ANNALS OF GEOPHYSICS, 68, 2, V227, 2025; doi:10.4401/ag-9262
OPEN ACCESS

Cellular Nonlinear Network for fluid dynamics: towards geophysical flows modeling

Giuseppe La Spina[*,1,2], Vito Zago[3], Luigi Fortuna[1], Arturo Buscarino[1]

(1) Università degli Studi di Catania, Dipartimento Ingegneria Elettrica Elettronica e Informatica, Catania, Italy
(2) Scuola Universitaria Superiore IUSS, Pavia, Italy
(3) Istituto Nazionale di Geofisica e Vulcanologia, Osservatorio Etneo, Catania, Italy

Article history: received January 21, 2025; accepted April 12, 2025

Abstract

This work proposes an application of Cellular Nonlinear Networks (CNNs) to fluid dynamics, implemented towards the application to geophysical flows. We revisit the CNN paradigm for the solution of Partial Differential Equations with a focus of some main physical properties of geophysical flows. We address numerical aspects of the obtained model, including the treatment of boundary conditions, stability and correctness of the results. This is done by applying our method to canonical testcases and comparing the results with accepted benchmark data and analytical solutions from the literature. The parallelizability of the method is also assessed by means of benchmarking tests on a multi-core processor. We validate the model by simulating classical geophysical flows. The results that we present show the capability of the model to reproduce typical velocity profiles of viscous flows, making it promising for applications that require the simulation of such materials.

Keywords: Cellular Nonlinear Networks; Navier-Stokes equations; Fluid dynamics; Lava flows

1. Introduction

The simulation of fluid dynamics phenomena is a key topic in the field of nonlinear systems, especially when aiming to understand and forecast complex phenomena such as the behavior of fluids with peculiar properties, like lava or gas plumes. These phenomena show a complex nature, characterized by multiple interactions between many different physical and chemical variables. Understanding and simulating these processes poses a great scientific challenge and becomes essential for monitoring tasks and prevention in those regions with high risk of volcanic activity.

A significant challenge in geophysical modeling involves finding a compromise between two opposite requirements: increasing the physical accuracy of the simulation and reducing computational cost, that could result in unpractical computation times (Zago et al., 2023). There exist many different approaches to modeling geophysical flows, ranging from highly detailed numerical models, such as Smoothed Particle Hydrodynamics (SPH) (Zago et al., 2017), to simpler grid-based analytical methods like Cellular Automata (Miyamoto and Sasaki, 1997; Rongo et al., 2016). The choice of the compromise is functional to the application: numerical models provide a detailed representation but at a high computational cost, while Cellular Automata produces fast results at the expense of generalizability to different physical scenarios.

Here we repropose the Cellular Nonlinear Network (CNN) approach (Chua and Yang, 1988a, 1988b; Del Negro et al., 2005; Kozek et al., 1995; Roska et al., 1995) to the problem as a further compromise option between detailed physical modeling and computational efficiency. CNNs have a structure consisting of many simple units with a local connectivity pattern capable of representing generic complex spatio-temporal phenomena. This approach directly operates on governing equations, guaranteeing accurate and versatile simulations, especially when compared to Cellular Automata or empirical models.

The CNN paradigm was originally conceived as an electronic implementation of a solver for Partially Differential Equations (PDEs) and image processing techniques in real-time, that exploits the simultaneous cooperation of electronic analog cells to obtain solutions in convenient times.

Other than relying on electronic implementation, the solving paradigm can be implemented numerically. In this context the intrinsic parallel nature of CNNs is of fundamental interest, considered the growing adoption of High-Performance Computing (HPC) architectures, which keep achieving new levels of parallelization. Beside modern CPUs, that offer great parallelization capability, scientific modeling is experiencing an expanding adoption of Graphic Processing Units (GPUs), which confer noticeable computational benefits to algorithms with an intrinsic parallel nature (Hérault et al., 2010; Zago et al., 2018). Moreover, newer computing architectures like Tensor Processing Units (TPUs) (Wang et al., 2019) offer deeper levels of parallelization, performing tensorial operations at extremely high efficiency thanks to their dedicated hardware.

Considering the momentum on Hardware parallelization, we revisit the CNN paradigm applied to the simulation of geophysical flows. Our work extends the developments of (Del Negro et al., 2005) that firstly studied the feasibility of the method applied to geophysical flows, specifically in the case of simplified lava flows. While their work focused on developing a solution with an analog implementation, we proceed with emphasizing the numerical aspects of the method with respect to the Eulerian discretization of the equations and their adaptability to a fully digital CNN structure. This is achieved by carrying out quantitative validation of the model using benchmark data and analytical formulas for well consolidated testcases. This was done while also considering potential improvements in computation time.

Another key contribution with respect to their work lies in the method used for pressure computation. In the formulation presented we adopted an approach that allows to compute the pressure layer independently, ensuring that the velocity vector field remains divergence-free also in the digital implementation, improving stability and physical consistency.

Even though a more general formulation can be implemented following the same approach described in this manuscript, many geophysical flows, such as lava and low-speed atmospheric flows, under certain flow regimes show negligible compressibility characteristics. For this reason, in the rest of this work we will consider a formulation for incompressible flows, based on the Navier-Stokes equation for momentum conservation and on the solution of the Poisson equation to solve pressure fields, which ensures that the incompressibility constraint is satisfied.

Section 2 of the manuscript presents the description of the CNN formalism adopted. In Section 3 we introduce the Navier Stokes equations and their implementation with the CNN approach, with a focus on the solution of the pressure field. Then Section 4 presents the validation results.

We also discuss the numerical results of the simulations and the possible application in the context of management of volcanic hazard and urban planning.

2. Cellular Nonlinear Networks

The invention of CNN has been correlated to the possibility of implementing highly parallel processing units in analog (electronic) circuits. This was strongly motivated by real-time applications in image processing.

Moreover, further applicability of CNN has been strongly confirmed over time. In fact, even if CNNs had their major impact due to their unique analog electronic features, what has been highly impressive is their structure which employs large-scale connections. Indeed, the high-depth processing units have led to the implementation of CNN in the discrete time domain, introducing a new parallel computation paradigm exclusively based on the CNN structure.

Furthermore, the architecture of CNN based systems aligns perfectly with that of a discretized, inherently nonlinear, system of differential equations (Arena et al., 2005). This fact has been strongly emphasized in the literature (Fortuna et al., 2003).

Recent studies have demonstrated the versatility of CNN architecture in various domains such as neurosciences and circuit modeling. This can be found, for example, in the work of (Wu et al., 2024) where a memristive CNN cell is used to replicate the spiking behavior of neurons in the brain by adjusting system and input parameters of the cells. Also, CNNs have been applied to explore the propagation of impulses in nerve membranes, as shown in the work of (Agarwal et al., 2024), where bifurcation study of the model and the emergence of chaos is analyzed. Furthermore, the work of (Ma et al., 2024) analyzes the stability and origin of bursting dynamics in a CNN based system, showing how these can exhibit different bursting oscillations with one or two periodic forcing signals.

CNNs are dynamical systems consisting of matrix-style interconnected cells. Each of these cells is paradigmatically modeled as an electrical circuit with a linear capacitor and a resistor, and a series of voltage-controlled current sources. The latter depend on the state of neighboring cells, thus providing the abovementioned interconnection, and on external stimuli. An additional nonlinear output stage, usually based on a simple saturation function, is also included.

Here we consider Multi-Layer CNNs (ML-CNNs), which are a particular kind of Cellular Nonlinear Network made up of many different layers (Roska et al., 1995). Each layer can be associated with a state variable of the modeled system.

Based on previous discussion, each cell on layer m can be described by the voltage across its capacitor $u_{i,j}^{m}$. This mathematical model can be used to describe its dynamics:

$$C\frac{du_{i,j}^{m}}{dt} = -\frac{1}{R_c} + \sum_{m'=1}^{M}\left[\sum_{(k,l)\in\mathcal{N}(i,j)} a_{k-i,l-j}^{m'm}\left(u_{k,l}^{m'}, u_{k,l}^{m}\right)\right] + \sum_{(k,l)\in\mathcal{N}(i,j)} b_{k-i,l-j}^{m}\left(F_{k,l}^{m}(t), F_{i,j}^{m}(t)\right) + I \tag{1}$$

Where C is the value of the cell capacitance, R_c is the value of the cell resistor, I is an external input to the cell. Eq. (1) states that the rate of change of state variable $u_{i,j}$ depends on itself and on its own input. Moreover, it is influenced by the cells that are in its neighborhood. The following set can be utilized to define the radius of connectivity, named as the r-neighborhood:

$$\mathcal{N}_r(i,j) = \{(k,l) : |k-i|, |l-j| \leq r\} \tag{2}$$

Furthermore, we have the nonlinear function $a_{k-i,l-j}^{m'm}\left(u_{k,l}^{m'}, u_{k,l}^{m}\right)$ that accounts for the effect of the neighborhood state variables from layer m', while the nonlinear function $b_{k-i,l-j}^{m}\left(F_{k,l}^{m}(t), F_{i,j}^{m}(t)\right)$ weights the effects of external inputs on neighboring cells on the dynamic of the cell.

We consider a 1-neighborhood CNN. This choice is justified, as we will see in the next section, by the fact that many physical problems are described by Partial Differential Equations (PDEs) that are at most of second order. By applying a central discretization scheme, the influence of a cell is limited only to adjacent neighbors. Therefore, we can express in the previously defined functions that characterize the strength of the interactions in a more compact way:

$$A^{m'm} = \begin{bmatrix} a_{-1,-1}^{m'm} & a_{-1,0}^{m'm} & a_{-1,1}^{m'm} \\ a_{0,-1}^{m'm} & a_{0,0}^{m'm} & a_{0,1}^{m'm} \\ a_{1,-1}^{m'm} & a_{1,0}^{m'm} & a_{1,1}^{m'm} \end{bmatrix} \tag{3}$$

$$B^{m} = \begin{bmatrix} b_{-1,-1}^{m} & b_{-1,0}^{m} & b_{-1,1}^{m} \\ b_{0,-1}^{m} & b_{0,0}^{m} & b_{0,1}^{m} \\ b_{1,-1}^{m} & b_{1,0}^{m} & b_{1,1}^{m} \end{bmatrix} \tag{4}$$

In particular, the matrices $A^{m'm}$ and B^{m}, which are referred to as feedback templates and control templates respectively, fully specify how a layer m' interact with layer m and how external inputs affect the cell.

This general formulation allows us to represent with CNNs a wide plethora of spatio-temporal phenomena. The exact choice of the templates, in fact, allows us to map a specific dynamic onto the CNN paradigm.

2.1 Boundary conditions

Given that CNNs are mainly used to solve boundary value problems, properly handling the boundary conditions is fundamental to maintain physical reliability and proper stability of the simulation. These practically consist in defining the behavior of the previously described structure at the border of the domain by imposing which value the state variables must assume at said border.

There are several possible choices for the boundary conditions. For example, Dirichlet Boundary Conditions consist in fixing the value of the variable to a constant; Periodic Boundary Conditions enforce a repeating pattern at the domain boundaries; Neumann boundary conditions consist in fixing the gradient of the considered field to a constant.

By properly choosing the boundary conditions it is possible to model different characteristics. Hence, by fixing the value of the velocity to zero, we model the effect of stationary walls, while imposing the velocity to a specific value models inflow/outflow regions (inlets and outlets).

In the following, we will describe the general procedure to discretize a Partial Differential Equation (PDE) problem and apply it to a CNN structure.

3. Governing equations

Many natural phenomena can be described by PDEs where derivatives do not exceed the second order. Examples can include Maxwell's equations, the heat equation, and reaction-diffusion equations. For such equations finding a general analytical solution can be difficult and sometimes impossible. For this reason, for many complex problems, it can be useful to adopt numerical techniques.

We propose to use central differencing scheme for discretizing the derivatives, as it is second order accurate approximation (LeVeque, 2007), thus having:

$$\frac{\partial f(x_j, \dots, x_i, \dots, x_k)}{\partial x_i} \approx \frac{f(x_j, \dots, x_i + h, \dots, x_k) - f(x_j, \dots, x_i - h, \dots, x_k)}{2h_i} \tag{5}$$

$$\frac{\partial^2 f(x_j, \dots, x_i, \dots, x_k)}{\partial x_i^2} \approx \frac{f(x_j, \dots, x_i + h, \dots, x_k) - 2f(x_j, \dots, x_i, \dots, x_k) + f(x_j, \dots, x_i - h, \dots, x_k)}{h_i^2} \tag{6}$$

Eq. (5) and (6) show a general formulation with $k \in \mathbb{N}$ spatial variables. In this work we consider a two-dimensional domain, given that extending it to higher dimensions is straightforward.

We choose a uniform step size h and a grid of $M \times N$ discrete points in space. Then L_x and L_y, which are the lengths of the simulated domain in the x-direction and in the y-direction respectively, can be obtained as $L_x = M \cdot h$ and $L_y = N \cdot h$.

By computing Eq. (5) for each cell, the PDEs are approximated by a set of $M \times N$ Ordinary Differential Equations (ODEs) with respect to the time variable.

This formulation is easily adapted to the CNN modeling paradigm because spatial derivatives have been converted to terms that depend on neighboring cells. The main advantage of this approach lies in its applicability to many physical systems, including the ones mentioned above.

We remark that, in the case where systems have derivation order higher than two the same modeling technique can be exploited but increasing the considered neighborhood to account for the higher order of differentiation.

Here we summarize the procedure to model a physical system with the CNN paradigm. These steps are quite general and can be applied to many problems:

– Identify the mathematical equations that describe the physical system.

- Define and then discretize the spatial domain with step size h, obtaining a grid of $M \times N$ cells.
- Apply central differencing scheme to spatial derivatives, converting the problem into a system $M \times N$ ODEs.
- Construct template matrices, that represent the influence of neighboring cells and external inputs.
- Define boundary conditions for the cells at the border of the domain.

In the following sections we will introduce and adapt the Navier-Stokes equations to the CNN paradigm.

3.1 Subsection Navier Stokes equations

Fluids are materials that undergo constant and continuous deformation when subjected to a force, and it can be difficult to build a mathematical model that completely represents their behavior.

The Navier-Stokes equations, which account for viscosity and interactions with objects, can be used to describe the complex mechanics of these materials, like lava flows. In this section we describe the Navier-Stokes equations applied only to the solution of incompressible fluids, even though this method could be extended to other kind of fluids.

The flow of a fluid can be described by the momentum equation:

$$\frac{\partial \mathbf{u}}{\partial t} + \mathbf{u} \cdot \nabla \mathbf{u} = -\frac{\nabla p}{\rho} + \mu \nabla^2 \mathbf{u} \tag{7}$$

Where p is the pressure in [kg m^{-1} s^{-2}], ρ is the density in [kg/m^3], μ is the dynamic viscosity in [kg m s^{-1}] and $\mathbf{u}$ is the vector of velocities:

$$\mathbf{u} = \begin{bmatrix} u \\ v \end{bmatrix} \tag{8}$$

with the components $u(x,y,t)$ and $v(x,y,t)$ which are, respectively, the velocities components over the x-direction and y-direction in [m/s].

In our case we suppose to examine incompressible flows, thus we also combine Eq. (5) with the continuity equation, written as:

$$\nabla \cdot \mathbf{u} = 0 \tag{9}$$

With the previous assumption, we can apply the divergence to Eq. (5) thus obtaining the pressure Poisson equation:

$$\nabla^2 p = -\rho \left[\left(\frac{\partial u}{\partial x} \right)^2 + 2 \frac{\partial u}{\partial y} \frac{\partial v}{\partial x} + \left(\frac{\partial v}{\partial y} \right)^2 \right] \tag{10}$$

Eq. (5) and (8) can be used to get three equations, with the three unknowns u, x and p for each point in space.

No exact solutions exist for solving the Navier Stokes equations, hence the only feasible way for determining how the flow develops is through numerical solution of the differential equations.

Supposing that we are interested in a solution over a confined domain Ω with:

$$\Omega = \{(x, y) \in \mathbb{R} | 0 < x < x_{max}; 0 < y < ymax\} \tag{11}$$

we can discretize it spatially with space steps Δx and Δy. So, for a function $u(x,y,t)$ we can write:

$$u(x,y) = u(i\Delta x, j\Delta y) \triangleq u_{i,j} \tag{12}$$

We implement the Navier Stokes equations by means of a two-layers CNN handling the two velocity components. If the spatial resolution is high enough, i.e. Δx and Δy are small, we can make the following approximations:

$$\frac{\partial u}{\partial x} \approx \frac{u_{i+1,j} - u_{i-1,j}}{2\Delta x}$$

$$\frac{\partial u}{\partial y} \approx \frac{u_{i,j+1} - u_{i,j-1}}{2\Delta \mathrm{y}}$$

$$\frac{\partial^2 u}{\partial x^2} \approx \frac{u_{i+1,j} - 2u_{i,j} + u_{i-1,j}}{\Delta x^2} \tag{13}$$

At first, we write Eq. (5) explicitly for each component, which results in:

$$\begin{cases} \dfrac{\partial u}{\partial t} + u\dfrac{\partial u}{\partial x} + v\dfrac{\partial u}{\partial y} = -\dfrac{1}{\rho}\dfrac{\partial p}{\partial x} + \nu\left(\dfrac{\partial^2 u}{\partial x^2} + \dfrac{\partial^2 u}{\partial y^2}\right) \\ \dfrac{\partial v}{\partial t} + u\dfrac{\partial v}{\partial x} + v\dfrac{\partial v}{\partial y} = -\dfrac{1}{\rho}\dfrac{\partial p}{\partial y} + \nu\left(\dfrac{\partial^2 v}{\partial x^2} + \dfrac{\partial^2 v}{\partial y^2}\right) \end{cases} \tag{14}$$

Then, applying the previously defined approximations for the spatial derivatives, we may discretize:

$$\frac{du_{i,j}}{dt} = -u_{i,j}\frac{u_{i+1,j} - u_{i-1,j}}{2\Delta x} - v_{i,j}\frac{u_{i,j+1} - u_{i,j-1}}{2\Delta \mathrm{y}} - \frac{1}{\rho}\frac{p_{i+1,j} - p_{i-1,j}}{2\Delta x} +$$

$$+\nu\left(\frac{u_{i+1,j} - 2\,u_{i,j} + u_{i-1,j}}{\Delta x^2} + \frac{u_{i,j+1} - 2\,u_{i,j} + u_{i,j-1}}{\Delta y^2}\right) \tag{15}$$

In this way we have a set of nonlinear ordinary differential equations in the variables $u_{i,j}$ and $v_{i,j}$

Following the CNN approach, we can write the state equation of the velocity along the x-direction as:

$$\frac{du_{i,j}}{dt} = -A_{uu1} * U_{i.j}^N - A_{uu2} * U_{i.j}^N - A_{up1} * P_{i.j}^N + A_{uu3} * U_{i.j}^N =$$

$$= (-A_{uu1} - A_{uu2} + A_{uu3}) * U_{i,j}^N - A_{up1} * P_{i.j}^N \tag{16}$$

Where we indicate with the symbol "$*$" the sum of the inner product, i.e. the Frobenius matrix product. The other matrices are spatial dependent terms that are hereby defined as:

$$A_{uu1}(i,j) = \frac{u_{i,j}}{2h}\begin{bmatrix} 0 & 0 & 0 \\ -1 & 0 & 1 \\ 0 & 0 & 0 \end{bmatrix}$$

$$A_{uu2}(i,j) = \frac{v_{i,j}}{2h}\begin{bmatrix} 0 & -1 & 0 \\ 0 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix}$$

$$A_{uu3}(i,j) = \frac{\nu}{h^2}\begin{bmatrix} 0 & 1 & 0 \\ 1 & -4 & 1 \\ 0 & 1 & 0 \end{bmatrix} \tag{17}$$

In addition, we can define for each state variable the neighbor matrix, like in the case of the discretized variable $u_{i,j}$:

$$U_{i,j}^{N} = \begin{bmatrix} u_{i-1,j-1} & u_{i-1,j} & u_{i-1,j+1} \\ u_{i,j-1} & u_{i,j} & u_{i,j+1} \\ u_{i+1,j-1} & u_{i+1,j} & u_{i+1,j+1} \end{bmatrix} \tag{18}$$

To investigate the Poisson Eq. (8) and to find a solution, the discretized differential operators of Eq. (11) and Eq. (8) are applied to both sides of the equations. For the left-hand side (LHS) we simply have:

$$\nabla^2 p \approx \frac{p_{i+1,j} - 2\,p_{i,j} + p_{i-1,j}}{\Delta x^2} + \frac{p_{i,j+1} - 2\,p_{i,j} + p_{i,j-1}}{\Delta y^2} \tag{19}$$

while for the right-hand side (RHS) we can write:

$$-\rho\left[\left(\frac{\partial u}{\partial x}\right)^2 + 2\frac{\partial u}{\partial y}\frac{\partial v}{\partial x} + \left(\frac{\partial v}{\partial y}\right)^2\right] =$$

$$= -\rho\left[\left(\frac{u_{i+1,j} - u_{i-1,j}}{2\Delta x}\right)^2 + 2\frac{v_{i+1,j} - v_{i-1,j}}{2\Delta x}\frac{u_{i,j+1} - u_{i,j-1}}{2\Delta y} + \left(\frac{v_{i,j+1} - v_{i,j-1}}{2\Delta y}\right)^2\right] =$$

$$= \rho b_{i,j} \tag{20}$$

Combining the two equations above we obtain a system of linear equations in the unknowns $p_{i,j}\ \forall\ i,j$ in the domain of interest.

When applying temporal discretization to this problem, a further consideration has to be made. In this way we have no guarantee that the velocity vector field remains divergence free and, to enforce this incompressibility condition, we consider a corrective term. This compensates for the instabilities that come from numerical errors building up during simulation. To guarantee that the vector field u is solenoidal, we consider the pressure projection method by (Chorin, 1968) so the following correction has to be kept into account:

$$\frac{1}{h^2}\left(p_{i-1,j} + p_{i+1,j} + p_{i,j-1} + p_{i,j+1} - 4p_{i,j}\right) =$$

$$= -\frac{\rho}{\Delta t}\left(b_{i,j} + \frac{u_{i+1,j} - u_{i-1,j}}{2h} + \frac{v_{i,j+1} - v_{i,j-1}}{2h}\right) \tag{21}$$

Where we assumed $h = \Delta x = \Delta y$. To compute the pressure layer, we have to solve the linear system of equations in (19). A possible way for solving a sparse linear system is to apply one of the existing methods, like Jacobi or

Gauss-Seidel over-relaxation method (Barrett et al., 1994). In our case, in order to keep computational times short, we proceeded by vectorizing the matrix of $M \times N$ unknowns P and directly solving for it:

$$A \cdot P_{vect} = b \Rightarrow (A^{-1}A)P_{vect} = A^{-1}b \Rightarrow P_{vect} = A^{-1}b \tag{22}$$

where matrix A has been appropriately defined according to the following rules:

- if $p_{i,j}$ belongs to the boundary we assign the corresponding term of matrix A to 1 and the term of the vectorization of b to the proper value so to have:

$$b = p_{i,j} \tag{23}$$

- if $p_{i,j}$ doesn't belong to boundary we assign to the corresponding row the value $1/h^2$ to the neighbors and −4 on the diagonal element

We can then calculate the inverse of matrix A and multiply it on the left to determine P.

4. Numerical Analysis

In this section, we discuss the numerical simulation of a CNN used for solving Navier-Stokes equations in a two-dimensional array, standing for a planar section of the fluid. For doing so, it has been written a code in python language that solves for the spatially discretized equations in Eq. (13), exploiting the simplification introduced by the CNN formulation that makes the problem suited for solving it with the calculator.

The equations have been solved using Euler method for discretizing the time derivative of the state variables:

$$\frac{du_{i,j}}{dt} \approx \frac{u_{i,j}^{N+1} - u_{i,j}^{N}}{\Delta t} \tag{24}$$

$$\frac{dv_{i,j}}{dt} \approx \frac{v_{i,j}^{N+1} - v_{i,j}^{N}}{\Delta t} \tag{25}$$

We are adapting a PDE problem to a CNN, and for this reason, the same rules that apply with discretized PDEs must be considered here. The time step Δt must be chosen accordingly to ensure that the simulation is numerically stable.

For a viscous fluid we consider two stability conditions (Zago et al., 2018), that are derived by the effect of the flow velocity:

- One that depends on the maximum magnitude of the flow velocity

$$\Delta t \leq \frac{h}{\max(|u|, |v|)}$$

- And one that depends on the viscous diffusivity:

$$\Delta t \leq \frac{h^2}{2\nu}$$

Where u and v are the velocity components along the x-direction and the y-direction respectively. The step size has to be chosen small enough to maintain numerical stability but it must also be large to produce faster simulations.

Next, we proceed with presenting a few scenarios analyzed using the model described in the previous section, examining their outcomes, and validating the effectiveness of the proposed approach.

For this analysis, we define some useful parameters for the simulations such as the Reynolds number. This is a dimensionless number that expresses the ratio between inertial forces and viscous forces, and can be derived as:

$$Re = \frac{LU}{\nu} \tag{26}$$

where L is the characteristic length of the system, U is the speed of the flow and ν is the kinematic viscosity of the fluid. This indicator can be used to estimate the nature of the motion and whether the flow regime is laminar or turbulent.

Also, the main aim of this task is to validate the numerical accuracy of the model and its capability to manifest generic scenarios. For this reason, the physical parameters of the systems have been properly chosen to assess that the latter arise correctly.

4.1 Simulation of a lid-driven flow

The first experiment to confirm the model behavior consists of a simulation of a lid-driven flow within a cavity. Lid-driven Cavity flow is a classical computational problem used as a standard validation example (Peng et al., 2003; Hou et al., 1995). Even though this setting does not provide general validation capabilities, it can be useful to highlight the capacity of the software to set flexible boundary conditions.

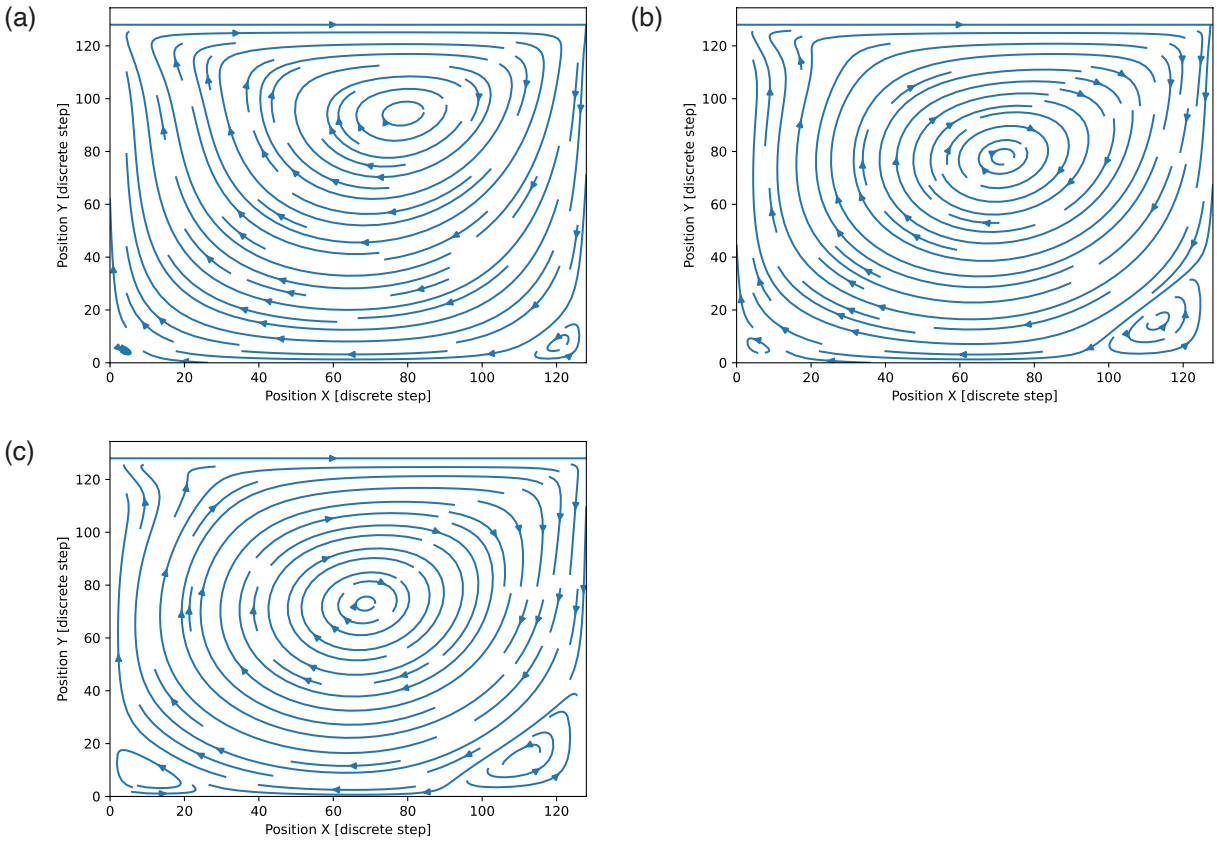

Figure 1. Streamline plots of the vector field for the lid driven cavity flow at regime with Reynolds Number (a) 100, (b) 400 and (c) 1000. As the Reynolds number increases, the primary vortex shifts position, moving towards the geometrical center of the cavity. In addition, secondary vortices increase in side, with their origin also translating towards the center.

The lid-driven flow scenario depicts a phenomenon that might not be present in physical scenarios directly related to lava flows but can be useful to assess the capability of the software to reproduce real spatio-temporal dynamic processes.

The validation procedures is related to the results of (Ghia et al., 1982), considered as a valid benchmark for fluid dynamics simulations. In this work the authors report the results of the simulation for a domain Ω consisting of a box with four walls, where we indicate with l_l, l_T, l_R, l_B the left, top, right and bottom wall respectively. The dimensions of the box are of $x_{max} = 1$ m and $y_{max} = 1$ m. The domain has been discretized with a grid 129x129, thus leading to a space step of $h = 1/129$ m. Then the simulation has been repeated for various values of Reynolds number Re. For this case the following boundary conditions were considered: at the top wall $u_{IT} = 1$ m/s and $v_{IT} = 0$ m/s, while $u = 0$ m/s and $v = 0$ m/s on all the other walls.

The image in Fig. 1 shows the outcome of a simulation at regime and highlights the behavior induced in the cavity by the movement at the upper lid. Flow appearing as going through the walls is a border effect of the streamline representation.

The obtained results are consistent with those in (Hou et al., 1995), where some discrepancies on the onset of vortices at the cavity corners are associated with the adopted spatial resolution of the plotting library. The comparison shows how, increasing the Reynolds number, the pictures show stronger circulation of the flow, with a corresponding increase in the size and prominence of the corner vortices.

Table 1 presents the locations of the primary, lower-left, and lower-right vortices for Reynolds numbers of 100, 400, and 1000. A comparison between our findings and those reported by (Hou et al., 1995) show good agreement and low percentage error between our results and previous the previously cited work.

To further reinforce the validity of our approach the results have then been compared to the tabbed data of (Ghia et al., 1982) to show the numerical consistency.

In Fig. 2 the u-velocities and the v-velocities are reported along the vertical and the horizontal center lines, respectively, and are compared with the reference data. This comparison shows a good correspondence

Re		Primary vortex		Lower left vortex		Lower right vortex	
		x [m]	y [m]	x [m]	y [m]	x [m]	y [m]
100	a)	0.6196	0.7373	0.0392	0.0353	0.9451	0.0627
	b)	0.6124	0.7326	0.0310	0.0388	0.9380	0.0620
	Error [%]	0.72%	0.47%	0.82%	0.35%	0.71%	0.07%
400	a)	0.5608	0.6078	0.0549	0.0510	0.8902	0.1255
	b)	0.55426	0.6046	0.0465	0.0465	0.8760	0.1162
	Error [%]	0.65%	0.32%	0.84%	0.45%	1.42%	0.93%
1000	a)	0.5333	0.5647	0.0902	0.0784	0.8667	0.1137
	b)	0.5349	0.5659	0.0775	0.0775	0.8605	0.1162
	Error [%]	0.16%	0.12%	1.27%	0.09%	0.62%	0.25%

Table 1. Location of primary, lower left and lower right vortex for Reynolds number equal to 100, 400 and 1000. a) (Hou et al., 1995); b) present work. The relative error is expressed in percentage respect to the domain length.

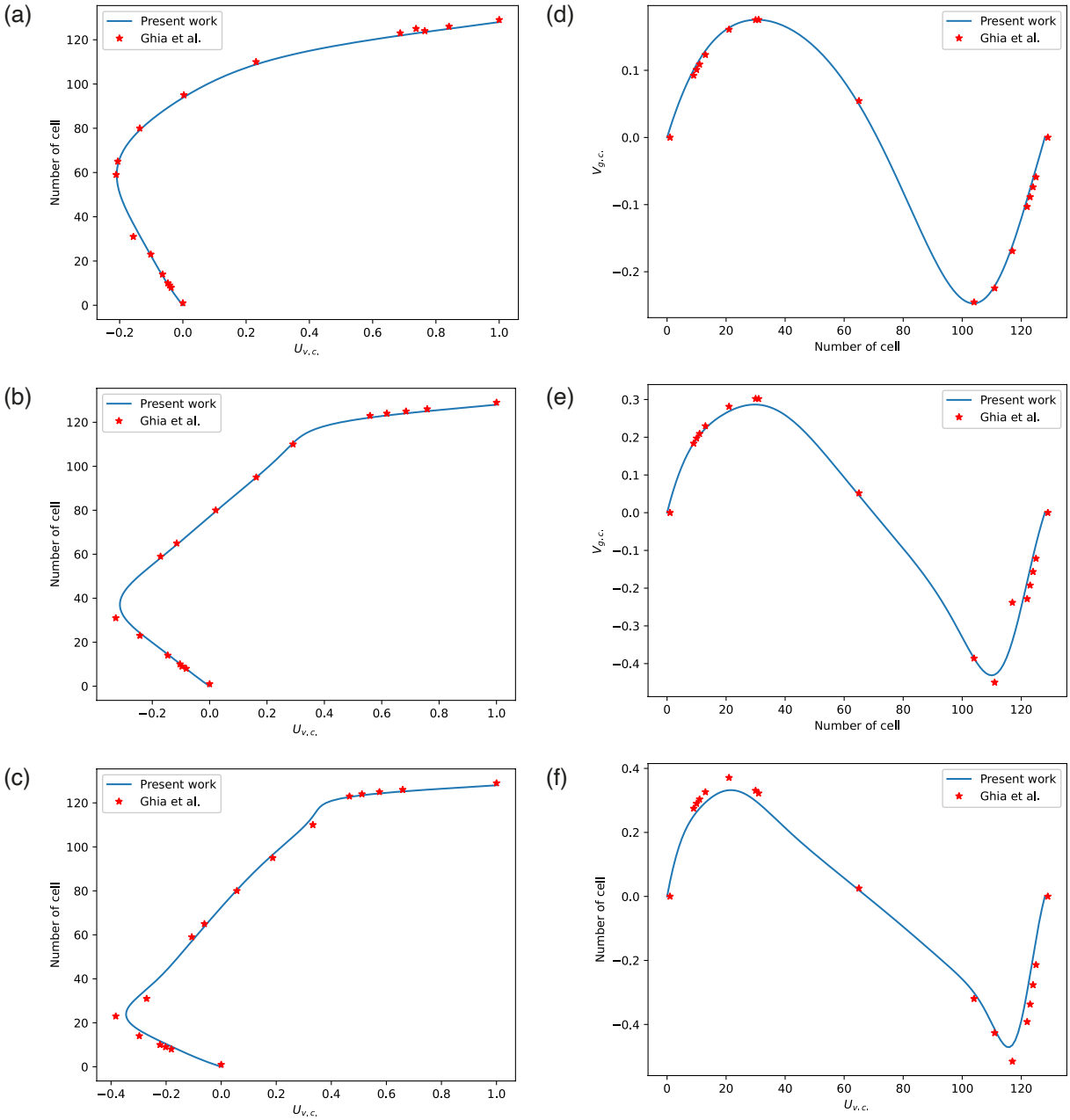

Figure 2. u-velocity profile along vertical centerline, corresponding to $x = 0.5$ m, with Reynolds number equal to (a) 100, (b) 400 and (c) 1000 digitized profiles. On the right column v-velocity components along the central horizontal line, corresponding to $y = 0.5$ m, with Reynolds number equal to (a) 100, (b) 400 and (c) 1000. The obtained results, indicated with blue lines, are compared with (Ghia et al., 1982) digitized profiles, indicated with red asterisks.

between the work of this paper and the results in literature, especially for $Re = 100$ and $Re = 400$. For $Re = 1000$ the correspondence remains high even though discrepancies can be observed near the local maxima and minima. These discrepancies can be attributed to sharper velocity changes and increasing turbulence of the flow. However, the observed numerical accuracy remains high and confirms the robustness of CNN approach regarding the simulation of such flows.

4.2 Vortex shedding

As a second case study, the emergence of vortex shedding around an obstacle put inside a canalized flow has been investigated. This is a physical phenomenon consisting in the formation of alternating vortices downstream

the obstacle if certain conditions of speed, size, and shape are met, and is caused by an imbalance between inertial and viscous forces. This phenomenon can be the cause for disruptive oscillating stresses on structures (Billah and Scanlan, 1991) hence its analysis is crucial in engineering. Moreover, this phenomenon can have practical implications in geophysical contexts such as airflows around volcanic domes.

We consider a domain Ω corresponding to a channel with $x_{max} = 9$ m and $y_{max} = 3.5$ m and a circular obstacle. Adopting a spatial discretization step $h = 0.06$ m we obtain a domain of (150×70) cells.

The velocity at the top and at the bottom of the channel is set to 0 m/s. On the vertical side edges of the domain, we set a constant velocity of $U = 5$ m/s so that they act as an inlet (on the left) and as outlet (on the right). To ensure that vortex shedding happens we consider a Reynolds number of 350. Considering $\nu = 0.01$ m^2/s, $\rho = 1$ kg/m^2 it corresponds to a diameter of the obstacle $d_s = 0.6$ m.

The circle is positioned close to the inlet, centered at cell 25×30, hence slightly under the channel midline. This slight asymmetry helps the onset of vortex shedding.

The results obtained, shown in Fig. 3, reproduce the formation and separation of the vortices beyond the circular obstacle, forming the well-known Kármán Vortex Street. The analysis of the velocity field and of the stream plot offers

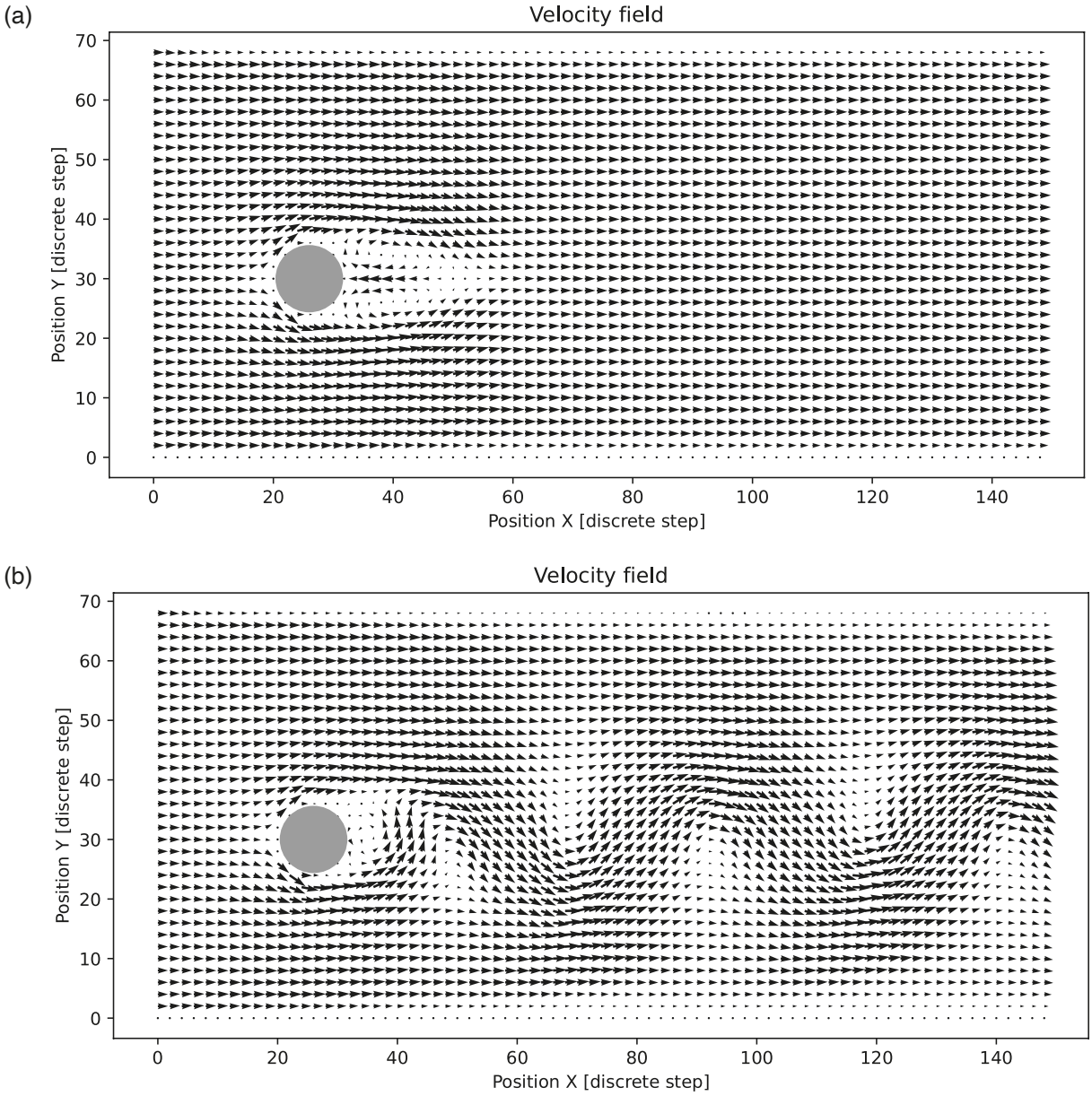

Figure 3. Quiver plot of the velocity field for the canalized flow (a) at $t = 0.5$ s the flow separation process begins (b) at $t = 5$ s the flow shows periodic patterns.

a visual representation of the process, highlighting the presence of periodic structures generated. This successful reproduction suggests the accuracy of the model for capturing this specific nonlinear and time-dependent complex phenomena.

To quantify the spatio-temporal periodicity, we consider the Strouhal number, defined as:

$$St = f\frac{d_s}{U} \tag{27}$$

where f is the oscillation frequency, U is the inlet flow velocity, and $d_s$is the circle diameter. For a high enough Reynolds number, the Strouhal number is constant and is approximately 0.2.

In our case the measured frequency for the vortex shedding is of 2 Hz, thus leading to a value of the Strouhal number of 0.24. While the result is close to the numerical value, the slight discrepancy can be attributed to the channel boundary and to the simplistic integration scheme. In fact, those factors can influence vortex formation and shedding frequency because of the reflecting conditions of the boundaries and numerical integration errors.

Despite this mismatch, the model still demonstrates to reproduce qualitative and quantitative aspects of vortex shedding. This confirms also the ability of the approach to reproduce unsteady flows and to go beyond laminar ones.

4.3 Poiseuille flow

In order to further numerically assess the goodness of the model for applications to geophysical flows, specifically with regards to the viscous effect, we consider a simulation of a Poiseuille flow with uniform external acceleration as a validating example as in (Bilotta et al., 2022).

In this case, we set the domain Ω of (50×50) cells with $h = 0.05$ m, no-slip boundary conditions at $y = 0$ and $y = 2.5$ m, and periodic boundary conditions on $x = 0$ m and $x = 2.5$ m. This allows us to approximate an infinite length pipe. The main objective of this task is to verify that the simulation can reproduce the analytical terminal velocity profile of a viscous flow along the direction orthogonal to the acceleration which is parabolic according to the following law:

$$u(y) = \frac{1}{2}\rho\frac{g}{\nu}\left(\frac{L^2}{4} - y^2\right) \tag{28}$$

Here $u(y)$ is the flow velocity, ρ is the fluid density, g is the uniform external acceleration, ν the viscosity term and L is the width of the channel.

If we set $\nu = 1$ m/s^2, $\rho = 1$ kg/m^2, then $L = 48 * 0.05 = 2.4$ m and $u(0) = 0.072$ m/s.

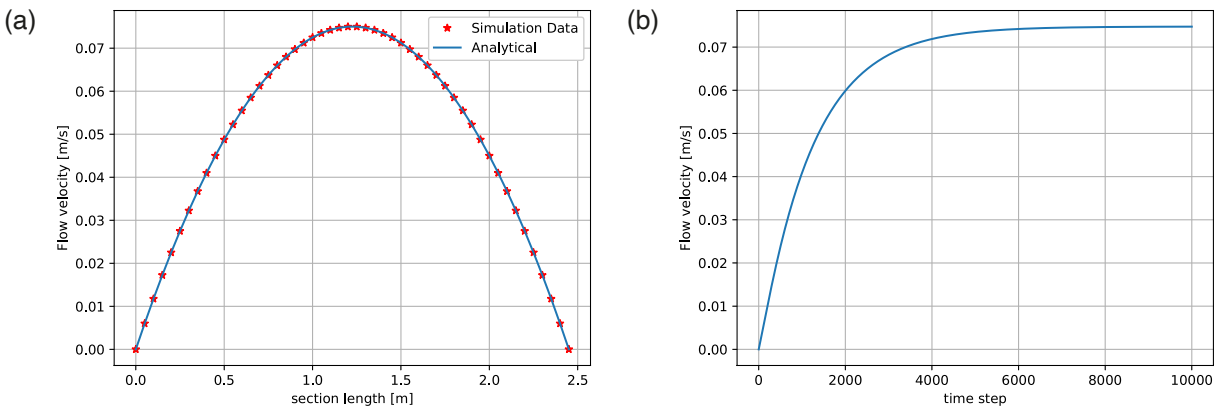

Figure 4. (a) Velocity profile for a section of the flow in the case of analytical results and simulation. (b) Time evolution of the velocity for a single cell set at the middle reaching regime velocity.

The results of the comparison are shown in Fig. 4 and confirm the accuracy of the numerical discretization utilized and the ability of the model to reproduce this physical phenomenon.

4.4 Comparison of the performances using multi-core architecture

To assess the efficiency of the proposed model and the achievable improvements of the architecture brought by new computing architectures, it has been used a modern multicore CPU, with parallel execution managed by the Numba v0.60 (Lam et al., 2015) Python package. This is a just-in-time (JIT) compiler for Python that optimizes computational performance by translating the instructions in efficient machine code. This is well suited for our case, in which we have many parallelizable loops over rows and columns of matrices, allowing to significantly speed-up the execution while keeping the simplicity offered by python environment.

Domain Size [Cells]	Time [s]		MC/SC
	Single-core (SC)	Multi-core (MC)	
30x30	16,74	2,25	7,44
40x40	30,42	3,82	7,96
50x50	49,99	5,99	8,35
60x60	74,00	9,71	7,62
70x70	112,00	13,97	8,02
80x80	149,00	15,46	9,64
90x90	182,00	20,83	8,74
100x100	237,00	33,16	7,15

Table 2. Time execution for different domain sizes with multi core (MC) and single core (SC) approaches. The MC/SC ratio quantifies the gain in performance of the first compared to the latter on an Intel® Core i7 11800H processor.

Table 2 presents the execution times for different domain sizes, comparing single core (SC) performances with multi core (MC) using 16 threads (8 Physical) on an Intel® Core i7 11800H processor. These results are obtained averaging over 1000 simulation timesteps. The ratio of these simulating times, expressed by MC/SC column, is also reported in the table and indicates the speed up in time that is obtained by using parallel execution.

The results indicate a great reduction in computation time when using multi core approach. For the smaller grid sizes, such as 30x30, the speedup is approximately of a factor of 7.44. As the domain size increases the computational efficiency gain remains approximately constant, suggesting that the workload is spread efficiently across the cores of the CPU for different sizes.

In the end this result demonstrates that parallel execution leads to a constant and substantial improvement in computational times with a factor of approximately 8.

5. Conclusions

The results discussed in this paper showed how the implementation of Navier-Stokes equations within the CNN paradigm for the simulation of generic fluids, combined with the hardware parallelization of last-gen hardware, allows to obtain not only a model that is physically convincing but also more computationally efficient. Even though computation time was sensibly reduced, the use of GPUs may further speed up the simulation process. This could be of significant relevance for applications that require solving geophysical flows whose formulation is intrinsically Eulerian. Examples could be the modeling of volcanic ashes clouds and of volcanic plumes, which can be treated as multiphase gaseous flows.

While this work shows the capability of the CNN structure to reproduce classical validation tests, some limitations are acknowledged. First, we assumed an incompressible single-phase flow with constant density and viscosity. This is a big simplification, given that geophysical flows are, by their nature, multi-phase and can exhibit compressible characteristics (e.g. volcanic plumes). Another limitation lies in the absence of a layer to model the thermal characteristic of the flow. Interaction with the environment is the main cause for viscosity changes, which is a key factor for lava modeling. Lastly, the simulations were carried in domains that are bidimensional. While this approach simplifies the implementation of the model and allows it to capture the main features of the flow, it also limits the ability to reproduce complexity of real topographies.

Future work aims at considering variable density, multiphase flows and thermal effects to improve the physical accuracy of the simulations. Adapting this approach to simulate lava rheology may offer valuable insights for volcanic hazard assessment and urban planning.

In brief, the proposed model is capable of reproducing, with a good degree of fidelity, fluid dynamic phenomena like the lid-driven cavity phenomenon or the vortex shedding, while improving computational efficiency by exploiting parallel multicore computing.

Acknowledgements. Giuseppe La Spina acknowledges the PhD programme in Sustainable Development And Climate Change at the University School for Advanced Studies IUSS Pavia, Cycle XXXVIII, and the support of the Italian Ministry of University and Research, DM. 351 – April 9th, 2022, and of the European Union – PNRR NextGenerationEU – Mission 4 "Education and Research", Component 1 "Enhancement of the offer of educational services: from nurseries to universities" – Investment 4.1 "Extension of the number of research doctorates and innovative doctorates for public administration and cultural heritage". Arturo Buscarino acknowledges the partial support by European Union (NextGeneration EU), through the MUR-PNRR project "FAIR" (E63C22001940006).
This work was developed within the framework of the Laboratory of Technologies for Volcanology (TechnoLab) at the Istituto Nazionale di Geofisica e Vulcanologia (INGV) in Catania (Italy).

References

Agarwal, R., A. Domoshnitsky, A. Slavova and V. Ignatov (2024). Edge of Chaos in Integro-Differential Model of Nerve Conduction, Mathematics, 12, 13, 2046, doi:10.3390/math12132046.

Arena, P., M. Bucolo, S. Fazzino, L. Fortuna et al. (2005). The CNN Paradigm: Shapes and complexity, Int. J. Bifurc. Chaos, 15, 7, 2063-2090, doi:10.1142/S0218127405013307.

Barrett, R., M. Berry, T. F. Chan, J. Demmel et al. (1994). Templates for the Solution of Linear Systems: Building Blocks for Iterative Methods, Society for Industrial and Applied Mathematics, SIAM.

Billah, K. Y. and R. H. Scanlan (1991). Resonance, Tacoma Narrows bridge failure, and undergraduate physics textbooks, Am. J. Phys., 59, 2, 118-124, doi:10.1119/1.16590.

Bilotta, G., V. Zago, V. Centorrino, R. A. Dalrymple et al. (2022). A numerically robust, parallel-friendly variant of BiCGSTAB for the semi-implicit integration of the viscous term in Smoothed Particle Hydrodynamics, J. Comput. Phys., 466, 111413, doi:10.1016/j.jcp.2022.111413.

Buscarino, A., L. Fortuna and M. Frasca (2017). Essentials of nonlinear circuit dynamics with MATLAB and laboratory experiments, Boca Raton, London, New York, CRC Press, Taylor & Francis Group, doi:10.1201/b22063.

Chorin, A. J. (1968). Numerical Solution of the Navier-Stokes Equations. Numerical Solution of the Navier-Stokes Equations, Math. Comput., 745-762, doi:10.2307/2004575.

Chua, L. O. and L. Yang (1988a). Cellular neural networks: applications, IEEE Trans. Circuits Syst., 35, 10, 1273-1290, doi:10.1109/31.7601.

Chua, L. O. and L. Yang (1988b). Cellular neural networks: theory, IEEE Trans. Circuits Syst., 35, 10, 1257-1272, doi:10.1109/31.7600.

Del Negro, C., L. Fortuna and A. Vicari (2005). Modelling lava flows by Cellular Nonlinear Networks (CNN): preliminary results, Nonlinear Process. Geophys., 12, 4, 505-513, doi:10.5194/npg-12-505-2005.

Fortuna, L., A. Rizzo and M. G. Xibilia (2003). Modeling Complex Dynamics via Extended PWL-Based CNNs, Int. J. Bifurc. Chaos, 13, 11, 3273-3286, doi:10.1142/S0218127403008727.

Ghia, U., K. N. Ghia and C. T. Shin (1982). High-Re solutions for incompressible flow using the Navier-Stokes equations and a multigrid method, J. Comput. Phys., 48, 3, 387-411, doi:10.1016/0021-9991(82)90058-4.

Hérault, A., G. Bilotta and R. A. Dalrymple (2010). SPH on GPU with CUDA, J. Hydraul. Res., 48, 1, 74-79, doi:10.1080/00221686.2010.9641247.

Hou, S., Q. Zou, S. Chen, G. Doolen et al. (1995). Simulation of Cavity Flow by the Lattice Boltzmann Method, J. Comput. Phys., 118, 2, 329-347, doi:10.1006/jcph.1995.1103.

Kozek, T., L. O. Chua, T. Roska, D. Wolf et al. (1995). Simulating nonlinear waves and partial differential equations via CNN. II. Typical examples, IEEE Trans. Circuits Syst., I, Fundam. Theory. Appl., 42, 10, 816-820, doi:10.1109/81.473591.

Lam, S. K., A. Pitrou and S. Seibert (2015). Numba: a LLVM-based Python JIT compiler. In Proceedings of the Second Workshop on the LLVM Compiler Infrastructure in HPC, Austin Texas, ACM, 1-6, doi:10.1145/2833157.2833162.

LeVeque, R. J. (2007). Finite Difference Methods for Ordinary and Partial Differential Equations: Steady-State and Time-Dependent Problems, Society for Industrial and Applied Mathematics, 341, doi:10.1137/1.9780898717839.

Ma, N., J. Song, Z. Zhang and Y. Yu (2024). Bursting dynamics in a state controlled cellular neural network based MLC circuit with periodic forcing signals, Commun. Nonlinear Sci. Numer. Simul., 138, 108203, doi:10.1016/j.cnsns.2024.108203.

Miyamoto, H. and S. Sasaki (1997). Simulating lava flows by an improved cellular automata method, Comput. Geosci., 23, 3, 283-292, doi:10.1016/S0098-3004(96)00089-1.

Peng, Y. F., Y. H. Shiau and R. R. Hwang (2003). Transition in a 2-D lid-driven cavity flow, Comput. Fluids, 32, 3, 337-352, doi:10.1016/S0045-7930(01)00053-6.

Rongo, R., V. Lupiano, W. Spataro, D. D'Ambrosio et al. (2016). SCIARA: cellular automata lava flow modelling and applications in hazard prediction and mitigation, Geol. Soc. Spec. Publ., 426, 1, 345-356, doi:10.1144/SP426.22.

Roska, T., L. O. Chua, D. Wolf, T. Kozek et al. (1995). Simulating nonlinear waves and partial differential equations via CNN. I. Basic techniques, IEEE Trans. Circuits Syst., I, Fundam. Theory Appl., 42, 10, 807-815, doi:10.1109/81.473590.

Szolgay, P., G. Voros and G. Eross (1993). On the applications of the cellular neural network paradigm in mechanical vibrating systems, IEEE Trans. Circuits Syst., I, Fundam. Theory Appl., 40, 3, 222-227, doi:10.1109/81.222805.

Wang, Y. E., G. Y. Wei and D. Brooks (2019). Benchmarking TPU, GPU, and CPU Platforms for Deep Learning, arXiv, doi:10.48550/arXiv.1907.10701.

Wu, H., J. Gu, Y. Guo, M. Chen et al. (2024). Biphasic action potentials in an individual cellular neural network cell, Chaos, Soliton. Fract., 182, 114792, doi:10.1016/j.chaos.2024.114792.

Zago, V., G. Bilotta, A. Cappello, R. Dalrymple et al. (2017). Simulating Complex Fluids with Smoothed Particle Hydrodynamics, Ann. Geophys., 60, 6, sup., doi:10.4401/ag-7362.

Zago, V., G. Bilotta, A. Hérault, R. A. Dalrymple et al. (2018). Semi-implicit 3D SPH on GPU for lava flows, J. Comput. Phys, 375, 854-870, doi:10.1016/j.jcp.2018.07.060.

Zago, V., R. A. Dalrymple, N. Almashan, G. Bilotta et al. (2023). Characterization and modeling of greenwater overtopping of a sea-level deck, Ocean Eng., 275, 114131, doi:10.1016/j.oceaneng.2023.114131.

***CORRESPONDING AUTHOR: Giuseppe LA SPINA**,
Scuola Universitaria Superiore IUSS, Pavia, Italy
e-mail: giuseppe.laspina@iusspavia.it