Received: 30 January 2023 | Revised: 10 May 2023 | Accepted: 13 June 2023

DOI: 10.1002/cpe.7853

RESEARCH ARTICLE

WILEY

Newton's method revisited: How accurate do we have to be?

Carl Christian Kjelgaard Mikkelsen[1] | **Lorién López-Villellas**[2] | **Pablo García-Risueño**[3]

[1]Department of Computing Science, Umeå University, Umeå, Sweden

[2]Barcelona Supercomputing Center, Barcelona, Spain

[3]Independent Scholar, Berlin, Germany

Correspondence
Carl Christian Kjelgaard Mikkelsen, Department of Computing Science, Umeå University, 90187 Umeå, Sweden.
Email: spock@cs.umu.se

Funding information
eSSENCE; Generalitat de Catalunya, Grant/Award Number: 2017-SGR-1328; Lenovo-BSC; Ministerio de Ciencia e Innovación, Grant/Award Number: PID2019-107255GB-C21/AEI/10.13039/501100011033

Summary

We analyze the convergence of quasi-Newton methods in exact and finite precision arithmetic using three different techniques. We derive an upper bound for the stagnation level and we show that any sufficiently exact quasi-Newton method will converge quadratically until stagnation. In the absence of sufficient accuracy, we are likely to retain rapid linear convergence. We confirm our analysis by computing square roots and solving bond constraint equations in the context of molecular dynamics. In particular, we apply both a symmetric variant and Forsgren's variant of the simplified Newton method. This work has implications for the implementation of quasi-Newton methods regardless of the scale of the calculation or the machine.

KEYWORDS

approximation error, convergence, quasi-Newton methods, rounding error, stagnation, systems of nonlinear equations

1 | INTRODUCTION

Newton's method is a well-known iterative method for solving systems of nonlinear equations.[1,2] Regardless, it remains an area of active research and a recent SIAM Review paper by Kelley[3] investigates the use of mixed precision arithmetic in the context of Newton's method. In this paper we consider the need for accuracy when executing general quasi-Newton methods. More formally, let $\Omega \subseteq \mathbb{R}^m$ be open, let $F : \Omega \to \mathbb{R}^m$ and consider the problem of solving

$$F(x) = 0,$$

with respect to $x \in \Omega$. If F is differentiable and if the Jacobian F' of F is nonsingular, then Newton's method is given by

$$x_{k+1} = x_k + s_k, \quad F(x_k) + F'(x_k)s_k = 0.$$

A quasi-Newton method is any iteration of the form

$$y_{k+1} = y_k + t_k, \quad F(y_k) + F'(y_k)t_k \approx 0.$$

In exact arithmetic, we expect local quadratic convergence from Newton's method.[4] Quasi-Newton methods normally converge locally and at least linearly and some methods, such as the secant method, have super-linear convergence.[1,2] In finite precision arithmetic, we cannot expect our iteration to converge and we must settle for stagnation near a zero. In this paper we analyze the convergence of quasi-Newton methods in exact and finite precision arithmetic using three different techniques. We extend work by Ypma,[5] Dennis and Walker[6] and Tisseur[7] and derive an upper bound for the relative error where a general quasi-Newton method will stagnate. We extend work by Forsgren[8] and show that any sufficiently exact

quasi-Newton method will in fact converge quadratically until stagnation. We confirm our analysis by computing square roots and solving bond constraint equations in the context of molecular dynamics. This is an extended version of a paper that was presented at PPAM-2022.[9] The new material includes a presentation of the techniques applied by previous authors in order to compare their approach to ours. Moreover, we include details and comments about our own work that were omitted from the conference paper due to page constraints. Finally, we include new experiments and observations related to Forsgren's variation[8] of the simplified Newton method.[1]

1.1 | Applications

In this section we explain why our topic is relevant for large-scale scientific computing. Consider the familiar problem of solving a nonsingular sparse linear system $Ax = b$ using a parallel computer. Let $\hat{x}$ denote the computed solution and let $\epsilon = \frac{\|x-\hat{x}\|_2}{\|x\|_2}$ denote the normwise relative error with respect to the 2-norm. Without additional information about the underlying application that produced the linear system our natural instinct is to reduce ϵ as much as feasible. How will we solve the linear system? We need to choose a method. It will either be an iterative method or a direct method. If we choose the GMRES algorithm, then the normwise residual $\|b - A\hat{x}\|_2$ will decay monotonically until stagnation. Therefore, we expect that ϵ will decrease with the number of iterations. If our accuracy goal is tiny, then a large number of iterations is a realistic expectation. Every iteration requires interprocessor communication which can be very expensive when the computer is large. In order to counter this issue, communication-avoiding Krylov subspace methods have been developed, see the PhD-theses by Hoemmen,[10] and Carson[11] and the many references therein. If we choose a direct method, say, a variant of Gaussian elimination, then we can reduce the backward error by applying pivoting during the LU-factorization phase. However, pivoting requires dynamic data structures and interprocessor communication which we are trying to avoid. An alternative to pivoting is static pivoting, diagonal boosting and iterative refinement as implemented in SuperLU$_{\text{dist}}$ by Sherry Li et al.[12] With this paper, we pause and question the need for accuracy in the first place. We explain that high accuracy is not strictly necessary when solving the linear systems needed for quasi-Newton methods for systems of nonlinear equations. We explain why the consequences of being inaccurate are rather mild. This information is relevant to anybody solving large-scale nonlinear equations on a parallel computer.

2 | BASIC ASSUMPTIONS AND NOTATION

We shall make the following assumptions. The set $\Omega \subseteq \mathbb{R}^m$ is open. The function $F : \Omega \to \mathbb{R}^m$ is of class C^1 and $F'(x)$ is nonsingular for all $x \in \Omega$. There exist positive real constants K and M such that for all $x \in \Omega$

$$\|F'(x)^{-1}\| \leq K, \quad \|F'(x)\| \leq M.$$

There exists a positive real constant L such that for all $x, y \in \Omega$

$$\|F'(x) - F'(y)\| \leq L\|x - y\|.$$

These assumptions are in effect throughout the paper. They are far more powerful than absolutely necessary, but they will allow us to complete the analysis rapidly and illustrate three different techniques without being distracted by technical details. If $z \in \mathbb{R}^m$ and $\delta > 0$ then $B(z, \delta) \subset \mathbb{R}^m$ is the open ball given by

$$B(z, \delta) = \{x \in \mathbb{R}^m : \|x - z\| < \delta\}.$$

If $\|\cdot\|$ is a norm on $\mathbb{R}^m$, then we induce a submultiplicative matrix norm on $\mathbb{R}^{m\times m}$ using the standard procedure, that is, if $A \in \mathbb{R}^{m\times m}$ then

$$\|A\| = \sup\{\|Ax\| : \|x\| \leq 1\}.$$

3 | AUXILIARY RESULTS

In this section we derive a few well-known results that are useful in the analysis of quasi-Newton methods. It is convenient to define Newton's method as the functional iteration

$$x_{k+1} = g(x_k), \quad g(x) = x - F'(x)^{-1}F(x),$$

and utilize this function when analyzing quasi-Newton methods. The following result shows that our assumptions ensure that F is Lipschitz with Lipschitz constant M. We shall use this result to bound the residual $F(x)$ near a zero z of F as $\|F(x)\| \leq M\|z - x\|$.

Lemma 1. *If $z \in \Omega$ and $B(z, \delta) \subseteq \Omega$, then for all $x, y \in B(z, \delta)$*

$$\|F(x) - F(y)\| \leq M\|x - y\|.$$

Proof. Let $x, y \in B(z, \delta)$ be given and consider the auxiliary function $\phi : [0, 1] \to \mathbb{R}^m$ given by

$$\phi(t) = F(tx + (1 - t)y).$$

The ϕ is well-defined because $B(z, \delta)$ is convex and the chain rule implies that $\phi \in C^1([0, 1], \mathbb{R}^m)$ with

$$\phi'(t) = F'(tx + (1 - t)y)(x - y).$$

By the fundamental theorem of calculus we have

$$F(x) - F(y) = \phi(1) - \phi(0) = \int_0^1 F'(tx + (1 - t)y)(x - y)dt = \left(\int_0^1 F'(tx + (1 - t)y)dt \right)(x - y).$$

By the triangle inequality we have

$$\|F(x) - F(y)\| \leq \int_0^1 \|F'(tx + (1 - t)y)\|dt\|(x - y)\|.$$

By applying our basic assumption that F' is bounded we can immediately conclude that

$$\|F(x) - F(y)\| \leq \int_0^1 Mdt\|x - y\| = M\|x - y\|.$$

This completes the proof. ∎

The following result allows us to measure the effect of a single iteration of Newton's method.

Lemma 2. *If z is a zero of F and $B(z, \delta) \subseteq \Omega$, then for all $x \in B(z, \delta)$*

$$z - g(x) = C(x)(z - x),$$

where

$$C(x) = F'(x)^{-1} \left(\int_0^1 \left[F'(x) - F'(tx + (1 - t)z) \right] dt \right).$$

Moreover, the new error $z - g(x)$ can be bounded in terms of the current error $z - x$, that is,

$$\|z - g(x)\| \leq \frac{1}{2} LK\|z - x\|^2.$$

Proof. Let $x \in B(z, \delta) \subseteq \Omega$. Then the function g is defined at x and

$$g(x) - z = (x - z) - F'(x)^{-1}(F(x) - F(z)) = F'(x)^{-1} \left[F'(x)(x - z) - (F(x) - F(z)) \right]$$
$$= F'(x)^{-1} \int_0^1 \left[F'(x) - F'(tx + (1 - t)z) \right] (x - z)dt = C(x)(x - z).$$

It follows that

$$\|z - g(x)\| \leq \|C(x)\|\|z - x\|.$$

By the triangle inequality we have

$$\|C(x)\| \leq \|F(x)^{-1}\| \int_0^1 \left\| F'(x) - F'(tx + (1 - t)z) \right\| dt\|(x - z)\|.$$

Since F' is Lipschitz continuous with Lipschitz constant L we have

$$\|C(x)\| \leq \|F'(x)^{-1}\| \int_0^1 L(1-t)\|x-z\|dt = \frac{1}{2}\|F'(x)^{-1}\|L\|x-z\|.$$

By applying our bound on the inverse of $F(x)$, we can now conclude that

$$\|z-g(x)\| \leq \|C(x)\|\|z-x\| \leq \frac{1}{2}\|F'(x)^{-1}\|L\|z-x\|^2 \leq \frac{1}{2}KL\|z-x\|^2.$$

This completes the proof. ∎

Lemma 2 implies that Newton's method will converge locally and that the order of convergence will be at least quadratic. The analysis of quasi-Newton methods in terms of the so-called relative residual will include an application of this lemma, see Section 4.1.

Lemma 3. *Let $z \in \Omega$ be a zero of F. Then there is a $\delta > 0$ such that if $x_0 \in B(z,\delta)$, then Newton's method is defined, $x_k \in B(z,\delta)$ and $\{x_k\}_{k=0}^{\infty}$ is convergent with limit z. Moreover, the order of convergence is at least quadratic.*

Proof. Since Ω is open there is a $\delta > 0$ such that $B(z,\delta) \subseteq \Omega$ and by Lemma 2 we have

$$\|z-g(x)\| \leq C\|z-x\|^2,$$

where $C = \frac{1}{2}LK$. Without loss of generality, we may assume that $\delta \leq C^{-1}$. It follows that

$$\|z-g(x)\| \leq (C\delta)\|z-x\| \leq \|z-x\| < \delta.$$

This shows that if $x_k \in B(z,\delta)$, then $x_{k+1} = g(x_k) \in B(z,\delta)$. Moreover, if $x_k \in B(z,\delta)$ then

$$C\|z-x_{k+1}\| \leq C^2\|z-x_k\|^2 = (C\|z-x_k|)^2.$$

By applying the principle of mathematical induction we conclude that

$$C\|z-x_k\| \leq (C\|z-x_0\|)^{2^k}.$$

Since $C\|z-x_0\| < C\delta < 1$ we see that the sequence $\{x_k\}_{k=0}^{\infty}$ is convergent with limit z and that the order of convergence is at least quadratic. This completes the proof. ∎

The following lemma allows us to write any approximation as a very simple function of the target vector. This lemma will be used to analyze quasi-Newton methods in terms of the distance to Newton's method, see Section 4.3.

Lemma 4. *Let $x \in \mathbb{R}^m$ be nonzero, let $y \in \mathbb{R}^m$ be an approximation of x and let $E \in \mathbb{R}^{n\times n}$ be given by*

$$E = \frac{1}{x^Tx}(y-x)x^T.$$

Then

$$y = (I+E)x, \quad \|E\| = O\left(\frac{\|x-y\|}{\|x\|}\right), \quad y \to x, \quad y \neq x.$$

In the special case of the 2-norm we have

$$\|E\|_2 = \frac{\|x-y\|_2}{\|x\|_2}.$$

Proof. It is straightforward to verify that

$$(I+E)x = x + \frac{1}{x^Tx}(y-x)x^Tx = x + (y-x) = y.$$

Moreover, if z is any vector, then

$$\|Ez\| \leq \frac{1}{\|x\|_2^2}\|y-x\|\|x^T\|\|z\| = \left(\frac{\|x^T\|\|x\|}{\|x\|_2^2}\right)\left(\frac{\|x-y\|}{\|x\|}\right)\|z\|.$$

In the case of the 2-norm, we have

$$\|Ez\|_2 \leq \frac{\|x-y\|_2}{\|x\|_2}\|z\|_2,$$

for all $z \neq 0$ and equality holds for $z = x$. This completes the proof. ∎

4 | ANALYSIS OF QUASI-NEWTON METHODS

In this section we shall illustrate three different techniques for analyzing the convergence of quasi-Newton methods.

4.1 | Using the relative residual

This is the approach introduced by Dembo, Eisenstat and Steinhaug in the seminal paper.[13] Let $x \in \Omega$ and let $t = t(x)$ denote the correction needed to compute the quasi-Newton approximation $x + t$, that is,

$$F(x) + F'(x)t \approx 0.$$

Then the relative residual $\eta = \eta(x, t)$ is given by

$$\eta = \frac{\|F(x) + F'(x)t\|}{\|F(x)\|}.$$

We shall now explain how quasi-Newton methods can be analyzed and controlled in terms of the relative residual. Let $z \in \Omega$ denote a zero of F and choose $\delta > 0$ such that $B(z, \delta) \subseteq \Omega$. Let $x \in B(z, \delta)$ and let s denote the correction needed to compute Newton's approximation $x + s$, that is,

$$F(x) + F'(x)s = 0.$$

Now consider the quasi-Newton approximation $x + t$. It is straightforward to use the definition of g to verify that

$$z - (x + t) = z - g(x) - F'(x)^{-1}(F(x) + F'(x)t).$$

By the triangle inequality and our bound for the inverse of $F'(x)$ we find

$$\|z - (x + t)\| \leq \|z - g(x)\| + K\|F(x) + F'(x)t\|.$$

At this point it becomes apparent how the relative residual enters into the analysis. We simply write $\|F(x) + F'(x)t\| = \eta\|F(x)\|$ so that

$$\|z - (x + t)\| \leq \|z - g(x)\| + \eta K\|F(x)\|.$$

We can now use Lemma 1 and Lemma 2 and estimate

$$\|z - (x + t)\| \leq \frac{1}{2}KL\|z - x\|^2 + \eta KM\|z - x\|.$$

We can now conclude that

$$\|z - (x + t)\| \leq \lambda(x, \eta)\|z - x\|,$$

where we have defined

$$\lambda(x, \eta) = \frac{1}{2} KL\|z - x\| + \eta KM.$$

Now given any fixed $\lambda \in (0, 1)$ we are free to choose $\eta > 0$ and $\delta > 0$ such that

$$\frac{1}{2} KL\delta + \eta KM \leq \lambda.$$

Now if $x_0 \in B(x, \delta)$ and if we ensure that the relative residuals all satisfy $\eta_k \leq \eta$, then

$$\|z - x_{k+1}\| \leq \lambda \|z - x_k\|.$$

We can now conclude that our quasi-Newton sequence $\{x_k\}_{k=0}^{\infty}$ is contained in $B(z, \delta)$ and convergent with limit z. Moreover, the order of convergence is at least linear. If we are prepared to gradually reduce the relative residual as we approach the zero, then higher orders of convergence can be achieved. In particular, if we ensure that

$$\eta \leq C' \|F(x)\|,$$

for some $C' > 0$ independent of x, then we can estimate

$$\|z - (x + t)\| \leq \|z - g(x)\| + C' K \|F(x)\|^2 \leq C\|z - x\|^2, \quad C = \frac{1}{2} KL + C' KM^2.$$

This is precisely the situation analyzed in Lemma 2. We need only choose $\delta > 0$ such that $B(z, \delta) \subseteq \Omega$ and $C\delta < 1$. If $x_0 \in B(z, \delta)$ and if the relative residuals satisfy

$$\eta_k \leq C' \|F(x_k)\|,$$

then our quasi-Newton sequence $\{x_k\}_{k=0}$ is contained in $B(z, \delta)$ and convergent with limit z, Moreover, the order of convergence is at least quadratic. The advantage of using the relative residual is particularly clear when we are using an iterative solver to solve the linear equation

$$F(x) + F'(x)s = 0,$$

with respect to s. Since the nonlinear residual is readily available, the iteration can be terminated when the linear residual satisfies

$$\|F(x) + F'(x)t\| \leq C' \|F(x)\|^2.$$

4.2 | Using forward and backward errors

This is the approach applied by Ypma,[5] Dennis and Walker,[6] and Tisseur.[7] While Tisseur's primary objective was to understand the impact of rounding errors on Newton's method, this approach applies to general quasi-Newton methods. We shall now illustrate the main idea.

Let $z \in \Omega$ be a zero of F and let $x \in \Omega$ denote our current approximation of z. Let $t = t(x)$ denote the correction needed to compute the quasi-Newton approximation $x + t$. Due to the limitations of floating point arithmetic we cannot hope to compute the exact value of t and we have to settle for an approximation which we denote $\hat{t}$. Let $\hat{F}(x)$ denote the computed value of $F(x)$. We assume that the function value $F(x)$ is computed with a forward error $E_1(x)$ that is small in the normwise relative sense, that is,

$$\hat{F}(x) = F(x) + E_1(x), \quad \|E_1(x)\| \leq \epsilon \|F(x)\|,$$

where the size of $\epsilon > 0$ reflects the difficulty of computing $F(x)$ and our skill as computer programmers. We shall now suppress most references to the point x and write, say, F instead of $F(x)$. Moreover, we shall write J (short for "Jacobian") instead of $F'(x)$ as this is shorter. We shall now treat $\hat{t}$ as an approximation of the solution of the linear system

$$Js + \hat{F} = 0. \tag{1}$$

Let $\eta = \eta(\hat{t})$ denote the normwise relative backward error. By theorem 7.1 of Higham's textbook[14] there exists a square matrix ΔJ and a vector $\Delta\hat{F}$ such that $\hat{t}$ is the exact solution of the perturbed equation

$$(J + \Delta J)\hat{t} + \hat{F} + \Delta\hat{F} = 0,$$

and

$$\|\Delta J\| = \eta\|J\|, \quad \|\Delta\hat{F}\| = \eta\|\hat{F}\|.$$

Let $\hat{y}$ denote the computed value of quasi-Newton approximation $y = x + t$. Our objective is to bound the error $z - \hat{y}$ in terms of the current error $z - x$. We shall now assume that

$$\eta\kappa(J) < 1,$$

where $\kappa(J)$ is the condition number of J, that is,

$$\kappa(J) = \|J^{-1}\|\|J\|.$$

This assumption ensures that

$$q = \|J^{-1}\Delta J\| < 1,$$

so that Banach's lemma implies that $J + \Delta J$ is nonsingular with

$$(J + \Delta J)^{-1} = (I + S)J^{-1}, \quad S = S(x) = \sum_{k=1}^{\infty}(-J^{-1}\Delta J)^k, \quad \|S\| \leq \frac{q}{1-q} \leq \frac{\eta\kappa(J)}{1 - \eta\kappa(J)}.$$

It follows that the computed value $\hat{y}$ of $y = x + t$ satisfies

$$\hat{y} = (I + D)(x - (I + S)J^{-1}(\hat{F} + \Delta\hat{F})),$$

where $D = D(x)$ is a diagonal matrix that represent the floating point error associated with the addition $x + \hat{t}$. We shall now isolate the expression for the function g that defines Newton's method, that is,

$$g(x) = x - F'(x)^{-1}F(x),$$

which we shall write as $g = x - J^{-1}F$. It follows that

$$\begin{aligned}\hat{y} &= (I + D)\left(x - (I + S)J^{-1}(F + E_1 + \Delta\hat{F})\right)\\ &= (I + D)\left(x - J^{-1}F - J^{-1}(E_1 + \Delta\hat{F}) - SJ^{-1}(F + E_1 + \Delta\hat{F})\right)\\ &= (I + D)(g - J^{-1}(E_1 + \Delta\hat{F}) - SJ^{-1}(\hat{F} + \Delta\hat{F}).\end{aligned}$$

We now define

$$h = J^{-1}(E_1 + \Delta\hat{F}) + SJ^{-1}(\hat{F} + \Delta\hat{F}),$$

so that we can write

$$\hat{y} = (I + D)(g - h).$$

Now let z denote a zero of F. We have

$$z - \hat{y} = z - (I + D)(g - h) = z - g + h - D(g - h).$$

It follows that

$$\|z - \hat{y}\| \leq \|z - g\| + \|h\| + \|D\|(\|g\| + \|h\|).$$

It is now convenient to write $g = z - (z - g)$ so that we may bound $\|g\|$ using

$$\|g\| \leq \|z\| + \|z - g\|,$$

and

$$\|z - \hat{y}\| \leq (1 + \|D\|)\|z - g\| + (1 + \|D\|)\|h\| + \|D\|\|z\|.$$

We shall now estimate h. We begin with the simple upper bound

$$\|h\| \leq \|J^{-1}\|(\|E_1\| + \|\Delta\hat{F}\|) + \|S\|\|J^{-1}\|(\|\hat{F}\| + \|\Delta\hat{F}\|).$$

We now consider the individual terms on the right-hand side. We have

$$\|J^{-1}\| \leq K, \quad \|E_1\| \leq \epsilon\|F\|, \quad \|\hat{F}\| \leq (1 + \epsilon)\|F\|, \quad \|\Delta\hat{F}\| = \eta\|\hat{F}\|, \quad \|S\| \leq \frac{\eta\kappa(J)}{1 - \eta\kappa(J)}, \quad \|F\| \leq M\|z - x\|.$$

It follows that

$$\|h\| \leq \left[\epsilon + \eta(1 + \epsilon) + \frac{\eta\kappa(J)}{1 - \eta\kappa(J)}(1 + \epsilon)(1 + \eta)\right] KM\|z - x\|.$$

We can now conclude that

$$\|z - \hat{y}\| \leq (1 + \|D\|)C(x, \epsilon, \eta)\|z - x\| + \|D\|\|z\|,$$

where we have defined

$$C(x, \epsilon, \eta) = \frac{1}{2}LK\|z - x\| + \left[\epsilon + \eta(1 + \epsilon) + \frac{\eta\kappa(J)}{1 - \eta\kappa(J)}(1 + \epsilon)(1 + \eta)\right] KM.$$

It follows that the normwise relative error satisfies

$$\frac{\|z - \hat{y}\|}{\|z\|} \leq (1 + \|D\|)C(x, \epsilon, \eta)\frac{\|z - x\|}{\|z\|} + \|D\|.$$

We observe that the function C satisfies

$$C(x, \epsilon, \eta) \to 0, \quad (x, \epsilon, \eta) \to (z, 0, 0).$$

We are certain that the relative error will be reduced provided that

$$(1 + \|D\|)C(x, \epsilon, \eta)\frac{\|z - x\|}{\|z\|} + \|D\| < \frac{\|z - x\|}{\|z\|},$$

or equivalently

$$\frac{\|D\|}{1 - (1 + \|D\|)C(x, \epsilon, \eta)} < \frac{\|z - x\|}{\|z\|}. \tag{2}$$

Conversely, if inequality (2) is not satisfied, then there is no guarantee that the relative error will be reduced. In this case we expect the iteration to stagnate. As observed by Ypma,[5] we note that this threshold, below which we cannot guarantee that the relative error will be reduced, is controlled by the size of ϵ and η. As observed by Tisseur,[7] we see that if the software used to evaluate the function F is sufficiently normwise forward stable (ϵ small) and if the software used to evaluate the Jacobian and solve the linear system $F(x) + F'(x)s = 0$ is sufficiently normwise backward stable (η

small) and if MK is not too large, then Newton's method will converge at least linearly until stagnation, provided that we start sufficiently close to our zero. This is very reassuring.

4.3 | Using the distance to Newton's correction

This is the approach that has recently been applied by the authors of the present paper, that is, Kjelgaard Mikkelsen, López-Villellas and García-Risueño.[9] We proceed as follows. Let $z \in \Omega$ be a zero of F and let $x \in B(z, \delta) \subseteq \Omega$. Let t denote the correction needed to compute the quasi-Newton approximation $y = x + t$ and let $\hat{t}$ denote the computed value of t. Let $\hat{y}$ denote the computed value of y. Our first objective is to bound the new error $z - \hat{y}$ in terms of the current error $z - x$. We shall write

$$\hat{y} = (I + D)(x + \hat{t}),$$

where D is a diagonal matrix that represents the floating point errors associated with the addition $x + \hat{t}$. Let s denote the correction needed to compute Newton's approximation $x + s$, that is, the solution of the linear system $F(x) + F'(x)s = 0$. We now choose to view $\hat{t}$ as an approximation of s. We now use Lemma 4 to write

$$\hat{t} = (I + E)s, \quad \|E\| = O\left(\frac{\|s - \hat{t}\|}{\|s\|}\right), \quad \|E\|_2 = \frac{\|s - \hat{t}\|_2}{\|s\|_2}.$$

We shall now relate $\hat{y}$ to $g = x + s$. We have

$$\hat{y} = (I + D)(x + (I + E)s) = x + (I + E)s + D(x + (I + E)s) = x + s + Es + D(x + s + Es) = g + Es + D(g + Es) = g + k,$$

where we have introduced

$$k = (I + D)Es + Dg.$$

It follows that

$$z - \hat{y} = (z - g) - k.$$

We shall use Lemma 2 to bound $z - g$ as

$$\|z - g\| \leq \frac{1}{2}LK\|z - x\|^2.$$

In order to bound k, it is convenient to write

$$g = z - (z - g),$$

and estimate

$$\|g\| \leq \|z - g\| + \|z\|.$$

It follows that

$$\|k\| \leq \|D\|\|z - g\| + (1 + \|D\|)\|E\|\|s\| + \|D\|\|z\|.$$

We can now conclude that

$$\|z - \hat{y}\| \leq (1 + \|D\|)\|z - g\| + (1 + \|D\|)\|E\|\|s\| + \|D\|\|z\|.$$

Since $s = F^{-1}(x)F(x)$ we can use Lemma 1 to obtain the bound

$$\|s\| \leq KM\|z - x\|.$$

It follows that

$$\|z - \hat{y}\| \leq \frac{1}{2}(1 + \|D\|)LK\|z - x\|^2 + (1 + \|D\|)\|E\|MK\|z - x\| + \|D\|MK\|z - x\| + \|D\|\|z\|.$$

In summary, we have established the following theorem (cf. theorem 1 of Reference 9).

Theorem 1. *Let z denote a zero of F and consider a general quasi-Newton method $x_{k+1} = x_k + t_k$. Then the computed values $\hat{x}_k$ of x_k satisfy a functional iteration of the form*

$$\hat{x}_{k+1} = (I + D_k)(\hat{x}_k + (I + E_k)s_k),$$

where D_k is a diagonal matrix that represents the rounding error associated with the addition $\hat{x}_t + \hat{t}_k$ and E_k represents the relative error between the computed value $\hat{t}_k$ of the correction t_k needed for the quasi-Newton approximation $\hat{x}_k + t_k$ and the correction s_k needed for the Newton approximation $\hat{x}_k + s_k$. Moreover, the error $z - \hat{x}_k$ satisfies the functional iteration

$$z - \hat{x}_{k+1} = z - g(\hat{x}_k) - E_k s_k - D_k(g(\hat{x}_k) + E_k s_k),$$

and the bound

$$\|z - \hat{x}_{k+1}\| \leq \frac{1}{2}(1 + \|D_k\|)LK\|z - \hat{x}_k\|^2 + (1 + \|D_k\|)\|E_k\|KM\|z - \hat{x}_k\| + \|D_k\|\|z\|. \tag{3}$$

We shall now explore the consequences of this result. It is convenient to focus on the case of $z \neq 0$ and restate inequality (3) as

$$r_{k+1} \leq \frac{1}{2}LK(1 + \|D_k\|)\|z\|r_k^2 + \|E_k\|KM(1 + \|D_k\|)r_k + \|D_k\|, \tag{4}$$

where r_k is the normwise relative forward error given by

$$r_k = \|z - x_k\|/\|z\|.$$

4.3.1 | Stagnation

There is every reason to believe that rounding errors will prevent our iteration from converging to z. It is clear that we cannot hope to do better than

$$r_{k+1} \approx \|D_k\|.$$

because inequality (4) reduces to $r_{k+1} \lesssim \|D_k\|$ when $r_k \approx 0$. In practice, we can only do finitely many iterations, so we choose $k_{\max} < \infty$, define

$$D = \max\{\|D_k\| : k \leq k_{\max}\}, \quad E = \max\{\|E_k\| : k \leq k_{\max}\}, \tag{5}$$

and impose the restriction

$$k \leq k_{\max},$$

for the remainder of this subsection. By inequality (4) we can now conclude that

$$r_{k+1} \leq \frac{1}{2}LK(1 + D)\|z\|r_k^2 + EMK(1 + D)r_k + D.$$

It is certain that the relative error will be reduced, that is, $r_{k+1} < r_k$ when

$$D < r_k - \left(\frac{1}{2}LK(1 + D)\|z\|r_k^2 + EMK(1 + D)r_k\right) = (1 - EMK(1 + D))r_k - \frac{1}{2}LK(1 + D)\|z\|r_k^2.$$

This condition is equivalent to the following inequality:

$$D - [1 - EMK(1 + D)]r_k + \frac{1}{2}LK(1 + D)\|z\|r_k^2 < 0.$$

The left-hand side is a polynomial in r_k of the second degree. We have equality when $r = r_\pm$ where $r_\pm$ is given by

$$r_\pm = \frac{[1 - EMK(1+D)] \pm \sqrt{[1 - EMK(1+D)]^2 - 2LK(1+D)D\|z\|}}{LK(1+D)\|z\|}.$$

If D and E are sufficiently small then the roots are positive real numbers and the error will certainly be reduced provided

$$r_- < r_k < r_+.$$

It follows that we cannot expect to do better than

$$r_k = \frac{\|z - x_k\|}{\|z\|} \approx r_-.$$

If D and E are sufficiently small, then a Taylor expansion ensures that

$$r_- \approx \frac{D}{(1 - EMK(1+D))^2},$$

is a good approximation. We cannot expect to do better than $r_{k+1} = r_-$, but this value is not particularly sensitive to the size of E.

4.3.2 | The decay of the error

As in Section 4.3.1 we accept that any practical application is necessarily limited to finitely many iterations. We therefore choose $k_{\max} < \infty$, define D and E using Equation (5) so that

$$\forall k \leq k_{\max} : (\|D_k\| \leq D \wedge \|E_k\| \leq E), \tag{6}$$

and impose the restriction $k \leq k_{\max}$ for the remainder of this subsection. Now suppose that

$$D \leq Cr_k. \tag{7}$$

for $C > 0$. In this case, inequality (4) implies

$$r_{k+1} \leq \rho_k r_k, \quad \rho_k = \frac{1}{2}LK(1+D)\|z\|r_k + EKM(1+D) + C. \tag{8}$$

If $C < 1$, then we may have $\rho_k < 1$, when r_k and E are sufficiently small. This explains when and why local linear decay is possible. We now strengthen our assumptions. Suppose that there is a $\lambda \in (0, 1]$ and $C_1 > 0$ such that

$$\|E_k\| \leq C_1 r_k^\lambda, \tag{9}$$

and that

$$D \leq C_2 r_k^{1+\lambda}, \tag{10}$$

for $C_2 > 0$. In this case, inequality (4) implies

$$r_{k+1} \leq \left[\frac{1}{2}LK(1+D)\|z\|r_k^{1-\lambda} + C_1KM(1+D) + C_2\right] r_k^{1+\lambda}. \tag{11}$$

This explains when and why local superlinear decay is possible.

4.3.3 | Convergence

We cannot expect a quasi-Newton method to converge unless the addition $x_{k+1} = x_k + t_k$ is exact. Then $D_k = 0$ and inequality (4) implies

$$r_{k+1} \leq \eta_k r_k, \quad \eta_k = \left(\frac{1}{2}LK\|z\|r_k + \|E_k\|KM\right).$$

We may have $\eta_k < 1$ for all k, provided $E = \sup \|E_k\|$ and r_0 are sufficiently small. This explains when and why local linear convergence is possible. We now strengthen our assumptions. Suppose that there is a $\lambda \in (0, 1]$ and a $C > 0$ such that

$$\forall k \in \mathbb{N} : \|E_k\| \leq C r_k^{\lambda}.$$

In this case, inequality (4) implies

$$r_{k+1} \leq \left(\frac{1}{2} LK\|z\| r_k^{1-\lambda} + CKM \right) r_k^{1+\lambda}.$$

This inequality allows us to establish local convergence of order at least $1 + \lambda$.

5 | HOW ACCURATE DO WE HAVE TO BE?

Our objective is to answer what is essentially the titular question of this paper, that is, what accuracy is required to achieve quadratic convergence when computing the corrections needed for Newton's method. We will assume the use of normal IEEE floating point numbers and we will apply the analysis given in Section 4.3.2. If we use the 1-norm, the 2-norm or the ∞-norm, then we may choose $D = u$, where u is the unit roundoff. Suppose that Equations (9) and (10) are satisfied with $\lambda = 1$. Then

$$\|E_k\| \leq C_1 r_k, \quad \|D\| \leq C_2 r_k,$$

and inequality (11) reduces to

$$r_{k+1} \leq \left[\frac{1}{2} LK(1+u)\|z\| + C_1 KM(1+u) + C_2 \right] r_k^2. \tag{12}$$

Due to the basic limitations of IEEE floating point arithmetic we cannot expect to do better than

$$r_{k+1} = O(u), \quad u \to 0, \quad u > 0. \tag{13}$$

From inequality (12) we see that Equation (13) is satisfied provided

$$r_k = O\left(\sqrt{u}\right).$$

Because we assumed that $\|E_k\| \leq C_1 r_k$ we can now conclude that we *never* need to do better than

$$\|E_k\| = O(\sqrt{u}), \quad u \to 0, \quad u > 0.$$

There is a simple argument that can be used to support this observation. Quadratic convergence is often *loosely* described by the fact that the number of correct significant figures doubles from one iteration to the next. This is not really true! What is true is that the number of *new* correct significant figures is doubling from one iteration to the next. In particular, a sequence such as $10^{-6}, 10^{-7}, 10^{-9}, 10^{-13}$ is displaying quadratic convergence toward zero. Now suppose that we are executing Newton's method using IEEE double precision floating point arithmetic and that we are experiencing quadratic convergence and that we have just gained four significant figures compared with the previous approximation. Rounding errors aside, we now expect to gain eight significant figures from computing $x_{k+1} = x_k + s_k$. To that end we do not need to know all 16 digits of s_k. It is sufficient to know the eight most significant digits of s_k.

6 | NUMERICAL EXPERIMENTS

In this section we consider three different numerical experiments that illustrate and support the main Theorem 1 and its consequences. In Section 6.1 we consider the problem of computing square roots of real numbers. This experiment is suitable for a lecture on elementary numerical analysis. In Section 6.2 we use quasi-Newton methods to solve nonlinear equations in the context of molecular dynamics with constraints using the GROMACS[15] package. Section 6.2.2 supports the main theorem through the use of a quasi-Newton method with a fixed symmetric approximation of the

Jacobian. This method exhibits linear convergence. Section 6.2.3 demonstrates the quadratic convergence of Forsgren's variation of the simplified Newton method.

6.1 | Computing square roots

Let $\alpha > 0$ and consider the problem of solving the nonlinear equation

$$f(x) = x^2 - \alpha = 0,$$

with respect to $x > 0$ using Newton's method. Let r_k denote the relative error after k Newton iterations. A simple calculation based on Lemma 2 yields

$$|r_{k+1}| \leq |r_k|^2/2, \quad |r_k| \leq 2(|r_0|/2)^{2^k}.$$

We see that convergence is certain when $|r_0| < 2$. The general case of $\alpha > 0$ can be reduced to the special case of $\alpha \in [1, 4)$ by accessing and manipulating the binary representation directly. Let $x_0 : [1, 4] \to \mathbb{R}$ denote the best uniform linear approximation of the square root function on the interval $[1, 4]$. Then

$$x_0(\alpha) = \alpha/3 + 17/24, \quad |r_0(\alpha)| \leq 1/24.$$

In order to illustrate Theorem 1 we execute the iteration

$$x_{k+1} = x_k - (1 + e_k)f(x_k)/f'(x_k)$$

where e_k is a randomly generated number. Specifically, given $\epsilon > 0$ we choose e_k such that $|e_k|$ is uniformly distributed in the interval $[\frac{1}{2}\epsilon, \epsilon]$ and the sign of e_k is positive or negative with equal probability. Three choices, namely $\epsilon = 10^{-2}$ (left), $\epsilon = 10^{-8}$ (center) and $\epsilon = 10^{-12}$ (right) are illustrated in Figure 1. We used IEEE double precision arithmetic to execute this experiment. In each case, eventually the perturbed iteration reproduces either the computer's internal representation of the square root or stagnates with a relative error that is essentially the double precision unit roundoff $u = 2^{-53} \approx 10^{-16}$. When $\epsilon = 10^{-2}$ the quadratic convergence is lost, but the relative error is decreased by a factor of approximately $\epsilon = 10^{-2}$ from one iteration to the next, that is, extremely rapid linear convergence. Quadratic convergence is restored when ϵ is reduced to $\epsilon = 10^{-8} \approx \sqrt{u}$. Further reductions of ϵ have no effect on the convergence as demonstrated by the case of $\epsilon = 10^{-12}$. We shall now explain exactly how far this experiment supports the theory that is presented in this paper.

6.1.1 | Stagnation

By Section 4.3.1 we expect that the level of stagnation is essentially independent of the size of E, the upper bound on the relative error between the computed step and the step needed for Newton's method. This is clearly confirmed by the experiment.

6.1.2 | Error decay

Since we are always very close to the positive zero of $f(x) = x^2 - \alpha$ we may choose

$$L \approx 2, \quad K|z| \approx 1/2, \quad MK \approx 1,$$

In the case of $\epsilon = 10^{-2}$, Figure 1 (left) shows that we satisfy inequality (7) with $D = u$ and $C = \epsilon < 1$, that is,

$$u \leq \epsilon r_k, \quad 0 \leq k < 5.$$

By Equation (8) we must have

$$r_{k+1} \leq \rho_k r_k, \quad \rho_k \approx 2\epsilon, \quad 0 < k < 5.$$

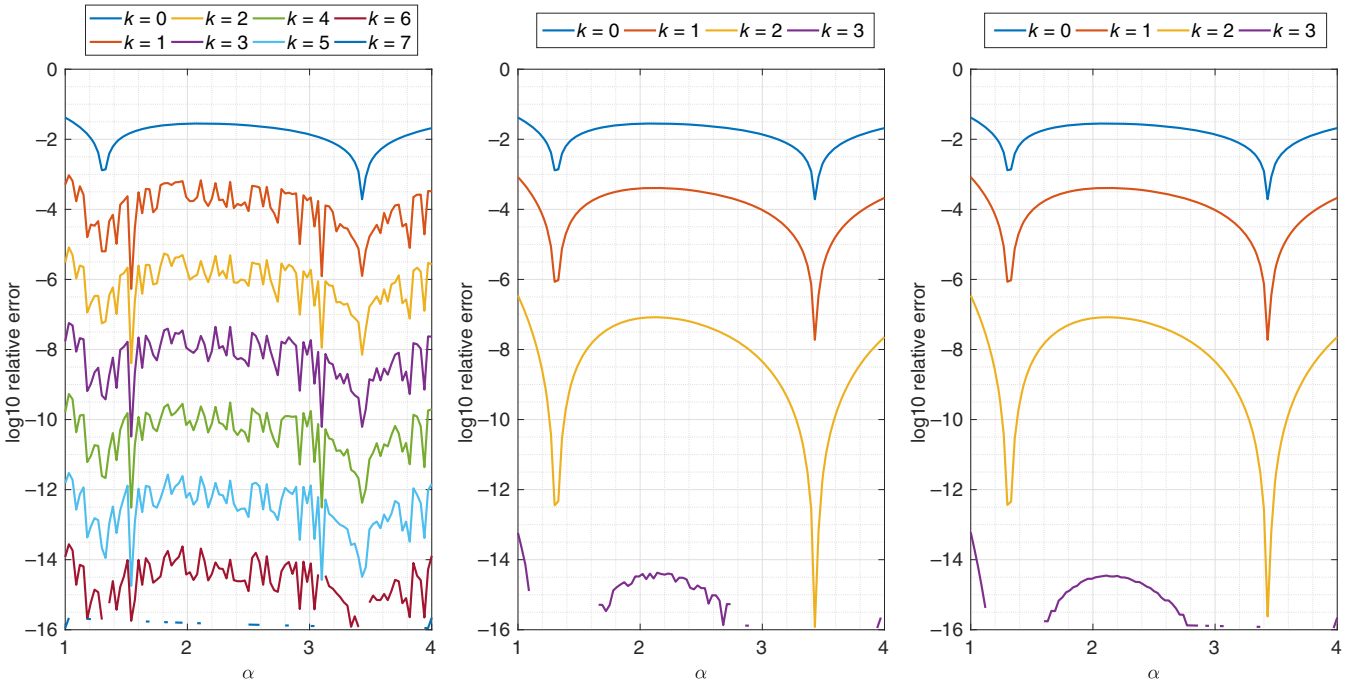

FIGURE 1 The impact of inaccuracies on the convergence of Newton's method for computing square roots. Newton's corrections have been perturbed with random relative errors of size $\epsilon \approx 10^{-2}$ (left), $\epsilon \approx 10^{-8}$ (center) and $\epsilon \approx 10^{-12}$ (right). In each case, the last iteration produces an approximation that matches the computer's value of the square root at many sample points. In such cases, the computed relative error is 0. Therefore, it is not possible to plot a data point and this is why the last curve of each plot is discontinuous.

This is exactly the linear convergence that we have observed. In the case of $\epsilon = 10^{-8}$, Figure 1 (center) shows that we satisfy inequality (10) with $C_2 = 1$ and $\lambda = 1$, that is,

$$u \leq r_k^2, \quad k = 0, 1.$$

By inequality (11) we must have quadratic decay in the sense that

$$r_{k+1} \leq C r_k^2, \quad C \approx \frac{3}{2}, \quad k = 0, 1.$$

Manual inspection of Figure 1 reveals that the actual constant is close to 1 and certainly smaller than $C \approx \frac{3}{2}$. By Section 5 we do not expect any benefits from using an ϵ that is substantially smaller than $\sqrt{u}$. This is also supported by the experiment.

6.2 | Constrained molecular dynamics

The present paper deals with a question that arose naturally during our ongoing investigation of constrained molecular dynamics.[9,16,17] We therefore choose to illustrate Theorem 1 in using constrained molecular dynamics. Molecular dynamics (MD) is a popular method for simulating time-evolution of atomic systems, MD simulations are frequently performed by imposing *constraints* on the internal degrees of freedom of the analyzed molecules. In this case, some degrees of freedom (usually bond lengths) are set to a constant value. This procedure permits the use of larger time steps, and hence the simulation of longer time intervals and the analysis of a wider range of physical phenomena. However, it is now necessary to compute the constraint forces that maintain the selected degrees of freedom at their chosen values. In constrained MD the objective is to solve a system of differential algebraic equations

$$\begin{aligned} \dot{q}(t) &= v(t), \\ M\dot{v}(t) &= f(q(t)) - g'(q(t))^T \lambda(t), \\ g(q(t)) &= 0. \end{aligned}$$

Here q and v are vectors that represent the position and velocity of all atoms, M is a nonsingular diagonal mass matrix and f represents the external forces acting on the atoms, $q \to g(q)$ is a vector function where each component represents a unique constraint, $g'(q)$ is the Jacobian of g at q and λ is the vector of Lagrange multipliers. The constraint forces are represented by the term $-g'(q)^T\lambda$. In the field of computational chemistry, the standard method for solving this particular system of differential algebraic equations is the SHAKE algorithm.[18] It uses a pair of staggered uniform grids and takes the form

$$\begin{aligned} v_{n+1/2} &= v_{n-1/2} + hM^{-1}\left(f(q_n) - g'(q_n)^T\lambda_n\right), \\ q_{n+1} &= q_n + hv_{n+1/2}, \\ g(q_{n+1}) &= 0, \end{aligned} \tag{14}$$

where $h > 0$ is the fixed time step and $q_n \approx q(t_n)$, $v_{n+\frac{1}{2}} \approx v(t_{n+1/2})$, where $t_n = nh$ and $t_{n+\frac{1}{2}} = (n+1/2)h$. Equation (14) is usually a nonlinear equation for the unknown Lagrange multiplier λ_n, specifically

$$g(\phi_n(\lambda)) = 0, \quad \phi_n(\lambda) = q_n + h\left(v_{n-\frac{1}{2}} + hM^{-1}\left(f(q_n) - g'(q_n)^T\lambda\right)\right).$$

In this context, Newton's method takes the form

$$\begin{aligned} g\left(\phi_n\left(\lambda_n^{(k)}\right)\right) + A_n(\lambda_n^{(k)})s_k &= 0 \\ \lambda_n^{(k+1)} &= \lambda_n^{(k)} + s_k, \end{aligned} \tag{15}$$

where n identifies the current time, k is the index used by Newton's method and the central matrix A_n is the Jacobian of the mapping $\lambda \to g(\phi_n(\lambda))$ with respect to λ. By the chain rule of differentiation we have

$$A_n(\lambda) = (g(\phi_n(\lambda)))' = g'(\phi_n(\lambda))\phi_n'(\lambda) = -h^2 g'(\phi_n(\lambda))M^{-1}g'(q_n)^T.$$

We observe that A_n is almost symmetric because

$$\phi_n(\lambda) = q_n + O(h), \quad h \to 0, \quad h > 0,$$

so that

$$A_n(\lambda) \approx S_n(\lambda),$$

where

$$S_n(\lambda) = -h^2 g'(q_n)M^{-1}g'(q_n)^T, \tag{16}$$

is symmetric negative semi-definite.

The most popular methods for constrained molecular dynamics are the SHAKE[18] and LINCS[19] algorithms. In the original SHAKE algorithm the constraint equations are solved using the nonlinear Gauss–Seidel method. Parallel versions of SHAKE exist that apply Newton's method and use the preconditioned conjugate gradient method to solve the necessary linear systems.[20] The LINCS[19] algorithm and its parallelization PLINCS[21] rely on a truncated Neumann series to approximate the solution of the relevant linear systems. As an alternative to SHAKE and (P)LINCS we are developing our own constraint algorithm based on direct solvers (ILVES[22]), which solves the same differential algebraic equations as SHAKE, yet using Newton's method together with a specialized parallel solver that exploits the topology of the molecule.[17]

6.2.1 | Description of our experiments

We shall now describe the physical details and the parameters that are shared between our experiments with constrained molecular dynamics. We simulated a single lysozyme molecule submerged in water inside a cubic box. Lysozyme[23] is a protein consisting of 129 amino acid residues and 1960 atoms with 1984 covalent bonds, four of them being disulfide bonds. Water molecules were simulated as rigid bodies. Our simulation included all the stages of a realistic MD calculation.[24] The first one was solvation, that is, including explicit water molecules which fill the simulated box.

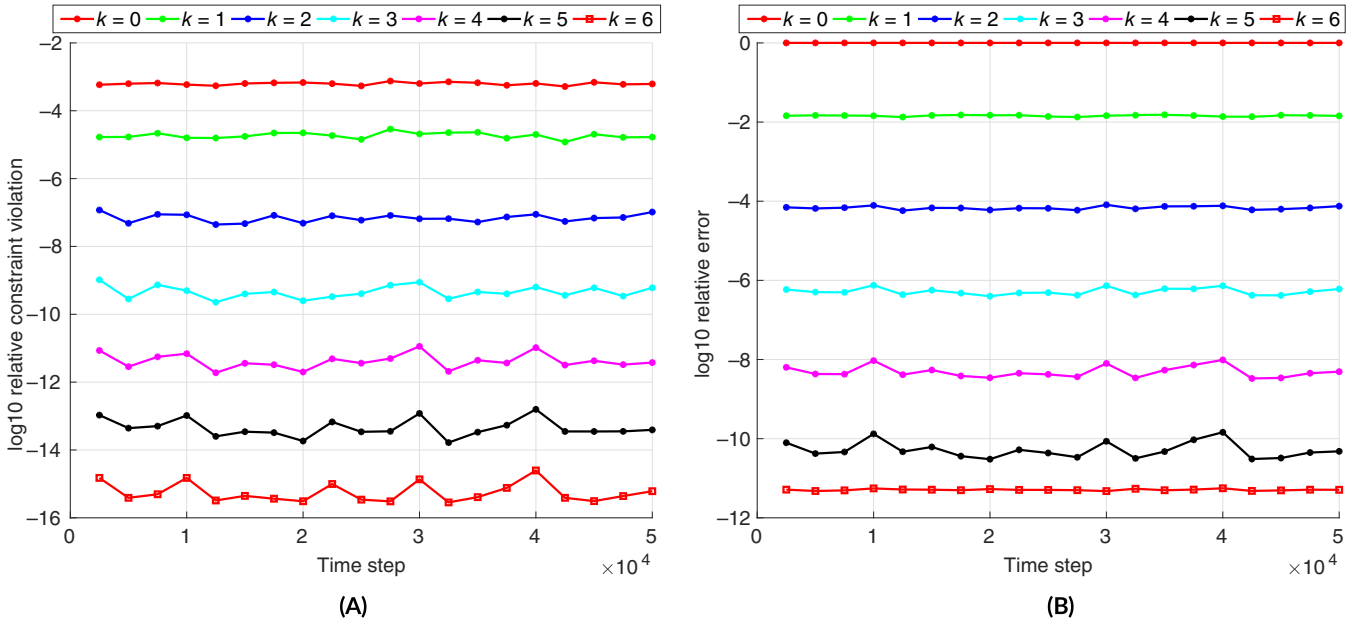

FIGURE 2 Data generated during a simulation of lysozyme in water using GROMACS. The GROMACS solver has been replaced with the quasi-Newton method given by Equation (17). (A) The maximum relative constraint violation always stagnates at a level that is essentially the IEEE double precision unit roundoff after 6 quasi-Newton iterations. The convergence is always linear and the rate of convergence is $\mu \approx 10^{-2}$. (B) The evolution of the relative error r_k between the relevant zero z, that is, the Lagrange multiplier λ_n for the current time step and the approximations generated by k iterations of the quasi-Newton method. The convergence is always linear and the rate of convergence is $\mu \approx 10^{-2}$. Further information about this experiment can be found in Figure 3. (A) Constraint violation; (B) relative error.

We then added ions to the system (protein and water) to make it electrically neutral. Afterwards we performed an energy minimization using the steepest descent algorithm until the module of the maximum force on every single atom was below 1000.0 kJ/(mol·nm). Then we executed 100 ps of a temperature equilibration step using a V-Rescale thermostat in the NVT ensemble (with τ_T set to 0.1 ps) to stabilize the temperature of the system at 310 K; in this stage the bond lengths of the protein which involve one hydrogen atom were constrained using LINCS with its default parameters in GROMACS. We then stabilized the pressure of the system at 1 bar for another 100 ps using a V-Rescale thermostat and a Parrinello-Rahman barostat (with τ_P set to 2.0 ps) in the NPT ensemble; the H-bonds were also constrained using LINCS. After these preparatory stages we executed the production run, from which we extracted the information summarized in Figures 2–6. The production run, in the NPT thermodynamic ensemble, simulated a total time of 100 ps, with a time step of 2 fs (i.e., 50k time steps). Again, the employed thermostat and barostat were V-Rescale and Parrinello-Rahman's, respectively (with τ_T= 0.1 ps, τ_P = 2.0 ps). In the production stage all bond lengths of the protein were constrained using our own implementation of Newton's method.

We shall now describe the details of the hardware and the software used to complete our simulations of molecular dynamics. The experiments were conducted using GROMACS 2021 compiled using GCC-10.1.0 in double precision mode (`-DGMX_DOUBLE=on`). Our experimental setup consisted of a system featuring one Intel Xeon Platinum 8160 processor and 48 GB of DDR4 main memory operating at 2667 MHz.

6.2.2 | A quasi-Newton method with a fixed symmetric approximation of the Jacobian

For this experiment we simulated lysozyme using the GROMACS package as explained in Section 6.2.1. We replaced GROMACS's constraint solver with our own implementation of Newton's method (15) as well as the quasi-Newton method given by

$$\begin{aligned} g\left(\phi_n\left(x_n^{(k)}\right)\right) + S_n t_k &= 0 \\ x_n^{(k+1)} &= x_n^{(k)} + t_k, \end{aligned} \tag{17}$$

where S_n is the symmetric approximation of the Jacobian given by Equation (16). For a subsequence of all time steps (every 5 ps, starting at $t = 5$ ps, for a total of 20 sample points) we first computed a good approximation of the Lagrange multiplier λ_n by running Newton's method until stagnation. We then recomputed the Lagrange multiplier using the quasi-Newton method (17). For each quasi-Newton iteration, we computed the following data:

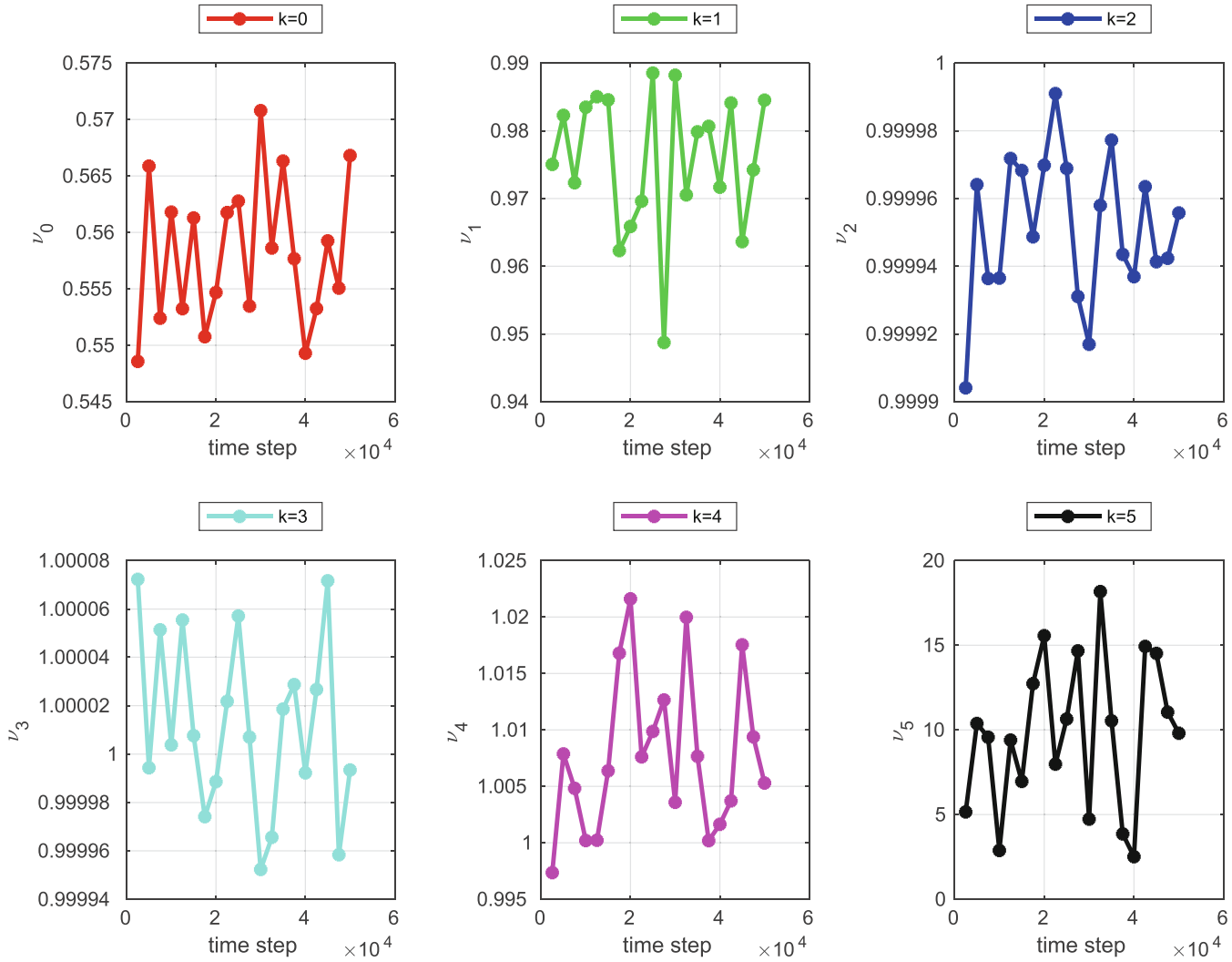

FIGURE 3 This figure and Figure 2 are based on the same experiment, that is, a simulation of lysozyme in water using GROMACS and the quasi-Newton method given by Equation (17). In this figure the fractions $\nu_k = r_{k+1}/(r_k\|E_k\|_2)$ are plotted for $k = 0, 1, 2, 3, 4, 5$. When ν_k is $O(1)$ we have experimental verification that the rate of convergence is essentially $\|E_k\|$. This is the case for $k = 0, 1, 2, 3, 4$, but not $k = 5$ because the iteration has stagnated when $k = 6$.

1. The maximum relative constraint violation, specifically the maximum relative error for any bond length, see Figure 2A.
2. The normwise relative error of $x_n^{(k)}$ against the value λ_n obtained by Newton's method, that is,

$$r_k = \frac{\|\lambda_n - x_n^{(k)}\|_2}{\|\lambda_n\|_2}.$$

3. The normwise relative error of the correction t_k against the correction s_k needed for Newton's method, that is,

$$\|E_k\|_2 = \frac{\|s_k - t_k\|_2}{\|s_k\|_2}.$$

Figure 2A shows that the quasi-Newton method solves the constraint equation and that the convergence is linear. Figure 2B shows that the quasi-Newton method finds the same solution as Newton's method. These two observations are mainly of interest to computational chemists. We now turn to the theory developed in this paper. By Equation (8) we have $r_{k+1} \leq \rho_k r_k$, but we cannot hope for more than $r_{k+1} \approx \rho_k r_k$ where $\rho_k = O(\|E_k\|_2)$ and this is indeed what we see in Figure 3 right until the point where we hit the level of stagnation and the impact of rounding errors is keenly felt. This shows that the rate of convergence is essentially $\|E_k\|_2$.

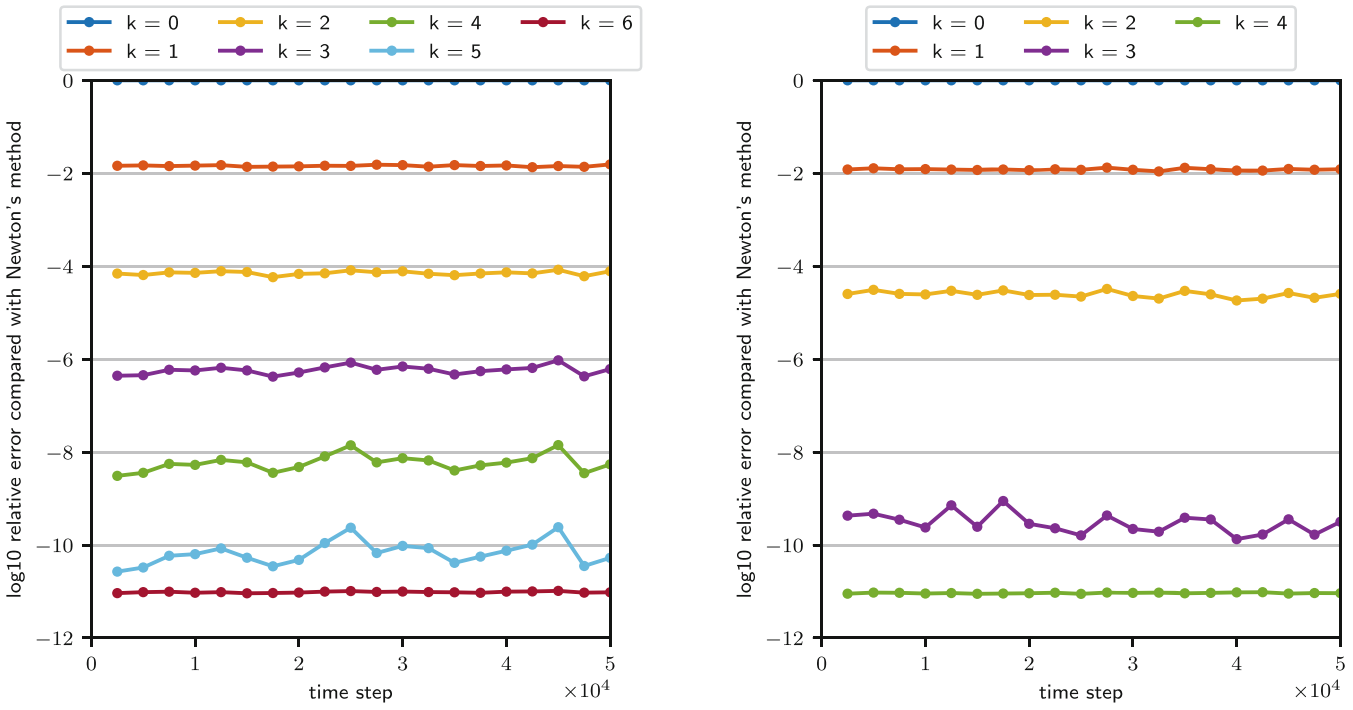

FIGURE 4 The convergence of the simplified Newton method and Forsgren's variant. The simplified Newton method shows rapid linear convergence toward the target, while Forsgren's variant shows quadratic convergence. Both methods stagnate at essentially the same level. (A) Simplified Newton method; (B) Forsgren's algorithm.

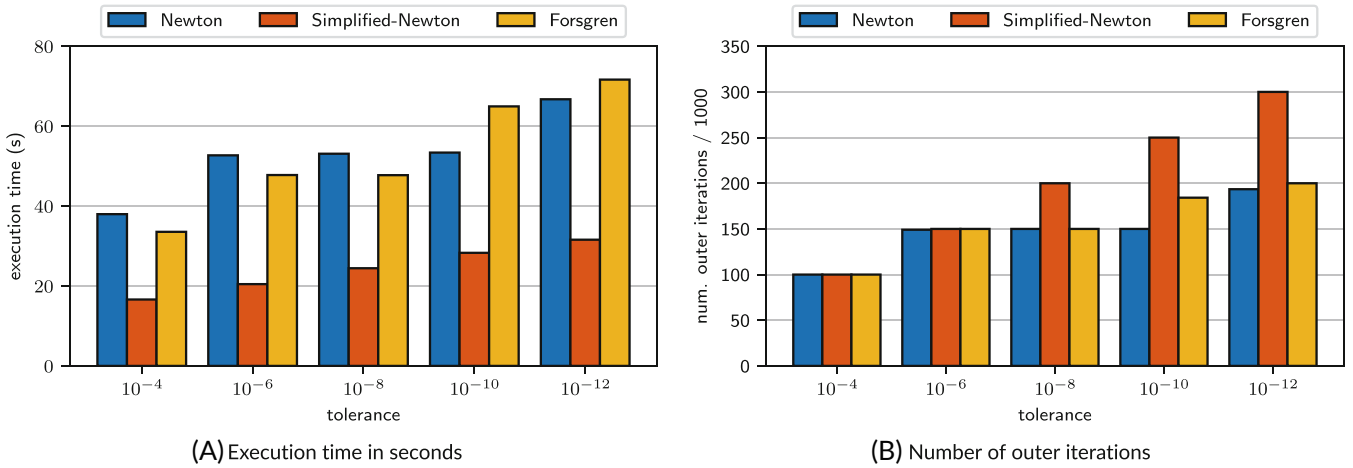

FIGURE 5 Total running time and total (outer) iteration count for the three analyzed algorithms for constrained molecular dynamics. (A) Execution time in seconds; (B) Number of outer iterations.

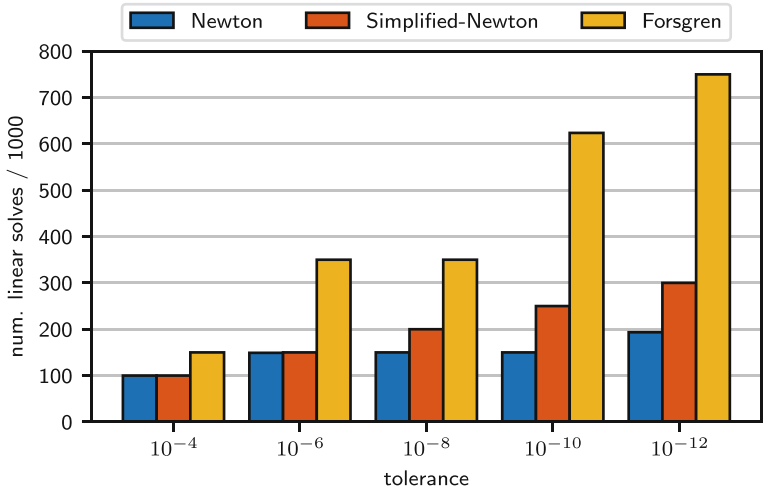

FIGURE 6 The total number of linear solves completed by three different algorithms.

6.2.3 | Forsgren's variant of the simplified Newton method

For this experiment we also simulated lysozyme using the GROMACS package, see Section 6.2.1. However, instead of GROMACS's constraint solvers, we used our own implementation of three different constraint solvers. Specifically, we used Newton's method (15), the simplified Newton method (Algorithm 1) and Forsgren's variant of the simplified Newton method (Algorithm 2).

The advantage of Algorithm 1 is that the Jacobian $F(x_0)$ is computed only once and if we can compute one of the standard factorizations (LU, QR, Cholesky) of $F(x_0)$, then this factorization can be recycled and used to compute the corrections t_k. The disadvantage is that the order of convergence is linear and not quadratic. Forsgren[8] has proposed an interesting variation of the simplified Newton method that restores the quadratic convergence. This is Algorithm 2. This variant uses $m_k = 2^k$ iterations of the simplified Newton method in order to approximate the solution of the *linear* equation

$$F(x_k) + F'(x_k)s_k = 0,$$

needed for Newton's method.

We see that $k_{\max}$ iterations of the outer for-loop of Algorithm 2 requires $\sum_{k=0}^{k_{\max}-1} 2^k = 2^{k_{\max}} - 1$ linear solves using the fixed (constant) matrix $F'(x_0)$. This is the same number of linear solves that is required to do $2^{k_{\max}} - 1$ iterations of the simplified Newton method. Therefore it may appear that $k_{\max}$ iterations of Algorithm 2 is essentially equivalent to $2^{k_{\max}} - 1$ iterations of the simplified Newton method. The key difference between the two algorithms is the number of function evaluations that are performed. In order to complete $k_{\max}$ outer iterations of Algorithm 2 we do k_{max} evaluations of the function F and the Jacobian F'. In order to complete $2^{k_{\max}} - 1$ iterations of the simplified Newton method we do $2^{k_{\max}} - 1$ evaluations of F and F'. The difference between k and $2^k - 1$ is significant even for modest values of k. Finally, by theorem 3.2 of Forsgren's paper[8] the convergence of Algorithm 2 is at least quadratic in the number of *outer* iterations.

We shall now describe the data that we collected. For a subsequence of time-steps we first computed a good approximation of the exact Lagrange multiplier by running Newton's method until stagnation. We then recomputed the Lagrange multiplier using the simplified Newton method as well as Forsgren's algorithm. We computed the normwise relative error of the Lagrange multipliers against the approximation produced by Newton's method. We did this for each iteration of the simplified Newton method and for each outer iteration of Forsgren's method. The results were

Algorithm 1. The simplified Newton method

1: **for** $k = 0, 1, 2, \ldots$ until convergence **do**
2: Solve $F(x_k) + F'(x_0)t_k = 0$ with respect to t_k
3: $x_{k+1} \leftarrow x_k + t_k$
4: **end for**

Algorithm 2. Forsgren's variation of the simplified Newton method

```
for k = 0, 1, 2, … until convergence do
    m_k ← 2^k
    d_{k,0} ← 0 ∈ ℝ^m
    for i = 0, 1, 2 … , m_k − 1 do
        Solve F′(x_0)p_{k,i} = −(F(x_k) + F′(x_k)d_{k,i}) with respect to p_{k,i}
        d_{k,i+1} ← d_{k,i} + p_{k,i}
    end for
    t_k ← d_{k,m_k}
    x_{k+1} ← x_k + t_k
end for
```

exactly as expected, see Figure 4. The quasi-Newton method converges linearly and Forsgren's variant converges quadratically with the number of outer iterations. We repeated the experiment and measured the total time needed to reduce the relative constraint violation below a variety of tolerances by applying Newton's method, the simplified Newton method and Forsgren's variant. The results are given in Figures 5 and 6. This experiment emphasizes a very practical limitation of Forsgren's variant. Specifically, the number of linear solves completed is limited to the set of numbers consisting of $\{1, 3, 7, 15, 31, \dots\}$, whereas the simplified Newton has the opportunity to terminate after the completion of each linear solve. As a specific example, consider the case of $\tau = 10^{-8}$. For each time step we need essentially four iterations (four linear solves) of the simplified Newton method while we need three outer iterations (seven linear solves) of Forsgren's variant. Since four is significantly smaller than seven, it is not surprising that the simplified Newton method is actually faster than Forsgren's variant in this particular case. This point is emphasized by Figure 6 which illustrates the total number of linear solves completed by each algorithm. Naturally, this does not imply that Forsgren's variant cannot be faster under the right circumstances. The issue here is the rapid convergence of the simplified Newton method. Had the simplified Newton method converged very slowly and had the function evaluations been very expensive, then the outcome of the experiment would have been quite different.

7 | CONCLUSION

The convergence of quasi-Newton methods can be analyzed and controlled in terms of the relative residual as done by Dembo, Eisenstat, and Steinhaug,[13] in terms of forward and backward errors as done by Ypma,[5] Dennis and Walker,[6] and Tisseur,[7] and in terms of the distance to the correction needed to execute a Newton iteration.[9] The first approach is very well suited to the situation where the correction needed for the next approximation is computed using an iterative solver. The second approach yields very mild conditions on the software that ensure that many quasi-Newton methods will converge at least linearly until stagnation. The third approach imposes even milder conditions on the implementation of general quasi-Newton methods $x_{k+1} = x_k + t_k$ and reveals that quadratic convergence is restored provided that we approximate the corrections $s_k = F'(x_k)^{-1}F(x_k)$ needed for Newton's method with an accuracy that is $O(\|s_k - t_k\|/\|s_k\|)$ and which need never exceed $O(\sqrt{u})$ where u is the unit roundoff. This fact represents an opportunity for improving the time-to-solution for certain nonlinear equations. When implementing a quasi-Newton method it is very natural and proper to rely on a general purpose library for solving the relevant linear systems. However, such libraries are normally written with the express goal of delivering solutions with maximum accuracy. In particular, a direct solver based on Gaussian elimination will usually pivot in order to reduce the backward error. However, parallel pivoting requires interprocessor communication and if the system is sufficiently sparse, then we may wish to consider whether pivoting is truly necessary. In the context of quasi-Newton methods, the consequences of an inaccurate solve are rather mild. Specifically, while we certainly expect to lose quadratic convergence, we have reason to expect rapid linear convergence. This may well be fast enough for the task at hand. Therefore, it may be worthwhile to use parallel sparse direct solvers that do not pivot for the sake of numerical stability.

ACKNOWLEDGMENTS

The anonymous reviewers did a splendid job which allowed the authors to improve the manuscript. The first author is supported by eSSENCE, a collaborative e-Science program funded by the Swedish Research Council within the framework of the strategic research areas designated by the Swedish Government. This work has been partially supported by the Spanish Ministry of Science and Innovation (contract PID2019-107255GB-C21/AEI/10.13039/501100011033), by the Generalitat de Catalunya (contract 2017-SGR-1328), and by Lenovo-BSC Contract-Framework Contract (2020).

CONFLICT OF INTEREST STATEMENT

The authors declare no potential conflict of interest.

DATA AVAILABILITY STATEMENT

The data that support the findings of this study are openly available in https://github.com/LorienLV/_PAPER_newtons-method-revisited-how-accurate-do-we-have-to-be.

ORCID

Carl Christian Kjelgaard Mikkelsen https://orcid.org/0000-0002-9158-1941

REFERENCES

1. Ortega JM, Rheinboldt WC. *Iterative Solution of Nonlinear Equations in Several Variables. Computer Science and Applied Mathematics*. Academic Press; 1970Reprinted 2000. doi:10.1137/1.9780898719468.
2. Kelley CT. *Iterative methods for linear and nonlinear equations*. Frontiers in Applied Mathematics. Vol 16. Society for Industrial and Applied Mathematics; 1995.
3. Kelley CT. Newton's method in mixed precision. *SIAM Rev*. 2022;64(1):191-211. doi:10.1137/20M1342902
4. Mysovskii IP. On the convergence of Newton's method. *Trudy Mat. Inst. Steklov*. 1949;28:145-147. (In Russian) https://www.mathnet.ru/php/archive.phtml?wshow=paper&jrnid=tm&paperid=1%060&option_lang=eng
5. Ypma TJ. The effect of rounding errors on Newton-like methods. *IMA J Num Anal*. 1983;3(1):109-118. doi:10.1093/imanum/3.1.109
6. Dennis JE, Walker HF. Inaccuracy in quasi-Newton methods: local improvement theorems. In: Korte B, Ritter K, eds. *Mathematical Programming at Oberwolfach II Mathematical Programming Studies*. Springer; 1984:70-85. doi:10.1007/BFb0121009
7. Tisseur F. Newton's method in floating point arithmetic and iterative refinement of generalized eigenvalue problems. *SIAM J Matrix Anal Appl*. 2001;22(4):1038-1057. doi:10.1137/S0895479899359837
8. Forsgren A. A sufficiently exact inexact Newton step based on reusing matrix information. TRITA-MAT OS7. Department of Mathematics, KTH. 2009. https://people.kth.se/%7Eandersf/doc/inexact.pdf
9. Mikkelsen CCK, López-Villellas L, García-Risueño P. How accurate does Newton have to be? In: Wyrzykowski R, Deelman E, eds. *Parallel Processing and Applied Mathematics*. LNCS. Vol 13826. Springer International Publishing; 2023:3-15. doi:10.1007/978-3-031-30442-2_1
10. Hoemmen M. *Communication-Avoiding Krylov Subspace Methods*. PhD Thesis. UC Berkeley, California; 2010. https://escholarship.org/uc/item/7757521k
11. Carson EC. *Communication-Avoiding Krylov Subspace Methods in Theory and Practice*. PhD Dhesis. UC Berkeley, California; 2015. https://escholarship.org/uc/item/6r91c407
12. Li XS. An overview of SuperLU: algorithms, implementation, and user interface. *ACM Trans Math Softw*. 2005;31(3):302-325. doi:10.1145/1089014.1089017
13. Dembo RS, Eisenstat SC, Steihaug T. Inexact Newton methods. *SIAM J Num Anal*. 1982;19(2):400-408. doi:10.1137/0719025
14. Higham NJ. *Accuracy and Stability of Numerical Algorithms*. 2nd ed. Society for Industrial and Applied Mathematics; 2002. doi:10.1137/1.9780898718027
15. Berendsen H, van der Spoel D, van Drunen R. GROMACS: a message-passing parallel molecular dynamics implementation. *Comput Phys Commun*. 1995;91(1):43-56. doi:10.1016/0010-4655(95)00042-E
16. Mikkelsen CCK, Alastruey-Benedé J, Ibáñez P, García-Risueño P. Accelerating sparse arithmetic in the context of Newton's method for small molecules with bond constraints. In: Wyrzykowski R, Deelman E, Dongarra J, Karczewski K, Kitowski J, Wiatr K, eds. *Parallel Processing and Applied Mathematics. LNCS*. Vol 9573. Springer International Publishing; 2016:160-171. doi:10.1007/978-3-319-32149-3_16
17. López-Villellas L, Kjelgaard Mikkelsen CC, Galano-Frutos JJ, et al. Accurate and efficient constrained molecular dynamics of polymers using Newton's method and special purpose code. *Comput Phys Commun*. 2023;288:108742. doi:10.1016/j.cpc.2023.108742
18. Ryckaert JP, Ciccotti G, Berendsen HJ. Numerical integration of the Cartesian equations of motion of a system with constraints: molecular dynamics of n-alkanes. *J Comput Phys*. 1977;23(3):327-341. doi:10.1016/0021-9991(77)90098-5
19. Hess B, Bekker H, Berendsen HJC, Fraaije JGEM. LINCS: a linear constraint solver for molecular simulations. *J Comput Chem*. 1997;18(12):1463-1472. doi:10.1002/(SICI)1096-987X(199709)18:12%3C1463::AID-JCC4%3E3.0.CO;2-H
20. Elber R, Ruymgaart AP, Hess B. SHAKE parallelization. *Eur Phys J Special Topics*. 2011;200(1):211-223. doi:10.1140/epjst/e2011-01525-9
21. Hess B. P-LINCS: a parallel linear constraint solver for molecular simulation. *J Chem Theory Comput*. 2008;4(1):116-122. doi:10.1021/ct700200b
22. García-Risueño P, Echenique P, Alonso JL. Exact and efficient calculation of Lagrange multipliers in biological polymers with constrained bond lengths and bond angles: proteins and nucleic acids as example cases. *J Comput Chem*. 2011;32(14):3039-3046. doi:10.1002/jcc.21885
23. RSCB. Protein data bank. Accessed June 26, 2023. https://www.rcsb.org/structure/1AKI
24. Lemkul JA. GROMACS tutorial: lysozyme in water. Accessed June 26, 2023. http://www.mdtutorials.com/gmx/lysozyme/index.html

How to cite this article: Kjelgaard Mikkelsen CC, López-Villellas L, García-Risueño P. Newton's method revisited: How accurate do we have to be?. *Concurrency Computat Pract Exper*. 2023;e7853. doi: 10.1002/cpe.7853