INTERNATIONAL JOURNAL FOR NUMERICAL METHODS IN ENGINEERING
Int. J. Numer. Meth. Engng 2008; **00**:1–6 *Prepared using* nmeauth.cls *[Version: 2002/09/18 v2.02]*

Characterization of the accuracy of the Fast Multipole Method in particle simulations

Felipe A. Cruz, L. A. Barba *†

Department of Mathematics, University of Bristol, Bristol, BS8 1TW, U.K.

SUMMARY

The Fast Multipole Method (FMM) is a fast summation algorithm capable of accelerating pairwise interaction calculations, known as N-body problems, from an algorithmic complexity of $\mathcal{O}(N^2)$ to $\mathcal{O}(N)$ for N particles. The algorithm has brought dramatic increase in the capability of particle simulations in many application areas, such as electrostatics, particle formulations of fluid mechanics, and others. Although the literature on the subject provides theoretical error bounds for the FMM approximation, there are not many reports of the measured errors in a suite of computational experiments. We have performed such an experimental investigation, and summarized the results of about 1000 calculations using the FMM algorithm, to characterize the accuracy of the method in relation with the different parameters available to the user. In addition to the more standard diagnostic of the maximum error, we supply illustrations of the spatial distribution of the errors, offering visual evidence of all the contributing factors to the overall approximation accuracy: multipole expansion, local expansion, hierarchical spatial decomposition (interaction lists, local domain, far domain). This presentation is a contribution to any researcher wishing to incorporate the FMM acceleration to their application code, as it aids in understanding where accuracy is gained or compromised.

KEY WORDS: fast multipole method; order-N algorithms; particle methods; vortex method; hierarchical algorithms

1. INTRODUCTION

In particle simulations, often the dynamics results from the summation of all pair-wise forces in the ensemble of particles. Such situations arise in astrophysics, molecular dynamics, plasma physics, and certain formulations of fluid dynamics problems, for example. The total field of interest (gravitational, electrostatic, etc.) at one evaluation point requires adding the contribution of all source points or particles, and so if both evaluation points and particles are

*Correspondence to: labarba@bu.edu
†New address: Department of Mechanical Engineering, Boston University, Boston MA 02215 U.S.A.

Contract/grant sponsor: Airbus and BAE Systems; contract/grant number: ACAD 01478 (2007)

numbered at N, a total of N^2 operations is needed. This fact was for a long time an impediment to the wider use of particle simulations, as the computational effort becomes prohibitive for large numbers of particles.

The above scenario changed dramatically with the introduction of tree-codes and the fast multipole method (FMM), which appear in the late 1980s for accelerated evaluation of N-body problems. Tree-codes [1, 5] are generally perceived to be easier to grasp and program, and provide a complexity of $\mathcal{O}(N \log N)$. The FMM was introduced as an algorithm for the rapid evaluation of gravitational or Coulombic interactions [12] and promises a reduction in computational complexity to $\mathcal{O}(N)$. It has, since its dissemination, been adapted for many applications: for fast evaluation of boundary elements [11], for vortex sheet problems with desingularized equations [13], for long-range electrostatics in DNA simulations [10], and many others. The impact of the FMM has been undeniably huge, resulting in it being chosen as one of the Top 10 Algorithms of the 20th Century [9].

Despite the great volume of work using and adapting the FMM in many application areas, there remains some lack of insight regarding how the algorithm can be efficiently used to obtain an accurate representation of the field of interest. The error of the FMM approximation is estimated by theoretical bounds, which as could be expected reveal a trade-off between accuracy and efficiency of the computation. However, there is not much literature providing measurements of the accuracy of the approximation, in practice. One may often find such assertions in published works as "only the first three moments of the expansion were used", or something to that effect. But just as often there is no information provided about the actual errors which are *observed*. Of course, it is not easy to provide such measures of observed error, as this would require additional computations using the direct $\mathcal{O}(N^2)$ method, for comparison purposes. Nevertheless, it is important for users of the algorithm to know what to expect in terms of accuracy and efficiency, depending on the choice of algorithm parameters.

We aim to contribute to this gap in understanding by presenting a methodical investigation into the accuracy of the FMM, when the underlying 'client' application is the calculation of the velocity field induced by N regularized particles of vorticity. This application is rather more demanding than the Newtonian force calculation, because in the latter case the gravitational interaction is dominated by the first moment —due to the fact that all mass is positive. Therefore, keeping only the first three moments could easily give the desired accuracy. On the other hand, as in Coulomb electrostatic calculations, the vortex particles can be both positive and negative, and thus an acceptable accuracy may require that more terms in the expansion be kept.

For the purposes of this study, a prototype code of the FMM computation of the velocity induced by N vortex particles was implemented using the Python[†] language. The nice features of Python —such as dynamic typing, extensive numerical libraries, and high programmer productivity— helped us produce a piece of software which is easy to use and easy to understand. We have recently used this Python code as a starting point for a parallel version of the code which, in collaboration with members of the developer team, is being incorporated to the PETSc library for scientific computing[2]. Preliminary results with the parallel code were presented [8] in the latest Parallel CFD Conference[‡]. Full code verification and scalability

[†] http://www.python.org/
[‡] http://www.parcfd.org//

studies are ongoing. Our final aim is to contribute to the community of particle simulations with an open source FMM implementation which is parallel, portable and extensible. For the time being, the Python code is being made available publicly and we welcome correspondence from interested researchers[§].

Using the Python FMM code, more than 1500 calculations were performed, varying the numerical parameters: N, the number of particles, l, the number of levels in the tree, and p, the truncation level of the multipole expansion. We looked not only at the maximum error in the domain, which would be the conventional approach; we also present results showing how the error varies in space, revealing some interesting features of the method. Through this presentation of the results, we believe a clear characterization of the nature of the FMM approximation is obtained.

The paper is organized as follows. The next section presents an outline of the vortex particle method, for completeness. In §3, we offer an overview of the FMM, with some details of our implementation. Following, in §4, we discuss the sources of errors in the FMM algorithm. And finally, §5 reports the detailed experiments using the FMM for evaluation of the velocity of N vortex particles; the behavior of the method will be illustrated for varying parameters, as well as the impact on the efficiency of the calculation, for different problem sizes.

2. REVIEW OF THE VORTEX PARTICLE METHOD

For incompressible flow one has that $\nabla \cdot \mathbf{u}(\mathbf{x}, t) = 0$, and the two-dimensional vorticity transport equation is expressed as,

$$\frac{\partial \omega}{\partial t} + \mathbf{u} \cdot \nabla \omega = \nu \Delta \omega. \tag{1}$$

The vortex particle method proceeds by spatially discretizing the vorticity field onto particles which have a radially symmetric distribution of vorticity, and thus the vorticity field is effectively approximated by a radial basis function expansion as follows:

$$\omega(\mathbf{x}, t) \approx \omega_\sigma(\mathbf{x}, t) = \sum_{i=1}^{N} \Gamma_i(t) \zeta_{\sigma_i} \left(\mathbf{x} - \mathbf{x}_i(t) \right). \tag{2}$$

Here, the $\mathbf{x}_i$ represent the particle positions, Γ_i are the circulation values assigned to each vortex particle, and the core size is σ_i. The core sizes are usually uniform ($\sigma_i = \sigma$), and the characteristic distribution of vorticity ζ_{σ_i}, commonly called the cut-off function, is frequently a Gaussian distribution, such as:

$$\zeta_\sigma(\mathbf{x}) = \frac{1}{2\pi\sigma^2} \exp\left(\frac{-|\mathbf{x}|^2}{2\sigma^2} \right). \tag{3}$$

The velocity field is obtained from the vorticity by means of the Biot-Savart law of vorticity dynamics:

$$\mathbf{u}(\mathbf{x}, t) = \int (\nabla \times \mathbf{G})(\mathbf{x} - \mathbf{x}') \omega(\mathbf{x}', t) d\mathbf{x}' = \int \mathbf{K}(\mathbf{x} - \mathbf{x}') \omega(\mathbf{x}', t) d\mathbf{x}' = (\mathbf{K} * \omega)(\mathbf{x}, t)$$

[§] http://code.google.com/p/pyfmm/

where $\mathbf{K} = \nabla \times \mathbf{G}$ is the Biot-Savart kernel, with $\mathbf{G}$ the Green's function for the Poisson equation, and $*$ representing convolution. For example, in 2D the Biot-Savart law is written explicitly as,

$$\mathbf{u}(\mathbf{x},t) = \frac{-1}{2\pi} \int \frac{(\mathbf{x}-\mathbf{x}') \times \omega(\mathbf{x}',t)\hat{\mathbf{k}}}{|\mathbf{x}-\mathbf{x}'|^2} d\mathbf{x}'. \tag{4}$$

The vorticity transport is solved in this discretized form by convecting the particles with the local fluid velocity, and accounting for viscous effects by changing the particle vorticity. Hence, the unbounded vortex method is expressed by the following system of equations:

$$\frac{d\mathbf{x}_i}{dt} = \mathbf{u}(\mathbf{x}_i,t) = (\mathbf{K} * \omega)(\mathbf{x}_i,t), \qquad \frac{d\omega}{dt} = \nu\nabla^2\omega. \tag{5}$$

There are a variety of ways to account for the viscous effects represented by the second of these equations, of which a discussion is offered in [4]. See also the book [7]. But the convection of vorticity is always treated in the same way: by integration of the ordinary differential equations for the particle trajectories. In this process, it is necessary at each step to calculate the velocity at each particle location, using the Biot Savart law. Using a radially symmetric distribution function, one can obtain an analytic result of the integral that appears in (4), which results in an expression for the velocity at each point as a sum over all particles. For the 2D Gaussian particle distribution, one has:

$$\mathbf{K}_\sigma(\mathbf{x}) = \frac{1}{2\pi r^2}(-y,x)\left(1-\exp(-\frac{r^2}{2\sigma^2})\right). \tag{6}$$

where $r^2 = x^2 + y^2$. Thus, the formula for the discrete Biot-Savart law in two dimensions gives the velocity as follows,

$$\mathbf{u}_\sigma(\mathbf{x},t) = \sum_{j=1}^{N} \Gamma_j \, \mathbf{K}_\sigma(\mathbf{x}-\mathbf{x}_j). \tag{7}$$

As can be seen in Equation (7), calculating the velocity at one point takes N operations. Thus, calculating the velocity at every point is an N-body problem.

The fact that the direct evaluation of the velocity field induced by N vortex particles is an $\mathcal{O}(N^2)$ calculation was for a long time an obstacle for wider acceptance of vortex methods. The dissemination of the fast multipole method quickly provided an opportunity to make vortex particle methods more efficient and perhaps even competitive with mainstream methods for direct numerical simulation. Indeed, the vortex particle method has matured in the last, say, 15 years, and we now have demonstrations of highly competitive applications, such as for example a recent calculation of aircraft trailing wakes involving a *billion* particles [6]. Such results would be impossible without the FMM approximation of the Biot-Savart calculation.

3. IMPLEMENTATION OF THE FMM

3.1. Overview of the algorithm

The fast multipole method (FMM) is a fast summation algorithm typically used for accelerating the computation of so-called N-body problems, that is, problems involving N bodies or particles interacting with each other.

Suppose that a function f can be defined to describe the influence of all the particles of a system at any place; f is written as the sum of pairwise interactions with a set of source particles, as follows:

$$f(y) = \sum_{i=1}^{N} c_i \mathbf{K}(y - x_i). \tag{8}$$

where c_i are scalar values and x_i represent the source particles. The assumptions when applying the FMM are:

1. The evaluation of $f(\cdot)$ occurs at a large number of *evaluation* points $\{y_i\}$. Typically, the evaluation points include the source points.
2. It is common that the set of source points $\{x_i\}$ and the set of evaluation points $\{y_i\}$ contain about the same number of members.
3. The kernel $\mathbf{K}$ that represents the pairwise interactions decays monotonically as it gets further away from the origin, and should be smooth far from the origin; such kernel, when evaluated far away from the source point, will enable us to compute the aggregate effect of a cluster of source points instead of computing all pairwise interactions.
4. The kernel function $\mathbf{K}$ can be decoupled into terms that depend on the source points and terms that depend on the evaluation points; this idea is clearly shown in equations (10) and (11).

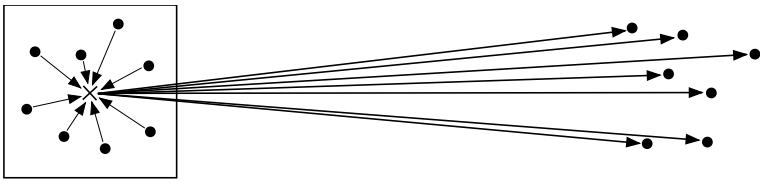
Under these conditions, in a system of N particles interacting with each other, the algorithmic complexity of the direct calculation of Equation (8) at all evaluation points is $O(N^2)$.

The fast multipole method is an order $O(N)$ algorithm, and is based on the idea that the influence of a cluster of particles can be approximated by an agglomerated quantity, when such influence is evaluated far enough away from the cluster itself. The algorithm divides the computational domain into a *near-domain* and a *far-domain*:

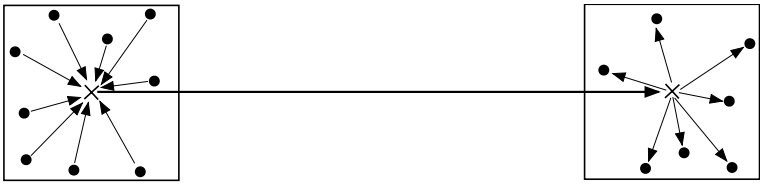
Near domain: contains all the particles that are near the evaluation point, usually a minor fraction of all the N particles. The influence of the particles in the near-domain is computed by directly evaluating the pair-wise particle interactions. The computational cost of directly evaluating the near domain is not dominant as the near-domain is kept small.

Far domain: contains all the particles that are far away from the evaluation point, which ideally means most of the N particles in the domain. The evaluation of the far domain will be sped-up by evaluating the approximated influence of clusters of particles rather than computing the interaction with every particle in the system; a sketch of this idea is presented in Figure 1.

The key idea behind the FMM is that particle influences can be clustered and expressed in two different representations: as Multipole Expansions (MEs) and as Local Expansions (LEs). These allow us to efficiently evaluate the particles' influence. The MEs and LEs are Taylor (or other) series that converge in different subdomains of space. The center of the series for an ME is the center of the cluster of source particles, and it only converges outside the cluster of particles. In the case of an LE, the series is centered near an evaluation point and converges locally. The exact location where a series is centered is important as we can add together series



(a) A multipole expansion (ME) of a cluster of source particles is used to efficiently evaluate the field at particles located far away from the evaluation points.



(b) A local expansion (LE) is used in conjunction with MEs; the LE allows efficient evaluation of a set of MEs at a common evaluation cluster.

Figure 1. Illustration of how multipole expansions (MEs) and local expansions (LEs) are used to efficiently evaluate particle interactions.

that: share the same center and overlap their convergence domains. In Figure 1, we illustrate how the FMM uses these expansion.

As previously stated, the first step of the FMM is to decompose the domain space. This is accomplished by means of a hierarchical data structure (usually a tree structure) that is used to decompose the domain into clusters of different sizes. We can always make a flat diagram representing the tree structure as in Figure 2, regardless of the dimension of the domain (in two dimensions, four branches stem from each node). With the branches of the tree structure representing different regions of the domain, combinations of nodes of the tree are used to cover the domain. Thus the tree structure supplies the definitions of the near and far domains. Using the nodes of the tree to cluster the particles, clusters are selected for the MEs representing the far domain.

The next step is to build the MEs for each node of the tree; this is done recursively: the first MEs are built at the deepest level of the tree, level l, and then efficiently translated to the center of the parent cell. This is referred to as the *upward sweep* of the tree. Then, in the *downward sweep*, the MEs are first translated into LEs for all the boxes in the *interaction list*. At each level, the interaction list corresponds to the cells of the same level that are in the far field for a given cell. Finally, the LEs of upper levels are added up to obtain the complete far domain influence for each box at the leaf level of the tree. These ideas are better visualized with an illustration, as provided in Figure 3.

The total field at each evaluation point is thus the sum of the contributions of the near and far domain:

$$\begin{aligned} f(y) &= f^{\text{near}}(y) + f^{\text{far}}(y) \\ &= \sum_{x_i \text{ near } y} c_i \, \mathbf{K}(y - x_i) + \sum_{x_i \text{ far from } y} c_i \, \mathbf{K}(y - x_i). \end{aligned} \tag{9}$$

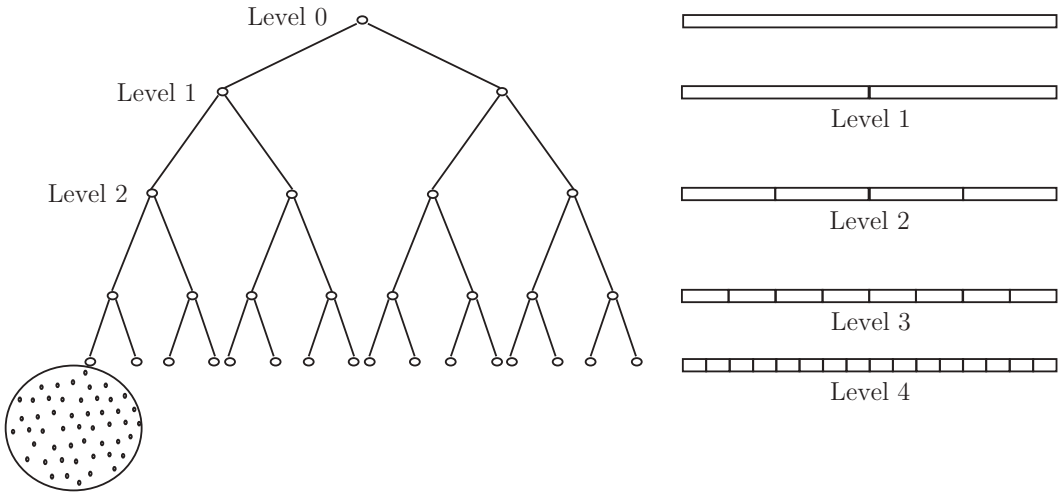


Figure 2. Sketch of a one-dimensional domain (right), divided hierarchically using a binary tree (left), to illustrate the meaning of levels in a tree and the idea of a final leaf holding a set of particles at the deepest level. In this one-dimensional example, level 0 is the whole domain, which is split in two halves at level 1, and so on up to level l. In two-dimensions, instead each domain is divided in four, to obtain a quadtree, while in three-dimensions, domains are split in eight to obtain an oct-tree

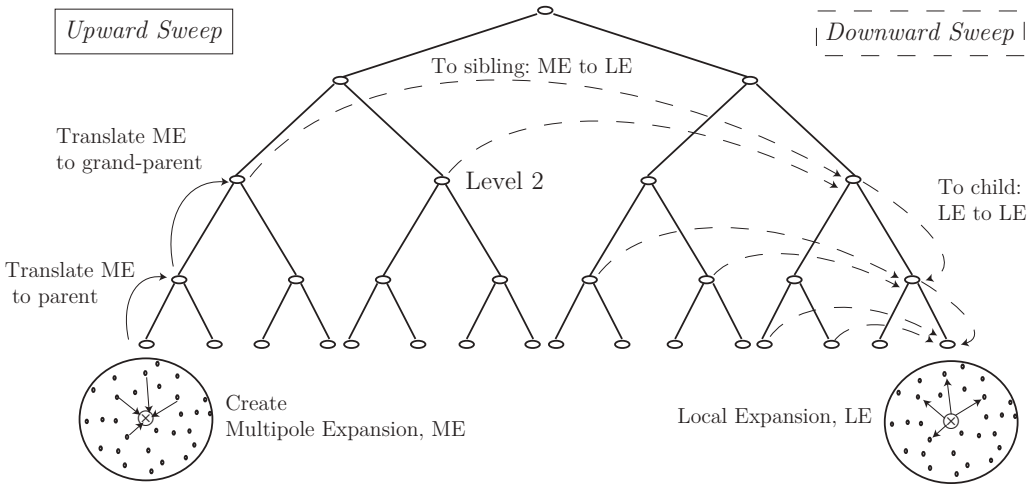


Figure 3. Illustration of the *upward sweep* and the *downward sweep* of the tree. *Upward sweep*: the multipole expansions (ME) are created at the deepest level, then translated upwards to the center of the parent cells. *Downward sweep*: the MEs are then translated to a local expansion (LE) for the siblings at all levels deeper than level 2, and then translated downward to children cells. Finally, the LEs are created at the deepest levels.

The first summation of (9) performs the computation of the function f using the direct method over the particles inside the near field. The second summation of (9) is approximated by means of the fast multipole method. In order to be able to use the FMM approximation, the kernel function $\mathbf{K}$ must allow the decoupling of the influence of the source particles x_i and the evaluation point y. After truncating the infinite series expansion, the kernel is approximated by:

$$\mathbf{K}(y-x) = \sum_{m=0}^{p-1} a_m(x) f_m(y). \tag{10}$$

Using (10) in (8) and rewriting the equations, we obtain an expression which is referred to as *multipole expansion*:

$$f(y_j) = \sum_{m=0}^{p-1} \left(\sum_{i=1}^{N} c_i a_m(x_i) \right) f_m(y_j). \tag{11}$$

The series expansion of the kernel in Eq. (10) might not exactly represent or approximate $\mathbf{K}$ for arbitrary values of x and y, but we expect that for particles located in the far-field domain the effect of $\mathbf{K}$ will be smooth, and be approximately decoupled due to the fact that the source particles $\{x_i\}$ and the evaluation points $\{y_j\}$ are well separated. By truncating the infinite series in Eq. (10), the number of terms left after the truncation procedure will control the accuracy of the approximation. The terms that are dependent on the source points can be computed a single time at cost $\mathcal{O}(N)$. Then for N evaluations of Eq. (11), only $\mathcal{O}(pN)$ operations are performed instead of the original $\mathcal{O}(N^2)$ operations needed for directly computing the interactions with Eq. (8). So far, we have introduced a general idea of how the FMM works, but much of the details of the final $\mathcal{O}(N)$ algorithm have been left out. For more details of the algorithm, we cite [12].

4. SOURCES OF ERROR IN THE FMM AS USED IN THE VORTEX METHOD

Before presenting our numerical experiments, let us review the sources of error in the FMM approximation, as it applies to our 'client' application, the vortex particle method. Our implementation is based on *(i)* replacing the kernel in the far field to ease the mathematical derivation of MEs, and *(ii)* using Taylor series for the MEs (other choices of series are possible). We identify two different sources of errors, which we will refer to as errors *Type I* and *Type II*.

4.1. Error Type I

The first type of error occurs from the approximation of the Biot-Savart kernel (6) for all far field interactions by a singular kernel (as in [13]). Thus, the original kernel has been replaced by one for which it is more simple to construct the Multipole Expansion. The following kernel is used for the series expansions:

$$u(z) = \sum_{i=1}^{N} \Gamma_i \, \mathbf{K}(z - z_i) \tag{12}$$

$$\mathbf{K}(z) = -\frac{1}{2\pi|z|^2} \begin{pmatrix} v \\ -u \end{pmatrix} \tag{13}$$

$$z = u + jv, \qquad u, v \in \mathbb{R}, z \in \mathbb{C}.$$

So, instead of using the original kernel (6) in the whole domain, we use the kernel of Equation (13) in the far field. This approximation can achieve machine precision if used far enough away from the source, due to the rapid decay of the Gaussian. An approximation error is introduced to the original velocity field, $u(z)$, when neglecting the Gaussian influence in (6). An error bound can be constructed for the approximate velocity field, $u'(z)$:

$$|u(z) - u'(z)| \leq \frac{1}{2\pi|a|} \exp\left(\frac{-|a|^2}{2\sigma^2}\right) \left(\sum_{i=1}^{N} \Gamma_i\right). \tag{14}$$

Here, a is the minimum distance between an evaluation point and any particle in the far field. The parameter σ corresponds to the particles' characteristic size, and the values of Γ_i are the circulation of the particles. The value of a, in practice, is given by the size of the smallest box of the FMM tree; therefore, to keep error Type I as small as possible, at machine error in fact, a must chosen so that it is a given multiple of the particle characteristic sizes, σ.

4.2. Error Type II

This type of error is an unavoidable part of using Multipole Expansions. As explained in § 3, an ME is an infinite series expansion, which exactly represents the kernel of interest, in our case the kernel in Eq. (13). In practice, the infinite series needs to be truncated in order to be used, as in Eq. (11), and by keeping only p terms of the infinite series we introduce a truncation error. Due to the nature of the series expansion used here, many terms must be retained if one wishes to accomplish high accuracy (say, machine precision). This will have an impact on the efficiency of the method, as there is a considerable computational overhead at the time of translating the expansions in both the upward and downward sweeps, when the number of terms p is large. In our client application, we based our approximation in the following series expansion:

$$\frac{1}{1-a} = \sum_{m=0}^{\infty} a^m. \tag{15}$$

We rewrite equation (13) in complex notation and using the series expansion (15) we obtain an expression for the conjugate velocity, as follows:

$$[u(y)]^* = \sum_{m=0}^{\infty} \left(\sum_{i=1}^{N} -\frac{j\Gamma_i}{2\pi} (x_i - x_*)^m \right) (y - x_*)^{-m-1}. \tag{16}$$

Here, x_i corresponds to the source points, y is the evaluation point, and x_* corresponds to the center of a cluster of particles approximated by a multipole expansion. It is important to note that Equation (16) converges to Equation (13) only when $|x - x_*| < |y - x_*|$, due to a

restriction in the series expansion used, Equation (15). In practice, Equation (16) can only be used if truncated; by keeping only p coefficients we introduce the truncation error as follows:

$$\left|[u(y)]^* - [u(y)]_p^*\right| = \left|\sum_{m=p}^{\infty}\left(\sum_{i=1}^{N} -\frac{j\Gamma_i}{2\pi}(x_i - x_*)^m\right)(y - x_*)^{-m-1}\right|. \tag{17}$$

Here, $[u(y)]^*$ is the original infinite series and $[u(y)]_p^*$ is the same series but truncated after p terms. Using Equation (17) as an starting point, we can build an error bound for the truncated Equation (16):

$$\left|[u(y)]^* - [u(y)]_p^*\right| \leq A\left(\frac{1}{|y - x_*|}\frac{b^m}{1-b}\right). \tag{18}$$

where,

$$A = \left|\sum_{i=1}^{N}\frac{\Gamma_i}{2\pi}\right| \tag{19}$$

$$b \geq \left|\frac{r}{y_j - x_*}\right|. \tag{20}$$

From Equation (18) it is important to note the parameter b, which is defined in (20) as the ratio between the radius of the multipole expansion cluster, r, and the distance from the cluster to the evaluation point. The effect of b on the error bound is that as the evaluation point is closer to the cluster, the error should also increase in magnitude.

5. RESULTS OF NUMERICAL EXPERIMENTS

In the previous section, we discussed the expected sources of errors when accelerating the particle interactions of our client application. In this section, we present a characterization of the errors by means of different graphical representations of numerical experiments, and discuss how the actual observed errors relate to our theoretical expectations.

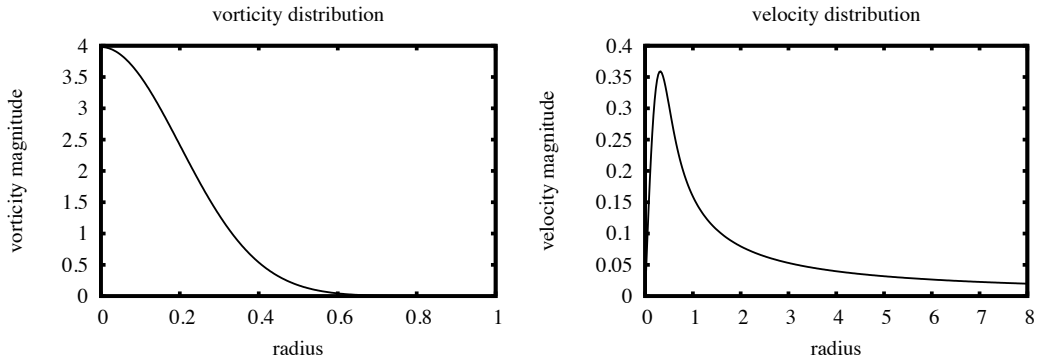
In order to characterize the errors introduced by our implementation, described in the beginning of §4, we present two different experimental setups, which were used in a combined total of more than 1500 numerical experiments. For a number of illustrative computational experiments, we present the results in the form of the maximum observed errors and the spatial distribution of the errors, in order to reveal the relationship between the different errors and the choice of algorithm parameters.

5.1. Experimental setups

In our numerical analysis we make use of two different experimental setups. For each experimental setup we define the particle parameters (position, circulation, and core radius of all particles) for which we study the effects of different choices of FMM parameters:

Level Number of levels in the hierarchical space decomposition, represented by the letter L.

Terms Number of terms of the Multipole Expansion that are kept, see Eq. (11), represented by the letter p.



(a) 1D profile of vorticity magnitude (b) 1D profile of velocity magnitude

Figure 4. One-dimensional profiles for the magnitudes of vorticity and velocity given by the Lamb-Oseen vortex. In the Lamb-Oseen vortex, the vorticity is concentrated at the origin and rapidly decays with distance, whereas the velocity is zero at the origin and its magnitude at first grows to then slowly decays with the distance. Parameters used: $\Gamma_0 = 1$, $\nu = 0.0005$ and $t = 4$ in Eq. (21).

5.1.1. LO-setup: Lamb-Oseen experiment setup — The first experimental setup corresponds to the use of the FMM in the evaluation of the velocity of the vortex particle representation of a viscous flow problem. The particles of the system are positioned on a lattice distribution in a square domain of a given size. The separation between particles is given by a constant spacing parameter h, obtained from the relation $\frac{h}{\sigma} = 0.8$ representing the overlap of the smooth vortex particles (more details about the significance of this parameter can be found in [4]). The parameter σ corresponds to the the core sizes of the particles, it is uniform for all the particles and set to $\sigma = 0.02$. The circulation of the particles, Γ_i, is obtained from the Lamb-Oseen vortex distribution as $\Gamma_i = h^2\omega(r_i, t - \sigma^2/2\nu)$, with r_i representing the particle location, and ω evaluated with Eq. (21). The justification of this shifted-time initialization for the Lamb-Oseen vortex distribution can be found in Ref. [3]. The time parameter t is fixed at $t = 4$ and $\nu = 0.0005$ is the constant viscosity value. For this case, the evaluation points correspond to the same source points and interactions are computed against all particles in the system (without self interaction). In practice we are solving an N-body problem.

The analytical solution of the velocity field of the Lamb-Oseen vortex is used to compare the results obtained using the Biot-Savart velocity, Equation (7) accelerated via the FMM. Also, the direct evaluation, obtained from summing all N^2 particle interactions, is used to cross check the results.

The analytical solution for the vorticity field of the Lamb-Oseen vortex is:

$$\omega(r, t) = \frac{\Gamma_0}{4\pi\nu t} \exp\left(\frac{-r^2}{4\nu t}\right). \tag{21}$$

where $r^2 = x^2 + y^2$. Figure 4 shows the spatial distribution of vorticity given by the Lamb Oseen vortex and the magnitude of the velocity field that it creates, with the parameters we have chosen, $\nu = 0.0005$ and $t = 4$.

5.1.2. SS-setup: Single source experiment setup — The second experiment corresponds to a setup without physical relevance, in which we have a single source particle with nonzero parameters, $\Gamma = 1$ and $\sigma = 0.02$, that is located at the center of a square shaped domain. The influence of the particle can be directly evaluated at any location of the domain by means of Eq. (7). A set of evaluation points were located on a lattice distribution across the square domain. For this experimental setup, we are interested in comparing the results obtained from the direct solution of the system against the results obtained from the FMM implementation.

5.2. Experimental Results

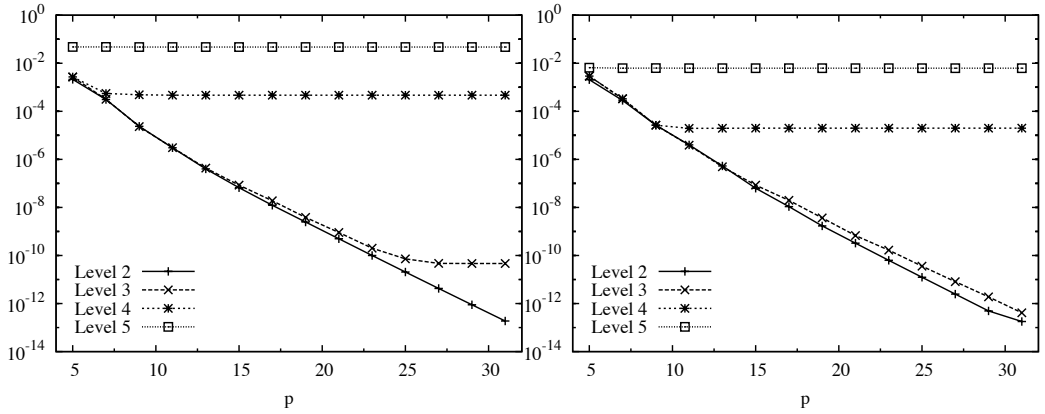
To start, we present the results obtained when using the Lamb-Oseen vortex as the problem set up. The first set of graphs, Figures 5 and 6, corresponds to the maximum measured error of the FMM approximation, for different algorithm parameters and problem sizes. The FMM is used to compute the velocity at all N particles of the system and the result is then compared against the analytical solution of the problem and normalized by the maximum velocity of the system. Each datapoint on the plots corresponds to a single experiment run for a given set of parameters. On these plots we can see the behavior of the measured error for the choice of level and truncation number in the algorithm. The curves in the plots represent the change of the measured error at a fixed FMM level (L) when varying the truncation number (p).

It is worth noting that as the separation between the particles is fixed (parameter h from §5.1.1), in order to introduce more particles into the calculation we increased the size of the problem domain. The domain size affects the space partitioning and hierarchical structure of the FMM. Bigger domains require that the boxes of the quadtree be bigger in order to cover the domain with a fixed number of levels, L. From the point of view of an evaluation point, a bigger domain implies that the near domain of a particle covers more space, and that the far field is located further away from the evaluation point.

A second set of plots, in Figures 7, 8, 9, and 10, characterizes the spatial structure of the error of the FMM approximation for a Lamb-Oseen problem setup with 11449 particles. Each plot corresponds to a single experiment with fixed parameters: level of refinement L and truncation number p. The magnitude of the error obtained when evaluating the velocity for every particle is represented by the color bar in $\log 10$. The plots of the spatial distribution of the error reveal the spatial structure of the approximations and the close relationship of the error with the data structure used in the algorithm. On the plots we can see the error structure that depends on the approximation of the far field and the one that depends on the number on terms retained for the Multipole Expansion, p (errors *Type I* and *Type II*).

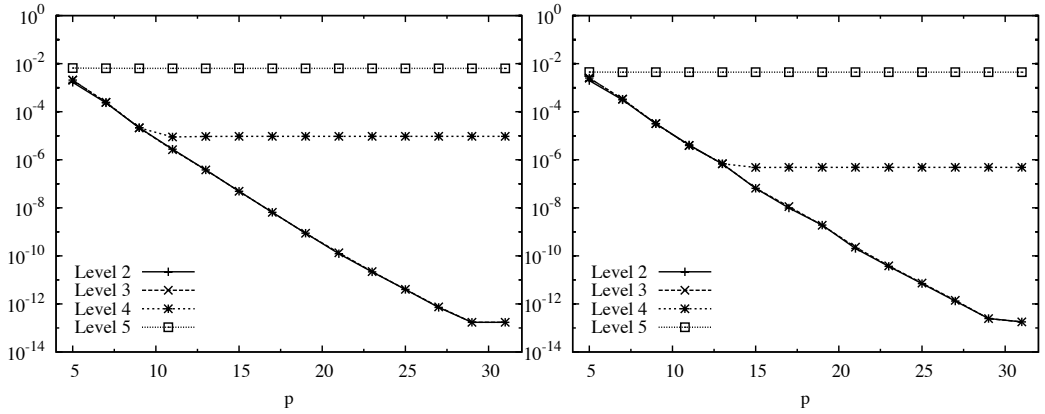
To focus on the change of the error of approximating the kernel with its singular version (error *Type I*) as the distance to the far field increases, we compare the Lamb-Oseen setup in three experiments with the same parameters for level of refinement (L) and truncation number (p) but different problem sizes: 3969, 7744, and 19044 particles. Using the setup described before, Figure 11 presents more clearly the effects of the approximation of the far field using the singular kernel; as the the size of the domain increases the distance to the far field also grows, decreasing the effect of the kernel substitution.

Figures 12, 13, 14 show the change of the spatial distribution of the error incurred by the FMM for a problem with a single source particle. On each figure, we compare the distribution of the error when more terms of the multipole expansion are kept for a fixed level, where each plot presents a single experiment for a fixed set of parameters (level and truncation



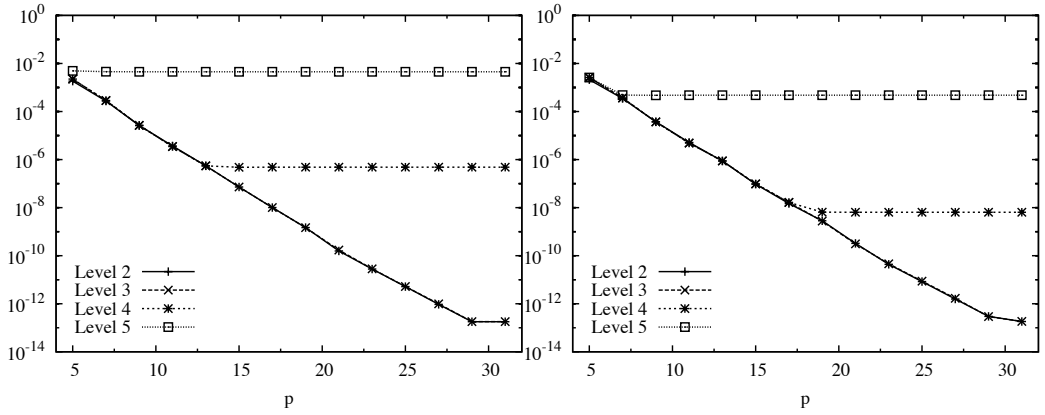
(a) $N = 3969$

(b) $N = 4761$



(c) $N = 5776$

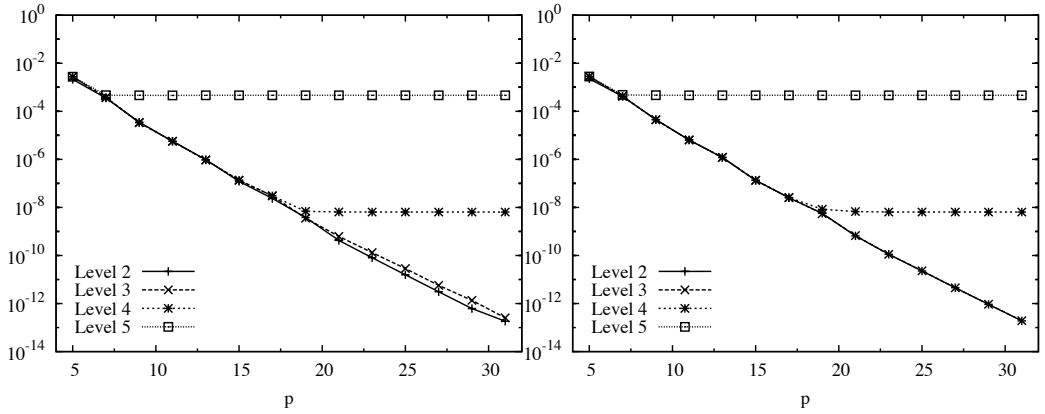
(d) $N = 6724$



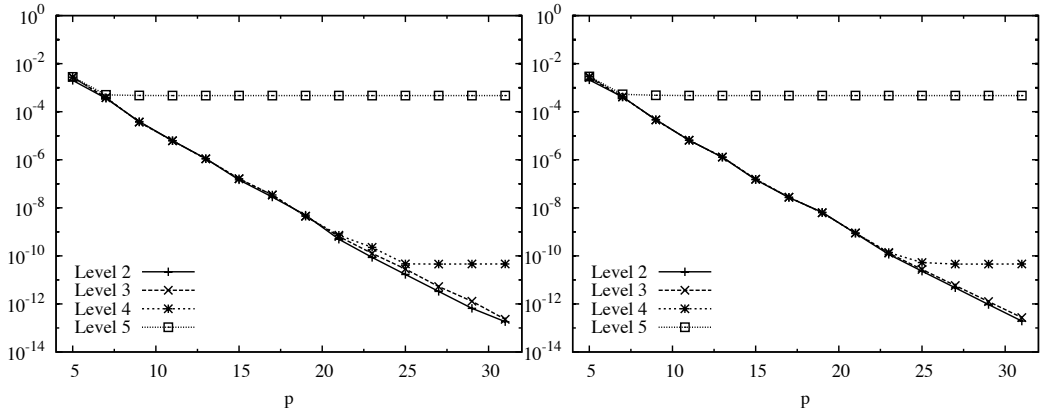
(e) $N = 7744$

(f) $N = 8836$

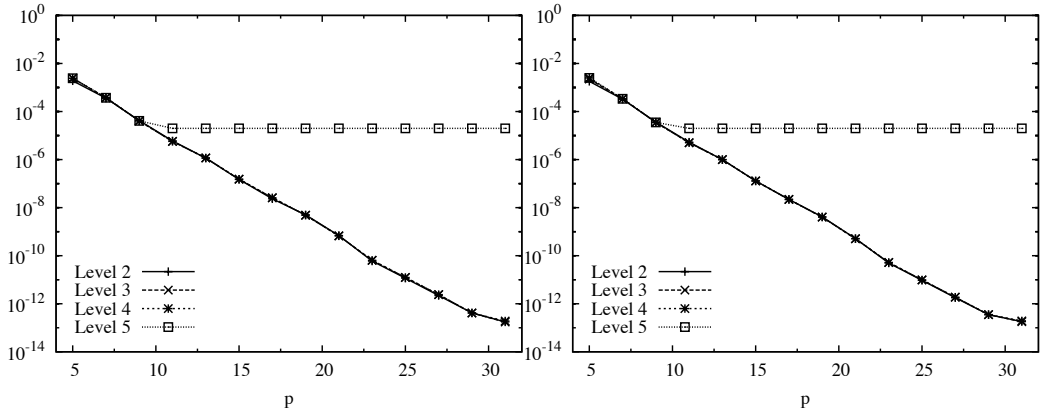
Figure 5. Maximum measured error of the FMM calculation of the velocity, with respect to the analytical value at every evaluation point. Each marker represents one full evaluation of the FMM velocity, with a given choice of level L and truncation p on the abscissa.



(a) $N = 10201$ (b) $N = 11449$



(c) $N = 12769$ (d) $N = 14161$



(e) $N = 15876$ (f) $N = 17424$

Figure 6. Maximum measured error of the FMM calculation, with respect to the analytical value of the velocity at every evaluation point. Each marker represents one full evaluation of the FMM velocity, with a given choice of level L and truncation p on the abscissa.

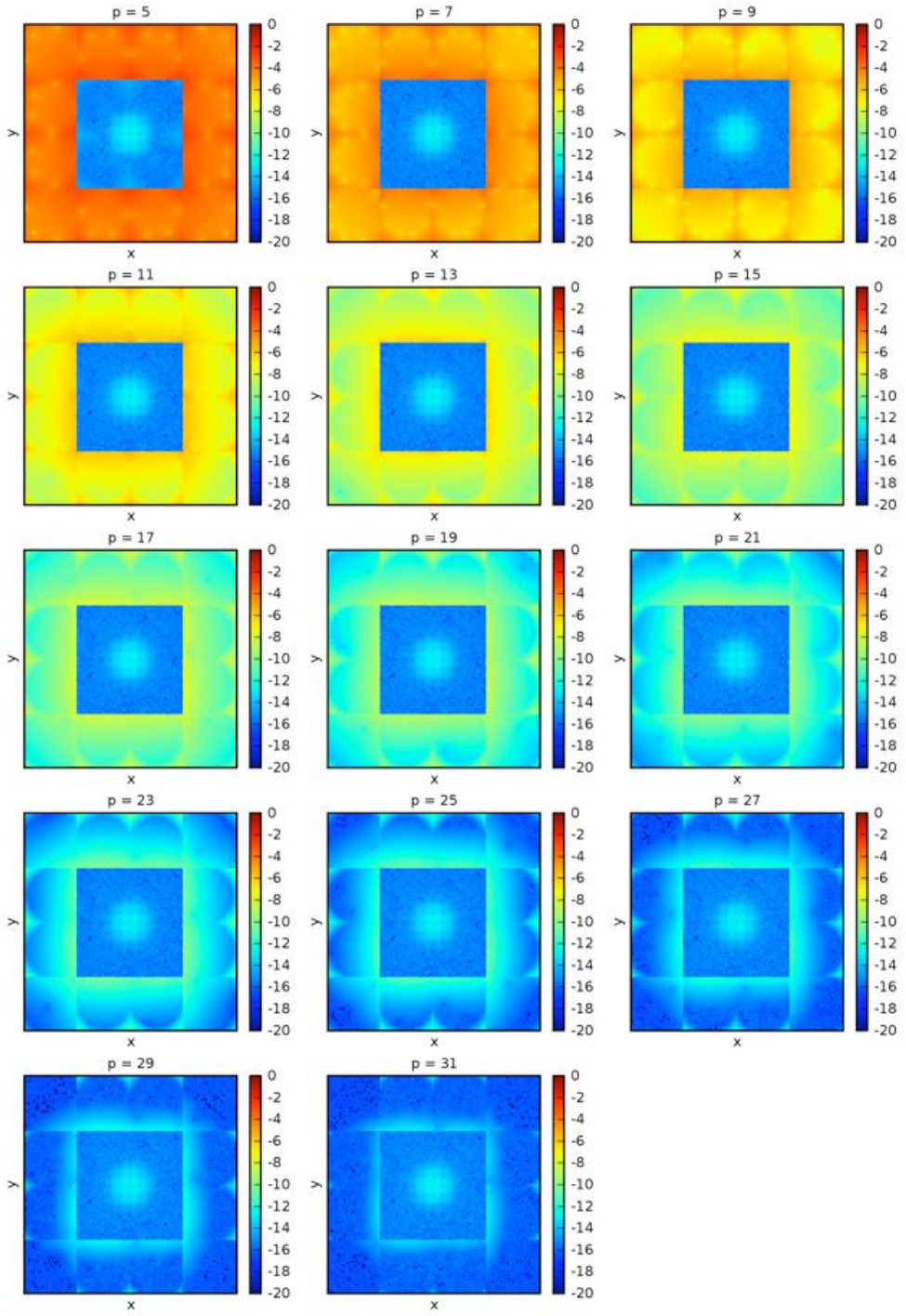


Figure 7. Evolution of the spatial distribution of the error made by the FMM when compared against the analytical solution of the test problem Lamb Oseen with problem size 11449, FMM hierarchy up to level 2 and varying the number p of Multipole Coefficients retained.

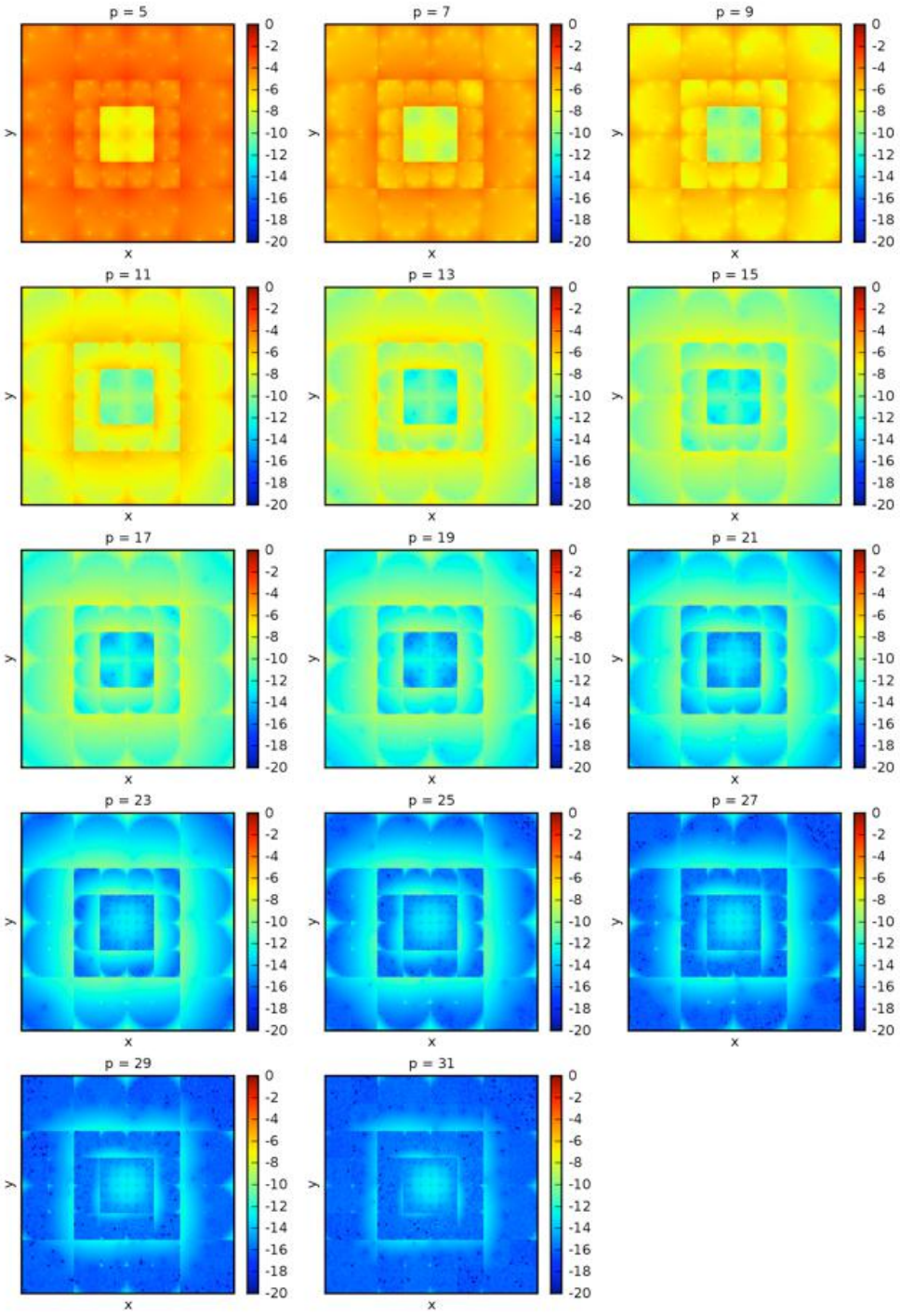


Figure 8. Evolution of the spatial distribution of the error made by the FMM when compared against the analytical solution of the test problem Lamb Oseen with problem size 11449, FMM hierarchy up to level 3 and varying the number p of Multipole Coefficients retained.

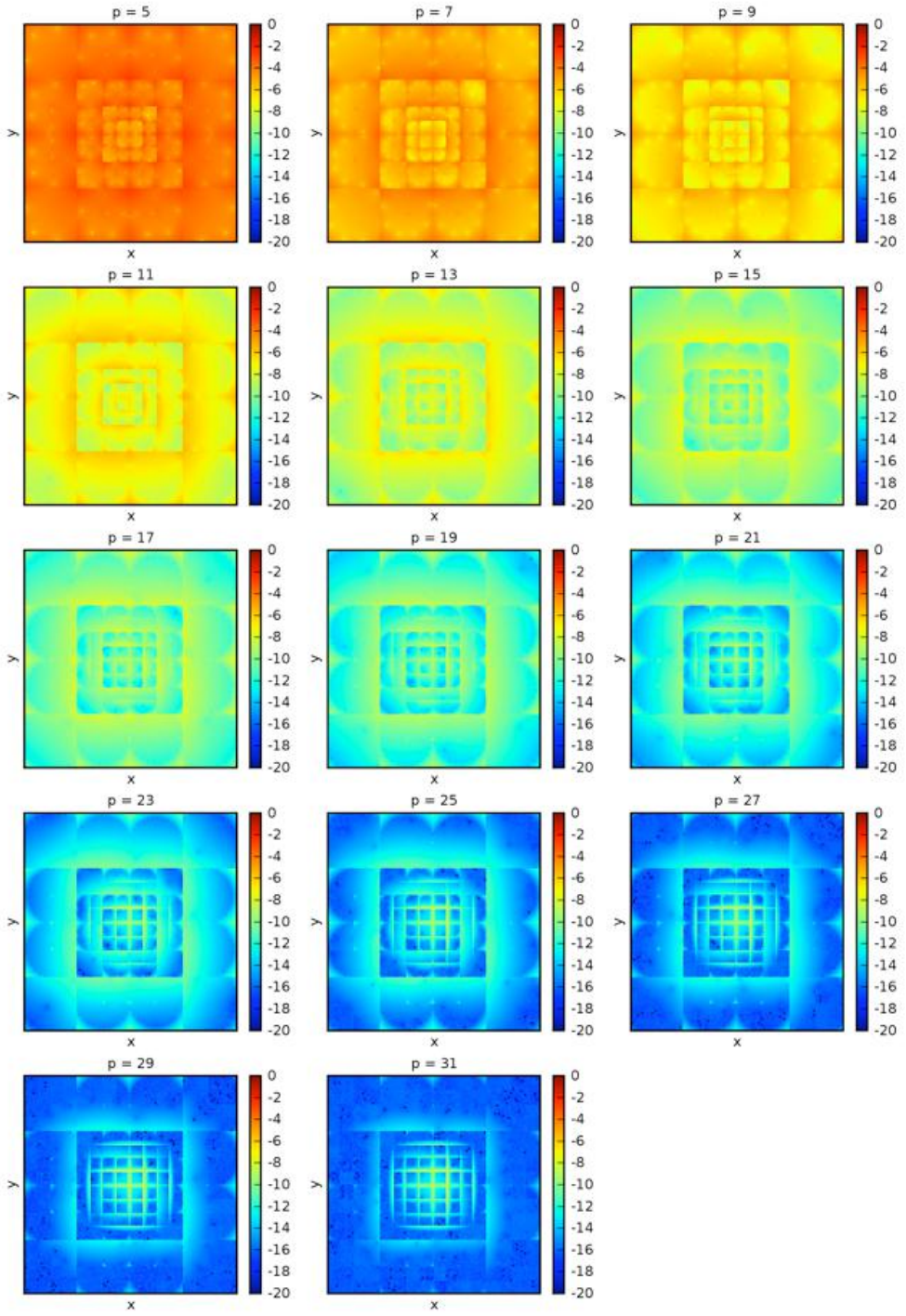


Figure 9. Evolution of the spatial distribution of the error made by the FMM when compared against the analytical solution of the test problem Lamb Oseen with problem size 11449, FMM hierarchy up to level 4 and varying the number p of Multipole Coefficients retained.

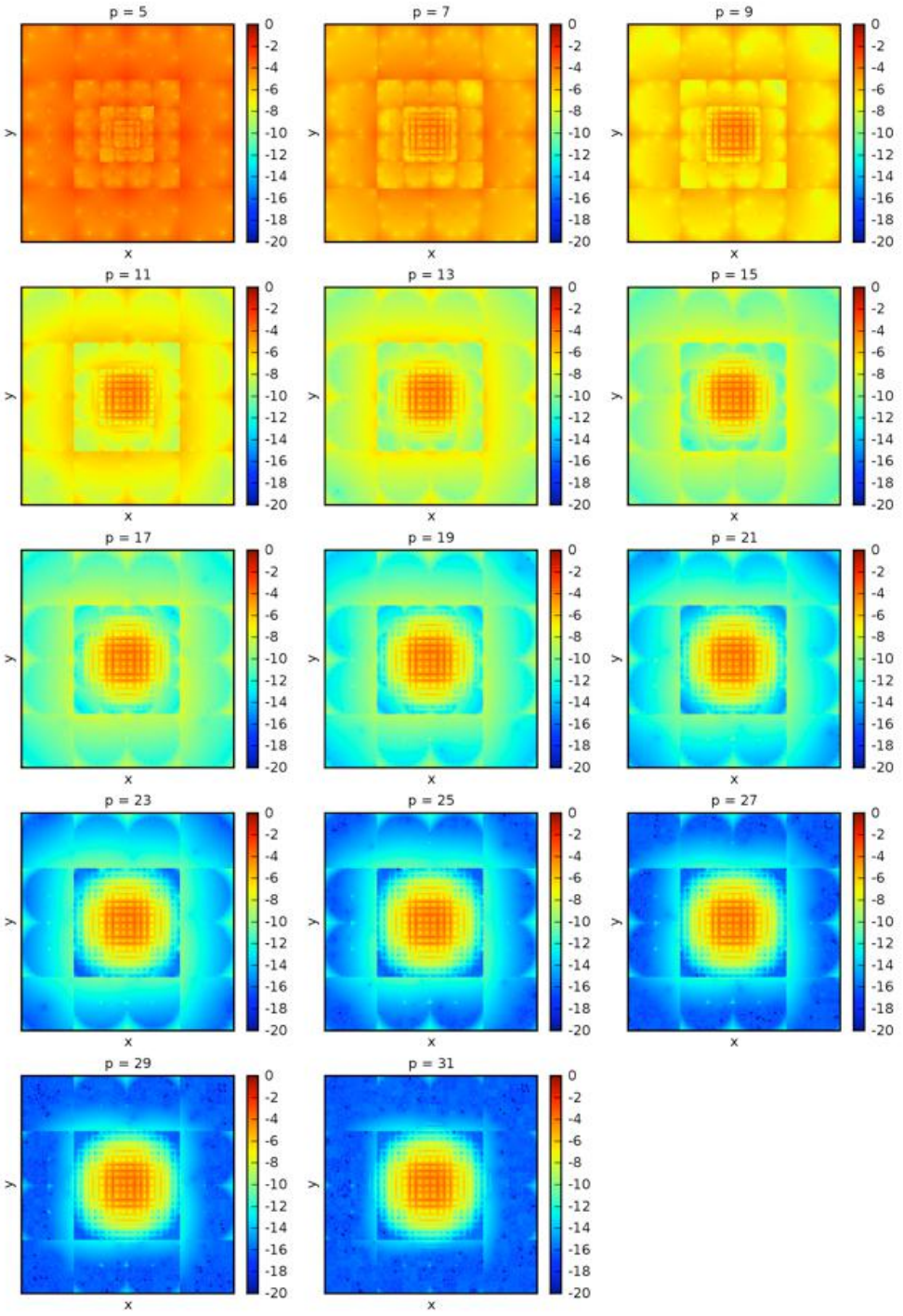


Figure 10. Evolution of the spatial distribution of the error made by the FMM when compared against the analytical solution of the test problem Lamb Oseen with problem size 11449, FMM hierarchy up to level 5 and varying the number p of Multipole Coefficients retained.

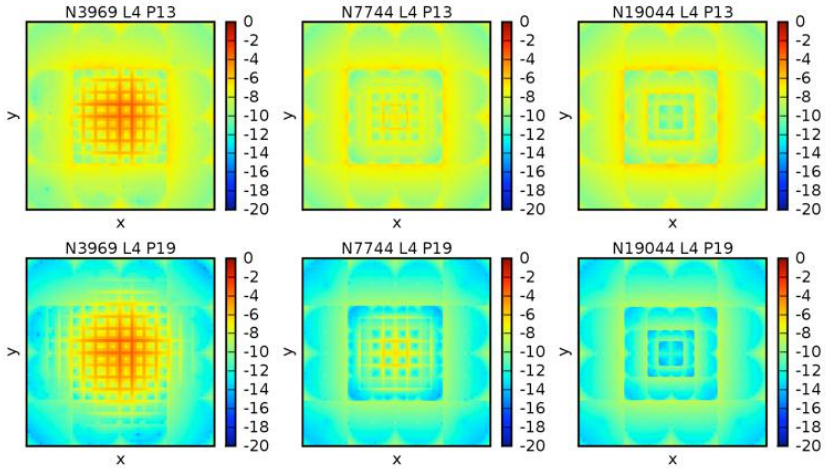


Figure 11. Evolution of the distribution of the error when the distance of the far domain approximation increases at a given level L and truncation point p. Columns show experiments for problem sizes $N = (3969, 7744, 19044)$ from left to right. First row, has parameters $l = 4$ and $p = 13$. Second row, has parameters $l = 4$ and $p = 19$.

point). Figures 12, 13, 14 present the experiments for 8836 evaluation points for levels 3, 4 and 5 respectively. Figure 15 shows the evolution of the distribution of the error when the refinement increases for a problem of fixed size.

A different view of the distribution of the error is presented in Figure 16, corresponding to a set of plots based on a LO-setup with 19044 particles for levels $2, 3, 4$, and 5. Each plot shows the measured error at each particle versus the index of the particle. It can be seen how, at level $L = 5$, error Type I first appears, above main band representing the FMM errors.

Finally, we present two plots where we compare the bounds for error Type I, §4.1, and error Type II, §4.2, against experimental setups. Figure 17 corresponds to errors Type I in an experiment based on SS-setup 5.1.2. The plot shows the maximum measured error when varying the minimum distances to far field (as explained in §4.1). Figure 18 corresponds to errors Type II based on the LO-setup with 11449 particles. The plot compares the maximum measured error on a fixed experiment, while varying the parameter p of the FMM for a fixed parameter $L = 4$.

5.3. Discussion of results

We structure the discussion of the results as follows: first we discuss the relationship between the spatial decomposition and the distribution of the measured errors, then we discuss the effects of the sources of errors by type.

5.3.1. Discussion of the spatial distribution of the error — On the figures that show the spatial distribution of the error (Figures 7, 8, 9, 10, 8, 9, and 10), at first glance, the errors seems to take a box-shaped distribution. The observed shapes are consequence of the underlying clustering chosen at the space decomposition stage. The figures of the spatial distribution of the error show that the field reconstructed by the FMM is not a continuous one.

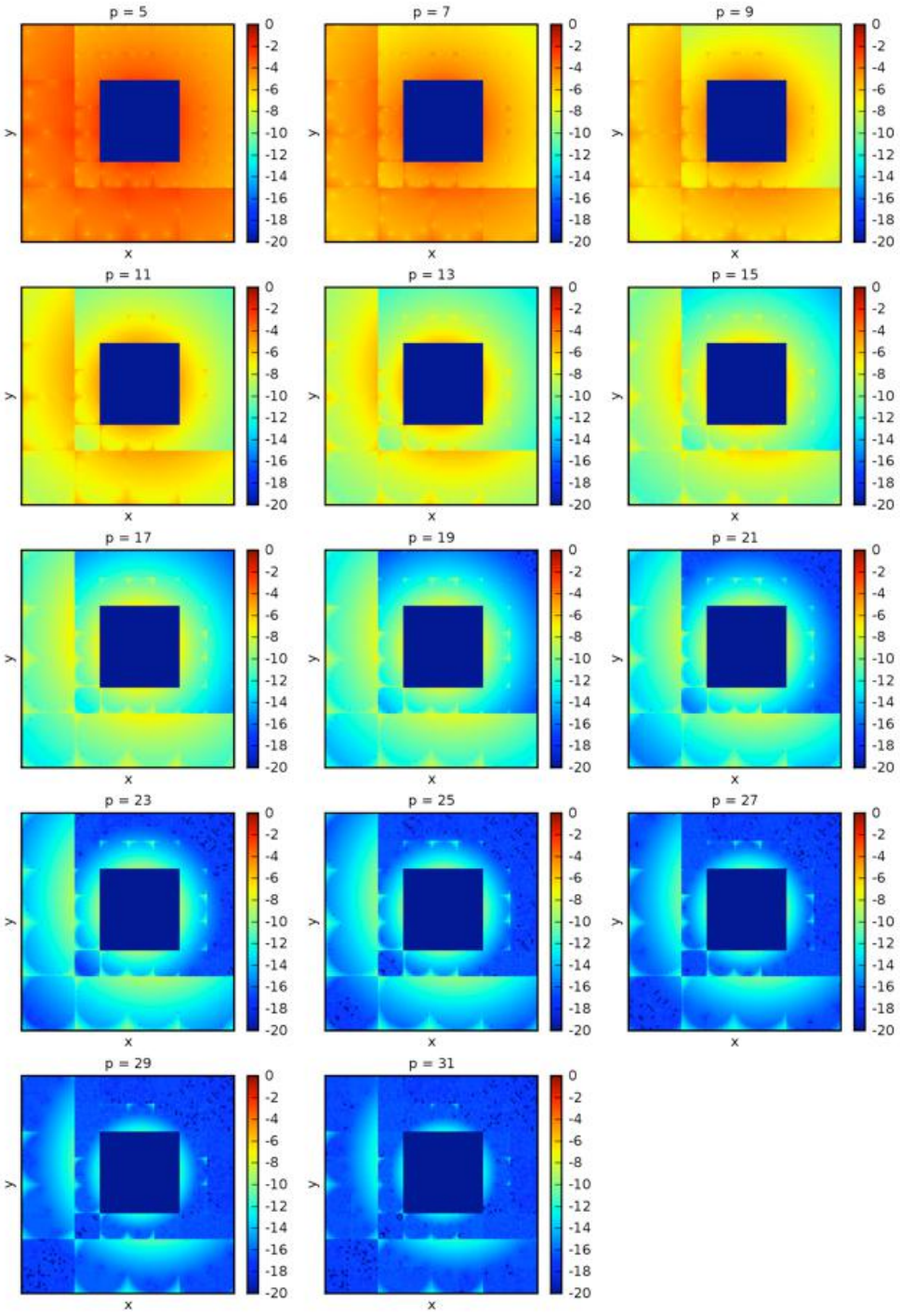


Figure 12. Evolution of the spatial distribution of the error made by the FMM for a problem with a single particle as a source and 8836 evaluation points distributed in a lattice, FMM hierarchy up to level 3 and varying the number p of Multipole Coefficients retained.

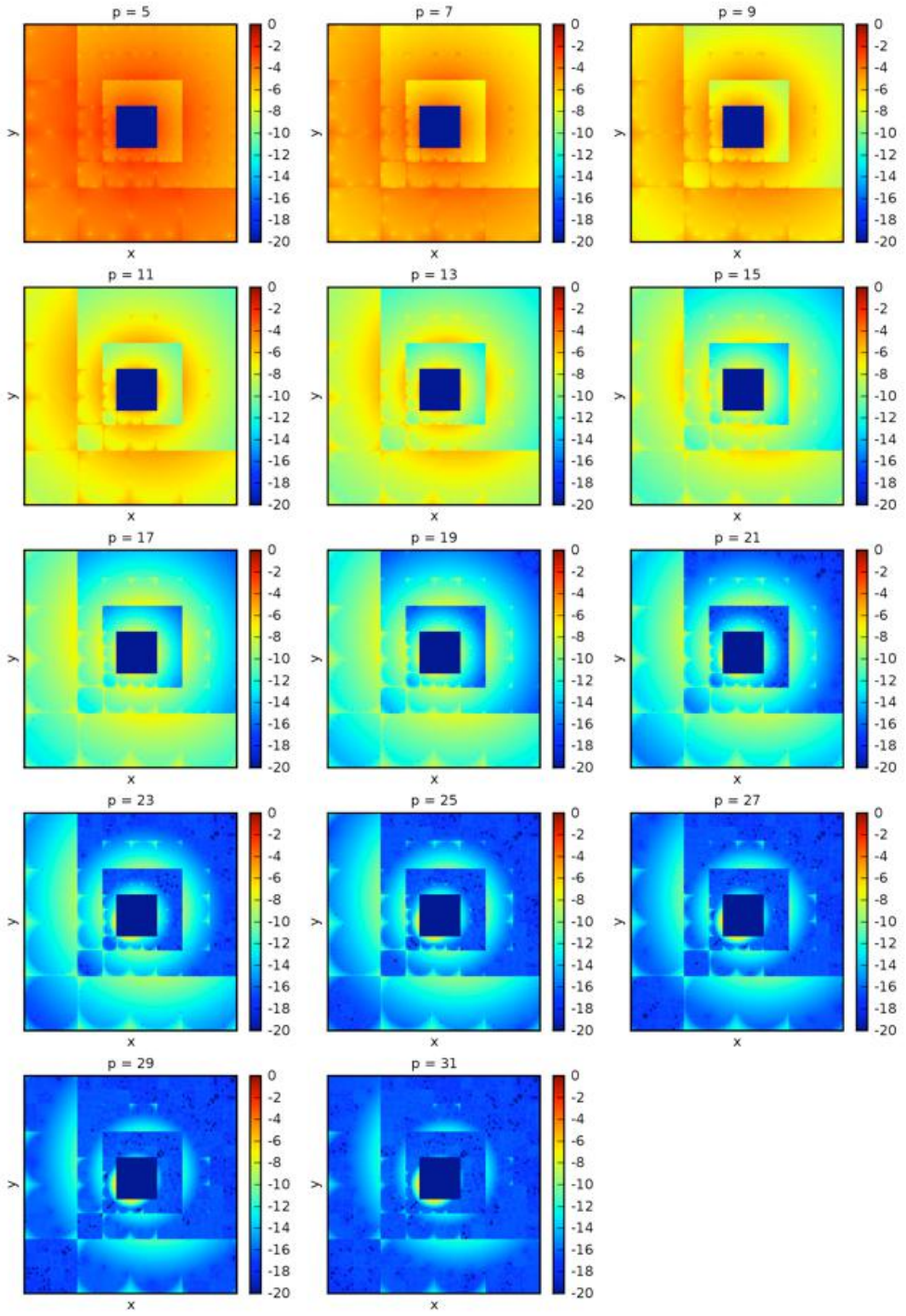


Figure 13. Evolution of the spatial distribution of the error made by the FMM for a problem with a single particle as a source and 8836 evaluation points distributed in a lattice, FMM hierarchy up to level 4 and varying the number p of Multipole Coefficients retained.

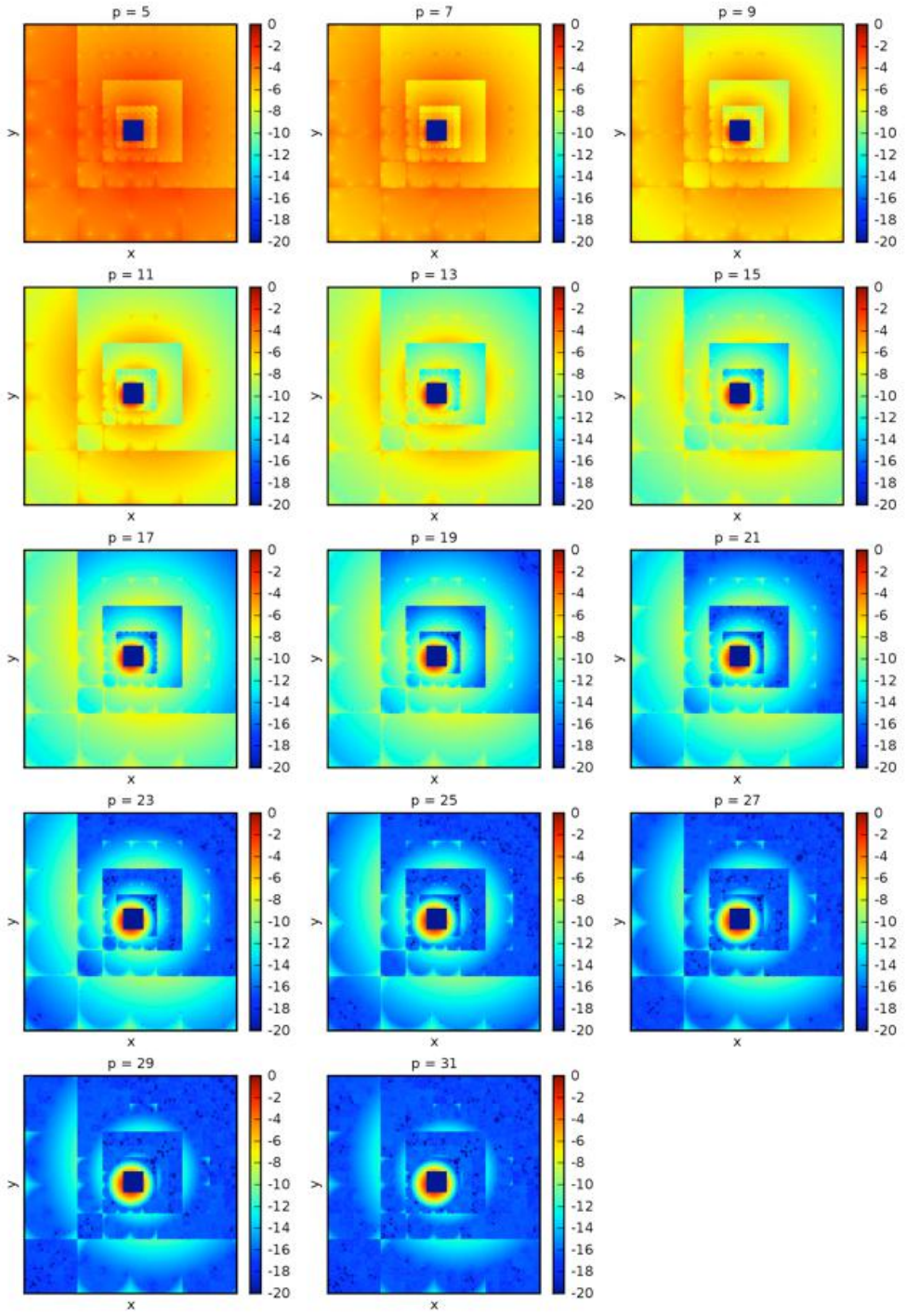


Figure 14. Evolution of the spatial distribution of the error made by the FMM for a problem with a single particle as a source and 8836 evaluation points distributed in a lattice, FMM hierarchy up to level 5 and varying the number p of Multipole Coefficients retained.

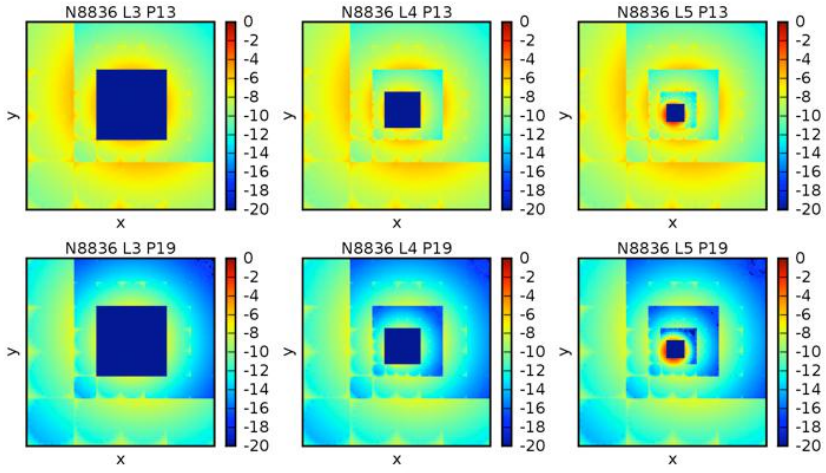
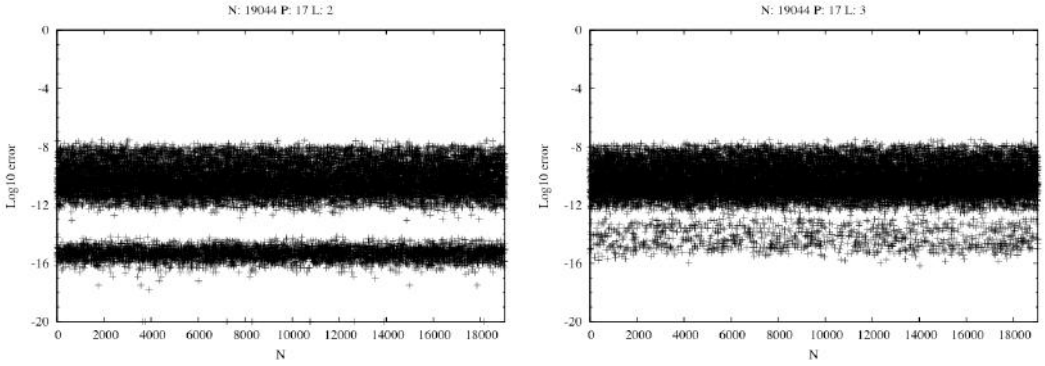


Figure 15. Evolution of the distribution of the error when the refinement increases for a problem of fixed size. In the Image is possible to see the evolution of the error as we vary the number of levels used by the FMM, columns from left to right show levels 3, 4 and 5 respectively. Row one and two compares the error against the FMM truncation numbers 13 and 19 respectively.

The accuracy of the field reconstructed by the FMM depends on the spatial decomposition and the particle distribution. Hence, how the near field and far field are constructed have an important role in the accuracy of the FMM. As explained in §3, the near and far fields vary in space, and can be significantly different for two evaluation points that are infinitesimally close to each other (when they fall in two adjacent boxes). Therefore, the spatial error distribution has the underlying geometry of the space decomposition.

As an example, let us use Figure 8; this figure is composed by a set of images that show the distribution of the error in the velocity field. Each image corresponds to an experiment based on the Lamb Oseen experimental setup of §5.1.1, where only the number of terms used by the FMM is varied from $p = 5$ to $p = 31$. In all the experiments presented in Figure 8, the initialization of all 11449 particles is the same, and the FMM level parameter is fixed at $L = 3$. In the spatial distribution of the error for the experiment with $p = 5$ (for small values of p the FMM approximation of the far field is of lower accuracy), there are two distinguishable areas of error, an inner area with errors in the order of 10^{-12}, and an outer area with errors in the order of 10^{-4}. The reason for the high contrast in the error observed between these two areas lies in the how the space is decomposed, and how the interactions are computed by the FMM for each of the evaluation points in the system. On the one hand, for the evaluation points located in the low error area, the space decomposition results in a near field —where all calculations are computed directly and without approximation— containing the source particles with the largest circulation (as presented in §2), and thus the inner area obtains a low error. On the other hand, for the evaluation points located in the high error area, all the source particles with the largest circulations are located in the far field. Therefore, almost all the contribution to the velocity comes from the source particles located in the far field and is approximated by the FMM, which for low values of p, as in this case with $p = 5$, achieves low accuracy. Finally,



(a) Level 2 (b) Level 3

(c) Level 4 (d) Level 5

Figure 16. Measured error for all the particles in a experiment based on LOsetup with 19044 particles, fixed $p = 31$. Each image shows the error for different parameter level, with $L = [2, 3, 4, 5]$.

for different values of p (Figure 8), the accuracy of the approximation improves as the number of terms p increases, as expected.

5.3.2. Discussion of error Type I — As explained in §4.1, this type of error results from neglecting the Gaussian part of the original velocity kernel exclusively for far field interactions. This type of error depends on the circulation of the source particles in the far field, and the distance to the far field (see §4.1 for details).

The first thing to note, is that errors Type I only happen for interactions with sources located in the far field of an evaluation point, as explained in §3. In the case of a source particle located in the near field of an evaluation point, the interaction with the particle is not approximated but calculated with the original kernel, therefore no error Type I is present. In order to further understand these ideas, let us look into the experiments based on SS-setup (see §5.1.2). In this setup, only one source particle is present and its influence is evaluated in the domain using a lattice of evaluation points, thus it is easy to isolate the different approximation errors. Figures 12, 13, and 14, correspond to experiments based on SS-setup for 8836 evaluation

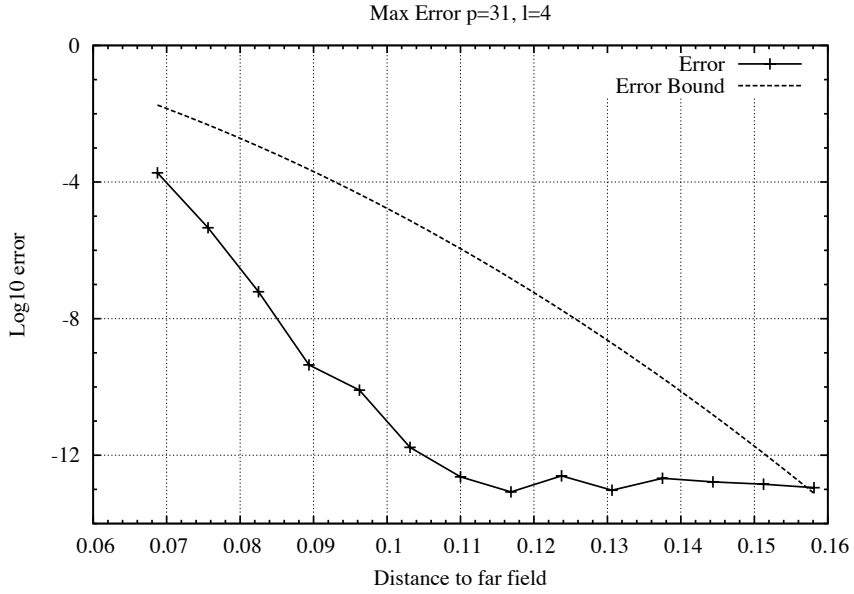


Figure 17. Experiments based on SS-setup showing the maximum measured error for a fixed set of parameters ($p = 31, L = 4$) but varying the distance to far field. Each marker represents one full evaluation of the FMM velocity. The numerical results are compared against the bound of error Type I.

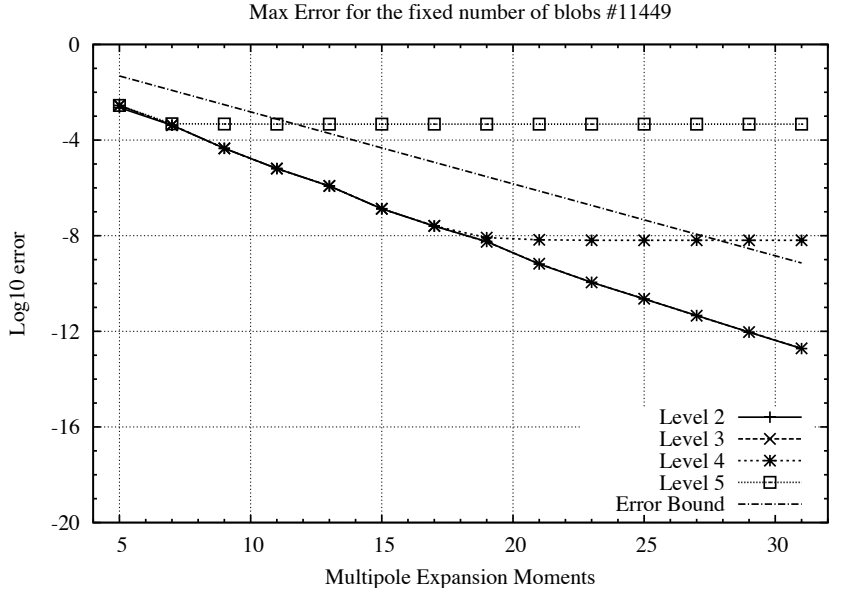


Figure 18. Experiment based on LO-setup with 11449 particles showing the maximum measured error of the FMM calculation of the velocity, with respect to the analytical value at every evaluation point. Each marker represents one full evaluation of the FMM velocity. The numerical results are compared against the bound of error Type II.

points for levels 3, 4, and 5 respectively. In the figures, a square-shaped low-error area can be observed. This area surrounds the source particle that is located at the center of the domain, and corresponds to the evaluation points that have the source particle located inside their near domain. Depending on the levels of the space decomposition, the size of the near domain varies and so does the size of the low-error area.

Let us look now at the error bound presented in §4.1. In the error bound we can directly observe the dependence of the bound with respect to the distance from the evaluation point to the source particle. Furthermore, from the bound we can expect the error to rapidly decay with the distance due to its exponential nature. This idea is confirmed in the numerical experiments; take as an example Figure 14 for $p = 31$: in the scatter plot of the error, a radially symmetric 'hot spot' of error with center on the source particle corresponds to error Type I, and as expected, the error rapidly decays with distance. In the same figure but for plots with different parameter p, this same spot can be observed but it is mixed with other sources of error and is harder to observe. This error can also be observed in Figure 13 but it is harder to see as the far field is located further away and this type of error is smaller. For the case of Figure 12, this error can no longer be observed as the distance to the far field is far enough to reduce the error to a lower level than the other errors present.

Let us look now at a more complex setup as in the case of the LO-setup (see §5.1.1). For this setup we performed the experiments presented in Figures 7, 8, 9, and 10. Every one of these experiments has 11449 source particles with non-zero circulation, thus there are 11449 sources of error Type I. Only the number of levels L varies among the figures, taking values of 2, 3, 4, and 5 respectively. For the LO-setup with multiple sources of error, the distribution of error Type I takes a shape that is significantly different from the previously presented 'hot spot' structure. For example, let us take the case of Figure 10 with $p = 31$; in this plot we can observe a 'grid' of high errors. The grid-shape is conformed by the particles at the borders of the boxes at the maximum level of refinement (L) in the quadtree. The error concentrates at the border of the clusters which accumulate the maximum of error Type I, as they are closer to the sources of the errors. As this type of error rapidly decays with the distance, points located at the center of a cluster are less exposed to this type of error.

The previous examples demonstrate that the distance between the source particles and the evaluation points is the driving factor behind error Type I, and that the error rapidly decays with the distance. These ideas are presented more clearly if we take the bound for the error Type I developed in §4.1 and compare it against measured errors in experimental setups. Figure 17 presents the measured error for numerical experiments based on SS-setup 5.1.2. In the experiments, we vary the distance to the far field while we keep all the other parameters the same: 1 source particle, 8836 evaluation points, and parameters of the FMM $p = 31$ and $L = 4$. Under this setup, the maximum errors correspond to errors Type I for all values of distance to far field smaller than 0.12; for values larger than this, the maximum error is dominated by errors Type II. It can also be seen that the bound holds, but the bound is overestimating the errors by a couple of orders of magnitude. The reason for this is that the source particle is located at a larger distance from the evaluation point than the estimated distance to the far field.

5.3.3. Discussion of error Type II — As explained in §4.2, this type of error is the product of truncating the infinite series that forms the Multipole Expansion that represents the far field interactions. The approximation depends on the number of terms (p) that are kept, and

the composition of the far field (on the particle distribution and circulation). The accuracy of the ME can be controlled by the user by choosing the number of terms kept in the expansion. In this way, any accuracy can potentially be achieved.

This type of error is determined by the accuracy of the Multipole Expansions used to represent the influence of the particles contained in the far field. Looking at the error bound in §4.2, the error Type II also depends on the ratio between the radius of the cluster and the distance from the cluster to an evaluation point $(r/(x_*-y))$, and the circulation of the particles that are inside the cluster. All of these factors can be controlled, so any given accuracy can potentially be achieved. Of all the factors, the number of terms kept by the ME is the more important one for controlling this error, as the other two factors are up to certain extent less flexible as they greatly rely on the spatial decomposition (near field and far field construction, and clustering of particles).

Knowing the factors that control error Type II, we can now observe the figures for the error distribution. The number of terms kept by the ME has a straight forward effect on the accuracy. Examples of varying p for different experimental parameters are shown in Figures 5 and 6. From the plots in both figures, we can see that by increasing p the accuracy of the method improves, as expected. Note that this happens for all the cases where error Type I is not dominant, when error Type I is dominant; the maximum error stays constant independently of how many terms are kept by the ME.

It is also worth noting the major role of the spatial decomposition on this type of error. As example, let us take the experiments presented in Figure 7 for lower values of p (say $p = [5, 7, 9, 11, 13, 15]$). In these plots, a clear distinction between two areas of high and low error can be appreciated, product of the differences in the spatial decomposition for different evaluation points. On the one hand, we have a square-shaped inner area of low error; the main contribution on the velocity for the points in this area is obtained by direct interactions, and a small contribution is added by the FMM. On the other hand, an outer area of high errors, where the main contribution for the evaluation of the velocity at these points comes from the source particles located in the far field, thus the ME approximation plays an important role in this case. It can also be seen that as the ME improves, so does the overall error (see same Fig. 7 for $p = [27, 29, 31]$).

6. CONCLUSIONS

The Fast Multipole Method (FMM) has offered dramatic increase in the computation capability for all processes dominated by pair-wise interactions, or N-body problems. In many applications, such as gravitational systems, high accuracy may not be an issue, and furthermore such a system is dominated strongly by the first moment (due to the fact that all mass is positive). In other applications, the accuracy of the overall numerical scheme may have been dominated by factors such as spatial discretization or time stepping errors. However, as high-order methods become more commonplace, and as the availability of powerful computers becomes more widespread, scientists may have the need to know and control the accuracy of the FMM approximation more finely.

The literature on the FMM algorithm supplies theoretical error bounds. However these may not be very tight and moreover they are not very illustrative as to the behavior of the method, except for providing simple ideas like a decrease of error with increasing truncation number.

In the literature, one rarely finds reports of *measured* errors when using the algorithm. This of course is not practical in most cases, as a comparison with either an analytical solution or with the direct $\mathcal{O}(N^2)$ summation would be required. We take precisely this approach, and use some simple experimental setups to discover the behavior of the FMM errors with respect to the different parameters and choices that the user would normally make: number of particles, levels in the tree, truncation level.

Users of the FMM will have a *client* application which will provide them with a given kernel, representing the influence among particles. One can approximate the kernel by another (such as $\frac{1}{r}$) with good accuracy, but this implies the need to keep box sizes at the deepest level large compared with the kernel width. The kernel substitution has the aim of simplyfying the mathematics of the expansions. It makes sense to reduce the human effort required for the implementation in this way, but only if the consequences of the approximation are well understood by the practitioner. We believe that the experiments presented in this work illustrate very well the effects of approximating the force or influence kernel by a simpler one which applies in the far field.

The spatial distribution of the error incurred by the FMM offers visual evidence of all the contributing factors to the overall approximation accuracy: multipole expansion, local expansion, hierarchical spatial decomposition (interaction lists, local domain, far domain). This presentation is a contribution to any researcher wishing to incorporate the FMM acceleration to their application code, as it aids in understanding where accuracy is gained or compromised.

Potential users of the FMM may wish to perform some experiments similar to those presented here, varying algorithm parameters, to get a good grasp of the error behavior for their problems. In this sense, we offer an experimental design which will help users apply the FMM algorithm with good knowledge of what it is doing.

In addition to the understanding of the accuracy of the FMM approximation, a user would also require knowledge of the efficiency obtained, as of course there is always a trade-off between accuracy and computational efficiency. We have not presented speed-up tests or measures of the calculation times, because using a Python prototype which is not optimal in terms of speed may offer misleading results. Moreover, the problem sizes which we manage are small and likely not in the range of noticeable acceleration with the FMM. Realistically, one will observe the expected speed-up of the algorithm with tens or hundreds of thousands of particles. In this range of problem size, experimentation aiming to measure the errors of the algorithm would be very cumbersome (unless an analytical solution were available).

In conclusion, we suggest that whenever the application calls for controlled and high accuracy, the user may want to follow the example of this work to characterize the errors for his/her application, and then follow this by some speed up tests at larger problem sizes.

We are making our Python FMM code available (via Google Code) to the community, and welcome any correspondence with interested readers.

To end, we offer an opinion with regards to the areas of future progress in the field. The FMM algorithm is mature and optimal, but still rather difficult to program. We anticipate that there remain the following areas for progress:

1. Algorithm acceleration through hardware. There are already a few researchers investigating, for example, the use of graphical processing units (GPUs) with the FMM. We are also carrying out some work in this area.
2. Better representation of kernels, that provide more accuracy with less truncation number.

As already mentioned, the Taylor series are the easiest to deal with mathematically, but converge rather slowly. In applications demanding high accuracy, perhaps other series representations would provide a better solution, as long as the human effort required for the mathematical derivations and programming are acceptable.

3. Generality and portability. We refer to having tools that work with different kernels and the availability of software library components for wider dissemination and impact of the algorithm in multiple applications. In this context, we have initiated collaboration with the development team of the PETSc library to produce a parallel and portable version of the FMM which can be distributed to the wider community of computational scientists. This work will be available in the coming months.

ACKNOWLEDGEMENTS

FAC acknowledges support from the SCAT project via EuropeAid contract II-0537-FC-FA, www.scat-alfa.eu for an extended research visit during which this work was initiated. LAB acknowledges support from EPSRC under grant contract EP/E033083/1.

REFERENCES

1. Andrew W. Appel. An efficient program for many-body simulation. *SIAM Journal on Scientific and Statistical Computing*, 6(1):85–103, 1985.
2. S. Balay, K. Buschelman, W. D. Gropp, D. Kaushik, M. Knepley, L. Curfman-McInnes, B. F. Smith, and H. Zhang. PETSc User's Manual. Technical Report ANL-95/11 - Revision 2.1.5, Argonne National Laboratory, 2002.
3. L. A. Barba, A. Leonard, and C. B. Allen. Numerical investigations on the accuracy of the vortex method with and without remeshing. AIAA #2003-3426, 16^{th} CFD Conference, Orlando FL, June 2003.
4. L. A. Barba, A. Leonard, and C. B. Allen. Advances in viscous vortex methods – meshless spatial adaption based on radial basis function interpolation. *Int. J. Num. Meth. Fluids*, 47(5):387–421, 2005.
5. J. Barnes and P. Hut, P. A hierarchical $O(N \log N)$ force-calculation algorithm. *Nature*, 324:446–449, December 1986.
6. P. Chatelain, A. Curioni, M. Bergdorf, D. Rossinelli, W. Andreoni, and P. Koumoutsakos. Billion vortex particle direct numerical simulations of aircraft wakes. *J. Comp. Phys.*, 197(13–16):1296–1304, 2008.
7. G.-H. Cottet and P. Koumoutsakos. *Vortex Methods. Theory and Practice*. Cambridge University Press, 2000.
8. Felipe A. Cruz, L. A. Barba, and Matthew G. Knepley. Fast multipole method for particle interactions: an open source parallel library component. 20th Parallel Computational Fluid Dynamics Conference, Lyon, May 2008.
9. J. Dongarra and F. Sullivan. The top 10 algorithms of the twentieth century. *Computing in Science and Engineering*, 2(1):22–23, January/February 2000.
10. Marcia O. Fenley, Wilma K. Olson, Kiat Chua, and Alexander H. Boschitsc. Fast adaptive multipole method for computation of electrostatic energy in simulations of polyelectrolyte DNA. *J. Comput. Chem.*, 17(8):976–991, 1996.
11. C. Gáspar. A multipole expansion technique in solving boundary integral equations. *Comput. Methods Appl. Mech. Engrg.*, 157:289–297, 1998.
12. L. Greengard and V. Rokhlin. A fast algorithm for particle simulations. *J. Comput. Phys.*, 73(2):325–348, 1987.
13. John T. Hamilton and George Majda. On the Rokhlin-Greengard method with vortex blobs for problems posed in all space or periodic in one direction. *J. Comput. Phys.*, 121(1):29–50, 1995.