Matrix-free Multilevel Moving Least-Squares Methods

Gregory E. Fasshauer

Abstract. We investigate matrix-free formulations for polynomial-based moving least-squares approximation. The well-known Shepard's method is one such formulation that leads to $\mathcal{O}(h)$ approximation order. We are interested in methods with higher approximation orders. Several possible approaches are identified, and one of them – based on the analytic solution of small linear systems – is presented here. Numerical experiments with a multilevel residual updating algorithm are also presented.

§1. Introduction

Moving least-squares (MLS) methods have emerged as the basis of numerous meshless approximation methods that are being suggested by engineers as an alternative to the traditional finite element method (see, e.g., [1] and the references therein). The general MLS method seems to be attributable to Lancaster and Šalkauskas [6] with special cases going back to McLain [8] and Shepard [10]. In the mathematics literature this method has so far not attracted too much attention. Besides the original paper [6] there are only a few more papers by Šalkauskas as well as Farwig (see, e.g., [3]). One of the reasons MLS has not been seen as a viable approximation method for multivariate scattered data may be the much cited paper [5] by Franke in which a large number of scattered data approximation methods were evaluated, and the MLS variants did not get very high marks. Very recently, however, mathematicians have shown renewed interest in the MLS method, see e.g. [7,9,11].

In this paper we intend to make a few suggestions which may lead to more efficient computation of the MLS approximation. The paper is organized as follows. In the following two sections we review two alternate

Approximation Theory X: Wavelets, Splines, and Applications

Charles K. Chui, Larry L. Schumaker, and Joachim Stöckler (eds.), pp. 271–281.

ISBN 0-8265-1416-2.

approaches to MLS approximation that show that evaluation of the MLS approximation at any given point involves the solution of a (small) system of linear equations. In Sect. 4 we summarize some of the main properties of MLS methods, and in Sect. 5 we present one possible way of obtaining a matrix-free MLS method. Then we investigate the use of MLS approximation within a multilevel approximation algorithm as studied previously in [4]. This investigation is supported by numerical experiments, as well as comments on connections to previous theoretical work. We conclude the paper with some remarks.

§2. MLS via Discrete Least Squares

We consider the following approximation problem. Given data $\{(\boldsymbol{x}_i, f(\boldsymbol{x}_i))\}_{i=1}^n \subset \mathbb{R}^d \times \mathbb{R}$ with distinct data sites $\boldsymbol{x}_i$ and f some (smooth) function, as well as an approximation space $\mathcal{U} = \mathrm{span}\{u_1, \ldots, u_m\}$ (with $m < n$), along with an inner product

$$\langle u, v\rangle_{\phi(\boldsymbol{x})} = \sum_{i=1}^n u(\boldsymbol{x}_i) v(\boldsymbol{x}_i) \phi_i(\boldsymbol{x}), \qquad \boldsymbol{x} \in \mathbb{R}^d \text{ fixed}, \tag{1}$$

where the positive weight functions ϕ_i, $i = 1, \ldots, n$, depend on the evaluation point $\boldsymbol{x}$, we wish to find the best approximation from $\mathcal{U}$ to f at the point $\boldsymbol{x}$ with respect to the norm induced by (1).

This means we will obtain the approximation (at the point $\boldsymbol{x}$) as

$$s(\boldsymbol{x}) = \sum_{j=1}^m c_j(\boldsymbol{x}) u_j(\boldsymbol{x}), \tag{2}$$

where the coefficients $c_j(\boldsymbol{x})$ are such that

$$\sum_{i=1}^n \left[s(\boldsymbol{x}_i) - f(\boldsymbol{x}_i)\right]^2 \phi_i(\boldsymbol{x}) \to \min.$$

Due to the definition of the inner product (1) whose weight function "moves" with the evaluation point $\boldsymbol{x}$, the coefficients c_j in (2) depend also on $\boldsymbol{x}$. As a consequence, one has to solve the normal equations

$$\sum_{j=1}^m c_j \langle u_j, u_k\rangle_{\phi(\boldsymbol{x})} = \langle f, u_k\rangle_{\phi(\boldsymbol{x})}, \qquad k = 1, \ldots, m, \tag{3}$$

anew each time the evaluation point is changed. In matrix notation (3) becomes

$$G\boldsymbol{c} = \boldsymbol{f}, \tag{4}$$

with the positive definite Gram matrix $G = \left(\langle u_j, u_k\rangle_{\phi(\boldsymbol{x})}\right)_{j,k=1}^m$, coefficient vector $\boldsymbol{c}$ and right-hand side vector $\boldsymbol{f}$ as in (3) all depending on $\boldsymbol{x}$.

In this paper we will focus on the case when $\mathcal{U} = \mathcal{P}_Q^d$, the space of d-variate polynomials of total degree at most Q.

Example: Shepard's Method. If we take $\mathcal{U} = \mathcal{P}_0^d = \operatorname{span}\{1\}$, then there is only a single normal equation

$$c_1(\boldsymbol{x})\langle 1, 1\rangle_{\phi(\boldsymbol{x})} = \langle f, 1\rangle_{\phi(\boldsymbol{x})},$$

which implies that

$$c_1(\boldsymbol{x}) = \frac{\sum_{i=1}^n f(\boldsymbol{x}_i)\phi_i(\boldsymbol{x})}{\sum_{i=1}^n \phi_i(\boldsymbol{x})},$$

and therefore (from (2))

$$s(\boldsymbol{x}) = \sum_{i=1}^n f(\boldsymbol{x}_i)\frac{\phi_i(\boldsymbol{x})}{\sum_{\ell=1}^n \phi_\ell(\boldsymbol{x})}. \tag{5}$$

This method is known as Shepard's method [10]. Note that Shepard's method (5) is now an explicit formula (valid for any $\boldsymbol{x} \in \mathbb{R}^d$). It no longer requires solution of a linear system.

It is our goal to derive explicit formulas similar to (5), but based on higher-degree polynomial spaces. As a first observation we note that we can simplify (2) by using shifted monomials, i.e., $\mathcal{U} = \mathcal{P}_Q^d = \operatorname{span}\{(\cdot - \boldsymbol{x})^{\boldsymbol{\alpha}} : |\boldsymbol{\alpha}| \le Q\}$. Here $\boldsymbol{\alpha} = (\alpha_1, \dots, \alpha_d) \in \mathbb{N}^d$ is a multi-index with length $|\boldsymbol{\alpha}| = \sum_{i=1}^d |\alpha_i|$. For this choice of local basis we have $s(\boldsymbol{x}) = c_1(\boldsymbol{x})$ for any degree Q.

A second observation is mostly of theoretical interest. If one were to use an orthonormal basis (with respect to $\langle\ ,\ \rangle_{\phi(\boldsymbol{x})}$), then (since $G = I$) (3) implies

$$c_k(\boldsymbol{x}) = \langle f, u_k\rangle_{\phi(\boldsymbol{x})}, \qquad k = 1, \dots, m,$$

and (from (2))

$$s(\boldsymbol{x}) = \sum_{j=1}^m \langle f, u_j\rangle_{\phi(\boldsymbol{x})} u_j(\boldsymbol{x}). \tag{6}$$

In this way MLS approximation can be viewed as a generalization of approximation by (truncated) Fourier series. Again, no linear system needs to be solved. However, discrete multivariate orthogonal polynomials over scattered centers are very difficult to construct.

§3. MLS via Constrained Optimization

The following connection between MLS approximation and constrained least squares minimization was first described by Bos and Šalkauskas [2]. It is referred to as the Backus-Gilbert formulation. The Backus-Gilbert formulation was also the approach used in [7] and [11] to obtain convergence order estimates for MLS approximation.

We now start with a quasi-interpolant of the form

$$\hat{f}(\boldsymbol{x}) = \sum_{i=1}^{n} f(\boldsymbol{x}_i)\gamma_i(\boldsymbol{x}), \tag{7}$$

where the data $\{(\boldsymbol{x}_i, f(\boldsymbol{x}_i))\}$ are as before. The goal is to determine the functions γ_i in such a way that (7) is easy to compute and has high approximation order. To this end we enforce polynomial reproduction, i.e.,

$$\sum_{i=1}^{n} p(\boldsymbol{x}_i)\gamma_i(\boldsymbol{x}) = p(\boldsymbol{x}), \qquad \text{for all } p \in \mathcal{P}_Q^d.$$

In addition, a weighted norm of the γ functions is minimized according to

$$\frac{1}{2}\sum_{i=1}^{n} \gamma_i^2(\boldsymbol{x}) w_i(\boldsymbol{x}) \to \min. \tag{8}$$

Here the w_i are positive weight functions whose relation to the weights ϕ_i used earlier we point out below. The norm minimization (8) is motivated physically in the Backus-Gilbert theory, since this minimizes the "spread" of γ (see [2]).

Using Lagrange multipliers $\lambda_j(\boldsymbol{x})$ (which will again depend on the evaluation point $\boldsymbol{x}$) we can compute

$$\gamma_i(\boldsymbol{x}) = \frac{1}{w_i(\boldsymbol{x})}\sum_{j=1}^{m} \lambda_j(\boldsymbol{x}) p_j(\boldsymbol{x}_i), \qquad i = 1, \ldots, n, \tag{9}$$

where the Lagrange multipliers $\lambda_j(\boldsymbol{x})$ are obtained as the solution of

$$G\boldsymbol{\lambda} = \boldsymbol{q}. \tag{10}$$

Here $\boldsymbol{q} = [p_1(\boldsymbol{x}), \ldots, p_m(\boldsymbol{x})]^T$, and the matrix G is identical with the Gram matrix G in (4) provided we identify $\dfrac{1}{w_i(\boldsymbol{x})} = \phi_i(\boldsymbol{x})$.

If we choose shifted monomials as basis of the polynomial reproduction space, then $\boldsymbol{q} = [1, 0, \ldots, 0]^T$ and the formulas we obtain for the λ_j below simplify. Moreover, the numerical computations become more stable since all terms are expanded about the evaluation point $\boldsymbol{x}$.

One can also see the connection to Fourier series approximation using the Backus-Gilbert formulation, but it is not quite as obvious. In this case we get (using (10) with $G = I$)

$$\lambda_k(\boldsymbol{x}) = p_k(\boldsymbol{x}), \qquad k = 1, \ldots, m,$$

and (from (9) with $\frac{1}{w_i(\boldsymbol{x})} = \phi_i(\boldsymbol{x})$)

$$\gamma_i(\boldsymbol{x}) = \phi_i(\boldsymbol{x}) \sum_{j=1}^{m} p_j(\boldsymbol{x}) p_j(\boldsymbol{x}_i). \tag{11}$$

Via (7), (11), and (1) this leads to

$$\begin{aligned}\hat{f}(\boldsymbol{x}) &= \sum_{i=1}^{n} f(\boldsymbol{x}_i)\gamma_i(\boldsymbol{x}) = \sum_{i=1}^{n} f(\boldsymbol{x}_i)\phi_i(\boldsymbol{x}) \sum_{j=1}^{m} p_j(\boldsymbol{x}) p_j(\boldsymbol{x}_i) \\ &= \sum_{j=1}^{m} \left[\sum_{i=1}^{n} f(\boldsymbol{x}_i) p_j(\boldsymbol{x}_i)\phi_i(\boldsymbol{x}) \right] p_j(\boldsymbol{x}) = \sum_{j=1}^{m} \langle f, p_j \rangle_{\phi(\boldsymbol{x})} p_j(\boldsymbol{x}).\end{aligned}$$

§4. Properties of MLS Approximation

We briefly summarize some of the main features of MLS approximation. First, an approximation (computed either via (2) or via (7)) is valid at one point, $\boldsymbol{x}$, only, and a new linear system needs to be solved when the evaluation point is changed. If one considers rendering of approximation surfaces, then thousands of evaluations are required, and this may be one of the main reasons for the reluctance among mathematicians to accept MLS approximation as a viable scattered data approximation method.

If the weight functions ϕ_i have local support then all summations are local, i.e., instead of summing from $i = 1$ to n we need only sum over all points $\boldsymbol{x}_i$ in a neighborhood $N_{\boldsymbol{x}}$ of $\boldsymbol{x}$ referred to below with the index set $I_{\boldsymbol{x}}$. Coupled with the direct formulas proposed here, this leads to very fast approximation methods. This shows that earlier objections concerning the computational cost of MLS methods (see [5]) were not justified.

MLS methods are also very accurate. In fact, Levin [7] and Wendland [11] (and to some extent Farwig [3]) showed that if $\mathcal{U} = \mathcal{P}_Q^d$, the support of the weight function is local, and proportional to some "meshsize" h then one has approximation order $\mathcal{O}(h^{Q+1})$ provided the data come from a smooth enough function f. According to the cited references, the choice of weight function has no effect on the approximation order. The weight function only determines the smoothness of the approximation. If higher-order polynomial reproduction is enforced, then undesirable features such as "flat spots" will be avoided (another objection in [5]).

By construction, functions in $\mathcal{U}$ are reproduced exactly by the MLS method, and it was shown in earlier papers (e.g., [6]) that the choice of singular weight functions leads to *interpolation* at the data points.

§5. Matrix-free Formulations

The first possibility to obtain a matrix-free MLS formulation – outlined above – would be the use of generalized Fourier series (6) (with compactly supported weight functions). Due to the difficulty associated with finding discrete multivariate orthogonal polynomials, this approach seems to be impractical.

Another approach may be the construction of weight functions which already have certain approximation properties built in. This would then lead to an approximation analogous (in structure) to Shepard's method (5). We reserve this approach for a future paper.

In the present work we suggest solving small (e.g., $m \le 4$) systems for λ_j in the Backus-Gilbert approach (10) analytically. This means that the resulting formulas can be directly programmed into the computer code, and no linear systems need to be solved at run-time.

For the following examples we define the moments

$$\mu_{\boldsymbol{\alpha}} = \sum_{i \in I_{\boldsymbol{x}}} (\boldsymbol{x}_i - \boldsymbol{x})^{\boldsymbol{\alpha}} \phi_i(\boldsymbol{x}), \qquad \boldsymbol{x} \in \mathbb{R}^d, \qquad |\boldsymbol{\alpha}| \le 2Q,$$

with $\boldsymbol{\alpha}$ a multi-index, and $I_{\boldsymbol{x}}$ the index set mentioned above.

Example 1. $d = 1$, $Q = 1$, $m = 2$. We choose the polynomial basis $\mathcal{U} = \mathrm{span}\{p_1(s) = 1,\ p_2(s) = s - x\}$. Then

$$\lambda_1(x) = \frac{\mu_2}{\mu_0\mu_2 - \mu_1^2}, \qquad \lambda_2(x) = \frac{\mu_1}{\mu_1^2 - \mu_0\mu_2},$$

and according to (9) the quasi-interpolant (7) is formed with

$$\gamma_i(x) = [\lambda_1(x) + \lambda_2(x)(x_i - x)]\, \phi_i(x), \qquad i = 1, \ldots, n,$$

where the ϕ_i are arbitrary (positive) weight functions.

Example 2. $d = 1$, $Q = 2$, $m = 3$. We choose $\mathcal{U} = \mathrm{span}\{p_1(s) = 1,\ p_2(s) = s - x,\ p_3(s) = (s - x)^2\}$. Then

$$\lambda_1(x) = \frac{\mu_2\mu_4 - \mu_3^2}{D}, \quad \lambda_2(x) = \frac{\mu_2\mu_3 - \mu_1\mu_4}{D}, \quad \lambda_3(x) = \frac{\mu_1\mu_3 - \mu_2^2}{D},$$

where $D = 2\mu_1\mu_2\mu_3 - \mu_0\mu_3^2 - \mu_2^3 - \mu_1^2\mu_4 + \mu_0\mu_2\mu_4$. Now the quasi-interpolant is computed with

$$\gamma_i(x) = \left[\lambda_1(x) + \lambda_2(x)(x_i - x) + \lambda_3(x)(x_i - x)^2\right] \phi_i(x), \qquad i = 1, \ldots, n.$$

Example 3. $d = 2$, $Q = 1$, $m = 3$. If $\mathcal{U} = \mathrm{span}\{p_1(s,t) = 1,\ p_2(s,t) = s - x,\ p_3(s,t) = (t-y)\}$ then

$$\lambda_1(x,y) = \frac{1}{D}\left[\mu_{11}^2 - \mu_{20}\mu_{02}\right], \qquad \lambda_2(x,y) = \frac{1}{D}\left[\mu_{10}\mu_{02} - \mu_{01}\mu_{11}\right],$$
$$\lambda_3(x,y) = \frac{1}{D}\left[\mu_{20}\mu_{01} - \mu_{10}\mu_{11}\right],$$

with $D = \mu_{10}^2\mu_{02} + \mu_{20}\mu_{01}^2 - \mu_{00}\mu_{20}\mu_{02} - 2\mu_{10}\mu_{01}\mu_{11} + \mu_{00}\mu_{11}^2$. Here we set $\gamma_i(x,y) = \left[\lambda_1(x,y) + \lambda_2(x,y)(x_i - x) + \lambda_3(x,y)(y_i - y)\right]\phi_i(x,y)$.

§6. Experiments with a Multilevel Approximation Algorithm

In [4] we studied an iterative multilevel Newton algorithm with postconditioning based on convolution smoothing. In its simplest form (used here) the algorithm can be described as follows.

Algorithm:

1) Create nested point sets $\mathcal{X}_1 \subset \cdots \subset \mathcal{X}_K \subset \mathbb{R}^d$, define a sequence of convolution kernels Φ_k, and let $s(\boldsymbol{x}) = 0$.
2) For $k = 1, 2, \ldots, K$:
   (a) Solve $g(\boldsymbol{x}) = f(\boldsymbol{x}) - s(\boldsymbol{x})$ on $\mathcal{X}_k$.
   (b) Smooth the Newton update $\tilde{g}(\boldsymbol{x}) = (\Phi_k * g)(\boldsymbol{x})$.
   (c) Update $s(\boldsymbol{x}) = s(\boldsymbol{x}) + \tilde{g}(\boldsymbol{x})$.

In this context the MLS quasi-interpolant (7) can be interpreted as a discrete convolution representing step 2(b) in the above algorithm. The solution step 2(a) in this case is trivial. We just sample the residual $f - s$ on the computational grid $\mathcal{X}_k$.

We present two sets of experiments below. In 1D we use MLS quasi-interpolants based on the formulas given in Examples 1 and 2, and the test function

$$f(x) = \frac{1}{2}e^{-(9x-2)^2/4} + \frac{3}{4}e^{-(9x+1)^2/49} + \frac{1}{2}e^{-(9x-7)^2/4} - \frac{1}{5}e^{-(9x-4)^2}.$$

In 2D we use Shepard's method and the formulas of Example 3 which reproduce linear polynomials locally. The test function for those experiments is Franke's function

$$f(x,y) = \frac{3}{4}e^{-\left((9x-2)^2+(9y-2)^2\right)/4} + \frac{3}{4}e^{-(9x+1)^2/49-(9y+1)^2/10}$$
$$+ \frac{1}{2}e^{-\left((9x-7)^2+(9y-3)^2\right)/4} - \frac{1}{5}e^{-(9x-4)^2-(9y-7)^2}.$$

The weight functions we use for both sets of experiments are the compactly supported radial basis functions (listed together with their smoothness)

$$\varphi(r) = (1-r)_+^4(4r+1), \qquad (C^2)$$
$$\varphi(r) = (1-r)_+^6(\frac{35}{3}r^2+6r+1), \qquad (C^4)$$
$$\varphi(r) = (1-r)_+^8(32r^3+25r^2+8r+1), \qquad (C^6)$$

where $r = \|\cdot - \boldsymbol{x}\|$, and $(\cdot)_+^\beta$ is the truncated power function. We then have $\phi_i(\boldsymbol{x}) = \varphi(\|\boldsymbol{x}_i - \boldsymbol{x}\|)$. For the multilevel algorithm we need scaled versions of φ, i.e., $\varphi_\rho(r) = \varphi(r/\rho)$, where ρ is the radius of support of φ.

The results of our 1D experiments are presented in Tables 1 and 2 as well as in the left part of Figure 1. The computational grids $\mathcal{X}_k$ are given by 2^k+1 equally spaced points in the interval $[0,1]$. According to the theory (see Sect. 4) we need to scale the support of the weight function proportionally to the meshsize. We do this, and take the initial support radius equal to 1 for Table 1 and $\rho = 2$ for Table 2 (since local reproduction of quadratic polynomials requires more points within the support for the method to be well-posed). The ℓ_∞ errors are computed on a grid of equally spaced points one level finer than the finest level used in the algorithm, i.e., 524289 points for Table 1, and 65537 points for Table 2. The reason we terminated the algorithm earlier in Table 2 is not runtime (execution time is only a few seconds per level), but rather the size of the error which is approaching machine accuracy.

According to the convergence theory for MLS approximation (see Sect. 4) we should expect $\mathcal{O}(h^{Q+1})$ convergence (without a multilevel algorithm). The rate-columns contain the approximate convergence exponent ($\approx Q+1$), i.e., $\ln(e_{k-1}/e_k)/\ln 2$, for the multilevel algorithm. These rates can also be observed as the slopes of the curves in Figure 1. The tables also contain a line labeled "direct" in which we list the error obtained by computing the MLS approximation directly on the finest level. From the data in the tables, one can infer that neither the multilevel algorithm nor a change of weight function have a significant effect on the accuracy of the MLS method. This needs to be contrasted with radial basis function interpolation (using the same weight functions we use) where a multilevel algorithm is essential to obtaining an accurate approximation in a computationally efficient way (see, e.g., [9]).

For our 2D experiments (summarized in Tables 3 and 4) we use $(2^k+1)^2$ equally spaced points in the unit square, and the errors are computed on a grid of 263169 equally spaced points. The initial support size of the radial weights is $\rho = 1$, and it is halved in each subsequent iteration.

		C^2		C^4		C^6	
k	# pts	ℓ_∞ error	rate	ℓ_∞ error	rate	ℓ_∞ error	rate
2	5	1.512(-1)		1.234(-1)		1.137(-1)	
3	9	6.543(-2)	1.21	5.696(-2)	1.11	5.977(-2)	0.93
4	17	1.168(-2)	2.49	9.961(-3)	2.52	1.162(-2)	2.36
5	33	1.210(-3)	3.27	1.476(-3)	2.75	2.185(-3)	2.41
6	65	2.277(-4)	2.41	2.394(-4)	2.62	5.409(-4)	2.01
7	129	5.926(-5)	1.94	6.155(-5)	1.96	1.400(-4)	1.95
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$
17	131073	5.762(-11)	2.00	6.048(-11)	2.00	6.122(-11)	2.00
18	262145	1.441(-11)	2.00	1.513(-11)	2.00	1.531(-11)	2.00
	direct	1.036(-10)		9.036(-11)		8.754(-11)	

Tab. 1. 1D-multilevel approximation with linear polynomial reproduction.

		C^2		C^4		C^6	
k	# pts	ℓ_∞ error	rate	ℓ_∞ error	rate	ℓ_∞ error	rate
2	5	2.273(-1)		2.058(-1)		1.795(-1)	
3	9	8.714(-2)	1.38	6.780(-2)	1.60	5.366(-2)	1.74
4	17	1.259(-2)	2.79	8.476(-3)	3.00	5.926(-3)	3.18
5	33	5.933(-4)	4.41	3.194(-4)	4.73	2.347(-4)	4.66
6	65	4.266(-5)	3.80	1.227(-5)	4.70	2.079(-5)	3.50
7	129	4.610(-6)	3.21	1.139(-6)	3.43	3.253(-6)	2.68
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$
14	16385	3.680(-12)	1.88	3.082(-13)	2.96	3.568(-12)	2.83
15	32769	6.303(-13)	2.55	4.208(-14)	2.87	5.039(-13)	2.82
	direct	2.863(-13)		1.976(-13)		1.626(-13)	

Tab. 2. 1D-multilevel approximation with quadratic polynomial reproduction.

§7. Concluding Remarks

The multilevel algorithm has virtually no effect on the convergence behavior for (scattered) data approximation. In some of the examples the error using the multilevel method was slightly smaller than for the direct solution (but also more expensive to obtain). However, initial experiments on a simple two point boundary value problem indicate that a multilevel approach will probably be essential for the solution of differential equations via collocation. The effect of changing the weight function on the accuracy is minimal. The error for the direct solution improves slightly for smoother weights.

		C^2		C^4		C^6	
k	# pts	ℓ_∞ error	rate	ℓ_∞ error	rate	ℓ_∞ error	rate
1	9	6.318(-1)		6.140(-1)		6.017(-1)	
2	25	2.548(-1)	1.31	1.894(-1)	1.70	1.648(-1)	1.87
3	81	7.517(-2)	1.76	6.433(-2)	1.56	5.552(-2)	1.57
4	289	2.047(-2)	1.88	1.634(-2)	1.98	1.693(-2)	1.71
5	1089	3.055(-3)	2.74	4.689(-3)	1.80	6.932(-3)	1.29
6	4225	1.136(-3)	1.43	1.822(-3)	1.36	3.008(-3)	1.20
7	16641	4.202(-4)	1.43	7.332(-4)	1.31	1.251(-3)	1.27
8	66049	1.604(-4)	1.39	2.985(-4)	1.30	5.417(-4)	1.21
	direct	1.474(-3)		8.508(-4)		4.920(-4)	

Tab. 3. 2D-multilevel approximation with reproduction of constants.

		C^2		C^4		C^6	
k	# pts	ℓ_∞ error	rate	ℓ_∞ error	rate	ℓ_∞ error	rate
1	9	6.163(-1)		6.174(-1)		6.178(-1)	
2	25	2.741(-1)	1.17	2.297(-1)	1.43	2.201(-1)	1.49
3	81	7.580(-2)	1.85	6.999(-2)	1.71	7.192(-2)	1.61
4	289	2.082(-2)	1.86	1.731(-2)	2.02	1.748(-2)	2.04
5	1089	2.378(-3)	3.13	2.426(-3)	2.83	3.607(-3)	2.28
6	4225	4.271(-4)	2.48	4.146(-4)	2.55	8.767(-4)	2.04
7	16641	1.085(-4)	1.98	1.043(-4)	1.99	2.141(-4)	2.03
8	66049	2.783(-5)	1.96	2.624(-5)	1.99	4.743(-5)	2.17
	direct	1.433(-4)		1.283(-4)		1.256(-4)	

Tab. 4. 2D-multilevel approximation with linear polynomial reproduction.

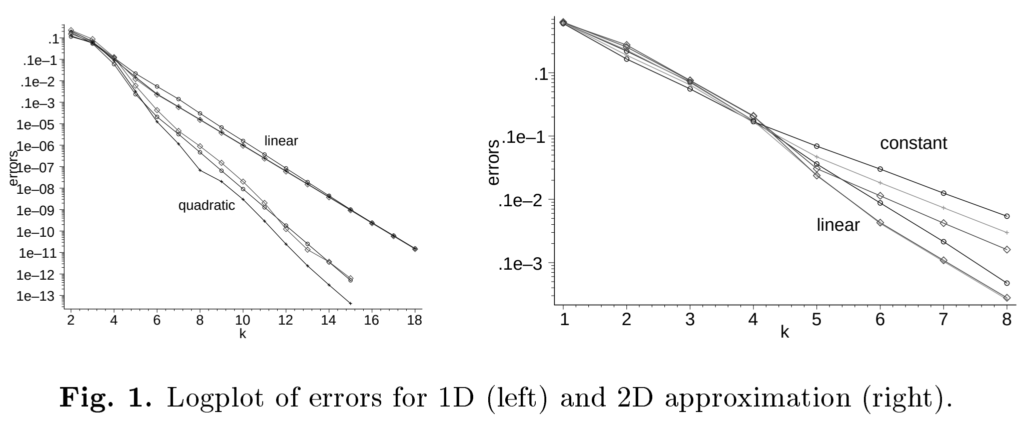

Fig. 1. Logplot of errors for 1D (left) and 2D approximation (right).

References

1. Belytschko, T., Y. Krongauz, D. Organ, M. Fleming, and P. Krysl, Meshless methods: an overview and recent developments, Computer Methods in Applied Mechanics and Engineering **139** (1996), 3–47.
2. Bos, L. P. and K. Šalkauskas, Moving least-squares are Backus-Gilbert optimal, J. Approx. Theory **59** (1989), 267–275.
3. Farwig, R., Multivariate interpolation of scattered data by moving least squares methods, in *Algorithms for the Approximation of Functions and Data,* J. C. Mason and M. G. Cox (eds), Oxford Univ. Press, Oxford, 1987, 193–211.
4. Fasshauer, G. E. and J. W. Jerome, Multistep approximation algorithms: Improved convergence rates through postconditioning with smoothing kernels, Advances in Comp. Math. **10** (1999), 1–27.
5. Franke, R., Scattered data interpolation: tests of some methods, Math. Comp. **48** (1982), 181–200.
6. Lancaster, P. and K. Šalkauskas, Surfaces generated by moving least squares methods, Math. Comp. **37** (1981), 141–158.
7. Levin, D., The approximation power of moving least-squares, Math. Comp. **67** (1998), 1517–1531.
8. McLain, D. H., Drawing contours from arbitrary data points, Comput. J. **17** (1974), 318–324.
9. Schaback, R., Remarks on meshless local construction of surfaces, *The Mathematics of Surfaces*, IX (Cambridge, 2000), Springer, London, 34–58.
10. Shepard, D., A two dimensional interpolation function for irregularly spaced data, Proc. 23rd Nat. Conf. ACM, 1968, 517–523.
11. Wendland, H., Local polynomial reproduction and moving least squares approximation, IMA J. Numer. Anal. **21** (2001), 285–300.

Greg Fasshauer
Department of Applied Mathematics
Illinois Institute of Technology
Chicago, IL 60616
`fass@amadeus.math.iit.edu`
`http://amadeus.math.iit.edu/~fass`