Geoscientific Model Development Discussions

Geosci. Model Dev. Discuss., author comment AC3
https://doi.org/10.5194/gmd-2020-445-AC3, 2021

Reply on RC1

Marco De Lucia and Michael Kühn

Author comment on "DecTree v1.0 – chemistry speedup in reactive transport simulations: purely data-driven and physics-based surrogates" by Marco De Lucia and Michael Kühn, Geosci. Model Dev. Discuss., https://doi.org/10.5194/gmd-2020-445-AC3, 2021

We thank the anonymous reviewer for his comments and suggestions.

Comment: *Line 60: "For this reason, we argue that the most sensible choice for a surrogate modelling framework is that of multiple multivariate regression: one multivariate regressor - making use of many or all inputs as predictors - is trained independently for each distinct output variable, while the choice of regressor may vary from variable to variable."*

So, each output variable has its own ML model? This seems like it is not well explained, and a reader might believe this to be more computationally expensive. A better framing is that it is difficult to get all output variables to perform well with a single ML model. A citation that shows this here (https://doi.org/10.1029/2020JD032759) where you can optimize a specific output variable's performance above others (thereby retaining the same input/output dimensionality), or just simply make new ML models for each output. This shows that one ML model is not effective in capturing all variables and that multiple specialized models must be made.

Response: We agree with this suggestion and rephrased the whole paragraph, including the suggested citation, in order to clarify our choice in pursuing multiple multivariate regression.

The new text reads:

"With algorithms such as Artificial Neural Networks (ANN) it is possible to train one single network and hence in practice one single surrogate model for all output variables at once. While ANN in particular usually require long CPU times for training and quite large training datasets, they offer large speedups when used for predictions (Jatnieks et al., 2016; Prasianakis et al., 2020) and furthermore they can efficiently leverage GPUs (Graphic Processing Units) for even larger acceleration. It is however difficult to achieve the required accuracy simultaneously for all output variables (Kelp et al., 2020). For this reason, we focus on a more flexible approach, multiple multivariate regression: one distinct multivariate regressor - i.e., making use of many or all inputs as predictors - is trained independently for each distinct output variable. This approach allows using different specialized models from variable to variable, including different regression methods altogether, preprocessing and hyperparameter tuning, while not necessarily requiring larger computing resources."

Comment: *Line 66: "In praxis, the CPU-time, the user interactions and the overall skills required for optimally training complex regressors cannot be underestimated and may prove overwhelming for a geoscientist. The whole process of hyperparameter tuning, required by most advanced machine learning algorithms, while being an active area of development and research, is still hardly fully automatable."*

I do not like this reasoning too much, yes neural networks (NNs) take longer and are more difficult to train, but they evaluate faster than regression trees and can take better advantage of GPUs for an even greater speed gain. Furthermore, grid search, Bayesian hyperparameter tuning, genetic neural algorithms are all automatable hyperparameter tuning methods that would be simple to implement with your 1-D, low-dimensional system (though I am not suggesting you do that here).

Response: It is indeed our opinion that the geoscientific modeller, with the actually required curricular skills, can be overwhelmed when confronted with complex tuning of AI models. This observation was rather directed at the necessity of increasing the weight of data science/AI in this domain than at the lack of fully automatic parameter tuning infrastructures/methods. In any case we reformulated this paragraph and mentioned the popularized frameworks for hyperparameter tuning in the discussion.

Comment: *Line 85: "Since chemistry is inherently an embarassing parallel task, the speedup achieved on a single CPU as in this work will transfer - given large enough simulation grids making up for the overhead - on parallel computations." Not sure what this means. Gas-phase chemistry is already parallelizable in air quality applications of atmospheric transport.*

Response: We mean here that, in an operator-splitting approach, at each coupling time step one single, independent simulation of chemistry is computed per each grid element. In this sense, this is a case of "embarassing parallelism". Rephrased accordingly.

Comment: *Paragraph starting at Line 210: "The choice of the regressor for each output is actually arbitrary, and nothing forbids to have different regressors for each variables, or even different regressors in different regions of parameter space of each variable. Without going into details on all kinds of algorithms that we tested, we found that decision-tree based methods such as Random Forest and their recent gradient boosting evolutions appear the most flexible and successful for our purposes. Their edge can in our opinion be resumed by: (1) implicit feature selection by construction, meaning that the algorithm automatically recognizes which input variables are most important for the estimation of the output; (2) no need for standardisation of both inputs and outputs; (3) ability to deal with any probability distributions; (4) fairly quick to train with sensible hyperparameter defaults; (5) extremely efficient implementations available." I do not like the reasoning of this paragraph from an ML perspective. In terms of your reasoning:*
1. The feature selection automation from RFs are not accurate if your inputs are colinear in any way, and I do not think your 7 features are completely independent of one another,

2. your next paragraph makes sense about this though you contradict it immediately in the following paragraph when writing about scaling outputs. I would not say that this is too much of an advantage, it just removes an annoyance,

3. most other ML algorithms other than Gaussian processes can deal with any type of distribution as they are (for the most part) nonparametric learners,

4. they are quicker to train than NNs but also slower to evaluate than a NN,

5. perhaps, but I would not say that a NN is much harder when using Python or R.

The main edge in terms of using an RF rather than a NN as a surrogate is that an RF will always fall back on data it has seen and predict a mean state when it does not know what to output, whereas a NN is a regressor that will extrapolate outside of its training domain and accumulate errors much faster than an RF. Thus, RF (or xgboost) is typically more robust than a NN.

Response: We agree with this comment. In our mind we did not actually intend to make a comparison between decision-tree methods and Neural Networks, however these remarks were incorporated in the manuscript since they are valuable to understand our reasoning. In detail:

1. Chemistry is indeed a highly non-linear process, and this is reflected in a poor colinearity between the features. They are not independent, but the dependency is non-linear, how on can see in the exemplary choice of "engineered features" used in the physics-informed decision tree approach.

2. we noted this fact about scaling of labels because it contradicts the documentation and the theory about decision tree based regressors such as xgboost.

3. we agree with this remark and we removed this point.

4. agreed and accordingly reformulated.

5. removed. Added the robustness property here.

Comment: *Figure 2, 12, 13: A little confused here, what are the axes labels?*

Response: We added axes and labels in figure 2; the other figures displaying variables profiles across the 1D simulation domain are left unchanged since they act as graphical comparison and the axes are always the same.

Comment: *Line 278: Can xgboost use a GPU? I know that RFs cannot really use them but perhaps a factor of 10x gain may be had by using a GPU in the future.*

Response: Yes, a GPU backend is available for xgboost, however we did not try it. This has been now noted in the manuscript.

Comment: *Figure 14: Why do you not call the PHREEQC solver here? Did it never fail the mass balance criteria or is that not implemented here. Wondering what would happen if you routinely called the PHREEQC solver (e.g. at every 10% of simulation progress time), and ran it for several iterations if the physics solver can be used to dampen error?*

Response: In the "physics informed approach" we casted the geochemical problem in such a way that the mass balance is honored by construction, since the regressed variables are the $Delta$ calcite and $Delta$ dolomite, and the changes in aqueous concentrations depend linearly on them (Cl and pH are treated separately). This is in contrast to the purely data-driven approach, in which we basically assume no knowledge on the colinearity between concentration changes on mineral deltas. We now added some clarifications in the description of the physics-based approach.

The second part of this question is addressed together with the next one. Briefly, calling PHREEQC does not dampen errors if the error or "unphysicality" is already present in the features at the beginning of the iterations.

Comment: *General clarifications: For your simulations, you call the full physics solver when the output variables fail the mass balance screening. Is it the case that once the*

surrogate fails the mass balance test the solver is just continually called or is there a dampening of error from using the full solver?

Response: At each iteration the full physics solver is called only for the grid elements where the surrogates fail the mass balance (this is valid only for the "purely data-driven" approach). The process repeats in whole at the following iteration, regardless whether in the previous iteration a surrogate or a full physics simulation had been retained. Note (also w.r.t. reviewer's previous remark) that calling the full physics simulator resets "locally" the mass balance error for the given grid elements and time step. However, these calls do not ensure that the corresponding results are brought back to adhere perfectly to the reference or "true" state; we are only guaranteed that the new solution is completely physical, given the inputs at the start of the iteration. However, it may be that the accumulated inaccuracies in specific grid elements already produced diverging trajectories for the chemical process. We don't have a way, at runtime, to check or evaluate this, and we must rely on the assumption that the previous iterations did not introduce significant inaccuracies. As figure 3 displays, inaccuracies propagate in time - and hence in space - through the coupled simulations even when many full physics geochemical simulations are called. We added a paragraph explaining this reasoning to section 2.4.

Comment: *Do you expect this relationship to change with a higher-dimensional system or higher grid resolution (2D, 3D model)?*

Response: We argue that the dimensionality (2D or 3D) of the hydrodynamic problem does not have much impact on this fact, if all other hypotheses are equal to our present 1D example. In such cases one will however also have much larger datasets to train the surrogate in the first place. These concepts are now mentioned in manuscript's discussion.

Of course things will change when considering variable local velocities, variable permeability and porosity and other hydrodynamic processes such as diffusion, or when time step must be treated as independent variable. There are possible workarounds there, some of which we briefly addressed in the discussion, but we believe this is outside the scope of the present paper.

Comment: *I thought the mass balance screening + feature engineering justification + Figure 11 reasoning was excellent. Well done!*

Response: Thank you.