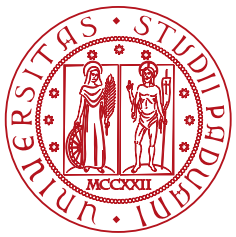


Università degli Studi di Padova

DEPARTMENT OF INFORMATION ENGINEERING
UNIVERSITY OF PADOVA

Dipartimento di Ingegneria dell'Informazione

Corso di Laurea in Control Systems Engineering

Model order reduction, filtering, and control of a pre-heating system for nuclear fusion machines towards real-time operation

Relatore:
Prof. Cenedese Angelo (DEI)

Laureando:
Schio Michele (2019122)

Correlatore:
Prof. Torchio Riccardo (DII)
Prof. Bettini Paolo (DII)

Anno Accademico 2023-2024

Data di Laurea: 3 Luglio 2024

abstract. The present work deals with the noisy state reconstruction problem aimed at virtual output prediction in the context of pre-heating operation of torus-shaped nuclear fusion machines. The creation of a dynamic model based on finite-element method is first studied on a 2D domain and the full-order reconstruction problem is tackled with traditional full-state and reduced-state asymptotic, feedback, observers. A simplified thermodynamic model of a typical nuclear fusion machine is then considered and reduced by means of different model order reduction techniques. Balanced-truncation reduction is retained and studied in the state reconstruction problem with infinite-horizon, optimal H_2, and sub-optimal H_∞ estimators. From a design viewpoint, despite the system being observable, pole allocation is problematic for traditional observer design, resulting in high-gain/peaking phenomena. Computation procedures for optimal, infinite-horizon filters are more elaborate but result in admissible solution of the withstanding Riccati equation. The observability property is investigated numerically as a function sampling time and dimension of the state space. Simulation results on the quality of the virtual output prediction indicate acceptable performance of traditional observer and superior precision of optimal, infinite-horizon, estimators on nominal measurements conditions. Results further indicate that traditional observers are unsuited for virtual measurement prediction in noisy conditions while H_∞ and H_2 estimators perform well with the latter being able to completely reject measurement white noise. Finally, preliminary considerations towards realistic control operations are described and the monitoring problem of the control loop during its operation is discussed.

abstract. Questo lavoro si occupa del problema di ricostruzione dello stato con il fine di stimare una serie di output virtuali nelle operazioni di pre-riscaldamento di macchine toroidali per fusione nucleare. Anzitutto, un modello termico dinamico basato su mesh 2D, e formulato secondo il metodo degli elementi finiti, viene studiato e il problema della ricostruzione viene risolto con osservatori di stato tradizionali. Poi, un modello termico semplificato di una macchina a fusione viene considerato e sottoposto a operazioni di riduzione d'ordine tramite diversi metodi. Il metodo *balanced-truncation* viene scelto e il problema della ricostruzione di stato viene studiato con osservatori basati su principi ottimi in norma H_2 e H_∞. Dal punto di vista del design, nonostante il sistema analizzato sia numericamente osservabile, il problema di allocazione di poli si rivela problematico per gli osservatori tradizionali dando luogo a fenomeni di oscillazioni molto ampie. Il calcolo degli stimatori in norma ottima, invece, si conclude con soluzioni precise dell'equazioni di Riccati sottostante I risultati di simulazione mostrano una qualità accettabile degli osservatori tradizionali e una precisione superiore degli stimatori in norma ottima, in condizioni di misure nominali. Si trova, inoltre, gli osservatori tradizionali non sono adatti alla stima degli output virtuali in condizioni rumorose, mentre gli osservatori H_2 e H_∞ funzionano bene in condizioni di rumore, con lo stimatore H_2 che si dimostra in grado di rigettare rumore bianco. Infine, alcune considerazioni preliminari relative all'implementazione realistica di operazioni di controllo vengono discusse, e il problema del monitoraggio dell'anello di controllo durante il suo funzionamento viene presentato.

Contents

List of Figures

List of Tables

List of Algorithms

Chapter 1

Introduction

In this work, a simplified thermal model of RFX machine hosted at *Consiglio Nazionale di Ricerca* in Padua, is studied as a representative model of a number of old and new nuclear fusion devices. In particular, this work focuses on the dynamic state reconstruction problem appearing as *digital twin* scenarios, where the measured outputs are used to infer predictions on un-accessible, virtual outputs that can be later exploited by a real-time controller. All this, is usually paired with a separate monitoring system that is able to explain the state-space representation used by the control solution.

The pre-heating operations are necessary before the interesting part of nuclear fusion machine performs its duty. This activity is carried out by induced currents applied to the shell of the machine. Such devices are usually torus-shaped and may be equipped with a great number of instruments and sensors. The details of the pre-heating physics by induced currents will be approximated by a direct heat injection but a realistic machine geometry will be used in finite-element analysis.

The goal of this work is to solve the reconstruction problem in order to make good predictions on non-measured output locations (virtual). In other words, given the input signals $u(\bullet)$ and the measured output signals $y_m(\bullet)$, we seek an estimator that is able to compute a state estimate $\hat{x}(\bullet)$ that provides good predictions on the virtual outputs $y_v(\bullet)$.

This thesis is structured as follows. Chapters 2 reviews the foundations of finite-element modelling applied to thermodynamic settings. The creation of a first, simple, synthetic model based on 2D geometry is described. Then, the RFX model is discussed. Chapter 3 reviews some basic concepts in model order reduction techniques, starting with practical considerations and continuing to the the problem of addressing the quality of the reduced model.

Chapters 3 and 4 are the heart of this work. First, traditional observer design is recalled and applied to a full-order thermodynamic model based on simple 2D geometry. Then, more advanced estimators are developed and applied to the same problem in a reduced-order model of RFX machine.

In particular, Chapter 4 deals with state reconstruction of simple full-order thermodynamic models aimed at virtual output predictions. The design of full-state and reduced-state observers is described and analysed from simulation results. Chapter 5, on the other hand, solves the same reconstruction problem on reduced-order model of RFX by describing the design of H_2 optimal, and H_∞ sub-optimal estimators framed for infinite-time prediction.

Finally, chapter 6 anticipates key issues in the design of control loop with an eye towards practical implementation and presents the problem of monitoring for realistic operation. A list of future work activities is proposed. A code repository (available online) has been developed during the thesis activity and fundamental algorithms have been described with great detail in appendix A. Appendix B contains further simulation results that could not fit inside the main structure.

Acronyms and notation

The following nomenclature conventions will be used. Finite-Element Method (FEM), partial differential equation (PDE), and Model Order Reduction (MOR). Differential Algebraic Riccati Equation (DARE), singular value decomposition (SVD), boundary condition (BC).

Output signals will be distinguished in *measured* and *virtual* . The former refer to signals stemming from instruments and real sensors, while the latter refer to un-accessible signals that the used wishes to predict.

Linear time-invariant (LTI) state-space systems will be represented with a tuple o four matrices (A, B, C, D) or (F, G, H, J) (F, G, L, M) depending on the context. The two sets of notations are used for distinguish fundamental changes in the representation of the same system, such as the transformation from continuous-time to discrete time. A second variant of the latter, is used when the virtual outputs need to be clearly separated from the measured outputs.

Dynamic matrices of multi-input multi-output (MIMO) refer to LTI matrices of the state space formulation, either in *descriptor* dynamic form (E, A, B, C, D)

$$\begin{aligned} E\dot{x} &= Ax + Bu \\ y &= Cx + Du \end{aligned}$$

or in *standard* dynamic form (A, B, C, D)

$$\begin{aligned} \dot{x} &= Ax + Bu \\ y &= Cx + Du \end{aligned}$$

In such representations, the letters x and u are general tokens for the state and the input, respectively. Letters u and v are used to address control and noise inputs (respectively), but the letter v can also be used in some contexts to represent measurement noise. Letters $y_m \in \mathbb{R}^p$ or $y_v \in \mathbb{R}^q$ are generally referred to measured and virtual inputs (respectively)

State-space models can be formulated in continuous-time (CT) and/or discrete-time (DT) depending on the context. When the state of a system is reduced though MOR routines, the resulting system is referred to as *reduced-order* Traditional observer, will be called *full-state* and *reduced-space*, to avoid confusion with the property of the state being *full-order* or *reduced-order*

Lastly, $\mathbb{M}_{\mathbb{R}}^{a\times b}$ denote the set of rectangular matrices of size $a \times b$ whose entries are real $\mathbb{R}$. The set of square matrices of size n is denoted by $\mathbb{M}^n$. Symbol † stands for pseudo inverse, and $*$ for complex conjugate. When • and ◦ are used inside a table, they represent graphically the boolean values yes (true), and no (false), respectively.

Chapter 2

Physical modelling: finite-element-based thermodynamics

Contents

2.1 Finite-element method for thermodynamic models

2.1.1 Foundations of finite-element theory

The traditional Partial Differential Equation (PDE) in *strong* formulation for a thermal problem can be written, at all times t and for all positions $\vec{s}$ in the domain $\mathcal{D}$, as

$$\rho c \partial_t T(t,\vec{s}) = -\lambda \nabla^2 T(t,\vec{s}) \qquad \forall t \quad \forall \vec{s} \in \mathcal{D}$$

where ρ (density), c (specific heat), λ (specific conductivity) are constant with respect to space and time.

Numerical treatment of this problem can be tackled with the definition of elementary subdivisions of the domain $\mathcal{D}$ with n nodes and a number of edges. By introducing *test functions* $v_1(\vec{s}), \ldots, v_n(\vec{s})$ [1] that allow to interpolate n mesh nodes (for example in linear fashion), the strong formulation can be given a *weak form*

$$\int_D \rho c \partial_t T(t,\vec{s}) v(\vec{s}) dV = \int_D -\lambda \nabla^2 T(t,\vec{s}) v(\vec{s}) dV \quad \forall t \quad \forall v(\vec{s})$$

Integration by parts (Green's first identity) allows to express second-order Laplacian $\nabla^2\varphi$ (with $\varphi \sim T$) in terms of two first-order gradients $\nabla v \cdot \nabla\varphi$ and additional conditions on the boundary $\partial\mathcal{D}$ of the problem domain $\mathcal{D}$

$$\int_d v \nabla^2 \varphi dV = \oint_{\partial\mathcal{D}} v \nabla\varphi \cdot d\vec{S} - \int_{\mathcal{D}} \nabla v \cdot \nabla\varphi dV$$

Galerkin method (FEM) consists of projecting the weak formulation onto *trial functions* $\phi_1(\vec{s}), \ldots, \phi_n(\vec{s})$ in order to obtain a solution that can be written in the form $T(t,\vec{s}) = \sum_j T_j(t)\phi_j(\vec{s})$, thus resulting, for $i = 1, \ldots, n$, in

$$\sum_{j=1}^{n} \dot{T}_j \left(\int_D \rho c \, v_i \phi_j dV \right) = \oint_{\partial D} -\lambda v_i(\vec{s}) \nabla T(t,x) \cdot d\vec{S} \; + \; \sum_{j=1}^{n} T_j \; \left(\int_D \lambda \nabla v_i \cdot \nabla \phi_j \right) dV$$

Often, the test functions $v_i(\vec{s})$ are chosen identical to the trial functions $\phi_j(\vec{s})$.

2.1.2 Elements of a thermodynamic model

Consider the partial differential equation

$$\rho c \partial_t T - \nabla\cdot(k\nabla T) = \varrho \qquad + \text{boundary conditions}$$

where:

- $T(t, \vec{s})$ is a time varying temperature field defined in a spatial domain D with boundary ∂D
- $\rho > 0$ denotes the *material density*
- $c > 0$ represents the *specific heat*
- $k > 0$ denotes the *thermal conductivity*
- ϱ is a power source density measured in Watt per unit volume (or unit surface, if the domain is 2D)

Boundary conditions may take one of the following formulation:

- **Dirichlet** (set temperature) on a sector Γ of the boundary ∂D, impose $T(t, \vec{s}) = T^* \quad \vec{s} \in \Gamma$
- **Robin** (convection) on a sector Γ of the boundary ∂D, impose the gradient $\vec{n} \cdot k\nabla T = h(T_{\text{amb}} - T(t, \vec{s})) \quad \vec{s} \in \Gamma$ where $h > 0$ is the convection coefficient
- **Neumann** (heat flow transfer) on a sector Γ of the boundary ∂D, impose the gradient $k\nabla T = q_{\text{Neu}}$

Let the temperature field be approximated as $T(t, \vec{s}) \approx \sum_i N_i(\vec{s}) X_i(t)$ where $N_i(\vec{s}) : D \to \mathbb{R}$ are suitable basis functions (for example hat functions in 1D) and $X_i(t) : \mathbb{R} \to \mathbb{R}$ are temperature values at mesh nodes. To apply *Galerkin testing* means to *weight* the differential equation with test functions equal to the basis functions, thus obtaining (skipping detailed integration)

$$M\dot{x} + Kx = q$$

where:

- $x(t)$ is the vector of all note temperatures $X_i(t)$, and $\dot{x}$ denotes its time-derivative
- $M \in \mathbb{M}^n$ is the *mass* matrix
- $K \in \mathbb{M}^n$ is the *stiffness* matrix
- q is the resulting power source (Watts), integrated over the domain D.

Applying the boundary conditions to the PDE model results in the following modifications:

- Robin (convection) on a segment Γ_1 of the domain boundary ∂D will produce two contributions: one on the temperature variable x and the other on the power source q

  $$M\dot{x} + Kx + Hx = q + q_{\text{conv}}$$

  where $q_{\text{conv}} = \int_{\Gamma_1} N_i h T_{\text{amb}} ds$ (line/ surface integral) and $H \in \mathbb{M}^n$

- Dirichlet on a segment Γ_2 , let d denote the (set of) nodes indexes where Dirichlet conditions have been imposed, then $x_{[d]} = T^*$

The modified system becomes:

$$M\dot{x} + Kx + Hx = q + q_{\text{conv}} + \text{rhs}_d$$

where:

- rhs_d denote the (right hand side) imposed contributions from Dirichlet conditions
- if $T^* = T^*(t)$ then $\dot{T}^*$ also affects the dynamics of the model (see below)

In practice, the following steps may be implemented:

1. store $q_K = K_{[:,d]}$, $q_H = K_{[:,d]}$ and $q_M = M_{[:,d]}$. These represent the contributions of the Dirichlet conditions on the *free* nodes
2. remove Dirichlet contributions from the system dynamics by setting `M(:,d)=0`, `K(:,d)=0`, and `H(:,d)=0`.
3. the resulting system has zero-dynamics rows (algebraic constrains on the Dirichlet nodes)
$$M'\dot{x} + K'x + H'x = q + q_{\text{conv}} + \text{rhs}_d + \underbrace{(q_K + q_H)T^* + q_M\dot{T}^*}_{\text{input signal}}$$
4. remove the rows corresponding to Dirichlet nodes, thus reducing the number of equations and restricting the state dynamics

In this work, the full-order models will have Dirichlet boundary conditions only for the part concerning chapter 4 (see SEC. 2.2 for details). chapter 5 will use a simplified state dynamic equation where no Dirichlet conditions need to be handled.

$$\underbrace{M}_{E}\dot{x} = \underbrace{-(K+H)}_{A}x + \underbrace{\begin{bmatrix} q_{\text{conv}} & q \end{bmatrix}}_{B}\begin{pmatrix} T_{\text{ref}} \\ u \end{pmatrix} \tag{2.1}$$

2.2 Creation of thermodynamic models based of 2D mesh

MATLAB's PDEs toolbox [2] can solve a system of the type (*strong* formulation):

$$m\frac{\partial^2 u}{\partial t^2} + d\frac{\partial u}{\partial t} - \nabla \cdot (c\nabla u) + au = f$$

After the geometry of the domain has been specified, and a mesh has been generated, the function `assembleFEMatrices` allows to compute (*weak* formulation) the mass matrix $M \in \mathbb{M}^n_{\mathbb{R}}$, the stiffness matrix $K \in \mathbb{M}^n_{\mathbb{R}}$, the internal matrix $A \in \mathbb{M}^n_{\mathbb{R}}$, and the load vector $F \in \mathbb{R}^n$. At the end of the research training activity it is not clear whether MATLAB's PDEs toolbox lets the user choose the basis functions used as projection for the FEM method.

The heat equation can be obtained by considering $T \sim u$ and setting

$$\rho c \partial_t T = -\lambda \nabla^2 T \qquad \{m, a, f\} \leftarrow 0, \quad d = \rho c, \quad c = -\lambda \tag{2.2}$$

where ρ (material density), c (specific heat capacitance), and λ (specific conductivity) a constant parameters with respect to space and time. This would result in a null internal matrix $A = 0$ and null load vector $F = 0$.

Two kinds of boundary conditions can be set:

- Dirichlet boundary conditions on $\bar{D}$ impose $hu = r$. This type of BC is assembled into matrix $H \in \mathbb{M}^n_{\mathbb{R}}$ and vector $R \in \mathbb{R}^n$ by `assembleFEMatrices`.
- Neumann/Robin boundary conditions on ∂D (having normal direction $\vec{n}$) impose $\vec{n} \cdot (c\nabla u) + qu = g$. These are collected into matrix $Q \in \mathbb{M}^n_{\mathbb{R}}$ and vector $G \in \mathbb{R}^n$ by `assembleFEMatrices`.

Imposing a gradient temperature on the boundary of the domain can be achieved by setting $q = 1$ and $g = 1$ where 1 is chosen instead of T_{ref} (constant, external temperature) because we only require the structure. This would assemble matrices Q so that to have a 1 if the node is in the boundary and vector G having component 1 in the corresponding node equation.

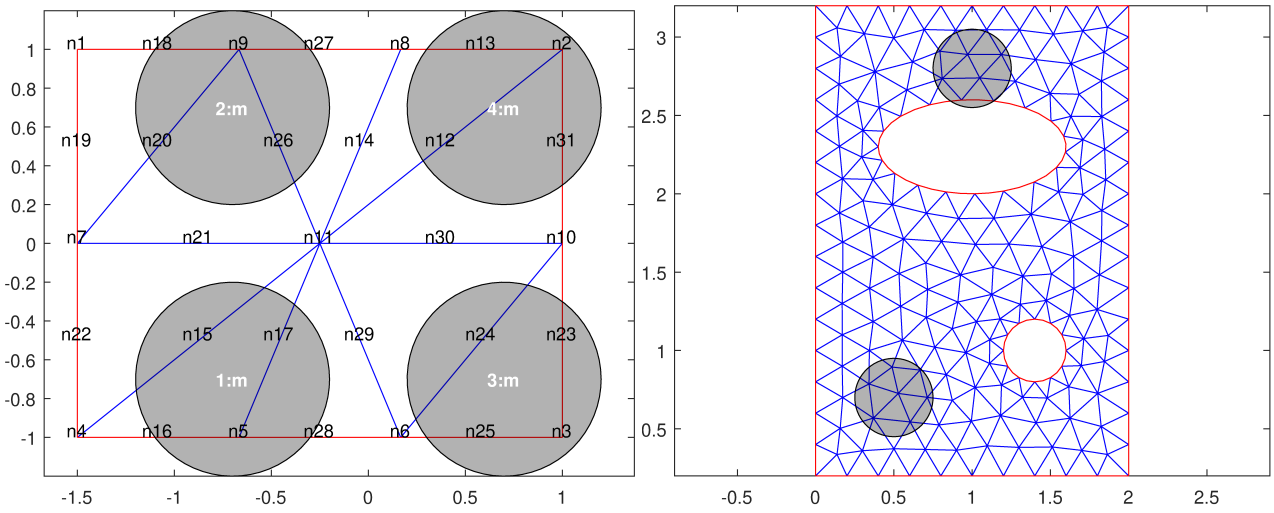
If Dirichlet conditions are imposed matrices H and R are not null and a suitable transformation can be found that eliminates the Dirichlet nodes from state dynamics. The FEM model in descriptor's form can then be obtained in the following way

$$\underbrace{M}_{E}\dot{x} = \underbrace{-\left[K + Q\right]}_{A} x + \underbrace{\begin{bmatrix} G & ATX_0 & AU_d & -EU_d \end{bmatrix}}_{B} \begin{pmatrix} T_{\text{Tref}} \\ u_0 \\ T_d \\ \dot{T}_d \end{pmatrix} \tag{2.3}$$

where the nodes temperature has been denoted with x. Matrices T, and U_d are suitable transformations that allow the removal of Dirichlet nodes from the state dynamics.

This workflow is implemented in ALG. 2 and ALG. 3 and is not described any further because it is only used in chapter 4, as the RFX model handled in chapter 5 will not have Dirichlet boundary conditions

FIG. 2.1 shows two possible 2D models obtained with the procedure described above and execution of ALG. 2 on two different mesh geometry. The former rectangular shape is retained for study in SEC. 4.4



(a) simple rectangle

(b) more complex geometry

FIGURE 2.1: Two possible full-order thermodynamic models based on 2D mesh (a) a simple rectangular domain meshed with a few elements: Nodes are printed on the image and prefixed with the letter n. Edges that form the boundary of the rectangular domain are displayed in red, while the color blue is used for edges in the boundary of mesh elements. The four shaded gray circles represent selection latches used in the construction for the output equations. Nodes falling inside each circle have been averaged to computed the corresponding output signal. Each output signal has been labelled in white color and prefixed with a letter m or v depending on its measured or virtual classification (respectively). (b) shows an example of a more complex geometry (a pierced plate) and a considerably finer mesh. The same color conventions apply. The second, smaller, hole in the lower right side if the plate can be used to model Dirichlet boundary conditions.

2.3 Analysis of thermodynamic model based on RFX machine

The *Reversed Field eXperiment* (RFX) machine has the objective of studying plasma physics and magnetic confinement using a configuration called *reversed field pinch* (RFP). The efforts of this design have the goal of testing whether RFP configuration performs as well as the *tokamak* configuration. These studies will be the foundations of future nuclear fusion reactors to come [3].

The pre-heating problem on the RFX machine is typical of torus-shaped, high-energy devices which need to be heated. In this work, a simplified model, representing the thermodynamic behaviour of one quarter of the torus, will be studied. In particular, the thermal model will be constructed with FEM techniques, in a similar fashion to what was presented in SEC. 2.1. The peculiarities of the physical modelling, however, will be overlooked in the presentation, and only *a posteriori* analysis will be performed (before control activities).

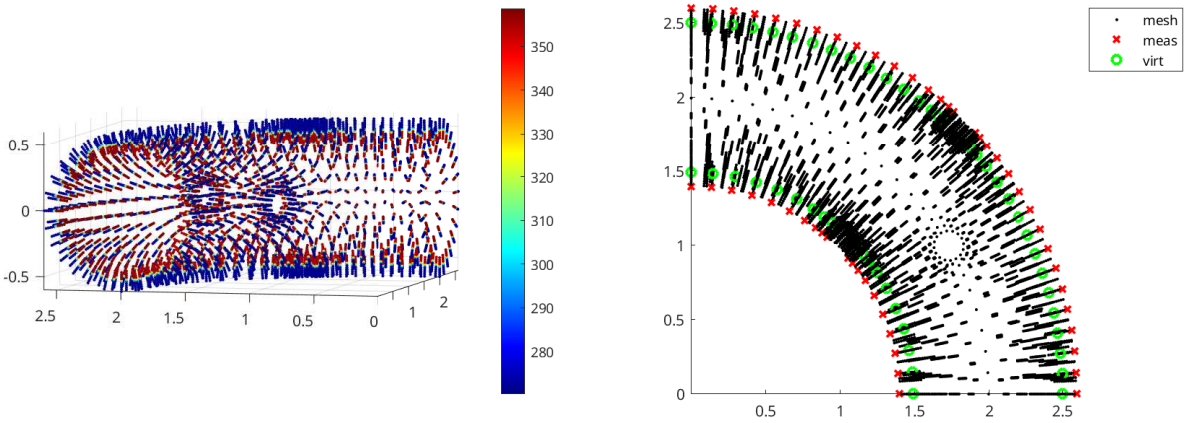
The meaning of the state space configuration can be summarised as follows. The state x of the full-order system represents the temperature in each node of the finite-element mesh. The will be two natural physical inputs: the external ambient temperature T_{ref}, and the direct heat injection u measured in Watts. The former is considered as noise input, while the latter is the true control input. No Dirichlet-type boundary conditions will be used for this simplified model, and another noise input will be added later, in order to allow for generic initial condition (see SEC. 4.1.1). The output equation remains to be defined after the physical model, and such definition will be presented later.

The overall request from the specialists of the FEM design department, is to find - if possible - a good algorithm for predicting the temperature output selected nodes that cannot be measured with a probe. Temperature sensors are placed elsewhere (depending on the geometry of the machine) and their output can be used to improve the prediction. While the ambient temperature is easily measurable, the control input u corresponding to direct input injection, is representative of a much more complex scenario deriving from electrical current flows. These particular aspects and modelling choices will not be discussed in the present document.

2.3.1 Geometry and problem definition

Sensor placement for both measurement and virtual signals is described in FIG. 2.2. The specific case for RFX, is given as a set of possible locations for measurement instruments, and for locations whose temperature should be predicted starting from measurements placed elsewhere. In particular, measurement locations are positioned on the outer shell of the torus, while desired virtual prediction should be made nearly inside the machine.

A static solution can be obtained by setting time derivatives in eq. (2.1) to zero. A heatmap of the resulting temperature distribution is shown in FIG. 2.2 where a noticeable temperature gradient can be seen. This information could be used for setting up a more advanced initial condition depending on the temperature difference between the ambient T_{ref} and the



(a) static solution $u = 1.5\,\mathrm{kW}$, $T_{\mathrm{ref}} = 30\,\mathrm{deg}$ (b) possible output locations

FIGURE 2.2: Description of RFX full-order thermodynamic model. (a) shows a heat map of a FEM static solution and gives a qualitative idea of the overall temperature gradient that the machine can withstand. (b) visualizes the possible output location used as specification for the design of the output equations. RFX mesh points are painted with black dots, measurement locations for potential temperature sensors are painted with a red cross, and virtual location for output prediction are marked with a green circle.

measured outputs. See eq. (3.1) for a detailed discussion on the implementation of the initial condition.

Selected properties of the resulting dynamic model have been summarised in TAB. 2.1: their importance will be explained in the next chapter (and in particular in FIG. 3.1) when a choice will be made with respect to possible computation paths in the model-order reduction workflow. Notice that the corresponding entries in eq. (2.1) are extremely sparse and feature condition numbers of the order of 1×10^6 which appear reasonable from a FEM point of view but are not a good starting point for control activities. Moreover, the inversion of the full-order mass matrix M loses this low-density property and makes the corresponding system in standard form, numerically cumbersome to simulate.

	M	$M^{-1}A$	$A = -(K + H)$	B	C
size	11 066 square	11 066 square	11 066 square	$11\,066 \times 2$ rect.	$(p+q) \times 11\,066$ rect.
sparse $\#_{\mathrm{nz}}/\#_{\mathrm{tot}}$	• 1.3×10^{-3}	○ 1	• 1.3×10^{-3}	○ 1	• depends
symmetric $\|X - X^T\|_2$	• 1.2×10^{-14}	○ 5.1	• 1.2×10^{-13}	○ N/A	○ N/A
condition number	5.1×10^5	1.9×10^7	2.6×10^6	1.5×10^1	depends

TABLE 2.1: Approximated numerical properties of full-order, FEM-thermodynamic model based on RFX machine with respect to its state-space representation in descriptor dynamic form $M\dot{x} = Ax + Bu$, C represents the output matrix. First the shape of these matrices is discussed, then the numerical density (sparsity) is displayed as a ratio between the number of non-zero elements $\#_{\mathrm{nz}}$ and the total number of entries in the matrix $\#_{\mathrm{tot}}$. Lastly, symmetry and condition number are estimated using MATLAB methods `normest` and `condest` (respectively).

Chapter 3

State of the art on model order reduction methods

Contents

3.1 Traditional model order reduction methods

The reader is referred to [4] for a well-structured introduction on the basic concepts of model order reduction of dynamic systems. In the following, a number reduction methods will be applied, and the balanced-truncation result will be retained and used for simulation in SEC. 5.4. Given an LTI dynamic system expressed in state space form through matrices $(A \in \mathbb{M}^n, B, C, D)$, the concept of *balancing* [4, sec. 2.2.1] relates to the task of finding a state transformation satisfying

$$X = Y = \mathrm{diag}(\sigma_1, \ldots, \sigma_n)$$

where X and Y denote the reachability and observability Gramians [5], respectively; $\{\sigma_i\}_1^n$ are called *Hankel* singular values and are conventionally numbered in decreasing size. The truncation procedure consists of choosing the reduced state-space size $r < n$ and to approximate original system with matrices $\left(\hat{A} \in \mathbb{M}^r, \hat{B}, \hat{C}, \hat{D}\right)$ obtained from discarding singular values $\sigma_{r+1}, \ldots, \sigma_n$ and therefore considering the reduced Gramians

$$\hat{X} = \hat{Y} = \mathrm{diag}(\sigma_1, \ldots, \sigma_r) \qquad r < n$$

All reduction methods aim at reducing the size of the state space by preserving the size and the dynamic input/output behaviour. The quality of the reduction may be analysed with different empirical metrics (see, for example, SEC. 3.3), and some methods may even offe analytical error bounds. Let $G(\bullet)$ and $\hat{G}(\bullet)$ denote the original and reduced system transfer function (respectively), then Balance-truncation, allow to quantify the reduction error by [4, eq. (2.19)]

$$\|G - \hat{G}\|_\infty \leq 2 \sum_{i=r+1}^{n} \sigma_i$$

Handling of nonzero (inhomogeneous) initial conditions is, in general, not trivial [4, sec. 2.2.5]. The main explanation is to be sought in the presence of an additional term in the system transfer function [6], [7] and the corresponding modification of the Gramian matrices. As a solution, a modified system is considered instead [8, sec. 2.1.8]:

$$\begin{aligned} \begin{bmatrix} E \end{bmatrix} \dot{\tilde{x}} &= \begin{bmatrix} A \end{bmatrix} \tilde{x} + \begin{bmatrix} B & AX_0 \end{bmatrix} \begin{pmatrix} u \\ u_0 \end{pmatrix} \quad \tilde{x} = x - x_0 \\ \tilde{y} &= \begin{bmatrix} C \end{bmatrix} \tilde{x} + \begin{bmatrix} 0 & CX_0 \end{bmatrix} \begin{pmatrix} u \\ u_0 \end{pmatrix} \qquad \tilde{x}(0) = 0 \end{aligned} \tag{3.1}$$

where the initial condition is modelled as $x_0 = X_0 u_0$. This choice allows a great flexibility in the description of the initial condition depending on the coordinates $u_0 \in \mathbb{R}^{n_0}$. In practice, the n_0 columns of matrix X_0 are vectors that span a subspace of dimension n_0 the state-space (node temperature) describing the initial condition. The signal u_0 allows to set different initial conditions depending on the basis chosen in X_0 For example, for spatially-constant initial temperature X_0 is chosen as a vector of all ones, and $u_0 \in \mathbb{R}$ is the desired initial

temperature for all nodes. If a more complex initial condition was to be considered (see FIG. 2.2) the mesh nodes could be divided in n_0 *zones* and the corresponding coordinate in u_0 would represent the initial temperature of each *zone*.

This procedure allows to handle non-zero initial conditions as standard homogeneous reduction methods but has at least two remarkable consequences:

- a new noise input signal u_0 appears on the input side of the state-space representation and augments by one unit the size of future observer/controller gain matrices.
- differently from other noise inputs, such as T_{ref}, the noise inputs signal u_0 cannot be directly measured, thus adding considerable sensitivity problems in the state reconstruction problem

3.2 Implementation of model order reduction methods

3.2.1 The MORLAB library

From a practical viewpoint, techniques of model order reduction are seen as black-box operations performed via the MATLAB package `MORlab` [9]. Table TAB. 3.1 summarises available methods in MORlab and highlights key references for possible future work were different reduction methods may be studied.

method	**alias**	**references**	**MATLAB**	**notes**	chosen
Bounded-Real BT	`brbt`	[4, sec. 2.3.2]	bug	dissipative (contraction)	○
Balanced Stochastic Truncation	`bst`	[4, sec. 2.3.4]		power spectrum, spectral factors	○
Balanced Truncation	`bt`	[4, sec. 2.2.3], nonzero IC [6]		fundamental	•
Frequency-Limited BT	`flbt`				○
H-infinity BT	`hinfbt`			difference with `hna` not explained	○
Krylov subspace methods	`krylov`			moment matching according to MORwiki	○
Hankel-Norm Approximation	`hna`	[10]			•
Linear-quadratic Gaussian BT	`lqgbt`	[4, sec. 2.3.3]		dissipative, also suited for unstable systems. No analytical bound on $\|G - \hat{G}\|$	•
Modal Truncation	`mt`		not implemented for sparse systems		○
Positive-Real BT	`prbt`	[4, sec. 2.3.1]		dissipative (passive)	○
Time-Limited BT	`tlbt`				○
Two-step model reduction			not implemented		○

TABLE 3.1: Routines available in MORlab for Continuous-time systems. Comments note down practical remarks that came up thinking to a broad systematic comparison of the reduction results computed with many available methods.

The *notes* column in TAB. 3.1 highlight practical considerations appeared while reading [4] as a theoretic reference for the reduction routines implemented in MORlab. The MORlab toolbox also features a dedicated online wikipedia documentation that sometimes does not correspond one-to-one with acronyms and nomenclature chosen in the code of the scripts and [4].

3.2.2 Computation of the reduced, control model

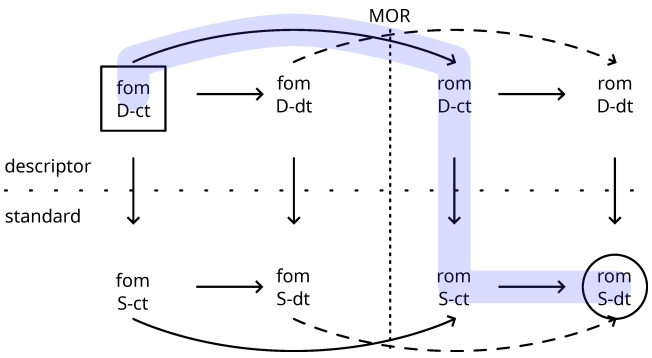


FIGURE 3.1: All possible computation paths to obtain a reduced, discrete-time, model in standard dynamic form. From top to bottom: models can be transformed from descriptor dynamic form to standard dynamic form by inversion of the mass matrix. From left to right: models can be sampled from CT to DT by means of a suitable discretisation method. Outer arrows represent model reduction steps: depending on the nature of the starting system, different families of methods could be considered. The highlighted path describes the only option considered in this thesis work.

FIG. 3.1 shows all possible computation paths that start from a continuous-time, full-order system in descriptor dynamic form to a discrete-time, reduced-order system in standard dynamic form. In the present work, two fundamental workflow choices are made:

1. all control operations are aimed at state-space systems in **standard** dynamic form (circled point in FIG. 3.1), therefore neglecting all possibilities related to a rectangular mass matrix [11, p.1] and observer design applied to state-space systems in descriptor dynamic form [12] as these methods heavily rely on algebra of matrix pencils.
2. discrete-time reduction methods have not been considered (dashed external arrows in FIG. 3.1). See [4].

The highlighted path in FIG. 3.1 shows the present computation choice: all achievements in this document are the result of some observer/controller design that starts from a dynamic system resulting from the highlighted path:

1. model order reduction of the full-order continuous-time system in descriptor dynamic form.
2. conversion of the reduced-order descriptor system to standard dynamic form
3. sampling of the continuous-time system to obtain a discrete-time, reduced-order, state-space model in standard dynamic form.

First, a reduction method is applied to the sparse, full-order system in descriptor dynamic form. On the one side, this is convenient because the alternative of transforming the system from descriptor to standard would result in a dense, quare, A matrix,of size $\approx 1 \times 10^4$ (TAB. 2.1). On the other side, sampling of the CT descriptor system to obtain DT descriptor system would have required DT model order reduction methods which were not considered in this work.

Second, inversion of mass matrix is applied to the reduced system in order to obtain a standard dynamic formulation. This poses no numerical problem as long as the condition number of the reduced mass matrix E is acceptable. Again, the discrete-time conversion of a system in descriptor dynamic form was not considered.

$$\dot{x} = (E^{-1}A)\, x + (E^{-1}B)\, u$$

Third, the reduced system in standard form can be converted from continuous-time to discrete-time with sampling time τ. The Euler-Backwards method has been used, by resorting to the following transformation [13, Table 3] (other methods could have been considered, such as the Tustin transform)

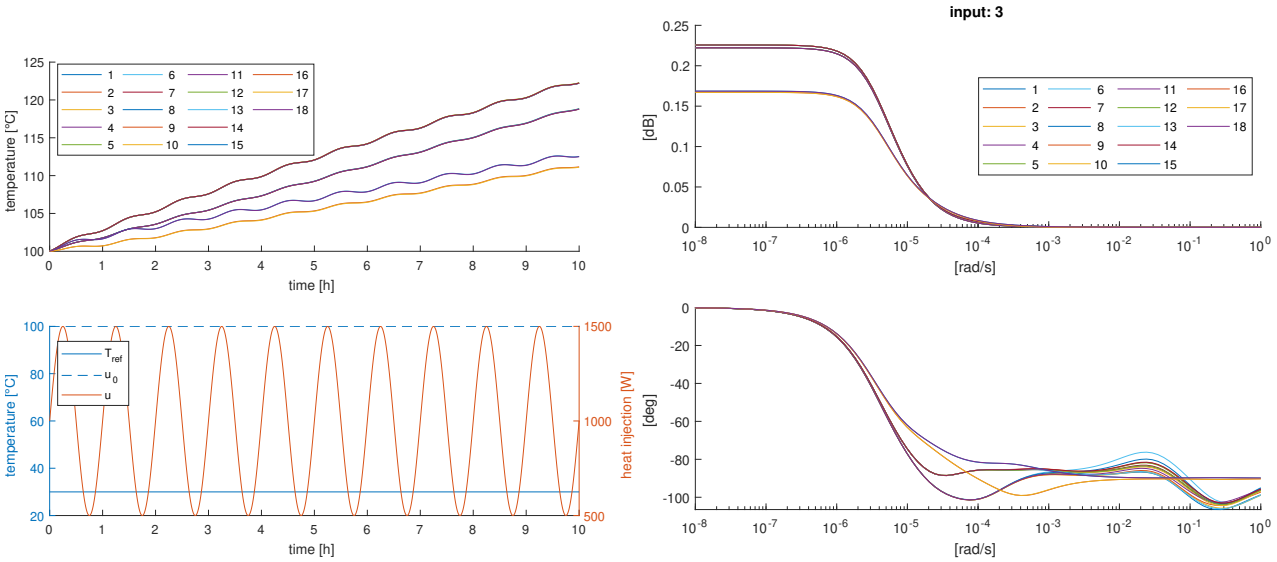
$$\begin{aligned} F &= \left[I - A\tau\right]^{-1} & G &= \left[I - A\tau\right]^{-1} B\tau \\ H &= C\left[I - A\tau\right]^{-1} & J &= D + C\left[I - A\tau\right]^{-1} B\tau \end{aligned} \tag{3.2}$$

Possibilities presented in FIG. 3.1 may be further expanded if transformation of noise input to state-space is applied. This was initially considered in the present work but later discarded for the reasons explained in SEC. 4.1.1.

3.3 Assessing the quality of the reduced, dynamic model

Once the reduced model has been obtained, there exists the problem of assessing the quality of the reduced dynamics as compared to the original system. In particular, the the system at hand is MIMO, the number of error signals and transfer functions increases considerably. For this reason, a few metrics were selected in this work in order to measure the quality of the reduced system among different reduction routines.

The proposed metric is twofold and may be expanded with different norms in future work. On the one hand the error signal of the time response is computed for all outputs (measured and virtual) and then two scores are computed: the ℓ_2 norm and the max amplitude of error signal. On the other hand, the error of Bode plots of all possible transfer functions is computed and measured with the same two scores, thus providing 4 more indexes of quality (2 for the magnitude, and 2 for the phase)

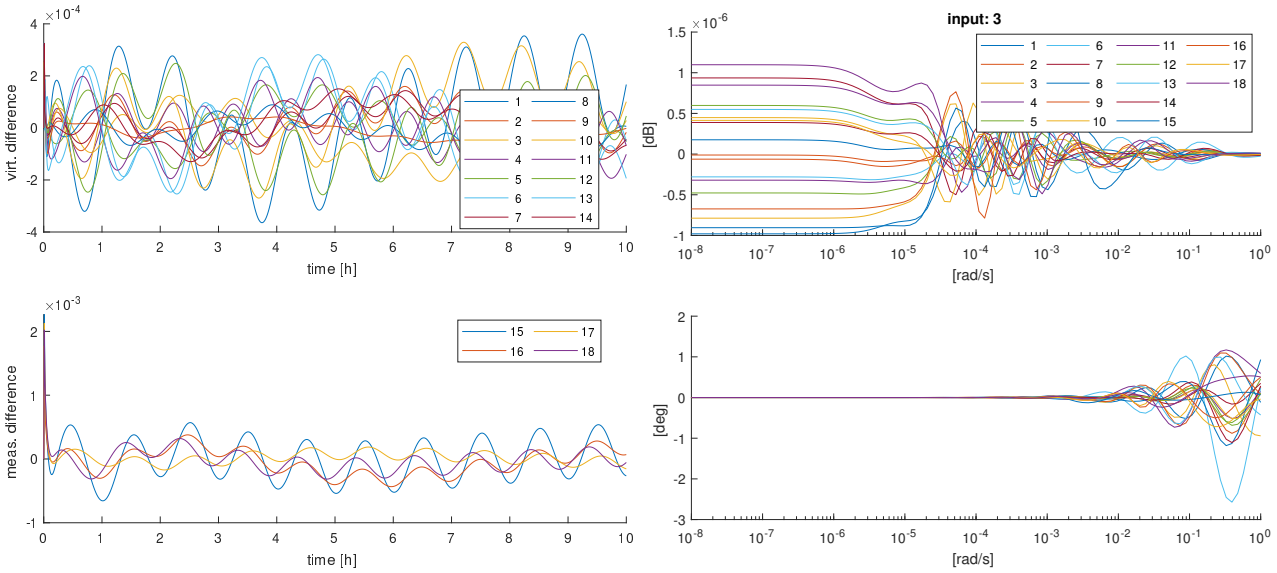


(a) step response

(b) bode plot

FIGURE 3.2: Reference input/output dynamics of full-order RFX thermodynamic model. (a) upper plot shows the simulated output of the full-order RFX system with inputs described in the lower plot. This dynamics is used to compute the qualitative error on the time response of the reduced model. (b) shows a Bode plot of the full-order model in descriptor form of all 18 transfer functions from the third input u (direct heat injection) to the outputs defined in SEC. 5.1.1

As a concrete example, reference dynamics of RFX is shown in FIG. 3.2 where the full-order system in descriptor dynamic form has been simulated with sinusoidal heat injection of about $1\,\mathrm{kW}$ for $10\,\mathrm{h}$. The Bode plots of all outputs (measured and virtual) with respect to the only control input u has been computed as well. Application of balanced-truncation method resulted in the errors shown in FIG. 3.3 where the difference between the time response and the Bode have been displayed. Application of other order reduction methods has been summarised in TAB. 3.2 where it can be seen that the balanced truncation is the only reduction method that stays withing the $\approx 1 \times 10^{-3}$ *worse* error in the time response **and** $\approx 1 \times 10^{-6}$ error in the Bode magnitude plot for all considered metrics.



(a) time response error

(b) bode plot error

FIGURE 3.3: Input/output dynamic error due to balanced-reduction of RFX continuous-time thermal model in descriptor dynamic form. (a) shows the error in time response with respect to the time response plotted in FIG. 3.2. (b) shows the error in Bode plot with respect to the plot on the right side of FIG. 3.2.

Error figures like FIG. 3.3 can be generated for any reduction method with the code provided in appendix A. Error plots qualitative information is similar to that presented for balanced truncation and are not reported here.

method (tolerance)	reduced state	metric	$y_{\text{fom}} - y_{\text{rom}}$	$\text{dB}_{\text{fom}} - \text{dB}_{\text{rom}}$	$\angle_{\text{fom}} - \angle_{\text{rom}}$
bt (1×10^{-5})	21	$\|\bullet\|_2$	5.9×10^{-3} (1)	6.7×10^{-6} dB (4)	$5.8°$ (6)
		$\max(\|\bullet\|)$	3.5×10^{-4} (1)	1.1×10^{-6} dB (4)	$1.2°$ (11)
hna (1×10^{-5})	21	$\|\bullet\|_2$	3.5×10^{4} (13)	1.1×10^{2} dB (6)	$1.8°$ (17)
		$\max(\|\bullet\|)$	1.9×10^{3} (14)	1.9×10^{1} dB (6)	$1.9°$ (1)
lqgbt (1×10^{-5})	20	$\|\bullet\|_2$	1.1×10^{-2} (13)	8.8×10^{-6} dB (1)	$5.1°$ (1)
		$\max(\|\bullet\|)$	1.9×10^{3} (14)	1.9×10^{1} dB (6)	$1.9°$ (1)

TABLE 3.2: Reduction comparison based on error analysis for different methods applied to RFX full-order model. Considered method: balanced truncation (bt), Hankel-norm approximation (hna), and linear quadratic Gaussian balanced truncation (lqgbt). Error metrics are applied to all output signals (measured and virtual) and the worst error is reported. Numbers within round brackets indicate the output signal index where the worse error was observed.

An interesting fact that is captured by neither FIG. 3.3 nor by TAB. 3.2 is the condition number of the reduced-order mass matrix. As described in FIG. 3.1, the next step after the reduction of the system's order, is the transformation of the dynamic formulation from descriptor to standard, and in this operation, the condition number of the mass matrix is not to be neglected. For example, it was found that the condition number of the balance-truncation mass matrix was always 1 while the one from Hankel-norm approximation was several orders of magnitude bigger. This was another factor adding to the choice of balanced-truncation over

other reduction methods.

Chapter 4

State reconstruction for virtual output prediction of full-order thermodynamic models based on 2D mesh

Contents

This chapter describes the design and simulation of two traditional observers on a simple full-order thermal model based on 2D geometry. The first section describes the problems encountered in handling this model while the next two sections recall essential formulation of full-state and reduced-state, feedback, asymptotic, observer design. The last section contains all design reports and simulation results, and it is longer than the preceding sections due to the presence of large amount of tables and figures. In particular, simulation figures presented at the end of this chapter show the behaviour of both traditional observers (two observers) in two different input working conditions (nominal and noisy) for a total of four figures.

4.1 Issues in the reconstruction problem of finite-element models

The aim of the present work is to solve the reconstruction problem in order to make good predictions on non-measured output locations (virtual). In other words, given the input signals $u(\bullet)$ and the measured output signals $y_m(\bullet)$, we seek an estimator that is able to compute a state estimate $\hat{x}(\bullet)$ that provides good predictions on the virtual outputs $y_v(\bullet)$.

4.1.1 Handling of modelled, noise, input signals

Following the assembly of the dynamic state-space matrices from FEM some of the signals appearing on the input side of the state-space formulation, do not correspond to control inputs, but rather to modelled noise inputs. For this reason, for a system where the original input contribution can be partitioned as $\underbrace{\begin{bmatrix} B_v & B_u \end{bmatrix}}_{B} \begin{pmatrix} v \\ u \end{pmatrix}$ with v and u representing noise and control input respectively, v can be moved to state with a modification of the state equation ($\tilde{x}$ has been used as a reminder of eq. (3.1))

- if the system is continuous-time then place a zero block corresponding to the dynamics of the noise input
$$\begin{bmatrix} E & 0 \\ 0 & I \end{bmatrix} \begin{pmatrix} \dot{\tilde{x}} \\ \dot{v} \end{pmatrix} = \begin{bmatrix} A & B_v \\ 0 & 0 \end{bmatrix} \begin{pmatrix} x \\ v \end{pmatrix} + \begin{bmatrix} B_u \end{bmatrix} u$$
- if the system is discrete-time, then place an identity block corresponding to the dynamics of v
$$\begin{bmatrix} E & 0 \\ 0 & I \end{bmatrix} \begin{pmatrix} \tilde{x}_{i+1} \\ v_{i+1} \end{pmatrix} = \begin{bmatrix} A & B_v \\ 0 & I \end{bmatrix} \begin{pmatrix} x_i \\ v_i \end{pmatrix} + \begin{bmatrix} B_u \end{bmatrix} u_i$$

This operation, however, seems to make the observer design more difficult and, in general, provides worse performance as compared to keeping the modelled, noise, input signals on

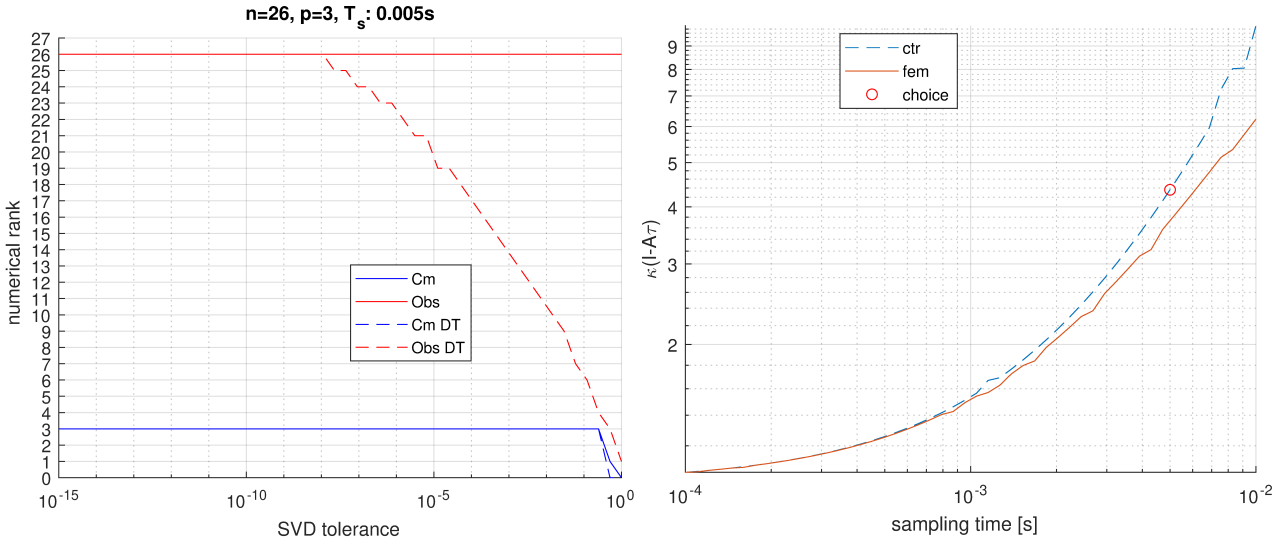
the input side of the state-space representation. This behaviour, could not be explained from a theoretical point of view, but a number of remarks are anticipated below:

- traditional observer design (SEC. 4.2 and SEC. 4.3) do not make use of input matrix B in the allocation procedure. Yet, the resulting observer dynamic state-space system uses an expanded version of B, and indeed, a controllability analysis could be performed on the observer system alone (see, for example, FIG. 4.4)
- more advanced observer design methods, such as H_2-optimal (SEC. 5.2) and H_∞ sub-optimal (SEC. 5.3), require assumptions on the controllability of the pair (A, B) on the unstable boundary of the complex plane, but allow to perform a similar operation when formatting the system as strictly causal (eq. (5.1))

4.1.2 Pole placement and bad numerical conditioning

FIG. 4.1 reports the dependency of the observability property with respect to the numerical precision used in MATLAB by the method `rank`. This gives an idea of why the method `place` may fail to allocate observer poles or generate high-gain entries in the observer gain that lead to considerable peaking phenomena in the prediction error transient. FIG. 4.1 also reports the conditioning the map $(I - A\tau)$ that needs to be inverted in order to sample the continuous time system with Euler Backwards method.

FIG. 4.2 (inspired by the representation of the four fundamental linear algebra subspaces presented in [14]) represent the illustrates the relation in high-dimensional spaces between the observable subspace, the non-observable subspace, the span of the virtual output matrix, and the associated null space. The subspaces hosted in the state-space form a number of angles (depending on the relative dimension of the subspaces) that are usually referred to as *angles between flats*. In practice, if the SVD basis vectors of the observability and virtual output matrices are available, the the projection cosines between the bases can be used to compute the angles between the observability flat and the virtual output flat. It seems that subspace angles between SVD basis may be used to assess the difficulty of the state reconstruction problem for virtual output prediction and it relation to the bad numerical conditioning of the observability matrix. For example, if a given system is not observable, but the virtual output matrix requires a state subspace that is very close - in terms of angles between flats - to the observable subspace, then it may be that the reconstruction problem for the *particular* set virtual outputs at hand, is still well posed.



(a) numerical observability (b) sampling conditioning

FIGURE 4.1: Numerical observability and discrete-time sampling condition number of full-order 2D model. The map $\tau \mapsto (I - A\tau)$ is required by the discrete time transformation of the original system formulation via Euler-Backwards eq. (3.2) Quadrant (a) reports the dependency of the observability matrix and of the (measured) output matrix from the numerical precision used in computing the their. A dashed line is used to distinguish the discrete-time system from the continuous-time counterpart. Quadrant (b) instead, reports the condition number of the map $\tau \mapsto (I - A\tau)$ whose inversion is required for sampling the CT system (see eq. (3.2)) The dashed line refers to transformation of the system where the non control inputs have been moved to system state (see SEC. 4.1.1).

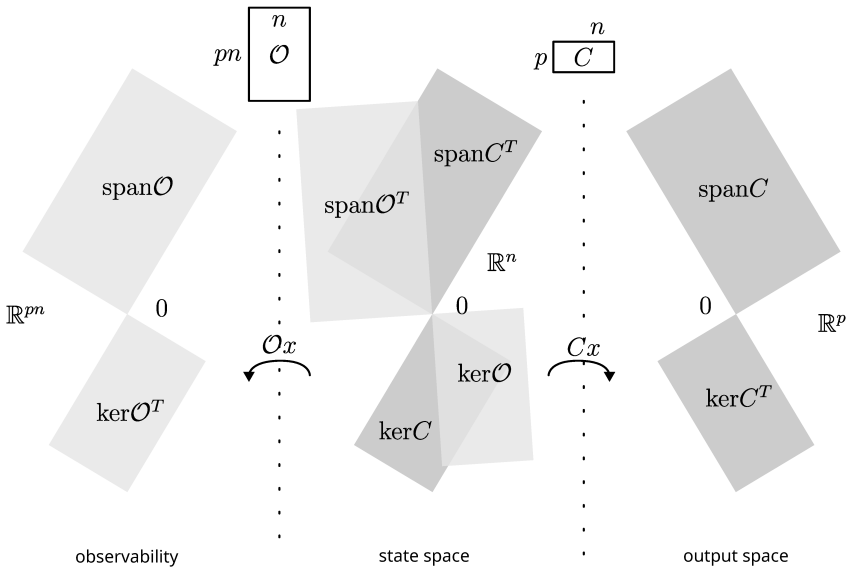


FIGURE 4.2: Observability problem and subspace angles in high dimensions. The observability matrix is obtained from the output equation associated to the measured outputs while the matrix C appearing on the darker shapes is referring to the virtual output equation. The observability subspaces (lighter rectangles) and the virtual output subspace in the state space (middle sector of the picture) form an angle in high dimension. If the system is observable $\ker \mathcal{O} = \{0\}$ whereas is the the virtual output matrix is full-rank, then $\ker C = \{0\}$ as well.

Clearly, these problems are independent of the full or reduced nature of the order of the system to be analysed. In this study, however, no remarkable result was achieved in this direc-

tion, and further exploration of these concepts should be considered in the future. In MATLAB , the angle between flats can be computed using the `subspace` function [15, p. 2014] which however uses row scaling in order to get an accurate estimate [16, p. 179]. Failure by MATLAB `place` function to allocate observer pole may be explained as a limit of high-gain and peaking phenomenon [17]. The state observation problem can be studied by resorting to Principal Component Analysis to quantify structural instability [18]. For example, the concept of *Weak* observability, as defined in [19] refers to measures of *un-observability* [20] using a local observability Gramian.

Further, if the bad numerical conditioning of the observability matrix was to be justified as fundamental cause of the high-gain/peaking phenomenon, a preconditioning transformation could be applied to the state dynamics in order to minimise the resulting condition number. The minimisation of the conditioning number of a matrix, through preconditioning can be formulated as standard optimisation problem [21] and solved fairly efficiently.

4.2 Design of full-state, feedback, asymptotic observers

The reader is pointed towards [22], [23] for standard literature on asymptotic, feedback full-state observers. Here, the term *full-state* is used to distinguish this design process from *reduced-state* observers (SEC. 4.3) and the idea of *full/reduced-order* connected to the reduction operations of the origina dynamic system.

Given a state space system (A, B) of dimension n with m inputs and a first, measured, output equation $y_m = C_m x + D_m u$, the key idea behind the full-state observer design is to build a state space dynamic system

$$\frac{\mathrm{d}}{\mathrm{d}t}\hat{x} = A\hat{x} + Bu + L(y - C_m\hat{x} - D_m u)$$

and to find a gain matrix L that stabilises the estimation error dynamics

$$\begin{aligned}\frac{\mathrm{d}}{\mathrm{d}t}(x - \hat{x}) &= Ax + Bu - A\hat{x} - Bu - L(C_m x + D_m u - C_m\hat{x} - D_m u)\\ &= Ax - A\hat{x} - L(C_m x - C_m\hat{x})\\ &= A(x - \hat{x}) - LC_m(x - \hat{x}) = \left[A - LC_m\right](x - \hat{x})\end{aligned}$$

After computing a stabilising gain L, the final observer structure with prediction on the virtual outputs y_v can be computed using the second, virtual, output equation:

$$\begin{aligned}\frac{\mathrm{d}}{\mathrm{d}t}\hat{x} &= \underbrace{\left[A + LC_m\right]}_{\text{obs.A}} x + \underbrace{\left[B + LD_m\right]}_{\text{obs.B}_\text{u}} u + \underbrace{-L}_{\text{obs.B}_\text{y}} y_m\\ \hat{y}_v &= \underbrace{C_v}_{\text{obs.C}} x + \underbrace{D_v}_{\text{obs.D}_\text{u}} u + \underbrace{0}_{\text{obs.D}_\text{y}} y_m\end{aligned} \tag{4.1}$$

A variation of this formulas was developed at the beginning of this work, allowing the initial condition coordinate signal u_0 to be handled separately, and fed analytically into the observer. The reason behind this necessity was explained in SEC. 4.1.1 and was later considered unnecessary. Such handling of u_0 will, however, become a requirement again in the future when the initial condition observation will be solved, as this signal will not be computed by the same block that reconstructs the state of the system. In practice this can be achieved fairly easily by moving u_0 to state dynamics and then introducing a partition of the state where certain coordinates are not estimated by the observer and are fed analytically. The details on such transformation are not presented here but older scripts are available in the code repository linked in appendix A.

4.3 Design of reduced-state, feedback, asymptotic observers

For traditional formulation of reduced-state observers, the reader is pointed towards [22]. The idea behind this observers is to use the output signal map $C \in \mathbb{M}^{p\times n}$ to reduce the dimension of the state-space from n to $n-p$. The user chooses a basis of row vectors V in order to make the state transformation $T^{-1} = \begin{bmatrix} V \\ C \end{bmatrix}$ full rank. The observer gain $L \in \mathbb{M}^{(n-p)\times p}$ is computed on the reduced vector $v = w + Ly$ where $z = T^{-1}x = \begin{pmatrix} w \\ y \end{pmatrix}$ denotes the transformed state space.

This work considers two different output equations: one for the p measured outputs having matrix C_m, and another for the q virtual outputs to be predicted having matrix C_v . The matrix V completing the change of basis matrix T has been chosen by using the basis V_v^T provided by SVD computation of the virtual output matrix $C_v = U_v \Sigma_v V_v^T$, eventually completed by additional vectors X to get a full rank matrix with C_m

$$T^{-1} = \begin{bmatrix} V \\ C_m \end{bmatrix} \qquad \mathbb{M}^{(n-p)\times n} \ni V = \begin{bmatrix} X \\ V_v^T \end{bmatrix} \tag{4.2}$$

Homogeneous formulation can be obtained from the transformation of the original dynamic pair (F, G) by T to (A, B) and then considering the partitioning associated to the transformed vector $z = \begin{pmatrix} w \\ y \end{pmatrix}$ with $w \in \mathbb{R}^{n-p}$ and $y \in \mathbb{R}^p$. ALG. 7 implements the following

$$\dot{v} = \left[A_{[11]} - LA_{[21]}\right] v + \underbrace{\left[A_{[12]} + LA_{[22]} - A_{[11]}L - LA_{[21]}L\right]}_{B_y} y + \underbrace{\left[B_{[1]} + LB_{[2]}\right]}_{B_u} u$$

After computing a stabilising gain L, the final observer structure with prediction on the virtual outputs y_v can be computed using the virtual output equation by substituting $y = y_m - Du$ with y_m denoting the measured output of the non-homogeneous system:

$$\begin{aligned} \dot{v} &= \underbrace{\left[A_{[11]} - LA_{[21]}\right]}_{\text{obs.A}} v + \underbrace{\left[B_u - B_y D_m\right]}_{\text{obs.B}_\text{u}} u + \underbrace{B_y}_{\text{obs.B}_\text{y}} y_m \\ \hat{y}_v &= \underbrace{C_v T Z_v}_{\text{obs.C}} v + \underbrace{\left[D_v - C_v T (Z_y - Z_v L) D_m\right]}_{\text{obs.D}_\text{u}} u + \underbrace{C_v T (Z_y - Z_v L)}_{\text{obs.D}_\text{y}} y_m \end{aligned} \tag{4.3}$$

where (Z_v, Z_y) is a partition of the identity matrix allowing to write $z = Z_v w + Z_y y$.

4.4 Simulation of simple, finite-element, full-order, dynamic models

4.4.1 System description

So far, the working model consisted of a series of transformations that led from a continuous-time full-order model in descriptor dynamic form (eq. (2.1) with the addition of output equation)

$$\begin{aligned} \underbrace{M}_{E}\dot{x} &= \underbrace{-(K+H)}_{A}x \quad + \underbrace{\begin{bmatrix} q_{\text{conv}} & q \end{bmatrix}}_{B}\begin{pmatrix} T_{\text{ref}} \\ u \end{pmatrix} \\ \begin{pmatrix} y_v \\ y_m \end{pmatrix} &= \begin{bmatrix} C_v \\ C_m \end{bmatrix} x \qquad\qquad\qquad +D\begin{pmatrix} T_{\text{ref}} \\ u \end{pmatrix} \end{aligned}$$

to a discrete-time full-order model in standard dynamic form (eq. (3.2))

$$\begin{aligned} x_{i+1} &= Fx_i + G\begin{pmatrix} T_{\text{ref}} \\ u \end{pmatrix}_i \\ \begin{pmatrix} y_v \\ y_m \end{pmatrix}_i &= \begin{bmatrix} H_v \\ H_m \end{bmatrix} x_i \qquad\qquad +J\begin{pmatrix} T_{\text{ref}} \\ u \end{pmatrix}_i \end{aligned}$$

Differently from the overall procedure described in SEC. 3.2.2, for models based on 2D geometry, no order reduction has been performed, with the intent of studying the difficulties related to a big state space. FIG. 4.3 illustrates the simulation diagram for testing the observer design. There are four dynamic blocks: one block contains the full-order model in descriptor dynamic form; two blocks represent the CT and DT full-order models in standard form, and one more block is used to plug in the observer design to be tested.

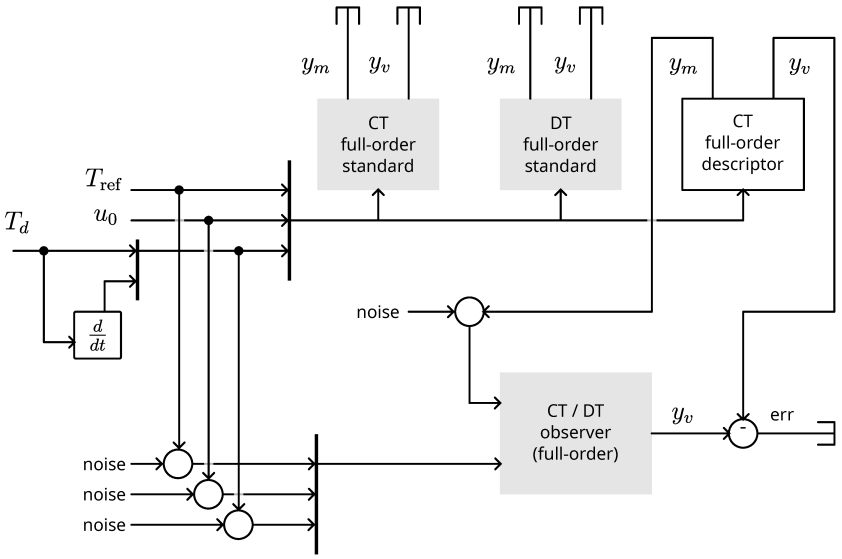


FIGURE 4.3: Block scheme for simulation of full-order models based on 2D mesh. Clean inputs $(T_{\text{ref}}, u_0, T_d)$ are fed directly to reference models. Noise can be added as the signals are fed to the observer. All models are in standard dynamic form except for the upper right full-order model representing the physical reference (descriptor dynamic form).

4.4.2 Design logs

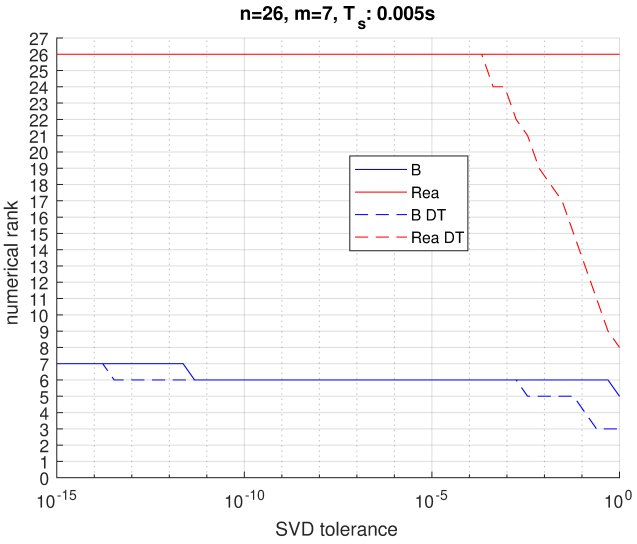
The application of ALG. 4 and ALG. 6 corresponding to the full-state and reduced-space design methods described above, resulted in FIG. 4.4 and FIG. 4.5. It can be seen that, the resulting observer is controllable in both continuous-time and discrete time but that the poles of the original dynamics could not be moved much. This difficulty in moving the original poles is summarised numerically in TAB. 4.1. The design of the observers starts with a specification for the pole allocation and finishes with a gain matrix that allows to change the poles of the error associated to the observer dynamics. This change may be easy, hard, or even impossible depending on the observability of the system, and this difficulty may be assessed by looking at the magnitude of the components in the resulting gain matrix. In TAB. 4.1 it can be seen that, for relatively slow error dynamics (first line), the magnitudes of the continuous-time design are more bigger than the discrete-time ones, only if the design is done in full-state (first column). Indeed, the opposite is observed for the reduced-state design (second column).

A further study is presented in appendix B.1 where observability of the discrete-time system is analysed as the size of the full-order state space increases considerably. The plots show that the bigger the state space, the more difficult it becomes to observe the system, to the point where, even at machine precision, the DT system results as un-observable while the CT system is observable

		full-state feedback observer	reduced-state feedback observer
eig.Val result	min	-2×10^{-2}	-2×10^{-2}
	max	-1×10^{-1}	-1×10^{-1}
CT high-gain	min	-2×10^{4}	-9×10^{2}
	max	3×10^{4}	8×10^{2}
DT high-gain	min	-2×10^{2}	-2×10^{3}
	max	3×10^{2}	1×10^{3}

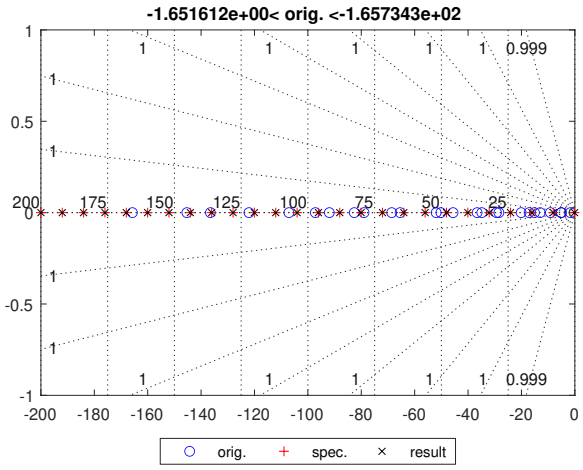
TABLE 4.1: Design results comparison between observers based on full-order 2D model. Three aspects are analyzed: the resulting eigenvalue placement, the bounds of the entries in gain matrix from the design based on the CT system, and the gain bounds resulting from the DT design.

These difficulties in the design of traditional observers seem to suggest that only certain poles should be moved through feedback and the remainder of the poles could be left untouched. This could be the objective of future improvements where a hybrid combination of feedback and open-loop observer is designed. In such case, the choice of the relevant poles to move is not trivial, for a pole may be important for the prediction of the virtual outputs but be very hard to observe due to the geometry of the problem. For this reason, analysis of the angles between the vectors of the state space basis provided by SVD of C_m, C_v and of the observability matrix, may lead to interesting results.

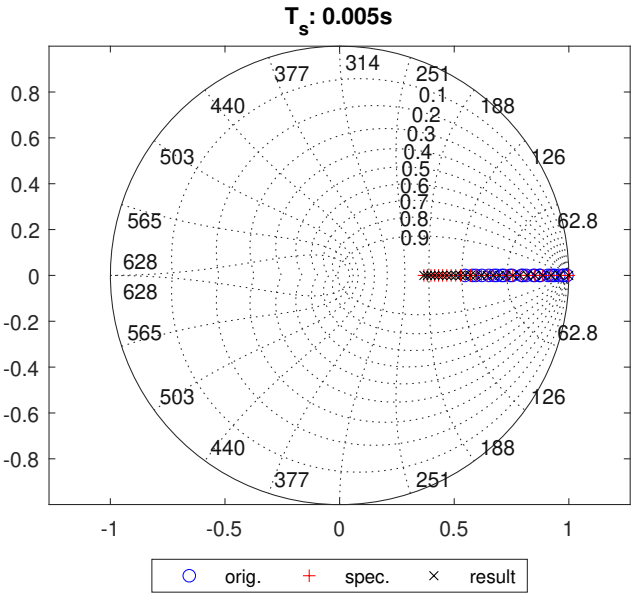


(a) observer num. controllability

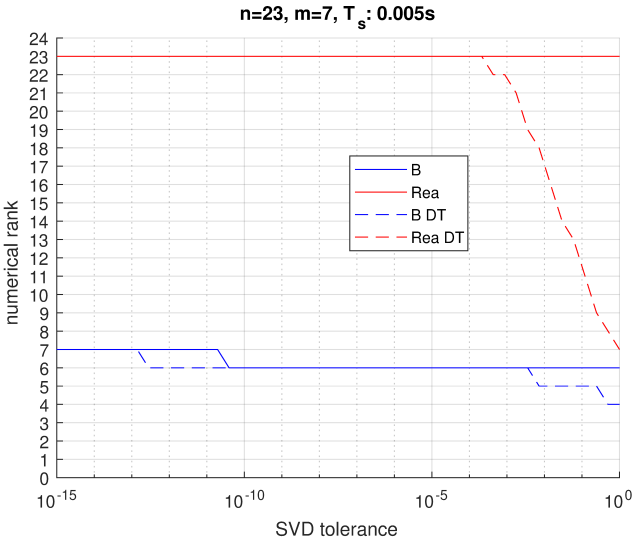
FIGURE 4.4: Design plots of continuous-time and discrete-time full-state observer based on 2D full-order model (a) shows the rank of the controllability matrix and of the input matrix for the CT and DT observer with respect to numerical precision used by MATLAB `rank` function. (b) depicts the CT placement of original, specification, and resulting poles after allocation with MATLAB `place` method. (c) depicts the DT placement of original, specification, and resulting poles after allocation with MATLAB `place` method.



(b) continuous-time poles

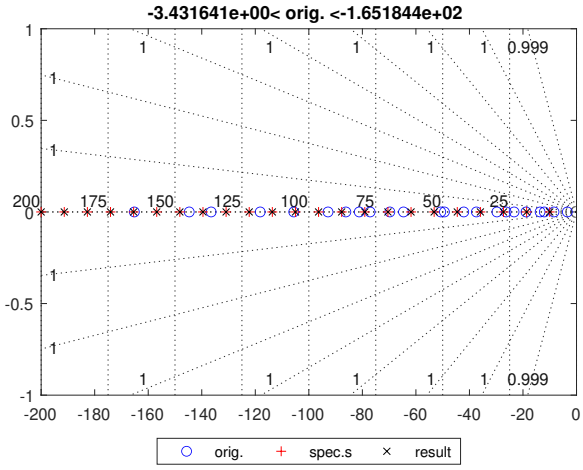


(c) discrete-time poles

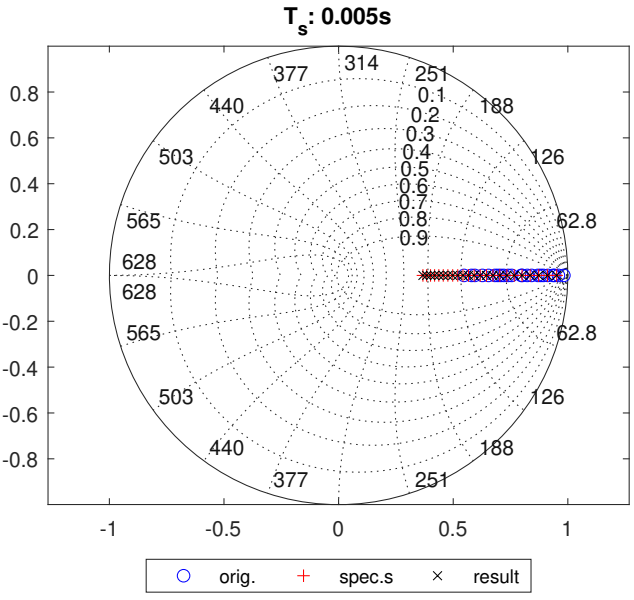


(a) observer num. controllability

FIGURE 4.5: Design plots of continuous-time and discrete-time reduced-state observer based on 2D full-order model. (a) shows the rank of the controllability matrix and of the input matrix for the CT and DT observer with respect to numerical precision used by MATLAB `rank` function. (b) depicts the CT placement of original, specification, and resulting poles after allocation with MATLAB `place` method. (c) depicts the DT placement of original, specification, and resulting poles after allocation with MATLAB `place` method.



(b) continuous-time poles



(c) discrete-time poles

4.4.3 Results on transient prediction of virtual outputs

Simulation results from FIG. 4.6 to FIG. 4.9 are obtained by computing the block scheme presented in FIG. 4.3 with full-state and reduced-state observers in nominal and noisy conditions. Nominal conditions do not inject any noise in the points defined in the block scheme. Noisy conditions, on the other hand, add white noise to T_{ref} and measurement outputs fed into the observer block before prediction.

As will be for SEC. 5.4, the structure of these figures is made of six quadrants each depicting a different type of signal: the upper two represent noise inputs and control inputs, the two quadrants at the center describe the measured outputs and their configuration with respect to the FEM mesh and virtual output. The latter are shown in the bottom two quadrants where the error with respect to the true model (simulation of full-order model in descriptor form) has been plot.

On the one hand, nominal simulations in FIG. 4.6 and FIG. 4.8 seem to perform reasonably well despite the difficulties encountered during the design process. The initial peaking due to high-gain entries in L vanishes considerably faster that the overall dynamics.

On the other hand, noisy simulations in FIG. 4.7 and FIG. 4.9 adding white noise to both the ambient temperature T_{ref} and all measured outputs, reveal a bad prediction performance where the injected noise is amplified of at least two order of magnitudes.

Overall, the transient behaviour of traditional observers on full-order the rectangular full-order model is summarised in TAB. 4.2.

		full-state observer nominal	observer noisy y_m	reduced-state observer nominal	observer noisy y_m
min peak	°C	-44	-4×10^2	-123	-3×10^3
max peak	°C	-11	3×10^2	23	4×10^3
peak time	s	5×10^{-3}	4.6	5×10^{-3}	5
err. Tol.	°C	1	10	1	50
asymptotic error	°C	-0.2	1	<0.1	-7.6
error variance	°C	<0.1	≈ 0	<0.1	≈ 0

TABLE 4.2: Transient characteristic comparison between observers based on full-order 2D model

4.4.4 Empirical robustness findings

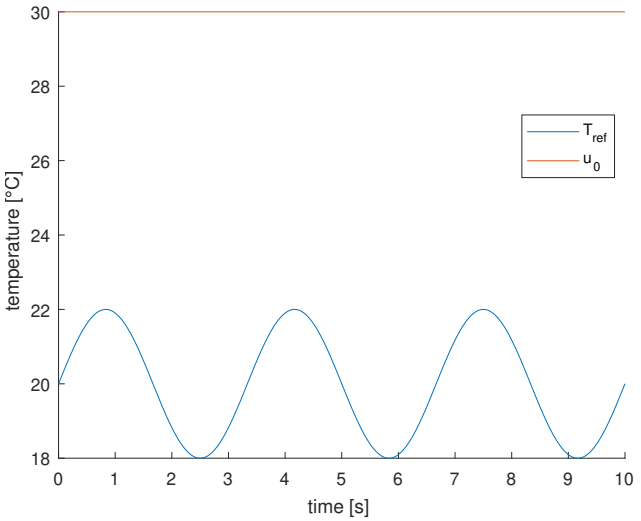
TAB. 4.3 summarises the findings on noisy measurements performed on a number of different simulation conditions. Indeed, from FIG. 4.3 noise could be injected in different point of the block scheme, and the noise could have different characteristics. In particular, two representative scenarios where tested: white noise addition and bias noise addition. These procedures were later applied to RFX simulation scheme and TAB. 5.3 was produced.

	noise type	full-state feedback observer	reduced-state feedback observer
T_{ref}	white	○	○
	bias	○	○
u_0	bias	○	○
y_{meas}	white	bad	bad
	bias	○	○

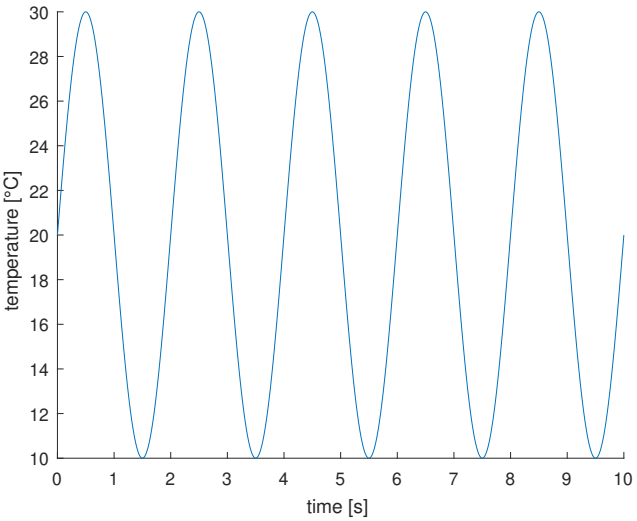
TABLE 4.3: Empiric, robustness, results comparison between observers based on full-order 2D model

A number of noisy simulations could be performed used the code provided in appendix A following the testing that has been done for RFX in appendix B. For example:

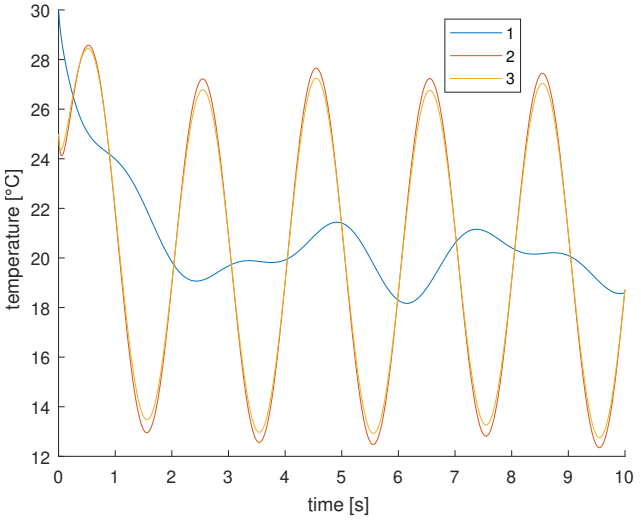
- noise can be injected in the ambient temperature signal T_{ref}. This has been done (white noise addition) in the noisy simulations of this section, and will be repeated for reduced-order RFX model in SEC. 5.4. Addition of bias noise to T_{ref} is explored in FIG. B.3.
- bias noise can be added to the measured output signal y_m. Addition of white noise has been done in this section and will be repeated for RFX reduced model as well. Effect of bias noise on the measured outputs could be seen as a modelling error with respect to reality, and is explored in FIG. B.4.
- noise can be added to the initial condition coordinate u_0. This is yet to be studied as u_0 is the only non-measured, modelled, noise input of the system, and required a separate treatment that has been reserved for future activity.



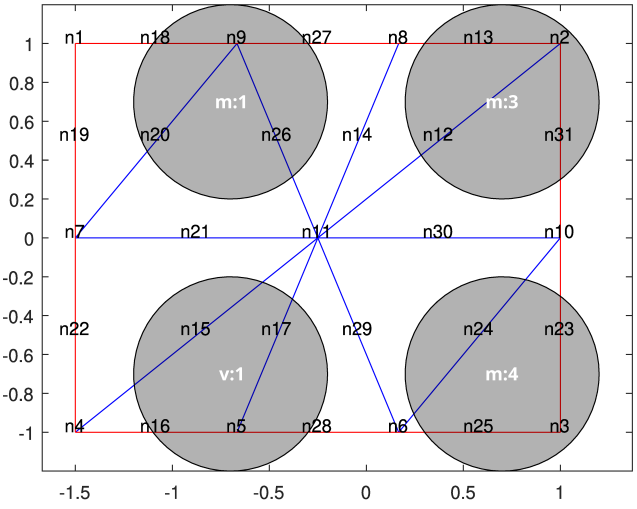
(a) noise input



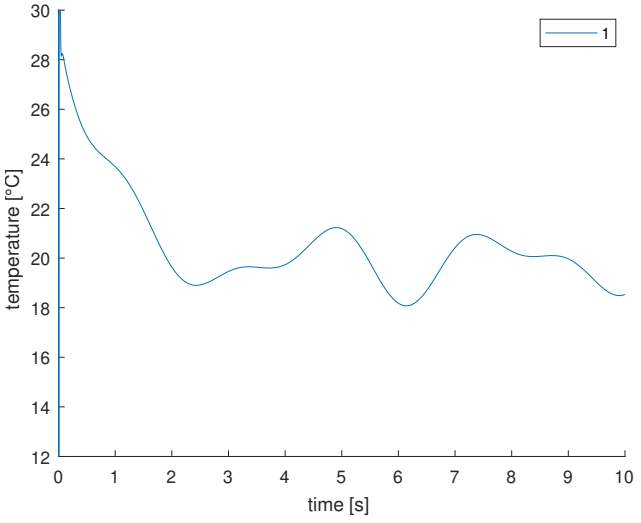
(b) control input



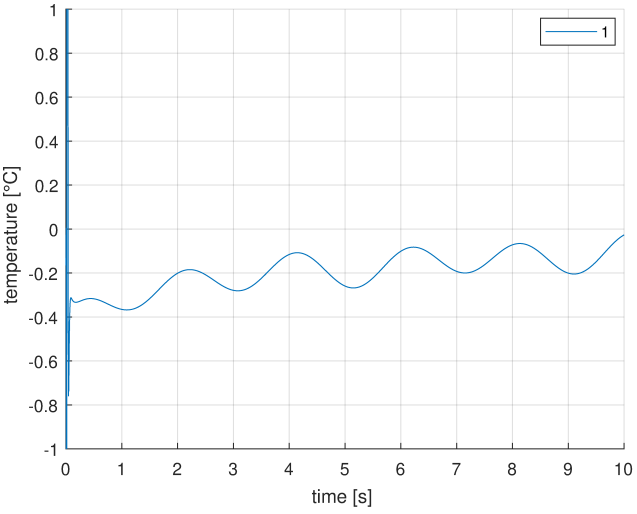
(c) measured output



(d) output configuration

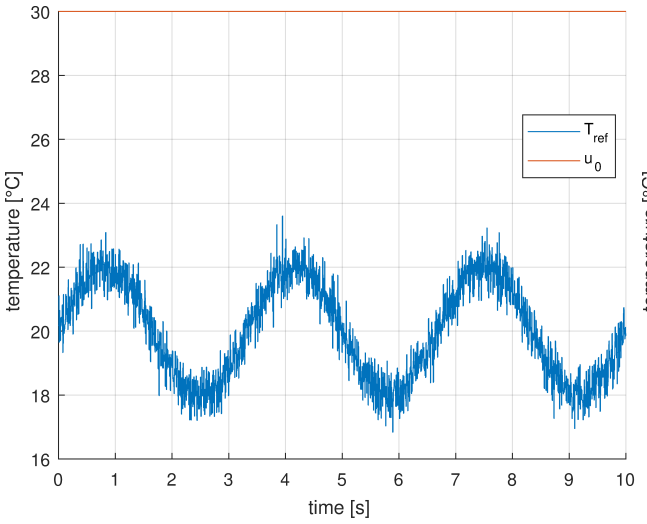


(e) virtual output

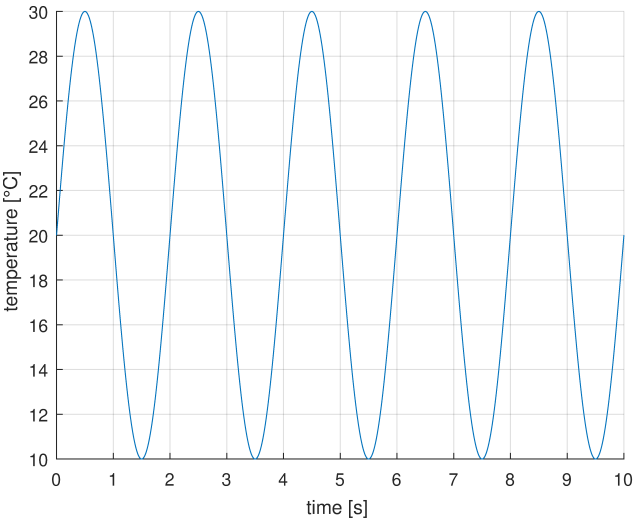


(f) virtual prediction error

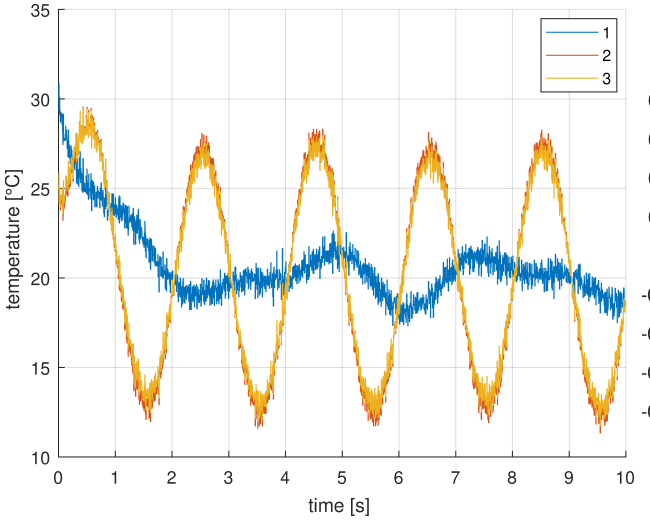
FIGURE 4.6: Results of *nominal* simulation using **full-state** observer based on 2D model. (a) and (b) show respectively noise and control input signals. (c) and (e) show measured and virtual output respectively, according to the mesh configuration shown in (d). The error of the virtual prediction against the true model is plotted in (f).



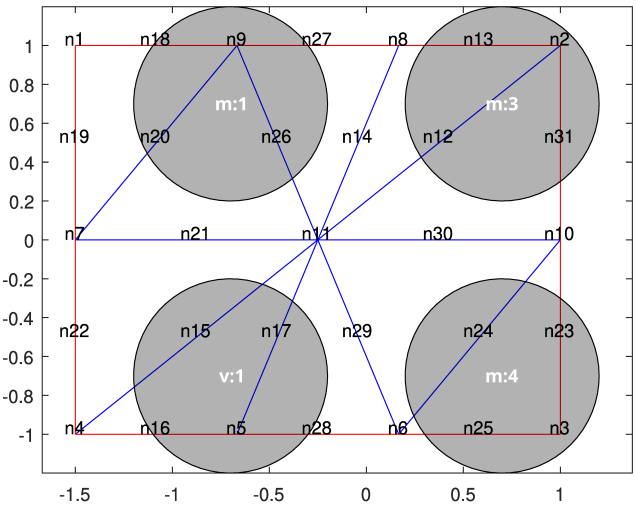
(a) noise input



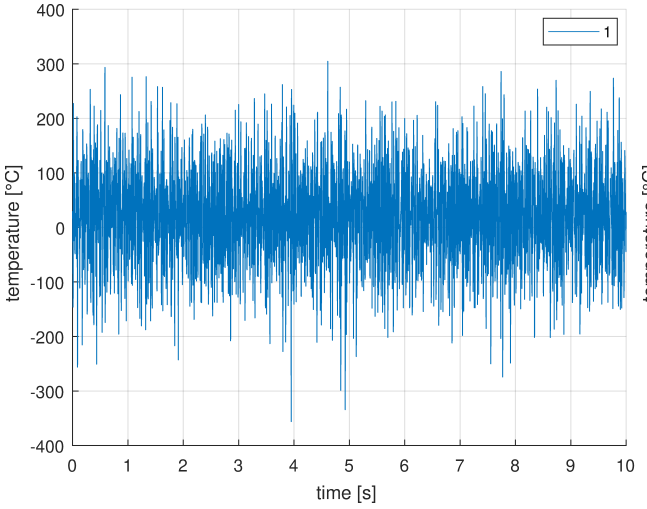
(b) control input



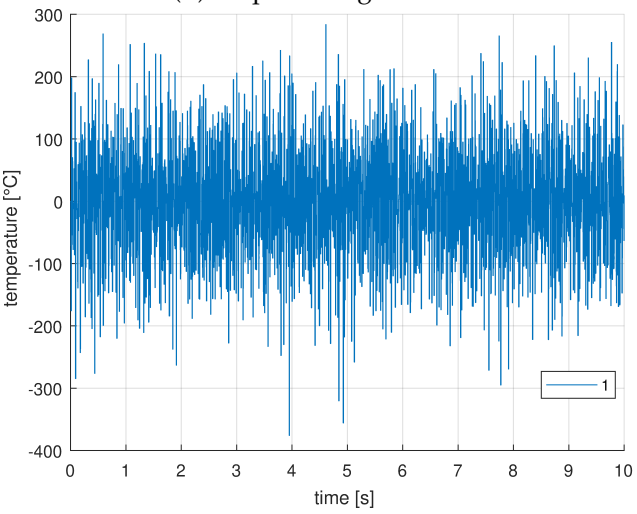
(c) measured output



(d) output configuration

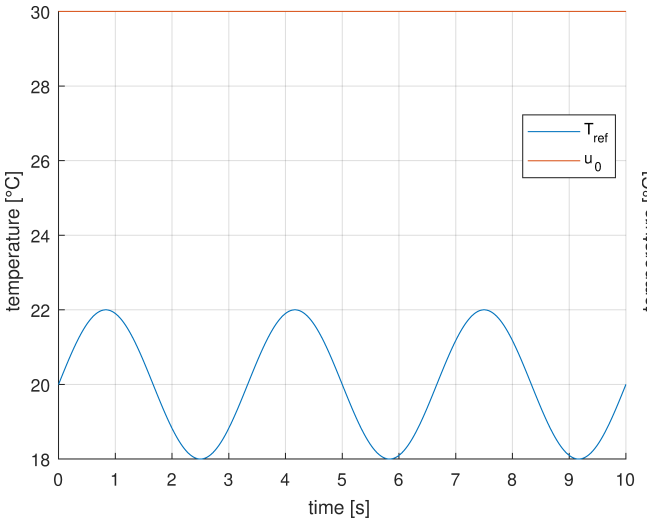


(e) virtual output

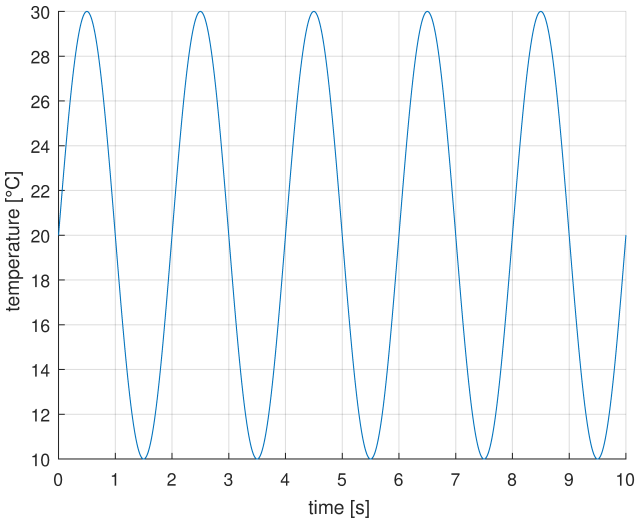


(f) virtual prediction error

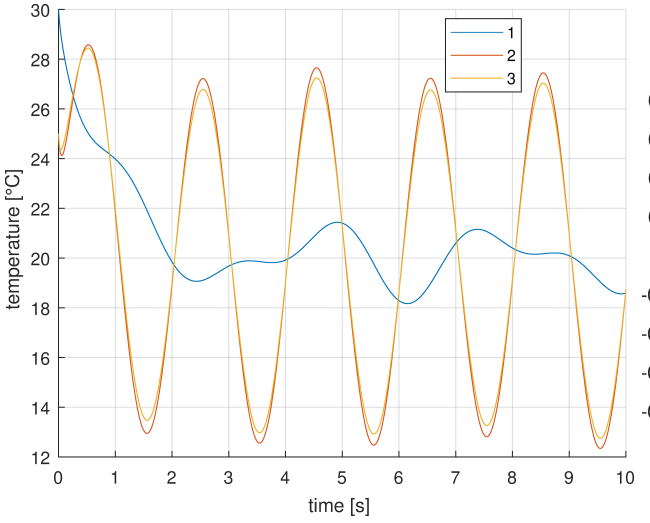
FIGURE 4.7: Results of *noisy* simulation using **full-state** observer based on 2D model. (a) and (b) show respectively noise and control input signals. (c) and (e) show measured and virtual output respectively, according to the mesh configuration shown in (d). The error of the virtual prediction against the true model is plotted in (f).



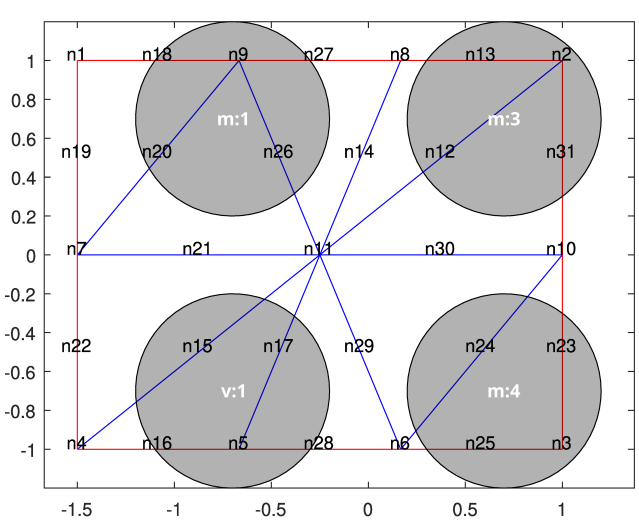
(a) noise input



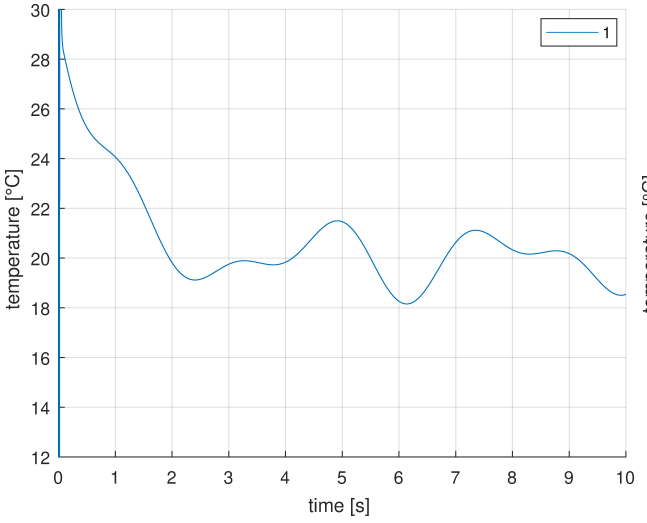
(b) control input



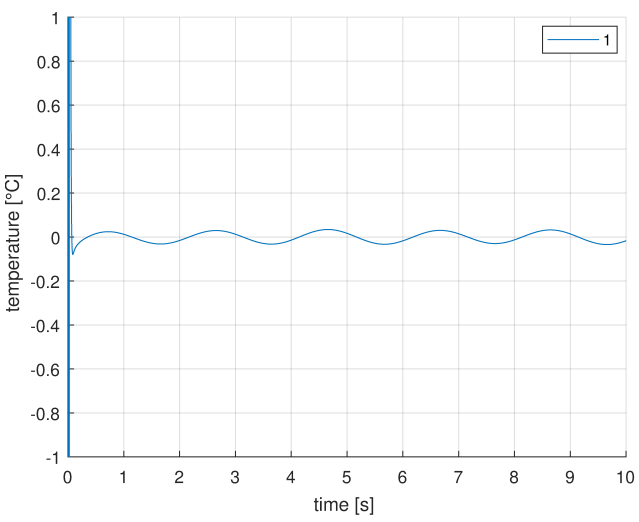
(c) measured output



(d) output configuration

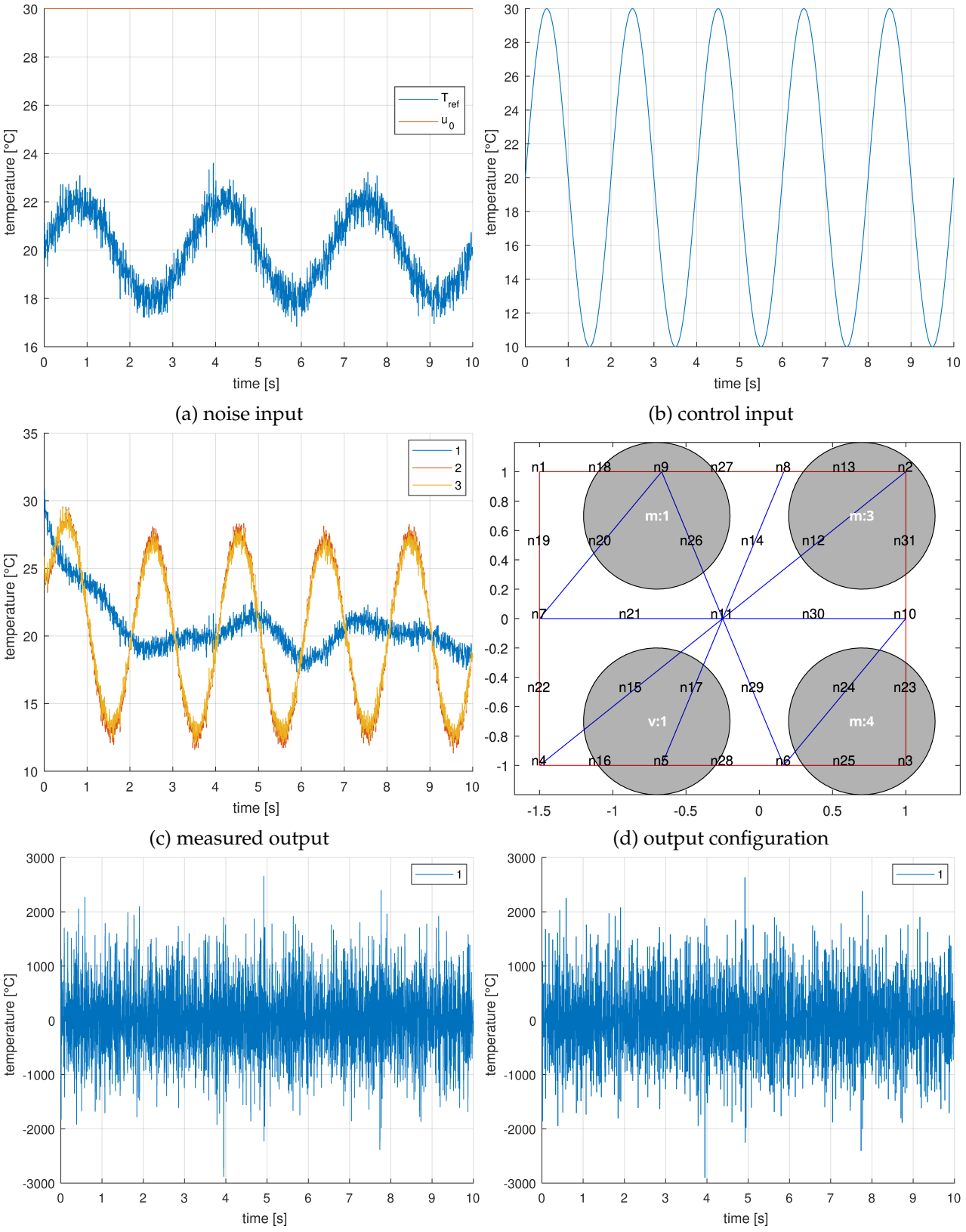


(e) virtual output



(f) virtual prediction error

FIGURE 4.8: Results of *nominal* simulation using**reduced-state** observer based on 2D model. (a) and (b) show respectively noise and control input signals. (c) and (e) show measured and virtual output respectively, according to the mesh configuration shown in (d). The error of the virtual prediction against the true model is plotted in (f).



(a) noise input

(b) control input

(c) measured output

(d) output configuration

(e) virtual output

(f) virtual prediction error

FIGURE 4.9: Results of *noisy* simulation using**reduced-state** observer based on 2D model. (a) and (b) show respectively noise and control input signals. (c) and (e) show measured and virtual output respectively, according to the mesh configuration shown in (d). The error of the virtual prediction against the true model is plotted in (f).

Chapter 5

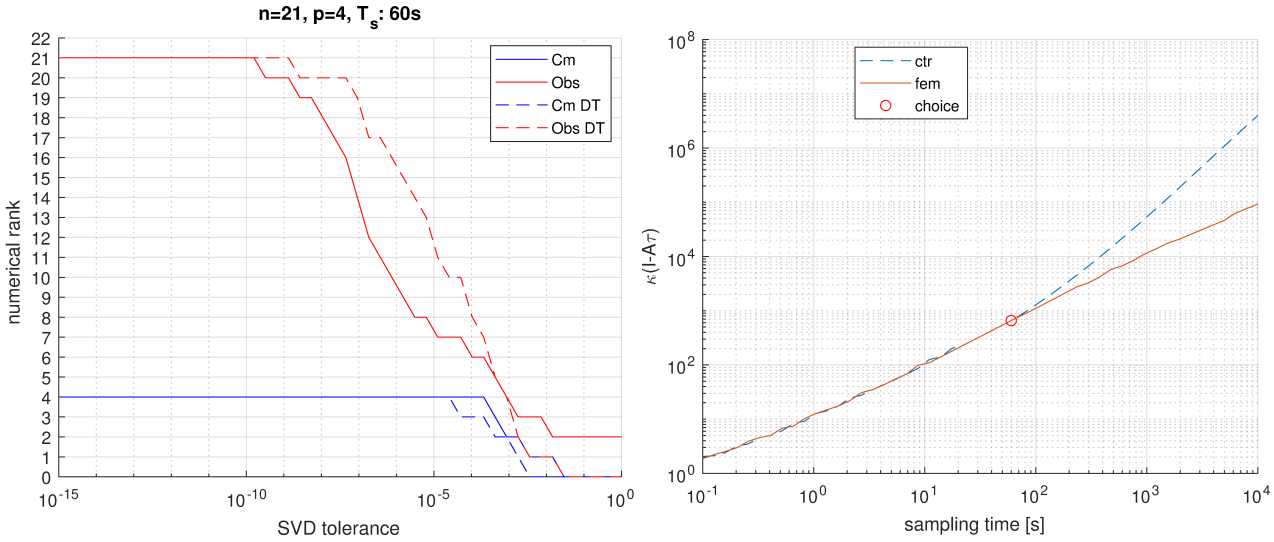
State reconstruction for virtual output prediction of reduced-order thermodynamic model based on RFX machine

Contents

This chapter describes the design and simulation of two new observers on RFX reduced-order thermal model. The first section describes the problems encountered in handling this new model while the next two sections recall essential formulation of H_2 and H_∞ optimal, infinite-horizon, design. The last section contains all design reports and simulation results, and it is longer than the preceding sections due to the presence of large amount of tables and figures. In particular, simulation figures presented at the end of this chapter show the behaviour of both traditional and norm-optimal observers (four observers) in two different input working conditions (nominal and noisy) for a total of eight figures.

5.1 Issues in the reconstruction problem of finite-element models

The problem of noise input handling was already addressed in SEC. 4.1.1. The same considerations apply to the RFX model and, in particular, to the fact that noise input makes observer design more difficult if moved to state space. For this reason, also in SEC. 5.2 and SEC. 5.3, noise inputs corresponding to ambient, external temperature T_{ref} and u_0 corresponding to the initial condition, are kept on the input side of the state space representation.

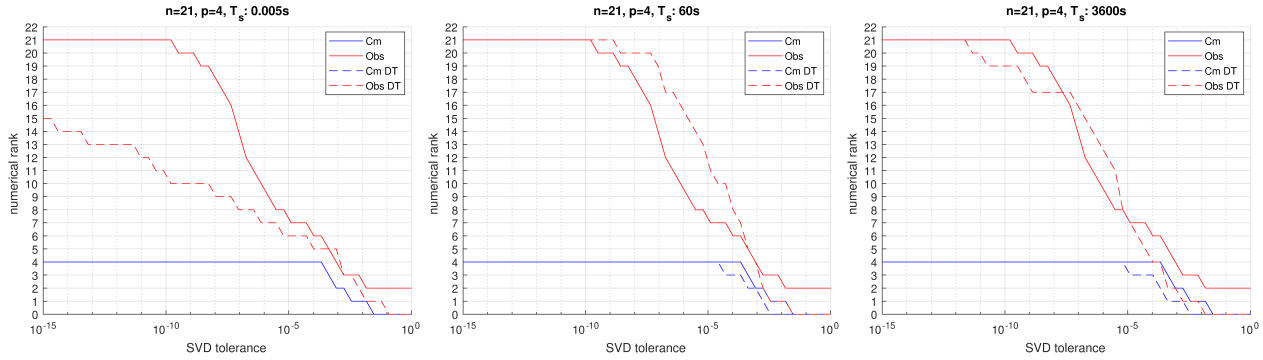


(a) numerical observability

(b) sampling conditioning

FIGURE 5.1: Numerical observability and discrete-time sampling condition number of reduced-order RFX model. Quadrant (a) reports the dependency of the observability matrix and of the (measured) output matrix from the numerical precision used in computing the their. A dashed line is used to distinguish the discrete-time system from the continuous-time counterpart. Quadrant (b) instead, reports the condition number of the map $\tau \mapsto (I - A\tau)$ whose inversion is required for sampling the CT system (see eq. (3.2)) The dashed line refers to transformation of the system where the non control inputs have been moved to system state (see SEC. 4.1.1).

FIG. 5.2 shows how the choice of the sampling time of the CT system affects the observability of the discrete-time system used as a starting point for design procedures of observers.



(a) $T_s = 5\,\mathrm{ms}$ (b) $T_s = 1\,\mathrm{minute}$ (c) $T_s = 1\,\mathrm{hour}$

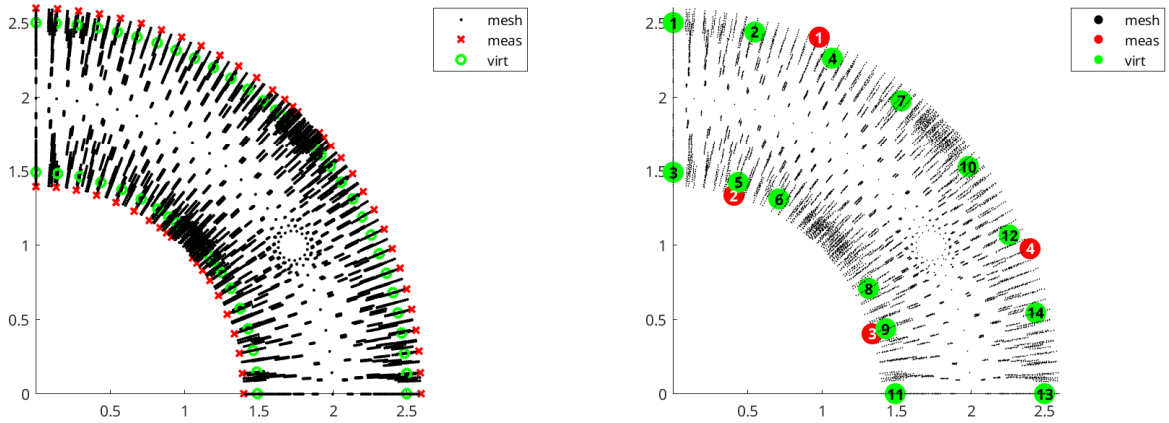
FIGURE 5.2: Effect of sampling time with respect to numerical observability of reduced-order RFX model

FIG. 5.1 plots the choice of the present works (1 minute) and displays the resulting conditioning of corresponding the Euler-Backwards map (eq. (3.2)). It can be seen that lowering the sampling time τ below $1\,\mathrm{s}$ in order to obtain a better conditioned sampling matrix $\left[I - A\tau\right]$, leads to loosing the observability property in the discrete-time systems. These aspects are important because

- the sampling time directly relates to the overall dynamics of the control loop and affects the requirements of the micro-controller board effectively implementing the reconstruction activity
- the loss of observability in the discrete-time system leads to high-gain entries in the gain matrix L with the appearance of peaking phenomena in the error transiente (see SEC. 5.4)

5.1.1 Choosing the number of outputs

The specification for sensor locations of RFX machine was described in FIG. 2.2 as possible locations, but no choice was made in terms of number of sensors and their relative distribution. FIG. 5.3 shows the choice made in terms of the location of the measured temperatures, and the position of the outputs to be predicted A combination of 4 measured outputs and 14 virtual outputs is retained. The location of the 18 outputs is selected in uniform fashion from the possible locations. Such decision is make with a qualitative approach in mind and could be the subject of further studies, in particular, it would be interesting to find a limiting threshold that ensures good prediction quality on the virtual outputs based on the ratio between the measured outputs and their locations.



(a) possible output locations

(b) measured and virtual location choice

FIGURE 5.3: Output definition for RFX full-order thermodynamic model (a) is the same as in FIG. 2.2 and visualizes the possible output locations used as specification for the design of the output equations. Measured outputs are marked in red while virtual outputs are marked in green (b) shows the choice of the locations for measured signals (red) and virtual outputs t be predicted (green).

5.2 Design of infinite-horizon, H_2 optimal, Kalman estimator

The following theorem describes sufficient conditions to design an infinite-horizon H_2 Wiener-Kalman filter and represents the heart of ALG. 10. Virtual outputs are denoted with the letter s while measured outputs are referred to using the letter y.

Theorem 1 (**[theorem 13.2.1**, 24]). *Consider a strictly causal discrete-time, state-space system of dimension n having m inputs, p measured outputs with transfer function $H(z) = H\left[Iz - F\right]^{-1} G$, and q virtual outputs with transfer function $L(z) = L\left[Iz - F\right]^{-1} G$. When the pair (F, H) is detectable, and $\left(F, GQ^{\frac{1}{2}}\right)$ is controllable on the unit circle, the problem*

$$\min_{\text{causal } K(z)} \| \begin{bmatrix} (L(z) - K(z)H(z))Q_u^{1/2} & -K(z)R^{1/2} \end{bmatrix} \|_2$$

where the process noise variance $Q_u \geq 0$ and the measurement noise variance $R > 0$; admits solution as state-system

$$\begin{cases} \hat{x}_{i+1} = (F - K_p H)\, x_i + K_p\, y_i \\ \hat{s}_i = L(I - PH^* R_e^{-1} H)\, x_i + LPH^* R_e^{-1}\, y_i \end{cases}$$

with $K_p = FPH^* R_e^{-1}$, and $R_e = R + HPH^*$ using the unique stabilising solution of the DARE

$$P = FPF^* + GQ_u G^* - K_p R_e K_p^*$$

According to [24, eq.(13.1.5)] any discrete-time, linear, time-invariant system with measurement noise v_i can be put in strictly-causal form by moving the control signals to state space, thus resulting in a new state vector $z_i = \begin{pmatrix} x_i \\ u_i \end{pmatrix}$ of dimension $\bar{n} = n + m$

$$\begin{aligned} z_{i+1} &= \overbrace{\begin{bmatrix} F & G \\ 0 & 0 \end{bmatrix}}^{\bar{F}} \begin{pmatrix} x_i \\ u_i \end{pmatrix} + \overbrace{\begin{bmatrix} 0 \\ I_m \end{bmatrix}}^{\bar{G}} u_{i+1} \\ y_i &= \overbrace{\begin{bmatrix} H & J \end{bmatrix}}^{\bar{H}} \begin{pmatrix} x_i \\ u_i \end{pmatrix} + v_i \\ s_i &= \overbrace{\begin{bmatrix} L & M \end{bmatrix}}^{\bar{L}} \begin{pmatrix} x_i \\ u_i \end{pmatrix} \end{aligned} \tag{5.1}$$

Standard Riccati equation solvers are available in MATLAB , for example the command `idare` [25] which solves a DARE of the form:

$$A^T X A - E^T X E - (A^T X B + S)[B^T X B + S]^{-1}(A^T X B + S)^T + Q = 0 \tag{5.2}$$

This solver can be set to compute a stabilising solution $X \sim P$ from the previous theorem, eventually transposing the notation to strictly causal form, by choosing

$$\begin{aligned} &E = I \qquad A = F^T \quad Q = GQ_u G^T \\ &B = H^T \quad S = 0 \qquad R \end{aligned} \tag{5.3}$$

The implemented algorithm is described in appendix A.3.2 with reference to MATLAB script in the repository source code. Depending on the particular choice of the solver, a number of necessary or sufficient conditions should be checked before starting the computation of P. For example, in ALG. 10, the former are checked by `assert` while the latter are logged with a `warning`.

The final observer structure for a dynamic state-space block can be computed using the original notation as:

$$\begin{aligned} \hat{x}_{i+1} = &\underbrace{Z_x(\bar{F} - K_p\bar{H})Z_x^T}_{\text{obs.A}}\ \hat{x}_i &&+\underbrace{Z_x(\bar{F} - K_p\bar{H})Z_u^T}_{\text{obs.B}_\text{u}}\ u_i &&+\underbrace{Z_x K_p}_{\text{obs.B}_\text{y}}\ y_i \\ \hat{s}_i = &\underbrace{\bar{L}(I - P\bar{H}^* R_e^{-1}\bar{H})Z_x^T}_{\text{obs.C}}\ \hat{x}_i &&+\underbrace{\bar{L}(I - P\bar{H}^* R_e^{-1}\bar{H})Z_u^T}_{\text{obs.D}_\text{u}}\ u_i &&+\underbrace{\bar{L}P\bar{H}^* R_e^{-1}}_{\text{obs.D}_\text{y}}\ y_i \end{aligned} \tag{5.4}$$

where (Z_x, Z_u) is a partition of the the identity matrix in the strictly causal space of dimension $\bar{n}$ allowing to write $z_i = Z_x x_i + Z_u u_i$

The structure of eq. (5.4) may appear different and cumbersome with respect to what is presented in [24]. This can be explained in two steps. First, the reference book presents a formulation that does not include the D term in the output matrix, which is required for the work presented here. Second, as a consequence of this modification, the formulas are solved in an augmented state space what requires further algebraic handling in order to obtain the estimator block in traditional input/output form. This is done by developing the original formulation with the partitioning (Z_x, Z_u) allowing to come back to classical observer inputs u and y.

As a final remark, notice that this formulation is LTI and different from a structural point of view to more classical finite-horizon Kalman filters [26], [27] where the covariance matrix is updated every at every step. The infinite-horizon formulation here implemented, has at least two advantages:

- it is static and therefore less demanding in terms of computations and memory usage of the controller board
- being LTI, it can be compared with traditional full-state and reduced-state, feedback observers, as the only difference consists on the values of the observer dynamic matrices.

5.3 Design of infinite-horizon, H_∞ sub-optimal estimator

The following theorem describes equivalent conditions for computing an infinite-horizon H_∞ sub-optimal estimator, is reported below, as it represents the heart of ALG. 9. Virtual outputs are denoted with the letter s while measured outputs are referred to using the letter y.

Theorem 2 (**[theorem 13.3.1**, 24]). *Consider a strictly causal discrete-time, state-space system of dimension n having m inputs, p measured outputs with transfer function $H(z) = H\left[Iz - F\right]^{-1} G$, and q virtual outputs with transfer function $L(z) = L\left[Iz - F\right]^{-1} G$. When the pair (F, H) is detectable, and (F, G) is controllable on the unit circle, a causal estimator with transfer function $K(z)$ solving the problem*

$$\|\begin{bmatrix} L(z) - K(z)H(z) & -K(z) \end{bmatrix}\|_\infty < \gamma$$

exists if and only if:

- $\exists P \in \mathbb{M}^n$ *solving the DARE* $P = FPF^* + GG^* - K_p R_e K_p^*$ *with* $K_p = FP\begin{bmatrix} H^* & L^* \end{bmatrix} R_e^{-1}$ *and* $R_e = \begin{bmatrix} I_p & 0 \\ 0 & -\gamma^2 I_q \end{bmatrix} + \begin{bmatrix} H \\ L \end{bmatrix} P \begin{bmatrix} H^* L^* \end{bmatrix}$
- P *is stabilising, in the sense that* $F - K_p \begin{bmatrix} H \\ L \end{bmatrix}$ *has all eigenvalues inside the unit circle*
- R_e *and* $\begin{bmatrix} I_p & 0 \\ 0 & -\gamma^2 I_q \end{bmatrix}$ *have the same inertia, meaning that they have the same number of positive, negative, and zero eigenvalues*
- $P \geq 0$ *has positive semi-definite character*

If the original system is given in non strictly-causal form, the eq. (5.1) can be used to transpose the notation to a strictly causal state space system, generally denoted with a bar over the dynamic matrices $(F, G, H, L) \to \left(\bar{F}, \bar{G}, \bar{H}, \bar{L}\right)$

The `idare`[25] solver, can be set as in the case of H_2 estimator eq. (5.2) compute a stabilising solution $X \sim P$ from the previous theorem, eventually transposing the notation to strictly causal form, by choosing

$$\begin{array}{lll} E = I_n & A = F^T & Q = GG^T \\ B = \begin{bmatrix} H^T & L^T \end{bmatrix} & S = 0 \in \mathbb{M}^{n \times (p+q)} & R = \begin{bmatrix} I_p & 0 \\ 0 & \gamma^2 I_q \end{bmatrix} \end{array} \tag{5.5}$$

The implemented algorithm is described in appendix A.3.2 with reference to MATLAB script in the repository source code. Depending on the particular choice of the solver, a number of necessary or sufficient conditions should be checked before starting the computation of P.

For example, in ALG. 9, the former are checked by `assert` while the latter are logged with a `warning`.

After a solution P meeting al criteria from theorem above, and from solver requirements, the Central estimator [24, eq. 13.3.8] can be formulated in state space form using the strictly causal notation $z_i = \begin{pmatrix} x_i & u_i \end{pmatrix}^T$:

$$\begin{cases} \hat{z}_{i+1} = \bar{F} z_i - \bar{F} P \bar{H}^* (I_p + \bar{H} P \bar{H}^*)^{-1} (y_i - \bar{H} \hat{z}_i) \\ \hat{s}_i = \bar{L} z_i + \bar{L} P \bar{H}^* (I_p + \bar{H} P \bar{H}^*)^{-1} (y_i - H \hat{z}_i) \end{cases}$$

which can be manipulated to obtain a gain matrix

$$K_c = P \bar{H}^* R_{He}^{-1} \quad R_{He} = I_p + \bar{H} P \bar{H}^*$$

The final observer structure for a dynamic state-space block can be computed using the original notation as:

$$\begin{aligned} \hat{x}_{i+1} = & \underbrace{(F + \begin{bmatrix} F & G \end{bmatrix} K_c H)}_{\text{obs.A}} \hat{x}_i + \underbrace{(G + \begin{bmatrix} F & G \end{bmatrix} K_c J)}_{\text{obs.B}_\text{u}} u_i + \underbrace{(- \begin{bmatrix} F & G \end{bmatrix} K_c)}_{\text{obs.B}_\text{y}} y_i \\ \hat{s}_i = & \underbrace{(L - \begin{bmatrix} L & M \end{bmatrix} K_c H)}_{\text{obs}-\text{C}} \hat{x}_i + \underbrace{(M - \begin{bmatrix} L & M \end{bmatrix} K_c J)}_{\text{obs.D}_\text{u}} u_i + \underbrace{\begin{bmatrix} M & L \end{bmatrix} K_c}_{\text{obs.D}_\text{y}} y_i \end{aligned} \tag{5.6}$$

The structure of eq. (5.6) may appear different and cumbersome with respect to what is presented in [24]. This can be explained with the same justification discussed previously for the Kalman estimator in SEC. 5.2

5.4 Simulation of RFX finite-element, reduced-order, dynamic model

5.4.1 System description

So far, the working model consisted of a series of transformations that led from a continuous-time full-order model in descriptor dynamic form (eq. (2.1) with the addition of output equation)

$$\begin{aligned} \underbrace{M}_{E}\dot{x} &= \underbrace{-(K+H)}_{A}x \quad + \underbrace{\begin{bmatrix} q_{\text{conv}} & q \end{bmatrix}}_{B}\begin{pmatrix} T_{\text{ref}} \\ u \end{pmatrix} \\ \begin{pmatrix} y_v \\ y_m \end{pmatrix} &= \begin{bmatrix} C_v \\ C_m \end{bmatrix} x \qquad\qquad\qquad +D\begin{pmatrix} T_{\text{ref}} \\ u \end{pmatrix} \end{aligned}$$

to a discrete-time full-order model in standard dynamic form (eq. (3.2))

$$\begin{aligned} x_{i+1} &= Fx_i + G\begin{pmatrix} T_{\text{ref}} \\ u \end{pmatrix}_i \\ \begin{pmatrix} y_v \\ y_m \end{pmatrix}_i &= \begin{bmatrix} H_v \\ H_m \end{bmatrix} x_i \qquad\qquad +J\begin{pmatrix} T_{\text{ref}} \\ u \end{pmatrix}_i \end{aligned}$$

According to the overall procedure described in SEC. 3.2.2, the model order reduction has been carried out in full-order continuous-time descriptor form. FIG. 5.4 illustrates the simulation diagram for testing the observer design. There are four dynamic blocks: one block contains the full-order model in descriptor dynamic form; two blocks represent the CT and DT full-order models in standard form, and one more block is used to plug in the observer design to be tested.

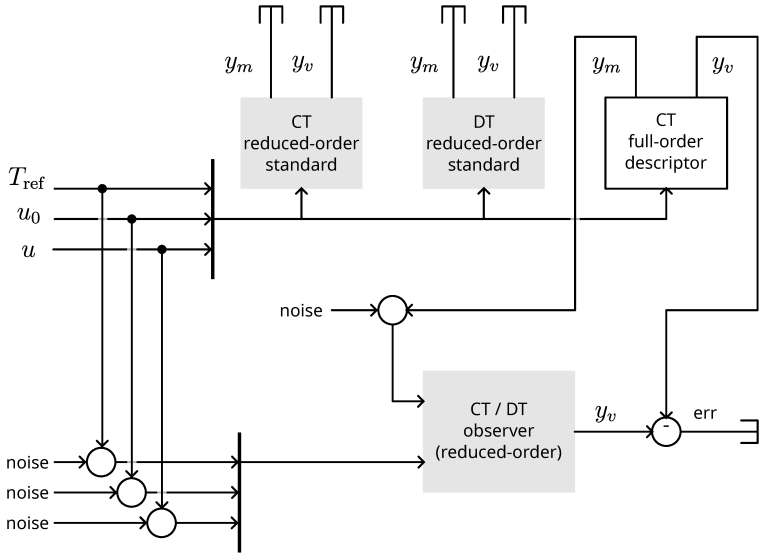


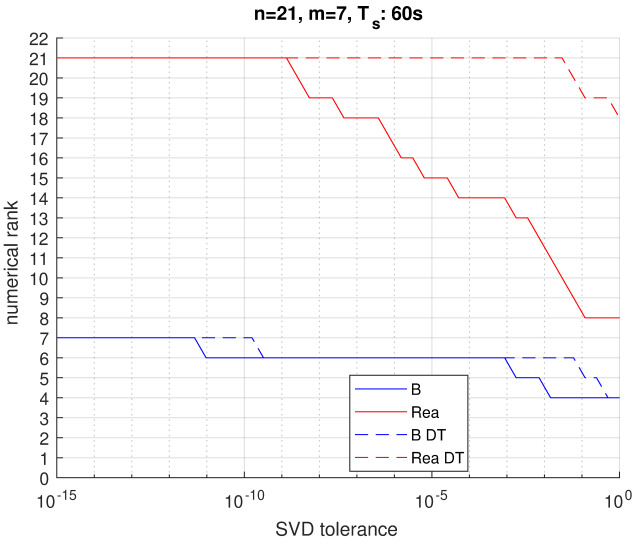
FIGURE 5.4: Block scheme for simulation of reduced-order models based on RFX machine. Clean inputs $(T_{\text{ref}}, u_0, T_d)$ are fed directly to reference models. Noise can be added as the signals are fed to the observer. All models are in standard dynamic form except for the upper right full-order model representing the physical reference (descriptor dynamic form).

5.4.2 Design logs

The application of ALG. 10, and ALG. 9 corresponding to the H_2 optimal and H_∞ sub-optimal design methods described above, resulted in FIG. 5.8, and FIG. 5.7. Traditional observers from ALG. 4 and ALG. 6 were also computed and results are shown, respectively, in FIG. 5.5 and FIG. 5.6, It can be seen that the pole placement of infinite-horizon design techniques preserves the slow poles of the original system. TAB. 5.1 shows a comparison between the new observer designs (H_2 and H_∞) to traditional observers. Results from TAB. 4.1 regarding the amplitude of gain components in traditional observer, are confirmed. Equivalent gain matrices could not be computed straight forwardly from eq. (5.4) and eq. (5.6) and therefore no a priori comparison could be made on possible high-gain effects. Interestingly, the resulting eigenvalue placement of the observer system for both H_2 optimal and H_∞ sub-optimal are much more relaxed than those obtained for traditional observer. This will be seen in the next section where considerably softer peaking phenomena will be observed in these new observer arhitectures.

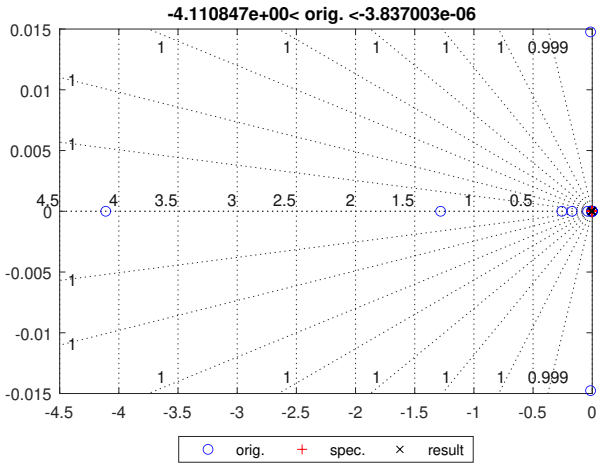
		full-state feedback observer	reduced-state feedback observer	infinite-horizon H_2 kalman	infinite-horizon H_∞ sub-optimal
eig.Val result	min	-6×10^{-3}	-5×10^{-3}	-8×10^{-1}	-8×10^{-1}
	max	-1×10^{-3}	-1×10^{-5}	-7×10^{-6}	-4×10^{-6}
CT high-gain	min	-2×10^{4}	-5×10^{4}	N/A	N/A
	max	3×10^{3}	8×10^{4}	N/A	N/A
DT high-gain	min	-5×10^{6}	-6×10^{5}	N/A	N/A
	max	6×10^{6}	1×10^{6}	N/A	N/A

TABLE 5.1: Design results comparison between observers based on rduced-order RFX model. Specifications are defined for continuous-time design and then converted to discrete-time poles. Three aspects are analyzed: the resulting eigenvalue placement, the bounds of the entries in gain matrix from the design based on the CT system, and the gain bounds resulting from the DT design.

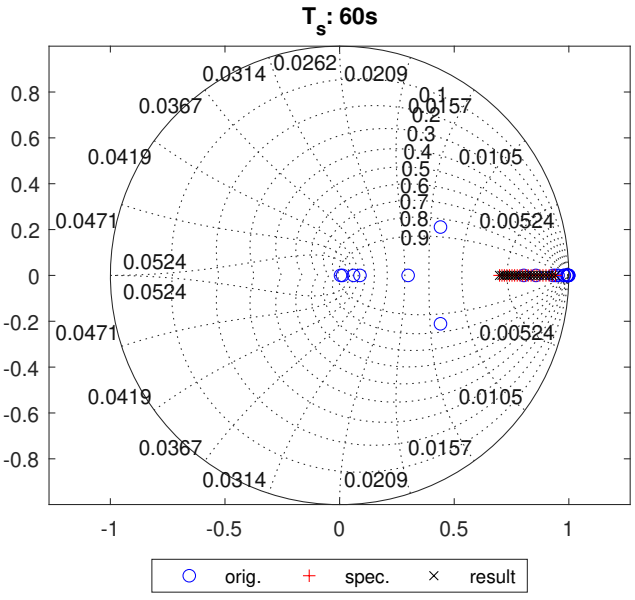


(a) observer num. controllability

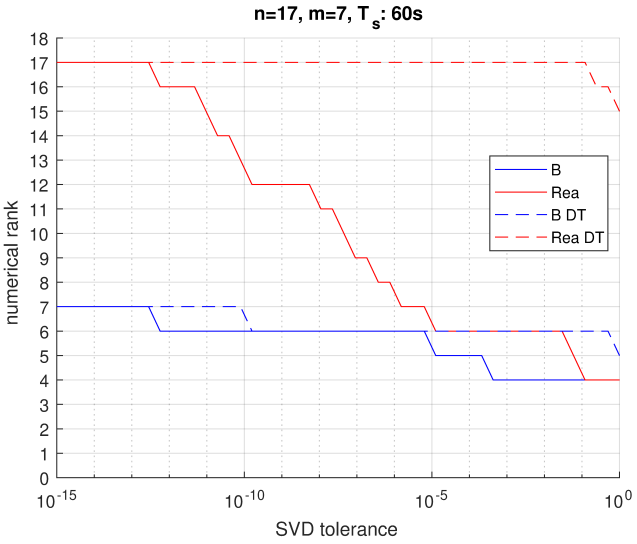
FIGURE 5.5: Design plots of continuous-time and discrete-time full-state observer based on RFX reduced-order model. (a) shows the rank of the controllability matrix and of the input matrix for the CT and DT observer with respect to numerical precision used by MATLAB `rank` function. (b) depicts the CT placement of original, specification, and resulting poles after allocation with MATLAB `place` method. (c) depicts the DT placement of original, specification, and resulting poles after allocation with MATLAB `place` method.



(b) continuous-time poles

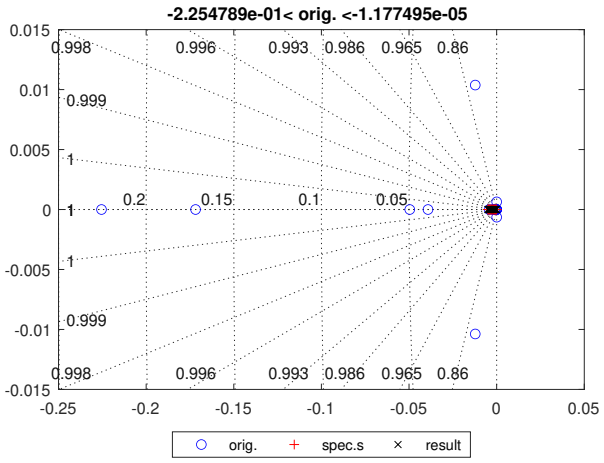


(c) discrete-time poles

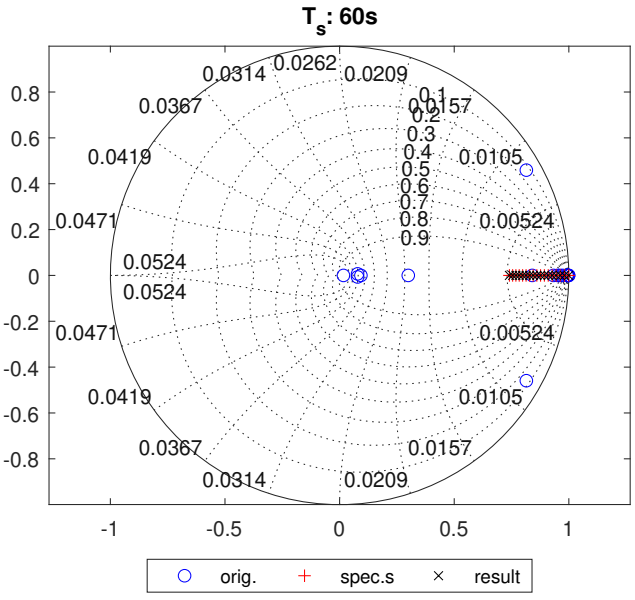


(a) observer num. controllability

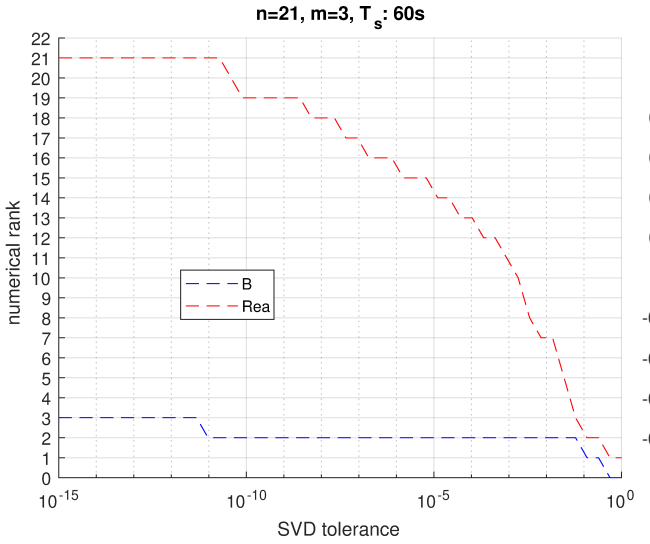
FIGURE 5.6: Design plots of continuous-time and discrete-time reduced-state observer based on RFX reduced-order model. (a) shows the rank of the controllability matrix and of the input matrix for the CT and DT observer with respect to numerical precision used by MATLAB `rank` function. (b) depicts the CT placement of original, specification, and resulting poles after allocation with MATLAB `place` method. (c) depicts the DT placement of original, specification, and resulting poles after allocation with MATLAB `place` method.



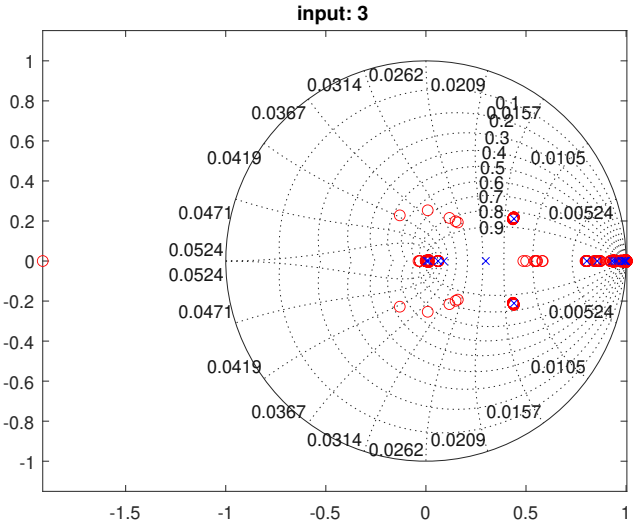
(b) continuous-time poles



(c) discrete-time poles

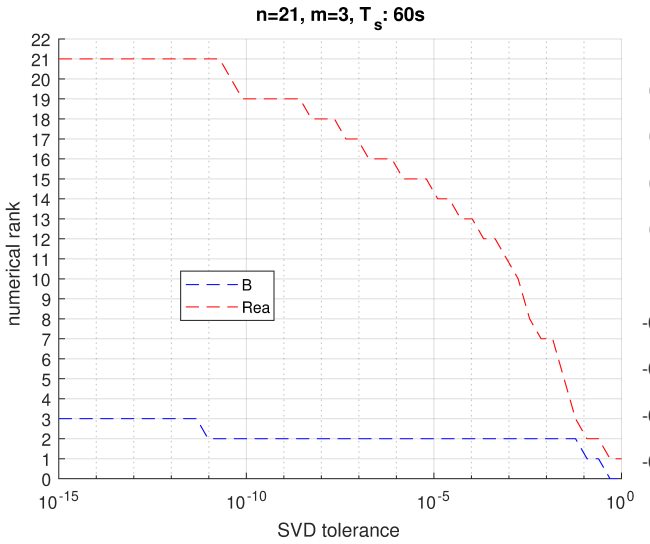


(a) observer num. controllability

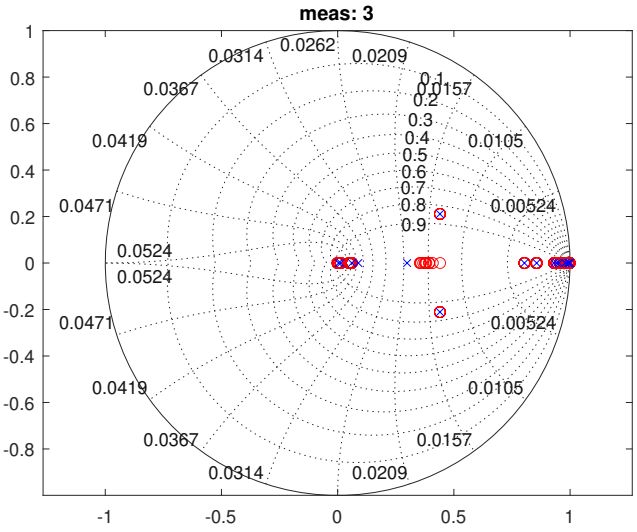


(b) resulting discrete-time poles and zeros

FIGURE 5.7: Design plots of H_∞ discrete-time observer based on RFX reduced-order model (a) shows the rank of the controllability matrix and of the input matrix for the DT observer with respect to numerical precision used by MATLAB `rank` function. (b) depicts the DT placement of resulting poles after allocation with MATLAB `idare` method.



(a) observer num. controllability



(b) resulting discrete-time poles and zeros

FIGURE 5.8: Design plots of H_2 discrete-time observer based on RFX reduced-order model. (a) shows the rank of the controllability matrix and of the input matrix for the DT observer with respect to numerical precision used by MATLAB `rank` function. (b) depicts the DT placement of resulting poles after allocation with MATLAB `idare` method.

5.4.3 Results on transient prediction of virtual outputs

Simulation results from FIG. 5.9 to FIG. 5.12 are obtained by computing the block scheme presented in FIG. 5.4 with full-state and reduced-state observers in nominal and noisy conditions. Nominal conditions, do not inject any noise in the points defined in the block scheme except for the external ambient temperature T_{ref}. Noisy conditions, on the other hand, add white noise to measurement outputs fed into the observer block before prediction.

As al ready presented in SEC. 4.4, the structure of these figures is made of six quadrants each depicting a different type of signal: the upper two represent noise inputs and control inputs, the two quadrants at the center describe the measured outputs and their configuration with respect to the FEM mesh and virtual output. The latter are shown in the bottom two quadrants where the error with respect to the true model (simulation of full-order model in descriptor form) has been plot. Differently from SEC. 4.4, however, only a few selected signals have been plotted in the results of this section: the scripts linked in appendix A allow the readed to reproduce fully all the output signals.

Firstly, clean simulations on traditional observers on FIG. 5.9 and FIG. 5.11 confirm the acceptable behaviour of full-state and reduced-state observers with RXF reduced model. This is also the case for noisy simulations on FIG. 5.10 and FIG. 5.12, where both designs amplify white noise in the measured output, thus proving these observers unsuitable to the real reconstruction problem.

Secondly, clean simulations on infinite-horizon estimators on FIG. 5.13 and FIG. 5.15 suggest better reconstruction performance of H_2 and H_∞ filters as they show considerably smaller error amplitude on the virtual inputs. Noisy simulation on H_2 and H_∞ filters on FIG. 5.14 and FIG. 5.16 confirm their superior performance as compared with traditional observers. In particular, H_∞ does not amplify white noise on measured outputs, and H_2 effectively filters out white noise from measured signals. Finally, transient behaviour is summarised in TAB. 5.2 where good properties of H_2 and H_∞estimators can be found.

		full-state observer (nominal)	reduced-state observer (nominal)	infinite-horizon H_2 (noisy y_m)	infinite-horizon H_∞ (noisy y_m)
min peak	°C	-3	-6.5	≈ 0	-3.7
max peak	°C	2.5	3.6	<0.1	3.2
peak time	h	1.8	3.2	1.9	1.3
err. Tol.	°C	3	5	0.1	2
asymptotic error	°C	0.7	0.6	<0.1	<0.1
error variance	°C	1.1	1.9	<0.1	≈ 1

TABLE 5.2: Transient characteristic comparison between observers based on reduced-order RFX model

5.4.4 Empirical robustness findings

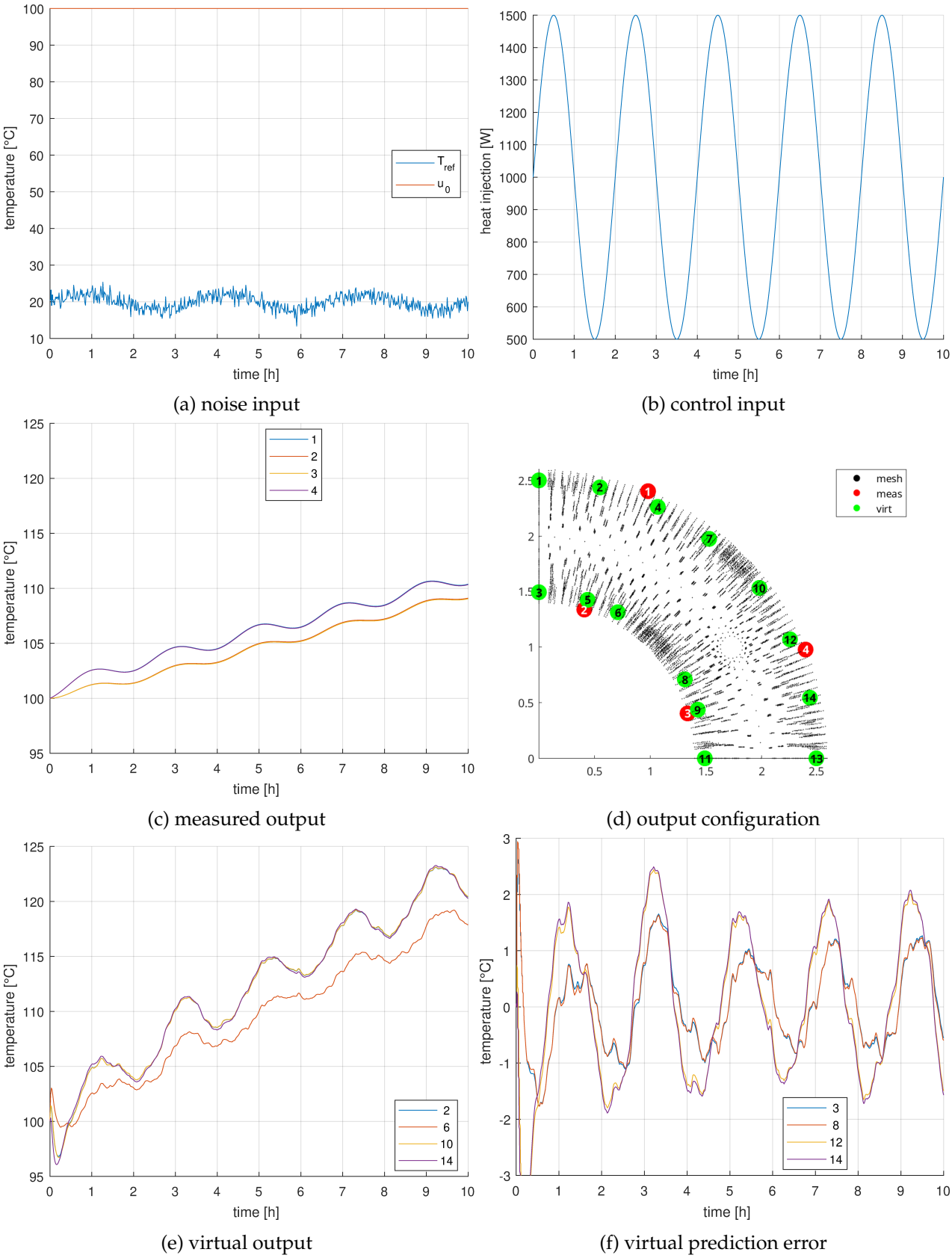
Considering the noise injection points in FIG. 5.4, many different simulations could be run. Here, only two settings are studied: white noise and bias noise. While the former has been plot in the noisy simulations of this section, the latter has been explored in appendix B.2.2. The results of this exploration have been summarised in TAB. 5.3 .

	noise type	full-state feedback observer	reduced-state feedback observer	infinite-horizon H_2 kalman	infinite-horizon H_∞ sub-optimal
T_{ref}	white	•	•	•	•
	bias	○	○	•	•
u_0	bias	○	○	○	○
y_{meas}	white	bad	bad	•	○
	bias	○	○	•	○

TABLE 5.3: Empiric, robustness, results comparison between observers based on reduced-order RFX model White noise corresponds to approximately $\pm 3\,^{\circ}\mathrm{C}$, bias noise were set to $5\,^{\circ}\mathrm{C}$.

Notice, that no noise was ever added in the control input (injected heat) u. Looking at TAB. 5.3, it can be seen that results from TAB. 4.3 are confirmed with the exception of robustness in white noise added to T_{ref} which may be explained as an added bonus from the order reduction operations. Un surprisingly, such kind of robustness is found on the more advanced H_2 and H_∞ estimators as well.

Particular attention, should be put to u_0 line in TAB. 5.3, where it is found that none of the proposed observers tolerates bias error on the initial condition coordinate (eq. (3.1)). This highlights a fundamental flaw in the proposed solution to the reconstruction problem analised so far. Indeed, if a wrong initial condition is set with respect to the physical reality of the machine, there is no good expectation of accurate virtual output prediction.



(a) noise input

(b) control input

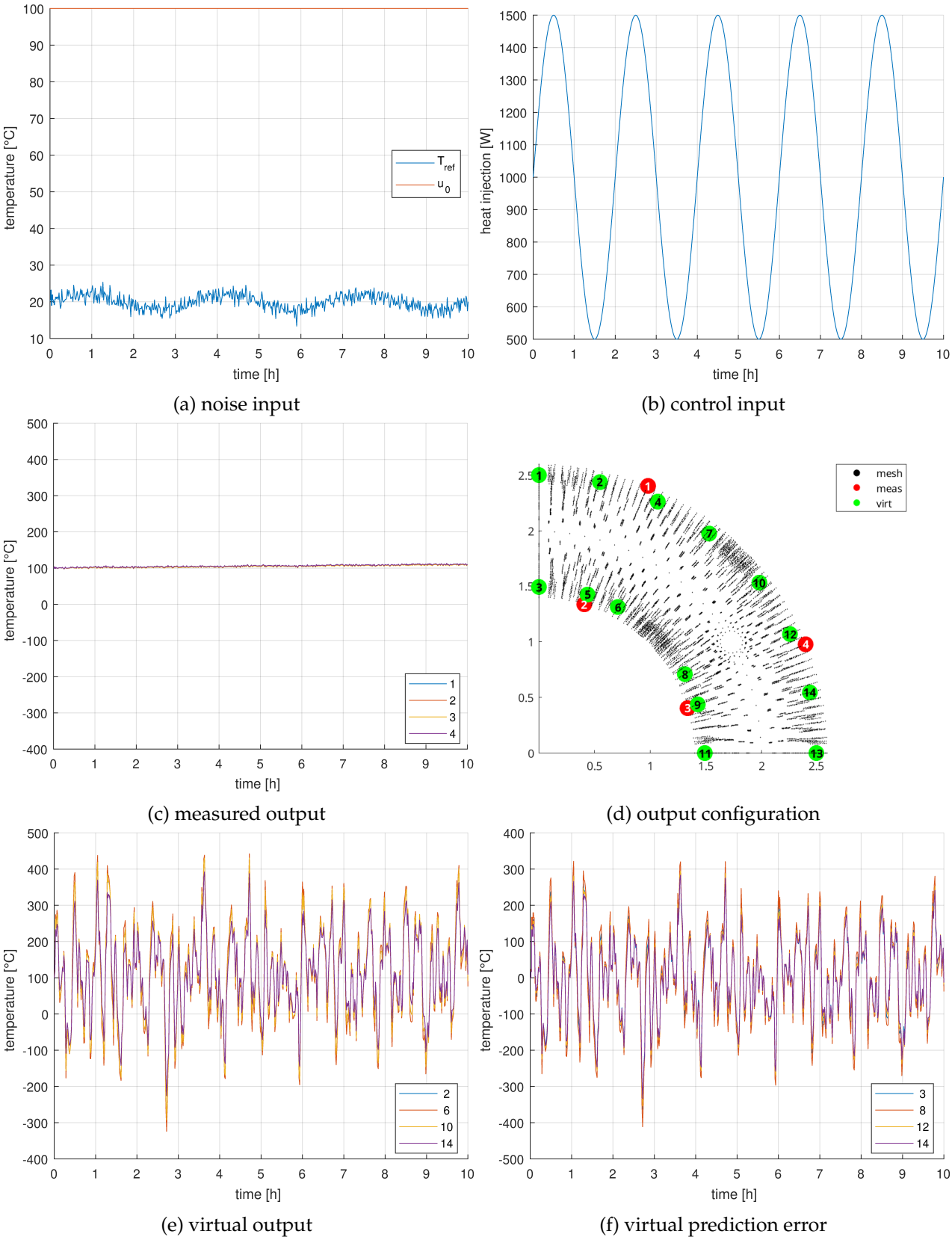
(c) measured output

(d) output configuration

(e) virtual output

(f) virtual prediction error

FIGURE 5.9: Results of *nominal* simulation using **full-state** observer based on RFX model. (a) and (b) show respectively noise and control input signals. (c) and (e) show measured and virtual output respectively, according to the mesh configuration shown in (d). The error of the virtual prediction against the true model is plotted in (f).



(a) noise input

(b) control input

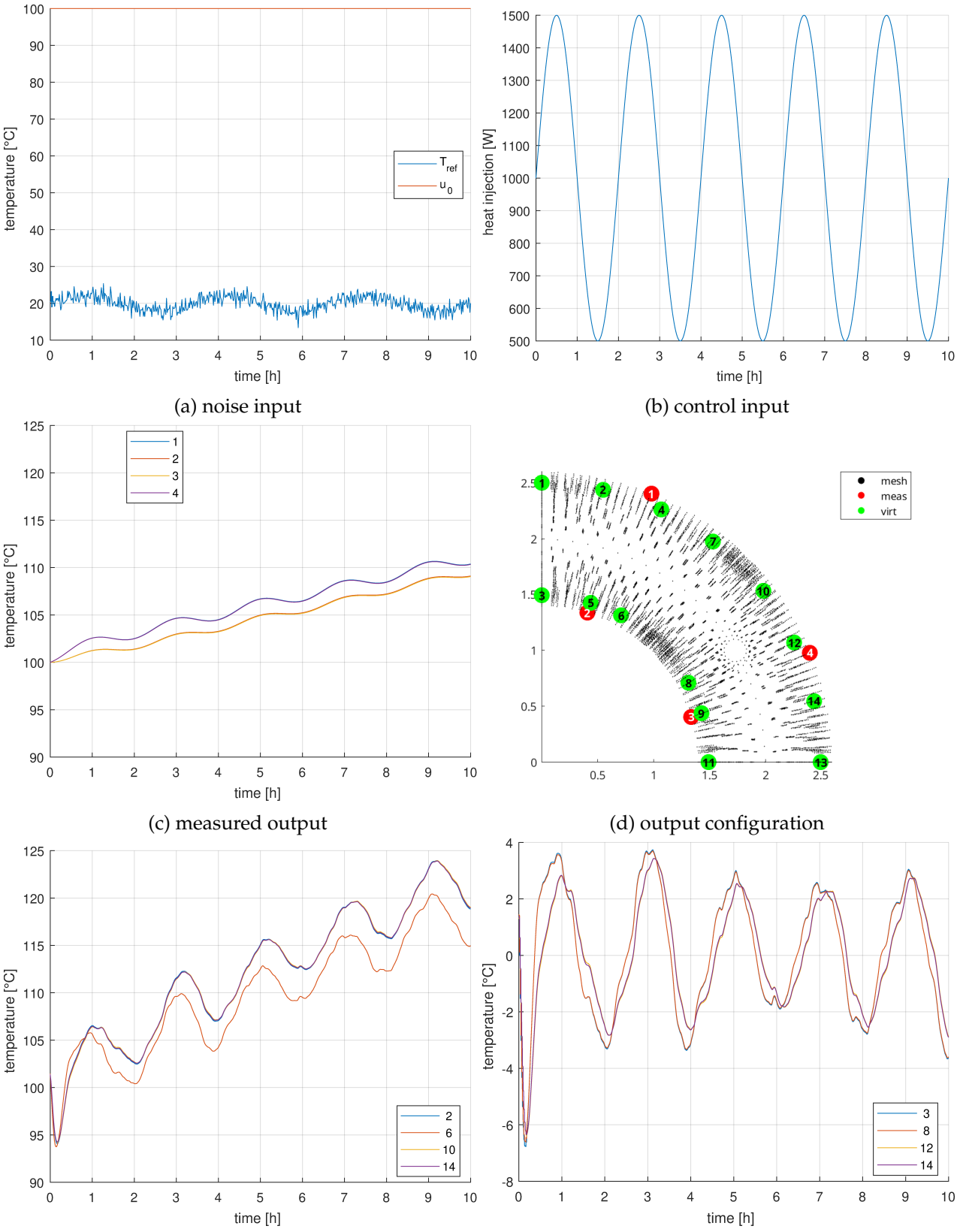
(c) measured output

(d) output configuration

(e) virtual output

(f) virtual prediction error

FIGURE 5.10: Results of *noisy* simulation using **full-state** observer based on RFX model. (a) and (b) show respectively noise and control input signals. (c) and (e) show measured and virtual output respectively, according to the mesh configuration shown in (d). The error of the virtual prediction against the true model is plotted in (f).



(a) noise input

(b) control input

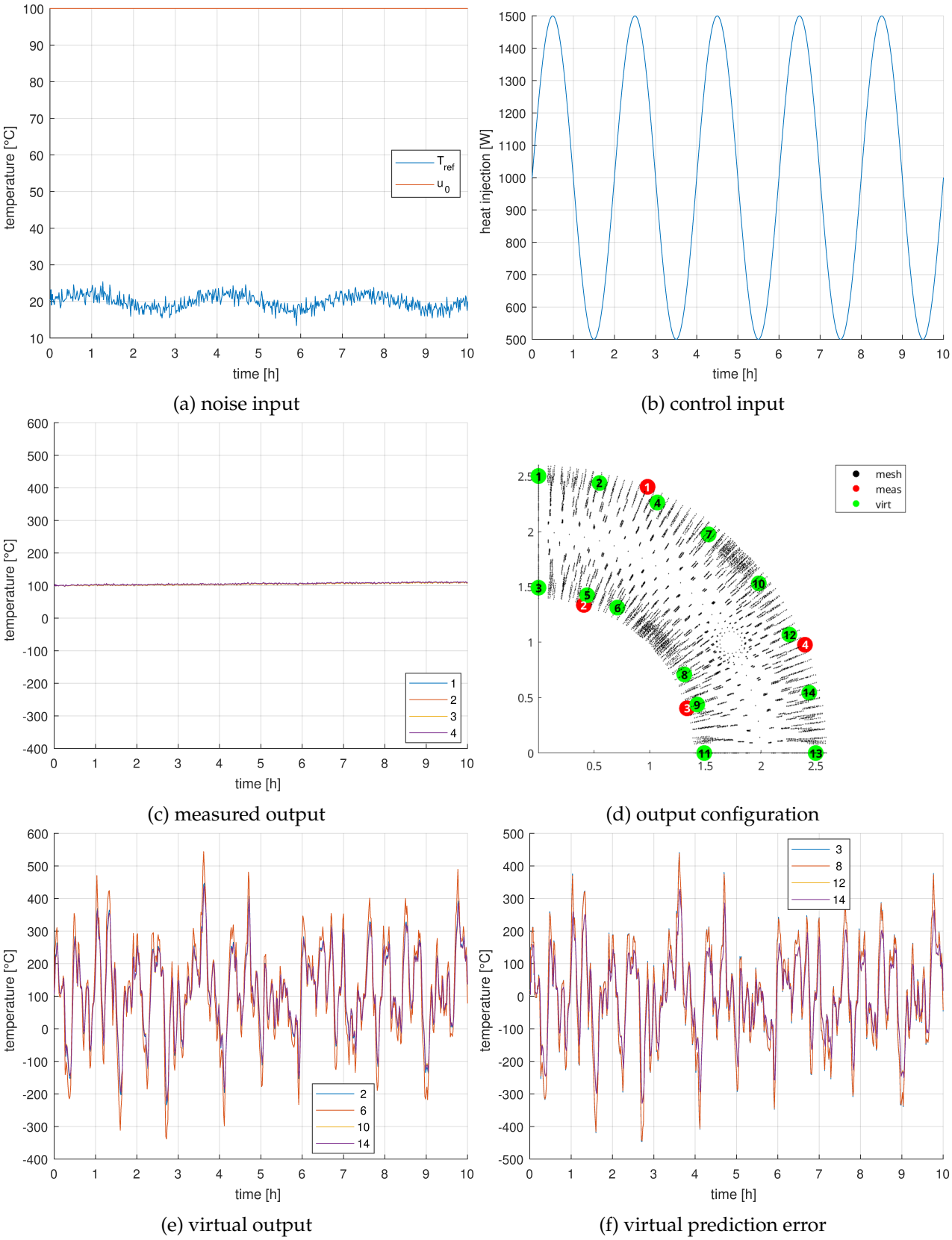
(c) measured output

(d) output configuration

(e) virtual output

(f) virtual prediction error

FIGURE 5.11: Results of *nominal* simulation using **reduced-state** observer based on RFX model. (a) and (b) show respectively noise and control input signals. (c) and (e) show measured and virtual output respectively, according to the mesh configuration shown in (d). The error of the virtual prediction against the true model is plotted in (f).



(a) noise input

(b) control input

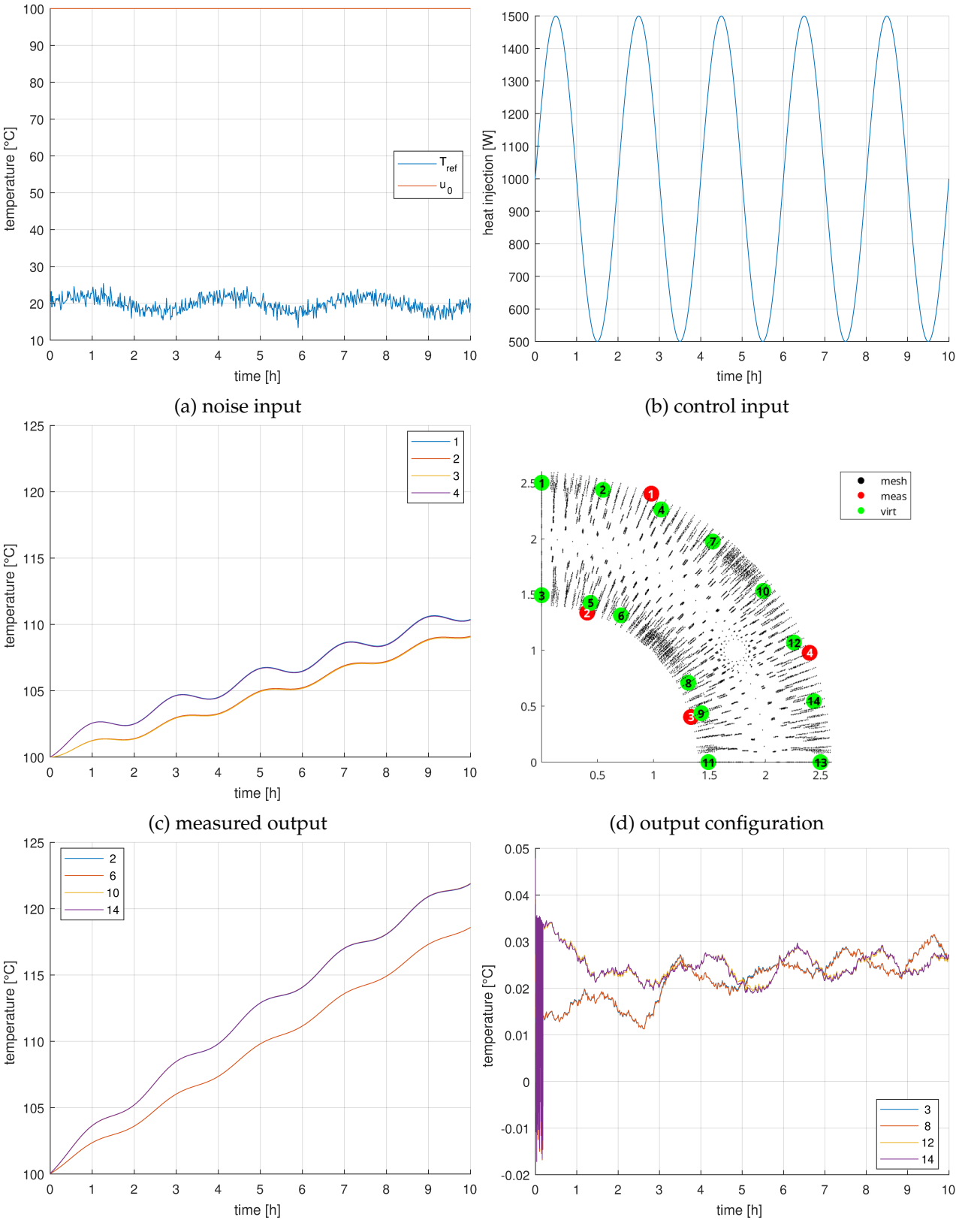
(c) measured output

(d) output configuration

(e) virtual output

(f) virtual prediction error

FIGURE 5.12: Results of *noisy* simulation using **reduced-state** observer based on RFX model. (a) and (b) show respectively noise and control input signals. (c) and (e) show measured and virtual output respectively, according to the mesh configuration shown in (d). The error of the virtual prediction against the true model is plotted in (f).



(a) noise input

(b) control input

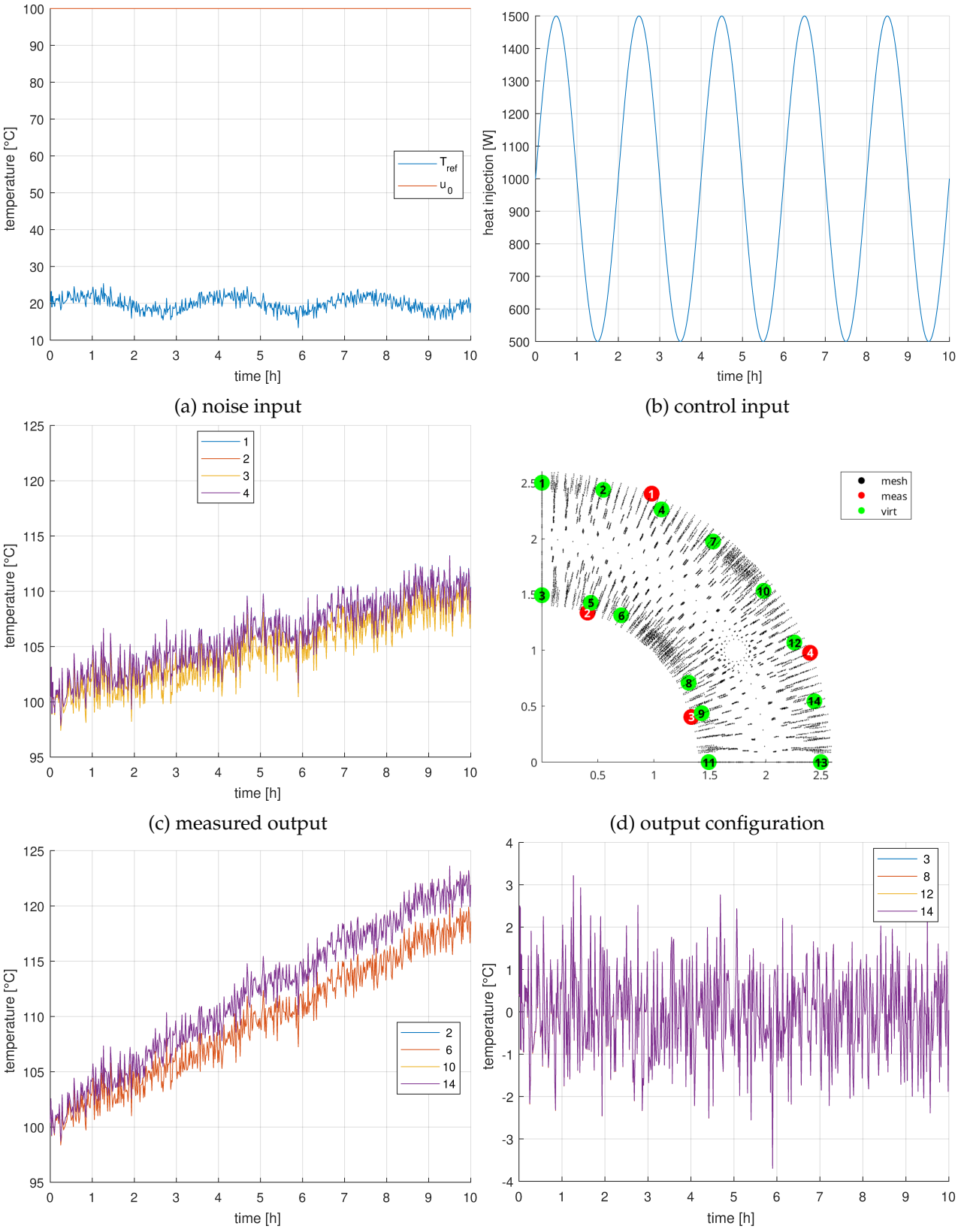
(c) measured output

(d) output configuration

(e) virtual output

(f) virtual prediction error

FIGURE 5.13: Results of *nominal* simulation using H_∞ **observer** based on RFX model. (a) and (b) show respectively noise and control input signals. (c) and (e) show measured and virtual output respectively, according to the mesh configuration shown in (d). The error of the virtual prediction against the true model is plotted in (f).



(a) noise input

(b) control input

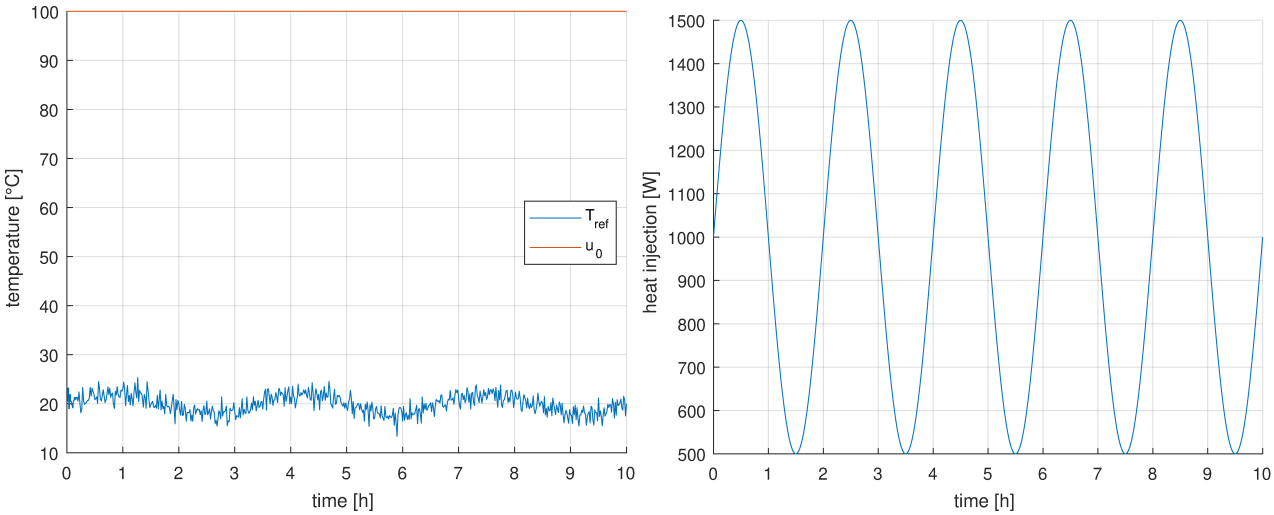
(c) measured output

(d) output configuration

(e) virtual output

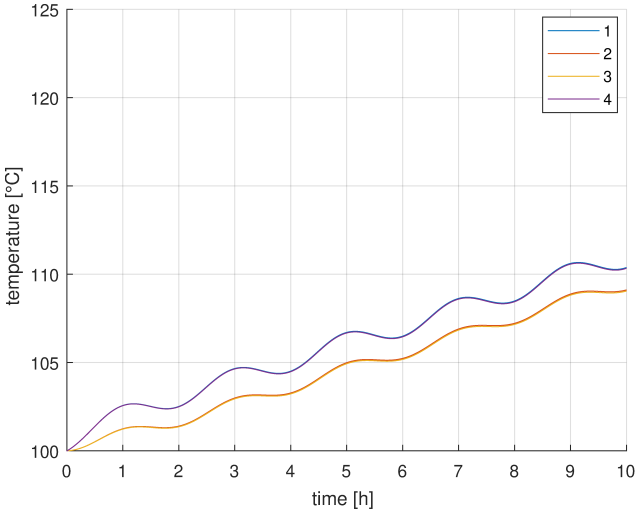
(f) virtual prediction error

FIGURE 5.14: Results of *noisy* simulation using H_∞ **observer** based on RFX model. (a) and (b) show respectively noise and control input signals. (c) and (e) show measured and virtual output respectively, according to the mesh configuration shown in (d). The error of the virtual prediction against the true model is plotted in (f).

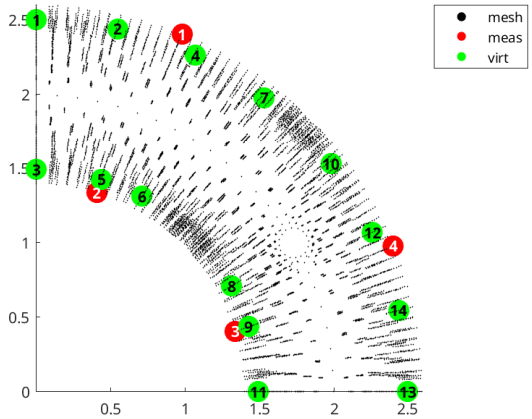


(a) noise input

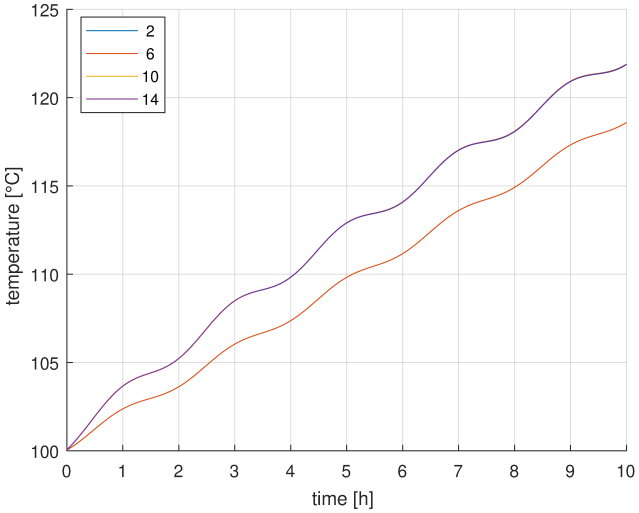
(b) control input



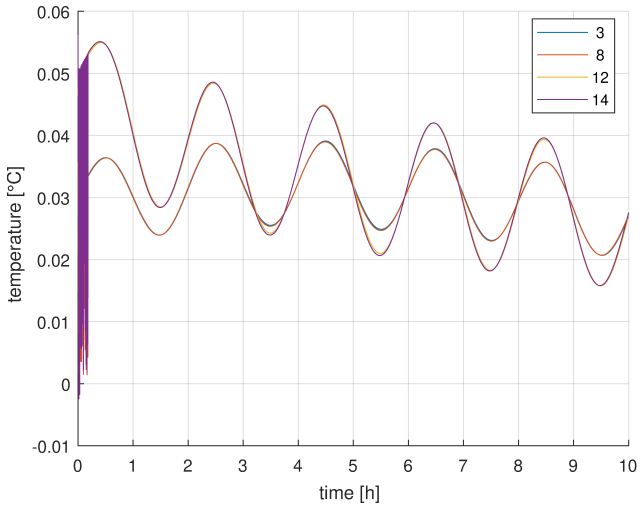
(c) measured output



(d) output configuration

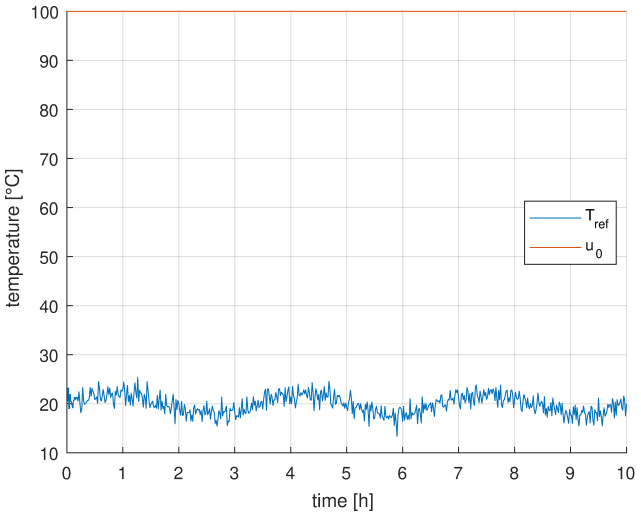


(e) virtual output

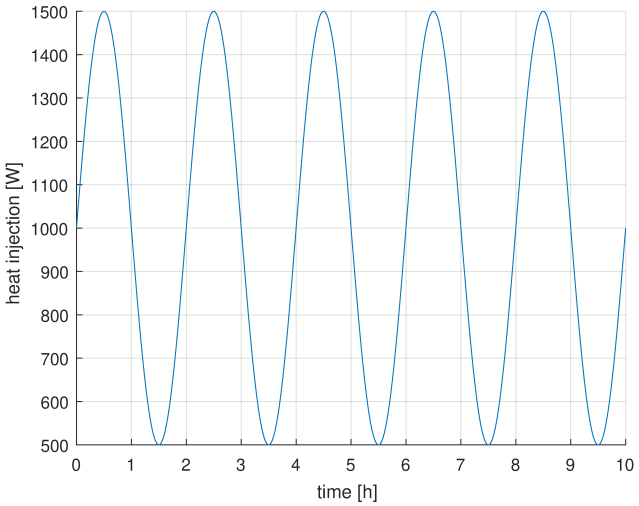


(f) virtual prediction error

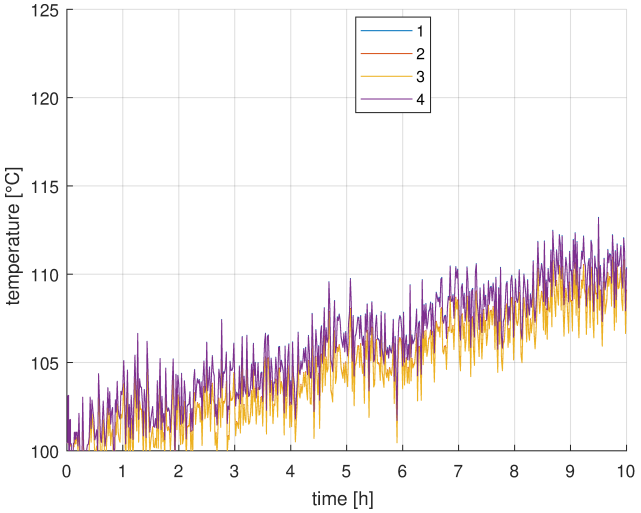
FIGURE 5.15: Results of *nominal* simulation using H_2 **observer** based on RFX model. (a) and (b) show respectively noise and control input signals. (c) and (e) show measured and virtual output respectively, according to the mesh configuration shown in (d). The error of the virtual prediction against the true model is plotted in (f).



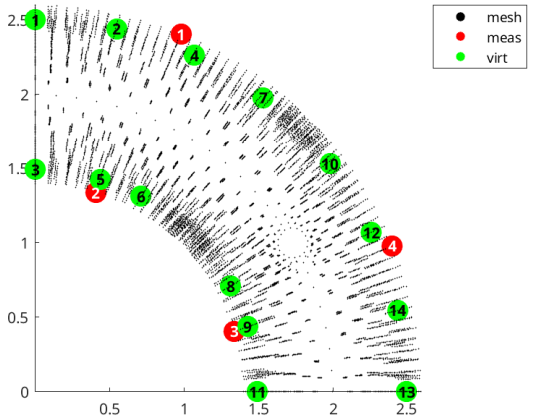
(a) noise input



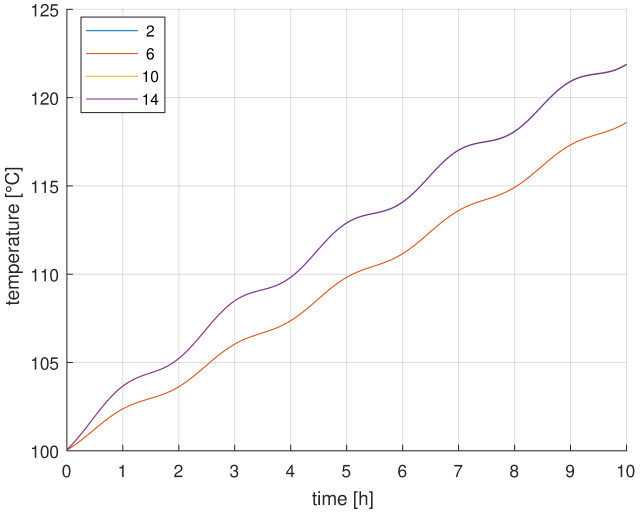
(b) control input



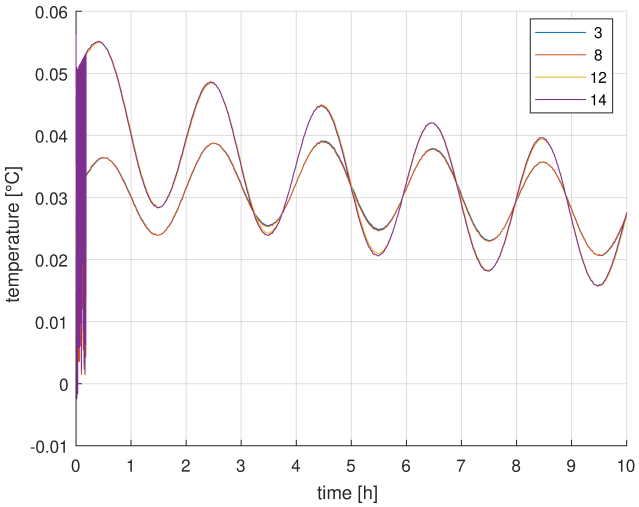
(c) measured output



(d) output configuration



(e) virtual output



(f) virtual prediction error

FIGURE 5.16: Results of *noisy* simulation using H_2 **observer** based on RFX model. (a) and (b) show respectively noise and control input signals. (c) and (e) show measured and virtual output respectively, according to the mesh configuration shown in (d). The error of the virtual prediction against the true model is plotted in (f).

Chapter 6

Preliminary considerations on control of reduced-order thermodynamic models

Contents

6.1 Missing physical value of reduced state coordinates

Suppose working observer/controller is implemented that efficiently solves the reconstruction/control problem with acceptable error. How can such control loop be monitored considering that the coordinates of the states appearing in the reduced state-space do not have the same physical meaning (temperature) of the original full-order model ? Ideally, a second machine, for monitoring operations should receive measurements of the current reduced state of the system and be able to *explain* these signals with the same physical meaning of the full-order model (formulation of the FEM thermodynamics).

A first, trivial attempt in solving this problem, is to look at the dimensions of the dynamic matrices of the *full-order* system

$$\left(A \in \mathbb{M}^n, B \in \mathbb{M}^{n\times m}, C \in \mathbb{M}^{p\times n}, D \in \mathbb{M}^{p\times m}\right)$$

and compare them with the corresponding matrices in the *reduced-order* system

$$\left(A' \in \mathbb{M}^r, B' \in \mathbb{M}^{r\times m}, C' \in \mathbb{M}^{p\times r}, D' \in \mathbb{M}^{p\times m}\right)$$

in order to find a meaningful relation. The following two propositions come from the fact that output signals and input signals are shared by both models, in fact this is a key principle for MOR methods

$$C^{\#} = \left[C^T C\right]^{\dagger} C^T \cdot C' \qquad B^{\#} = \left[BB^T\right]^{\dagger} B \cdot B'^T$$

A second, more advanced attempt on the monitoring problem, stems from the equation $AX = XB$ referred to as the *intertwining* relation [28, p.112] and the fact that projection-based MOR can be set using Sylvester equation [29], [30]. Let (A, C) and (A', C') denote the state-output pair dynamic matrices of the full-order and reduced-order model respectively, then solving either

$$A^T\mathcal{P}_y + \mathcal{P}_y A' + C^T C' = 0 \qquad A^T\mathcal{P}_x + \mathcal{P}_x A' + BB'^T = 0$$

for the projectors $\mathcal{P}_y$ or $\mathcal{P}_x$ should not only be numerically feasible (by resorting to standard numerical methods, such as [31]), but also provide a static, linear approximation to the monitor problem.

Unfortunately, due to time constraints on the duration of the present work, none of the claims in the two paragraphs above could be studied in depth, and both ideas remain to be explored in future studies.

6.2 Minimum requirements of real-time controller board

The importance of sampling time for the discrete-time, reduced-order model has been shown in FIG. 5.2. From the experience gained in the present work, It appears that a good sampling time is of the order of minutes.

Concerning the memory dimension required by a controller board, the total number of entries to be stored may be computed as

$$(\underbrace{n^2}_{A} + \underbrace{nm}_{B} + \underbrace{nn_0}_{AX_0}) + (\underbrace{pn}_{C} + \underbrace{pn_0}_{CX_0})$$

where X_0 is a basis in the state space $\mathbb{R}^n$ describing an initial condition of dimension n_0. This number is to be compared against the number of nonzero elements (nnz) in order to assess the sparsity of the matrices.

Finally, the multiplication of a matrix $\mathbb{M}_{\mathbb{R}}^{a\times b}$ by a vector in $\mathbb{R}^b$ requires b sums of vectors in $\mathbb{R}^a$ (each obtained with a multiplications) thus giving a total of $ba+a = a(b+1)$ scalar operations (sums or multiplications). For a linear dynamic system, we can summarise the number of operations as follows:

$$\begin{aligned} \dot{x} &= \overbrace{Ax}^{n(n+1)} + \overbrace{Bu}^{n(m+1)} + \overbrace{\left[AX_0\right] u_0}^{n(n_0+1)} \\ y &= \underbrace{Cx}_{p(n+1)} + \underbrace{\left[CX_0\right] u_0}_{p(n_0+1)} \end{aligned}$$

Further, after the model order reduction has been performed, more storage and computation efficiency may be gained by considering state of the art lossless compression algorithms for real-valued matrices, and in particular, for matrix-vector multiplications and inversion. Possible directions might include recent articles such like [32] and [33].

Chapter 7

Conclusion and future work

Peculiar properties of the dynamic systems generated from finite-element formulation of thermodynamic models have been analysed. It was found (TAB. 2.1) that the resulting systems tend to be extremely sparse and badly conditioned from an algebraic point of view.

These properties led to the fundamental choice of the reduction path highlighted in FIG. 3.1 , where different issues about model order reduction were addressed. The RFX themal model was reduced with several methods and the reduction based on balanced-truncation was retained for its superior performance (TAB. 3.2).

Then, the study of the reconstruction problem on full-order synthetic thermal models showed key issues in the design of traditional observers. Despite the discrete-time being observable to machine precision (FIG. 4.1), both full-state and reduced-state feedback observers featured difficult pole placement and resulted in high-gain peaking phenomena (TAB. 4.1). Moreover, their behaviour in response to noisy measurements was not acceptable.

Next, the state reconstruction problem of the balanced-truncation reduced-order RFX model was studied. Drawbacks of traditional observers were confirmed, and importance of sampling time for model observability was discovered. Two more advanced observer design procedures were implemented based on the concepts of infinite-horizon H_2 and H_∞ norm optimisation. Both showed superior prediction capabilities (TAB. 5.2) as compared to traditional observers. The former H_2 estimator was able effectively reject noise measurement (TAB. 5.3), while the latter overcame the error amplification issue found in traditional observers.

Lastly, some practical considerations about real-time micro-controller requirements were formulated and a number of ideas were presented for the problem of monitoring the the control loop.

Future work directions may include the following deviations from basic theory settings

- to consider descriptor form systems theory which may unlock more efficient simulation procedures at the cost more advanced theoretical tools such as matrix pencils.
- to test different reduction methods other than those already selected, with the possibility of preserving useful properties of the original system
- to apply discrete-time MOR methods
- to investigate the potential application of algebraic lossless compression models as a further reduction step

Concerning the observer design for state reconstruction aimed at virtual output prediction, the following should not be overseen

- develop an initial condition estimator that allows to infer the initial condition based on the first few measured samples
- implement hybrid traditional observer allowing to place only selected poles through feedback, and to estimate the rest with an open loop-observer
- study the effect of designing state observers based on models that differ from reality
- advanced optimal controller design based on linear matrix inequalities enforcing more robust properties on H_2 and H_∞ estimators [34], [35]

Finally,

- the angles between the observable subspace and the spanning subspace of the virtual output matrix may provide a useful tool for assessing numerically the difficulty of the reconstruction problem for virtual output prediction.
- the two maps proposed for the monitor problem could be investigated by solving a Sylvester equation.

- the ratio between measured and virtual outputs should be the subject of a separate study, allowing the designer to find empirical rules to place confidently a minimum number of sensor.

Bibliography

[1] G. Strang and G. J. Fix, *An Analysis of the Finite Element Method*, 2. ed. Wellesley: - Cambridge press, 2008, 402 pp., ISBN: 978-0-9802327-0-7.

[2] M. MATLAB. "PDEs Toolbox Documentation." (), [Online]. Available: https://it.mathworks.com/help/pde/index.html?s_tid=CRUX_lftnav (visited on 03/18/2024).

[3] "RFX-mod2," Consorzio RFX. (), [Online]. Available: https://www.igi.cnr.it/ricerca/magnetic-confinement-research-in-padova/rfx-mod2/ (visited on 06/30/2024).

[4] Peter Benner, Stefano Grivet-Talocia, Alfio Quarteroni, Gianluigi Rozza, and Wil Schilders, *Volume 1 System- and Data-Driven Methods and Algorithms* (Model Order Reduction), 3 vols. Berlin, Boston: De Gruyter, 2021, vol. Volume 1, ISBN: 978-3-11-049896-7. DOI: doi:10.1515/9783110498967. [Online]. Available: https://doi.org/10.1515/9783110498967 (visited on 07/15/2023).

[5] A. C. Antoulas, *Approximation of Large-Scale Dynamical Systems* (Advances in Design and Control). Philadelphia: SIAM, 2005, ISBN: 0-89871-529-6.

[6] M. Heinkenschloss, T. Reis, and A. C. Antoulas, "Balanced truncation model reduction for systems with inhomogeneous initial conditions," *Automatica*, vol. 47, no. 3, pp. 559–564, Mar. 1, 2011, ISSN: 0005-1098. DOI: 10.1016/j.automatica.2010.12.002. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0005109810004905 (visited on 04/03/2024).

[7] C. Beattie, S. Gugercin, and V. Mehrmann, "Model reduction for systems with inhomogeneous initial conditions," *Systems & Control Letters*, vol. 99, pp. 99–106, Jan. 1, 2017, ISSN: 0167-6911. DOI: 10.1016/j.sysconle.2016.11.007. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0167691116301864 (visited on 04/04/2024).

[8] U. Baur, P. Benner, and L. Feng, "Model Order Reduction for Linear and Nonlinear Systems: A System-Theoretic Perspective," *Archives of Computational Methods in Engineering*, vol. 21, no. 4, pp. 331–358, Dec. 2014, ISSN: 1134-3060, 1886-1784. DOI: 10.1007/s11831-014-9111-2. [Online]. Available: http://link.springer.com/10.1007/s11831-014-9111-2 (visited on 04/04/2024).

[9] "MORLAB - Model Order Reduction LABoratory," DOI: 10.5281/zenodo.3332716. [Online]. Available: https://zenodo.org/records/3332716 (visited on 10/25/2023).

[10] P. Benner, E. S. Quintana-Ortí, and G. Quintana-Ortí, "Balanced Truncation Model Reduction of Large-Scale Dense Systems on Parallel Computers," *Mathematical and Computer Modelling of Dynamical Systems*, vol. 6, no. 4, pp. 383–405, Dec. 2, 2000, ISSN: 1387-3954. DOI: 10.1076/mcmd.6.4.383.3658. [Online]. Available: https://www.tandfonline.com/doi/abs/10.1076/mcmd.6.4.383.3658 (visited on 04/03/2024).

[11] M. Hou and P. Muller, "Causal observability of descriptor systems," *IEEE Transactions on Automatic Control*, vol. 44, no. 1, pp. 158–163, Jan. 1999, ISSN: 1558-2523. DOI: 10.1109/9.739111. [Online]. Available: https://ieeexplore.ieee.org/document/739111 (visited on 05/05/2024).

[12] M. Hou and P. Muller, "Observer design for descriptor systems," *IEEE Transactions on Automatic Control*, vol. 44, no. 1, pp. 164–169, Jan. 1999, ISSN: 1558-2523. DOI: 10.1109/9.739112. [Online]. Available: https://ieeexplore.ieee.org/document/739112 (visited on 05/05/2024).

[13] A. Riccardo and F. Francesco, "Digital position control of a DC servomotor," Dept. of Information Engineering (DEI), University of Padova, Laboratory handout 2, Apr. 14, 2021, p. 17.

[14] G. Strang, *Linear Algebra and Learning from Data*. Wellesley: Wellesley-Cambridge press, 2019, ISBN: 978-0-692-19638-0.

[15] A. V. Knyazev and M. E. Argentati, "Principal Angles between Subspaces in an *A* -Based Scalar Product: Algorithms and Perturbation Estimates," *SIAM Journal on Scientific Computing*, vol. 23, no. 6, pp. 2008–2040, Jan. 2002, ISSN: 1064-8275, 1095-7197. DOI: 10.1137/S1064827500377332. [Online]. Available: http://epubs.siam.org/doi/10.1137/S1064827500377332 (visited on 05/10/2024).

[16] Z. Drmac, "On Principal Angles between Subspaces of Euclidean Space," *SIAM Journal on Matrix Analysis and Applications*, vol. 22, no. 1, pp. 173–194, Jan. 2000, ISSN: 0895-4798, 1095-7162. DOI: 10.1137/S0895479897320824. [Online]. Available: http://epubs.siam.org/doi/10.1137/S0895479897320824 (visited on 05/10/2024).

[17] H. K. Khalil, "High-gain observers in nonlinear feedback control," in *2008 International Conference on Control, Automation and Systems*, Oct. 2008, pp. xlvii–lvii. DOI: 10.1109/ICCAS.2008.4694705. [Online]. Available: https://ieeexplore.ieee.org/document/4694705 (visited on 05/04/2024).

[18] B. Moore, "Principal component analysis in linear systems: Controllability, observability, and model reduction," *IEEE Transactions on Automatic Control*, vol. 26, no. 1, pp. 17–32, Feb. 1981, ISSN: 0018-9286. DOI: 10.1109/TAC.1981.1102568. [Online]. Available: http://ieeexplore.ieee.org/document/1102568/ (visited on 10/31/2023).

[19] N. D. Powel and K. A. Morgansen, "Empirical observability Gramian rank condition for weak observability of nonlinear systems with control," in *2015 54th IEEE Conference on Decision and Control (CDC)*, Dec. 2015, pp. 6342–6348. DOI: 10.1109/CDC.2015.7403218. [Online]. Available: https://ieeexplore.ieee.org/document/7403218 (visited on 05/11/2024).

[20] A. J. Krener and K. Ide, "Measures of unobservability," in *Proceedings of the 48h IEEE Conference on Decision and Control (CDC) Held Jointly with 2009 28th Chinese Control Conference*, Dec. 2009, pp. 6401–6406. DOI: 10.1109/CDC.2009.5400067. [Online]. Available: https://ieeexplore.ieee.org/document/5400067 (visited on 05/14/2024).

[21] Stephen Boyd, Laurent El Ghaoui, Eric Feron, and Venkataramanan Balakrishnan, *Linear Matrix Inequalities in System and Control Theory*. SIAM, ISBN: ISBN 0-89871-334-X.

[22] E. Fornasini, *Appunti di teoria dei sistemi / E. Fornasini*. Padova: Progetto, 2015, viii+681, ISBN: 978-88-96477-32-8.

[23] R. M. Murray and K. J. Åström, *Feedback Systems: An Introduction for Scientists and Engineers*. [Online]. Available: https://fbswiki.org/wiki/index.php/Feedback_Systems:_An_Introduction_for_Scientists_and_Engineers (visited on 04/16/2024).

[24] B. Hassibi, A. H. Sayed, and T. Kailath, *Indefinite-Quadratic Estimation and Control* (Studies in Applied and Numerical Mathematics). Society for Industrial and Applied Mathematics, Jan. 1999, 564 pp., ISBN: 978-0-89871-411-1. DOI: 10.1137/1.9781611970760. [Online]. Available: https://epubs.siam.org/doi/book/10.1137/1.9781611970760 (visited on 06/18/2024).

[25] M. MATLAB. "Implicit solver for discrete-time algebraic Riccati equations - idare." (), [Online]. Available: https://it.mathworks.com/help/control/ref/idare.html (visited on 06/27/2024).

[26] Y. Pei, S. Biswas, D. S. Fussell, and K. Pingali. "An Elementary Introduction to Kalman Filtering." arXiv: 1710.04055 [cs, eess]. (Jun. 27, 2019), [Online]. Available: http://arxiv.org/abs/1710.04055 (visited on 04/16/2024), pre-published.

[27] F. Auger, M. Hilairet, J. M. Guerrero, E. Monmasson, T. Orlowska-Kowalska, and S. Katsura, "Industrial Applications of the Kalman Filter: A Review," *IEEE Transactions on Industrial Electronics*, vol. 60, no. 12, pp. 5458–5471, Dec. 2013, ISSN: 0278-0046, 1557-9948. DOI: 10.1109/TIE.2012.2236994. [Online]. Available: http://ieeexplore.ieee.org/document/6400245/ (visited on 10/13/2023).

[28] R. A. Horn and C. R. Johnson, *Matrix Analysis*, 2. ed. New York [etc: Cambridge University press, 2013, xviii+643, ISBN: 978-0-521-54823-6.

[29] V. Simoncini, "Computational Methods for Linear Matrix Equations," *SIAM Review*, vol. 58, no. 3, pp. 377–441, Jan. 2016, ISSN: 0036-1445, 1095-7200. DOI: 10.1137/130912839. [Online]. Available: http://epubs.siam.org/doi/10.1137/130912839 (visited on 12/12/2023).

[30] K. Gallivan, A. Vandendorpe, and P. Van Dooren, "Sylvester equations and projection-based model reduction," *Journal of Computational and Applied Mathematics*, Proceedings of the International Conference on Linear Algebra and Arithmetic 2001, vol. 162, no. 1, pp. 213–229, Jan. 1, 2004, ISSN: 0377-0427. DOI: 10.1016/j.cam.2003.08.026. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0377042703007143 (visited on 05/13/2024).

[31] J. Saak, M. Köhler, and P. Benner, *M-M.E.S.S.-3.0 – the matrix equations sparse solvers library*, Aug. 2023. DOI: 10.5281/zenodo.7701424.

[32] P. Ferragina, T. Gagie, D. Köppl, *et al.* "Improving Matrix-vector Multiplication via Lossless Grammar-Compressed Matrices." arXiv: 2203.14540 [cs]. (Mar. 30, 2022), [Online]. Available: http://arxiv.org/abs/2203.14540 (visited on 04/05/2024), pre-published.

[33] A. Elgohary, M. Boehm, P. J. Haas, F. R. Reiss, and B. Reinwald, "Compressed Linear Algebra for Large-Scale Machine Learning,"

[34] X. Lihua, Y. C. Soh, D. Chunling, and Y. Zou, "Robust H2 estimation and control," *Journal of Control Theory and Applications*, vol. 2, no. 1, pp. 20–26, Feb. 1, 2004, ISSN: 1993-0623. DOI: 10.1007/s11768-004-0019-5. [Online]. Available: https://doi.org/10.1007/s11768-004-0019-5 (visited on 06/15/2024).

[35] L. Xie and E. de Souza Carlos, "Robust H/sub infinity / control for linear systems with norm-bounded time-varying uncertainty," *IEEE Transactions on Automatic Control*, vol. 37, no. 8, pp. 1188–1191, 1992. DOI: 10.1109/9.151101.

Appendix A

Codebase and algorithms

Contents

A.1 Repository conventions and overview

The entire code is available in a git repository at gitlab.dei.unipd.it. The scripts have been developed using MATLAB revision 2023b and using primarily the `pdetoolbox` [2] and `MORlab` [9] toolbox.

A number of progressively numbered folders have been stored as the scripts for this project were developed. The progressive number corresponds to a date expressed in the `YYMMDD` format, where `Y`,`M`,`D` denote year, month, and day (respectively).

A structured file name convention has been used for naming the `.mat` files resulting from the computation of the dynamical state-space systems in the steps defined below.

$$\diamondsuit(\mathrm{n}-\mathrm{m}-\mathrm{p})\spadesuit(\mathrm{r}-\mathrm{m}-\mathrm{p})\#\heartsuit(\mathrm{n}'-\mathrm{m}'-\mathrm{p}')\_\clubsuit.\mathrm{mat}$$

where:

- $\diamondsuit \in \{\mathrm{D}, \mathrm{S}, \ldots\}$ identifies the dynamic formulation of the first model. `D` represents descriptor dynamic form, while `S` denotes a standard dynamic form. Most of the models start from descriptor dynamic form. The following (n, m, p) denote the size of the system's state, input, and output (respectively).
- $\spadesuit \in \{\mathrm{bt}, \mathrm{hna}, \ldots\}$ corresponds to the short name abbreviation of the
- $\#$ identifies when the transformation from descriptor system to standard dynamic form took place (inversion of the mass matrix E). The following (r, m, p) describe the size of the reduced system's state, input, and output (respectively).
- $\heartsuit \in \{\mathrm{C}, \mathrm{O}, \ldots\}$ is a short hand for describing either control-type transformations (letter `C`) or observer design (letter `O`). The following (n', m', p') allow to note down the size of the dynamic system after such operation
- $\clubsuit$ represents a final alias given to similar models in order to distinguish the content of the `.mat` file more easily

Most importantly, the proposed naming convention exploits the following:

- the order of the elements of the file name may change depending on the order of the computations, see discussion about FIG. 3.1
- missing elements correspond to missing operations and have a meaning in describing the workflows that has been followed for computing a particular model.
- the final alias description is very often neglected, especially when there is no conflict in the naming of the files.

The overall workflow to reproduce the results of this thesis can be split in six steps:

1. Creation of the full-order, continuous-time, LTI state-space model in descriptor dynamic form, this includes the definition of an output matrix
2. Computation of the reduced model and assessment of the reduction quality
3. general pre-processing required to transform the system to a discrete-time state-space model in standard dynamic form. Formatting on the noise input may be applied, see SEC. 4.1.1
4. Design of state observers
5. block-scheme initialisation and simulation run
6. post-processing required to compute properties of the error and plot selected signals

Gained experience with long simulations containing both continuous-time and discrete-time models, suggests to:

- leave Simulink solver to all *auto/default* values and to impose discrete-time operations with explicit discrete-time blocks and/or zero-order hold.
- prefer data-inspector to traditional scopes and limit the amount of data stored by logged signals by tweaking the settings
- limit the number of simulated continuous-systems as they tend to slow down the overall simulation performance.

Most of the scripts described in the following algorithms are meant to take a dynamic system as an input, to perform some computations, and to store a new dynamic system as output. During such computations, however it may be necessary to assess quantities and check hypotheses, and for this reason the scripts stores a text file with `.log` extension in order to save error messages, and user choices. Plots of various kind that are computed during the execution of the script may be saved to `.pdf` file through the use of the global, boolean, variable `EXPORT`. While the logging procedure appends information to any existing log, the export task overwrites any existing file matching the same name without warning the user.

A.2 Working with full-order models based on 2D mesh

Algorithm 1: Description (one) of (possible) script `OUTFOLDER/init.m`

Input : filenames containing a structure with LTI matrices (A, B, C, D) of the dynamic systems in the simulation scheme

Input : $(J_m, J_v) \leftarrow$ row indexes corresponding to measured and virtual outputs (respectively)

Output: no output

`// true models`

`DsysCt` $\leftarrow$ ground truth dynamic system in descriptor form ;

`SsysCt` $\leftarrow$ continuous-time dynamic system in standard form;

`SsysDt` $\leftarrow$ discrete-time dynamic system in standard form;

`// (`J_m, J_v`) are used to replicate true signals for measured and virtual outputs`

`fullObs` $\leftarrow$ load full-state feedback observer ; `// optional`

`redObs` $\leftarrow$ load reduced-state feedback observer ; `// optional`

`HooObsDt` $\leftarrow$ load H_∞ sub-optimal estimator; `// optional`

`H2KObsDt` $\leftarrow$ load H_2 Kalman estimator; `// optional`

`// simulation configuration`

Warning *display solver name and type*;

$t_s \leftarrow$ sampling-time for discrete-time systems;

$t_{\text{end}} \leftarrow$ simulation stop time in seconds;

`// input signals`

$T_{\text{ref}}(\bullet) \leftarrow$ reference temperature (external ambient) modelled as noise input;

$u_0 \leftarrow$ initial condition amplitude with respect to the spanning basis X_0 used in the models, $x(0) = X_0 u_0$;

$u(\bullet) \leftarrow$ remaining signals in the input matrix, modelled as control input;

`// blocks initial conditions`

`fullObs.x0` $\leftarrow$ observer initial condition ; `// optional`

`redObs.x0` $\leftarrow$ observer initial condition ; `// optional`

`HooObs.x0` $\leftarrow$ observer initial condition ; `// optional`

`H2KObs.x0` $\leftarrow$ observer initial condition ; `// optional`

A.2.1 Algorithms for the creation of dynamic models

Theory formulas for ALG. 2 and ALG. 3 can be found in SEC. 2.2

A.2.2 Algorithms for computing traditional observers

Theory formulas for ALG. 5 used by ALG. 4 can be found in SEC. 4.2 which uses function

Algorithm 2: Description of script `makePDEnd.m`

Input : λ thermal, specific, conductivity
Input : ρc density by specific heat
Input : $h_{\max}$ maximum allowed distance between adjacent mesh nodes
Input : `gd,sf,ns` domain geometry definition from pdetoolbox
Output: `fem` structure containing finite-element matrices $(M \in \mathbb{M}^n, H \in \mathbb{M}^n, K \in \mathbb{M}^n, G \in \mathbb{M}^{n \times m}, Q \in \mathbb{M}^n)$ assembled form 2D geometry

1 `pdeModel`= initialize partial, differential, equation model;
2 `geo` $\leftarrow$ `decsg(`*gd*`,`*sf*`,`*ns*`);`
3 `geometryFromEdges(`*pdeModel*`,`*geo*`);`
4 **Plot** *loaded geometry displaying edge labels*;
5 **Choose** *edge indexes* $\mathcal{E}_n$ *where to apply Neumann-type boundary conditions*;
6 `applyBoundaryCondition(`*pdeModel*`,`*Neumann*`,` $\mathcal{E}_n$`);`
7 **Choose** *edge indexes* $\mathcal{E}_d$ *where to apply Dirichlet-type boundary conditions*;
8 `applyBoundaryCondition(`*pdeModel*`,`*Dirichlet*`,` $\mathcal{E}_d$`);`
9 `pdeModel` $\leftarrow$ specify coefficients using λ and ρc according to eq. (2.2);
10 `generateMesh(`*pdeModel*`,` h_{max}`);`
11 `fem` = `assembleFEMatrices(`*pdeModel*`)`

Theory formulas for ALG. 7 used by ALG. 6 can be found in SEC. 4.3

which uses function

Algorithm 3: Description of script `dynamicModel.m`

Input : `fem` structure containing finite-element matrices $(M \in \mathbb{M}^n, H \in \mathbb{M}^n, K \in \mathbb{M}^n, G \in \mathbb{M}^{n\times m}, Q \in \mathbb{M}^n)$ assembled form 2D mesh

Output: Σ structure containing matrices $(E \in \mathbb{M}^n, A \in \mathbb{M}^n, B \in \mathbb{M}^{n\times m}, C \in \mathbb{M}^{p+q\times n}, D \in \mathbb{M}^{p+q\times m})$ of the LTI system in descriptor dynamic form

`// handle Dirichlet boundary conditions`

$(T, O_r) \leftarrow$ `kerSpanBasis(`H`)`

Choose X_0 *basis vector spanning the initial condition in the nodes space* $x(0) = X_0 u_0$ *according to eq.* (3.1);

Let n *denote the number of nodes in the 2D mesh, and* m *denote the number of input signals comprising modelled noise and initial condition*;

$U_d \leftarrow$ contribution of Dirichlet inputs;

$(E \in \mathbb{M}^n, A \in \mathbb{M}^n, B \in \mathbb{M}^{n\times m}) \leftarrow$ compute state dynamics according to eq. (2.3);

Let p *denote the number of measured output signals, and* q *denote the number of virtual outputs to be predicted*;

`// define measured output (nodes average)`

Choose $\{x_i\}_1^p$ *centers for measurement sensors*;

Choose R *radius for circular selection of nodes*;

$\mathcal{I}_m \leftarrow$ set of p indexes resulting from the intersection of all mesh nodes with circles of radius R and centers x_i;

Plot *circular patches and nodes to be considered as measured output against full 3D mesh*;

`// define virtual output (nodes average)`

Choose $\{x_i\}_1^q$ *centers for virtual sensors*;

Choose R *radius for circular selection of nodes*;

$\mathcal{I}_v \leftarrow$ set of q indexes resulting from the intersection of all mesh nodes with circles of radius R and centers x_i;

Plot *circular patches and nodes to be considered as virtual output against full 3D mesh*;

Save *resulting output node selection against full 3D mesh*;

$(C^m \in \mathbb{M}^{p\times n}, C^v \in \mathbb{M}^{q\times n}) \leftarrow$ initialize sparse matrices;

Let n_i *be the number of nodes inside the* $i-$*th circular patch with center* x_i;

$C = \begin{bmatrix} C^v \\ C^m \end{bmatrix}$ where $C^v_{[i,\mathcal{I}_v]} = \frac{1}{n_i}\ i = 1, \ldots, q \qquad C^m_{[i,\mathcal{I}_m]} = \frac{1}{n_i}\ i = 1, \ldots, p$;

$D \leftarrow$ contribution of initial condition, according to eq. (3.1);

Algorithm 4: Description of script `designFullstateAsymptoticObserver.m`

Input : Σ continuous-time system in standard dynamic form
Input : Σ_d corresponding discrete-time system
Output: `obsCt` structure containing dynamic matrices of continuous-time asymptotic observer
Output: `obsDt` structure containing corresponding dynamic matrices of discrete-time asymptotic observer

$J_v \leftarrow$ row indexes of the output matrix corresponding to *virtual* output ;
$J_m \leftarrow$ row indexes of the output matrix corresponding to *measured* output ;

`// inspect properties of the original systems`
info $\leftarrow$ `studyObservability(`Σ, J_m, J_v`);`
info $\leftarrow$ `studyObservability(`Σ_d, J_m, J_v`);`
Save *plot observability against numerical tolerance*;

`// compute continuous-time observer`
Choose *desired pole locations* `eeigs`;
`obsCt` $\leftarrow$ `fullstateFeedbackObserver(`Σ, *eeigs*, J_m, J_v`);`
Save *plot*

`// compute corresponding discrete-time observer`
Choose *desired pole locations* `eeigz`;
`obsDt` $\leftarrow$ `fullstateFeedbackObserver(`Σ_d, *eeigz*, J_m, J_v`);`
Save *plot*

`// check properties of the resulting observer systems`
info $\leftarrow$ `studyReachability(`*obsCt*`);`
info $\leftarrow$ `studyReachability(`*obsDt*`);`
Save *plot reachability against numerical tolerance*;

Algorithm 5: Description of function `fullstateFeedbackObserver.m`

Input : `stdSys` containing matrices (A, B, C, D) of LTI system in standard dynamic form
Input : `eigVals` specified pole locations for the eigenvalue allocation problem
Input : J_m list of row indexes in the output matrix corresponding to measured signals
Input : J_v list of row indexes in the output matrix corresponding to virtual prediction
Output: `obs` structure containing dynamic matrices of allocated asymptotic observer

1 **Plot** *original eigenvalues of A against desired specification in `eigVals`;*
2 **Let** (C_m, D_m) *denote matrices of the measured output equation $y_m = C_m x + D_m u$;*
3 **Let** (C_v, D_v) *denote matrices of the virtual output equation $y_v = C_v x + D_v u$;*
4 $L \leftarrow$ `allocationSolver(`A, C_m`, eigVals);`
5 **Warning** *show min-max bounds of high-gain entries in L;*
`// observer block scheme matrices`
6 `obs.A, obs.B, obs.C, obs.D` $\leftarrow$ compute observer dynamic as in eq. (4.1) using measurements y_m and predicting y_v as output;
7 **Plot** *resulting observer poles in `obs.A` against original and specified eigenvalues;*
8 **Plot** *zeros of `obs` system;*

Algorithm 6: Description of script `designReducedstateAsymptoticObserver.m`

Input : Σ continuous-time system in standard dynamic form
Input : Σ_d corresponding discrete-time system
Output: `obsCt` structure containing dynamic matrices of continuous-time asymptotic observer
Output: `obsDt` structure containing corresponding dynamic matrices of discrete-time asymptotic observer

$J_v \leftarrow$ row indexes of the output matrix corresponding to *virtual* output ;
$J_m \leftarrow$ row indexes of the output matrix corresponding to *measured* output ;

`// inspect properties of the original systems`
info $\leftarrow$ `studyObservability(`Σ, J_m, J_v`);`
info $\leftarrow$ `studyObservability(`Σ_d, J_m, J_v`);`
Save *plot observability against numerical tolerance*;

`// compute continuous-time observer`
Choose *basis vectors* V *defining the reduced space*;
Assert *that the resulting state space transform* $T^{-1} = \begin{bmatrix} V \\ C_m \end{bmatrix}$ *is square full-rank*;
Choose *desired pole locations* `eeigs`;
`obsCt` $\leftarrow$ `reducedstateFeedbackObserver(`Σ, `eeigs`, J_m, J_v, V`);`
Save *plot*;

`// compute corresponding discrete-time observer`
Choose *basis vectors* V *defining the reduced space*;
Assert *that the resulting state space transform* $T^{-1} = \begin{bmatrix} V \\ C_m \end{bmatrix}$ *is square full-rank*;
Choose *desired pole locations* `eeigz`;
`obsDt` $\leftarrow$ `reducedstateFeedbackObserver(`Σ_d, `eeigz`, J_m, J_v, V`);`
Save *plot*

`// check properties of the resulting observer systems`
info $\leftarrow$ `studyReachability(obsCt);`
info $\leftarrow$ `studyReachability(obsDt);`
Save *plot reachability against numerical tolerance*;

Algorithm 7: Description of function `reducedstateFeedbackObserver.m`

Input : `stdSys` containing matrices $(A \in \mathbb{M}^n, B \in \mathbb{M}^{n\times m}, C \in \mathbb{M}^{p+q\times n}, D \in \mathbb{M}^{p+q\times m})$ of LTI system in standard dynamic form

Input : `eigVals` $\in \mathbb{R}^{n-p}$ specified pole locations for the eigenvalue allocation problem in the reduced state space

Input : J_m list of row indexes in the output matrix corresponding to measured signals

Input : J_v list of row indexes in the output matrix corresponding to virtual prediction

Input : $V \in \mathbb{M}^{n-p\times n}$ row basis vector defining a basis for the reduced state space

Output: `obs` structure containing dynamic matrices of allocated asymptotic observer

1 **Let** $(C_m \in \mathbb{M}^{p\times n}, D_m \in \mathbb{M}^{p\times m})$ *denote matrices of the measured output equation* $y_m = C_m x + D_m u$;

2 **Let** $(C_v \in \mathbb{M}^{q\times n}, D_v \in \mathbb{M}^{q\times m})$ *denote matrices of the virtual output equation* $y_v = C_v x + D_v u$;

`// state space reduction transformation`

3 $\left(\begin{bmatrix} A_{[11]} & A_{[12]} \\ A_{[21]} & A_{[22]} \end{bmatrix}, \begin{bmatrix} B_{[1]} \\ B_{[2]} \end{bmatrix}\right) \leftarrow$ compute partitioned dynamics in the reduced state-space according to eq. (4.2);

4 **Plot** *original eigenvalues of A against desired specification in* `eigVals`;

5 $L \leftarrow$ `allocationSolver(`$A_{[11]}, A_{[21]}$`, eigVals);`

6 **Warning** *show min-max bounds of high-gain entries in L;*

`// observer block scheme matrices`

7 `obs.A, obs.B, obs.C, obs.D` $\leftarrow$ compute observer dynamic as in eq. (4.3) using measurements y_m and predicting y_v as output;

8 **Plot** *resulting observer poles in* `obs.A` *against original and specified eigenvalues*;

9 **Plot** *zeros of* `obs` *system*;

A.3 Working with reduced-order models based on RFX machine

A.3.1 Algorithms for the creation of dynamic models

Theory reference formulas for ALG. 8 can be found in SEC. 2.3.1

Algorithm 8: Description of script `dynamicModel_RFX.m`

Input : `femRFX` structure containing finite-element matrices $(M \in \mathbb{M}^n, H \in \mathbb{M}^n, K \in \mathbb{M}^n, q_{\text{conv}} \in \mathbb{R}^n, q \in \mathbb{R}^n)$ assembled form RFX mesh

Output: Σ structure containing matrices $(E \in \mathbb{M}^n, A \in \mathbb{M}^n, B \in \mathbb{M}^{n\times m}, C \in \mathbb{M}^{p+q\times n}, D \in \mathbb{M}^{p+q\times m})$ of the LTI system in descriptor dynamic form

`// Dirichlet boundary conditions are not considered for RFX model`

Choose X_0 *basis vector spanning the initial condition in the nodes space* $x(0) = X_0 u_0$ *according to eq.* (3.1);

Let n *denote the number of nodes in the 3D mesh, and* m *denote the number of input signals comprising modelled noise and initial condition*;

$(E \in \mathbb{M}^n, A \in \mathbb{M}^n, B \in \mathbb{M}^{n\times m}) \leftarrow$ compute state dynamics according to eq. (2.1);

`locs` $\leftarrow$ manual definition of possible nodes representing measurement sensor and virtual signals;

Let p *denote the number of measured output signals, and* q *denote the number of virtual outputs to be predicted*;

`// define measured output`

Choose $\{\theta_i\}_1^p$ *corresponding to desired locations of measurement sensors*;

$\mathcal{I}_m \leftarrow$ set of p indexes resulting from the intersection of possible locations `locs` and selected angles θ_i;

Plot *nodes to be considered as measured output against full 3D mesh*;

`// define virtual output`

Choose $\{\theta_i\}_1^q$ *corresponding to desired locations of virtual sensors*;

$\mathcal{I}_v \leftarrow$ set of q indexes resulting from the intersection of possible locations `locs` and selected angles θ_i;

Plot *nodes to be considered as virtual output against full 3D mesh*;

Save *resulting output node selection against full 3D mesh*;

$(C^m \in \mathbb{M}^{p\times n}, C^v \in \mathbb{M}^{q\times n}) \leftarrow$ initialize sparse matrices;

$C = \begin{bmatrix} C^v \\ C^m \end{bmatrix}$ where $C^v_{[\mathcal{I}_v]} = 1 \qquad C^m_{[\mathcal{I}_m]} = 1$;

$D \leftarrow$ contribution of initial condition, according to eq. (3.1);

A.3.2 Algorithms for computing infinite-horizon observers

Theory reference formulas for ALG. 9 can be found in SEC. 5.3

Theory reference formulas for ALG. 10 can be found in SEC. 5.2

Algorithm 9: Description of script `designHinftyObserver.m`

Input : Σ_d discrete-time system in standard dynamic form
Output: `obs` structure containing dynamic matrices of infinite-horizon, H_∞ sub-optimal, discrete-time estimator

$I_c \leftarrow$ columns indexes of the input matrix to be considered ;
$J_v \leftarrow$ row indexes of the output matrix corresponding to *virtual* output ;
$J_m \leftarrow$ row indexes of the output matrix corresponding to *measured* output ;

$(F, G, H, L) \leftarrow$ dynamic matrices of Σ_d in strictly proper form according to eq. (5.1);
Let n, m denote the resulting state and input size (respectively);
Let p, q denote the number of measured and virtual outputs (respectively);

`// inspect properties of the original system`
info $\leftarrow$ `studyReachability(`Σ_d, J_m, J_v`)`;
Save *plot reachability against numerical tolerance*;
info $\leftarrow$ `studyObservability(`Σ_d, J_m, J_v`)`;
Save *plot observability against numerical tolerance*;

Choose γ suboptimal H_∞ norm bound;
$(E, A, Q, B, S, R) \leftarrow$ transpose notation and compute arguments for Riccati solver, as defined in eq. (5.5);

`// Check hypotheses for DARE solver`
Assert *necessary condition:* full rank reachability of pair (A, B);
Warning *condition number of matrices E and R*;
Warning *sufficient condition:* $\begin{bmatrix} B \\ S \\ R \end{bmatrix}$ has column rank `p+q`;
Warning *sufficient condition:* $\begin{bmatrix} Q & S \\ S^T & R \end{bmatrix} \geq 0$ by computing its eigenvalues ;
Warning *sufficient condition:* observability matrix of pair $\left(A - BR^{-1}S^T, Q - SR^{-1}S^T\right)$ has rank `n` ;

$P, K,$ `eeigz` $=$ `DAREsolver(`A, B, Q, R, S, E`)`;
Warning *check if solver found an accurate solution*;

`// check conditions for infinite-horizon observer`
Assert *stability of the closed loop:* $|p| < 1 \quad \forall p \in$ `eeigz`;
Assert *matrices (R_e, R) have the same inertia:* compute and compare positive, negative, and zero eigenvalues;
Assert *positive semi-definite character of solution:* $P \geq 0$ by computing its eigenvalues;

`// if all conditions are met, compute central estimator`
`obs.A, obs.B, obs.C, obs.D` $\leftarrow$ compute dynamic system with stabilizing gain $K_P = K^T$, according to eq. (5.6);
`// check poles and zeros of the observer dynamic system`
`zz, pp` $\leftarrow$ compute zeros and poles of `obs` dynamic system;
Save *plot of `zz, pp` on the complex plane*

Algorithm 10: Description of script `designH2KalmanObserver.m`

Input : Σ_d discrete-time system in standard dynamic form
Output: `obs` structure containing dynamic matrices of infinite-horizon, H_2 optimal, discrete-time Kalman estimator

$I_c \leftarrow$ columns indexes of the input matrix to be considered ;
$J_v \leftarrow$ row indexes of the output matrix corresponding to *virtual* output ;
$J_m \leftarrow$ row indexes of the output matrix corresponding to *measured* output ;

$(F, G, H, L) \leftarrow$ dynamic matrices of Σ_d in strictly proper form according to eq. (5.1);
Let n, m denote the resulting state and input size (respectively);
Let p denote the number of measured outputs;

`// define variance matrices on process and measurement noise`
Choose *noise variance matrix* $Q \in \mathbb{M}^n$;
Choose *noise variance matrix* $R \in \mathbb{M}^p$;
Assert *positive-definite character of* Q*:* $\lambda \geq 0 \forall \lambda \in \sigma(Q)$;
Assert *positive character of* R;

`// inspect properties of the original system`
info $\leftarrow$ `studyReachability(`Σ_d, J_m, J_v`);`
Save *plot reachability against numerical tolerance*;
info $\leftarrow$ `studyObservability(`Σ_d, J_m, J_v`);`
Save *plot observability against numerical tolerance*;

$(E, A, Q, B, S, R) \leftarrow$ transpose notation and compute arguments for Riccati solver, as defined in eq. (5.3)

`// Check hypotheses for DARE solver`
Assert *necessary condition:* full rank reachability of pair (A, B);
Warning *condition number of matrices* E *and* R;
Warning *sufficient condition:* $\begin{bmatrix} B \\ S \\ R \end{bmatrix}$ has column rank `p+q`;
Warning *sufficient condition:* $\begin{bmatrix} Q & S \\ S^T & R \end{bmatrix} \geq 0$ by computing its eigenvalues ;
Warning *sufficient condition:* observability matrix of pair $\left(A - BR^{-1}S^T, Q - SR^{-1}S^T\right)$ has rank `n` ;

$P, K,$ `eeigz` $=$ `DAREsolver(`A, B, Q, R, S, E`);`
Warning *check if solver found an accurate solution*;

`// check conditions for infinite-horizon observer`
;
Assert *stability of the closed loop:* $|p| < 1 \quad \forall p \in$ `eeigz`;
Warning *compute corresponding continuous-time poles*

if all conditions are met, compute estimator;
`obs.A, obs.B, obs.C, obs.D` $\leftarrow$ compute dynamic system according to eq. (5.4);

check poles and zeros of the resulting observer dynamic system;
`zz, pp` $\leftarrow$ compute zeros and poles of `obs` dynamic system;
Save *plot of* `zz, pp` *on the complex plane*

Appendix B

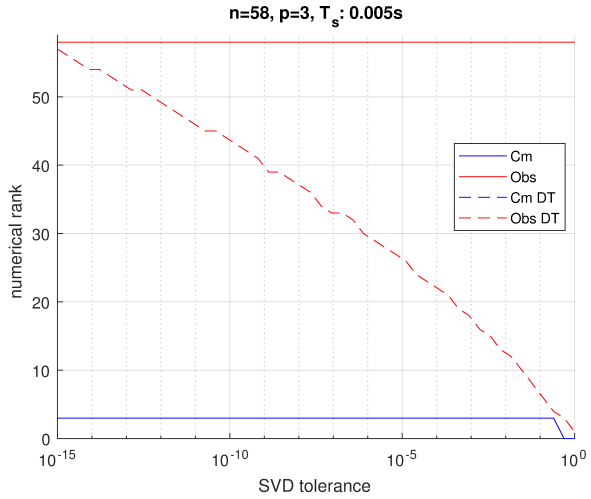
More simulation results

Contents

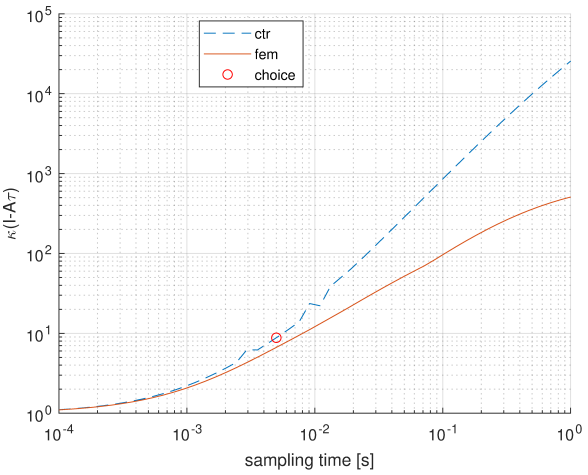
B.1 Full-order models based on 2D mesh

Initial exploration on bad behaviour of full-order models based on 2D mesh started as a search for finding a ratio between the number of outputs and the dimension of the state space, that lead to unacceptable observer response, if any. With the workflow described in SEC. 2.2 implemented in ALG. 2, this study can be asily implemented by tweaking the size of the maximum allowed mesh element, as this indirectly controls the size of the resulting state space for the assembled dynamics. The number of outputs is kept constant.

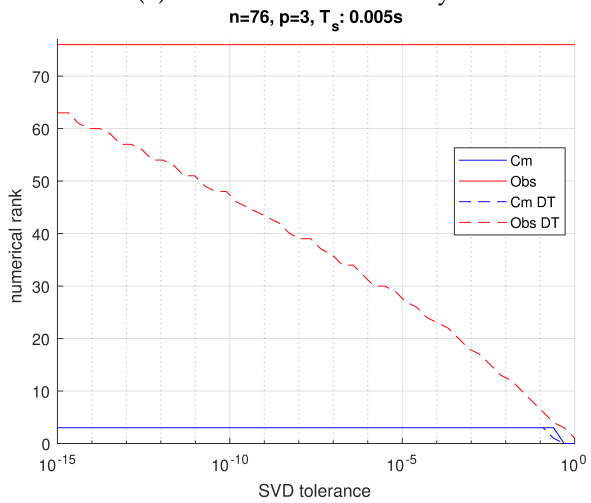
FIG. B.1 shows why after certain threshold, design of traditional observes, such as the ones proposed in SEC. 4.2 and SEC. 4.3, fails without being able to place any of the system's poles. As the size of the state space increases, the observability of the discrete-time system is lost with respect to numerical precision, which can be assumed to be of about 1×10^{-16}.



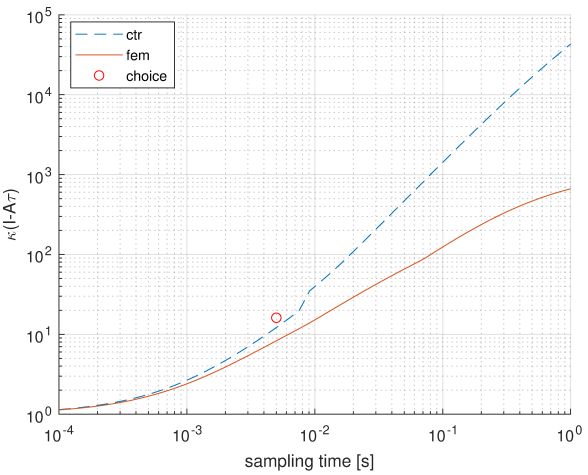
(a) numerical observability



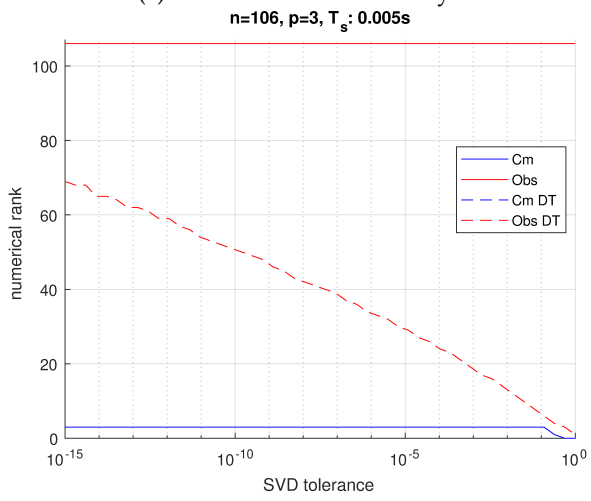
(b) sampling conditioning



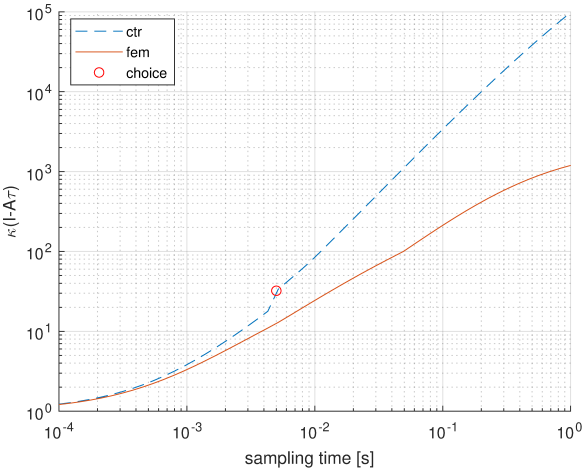
(c) numerical observability



(d) sampling conditioning



(e) numerical observability



(f) sampling conditioning

FIGURE B.1: Bad conditions of full-order model based on 2D model. As the size of the state is increased too much (by lowering the size of mesh elements), the DT model loses observability property, this can be seen in (a), (c), and (e). On the other hand (b), (d), and (f) show that the size of the state space does not affect the conditioning of the map $(I - A\tau)$ required by Euler-Backwards method eq. (3.2)

B.2 Reduced-order models based on RFX machine

B.2.1 Response to step input heat injection

On real operation scenarios, the expected heat injection profile is that of a step signal rather than a sinusoid like it was discussed in SEC. 4.4 and SEC. 5.4. The reason for the choice of the sinusoid the main results of this work was to force major excitation on error profile so to characterise the precision of the estimators in a more difficult operation senario.

FIG. B.2 shows the result of the simulation of block scheme FIG. 5.4 with constant $1.5\,\mathrm{kW}$ heat injection. and bias noise of about $3\,^{\circ}\mathrm{C}$ in both the ambient temperature T_{ref} and in the measured output sensors y_m. The observer is the same H_∞ sub-optimal estimator described in FIG. 5.7. It can be seen that properties observed in SEC. 5.4 are confirmed and the acceptable behaviour of the estimator is presented.

B.2.2 Error with respect to variations of original model

When designing any type of observer/controller, the precision of the resulting application can only be as good as the original model compared to reality. In other words, it the design of any of the observers described on this work starts with a bad original full-order model, it is unlikely that the resulting control loop is accurate. In practice, this may be tweaked depending on the robustness properties featured by the estimator at hand. Figures below, show the behaviour of FIG. 5.4 when *bias* noise is injected in input signals T_{ref}, u_0, and y_m.

FIG. B.3, shows the simulation result of H_∞ observer described in FIG. 5.7 to bias T_{ref} noise. The clean T_{ref} signal is **not** plotted in sub-quadrant (a). We see that that the estimator well behaves in the $10\,\mathrm{h}$ time span but shows the development of a drift that could lead to asymptotic estimation error within days.

FIG. B.4 depicts the response of H_2 estimator described in FIG. 5.8 o bias error in all measurement outputs. The clean measurement signals are **not** shown in sub-quadrant (c). We observe robustness in the proposed observer with respect to this kind of noise. In particular, the bias error introduced in the measurement could be seen as a difference between the original full-order model and reality, there the true dynamic matrices would differ from those used for the design of the observer.

Finally, FIG. B.5, reports the response of the same H_2 estimator described in FIG. 5.8 to bias error added to the initial condition coordinate eq. (3.1). It can be seen that the observer is not robust to this kink of noise, and this behaviour is representative of that of all observers studied in this work. For this reason the precision of the estimate of the initial condition, which is the only non-measurable, modelled, noise input of the reconstruction problem, is of paramount importance. This important aspect should be the object of detailed research in future works.

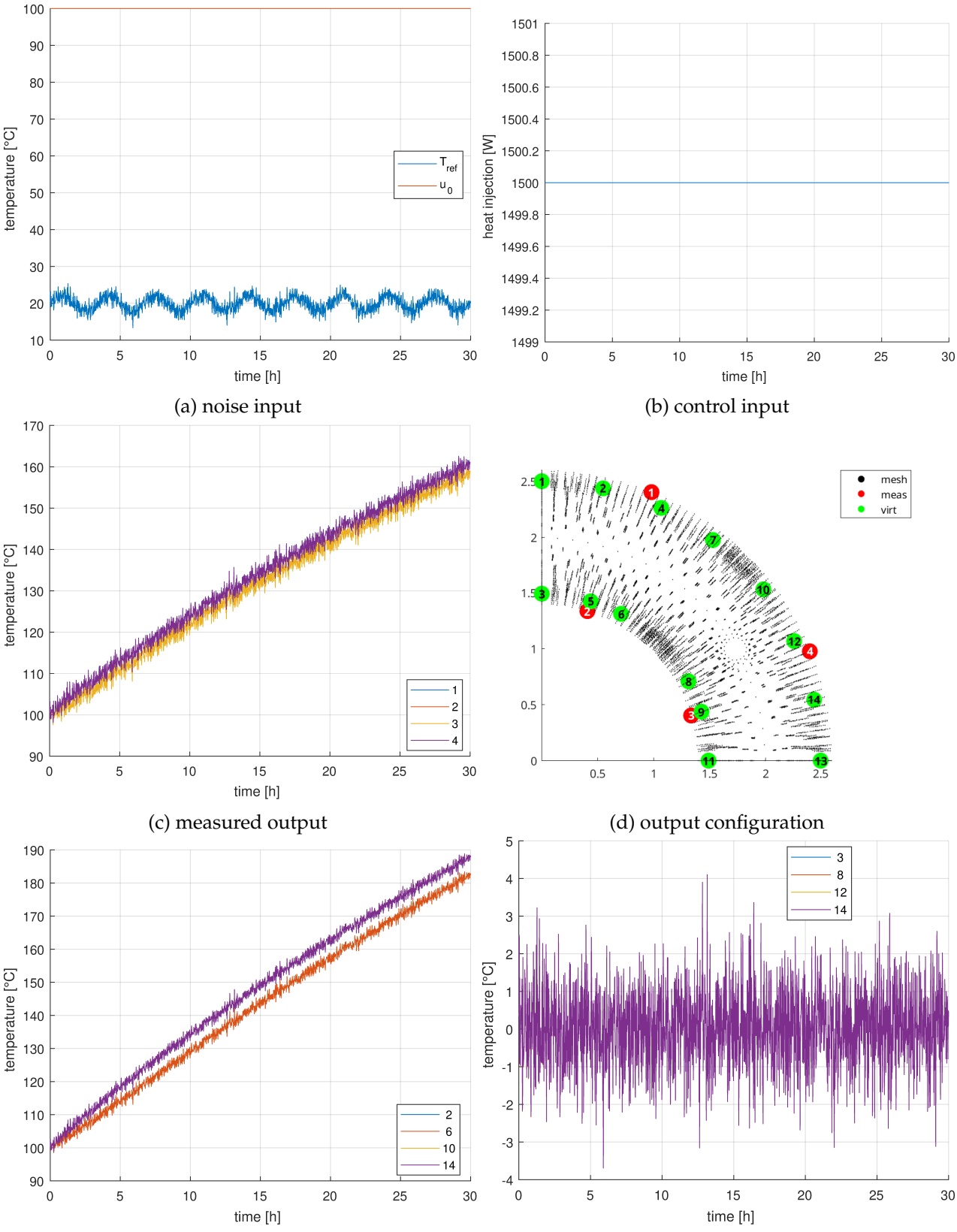
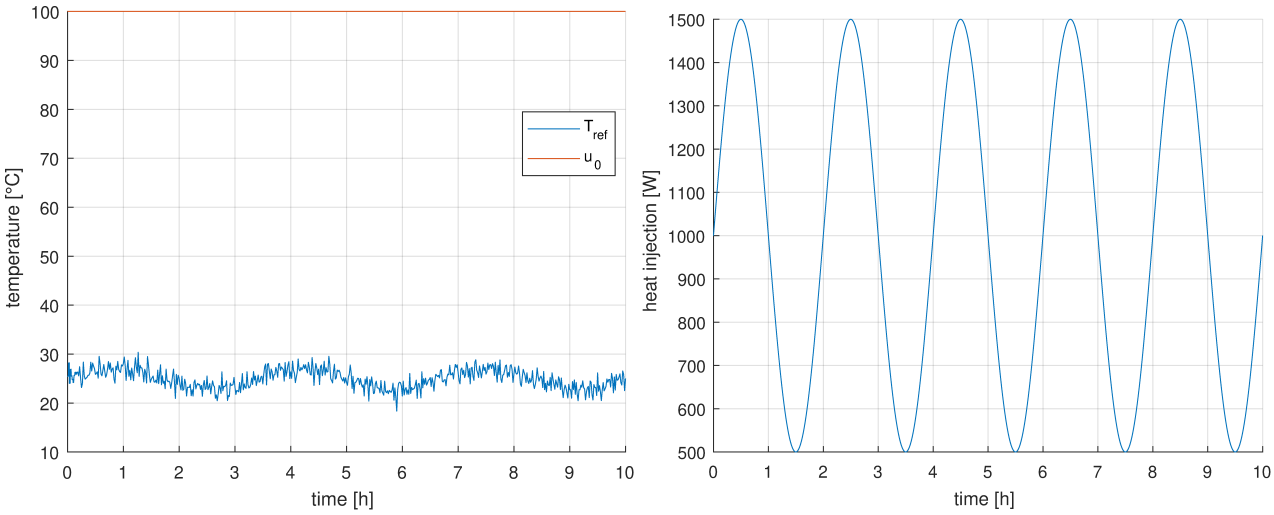
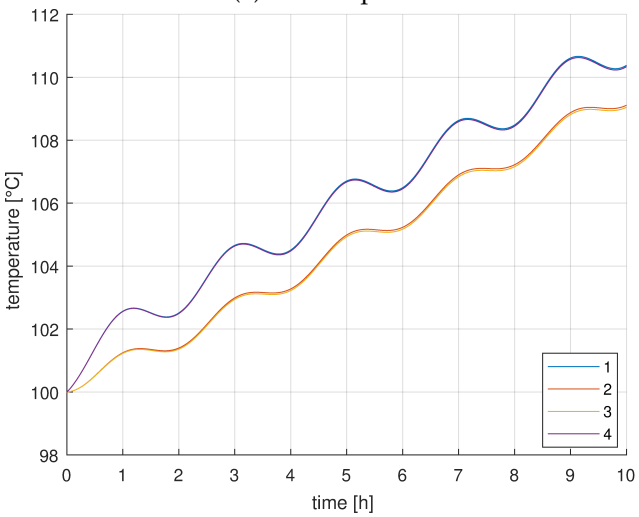


FIGURE B.2: Result of simulation using H_∞ observer based on RFX model with step heat injection. (a) and (b) show respectively noise and control input signals. (c) and (e) show measured and virtual output respectively, according to the mesh configuration shown in (d). The error of the virtual prediction against the true model is plotted in (f).

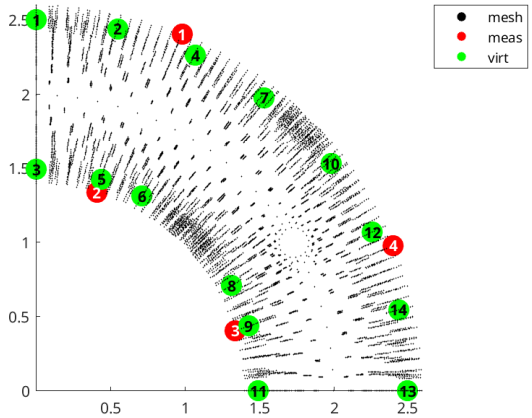


(a) noise input

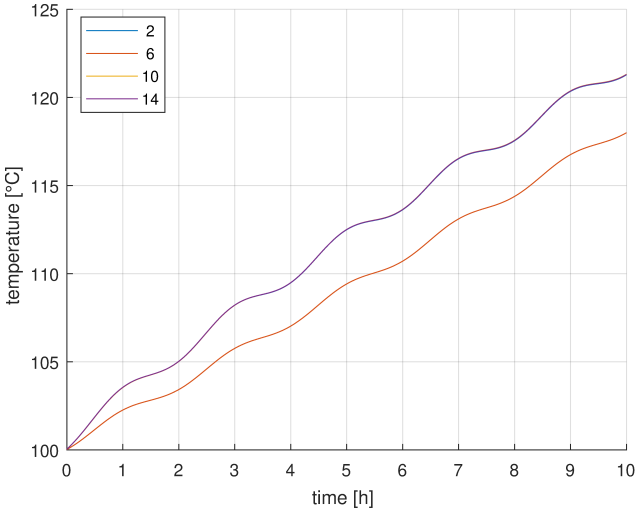
(b) control input



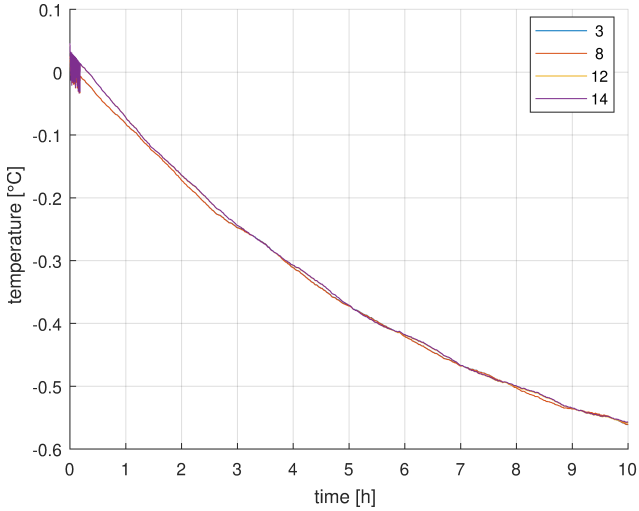
(c) measured output



(d) output configuration

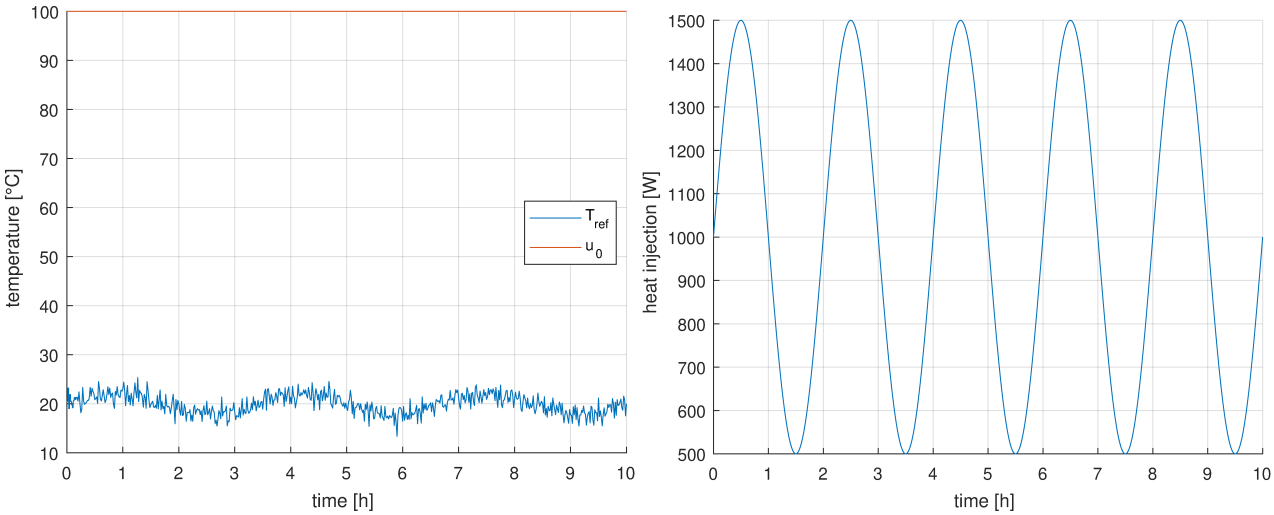


(e) virtual output



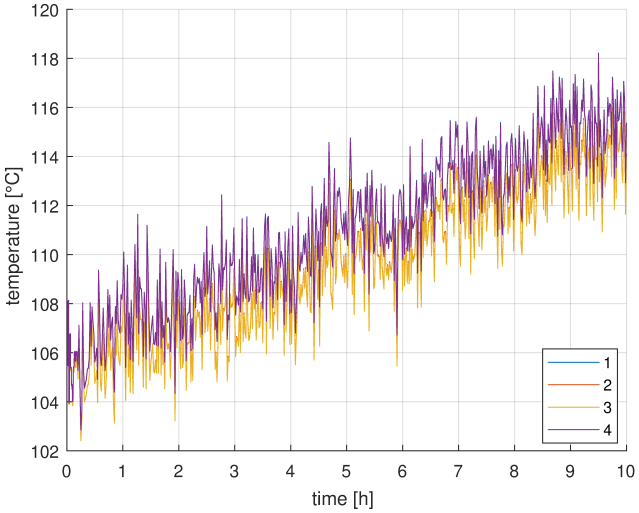
(f) virtual prediction error

FIGURE B.3: Result of simulation using H_∞ observer based on RFX model with 5 °C bias noise on input T_{ref}. (a) and (b) show respectively noise and control input signals. (c) and (e) show measured and virtual output respectively, according to the mesh configuration shown in (d). The error of the virtual prediction against the true model is plotted in (f).

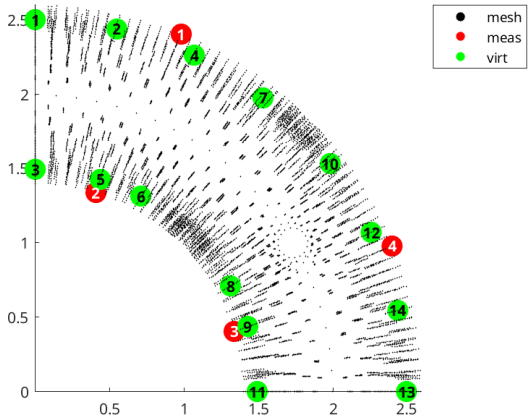


(a) noise input

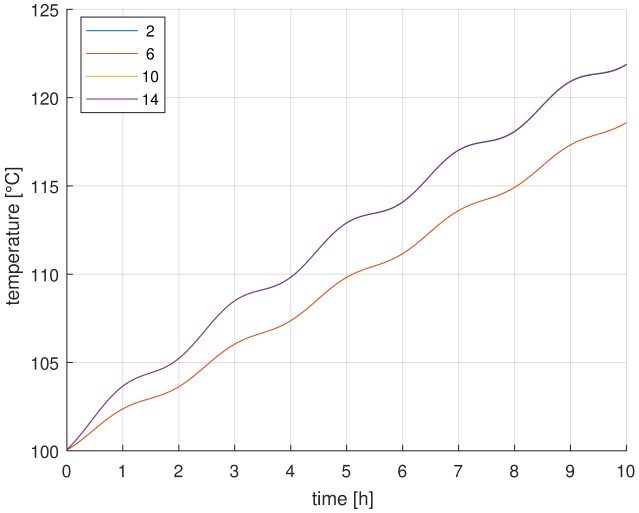
(b) control input



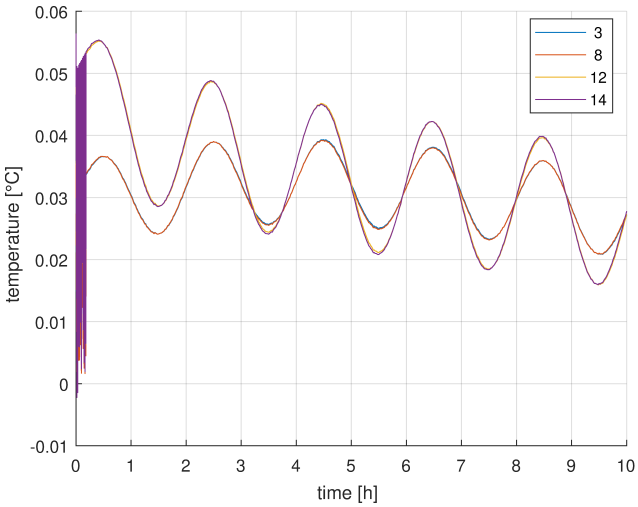
(c) measured output



(d) output configuration

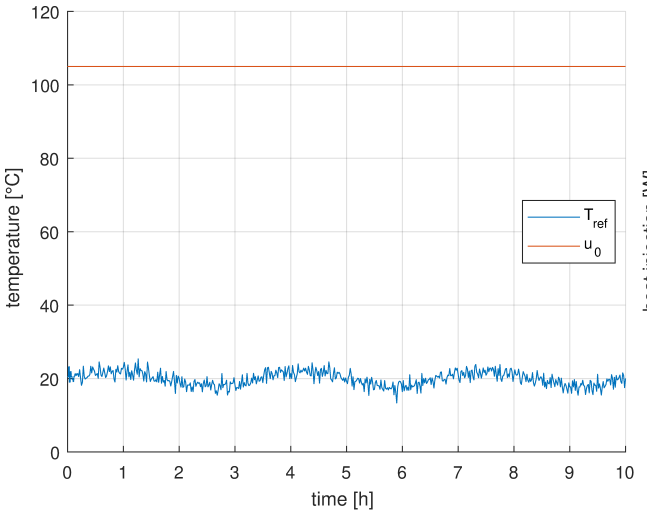


(e) virtual output

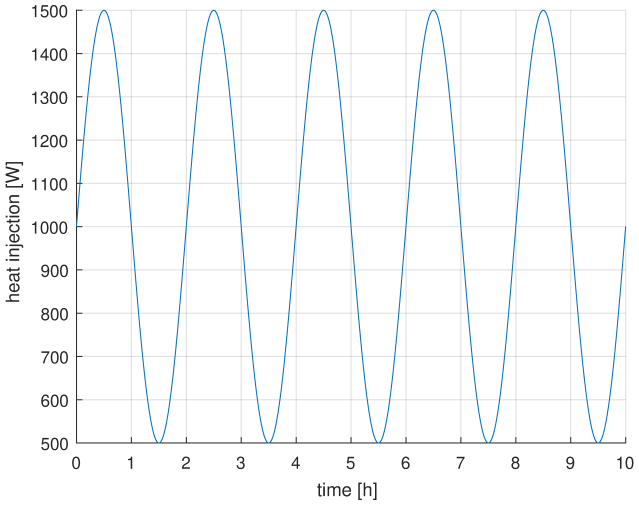


(f) virtual prediction error

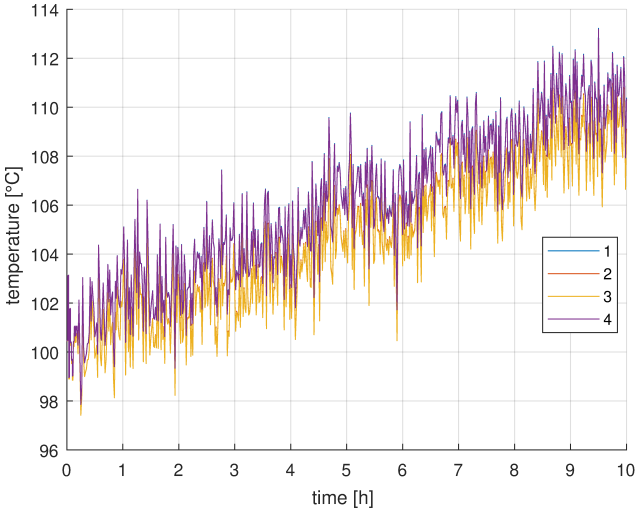
FIGURE B.4: Result of simulation using H_2 observer based on RFX model with 5 °C bias noise on all measured outputs. (a) and (b) show respectively noise and control input signals. (c) and (e) show measured and virtual output respectively, according to the mesh configuration shown in (d). The error of the virtual prediction against the true model is plotted in (f).



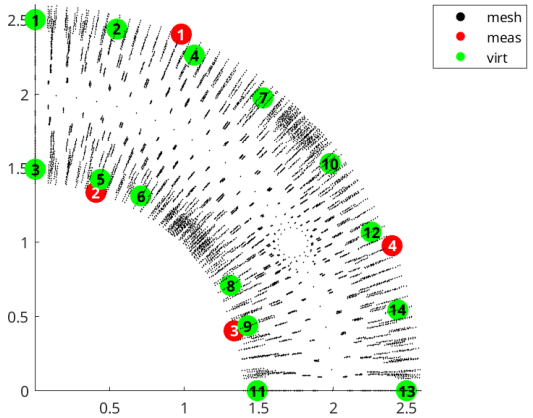
(a) noise input



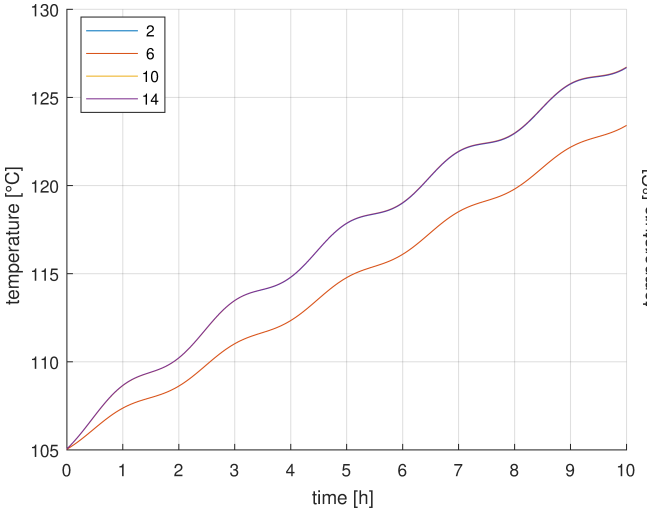
(b) control input



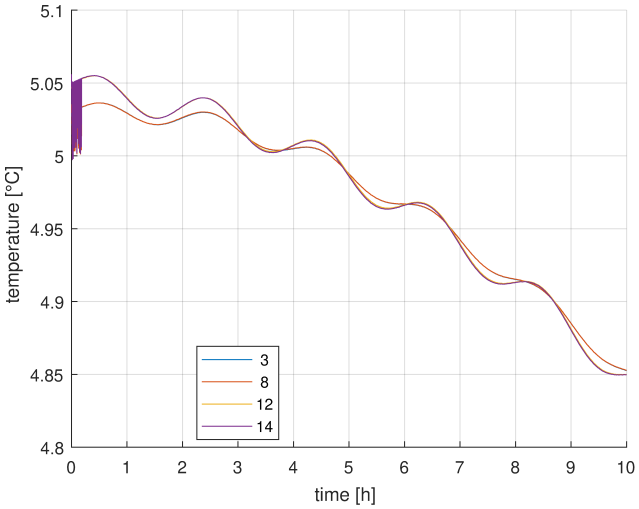
(c) measured output



(d) output configuration



(e) virtual output



(f) virtual prediction error

FIGURE B.5: Result of simulation using H_2 observer based on RFX model with 5 °C bias noise on u_0 initial condition. (a) and (b) show respectively noise and control input signals. (c) and (e) show measured and virtual output respectively, according to the mesh configuration shown in (d). The error of the virtual prediction against the true model is plotted in (f).