Newton's Method without Division

Jeffrey D. Blanchard and Marc Chamberland

Abstract. Newtons's Method for root-finding is modified to avoid the division step while maintaining quadratic convergence.

1. INTRODUCTION. Newton's Method is the most well-known superlinear method for root-finding. If x^* is a root of the smooth function $f : \mathbb{C} \to \mathbb{C}$, $f'(x^*) \neq 0$, and x_0 is sufficiently close to x^*, then the iterative scheme

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

converges quadratically to the root. This means that if $\epsilon_n = x_n - x^*$, then $\epsilon_n \to 0$ and there exists a constant C such that $|\epsilon_{n+1}| \leq C|\epsilon_n|^2$ for all n. In non-technical terms, this means that the number of digits of accuracy roughly doubles with each iteration. Newton's Method is not without its challenges: it can fail spectacularly if the initial guess is not sufficiently close to a root and each step requires a fresh evaluation of both $f(x)$ and $f'(x)$. In comparing various root-finding algorithms, factors to consider include rates of convergence, the number of function evaluations, the required smoothness of the function, the challenge of making initial guesses, the cost of arithmetic operations, the amenability of parallel processing, and the ease of implementation.

An important application of Newton's Method is to compute the reciprocal of a number. To calculate $x^* = 1/a$, note that x^* is a root of the function $f(x) = 1/x - a$. Applying (1) to this function leads to the iterative scheme[1]

$$x_{n+1} = x_n(2 - ax_n). \tag{2}$$

Since this division algorithm was created with Newton's Method, arbitrarily accurate division can be accomplished quadratically with only multiplication, addition, and subtraction. Indeed, using Newtons method to calculate multiplicative inverses forms the basis of many implementations of the division process in a vast number of computer processors; see [**14**].

Reflecting back on Newton's Method itself, equation (1) has a curious feature: each iteration of the Newton scheme employs a division step. Specifically, each iteration must calculate $1/f'(x)$, which itself requires several iterations. Since each iteration of Newtons Method only approximates a root of $f(x)$, a highly precise calculation of the reciprocal within each iteration may be wasteful. We can instead approximate the reciprocal with a single step of the division algorithm. Consider the following two-step Division-free Newton's Method:

$$y_{n+1} = y_n(2 - f'(x_n)y_n), \tag{3}$$

$$x_{n+1} = x_n - y_{n+1}f(x_n). \tag{4}$$

[1] To implement this iterative scheme on a computer, it is better to write it as

$$x_{n+1} = x_n + x_n(1 - ax_n)$$

to avoid cancellation errors.

Accepted, June 2022

Equation (3) approximates $1/f'(x_n)$ with one step of (2), which is subsequently used for the Newton step (4). If $f'(x^*) \neq 0$, define $y^* = 1/f'(x^*)$. The point (x^*, y^*) is a fixed point of the system (3)–(4). Choosing x_0 as an initial guess to x^* and $y_0 = 1/f'(x_0)$, we want to see how quickly this initial point approaches the fixed point.

This idea of approximating the division step has been used in one narrow but important application, namely, in calculating square roots. The so-called Babylonian Method for computing the square root of a positive, real number S starts with an initial guess $x_0 > 0$, then employs the iterative scheme

$$x_{n+1} = \frac{1}{2}\left(x_n + \frac{S}{x_n}\right) \tag{5}$$

successively to build better approximations. The intuition behind this formula is that if $x_n > \sqrt{S}$ (respectively "$<$"), then $S/x_n < \sqrt{S}$ (respectively "$>$") and the average of these two values will better approximate $\sqrt{S}$. The iteration (5) is alternatively seen as an instance of Newton's Method with the function $f(x) = x^2 - S$ and can be written as

$$x_{n+1} = x_n - \frac{x_n^2 - S}{2x_n}.$$

Decades ago [**4**, pp.226–227], [**11**, pp.92–94], it was realized that $1/\sqrt{S}$ could be calculated by applying Newton's Method to the function $f(x) = 1/x^2 - S$, resulting in the division-free iterative scheme

$$x_{n+1} = x_n + x_n\frac{1 - Sx_n^2}{2}.$$

This quadratically-converging iterative scheme generates an approximation to $1/\sqrt{S}$ which, when multiplied by S, produces $\sqrt{S}$. In contrast, (3)–(4) can be applied directly to $f(x) = x^2 - S$ to produce a division-free iterative scheme. The division operation in this procedure is replaced with only one step of the division algorithm (2). If x_0 is the initial guess to $\sqrt{S}$, define $y_0 = 1/(2x_0)$, the reciprocal of $f'(x_0)$. The two-step iterative scheme becomes

$$\begin{aligned} y_{n+1} &= y_n(2 - 2x_n y_n), \\ x_{n+1} &= x_n - (x_n^2 - S)y_{n+1}. \end{aligned}$$

As with Newton's Method, the terms x_n are intended to approach $\sqrt{S}$. The terms y_n approximate $1/(2x_n)$ and so are intended to approach $1/(2\sqrt{S})$. This modified approach for calculating square roots is credited to Arnold Schönhage and has been called *coupled Newton iteration* [**1**, pp.146–149]. Although this technique has been hailed as faster than other square root procedures, a mathematical analysis could not be found.

The goal of this paper is to explore the division-free adaptation (3)–(4) of Newton's Method. Most significantly, we show that the division-free scheme preserves quadratic convergence. Note that this technique applies to finding simple roots of any smooth function and requires no special insight into the function's structure.

2. EXPLORATIONS. To motivate the main theorem, we study the simple problem of finding the unique real root x^* of the function $f(x) = x^3 - x^2 - 1$, which is approximately 1.46557. We first explore the convergence with Newton's method, using the initial guess $x_0 = 1.4$. Table 1 shows approximations for the gap between x_n and x^* and the coefficient of $(x_n - x^*)/(x_{n-1} - x^*)^2$. As expected, the doubling powers of ten appearing in the second column manifest the quadratic convergence. This suspicion is further evidenced by the convergence of the values in the last column.

Table 1. Newton iterates with $f(x) = x^3 - x^2 - 1$ and initial guess 1.4.

n	$x_n - x^*$	$(x_n - x^*)/(x_{n-1} - x^*)^2$
1	4.558e-3	1.060
2	1.997e-5	0.961
3	3.857e-10	0.966
4	1.439e-19	0.967
5	2.002e-38	0.967
6	3.878e-76	0.967

In contrast, apply the division-free method (3)–(4) to the same function. Using $x_0 = 1.4$ and $y_0 = 1/f'(x_0)$, Table 2 mimics the earlier calculations. A cursory glance at the exponents in the second column again suggests quadratic convergence, although the doubling of digits of accuracy seems to lag. Also, the apparent lack of convergence of the coefficients in the third column suggests the convergence might not be quadratic. A closer look at the third column is even more suggestive: the *difference* between successive terms seems to approach a constant, roughly 2.9. This leads to a slightly weaker concept of convergence speed: letting $\epsilon_n = x_n - x^*$, we say the sequence $\{x_n\}$ converges *quasi-quadratically* to x^* if there exists a constant C such that $|\epsilon_{n+1}| \leq Cn|\epsilon_n|^2$. The fourth column, simply the third column divided by n, demonstrates the quasi-quadratic nature of the iterates. This example inspires the main result.

Table 2. Division-free Newton iterates with $f(x) = x^3 - x^2 - 1$ and initial guess 1.4.

n	$x_n - x^*$	$(x_n - x^*)/(x_{n-1} - x^*)^2$	$(x_n - x^*)/n(x_{n-1} - x^*)^2$
1	4.558e-3	1.060	1.060
2	1.227e-4	5.908	2.954
3	1.324e-7	8.783	2.927
4	2.067e-13	11.783	2.945
5	6.308e-25	14.764	2.952
6	7.055e-48	17.728	2.954

Main Theorem

Let $f : \mathbb{C} \to \mathbb{C}$ be a twice-continuously differentiable function. If $f(x^*) = 0$ and $f'(x^*) \neq 0$, then there is a neighborhood $U \subset \mathbb{C}^2$ of $(x^*, 1/f'(x^*))$ such that for almost all $(x_0, y_0) \in U$, the sequence $\{x_n\}$ produced by (3)–(4) converges quasi-quadratically to x^*.

The proof of the theorem is actually more explicit about the rate of convergence. The error decay can be measured asymptotically:

$$\epsilon_{n+1} \sim (3n) \frac{f''(x^*)}{2f'(x^*)} \epsilon_n^2. \tag{6}$$

The proportionality constant is virtually the same as in Newtons Method, the only difference being the factor $3n$. Going back to the example with $f(x) = x^3 - x^2 - 1$, the zero $x^* \approx 1.46557$ has corresponding values $c_1 = f'(x^*) \approx 3.512555$ and $c_2 = f''(x^*) \approx 6.793427$. The constant $3c_2/2c_1 \approx 2.90106$ matches the observed step size in successive quadratic coefficients one may calculate from the third column of Table 2.

3. PROOF OF THE MAIN THEOREM. The iteration (3)–(4) can be rewritten as

$$x_{n+1} = F(x_n, y_n), \tag{7}$$

$$y_{n+1} = G(x_n, y_n) \tag{8}$$

where

$$F(x, y) = x - y(2 - f'(x)y)f(x),$$
$$G(x, y) = y(2 - f'(x)y).$$

To determine the stability of the fixed point (x^*, y^*), the standard procedure is to compute the Jacobian of the transformation at this point. One finds that

$$J(x^*, y^*) = \begin{bmatrix} \frac{\partial F}{\partial x}(x^*, y^*) & \frac{\partial F}{\partial y}(x^*, y^*) \\ \frac{\partial G}{\partial x}(x^*, y^*) & \frac{\partial G}{\partial y}(x^*, y^*) \end{bmatrix} = \begin{bmatrix} 0 & 0 \\ -f''(x^*)/f'(x^*)^2 & 0 \end{bmatrix}. \tag{9}$$

The eigenvalues of this matrix are both zero, so (x^*, y^*) is an attractive fixed point [**8**, p.170]. This means that there exists a neighborhood $U \subset \mathbb{C}^2$ of (x^*, y^*) such that for all $(x_0, y_0) \in U$, the iteration (3)–(4) approaches (x^*, y^*). Additionally, the zero eigenvalues imply that the speed of convergence is superlinear, but further analysis is required to determine the precise rate.

Define the error terms

$$\epsilon_n = x_n - x^*,$$
$$\delta_n = y_n - y^*.$$

The aforementioned superlinear convergence ensures that ϵ_n and δ_n both approach zero if ϵ_0 and δ_0 are both sufficiently small. For brevity, let $c_1 = f'(x^*)$ and $c_2 = f''(x^*)$. This lets us write $y^* = 1/c_1$. The Taylor series expansions of $f(x)$ and $f'(x)$ near $x = x^*$ are

$$f(x_n) = c_1 \epsilon_n + \frac{c_2}{2} \epsilon_n^2 + O(\epsilon_n^3)$$

and

$$f'(x_n) = c_1 + c_2 \epsilon_n + O(\epsilon_n^2).$$

Accepted, June 2022

Using these expansions, equation (3) lets us express y_{n+1} as

$$\begin{aligned}\frac{1}{c_1}+\delta_{n+1} &= \left(\frac{1}{c_1}+\delta_n\right)\left[2-\left(c_1+c_2\epsilon_n+O(\epsilon_n^2)\right)\left(\frac{1}{c_1}+\delta_n\right)\right] \\ &= \left(\frac{1}{c_1}+\delta_n\right)\left[1-c_1\delta_n-\frac{c_2}{c_1}\epsilon_n+O(\epsilon_n^2)+O(\epsilon_n\delta_n)\right] \\ &= \frac{1}{c_1}-c_1\delta_n^2-\frac{c_2}{c_1^2}\epsilon_n+O(\epsilon_n^2)+O(\epsilon_n\delta_n),\end{aligned}$$

thus producing

$$\delta_{n+1}=-c_1\delta_n^2-\frac{c_2}{c_1^2}\epsilon_n+O(\epsilon_n^2)+O(\epsilon_n\delta_n), \tag{10}$$

Using this analysis with equation (4) generates

$$\begin{aligned}x^*+\epsilon_{n+1} &= x^*+\epsilon_n-y_{n+1}f(x_n) \\ &= x^*+\epsilon_n-\left(\frac{1}{c_1}-c_1\delta_n^2-\frac{c_2}{c_1^2}\epsilon_n+O(\epsilon_n^2)+O(\epsilon_n\delta_n)\right)\left(c_1\epsilon_n+\frac{c_2}{2}\epsilon_n^2+O(\epsilon_n^3)\right) \\ &= x^*+\frac{c_2}{2c_1}\epsilon_n^2+c_1^2\delta_n^2\epsilon_n+O(\epsilon_n^3)+O(\epsilon_n^2\delta_n),\end{aligned}$$

which in turn simplifies to

$$\epsilon_{n+1}=\frac{c_2}{2c_1}\epsilon_n^2+c_1^2\delta_n^2\epsilon_n+O(\epsilon_n^3)+O(\epsilon_n^2\delta_n). \tag{11}$$

The dominant terms in these expressions leave us with a nonlinear first-order system of difference equations. The analysis would be simplified if one could combine the variables to form a single first-order difference equation in one variable. To this end, defining $\gamma_n=c_1^3\delta_n^2/(c_2\epsilon_n)$ leads to

$$\begin{aligned}\gamma_{n+1} &= \frac{c_1^3}{c_2}\frac{\delta_{n+1}^2}{\epsilon_{n+1}} \\ &= \frac{c_1^3}{c_2}\frac{\left(-c_1\delta_n^2-\frac{c_2}{c_1^2}\epsilon_n+O(\epsilon_n^2)+O(\epsilon_n\delta_n)\right)^2}{\left(\frac{c_2}{2c_1}\epsilon_n^2+c_1^2\delta_n^2\epsilon_n+O(\epsilon_n^3)+O(\epsilon_n^2\delta_n)\right)} \\ &= \frac{(\gamma_n+1+O(\epsilon_n)+O(\delta_n))^2}{1/2+\gamma_n+O(\epsilon_n)+O(\delta_n)}.\end{aligned}$$

Any concerns about the denominator of γ_n equalling zero are easy to dispose of. If $c_2=0$, then the Jacobian matrix (9) is the zero matrix, hence the map (F,G) has quadratic convergence at the fixed point. If $\epsilon_n=0$ for some n, then the sequence $\{x_n\}$ has converged to x^* in a finite number of steps. These situations can therefore be ignored for the rest of the proof.

This rearrangement of γ_{n+1} as a function of γ_n compels us to consider the dynamics of the rational map

$$H(z)=\frac{(z+1)^2}{1/2+z}.$$

Rational maps on the extended complex plane (the Riemann sphere) have been studied for a century, especially over the last few decades. The ensuing analysis will make liberal use of this well-developed theory [**2**].

Since H maps the extended real axis to itself, it will be instructive to graph H on $\mathbb{R}$; see Figure 1. It is also helpful to rewrite H in the form

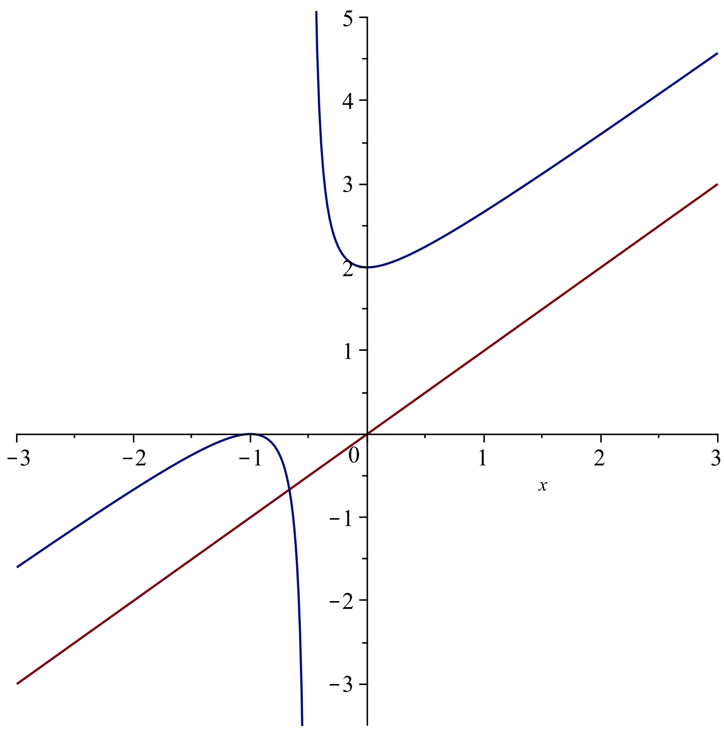

Figure 1. Plot of $y = H(x)$ restricted to the real axis and $y = x$.

$$H(z) = z + \frac{3}{2} + \frac{1}{4\left(z + \frac{1}{2}\right)}. \tag{12}$$

The map H has exactly two fixed points, easily found at $z = -2/3$ and $z = \infty$. Since $H'(-2/3) = -8$, the fixed point $z = -2/3$ is repelling. On the other hand, $H'(\infty) = 1$, so $z = \infty$ is a rationally indifferent fixed point. There are two critical points of H, located at $z = 0$ and $z = -1$. The dynamics of the critical points are telling for the dynamics in the whole plane. Note that $H(-1) = 0$ and, from either equation (12) or the graph, one sees that $z = 0$ iterates under H to the fixed point $z = \infty$. However, points close to $z = -\infty$ on the negative real axis are repelled away, moving to the right. Since H has degree two, each point in $\mathbb{C}$ has two pre-images. The graph then shows that the pre-image of the interval $[-\infty, 0]$ is itself, implying [**2**, p.71] that the Julia set J is contained in $[-\infty, 0]$. Collectively, this information implies that there is exactly one Fatou component (it is of parabolic type [**2**, p.160]). In summary, we have that almost all points in the complex plane iterate under H to infinity.

Since ϵ_n and δ_n tend to zero, we combine this with the dynamics of H to conclude that for almost all points in U, γ_n tends to infinity. To be more precise, note that if a

 Accepted, June 2022

and b are real, one finds that

$$H(a+ib) = a + \frac{3}{2} + \frac{a+\frac{1}{2}}{\left((a+\frac{1}{2})^2+b^2\right)} + ib\left(1 - \frac{1}{4\left((a+\frac{1}{2})^2+b^2\right)}\right).$$

This expression implies that if γ_n approaches infinity, its imaginary part approaches a constant, which in turn forces the real part to approach positive infinity. Using equation (12), a standard argument [**8**, p.220] shows that for almost all points in U, we have $\gamma_n = \frac{3n}{2} + o(n)$. Returning to the earlier variables, this yields the asymptotic approximation

$$\delta_n^2 \sim \frac{3n}{2}\frac{c_2}{c_1^3}\epsilon_n.$$

Equation (11) lets us conclude with

$$\epsilon_{n+1} \sim (3n)\frac{c_2}{2c_1}\epsilon_n^2.$$

This asymptotic behavior allows one to construct a constant C such that $|\epsilon_{n+1}| \leq Cn|\epsilon_n|^2$. We have therefore shown that for almost all starting points in a neighborhood of the fixed point (x^*, y^*), the division-free process is quasi-quadratic.

4. COMPARING THE CLASSICAL AND THE DIVISION-FREE NEWTON'S METHODS. With the Main Theorem proven, it is natural to wonder how the division-free method compares to Newton's Method. Some have argued [**6**] that the focus on the local convergence speed in root-finding algorithms is overemphasized because most of the iterations involve the initial point wandering aimlessly until it gets close to the root. However, these local results are valuable when many digits of a root are needed, as the following examples make evident. In massive computations of π to billions and trillions of digits — a record set in 2022 produced 100 trillion decimal digits — efficient implementations of basic operations such as square roots are critical [**1**]. Approximations of the Feigenbaum constant demand the accurate calculation of bifurcation points, a computationally heavy lift that can be refined by finding the roots of massive polynomials [**4**, pp.9–12]. In order to compute an approximation to the invariant measure for Hénon mappings, finding all the tightly packed roots of some high degree polynomials — the degree can range from a few hundred to a few thousand — requires punishing accuracy [**7**]. Lastly, a study employed 64,000-digit precision to recover the minimal polynomials of algebraic numbers connected to the Poisson potential function [**3**].

We revisit the example from Section 2 with an eye to high-precision calculations. Table 3 shows the number of steps needed for successive iterates to differ by a prescribed tolerance. This data shows that the division-free Newton's Method is competitive — if one simply counts the number of iterations — with its classical counter-part. Indeed, the division-free method seems to take at most one extra iteration. As we show, quasi-quadratic convergence is essentially the same as quadratic convergence.

The asymptotic formula (6) implies that there is a constant C such that $|\epsilon_{n+1}| \leq Cn|\epsilon_n|^2$. Scaling this inequality by C and letting $e_n = |C\epsilon_n|$ produces $e_n \leq ne_{n-1}^2$. Now build a sequence of inequalities that relate e_n to e_0:

$$e_1 \leq e_0^2,$$

Table 3. Number of steps needed to converge to the real root of $x^3 - x^2 - 1$ with initial guess 1.4.

Precision	Newton	Division-free
1e-10	5	5
1e-100	8	9
1e-1000	11	12
1e-10000	15	15
1e-100000	18	19
1e-1000000	21	22

$$
\begin{aligned}
e_2 &\le 2e_1^2 \le 2e_0^4 = \left(2^{1/2^2} 1^{1/2^1} e_0\right)^{2^2}, \\
e_3 &\le 3e_2^2 \le \left(3^{1/2^3} 2^{1/2^2} 1^{1/2^1} e_0\right)^{2^3}, \\
e_4 &\le 4e_2^2 \le \left(4^{1/2^4} 3^{1/2^3} 2^{1/2^2} 1^{1/2^1} e_0\right)^{2^4}, \\
&\vdots \\
e_n &\le ne_{n-1}^2 \le \left(n^{1/2^n} \cdots 2^{1/2^2} 1^{1/2^1} e_0\right)^{2^n}.
\end{aligned}
$$

This produces

$$e_n \le (Pe_0)^{2^n},$$

where $P = \prod_{n=1}^{\infty} n^{1/2^n}$ is a convergent product. Note that

$$\ln P = \sum_{n=1}^{\infty} \frac{\ln n}{2^n} = 0.507833922\ldots,$$

hence

$$P = 1.6616\ldots < 5/3.$$

Put together, this implies that if $e_0 < 3/5$, then e_n tends to zero and

$$e_n < (5e_0/3)^{2^n}.$$

This shows that quasi-quadratic convergence is essentially the same as quadratic convergence.

Another important difference between these iterative schemes regards choosing the initial guess. Even for Newton's Method applied to cubic polynomials, one is not guaranteed that almost every initial point $x_0 \in \mathbb{C}$ will converge to a root. For the function $f(x) = x^3 - 2x + 2$, there is an open set S in the complex plane where if $x_0 \in S$, then the Newton iterates do not approach a root of f, but rather a super-attracting 2-cycle [**9, 10**]. A technique to find initial points that will converge to the roots of polynomials with Newton's Method was developed in 2001 [**7**]. Notwithstanding the challenges with understanding the global dynamics of Newton's Method, these iterates seem more stable than those of its division-free counterpart. Revisiting the example from Section 2, namely $f(x) = x^3 - x^2 - 1$, the dynamics of the two methods differ markedly.

For the classical method, the iteration (1) becomes

$$x_{n+1} = x_n - \frac{x_n^3 - x_n^2 - 1}{3x_n^2 - 2x_n} \approx \frac{2}{3}x_n$$

when $|x_n|$ is large, thus the fixed point $x = \infty$ is repelling. On the other hand, the division-free Newton's Method (3)-(4) becomes

$$y_{n+1} = y_n(2 - (3x_n^2 - 2x_n)y_n),$$
$$x_{n+1} = x_n - (x_n^3 - x_n^2 - 1)y_{n+1}.$$

The global behavior of the classical and the division-free schemes for Newton's Method are illustrated in Figure 2. Each image depicts the eventual dynamics of a given (complex) initial value using the two different methods for the map $f(x) = x^3 - x^2 - 1$. The basin of attraction of each of the three roots is colored in shades of red, green, or blue[2]. Color saturation indicates the number of iterations required, where black points do not converge within 2000 iterations. The left image was created using the classical method, while the right used the division-free method, where $y_0 = 1/f'(x_0)$.

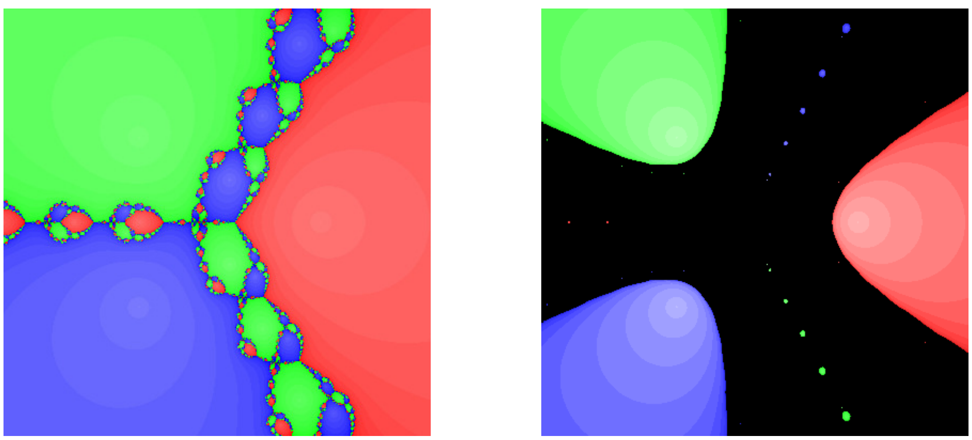

Figure 2. Basins of attraction using the classical method (left) and the division-free method (right).

Using the classical method, almost every starting point approaches one of the roots, although the boundary between the basins — the Julia set — is complicated. With the division-free method, one sees vast dark areas where the iterations failed to converge, highlighting the sensitivity of this new scheme.

5. HIGHER DIMENSIONS. In higher dimensions, Newton's Method applied to smooth functions $f : \mathbb{C}^m \to \mathbb{C}^m$ takes the form

$$x_{n+1} = x_n - (Df(x_n))^{-1} f(x_n), \tag{13}$$

[2]Color images are available in the open-access online version. In greyscale, blue is the darkest grey predominantly in the lower left, green the lightest grey in the upper left, and red the medium grey on the right-hand side of the images.

where Df is the $m \times m$ Jacobian matrix of f. To evaluate the last term, one usually does not compute the inverse but solves a linear system, still requiring extensive work, including many divisions. We note, however, that the inverse of a matrix A can be computed iteratively just like the reciprocal was solved by equation (2). The iteration takes the form

$$X_{n+1} = 2X_n - X_n A X_n. \tag{14}$$

This iterative scheme for matrix inversion was first proposed by Schulz[**13**] (also see [**12**]) and has been the object of extensive study, including for calculating the Moore-Penrose inverse of a matrix.

As in the scalar case considered earlier, the matrix inversion can be replaced with only one iteration of equation (14). The new process now involves the vectors x_n, which will approach the zero of the function, and matrices y_n that approximate the inverse of the Jacobian matrix at each step. Letting x_0 be the initial guess for x, $y_0 = (Df(x_0))^{-1}$, and I the $m \times m$ identity matrix, the iterative scheme becomes

$$y_{n+1} = y_n(2I - Df(x_n)y_n), \tag{15}$$

$$x_{n+1} = x_n - y_{n+1}x_n. \tag{16}$$

This process replaces matrix inversion with two matrix-matrix multiplications (or solving a linear system with two matrix-matrix multiplications and one matrix-vector multiplication).

A two-dimensional example involves the system

$$3\sin(2x_1 + x_2) - e^{x_1+x_2} = 0, \tag{17}$$

$$5\cos(x_1 + 2x_2) + \ln(3 + 7x_2) = 0. \tag{18}$$

An approximate solution is $x_1 = -7.1$ and $x_2 = 4.7$. Tables 4 and 5 depict the iterates using the Newton and inverse-free methods. Like the tables from Section 2, these new tables show the distinctions between the gaps $||x_n - x^*||$ and the quadratic coefficients $||x_n - x^*||/||x_{n-1} - x^*||^2$ in the two methods (here the standard Euclidean norm is used).

Table 4. Newton iterates for system (17)–(18) with initial guess $x_1 = -7.1$, $x_2 = 4.7$.

n	$\|\|x_n - x^*\|\|$	$\|\|x_n - x^*\|\|/\|\|x_{n-1} - x^*\|\|^2$
1	1.838e-2	1.675
2	2.266e-5	0.670
3	3.631e-11	0.706
4	9.319e-23	0.706
5	6.137e-46	0.706
6	2.661e-92	0.706
7	5.003e-185	0.706
8	1.768e-370	0.706

In a head-to-head comparison between the two methods, Table 6 shows the number of steps needed for successive iterates to differ by a prescribed tolerance. Although there is numerical evidence that the inverse-free method is quasi-quadratic, it is not obvious how to generalize the proof of the main theorem to the higher-dimensional setting.

Table 5. Inverse-free iterates for system (17)–(18) with initial guess $x_1 = -7.1$, $x_2 = 4.7$.

n	$\|\|x_n - x^*\|\|$	$\|\|x_n - x^*\|\|/\|\|x_{n-1} - x^*\|\|^2$	$\|\|x_n - x^*\|\|/n\|\|x_{n-1} - x^*\|\|^2$
1	1.838e-2	1.675	1.675
2	9.991e-5	2.956	1.478
3	5.329e-9	5.338	1.779
4	2.145e-17	7.552	1.888
5	4.481e-34	9.738	1.947
6	2.391e-67	11.909	1.984
7	8.048e-134	14.070	2.010
8	1.051e-266	16.225	2.028

Table 6. Steps needed to converge for system (17)–(18) with initial guess $x_1 = -7.1$, $x_2 = 4.7$.

Precision	Newton	Inverse-free
1e-10	4	5
1e-100	8	8
1e-1000	11	11
1e-10000	14	15
1e-100000	18	18

6. CONCLUSION. A modification of Newton's Method for root-finding has been developed that approximates the division step while maintaining quadratic convergence. Division-free Newtons's Method completely eliminates the need for any division when searching for roots of single-variable polynomials. Numerical evidence suggests that the technique also applies to multivariable functions. We suspect that this division-free approach can be applied to other iterative methods.

ACKNOWLEDGMENT. The authors heartily thank Richard Brent for the insight in Section 4 connecting quasi-quadratic convergence to essentially quadratic convergence. Sincere thanks also go the anonymous referees, particularly the reviewer who provided example code for generating the plots in Figure 2 using Dynamics Explorer [**5**].

REFERENCES

1. Arndt, J., Haenel, C. (2001). *Pi — unleashed.* New York: Springer.
2. Beardon, A. F. (1991). *Iteration of rational functions. Complex analytic dynamical systems.* New York: Springer.
3. Bailey, D.H., Borwein, J.M., Kimberley, J. and Ladd,W. (2017). Computer discovery and analysis of large Poisson polynomials. *Exp. Math.*. 26(3):349–363.
4. Borwein,J., Bailey,D. (2008). *Mathematics by experiment: Plausible reasoning in the 21st century*, second edition. Wellesley, MA: AK Peters.
5. Boyd, B., Boyd, S. (2016). Dynamics Explorer [Computer software], Retrieved from `https://sourceforge.net/projects/detool/`
6. Gonnet, G. (2001). A study of iteration formulas for root finding, where mathematics, computer algebra and software engineering meet. In: Casacuberta, C., Miró-Roig, R.M., Verdera, J., Xambó-Descamps, S. eds. *European Congress of Mathematics*, Vol. 202. Boston, MA: Birkhäuser, pp. 143–156.
7. Hubbard, J., Schleicher, D. and Sutherland, S. (2001). How to find all roots of complex polynomials by Newtons method. *Invent. Math.*. 146(1):1-33.
8. Kelley,W., Peterson,A. (2001). *Difference equations. An introduction with applications,* 2nd ed. San Diego: Harcourt/Academic Press.
9. McMullen, C. (1987). Families of rational maps and iterative root-finding algorithms. *Ann. of Math.*. 125(3):467–493.

10. McMullen, C. (1988). Braiding of the attractor and the failure of iterative algorithms. *Invent. Math.*. 91(2):259–272.
11. Muller, J.-M. (2006). *Elementary Functions: Algorithms and Implementation,* 2nd ed. Boston, MA: Birkhäuser.
12. Pan, V., Reif, J. (1985). Efficient parallel solution of linear systems. In: *Proceedings of the 17th ACM Symposium on Theory of Computing*, December 1985, available at: https://dl.acm.org/doi/10.1145/22145.22161
13. Schulz, G. (1933). Iterative Berechnung der Reziproken matrix. *Z. Angew. Math. Mech.*. 13(1): 57–59.
14. Soderquist, P., Leeser, M. (1997). Division and Square Root: Choosing the Right Implementation. *IEEE Micro* 17(4): 56–66. doi:10.1109/40.612224.

JEFF BLANCHARD (MR Author ID: 856216) is an associate professor of mathematics at Grinnell College. He has been an NSF International Research Fellow, a Project NExT Fellow, and a Harris Faculty Fellow. His primary research areas are compressed sensing, sparse approximation, and high performance computing with graphics processing units.

MARC CHAMBERLAND (MR Author ID: 335642) is the Myra Steele Professor of Mathematics at Grinnell College. He has published research in diverse areas such as differential equations and number theory, and his work is often colored by experimental mathematics. His attempts to popularize mathematics include the 2015 book *Single Digits* (Princeton) and the YouTube channel *Tipping Point Math*.

Grinnell College, Grinnell, IA 50112
blanchaj@grinnell.edu
chamberl@grinnell.edu