Adaptive Least Squares Fitting with Radial Basis Functions on the Sphere

Gregory E. Fasshauer

Abstract. We investigate adaptive least squares approximation to scattered data given over the surface of the unit sphere in $\mathbb{R}^3$. Two basic algorithms (as well as some modifications) are described and evaluated: knot insertion and knot removal.

§1. Introduction

Since their introduction by R. L. Hardy in the 1970s, radial basis functions have become a standard tool in CAGD and approximation theory for the construction of surfaces interpolating and approximating scattered data. Most of the work on scattered data approximation with radial basis functions has been in the Euclidean setting (see e.g. [4], and some references therein). On the sphere interpolation was treated e.g. in [1,3,7,8]. So far there has been no discussion of the use of radial basis functions for approximation over the sphere. This, however, is a problem of great interest in many areas of the applied sciences, in particular geodesy and meteorology.

The purpose of this paper is to investigate adaptive approximation to scattered data defined over the surface of the sphere. Our main goal is to use only as few knots as necessary (selected according to some quality criterion) to achieve a desired accuracy for the fit. We describe two algorithms – one based on adaptively *inserting* knots, the other on knot *removal* – and compare the results.

The paper is organized as follows. We formulate the general approximation problem in Section 2, and specialize the discussion to adaptive least squares approximation using knot insertion and knot removal in Sections 3 and 4, respectively. In Section 5 we briefly mention some possible modifications of the two basic algorithms and our experience with them. We close the paper with a few examples in Section 6.

Mathematical Methods in CAGD III
M. Dæhlen, T. Lyche, and L. L. Schumaker (eds.), pp. 1–10.

ISBN xxx.

§2. Least Squares Approximation with Radial Basis Functions

Suppose we are given n distinct data points $X = \{x_1, \dots, x_n\}$ on the *unit sphere* S_2 in $\mathbb{R}^3$ as well as associated data $d_i \in \mathbb{R}$, $i = 1, \dots, n$, and a set of m distinct *knots* $\Xi = \{\xi_1, \dots, \xi_m\}$ on S_2. In general we will assume that we have fewer knots than data points, i.e. $m \leq n$. Our goal is to find an approximant of the form

$$s(x) = \sum_{j=1}^{m} c_j g\big(\delta(x, \xi_j)\big), \qquad x \in S_2, \tag{1}$$

such that

$$\sum_{i=1}^{n} [d_i - s(x_i)]^2 \to min.$$

Here $g(t) : [0, \pi] \to \mathbb{R}$ is a univariate *radial function* and $\delta(x, y)$ is the *geodesic distance* between two points x and y on the unit sphere, i.e.

$$\delta(x, y) = \arccos(x \cdot y),$$

where $(\cdot)$ stands for the dot product in $\mathbb{R}^3$.

The approximation problem (1) has a unique solution if the matrix

$$A_{ij} = g\big(\delta(x_i, \xi_j)\big), \quad i = 1, \dots, n, \quad j = 1, \dots, m,$$

has full rank. Nothing is known about the rank of the matrix A if the knots Ξ are chosen different from the data sites X; e.g., whether interpolation under these circumstances is possible, is an open question even in the planar setting (see e.g. [2]). However, if $\Xi \subset X$ and the radial function g is chosen appropriately then A has full rank, since in this case the approximation problem contains interpolation with knots at the data sites as a sub-problem.

Nonsingularity of *interpolation* matrices for the sphere has been investigated by various authors. Cheney, Light and Xu (see e.g. [1]), and Ron and Sun [8] showed that if $\Xi = X$ and g is strictly positive definite on the sphere, then A is non-singular. In [3,7] these results were extended for the case in which g is strictly conditionally negative definite. The well-established spherical multi-quadric (SMQ) introduced by Hardy (see e.g. [5]) is strictly conditionally negative definite as shown in [3]. For the sake of definiteness and since for practical purposes there seems to be no great difference between the various methods on the sphere, we will use only SMQs in this paper, i.e. radial functions of the form

$$g(t) = \sqrt{1 + r^2 - 2r\cos t}, \qquad r > 0, \tag{2}$$

where r is a user-selectable parameter. Nevertheless, each of the following algorithms can be applied to any radial function which is covered by the results mentioned in the previous paragraph.

The least squares problem (1) can be solved by employing standard techniques such as QR or SVD factorization.

§3. Knot Insertion

When fitting data by linear combinations of certain basis functions, it is a classical technique to improve the quality of a given initial approximation by increasing the number of basis functions used for the fit. Since every radial basis function is associated with one particular knot (or center), this can be achieved by adding new knots to the existing ones. For planar multi-quadrics this idea was explored in [4]. We will now describe the analogous algorithm for the sphere, and later compare it to the "reverse" process of knot removal.

Let us assume we are given a large number of data and we want to fit them with an SMQ surface within a given tolerance. The idea is to start with very few (e.g. six equally spaced) knots on the surface of the unit sphere, and then, as long as the least squares error exceeds the tolerance, repeatedly insert a knot at that data location whose least squares error component is largest. Here is the algorithm in detail:

Algorithm 1: Knot insertion.

(1) Let data points $X = \{x_1, \ldots, x_n\}$, data $d_i, i = 1, \ldots, n$, and a tolerance TOL be given.

(2) Choose m_0 initial knots Ξ and let $m = m_0$, and calculate the least squares fit with its associated error

$$LSQ := \sum_{i=1}^{n} [d_i - s(x_i)]^2 .$$

While $LSQ > TOL$ **do**

(3) "Weight" each data point x_i according to its error component, i.e. let

$$w_i := |d_i - s(x_i)|, \quad i = 1, \ldots, n.$$

(4) Find the data point $x_\nu \notin \Xi$ with maximum weight w_ν and insert it as a knot, i.e.

$$\Xi = \Xi \cup \{x_\nu\} \quad \text{and} \quad m = m + 1.$$

A few comments are in order. First we note that we have to solve one linear least squares problem for every knot we add. The size of these problems increases by one at each step. Secondly, if the initial knots are chosen to lie at data sites, this process will always have a full rank observation matrices A. This is guaranteed, since we only add data sites as new knots, and we make sure in step (5) that no multiple knots are created (which would obviously lead to a rank deficiency).

§4. Knot Removal

The idea of knot removal was primarily motivated by the need for data reduction, but it can also be used for the purpose of adaptive approximation (for a survey on knot removal see e.g. [6]). The basic idea is to start with a good fit (e.g. an interpolation to the data), and then successively reduce the number of knots (and therefore basis functions) used until a certain given tolerance is reached. Specifically, this means we will start with an initial fit and then use some kind of weighting strategy for the knots, so that we can repeatedly remove those contributing least to the accuracy of the fit. Our algorithm is as follows:

Algorithm 2: Knot removal.

(1) Let data points $X = \{x_1, \ldots, x_n\}$, data $d_i, i = 1, \ldots, n$, and a tolerance TOL be given.

(2) Choose m_0 initial knots Ξ and let $m = m_0$, and calculate an initial fit with its associated LSQ.

While $LSQ < TOL$ **do**

(3) "Weight" each knot ξ_j according to its least squares error, i.e. form

$$\tilde{\Xi} = \Xi \setminus \{\xi_j\},$$

and calculate the weights

$$w_j := \sum_{i=1}^{n} \left[d_i - \tilde{s}(x_i)\right]^2,$$

where

$$\tilde{s}(x) := \sum_{j=1}^{m-1} c_j g\big(\delta(x, \tilde{\xi}_j)\big)$$

is the surface fit based on the reduced set of knots $\tilde{\Xi}$.

(4) Find the knot ξ_μ with lowest weight $w_\mu < TOL$ and permanently remove it, i.e.

$$\Xi = \Xi \setminus \{\xi_\mu\} \quad \text{and} \quad m = m - 1.$$

Again we would like to comment on the algorithm. First, as far as computational times are concerned, this algorithm is *much* slower than Algorithm 1, since the weight for every knot is determined by the solution of a least squares problem, i.e., in every iteration we solve m least squares problems. These problems, however, become increasingly smaller. Secondly, if the initial knots are chosen at the data sites, then again there will be no problems with the observation matrix becoming rank deficient.

Since the weighting process in Algorithm 2 requires so much time, it is natural to think of some kind of strategy to speed up the knot removal process. It is a well-established technique when doing knot removal to remove certain groups of knots at once after every weighting cycle. We use the following idea which considerably improves the runtime of Algorithm 2. Instead of removing only a single knot, we temporarily remove half of the knots with weights less than the tolerance. If this keeps the least squares error of the resulting fit below the tolerance, we permanently remove the knots and reweight. However, if this would lead to a violation of the tolerance, we simply try half of these knots, and repeat this process until we are successful. Here are the details of the alternative removal strategy:

Algorithm 2′: Knot removal in groups.

(1) – (3) as in Algorithm 2.

(4′) Find half the knots with weights less than TOL, i.e. let

$$\kappa := \frac{1}{2}\#\{\xi_\mu \in \Xi : w_\mu < TOL\}.$$

While $\kappa > 0$ **do**

(a) Temporarily remove those κ knots Ξ_κ with lowest weights, i.e.

$$\tilde{\tilde{\Xi}} = \Xi \setminus \Xi_\kappa.$$

(b) Compute the resulting fit and its LSQ.

If $LSQ < TOL$

Remove the knots permanently, i.e.

$$\Xi = \tilde{\tilde{\Xi}} \quad \text{and} \quad m = m - \kappa.$$

and **exit loop**.

Else

Try again with half the knots, i.e. let

$$\kappa = \kappa/2$$

and **repeat**.

We note that the strategy for removal of a whole group of knots depends heavily on the particular application at hand and should therefore be chosen by the user. For our experiments the above strategy has worked quite well, and it has the advantage that the tolerance is never exceeded, which could be the case for a more aggressive removal strategy, such as simply removing a certain percentage of the eligible knots.

Finally, one could also imagine a very crude weighting strategy according to which the knot to be removed is that whose coefficient c_j is smallest. Thus, one only has to solve *one* least squares problem per weighting cycle. Although this process is quite fast, we do not recommend it. For an illustration of its poor overall performance (especially when then tolerance is tight) see the tables in Section 6.

§5. Modifications of Algorithms 1 and 2

In this section we briefly mention two possible modifications of our basic algorithms. Since both Algorithm 1 and 2 add/remove one knot at a time, one could use an updating QR factorization to make the process more efficient. However, we have not implemented this yet for two reasons. First, the knot insertion algorithm starts with very small (and therefore fast) problems, and thus we have not found it necessary to update our factorization. Secondly, in the knot removal case the updating process could only be applied to the basic single-knot version, and we believe that the grouping strategy mentioned in the previous section is superior. However, we do intend to investigate this problem in the near future.

The second modification is of a different nature. So far we have dealt only with linear least squares problems, i.e. we have been considering problems of the form

$$\min_{c_j} \sum_{i=1}^{n} \left[d_i - s(x_i)\right]^2 ,$$

where the c_j, $j = 1, \ldots, m$, are the coefficients in the expansion of our approximant. Instead of minimizing only over these coefficients, we can also consider a nonlinear least squares problem, as suggested by [4] for the knot insertion process in the planar case, where the knot locations are also parameters in the minimization. Then the problem has the form

$$\min_{\xi_j} \min_{c_j} \sum_{i=1}^{n} \left[d_i - s(x_i)\right]^2 .$$

We now have to add a nonlinear minimization step (e.g. using the standard Levenberg-Marquardt algorithm) to our two basic algorithms. For Algorithm 1 this could be done either every time a knot is added, or after all the necessary knots have been inserted. We have programmed both approaches. Our experience (see the following section) shows that some improvement of the error can be achieved with a single nonlinear step at the end, but significant improvement (as well as a reduced number of knots) is only seen if we optimize the knot locations every time a new knot is added. This, however, is a very expensive process. We should also note that after moving the knots in this way there no longer remains any guarantee for a full rank observation matrix.

We have also experimented with adding the nonlinear optimization to the knot removal algorithm. Thus, every time a group of knots is removed, we adjust the remaining ones "optimally". Again, we get an improvement, but there is no guarantee that the observation matrix has full rank.

§6. Examples

In the closing section we give some examples and make a few observations based on our experience with the methods discussed above. For the following examples we used the SMQ method (2) with the parameter $r = 0.4$. In both of the following examples we use the function

$$f(x, y, z) = \frac{1 + 10xyz + x^8 + e^{2y^3} + e^{2z^2}}{14} \tag{3}$$

and then compare the knot insertion (Algorithm 1), the crude knot removal, knot removal in groups (Algorithm 2′), as well as the two nonlinear versions of the knot insertion algorithm, and the nonlinear version of knot removal in groups. All of the experiments are performed using a specified tolerance TOL for a given set of data. We measure the maximum error of the approximants on a grid of 3846 nearly equally spaced points on the sphere. The tables below also contain the root-mean-square errors on the grid, the number of knots required to satisfy the tolerance, and the execution time.

Example 1. We sample the test function (3) at 66 nearly uniformly spaced points with $x^2 + y^2 + z^2 = 1$, and run our algorithms with $TOL = 0.1$ and $TOL = 0.0001$.

Method	max (ρ)	rms (grid)	# knots	time
KI	8.922212e-02	1.320485e-03	11	1 sec
KRc	1.123910e-01	1.598014e-03	18	33 sec
KRg	9.965815e-02	1.340359e-03	11	104 sec
KInl1	6.321324e-02	7.319459e-04	11	6 sec
KInl2	8.931183e-02	9.370717e-04	11	8 sec
KRnl	8.613934e-02	1.254269e-03	11	164 sec

Table 1. Example 1 with $TOL = 0.1$

Method	max (ρ)	rms (grid)	# knots	time
KI	3.773904e-03	1.411989e-06	46	11 sec
KRc	6.157838e-03	2.180199e-06	64	4 sec
KRg	4.671680e-03	1.304296e-06	40	328 sec
KInl1	3.728481e-03	1.390322e-06	46	48 sec
KInl2	6.599700e-03	1.962770e-06	45	426 sec
KRnl	3.517820e-03	9.852247e-07	39	801 sec

Table 2. Example 1 with $TOL = 0.0001$

In Figure 1 we display the original data sites as well as the knot locations calculated by three of the methods listed in Table 2. The intersection of each ray with the unit sphere represents one knot location. For illustrative purposes, however, we have chosen to graph the "optimized" knots in conjunction

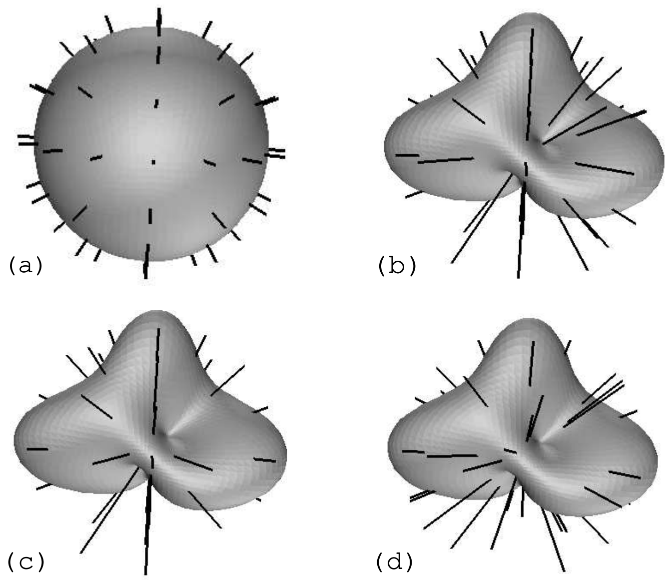

Fig. 1. (a) Original data sites, Knots for (b) KI, (c) KRg, (d) KInl2.

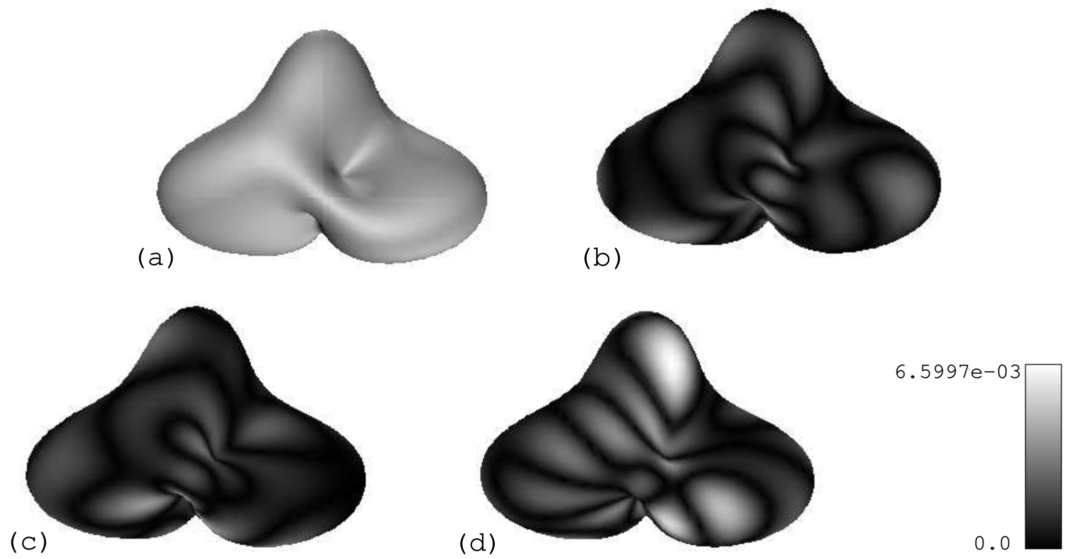

Fig. 2. (a) Test function, (b) KI, (c) KRg, (d) KInl2.

with the test function. The corresponding graphs of the actual surfaces are shown in Figure 2, where the function value is displayed as a radial "height". The graphs are shaded according to their maximum errors (dark $\widehat{=}$ small). It might also be of interest to note that interpolation to this set of data yields a maximum error of $\rho = 1.370892\text{e-}03$.

Example 2. Now we sample the test function at 6 equally spaced and an additional 94 pseudo-randomly spaced points on the sphere, and run our algorithms with $TOL = 0.1$ and $TOL = 0.001$.

For the examples listed in Tables 3 and 4 we select the six equally spaced, plus an additional 44 random knots from the initial 94 in order to obtain the initial fit for the knot removal algorithm. This results in an initial least squares error of $LSQ = 0.000537$.

Method	max (ρ)	rms (grid)	# knots	time
KI	1.182602e-01	1.013084e-03	13	1 sec
KRc	1.963647e-01	2.997443e-03	17	21 sec
KRg	1.384517e-01	1.641060e-03	16	67 sec
KInl1	9.391227e-02	5.677149e-04	13	8 sec
KInl2	8.189870e-02	6.118573e-04	12	14 sec
KRnl	1.135853e-01	8.611450e-04	14	91 sec

Table 3. Example 2 with $TOL = 0.1$

Method	max (ρ)	rms (grid)	# knots	time
KI	6.774351e-02	1.050028e-04	36	10 sec
KRc	4.738008e-02	1.369481e-04	48	4 sec
KRg	5.683684e-02	8.696657e-05	38	217 sec
KInl1	6.734380e-02	1.036336e-04	36	36 sec
KInl2	2.693276e-02	4.422667e-05	31	146 sec
KRnl	4.969258e-02	6.851652e-05	27	403 sec

Table 4. Example 2 with $TOL = 0.001$

Again we note that interpolation to these 100 points results in a maximum error of $\rho = 2.619738\text{e-}02$. The maximum error for a least squares approximation based on the subset of 50 randomly selected knots is $\rho = 5.134044\text{e-}02$.

Discussion: Although the crude version of the knot removal algorithm is quite fast, we do not recommend its use since for small tolerances hardly any knots are removed (see Tables 2 and 4).

If *speed* is the main criterion when calculating the approximant we recommend knot insertion in its basic form. Adding only a single nonlinear step (KInl1) does not improve the error significantly enough (and the number of knots is unchanged). If it is the *number of knots* one would like to minimize (i.e. for situations in which many evaluations based on the same data are required, e.g. in manufacturing), then intuitively knot removal should have an advantage over knot insertion, since for knot removal one begins with a "good" initial fit, whereas in the case of knot insertion one starts with a fit which has almost no relation to the final result. This is confirmed by our examples. However, sometimes it takes the nonlinear version of the knot removal algorithm to get really good results (see Table 4, where the number of knots with weights small enough to be taken out is quite small). A possible compromise is given by the knot insertion algorithm with nonlinear optimization after every addidtion of a knot (KInl2). This is true especially if one anticipates that only a small number of knots will be needed (and thus a large number would have to be removed).

The errors given in Tables 2 and 4 and the maximum errors mentioned for the case of interpolation show that adaptive approximation can be very effective. The errors after the adaptive algorithms are almost as good as the interpolation errors. In Example 2 the nonlinear variants of the algorithms actually performed better than a straightforward approximation with 50 random knots (compare $\rho = 5.134044\text{e-}02$ and rms $= 6.201014\text{e-}05$ to the entries in Table 4). This is due to the optimized knot locations achieved by the nonlinear methods.

References

1. Cheney, E. W., Approximation by functions of nonclassical form, *Approximation Theory, Spline Functions and Applications*, S. P. Singh (ed.), Kluwer, 1992, 1–18.
2. Cheney, E. W., and Y. Xu A set of research problems in approximation theory, *Topics in Polynomials of One and Several Variables and Their Applications*, T. M. Rassias, H. M. Srivastava, and A. Yanushauskas (eds.), World Scientific Publishers (London), 1993, 109–123.
3. Fasshauer, G., Scattered Data Interpolation with Radial Basis Functions on Spheres, manuscript, 1994.
4. Franke, R., H. Hagen, and G. Nielson, Least squares surface approximation to scattered data using multiquadric functions, report Universität Kaiserslautern, 1993.
5. Hardy, R. L., and W. M. Göpfert, Least squares prediction of gravity anomalies, geodial undulations, and deflections of the vertical with multiquadric harmonic functions, Geophys. Res. Letters **2** (1975), 423–426.
6. Lyche, T., Knot Removal for Spline Curves and Surfaces, *Approximation Theory VII*, E. W. Cheney, C. Chui, and L. Schumaker (eds.), Academic Press (New York), 1992, 207–226.
7. Menegatto, V. A., Strictly positive definite kernels on the Hilbert sphere, Applicable Analysis, to appear.
8. Ron, A, and X. Sun, Strictly positive definite functions on spheres, University of Wisconsin, 1994.

Gregory E. Fasshauer
Department of Mathematics
Vanderbilt University
Nashville, TN 37240
email: fass@math.vanderbilt.edu