Entre Ciencia e Ingeniería, vol. 14, no. 28, julio-diciembre de 2020, páginas 82-87. DOI: https://doi.org 10.31908/19098367.2058.
ISSN 1909-8367 (Impreso), ISSN 2539-4169 (En línea)

Cluster analysis for granular mechanics simulations using Machine Learning Algorithms[1]

Análisis de clústeres para simulaciones de mecánica granular mediante algoritmos de aprendizaje automático

D. Rim, E. N. Millán, B. Planes, E. M. Bringa y L. G. Moyano

Recibido: octubre 20 de 2020 – Aceptado: diciembre 28 de 2020.

***Abstract*— Molecular Dynamics (MD) simulations on grain collisions allow to incorporate complex properties of dust interactions. We performed simulations of collisions of porous grains, each with many particles, using the MD software LAMMPS. The simulations consisted of a projectile grain striking a larger immobile target grain, with different impact velocities. The disadvantage of this method is the large computational cost due to a large number of particles being modeled. Machine Learning (ML) has the power to manipulate large data and build predictive models that could reduce MD simulation times. Using ML algorithms (Support Vector Machine and Random Forest), we are able to predict the outcome of MD simulations regarding fragment formation after a number of steps smaller than in usual MD simulations. We achieved a time reduction of at least 46%, for 90% accuracy. These results show that SVM and RF can be powerful yet simple tools to reduce computational cost in collision fragmentation simulations.**

***Keywords*— granular simulations, machine learning, classification analysis, performance analysis.**

***Resumen*— Las simulaciones de dinámica molecular (MD) en colisiones de granos permiten incorporar propiedades complejas de interacciones de polvo. Realizamos simulaciones de colisiones de granos porosos, cada uno con muchas partículas, utilizando el software LAMMPS de MD. Las simulaciones consistieron en un grano de proyectil que golpeó un grano objetivo inmóvil más grande, con diferentes velocidades de impacto. La desventaja de este método es el gran costo computacional debido a que se modela una gran cantidad de partículas. Machine Learning (ML) tiene el poder de manipular grandes datos y construir modelos predictivos que podrían reducir los tiempos de simulación MD. Usando algoritmos ML (Support Vector Machine y Random Forest) podemos predecir el resultado de las simulaciones MD con respecto a la formación de fragmentos, después de varios pasos más pequeños que en las simulaciones MD habituales. Logramos una reducción de tiempo de al menos un 46%, para una precisión del 90%. Estos resultados muestran que SVM y RF pueden ser herramientas poderosas pero simples para reducir el costo computacional en simulaciones de fragmentación de colisiones.**

***Palabras Clave*— simulaciones granulares, aprendizaje automático, análisis de clasificación, análisis de rendimiento.**

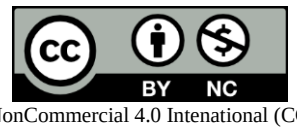
I. INTRODUCTION

THE study of mechanical impacts in physics has been of interest for decades, particularly collisions between small aggregates of dust grains (granular clusters). To model these clusters, one of the commonly used strategies is molecular dynamics (MD) simulations, which solve Newton's equations of motion for an ensemble of particles interacting through a force field. This method allows incorporating complex properties of dust interactions. Typically, the grains themselves represent solid material that remains unchanged internally during the collision process, while the whole ensemble (the cluster) will undergo restructuring, aggregation, or fragmentation. This strategy has been used successfully to describe the collision of granular clusters containing up to several thousand grains [1].

[1] Producto apoyado por la Universidad Nacional de Cuyo y CONICET, Argentina.

D. Rim, Fa, UNCuyo, Mendoza, Argentina, email: rim.dan96@gmail.com
E. N. Millán, UNCuyo and CONICET, Mendoza, Argentina, email: emmanueln@gmail.com
B. Planes, Universidad de Mendoza and CONICET, Mendoza, Argentina, email: belenplanes.88@gmail.com
E. M. Bringa, Universidad de Mendoza and CONICET, Mendoza, Argentina, email: ebringa@yahoo.com
L. G. Moyano, UNCuyo and CONICET, Mendoza, Argentina, email: lgmoyano@fcen.uncu.edu.ar

How to cite: D. Rim, E. N. Millán, B. Planes, E. M. Bringa y L. G. Moyano. Cluster analysis for granular mechanics simulations using Machine Learning Algorithms, Entre Ciencia e Ingeniería, vol. 14, no. 28, pp. 82-87, julio-diciembre, 2020. DOI: https://doi.org/ 10.31908/19098367.2058.



MD granular simulations usually demand a high computational cost (in hardware and compute time) due to the number of objects simulated. For statistical analysis, it is always an advantage to decrease this time. ML supervised algorithms are good candidates for this purpose. The basic aim of these algorithms is to build predictive models, which attempt to discover and model the relationship between the predictor variable (independent variables) and the other variables. To build a predictive model, it is necessary to train the algorithm by giving instruction on what it needs to learn and how it is intended to learn it. Specifically, given a dataset, the learning algorithm attempts to optimize the model to find the combination of features that result in the target output. ML has been used successfully in combination with MD simulations in previous works such as: to accelerate ab initio MD simulations [2], to predict atomization energies in organic molecules [3], and to improve protein recognition [4], amongst others.

Machine learning is a branch of artificial intelligence interested in the development of computer algorithms for transforming data into intelligent action, and it has become the most popular and suitable tool for manipulating large data. The aim of machine learning is to uncover hidden patterns, unknown correlations and find useful information from data. Furthermore, it has extensively succeeded in performing predictive analysis [5], which is particularly of interest in this work.

For this work, the main purpose was to build such a model using ML algorithms to classify the grains of a collision, trying to find a "threshold time" for which the analyst knows with a certain accuracy what will be the outcome of the collision. With this threshold, the MD simulation can be stopped without simulating the entire collision process. It is expected that this cut time is shorter than the overall time of the simulation. The results showed that this method allows an improvement of at least more than 40 % of the total simulation time, which is encouraging.

This work is organized as follows. Section II presents a description of the type of granular simulations (subsection II-A) that are analyzed along with a description of the Machine Learning algorithm used (subsection II-B). The description of the software tools used can be seen in subsection II-C and the hardware used is in subsection II-D. The results and discussion are in section III. Finally, the main conclusions are in section IV.

II. Material and Methods

This section describes the type of granular simulations executed and the software that was used to perform them. Next, a description of the Machine Learning algorithms with their corresponding parameters is presented with the computational tools used.

A. *Granular Mechanics Simulations*

The collision of micro-metric dust grains is analyzed to find the speed limit that separates a process of agglomeration from the fragmentation. Understanding the mechanism of these processes in this scale is fundamental when trying to explain the formation of planetary rings, protoplanets, the distributions of powder grain sizes in different scenarios, etc. [6],[7].

Considering a model of granular matter where we incorporate cohesion forces [8], we perform simulations using Molecular Dynamics of collisions of porous grains composed of a N_i number of identical SiO2 particles of size 0.76 μm, where each grain has a variable fill factor between 0.15 and 0.35. We have studied the effect of the collision of two grains, a small one (the projectile) against a much larger one (the target), initially at rest. The projectile moves along the z-axis with a certain initial velocity between 0.1 and 1 m/s, and both the projectile and the target have the same filling factor (see figure 1). The filling factor is defined as the total grain volume of the number of grains N_δ in a sphere of radius δ around a certain grain *i*, divided by the sphere volume [9]. Once the projectile hits the target, the system is fragmented into two parts: one carried along by the projectile (blue grains in figure 1) and one that remains immobile (red grains in figure 1).

The critical rate of fragmentation depends on the grain's filling factor and is the result of a "piston" effect that moves in a sustained manner some of the particles of the larger cluster. The usual models estimate that the minimum speed for grain fragmentation of these characteristics is of the order of 1-10 m/s [6], [10], [11]. In our work, we found fragmentation for speeds higher than 0.02 m/s, well below the previous estimates. We also found strong dependencies on fragment sizes with the filling factor, which has not been taken into account in current models. These results may involve modifications in the agglomeration/fragmentation models used.

The simulations were executed using LAMMPS (http://lammps.sandia.gov) in GPUs (Graphics Processing Unit). The granular pair style used was initially developed by Ringl et al. [8] to run in CPUs and later ported by Millán et al. [12] to run in NVIDIA GPUs with CUDA. As a reference, the simulations in GPU run ~3.6x times faster than in one CPU core and ~1.6x faster than 8 CPU cores (NVIDIA Titan Xp compared with an AMD EPYC 7281 CPU). More in-depth benchmarks can be seen in section III-B.

The granular simulations use input data generated with a model presented in [9] and a code developed by Millán and Planes (both authors of this work). The input data is composed of two spheres of grains: a projectile with a radius of ~14 μm and a target with a radius of ~31 μm. The samples have a filling factor of 15%, 25%, and 35%. The filling factor defines how much volume is occupied in both spheres (projectile and target). The number of grains in each simulation depends on the size of the spheres and the filling factor.

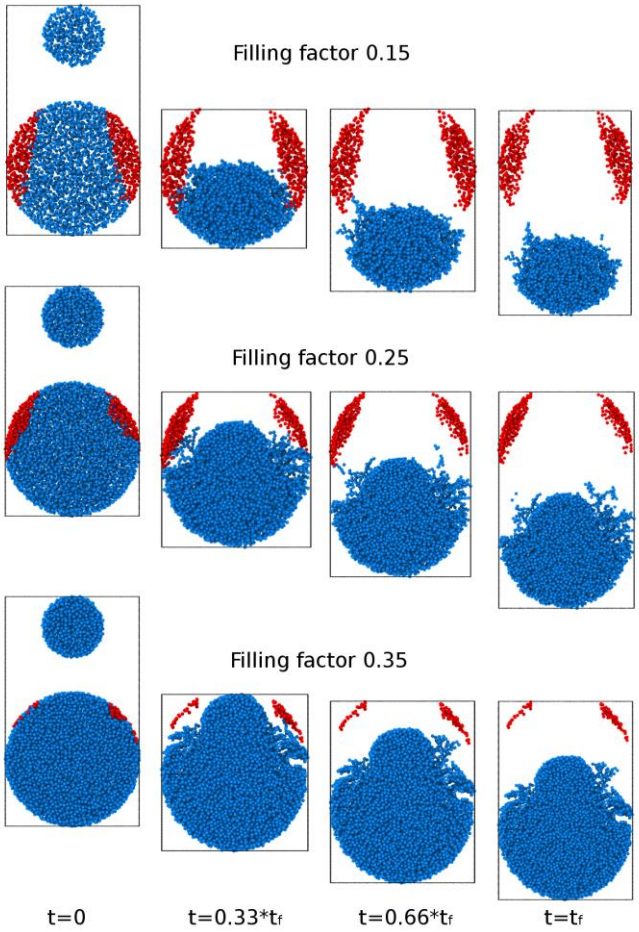


Fig. 1. Slice snapshots at different time steps (*t=0* initial time step and t_f final step) of granular simulations with three filling factors for the same initial velocity of 1m/s. Grains in different color indicate the two fragments at the end of the simulation.

The simulations were configured to generate one dump file every 10000 steps. A dump file is a text file (with a format similar to *csv* or *comma separated values* with the configuration of the simulated system at a certain time step, including positions and velocities of each grain. For our simulations, each compressed (gzip) dump file sizes are between approx. 400 KB to 1000 KB (from 0.15 to 0.35 filling factor) and the number of dumps files generated varies between the different simulated configurations. Simulations with a faster velocity (1 m/s) ran fewer Total Steps than simulations with slower velocities (0.1 m/s), which needed more Total Steps to produce fragmentation. The total size of each compressed simulation varies from approx. 1 GB (0.15 fill factor for 1 m/s) to 8.5 GB (0.35 fill factor for 0.1 m/s) with a total of 42 GB of compressed data considering all 15 simulations tested in this work.

We performed the ML algorithms analysis on five different initial projectile velocities in small simulations, each with three filling factors (0.15, 0.25, and 0.35), giving a total of 15 simulations with different input conditions. These filling factors determined the total number of grains in each simulation (11200, 18785, and 26206 respectively). The same radius of the projectile was maintained in all three cases, and it represented around 9% of the total number of grains. Five projectile velocities were tested: 1.0, 0.75, 0.5, 0.25 and 0.1 m/s.

B. *Machine Learning Algorithms*

In this work, we have tested two supervised classification models: Support Vector Machine (SVM) and Random Forest(RF).

SVM is a class of powerful, highly flexible modeling techniques. It uses a surface that defines a boundary (called hyperplane) between various points of data leading to fairly homogeneous partitions of data according to the class labels (discrete attributes whose value is to be predicted based on the values of other attributes). SVM basic task is to find this separation among a set (sometimes infinite) of possibilities, choosing the Maximum Margin Hyperplane (MMH) that creates the greatest separation between classes. For no linearly separable data, a slack variable creates a soft margin, and a cost value *C* is applied to all points that violate these constraints, and rather than finding the maximum margin, the algorithm attempts to minimize the total cost. If the cost parameter is increased, it will be more difficult to achieve a 100 percent separation. On the other hand, a lower cost parameter will place emphasis on a wider overall margin. A balance between these two must be created in order to create a model that generalizes well to future data [13].

Random Forest is an ensemble-based method that focuses only on ensembles of decision trees (a recursive partitioning method that chooses the best candidate feature each time until a stopping criterion is reached). After the ensemble of trees is generated, the model uses a vote to combine the predictions. It is possible to define the number of decision trees in the forest and also how many features are randomly selected at each split. Random forests can handle extremely large datasets, but at the same time, its error rates for most learning tasks are on par with nearly any other method [13].

C. *Computational Tools*

Several computational tools were used in this work. To perform the granular simulations, the LAMMPS software was used, running primarily in GPUs. The generated output for each simulation is in a compressed gzip format to save disk space and transfer time between computers. The Classification Analysis was performed using the R language with several packages: *dplyr* for data manipulation, *ggplot2* for plot generation, *caret* for machine learning algorithms, *purrr* to use the *map()* function and *tictoc* to easily measure compute time of sections of R code. The source code used to perform the analysis is available in the following url: https://sites.google.com/site/simafweb/.

The next subsection describes the hardware and software used to perform the granular simulations and the ML analysis.

D. Computational Tools

The MD simulations were executed using two different GPUs. The ML analysis was executed using one Workstation and the Toko cluster at FCEN-UNCuyo using only CPU cores.

The following list details the hardware and software specifications:

- Workstation FX-8350 with: AMD FX-8350 with 8 cores running at 4 GHz with 32 GB of DDR3 RAM memory. With one NVIDIA GeForce GTX Titan X (Maxwell GM200 architecture) with 12 GB of memory. Slackware Linux 14.2 64 bit operating system with kernel 4.4.14, OpenMPI 1.8.4, GCC 5.3.0, R language version 3.5.1, and Cuda 6.5 with NVIDIA driver 375.66.
- Workstation FX-8350 with: same specifications than previous workstation but with a NVIDIA GeForce GTX Titan Xp (Pascal GP102 architecture) with 12 GB of memory.
- Cluster Toko at the Universidad Nacional de Cuyo:
  - One node with four AMD Opteron 6376 CPU, with 16 CPU cores at 2.3GHz (each, 64 cores total), 128 GB of RAM, and Gigabit Ethernet.
  - One node with two AMD EPYC 7281 CPU, with 16 CPU cores at 2.1GHz (each, 32 cores total), 128 GB of RAM and Gigabit Ethernet.
  - Both nodes with Slackware Linux 14.1 64 bit with kernel 4.4.14, OpenMPI 1.8.8, GCC 4.8.2 and Cuda 6.5 with NVIDIA driver 396.26.

III. Results and Discussion

In this section, the details of the performed simulations is described along with the ML analysis. Also, a small comparison of computational performance is shown for the hardware previously described.

A. Classification Analysis

Each simulation has a large number of output files (dump files or snapshots of the state of the simulations); we chose to apply the SVM and RF prediction algorithms to only 50 evenly spaced time steps of those output files. These algorithms were trained with the velocity in *z* as the predictor variable. The accuracy (number of correct classifications over the total number of grains) of each prediction is shown in detail in figure 2. As can be seen from the figure, as the initial velocity of the projectile increases, each test takes more steps to reach an accuracy of 90%. Also, for each speed, the 0.35 fill factor takes longer to reach this accuracy.

In general, the superiority of RF performance over SVM is evident. For lower speeds and higher fill factors, the SVM performance declines. We note in some cases, that the algorithms start with an accuracy higher than 0%. This is due to the predictor variable; the grains that belong to the projectile (in movement) and the grains belonging to the target which remains immobile throughout the collision are classified correctly at the initial step. For 0.35 fill factor, the amount of target grains that remain in the immobile fragment is very small, so the initial accuracy is quite poor.

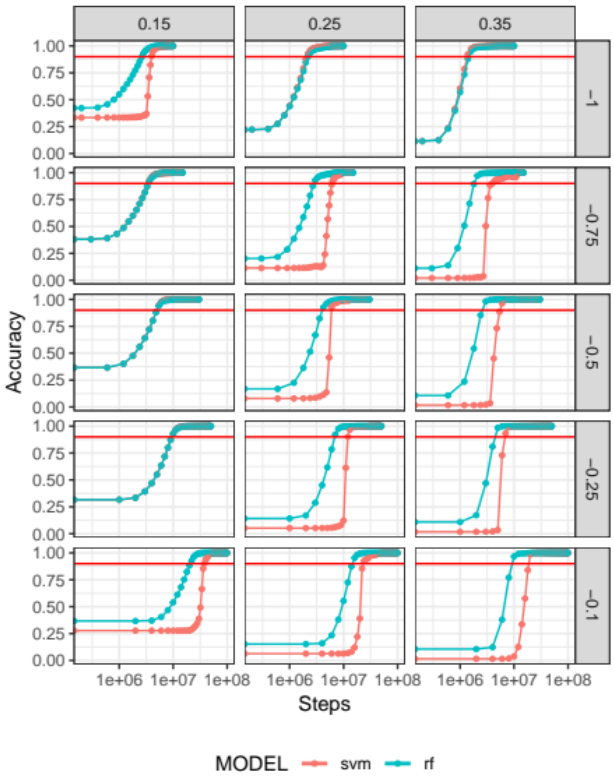


Fig 2. ML accuracy for 15 granular simulations with SVM (pink) and RF (blue) algorithms. Three filling factors are shown (0.15, 0.25 and 0.35) with five velocities (from -1 to -0.1 m/s). The dots represent 50 evenly spaced predictions for different time steps, to facilitate interpretation, the x-axis was rescaled with log function.

Results of simulation time improvements are presented for two representative speeds and two fill factors in table 1. We define the *gain percentage* as the ratio of the time taken by the algorithms in reaching a prediction accuracy of 90% and the total computation time of the simulation. We have chosen this 90% accuracy value as a reference point from which predictions start to be reliable. Analysts will determine the best accuracy value suitable for their purposes, particularly if it is desired to analyze fragment sizes. For example, to obtain 99% accuracy in the case of 0.1 m/s initial velocity and a fill factor of 0.35, the gaining percentage is 79.7% using RF (90.6% with 90% accuracy) and 65.5% using SVM (80.4% with 90% accuracy).

As can be seen from the table, in both velocities, the algorithms take longer to reach the 90% accuracy value for a 0.15 fill factor. We are currently working in the physics of this behavior in a work which is soon to be published.

As it was said previously, in general, RF has a better performance than SVM with a gap of 10% or more between gainings. The time it takes to train each algorithm is less than 1 minute, and the average time it takes to get the predictions of a file is less than 1 second, so they can be neglected.

TABLA I

WALLCLOCK TIME (COMPUTATION TIME) FOR FOUR SELECTED GRANULAR SIMULATIONS COMPARED WITH THE TIME IT TOOK EACH ML ALGORITHM TO ACHIEVE A PREDICTION WITH AN 90% OF ACCURACY. THE *GAIN PERCENTAGE* COLUMN SHOWS THE REDUCTION IN *COMPUTE TIME* THAT CAN BE ACHIEVE BY USING THE ML PREDICTIONS.

Velocity	Fill factor	Compute time	Time at 90% prediction	% gain
0.1m/s	0.15	57h 42min	rf: 14h 5min	75.6%
			svm: 25h 48min	55.3%
	0.35	61h 8min	rf: 5h 44min	90.6%
			svm: 11h 58min	80.4%
1.0m/s	0.15	4h 45min	rf: 1h 39min	65.3%
			svm: 2h 32min	46.7%
	0.35	8h 5min	rf: 1h 18min	83.9%
			svm: 1h 12min	85.1%

B. Computational Benchmarks

Granular simulations were executed in two NVIDIA GPUs, the Titan X and Titan Xp; see subsection II-D for more information on hardware and software infrastructure. In this section, a comparison between CPU and GPU performance is shown. Figure 3 shows the performance of one granular simulation executed in two CPU nodes from Toko cluster from 1 to 32 CPU cores. For this size of simulation (11200 grains), the best performance is obtained with 16 CPU cores in both CPUs (AMD Epyc and Opteron). The Epyc processor is between ~1.5 and 2.1x times faster than the Opteron processor.

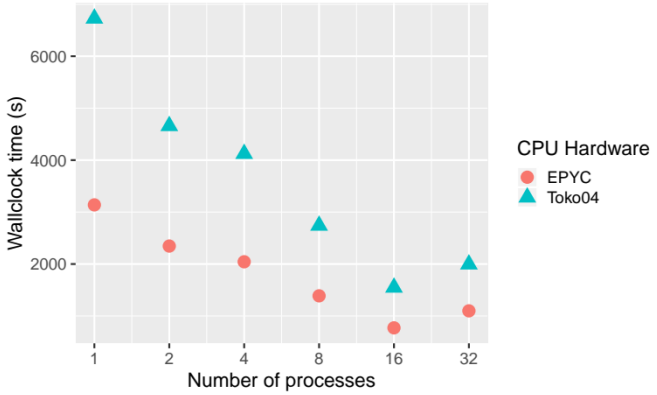


Fig. 3. Results for granular simulation with 11200 grains, velocity 5.0m/s and filling factor of 0.15, executed in two CPU cluster nodes from 1 to 32 cores. Time in seconds, lower is better.

For the same simulation shown in figure 3, the NVIDIA Titan Xp GPU has performance up to ~3.6x times faster than in one EPYC 7281 CPU core and ~1.6x faster than 8 EPYC 7281 CPU cores. The 16 EPYC CPU cores are ~1.1x faster than the Titan Xp GPU. The size of the simulations is not ideal for benchmarking, bigger simulations can produce a better performance speedup using GPUs. See [14] and [15] for more benchmarks.

IV. CONCLUSIONS AND FUTURE WORK

The computing time and HPC infrastructure needed to execute numerical simulations like the granular simulations presented in this work are a limiting factor for many researchers. Different tactics are required to reduce computational time or to improve the use of the available hardware infrastructure. Supervised ML algorithms used in this work were found suitable to build a predictive model to decrease simulation time. The overall results were encouraging: for the longest simulations, a potential 90% time reduction was achieved. The fill factor of the grains plays an important role in the algorithm's prediction; for larger filling factors, the algorithms take more time to reach the desired accuracy.

As future work, we intend to try other suitable supervised algorithms and compare their performance with the ones used so far. We also plan to explore unsupervised algorithms such as DBSCAN [16], to improve the performance of SVM and RF. We also want to explore the dependence of the supervised algorithms with the different predictor variables (positions, velocities, angular velocities). An important subject that has to be addressed in future research is Transfer Learning, to be able to train the ML models with small simulations like the ones used in this work and make predictions about larger simulations.

We are also planning to use Hadoop with Mahout (https://mahout.apache.org/) to test SVM and RF (along with other algorithms) with granular simulations including 1e5 to 1e6 grains.

ACKNOWLEDGMENTS

We acknowledge support from CONICET and grant M042-2016 SECTyP-UNCuyo. This work used Toko Cluster from FCEN-UNCuyo, that is part of SNCAD MinCyT, Argentina.

REFERENCES

[1] C. Ringl, E. M. Bringa, D. S. Bertoldi, and H. M. Urbassek, "Collisions of porous clusters: A granular-mechanics study of compaction and fragmentation," The Astrophysical Journal, vol. 752, no. 2, p. 151, Jun 2012. [Online]. Available: http://dx.doi.org/10.1088/0004-637X/752/2/151

[2] V. Botu and R. Ramprasad, "Adaptive machine learning framework to accelerate ab initio molecular dynamics," International Journal of Quantum Chemistry, vol. 115, no. 16, pp. 1074–1083, dec 2014.

[3] M. Rupp, A. Tkatchenko, K.-R. Muller, and O. A. von Lilienfeld, "Fast and accurate modeling of molecular atomization energies with machine learning," Physical Review Letters, vol. 108, no. 5, jan 2012.

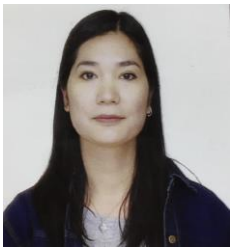
[4] D. S. Glazer, R. J. Radmer, and R. B. Altman, "Combining molecular dynamics and machine learning to improve protein function recognition," in Pacific Symposium on Biocomputing. Pacific Symposium on Biocomputing. NIH Public Access, 2008, p. 332.

[5] B. Lantz, Machine learning with R. Packt Publishing Ltd, 2013.

[6] A. D. Whizin, J. Blum, and J. E. Colwell, "The physics of protoplanetesimal dust agglomerates. VIII. microgravity collisions between porous SiO2aggregates and loosely bound agglomerates," The Astrophysical Journal, vol. 836, no. 1, p. 94, feb 2017.

[7] M. B. Planes, E. N. Millán, H. M. Urbassek, and E. M. Bringa, "Dust-aggregate impact into granular matter: A systematic study of the influence of projectile velocity and size on crater formation and grain ejection,"

Astronomy & Astrophysics, jun 2017.
[8] C. Ringl and H. M. Urbassek, “A lammps implementation of granular mechanics: Inclusion of adhesive and microscopic friction forces” Computer Physics Communications, vol. 183, no. 4, pp. 986–992, Apr2012. [Online]. Available: http://dx.doi.org/10.1016/j.cpc.2012.01.004
[9] C. Ringl and H. M. Urbassek, “A simple algorithm for constructing fractal aggregates with pre-determined fractal dimension,” Computer Physics Communications, vol. 184, no. 7, pp. 1683–1685, jul 2013.
[10] A. Seizinger, R. Speith, and W. Kley, “Compression behavior of porous dust agglomerates,” Astronomy & Astrophysics, vol. 541, p. A59, apr 2012.
[11] J. Blum, “Experiments on sticking, restructuring, and fragmentation of preplanetary dust aggregates,” Icarus, vol. 143, no. 1, pp. 138–146, jan 2000.
[12] E. N. Millán, C. Ringl, C. S. Bederián, M. F. Piccoli, C. G. Garino, H. M. Urbassek, and E. M. Bringa, “A gpu implementation for improved granular simulations with lammps,” in VI Latin American Symposium on High-Performance Computing HPCLatAm 2013, C. G. Garino and M. Printista, Eds., 2013, pp. 89–100. [Online]. Available: http://hpc2013.hpclatam.org/papers/HPCLatAm2013-paper-10.pdf
[13] A. Kassambara, Practical guide to cluster analysis in R: Unsupervised machine learning. STHDA, 2017, vol. 1.
[14] E. N. Millán, C. A. Ruestes, N. Wolovick, and E. M. Bringa, “Boosting materials science simulations by high performance computing,” in Actas de ENIEF 2017: Mecánica Computacional Vol. XXXV, AMCA. Asociación Argentina de Mecánica Computacional, Nov. 2017. [Online]. Available: https://cimec.org.ar/ojs/index.php/mc/issue/archive
[15] E. N. Millán, N. Wolovick, M. F. Piccoli, C. G. Garino, and E. M. Bringa, “Performance analysis and comparison of cellular automata GPU implementations,” Cluster Computing, apr 2017.
[16] H.-P. K. J. S. Ester, Martin and X. Xu, “A density-based algorithm for discovering clusters in large spatial databases with noise.” vol. 96, no. 34, 1996, pp. 226–231.



Daniela Noemi Rim. Bachelor’s degree in Basic Sciences with Orientation in Physics (Faculty of Exact and Natural Sciences, National University of Cuyo, Mendoza, Argentina, emitted April 17 th 2019). Thesis title: ‘Machine Learning techniques applied to Numerical Simulation of Granular Porous Materials’. Currently studying an MS in South Korea (‘MS in Information Technology’, Department of Information and Communication Engineering, Handong Global University, Pohang, Republic of Korea, since February 2020). Member of research group MILab (Machine Intelligence Lab, Handong Global University, Pohang, Republic of Korea). The topic of research focuses in Deep Learning applied to Natural Language Processing tasks (such as Neural Machine Translations) and audio signal compressions using Variational Autoencoders.

Emmanuel N. Millán. Received his Ph.D. degree from Universidad Nacional de San Luis (UNSL), Argentina in 2016, and a BSc. in Software Engineering from Universidad del Aconcagua, Argentina, in 2010. He is a researcher within CONICET. He is interested in the implementation of parallel problems in hybrid clusters including Graphics Processing Units (GPUs), with applications in Molecular Dynamics, Machine Learning, Cellular Automata and Monte Carlo methods.

María Belén Planes. 2016 “Licenciada en Ciencias Básicas con orientación en Física” FCEN – Universidad Nacional de Cuyo – Mendoza, Argentina. 2018 “Profesora de grado universitario en Ciencias Básicas con orientación en Física” FCEN – Universidad Nacional de Cuyo – Mendoza, Argentina. 2020 last-year student “Doctorado en Astronomía” FCEFN - Universidad Nacional de San Juan – San Juan, Argentina. Dust aggregate collisions have been studied through complex molecular dynamics simulations with a focus on astrophysical topics that are not currently understood, such as the formation of planets in their early stage, high speed collisions in debris discs and the evolution of dust emitted by comets in their internal coma. [Planes M. B., et al, A&A 607, A19 (2017)] [Planes M. B., et al, MNRAS: Letters 487, L13 (2019)] [Planes M. B., et al, MNRAS 492, 1937 (2020)]. SIMAF – Universidad de Mendoza. CONICET – Argentina Research areas: planetary formation – comets – granular mechanics.

Eduardo M. Bringa. Ph.D. Physics. 2000. University of Virginia (UVa), Charlottesville, USA. 1994. Licenciado en Física, Instituto Balseiro, Bariloche, Argentina. After obtaining his Ph.D., he was a postdoctoral researcher at the Astronomy Department (UVa, 2000-2001), and then postdoctoral researcher at Lawrence Livermore National Laboratory (LLNL, 2001-2003), where he later became part of the permanent research staff. In 2008 he returned to Argentina, where he currently is Principal Researcher in CONICET at the “Universidad de Mendoza”, and full Professor at “Universidad Nacional de Cuyo”. He is a member of AFA (Argentinean Physical Society) and APS (American Physical Society). **Research area:** simulations in physics, astrophysics and materials sciences, including Molecular Dynamics, Spin Dynamics and Monte Carlo simulations. https://orcid.org/0000-0002-1403-1954.

Luis G. Moyano. Is an adjunct researcher at CONICET and associate professor at Instituto Balseiro (UNCuyo/Comisión Nacional de Energía Atómica). He is invited professor at Facultad de Ciencias Exactas y Naturales, UNCuyo. He graduated in Physics from Instituto Balseiro (Bariloche, Argentina, 2000) and holds a PhD also in Physics from CBPF/UFRJ (Rio de Janeiro, Brazil, 2006). Dr. Moyano was research staff member at IBM Research Brazil until 2016. Prior to working at IBM, he was leader data scientist at BBVA Data & Analytics (Madrid, Spain). He was also staff researcher at Telefónica Research (2008-2013, Madrid/Barcelona). He is currently a permanent member of the Statistical and Interdisciplinary Physics Group at Centro Atómico Bariloche. Dr. Luis Gregorio Moyano research interests lie in the intersection of physics and machine learning. His lines of work include network representation learning, with applications to biological and social systems. Orcid: http://orcid.org/0000-0001-7704-0877.