Machine learning acceleration for nonlinear solvers applied to multiphase porous media flow

V. L. S. Silva[a,b,c,*], P. Salinas[a,b], M. D. Jackson[b], C. C. Pain[a,b]

[a] *Applied Modelling and Computation Group, Department of Earth Science and Engineering, Imperial College London, UK*
[b] *Novel Reservoir Modelling and Simulation Group, Department of Earth Science and Engineering, Imperial College London, UK*
[c] *Petrobras, Brazil*

Abstract

A machine learning approach to accelerate convergence of the nonlinear solver in multiphase flow problems is presented here. The approach dynamically controls an acceleration method based on numerical relaxation. It is demonstrated in a Picard iterative solver but is applicable to other types of nonlinear solvers. The aim of the machine learning acceleration is to reduce the computational cost of the nonlinear solver by adjusting to the complexity/physics of the system. Using dimensionless parameters to train and control the machine learning enables the use of a simple two-dimensional layered reservoir for training, while also exploring a wide range of the parameter space. Hence, the training process is simplified and it does not need to be rerun when the machine learning acceleration is applied to other reservoir models. We show that the method can significantly reduce the number of nonlinear iterations without compromising the simulation results, including models that are considerably more complex than the training case.

Keywords: Nonlinear solver, machine learning, numerical relaxation, multiphase flows, porous media

*Corresponding author.
Email address: viluiz@gmail.com (V. L. S. Silva)

1. Introduction

Modeling multiphase flow in porous media is of paramount importance to understand, predict and manage subsurface reservoirs with applications to hydrocarbon recovery, geothermal energy resources, CO2 geological sequestration, groundwater sources and magma reservoirs. Mathematical models of such reservoirs consist of partial differential equations, that for most problems of practical interest cannot be solved analytically. To this end, it is necessary to use numerical methods to solve the governing partial differential equations (Aziz, 1979; Jackson et al., 2015). However, the numerical solution of the governing equations is very challenging due to the nonlinear nature of the problem and the strong coupling between the different equations (Li and Tchelepi, 2015). These nonlinearities in the multiphase flow equations can be divided in two groups. Weak nonlinearities appear in the variables that are function of pressure, such as density and viscosity; strong nonlinearities are related with variables that depend on saturation, such as relative permeability and capillary pressure (Aziz, 1979; Ertekin et al., 2001).

In order to solve the discretized equations, nonlinear solvers based on Newton methods in a fully implicit formulation have been traditionally used. However, sequential methods have been gaining ground in recent years (Salinas et al., 2017a; Jiang and Tchelepi, 2019; Freitas et al., 2020). These type of methods are attractive because the multiphysics problem can be subdivided and each subproblem solved separately, which gives wide flexibility and extensibility. For example, each subproblem can be solved in an efficient manner through specialized solvers. The Picard iterative method can work as a sequential approach, that can also be interpreted as a sequential fixed-point iteration or a nonlinear block Gauss-Seidel process. This kind of method requires fewer conditions to achieve convergence than Newton methods, and the discretized equations can be directly solved which simplifies the implementation (Elman et al., 2005; Lott et al., 2012). Newton methods are also used in sequential approaches both in the inner and/or outer loops, as they have quadratic convergence and in general are faster than Picard iterative processes (Ortega and Rheinboldt, 1970; Jenny et al., 2006; Wong et al., 2019).

Jenny et al. (2006) proposed a sequential-implicit fixed point multi-scale strategy for isothermal multiphase flow in porous media where in each fixed point iteration (outer loop) a sequence of subproblems are solved using the Newton method (inner loop). Salinas et al. (2017a) proposed a different ap-

proach, in which the Picard iterative method is also used in the inner loop along with different methods to improve the convergence behavior. A drawback of sequential methods is that they often suffer from slow convergence when there is strong coupling between the equations (Wong et al., 2019; Jiang and Tchelepi, 2019).

Rapid convergence of the nonlinear solver is of vital importance as it strongly affects the overall computational time. Therefore, a great deal of effort has been put on obtaining robust and stable convergence rates in sequential methods. Salinas et al. (2017a) proposed an acceleration technique based on numerical relaxation that is applied to the saturation field for the multiphase flow in porous media problem. Jiang and Tchelepi (2019) applied three nonlinear acceleration techniques to the sequential-implicit fixed point method for improving the outer-loop convergence. Among the numerical relaxation, quasi-Newton and Anderson acceleration, numerical relaxation achieved the best overall performance. Salinas et al. (2020) presented a mass conservative vanishing artificial diffusion that accelerates the nonlinear solver. The proposed artificial diffusion vanishes when convergence is achieved and targets sharp displacement (shock) fronts that arise in many problems when one fluid phase displaces another.

The success of machine learning in different fields has inspired recent applications with the aim of reducing the computational cost in numerical simulations by accelerating the convergence of the numerical solution. Oladokun et al. (2020) used a random forest regression to determine the linear convergence tolerance of the nonlinear solver. Random forests is an ensemble method that fits a number of decision trees in sub-samples of the dataset and combine/average their result to improve predictive accuracy and control over-fitting (Breiman, 2001). The method proposed by Oladokun et al. (2020) reduced the number of linear iterations while not compromising the number of nonlinear iterations.

In this work, a machine learning acceleration to dynamically control the numerical relaxation is evaluated. Two approaches are proposed and applied to a Picard iterative solver in a multiphase porous media flow. The proposed acceleration method can be applicable to other types of nonlinear solvers that uses numerical relaxation. For the machine learning training, data from a multiphase flow simulation in a simple two-dimensional layered reservoir is used, which considerably simplifies this step. This approach is possible because the training process is performed based on the reservoir parameters scaled to a dimensionless space, which can represent many of the main flow

features observed in models of natural reservoirs. The aim of the machine learning acceleration is to reduce the computational cost of the nonlinear solver by adjusting itself to the complexity/physics of the system.

The remaining of this paper is structured as follows. In Section 2, the governing equations of multiphase flow in porous media are described. Following that, Section 3 details the nonlinear Picard iterative solver used here. Subsequently, the proposed machine learning acceleration is described in Section 4, and then in Section 5 this method is applied to four test cases. The results are discussed in Section 6, and finally, concluding remarks are presented in Section 7.

2. Governing equations

The formulation for incompressible flow with gravity and capillary pressure is reported here. The mass-balance equation for phase α of immiscible-fluid phases is

$$\phi \frac{\partial S_\alpha}{\partial t} + \nabla \cdot \mathbf{q}_\alpha = s_{cty,\alpha}, \tag{1}$$

where t represents time and ϕ is the rock porosity. S_α is the saturation of phase α, $\mathbf{q}_\alpha$ is the Darcy velocity, and $s_{cty,\alpha}$ is a source term.

The multiphase Darcy's law for phase α is given by

$$\mathbf{q}_\alpha = \frac{k_{r\alpha}\mathbf{K}}{\mu_\alpha}(-\nabla p_\alpha + \rho_\alpha g \nabla z), \tag{2}$$

where p is the pressure, ρ is the density, g is the gravitational acceleration, and ∇z is the gravity direction. $k_{r\alpha}$, $\mathbf{K}$, and μ_α are the relative permeability, permeability tensor, and viscosity, respectively. The Darcy law can also be written in a slightly modified form as

$$\underline{\underline{\sigma_\alpha}}\mathbf{u}_\alpha = -\nabla p_\alpha + \rho_\alpha g \nabla z, \tag{3}$$

with

$$\underline{\underline{\sigma_\alpha}} = \mu_\alpha S_\alpha (k_{r\alpha}\mathbf{K})^{-1}, \tag{4}$$

and

$$\mathbf{u}_\alpha = \frac{\mathbf{q}_\alpha}{S_\alpha}, \tag{5}$$

where $\mathbf{u}_\alpha$ is the saturation-weighted Darcy velocity. The relative permeability is a power-law function of saturation. Hence, writing the Darcy law as Eq.

(3) helps to reduce the nonlinearity of the equation (Jackson et al., 2015; Gomes et al., 2017).

Considering a wetting and non-wetting phase and including capillary pressure,

$$p_c = p_{nw} - p_w, \tag{6}$$

where p_w and p_{nw} are the pressures of the wetting and non-wetting phases, respectively. p_c is the capillary pressure. To close the system of equations, the saturation constraint is given by

$$S_w + S_{nw} = 1, \tag{7}$$

where S_w and S_{nw} are the saturations of the wetting and non-wetting phases, respectively.

Here, the time is discretized using a θ-method where θ smoothly varies between 0.5 (Crank-Nicolson) and 1 (implicit Euler) based on total variation diminishing (TVD) criterion (Pavlidis et al., 2014). The spatial discretization is based on the Double Control Volume Finite Element method, described in Salinas et al. (2017b), which is an improvement for highly distorted meshes of the Control Volume Finite Element method (Fung et al., 1992; Durlofsky, 1993; Jackson et al., 2015; Gomes et al., 2017). The velocity is discretized using finite elements, but pressure and saturation are discretized using control volumes. The Imperial College Finite Element Reservoir SimulaTor (IC-FERST) is used here for the solutions of the reported equations (Gomes et al., 2017; Salinas et al., 2017a,b).

3. Nonlinear solver

A Picard iterative method is used to solve the discretized form of the nonlinear system of equations formed by Eqs. (1), (3), (5), (6) and (7). The overall method is described in Figure 1. The solver comprises three main loops: the solid line (C) represents the time loop; the dotted line (B) denotes the coupling between the saturation and pressure (nonlinear outer loop); and the dashed line (A) iterates over saturation and velocity (nonlinear inner loop). In the latter loop after calculating the saturation, $\underline{\underline{\sigma_\alpha}}$ is recalculated and the velocity is updated. A new saturation is calculated using the new velocity. The process continues until it reaches convergence or the maximum number of iterations. After that, having the new saturation, a new pressure is estimated in the outer loop. Then a new velocity is calculated from the

pressure estimation. A new inner loop iteration starts. Until convergence or the maximum number of iterations is reached, the process repeats.

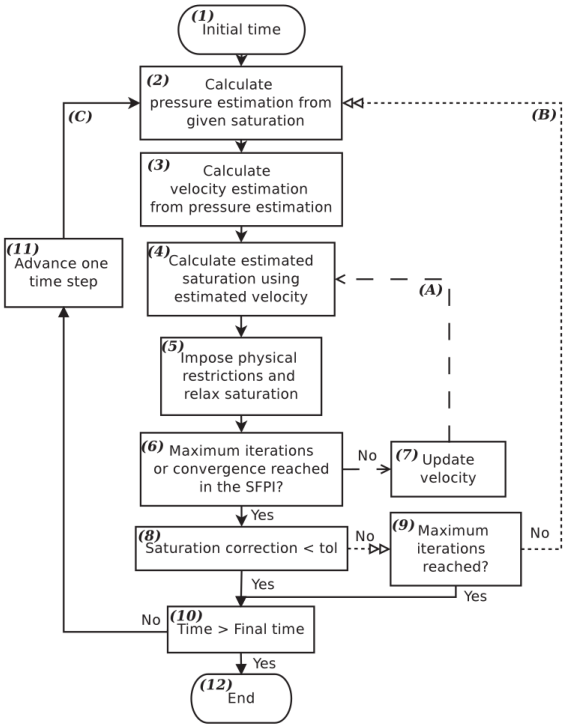


Figure 1: Flow chart of the nonlinear solver iterations. Reprinted from (Salinas et al., 2017a).

A relaxation parameter is used here in order to accelerate the convergence of the nonlinear solver. The new saturation is updated by weighting the new calculated field with the saturations obtained in the previous iterations of the inner loop. The new saturation is calculated as

$$S^k = \omega^k \tilde{S}^k + (1-\omega^k)S^{k-1} + (1-\omega^k)^{\beta+1}\omega^k(S^{k-2} - S^{k-1}), \tag{8}$$

where k is the index representing the current inner-loop iteration, S^k is the new saturation after relaxation, and $\tilde{S}^k$ is the saturation obtained after (4) in Figure 1. ω is the relaxation parameter, and β is the exponent that controls the relative importance of S^{k-1} and S^{k-2}. In this work, β is considered constant with the value of 0.4 as in Salinas et al. (2017a).

Having the initial value of w^0 the subsequent relaxation parameters (w^k) are calculated to yield the best convergence ratio reducing the residual of the saturation. A second order curve is fitted to the last three values of the residual ratio of convergence and their corresponding relaxation factor. After that, an optimal is calculated and assigned to the next w^k. If a minimum cannot be obtained or the last three values are not available, the algorithm calculates w^k based on the last value and its respective convergence ratio (Salinas et al., 2017a).

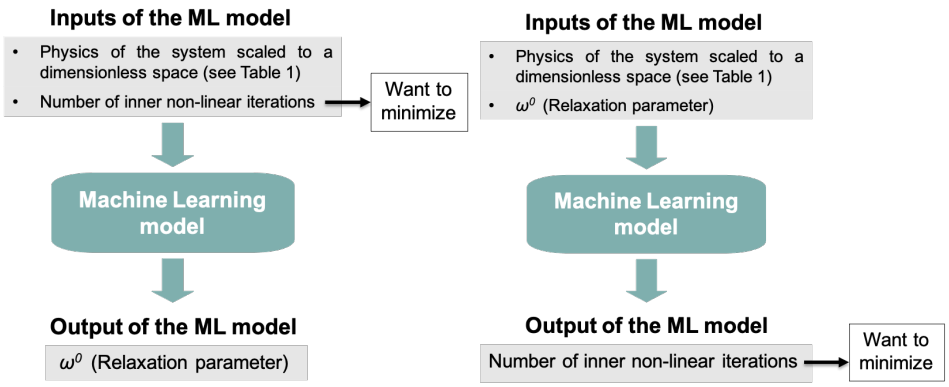
How w^0 is chosen is a significant topic. A naive method could choose a static value for all outer iterations. Nevertheless, the relaxation parameters need to be small enough to avoid divergence and as large as possible to accelerate convergence. In the convergence of nonlinear loops, the value of w^0 plays a significant role, but is not known *a-priori* and is problem-specific. In addition, even the optimal static value can result in more iterations than an effective dynamic relaxation parameter (Küttler and Wall, 2008; Jiang and Tchelepi, 2019).

To this end, a machine learning algorithm is proposed to produce the values of w^0 based on the physics of the system scaled to a dimensionless space, at the beginning of each outer nonlinear iteration. The primary objective is to minimize the computational cost by decreasing the number of nonlinear iterations.

4. Machine learning acceleration

Two new approaches of machine learning to control the relaxation parameter in each outer nonlinear iteration are tested here. In the first approach, the machine learning model inputs are dimensionless numbers, simulation properties, and the number of inner nonlinear iterations at each outer nonlinear iteration. The output of the model is the value of the relaxation parameter, as shown in Figure 2a. Thus, after training the model, the number of inner nonlinear iterations can be set to a low value, such as one, and the corresponding relaxation factor can be determined. In the second approach the machine learning model inputs are dimensionless numbers, simulation properties, and the relaxation parameter. The output of the model is the number of inner nonlinear iterations, as shown in Figure 2b. For this case, after training the model the relaxation factor that produces the minimum number of nonlinear iterations can be found. It is worth mentioning that in the approach presented in Figure 2b one more step is needed after running

the machine learning model. This step is a single variable optimization to find the relaxation parameter that minimizes the number of inner nonlinear iterations.



(a) Approach 1. (b) Approach 2.

Figure 2: Two new approaches to control the value of the relaxation parameter in each outer nonlinear iteration.

4.1. Input parameters and dimensionless numbers

The spatial distribution of fluids during multiphase flow in porous media is controlled by the combination of the viscous, capillary and gravitational forces that drive the fluid flow (Hoteit and Firoozabadi, 2008; Li and Tchelepi, 2015; Debbabi et al., 2017a,b, 2018b). Furthermore, the relative importance of these mechanisms depends on the combination of the flow rate, the system length scales and fluid rock properties (Debbabi et al., 2018b). Therefore, the inputs of the machine learning model are the dimensionless numbers presented on Debbabi et al. (2017a,b, 2018a). Using dimensionless parameters to train and control the machine learning acceleration enables us to use a simple two-dimensional layered reservoir for training, while also exploring a wide range of the parameter space. The following parameters were also added to the inputs: the total mobility, the Darcy velocity, the Courant–Friedrichs–Lewy (CFL) number, the CFL number at the shock front, the value of the vanishing artificial diffusion (as in Salinas et al. (2020)), the two last transport equation residuals, and their ratio. The percentage of elements in the shock front is also used as an input. It is calculated as the number of elements in the shock front divided by the total number of

elements. If there is no element in the shock front, such as in a smooth variation of saturation, the percentage of elements in the shock front is zero.

Table 1 summarizes all the input parameters. L is the domain length, H is domain thickness, $\overline{k_h}$ and $\overline{k_v}$ are the average horizontal and vertical permeabilities, Δt is the time step size, Δl is the characteristic length of one element, $\lambda_T(s_f)$ is the total mobility just behind the shock front and $\lambda_T(s_\infty)$ is the total mobility in front of the shock front. k^e_{rd} is the end-point relative permeability of the displaced phase, μ_d is the viscosity of the displaced phase, P^e_c is the entry capillary pressure, θ is the angle between the flow main direction and the horizontal plane, and $\Delta\rho$ is the density contrast between phases.

4.2. Training process

The machine learning model is trained using results from a multiphase flow simulation in a simple two-dimensional layered model, as shown in Figure 3. Φ, k_h and k_v are the porosity, horizontal and vertical permeabilities, respectively. As the reservoir model needs to be run several times to create the dataset for training, this approach considerably simplifies the training process. Furthermore, the training step does not need to be rerun when the machine learning acceleration is applied to other reservoir models.

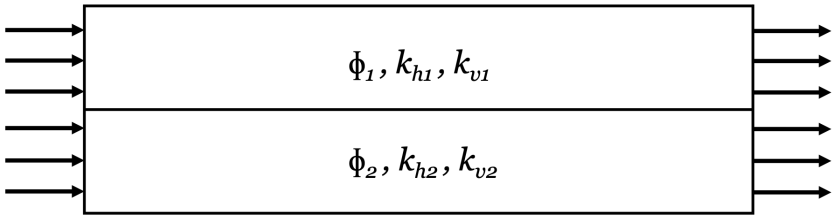


Figure 3: Two dimensional homogeneous layered reservoir used for training.

In order to perturb the physical parameters presented in Table 1 for generating the training set, 6500 simulations were performed changing the following simulations inputs: porosities, horizontal and vertical permeabilities, time step, relative permeability, capillary pressure, viscosity, gravity magnitude and direction, and the relaxation factor. The maximum number of inner nonlinear iterations was set to 50 in all simulations. Table 2 show the resulting range of the physical parameters in the training set. It can be seen from the values of the variables in Table 2, that a wide range of the

Table 1: Input parameters for the machine learning model.

Parameter	**Expression**	**Comments**
Effective aspect ratio	$R = \frac{L}{H}\sqrt{\frac{\overline{k_v}}{\overline{k_h}}}$	
Darcy velocity	$q = \sum_\alpha \|\|\mathbf{q}_\alpha\|\|_2$	Calculated control-volume wise. Used as average, max and min
Total mobility	$\lambda_T = \sum_\alpha \frac{k_{r\alpha}}{\mu_\alpha}$	Calculated control-volume wise. Used as average, max and min
CFL number	$CFL = \frac{q\Delta t}{\Delta l}$	Maximum value of all elements.
Shock-front CFL number	$CFL_f = \frac{q\Delta t}{\Delta l}$	Maximum value of all elements in the shock front
Shock-front number ratio	-	Percentage of elements in the shock front
Shock-front mobility ratio	$M_f = \frac{\lambda_T(s_f)}{\lambda_T(s_\infty)}$	Calculated control-volume wise. Used as average, max and min
Longitudinal capillary	$N_{cv,L} = \frac{\overline{k_h k_{rd}^e} P_c^e}{q\mu_d L}$	Calculated control-volume wise. Used as average, max and min
Transverse capillary	$N_{cv,T} = \frac{\overline{k_v k_{rd}^e} P_c^e L}{q\mu_d H^2}$	Calculated control-volume wise. Used as average, max and min
Buoyancy number	$N_{bv} = \frac{\overline{k_h k_{rd}^e}\Delta\rho g\cos(\theta) H}{q\mu_d L}$	Calculated control-volume wise. Used as average, max and min
Longitudinal buoyancy	$N_{bv,L} = \frac{\overline{k_h k_{rd}^e}\Delta\rho g\sin(\theta)}{q\mu_d}$	Calculated control-volume wise. Used as average, max and min
Transverse buoyancy	$N_{bv,T} = \frac{\overline{k_v k_{rd}^e}\Delta\rho g\cos(\theta) L}{q\mu_d H}$	Calculated control-volume wise. Used as average, max and min
Vanishing artificial diffusion	-	Calculated as in Salinas et al. (2020). Used as average, max and min
Transport equation residual	-	Two last residuals and their ratio.
Number of inner nonlinear iterations	-	Only for approach 1
Relaxation parameter	-	Only for approach 2

parameter space is explored, although using a simple two-layered reservoir for training.

Table 2: Range of the physical parameters in the training set.

Parameter	**Min**	**Average**	**Max**
Effective aspect ratio	1.58	4.61	7.07
Darcy velocity (m.s^{-1})	1.32×10^{-12}	3.21×10^{-6}	2.79
Total mobility (Pa^{-1}.s^{-1})	2.50	7.13×10^{2}	1.00×10^{4}
CFL number	1.78×10^{-3}	4.15×10^{1}	1.78×10^{6}
Shock-front CFL number	0.00	1.34×10^{1}	8.08×10^{5}
Shock-front number ratio	0.00	1.87×10^{-2}	8.57×10^{-1}
Shock-front mobility ratio	9.99×10^{-4}	1.01	1.00×10^{3}
Longitudinal capillary	0.00	3.13×10^{-3}	2.87×10^{2}
Transverse capillary	0.00	8.24×10^{-2}	9.02×10^{3}
Buoyancy number	0.00	9.50×10^{-2}	4.32×10^{3}
Longitudinal buoyancy	0.00	4.55×10^{-1}	2.46×10^{4}
Transverse buoyancy	0.00	2.60	1.49×10^{5}

4.3. Machine learning model

Different machine learning models were tested and the best performance was achieved by using random forest regression. Random forests, introduced by Breiman (2001), are a classification and regression technique that combine decision tree predictions and are a widely used algorithm. In the random forests algorithm, each tree in the ensemble is trained based on samples drawn from the training set with or without replacement. Moreover, when splitting each node during the construction of the tree, the best split is found in a random subset of the inputs. These two random aspects were added in order to diminish the variance of the forest estimator (Breiman et al., 1998; Breiman, 2001). In a broader view, each decision tree is constructed by randomly selecting a subset of features and training examples. Thus, the final result is taken by averaging the values generated in each tree, as shown in Figure 4.

After training, each instance in Figure 4 represents a vector containing all the input parameters in Table 1. Thus, a subset of the parameters in this vector goes to each decision tree, depending on the best split found during the training process. The outcome of each tree is averaged to produce the final result. For the approach 1 the result is the relaxation parameter and for the approach 2 the result is the number of inner nonlinear iterations.

The implementation of the random forest used in this work is the one in Pedregosa et al. (2011). The hyperparameters were optimized using a

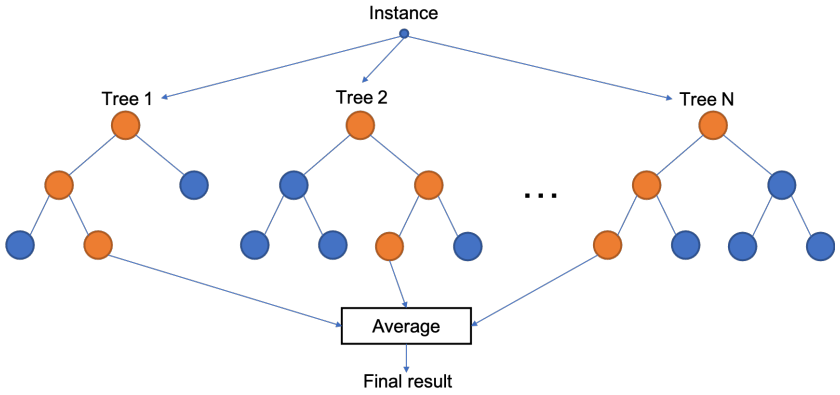


Figure 4: Random forest regressor. Each instance represents the input parameters in Table 1, and the final result is the relaxation parameter for the approach 1 or the number of inner nonlinear iterations for the approaches 2.

grid search and the values that resulted in the lowest error for the validation data are presented in Table 3. The loss function is the mean square error. Figure 5 shows the learning curve of the machine learning model for approach 1. The learning curve shows the validation and training errors for different numbers of training instances and can be used to quantify the benefit of adding more data to the training set. In our study, 6500 simulations, that represents nearly 1.8×10^6 instances (outer nonlinear iterations), were used for the training (80%) and validation data (20%). Figure 5 shows that the rate of error reduction once we reach this number of instances is small, so continuing to add further training simulations would not result in significant error reduction. Hence our choice of dataset size. The learning curve for approach 2 showed comparable results, so is not presented here.

Table 3: Random forest hyperparameters.

Parameter	**Approach 1**	**Approach 2**
Bootstrap	False	False
The number of trees in the forest	50	50
The number of features to consider for splitting	10	5
The maximum depth of the tree	30	30
The minimum number of samples to be a leaf node	2	2

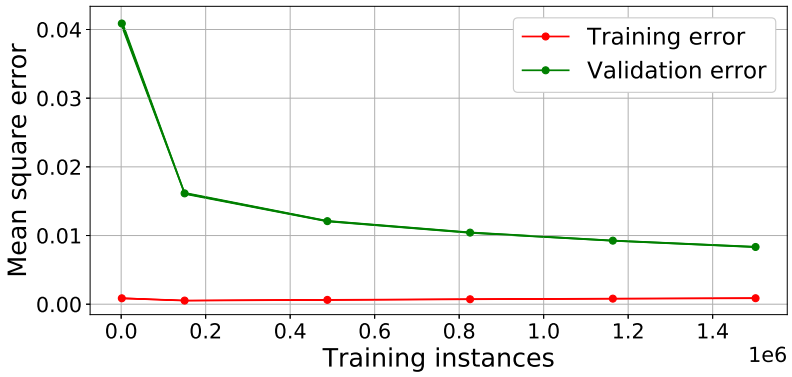


Figure 5: Learning curve for the approach 1. The vertical axis is the error of the machine learning model, and the horizontal axis represents the size of the dataset used for training.

4.4. Comparing the two approaches

When comparing the two approaches, the first one outperforms the second in all tests. Moreover, the approach 2 needs to perform an additional single variable optimization to determine the relaxation factor. Therefore, the approach 1 was selected to continue the experiments, and for the remaining of this paper it will be also referred as the machine learning acceleration.

5. Numerical experiments

The machine learning acceleration is tested in four reservoir cases. All test cases represent two phase immiscible flow in a porous medium. The first is the two-dimensional layered reservoir used for training. The second is a two-dimensional heterogeneous reservoir with permeability varying in four quadrants. The third is a more realistic three-dimensional channelized model, and the fourth is a three-dimensional faulted reservoir. All were tested with and without gravity and capillary pressure. A Brooks-Corey model (Brooks and Corey, 1964) is used for the relative permeability and capillary pressure.

The convergence criteria used for the nonlinear solver is that the relative mass conservation of the system within a time step has to be below 10^{-3} and the infinite norm of the saturation difference between two consecutive nonlinear iterations has to be below 10^{-2}. The maximum number of outer and inner nonlinear iterations is set to 30 and 10, respectively. When comparing different cases, the number of nonlinear iterations is calculated as the number

of outer nonlinear iterations plus the number of inner nonlinear iterations divided by 3, as in Salinas et al. (2017a). As mentioned before, the approach 1 was selected to continue the experiments and it is compared with the default dynamic relaxation presented in IC-FERST (Salinas et al., 2017a), with fixed relaxations and with no relaxation. The default dynamic acceleration in IC-FERST is based mainly on the CFL number.

5.1. Test case 1

Test case 1 is the same reservoir model used to train the random forest. It is a two-dimensional model with two layers (Figure 6). The porosity in the top layer is 10% and the horizontal permeability is 200 mD. The bottom layer has 20% of porosity and 100 mD of horizontal permeability. The vertical permeability is equal to 10% of the horizontal. We inject water on the left and produce oil and water on the right by imposing a fixed pressure on both boundaries. The second column of Table 4 shows the rock-fluid properties of this case.

Table 4: Rock-fluid properties of the four test cases.

Property	**Cases 1 and 2**	**Case 3**	**Case 4**
Viscosity ratio	5	5	5
Density contrast	300 kg/m^3	289 kg/m^3	300 kg/m^3
Entry capillary pressure	1000 Pa	100 Pa	10000 Pa
Capillary exponent	1.0	1.0	1.0
Relative permeability end point of water	1.0	0.3	0.3
Relative permeability end point of oil	1.0	0.8	0.8
Relative permeability exponent	2.0	2.0	2.0
Immobile fraction of water	20 %	20 %	20 %
Immobile fraction of oil	30 %	20 %	20 %

Figure 6 shows the permeability and the saturation profile after 62 days along with the simulation mesh. The total simulation time is 100 days and the time step is set to 1 day. The breakthrough of water - i.e., the time at which injected water reaches the opposite model boundary - is within this time scale. The machine learning acceleration is compared with the default dynamic acceleration in IC-FERST, and with fixed relaxations. Figure 7 shows the cumulative number of nonlinear iterations over the simulation time for all cases. The machine learning acceleration performs better than

the other relaxations practically over all time steps. The values of the relaxation parameter generated by the machine learning acceleration, for the case without gravity and capillary pressure, are presented in Figure 8. At around 300 outer nonlinear iterations, a major change occurs in the value of the relaxation factor with increasing non linear iterations. This change occurs just after the breakthrough of water (around 65 days or 275 outer iterations) when the displacement front - the sharp change in saturation at the leading edge of the displacement and the most challenging solution feature to resolve - has passed out of the domain. Solution of the nonlinear system of equations is therefore easier so the machine learning acceleration increases the value of the relaxation.

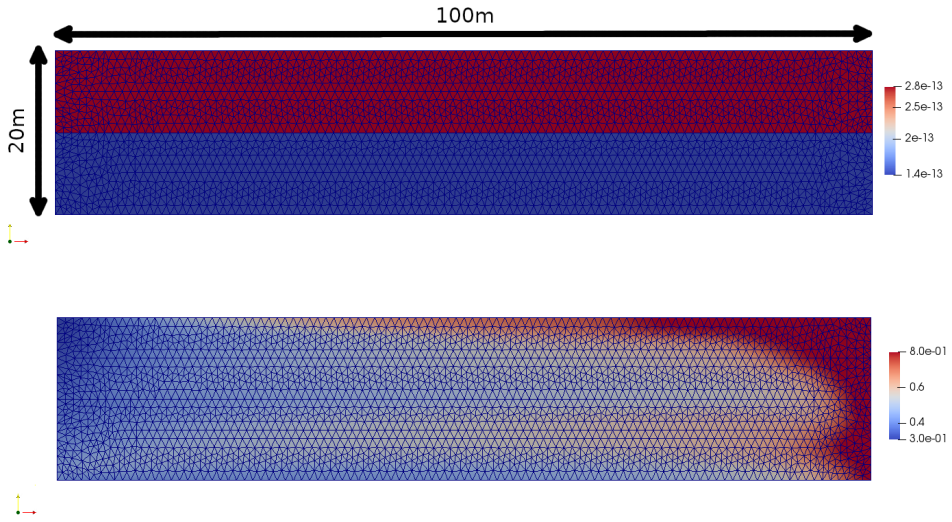
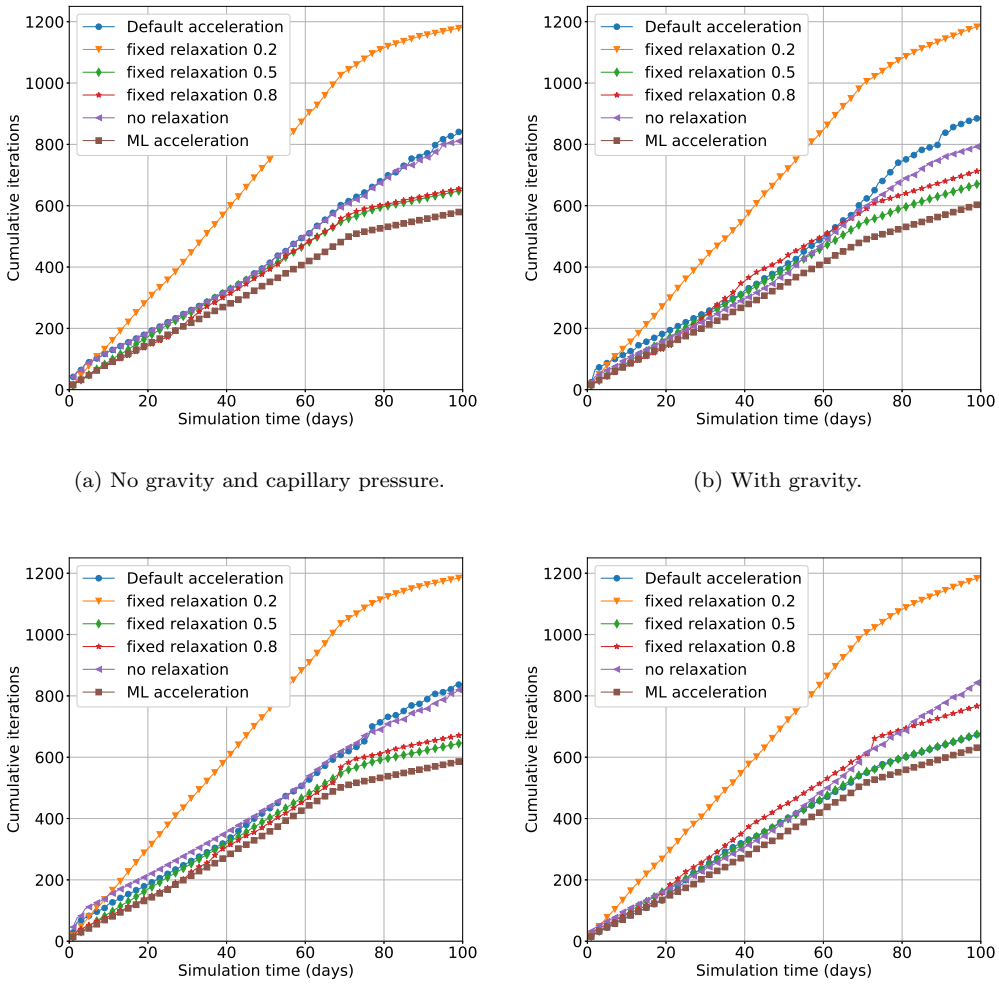


Figure 6: Permeability field in m^2 (top) and oil saturation after 62 days (bottom).

5.2. Test case 2

Test case 2 is a two-dimensional reservoir with four domains arranged as quadrants (Figure 9). The permeabilities in the top layer are 200 mD and 20 mD, whereas the permeabilities in the bottom layer are 10 mD and 100 mD. The vertical permeability is equal to the horizontal and the porosity is 20% in all layers. We inject water on the left and produce oil and water on the right by imposing a fixed pressure on both boundaries. The second column of Table 4 shows the rock-fluid properties of this case.



(a) No gravity and capillary pressure.

(b) With gravity.

(c) With capillary pressure.

(d) With gravity and capillary pressure.

Figure 7: Cumulative number of nonlinear iterations for test case 1.

Figure 9 shows the permeability and the saturation profile after 100 days along with the simulation mesh. The total time of the simulation is 150 days and the time step is set to 1 day. The breakthrough of water is within this time scale. Figure 10 shows the cumulative number of nonlinear iterations over the simulation time for all cases. In contrast to test case 1 the machine learning acceleration does not perform better than the other relaxations in

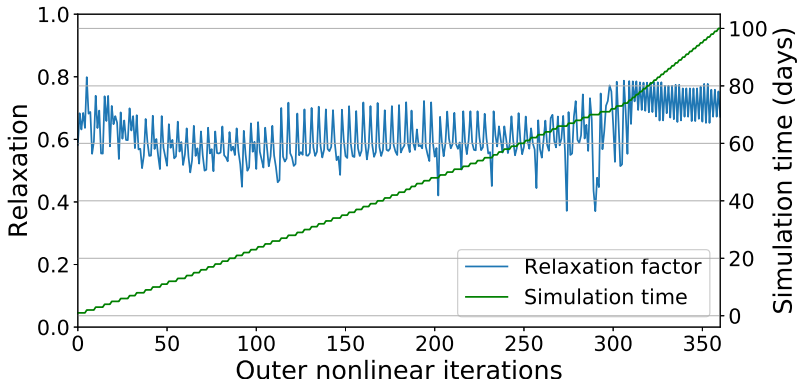


Figure 8: Values of the relaxation parameter generated by the machine learning acceleration.

all cases. Figure 10b shows that the fixed relaxation of 0.5 performs better, after 75 days of simulation, in the case with gravity and no capillary pressure. However, for the other three cases (Figures 10a, 10c and 10d) the proposed acceleration achieved the best performance.

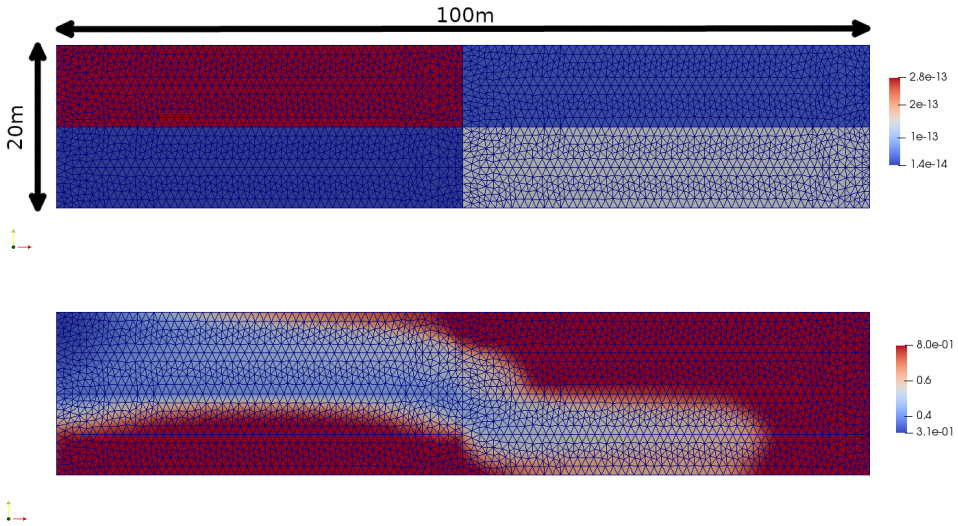
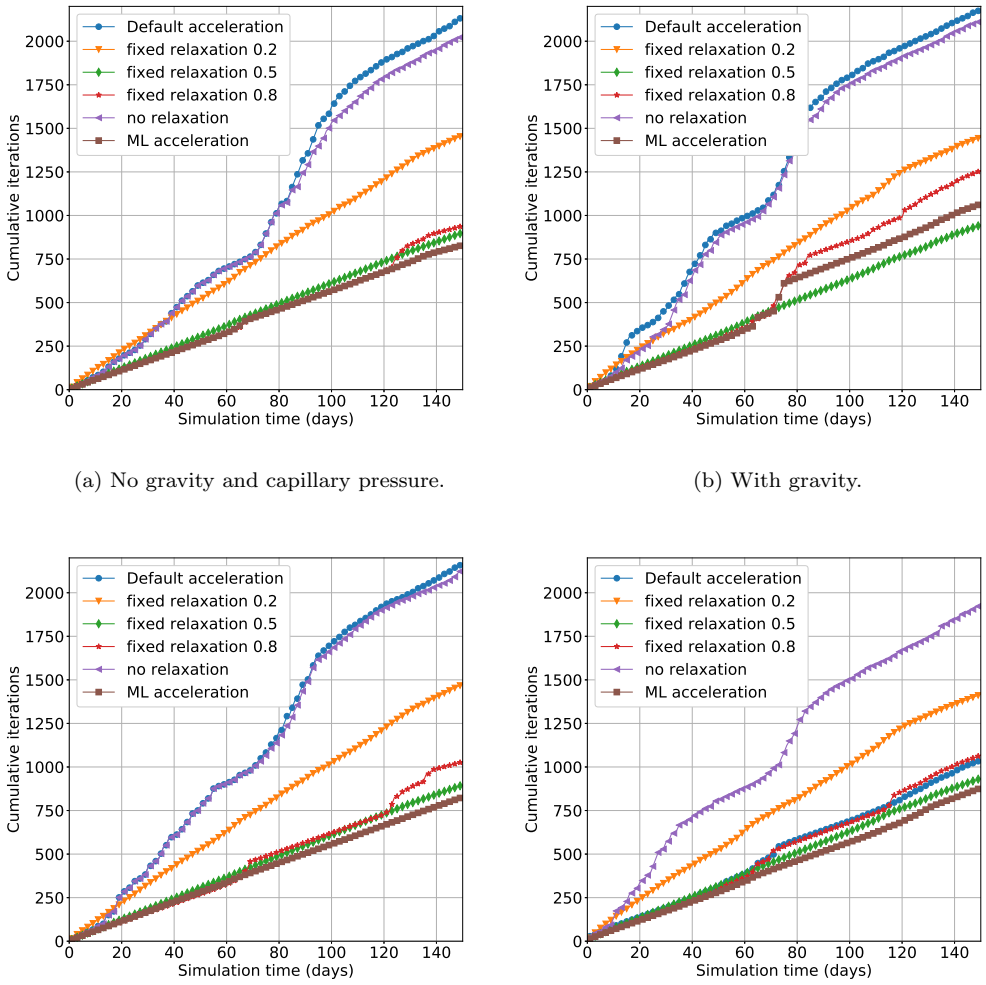


Figure 9: Permeability field in m^2 (top) and oil saturation after 100 days (bottom).



(a) No gravity and capillary pressure.

(b) With gravity.

(c) With capillary pressure.

(d) With gravity and capillary pressure.

Figure 10: Cumulative number of nonlinear iterations for test case 2.

5.3. Test case 3

A more realistic three-dimensional model of fluvial sandstone channels embedded in a low permeability mudstone background is also tested (Figure 11). The channels are divided in three sets, thin channels, medium size channels, and wide channels. The permeabilities in each set of channels are 1000 mD, 200 mD, and 100 mD, respectively. The vertical permeability is

equal to the horizontal and the porosity is 20% in all channels. We inject water on one side and produce oil and water on the opposite one by imposing a fixed pressure on both boundaries. The third column of Table 4 shows the rock-fluid properties of this case.

Figure 11 shows the permeability and the saturation profile after 1000 days along with the simulation mesh. The total time of the simulation is 1000 days and the time step is set to 10 days. The breakthrough of water is within this time scale. Figure 12 shows the cumulative number of nonlinear iterations over the simulation time for all cases. The machine learning acceleration performs better than the other relaxations practically over all time steps. It is worth mentioning that in the case with gravity and capillary pressure (Figure 12d) the simulation with no relaxation fails to converge.

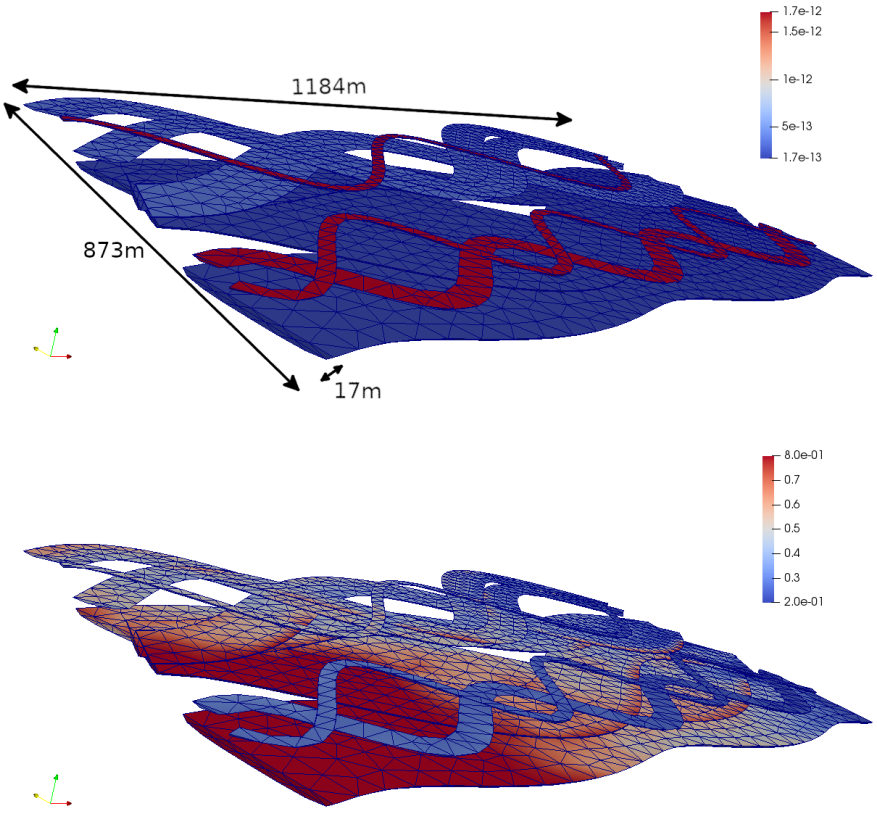
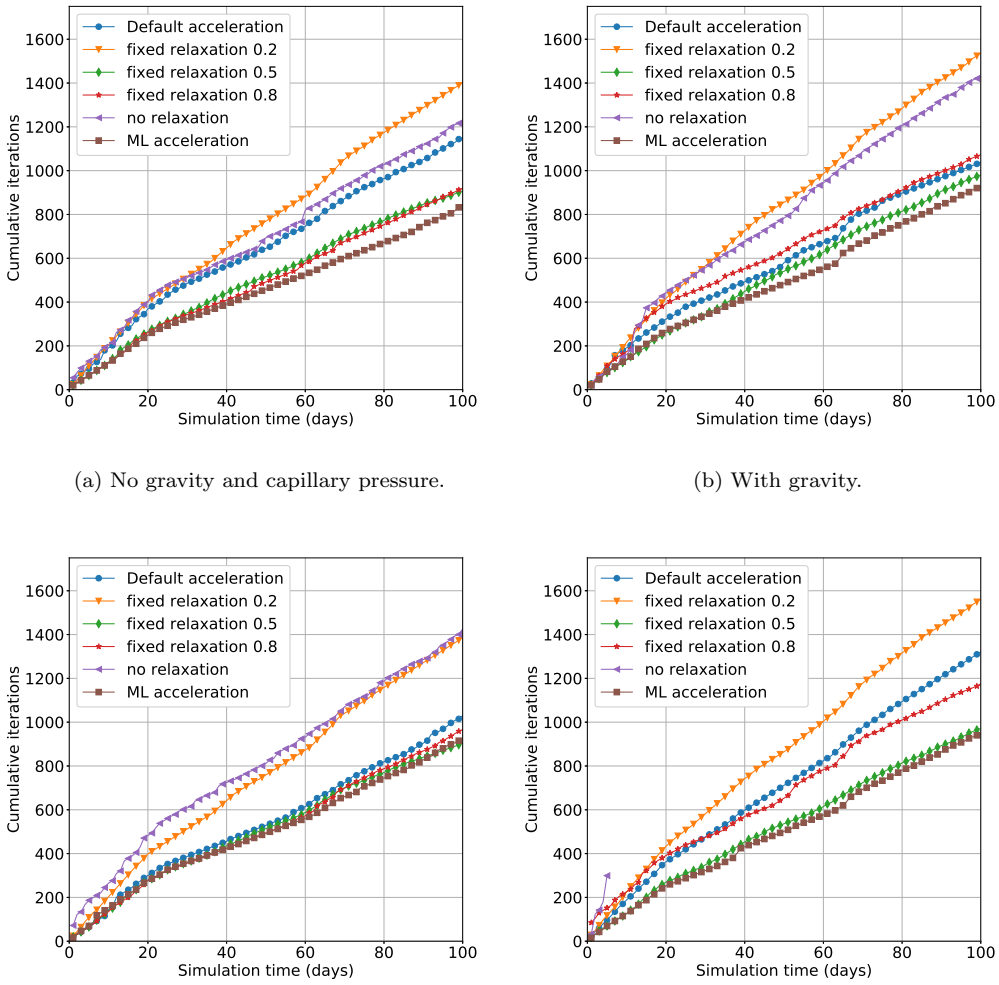


Figure 11: Permeability field in m^2 (top) and oil saturation after 1000 days (bottom).



(a) No gravity and capillary pressure.

(b) With gravity.

(c) With capillary pressure.

(d) With gravity and capillary pressure.

Figure 12: Cumulative number of nonlinear iterations for test case 3.

5.4. Test case 4

Test case 4 is a second more realistic three-dimensional model of a faulted reservoir (Figure 13). The reservoir comprises a sequence of alternated sandstone and mudstone layers. The porosity in the sandstone layers is 10% and the horizontal permeability is 1000 mD. The mudstone layers have 20% porosity and 1 mD of horizontal permeability. The vertical permeability is

equal to the horizontal. We inject water on one side and produce oil and water on the opposite one by imposing a fixed pressure on both boundaries. The fourth column of Table 4 shows the rock-fluid properties of this case.

Figure 13 shows the permeability and the saturation profile after 100 days along with the simulation mesh. The total time of the simulation is 200 days and the time step is set to 1 days. The breakthrough of water is within this time scale. Figure 14 shows the cumulative number of nonlinear iterations over the simulation time for all cases. In Figures 14a and 14b the fixed relaxation of 0.8 has the best results, although it is closely followed by the machine learning acceleration. When capillary pressure is added (Figures 14c and 14d) the fixed relaxation of 0.8 fails to converge. In these cases the machine learning acceleration continues to work well and delivers the best performance.

6. Discussion

Handling properly nonlinearities in systems of partial derivative equations is always a challenging and time-consuming issue that severely affects the performance of the overall system and that requires an extensive knowledge of the system to be solved for to be able to handle it properly. The approach presented here shows, in the authors opinion, an easy way to deal with such nonlinear systems without requiring as much effort as a tailored nonlinear solver, while providing excellent results. In this way, we think that the machine learning acceleration is not constrained to the porous media flow equations but extendable to any other system that can be studied based on dimensionless numbers and that a relaxation approach can be used to stabilize the nonlinear solver.

The machine learning acceleration presented here performed better than the fixed relaxations and the default acceleration presented in IC-FERST for almost all the cases, see Figures 7, 10, 12 and 14. Note that in our 'proof of concept' implementation reported here, there was no simulation walltime improvement despite the observed reduction in non-linear iterations. This is simply because an external python library was used to implement the random forest, with the outputs passed 'on-the-fly' to the numerical simulator on each outer non-linear iteration. Measurements of the loading time of the libraries and the random forest cost shows that changing to an embedded random forest library will significantly reduce this cost, consequently

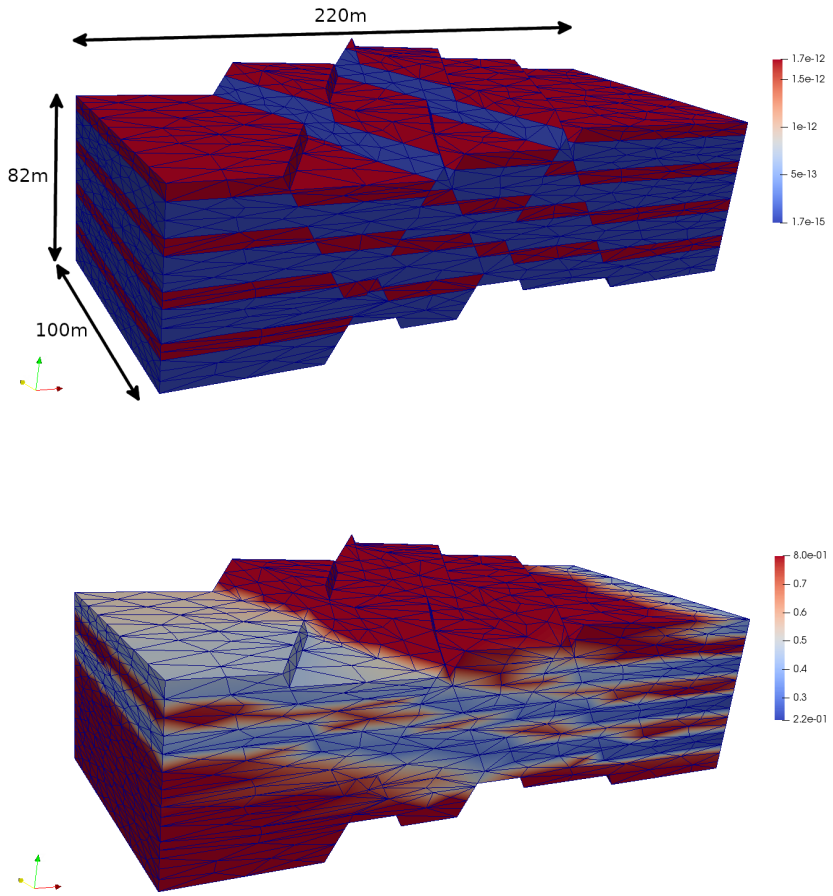
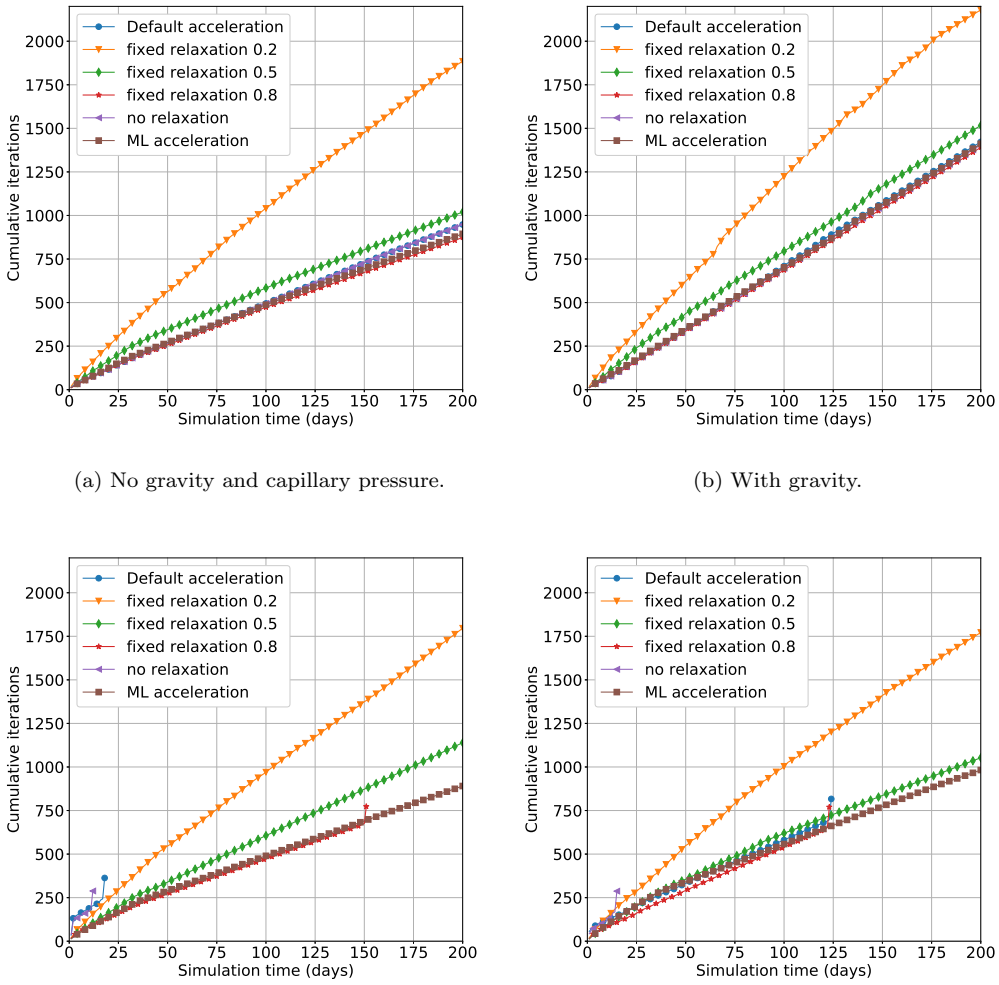


Figure 13: Permeability field in m^2 (top) and oil saturation after 100 days (bottom).

decreasing the walltime proportionally to the reduction of the number of nonlinear iterations.

The average reduction in the number of nonlinear iterations obtained due to the presented method is above 24%. Even the cases where a fixed relaxation is better than the machine learning acceleration, the number of iterations of the latter is close to the best value. It is worth mentioning that the optimum value of the relaxation factor is not known *a-priori* and it is problem specific. Therefore, having an acceleration that adapts itself to the complexity/physics of the system throughout the numerical simulation is of



(a) No gravity and capillary pressure.

(b) With gravity.

(c) With capillary pressure.

(d) With gravity and capillary pressure.

Figure 14: Cumulative number of nonlinear iterations for test case 4.

great value and has driven several publications for different fields. Although a particular fixed relaxation can work well for one specific simulation, for other cases it may not produce goods results. For example, the fixed relaxation 0.8 has the best results for two simulations of the test case 4. Nonetheless, it fail to converge in the same test case when capillary pressure is incorporated.

7. Conclusions

A robust and efficient machine learning nonlinear solvers is presented. The proposed method is implemented within a Picard iterative nonlinear solver with numerical relaxation for multiphase flow in porous media. The main strengths of the presented method are two: (i) The training process is considerably simplified by using dimensionless parameters to train and control the machine learning. As a result, only a simple two-dimensional layered reservoir can be used for training and the machine learning model do not need to be retrained when applied to other reservoirs. (ii) Since it is based on machine learning, it is not completely necessary to deeply understand the physics of the problem to improve the efficiency and robustness of the nonlinear solver, making it, from the authors perspective, easy to implement in different areas. The presented methodology ensures that it is applicable to other types of nonlinear solvers as long as they are based on numerical relaxation. The presented method has shown that it can significantly improve the efficiency and stability of a nonlinear solver. In the numerical experiments studied (the training model and more realistic and challenging reservoir models) the presented method is able to increase the performance more than 24% while also increasing its robustness.

Acknowledgements

The authors gratefully acknowledge PhD funding from Petrobras for Silva, funding from NERC (FAMOS NE/P017452/1) for Salinas, and, from the EPSRC, the PREMIERE programme grant (EP/T000414/1) and MUFFINS grant (EP/P033180/1). The authors also thank Carl Jacquemyn for providing the mesh and models for the numerical test case 4.

References

Aziz, K., 1979. Petroleum reservoir simulation. Applied Science Publishers 476.

Breiman, L., 2001. Random forests. Machine learning 45 (1), 5–32.

Breiman, L., et al., 1998. Arcing classifier (with discussion and a rejoinder by the author). The annals of statistics 26 (3), 801–849.

Brooks, R., Corey, T., 1964. Hydraulic properties of porous media. Hydrology Papers, Colorado State University 24, 37.

Debbabi, Y., Jackson, M. D., Hampson, G. J., Fitch, P. J., Salinas, P., 2017a. Viscous crossflow in layered porous media. Transport in Porous Media 117 (2), 281–309.

Debbabi, Y., Jackson, M. D., Hampson, G. J., Salinas, P., 2017b. Capillary heterogeneity trapping and crossflow in layered porous media. Transport in Porous Media 120 (1), 183–206.

Debbabi, Y., Jackson, M. D., Hampson, G. J., Salinas, P., 2018a. Impact of the buoyancy–viscous force balance on two-phase flow in layered porous media. Transport in Porous Media 124 (1), 263–287.

Debbabi, Y., Stern, D., Hampson, G. J., Jackson, M. D., 2018b. Use of dimensionless scaling groups to interpret reservoir simulation results. Journal of Petroleum Science and Engineering 163, 270–282.

Durlofsky, L. J., 1993. A triangle based mixed finite element?finite volume technique for modeling two phase flow through porous media. Journal of Computational Physics 105 (2), 252–266.

Elman, H., Silvester, D., , Wathen, A., 2005. Finite Elements and Fast Iterative Solvers: With Applications in Incompressible Fluid Dynamics, 1st Edition. Oxford University Press.

Ertekin, T., Abou-Kassem, J., King, G., 2001. Basic Applied Reservoir Simulation. SPE textbook series. Society of Petroleum Engineers.

Freitas, M. M. d., Souza, G. d., Souto, H. P. A., 2020. A picard-newton approach for simulating two-phase flow in petroleum reservoirs. International Journal of Advanced Engineering Research and Science 7 (4), 428–457.

Fung, L.-K., Hiebert, A., Nghiem, L. X., et al., 1992. Reservoir simulation with a control-volume finite-element method. SPE Reservoir Engineering 7 (03), 349–357.

Gomes, J., Pavlidis, D., Salinas, P., Xie, Z., Percival, J. R., Melnikova, Y., Pain, C. C., Jackson, M. D., 2017. A force-balanced control volume finite element method for multi-phase porous media flow modelling. International Journal for Numerical Methods in Fluids 83 (5), 431–445.

Hoteit, H., Firoozabadi, A., 2008. Numerical modeling of two-phase flow in heterogeneous permeable media with different capillarity pressures. Advances in water resources 31 (1), 56–73.

Jackson, M., Percival, J., Mostaghimi, P., Tollit, B., Pavlidis, D., Pain, C., Gomes, J., Elsheikh, A. H., Salinas, P., Muggeridge, A., et al., 2015. Reservoir modeling for flow simulation by use of surfaces, adaptive unstructured meshes, and an overlapping-control-volume finite-element method. SPE Reservoir Evaluation & Engineering 18 (02), 115–132.

Jenny, P., Lee, S. H., Tchelepi, H. A., 2006. Adaptive fully implicit multi-scale finite-volume method for multi-phase flow and transport in heterogeneous porous media. Journal of Computational Physics 217 (2), 627–641.

Jiang, J., Tchelepi, H. A., 2019. Nonlinear acceleration of sequential fully implicit (sfi) method for coupled flow and transport in porous media. Computer Methods in Applied Mechanics and Engineering 352, 246–275.

Küttler, U., Wall, W. A., 2008. Fixed-point fluid–structure interaction solvers with dynamic relaxation. Computational mechanics 43 (1), 61–72.

Li, B., Tchelepi, H. A., 2015. Nonlinear analysis of multiphase transport in porous media in the presence of viscous, buoyancy, and capillary forces. Journal of Computational Physics 297, 104–131.

Lott, P., Walker, H., Woodward, C., Yang, U., 2012. An accelerated picard method for nonlinear systems related to variably saturated flow. Advances in Water Resources 38, 92–101.

Oladokun, E., Sheth, S., Jönsthövel, T., Neylon, K., 2020. Machine-learning informed prediction of linear solver tolerance for non-linear solution methods in numerical simulation. In: ECMOR XVII. Vol. 2020. European Association of Geoscientists & Engineers, pp. 1–10.

Ortega, J. M., Rheinboldt, W. C., 1970. Iterative solution of nonlinear equations in several variables. In: Computer science and applied mathematics.

Pavlidis, D., Xie, Z., Percival, J. R., Gomes, J. L., Pain, C. C., Matar, O. K., 2014. Two-and three-phase horizontal slug flow simulations using an interface-capturing compositional approach. International Journal of Multiphase Flow 67, 85–91.

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., Duchesnay, E., 2011. Scikit-learn: Machine learning in Python. Journal of Machine Learning Research 12, 2825–2830.

Salinas, P., Pain, C., Osman, H., Jacquemyn, C., Xie, Z., Jackson, M., 2020. Vanishing artificial diffusion as a mechanism to accelerate convergence for multiphase porous media flow. Computer Methods in Applied Mechanics and Engineering 359, 112535.

Salinas, P., Pavlidis, D., Xie, Z., Adam, A., Pain, C., Jackson, M., 2017a. Improving the convergence behaviour of a fixed-point-iteration solver for multiphase flow in porous media. International Journal for Numerical Methods in Fluids 84 (8), 466–476.

Salinas, P., Pavlidis, D., Xie, Z., Jacquemyn, C., Melnikova, Y., Jackson, M. D., Pain, C. C., 2017b. Improving the robustness of the control volume finite element method with application to multiphase porous media flow. International Journal for Numerical Methods in Fluids 85 (4), 235–246.

Wong, Z. Y., Kwok, F., Horne, R. N., Tchelepi, H. A., 2019. Sequential-implicit newton method for multiphysics simulation. Journal of Computational Physics 391, 155–178.