Bernal, F. (2016). Trust-region methods for nonlinear elliptic equations with radial basis functions. *Computers & Mathematics with Applications*, 72(7), 1743–1763.

DOI: https://doi.org/10.1016/j.camwa.2016.07.014

Trust-Region Methods for Nonlinear Elliptic Equations with Radial Basis Functions

Francisco Bernal [*] [†]

Abstract

We consider the numerical solution of nonlinear elliptic boundary value problems with Kansa's method. We derive analytic formulas for the Jacobian and Hessian of the resulting nonlinear collocation system and exploit them within the framework of the trust-region algorithm. This ansatz is tested on semilinear, quasilinear and fully nonlinear elliptic PDEs (including Plateau's problem, Hele-Shaw flow and the Monge-Ampère equation) with excellent results. The new approach distinctly outperforms previous ones based on linearization or finite-difference Jacobians.

Keywords. Kansa's method, radial basis function, nonlinear elliptic PDE, trust-region method, Monge-Ampère, Plateau's problem, p-Laplacian.

1 Introduction

1.1 RBF interpolation

Given the scalar data $u_1, \ldots, u_N$ on a set (called *pointset*) of distinct points $\mathbf{x}_1, \ldots, \mathbf{x}_N \in \mathbb{R}^d$ (called *centres*), the RBF interpolant is defined as

$$\tilde{u}(\mathbf{x}) = \sum_{j=1}^{N} \alpha_j \phi(\| \mathbf{x} - \mathbf{x}_j \|), \tag{1}$$

where the function $\phi(r) : [0, \infty) \to \mathbb{R}$ is the chosen radial basis function (RBF). A few popular RBFs are shown in Table 1. Thoughout this paper, $\| \cdot \|$ is always the 2-norm. The coefficients $\alpha_1, \ldots, \alpha_N$ are determined by collocation

$$\begin{bmatrix} \phi(\|\mathbf{x}_1 - \mathbf{x}_1\|) & \ldots & \phi(\|\mathbf{x}_1 - \mathbf{x}_N\|) \\ \vdots & \ddots & \vdots \\ \phi(\|\mathbf{x}_N - \mathbf{x}_1\|) & \ldots & \phi(\|\mathbf{x}_N - \mathbf{x}_N\|) \end{bmatrix} \begin{bmatrix} \alpha_1 \\ \vdots \\ \alpha_N \end{bmatrix} = \begin{bmatrix} u_1 \\ \vdots \\ u_N \end{bmatrix}, \tag{2}$$

[*]INESC-ID\IST, TU Lisbon. Rua Alves Redol 9, 1000-029 Lisbon, Portugal.

[†]Center for Mathematics and its Applications, Department of Mathematics, Instituto Superior Técnico. Av. Rovisco Pais 1049-001 Lisbon, Portugal. (francisco.bernal@ist.utl.pt)

or, more compactly, $[\phi]\vec{\alpha} = \vec{u}$. Thanks to the radial argument of ϕ, $[\phi]$–called the *RBF interpolation matrix*–is symmetric. Guaranteed non-singularity of $[\phi]$ depends on the RBF ϕ being strictly conditionally positive definite (SCPD)–i.e. bound to yield positive-definite $[\phi]$. For instance, in Table 1 all the RBFs are SCPD except for the multiquadric, where the RBF interpolant needs to be augmented with a constant to yield a positive definite $[\phi]$ [13, 27].

RBFs used in this paper

RBF	$\phi(r)$	notation	support	convergence rate
multiquadric	$\sqrt{r^2+c^2}$	MQ(c)	$r \leq \infty$	spectral
inverse multiquadric	$1/\sqrt{r^2+c^2}$	IMQ(c)	$r \leq \infty$	spectral
Matérn	$(r/c)^{(\alpha-d)/2} K_{(\alpha-d)/2}(r/c)$	MATERN(α, c)	$r \leq \infty$	spectral
Wendland C^4	$[1-(r/L)]_+^{s+2} P(r/L, s)$	WC4(L)	$r \leq L$	algebraic

Table 1: d is the space dimension ($\mathbf{x} \in \mathbb{R}^d$). In MATERN($\alpha, c$), $K_\nu(t)$ is the modified Bessel function of the second kind. In WC4(L), $[f(r)]_+ = 0$ if $r \geq L$, $s = 3 + \lfloor d/2 \rfloor$, and $P(t,s) = (s^2+4s+3)t^2 + (3s+6)t + 3$ (WC4 works up to $d = 3$).

1.2 Kansa's method

In 1990, Kansa adapted this approach to the solution of linear boundary value problems (BVPs) [19, 20]. Consider the elliptic BVP

$$\begin{cases} \mathcal{L}^{PDE} u(\mathbf{x}) = f, & \text{if } \mathbf{x} \in \Omega \\ \mathcal{L}^{BC} u(\mathbf{x}) = g, & \text{if } \mathbf{x} \in \partial\Omega, \end{cases} \tag{3}$$

where Ω is a bounded domain in $\mathbb{R}^d$, $d \geq 1$, $u : \Omega \to \mathbb{R}$ is smooth, and $\mathcal{L}^{PDE}$ and $\mathcal{L}^{BC}$ are the interior and boundary linear operators, respectively. Kansa's idea was to discretize $\Omega \cup \partial\Omega$ into a pointset $\Xi_N = \{\mathbf{x}_i\}_{i=1}^N$, and look for an approximation $\tilde{u}$ to u with an RBF interpolant like (1). Without loss of generality, we can assume that the first M nodes in Ξ_N belong to the interior of Ω and the last $N - M$ are discretizing its boundary. By linearity, collocation of (3) on that interpolant leads to

$$[\mathcal{L}\phi]\vec{\alpha} := \begin{bmatrix} [\mathcal{L}^{PDE}\phi]_\Omega \\ [\mathcal{L}^{BC}\phi]_{\partial\Omega} \end{bmatrix} \vec{\alpha} := \begin{bmatrix} \mathcal{L}^{PDE}\phi_{11} & \dots & \mathcal{L}^{PDE}\phi_{1N} \\ \vdots & \ddots & \vdots \\ \mathcal{L}^{PDE}\phi_{M1} & \dots & \mathcal{L}^{PDE}\phi_{MN} \\ \mathcal{L}^{BC}\phi_{M+1,1} & \dots & \mathcal{L}^{BC}\phi_{M+1,N} \\ \vdots & \ddots & \vdots \\ \mathcal{L}^{BC}\phi_{N1} & \dots & \mathcal{L}^{BC}\phi_{NN} \end{bmatrix} \vec{\alpha} = \begin{bmatrix} f(\mathbf{x}_1) \\ \vdots \\ f(\mathbf{x}_M) \\ g(\mathbf{x}_{M+1}) \\ \vdots \\ g(\mathbf{x}_N) \end{bmatrix}. \tag{4}$$

(Check Section 1.7 for the notation.) This method for solving PDEs has many appealing features: it is meshless,very easy to code, appropriate for

high-dimensional PDEs (thanks to the radial argument of the RBFs, which is dimension-blind) and–as long as the solution is smooth–enjoys exponential convergence with respect to the fill distance of the pointset Ξ_N (for many RBFs at least, see Table 1). For a complete exposition, the reader is referred to [13]. Regarding solvability, conditions which guarantee that the *differentiation matrix* in (4) be nonsingular have not yet been established. In fact, there are crafted examples which yield a singular matrix [18]), but such cases should be exceedingly rare, as also confirmed by years of praxis. On the other hand, Kansa's method may lead to very ill-conditioned matrices, meaning that only pointsets with up to a few thousands of nodes can be used before the matrix in (4) becomes numerically singular. Larger problems can be tackled by using compactly supported RBFs such as WC4 in Table 1 (at the expense of sacrificing spectral convergence), by the RBF-QR method [16] (for some RBFs), and/or by using the novel RBF-partition of unity method [21].

1.3 Nonlinear equations

Extending Kansa's method to nonlinear equations is straightforward. Let us introduce the following compact notation for a nonlinear elliptic BVP:

$$\mathcal{W}[\mathbf{x}, u(\mathbf{x}), Du(\mathbf{x})] = 0 \Rightarrow \begin{cases} \mathcal{W}^{PDE} = 0, & \text{if } x \in \Omega \\ \mathcal{W}^{BC} = 0, & \text{if } x \in \partial\Omega, \end{cases} \tag{5}$$

where $Du(\mathbf{x})$ is shorthand notation for any kind of derivatives present in (5), such as $\partial/\partial x, \nabla^2$, etc. Collocation of (1) on (5) leads to the nonlinear system

$$W_i(\vec{\alpha}) := \mathcal{W}[\mathbf{x}_i, \tilde{u}(\mathbf{x}_i), D\tilde{u}(\mathbf{x}_i)] = 0, \qquad 1 \le i \le N. \tag{6}$$

A root $\vec{\alpha}_*$ of (6)–i.e. $\{W_i(\vec{\alpha}_*) = 0\}_{i=1}^N$ or simply $\vec{W} = 0$–represents an RBF solution $\tilde{u}(\vec{\alpha}_*)$ of the BVP (5). Even if the nonlinear BVP (5) has one unique solution, the meshless discretization (6) may have none, one, multiple or infinitely many roots, regardless of the fact that the system is square. Therefore, it is not evident that collocation is the best approach to RBF representations of solutions to nonlinear BVPs, especially given that least-squares RBF approximations have been found to be preferable to strict collocation in other contexts [22, 25]. Interestingly enough, we have found apparently unique strict roots in every well-conditioned square RBF collocation system arising from the various PDEs in our numerical experiments, provided that the domain discretization is reasonable enough.

1.4 Rootfinding approach

We are seeking a root $\vec{\alpha}_*$ of (6). The most well-known rootfinding algorithm is Newton's method for systems [24], which proceeds as:

$$\vec{\alpha}_{k+1} = \vec{\alpha}_k - J_k^{-1}\vec{W}_k \tag{7}$$

where $\vec{W}_k = \Big(W_1(\vec{\alpha}_k), \ldots, W_N(\vec{\alpha}_k) \Big)^T$ and J_k is the Jacobian evaluated at $\vec{\alpha}_k$. Like all the solvers considered in this paper, Newton's method requires an initial guess $\vec{\alpha}_0$ (which may be the interpolation coefficients of a guess function $\tilde{u}_0$) to kick off the iterations. One advantage of Newton's method is that, if $\vec{\alpha}_0$ is close enough to a root $\vec{\alpha}_*$, if $\det J(\vec{\alpha}_*) \neq 0$, and if every Jacobian in the sequence (7) is well conditioned enough, then $\{\vec{\alpha}_k\}$ will converge to $\vec{\alpha}_*$ quadratically. However, the sequence may not be convergent at all if $\vec{\alpha}_0$ is not a good enough guess. In this paper, we analyze and advocate the trust-region algorithm (TRA) for nonlinear RBF collocation. A TRA approach was implicitly used (via Matlab's `fsolve`) to solve Navier-Stokes equations in [9], with numerical Jacobians constructed via finite differences. Finite difference Jacobians are very expensive to construct and not as accurate as analytic ones; in particular for RBF collocation, they become numerically unstable long before. In this paper we derive–for the first time, to the best of our knowledge–analytic formulas for the Jacobian and Hessian of a wide range of nonlinear operators. Not only do they substantially improve the performance of the RBF/TRA method, but also enable a root of the nonlinear system to be found where the previous approaches fail, in the first place. Moreover, they also might offer theoretical insight into the root structure of the system.

Let us briefly mention two directions that we have not pursued further. When all nonlinearities are made up of sums and products of derivatives–such as $u\nabla^2 u + (\partial u/\partial x)(\partial u/\partial y)$, for instance–RBF collocation gives rise to a system of polynomials in $\alpha_1, \ldots, \alpha_N$. In principle, the complete root structure of such a system could be revealed in the framework of Groebner bases [5], using for instance SINGULAR. (In practice, however, typical RBF systems are too large, and probably too ill-conditioned, to be tackled this way.) Another interesting method of tackling polynomial systems is homotopy/continuation.

1.5 Operator-Newton (linearization) approach

An alternative way of solving nonlinear elliptic BVPs with RBFs is the operator-Newton method introduced by Fasshauer [11]. The idea is to recast the original nonlinear BVP into a sequence of linear BVPs yielding ever smaller contributions. Those linear BVPs can then be solved straight away with RBF collocation, i.e. working in the PDE space rather than in the RBF coefficient space. This approach was used for instance in [2] to solve a quasilinear PDE arising in fluid dynamics. However, we will show in Section 4 that the operator-Newton method is equivalent–at least in its most straightforward version–to Newton's method for systems and thus susceptible of erratic behaviour in the event of an inadequate starting guess.

1.6 Outline of the paper

The remainder of the paper is organized as follows. We review the TRA in Section 2, with an eye on RBF collocation. Particular attention is paid to the so-

called trust-region subproblem and three schemes for solving it are surveyed. Section 3 is the core of the paper, for it derives specific formulas for the Jacobian and Hessian of nonlinear RBF collocation. In Section 4, we show the equivalence between the operator-Newton approach and Newton's method for nonlinear systems of equations. Section 5 derives formulas for three important classes of nonlinear elliptic BVPs in detail, illustrating how to apply the RBF/TRA ansatz to general equations. Some comments on solvability and uniqueness of the nonlinear collocation system are made in Section 6. Section 7 reports extensive numerical experiments on four different BVPs, and Section 8 discusses them. Finally, Section 9 concludes the paper.

1.7 Notation

- In the space $\mathbb{R}^d$ where the BVP $(\mathcal{W}^{PDE}, \mathcal{W}^{BC})$ is defined, vectors are written in bold (like $\mathbf{x}$ or $\mathbf{N}$), and operators in italics (like $\mathcal{W}$).
- $\mathcal{L}\phi_{ij} = (\mathcal{L}\phi)(\|\mathbf{x}_i - \mathbf{x}_j\|)$–like $\nabla^2\phi_{ij}$–are the entries of matrix $[\mathcal{L}\phi]$.
- *Nodal vectors* (in $\mathbb{R}^N$) are associated to the RBF centres, such as $\vec{\alpha}$, or to a function $f(\mathbf{x})$ evaluated over the pointset, $\vec{f} = \Big(f(\mathbf{x}_1), \ldots, f(\mathbf{x}_N)\Big)$, or to an operator $\mathcal{W}$ collocated on the pointset nodes, $\vec{W}$.
- Matrices acting in the RBF coefficient space are denoted with capital letters (J, H) or like $[\phi]$, $[\mathcal{L}\phi]$, if they are the collocation matrix of ϕ, $\mathcal{L}\phi$, etc. Finally, $diag[f]$ stands for a diagonal matrix with diagonal $\vec{f}$.
- With the ordering $\Xi_N = \Big(\{\mathbf{x}_1, \ldots, \mathbf{x}_M\} \in \Omega\Big) \cup \Big(\{\mathbf{x}_{M+1}, \ldots, \mathbf{x}_N\} \in \partial\Omega\Big)$, $[\mathcal{L}\phi]_\Omega \in \mathbb{R}^{M\times N}$ is the upper block of the matrix $[\mathcal{L}\phi]$ in (4) and $[\mathcal{L}\phi]_{\partial\Omega} \in \mathbb{R}^{(N-M)\times N}$ the lower block.
- $A > 0$ ($A < 0$) stands for a positive (negative) definite square matrix A.

2 Overview of the trust-region algorithm

In this section, we discuss the TRA mainly following [24, chapters 4 and 11], focussing on those aspects which best meet the features of RBF collocation, namely: non-sparse matrices, bad conditioning, and middle-size discretizations ($N \lesssim 3000$). Very special attention has been paid to the possibility of using the exact Hessian, which will be derived in Section 3. First, a sum-of-squares scalar merit function $\mu(\vec{\alpha})$ is chosen:

$$\mu(\vec{\alpha}) = \vec{W}^T\vec{W}/2 = \frac{1}{2}\sum_{i=1}^{N} W_i^2(\vec{\alpha}) \geq 0. \tag{8}$$

The merit function $\mu(\vec{\alpha})$ inherits the smoothness of the RBF ϕ. Rootfinding is then recast as minimization of μ:

$$\vec{\alpha}_* = \arg\min_{\vec{\alpha}\in\mathbb{R}^N} \mu(\vec{\alpha}). \tag{9}$$

A zero of μ is a root of the system $\vec{W} = 0$, and vice versa. Moreover, a zero of μ is an absolute minimum of μ. The gradient and Hessian of $\mu(\vec{\alpha})$ are

$$\nabla\mu = J^T\vec{W}, \qquad H = \nabla^2\mu = J^TJ + \sum_{i=1}^{N} W_i\nabla^2 W_i, \tag{10}$$

where J is the Jacobian of $\vec{W}$ (22). Therefore, starting from an initial guess $\vec{\alpha}_0$, we seek a descending sequence $\mu(\vec{\alpha}_0) > 0, \mu(\vec{\alpha}_1), \mu(\vec{\alpha}_2), ..., \mu(\vec{\alpha}_\infty)$ hopefully leading to the absolute minimum of μ. Unfortunately, state-of-the-art minimization algorithms (not only the TRA) cannot rule out the possibility of getting trapped in a local minimum (one where $\mu > 0$ and thus not a root), even if a zero of μ does exist. (The exception, as mentioned in the Introduction, are polynomial systems tackled with Groebner bases or homotopy/continuation, which pose other kind of difficulties anyway.) Moreover, minimization algorithms based on derivative information (such as the TRA) can only find *stationary points* (where $\nabla\mu(\vec{\alpha}_\infty) = 0$), rather than minima.

The advantage of the TRA over other minimization algorithms is that it can deliver *global convergence*, which is assured convergence to *some* minimum of μ from *any* initial guess $\vec{\alpha}_0$. Precise conditions which guarantee this will be discussed later. In order to generate the iterates $\vec{\alpha}_k$, the TRA proceeds as $\vec{\alpha}_{k+1} = \vec{\alpha}_k + \vec{\gamma}_k$, taking at every iteration a step $\vec{\gamma}_k$ such that

$$\vec{\gamma}_k \approx \vec{\gamma}_{k*} := \arg\min_{\|\vec{\gamma}\|\le\Delta_k} \left\{\theta_k(\vec{\gamma}) := \mu_k + \nabla\mu_k^T\vec{\gamma} + \frac{1}{2}\vec{\gamma}^T A_k\vec{\gamma}\right\}. \tag{11}$$

In (11), A_k is a symmetric approximation to the Hessian of $\mu_k := \mu(\vec{\alpha}_k)$, and $\Delta_k > 0$ is the *trust-region radius*. θ_k is a second-order approximation (or *model*) of μ which is only *trusted*–and this is the hallmark of the TRA–within a distance Δ_k from the current iterate $\vec{\alpha}_k$. Problem (11)–namely, the minimization of a quadratic polynomial inside a sphere–is referred to as the *trust region subproblem* (TRS). Once the TRS is solved (exactly or approximately), the fidelity of the model can be assessed *a posteriori* by

$$\rho_k = \frac{\mu(\vec{\alpha}_k) - \mu(\vec{\alpha}_k + \vec{\gamma}_k)}{\theta_k(0) - \theta_k(\vec{\gamma}_k)}. \tag{12}$$

The trust-region radius can be then dynamically adjusted according to Algorithm 1. Note that a decreasing step may also be rejected if the model is deemed poor ($\rho_k \le \eta$ for some threshold $0 < \eta < 1$).

Algorithm 1 Trust Region Algorithm (see [24, algorithms 4.1 and 11.5]).

Data: $\vec{\alpha}_0, \Delta_{max} > 0, \Delta_0 \in (0, \Delta_{max})$, and $\eta \in [0, 1/4)$
Result: Convergence to a stationary point/minimum of μ (see Theorem 1)
for $k = 1, 2, \ldots$ until convergence **do**
  obtain $\vec{\gamma}_k$ by (approximately) solving (11)
  evaluate ρ_k according to (12)
  if $\rho_k < 1/4$ **then**
    $\Delta_{k+1} = \Delta_k/4$
  else
    if $\rho_k > 3/4$ and $\| \vec{\gamma}_k \| = \Delta_k$ **then**
      $\Delta_{k+1} = \min\{2\Delta_k, \Delta_{max}\}$
    else
      $\Delta_{k+1} = \Delta_k$
    end if
  end if
  if $\rho_k > \eta$ **then**
    $\vec{\alpha}_{k+1} = \vec{\alpha}_k + \vec{\gamma}_k$
  else
    $\vec{\alpha}_{k+1} = \vec{\alpha}_k$
  end if
end for

There are two more important definitions. The *Cauchy step*, $\vec{\gamma}_C$, is the minimizer of θ_k inside the trust region along the steepest descent direction–so that $\theta_k(\vec{\gamma}_C) \geq \theta_k(\vec{\gamma}_*)$. It turns out to be [24, section 4.1]:

$$\vec{\gamma}_C = -\tau_k \frac{\Delta_k}{\| \nabla \mu_k \|} \nabla \mu_k, \text{ with } \tau_k = \begin{cases} 1, & \text{if } \nabla \mu_k^T A_k \nabla \mu_k \leq 0, \\ \min\{ \| \nabla_k \mu \|^3 / (\Delta_k \nabla \mu_k^T A_k \nabla \mu_k), 1 \}, & \text{otherwise.} \end{cases} \tag{13}$$

Note that $\vec{\gamma}_C$ involves no linear systems and yet provides some drop in μ–hence it is useful to fall back on in the event of severe ill-conditioning of A_k. The *full step*, $\vec{\gamma}_F$, is the unconstrained minimizer of a quadratic polynomial:

$$\vec{\gamma}_F = \arg\min_{\vec{\gamma} \in \mathbb{R}^N} \theta_k(\vec{\gamma}) = -A_k^{-1} \nabla \mu_k. \tag{14}$$

The convergence properties of the TRA depend on the method employed to tackle the TRS, which we address next. Let us drop iteration subindex k while discussing the TRS. The critical insight is that the TRS need not be solved exactly for the TRA to converge. In fact, a TRS approximation $\vec{\gamma} \approx \vec{\gamma}_*$ resulting in a drop in μ which is at least a fixed positive fraction of the drop achieved by the Cauchy step suffices for convergence [24, theorems 4.8 and 4.9]. If $A > 0$, $\vec{\gamma}_F$ is calculated. If, moreover, $\vec{\gamma}_F$ lies inside the trust region, then $\vec{\gamma}_* = \vec{\gamma}_F$. Otherwise, either $A > 0$ but the full step is not feasible (i.e. $\|\vec{\gamma}_F\| > \Delta$), so that $\vec{\gamma}_*$ must lie on the boundary of the trust region; or A is indefinite–and the full step $\vec{\gamma}_F$ may not be a minimum of μ in the first place. In both cases, the TRS–a nonlinear

problem itself–must be solved iteratively. Bearing in mind the application to RBF collocation, we will consider three TRS approximations:

1. **Nearly exact (or "full")**. The exact solution $\vec{\gamma}_*$ of the TRS (11) can be found following an approach due to Moré and Sorensen [23] (see also [24, chapter 4]). The higher computational cost of this TRS method is only warranted if the full Hessian is available, so we will assume that $A = H$. A canned implementation is Matlab's `trust`–albeit it is not given as an option in Matlab's `fsolve`–where the full eigendecomposition of H is carried out. An important case is when $\nabla\mu^T\vec{q}_1 = 0$ ($\vec{q}_1$ is the eigenvector of λ_1, the smallest eigenvalue of $A = H$), called by Moré and Sorensen the *hard case*, which may appear due to ill-conditioning of the RBF matrices. Alternatively, the method described in detail in [24, section 4.2]) involves several factorizations of H rather than the full eigendecomposition. Due to space limitations, the reader is referred to those papers for further details. Despite its significantly higher computational cost, this nearly exact solution will serve as a benchmark to assess the performance of the remaining two approximations.
2. The **dogleg method**, involving just one matrix factorization ($A = J^TJ$).
3. **2D subspace minimization**, involving two factorizations of $A = H$.

Conjugate-gradient-based methods for the TRS have not been included because they are mostly meant for large and sparse matrices and are especially sensitive to bad conditioning, while–to the best of our knowledge–there are not really efficient general preconditioners available for our problem.

2.1 The dogleg method

When the full step $\vec{\gamma}_F$ is not feasible, the minimum in the TRS is approximated by the intersection of the trust region with the "dogleg path" [24, figure 4.3]

$$\vec{p}(\tau) = \begin{cases} \tau\vec{\gamma}_u, & \text{if } 0 \le \tau \le 1, \\ \vec{\gamma}_u + (\tau - 1)(\vec{\gamma}_F - \vec{\gamma}_u), & \text{if } 1 \le \tau \le 2, \end{cases} \tag{15}$$

where

$$\vec{\gamma}_u = -\frac{\nabla\mu^T\nabla\mu}{\nabla\mu^TA\nabla\mu}\nabla\mu. \tag{16}$$

It can be proved that $\tau \in [0, 2]$ is the solution of the scalar equation

$$||\vec{\gamma}_u + (\tau - 1)(\vec{\gamma}_F - \vec{\gamma}_u)||^2 = \Delta^2. \tag{17}$$

See [24, sections 4.1 and 11.2] for further details. Therefore, minimization in N dimensions is replaced by minimization along the dogleg path. The standard choice (in fact the default in many solvers such as Matlab's *fsolve*, and in the remainder of this paper) is $A = J^TJ \ge 0$, which incorporates partial Hessian information without constructing H in the first place, and provides second-order convergence with just first-derivative information [24, section 11.2].

2.2 2D subspace minimization of the TRS

Byrd, Schnabel and Schultz extended the minimization to the whole bidimensional subspace containing the dogleg path in such a way that also indefinite A can be used [6, 26]. With this method, we will also understand that $A = H$. Practical experience shows that often, the drop in μ in the bidimensional subspace compares to that in $\mathbb{R}^N$, but at a fraction of the cost. We put together the pseudocode in Algorithm 2.

Algorithm 2 Two-dimensional subspace approximation of the TRS (2Dsub)

Data: $\nabla\mu, H, \Delta > 0, tol > 0$
Require: $\nu \approx \lambda_1 := \min eig(H)$, such that $|\nu| \in (-\lambda_1, -2\lambda_1]$ if $\lambda_1 < 0$
Result: approximate minimizer $\vec{\gamma} \approx \vec{\gamma}_*$ of the TRS in (11)

1) Check definiteness of H
if $\nu > tol$ **then**
  $H > 0$. Compute $\vec{\gamma}_F$ in (14)
  if $\|\vec{\gamma}_F\| \leq \Delta$ **then**
    $\vec{\gamma} = \vec{\gamma}_F = \vec{\gamma}_*$ and **return**
  else
    let $S_2 = span[\nabla\mu, \vec{\gamma}_F]$ and **goto 2**)
  end if
else if $|\nu| < tol$ **then**
  H is numerically singular. $\vec{\gamma} = \vec{\gamma}_C$ and **return**
else
  H is indefinite. Compute $\vec{p} = -(H - \nu I)^{-1}\nabla\mu$
  if $\|\vec{p}\| \leq \Delta$ **then**
    let $\vec{q}$ be a descent direction of μ and $\|\vec{q}\| = 1$
    let $\vec{v} = -v\vec{q}$, where $v = -\vec{p}^T\vec{q} + \sqrt{(\vec{p}^T\vec{q})^2 + \Delta^2 - \|\vec{p}\|^2}$
    let $\vec{\gamma} = \vec{p} + \vec{v}$ and **return**
  else
    let $S_2 = span[\nabla\mu, \vec{p}]$ and **goto 2**)
  end if
end if

2) Let $S_2 = span[\vec{s}_1, \vec{s}_2]$, with $\|\vec{s}_1\| = \|\vec{s}_2\| = 1$, and $P_2 = [\vec{s}_1, \vec{s}_2]$ (a projector)
find the minimizer $\vec{\xi}_* = \sigma_1\vec{s}_1 + \sigma_2\vec{s}_2 = (\sigma_1, \sigma_2)^T$ of the model θ in subspace S_2:
$\vec{\xi}^* = \arg \min_{\sqrt{\sigma_1^2+\sigma_2^2} \leq \Delta} \mu + \nabla\mu^T P_2(\sigma_1, \sigma_2)^T + \frac{1}{2}(\sigma_1, \sigma_2)P_2^T A P_2(\sigma_1, \sigma_2)^T$
$\vec{\gamma} = \vec{\xi}_*$ and **return**.

Once ν is available, the descent direction $\vec{q}$ can be the eigenvector, i.e. $H\vec{q} \approx \nu\vec{q}$. In [6], ν is meant to be efficiently estimated with the Lanczos method.

2.3 Convergence properties of the TRA

The following result is a corollary from theorems 4.8 and 4.9 in [24], which in turn were proved in [26] and [23], respectively. For the dogleg method, it is possible to derive conditions on J rather than on μ [24, theorems 11.8 and 11.9].

Theorem 1 (Convergence of the TRA with the three TRS methods in this paper) *Assume that the TRS is solved either by the nearly exact method ($A = H$), or the dogleg method ($A = J^T J$), or by 2D subspace minimization ($A = P_2^T H P_2$). Furter assume that $\|A\|$ is bounded above and that μ is Lipschitz continuously differentiable and bounded below on the level set $\{\vec{\alpha} \,|\, \mu(\vec{\alpha}) \leq \mu(\vec{\alpha}_0)\}$. Then, the TRA (Algorithm 1) with constant η converges to a stationary point–which may be either an absolute minimum, a local minimum, or a saddle point. If, additionally, that level set is compact, the TRA with nearly-exact TRS solution either converges to a minimum (local or absolute), or $\vec{\alpha}_k$ has a limit point in the level set at which second-order necessary minimality conditions hold (but not a saddle point).*

The convergence of the TRA close to a non-degenerate root (i.e. where $\det J(\vec{\alpha}_*) \neq 0$) is quadratic if J is Lipschitz-continuous around $\vec{\alpha}_*$ and the TRS is solved exactly [24, theorem 11.10]. Note however that, as the iterates $\vec{\alpha}_k$ approach the root more closely, eventually the root will lie within the trust region, and then $\vec{\gamma} = \vec{\gamma}_F$ is nearly exactly the Newton step in (7), since $J^T J \to H$ because $\{W_i \approx 0\}_{i=1}^N$ in (10). Since Newton's method converges quadratically to a non-degenerate root, so do all three TRS methods considered in this paper.

2.4 Scaling

The TRA with spherical trust regions may perform poorly when the merit function is posed with poor scaling–i.e. changes much faster along some directions than other. Linearly rescaling the coefficients vector,

$$\vec{\alpha}' := \Gamma\vec{\alpha}, \qquad (\det\Gamma \neq 0) \tag{18}$$

may make up for it. Then,

$$J_{ij} = \frac{\partial W_i}{\partial \alpha_j} = \sum_{k=1}^{N} \frac{\partial W_i}{\partial \alpha'_k}\frac{\partial \alpha'_k}{\partial \alpha_j} = \sum_{k=1}^{N} J'_{ik}\Gamma_{kj}, \tag{19}$$

so that

$$J' = J\Gamma^{-1}, \qquad A' = \Gamma^{-T}A\Gamma^{-1}. \tag{20}$$

Notice that this may change the conditioning of J' and A'. The simplest choice is to take Γ diagonal with positive elements–which is equivalent to keeping the original variables and replacing the spherical trust region by an elliptical one defined by $\|\Gamma\gamma_k\| \leq \Delta_k$ [24, section 4.4]. The diagonal entries Γ_{ii} must be a reflection of the sensitivity of the merit function to changes along the i^{th} coordinate axis. A reliable option is setting

$$\Gamma_{ii} = \frac{\partial^2 \mu}{\partial \alpha_i^2}. \tag{21}$$

3 The trust-region algorithm for RBF collocation

In this section, we address specific aspects of the TRA when applied to RBF collocation, including analytical formulas for J and H. Henceforth, the combined method will be referred to as RBFTrust.

3.1 RBF Jacobian and Hessian

The Jacobian of (6) is:

$$J(\vec{\alpha}) = \begin{bmatrix} \frac{\partial W_1}{\partial \alpha_1} & \cdots & \frac{\partial W_1}{\partial \alpha_N} \\ \vdots & \ddots & \vdots \\ \frac{\partial W_N}{\partial \alpha_1} & \cdots & \frac{\partial W_N}{\partial \alpha_N} \end{bmatrix}. \tag{22}$$

Recall that W_i refers to either $\mathcal{W}^{PDE}$ or $\mathcal{W}^{BC}$ depending on whether $\mathbf{x}_i$ is in Ω or on $\partial\Omega$, and thus $J \neq J^T$. We will assume that $\mathcal{W}$ is smooth and consists of S functions of u and its derivatives with respect to $\mathbf{x}$ (including the identity operator I). For instance, in $\mathcal{W}u = \nabla^2 u + \sqrt{u}(\partial u/\partial x)^2 - u$, there are three such components, namely $D_1 = I, D_2 = \partial u/\partial x$, and $D_3 = \nabla^2$ (the order is irrelevant). Replacing u by $\tilde{u}$ and applying the chain rule,

$$\frac{\partial W_i}{\partial \alpha_j} = \sum_{m=1}^{S} \frac{\partial W_i}{\partial D_m \tilde{u}(\mathbf{x}_i)} \frac{\partial D_m \tilde{u}(\mathbf{x}_i)}{\partial \alpha_j}. \tag{23}$$

Let us use the shorthand notation $\partial W_i/\partial D_m$ for $\partial W_i/\partial D_m \tilde{u}(\mathbf{x}_i)$. By linearity,

$$\frac{\partial W_i}{\partial \alpha_j} = \sum_{m=1}^{S} \frac{\partial W_i}{\partial D_m} \frac{\partial}{\partial \alpha_j} \sum_{k=1}^{N} \alpha_k D_m \phi_{ik} = \sum_{m=1}^{S} \frac{\partial W_i}{\partial D_m} D_m \phi_{ij}. \tag{24}$$

Then, using the notation introduced in Section 1.7:

$$J(\vec{\alpha}) = \begin{bmatrix} \sum_{m=1}^{S} \frac{\partial W_1}{\partial D_m} D_m \phi_{11} & \cdots & \sum_{m=1}^{S} \frac{\partial W_1}{\partial D_m} D_m \phi_{1N} \\ \vdots & \ddots & \vdots \\ \sum_{m=1}^{S} \frac{\partial W_N}{\partial D_m} D_m \phi_{N1} & \cdots & \sum_{m=1}^{S} \frac{\partial W_N}{\partial D_m} D_m \phi_{NN} \end{bmatrix} = \sum_{m=1}^{S} diag\Big[\frac{\partial W}{\partial D_m}\Big][D_m \phi]. \tag{25}$$

The Hessian of the merit function $\mu = \vec{W}^T \vec{W}/2$ is

$$H = \nabla^2 \mu = J^T J + \sum_{k=1}^{N} W_k \nabla^2 W_k, \tag{26}$$

where

$$\nabla^2 W_k = \begin{bmatrix} \frac{\partial^2 W_k}{\partial \alpha_1^2} & \cdots & \frac{\partial^2 W_k}{\partial \alpha_1 \partial \alpha_N} \\ \vdots & \ddots & \vdots \\ \frac{\partial^2 W_k}{\partial \alpha_N \partial \alpha_1} & \cdots & \frac{\partial^2 W_k}{\partial \alpha_N^2} \end{bmatrix}. \tag{27}$$

Notice that H, J^TJ and $W_k\nabla^2 W_k$ are symmetric. Now,

$$\frac{\partial^2 W_k}{\partial\alpha_i\alpha_j} = \frac{\partial}{\partial\alpha_i}\Big(\sum_{m=1}^{S}\frac{\partial W_k}{\partial D_m}D_m\phi_{kj}\Big) = \sum_{m=1}^{S}\sum_{n=1}^{S}\frac{\partial^2 W_k}{\partial D_m\partial D_n}D_n\phi_{ki}D_m\phi_{kj}. \tag{28}$$

Note that $D_m\phi_{ki} = \pi_m D_m\phi_{ik}$, with $\pi_m = \pm 1$, and thus $[D_m\phi]^T = \pi_m[D_m\phi]$. For instance, $\nabla^2\phi_{ij} = \nabla^2\phi_{ji}$ but $\frac{\partial\phi}{\partial x}|_{ij} = -\frac{\partial\phi}{\partial x}|_{ji}$. Then

$$\Big(\sum_{k=1}^{N} W_k\nabla^2 W_k\Big)_{ij} = \sum_{m=1}^{S}\sum_{n=1}^{S}\pi_m\sum_{k=1}^{N}D_m\phi_{ik}\Big(W_k\frac{\partial^2 W_k}{\partial D_m\partial D_n}\Big)D_n\phi_{kj}. \tag{29}$$

From (25),

$$J^TJ = \sum_{m=1}^{S}\sum_{n=1}^{S}[D_m\phi]^T diag[\frac{\partial W}{\partial D_m}\frac{\partial W}{\partial D_n}][D_n\phi]. \tag{30}$$

Summing the two parts,

$$\begin{aligned} H(\vec{\alpha}) = \sum_{m=1}^{S}\sum_{n=1}^{S}[D_m\phi]^T\Big(diag[\frac{\partial W}{\partial D_m}\frac{\partial W}{\partial D_n}] + diag[W\frac{\partial^2 W}{\partial D_m\partial D_n}]\Big)[D_n\phi] = \\ = \frac{1}{2}\sum_{m=1}^{S}\sum_{n=1}^{S}[D_m\phi]^T diag[\frac{\partial^2 W^2}{\partial D_m\partial D_n}][D_n\phi]. \end{aligned} \tag{31}$$

Note that (31) is symmetric with respect to the matrices involved.

Remark 2. The matrices $[D_1\phi], \ldots, [D_S\phi]$ are filled at start and stored. At every iteration of RBFTrust, only the diagonal matrices depend on $\vec{\alpha}_k$ and have to be recalculated. Thus the dogleg method–where only J is explicitly computed–involves only matrix-vector multiplications, while setting $A = H$ takes $S(S+1)/2$ extra matrix multiplications.

3.2 Elimination of linear equations

Often, the system $\vec{W}(\vec{\alpha}) = 0$ will contain linear equations, such as those representing the collocation of Dirichlet and other linear BCs. Another source of linear equations in the system is the enrichment of the RBF interpolant (see for instance [3, 4], and Example III in Section 7) with n special functions h_k:

$$\tilde{u} = \sum_{j=1}^{N}\alpha_j\phi(\|\,\mathbf{x} - \mathbf{x}_j\,\|_2) + \sum_{k=1}^{n}h_k(\mathbf{x}). \tag{32}$$

In this case, the system must be augmented with n ancillary equations in order to keep it square:

$$\alpha_1 h_k(\mathbf{x}_1) + \ldots + \alpha_N h_k(\mathbf{x}_N) = 0, \qquad k = 1, \ldots, n. \tag{33}$$

Whenever there are $m \leq N$ linearly independent equations, m degrees of freedom can be eliminated, and the minimum for μ sought in a shrunken $(N-m)$-dimensional space. An elimination method which is optimally stable is described in [24, section 15.2]. Assume that the linear block in (6) is given by $B\vec{\alpha} = \vec{b}$, with $B \in \mathbb{R}^{m \times N}$. Consider the QR decomposition

$$B^T \Pi = [Q_1 Q_2] \begin{bmatrix} R \\ 0 \end{bmatrix}, \tag{34}$$

where Π is an $m \times m$ permutation matrix, $Q_B = [Q_1 Q_2]$ is orthonormal, $Q_1 \in \mathbb{R}^{N \times m}$ and $Q_2 \in \mathbb{R}^{N \times (N-m)}$ are made up of orthonormal columns and $R \in \mathbb{R}^{m \times m}$ is upper triangular and nonsingular because $rank(B) = m$. Let $\vec{\alpha} = Q_1 \vec{v} + Q_2 \vec{\beta}$ and insert it into $B\vec{\alpha} = \vec{b}$, yielding the optimally stable decomposition of $\vec{\alpha}$

$$\vec{\alpha} = Q_1 R^{-T} \Pi^T \vec{b} + Q_2 \vec{\beta}. \tag{35}$$

In any elimination of degrees of freedom like $\vec{\alpha} = \vec{v} + Z\vec{\beta}$, where $\vec{v}$ is a constant vector and $\vec{\beta} \in \mathbb{R}^{N-m}$, the columns of Z represent a basis of the null space of B, since $BZ = 0$. (It is easy to prove that $BQ_2 = 0$). Whether $Z = Q_2$ or a different basis of $null(B)$, Jacobian entries are transformed as in (19) and

$$J(\vec{\beta}) = J(\vec{\alpha})Z, \qquad H(\vec{\beta}) = Z^T H(\vec{\alpha}) Z. \tag{36}$$

In (36), $J(\vec{\alpha})$ and $H(\vec{\alpha})$ are (25) and (31), respectively. (Or J' and H' like in (20) if rescaling like (18) has been included.)

4 The operator-Newton approach

Let us define the linearization of the nonlinear operator $\mathcal{W}$ around a function $u(\mathbf{x})$ as the linear operator $\mathcal{L}_u$ such that

$$\lim_{\|v\| \to 0} \frac{\mathcal{W}(u+v) - \mathcal{W}u - \mathcal{L}_u v}{\| v \|} = 0. \tag{37}$$

If that limit exists, $\mathcal{L}_u$ is unique and is called the Fréchet derivative of $\mathcal{W}$ around u [10]. Definition (37) is non-constructive. For our purposes, let us assume that $\mathcal{L}_u$ exists and that all operators are regular enough so that

$$\mathcal{W}(u+v) = \mathcal{W}(u) + \mathcal{L}_u v + O(\| v \|^2). \tag{38}$$

Let $\mathcal{L}_u^{PDE}$ and $\mathcal{L}_u^{BC}$ be the linearization of $\mathcal{W}^{PDE}$ and $\mathcal{W}^{BC}$, respectively. The operator-Newton method for nonlinear elliptic BVPs is Algorithm 3 below.

Algorithm 3 Operator-Newton method without smoothing.

Data: initial guess $u_0(\mathbf{x})$, the linearized operators $\mathcal{L}_u = \{\mathcal{L}_u^{PDE}, \mathcal{L}_u^{BC}\}$
for $k = 1, 2, ...$ until $R_k = \{R_k^{PDE}, R_k^{BC}\}$ stagnates **do**

A3.1. Compute the residuals $\begin{cases} R_k^{PDE} = -\mathcal{W}^{PDE} u_{k-1} & \text{if } \mathbf{x} \in \Omega, \\ R_k^{BC} = -\mathcal{W}^{BC} u_{k-1} & \text{if } \mathbf{x} \in \partial\Omega \end{cases}$

A3.2. Solve the BVP $\begin{cases} \mathcal{L}_{u_k}^{PDE} v_k = R_k^{PDE} & \text{if } \mathbf{x} \in \Omega, \\ \mathcal{L}_{u_k}^{BC} v_k = R_k^{BC} & \text{if } \mathbf{x} \in \partial\Omega \end{cases}$

A3.3. Update $u_k = u_{k-1} + v_k$

end for

4.1 Equivalence to Newton's method

Note that Algorithm 3 is independent of the discretization. Let us assume that every iteration of it is implemented on a pointset Ξ_N using the RBF ϕ, i.e. $\tilde{u}_k = [\phi]\vec{\alpha}_k$ and $\tilde{v}_k = [\phi]\vec{\gamma}_k$. Then, the updating step (**A3.3**) is equivalent to $\vec{\alpha}_{k+1} = \vec{\alpha}_k + \vec{\gamma}_{k+1}$. The matrix version of an iteration of Algorithm 3 reads $[\mathcal{L}_k\phi]\vec{\gamma}_k = -\vec{W}_k$, –i.e. exactly as in (4) but replacing $\vec{\alpha}$ by $\vec{\gamma}_k$, $(\mathcal{L}^{PDE}, \mathcal{L}^{BC})$ by $(\mathcal{L}_k^{PDE}, \mathcal{L}_k^{BC})$, and the right-hand side by $-\vec{W}_k$.

Inverting that system for $\vec{\gamma}_k$, the matrix form of the updating step (**A3.3**) in Algorithm 3 gives $\vec{\alpha}_{k+1} = \vec{\alpha}_k - [\mathcal{L}_k\phi]^{-1}\vec{\gamma}_{k+1}$, which is Newton's method if $[\mathcal{L}_k\phi] = J_k$. To justify that this is indeed the case, consider the limit

$$\lim_{\|v\|\to 0} \frac{\mathcal{W}(u+v) - \mathcal{W}u}{\| v \|} = \lim_{\|v\|\to 0} \frac{\mathcal{L}_u v}{\| v \|}, \tag{39}$$

according to (37) and (38). Let $\vec{\gamma}_k = (\gamma_k^{(1)}, \ldots, \gamma_k^{(N)})^T$. Substituting $\tilde{u}(\vec{\alpha}_k)$ and $\tilde{v}(\vec{\gamma}_k)$ for u and v and evaluating on $\mathbf{x}_i \in \Xi_N$,

$$\lim_{\|\vec{\gamma}_k\|\to 0} \frac{\mathcal{W}_i(\vec{\alpha}_k + \vec{\gamma}_k) - \mathcal{W}_i(\vec{\alpha}_k)}{\| \vec{\gamma}_k \|} = \lim_{\|\vec{\gamma}_k\|\to 0} \frac{\sum_{n=1}^{N} \gamma_k^{(n)} \mathcal{L}_k \phi_{in}}{\| \vec{\gamma} \|}. \tag{40}$$

Now, let us particularize to the direction $\gamma_k^{(n)} = \delta_{jn}$, where δ_{jn} is Kronecker's delta. Then, the left-hand side of (40) is $\partial W_i / \partial \alpha_j = J_{ij}$, and

$$J_{ij} = L_k \phi_{ij}. \tag{41}$$

Thus, the collocated version of Algorithm 3 is not globally convergent.

Remark 3. The original operator-Newton algorithm by Fasshauer includes an additional step for residual smoothing, and therefore our result here does

not pertain to that case. In fact, his research shows that smoothing may be critical for performance, although difficult to implement in practice [12, 2].

5 Explicit formulas for prototypical PDEs

In this section, we derive explicit formulas for the Jacobian and Hessian of three relevant classes of nonlinear elliptic differential operators, as well as for the linearized operator required by the operator-Newton method. Those formulas will be used later in Examples I-IV in Section 7.

5.1 Semilinear equations

In this case, the nonlinearity involves u but none of its derivatives. We consider only the following equation, taken from [28]:

$$\nabla^2 u - u^3 = f \text{ if } \mathbf{x} \in \Omega, \qquad u = g \text{ if } \mathbf{x} \in \partial\Omega. \tag{42}$$

The Fréchet derivatives for the operator-Newton method are

$$\mathcal{L}^{PDE} = \nabla^2 - 3u^2 I, \qquad \mathcal{L}^{BC} = I. \tag{43}$$

For RBFTrust, since the BCs are linear, Z is obtained (along with Π and Q_1) from the QR decomposition of the block with the BCs, arranged as in (34):

$$[\phi]^T_{\partial\Omega}\Pi = [Q_1 Z]\begin{bmatrix} R \\ 0 \end{bmatrix}. \tag{44}$$

Recall from (35) that, after finding the solution $\vec{\beta}_\infty$ in the shrunken space $\mathbb{R}^M$, the coefficients of the RBF interpolant are transformed according to

$$\vec{\alpha}(\vec{\beta}) = Q_1 R^{-T}\Pi^T [g(\mathbf{x}_{M+1}), \ldots, g(\mathbf{x}_N)]^T + Z\vec{\beta}. \tag{45}$$

The Jacobian and Hessian in the shrunken space are given by

$$J(\vec{\beta}) = \Big([\nabla^2\phi]_\Omega - 3diag[u^2(\vec{\beta})]_\Omega [\phi]_\Omega \Big) Z, \tag{46}$$

$$\begin{aligned} H(\vec{\beta}) = Z^T \Big(& [\nabla^2\phi]^2_\Omega + [\phi]_\Omega diag[15u^4(\vec{\beta}) - 6u(\vec{\beta})\nabla^2 u(\vec{\beta})]_\Omega [\phi]_\Omega - \\ & -\Big([\phi]_\Omega + [\nabla^2\phi]_\Omega\Big) diag[3u^2(\vec{\beta})]_\Omega \Big([\phi]_\Omega + [\nabla^2\phi]_\Omega\Big) \Big) Z. \end{aligned} \tag{47}$$

The nodal values for the diagonal matrices are picked from $\vec{u}(\vec{\beta}) = [\phi]\vec{\alpha}(\vec{\beta})$.

5.2 Quasilinear equations

Let us consider the quasilinear operator

$$\mathcal{W}^{PDE}u = \nabla \cdot \Big(G(u)\nabla u\Big) - f, \tag{48}$$

where $G(t) : [0, \infty) \to \mathbb{R}$ is smooth and t actually stands for $|\nabla u|$, i.e. $G(u) = G(|\nabla u|)$. This class includes:

$$\begin{cases} \text{the operator } G(t) = 1/\sqrt{1+t^2} \text{ (see Example } II\text{), and} \\ \text{the } p\text{-Laplacian, } G(t) = t^{p-2},\ (p \geq 2) \text{ (see Example } III\text{).} \end{cases}$$

First, we will produce a linearization of the PDE suitable for the operator-Newton using a "small increment" argument. Consider the gradient modulus:

$$|\nabla(u+v)| = \sqrt{|\nabla u|^2 + |\nabla v|^2 + 2(\nabla u \cdot \nabla v)} = |\nabla u| \sqrt{1 + 2\frac{\nabla u \cdot \nabla v}{|\nabla u|^2} + \Big(\frac{|\nabla v|}{|\nabla u|}\Big)^2}. \tag{49}$$

Under the condition $|\nabla v|/|\nabla u| << 1$, a Taylor approximation yields

$$|\nabla(u+v)| = |\nabla u|\Big(1 + \frac{\nabla u \cdot \nabla v}{|\nabla u|^2} + \frac{1}{2}\Big(\frac{|\nabla v|}{|\nabla u|}\Big)^2\Big) \approx |\nabla u| + \frac{\nabla u \cdot \nabla v}{|\nabla u|} + O(|\nabla v|/|\nabla u|)^2. \tag{50}$$

Using the definition (38), it is clear that $\mathcal{L}^{(|\nabla|)}u = \frac{\nabla u \cdot \nabla}{|\nabla u|}$. By the chain rule,

$$\mathcal{L}^{(G)}u = G'(u)\frac{\nabla u \cdot \nabla}{|\nabla u|} \tag{51}$$

and therefore, to first order in $|\nabla v|/|\nabla u|$,

$$G(u+v) \approx Gu + G'(u)\frac{\nabla u \cdot \nabla v}{|\nabla u|}. \tag{52}$$

with G, G' and G'' evaluated at $|\nabla u|$. Coming back to (48),

$$\begin{aligned} \nabla \cdot [G(u+v)\nabla(u+v)] = \nabla \cdot [G\nabla u] + \nabla \cdot [G\nabla v] + \\ + \nabla \cdot [G'\frac{\nabla u \cdot \nabla v}{|\nabla u|}\nabla u] + \nabla \cdot [G'\frac{\nabla u \cdot \nabla v}{|\nabla u|}\nabla v] + O(|\nabla v|/|\nabla u|^2). \end{aligned} \tag{53}$$

The first three terms on the rhs can be simplified to:

$$\nabla \cdot [G\nabla u] = G\nabla^2 u + G'\frac{\Delta_\infty u}{|\nabla u|} \tag{54}$$

$$\nabla \cdot [G\nabla v] = G\nabla^2 v + G'\frac{\nabla v \cdot (\nabla u \cdot \nabla)\nabla u}{|\nabla u|} \tag{55}$$

$$\begin{aligned} \nabla \cdot [G'\frac{\nabla u \cdot \nabla v}{|\nabla u|}\nabla u] = G'\frac{\nabla u \cdot (\nabla u \cdot \nabla)\nabla v}{|\nabla u|} + \\ + \Big(G''\frac{\Delta_\infty u}{|\nabla u|^2} - G'\frac{\Delta_\infty u}{|\nabla u|^3} + G'\frac{\nabla^2 u}{|\nabla u|}\Big)\nabla u \cdot \nabla v + G'\frac{\nabla v \cdot (\nabla u \cdot \nabla)\nabla u}{|\nabla u|}, \end{aligned} \tag{56}$$

where we have introduced the infinity Laplacian in $\mathbb{R}^d$, $\Delta_\infty = \sum_{i,j=1}^d \frac{\partial^2}{\partial x_i \partial x_j} \frac{\partial}{\partial x_i} \frac{\partial}{\partial x_j}$. Regarding the rightmost term of (53), which is nonlinear in v, let

$$\mathbf{V} := G' \frac{\nabla u \cdot \nabla v}{|\nabla u|} \nabla v. \tag{57}$$

In order that $|\mathbf{V}| \leq |G'||\nabla v|^2 \leq \left(\frac{|\nabla v|}{|\nabla u|}\right)^2 \approx 0$, it is sufficient that

$$|G'| \leq \frac{1}{|\nabla u|^2}, \qquad \text{or} \qquad |G'(t)| \leq 1/t^2. \tag{58}$$

Thus, if (58) holds, the surviving nonlinear term in (53) can be dropped. In the operator-Newton method, the correction v around u obeys the linear PDE:

$$G\nabla^2 v + G' \frac{\nabla u}{|\nabla u|} \cdot (\nabla u \cdot \nabla)\nabla v + \frac{G''|\nabla u|\Delta_\infty u + G'|\nabla u|^2 \nabla^2 u - G'\Delta_\infty u}{|\nabla u|^3} \nabla u \cdot \nabla v + \\ +2\frac{G'}{|\nabla u|} \nabla v \cdot (\nabla u \cdot \nabla)\nabla u = R^{PDE} \text{ if } \mathbf{x} \in \Omega, \tag{59}$$

where $R^{PDE} = f - div[G(u)\nabla u]$ is the residual to (48). In $d = 2$, (59) is

$$A\frac{\partial^2 v}{\partial x^2} + B\frac{\partial^2 v}{\partial x \partial y} + C\frac{\partial^2 v}{\partial y^2} + D\frac{\partial v}{\partial x} + E\frac{\partial v}{\partial y} = |\nabla u|^3 R^{PDE} \tag{60}$$

where:

$$A(x,y) = G|\nabla u|^3 + G'|\nabla u|^2 u_x^2, \tag{61}$$

$$B(x,y) = 2G'|\nabla u|^2 u_x u_y, \tag{62}$$

$$C(x,y) = G|\nabla u|^3 + G'|\nabla u|^2 u_y^2, \tag{63}$$

$$D(x,y) = \Big(G''|\nabla u|\Delta_\infty u + G'|\nabla u|^2(3u_{xx} + u_{yy}) - G'\Delta_\infty u\Big)u_x + 2G'|\nabla u|^2 u_{xy} u_y, \tag{64}$$

$$E(x,y) = \Big(G''|\nabla u|\Delta_\infty u + G'|\nabla u|^2(3u_{yy} + u_{xx}) - G'\Delta_\infty u\Big)u_y + 2G'|\nabla u|^2 u_{xy} u_x. \tag{65}$$

The linearized PDE is elliptic as long as $B^2 - 4AC < 0$, i.e. $1 + t\frac{G'}{G} > 0$ or

$$G'/G > -1/t \tag{66}$$

Let us check the linearization (58) and ellipticity (66) conditions for the PDEs (48). For the least area operator, $G'(t) = -tG^3(t)$ and $0 \leq G \leq 1$, so that both (66) and (58) hold for any $|\nabla u|$. Regarding the p-Laplace operator, (66) holds,

but (58) leads to $(p-2)t^{p-1} \leq 1$. For instance, for $p = 2.6$ as in Example III, the linearization breaks down if $|\nabla u| > (p-2)^{1/(1-p)} = 1.37$.

Remark 4. The derivation of the linearized operator in terms of a "small increment" argument, like above, has the advantage that it sheds light on why the operator-Newton method breaks down in the presence of high gradients of the solution: it neglects contributions which are too large to be dropped, namely the nonlinear term in v on the right-hand side of (53).

We write down now the formulas for J and H needed in RBFTrust. Expanding the divergence before linearization, (48) reads:

$$G\nabla^2 u + \frac{G'\Delta_\infty u}{|\nabla u|} = f \tag{67}$$

For Plateau's problem (Example II) $f = 0$, $d = 2$ and, since $1 + |\nabla u|^2 \neq 0$,

$$W^{PDE} = (1 + |\nabla u|^2)\nabla^2 u - \Delta_\infty u, \qquad W^{BC} = u \tag{68}$$

so that the non-zero entries of the RBF Jacobian (25) can be filled up with:

$$\frac{\partial W^{PDE}}{\partial u_{xx}} = 1 + u_y^2, \quad \frac{\partial W^{PDE}}{\partial u_{xy}} = -2u_x u_y, \quad \frac{\partial W^{PDE}}{\partial u_{yy}} = 1 + u_x^2,$$
$$\frac{\partial W^{PDE}}{\partial u_x} = 2u_x u_{yy} - 2u_y u_{xy}, \qquad \frac{\partial W^{PDE}}{\partial u_y} = 2u_y u_{xx} - 2u_x u_{xy}, \qquad \frac{\partial W^{BC}}{\partial u} = 1. \tag{69}$$

The non-zero second derivatives needed for the RBF Hessian are:

$$\frac{\partial^2 W^{PDE}}{\partial u_x \partial u_{xy}} = -2u_y, \qquad \frac{\partial^2 W^{PDE}}{\partial u_y \partial u_{xy}} = -2u_x, \qquad \frac{\partial^2 W^{PDE}}{\partial u_x \partial u_{yy}} = 2u_x, \qquad \frac{\partial^2 W^{PDE}}{\partial u_y \partial u_{xx}} = 2u_y,$$
$$\frac{\partial^2 W^{PDE}}{\partial u_x \partial u_y} = -2u_{xy}, \qquad \frac{\partial^2 W^{PDE}}{\partial u_x^2} = 2u_{yy}, \qquad \frac{\partial^2 W^{PDE}}{\partial u_y^2} = 2u_{xx}. \tag{70}$$

For the Hele-Shaw equation, $f = 0$ also, but the BCs either of Neumann or Dirichlet kind, depending on the point on the boundary (see Example III):

$$\mathcal{W}^{PDE} = |\nabla u|^{p-2}\Big(\nabla^2 u + (p-2)|\nabla u|^{p-2}\Delta_\infty u\Big), \qquad \mathcal{W}^{BC} = \begin{cases} \mathcal{W}^N = \mathbf{N}\cdot\nabla u \\ \mathcal{W}^D = u - g \end{cases} \tag{71}$$

In $d = 2$, $\mathbf{N} = (N_x, N_y)$. The first and second derivatives of $\mathcal{W}^{PDE}$ with respect to $u_{xx}, u_{xy}, u_{yy}, u_x$ and u_y are found as before. (Since they are rather lengthy we do not write them down.) Since the BCs are linear, only the first derivatives of $\mathcal{W}^{BC}$ are nonzero, namely

$$\frac{\partial W^D}{\partial u} = 1, \qquad \frac{\partial W^N}{\partial u_x} = N_x, \qquad \frac{\partial W^N}{\partial u_y} = N_y. \tag{72}$$

When constructing the Jacobian and Hessian, notice that $[\phi_x]$ and $[\phi_y]$ are antisymmetric. The linear block entering the QR decomposition (34) includes not only the BCs (both Neumann and Dirichlet), but also the complementary equations (33) for the Motz functions (80) if they are incorporated into the RBF interpolant, as in Example III.

5.3 Fully nonlinear operator

Here, the nonlinearity involves the highest-order derivatives. In this class, we consider the Monge-Ampère operator in $\mathbb{R}^d$,

$$\mathcal{W}^{PDE}u = \det D_d^2 u, \tag{73}$$

where $D_d^2 u$ is the Hessian matrix of $u : \mathbb{R}^d \to \mathbb{R}$. In $d = 2$ and $d = 3$,

$$D_2^2 u = u_{xx}u_{yy} - u_{xy}^2, \qquad D_3^2 u = \det \begin{bmatrix} \frac{\partial^2 u}{\partial x^2} & \frac{\partial^2 u}{\partial x \partial y} & \frac{\partial^2 u}{\partial x \partial z} \\ \frac{\partial^2 u}{\partial y \partial x} & \frac{\partial^2 u}{\partial y^2} & \frac{\partial^2 u}{\partial y \partial z} \\ \frac{\partial^2 u}{\partial z \partial x} & \frac{\partial^2 u}{\partial z \partial y} & \frac{\partial^2 u}{\partial z^2} \end{bmatrix} \tag{74}$$

This PDE gives rise to a polynomial system of collocation equations and is simpler to linearize. We assume Dirichlet BCs and write down the formulas for $d = 2$. For the operator-Newton method,

$$\mathcal{L}^{PDE} = u_{yy}\frac{\partial^2}{\partial x^2} + u_{xx}\frac{\partial^2}{\partial y^2} - 2u_{xy}\frac{\partial^2}{\partial x \partial y}. \tag{75}$$

For RBFTrust, Z, Q_1 and Π are given by (44), and the Jacobian and Hessian are:

$$J(\vec{\beta}) = \Big(diag[u_{yy}(\vec{\beta})]_\Omega[\phi_{xx}]_\Omega + diag[u_{xx}(\vec{\beta})]_\Omega[\phi_{xx}]_\Omega - 2diag[u_{xy}(\vec{\beta})]_\Omega[\phi_{xy}]_\Omega \Big) Z \tag{76}$$

$$\begin{aligned} H(\vec{\beta}) = \frac{1}{2}Z^T \Big(& [\phi_{xx}]_\Omega diag[2u_{yy}^2(\vec{\beta})]_\Omega[\phi_{xx}]_\Omega + [\phi_{yy}]_\Omega diag[2u_{xx}^2(\vec{\beta})]_\Omega[\phi_{yy}]_\Omega + \\ & [\phi_{xy}]_\Omega diag[12u_{xy}^2(\vec{\beta}) - 4u_{xx}(\vec{\beta})u_{yy}(\vec{\beta})]_\Omega[\phi_{xy}]_\Omega + \\ & \Big([\phi_{xx}]_\Omega + [\phi_{yy}]_\Omega\Big) diag[4u_{xx}(\vec{\beta})u_{yy}(\vec{\beta}) - 2u_{xy}^2(\vec{\beta})]_\Omega \Big([\phi_{xx}]_\Omega + [\phi_{yy}]_\Omega\Big) + \\ & \Big([\phi_{xx}]_\Omega + [\phi_{xy}]_\Omega\Big) diag[-4u_{yy}(\vec{\beta})u_{xy}(\vec{\beta})]_\Omega \Big([\phi_{xx}]_\Omega + [\phi_{xy}]_\Omega\Big) + \\ & \Big([\phi_{yy}]_\Omega + [\phi_{xy}]_\Omega\Big) diag[-4u_{xx}(\vec{\beta})u_{xy(\vec{\beta})}]_\Omega \Big([\phi_{yy}]_\Omega + [\phi_{xy}]_\Omega\Big) \Big) Z. \end{aligned} \tag{77}$$

6 Remarks on solvability and uniqueness

The analytical formulas for the Jacobian and the Hessian offer insight into the structure of the nonlinear RBF system $\vec{W}(\vec{\alpha}) = 0$. Since μ inherits the smoothness of $\phi(r)$, the possible minima of μ are all critical points.

Local minima of the merit function (i.e. those for which $\mu > 0$) have the important property that $\det J = 0$, because $\nabla\mu = J^T\vec{W} = 0$ but $\vec{W} \neq 0$. (Obviously, the same property holds for any critical point of μ.) Therefore, nonsingularity of $J(\vec{\alpha})$ allows one to guarantee that RBFTrust will converge to a root–unless it can drift towards $\|\vec{\alpha}\| \to \infty$ seeking to reduce μ. This is the idea of Theorem 3, for which Theorem 2 below will be needed (see [17] and references therein).

Theorem 2 (Courant's finite-dimensional mountain pass theorem) *Suppose that $\mu(\vec{\alpha})$ is*

- *continuous with continuous derivatives in $\mathbb{R}^N$,*
- *coercive (i.e. $\lim_{\|\vec{\alpha}\|\to\infty} \mu(\vec{\alpha}) = \infty$), and*
- *possesses two different strict (i.e. isolated) minima $\vec{\alpha}_1$ and $\vec{\alpha}_2$.*

Then, μ possesses a third critical point $\vec{\alpha}_3$ such that $\mu(\vec{\alpha}_1) < \mu(\vec{\alpha}_3) > \mu(\vec{\alpha}_2)$–therefore distinct from $\vec{\alpha}_1$ and $\vec{\alpha}_2$.

Theorem 3 (Sufficient conditions for solvability and uniqueness of nonlinear RBF collocation) *Let $\mu(\vec{\alpha}) = \vec{W}^T(\vec{\alpha})\vec{W}(\vec{\alpha})/2$ with $\vec{W} : \mathbb{R}^N \mapsto \mathbb{R}$, and let $J(\vec{\alpha})$ be the Jacobian of $\vec{W}$. Assume that μ is coercive and that $\det J \neq 0$ in $\mathbb{R}^N$. Then,*

- *There is one unique root $\vec{\alpha}_*$ to $\vec{W}(\vec{\alpha}) = 0$.*
- *Under the further conditions in Theorem 1, and assuming numerical precision high enough to overcome possible ill-conditioning, RBFTrust with the full TRS method will find it from any initial guess.*

Proof. If the function is coercive, there is a minimizer of μ and since μ is twice differentiable and J is nonsingular, that minimizer is a root. For uniqueness, assume that there were two roots. Then, Theorem 2 ensures the existence of a third critical point with $\mu > 0$ and therefore singular J, which is a contradiction. For the second part, simply apply Theorem 1. □

Theorem 3 establishes a parallelism between the linear and nonlinear versions of Kansa's method, with J playing the same role as the collocation matrix $K = [\mathcal{L}\phi]$ from (4). Unfortunately, the assumptions of Theorem 3 are quite elusive–even assuming numerical stability. For instance, consider a linear BVP like (3), which can be regarded as a particular nonlinear one, and let us analyze its root structure from the point of view of $\mu(\vec{\alpha})$. The Jacobian of (4) is K itself, and $H = K^TK$. If $\det K \neq 0$, $H - \left(\lambda_{min}(K)\right)^2 I > 0$, so that $\mu(\vec{\alpha})$ is strongly convex, and thus has a unique minimum, which must be a root. Nonetheless, this could not be proved by Theorem 3 since $\lim_{\|\vec{\alpha}\|\to\infty} \mu(\vec{\alpha}) = 0$ along the direction $\vec{\alpha} = K^{-1}[f(\mathbf{x}_1), \ldots, g(\mathbf{x}_N)]^T$ –the rhs vector in (4)–so that μ is not coercive in the first place. Let us now come back to nonlinear problems. Even in the

simple and well-behaved Example I in Section 7, it does not look trivial to prove (or disprove) coercivity and nonsingularity of J from (46). Summing up, while combining the formulas for J and H with Theorem 3 looks theoretically exploitable, we have been unable to do so.

7 Numerical experiments

In this section we test RBFTrust on four nonlinear elliptic BVPs with increasing level of difficulty, taken from the literature. For comparison purposes, the operator-Newton method and a Matlab canned implementation are also sometimes used. In sum, each of the problems I-IV is solved with one or more of the following methods (see also Table 2):

- The operator-Newton method without smoothing (Section 4). We just report whether convergence was or not achieved, mostly with the goal of illustrating the shortcomings of this approach.
- Matlab's `fsolve` using the TRA with dogleg TRS scheme. This method relies on a finite-difference approximation to J and, as far as we know, is the only way that the TRA has been employed in the RBF literature so far.
- RBFTrust, meaning that one of the three TRS schemes reviewed in Section 2 is combined with the analytic approximations to J and H in Section 3. Recall that those three TRS schemes are: dogleg (Section 2.1), full (end of preamble of Section 2), and 2Dsub (Section 2.2). In order to compare the various possibilities, they have been incorporated into a single Matlab code written from scratch . Moreover, we sometimes test scaling as described in Section 2.4. RBFTrust variations like those are the new approach introduced and advocated in this paper.

With RBFTrust, the RBF formulas for the Jacobian (25) and Hessian (31) were checked against finite-difference approximations produced with DERIVEST (free Matlab program by John D'Errico), and found to agree within the numerical tolerances. The RBFs used in the experiments are those in Table 1.

For every domain Ω, we consider an evaluation set of $N_e >> 1$ points (different from collocation nodes) scattered over Ω. The error ϵ and the interpolation residual to the PDE, R, are monitored there. For instance, the accuracy is estimated via $RMS(\epsilon) := \sqrt{\sum_{i=1}^{N_e} \epsilon_i^2 / N_e}$. The *collocation* residual R_c, of course, is evaluated on the collocation nodes.

Convergence of RBFTrust to a root is declared as soon as μ stagnates as $\mu = \mu_\infty$ and μ_∞ is negligible small. This is a conservative criterion, for when it happens, $RMS(R)$ and $RMS(R_c)$ will typically have stagnated some iterations before, and $RMS(\epsilon)$, even before that.

Table 2: Methods used in Problems I-IV (U/S= unscaled/scaled J and H).

	Method	I **Cubic**	II **Plateau's**	III **Hele-Shaw**	IV **Monge-Ampère**
`fsolve`	dogleg + finite-differences J	U			
RBFTrust	dogleg + analytic J	U	U	U	U
	2Dsub + analytic J and H	U	U,S		U,S
	full + analytic J and H	U	U,S		U,S
linearization	operator-Newton	U	U	U	U

7.1 Example I: diffusion equation with a cubic nonlinearity

The first example is the semilinear equation (42) solved on the square $[0,1]^2$ with f such that the exact solution is $u_{ex}(x,y) = \sin(\pi x)\sin(\pi y)$, and $g = u_{ex}|_{\partial\Omega} = 0$ (see Figure 1). The shrunken J and H are (46) and (47), respectively. We solve this problem on a grid-like pointset. As a starting guess, $\vec{\beta}_0 = (0,\ldots,0)$. No scaling was implemented.

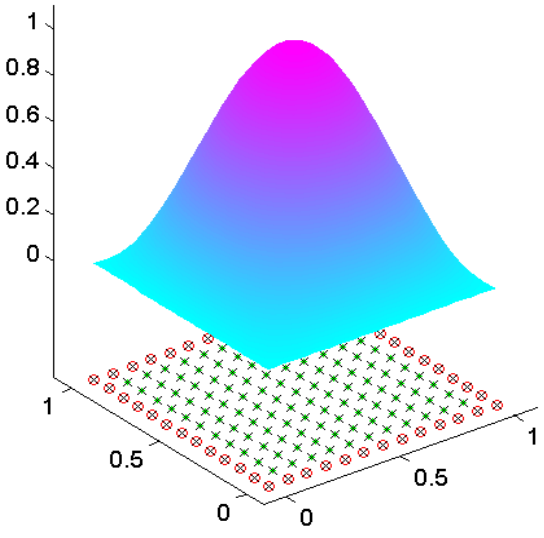

Figure 1: Solution of Example I on an illustrative pointset. Symbols stand for: green • = PDE collocation node; red o = BC collocation node; × = RBF center.

Let us start by picking Wendland's $WC4$ RBF. On Table 3, all three TRS approximations for RBFTrust converge in a similar number of iterations, albeit they do not take the same CPU time. This is so because each TRS approximation takes a different number of function evaluations (each requiring to solve a linear system), although the ratio evaluations/iteration is always $O(1)$–see Section 2. (In fact, 2Dsub ought to be substantially faster than full. The reasons why this does not happen may be that N is not yet large enough, and/or that full uses

Table 3: N is the number of RBFs, μ_∞ the value of the merit function at the converged minimum, ϵ the error, κ the condition number, and the figures in parentheses along the number of iterations are the CPU time in seconds.

Example I solved with RBFTrust and RBF=WC4 ($L = 0.3$)

N	μ_∞	RMS(ϵ)	$\kappa(J)$	iter(dogleg)	iter(full)	iter(2Dsub)
529	$O(10^{-26})$	.01103	109	16(2.18)	16(5.35)	16(5.34)
676	$O(10^{-26})$	.00675	219	16(4.49)	17(11.31)	17(11.20)
784	$O(10^{-26})$	.00511	304	17(7.11)	17(17.60)	17(17.50)
1024	$O(10^{-25})$	.00309	615	17(15.98)	17(39.45)	17(39.39)
1296	$O(10^{-25})$	.00204	1157	17(32.50)	17(122.72)	17(73.08)
1681	$O(10^{-24})$	.00130	2231	17(67.53)	17(161.81)	17(171.64)
2116	$O(10^{-24})$	.00087	3982	18(144.91)	17(322.09)	17(328.72)

Table 4: Same notation as in Table 3, plus the total number of function evaluations (fevals). $N = 2116$ not reported because it takes unacceptably long.

Example I solved with *fsolve* and RBF=WC4 ($L = 0.3$)

N	μ_∞	RMS(ϵ)	$\kappa(J)$	iterations	CPU time (s.)	fevals
529	$O(10^{-25})$	.01103	109	4	308	2210
676	$O(10^{-25})$	.00675	219	4	1063	2885
784	$O(10^{-24})$	.00511	304	4	1446	3385
1024	$O(10^{-24})$	.00309	615	4	4071	4505
1296	$O(10^{-23})$	.00204	1157	4	10577	5785
1681	$O(10^{-23})$	.00130	2231	4	32061	7610

Matlab's built-in `trust` while 2Dsub is a non-optimized code.)

Given the mild nonlinearity of this problem, the operator-Newton method was able to converge from $\vec{\beta}_0$ (as well as from all the other tested initial guesses). Moreover, H is not required in order to enforce convergence, and hence the simplest TRS scheme, dogleg–which only needs J–is the more efficient one. Recall that the three TRS solvers of RBFTrust in Table 3 all use the analytical J and H from Section 2. On the other hand, on Table 4, Example I is solved with `fsolve`, whereby every iteration takes $N - M$ function evaluations (M boundary nodes) in order to approximate J with finite differences. Since the Jacobians are well conditioned, `fsolve` is able to find the root the imperfect J notwithstanding. However, computational times are much longer than with RBFTrust. (Note that μ_∞ is different from before because the convergence criterion of `fsolve` is independent of that used in RBFTrust.)

Let us now pick the MQ RBF. On Tables 5 and 6, only the dogleg TRS scheme was used, both with RBFTrust (i.e. analytical J, Table 5) and `fsolve` (i.e. finite-differences approximation to J, Table 6). Comparing both, not only does `fsolve` take much longer, but the finite-differences J is inadequate to provide convergence already at mild condition numbers (larger than $\kappa(J) \approx 10^8$). After Table 6, we will not show more results involving finite-differences Jacobians.

Also on Table 5, asymptotic exponential convergence–as in linear elliptic

PDEs, see [8, 4]–is hinted: $RMS(\epsilon) = O\left(C^{c/h}\right)$, $0 < C < 1$. The fill distance h is here proportional to $1/\sqrt{N}$ (see Figure 2). The possibility of $c-$convergence remains an appealing feature of RBF collocation also with nonlinear problems.

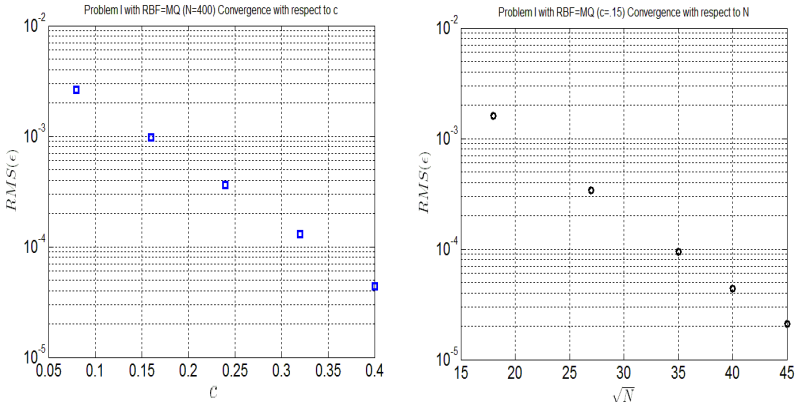

Figure 2: $h - c$ convergence of $RMS(\epsilon)$ in Example I (data from Table 5). Left: error vs. c. Right: error vs. $\sqrt{N}$ (note that in gridlike pointsets, $1/h \sim \sqrt{N}$)

Table 5: Convergence with respect to c and N. R is the interpolation residual.

Example I solved with RBFTrust and $RBF = MQ$

N	c	RMS(ϵ)	RMS(R)	$\kappa(J)$	μ_∞	iterations	CPU t (s.)
400	.08	.00264	3.65	$O(10^2)$	$O(10^{-25})$	23	1.07
400	.16	.00098	1.40	$O(10^4)$	$O(10^{-23})$	28	1.65
400	.24	.00036	0.52	$O(10^7)$	$O(10^{-21})$	28	1.40
400	.32	.00013	0.19	$O(10^9)$	$O(10^{-18})$	37	1.71
400	.40	4.4×10^{-5}	0.07	$O(10^{12})$	$O(10^{-15})$	42	1.77
324	.15	.00160	1.86	$O(10^3)$	$O(10^{-24})$	26	0.65
729	.15	.00034	0.88	$O(10^6)$	$O(10^{-21})$	28	10.3
1225	.15	9.5×10^{-5}	0.44	$O(10^8)$	$O(10^{-18})$	31	62.5
1600	.15	4.4×10^{-5}	0.28	$O(10^{10})$	$O(10^{-17})$	32	136.7
2025	.15	2.1×10^{-5}	0.17	$O(10^{11})$	$O(10^{-15})$	33	296.0

7.2 Example II: Plateau's problem

Plateau's problem (78) is a case of the least-surface equation where the solution can be expressed as a function of (x, y). The solution on the circle Ω centred at the origin and with radius $R < \pi/2$ is known as Scherk's first minimal surface. Let $R = \pi/2 - s$ with $s > 0$. As $s \to 0$, $|\nabla u|$ close to $\partial\Omega$ grows unbounded, thus

Table 6: Same experiments as in Table 5. The case $N = 1225$ was aborted after 94 iterations due to exceedingly slow convergence (if at all).

Example I solved with *fsolve* and $RBF = MQ$

N	c	RMS(ϵ)	RMS(R)	$\kappa(J)$	μ_∞	iterations	CPU time (s.)	fevals
400	.08	.00264	3.65	$O(10^2)$	$O(10^{-24})$	6	117	2382
400	.16	.00098	1.40	$O(10^4)$	$O(10^{-19})$	10	189	3575
400	.24	.00036	0.52	$O(10^7)$	$O(10^{-14})$	129	1765	32530
324	.15	.00160	1.86	$O(10^3)$	$O(10^{-19})$	8	80	2313
729	.15	.00034	0.88	$O(10^6)$	$O(10^{-17})$	33	6956	20659
1225	.15	–	–	$O(10^8)$	11.98	94	131181	78503

becoming numerically more challenging–see Figure 3.

$$\nabla \cdot \Big(\frac{\nabla u}{\sqrt{1 + |\nabla u|^2}} \Big) = 0 \text{ if } \mathbf{x} \in \Omega, \qquad u = \log(\cos x / \cos y) = u_{ex}|_{\partial\Omega} \text{ if } \mathbf{x} \in \partial\Omega. \quad (78)$$

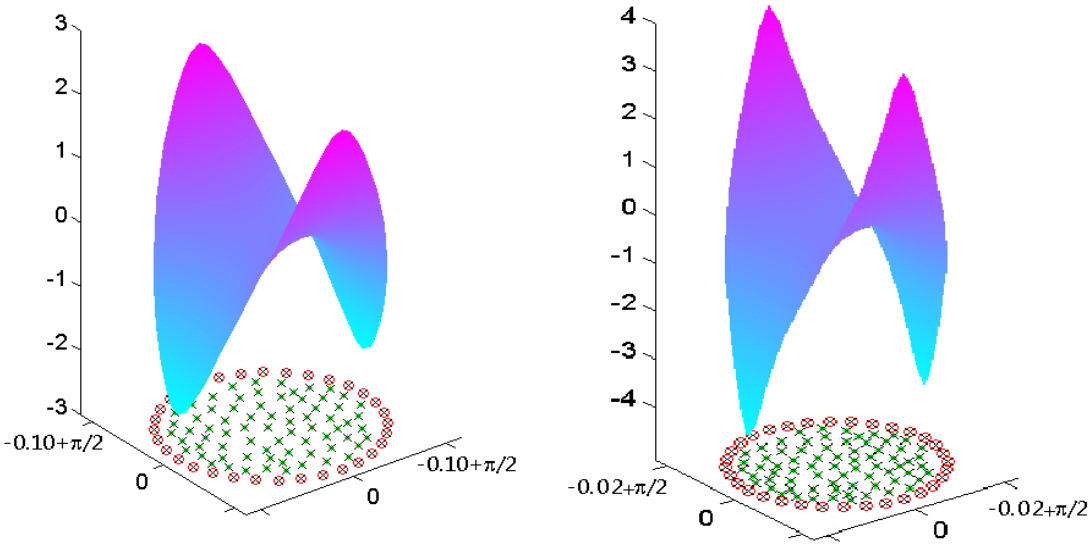

Figure 3: Solutions of Example II with radius $R = -.10 + \pi/2$ (left) and $R = -.02 + \pi/2$ (right). Note that scales vary.

We solve two instances of this problem (with $s = .10$ and $s = .02$) with RBFTrust on a scattered pointset with $N = 795$ nodes, of which 80 are along the boundary. The RBF is $MATERN(\alpha = 11, c = .10)$. With Example II, we test five TRS methods: dogleg, full (scaled and unscaled), and 2Dsub (scaled and unscaled). The scaling matrix is (21), and Δ_0 is 1 if unscaled and 10^{10} if scaled. The initial guess $\vec{\beta}_0$ is a random vector (but the same for all experiments).

Convergence is plotted in Figure 4. In both cases there is apparently a unique root:

- For s=.02, $\vec{\alpha}_*$: $\{RMS(\epsilon) = .00550132, RMS(R) = 1021.08, \kappa(J) = 112276\}$.

- For s=.10, $\vec{\alpha}_*$: $\{RMS(\epsilon) = .00444585, RMS(R) = 42.576, \kappa(J) = 74966\}$.

While for $s = .10$ all five TRS variants of RBFTrust correctly converge to the root, in the more difficult case ($s = .02$) dogleg stagnates at $\mu >> 0$ (with $RMS(\epsilon) \approx .443$). Actually, this value is not a local, non-root minimum (in fact $\kappa(J) = 275042$), but rather the dogleg steps yield the same very small drop as the Cauchy steps (≈ 193). Therefore, Hessian information is absolutely critical in order to solve Example II with $s = .02$. Both the full and 2Dsub use H. However, since 2Dsub solves the TRS in a two-dimensional, rather than $(N - M)-$dimensional, space, each 2Dsub iteration is much cheaper than a full iteration. Therefore, 2Dsub solves the problem faster, even though it may take more iterations.

We also stress the robustness of RBFTrust, spanning 33 orders of magnitude (from $\mu(\vec{\alpha}_0) \approx 10^{12}$ to $\mu(\vec{\alpha}_\infty) \approx 10^{-21}$) over a non-convex merit function landscape and from a Gaussian random guess. Many more unreported experiments show that the qualitative results in Figure 4 hold for other RBFs and for other random guesses, always finding the same RBF solution.

On the other hand, scaling proves a disappointment, and in three out of four cases scaled versions take longer to converge. This is seemingly due to the worsening of $\kappa(H)$.

Table 7: Example II (with $s = 1$) solved with the operator-Newton method from a perturbed Laplacian guess. Left: no perturbation. Middle: convergence still takes place. Right: a slightly larger perturbation leads to divergence.

	$\vec{\alpha}_0$=Laplacian		$\vec{\alpha}_0$=Laplacian + $\vec{\delta}$/250		$\vec{\alpha}_0$=Laplacian + $\vec{\delta}$/247	
iteration ♯	RMS(ϵ)	RMS(R_c)	RMS(ϵ)	RMS(R_c)	RMS(ϵ)	RMS(R_c)
0	.00124	.1464	.00124	0.1464	0.00124	0.1464
1	.000378	.059	.0603	2.29	0.1221	3.93
2	.000162	.030	.0067	1.18	0.0594	5.42
3	5.00×10^{-5}	.0151	.0020	0.4053	0.0781	17.29
4	4.20×10^{-5}	.0081	.00071	0.2001	0.5668	19185
10	1.43×10^{-5}	.00024	1.73×10^{-5}	0.0041	∞	∞
60	1.39×10^{-5}	6.6×10^{-16}	1.39×10^{-5}	6.15×10^{-15}	∞	∞

Let us use Example II to illustrate the shortcomings of the operator-Newton method. As expected, and unlike RBFTrust before, the operator-Newton method was unable to find a root from a random guess. Therefore, we picked $\vec{\beta}_0$ from $\vec{\alpha}_0$ resulting of interpolating the solution of a Laplace equation with the same BCs over the same pointset (we call this the "Laplacian guess"). Even with the Laplacian guess, the radius of Ω must be substantially smaller than

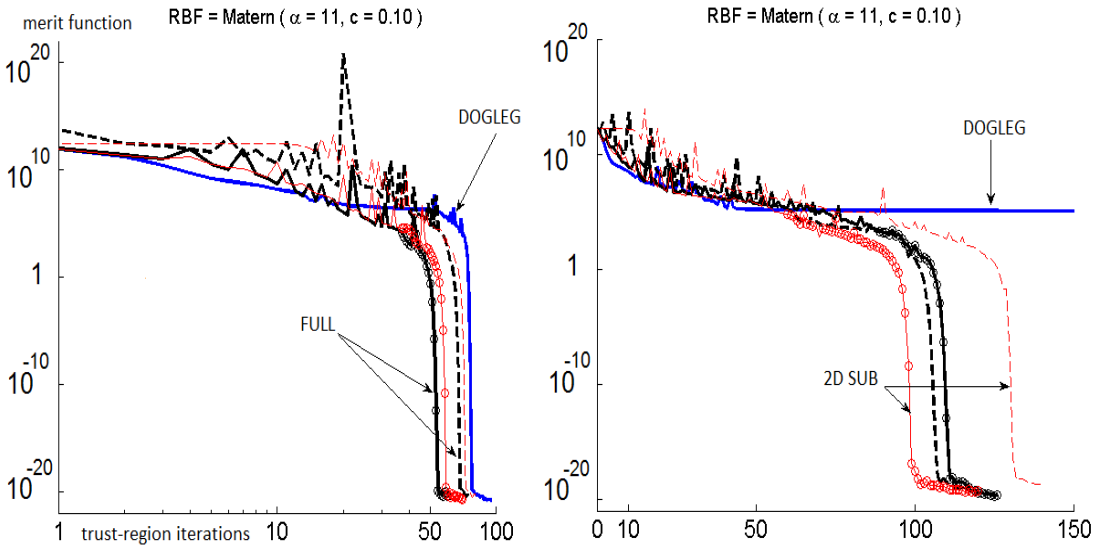

Figure 4: Merit function μ vs. RBFTrust iterations in Example II (left: radius $R = \pi/2-.10$; right: $R = \pi/2 - .02$). TRS methods are: dogleg (blue), full (black) and 2Dsub (red). Dashed lines stand for scaled J and H, and circles (only on the unscaled curves) for a positive-definite Hessian at that iteration. The Hessian is critical for finding a root as $s \to 0^+$ (and $|\nabla u| \to \infty$ close to $\partial\Omega$). Note also that scaled methods tend to take more iterations, which is undesirable.

before in order to get the operator-Newton method to converge. Consequently, we kept the same RBFs and N but shrank Ω to $R = \pi/2 - 1$(i.e. we let $s = 1$). Notice that the solution is much flatter now and, in sum, the BVP is much less challenging than the one solved before with RBFTrust. Only after those simplifications, the operator-Newton method was finally able to converge. Further investigating the performance of the operator-Newton, let $\vec{\delta}$ be a random vector drawn from a standard normal distribution, $\mathcal{N}(0, 1)$. Table 7 shows the effect on convergence of a small perturbation over the Laplacian guess, highlighting the lack of robustness of the operator-Newton method. Notice that the residual R_c in Table 7 is the *collocation* residual, and that values of RMS$(R_c) \lesssim 10^{-14}$ strongly hint to a root of the collocation system (6).

Wrapping up Example II, the operator-Newton method performs very poorly compared to RBFTrust with dogleg, and the latter worse than Hessian-based RBFTrust. In fact, in the presence of high gradients, even dogleg with analytic J fails, and second-order information derived from the analytic H becomes indispensable.

7.3 Example III: simulation of powder injection molding

The three-dimensional creeping flow of molten polymer into a thin cavity can in many cases be modeled by a two-dimensional free-boundary problem involving the p-Laplace operator. This is known as the Hele-Shaw approximation [1, 2]. At every timestep, the following nonlinear elliptic BVP for the pressure field $u(x, y)$ must be solved (here in dimensionless units):

$$\nabla \cdot \Big(|\nabla u|^{\gamma} \nabla u \Big) = 0 \text{ if } \mathbf{x} \in \Omega, \qquad \text{with mixed BCs} \begin{cases} u = 1 & \text{if } \mathbf{x} \in \Gamma_I \\ u = 0 & \text{if } \mathbf{x} \in \Gamma_F \\ \partial u / \partial N = 0 & \text{if } \mathbf{x} \in \Gamma_W. \end{cases} \tag{79}$$

Γ_I represents the injection slit (where the pressure is enforced by the injection machine), Γ_F is the (frozen) free boundary, Γ_W are the walls of the floor view of the mold, and $\Gamma_I \cup \Gamma_F \cup \Gamma_W = \partial\Omega$. A typical value is $\gamma = 0.6$, which models polyethylene. Figure 5 (left) shows a FEM numerical solution obtained over a very fine mesh, which we take as a reference.

In addition to the nonlinearity, this problem has two more challenging features. The first one are the BCs of derivative type, which degrade the quality of the RBF solution. The second issue is the singularity in u which takes place on both ends of Γ_I due to the change of type of the BCs. Since the RBF interpolant is made up of infinitely smooth functions, it cannot reproduce the singularities and brings about oscillations around them, resulting in large residual peaks–see Figure 5 (right).

We counter the former difficulty by enforcing the PDE also on some (here, all but two) of the boundary nodes. This is the so-called PDEBC strategy [14], whereby as many additional RBFs are added (usually placed off the boundary) to keep the system square–see Figure 5 (left). Let us now address the presence of nonsmooth boundary singularities. In the case of Laplacian flows (i.e. $\gamma = 0$), the singularities can be effectively captured by enriching the RBF interpolant as in (32) with Motz functions [4]:

$$h_k(r, \theta) = r^{(2k-1)/2} \cos\Big[\Big(\frac{2k-1}{2}\Big)\theta\Big], \qquad k \geq 1. \tag{80}$$

For $\gamma = 0$, Motz functions (80) do not contribute residual, because they are harmonic. Table 8 shows the results of testing the same idea with the p-Laplacian flow (79). The collocation pointset is made up of $N = 1152$ scattered nodes (166 along the boundary). Note that now there are $1152+164$ RBFs due to PDEBC (the PDE is not enforced on the singularities, but the Dirichlet BCs are). Two RBFs were used, the MQ and the IMQ. The initial guess is the solution of a Laplace equation with the same BCs. While the overall effect is not as dramatic as in the linear case, adding a few (n) Motz functions still shows its benefits as the condition numbers grow ($n = a + a$ means that a Motz functions (80) with $k = 1, \ldots, a$ are centred on each singularity). Without enrichment, accuracy does not improve out of a larger value of c (it actually worsens). Moreover, iterations are fewer, and the derivatives of the solution close to the injection area

are also much improved, which is important if, for instance, the velocity field is needed. On the other side, because Motz functions *do* contribute residual when $\gamma \neq 0$, there seems to be an optimal number of them, depending on the RBF discretization.

Table 8: Effect of n Motz functions on the RBF interpolant in Example III.

RBF=IMQ (c = .75). Solver: RBFTrust+dogleg TRS

n	iter	μ_∞	RMS(ϵ)	MAX(ϵ)	RMS(R)	MAX(R)	$\kappa(J)$
0 + 0	16	1.06×10^{-22}	.0028	.015	20.35	497.07	1.59×10^{10}
1 + 1	15	2.01×10^{-23}	.0101	.020	1.94	59.11	4.25×10^{10}
3 + 3	14	5.84×10^{-24}	.0032	.009	0.23	6.14	8.37×10^{10}
6 + 6	16	3.60×10^{-23}	.0027	.014	0.06	1.94	3.88×10^{10}
9 + 9	19	5.77×10^{-22}	.0034	.013	1.06	25.96	1.63×10^{10}

RBF=IMQ (c = 1.50). Solver: RBFTrust+dogleg TRS

n	iter	μ_∞	RMS(ϵ)	MAX(ϵ)	RMS(R)	MAX(R)	$\kappa(J)$
0 + 0	49	1.35×10^{-16}	.0051	.016	22.69	547.59	1.32×10^{11}
1 + 1	25	9.90×10^{-19}	.0018	.005	0.01	0.27	2.37×10^{11}
2 + 2	20	9.28×10^{-19}	.0018	.005	0.01	0.28	2.73×10^{11}
3 + 3	23	1.47×10^{-18}	.0028	.005	0.01	0.40	1.02×10^{13}
4 + 4	22	9.66×10^{-19}	.0033	.008	0.02	0.55	1.31×10^{14}

Despite the remarkable "filing" of the residual peaks by the Motz functions, they do not entirely vanish (see Figure 5, bottom right). For that reason, the operator-Newton method also fails to solve this problem, even if Motz functions are incorporated into the interpolant (results not shown).

Example III was solved exclusively with the RBFTrust+dogleg method. The reason is that condition numbers are higher now and they actually worsen with n (see Table 8). The dogleg TRS scheme is the only one which can handle A matrices with such condition numbers, thanks to the fact that for it, $A = J^T J$–see item 3 in Section 8 for further details.

7.4 Example IV: Monge-Ampère equation in 3D

The numerical handling of Monge-Ampère (and, in general, of second-order fully nonlinear equations) is considered very difficult. According to [15], the following was the only numerical example of a Monge-Ampère PDE in 3D at the time of publication:

$$\det(D_3^2 u) = f = (1 + r^2)\exp(3r^2/2) \text{ if } \mathbf{x} \in \Omega, \qquad u = \exp(r^2/2) \text{ if } \mathbf{x} \in \partial\Omega, \quad (81)$$

where $r = \sqrt{x^2 + y^2 + z^2}$. The domain is the unit cube $[0, 1]^3$ with u_{ex} being the same as the BC. When $f > 0$ (as here), (81) has a unique convex solution (for which the PDE is elliptic), but may have other non-convex solutions [7].

We solved (81) on a lattice-like pointset with $N = 2197$ nodes (of which 866 are on $\partial\Omega$), using the solution of the related Poisson equation as a starting

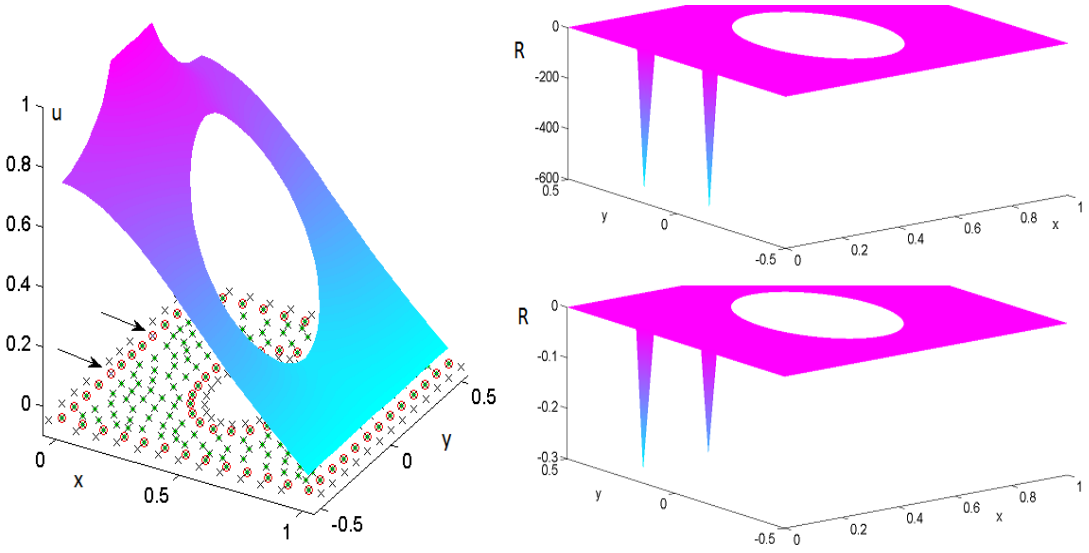

Figure 5: Left: Solution of Example III. The injection inlet Γ_I is between the arrows. Notice the extra RBF centres ($\times$) placed outside for PDEBC; except at both ends of Γ_I– where only the Dirichlet BC is enforced, for u is not smooth there. Along the front Γ_F, $u = 0$. Right: residual peaks at the singularities with $0 + 0$ Motz functions (up) and $2 + 2$ (bottom). Note that the vertical scale varies.

guess. The highly nonlinear character of this PDE is reflected in the fact that, for $N \gtrsim 900$ RBFs, RBFTrust with dogleg TRS ceases to converge to a root. For larger problems than that, the Hessian is needed in order to obtain convergence. This pattern is confirmed by all three RBFs used on Figure 6.

In Figure 6, the dogleg iterations end up having a similar (and extremely small) convergence rate with steepest descent and, for practical purposes, get all but stuck. On the other hand, using H results in μ dropping by several orders of magnitude the value of the Cauchy step, and eventually in convergence. For each of the RBFs in Figure 6, the full and 2Dsub TRS schemes converge to the same minimum, suggesting that RBFTrust is correctly picking the unique convex solution (see Table 9).

In Example IV, scaling of 2Dsub and full (dashed lines on Figure 6; implemented as in Example II) had again a mixed performance, taking sometimes more iterations than the unscaled versions of RBFTrust and even failing to converge with the MQ RBF (red dashed curve on the leftmost part on Figure 6). Seemingly, diagonal scaling according to (21) was destabilizing RBFTrust.

As was to be expected, the operator-Newton method systematically failed to solve Example IV except for small values of N.

Table 9 shows that the final accuracy (and conditioning) of the root is quite dependent on the RBF used, and roughly proportional to the final interpolation residual. We underscore that all accuracies on Table 9 are at least one order of

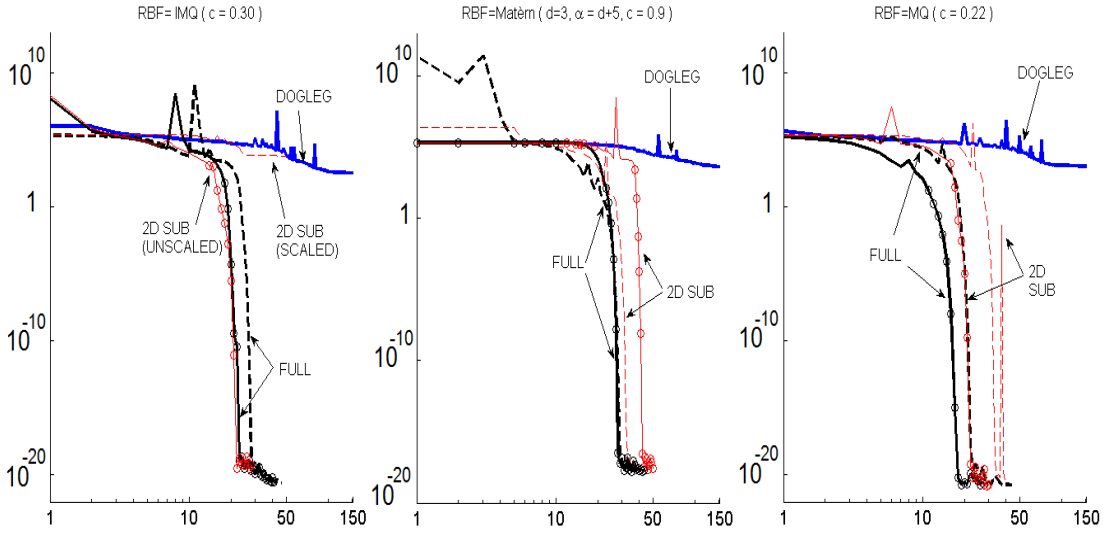

Figure 6: Merit function vs. RBFTrust iterations in Example IV, with $N = 2197$ and using 3 RBFs: IMQ($c = .30$) (left), MATERN($\alpha = 8, c = .90$) (centre), MQ($c = .22$) (right). Symbols as in Figure 4. Again, H is vital for convergence.

Table 9: Properties of $\vec{\alpha}_\infty$ in Problem IV for several RBFs.

RBF	μ_∞	RMS(ϵ)	RMS(R)	$\kappa(J)$	$\kappa(H)$
Matérn($\alpha = 12, c = .80$)	3.2×10^{-20}	.00760	121.51	202481	4.10×10^{10}
IMQ($c = .30$)	2.5×10^{-21}	.00187	26.61	1.16×10^{6}	1.36×10^{12}
MQ($c = .22$)	1.8×10^{-21}	.00027	6.03	173323	3.00×10^{10}
Matérn($\alpha = 8, c = .90$)	2.6×10^{-18}	5.08×10^{-5}	1.13	14828	2.20×10^{8}

magnitude better that those reported in [15].

8 Discussion of the results of problems I-IV

1. In all well-conditioned problems which we have run, we have found a distinct minimum of the merit function μ below or around the machine error ($\approx 10^{-15}$). In such cases, we observe that, at the converged minimum: i) $\kappa(J)$ clearly is not numerically singular; ii) $H > 0$; and iii) the final rate of convergence is quadratic. For those reasons we tend to believe that those converged minima are, in fact, roots of the collocation system. Moreover, those roots are independent of the initial guess or TRS method used, strongly suggesting that they are, in fact, unique.

2. Not in a single numerically stable experiment, whether reported or not,

did we find evidence of multiple roots.

3. The tradeoff between accuracy and stability–a hallmark of Kansa's method–carries over to nonlinear equations, albeit via a different mechanism. Picking RBF interpolation spaces with better approximation properties (by letting N or the RBF shape parameter grow), pushes the condition number of the matrices involved (J and A) towards numerical singularity. This, in turn, slows down the convergence rate of RBFTrust and lifts the value of the minimum of the merit function that can be resolved. Beyond the threshold of numerical instability, the RBFTrust iterations get stuck in a local minimum with much residual and poor accuracy.

   This tradeoff has a bearing on the better choice of the TRS approximation method when higher accuracy is desired. On the one hand, $A = H$ (in the full and 2Dsub methods) should entail a better modeling of the complicated residual landscape and thus allow deeper steps to be taken, resulting in fewer iterations and better probability to find the root. On the other hand, $A = J^T J$ (in the dogleg method) is numerically advantageous because inverting $J^T J\vec{x} = \vec{b}$ can be split into $J^T\vec{y} = \vec{b}$ and then $J\vec{x} = \vec{y}$. Assuming–from (10)–that $\kappa(H) \approx \kappa(J^T J) = \kappa^2(J)$, the dogleg method can in effect yield better RBF approximations of the solution before becoming unstable than would be possible using the full Hessian.

4. The performance of the RBFTrust with the 2Dsub TRS solver is also remarkably affected by ill-conditioning. We observed that for Hessians with $\kappa(H) \gtrsim 10^{10}$, the estimation of the smallest algebraic eigenpair via Matlab *eigs* failed because the Lanczos iterations did not converge. In such cases, in order to test the 2Dsub scheme, we computed the exact eigenpair via the QR decomposition of H–thus defeating the purpose of speeding up the computations. In order to fully exploit the two-dimensional TRS approximation within RBFTrust, the Lanczos method must be replaced by another algorithm more resistent to RBF ill conditioning.

5. More often than not, diagonal scaling (21) proved to be a counter–productive addition which compounded the conditioning of H.

9 Conclusions

The RBFTrust approach developed here is a robust, easy to code, fast and accurate solver for quite general nonlinear elliptic equations. Experiments show that it is vastly superior to previous approaches based on the operator-Newton method or the dogleg method with finite-difference Jacobians. Particularly critical is the use of the analytic formulas for the Jacobian and Hessian of the collocation system made available in this paper–especially so when the equation is highly nonlinear.

This paper deals exclusively with strict (Kansa-like) RBF collocation. While the numerical results are very good, the theoretical considerations about solvability and uniqueness in Section 6 were left undecided in either sense. In practice, anyways, strict collocation is moot as long as the converged residuals are negligible. The ultimate goal is to tackle much larger problems, along the lines of the RBF-PU method [21].

10 Acknowledgements

Portuguese FCT funding under grant SFRH/BPD/79986/2011 and a KAUST Visiting Scholarship at OCCAM in the University of Oxford are acknowledged.

The author thanks Holger Wendland for suggesting Example IV and for helfpul discussions while in Oxford. The referees are acknowledged for their meticulous reading, which led to a better and clearer presentation of the paper.

References

[1] F. Bernal and M. Kindelan, *RBF meshless modelling of non-Newtonian Hele-Shaw flow*, Engng. Anal. Bound. Elem. **31**, 863-874 (2007).

[2] F. Bernal and M. Kindelan, *A meshless solution to the p-Laplace equation*, in Progress on Meshless Methods, A.J.M. Ferreira, E.J. Kansa, G.E. Fasshauer and V.M.A. Leitão (eds.) Springer, 17-35 (2009).

[3] F. Bernal, G.Gutiérrez and M.Kindelan, *Use of singularity capturing functions in the solution of problems with discontinuous boundary conditions*, Eng. Anal. Bound. Elem. **33**(2), 200-208 (2009).

[4] F. Bernal and M. Kindelan, *On the enriched RBF method for singular potential problems*, Eng. Anal. Bound. Elem. **33**, 1062-1073 (2009).

[5] B. Buchberger, *Groebner bases: an algorithmic method in polynomial ideal theory*. N.K. Bose (ed.), Recent Trends in Multidimensional Systems Theory, Reidel 184-232 (1985).

[6] R.H. Byrd, R.B. Schnabel and G. A. Schultz, *Approximate solution of the trust regions problem by minimization over two-dimensional subspaces*, Mathematical Programming **40**, 247-263 (1988).

[7] S.Y. Cheng and S.T Yau, *On the regularity of the Monge-Ampère equation* $\det(\partial^2 u/\partial x_i \partial x_j) = F(x,u)$. Commun. Pure Appl. Math. **30**(1), 41-68 (1977).

[8] A.H.D. Cheng, M.A. Golberg, E.J. Kansa and T. Zammito, *Exponential convergence and h-c multiquadric collocation method for partial differential equations*. Numer. Meth. Part. Differ. Equat. **19**, 571-594 (2003).

[9] P.P. Chinchapatnam, K. Djidjeli and P.B. Nair, *Radial basis function meshless method for the steady incompressible Navier-Stokes equations*, Intern. Jour. Comp. Math. **84**, 1509-1521 (2007).

[10] T. Dierkes, O. Dorn, F. Natterer, V. Palamodov and H. Sielschott, *Fréchet derivatives for some bilinear inverse problems*, SIAM J. Appl. Math. **62**(6), 2092-2113 (2002).

[11] G.E. Fasshauer, *Newton Iteration with Multiquadrics for the Solution of Nonlinear PDEs*, Comput. Math. Applic. (**43**), 423-438 (2002).

[12] G.E. Fasshauer, C. Gartlang and J. Jerome, *Algorithms Defined by Nash Iteration: Some Implementations via Multilevel Collocation and Smoothing*, J. Comp. Appl. Math. (**119**), 161-183 (2000).

[13] G.E. Fasshauer, *Meshfree Approximation Methods with Matlab*. Interdisciplinary Mathematical Sciences–Vol. 6. World Scientific Publishers, Singapore (2007).

[14] A.I. Fedoseyev, M.J. Friedman and E.J. Kansa, *Improved multiquadric method for elliptic partial differential equations via PDE collocation on the Boundary*, Comput. Math. Appl. **43**, 439-455 (2002).

[15] X. Feng and M. Neilan, *Vanishing moment method and moment solutions for fully nonlinear second order partial differential equations*, J. Sci. Comput. **38**(1), 74-98 (2009).

[16] B. Fornberg, E. Larsson and N. Flyer, *Stable computations with Gaussian radial basis functions*, SIAM J. Sci. Comput. **33**, 869-892 (2011).

[17] H. Ghaderi, *Mountain Pass Theorems with Ekeland's Variational Principle and an Application to the Semilinear Dirichlet Problem.* Uppsala University, U.U.D.M. Project Report 2011:30 (2011).

[18] Y.C. Hon and R. Schaback, *On unsymmetric collocation by radial basis functions*, Applied Mathematics and Computation **119**, 177-186 (2001).

[19] E.J. Kansa, *Multiquadrics–a scattered data approximation scheme with applications to computational fluid-dynamics. I. Surface approximations and partial derivative estimates*, Comput. Math. Appls. **19**, 127-145 (1990).

[20] E.J. Kansa, *Multiquadrics–a scattered data approximation scheme with applications to computational fluid-dynamics. II. Solutions to parabolic, hyperbolic and elliptic partial differential equations*, Comput. Math. Appls. **19**, 147-161 (1990).

[21] A. Safdari-Vaighani, A. Heryudono and E. Larsson, *A radial basis function partition of unity collocation method for convection-diffusion equations arising in financial applications*, Journal of Scientific Computing **64**, 341-367 (2015).

[22] L. Ling, R. Opfer and R. Schaback, *Results on meshless collocation techniques*. Engineering Analysis with Boundary Elements **30**(4), 247-253 (2006).

[23] J.J. Moré and D.C. Sorensen, *Computing a trust region step*, SIAM Journal on Scientific and Statistical Computing, **4** 553-572 (1983).

[24] J. Nocedal and S.J. Wright, *Numerical Optimization*. Springer Series in Operations Research (1999).

[25] R.B. Platte and T.A. Driscoll, *Eigenvalue stability of radial basis function discretizations for time-dependent problems*. Computers Math. Applic. **51**. 1251-1268 (2006).

[26] G.A. Schultz, R.B. Schnabel and R.H. Byrd, *A family of trust-region-based algorithms for unconstrained minimization with strong global convergence properties*. SIAM J. Num. Anal. **22**, 47-67 (1985).

[27] H. Wendland, *Scattered Data Approximation*. Cambridge Monographs on Applied and Computational Mathematics. Cambridge University Press, Cambridge, UK (2004).

[28] J. Xu, *A Novel Two-Grid Method for Semilinear Elliptic Equations*, SIAM J. Sci. Comput. **15**(1), 231-237 (1993).