Journal of Applied Analysis and Computation
Volume 15, Number 4, August 2025, 2238–2257
Website:http://www.jaac-online.com
DOI:10.11948/20240428

INVERSE-FREE NEWTON'S METHOD

Marcin Massalski[1] and Magdalena Nockowska-Rosiak[1,†]

Abstract We present a modification of Newton's method for finding a zero of a multivariable function without an inverse of a matrix in a recurrence. The aim of this paper is twofold: demonstrating at least quadratic convergence of a Newton-type method avoiding matrix inversion under standard assumptions, and then comparing modified and classical Newton's methods numerically.

Keywords Newton's method, at least quadratic convergence, approximate inverse.

MSC(2010) 65H10, 65Q30.

1. Introduction

Blanchard and Chamberland in [3] proposed a modification of Newton's method for root-finding of a function in the one-dimensional case. Instead of the classical recursion

$$x_{n+1} = x_n - \frac{f(x_n)}{f'(x_n)} \tag{1.1}$$

considering

$$\begin{cases} y_{n+1} = y_n(2 - f'(x_n)y_n), \\ x_{n+1} = x_n - y_{n+1}f(x_n) \end{cases} \tag{1.2}$$

where $f : \mathbb{C} \to \mathbb{C}$ is twice differentiable and a starting point of the given recurrence is close to the simple root of f - x^*, which means that $f(x^*) = 0$ and $f'(x^*) \neq 0$. y_{n+1} from the first equation in system in Eq. (1.2) approximates $1/f'(x_n)$. The authors proved the quasi-quadratically convergence of modified Newton's method, and left the case of Newton's multivariate modification as an open problem. More precisely, in the main theorem Blanchard and Chamberland showed that $\varepsilon_{n+1} \backsim (3n)\frac{f''(x^*)}{2f'(x^*)}\varepsilon_n^2$, where ε_n denotes an error in the n-th iteration.

The standard result in the multivariable case guarantees at least quadratic convergence to the simple zero of $\boldsymbol{f}$ of Newton's method. Let's us recall the notion of a sequence convergent with at least quadratic order.

Definition 1.1 ([33]). Let $(\boldsymbol{x}_n)_{n\in\mathbb{N}} \subset \mathbb{R}^N$ be a sequence convergent to $\boldsymbol{x}^*$. The sequence $(\boldsymbol{x}_n)_{n\in\mathbb{N}}$ is said to be convergent with at least quadratic order (at least quadratically), if there exists a sequence $(a_n)_{n\in\mathbb{N}}$ of positive numbers convergent to

†The corresponding author.
[1]Institute of Mathematics, Lodz University of Technology, al. Politechniki 8, 93-590, Łódź, Poland
Email: marcin7cd@gmail.com(M. Massalski), magdalena.nockowska@p.lodz.pl(M. Nockowska-Rosiak)

0 and numbers $\mu > 0$, $n' \in \mathbb{N}$ such that

$$\|\boldsymbol{x}_n - \boldsymbol{x}^*\|_\infty \leqslant a_n \quad \text{for } n \geqslant n', \quad \text{and} \quad \lim_{n\to\infty} \frac{a_{n+1}}{a_n^2} = \mu.$$

Newton's method in multivariable case is given by the following recurrence

$$\boldsymbol{x}_{n+1} = \boldsymbol{x}_n - (\boldsymbol{f}'(\boldsymbol{x}_n))^{-1}\boldsymbol{f}(\boldsymbol{x}_n) \tag{1.3}$$

for $\boldsymbol{f} : \mathbb{R}^N \to \mathbb{R}^N$, with initial point $\boldsymbol{x}_1 \in \mathbb{R}^N$. Now we present the classical result about convergence of Newton's method.

Theorem 1.1 (Theorem 4.4, 119, [33]). *Assume that $\boldsymbol{f} : \mathbb{R}^N \to \mathbb{R}^N$, $\boldsymbol{f} \in C^2(\mathbb{R}^N)$. If there exists $\boldsymbol{x}^* \in \mathbb{R}^N$ such that $\boldsymbol{f}(\boldsymbol{x}^*) = \boldsymbol{0}_N$ and $\det(\boldsymbol{f}'(\boldsymbol{x}^*)) \neq 0$, then there exists a neighbourhood S of the point $\boldsymbol{x}^*$ such that, if $\boldsymbol{x}_1 \in S$, then $(\boldsymbol{x}_n)_{n\in\mathbb{N}}$ the sequence generated by Eq. (1.3) is convergent to $\boldsymbol{x}^*$ at least quadratically.*

As in practice the matrix $(\boldsymbol{f}'(\boldsymbol{x}_n))^{-1}$ is not calculated, here we propose to apply an approximation of $(\boldsymbol{f}'(\boldsymbol{x}_n))^{-1}$ in the recurrence. Inverse of a matrix B can by computed iteratively by

$$X_{n+1} = 2X_n - X_nBX_n,$$

where X_1 is an initial approximation of B^{-1}. This recursion for matrix inversion was first proposed by Schulz [30] (see also [3, 20, 34]). Following the open problem from [3], we consider a recursion below called inverse-free Newton's method

$$\begin{cases} Y_{n+1} = Y_n\left(2I - \boldsymbol{f}'(\boldsymbol{x}_n)Y_n\right), \\ \boldsymbol{x}_{n+1} = \boldsymbol{x}_n - Y_{n+1}\boldsymbol{f}(\boldsymbol{x}_n) \end{cases} \tag{1.4}$$

with an initial state $(\boldsymbol{x}_1, Y_1)$, where $\boldsymbol{x}_1$ is a point from $\mathbb{R}^N$ and Y_1 is a matrix of dimension $N \times N$.

In the main result of this paper, we prove that under standard assumptions the recurrence for inverse-free Newton's method given by Eq. (1.4) is convergent to a zero of $\boldsymbol{f}$ with at least quadratic order. Modification of Newton's method without inverse of matrix, even in Banach space has been studied for example in [1, 2, 14]. In [1, 2] the authors proposed the Newton-like algorithm which requires no inversion or only one inversion of linear operator at each step. The inverse of the operator is replaced by a finite sum of linear operators depending on the Fréchet derivative of the operator. It was proved that the Newton-like algorithm is comparable to Newton's in the sense that the number of iteration steps to reach a given tolerance is essentially the same. In [14] the authors proved quadratic convergence of the proposed directional Newton method under typical assumptions for Newton's method.

The second part of the paper is devoted to comparing both Newton's methods in terms of the running time, the number of iterations and basins of attraction. For a fair comparison, we calculate the step of multivariate classical Newton's method Eq. (1.3) more efficiently, instead of determining $(\boldsymbol{f}'(\boldsymbol{x}_n))^{-1}$ we directly calculate $\boldsymbol{x}_{n+1}$ as a solution to the following system of equations

$$\boldsymbol{f}'(\boldsymbol{x}_n)(\boldsymbol{x}_{n+1} - \boldsymbol{x}_n) = \boldsymbol{f}(\boldsymbol{x}_n).$$

Inverse-free Newton's method Eq. (4) is usually better than classical Newton's method Eq. (3) when multiplication of three matrices is faster than solving a

system of linear equations. We will perform the analysis relying on a standard test functions. Numerical experiments will show that in terms of the number of iterations or the computational order of convergence, both methods work similarly. Finally, we will compare classical Newton's method and inverse-free Newton's method in terms of basins of attraction.

In this paper we are only interested in simple zeros of $\boldsymbol{f}$ which means that $\boldsymbol{x}^*$ is a zero of $\boldsymbol{f}$ with nonsingular Jacobi matrix. In this case Newton's method is called nondegenerate. Nondegenerate Newton's method in the one-dimensional case as in the multidimensional case has been the subject of much research see for example, [11, 16, 17, 21, 25, 26, 32, 33]. In [25] the authors proposed algorithms based on Daftardar-Gejji and Jafari decomposition, proving their convergence and stability. Improvement of Newton-method with rate of convergence grater than two has been considered among others in [5, 10, 27]. In [27] the authors proved not only in high rate of convergence, but also gave dynamical analysis to assess stability. Newton's method for degenerate case has been also widely studied in literature for example in [7–9, 12, 15, 22–24, 28, 29, 31].

The paper is structured as follows: In Section 2, we show sufficient conditions on convergence with at least quadratic order of inverse-free Newton's method. At the beginning of the section we collect the required basic results. In Section 3 we compare numerically modified Newton's method to classical Newton's method, using functions from the literature with different numbers of variables, up to functions with 500 variables. We will focus on the running time, the number of iterations and the computational order of convergence. Furthermore, we will compare basins of attraction obtained from both methods.

2. At least quadratic convergence of inverse-free Newton's method

Let us start with recalling very known sufficient condition for the existence of an attractive fixed point of $\boldsymbol{f} : \mathbb{R}^N \to \mathbb{R}^N$.

Definition 2.1 ([19]). Let $\boldsymbol{f} : \mathbb{R}^N \supset D \to \mathbb{R}^N$ be such that $\boldsymbol{f}[D] \subset D$. A point $\boldsymbol{x}^* \in D$ is said to be an attractive point of a recurrence $\boldsymbol{x}_{n+1} = \boldsymbol{f}(\boldsymbol{x}_n)$, if there exists a neighbourhood $S \subset D$ of the point $\boldsymbol{x}^*$ such that $(\boldsymbol{x}_n)_{n\in\mathbb{N}}$ satisfying $\boldsymbol{x}_{n+1} = \boldsymbol{f}(\boldsymbol{x}_n)$ is convergent to $\boldsymbol{x}^*$ for any initial point $\boldsymbol{x}_1 \in S$.

By $M_{N\times N}(\mathbb{R})$ we denote a linear space of real matrices of dimension $N \times N$. For a matrix $B = [b_{ij}] \in M_{N\times N}(\mathbb{R})$ we define a spectral radius of the matrix B, denoted by $\rho(B)$, by formula

$$\rho(B) := \max\{|\lambda_i| : \lambda_i \text{ an eigenvalue of } B, i = 1, \ldots, N\},$$

(see [19]).

Theorem 2.1 ([13, 19]). *Let $\boldsymbol{f} : \mathbb{R}^N \supset D \to \mathbb{R}^N$ be such that $\boldsymbol{f}[D] \subset D$. Let $\boldsymbol{x}^* \in \operatorname{Int} D$ be a fixed point of $\boldsymbol{f}$. If $\boldsymbol{f}$ is differentiable at $\boldsymbol{x}^*$ and the spectral radius of the Jacobi matrix $\boldsymbol{f}'(\boldsymbol{x}^*)$ satisfies $\rho(\boldsymbol{f}'(\boldsymbol{x}^*)) < 1$, then $\boldsymbol{x}^*$ is an attractive point of a recurrence $\boldsymbol{x}_{n+1} = \boldsymbol{f}(\boldsymbol{x}_n)$.*

During this paper we use $\|\cdot\|_\infty$ at $\mathbb{R}^N$, which means

$$\|\boldsymbol{x}\|_\infty = \max_{1\leq i\leq N} |x_i|, \quad \boldsymbol{x} = (x_1, \ldots, x_N) \in \mathbb{R}^N$$

and the operator norm $\|\cdot\|_\infty$ for a linear operator $B : \mathbb{R}^N \to \mathbb{R}^N$ or norm of a matrix $B = [b_{ij}] \in M_{N\times N}(\mathbb{R})$ given by formula

$$\|B\|_\infty = \max_{1\leq i\leq N} \sum_{j=1}^{N} |b_{ij}|.$$

Standard tool for proving the convergence in numerical methods is the Taylor formula. Here we will use the following version of this theorem.

Theorem 2.2 (Theorem. A7, 422, [33])**.** (Taylor) *Let $M, N \in \mathbb{N}$ and $k \in \mathbb{N}\cup\{0\}$. Let U be a convex and open subset of $\mathbb{R}^N$, $\boldsymbol{a} \in U$, $\boldsymbol{f} = (f_1, \ldots, f_M) : U \to \mathbb{R}^M$. Assume that partial derivatives of the order $k+1$ of the function $\boldsymbol{f}$ are bounded on U by a number A, $A > 0$. Then for any $\boldsymbol{\eta} \in \mathbb{R}^N$ such that $\boldsymbol{a}+\boldsymbol{\eta} \in U$ we have*

$$\boldsymbol{f}(\boldsymbol{a}+\boldsymbol{\eta}) = \boldsymbol{f}(\boldsymbol{a}) + \boldsymbol{f}'(\boldsymbol{a})\cdot(\boldsymbol{\eta}) + \frac{1}{2!}\boldsymbol{f}^{(2)}(\boldsymbol{a})\cdot(\boldsymbol{\eta})^2 + \cdots + \frac{1}{k!}\boldsymbol{f}^{(k)}(\boldsymbol{a})\cdot(\boldsymbol{\eta})^k + \boldsymbol{R}_k(\boldsymbol{a}+\boldsymbol{\eta}),$$

where $\boldsymbol{f}^{(r)}(\boldsymbol{a})$ denotes r-linear operator of the derivative of the function $\boldsymbol{f}$ of the order r, $\boldsymbol{f}^{(r)}(\boldsymbol{a})\cdot(\boldsymbol{\eta})^r := \boldsymbol{f}^{(r)}(\boldsymbol{a})\underbrace{(\boldsymbol{\eta}, \ldots, \boldsymbol{\eta})}_{r}$ for $r \in \{1, \ldots, k\}$, and the Taylor remainder $\boldsymbol{R}_k(\boldsymbol{a}+\boldsymbol{\eta})$ satisfies

$$\|\boldsymbol{R}_k(\boldsymbol{a}+\boldsymbol{\eta})\|_\infty \leqslant \frac{1}{(k+1)!} A N^{k+1} \|\boldsymbol{\eta}\|_\infty^{k+1}. \tag{2.1}$$

Now we prove the main result of this paper.

Theorem 2.3. *Let $\boldsymbol{f} : \mathbb{R}^N \to \mathbb{R}^N$ be a C^2 function and $\boldsymbol{x}^* \in \mathbb{R}^N$ be such that $\boldsymbol{f}(\boldsymbol{x}^*) = \boldsymbol{0}_N$ and $\det(\boldsymbol{f}'(\boldsymbol{x}^*)) \neq 0$. Assume that $(Y_n)_{n\in\mathbb{N}} \subset M_{N\times N}(\mathbb{R})$, $(\boldsymbol{x}_n)_{n\in\mathbb{N}} \subset \mathbb{R}^N$ satisfy a recurrence*

$$\begin{cases} Y_{n+1} = Y_n(2I - \boldsymbol{f}'(\boldsymbol{x}_n)Y_n), \\ \boldsymbol{x}_{n+1} = \boldsymbol{x}_n - Y_{n+1}\boldsymbol{f}(\boldsymbol{x}_n), \end{cases} \quad \text{for } n \in \mathbb{N}, \tag{2.2}$$

with an initial state $(\boldsymbol{x}_1, Y_1) \in \mathbb{R}^N \times M_{N\times N}(\mathbb{R})$. Then there exist a neighbourhood of $S_{\boldsymbol{x}^*} \subset \mathbb{R}^N$ of the point $\boldsymbol{x}^*$ and a neighbourhood $S_{Y^*} \subset M_{N\times N}(\mathbb{R})$ of the matrix $Y^* := (\boldsymbol{f}'(\boldsymbol{x}^*))^{-1}$, such that

i) $(\boldsymbol{x}_n)_{n\in\mathbb{N}}$ and $(Y_n)_{n\in\mathbb{N}}$ are convergent to $\boldsymbol{x}^*$ and $(\boldsymbol{f}'(\boldsymbol{x}^*))^{-1}$ respectively,

ii) $(\boldsymbol{x}_n)_{n\in\mathbb{N}}$ is convergent to $\boldsymbol{x}^*$ at least quadratically,

for any $\boldsymbol{x}_1 \in S_{\boldsymbol{x}^*}$ and $Y_1 \in S_{Y^*}$.

Proof. Let us denote $\boldsymbol{f} = (f_1, \ldots, f_N)$ and $f'_{ij} := (f_i)'_{x_j} = \frac{\partial f_i}{\partial x_j}$, $i, j = 1, \ldots, N$. We define $\boldsymbol{g} := (g_{11}, \ldots, g_{1N}, \ldots, g_{NN}, g_1, \ldots, g_N) : \mathbb{R}^{N^2+N} \to \mathbb{R}^{N^2+N}$ by formula

$$g_{ij}(\boldsymbol{y}, \boldsymbol{x}) = 2y_{ij} - \sum_{k=1}^{N}\sum_{l=1}^{N} y_{ik} f'_{kl}(\boldsymbol{x}) y_{lj}, \quad g_i(\boldsymbol{y}, \boldsymbol{x}) = x_i - \sum_{k=1}^{N} g_{ik}(\boldsymbol{y}, \boldsymbol{x}) f_k(\boldsymbol{x}),$$

where $\boldsymbol{y} = (y_{11}, \ldots, y_{1N}, \ldots, y_{NN}) \in \mathbb{R}^{N^2}$, $\boldsymbol{x} = (x_1, \ldots, x_N) \in \mathbb{R}^N$.

In order to simplify the notation, a matrix $Y = [y_{ij}]_{1 \le i,j \le N}$ will be identity with $\boldsymbol{y} = (y_{11}, \ldots, y_{1N}, \ldots, y_{NN}) \in \mathbb{R}^{N^2}$. The recurrence (2.2) can be written as $(\boldsymbol{y}_{n+1}, \boldsymbol{x}_{n+1}) = \boldsymbol{g}(\boldsymbol{y}_n, \boldsymbol{x}_n)$ for $n \in \mathbb{N}$. Additionally, we will define functions $G_{\boldsymbol{y}} : M_{N\times N}(\mathbb{R}) \times \mathbb{R}^N \to M_{N\times N}(\mathbb{R})$ and $G_{\boldsymbol{x}} = (g_1, \ldots, g_N) : M_{N\times N}(\mathbb{R}) \times \mathbb{R}^N \to \mathbb{R}^N$ by formulas

$$G_{\boldsymbol{y}}(Y, \boldsymbol{x}) := Y(2I - \boldsymbol{f}'(\boldsymbol{x})Y), \; G_{\boldsymbol{x}}(Y, \boldsymbol{x}) := \boldsymbol{x} - G_{\boldsymbol{y}}(Y, \boldsymbol{x})\boldsymbol{f}(\boldsymbol{x}),$$
$$Y \in M_{N\times N}(\mathbb{R}), \; \boldsymbol{x} \in \mathbb{R}^n.$$

Note that for $\boldsymbol{y}^*$ which coordinates are equal to appropriate elements of $Y^* = (\boldsymbol{f}'(\boldsymbol{x}^*))^{-1}$ we have

$$G_{\boldsymbol{y}}(Y^*, \boldsymbol{x}^*) = Y^*(2I - \boldsymbol{f}'(\boldsymbol{x}^*)Y^*) = Y^*(2I - I) = Y^*,$$
$$G_{\boldsymbol{x}}(Y^*, \boldsymbol{x}^*) = \boldsymbol{x}^* - Y^*\boldsymbol{f}(\boldsymbol{x}^*) = \boldsymbol{x}^* - \mathbf{0}_N = \boldsymbol{x}^*,$$

which means that $(\boldsymbol{y}^*, \boldsymbol{x}^*)$ is a fixed point of $\boldsymbol{g}$. Moreover,

$$\frac{\partial g_{ij}}{\partial y_{mn}}(\boldsymbol{y}, \boldsymbol{x}) = 2\delta_{im}\delta_{jn} - \delta_{im}[\boldsymbol{f}'(\boldsymbol{x})Y]_{(n,j)} - \delta_{jn}[Y\boldsymbol{f}'(\boldsymbol{x})]_{(i,m)},$$
$$\frac{\partial g_{ij}}{\partial y_{mn}}(\boldsymbol{y}^*, \boldsymbol{x}^*) = 2\delta_{im}\delta_{jn} - \delta_{im}[\boldsymbol{f}'(\boldsymbol{x}^*)Y^*]_{(n,j)} - \delta_{jn}[Y^*\boldsymbol{f}'(\boldsymbol{x}^*)]_{(i,m)} = 0,$$

for $1 \le i, j, l, m \le N$, where δ_{im} denotes the Kronecker delta, $[B]_{(n,j)}$ denotes nj element of matrix B. This means that $(\boldsymbol{g}_{\boldsymbol{y}})'_{\boldsymbol{y}}(\boldsymbol{y}^*, \boldsymbol{x}^*) = \mathbf{0}_{N^2\times N^2}$, where $\mathbf{0}_{k\times k}$ denotes null matrix of dimension $k \times k$. Next

$$(G_{\boldsymbol{x}})'_{\boldsymbol{x}}(Y, \boldsymbol{x}) = I - (G_{\boldsymbol{y}})'_{\boldsymbol{x}}(Y, \boldsymbol{x})\boldsymbol{f}(\boldsymbol{x}) - G_{\boldsymbol{y}}(Y, \boldsymbol{x})\boldsymbol{f}'(\boldsymbol{x}),$$
$$(G_{\boldsymbol{x}})'_{\boldsymbol{x}}(Y^*, \boldsymbol{x}^*) = I - (G_{\boldsymbol{y}})'_{\boldsymbol{x}}(Y^*, \boldsymbol{x}^*)\mathbf{0}_N - (\boldsymbol{f}'(\boldsymbol{x}^*))^{-1}\boldsymbol{f}'(\boldsymbol{x}^*) = \mathbf{0}_{N\times N},$$

$(G_{\boldsymbol{y}})'_{\boldsymbol{x}}(Y^*, \boldsymbol{x}^*)$ can be identified with matrix of dimension $N^2 \times N$. $(G_{\boldsymbol{y}})'_{\boldsymbol{x}}(Y^*, \boldsymbol{x}^*)\mathbf{0}_N$ is a vector from $\mathbb{R}^{N^2}$ and it can be identified with null matrix of dimension $N \times N$. So, $(\boldsymbol{g}_{\boldsymbol{x}})'_{\boldsymbol{x}}(\boldsymbol{y}^*, \boldsymbol{x}^*) = \mathbf{0}_{N\times N}$. Moreover,

$$\frac{\partial g_i}{\partial y_{lj}}(\boldsymbol{y}, \boldsymbol{x}) = -\sum_{k=1}^{N} \frac{\partial g_{ik}}{\partial y_{lj}}(\boldsymbol{y}, \boldsymbol{x}) f_k(\boldsymbol{x}), \quad \frac{\partial g_i}{\partial y_{lj}}(\boldsymbol{y}^*, \boldsymbol{x}^*) = -\sum_{k=1}^{N} \frac{\partial g_{ik}}{\partial y_{lj}}(\boldsymbol{y}^*, \boldsymbol{x}^*)0 = 0$$

for $1 \le i, j, l \le N$. Hence $(\boldsymbol{g}_{\boldsymbol{x}})'_{\boldsymbol{y}}(\boldsymbol{y}^*, \boldsymbol{x}^*) = \mathbf{0}_{N\times N^2}$ and finally,

$$\boldsymbol{g}'(\boldsymbol{y}^*, \boldsymbol{x}^*) = \begin{bmatrix} \mathbf{0}_{N^2\times N^2} & (\boldsymbol{g}_{\boldsymbol{y}})'_{\boldsymbol{x}}(\boldsymbol{y}^*, \boldsymbol{x}^*) \\ \mathbf{0}_{N\times N^2} & \mathbf{0}_{N\times N} \end{bmatrix}.$$

Jacobi matrix of $\boldsymbol{g}$ at $(\boldsymbol{y}^*, \boldsymbol{x}^*)$ is an uppertriangular matrix with zeros on the diagonal, hence all eigenvalues of it are equal to 0 and the spectral radius of $\boldsymbol{g}'(\boldsymbol{y}^*, \boldsymbol{x}^*)$ is equal to 0. By Theorem 2.1 $(\boldsymbol{y}^*, \boldsymbol{x}^*)$ is the attractive fixed point of $\boldsymbol{g}$ and there exist $S_{\boldsymbol{y}^*} \subset \mathbb{R}^{N^2}$ a neighbourhood of $\boldsymbol{y}^*$ and $S_{\boldsymbol{x}^*} \subset \mathbb{R}^N$ a neighbourhood of $\boldsymbol{x}^*$ such that for any initial point $(\boldsymbol{y}_1, \boldsymbol{x}_1) \in S_{\boldsymbol{y}^*} \times S_{\boldsymbol{x}^*}$ the sequence $((\boldsymbol{y}_n, \boldsymbol{x}_n))_{n\in\mathbb{N}}$ satisfying $(\boldsymbol{y}_{n+1}, \boldsymbol{x}_{n+1}) = \boldsymbol{g}(\boldsymbol{y}_n, \boldsymbol{x}_n)$ converges to $(\boldsymbol{y}^*, \boldsymbol{x}^*)$. Or equivalently, there exist $S_{\boldsymbol{x}^*} \subset \mathbb{R}^N$ a neighbourhood of $\boldsymbol{x}^*$ and $S_{Y^*} \subset M_{N\times N}(\mathbb{R})$ a neighbourhood of

Y^* such that for any initial state $(\boldsymbol{x}_1, Y_1) \in S_{\boldsymbol{x}^*} \times S_{Y^*}$ the sequence $((\boldsymbol{x}_n, Y_n))_{n\in\mathbb{N}}$ satisfying recurrence Eq. (2.2) converges to $(\boldsymbol{x}^*, Y^*)$.

Let $\boldsymbol{x}_1 \in S_{\boldsymbol{x}^*}$, $Y_1 \in S_{Y^*}$. Now we show that $(\boldsymbol{x}_n)_{n\in\mathbb{N}}$ converges to $\boldsymbol{x}^*$ at least quadratically. Define

$$\varepsilon_n := \|\boldsymbol{x}^* - \boldsymbol{x}_n\|_\infty,\ \xi_n := \|Y_{n+1} - (\boldsymbol{f}'(\boldsymbol{x}_n))^{-1}\|_\infty,\ a_n := \max\{\varepsilon_n, \xi_n\},\ n \in \mathbb{N}. \quad (2.3)$$

From previous part and continuity of $\boldsymbol{f}'$ we get that $a_n \to 0$. By continuity of $\boldsymbol{f}'$ and $\det(\boldsymbol{f}'(\boldsymbol{x}^*)) \neq 0$, there exists $r \in (0,1)$ such that $\det(\boldsymbol{f}'(\boldsymbol{x})) \neq 0$ for all $\boldsymbol{x} \in \overline{B}(\boldsymbol{x}^*, r)$. By $\boldsymbol{x}_n \to \boldsymbol{x}^*$ there exists $n_0 \in \mathbb{N}$ such that $\boldsymbol{x}_n \in \overline{B}(\boldsymbol{x}^*, r)$ for any $n \geq n_0$. Define

$$A_{\boldsymbol{f}} := \max_{1\leq k,i,j\leq N} \max_{\boldsymbol{x}\in\overline{B}(\boldsymbol{x}^*,r)} \left|\frac{\partial^2 f_k}{\partial x_i \partial x_j}(\boldsymbol{x})\right|,\ C_{\boldsymbol{f}} := \max_{\boldsymbol{x}\in\overline{B}(\boldsymbol{x}^*,r)} \|(\boldsymbol{f}'(\boldsymbol{x}))^{-1}\|_\infty,$$
$$D_{\boldsymbol{f}} := \max_{\boldsymbol{x}\in\overline{B}(\boldsymbol{x}^*,r)} \|\boldsymbol{f}'(\boldsymbol{x})\|_\infty.$$

Let $n \geq n_0$. By the Taylor formula (Theorem 2.2) we get

$$\boldsymbol{f}(\boldsymbol{x}_n) = \boldsymbol{f}(\boldsymbol{x}^*) + \boldsymbol{f}'(\boldsymbol{x}^*)(\boldsymbol{x}_n - \boldsymbol{x}^*) + \boldsymbol{R}_1(\boldsymbol{x}_n) = \boldsymbol{f}'(\boldsymbol{x}^*)(\boldsymbol{x}_n - \boldsymbol{x}^*) + \boldsymbol{R}_1(\boldsymbol{x}_n)$$

with

$$\boldsymbol{R}_1(\boldsymbol{x}_n) \leq \frac{1}{2} N^2 A_{\boldsymbol{f}} \|\boldsymbol{x}^* - \boldsymbol{x}_n\|_\infty^2. \quad (2.4)$$

Moreover, we have

$$\begin{aligned}
\|\boldsymbol{f}(\boldsymbol{x}_n)\|_\infty &\leq \|\boldsymbol{f}'(\boldsymbol{x}^*)\|_\infty \|\boldsymbol{x}_n - \boldsymbol{x}^*\|_\infty + \frac{1}{2} N^2 A_{\boldsymbol{f}} \|\boldsymbol{x}^* - \boldsymbol{x}_n\|_\infty^2 \\
&\leq \|\boldsymbol{f}'(\boldsymbol{x}^*)\|_\infty a_n + \frac{1}{2} N^2 A_{\boldsymbol{f}} \|\boldsymbol{x}^* - \boldsymbol{x}_n\|_\infty r \\
&\leq \left(\|\boldsymbol{f}'(\boldsymbol{x}^*)\|_\infty + \frac{1}{2} N^2 A_{\boldsymbol{f}}\right) a_n \\
&= D a_n, \qquad (2.5)
\end{aligned}$$

where $D := \|\boldsymbol{f}'(\boldsymbol{x}^*)\|_\infty + \frac{1}{2} N^2 A_{\boldsymbol{f}}$. Next from the second equation of system Eq. (2.2)

$$\begin{aligned}
\boldsymbol{x}^* - \boldsymbol{x}_{n+1} &= \boldsymbol{x}^* - \boldsymbol{x}_n + Y_{n+1}\boldsymbol{f}(\boldsymbol{x}_n) \\
&= (\boldsymbol{x}^* - \boldsymbol{x}_n + (\boldsymbol{f}'(\boldsymbol{x}_n))^{-1}\boldsymbol{f}(\boldsymbol{x}_n)) \\
&\quad + (Y_{n+1} - (\boldsymbol{f}'(\boldsymbol{x}_n))^{-1})\boldsymbol{f}(\boldsymbol{x}_n). \qquad (2.6)
\end{aligned}$$

By Taylor's formula (Theorem 2.2) we get

$$\boldsymbol{0}_N = \boldsymbol{f}(\boldsymbol{x}^*) = \boldsymbol{f}(\boldsymbol{x}_n) + \boldsymbol{f}'(\boldsymbol{x}_n)(\boldsymbol{x}^* - \boldsymbol{x}_n) + \boldsymbol{R}_1(\boldsymbol{x}^*)$$

hence

$$\boldsymbol{x}^* - \boldsymbol{x}_n + (\boldsymbol{f}'(\boldsymbol{x}_n))^{-1}\boldsymbol{f}(\boldsymbol{x}_n) = -(\boldsymbol{f}'(\boldsymbol{x}_n))^{-1}\boldsymbol{R}_1(\boldsymbol{x}^*)$$

and from Eq. (2.1)

$$\|\boldsymbol{x}^* - \boldsymbol{x}_n + (\boldsymbol{f}'(\boldsymbol{x}_n))^{-1}\boldsymbol{f}(\boldsymbol{x}_n)\|_\infty \leq \frac{1}{2} C_{\boldsymbol{f}} N^2 A_{\boldsymbol{f}} \|\boldsymbol{x}^* - \boldsymbol{x}_n\|_\infty^2.$$

Putting the above to Eq. (2.6) and from Eq. (2.5) we get

$$\|\boldsymbol{x}^*-\boldsymbol{x}_{n+1}\|_\infty \le \frac{1}{2}C_{\boldsymbol{f}}N^2A_{\boldsymbol{f}}\|\boldsymbol{x}^*-\boldsymbol{x}_n\|_\infty^2+a_n\|\boldsymbol{f}(\boldsymbol{x}_n)\|_\infty \le (\frac{1}{2}C_{\boldsymbol{f}}N^2A_{\boldsymbol{f}}+D)a_n^2=Ea_n^2, \tag{2.7}$$

where $E := \frac{1}{2}C_{\boldsymbol{f}}N^2A_{\boldsymbol{f}} + D$. Note that by the first equation in system Eq. (2.2)

$$\begin{aligned}(I - \boldsymbol{f}'(\boldsymbol{x}_{n+1})Y_{n+1})^2 &= I - \boldsymbol{f}'(\boldsymbol{x}_{n+1})2Y_{n+1} + \boldsymbol{f}'(\boldsymbol{x}_{n+1})Y_{n+1}\boldsymbol{f}'(\boldsymbol{x}_{n+1})Y_{n+1}\\ &= I - \boldsymbol{f}'(\boldsymbol{x}_{n+1})Y_{n+2}.\end{aligned}$$

Multiplying by $(\boldsymbol{f}'(\boldsymbol{x}_{n+1}))^{-1}$ we have

$$\begin{aligned}(\boldsymbol{f}'(\boldsymbol{x}_{n+1}))^{-1} - Y_{n+2} &= (\boldsymbol{f}'(\boldsymbol{x}_{n+1}))^{-1}(I - \boldsymbol{f}'(\boldsymbol{x}_{n+1})Y_{n+1})^2\\ &= \left((\boldsymbol{f}'(\boldsymbol{x}_{n+1}))^{-1} - Y_{n+1}\right)\boldsymbol{f}'(\boldsymbol{x}_{n+1})\left((\boldsymbol{f}'(\boldsymbol{x}_{n+1}))^{-1} - Y_{n+1}\right).\end{aligned}$$

This implies that

$$\|(\boldsymbol{f}'(\boldsymbol{x}_{n+1}))^{-1} - Y_{n+2}\|_\infty \le D_{\boldsymbol{f}}\|(\boldsymbol{f}'(\boldsymbol{x}_{n+1}))^{-1} - Y_{n+1}\|_\infty^2. \tag{2.8}$$

Because $\boldsymbol{f}$ is a C^2 class, then $\boldsymbol{f}'$ is locally Lipschitz and there exist $n_1 \in \mathbb{N}$, $n_1 \ge n_0$ and $F > 0$ such

$$\|\boldsymbol{f}'(\boldsymbol{x}_n) - \boldsymbol{f}'(\boldsymbol{x}^*)\|_\infty \le F\|\boldsymbol{x}^* - \boldsymbol{x}_n\|_\infty$$

for $n \ge n_1$. From the above and Eq. (2.7) we get

$$\begin{aligned}&\|(\boldsymbol{f}'(\boldsymbol{x}_{n+1}))^{-1} - Y_{n+1}\|_\infty\\ &\le \|(\boldsymbol{f}'(\boldsymbol{x}_n))^{-1} - Y_{n+1}\|_\infty + \|(\boldsymbol{f}'(\boldsymbol{x}_{n+1}))^{-1} - (\boldsymbol{f}'(\boldsymbol{x}_n))^{-1}\|_\infty\\ &\le a_n + \|(\boldsymbol{f}'(\boldsymbol{x}_{n+1}))^{-1}\left(\boldsymbol{f}'(\boldsymbol{x}_n) - \boldsymbol{f}'(\boldsymbol{x}_{n+1})\right)(\boldsymbol{f}'(\boldsymbol{x}_n))^{-1}\|_\infty\\ &\le a_n + C_{\boldsymbol{f}}^2\|\boldsymbol{f}'(\boldsymbol{x}_n) - \boldsymbol{f}'(\boldsymbol{x}_{n+1})\|_\infty\\ &\le a_n + C_{\boldsymbol{f}}^2\|\boldsymbol{f}'(\boldsymbol{x}^*) - \boldsymbol{f}'(\boldsymbol{x}_{n+1})\|_\infty + C_{\boldsymbol{f}}^2\|\boldsymbol{f}'(\boldsymbol{x}_n) - \boldsymbol{f}'(\boldsymbol{x}^*)\|_\infty\\ &\le a_n + C_{\boldsymbol{f}}^2F(Ea_n^2 + a_n)\\ &= (1 + C_{\boldsymbol{f}}^2F(Ea_n + 1))a_n,\end{aligned}$$

for $n \ge n_1$. $a_n \to 0$ hence there exists $n_2 \ge n_1$ such that $0 < a_n < 1$ for $n \ge n_2$. From the above and Eq. (2.8) we get

$$\|(\boldsymbol{f}'(\boldsymbol{x}_{n+1}))^{-1} - Y_{n+2}\|_\infty \le D_{\boldsymbol{f}}(1 + C_{\boldsymbol{f}}^2F(E + 1))^2a_n^2$$

for $n \ge n_2$. From this and Eq. (2.7) we get

$$\begin{aligned}a_{n+1} &= \max\{\|\boldsymbol{x}^* - \boldsymbol{x}_{n+1}\|_\infty, \|(\boldsymbol{f}'(\boldsymbol{x}_{n+1}))^{-1} - Y_{n+2}\|_\infty\}\\ &\le \max\{E, D_{\boldsymbol{f}}(1 + C_{\boldsymbol{f}}^2F(E + 1))^2\}a_n^2\end{aligned} \tag{2.9}$$

for $n \ge n_2$. We define $C := \max\{E, D_{\boldsymbol{f}}(1 + C_{\boldsymbol{f}}^2F(E + 1))^2\}$. From the fact that a_n is convergent to 0 we get the existence of numbers $n_3 \in \mathbb{N}$, $n_3 \ge n_2$ and $a > 0$ such that $0 \le a_n < a < \frac{1}{C}$ for $n \ge n_3$. By Eq. (2.9) and the induction we have that

$$a_n \le \frac{1}{C}(aC)^{2^{(n-n_3)}}$$

for $n \ge n_3$. Put $b_n := \frac{1}{C}(aC)^{2^{(n-n_3)}}$ for $n \ge n_3$. Then $a_n \le b_n$, $b_n > 0$ for $n \ge n_3$, $\lim\limits_{n\to\infty} b_n = 0$ as $Ca < 1$ and $\lim\limits_{n\to\infty} \frac{b_{n+1}}{b_n^2} = C > 0$ which implies that $(\boldsymbol{x}_n)_{n\in\mathbb{N}}$ is convergent to $\boldsymbol{x}^*$ with at least quadratic order. □

Remark 2.1. The proof of the above theorem shows that inverse-free Newton's method as Newton's method is sensitive to the starting point. Moreover, we can see the strong dependence of the modified method on $\|\boldsymbol{f}(\boldsymbol{x}^*)\|_\infty$ and $\|H\boldsymbol{f}(\boldsymbol{x}^*)\|_\infty$ (where $H\boldsymbol{f}(\boldsymbol{x}^*)$ denotes Hessian matrix of $\boldsymbol{f}$ at $\boldsymbol{x}^*$), in this sense that $\|\boldsymbol{f}(\boldsymbol{x}^*)\|_\infty$ can not be "too small" and $H\boldsymbol{f}(\boldsymbol{x}^*)$ can't be "too large".

3. Comparison of classical and inverse-free Newton's methods

Implementations of both algorithms were written in the Python programming language using numpy library. All calculations were done on an Intel(R) Core(TM) i7-7700HQ processor @ 2.80GHz.

In numerical experiments of inverse-free Newton's method we considered initial states of the form $(\boldsymbol{x}_1, (\boldsymbol{f}'(\boldsymbol{x}_1))^{-1})$, where $\boldsymbol{x}_1 \in \mathbb{R}^N$. In this case, we will only give the point $\boldsymbol{x}_1$ and refer to it as the initial point of the method. In the implementation of classical Newton's method, we solve a system of linear equations instead of using inversion of a matrix. Those operations are calculated using numpy methods linalg.inv and linalg.solve, which perform them by calling LPACK software library. The LPACK uses the LU decomposition with partial pivoting and row interchanges.

3.1. Comparison in terms of the running time, the number of iterations and the computational order of convergence

To test the time of the modified method against classical Newton's method, we will use a Broyden tridiagonal function taken as problem number 30 from [18], and originally appearing in [4], (see also [23]). Namely, the function $\boldsymbol{f} = (f_1, \ldots, f_N) : \mathbb{R}^N \to \mathbb{R}^N$ defined by

$$f_i(\boldsymbol{x}) = (3 - 2x_i)x_i - x_{i-1} - 2x_{i+1} + 1, \quad \text{with } x_0 = x_{N+1} = 0 \tag{3.1}$$

for $i \in \{1, \ldots, N\}, \boldsymbol{x} = (x_1, \ldots, x_N) \in \mathbb{R}^N$. The initial point for both methods is $\boldsymbol{x}_1 = (-1, \ldots, -1) \in \mathbb{R}^N$.

The function $\boldsymbol{f}$ is C^2 class function as its coordinates are polynomials. Note that

$$\frac{\partial f_i}{\partial x_j}(\boldsymbol{x}) = \delta_{ij}(3 - 4x_i) - \delta_{(i-1)j} - 2\delta_{(i+1)j},$$

for $i, j \in \{1, \ldots, N\}$ and $\boldsymbol{x} \in \mathbb{R}^N$, where δ_{ij} denotes the Kronecker delta. For complicated functions, the assumption of non-zeroing of the Jacobian at the point $\boldsymbol{x}^*$, is difficult to check when we do not know $\boldsymbol{x}^*$, so in practice this issue is ignored. Here, we will check this assumption numerically, more precisely after obtaining the approximation of $\boldsymbol{x}^*$ we check that $\det(\boldsymbol{f}'(\boldsymbol{x}^*)) \neq 0$. If this condition holds together with $\boldsymbol{f} \in C^2$, then for a sufficiently small distance of the initial point $\boldsymbol{x}_1$ from $\boldsymbol{x}^*$ we will have at least quadratic convergence of both methods.

We test both methods on the function $\boldsymbol{f}$ for $N = 3$, 10, 20, 30, 100, 200, 300, 500. For each value of N we run both methods using the same starting point $\boldsymbol{x}_1 = (-1, \ldots, -1) \in \mathbb{R}^N$ and as stopping condition we use the distance between the value of the function and zero which is at most $\gamma = 10^{-8}$ and we call it accuracy level. In addition, in order to evaluate the modified method, we calculate the zero

from classical Newton's method with the greater level of accuracy and we call such a point the exact zero.

In Table 1 we provide the obtained results, which include: the number of iterations, the final distance from the exact zero, the final distance of the function value from zero, the time (average) and how many times modified Newton's method is faster than classical one, i.e. the ratio of the running time of classical Newton's method to that of inverse-free Newton's method. In addition, we provide the rank of the Jacobi matrix at the exact zero. Each problem was run 50 times and results were averaged to produce times shown in Table 1. To measure the execution time we use the function perf_counter_ns() from the Python's built-in time library.

Table 1. The results of the execution time with the accuracy level $\gamma = 10^{-8}$ of both methods for the function $\boldsymbol{f}$ given by Eq. (3.1) for different N. The columns use the labels: k- number of iterations to achieve accuracy, $\boldsymbol{x}^*$ - zero of the function $\boldsymbol{f}$, $\boldsymbol{x}_k$ - k-th approximation of the given method, r_N - the rank of the Jacobi matrix $\boldsymbol{f}'(\boldsymbol{x}^*)$. Moreover, N. denotes the classical Newton's method, N. mod. denotes the modified Newton's method, impr. denotes improvement.

N	r_N	method	iter.(k)	$\|\boldsymbol{f}(\boldsymbol{x}_k)\|_\infty$	$\|\boldsymbol{x}_k - \boldsymbol{x}^*\|_\infty$	time[sec]	impr.
3	3	N.	4	1.85e-09	5.77e-10	0.00038	0.6
		N. mod.	5	1.90e-10	6.58e-11	0.00064	
10	10	N.	4	7.55e-10	2.41e-10	0.00040	0.6
		N. mod.	5	6.46e-11	1.91e-11	0.00073	
20	20	N.	4	7.55e-10	2.41e-10	0.00155	1.6
		N. mod.	5	6.46e-11	1.91e-11	0.00096	
30	30	N.	4	7.55e-10	2.41e-10	0.00173	2.0
		N. mod.	5	6.46e-11	1.91e-11	0.00087	
100	100	N.	4	7.55e-10	2.41e-10	0.01427	2.7
		N. mod.	5	6.46e-11	1.91e-11	0.00538	
200	200	N.	4	7.55e-10	2.41e-10	0.02084	2.3
		N. mod.	5	6.46e-11	1.91e-11	0.00906	
300	300	N.	4	7.55e-10	2.41e-10	0.03741	1.0
		N. mod.	5	6.46e-11	1.91e-11	0.03862	
500	500	N.	4	7.55e-10	2.41e-10	0.07029	0.3
		N. mod.	5	6.46e-11	1.91e-11	0.21799	

In terms of real time performance the modified method is faster for $N = 20$, 30, 100, 200, and slower for $N = 3$, 10, 300, 500. In the best case of $N = 100$ inverse-free Newton's method works 2.7 times faster, while in the others, i.e. for $N = 20$, 30, 200, it is at least 1.5 times faster. This means that the method is not optimal when the number of variables N is small ($N < 20$), or large ($N > 200$). This is due to the fact that we need to inverse a matrix once at the beginning and multiply matrices twice at each iteration for the modified method. When these times are large compared to the time of solving systems of linear equations, which classical Newton's method performs once at each iteration, the modified method is unprofitable. Figure 1 shows times of these operations and it can be deduced that for small $N \leq 10$ the large cost of inversion of a matrix for a small number of iterations makes the modified method worthless. In contrast, for large N the unprofitability of Newton's method results from the large computational cost of

multiplication of matrices.

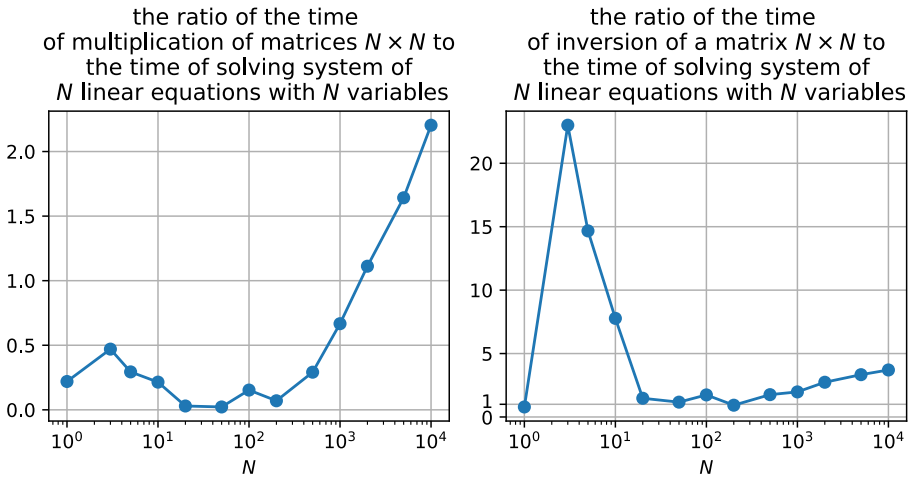

Figure 1. Comparison of the time of inversion and multiplication of matrices $N \times N$ to the time of solving systems of N linear equations with N variables for different values of N.

In terms of the number of iterations, the modified method is, according to Table 1 only slightly worse to classical Newton's method, taking one iteration longer. Furthermore, the calculated rank of the matrix Jacobi at the zero is equal to the dimension of the matrix, so it is nonsingular. This means that assumptions of both methods are satisfied.

Now we calculate the approximate computational order of convergence (ACOC) and the computational order of convergence (COC) for both methods. The approximate computational order of convergence (ACOC) is defined by formula

$$\widehat{p}_n = \frac{\ln(\|\boldsymbol{x}_{n+1} - \boldsymbol{x}_n\|/\|\boldsymbol{x}_n - \boldsymbol{x}_{n-1}\|)}{\ln(\|\boldsymbol{x}_n - \boldsymbol{x}_{n-1}\|/\|\boldsymbol{x}_{n-1} - \boldsymbol{x}_{n-2}\|}.$$

ACOC was introduced in [6]. The computational order of convergence (COC) is given by formula

$$\overline{p}_n = \frac{\ln(\|\boldsymbol{x}_{n+1} - \boldsymbol{x}^*\|/\|\boldsymbol{x}_n - \boldsymbol{x}^*\|)}{\ln(\|\boldsymbol{x}_n - \boldsymbol{x}^*\|/\|\boldsymbol{x}_{n-1} - \boldsymbol{x}^*\|)},$$

see [35].

Both metrics where computed for the last available n and for $\boldsymbol{x}^*$ we use the zero calculated to much greater precision using the Netwon's method. We calculate ACOC and COC for three test functions $\boldsymbol{f}$, $\boldsymbol{f}_T$, $\boldsymbol{f}_B$ (given below), for N =3, 10, 20 30, 100, 200, 300, 500 and given initial point.

Firstly, we test $\boldsymbol{f}$ Broyden tridiagonal function given by Eq. (3.1). In Table 2 we provide the obtained results, which include: the number of variables N, method, ACOC, COC.

Now, we test $\boldsymbol{f}_T$ trigonometric function from [18]. $\boldsymbol{f}_T = (f_1, \ldots, f_N)$ given by

$$f_i(\boldsymbol{x}) = N - \sum_{k=1}^{N} \cos(x_k) + i(1 - \cos(x_i)) - \sin(x_i), \text{ for } i = 1, \ldots, N, \boldsymbol{x} \in \mathbb{R}^N \quad (3.2)$$

Table 2. The results of ACOC, COC of both methods for the function $\boldsymbol{f}$ given by Eq. (3.1) for different N. N. denotes the classical Newton's method, N. mod. denotes the modified Newton's method.

N	method	ACOC	COC
3	N.	1.999	1.511
	N. mod.	1.869	1.940
10	N.	1.929	1.973
	N. mod.	1.829	1.935
20	N.	1.878	1.951
	N. mod.	1.789	1.930
30	N.	1.853	1.933
	N. mod.	1.758	1.926
100	N.	1.827	1.857
	N. mod.	1.657	1.897
200	N.	1.848	1.807
	N. mod.	1.614	1.864
300	N.	1.868	1.780
	N. mod.	1.601	1.838
500	N.	1.896	1.754
	N. mod.	1.599	1.800

with $\boldsymbol{x}_0 = \left(\frac{1}{5N}, \ldots, \frac{1}{5N}\right)$. In Table 3 we provide the obtained results for $\boldsymbol{f}_T$, which include: the number of variables N, method, the number of iterations, the final distance from the exact zero, the final distance of the function value from zero, ACOC, COC and the time (average).

As the last test function, we test $\boldsymbol{f}_B = (f_1, \ldots, f_N)$ Brown almost linear function from [18] given by

$$\begin{aligned} f_i(\boldsymbol{x}) &= x_i + \sum_{j=1}^{N} x_j - N - 1, \quad \text{for } i = 1, \ldots, N-1, \\ f_N(\boldsymbol{x}) &= x_1 \cdot \ldots \cdot x_N - 1 \end{aligned} \tag{3.3}$$

for $\boldsymbol{x} = (x_1, \ldots, x_N) \in \mathbb{R}^N$ with $\boldsymbol{x}_0 = \left(1 - \frac{1}{N^2}, \ldots, 1 - \frac{1}{N^2}\right)$.

In Table 4 we provide the obtained results for $\boldsymbol{f}_B$, which include: the number of variables N, method, the number of iterations, the final distance of the function value from zero, the final distance from the exact zero, ACOC, COC and the time (average).

In terms of ACOC and COC both methods work similarly. For $\boldsymbol{f}_B$ and $\boldsymbol{f}_T$ inverse-free Newton's method is slightly better than classical one.

3.2. Comparison in terms of basins of attraction

We investigate basins of attraction using the test function, which was used in the [8] or [23] to test the case when the function has a zero at which Jacobian is close to zero. The test function $\boldsymbol{f}_\varepsilon = (f_{\varepsilon,1}, f_{\varepsilon,2}) : \mathbb{R}^2 \to \mathbb{R}^2$ is parametrized by $\varepsilon \geqslant 0$ and is

Table 3. The results of ACOC, COC and the execution time of both methods for the function $\boldsymbol{f}_T$ given by Eq. (3.2) for different N. N. denotes the classical Newton's method, N. mod. denotes the modified Newton's method.

N	method	iter.(k)	$\|\boldsymbol{f}_T(\boldsymbol{x}_k)\|_\infty$	$\|\boldsymbol{x}_k - \boldsymbol{x}^*\|_\infty$	ACOC	COC	time[s]
3	N.	5	1.29e-11	1.29e-11	1.979	1.992	0.00042
	N. mod.	7	1.43e-10	1.43e-10	1.980	1.961	0.00074
10	N.	5	1.81e-12	1.81e-12	1.977	1.987	0.00045
	N. mod.	7	1.00e-13	1.00e-13	1.956	1.970	0.00096
20	N.	5	7.70e-13	7.70e-13	1.977	1.989	0.00111
	N. mod.	7	1.39e-14	1.39e-14	1.954	1.964	0.00098
30	N.	5	4.85e-13	4.85e-13	1.977	1.992	0.00175
	N. mod.	7	5.92e-15	5.92e-15	1.954	1.965	0.00118
100	N.	5	1.35e-13	1.35e-13	1.977	1.992	0.12347
	N. mod.	6	9.78e-10	9.78e-10	2.021	1.940	0.02598
200	N.	5	9.08e-14	9.08e-14	1.978	1.925	0.02471
	N. mod.	6	4.58e-10	4.58e-10	2.019	1.940	0.01998
300	N.	5	5.64e-14	5.64e-14	1.978	1.920	0.02695
	N. mod.	6	2.99e-10	2.99e-10	2.019	1.941	0.03838
500	N.	5	5.36e-14	5.36e-14	1.978	1.881	0.05188
	N. mod.	6	1.76e-10	1.76e-10	2.018	1.941	0.09280

Table 4. The results of ACOC, COC and the execution time of both methods for the function $\boldsymbol{f}_B$ given by Eq. (3.3) for different N. N. denotes the classical Newton's method, N. mod. denotes the modified Newton's method.

N	method	iter.(k)	$\|\boldsymbol{f}_B(\boldsymbol{x}_k)\|_\infty$	$\|\boldsymbol{x}_k - \boldsymbol{x}^*\|_\infty$	ACOC	COC	time[s]
3	N.	6	1.40e-10	4.21e-10	2.003	1.998	0.00100
	N. mod.	9	1.24e-13	3.71e-13	1.968	1.975	0.00171
10	N.	6	3.10e-10	3.10e-09	1.999	1.995	0.00108
	N. mod.	8	4.84e-11	4.84e-10	2.011	1.960	0.00172
20	N.	6	1.02e-10	2.04e-09	1.995	1.995	0.00165
	N. mod.	8	1.11e-13	2.12e-12	1.977	1.982	0.00151
30	N.	6	4.82e-11	1.45e-09	1.993	1.995	0.00300
	N. mod.	7	8.74e-10	2.62e-08	2.119	1.943	0.00189
100	N.	6	4.59e-12	4.56e-10	1.990	1.996	0.01098
	N. mod.	7	1.68e-11	1.68e-09	2.071	1.949	0.00835
200	N.	6	1.17e-12	2.24e-10	1.990	2.002	0.02194
	N. mod.	7	2.92e-12	5.72e-10	2.062	1.956	0.03105
300	N.	6	5.07e-13	1.50e-10	1.989	2.000	0.03254
	N. mod.	7	1.11e-12	3.67e-10	2.059	1.932	0.04796
500	N.	6	2.84e-13	1.28e-11	1.989	2.444	0.44561
	N. mod.	6	4.06e-10	2.03e-07	2.553	1.918	0.16561

defined as follows:

$$\begin{aligned} f_{\varepsilon,1}(x_1,x_2) &= (x_1-1)+(x_2-3)^2, \\ f_{\varepsilon,2}(x_1,x_2) &= \varepsilon(x_2-3)+\frac{3}{2}(x_1-1)(x_2-3)+(x_2-3)^2+(x_2-3)^3 \end{aligned} \tag{3.4}$$

for $\boldsymbol{x}=(x_1,x_2)\in\mathbb{R}^2$. Zeros of the function $\boldsymbol{f}_\varepsilon$ are:

$$\boldsymbol{x}_0^*=(1,3),\quad \boldsymbol{x}_-^*=(1-\eta_-^2,3+\eta_-),\quad \boldsymbol{x}_+^*=(1-\eta_+^2,3+\eta_+), \tag{3.5}$$

where $\eta_+=1+\sqrt{1+2\varepsilon}$, $\eta_-=1-\sqrt{1+2\varepsilon}$. Indeed, to solve $\boldsymbol{f}_\varepsilon(x_1,x_2)=(0,0)$ we use substitution $\eta=x_2-3$, hence $f_{\varepsilon,1}(x_1,x_2)=0$, which implies that $x_1-1=-\eta^2$. Next from $\boldsymbol{f}_{\varepsilon,2}(x_1,x_2)=0$ we get $0=\varepsilon\eta-\frac{3}{2}\eta^3+\eta^2+\eta^3$, and hence

$$\eta=0\quad\vee\quad\eta=1+\sqrt{1+2\varepsilon}\quad\vee\quad\eta=1-\sqrt{1+2\varepsilon}.$$

$\boldsymbol{f}_\varepsilon$ is C^2 class as its coordinates are polynomials for $\varepsilon\geq 0$. The Jacobi matrix of the function $\boldsymbol{f}_\varepsilon$ at $\boldsymbol{x}=(x_1,x_2)\in\mathbb{R}^2$ is equal to

$$\boldsymbol{f}_\varepsilon'(x_1,x_2)=\begin{bmatrix} 1 & 2(x_2-3) \\ \frac{3}{2}(x_2-3) & \frac{3}{2}(x_1-1)+2(x_2-3)+3(x_2-3)^2+\varepsilon \end{bmatrix}.$$

Then

$$|\eta_-|=\frac{2\varepsilon}{1+\sqrt{1+2\varepsilon}}\leq\frac{2\varepsilon}{2}=\varepsilon,\quad |\eta_+|=\sqrt{1+2\varepsilon}+1\geq 2,\quad \det(\boldsymbol{f}_\varepsilon'(1,3))=\varepsilon>0.$$

For $\varepsilon\in(0,1]$ we have $\|\boldsymbol{x}_0^*-\boldsymbol{x}_-^*\|_\infty=\max\{\eta_-^2,|\eta_-|\}\leqslant\max\{\varepsilon^2,\varepsilon\}\leqslant\varepsilon$, and $\|\boldsymbol{x}_0^*-\boldsymbol{x}_+^*\|_\infty=\max\{\eta_+^2,|\eta_+|\}\geqslant\max\{4,2\}=4$. It means that zeros $\boldsymbol{x}_0^*$ and $\boldsymbol{x}_-^*$ are closer to each other as $\varepsilon\to 0^+$, and are equal when $\varepsilon=0$. On the other hand distance between $\boldsymbol{x}_0^*$, $\boldsymbol{x}_+^*$ is at least 4.

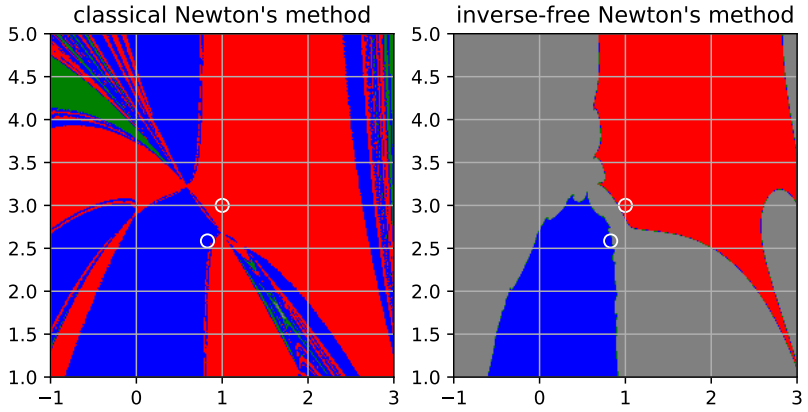

Figure 2. Basins of attraction for the function $\boldsymbol{f}_\varepsilon$ from Eq. (3.4) with $\varepsilon=\frac{1}{2}$. In grey, red, blue and green we mark respectively: divergence, convergence to $\boldsymbol{x}_0^*$, convergence to $\boldsymbol{x}_-^*$ and convergence to $\boldsymbol{x}_+^*$. $\boldsymbol{x}_0^*$, $\boldsymbol{x}_-^*$ are marked by white circle.

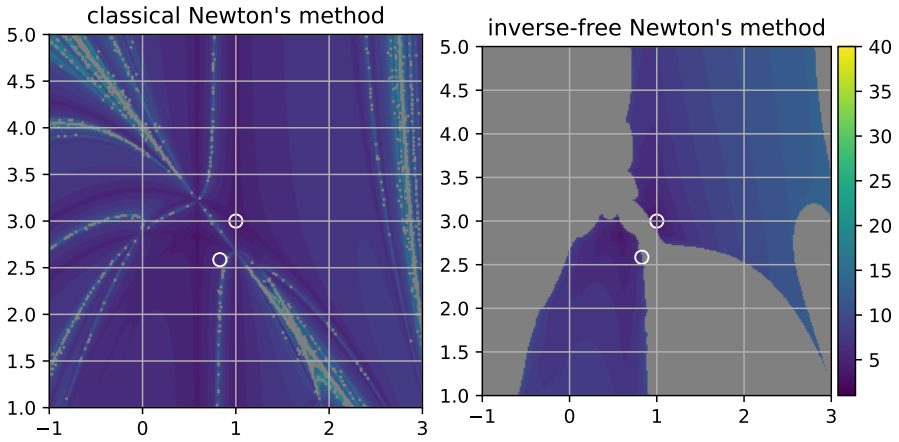

Figure 3. Basins of attraction for the function $\boldsymbol{f}_\varepsilon$ from Eq. (3.4) with $\varepsilon = \frac{1}{2}$. In grey we mark divergence, colours from strong dark blue to dark blue indicate the number of iterations needed to reach the accuracy level $\gamma = 10^{-10}$ to $\boldsymbol{x}_0^*$ or $\boldsymbol{x}_-^*$ or $\boldsymbol{x}_+^*$. $\boldsymbol{x}_0^*$, $\boldsymbol{x}_-^*$ are marked by white circle.

Let us draw graphs of basins of attraction of both methods for $\varepsilon = 2^{-1}$, 10^{-1}, 10^{-3} on the set $[-1,3] \times [1,5]$. To produce graphs, we choose $1000 \cdot 1000$ equally spaced starting points over the square $[-1,3] \times [1,5]$ and run both methods for each with accuracy level of $\gamma = 10^{-20}$, see [23]. In grey, we mark initial points for which the method diverges, i.e. the case when the method needs more than 100 iterations or the condition $\|\boldsymbol{x_n}\|_\infty > 10^{10}$ occurs. We detect a convergence to a point $\boldsymbol{x}_0^*$, $\boldsymbol{x}_-^*$ and $\boldsymbol{x}_+^*$ when the method returns a point at most 10^{-9} away from it with respect to the norm $\|\cdot\|_\infty$. In addition, we use white circles to mark zeros $\boldsymbol{x}_0^*$ and $\boldsymbol{x}_-^*$ (the point $\boldsymbol{x}_+^*$ was not marked because $\boldsymbol{x}_+^* \notin [-1,3] \times [1,5]$). Graphs created in this way are presented in Figures 2, 4 and 6.

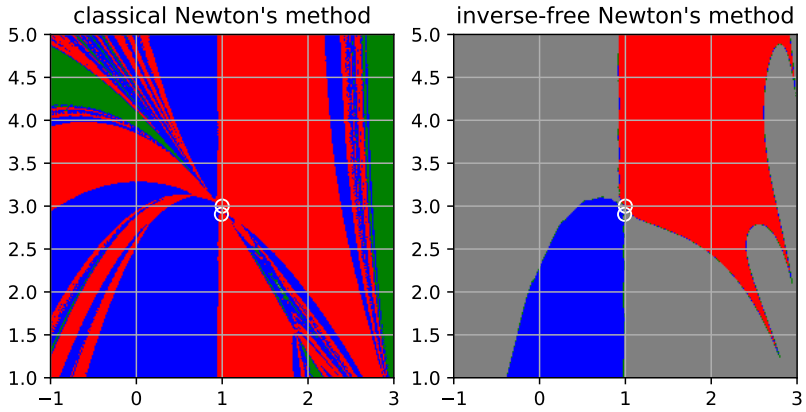

Figure 4. Basins of attraction for the function $\boldsymbol{f}_\varepsilon$ from Eq. (3.4) with $\varepsilon = 10^{-1}$. In grey, red, blue and green we mark respectively: divergence, convergence to $\boldsymbol{x}_0^*$, convergence to $\boldsymbol{x}_-^*$ and convergence to $\boldsymbol{x}_+^*$. $\boldsymbol{x}_0^*$, $\boldsymbol{x}_-^*$ are marked by white circle.

Moreover, we draw graphs of the speed of convergence of both methods for $\varepsilon = 2^{-1}$, 10^{-1}, 10^{-3} on the set $[-1,3] \times [1,5]$. To produce graphs, we choose $1000 \cdot 1000$

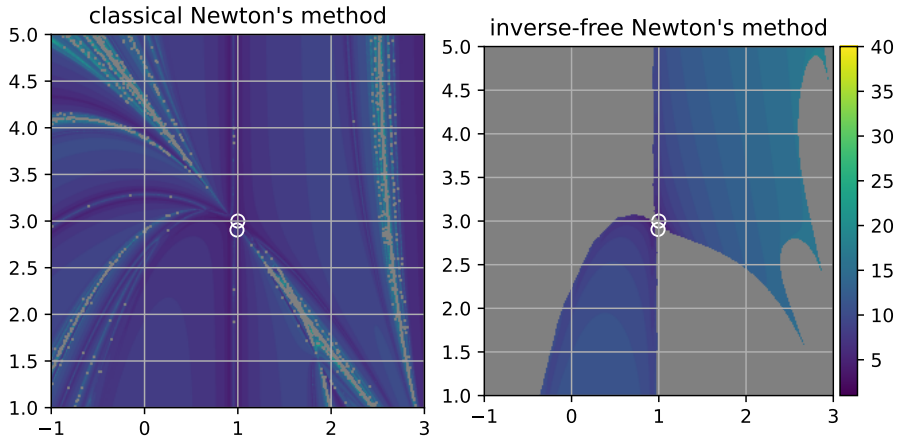

Figure 5. Basins of attraction for the function $\boldsymbol{f}_\varepsilon$ from blue Eq. (3.4) with $\varepsilon = 0, 1$. In grey we mark divergence, colours from strong dark blue to dark blue indicate the number of iterations needed to reach the accuracy level $\gamma = 10^{-10}$ to $\boldsymbol{x}_0^*$ or $\boldsymbol{x}_-^*$ or convergence to $\boldsymbol{x}_+^*$. $\boldsymbol{x}_0^*$, $\boldsymbol{x}_-^*$ are marked by white circle.

equally spaced starting points over the square $[-1, 3] \times [1, 5]$ run both methods for each with accuracy level of $\gamma = 10^{-10}$. In grey, we mark initial points for which the method diverges. Colours from strong dark blue to yellow indicate the number of iterations needed to reach the accuracy level γ to $\boldsymbol{x}_0^*$, $\boldsymbol{x}_+^*$, $\boldsymbol{x}_-^*$, where $\boldsymbol{x}_0^*, \boldsymbol{x}_-^*, \boldsymbol{x}_+^*$ are zeros of the function $\boldsymbol{f}_\varepsilon$ defined in Eq. (3.5). Graphs created in this way are presented in Figures 3, 5 and 7.

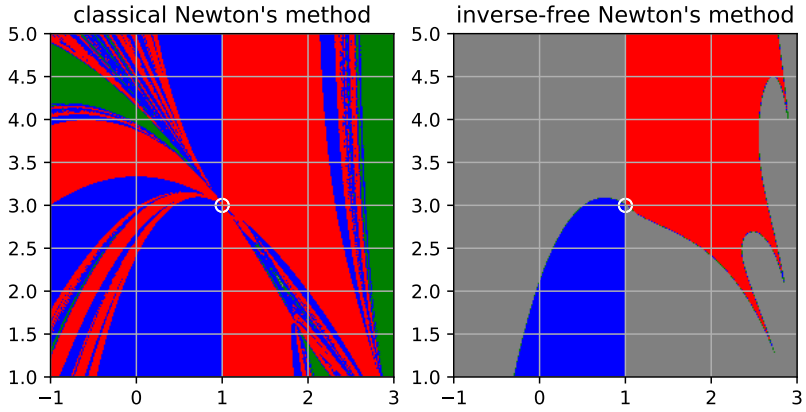

Figure 6. Basins of attraction for the function $\boldsymbol{f}_\varepsilon$ from Eq. (3.4) with $\varepsilon = 10^{-3}$. In grey, red, blue and green we mark respectively: divergence, convergence to $\boldsymbol{x}_0^*$, convergence to $\boldsymbol{x}_-^*$ and convergence to $\boldsymbol{x}_+^*$. $\boldsymbol{x}_0^*$, $\boldsymbol{x}_-^*$ are marked by white circle.

It can be seen from figures that basins of attraction of methods differ significantly. At most of starting points where classical Newton's method is convergent, modified Newton's method did not converge. Furthermore, even when the initial point is close to the zero, inverse-free Newton's method may still not converge to any of them and this effect is more pronounced, when ε is smaller. Furthermore, it can be seen that if inverse-free Newton's method is convergent to some point,

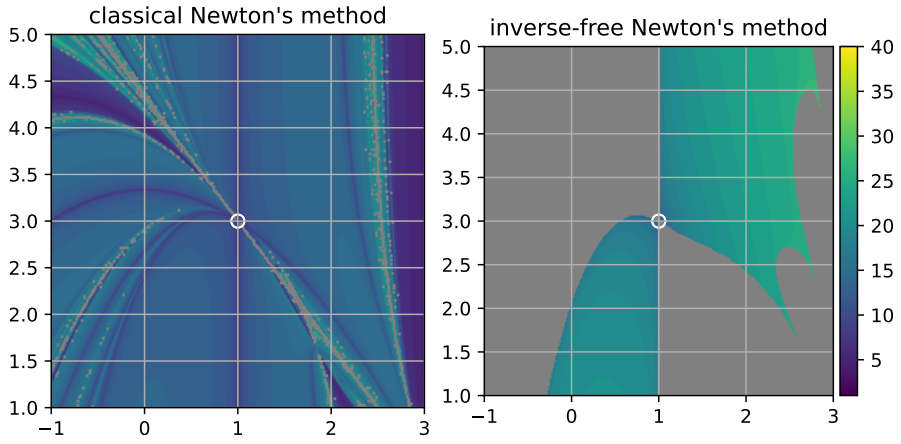

Figure 7. Basins of attraction for the function $\boldsymbol{f}_\varepsilon$ from blue Eq. (3.4) with $\varepsilon = 10^{-3}$. In grey we mark divergence, colours from blue to green indicate the number of iterations needed to reach the accuracy level $\gamma = 10^{-10}$ to $\boldsymbol{x}_0^*$ or $\boldsymbol{x}_-^*$ or $\boldsymbol{x}_+^*$. $\boldsymbol{x}_0^*$, $\boldsymbol{x}_-^*$ are marked by white circle.

then usually classical Newton's method converges to this point. In terms of speed of convergence if $\varepsilon = 2^{-1}$ both methods need almost the same number of iterations to reach give accuracy. For $\varepsilon = 10^{-3}$ inverse-free Newton's method need about 10 iterations more then classical method. Inverse-free Newton's method needs 35 iterations in the worst case for $\varepsilon = 10^{-3}$.

4. Potential improvements

In order to improve on inverse-free Newton's method, we propose two possible modifications. The first one consists of a different choice of the initial matrix Y_1, and the second one of a more efficient computation of the last iterations.

In the first modification instead of $(\boldsymbol{f}'(\boldsymbol{x}_1))^{-1}$ as the initial matrix we will choose

$$Y_1 = \frac{\boldsymbol{f}'(\boldsymbol{x}_1)^T}{\|\boldsymbol{f}'(\boldsymbol{x}_1)\|_1 \|\boldsymbol{f}'(\boldsymbol{x}^{(1)})\|_\infty}, \tag{4.1}$$

where $\|\cdot\|_1$ is defined as $\|B\|_1 := \max_{1\leqslant j\leqslant n} \sum_{i=1}^n |b_{ij}|$ for $B = [b_{ij}] \in M_{N\times N}(\mathbb{R})$, $\boldsymbol{f}'(\boldsymbol{x}_1)^T$ denotes transposition of $\boldsymbol{f}'(\boldsymbol{x}_1)$. This is motivated by paper [21], in which the recurrence below on the sequence of matrices $(Z_n)_{n\in\mathbb{N}} \subset M_{N\times N}(\mathbb{R})$ was considered

$$\begin{cases} Z_{n+1} = 2Z_n - Z_n B Z_n, & \text{for } n \in \mathbb{N}, \\ Z_1 = \alpha B^T, \end{cases}$$

where $\alpha > 0$ and $B \in M_{N\times N}(\mathbb{R})$ is a nonsingular matrix. Note, the above recurrence equation agrees with the first equation in system Eq. (2.2) in the case when the function $\boldsymbol{f} : \mathbb{R}^N \to \mathbb{R}^N$ is given by $\boldsymbol{f}(\boldsymbol{x}) := B\boldsymbol{x}$ for $\boldsymbol{x} \in \mathbb{R}^N$. From [21] it is known that for sufficient small $\alpha > 0$ the sequence $(Z_n)_{n\in\mathbb{N}}$ is convergent to B^{-1} and the sufficient constant α is

$$\alpha_0 = \frac{1}{\|B\|_1 \|B\|_\infty}.$$

This suggests a choice of the matrix Y_1 according to Eq. (4.1). Then the number of operations needed to compute Y_1 would be of order $O(N^2)$, which is better than calculating the inverse of the matrix $\boldsymbol{f}'(\boldsymbol{x}_1)$. Numerical experiments show, that this choice of matrix Y_1 is also good, in terms of basins of attraction. For example, for the problem in Section 3.2 with this modification we obtain basins of attraction shown in Figure 8.

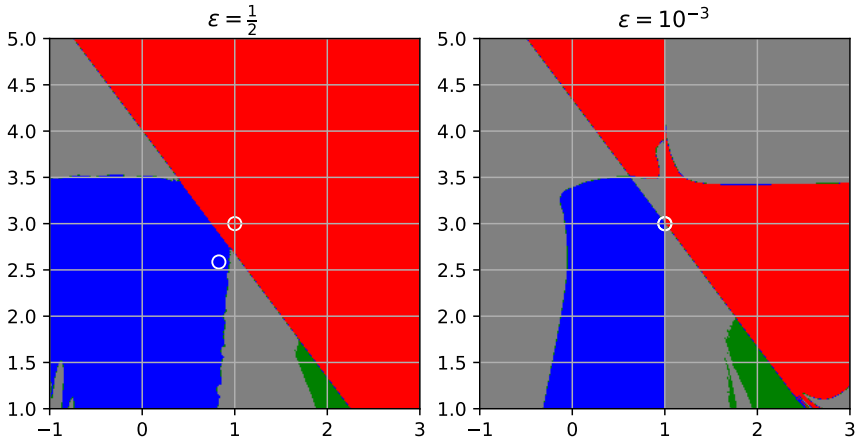

Figure 8. Basins of attraction inverse-free Newton's method with the first modification for the function $\boldsymbol{f}_\varepsilon$ from Eq. (3.4) with $\varepsilon = \frac{1}{2}$ and $\varepsilon = 10^{-3}$. In grey, red, blue and green we mark respectively: divergence, convergence to $\boldsymbol{x}_0^*$, convergence to $\boldsymbol{x}_-^*$ and convergence to $\boldsymbol{x}_+^*$. $\boldsymbol{x}_0^*$, $\boldsymbol{x}_-^*$ are marked by white circle.

In terms of the number of iterations needed, experiments suggest they may increase. See on Figure 9 which presents basins of attraction inverse-free Newton's method for $\boldsymbol{f}_\varepsilon$ for $\varepsilon = \frac{1}{2}$ and $\varepsilon = 10^{-3}$ with a choice of the matrix Y_1 according to Eq. (4.1) with accuracy level $\gamma = 10^{-10}$ coloured according to numbers of performed iterations. For not too small $\varepsilon = \frac{1}{2}$ (which means that $\boldsymbol{f}_{\frac{1}{2}}$ doesn't have singular zero at $(1,3)$, because $\det(\boldsymbol{f}'_{\frac{1}{2}}(1,3)) = \frac{1}{2}$) to reach given accuracy we need not many iterations. Choosing starting point for $B((1,3),\frac{1}{10})$ we need only 5-10 iterations to achieve 10^{-10}. Otherwise when, $\varepsilon = 10^{-3}$ which means that $(1,3)$ is almost singular point to $\boldsymbol{f}_{10^{-3}}$, basin of attraction of (1,3) is not only smaller, but numbers of iterations to reach γ is at least 25.

The second modification is to calculate the last iterations in a different way, to change multiplication of matrices into multiplication of a matrix by a vector. For the sake of clarity, we will explain this procedure when all iterations are computed this way. Let us assume that we have already computed $\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_k$, where $k \in \mathbb{N}$ and we want to calculate $\boldsymbol{x}_{k+1}$. In order to calculate $\boldsymbol{x}_{k+1}$ we need to know only the vector $Y_{k+1}\boldsymbol{f}(\boldsymbol{x}_k)$ and the vector $\boldsymbol{x}_k$. The matrix Y_{k+1}, due to its recursive definition, can be represented as an expression consisting of matrices $\boldsymbol{f}'(\boldsymbol{x}_k), \ldots, \boldsymbol{f}'(\boldsymbol{x}_1), Y_1$. Each of these matrices has been calculated in previous iterations. Then we multiply this expression by vector $\boldsymbol{f}(\boldsymbol{x}_k)$ and we perform multiplications always starting from the vector and, if necessary, using distributivity of multiplication over addition. We will thus calculate the vector $Y_{k+1}\boldsymbol{f}(\boldsymbol{x}_k)$ by performing $2^{k+1} - 1$ multiplications of a matrix by a vector and $2^k - 1$ differences of vectors. Finally, by calculating the difference $\boldsymbol{x}_k - Y_{k+1}\boldsymbol{f}(\boldsymbol{x}_k)$ we obtain $\boldsymbol{x}_{k+1}$.

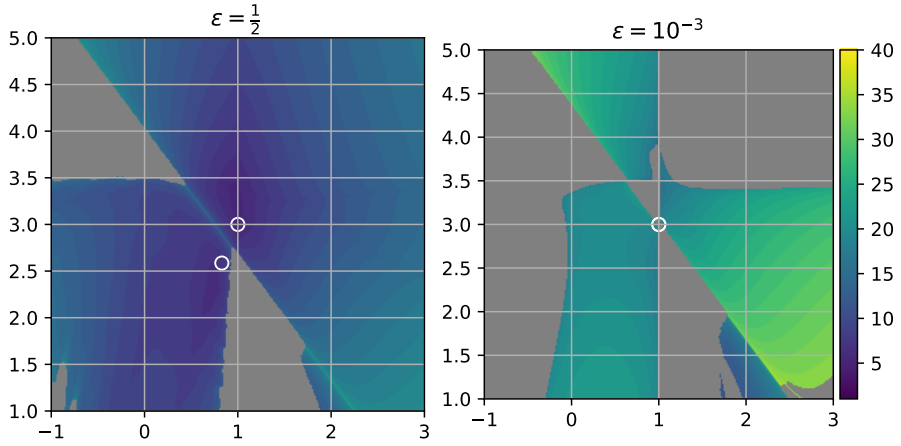

Figure 9. Basins of attraction inverse-free Newton's method with the first modification for the function $\boldsymbol{f}_\varepsilon$ from Eq. (3.4) with $\varepsilon = \frac{1}{2}$ and $\varepsilon = 10^{-3}$. In grey we mark divergence, colours from dark blue to yellow indicate the number of iterations needed to reach the accuracy level $\gamma = 10^{-10}$ to $\boldsymbol{x}_0^*$ or $\boldsymbol{x}_-^*$ or $\boldsymbol{x}_+^*$. $\boldsymbol{x}_0^*$, $\boldsymbol{x}_-^*$ are marked by white circle.

Taking into account that multiplication of a matrix by a vector needs about N times less operations than multiplication of matrices, then for small k the improvement can be obtained. However, as k increases, the computational cost increases very quickly, so it is important to use this method only for small k. In practice, it would be possible to apply it only for the last few iterations, while all the previous iterations are calculated in the standard way. This can be done, because in the presented method it is sufficient to set Y_1 and $\boldsymbol{x}_1$ to the result of the n-th iteration of inverse-free Newton's method.

5. Conclusions

Firstly, we proved that inverse-free Newton's method has the same order of convergence as the classical method under the same assumptions. Moreover, numerical experiments showed that the modified Newton method in comparison to classical Newton's method, is faster if the number of variables in the problem is between 20 and 200, and for a 100 can be even 2.5 times faster. It is worth noting that the running time of methods on the considered problems will depend on the processor used, what algorithms for multiplication of matrices or solving systems of equations were used. In terms of basins of attraction, the modified method performs worse, because it converges less frequently than Newton's method and, in order to guarantee convergence, the starting point must lie closer to the zero than in the classical method.

Acknowledgements

The authors are grateful to the referees for constructive remarks improving the accessibility and potential impact of the paper.

References

[1] I. K. Argyros, S. George, S. Regmi and C. I. Argyros, *Hybrid Newton-like inverse free algorithms for solving nonlinear equations*, Algorithms, 2024, 17(4), 154.

[2] I. K. Argyros, S. George, S. Shakhno, S. Regmi, M. Havdiak and M. I. Argyros, *Asymptotically Newton-type methods without inverses for solving equations*, Mathematics, 2024, 12(7), 1069.

[3] J. D. Blanchard and M. Chamberland, *Newton's method without division*, Amer. Math. Monthly, 2023, 130(7), 606–617.

[4] C. G. Broyden, *A class of methods for solving nonlinear simultaneous equations*, Math. Comput., 1965, 19(92), 577–593.

[5] C. Chun, *Iterative methods improving Newton's method by the decomposition method*, Comput. Math. Appl., 2005, 50(10–12), 1559–1568.

[6] A. Cordero and J. R. Torregrosa, *Variants of Newton's method using fifth-order quadrature formulas*, Appl. Math. Comput., 2007, 190, 686–698.

[7] M. Dallas and S. Pollock, *Newton-Anderson at singular points*, Int. J. Numer. Anal. Model., 2023, 20(5), 667–692.

[8] D. W. Decker and C. T. Kelley, *Expanded convergence domains for Newton's method at nearly singular roots*, SIAM J. Sci. Stat. Comp., 1985, 6(4), 951–966.

[9] A. Griewank, *On solving nonlinear equations with simple singularities or nearly singular solutions*, SIAM Rev., 1985, 27(4), 537–563.

[10] H. H. H. Homeier, *On Newton-type methods with cubic convergence*, J. Comput. Appl. Math., 2005, 176(2), 425–432.

[11] K. Jisheng, L. Yitian and W. Xiuhua, *Third-order modification of Newton's method*, J. Comput. Appl. Math., 2007, 205(1), 1–5.

[12] C. T. Kelley and R. Suresh, *A new acceleration method for Newton's method at singular points*, SIAM J. Numer. Anal., 1983, 20(5), 1001–1009.

[13] W. Kelley and A. Peterson, *Difference Equations, An Introduction with Applications*, 2nd ed., Harcourt/Academic Press, San Diego, 2001.

[14] Y. Levin and A. Ben-Israel, *An inverse-free directional Newton method for solving systems of nonlinear equations*, Progress in Analysis, 2003, 1447–1457.

[15] R. Lin, H. Ren, Q. Wu, Y. Khan and J. Hu, *Convergence analysis of the modified Chebyshev's method for finding multiple roots*. Vietnam J. Math., 2022, 50(1), 59–68.

[16] C. Ma, Y. Wu and Y. Xie, *The Newton-type splitting iterative method for a class of coupled Sylvester-like absolute value equation*, J. Appl. Anal. Comp., 2024, 14(6), 3306–3331.

[17] K. Madhu, D. K. R. Babajee and J. Jayaraman, *An improvement to double-step Newton method and its multi-step version for solving system of nonlinear equations and its applications*, Numer. Algorithms, 2017, 74(2), 593–607.

[18] J. J. Moré, B. S. Garbow and K. E. Hillstrom, *Testing unconstrained optimization software*, ACM Trans. Math. Software, 1981, 7(1), 17–41.

[19] J. M. Ortega and W. C. Rheinboldt, *Iterative Solution of Nonlinear Equations in Several Variables*, Academic Press, New York, 1970.

[20] V. Pan and J. Reif, *Efficient parallel solution of linear systems*, in: Proceedings of the 17th ACM Symposium on Theory of Computing, December 1985, 1985.

[21] V. Pan and R. Schreiber, *An improved Newton iteration for the generalized inverse of a matrix, with applications*, SIAM J. Sci. Statist. Comput., 1991, 12(5), 1109–1130.

[22] S. Pollock, *Fast convergence to higher-multiplicity zeros*, 2019, available as: arXiv:1911.10647.

[23] S. Pollock and H. Schwartz, *Benchmarking results for the Newton-Anderson method*, Results Appl. Math., 2020, 8, Paper No. 100095.

[24] A. Quarteroni, R. Sacco and F. Saleri, *Numerical Mathematics, Texts in Applied Mathematics*, Springer-Verlag, Berlin, Heidelberg, 2006.

[25] S. Qureshi, I. K. Argyros, H. Jafari, A. Soomro and K. Gdawiec, *A highly accurate family of stable and convergent numerical solvers based on Daftardar-Gejji and Jafari decomposition technique for systems of nonlinear equations*, MethodsX, 2024, 13, 102865.

[26] S. Qureshi, I. K. Argyros, A. Soomro, K. Gdawiec, A. A. Shaikh and E. Hincal, *A new optimal root-finding iterative algorithm: Local and semilocal analysis with polynomiography*, Numer. Algorithms, 2024, 95(4), 1715–1745.

[27] S. Qureshi, F. Chicharro, I. K. Argyros, A. Soomro, J. Alahmadi and E. Hincal, *A new optimal numerical root-solver for solving systems of nonlinear equations using local, semi-local, and stability analysis*, Axioms, 2024, 13, 341.

[28] G. W. Reddien, *On Newton's method for singular problems*, SIAM J. Numer. Anal., 1978, 15(5), 993–996.

[29] G. W. Reddien, *Newton's method and high order singularities*, Comput. Math. Appl., 1979, 5(2), 79–86.

[30] G. Schulz, *Iterative Berechnung der Reziproken matrix*, Z. Angew. Math. Mech., 1933, 13(1), 57–59.

[31] Y.-Q. Shen and T. J. Ypma, *Newton's method for singular nonlinear equations using approximate left and right nullspaces of the Jacobian*, Appl. Numer. Math., 2005, 54(2), 256–265.

[32] M. Singh, A. K. Verma and R. Agarwal, *On an iterative method for a class of 2 point & 3 point nonlinear SBVPs*, J. Appl. Anal. Comput., 2019, 9(4), 1242–1260.

[33] E. Suli and D. F. Mayers, *An Introduction to Numerical Analysis*, Cambridge University Press, New York, 2003.

[34] F. Toutounian and F. Soleymani, *An iterative method for computing the approximate inverse of a square matrix and the Moore-Penrose inverse of a non-square matrix*, Appl. Math. Comput., 2013, 224, 671–680.

[35] S. Weerakoon and T. G. I. Fernando, *A Variant of Newton's method with accelerated third-order convergence*, Appl. Math. Lett., 2000, 13, 87–93.