algorithms

MDPI

Article

emgr—The Empirical Gramian Framework

Christian Himpe

Computational Methods in Systems and Control Theory Group at the Max Planck Institute for Dynamics of Complex Technical Systems, Sandtorstraße 1, D-39106 Magdeburg, Germany; himpe@mpi-magdeburg.mpg.de

Received: 28 May 2018; Accepted: 24 June 2018; Published: 26 June 2018

Abstract: System Gramian matrices are a well-known encoding for properties of input-output systems such as controllability, observability or minimality. These so-called system Gramians were developed in linear system theory for applications such as model order reduction of control systems. Empirical Gramians are an extension to the system Gramians for parametric and nonlinear systems as well as a data-driven method of computation. The empirical Gramian framework - `emgr` - implements the empirical Gramians in a uniform and configurable manner, with applications such as Gramian-based (nonlinear) model reduction, decentralized control, sensitivity analysis, parameter identification and combined state and parameter reduction.

Keywords: model reduction; model order reduction; decentralized control; sensitivity analysis; parameter identification; empirical gramians; nonlinear systems; reduced order systems; controllability; observability

PACS: 02.30.Yy

MSC: 93A15; 93B20; 93C10

Code Meta Data

name (shortname)	EMpirical GRamian Framework (emgr)
version (release-date)	5.4 (2018-05-05)
identifier (type)	doi:10.5281/zenodo.1241532 (doi)
authors (ORCIDs)	Christian Himpe (0000-0003-2194-6754)
topic (type)	Model Reduction (toolbox)
license (type)	2-Clause BSD (open)
repository (type)	git:github.com/gramian/emgr (git)
languages	Matlab
dependencies	OCTAVE >= 4.2, MATLAB >= 2016b
systems	Linux, Windows
website	http://gramian.de
keywords	empirical-gramians, cross-gramian, combined-reduction

1. Introduction

Attributes of input-output systems, such as controllability, observability or minimality can be assessed by special matrices. These so-called system Gramian matrices, or short system Gramians (Also: Grammian or Gram matrix.), have manifold applications in system theory, control theory and mathematical engineering.

Originally, Gramian-based methods were developed for linear systems [1]. The empirical Gramian matrices [2] are an extension of Gramian-based methods to nonlinear and parametric systems. This work summarizes the **em**pirical **Gr**amian framework (`emgr`) [3], a compact software toolbox, which implements various types of empirical Gramians as well as the related empirical covariance matrices.

An important use of empirical Gramian matrices is model order reduction (MOR), utilizing the capability of system Gramians to quantify the input-output importance of the underlying system's states based on controllability and observability. Several variants of Gramian-based model reduction are available, for example:

- Linear Model Order Reduction [4],
- Robust Model Reduction [5],
- Parametric Model Order Reduction (pMOR) [6],
- Nonlinear Model Order Reduction (nMOR) [2,7–9],
- Second-Order System Model Reduction [10],
- Combined State and Parameter Reduction [11].

Beyond model reduction, (empirical) system Gramians can also be utilized for tasks like:

- Sensitivity Analysis [12,13],
- Parameter Identification [14,15],
- Decentralized Control [16–18],
- Optimal Sensor Placement [19,20], Optimal Actuator Placement [21],
- Optimal Control [22], Model Predictive Control [23],
- Nonlinearity Quantification [24,25].

In addition, various system invariants and indices are computable using system Gramians, and thus also by empirical Gramians:

- System gain [12],
- Cauchy index [26,27],
- Information entropy index [28],
- Nyquist plot enclosed area [29],
- System Frobenius norm and ellipsoid volume [30],
- System H_2-norm [31].

This wide range of applications and the compatibility to nonlinear systems make empirical Gramians a versatile tool in many system-theoretic computations. Furthermore, the empirical Gramians provide a data-driven method of computation with close relations to proper orthogonal decomposition (POD) and balanced POD (bPOD) [32,33].

Various (Matlab) implementations are available for the computation of linear system Gramians by the solution of associated matrix equations, such as the basic `sylvester` command, the `gram`, `lyap` and `covar` commands from the MATLAB Control Toolbox (http://mathworks.com/products/control) and OCTAVE Control Package (http://octave.sourceforge.net/control). For empirical Gramians, the only other generic implementation, to the author's best knowledge, is [34], which provides only the empirical controllability Gramian and the empirical observability Gramian, but *not* any empirical cross Gramian (see Sections 3.1.3 and 3.1.4). This makes `emgr` the unique (open-source) implementation of all three: the empirical controllability Gramian W_C, the empirical observability Gramian W_O and the empirical cross Gramian W_X (sometimes also symbolized by W_{CO} and X_{CG}).

Lastly, it is noted that the term *empirical Gramian* is used as an umbrella term for the original empirical Gramians [2], the empirical covariance matrices [35], modified empirical Gramians [36] or local Gramians [37].

1.1. Aim

After its initial version 1.0 (2013) release, accompanied by [38], the empirical Gramian framework (emgr is also listed in the **Oberwolfach References on Mathematical Software** (ORMS), no. 345: http://orms.mfo.de/project?id=345.) has been significantly enhanced. Apart from extended functionality and accelerated performance, various new concepts and features were implemented. Now, with the release of version 5.4 (2018) [3], this is a follow-up work illustrating the current state of `emgr` and its applicability, as well as documenting the flexibility of this toolbox. In short, the major changes involve:

- Non-symmetric cross Gramian variant,
- linear cross Gramian variant,
- distributed cross Gramian variant and interface,
- inner product kernel interface,
- time-integrator interface,
- time-varying system compatibility,
- tensor-based trajectory storage,
- functional paradigm software design.

1.2. Outline

This work is structured as follows: In Section 2 the empirical Gramian's main application, projection-based model order reduction, is briefly described; followed by Section 3, presenting the mathematical definitions of the computable empirical Gramians. Section 4 summarizes the design decision for `emgr`, while Section 5 documents usage and configuration. Numerical examples are demonstrated in Section 6 and lastly, in Section 7, a short concluding remark is given.

2. Mathematical Preliminaries

The mathematical objects of interest are nonlinear parametric input-output systems, which frequently occur in physical, chemical, biological and technical models or spatially discretized partial differential equations (PDE). These control system models consist of a dynamical system (typically on $\mathbb{R}$, i.e., an ordinary differential equation (ODE)) as well as an output function, and maps the input $u : \mathbb{R}_{>0} \to \mathbb{R}^M$ via the state $x : \mathbb{R}_{>0} \to \mathbb{R}^N$ to the output $y : \mathbb{R}_{>0} \to \mathbb{R}^Q$:

$$\begin{aligned} \dot{x}(t) &= f(t, x(t), u(t), \theta), \\ y(t) &= g(t, x(t), u(t), \theta). \end{aligned} \tag{1}$$

The potentially nonlinear vector-field $f : \mathbb{R}_{>0} \times \mathbb{R}^N \times \mathbb{R}^M \times \mathbb{R}^P \to \mathbb{R}^N$ and output functional $g : \mathbb{R}_{>0} \times \mathbb{R}^N \times \mathbb{R}^M \times \mathbb{R}^P \to \mathbb{R}^Q$ both depend on the time $t \in \mathbb{R}_{>0}$, the state $x(t)$, input or control $u(t)$ and the parameters $\theta \in \mathbb{R}^P$. Together with an initial condition $x(0) = x_0 \in \mathbb{R}^N$, this setup constitutes an initial value problem.

Model Reduction

The aim of model reduction is the algorithmic computation of surrogate reduced order models with lower computational complexity or memory footprint than the original full order model. For the sake of brevity, only combined state and parameter reduction is summarized here, which includes state-space reduction, parametric state-space reduction and parameter-space reduction as special cases; for an elaborate layout see [39].

Given the general, possibly nonlinear, input-output system (1), a combined state and parameter reduced order model:

$$\begin{aligned} \dot{x}_r(t) &= f_r(t, x_r(t), u(t), \theta_r), \\ \tilde{y}(t) &= g_r(t, x_r(t), u(t), \theta_r), \\ x_r(0) &= x_{r,0}, \end{aligned}$$

with a reduced state $x_r : \mathbb{R}_{>0} \to \mathbb{R}^n$, $n \ll N$, and a reduced parameter $\theta_r \in \mathbb{R}^p$, $p \ll P$, is sought. Accordingly, a reduced vector-field $f_r : \mathbb{R}_{>0} \times \mathbb{R}^n \times \mathbb{R}^M \times \mathbb{R}^p \to \mathbb{R}^n$ and a reduced output functional $g_r : \mathbb{R}_{>0} \times \mathbb{R}^n \times \mathbb{R}^M \times \mathbb{R}^p \to \mathbb{R}^Q$ describe the reduced system, for which the reduced system's outputs $\tilde{y} : \mathbb{R}_{>0} \to \mathbb{R}^Q$ should exhibit a small error compared to the full order model, yet preserving the parameter dependency:

$$\|y(\theta) - \tilde{y}(\theta_r)\| \ll 1.$$

A class of methods to obtain such a reduced order model with the associated requirements is described next.

Projection-Based Combined Reduction

Projection-based combined state and parameter reduction is based on (bi-)orthogonal truncated projections for the state- and parameter-space respectively. The state-space trajectory $x(t)$, not too far from a steady-state $\bar{x} \in \mathbb{R}^N$, $\bar{u} \in \mathbb{R}^M$, $f(t, \bar{x}, \bar{u}, \theta) = 0 \ \ \forall t$, is approximated affinely using truncated reducing and reconstructing projections $U_1 \in \mathbb{R}^{N \times n}$ and $V_1 \in \mathbb{R}^{N \times n}$, with $V_1^\intercal U_1 = \mathbb{1}_n$:

$$x_r(t) := V_1^\intercal (x(t) - \bar{x}) \Rightarrow x(t) \approx \bar{x} + U_1 x_r(t).$$

The relevant parameter-space volume is also approximated by truncated reducing and reconstructing projections $\Pi_1 \in \mathbb{R}^{P \times p}$ and $\Lambda_1 \in \mathbb{R}^{P \times p}$, with $\Lambda_1^\intercal \Pi_1 = \mathbb{1}_p$:

$$\theta_r := \Lambda_1^\intercal (\theta - \bar{\theta}) \Rightarrow \theta \approx \bar{\theta} + \Pi_1 \theta_r,$$

relative to a nominal parameter $\bar{\theta}$. Given these truncated projections, a projection-based reduced order model is then obtained by:

$$\begin{aligned} \dot{x}_r(t) &= V_1^\intercal f(t, \bar{x} + U_1 x_r(t), u(t), \bar{\theta} + \Pi_1 \theta_r), \\ \tilde{y}(t) &= \quad g(t, \bar{x} + U_1 x_r(t), u(t), \bar{\theta} + \Pi_1 \theta_r), \\ x_r(0) &= V_1^\intercal (x_0 - \bar{x}), \\ \theta_r &= \Lambda_1^\intercal (\theta - \bar{\theta}). \end{aligned} \tag{2}$$

Thus, to obtain a projection-based reduced order model with respect to the state- and parameter-space, the overall task is determining the truncated projections U_1, V_1, Λ_1 and Π_1.

It should be noted that this approach produces globally reduced order models, meaning U_1, V_1, Λ_1, Π_1 are valid over the whole operating region, which is an application-specific subspace of the Cartesian product of the full order state- and parameter-space $\mathbb{R}^N \times \mathbb{R}^P$.

Gramian-Based Model Reduction

Gramian-based model reduction approximates the input-output behavior of a system by removing the least controllable *and* observable state components. To this end, the system is transformed to a representation in which controllability and observability are balanced. Given a controllability Gramian W_C (Section 3.1.1) and observability Gramian W_O (Section 3.1.2) to an input-output system,

a balancing transformation [4] is computable; here in the variant from [40], utilizing the singular value decomposition (SVD):

$$W_C \overset{\text{SVD}}{=} U_C D_C U_C^\intercal,$$
$$W_O \overset{\text{SVD}}{=} U_O D_O U_O^\intercal$$
$$\rightarrow U_C D_C^{\frac{1}{2}} U_C^\intercal U_O D_O^{\frac{1}{2}} U_O^\intercal = W_C^{\frac{1}{2}} W_O^{\frac{1}{2}} \overset{\text{SVD}}{=} UDV^\intercal.$$

Partitioning the columns of U and V based on the Hankel singular values (HSV) in D, $D_{ii} = \sigma_i > \sigma_{i+1}$, which indicate the balanced state's relevance to the system's input-output behavior, into $U_1 \in \mathbb{R}^{N \times n}$, $V_1 \in \mathbb{R}^{N \times n}$ related to the large HSV and $U_2 \in \mathbb{R}^{N \times N-n}$, $V_2 \in \mathbb{R}^{N \times N-n}$ associated to the small HSV,

$$U = \begin{pmatrix} U_1 & U_2 \end{pmatrix},$$
$$V = \begin{pmatrix} V_1 & V_2 \end{pmatrix},$$

then discarding the partitions associated to small singular values $\sigma_{n+1} \ll \sigma_n$, corresponds to the balanced truncation method [4,41].

A cross Gramian W_X (Section 3.1.4) encodes both controllability and observability in a single linear operator. For a symmetric system, a balancing transformation can then be obtained from an eigenvalue decomposition (EVD) of the cross Gramian [42,43]:

$$W_X \overset{\text{EVD}}{=} UDV^\intercal.$$

Alternatively, an approximate balancing transformation is obtained from an SVD of the cross Gramian [11,44]:

$$W_X \overset{\text{SVD}}{=} UDV^\intercal.$$

The truncated projections, U_1 and V_1, are obtained in the same way as for balanced truncation. Using only the left or only the right singular vectors of W_X for the (truncated) projections of the state-space, and their transpose as reverse transformation, results in orthogonal (Galerkin) projections [45]. This approach is called direct truncation method [11,27], i.e., $V := U^\intercal$.

Similarly, the parameter projection can be based on associated covariance matrices. A transformation aligning the parameters along their principal axes, resulting from an SVD of such a parameter covariance matrix ω [11,46,47]:

$$\omega \overset{\text{SVD}}{=} \Pi \Delta \Lambda,$$

yields truncatable projections given by the singular vectors, with partitioning of Π and Λ based on the singular values in Δ.

3. Empirical Gramians

Classically, the controllability, observability and cross Gramians are computed for linear systems by solving (linear) matrix equations. The empirical Gramians are a data-driven extension to the classic system Gramians, and do not depend on the linear system structure. Computing system Gramians empirically by trajectory simulations was already motivated in [4], but systematically introduced in [2]. The central idea behind the empirical Gramians is the averaging over local Gramians for any varying quantity, such as inputs, initial states, parameters or time-dependent components

around an operating point [48]. This approach is closely related to the concept of local controllability and local observability for nonlinear systems [49].

In the following, first the empirical Gramians for state-space system input-output coherence are summarized, then the empirical Gramians for parameter-space identifiability and combined state and parameter evaluation are described.

3.1. State-Space Empirical Gramians

Gramian-based controllability and observability analysis originates in linear system theory [50], which investigates linear (time-invariant) systems,

$$\begin{aligned} \dot{x}(t) &= Ax(t) + Bu(t), \\ y(t) &= Cx(t). \end{aligned} \tag{3}$$

An obvious approach for nonlinear systems is a linearization at a steady-state [51], but this may obfuscate the original transient dynamics [52,53]. Alternatively, the nonlinear balancing for control affine systems from [54], based on controllability and observability energy functions, could be used. Yet practically, the associated nonlinear system Gramians require solutions to a Hamilton-Jacobi partial differential equation (nonlinear controllability Gramian) and a nonlinear Lyapunov equation (nonlinear observability Gramian) or a nonlinear Sylvester equation (nonlinear cross Gramian), which is currently not feasible for large-scale systems. A compromise between linearized and nonlinear Gramians are empirical Gramians [2,7].

Empirical Gramians are computed by systematically averaging system Gramians obtained from numerical simulations over locations in an operating region near a steady-state. An operating region is defined in this context by sets of perturbations for inputs/controls and the steady-state. Originally in [2], these perturbation sets are constructed by the Cartesian product of sets of directions (standard unit vectors), rotations (orthogonal matrices) and scales (positive scalars) for the input and steady-state respectively. In the empirical Gramian framework, the rotations are limited to the set of the unit matrix and negative unit matrix, as suggested in [2]. This constraint on the rotation entails many numerical simplifications and reduces the perturbation sets to directions (standard unit vectors) and scales (non-zero scalars):

$$\begin{aligned} E_u &= \{e^m \in \mathbb{R}^M : m = 1 \dots M, e_i^m = \delta_{im}\}, \\ S_u &= \{c_k \in \mathbb{R} : k = 1 \dots K, c_k \neq 0\}, \\ E_x &= \{\epsilon^j \in \mathbb{R}^N : j = 1 \dots N, \epsilon_i^j = \delta_{ij}\}, \\ S_x &= \{d_l \in \mathbb{R} : l = 1 \dots L, d_l \neq 0\}. \end{aligned}$$

Yet, only single input and state components can be perturbed at a time in this manner, which is often practically sufficient.

The original empirical Gramians use a centering of the trajectories around the temporal average and solely use impulse input type controls $u(t) = \delta(t)$ [2]. The related empirical covariance matrices center the trajectories around a steady-state and allow arbitrary step functions $u(t) = \sum_k v_k \chi_{[t_k, t_{k+1})}(t)$, $v_k \in \mathbb{R}$, $t_k \in R_{\geq 0}$, $t_{k+1} > t_k$ [35,55]. The empirical Gramian framework allows to compute either as well as further centering variants (Section 5.4). In the following, empirical Gramians and empirical covariance matrices will be jointly referred to by the term "empirical Gramian".

3.1.1. Empirical Controllability Gramian

The (linear) controllability (The term **controllability** is used instead of **reachability** as in [2,4,56].) Gramian quantifies how well the state of an underlying linear system is driven by the input and is defined as:

$$W_C := \int_0^\infty \mathrm{e}^{At}\, BB^\intercal\, \mathrm{e}^{A^\intercal t}\, \mathrm{d}t = \int_0^\infty (\mathrm{e}^{At}\, B)(\mathrm{e}^{At}\, B)^\intercal\, \mathrm{d}t.$$

The empirical variant is given by the following definition based on [2,35].

Definition 1 (Empirical Controllability Gramian)**.** *Given non-empty sets E_u and S_u, the **empirical controllability Gramian** $\widehat{W}_C \in \mathbb{R}^{N\times N}$ is defined as:*

$$\widehat{W}_C = \frac{1}{|S_u|} \sum_{k=1}^{|S_u|} \sum_{m=1}^{M} \frac{1}{c_k^2} \int_0^T \Psi^{km}(t)\, \mathrm{d}t,$$
$$\Psi^{km}(t) = (x^{km}(t) - \bar{x}^{km})(x^{km}(t) - \bar{x}^{km})^\intercal \in \mathbb{R}^{N\times N},$$

with the state trajectories $x^{km}(t)$ for the input configurations $\hat{u}^{km}(t) = c_k e^m \circ u(t) + \bar{u}$, and the offsets $\bar{u}$, $\bar{x}^{km}$.

For an asymptotically stable linear system, delta impulse input $u_i(t) = \delta(t)$ and an arithmetic average over time as offset $\bar{x} = \frac{1}{T}\int_0^T x(t)\, \mathrm{d}t$, the empirical controllability Gramian is equal to the controllability Gramian $\widehat{W}_C = W_C$ [2]. The numerical computation of this empirical Gramian requires $(|S_u| \cdot M)$ trajectories and is related to POD.

3.1.2. Empirical Observability Gramian

The (linear) observability Gramian quantifies how well a change in the state of an underlying linear system is visible in the outputs and is defined as:

$$W_O := \int_0^\infty \mathrm{e}^{A^\intercal t}\, C^\intercal C\, \mathrm{e}^{At}\, \mathrm{d}t = \int_0^\infty (\mathrm{e}^{A^\intercal t}\, C^\intercal)(\mathrm{e}^{A^\intercal t}\, C^\intercal)^\intercal\, \mathrm{d}t.$$

The empirical variant is given by the following definition based on [2,35].

Definition 2 (Empirical Observability Gramian)**.** *Given non-empty sets E_x and S_x, the **empirical observability Gramian** $\widehat{W}_O \in \mathbb{R}^{N\times N}$ is defined as:*

$$\widehat{W}_O = \frac{1}{|S_x|} \sum_{l=1}^{|S_x|} \frac{1}{d_l^2} \int_0^\infty \Psi^{l}(t)\, \mathrm{d}t,$$
$$\Psi^{l}_{ij}(t) = (y^{li}(t) - \bar{y}^{li})^\intercal (y^{lj}(t) - \bar{y}^{lj}) \in \mathbb{R},$$

with the output trajectories $y^{li}(t)$ for the initial state configurations $x_0^{li} = d_l \epsilon^i + \bar{x}$, $u(t) = \bar{u}$, and the offsets $\bar{u}$, $\bar{x}$, $\bar{y}^{li}$.

For an asymptotically stable linear system, no input and an arithmetic average over time as offset $\bar{y} = \frac{1}{T}\int_0^T y(t)\, \mathrm{d}t$, the empirical observability Gramian is equal to the observability Gramian $\widehat{W}_O = W_O$ [2]. The numerical computation of this empirical Gramians requires $(|S_x| \cdot N)$ trajectories.

3.1.3. Empirical Linear Cross Gramian

The (linear) cross Gramian [56,57] quantifies the controllability and observability, and thus minimality, of an underlying square, $M := \dim(u(t)) = \dim(y(t)) =: Q$, linear system and is defined as:

$$W_X := \int_0^\infty \mathrm{e}^{At} BC\, \mathrm{e}^{At}\, \mathrm{d}t = \int_0^\infty (\mathrm{e}^{At} B)(\mathrm{e}^{A^\intercal t} C^\intercal)^\intercal\, \mathrm{d}t.$$

Augmenting the linear system's dynamical system component with its transposed system (The transposed system is equivalent to the negative adjoint system), induces an associated controllability Gramian of which the upper right block corresponds to the cross Gramian [58,59]:

$$\begin{pmatrix} \dot{x}(t) \\ \dot{z}(t) \end{pmatrix} = \begin{pmatrix} A & 0 \\ 0 & A^\intercal \end{pmatrix} \begin{pmatrix} x(t) \\ z(t) \end{pmatrix} + \begin{pmatrix} B \\ C^\intercal \end{pmatrix} u(t) \Rightarrow \overline{W}_C = \begin{pmatrix} W_C & W_X \\ W_X^\intercal & W_O \end{pmatrix}. \tag{4}$$

The empirical variant restricted to the upper right block of this augmented controllability Gramian (4) is given by the following definition based on [60].

Definition 3 (Empirical Linear Cross Gramian)**.** *Given non-empty sets E_u and S_u, the **empirical linear cross Gramian** $\widehat{W}_Y \in \mathbb{R}^{N\times N}$ is defined as:*

$$\widehat{W}_Y = \frac{1}{|S_u|} \sum_{k=1}^{|S_u|} \sum_{m=1}^{M} \frac{1}{c_k^2} \int_0^T \Psi^{km}(t)\, \mathrm{d}t,$$
$$\Psi^{km}(t) = (x^{km}(t) - \bar{x}^{km})(z^{km}(t) - \bar{z}^{km})^\intercal \in \mathbb{R}^{N\times N},$$

with the state trajectories $x^{km}(t)$ and adjoint state trajectories $z^{km}(t)$ for the input configurations $\hat{u}^{km}(t) = c_k e^m \circ u(t) + \bar{u}$, and the offsets $\bar{u}$, $\bar{x}^{km}$, $\bar{z}^{km}$.

For an asymptotically stable linear system, delta impulse input $u_i(t) = \delta(t)$ and an arithmetic average over time as offset $\bar{x} = \frac{1}{T}\int_0^T x(t)\, \mathrm{d}t$, $\bar{z} = \frac{1}{T}\int_0^T z(t)\, \mathrm{d}t$, the empirical linear cross Gramian is equal to the cross Gramian due to the result of the empirical controllability Gramian. The numerical computation of this empirical Gramian requires $(2 \cdot |S_u| \cdot M)$ trajectories and is related to balanced POD [61].

3.1.4. Empirical Cross Gramian

Analytically, a cross Gramian for (control-affine) nonlinear gradient systems was developed in [62,63], yet the computation of this nonlinear cross Gramian (This is also called **cross operator** or **cross map** in this context) is infeasible for large systems. For (nonlinear) Single-Input-Single-Output (SISO) systems, the empirical variant of the cross Gramian is developed in [12], for (nonlinear) Multiple-Input-Multiple-Output (MIMO) systems in [11].

Definition 4 (Empirical Cross Gramian)**.** *Given non-empty sets E_u, E_x, S_u and S_x, the **empirical cross Gramian** $\widehat{W}_X \in \mathbb{R}^{N\times N}$ is defined as:*

$$\widehat{W}_X = \frac{1}{|S_u||S_x|M} \sum_{k=1}^{|S_u|} \sum_{l=1}^{|S_x|} \sum_{m=1}^{M} \frac{1}{c_k d_l} \int_0^\infty \Psi^{klm}(t)\, \mathrm{d}t,$$
$$\Psi_{ij}^{klm} = (x_i^{km}(t) - \bar{x}_i^{km})(y_m^{lj}(t) - \bar{y}_m^{lj}) \in \mathbb{R},$$

with the state trajectories $x^{km}(t)$ for the input configurations $\hat{u}^{km}(t) = c_k e^m \circ u(t) + \bar{u}$, the output trajectories $y^{lj}(t)$ for the initial state configurations $x_0^{lj} = d_l \epsilon^j + \bar{x}$, and the offsets $\bar{u}$, $\bar{x}^{km}$, $\bar{x}$, $\bar{y}^{lj}$.

For an asymptotically stable linear system, delta impulse input $u_i(t) = \delta(t)$ and an arithmetic averages over time as offsets $\bar{x} = \frac{1}{T}\int_0^T x(t)\,\mathrm{d}t$, $\bar{y} = \frac{1}{T}\int_0^T y(t)\,\mathrm{d}t$, the empirical cross Gramian is equal to the cross Gramian $\widehat{W}_X = W_X$ [11,12]. The numerical computation of this empirical Gramian requires $(|S_x| \cdot (N + |S_u| \cdot M))$ trajectories.

3.1.5. Empirical Non-Symmetric Cross Gramians

The (empirical) cross Gramian is only computable for square systems, and verifiably useful for symmetric or gradient systems [11,44,57]. In [39] an extension to the classic cross Gramian is proposed. Based on results from decentralized control [16], a non-symmetric cross Gramian is computable for non-square systems and thus non-symmetric systems. Given a partitioning of $B = [b_1, \ldots, b_M]$, $b_i \in \mathbb{R}^{M\times 1}$ and $C = [c_1, \ldots, c_Q]^\intercal$, $c_j \in \mathbb{R}^{1\times Q}$ from the linear system (3), the (linear) non-symmetric cross Gramian is defined as:

$$W_Z := \sum_{i=1}^{M}\sum_{j=1}^{Q}\int_0^\infty \mathrm{e}^{At}\, b_i c_j\, \mathrm{e}^{At}\,\mathrm{d}t = \int_0^\infty \mathrm{e}^{At}\left(\sum_{i=1}^{M} b_i\right)\left(\sum_{j=1}^{Q} c_j\right)\mathrm{e}^{At}\,\mathrm{d}t.$$

For this cross Gramian to the associated "average" SISO system, an empirical variant is then given by:

Definition 5 (Empirical Non-Symmetric Cross Gramian)**.** *Given non-empty sets E_u, E_x, S_u and S_x, the **empirical non-symmetric cross Gramian** $\widehat{W}_Z \in \mathbb{R}^{N\times N}$ is defined as:*

$$\widehat{W}_Z = \frac{1}{|S_u||S_x|M}\sum_{k=1}^{|S_u|}\sum_{l=1}^{|S_x|}\sum_{m=1}^{M}\sum_{q=1}^{Q}\frac{1}{c_k d_l}\int_0^\infty \Psi^{klmq}(t)\,\mathrm{d}t,$$

$$\Psi_{ij}^{klmq} = (x_i^{km}(t) - \bar{x}_i^{km})(y_q^{lj}(t) - \bar{y}_q^{lj}) \in \mathbb{R},$$

with the state trajectories $x^{km}(t)$ for the input configurations $\hat{u}^{km}(t) = c_k e^m \circ u(t) + \bar{u}$, the output trajectories $y^{lj}(t)$ for the initial state configurations $x_0^{lj} = d_l \epsilon^j + \bar{x}$, and the offsets $\bar{u}$, $\bar{x}^{km}$, $\bar{x}$, $\bar{y}^{lj}$.

Corollary 1. *For an asymptotically stable linear system, delta impulse input $u_i(t) = \delta(t)$ and arithmetic averages over time as offsets $\bar{x} = \frac{1}{T}\int_0^T x(t)\,\mathrm{d}t$, $\bar{y} = \frac{1}{T}\int_0^T y(t)\,\mathrm{d}t$, the empirical non-symmetric cross Gramian is equal to the cross Gramian $\widehat{W}_Z = W_Z$ of the average SISO system.*

Proof. This is a direct consequence of [11] (Lemma 3). □

The numerical computation of the empirical non-symmetric cross Gramian requires $(|S_x| \cdot (N + |S_u| \cdot M))$ trajectories; the same as for the empirical cross Gramian.

3.2. Parameter-Space Empirical Gramians

To transfer the idea of Gramian-based state-space reduction to parameter-space reduction, the concepts of *controllability* and *observability* are extended to the parameter-space. This leads to controllability-based parameter identification and observability-based parameter identification. Note that the observability-based parameter identification (and parameter reduction) is related to the active subspaces method [64].

3.2.1. Empirical Sensitivity Gramian

Controllability-based parameter identification can be realized using an approach from [46], which treats the parameters as (additional) constant inputs. The controllability Gramian for a linear

system with linear parametrization (constant source or load) can be decomposed additively based on linear superposition:

$$\dot{x}(t) = Ax(t) + Bu(t) + F\theta = Ax(t) + \begin{pmatrix} B & F \end{pmatrix} \begin{pmatrix} u(t) \\ \theta \end{pmatrix}$$
$$\Rightarrow W_C = W_C(A, B) + \sum_{i=1}^{P} W_{C,i}(A, F_{*i}).$$

Similar to [12], the trace of the parameter-controllability Gramians $W_{C,i}$ embodies a measure of (average) sensitivity, and holds approximately for systems with nonlinear parametrization [38].

Definition 6 (Empirical Sensitivity Gramian)**.** *The **empirical sensitivity Gramian** is given by a diagonal matrix with entries corresponding to the traces of the parameter-controllability Gramians,*

$$W_{S,ii} := \mathrm{tr}(W_{C,i}).$$

The sum over all controllability Gramians can also be used for robust model reduction [5]. Similarly, treating the parameters as inputs, the cross Gramian's trace can be utilized as a sensitivity measure [13].

3.2.2. Empirical Identifiability Gramian

For an observability-based parameter identification, the parameters are interpreted as additional states of the system [14,19]. This approach leads to the **augmented system**, in which the system's state x is appended by the parameter θ and, since the parameters are (assumed) constant over time, the components of the vector-field associated to the *parameter-states* are zero:

$$\begin{aligned} \begin{pmatrix} \dot{x}(t) \\ \dot{\theta}(t) \end{pmatrix} &= \begin{pmatrix} f(t, x(t), u(t), \theta(t)) \\ 0 \end{pmatrix}, \\ y(t) &= g(t, x(t), u(t), \theta(t)), \\ \begin{pmatrix} x(0) \\ \theta(0) \end{pmatrix} &= \begin{pmatrix} x_0 \\ \theta \end{pmatrix}, \end{aligned} \tag{5}$$

leaving the parameter-state's initial value for testing parameter perturbations. The observability Gramian to this augmented system, the *augmented observability Gramian*, has the block structure:

$$\widehat{W}_O = \begin{pmatrix} W_O & W_M \\ W_M^\mathsf{T} & W_P \end{pmatrix},$$

with the state-space observability Gramian W_O, the parameter-space observability Gramian W_I and the mixed state and parameter block $W_M = W_M^\mathsf{T}$. To isolate the parameter identifiability information, the state-space block is eliminated.

Definition 7 (Empirical Identifiability Gramian)**.** *The **empirical identifiability Gramian** is given by the Schur-complement of the empirical augmented observability Gramian for the lower right block:*

$$W_I = W_P - W_M^\mathsf{T} W_O^{-1} W_M.$$

Often, it is sufficient to approximate the empirical identifiability Gramian by the lower right block of the augmented observability Gramian W_P:

$$W_I \approx W_P.$$

Apart from the relation of the identifiability Gramian to the Fischer information matrix [19], also the connection of the (parameter) observability Gramian to the (parameter) Hessian matrix [65] is noted here.

3.2.3. Empirical Cross-Identifiability Gramian

If a system is square, the augmented system (5) remains square and for linear systems symmetry is also preserved. Hence, a cross Gramian of the augmented system is computable [11].

Definition 8 (Empirical Joint Gramian)**.** *The **empirical joint Gramian** is given by the empirical cross Gramian of the augmented system.*

The joint Gramian is an augmented cross Gramian and has a similar block structure as the augmented observability Gramian,

$$W_J = \begin{pmatrix} W_X & W_m \\ 0 & 0 \end{pmatrix},$$

but due to the uncontrollable parameter-states, the lower (parameter-related) blocks are identically zero. Nonetheless, the observability-based parameter identifiability information can be extracted from the mixed block W_m.

Definition 9 (Empirical Cross-Identifiability Gramian)**.** *The **empirical cross-identifiability Gramian** is given by the Schur-complement of the symmetric part of the empirical joint Gramian for the lower right block:*

$$W_{\tilde{I}} = 0 - \frac{1}{2} W_m^\intercal (W_X + W_X^\intercal)^{-1} W_m.$$

Thus, the empirical joint Gramian enables the combined state and parameter analysis by the empirical cross Gramian W_X and empirical cross-identifiability Gramian $W_{\tilde{I}}$ from a single $N \times N + P$ matrix. Note that the empirical joint Gramian may also be computed based on the non-symmetric cross Gramian. Additionally, a parameter Gramian, such as W_I or the $W_{\tilde{I}}$, could be balanced with the loadability Gramian from [15].

3.3. Notes on Empirical Gramians

It should be noted that empirical Gramians only yield workable results if the operating region of the system is restricted to a single steady-state. If the trajectories used for the assembly of empirical Gramians are periodic or do not attain this steady-state, their performance is similar to time-limited balancing methods, see for example [66] and references therein. Overall, the quality of empirical-gramian-based methods depend largely on the quality of the measured or simulated (output) trajectory data. Yet, due to the data-driven nature of the empirical Gramians, even unstable systems or systems with inhomogeneous initial conditions are admissible.

The dominant computational cost in the assembly of empirical Gramians is the computation of trajectories. This is especially obvious from the observability-based empirical Gramians, which appear to perturb every component of the steady or initial state. Yet in practice, the number of trajectories can often be reduced based on knowledge about the underlying model or operating region. Furthermore, the trajectories are computable in parallel.

4. Implementation Details

This section states concisely the theoretical, practical and technical design decisions in the implementation of the empirical Gramian framework–`emgr` [3], as well as describes the unified and configurable approach to empirical Gramian computation.

4.1. Design Principles

`emgr` is realized using the high-level, interpreted Matlab programming language, which is chosen due to its widespread use, long-term compatibility and mathematical expressiveness. This enables first, a wide circulation due to compatibility with MATHWORKS MATLAB® [67], and second, the usage of the open-source variant GNU OCTAVE [68]. Generally, the implementation of `emgr` follows the procedural programming paradigm, includes various functional programming techniques and avoids object-oriented programming.

Since empirical Gramians are computable by mere basic linear algebra operations, Matlab code can be evaluated efficiently by vectorization, which transfers computationally intensive tasks as bulk operations to the Basic Linear Algebra Subroutines (BLAS) back-end.

Overall, `emgr` is a reusable open-source toolbox, and encompasses less than 500 LoC (Lines of Code) in a single file and a cyclomatic complexity of <100 of the main function. Apart from a Matlab interpreter, `emgr` has **no** further dependencies, such as on other toolboxes. The source code is engineered with regard to the best practice guides [69] (coding style) and [70] (performance).

Furthermore, two variants of `emgr` are maintained: First, `emgr_oct` (See http://gramian.de/emgr_oct.m), uses OCTAVE-specific language extension: *default arguments* and *assignment operations*, second, `emgr_lgc` (See http://gramian.de/emgr_lgc.m), enables compatibility to MATLAB versions before 2016b not supporting *implicit expansion*, also known as *automatic broadcasting*.

4.2. Parallelization

Apart from vectorization allowing the implicit use of single-instruction-multiple-data (SIMD) functionality for vectorized block operations, also multi-core parallelization is used to maximize use of available compute resources.

4.2.1. Shared Memory Parallelization

For shared memory systems with uniform memory access (UMA), two types of parallelization are utilized. First, an **implicit parallelization** may be applied by the interpreter for an additional acceleration of block operations. Second, **explicit parallelization** is available for the computation of different state and output trajectories, using parallel for-loops `parfor` (See http://mathworks.com/help/matlab/ref/parfor.html), but deactivated by default to guarantee replicable results, as the use of `parfor` does not guarantee a unique order of execution.

4.2.2. Heterogeneous Parallelization

The actual empirical Gramians result from N^2 inner products. In case of the default Euclidean inner product, this amounts to a dense matrix-matrix-multiplication (Implemented as Generalized Matrix Multiplication (GEMM) $R = AB + \alpha C$) by BLAS), which can be efficiently computed by General Purpose Graphics Processing Units (GPGPUs). In the case of an integrated GPU with zero-copy shared memory architecture, such as uniform memory model (UMM) or heterogeneous unified memory access (hUMA) [71], is used, the assembly of the Gramian matrices can be performed with practically no overhead, since the trajectories, which are usually computed and stored in CPU memory space, do not need to be copied between CPU and GPU memory spaces. The GPU can directly operate on the shared memory.

4.2.3. Distributed Memory Parallelization

A disadvantage of empirical Gramian matrices is the quadratically growing memory requirements with respect to the state-space (and parameter-space) dimension, since for an N dimensional system, a (dense) empirical Gramian of dimension $N \times N$ is computed. To combat this shortcoming, a specific property of the empirical cross Gramian can be exploited: The columns of the empirical cross Gramian, and thus the empirical joint Gramian, may be computed separately,

$$\begin{aligned}\widehat{W}_X &= [\widehat{w}_X^1, \ldots, \widehat{w}_X^N],\\ \widehat{w}_X^j &= \frac{1}{|S_u||S_x|M} \sum_{k=1}^{|S_u|} \sum_{l=1}^{|S_x|} \sum_{m=1}^{M} \frac{1}{c_k d_l} \int_0^\infty \psi^{klmj}(t)\,\mathrm{d}t \in \mathbb{R}^{N\times 1},\\ \psi_i^{klmj} &= (x_i^{km}(t) - \bar{x}_i^{km})(y_m^{lj}(t) - \bar{y}_m^{lj}) \in \mathbb{R},\end{aligned}$$

hence this **distributed empirical cross Gramian** [72] (Section 4.2) is computable in parallel and communication-free on a distributed memory computer system, or sequentially in a memory-economical manner as a **low-rank empirical cross Gramian** [73] on a unified memory computer system. This column-wise computability translates also to the empirical joint Gramian and the non-symmetric variants of the empirical cross and joint Gramian.

Based on this partitioning, a low-rank representation can be obtained in a memory-bound or compute-bound setting together with the hierarchical approximate proper orthogonal decomposition (HAPOD) [72]. This POD variant allows to directly compute a Galerkin projection from an arbitrary column-wise partitioning of the empirical cross Gramian.

5. Interface

`emgr` provides a uniform function call for the computation of all empirical Gramian types. The subsequent signature documentation is based on [47] and http://gramian.de (the current instance (2018-06) of http://gramian.de is preserved at https://archive.li/p0cIO.) Minimally, the `emgr` function requires five mandatory arguments (single letter):

```
emgr(f,g,s,t,w)
```

additionally eight optional arguments (double letter) allow a usage by:

```
emgr(f,g,s,t,w,pr,nf,ut,us,xs,um,xm,dp)
```

furthermore, a single argument variant may also be used,

```
emgr('version')
```

which returns the current version number.

5.1. Mandatory Arguments

For the minimal usage, the following five arguments are required:

`f` handle to a function with the signature `xdot = f(x,u,p,t)` representing the system's vector-field and expecting the arguments: current state `x`, current input `u`, (current) parameter `p` and current time `t`.

`g` handle to a function with the signature `y = g(x,u,p,t)` representing the system's output functional and expecting the arguments: current state `x`, current input `u`, (current) parameter `p` and current time `t`.

If `g = 1`, the identity output functional $g(t, x(t), u(t), \theta) = x(t)$ is assumed.

`s` three component vector `s = [M,N,Q]` setting the dimensions of the input $M := \dim(u(t))$, state $N := \dim(x(t))$ and output $Q := \dim(y(t))$.

`t` two component vector `t = [h,T]` specifying the time-step width h and time horizon T.

`w` character selecting the empirical Gramian type; for details see Section 5.2.

5.2. Features

The admissible characters to select the empirical Gramian type are as follows:

`'c'` Empirical controllability Gramian (see Section 3.1.1),
`emgr` returns a matrix:
$N \times N$ empirical controllability Gramian matrix W_C.

`'o'` Empirical observability Gramian (see Section 3.1.2),
`emgr` returns a matrix:
$N \times N$ empirical observability Gramian matrix W_O.

`'x'` Empirical cross Gramian (see Section 3.1.4),
`emgr` returns a matrix:
$N \times N$ empirical cross Gramian matrix W_X.

`'y'` Empirical linear cross Gramian (see Section 3.1.3),
`emgr` returns a matrix:
$N \times N$ empirical linear cross Gramian matrix W_Y.

`'s'` Empirical sensitivity Gramian (see Section 3.2.1),
`emgr` returns a cell array. holding:
$N \times N$ empirical controllability Gramian matrix W_C,
$P \times 1$ empirical sensitivity Gramian diagonal W_S.

`'i'` Empirical identifiability Gramian (see Section 3.2.2),
`emgr` returns a cell array holding:
$N \times N$ empirical observability Gramian matrix W_O,
$P \times P$ empirical identifiability Gramian matrix W_I.

`'j'` Empirical joint Gramian (see Section 3.2.3),
`emgr` returns a cell array holding:
$N \times N$ empirical cross Gramian matrix W_X,
$P \times P$ empirical cross-identifiability Gramian matrix $W_{\check{I}}$.

A cell array is a generic container (array) in the Matlab language.

5.2.1. Non-Symmetric Cross Gramian

The non-symmetric cross Gramian [39] (see Section 3.1.5) is a special variant of the cross Gramian for non-square and non-symmetric MIMO systems, which reduces to the regular cross Gramian for SISO systems. Since the computation is similar to the empirical cross Gramian and a non-symmetric variant of the empirical joint Gramian shall be computable too, instead of a Gramian type selected through the argument `w`, it is selectable via an option flag: Non-symmetric variants may be computed for the empirical cross Gramian (`w = 'x'`), empirical linear cross Gramian (`w = 'y'`) or empirical joint Gramian (`w = 'j'`) by activating the flag `nf(7) = 1`.

5.2.2. Parametric Systems

Parametric model order reduction is accomplished by averaging an empirical Gramian over a discretized parameter-space [6]. To this end the parameter sampling points, arranged as columns of a matrix, can be supplied via the optional argument `pr`.

5.2.3. Time-Varying Systems

Since empirical Gramians are purely based on trajectory data, they are also computable for time varying systems as described in [74]. The empirical Gramian framework can compute averaged Gramians for time varying systems [75], which are time independent matrices.

5.3. Optional Arguments

The eight optional arguments allow a detailed definition of the operating region and configuration of the computation.

`pr` system parameters (Default value: 0)
vector a column vector holding the parameter components,
matrix a set of parameters, each column holding one parameter.

`nf` twelve component vector encoding the option flags, for details see Section 5.4.

`ut` input function (Default value: 1)
handle function handle expecting a signature `u_t = u(t)`,
0 pseudo-random binary input,
1 delta impulse input,
∞ decreasing frequency exponential chirp.

`us` steady-state input (Default value: 0)
scalar set all M steady-state input components to argument,
vector set steady-state input to argument of expected dimension $M \times 1$.

`xs` steady-state (Default value: 0)
scalar set all N steady-state components to argument,
vector set steady-state to argument of expected dimension $N \times 1$.

`um` input scales (Default value: 1)
scalar set all M maximum input scales to argument,
vector set maximum input scales to argument of expected dimension $M \times 1$,
matrix set scales to argument with M rows; used as is.

`xm` initial state scales (Default value: 1)
scalar set all N maximum initial state scales to argument,
vector set maximum steady-state scales to argument of expected dimension $N \times 1$,
matrix set scales to argument with N rows; used as is.

`dp` inner product interface via a handle to a function with the signature `z = dp(x,y)` defining the dot product for the Gramian matrix computation (Default value: `[]`).

Inner Product Interface

The empirical Gramian matrices are computed by inner products of trajectory data. A custom inner products for the assembly of the empirical Gramians matrix, can be set by the argument `dp`, which expects a handle to a function with the signature:

```
z = dp(x,y)
```

and the arguments:

`x` matrix of dimension $N \times \frac{T}{h}$,
`y` matrix of dimension $\frac{T}{h} \times n$ for $n \leq N$.

The return value z is typically an $N \times n$ matrix, but scalar or vector-valued z are admissible, too. By default, the Euclidean inner product, the standard matrix multiplication is used:

```
dp = @(x,y) mtimes(x,y).
```

Other choices are for example: covariance-weighted products for Gaussian-noise-driven systems yielding system covariances [76], reproducing kernel Hilbert spaces (RKHS) [77], such as the polynomial, Gaussian or Sigmoid kernels [78], or energy-stable inner products [79]. Also, weighted Gramians [36] and time-weighted system Gramians [80] can be computed using this interface, i.e.,

```
dp = @(x,y) mtimes([0:h:T].^k.*x,y)
```

for a monomial of order k time-domain weighted inner product. Furthermore, the inner product interface may be used to directly compute the trace of an empirical Gramian by using a pseudo-kernel:

```
dp = @(x,y) sum(sum(x.*y'))
```

which exploits a property for computing the trace of a matrix product $\mathrm{tr}(AB) = \sum_i \sum_j A_{ij} B_{ji}$. This interface may also be used to compute only the empirical Gramian's diagonal:

```
dp = @(x,y) sum(x.*y',2)
```

for input-output importance [81] or input-output coherence [57] (Ch. 13). Lastly, it is noted that offloading matrix multiplications to an accelerator such as a GPGPU, motivated in Section 4.2.2, can also be achieved using this interface.

5.4. Option Flags

The vector `nf` contains 12 components, each representing an option with the default value zero and the following functionality:

`nf(1)` Time series centering:
- = 0 No centering,
- = 1 Steady-state (for empirical covariance matrices),
- = 2 Final state,
- = 3 Arithmetic average over time (for empirical Gramians),
- = 4 Root-mean-square over time,
- = 5 Mid-range over time.

`nf(2)` Input scale sequence:
- = 0 Single scale: `um` ← `um`,
- = 1 Linear scale subdivision: `um` ← `um * [0.25, 0.5, 0.75, 1.0]`,
- = 2 Geometric scale subdivision: `um` ← `um * [0.125, 0.25, 0.5, 1.0]`,
- = 3 Logarithmic scale subdivision: `um` ← `um * [0.001, 0.01, 0.1, 1.0]`,
- = 4 Sparse scale subdivision: `um` ← `um * [0.01, 0.5, 0.99, 1.0]`.

`nf(3)` Initial state scale sequence:
- = 0 Single scale: `xm` ← `xm`,
- = 1 Linear scale subdivision: `xm` ← `xm * [0.25, 0.5, 0.75, 1.0]`,
- = 2 Geometric scale subdivision: `xm` ← `xm * [0.125, 0.25, 0.5, 1.0]`,
- = 3 Logarithmic scale subdivision: `xm` ← `xm * [0.001, 0.01, 0.1, 1.0]`,
- = 4 Sparse scale subdivision: `xm` ← `xm * [0.01, 0.5, 0.99, 1.0]`.

`nf(4)` Input directions:
- = 0 Positive and negative: `um` ← `[-um, um]`,
- = 1 Only positive: `um` ← `um`.

`nf(5)` Initial state directions:
- = 0 Positive and negative: `xm` ← `[-xm, xm]`,
- = 1 Only positive: `xm` ← `xm`.

`nf(6)` Normalizing:
- = 0 No normalization,
- = 1 Scale with Gramian diagonal (see [82]),
- = 2 Scale with steady-state (see [34]).

`nf(7)` Non-Symmetric Cross Gramian, only W_X, W_Y, W_J:
- = 0 Regular cross Gramian,
- = 1 Non-symmetric cross Gramian.

`nf(8)` Extra input for state and parameter perturbation trajectories, only W_O, W_X, W_S, W_I, W_J:
- = 0 No extra input,
- = 1 Apply extra input (see [83]).

`nf(9)` Center parameter scales, only W_S, W_I, W_J:
- = 0 No centering,
- = 1 Center around arithmetic mean,
- = 2 Center around logarithmic mean.

`nf(10)` Parameter Gramian variant, only W_S, W_I, W_J:
- = 0 Average input-to-state (W_S), detailed Schur-complement (W_I, W_J),
- = 1 Average input-to-output (W_S), approximate Schur-complement (W_I, W_J).

`nf(11)` Empirical cross Gramian partition width, only W_X, W_J:
- = 0 Full cross Gramian computation, no partitioning.
- < N Maximum partition size in terms of cross Gramian columns.

`nf(12)` Partitioned empirical cross Gramian running index, only W_X, W_J:
- = 0 No partitioning.
- > 0 Index of the set of cross Gramian columns to be computed.

5.4.1. Schur-Complement

The observability-based parameter Gramians, the empirical identifiability Gramian W_I and the empirical cross-identifiability Gramian $W_{\tilde{I}}$, utilize an inversion as part of a (approximated) Schur-complement. Instead of using a Schur-complement solver or the pseudo-inverse, an approximate inverse with computational complexity $\mathcal{O}(N^2)$ based on [84] is utilized,

$$A^{-1} \approx D^{-1} - D^{-1}ED^{-1},$$

with the diagonal matrix D, $D_{ii} = A_{ii}$ and the matrix of off-diagonal elements $E = A - D$. This approximate inverse is used by default for W_I and $W_{\tilde{I}}$.

5.4.2. Partitioned Computation

The partitioned empirical cross Gramian (Section 4.2.3) can be configured by the option flags `nf(11)` and `nf(12)`, with `nf(11)` defining the maximum number of columns per partition, and `nf(12)` setting the running index of the current partition. Together with a partitioned singular value decomposition, such as the HAPOD [72], an empirical-cross-Gramian-based Galerkin projection is computable with minimal communication in parallel on a distributed memory system, or sequentially on a shared memory system [73].

5.5. Solver Configuration

To provide a problem-specific integrator to generate the state and output trajectories, a global variable named ODE is available, and expects a handle to a function with the signature:

```
y = ODE(f,g,t,x0,u,p)
```

comprising the arguments:

- `f` handle to a function with the signature `xdot = f(x,u,p,t)` representing the system's vector-field and expecting the arguments: current state `x`, current input `u`, (current) parameter `p` and current time `t`.
- `g` handle to a function with the signature `y = g(x,u,p,t)` representing the system's output functional and expecting the arguments: the current state `x`, current input `u`, (current) parameter `p` and current time `t`.
- `t` two component vector `t = [h,T]` specifying the time-step width h and time horizon T.
- `x0` column vector of dimension N setting the initial condition.
- `u` handle to a function with the signature `u_t = u(t)`.
- `p` column vector of dimension P holding the (current) parameter.

The solver is expected to return a discrete trajectory matrix of dimension $\dim(g(x(t),u(t),\theta,t)) \times \frac{T}{h}$.

As a default solver for (nonlinear) initial value problems, the optimal second-order strong stability preserving (SSP) explicit Runge-Kutta method [85] is included in `emgr`. This single-step integrator is implemented in a low-storage variant, and the stability of this method can be increased by additional stages, which is configurable by a global variable named STAGES. The default number of stages is STAGES $= 3$, yielding the SSP32 method.

5.6. Sample Usage

To illustrate the usage of `emgr`, the Matlab code for the computation of the empirical cross Gramian of a small linear system is presented in Figure 1. For demonstration purposes, this system has one uncontrollable and unobservable, one controllable and unobservable, one uncontrollable and observable, and one controllable and observable state:

$$A := -\frac{1}{2}\,\mathbb{1}, \quad B := \begin{pmatrix} 0 & 1 & 0 & 1 \end{pmatrix}^{\mathsf{T}}, \quad C := \begin{pmatrix} 0 & 0 & 1 & 1 \end{pmatrix}.$$

The cross Gramian computes as:

$$AW_X + W_X A = BC \Rightarrow W_X = BC,$$

which is approximately computed empirically following Section 3.1.4 in Figure 1.

```
M = 1;                                  % Number of inputs
N = 4;                                  % Number of states
Q = 1;                                  % Number of outputs
A = -0.5*eye(N);                        % System matrix
B = [0;1;0;1];                          % Input matrix
C = [0,0,1,1];                          % Output matrix
f = @(x,u,p,t) A*x + B*u;               % Vector field
g = @(x,u,p,t) C*x;                     % Output functional
h = 0.1;                                % Time step size
T = 10.0;                               % Time horizon
Wx = emgr(f,g,[M,N,Q],[h,T],'x'); % ≈ B*C
```

Figure 1. Sample code for the computation of the empirical cross Gramian of a non-minimal fourth order system, see Section 5.6.

6. Numerical Examples

In this section, empirical-Gramian-based model reduction techniques are demonstrated for three test systems using [68]; first, for a linear state-space symmetric MIMO system, second, for a hyperbolic SISO system, and third for a nonlinear SIMO system. All numerical tests are performed using OCTAVE 4.4 [86].

6.1. Linear Verification

The first example is a linear test system of the form (3) and generated using the inverse Lyapunov procedure [87], in a variant that enforces state-space symmetric systems [88]. For state-space symmetric systems, $A = A^\intercal$, $B = C^\intercal$, all system Gramians are symmetric and equal [89]. The system is configured to have $N = \dim(x(t)) = 256$ states and $\dim(u(t)) = \dim(y(t)) = 4$ inputs and outputs. For the computation of the reduced order model, the empirical linear cross Gramian with an impulse input $u_i(t) = \delta(t)$ and a zero initial state $x_{0,i} = 0$ is used, while for the trajectory simulation the default SSP32 (Section 5.5) integrator is utilized.

To quantify the quality of the resulting reduced order models, the error between the original full order model output and the reduced order model's output is compared in the time-domain $\mathcal{L}_2$-norm,

$$\|y - \tilde{y}\|_{\mathcal{L}_2} = \sqrt{\int_0^\infty \|y(t) - \tilde{y}(t)\|_2^2 \,\mathrm{d}t}.$$

In addition, the balanced truncation upper bound is assessed [41]:

$$\|y - \tilde{y}\|_{\mathcal{L}_2} \leq 2\|u\|_{\mathcal{L}_2} \sum_{k=n+1}^{N} \sigma_k,$$

for a reduced model of order n. Instead of impulse input, zero-centered, unit-variance Gaussian noise is used as input time series for the evaluation.

Figure 2 shows the reduced order model's relative $\mathcal{L}_2$-norm model reduction error, as well as the upper bound for increasing reduced orders. The error evolves correctly tightly below the bound, until the numerical accuracy (double-precision floating point arithmetic) is reached.

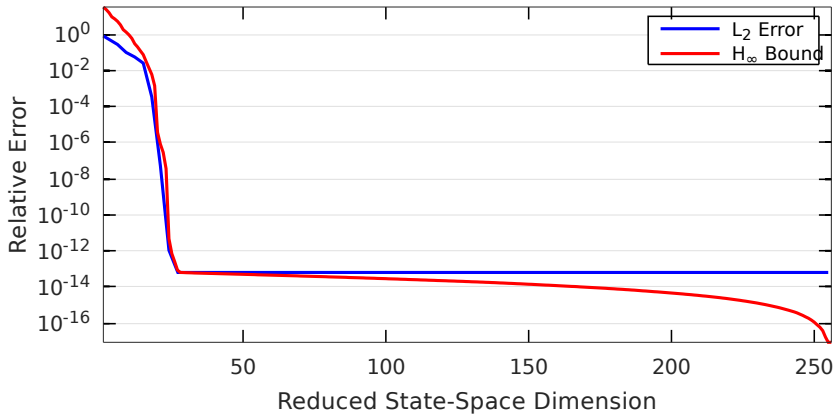

Figure 2. Model reduction error for the linear test system, see Section 6.1.

6.2. Hyperbolic Evaluation

The second numerical example is given by a one-dimensional transport equation. An input-output system is constructed by selecting the left boundary as the input and the right boundary as the output:

$$\begin{aligned} \frac{\partial}{\partial t} z(x,t) &= -\theta \frac{\partial}{\partial x}, \quad x \in [0,1], \\ z(0,t) &= u(t), \\ z(x,0) &= 0, \\ y(t) &= z(1,t), \end{aligned}$$

while the transport velocity $\theta \in [1.0, 1.5]$ is treated as a parameter. This partial differential equation system is spatially discretized by a first-order finite-difference upwind scheme, yielding a SISO ordinary differential equation system:

$$\begin{aligned} \dot{x}(t) &= A(\theta)x(t) + bu(t), \\ y(t) &= cx(t), \end{aligned}$$

with $A(\theta) = \theta A$. For this example, a spatial resolution of $\dim(x(t)) = 256$ is chosen, hence $A \in \mathbb{R}^{256\times256}$, $b \in \mathbb{R}^{256}$ and $c \in \mathbb{R}^{256}$. Since the system matrix is non-normal, using techniques such as POD may lead to unstable reduced order models. Thus, here a cross-Gramian-based balancing technique is used, guaranteeing stability of the reduced model. For training, impulse responses for the extremal velocities are used; for testing, a Gauss bell input is utilized over 10 uniformly random velocities, both utilizing the default integrator. The reduced order model quality is evaluated by the parametric norm [47]:

$$\|y(\theta) - \tilde{y}(\theta)\|_{\mathcal{L}_2 \otimes \mathcal{L}_2} = \sqrt{\int_\Theta \|y(\theta) - \tilde{y}(\theta_r)\|_{\mathcal{L}_2}^2 \, \mathrm{d}\theta}.$$

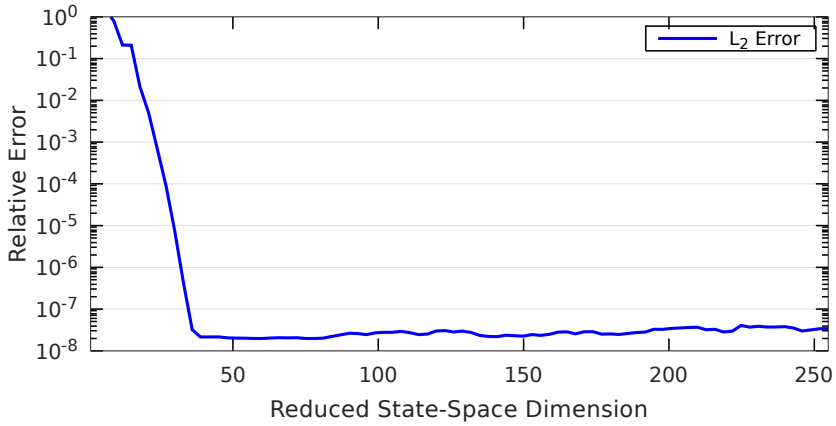

Figure 3. Model reduction error for the hyperbolic test system, see Section 6.2.

Even though the system is hyperbolic, a steep decay in error is obtained (Figure 3). Yet, due to the hyperbolicity and the parameter-dependence of the reduced order model, a lower overall numerical accuracy ($\approx 10^{-7}$) is achieved.

6.3. Nonlinear Validation

The third example involves a parametric nonlinear system, based on the hyperbolic network model [90],

$$\dot{x}(t) = A \tanh(K(\theta)x(t)) + Bu(t),$$
$$y(t) = Cx(t).$$

The structure of this system is similar to the linear system model (3), yet the vector-field includes a hyperbolic tangent nonlinearity, in which the parametrized activation is described by a diagonal gain matrix $K(\theta)$, $K_{ii} = \theta_i$. A negative Lehmer matrix (a Lehmer matrix is defined as $A_{ij} := \min(i,j)/\max(i,j)$, and is positive definite) is selected as system matrix $A \in \mathbb{R}^{256\times 256}$, a vector of sequential cosine evaluations as input matrix $B \in \mathbb{R}^{256\times 1}$, a binary matrix $C^{4\times 256}$ as output matrix, and parameters $\theta \in \mathbb{R}^{256}$ constrained to the interval $\theta_i \in [\frac{1}{2}, 1]$. For this system a combined state and parameter reduction is demonstrated. To this end an empirical non-symmetric joint Gramian is computed, using again an impulse input $u_i(t) = \delta(t)$, a zero initial state $x_{0,i} = 0$ and the default integrator. The reduced order model quality is evaluated for the same input and initial state by the joint state and parameter norm $\|y(\theta) - \tilde{y}(\theta_r)\|_{\mathcal{L}_2 \otimes \mathcal{L}_2}$, with respect to the reduced parameters for ten uniformly random samples from the admissible parameter-space.

Figure 4 depicts the $\mathcal{L}_2 \otimes \mathcal{L}_2$-norm model reduction error for increasing state- and parameter-space dimensions. The combined reduction errors decay for both: The reduced state-space and reduced parameter-space, yet faster for the state-space. As for the parametric model, the numerical accuracy is reduced due to the combined reduction.

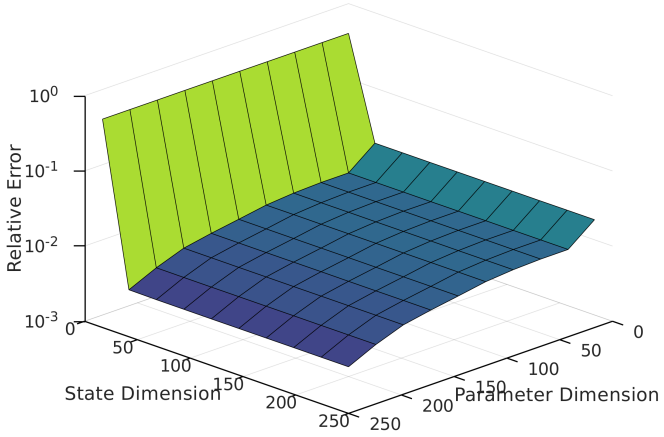

Figure 4. Model reduction error for the nonlinear test system, see Section 6.3.

6.4. On Hyper-Reduction

Projection-based model reduction for nonlinear systems (such as in Section 6.3) leads to a computational issue, the so-called lifting bottleneck, which results from the composition of the reconstructing projection U_1 with the original, (nonlinear) vector field f and the reducing projection V_1:

$$f_r := V_1^{\mathsf{T}} \circ f \circ U_1,$$

in the reduced vector field (2). This means for nonlinear systems, the high-dimensional state has to be used in the reduced order model, as opposed to linear systems (3), where the truncated projections can be applied directly to the linear vector field components (The composition operators become matrix-matrix multiplications in the linear case) beforehand.

To reduce this computational complexity, hyper-reduction methods are utilized. Examples of hyper-reduction methods are: Gappy-POD [91], Missing Point Estimation (MPE) [92], Discrete Empirical Interpolation Method (DEIM) [93], Dynamic Mode Decomposition (DMD) [94] or numerical linearization [95]; for a comparison see [96]. These methods construct (linear) reduced representations of the nonlinearity in a data-driven manner.

`emgr` does not address hyper-reduction and thus it has to be applied in a separate post-processing step [97]. Yet, the trajectory data used to assemble the empirical Gramians can be recycled by interposing the hyper-reduction computation via the solver configuration (Section 5.5).

7. Concluding Remark

Empirical Gramians are a universal tool for nonlinear system and control theoretic applications with a simple, data-driven construction. The empirical Gramian framework - `emgr` - implements empirical Gramian computation for system input-output coherence and parameter identifiability evaluation. Possible future extensions of `emgr` may include Koopman Gramians [98], empirical Riccati covariance matrices [99], or empirical differential balancing [100]. Finally, further examples and applications can be found at the `emgr` project website: http://gramian.de.

Code Availability

The source code of the presented numerical examples can be obtained from:

http://runmycode.org/companion/view/2077

and is authored by: CHRISTIAN HIMPE.

Funding: Supported by the German Federal Ministry for Economic Affairs and Energy, in the joint project: "**MathEnergy**—Mathematical Key Technologies for Evolving Energy Grids", sub-project: Model Order Reduction (Grant number: 0324019**B**). Open access funding provided by Max Planck Society.

Conflicts of Interest: The author declares no conflict of interest.

Abbreviations

The following abbreviations are used in this manuscript:

PDE	Partial Differential Equation
ODE	Ordinary Differential Equation
MOR	Model Order Reduction
pMOR	parametric Model Order Reduction
nMOR	nonlinear Model Order Reduction
POD	Proper Orthogonal Decomposition
bPOD	balanced Proper Orthogonal Decomposition
SVD	Singular Value Decomposition
HSV	Hankel Singular Values
EVD	Eigenvalue Decomposition
SISO	Single-Input-Single-Output
MIMO	Multiple-Input-Multiple-Output
BLAS	Basic Linear Algebra System
LoC	Lines of Code
SIMD	Single Instruction Multiple Data
UMA	Unified Memory Access
GPGPU	General Purpose Graphics Processing Unit
GPU	Graphics Processing Unit
UMM	Unified Memory Model
hUMA	heterogeneous Unified Memory Access
CPU	Central Processing Unit
GEMM	GEneralized Matrix Multiplication
HAPOD	Hierarchical Approximate Proper Orthgoonal Decomposition
RKHS	Reproducing Kernel Hilbert Spaces
SSP	Strong Stability Preserving
SIMO	Single-Input-Multiple-Output
MPE	Missing Point Estimation
DEIM	Discrete Empirical Interpolation Method
DMD	Dynamic Mode Decomposition

References

1. Kalman, R.E. Mathematical description of linear dynamical systems. *SIAM J. Control Optim.* **1963**, *1*, 182–192. [CrossRef]
2. Lall, S.; Marsden, J.E.; Glavaški, S. Empirical model reduction of controlled nonlinear systems. *IFAC Proc. Vol.* **1999**, *32*, 2598–2603. [CrossRef]
3. Himpe, C. `emgr`—EMpirical GRamian Framework (Version 5.4). Available online: http://gramian.de (accessed on 26 June 2018).
4. Moore, B.C. Principal component analysis in linear systems: Controllability, observability, and model reduction. *IEEE Trans. Autom. Control* **1981**, *26*, 17–32. [CrossRef]
5. Sun, C.; Hahn, J. Model reduction in the presence of uncertainty in model parameters. *J. Process Control* **2006**, *16*, 645–649. [CrossRef]
6. Himpe, C.; Ohlberger, M. The Empirical Cross Gramian for Parametrized Nonlinear Systems. *IFAC-PapersOnLine* **2015**, *48*, 727–728. [CrossRef]
7. Hahn, J.; Edgar, T.F. Reduction of nonlinear models using balancing of empirical Gramians and Galerkin projections. In Proceedings of the 2000 American Control Conference, Chicago, IL, USA, 28–30 June 2000; Volume 4, pp. 2864–2868.

8. Condon, M.; Ivanov, R. Model reduction of nonlinear systems. *Compel-Int. J. Comp. Math. Electr. Electron. Eng.* **2004**, *23*, 547–557. [CrossRef]
9. Yao, S.; Deng, Y.; Yu, Z. Balanced Truncation on Empirical Gramians for Model-Order-Reduction of Non-Quasi-Static Effects in MOSFETs. In Proceedings of the 9th International Conference on Solid-State and Integrated-Circuit Technology, Beijing, China, 20–23 October 2008; pp. 309–312.
10. Zhanfeng, M.; Chao, H. Structure-preserving balanced truncation for flexible spacecraft using cross Gramian. *J. Beijing Univ. Aeronaut. Astronaut.* **2008**, *34*, 1437–1440.
11. Himpe, C.; Ohlberger, M. Cross-Gramian Based Combined State and Parameter Reduction for Large-Scale Control Systems. *Math. Probl. Eng.* **2014**, *2014*, 1–13. [CrossRef]
12. Streif, S.; Findeisen, R.; Bullinger, E. Relating Cross Gramians and Sensitivity Analysis in Systems Biology. *Theory Netw. Syst.* **2006**, *10*, 437–442.
13. Lystianingrum, V.; Hredzak, B.; Agelidis, V.G. Abnormal overheating detectability analysis based on cross Gramian for a supercapacitors string. In Proceedings of the Power and Energy Society General Meeting, Boston, MA, USA, 17–21 July 2016.
14. Geffen, D.; Findeisen, R.; Schliemann, M.; Allgöwer, F.; Guay, M. Observability Based Parameter Identifiability for Biochemical Reaction Networks. In Proceedings of the 2008 American Control Conference, Seattle, WA, USA, 11–13 June 2008; pp. 2130–2135.
15. Tolks, C.; Ament, C. Model Order Reduction of Glucose-Insulin Homeostasis Using Empirical Gramians and Balanced Truncation. *IFAC-PapersOnline* **2017**, *50*, 14735–14740. [CrossRef]
16. Moaveni, B.; Khaki-Sedigh, A. Input-Output Pairing based on Cross-Gramian Matrix. In Proceedings of the International Joint Conference SICE-ICAS, Busan, Korea, 18–21 October 2006; pp. 2378–2380.
17. Shaker, H.R.; Komareji, M. Control Configuration Selection for Multivariable Nonlinear Systems. *Ind. Eng. Chem. Res.* **2012**, *51*, 8583–8587. [CrossRef]
18. Shaker, H.R.; Stoustrup, J. An interaction measure for control configuration selection for multivariable bilinear systems. *Nonlinear Dyn.* **2013**, *72*, 165–174. [CrossRef]
19. Singh, A.K.; Hahn, J. Determining Optimal Sensor Locations for State and Parameter Estimation for Stable Nonlinear Systems. *Ind. Eng. Chem. Res.* **2005**, *44*, 5645–5659. [CrossRef]
20. Saltik, M.B.; Özkan, L.; Weiland, S.; van den Hof, P.M.J. Sensor Configuration Problem: Application to a Membrane Separation Unit. *IFAC-PapersOnLine* **2016**, *49*, 189–194. [CrossRef]
21. Summers, T.H.; Cortesi, F.L.; Lygeros, J. On Submodularity and Controllability in Complex Dynamical Networks. *IEEE Trans. Control Netw. Syst.* **2016**, *3*, 91–101. [CrossRef]
22. Lawrence, D.; Myatt, J.H.; Camphouse, R.C. On Model Reduction via Empirical Balanced Truncation. In Proceedings of the American Control Conference, Portland, OR, USA, 8–10 June 2005; Volume 5, pp. 3139–3144.
23. Hahn, J.; Kruger, U.; Edgar, T.F. Application of Model Reduction for Model Predictive Control. *IFAC Proc. Vol.* **2002**, *15*, 393–398. [CrossRef]
24. Hahn, J.; Edgar, T.F. A Gramian Based Approach to Nonlinearity Quantification and Model Classification. *Ind. Eng. Chem. Res.* **2001**, *40*, 5724–5731. [CrossRef]
25. Jiang, M.; Wu, J.; Jiang, L.; Li, X. A Gramians Based Method for Nonlinearity Quantification of Spatio-Temporal Systems. In *Advanced Science and Technology Letters*; SERSC: Haikou, China, 2016; Volume 121, pp. 38–42.
26. Fernando, K.V.; Nicholson, H. On the Cauchy Index of Linear Systems. *IEEE Trans. Autom. Control* **1983**, *28*, 222–224. [CrossRef]
27. Fortuna, L.; Fransca, M. *Optimal and Robust Control: Advanced Topics with MATLAB*; CRC Press: Boca Raton, FL, USA, 2012.
28. Fu, J.; Zhong, C.; Ding, Y.; Zhou, J.; Zhong, C. An Information Theoretic Approach to Model Reduction based on Frequency-domain Cross-Gramian Information. In Proceedings of the 8th World Congress on Intelligent Control and Automation, Jinan, China, 7–9 July 2010; pp. 3679–3683.
29. Halvarsson, B.; Castaño, M.; Birk, W. Uncertainty Bounds for Gramian-Based Interaction Measures. In Proceedings of the 14th WSEAS international conference on Systems: part of the 14th WSEAS CSCC multiconference, Corfu Island, Greece, 22–24 July 2010; Volume 2, pp. 393–398.
30. Hrishikeshavan, V.; Humbert, J.S.; Chopra, I. Gramian Analysis of a Shrouded Rotor Micro Air Vehicle in Hover. *J. Guid. Control Dyn.* **2014**, *37*, 1684–1690. [CrossRef]

31. Gugercin, S.; Antoulas, A.C.; Beattie, C. $\mathcal{H}_2$ Model Reduction for Large-Scale Linear Dynamical Systems. *SIAM J. Matrix Anal. Appl.* **2008**, *30*, 609–638. [CrossRef]
32. Willcox, K.; Peraire, J. Balanced Model Reduction via the Proper Orthogonal Decomposition. *AIAA J.* **2002**, *40*, 2323–2330. [CrossRef]
33. Rowley, C.W. Model reduction for fluids, using balanced proper orthogonal decomposition. *Int. J. Bifurcat. Chaos* **2005**, *15*, 997–1013. [CrossRef]
34. Sun, C.; Hahn, J. *Nonlinear Model Reduction Routines for MATLAB*; Technical Report; Rensselaer Polytechnic Institute: Troy, NY, USA, 2006.
35. Hahn, J.; Edgar, T.F. Balancing Approach to Minimal Realization and Model Reduction of Stable Nonlinear Systems. *Ind. Eng. Chem. Res.* **2002**, *41*, 2204–2212. [CrossRef]
36. Choroszucha, R.B.; Sun, J.; Butts, K. Nonlinear Model Order Reduction for Predictive Control of the Diesel Engine Airpath. In Proceedings of the American Control Conference, Boston, MA, USA, 6–8 July 2016; pp. 5081–5086.
37. Krener, A.; Ide, K. Measures of Unobservability. In Proceedings of the 48th IEEE Conference on Decision and Control, 2009 held jointly with the 2009 28th Chinese Control Conference, Shanghai, China, 16–18 December 2009; pp. 6401–6406.
38. Himpe, C.; Ohlberger, M. A Unified Software Framework for Empirical Gramians. *J. Math.* **2013**, *2013*, 1–6. [CrossRef]
39. Himpe, C.; Ohlberger, M. A note on the cross Gramian for non-symmetric systems. *Syst. Sci. Control Eng.* **2016**, *4*, 199–208. [CrossRef]
40. Garcia, J.S.; Basilio, J.C. Computation of reduced-order models of multivariable systems by balanced truncation. *Int. J. Syst. Sci.* **2002**, *33*, 847–854. [CrossRef]
41. Antoulas, A.C. *Approximation of Large-Scale Dynamical Systems Volume 6 Advances in Design and Control*; SIAM Publications: Philadelphia, PA, USA, 2005.
42. Aldhaheri, R.W. Model order reduction via real Schur-form decomposition. *Int. J. Control* **1991**, *53*, 709–716. [CrossRef]
43. Baur, U.; Benner, P. Gramian-Based Model Reduction for Data-Sparse Systems. *SIAM J. Sci. Comput.* **2008**, *31*, 776–798. [CrossRef]
44. Sorensen, D.C.; Antoulas, A.C. The Sylvester equation and approximate balanced reduction. *Numer. Linear Algebra Appl.* **2002**, 671–700. [CrossRef]
45. Hahn, J.; Edgar, T.F. An improved method for nonlinear model reduction using balancing of empirical Gramians. *Comput. Chem. Eng.* **2002**, *26*, 1379–1397. [CrossRef]
46. Sun, C.; Hahn, J. Parameter reduction for stable dynamical systems based on Hankel singular values and sensitivity analysis. *Chem. Eng. Sci.* **2006**, *61*, 5393–5403. [CrossRef]
47. Himpe, C. Combined State and Parameter Reduction for Nonlinear Systems with an Application in Neuroscience. Ph.D. Thesis, Westfälische Wilhelms-Universität Münster, Münster, Germany, 2017.
48. Keil, A.; Gouzé, J.L. *Model Reduction of Modular Systems Using Balancing Methods*; Technical Report; Technische Universität München: München, Germany, 2003.
49. Stigter, J.D.; van Willigenburg, L.G.; Molenaar, J. An Efficient Method to Assess Local Controllability and Observability for Non-Linear Systems. *IFAC-PapersOnLine* **2018**, *51*, 535–540. [CrossRef]
50. Hespanha, J. *Linear Systems Theory*; Princeton University Press: Princeton, NJ, USA, 2009.
51. Ma, X.; De Abreu-Garcia, J.A. On the Computation of Reduced Order Models of Nonlinear Systems using Balancing Technique. In Proceedings of the 27th IEEE Conference on Decision and Control, Austin, TX, USA, 7–9 December 1988; Volume 2, pp. 1165–1166.
52. Singh, A.K.; Hahn, J. On the Use of Empirical Gramians for Controllability and Observability Analysis. In Proceedings of the 2005 American Control Conference, Portland, OR, USA, 8–10 June 2005; Volume 2005, pp. 140–141.
53. Dones, I.; Skogestad, S.; Preisig, H.A. Application of Balanced Truncation to Nonlinear Systems. *Ind. Eng. Chem. Res.* **2011**, *50*, 10093–10101. [CrossRef]
54. Scherpen, J.M.A. Balancing for nonlinear systems. *Syst. Control Lett.* **1993**, *21*, 143–153. [CrossRef]
55. Hahn, J.; Edgar, T.F.; Marquardt, W. Controllability and observability covariance matrices for the analysis and order reduction of stable nonlinear systems. *J. Process Control* **2003**, *13*, 115–127. [CrossRef]

56. Fernando, K.V.; Nicholson, H. On the Structure of Balanced and Other Principal Representations of SISO Systems. *IEEE Trans. Autom. Control* **1983**, *28*, 228–231. [CrossRef]
57. Fernando, K.V. Covariance and Gramian Matrices in Control and Systems Theory. Ph.D. Thesis, University of Sheffield, Sheffield, UK, 1982.
58. Fernando, K.V.; Nicholson, H. On the Cross-Gramian for Symmetric MIMO Systems. *IEEE Trans. Circuits Syst.* **1985**, *32*, 487–489. [CrossRef]
59. Shaker, H.R. Generalized Cross-Gramian for Linear Systems. In Proceedings of the 7th IEEE Conference on Industrial Electronics and Applications (ICIEA), Singapore, 18–20 July 2012; pp. 749–751.
60. Baur, U.; Benner, P.; Haasdonk, B.; Himpe, C.; Martini, I.; Ohlberger, M. Comparison of Methods for Parametric Model Order Reduction of Time-Dependent Problems. In *Model Reduction and Approximation: Theory and Algorithms*; Benner, P., Cohen, A., Ohlberger, M., Willcox, K., Eds.; SIAM: Philadelphia, PA, USA, 2017; pp. 377–407.
61. Barbagallo, A.; De Felice, V.F.; Nagarajan, K.K. Reduced Order Modelling of a Couette Flow Using Balanced Proper Orthogonal Decomposition. In Proceedings of the 2nd Young ERCOFTAC Workshop, Montestigliano, Italy, 10 March 2008.
62. Ionescu, T.C.; Fujimoto, K.; Scherpen, J.M.A. Singular Value Analysis of Nonlinear Symmetric Systems. *IEEE Trans. Autom. Control* **2011**, *56*, 2073–2086. [CrossRef]
63. Fujimoto, K.; Scherpen, J.M.A. On balanced truncation for symmetric nonlinear systems. In Proceedings of the International Symposium on Mathematical Theory of Networks and Systems, Groningen, The Netherlands, 7–11 July 2014; Volume 21, pp. 1498–1502.
64. Constantine, P. *Active Subspaces: Emerging Ideas for Dimension Reduction in Parameter Studies*; SIAM Spotlights, SIAM: Philadelphia, PA, USA, 2015.
65. Lieberman, C.E.; Fidkowski, K.; Willcox, K.; van Bloemen Waanders, B. Hessian-based model reduction: Large-scale inversion and prediction. *Int. J. Numer. Methods Fluids* **2013**, *71*, 135–150. [CrossRef]
66. Jazlan, A.; Sreeram, V.; Togneri, R. Cross Gramian Based Time Interval Model Reduction. In Proceedings of the 5th Australien Control Conference (AUCC), Gold Coast, Australia, 5–6 November 2015; pp. 274–276.
67. The MathWorks, Inc. MATLAB. Available online: http://www.matlab.com (accessed on 26 June 2018).
68. The Octave Developers. GNU Octave. Available online: http://octave.org (accessed on 26 June 2018).
69. Johnson, R.K. *The Elements of MATLAB Style*; Cambridge University Press: Cambridge, UK, 2011.
70. Altman, Y.M. *Accelerating MATLAB Performance: 1001 Tips to Speed up MATLAB Programs*; CRC Press: Boca Raton, FL, USA, 2015.
71. Rogers, P.; Marci, J.; Marinkovic, S. *Heterogeneous Uniform Memory Access*; AMD: Santa Clara, CA, USA, 2013.
72. Himpe, C.; Leibner, T.; Rave, S. Hierarchical Approximate Proper Orthogonal Decomposition. *arXiv* **2018**, arXiv:1607.05210.
73. Himpe, C.; Leibner, T.; Rave, S.; Saak, J. Fast Low-Rank Empirical Cross Gramians. *Proc. Appl. Math. Mech.* **2017**, *17*, 841–842. [CrossRef]
74. Condon, M.; Ivanov, R. Empirical Balanced Truncation of Nonlinear Systems. *J. Nonlinear Sci.* **2004**, *14*, 405–414. [CrossRef]
75. Nilsson, O.; Rantzer, A. A novel approach to balanced truncation of nonlinear systems. In Proceedings of the 2009 European Control Conference (ECC), Budapest, Hungary, 23–26 August 2009.
76. Nikiforuk, P.N.; Gupta, M.M. On stochastic perturbation theory for linear systems. In Proceedings of the 1969 IEEE Symposium on Adaptive Processes (8th) Decision and Control, University Park, PA, USA, 17–19 November 1969.
77. Bouvrie, J.; Hamzi, B. Kernel Methods for the Approximation of Nonlinear Systems. *SIAM J. Control Optim.* **2017**, *55*, 2460–2492. [CrossRef]
78. Fasshauer, G.; McCourt, M. *Kernel-Based Approximation Methods Using MATLAB Volume 19 Interdisciplinary Mathematical Sciences*; World Scientific: Singapore, 2015.
79. Kalashnikova, I.; Barone, M.; Arunajatesan, S.; van Bloemen Waanders, B. Construction of energy-stable projection-based reduced order models. *Appl. Math. Comput.* **2014**, *249*, 569–596. [CrossRef]
80. Schelfhout, G.; de Moor, B. Time-Domain Weighted Balanced Truncation. In Proceedings of the 3rd European Control Conference, Rome, Italy, 5–8 September 1995; pp. 1–4.
81. Snowden, T.J.; van der Graaf, P.H.; Tindall, M.J. A combined model reduction algorithm for controlled biochemical systems. *BMC Syst. Biol.* **2017**, *11*, 1–18. [CrossRef] [PubMed]

82. Eberle, C.; Ament, C. Identifiability and online estimation of diagnostic parameters with in the glucose insulin homeostasis. *Biosystems* **2012**, *107*, 135–141. [CrossRef] [PubMed]
83. Powel, N.D.; Morgansen, K.A. Empirical Observability Gramian Rank Condition for Weak Observability of Nonlinear Systems with Control. In Proceedings of the 54th Annual Conference on Decision and Control, Osaka, Japan, 15–18 December 2015; pp. 6342–6348.
84. Wu, M.; Yin, B.; Vosoughi, A.; Studer, C.; Cavallaro, J.R.; Dick, C. Approximate Matrix Inversion for High-Throughput Data Detection in the Large-Scale MIMO Uplink. In Proceedings of the 2013 IEEE International Symposium on Circuits and Systems (ISCAS2013), Beijing, China, 19–23 May 2013; pp. 2155–2158.
85. Ketcheson, D.I. Highly Efficient Strong Stability-Preserving Runge-Kutta Methods with Low-Storage Implementations. *SIAM J. Sci. Comput.* **2008**, *30*, 2113–2136. [CrossRef]
86. Eaton, J.W.; Bateman, D.; Hauberg, S.; Wehbring, R. GNU Octave Version 4.4.0 Manual: A High-Level Interactive Language for Numerical Computations. Available online: https://octave.org/octave.pdf (accessed on 26 June 2018).
87. The MORwiki Community. MORwiki-Model Order Reduction Wiki. Available online: http://modelreduction.org (accessed on 26 June 2018).
88. Himpe, C.; Ohlberger, M. Cross-Gramian-Based Model Reduction: A Comparison. In *Model Reduction of Parametrized Systems*; Benner, P., Ohlberger, M., Patera, A., Rozza, G., Urban, K., Eds.; Springer: Cham, Switzerland, 2017; Volume 17, pp. 271–283.
89. Liu, W.Q.; Sreeram, V.; Teo, K.L. Model reduction for state-space symmetric systems. *Syst. Control Lett.* **1998**, *34*, 209–215. [CrossRef]
90. Quan, Y.; Zhang, H.; Cai, L. Modeling and Control Based on a New Neural Network Model. In Proceedings of the American Control Conference, Arlington, VA, USA, 25–27 June 2001; Volume 3, pp. 1928–1929.
91. Everson, R.; Sirovich, L. Karhunen-Loéve Procedure for Gappy Data. *J. Opt. Soc. Am. A* **1995**, *12*, 1657–1664. [CrossRef]
92. Astrid, P. Fast Reduced Order Modeling Technique for Large Scale LTV Systems. In Proceedings of the American Control Conference, Boston, MA, USA, 30 June–2 July 2004; pp. 762–767.
93. Chaturantabut, S.; Sorensen, D.C. Nonlinear model reduction via discrete empirical interpolation. *SIAM J. Sci. Comput.* **2010**, *32*, 2737–2764. [CrossRef]
94. Williams, M.O.; Schmid, P.J.; Kutz, J.N. Hybrid Reduced-Order Integration with Proper Orthogonal Decomposition and Dynamic Mode Decomposition. *Multiscale Model. Simul.* **2013**, *11*, 522–544. [CrossRef]
95. Moore, B.C. Principal Component Analysis in Nonlinear Systems: Preliminary Results. In Proceedings of the 18th IEEE Conference on Decision and Control including the Symposium on Adaptive Processes, Fort Lauderdale, FL, USA, 12–14 December 1979; Volume 2, pp. 1057–1060.
96. Dimitriu, G.; Ştefănescu, R.; Navon, I.M. Comparative numerical analysis using reduced-order modeling strategies for nonlinear large-scale systems. *J. Comput. Appl. Math.* **2017**, *310*, 32–42. [CrossRef]
97. Melchior, S.; Legat, V.; van Dooren, P. Gramian Based Model Reduction of Nonlinear MIMO Systems. In Proceedings of the Mathematical Theory of Networks and Systems, Melbourne, Australia, 9–13 July 2012.
98. Yeung, E.; Liu, Z.; Hodas, N.O. A Koopman Operator Approach for Computing and Balancing Gramians for Discrete Time Nonlinear Systems. *arXiv* **2017**, arXiv:1709.08712.
99. Choroszucha, R.B.; Sun, J. Empirical Riccati covariance matrices for closed-loop model order reduction of nonlinear systems by balanced truncation. In Proceedings of the American Control Conference, Seattle, WA, USA, 24–26 May 2017; pp. 3476–3482.
100. Kawano, Y.; Scherpen, J.M.A. Empirical Differential Balancing for Nonlinear Systems. *IFAC-PapersOnLine* **2017**, *50*, 6326–6331. [CrossRef]