Towards an accurate pressure estimation in injection molding simulation using surrogate modeling

Sandra Saad[1,2] · Alankar Sinha[3] · Camilo Cruz[1] · Gilles Régnier[4] · Amine Ammar[2]

Abstract
The computational cost of high-fidelity injection molding simulations has been growing in the past years making it more and more challenging to use them for performing analyses such as optimizations or uncertainty quantification. Surrogate modeling offers a cheaper way to realize such studies and has been gaining attention in the field of injection molding simulation. In this work, we propose to compare three surrogate modeling techniques along with two design of experiment methods in their ability to predict the pressure signal at a surface node in a Moldflow simulation by varying process and modeling parameters. A Sobol sensitivity analysis is performed to study the contribution of the varied parameters on the pressure results. In addition, one of the generated models is used along with experimental pressure sensor data to improve the pressure estimation by calibrating the heat transfer coefficients during filling and packing as well as the pressure-dependency coefficient in the Cross-WLF viscosity model. This resulted in major improvements of the pressure predictions for all 27 considered cases in comparison to using the default heat transfer coefficients and viscosity model parameter.

Keywords Surrogate Modeling · Injection Molding · Process Simulation · Parameter Optimization · Sensitivity Analysis

Introduction

The injection molding process is one of the most widely used plastics manufacturing technique in the industry. It enables the high production volumes of complex plastic parts while having short cycle times [1]. A cycle constitute mainly of six phases starting with the mold closing to form the cavity, followed by the filling phase where a plastic melt is injected into the cavity. After the cavity is filled, the packing phase begins by applying pressure on the melt, which simultaneously solidifies in contact with the cold mold. In parallel to this in-mold cooling phase, the screw starts to move back helping in the plastification of the next shot. And, finally, the produced part is ejected from the mold and the cycle restarts [1]. To ensure a high product quality and minimize design/production costs, injection molding simulation is broadly used in order to efficiently design molds and enable the identification of optimal process settings that mitigate common defects such as warpage, shrinkage, weldline and short shots.

However, over the last years, the growing interest in highly accurate simulations resulted in the increase of the complexity of the embedded models and thus to higher computational costs. This is especially problematic in case of using such simulation models for optimization routines or

✉ Sandra Saad
sandra.saad@de.bosch.com

Alankar Sinha
alankar.sinha@rwth-aachen.de

Camilo Cruz
camilo.cruz@de.bosch.com

Gilles Régnier
gilles.regnier@ensam.eu

Amine Ammar
amine.ammar@ensam.eu

[1] Corporate Sector Research and Advance Engineering, Robert Bosch GmbH, Robert-Bosch-Campus 1, Renningen 71272, Germany

[2] LAMPA Laboratory, Arts et Métiers Institute of Technology, Boulevard du Ronceray 2, Angers 49035, France

[3] Institute of General Mechanics, RWTH Aachen University, Eilfschornsteinstrasse 18, Aachen 52062, Germany

[4] PIMM Laboratory, Arts et Métiers Institute of Technology, Boulevard de l'Hôpital 151, Paris 75013, France

uncertainty quantification. One way to overcome this challenge is by using a surrogate model, also known as a metamodel, of these high-fidelity simulations, to perform such analyses in a cheaper way. These models are usually generated using a relatively low number of simulations obtained by varying a specific number of input parameters and are then able to approximate the output of interest in a matter of seconds instead of minutes or hours using the high-fidelity simulation.

In the last two decades, the use of surrogate models to approximate outputs from the injection molding simulation has been growing steadily especially in the field of process parameter optimization to enhance product quality and molding efficiency. Gao and Wang [2] employed a Kriging approximation model along with an adaptive optimization technique to minimize the warpage in produced parts by varying process parameters such as the mold and melt temperature, injection time as well as the holding pressure and time. Similar works were performed by Chen et al. [3], Wang et al. [4] and Kang et al. [5]. Others used radial basis function [6–8], artificial neural networks [9, 10], Gaussian process [11] as surrogate modeling technique to optimize process parameters for controlling shrinkage and warpage in the final part. Additional applications for surrogate models seen in literature were used for the optimization of cycle time [7] and part weight [12]. More comprehensive reviews can be found in [10, 13].

Mainly, most of the research in this field is directed towards optimizing process parameters using a surrogate model representing the simulation. For this, the simulation needs to accurately reproduce the physical phenomena, which are subject of optimization. Kurt et al. [14] highlighted the importance of the cavity pressure in prescribing the final quality of an injection molded part. However, there still exist discrepancies in predicting these pressures accurately using simulations which in turn leads to errors in the estimations of shrinkage and warpage. Therefore, in this work, we use surrogate modeling to improve the pressure estimation in the simulation by optimizing three uncertain parameters using experimental pressure sensor data. As example of the methodology, we choose to optimize the heat transfer coefficients during filling and packing as well as the pressure-dependency coefficient in the Cross-WLF viscosity model [15, 16]. The previous parameters are known to affect the pressure results in the simulation and in addition their experimental determination is challenging and time-consuming [17–19]. It is important to state that there exist additional parameters affecting the pressure field evolution in state-of-the-art injection molding simulation such as the pressure-dependency coefficient b_6 in the two-domain Tait PVT model and the value of the transition temperature, which corresponds to the solidification criterion. However, for the current work, we decided to focus on the three mentioned variables in order to test the feasibility of using surrogate modeling to improve the simulation pressure predictions.

For these purposes, we compare the accuracy of three different regression-based surrogate modeling techniques along with two design of experiment (DoE) methods in their ability to approximate the pressure signal at a single node in the simulation. The proposed modeling approaches are the following:

1. a non-linear regression model of proper orthogonal decomposition coefficients trained using a fixed Latin Hypercube sampling,
2. a polynomial chaos expansion model fitted with a least-squares optimization technique using the same fixed Latin Hypercube sampling points, and
3. a polynomial chaos expansion model fitted with a least-squares optimization technique but using an adaptive DoE that is enriched while generating the model.

The models are trained using 30, 60 and 120 simulation runs and their predictions are assessed accordingly. The three models are built by varying a total of six input parameters, including process settings and modeling variables. The process parameters are the injection velocity, coolant inlet temperature, holding pressure and the modeling parameters are the heat transfer coefficients during filling and packing as well as the pressure-dependent coefficient in the Cross-WLF viscosity model. After obtaining the surrogate models, we perform a sensitivity analysis to study the contribution of each varied parameter to the pressure signal as well as their interaction with each other. Finally, using the measured experimental pressure data and one of the generated surrogate models, we calibrate the modeling parameters and compare the error between the optimized simulation and the default one.

Theoretical background

The construction of a surrogate model consists of multiple steps. The basic process can be summarized as follows [20–22]:

1. *Design Variables Choice*: Selection of variables, which presumably have a non-negligible impact on the model output, this choice is usually supported by preliminary experiments, whether physical or numerical experiments.
2. *Design Space Sampling*: Definition of a sampling plan also referred to as design of experiments and evaluation of the respective design points by means of a high-fidelity simulation or actual experiments.

3. *Surrogate Model Generation*: Selection of a type of surrogate model in accordance with the problem at hand and the construction of the model by fitting the data obtained at the chosen points in the design space.
4. *Model Validation*: Checking of the accuracy of the generated model according to a predefined statistical criteria, in case of unsatisfactory results, identification of new design points for further model enrichment.
5. *(Optional) Model Updating*: Building an updated surrogate model using the additional design points along with the previous ones.
6. *Model Exploitation*: Use of the generated surrogate model for further analyses such as parameter sensitivity analysis, optimizations or uncertainty quantification.

Design of experiments

Design of experiments or sampling plan are terms used to represent the physical or computational experiments needed to be ran in order to capture the behavior of an underlying system over a limited number of variables [21, 23]. To build a surrogate model of a system, it is crucial to cleverly select the design points in order to cover the complete design space using the least possible number of samples. There exist two main categories of DoEs: classical and modern DoEs. Classical DoE methods are typically used for physical experiments since these experiments are non-repetitive due to the presence of random error sources. As for modern DoEs, they are commonly applied to deterministic computer experiments where systematic errors are mostly involved. In this work, two different modern DoE methods are utilized, Latin Hypercube sampling and Quasi-Monte Carlo sampling.

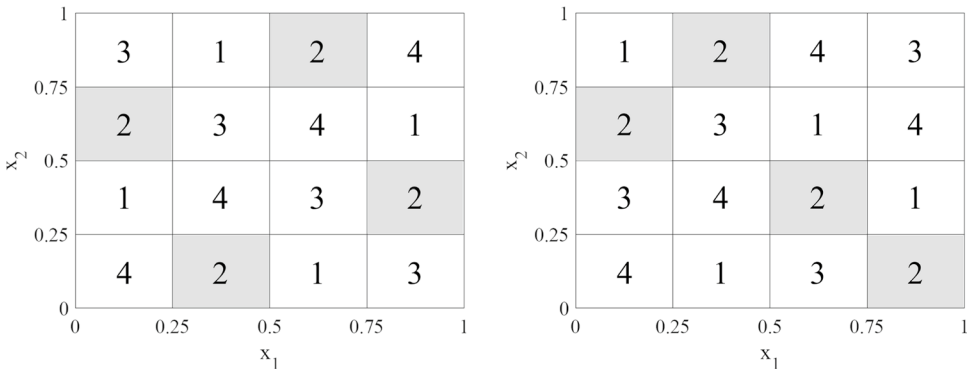
Latin hypercube sampling

The Latin hypercube sampling (LHS) technique is widely used in computational applications as it can work with any sample size which gives the user the freedom to choose the number of samples according to the available computational resources [24, 25]. LHS is a space-filling method such as it tries to find a design that fills the design space given a specific number of samples. To demonstrate the technique, let us consider a 2-dimensional design space $\Omega = [0,1]^2$ with variables having uniform probability distribution functions. If the desired number of sample points is, for example, $N = 4$, then the design space is divided into 4^2 equally sized cells. Next, values from 1 through N are placed in each row so that no two columns have an integer repeating, similar to Sudoku [24, 25]. Figure 1 shows two of the many possible arrangements. Next, a random integer between 1 and N is selected to specify the N cells in which a sample point is randomly picked. In the examples shown in Fig. 1, the chosen integer is 2 such as the shaded cells represent the sampling sites.

The Latin square presented in Fig. 1 generalizes to a Latin hypercube for higher dimensional ($n > 2$) design spaces. For instance, a design space with $n = p$ design variables requiring N samples points will form N^p hypercubes [24]. It is also worth noting that there can exist some cases, such as diagonal arrangements, for which the chosen sample sites are not optimally positioned and do not fill the design space. One way to eliminate such an arrangement is by introducing additional conditions to check the minimum distance between the design points and selecting the largest one [24].

Quasi-Monte Carlo sampling

Quasi-Monte Carlo (QMC) sampling is the deterministic counterpart of the classical Monte Carlo (MC) sampling method [26]. In classical MC, sample points are randomly selected in the design space for a given interval which can lead to having some regions of the design space unexplored [27]. Therefore, to overcome this shortcoming, QMC methods were developed to provide deterministic sample points with the optimal spread over the design space. These methods are also known as low-discrepancy sequences since they fill the space with some uniformity in order not to leave big gaps [24]. There exist various number of such sequences



(a) LHS example arrangement 1 (b) LHS example arrangement 2

Fig. 1 Latin Hypercube sampling in a 2-dimensional design space $\Omega = [0,1]^2$ with a set of sampling sites denoted by integers from which the shaded cells are the selected sampling sites

such as the Halton sequence [28], the Niederreiter sequence [29], the Sobol sequence [30]...

Many of the mentioned sequences are based on the van der Corput one, which is the simplest one-dimensional low-discrepancy sequence. One can refer to [31] for detailed explanation concerning this sequence and its generalizations. For this work, the Sobol sequence is mainly used to obtain quasi-random DoE samples. In this type of sequence, the prime number 2 is used as the base for all the dimensions of the sequence. The first dimension is the van der Corput sequence with a base of 2 and the higher dimensions are permutations of this first dimension [24, 30]. Figure 2a shows an example of 1000 sample points obtained using QMC with the Sobol sequencing method in comparison to those gotten using LHS in Fig. 2b for a 2-dimensional design space $\Omega = [0,1]^2$ with variables having uniform probability distribution functions.

Surrogate modeling techniques

In the following, only those modeling techniques that are used in this work are discussed in detail. For this discussion, let us consider Y to be one model output and $\mathbf{X} = (X_1,\cdots,X_p)$ the set of input parameters defined in the design space D_{X_i} such as:

$$Y = f(\mathbf{X}) = f(X_1,\cdots,X_p) \approx F(X_1,\cdots,X_p) \tag{1}$$

where the function $f(\mathbf{X})$ represents the system or simulation output that is being approximated using a surrogate model $F(\mathbf{X})$ and p the parameter space design size. These notations are used to describe the methods presented in this section for the case of one output result which can be extended to the multi-output case.

Proper orthogonal decomposition

Model order reduction techniques provide an efficient way to generate surrogate models since they work on the discretization of a state equation's dimensionality instead of on its design space [22]. In this section, the main focus is on presenting the Proper Orthogonal Decomposition (POD) technique, a commonly used model order reduction method. It is known by different names such as the Karhumen-Loève (KL) expansion as well as Principle Component Analysis (PCA) and an extension of the Singular Value Decomposition [32].

POD is a mathematical procedure that provides orthonormal basis functions, known as empirical eigenvectors, in order to obtain a simplified representation of a set of data or a state's evolution [22, 33]. These obtained eigenvectors correspond to the highest eigenvalues and they represent the basis functions that are able to describe the main modes or events involved in a certain state evolution [33]. Let $\mathbf{M}$ be a model state variable considered in a specific system. The variable can be represented at specific time steps and for the different nodes present using a matrix A_M such as:

$$\mathbf{A_M} = \begin{pmatrix} M_1^1 & M_1^2 & \cdots & M_1^\kappa \\ M_2^1 & M_2^2 & \cdots & M_2^\kappa \\ \vdots & \vdots & \vdots & \vdots \\ M_\nu^1 & M_\nu^2 & \cdots & M_\nu^\kappa \end{pmatrix} \tag{2}$$

where κ is the number of time steps and ν is the number of nodes. The main objective of the POD technique is to find a set of orthogonal basis functions ϕ_i $(i = 1,\cdots,\kappa)$ able to describe the deviation $\hat{M}_i$ of the model variables from their mean $\bar{M}_i$. These basis functions can be represented through a linear relation with the snapshots as follows:

$$\boldsymbol{\phi} = \sum_{i=1}^{\kappa} a_i \hat{M}_i. \tag{3}$$

To obtain the POD basis vectors, also known as POD modes, the following eigenproblem needs be solved to obtain a

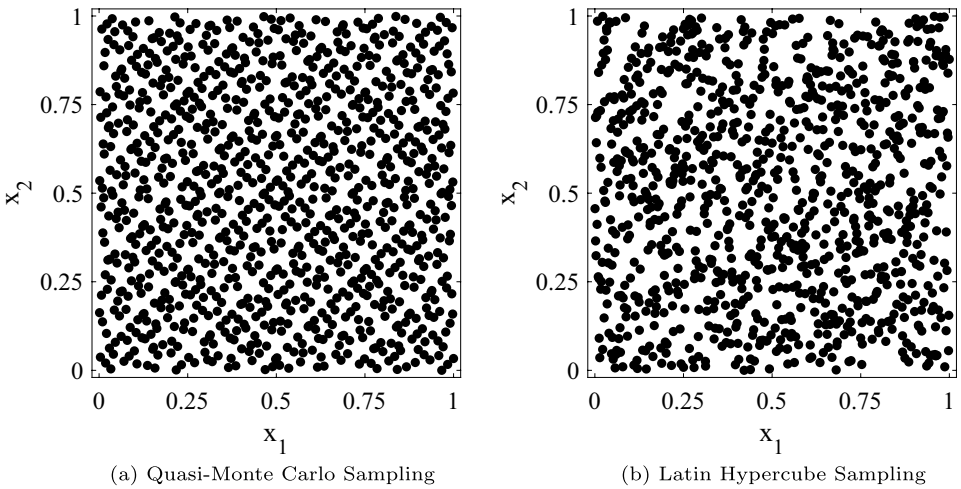


Fig. 2 Comparison of 2-dimensional design spaces $\Omega = [0,1]^2$ with 1000 sample points obtained using (**a**) QMC, (**b**) LHS

subspace with a low dimension m able to provide a good approximation of the true data:

$$\mathbf{A}\mathbf{A}^T\phi = \lambda_i\phi \tag{4}$$

with λ_i as the eigenvalues. The state variable M can be then represented using a linear combination of the calculated POD basis functions with the following relation:

$$M(x,t) = \bar{M} + \sum_{i=1}^{m} \alpha_i(t)\phi_i(x) \tag{5}$$

where $\alpha_i(t)$ are the POD coefficients [34]. More in-depth information concerning POD and model order reduction can be found in [22, 32, 34].

The POD model by itself is unable to approximate the state variable at sites not included in the original data set [22]. Therefore, the next step is to generate a surrogate model to predict the POD coefficients. For this, the calculated coefficients are used to train a surrogate model. This is done by first choosing a model to fit the POD parameters such as Kriging, radial basis functions, polynomial functions then use an appropriate fitting method like least-squares regression, best linear unbiased predictor to determine the surrogate model's coefficients. As such, one can obtain a POD-based surrogate model. Simpson et al. [35] as well as Wang and Shan [20] gave detailed reviews on the equations and fitting techniques used for various surrogate modeling methods.

Polynomial chaos expansion

Polynomial chaos expansions (PCE) are surrogate modeling techniques that expand $f(\mathbf{X})$ using a series of multivariate basis functions orthogonal to the probability density function g_{X_i} of an input variable X_i [36]. In general, polynomial chaos expansions can be considered as a special case of KL or PCA since orthogonal polynomial families are used to describe the basis functions instead of eigenvectors. The resulting random model response Y is assumed to have a finite variance and thus belonging to the so-called Hilbert space, allowing for the following spectral representation [37]:

$$Y = \sum_{j=0}^{\infty} c_j\psi_j. \tag{6}$$

The random variable Y is an infinite series, where $\{\psi_j\}_{j=0}^{\infty}$ are a set of countable random variables forming the basis of the Hilbert space and $\{c_j\}_{j=0}^{\infty}$ are the coefficients representing the coordinates of Y in this basis [37]. With the assumption that the input variables are independent, an inner product can be defined for each variable X_i with any two functions φ_1,φ_2 such as:

$$\langle\varphi_1,\varphi_2\rangle_i = \int_{D_{X_i}} \varphi_1(x)\varphi_2(x)g_{X_i}(x)dx. \tag{7}$$

When replacing the arbitrary functions in Eq. 7 with orthogonal polynomials $P_k^{(i)}$:

$$\langle P_j^{(i)}, P_k^{(i)}\rangle_i = \int_{D_{X_i}} P_j^{(i)}(x)P_k^{(i)}(x)g_{X_i}(x)dx = a_j^{(i)}\delta_{jk} \tag{8}$$

where k is the polynomial degree and δ_{jk} is the Kronecker symbol equal to 1 for $j = k$ and 0 otherwise. As for the term $a_j^{(i)}$, it is the squared norm of $P_j^{(i)}$ and is equal to 1 for orthonormal polynomials:

$$a_j^{(i)} = ||P_j^{(i)}||_i^2 = \langle P_j^{(i)}, P_j^{(i)}\rangle_i. \tag{9}$$

The family of orthonormal polynomials $\{\psi_j^{(i)}\}$ is obtained by normalizing the $P_j^{(i)}$ functions :

$$\psi_j^{(i)} = P_j^{(i)}/\sqrt{a_j^{(i)}}, \quad i = 1,2,\cdots,p. \tag{10}$$

Depending on the distribution of the input variables, there exist well-known orthogonal polynomial families. For example, if X_i has a uniform distribution, the corresponding family is the Legendre polynomials or if X_i has a Beta distribution then the Jacobi polynomials constitute the basis functions for PCE [38].

In order to estimate the polynomial chaos coefficients, there exist various intrusive (Galerkin projection) and non-intrusive methods. Popular non-intrusive techniques are error minimization ones which solve a least squares (LS) or least angle regression (LAR) problem [24]. For more detailed description concerning PCE, the reader is advised to refer to [36–38].

Variance-based sensitivity analysis

Variance-based sensitivity analyses are used to quantify the variance contribution of an input parameter to the unconditional variance of the model output [39]. A commonly utilized method in this field is the Sobol method [40]. An attractive feature of this technique is its ability to not only measure the amount of variance caused by one input but also the interaction of two or more inputs and their contribution to the output. These are known as the Sobol sensitivity indices. The method utilizes approximate Monte Carlo integration to calculate the different indices [41].

The Sobol approach decomposes the function f into terms of increasing dimensionality [39]

$$f(\mathbf{X}) = f_0 + \sum_{i=1}^{p} f_i(X_i) + \sum_{i=1}^{p}\sum_{j=i+1}^{p} f_{ij}(X_i,X_j) + \cdots + f_{1,\cdots,p}(X_1,\cdots,X_p) \tag{11}$$

such as each successive term represents the increasing degrees of interactions between the various parameters. The total variance $V(Y)$ can be then defined using the sum of the partial variances through a similar decomposition to Eq. 11 with the assumption that the parameters are mutually orthogonal:

$$V(Y) = \sum_{i=1}^{p} V_i + \sum_{i=1}^{p-1} \sum_{j=i+1}^{p} V_{ij} + \cdots + V_{1,\cdots,p}. \tag{12}$$

As such, the Sobol sensitivity indices can be formulated as follows:

$$S_i = \frac{V_i}{V} \tag{13}$$

for the first order sensitivity indices and

$$S_{ij} = \frac{V_{ij}}{V} \tag{14}$$

for the second order sensitivity indices.

Method implementation

Software tools

Injection molding simulation

The injection molding simulations are performed using Autodesk® Moldflow® Insight 2021.1 (AMI2021.1). The simulations are automated with the help of the three command line control functions. This set of utilities allow the use of third-party software in tandem with the simulation runs in order to automatize extensive studies such as optimizations and sensitivity analyses... The main functions are editing, running, and retrieving results from a user-customized Moldflow analysis automatically through a command line interface. The following are the available utilities:

1. *studymod*: This command generates a modified simulation study from a base study using an XML format modifier file (modified geometry, boundary condition, process/model parameters).
2. *runstudy*: This command launches a new simulation analysis.
3. *studyrlt*: This command extracts simulation results from a finalized analysis in simple text or XML file format.

Python uncertainty quantification library

The Python Uncertainty Quantification (pyUQ) library is a Bosch proprietary Python tool providing the user the ability to perform uncertainty quantification (UQ) on simulation models using state-of-the-art UQ models and methods. The methods cover the design of experiments, surrogate modeling, sensitivity analysis and statistical analysis. The pyUQ module provides functions that can be modified or extended to act as an interface between simulation tools and UQ methods. Parameter uncertainties are defined in this tool using standard distributions (uniform, normal, Gamma, Beta) by stating the parameter's upper and lower limits. The module offers four basic modern DoEs such as Standard-Monte Carlo, Quasi-Monte Carlo, Latin Hypercube sampling and more refined sparse grid experimental methods like Non-Intrusive Spectral Projection. There are five different surrogate modeling techniques that this library supports:

1. Polynomial Chaos Expansion (PCE) using least square (LS) as optimization methods,
2. PCE using least angle regression (LAR) as optimization method,
3. Gaussian Process Regression (GPR),
4. Pseudo Spectral Projection method (PSP),
5. Stochastic Collocation method.

The simulation results are stored in a Hierarchical Data Format (HDF5) to be able to efficiently perform statistical analysis or surrogate model generation using this tool. HDF5 is a folder-like storage structure, which saves data in compressed format and allows data slicing leading to efficient memory usage. Apart from these features, the main highlight of this tool is its active learning algorithm with an adaptive DoE generation which is one of the methods evaluated in this work.

MATLAB

The POD model order reduction and the non-linear regression of the respective POD-based coefficients is implemented in MATLAB R2019b. The fmincon built-in function of the optimization toolbox is used for the optimization of the POD-based surrogate model.

Surrogate models set-up

Physical experiments

A simple mold geometry is used for varying several process settings on an electrical injection molding machine (ENGEL E-Motion 440/220 T). The geometry sketch along with the design dimensions are shown in Fig. 3. The mold has three p-T-sensors to measure pressure and temperature during the injection molding process. The sensors' location is specified by the gray circles in Fig. 3. An unreinforced polyoxymethylene homopolymer (POM) is used for the injection

molding trials, in which the mold temperature, the injection velocity and the holding pressure are varied. The upper and lower bounds of these variables are presented in Table 1. A design of experiments consisting of 27 sampling points is performed. The sensor data collected during the actual molding trials were used in the optimization process to identify the heat transfer coefficient values and the viscosity's pressure-dependency coefficient.

Table 1 Variable process settings in the experimental runs with their upper and lower bounds

Process Settings	Lower Bound	Upper Bound
Injection Velocity, V_{inj} (cm^3/s)	10	50
Mold Temperature, T_{mold} (°C)	80	110
Holding Pressure, P_{hold} (MPa)	20	80

Simulation environment

The 3D high-fidelity simulation model was set up in AMI2021.1 using the respective POM material card given in the Moldflow database. The simulation consists of a cooling, fill and pack analyses. Therefore, the simulation model included cooling channels meshed as beam elements and the part as well as the feed system (nozzle & flange) meshed using tetrahedral elements with 24 layers through the thickness. The feed system is defined as hot runner in the simulation model. Lateral and top views from the meshed model are shown in Fig. 4a and b, respectively. A single simulation requires around 75 minutes to be completed on a workstation with a 4.10 GHz processor and 32 GB RAM.

The various surrogate models are obtained by modifying six parameters in the simulation. Three of which are the process parameters changed during the experimental trials as presented in Table 1. Since a cooling analysis is performed in the simulations, the inlet cooling temperature is varied such as $T_{c,in} = T_{mold} + 4$ of the T_{mold} set experimentally. The other three are the heat transfer coefficient during filling ($HTC_{filling}$) and during packing ($HTC_{packing}$) as well as the pressure-dependency coefficient (D_3) of the Cross-WLF viscosity model (refer to Appendix A). Table 2 summarizes the surrogate models' variables and their bounds. The HTC ranges are chosen in reference to the default Moldflow values ($HTC_{filling} = 5000 W/m^2°C, HTC_{packing} = 2500 W/m^2°C$). As for the pressure-dependent viscosity model parameter, the variation range goes from 0 up to 0.4 K/MPa in order to cover the actual behavior of POM and other typical semi-crystalline thermoplastics such as PP [42].

Proposed DoE and surrogate modeling techniques

In this work, three different combinations of DoEs and surrogate modeling methods are used in order to compare their performance in predicting the pressure signal results at one surface node corresponding to the location of the second sensor in Fig. 3. Additionally, the total number of simulations used for the training phase is varied such as 30, 60 and 120 simulation runs are considered and their performance is tested using four additional runs. The testing simulations are summarized in Table 3.

POD-NLR The POD-based surrogate model whose parameters are fitted using non-linear regression (NLR) is referred to as POD-NLR. To generate this model, LHS is used to obtain the DoEs according to the needed total number of training simulations S. The obtaining of the final model requires the following steps after running all the simulations according to the DoE:

1. *Pre-processing*: the various nodal pressure results P_i ($i = 1, \cdots, S$) are pre-processed by shifting all corresponding time series by t_{start} so that all non-zero signals start

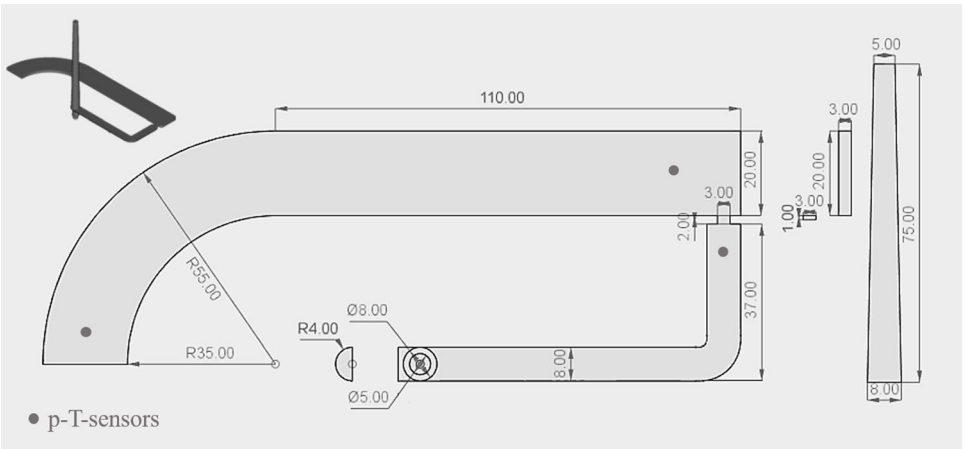
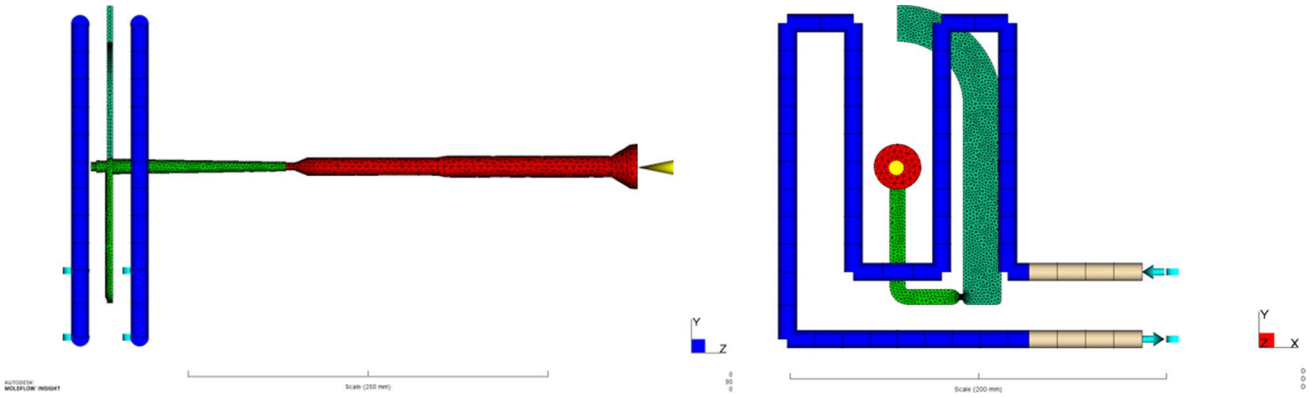


Fig. 3 Sketch of the injection-molded part including some characteristic dimensions in mm and the location of the three pressure and temperature combination sensors (p-T-sensors)



(a) Lateral view (b) Top view

Fig. 4 The meshed simulation model including the part (dark green), runner and sprue (light green), cooling channels (blue) and feed system (red)

Table 2 Varied parameters in the simulation runs used to generate the surrogate models along with their upper and lower bounds

Surrogate Model Variables	Lower Bound	Upper Bound
Injection Velocity, V_{inj} (cm^3/s)	10	50
Coolant Inlet Temperature, $T_{c,in}$ ($^{\circ}C$)	84	114
Holding Pressure, P_{hold} (MPa)	20	80
Heat transfer coefficient (filling), $HTC_{filling}$ ($W/m^{2\circ}C$)	5000	10000
Heat transfer coefficient (packing), $HTC_{packing}$ ($W/m^{2\circ}C$)	1000	5000
Viscosity model parameter, D_3 (K/MPa)	0	0.4

Table 3 Definition of the testing simulations for evaluating the performance of the surrogate models

DoE Name	V_{inj} (cm^3/s)	T_{mold} ($^{\circ}C$)	P_{hold} (MPa)	$HTC_{filling}$ ($W/m^{2\circ}C$)	$HTC_{packing}$ ($W/m^{2\circ}C$)	D_3 (K/MPa)
Test 1	36.0	94.2	55.1	5175	4522	0.33
Test 2	15.5	83.3	74.7	9751	2143	0.07
Test 3	45.6	105.4	47.5	8092	3945	0.21
Test 4	27.2	102.2	25.9	7338	1640	0.12

at $t_0 = 0$ s and followed by the resampling of the pressure data using a common time vector from 0 to 50 s with 0.02 s time steps. This step is important given that the different simulation settings lead to different times at which the flow front reaches the sensor node and produces a non-zero pressure signal.

2. *Model order reduction*: the POD basis functions ϕ are calculated by solving the eigenvalue problem $PP^TV = \lambda V$ where the eigenvectors V corresponding to the most influential eigenvalues λ constitute the basis functions able to reconstruct the pressure signal by:

$$P^{(s)}_{reconstructed} = \sum_{i=1}^{n} \Gamma^{(s)}_n \phi_n \tag{15}$$

where s is the simulation number, n is the number of modes or basis obtained according to a specified error value and $\Gamma^{(s)}_n$ is the POD parameter for a specific mode n and simulation s. The truncation criterion is done according to an error value of $\epsilon = 10^{-3}$ which only selects the modes corresponding to eigenvalues that fulfill the condition $\frac{\lambda}{\lambda_{max}} \geq \epsilon$.

3. *Model fitting*: a least-squares regression of a second-order polynomial is used to train a model to predict the POD parameters Γ and the time shift value t_{start} as follows:

$$\Gamma^{(s)}_n = a^{(n)} + b^{(n)}_i X^{(s)}_i + c^{(n)}_{ij} X^{(s)}_i X^{(s)}_j \tag{16}$$

$$t_{start}^{(s)} = \alpha + \beta_i X_i^{(s)} + \gamma_{ij} X_i^{(s)} X_j^{(s)} \quad (17)$$

where X_i and X_j are the surrogate model input variables with $i \neq j$ representing the number of variables $(1,\cdots,6)$ and $a,b,c,\alpha,\beta,\gamma$ are the surrogate parameters. In Eqs. 16 and 17 the implicit Einstein summation convention is used for indexes i and j. This implementation is restricted to bilinear regression in order to mitigate overfitting the data.

Regular-PCE-LS The same LHS DoEs used to train the POD-NLR model are employed as inputs to a second-order PCE-based surrogate model. The model parameters are trained using least-squares regression and thus referred to as regular-PCE-LS since the DoE is fully predefined. The PCE model is generated using the pyUQ library using the following steps:

1. *Pre-processing*: similar to POD-NLR, a resampling step is performed to have a unique time index starting with $t_0 = 0$ s and spanning to 50s with 0.02 s time steps.
2. *Polynomial chaos expansion*: since the surrogate model input variables have a uniform distribution, after applying the Gram-Schmidt orthogonalization to Eq. 8, the Legendre polynomial family is obtained and thus used as basis functions to represent the pressure signal results from the simulation. A hyperbolic truncation of 1 is utilized to generate the model which signifies that all interaction effects are considered in the PCE basis. Additionally, the PCE regression order is estimated by pyUQ using heuristics and in this case a second-order regression is used.
3. *Model fitting*: the algorithm uses the pressure time-dependent training data to solve an optimization problem which minimizes the squared residuals between the simulation's pressure signals and the predicted output of the PCE model. The solution of the least-squares optimization problem is the vector containing the polynomial coefficients.

Adaptive-PCE-LS A sequential design of experiment (SDOE) is used to build iteratively a DoE from a big candidate DoE obtained with QMC according to the importance of the input variables. The surrogate model utilizes PCE and is trained using least-squares regression. This model is generated using the pyUQ library and is referred to as adaptive-PCE-LS since its DoE is enriched as the model is being generated making it an adaptive approach. Therefore, given a sample set, the surrogate model generation follows the same steps presented in regular-PCE-LS. The main difference between the two methods lies in the iterative steps needed to build the whole surrogate model by updating the DoE.

The algorithm starts with an initial small LHS DoE of 10 simulations and is enriched after each iteration with 5 additional sample points from a large QMC candidate DoE (1000 samples). The enrichment is done via the bootstrap method [43] which calculates the local variances of the surrogate model and chooses the new sample points that lead to the maximum variance. In other words, it performs a UQ analysis on the simulation results to estimate which parameters contribute most to the statistical fluctuations. This process is repeated until convergence in terms of maximum simulations S of 120 or a global error threshold E of 10^{-2} is achieved.

The surrogate model is assessed by cross-validation using the training data set with the leave-one-out error method such as the model is generated from $S-1$ simulations and evaluated on the remaining one simulation result. The standard error metric used to evaluate the surrogate model after each iteration is its global error. This error value is an average of a scalar model output predefined by the user. In this work, the selected scalar quantity corresponds to the mean value of the pressure vector at the studied node.

The flowchart in Fig. 5 presents a summary of the steps followed while generating the adaptive-PCE-LS surrogate model using an active learning/adaptive approach.

For the sake of clarity, the characteristics of the three studied surrogate models are summarized in Table 4.

Evaluation strategy

To evaluate and compare the various surrogate models, two error metrics are considered: the root mean squared error (RMSE) and the coefficient of determination (R^2 score).

Let the surrogate model response and the true values of the high-fidelity simulation model or experiment be represented by $\hat{Y}$ and Y, respectively. Using these notations, the RMSE is computed by evaluating the surrogate model pointwise on a given test data set, such as:

$$RMSE(Y,\hat{Y}) = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(Y^i - \hat{Y}^i)^2} \quad (18)$$

where n is the total number of data points. In this work, this error is normalized using the min-max normalization method to be able to compare the various surrogate modeling techniques.

The R^2 score is a statistical measure that indicates how well the data fit the regression model and how well the unseen samples are likely to be predicted by the model. It determines the proportion of variance of the model response that can be explained by its independent variables. The best

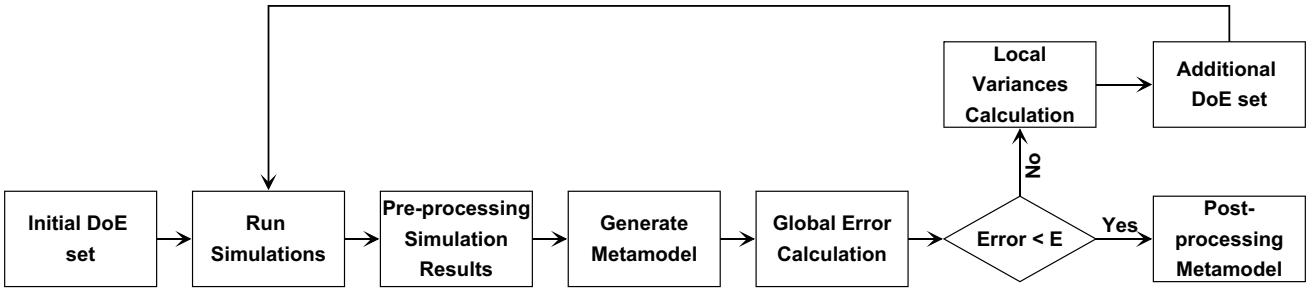


Fig. 5 Workflow for generating an adaptive surrogate model using the pyUQ active learning algorithm

Table 4 An overview of the proposed DoE and surrogate modeling techniques

Surrogate model Acronym	DoE Technique	Surrogate model Technique	Fitting Method
POD-NLR	Latin Hypercube	Proper orthogonal decomposition	Least-squares regression
Regular-PCE-LS	Latin Hypercube	Polynomial chaos expansion	Least-squares regression
Adaptive-PCE-LS	Sequential DoE (LHS & QMC)	Polynomial chaos expansion	Least-squares regression

score is 1.0 and has a range between $-\infty < R^2 \leq 1$. A negative score means that the surrogate model response is unable to predict accurately the output results. R^2 is calculated as follows:

$$R^2(Y, \hat{Y}) = 1 - \frac{\sum_{i=1}^{n}(Y^i - \hat{Y}^i)^2}{\sum_{i=1}^{n}(Y^i - \bar{Y})^2} \tag{19}$$

where $\bar{Y} = \frac{1}{n}\sum_{i=1}^{n} Y^i$ is the mean value of Y.

Optimization

In this work, an optimization routine is performed to identify the uncertain simulation model parameters: $HTC_{filling}$, $HTC_{packing}$ and D_3. For these purposes, the sensor data from the 27 experimental runs presented in "Physical experiments" section are utilized. The optimization problem aims at minimizing the difference between the experimental results Y^{exp} and the surrogate model predictions $\hat{Y}$ by varying the respective uncertain input parameters of the surrogate model. It can be represented as follows:

$$\begin{aligned}
&\text{find} \quad Var = (HTC_{filling}, HTC_{packing}, D_3),\\
&\min \quad \sum_{i=1}^{27} \|Y_i^{exp}(\mathbf{V}) - \hat{Y}_i(\mathbf{X})\|,\\
&\text{with} \quad \mathbf{V} = (V_{inj}, T_{mold}, P_{hold}),\\
&\qquad \mathbf{X} = (V_{inj}, T_{mold}, P_{hold}, HTC_{filling}, HTC_{packing}, D_3),\\
&\text{s.t.} \quad 5000 \leq HTC_{filling} \leq 10000 \quad (\mathrm{W/m^2{}^\circ C}),\\
&\qquad 1000 \leq HTC_{packing} \leq 5000 \quad (\mathrm{W/m^2{}^\circ C}),\\
&\qquad 0 \leq D_3 \leq 0.40 \quad (\mathrm{K/MPa}).
\end{aligned}$$

Results and discussion

Comparison of surrogate models

The performance of the three surrogate model techniques is assessed by their ability to predict the pressure signal at a surface node in simulations unseen by the model. In this work, those test simulations are presented in Table 3. Although the total simulated time is around 48 s, the main focus is on the pressure signal between 0 and 22 s. This interval includes the filling phase between 0 and ~2 s, the packing phase till ~18 s and the start of the cooling phase until the pressure goes to 0 MPa before 22 s. Therefore, all the error metrics and plots consider only this time interval.

Figure 6 presents the models' predictions obtained after using 30 training simulation runs in comparison to the true high-fidelity simulation results. At first glance, it seems that all three surrogate models are able to capture the main features of the pressure signal after only 30 training simulations. However, one can distinguish some particular discrepancies such as the inability of the POD-based model to predict the t_{start} for Test 2 and error of the adaptive-PCE for estimating the pressure at the end of packing in Tests 3 and 4. Additionally, there are deviations by predicting the pressure peak at the end of the filling phase for all surrogate models.

A quantitative analysis of the performance of the different surrogate models are given in Fig. 7a and b, where the normalized RMSE and R^2 score are plotted for the four test simulations. The average normalized RMSEs of 0.084, 0.093 and 0.119 for POD-NLR, regular-PCE-LS and

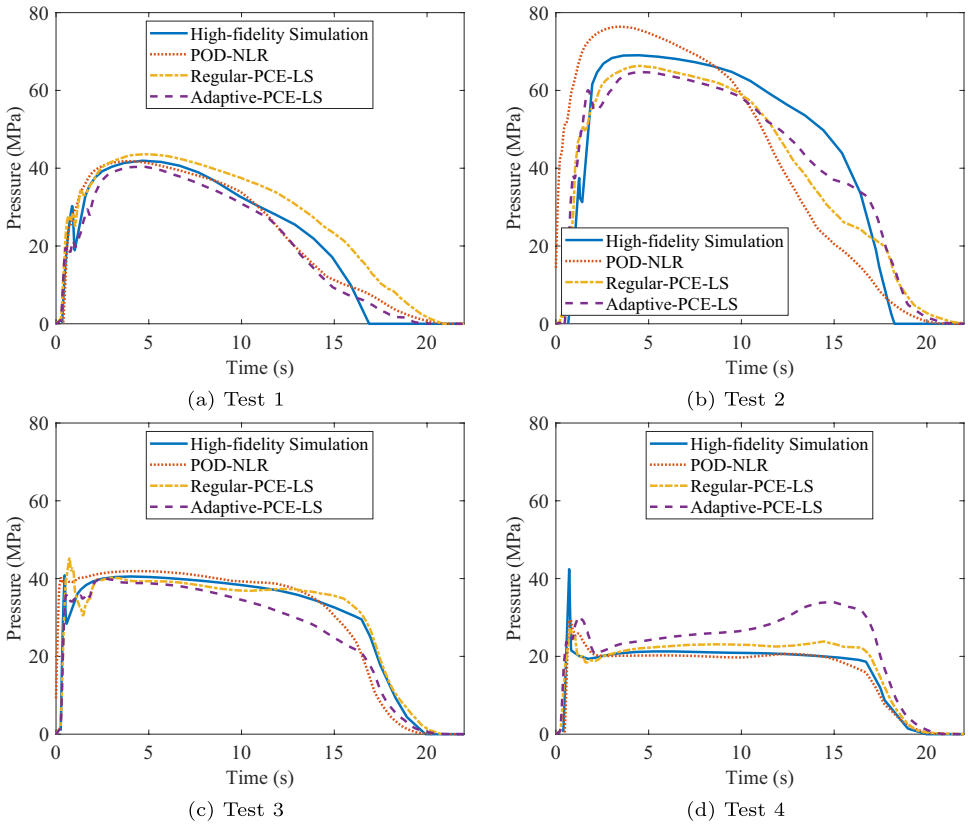


Fig. 6 Predicted pressure signals by the three proposed surrogate models after training with 30 simulations in comparison to the true high-fidelity simulation results for four test cases

adaptive-PCE-LS, respectively, show that the POD-based surrogate model is slightly more accurate in recreating the simulation results after training with 30 high-fidelity simulations. Whereas, the R^2 score shows that the adaptive-based surrogate model is not able to estimate accurately the pressure results which is especially underlined by the R^2 score of 0.16 for Test 4.

Looking to increase the prediction capability, the number of simulations used to train the surrogate models is doubled to 60. Overall, this increase in training simulations led to noticeable improvements in the pressure signal predictions of the test runs as shown in Fig. 8. For example, by comparing Figs. 6b and 8b, the POD-based model is now able to better predict the t_{start} for Test 2 as well as both PCE-based models can estimate the pressure result more accurately during the packing phase. On the other hand, for Test 4 the R^2 score of the adaptive model prediction increased slightly with the higher number of training points, but both predictions of the regular-PCE-LS and POD-NLR reduced their performance. In general, there is still room for improvements when estimating the pressure for Test 4 as well as predicting the pressure peak at the end of filling and the time at which the pressure goes to zero at the start of the cooling phase.

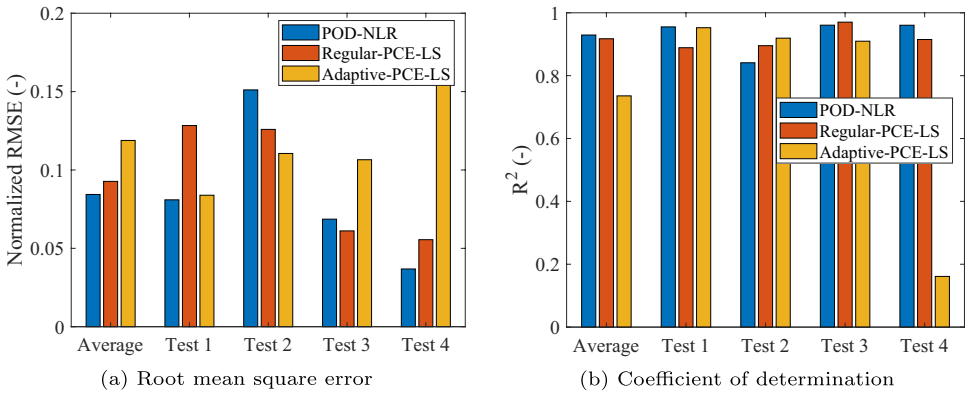


Fig. 7 Testing error metrics for the three studied surrogate modeling methods after training with 30 simulation runs

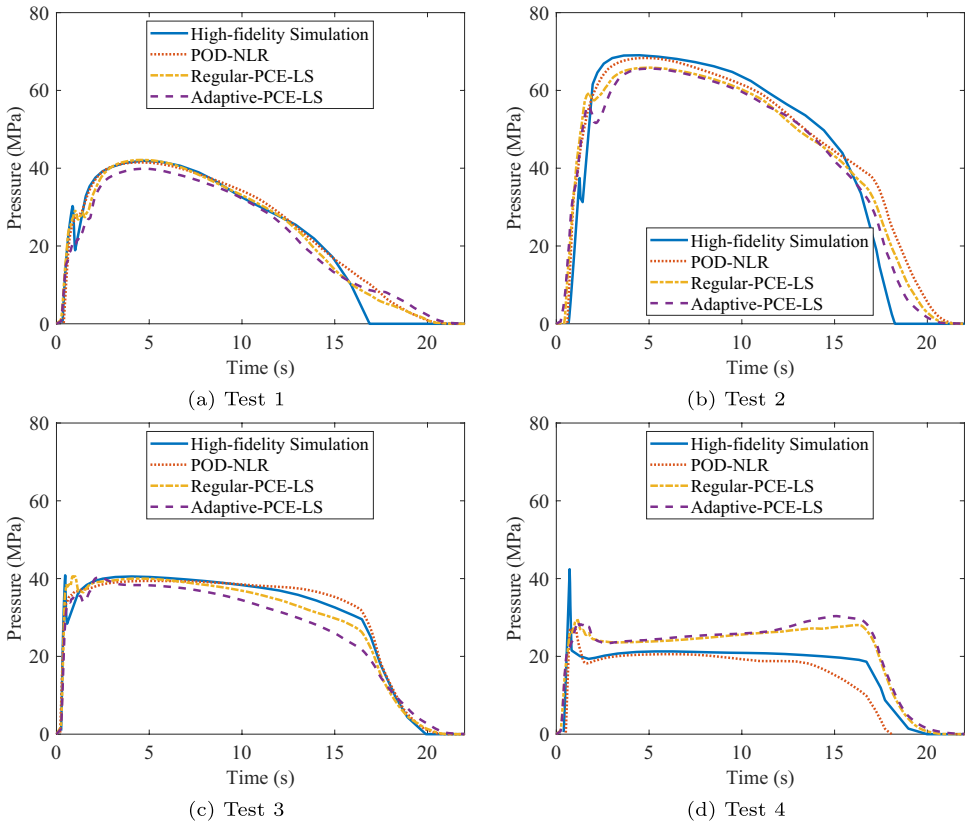


Fig. 8 Predicted pressure signals by the three proposed surrogate models after training with 60 simulations in comparison to the true high-fidelity simulation results for four test cases

Figure 9a and b show respectively the normalized RMSE and R^2 score for the three studied surrogate modeling methods after training them with 60 simulation runs. The increase in training points improved the accuracy of all models as shown by the average normalized RMSE of of 0.076, 0.082 and 0.10 for POD-NLR, regular-PCE-LS and adaptive-PCE-LS, respectively. As mentioned before, unexpectedly the RMSE for the POD-NLR and regular-PCE-LS in the Test 4 increased from 3.7 % to 9.2 % and from 5.5 % to 12.1 %, respectively, when increasing the number of training simulations.

As final attempt to improve the surrogate models' performance,, especially for the case of Test 4, 120 simulations are used to train them. Figure 10 presents the pressure evolution as a function of time for all four test simulations. By comparing with Fig. 6, it turns out that the performance of all surrogate models improved after 120 training simulations and they appear to converge to the same result. The models still struggle to predict the early pressure drop at the end of the packing phase in Test 1 and Test 2 as shown in Fig. 10a and b, respectively. Additionally, the estimation of the pressure peak in the

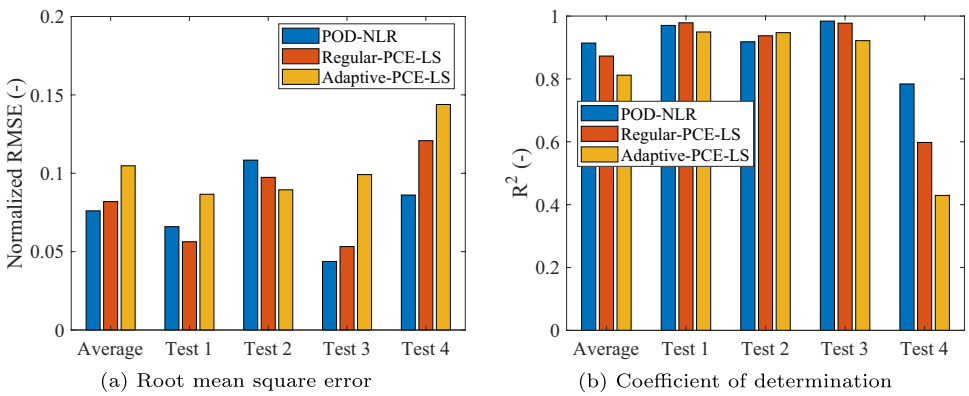


Fig. 9 Testing error metrics for the three studied surrogate modeling methods after training with 60 simulation runs

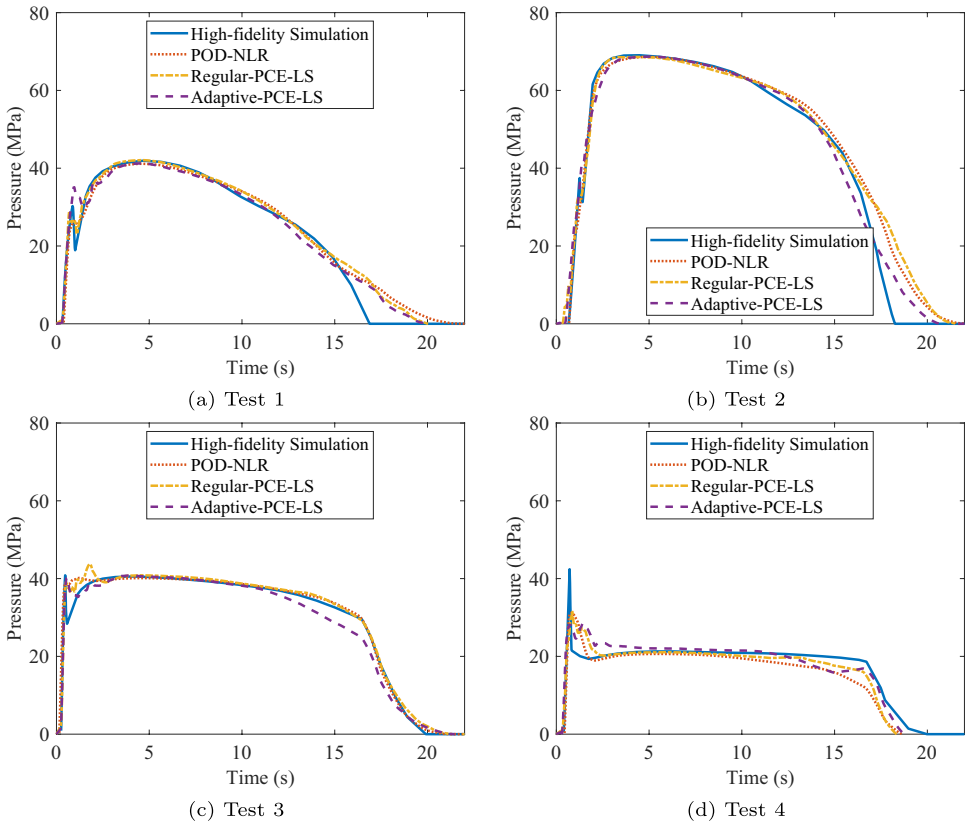


Fig. 10 Predicted pressure signals by the three proposed surrogate models after training with 120 simulations in comparison to the true high-fidelity simulation results for four test cases

filling phase is a problematic location for all models and especially highlighted in Test 4 as seen in Fig. 10d. As for the final error metrics, they are presented in Fig. 11. The RMSE of all surrogate models is lower than 8 % after training with 120 high-fidelity simulation runs. The model based on PCE-LS using an adaptive DoE exhibits the lowest average error of 5.8 % and an average R^2 score of 0.96.

Based on the previous results, all three surrogate modeling methods improved gradually their performance with the increase of the number of training simulations. This fact is depicted in Fig. 12, in which the average normalized RMSE of each model is plotted versus the number of simulations used to train it. Both the POD-NLR and regular-PCE-LS surrogate models reached an average RMSE below 10 % already after training with 30 simulations. Whereas the adaptive-PCE-LS model showed the highest average error after 30 training runs and only reached comparable low errors to the other two surrogate modeling techniques after 120 training simulations. For the two models using a fixed DoE, increasing the number of training simulations from 30 to 120 led to a relatively small decrement of the prediction error from 8.4 % to 6.3 % for the POD-based model and from 9.3 % to 6.3 % for the PCE-based one. As for the adaptive-PCE-LS model, the average RMSE decreased significantly

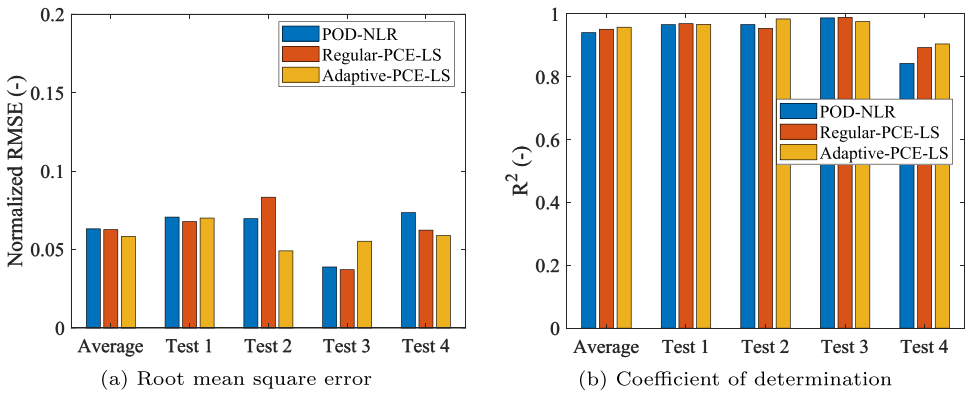


Fig. 11 Testing error metrics for the three studied surrogate modeling methods after training with 120 simulation runs

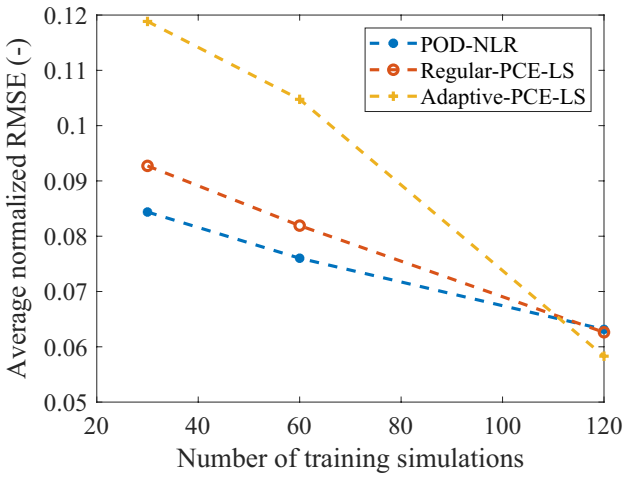


Fig. 12 Average normalized RMSE over the five test runs for each surrogate model generated as a function of the number of training simulations used

from 11.9 % to 5.8 % after training with 120 runs leading to much improved predictions.

By comparing the results of the two surrogate models based on PCE-LS, it appears that not only the number of training simulations but the choice of DoE play an important role in determining the model performance. An adaptive DoE only leads to a better surrogate model performance when using a relatively high number of training runs in comparison to another which uses a fixed random DoE. In this particular work, running 120 high-fidelity simulations necessitates around 6 days and represents an acceptable computational cost for gaining higher accuracy in front of a surrogate model generated with a fixed random DoE. However, in case of a low number of available training simulation, a pre-defined DoE is a pertinent alternative for surrogate model generation as corroborated by the results of the POD- and PCE-based models with a fixed random DoE. In terms of global performance, the POD-based model is slightly more accurate than the PCE-based models when using low number of training simulations but as this number increases the difference between the two techniques becomes negligible.

Sensitivity analysis

A Sobol sensitivity analysis is performed using the generated adaptive-PCE-LS surrogate model trained with 120 simulation runs. Figures 13 and 14 show the resultant first order and some of the second order sensitivity indices, respectively.

The first order indices underline the contribution of a certain surrogate model parameter to the variance of the pressure output as explained in "Variance-based sensitivity analysis" section. The sensitivity of the different process settings shown in Fig. 13a are in agreement with the common understanding of the injection molding process. The injection velocity is the more sensitive factor during the filling stage because it determines directly the

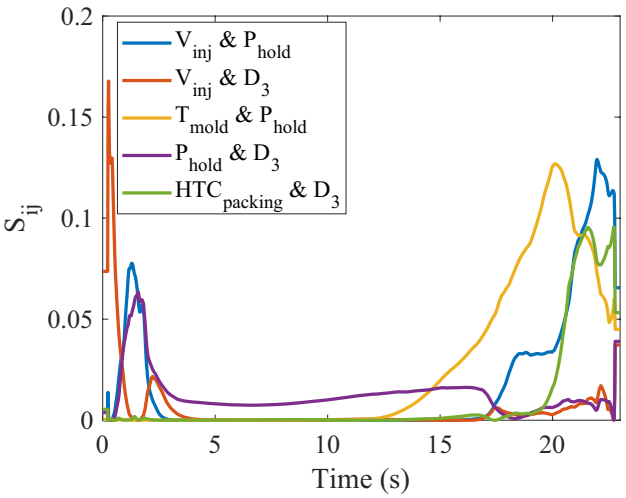
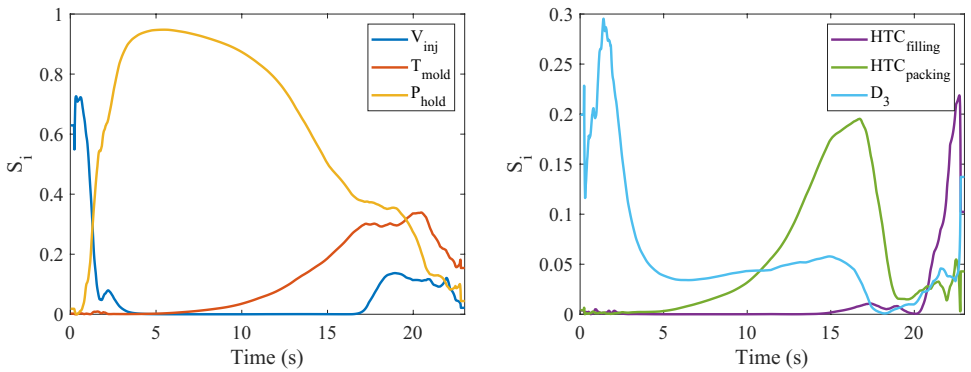


Fig. 14 The second order Sobol sensitivity indices obtained using the generated adaptive-PCE-LS surrogate model trained with 120 simulation runs

Fig. 13 The first order Sobol sensitivity indices obtained using the generated adaptive-PCE-LS surrogate model trained with 120 simulation runs



(a) Process parameters (b) Modeling parameters

pressure need. On the other hand, the holding pressure is naturally the more sensitive parameter during the packing phase, just after the switch-over till the time when the gate is completely solidified. After which the contribution of the coolant inlet temperature is the main driver of part shrinkage and therefore pressure development. The sensitivity of the HTC values and the pressure-dependent viscosity coefficient is presented in Fig. 13b. During the filling and the beginning of the packing phase, the pressure signal is highly sensitive to changes of the pressure-dependent viscosity coefficient D_3, which is in principle an expected result given the natural impact of the viscosity model in the simulation of the filing stage. As for $HTC_{packing}$, it appears to affect the pressure output mainly at the time in which the gate freezes during the packing phase. Finally, it turns out that the $HTC_{filling}$ has no effect on the pressure results during the filling phase. The sensitivity of $HTC_{filling}$ after 20 s is certainly an artifact of the surrogate model, because this parameter should not play any role at this time in the high-fidelity simulation.

Figure 14 presents some of the second order sensitivity indices representing the most important interactions between the surrogate model parameters as a function of time in relation to their combined contribution to the output pressure. During the filling phase, the main interaction is given by the injection velocity and D_3. As for the beginning of the packing phase, the most important interactions are given by the holding pressure along with both the injection velocity and D_3. Whereas, along the packing phase, it turns out that there is not significant two-parameter relations affecting the pressure development. Interestingly, at the time when pressure strongly drops during the packing phase, probably associated to the freezing of the gate, pressure development is controlled by the interaction between the holding pressure and the coolant inlet temperature. The interaction between $HTC_{packing}$ and D_3 after 20 s is difficult to explain in terms of polymer physics, but eventually reveals some subtleties caught in the complexity of the simulated phenomena.

In general, such sensitivity analyses are helpful in understanding the effect of parameters on specific output results. One way they can play an important role in surrogate model generation is by pinpointing the most influential parameters at the regions in which the model exhibits the large errors. For example, at the end of the filling phase, the models are not fully capable to predict the correct pressure drop, as can be seen in Fig. 10c and d. A possible way to improve these predictions is by adding DoE points where we mainly vary the injection velocity, holding pressure and D_3 as their first and second order sensitivity indices show their high contribution to the output in this region.

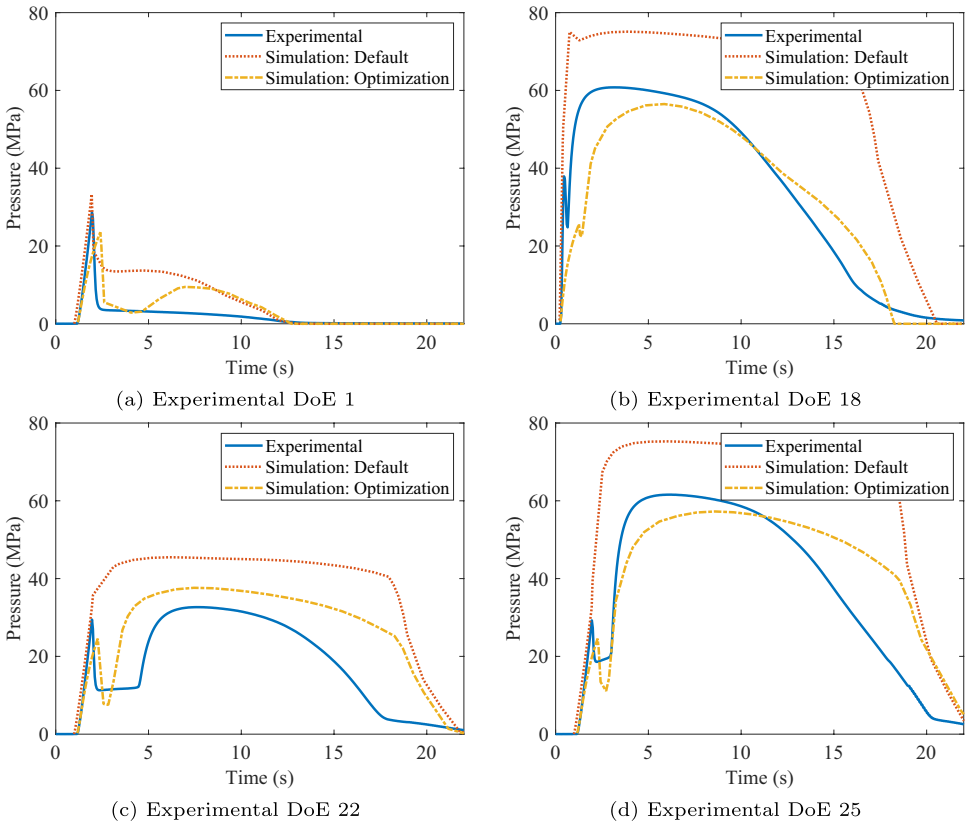
Simulation parameters optimization

All three models performed similarly after training with 120 high-fidelity simulations as shown in Fig. 12. The POD-based surrogate model is used for the parameter optimization as the coupling of this particular model implementation with the optimization algorithm presented in "Optimization" section was straightforward in comparison with the other surrogate model implementations. The goal of this optimization is to identify the uncertain simulation model parameters: $HTC_{filling}$, $HTC_{packing}$ and D_3. The optimized parameters are summarized in Table 5 along with the default Moldflow values for the $HTC_{filling}$ and $HTC_{packing}$ as well as the pressure-dependent viscosity parameter given in the Moldflow material card (D_3) for the utilized POM material.

One of the main objectives of this optimization is the identification of material parameters or boundary conditions for enabling a more accurate estimation of the pressure field in a high-fidelity injection molding simulation. To show the impact of this parameter calibration, the experimental DoE is simulated using both the optimized parameters and the default parameters given in Table 5. The simulated pressure signals at the sensor location (taken from a surface node) are compared to the experimental pressure signals. Four example DoEs are shown in Fig. 15. In all cases, the simulation using the optimized parameters is closer to the experimental data than using the default parameters. In the experimental setting showed in Fig. 15a, the cavity was actually not filled completely (short shot) and both default and optimized simulations were not able to predict this event. This failure in the simulation can be related to the default solidification criterion, which defines the melt-to-solid transition at a constant temperature and oversimplifies the crystallization phenomena occurring in a wide range of cooling rates. In any case, the simulation results using the optimized parameters show more accurate pressure estimations during the packing phase as seen in Fig. 15b and d. However, the estimation of the filling time and maximum pressure at the end of filling are less accurate than those obtained with default parameters. This issue is certainly due to the high value of the optimized pressure-dependent viscosity coefficient, which leads to higher viscosities and thus longer filling times. The prediction of the time when the pressure goes to zero is also prone to further improvement for both default

Table 5 The model and boundary condition parameters used in the default simulation and the ones obtained in the optimization use case

Simulation	$HTC_{filling}$ (W/m$^{2\cdot}$C)	$HTC_{packing}$ (W/m$^{2\cdot}$C)	D_3 (K/MPa)
Default	5000	2500	0.076
Optimization	9520	3520	0.400



(a) Experimental DoE 1 (b) Experimental DoE 18

(c) Experimental DoE 22 (d) Experimental DoE 25

Fig. 15 Comparison between the experimental pressure signals and those obtained by the simulation using the default *HTC*$_{filling}$, *HTC*$_{packing}$ and D_3 values (Simulation: Default) and the optimized ones (Simulation: Optimization) for four example DoEs

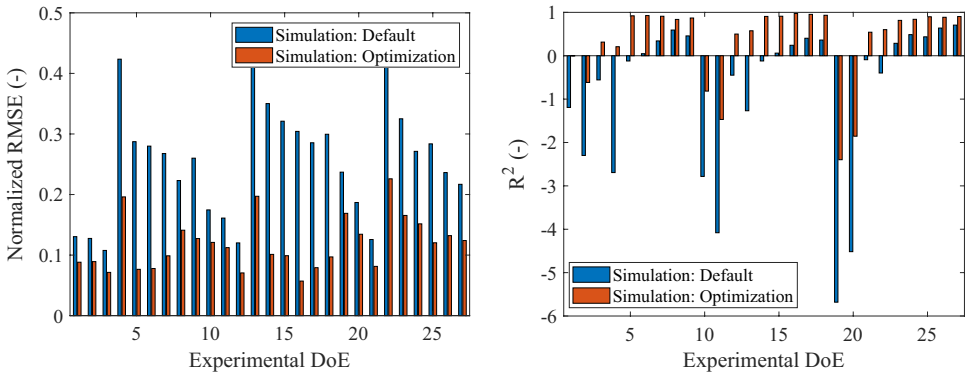
and optimized simulations. This shortcoming should be also strongly related to the current simple solidification criterion.

The error metrics comparing the pressure predictions using the default and optimized parameters for all 27 experimental DoEs are given in Fig. 16. The use of the optimized parameters led to a huge decrease in the RMSE over all considered DoE cases. The simulation using default values exhibited for some cases errors around 40 % which are decreased by more than half while using the optimized parameters. However, as showcased by the Fig. 16b, negative R^2 scores are still calculated even when using optimized modeling parameters. On one hand, this is mainly due to the inability of the simulation to recreate the short shots that were observed experimentally for the molding trials with a low injection velocity of 10 cm^3/s and a holding pressure of 20 MPa, as seen in Fig. 15a.

On the other hand, the optimized value for pressure-dependent viscosity parameter D_3 is found at the upper bound of the surrogate model generation range, which can indicate that the chosen intervals are insufficient or that the implemented models are intrinsically limited to mimic all physical phenomena during the injection molding process.

Conclusion

Surrogate models of high-fidelity simulations, where the variables are material-dependent parameters, offer an alternative to standard experimental identification methods, as shown for the case of injection molding in this work. The main aim of



(a) Root mean square error (b) Coefficient of determination

Fig. 16 Error metrics of the simulated pressure results obtained using the default *HTC*$_{filling}$, *HTC*$_{packing}$ and D_3 values (Simulation: Default) and the optimized ones (Simulation: Optimization) for the 27 experimental DoEs

the method is the generation of an accurate surrogate model, which enables the reverse engineering of the material-dependent parameters using optimization techniques. Considering six independent variables, the POD-NLR and the regular-PCE-LS surrogate models exhibit basically the same performance especially when using a low number of training data. By increasing the number of training data, the adaptive-PCE-LS technique improves steadily the accuracy of the surrogate model and reaches a comparable performance with the fixed-DoE techniques for 120 training simulations. The advantages of the adaptive-PCE-LS should appear more evident when having a high number of variables.

Based on a Sobol sensitivity analysis using the generated surrogate model, we found that the heat transfer coefficient during filling has a low impact on the pressure evolution at a sensor node in front of the heat transfer coefficient during packing or the pressure-dependent viscosity coefficient. By optimizing the $HTC_{filling}$, $HTC_{packing}$ and the pressure-dependent viscosity model parameter D_3 using measured pressure signals, it was possible to calibrate the high-fidelity simulation in order to decrease the deviations in the pressure estimation when compared to those obtained using default simulation values. However, there is still a gap for getting an accurate estimation of the pressure fields in injection molding, which could be closed with this methodology by including additional simulation model parameters or with a refinement of the models implemented in the simulation as for example using crystallization-dependent solidification criterion for thermoplastic polymers.

Appendix A: Cross-WLF viscosity model

The Cross-WLF viscosity model [15] describes the temperature, shear rate, and pressure dependency of the viscosity for thermoplastic materials. This model is used in Autodesk Moldflow Insight 2021.1 to calculate the viscosity of the polymer during its injection molding.

$$\eta = \frac{\eta_0}{1 + \left(\frac{\eta_0 \dot{\gamma}}{\tau^*}\right)^{1-n}} \tag{A1}$$

where:

- η is the viscosity of the melt,
- η_0 is the zero shear viscosity,
- $\dot{\gamma}$ is the shear rate,
- τ^* is the critical stress at the transition to shear thinning,
- n is the power law index in the high shear rate regime.

The zero shear viscosity parameter, η_0, in the above equation is given by the WLF model [16]:

$$\eta_0 = D_1 \exp\left[-\frac{A_1(T - T^*)}{A_2 + (T - T^*)}\right] \tag{A2}$$

where:

- T is the temperature,
- $T^* = D_2 + D_3 P$ is the glass transition temperature,
- $A_2 = A_3 + D_3 P$,
- P is the pressure,
- A_1, A_2, D_1, D_2 and D_3 are data-fitted coefficients.

Acknowledgements The authors wish to thank Dr. Michael Schick (Robert Bosch GmbH) for the helpful discussions concerning the Python Uncertainty Quantification library.

Declarations

Conflict of interests The authors declare that they have no conflict of interest.

References

1. Kennedy P, Zheng R (2013) Flow analysis of injection molds, 2nd edn. Hanser Publishers, Munich; Cincinnati
2. Gao Y, Wang X (2009) Surrogate-based process optimization for reducing warpage in injection molding. J Mater Process Technol 209(3):1302–1309. https://doi.org/https://linkinghub.elsevier.com/retrieve/pii/S0924013608002586. https://doi.org/10.1016/j.jmatprotec.2008.03.048
3. Chen W, Zhou X-h, Wang H-f, Wang W (2010) Multi-objective optimal approach for injection molding based on surrogate model and particle swarm optimization algorithm. J Shanghai Jiaotong Univ (Sci) 15(1):88–93. https://doi.org/10.1007/s12204-010-9517-4
4. Wang X, Gu J, Shen C, Wang X (2015) Warpage optimization with dynamic injection molding technology and sequential optimization method. Int J Adv Manuf Technol 78(1-4):177–187. https://doi.org/10.1007/s00170-014-6621-x
5. Kang G-J, Park C-H, Choi D-H (2016) Metamodel-based design optimization of injection molding process variables and gates of an automotive glove box for enhancing its quality. J Mech Sci Technol 30(4):1723–1732. https://doi.org/10.1007/s12206-016-0328-x
6. Li C, Wang F-L, Chang Y-Q, Liu Y (2010) A modified global optimization method based on surrogate model and its application in packing profile optimization of injection molding process. Int J Adv Manuf Technol 48(5-8):505–511. https://doi.org/10.1007/s00170-009-2302-6
7. Kitayama S, Miyakawa H, Takano M, Aiba S (2017) Multi-objective optimization of injection molding process parameters for short cycle time and warpage reduction using conformal cooling channel. Int J Adv Manuf Technol 88(5-8):1735–1744. https://doi.org/10.1007/s00170-016-8904-x
8. Feng Q, Zhou X (2019) Automated and robust multi-objective optimal design of thin-walled product injection process based on hybrid RBF-MOGA. Int J Adv Manuf Technol 101(9-12):2217–2231. https://doi.org/10.1007/s00170-018-3084-5
9. Shi H, Xie S, Wang X (2013) A warpage optimization method for injection molding using artificial neural network with parametric

sampling evaluation strategy. Int J Adv Manuf Technol 65(1-4):343–353. https://doi.org/10.1007/s00170-012-4173-5
10. Cheng J, Liu Z, Tan J (2013) Multiobjective optimization of injection molding parameters based on soft computing and variable complexity method. Int J Adv Manuf Technol 66(5-8):907–916. https://doi.org/10.1007/s00170-012-4376-9
11. Xia W, Luo B, Liao X-p (2011) An enhanced optimization approach based on Gaussian process surrogate model for process control in injection molding. Int J Adv Manuf Technol 56(9-12):929–942. https://doi.org/10.1007/s00170-011-3227-4
12. Villarreal-Marroquín MG et al (2013) A comparison of two metamodel-based methodologies for multiple criteria simulation optimization using an injection molding case study. J Polym Eng 33(3):193–209. https://www.degruyter.com/document/doi/10.1515/polyeng-2013-0022/html. https://doi.org/10.1515/polyeng-2013-0022
13. Gao H, Zhang Y, Zhou X, Li D (2018) Intelligent methods for the process parameter determination of plastic injection molding. Front Mech Eng 13(1):85–95. https://doi.org/10.1007/s11465-018-0491-0
14. Kurt M, Saban Kamber O, Kaynak Y, Atakok G, Girit O (2009) Experimental investigation of plastic injection molding: Assessment of the effects of cavity pressure and mold temperature on the quality of the final products. Mater Des 30(8):3217–3224. https://linkinghub.elsevier.com/retrieve/pii/S0261306909000089. https://doi.org/10.1016/j.matdes.2009.01.004
15. Cross MM (1965) Rheology of non-Newtonian fluids: A new flow equation for pseudoplastic systems. J Colloid Sci 20(5):417–437. https://linkinghub.elsevier.com/retrieve/pii/009585226590022X. https://doi.org/10.1016/0095-8522(65)90022-X
16. Williams ML, Landel RF, Ferry JD (1955) Thetemperature dependence of relaxation mechanisms ina morphous polymers and other Glass-forming liquids. J Am Chem Soc 77 (14):3701–3707. https://doi.org/10.1021/ja01619a008
17. Nguyen-Chung T, Jüttner G, Löser C, Pham T, Gehde M (2010) Determination of the heat transfer coefficient from short-shots studies and precise simulation of microinjection molding. Polymer Eng Sci 50(1):165–173. https://doi.org/10.1002/pen.21536
18. Stricker M, Steinbichler G (2014) Determination of heat transfer coefficients at the polymer-mold-interface for injection molding simulation by means of calorimetry, 137–141. http://aip.scitation.org/doi/abs/10.1063/1.4873750
19. Kleindel S, Eder R, Schretter H, Hochenauer C (2014) The elastic mold deformation during the filling and packing stage of the injection Molding Process. Smart Sci 2 (1):44–53. https://doi.org/10.1080/23080477.2014.11665603
20. Wang GG, Shan S (2007) Review of metamodeling techniques in support of engineering design optimization. J Mech Des 129(4):370–380. https://asmedigitalcollection.asme.org/mechanicaldesign/article/129/4/370/466824/Review-of-Metamodeling-Techniques-in-Support-of. https://doi.org/10.1115/1.2429697
21. Forrester AI, Keane AJ (2009) Recent advances in surrogate-based optimization. Prog Aerosp Sci 45(1-3):50–79. https://linkinghub.elsevier.com/retrieve/pii/S0376042108000766. https://doi.org/10.1016/j.paerosci.2008.11.001
22. Iuliano E, Quagliarella D (2013) Proper Orthogonal Decomposition, surrogate modelling and evolutionary optimization in aerodynamic design. Comput Fluids 84:327–350. https://linkinghub.elsevier.com/retrieve/pii/S0045793013002223. https://doi.org/10.1016/j.compfluid.2013.06.007
23. Razavi S, Tolson BA, Burn DH (2012) Review of surrogate modeling in water resources: REVIEW. Water Resour Res 48(7). https://doi.org/10.1029/2011WR011527
24. McClarren RG (2018) Uncertainty quantification and predictive computational science: A foundation for physical scientists and engineers, 1st ed. 2018. Springer International Publishing : Imprint, Springer, Cham
25. Giunta A, Wojtkiewicz S, Eldred M. (2003) Overview design of experiments methods for computational simulations (American Institute of aeronautics and astronautics). https://doi.org/10.2514/6.2003-649
26. Jank W (2005) Quasi-Monte Carlo sampling to improve the efficiency of Monte Carlo EM. Comput Stat Data Anal 48(4):685–701. https://linkinghub.elsevier.com/retrieve/pii/S0167947304001033. https://doi.org/10.1016/j.csda.2004.03.019
27. Metropolis N, Ulam S (1949) The Monte Carlo method. J Am Stat Assoc 44(247):335–341. https://doi.org/10.1080/01621459.1949.10483310
28. Halton JH (1960) On the efficiency of certain quasi-random sequences of points in evaluating multi-dimensional integrals. Numer Math 2(1):84–90. https://doi.org/10.1007/BF01386213
29. Niederreiter H (1992) Society for Industrial and Applied Mathematics.Random number generation and quasi-Monte Carlomethods: Author's expanded lecture notes from his talks at the NSF-CBMS Regional Research Conference on Random Number Generation and Quasi-Monte Carlo Methods held at the University of Alaskaat Fairbanks, Aug. 13-17, 1990 (Society for Industrial and Applied Mathematics, Philadelphia, Pa.) OCLC: 775728856
30. Sobol' I (1967) On the distribution of points in a cube and the approximate evaluation of integrals. USSR Comput Math Math Phys 7 (4):86–112. https://linkinghub.elsevier.com/retrieve/pii/0041555367901449. https://doi.org/10.1016/0041-5553(67)90144-9
31. Faure H, Kritzer P, Pillichshammer F (2015) From van derCorputto modern constructions of sequences for quasi-Monte Carlo rules. Indag Math 26(5):760–822. https://linkinghub.elsevier.com/retrieve/pii/S0019357715000683. https://doi.org/10.1016/j.indag.2015.09.001
32. Nouy A (2010) A priori model reduction throughProper Generalized Decompositionfor solving time-dependent partial differential equations. Comput Methods Appl Mech Eng 199(23-24):1603–1626. https://linkinghub.elsevier.com/retrieve/pii/S0045782510000186. https://doi.org/10.1016/j.cma.2010.01.009
33. Ryckelynck D, Chinesta F, Cueto E, Ammar A (2006) On the a priori model reduction: Overview and recent developments. Arch Comput Methods Eng 13(1):91–128. https://link.springer.com/article/10.1007/BF02905932
34. Du J et al (2013) POD reduced-order unstructured mesh modeling applied to 2d and 3d fluid flow. Comput Math Appl 65(3):362–379. https://linkinghub.elsevier.com/retrieve/pii/S0898122112004397. https://doi.org/10.1016/j.camwa.2012.06.009
35. Simpson T, Poplinski J, Koch PN, Allen J (2001) Meta models for computer-based engineering design: Surveyand recommendations. Eng Comput 17(2):129–150. https://doi.org/10.1007/PL00007198
36. Teixeira R, Nogal M, O'Connor A (2021) Adaptive approaches in metamodel-based reliability analysis: A review. Struct Saf 89:102019. https://linkinghub.elsevier.com/retrieve/pii/S0167473020300989. https://doi.org/10.1016/j.strusafe.2020.102019
37. Gratiet LL, Marelli S, Sudret B (2015) Metamodel-based sensitivity analysis: Polynomial chaos expansions and gaussian processes. In: Ghanem R, Higdon D, Owhadi H (eds) Handbook of uncertainty quantification. Springer International Publishing, Cham, pp 1–37. https://doi.org/10.1007/978-3-319-11259-6_38-1

38. Crestaux T, Le Maıtre O, Martinez J-M (2009) Polynomial chaos expansion for sensitivity analysis. Reliab Eng Syst Saf 94(7):1161–1172. https://linkinghub.elsevier.com/retrieve/pii/S0951832008002561. https://doi.org/10.1016/j.ress.2008.10.008
39. Nossent J, Elsen P, Bauwens W (2011) Sobol' sensitivity analysis of a complex environmental model. Environ Modell Softw 26 (12):1515–1525. https://linkinghub.elsevier.com/retrieve/pii/S1364815211001939. https://doi.org/10.1016/j.envsoft.2011.08.010
40. Sobol I (2001) Global sensitivity indices for nonlinear mathematical models and their Monte Carlo estimates. Math Comput Simul 55(1-3):271–280. https://linkinghub.elsevier.com/retrieve/pii/S0378475400002706. https://doi.org/10.1016/S0378-4754(00)00270-6
41. Rosolem R, Gupta HV, Shuttleworth WJ, Zeng X, de Gonçalves LGG (2012) A fully multiple-criteria implementation of the Sobol method for parameter sensitivity analysis: Multicriteria sobol sensitivity analysis. J Geophys Res: Atmos 117(D7). https://doi.org/10.1029/2011JD016355
42. Spoerer Y, Boldt R, Androsch R, Kuehnert I (2021) Pressure-andtemperature-dependent crystallization kinetics of isotactic polypropylene under process relevant conditions. Curr Comput-Aided Drug Des 11 (9):1138. https://www.mdpi.com/2073-4352/11/9/1138. https://doi.org/10.3390/cryst11091138
43. Efron B (1981) Nonparametric estimates of standard error: The jackknife, the bootstrap and other methods. Biometrika 68(3):589–599. https://doi.org/10.1093/biomet/68.3.589