Gaussian Process-Based Identification and Control of Polydôme

Semester Project J061

Radu Constantin Martin

Professor: Colin Jones

PhD Students: Emilio Maddalena, Yingzhao Lian

June 15, 2019

Contents

Acronyms

AGC	Automatic Generation Control. 4, 5, 11, 14–16
BEMS	Building Energy Management System. 4, 5, 9, 10, 12
DR	Demand Response. 4, 5, 11
ESS	Electrical Storage Systems. 11, 15
GP	Gaussian Process. 3, 5–13, 15, 16, 19, 20, 22
HP	Heat Pump. 6, 9, 18
HVAC	Heating, Ventilation, Air Conditioning. 4
MPC	Model Predictive Control. 5, 9–16, 18, 20, 22
MSLL	Mean standard log loss. 12, 20
PRBS	Pseudo-Random Binary Signal. 7, 16
RMSE	Root-mean squared error. 12, 20
SMSE	Standardised mean-squared error. 12, 20
SQP	Sequential Quadratic Programming. 16
SS	State Space. 5–8, 10, 11, 18, 22

Nomenclature

General

a	AGC signal	–
Q_{sun}	Global Horizontal Irradiance	W/m^2
T_{out}	Outside Temperature	°C
T_{set}	Set Temperature	°C
T_{base}	Base Set Temperature	°C
Δ_T	Set Temperature flexibility	°C
P_i^h	Power consumption at step i over the prediction horizon h	kW
P_{max}	Maximal power consumption	kW
γ	Power flexibility	kW

Gaussian Processes

y_i	Experimental observation at i'th step	–
$\hat{y}_i$	Gaussian Process (GP) model prediction at i'th step	–
$\mathbb{E}(\mathbf{y})$	Mean of the experimental measures	–
σ_y^2	Variance of the experimental measures	–
σ_i^2	Variance of the model prediction at i'th step	–

1 Introduction

The traditional way of supplying electric power to a consumer is simple and straight-forward: the supplier adjusts its energy supply volume so that it tracks the current combined consumption of all the consumers, and the system is in equilibrium.

With the increasing number of electrical devices and the steady shift of electrical power consumption from industry dominated to residential and service sectors [1] the consumer side of the equation has become not only more demanding, but also more volatile, with peak power demands being possibly larger than the total power supply.

On the first side, this increased volatility of the electrical grid also means that it is becoming less and less economically viable for the supplier to dynamically adjust its supply levels (which is usually done by spinning up or turning down turbines for traditional power plants or by adjusting the water level in a set of reservoirs in the case of hydroelectric plants[2]).

On the flip side, with the advent of *smart grids*, consumers can also supply electricity back to the grid, such as those using solar panels, batteries and other means of energy conversion and storage.

Finally, the smaller *consumption inertia* of the consumer side can be leveraged and used as complementary adjustment for the supply side. Now both the supply and the demand side of the grid can be adjusted dynamically to keep the grid balanced. Giving this agency to the consumers is the main idea behind *Demand Response (DR)*, in which the stability of the electrical grid is regulated both by changing the amount of energy supplied by the plants, as well as incentivising some clients to alter their consumption (either increase or decrease), deviating it from the baseline.

1.1 Previous studies in Demand Response Control

This new "degree of freedom" in the operation of electrical grids provides an interesting area of research once the question of *optimality* comes in. From the consumer's point of view, the optimal solution of this regulation problem depends on the definition of their *optimization objective*, which could be set as the minimization of the tracking error of the Automatic Generation Control (AGC) signal and the consumer's power consumption (i.e. individual building, group of buildings, battery or any other agent).

This has therefore become an interesting area of research. Multiple approaches have been taken for the implementation of these ideas, such as the ones presented in [3], [4], [5], [6]. They have taken such approaches as controlling the response of a single building, combining it with the use of an additional energy storage in the form of , or taking the role of an aggregator with the task of controlling a set of multiple buildings.

1.2 High level Demand Response Control

The implementation of such DR control, while very desirable, might turn out to be impractical for buildings already in exploitation, who most likely have an existing Building Energy Management System (BEMS), built to follow a particular temperature/power consumption reference, and not providing direct access to all its sensors and internal states.

It would therefore be very helpful to have a method of providing DR control capabilities to buildings already disposing of an integrated BEMS, by including it in the building's control model. Such a building model would also be helpful at abstracting away some of the constraints of more traditional DR control, since the building's Incorporated system would be responsible of complying with those constraints.

DR Control in the scope of this project

For the particular case of this project the building to be controlled is the EPFL's *Polydôme* building, since it already disposes of an integrated Heating, Ventilation, Air Conditioning (HVAC) system, as well as having access to past experimental data for the identification of it's thermodynamic model.

The goals of this project could then be naturally divided in two complementary parts: The first part, consisting in the *identification* of a GP model emulating the system's thermodynamic response, and the second part, consisting of the *design of a Model Predictive Control (MPC) scheme* to respond to the incoming AGC requests.

2 System Identification

In order to have a viable MPC controller, the plant's behaviour should be replicated as close as possible since at each step the objective function is minimized over the *prediction horizon*, and therefore over the prediction of the future states of the plant, based on the provided model.

An MPC control scheme is a particular case of an *optimization problem*. There exist a number of efficient algorithms for solving a particular subset of these optimization problems, called *convex optimization problems*. In order for an optimization problem to be convex, both its objective function and all of its constraints have to be convex functions.

The obvious choice of system model to have a convex MPC problem is to use a *State Space (SS)* representation. It would therefore only be left to ensure that all the other constraints of the system are convex, which is, in most cases, a weak assumption.

This was the case for the previous study conducted on Polydôme [5], which approximated the building's thermal behaviour with a third order SS model, resulting in a convex MPC problem.

Sadly, only very simple systems can be represented by a linear model, and most systems exhibit, to a certain degree and in certain situations at least, non-linear behaviour.

Most commercial buildings with integrated BEMS have integrated rules that maintain the temperature, power consumption and other within a certain range. In certain implementations, these controllers can make decisions based on discrete logic, depending on the type of the controller installed, the number and variety of sensors, etc.

In the case of such non-linear behaviour, these non-linearities should be incorporated into the plant's model or the MPC model risks giving erroneous state predictions, minimizing an objective that does not necessarily make sense anymore. One approach, taken in [3], involves building a data-driven GP plant model. In this case the optimization problem becomes non-convex, requiring the use of *global optimization* algorithms, but gives us a more representative behaviour of the real system, where such behaviour could not be captured otherwise. This is the identification approach taken during this project.

The identification of the *Polydôme's thermodynamic response model* is to be done on a set of experimental data, previously used for identification of a *SS Model*. The specific dates of the experiments are presented in Table 3, as well as an overview of the available experimental data in the whole of Annex A.

2.1 Three different views

In the ideal case, a very *high-level* model of the building's dynamics is going to be identified. This would allow an aggregator to consider the whole building, possibly in conjunction with other buildings or electrical generation/storage units, as an additional degree of freedom in solving the DR *the bidding problem*, namely calculating the amount of *power flexibility* that the consumer can provide to the grid operator. In order to find this final model and better understand its overall behaviour it is desirable, if not required, to start with simpler models and use them as an additional verification of the more complex models.

The identification of the Polydôme thermodynamic and DR models is therefore going to be made on three different views, from a lower to a higher-level view, which also mean an increase in the plant dynamics' complexity. General information on GP models and their use cases is presented in Annex B and a presentation of the experimental data available for the identification is given in Annex A.

2.1.1 First Level Model

The simplest modelization of the building only takes into account the change of internal temperature as a response to the power provided by the building's Heat Pump (HP). A schematic representation is shown in Figure 1.

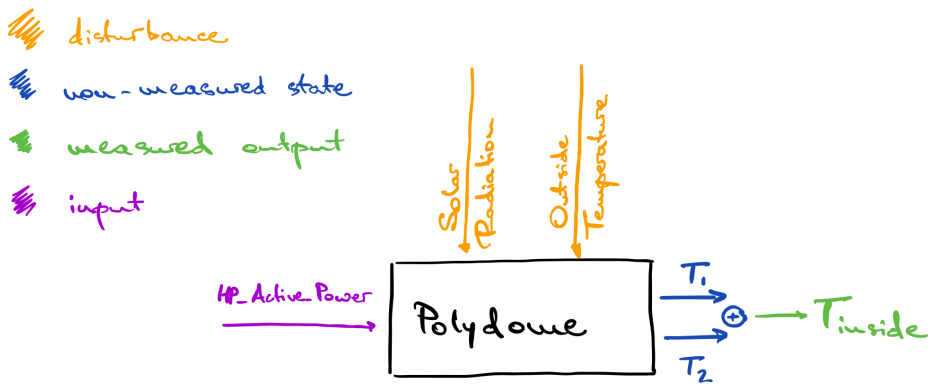


Figure 1: *Level 1 Model block diagram*

This low-level view of the Polydome presents an interesting perspective for the usability of GP models based on real data in the case of low-complexity plants. More so, this being the plant identified in [5] as a SS model, a comparison of a new SS model based on the same data with the performance of the GP will be included.

The experiments were conducted over a period of four weeks, with the data being divided between an identification and a validation set. The data for the first two days of the experiments is presented in Figure 2.

```
Data_ident = merge(Exp7_iddata, Exp6_iddata, Exp4_iddata, Exp2_iddata);
Data_val = merge(Exp1_iddata, Exp5_iddata, Exp3_iddata);
save([save_dir 'Identification_Validation.mat'], 'Data_ident', 'Data_val')
```

The SS model was identified using the MATLAB System Identification Toolbox. The model with the least overall error on the validation data set was chosen.

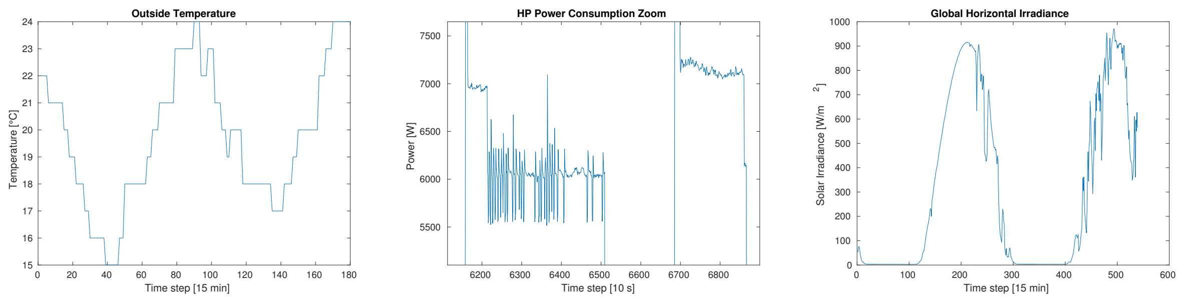
```
nx = 3;
opt = ssestOptions('SearchMethod','auto');
opt.Focus = 'prediction';
opt.N4Horizon = 6*3600/T_m * [1 1 1];

[sys, x0] = n4sid(Data_ident, nx, opt);
```

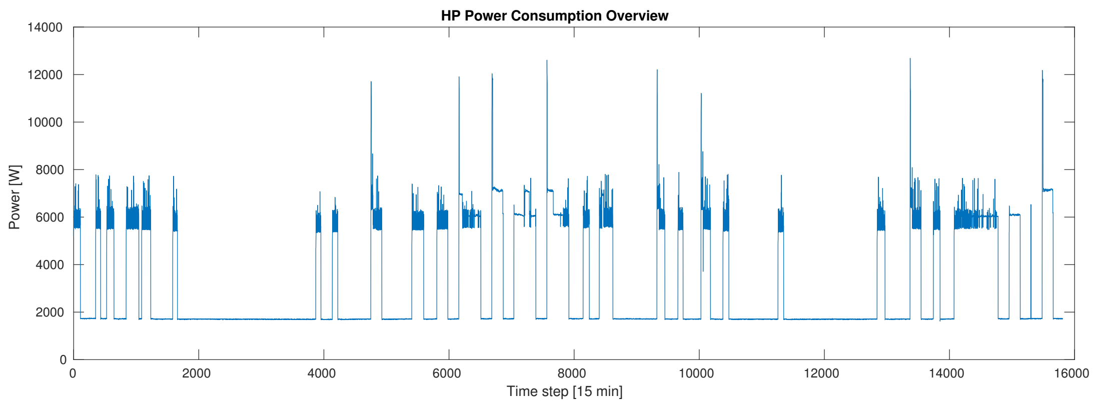
The GP model was identified on the same data-sets using the built-in MATLAB function *fitgp*.

Since the Polydome represents a dynamical system (cf. Annex B for details on representing dynamical systems with GPs), its future states are going to be based not only on the present, but also the past states. To include this information in the GP model that is to be identified a certain number of past inputs and outputs can be included along the current information about the system and disturbances. These additional inputs are known as the *lags*, denoted l_u for the input lags and l_y for the output lags. In the first instance, the lags considered for this model are $l_u = 1$, $l_y = 3$, but different models will be compared for the final, high-level identification to give a better fitting model.

The kernel used for the GP identification is the standard kernel used for representation of dynamical systems, the Automatic Relevance Determination (ARD) Squared Exponential Kernel, which is an implementation of the Squared Exponential Kernel that provides a separate length scale for each model predictor:



(a) *Overview of the three system inputs*



(b) *Broad overview of the HP Power consumption*

Figure 2: *Overview of the experimental data over the first two days*

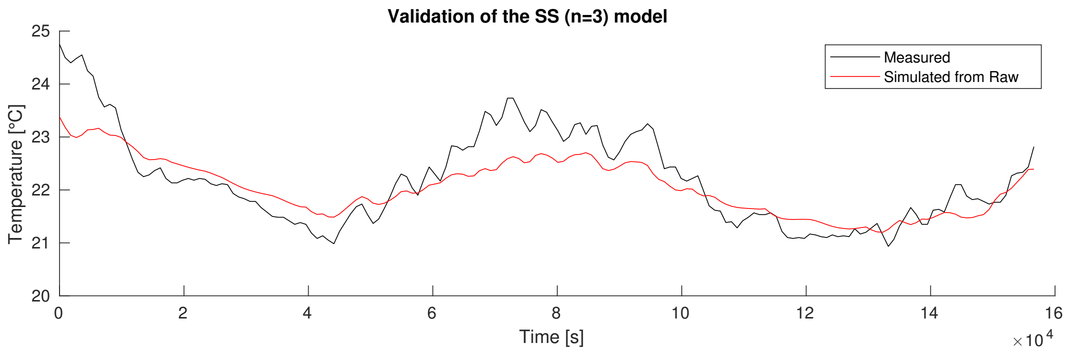


Figure 3: *Validation of the third order State Space Model*

$$k(x_i, x_j|\theta) = \sigma_f^2 \text{exp}\left[-\frac{1}{2}\sum_{m=1}^{d}\frac{(x_{im}-x_{jm})^2}{\sigma_m^2}\right] \tag{2.1}$$

A good choice of input signal when identifying a system is the Pseudo-Random Binary Signal (PRBS) signal, which by it's pseudo-random nature, includes a large set of frequencies, without any prominent peaks. The Fourier Transform of the input signal used for *Experiment 1* is presented in Figure 4, where it is possible to observe that, within the transform's resolution, the signal presents some of these desirable qualities, and, while not perfect, provides a good enough input to identify the behaviour of the system.

A total of 1109 experimental points was used to identify the GP model. The results, consisting of the predicted outputs for each input, the 95% confidence interval, the SS model output for the same inputs and the real experimental output is presented in Figure 5.

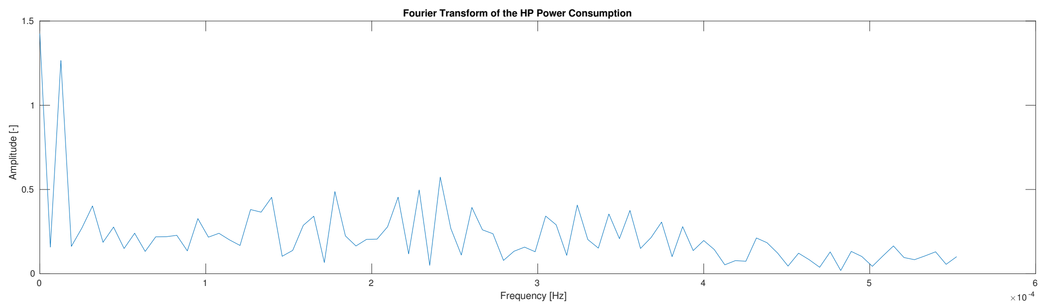


Figure 4: *Frequency Spectrum of the input signal (power consumption) used for identification*

```matlab
if lu > 0
    u_lag_1 = [0;table2array(Data_ident_table(1:end-1,2))];
    Data_id_ar = addvars(Data_id_ar, u_lag_1, 'After', ControlledInputName);
end
for idx = 2:lu
    eval(['u_lag_' int2str(idx) ' = [0;u_lag_' int2str(idx-1) '(1:end-1)];'])
    eval(['Data_id_ar = addvars(Data_id_ar, u_lag_' int2str(idx) ', ''After'', ...
        ''u_lag_' int2str(idx-1) ''');'])
end

% Default value: ly = 3
if ly > 0
    y_lag_1 = [0;table2array(Data_ident_table(1:end-1, 4))];
    Data_id_ar = addvars(Data_id_ar, y_lag_1, 'After', DisturbanceNames{end});
end
for idx = 2:ly
    eval(['y_lag_' int2str(idx) ' = [0;y_lag_' int2str(idx-1) '(1:end-1)];'])
    eval(['Data_id_ar = addvars(Data_id_ar, y_lag_' int2str(idx) ', ''After'', ...
        ''y_lag_' int2str(idx-1) ''');'])
end

% Delete the first rows, that have no lag info
clr_rows = max(lu,ly) + 1;
Data_id_ar(1:clr_rows,:) = [];

gprMdl_ar = fitrgp(Data_id_ar, OutputName, 'KernelFunction', ...
    'ardsquaredexponential', ...
                'FitMethod', 'sr', 'PredictMethod', 'fic', 'Standardize', 1);
```

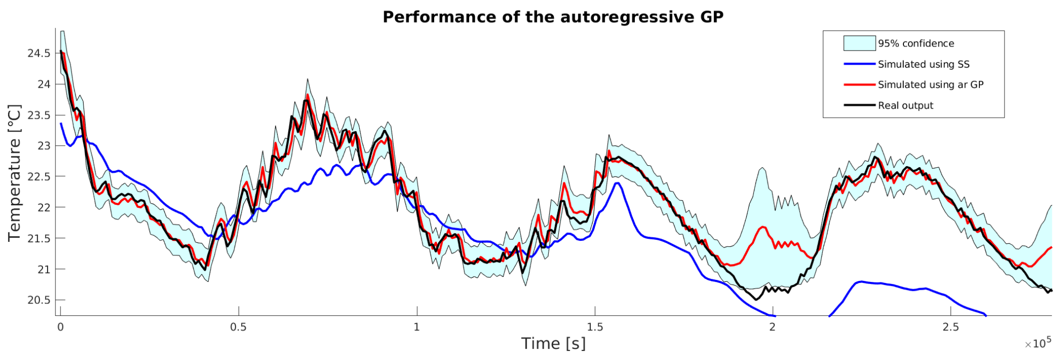


Figure 5: *Performance of the level 1 autoregressive GP model on the validation dataset*

At a first glance, we can see that the GP model is more accurate at predicting the behaviour of the system. At most points in the validation set the error of the 95% confidence interval level is lower than that of the SS model. Due to the non-linearities in the system, the discrepancy of the SS model and the real plant winds up over time. Even though this is already an important difference with the GP model, which is able to capture these non-linearities in the data, this is not a vital flaw in real use, as the validation graph shows the evolution of the system over a week's worth of data, but the MPC control scheme would only look at a prediction horizon of at most a few dozen steps to make the optimization.

2.1.2 Second Level Model

Having tested the ability of the GP model to capture the thermal dynamics of the building, the natural next step is re-identifying the system from a higher level view, this time including the building's HP and its embedded controller, the Polydome's BEMS. A block diagram of the new system is presented in Figure 6.

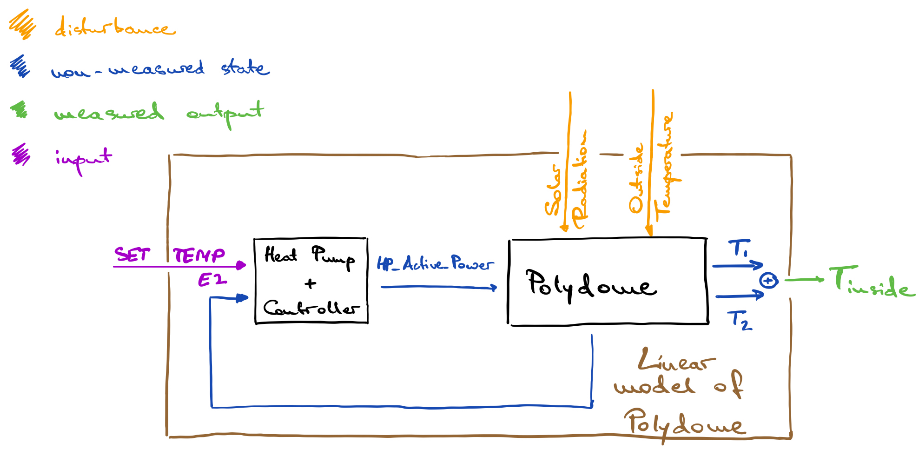


Figure 6: *Level 2 Model block diagram*

The same identification data was used, as this is the only time the input signals had enough composing frequencies to excite the system. The GP model was then identified following the same procedure as the one described in Section 2.1.1, but this time the *Set Temperature* was given as the controlled input. Its frequency spectrum is presented in Figure 7.

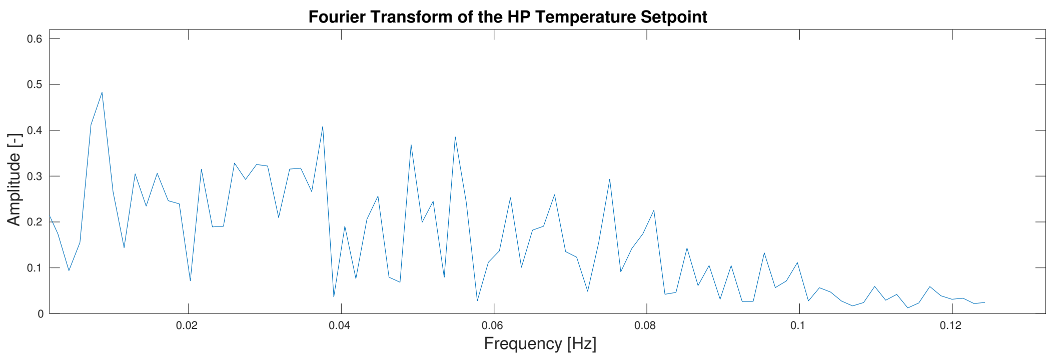


Figure 7: *Frequency Spectrum of the Set Temperature signal*

The performance of this new GP on the validation dataset is presented in Figure 8:

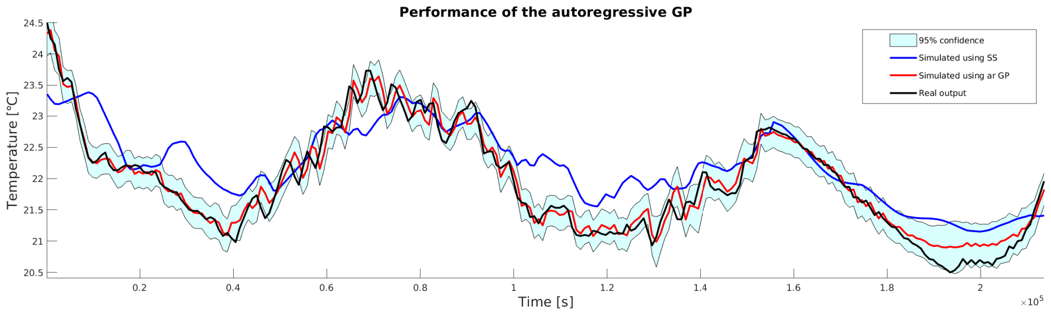


Figure 8: *Performance of the level 2 autoregressive GP model on the validation dataset*

We can see that in this situation the behaviour of the system can be better approximated by a linear SS model. The GP model also presents more accurate predictions. This result is consistent with our basic expectation: in this configuration, the closed-loop system consisting of the Polydome building and its controller is tracking the reference temperature, given as input.

While this model presents very desirable qualities, it has a few drawbacks that prevent us from using it for the MPC control scheme:

• The final GP model gives no information on the power consumption, so imposing constraints or optimizing the power consumption during the optimization phase of the MPC would require the use of an additional model.

• The frequency of the *Set Temperature* signal is much higher than that of the incoming AGC signal, which has typical frequencies in the range of at most 0.02 Hz [5]. The reason for this discrepancy is that in [5]'s implementation, the MPC controller responsible of the building's operation is based on a "low level" view of the building and an additional function which, for a given reference power over the next 15 minutes, $\hat{P}$, gives a set of *Set Temperatures* with a step time of ≈ 10s, so that the mean power consumption of the building approaches $\hat{P}$.

The overall result of the second level model proves that it is possible to accurately model the thermal behaviour of a building that includes its own simple BEMS control circuit.

2.1.3 Third Level Model

Since the final goal of this project is to have a high-level view of the system, which includes its control schemes, and does not require additional sensors, we now turn our attention to the highest level view, the one presented in Figure 9.

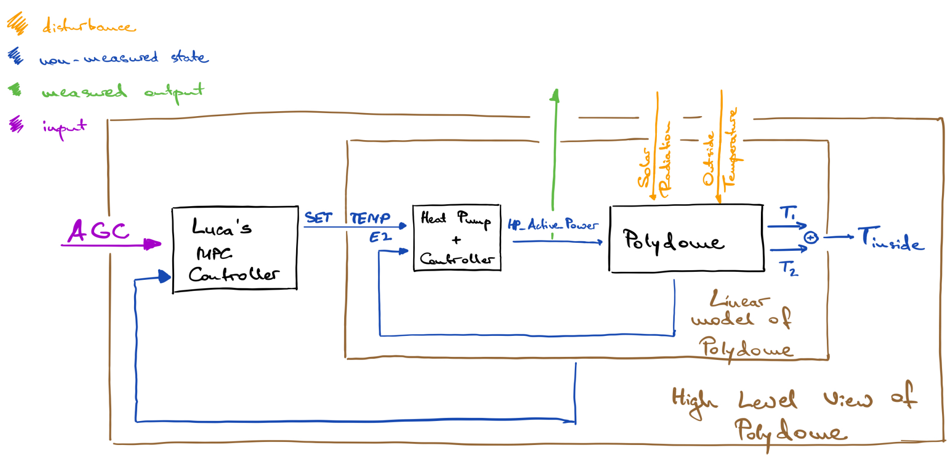


Figure 9: *Level 3 Model block diagram*

In this case we only look at the information that would be available to an aggregator, an entity which could be responsible for the generation of the input signals for (potentially) multiple buildings and with no access to the low-level information (e.g. the inside temperature measurements) that the building administrator would have.

The only output which an aggregator would have simple access to, without requiring the need of installing additional infrastructure is the building's *power consumption*. Since this highest-level view already includes an MPC control scheme responding to an AGC signal, this would have to be the model's input.

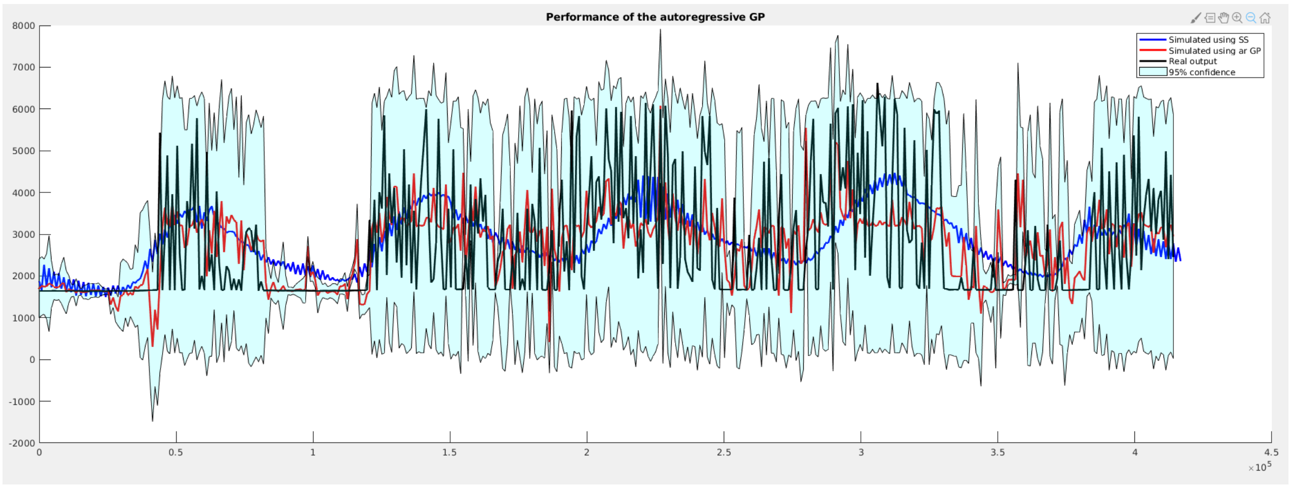


Figure 10: *Performance of the level 3 autoregressive GP model on the validation dataset*

The performance of this model, presented in Figure 10 is underwhelming. We can see that neither the SS nor the GP can capture the system's dynamics.

This result may be due to several factors, that make this view of the plant very different from the first ones:

- The Power Consumption does not represent neither a continuous, nor a simple switched system (as can be seen in Figure 3). Since the building's integrated controller could make decisions based on discrete logic when deciding the heat output/power consumption, it becomes vastly more complicated to predict its behaviour.
- Since the building's internal temperature represents a periodic signal that closely resembles the period of the two disturbance inputs, The Outside Temperature and the Global Horizontal Irradiance, the GP model gives much larger weights to those inputs, potentially completely ignoring the AGC signal.
- The MPC control implementation used in [5] includes implementations of Electrical Storage Systems (ESS) in the form of a battery, as well as adjusting the AGC signal through the use of the *Intraday Energy Market*. This makes the system an *open system*, since some of the power consumption either gets distributed to, or compensated by these external elements.

This high-level view of the Polydome's DR model would give much help in the simplification of the further MPC control problem, since in this case the majority of the comfort and maximum allowable energy consumption constraints would be handled by the building's controller and the appended MPC controller that has already been installed.

The building's complex power consumption cannot be described by the GP model. A more suited building for directly outputting the power consumption would be a more complex office/residential building, as the ones studied in [3], which presents a continuous variation of the power consumption.

2.2 The model used for MPC

The second level model presented very interesting traits, but due to the absence of Power Consumption information in this view, it is more versatile to use the *first level* model. In practice, the experimental system in [5] also incorporates a *Set Temperature* function which is be responsible for guaranteeing a mean power consumption as close as possible to the input signal by varying the BEMS set temperature in a rapid manner. This means that this view can be used in practice for predicting the behaviour of the closed-loop system including both the BEMS controller, and the *Set Temperature* function responsible of tracking the reference power signal.

This view of the system gives us more versatility in solving the MPC problem, since we could impose restrictions on power consumption as well as inside temperature of the building. The draw-backs are that, in the case of the aggregator, temperature sensors would have to be taken into account, which could mean a change in control/sensor infrastructure of the building.

Having chosen the model to be used in the control scheme, it is now time to turn to tuning its parameters.

Since this model is going to be used to predict the system's behaviour for the whole prediction horizon, an accurate representation is needed to minimize the error. This can be accomplished by building a GP model of higher degree, by including a larger number of inputs. The upside of this would be that the system would have a larger liberty in distributing the weights of these inputs for predicting the future states.

The GPs implemented in [3] use the hour of the day as an additional input to the model. This allows assigning some weights to this input that are representative of the overall building's periodic behaviour, which was noted to have power consumption that are closely related with each other on workdays, providing an output with a period of around 24 hours.

In the case of the Polydome, the two disturbance inputs are already presenting a prominent period of 24 hours, and thus, adding another signal of the same periodicity would only diminish the weights attributed to the controlled signal.

One other way of augmenting the system's degree is to include more information of the past states of the system, since those measurements are readily available in an experiment.

A number of models of different lags, ranging from 1 to 3 for the past inputs, and 2 to 4 for past outputs have been considered, and the best fit was selected for use in the MPC scheme.

For validation, a large number of different criteria can be used. For example, three indices of the system's performance are the Root-mean squared error (RMSE), Standardised mean-squared error (SMSE), Mean standard log loss (MSLL) [7]. These values, along with the length scales corresponding to the particular lags, are presented for the final model in Table 1.

Model	RMSE	SMSE	MSLL	λ_{y4}	λ_{y3}	λ_{y2}	λ_{y1}	λ_{u3}	λ_{u2}	λ_{u1}	λ_{u0}
$\mathcal{M}(l_u = 0, l_y = 0)$	1.8337	5.8276	-0.204								3.6413
$\mathcal{M}(l_u = 1, l_y = 2)$	0.1548	0.041572	-1.4431			5.2163	1.5967			8532.1	130.31
$\mathcal{M}(l_u = 2, l_y = 3)$	0.20755	0.074801	-1.4265		1.3146	4.9561	1.1538		29.541	1406.9	14.72
$\mathcal{M}(l_u = 3, l_y = 4)$	0.0041869	3.04E-05	-1.3589	2.324	1.4663	2.6068	1.0987	5.3414	14.167	8.2271	11.345

Table 1: *Errors and length scales for different lags for the final model*

At first glance, the errors between the GP model output and the experimental outputs decrease with the increase of the order of the lags. This is to be expected, since the model has access to more parameters that it can tune to better copy the behaviour of the system.

One problem with this result could be that of an over-fitted model. With the increase of model degree, there exists a risk that it will start getting more and more influenced by the noise of the input signals [8]. This would actually reduce the accuracy of the predictions in the case of a larger prediction horizon, where the output of the model is fed back into the model, without having yet access to any real new information.

While a data set of over 1000 points should be enough to accurately train a GP model of degree 10 [8], in this experiment the available data could prove to be too noisy and over a too narrow frequency bandwidth to provide an accurate representation for such a large number of inputs.

The model with the best validation scores ($\mathcal{M}(l_u = 3, l_y = 4)$) is compared with the model discussed in Section 2.1.1 in a multistep simulation of the GP model. In this case, the simulation method used is the *zero-variance method*. This method does not propagate the output uncertainty over the future prediction steps [7]. This approach underestimates the real variance of the output, but is simpler and faster from a computational point of view. Since the error induced by not propagating the variance is small compared to the other sources of error (weather forecast, the averaging of power consumption over a long period, variance of the temperature sensors), this simulation method is more than appropriate.

The comparison of different prediction horizons is presented for $\mathcal{M}(l_u = 1, l_y = 3)$ (cf. Figure 11) and for for $\mathcal{M}(l_u = 3, l_y = 4)$ (cf. Figure 12). The simulation starts with one step of experimental information, simulates the outputs for the length of the prediction horizon, then resets with the new experimental outputs.

These results show that increasing the prediction horizon for the multistep simulation decreases the accuracy of the system. Nonetheless, the model presented in Figure 11 exhibits less overall error and is less aggressive than the more complex system in Figure 12.

During the MPC optimization step, the simpler model was better at predicting the behaviour of the system for new, unseen before data. The more complex model would give optimization results that were not at all in agreement with our intuition, even in simple optimization scenarios. This strongly suggests that the more complex system was over-fitted to better predict the validation data, resulting in a system that is influenced by the random noise in the system more than by the controllable input.

The final choice of the model to be used by the MPC scheme is therefore the model presented in Section 2.1.1.

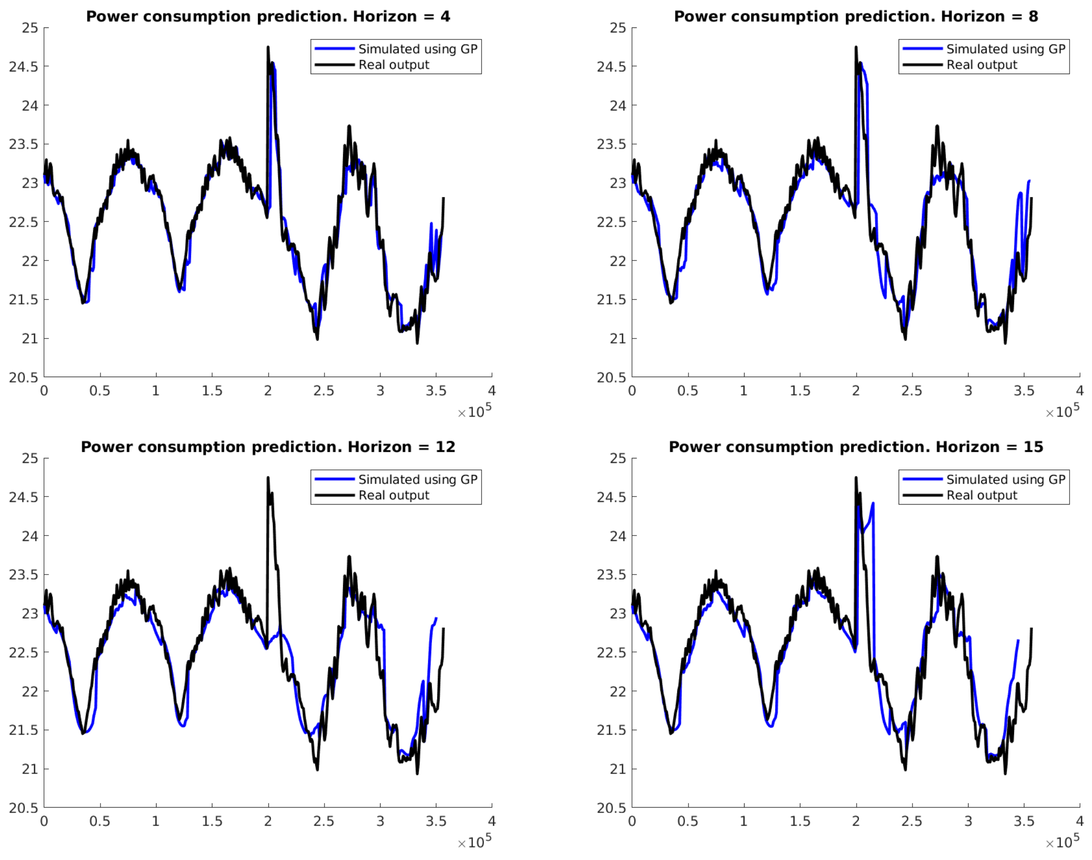


Figure 11: *Multistep predictions for* $\mathcal{M}(l_u = 1, l_y = 3)$

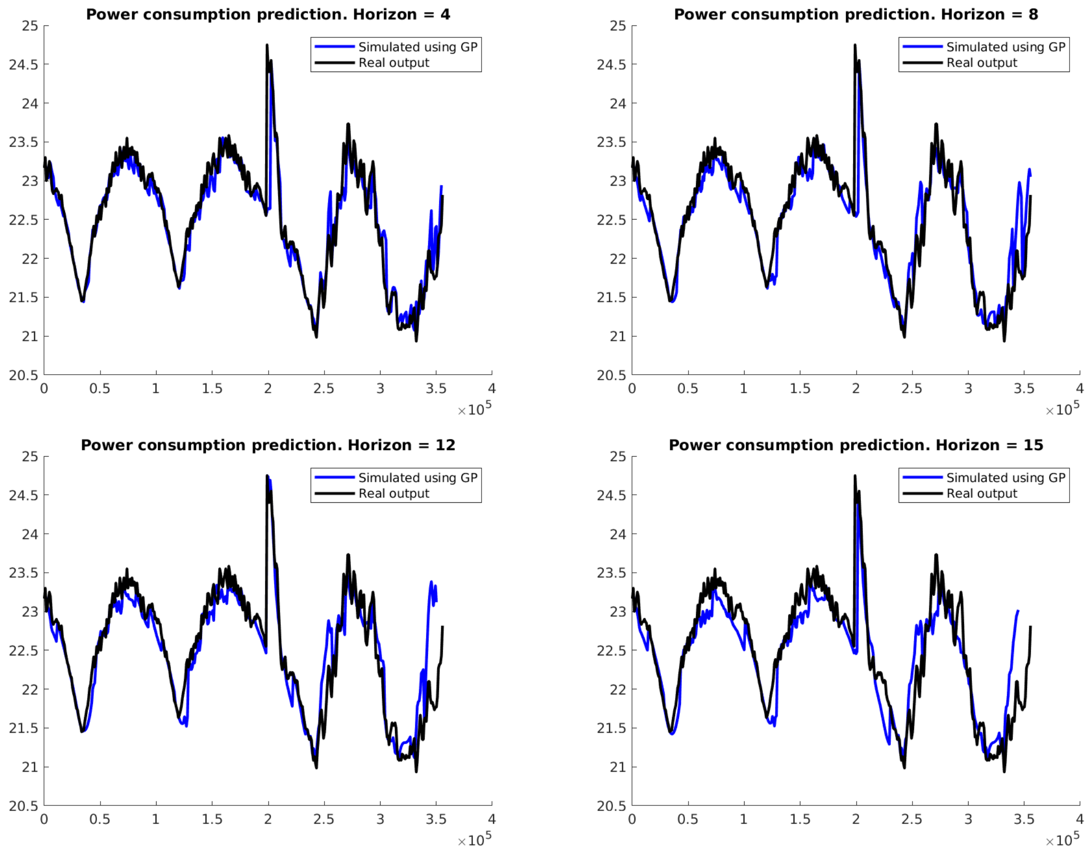


Figure 12: *Multistep predictions for* $\mathcal{M}(l_u = 3, l_y = 4)$

3 Demand Response Control

The MPC problem is posed based on the final model chosen for prediction and its available input/output information. In this case, since the chosen model represents the *low-level view* of the Polydome's thermodynamic response, the job of the controller is not only to track the AGC demand signal, but also to ensure the compliance of the comfort constraints.

In its simplest implementation, an MPC control scheme can respond to a normalized AGC signal ($a \in [0, 1]$) which requires the direct adjustment of the inside temperature. The set temperature for the next step would become:

$$T_{set} = T_{base} + \Delta_T \, a \tag{3.1}$$

The goal in this case would be to minimize the largest difference between T_{in} and T_{set}, which gives rise to the first MPC problem:

$$\begin{aligned} \min_{\mathbf{P}^h} \quad & \|T_{set} - T_{in}\|_\infty \\ s.t. \quad & T_{in,i} = g_m(\mathbf{x}_i^h) \\ & 0 \le P_i^h \le P_{max} \\ & T_{set,i} = T_{base} + \Delta_T \, a_i \end{aligned} \tag{3.2}$$

Where

$$\begin{aligned} \mathbf{x}_i^h &= [y_{i-l_y}, \dots, y_{i-1}, \dots, u_{i-l_u}, \dots, u_i, w_i]^T \\ u_i &= P_i^h \\ y_i &= T_{in,i} \\ w_i &= [T_{out}, Q_{sun}]_i^T \end{aligned}$$

is the input of the GP model g_m.

This formulation implicitly imposes hard constraints on the temperature inside the building: assuming the AGC signal a is normalized, the set temperature T_{set} for the next step will never get outside of the range $[T_{base} - \Delta_T; T_{base} + \Delta_T]$. In the controller's implementation the temperature constraints have been chosen as follows:

$$\begin{aligned} T_{base} &= 22 \quad ^\circ\text{C} \\ \Delta_T &= 2 \quad ^\circ\text{C} \end{aligned}$$

This control scheme is quite simple to implement since the constraints on temperature do not have to be defined separately. The drawback is that it is not easily extensible to the use of ESS, multiple buildings, or any other power compensation methods. **This is the MPC control scheme that has been implemented in this project.**

The next step is to extend the MPC formulation to the tracking of the power AGC signal. In this case the goal becomes that of minimizing the total difference between the real power consumption and the prescribed AGC signal.

$$\begin{aligned} \min_{\mathbf{P}^h} \quad & \left\| \mathbf{P}^h - \gamma\, \mathbf{a} \right\|_2 \\ s.t. \quad & T_{in,i} = g_m(\mathbf{x}_i^h) \\ & 0 \le P_i^h \le P_{max} \\ & T_{in} \in [T_{base} \pm \Delta_T] \\ & \gamma > 0 \end{aligned} \tag{3.3}$$

The power flexibility, γ, can be passed as a parameter to the MPC controller, which could allow the implementation of this controller in larger, distributed control schemes, such as those presented in [5] and [3].

In this formulation a hard constraint has been imposed on T_{in} but a soft constraint with a penalty on the objective function for its violation could also be implemented.

4 Experimental validation and possible future steps

The MPC problem defined in Section 3 is non-convex due to the use of a GP model. Its implementation requires, therefore, the use of specialized optimization algorithms. For this project, the choice to use a *genetic algorithm* for solving the MPC problem has been made. This is due to its rather simple implementation,large versatility as well as its ability to easily scale to a larger number of generations, or reduce the total computation time by parallelization [9]. The drawbacks of this optimization algorithm are that, in the MATLAB implementation, the *genetic algorithm* optimizer is unable to use any GPU resources, which reduces the computation speed.

The MPC controller defined in (3.3) has been tested on the experimental data used for the validation of the models presented in [5], since during that time period all the necessary input signals were active.

The current implementation of the problem takes around 10 minutes to solve for a prediction horizon of 10 steps, which is an acceptable time-frame, even if higher than the expected for a problem of this complexity.

A faster implementation could be achieved by exploiting the twice-differentiability of the GP kernel to find the local optima using Sequential Quadratic Programming (SQP)-capable solvers, such as IPOPT and CasADi [10].

A next step in this project would be to either test a version of the controller implemented in (3.3) experimentally or through the use of an EnergyPlus building model. The benefits of the first approach are that it would provide experimental confirmation of the results presented in the previous sections, while choosing the second approach would allow for a simpler tuning of the MPC controller parameters, as well as an easier comparison of different optimization schemes.

5 Conclusion

The goal of this project was to identify a GP model of the thermal behaviour the EPFL's Polydome building with the use of already available experimental data and construct an MPC control scheme that would be capable of minimizing the tracking error of an AGC signal while ensuring the compliance with the building's comfort constraints. A number of different models was analyzed, and the best-fitting model was chosen and tuned to provide a better multistep prediction for solving the MPC optimization problem. Finally an MPC control scheme was designed and implemented in MATLAB using a *genetic algorithm* global optimizer.

It is of note that even with the input signal not being a PRBS signal, it was still possible to identify a viable model. This means that it could be possible to run experiments to identify an office/residential building without much disturbance to it's normal performance.

The use of a *genetic algorithm* global optimizer allowed for a very large versatility in the definition of the problem, even allowing for the use of discrete logic and easy implementation of hard constraints on the output of the GP model, as well as being potentially easier to parallelize. On the flip side, this solution is quite computationally expensive for a problem with this small of a complexity.

Acknowledgements

I would like to thank Prof. Colin Jones for the opportunity of working on this project that combines multiple very interesting subjects to solve a practical problem.

Additionally I take this opportunity to thank Emilio Maddalena and Yingzhao Lian who not only gave me a crash course in MPC control and GP modeling of dynamic systems, but also supported me throughout this project by answering all my questions and listening to my (sometimes impractical) ideas.

References

[1] Paolo Bertoldi and Bogdan Atanasiu. "Electricity Consumption and Efficiency Trends in European Union". en. In: (), p. 95.

[2] "Hydroelectric Power". English. In: *Reclamation: Managing Water in the West* (July 2005).

[3] Truong X. Nghiem and Colin N. Jones. "Data-Driven Demand Response Modeling and Control of Buildings with Gaussian Processes". en. In: *2017 American Control Conference (ACC)*. Seattle, WA, USA: IEEE, May 2017, pp. 2919–2924. ISBN: 978-1-5090-5992-8. DOI: 10.23919/ACC.2017.7963394.

[4] Luca Fabietti et al. "Experimental Implementation of Frequency Regulation Services Using Commercial Buildings". en. In: *IEEE Transactions on Smart Grid* 9.3 (May 2018), pp. 1657–1666. ISSN: 1949-3053, 1949-3061. DOI: 10.1109/TSG.2016.2597002.

[5] Luca Fabietti et al. "Multi-Time Scale Coordination of Complementary Resources for the Provision of Ancillary Services". en. In: *Applied Energy* 229 (Nov. 2018), pp. 1164–1180. ISSN: 03062619. DOI: 10.1016/j.apenergy.2018.08.045.

[6] Kevin J. Kircher and K. Max Zhang. "Model Predictive Control of Thermal Storage for Demand Response". en. In: *2015 American Control Conference (ACC)*. Chicago, IL, USA: IEEE, July 2015, pp. 956–961. ISBN: 978-1-4799-8684-2. DOI: 10.1109/ACC.2015.7170857.

[7] Juš Kocijan. *Modelling and Control of Dynamic Systems Using Gaussian Process Models*. Advances in Industrial Control. Cham: Springer International Publishing, 2016. ISBN: 978-3-319-21020-9 978-3-319-21021-6. DOI: 10.1007/978-3-319-21021-6.

[8] Rekar O. Mohammed and Gavin C. Cawley. "Over-Fitting in Model Selection with Gaussian Process Regression". In: *Machine Learning and Data Mining in Pattern Recognition*. Ed. by Petra Perner. Vol. 10358. Cham: Springer International Publishing, 2017, pp. 192–205. ISBN: 978-3-319-62415-0 978-3-319-62416-7. DOI: 10.1007/978-3-319-62416-7_14.

[9] Jarosław Jajczyk. "Genetic Algorithm and Parallel Computing". en. In: (), p. 2.

[10] Lukas Hewing, Juraj Kabzan, and Melanie N. Zeilinger. "Cautious Model Predictive Control Using Gaussian Process Regression". en. In: *arXiv:1705.10702 [cs, math]* (May 2017). arXiv: 1705.10702 [cs, math].

[11] Carl Edward Rasmussen and Christopher K. I. Williams. *Gaussian Processes for Machine Learning*. en. Adaptive Computation and Machine Learning. OCLC: ocm61285753. Cambridge, Mass: MIT Press, 2006. ISBN: 978-0-262-18253-9.

[12] *Kernel Cookbook*. https://www.cs.toronto.edu/ duvenaud/cookbook/.

Appendix

A Available Data

All the experimental data used for the identification of the systems, and validation of the MPC control scheme comes from the experimental setup used for [5]. The time-span of the data ranges from June to August 2017, beginning with the data sets used for the identification of the SS model and finishing with the experimental validation of the control scheme.

In light of the multi-scale nature of the original experiment, data was available for multiple inputs/outputs of the system, as well as the measurements and predictions for the exogenous variables, which allowed the analysis of multiple points of view of the system dynamics for this project. An overview of the available variables used in this project and their naming convention in the project's InfluxDB database is presented in Table 2.

Variable	InfluxDB name
Inside temperature sensor 1	TEMP_TSOL
Inside temperature sensor 2	TEMP_Lake
Real power consumption of the HP	HP_Active_Power
Forecast of the outside temperature	Tair
Realization of the outside temperature	OutsideTemp
Forecast of the solar irradiation	GHI (epfl_intraday)
Realization of the solar irradiation	GHI (PVroof-mirror)

Table 2: *Naming convention in the [5] experimental implementation*

The experimental data has been downloaded locally using Python, along with a number of libraries for data treatment and visualization. The most potentially useful data has been exported to MATLAB to be used for system identification and control. The jupyter notebook used for the retrieval and visualization of data is presented as an HTML document in the project's files (*Gaussiandome/Notebooks*).

Along with it, the python script *Gaussiandome/Identification/Utilities/Python/local_parse.py* provides an easy utility to extract the compressed data from the InfluxDB database. It is configured now to retrieve the experimental data for the dates provided in Table 3 since that is the data used for later system identification.

No.	Start Date	End Date
1	01.06.2017 20:00	03.06.2017 17:00
2	10.06.2017 16:00	12.06.2017 06:00
3	16.06.2017 20:00	19.06.2017 06:00
4	19.06.2017 20:00	22.06.2017 06:00
5	30.06.2017 20:00	03.07.2017 06:00
6	07.07.2017 20:00	10.07.2017 06:00
7	13.07.2017 20:00	20.07.2017 06:00
8	21.07.2017 20:00	24.07.2017 06:00

Table 3: *System Identification Experiment Dates*

The MPC control scheme has been validated on data collected during *08.08.2017 - 23.08.2017*. It has been exported using the same script presented before.

B Gaussian Processes

Gaussian Processes (GP) can be viewed as a generalization of the Gaussian probability distribution to functions. A formal definition [11] states that :

> A Gaussian process is a collection of random variables, any finite number of which have a joint Gaussian distribution.

A GP is completely specified by a mean and a co-variance function:

$$\begin{aligned} m(\mathbf{x}) &= \mathbb{E}[f(\mathbf{x})] \\ k(\mathbf{x}, \mathbf{x}') &= \mathbb{E}[(f(\mathbf{x}) - m(\mathbf{x}))(f(\mathbf{x}') - m(\mathbf{x}'))] \end{aligned} \tag{B.1}$$

This notation is commonly abbreviated as:

$$f(\mathbf{x}) \sim \mathcal{GP}(m(\mathbf{x}), k(\mathbf{x}, \mathbf{x}'))) \tag{B.2}$$

GPs are used to predict the behaviour of a function for any input based on the model identified by regression on a training data set. The outputs of the test outputs $\mathbf{f}_*$ are predicted using the joint distribution of the training and the test outputs:

$$\begin{bmatrix} \mathbf{f} \\ \mathbf{f}_* \end{bmatrix} = \mathcal{N}\left(\mathbf{0}, \begin{bmatrix} K(X,X) & K(X,X_*) \\ K(X_*,X) & K(X_*,X_*) \end{bmatrix}\right) \tag{B.3}$$

For prediction of a system based on noisy observations $\mathbf{y}$ (of mean 0 and variance σ_n^2) we can then write:

$$\begin{bmatrix} \mathbf{y} \\ \mathbf{f}_* \end{bmatrix} = \mathcal{N}\left(\mathbf{0}, \begin{bmatrix} K(X,X) + \sigma_n^2 I & K(X,X_*) \\ K(X_*,X) & K(X_*,X_*) \end{bmatrix}\right) \tag{B.4}$$

which gives us a *predictive distribution* of :

$$\begin{aligned} \mathbf{f}_* &= \mathbb{E}[f_*|X, \mathbf{y}, X_*] = K(X_*,X)[K(X,X) + \sigma_n^2 I]^{-1}\mathbf{y} \\ \text{cov}(\mathbf{f}_*) &= K(X_*,X_*) - K(X_*,X)[K(X,X) + \sigma_n^2 I]^{-1}K(X,X_*) \end{aligned} \tag{B.5}$$

A MATLAB implementation of these results for the training of a GP model and its use for prediction is presented in *Gaussiandome/Introduction/GaussianProcess.m*. For the rest of the project the MATLAB GP toolbox was used for speed optimization.

B.1 Usual choices of Kernel Functions

Choosing a good Kernel Function during the identification of the GP model can have beneficial effects on the utility of the results by providing a better *mean* approximation, as well as tighter bounds on the *output variance*.

Below is presented a list of the most popular kernels used in GP models, as presented in [12].

Probably the most used kernel is the *Squared Exponential Kernel*, since it is an universal kernel, that can be integrated against most functions

$$k_{SE}(x, x') = \sigma^2 \text{exp}\left(-\frac{(x - x')^2}{2l^2}\right) \tag{B.6}$$

The *Rational Quadratic Kernel* can be seen as an extension of the SE Kernel to multiple lengthscales:

$$k_{RQ}(x,x') = \sigma^2 \text{exp}\left(1 + \frac{(x-x')^2}{2\alpha l^2}\right)^{\alpha} \tag{B.7}$$

The *Periodic Kernels* allow to impose a fixed period *p* on the GP model:

$$k_{per}(x,x') = \sigma^2 \text{exp}\left(-\frac{2\sin^2(\pi|x-x'|/p)^2}{l^2}\right) \tag{B.8}$$

The most widely used Kernel in identification of dynamical models is the *Squared Exponential* kernel, and as such, it was also used in this project.

B.2 Algorithmic complexity of GP models

The most computationally expensive part of using GP models, apart from identification (which takes a significant amount of time) is the need of multiplying and inverting matrices, which has a time complexity of $\mathcal{O}(n^3)$, and as such dominates the overall complexity of the GP model prediction.

When paired with a *genetic algorithm* for optimization, the time complexity of the complete MPC problem becomes $\mathcal{O}(n^3ml)$, where m is the number of individuals in a generation and l is the limit of generations of the *genetic algorithm*.

B.3 Gaussian Processes in Dynamical Systems

A consequence of the versatility of GP models is their utility in the identification and simulation of nonlinear dynamical systems.

Nonlinear behaviour can express itself in unexpected and hard to model ways, which has meant that the (sometimes very strong) assumption of linearity has been usually made when identifying dynamical systems.

The black-box approach of identifying systems using GP models presents an alternative way of data-driven model identification, where the non-linearities of the model can still be well approximated even without their prior inclusion in the model.

B.4 Validation of GP models for use in Dynamical Systems

The validation of a GP based dynamical model comes from the necessity of agreement between the mathematical model and the real dynamical system that has been identified. Different weights are computed during the identification phase of the GP model, and the results are compared on a set of experimental data, called the *validation data* using different *performance measures*. The best model is then that model which presents the best performance measure.

The commonly used performance indices for the validation of GP modeled dynamical systems are as follows:

The standardised mean-squared error (SMSE):

$$\text{SMSE} = \frac{1}{N}\frac{\sum_{i=1}^{N}(y_i - \mathbb{E}(\hat{y}_i))^2}{\sigma_y^2} \tag{B.9}$$

The root mean-squared error (RMSE):

$$\text{RMSE} = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(y_i - \mathbb{E}(\hat{y}_i))^2} \tag{B.10}$$

The mean standardized log loss (MSLL):

$$\textsf{\textbf{MSLL}} = \frac{1}{2N}\sum_{i=1}^{N}\left[\textsf{ln}\left(\sigma_i^2\right) + \frac{\left(y_i - \mathbb{E}(\hat{y}_i)\right)^2}{\sigma_i^2}\right] - \frac{1}{2N}\sum_{i=1}^{N}\left[\textsf{ln}\left(\sigma_y^2\right) + \frac{\left(y_i - \mathbb{E}(\mathbf{y})\right)^2}{\sigma_y^2}\right] \tag{B.11}$$

C Project scripts

This section provides short notes on the implementation and usage of the MATLAB scripts written for this project.

raw_data_systemID

The script *Gaussiandome/Identification/raw_data_systemID* imports the experimental data exported by the python script, down-samples the data to the model time sample T_m, and then identifies and validates a SS model of order 3 and an auto-regressive GP model with the lags *lu* and *ly* given as parameters.

test_horizon

The script *Gaussiandome/Control/test_horizon.m* computes a multistep ahead prediction for the given prediction horizon using the GP model identified previously.

test_mpc

The script *Gaussiandome/Control/test_mpc.m* provides an implementation of the MPC scheme presented in Section 3 using MATLAB's implementation of a *genetic algorithm*.

Download of InfluxDB data

The experimental data which was saved in an external InfluxDB database is downloaded locally for further processing in Python. The scripts used in this part are found in *Gaussiandome/InfluxDB import*.

Analysis of the available experimental data

The local data was then parsed and analyzed locally with the use of Python. The results, along with some comments, are presented in *Gaussiandome/Notebooks*.