DEFLATION TECHNIQUES FOR FINDING MULTIPLE LOCAL MINIMA OF A NONLINEAR LEAST SQUARES PROBLEM

ALBAN BLOOR RILEY*†, MARCUS WEBB†, AND MICHAEL L. BAKER‡

Abstract. In this paper we generalize the technique of deflation to define two new methods to systematically find many local minima of a nonlinear least squares problem. The methods are based on the Gauss–Newton algorithm, and as such do not require the calculation of a Hessian matrix. They also require fewer deflations than for applying the deflated Newton method on the first order optimality conditions, as the latter finds all stationary points, not just local minima. One application of interest covered in this paper is the inverse eigenvalue problem (IEP) associated with the modelling of spectroscopic data of relevance to the physical and chemical sciences. Open source MATLAB code is provided at https://github.com/AlbanBloorRiley/DeflatedGaussNewton.

AMS subject classifications. 90C53, 65K05, 65K10, 90C26, 49M15

Key words. Gauss–Newton method, deflation, nonlinear least squares

1. Introduction. Nonlinear optimization problems often have many local minima. In many cases, finding an arbitrary one of them is not sufficient for the underlying application, and it is a global minimum that is sought, or several local minima that are all of interest. In this paper we generalize the technique of deflation to systematically find multiple minima of a nonlinear least squares problem.

Deflation techniques were first analysed by Wilkinson in 1963 to find multiple roots of polynomial equations by dividing the polynomial by the linear factor corresponding to the computed roots [27]. He demonstrated that the computed roots could become inaccurate if the roots are not found in ascending order of size. The method was later extended to systems of nonlinear algebraic equations in 1971 by Brown and Gearhart [6]. Unfortunately they also found that deflation was often unstable and could lead to divergence.

Decades later, in 2015 Farrell et al. discovered a simple improvement to the deflation process [8]: rather than multiplying the rootfinding problem by $|y-x|^{-1}$, for example, where y is a previously computed root, one can instead multiply by $1+|y-x|^{-1}$. This general idea of "shifting" the deflation operation means that away from the previously computed root, the deflated problem is well-approximated by the original problem, stabilizing the process of computing subsequent roots.

How to apply deflation techniques to optimization problems is an interesting question. The obvious approach is to apply deflation to the rootfinding problem associated with the first order optimality conditions [21, 9]. In this paper, however, we apply deflation techniques directly to the Gauss–Newton nonlinear least squares optimization algorithm. To do this, we define conditions on deflation operators specifically for optimization algorithms, leveraging a finer understanding of what deflation does to the behaviour of rootfinding algorithms. These new methods do not require the (possibly expensive) calculation of Hessian matrices nor do they converge to local maxima or saddle points.

In Section 2 we discuss the theory of deflation for rootfinding problems. In Section 3 we discuss the deflated Newton method, used in [8] and [21] to find multiple solutions to nonlinear PDEs, then introduce and prove properties of two new deflated methods, the "good" and "bad" deflated Gauss–Newton methods. Finally, in Section 4 we conduct numerical experiments to compare our new methods to the

*CORRESPONDING AUTHOR.

†Department of Mathematics, The University of Manchester, Manchester M13 9PL, United Kingdom, alban.bloorriley@manchester.ac.uk, marcus.webb@manchester.ac.uk

‡The University of Manchester, Department of Chemistry, Manchester M13 9PL, United Kingdom, The University of Manchester at Harwell, Diamond Light Source, Harwell Campus, OX11 0DE, UK, michael.baker@manchester.ac.uk

existing deflated Newton method and the MultiStart method in MATLAB [16]. Open source MATLAB code for all experiments is provided at https://github.com/AlbanBloorRiley/DeflatedGaussNewton.

2. Deflation techniques for rootfinding. Given a smooth function $r : \mathbb{R}^m \to \mathbb{R}^m$, and its Jacobian $J_r : \mathbb{R}^m \to \mathbb{R}^{m\times m}$ the system of equations $r(x) = 0$ can be solved using the Newton method given in Algorithm 2.1. If we wish to find multiple solutions of r we may use the deflated Newton method in Algorithm 2.2. A deflation operator should satisfy the following to be effective [8]:

DEFINITION 2.1. *([8]) Let r be as above and let $y_1, \dots, y_n \in \mathbb{R}^m$. We say that $\mu = \mu(\cdot; y_1, \dots, y_n) : \mathbb{R}^m \setminus \{y_1, \dots, y_n\} \to \mathbb{R}$ is a deflation operator for r if it satisfies the following:*

1. *The function $\nabla\mu(x; y_1, \dots, y_n)$ exists and is continuous for all $x \in \mathbb{R}^m \setminus \{y_1, \dots, y_n\}$.*
2. $\liminf_{x\to y_i} ||\mu(x; y_1, \dots, y_n)r(x)||_2 > 0$.
3. *If $r(x) = 0$ and $x \notin \{y_1, \dots, y_n\}$ then $\mu(x; y_1, \dots, y_n)r(x) = 0$.*
4. *If $r(x) \neq 0$ then $\mu(x; y_1, \dots, y_n)r(x) \neq 0$.*

Algorithm 2.1 The Newton method for nonlinear equations

$x^0 =$ initial guess
while $||r(x^k)||_2 <$ tol **do**
  Solve $J_r(x^k)p^k = -r(x^k)$
  $x^{k+1} = x^k + p^k$
end while

Algorithm 2.2 The deflated Newton method for nonlinear equations [8]

$x^0 =$ initial guess
while $||r(x^k)||_2 <$ tol **do**
  Solve $J_r(x^k)p^k = -r(x^k)$
  $\beta = 1 - \langle\mu(x^k)^{-1}\nabla\mu(x^k), p^k\rangle$
  $x^{k+1} = x^k + \beta^{-1}p^k$
end while

THEOREM 2.2 (Farrell et al. [8]). *A step of Algorithm 2.2 is equivalent to a step of Algorithm 2.1 applied to the system $\mu(x)r(x)$.*

Proof. A short proof can be found in Appendix A. ∎

Notice that in Algorithm 2.2 the value $\beta = 1 - \langle\mu^{-1}\nabla\mu, p^k\rangle$ is a scalar, and so the only effect that deflation has on the Newton method is that each step is just a scalar multiple of the undeflated Newton step p^k. This means that the effect of deflation can be fully described by the value of β, and in fact the behaviour can be split up into three distinct cases, which are displayed graphically in Figure 1 for Himmelblau's function,

$$r(x, y) = (x^2 + y - 11, x + y^2 - 7)^T. \tag{2.1}$$

1. When $\beta < 0$ (alternatively $\langle\mu^{-1}\nabla\mu, p^k\rangle > 1$) p^k is μ-ascending and the deflation update reverses the direction of the step.
2. When $0 < \beta < 1$ (or $0 < \langle\mu^{-1}\nabla\mu, p^k\rangle < 1$) p^k is moderately μ-ascending and deflation increases the step length. This can be interpreted as destabilising potential convergence to a deflated point.

3. Finally, when $\beta > 1$ ($\langle \mu^{-1}\nabla\mu, p^k\rangle < 0$)$p^k$ is μ-descending and deflation decreases the step length. If μ is relatively flat away from deflated points then this decrease is minimal there.

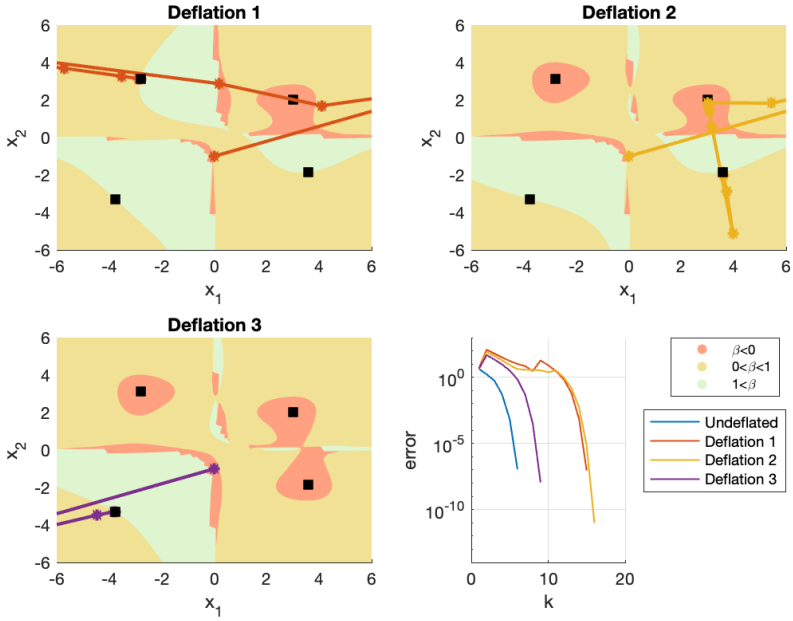

FIG. 1. *Multiple solutions to Himmelblau's function [14], defined above, computed using the deflated Newton method. Bottom Right: Shows the convergence rate of the method to all 4 roots. The other plots show the contours of β after 1, 2 and 3 deflations respectively. Notice that unfound solutions lie on the contour $\beta = 1$ and deflated solutions lie in a region where $\beta < 0$.*

2.1. Deflation operators. Early use of deflation was to find multiple roots of polynomial equations [27]. Given a polynomial, $p_n(x)$, and a root y that has been found by some iterative method, then the quotient polynomial

$$q_n(x) = \frac{p_n(x)}{y - x}, \tag{2.2}$$

is formed and a new root can be found by applying the same iterative rootfinding method to $q_n(x)$, a process which can be repeated. Wilkinson's careful analysis revealed numerical instabilities to this approach that depend on the order in which the roots are found, with ascending order in magnitude being the most stable order.

Brown and Gearhart [6] introduced the deflation operator

$$\mu(x; y) = \frac{1}{||y - x||} \tag{2.3}$$

for some vector norm, along with other matrix-based operators. In principle, these deflation operators can be applied to any rootfinding problem for $r : \mathbb{R}^m \to \mathbb{R}^m$. Unfortunately, they observed similar instabilities in this approach.

Farrell et al. modified the deflation operator by adding a shift, σ [8]. This results in the deflated system resembling the undeflated system when far enough away from any deflated points. The value of σ, is of course a parameter that needs to be defined. In this paper we will set $\sigma = 1$. However, we note that other choices are valid and can in theory lead to better results [8]. There is also the choice

of norm, for simplicity we will use the 2-norm, but it is possible to use different norms (see Subsection 4.2). It is also often advantageous to raise the norm to some power, θ, intended to overcome a higher multiplicity in the roots. In practice, $\theta = 2$ is effective. This all leads to the deflation operator that we will use in this paper:

$$\mu(x; y) = \frac{1}{||y - x||_2^\theta} + \sigma, \tag{2.4}$$

which satisfies both the root deflation Definition (2.1) and the optimization deflation Definition (3.2) defined in Section 3.

2.1.1. Multiple deflations. After a root, say y_1 has been found to a system of equations $r(x)$ we can deflate out this found root by defining the deflated system $\mu(x; y_1)r(x)$ and running the same rootfinding method on this new problem. This can be iterated to obtain $\mu(x; y_n) \cdots \mu(x; y_1)r(x)$ leading to the 'multi-shift' deflation operator:

$$\mu(x; y_1, \ldots, y_n) = \mu(x; y_1) \ldots \mu(x; y_n) = \left(\sigma + \frac{1}{||x - y_1||_2^\theta}\right) \ldots \left(\sigma + \frac{1}{||x - y_n||_2^\theta}\right) \tag{2.5}$$

An alternative to consider is a 'single shift' deflation operator:

$$\mu(x; y_1, \ldots, y_n) = \sigma + \frac{1}{||x - y_1||_2^\theta} \cdots \frac{1}{||x - y_n||_2^\theta}. \tag{2.6}$$

Note that sometimes, $y_i = y_j$ for distinct i and j. This can happen if the root of r at y_i is of multiplicity too great to be cancelled out by μ [6].

2.1.2. A novel deflation operator. The choice of θ was motivated by the expected multiplicity of the roots to be found. However, in many cases, this information will not be known, and a somewhat arbitrary choice needs to be made. We now provide an alternative operator that aims to incorporate all possible values of θ. Consider the expansion,

$$\exp\left(\frac{1}{||y - x||_2}\right) = 1 + \frac{1}{||y - x||_2} + \frac{1}{2}\frac{1}{||y - x||_2^2} + \cdots. \tag{2.7}$$

This expression incorporates all powers θ, so we define a new deflation operator $\mu(x; y) = \exp(\frac{1}{||y-x||_2})$. This deflation operator has not been investigated before.

This deflation operator is natural because deflation only involves $\mu^{-1}\nabla\mu$, which is relatively simple in this case:

$$\mu(x)^{-1}\nabla\mu(x) = \frac{y - x}{||y - x||_2^3}. \tag{2.8}$$

3. Deflation techniques for nonlinear least squares problems. A nonlinear least squares problem is that of minimising the function $f : \mathbb{R}^\ell \to \mathbb{R}$ given by

$$f(x) = \frac{1}{2}||r(x)||_2^2, \tag{3.1}$$

where $r : \mathbb{R}^\ell \to \mathbb{R}^m$. Such problems arise naturally when fitting m data-points r_i with ℓ parameters x_j in the presence of Gaussian noise [20].

In this section we discuss three different deflated methods for finding multiple minimizers of a given f. The first method we will discuss is the deflated Newton method given here as Algorithm 3.1.

Algorithm 3.1 The deflated Newton method for optimization

$x^0 =$ initial guess
while $||\nabla f(x^k)||_2 <$ tol **do**
  Solve $H_f(x^k)p^k = -\nabla f(x^k)$
  $\beta = 1 - \langle \mu(x^k)^{-1}\nabla\mu(x^k), p^k\rangle$
  $x^{k+1} = x^k + \beta^{-1}p^k$
end while

One of the main downsides to the Newton method for optimization, is that it requires calculating second derivatives of the objective function, which can be a relatively expensive operation, or indeed simply not available. Another disadvantage compared to Gauss–Newton methods is that the method converges to all stationary points of $f(x)$, not just to the local minima, so often computation is wasted on these unwanted stationary points. To remedy these two disadvantages, we introduce the "good" and the "bad" Deflated Gauss–Newton methods, presented here as Algorithms 3.2 and 3.3.

DEFINITION 3.1. *For a function $g : \mathbb{R}^d \to \mathbb{R} \cup \{+\infty\}$ and $c \in \mathbb{R}$, the c-level set of g is the set*

$$L_c(g) := \{x \,|\, g(x) > c\}\,.$$

In the rest of this section we will discuss the motivation and convergence analysis for the three methods. However, we first need a new definition of deflation operator for nonlinear least square problems. It is helpful to define the function $\eta(x) = \log(\mu(x))$.

DEFINITION 3.2. *For an iterative minimization method $x^{k+1} = x^k + p(x^k)$ and a set of deflated points $y_1, \ldots, y_n$, we say that $\mu(x; y_1, \ldots, y_n) : \mathbb{R}^\ell \setminus \{y_1, \ldots, y_n\} \to \mathbb{R}_{>0}$ is a deflation operator if it satisfies:*

1. *The function $\nabla\eta(x) = \mu(x)^{-1}\nabla\mu(x)$ exists and is continuous for all $x \in \mathbb{R}^\ell \setminus \{y_1, \ldots, y_n\}$.*
2. *If $p(y_i) = 0$ then $\liminf_{x\to y_i}\langle\nabla\eta(x), p(x)\rangle > 1$.*
3. *There exists a number $c^* > 0$ such that for any $c > c^*$ the level set $L_c(\mu)$ consists of n nonempty convex sets, each containing one of $y_1, \ldots, y_n$.*

Note that this definition differs from Definition 2.1 given earlier for systems of equations. In this definition neither f nor r is referred to directly, only the optimization step p.

This definition also does not explicitly exclude the possibility that deflation can create spurious solutions. We believe that his property depends on how the deflation operator is applied (we define two ways below). In practice we don't observe spurious solutions, but a theoretical understanding is lacking at this point in time.

3.1. Newton's method for optimization. Algorithm 3.1 is in fact still a rootfinding method and is the same as Algorithm 2.2 applied to $\nabla f(x)$ the gradient of $f(x)$, thus finding the stationary points of f. The algorithm requires the calculation of the gradient, $\nabla f \in \mathbb{R}^\ell$, and Hessian, $H_f \in \mathbb{R}^{\ell\times\ell}$, of f. The gradient is given by the formula

$$\nabla f(x) = J_r(x)^T r(x) \tag{3.2}$$

and the Hessian by

$$H_f = J_r(x)^T J_r(x) + \sum_{i=1}^{m} r_i(x) H_{r_i}(x) \tag{3.3}$$

where $r_i(x)$ represents the ith component of r and $H_{r_i}(x)$ represents the Hessian of r_i.

Algorithm 3.2 The "good" deflated Gauss–Newton method

$x^0 =$ initial guess
$\epsilon \in [0, 1]$ is as described in (3.13)
while $||p^k||_2 <$ tol **do**
  $p^k = \text{argmin}_p \|r(x^k) + J_r(x^k)\, p\|_2$
  if $\langle p^k, \nabla\eta(x^k)\rangle > \epsilon$ **then**
    $\beta = 1 - \langle \nabla\eta(x^k), p^k\rangle$
    $x^k = x^{k-1} + \beta^{-1} p^k$
  else
    $x^{k+1} = x^k + \alpha\, p^k$, where α is determined by line search on f.
  end if
end while

Algorithm 3.3 The "bad" deflated Gauss–Newton method

$x^0 =$ initial guess
$\epsilon \in [0, 1]$ is as described in (3.13)
while $||p^k||_2 <$ tol **do**
  $p^k = \text{argmin}_p \|r(x^k) + J_r(x^k)\, p\|_2$
  if $\langle p^k, \nabla\eta(x^k)\rangle > \epsilon$ **then**
    $\hat{p}^k = \beta_1\, p^k + \beta_2 \left(J_r(x^k)^T J_r(x^k)\right)^{-1} \nabla\eta(x^k)$ where β_1 and β_2 are defined in equation (3.20).
    $x^{k+1} = x^k + \hat{p}^k$
  else
    $x^{k+1} = x^k + \alpha\, p^k$, where α is determined by line search on f.
  end if
end while

Since the Newton algorithm for optimization can equivalently be thought of as the rootfinding Newton method (Algorithm 2.1) applied to ∇f, it will converge to stationary points of f given a sufficiently good initial guess.

The work of Papadopoulos et al. in [21] has shown that this deflated method can be effective at solving large-scale topology optimization problems.

3.1.1. Convergence. The Newton method for optimization is not be guaranteed to be a globally convergent method, this is due to the fact that H_f may not be positive definite far from a minimum. There are results, however, about the local convergence properties in the neighbourhood around a minimum.

THEOREM 3.3 ([20]). *Let $f(x)$ be twice differentiable, and $H_f(x)$ lipshitz continuous in a neighbourhood around a local minimum, say x^*. Then if the starting point, x^0, is sufficiently close to x^* the method will converge quadratically to x^*, and the sequence of norms of the gradients will also converge quadratically to* 0.

The Deflated Newton method will not converge to previously deflated points, given that large enough exponent, θ, is chosen, or after a sufficient number of deflations [6].

3.2. Gauss–Newton methods. First we will look at the undeflated method. This is an algorithm specifically designed for solving the nonlinear least squares problem, it is given here as Algorithm

3.4. Importantly it does not require the calculation of the Hessian of f, nor does it converge to all stationary points of f, just the minima (or maxima) that are of interest.

Algorithm 3.4 The Gauss–Newton method

$x^0 =$ initial guess
while $\|p^k\|_2 <$ tol **do**
  $p^k = \text{argmin}_p \|r(x^k) + J_r(x^k)\, p\|_2$
  $x^{k+1} = x^k + \alpha\, p^k$, where α is determined by line search on f.
end while

Interestingly in most textbooks the calculation of p^k in the algorithm is given by solving:

$$(J_r(x^k)^T J_r(x^k))p^k = -J_r(x^k)^T r(x^k). \tag{3.4}$$

This might due to the fact that the method is frequently described as a truncated Newton method; since $J_r^T J_r$ can be thought of as the approximation to H_f given by neglecting the second order terms, and $J_r r = \nabla f$. Note that this observation motivates one approach to deflating the method, covered in Section 3.4. A more formal motivation of the method however comes from looking at the first two terms of the Taylor series expansion of r at point x^k:

$$r(x) \approx r(x^k) + J_r(x^k)(x - x^k). \tag{3.5}$$

We then try to find the $-p = (x^k - x^{k+1})$ that minimizes this surrogate linear least squares problem:

$$\text{argmin}_p ||r(x^k) + J_r(x^k)p||_2^2. \tag{3.6}$$

Clearly in the square case the minimum occurs when $p = -J_r(x^k)^{-1} r(x^k)$. This result generalizes in the rectangular case, such that the minimum can be given by using the pseudoinverse: $p = -J_r(x^k)^+ r(x^k)$, although of course it is in general best to avoid explicitly calculating the pseudoinverse. Note that if J_r is left invertible — which is always the case when J_r has full rank — we can rewrite the pseudoinverse as $J_r^+ = (J_r^T J_r)^{-1} J_r^T$. Thus recovering the formula above.

Note that the use of the pseudoinverse shows that the method can be thought of as a generalization of the rootfinding Newton method applied to r. This provides the motivation for the other approach to applying deflation to the Gauss–Newton method, covered in Section 3.5.

3.3. Convergence of the undeflated Gauss–Newton method. Like the Newton method it is known that the Gauss–Newton method is locally convergent, under certain assumptions, but not necessarily globally convergent.

THEOREM 3.4 ([20]). *Assume that $r(x)$ is Lipschitz continuously differentiable and that $J_r(x)$ is uniformly full rank*[1] *for all x in a neighbourhood of $\{x|f(x) \leq f(x^0)\}$. If x^k are iterates generated by the Gauss–Newton method using a line search that satisfies the Wolfe conditions then*

$$\lim_{k\to\infty} J_r(x^k)^T r(x^k) = 0. \tag{3.7}$$

There are, however, no guarantees if J_r is not uniformly fully rank [20].

3.4. The "good" deflated Gauss–Newton method. Our first approach to deflating the Gauss–Newton method is motivated by the fact that it can be thought of as a truncated Newton method, and thus defined by the step:

$$x^{k+1} = x^k - (J_r^T J_r)^{-1} J_r^T r. \tag{3.8}$$

[1] i.e. there exists a constant $\gamma > 0$ such that $\|J_r(x)v\| \geq \gamma\|v\|$ for all $v \in \mathbb{R}^\ell$ and all $x \in \mathbb{R}^m$.

To derive a deflated method, we can apply similar logic that derives Gauss–Newton from optimization Newton by dropping Hessian terms. First we need to calculate the derivative of the deflated gradient:

$$\nabla(\mu J_r^T r) = J_r^T r \nabla\mu^T + \mu J_r^T J_r + \mu \sum_{i=1}^{m} r_i H_{r_i}. \tag{3.9}$$

Substituting this into the formula for the step of the optimization Newton algorithm

$$\left(\frac{1}{\mu} J_r^T r \nabla\mu^T + J_r^T J_r + \sum_{i=1}^{m} r_i H_{r_i}\right)^{-1} J_r^T r. \tag{3.10}$$

Since the Gauss–Newton method is given by neglecting the second order terms in Newton's method, we do the same here with the deflated components to get the deflated step:

$$\left(\frac{1}{\mu} J_r^T r \nabla\mu^T + J_r^T J_r\right)^{-1} J_r^T r. \tag{3.11}$$

The same logic as in the proof of Theorem 2.2 gives the same simplification:

$$x^{k+1} = x^k + \beta^{-1} p^k \tag{3.12}$$

where p^k is the Gauss–Newton step and $\beta = 1 - \langle \nabla\eta, p^k \rangle$.

3.4.1. Convergence and non-convergence. In order to talk about the convergence and indeed non-convergence of this method it is again useful to look at the three different cases for β, and what the effect of deflation is in each of these cases. In fact we propose adapting the method depending on which case the current iterate satisfies.

- When $1 < \beta$ (alternatively $\langle \nabla\eta, p \rangle < 0$), as stated before, the step p is already in a descent direction for μ, and the effect of deflation reduces the step size. In this case we suggest using an undeflated step with a line search on the undeflated objective function, thus keeping the convergence guarantees of the undeflated method covered in Section 3.3. This approach could be thought of as incorporating the method of 'purification' of the found minima; a technique introduced by Wilkinson in [27] to more accurately calculate all roots.
- When $\beta < 1$ ($0 > \langle \nabla\eta, p \rangle$), the step is in an ascent direction for μ. However, when $\beta < 0$ ($1 > \langle \nabla\eta, p \rangle$) the effect of deflation is to reverse the direction of the step and so is a descent direction for μ.

We know that all unfound solutions lie on the boundary $\beta = 1$ ($\langle \nabla\eta, p \rangle = 0$), since on this contour $p = 0$, and $\nabla\eta$ is finite by Definition 3.2 part (1). Ideally, a line search would be applied within a neighbourhood around each minimum to guarantee convergence. To achieve this we suggest relaxing the condition on β slightly so that the undeflated Gauss–Newton step with line search is used when

$$\beta > 1 - \epsilon, \quad \text{equivalently} \quad \langle \nabla\eta, p \rangle < \epsilon \tag{3.13}$$

for small $\epsilon \in [0, 1]$. The effect of different values of ϵ can be seen in Figure 2 where the green region, where undeflated line search Gauss–Newton is used, is significantly expanded for a positive value of ϵ.

Thus the "good" deflated Gauss–Newton method, Algorithm 3.2, uses the undeflated Gauss–Newton step with a line search on f when $\langle \nabla\eta, p \rangle \leq \epsilon$, and uses the deflated step, utilising the scalar update, with no line search applied when $\langle \nabla\eta, p \rangle > \epsilon$.

It is important to show that these deflated methods do not converge to any point that has been deflated.

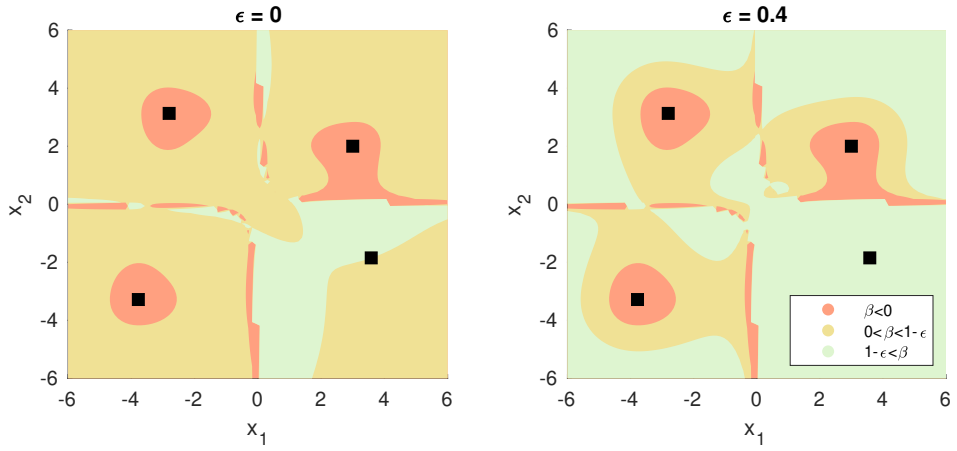

FIG. 2. *The effect of setting $\epsilon > 0$ on the pertinent contours of β, for the Himmelblau function (2.1) after three deflations. Notice that fourth solution lies on the contour $\beta = 1$.*

THEOREM 3.5. *Let $y_1, \ldots, y_n \in \mathbb{R}^\ell$ be a set of deflated points and $\mu(x; y_1, \ldots, y_n)$ be a deflation operator as in Definition 3.2. Let $\{x^k\}_{k=0}^\infty$ be the iterates of the "good" deflated Gauss–Newton method (Algorithm 3.2). Assume that r is Lipschitz continuously differentiable, J_r is uniformly full rank in a neighbourhood of $\{x^k\}_{k=0}^\infty$ and assume all steps are well-defined. Then x^k will not converge to any of the deflated points $y_1, \ldots, y_n$.*

Proof. Assume for contradiction that x^k converges to $y \in \{y_1, \ldots, y_n\}$.

Recall that the deflated step, which we will denote by $\hat{p}^k$, is

$$\hat{p}^k = \frac{p^k}{\beta(x^k)} = \frac{p^k}{1 - \langle \nabla\eta, p^k \rangle}. \tag{3.14}$$

When $\beta(x^k) < 1 - \epsilon$ then the deflated step is taken: $x^{k+1} = x^k + \hat{p}^k$, and when $\beta(x^k) \geq 1 - \epsilon$ then the undeflated step is taken with a line search on f: $x^{k+1} = x^k + \alpha\, p^k$. Thus we need to investigate the convergence behaviour of both of these steps. By the continuity of β (as a function of $x \in \mathbb{R}^d \setminus \{y_1, y_2, \ldots, y_n\}$), the tail of the sequence $\{x^k\}_{k=0}^\infty$ is formed by taking either all undeflated steps, $\alpha\, p^k$, or all deflated steps, $\hat{p}^k$.

First we assume that the tail of $\{x^k\}_{k=0}^\infty$ only takes undeflated steps. Then, as we have assumed the sequence (with a line search) converges, by Theorem 3.4, we have $J_r(x^k)^T r(x^k) \to 0$. Since J_r is uniformly full rank, this implies that $p^k \to 0$. By Definition 3.2, part (2),

$$\liminf_{k\to\infty} \langle \nabla\eta(x^k), p^k \rangle > 1. \tag{3.15}$$

By definition of the algorithm, however, this means the algorithm takes the the deflated step, $\hat{p}^k$, a contradiction.

The remaining case to consider is that the tail of $\{x^k\}_{k=0}^\infty$ takes only deflated steps. The convergence of the sequence implies $\lim_{k\to\infty} \hat{p}^k = 0$. Since $\hat{p}^k = \beta(x^k)^{-1} p^k$, this implies that $p^k \to 0$ or $\beta(x^k) \to -\infty$ (note that deflated steps are taken when $\beta(x^k) < 1 - \epsilon$, so $\beta(x^k) \to +\infty$ is impossible). First we assume that $p^k \to 0$. Then by part (2) of Definition 3.2 we again have the property in equation (3.15) above. If we assume instead that $\beta(x^k) \to -\infty$ then we also satisfy (3.15).

The property in equation (3.15) implies that the level set $L_1(\delta)$, where $\delta(x) = \langle \nabla\eta(x), p(x) \rangle$, contains the tail of $\{x^k\}_{k=0}^\infty$. Note that $L_1(\delta)$ is the red region in Figure 2. For all points x^k within the region $L_1(\delta)$ we know that $1 < \langle \nabla\eta(x^k), p^k \rangle$, and since $\beta(x^k) < 0$, we have $\langle \nabla\eta(x^k), \hat{p}^k \rangle < 0$, and hence $\langle \nabla\mu(x^k), \hat{p}^k \rangle < 0$.

Now, by part (3) of Definition 3.2 there exists a level set $L_c(\mu)$ containing y, and also the tail of $\{x^k\}_{k=0}^{\infty}$ (since the level set is an open set), such that the connected component containing y is convex and the same holds for all higher level sets. Let C be the closure of the particular convex subset of $L_{\mu(x^k)}(\mu)$ that contains y. Since x^k is on the boundary of the closed convex set C, and $\langle \mu(x^k), \hat{p}^k \rangle < 0$, we have that $x^{k+1} = x^k + \hat{p}^k$ lies outside of C (because the hyperplane associated to $\langle \nabla\mu(x^k), \hat{p}^k \rangle < 0$ separates C). This contradicts that the tail of $\{x^k\}_{k=0}^{\infty}$ lies within C. ∎

In this section we have shown that the "good" deflated Gauss–Newton method is locally convergent and does not converge to deflated points, as in Definition 3.2.

3.5. The "bad" deflated Gauss–Newton method. This method results from applying the Gauss–Newton method to μr. Why is it so "bad"? Local minima x^* of the deflated nonlinear least squares problem satisfy $J_{\mu r}^T \mu r = 0$, so are not local minima of the undeflated problem (unless $r(x^*) = 0$)! Thankfully, the use of the undeflated step when $\langle p^k, \nabla\eta(x^k) \rangle \leq \varepsilon$ resolves this issue, because the these steps converge to local minima of the undeflated problem.

The derivation taking this definition to Algorithm 3.3 requires that we understand the Moore-Penrose pseudoinverse of $J_{\mu r}$. The Sherman–Morrison formula does not apply to non-square matrices, but fortunately we can use the work of Meyer in [19] (later generalized in [13]) to adapt the update formula.

We assume that J_r is full rank, and thus $\nabla\eta \in R(J_r^T)$. Therefore we can utilize Theorem 5 from [19]. The possibility that J_r may be rank-deficient is beyond the scope of this paper.

THEOREM 3.6. *Let $A \in \mathbb{R}^{m\times l}$ be full rank, $u \in \mathbb{R}^m$ and $v \in \mathbb{R}^\ell$. Then*

$$(A + uv^T)^+ u = \frac{\beta}{\omega} A^+ u + \frac{\|Pu\|_2^2}{\omega} (A^T A)^{-1} v \tag{3.16}$$

where $\beta = 1 + \mu^{-1} \nabla\mu^T A^+ u, \quad \omega = \|Pu\|_2^2 \|A^{+T} v\|_2^2 + \beta^2, \quad$ and $P = I - AA^+$.

Proof. From [19, p.316] we use Theorem 5:

$$(A + uv^T)^+ = A^+ + \frac{1}{\beta} A^+ A^{+T} v u^T P - \frac{\beta}{\omega} p q^T, \tag{3.17}$$

where

$$p = \frac{\|Pu\|_2^2}{\beta} A^+ A^{+T} v + A^+ u, \qquad q^T = \frac{v^T A^+ A^{+T} v}{\beta} u^T P + v^T A^+.$$

Similarly to the original update formula, we do not need to store the pseudoinverse explicitly, since we only need the result of its multiplication with u. We thus calculate this action:

$$(A + uv^T)^+ u = A^+ u + \frac{\|Pu\|_2^2}{\beta} A^+ A^{+T} v - \frac{\beta}{\omega} p q^T u$$

Substituting in p:

$$= A^+ u + \frac{\|Pu\|_2^2}{\beta} A^+ A^{+T} v - \frac{\beta}{\omega} \left(\frac{\|Pu\|_2^2}{\beta} A^+ A^{+T} v + A^+ u \right) q^T u$$

Expanding out the brackets and rearranging the scalar:

$$= A^+ u + \frac{\|Pu\|_2^2}{\beta} A^+ A^{+T} v - \frac{\|Pu\|_2^2 q^T u}{\omega} A^+ A^{+T} v - \frac{\beta q^T u}{\omega} A^+ u$$

Collecting like terms:

$$= \left(1 - \frac{\beta q^T u}{\omega}\right)\left(A^+ u + \frac{\|Pu\|_2^2}{\beta} A^+ A^{+T} v\right)$$

Using the inverse since $A^T A$ is non singular:

$$= \left(1 - \frac{\beta q^T u}{\omega}\right)\left(A^+ u + \frac{\|Pu\|_2^2}{\beta} (A^T A)^{-1} v\right)$$

Expanding $\beta q^T u$

$$= \left(1 - \frac{v^T A^+ A^{+T} v \|Pu\|_2^2 + \beta v^T A^+ u}{\omega}\right)\left(A^+ u + \frac{\|Pu\|_2^2}{\beta} (A^T A)^{-1} v\right)$$

Substituting in ω

$$= \left(1 - \frac{\omega - \beta^2 + \beta v^T A^+ u}{\omega}\right)\left(A^+ u + \frac{\|Pu\|_2^2}{\beta} (A^T A)^{-1} v\right)$$

Finally, rearranging we get:

$$\begin{aligned} &= \left(1 - 1 - \frac{\beta v^T A^+ u - \beta^2}{\omega}\right)\left(A^+ u + \frac{\|Pu\|_2^2}{\beta} (A^T A)^{-1} v\right) \\ &= \frac{\beta}{\omega}\left(A^+ u + \frac{\|Pu\|_2^2}{\beta} (A^T A)^{-1} v\right) \\ &= \frac{\beta}{\omega} A^+ u + \frac{\|Pu\|_2^2}{\omega} (A^T A)^{-1} v \end{aligned} \tag{3.18}$$

∎

Setting $A = J_r$, $u = -r$ $v = -\nabla\eta$, Theorem 3.6 implies that the "bad" deflated Gauss–Newton step may be written as:

$$\hat{p}^k = \beta_1 \, p^k + \beta_2 \, \left(J_r^T J_r\right)^{-1} \nabla\eta \tag{3.19}$$

with

$$\beta_1 = \frac{\beta}{\omega} \quad \text{and} \quad \beta_2 = -\frac{\|Pr\|_2^2}{\omega}, \tag{3.20}$$

where

$$\beta = 1 - \langle p^k, \nabla\eta\rangle, \quad P = I - J_r(x^k) J_r(x^k)^+, \quad \omega = \|Pr\|_2^2 \|J_r^{+T} \nabla\eta\|_2^2 + \beta^2. \tag{3.21}$$

Interestingly, this deflated step contains a scalar multiple of the undeflated step, just like the deflated Newton and "good" Gauss–Newton methods, but also a term in the $-(J_r^T J_r)^{-1}\nabla\eta$ direction. This is a step in a direction that reduces η since $(J_r^T J_r)$ is positive definite.

Clearly (3.18) is a generalization of the original deflation update formula discussed in [8], as the square case can be easily recovered: if we assume that J_r is square, then $J_r^+ = J_r^{-1}$ so $J_r J_r^+ = J_r J_r^{-1} = I$ and $P = 0$, therefore $\omega = \beta^2$ and the inverse formula simplifies to the original Newton deflation update formula:

$$(J_r + r\nabla\eta^T)^{-1} r = \frac{\beta}{\omega} J_r^{-1} r = \frac{\beta}{\beta^2} J_r^{-1} r = \frac{p^k}{\beta} \tag{3.22}$$

It is important to note that the "bad" deflated Gauss–Newton method does not actually solve the minimization problem given in (3.1) but instead can be thought of as solving a nearby problem. This is because the "bad" deflated Gauss–Newton method minimizes the modified function $\mu^2 f$. We therefore suggest using the modification used in the "good" Gauss–Newton method, where the undeflated method is used when $\langle p, \nabla\eta\rangle < \epsilon$. This guarantees that the method will converge to local minima of f instead of those of $\mu^2 f$.

3.5.1. Convergence and non-convergence. Similar to the "good" deflated Gauss–Newton method discussed above we retain the local convergence properties of the Gauss–Newton method away from any deflated points.

Again it is important to show that the method will not converge to any deflated points.

THEOREM 3.7. *Let $y_1, \ldots, y_n \in \mathbb{R}^\ell$ be a set of deflated points and $\mu(x; y_1, \ldots, y_n)$ be a deflation operator as in Definition 3.2. Let $\{x^k\}_{k=0}^\infty$ be the iterates of the "bad" deflated Gauss–Newton method (Algorithm 3.3). Assume that r is Lipschitz continuously differentiable, J_r is uniformly full rank in a neighbourhood of $\{x^k\}_{k=0}^\infty$ and assume all steps are well-defined. Then x^k will not converge to any of the deflated points $y_1, \ldots, y_n$.*

Proof. Assume for contradiction that x^k converges to $y \in \{y_1, \ldots, y_n\}$.

Recall that the deflated step, which we will denote by $\hat{p}^k$, is

$$\hat{p}^k = \frac{\beta}{\omega}\left(p^k - \frac{r^T P r}{\beta}(J_r^T J_r)^{-1}\nabla\eta\right)$$

Recall also that the deflated step, $\hat{p}^k$, is taken when $\beta(x^k) < 1 - \epsilon$, and when $\beta(x^k) \geq 1 - \epsilon$ the undeflated step is taken with a line search applied, $\alpha\, p^k$. Thus similarly to the proof of Theorem 3.5 we need to consider the convergence behaviour of both of these steps. Again by the continuity of β (as a function of $x \in \mathbb{R}^d \backslash \{y_1, y_2, \ldots, y_n\}$), the tail of the sequence $\{x^k\}_{k=0}^\infty$ is formed by taking either all undeflated steps, p^k, or all deflated steps, $\hat{p}^k$.

First we assume that the tail of $\{x^k\}_{k=0}^\infty$ only takes undeflated steps, and the proof follows as in Theorem 3.5: as we have assumed the sequence (with a line search) converges, by Theorem 3.4, we have $J_r(x^k)^T r(x^k) \to 0$. Since J_r is uniformly full rank, this implies that $p^k \to 0$. By Definition 3.2 part (2)

$$\liminf_{k\to\infty} \langle \nabla\eta(x^k), p^k\rangle > 1. \tag{3.23}$$

By definition of the algorithm, however, this means the algorithm takes the the deflated step in the tail, a contradiction.

The remaining case to consider is that the tail of $\{x^k\}_{k=0}^\infty$ takes only deflated steps. Convergence of the x^k implies that $\liminf_{k\to\infty} \hat{p}^k = 0$. Again, by the continuity of β it must be the case that for the tail of $\{x^k\}_{k=0}^\infty$ either

$$\beta(x^k) < 0 \qquad \text{or} \tag{3.24}$$

$$0 < \beta(x^k) < 1 - \epsilon, \tag{3.25}$$

which correspond to the red and yellow regions (respectively) in Figure 2. The case $\beta = 0$ yields an undefined step.

First let us assume that property (3.24) is satisfied. Let us examine $\langle \hat{p}^k, \nabla\eta \rangle$:

$$\begin{aligned}\langle \hat{p}^k, \nabla\eta \rangle &= \frac{\beta}{\omega}\langle p^k, \nabla\eta \rangle - \frac{\|Pr\|^2}{\omega}\langle \nabla\eta, (J_r^T J_r)^{-1}\nabla\eta \rangle \\ &= \frac{\beta - \beta^2}{\omega} - \frac{\|Pr\|^2 \|J_r^{+T}\nabla\eta\|^2}{\omega} \\ &= \frac{\beta}{\omega} - 1. \end{aligned} \tag{3.26}$$

Since $\beta < 0$ and $\omega > 0$, we have that the deflated step, $\hat{p}^k$, is a descent direction for μ. Therefore we can derive the same contradiction as in the second half of the proof of Theorem 3.5.

The remaining case to consider is when the tail of $\{x^k\}_{k=0}^{\infty}$ satisfies property (3.25). Note that if $\beta < \omega$ then by (3.26) we know that $\hat{p}^k$, is a descent direction for μ, and come to the same contradiction as above. Thus we must conclude that $\beta \geq \omega$, which leads to the following:

$$\omega \leq \beta \implies \|Pr\|_2^2 \|J_r^{+T}\nabla\eta\|_2^2 \leq \beta(1-\beta). \tag{3.27}$$

From this we will deduce that $p^k \to 0$, as follows. By the definition of $\hat{p}^k$,

$$J_r p^k = \frac{\omega}{\beta} J_r \hat{p}^k + \frac{\|Pr\|_2^2}{\beta} J_r^{+T}\nabla\eta, \tag{3.28}$$

since $J_r^{+T} = J_r(J_r^T J_r)^{-1}$. Taking norms and using the inequalities in (3.28), we have

$$\begin{aligned}\|J_r p^k\|_2 &\leq \frac{\omega}{\beta}\|J_r\hat{p}^k\|_2 + \frac{\|Pr\|_2^2 \|J_r^{+T}\nabla\eta\|_2}{\beta} \\ &\leq \|J_r\hat{p}^k\|_2 + \frac{1-\beta}{\|J_r^{+T}\nabla\eta\|_2} \\ &\leq \|J_r\|_2 \left(\|\hat{p}^k\|_2 + \frac{1}{\|\nabla\eta\|_2} \right). \end{aligned} \tag{3.29}$$

The last line follows from $\beta > 0$ and $\|\nabla\eta\|_2 = \|J_r^T J_r^{+T}\nabla\eta\|_2 \leq \|J_r\|_2 \|J_r^{+T}\nabla\eta\|_2$.

The term in equation (3.29) tends to zero because $\|J_r(x^k)\|_2$ is uniformly bounded by continuity of J_r, and because $\lim_{x\to y}\|\nabla\eta(x)\|_2 = \infty$ by Definition 3.2 part (3). Because J_r is uniformly full rank, $J_r p^k \to 0$ implies $p^k \to 0$. However, $p^k \to 0$ implies (3.24) by Definition 3.2 part (2), which is a contradiction. ∎

3.6. Gauss–Newton comparison. In this section we have introduced the "good" and "bad" deflated Gauss–Newton methods. The two big advantages of these methods over the deflated Newton method are that each step is computationally cheaper due to the fact that they do not require the calculation of the Hessian of f, and that they only converge to local minima, meaning that fewer deflations are required to find all local minima.

There are also advantages and disadvantages between the two different deflated Gauss–Newton methods we have discussed in this section. We note that each step of the "good" method is cheaper than the "bad" method, but it is possible that the extra term in the deflation step means that the method may require fewer iterations to converge. In general we have not found that one method or the other is consistently faster than the other.

4. Implementation and experiments. We will first discuss a couple two dimensional examples, then move onto some high dimensional problems, and finally we will investigate a physical example coming from an Inelastic Neutron Scattering experiment. We will cover a range of zero and non-zero residual problems.

The value of ϵ, discussed in Figure 2, is 0.01 in all of the following experiments. We used a quadratic line search whenever a line search was applied. The least squares problems are solved using `lsqminnorm` for the "good" method and by a QR factorization for the "bad" method. This QR factorization is recycled to compute the extra terms in the deflated Gauss–Newton method. Open source MATLAB code for all experiments can be found at https://github.com/AlbanBloorRiley/DeflatedGaussNewton.

4.1. A test problem with many local minima. This is a nonlinear least squares problem in two variables with 42 local minima (143 stationary points), defined using a truncated product expansion in cos and sin:

$$r(x,y) = \begin{pmatrix} a\displaystyle\prod_{k=1}^{3} 1 - \frac{(x+y)^2}{k^2\pi^2} \\ a\displaystyle\prod_{k=1}^{3} 1 - \frac{(x-y)^2}{(k-1/2)^2\pi^2} \\ a + 0.01(x^2+y^2) \end{pmatrix} \tag{4.1}$$

The parameter a defines defines the height of the objective function, which can be used to test how the various methods perform with potentially large residual problems. In all the experiments here $a = 10$. As seen in Figure 3 the Newton method requires 143 deflations to find all of the stationary points, and so guarantee finding all the local minima. Both the "good" and "bad" Gauss–Newton methods find all the local minima in just 42 deflations. Only the plot from the "good" Gauss–Newton method is shown as the result is almost identical to the one produced by the "bad" method.

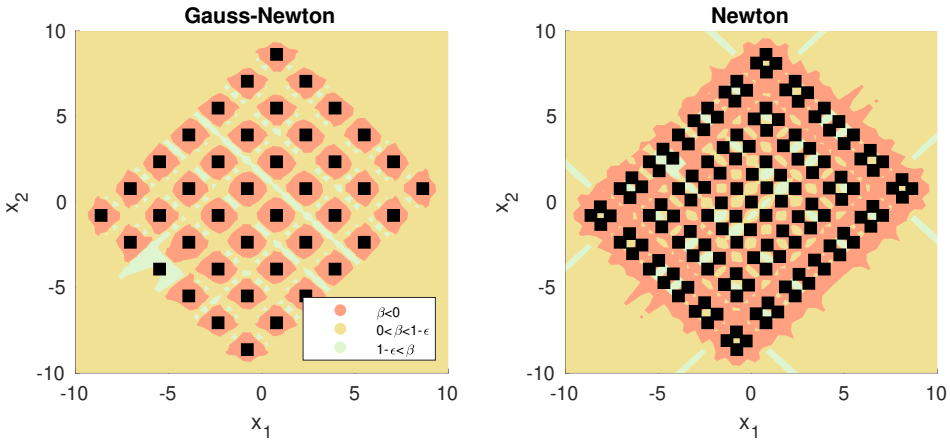

FIG. 3. *Multiple minima found to the nonlinear least squares problem given by (4.1). Left: all 42 minima found by the "good" Gauss–Newton algorithm. Right: the 143 stationary points found by using the Newton method for optimization algorithm. Note that in the green region $1-\epsilon < \beta$, in the yellow region $0 < \beta < 1-\epsilon$, and in the red region $\beta < 0$, for $\epsilon = 0.01$.*

We compare the convergence rates of all three methods discussed in this paper in Figure 4, clearly showing that all three methods have a quadratic local convergence rate after period of non-convergence. It also shows that the number of iterations required for each deflation can vary wildly.

We may also use this example to compare with other global minimization methods, such as Matlab's 'MultiStart' algorithm, from the global minimization toolbox [16].

Using MultiStart with the 'lsqnonlin' solver on the unconstrained problem only finds a handful of minima, even when 10000 start points are used. When the method is constrained to the 20 by 20 square, centred about the origin, MultiStart finds all 42 minima consistently with only 300 start points; when constrained to the 40 by 40 square however not all 42 minima are consistently found.

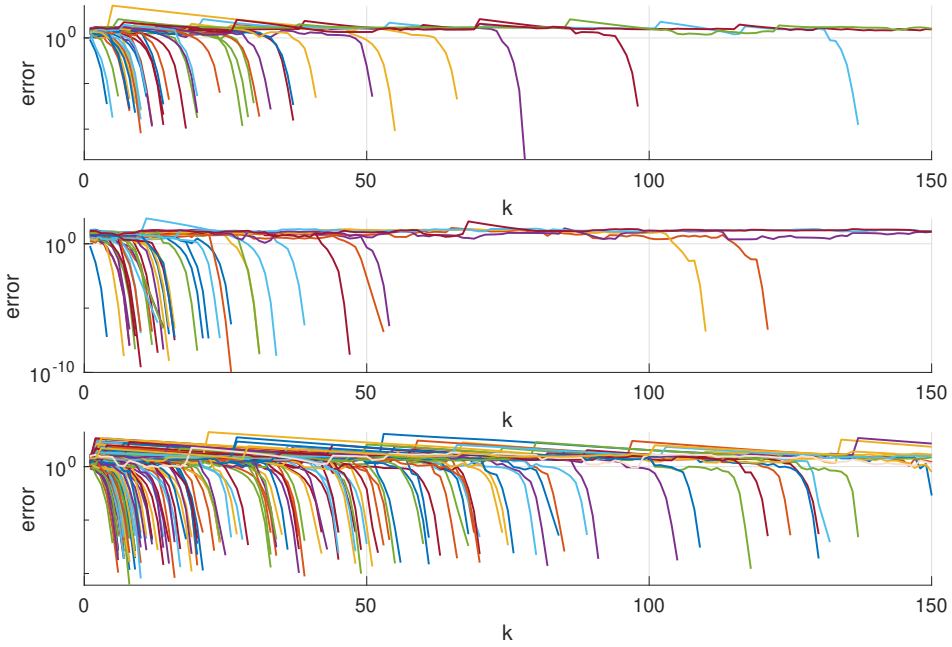

FIG. 4. *Comparison of the convergence behaviour for computing all solutions to the nonlinear least squares problem given by (4.1). Top: "good" Gauss–Newton (42 local minima). Middle: "bad" Gauss–Newton (42 local minima). Bottom: Newton for Optimization (143 stationary points). All methods started from an initial guess of* `[1;3]`.

The bounded MultiStart method still requires many more functions evaluations than the Deflated Gauss–Newton methods as can be seen in Figure 5, even when lsqnonlin is set up to use the Jacobian not just use a finite difference approximation. The figure also shows how the Gauss-newton methods are faster with respect to computation time.

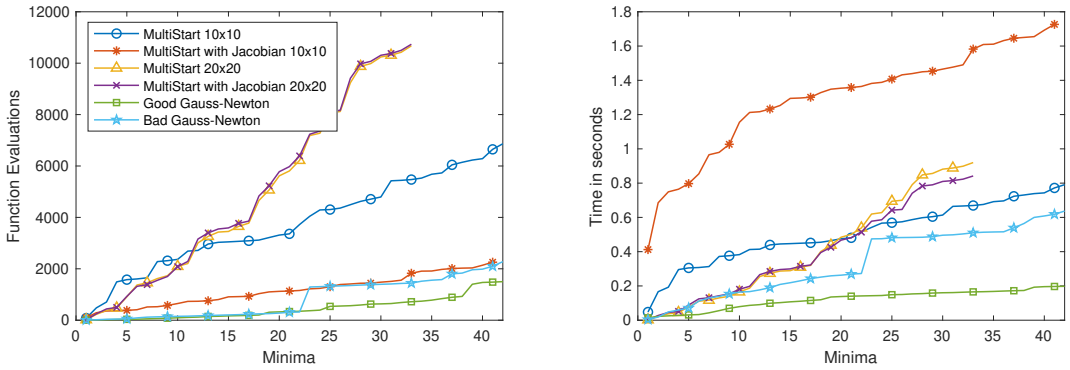

FIG. 5. *Left: Cumulative number of function evaluations required for each method to find all 42 local minima. Right: cumulative time to compute the local minima on a 2021 M1 Macbook Pro. For the multistart methods 300 initial points are used, with function evaluations/timings including all the points tried up until the 42nd local minimum is found.*

4.2. Multiple solutions of nonlinear BVPs by Fourier extension. Fourier extension is a technique to approximate functions on an arbitrary bounded set $\Omega \subset \mathbb{R}^d$ using Fourier series that are periodic on a box $B = [a_1, b_1] \times \cdots \times [a_d, b_d]$ containing Ω. We stick to the case $d = 1$, $\Omega = [0, 1]$ and $B = [-1, 1]$ in what follows. For a smooth function $f : [0, 1] \to \mathbb{R}$, a Fourier extension approximation

is given by

$$f_N(x) = \sum_{j=-n}^{n} c_j \mathrm{e}^{\mathrm{i}j\pi x}, \tag{4.2}$$

where $N = 2n + 1$. Note that f_N is periodic on $[-1, 1]$.

Fourier extensions inherit several nice properties from the Fourier basis: we can find the derivative of f_N by simply multiplying its coefficients by $\mathrm{i}j\pi$, we can evaluate $f_N(x_k)$ on an equispaced grid

$$\{x_k = k/m : k = 0, \dots, m\}, \tag{4.3}$$

where $m \geq 2n$ in $\mathcal{O}(m \log m)$ operations by the Inverse Fast Fourier Transform (IFFT), they have spectral approximation properties [26], and we can compute convolutions in $\mathcal{O}(N \log N)$ operations [28]. However, computation of the coefficients c_j is extremely ill-conditioned because there exist nonzero coefficients such that f_N approximates 0 on $[0, 1]$ while its values on $[-1, 0]$ are unconstrained. Despite this ill-conditioning, adequate coefficients c_j can be computed stably by solving an oversampled least squares collocation problem at equally spaced points in $[0, 1]$, [2, 3, 17, 18].

In the following examples we discretize and solve nonlinear Boundary Value Problems (BVPs) as oversampled nonlinear least squares collocation problems at equally spaced points in $[0, 1]$. We will give two examples to show that the "good" and "bad" deflated Gauss–Newton methods can be effective in the cases where there are multiple isolated solutions to a nonlinear boundary value problem.

There are two ways in which the resulting optimization problems are more complicated than those presented so far in the paper. First, the solution is a complex-valued vector $\mathbf{c}$, which might appear to cause problems for the condition $\langle \nabla\eta, p^k \rangle > \epsilon$ in our algorithms. The analogous condition in complex arithmetic for measuring to what extent p^k is a direction of ascent for η, is $\mathrm{Re}\langle \nabla\eta, p^k \rangle > \epsilon$. Second, the norm used to measure distance in the deflation operator should not be the 2-norm of $\mathbf{c}$. This is because there are many different coefficient vectors that yield approximately the same function, so deflation measuring distance between coefficient vectors will not be particularly effective. We instead use the norm

$$\|\mathbf{c}\|_{\mathrm{FE}} := \left(\sum_{k=0}^{m} \left| \sum_{j=-n}^{n} c_j \mathrm{e}^{\mathrm{i}j\pi x_k} \right|^2 \right)^{1/2}, \tag{4.4}$$

which is the 2-norm of the function values on the equispaced grid (4.3). This means that the deflation process measures distance between coefficients by measuring the difference between the associated function values.

4.2.1. Bratu equation. The Bratu equation is given by [25, Chapter 16]:

$$u'' + 3\exp(u) = 0, \qquad u(0) = 0, \qquad u(1) = 0 \tag{4.5}$$

After discretization, we minimize the following nonlinear least squares residual for $\mathbf{c} \in \mathbb{C}^N$:

$$\begin{aligned} f(\mathbf{c}) = &\frac{1}{2} \sum_{k=0}^{m} \left| \sum_{j=-n}^{n} -j^2\pi^2 c_j \mathrm{e}^{\mathrm{i}j\pi x_k} + 3\exp\left(\sum_{j=-n}^{n} c_j \mathrm{e}^{\mathrm{i}j\pi x_k} \right) \right|^2 \\ &+ \frac{1}{2} \left| \sum_{j=-n}^{n} c_j \right|^2 + \frac{1}{2} \left| \sum_{j=-n}^{n} c_j \, \mathrm{i}^j \right|^2. \end{aligned} \tag{4.6}$$

The first sum corresponds to enforcing the ODE at the collocation points x_k (defined in equation (4.3)) and the last two terms correspond to enforcing the two boundary conditions.

In the experiment depicted in Figure 6, we set $n = 100$, $m = 400$, so that there 403 least squares constraints and 201 unknowns. The initial guess is the zero function, both initially, and each time we deflate.

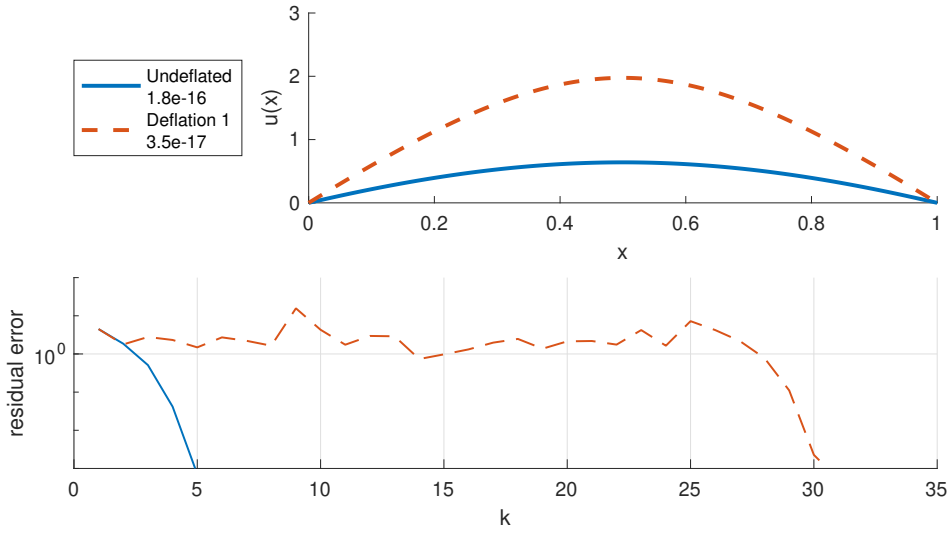

FIG. 6. *Multiple solutions of the Bratu equation (4.5), computed by solving the nonlinear least squares problem (4.6). Top: plots of the functions that the "good" Deflated Gauss–Newton algorithm found. Bottom: the residual against each iteration.*

The convergence behaviour seen in Figure 6 is typical, and the analogous plot using the Bad deflated Gauss–Newton method is almost the same. What we see for the deflated iteration is that there is a long period of non-convergence, before finding a basin of attraction and converging at the same rate as the undeflated Gauss–Newton method in that basin. One reason why this non-convergence happens is because the initial guess for the iteration is set to the be the same as it was for the first undeflated iteration (which in this case is zero). The local convergence is the same as for the undeflated Gauss–Newton method by design — at a new local minima x we must have $\langle \nabla \eta(x), p(x) \rangle = 0$, so in a neighbourhood of the new local minima we must have $\langle \nabla \eta(x), p(x) \rangle \leq \epsilon$, so the undeflated step is taken.

4.2.2. Carrier equation. The Carrier equation [25, Chapter 16], is

$$0.05\, u'' + 8x(1-x)u + u^2 = 1, \qquad u(0) = 0, \qquad u(1) = 0 \tag{4.7}$$

Similarly to the Bratu example, we minimize the following nonlinear least squares residual for $\mathbf{c} \in \mathbb{C}^N$:

$$\begin{aligned} f(\mathbf{c}) = &\frac{1}{2} \sum_{k=0}^{m} \left| \sum_{j=-n}^{n} \left(-0.05\, j^2 \pi^2 + 8x_k(1-x_k)\right) c_j \mathrm{e}^{\mathrm{i}j\pi x_k} + \left(\sum_{j=-n}^{n} c_j \mathrm{e}^{\mathrm{i}j\pi x_k} \right)^2 - 1 \right|^2 \\ &+ \frac{1}{2} \left| \sum_{j=-n}^{n} c_j \right|^2 + \frac{1}{2} \left| \sum_{j=-n}^{n} c_j\, \mathrm{i}^j \right|^2 . \end{aligned} \tag{4.8}$$

In the experiment depicted in Figure 7, we set $n = 100$, $m = 400$, so that there are 403 least squares

constraints and 201 unknowns. The initial guess is the zero function, both initially, and each time we deflate.

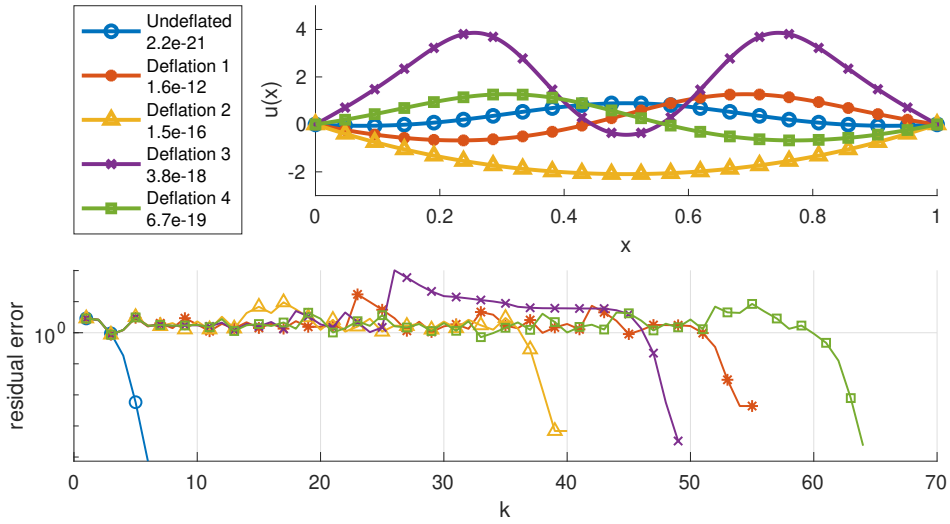

FIG. 7. *Multiple solutions of the carrier equation (4.7), computed by solving the nonlinear least squares problem (4.8). Top: plots of the functions that the "good" Deflated Gauss–Newton algorithm found. Bottom: the residual against each iteration.*

Interestingly, the fourth solution found in Figure 7 is distinct from the solutions found in [25]. The solution persists when we increase the degrees of freedom in the approximation, and the residual is very small, so we conclude that this is likely a proper solution of the differential equation that was missed, and not a feature of the discretization.

4.3. Inverse eigenvalue problems. One application where the deflated methods outlined in this paper are applicable is to the solution of inverse eigenvalue problems (IEP) that arise from spectroscopic techniques, such as inelastic neutron scattering (INS) experiments [15]. INS can be used to accurately measure magnetic excitations in materials that possess interacting electron spins. When studying finite spin systems, such as single ions or molecular-based magnets, INS is used to experimentally quantify the energy between quantum spin states that relate directly to eigenvalues of a Hamiltonian that describes the quantum spin dynamics of the compound [11, 4, 1]. Such information is of both fundamental and technological importance for further understanding quantum phenomena and how electronic quantum spins could be implemented for novel quantum applications such as information processes and sensing. Accurate quantification of the quantum spin dynamics of magnetic molecules following an INS experiment requires determining parameters associated with an appropriate spin Hamiltonian model. The spin Hamiltonian operator that gives rise to these experimentally determined eigenvalues is to be computed by fitting, such as nonlinear least squares.

DEFINITION 4.1 (Inverse Eigenvalue Problem [7]). *Given complex numbers $\lambda_1, \ldots, \lambda_n$ and a basis of matrices $A_0, \ldots, A_\ell \in \mathbb{C}^{n\times n}$, let*

$$r(x) = \begin{pmatrix} \lambda_1(x) - \lambda_1 \\ \vdots \\ \lambda_\ell(x) - \lambda_\ell \end{pmatrix}. \tag{4.9}$$

and $\lambda_1(x), \ldots \lambda_\ell(x)$ be the eigenvalues of the matrix

$$A(x) = A_0 + \sum_{i=1}^{\ell} x_i A_i. \tag{4.10}$$

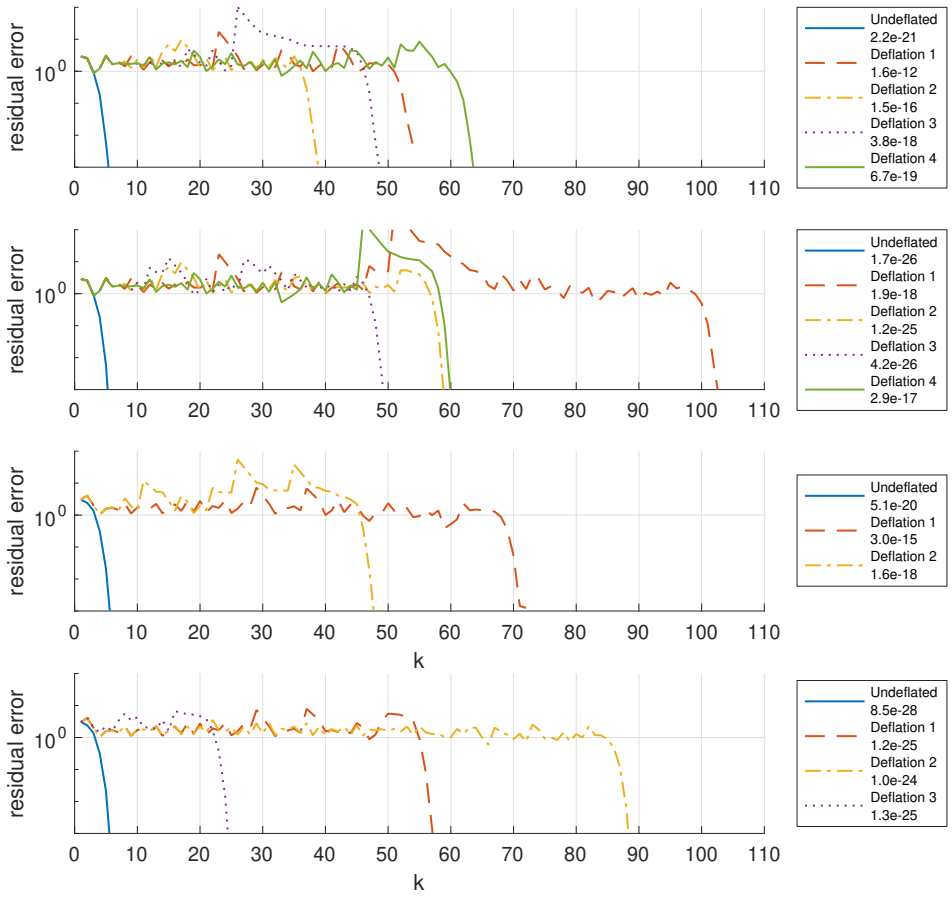

FIG. 8. *Comparison of the convergence behaviour for computing multiple solutions to the Carrier equation. Top: "good" Gauss–Newton with initial guess* $\mathbf{c} = \mathbf{0}$. *Second: "bad" Gauss–Newton with initial guess* $\mathbf{c} = \mathbf{0}$. *Third: "good" Gauss–Newton with initial guess* $u(x) = x(1-x)$. *Bottom: "bad" Gauss–Newton with initial guess* $u(x) = x(1-x)$. *We do not mean to imply any value judgement on "good" or "bad" deflation techniques' convergence properties for this problem, merely that the initial period of non-convergence can vary wildly in length, and in unpredictable ways.*

Then the inverse eigenvalue problem is to find the parameters $x \in \mathbb{R}^\ell$ *that minimize*

$$F(x) = \frac{1}{2}||r(x)||_2^2. \tag{4.11}$$

For the INS example that we will look the $\lambda_1, \ldots, \lambda_\ell$ are the experimental eigenvalues and calculated from the experimental data and the basis matrices are Stevens operators[2]. The operators are all calculated using the easyspin MATLAB package [24].

4.3.1. Mn12. The selected example concerns the INS spectrum of Manganese-12-acetate. This molecule gained importance following the identification that it acts like a nano-sized magnet with a molecular magnetic coercivity and the identification of quantum tunnelling of magnetisation see [10, 23]. The INS spectrum of Manganese-12-acetate was measured in order to obtain a precise description of an appropriate Spin Hamiltonian model that accurately describes its magnetic properties. The Hamiltonian of the system is a 21×21 matrix and and can be modelled using 4 basis matrices [5]:

$$A(B_2^0, B_4^0, B_2^2, B_4^4) = B_2^0 O_2^0 + B_4^0 O_4^0 + B_2^2 O_2^2 + B_4^4 O_4^4 \in \mathbb{R}^{21\times 21} \tag{4.12}$$

[2]See [12] and [22] for more general details on Stevens operators.

Where the Bs are the parameters to be found and the Os are the Stevens operators are defined, in this case with a spin of $S = 10$, as:

$$\begin{aligned} O_2^0 &= 3S_z^2 - X \\ O_2^2 &= \frac{1}{2}(S_+^2 + S_-^2) \\ O_4^0 &= 35S_z^4 - (30X - 25)S_z^2 + (3X^2 - 6X) \\ O_4^4 &= \frac{1}{2}(S_+^4 + S_-^4) \end{aligned}$$

where $X = S(S+1)I \in \mathbb{R}^{S(S+1)\times S(S+1)}$, and

$$\begin{aligned} (S_z)_{k,k} &= (S+1-k) \\ (S_+)_{k,k+1} &= \sqrt{k(2S+1-k)} \\ (S_-)_{k+1,k} &= \sqrt{k(2S+1-k)} \end{aligned}$$

Four distinct parameter sets are given as solutions to this problem, as found in [5]. Since we do not have access to the original experimental data the eigenvalues used were reverse engineered from the solutions given in [5]. The experiment depicted in Figure 9 shows the convergence rates of the three different methods applied to the inverse eigenvalue problem (4.9).

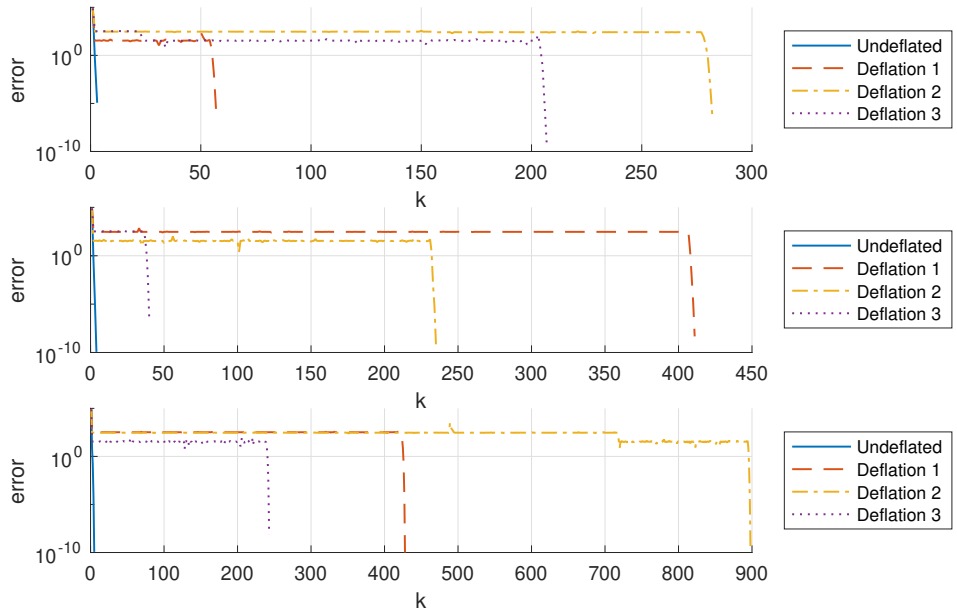

FIG. 9. *Comparison of the convergence behaviour for computing all four solutions to the Mn_{12} inverse eigenvalue problem. Top: "good" Gauss–Newton. Middle: "bad" Gauss–Newton. Bottom: Newton for Optimization.*

5. Conclusions. We have presented two new deflated optimization algorithms, both based on the Gauss–Newton method. Both algorithms find local minima, without converging to other stationary points, thus reducing the number of deflations needed compared to the deflated Newton method applied to the first order optimality conditions. They also do not require the calculation of a Hessian matrix, allowing the application of the method to a wider range of problems. We then showed how these methods can be effective for problems with a high dimension or a high number of local minima. We limited ourselves to cases where J_r is full rank, a constraint which could be lifted in future work. It is not immediately apparent which of the "good" or "bad" methods is best, but both have been shown

to have locally quadratic convergence to local minima comparable to the undeflated Gauss–Newton method (under similar assumptions). Lastly, we have shown that these methods can be used to solve least squares inverse eigenvalue problems arising from Inelastic Neutron Scattering experiments.

Acknowledgements. We are grateful to Nick Higham, Jonas Latz and Françoise Tisseur for feedback on early drafts of this paper. MW thanks the Polish National Science Centre (SONATA-BIS-9), project no. 2019/34/E/ST1/00390, for the funding that supported some of this research. ABR thanks the University of Manchester for a Dean's Doctoral Scholarship.

REFERENCES

[1] *Spectroscopy Methods for Molecular Nanomagnets*, in Structure and Bonding, Springer Berlin Heidelberg, Berlin, Heidelberg, 2014, pp. 231–291, https://doi.org/10.1007/430_2014_155, https://link.springer.com/10.1007/430_2014_155. ISSN: 0081-5993, 1616-8550.

[2] B. Adcock and D. Huybrechs, *Frames and numerical approximation*, SIAM Rev., 61 (2019), pp. 443–473. SIAM.

[3] B. Adcock and D. Huybrechs, *Frames and numerical approximation II: generalized sampling*, J. Fourier Anal. Appl., 26 (2020), p. 87. Springer.

[4] M. Baker and H. Mutka, *Neutron spectroscopy of molecular nanomagnets*, Eur. Phys. J. Spec. Top., 213 (2012), pp. 53–68, https://doi.org/10.1140/epjst/e2012-01663-6, http://link.springer.com/10.1140/epjst/e2012-01663-6. Publisher: Springer Science and Business Media LLC.

[5] R. Bircher, G. Chaboussant, A. Sieber, H. U. Güdel, and H. Mutka, *Transverse magnetic anisotropy in Mn 12 acetate: Direct determination by inelastic neutron scattering*, Phys. Rev. B, 70 (2004), p. 212413, https://doi.org/10.1103/PhysRevB.70.212413.

[6] K. M. Brown and W. B. Gearhart, *Deflation techniques for the calculation of further solutions of a nonlinear system*, Numer. Math., 16 (1971), pp. 334–342, https://doi.org/10.1007/BF02165004.

[7] M. T.-C. Chu and G. H. Golub, *Inverse Eigenvalue problems: theory, algorithms, and applications*, Numerical mathematics and scientific computation, OUP, Oxford ; New York, 2005. OCLC: ocm59877517.

[8] P. E. Farrell, Á. Birkisson, and S. W. Funke, *Deflation Techniques for Finding Distinct Solutions of Nonlinear Partial Differential Equations*, SIAM J. Sci. Comp., 37 (2015), pp. A2026–A2045, https://doi.org/10.1137/140984798.

[9] S. Friedland, J. Nocedal, and M. L. Overton, *The Formulation and Analysis of Numerical Methods for Inverse Eigenvalue Problems*, SIAM J. Numer. Anal., 24 (1987), pp. 634–667, https://doi.org/10.1137/0724043.

[10] J. R. Friedman, M. P. Sarachik, J. Tejada, and R. Ziolo, *Macroscopic Measurement of Resonant Magnetization Tunneling in High-Spin Molecules*, Phys. Rev. Lett., 76 (1996), pp. 3830–3833, https://doi.org/10.1103/physrevlett.76.3830, https://link.aps.org/doi/10.1103/PhysRevLett.76.3830. Publisher: American Physical Society (APS).

[11] A. Furrer and O. Waldmann, *Magnetic cluster excitations*, Rev. Mod. Phys., 85 (2013), pp. 367–420, https://doi.org/10.1103/RevModPhys.85.367, https://link.aps.org/doi/10.1103/RevModPhys.85.367.

[12] D. Gatteschi, R. Sessoli, and J. Villain, *Molecular nanomagnets*, no. 5 in Mesoscopic physics and nanotechnology, OUP, Oxford ; New York, 2006. OCLC: ocm62133760.

[13] S. Güttel, Y. Nakatsukasa, M. Webb, and A. B. Riley, *A Sherman–Morrison–Woodbury approach to solving least squares problems with low-rank updates*, 2024, https://doi.org/10.48550/ARXIV.2406.15120. Version Number: 2.

[14] D. M. Himmelblau, *Applied nonlinear programming*, McGraw-Hill, New York Düsseldorf, 1972.

[15] S. W. Lovesey, *Theory of neutron scattering from condensed matter. 2: Polarization effects and magnetic scattering*, no. 72 in International series of monographs on physics, Clarendon Pr, Oxford, repr ed., 2003.

[16] T. MathWorks, *Global Optimization Toolbox version: 24.1 (R2024a)*, 2024, https://www.mathworks.com.

[17] R. Matthysen and D. Huybrechs, *Fast algorithms for the computation of Fourier extensions of arbitrary length*, SIAM J. Sci. Comp., 38 (2016), pp. A899–A922. SIAM.

[18] R. Matthysen and D. Huybrechs, *Function approximation on arbitrary domains using Fourier extension frames*, SIAM J. Numer. Anal., 56 (2018), pp. 1360–1385. SIAM.

[19] C. D. Meyer, *Generalized Inversion of Modified Matrices*, SIAM J. Appl. Math., 24 (1973), pp. 315–323, http://www.jstor.org/stable/2099767. SIAM.

[20] J. Nocedal and S. J. Wright, *Numerical optimization*, Springer, New York, 2nd ed., 2006.

[21] I. P. A. Papadopoulos, P. E. Farrell, and T. M. Surowiec, *Computing Multiple Solutions of Topology Optimization Problems*, SIAM J. Sci. Comp., 43 (2021), pp. A1555–A1582, https://doi.org/10.1137/20M1326209.

[22] C. Rudowicz and C. Y. Chung, *The generalization of the extended Stevens operators to higher ranks and spins, and a systematic review of the tables of the tensor operators and their matrix elements*, J. Phys.: Condens. Matter, 16 (2004), pp. 5825–5847, https://doi.org/10.1088/0953-8984/16/32/018.

[23] R. Sessoli, D. Gatteschi, A. Caneschi, and M. A. Novak, *Magnetic bistability in a metal-ion cluster*, Nature, 365 (1993), pp. 141–143, https://doi.org/10.1038/365141a0, https://www.nature.com/articles/365141a0. Publisher: Springer Science and Business Media LLC.

[24] S. Stoll and A. Schweiger, *EasySpin, a comprehensive software package for spectral simulation and analysis in EPR*, J. Magn. Reson., 178 (2006), pp. 42–55, https://doi.org/10.1016/j.jmr.2005.08.013.
[25] L. N. Trefethen, Á. Birkisson, and T. A. Driscoll, *Exploring ODEs*, SIAM, 2017.
[26] M. Webb, V. Coppé, and D. Huybrechs, *Pointwise and uniform convergence of Fourier extensions*, Constr. Approx., 52 (2020), pp. 139–175. Springer.
[27] J. H. Wilkinson, *Rounding errors in algebraic processes*, Notes on applied science no. 32, Prentice-Hall, 1963.
[28] K. Xu, A. P. Austin, and K. Wei, *A fast algorithm for the convolution of functions with compact support using Fourier extensions*, SIAM J. Sci. Comp., 39 (2017), pp. A3089–A3106. SIAM.

Appendix A. Proof of Theorem 2.2.

Proof. The Newton step for the deflated problem, $\hat{p}^k$, is given by

$$\hat{p}^k = -J_{\mu r}(x^k)^{-1}\mu(x^k)r(x^k). \tag{A.1}$$

The Jacobian of the deflated system can be expanded using the product rule,

$$J_{\mu r}(x) = \mu(x)J_r(x) + r(x)\nabla\mu(x)^T. \tag{A.2}$$

For ease of notation, let us write

$$\mu(x)^{-1}J_{\mu r}(x) = J_r(x) + r(x)\nabla\eta(x)^T,$$

where $\eta(x) = \log(\mu(x))$, so $\nabla\eta = \mu^{-1}\nabla\mu$. The Sherman–Morrison formula implies

$$J_{\mu r}(x)^{-1}\mu(x) = J_r(x)^{-1} - \frac{J_r(x)^{-1}r(x)\nabla\eta(x)^T J_r(x)^{-1}}{1 + \nabla\eta(x)^T J_r(x)^{-1}r(x)}. \tag{A.3}$$

Using the undeflated step $p^k = -J_r(x^k)^{-1}r(x_k)$, we can write

$$J_{\mu r}(x^k)^{-1}\mu(x^k) = J_r(x^k)^{-1} + \frac{p^k\nabla\eta(x^k)^T J_r(x^k)^{-1}}{1 - \langle\nabla\eta(x), p^k\rangle}. \tag{A.4}$$

Now we can substitute this into the formula for $\hat{p}^k$, and use the formula for p^k to obtain

$$\hat{p}^k = -J_r(x^k)^{-1}r(x^k) - \frac{p^k\nabla\eta(x^k)^T J_r(x^k)^{-1}r(x^k)}{1 - \langle\nabla\eta(x^k), p^k\rangle} = (1 - \langle\nabla\eta(x^k), p^k\rangle)^{-1}p^k, \tag{A.5}$$

which is equivalent to the deflated Newton step given in Algorithm 2.2. ∎