

# A branch and bound algorithm for the global optimization of Hessian Lipschitz continuous functions

Jaroslav M. Fowkes · Nicholas I. M. Gould ·  
Chris L. Farmer

Received: 31 October 2011 / Accepted: 5 June 2012  
© Springer Science+Business Media, LLC. 2012

**Abstract** We present a branch and bound algorithm for the global optimization of a twice differentiable nonconvex objective function with a Lipschitz continuous Hessian over a compact, convex set. The algorithm is based on applying cubic regularisation techniques to the objective function within an overlapping branch and bound algorithm for convex constrained global optimization. Unlike other branch and bound algorithms, lower bounds are obtained via nonconvex underestimators of the function. For a numerical example, we apply the proposed branch and bound algorithm to radial basis function approximations.

**Keywords** Global optimization · Lipschitzian optimization · Branch and bound · Nonconvex programming

## 1 Introduction

In this paper, we are interested in solving the global optimization problem

$$\min_{x \in \mathcal{D}} f(x) \quad (1.1)$$

where  $\mathcal{D} \subset \mathbb{R}^n$  is a compact, convex set and  $f: \mathcal{C} \rightarrow \mathbb{R}$  is a twice-continuously differentiable nonconvex function defined on a suitable compact set  $\mathcal{C} \subset \mathbb{R}^n$  containing  $\mathcal{D}$ . Additionally,

---

J. M. Fowkes (✉)  
School of Mathematics, University of Edinburgh, The King's Buildings, Edinburgh EH9 3JZ, UK  
e-mail: jaroslav.fowkes@ed.ac.uk

N. I. M. Gould  
Computational Science and Engineering Department, Rutherford Appleton Laboratory, Chilton,  
Oxfordshire OX11 0QX, UK  
e-mail: nick.gould@stfc.ac.uk

C. L. Farmer  
Mathematical Institute, University of Oxford, 24-29 St Giles', Oxford OX1 3LB, UK  
e-mail: chris.farmer@maths.ox.ac.uk

we assume that  $f$  has a Lipschitz continuous Hessian on  $\mathcal{C}$ . Global optimization arises in many application areas including predicting the structures of proteins, managing financial portfolios, modelling chemical processes as well as managing environmental systems, to name but a few (see Floudas and Pardalos 1999 for more application areas). While global optimization is a challenging problem (Kreinovich and Kearfott 2005), a variety of deterministic and stochastic solution methods have been suggested (Androulakis et al. 1995; Horst and Pardalos 1995; Pardalos and Romeijn 2002) and this remains a very active area of research (see Neumaier 2004 and extensive references therein). Most of these approaches fall into one of three categories: exact or complete methods (mostly based on branch and bound, see Neumaier 2004), heuristic or incomplete methods (mostly based on stochastic search, see Spall 2003) and surrogate based methods (based on replacing the objective function by a cheap approximation, see Jones 2001). Our approach to solving the global optimization problem (1.1) falls into the first category, it is a deterministic branch and bound algorithm drawing on established ideas from both the local and global optimization communities. To the best of our knowledge, it differs from other branch and bound algorithms in that it employs nonconvex underestimators of the function to obtain the required lower bounds.

The outline of the paper is as follows. In Sect. 2 we describe in detail our proposed branch and bound algorithm for solving the global optimization (1.1). We then go on to prove that under suitable assumptions the proposed algorithm converges to the global minimum of (1.1) in Sect. 3. Section 4 describes how we can efficiently calculate the necessary lower bounds required by our branch and bound algorithm while Sect. 5 outlines a faster heuristic version of our algorithm. Then in Sect. 6 we apply our proposed algorithm to radial basis functions and provide numerical examples to establish performance of the algorithm in this setting. Finally, we draw conclusions in Sect. 7.

### 1.1 Notation

Let  $g(x) := \nabla_x f(x)$  and  $H(x) := \nabla_{xx} f(x)$  be the gradient and Hessian of  $f(x)$ , and  $\|\cdot\|$  denote the  $\ell_2$ -norm. For matrices the  $\ell_2$ -norm becomes the induced (or spectral) norm, i.e. for a matrix  $A$

$$\|A\| := \max_{\|x\|=1} \|Ax\|$$

where the  $\|\cdot\|$  on the right hand side denotes the usual vector  $\ell_2$ -norm. Following convention, let  $C^k$  denote the space of  $k$ -continuously differentiable functions. Since  $\mathcal{C}$  is compact and  $f \in C^2(\mathcal{C})$ , there are constants  $L > 0$  and  $L_g > 0$  for which

$$\|g(x)\| \leq L \quad \text{and} \quad \|H(x)\| \leq L_g \tag{1.2}$$

for all  $x \in \mathcal{C}$ . It follows that  $L$  is an  $\ell_2$ -norm Lipschitz constant for  $f(x)$ , and thus

$$|f(x) - f(y)| \leq L\|x - y\| \tag{1.3}$$

for all  $x, y \in \mathcal{C}$ . Furthermore,  $L_g$  is a gradient Lipschitz constant for  $f(x)$ , i.e.

$$\|g(x) - g(y)\| \leq L_g\|x - y\|$$

for all  $x, y \in \mathcal{C}$ . In addition, we have assumed that  $f$  has a Lipschitz continuous Hessian, so let  $L_H$  be a Hessian Lipschitz constant for  $f$  over  $\mathcal{C}$ , i.e. a constant  $L_H > 0$  such that

$$\|H(x) - H(y)\| \leq L_H\|x - y\|$$

for all  $x, y \in \mathcal{C}$ , where the  $\|\cdot\|$  on the left hand side denotes the induced matrix norm. Note that, within  $\mathcal{C}$ , these are all global Lipschitz constants.

Now let  $\mathcal{B} \subset \mathcal{C}$  denote the  $n$ -dimensional closed ball of radius  $r(\mathcal{B}) > 0$  centred at some  $x_{\mathcal{B}} \in \mathcal{B}$ , i.e.

$$\mathcal{B} = \{x \in \mathbb{R}^n : \|x - x_{\mathcal{B}}\| \leq r(\mathcal{B})\}.$$

Furthermore, let  $L_H(\mathcal{B})$  denote a local Hessian Lipschitz constant for  $f(x)$  over the ball  $\mathcal{B}$ . Unlike the global Lipschitz constants, we need to be able to calculate a numerical approximation to the local Hessian Lipschitz constant for our proposed branch and bound algorithm. We discuss how this is done in detail in an application to radial basis functions in Sect. 6.2.

## 2 Description of the algorithm

In an attempt to solve the global optimization problem (1.1) we develop an extension of the canonical branch-and-bound algorithm (see e.g. Horst 1986) with bounds inspired by the trust region subproblem (see Chapter 7 of Conn et al. 2000). Our branching comprises a systematic covering and refinement of  $\mathcal{D}$  by balls  $\mathcal{B}$ , while our bounding requires we compute both lower and upper bounds on the minimum of  $f$  over each  $\mathcal{B}$ . We will return to the branching when we define our algorithm.

To find a lower bound for the minimum of  $f$  over  $\mathcal{B}$ , we proceed as follows. The first order Taylor expansion with the integral form for the remainder is

$$f(x) = f(x_{\mathcal{B}}) + (x - x_{\mathcal{B}})^T g(x_{\mathcal{B}}) + \int_0^1 (1 - \tau)(x - x_{\mathcal{B}})^T H(x_{\mathcal{B}} + \tau(x - x_{\mathcal{B}}))(x - x_{\mathcal{B}})d\tau \tag{2.1}$$

for all  $x, x_{\mathcal{B}} \in \mathcal{B}$  (see e.g. Cartan 1971, Thm. 5.6.1, p. 70). Notice that

$$\begin{aligned} \int_0^1 (1 - \tau)(x - x_{\mathcal{B}})^T H(x_{\mathcal{B}})(x - x_{\mathcal{B}})d\tau &= (x - x_{\mathcal{B}})^T H(x_{\mathcal{B}})(x - x_{\mathcal{B}}) \int_0^1 (1 - \tau)d\tau \\ &= \frac{1}{2}(x - x_{\mathcal{B}})^T H(x_{\mathcal{B}})(x - x_{\mathcal{B}}) \end{aligned}$$

and hence

$$\frac{1}{2}(x - x_{\mathcal{B}})^T H(x_{\mathcal{B}})(x - x_{\mathcal{B}}) - \int_0^1 (1 - \tau)(x - x_{\mathcal{B}})^T H(x_{\mathcal{B}} + \tau(x - x_{\mathcal{B}}))(x - x_{\mathcal{B}})d\tau = 0 \tag{2.2}$$

for all  $x, x_{\mathcal{B}} \in \mathcal{B}$ . Adding (2.2) to the right hand side of (2.1) we deduce that for all  $x, x_{\mathcal{B}} \in \mathcal{B}$

$$f(x) = f(x_{\mathcal{B}}) + (x - x_{\mathcal{B}})^T g(x_{\mathcal{B}}) + \frac{1}{2}(x - x_{\mathcal{B}})^T H(x_{\mathcal{B}})(x - x_{\mathcal{B}}) + \int_0^1 (1 - \tau)(x - x_{\mathcal{B}})^T [H(x_{\mathcal{B}} + \tau(x - x_{\mathcal{B}})) - H(x_{\mathcal{B}})](x - x_{\mathcal{B}})d\tau. \tag{2.3}$$

Taking the absolute value of the integral and invoking the Hessian Lipschitz continuity, we find that for all  $x \in \mathcal{B}$

$$\begin{aligned} & \left| \int_0^1 (1-\tau)(x-x_{\mathcal{B}})^T [H(x_{\mathcal{B}}+\tau(x-x_{\mathcal{B}})) - H(x_{\mathcal{B}})](x-x_{\mathcal{B}})d\tau \right| \\ & \leq \int_0^1 (1-\tau)\|H(x_{\mathcal{B}}+\tau(x-x_{\mathcal{B}})) - H(x_{\mathcal{B}})\|\|x-x_{\mathcal{B}}\|^2d\tau \\ & \leq \int_0^1 (1-\tau)L_H(\mathcal{B})\|\tau(x-x_{\mathcal{B}})\|\|x-x_{\mathcal{B}}\|^2d\tau \\ & = \int_0^1 (1-\tau)\tau d\tau L_H(\mathcal{B})\|x-x_{\mathcal{B}}\|^3 = \frac{L_H(\mathcal{B})}{6}\|x-x_{\mathcal{B}}\|^3. \end{aligned}$$

Thus if we consider the lower cubic bounding function  $m_{\mathcal{B}}^-: \mathcal{B} \subset \mathbb{R}^n \rightarrow \mathbb{R}$  (as in [Nesterov and Polyak 2006](#))

$$m_{\mathcal{B}}^-(x) = f(x_{\mathcal{B}}) + (x-x_{\mathcal{B}})^T g(x_{\mathcal{B}}) + \frac{1}{2}(x-x_{\mathcal{B}})^T H(x_{\mathcal{B}})(x-x_{\mathcal{B}}) - \frac{L_H(\mathcal{B})}{6}\|x-x_{\mathcal{B}}\|^3,$$

we have that

$$m_{\mathcal{B}}^-(x) \leq f(x) \tag{2.4}$$

provides a non-convex under-estimator for all  $x \in \mathcal{B}$ . We can therefore use

$$\alpha(\mathcal{B}) = \min_{x \in \mathcal{B}} m_{\mathcal{B}}^-(x) \tag{2.5}$$

as a lower bound for the global minimum of  $f(x)$  over  $\mathcal{B}$ . It is important to note that  $\alpha(\mathcal{B})$  can be calculated efficiently and we discuss how this is done in Sect. 4.

It is also possible to use a lower bound based on the gradient Lipschitz constant

$$f(x_{\mathcal{B}}) + (x-x_{\mathcal{B}})^T g(x_{\mathcal{B}}) - \frac{L_g(\mathcal{B})}{2}\|x-x_{\mathcal{B}}\|^2$$

although numerical results suggest the cubic lower bounding function detailed above almost always provides a tighter underestimator. Nevertheless, as it is faster to compute it may be more efficient to use this quadratic lower bound in the early stages of the branch and bound algorithm when the cubic lower bound is rather crude.

To find an upper bound for the minimum of  $f$  over  $\mathcal{B}$ , we simply evaluate  $f$  at a feasible point  $x_{\mathcal{B}}^+$  in  $\mathcal{B}$ . We discuss how  $x_{\mathcal{B}}^+$  is calculated in Sect. 2.1. We therefore let

$$\beta(\mathcal{B}) = f(x_{\mathcal{B}}^+) \tag{2.6}$$

be the upper bound for the global minimum of  $f(x)$  over  $\mathcal{B}$ .

The idea behind the algorithm is to recursively partition an initial ball covering the domain  $\mathcal{D}$  into sub-balls until we find a ball (or balls) of sufficiently small size containing the global minimiser of  $f(x)$  over  $\mathcal{D}$ . Since we are able to obtain bounds on the minimum of  $f(x)$  over any ball in  $\mathcal{D}$ , we can use them to discard balls which cannot contain the global minimum, i.e. balls whose lower bound is greater than the smallest upper bound. The complete branch and bound algorithm then proceeds as follows:

**Algorithm 2.1 Branch and bound algorithm for Hessian Lipschitz optimization**

0. *Initialisation:*

- (a) Set  $k = 0$ .
- (b) Let  $\mathcal{B}_0$  be a ball with centre  $x_{\mathcal{B}} \in \mathcal{D}$  of sufficiently large radius to cover  $\mathcal{D}$ .
- (c) Let  $\mathcal{L}_0 = \{\mathcal{B}_0\}$  be the initial list of balls.
- (d) Let  $U_0 = \beta(\mathcal{B}_0)$  be the initial upper bound for  $\min_{x \in \mathcal{D}} f(x)$ .
- (e) Let  $L_0 = \alpha(\mathcal{B}_0)$  be the initial lower bound for  $\min_{x \in \mathcal{D}} f(x)$ .

1. While  $U_k - L_k > \varepsilon$ , repeat the following procedure:

- (a) Remove from  $\mathcal{L}_k$  balls  $\mathcal{B} \in \mathcal{L}_k$  such that  $\alpha(\mathcal{B}) > U_k$ .
- (b) Choose  $\mathcal{B} \in \mathcal{L}_k$  such that  $\alpha(\mathcal{B}) = L_k$ .
- (c) Split  $\mathcal{B}$  into  $3^n$  overlapping sub-balls  $\mathcal{B}_1, \dots, \mathcal{B}_{3^n}$  according to our splitting rule (see Sect. 2.2) and discard any sub-balls which lie entirely outside of  $\mathcal{D}$ . Let  $\mathcal{R}_k$  denote the list of remaining sub-balls and let  $\mathcal{L}_{k+1} := (\mathcal{L}_k \setminus \{\mathcal{B}\}) \cup \mathcal{R}_k$ .
- (d) Set  $U_{k+1} := \min_{\mathcal{B} \in \mathcal{L}_{k+1}} \beta(\mathcal{B})$ .
- (e) Set  $L_{k+1} := \min_{\mathcal{B} \in \mathcal{L}_{k+1}} \alpha(\mathcal{B})$ .
- (f) Set  $k = k + 1$ .

2. Return  $U_k$  as the estimate of the global minimum of  $f(x)$  over  $\mathcal{D}$ .

Note that infeasible balls, i.e. balls which lie entirely outside of  $\mathcal{D}$ , are discarded by the branch and bound algorithm (see step 1c) and that the initial ball  $\mathcal{B}_0$  contains  $\mathcal{D}$ .

2.1 Discarding balls and feasible points

Algorithm 2.1 discards balls  $\mathcal{B}$  which lie entirely outside of  $\mathcal{D}$ . As  $\mathcal{D}$  is a convex set this is easy to check since then the convex programming problem

$$\begin{aligned} \min_{x \in \mathbb{R}^n} & \|x - x_{\mathcal{B}}\|^2 \\ \text{s.t. } & x \in \mathcal{D} \end{aligned}$$

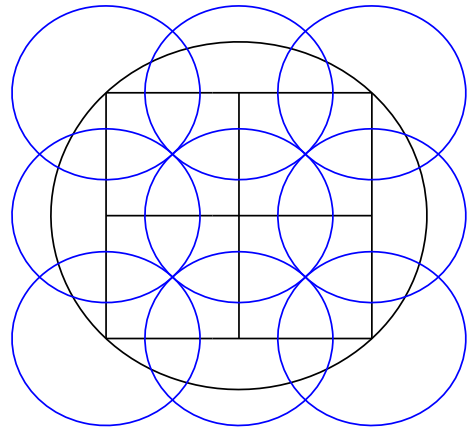
provides a feasible minimiser  $x_{\mathcal{B}}^+$  if the minimum is smaller than  $r(\mathcal{B})^2$ . Moreover, if the minimum of the convex program is larger than  $r(\mathcal{B})^2$ , we know that the ball  $\mathcal{B}$  lies entirely outside of  $\mathcal{D}$  and can be discarded. The convex programming problem can be efficiently solved using standard convex optimization techniques (see Boyd and Vandenberghe 2004).

2.2 Splitting rule

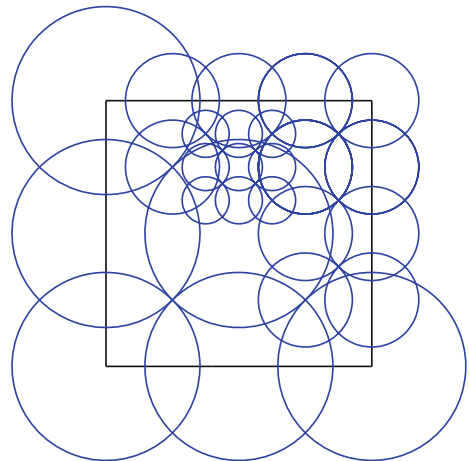
We split a ball  $\mathcal{B} \subset \mathbb{R}^n$  in step 1c of Algorithm 2.1 as follows. Let  $\mathcal{B}$  have centre  $x_{\mathcal{B}}$  and radius  $r(\mathcal{B})$ . Split  $\mathcal{B}$  into  $3^n$  sub-balls of radius  $r(\mathcal{B})/2$  centred at the vertices of a hypercubic tessellation around  $x_{\mathcal{B}}$  of edge length  $r(\mathcal{B})/\sqrt{n}$ . Formally, construct  $3^n$  sub-balls  $\mathcal{B}_1, \dots, \mathcal{B}_{3^n}$  all of radius  $r(\mathcal{B})/2$  centred at

$$x_{\mathcal{B}_i} = x_{\mathcal{B}} + \rho_i^n \left( \frac{-r(\mathcal{B})}{\sqrt{n}}, 0, \frac{r(\mathcal{B})}{\sqrt{n}} \right)$$

**Fig. 1** An illustration of our splitting rule in two dimensions. The *black circle* is split into nine *blue circles* of half radius centred at the vertices of the square tessellation. (Color figure online)



**Fig. 2** An illustration of the coverings of balls at different levels in two dimensions. Note how the configuration and thus the overlap between neighbouring balls is preserved across the different levels



for  $i = 1, \dots, 3^n$ . Here  $\rho_i^n(s_1, s_2, s_3)$  is a vector in  $\mathbb{R}^n$  whose elements are the  $i$ -th permutation of  $s_1, s_2, s_3$  taken  $n$  at a time with repetition. We illustrate this for the case  $n = 2$  in Fig. 1. Note that the choice of centres and radii of the sub-balls ensures that they cover the original ball  $\mathcal{B}$ . Furthermore, this means that at any iteration of Algorithm 2.1 we always have a covering of closed balls of the convex set  $\mathcal{D}$ . One can see from Fig. 1 that neighbouring balls in the covering intersect and thus the minimiser could exist in two of the balls. However, our splitting rule is such that this overlapping configuration is preserved each time the balls are split into sub-balls, creating coverings with exactly the same amount of overlap on many different levels, see Fig. 2. In particular, this means that the minimiser can only exist in at most two balls in each level, i.e. of the same radius.

It is possible to use instead a standard rectangular partitioning strategy (e.g. as used in Jones et al. 1993) where a hyper-rectangle is split into a partition with at most  $2n + 1$  subrectangles. One can then circumscribe balls around the hyper-rectangles and apply Algorithm 2.1 as before. However, while this has the seeming advantage of lowering the number of subregions created at each step of the algorithm, as the hyper-rectangles are not in general regular this creates a large amount of overlap between neighbouring balls making it difficult

to discard uninteresting regions. Nonetheless, it may be possible to adapt such rectangular partitioning strategies to our approach and this is an avenue we are currently exploring.

### 3 Proof of convergence

In this section we will prove that, under suitable assumptions, Algorithm 2.1 converges in a finite number of iterations to within a tolerance  $\varepsilon > 0$  of the global minimum of  $f(x)$ . Our proof is based on the convergence proof of the canonical box-based bound constrained branch and bound algorithm in Balakrishnan et al. (1991). First we state and prove a pair of Lemmata before giving the main convergence theorem.

**Lemma 3.1** *The bounds  $\alpha$  and  $\beta$  given above in (2.5) and (2.6), respectively satisfy*

- (C1)  $\alpha(\mathcal{B}) \leq \min_{x \in \mathcal{B}} f(x) \leq \beta(\mathcal{B}) \quad \forall \mathcal{B} \subset \mathbb{R}^n$
- (C2)  $\forall \varepsilon > 0 \exists \delta > 0 \text{ s.t. } \forall \mathcal{B} \subset \mathbb{R}^n, r(\mathcal{B}) \leq \delta \implies \beta(\mathcal{B}) - \alpha(\mathcal{B}) \leq \varepsilon$

*Proof* (C1) Recall that  $\alpha(\mathcal{B}) = \min_{x \in \mathcal{B}} m_{\mathcal{B}}^-(x)$  and  $\beta(\mathcal{B}) = f(x_{\mathcal{B}}^+)$ . From (2.4) we have that

$$\min_{x \in \mathcal{B}} m_{\mathcal{B}}^-(x) \leq \min_{x \in \mathcal{B}} f(x) \tag{3.1}$$

and clearly

$$\min_{x \in \mathcal{B}} f(x) \leq f(x_{\mathcal{B}}^+).$$

Thus we see that the bounds satisfy condition (C1).

- (C2) Let  $\varepsilon > 0$  be arbitrary. For clarity of exposition define for all  $\mathcal{B} \subset \mathbb{R}^n$ ,

$$x_{\mathcal{B}}^- := \arg \min_{x \in \mathcal{B}} m_{\mathcal{B}}^-(x).$$

Note that  $x_{\mathcal{B}}^-$  may not be unique but this does not matter. Then  $r(\mathcal{B}) \leq \delta$  means that

$$\|x_{\mathcal{B}}^- - x_{\mathcal{B}}\| \leq \delta. \tag{3.2}$$

Consider

$$\begin{aligned} \beta(\mathcal{B}) - \alpha(\mathcal{B}) &= |f(x_{\mathcal{B}}^+) - m_{\mathcal{B}}^-(x_{\mathcal{B}}^-)| \\ &\leq |f(x_{\mathcal{B}}^+) - f(x_{\mathcal{B}})| + \|x_{\mathcal{B}}^- - x_{\mathcal{B}}\| \|g(x_{\mathcal{B}})\| \\ &\quad + \frac{1}{2} \|x_{\mathcal{B}}^- - x_{\mathcal{B}}\|^2 \|H(x_{\mathcal{B}})\| + \frac{LH}{6} \|x_{\mathcal{B}}^- - x_{\mathcal{B}}\|^3 \\ &\leq L \|x_{\mathcal{B}}^+ - x_{\mathcal{B}}\| + \|x_{\mathcal{B}}^- - x_{\mathcal{B}}\| \|g(x_{\mathcal{B}})\| \\ &\quad + \frac{1}{2} \|x_{\mathcal{B}}^- - x_{\mathcal{B}}\|^2 \|H(x_{\mathcal{B}})\| + \frac{LH}{6} \|x_{\mathcal{B}}^- - x_{\mathcal{B}}\|^3 \\ &\leq L\delta + \|g(x_{\mathcal{B}})\|\delta + \frac{1}{2} \|H(x_{\mathcal{B}})\|\delta^2 + \frac{LH}{6} \delta^3 \\ &\leq 2L\delta + \frac{Lg}{2} \delta^2 + \frac{LH}{6} \delta^3 \end{aligned}$$

where the first inequality follows directly from the triangle and Cauchy–Schwarz inequalities, the second from (1.3), the third from (3.2) and the fourth from (1.2). It therefore suffices to choose  $\delta$  sufficiently small such that

$$2L\delta + \frac{Lg}{2} \delta^2 + \frac{LH}{6} \delta^3 \leq \varepsilon$$

so we simply need to pick  $\delta \leq \min\{\varepsilon/6L, \sqrt{2\varepsilon/3Lg}, \sqrt[3]{2\varepsilon/LH}\}$ , i.e. such that each of the three terms on the left of the bounding equation for  $\varepsilon$  is less than or equal to  $\varepsilon/3$ . □

Our second Lemma shows that Algorithm 2.1 eventually creates a ball of arbitrarily small radius.

**Lemma 3.2** For  $k \in \mathbb{N}$

$$\min_{\mathcal{B} \in \mathcal{L}_k} r(\mathcal{B}) \leq \frac{r(\mathcal{B}_0)}{2^{\lceil (\log_3 k)/2 \rceil}}$$

and thus for any  $\delta > 0$  there exists  $K \in \mathbb{N}$  such that

$$\min_{\mathcal{B} \in \mathcal{L}_K} r(\mathcal{B}) \leq \delta.$$

*Proof* Firstly recall that our splitting rule splits each ball into  $3^n$  sub-balls. We start at iteration  $k = 0$  with our initial covering ball  $\mathcal{B}_0$  of radius  $r(\mathcal{B}_0)$ . We split  $\mathcal{B}_0$  into  $3^n$  sub-balls of radius  $r(\mathcal{B}_0)/2$  at iteration  $k = 1$ . Assuming a worst-case scenario, each of these  $3^n$  sub-balls has to be split into  $3^n$  subsub-balls of radius  $r(\mathcal{B}_0)/4$  before we can consider any of the subsub-balls for splitting. Following this argument through inductively, we deduce that for  $k \in \mathbb{N}$  it takes at most

$$k = \sum_{j=1}^m (3^n)^{j-1} \tag{3.3}$$

iterations to reduce the radius of the smallest ball in the covering to less than or equal to

$$\frac{r(\mathcal{B}_0)}{2^m}. \tag{3.4}$$

We can bound (3.3) by

$$k \leq m(3^n)^{m-1} \leq (3^n)^{2m}$$

which when combined with (3.4) gives the required bound. The second part of the Lemma then follows trivially. □

**Theorem 3.1** Algorithm 2.1 converges in a finite number of iterations to within a tolerance  $\varepsilon > 0$  of the global minimum of  $f(x)$  over  $\mathcal{D}$ . Formally, for any  $\varepsilon > 0$  there exists  $M_\varepsilon \in \mathbb{N}$  such that

$$U_{M_\varepsilon} - L_{M_\varepsilon} \leq \varepsilon$$

and  $U_{M_\varepsilon}$  is within a tolerance  $\varepsilon$  of the global minimum of  $f(x)$  over  $\mathcal{D}$ .

*Proof* Let  $\mathcal{B}_k := \arg \min_{\mathcal{B} \in \mathcal{L}_k} r(\mathcal{B})$  and let  $\mathcal{A}_k \in \mathcal{L}_{M_k}$  be the ball which for some  $M_k < k$  we split to obtain  $\mathcal{B}_k$ . Note that  $\mathcal{B}_k$  may not be unique but this does not matter. Let  $\varepsilon > 0$  be arbitrary. Then there exists  $\delta > 0$  such that for any  $\mathcal{B} \subset \mathbb{R}^n$

$$r(\mathcal{B}) \leq 2\delta \implies \beta(\mathcal{B}) - \alpha(\mathcal{B}) \leq \varepsilon \tag{3.5}$$

by condition (C2) of Lemma 3.1. Choose  $K \in \mathbb{N}$  sufficiently large such that

$$r(\mathcal{B}_K) \leq \delta$$



which is possible by Lemma 3.2. Then  $\mathcal{A}_K$  must have  $r(\mathcal{A}_K) \leq 2\delta$  as we split it to obtain  $\mathcal{B}_k$  and thus from (3.5) we have that

$$\beta(\mathcal{A}_K) - \alpha(\mathcal{A}_K) \leq \varepsilon. \tag{3.6}$$

Now, as  $\mathcal{A}_K$  was split at iteration  $M_K$ , it must have satisfied  $\alpha(\mathcal{A}_K) = L_{M_K}$ . Hence we get that

$$U_{M_K} - L_{M_K} \leq \beta(\mathcal{A}_K) - L_{M_K} \leq \varepsilon \tag{3.7}$$

since  $U_{M_K} \leq \beta(\mathcal{A}_K)$  by definition and using (3.6). We therefore have an upper bound  $M_K$  on the number of branch and bound iterations.

It remains to show that  $U_{M_K}$  is within a tolerance  $\varepsilon$  of the global minimum of  $f(x)$  over  $\mathcal{D}$ . Assume condition (C1) of Lemma 3.1 holds and suppose that the global minimum  $l^*$  of  $f(x)$  over  $\mathcal{D}$  is attained at  $x^* \in \mathcal{D}$ . First, we show that  $x^*$  is contained in a ball in  $\mathcal{L}_{M_K}$ . To see this, observe that for all  $k \in \mathbb{N}$ ,  $\mathcal{L}_k$  is a partition of the bounding ball  $\mathcal{B}_0$  with balls which cannot possibly contain  $x^*$  removed, that is to say balls  $\mathcal{B}$  which have lower bound

$$\alpha(\mathcal{B}) > U_k$$

i.e.  $\alpha(\mathcal{B}) > f(x_{\mathcal{B}}^+)$  for a feasible point  $x_{\mathcal{B}}^+ \in \mathcal{D}$ . As  $x^*$  is contained in a ball in  $\mathcal{L}_{M_K}$  it follows that  $L_{M_K} \leq l^*$  and thus

$$U_{M_K} - l^* \leq U_{M_K} - L_{M_K} \leq \varepsilon$$

by (3.7). □

### 4 Computing the lower bound

We have mentioned earlier in Sect. 2 that we use lower bounds based on globally minimising the cubic bounding function  $m_{\mathcal{B}}^-(x)$  over balls  $\mathcal{B} \subset \mathbb{R}^n$ , in a similar vein to [Nesterov and Polyak \(2006\)](#). In this subsection we will show how we can efficiently globally minimise  $m_{\mathcal{B}}^-(x)$  over any closed ball  $\mathcal{B}$  centred at  $x_{\mathcal{B}}$ , i.e.

$$\begin{aligned} \text{minimise } m_{\mathcal{B}}^-(x) &= f(x_{\mathcal{B}}) + (x - x_{\mathcal{B}})^T g(x_{\mathcal{B}}) + \frac{1}{2}(x - x_{\mathcal{B}})^T H(x_{\mathcal{B}})(x - x_{\mathcal{B}}) \\ &\quad - \frac{L_H(\mathcal{B})}{6} \|x - x_{\mathcal{B}}\|^3 \\ \text{subject to } \|x - x_{\mathcal{B}}\| &\leq \Delta \end{aligned}$$

for some  $\Delta > 0$ . For clarity of exposition, we rewrite the above minimisation problem in the equivalent form

$$\begin{aligned} \text{minimise } m^-(x) &:= f + x^T g + \frac{1}{2}x^T Hx - \frac{\sigma}{3}\|x\|^3 \\ \text{subject to } \|x\| &\leq \Delta \end{aligned}$$

where  $\sigma := L_H(\mathcal{B})/2$ , we have shifted  $x$  by  $x_{\mathcal{B}}$ , and we have dropped the explicit dependence on  $\mathcal{B}$  from the notation. It is clear that the global minimum of the above problem will occur either on the boundary or in the interior of the  $\Delta$ -ball. We solve for these two cases in turn, starting with the case where the minimum lies on the boundary.

4.1 Finding a minimiser on the  $\Delta$ -ball boundary

For the bounding function  $m^-(x)$  we have the following global optimality result (cf. Theorem 7.2.1 in Conn et al. 2000 and Theorem 3.1 in Cartis et al. 2009).

**Theorem 4.1** Any  $x^*$  is a global minimiser of  $m^-(x)$  over  $\mathbb{R}^n$  subject to  $\|x\| = \Delta$  if and only if it satisfies the system of equations

$$(H + (\lambda^* - \sigma \Delta)I)x^* = -g \tag{4.1}$$

where  $H + (\lambda^* - \sigma \Delta)I$  is positive semidefinite for some Lagrange multiplier  $\lambda^*$  and  $\|x^*\| = \Delta$ . If  $H + (\lambda^* - \sigma \Delta)I$  is positive definite,  $x^*$  is unique.

*Proof* First we rewrite the constraint  $\|x\| = \Delta$  as  $\frac{1}{2}\|x\|^2 - \frac{1}{2}\Delta^2 = 0$ . Now, let  $x^*$  be a global minimiser of  $m^-(x)$  over  $\mathbb{R}^n$  subject to the constraint. We have from the first order necessary optimality conditions (see Section 3.2.2 of Conn et al. 2000) that  $x^*$  satisfies

$$(H + (\lambda^* - \sigma \|x^*\|)I)x^* = -g. \tag{4.2}$$

where  $\lambda^*$  is the corresponding Lagrange multiplier. We have by assumption that  $\|x^*\| = \Delta$  and substituting this into (4.2) gives the required system (4.1). Now, suppose  $u^*$  is a feasible point, i.e. that  $\|u^*\| = \Delta$ . We have that

$$\begin{aligned} m^-(u^*) - m^-(x^*) &= g^T(u^* - x^*) + \frac{1}{2}(u^*)^T H u^* - \frac{1}{2}(x^*)^T H x^* + \frac{\sigma}{3} (\|u^*\|^3 - \|x^*\|^3) \\ &= g^T(u^* - x^*) + \frac{1}{2}(u^*)^T H u^* - \frac{1}{2}(x^*)^T H x^* \end{aligned} \tag{4.3}$$

where the last equality follows from the fact that  $\|x^*\| = \|u^*\| = \Delta$ . But (4.2) gives that

$$g^T(u^* - x^*) = (x^* - u^*)^T H x^* + (\lambda^* - \sigma \Delta)(x^* - u^*)^T x^*. \tag{4.4}$$

Also, the fact that  $\|x^*\| = \|u^*\| = \Delta$  implies that

$$(x^* - u^*)^T x^* = \frac{1}{2}(x^*)^T x^* + \frac{1}{2}(u^*)^T u^* - (u^*)^T x^* = \frac{1}{2}(u^* - x^*)^T (u^* - x^*). \tag{4.5}$$

Combining (4.3) with (4.4) and (4.5), we find that

$$\begin{aligned} m^-(u^*) - m^-(x^*) &= \frac{1}{2}(\lambda^* - \sigma \Delta)(u^* - x^*)^T (u^* - x^*) + \frac{1}{2}(u^*)^T H u^* \\ &\quad - \frac{1}{2}(x^*)^T H x^* + (x^*)^T H x^* - (u^*)^T H x^* \\ &= \frac{1}{2}(u^* - x^*)^T (H + (\lambda^* - \sigma \Delta)I)(u^* - x^*). \end{aligned} \tag{4.6}$$

We also have from the second order necessary optimality conditions (see Section 3.2.2 of Conn et al. 2000) that

$$H + (\lambda^* - \sigma \|x^*\|)I - \frac{\sigma}{\|x^*\|} x^*(x^*)^T$$

is positive semidefinite on the null-space of the constraint gradient  $x^*$ , i.e. that

$$v^T \left( H + (\lambda^* - \sigma \Delta)I - \frac{\sigma}{\Delta} x^*(x^*)^T \right) v \geq 0 \tag{4.7}$$

for all  $v$  for which  $v^T x^* = 0$ , where we have used the fact that  $\|x^*\| = \Delta$ . In this case it immediately follows from (4.7) that

$$v^T (H + (\lambda^* - \sigma \Delta)I) v \geq 0$$

for all  $v$  for which  $v^T x^* = 0$ . It thus remains to consider vectors  $v$  for which  $v^T x^* \neq 0$ . Since  $v$  and  $x^*$  are not orthogonal, the line  $x^* + \alpha v$  intersects the constraint  $\|x\| = \Delta$  at two points,  $x^*$  and  $u^*$ . Let  $v^* = u^* - x^*$  and note that  $v^*$  is parallel to  $v$ . As  $x^*$  is a global minimiser we have that  $m^-(u^*) \geq m^-(x^*)$  and thus we have from (4.6) that

$$\begin{aligned} 0 \leq m^-(u^*) - m^-(x^*) &= \frac{1}{2}(u^* - x^*)^T (H + (\lambda^* - \sigma \Delta)I)(u^* - x^*) \\ &= \frac{1}{2}(v^*)^T (H + (\lambda^* - \sigma \Delta)I)v^* \end{aligned} \tag{4.8}$$

from which we deduce that

$$v^T (H + (\lambda^* - \sigma \Delta)I) v \geq 0$$

for all  $v$  for which  $v^T x^* \neq 0$ . In summary, we have shown that

$$v^T (H + (\lambda^* - \sigma \Delta)I) v \geq 0$$

for any vector, which is the same as saying that  $H + (\lambda^* - \sigma \Delta)I$  must be positive semidefinite. Conversely, if  $H + (\lambda^* - \sigma \Delta)I$  is positive definite,  $\frac{1}{2}(u^* - x^*)^T (H + (\lambda^* - \sigma \Delta)I)(u^* - x^*) > 0$  for any  $u^* \neq x^*$  and therefore (4.8) shows that  $m^-(u^*) > m^-(x^*)$  whenever  $u^*$  is feasible. Thus  $x^*$  is the unique global minimiser.  $\square$

The global minimiser can be efficiently found by applying a safeguarded version of Newton's method as detailed in Section 2.1 of Gould et al. (2010) to the scalar equation

$$\begin{aligned} \|x(\lambda)\| &= \Delta, \\ \text{where } (H + (\lambda - \sigma \Delta)I)x(\lambda) &= -g, \end{aligned} \tag{4.9}$$

and this is the approach we take (see Fig. 3).

Note that since  $H$  is of low dimension it is more efficient to find the spectral decomposition  $H = Q\Lambda Q^T$ , and then to solve an equivalent problem to (4.9) in transformed variables  $y = Q^T x$  for which the Hessian,  $\Lambda$ , is diagonal.

#### 4.2 Finding a minimiser in the $\Delta$ -ball interior

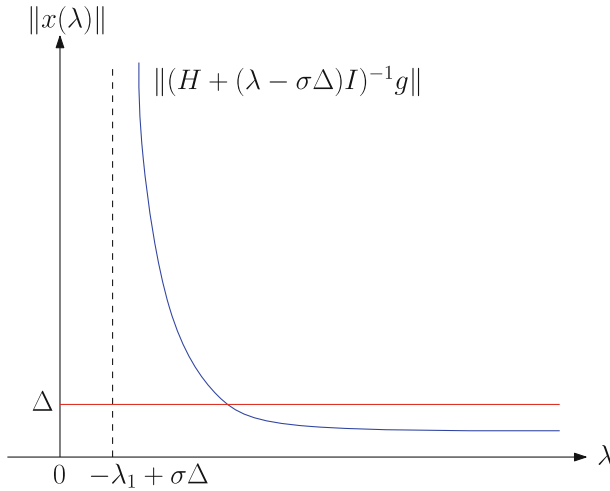
If, by contrast, the solution we seek lies in the interior of  $\mathcal{B}$ , we have the following result:

**Theorem 4.2** *The lower bounding function  $m^-(x)$  can only have finite global minimisers if  $H$  is positive semidefinite. In this case, any  $x^*$  is a global minimiser of  $m^-(x)$  over  $\mathbb{R}^n$  if and only if it satisfies the system of equations*

$$(H + \omega^* I)x^* = -g \tag{4.10}$$

where  $\omega^* = -\sigma \|x^*\|$  and  $H + \omega^* I$  is positive semidefinite.

*Proof* For the first part, suppose  $H$  is not positive semidefinite and consider  $x = \alpha u$  for any eigenvector  $u$  of  $H$  corresponding to a negative eigenvalue. Then clearly  $m^-(x) \rightarrow -\infty$  as  $\alpha \rightarrow \infty$  and so the function  $m^-(x)$  is unbounded below. The second part of the proof is analogous to the proof of the first part of Theorem 3.1 in Cartis et al. (2009).  $\square$



**Fig. 3** Solutions to the system (4.1) are the intersections of the two curves

Note that in this case  $\omega^* = -\sigma \|x^*\|$  and so there can only be a solution for  $\omega \leq 0$ . Assuming  $H$  is positive semidefinite, let  $\lambda_1$  denote the smallest eigenvalue of  $H$  and note that if  $\lambda_1 = 0$  there can only be a trivial solution to the system (4.10) when  $x = 0$  and  $g = 0$ . When  $\lambda_1 > 0$  there will be at most two possible solutions to the system (4.10) which, once again, can be found using Newton's method with suitable starting points (i.e. Algorithm 6.1 in [Cartis et al. 2009](#)). Numerical results suggest that in this case the solution closest to zero is always the best local minimiser in the  $\Delta$ -ball interior, and this is indeed the case as we show in Theorem 4.3 below. Figure 4 illustrates this typical case when there are two possible solutions. (Note that there may be no solutions and an example of this is the case where the straight line lies under the curve in Fig. 4.) We have the following theorem (based on Theorem 3 from [Griewank 1981](#)) which shows that whenever  $m^-(x)$  has a finite global minimiser over  $\mathbb{R}^n$ , the global minimiser is unique.

**Theorem 4.3** *For any pair of potential finite global minimisers  $u^*, v^*$  of the lower bounding function  $m^-(x)$  over  $\mathbb{R}^n$  given by Theorem 4.2 with  $\|u^*\| \leq \|v^*\|$ , the corresponding lower bounding function values satisfy*

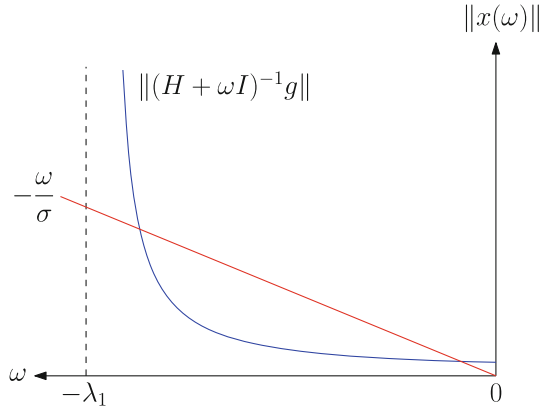
$$m^-(v^*) - m^-(u^*) \geq \frac{\sigma}{6} (\|v^*\| - \|u^*\|)^3 \geq 0.$$

*In particular this means that when there are two possible solutions to the system (4.10) (as in Fig. 4), the solution  $x^*$  with smaller norm  $\|x^*\|$  is the global minimiser of  $m^-(x)$  over  $\mathbb{R}^n$ .*

*Proof* First of all consider a general potential finite global minimiser  $x^*$  of  $m^-(x)$ . As it is a potential finite global minimiser it must satisfy (4.10). Multiplying (4.10) on the left by  $(x^*)^T$  and dividing by two gives

$$\frac{1}{2}(x^*)^T H x^* = -\frac{1}{2}g^T x^* - \frac{1}{2}\omega^*(x^*)^T x^* \tag{4.11}$$

**Fig. 4** Solutions to the system (4.10) for  $m^-(x)$  are the intersections of the two curves



where  $\omega^* = -\sigma \|x^*\|$ . From the definition of  $m^-(x)$  we have that

$$\begin{aligned}
 m^-(x^*) &= f + g^T x^* + \frac{1}{2}(x^*)^T H x^* - \frac{\sigma}{3} \|x^*\|^3 \\
 &= f + g^T x^* - \frac{1}{2} g^T x^* - \frac{1}{2} \omega^* (x^*)^T x^* - \frac{\sigma}{3} \|x^*\|^3 \quad \text{by (4.11)} \\
 &= f + \frac{1}{2} g^T x^* + \frac{\sigma}{2} \|x^*\| \|x^*\|^2 - \frac{\sigma}{3} \|x^*\|^3 \\
 &= f + \frac{1}{2} g^T x^* + \frac{\sigma}{6} \|x^*\|^3 \tag{4.12}
 \end{aligned}$$

where the third equality follows from the fact that  $\omega^* = -\sigma \|x^*\|$  and  $(x^*)^T x^* = \|x^*\|^2$ . Now, assume  $u^*, v^*$  are potential finite global minimisers of  $m^-(x)$ . We have from Theorem 4.2 that

$$(H + \alpha^* I)u^* = -g = (H + \beta^* I)v^* \tag{4.13}$$

where

$$\alpha^* = -\sigma \|u^*\| \geq -\sigma \|v^*\| = \beta^*. \tag{4.14}$$

Multiplying (4.13) by  $(v^*)^T$  and  $(u^*)^T$  we obtain

$$\begin{aligned}
 -g^T v^* &= (v^*)^T H u^* + \alpha^* (v^*)^T u^* \\
 -g^T u^* &= (u^*)^T H v^* + \beta^* (u^*)^T v^*
 \end{aligned}$$

so that (since  $H$  is symmetric)

$$g^T (v^* - u^*) = (\beta^* - \alpha^*) (u^*)^T v^* \geq (\beta^* - \alpha^*) \|u^*\| \|v^*\| = -\sigma (\|v^*\| - \|u^*\|) \|u^*\| \|v^*\| \tag{4.15}$$

where we have used the Cauchy–Schwarz inequality (multiplied by  $\beta^* - \alpha^*$  which is negative by (4.14)). We now have from (4.12) that

$$\begin{aligned}
 m^-(v^*) - m^-(u^*) &= \frac{1}{2}g^T(v^* - u^*) + \frac{\sigma}{6}(\|v^*\|^3 - \|u^*\|^3) \\
 &\geq -\frac{\sigma}{2}(\|v^*\| - \|u^*\|)\|u^*\|\|v^*\| + \frac{\sigma}{6}(\|v^*\|^3 - \|u^*\|^3) \quad \text{by (4.15)} \\
 &= \frac{\sigma}{6}(-3\|v^*\|^2\|u^*\| + 3\|u^*\|^2\|v^*\| + \|v^*\|^3 - \|u^*\|^3) \\
 &= \frac{\sigma}{6}(\|v^*\| - \|u^*\|)^3 \geq 0
 \end{aligned}$$

which completes the proof. □

### 5 A faster heuristic algorithm

It is not always necessary or desirable to find the global minimum to a high degree of accuracy. With this in mind we present a faster heuristic version of the Lipschitz based branch and bound algorithm which has no theoretical convergence guarantees but still exhibits reasonable performance.

The main drawback with regards to performance of the existing algorithm is the splitting of each ball into  $3^n$  sub-balls since the number of sub-balls grows rapidly as  $n$  (the dimension of the problem) increases. Rather than using overlapping balls, for the heuristic version of the algorithm we split each ball into a dense lattice of non-overlapping sub-balls. The maximum number of same-size balls one can pack around a central ball without overlap is given by the kissing number  $\kappa$  (Conway and Sloane 1999). Optimal kissing numbers and the corresponding lattices we use which give rise to them are known up to 9 dimensions (see Conway and Sloane 1999, for details). Running the algorithm with this splitting rule means that each ball is only split into  $\kappa + 1$  sub-balls, considerably less than  $3^n$  (cf. Table 1). The disadvantage is that it leaves holes in the domain we are trying to optimize over and so convergence to the global optimum is not guaranteed. However, by running a local solver from the global minimum proposed by the algorithm, we will always find a local minimum which is often a good candidate for being the global optimum (and which can be used as an upper bound in the original slower version of Algorithm 2.1 if desired). Figure 5 illustrates this heuristic splitting rule using the hexagonal lattice which is known to be the optimal lattice in two dimensions.

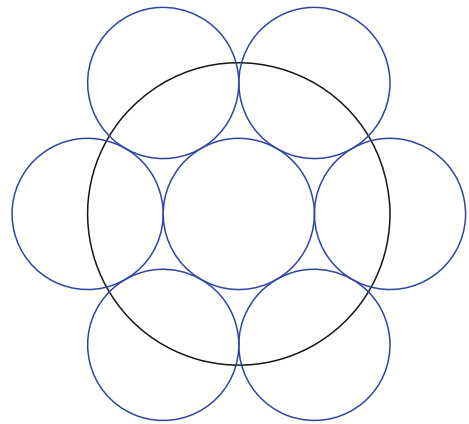
### 6 Application to radial basis functions

In recent years radial basis function (RBF) interpolation has become a popular and well established approximation method for real valued functions, particularly in higher dimensions (see Wendland 2005). This approach is also known as intrinsic random function Kriging in the geostatistical literature (as proposed by Matheron, see Chilès and Delfiner 1999) and Gaussian process regression in Bayesian statistics (an early reference is O’Hagan 1978). Under these synonyms RBF interpolation has been applied to the design and analysis of computer experiments (Santner et al. 2003), machine learning (Rasmussen and Williams 2006) and engineering design (Forrester et al. 2008) to name but a few. The RBF interpolant can also be viewed as a neural network (see Chapter 5 of Bishop 1996). One of the main advantages of

**Table 1** The optimal kissing number plus one ( $\kappa + 1$ ) compared against  $3^n$  for the first 9 dimensions

$n$	$\kappa + 1$	$3^n$
1	3	3
2	7	9
3	13	27
4	25	81
5	41	243
6	73	729
7	127	2187
8	241	6561
9	273	19683

**Fig. 5** An illustration of our heuristic splitting rule in two dimensions. The black circle is split into seven blue circles arranged in a hexagonal lattice. (Color figure online)



using RBFs to approximate an underlying real valued function is that one can cheaply obtain derivatives of the RBF of any order. In particular, this allows us to easily obtain the Hessian Lipschitz constant over any ball in  $\mathbb{R}^n$  and thus apply our Lipschitz based branch and bound algorithm, Algorithm 2.1 from Sect. 2. Note that this is an entirely new approach, distinct from the canonical Lipschitz branch and bound algorithm (as described in e.g. Section 5.3 of Pardalos et al. 1995). With this in mind we will apply the proposed branch and bound algorithm to the special case where the objective function  $f(x)$  is a radial basis function approximation to some real valued function. Of course, the real motivation behind using RBFs is to use the global minimizer of the RBF approximation to infer something about the original problem, i.e. the global minimum of the underlying real valued function that is approximated (see Farmer et al. 2010, for an example application).

### 6.1 Introduction to radial basis functions

First of all, let us give a brief introduction to radial basis function interpolation. We begin by defining the weighted  $\ell_2$ -norm  $\|\cdot\|_w := \|W\cdot\|_2$  with diagonal weight matrix  $W$ , and suppose we have  $N$  samples  $y = (y_1, \dots, y_N)^T$  of some real valued function we wish to approximate at the corresponding sample points  $x_1, \dots, x_N \in \mathcal{D}$ . An RBF approximation  $f : \mathcal{D} \subset \mathbb{R}^n \rightarrow \mathbb{R}$  is then constructed as a linear combination of basis functions  $\varphi(\cdot)$

composed with a weighted  $\ell_2$ -norm, together with an additional polynomial term to guarantee uniqueness:

$$f(x) = \sum_{k=1}^M \mu_k \pi_k(x) + \sum_{j=1}^N \lambda_j \varphi(\|x - x_j\|_W) \tag{6.1}$$

where  $\{\pi_k(x)\}_{k=1}^M$  is a basis for  $\Pi_d^n$ , the space of polynomials in  $\mathbb{R}^n$  of degree less than  $d$ , with the notation  $\Pi_0^n = \{0\}$ . Typical choices of the basis function  $\varphi(\cdot)$  are

the spline	$\varphi(r) = \begin{cases} r^p & \text{if } p \text{ is odd} \\ r^p \log r & \text{if } p \text{ is even} \end{cases}$	;	
the multiquadric	$\varphi(r) = (r^2 + \gamma^2)^\beta$	$\beta > 0, \beta \notin \mathbb{N};$	(6.2)
the inverse multiquadric	$\varphi(r) = (r^2 + \gamma^2)^{-\beta}$	$\beta > 0;$	and
the Gaussian	$\varphi(r) = \exp(-\gamma^2 r^2),$		

where  $\gamma$  is a nonzero constant referred to as the shape parameter (see, for example, Chapters 6, 7, 8 of [Wendland 2005](#)). Note that spline type RBFs are not  $C^2$  if  $p = 1$  nor  $C^3$  unless  $p$  is greater than two. As we use a weighted norm, we often let  $\gamma = 1$  for the Gaussian basis function. The coefficients  $\mu_k, \lambda_j$  are determined by solving the linear interpolation system

$$y_i = \sum_{k=1}^M \mu_k \pi_k(x_i) + \sum_{j=1}^N \lambda_j \varphi(\|x_i - x_j\|_W), \quad i = 1, \dots, N$$

along with the additional conditions

$$\sum_{j=1}^N \lambda_j \pi_k(x_j) = 0, \quad k = 1, \dots, M$$

which complete the system and ensure that polynomials of degree less than  $d$  are interpolated exactly. In matrix form this gives the non-singular (provided  $\{x_i\}_{i=1}^N$  is a  $\Pi_d^n$ -unisolvent set, see [Wendland 2005](#)) symmetric saddle-point system

$$\begin{pmatrix} R & P \\ P^T & 0 \end{pmatrix} \begin{pmatrix} \lambda \\ \mu \end{pmatrix} = \begin{pmatrix} y \\ 0 \end{pmatrix} \tag{6.3}$$

where  $P_{i,j} = \pi_j(x_i)$  is a polynomial basis matrix and  $R$  is the correlation matrix given by

$$R_{i,j} = \varphi(\|x_i - x_j\|_W).$$

There are two main approaches often used in the literature to find the weights in the weight matrix  $W$ , which we will now assume to be diagonal. The first approach consists of choosing  $W$  to maximise the likelihood of the observed data  $x_1, \dots, x_N$  and leads one to choose  $W$  to be the maximiser of the log-likelihood function (see [Busby et al. 2007](#))

$$\ell(W) = -\frac{1}{2}(N \log \sigma^2 + \log \det(R))$$

which we optimise using a general purpose global optimisation algorithm ([DIRECT](#), [Jones et al. 1993](#)). The second approach is to use leave-one-out cross-validation (see [Rippa 1999](#)) and leads one to choose  $W$  to minimise the  $\ell_2$ -norm of the cross-validation error  $\varepsilon(W) \in \mathbb{R}^N$ .



The  $k$ -th element of the cross-validation error  $\varepsilon(W)$  is the error at the validation point  $x_k$ , given by

$$\varepsilon_k(W) = \frac{\lambda_k}{A_{k,k}^{-1}}$$

where  $A_{k,k}^{-1}$  is the  $k$ -th diagonal element of the inverse of the interpolation matrix  $A = \begin{pmatrix} R & P \\ P^T & 0 \end{pmatrix}$ .

### 6.2 Calculating the Lipschitz constant

We now proceed to calculating a suitable Hessian Lipschitz constant for our proposed branch and bound algorithm in the special case where the objective function  $f(x)$  is a radial basis function approximation (6.1). We start by introducing tensors. A *third order tensor*  $T$  is a generalisation of a matrix to three indices, that is to say a 3-dimensional array. As with matrices  $T_{i,j,k}$  denotes the  $i, j, k$ -th component (i.e. element in the array) of the tensor  $T$ . Recall that the matrix Frobenius norm can be defined as

$$\|A\|_F^2 := \sum_{i=1}^m \sum_{j=1}^n A_{i,j}^2$$

and this can be extended to a third order tensor  $T$  as

$$\|T\|_F^2 := \sum_{i=1}^m \sum_{j=1}^n \sum_{k=1}^o T_{i,j,k}^2$$

Similarly, we can define the induced  $\ell_2$ -norm for tensors by

$$\|T\|_2 := \max_{\|x\|_2=1} \|Tx\|_2$$

where  $\|Tx\|_2$  denotes the usual induced matrix norm. We are now in a position to prove the following Lemma.

**Lemma 6.1** *Let  $T$  be a third order tensor. Then  $\|T\|_2 \leq \|T\|_F$ .*

*Proof* We have that

$$\begin{aligned} \|T\|_2^2 &= \max_{\|x\|_2=1} \|Tx\|_2^2 \leq \max_{\|x\|_2=1} \|Tx\|_F^2 \quad \text{as } \|A\|_2 \leq \|A\|_F \text{ for matrices} \\ &= \max_{\|x\|_2=1} \sum_{i=1}^m \sum_{j=1}^n \left( \sum_{k=1}^o T_{i,j,k} x_k \right)^2 \\ &= \max_{\|x\|_2=1} \sum_{i=1}^m \sum_{j=1}^n \left( \sum_{k=1}^o (a_{i,j})_k x_k \right)^2 \quad \text{where the vector } a_{i,j} \text{ is such that } (a_{i,j})_k = T_{i,j,k} \\ &\leq \max_{\|x\|_2=1} \sum_{i=1}^m \sum_{j=1}^n \|a_{i,j}\|_2^2 \|x\|_2^2 \quad \text{by the Cauchy–Schwarz inequality} \\ &= \sum_{i=1}^m \sum_{j=1}^n \|a_{i,j}\|_2^2 = \sum_{i=1}^m \sum_{j=1}^n \sum_{k=1}^o T_{i,j,k}^2 = \|T\|_F^2 \end{aligned}$$

□

Let  $T(x) := \nabla_{xxx} f(x)$  denote the third order derivative tensor of the RBF approximation  $f(x)$  to some real valued function. We have from Taylor's theorem that for any  $x, y \in \mathcal{B}$

$$\|H(x) - H(y)\|_2 \leq \left\| \int_0^1 T(y + \tau(x - y))(x - y) d\tau \right\|_2 \leq \max_{0 \leq \tau \leq 1} \|T(y + \tau(x - y))\|_2 \|x - y\|_2$$

where  $\|T(\cdot)\|_2$  denotes the tensor  $\ell_2$ -norm defined above. Thus the Hessian  $H(x) := \nabla_{xx} f(x)$  is Lipschitz continuous on a ball  $\mathcal{B} \subset \mathbb{R}^n$  if there exists a  $\ell_2$ -norm Lipschitz constant  $L_H(\mathcal{B}) > 0$  such that for all  $x \in \mathcal{B}$

$$\|T(x)\|_2 \leq L_H(\mathcal{B}).$$

It suffices to find an upper bound  $\tau(\mathcal{B})$  on  $T(x)$  over  $\mathcal{B}$  and we can then use Lemma 6.1 to calculate an upper bound  $L_H(\mathcal{B})$  on the optimal Hessian Lipschitz constant as

$$\|T(x)\|_2 \leq \|\tau(\mathcal{B})\|_2 \leq \|\tau(\mathcal{B})\|_F = \left( \sum_{i=1}^m \sum_{j=1}^n \sum_{k=1}^o \tau_{i,j,k}^2 \right)^{1/2} = L_H(\mathcal{B}).$$

Thus it remains to determine the upper bound for the RBF approximation  $f(x)$ . The approximation  $f(x)$  has the form (6.1)

$$f(x) = \sum_{k=1}^M \mu_k \pi_k(x) + \sum_{j=1}^N \lambda_j \varphi(\|x - x_j\|_w)$$

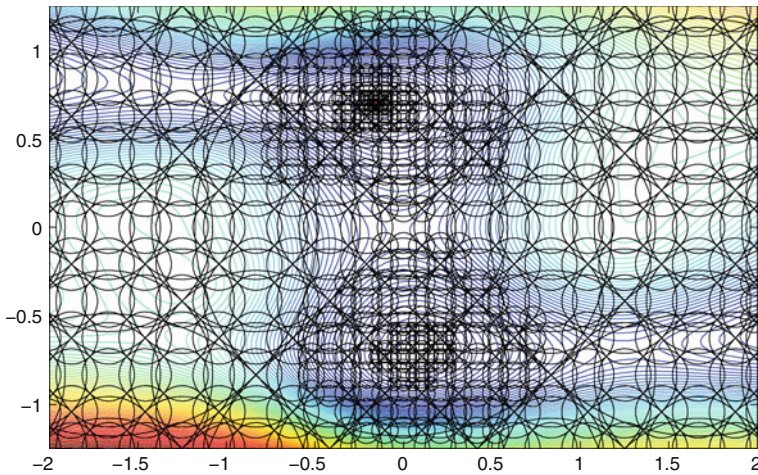
with associated third order derivative tensor  $T(x)$  given by

$$T(x) = \sum_{k=1}^M \mu_k \nabla_{xxx} \pi_k(x) + \sum_{j=1}^N \lambda_j \nabla_{xxx} \varphi(\|x - x_j\|_w).$$

To calculate the upper bound  $\tau(\mathcal{B})$ , it therefore suffices to calculate upper and lower bounds on the tensors  $\nabla_{xxx} \pi_k(x)$  and  $\nabla_{xxx} \varphi(\|x - x_j\|_w)$  over  $\mathcal{B}$  depending on the signs of the coefficients  $\lambda_j, \mu_k$ . For example, for the cubic spline RBF  $\varphi(r) = r^3$  we have  $\nabla_{xxx} \pi_k(x) = 0$  as the polynomial term is linear and

$$\begin{aligned} & (\nabla_{xxx} \varphi(\|x - x_j\|_w))_{a,b,c} \\ &= \begin{cases} \frac{-3w_a^6(x_a - x_{j_a})^3}{\|x - x_j\|_w^3} + \frac{9w_a^2 w_a^2 (x_a - x_{j_a})}{\|x - x_j\|_w} & \text{if } a = b = c \\ \frac{-3w_c^2(x_c - x_{j_c})w_a^4(x_a - x_{j_a})^2}{\|x - x_j\|_w^3} + \frac{3w_a^2 w_c^2(x_c - x_{j_c})}{\|x - x_j\|_w} & \text{if e.g. } a = b \neq c \\ \frac{-3w_a^2(x_a - x_{j_a})w_b^2(x_b - x_{j_b})w_c^2(x_c - x_{j_c})}{\|x - x_j\|_w^3} & \text{otherwise.} \end{cases} \end{aligned}$$

It is trivial to find upper and lower bounds on  $w_a^2(x_a - x_{j_a})/\|x - x_j\|_w$  over the smallest box containing  $\mathcal{B}$  which we can substitute into the above to obtain bounds on  $\nabla_{xxx} \varphi(\|x - x_j\|_w)$ . A similar approach can be used for other radial basis functions (6.2) with interval arithmetic techniques for higher order polynomial terms.



**Fig. 6** Contours of the RBF approximation  $f(x, y)$  to the camel function. The *black circles* denote the overlapping closed balls  $\mathcal{B}$  used by the branch and bound algorithm. Note that they cluster at the global minimum of  $f(x, y)$  which is denoted by a *red circle*. (Color figure online)

### 6.3 Numerical examples

To begin with let us look at an illustrative two dimensional example. Consider the problem of finding the global minimum of a cubic spline RBF approximation  $f(x, y)$  to the Dixon–Szegő six hump camel back function at thirty scattered points in  $[-2, 2] \times [-1.25, 1.25]$ . We will use Algorithm 2.1 with overlapping balls and local Lipschitz constants from Sect. 2. The optimal solution as found in 220 iterations of Step 1 of the algorithm with a tolerance of  $4 \times 10^{-6}$  is shown in Fig. 6. This took about 20 s of cpu time for a Matlab implementation on an AMD Phenom II X4 955 processor machine and used 3, 014 evaluations of  $f$ .

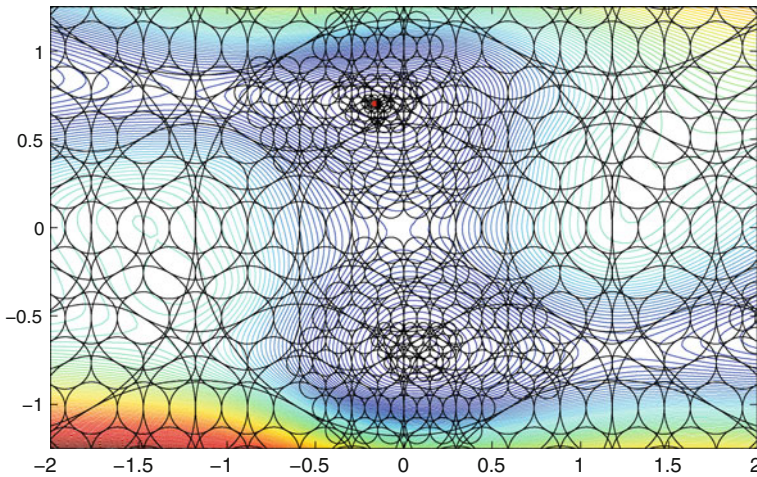
Compare this with the heuristic version using a lattice of balls from Sect. 5. In this case, the optimal solution as found in 134 iterations of Step 1 of the algorithm with a tolerance of  $6 \times 10^{-6}$  is shown in Fig. 7. This took about 10 s of cpu time and used 1, 686 evaluations of  $f$ , considerably less. Notice how the alternate splitting rule is more efficient at filling the available space (albeit with gaps).

For a more interesting example, we will now compare the performance of the two branch and bound algorithm variants for dimensions from 2 to 5. In addition, we will also compare our algorithms to the canonical Lipschitz branch and bound algorithm (see Section 5.3 of Pardalos et al. 1995). This is simply the canonical branch and bound algorithm (see e.g. Horst 1986) with the lower bound

$$\alpha(\mathcal{B}) = f(x_{\mathcal{B}}) - L(\mathcal{B}) \max_{x \in \mathcal{B}} \|x - x_{\mathcal{B}}\| \tag{6.4}$$

where  $x_{\mathcal{B}}$  is the midpoint of  $\mathcal{B}$  and  $L(\mathcal{B})$  an upper bound on the optimal Lipschitz constant over  $\mathcal{B}$ , calculated by bounding the norm of the gradient over  $\mathcal{B}$  similarly to how we calculate  $L_H(\mathcal{B})$  in Sect. 6.2. For this example, the objective function  $f : [-4, 4]^n \subset \mathbb{R}^n \rightarrow \mathbb{R}$  will be a cubic spline radial basis function approximation to the sum of sine functions given by

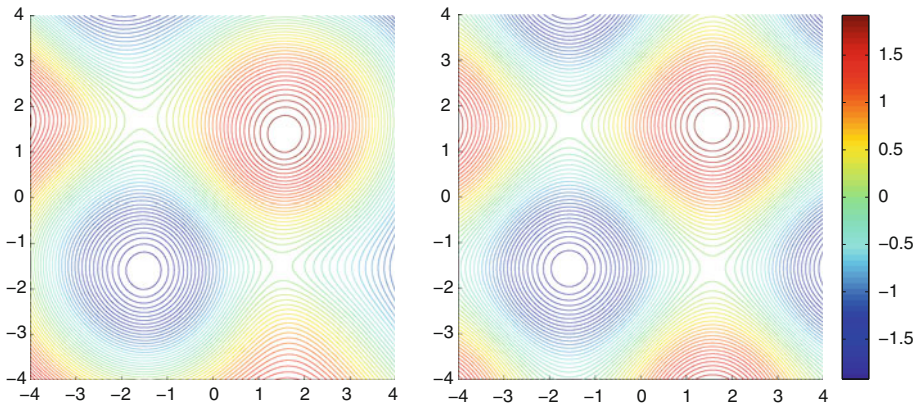
$$s(x) = \sum_{k=1}^n \sin x_k$$



**Fig. 7** Contours of the RBF approximation  $f(x, y)$  to the camel function. The black circles denote the hexagonal lattice of balls  $\mathcal{B}$  used by the branch and bound algorithm. Note that they cluster at the global minimum of  $f(x, y)$  which is denoted by a red circle. (Color figure online)

at  $10n$  maximin Latin hypercube sample points in  $[-4, 4]^n$ . The approximation typically has a number of local minima with one global minimum as one can see in Fig. 8 for dimension  $n = 2$ . Table 2 shows the run time in seconds of a Matlab implementation of each branch and bound algorithm for dimensions  $n$  from 2 to 5. The number of evaluations of the radial basis function approximation  $f$  is given in Table 3. The algorithm was stopped if it verifiably found the global minimum to within a tolerance of  $10^{-2}$ , i.e.  $U_k - L_k < 10^{-2}$ . If this was not possible in 3,000 s (50 min) we give the tolerance reached by the algorithm instead. The experiments were performed on an AMD Phenom II X4 955 processor machine with 4 GB of RAM running Matlab R2011a and the NAG toolbox for Matlab, Mark 22.4 which provided the local optimization solvers. As one can see from the results, the canonical Lipschitz branch and bound algorithm shows the worst performance and takes considerably more iterations than all the other algorithms. This is because the lower bounding function (6.4) used in the algorithm only makes use of the function Lipschitz constant and is therefore quite crude. The other algorithms use much tighter bounds which explains their superior performance in lower dimensions. As the dimension increases, the need to split  $3^n$  balls at each iteration hampers the performance of our Lipschitz algorithm, however, the heuristic version (which splits considerably fewer balls at each iteration) consistently outperforms the overlapping balls algorithm. Nonetheless, one must concede that the tolerance returned by the heuristic algorithm is often that of a global minimum on a subset of  $[-4, 4]^n$ , particularly in higher dimensions. While the inability of the algorithms to complete in higher dimensions may seem disappointing one must bear in mind that these are research implementations written in object oriented Matlab which is very slow. Moreover, our Lipschitz algorithm can be trivially parallelised as the bounds on each ball can be computed independently which should lead to a significant speedup on modern multiprocessor hardware.

Let us now compare our proposed branch and bound algorithm with overlapping balls and its heuristic counterpart subject to convex constraints. As in the previous example, the objective function  $f : [-4, 4]^n \subset \mathbb{R}^n \rightarrow \mathbb{R}$  will be a cubic spline radial basis function approximation to the sum of sine functions, this time with the elliptical constraint



**Fig. 8** Contours of the radial basis function approximation (left) to the sum of sines function (right) on  $[-4, 4] \times [-4, 4]$ . As one can see the approximation is particularly accurate for this example

**Table 2** Run times of the different branch and bound algorithms on a radial basis function approximation to the sum of sines function for dimensions from 2 to 5

	2	3	4	5
Canonical Lipschitz ( $L_f$ )	101 s	$4 \times 10^{1a}$	$1 \times 10^{2a}$	$2 \times 10^{2a}$
Algorithm 2.1 ( $L_H$ )	3 s	542 s	$2 \times 10^{1a}$	$2 \times 10^{3a}$
Heuristic Algorithm 2.1 ( $L_H$ )	<b>2 s</b>	<b>56 s</b>	<b>2,390 s</b>	<b><math>2 \times 10^{1a}</math></b>

The best results for each dimension are denoted in bold

<sup>a</sup> The tolerance reached is given instead if the algorithm did not complete in 50 min

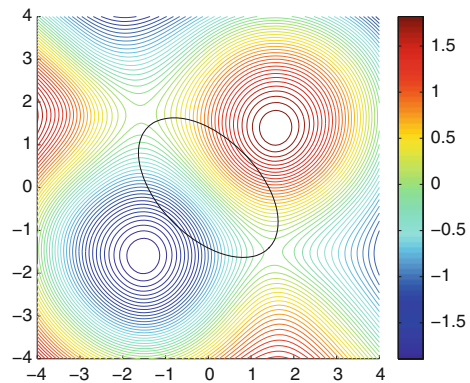
**Table 3** Number of evaluations of a radial basis function approximation to the sum of sines function for dimensions from 2 to 5 for the different branch and bound algorithms

	2	3	4	5
Canonical Lipschitz ( $L_f$ )	51308	334876	596512	890341
Algorithm 2.1 ( $L_H$ )	898	52736	33864	18490
Heuristic Algorithm 2.1 ( $L_H$ )	538	7036	96789	50921

$$c(x) = x^T Cx - 1 \leq 0$$

where  $C$  is a matrix with  $1/2$  on the diagonal and  $1/4$  elsewhere. This time the global minimum of the constrained surrogate typically lies on the boundary as one can see in Fig. 9. It is our intention to test the convergence of the algorithms on the boundary, for if the global minimum were to lie in the interior, performance would be comparable to the bound constrained example above. Table 4 shows the run time in seconds of a Matlab implementation of each constrained branch and bound algorithm for dimensions  $n$  from 2 to 5. The number of evaluations of the radial basis function approximation  $f$  is given in Table 5. The testing methodology and hardware is the same as in the previous experiments. The speed advantage of the heuristic Lipschitz algorithm is evident, once again it consistently outperforms the algorithm using balls for dimensions greater than 2. Even so, the results are worse than in the bound constrained example above (except for  $n = 4$ ). This is because the boundary

**Fig. 9** Contours of the radial basis function approximation to the sum of sines function on  $[-4, 4] \times [-4, 4]$  with the elliptical constraint region denoted in *black*



**Table 4** Run times of the different convex constrained branch and bound algorithms on a radial basis function approximation to the sum of sines function for dimensions from 2 to 5

	2	3	4	5
Algorithm 2.1 ( $L_H$ )	<b>2 s</b>	$8 \times 10^{-2a}$	$1 \times 10^{0a}$	$2 \times 10^{1a}$
Heuristic Algorithm 2.1 ( $L_H$ )	3 s	<b>1,069 s</b>	<b>405 s</b>	<b>2,443 s</b>

The best results for each dimension are denoted in bold

<sup>a</sup> The tolerance reached is given instead if the algorithm did not complete in 50 min

**Table 5** Number of evaluations of a radial basis function approximation to the sum of sines function for dimensions from 2 to 5 for the different branch and bound algorithms

	2	3	4	5
Algorithm 2.1 ( $L_H$ )	570	66082	34798	13746
Heuristic Algorithm 2.1 ( $L_H$ )	727	32192	20811	58269

contains a large region where the function takes similar values and the algorithms expend a significant effort searching this region for the global minimiser. However, in such situations a constrained local search algorithm can easily locate the global minimum once the branch and bound algorithm has located its basin of attraction.

Finally, let us consider the performance of Algorithm 2.1 and its heuristic counterpart on RBF approximations to the extended Dixon–Szegő test set (see Dixon and Szegő 1978) which is widely used in the global optimization literature (see e.g. Huyer and Neumaier 2008). The test set consists of nine functions with dimensions  $n$  ranging from two to six all defined on rectangular domains. Table 6 gives a brief overview of the test functions and Table 7 gives results for the algorithms on RBF approximations to the test functions. The number of evaluations of the radial basis function approximation  $f$  is given in Table 8. These approximations interpolate each test function at  $10n$  Halton samples (Halton 1960). The testing methodology and hardware is the same as for the previous examples and we once again compare against the canonical Lipschitz branch and bound algorithm.

**Table 6** The dimension  $n$ , number of local and global minima and domain  $\mathcal{D}$  for each of the Dixon–Szegő test functions

	$n$	Local minima	Global minima	$\mathcal{D}$
Branin	2	3	3	$[-5, 10] \times [0, 15]$
Six-hump Camel	2	6	2	$[-3, 3] \times [-2, 2]$
Goldstein-Price	2	4	1	$[-2, 2]^2$
Shubert	2	760	18	$[-10, 10]^2$
Hartman 3	3	4	1	$[0, 1]^3$
Shekel 5	4	5	1	$[0, 10]^4$
Shekel 7	4	7	1	$[0, 10]^4$
Shekel 10	4	10	1	$[0, 10]^4$
Hartman 6	6	4	1	$[0, 1]^6$

**Table 7** Run times of the two variants of Algorithm 2.1 and the canonical Lipschitz algorithm on a radial basis function approximation to functions from the Dixon–Szegő test set

RBF approx. to:	Algorithm 2.1 ( $L_H$ )	Heuristic Algorithm 2.1 ( $L_H$ )	Canonical Lipschitz ( $L_f$ )
Branin	5 s	<b>3 s</b>	63 s
Six-hump Camel	7 s	<b>3 s</b>	$5 \times 10^{-2a}$
Goldstein-Price	13 s	<b>12 s</b>	$4 \times 10^{3a}$
Shubert	12 s	<b>10 s</b>	$2 \times 10^{2a}$
Hartman 3	271 s	<b>39 s</b>	$5 \times 10^{0a}$
Shekel 5	$3 \times 10^{-1a}$	<b>1,080 s</b>	$2 \times 10^{0a}$
Shekel 7	$4 \times 10^{-1a}$	<b>671 s</b>	$4 \times 10^{0a}$
Shekel 10	$5 \times 10^{-1a}$	<b>682 s</b>	$4 \times 10^{0a}$
Hartman 6	$6 \times 10^{2a}$	<b><math>5 \times 10^{1a}</math></b>	$1 \times 10^{2a}$

The best results for each dimension are denoted in bold

<sup>a</sup> The tolerance reached is given instead if the algorithm did not complete in 50 min

## 7 Conclusions

We have presented an entirely new algorithm for the minimisation of a twice differentiable nonconvex objective function with a Lipschitz continuous Hessian over a compact, convex set. The algorithm is based on the canonical branch and bound global optimization algorithm with bounds inspired by the trust region subproblem from local optimization. Numerical results suggest that the proposed algorithm outperforms existing Lipschitz based approaches in the literature when applied to a radial basis function approximation and one would expect good performance in other settings where a Hessian Lipschitz constant is available. One main advantage of the proposed algorithm is that it is naturally parallelisable, especially in higher dimensions, and we are currently looking into a large-scale parallel implementation. The algorithm as presented here is, however, more conceptual since the paper is aimed at establishing ideas and exploring methods rather than providing a practical algorithm for day to day use. We are currently working on a more sophisticated implementation that will explore more practical approaches and other relevant ideas including the use of parallelism in more detail.

**Table 8** Number of evaluations of a radial basis function approximation to functions from the Dixon–Szegő test set for the two variants of Algorithm 2.1 and the canonical Lipschitz algorithm

RBF approx. to:	Algorithm 2.1 ( $L_H$ )	Heuristic algorithm 2.1 ( $L_H$ )	Canonical Lipschitz ( $L_f$ )
Branin	1246	909	32108
Six-hump Camel	1750	880	266412
Goldstein-Price	3396	3151	252655
Shubert	3278	2663	293122
Hartman 3	29516	4926	277710
Shekel 5	22340	36720	526955
Shekel 7	41894	23731	528336
Shekel 10	47568	24458	523643
Hartman 6	20794	24527	547412

**Acknowledgments** We would like to thank Coralia Cartis for helpful suggestions on an early draft of this paper. We would also like to thank an anonymous referee for positive comments and revisions which have helped improve the paper. This research was supported through an EPSRC Industrial CASE studentship in conjunction with Schlumberger. The work of Nick Gould was supported by the EPSRC grants EP/E053351/1, EP/F005369/1 and EP/I013067/1. This publication was also based on work supported in part by Award No KUK-C1-013-04, made by King Abdullah University of Science and Technology (KAUST) (CLF).

## References

- Androulakis, I.P., Maranas, C.D., Floudas, C.A.:  $\alpha$ BB: a global optimization method for general constrained nonconvex problems. *J. Glob. Optim.* **7**, 337–363 (1995). doi:[10.1007/BF01099647](https://doi.org/10.1007/BF01099647)
- Balakrishnan, V., Boyd, S., Balemi, S.: Branch and Bound algorithm for computing the minimum stability degree of parameter-dependent linear systems. *Int. J. Robust Nonlinear Control* **1**(4), 295–317 (1991). doi:[10.1002/rnc.4590010404](https://doi.org/10.1002/rnc.4590010404)
- Bishop, C.M.: *Neural Networks for Pattern Recognition*. Oxford University Press, Oxford. [http://books.google.co.uk/books?id=-aAwQO\\_-rXwC](http://books.google.co.uk/books?id=-aAwQO_-rXwC) (1996)
- Boyd, S., Vandenberghe, L.: *Convex Optimization*. Cambridge University Press, Cambridge. <http://www.stanford.edu/~boyd/cvxbook/> (2004)
- Busby, D., Farmer, C.L., Iske, A.: Hierarchical nonlinear approximation for experimental design and statistical data fitting. *SIAM J. Sci. Comput.* **29**(1), 49–69 (2007). doi:[10.1137/050639983](https://doi.org/10.1137/050639983)
- Cartan, H.: *Differential calculus. Cours de mathématiques II*, Hermann. [http://books.google.co.uk/books?id=PIg\\_AQAIAAJ](http://books.google.co.uk/books?id=PIg_AQAIAAJ) (1971)
- Cartis, C., Gould, N.I.M., Toint, P.L.: Adaptive cubic regularisation methods for unconstrained optimization. Part I: motivation, convergence and numerical results. *Math. Program.* (2009). doi:[10.1007/s10107-009-0286-5](https://doi.org/10.1007/s10107-009-0286-5)
- Chilès, J., Delfiner, P.: *Geostatistics: Modeling Spatial Uncertainty*. Wiley Series in Probability and Statistics. Wiley, New York. <http://books.google.co.uk/books?id=adkSAQAIAAJ> (1999)
- Conn, A.R., Gould, N.I.M., Toint, P.L.: *Trust region methods*. MPS-SIAM Series on Optimization, SIAM. <http://books.google.co.uk/books?id=5kNC4fqssYQC> (2000)
- Conway, J., Sloane, N.: *Sphere Packings, Lattices, and Groups*, Grundlehren der mathematischen Wissenschaften, vol 290. Springer, Berlin. <http://www.springer.com/mathematics/algebra/book/978-0-387-98585-5> (1999)
- Dixon, L.C.W., Szegő, G.P.: The global optimisation problem: an introduction. In: Dixon, L.C.W., Szegő, G.P. (eds.) *Towards Global Optimisation*, vol. 2, pp 1–15, North-Holland. <http://books.google.co.uk/books?id=SUSrAAAIAAJ> (1978)
- Farmer, C.L., Fowkes, J.M., Gould, N.I.M.: Optimal well placement. Presented at the 12th European Conference on the Mathematics of Oil Recovery, Oxford, 6–9 September 2010. <http://www.earthdoc.org/detail.php?pubid=41313>



- Floudas, C., Pardalos, P.: Handbook of Test Problems in Local and Global Optimization. Nonconvex Optimization and its Applications. Kluwer, Dordrecht. <http://books.google.co.uk/books?id=vndwQgAACAAJ> (1999)
- Forrester, A., Söbester, A., Keane, A.: Engineering Design via Surrogate Modelling: A Practical Guide. Progress in Astronautics and Aeronautics. Wiley, New York. <http://books.google.co.uk/books?id=AukeAQAAIAAJ> (2008)
- Gould, N.I.M., Robinson, D.P., Thorne, H.S.: On solving trust-region and other regularised subproblems in optimization. *Math. Program. Comput.* **2**(1), 21–57 (2010). doi:[10.1007/s12532-010-0011-7](https://doi.org/10.1007/s12532-010-0011-7)
- Griewank, A.: The modification of newton's method for unconstrained optimization by bounding cubic terms. Tech. Rep. NA/12 (1981), Department of Applied Mathematics and Theoretical Physics, University of Cambridge (1981)
- Halton, J.H.: On the efficiency of certain quasi-random sequences of points in evaluating multi-dimensional integrals. *Numerische Mathematik* **2**, 84–90 (1960). doi:[10.1007/BF01386213](https://doi.org/10.1007/BF01386213)
- Horst, R.: A general class of branch-and-bound methods in global optimization with some new approaches for concave minimization. *J. Optim. Theory Appl.* **51**(2), 271–291 (1986). doi:[10.1007/BF00939825](https://doi.org/10.1007/BF00939825)
- Horst, R., Pardalos, P.M.: Handbook of Global Optimization, Nonconvex Optimization and its Applications, vol. 2. Springer, Berlin. <http://www.springer.com/mathematics/book/978-0-7923-3120-9> (1995)
- Huyer, W., Neumaier, A.: SNOBFIT—stable noisy optimization by branch and fit. *ACM Trans. Math. Softw.* **35**(2):9:1–9:25 (2008). doi:[10.1145/1377612.1377613](https://doi.org/10.1145/1377612.1377613)
- Jones, D.R.: A taxonomy of global optimization methods based on response surfaces. *J. Glob. Optim.* **21**(4), 345–383 (2001). doi:[10.1023/A:1012771025575](https://doi.org/10.1023/A:1012771025575)
- Jones, D.R., Perttunen, C.D., Stuckman, B.E.: Lipschitzian optimization without the Lipschitz constant. *J. Optim. Theory Appl.* **79**(1), 157–181 (1993). doi:[10.1007/BF00941892](https://doi.org/10.1007/BF00941892)
- Kreinovich, V., Kearfott, R.: Beyond convex? Global optimization is feasible only for convex objective functions: a theorem. *J. Glob. Optim.* **33**, 617–624 (2005). doi:[10.1007/s10898-004-2120-1](https://doi.org/10.1007/s10898-004-2120-1)
- Nesterov, Y., Polyak, B.: Cubic regularization of Newton method and its global performance. *Math. Program.* **108**(1), 177–205 (2006). doi:[10.1007/s10107-006-0706-8](https://doi.org/10.1007/s10107-006-0706-8)
- Neumaier, A.: Complete search in continuous global optimization and constraint satisfaction. *Acta Numerica* **13**, 271–369 (2004). doi:[10.1017/S0962492904000194](https://doi.org/10.1017/S0962492904000194)
- O'Hagan, A.: Curve fitting and optimal design for prediction. *J. R. Stat. Soc. B* **40**(1), 1–42 (1978). doi:[10.2307/2984861](https://doi.org/10.2307/2984861)
- Pardalos, P.M., Romeijn, H.E.: Handbook of Global Optimization Volume 2, Nonconvex Optimization and its Applications, vol. 62. Springer, Berlin. <http://www.springer.com/mathematics/book/978-1-4020-0632-6> (2002)
- Pardalos, P.M., Horst, R., Thoai, N.V.: Introduction To Global Optimization, Nonconvex Optimization and its Applications, vol. 3. Springer, Berlin. <http://www.springer.com/mathematics/book/978-0-7923-3556-6> (1995)
- Rasmussen, C.E., Williams, C.K.I.: Gaussian Processes for Machine Learning. Adaptive Computation and Machine Learning, MIT Press, Cambridge. <http://www.gaussianprocess.org/gpml/chapters/RW.pdf> (2006)
- Rippa, S.: An algorithm for selecting a good value for the parameter  $c$  in radial basis function interpolation. *Adv. Comput. Math.* **11**(2), 193–210 (1999). doi:[10.1023/A:1018975909870](https://doi.org/10.1023/A:1018975909870)
- Santner, T.J., Williams, B.J., Notz, W.: The Design and Analysis of Computer Experiments. Springer Series in Statistics. Springer, Berlin. <http://www.springer.com/statistics/statistical+theory+and+methods/book/978-0-387-95420-2> (2003)
- Spall, J.C.: Introduction to Stochastic Search and Optimization. Wiley-Interscience Series in Discrete Mathematics and Optimization. Wiley, New York. <http://books.google.co.uk/books?id=f66OivvKKnAC> (2003)
- Wendland, H.: Scattered Data Approximation. Cambridge Monographs on Applied and Computational Mathematics. Cambridge University Press, Cambridge. <http://books.google.co.uk/books?id=qy4cbWUMSyYC> (2005)