D. Pardo, Machine learning discovery of optimal quadrature rules for isogeometric analysis, Comput. Methods Appl. Mech. Eng. 416 (2023) 116310.

https://doi.org/10.1016/j.cma.2023.116310.
[15] H.M. Ridha, H. Hizam, C. Gomes, A.A. Heidari, H. Chen, M. Ahmadipour, D.H. Muhsen, M. Alghrairi, Parameters extraction of three diode photovoltaic models using boosted LSHADE algorithm and Newton Raphson method, Energy 224 (2021) 120136. https://doi.org/10.1016/j.energy.2021.120136.
[16] C. Kumar, D.M. Mary, Parameter estimation of three-diode solar photovoltaic model using an Improved-African Vultures optimization algorithm with Newton--Raphson method, J. Comput. Electron. 20 (2021) 2563–2593. https://doi.org/10.1007/s10825-021-01812-6.
[17] H.M. Ridha, H. Hizam, S. Mirjalili, M.L. Othman, M.E. Ya'Acob, M. Ahmadipour, Parameter extraction of single, double, and three diodes photovoltaic model based on guaranteed convergence arithmetic optimization algorithm and modified third order Newton Raphson methods, Renew. Sustain. Energy Rev. 162 (2022) 112436. https://doi.org/10.1016/j.rser.2022.112436.
[18] D. Wang, Z. Gao, Distributed finite-time optimization algorithms with a modified Newton--Raphson method, Neurocomputing 536 (2023) 73–79. https://doi.org/10.1016/j.neucom.2023.03.027.
[19] M. Salvador, A.L. Marsden, Branched latent neural maps, Comput. Methods Appl. Mech. Eng. 418 (2024) 116499. https://doi.org/10.1016/j.cma.2023.116499.