PERFORMANCE OF A VORTEX MERGING ALGORITHM USING THE COMPUTATIONAL BOX OF THE ADAPTIVE FAST MULTIPOLE METHOD

Victor Santoro Santiago, vsantoro@ig.com.br
Gustavo C. R. Bodstein, gustavo@mecanica.coppe.ufrj.br
Federal University of Rio de Janeiro, Department of Mechanical Engineering – Politécnica/COPPE
Centro de Tecnologia, Bloco G, sala 204 – Ilha do Fundão, 21945-970 Rio de Janeiro, RJ – Brazil

Abstract. *The Vortex Method is based on the discretization of the vorticity field as a superposition of Lamb vortices that move within a Lagrangian framework to simulate the convective and diffusive transports of vorticity in the flow. Among the several vorticity diffusion methods available in the literature, the Corrected Core-Spreading Method is characterized by the expansion of the vortex core radius in time, followed by the splitting of the vortex into four new vortices with smaller cores as soon as the radius reaches a maximum value. The splitting process results in an exponential growth of the number of vortices, N. This problem may be circumvented through the use of an algorithm that merges groups of vortices with a high degree of spatial overlapping. However, this procedure requires that the distance between vortices be calculated, producing a floating-point operation count of the order of N^2. For large N, this step becomes a bottleneck for long-time vortex-method simulations. In this paper we propose to merge vortices located in small meshes, or boxes, obtained by refining the largest computational box, drawn to encompass all vortices in the flow, into a hierarchy of smaller boxes, with a non-uniform size and spatial distribution. This hierarchy of boxes was originally developed for the Adaptive Fast Multipole Method (AFMM) to accelerate the induced velocity calculation at each vortex location, since the direct calculation using the Biot-Savart law also requires an operation count of the order of N^2. The use of the AFMM boxes produces a large reduction of the CPU time required to merge vortices in the cloud. Results for several clouds of vortices with different non-uniform spatial distributions are presented and their CPU times are compared to those obtained with the direct merging algorithm, which does not use the AFMM boxes.*

Keywords*: vortex method, corrected core-spreading method, fast multipole method, vortex merging.*

1. INTRODUCTION

The Vortex Method (VM) is a numerical approach for the simulation of incompressible flows that solves the Navier-Stokes equations in terms of the vorticity field, instead of the velocity field (Barba *et al*., 2005). The vorticity field is discretized into a cloud of Lamb vortices, also called Gaussian basis functions (Rossi, 1996), which move with the fluid within a Lagrangian framework to simulate the convective and diffusive transports of vorticity in the flow. Vortex methods make use of the Helmholtz theorems that state that in an inviscid incompressible flow a vortex tube convects with the fluid and its strength remains constant. From a given discretized vorticity field $\boldsymbol{\omega}$, the velocity field $\mathbf{u}$ may be obtained by inverting the mathematical definition of the vorticity vector, $\boldsymbol{\omega} \equiv \nabla \times \mathbf{u}$, to produce the velocity field in terms of an integral over the vorticity field, well known as the Biot-Savart law. The vortex method converges if the vortex core radius σ tends to zero (Rossi, 1996) and its accuracy depends on whether the overlap ratio h/σ is small enough (Barba *et al*., 2005), where h is the inter-particle spacing. The latter requirement means that high core overlapping must exist for high resolution simulations to be carried out.

In order to convect the vortices in space as Lagrangian particles the velocity vector at the centroid of each vortex must be determined. The use of the Biot-Savart law for the velocity calculation due to the vortex-vortex interaction produces a numerical scheme with a floating-point operation count of order N^2, if the vortex cloud has N elements. This difficulty has been overcome by the application of the Adaptive Fast Multipole Method (AFMM), devised by Carrier *et al*. (1988), which consists of clustering vortices into a computational box divided up into a hierarchy of meshes, or smaller boxes, that expresses different mesh levels for particles located at different spatial lengths. The refinement process produces boxes with a non-uniform size distribution in space. The algorithm consists of using multipole expansions to evaluate the interactions between boxes that are sufficiently far away from each other – the Box-Box scheme, whereas interactions between nearby particles are computed directly. This procedure transforms a particle-particle interaction scheme into a box-box interaction scheme, which has an operation count of order N.

At least seven numerical methods are available in the literature to simulate vorticity diffusion using particles (Barba *et al*., 2005). The Corrected Core-Spreading Method (Rossi, 1996) simulates vorticity diffusion through the expansion of the vortex core radius in time and its subsequent splitting into four new vortices as soon as the core radius reaches a prescribed maximum value. The splitting process creates four new vortices around the original vortex, positioned within 90° from each other and with smaller core sizes. If no action is taken this procedure results in an exponential growth of the number of vortices. To attenuate this problem Rossi (1997) proposed an algorithm that merges groups of vortices with a high degree of overlapping in space. Similarly to the Biot-Savart direct vortex-vortex velocity calculation,

Rossi's scheme requires the computation of the distance between vortices, producing again an N^2 operation count. For large N, the merging step becomes a bottleneck for long-time vortex-method simulations.

The idea of this paper is to use the AFMM computational box, with its associate hierarchy of boxes, to merge vortices located in each box instead of merging vortices in the cloud directly. This algorithm produces a large reduction of the CPU time required to merge vortices in the cloud. As shown below, results for several types of vortex clouds with different non-uniform spatial distributions are presented and their CPU times are compared to those obtained with the direct merging algorithm that does not use the AFMM boxes.

2. THE VORTEX METHOD

The vortex method employs a cloud of vortices to discretize the vorticity field and simulate the flow dynamics. Through the Lagrangian motion of these vortices, the velocity field can be evaluated everywhere. For two-dimensional flows, the VM attempts to approximate the solution to the vorticity transport equation in the form

$$\frac{\mathrm{D}\omega}{\mathrm{D}t} \equiv \frac{\partial \omega}{\partial t} + \mathbf{u}\cdot\nabla\omega = \nu\nabla^2\omega\,, \tag{1}$$

through the superposition of Lamb vortices with a Gaussian vorticity distribution $\omega(\mathbf{x})$, given by

$$\omega(\mathbf{x}) = \sum_{i=1}^{N} \frac{\Gamma_i}{4\pi\sigma_i^2} \exp\left(-\frac{|\mathbf{x}-\mathbf{x}_i|^2}{4\sigma_i^2}\right), \tag{2}$$

where Γ_i is the strength of vortex i with core radius σ_i, for $1 \le i \le N$. Equation (2) is an exact solution to the Navier-Stokes equation for a point vortex in an infinite domain that undergoes viscous diffusion with $\sigma_i \equiv (\nu t)^{1/2}$. To capture the solution of Eq. (1), each vortex in the cloud must convect and diffuse, according to the following system of ordinary differential equations

$$\frac{d\mathbf{x}_i}{dt} = \begin{bmatrix} 0 & -1 \\ 1 & 0 \end{bmatrix} \sum_{j=1}^{N} \frac{\Gamma_j}{2\pi} \frac{\mathbf{x}_i - \mathbf{x}_j}{|\mathbf{x}_i - \mathbf{x}_j|^2} \left[1 - \exp\left(-\frac{|\mathbf{x}_i - \mathbf{x}_j|^2}{4\sigma_i^2}\right)\right], \tag{3}$$

$$\frac{d\sigma_i^2}{dt} = \nu\,. \tag{4}$$

The body contribution to the total flow can be calculated to satisfy the continuity equation, the no-penetration boundary condition and the boundary condition at infinity. To this end, the Panel Method (Katz and Plotkin, 2001) is used very often due to its ability to handle bodies with general shape. Because vorticity is generated on the body surface, vortices are generated close to the surface such that the no-slip boundary condition and the condition of conservation of circulation are enforced. The trajectory of each vortex is computed from the integration of Eq. (3) using, in general, the 2nd-order Adams-Basforth time-marching scheme.

Using the AFMM instead of the Biot-Savart law the induced velocities at each vortex centroid are calculated. The AFMM algorithm reduces the operation count to order N instead of order N^2, as shown by Santiago *et al*. (2006) for typical wakes obtained from 2D flow simulations around bodies. The diffusive vorticity transport using the Corrected Core-Spreading method (Rossi 1996) is based on Eq. (4). The original core-spreading method (Leonard, 1980) simulates vorticity diffusion through the core expansion of the vortex. Greengard (1985) showed that the solution obtained from this procedure does not converge to the Navier-Stokes equation. Rossi's Corrected Core-Spreading method (1996) turns Leonard's scheme convergent by splitting the vortex into four overlapping vortices with smaller cores. More details of the VM can be found in Cottet and Koumoutsakos (1999) and Silva and Bodstein (2005). Figure 1 shows the basic algorithm of the VM.

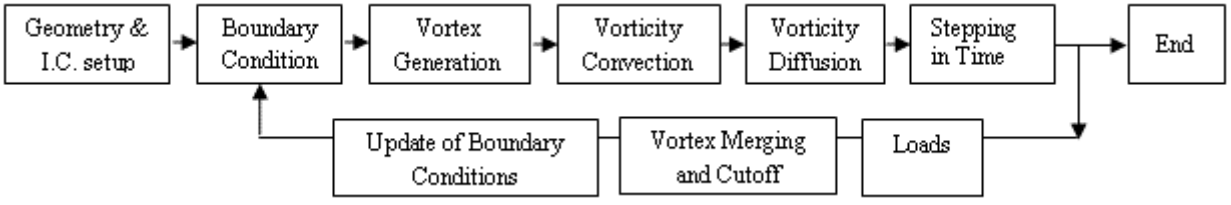

Figure 1. Basic Algorithm of the Vortex Method.

2.1. Convection Step: Adaptive Fast Multipole Method (AFMM)

The calculation of the velocity at the centroid of each vortex in the cloud is performed using the Adaptive Fast Multipole Method (Carrier *et al*. 1988). The idea behind the AFMM is to group the cloud vortices in boxes of different sizes according to the local density of vortices and evaluate the interaction between boxes that are far enough apart. The evaluation of the interaction between vortices that belong to either the same box or to adjacent boxes is performed directly using the Biot-Savart law. In summary, the AFMM may be described as follows. Suppose there are m particles with strength q_i, located at z_i, $i = 1, \ldots, m$, such that $z_i < r$, where r is the radius of a finite region in the cloud containing the particles. For all $z \in C$, where C the set of complex numbers, and $z_i > r$, the complex potential $\phi(z)$ defined by the cloud of particles can be written as

$$\phi(z) = Q\log(z) + \sum_{k=1}^{\infty} \frac{a_k}{z^k}, \tag{5}$$

with $Q = \sum_{i=1}^{m} q_i$ and $a_k = \sum_{i=1}^{m} \frac{q_i z_i^k}{k}$. Truncating the summation in Eq. (5) after p terms, we may write, for $p \geq 1$, that

$$\left| \phi(z) - Q\log(z) - \sum_{k=1}^{p} \frac{a_k}{z^k} \right| \leq \alpha \left| \frac{r}{z} \right|^{p+1} \leq \left(\frac{A}{c-1} \right) \left(\frac{1}{c} \right)^p, \tag{6}$$

where $c = \left| \frac{z}{r} \right|$, $A = \sum_{i=1}^{m} |q_i|$ and $\alpha = \frac{A}{1 - |r/z|}$.

Carrier *et al.* (1988) consider $c = 2$ for boxes that are well separated. Therefore, in order to obtain a relative precision ε, p must be of the order $-\log_2 \varepsilon$ according to Eq. (6). The algorithm begins with the choice of ε, which determines the number of terms p in the multipole expansion. The domain is then divided sequentially into square boxes up to a point where the number of vortices in each box is less than a maximum prescribed value. This parameter has influence on the algorithm performance, but it does not imply in loss of accuracy. A list of criteria is then specified to establish the box hierarchy according to box size and distance to its neighbors.

The algorithm starts by dividing up the fluid domain, or computational box, in four square boxes, which are divided again in four more boxes and so on, until each box contains a number of particles less than the maximum value. A tree structure is imposed on this box hierarchy such that several levels of boxes are formed, called parents, children or colleagues, depending on whether they are the boxes that gave birth to, were born from or belong to the same stage of the division process (that is, those that have the same size as the box being considered), respectively. At the end of the division process of the fluid domain, each box has five lists of boxes associated to it, as illustrated in Fig. (2), where all five lists, numbered 1 to 5, are related to box b in the following manner: List 1 of a box b is empty if b is a parent box and, if b is childless, it consists of b and all childless boxes adjacent to b; List 2 of a box b is formed by all the children of the colleagues of b's parent that are well separated from b; List 3 of a box b is empty if b is a parent box, and consists of all descendants of b's colleagues whose parents are adjacent to b, but who are not adjacent to b themselves, if b is a childless box; List 4 of a box b is formed by all boxes that have b in their List 3; finally, List 5 of a box b consists of all boxes that are well separated from b's parents. The reader is referred to the original paper by Carrier *et al*. (1988) for a detailed description of the AFMM algorithm.

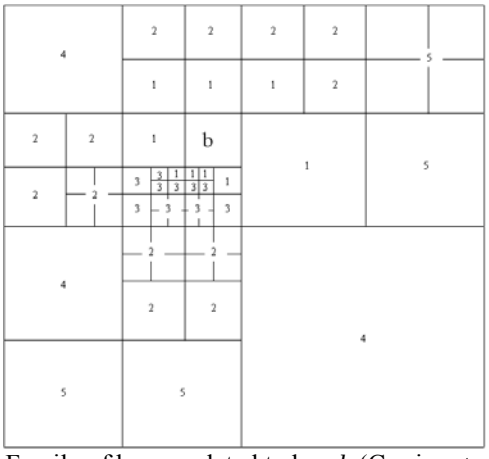

Figure 2. Family of boxes related to box b (Carrier *et al*., 1988).

Santiago *et al*. (2006) applied the original AFMM Fortran code developed by Carrier *et al*. (1988) to several different patterns of body wakes and concluded that the CPU time obtained with the AFMM algorithm indeed reduces from order N^2 to order N the operation count of the induced velocity calculation for large vortex clouds. The RMS error, defined with respect to the direct vortex-vortex Biot-Savart calculation, obtained for all test cases were kept below 10^{-11}, showing the expected accuracy of the AFMM algorithm.

In this work we demonstrate that the use of the AFMM box hierarchy to conduct the merging process of overlapping vortices within each box instead of merging directly the vortices of the entire cloud is extremely more efficient. As we show below for several test cases, the AFMM box structure provides and efficient grid to merge vortices of highly non-uniform wake patterns.

2.2. Diffusive step: Corrected Core-Spreading Method (CCSM)

The VM adoption of a Lagrangian flow description adds the advantage of turning the method mesh-free. On the other hand, simulating diffusion without a mesh is not as simple as simulating diffusion on a mesh. Historically, the first attempt to simulate vorticity diffusion using vortices is due to Chorin (1973), who devised the Random-Walk Method (RWM) based on a Brownian motion of particles. The idea is to simulate the vorticity diffusion transport through random displacements of the vortices calculated from a Gaussian probability distribution, which is an interpretation of the solution to the diffusion equation. The RWM is grid-free, easily and efficiently implemented in a numerical code and, for these reasons, has been widely applied to engineering problems. However, the RWM relies on the operator splitting of the vorticity equation, presents low convergence rates, needs a large number of particles to obtain reasonable accuracy, requires averaging over several time steps to calculate the loadings because of its stochastic nature, and is intended only for high Reynolds number flows.

As discussed in detail by Barba *et al.* (2005), there are many proven Lagrangian numerical methods available in the literature to simulate vorticity diffusion nowadays. The Corrected Core-Spreading Method – CCSM (Rossi, 1996) is a fully localized, Lagrangian and deterministic method that is very simple to implement, has been proven to converge and provides spatial refinement automatically because of the splitting of the vortices. The method is based on the original core-spreading method (Leonard, 1980), which allows the vortex core radius to increase in time according to Eq. (4). Leonard's core-spreading method is inconsistent (Greengard, 1985) because it simulates convection without deformation of larger and larger vortices as they spread. Rossi proposed a correction to Leonard's method that allows the vortex to split as soon as the core radius reaches a prescribed maximum value and, as a consequence, turns the method consistent. Rossi's CCSM has an additional advantage that does not require the operator splitting of the vorticity equation as the RWM method does. The RWM operator splitting algorithm does not affect convergence but introduces a numerical error that is not present in the CCSM.

The CCSM simulates diffusion by letting the core radius $\sigma_i(t)$ grow in time according to Eq. (4). The core radius of a vortex with strength Γ_i grows from a minimum initial value $l_m = \alpha l$ up to a maximum value $l_M = l$, when it splits into four new vortices with strength $\Gamma_i/4$. The numerical parameters α and l, where $0 \le \alpha \le 1$, determine the splitting frequency and the resulting spatial refinement. These new vortices are positioned 90° from each other, at a radial distance $r = 2\sigma(1-\alpha)^{1/2}$ from the center of the original vortex, where r is calculated to conserve the second moment of vorticity. This arrangement and the value of α guarantee the core overlapping that is required for convergence and good spatial accuracy of the CCSM.

2.3. Vortex Merging: Rossi Merging Method (RMM)

The splitting mechanism associated to the CCSM produces an exponential growth of the number of vortices N. In order to control this process and avoid extremely large problem sizes (value of N), Rossi (1997) proposed a merging method that conserves zeroth, first and second moments of vorticity.

Equation (2) shows that the vorticity ω_i of each Lamb vortex is well defined by the three parameters (Γ_i, $\mathbf{x}_i$, σ_i), *i.e*., its strength, position and core radius. The vortex strength Γ_i is calculated during the vortex generation step of the VM, such that the boundary conditions are satisfied on a solid wall and the total vorticity is conserved to obey Kelvin's theorem. As the flow evolves in time the position $\mathbf{x}_i$ is evaluated from the convective displacement of the vortex. The initial core radius σ_i is determined as a function of the model used to estimate the vorticity flux at the wall during the vortex generation step of the VM, which depends on the flow Reynolds number. The value of σ_i is related to the resolution one wishes to obtain for the simulation, since small values tend to improve the overall resolution and guarantee convergence of the VM, as long as the required overlapping is maintained, but requires larger N. Smaller values of N may be obtained when we use large values of σ_i, obtained either from the vortex generation step or from the merging of several vortices, but this choice may cause consistency and convergence errors. Therefore, we must keep σ_i as small as possible and in the range $l_m \le \sigma_i \le l_M$, with $1 \le i \le N$, $l_m = \alpha l$, $l_M = l$ and $0 \le \alpha \le 1$, where l_m and l_M define the maximum and minimum resolutions, respectively.

We may summarize the Rossi Merging Method (RMM) as follows. If n vortices are merged, the error $e(\mathbf{x})$ introduced in the local vorticity field at $\mathbf{x}$ is given by

$$e(\mathbf{x}) = \frac{\Gamma_0}{4\pi\sigma_0^2}\left[\exp\left(-\frac{|\mathbf{x}|^2}{4\sigma_0^2}\right) - \sum_{i=1}^{n}\frac{\Gamma_i}{\Gamma_0}\frac{\sigma_0^2}{\sigma_i^2}\exp\left(-\frac{|\mathbf{x}-\mathbf{x}_i|^2}{4\sigma_i^2}\right)\right]. \tag{7}$$

The objective of the RMM is to obtain one vortex with parameters (Γ_0, $\mathbf{x}_0$, σ_0) after merging a set of n vortices such that the error is small and bounded. Initially, the choice of the postmerger element such that $e(\mathbf{x})$ is bounded requires conservation of the zeroth, first and second moments of vorticity. These constraints define the numerical merging process and determine the parameters (Γ_0, $\mathbf{x}_0$, σ_0) as follows

$$\Gamma_0 = \sum_{i=1}^{n}\Gamma_i\,, \qquad \Gamma_0\mathbf{x}_0 = \sum_{i=1}^{n}\Gamma_i\mathbf{x}_i \qquad \text{and} \qquad 4\Gamma_0\sigma_0^2 = \sum_{i=1}^{n}\Gamma_i\left(4\sigma_i^2 + |\mathbf{x}_i - \mathbf{x}_0|^2\right). \tag{8a, b, c}$$

Rossi's algorithm consists in identifying the set of n vortices that can be merged according to Eqs. (8), for $e(\mathbf{x})$, given by Eq. (7), under control and subject to the condition $l_m \le \sigma_i \le l_M$. Keeping $e(\mathbf{x})$ below a specified tolerance without altering l_m or l_M ensures that the computational vorticity field experiences only "controllably small instantaneous disturbances", as stated by Rossi. The condition $l_m \le \sigma_i \le l_M$ is necessary to maintain accuracy. Because the algorithm requires that only overlapping vortices are merged, the error related to the numerical merging of a set of n vortices is restricted to the spatial region where it occurs and, therefore, globally merging errors will not accumulate in the same spatial region. In other words, one expects that the merging process implies that the greatest pointwise error in the total vorticity field is likely to be of the same order of the greatest pointwise error in an individual merging event. Using the nondimensional variable $\hat{\mathbf{x}} = \mathbf{x}/2\sigma_0$, the merging error $e(\mathbf{x})$ in one single event given by Eq. (7) becomes

$$e(\hat{\mathbf{x}}) = \frac{\Gamma_0}{4\pi\sigma_0^2}\sum_{i=1}^{n}\frac{\Gamma_i}{\Gamma_0}\left\{\exp\left(-|\hat{\mathbf{x}}|^2\right) - \frac{\sigma_0^2}{\sigma_i^2}\exp\left[-|\hat{\mathbf{x}}-\hat{\mathbf{x}}_i|^2\frac{\sigma_0^2}{\sigma_i^2}\right]\right\}. \tag{9}$$

For all Γ_i's with the same sign and using Eq. (8a), Eq. (9) may be interpreted as an arithmetic mean of the quantity

$$z_i \equiv \exp\left(-|\hat{\mathbf{x}}|^2\right) - \frac{\sigma_0^2}{\sigma_i^2}\exp\left[-|\hat{\mathbf{x}}-\hat{\mathbf{x}}_i|^2\frac{\sigma_0^2}{\sigma_i^2}\right]. \tag{10}$$

Using Eq. (10), we may replace Eq. (9) for the merging error of a set of n vortices for

$$e(\hat{\mathbf{x}}) \le \frac{|\Gamma_0|}{4\pi\sigma_0^2}\max_{1\le i\le n}|z_i|\,. \tag{11}$$

The inequality (11) states that the error $e(\mathbf{x})$ can be bounded in terms of the largest value of Eq. (10). Thus, bounding the total error of merging n vortices is equivalent to bounding the difference between a pair of Gaussians. The error is zero when both functions overlap perfectly, that is, $\hat{\mathbf{x}} \to 0$ and $\sigma_0/\sigma_i \to 1$. If the parameters σ_0/σ_i and $\hat{\mathbf{x}}$ are restricted to a compact set, the continuity of Eq. (10) ensures the existence of a maximum value, M, for z_i, with $1 \le i \le n$. If that is the case, all Γ_i's have the same sign and

$$\frac{|\mathbf{x}_i - \mathbf{x}_0|^2}{4\sigma_0^2} \le R \qquad \text{and} \qquad b_1 \le \left(\frac{\sigma_0}{\sigma_i}\right)^2 \le b_2 \tag{12a, b}$$

are true for each computational element, then these four conditions translates into a theorem (Rossi, 1997) which states that the uniform-induced field error $\|e(\hat{\mathbf{x}})\|_\infty$ is less than $(|\Gamma_0|M)/(4\pi\sigma_0^2)$. Rossi shows that the extreme values of z_i are obtained by setting $(|\mathbf{x}_i^2|, \sigma_0^2/\sigma_i^2) = (R, b_1)$ or $(|\mathbf{x}_i^2|, \sigma_0^2/\sigma_i^2) = (R, b_2)$. Once M is determined as a function of R, b_1 e b_2, the conditions for the merging to occur with $\|e(\hat{\mathbf{x}})\|_\infty$ less than a specified tolerance ε can be written as

$$\frac{|\Gamma_0|M}{4\pi\sigma_0^2} \le \varepsilon\,. \tag{13}$$

Equations (12) and (13) comprise the numerical conditions required to merge n vortices into one single vortex with parameters (Γ_0, $\mathbf{x}_0$, σ_0) subject to a merging error ε. However, the application of these conditions is extremely expensive from a computational point of view, since it is necessary to calculate (Γ_0, $\mathbf{x}_0$, σ_0) using Eqs. (8) every time a new candidate is added to the merging set. For this reason, Rossi proposes to use the following more restrictive conditions based on Eqs. (12a) and (13)

$$\left|\mathbf{x}_i - \mathbf{x}_j\right|^2 \le R l_m^2 \qquad \text{and} \qquad \left|\Gamma_0\right| \le \varepsilon \frac{4\pi l_m^2}{M}, \tag{14a, b}$$

for all j and one particular i of the merging candidate set. Equation (14a) replaces Eq. (12a) and Eq. (14b) replaces Eq. (13), since we know that $\sigma_0 \ge l_m$ and $|\,x_j - x_0\,|^2 \le D$. The latter inequality comes from the fact that $|\,x_j - x_i\,|^2 \le D/4$. Equations (14) and (12b) form the conditions to merge the set of candidates with the same sign of Γ_i.

Although very effective, Rossi's merging algorithm still requires N^2 evaluations of the distance d_{ij} between particles. The computational cost of this procedure is very high, and it brings back the N^2 bottleneck on the overall performance of the VM algorithm. For this reason, we propose in this paper to use the AFMM hierarchy of boxes, shown in Fig. (2), to merge vortices in such a way that Rossi's merging algorithm is applied locally to the vortices in each box instead of being applied directly to the entire cloud of vortices.

3. COMPARATIVE RESULTS BETWEEN THE DIRECT MERGING ALGORITHM AND BOX MERGING ALGORITHM

We refer to the original RMM algorithm applied directly to the entire cloud of vortices as "Direct Merging" (DM), whereas the proposed algorithm that uses the AFMM boxes as "Box Merging" (BM). In what follows we present the results obtained for several types of vortex wakes, having different degrees of non-uniformity in space. We compare the CPU time and the efficiency of the merging algorithm obtained with the DM and BM algorithms.

For the test case shown in Fig. (3), which corresponds to a statistically-uniform distribution of vortices in a square box, Fig. (4) allows us to infer that the behavior of the CPU time as a function of N changes from order N^2 for the DM algorithm to order N when the BM algorithm is employed. Additionally, the efficiency of the merging process, defined as the number of merging events that occurs in the cloud with N vortices, is the same regardless the algorithm used.

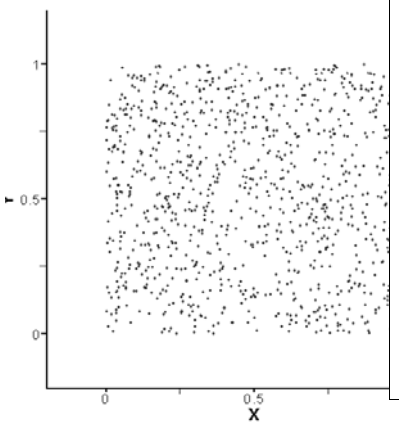

Figure 3. Uniform distribution

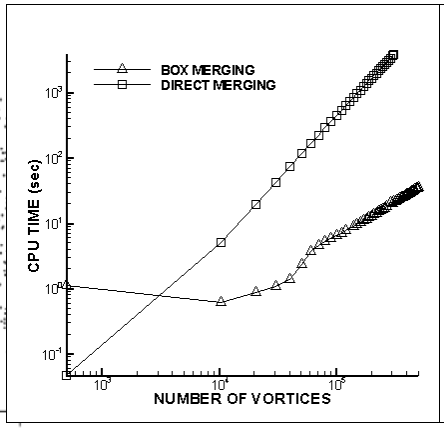

Figure 4. CPU time for the uniform distribution

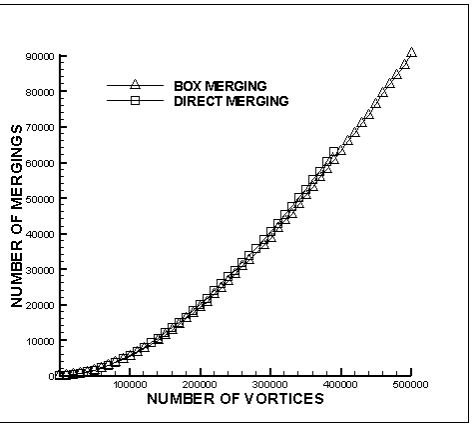

Figure 5. Number of vortices for the uniform distribution

In Figures (6), (9) and (12) we present results for three test cases characterized by statistically non-uniform vortex distributions. The results obtained for the CPU time, illustrated by the graphs of Figs. (7), (10) and (13), respectively, demonstrate that the BM algorithm is one order of magnitude faster than the DM algorithm, although the BM algorithm still presents a order N^2-type behavior. Although we expected an N-type behavior for these cases also, the results indicate that there is a dependence of the AFMM algorithm on the relation between the core radius and the box size, for a fixed maximum number of vortices per box, that affects the performance of the AFMM algorithm during the box refinement process (Santiago *et al*., 2006). For clouds that are very dense locally, the box size may become so small that it gets smaller than the core radius. In these cases the AFMM algorithm requires an increase of the maximum number of vortices per box, so that the box size becomes greater than core radius. In such situations, the total number of boxes in the computational box decreases and the number of vortices per box increases, producing a dominance of the N^2-type behavior in each box. As a consequence, the CPU time does not follow the N-type behavior. Regardless of this observation, the reduction on the CPU time in one order of magnitude is outstanding, and the BM algorithm is shown to be much faster than the DM algorithm for all cases studied. Figures (8), (11) and (14) again demonstrate that the merging efficiency is the same for both algorithms. For all these cases studied we see that the DM algorithm is more efficient only if N is small, which only occurs at early stages of the simulation for fluid dynamics applications.

The last test case chosen to study the performance of the BM algorithm in comparison with the DM algorithm corresponds to a VM simulation (Santiago *et al*., 2006) of the flow around a NACA 0012 airfoil set at an angle of attack equal to 10°. Figure (15) shows the position of the wake vortices at $t = 10$, illustrating the wake pattern and the non-uniform vortex distribution in space that we obtain in a simulation of this type. As shown in Fig. (16), the results of the CPU time as a function of N obtained for the merging step of the VM code using the BM algorithm are one order of magnitude lower than the results for the DM algorithm. Each value of N corresponds to a specific time t and wake pattern of the simulation. This result corroborates the main conclusion drawn for the previous test cases that the computational reduction observed in the CPU time of the merging step is vital to turn feasible any long-time simulation of the flow around a body. We may also appreciate from Fig. (17) that, again, both algorithms merge the same number of vortices for a given wake (or a given time t).

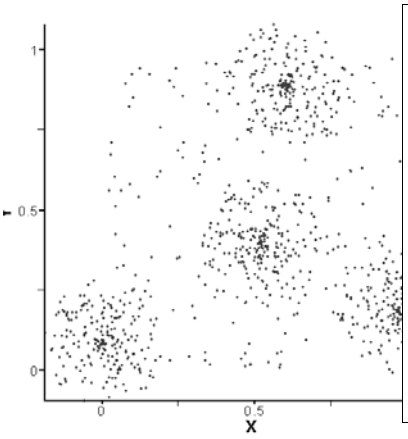

Figure 6. Non-uniform distribution

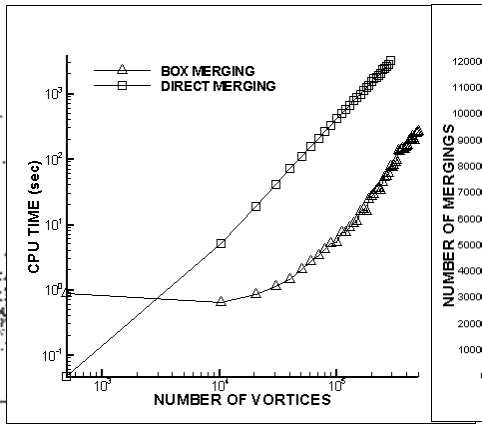

Figure 7. CPU time as a function of the number of vortices

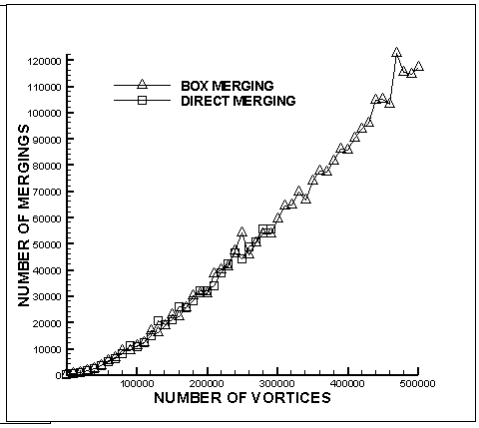

Figure 8. Number of merging events

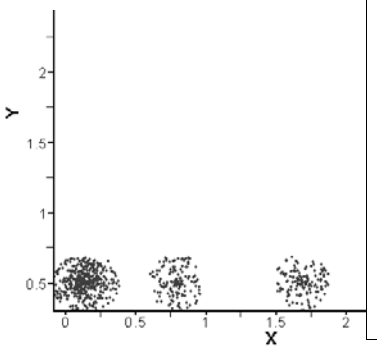

Figure 9. Non-uniform distribution on a straight line

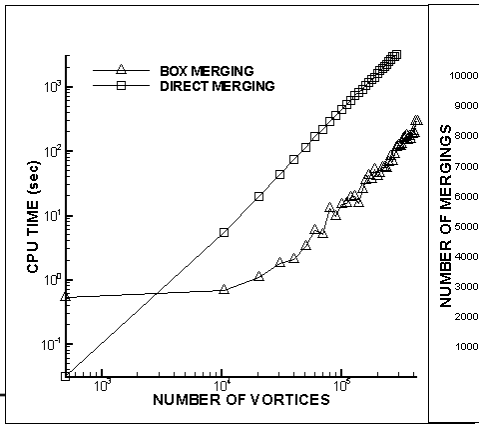

Figure 10. CPU time as a function of the number of vortices

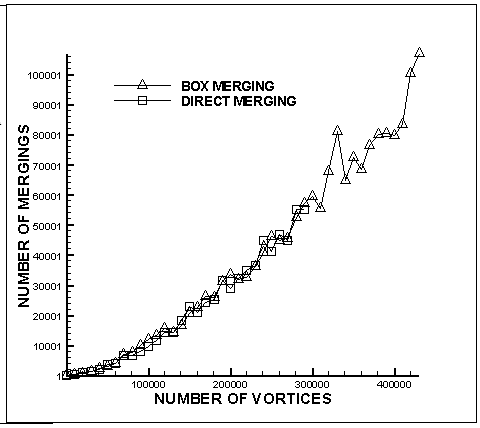

Figure 11. Number of merging events

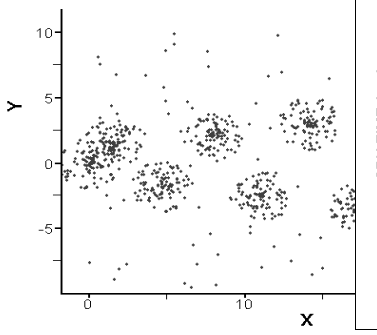

Figure 12. Non-uniform sinusoidal distribution

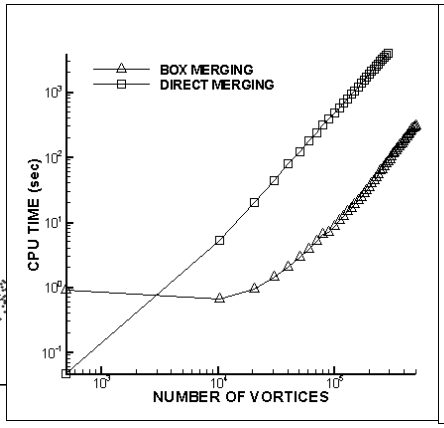

Figure 13. CPU time as a function of the number of vortices

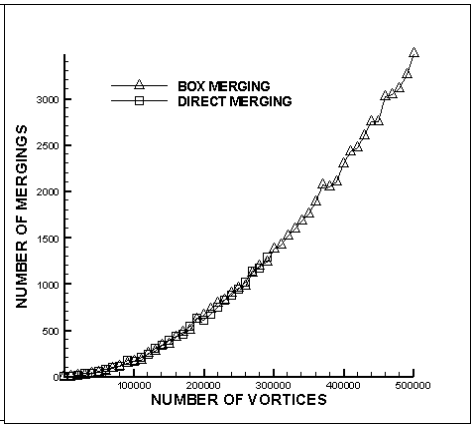

Figure 14. Number of merging events

Figure 15. NACA 0012 airfoil at 10º angle of attack

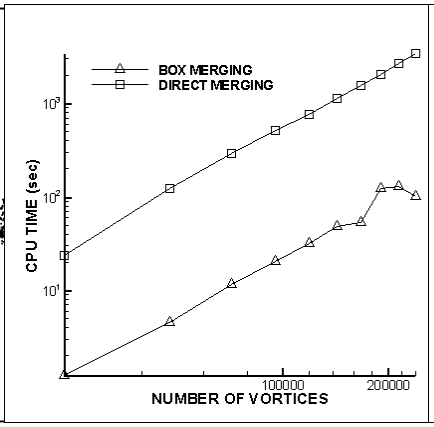

Figure 16. CPU time as a function of the number of vortices

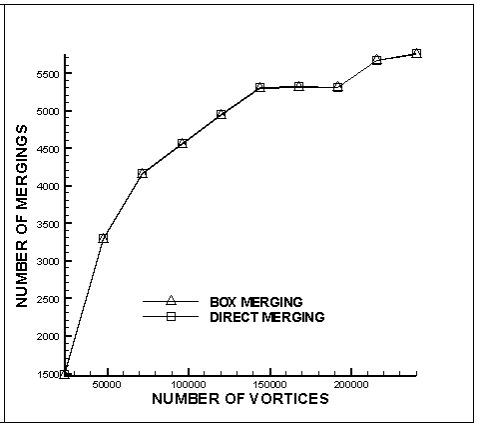

Figure 17. Number of merging events

4. CONCLUSIONS

The calculations presented in this paper demonstrate that there is an exceptional reduction on the CPU time of long-time VM simulations around bodies if the BM algorithm is used to replace the DM algorithm. Three test cases with non-uniform vortex distribution in space show a one order of magnitude reduction on the CPU time for large N, whereas the case with uniform distribution indicate an even higher reduction, since the slope of the CPU time versus N curve changes from N^2 to N. In all cases the BM algorithm allowed the merging of 500,000 vortices to be carried out in a reasonable amount of time. When the BM algorithm is used in a typical VM simulation around an airfoil the reduction observed on the CPU time is also one order of magnitude lower than when the DM algorithm is used. In conclusion, our results show that the idea of employing the AFMM hierarchy of boxes to merge vortices in the wake during a VM simulation is very effective, since it produces an enormous reduction on the total CPU time of the complete simulation.

5. ACKNOWLEDGEMENTS

The authors would like to thank the Prof. L. Greengard for making the AFMM original Fortran code available to us and Prof. L. Rossi for the fruitful discussions about the vortex method and its efficient application to flow simulations. The authors also acknowledge the financial support to this research project provided by the Brazilian Army and CNPq.

6. REFERENCES

Barba, L. A., Leonard, A. and Allen, C. B., 2005, "Advances in Viscous Vortex Methods – Meshless Spatial Adaptation Based on Radial Basis Function Interpolation", International Journal for Numerical Methods in Fluids, Vol. 47, pp. 387-421.

Carrier, J., Greengard, L. and Rokhlin, V., 1988, "A Fast Adaptive Multipole Algorithm for Particle Simulation", SIAM Journal of Scientific Statistics and Computation, Vol. 9, n. 4, pp. 669-686.

Chorin, A. J., 1973. "Numerical Study of Slightly Viscous Flow". Jounal of Fluid Mechanics , Vol. 57, pp. 785-796.

Cottet, G.-H. and Koumoutsakos, P., 1999. "Vortex Methods. Theory and Practice". Cambridge University Press.

Greengard, C., 1985, "The Core Spreading Vortex Method Approximates the Wrong Equation", Journal of Computational Physics, Vol. 61, pp. 345-348.

Katz, J. & Plotkin, A., 2001. "Low-Speed Aerodynamics", McGraw-Hill, New York, 2nd Edition.

Leonard, A., 1980, "Vortex Methods for Flow Simulation", *Journal of Computational Physics*, Vol. 37, pp. 289-335.

Rossi, L. F., 1996, "Resurrecting core spreading vortex methods: A New Scheme That Is Both Deterministic and Convergent", SIAM J. Sci. Comp., Vol. 17, nº 2, pp. 370-397.

Rossi, L. F., 1997, "Merging Computational Elements in Vortex Simulations", SIAM J. Sci. Comput., vol. 18 , nº 4, pp. 1014-1027.

Santiago, V. S., Silva, D. F. C and Bodstein, G. C. R., 2006, "Análise de Desempenho do Método de Expansão em Multipolos Adaptativo Aplicado a Simulações Numéricas Via Método de Vórtices", Anais do XI Congresso Brasileiro de Engenharia e Ciências Térmicas - ENCIT, Curitiba, PR, 05 a 08 de dezembro, 2006, pp. 1-10.

Silva, D. F. C., Bodstein, G. C. R., "Numerical Simulation of Usteady Two-Dimensional Incompressible Flows Around Airfoils Using the Vortex Method", Proceedings of the 3rd International Conference on Vortex Flows and Vortex Models (ICVFM2005), Yokohama, Japan, November 21 - 23, 2005, pp. 173-178.

7. RESPONSIBILITY NOTICE

The authors are the only responsible for the printed material included in this paper.