Using Artificial Intelligence (AI) within COMSOL Multiphysics® to Create Machine Learning Tools

F. Viry[1], M. Sturma[2], P. Namy[1], B. Barbet[2].

1. SIMTEC, Grenoble, France.

2. MARKEM-IMAJE, Bourg-lès-Valence, France.

Abstract

In recent years, the development of artificial intelligence has enabled some disruptive innovations by changing the way research is conducted. More specifically, in computer science, a new way of developing algorithms has emerged, based on machine learning techniques. The core idea is to train the program on many cases to teach the algorithms and then use them with new configurations.

In any industrial process, optimizing the operating conditions can be a heavy trial-and-error work. Traditional optimization methods to find them can be very time consuming, so a machine learning approach can be useful to solve such inverse problems. This article shows how to implement an inverse problem solving strategy using COMSOL Multiphysics® and Python machine learning tools through a practical use case: continuous inkjet technology and viscosity deduction by droplet shape detection.

In the field of industrial marking, continuous inkjet technology is based on high-speed emission of ink droplets. The shape of the emitted droplets is a combination of ink properties and stimulation operating point and has a direct impact on the printing quality. This article explores the role of viscosity by simulating droplet shape for multiple viscosities using COMSOL Multiphysics® (forward problem) and using machine learning techniques to infer viscosity from droplet shape (inverse problem). This use case illustrates how to set up the main stages of a machine learning inverse problem solving strategy: collecting data, selecting and training a model, testing the model and improving its predictive capabilities. The flexibility of COMSOL Multiphysics® makes it easy to interface with Python machine learning tools to efficiently produce valuable results.

Keywords: Artificial Intelligence (AI), Machine Learning (ML), inverse problem, CFD, two-phase flow, continuous inkjet printing.

1 Introduction

In 2024, it seems crucial to be able to couple COMSOL with AI/ML techniques in order to investigate new possibilities and to create efficiently disruptive new technologies. To explore this possibility, we decided to apply it to one of our main areas of work in recent years: the modeling of continuous inkjet printing (CIJ) [1] [2]. More specifically, we would like to assess if the shape of generated droplets could give some indication about the characteristics of the ink.

Indeed, from an industrial point of view, one challenge of the CIJ technology is to be able to print as fast as possible while maintaining a sufficient printing quality. Droplet generation is the first printing step, and all defects generated at this stage may dramatically impact the rest of the printing process. Controlling the shape of the generated droplets is of prime importance, and the technical challenge is to find an operating point capable to produce the optimal shape.

This article proposes a generic numerical workflow to couple and use COMSOL with Python AI/ML tools.

2 Numerical Workflow

Problem Statement: the Use-Case

In CIJ printing, the shape of the droplets produced has a direct impact on print quality. In fact, there is an optimum droplet shape. Many physical parameters in the droplet generation process may impact the shape of the droplets: the geometry of the nozzle, the ink properties, and the operating point. To ensure a certain level of printing quality, optimizing these parameters becomes necessary, and it can be a heavy trial-and-error work, even purely numerically.

In previous works, we developed a numerical model in COMSOL Multiphysics® to simulate the inkjet breakup process from all these parameters [1]. In the following text, simulating this process is called *solving the direct problem.* Optimizing the parameters then becomes *solving an inverse* *problem*, *i.e.* the desired droplet shape is an input, and the values of the parameters generating this shape are the output. The ambition of this work is to use Artificial Intelligence (AI)/Machine Learning (ML) techniques to solve this inverse problem.

One of the many parameters to optimize is the ink viscosity, which can physically be controlled by varying the ink composition and/or the ink temperature. This article focuses on building a numerical model capable to predict the value of the ink viscosity from a given droplet shape.

Direct Problem: Simulating the Inkjet Breakup

The inkjet breakup process is simulated using the 2D-axysymetric diphasic CFD model developed in [1]. This model represents the ink flow inside a droplet generator, composed of a tank and a nozzle, and the free-surface flow of ink in the surrounding air. Between the tank and the air, there is a positive pressure drop in the ink, so that it flows only from the tank to the air. This pressure drop oscillates, and this perturbation propagates along the ink jet, which breaks at some point thanks to the well-known Rayleigh-Plateau instability. The ink is considered as a newtonian and incompressible fluid. Given a periodic pressure stimulation, the numerical model computes the periodic velocity field and phase field of ink and air inside and downstream of the droplet generator.

The inputs of this numerical model are the droplet generator geometry (tank and nozzle dimensions), the ink properties (density, viscosity, and ink/air surface tension), and the operating point (mean flow rate of the ink, perturbation frequency and amplitude).

In the context of this work, the output of this numerical model is the *droplet shape at break*. Physically, the jet is in fact separated in two parts: the continuous one (the jet) and the discontinuous one (the droplets). At the exact instant when the length of the continuous part of the jet suddenly decreases, it means that a droplet has just formed at its end and is breaking away from the jet. The shape of the newly formed droplet at this exact instant defines what is called the *droplet shape at break*. In the model, this shape may be recognized from a 2D or 3D representation of one isolevel of the phase field variable. To simplify its representation, this shape is represented as a point cloud $(z_i, r_i)_{i=1}^{m}$, where z_i is the height coordinate and r_i is the local radius in the cylindrical coordinate system along the jet axis. This point cloud can be implemented in COMSOL using projection operators, to compute the radius of the droplet at each height, and represented in a 1D plot. This point cloud can then be exported as a TXT file, which forms the output of this numerical model.

To avoid any confusion with the upcoming models, this numerical model is called in this paper *the direct problem.*

Data Generation

In this work, supervised Machine Learning techniques are used, which consists in finding and optimizing a model, from known *inputs* → *outputs*.

The goal of this work is to build the inverse of the function giving the generated droplet shape in function of the ink viscosity, provided that all other parameters of the inkjet breakup process remain the same. Then, given a droplet generator geometry, the density and the ink/air surface tension, and the operating point, the direct problem is solved for multiple viscosity values. For each viscosity, a TXT file containing the point cloud representing the droplet shape at break is produced.

The dataset used to optimize a model solving the inverse problem *droplet shape at break to viscosity* can mathematically represented as the set:

$$D_{raw} = \left\{ (z_i, r_i)_{i=1}^{m_j} \to \mu_j \quad \text{for } j = 1, \dots, N \right\} \qquad (1)$$

where μ_j designates the value of viscosity, and N is the number of values of viscosity considered.

Data Pre-Processing

From here, all the upcoming operations are performed in a Python script. The output data of the direct problem have a disadvantage: the droplet shapes may not be represented on exactly the same height grid, and cannot be directly used in an AI/ML workflow. This issue is solved by defining a unique height grid $(z_i)_{i=1}^{m}$ of size m, and interpolating each droplet on this grid, forming the pre-processed dataset:

$$D = \left\{ (r_i)_{i=1}^{m} \to \mu_j \quad \text{for } j = 1, \dots, N \right\}. \qquad (2)$$

Development of the Machine Learning Model

A supervised Machine Learning model is a mathematical function *inputs* → *outputs* depending on *parameters* that have to be optimized. Multiple models may explain the relationship between the known inputs and outputs, but only a few are capable to extrapolate to unseen data.

Usually, the dataset is divided in two groups: a *training set* D_{train} and a *test set* D_{test}:

$$D = D_{train} \cup D_{test}, \quad D_{train} \cap D_{test} = \emptyset. \qquad (3)$$

The training set is used to optimize the model provided a certain error metric, and the test set is helpful to estimate the extrapolation capacities of the model. A usual error metric is the least squares metric:

$$LS(D, \mathcal{M}) = \sum_i (\mathcal{M}(x_i) - y_i)^2 \qquad (4)$$

where $D = \{x_i \rightarrow y_i \text{ for } i = 1 \dots N\}$ designates the *inputs* → *outputs* dataset, and $\mathcal{M}$ designates the model. When this metric has a high value over the training set, the model faces the *underfitting* issue: it is incapable to predict outputs from known inputs. When this metric has a low value over the train set, but a high value over the test set, the model faces the *overfitting* issue: the model predicts perfectly outputs from known inputs, but is unable to extrapolate with new inputs.

A good prediction model does not encounter these issues, *i.e.* the error metric is low for both train and test sets. For that, a *simple* enough model must be chosen. The degree of complexity of a model is often measured by the number of degrees of freedom, *i.e.* the number of parameters of the model.

In this work, a linear model has been chosen to explain the relationship between the droplet shape at break, and the viscosity. Mathematically, the viscosity prediction $\hat{\mu}$ is expressed as a linear combination of the radii of the droplet shape at break $(r_i)_{i=1}^{m}$:

$$\hat{\mu} = \mathcal{M}_\alpha(r_1, \dots, r_m) = \alpha_0 + \sum_{i=1}^{m} \alpha_i \cdot r_i \qquad (5)$$

where $\alpha = (\alpha_i)_{i=1}^{m+1}$ are the parameters to optimize.

Once the model is selected, the next step consists in *training* and *refining* the model. Training is the action of optimizing the model parameters to best fit the data in the training set, in the sense of the error metric. Mathematically, the optimization problem to solve writes:

$$\min_{\alpha} \quad LS(D_{train}, \mathcal{M}_\alpha). \qquad (6)$$

High values of m ($i.e.$ the discretization of the droplet shapes is fine) makes the optimization problem Eq. 6 with model Eq. 5 ill-posed: there are more degrees of freedom to solve than “equations”. A common way to solve this issue is to use *regularization* techniques, *e.g.* the LASSO regularization, consisting in adding another term in the objective function of Eq. 6 aiming to eliminate some degrees of freedom:

$$\min_{\alpha} \quad LS(D_{train}, \mathcal{M}_\alpha) + \lambda \sum_{i=1}^{m+1} |\alpha_i| \qquad (7)$$

where λ is a coefficient controlling the regularization strength. The optimization problem of Eq. 7 with model Eq. 5 is well-posed. When $\lambda \rightarrow 0$, solving Eq. 7 leads to one of the many solutions of Eq. 6, and certainly to overfitting. When $\lambda \rightarrow \infty$, the solution becomes $\alpha_i = 0$, which leads with high probability to underfitting. But a good model may emerge by choosing a value of λ between these boundaries.

A good value of λ may be chosen by minimizing the prediction error over the test set, *i.e.*:

$$\min_{\lambda} \quad LS(D_{test}, \mathcal{M}_{\alpha(\lambda)}). \qquad (8)$$

In practice, the optimization problem Eq. 7 can be solved numerically in our Python script by using the `Lasso` model of the `scikit-learn` module. Then, multiple values of λ can be evaluated to solve the optimization problem Eq. 8 and find a value λ^* that may prevent both overfitting and underfitting issues. By solving Eq. 7 with this optimal value λ^*, the model predicting the viscosity from the droplet shape at break is obtained.

3 Results and Discussion

Dataset, Training Set, Test Set

The whole dataset, training set and test set are illustrated in Figure 1.

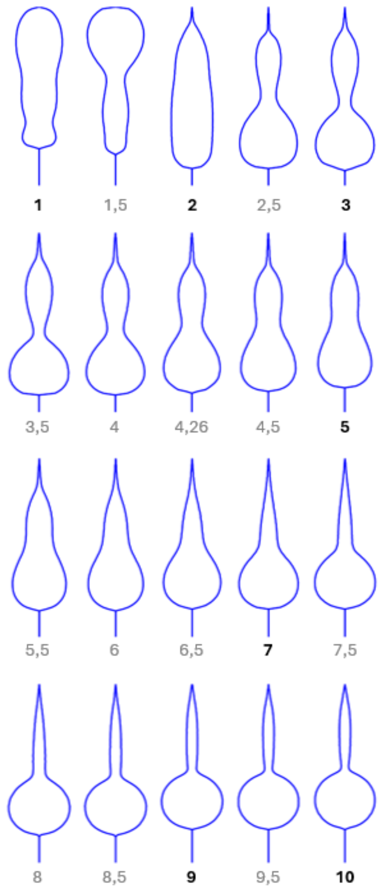

*Figure 1. Dataset: droplets shapes at break simulated for multiple values of viscosities (in cP). Viscosities **in bold** indicates the values constituting the training set. The other values indicate the test set.*

Performance of the Trained Models

The performance of the models Eq. 5 trained using Lasso regularization coefficients $\lambda = 10^{-9}$, 10^{-1}, 10^3 are illustrated in Figure 2, Figure 3, and Figure 4. In these graphs, each point represents the expected and the predicted viscosity for the data coming from both the training and the test sets. The grey dashed line represents the equality between the expectation and the prediction: it represents the perfect model. For good models, all the points must be close to this line.

For $\lambda = 10^{-9}$ (Figure 2), the viscosity is perfectly predicted in the training set, while the viscosities of the test set are very badly predicted. This is typical of *overfitting*. For $\lambda = 10^3$ (Figure 4), bad predictions of the viscosities are obtained on both the training set and the test set. This is typical of *underfitting*. To explain these results, it must be recalled that one of the specificities of the LASSO regularization is to nullify a subset of the variables to optimize, as large as λ. With $\lambda = 10^{-9}$ (Figure 2), the regularization is too weak and almost all the m coefficients ($m = 100$ in this experiment) are non-zero, meaning that, for this model, the viscosity may be explained by almost all radii defining the droplet shape. With $\lambda = 10^3$ (Figure 4), only one coefficient (the mean of the training viscosities) is non-zero: the model is too simple to explain the viscosity. For $\lambda = 10^1$ (Figure 3), good predictions are globally made for both the training and the test set. In that case, 7 coefficients of the model are non-zero: the LASSO regularization selected the most relevant radii that explain the viscosity.

Concerning the model obtained with $\lambda = 10^{-1}$, one particular case of "bad prediction" is the prediction of $\hat{\mu} = 4.2$ cP for the droplet shape obtained with $\mu = 2.5$ cP. In this case, it has to be noted that the droplet shape obtained with $\mu = 2.5$ cP is very close to the one obtained with $\mu = 4$ cP (Figure 1). The prediction error is then not completely surprising. In fact, inverse problems have almost never a unique solution. For a physical process, it means that it is possible to produce the same output with different inputs. In our case, this is not a problem. Imagine that the optimal droplet shape is the one obtained with $\mu = 2.5$ cP (Figure 1). With this droplet shape, the model predicts that a viscosity $\hat{\mu} = 4.2$ cP can produce this shape, which is more or less the case.

These results are very promising, so that the next step of this work concerns the prediction of more process inputs, one-by-one.

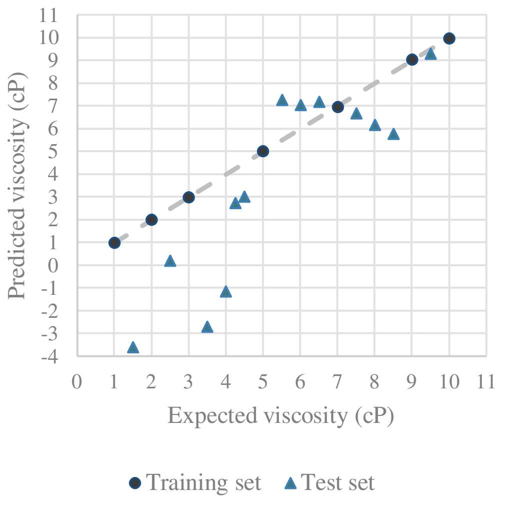

Figure 2. Performance of the model with a regularization coefficient $\lambda = 10^{-9}$.

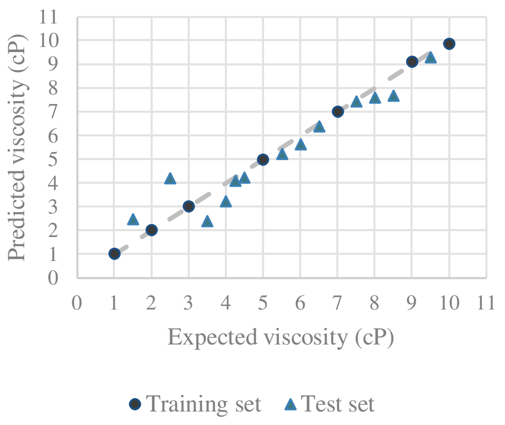

Figure 3. Performance of the model with a regularization coefficient $\lambda = 10^{-1}$.

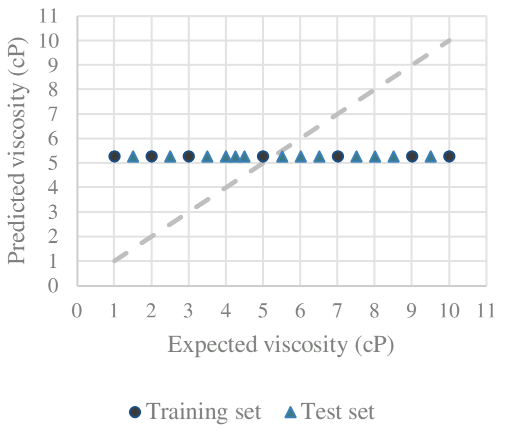

Figure 4. Performance of the model with a regularization coefficient $\lambda = 10^3$.

4 Conclusions

Having the possibility to couple COMSOL with AI/ML tools is necessary to investigate new directions and push the optimization of industrial processes even further.

This article proposed a numerical workflow to achieve such coupling on an industrial inverse problem: determining the value of an ink property allowing to generate a specific droplet shape in the context of the CIJ technology. This issue has been addressed by detailing how to generate (in COMSOL) and pre-process data, and how to choose, train, and refine a prediction model (in Python). The communication between both tools is facilitated by the advanced post-processing tools and the export TXT feature within COMSOL. The results are very promising as a very relevant and accurate prediction model has emerged from this work. The coupling of COMSOL with AI/ML tools is then shown to be possible and to be driving major scientific advances in every industry.

References

[1] M. Sturma, A. Monlon, P. Namy, F. Viry and B. Barbet, "Modelling of Droplet Charge Dynamics during an Ink Jet Breakup using COMSOL Multiphysics®," in *COMSOL Conference*, Munich, 2023.

[2] F. Viry, M. Sturma, P. Namy and B. Barbet, "Towards Accurate Modelling of Aeraulic Droplets Interactions within COMSOL Multiphysics®," in *Comsol Conference*, Munich, 2023.

[3] F. Pedregosa et al., "Scikit-learn: Machine Learning in Python," *JMLR 12,* pp. 2825-2830, 2011.

Acknowledgements

This work has been founded by MARKEM-Imaje and made in a fruitful collaboration between MARKEM-Imaje and SIMTEC.