Available online at www.sciencedirect.com
ScienceDirect
Performance Evaluation 00 (2017) 1–33
Performance Evaluation

Certified policy synthesis for general Markov decision processes: An application in building automation systems

Sofie Haesaert[a], Nathalie Cauchi[b], Alessandro Abate[b]

[a]*Department of Electrical Engineering, Technische Universiteit Eindhoven, Eindhoven, Netherlands*
[b]*Department of Computer Science, University of Oxford, Wolfson Building, Parks Road, Oxford, United Kingdom*

Abstract

In this paper, we present an industrial application of new approximate similarity relations for Markov models, and show that they are key for the synthesis of control strategies. Typically, modern engineering systems are modelled using complex and high-order models which make the correct-by-design controller construction computationally hard. Using the new approximate similarity relations, this complexity is reduced and we provide certificates on the performance of the synthesised policies. The application deals with stochastic models for the thermal dynamics in a "smart building" setup: such building automation system set-up can be described by discrete-time Markov decision processes evolving over an uncountable state space and endowed with an output quantifying the room temperature. The new similarity relations draw a quantitative connection between different levels of model abstraction, and allow to quantitatively refine over complex models control strategies synthesised on simpler ones. The new relations, underpinned by the use of metrics, allow in particular for a useful trade-off between deviations over probability distributions on states and distances between model outputs. We develop a software toolbox supporting the application and the computational implementation of these new relations.

Keywords: Verification, Synthesis, general Markov decision processes, Safety, Building automation systems, Temperature control

1. Introduction

Buildings consume more than 40% of the energy in Europe [1] and in the United States [2]. The design of their heating, ventilation and air-conditioning (HVAC) systems has become a focal point not only in the buildings and HVAC community [3, 4], but also in the control community [5, 6, 7].

The goal of modern building automation systems (BAS) is to control the climate in an energy-efficient manner, while allowing for diverse and complex functionality imposed by the users. For this, a myriad of possible control strategies have been proposed, the most modern of which hinge on accurate mathematical models of the thermal dynamics [8, 9, 10]. The quantification of building dynamics leads to models evolving stochastically over continuous spaces. Stochastic dynamics are needed to capture the effect of external disturbances, such as weather fluctuations and occupancy changes, both of which can hardly be described with deterministic models. While accurate models are needed, their inherent level of complexity, related to stochasticity and dimensionality, limits the type of controllers that can be designed. Recent literature [11, 12, 13, 14] has seen control designs targeted to optimally maintain a comfortable inside climate. To satisfy the goal for future modern BAS, such control strategies should not only be optimal but also guarantee user-imposed diverse and complex requirements and specifications. The use of temporal logics allows for a formalised expression of such diverse functional requirements, and enables for the automated

verification and synthesis of control strategies [15, 16, 17]. An interesting formal approach is the approximation of the original (concrete) models by simpler (abstract) models that are prone to be analysed or algorithmically verified. Further, if a strategy can be synthesised on these abstractions, it should then be certifiably refined over the concrete model. Building on the similarity between the abstract and concrete models, key formal guarantees on the latter can be raised. In this work, we employ notions of approximate probabilistic similarity relations, originally introduced in [18]. Underpinned by the use of metrics, these relations allow in particular for a useful trade-off between deviations over probability distributions on states, and distances between model outputs. Other relations, also targeting general, uncountable-state spaces, such as those established via martingale theory [19, 20] require stringent stability criteria, or alternatively enforce structural abstractions [21] based on state-space gridding [22].

Within the control community, the computational complexity of a model is tackled by abstracting it to a lower dimensional one. Numerical model-order reduction (MOR) techniques reduce the dimensionality of models with minimal loss in input-output accuracy [9, 23, 24, 25, 26]. A number of different MOR techniques are in use [27]: Gramian-based MOR involve balancing the observability and controllability Gramians via truncations [27]; Schur-based MOR techniques compute a reduced-order system through a set of projections [28]; alternative MOR techniques reduce model orders with respect to the Hankel norm [29]. These numerical model-order reduction techniques are oblivious of the underlying physics associated to the dynamics and, as a result, the obtained low-order models often no longer have a physical interpretation. This can be of a disadvantage when faults arise and an analysis on the underlying model is needed to detect and interpret the physical source causing faults. In this paper, we perform model reduction based on assumptions made on the underlying physical quantities. We present a methodology to provide guarantees for control strategy synthesis over a library of thermal models. Models could in general be obtained via MOR techniques, however key to this library of models is that they maintain correspondences via the underlying physical variables. The provided guarantees are computed by a new toolbox that implements the synthesised control strategy via (ε, δ)-approximate simulation relations. The new relations serve as a performance metric between models of different orders of complexity and allow the user to choose the most suitable model abstraction for achieving the required level of performance by the synthesised and refined control policies. The toolbox is configured via a graphical user interface (GUI) that simplifies the certified policy synthesis process. Consequently, the toolbox facilitates the process of designing certified control strategies for BAS.

The used notions of approximate similarity relations for control refinement were first introduced in [18]. The theoretical background is given more extensively in [30], and a case study using FAUST^2 [17]. In this paper, we move beyond the purely theoretical treatment of these relations. For building management, we show how these relations can be used. More precisely, we explain the computational procedures that can be used to compute the accuracy of these relations for given models.

The article has the following structure: Section 2 introduces the problem statement and gives a demonstration of the developed policy synthesis toolbox. Section 3 introduces the library of models that capture the thermal dynamics of building zones. This is followed by the theory behind policy synthesis via approximate simulation relations in Section 4. Section 5 presents how certified control refinement based on the approximate simulation relations can be performed. The case study, which highlights the advantages and applicability of the devised framework, is presented in Section 6. This is then followed by the computational implementation of the toolbox in Section 7.

2. Certified control synthesis

With as goal the future industrial application of certified control synthesis for complex engineering systems, we want to develop a first application-oriented framework to enable the design and evaluation of this methodology. We target the application in BAS and single out an exemplar case study in room heating. We want to provide quantified performance of the synthesised policies with certificates.

Often a single system can be described by several models of varying complexity and accuracy. For the BAS case study, we develop such a library of models, of use for advanced control solutions. Given a library of different but related models, we want to reason over different levels of abstraction, while maintaining guarantees. Such guarantees are established based on the computation of $(\varepsilon, \delta)-$approximate (bi)-simulation relations, which are capable of refining assertions on synthesised control policies amongst models. The pair (ε, δ) represent the deviation in the output trajectories between complex and abstract models and the differences in the stochastic transition kernels, respectively. In particular, we aim at automating the generation of

1. (ε, δ)-approximate simulation relations between models of varying orders,
2. control policies with guarantees for time bounded safety specifications,

These aims are tackled by developing a software toolbox which is enhanced by a graphical user interface (GUI). A block diagram describing the structure of the toolbox is presented in Figure 1. It is divided into three main sections. First, a given concrete high-order model is abstracted into a reduced-order model by either making use of the inbuilt library of reduced-order models, or manually inputting user-specific models. Second, the user can perform the computation of the (ϵ, δ)-approximate simulation relations (cf. step 2 in Figure 1), and based on these approximations, the user can synthesise and refine policies for a time-bounded PCTL safety property given in the form of

$$\psi = \mathbb{P}_{\geq p}\left[\Box^{\leq N}|y| \leq a\right], \tag{1}$$

where p is the probability that is to optimised, N is the time horizon, a is the bound within which the output y is in the safe region. We will follow the paradigm of abstraction-based control synthesis [31], wherein we first synthesise the control strategy for a reduced-order model using a modified specification $\psi_{(\varepsilon,\delta)}$. This control strategy is then refined such that it satisfies the original specification ψ. Alternatively, the user can opt to manually input a control strategy for the built abstract model, select a specific (ε, δ) pair and compute a refined policy for the input guarantees (cf. 3b in Figure 1).

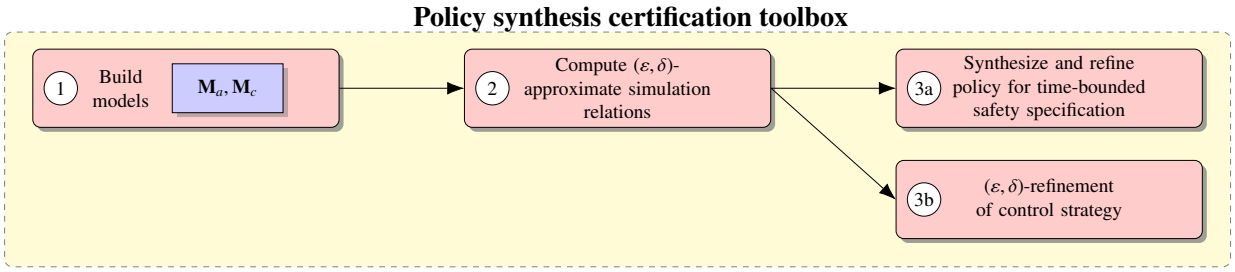


Figure 1: Toolbox block diagram showing the different steps (found within numbered labels) to certify policy synthesis for a pair of concrete and abstract models. The user first builds the concrete ($\mathbf{M}_c$) and abstract ($\mathbf{M}_a$) models in step 1. For the two models the user first computes the (ε, δ)-approximate simulation relation in step 2 and is then given the option to either compute the optimal policy for the time-bounded safety specification ψ in step 3a or to refine a user input control policy in step 3b.

3. A library of thermal models for building automation systems

Buildings have the capability of both storing and transmitting heat. By increasing zones temperatures, energy is stored in the walls, ceilings, floors and in the air inside enclosed spaces. The thermal capacity with which these building components can store energy is a function of their mass and the specific heat capacity of the material making up the building component. Heat can be transmitted through building elements via conduction and convection over surfaces and volumes. Heat can be further gained from radiators, solar radiation through windows, and thermal heat generated by occupants or devices.

By virtue of these thermal properties, mathematical models representing the temperature dynamics of the building zones can be built. For ease of interpretation, these models are often represented by an electrical circuit with corresponding current and voltage dynamics. These equivalent models are called Resistor-Capacitor (RC) circuit networks: resistors represent heat transmission and capacitors represent heat storage [32, 33, 34]. External heat gains such as heat gains due to solar energy are added to the network in the form of equivalent current sources.

The size and complexity of the resulting models depend on the precision with which thermal phenomena are modelled. By neglecting specific thermal phenomena or influences, we can obtain models with different levels of complexity, and regard these models at different levels of abstractions. Whilst abstract and simplified, low-order models are often key. First, they have been developed to isolate the dominant thermal phenomena in the building, and as such, they provide accessible insight to engineers. Second, abstract models are simpler and of a lower order, and thus suitable for the design of control architectures and prone to be computed. On the other hand, complex and

high-dimensional models, taking into account many thermal phenomena, provide a more realistic description of the building dynamics.

In this work, the goal is to show how control architectures designed for the different levels of abstraction can be related to each other. We will consider an actual Building Management System (BMS) set-up in two seminar rooms of the Computer Science Department, at the University of Oxford. Resorting to a few assumptions that are typical in literature for the modelling of thermal dynamics, we first obtain a set of continuous-time models. This collection of models is then translated to discrete-time models, which are expressed as general Markov decision processes. Within the obtained library of models, we quantify their differences by means of new approximate similarity relations, which are introduced in the next section.

3.1. Models for two teaching rooms

We consider two teaching rooms within the Department of Computer Science, University of Oxford. The teaching rooms are connected back to back. Three of the walls of the two rooms are exposed to the outside, while the rest connects to the interior hall. The rooms are used between 8 am and 5 pm during term time. The dimensions of the floor plan areas, for each respective room, are $94m^2$ and $96m^2$. Both rooms have two external windows and contain twenty desktop computers. A layout of the teaching rooms is shown in Figure 2. The two rooms form part of a larger BMS that controls the rest of the building. Indoor heating control is primarily managed with thermostat controlled radiators and fan coil units (FCU), which form part of a variable-air ventilation system. The rooms are further equipped with temperature, humidity, and CO_2 sensors. Measurements consisting of one-minute sampled values are recorded and stored on a daily basis.

(a) Layout of first room

(b) Layout of second room

Figure 2: Layout of teaching rooms for modeling room dynamics, with desks in the front and computer servers at the back.

The thermal dynamics of the two rooms are translated into an equivalent RC circuit, which is shown in Figure 3, and which can be described by a 7^{th}-order ODE model. The thermal dynamics of the two rooms is captured by the average room temperatures T_1, T_2 and which exchange heat with the outside (temperature T_{out}) and with the hall (temperature T_{hall}) through corresponding walls w_j, $j = 1 \dots 7$. The equivalent RC-circuit in Figure 3 is built by applying the following assumptions on the thermal dynamics:

1. The air in the rooms has a uniform temperature over its volume (temperature nodes T_1 and T_2 in Figure 3).
2. The temperature is uniform across each of the modelled walls.
3. The outer north and west walls of room 1 (labelled w_1 and w_2) can be modelled as a single wall (cf. w_2 in Figure 3). To do this lumping, we have assumed a uniform wall temperature and negligible differences in their captive and resistive values, because they are constructed using the same material.
4. Similarly in room 2, the inner walls to the east (w_4) and south (w_5) are lumped into w_4.

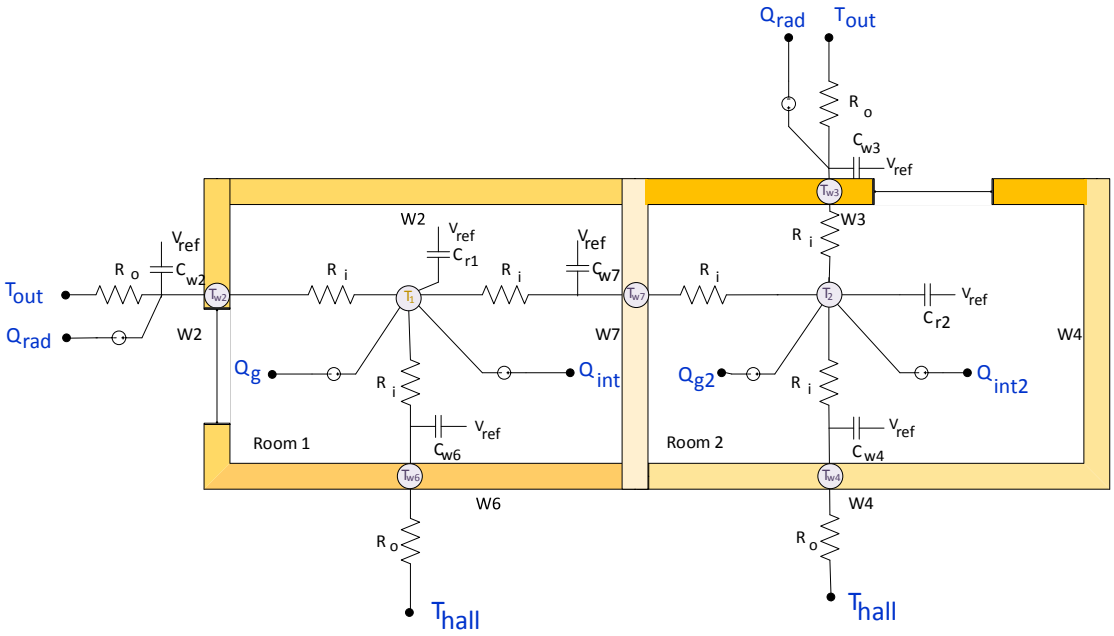


Figure 3: Equivalent RC-circuit for the two teaching rooms in the Department of Computer Science. Each of the modelled walls (w_2, w_3, w_4, w_6, w_7) has a unique hue of yellow. The model has seven states, depicted by nodes, representing both the wall temperatures T_{wj}, $j = 2, 3, 4, 6, 7$ and the zone temperatures $T_i, i = 1, 2$. The zone temperature T_1 of room 1 is highlighted in orange as it is the output of the model. The temperatures T_{out}, T_{hall} and heat sources $Q_{rad}, Q_g, Q_{g2}, Q_{int}, Q_{int2}$ are input disturbances. Here, $Q_g = Q_{heater_1} + Q_{HVAC_1}$ represents the input heat gain in room 1, $Q_{g2} = Q_{heater_2} + Q_{HVAC_2}$ represents the input heat in room 2 and Q_{int2} is the internal heat gained from occupants in room 2. The effect of heat stored in the walls and in the rooms is depicted with the capacitors (⊣⊢) and quantified based on the reference temperature V_{ref}. This includes the wall capacitance C_{wj}, $j = 2, 3, 4, 6, 7$ and the individual room capacitances $C_{ri}, i = 1, 2$. In the circuit, the resistors (⋀⋀⋀) model the resistance R_o to heat transfer from walls to the outside, and the resistance R_i to heat transfer for walls connected internally within zones.

The resulting model is a 7^{th}-order, continuous-time model described by

$$C_{r1}\frac{d(T_1)}{dt} = \frac{T_{w2} - T_1}{R_i} + \frac{T_{w6} - T_1}{R_i} + \frac{T_{w7} - T_1}{R_i} + Q_{HVAC_1} + Q_{int_1} + Q_{heater_1}, \tag{2a}$$

$$C_{r2}\frac{d(T_2)}{dt} = \frac{T_{w4} - T_2}{R_i} + \frac{T_{w3} - T_2}{R_i} + \frac{T_{w7} - T_2}{R_i} + Q_{HVAC_2} + Q_{int_2} + Q_{heater_2}, \tag{2b}$$

$$C_{w2}\frac{d(T_{w2})}{dt} = \frac{T_1 - T_{w_2}}{R_i} + \frac{T_{out} - T_{w_2}}{R_o} + \alpha A_1 Q_{rad}, \tag{2c}$$

$$C_{w3}\frac{d(T_{w3})}{dt} = \frac{T_2 - T_{w_3}}{R_i} + \frac{T_{out} - T_{w_3}}{R_o} + \alpha A_2 Q_{rad}, \tag{2d}$$

$$C_{w4}\frac{d(T_{w4})}{dt} = \frac{T_2 - T_{w_4}}{R_i} + \frac{T_{hall} - T_{w_4}}{R_o}, \tag{2e}$$

$$C_{w6}\frac{d(T_{w6})}{dt} = \frac{T_1 - T_{w_6}}{R_i} + \frac{T_{hall} - T_{w_6}}{R_o}, \tag{2f}$$

$$C_{w7}\frac{d(T_{w7})}{dt} = \frac{T_1 - T_{w_7}}{R_i} + \frac{T_2 - T_{w_7}}{R_i}. \tag{2g}$$

Here C_{wj} is the capacitance of the corresponding j-th wall; C_{ri} represents the capacitance in the i-th room; R_o represents the resistance of walls connected to the outside; R_i represents the resistance offered by the internal walls. Furthermore in (2) Q_{HVAC_i}, Q_{heater_i} are the heat sources for the i-th room that can be controlled, whereas Q_{rad_i}, Q_{int_i} are heat sources that cannot be controlled. Firstly, heat is pumped in by the heating, ventilation and air-conditioning

(HVAC) system, represented by the quantity Q_{HVAC_i}, and quantified as:

$$Q_{HVAC_i} = \dot{m}C_{pa}(T_{fsi} - T_i), \qquad i = 1, 2,$$

where $\dot{m}$ represents the mass airflow, C_{pa} the specific heat capacity and T_{fsi} the HVAC supply temperature. The mass air flow $\dot{m}$ is also fixed. Additionally for each room, Q_{heater_i} represents the total power being output by the radiators in the i-th room, and is modelled as:

$$Q_{heater_i} = k_i \delta T_i P_{out}, \quad \delta T_i = f(T_{rad} - T_i), \qquad \text{i} = 1, 2,$$

where k_i represents the number of radiators in the room i; δT_i is a function f of the radiator mean flow temperature T_{rad} and room temperature T_i, which is obtained from standard radiator conversion look-up tables; and finally P_{out} is the rated output power of the radiator. δT_i will be modelled as a stochastic signal.

Radiative solar energy is absorbed by the walls and is labelled Q_{rad}, for this α is the absorptivity coefficient of solar energy of the windows, and A_i is the total window area for zone i. Occupants or objects generating heat in each room $i = 1, 2$ are modelled using Q_{int_i}. Similarly to [35] we model Q_{rad} and Q_{int_i}, $i = 1, 2$ as

$$Q_{rad}(t) = a_0 T_{out}(t) + a_1 \zeta(t), \tag{3}$$
$$Q_{int_i}(t) = b_{0i} CO_{2_i}(t) + b_{1i} \nu_i(t), \qquad \text{i} = 1, 2. \tag{4}$$

Here the total solar heat gain is a function of T_{out} and an additive signal $\zeta(t)$ is modelled as a random noise signal. Since the internal heat gain is related to (inter alia) the number of people in the room, it can as a consequence be modelled by the CO_2 (which can be monitored) in combination with a random signal $\nu_i(t)$.

Remark 1. *The model is composed of a set of unknown parameters that must be estimated, in order to have a fully identified model that is able to represent the underlying dynamics of the system. There are a number of techniques that can be used to estimate these unknown parameters, which include methods based on maximum likelihood [33, 36] or Kalman filtering [37]. In this work, the algorithm developed in [38] is used to estimate the unknown parameters: this makes use of the extended Kalman filter to predict the state evolution of the model and then applies maximum likelihood to obtain the parameters that best fit the underlying model.*

3.2. Discrete-time models for building automation systems

We construct a set of discrete-time models, with $\triangle T = 5$ *minutes* sampling time having the following general structure:

$$\tilde{\mathbf{M}} : \begin{cases} x(n+1) & = Ax(n) + Bu(n) + Fd(n), \\ y(n) & = Cx(n). \end{cases} \tag{5}$$

Here, x is a column vector representing the states, u represents the control input, d represent the disturbance, y represents the output observations of the model and matrices $A \in \mathbb{R}^{p \times p}$, $B \in \mathbb{R}^{p \times m}$, $F \in \mathbb{R}^{p \times m_d}$, $C \in \mathbb{R}^{1 \times p}$. The models are initialised using such, $x(0) = x_0$ where x_0 is a deterministic vector representing the initial temperatures. The length of the sampling period is assumed to be $\triangle T$ and the Euler-Maruyama method is used to discretise time in the model. For the constructed library of models, the analogous RC circuits are depicted in Figure 4. The continuous-time model given by Equation (2) is discretised to obtain the equivalent 7-th order ($\tilde{\mathbf{M}}_7$) discrete time model

$$\tilde{\mathbf{M}}_7 : \begin{cases} x(n+1) & = Ax(n) + Bu(n) + F_1 T_{out}(n) + F_2 T_{hall}(n) + F_3 CO_2(n) \\ & \qquad + F_4 CO_{2_2}(n) + F_5 \delta T(n) + F_6 \zeta(n) + F_7 \nu_1(n) + F_8 \nu_2(n), \\ y(n) & = Cx(n). \end{cases} \tag{6}$$

$\tilde{\mathbf{M}}_7$ models the indoor zone temperature of room 1 (T_1) and of room 2 (T_2). It includes, $x \in \mathbb{R}^7$ with elements $x = (T_{w2},\ T_{w3},\ T_{w4},\ T_{w6},\ T_{w7},\ T_1,\ T_2)$. For our library of models, we are only interested in the temperature in zone 1, thus we set $C = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 1 & 0 \end{bmatrix}$. The actual values of the matrices A, B, C, F_i are provided in the Appendix. The model has one control input, $u = T_{fs1} \in \mathbb{R} | 0 \leq T_{fs1} \leq 30$, which affects the input supply air that is fed to the two rooms. The model considers eight major sources of disturbances. We quantify and model these disturbances as random effects entering in the room temperature dynamics.

(a) T_{out} is the randomness generated by the outside air temperature. This is modelled as a noise signal generated independently in time from a Gaussian distribution with a mean of 9 $^{\circ}C$ and unit variance. Namely, at each time instant the outside temperature is modelled via independent realisations, as $T_{out}(n) \sim \mathcal{N}(9, 1)$.

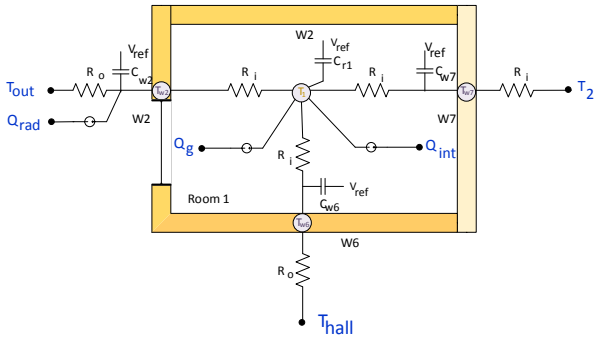
(b) T_{hall} represents the external temperature surrounding the room for all walls not exposed to the outside air temperature (w_6, w_4). This is captured namely using an i.i.d. Gaussian distribution, i.e., a random signal modelled with an identical and independent distribution over time, having a mean of 15 $^{\circ}C$ and unit variance. Namely, at each time instant t it holds the hall temperature is modelled as an independent realisation $T_{hall}(n) \sim \mathcal{N}(15, 1)$.

(c) There is a relation between the internal heat gain in each room caused by people and objects and the CO_2 levels (cf. Equation (4)). Based on the available CO_2 measurements, we choose to model its behaviour using an i.i.d. stochastic signal with a Gaussian distribution given as $CO_2(n) \sim \mathcal{N}(400, 100)$. Notice that these two signals represent the third and fourth stochastic disturbances considered in the model.

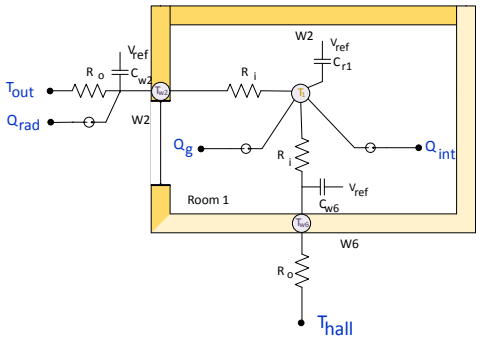
(d) ζ and $\nu_i, i = 1, 2$ are stochastic inputs that come from modelling the heat gain from solar radiation and the internal heat gain in the i-th room respectively. They are all modelled using as zero-mean i.i.d. Gaussian distributions with variance of 0.1, that is, $\zeta(n) \sim \mathcal{N}(0, 0.1)$ and $\nu_{1,2}(n) \sim \mathcal{N}(0, 0.1)$.

(e) δT is a stochastic input that is dependent on the radiator mean flow temperature and is modelled as an i.i.d. random signal where at each time instant $\delta T(n)$ is a realisation of a Gaussian given as $\delta T(n) \sim \mathcal{N}(0.82, 0.1)$.

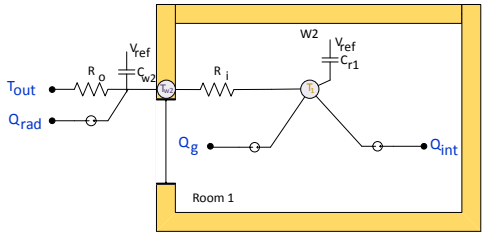
The above disturbances are modelled as independent random variables. Thus, as an example, we model the CO_2 signal as a random signal whose realisations are independent of the realisations of $\zeta(n)$.



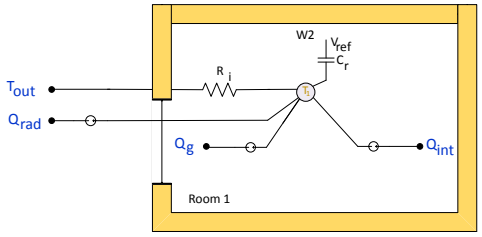
(a) 4^{th} order model ($\tilde{\mathbf{M}}_4$) representing dynamics of room 1. The temperature measurements of room 2, T_2 are now considered as a stochastic disturbance.



(b) 3^{rd} order model ($\tilde{\mathbf{M}}_3$) representing dynamics of room 1 obtained by assuming there is negligible temperature difference between T_2 and T_1.



(c) 2^{nd} order model ($\tilde{\mathbf{M}}_2$) representing dynamics of room 1 obtained by assuming there is negligible temperature difference between T_1, the surrounding walls and T_{hall}.



(d) 1^{st} order model ($\tilde{\mathbf{M}}_1$) representing dynamics of room 1 obtained by assuming steady-state wall dynamics and only considers heat exchange with the outside air. Here, $C_r = C_{r1} + C_{w2}$.

Figure 4: Library of thermal models and corresponding analogous RC circuits. The models states are highlighted using purple nodes, observed output is in orange and the stochastic disturbances are in blue. The states in the models are in turn shown with purple nodes. Here $Q_g = Q_{heater} + Q_{HVAC}$.

The fourth-order model of the temperature in room 1 ($\tilde{\mathbf{M}}_4$), neglects the dynamics of room 2 and replaces its state by a stochastic signal T_2 modelled as an i.i.d. Gaussian distribution having a mean of 20 $^{\circ}C$ and unit variance.

Namely, independent realisations $T_2(n) \sim \mathcal{N}(20, 1)$, as can be seen in Figure 4a. The model comprises variables $x_{s_4} = (T_{w_6}, T_{w_2}, T_{w_7}, T_1)$, which represent the inner and outer wall temperatures T_{w6}, T_{w_2} of the zone, the temperature of the wall which separates the two neighbouring zones T_{w_7} and the indoor zone temperature T_1. Similarly to the original model it is affected by several stochastic sources. The model is represented by

$$\tilde{\mathbf{M}}_4 : \begin{cases} x_{s_4}(n+1) & = A_{s_4}x_{s_4}(n) + B_{s_4}u_{s_4}(n) + F_{1_{s_4}}T_{out}(n) + F_{2_{s_4}}T_{hall}(n) + F_{3_{s_4}}CO_2(n) \\ & \quad + F_{4_{s_4}}T_2(n) + F_{5_{s_4}}\delta T(n) + F_{6_{s_4}}\zeta(n) + F_{7_{s_4}}\nu(n), \\ y(n) & = C_{s_4}x_{s_4}(n). \end{cases} \tag{7}$$

The parametrised matrices $A_{s_4}, B_{s_4}, C_{s_4}, F_{s_4}$ are again detailed in the Appendix.

The third-order model of the temperature in room 1 ($\tilde{\mathbf{M}}_3$), shown in Figure 4b, is constructed by assuming negligible temperature difference between the two neighbouring zones $T_2 = T_1$. The model is represented by

$$\tilde{\mathbf{M}}_3 : \begin{cases} x_{s_3}(n+1) & = A_{s_3}x_{s_3}(n) + B_{s_3}u_{s_3}(n) + F_{1_{s_3}}T_{out}(n) + F_{2_{s_3}}T_{hall}(n) + F_{3_{s_3}}CO_2(n) \\ & \quad + F_{4_{s_3}}\delta T(n) + F_{5_{s_3}}\zeta(n) + F_{6_{s_3}}\nu_1(n), \\ y(n) & = C_{s_3}x_{s_3}(n). \end{cases} \tag{8}$$

$\tilde{\mathbf{M}}_3$ has $x_{s_3} \in \mathbb{R}^3, x_{s_3} = (T_{w_6}, T_{w_2}, T_1)$ and six major sources of stochastic disturbance $(T_{out}, T_{hall}, CO_2, \delta T, \zeta, \nu_1)$. Unlike $\tilde{\mathbf{M}}_4$, the stochastic disturbance coming from T_2 is ignored as it is assumed that $T_2 = T_1$.

The second-order model of the temperature in room 1 ($\tilde{\mathbf{M}}_2$), depicted in Figure 4c, assumes that the inner walls have a similar temperature to the zone temperature and $T_{hall} = T_1 = T_{w_2}$. The model includes variables $x_{s_2} \in \mathbb{R}^2, x_{s_2} = (T_{w_2}, T_1)$ and considers five major sources of stochastic disturbances $(T_{out}, CO_2, \delta T, \zeta, \nu_1)$:

$$\tilde{\mathbf{M}}_2 : \begin{cases} x_{s_2}(n+1) & = A_{s_2}x_{s_2}(n) + B_{s_2}u_{s_2}(n) + F_{1_{s_2}}T_{out}(n) + F_{2_{s_2}}CO_2(n) + F_{3_{s_2}}\delta T(n) + F_{4_{s_2}}\zeta(n) + F_{5_{s_2}}\nu_1(n), \\ y(n) & = C_{s_2}x_{s_2}(n). \end{cases} \tag{9}$$

Lastly, the first-order model representing the temperature in room 1 ($\tilde{\mathbf{M}}_1$) no longer considers dynamics of any of the walls and only takes into account the heat exchange with the outside air. The analogous RC circuit is shown in Figure 4d and it is described by

$$\tilde{\mathbf{M}}_1 : \begin{cases} x_{s_1}(n+1) & = A_{s_1}x_{s_1}(n) + B_{s_1}u_{s_1}(n) + F_{1_{s_1}}T_{out}(n) + F_{2_{s_1}}CO_2(n) + F_{3_{s_1}}\delta T(n) + F_{4_{s_1}}\zeta(n) + F_{5_{s_1}}\nu_1(n), \\ y(n) & = C_{s_1}x_{s_1}(n). \end{cases} \tag{10}$$

The model comprises $x_{s_1} = T_1$ and considers five major sources of stochastic disturbance to the system ($T_{out}, CO_2, \delta T, \zeta$, ν_1), which act as an additional source of heat gain in the zone. The internal heat storage capacity of the room encompasses both the heat stored by the walls and the room itself and, thus, the new capacitance is $C_r = C_{r1} + C_{w2}$ [39].

In all the given model descriptions, the output space of the model represents the temperature T_1 in room 1. We introduce a norm on the output space y. For a given set $\mathbb{X}$ a metric or distance function $\mathbf{d}_{\mathbb{X}}$ is a function $\mathbf{d}_{\mathbb{X}} : \mathbb{X} \times \mathbb{X} \to \mathbb{R}_0^+$. So the output space $y \in \mathbb{Y} \subset \mathbb{R}$ is then endowed with the Euclidean norm $\mathbf{d}_{\mathbb{Y}} = \|\cdot\|$.

Remark 2 (Extensions of the given models)**.** *The library of models has been built based on the two teaching rooms set-up within the Department of Computer Science and follow standard RC circuit modelling techniques. This does not limit the application of the models to exclusively the current scenario. The reduced-order models can be extended to model more zones by duplicating the model and adding or removing disturbance and control signals, depending on the actual configuration of the new zones. More sophisticated weather forecasting and occupancy detection models can be added to the thermal models via Q_{rad} and Q_{int} in a straightforward manner. Further details can be added into the models by introducing new additive RC input nodes. These extensions increase the flexibility of the library of models, with a multitude of applications in different building set-ups.*

3.3. General Markov decision processes and synthesis of control strategies

In this section, we formalise the previously introduced models as general Markov decision processes. We generally consider probability measures $\mathbb{P}$ defined over a Borel measurable space $(\mathbb{X}, \mathcal{B}(\mathbb{X}))$ [40] for which $\mathbb{X}$ is a Polish space, e.g., a Euclidean space. Together with the measurable space, such a probability measure $\mathbb{P}$ defines a probability space, which is denoted as $(\mathbb{X}, \mathcal{B}(\mathbb{X}), \mathbb{P})$ and has realisations $x \sim \mathbb{P}$. Let us further denote the set of all probability measures for a given measurable pair $(\mathbb{X}, \mathcal{B}(\mathbb{X}))$ as $\mathcal{P}(\mathbb{X}, \mathcal{B}(\mathbb{X}))$.

Given a probability space $(\mathbb{X}, \mathcal{B}(\mathbb{X}), \mathbb{P})$, a measurable function $y : \mathbb{X} \to \mathbb{Y}$, which induces a probability measure in $\mathcal{P}(\mathbb{Y}, \mathcal{B}(\mathbb{Y}))$, is referred to as a *random variable*. As such all the disturbances affecting the building automation system in Section 3.2 (cf. items (a) to (e)) are random variables originating from a shared probability space $(\Omega, \mathcal{B}(\Omega), \mathbb{P})$. Let us consider the hall temperature T_{hall} to illustrate this. At each time instant n, the distribution of the temperature $T_{hall}(n) \in \mathbb{R}$ is modelled as a $(\mathbb{R}, \mathcal{B}(\mathbb{R}))$-valued random variable whose probability distribution is given as $\mathcal{N}(15, 1) \in \mathcal{P}(\mathbb{R}, \mathcal{B}(\mathbb{R}))$. The probability that a temperature realisation, $T_{hall}(n) \sim \mathcal{N}(15, 1)$, is between 13 and 17 degrees, i.e., $T_{hall}(n) \in [13, 17] \in \mathcal{B}(\mathbb{R})$ is denoted as $\mathcal{N}([13, 17] \mid 15, 1)$. Furthermore at each time instant, the set of disturbances affecting the state evolutions of the model $\tilde{\mathbf{M}}_7$ in (6) can be modelled as a single random variable with measure space $(\mathbb{R}^7, \mathcal{B}(\mathbb{R}^7))$ and realisations

$$\begin{bmatrix} T_{out}(n) \\ T_{hall}(n) \\ CO_2(n) \\ \delta T(n) \\ \zeta(n) \\ \nu_1(n) \\ \nu_2(n) \end{bmatrix} \sim \mathcal{N}(d_m, \Sigma_d), \text{ with mean } d_m = \begin{bmatrix} 9 \\ 15 \\ 400 \\ 0.82 \\ 0 \\ 0 \\ 0 \end{bmatrix} \text{ and variance } \Sigma_d = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 100 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0.1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0.1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0.1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0.1 \end{bmatrix}. \quad (11)$$

At each time instant these disturbances are modelled as realisations of identical and independent distributions (i.i.d.). This means that the state evolutions of model $\tilde{\mathbf{M}}_7$ affected by the random disturbances in (11) are Markovian and that the stochastic system can be modelled as a Markov decision process [41, 42, 43], which is introduced next.

Definition 1 (Markov decision process (MDP))**.** *The tuple* $\mathbf{M} = (\mathbb{X}, \pi, \mathbb{T}, \mathbb{U})$ *defines a discrete-time MDP over an uncountable state space* $\mathbb{X}$*, and is characterised by* $\mathbb{T}$*, a conditional stochastic kernel that assigns to each point* $x \in \mathbb{X}$ *and control* $u \in \mathbb{U}$ *where* $\mathbb{U}$ *is a continuous input space, a probability measure* $\mathbb{T}(\cdot \mid x, u)$ *over* $(\mathbb{X}, \mathcal{B}(\mathbb{X}))$*. The initial probability distribution is* $\pi : \mathcal{B}(\mathbb{X}) \to [0, 1]$*.*

Consider models of the form $\tilde{\mathbf{M}}_7$ in (6) (or more abstractly (5)). Their state evolution can be described by the conditional stochastic kernel of a Markov decision process $(\mathbb{X}, \pi, \mathcal{N}(\cdot \mid Ax + Bu + Fd_m, F\Sigma_d F^T), \mathbb{U})$ with state x in $\mathbb{X}$, control $u \in \mathbb{U}$ and initialisation $\pi = \delta_{x(0)}$, that is, for realisations $x(0) \sim \delta_{x(0)}$ it holds that $x(0) = x_0$ with probability 1. At every state, the state transition depends on the choice of $u \in \mathbb{U}$. More precisely, given a string of inputs $u(0), u(1), \ldots, u(N)$, over a finite time horizon $\{0, 1, \ldots, N\}$, and an initial condition $x(0)$ (sampled from distribution π), the state at the $(n + 1)$-st time instant, $x(n + 1)$, is obtained as a realisation of the controlled Borel-measurable stochastic kernel $\mathbb{T}(\cdot \mid x(n), u(n))$ – these semantics induce paths (or executions) of the MDP.

Consider next a generalisation of the MDP that includes an output mapping to a metric space, such as the mapping $y(n) = Cx(n)$ in (6) of $\tilde{\mathbf{M}}_7$.

Definition 2 (General Markov decision process (gMDP))**.** $\mathbf{M} = (\mathbb{X}, \pi, \mathbb{T}, \mathbb{U}, h, \mathbb{Y})$ *is a discrete-time gMDP consisting of an MDP combined with output space* $\mathbb{Y}$ *and a measurable output mapping* $h : \mathbb{X} \to \mathbb{Y}$*. A metric* $\mathbf{d}_{\mathbb{Y}}$ *decorates the output space* $\mathbb{Y}$*.*

The semantics of the gMDP are directly inherited from those of the MDP (cf. Def 1). Further, output traces of gMDPs are obtained as mappings of MDP paths, namely $\{y(n)\}_{0:N} = y(0), y(1), \ldots, y(N)$, where $y(n) = h(x(n))$. Denote the class of all gMDP with metric output space $\mathbb{Y}$ as $\mathcal{M}_{\mathbb{Y}}$. Notice that the models introduced in the previous section all have as output the temperature in the first room, and as such they can be represented by gMDPs in a common class $\mathcal{M}_{\mathbb{Y}}$ with a metric output space $\mathbb{Y} = \mathbb{R}$.

In the sequel, we will allow the input u to also be chosen according to a distribution $\mu_u : \mathcal{B}(\mathbb{U}) \to [0, 1]$, i.e., $u \sim \mu_u$. This stochastic control input, referred to as μ_u, induces a stochastic transition kernel denoted as $\mathbb{T}(\cdot|x, \mu_u) =$

$\int_{\mathbb{U}} \mathbb{T}(\cdot|x,u)\mu_u(du) \in \mathcal{P}(\mathbb{X}, \mathcal{B}(\mathbb{X}))$. A policy is a selection of such control inputs chosen to achieve the desired controlled behaviour based on the past history of states and controls. We allow controls to be selected from the state to the control space. When the selected controls are only dependent on the current states, and thus conditionally independent of history (or memoryless), the policy is referred to as Markov.

Definition 3 (Markov policy)**.** *For a gMDP* $\mathbf{M} = (\mathbb{X}, \pi, \mathbb{T}, \mathbb{U}, h, \mathbb{Y})$*, a Markov policy* μ *is a sequence* $\mu = (\mu_1, \mu_2, \mu_3, \ldots)$ *of maps* $\mu_n = \mathbb{X} \to \mathcal{P}(\mathbb{U}, \mathcal{B}(\mathbb{U}))$ *for* $n = 0, 1, 2, \ldots$*, from the state space* $\mathbb{X}$ *to the set of controls.*

The execution $\{x(n), n \in [0, N]\}$, initialised with $x(0) \in \mathbb{X}$ and controlled with Markov policy μ, is a stochastic process defined on the canonical sample space $\Omega = \mathbb{X}^{N+1}$ endowed with its product topology $\mathcal{B}(\Omega)$ and with a probability measure $\mathbb{P}$ defined by the transition kernel $\mathbb{T}$, policy μ, and initial distribution π [41, Prop. 7.45]. One can intuitively realise that the optimal policy leading to the maximal probability might not be Markov (memoryless), as introduced in Def. 3. We introduce the notion of a *control strategy*, and define it as a broader, memory-dependent version of the Markov policy above. This strategy is formulated as a Markov process that takes as an input the state of the given gMDP and is time homogeneous.

Definition 4 (Control strategy)**.** *A control strategy* $\mathbf{C} = (\mathbb{X}_\mathbf{C}, x_{\mathbf{C}0}, \mathbb{X}, \mathbb{T}^n_\mathbf{C}, h^n_\mathbf{C})$ *for a gMDP* $\mathbf{M}$ *with state space* $\mathbb{X}$ *and control space* $\mathbb{U}$ *over the time horizon* $n = 0, 1, 2, \ldots, N$ *is an* inhomogenous Markov process *with state space* $\mathbb{X}_\mathbf{C}$*; an initial state* $x_{\mathbf{C}0}$*; inputs* $x \in \mathbb{X}$*; time-dependent, kernels* $\mathbb{T}^n_\mathbf{C}$*,* $n = 0, 1, \ldots, N$*; and with output maps* $h^n_\mathbf{C} : \mathbb{X}_\mathbf{C} \to \mathcal{P}(\mathbb{U}, \mathcal{B}(\mathbb{U}))$*,* $n = 1, \ldots, N$*, with elements* $\mu \in \mathcal{P}(\mathbb{U}, \mathcal{B}(\mathbb{U}))$*.* □

Unlike a Markov policy, the defined control strategy $\mathbf{C}$ is in general history dependent, as it has an internal state that can be used to remember relevant past events.

The composition of a given control strategy $\mathbf{C}$ with a model $\mathbf{M}$ yields a controlled model $\mathbf{C} \times \mathbf{M}$, which is a Markov process whose transitions are explained in Algorithm 1. Observe that in this execution algorithm the first control $u(0)$ is selected by drawing $x_\mathbf{C}(1)$ according to $\mathbb{T}^0_\mathbf{C}(\cdot\,|x_\mathbf{C}(0), x(0))$, where $x_\mathbf{C}(0) = x_{\mathbf{C}0}$ is the initial state, and selecting $u(0)$ from measure $\mu^0_\mathbf{C} = h^0_\mathbf{C}(x_\mathbf{C}(1))$. Also note that the stochastic transitions for the control strategy and the gMDP are selected in an alternating fashion. The output map of the strategy is indexed based on the time instant at which the resulting policy will be applied to the gMDP. The control strategy $\mathbf{C}$ applied to $\mathbf{M}$ can be both stochastic (as a realisation of $\mathbb{T}^0_\mathbf{C}(\cdot\,|\,x_\mathbf{C}(0), x(0))$), a function of the initial state $x(0)$, and time dependent. As such it is capable of representing both classical control strategies and more complex strategies. In the following, we provide a few control examples. The execution $\{(x(n), x_\mathbf{C}(n)), n \in [0, N]\}$ of a gMDP $\mathbf{M}$ controlled with strategy $\mathbf{C}$ is defined on the canonical sample space $\Omega = (\mathbb{X} \times \mathbb{X}_\mathbf{C})^{N+1}$ endowed with its product topology $\mathcal{B}(\Omega)$, and with the probability measure $\mathbb{P}_{\mathbf{C}\circledast\mathbf{M}}$.

Algorithm 1 Execution of the controlled model $\mathbf{C} \times \mathbf{M}$

```
set n = 0 and x_C(0) = x_C0
draw x(0) ~ π                                              {from M}
while n < N do
    draw x_C(n+1) ~ T^n_C(· | x_C(n), x(n))                {from C}
    set μ_n = h^n_C(x_C(n+1)), draw u(n) from μ_n
    draw x(n+1) ~ T(· | x(n), u(n))                        {from M}
    set n = n + 1
end while
```

Example 3.

3.a. *For the first-order model* $\tilde{\mathbf{M}}_1$ *consider an open-loop control strategy*[1] *we take as control input a constant supply temperature of* 22 °*C. Thus* $u_{s_1}(n) = 22, \forall n \in [0, \infty)$*. Then this input can be trivially written as a control strategy* $\mathbf{C}_{22°C} = (\{q_0\}, q_0, \mathbb{X}_{s_1}, \mathbb{T}^n_{22°C}, h^n_{22°C})$ *with*

[1] We refer to a strategy as being open-loop if it does not depend on the value of state or output of model M.

- $\{q_0\}$ *the set of possible control states.*
- $\mathbb{X}_{s_1} = \mathbb{R}$ *is the embedding of the state space of* $\tilde{\mathbf{M}}_1$.
- $\mathbb{T}^n_{22^\circ C}$ *is the stochastic kernel defining the distribution of transitions over the set of control states* $x_{\mathbf{C}} \in \{q_0\}$. *Since there are only transitions within the singleton set* $\mathbb{X}_{\mathbf{C}_{22^\circ C}} = \{q_0\}$, *the conditional stochastic kernel is given as* $\forall x_{\mathbf{C}} \in \{q_0\}$, $\forall x_{s_1} \in \mathbb{X}_{s_1}$, $\forall n \in [0, \infty)$

$$\mathbb{T}^n_{22^\circ C}(q_1 | x_{\mathbf{C}}, x_{s_1}) = 1.$$

- $h^n_{22^\circ C} : \{q_0\} \to \mathcal{P}(\mathbb{U}, \mathcal{B}(\mathbb{U}))$ *is the output mapping defined as*

$$h^n_{22\,^\circ C}(x_{\mathbf{C}}) = \delta_{22} \in \mathcal{P}(\mathbb{U}, \mathcal{B}(\mathbb{U})), \quad \forall n.$$

Here δ_{22} *is a Dirac distribution over* $\mathbb{U} = \mathbb{R}$, *such that with probability 1 it holds that* $u_{s_1}(n) = 22$.

3.b. *As an alternative, we now consider a state-feedback control strategy. In this case the control strategy again selects the input supply temperature in the rooms* (T_{fs1}), *but now it takes into consideration the difference between the desired room temperature of* $21\ ^\circ C$ *and the actual temperature at each time step. Thus we take* $u_{s_1} = k(21 - x_{s_1})$ *with an appropriate gain* $k > 0$ *such that* $|A_{s_1} - B_{s_1} k| < 1$. *This is equal to the feedback control strategy* $\mathbf{C}_k = \{\mathbb{R}, 0, \mathbb{X}_{s_1}, \mathbb{T}^n_k, h^n_k\}$

- $\mathbb{R}$ *the set of possible control states is equal to the state space of* $\tilde{\mathbf{M}}_1$ *and is trivially initiated at* $\{q_0\}$.
- $\mathbb{X}_{s_1} = \mathbb{R}$ *is again the embedding of the state space of* $\tilde{\mathbf{M}}_1$,
- $\mathbb{T}^n_k$ *is the stochastic kernel and is chosen as the current error in the temperature, that is* $21\ ^\circ C - x_{s_1}$, *and this is formalised as as* $\forall x_{\mathbf{C}} \in \{q_0\}$, $\forall x_{s_1} \in \mathbb{X}_{s_1}$, $\forall n \in [0, \infty)$

$$\mathbb{T}^n_k(\cdot | x_{\mathbf{C}}, x_{s_1}) = \delta_{21 - x_{s_1}}(\cdot).$$

- $h^n_k : \{0\} \to \mathcal{P}(\mathbb{U}, \mathcal{B}(\mathbb{U}))$ *is the output mapping defined as* $h^n_k(x_{\mathbf{C}}) = \delta_{k x_{\mathbf{C}}} \in \mathcal{P}(\mathbb{U}, \mathcal{B}(\mathbb{U}))$, $\quad \forall n$.

Both control strategies can be employed to regulate the room temperature. In this specific case they are also Markov policies.

4. Approximate simulation relations for gMDPs

In this section, we relate pairs of gMDPs based on their output behaviours with as objective the hierarchical synthesis of control strategies. The gMDPs are considered to be related if a control strategy synthesised for one can be refined to the other, in such a way that the difference in the output behaviour is bounded. More precisely, we define approximate probabilistic simulation and bisimulation relations. These quantify differences in both probability and deviations in the outputs of two controlled gMDPs. These relations, first introduced in [30], characterise the similarity in the controllable behaviours of the two gMDPs.

4.1. Lifting of probability distributions

For sets A and B a relation $\mathcal{R} \subset A \times B$ is a subset of their Cartesian product that relates elements $x \in A$ with elements $y \in B$, denoted as $x\mathcal{R}y$. We use the following notation for the mappings $\mathcal{R}(\tilde{A}) = \{y : x\mathcal{R}y,\ x \in \tilde{A}\}$ and $\mathcal{R}^{-1}(\tilde{B}) = \{x : x\mathcal{R}y,\ y \in \tilde{B}\}$ for $\tilde{A} \subseteq A$ and $\tilde{B} \subseteq B$.

As will be detailed in the next section, $\mathcal{R}$ will be used over state spaces of gMDPs, and it should be selected so that we can relate their output behaviour. Furthermore, $\mathcal{R}$ should be such that the transitions are similar in a probabilistic sense. For *finite- or countable-state* stochastic processes this concept has been introduced in [44, 45, 46] and referred to as *lifting*: the transition probabilities are coupled using a weight function in a way that respects a given *relation* over the combined state spaces. A simple example is portrayed in Fig. 5. Consider the two models $\mathbf{M}_1$ and $\mathbf{M}_2$ depicted on the left and in the middle, then the pairs of states (q_i, x_j) with the same colouring constitute to a relation $\mathcal{R}$. For $\mathbf{M}_1$ and $\mathbf{M}_2$ the transition probabilities are noted on the transition edges. Initiated at q_1 the stochastic transition kernel

of $\mathbf{M}_1$ assigns the probability measure over the discrete state space $S = \{q_1, q_2, q_3, q_4\}$ as $\{0, 1/2, 1/3, 1/6\}$. Similarity for $\mathbf{M}_2$, at x_1 a probability distribution $\{0, 5/6, 1/6\}$ over $\{x_1, x_2, x_4\}$ is given. For these probability distributions, the rightmost figure depicts a lifting with respect to the given relation for the initial state pair (q_1, x_1). Observe that this lifting assigns probabilities to pairs of next states. This lifting is a specific type of probability coupling in which not only transition kernels for each of the models are maintained, but also the pairs of next states are elements of the relation $\mathcal{R}$.

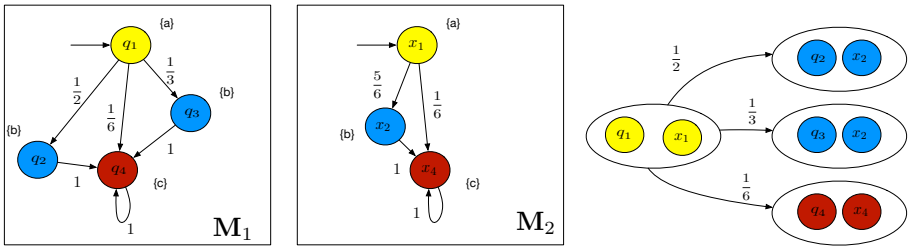


Figure 5: Finite-state Markov processes $\mathbf{M}_1$ and $\mathbf{M}_2$ (left & middle) with $S = \{q_1, q_2, q_3, q_4\}$ and $T = \{x_1, x_2, x_4\}$ the respective state spaces. The states are labelled with three different colours. Lifting probabilities of the transition kernels for (q_1, x_1) are given on the edges of the rightmost figure.

Since we assume that the state spaces are Polish and have a corresponding Borel σ-field for the given probability spaces $(\mathbb{X}_1, \mathcal{B}(\mathbb{X}_1), \mathbb{P}_1)$ and $(\mathbb{X}_2, \mathcal{B}(\mathbb{X}_2), \mathbb{P}_2)$ with $\mathbb{P}_1 = \mathbb{T}_1(\cdot \mid x_1, u_1)$ and $\mathbb{P}_2 = \mathbb{T}_2(\cdot \mid x_2, u_2)$, the question of finding a coupling (cf. [47, 48]) can be reduced to finding a probability measure in $\mathcal{P}(\mathbb{X}_1 \times \mathbb{X}_2, \mathcal{B}(\mathbb{X}_1 \times \mathbb{X}_2))$. This is depicted in Figure 6, where on the left there is a measurable space with an unknown probability measure. For this coupling problems the unknown measure should be specified, such that events mapped to the spaces $\mathbb{X}_1$ and $\mathbb{X}_2$ induce probability measures $\mathbb{P}_1$ and $\mathbb{P}_2$, respectively. Of interest is the case of lifting where this coupling is restricted to a given mapping $\mathcal{R}$, which relates states that are similar.

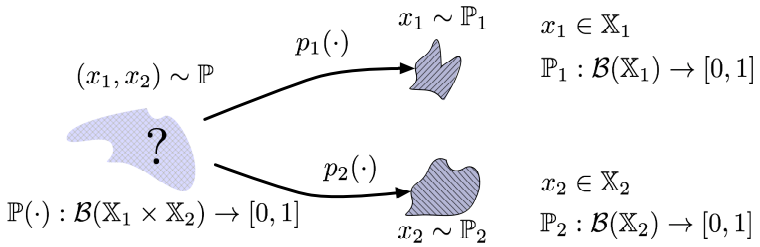


Figure 6: Coupling of probability distributions.

Definition 5 (δ-lifting for general state spaces)**.** *Let $\mathbb{X}_1, \mathbb{X}_2$ be two sets with associated probability spaces $(\mathbb{X}_1, \mathcal{B}(\mathbb{X}_1), \Delta)$, $(\mathbb{X}_2, \mathcal{B}(\mathbb{X}_2), \Theta)$, and let $\mathcal{R} \subseteq \mathbb{X}_1 \times \mathbb{X}_2$ be a relation for which $\mathcal{R} \in \mathcal{B}(\mathbb{X}_1 \times \mathbb{X}_2)$. We denote by $\bar{\mathcal{R}}_\delta \subseteq \mathcal{P}(\mathbb{X}_1, \mathcal{B}(\mathbb{X}_1)) \times \mathcal{P}(\mathbb{X}_2, \mathcal{B}(\mathbb{X}_2))$ [2] the corresponding lifted relation (acting on $\Delta \bar{\mathcal{R}}_\delta \Theta$), if there exists a probability space $(\mathbb{X}_1 \times \mathbb{X}_2, \mathcal{B}(\mathbb{X}_1 \times \mathbb{X}_2), \mathbb{W})$ satisfying*

1. *for all $X_1 \in \mathcal{B}(\mathbb{X}_1)$: $\mathbb{W}(X_1 \times \mathbb{X}_2) = \Delta(X_1)$;*
2. *for all $X_2 \in \mathcal{B}(\mathbb{X}_2)$: $\mathbb{W}(\mathbb{X}_1 \times X_2) = \Theta(X_2)$;*
3. *for the probability space $(\mathbb{X}_1 \times \mathbb{X}_2, \mathcal{B}(\mathbb{X}_1 \times \mathbb{X}_2), \mathbb{W})$ it holds that $x_1 \mathcal{R} x_2$ with probability at least $1 - \delta$, or equivalently that $\mathbb{W}(\mathcal{R}) \geq 1 - \delta$.*

For $\delta = 0$ the exact lifting with respect to $\mathcal{R}$, as introduced in [18], is recovered.

[2] Details on how to compute the Cartesian product of $\mathcal{P}(\mathbb{X}_1, \mathcal{B}(\mathbb{X}_1)) \times \mathcal{P}(\mathbb{X}_2, \mathcal{B}(\mathbb{X}_2))$ can be found in [30] .

4.2. Exact and (ε, δ)- approximate simulation relations for gMDPs

We work with pairs of gMDP put in a relationship, with the intention to apply the developed notions to an abstraction $\mathbf{M}_a$ of a concrete model $\mathbf{M}_c$. Consider two gMDP $\mathbf{M}_a, \mathbf{M}_c \in \mathcal{M}_{\mathbb{Y}}$ mapping to a common output space $\mathbb{Y}$ with metric $\mathbf{d}_{\mathbb{Y}}$. For $\mathbf{M}_a = (\mathbb{X}_a, \pi_a, \mathbb{T}_a, \mathbb{U}_a, h_a, \mathbb{Y})$ and $\mathbf{M}_c = (\mathbb{X}_c, \pi_c, \mathbb{T}_c, \mathbb{U}_c, h_c, \mathbb{Y})$ at given state-action pairs $x_a \in \mathbb{X}_a, u_a \in \mathbb{U}_a$ and $x_c \in \mathbb{X}_c, u_c \in \mathbb{U}_c$, respectively. We will now use the introduced notions to relate the corresponding transition kernels, namely the probability measures $\mathbb{T}_a(\cdot \mid x_a, u_a) \in \mathcal{P}(\mathbb{X}_a, \mathcal{B}(\mathbb{X}_a))$ and $\mathbb{T}_c(\cdot \mid x_c, u_c) \in \mathcal{P}(\mathbb{X}_c, \mathcal{B}(\mathbb{X}_c))$.

Similar to the alternating notions for probabilistic game structures in [49], we provide a simulation that relates any input chosen for the first process with one for the second process. For this, we introduce the notion of *interface function*:

$$\mathcal{U}_v : \mathbb{U}_a \times \mathbb{X}_a \times \mathbb{X}_c \to \mathcal{P}(\mathbb{U}_c, \mathcal{B}(\mathbb{U}_c)), \tag{12}$$

where we require that $\mathcal{U}_v$ is a Borel measurable function. This means that $\mathcal{U}_v$ induces a Borel measurable stochastic kernel, again denoted by $\mathcal{U}_v$, over $\mathbb{U}_c$ given $(u_a, x_a, x_c) \in \mathbb{U}_a \times \mathbb{X}_a \times \mathbb{X}_c$. The notion of interface function is known in the context of correct-by-design controller synthesis and of hierarchical controller refinement [31, 50]. For the objective of hierarchical controller refinement, an interface function implements (or refines) any control action synthesised over the abstract model to an action for the concrete model. Note that we extend standard interface functions for deterministic systems by allowing randomised actions $\mu_c \in \mathcal{P}(\mathbb{U}_c, \mathcal{B}(\mathbb{U}_c))$. In most cases a deterministic, (that is, non-randomised), control action will suffice. These deterministic interfaces, generally denoted as u_v, can be written as stochastic kernels using a Dirac measure as

$$\mathcal{U}_v = \delta_{u_v} \in \mathcal{P}(\mathbb{U}_c, \mathcal{B}(\mathbb{U}_c)) \text{ with } u_v : \mathbb{U}_a \times \mathbb{X}_a \times \mathbb{X}_c \to \mathbb{U}_c. \tag{13}$$

The notion of approximate probabilistic simulation can be defined based on the lifting of the transition kernels (cf Definition 5) for this interface, thereby generating a stochastic kernel $\mathbb{W}_{\mathbb{T}}$ conditional on the values of signals in $\mathbb{U}_a$ and in $\mathbb{X}_a \times \mathbb{X}_c$. The relation defined next encompasses two approximation requirements: allowing for both a deviation in probability (δ) and a deviation in output accuracy (ε), and is referred to as an (ε, δ)-approximate probabilistic simulation.

Definition 6 ((ε, δ)-approximate probabilistic simulation)**.** *Consider two gMDP $\mathbf{M}_i = (\mathbb{X}_i, \pi_i, \mathbb{T}_i, \mathbb{U}_i, h_i, \mathbb{Y}), i = a, c$, over a shared metric output space $(\mathbb{Y}, \mathbf{d}_{\mathbb{Y}})$. $\mathbf{M}_a$ is (ε, δ)-probabilistically simulated by $\mathbf{M}_c$ if there exists an interface function $\mathcal{U}_v$ and a relation $\mathcal{R} \subseteq \mathbb{X}_a \times \mathbb{X}_c$, for which there exists a Borel measurable stochastic kernel $\mathbb{W}_{\mathbb{T}}(\cdot \,|u_a, x_a, x_c)$ on $\mathbb{X}_a \times \mathbb{X}_c$ given $\mathbb{U}_a \times \mathbb{X}_a \times \mathbb{X}_c$, such that:*

1. $\forall (x_a, x_c) \in \mathcal{R}$, $\mathbf{d}_{\mathbb{Y}}\left(h_a(x_a), h_c(x_c)\right) \leq \varepsilon$;
2. $\forall (x_a, x_c) \in \mathcal{R}$, $\forall u_a \in \mathbb{U}_a$: $\mathbb{T}_a(\cdot|x_a, u_a)\, \bar{\mathcal{R}}_\delta\, \mathbb{T}_c(\cdot|x_c, \mathcal{U}_v(u_a, x_a, x_c))$, *with lifted probability measure* $\mathbb{W}_{\mathbb{T}}(\cdot\,|u_a, x_a, x_c)$;
3. $\pi_a \bar{\mathcal{R}}_\delta \pi_c$.

The simulation relation is denoted as $\mathbf{M}_a \preceq_\varepsilon^\delta \mathbf{M}_c$.

That is, $\mathbf{M}_a$ is in (ε, δ)-approximate probabilistic simulation relation with $\mathbf{M}_c$, if there exists an interface function $\mathcal{U}_v$ and a relation $\mathcal{R}$ such that for all state pairs (x_a, x_c) within the relation $\mathcal{R}$, the difference in the output between each of the models is less than or equal to ε and for all control inputs given to $\mathbf{M}_a$, the transition probabilities of the two models are related according the lifting probability measure $\mathbb{W}_{\mathbb{T}}(\cdot\,|u_a, x_a, x_c)$. Also the initial probability distribution of the two models are in the lifted relation $\bar{R}$.

If $\delta = 0$ and $\varepsilon = 0$ we refer to the probabilistic simulation relation as *exact*, and for this probabilistic simulation relation we drop the indices, i.e, $\mathbf{M}_a \preceq \mathbf{M}_c$. If there exists a relation $\mathcal{R}$ such that $\mathbf{M}_a \preceq \mathbf{M}_c$ (respectively, $\mathbf{M}_a \preceq_\varepsilon^\delta \mathbf{M}_c$) and such that for $\mathcal{R}^{-1}$ also $\mathbf{M}_c \preceq \mathbf{M}_a$ (respectively, $\mathbf{M}_c \preceq_\varepsilon^\delta \mathbf{M}_a$), then we say that $\mathbf{M}_a \approx \mathbf{M}_c$ (respectively, $\mathbf{M}_a \approx_\varepsilon^\delta \mathbf{M}_c$). The latter is referred to as a ((ε, δ)-approximate) probabilistic bisimulation relation.

For every gMDP $\mathbf{M}$: $\mathbf{M} \preceq \mathbf{M}$ and $\mathbf{M} \approx \mathbf{M}$. This can be seen by considering the diagonal relation $\mathcal{R}_{diag} = \{(x_1, x_2) \in \mathbb{X} \times \mathbb{X} \mid x_1 = x_2\}$ and selecting equal inputs for the associated interfaces. The resulting equal transition kernels $\mathbb{T}(\cdot|x, u)\bar{\mathcal{R}}_{diag}\mathbb{T}(\cdot|x, u)$ are lifted by the measure $\mathbb{W}_{\mathbb{T}}(dx_1' \times dx_2'|u, x_1, x_2) = \delta_{x_1'}(dx_2')\mathbb{T}(dx_1'|x_1, u)$ where $\delta_{x_1'}$ denotes the Dirac distribution located at x_1'. Let us use this diagonal relation to give some additional examples.

Example 4. **a**. *We show the notion of lifting for diagonal relations. Consider two models as in* (5)*, that is, they can be seen as Gaussian processes written as*

$$\begin{aligned} \mathbf{M}_1 &: x_1(n+1) = A_1 x_1(n) + Bu_1(n) + Fd(n), & y_1(n) &= Cx_1(n), \\ \mathbf{M}_2 &: x_2(n+1) = A_2 x_1(n) + Bu_2(n) + Fd(n), & y_2(n) &= Cx_2(n), \end{aligned}$$

with $A_1 = A_2 + BK$ and variables $x(n), x(n+1), d(n)$ taking values in $\mathbb{R}^p$ and $u(n) \in \mathbb{R}^m$, matrices $A_{1,2} \in \mathbb{R}^{p\times p}$, $B \in \mathbb{R}^{p\times m}$, $K \in \mathbb{R}^{m\times p}$, $F \in \mathbb{R}^{p\times p}$. Both systems are disturbed by the signal d, which is a white noise signal with variance Σ_d. For any pair of states $(x_1, x_2) \in \mathcal{R}_{diag}$, that is $x_1 = x_2 = x$, it holds that for the corresponding transition kernels, given as $\mathcal{N}(\cdot\,|A_1 x + Bu_1, F\Sigma_d F^T)$ and $\mathcal{N}(\cdot\,|A_2 x + Bu_2, F\Sigma_d F^T)$, and taking $u_2 = u_v(u_1, x_1, x_2) = Kx_2 + u_1$, we can lift the conditional probability kernels as

$$\mathcal{N}(\cdot\,|A_1 x + Bu_1, F\Sigma_d F^T)\bar{\mathcal{R}}_{diag}\mathcal{N}(\cdot\,|A_1 x + Bu_1, F\Sigma_d F^T).$$

Hence, if the systems are both initialised with the same π, then we have shown that $\mathbf{M}_1 \preceq \mathbf{M}_2$. This follows trivially because the models have the same output mapping and satisfy the lifting criteria.
b. *Consider the gMDP $\mathbf{M}_1$ and $\mathbf{M}_2$, given in a slightly more general form than* (5)*, as*

$$\begin{aligned} \mathbf{M}_1 &: x(n+1) = f(x(n), u(n)) + d(n), & y(n) &= h(x(n)), \\ \mathbf{M}_2 &: x(n+1) = f(x(n), u(n)) + \tilde{d}(n) + \tilde{u}(n), & y(n) &= h(x(n)), \end{aligned}$$

with variables $x(n), x(n+1), u(n), \tilde{u}(n), d(n), \tilde{d}(n)$ taking values in $\mathbb{R}^p$, and with dynamics initialised with the same probability distribution at $t = 0$ and driven by white noise sequences $d(n), \tilde{d}(n)$, both with zero mean normal distributions and with variance $\Sigma_d, \Sigma_{\tilde{d}}$, respectively. Notice that if $\Sigma_d - \Sigma_{\tilde{d}}$ is positive definite then $\mathbf{M}_1 \preceq \mathbf{M}_2$. To see this, select the control input pair $(u_2, \tilde{u}_2) \in \mathbb{U}_2$ as $u_2 = u_1$, and $\tilde{u}_2$ according to the zero-mean normal distribution with variance $\Sigma_d - \Sigma_{\tilde{d}}$, then the associated interface *is $\mathcal{U}_v(du_2 \times d\tilde{u}_2|u_1, x_1, x_2) = \delta_{u_1}(du_2)\mathcal{N}(d\tilde{u}_2|0, \Sigma_d - \Sigma_{\tilde{d}})$. For this interface the stochastic dynamics of the two processes are equal, and can be lifted with $\mathcal{R}_{diag}$. Note that unlike the previous example, in this case the models are clearly ordered, that is, it does not hold that $\mathbf{M}_2$ is probabilistically simulated by $\mathbf{M}_1$.*

In this section, we have provided similarity relations quantifying the difference between two Markov processes. The end use of the introduced similarity relations is to quantify the probability of events of a gMDP via its abstraction and to refine controllers: this is achieved in the next section. In the sequel of this section, we will first give computation details for relations over models of the form (5) for the rooms models.

4.3. Computation of approximate simulation relations for models of thermal dynamics

Let us consider a concrete model $\mathbf{M}_c$ and abstract model $\mathbf{M}_a$ in the same form as (5), that is,

$$\mathbf{M}_c : \begin{cases} x(n+1) &= Ax(n) + Bu(n) + Fd(n), \\ y(n) &= Cx(n), \end{cases} \quad \text{and } \mathbf{M}_a : \begin{cases} x_s(n+1) &= A_s x_s(n) + B_s u_s(n) + F_s d(n), \\ y_s(n) &= C_s x_s(n). \end{cases} \tag{14}$$

In contrast to (5), we will assume that d is a m_d-dimensional white noise sequence with Gaussian distribution and unit variance. This is not a restrictive assumption as any of the models in (5) can be rewritten as (14). Furthermore, we have assumed that the noise term d is shared between the models. By making this choice we will be able to naturally define our lifted transition kernel based on d. For the models of Section 3, we know the origin of the noise and this sharing is hence a sensible choice. Notice that the models under study define gMDPs over the same output space endowed with the Euclidean norm $\mathbf{d}_{\mathbb{Y}}(y_s, y) = \|y - y_s\|$, which we will study under assumption of bounded inputs.

We are now ready to investigate under which conditions for given (ε, δ)-values it holds that $\mathbf{M}_a \preceq^{\delta}_{\varepsilon} \mathbf{M}_c$. Let us first introduce the set of symmetric matrices of dimension m, denoted as $\mathbb{S}^m$. We say that $M \in \mathbb{S}^m$ is positive definite, i.e., $M \succ 0$ if for all $x \neq 0 \in \mathbb{R}^m$ it holds that $x^T Mx > 0$. Similarly, any $M \in \mathbb{S}^m$ is positive semi-definite ($M \succeq 0$) if it holds that $x^T Mx \geq 0$ for all $x \in \mathbb{R}^m$. Moreover for $M_1, M_2 \in \mathbb{S}^m$, $M_1 \succ M_2$ if $M_1 - M_2 \succ 0$ and $M_1 \succeq M_2$ if $M_1 - M_2 \succeq 0$.

To investigate $\mathbf{M}_a \preceq_\varepsilon^\delta \mathbf{M}_c$, we will consider relations $\mathcal{R}$ of the form

$$\mathcal{R} := \left\{(x, x_s) \mid (x - Px_s)^T M(x - Px_s) \leq \varepsilon^2\right\} \quad \text{with } \mathbb{S}^m \ni M \geq 0, \tag{15}$$

where P is a properly-sized matrix. This allows us to rewrite the conditions for (ε, δ)-approximate probabilistic simulation (as in Definition 6) on the respective gMDPs into a set of conditions on the system matrices, as detailed next.

Theorem 7. *Consider the Gaussian models of* (14) *at gMDPs with*

$$\begin{aligned}
&\mathbf{M}_c = (\mathbb{X}, \pi, \mathbb{T}, \mathbb{U}, h, \mathbb{Y}) && \mathbf{M}_a = (\mathbb{X}_s, \pi_s, \mathbb{T}_s, \mathbb{U}_s, h_s, \mathbb{Y})\\
&\quad \mathbb{T}(\cdot \,|x, u) = \mathcal{N}(\cdot \,|Ax + Bu, FF^T), && \quad \mathbb{T}_s(\cdot \,|x, u) = \mathcal{N}(\cdot \,|A_s x_s + B_s u_s, F_s F_s^T),\\
&\quad x \in \mathbb{X} = \mathbb{R}^m,\ \pi = \delta_0 \in \mathcal{P}(\mathbb{R}^m, \mathcal{B}(\mathbb{R}^m)), && \quad x_s \in \mathbb{X}_s = \mathbb{R}^{m_s},\ \pi_s = \delta_0 \in \mathcal{P}(\mathbb{R}^{m_s}, \mathcal{B}(\mathbb{R}^{m_s})),\\
&\quad \textit{and } \mathbb{U} = \{u \in \mathbb{R}\}; && \quad \textit{and } \mathbb{U}_s = \{u_s \in \mathbb{R} \,\big|\, |u_s| \leq c_u\},
\end{aligned}$$

for which we assume that the abstract model M_a is of lower order than the concrete model M_c, that is $m \geq m_s$. To obtain a bounded output error ε, we have required the input of the abstract model to be bounded by c_u. Consider the relation

$$\mathcal{R} := \left\{(x_s, x) \mid (x - Px_s)^T M(x - Px_s) \leq \varepsilon^2\right\}, \tag{16}$$

with $M \geq 0$, and the interface $u_v : \mathbb{U}_s \times \mathbb{X}_s \times \mathbb{X} \to \mathbb{U}$, where

$$u = u_v(u_s, x_s, x) = Ru_s + Qx_s + K(x - Px_s). \tag{17}$$

Then $\mathbf{M}_a \preceq_\varepsilon^\delta \mathbf{M}_c$ with respect to the relation (16) *and the interface* (17) *if the matrices $M \in \mathbb{S}^m$, $P \in \mathbb{R}^{m \times m_s}$, $Q \in \mathbb{R}^{1 \times m_s}$ and $K \in \mathbb{R}^{1 \times m}$ satisfy the following conditions*

$$PA_s = AP + BQ \text{ and } C_s = CP \tag{18}$$

$$M - C^T C \geq 0, \tag{19}$$

together with a condition on the state transitions given as

$$(x_s', x') \in \mathcal{R}, \quad \forall d^T d \leq c_d,\ u_s \in \mathbb{U}_s,\ \forall (x_s, x) \in \mathcal{R} \tag{20}$$

with transitions $x' = Ax + Bu_v(u_s, x_s, x) + Fd$, $x_s' = A_s x_s + B_s u_s + F_s d$, with the noise bound c_d based on the inverse Chi-square $c_d := \chi_2^{-1}(1 - \delta, m_d)$.

The first equality in (18) is the Sylvester equality. The given formulation is very similar to the alternating refinement problems and the hierarchical control work of [31]. The main difference is that in this case the bound on the disturbance input is directly linked to the probability of staying in the relation $\mathcal{R}$. The proposed form of the equations is such that much of the computations can be performed with tools tailored to convex optimisation of linear quadratic control problems [51].

Proof. Furthermore, the relation $\mathbf{M}_a \preceq_\varepsilon^\delta \mathbf{M}_c$ must satisfy three conditions:

1. $\forall (x, x_s) \in \mathcal{R} : d_{\mathbb{Y}}(y(n), y_s(n)) \leq \varepsilon$ must be satisfied. This is indeed the case as $\|y - y_s\|^2 = \|Cx - CPx_s\|^2$ and $(x - Px_s)^T C^T C(x - Px_s) \leq (x - Px_s)^T M(x - Px_s)$, and the latter is bounded by ε^2 for $(x, x_s) \in \mathcal{R}$.
2. $\forall (x, x_s)$ and $\forall u_s \in \mathbb{U}_s$: $\mathbb{T}_s(\cdot \mid x_s, u_s)\, \bar{\mathcal{R}}_\delta \mathbb{T}(\cdot \mid x, u_v(u_s, x_s, x))$. This is satisfied by constructing a lifted probability measure $\mathbb{W}_{\mathbb{T}}(\cdot \mid u_s, x_s, x)$ based on the shared input noise $d(n)$, as

$$\mathbb{W}_{\mathbb{T}}(dx_s' \times dx' | u_s, x_s, x) = \mathcal{N}\left(\begin{bmatrix} dx_s' \\ dx' \end{bmatrix} | \begin{bmatrix} A_s x_s + B_s u_s \\ Ax + Bu_v(u_s, x_s, x) \end{bmatrix}, \begin{bmatrix} F_s \\ F \end{bmatrix} \begin{bmatrix} F_s \\ F \end{bmatrix}^T\right)$$

From this lifting measure, the original transition kernels can easily be recovered by marginalising over $\mathbb{X}_s$ and over $\mathbb{X}$, respectively, as $\mathbb{T}(\cdot \mid x, u) = \mathcal{N}(\cdot|Ax + Bu_v(u_s, x_s, x), FF^T)$, and $\mathbb{T}_s(\cdot \mid x_s, u_s) = \mathcal{N}(\cdot|A_s x_s + B_s u_s, F_s F_s^T)$. Now we need to prove that with probability at least $1-\delta$ it holds that the pair $(x', x'_s) \in \mathcal{R}$ for realisation $(x', x'_s) \sim \mathbb{W}_{\mathbb{T}}(\cdot \mid u_s, x_s, x)$. We know that the shared input noise $d(n)$ is governed by an i.i.d. Gaussian distribution $d \sim \mathcal{N}(0, I)$ and $d^T d$ has a Chi-square distribution with m_d degrees of freedom. Hence if for all $d^T d \leq c_d$ it holds that the next state belongs to the relation $\mathcal{R}$, then it holds that the next state belongs to the relation with a probability of at least $1 - \delta$.

3. The initialisation can be lifted, that is, $\pi\bar{\mathcal{R}}\pi_s$. Define $\mathbb{W}_\pi = \delta_{x=0}\delta_{x_s=0}$, then with probability $1(\geq 1-\delta)$: $(x_s, x) \in \mathcal{R}$ since $(0, 0) \in \mathcal{R}$.

Notice that also the imposed measurability conditions have been trivially achieved, since all mappings are linear mappings and are hence continuous mappings, see [40] or [41].

□

Computations related to Theorem 7. We can write (20) as an implication based on several quadratic forms:

$$\forall z = \begin{bmatrix} x \\ u_s \\ d \\ 1 \end{bmatrix} : z^T F_x z \geq 0,\ z^T F_u z \geq 0 \text{ and } z^T F_d z \geq 0 \implies z^T F_M z \geq 0. \tag{21}$$

Introduce auxiliary variables $\tau_x, \tau_u, \tau_d > 0$, then a sufficient condition based on the S-procedure [52] for (21) is

$$F_M \geq \tau_x F_x + \tau_u F_u + \tau_d F_d \tag{22}$$

with matrices

$$F_M = \begin{bmatrix} -(A+BK)^T M(A+BK) & -(A+BK)^T M(BR-PB_s) & -(A+BK)^T M(F-PF_s) & 0 \\ -(BR-PB_s)^T M(A+BK) & -(BR-PB_s)^T M(BR-PB_s) & -(BR-PB_s)^T M(F-PF_s) & 0 \\ -(F-PF_s)^T M(A+BK) & -(F-PF_s)^T M(BR-PB_s) & -(F-PF_s)^T M(F-PF_s) & 0 \\ 0 & 0 & 0 & \varepsilon^2 \end{bmatrix}, \tag{23}$$

$$F_x = \begin{bmatrix} -M & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & \varepsilon^2 \end{bmatrix}, \qquad F_u = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & -I & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & c_u \end{bmatrix}, \qquad F_d = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & -I & 0 \\ 0 & 0 & 0 & c_d \end{bmatrix},$$

where I denotes an identity matrix of appropriate size. Written out, this matrix inequality can be split in two inequalities as

$$\begin{bmatrix} (A+BK)^T M(A+BK) - \tau_x M & (A+BK)^T M(BR-PB_s) & (A+BK)^T M(F-PF_s) \\ (BR-PB_s)^T M(A+BK) & (BR-PB_s)^T M(BR-PB_s) - \tau_u I & (BR-PB_s)^T M(F-PF_s) \\ (F-PF_s)^T M(A+BK) & (F-PF_s)^T M(BR-PB_s) & (F-PF_s)^T M(F-PF_s) - \tau_d I \end{bmatrix} \leq 0, \tag{24}$$

$$\tau_u c_u + \tau_d c_d + \tau_x \varepsilon^2 - \varepsilon^2 \leq 0.$$

The latter inequality can be rewritten as

$$\tau_u c_u + \tau_d c_d \leq (1 - \tau_x)\varepsilon^2.$$

This implies that $0 < \tau_x < 1$ and $0 < \varepsilon^2 \leq \frac{\tau_u c_u + \tau_d c_d}{(1-\tau_x)}$. Hence, for given values $\tau_u, c_u, \tau_d, c_d, \tau_x$, the maximal ε that can be obtained is $\sqrt{\frac{\tau_u c_u + \tau_d c_d}{(1-\tau_x)}}$. The inequality in (23) clearly limits the achievable accuracy ε as a function of the variables τ_x, τ_u, τ_d, and of M. As such, in Algorithm 2 we optimise these variables to obtain the best possible accuracy. The given algorithm requires that an interface (17) for which the Sylvester equation has been solved beforehand.

The choice of the interface (17), together with the selection of P and Q such that the Sylvester equation is satisfied,

have a direct impact on the accuracy of the computed ε and δ values. To observe this, note that in inequality (24), which is a crucial part of Algorithm 2, we can extract three sufficient conditions

$$(A + BK)^T M(A + BK) \preceq \tau_x M, \tag{25}$$

$$(BR - PB_s)^T M(BR - PB_s) \preceq \tau_u I, \tag{26}$$

$$(F - PF_s)^T M(F - PF_s) \preceq \tau_d I. \tag{27}$$

Each of the variables τ_u, τ_d and $\tau_x \in (0, 1)$ increases the output deviation ε. Hence we can use these inequalities to resolve the free design variable before using Algorithm 2.

Solving the Sylvester equations. We evaluate the choice of P, Q and R with respect to γ_d and γ_u based on (26) and (27) subject to

$$\gamma_u I \geq (BR - PB_s)^T M(BR - PB_s) \text{ and } \gamma_d I \geq (F - PF_s)^T M(F - PF_s). \tag{28}$$

For a given M this allows us to select P, Q and R by utilising an optimisation algorithm over the corresponding linear matrix inequalities, as in (29) and as summarised in Algorithm 3. The algorithm can be implemented using a convex optimisation toolbox, such as CVX [51, 53], which can solve this problem in polynomial time. Of course, when computing P, Q and R we do not have access to the final M, which will be computed based on Algorithm 2. Instead, we can use a preliminary estimate for M. Notice that Algorithm 3 is currently given with respect to the objective $(\gamma_u + \gamma_d)$, which however can be altered with a weighting factor to trade-off the importance of γ_u and γ_d.

Interface Design. Given P, Q and R, the interface in (17) has one free variable the gain K. The choice of K defines how hard the interface counteracts to deviations in the states. A selection of K to purely minimise these deviations will lead to a very aggressive interface. Therefore in an actual implementation, it makes more sense to select K based both on the deviation in the state (cf. (25)) and based on the input deviation it creates.

Full computational implementation. A full design procedure to obtain the relation, the interface and the quantification of ε and δ is defined as a combination of Algorithm 3 and Algorithm 2, together with the design procedure for K. The main parts of each of the algorithms can be implemented using convex programming in polynomial time. In the case study and toolbox, we will use the CVX toolbox of [53] for these optimisation steps. To design all the free variables, the matrices P, Q and R are first optimised using Algorithm 3, by solving (18) together with the imposed constraints in (28) for an estimate of the design matrix M. We design a gain K by applying an algorithm similar to Algorithm 3 with an additional penalty for large K values. Next, for a given range of δ values the corresponding ε values are computed using Algorithm 2.

Algorithm 2 Computing ε given δ

Given K, P, R, δ
Set $c_d := \chi_2^{-1}(\delta, m_d)$
Initialise $\tau_x := \max(|\operatorname{eig}(A + BK)|)^2$

Minimise* $\quad (\tau_u c_u + \tau_d c_d)/(1 - \tau_x)$

with **arguments**: $\quad M \in \mathbb{R}^{m \times m},\ 0 < \tau_u \in \mathbb{R},\ 0 < \tau_d \in \mathbb{R},\ \tau_x \in (0, 1)$

and **subject to:** $\quad M - C^T C \succeq 0,$

$$\begin{bmatrix} (A + BK)^T M(A + BK) - \tau_x M & (A + BK)^T M(BR - PB_s) & (A + BK)^T M(F - PF_s) \\ (BR - PB_s)^T M(A + BK) & (BR - PB_s)^T M(BR - PB_s) - \tau_u I & (BR - PB_s)^T M(F - PF_s) \\ (F - PF_s)^T M(A + BK) & (F - PF_s)^T M(BR - PB_s) & (F - PF_s)^T M(F - PF_s) - \tau_d I \end{bmatrix} \preceq 0.$$

Set $\varepsilon = \sqrt{(\tau_u c_u + \tau_d c_d)/(1 - \tau_x)}$

[*: Can be implemented as a line search over τ_x together with a convex programming solution]

Algorithm 3 Compute P, Q, R

Given M

Minimise $(\gamma_u + \gamma_d)$

with **arguments**: $P \in \mathbb{R}^{m\times m_s}, Q \in \mathbb{R}^{p\times m_s}, R \in \mathbb{R}^{p\times p}, \gamma_u > 0 \in \mathbb{R}, \gamma_d > 0 \in \mathbb{R}$

and **subject to**: $AP + BQ - PA_s = 0, \quad C_s = CP,$

$$\begin{bmatrix} M & M(BR - PB_s) \\ (BR - PB_s)^T M & \gamma_u I \end{bmatrix} \geq 0, \qquad \begin{bmatrix} M & M(F - PF_s) \\ (F - PF_s)^T M & \gamma_d I \end{bmatrix} \geq 0. \tag{29}$$

Return P, Q, and R.

Remark 5. *We have used the relation $\mathcal{R} := (x - Px_s)^T M(x - Px_s) \leq \varepsilon^2$ and the interface $u := Ru_s + Qx_s + K(x - Px_s)$. This is similar to the hierarchical control problems leveraging approximate simulation relations and interfaces for deterministic systems in [31]. Note that the resulting (ε, δ) relations are dependent on the chosen $\mathcal{R}$ and $\mathcal{U}_v$. For a different choice of relation $\mathcal{R}'$ or of interface $\mathcal{U}'_v$, the optimisation procedure needs to be updated.*

Example 6. *We illustrate the full computational procedure for approximate simulation relations by running the optimisation procedure between the second order and the fourth order models in section 3, namely $\tilde{\mathbf{M}}_2$ in* (9) *and $\tilde{\mathbf{M}}_4$ in* (7).

The models are mapped to the form of (11)*, by splitting the states of the model into a steady state (x_{ss}) and a transient (x_{tr}) component, as follows:*

$$\begin{cases} x_{ss} & = Ax_{ss} + Bu_{ss} + Fd_m, \\ x_{tr}(n+1) & = Ax_{tr}(n) + Bu_{tr}(n) + F\Sigma_d^{0.5} e(n), \\ y(n) & = Cx_{ss} + Cx_{tr}(n). \end{cases} \tag{30}$$

Here, u_{ss} represents the nominal input required to achieve the desired steady-state temperature, u_{tr} is the additional input required from the nominal one to reach a new desired temperature, d_m represents the mean noise signal, e is white noise described by $e(n) \sim \mathcal{N}(0, I)$, y_{ss} is the steady-state output, and $y_{tr}(n)$ is the output of the transient state. In order to avoid ill-conditioning, the input signal u is rescaled such that a full heating input $T_{fs1} = 30\,^\circ C$ corresponds to value 1*. Similarly, a steady-state temperature of $T_1 = 20\,^\circ C$ corresponds to $u_{ss} = 0.65$.*

We first proceed by solving the Sylvester equations using a preliminary estimate of matrix M. This is computed as the solution of an infinite-horizon linear quadratic stochastic optimal control problem with respect to the disturbances [54]. The solution is found using the Matlab function `dare`

$$M = \begin{bmatrix} 11.6229 & -0.0224 & 0.0188 & 0.6953 \\ -0.0224 & 11.6229 & -0.0188 & 0.6953 \\ -0.0188 & -0.0188 & 11.6295 & 0.6107 \\ 0.6953 & 0.6953 & 0.6107 & 5.5080 \end{bmatrix}.$$

Using this estimate for M, we obtain the design matrices P, Q and R using Algorithm 3, as

$$P = \begin{bmatrix} 1 & 2.12e-7 \\ 1 & 2.87e-7 \\ 0.8762 & -4.13e-6 \\ 0 & 1 \end{bmatrix}, \qquad Q = \begin{bmatrix} -0.1611 & 0.1597 \end{bmatrix} \qquad \textit{and} \qquad R = 1.007.$$

The stabilising gain K is computed using a procedure similar to Algorithm 3, which incorporates both the state deviations and the additional input required for minimising these state deviations. In this example we obtain

$$K = \begin{bmatrix} -0.0122 & -0.0122 & -0.012 & -4.6903 \end{bmatrix}. \tag{31}$$

Next, we proceed with the computation of (ε, δ) *over logarithmically spaced points* $\delta \in [0.001, 1]$*, under a bounded input set* $\mathbb{U}_{s_2} = \{u_{s_2} \in \mathbb{R} \| u_{s_2}| \leq \frac{1}{30}\}$*, using Algorithm 2. The input set is bounded within an absolute deviation of* $1\ ^\circ C$ *of fluctuations. Algorithm 2 optimises the M matrix for each* δ *value – for instance, for* $\delta = 1$

$$M_{\delta=1} = \begin{bmatrix} 0.5348 & -0.1676 & -0.1508 & 0.0147 \\ -0.1676 & 0.5413 & -0.1433 & 0.0156 \\ -0.1508 & -0.1433 & 0.5806 & 0.0131 \\ 0.0147 & 0.0156 & 0.0131 & 1.3862 \end{bmatrix},$$

whereas for $\delta = 10^{-2}$

$$M_{\delta=10^{-2}} = \begin{bmatrix} 0.0019 & -0.0006 & -0.0013 & -0.0134 \\ -0.0006 & 0.0011 & -0.0003 & -0.0092 \\ -0.0013 & -0.0003 & 0.0021 & -0.0113 \\ -0.0134 & -0.0092 & -0.0113 & 3.1350 \end{bmatrix}.$$

The resulting (ε, δ)*-pairs are shown in Table 1: it can be seen that for increasing values of* δ*,* ε *decreases to a positive lower bound. This lower bound is a function of the size of the set* $\mathbb{U}_{s_2}$*. Note that a large value of* δ *corresponds to a discount in the probability of the controller satisfying* ψ*, while a large value of* ε *corresponds to a larger error in the output bound.*

δ	1	$10^{\frac{-1}{3}}$	$10^{\frac{-2}{3}}$	10^{-1}	$10^{\frac{-4}{3}}$	$10^{\frac{-5}{3}}$	10^{-2}	$10^{\frac{-7}{3}}$	$10^{\frac{-8}{3}}$	10^{-3}
ε	5.021e-5	0.1061	0.1266	0.1420	0.1548	0.1660	0.1761	0.1854	0.1941	0.2022

Table 1: Trade-off between $\delta \in [0.001, 1]$ and ε for pair of concrete and abstract models $(\tilde{\mathbf{M}}_4, \tilde{\mathbf{M}}_2)$.

For a chosen (ε, δ)*-pair, the resulting relation is described using*

$$\mathcal{R} = \left\{(\hat{x}_{s_4}, \hat{x}_{s_2}) \mid (\hat{x}_{s_4} - P\hat{x}_{s_2})^T M_\delta (\hat{x}_{s_4} - P\hat{x}_{s_2}) \leq \varepsilon^2\right\}, \tag{32}$$

with the corresponding interface given as

$$\hat{u}_{s_4} = u_v(\hat{u}_{s_2}, \hat{x}_{s_2}, \hat{x}_{s_4}) = R\hat{u}_{s_2} + Q\hat{x}_{s_2} + K(\hat{x}_{s_4} - P\hat{x}_{s_2}). \tag{33}$$

Here we have $\hat{x}_{s_4} = x_{s_4} - x_{s_{4_{ss}}}$*,* $\hat{x}_{s_2} = x_{s_2} - x_{s_{2_{ss}}}$ *and* $\hat{u}_{s_4} = u_{s_4} - u_{s_{4_{ss}}}$*,* $\hat{u}_{s_2} = u_{s_2} - u_{s_{2_{ss}}}$*.*

4.4. Quality of computed (ε, δ)*-pair*

The algorithms for computing the approximate simulation relations (cf. Algorithms 2 and 3) are composed of a set of linear matrix inequalities that are solved as a semi-definite program. We make use of CVX which solves the semi-definite program in polynomial time [53]. We perform a set of experiments on Algorithm 2 to show the sensitivity and time-complexity of the algorithm. Both tests are run on an Intel(R) Core(TM) i7-4702HQ CPU running at 2.20GHz with 16GB of RAM. First, we compare the time taken to compute the (ε, δ)-relation over the range $\delta \in [0.001, 1]$, for different model orders. The recorded total time taken (in seconds) is presented in Table 2, from which we deduce that

1. the total time taken is dependent on the complexity of the concrete model order. The higher the complexity of the concrete model, the longer it takes to compute the (ε, δ)-relation (cf. $\mathbf{M}_7$ vs the rest of the models).
2. for a specific concrete model order, the total time taken is not directly related to the abstract model order but is dependent on the structure of the underlying dynamics between the concrete and abstract model.

Second, we test the sensitivity of the algorithm to the (user-defined) gain matrix K, which directly effects the additional input required for minimising state deviations on the (ε, δ)-relation. In Algorithms 2 and 3, the gain matrix K is being calculated in a less optimal manner, thus changes in K may effect the (ε, δ)-relation. We proceed with analysing the sensitivity to variations in K over the range of $\delta \in [0.001, 1]$ and using $\mathbf{M}_c = \tilde{\mathbf{M}}_4$ and $\mathbf{M}_a = \tilde{\mathbf{M}}_2$. The gain matrix K is is perturbed using a uniformly distributed array of random numbers ranging between 0 and 2, where

K corresponds to the original set of values given in (31). The experiment is run for 100 times. We show the obtained results in Figure 7, where we witness a maximum variance of 0.0052 (this occurred when $\delta = 10^{\frac{-7}{3}}$).

Similarly, we test the effect of changing the the maximum iteration and the learning rate of the line search employed within Algorithm 2. In both cases, negligible difference in the obtained ε values is achieved with a maximum variance of 6.7650e-8.

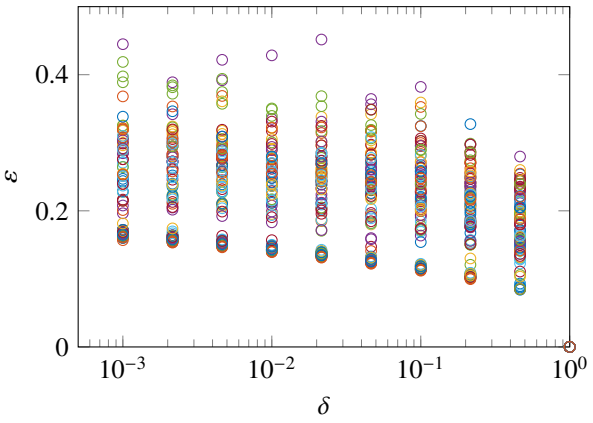


Figure 7: Effect of varying K matrix on (ε, δ) relation.

Concrete Model Order	**Abstract Model Order**	**Time taken**
7	4	168.60
7	3	175.2
7	2	210.0
7	1	176.4
4	3	100.8
4	2	167.4
4	1	152.4
3	2	126.0
3	1	153.0
2	1	109.2

Table 2: Time taken, in seconds, to compute (ε, δ)-relation for different model orders

5. Certified control refinement based on approximate simulation relations

For low-order models, certified control policies can be synthesised using controller synthesis tools such as FAUST^2 [17] - a software tool that generates formal abstractions of a gMDP $\mathbf{M}$ as a finite-state Markov decision process. The abstract model is formally related with $\mathbf{M}$ via a user-defined maximum approximation error threshold. For higher order models, we propose the use of certified controller refinement based on approximate simulation relations, which reduce the computational complexity of the policy synthesis procedure from the high-order model to a lower order model, while providing performance guarantees. Next, we discuss the ideas underlying certified control refinement after which it is shown that the refined controller induces a strategy, as per Def. 4.

Consider two gMDP $\mathbf{M}_i = (\mathbb{X}_i, \pi_i, \mathbb{T}_i, \mathbb{U}, h_i, \mathbb{Y})$ $i = a, c$ with $\mathbf{M}_a \preceq \mathbf{M}_c$. Given the entities $\mathcal{U}_v$ and $\mathbb{W}_\mathbb{T}$ associated to $\mathbf{M}_a \preceq \mathbf{M}_c$, the distribution of the next state $x_c{}'$ of $\mathbf{M}_c$ is given as $\mathbb{T}_c(\cdot\,|x_c, \mathcal{U}_v(u_a, x_a, x_c))$, and is equivalently defined via the lifted measure as the marginal of $\mathbb{W}_\mathbb{T}(\cdot\,|u_a, x_a, x_c)$ on $\mathbb{X}_c$. Therefore, the distribution of the combined next state $(x_a{}', x_c{}')$, defined as $\mathbb{W}_\mathbb{T}(\cdot\,|u_a, x_a, x_c)$, can be expressed as

$$\mathbb{W}_\mathbb{T}(dx_a{}' \times dx_c{}'|u_a, x_a, x_c) = \mathbb{W}_\mathbb{T}(dx_a{}'|x_c{}', u_a, x_a, x_c)\mathbb{T}_c(dx_c{}'|x_c, \mathcal{U}_v(u_a, x_a, x_c)), \tag{34}$$

where $\mathbb{W}_\mathbb{T}(dx_a{}'|x_c{}', u_a, x_a, x_c)$ is referred to as the conditional probability given $x_c{}'$ [55, Corollary 3.1.2]. Similarly, the conditional measure for the initialisation $\mathbb{W}_\pi$ is denoted as

$$\mathbb{W}_\pi(dx_a(0) \times dx_c(0)) = \mathbb{W}_\pi(dx_a(0)|x_c(0))\pi_c(dx_c(0)). \tag{35}$$

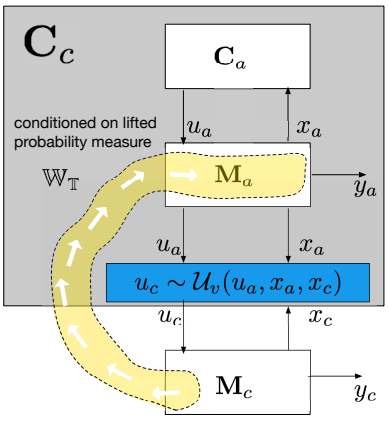
Now suppose that we have a control strategy for $\mathbf{M}_a$, referred to as $\mathbf{C}_a$, and we want to construct the refined control strategy $\mathbf{C}_c$ for $\mathbf{M}_c$, which is such that events defined over the output space have equal probability. This refinement procedure follows directly from the interface and the conditional probability distributions, and is described in Algorithm 4. In the figure within Algorithm 4, a pictorial description is portrayed, the algorithm is separated into the refined control strategy $\mathbf{C}_c$ and its gMDP $\mathbf{M}_c$; $\mathbf{C}_c$ is composed of $\mathbf{C}_a$, the stochastic kernel $\mathbb{W}_\mathbb{T}$, and the interface $\mathcal{U}_v$, and it stores the previous state of $\mathbf{M}_c$.

Algorithm 4 Refinement of control strategy $\mathbf{C}_a$ as $\mathbf{C}_c$

Given

- the interface function $\mathcal{U}_v$ as in (12), and
- the conditional stochastic kernels (34) and (35).

1: $n = 0$
2: draw $x_c(0)$ from π_c,
3: draw $x_a(0)$ from $\mathbb{W}_\pi(\cdot \mid x_c(0))$.
4: **loop**
5: given $x_a(n)$, select $u_a(n)$ according $\mathbf{C}_a$,
6: set $\mu_{a_n} = \mathcal{U}_v(u_a(n), x_a(n), x_c(n))$,
7: draw $x_c(n+1)$ from $\mathbb{T}_c(\cdot \mid x_c(n), \mu_{a_n})$,
8: draw $x_a(n+1)$ from $\mathbb{W}_\mathbb{T}(\cdot \,|x_c(n+1), u_a(n), x_a(n), x_c(n))$,
9: set $n = n + 1$.
10: **end loop**



Theorem 8 (Refined Control Strategy)**.** *Let gMDP* $\mathbf{M}_a$ *and* $\mathbf{M}_c$ *be related as* $\mathbf{M}_a \preceq \mathbf{M}_c$*, and consider the control strategy* $\mathbf{C}_a = (\mathbb{X}_{\mathbf{C}_a}, x_{\mathbf{C}_a 0}, \mathbb{X}_a, \mathbb{T}^n_{\mathbf{C}_a}, h^n_{\mathbf{C}_a})$ *for* $\mathbf{M}_a$ *as given. Then there exists at least one refined control strategy* $\mathbf{C}_c = (\mathbb{X}_{\mathbf{C}_c}, x_{\mathbf{C}_c 0}, \mathbb{X}_c, \mathbb{T}^n_{\mathbf{C}_c}, h^n_{\mathbf{C}_c})$ *as defined in Def. 4, which is an* inhomogenous Markov process *with*

- *state space* $\mathbb{X}_{\mathbf{C}_c} = \mathbb{X}_{\mathbf{C}_a} \times \mathbb{X}_a \times \mathbb{X}_c$*, with elements* $x_{\mathbf{C}_c} = (x_{\mathbf{C}_a}, x_a, x_c)$*;*
- *initial state* $x_{\mathbf{C}_c 0} = (x_{\mathbf{C}_a 0}, 0, 0)$*;*
- *input variable* $x_c \in \mathbb{X}_c$*, namely the state variable of* $\mathbf{M}_c$*;*
- *time-dependent stochastic kernels* $\mathbb{T}^n_{\mathbf{C}_c}$*, defined as*

$$\begin{aligned}\mathbb{T}^0_{\mathbf{C}_c}(dx_{\mathbf{C}_c}|x_{\mathbf{C}_c 0}, x_c(0)) &= \mathbb{T}^0_{\mathbf{C}_a}(dx_{\mathbf{C}_a}|x_{\mathbf{C}_a 0}, x_a)\mathbb{W}_\pi(dx_a|x_c)\delta_{x_c(0)}(dx_c) \text{ and}\\ \mathbb{T}^n_{\mathbf{C}_c}(dx'_{\mathbf{C}_c}|x_{\mathbf{C}_c}(n), x_c(n)) &= \mathbb{T}^n_{\mathbf{C}_a}(dx'_{\mathbf{C}_a}|x_{\mathbf{C}_a}, x_a')\mathbb{W}_\mathbb{T}(dx_a'|x_c', h^n_{\mathbf{C}_a}(x_{\mathbf{C}_a}), x_c, x_a)\delta_{x_c(n)}(dx_c') \text{ for } n \in [1, N];\end{aligned}$$

- *measurable output maps* $h^n_{\mathbf{C}_c}(x_{\mathbf{C}_a}, x_a, x_c) = \mathcal{U}_v(h^n_{\mathbf{C}_a}(x_{\mathbf{C}_a}), x_a, x_c)$. □

since Borel measurable maps are universally measurable and the latter are closed under composition [41, Chapter 7].

Example 7 (Conditional stochastic kernel)**.** *Let us give an example for which it is possible to use elementary computations to get the conditional kernels. Consider an interface* $u_v : \mathbb{U}_a \times \mathbb{X}_a \times \mathbb{X}_c \to \mathbb{U}_a$ *and let* $\mathbb{W}_\mathbb{T}(\cdot \mid u_a, x_a, x_c)$ *be the corresponding lifted transition kernel, which is based on an auxiliary shared noise input* e *taking values in* $\mathbb{R}^{m_d}$ *for models in* (14)*, given as*

$$\mathbb{W}_\mathbb{T}(dx_a' \times dx_c'|u_a, x_a, x_c) = \int_{e\in\mathbb{R}^{m_d}} \delta_{f_a(e)}(dx_a')\delta_{f_c(e)}(dx_c')\mathcal{N}(de\,|\,0, I) \text{ with } \begin{cases} f_a(e) = A_a x_a + B_a u_a + F_a e \\ f_c(e) = A_c x_c + B_c u_v(u_a, x_a, x_c) + F_c e. \end{cases}$$

Notice that $f_a(e)$ *and* $f_c(e)$ *are linear transformations that yield, respectively,* x_a' *and* x_c' *based on the shared noise* e*. The conditional kernel* $\mathbb{T}_c(dx_c'|x_c, u_v(u_a, x_a, x_c))$ *is equal to* $\int_{e\in\mathbb{R}^{m_d}} \delta_{f_c(e)}(dx_c')\mathcal{N}(de\,|\,0, I)$*. To obtain* $\mathbb{W}_\mathbb{T}(dx_a'|x_c', u_a, x_a, x_c)$*, we need to obtain a distribution that is a function of* x_c' *instead of* e*. Assume that* F_c *has full column rank, then we can compute its left inverse as* $F_c{}^+ = (F_c{}^T F_c)^{-1} F_c{}^T$ *for which* $F_c{}^+ F_c = I$*. As such we can define the function inverse* $f_c{}^{-1}(\cdot) : X_c \to \mathbb{R}^{m_d}$ *given as* $f_c{}^{-1}(x_c') = F_c{}^+(x_c' - A_c x_c - B_c u_v(u_a, x_a, x_c))$*. Denoting*[3] *with* $(f_a \circ f_c{}^{-1})$ *the composition*

[3]Following notational standards, $(f_a \circ f_c{}^{-1})(x_c') = f_a(f_c{}^{-1}(x_c')) = A_a x_a + B_a u_a + F_a F_c{}^+(x_c' - A_c x_c - B_c u_v(u_a, x_a, x_c))$.

with f_a, this yields a mapping from $x_c{}'$ to $x_a{}'$. We now have as conditional kernel

$$\mathbb{W}_{\mathbb{T}}(dx_a{}'|x_c{}', u_a, x_a, x_c) = \delta_{(f_a \circ f_c{}^{-1})(x_c{}')}(dx_a{}'), \tag{36}$$

practically this means that the next state of the abstract model can be computed deterministically as a linear function of the next state of the concrete model using the function $(f_a \circ f_c{}^{-1})$, which is implicitly also a function of u_a, x_a, x_c. Based on (36)*, we can now compute the time-dependent stochastic kernels as defined in Theorem 8:*

$$\mathbb{T}^n_{\mathbf{C}_c}(dx'_{\mathbf{C}_c}|x_{\mathbf{C}_c}(n), x_c(n)) = \mathbb{T}^n_{\mathbf{C}_a}(dx'_{\mathbf{C}_a}|x_{\mathbf{C}_a}, x_a{}')\delta_{(f_a \circ f_c{}^{-1})(x_c{}')}(dx_a{}')\delta_{x_c(n)}(dx_c{}') \text{ for } n \in [1, N]. \tag{37}$$

By the above construction of $\mathbf{C}_a$, traces in the output spaces of the closed loop systems $\mathbf{C}_a \times \mathbf{M}_a$ and $\mathbf{C}_c \times \mathbf{M}_c$ have equal distributions, thus it follows that measurable events have equal probability, as stated next.

Theorem 9. *If $\mathbf{M}_a \preceq \mathbf{M}_c$, then for all control strategies $\mathbf{C}_a$ there exists a control strategy $\mathbf{C}_c$ such that, for all measurable events $A \in \mathcal{B}\left(\mathbb{Y}^{N+1}\right)$,*

$$\mathbb{P}_{\mathbf{C}_a \times \mathbf{M}_a}\left(\{y_a(n)\}_{0:N} \in A\right) = \mathbb{P}_{\mathbf{C}_c \times \mathbf{M}_c}\left(\{y_c(n)\}_{0:N} \in A\right),$$

with respective output traces $\{y_a(n)\}_{0:N}$ and $\{y_c(n)\}_{0:N}$ of $\mathbf{C}_a \times \mathbf{M}_a$ and $\mathbf{C}_c \times \mathbf{M}_c$. □

Theorem 10. *If $\mathbf{M}_a \preceq^{\delta}_{\varepsilon} \mathbf{M}_c$, then for all control strategies $\mathbf{C}_a$ there exists a control strategy $\mathbf{C}_c$ such that, for all measurable events $A \subset \mathbb{Y}^{N+1}$*

$$\mathbb{P}_{\mathbf{C}_a \times \mathbf{M}_a}(\{y_a(n)\}_{0:N} \in A_{-\varepsilon}) - \gamma \leq \mathbb{P}_{\mathbf{C}_c \times \mathbf{M}_c}(\{y_c(n)\}_{0:N} \in A) \leq \mathbb{P}_{\mathbf{C}_a \times \mathbf{M}_a}(\{y_a(n)\}_{0:N} \in A_{\varepsilon}) + \gamma,$$

with constant $1 - \gamma = (1 - \delta)^{N+1}$, and with the ε-expansion of A defined as

$$A_{\varepsilon} = \{\{y_{\varepsilon}(n)\}_{0:N} | \exists \{y(n)\}_{0:N} \in A : \max\nolimits_{n \in [0,N]} \mathbf{d}_{\mathbb{Y}}(y_{\varepsilon}(n), y(n)) \leq \varepsilon\}$$

and similarly the ε-contraction defined as $A_{-\varepsilon} = \{\{y(n)\}_{0:N} | \{\{y(n)\}_{0:N}\}_{\varepsilon} \subset A\}$, where $\{\{y(n)\}_{0:N}\}_{\varepsilon}$ is the point-wise ε-expansion of $\{y(n)\}_{0:N}$.

While the details of the proof can be found in [30] its key aspect is the existence of a refined control strategy $\mathbf{C}_c$, which we detail next. Given a control strategy $\mathbf{C}_a$ over the time horizon $n \in \{0, \ldots, N\}$, we denote control strategies that refine $\mathbf{C}_a$ over $\mathbf{M}_c$ as $\mathbf{C}_c$. For $\delta > 0$ the control strategy that refines $\mathbf{C}_a$ for a given interface is non-unique, as argued next. The approximate refinement procedure starts exactly as the exact refinement one, and follows the same execution. This gives again combined state transitions $(x_a, x_c) \rightarrow (x_a{}', x_c{}')$, but now there is a probability (bounded by δ) that $(x_a{}', x_c{}') \notin \mathcal{R}$. When $(x_a, x_c) \notin \mathcal{R}$ the subsequent control strategy can no longer be certified based on $\mathbf{C}_a$. Once $(x_a{}', x_c{}') \notin \mathcal{R}$ we refer to the controls as being in "recovery stage". Since the latter is of no importance for the refinement, the control strategy that refines $\mathbf{C}_a$ is non-unique.

The control strategy is given in Algorithm 5. Whilst the state (x_a, x_c) of $\mathbf{C}_c$ is in $\mathcal{R}$, the control refinement from $\mathbf{C}_a$ (cf. Alg.5 line 4-9) follows the same steps as the exact case in Algorithm 4. Hence, similar to the control refinement for exact probabilistic simulations, the *basic ingredients* of $\mathbf{C}_c$ are the states x_a and x_c, whose stochastic transition to the pair $(x_a{}', x_c{}')$ is governed firstly by a point distribution $\delta_{x_c(n)}(dx_c{}')$ based on the measured state $x_c(n)$ of $\mathbf{M}_c$; and, subsequently, by the lifted probability measure $\mathbb{W}_{\mathbb{T}}(dx_a{}' \mid x_c{}', u_a, x_c, x_a)$, *conditioned* on $x_c{}'$.

On the other hand, whenever the state (x_a, x_c) leaves $\mathcal{R}$ the control chosen by strategy $\mathbf{C}_a$ cannot be refined to $\mathbf{M}_c$: instead, an alternative control strategy $\mathbf{C}_{rec}$ has to be used to control the residual trajectory of $\mathbf{M}_c$. The choice is of no importance to the result in Theorem 10. This recovery stage of the execution (cf. Alg. 5 line 11-15) makes the overall control strategy $\mathbf{C}_c$ non-unique.

By splitting the execution in Algorithm 5 into a control strategy and a gMDP $\mathbf{M}_c$, we can again obtain the refined control strategy.

Theorem 11 (Refined control strategy)**.** *Let gMDP $\mathbf{M}_a$ and $\mathbf{M}_c$, with $\mathbf{M}_a \preceq^{\delta}_{\varepsilon} \mathbf{M}_c$, and control strategy $\mathbf{C}_a = (\mathbb{X}_{\mathbf{C}_a}, x_{\mathbf{C}_a 0}, \mathbb{X}_a, \mathbb{T}^n_{\mathbf{C}_a}, h^n_{\mathbf{C}_a})$ for $\mathbf{M}_a$ be given. Then for any given recovery control strategy $\mathbf{C}_{rec}$, a refined control*

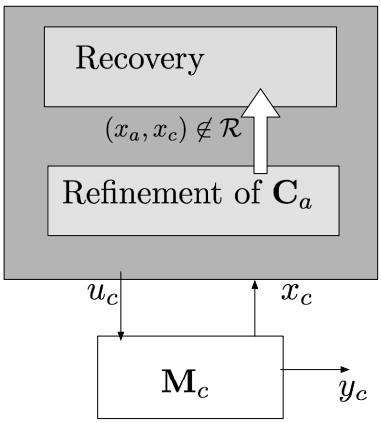
Algorithm 5 (ε, δ)-Refinement of control strategy $\mathbf{C}_a$ as $\mathbf{C}_c$

Input: ∘ the interface function $\mathcal{U}_v$,
∘ the relation $\mathcal{R}$,
∘ the stochastic kernels

$$\mathbb{W}_{\mathbb{T}}(dx_a{}'|x_c{}', u_a, x_a, x_c) \text{ and } \mathbb{W}_{\pi}(dx_a(0)|x_c(0)),$$

∘ the control strategy $\mathbf{C}_a$, and {Control strategy}
∘ the chosen recovery strategy $\mathbf{C}_{rec}$. {Recovery strategy}

Execute:
set $n = 0$ {Start}
draw $x_c(0)$ from π_c
draw $x_a(0)$ from $\mathbb{W}_{\pi}(\cdot \mid x_c(0))$
while $(x_a(n), x_c(n)) \in \mathcal{R}$ **do** {Refine}
  given $x_a(n)$, select $u_a(n)$ from $\mathbf{C}_a$,
  set input $\mu_{cn} = \mathcal{U}_v(u_a(n), x_a(n), x_c(n))$,
  draw $x_c(n+1)$ from $\mathbb{T}_c(\cdot \mid x_c(n), \mu_{cn})$,
  draw $x_a(n+1)$ from $\mathbb{W}_{\mathbb{T}}(\cdot \mid x_c(n+1), u_a(n), x_a(n), x_c(n))$,
  set $n = n + 1$
end while
loop {Recover}
  given $x_c(n)$, select μ_n (from $\mathbf{C}_{rec}$),
  draw $x_c(n+1)$ from $\mathbb{T}_c(\cdot \mid x_c(n), \mu_n)$,
  set $n = n + 1$
end loop



strategy, denoted as

$$\mathbf{C}_c = (\mathbb{X}_{\mathbf{C}_c}, x_{\mathbf{C}_c 0}, \mathbb{X}_c, \mathbb{T}^n_{\mathbf{C}_c}, h^n_{\mathbf{C}_c}),$$

can be obtained as an inhomogeneous Markov process *whose modes of operation can be categorised as* {refinement} *and* {recovery}*, based on Algorithm 5.*

Example 8. *In this example we use the second* $\tilde{\mathbf{M}}_2$ *and fourth* $\tilde{\mathbf{M}}_4$ *models , for which we have computed the approximate similarity relation in* (32)*, with the respective interface* u_v *as given in* (33)*. To clarify the control refinement step, we take a trivial control strategy that applies a constant input to* $\tilde{\mathbf{M}}_2$ *(cf. Example* ***3.a*** *with* $\mathbf{C}_{22^\circ C}$*) and we refine it to* $\tilde{\mathbf{M}}_4$ *as a strategy denoted with* $\mathbf{C}_{r22^\circ C}$*. We first compute the conditional stochastic kernel as in* (36)*,*

$$\mathbb{W}_{\mathbb{T}}(dx'_{s_2}|x'_{s_4}, u_{s_2}, x_{s_2}, x_{s_4}) = \delta_{(f_{s_2} \circ f_{s_4}{}^{-1})(x'_{s_4})}(dx'_{s_2}), \tag{38}$$

where $f_{s_2} \circ f^{-1}_{s_4} = A_{s_2}x_{s_2} + B_{s_2}u_{s_2} + F_{s_2}F^+_{s_4}(x'_{s_4} - A_{s_4}x_{s_4} - B_{s_4}u_v(u_{s_2}, x_{s_2}, x_{s_4}))$ *and* $F^+_{s_4} = (F^T_{s_4}F_{s_4})^{-1}F^T_{s_4}$ *for which* $F^+_{s_4}F_{s_4} = I$*. Thus,* $\mathbb{W}_{\mathbb{T}}$ *is a deterministic function given by* $f_{s_2} \circ f^{-1}_{s_4}$ *based on the signals* $x'_{s_4}, u_{s_2}, x_{s_2}, x_{s_4}$*.*

This allows us to compute the time-dependent conditional kernels of the strategy refinement for $n \in [1, N]$ *as*

$$\mathbb{T}^n_{\mathbf{C}_{r22\,\circ C}}(dx'_{\mathbf{C}_{r22\,\circ C}}|x_{\mathbf{C}_{r22\,\circ C}}(n), x_{s_4}(n)) = \mathbb{T}^n_{\mathbf{C}_{22\,\circ C}}(dx'_{\mathbf{C}_{22\,\circ C}}|x_{\mathbf{C}_{22\,\circ C}}, x'_{s_2})\mathbb{W}_{\mathbb{T}}(dx'_{s_2}|x'_{s_4}, u_{s_2}, x_{s_2}, x_{s_4})\delta_{x_{s_4}(n)}(dx'_{s_4}). \tag{39}$$

Since the chosen controller is trivially simple with a single auxiliary state, and since the conditional kernel (39) *is*

given as a Dirac distribution, it follows that we can define the state updates as a deterministic mapping:

$$x_{\mathbf{C}_{22}\circ_C}(n+1) = q_1, \tag{40}$$
$$x_{s_2}(n+1) = A_{s_2}x_{s_2}(n) + B_{s_2}u_{s_2}(n) + F_{s_2}F^{+}_{s_4}\left(x_{s_4}(n+1) - A_{s_4}x_{s_4}(n) - B_{s_4}u_{s_4}(n)\right), \tag{41}$$
$$x_{s_4}(n+1) = A_{s_4}x_{s_4}(n) + B_{s_2}u_{s_4}(n) + F_{s_4}e(n). \tag{42}$$

The output mapping of the refined control strategy is then defined as,

$$h^n_{\mathbf{C}_{r22}\circ_C}(x_{\mathbf{C}_{22}\circ_C}, x_{s_2}, x_{s_4}) = u_v(h^n_{\mathbf{C}_{22}\circ_C}(x_{\mathbf{C}_{22}\circ_C}), x_{s_2}, x_{s_4}), \tag{43}$$

where $h^n_{\mathbf{C}_{22}\circ_C}$ denotes the input action from the original strategy which corresponds to input $u_{s_2} = 22\,^\circ C$ at every time step, and u_v corresponds to the interface between $\tilde{\mathbf{M}}_4$ and $\tilde{\mathbf{M}}_2$ given by (33)*. Consequently, we can update* (42) *to,*

$$x_{s_4}(n+1) = A_{s_4}x_{s_4}(n) + B_{s_2}(Ru_{s_2} + Qx_{s_2} + K(x_{s_4} - Px_{s_2})) + F_{s_4}e(n). \tag{44}$$

We can further show that the strategy $\mathbf{C}_{r22\circ_C}$ is a refinement of $\mathbf{C}_{22\circ_C}$: substituting (42) *into* (41) *we get,*

$$x_{s_2}(n+1) = A_{s_2}x_{s_2}(n) + B_{s_2}u_{s_2}(n) + F_{s_2}F^{+}_{s_4}(F_{s_4})e(n), \tag{45}$$

which corresponds to the original state equation of $\tilde{\mathbf{M}}_2$. Thus, while $\tilde{\mathbf{M}}_2$ is controlled with $\mathbf{C}_{22\circ_C}$, $\tilde{\mathbf{M}}_4$ evolves according to (44)*, which corresponds to having the control input given by $\mathbf{C}_{r22\circ_C}$.*

This refinement strategy is not unique (cf. Theorem 11). We define a recovery policy $\mathbf{C}_{rec}$ that corresponds to a constant $u_{s_4}(n) := 21\,^\circ C,\ \forall n \in [0,\infty)$, in a similar way to $\mathbf{C}_{r22\circ_C}$. In this case, the conditional kernel is also given by (39) *which once again is given as a Dirac distribution and the state updates can be represented using* (41) *and* (42) *with $u_{s_4}(n) = 21\,^\circ C,\ \forall n \in [n,\infty)$. The resulting control strategy is then defined as inhomogeneous Markov process with two operating modes corresponding to* {refinement} *and* {recovery}*,*

$$\mathbf{C}_{r22\circ_C} = (\mathbb{X}_{\mathbf{C}_{r22}\circ_C}, x_{\mathbf{C}_{r22}\circ_C 0}, \mathbb{X}_{s_4}, \mathbb{T}^n_{\mathbf{C}_{r22}\circ_C}, h^n_{\mathbf{C}_{r22}\circ_C})$$

where $\mathbb{X}_{\mathbf{C}_{r22}\circ_C}$ is the state space with elements $x_{\mathbf{C}_{r22}\circ_C} = (x_{\mathbf{C}_{22}\circ_C}, x_{s_2}, x_{s_4})$, $x_{\mathbf{C}_{r22}\circ_C 0} = (x_{s_2}(0), 0, 0)$ is the initial state and input variable $x_{s_4} \in \mathbb{X}_{s_4}$. During the {refinement} *mode $T^n_{\mathbf{C}_{r22}\circ_C}$ and $h^n_{\mathbf{C}_{r22}\circ_C}$ correspond to* (44) *and* (43) *respectively, while during the* {recovery} *mode, $T^n_{\mathbf{C}_{r22}\circ_C}$ corresponds to* (42) *and $h^n_{\mathbf{C}_{r22}\circ_C}$ is set to $u_{s_4} = 21\,^\circ C$.*

We further highlight the computation of $\mathbf{C}_{r22\circ_C}$ by executing Algorithm 5 over a time horizon $N := 60$, using the pair $\varepsilon := 0.1761$ and $\delta := 10^{-2}$ obtained from Table 1. We further make use of an initial distribution of $\tilde{\mathbf{M}}_4$ as $\pi_{s_4} = \delta_{x_{s_4}(0)}$ where $x_{s_4}(0) = \begin{bmatrix}20 & 20 & 20 & 20\end{bmatrix}^T$. The algorithm proceeds by computing (44) *and* (41) *using the interface function defined by* (33) *for every time step that the relation given by* (32) *with $\varepsilon := 0.1761$ and $\delta := 10^{-2}$ is satisfied, else it makes use of the recovery strategy C_{ref}. For this example, the relation was always satisfied and the resulting time evolution of $\tilde{\mathbf{M}}_4$ and $\tilde{\mathbf{M}}_2$ are shown in Figure 8.*

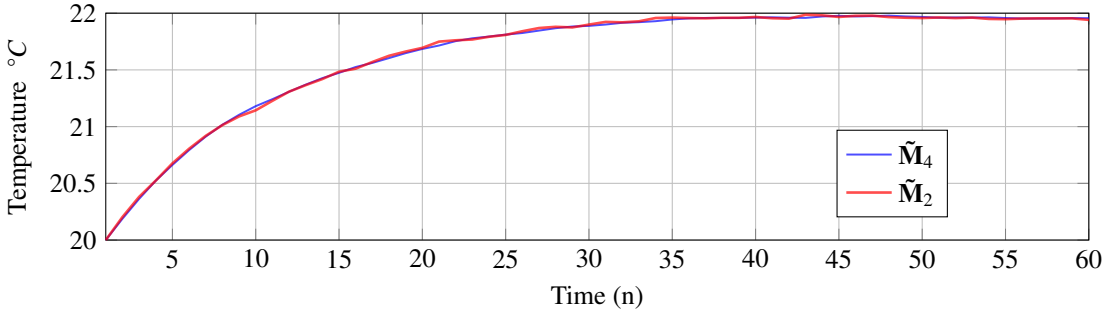


Figure 8: Output trajectories of y_{s_4} and y_{s2} over time horizon ($N = 60$) generated during the first run.

Remark 9. *Algorithm 5 can be extended to attain a quantitative answer on the question whether* $\mathbf{M}_a$ *is an approximation of* $\mathbf{M}_c$*, through the application of the transitivity properties of* $\preceq^{\delta}_{\varepsilon}$*, as presented in [30]. If we consider three gMDP* $\mathbf{M}_i$*,* $i = a, b, c$*, defined by tuples* $(\mathbb{X}_i, \pi_i, \mathbb{T}_i, \mathbb{U}_i, h_i, \mathbb{Y})$ *and if* $\mathbf{M}_a$ *is* $(\varepsilon_a, \delta_a)$*-stochastically simulated by* $\mathbf{M}_b$*, and* $\mathbf{M}_b$ *is* $(\varepsilon_b, \delta_b)$*-stochastically simulated by* $\mathbf{M}_c$*, then* $\mathbf{M}_a$ *is* $(\varepsilon_a + \varepsilon_b), (\delta_a + \delta_b)$*-stochastically simulated by* $\mathbf{M}_c$*. Equivalently, if* $\mathbf{M}_a \preceq^{\delta_a}_{\varepsilon_a} \mathbf{M}_b$ *and* $\mathbf{M}_b \preceq^{\delta_b}_{\varepsilon_b} \mathbf{M}_c$*, then* $\mathbf{M}_a \preceq^{\delta_a+\delta_b}_{\varepsilon_a+\varepsilon_b} \mathbf{M}_c$*.*

6. Case study

This section discusses a simple example that makes use of the inbuilt thermal modelling library (cf. Section 3) and generates certified policies via approximate simulation relations. We consider pairs of concrete and reduced order models: $(\tilde{\mathbf{M}}_7, \tilde{\mathbf{M}}_4)$, $(\tilde{\mathbf{M}}_7, \tilde{\mathbf{M}}_3)$, $(\tilde{\mathbf{M}}_7, \tilde{\mathbf{M}}_2)$, $(\tilde{\mathbf{M}}_7, \tilde{\mathbf{M}}_1)$ and we will index all the model pairs using $i = 4, \ldots, 1$ respectively (corresponding to the model order of the abstract model in the pair). We repeat the process of computing the (ε, δ)-relations for each pair of models, and obtain the model pair that provides the best (ε, δ) trade-off. In other words, we identify the abstract model that simulates the seventh-order concrete model with the best guarantees. For the identified model pair, we then synthesise and refine policies that attain the following specification over the concrete model: to maximise the probability that the deviation of the inner air temperature in room 1 stays within a 0.5 degrees difference from the nominal temperature, over a 45 minute time horizon. This corresponds to the following PCTL property: $\psi = \mathbb{P}_{\geq p}\left[\square^{\leq N=9} |y| \leq 0.5\right]$, for which we want to maximise the probability of staying in this safe set (the models have a 5-minute sampling time, thus $N = 9$).

Computation of (ε, δ) *relation for pairs of models.* The model pairs are first transformed using (30), with input u_{ss} chosen so that a steady-state temperature of $T_1 = 20\ ^\circ C$ is achieved by all the models. This transformation aligns the models to (14) and thus can be used within the framework developed in Section 4.3. We further restrict the input set with an upper bound $c_d = \frac{1}{30}$ that ensures an absolute deviation of $1\ ^\circ C$. The (ε, δ)-approximate simulation relations are computed by first solving the Sylvester equations using Algorithm 3, to obtain the design matrices P, Q, R and then for a given interface gain matrix K. Algorithm 2 is then applied over a range of logarithmically spaced points $\delta \in [0.001, 1]$ and the corresponding ε values are computed. This follows the same procedure in Example 6.

This process is repeated for all the model pairs, and we list the resulting optimised matrices $R_i,\ Q_i,\ K_i,\ P_i,\ M_i,\ i = 4, \ldots, 1$ below. Note that, since Algorithm 2 optimises matrix M for each δ value, we present the optimal matrix M_i for $\delta = 10^{-2}$.

For $\tilde{\mathbf{M}}_4$ we select $R_4 = 1.00004$ and we obtain,

$$Q_4 = \begin{bmatrix} -0.03 & -0.0298 & -0.0298 & -0.0005 \end{bmatrix}, \qquad K_4 = \begin{bmatrix} -0.0122 & -0.0122 & -0.0122 & -1.72e{-}6 & 1.5e{-}6 & -4.6709 & -0.0004 \end{bmatrix},$$

$$P_4 = \begin{bmatrix} -0.0859 & 1.0747 & 0.0128 & 7.27e{-}6 \\ -0.2268 & 1.2269 & -4.6e{-}10 & 7.27e{-}6 \\ 1.6614 & -0.9542 & 1.3344 & 1.39e{-}5 \\ 2.3362 & -1.3473 & 0.0128 & 7.54e{-}6 \\ -2.6488 & 3.6489 & -1.75e{-}9 & 7.54e{-}6 \\ 0 & 0 & 0 & 1 \\ 2.82e{-}9 & 1.11e{-}8 & 6.33e{-}9 & 1 \end{bmatrix}, \qquad M_4 = \begin{bmatrix} 0.2783 & -0.4785 & 0.2074 & -0.2831 & 0.0256 & -0.0276 & -0.0014 \\ -0.4785 & 0.8586 & -0.4024 & 0.4887 & -0.0459 & 0.0487 & 0.0055 \\ 0.2074 & -0.4024 & 0.2141 & -0.2126 & 0.0215 & -0.0558 & -0.0053 \\ -0.2831 & 0.4887 & -0.2126 & 0.2915 & -0.0262 & 0.0237 & 0.0016 \\ 0.0256 & -0.0459 & 0.0215 & -0.0262 & 0.0025 & -0.0030 & -0.0003 \\ -0.0276 & 0.0487 & -0.0558 & 0.0237 & -0.0030 & 2.7883 & 0.0512 \\ -0.0014 & 0.0055 & -0.0053 & 0.0016 & -0.0003 & 0.0512 & 0.0149 \end{bmatrix}.$$

For $\tilde{\mathbf{M}}_3$ we take $R_3 = 1.0001$ and we obtain,

$$Q_3 = \begin{bmatrix} -0.6156 & 0.4546 & 0.0794 \end{bmatrix}, \qquad K_3 = \begin{bmatrix} -0.0123 & -0.0123 & -0.0123 & -9.62e{-}7 & 9.61e{-}7 & -4.7432 & -0.0004 \end{bmatrix},$$

$$P_3 = \begin{bmatrix} -1.8110 & 2.8118 & 0.0001 \\ -2.9031 & 3.9039 & 0.0001 \\ 12.8787 & -11.0053 & -0.0002 \\ 14.0926 & -13.0960 & -0.0003 \\ -18.8129 & 19.8095 & -0.0003 \\ 0 & 0 & 1.0000 \\ -2.02e{-}5 & -6.61e{-}6 & 0.9999 \end{bmatrix}, \qquad M_3 = \begin{bmatrix} 0.1682 & -0.3194 & 0.1330 & -0.1704 & 0.0171 & -0.0246 & -0.0016 \\ -0.3194 & 0.6103 & -0.2592 & 0.3266 & -0.0327 & 0.0274 & 0.0034 \\ 0.1330 & -0.2592 & 0.1704 & -0.1358 & 0.0138 & -0.0502 & -0.0051 \\ -0.1704 & 0.3266 & -0.1358 & 0.1759 & -0.0175 & 0.0211 & 0.0018 \\ 0.0171 & -0.0327 & 0.0138 & -0.0175 & 0.0018 & -0.0017 & -0.0002 \\ -0.0246 & 0.0274 & -0.0502 & 0.0211 & -0.0017 & 2.2569 & 0.0407 \\ -0.0016 & 0.0034 & -0.0051 & 0.0018 & -0.0002 & 0.0407 & 0.0129 \end{bmatrix}.$$

For $\tilde{\mathbf{M}}_2$ we select $R_2 = 1.0006$ and we obtain,

$$Q_2 = \begin{bmatrix} -0.2478 & 0.1592 \end{bmatrix}, \qquad K_2 = \begin{bmatrix} -0.0121 & -0.0121 & -0.0121 & -1.43e{-}6 & -9.70e{-}7 & -4.6464 & -0.00042 \end{bmatrix},$$

$$P_2 = \begin{bmatrix} 1.0019 & 0.00017 \\ 1.0019 & 0.00017 \\ 1.8802 & 0.00034 \\ 1.0026 & 0.0002 \\ 1.00455 & 0.0004 \\ 0 & 1 \\ 1.06e{-}5 & 1 \end{bmatrix}, \qquad M_2 = \begin{bmatrix} 0.5698 & -0.7188 & 0.36372 & -0.5789 & 0.0385 & 0.0489 & 0.0028 \\ -0.7188 & 1.2768 & -0.6480 & 0.7324 & -0.0682 & 0.0851 & 0.0105 \\ 0.3637 & -0.6480 & 0.3310 & -0.3742 & 0.0346 & -0.0733 & -0.0060 \\ -0.5789 & 0.73244 & -0.3742 & 0.5957 & -0.0392 & -0.05465 & -0.0022 \\ 0.0385 & -0.0682 & 0.03462 & -0.0392 & 0.0036 & -0.0049 & -0.00059 \\ 0.0489 & 0.0851 & -0.0733 & -0.0546 & -0.0049 & 4.9291 & 0.0843 \\ 0.0028 & 0.0105 & -0.0060 & -0.0022 & -0.00059 & 0.0843 & 0.021 \end{bmatrix}.$$

For $\tilde{\mathbf{M}}_1$ we take $R_1 = 0.9312$ and we obtain,

$$Q_1 = [\,0.1980\,], \qquad K_1 = [\,-0.0126\ -0.0126\ -0.0126\ -9.86e{-7}\ 1.07e{-7}\ -4.8557\ -0.0004\,],$$

$$P_1 = \begin{bmatrix} -0.0879 \\ -0.0879 \\ -0.1649 \\ -0.0879 \\ -0.0879 \\ 1.0000 \\ 0.9999 \end{bmatrix}, \qquad M_1 = \begin{bmatrix} 0.0792 & -0.0806 & 0.0683 & -0.0814 & 0.0044 & -0.0053 & -3.66e{-5} \\ -0.0806 & 0.1346 & -0.0726 & 0.0810 & -0.0072 & 0.0077 & 0.0007 \\ 0.0683 & -0.0726 & 0.0632 & -0.0709 & 0.0039 & -0.0064 & -0.0006 \\ -0.0814 & 0.0810 & -0.0709 & 0.0845 & -0.0044 & 0.0035 & -0.0001 \\ 0.0044 & -0.0072 & 0.0039 & -0.0044 & 0.0004 & -0.0004 & -0.0002 \\ -0.0053 & 0.0077 & -0.0064 & 0.0035 & -0.0004 & 1.0256 & 0.0274 \\ -3.66e{-5} & 0.0007 & -0.0006 & -0.0001 & -0.0002 & 0.0274 & 0.0475 \end{bmatrix}.$$

Remark 10. *From the resulting projection matrices $P_i, i = 4, \ldots 1$, a relation between the different levels of abstractions is deduced. We note that there is coupling between walls w_2 and w_3, w_4 and w_6. This is expected as these walls are physically connected to each other (cf. Figure 3). From P_4, it can be seen that the states T_{w2} and T_{w6} play a larger role in the wall dynamics than T_{w7} and thus for a lower-order model T_{w7} can be ignored. This directly corresponds to the realistic assumption taken to construct $\tilde{\mathbf{M}}_3$ from $\tilde{\mathbf{M}}_4$. Similarly, from P_3 one can note that T_{w2} plays a larger role than T_{w6} and thus a possible abstraction step is to ignore the dynamics of T_{w6}.*

From the computed reduced-order models, we obtain the resulting trade-off for parameters (ε, δ) in the simulation relation. These are listed in Table 3. Note that with decreasing model order, ε increases: this is due to the assumptions taken to construct the different levels of abstractions (cf. Section 3). For instance, recall that to construct the third order model from the second order model, we assume that the inner walls between the two zones have similar temperature. This assumption reduces the model complexity but consequently increases the output deviation, as witnessed from the obtained ε values. Furthermore, for increasing values of δ, ε decreases to a positive lower bound which is a function of the bounded input set.

δ	1	$10^{\frac{-1}{3}}$	$10^{\frac{-2}{3}}$	10^{-1}	$10^{\frac{-4}{3}}$	$10^{\frac{-5}{3}}$	10^{-2}	$10^{\frac{-7}{3}}$	$10^{\frac{-8}{3}}$	10^{-3}
$\tilde{\mathbf{M}}_4$	0.0002	0.0993	0.1126	0.1200	0.1295	0.1384	0.1463	0.1536	0.1604	0.1668
$\tilde{\mathbf{M}}_3$	0.0001	0.1094	0.1035	0.1153	0.1251	0.1337	0.1413	0.1484	0.1550	0.1611
$\tilde{\mathbf{M}}_2$	0.0001	0.1403	0.1418	0.1580	0.1712	0.1833	<u>0.1939</u>	0.2032	0.21222	0.2206
$\tilde{\mathbf{M}}_1$	0.0006	0.1461	0.1727	0.1924	0.2086	0.2227	0.2358	0.2475	0.2585	0.2688

Table 3: Trade-off between (ε, δ) for concrete ($\tilde{\mathbf{M}}_7$) and abstract model pairs ($\tilde{\mathbf{M}}_i, i = 4, \ldots 1$).

We are interested to select a low-order model that introduces the least errors in the specification and is computationally feasible. Based on these outcomes we proceed with the pair $(\tilde{\mathbf{M}}_7, \tilde{\mathbf{M}}_2)$ and choose the parameters pair $(\varepsilon, \delta) = (0.1939, 10^{-2})$. The resulting relation and interface functions are described as

$$\mathcal{R} = \left\{(\hat{x}, \hat{x}_{s_2}) \mid (\hat{x} - P\hat{x}_{s_2})^T M_2 (\hat{x} - P\hat{x}_{s_2}) \leq (0.1939)^2\right\} \quad \text{and} \quad \hat{u} = u_v(\hat{u}_{s_2}, \hat{x}_{s_2}, \hat{x}) = R\hat{u}_s + Q\hat{x}_s + K(\hat{x} - P\hat{x}_{s_2}), \quad (46)$$

where $\hat{x} = x - x_{ss}$, $\hat{x}_{s_2} = x_{s_2} - x_{s_{2_{ss}}}$ and $\hat{u} = u - u_{ss}$, $\hat{u}_{s_2} = u_{s_2} - u_{s_{2_{ss}}}$.

Controller synthesis for $\tilde{\mathbf{M}}_2$. We set the safety property to $\psi = \mathbb{P}_{\geq p}\left[\Box^{\leq N=9}|y| \leq 0.5\right]$ for the concrete model. For the abstract model we modify it as follows: $\psi_{(\varepsilon,\delta)} = \mathbb{P}_{\geq p+\gamma}\left[\Box^{\leq N=9}|y| < 0.5 - \varepsilon\right] = \mathbb{P}_{\geq p+0.09}\left[\Box^{\leq N=9}|y| \leq 0.30615\right]$. Here γ gives the accumulation of the error in probability over the time horizon of interest, and is obtained from $1-\gamma = (1-\delta)^9$ then $\gamma \leq 9\delta$. Note that y is replaced with the computation of y_{tr} by using (30). This is done since we are only interested in computing the deviations in fluctuations from the nominal zone temperature y_{ss}. Policy synthesis is performed by employing algorithms in FAUST^2 [17], which convert the gMDPs into finite-action models, grid the state space and discretise the input space within specific precision. This precision is computed using the technique presented in [30, Appendix B.1.1] and ensures that the required minimal quantisation error is achieved. Subsequently, a stochastic dynamic programming scheme is set up such that the final value function provides the maximal probability p for the specification $\psi_{(\varepsilon,\delta)}$. This grid-based computation of the safety probability over the nine times steps of the formula leads to a model of size of 9.4×10^7 and to an overall accuracy of 0.15. Over this approximation, we compute the optimal policy $\mu_{(\varepsilon,\delta)}$. This results in $\psi_{(\varepsilon,\delta)}$ being satisfied with a probability of at least $0.99913 - 0.15 = 0.84913$ for the

reduced order model $\tilde{\mathbf{M}}_2$ initialised at origin (zero fluctuation in air temperature in the room). [4]

Controller refinement for $\tilde{\mathbf{M}}_7$. The resulting policy $\mu_{(\varepsilon,\delta)}$ is refined back to μ by employing Algorithm 5 such that ψ is satisfied by the concrete model, in a similar way to the procedure presented in Example 8. The algorithm makes use of the designed relation and interface functions given in (46) and the conditional stochastic kernel given as,

$$\mathbb{T}^n_{\mu}(dx'_{\mu_{(\varepsilon,\delta)}}|x_\mu(n), x_{s_7}(n)) = \mathbb{T}^n_{\mu_{(\varepsilon,\delta)}}(dx'_{\mu_{(\varepsilon,\delta)}}|x_{\mu_{(\varepsilon,\delta)}}, x'_{s_2})\mathbb{W}_{\mathbb{T}}(dx'_{s_2}|x'_{s_7}, u_{s_2}, x_{s_2}, x_{s_7}) \text{ for } n \in [1, 9],$$

to compute the new strategy μ with $\mathbb{W}_{\mathbb{T}}(dx'_{s_2}|x'_{s_7}, u_{s_2}, x_{s_2}, x_{s_7})$ computed using (36). This results in the safety probability for $\tilde{\mathbf{M}}_7$, initialised at the origin, being satisfied with a lower bounded probability of $p = (0.99913 - 0.15 - 0.09) = 0.75913$ with the inner air temperature fluctuations remaining in the bound between $[-0.5, 0.5]$. This is according to Theorem 10.

6.1. Experiments

We show the advantages of employing the developed framework over the current state of the art where certified policies for complex models are synthesised directly. We make use of the software tool FAUST^2 [17] to synthesise the optimal policy that satisfies the specification $\psi := \mathbb{P}_{\geq p-0.15}\left[\Box^{\leq N=4}|y| \leq 0.5\right]$, directly on all the models $(\tilde{\mathbf{M}}_7, \tilde{\mathbf{M}}_4, \tilde{\mathbf{M}}_3, \tilde{\mathbf{M}}_2)$, while we make use of the developed framework to synthesise and refine the policies for the same set of models. For our framework, we compute the policies using the following pair of models $(\tilde{\mathbf{M}}_7, \tilde{\mathbf{M}}_2)$, $(\tilde{\mathbf{M}}_4, \tilde{\mathbf{M}}_2)$, $(\tilde{\mathbf{M}}_3, \tilde{\mathbf{M}}_2)$, $(\tilde{\mathbf{M}}_2, \tilde{\mathbf{M}}_1)$ and the refined policies must satisfy the following specification,$\psi := \mathbb{P}_{\geq p+4\delta-0.15}\left[\Box^{\leq N=4}|y| \leq 0.5\right]$. The policies for the abstract models are also computed using FAUST^2. In both cases, the abstraction procedure performed within FAUST^2 introduces an error of 0.15 in the overall accuracy of the achieved probability of satisfying ψ. All the policies are computed using an Intel(R) Core(TM) i7-4702HQ CPU running at 2.20GHz with 16GB of RAM. We record the total time taken to compute the policies, in seconds, the amount of memory required to store the gridded state-space and the probability of satisfiability of ψ when using the directly synthesised policy or the refined policy. The results are shown in Table 4. One should note that when using FAUST^2 directly, one needs to tighten the bounds of the state-space such that the gridding of the underlying model is of sufficiently small size which allows the synthesis procedure to be performed. This however comes at the expense of the overall probability of satisfying the specification ψ (cf. Table 4). For $\tilde{\mathbf{M}}_7$ the policy could not be computed directly due to large memory usage required: this highlights the advantages of using the developed framework where policies for higher complex order models can be achieved with good certificates.

Model Order of $\tilde{\mathbf{M}}_c$	**Method**	**Time taken**	**Memory**	**p**
7	Refinement, $\tilde{\mathbf{M}}_2, (\varepsilon, \delta) = (0.1939, 0.01)$	279.0	9773664	0.802
4	Refinement, $\tilde{\mathbf{M}}_2, (\varepsilon, \delta) = (0.1761, 0.01)$	145.8	3703425	0.809
3	Refinement, $\tilde{\mathbf{M}}_2, (\varepsilon, \delta) = (0.0503, 0.01)$	157.8	3849772	0.810
2	Refinement, $\tilde{\mathbf{M}}_1, (\varepsilon, \delta) = (0.0470, 0.01)$	175.2	4003725	0.809
7	Directly	-	-	-
4	Directly	999.0	9148370	0.505
3	Directly	874.8	8136942	0.602
2	Directly	155.4	4764384	0.844

Table 4: Comparison between computing the policies directly using the high order model vs computing the refined policies using the developed framework.

Next, we further show the effect of varying the time horizon N of the specification ψ on the total time taken to compute the policy. We run the experiments using the pair of models $(\tilde{\mathbf{M}}_4, \tilde{\mathbf{M}}_2)$ and synthesise the refined policy which satisfies the specification $\psi = \mathbb{P}_{\geq p+N\delta-0.15}\left[\Box^{\leq N}|y| \leq 0.5\right]$. For this case we have, $(\varepsilon, \delta) = (0.1761, 0.01)$. We depict the total time to synthesise policies for a varying maximum time horizon in Figure 9, from which it can be seen that it follows an exponential curve.

[4] The 0.15 error is being introduced due to the gridding in FAUST^2.

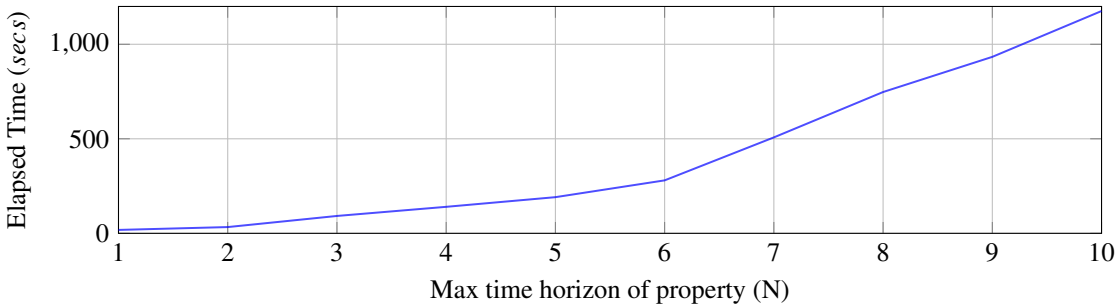


Figure 9: Total time take to synthesise policies as a function of the maximum time horizon considered.

7. Computational implementation

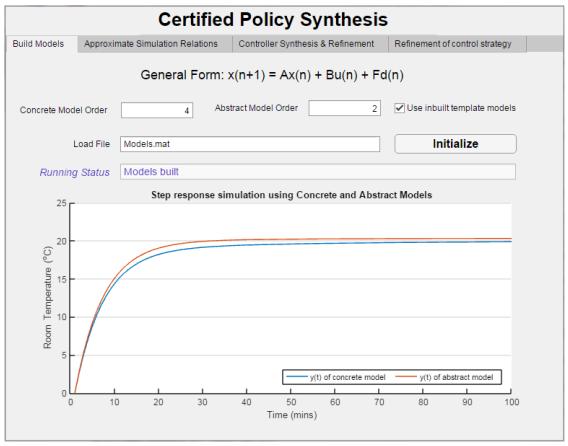
In this section, we introduce a software toolbox for certified policy synthesis, which supports the previously introduced algorithms to compute (ε, δ)-simulation relations (cf. Subsection 4.3), as well as to synthesise refinement strategies for models in the form of (5) (cf. Section 5). The toolbox is implemented in MATLAB and its user interaction is enhanced by a GUI displayed in Figures 10a - 10c. It provides flexibility of synthesising and refining policies for (i) models defined by the user where both the concrete and abstract models are known, (ii) models defined by the user where only the concrete model is known and the abstract model is generated by the toolbox using balancing truncation techniques and (iii) the provided library of models. It also provides the option for the user to input a control policy and refine the policy based on the input interface function and initial conditions.

Constructing models using the toolbox. The first tab, shown in Figure 10a, sets up the models, the concrete ($\mathbf{M}_c$) and abstract ($\mathbf{M}_a$) models, that will be used for synthesising and refining policies. The user has the option of defining models by either making use of

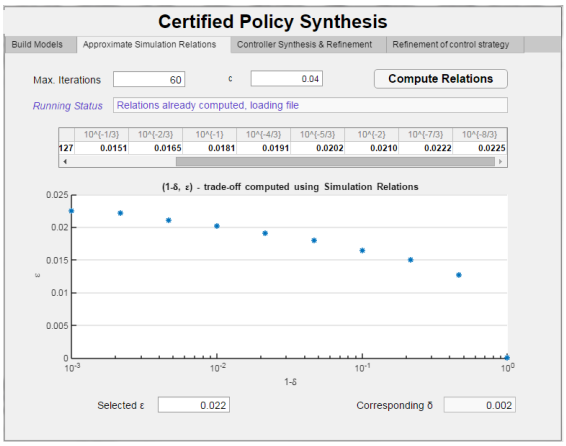
1. the available library of models, each having one control input representing the supply temperature from the HVAC system T_{fs1} (cf. Section 3);
2. or defining concrete and abstract models, entered in the form of corresponding structures stored in a ".mat" file;
3. or defining a concrete model and an order for the abstract model. The corresponding abstract model is constructed using the Matlab `balred` function, which applies a balance reduction algorithm to obtain a lower-order model.

All user-defined models should be of the form given by (5) with one control input, where the input disturbance signals $d(n)$ are described using i.i.d. Gaussian distributions with the mean and covariance of the signals also being supplied to the toolbox. Furthermore, the nominal input should be also supplied by the users. The toolbox normalises and transforms the models to adhere to (14): this allows for the application of the algorithms developed in Section 4.3. It also simulates the models response, when subjected to a unit-step input.

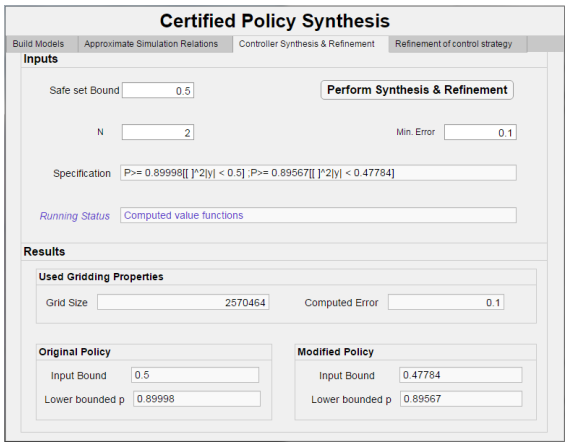
Computing (ε, δ)-relations using toolbox. The second tab (cf. Figure 10b) performs the computation of the (ε, δ)-approximate simulation relations which are used to establish the relationship $\mathbf{M}_a \preceq_\varepsilon^\delta \mathbf{M}_c$. In this step, the models are converted into the equivalent gMDP (cf. Subsection 3.3) and the simulation relations are computed using the optimisation procedure presented in Section 4.3 and highlighted in the case study (cf. Section 6). Algorithms 2 and 3 are solved using CVX, a package for specifying and solving convex programs [51, 53]. In Algorithm 2, the value of c_d is computed using the Matlab function `chi2inv`, while in Algorithm 3 the Matlab `dare` function is used to compute the initial estimates of th M matrix. The ε values are computed over logarithmically spaced points $\delta \in [0.001, 1]$ with a bounded input set $\mathbb{U} = \{u_a \in \mathbb{R} | \, |u_a| \leq c_d\}$, where c_d is user-defined. The toolbox presents the resulting values in the form of a table and a corresponding figure, from which the optimal trade-off between (ε, δ) is selected.



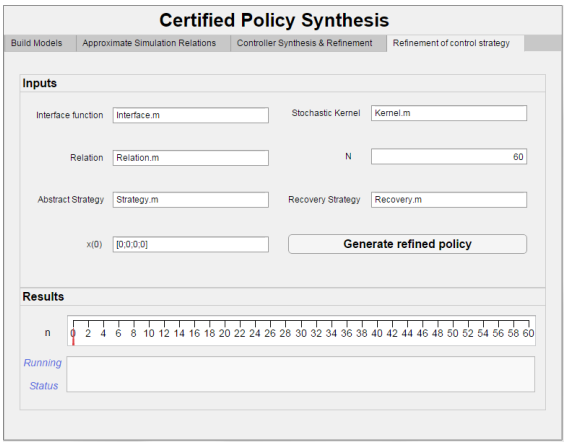
(a) Certified policy synthesis toolbox: models are set up either from the given library, or can be manually entered.



(b) Certified policy synthesis toolbox: computation of simulation relations for the given models.



(c) Certified policy synthesis toolbox: synthesis and refinement of the control strategy.



(d) Certified policy synthesis toolbox: computation of (ε, δ) refinement of user control policy $\mathbf{C}_1$, with user-defined $\mathcal{U}_v$ and $\mathcal{R}$.

Synthesising and refining certified policies using toolbox. This step is performed using the third tab, shown in Figure 10c. The optimal policy μ for a time bounded safety specification having the form (1) is synthesised. The policy is computed based on the previously selected (ε, δ) pair from which the policy $\mu_{(\varepsilon,\delta)}$ for the abstract model is synthesised by employing techniques from FAUST^2 [17], following the same technique presented in the case study (cf. Section 6). Furthermore, we optimise the algorithms in FAUST^2 to use less memory by decoupling noise through a state transform. This allows for storing the discretised probability transitions in a structured and efficient manner. We make use of the Multi-Parametric Toolbox 3 [56] to define polyhedrons representing the transformed state spaces of the abstract model and find the minimum and maximum bounds over which the state space is defined. We proceed by first discretising the input and the state space based on the corresponding bounds. Second, we obtain the stochastic transitions along each of the dimensions. Third, we obtain the deterministic transitions and convert the large deterministic transitions matrix into a sparse matrix using the Matlab function `sparse`. Next, we compute the optimal strategy by solving a stochastic dynamic programming problem for the abstract model. This policy is further refined to μ by using the procedure presented in Example 8. During this refinement step, Algorithm 5 is executed. This employs the developed interface given by (17), the relation described by (16), the conditional kernel computed using (36) and the time-dependent stochastic kernel defined as (37). The design matrices P, K, Q, M for u_v and $\mathcal{R}$ are computed within the computation of (ε, δ)-relations in the second tab using Algorithms 2 and 3.

(ε, δ)-refinement for given policy using toolbox. In the fourth tab, the toolbox presents the option to compute the (ε, δ)-refinement of a user input control strategy $\mathbf{C}_1$ (cf. Figure 10d). This tab executes Algorithm 5, in a similar manner

to the procedure presented in Example 8 over a finite time horizon N. The inputs to the algorithm are specified by the user and represent the control interface u_v, the corresponding relation $\mathcal{R}$ with the selected ε value, the stochastic kernel $\mathbb{W}_T$ having a lower bound of $1-\delta$, the original control policy $\mathbf{C}_1$, and the recovery policy $\mathbf{C}_{rec}$. The (ε, δ) pair are obtained from the previously computed pair in the second tab. This tab gives further flexibility to the toolbox since the abstract policies can be generated, using any software tool and for this policy, the toolbox provides the option to perform policy refinement such that it can be used on more complex models.

In all tabs within the toolbox, a "Running Status" field is present and serves as user feedback that indicates that computations are running or in case of errors what steps must be taken by the user to resolve them.

8. Conclusion

In this work, we have discussed methods for reasoning about levels of abstractions with guarantees. The guarantees are formulated in the form of (ϵ, δ)-approximate simulation relations. Such relations transfer the controller synthesis task from complex models to simpler models, and guarantees are asserted on the refinement of the control policy over the complex models. The concepts have been illustrated via a toolbox and a library of thermal models for a building automation system setup.

We plan to extend the framework, together with the toolbox, to more complex models. The toolbox is a first step in facilitating the use and application of the devised framework to large and complex models. Second, we want to extend the developed toolbox so that controller synthesis and refinement for more general properties in the form of pLTL and PCTL properties can be performed. This will include integrating the toolbox with TULip [57] which is able to synthesise properties for a subset of pLTL specifications. TULip has been developed only for deterministic models, thus further work needs to be done to extend the computation of simulation relations to non-deterministic models.

Acknowledgements

This work has been funded by the European Commission in the Seventh Framework Programme project AMBI (Grant Agreement no. 324432). This work is in part supported by the Alan Turing Institute, London, UK and Malta's ENDEAVOUR Scholarships Scheme.

References

[1] European Parliament and Council of the European Union, Directive 2010/31/EU (2010).
[2] US Department of Energy, Office of Energy Efficiency and Renewable Energy, Energy efficiency trends in residential and commercial buildings (2010).
[3] F. Bernier, J. Ploennigs, D. Pesch, S. Lesecq, T. Basten, M. Boubekeur, D. Denteneer, F. Oltmanns, F. Bonnard, M. Lehmann, et al., Architecture for self-organizing, co-operative and robust building automation systems, in: Industrial Electronics Society, IECON 2013-39th Annual Conference of the IEEE, IEEE, 2013, pp. 7708–7713.
[4] A. Nagy, A. Bratukhin, T. Sauter, Efficient thermal modeling for distributed energy management in industrial buildings, in: Systems Conference (SysCon), 2015 9th Annual IEEE International, 2015, pp. 309–316.
[5] D. Sturzenegger, D. Gyalistras, M. Morari, R. S. Smith, Model predictive climate control of a swiss office building: Implementation, results, and cost–benefit analysis, IEEE Transactions on Control Systems Technology 24 (1) (2016) 1–12.
[6] N. Yu, S. Salakij, R. Chavez, S. Paolucci, M. Sen, P. Antsaklis, Model-based predictive control for building energy management: Part ii–experimental validations, Energy and Buildings 146 (2017) 19–26.
[7] M. B. Rasheed, N. Javaid, A. Ahmad, M. Jamil, Z. A. Khan, U. Qasim, N. Alrajeh, Energy optimization in smart homes using customer preference and dynamic pricing, Energies 9 (8) (2016) 593.
[8] D. Sturzenegger, D. Gyalistras, M. Morari, R. Smith, Model predictive climate control of a swiss office building: Implementation, results and cost-benefit analysis, IEEE Transactions on Control Systems Technology 24 (1) (2016) 1–12.
[9] H. Harb, N. Boyanov, L. Hernandez, R. Streblow, D. Müller, Development and validation of grey-box models for forecasting the thermal response of occupied buildings, Energy and Buildings 117 (2016) 199–207.
[10] M. Fiorentini, J. Wall, Z. Ma, J. H. Braslavsky, P. Cooper, Hybrid model predictive control of a residential {HVAC} system with on-site thermal energy generation and storage, Applied Energy 187 (2017) 465 – 479.
[11] A. Parisio, M. Molinari, D. Varagnolo, K. Johansson, A scenario-based predictive control approach to building HVAC management systems, in: Automation Science and Engineering (CASE), 2013 IEEE International Conference on, 2013, pp. 428–435.
[12] Z. Wu, Q. S. Jia, X. Guan, Optimal control of multiroom hvac system: An event-based approach, IEEE Transactions on Control Systems Technology 24 (2) (2016) 662–669.

[13] K. Li, X. Wenping, C. Xu, H. Mao, A multiple model approach for predictive control of indoor thermal environment with high resolution, Journal of Building Performance Simulation 4 (1) (2017) 1–15.
[14] T. V. Pham, D. H. Nguyen, D. Banjerdpongchai, Decentralized iterative learning control of building temperature control system, in: 2017 SICE International Symposium on Control Systems (SICE ISCS), 2017, pp. 1–7.
[15] T. Brázdil, K. Chatterjee, M. Chmelík, V. Forejt, J. Křetínský, M. Kwiatkowska, D. Parker, M. Ujma, Verification of Markov decision processes using learning algorithms, in: Proceedings 12th International Symposium on Automated Technology for Verification and Analysis (ATVA'14), Vol. 8837 of LNCS, Springer, 2014, pp. 98–114.
[16] O. Holub, K. Macek, HVAC simulation model for advanced diagnostics, in: Intelligent Signal Processing (WISP), 2013 IEEE 8th International Symposium on, 2013, pp. 93–96.
[17] S. Esmaeil Zadeh Soudjani, C. Gevaerts, A. Abate, FAUST^2: Formal abstractions of Uncountable-STate STochastic Processes, in: Tools and Algorithms for the Construction and Analysis of Systems (TACAS), Lecture Notes in Computer Science, Springer Berlin Heidelberg, 2015, pp. 272–286.
[18] S. Haesaert, A. Abate, P. M. J. Van den Hof, Verification of general Markov decision processes by approximate similarity relations and policy refinement, in: G. Agha, B. V. Houdt (Eds.), Quantitative Evaluation of Systems - 13th International Conference, QEST 2016, Quebec City, QC, Canada, August 23-25, 2016, Proceedings, Vol. 9826 of Lecture Notes in Computer Science, Springer, 2016, pp. 227–243.
[19] A. A. Julius, G. J. Pappas, Approximations of stochastic hybrid systems, IEEE Transactions on Automatic Control 54 (6) (2009) 1193–1203.
[20] M. Zamani, P. M. Esfahani, R. Majumdar, A. Abate, J. Lygeros, Symbolic control of stochastic systems via approximately bisimilar finite abstractions, IEEE Transactions on Automatic Control 59 (12) (2014) 3135–3150.
[21] J. Desharnais, V. Gupta, R. Jagadeesan, P. Panangaden, Metrics for labelled Markov processes, Theoretical Computer Science 318 (3) (2004) 323–354.
[22] S. Esmaeil Zadeh Soudjani, A. Abate, Adaptive and sequential gridding procedures for the abstraction and verification of stochastic processes, SIAM Journal on Applied Dynamical Systems 12 (2) (2013) 921–956.
[23] S. Goyal, P.Barooah, A method for model-reduction of non-linear thermal dynamics of multi-zone buildings, Energy and Buildings 47 (2012) 332–340.
[24] M. Mossolly, K. Ghali, N. Ghaddar, Optimal control strategy for a multi-zone air conditioning system using a genetic algorithm, Energy 34 (1) (2009) 58–66.
[25] K. Deng, Model reduction of Markov chains with applications to building systems, Ph.D. thesis, University of Illinois at Urbana-Champaign (2013).
[26] L. Zhao, W. Zhang, A stochastic hybrid system approach to aggregated load modeling for demand response, CoRR abs/1503.06911.
[27] U. Baur, P. Benner, L. Feng, Model order reduction for linear and nonlinear systems: A system-theoretic perspective, Archives of Computational Methods in Engineering 21 (4) (2014) 331–358.
[28] M. G. Safonov, R. Chiang, A schur method for balanced-truncation model reduction, IEEE Transactions on Automatic Control 34 (7) (1989) 729–733.
[29] S. Kung, D. Lin, Optimal hankel-norm model reductions: Multivariable systems, IEEE Transactions on Automatic Control 26 (4) (1981) 832–852. doi:10.1109/TAC.1981.1102736.
[30] S. Haesaert, S. Esmaeil Zadeh Soudjani, A. Abate, Verification of general Markov decision processes by approximate similarity relations and policy refinement, CoRR abs/1605.09557.
URL http://arxiv.org/abs/1605.09557
[31] A. Girard, G. J. Pappas, Hierarchical control system design using approximate simulation, Automatica 45 (2) (2009) 566–571.
[32] M. Maasoumy, A. Pinto, A. Sangiovanni-Vincentelli, Model-based hierarchical optimal control design for HVAC systems, in: ASME 2011 Dynamic Systems and Control Conference and Bath/ASME Symposium on Fluid Power and Motion Control, American Society of Mechanical Engineers, 2011, pp. 271–278.
[33] P. Bacher, H. Madsen, Identifying Suitable Models for the Heat Dynamics of Buildings, Energy and Buildings 43 (7) (2011) 1511–1522.
[34] Y. Lin, T. Middelkoop, P. Barooah, Issues in identification of control-oriented thermal models of zones in multi-zone buildings, in: Decision and Control (CDC), 2012 IEEE 51st Annual Conference on, 2012, pp. 6932–6937.
[35] M. Maasoumy, M. Razmara, M. Shahbakhti, A. Sangiovanni-Vincentelli, Handling Model Uncertainty in Model Predictive Control for Energy Efficient Buildings, Energy and Buildings 77 (2014) 377–392.
[36] M. Maasoumy, A. Sangiovanni-Vincentelli, Total and peak energy consumption minimization of building HVAC systems using model predictive control, IEEE Design Test of Computers 29 (4) (2012) 26–35.
[37] S. Baldi, S. Yuan, P. Endel, O. Holub, Dual estimation: Constructing building energy models from data sampled at low rate, Applied Energy 169 (2016) 81–92.
[38] N. R. Kristensen, H. Madsen, S. B. Jørgensen, Parameter Estimation in Stochastic Grey-box Models, Automatica 40 (2) (2004) 225–237.
[39] K. Kalsi, M. Elizondo, J. Fuller, S. Lu, D. Chassin, Development and validation of aggregated models for thermostatic controlled loads with demand response, in: System Science (HICSS), 2012 45th Hawaii International Conference on, IEEE, 2012, pp. 1959–1966.
[40] V. I. Bogachev, Measure theory, Springer Science & Business Media, 2007.
[41] D. Bertsekas, S. E. Shreve, Stochastic Optimal control : The discrete time case, Athena Scientific, 1996.
[42] S. P. Meyn, R. L. Tweedie, Markov chains and stochastic stability, Communications and Control Engineering Series, Springer-Verlag London Ltd., 1993.
[43] O. Hernández-Lerma, J. B. Lasserre, Discrete-time Markov control processes, Vol. 30 of Applications of Mathematics (New York), Springer Verlag, 1996.
[44] R. Segala, Modeling and verification of randomized distributed real-time systems, Ph.D. thesis, Massachusetts Institute of Technology (1995).
[45] R. Segala, N. Lynch, Probabilistic simulations for probabilistic processes, Nordic Journal of Computing.
[46] B. Jonsson, K. G. Larsen, Specification and refinement of probabilistic processes, in: Proceedings Sixth Annual IEEE Symposium on Logic in Computer Science, 1991, pp. 266–277. doi:10.1109/LICS.1991.151651.
[47] H. J. Skala, The existence of probability measures with given marginals, Ann. Probab. 21 (1) (1993) 136–142.

[48] V. Strassen, The existence of probability measures with given marginals, Ann. Math. Statist. 36 (2) (1965) 423–439.
[49] C. Zhang, J. Pang, On probabilistic alternating simulations, in: C. Calude, V. Sassone (Eds.), Theoretical Computer Science, Vol. 323 of IFIP Advances in Information and Communication Technology, Springer Berlin Heidelberg, 2010, pp. 71–85.
[50] P. Tabuada, Verification and control of hybrid systems, Springer US, 2009.
[51] M. Grant, S. Boyd, CVX: Matlab software for disciplined convex programming, version 2.1, http://cvxr.com/cvx (Mar. 2014).
[52] S. Boyd, L. Vandenberghe, Convex Optimization, CUP, Cambridge, 2004.
[53] M. Grant, S. Boyd, Graph implementations for nonsmooth convex programs, in: V. Blondel, S. Boyd, H. Kimura (Eds.), Recent Advances in Learning and Control, Lecture Notes in Control and Information Sciences, Springer-Verlag Limited, 2008, pp. 95–110, http://stanford.edu/~boyd/graph_dcp.html.
[54] Y. Huang, W. Zhang, H. Zhang, Infinite horizon linear quadratic optimal control for discrete-time stochastic systems, Asian Journal of Control 10 (5) (2008) 608–615.
[55] V. S. Borkar, Probability theory: an advanced course, Springer Verlag, 2012.
[56] M. Herceg, M. Kvasnica, C. Jones, M. Morari, Multi-Parametric Toolbox 3.0, in: Proc. of the European Control Conference, Zürich, Switzerland, 2013, pp. 502–510.
[57] T. Wongpiromsarn, U. Topcu, N. Ozay, H. Xu, R. M. Murray, Tulip: a software toolbox for receding horizon temporal logic planning, in: Proceedings of the 14th international conference on Hybrid systems: computation and control, ACM, 2011, pp. 313–314.

Appendix A. Details on the discrete-time models for building automation systems

We present the corresponding system matrices of the library of discrete time order models described in subsection 3.2. $\tilde{\mathbf{M}}_7$ is given by Equation (6) and is characterised by the following system matrices:

$$A = \begin{bmatrix} 0.9782 & 2.45e{-}6 & 2.45e{-}6 & 1.06e{-}12 & 1.06e{-}12 & 0.0107 & 9.34e{-}9 \\ 2.45e{-}6 & 0.9782 & 2.45e{-}6 & 1.06e{-}12 & 1.06e{-}12 & 0.0107 & 9.34e{-}9 \\ 2.14e{-}6 & 2.14e{-}6 & 0.9783 & 2.45e{-}6 & 2.45e{-}6 & 0.0093 & 0.0107 \\ 9.29e{-}13 & 9.29e{-}13 & 2.45e{-}6 & 0.9782 & 2.45e{-}6 & 8.18e{-}9 & 0.0107 \\ 9.29e{-}13 & 9.29e{-}13 & 2.45e{-}6 & 2.45e{-}6 & 0.9782 & 8.18e{-}9 & 0.0107 \\ 0.00041 & 0.00041 & 0.00041 & 3.57e{-}10 & 3.57e{-}10 & 0.8556 & 2.34e{-}6 \\ 3.13e{-}10 & 3.13e{-}10 & 0.00041 & 0.00041 & 0.00041 & 2.05e{-}6 & 0.8556 \end{bmatrix}, \quad B = \begin{bmatrix} 0.0008 \\ 2.84e{-}5 \\ 2.84e{-}5 \\ 5.33e{-}5 \\ 2.84e{-}5 \\ 2.84e{-}5 \\ 0.0047 \\ 0.0047 \end{bmatrix}, \quad C = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 1 & 0 \end{bmatrix},$$

$$F = \begin{bmatrix} 1.52e{-}8 & 0.0101 & 8.06e{-}7 & 5.91e{-}11 & 4.58e{-}6 & 5.08e{-}10 & -1.29e{-}9 & -2.81e{-}14 \\ 0.0181 & 8.51e{-}9 & 8.06e{-}7 & 5.91e{-}11 & 4.58e{-}6 & 0.0006 & -1.29e{-}9 & -2.81e{-}14 \\ 2.97e{-}7 & 1.59e{-}8 & 7.06e{-}7 & 0.0001 & 1.16e{-}5 & 1.98e{-}10 & -1.13e{-}9 & -6.52e{-}8 \\ 2.83e{-}7 & 0.0101 & 3.04e{-}13 & 0.0001 & 7.64e{-}6 & -2.47e{-}10 & -4.90e{-}16 & -6.52e{-}8 \\ 0.3381 & 8.51e{-}9 & 3.04e{-}13 & 0.0001 & 7.64e{-}6 & -0.0002 & -4.90e{-}16 & -6.52e{-}8 \\ 3.87e{-}6 & 2.16e{-}6 & 0.0001 & 1.99e{-}8 & 0.0007 & 1.29e{-}7 & -2.17e{-}7 & -9.50e{-}12 \\ 7.21e{-}5 & 2.16e{-}6 & 1.029e{-}10 & 0.023 & 0.0012 & -6.27e{-}8 & -1.65e{-}13 & -1.09e{-}5 \end{bmatrix} = \left[\begin{array}{c|c|c|c|c|c|c|c} F_1 & F_2 & F_3 & F_4 & F_5 & F_6 & F_7 & F_8 \end{array}\right].$$

The fourth order models $\tilde{\mathbf{M}}_4$, given by Equation (7), is described using,

$$A_{s4} = \begin{bmatrix} 0.9782 & 2.45e{-}6 & 2.45e{-}6 & 0.0107 \\ 2.45e{-}6 & 0.9782 & 2.45e{-}6 & 0.0107 \\ 2.14e{-}6 & 2.14e{-}6 & 0.9782 & 0.0093 \\ 0.00041 & 0.00041 & 0.00041 & 0.8556 \end{bmatrix}, \quad B_{s4} = \begin{bmatrix} 0.0008 \\ 2.84e{-}5 \\ 2.84e{-}5 \\ 2.49e{-}5 \\ 0.0047 \end{bmatrix}, \quad C_{s4} = \begin{bmatrix} 0 & 0 & 0 & 1 \end{bmatrix},$$

$$F_{s4} = \begin{bmatrix} 1.52e{-}8 & 0.0101 & 8.06e{-}7 & 9.10e{-}9 & 4.58e{-}6 & 5.08e{-}10 & -1.29e{-}9 \\ 0.0181 & 8.51e{-}9 & 8.00e{-}7 & 9.75e{-}9 & 4.08e{-}6 & 0.0006 & -1.70e{-}9 \\ 1.33e{-}8 & 7.46e{-}9 & 7.06e{-}7 & 0.0115 & 4.02e{-}6 & 4.45e{-}10 & -1.13e{-}9 \\ 3.87e{-}6 & 2.1{=}e{-}6 & 0.0001 & 2.46e{-}6 & 0.00077 & 1.29e{-}7 & -2.17e{-}7 \end{bmatrix} = \left[\begin{array}{c|c|c|c|c|c|c} F_{1_{s4}} & F_{2_{s4}} & F_{3_{s4}} & F_{4_{s4}} & F_{5_{s4}} & F_{6_{s4}} & F_{7_{s4}} \end{array}\right].$$

$\tilde{\mathbf{M}}_3$ is given by Equation (8) and has,

$$A_{s3} = \begin{bmatrix} 0.9782 & 2.45e{-}6 & 0.0107 \\ 2.45e{-}6 & 0.9782 & 0.0107 \\ 0.0004 & 0.0004 & 0.8559 \end{bmatrix}, \quad B_{s3} = \begin{bmatrix} 0.0008 \\ 2.842e{-}5 \\ 2.842e{-}5 \\ 0.004 \end{bmatrix}, \quad C_{s3} = \begin{bmatrix} 0 & 0 & 1 \end{bmatrix},$$

$$F_{s3} = \begin{bmatrix} 9.34e{-}9 & 0.0101 & 8.06e{-}7 & 4.58e{-}6 & 5.08e{-}10 & -1.29e{-}9 \\ 0.0111 & 8.51e{-}9 & 8.06e{-}7 & 4.58e{-}6 & 0.0006 & -1.29e{-}9 \\ 2.37e{-}6 & 2.16e{-}6 & 0.0001 & 0.0007 & 1.29e{-}7 & -2.18e{-}7 \end{bmatrix} = \left[\begin{array}{c|c|c|c|c|c} F_{1_{s3}} & F_{2_{s3}} & F_{3_{s3}} & F_{4_{s3}} & F_{5_{s3}} & F_{6_{s3}} \end{array}\right].$$

The second order model $\tilde{\mathbf{M}}_2$ is described using Equation (9) with,

$$A_{s2} = \begin{bmatrix} 0.9782 & 0.0107 \\ 0.0004 & 0.8559 \end{bmatrix}, \quad B_{s2} = \begin{bmatrix} 2.84e{-}5 \\ 0.0047 \end{bmatrix}, \quad C_{s2} = \begin{bmatrix} 0 & 1 \end{bmatrix}$$

$$F_{s2} = \begin{bmatrix} 0.018 & 8.06e{-}7 & 4.59e{-}6 & 0.0006 & -1.29e{-}9 \\ 3.87e{-}6 & 0.0001 & 0.0007 & 1.29e{-}7 & -2.18e{-}7 \end{bmatrix} = \left[\begin{array}{c|c|c|c|c} F_{1_{s2}} & F_{2_{s2}} & F_{3_{s2}} & F_{4_{s2}} & F_{5_{s2}} \end{array}\right].$$

$\tilde{\mathbf{M}}_1$ is described using Equation (10) and has the following system matrices:

$$A_{s1} = [\,0.8564\,], \quad B_{s1} = [\,0.0048\,], \quad C_{s1} = \left[1\right],$$
$$F_{s1} = [\,0.0011\ 0.0001\ 0.00077\ 2.17e{-}5\ -2.18e{-}7\,] = \left[\ F_{1_{s1}} \mid F_{2_{s1}} \mid F_{3_{s1}} \mid F_{4_{s1}} \mid F_{5_{s1}}\ \right].$$