Model reduction of modular systems using balancing methods

Andreas Keil*‡ and Jean-Luc Gouzé*§

Contents

*Institut National de Recherche en Informatique et en Automatique, Projet Comore, 2004 Route des Lucioles, B.P. 93, 06902 Sophia-Antipolis cedex, France
‡andreas.keil@gmx.de
§jean-luc.gouze@inria.fr

1. Introduction

The goal of this work is to reduce the order of state space models describing biological or chemical networks. The state equations are assumed to be (generally) nonlinear, time-invariant, ordinary differential equations which build a modular structure together. This structure should be preserved during reduction, which can be achieved by applying existing reduction methods to modular parts of the whole system.

Remark that the investigated methods here are linear which means that the transformation and projection are linear operations. However, applying them to nonlinear systems most likely produces nonlinear systems again. There exist also nonlinear methods for nonlinear systems but they are computationally difficult and not investigated here.

Model reduction via proper orthogonal decomposition (POD), also known as Karhunen-Loève expansion or principal component analysis, does not take the input-output behavior into account but rather focuses on the most energetic dynamics of the state space. In our case the I/O behavior should be approximated and therefore balanced truncation seems to be a suitable reduction method.

But first, we start with a review of methods for model reduction via balancing transformations (for linear systems as well as for nonlinear systems) without focusing on modular systems.

2. Model reduction via balancing techniques

Model reduction is often being carried out in two major steps:

1. Find a transformation of the state space so that the new basis allows an identification of the important (i.e. in terms of the input-output behavior) subspace.

2. Project the system (by singular perturbation or simple truncation) onto this important subspace.

The first step transforms a system[1]

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

by $z(t) = Tx(t)$ to

$$\begin{aligned} \dot{z}(t) &= \widehat{f}\big(z(t), u(t)\big) := Tf\big(T^{-1}z(t), u(t)\big) \\ y(t) &= \widehat{g}\big(z(t)\big) \qquad\quad := g\big(T^{-1}z(t)\big), \end{aligned} \tag{2}$$

[1]Note that throughout this document any direct throughput from u to y is left out of consideration since it would not be affected by state space transformations and model reduction.

where $u(t) \in \mathbb{R}^p$ is the input function, $x(t) \in \mathbb{R}^n$ and $z(t) \in \mathbb{R}^n$ are the original and transformed state variables, respectively, $y(t) \in \mathbb{R}^q$ is the output, and $T \in \mathbb{R}^{n \times n}$ is a non-singular linear transformation.

Since the reduced order model should approximate the input-output behavior of the system, one searches a state space realization reflecting the grade of contribution to the I/O behavior. Such a system realization is the "balanced realization". A balanced system has states that are "as good to control as to observe". That means that any vector of the state space always has the same magnitude of controllability and observability which can be measured using the corresponding gramians.

2.1. Gramians for linear, time-invariant systems

Calculation of those gramians is explained for *linear* systems $(A, B, C, 0)$ first, where

$$\begin{aligned} f\big(x(t), u(t)\big) &= Ax(t) + Bu(t) \\ g\big(x(t)\big) &= Cx(t) \end{aligned}$$

has to be set in equations (1) and (2).

Note that vectors and matrices have real coefficients throughout this document and that the conjugate complex transpose is used even if the transpose would produce the same result.

Definition 1 (Controllability gramian) *Given a pair (A, B), the positive semidefinite matrix*

$$X_t = \int_0^t \mathrm{e}^{A\tau} B B^* \mathrm{e}^{A^*\tau} \,\mathrm{d}\tau$$

is called the controllability gramian of the pair (A, B) at time t.

Theorem 2 *The set of reachable states is given by*

$$\mathcal{R} := \operatorname{im} X_t = \operatorname{im} \begin{pmatrix} B & AB & \dots & A^{n-1}B \end{pmatrix},$$

and is therefore constant for all $t > 0$.

The input function with minimal energy $\|u\|$ needed to drive a system (A, B) to a desired state x_t from the set of reachable states $\mathcal{R}$ *(assuming zero initial values) can be determined by calculating the solution v to*

$$X_t v = x_t,$$

and setting

$$u(\tau) = B^* \mathrm{e}^{A^*(t-\tau)} v, \quad \text{for } 0 \le \tau \le t.$$

The squared norm of this input function is then

$$\|u\|^2 = \int_0^t u^*(\tau) u(\tau) \,\mathrm{d}\tau = v^* X_t v.$$

When speaking of the controllability gramian, one normally refers to the following version of the gramian, because it allows a judgment over controllability without a limited time horizon:

Theorem 3 (Limit of the controllability gramian and the Lyapunov equation) *Let A be Hurwitz (all eigenvalues have negative real parts), then the limit of the controllability gramian*

$$X_\infty = \int_0^\infty \mathrm{e}^{At} B B^* \mathrm{e}^{A^* t} \,\mathrm{d}t \tag{3}$$

is the unique solution to the Lyapunov equation

$$A X_\infty + X_\infty A^* = -B B^*. \tag{4}$$

If BB^* is positive definite, then X_∞ is also positive definite.

Corollary 4 *If and only if (A, B) is controllable, then the controllability gramian X_t is positive definite (and therefore invertible), $\mathcal{R} = \mathbb{R}^n$, and the energy needed to drive the system to a reachable state x_t at time t is given by*

$$\|u\|^2 = x_t^* X_t^{-1} x_t.$$

Corollary 4 also applies to the infinite version of the controllability gramian. Thus the minimum squared energy of an input function u needed to reach a given state x_∞ in infinite time (which could be as steady state) is $x_\infty^* X_\infty^{-1} x_\infty$ (assuming a controllable pair (A, B)). That means that state vectors corresponding to big singular values of X_∞^{-1} need a high energy input function to be reached. The same states correspond to the reciprocal (and therefore small) singular values of X_∞. (If (A, B) is not controllable, the unreachable states correspond to zero singular values of the controllability gramian.)

Therefore, we can say that states corresponding to small (or zero) singular values of X_∞ are "hard to reach" and contribute little to the input-to-state behavior of the system.

Remark. Equation (3) is never used to compute the controllability gramian since one would have to calculate the limit of the integral for that. The Lyapunov equation (4) is essentially a linear equation for the entries of the controllability gramian and can therefore be used to calculate the gramian. But instead of using the Gauß-Algorithm, singular value and Schur decompositions are normally applied to solve it (taking advantage of the symmetry of X_∞ and without performing the multiplication BB^*).

After having examined the input-to-state behavior using the controllability gramian, it is a logical consequence to examine the state-to-output behavior using an observability gramian.

Definition 5 (Observability gramian) *Given a pair (C, A), the positive semidefinite matrix*

$$Y_t = \int_0^t \mathrm{e}^{A^* \tau} C^* C \mathrm{e}^{A \tau} \,\mathrm{d}\tau$$

is called the observability gramian of the pair (C, A) at time t.

Theorem 3 also applies in a dual sense ($A \leftrightarrow A^*$, $C \leftrightarrow B^*$) to the observability gramian, where the infinite version solves the Lyapunov equation

$$A^* Y_\infty + Y_\infty A = -C^* C,$$

the set of observable states is given by

$$\mathcal{O} := \operatorname{im} Y_t = \operatorname{im} \begin{pmatrix} C^* & A^* C^* & \dots & (A^*)^{n-1} C^* \end{pmatrix} = \left\{ \ker \begin{pmatrix} C \\ CA \\ \vdots \\ CA^{n-1} \end{pmatrix} \right\}^\perp,$$

and the squared energy of the output function y produced by a given initial state x_0 and zero input is

$$\|y\|^2 = \int_0^\infty y^*(t)y(t)\,\mathrm{d}t = \int_0^\infty x_0^* \mathrm{e}^{A^*t} C^* C \mathrm{e}^{At} x_0\,\mathrm{d}t = x_0^* Y_\infty x_0.$$

That means that state vectors corresponding to small singular values of Y_∞ produce low energy output functions. (If (C, A) is not observable, the unobservable states correspond to zero singular values of the observability gramian.)

Therefore, we can say that states corresponding to small (or zero) singular values of Y_∞ are "weakly observable" and contribute little to the state-to-output behavior of the system.

2.2. Empirical gramians for nonlinear systems

Since nonlinear systems cannot be balanced by calculating linear gramians, the paper [6] proposes an empirical way of calculating the subspace of interest and to proceed then by calculating a linear transformation in the same way as for balancing linear systems. The empirical counterparts to linear systems' gramians are based on data that is gained from experiments or (like in our case) from simulations.

For the definitions of the empirical gramians, we need a set $\{T_1, \ldots, T_r\}$ of orthogonal test matrices and a set $\{c_1, \ldots, c_s\}$ of positive scalar constants. The choice for those two sets is normally different for the calculation of the two gramians (especially because the dimensions of the set of test matrices have to correspond to the dimension of the input and state space, respectively). Therefore, we will refer to them as $\mathcal{T}_\mathcal{X}$ / $\mathcal{C}_\mathcal{X}$, and $\mathcal{T}_\mathcal{Y}$ / $\mathcal{C}_\mathcal{Y}$, for the empirical controllability and observability gramian, respectively. Furthermore denote the i^{th} unit vector with e_i and define the mean value of a function $v(t)$ by

$$\bar{v} := \lim_{t\to\infty} \frac{1}{t} \int_0^t v(\tau)\,\mathrm{d}\tau\,.$$

Definition 6 (Empirical controllability gramian) *Define the empirical controllability gramian by*

$$\mathcal{X} = \frac{1}{rs} \sum_{l=1}^{r} \sum_{m=1}^{s} \frac{1}{c_m^2} \sum_{i=1}^{p} \int_0^\infty \Phi^{ilm}(t)\,\mathrm{d}t,$$

where $\Phi^{ilm}(t) \in \mathbb{R}^{n\times n}$ is given by

$$\Phi^{ilm}(t) := \left(x^{ilm}(t) - \bar{x}^{ilm}\right)\left(x^{ilm}(t) - \bar{x}^{ilm}\right)^*,$$

and $x^{ilm}(t)$ is the state of the system (1) corresponding to the impulsive input $u(t) = c_m T_l e_i \delta(t)$.

Definition 7 (Empirical observability gramian) *Define the empirical observability gramian by*

$$\mathcal{Y} = \frac{1}{rs} \sum_{l=1}^{r} \sum_{m=1}^{s} \frac{1}{c_m^2} T_l \int_0^\infty \Psi^{lm}(t)\,\mathrm{d}t\ T_l^*,$$

where $\Psi^{lm}(t) \in \mathbb{R}^{n\times n}$ is given by

$$\Psi_{ij}^{lm}(t) := \left(y^{ilm}(t) - \bar{y}^{ilm}\right)^* \left(y^{jlm}(t) - \bar{y}^{jlm}\right),$$

and $y^{ilm}(t)$ is the output of the system (1) corresponding to the initial condition $x(0) = c_m T_l e_i$ with $u \equiv 0$.

Remark. Measuring the state-to-output behavior by setting initial values for the states is the same as injecting corresponding δ signals into the state equations (cf. [7]).

Notice, that both definitions contain the simple averaging part $\frac{1}{rs}\sum_{l=1}^{r}\sum_{m=1}^{s}$. Its purpose is to build a mean gramian from the individual gramians, calculated for some test matrix and test magnitude. Thus, one has to perform $r \cdot s \cdot p$ and $r \cdot s \cdot n$ simulations during the computation of the empirical controllability and observability gramian, respectively.

For linear systems, the empirical controllability and observability gramians are equal to the usual controllability and observability gramian, respectively (see [6]).

In practice, the integration for the computation of the empirical gramians is finite, but this is not a big problem, since too short integration intervals lead to less separated Hankel singular values, whereas the transformation matrix may still be good enough for model reduction.

The set of orthogonal matrices T_l should be chosen in a way that reflects the "natural operating region" of the system. For the controllability gramian, this means that the columns of those matrices reflect typical directions of input, whereas for the observability gramian, they should reflect typical state vectors. Similar suggestions apply to the set of constants c_m: They should render typical magnitudes of input signals and state values.

Definition 7 proposes to use zero input for computing the observability gramian (following the construction of the gramians for linear systems in [7]). Furthermore, the system excitations in both Definition 6 and Definition 7 are performed around zero. This may not be suitable for some systems, since the operating region defined by this setting may significantly differ from the natural operation region. This may even render the resulting transformation useless. Therefore, we recommend to apply system excitations around some "point of interest" (as proposed in [5]). This can be easily achieved by choosing a typical constant input u_{ss} (or more of them), calculating the corresponding steady state x_{ss}, and then performing the input and state excitations around those values. So, we propose to substitute

$$u(t) = c_m T_l e_i \delta(t) \quad \text{by} \quad u(t) = c_m T_l e_i \delta(t) + u_{\mathrm{ss}} \tag{5}$$

in Definition 6. The initial state values (upon which nothing is said in [6]) should be set to the corresponding steady state x_{ss} then. Accordingly, one should replace

$$\begin{gathered} x(0) = c_m T_l e_i \quad \text{with} \quad x(0) = c_m T_l e_i + x_{\mathrm{ss}} \\ \text{and} \\ u \equiv 0 \quad \text{with} \quad u \equiv u_{\mathrm{ss}} \end{gathered} \tag{6}$$

in Definition 7.

Another improvement may be to choose other input test functions when calculating the controllability gramian. Instead of using the Dirac impulse, one could apply functions like a step, sine, or whatever is more typical to the input. But one has to pay more attention to the resulting gramians then. The controllability gramian may have much more distinct singular values than the observability gramian and therefore the controllability is emphasized during the calculation of a balancing transformation. (Notice, however, that a pure scaling of a gramian has no effect on the calculated balancing transformation.)

Remark. The paper [5] is focused on applying a sequence of system excitations to compute the empirical gramians. Therefore, a sequence of steady state values (corresponding to the excitations) is used instead of mean values in both definitions of the empirical gramians. But instead of performing a sequence of excitations, one can also apply only one excitation at each simulation run and add the resulting gramians then. Both approaches seem to be appropriate, but the latter one is easier to state using the definitions from [6].

2.3. Calculating the balancing transformation

For reducing a system, we now want to have a transformation that does two jobs: First, it should "balance" a system which means that all state vectors have equal magnitudes of controllability and observability. This is achieved by transformations producing equal controllability and observability gramians. Second, the gramians should be in diagonal from, making it easy to decide which states could be eliminated.

In fact, both steps are achieved in one transformation $z(t) = Tx(t)$ which can be computed as follows: Let X be the controllability gramian (X_t, X_∞, or $\mathcal{X}$) and Y the observability gramian (Y_t, Y_∞, or $\mathcal{Y}$). Then

1. compute the Cholesky factorization of the controllability gramian

$$X = LL^*,$$

   where L is a lower left triangular matrix,

2. compute the singular value decomposition of L^*YL

$$L^*YL = U\Sigma^2U^*,$$

   where U is an orthogonal matrix and Σ is a diagonal matrix having the positive values $\sigma_1 \geq \ldots \geq \sigma_n$ (called *Hankel singular values*) on its main diagonal,

3. and set the transformation matrix to

$$T := \Sigma^{\frac{1}{2}}U^*L^{-1}.$$

The transformation T is then used to obtain system (2).

In the linear case, the resulting system realization

$$\widehat{A} = TAT^{-1},\ \widehat{B} = TB,\ \widehat{C} = CT^{-1}$$

is called "principal axis balanced" and its controllability and observability gramian both equal Σ:

$$\begin{aligned}\widehat{X} &= \int_0^\infty \mathrm{e}^{\widehat{A}t}\widehat{B}\widehat{B}^*\mathrm{e}^{\widehat{A}^*t}\,\mathrm{d}t = TXT^* \\ &= \left(\Sigma^{\frac{1}{2}}U^*L^{-1}\right)(LL^*)\left(L^{-*}U\Sigma^{\frac{1}{2}}\right) = \Sigma\end{aligned}$$

$$\begin{aligned}\widehat{Y} &= \int_0^\infty \mathrm{e}^{\widehat{A}^*t}\widehat{C}^*\widehat{C}\mathrm{e}^{\widehat{A}t}\,\mathrm{d}t = T^{-*}YT^{-1} \\ &= \left(\Sigma^{-\frac{1}{2}}U^{-1}L^*\right)Y\left(LU^{-*}\Sigma^{-\frac{1}{2}}\right) \\ &= \Sigma^{-\frac{1}{2}}U^{-1}U\Sigma^2U^*U^{-*}\Sigma^{-\frac{1}{2}} = \Sigma\end{aligned}$$

Since Σ consists only of the sorted Hankel singular values on the main diagonal, this allows an easy decision which states to delete from the system because of low influence on the I/O behavior: Those are the states z_l (i.e. the *rows* of T) corresponding to (relatively) "small" entries σ_l in the new gramians, because they need a high (squared) input energy of σ_l^{-1} to be

reached and also produce a low output energy of σ_l. One should observe that it's important to perform the truncation between two distinct eigenvalues of Σ. Otherwise, stability of the reduced system is not guaranteed.

In the nonlinear case, the transformation and truncation are performed in the same way, but it has to be observed that, generally,

- the Hankel singular values are not invariants under state space transformations,
- the empirical gramians of the transformed system are not equal or diagonal, and
- there is no proposition about the stability of the truncated system (even when truncating between distinct singular values).

Remark. There exist some other algorithms to calculate the balancing transformation but they all produce the same result and the computation always requires positive definite gramians (e.g., for computing the Cholesky factorization). For linear systems, this means that (A, B, C) has to be controllable and observable. If this is not the case, the uncontrollable as well as the unobservable states have to be removed in a first step. This can be achieved by calculating the so-called Kalman decomposition where the new states are sorted into 4 groups:

- controllable and observable states
- controllable and unobservable states
- uncontrollable and observable states
- uncontrollable and unobservable states

In this form, states that are either uncontrollable or unobservable can be easily removed to obtain a minimal realization. For nonlinear systems, however, this is not as clear. But it must also be assured that the empirical gramians are positive definite to be able to calculate a "balancing" transformation.

2.4. Model reduction

There are two main possibilities to eliminate the now identified unimportant states: Either

- truncate the system by simple deletion or
- use the method of singular perturbation to calculate the values of the unimportant states.

The first method exactly matches the original system at $\omega = \infty$ and gives a better approximation for high frequencies whereas the second method matches the original system at $\omega = 0$ (which means matching DC gains) and gives a better approximation for low frequencies (cf. [8, chapter 1]).

However, the n^{th} order state space will now be divided into r "important" and $n - r$ "unimportant" states by the two Galerkin projections P and Q,

$$P = \begin{pmatrix} I_r & O_{r,n-r} \end{pmatrix} \in \mathbb{R}^{r \times n}$$
$$Q = \begin{pmatrix} O_{n-r,r} & I_{n-r} \end{pmatrix} \in \mathbb{R}^{(n-r) \times n}$$

(where I_n is the $n \times n$ identity matrix and O_{n_1,n_2} is the $n_1 \times n_2$ zero matrix) so that we get the following partitioning of the (transformed) state vector

$$z(t) = \begin{pmatrix} \widetilde{z}(t) \\ z_e(t) \end{pmatrix}$$

where $\widetilde{z}(t) := Pz(t) \in \mathbb{R}^r$ are the states to be kept for the reduced order model and $z_e(t) := Qz(t) \in \mathbb{R}^{n-r}$ are the states to be eliminated. Therefore we can rewrite the transformed system (2) in the following form:

$$\dot{\widetilde{z}}(t) = P\widehat{f}\left(\begin{pmatrix}\widetilde{z}(t)\\ z_e(t)\end{pmatrix}, u(t)\right) \tag{7a}$$

$$\dot{z}_e(t) = Q\widehat{f}\left(\begin{pmatrix}\widetilde{z}(t)\\ z_e(t)\end{pmatrix}, u(t)\right) \tag{7b}$$

$$y(t) = \widehat{g}\left(\begin{pmatrix}\widetilde{z}(t)\\ z_e(t)\end{pmatrix}\right). \tag{7c}$$

2.4.1. Reduction by truncation

One can reduce a system by simply cutting off all unimportant states:

$$\begin{aligned} & z_e(t) \stackrel{!}{=} 0 \\ \Rightarrow \ & \begin{pmatrix}\widetilde{z}(t)\\ z_e(t)\end{pmatrix} = \begin{pmatrix}\widetilde{z}(t)\\ 0\end{pmatrix} = P^*\widetilde{z}(t) \end{aligned}$$

Substituting that into (7a) and (7c), we obtain

$$\begin{aligned} \dot{\widetilde{z}}(t) = \widetilde{f}\big(\widetilde{z}(t), u(t)\big) &:= P\widehat{f}\big(P^*\widetilde{z}(t), u(t)\big) &&= PTf\big(T^{-1}P^*\widetilde{z}(t), u(t)\big) \\ y(t) = \widetilde{g}\big(\widetilde{z}(t)\big) &:= \widehat{g}\big(P^*\widetilde{z}(t)\big) &&= g\big(T^{-1}P^*\widetilde{z}(t)\big), \end{aligned}$$

and for linear systems $(A, B, C, 0)$ this leads to

$$\widetilde{A} = PTAT^{-1}P^*,\ \widetilde{B} = PTB,\ \widetilde{C} = CT^{-1}P^*.$$

2.4.2. Reduction by singular perturbation

The other method is based on singular perturbation: One can assume that the unimportant states are much faster than the others and set $\dot{z}_e(t) = 0$ in equation (7b):

$$Q\widehat{f}\left(\begin{pmatrix}\widetilde{z}(t)\\ z_e(t)\end{pmatrix}, u(t)\right) = QTf\left(T^{-1}\begin{pmatrix}\widetilde{z}(t)\\ z_e(t)\end{pmatrix}, u(t)\right) = 0 \tag{8}$$

Assuming that (8) is resolvable for $z_e(t)$, one obtains the quasi steady state values of $z_e(t)$ as

$$z_e(t) = \psi\big(\widetilde{z}(t), u(t)\big) \tag{9}$$

with some function ψ, depending on $\widetilde{z}(t)$. (For linear systems, ψ exists if and only if $Q\widehat{A}Q^*$ is invertible.) This can be substituted into (7a) and (7c) to obtain the reduced order system

$$\begin{aligned} \dot{\widetilde{z}}(t) = \widetilde{f}\big(\widetilde{z}(t), u(t)\big) &:= P\widehat{f}\left(\begin{pmatrix}\widetilde{z}(t)\\ \psi\big(\widetilde{z}(t)\big)\end{pmatrix}, u(t)\right) &&= PTf\left(T^{-1}\begin{pmatrix}\widetilde{z}(t)\\ \psi\big(\widetilde{z}(t)\big)\end{pmatrix}, u(t)\right) \\ y(t) = \widetilde{g}\big(\widetilde{z}(t)\big) &:= \widehat{g}\left(\begin{pmatrix}\widetilde{z}(t)\\ \psi\big(\widetilde{z}(t)\big)\end{pmatrix}\right) &&= g\left(T^{-1}\begin{pmatrix}\widetilde{z}(t)\\ \psi\big(\widetilde{z}(t)\big)\end{pmatrix}\right). \end{aligned}$$

2.4.3. Comparison

In general, the method of truncation is considered to be superior because it always leads to a low order ODE system which can be solved faster than the original system. It also better approximates the system for high input frequencies.

On the other hand, singular perturbation better approximates "slow systems" but has the disadvantage that it is only practicable in special cases like linear systems, e.g., where it is possible to find an analytic solution (9) to (8). Generally, this will be impossible and $z_e(t)$ has to be calculated numerically for every time step. Then, the reduction of a nonlinear system using singular perturbation leads to the following differential algebraic equation system which is more complex to solve than the full order ODE system:

$$\begin{pmatrix} \dot{\widetilde{z}}(t) \\ 0 \end{pmatrix} = \widehat{f}\left(\begin{pmatrix} \widetilde{z}(t) \\ z_e(t) \end{pmatrix}, u(t) \right)$$

3. Problems and solutions specific to the application to modular systems

There may be a pitfall when trying to reduce a modular system: The straightforward way of reducing such a system would be to split it into its modules (defining the modules' inputs and outputs in a way that they can be "plugged" together to the whole system again) and to apply balanced reduction to each module separately in order to avoid a destruction of the modular structure. This may emphasize the importance of states which are only relevant for the input-output behavior of the module but unimportant for the whole system.

This problem is illustrated by a small linear example which is rather artificial but serves well for demonstration purposes. It does not look modular, because it is only of order 4, but sth. like that could also appear in a bigger context.

Remark. The following linear systems are given in the form

$$\left(\begin{array}{c} \dot{x} \\ \hline y \end{array}\right) = \left(\begin{array}{c|c} A & B \\ \hline C & \end{array}\right) \left(\begin{array}{c} x \\ \hline u \end{array}\right),$$

which is equivalent to

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

Entries of matrices which are zero per definition are not printed. This should make it easier to understand the connection structure of the modules (which are separated by dashed lines).

The value "0" indicates an entry which was calculated to be exactly zero, and "0.000" means that the entry is greater than zero but rounded to 4 digits after the decimal point.

Time dependency should be clear from the context and is omitted from now on.

The system

$$\left(\begin{array}{c} \dot{x}_1 \\ \dot{x}_2 \\ \hdashline \dot{x}_3 \\ \dot{x}_4 \\ \hline y \end{array}\right) = \left(\begin{array}{cc:cc|c} -1 & 0.1 & & & 1 \\ 0.1 & -1 & & 10 & \\ \hdashline 1 & & -1 & & \\ & -0.1 & & -1 & \\ \hline & & 1 & & \end{array}\right) \left(\begin{array}{c} x_1 \\ x_2 \\ \hdashline x_3 \\ x_4 \\ \hline u \end{array}\right)$$

can be divided into the two modules

$$\left(\begin{array}{c} \dot{x}_1 \\ \dot{x}_2 \\ \hline c_1 \\ c_2 \end{array}\right) = \left(\begin{array}{cc|c} -1 & 0.1 & 1 \\ 0.1 & -1 & 1 \\ \hline 1 & & \\ & -0.1 & \end{array}\right) \left(\begin{array}{c} x_1 \\ x_2 \\ \hline u \\ c_3 \end{array}\right)$$

and

$$\left(\begin{array}{c} \dot{x}_3 \\ \dot{x}_4 \\ \hline y \\ c_3 \end{array}\right) = \left(\begin{array}{cc|c} -1 & & 1 \\ & -1 & 1 \\ \hline 1 & & \\ & 10 & \end{array}\right) \left(\begin{array}{c} x_3 \\ x_4 \\ \hline c_1 \\ c_2 \end{array}\right)$$

with the connections c_1, c_2, and c_3.

Obviously, the two most important states for the input-output behavior of the whole system are x_1 (which receives the input) and x_3 (which is the output). Those two states are directly connected by the entry in position (3, 1) and the other state couplings are either very small (entries "± 0.1") or they are not directly related to the "important" states (entry "10"). This also results by balancing the whole system, where one gets the Hankel and transformation matrices

$$\Sigma = \begin{pmatrix} 0.6068 & & & \\ & 0.1042 & & \\ & & 0.0001 & \\ & & & 0.0000 \end{pmatrix} \text{ and } T = \begin{pmatrix} -0.5952 & -0.0280 & -0.8381 & -0.1751 \\ 0.5949 & 0.0482 & -0.8417 & 0.4453 \\ -0.0201 & 0.4904 & -0.0416 & 10.1576 \\ -0.0043 & -0.4920 & 0.0886 & 10.1659 \end{pmatrix},$$

respectively. Note that the first two Hankel singular values (being significantly bigger than the third and forth ones) correspond to rows of the transformation matrix which project the system mainly onto the states x_3, x_1, and x_4 (in decreasing order of importance).

This situation changes after cutting the system into two modules as shown above. Combining the "local" Hankel and transformation matrices of the two modules[2], one gets

$$\Sigma_{\text{loc}} = \left(\begin{array}{cc:cc} 0.5071 & & & \\ & 0.0556 & & \\ \hdashline & & 5.0000 & \\ & & & 0.5000 \end{array}\right) \text{ and } T_{\text{loc}} = \left(\begin{array}{cc:cc} -0.9954 & -0.0524 & & \\ -0.0505 & 0.3329 & & \\ \hdashline & & 0 & 3.1623 \\ & & 1.0000 & 0 \end{array}\right).$$

The problem is, that a truncation of the two modules to one state each would now completely reject the state x_3, because only the first and third row of T_{loc} would be preserved. This contradicts the results of the global input-output analysis, where x_3 was considered to be the most important state. The reason for such different results is, that the importance of x_4 is raised by the "connecting" output of c_3 in the second module. In fact, truncation cannot be used here at all, because it deletes the state x_3 which subsequently sets the system's output to zero. The only solution in this case would be to reduce the module by singular perturbation (see section 2.4.2) which would preserve the steady state value of x_3, but still poorly approximate the dynamic behavior.

A much better approximation (regardless of the method used to eliminate the unimportant states) can be achieved by using global I/O analysis and then projecting the obtained gramians

[2]The matrices of the modules are combined for easier comparison. Moreover, it is possible to proceed by transforming the whole system with the combined transformation matrix T_{loc} without mixing states of different modules and then suppress the unimportant states as indicated by Σ_{loc}.

onto the modules. The resulting matrices

$$\Sigma_{\text{glo}} = \left(\begin{array}{cc:cc} 0.3563 & & & \\ & 0.0002 & & \\ \hdashline & & 0.3546 & \\ & & & 0.0002 \end{array}\right) \text{ and } T_{\text{glo}} = \left(\begin{array}{cc:cc} -0.8407 & -0.0437 & & \\ -0.0255 & 0.6361 & & \\ \hdashline & & -1.1874 & -0.1184 \\ & & 0.0840 & 16.1596 \end{array}\right)$$

suggest a projection onto x_1 and x_3 mainly, when truncating each module to one state. They are obtained by the following procedure:

1. Calculate the controllability and observability gramian of the whole system, X and Y respectively, using standard formulas or empirical balancing. (This also has the advantage that one does not have to figure out which inputs and initial conditions are "natural" for each module. That has only to be done once for the whole system.)

2. Prevent state transformations that would mix states of different modules by projecting the gramians onto their "local parts" only to obtain

$$X_{\text{glo}} = \left(\begin{array}{c:c:c} P_1 X P_1^* & & \\ \hdashline & \ddots & \\ \hdashline & & P_m X P_m^* \end{array}\right) \quad \text{and} \quad Y_{\text{glo}} = \left(\begin{array}{c:c:c} P_1 Y P_1^* & & \\ \hdashline & \ddots & \\ \hdashline & & P_m Y P_m^* \end{array}\right),$$

   where P_i is the Galerkin projection onto the states of the i^{th} module.

3. Proceed by calculating the balancing transformation for each module separately[3].

This procedure may be justified with the argumentation, that only the local effects of a system excitation should be measured by the corresponding parts of the gramians. This is what happens if one carries out the multiplications $P_i \mathcal{X} P_i^*$ and $P_i \mathcal{Y} P_i^*$ in Definition 6 and Definition 7, respectively.

Another advantage of this method is, that it always produces the same results, no matter *how* the connections of the split system are built. In contrast, it makes a big difference for "totally local" balancing, *where* the factor 10 is placed in our example. If it would be placed in the input matrix of the first module instead of placing it in the output matrix of the second one, the results would not be the same. The proposed "global" I/O analysis, however, is invariant to this choice.

4. Comparison of different reduction methods with an example system

In general, it is very difficult to rate the quality of approximations for a nonlinear system because tools like Bode diagrams are limited to linear systems. Therefore, we created an ODE system that has a typical structure for our applications. Systems like the one presented below could be models of metabolic processes where the states are concentrations of metabolites, e.g.

[3]This corresponds to calculating the balancing transformation for the whole system using the projected gramians X_{glo} and Y_{glo} with one exception: The SVD has to be done separately for each module.

4.1. System description

We will consider a positive ODE system which is based on flows of the type

$$\varphi_\alpha(x):\ (-1,\infty)\to(-\infty,1),\ x\mapsto\frac{x^\alpha}{1+x^\alpha}$$

with a parameter $\alpha\in\{1,3,5,\ldots\}$. This flow function is increasing for all $x\in(-1,\infty)$ and $\varphi_\alpha(0)=0$ holds true. Therefore, it is positive if and only if the concentration is positive so that it can be used to construct a positive system, and we will consider φ_α for the nonnegative axis $\mathbb{R}_+:=[0,\infty)$ only. For big values of α, φ_α has a switching point at $\varphi_\alpha(1)=\frac{1}{2}$. This shape gets sharper for increasing α and leads to an approximation of the step function:

$$\lim_{\alpha\to\infty}\varphi_\alpha(x)=\sigma(x-1)$$

Remark. Usually, the switching point of such flows is not always at $x=1$, but we prevented introducing another parameter here, because there are already enough parameters to obtain a system that does not have very special properties and is complicated enough for testing empirical reduction methods.

Fixing the parameter α and multiplying the function with a nonnegative constant leads to the definition of the flow matrix

$$F:\ \mathbb{R}_+^n\to\mathbb{R}_+^{n\times n},\ F_{ij}(x):=\mu_{ij}\cdot\varphi_\alpha(x_j),$$

where $F_{ij}(x)$ is the flow from x_j to x_i, *depending only on* x_j. $\mu\in\mathbb{R}_+^{n\times n}$ is a parameter matrix holding all the inflow coefficients, but since flows from an element to itself do not make any sense, its diagonal equals zero:

$$\mu_{ii}=0\ \Rightarrow\ F_{ii}\equiv 0,\quad 1\le i\le n$$

Remark. Note that it's not necessary to take this precaution, because a flow from an element to itself always vanishes. Therefore we will not use this assumption in the following.

Using the flow matrix, we can now build the compartmental n^{th}-order system

$$\begin{aligned}
\dot{x}_1&=\sum\nolimits_{k=1}^n\big(-F_{k1}(x)+F_{1k}(x)\big)-\lambda x_1+u && :=f_1(x,u),\\
\dot{x}_i&=\sum\nolimits_{k=1}^n\big(-F_{ki}(x)+F_{ik}(x)\big)-\lambda x_i && :=f_i(x,u),\quad \text{for } 2\le i\le n-1,\\
\dot{x}_n&=\sum\nolimits_{k=1}^n\big(-F_{kn}(x)+F_{nk}(x)\big)-\lambda x_n-\varphi_\alpha(x_n) && :=f_n(x,u),\\
y&=\varphi_\alpha(x_n),
\end{aligned}$$

where the input u is an inflow to x_1 and the output $y=\varphi_\alpha(x_n)$ is an outflow of x_n. The mortality parameter $\lambda\in\mathbb{R}_+$ is introduced in addition to the parameters $\mu\in\mathbb{R}_+^{n\times n}$ and $\alpha\in\{1,3,5,\ldots\}$.

Remark. It may be disturbing at first sight, that the flow matrix entry F_{ij} determines the flow from x_j to x_i. The reason for "swapping" the indices is, that F then reflects the form of the Jacobian of the

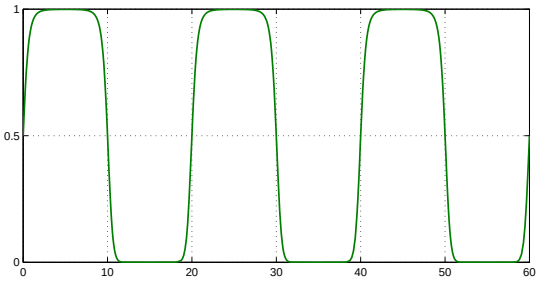


Figure 1: The input function $u(t) = \varphi_\alpha\left(\sin\left(\frac{\pi}{10}t\right) + 1\right)$ for $\alpha = 10$.

right hand side function $f(x, u)$:

$$\frac{\partial f(x,u)}{\partial x} = \begin{pmatrix} -\sum\limits_{k\neq 1} F'_{k1}(x) - \lambda & F'_{12}(x) & \cdots & F'_{1n}(x) \\ F'_{21}(x) & -\sum\limits_{k\neq 2} F'_{k2}(x) - \lambda & \cdots & F'_{2n}(x) \\ \vdots & \vdots & \ddots & \vdots \\ F'_{n1}(x) & F'_{n2}(x) & \cdots & -\sum\limits_{k\neq n} F'_{kn}(x) - \lambda - \varphi'_\alpha(x_n) \end{pmatrix},$$

with $F'_{ij}(x) := \frac{\partial F_{ij}(x)}{\partial x_j}$.

We fix the parameter values $\alpha = 10$, $\lambda = 0.03$,

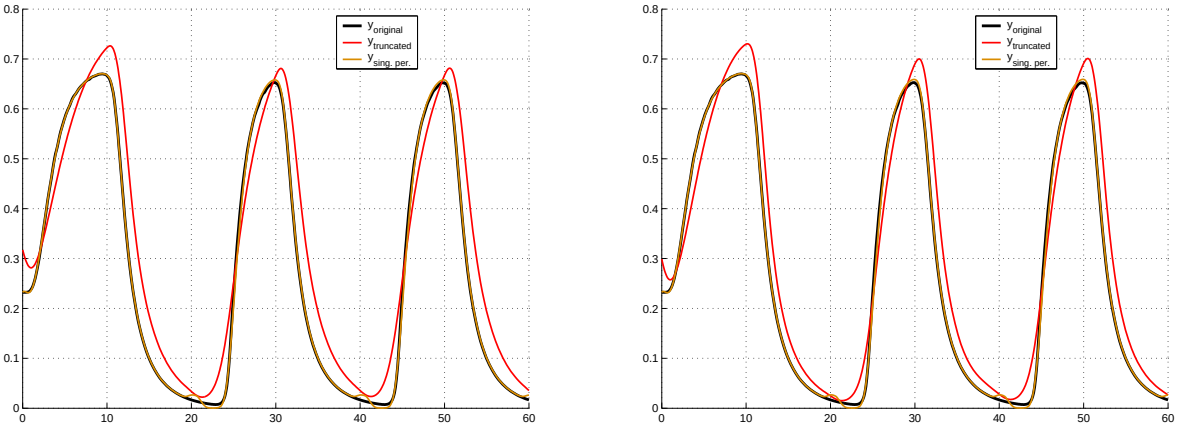
$$\mu = \left(\begin{array}{ccccc:ccccc} & & & & 1.23 & & & & & \\ 1.05 & & 1.13 & & & & & & & \\ 0.76 & & & 0.75 & & & & & & \\ 1.10 & 1.25 & & & & & & & & \\ & 1.24 & 0.92 & 1.01 & & & & & & \\ \hdashline & & & & 1 & & & & & 1.19 \\ & & & & & 0.84 & & 1.16 & & \\ & & & & & 1.01 & & & 1.20 & \\ & & & & & 0.97 & 0.82 & & & \\ & & & & & & 0.93 & 1.29 & 0.77 & \end{array}\right), \tag{10}$$

and use the input function $u(t) = \varphi_\alpha\left(\sin\left(\frac{\pi}{10}t\right) + 1\right)$ for all simulations (see Figure 1). According to (10), we defined a 10^{th} order system that consists of two modules with 5 states each. The states are strongly coupled within the two modules but the modules are connected to each other through $\mu_{6,5}$ only. This means that there is only one flow from module 1 to module 2 and no flow in the reverse direction.

4.2. Importance of the model reduction parameters

For the time being, we try to reduce the entire system without considering its modular structure.

If one uses Definition 6 and Definition 7 without the adjustments (5) and (6), respectively, the resulting gramians for this system are not usable at all: The controllability gramian has non-zero entries for the first 4 states only, because the rest of the system does not profit from



(a) $\mathcal{T}_\mathcal{X} = \{I_1\}$, $\mathcal{T}_\mathcal{Y} = \{I_{10}\}$ **(b)** $\mathcal{T}_\mathcal{X} = \{I_1, -I_1\}$, $\mathcal{T}_\mathcal{Y} = \{I_{10}, -I_{10}\}$

Figure 2: Original and reduced (order 2) systems. The input test function used for calculating the empirical controllability gramian was $\delta(t)$, $\mathcal{C}_\mathcal{X} = \{0.5\}$, and $\mathcal{C}_\mathcal{Y} = \{0.5\}$.

the input (due to the rather steep flow functions). The observability gramian even equals zero in all entries. Therefore, we define our "point of interest" for

$$u_{\text{ss}} = 0.5 \tag{11}$$

which results in the steady state

$$x_{\text{ss}} \approx \begin{pmatrix} 0.93 & 0.88 & 0.89 & 0.94 & 0.95 & 0.88 & 0.87 & 0.85 & 0.86 & 0.89 \end{pmatrix}^T. \tag{12}$$

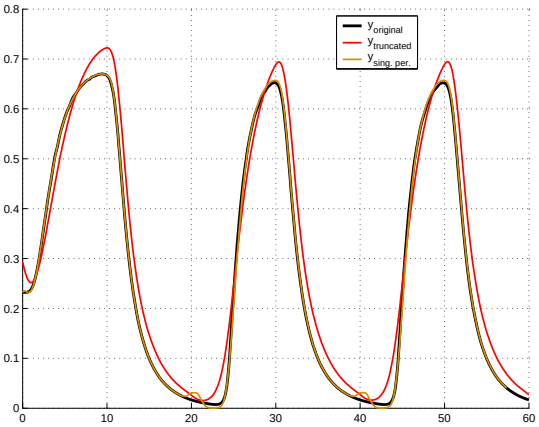
As a first try, we reduce the system to order 2 by truncation and singular perturbation, using the approximate Dirac impulse δ as input test signal, $\mathcal{T}_\mathcal{X} = \{I_1\}$, $\mathcal{T}_\mathcal{Y} = \{I_{10}\}$, $\mathcal{C}_\mathcal{X} = \{0.5\}$, and $\mathcal{C}_\mathcal{Y} = \{0.5\}$. The simulation results for the original system and the two reduced systems are plotted in Figure 2(a).

Obviously, the system which was reduced using singular perturbation approximates the original system much better. Unfortunately, solving this DAE system takes much more time than solving the original 10^{th} order system. But, to show the power of this method, we will continue to plot its results in some of the following attempts.

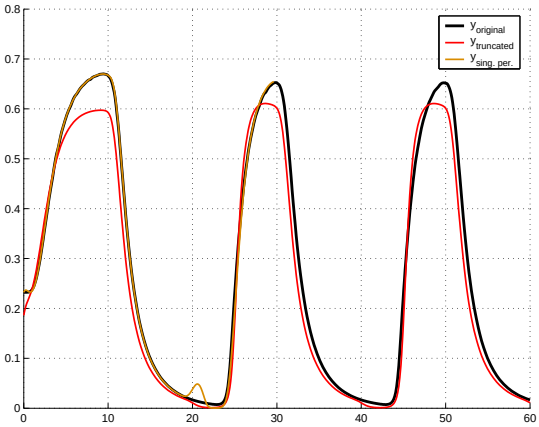
Figure 2(b) shows a slight improvement of the truncated approximation for small output values and the steep parts of the simulation. This was achieved using different test matrices which should exploit the operating region into both the positive and the negative direction. On the other hand, the approximation gets a little bit worse at the peak values of y.

Another parameter which was suggested to be adjusted is the input test function used during computation of the empirical controllability gramian. In order to use a more "natural" input test signal, we set this function to a constant excitation of 0.5 which lasts for the first 2 seconds and to an ongoing constant excitation of 0.5 (see Figures 3(a) and 3(b), respectively). Note, that the simulation of the DAE systems could not be completed, because the solver was not able to calculate solutions within the given error bounds.[4] But the resulting truncated models are now even better approximations (at least for the steep parts and small values of the output in Figure 3(b)). It should also be observed, that the empirical controllability gramian

[4]Therefore, we will not continue to plot systems reduced by singular perturbation.



(a) The input test function was a constant excitation of magnitude 0.5 during the first 2 seconds.



(b) The input test function was a constant excitation of magnitude 0.5.

Figure 3: Original and reduced (order 2) systems. The parameters were $\mathcal{T}_\mathcal{X} = \{I_1, -I_1\}$, $\mathcal{T}_\mathcal{Y} = \{I_{10}, -I_{10}\}$, $\mathcal{C}_\mathcal{X} = \{1\}$, and $\mathcal{C}_\mathcal{Y} = \{0.5\}$.

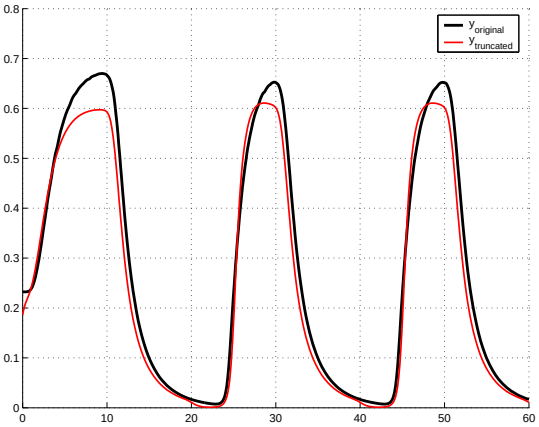
corresponding to Figure 3(b) has much bigger entries and also much more varying singular values than before (which is not a problem in this case).

4.3. Comparison with "linear" gramians

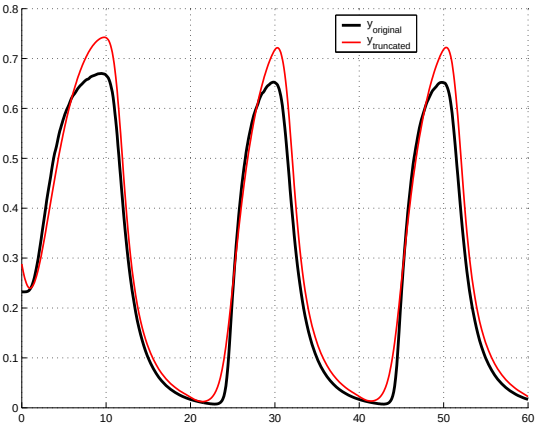
Figure 4 is used to compare our result (which is repeated in Figure 4(a)) with another possibility to obtain the gramians around some point of interest: Linearize the system around the steady state defined by (11) and (12), and calculate the usual gramians for the linear system. Then proceed as before by calculating a balancing transformation and reducing the nonlinear system using this transformation. In Figure 4(b), we used the gramians of the linearized system only. Comparing Figures 4(a) and 4(b), one wishes to obtain some kind of mean solution which incorporates the best features of both. Therefore, we built a mean gramian in Figure 4(c) and tried to further improve the result by weighting the two versions of the gramians 4:1 in Figure 4(d).

Building such an average from several calculated gramians seems to be a reasonable method, since it is also done inside the definitions of the empirical gramians. In fact, one might wonder if the (time consuming) calculation of the empirical gramians is really necessary, or if it would be sufficient to linearize a system around several points of interest and build a mean gramian then. This method may be practicable for improving the empirical gramians. But in general, it will be very difficult to find a good "mixture" of linear gramians and using the empirical gramians is a better starting point.

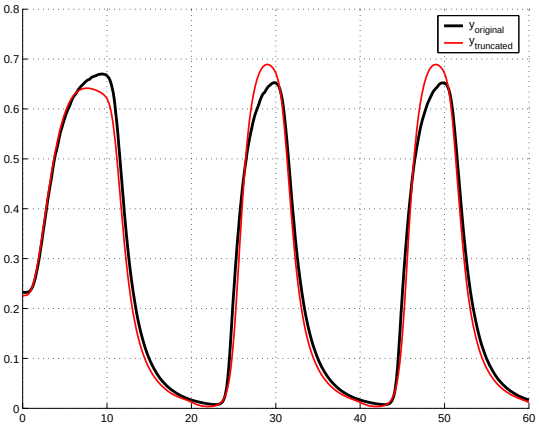
Comparing Figures 4(a) and 4(b) it even seems as if calculating the gramians for one linearization would be enough. That may be true for $\alpha = 10$, but for more nonlinear systems, the empirical gramians clearly outperform the linear ones (see Figure 5, where α was increased to 100 and the order of the reduced systems was increased to 3).



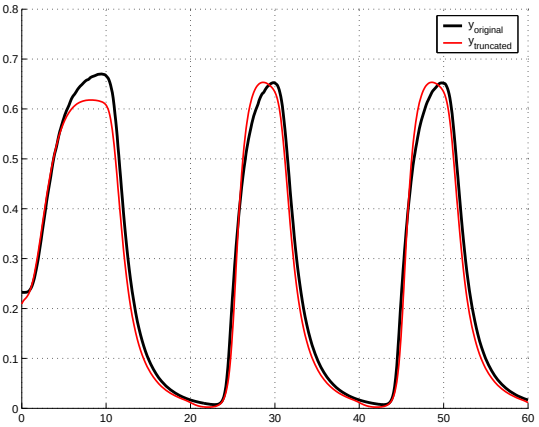
(a) Using the empirical gramians only.



(b) Using the gramians of the linearized system only.

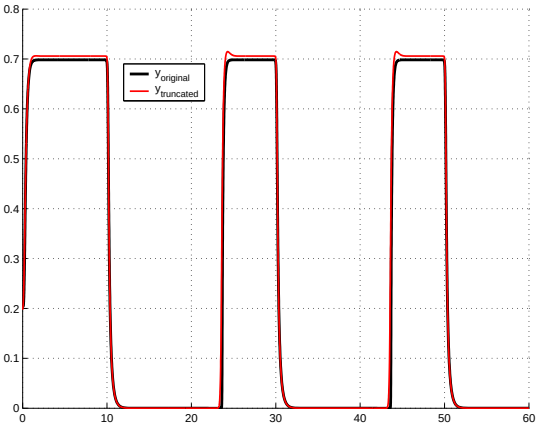


(c) Building the mean of the empirical and "usual" gramians.

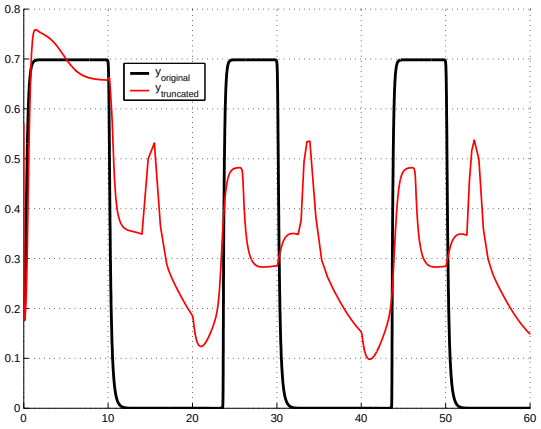


(d) Using a weighted average (4:1) of the empirical and "usual" gramians.

Figure 4: Original and truncated (order 2) systems. The parameters for obtaining the empirical gramians were the same as in Figure 3(b).



(a) Good approximation using empirical gramians. The parameters for obtaining the empirical gramians were the same as in Figure 3(b).



(b) Unacceptable reduced model using gramians of the linearized system.

Figure 5: Original and truncated (order 3) systems for $\alpha = 100$.

4.4. Comparison of methods for modular reduction

After having examined how to reduce the entire system, we now want to preserve the example's modular structure and reduce each module separately. This will mean a loss of degrees of freedom, as we are no longer allowed to transform the full 10-dimensional state space, but are restricted to the two 5-dimensional state spaces. Therefore, we expect the approximations to be worse. We now set $\alpha = 50$ in order to get distinguishable results for the following three reduction methods:

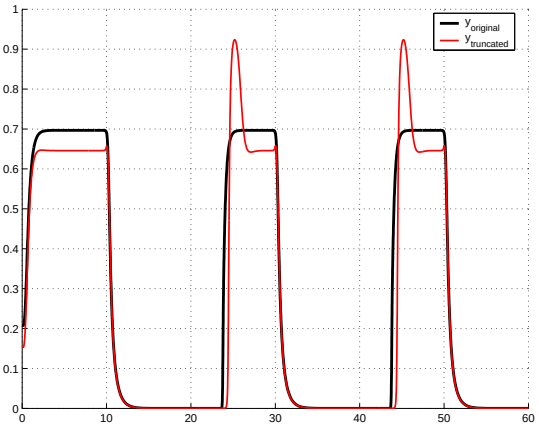
- Calculate the empirical gramians globally (for the full system) and perform a global transformation of the full state space (see Figure 6(a)).
- Calculate the empirical gramians locally (for each module separately) and perform a local transformation of the two modules' state spaces (see Figure 6(b)).
- Calculate the empirical gramians globally (for the full system) but perform a local transformation of the two modules' state spaces (see Figure 6(c)).

Surprisingly, the first method produces a rather bad result, whereas the two methods with local transformations perform very well. Among them, the mixed method which uses global gramians gives a better approximation. It is difficult to find an explanation for these results, but one could imagine that the quality of the projected gramians is better than the quality of the other two versions of gramians. However, the reason for this is quite unclear.

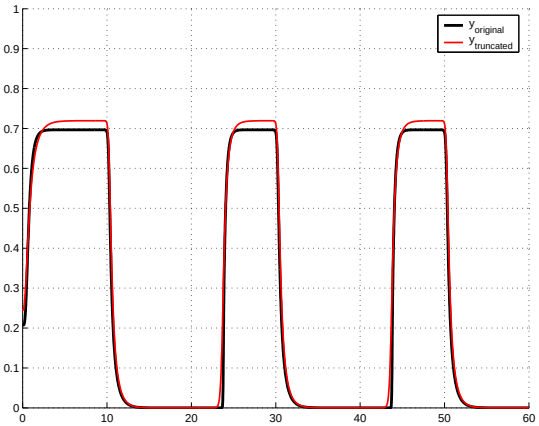
Remark. Using the mixed method, the full vector of Hankel singular values gives a hint of how important a module is for the whole system: If the module's Hankel singular values are small compared to the others, many of the modules states may be reduced.

5. Conclusion

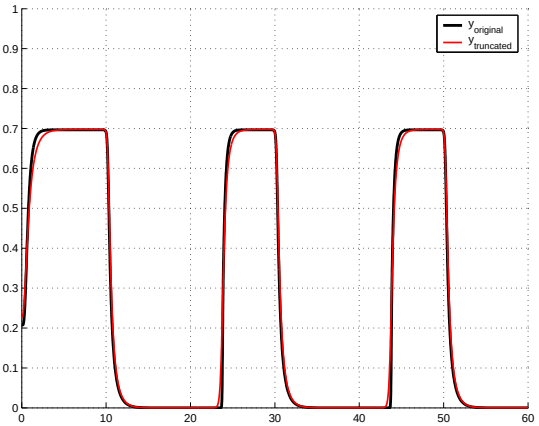
The most important parameters when calculating the empirical gramians seem to be a reasonable point of interest, as well as good magnitude coefficients $\mathcal{C}_\mathcal{X}$ and $\mathcal{C}_\mathcal{Y}$ and test matrices $\mathcal{T}_\mathcal{X}$



(a) Global gramians and global transformation.



(b) Local gramians and local transformation.



(c) Global gramians but local transformation.

Figure 6: Comparison of different global/local combinations for $\alpha = 50$. The full system was always truncated to 2 states, where modules were truncated to 1 state each when a local transformation was used. The parameters for obtaining the empirical gramians were the same as in Figure 3(b) (for the full system as well as for the two modules).

and $\mathcal{T}_\mathcal{Y}$, in order to exploit the system's operating region.

A change of the input test signal might also improve the results, but often the Dirac impulse δ will already do its job.

One can try to improve the approximation in some regions of the state space by linearizing the system in this region and adding the linear gramians to the empirical ones. In doing so, a weighting factor has to be found because the singular values of the empirical gramians might have bigger order of magnitudes than the linear ones.

The empirical gramians are much better suited for highly nonlinear systems than the gramians of a linearization.

Modular reduction may even work better than reducing the full system for systems which really have a modular structure. To improve the reduction results and to circumvent having to choose input test signals for every inter-modular connection, one should consider to compute the empirical gramians for the full system and project them onto the modules as described in section 3.

To summarize: Model reduction for nonlinear and modular systems using empirical gramians is much more experimental than the reduction of linear systems, because the quality of the approximation cannot be determined without simulations and many parameters have to be considered and adjusted.

A. Description of the `Matlab` functions

This appendix contains a documentation of the `Matlab` functions which were used to obtain the results of section 4. There exists a short (technical) help text for every function and script file which is available via `Matlab`'s `help` command. The code is also well documented so that having a look with `Matlab`'s `edit` command should explain the functionalities. More abstract descriptions are provided below.

A.1. Core functions

`[sigma, T] = calcbal(X, Y)`

calculates the Hankel singular values `sigma` and a balancing transformation `T` from given gramians. The algorithm is based on a Cholesky factorization and a singular value decomposition as described in section 2.3.

The gramians `X` and `Y` must be positive definite, which means that the system realization has to be minimal.

`[sigma, T] = calclocbal(X, Y, m)`

calculates the Hankel singular values and a balancing transformation from given gramians. The calculated transformation has a block-diagonal shape so that a system's modular structure is preserved during transformation. The modules' sizes must be passed as a vector to parameter `m`.

`X` and `Y` are projected onto the modules and the balancing transformation is then calculated (using `calcbal`) for each module separately. After that, the solutions are concatenated to obtain the full vector of Hankel singular values `sigma` and the whole balancing transformation `T`.

It has to be observed, that the Hankel singular values are only sorted within each module, so that one has to reduce the system in a modular way, too.

`[X, t, x] = empctrb(rhsfct, testfct, c, T, tsim, xss, uss)`
computes the empirical controllability gramian. In addition to the gramian `X`, the function also returns the simulation data, that was generated to compute the gramian. If one is not interested in this additional data, only the gramian may be requested by entering `X = empctrb(...)` on the command line.

The function has to be called once for every test matrix `T` or test magnitude `c` and a mean gramian has to be build according to Definition 6.

`[Y, t, x, y] = empobsv(rhsfct, outfct, c, T, tsim, xss, uss)`
computes the empirical observability gramian. In addition to the gramian `Y`, the function also returns the simulation data, that was generated to compute the gramian. But this function can also be called by entering `Y = empobsv(...)`.

The function has to be called once for every test matrix `T` or test magnitude `c` and a mean gramian has to be build according to Definition 7.

A.2. Functions for the example system

The example system of section 4 consists of three functions containing the right hand side of the differential equations, the nonlinear flow function φ_α, and the output function, respectively, as well as the data file `exparams.mat`. This data file contains the parameters `ALPHA` $\in \mathbb{R}$ (nonlinearity), `MU1` $\in \mathbb{R}^{5\times5}$, `MU2` $\in \mathbb{R}^{5\times5}$, `MUfull` $\in \mathbb{R}^{10\times10}$ (flow coefficients for module 1, module 2, and the full system, respectively), and `LAMBDA` $\in \mathbb{R}$ (mortality).

`y = exout(x)`
Nonlinear output for the example system. This returns $\varphi_\alpha(x_n)$.

`result = exphi(x, alpha)`
Nonlinear flow function φ_α. The parameter `alpha` is passed to this function instead of using the global variable `ALPHA`, because it should be easily possible to use this function without the requirement of having defined a global variable.

`dxdt = exrhs(t, x, u)`
Nonlinear RHS for the example system which consists of input, flows, mortality, and output. Be aware, that this function has the parameter `u` in addition to the parameters `t` and `x`, which are required by the `Matlab` solvers in this order. Since $u(t)$ may be a function not only depending on time, but also on an arbitrary number of additional parameters (what actually happens when calculating the empirical controllability gramian), it would be difficult to pass those parameters to the right hand side if the right hand side also needs additional parameters. To handle this problem in a consistent and general way, one can use the wrapper function `applyinputfct` which is explained below. This saves the user from evaluating the input function in his differential equations and the right hand side only takes a scalar input `u`.

Furthermore, the derivative φ'_α of the flow function and the Jacobian are provided for calculating a linearized system:

`result = exdphi(x, alpha)`
Derivative $\varphi'_\alpha(x)$ of the nonlinear flow function.

`J = exjac(x)`
Jacobian $\frac{\partial f(x,u)}{\partial x}$ for the RHS of the example system.

A.3. Helper functions

`result = applyinputfct(t, x, rhsfct, inputfct, inputargs, varargin)`
is used as a wrapper for other functions which depend on a time-dependent input function and should be simulated using `Matlab`'s ODE or DAE solvers. Instead of passing the right hand side of a ODE or DAE system to those solvers (which would make it impossible to apply the input function), one can pass `applyinputfct` to the solvers and the right hand side and input function in turn are given as parameters to `applyinputfct`.

`delta = impulsefct(t)`
generates an approximate Dirac impulse. This function has the properties that its support is very small (determined by the local variable `base`) and the integral of it equals one. One has to observe, that the initial step size of ODE solvers has to be set smaller than the length of the support in oder to really integrate over the impulse. This can be achieved by the option `'InitialStep'` which was chosen to be 10^{-6} for a support of 10^{-3}.

`ss = steadystate(rhsfct, uss, ssguess, maxtime)`
calculates the steady state corresponding to the constant input `uss`. First, the system (specified by the right hand side `rhsfct`) is simulated using the initial values `ssguess`, and then an optimization is run to minimize the norm of the residual. The function tries to find the solution within `maxtime` seconds (approximately).

B. Example work flow

The `m`-files `workflow1`, `workflow2`, and `workflow3` are not functions but script files containing sequences of `Matlab` commands as they would be entered on the command line. They look rather big, but the greater part of them is used for defining parameters or displaying information to the user. The contents of those three files explain the necessary steps during model reduction and they could be copied and modified in order to work with different systems or parameters.

B.1. Calculation of the empirical gramians (`workflow1.m`)

This script file is used to calculate empirical gramians for the example systems. All parameters are set in the first section of the file and can simply be modified when treating other models.

In the file's second section, the gramians are calculated by calling `empctrb` and `empobsv` for all combinations of given test matrices and test coefficients.

B.2. Calculation of the Hankel singular values and the balancing transformation (`workflow2.m`)

This script file relies on already defined gramians and calculates the balancing transformation `T` and the Hankel singular values `sigma`. Furthermore, it sets the Galerkin projection matrix `P`. The user may choose global or local balancing by modifying the comments.

B.3. Simulation and plot of original and reduced systems for comparison (`workflow3.m`)

To simulate the original system as well as reduced systems, one may use this file, which has the least functionality but is the most complicated.

The top section again contains some parameters like the simulation time or the input function, e.g.

After that, the original system as well as two reduced (by truncation and singular perturbation) systems are simulated. In order to apply the input function during simulation, the helper function `applyinputfct` has to be used again. The reduced models are generated from the original system by applying the transformation and Galerkin projection which were calculated in `workflow2`.

The last part plots the input function, the simulated outputs, and the error functions including legends and titles.

This file is should be considered to act as a template and the user may want to comment out the parts concerning the system reduced by singular perturbation, because this simulation often cannot be completed.

References

[1] Athanasios C. Antoulas, *Approximation of linear dynamical systems*, Wiley Encyclopedia of Electrical and Electronics Engineering (John G. Webster, ed.), vol. 11, John Wiley and Sons, Inc., 1999, pp. 403–422.

[2] Athanasios C. Antoulas and Dan C. Sorensen, *Approximation of large-scale dynamical systems: An overview*, International Journal of Applied Mathematics and Computational Science **11** (2001), no. 5, 1093–1121.

[3] Martin Brokate, *Steuerungstheorie I*, lecture notes, Mathematisches Seminar, Universität Kiel, Germany, 1996.

[4] Geir E. Dullerud and Fernando G. Paganini, *A course in robust control theory: A convex approach*, Texts in Applied Mathematics, vol. 36, Springer Verlag, 2000.

[5] Juergen Hahn and Thomas F. Edgar, *A gramian based approach to nonlinearity quantification and model classification*, Industrial & Engineering Chemistry Research **40** (2001), no. 24, 5724–5731.

[6] Sanjay Lall, Jerrold E. Marsden, and Sonja Glavaški, *A subspace approach to balanced truncation for model reduction of nonlinear control systems*, International Journal on Robust and Nonlinear Control **12** (2002), no. 6, 519–535.

[7] Bruce C. Moore, *Principal component analysis in linear systems: Controllability, observability, and model reduction*, IEEE Transactions on Automatic Control **26** (1981), no. 1, 17–32.

[8] Goro Obinata and Brian D. O. Anderson, *Model reduction for control system design*, Communications and Control Engineering, Springer Verlag, 2001.

[9] Muruhan Rathinam and Linda R. Petzold, *An iterative method for simulation of large scale modular systems using reduced order models*, Proceedings of the IEEE Conference on Decision and Control, 2000.