MECSOL 2019 - Proceedings of the 7th International Symposium on Solid Mechanics
V. Tita, A. Ferreira, M.L. Ribeiro, A. Vieira and A.T. Beck (Editors), ABCM, Sao Carlos, SP, Brazil, April 15th to 17th, 2019

An analysis of Material Point Method in one-dimensional space

Leonardo Tolêdo Ferreira[1], Luciana Correia Laurindo Martins Vieira[1] and Tiago Peixoto da Silva Lobo[1]

[1] Universidade Federal de Alagoas, Av. Lourival Melo Mota, Tabuleiro do Martins, Maceió, AL, Brazil

Abstract. When studying dynamic problems that cover large displacements, one might find difficulties in handling mesh distortion problems and calculating the convective term in momentum conservation equation with traditional numerical methods like the Finite Element Method (FEM). Moreover, these methods do not handle fractures and contact surfaces naturally, needing sophisticated and, oftentimes, complicated algorithms to do so. Furthermore, usage of FEM in dynamic problems becomes limited and, because of this, attention is turned to the Material Point Method (MPM), for it enables the handling of continuum mechanics problems without mesh distortion. This is achieved by filtering the best features of Lagrangian and Eulerian formulations, for the mesh in MPM is generally fixed, avoiding the convective term in momentum conservation equation. Stimulated by these advantages, a detailed analysis of MPM is done by studying the influence of the number of particles (material points), number of elements and time step in the final quality of simulations.

Keywords: solid mechanics, dynamics, material point method, numerical methods, computer simulation

INTRODUCTION

The Material Point Method (MPM) was first proposed by Sulsky, Chen and Schreyer (1994) as an extension of particle-in-cell method. In MPM, particles are interpreted as material points, which are the basis of calculations throughout the simulation. That means, instead of using a deformable mesh that stores its information inside its nodes, all relevant properties of the continuum are stored inside particles (eg. position, mass, momentum, stress, strain). By doing this, mesh distortion ceases to be a problem, because node position will be fixed. To illustrate this, Figure 1 shows the discretization process for this method, taking as basis an initial domain Ω_0 in two dimensions.

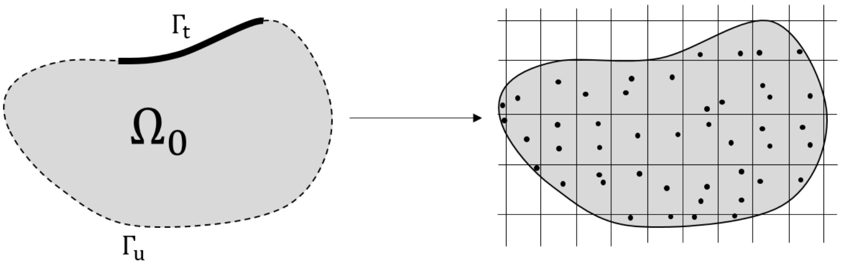


Figure 1: Discretization process in MPM

Contrary to other numerical methods, such as Finite Element Method, MPM does its interpolation process twice: at the beginning of each time step, information is mapped from particles to nodes and, by the end of it, information is mapped back from nodes to particles. Two algorithms are commonly used with MPM to compute stresses and they differ by the moment in time integration in which they are calculated (Bardenhagen, 2002): the first one computes stresses at the beginning of the time step and is called Update Stresses First (USF); the second one computes stresses at the end of the time step and is called Update Stresses Last (USL).

In recent years, the method has been applied to solve a vast number of engineering problems, like the modelling of anchors (Coetzee, Vermeer and Basson, 2005), offshore geotechnical applications (Brinkgreve et al., 2017), large-strain problems (Ardersen, 2009), large scale run-out processes in landslides (Llano-Serna, Farias and Pedroso, 2016), impact forces on pipeline by submarine landslide (Dong, Wang and Randolph, 2017) and many more.

Using both USF and USL algorithms, this article intends to analyze the results obtained as the number of particles, elements and the time step are changed. An analysis of the influence of these parameters on the computation time is also presented for a simple dynamic problem.

FORMULATION

The setup of MPM begins with the strong form of momentum conservation equation, which for simplicity, in this article, is solved in one-dimensional space, as taken from Zienkiewicz, Taylor and Zhu (2013):

$$\frac{\partial \sigma_x}{\partial x} + b_x = \rho \cdot \frac{\partial^2 u}{\partial t^2}, \tag{1}$$

where σ_x is the Cauchy stress vector (a scalar in one-dimension), b_x is the body force, ρ is the mass density and u is the displacement.

By manipulation of Eq. (1), establishment of a weight function (ω) and application of boundary conditions - prescribed displacements (Γ_u) and prescribed tractions (Γ_t), see Fig. 1 - , the weak form of the momentum conservation equation is obtained, as follows:

$$\int_\Omega \omega \cdot \rho \cdot \frac{\partial^2 u}{\partial t^2} dx + \int_\Omega \frac{\partial \omega}{\partial x} \cdot \sigma_x dx = \int_\Omega \omega \cdot b_x dx + \omega \left(t_x|_{\Gamma_u} + \overline{t_x}|_{\Gamma_t} \right). \tag{2}$$

For simplicity, without loss of generality, it is assumed that no external loads or body forces are present. Thus, the right-hand side of Eq. (2) will be zero. Then, Langrangian shape functions (represented by N^e) are applied to approximate ω and u in relation to the finite elements of the fixed background mesh.

The material domain is discretized into a set of material points, which move through an underlying mesh. This way, the mass density function can be writen as a sum of Dirac delta functions (it is assumed that the particles concentrate all mass of their elements):

$$\rho(x,t) = \sum_{p=1}^{n_p} m_p \cdot \delta(x - x_p), \tag{3}$$

where n_p is the number of particles per element, x_p is the particle position and m_p is the particle mass.

To make it possible to apply Eq. (3) to the stress term in Eq. (2), the stress is redefined in terms of mass density:

$$\sigma_s = \frac{\sigma_x}{\rho}, \tag{4}$$

where σ_s is the stress in terms of density ρ.

Applying Eq. (3) and Eq. (4) into Eq. (2), and using the aforementioned Lagrangian shape functions to approximate the variables, the general equation of the problem is obtained, as showed in Eq. (5):

$$\mathbf{M} \cdot \mathbf{a} = \mathbf{f_{int}}. \tag{5}$$

where $\mathbf{M}$ is the mass matrix, $\mathbf{a}$ is the vector of nodal accelerations and $\mathbf{f}_{int}$ is the vector of internal forces.

Nodal results for the terms in Eq. (5) are explicitly shown in the equations below:

$$\mathbf{M^e} = \sum_{p=1}^{n_p} N^e(x_p) \cdot m_p \cdot N^e(x_p), \tag{6}$$

$$\mathbf{a^e} = \ddot{u}^e(t), \tag{7}$$

$$\mathbf{f_{int}^e} = -\sum_{p=1}^{n_p} \frac{dN^e}{dx}|_{x=x_p} \cdot m_p \cdot \sigma_s(x_p,t). \tag{8}$$

In this work, the lumped mass matrix was adopted, for it enables faster computation times without great loss of precision. Since time integration will be done with explicit Euler method, it is of importance to define a critical time step (dt_{max} or critical dt) which defines the maximum time step value for which the simulation will still converge to the desired values. Equation (9) shows how to find that value.

$$dt_{max} = \frac{\Delta x}{c}, \tag{9}$$

where Δx is the size of the element used in the discretization process, $c = \sqrt{\frac{E}{\rho}}$ is the wave speed of vibration, E is the elastic modulus of the body and ρ is the mass density (as stated previously in this section).

That way, to ensure convergence, the time step used in this article will always be a fraction (limited to 50%) of the critical time step. In general terms, any time step adopted in this work can be expressed in the following manner:

$$dt = k \cdot dt_{max}, \tag{10}$$

where k is a real number between 0 and 1.

At each time step, as cited in the introduction, mapping is done between particles and nodes and vice versa (more details can be found at Kafaji (2013)). The flowcharts in Figures 2a and 2b summarize this process for the USF and USL algorithms.

```
While end time is not reached, perform:

1. Mapping from particles to nodes:
    a. Calculation of nodal mass.
    b. Calculation of nodal momentum.
    c. Calculation of nodal velocity.
2. Update of particle stress.
3. Calculation of nodal force and acceleration.
4. Update of nodal momentum
5. Mapping from nodes to particles:
    a. Update of particle velocity.
    b. Update of particle position.
```

(a) USF algorithm

```
While end time is not reached, perform:

1. Mapping from particles to nodes:
    a. Calculation of nodal mass.
    b. Calculation of nodal momentum.
    c. Calculation of nodal velocity.
2. Calculation of nodal force and acceleration.
3. Mapping from nodes to particles:
    a. Update of particle velocity.
    b. Update of particle position.
4. Update of nodal velocity
5. Update of particle stress.
```

(b) USL algorithm

Figure 2: Flowchart of USF and USF algorithms

BAR PROBLEM

As presented by Bardenhagen (2002), the problem involving the axial vibration of a continuum bar (Fig. 3a) was simulated in this article. This problem consists of a bar with one of its ends fixed and the other one free to move. No external loads or body forces are applied. Throughout the length (L) of the bar, a prescribed sinusoidal velocity is applied. Equation (11) shows the boundary and initial conditions.

$$u(0,t) = 0, u(x,0) = 0, v(x,0) = v_0 \cdot \sin(\beta_n x), \tag{11}$$

where v is the velocity, v_0 is the amplitude of the initial velocity and $\beta_n = \frac{2n-1}{2}\frac{\pi}{L}$, n being the mode of vibration.

Bardenhagen (2002) showed that the solutions for displacement and velocity are the following:

$$u(x,t) = \frac{v_0}{\omega_n} \cdot \sin(\omega_n t) \cdot \sin(\beta_n x), \tag{12}$$

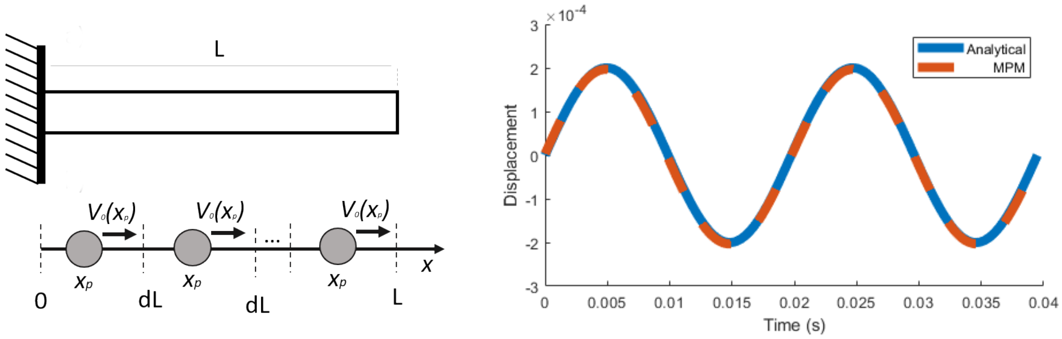
$$v(x,t) = v_0 \cdot \cos(\omega_n t) \cdot \sin(\beta_n x), \tag{13}$$

where $\omega_n = \beta_n \cdot c$ is the frequency of vibration of the body.

By assuming the first mode of vibration ($n = 1$), the center-of-mass displacement is as shown in Eq. (14).

$$u_{cm}(t) = \frac{v_0}{\omega_n \beta_n L} \cdot \sin(\omega_n t). \tag{14}$$

To verify the algorithm, this problem was executed with time step of 10% of critical dt, 20 elements and 1 particle per element, yielding the results shown in Fig. 3b.



(a) Bar problem description

(b) Validation of the displacement results

Figure 3: Scheme of the bar problem

USF ALGORITHM RESULTS

To avaliate how MPM behaves with multiple parameters, color graphs were plotted. In these graphs, the color represents the maximum absolute error when comparing the numerical center-of-mass displacement results with the analytical ones, provided by Eq. 14. In this representation, a darker color is synonymous to a smaller error, while a lighter color is synonymous to a larger error. The abscissa shows the range of particles for which the problem was simulated, while the ordinate shows the range of time steps (with percentages regarding the critical time step). A small plot inset shows the results for 15% of the critical time step in data points format. Table 1 shows the parameters used in this simulation (the same as ASTM A36 steel), Tab. 2 shows the numerical domain adopted and the results are showed in Fig. 4.

Elastic modulus (E)	$200 \cdot 10^9$ Pa
Mass density (ρ)	$7,800$ kg/m^3
Bar length (L)	25 m
Velocity amplitude (v_0)	0.1 m/s
Mode of vibration (n)	1

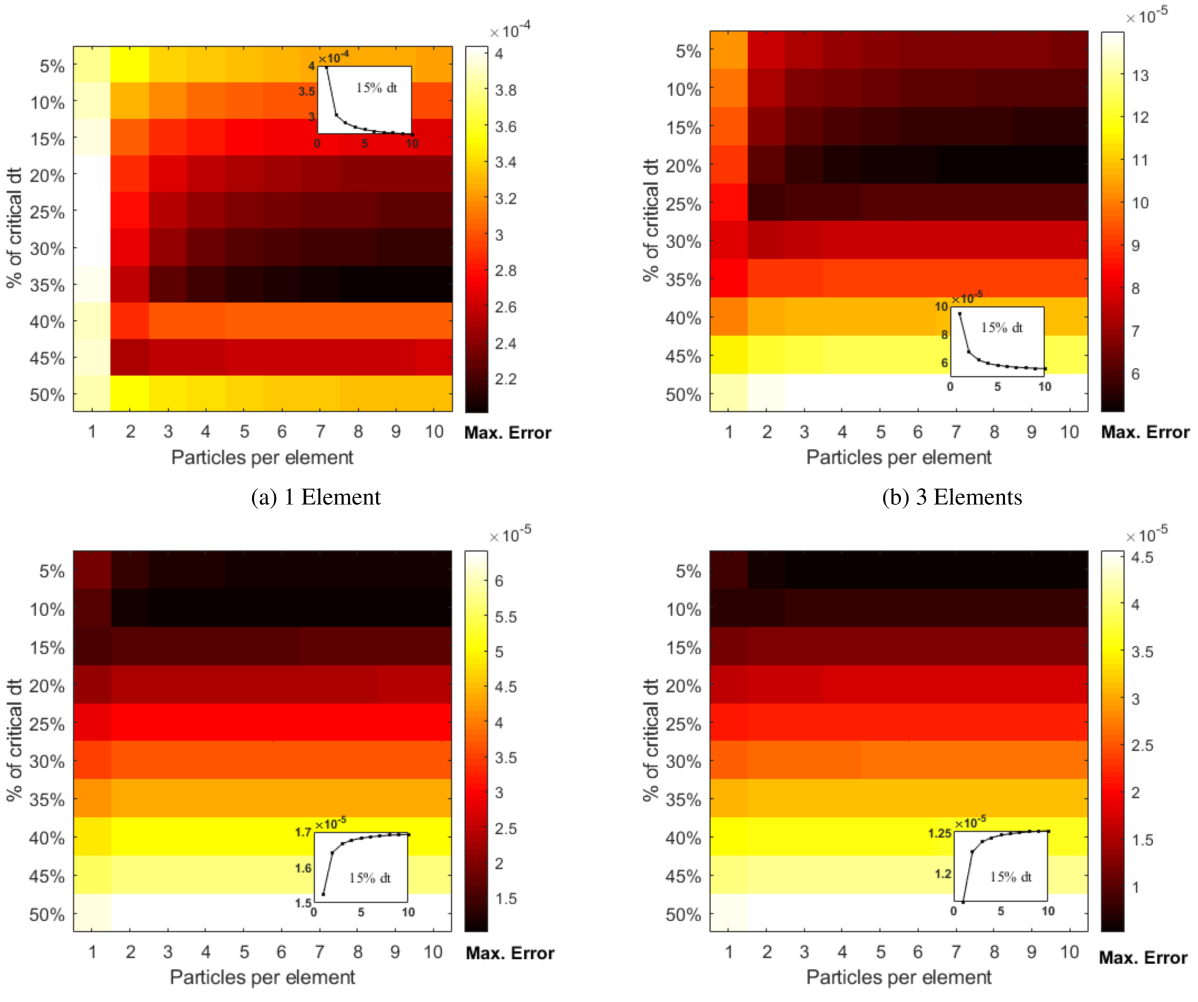
Table 1: Parameters used in the simulation

Simulation time (t_{sim})	0.0395 s (2 periods)
Critical time step (dt_{max})	$2.47 \cdot 10^{-4}$ s
Time step (% of dt_{max})	5%, 10%, 15%, 20%, 25%, 30%, 35%, 40%, 45%, 50%
Number of elements	1, 3, 7, 10, 20, 50
Nodes per element	2
Particles per element	1, 2, 3, 4, 5, 6, 7, 8, 9, 10

Table 2: Numerical domain of simulations

The results in Fig. 4 were displayed for 1 to 10 elements because it was noticed in the simulations of this article that, for a number of elements bigger than 10, the shape of the graphs didn't change significantly. Computation time for these results were also analyzed. Figure 5 shows the results for computation time using a mesh with 3 elements. Results for the other case tests (1, 7 and 10 elements) showed very similar patterns to the one with 3 elements, with computation time varying between 0.002 to 0.2 seconds for 1 element, 0.07 to 7 seconds, for 7 elements and 0.14 to 14 seconds, for 10 elements.

To examine the results in Fig. 5 more closely, Fig. 6a shows these results for a time step of 5% of the critical time step (the first row of Fig. 5) and Fig. 6b shows the results for a number of 10 particles (the last column of Fig. 5).



(a) 1 Element

(b) 3 Elements

(c) 7 Elements

(d) 10 Elements

Figure 4: Maximum center-of-mass displacement error with multiple discretizations with USF

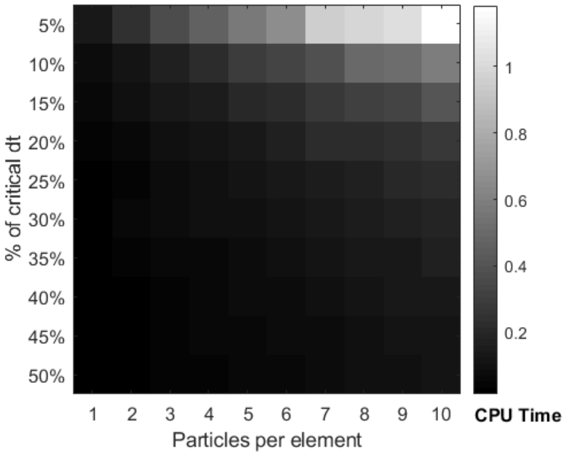
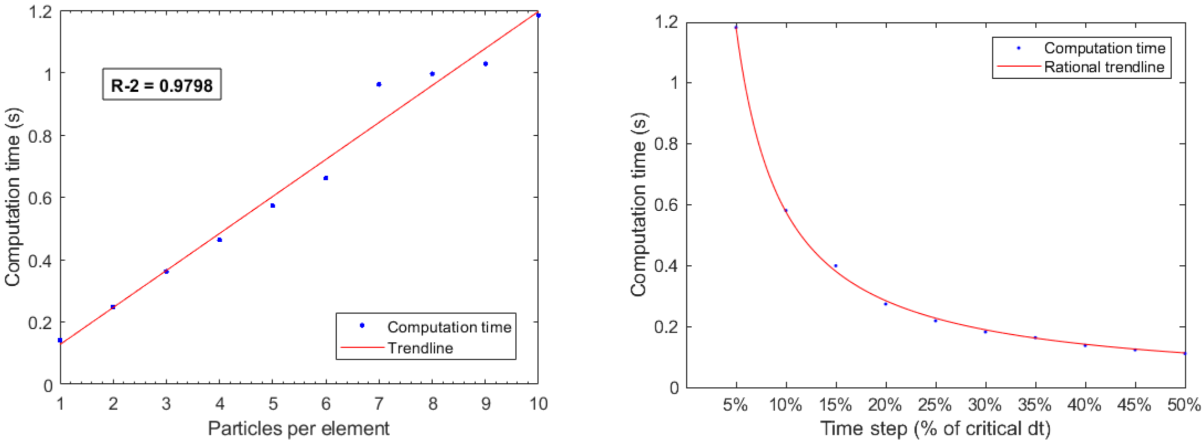


Figure 5: Computation time for a mesh of 3 elements with USF



(a) Computation time for of 5% of critical dt

(b) Computation time for 10 particles

Figure 6: Detailed view of computation time for a mesh of 3 elements with USF

USL ALGORITHM RESULTS

The same procedure adopted for the USF algorithm was done using the USL algorithm. In this simulation, the same parameters of table 1 and the same numerical domain of table 2 were used. The results for the maximum absolute error of displacement in the center-of-mass are displayed in Fig. 7.

Computation time for these results were also analyzed. Figure 8 shows the results for computation time using a mesh with 3 elements. Results for the other case tests (1, 7 and 10 elements) showed very similar patterns to the one with 3 elements, with computation time varying between 0.003 to 0.2 seconds for 1 element, 0.07 to 8 seconds, for 7 elements and 0.13 to 15 seconds, for 10 elements.

To examine the results in Fig. 8 more closely, Fig. 9a shows these results for a time step of 5% of the critical time step (the first row of Fig. 8) and Fig. 9b shows the results for a number of 10 particles (the last column of Fig. 8).

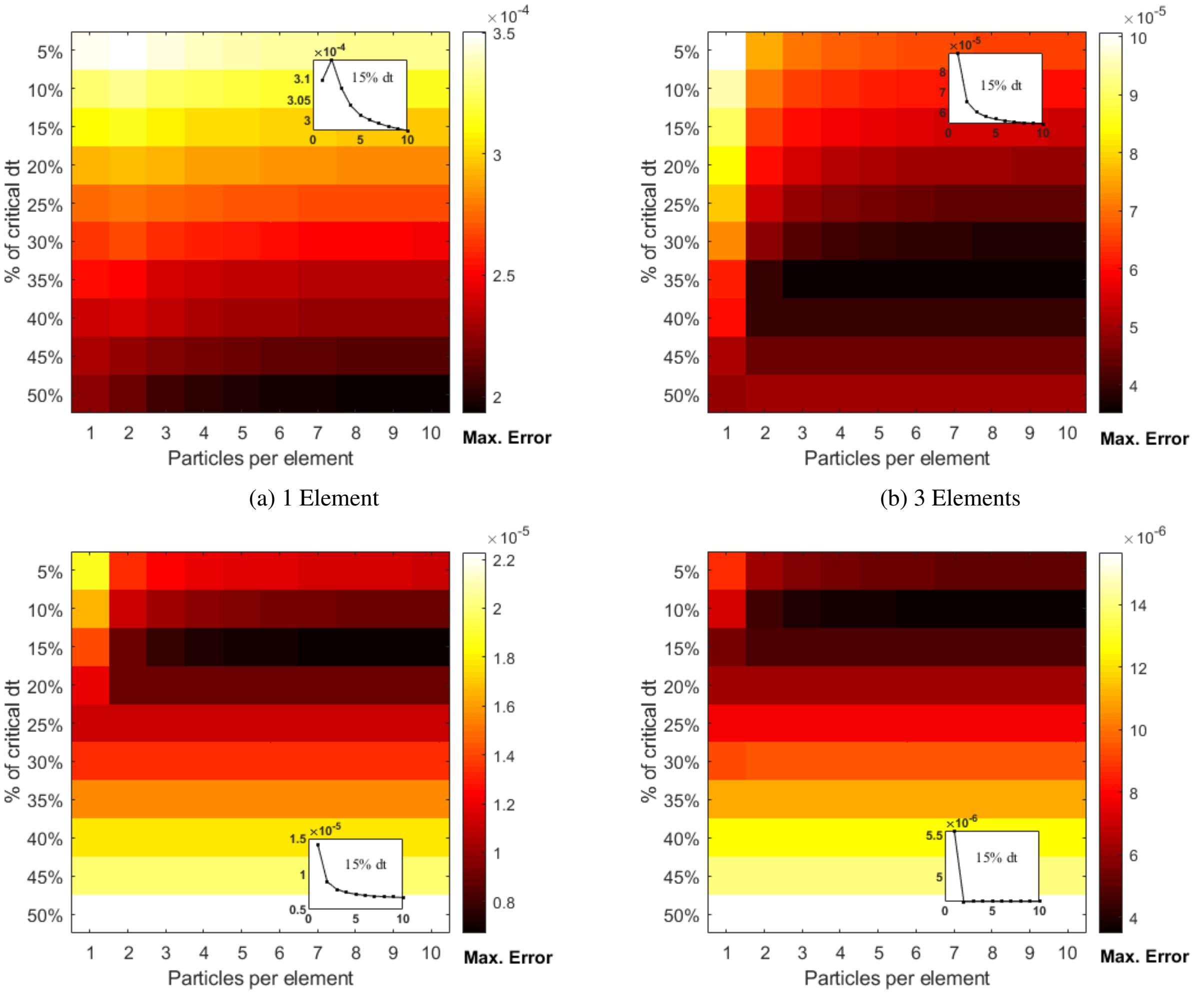
CONCLUSION

From these results, it is clear that, in general, the quality of simulation is improved when the number of elements increases and the time step decreases. However, it was noticed, as seen in Fig. 4a and Fig. 7a, that the result doesn't converge to the expected displacement field when the discretization of the mesh is too disperse. This effect happens because, when the number of elements is too small, the initial velocity field is not properly represented, as seen in Fig. 10, in which is notable that the method's representation of the velocity field does not agree with the initial analytical condition simulated. For this reason, in the test cases with 1 and 3 elements, the initial results for both USF and USL algorithms seem to get worse when the time step decreases too much because, in these cases, the method is converging to a solution that is different to the analytical one. Moreover, the error seems to get overall a bit smaller when the number of particles is increased, but this effect seems to get less significant as the number of elements increases.

The reduction of the size of the time step, as expected, minimizes the error in the displacement. This effect, however, only is noticed when the mesh discretization is not too disperse, for the reasons stated in the last paragraph. When the mesh discretization is insufficient to represent the problem (with 1 and 3 elements, for example), the results don't follow this pattern.

The analysis of the computation time shows that, as expected, the time needed to execute the algorithm increases when the number of particles or the number of elements are increased. It also shows that the computation time increases with the reduction of the time step. The correlation between the computation time and the number of particles is linear, as show in Fig. 6a and Fig. 9a, with an R-2 bigger than 0.97 for both. On the other hand, the correlation between the computation time and the time step is that of a rational function, as showed in Fig. 6b and Fig. 9b.

Overall, the difference between the USF and the USL algorithms is that the latter seems to converge quicker to the analytical solution that the former for the problem simulated in this article.



(a) 1 Element

(b) 3 Elements

(c) 7 Elements

(d) 10 Elements

Figure 7: Maximum center-of-mass displacement error with multiple discretizations with USL

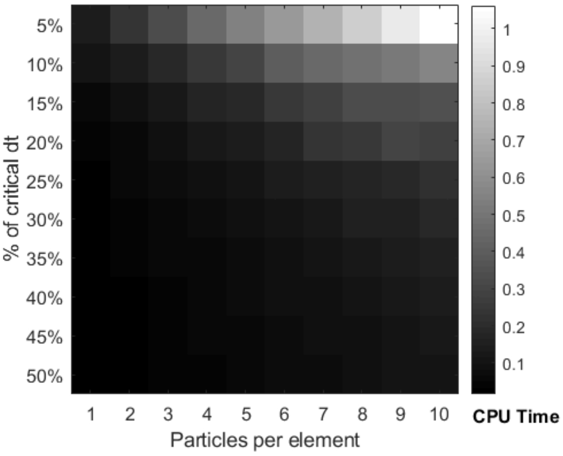
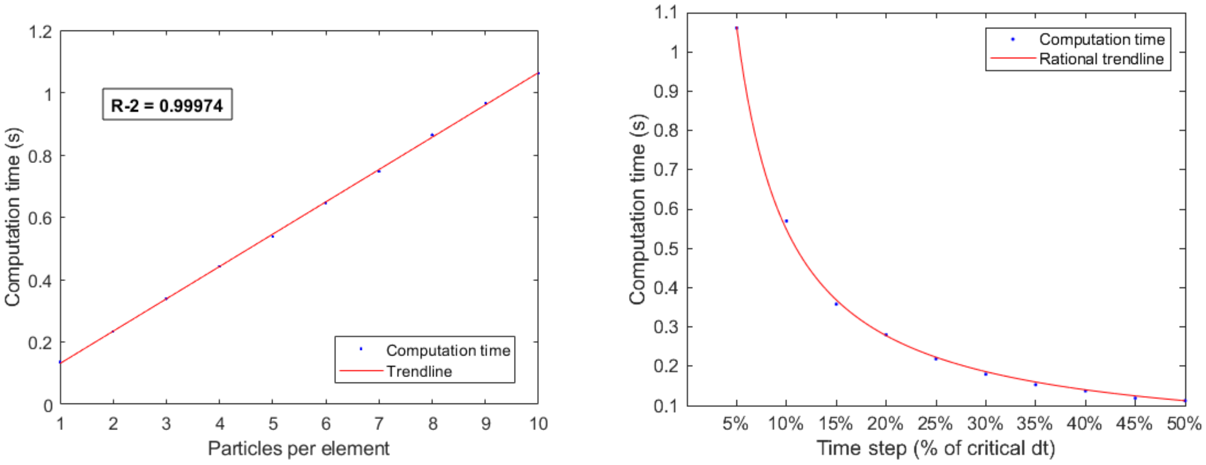


Figure 8: Computation time for a mesh of 3 elements with USL



(a) Computation time for of 5% of critical dt

(b) Computation time for 10 particles

Figure 9: Detailed view of computation time for a mesh of 3 elements with USL

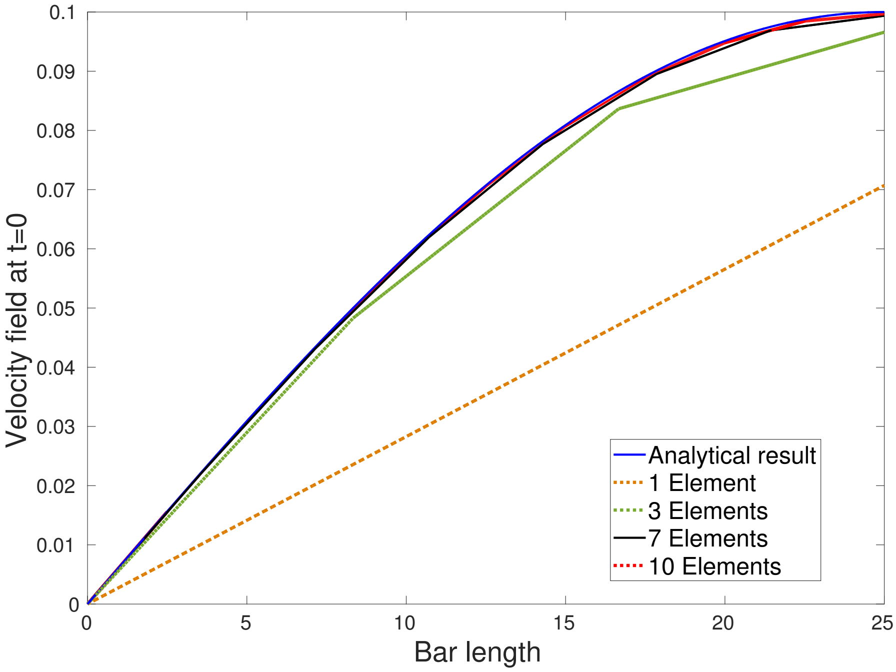


Figure 10: Comparison of velocity field results for 1, 3, 7 and 10 elements

ACKNOWLEDGMENTS

Our thanks to CENPES/PETROBRAS and the Laboratory of Scientific Computing and Visualization (LCCV) for the financial support.

REFERENCES

Andersen, S.M., 2009, ”Material-Point Analysis of Large-Strain Problems”, Aalborg University: Department of Civil Engineering (PhD thesis).

Bardenhagen, S.G., 2002, “Energy conservation error in the material point method for solid mechanics”, Journal of Computational Physics, Vol. 180, No. 1, pp. 383-403.

Brinkgreve, R.B.J., Burg, M., Liim, L.J. and Andreykiv, A., 2017, ”On the practical use of the Material Point Method for offshore geotechnical applications”, 19th International Conference on Soil Mechanics and Geotechnical Engineering, pp. 2269-2272.

Coetzee, C.J., Vermeer, P.A. and Basson, A.H., 2005, ”The modelling of anchors using the material point method”, International Journal for Numerical and Analytical Methods in Geomechanics, Vol. 29, No. 9, pp. 879-895.

Dong, Y., Wang, D. and Randolph, M.F., 2017, ”Investigation of impact forces on pipeline by submarine landslide using material point method”, Ocean Engineering, Vol. 146, pp. 21-28.

Kafaji, I.K.J. al-., 2013, ”Formulation of a dynamic material point method (MPM) for geomechanical problems”, Universität Stuttgart (PhD thesis).

Llano-Serna, M.A., Farias, M.M. and Pedroso, D.M., 2016, ”An assessment of the material point method for modelling large scale run-out processes in landslides”, Journal of the International Consortium on Landslides, Vol. 13, No. 5, pp. 1057–1066.

Sulsky, D., Chen, Z. and Schreyer, H.L., 1994, “A particle method for history-dependent materials”, Computer Methods in Applied Mechanics and Engineering, Vol. 118, Nos. 1-2, pp. 179-196.

Zienkiewicz, O.C., Taylor, R.L. and Zhu, J.Z, 2013, “The Finite Element Method: Its Basis and Fundamentals (Seventh Edition)”, Butterworth-Heinemann.

RESPONSIBILITY NOTICE

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