12th IFAC International Symposium on
Advanced Control of Chemical Processes
July 14-17, 2024. Toronto, Canada

Robust nonlinear model predictive control of continuous crystallization using Bayesian last layer surrogate models

Collin R. Johnson, Felix Fiedler and Sergio Lucia

* *Chair of Process Automation Systems, TU Dortmund University, Emil-Figge-Str. 70, 44227 Dortmund (e-mail: {collin.johnson, felix.fiedler, sergio.lucia}@tu-dortmund.de).*

Abstract: In scenarios where high-fidelity physical models are either unavailable or are impractical due to their high complexity, data-based models offer a viable solution to obtain the system model necessary for predictive control. However, the accuracy of the predictions obtained by data-based models is limited. We propose to use neural networks with Bayesian last layer to obtain information about the uncertainty of the predictions. This paper demonstrates the use of Bayesian last layer surrogate models in a robust nonlinear model predictive control setting. The nonlinear model predictive control problem is adapted by considering the predicted uncertainty of the surrogate model, which can be efficiently computed using the Bayesian last layer method, in the cost function. The controller thus takes model uncertainty explicitly into account and by its formulation also avoids areas of extrapolation. The proposed method is applied to a mixed-suspension, mixed-product-removal crystallizer and simulation studies show that it outperforms a standard data-based model.

Keywords: model predictive control, uncertainty quantification, neural networks, robust control.

1. INTRODUCTION

Advanced control techniques such as nonlinear model predictive control (NMPC) have been applied to a broad spectrum of fields and are especially useful for challenging control tasks (Rawlings et al., 2017; Lopez-Negrete et al., 2013). The key element of most modern control methods is the model. Building a high-fidelity model using first-principle equations is usually the preferred and most accurate approach. In practice, however, this type of modeling can prove to be complicated. Some phenomena are hardly possible to model because they are too complex and sometimes stochastic in nature, such as fouling in crystallization (Zhang et al., 2015) or climate modeling (Kashinath et al., 2021). For other systems it may be possible to build a model, but the model can become computationally expensive, which might prohibit any online optimization.

For issues of modeling difficulties and of computational complexity, data-based models can provide a solution to enable the use of advanced model-based control techniques (Bhat and McAvov, 1989). However, data-based models are generally only valid for the domain in which they were trained. In practice, it must therefore be ensured that data-based models are only used in the area of training data, that is, for interpolation. However, even the description of interpolation is for high-dimensional data no longer entirely clear (Balestriero et al., 2021). The challenge is therefore to ensure that data-based models are only used in regions where they can make predictions with sufficient certainty. This is particularly important for safety-relevant processes, as seen in Hewing et al. (2020); McKinnon and Schoellig (2019).

Standard feedforward neural networks are a possible data-based solution to model nonlinear dynamic models. Unfortunately, these typically do not provide any information regarding the uncertainty of their predictions. One possible solution to enforce operation into high-certainty regions is to constrain inputs within the training data bounds. The prerequisite for this, however, is for the training data to completely cover the space within the constraints, which can be challenging for high-dimensional data (Balestriero et al., 2021). A more appropriate solution is therefore to use a data-based model that indicates how certain the prediction is. A prominent example of a model that is capable to give a measure of uncertainty are Gaussian processes (GPs) (Rasmussen and Williams, 2006). GPs are non-parametric models in which predictions are made directly on the basis of training data. Predictions are Gaussian distributed, and the variance of the prediction can be interpreted as the uncertainty of the prediction. Due to the time-consuming evaluation of the model (Lázaro-Gredilla and Figueiras-Vidal, 2010), GPs are typically only of limited use for online optimization. Another example of probabilistic models is Bayesian linear regression (BLR) (Bishop, 2006). For BLR, the weights of the regression model are determined as distribution functions. As it is typical for linear regression, the disadvantage lies in the necessary and nontrivial manual selection of features.

* This work was funded by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) – 504676854 – within the Priority Program "SPP 2364: Autonomous processes in particle technology".

Bayesian neural networks (BNNs) represent a probabilistic method in which the features do not have to be selected manually (Jospin et al., 2020). As with BLR, the weights of the model follow distribution functions, although the non-linearity of the activation function of the neural network prohibits an analytical solution of the posterior distribution (Jospin et al., 2020). Hence, approximate methods such as approximate inference or sampling methods must be used to train a BNN (Jospin et al., 2020).

Bayesian last layer (BLL) offers a compromise between BLR and BNNs (Fiedler and Lucia, 2023). Here, neural networks are trained whereby only the weights of the last layer are assumed to follow a distribution function. By choosing a linear activation function for the last layer of the network, the simple methods from BLR can be applied. Hence, an analytical solution of the posterior distribution of the weights of the last layer can be obtained. However, the features do not have to be chosen manually, but are learned by the hidden layers of the neural network. To approximate a full BNN, the weights of the hidden layers are interpreted as hyperparameters (Fiedler and Lucia, 2023) and can be determined by marginal likelihood maximization (Bishop, 2006). The uncertainty information of the predictions can be used to modify an NMPC controller based on a data-based model, for example by defining trust regions for areas where the predicted uncertainty is low and constraining the optimization problem accordingly (Fiedler and Lucia, 2022). However, this requires that the data-based model approximates reality sufficiently well in these areas. If the system identification is not sufficient in the trust regions, for example when only a limited amount of data points are available, the uncertainty of the data-based model should be explicitly considered by employing robust NMPC approaches as for example the multi-stage robust NMPC presented in Lucia et al. (2013).

The main contribution of this work is the proposal of a robust multi-stage NMPC method for which the required scenario trees are constructed and adapted online based on the prediction uncertainty given by the Bayesian last layer approach. We show in a simulation of a crystallization case-study that the proposed approach leads to an improved performance when compared to the naive use of a data-based model within a standard NMPC controller.

This work is structured as follows. The background on NMPC and system identification is presented in Section 2. The formulation of the BLL framework and the robust data-based NMPC are then presented in Section 3. The results for a mixed-suspension, mixed-product-removal (MSMPR) crystallizer system are presented in Section 4 and the paper is concluded in Section 5.

2. NONLINEAR MODEL PREDICTIVE CONTROL AND DATA-BASED MODELS

2.1 Nonlinear model predictive control problem

We consider discrete-time nonlinear dynamic systems of the form:

$$s_{k+1} = f(s_k, u_k), \tag{1}$$

where s_k and u_k represent the states and inputs of the system at the discrete time step k. The optimization problem that needs to be solved to control the system (1) via NMPC is:

$$\min_{u_k} \quad \sum_{k=0}^{N-1} l(s_k, u_k) + V_f(s_N), \tag{2a}$$

$$\text{s.t.} \quad s_{k+1} = f(s_k, u_k), \tag{2b}$$

$$s_k \in \mathbb{S}, \tag{2c}$$

$$u_k \in \mathbb{U}, \tag{2d}$$

$$s_0 = s_{\text{initial}}, \tag{2e}$$

where the prediction horizon is denoted by N. The objective function is split into stage cost $l(s_k, u_k)$ and terminal cost $V_f(s_N)$. The constraints include the system model $f(s_k, u_k)$ and the initial state of the system s_{initial}. Additionally, the state constraints and the input constraints are denoted by $\mathbb{S}$ and $\mathbb{U}$.

2.2 Generation of a data-based model

The system model is the core element of the NMPC problem. If a first principle model is difficult to obtain or too complex to solve problem (2) in real-time, a data-based model can enable the development of an NMPC controller.

Inferring a dynamic model from data is referred to as system identification (Ljung, 2017). This is typically formalized with the following mathematical setting:

$$t = y(x) + \epsilon, \tag{3a}$$

$$\epsilon \sim \mathcal{N}(0, \beta^{-1}), \tag{3b}$$

where t represents the targets, which are assumed to be generated by some unknown function y with some additive noise ϵ. Here, x denotes the inputs of the hidden function. For the setting shown in (1), x corresponds to s_k and u_k. It is assumed that ϵ is distributed as white Gaussian noise governed by the precision parameter β. The goal of regression is to determine the unknown function y.

As a first step, we choose y to be a linear model. Incorporating a Bayesian perspective to linear regression results in additional uncertainty information of the predictions leading to a Bayesian linear regression (BLR) approach. To achieve this, we introduce a prior probability of the weights:

$$p(w|\alpha) = \mathcal{N}(w|0, \alpha^{-1}), \tag{4}$$

where α is the precision of the distribution. The weights w are the parameters of the model. The posterior distribution can be computed by applying Bayes' Law:

$$p(w|t, \alpha, \beta) = \frac{p(t|w, \beta)p(w|\alpha)}{p(t|\alpha, \beta)}, \tag{5}$$

where we have omitted the inputs x to keep the notation uncluttered. The term $p(t|w, \beta)$ represents the likelihood, and $p(t|\alpha, \beta)$ is referred to as the marginal likelihood since the weights w have been marginalized. It represents the probability over all possible values for the weights. Using properties of normally distributed variables, it is possible to determine the mean value and the covariance of the posterior as:

$$C = \alpha I + \beta \Phi^T \Phi, \tag{6a}$$

$$\bar{w} = \beta C^{-1} \Phi^T t, \tag{6b}$$

where C represents the covariance of the posterior and $\bar{w}$ the mean of the posterior. The feature matrix is given by Φ which is an $N \times M$ dimensional matrix. The hyperparameters α and β are generally not known and must therefore be determined.

3. ROBUST NMPC WITH BAYESIAN LAST LAYER MODELS

3.1 Bayesian last layer models

Bayesian last layer (BLL) provides a method to calculate the uncertainty information as shown in the previous section for Bayesian linear regression. However, a neural network is used to identify the nonlinear features. In contrast to full Bayesian neural networks, only the weights of the last layer of the network are assumed to follow a distribution. Additionally, the activation function of the output layer of the network is chosen as a linear activation function. The subsequent absence of a nonlinear transformation of the last layer allows the analytical computation of the posterior distribution for the weights of the last layer, as shown for BLR.

For BLL, the approximation of a full BNN is desired, therefore, in addition to α and β, the weights and biases of the hidden layers W are treated as hyperparameters of the model. A method where hyperparameters are determined exclusively on the basis of the given data is marginal likelihood maximization (Bishop, 2006). The marginal likelihood is represented by the denominator in (5) for BLR and must be extended by W for BLL. After performing the marginalization of the parameters on the last layer w, the log marginal likelihood is given by (see Fiedler and Lucia (2023) for more details):

$$\log p(t|\alpha,\beta,W) = \frac{M}{2}\log\alpha + \frac{N}{2}\log\beta - \frac{\beta}{2}|t-y|_2^2 - \frac{\alpha}{2}|\bar{w}|_2^2 - \frac{1}{2}\log|C| - \frac{N}{2}\log 2\pi, \quad (7)$$

where the number of features is denoted by M. The covariance of the posterior C and the mean of the posterior $\bar{w}$ are given by (6a) and (6b). The distributions for additive noise and prior of the weights were both selected as Gaussian distributions. Hence, the posterior distribution from (7) is also Gaussian distributed. The predictive distribution is given by:

$$p(t_{\text{pred}}|t,\alpha,\beta,W) = \mathcal{N}(t_{\text{pred}}|\mu_{\text{pred}},\sigma^2_{\text{pred}}), \quad (8a)$$

with

$$\mu_{\text{pred}} = \bar{w}^T\phi(x), \quad (8b)$$

$$\sigma^2_{\text{w}} = \phi(x)^T C^{-1}\phi(x), \quad (8c)$$

$$\sigma^2_{\text{pred}} = \beta^{-1} + \sigma^2_{\text{w}}. \quad (8d)$$

The variance σ^2_{w} represents the uncertainty of the model in the weights. The derivation of the equations for the multivariate case of Bayesian last layer can be seen in Fiedler and Lucia (2023).

3.2 Robust multi-stage NMPC using model uncertainty

Robust multi-stage NMPC is a method to incorporate uncertainties on the model into the controller (Lucia et al., 2013). A scenario tree is developed based on possible realizations of the uncertainty. Thus, in addition to the nominal prediction of the model, the predictions for the different scenarios are also computed and taken into account in the NMPC problem formulation. The incorporation of the possible scenarios in the optimization problem leads to constraint satisfaction for the considered scenarios and the tree structure enables the introduction of feedback in the predictions to avoid overly conservative solutions.

While traditionally different parameter or disturbance values are used to define different scenarios, we propose in this paper to consider the predictions of the states of the data-based model as uncertain. The magnitude of the uncertainty is computed by the standard deviation given by the BLL approach:

$$\begin{pmatrix} s_k^1 \\ s_k^2 \\ s_k^3 \end{pmatrix} = \begin{pmatrix} \mu_{\text{pred},k} + 3\,\text{diag}(\Sigma_w) \\ \mu_{\text{pred},k} \\ \mu_{\text{pred},k} - 3\,\text{diag}(\Sigma_w) \end{pmatrix} \quad (9)$$

where $\mu_{\text{pred},k}$ corresponds to the mean of the prediction of the data-based model from (8b) for the k-th time step. The matrix Σ_w is a diagonal matrix originating from the multivariate case, containing the variances σ_w of the individual states, given by (8c). The superscript indicates the respective realization of the uncertainty at the time step k. To build the scenario tree used for robust NMPC, we employ the concept of a robust horizon as shown in Lucia et al. (2013) to avoid the exponential growth of the optimization problem. Branching is performed according to (9) for the time steps inside of the robust horizon. If the prediction horizon is larger than the robust horizon no further branching is performed at the following time steps and the NMPC problem is solved for all scenarios of the scenario tree.

4. CASE STUDY FOR A MSMPR CRYSTALLIZER

4.1 MSMPR model

The presented methodology is applied to a continuous crystallization model. The model includes a stirred tank that can be cooled with a cooling jacket. The crystallizer is assumed to be ideally mixed, with the stirrer providing no energy input and not influencing the crystals. The model equations of the continuous phase are derived by material and energy balances, and are obtained as follows:

$$\frac{dc}{dt} = \frac{1}{m}(-\dot{m}_{\text{cryst}} + \rho F_{\text{feed}}(c_{\text{feed}} - c)), \quad (10a)$$

$$\frac{dT}{dt} = \frac{1}{m\,c_p}(-\Delta H_{\text{cryst}}\dot{m}_{\text{cryst}} + \rho F_{\text{feed}}c_p (T_{\text{feed}} - T) - UA(T - T_J)), \quad (10b)$$

$$\frac{dT_J}{dt} = \frac{1}{m_J\,c_{p,J}}(\rho_J F_J c_p(T_{J,\text{in}} - T_J) - UA(T_J - T)), \quad (10c)$$

where the states are the concentration c, the temperature of the crystallization medium T, and the temperature of the cooling jacket T_J. The mass of the crystallization medium is m and the specific heat capacity of the crystallization medium is c_p which are both assumed to be constant. Similarly, the mass in the cooling jacket is m_J and its specific heat capacity is $c_{p,J}$. The densities of the

crystallization medium ρ and of the cooling medium ρ_J are also considered as constant. The properties of the inlet streams include the volume flow of the crystallization medium F_{feed}, the temperature of the crystallization inlet flow T_{feed}, the volume flow of the cooling medium F_J , and the temperature of the cooling inlet flow $T_{J,in}$. The heat transfer between crystallization medium and cooling jacket is computed by the heat transfer coefficient U and the area of heat transfer A. The mass removed by crystallization is $\dot{m}_{\text{cryst}}$ and the heat of crystallization is ΔH_{cryst}.

The change of the disperse phase, i.e. the crystals, is modeled by the method-of-moments (Hulburt and Katz, 1964). For this method, the crystal size distribution is tracked by its moments. For the crystallization model, we consider the first three moments μ_0, μ_1, and μ_2 as states. The model equations of the disperse phase are given as follows:

$$\frac{d\mu_0}{dt} = \frac{\rho F_{\text{feed}}}{m}(\mu_{0,\text{in}} - \mu_0), \tag{11a}$$

$$\frac{d\mu_1}{dt} = G\mu_0 + \frac{\rho F_{\text{feed}}}{m}(\mu_{1,\text{in}} - \mu_1), \tag{11b}$$

$$\frac{d\mu_2}{dt} = 2G\mu_1 + \frac{\rho F_{\text{feed}}}{m}(\mu_{2,\text{in}} - \mu_2). \tag{11c}$$

The inlet flow is assumed to contain seed crystals. The k-th moment of the seed crystal distribution is denoted by $\mu_{k,\text{in}}$. Crystal growth is considered to be the mechanism that can influence the crystal size distribution. The size-independent growth rate is presented by G. Phenomena such as crystal birth, agglomeration or breakage are neglected.

The model includes 6 states c, T, T_J, μ_0, μ_1, and μ_2. The volume flows of the two inlets F_{feed} and F_J, as well as the temperature of the cooling flow $T_{J,\text{in}}$ are selected as inputs of the system. In addition, characteristic values can be calculated from the moments which can be part of the cost function of the NMPC problem. This allows an average crystal size to be calculated from the first two moments:

$$L_{10} = \frac{\mu_1}{\mu_0}. \tag{12}$$

The used parameters and auxiliary equations needed for the model are shown in Table A.2

4.2 Results

For a thorough analysis of the proposed method we evaluate for training data sets of different sizes, 10 data-based models for each size. For each data set we train a standard NN and a NN with BLL. For the generation of the training data sets the system was excited by changing the inputs. Values for the inputs were randomly uniformly sampled from the range shown in Table A.1 and subsequently kept constant for a random time. The training data was gathered at time steps of 5 seconds. For the simulation of the model the tool do-mpc (Fiedler et al., 2023) together with CasADi (Andersson et al., 2019) and the solvers from SUNDIALS (Hindmarsh et al., 2005) were used.

We choose the same architecture for both types of data-based models of two hidden layers with 30 neurons per layer and a tanh activation function with an additional subsequent linear transformation. Keras (Chollet et al., 2015) and Tensorflow (Abadi et al., 2015) were used for model training. For the training of the standard neural network models the mean-squared-error was used as loss function. The method presented in (Fiedler and Lucia, 2023) was used for the training of the neural network models with BLL.

The model with standard NN is used in a nominal NMPC scheme because no uncertainty quantification is provided. The models with BLL are used as shown in Section 3.2 with the inclusion of uncertainty for the construction of the uncertain scenarios. In both cases, the simulator uses the physical model from (10). We choose $N = 6$ for the prediction horizon and in case of multi-stage NMPC, a robust horizon of 1. We choose the temperature as an uncertain state and construct the scenario tree according to (9), leading to 3 scenarios. As a control objective, we choose the maximization of the crystal size L_{10}. Thus, for the stage and terminal cost we define:

$$l(s_k, u_k) = -L_{10,k}^2 + \Delta u_k^T W_u \Delta u_k, \tag{13a}$$

$$V_f(s_N) = -L_{10,N}^2, \tag{13b}$$

where W_u represents the diagonal weighting matrix of the penalization of the change of the inputs Δu_k. The weighting matrix was chosen as $W_u = \text{diag}(\,10,\ 100,\ 0.1\,)$. Additionally, the input constraints for the NMPC problem were chosen equal to the ranges used for the generation of the training data shown in Table A.1. For the setup and solution of the robust multi-stage NMPC, do-mpc, CasADi and the solvers from SUNDIALS and IPOPT (Wächter and Biegler, 2006) were used. All code to reproduce the results is openly available. [1]

To simplify the comparison, we perform the evaluation of each case for the same initial state:

$$s_0 = \begin{pmatrix} c_0 \\ T_0 \\ T_{J,0} \end{pmatrix} = \begin{pmatrix} 0.226 \\ 350 \\ 350 \end{pmatrix}. \tag{14}$$

To maximize the crystal size, the temperature in the crystallizer must be kept as low as possible. A lower bound constraint for this state should therefore inevitably be active for optimal operation. We add a lower bound constraint for the temperature in the crystallizer at $T = 320\,\text{K}$.

Figure 1 shows the obtained crystal size L_{10} after a closed-loop simulation of 300 time steps (25 minutes) when a standard NN and the proposed NN with BLL are used as models inside of the respective controller for different numbers of data points. Since the data generation is random we computed the results for 10 different models for each size of the training data. As expected, system identification improves with more data. This is reflected in the increasing trend of the values achieved for L_{10}. For very small data sets, the data sets may be insufficient in some cases to describe the relevant dynamics of the system. The performance of both methods therefore varies greatly for different data sets. Nevertheless, even for small data sets it is clear that the control performance for the proposed method with BLL works better than a standard NN. This is primarily due to the fact that training by marginal

[1] `https://github.com/collinj2812/multistageNMPC_BLL`

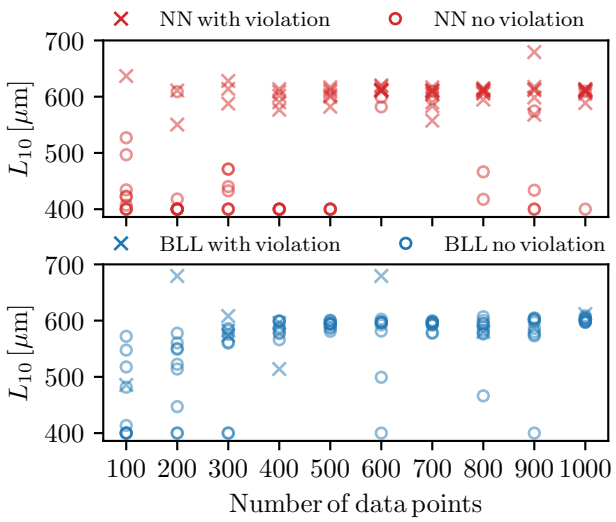

Fig. 1. Results for the attained crystal sizes at the end of a closed-loop NMPC simulation with data-based models trained with data sets of different sizes. The results for the NN without BLL are shown in red in the top and for the proposed method in blue in the bottom. Constraint violations in the course of the respective simulation are indicated by $\times$.

likelihood maximization leads to a more accurate model on average. As the size of the data sets increases, the system identification of the standard neural networks also improves. However, the performance of the proposed method is still superior due to the consideration of uncertainty in the multi-stage NMPC. Although for both methods a larger diameter L_{10} is achieved, the standard neural network consistently violates the constraints. This can also be seen in Table 1. The standard neural network violates the constraints for larger data sets in up to 90 % of cases. Although the maximum and average constraint violations are not as high as constraint violations that can occur with both methods due to a too small and too sparse data set, the standard NN cannot avoid constraint violations even when larger amounts of data are used. It can be seen that the proposed method achieves better results on average for all data set sizes. In some cases, the average achieved L_{10} for the standard neural network can be larger than that for the trajectories of the proposed method, but the standard neural network violates the constraints in most of the 10 cases. Therefore, the performance of the proposed method should be preferred.

Figure 2 shows an example of the trajectories for data sets with a size of 900 data points. On average, the performance is significantly more consistent for the proposed method. Although there is also one trajectory for the proposed method that does not obtain a good control performance, it can be seen that for the rest the system identification is better and no constraint violation occurs. With the standard neural network, either the system identification is poor or the constraints are violated.

5. CONCLUSION

Data-based surrogate models allow the application of model-based control methods to systems where first-

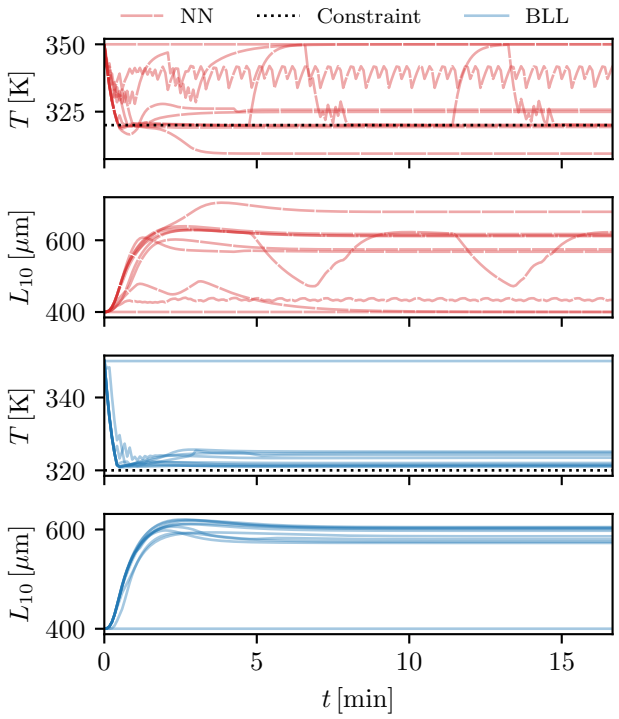

Fig. 2. Temperature and average crystal size for 10 models generated with a different set of 900 data points using a standard NN (red dashed line) and the proposed approach (blue solid line).

principle models are not available. An uncertainty description of the predictions is vital for safe operation when using uncertain models. For the surrogate model, we propose to train a NN with Bayesian last layer using marginal likelihood maximization. We propose a method to utilize the predicted uncertainty of the Bayesian last layer by constructing uncertain scenarios in a robust multi-stage NMPC framework. We show that this controller has better performance than using NMPC with a standard NN irrespective of the size of the training data set. A simulation example of continuous crystallization shows that system identification using marginal likelihood maximization leads to more accurate models on average and the use of uncertainty in the controller results in fewer or no constraint violations.

REFERENCES

M. Abadi, A. Agarwal et al. (2015). TensorFlow: Large-scale machine learning on heterogeneous systems.

J. A. E. Andersson, J. Gillis et al. (2019). CasADi: A software framework for nonlinear optimization and optimal control. *Mathematical Programming Computation*, 11(1), 1–36.

R. Balestriero, J. Pesenti et al. (2021). Learning in High Dimension Always Amounts to Extrapolation.

N. Bhat and T. J. McAvov (1989). Use of neural nets for dynamic modeling and control of chemical process systems.

C. M. Bishop (2006). *Pattern Recognition and Machine Learning*. Information Science and Statistics. Springer, New York.

F. Chollet et al. (2015). Keras. `https://keras.io`.

Table 1. Results of the case study for a standard NN and the proposed method. The mean attained diameter is based only on trajectories where the constraints were not violated. The average constraint violation was calculated based on all time steps.

Performance indicator	**Number of data points used**									
	100	200	300	400	500	600	700	800	900	1000
Standard NN										
Mean attained diameter [μm]	434.5	428.5	430.8	400.1	400.1	590.5	609.5	442.0	452.1	504.8
% of trajectories with constraint violation	10	20	30	50	60	80	90	80	60	80
Average constraint violation [K]	0.21	0.01	0.60	0.44	0.11	0.23	0.07	0.10	1.04	0.07
Proposed NN with BLL										
Mean attained diameter [μm]	459.2	502.4	523.7	586.7	592.9	562.7	592.3	576.5	572.4	601.6
% of trajectories with constraint violation	10	10	30	20	0	10	0	10	0	10
Average constraint violation [K]	0.01	0.92	0.23	0.28	0	0.74	0	$1.4 \cdot 10^{-5}$	0	$1.2 \cdot 10^{-4}$

F. Fiedler, B. Karg et al. (2023). Do-mpc: Towards FAIR nonlinear and robust model predictive control. *Control Engineering Practice*, 140, 105676.

F. Fiedler and S. Lucia (2022). Model predictive control with neural network system model and Bayesian last layer trust regions. In *2022 IEEE 17th International Conference on Control & Automation (ICCA)*, 141–147. IEEE, Naples, Italy.

F. Fiedler and S. Lucia (2023). Improved Uncertainty Quantification for Neural Networks With Bayesian Last Layer. *IEEE Access*, 11, 123149–123160.

L. Hewing, J. Kabzan et al. (2020). Cautious Model Predictive Control Using Gaussian Process Regression. *IEEE Transactions on Control Systems Technology*, 28(6), 2736–2743.

A. C. Hindmarsh, P. N. Brown et al. (2005). SUNDIALS: Suite of nonlinear and differential/algebraic equation solvers. *ACM Transactions on Mathematical Software*, 31(3), 363–396.

L. Hohmann, T. Greinert et al. (2018). Analysis of Crystal Size Dispersion Effects in a Continuous Coiled Tubular Crystallizer: Experiments and Modeling. *Crystal Growth & Design*, 18(3), 1459–1473.

H. Hulburt and S. Katz (1964). Some problems in particle technology. *Chemical Engineering Science*, 19(8), 555–574.

L. V. Jospin, W. Buntine et al. (2020). Hands-on Bayesian Neural Networks – a Tutorial for Deep Learning Users. *arXiv:2007.06823 [cs, stat]*.

K. Kashinath, M. Mustafa et al. (2021). Physics-informed machine learning: Case studies for weather and climate modelling. *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences*, 379(2194), 20200093.

M. Lázaro-Gredilla and A. R. Figueiras-Vidal (2010). Marginalized neural network mixtures for large-scale regression. *IEEE transactions on neural networks*, 21(8), 1345–1351.

L. Ljung (2017). *System Identification*, 1–19. John Wiley & Sons, Inc., Hoboken, NJ, USA.

R. Lopez-Negrete, F. J. D'Amato et al. (2013). Fast nonlinear model predictive control: Formulation and industrial process applications. *Computers & Chemical Engineering*, 51, 55–64.

S. Lucia, T. Finkler et al. (2013). Multi-stage nonlinear model predictive control applied to a semi-batch polymerization reactor under uncertainty. *Journal of Process Control*, 23(9), 1306–1319.

C. D. McKinnon and A. P. Schoellig (2019). Learning Probabilistic Models for Safe Predictive Control in Unknown Environments. In *2019 18th European Control Conference (ECC)*, 2472–2479. IEEE, Naples, Italy.

C. E. Rasmussen and C. K. I. Williams (2006). *Gaussian Processes for Machine Learning*. Adaptive Computation and Machine Learning. MIT Press, Cambridge, Mass.

J. B. Rawlings, D. Q. Mayne et al. (2017). *Model Predictive Control: Theory, Computation, and Design*. Nob Hill Publishing, Madison, Wisconsin, 2nd edition edition.

A. Wächter and L. T. Biegler (2006). On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming. *Mathematical Programming*, 106(1), 25–57.

K. Wohlgemuth (2012). *Induced Nucleation Processes during Batch Cooling Crystallization*. Ph.D. thesis.

F. Zhang, J. Xiao et al. (2015). Towards predictive modeling of crystallization fouling: A pseudo-dynamic approach. *Food and Bioproducts Processing*, 93, 188–196.

Appendix A. TRAINING AND MODEL PARAMETERS

Table A.1. Input ranges.

	$T_{J,\text{in}}$		F_J		F_{feed}	
Training	300	350	0.1	0.5	0.1	0.3
Testing	300	350	0.1	0.5	0.1	0.3

Table A.2. Equations and parameters used in the MSMPR model. Parameters for modeling the chemical system of L-Alanine/Water are from (Wohlgemuth, 2012) and (Hohmann et al., 2018).

V	$10\,[\text{m}^3]$	$\mu_{0,\text{in}}$	$1.0292 \times 10^8\,[-]$
c_p	$4.2\,[\text{kJ kg K}^{-1}]$	$\mu_{1,\text{in}}$	$4.1177 \times 10^4\,[\text{m}]$
$c_{p,J}$	$4.2\,[\text{kJ kg K}^{-1}]$	$\mu_{2,\text{in}}$	$1.7501 \times 10^1\,[\text{m}^2]$
V_J	$1\,[\text{m}^3]$	A	$10\,[\text{m}^2]$
ρ	$1043\,[\text{kg m}^{-3}]$	k_V	$\frac{\pi}{6}\,[-]$
ρ_{cryst}	$1432\,[\text{kg m}^{-3}]$	U	$1000\,[\text{W m}^{-2}\,\text{K}]$

$G = 5.857 \times 10^{-5} \Delta S^2 \tanh\left(\frac{0.913}{\Delta S}\right)$ (Hohmann et al., 2018)

$c^* = 0.11238 e^{9.0849 \times 10^{-3} T}$ (Wohlgemuth, 2012)

$\Delta S = \frac{c - c^*}{c^*}$

$\dot{m}_{\text{cryst}} = 3 V k_V \rho_{\text{cryst}} G \mu_2$