Preconditioners for Low Order Thin Plate Spline Approximations

Linda Stals[1] and Stephen Roberts[1]

Department of Mathematics, Australian National University,
Canberra, ACT, 0200, Australia. stals@maths.anu.edu.au

A commonly used method for fitting smooth functions to noisy data is the thin-plate spline method. Traditional thin-plate splines use radial basis functions and consequently require the solution of a dense linear system of equations whose dimension grows linearly with the number of data points. Here we discuss a method based on low order polynomial functions with locally supported basis functions. An advantage of such an approach is that the resulting system of equations is sparse and its dimension depends linearly on the number of nodes in the finite element grid instead of the number of data points.

Another advantage is that an iterative solver, such as the conjugate gradient method, can be used. However it can be shown that the system of equations is similar to those arising from Tikhonov regularisation, and consequently the equations are ill-conditioned for certain choices of the parameters. To ensure that the method is robust an appropriate preconditioner must be used.

In this paper we present the discrete thin-plate spline method and explore a set of preconditioners. We discuss some of the properties that are unique to our particular formulation and verify that the multiplicative Schwarz method is an effective preconditioner.

1 Introduction

The thin-plate spline method is a popular data fitting technique because it is insensitive to noise in the data. For a general domain Ω the thin-plate spline f (as discussed by [10] and [3]) minimises the functional

$$J_\alpha(f) = \frac{1}{n}\sum_{i=1}^{n}(f(\mathbf{x}^{(i)}) - y^{(i)})^2 + \alpha \int_\Omega \sum_{|\nu|=2} \binom{2}{\nu} (D^\nu f(\mathbf{x}))^2 d\mathbf{x}, \tag{1}$$

where $\nu = (\nu_1, ..., \nu_d)$ is a d dimensional multi-index, $|\nu| = \sum_{s=1}^{d} \nu_s$, $\mathbf{x}$ is a predictor variable in $\mathbb{R}^d$, and $\mathbf{x}^{(i)}$ and $y^{(i)}$ are respectively the corresponding i-th predictor and response data value ($1 \le i \le n$).

The parameter α controls the trade-off between smoothness and fit. Techniques for choosing α automatically, such as generalised cross validation (GCV), can be found in [5, 10].

Radial basis functions are often used to represent f, as they give an analytical solution of the minimiser of the functional in (1). However the resulting system of equations is dense, and furthermore its dimension is directly proportional to the number of data points.

In [7, 8] we proposed a discrete thin-plate spline method that uses piecewise functions with local support defined on a finite element mesh. In particular, the method described in Section 2 uses standard multi-linear finite element basis functions. The advantage of using functions with local support is that the dimension of the resulting system of sparse equations depends only on the number of grid points in the finite element mesh.

The system of equations resulting from the finite element discretisation can be manipulated to form a symmetric positive definite system, as shown in Section 3. However for small values of α this system is ill-conditioned and the convergence slows down markedly. Section 4 discusses some of reasons causing the difficulties with the convergence rate, and Section 5 shows that the convergence rate can be improved by using the multiplicative Schwarz preconditioner.

2 Discrete Thin Plate Splines

For simplicity most of the discussion is focused on two dimensional (2D) examples, although the theory generalises to three dimensions and the code has been developed for both two and three dimensions.

The smoothing problem from (1) can be approximated with finite elements so that the discrete smoother f is a linear combination of piecewise multi-linear basis functions (hat functions) $b_i(\mathbf{x}) \in H_0^1$,

$$f(\mathbf{x}) = \sum_{i=1}^{m} c_i b_i(\mathbf{x}) = \mathbf{b}(\mathbf{x})^T \mathbf{c}.$$

The idea is to minimise J_α over all f of this form. The smoothing term (the second term in (1)) is not defined for piecewise multi-linear functions, but the non-conforming finite element principle can be used to introduce piecewise multi-linear functions $\mathbf{u} = (\mathbf{b}(\mathbf{x})^T\mathbf{g}_1, \mathbf{b}(\mathbf{x})^T\mathbf{g}_2)$ to represent the gradient of f. The functions f and $\mathbf{u}$ satisfy the relationship

$$\int_\Omega \nabla f(\mathbf{x}) \cdot \nabla b_j(\mathbf{x})\, d\mathbf{x} = \int_\Omega \mathbf{u}(\mathbf{x}) \cdot \nabla b_j(\mathbf{x})\, d\mathbf{x}, \tag{2}$$

for all the basis functions b_j. This relationship ensures that $\mathbf{u}$ is an approximation of the gradient of f in a weak sense.

Constraint (2) is equivalent to the relationship

$$L\mathbf{c} - G_1\mathbf{g}_1 - G_2\mathbf{g}_2 = 0, \tag{3}$$

where L is a discrete approximation to the negative Laplace operator and (G_1, G_2) is a discrete approximation to the transpose of the gradient operator.

We now consider the minimiser of the functional

$$\begin{aligned} J_\alpha(\mathbf{c}, \mathbf{g}_1, \mathbf{g}_2) &= \frac{1}{n}\sum_{i=1}^{n}(\mathbf{b}(\mathbf{x}^{(i)})^T\mathbf{c} - y^{(i)})^2 + \alpha\int_{\Omega}\sum_{s=1}^{2}\nabla(\mathbf{b}^T\mathbf{g}_s)\cdot\nabla(\mathbf{b}^T\mathbf{g}_s)\,d\mathbf{x} \\ &= \mathbf{c}^T A\mathbf{c} - 2\mathbf{d}^T\mathbf{c} + \mathbf{y}^T\mathbf{y}/n + \alpha\left(\mathbf{g}_1^T L\mathbf{g}_1 + \mathbf{g}_2^T L\mathbf{g}_2\right). \end{aligned} \tag{4}$$

Our smoothing problem consists of minimising this functional over all vectors $\mathbf{c}, \mathbf{g}_1, \mathbf{g}_2$ defined on the domain Ω_h, subject to the constraint (3).

The matrices L, G_1 and G_2 are constructed independent of the data points but the matrix

$$A = \frac{1}{n}\sum_{i=1}^{n}\mathbf{b}(\mathbf{x}^{(i)})\mathbf{b}(\mathbf{x}^{(i)})^T,$$

and vector

$$\mathbf{d} = \frac{1}{n}\sum_{i=1}^{n}\mathbf{b}(\mathbf{x}^{(i)})y^{(i)},$$

are assembled by sweeping through the data points. Matrix A is symmetric, nonnegative, and sparse. In regions where the support of the basis functions do not contain any data points the matrix is zero.

The smoothing function defined by $f(\mathbf{x}) = \mathbf{b}(\mathbf{x})^T\mathbf{c}$ has essentially the same smoothing properties as the original thin plate smoothing spline, provided the discretisation is small enough, see [6].

By using Lagrange multipliers, the minimisation problem may be rewritten as the solution of the following linear system of equations

$$\begin{bmatrix} A & 0 & 0 & L \\ 0 & \alpha L & 0 & -G_1^T \\ 0 & 0 & \alpha L & -G_2^T \\ L & -G_1 & -G_2 & 0 \end{bmatrix}\begin{bmatrix}\mathbf{c}\\ \mathbf{g}_1\\ \mathbf{g}_2\\ \mathbf{w}\end{bmatrix} = \begin{bmatrix}\mathbf{d}\\0\\0\\0\end{bmatrix} - \begin{bmatrix}\mathbf{h}_1\\ \mathbf{h}_2\\ \mathbf{h}_3\\ \mathbf{h}_4\end{bmatrix}, \tag{5}$$

where $\mathbf{w}$ is a Lagrange multiplier associated with constraint (3). The vectors $\mathbf{h}_1, \cdots, \mathbf{h}_4$ store the Dirichlet boundary information.

For examples where the exact form of a the minimiser is known, the Dirichlet boundary conditions can be set accordingly, see [8]. Different boundary conditions will give different forms of the minimiser, we plan to explore this idea further as a means to incorporate prior information.

3 Solution of the Linear System

One way to solve (5) is to eliminate all the variables except $\mathbf{g}_1$ and $\mathbf{g}_2$, which gives

$$\begin{bmatrix} \alpha L + G_1^T Z G_1 & G_1^T Z G_2 \\ G_2^T Z G_1 & \alpha L + G_2^T Z G_2 \end{bmatrix}\begin{bmatrix}\mathbf{g}_1\\ \mathbf{g}_2\end{bmatrix} = \begin{bmatrix}G_1^T L^{-1}\mathbf{d}\\ G_2^T L^{-1}\mathbf{d}\end{bmatrix} - \begin{bmatrix}\mathbf{h}_2 + G_1^T L^{-1}\mathbf{h}\\ \mathbf{h}_3 + G_2^T L^{-1}\mathbf{h}\end{bmatrix}, \tag{6}$$

where $Z = L^{-1}AL^{-1}$, $\mathbf{h} = \mathbf{h}_1 - AL^{-1}\mathbf{h}_4$ and $\mathbf{c} = L^{-1}\left(G_1\mathbf{g}_1 + G_2\mathbf{g}_2 - \mathbf{h}_4\right)$.

Applying Z to a vector is equivalent to solving two systems of equations involving the Laplacian, so it is important to use an efficient Poisson solver. Fortunately there are techniques, such as the multigrid method, that are optimal for the solution of such problems.

System (6) is symmetric positive definite and may be solved using the preconditioned Conjugate Gradient (PCG) method.

The matrix on the left-hand side of (6) can be rewritten as

$$\left[\alpha\widehat{L} + \widehat{K}^T\widehat{K}\right], \tag{7}$$

where

$$\widehat{L} = \begin{bmatrix} L & 0 \\ 0 & L \end{bmatrix} \quad \text{and } \widehat{K}^T = \begin{bmatrix} G_1^T \\ G_2^T \end{bmatrix} L^{-1}A^{1/2}.$$

This is similar to the type of matrix arising in Tikhonov regularisation.

The system $\widehat{K}^T\widehat{K}$ is symmetric positive semidefinite, and for small values of α (6) is close to a semidefinite system. As shown in Section 5 the convergence rate of the PCG method deteriorates as α is reduced and in some cases the PCG method diverges.

3.1 Properties of Gradient Matrices

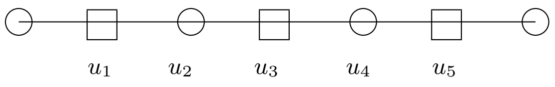

Fig. 1. Example grid in 1D used to demonstrate that the matrix G_1 may be singular.

The matrices G_s may be singular. To illustrate this consider the 1D grid shown in Figure 1. The stencil corresponding to the finite element approximation of the gradient is

$$h\begin{bmatrix} -1 & 0 & 1 \end{bmatrix}$$

and thus the vector $\mathbf{u} = \begin{bmatrix} a & 0 & a & 0 & a \end{bmatrix}^T$ belongs to the null space of G_1. In other words we can assign any constant value to the points marked by squares in Figure 1 and the gradient will be zero. Similar examples can be constructed for higher dimensional grids.

Instead of the domain shown in Figure 1, consider the domain labelled Subgrid 1 in Figure 2. The matrix G_1 defined on this domain is non-singular. We require domains like those shown in Figure 1 in order to use the multigrid method to solve the Laplacian. This lead to the use of the domain decomposition method to define subgrids where the matrices G_s are non-singular, thus reducing the convergence rate (see Section 4). As the subgrids are relatively small it is possible to use a different Laplacian solver, such as a sparse direct method.

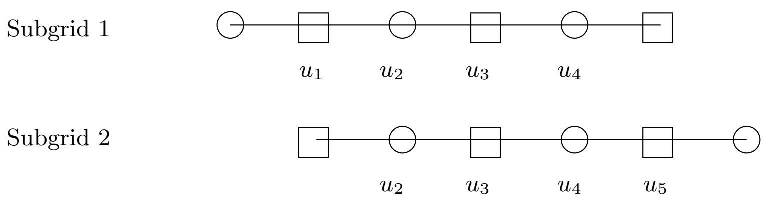

Fig. 2. Example subdivision of a grid in 1D where the gradient matrix is non-singular.

4 Preconditioners

For practical applications small α values are not usually of interest because the results will contain too much noise, however search algorithms like GCV may require some evaluations for small α before they find the optimal value of α. For larger values of α ($\alpha > 10^{-5}$) the preconditioner

$$M = \begin{bmatrix} L^{-1} & 0 \\ 0 & L^{-1} \end{bmatrix} \tag{8}$$

works very well, but for smaller α a different preconditioner is required. Finding an effective preconditioner for small values of α was a challenge.

The type of preconditioners we consider here are subspace correction preconditioners. In particular we focused on the algorithms presented in [4] and [11]. Recall that we are trying to solve (6) using the PCG method as it is a positive definite system. Solving (6) directly on a subspace is difficult because of the presence of the inverse Laplacian. Therefore we noted that (6) is a short cut to solving (5). It is straightforward to project (5) onto the subgrid and, once again, eliminate all of the variables except $\mathbf{g}_1$ and $\mathbf{g}_2$ (defined on the subgrid). Note that by using (5) and then eliminating the variables, the right-hand-side of the equation will contain some information about $\mathbf{c}$ and $\mathbf{w}$; this is how the global L^{-1} is incorporated into the system. As a consequence, values for $\mathbf{c}$ and $\mathbf{w}$ must be generated during the preconditioning step.

The first approach we looked at was the two-level preconditioner presented by [4], which is designed for matrices similar to those given by (7). Unfortunately the form of the matrices G_s would not allow the use of the injection operator as in the paper referenced above.

The next approach we tried was to use multiplicative and additive Schwarz methods with the subgrids defined in such a way as to ensure that the matrices G_s are non-singular. The PCG method was also used to solve (6) defined on each subgrid as described at the beginning of this section. The approach improved the convergence rate slightly, but not enough to offset the extra cost of generating the $\mathbf{c}$ and $\mathbf{w}$ vectors.

5 Multiplicative Schwarz

A preconditioner which gives robust results is the multiplicative Schwarz preconditioner, combined with a sparse direct method to solve (5) on each subgrids. This avoids the inverse Laplacian found in (6). The sparse direct method we used is `umfpack_di_numeric` from the UMFPACK package [2].

The (symmetric) multiplicative Schwarz approach is the same as Algorithm 3.4 in [11]. Equation 2.1 in [11] was repeatedly applied until either $\|\mathbf{u}^{k+1} - \mathbf{u}^k\|/\|\mathbf{u}^{k+1}\| < 10^{-2}$ or the number of iterations reached an upper bound (currently 20). We have chosen this approach as it is cheap and the tolerance does not have to be too accurate for the preconditioner.

We now present two examples highlighting the performance of the multiplicative Schwarz algorithm. The test runs were carried out using a code developed by Stals. The code is a parallel finite element code and we plan to move these test runs to a parallel machine, at which time we will explore the use of the additive Schwarz algorithm.

In the first example the data points $\mathbf{x}^{(i)}$ $(1 \leq i \leq n)$ sit on the lattice defined by dividing the square $[1/p, 1-1/p] \times [1/p, 1-1/p]$ into $p-1$ equally spaced subsquares, where $p = 100$ and hence $n = 99^2$. The values assigned to each data point were $y^{(i)} = \mathbf{x}_1^{(i)} + \mathbf{x}_2^{(i)}$. The Dirichlet boundary conditions were set so that the expected value for each entry of $\mathbf{g}_s$ is 1 and $\mathbf{w} = 0$ (for all values of α).

The multi-linear basis functions in the finite element formulation fit the solution exactly, so any error in the solution is due to algebraic error. This problem is a good test of the convergence of the PCG method. The stopping criterion used in the PCG algorithm is based on the Hestenes-Stiefel rule [1, 9]. A small tolerance of 10^{-12} was set to ensure that the error remained small for small α.

Table 1 shows the difference in the convergence between the inverse Laplacian preconditioner and the multiplicative Schwarz preconditioner for different values of m and α. The number of subgrids was kept at 4 and four levels of overlap was used. In all examples the multiplicative Schwarz preconditioner was faster than the inverse Laplacian preconditioner. There is a sudden jump in the number of iterations for the $\alpha = 10^{-7}$ and $m = 4225$ case indicating that we may need to look at increasing the overlap.

The second example also used a uniform grid with $p = 1000$ and $n = 999^2$. The values assigned to each data point were $y^{(i)} = f_y(\mathbf{x}^{(i)})$ where $f_y(\mathbf{x}) = \exp\left(-30\|0.65 - \mathbf{x}\|_2^2\right) + \exp\left(-30\|0.35 - \mathbf{x}\|_2^2\right)$. The boundary conditions are $\mathbf{h}_1 = f_y|_\Gamma$, $(\mathbf{h}_2, \mathbf{h}_3) = \nabla f_y|_\Gamma$ and $\mathbf{h}_4 = -\alpha \Delta f_y|_\Gamma$ where Γ is the boundary of Ω_h.

The solution depends on the choice of α. The tolerance in the stopping criterion was decreased to 10^{-6} and the number of subgrids remained at four with four levels of overlap. Table 2 tabulates the convergence results.

6 Conclusion

The multiplicative Schwarz algorithm with a sparse-direct solver on the subgrids is an efficient preconditioner for the systems of equations arising from the discrete thin-plate spline method.

Table 1. Convergence rate for the first test problem. The time is in seconds. The column labelled Laplace is the results for the preconditioner given by (8). The column labelled Mult S is the multiplicative Schwarz preconditioner. CG It is the total number of PCG iterations and MS It is the total number of times the Multiplicative Schwarz algorithm was called.

m	$\alpha = 10^{-6}$					$\alpha = 10^{-7}$				
	Laplace		Mult S			Laplace		Mult S		
	CG It	Time	CG It	MS It	Time	CG It	Time	CG It	MS It	Time
1089	72	16	4	21	4	266	96	5	39	7
4225	120	116	4	21	18	254	247	49	370	187
16641	147	713	6	56	146	283	1424	7	54	147
66049	141	2999	16	164	1612	309	6611	12	144	1401

Table 2. Convergence rate for the second test problem. The column labelling is the same as the first example.

m	$\alpha = 10^{-6}$					$\alpha = 10^{-7}$				
	Laplace		Mult S			Laplace		Mult S		
	CG It	Time	CG It	MS It	Time	CG It	Time	CG It	MS It	Time
1089	36	27	3	23	22	66	33	3	34	23
4225	37	66	2	17	33	93	122	3	42	45
16641	37	268	2	27	104	98	590	3	36	118
66049	38	1133	3	59	639	110	2884	3	50	581

References

[1] M. Arioli. Backward error analysis and stopping criteria for Krylov space method. In *20th Biennial Conference on Numerical Analysis*, pages 1–41, University of Dundee, June 2003.

[2] T. Davis. UMFPACK, Version 5. University of Florida, Gainsevilla, Florida, May 2006.

[3] J. Duchon. Splines minimizing rotation-invariant. In *Lecture Notes in Math*, volume 571, pages 85–100. Springer-Verlag, 1977.

[4] M. Hanke and C. R. Vogel. Two-level preconditioners for regularized inverse problems I: Theory. *Numer. Math.*, 83:385–402, 1999.

[5] M. F. Hutchinson. A stochastic estimator of the trace of the influence matrix for Laplacian smoothing splines. *Comm. Statist. Simulation Comput.*, 19(2):433–450, 1990.

[6] S. Roberts, M. Hegland, and I. Altas. Approximation of a thin plate spline smoother using continuous piecewise polynomial functions. *SIAM J. Numer. Anal.*, 41(1):208–234, 2003.

[7] L. Stals and S. Roberts. Verifying convergence rates of discrete thin-plate splines in 3D. In Rob May and A. J. Roberts, editors, *Proc. of 12th Computational Techniques and Applications Conference CTAC-2002*, volume 46, pages C515–C529, June 2005. Online `http://anziamj.austms.org.au/V46/CTAC2004/home.html`.

[8] L. Stals and S. Roberts. Smoothing and filling holes with Dirichlet boundary conditions. Submitted to the Proceedings of the International Conference on High Performance Scientific Computing, 2006.

[9] Z. Strakoš and P. Tichý. On estimation of the A-norm of the error in CG and PCG. *PAMM*, 3(1):553–554, December 2003.

[10] G. Wahba. *Spline Models for Observational Data*, volume 59 of *CBMS-NSF Regional Conference Series in Applied Mathematics*. Society for Industrial and Applied Mathematics (SIAM), Philadelphia, PA, 1990.

[11] J. Xu. Iterative methods by space decomposition and subspace correction. *SIAM Review*, 34(4):581–613, December 1992.