Fast Multipole Method for Accelerating the Evaluation of Splines

F. Chen and D. Suter

Dept. Elect. and Comp. Syst. Eng.

Monash University, Clayton, 3168, Australia

24 June, 1998

Abstract

We consider the problem of interpolating scattered data using spline methods and present a general framework of using the multipole method to accelerate the evaluation of splines. The method depends on a tree-data structure and two hierarchical approximations: an upward multipole expansion approximation and a downward local Taylor series approximation. We also illustrate the performance of the efficiency and the accuracy of the fast multipole algorithm for 2D vector spline. In comparison with the CPU time of direct calculation, which increases at a quadratic rate with the number of points, the fast algorithm achieves a higher speed in evaluation at a linear rate.

Key words: PDE splines, vector splines, fast multipole method, multipole expansion, Taylor series expansion.

1 Introduction

Scattered data "interpolation" or "smoothing" refers to a problem of approximating a function from some scattered measurements. This subject is of practical importance in many science and engineering fields such as in meteorology, geophysics, biomedical science; as well as computer graphics and image processing. Basically, the interpolation problem is to define a function that goes through (in the interpolating sense) or close to (in the smoothing sense) a given set of data measurements $\{y_i\}_1^N$. Each of the y_i corresponds to a data location x_i. The data y_i can be measurements with any dimension: that is, the measurement can be multiple valued or vector at each point.

Spline interpolation techniques provide a very useful tool for solving the scattered data interpolation (or smoothing) problem. The term "spline" can be employed for a whole variety of interpolation/approximation methods. Here we restrict ourselves to PDE splines [1] because of their abilities to model physical properties (such as deformation etc.). Moreover, unlike some other spline methods (i.e. contrary to tensor product splines and NURBS popular in some fields of CAD), they do not prefer any specific direction. In the spline methods, the spline function minimizes a smoothness constraint functional. This functional is defined

1

Fast multipole method for accelerating the evaluation of splines, F. Chen and D. Suter - to appear IEEE CS&E Sept. 3 1998

2

in terms of the derivatives of the interpolant and is usually related to a Partial Differential Equation (PDE) (or an ODE for a univariate function). Hence we call these splines PDE splines. A well-known example of a PDE spline is the thin-plate spline, which is defined as a flexible thin-plate with minimum bending energy in terms of the second order derivatives and is related to a biharmonic kernel. The thin-plate spline is a scalar function. Recently, interest has developed in the study of vector splines. Vector splines have been used for restitution of wind velocity fields in meteorology [2] and for other vector field reconstructions. We have developed FMM based schemes to rapidly evaluate both scalar and vector splines.

The splines to which we [3, 6, 7] and others [4, 12] have applied FMM method for evaluation are ones whose kernels are members of the polyharmonic family or closely related to this family. Although some of the methods [3] appear to depend upon specific properties of the kernel (in the case of [3] it is a biharmonic kernel), all that is required by the techniques followed here is that the shifted summation of the kernel function has an expression that can be truncated (with error rapidly decreasing with distance) at a certain point. We do not know precisely what class of kernels this class of techniques cannot be used for but it cannot be applied to completely general kernels (for example general oscillatory kernels would be beyond the scope of this technique, see [5] for oscillatory kernels).

In general, the spline solution can be expressed as a linear combination of the shifted versions of a kernel function, plus a polynomial term. That is, the spline is of the form:

$$f(x) = \sum_{i=1}^{N} c_i \Phi(|x - x_i|) + p_{m-1}(x) \tag{1}$$

where $\Phi(x)$ is the kernel function that is determined by the smoothness functional, or the associated PDE. The kernel Φ is a radial function $r^2 \log r$ for a thin-plate spline (where $r = |x - x_i|$) and has a matrix form in the case of vector splines. For example, a 2D second order vector spline kernel has the form of:

$$\Phi(r) = \begin{pmatrix} \frac{1}{\alpha}\partial_{xx} + \frac{1}{\beta}\partial_{yy} & (\frac{1}{\alpha} - \frac{1}{\beta})\partial_{xy} \\ (\frac{1}{\alpha} - \frac{1}{\beta})\partial_{xy} & \frac{1}{\alpha}\partial_{yy} + \frac{1}{\beta}\partial_{xx} \end{pmatrix} K(r) \tag{2}$$

where α and β are model parameters, and $K(r) = r^4 \log r$ is a tri-harmonic scalar kernel that satisfies $\Delta^3 K = \delta$. The other term $p_{m-1}(x) \in \Pi_{m-1}$ in (1) is a polynomial with degree at most $m-1$, where m is the lowest order of the derivative appearing in the smoothness functional. c_i $(i = 1, \ldots, N)$ are coefficients which satisfy the constraints: $\sum_{i=1}^{N} c_i q_{m-1}(x_i) = 0$ for any $q_{m-1}(x) \in \Pi_{m-1}$. In this spline, α and β control the amount of variation in the divergence and curl of the reconstructed field. The details for this 2D vector spline and a 3D vector analogy can be found in [6][7].

To determine the spline coefficients c_i, the following linear system must be solved:

$$\begin{cases} f(x_i) = y_i \\ \sum_{i=1}^{N} c_i q_{m-1}(x_i) = 0 \end{cases} \tag{3}$$

Hence, an explicit spline interpolant is obtained. An example of the vector spline interpolant is given in Figure 1, in which a set of sample points and the vector spline reconstruction are shown in (a) and (b).
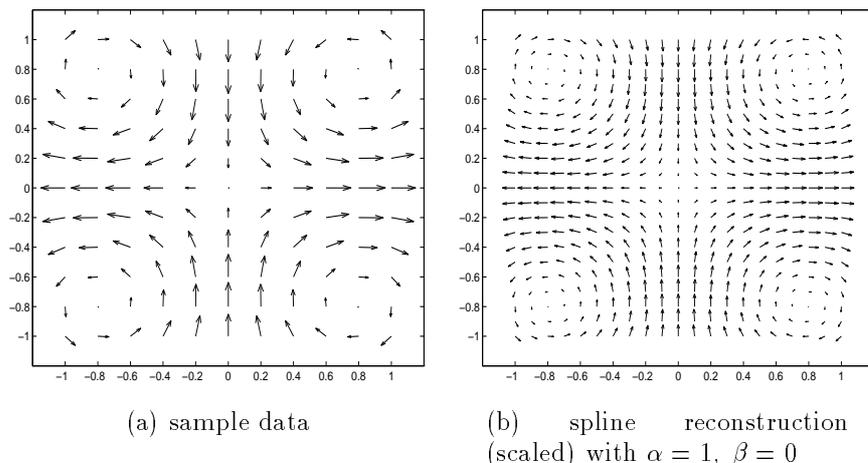
Fast multipole method for accelerating the evaluation of splines, F. Chen and D. Suter - to appear IEEE CS&E Sept. 3 1998

3

(a) sample data

(b) spline reconstruction (scaled) with $\alpha = 1$, $\beta = 0$

Figure 1: (a) shows the sample data. The reconstruction based on the vector spline (1)-(2) is shown in (b) with $\alpha = 1$, $\beta = 0$.

In this paper, we are only interested in the evaluation process with given c_i, i.e. for given c_i and x_i, to evaluate spline interpolant (1) at x [1]. It is clear to see from (1) that every data point x_i, $(i = 1, \ldots, N)$ contributes to the evaluation at the point x. Hence, the evaluation of the spline at a given point requires $O(N)$ operations. The complete evaluation of the spline at M points requires $O(MN)$ operations. This causes the evaluation process to be very computationally expensive when both the data points and the evaluation points are large in number. This high computational cost is considered as a major disadvantage of the spline methods and precludes their broad application to problems with large data sets. Obviously, for a problem involving large data in multi-dimensions, such as data derived from modern imaging methods, e.g. satellite data, MRI image data etc., direct calculation of a spline is inefficient or prohibitive.

We describe and illustrate a general fast method for efficiently evaluating of splines, including vector splines in high dimensions. The method is based on a fast multipole method (FMM) for potential calculation (see Greengard and Rokhlin [8]). The algorithm involves a tree-data structure and two hierarchical approximations: an upward multipole expansion approximation and a downward local Taylor series approximation. The CPU time of direct calculation of the vector kernel at N points with N data points increases at $O(N^2)$. The computational complexity of the presented fast algorithm reduces to a linear order $O(N)$.

In section 2, we describe the fundamental ideas of the fast multipole method and review the related work. In section 3, we show numerical results demonstrating the efficiency and the accuracy of FMM in actual performance for a 2D vector spline. Finally, in section 4, we briefly summarise this work.

---

[1] We are not concerned here with the polynomial p_{m-1} since the key to fast evaluation of f is the fast evaluation of the summation of the shifted kernels.

Fast multipole method for accelerating the evaluation of splines, F. Chen and D. Suter - to appear IEEE CS&E Sept. 3 1998

4

2 Fast multipole method and related work

2.1 General description of FMM

The fast multipole method provides a general approach of rapid calculation of a finite summation of *a suitable* kernel function K with individual sources $\{x_n\}_1^N$ and weights $\{a_n\}_1^N$, e.g. $\sum_{n=1}^N a_n K(x - x_n)$. A *suitable* kernel function means a kernel function whose pairwise interaction can be expressed in a multipole-like series expansion as long as the two points are separated by an appropriate distance, which is referred to a "far-field" in FMM literature. The essential idea of the fast multipole algorithm is to combine a cluster of far-field data "sources",[2] each of them corresponding to a multipole expansion, into a single "source" with a combined multipole expansion.

The fast multipole method has two main features: a tree data structure and two hierarchical approximations - an upward multipole approximation and a downward local Taylor series approximation. More precisely, a hierarchical tree-like data structure is constructed by repeatedly subdividing the original domain into four equal size sub-domains. The algorithm is then implemented based on this data tree. For every evaluation point at the bottom of the data tree, the contributions coming from all data sources are separated into two parts: one is from the near-field and the other is from the far-field. The near-field contributions are computed using a direct method, and the far-field sources are divided into a number of clusters, in which each of them is associated with a combined multipole expansion. These multipole expansions are propagated upward through the tree for effectively creating large clusters, then shifted, under a convergence condition, into the local Taylor series expansions. Finally, the local Taylor expansions are propagated downward to the bottom for approximating all far-field contributions. Above process can be classified into the following four stages:

Stage 1: data-tree construction. Assume that the whole data (both data sources and evaluation points) are within a regular n-dimensional region, named a parent "panel"[3] or root (level 0), a n-dimensional tree data structure is formed by repeatedly subdividing each parent panel into 2^n children panels. Suppose this procedure is terminated at level l, then $(2^n)^l$ panels are obtained at the finest level. For each panel P in this tree, we define a few notations used in the FMM:

- **near-field of** P: a set of panels that are at the same level as P and share a common corner, edge or face with P.

- **far-field of** P: a set of panels that are at the same level as P and are not in the near-field of P.

- **well-separated neighbours of** P: a set of panels that are in the far-field of P and whose parents are in the near-field of the parent of P.

---

[2] The analogy hinted at here is with electric field sources; one of the first applications of multipole methods was in the calculation of electric field/potential - see [8].

[3] Here we use panel as general description: it will refer to a square in 2D space, or a cube in 3D space

Fast multipole method for accelerating the evaluation of splines, F. Chen and D. Suter - to appear IEEE CS&E Sept. 3 1998
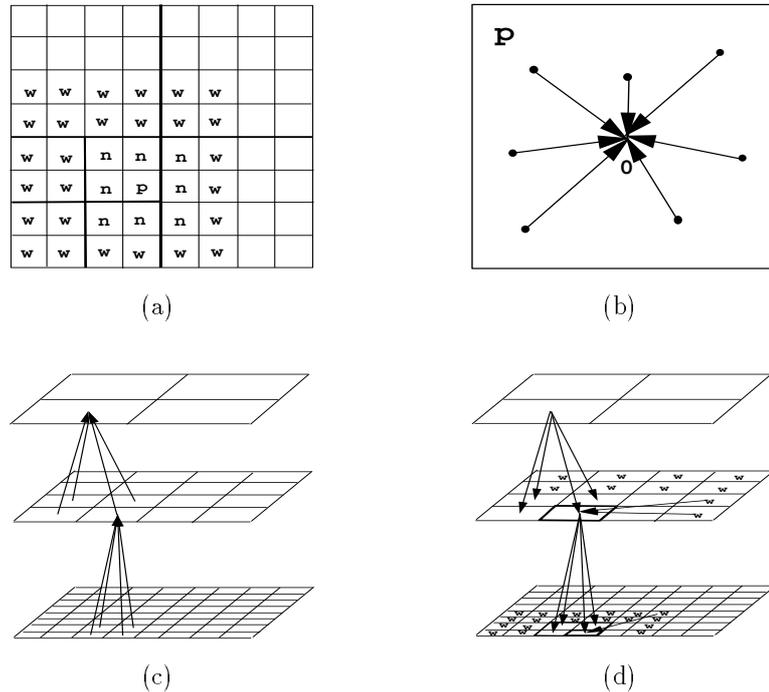
5

(a)

(b)

(c)

(d)

Figure 2: (a) depicts the nearest neighbours (denoted by n) and the well-separated neighbours (denoted by w) of P. (b) combine a cluster of data sources within P into a single source at the centre of P. (c) build upward multipole expansion approximations by shifting all existed children's multipole expansions to their parent. (d) build downward local Taylor series expansions approximations by adding together two types of local Taylor expansions - one is translated from the well-separated field (denoted by an arrow from w) and the other is shifted from its parent.

In general, a panel in the n-dimensional tree has at most $3^n - 1$ near-field elements and at most $6^n - 3^n$ well-separated neighbours. In particular, there are at most 8 near-field squares and 27 well-separated neighbours in two dimensions, see Figure 1 (a) of "near-field" and "well-separated field".

Stage 2: Upward approximation - to build multipole expansion. Begin from the finest level, for each panel that contains data sources, a combined multipole expansion at the centre of this panel is constructed to approximate the whole contributions from those sources (see Figure 2 (b)). Then all the children multipole expansions are shifted to their parent to form the upward multipole approximation (see Figure 2 (c)).

Stage 3: Downward approximation - to build Taylor series. Begin from the coarsest level, for each panel that contains the evaluation points, a local Taylor series at the centre of this panel is constructed from two sources: one is obtained by translating its well-separated neighbour multipole expansions, which cover contributions from all well-separated neighbours, and the other comes from the shifted parent local Taylor series, which cover contributions from all far-field sources excluding well-separated

Fast multipole method for accelerating the evaluation of splines, F. Chen and D. Suter - to appear IEEE CS&E Sept. 3 1998

6

neighbours, see Figure 2 (d).

Stage 4: Evaluation. At the finest level, for each evaluation point within the panel P, the evaluation result is obtained by adding together two parts: (a) the truncated local Taylor series approximation to all far field interactions, and (b) the direct calculation with data sources within P and the near-field of P.

2.2   Related work

The fast multipole method was first proposed by Greengard and Rokhlin [8] for 2-dimensional potential calculation. This method has been popular used in solving various problems of calculating N-body interactions. Various attempts have been made to improve the FMM algorithm. For instance, Anderson [9] presented an alternative method of constructing the coefficients of the multipole expansions by using a Poisson formula. Fast Fourier transform techniques have been applied to accelerate the local Taylor approximation stage since this stage dominates the overall runtime of the FMM [10]. An exhaustive survey of this large literature is beyond of the scope of this article.

In fact the applications of the FMM is not restricted to N-body simulations: it provides a general approach for rapid calculation of a finite summation. Boyd [11] has discovered the similarity between multipole method and the fast Fourier transform for some kinds of problems. More closely related to the work summarised here: Beaston, Newsam [4] and Suter [3] have used FMM to fast evaluate a 2-dimensional thin-plate spline kernel, as well as other radial basis functions [12]. Suter's method, close to Anderson's for the harmonic potential kernel [9], employed a Poisson formula for a biharmonic thin-plate spline kernel to calculate the multipole expansion coefficients by an integral along a closed contour surrounding the centres of the kernel. Our work further extended the use of FMM application by using them for the rapid evaluation of the DIV-CURL vector splines in two and three dimensions. The precise mathematical formulation and the error analysis of the algorithms in 2D and 3D can be found in [6] and [7].

3   Numerical results

In this section, we show the numerical results of the implementation of the FMM for the evaluation of 2D vector splines (1) (for details see [6]). Such vector spline kernels are defined in terms of the second order derivatives of the related scalar kernel, see equation (2). Moreover, these vector kernels are related to parameters α and β, which depend on the modelled data. The issue of selection of α and β in practical applications (ref. [2]) is beyond of the scope of this paper. Here we only emphasize that the proposed fast multipole method is independent of the value of α and β except for the special case where $\alpha = \beta$. In such a case, the vector kernel degenerates to the thin-plate spline kernel. Therefore, the evaluation of the vector spline, when $\alpha = \beta$, is equivalent to the evaluation of two independent thin-plate splines [4]. The corresponding FMM for the thin-plate spline is certainly more efficient than that for the vector case (principally because there are no off-diagonal terms in equation (2)). When $\alpha = \beta$, since the computational cost for the same truncation index p is less than for

Fast multipole method for accelerating the evaluation of splines, F. Chen and D. Suter - to appear IEEE CS&E Sept. 3 1998
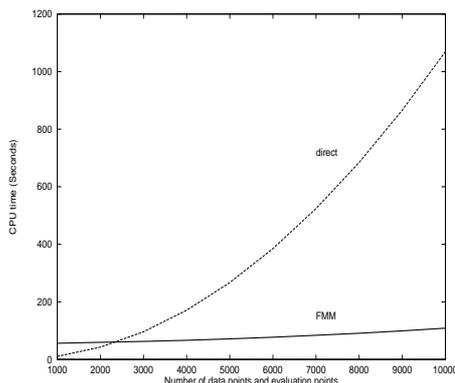
7

Figure 3: shows CPU time for a 5-level fast multipole scheme and for direct calculation of a 2D vector spline. The number of truncation for both multipole and Taylor expansions are set to 10. The solid line represents the CPU time for a 5-level multipole algorithm, which increases linearly with the number of points N. The dash line represents the CPU time for direct calculation, which has a order of $O(N^2)$.

$\alpha \neq \beta$, one can increase the truncation index and gain more accuracy for the same total computational cost. That is, one can trade off the time saving for increased accuracy - in this sense the efficiency of the $\alpha = \beta$ solution is greater. However, the issue should not be so much as to whether the case $\alpha = \beta$ is more efficient for a given physical problem but whether the case $\alpha = \beta$ is appropriate in terms of the underlying physics of the problem. The numerical results, shown in this section, are all based on the general case of $\alpha \neq \beta$. The algorithm was implemented in a C program on a Silicon Graphics Indy (R4600 processor) to illustrate the efficiency and the accuracy of the algorithm.

3.1   Timing

Figure 3 shows the CPU time for a 5 level fast multipole method, and the CPU time for direct calculation. The data points and evaluation points are generated randomly and distributed uniformly over $[0, 1] \times [0, 1]$. The series truncation is set to 10 for both the multipole and the local Taylor expansions. As shown from Figure 3, the computation time using the fast multipole method has a linear relation with N, while the timing using direct method increases in a tendency of $O(N^2)$.

In FMM, the refinement level l influences the runtime of the fast multipole algorithm. More specifically, increasing the refinement level produces more panels, which results in more computational costs in the implementation. On the other hand, increasing the number of data points N requires a finer subdivision of the data tree to avoid too much near-field direct calculations. Therefore, the choice of the refinement level mainly depends on the number of data points. Figure 4 (a) depicts the CPU time of fast multipole method changing with the different levels (fixed $p = 10$), where level l means a $2^l \times 2^l$ grid of panels. As shown from Figure 4 (a), when the number of points is larger than 6500, a level 5 scheme performs much efficiently than a level 4 scheme. In general, for a nearly uniformed data system, the refinement level is chosen $l \approx \log_{2^n} N$ so that most of the panels at this level contain at least one data source (where n is the dimensions of the space where the problem to be solved).

Fast multipole method for accelerating the evaluation of splines, F. Chen and D. Suter - to appear IEEE CS&E Sept. 3 1998

8

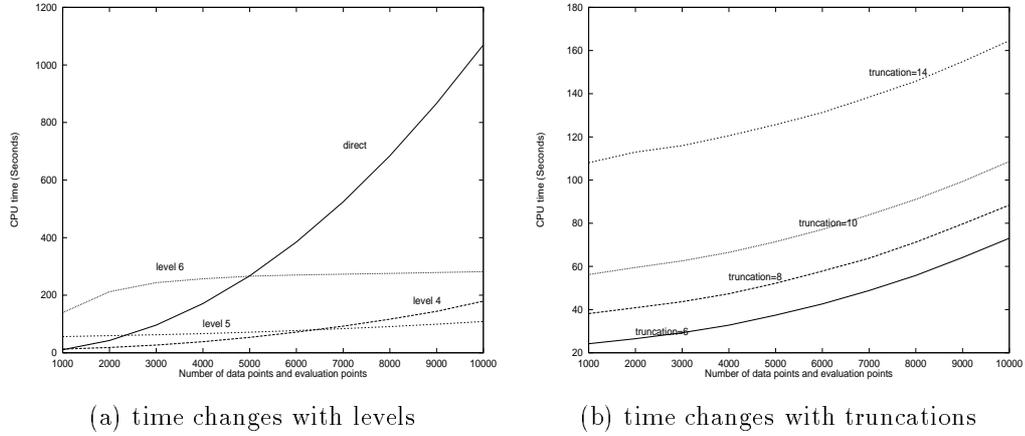(a) time changes with levels　　　　(b) time changes with truncations

Figure 4: *shows that CPU time of the fast algorithm is influenced by the parameters l and p. (a) shows the change of the CPU time with different levels, where level l means a $2^l \times 2^l$ grid of squares, and truncation index $p = 10$. (b) shows the change of the CPU time with different truncation points in a 5-level FMM scheme. The runtime increases rapidly when the truncation index increases.*
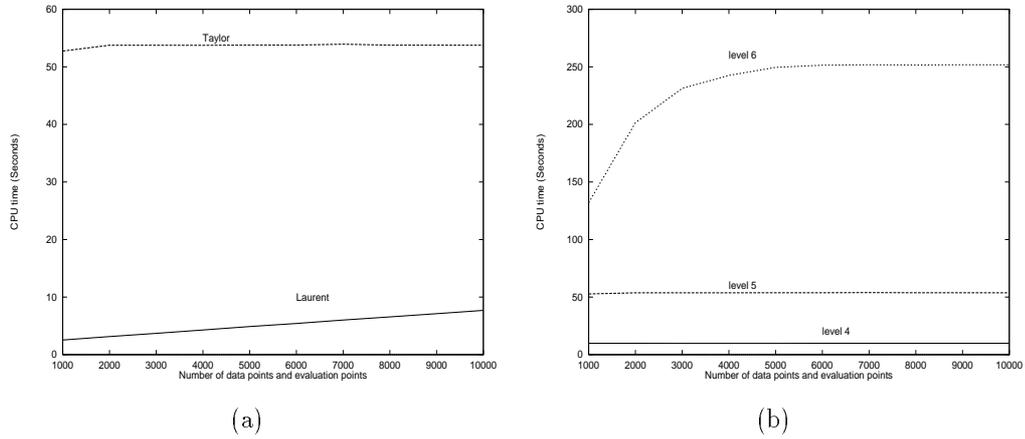

(a)　　　　(b)

Figure 5: shows CPU time spending on constructing local Taylor series. (a) CPU time spent on the Taylor approximation and on the multipole approximation for a 5-level scheme shows that the Taylor approximation process is time dominated. (b) CPU time on building Taylor series increases sharply as the refinement level becomes large, but with a fixed level the time spent on building the Taylor series depends less on the number of evaluation points.

Fast multipole method for accelerating the evaluation of splines, F. Chen and D. Suter - to appear IEEE CS&E Sept. 3 1998

9

Another parameter involved in FMM is the truncation index p of the series expansions. This truncation index also influences the efficiency of the FMM and can be chosen according to the required accuracy. Figure 4 (b) shows that the runtime of the fast algorithm varies with the truncation index (fixed $l = 5$). Obviously, the calculation will take longer if the series expansion is calculated using more terms. However, the trade-off between the runtime and the accuracy should be considered when choosing the truncation index since the series expansion with less terms will result in a larger error, see Table 1. In practical implementations of the FMM, for a given precision ϵ to be achieved, the truncation index can be chosen as $p \geq |\log_2(\epsilon)|$ [7].

The large amount of time in implementing FMM is spent on constructing the local Taylor series expansion due to the relative large number of well-separated neighbours (particularly, there are 27 well-separated neighbours for 2D FMM and 189 well-separated neighbours for 3D FMM). Figure 5 (a) shows that the local Taylor series approximation dominates the runtime of the FMM algorithm. In addition, the CPU time spent on the local Taylor series approximation increases sharply as the refinement levels l increases, see Figure 5 (b), but depends less on the number of points N. The last property implies that when the number of points becomes larger, a higher level scheme can be employed to reduce the amount of the direct calculations between the nearest neighbours.

3.2 Error estimation

The error of FMM algorithm strongly depends on the truncation index of the series expansions. The error bounds decrease as the truncation index increases. Table 1 shows the maximum absolute error and maximum relative error with different number of truncations for each component f_1 and f_2.

truncation p	$\epsilon_{max}^{f_1}$	$\epsilon_{re-max}^{f_1}$	$\epsilon_{max}^{f_2}$	$\epsilon_{re-max}^{f_2}$
6	0.104688	0.000687582	0.135803	0.00330963
8	0.00866754	0.000266301	0.0116244	0.000253477
10	0.000930466	3.87372e-05	0.00125236	4.21111e-05
14	1.7925e-05	2.72763e-07	1.51653e-05	3.27185e-07

Table 1: The largest absolute error and relative error of FMM with respect to different truncation index for a 5-level scheme, where number of data points and evaluation points are both equal 4000. The numerical results show that the maximum errors reduce quickly as the truncation index becomes larger.

4 Conclusion

We have presented a general approach of the speed up the spline calculation by the fast multipole method. This approach has proven to be very efficient and generally applicable as no regular grid is necessary and it is able to encompass whole families of PDE splines (including scalar and vector). We presented some of the numerical results of our experiments with the implementation of the fast algorithm for 2D vector splines evaluation. We also discussed the effect on timing and accuracy of the algorithm-based parameters.

References

[1] G. Wahba. *Spline models for observational data.* SIAM, Philadelphia, PA, 1990.

[2] L. Amodei and M. N. Benbourhim. A vector spline approximation with application to meteorology. *Curves and Surfaces, P. J. Laurent, A. Le Mehaute, and L. L. Schumaker (eds.).*, pages 5–10, Acamdemic Press, Boston, 1991.

[3] D. Suter. Fast evaluation of splines using poisson formula. *Journal of Applied Science and Computations*, 1(1):70–87, June 1994.

[4] R. K. Beatson and G. N. Newsam. Fast evaluation of radial basis functions: 1. *Comp. Math. Applic.*, 24(12):7–20, December 1992.

[5] A. Brandt. Multilevel computations of integral transforms and particle interactions with oscillatory kernels. *Computer Physics Communications*, 65:24–38, 1991.

[6] F. Chen and D. Suter. Fast evaluation of vector splines in two dimensions. *The International Association for Mathematics and Computers in Simulation - IMACS*, 1:469–474, August 1997.

[7] F. Chen and D. Suter. Fast evaluation of vector splines in three dimensions. *Technical Report MECSE97-6, Monash University*, Submitted to Journal of Computing, December 1997.

[8] L. Greengard and V. Rokhlin. A fast algorithm for particle simulations. *J. Comput. Phys.*, 73:325–348, 1987.

[9] C. R. Anderson. An implementation of the fast multipole method without multipoles. *SIAM J. Sci. Stat. Comput.*, 13(4):923–947, July 1992.

[10] W. D. Elliott and JR. J. A. Board. Fast fourier transform accelerated fast multipole algorithm. *SIAM J. Sci. Comput.*, 17(2):398–415, March 1996.

[11] J. P. Boyd. Multipole expansions and pseudospectral cardinal functions: A new generalization of the fast fourier transform. *Journal of Computational Physics*, 103:184–186, 1992.

[12] R. K. Beatson and W. A. Light. Fast evaluation of radial basis functions: Methods for 2-dimensional polyharmonic splines. *Research Report, No. 119. University of Canterbury*, December 1994.