Course notes on model reduction

for ACM 270, Winter 2021

Elizabeth Qian

February 16, 2021

These are my course notes for ACM 270 offered in Winter Quarter 2021 at Caltech. I'm particularly grateful to Boris Kramer for sharing his materials from a similar course offered at UC San Diego and to Serkan Gugercin for sharing his model reduction course materials as well.

General philosophy is to convey the main idea without going into too much painful detail. Fluency with linear algebra in Euclidean space is assumed. Students are expected to have seen numerical methods for solving PDEs and ODEs in a prior course, although this course does not rely heavily on those details.

1. Proper Orthogonal Decomposition

The attractiveness of the POD lies in the fact that it is a linear procedure. The mathematical theory behind it is the spectral theory of compact, self-adjoint operators. This robustness makes it a safe haven in the intimidating world of nonlinearity; although this may not do the physical violence of linearization methods, the linear nature of the POD is the source of its limitations. . . However, it should be made clear that the POD makes no assumptions about the linearity of the problem to which it is applied. In this respect it is as blind as Fourier analysis, and as general.
— Berkooz, Holmes, and Lumley 1993 [3]

1.1. Overview

We start our model reduction journey with POD because it is the most general and widely-used method. The use of POD to obtain reduced models requires no assumptions of the system to be reduced, although the performance of the POD reduced model is generally better when certain types of structure exist, as we will see. POD is a data-based method, and the underlying math is known to other communities by other names, including the Karhunen-Loève expansion, principal component analysis, and the Hotelling decomposition. The term *POD* has origins in the analysis of turbulent flows.

Today's focus is on the mechanics of POD, i.e., how the method works. We will start by deriving a reduced model for a high-dimensional system of linear ODEs. We will then formulate the problem for an infinite-dimensional linear PDE in order to show connections to the ODE case.

1.2. Semidiscrete formulation

1.2.1. Set-up

Our starting point will be the following high-dimensional system of ODEs:

$$\frac{\mathrm{d}\mathbf{s}}{\mathrm{d}t} = \mathbf{A}\mathbf{s}, \tag{1.1}$$

where $\mathbf{s} \in \mathbb{R}^N$ is called the *full state*, and $\mathbf{A} \in \mathbb{R}^{N\times N}$ is a linear matrix operator. Equation (1.1) is sometimes called the *full-order model* or *FOM* in model reduction literature. Such a system of high-dimensional ODEs can result from the spatial discretization of a PDE, but may also arise as the governing equations describing many interconnected objects (such as exciteable neurons in biology).

If N is large so that simulating eq. (1.1) for a time interval of interest, $[0, T]$, is expensive — especially if we wish to simulate the system for e.g. multiple initial conditions — then we would like to find a *reduced model* of dimension $r \ll N$ which can approximate eq. (1.1) more cheaply.

1.2.2. Snapshot data

The POD basis is based on data in the form of 'snapshots' of the solution. Let

$$\mathbf{S} = \begin{bmatrix} \mathbf{s}_1 & \mathbf{s}_2 & \cdots & \mathbf{s}_K \end{bmatrix} \in \mathbb{R}^{N\times K} \tag{1.2}$$

denote the *snapshot data matrix*, where each snapshot in the set $\{\mathbf{s}_l\}_{l=1}^K$ represents the solution to eq. (1.1) for some initial condition at some time.

In model reduction, snapshot data usually comes from a full simulation, i.e. code which solves eq. (1.1). Experimental snapshot data can be treated too but is less common and presents some challenges. The snapshot data can be provided at the outset (say you have a collaborator who uploads some data for you to download) or, if you have access to code that solves eq. (1.1) and outputs trajectory data, you can generate the snapshot data yourself. There are various strategies to generate snapshot data, including:

- randomly drawn from some distribution of initial conditions/parameters,
- using a greedy algorithm to select parameters (core part of reduced-basis methods later),
- doing a parameter sweep,
- simulating eq. (1.1) for $t \in [0, t_1]$ to get snapshot data and then using the resulting POD model to predict $t \in [t_1, t_2]$.

How the data are obtained will depend on the problem. For now we will just assume that somehow we have obtained K snapshots which we collect in the matrix in eq. (1.2).

1.2.3. The POD basis

The POD basis consists of the leading left singular vectors of the snapshot data matrix. That is, let

$$\mathbf{S} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^\top \tag{1.3}$$

denote the singular value decomposition (SVD) of $\mathbf{S}$, where $\mathbf{U} \in \mathbb{R}^{N\times N}$ and $\mathbf{V} \in \mathbb{R}^{K\times K}$ are orthogonal matrices and $\mathbf{\Sigma} \in \mathbb{R}^{N\times K}$ has diagonal entries $\sigma_1 \geq \sigma_2 \geq \sigma_3 \geq \cdots$. Let

$$\mathbf{U} = \begin{bmatrix} \mathbf{u}_1 & \mathbf{u}_2 & \cdots & \mathbf{u}_N \end{bmatrix}, \tag{1.4}$$

where $\mathbf{u}_i \in \mathbb{R}^N$, $i = 1, 2, \ldots, N$. The leading r left singular vectors, $\{\mathbf{u}_i\}_{i=1}^r$, define a *POD basis of rank* r. We denote by

$$\mathbf{U}_r = \begin{bmatrix} \mathbf{u}_1 & \mathbf{u}_2 & \cdots & \mathbf{u}_r \end{bmatrix} \in \mathbb{R}^{N\times r} \tag{1.5}$$

the POD basis matrix of size r. The span of the POD basis is called the *POD subspace.*

Optimality property You may recall from numerical linear algebra the rank-r truncated SVD of a matrix $\mathbf{S}$ is the best rank-r approximation of that matrix. The POD subspace of rank r is thus the rank-r subspace which minimizes the difference between the snapshots and their projections onto the subspace. That is, $\mathbf{U}_r$ satisfies:

$$\arg\min_{\mathbf{\Psi}_r \in \mathbb{R}^{N\times r}} \left\| \mathbf{S} - \mathbf{\Psi}_r \mathbf{\Psi}_r^\top \mathbf{S} \right\|_F^2, \tag{1.6}$$

and furthermore,

$$\left\| \mathbf{S} - \mathbf{U}_r \mathbf{U}_r^\top \mathbf{S} \right\|_F^2 = \sum_{i=r+1}^N \sigma_i^2. \tag{1.7}$$

Relative cumulative energy of the POD basis The quantity

$$\sum_{i=1}^{r} \sigma_i^2 \Big/ \sum_{i=1}^{N} \sigma_i^2, \tag{1.8}$$

which takes on values in $[0, 1]$, is referred to as the *relative cumulative energy* of the POD basis (or sometimes just the relative energy, energy, or total energy of the POD basis). The closer to 1 this quantity is, the better the ability of the POD basis to represent the data.

Choice of basis size Ideally, we would like to choose r so that the relative energy is high, say, above 0.9999. In problems that exhibit strong spectral decay, this can be achieved with very small r. For other problems, including many transport-dominated problems, many basis functions are required to represent the snapshot data well.

Connection to snapshot covariance and method of snapshots Note that the POD basis functions can alternatively be defined as satisfying

$$\mathbf{S}\mathbf{S}^\top \mathbf{u}_i = \sigma_i^2 \mathbf{u}_i. \tag{1.9}$$

In fact this definition as the leading eigenvectors of the covariance is the initial way POD was defined, and this is how we will define the POD basis in the continuous formulation.

Another term you may find in the literature is the 'method of snapshots'. This arises from the fact that the covariance $\mathbf{S}\mathbf{S}^\top \in \mathbb{R}^{N\times N}$ may be very large if N is large, making computing its eigendecomposition a computationally expensive task. In the method of snapshots, the POD basis is obtained by computing the eigendecomposition of the product $\mathbf{S}^\top\mathbf{S} \in \mathbb{R}^{K\times K}$:

$$\mathbf{S}^\top\mathbf{S} = \mathbf{V}\boldsymbol{\Lambda}\mathbf{V}^\top, \tag{1.10}$$

$$\mathbf{U} = \mathbf{S}\mathbf{V}\boldsymbol{\Lambda}^{-1/2}. \tag{1.11}$$

The above computation is cheaper than the covariance computation if we have few snapshots relative to the size of the problem. However, I introduce the SVD formulation first though because this is more numerically stable and is always how you should compute the POD basis practically.

1.2.4. The Galerkin-POD reduced model for the evolution problem

How does the POD subspace of rank r define a reduced model of *dimension* r? Via two steps: (i) approximation and (ii) projection. By approximation, I mean that we approximate the state within the span of the POD basis:

$$\mathbf{s}_{\text{POD}} = \sum_{i=1}^{r} \hat{\mathbf{s}}_i \mathbf{u}_i = \mathbf{U}_r \hat{\mathbf{s}}, \tag{1.12}$$

where $\hat{\mathbf{s}} \in \mathbb{R}^r$ is called the *reduced state.* The reduced state consists of the coefficients of the POD basis vector. To derive equations governing how the reduced state evolves, we enforce a Galerkin orthogonality condition:

$$\mathbf{U}_r^\top \left(\frac{\mathrm{d}\mathbf{s}_{\text{POD}}}{\mathrm{d}t} - \mathbf{A}\mathbf{s}_{\text{POD}} \right) = 0, \tag{1.13}$$

i.e., the residual of eq. (1.1) for our approximation has to be orthogonal to the POD basis. Substituting eq. (1.12) into eq. (1.13) and moving matrices around a bit yields

$$\frac{\mathrm{d}}{\mathrm{d}t}\mathbf{U}_r\mathbf{U}_r^\top\hat{\mathbf{s}} = \mathbf{U}_r^\top\mathbf{A}\mathbf{U}_r\hat{\mathbf{s}}. \tag{1.14}$$

Define $\hat{\mathbf{A}} = \mathbf{U}_r^\top\mathbf{A}\mathbf{U}_r \in \mathbb{R}^{r\times r}$ and recall that $\mathbf{U}_r$ is orthogonal so eq. (1.14) yields the following:

$$\frac{\mathrm{d}\hat{\mathbf{s}}}{\mathrm{d}t} = \hat{\mathbf{A}}\hat{\mathbf{s}}. \tag{1.15}$$

Equation (1.15) is called the *reduced model* or the *reduced-order model (ROM)* in the literature. The operator $\hat{\mathbf{A}}$ is sometimes called a *reduced operator* or *ROM operator*. Note that eq. (1.15) can be simulated at cost independent of the full model dimension N.

Remark: Petrov-Galerkin projection is also sometimes used, generally to obtain optimality results in certain norms. But vanilla Galerkin projection is more common.

Offline-online decomposition Note that $\hat{\mathbf{A}}$ depends only on the POD basis, which depends on the data. Once computed, it can be stored and used to rapidly simulate eq. (1.15) for many initial conditions. This lends itself to the following offline-online decomposition for building and then using reduced models:

1. Acquire snapshot data $\mathbf{S} \in \mathbb{R}^{N\times K}$, compute POD basis $\mathbf{U}_r \in \mathbb{R}^{N\times r}$, and project full operator $\mathbf{A} \in \mathbb{R}^{N\times N}$ onto POD subspace to obtain $\hat{\mathbf{A}} \in \mathbb{R}^{r\times r}$.
2. Use reduced operator $\hat{\mathbf{A}}$ to evaluate eq. (1.15) at many initial conditions/parameters, e.g. online in a control loop or for many samples in a Kalman filter or in a parameter optimization.

The offline-online decomposition is not the only way reduced models can be used. Many recent papers have been published which investigate *adaptive* reduced modeling methods which update the basis on-the-fly, e.g., along an optimization trajectory.

1.3. References

1. Peter Benner, Mario Ohlberger, Albert Cohen, and Karen Willcox. *Model reduction and approximation: Theory and algorithms.* SIAM, 2017
2. Kunihiko Taira et al. Modal analysis of fluid flows: An overview. *AIAA Journal*, 55(12):4013–4041, 2017

1.4. Continuous formulation

We now briefly present a formulation for POD-Galerkin reduced models in the spatially continuous PDE setting. Starting from the continuous PDE rather than the semidiscrete ODEs can be valuable from an analysis perspective, enabling some types of convergence and error analysis, and allows POD to be formulated in a way that is independent of the discretization strategy used to obtain the semi-discrete ODEs. This perspective is particularly common in the reduced-basis community, as we will see in later classes. My goal here is to present the continuous formulation by analogue to the semi-discrete formulation, glossing over some of the precise functional analysis details to emphasize key points. These details can be found in [2].

1.4.1. Set-up

For the sake of concreteness, we will now consider the one-dimensional heat equation as our starting point:

$$\frac{\partial s}{\partial t} = \frac{\partial^2 s}{\partial x^2}, \tag{1.16}$$

where $s \in L^2(\Omega)$ for some spatial domain $\Omega \subset \mathbb{R}$. Note that eq. (1.16) when discretized by, say, a central difference scheme on a uniform grid, would lead to a linear system of ODEs. The state $s(x,t)$ is now a function of time and space and thus at any given time t is *infinite*-dimensional. We can formulate the POD reduced model analogously to the discrete case to obtain a reduced model of finite dimension r.

1.4.2. Snapshot data

In this setting, the snapshot data are not vectors in Euclidean space but functions in L^2. Let

$$s_1, s_2, \dots, s_K \in L^2(\Omega) \tag{1.17}$$

be snapshots of the solution to eq. (1.16). Our previous discussion of the varied ways snapshots can be generated applies.

1.4.3. POD basis

The snapshots define an empirical covariance operator $C : L^2(\Omega) \to L^2(\Omega)$ whose action is given for any $\phi \in L^2(\Omega)$ by

$$C\phi = \sum_{l=1}^{K} \langle \phi, s_l \rangle \, \phi. \tag{1.18}$$

Because C is compact, non-negative, and self-adjoint, it has an orthonormal basis of eigenvectors satisfying

$$Cu_i = \lambda_i u_i, \tag{1.19}$$

where $\lambda_i \geq 0$ are in non-increasing order and $u_i \in L^2(\Omega)$ are are the corresponding orthogonal eigenfunctions. The leading r eigenfunctions of C define a POD basis of rank r:

$$\{u_i\}_{i=1}^{r}. \tag{1.20}$$

The span of the leading r eigenfunctions is the POD subspace.

Optimality property The POD basis satisfies the following optimality property:

$$\min \sum_{l=1}^{K} \left\| s_l - \sum_{i=1}^{r} \langle s_l, u_i \rangle \, u_i \right\|^2 = \sum_{i=r+1}^{K} \lambda_i. \tag{1.21}$$

Weighted POD We can consider the above formulation for snapshots in a Hilbert space to be a generalization of our initial presentation – if we replace $L^2(\Omega)$ with $\mathbb{R}^N$ and the inner product with the Euclidean inner product, we recover the semi-discrete presentation. In fact, when we use a discretization on a uniform grid to go from the continuous PDE to the semi-discrete ODEs, we can interpret the semi-discrete POD formulation with basis vectors for $\mathbb{R}^N$ as a discretization of the continuous POD formulation with basis functions in L^2.

However, if the discretization is non-uniform, the Euclidean norm of the discrete state may not be a good proxy for the L^2-norm of the state of the underlying continuous problem. This motivates the need for *weighted POD* for the discrete setting, where the POD basis vectors are obtained by solving the following weighted eigenvalue problem:

$$\mathbf{S}^\top \mathbf{W}\mathbf{S}\mathbf{u}_i = \lambda_i \mathbf{u}_i, \tag{1.22}$$

where $\mathbf{W}$ is a symmetric positive definite weight matrix.

1.4.4. Galerkin-POD for the PDE evolution problem

As in the discrete case, obtaining the reduced model in the continuous case proceeds first by approximation:

$$s_{\text{POD}}(x,t) = \sum_{i=1}^{r} \hat{s}_i(t) u_i(x), \tag{1.23}$$

and then by enforcing a Galerkin orthogonality condition:

$$\left\langle u_i, \frac{\partial s_{\text{POD}}}{\partial t} \right\rangle = \left\langle u_i, \nabla^2 s_{\text{POD}} \right\rangle, \qquad i = 1, 2, \dots, r. \tag{1.24}$$

Integration by parts and substituting the approximation into the above yields

$$\frac{\partial \hat{s}_i}{\partial t} = \sum_{j=1}^{r} \hat{s}_j(t) a(u_i, u_j), \qquad i = 1, 2, \dots, r, \tag{1.25}$$

where $a(\cdot,\cdot)$ is the bilinear form associated with the weak form of the heat equation. This is a system of r ODEs:

$$\frac{\partial \hat{s}}{\partial t} = \hat{A}\hat{s}, \tag{1.26}$$

where $\hat{A}_{ij} = a(u_i, u_j)$ and $\hat{s}$ is the reduced state. As in the discrete case, the reduced operator $\hat{A}$ can be precomputed and stored offline, allowing eq. (1.26) to be rapidly evaluated online.

Interpretation as spectral element method If you are familiar with the finite element method of solving PDEs, you can note that the form of eq. (1.25) is the same as what would result from a finite element discretization, with the difference being that in finite elements the basis functions u_i and u_j would be the hat functions (or other elements) and that there would generally be N of these basis functions. Thus, one can view the Galerkin-POD model as a discretization of the underlying PDE that uses global basis functions determined from data.

1.5. Extensions of POD

POD is very flexible. Let's talk about how we would adapt POD to the steady and parametrized cases:

1.5.1. Steady

Let's talk about how we would apply POD to the following steady problem:

$$\mathbf{A}\mathbf{s} = \mathbf{f}, \tag{1.27}$$

where $\mathbf{f} \in \mathbb{R}^N$ represents a source or forcing term. This kind of set up with no time-dependence shows up a lot in inverse problems with geological or medical imaging applications. To use POD we apply the core principles - use data to compute the POD basis, approximate the state in the POD subspace, and apply Galerkin projection.

In the steady case, the snapshots will comes from different realizations of the forcing vector $\mathbf{f}$. Let

$$\mathbf{A}\mathbf{s}_i = \mathbf{f}_i, \qquad i = 1, 2, \ldots, K. \tag{1.28}$$

The basis $\mathbf{U}_r$ is computed from the snapshots $\{\mathbf{s}_i\}_{i=1}^K$ using the SVD as before, and we again make the POD approximation $\mathbf{s}_{\text{POD}} = \mathbf{U}_r\hat{\mathbf{s}}$. We then substitute this into eq. (1.27) and apply Galerkin projection to get

$$\mathbf{U}_r^\top(\mathbf{A}\mathbf{U}_r\hat{\mathbf{s}} = \mathbf{f}), \tag{1.29}$$

$$\hat{\mathbf{A}}\hat{\mathbf{s}} = \hat{\mathbf{f}}, \tag{1.30}$$

where $\hat{\mathbf{f}} \in \mathbb{R}^r$ is the reduced source vector, containing the coefficients of $\mathbf{f}$ for the POD basis vectors.

Note that the computational savings for the steady problem are generally not as great as they are for the unsteady problem, because you only need to use your reduced operator once (to solve the system) rather than many times (to numerically integrate the system). Nevertheless, model reduction has real utility in settings where you need to solve for multiple source terms, as is the case in some tomography applications.

1.5.2. Parametrized

Another setting to which POD can be adapted is the parametrized setting. Let's return to the unsteady case:

$$\frac{\mathrm{d}\mathbf{s}}{\mathrm{d}t} = \mathbf{A}(\mu)\mathbf{s}, \tag{1.31}$$

where the operator $\mathbf{A}(\mu)$ depends on a parameter $\mu \in \mathcal{D}$, where $\mathcal{D} \subset \mathbb{R}^p$ is the parameter domain. For example, μ could be the thermal conductivity in the heat equation.

If we want to solve eq. (1.31) many times for different initial conditions but one parameter, we can apply POD as discussed last time. But if we want to approximate eq. (1.31) for many different parameters, we will run into trouble. Suppose we have snapshots from many representative parameters of the parameters we want to predict for. Then we can compute $\mathbf{U}_r$ via SVD as usual, and apply Galerkin projection to get:

$$\frac{\mathrm{d}\hat{\mathbf{s}}}{\mathrm{d}t} = \mathbf{U}_r^\top\mathbf{A}(\mu)\mathbf{U}_r\hat{\mathbf{s}}. \tag{1.32}$$

The problem with the above is that because $\mathbf{A}(\mu)$ changes with each new μ, we cannot in general pre-compute a reduced operator that can be rapidly used to simulate the system at new parameters. The reduced basis method, which we will cover next in the course, exploits *affine parameter dependence* to yield an efficient offline-online decomposition. But even absent structure, there are a few strategies that have been used to apply POD to the parametrized setting:

1. Local/nearest neighbor: during the offline phase, choose several training parameter values in $\mathcal{D}$ and compute a reduced operator for each of these training parameters. Then in the online phase, for each new parameter encountered, use the reduced operator for the training parameter closest to the new parameter.

2. Interpolate: during the offline phase, choose several training parameter values in $\mathcal{D}$ and compute a reduced operator for each of these training parameters. Then in the online phase, for each new parameter encountered, interpolate the operators of surrounding training parameters appropriately.

These are basic ideas and there are a number of papers that spin off of these basic ideas.

1.5.3. Polynomial nonlinearities

Consider the following ODE with quadratic nonlinearity:

$$\frac{\mathrm{d}\mathbf{s}}{\mathrm{d}t} = \mathbf{A}\mathbf{s} + \mathbf{H}(\mathbf{s} \otimes \mathbf{s}), \tag{1.33}$$

where $\otimes$ denotes the Kronecker product and $\mathbf{H} \in \mathbb{R}^{N \times N^2}$. Approximating $\mathbf{s}_{\text{POD}} = \mathbf{U}_r\hat{\mathbf{s}}$ and applying Galerkin projection yields

$$\frac{\mathrm{d}\hat{\mathbf{s}}}{\mathrm{d}t} = \hat{\mathbf{A}}\hat{\mathbf{s}} + \hat{\mathbf{H}}(\hat{\mathbf{s}} \otimes \hat{\mathbf{s}}), \tag{1.34}$$

where $\hat{\mathbf{A}} \in \mathbb{R}^{r \times r}$ as before and $\hat{\mathbf{H}} = \mathbf{U}_r^\top \mathbf{H}(\mathbf{U}_r \otimes \mathbf{U}_r) \in \mathbb{R}^{r \times r^2}$. In the polynomial case we see that the reduced operators have an efficient representation that lets the polynomial reduced model be efficiently evaluated online.

1.5.4. General nonlinearities

Consider the following ODE with general nonlinearity:

$$\frac{\mathrm{d}\mathbf{s}}{\mathrm{d}t} = \mathbf{f}(\mathbf{s}), \tag{1.35}$$

where $\mathbf{f} : \mathbb{R}^N \to \mathbb{R}^N$ is Lipschitz continuous (so that the ODEs admit a unique solution). Approximating $\mathbf{s}_{\text{POD}} = \mathbf{U}_r\hat{\mathbf{s}}$ and applying Galerkin projection yields

$$\frac{\mathrm{d}\hat{\mathbf{s}}}{\mathrm{d}t} = \mathbf{U}_r^\top \mathbf{f}(\mathbf{U}_r\hat{\mathbf{s}}). \tag{1.36}$$

This may be an r-dimensional representation, but computing it requires evaluating the N-dimensional $\mathbf{f}$. This is expensive, and is referred to as the computational bottleneck of nonlinear model reduction.

The state-of-the-art to make this more efficient is to introduce another layer of approximation by interpolating $\mathbf{f}$. We will describe this in more detail when we get to nonlinear model reduction.

1.6. POD error/performance

1.6.1. Components of POD error

Now let's consider the error between the POD solution to a given initial value problem and the solution of the full model:

$$\mathbf{e}_{\mathrm{POD}}(t) = \|\mathbf{s}(t) - \mathbf{s}_{\mathrm{POD}}(t)\|_2. \tag{1.37}$$

There are two components of this error, we will denote them $\mathbf{e}_i$ and $\mathbf{e}_o$ for the 'in-space' and 'out-of-space' errors. As their names imply, $\mathbf{e}_i$ lies within the POD subspace and $\mathbf{e}_o$ is perpendicular to the POD subspace. The in-space error arises due to the fact that the POD model neglects the effects of the neglected modes on the modeled modes - this error accumulates through time integration. If the characteristics of the problem are such that high modes have a big effect on low modes, or if we do not retain enough modes to represent the state well, this can lead to a very high POD error. This effect is called the closure, and is conceptually very closely related to subgrid scale modeling, although the math looks a little different.

1.6.2. Characteristics that make problems amenable to/difficult for POD

Characteristics of problems amenable to POD:

- POD basis can represent well the state in the desired range of prediction – depends on your snapshot data used to compute the basis as well as the number of basis vectors you choose to retain. Lots of strategies for choosing snapshot data to try to get a better prediction exist, very problem dependent.
- not too much energy transfer between retained and unmodeled POD modes – this is a characteristic of the problem. POD alone can't really get around this in its vanilla form, but there has been work to model the closure in different ways.

Characteristics known to be tough for POD:

- transport-dominated problems (leads to slow spectral decay of state data)
- high-dimensional parametrized problems without some additional regularity (expensive to get snapshot data that represents the desired prediction range well)

1.6.3. Error bounds

We would like an *a posteriori* error bound over the following integrated POD error:

$$\int_0^T \|\mathbf{s}(t) - \mathbf{s}_{\mathrm{POD}}(t)\|^2 \, \mathrm{d}t. \tag{1.38}$$

To bound this it will be convenient to consider a POD basis satisfying a continuous-time version of the POD optimality condition.

Let the POD basis $\mathbf{U}_r$ solve the problem

$$\min_{\substack{\mathbf{\Psi}_r \in \mathbb{R}^{N \times r} \\ \mathbf{\Psi}_r^\top \mathbf{\Psi}_r = I}} \int_0^T \left\| \mathbf{s}(t) - \mathbf{\Psi}_r \mathbf{\Psi}_r^\top \mathbf{s}(t) \right\|^2 \mathrm{d}t. \tag{1.39}$$

If the snapshots for POD come from sampling $\mathbf{s}(t)$ at uniformly-spaced times in $[0, T]$, we can think of our usual formulation as an approximation to this continuous-time formulation.

Then, if $\mathbf{s}_{\text{POD}}(t)$ is the POD approximation using the basis satisfying eq. (1.39), we have the following bound for some $C > 0$:

$$\int_0^T \|\mathbf{s}(t) - \mathbf{s}_{\text{POD}}(t)\|^2 \,\mathrm{d}t \leq C \sum_{i=r+1}^{n} (\sigma_i^2 + \int_0^T \left(\dot{\mathbf{s}}(t)^\top \mathbf{u}_i)^2 \mathrm{d}t\right). \tag{1.40}$$

This structure is typical of the types of error estimates and bounds that we will encounter - there is a term due to the projection error of the initial condition and an error accumulation term. Note that the integral incorporates both $\mathbf{e}_i$ and $\mathbf{e}_o$ discussed above because it is integrating the neglected components of the *true* time derivative.

This bound is not particularly useful though because it requires that $\dot{\mathbf{s}}(t)$ – the time derivative the true solution – be available. Theorists have proposed including snapshots of $\dot{\mathbf{s}}$ in the computation of the basis so that the bound will only depend on the trailing singular values of the data. This leads to a nice bound, but in practice it is unclear that this makes a difference, because usually the time derivative is approximated from state data.

The research community in reduced-basis methods has developed efficiently computable *a posteriori* error estimates.

2. The reduced-basis method for affine elliptic problems

2.1. Finite element refresher

Because of the focus on error analysis in the reduced-basis community, the departure point for deriving reduced basis models is usually the weak form of an elliptic PDE. We will consider a concrete example first and then make things more general.

2.1.1. Parameter-dependent weak form

Consider the heat equation on the one-dimensional unit domain $\Omega = (0, 1)$:

$$\mu \frac{\partial^2 s}{\partial x^2} = \mathfrak{f}, \tag{2.1}$$

where $\mathfrak{f} \in L^2(\Omega)$. We assume homogeneous boundary conditions, and seek a weak solution in L^2, i.e., we seek to find $s \in L^2(\Omega)$ satisfying

$$\mu \int_0^1 s' \frac{\partial^2 s}{\partial x^2} = \int_0^1 \mathfrak{f} s', \qquad \forall s' \in L^2(\Omega). \tag{2.2}$$

Integrating by parts yields:

$$-\mu \int_0^1 \frac{\partial s}{\partial x} \frac{\partial s'}{\partial x} = \int_0^1 \mathfrak{f} s' \qquad \forall s' \in L^2(\Omega). \tag{2.3}$$

We define $a(s, s'; \mu) = -\mu \int_0^1 \frac{\partial s}{\partial x} \frac{\partial s'}{\partial x}$ and $f(s') = \int_0^1 \mathfrak{f} s'$. Thus the weak form can be written in the following familiar general way: find $s \in L^2(\Omega)$ satisfying

$$a(s, s'; \mu) = f(s'), \qquad s' \in L^2(\Omega). \tag{2.4}$$

The parameter-dependent $a(s, s'; \mu)$ is assumed to be symmetric and coercive (we will define this later), and the parameter μ is allowed to take on values in $\mathcal{D} \in \mathbb{R}^p$.

2.1.2. Finite element solution

To numerically solve eq. (2.4) using finite elements, we let $\{\phi_i\}_{i=1}^N$ denote a set of basis functions in $L^2(\Omega)$. In finite elements, these $\{\phi_i\}_{i=1}^N$ are often 'hat functions' which define a nodal basis on a grid $\{x_i\}_{i=1}^N$. The span of such a set of nodal basis functions is the set of all piecewise linear functions on the grid. To discretize eq. (2.4), we approximate the solution s in the span of the basis functions:

$$s_{\text{FE}}(x) \approx \sum_{i=1}^N \mathbf{s}_i \phi_i(x), \tag{2.5}$$

and enforce the following Galerkin orthogonality condition for our approximation:

$$a(s_{\mathrm{FE}}, \phi_i\,; \mu) = f(\phi_i), \qquad i = 1, 2, \ldots, N. \tag{2.6}$$

Equation (2.6) describes a set of N equations, one for each i. Substituting eq. (2.5) into eq. (2.6) and invoking bilinearity of the form $a(\cdot, \cdot; \mu)$ yields

$$\sum_{j=1}^{N} \mathbf{s}_j a(\phi_j, \phi_i\,; \mu) = f(\phi_i). \tag{2.7}$$

We can collect the $f(\phi_i)$ into a vector $\mathbf{f} \in \mathbb{R}^N$ and the $a(\phi_j, \phi_i\,; \mu)$ into a matrix $\mathbf{A}(\mu) \in \mathbb{R}^{N\times N}$ as follows:

$$\mathbf{A} = \begin{bmatrix} a(\phi_1, \phi_1\,; \mu) & a(\phi_2, \phi_1\,; \mu) & \cdots & a(\phi_N, \phi_1\,; \mu) \\ a(\phi_1, \phi_2\,; \mu) & a(\phi_2, \phi_2\,; \mu) & \cdots & a(\phi_N, \phi_2\,; \mu) \\ \vdots & \vdots & \ddots & \vdots \\ a(\phi_1, \phi_N\,; \mu) & a(\phi_2, \phi_N\,; \mu) & \cdots & a(\phi_N, \phi_N\,; \mu) \end{bmatrix}, \qquad \mathbf{f} = \begin{bmatrix} f(\phi_1) \\ f(\phi_2) \\ \vdots \\ f(\phi_N), \end{bmatrix} \tag{2.8}$$

or, compactly, $\mathbf{f}_i = f(\phi_i)$, and $\mathbf{A}_{ij}(\mu) = a(\phi_j, \phi_i\,; \mu)$. This leads to the following set of matrix equations that we solve using a computer:

$$\mathbf{A}\mathbf{s} = \mathbf{f}, \tag{2.9}$$

where $\mathbf{s} \in \mathbb{R}^N$ contains the coefficients of the finite element basis functions. For a nodal basis, this means that $\mathbf{s}_i \approx s(x_i)$.

2.2. Affine parameter dependence

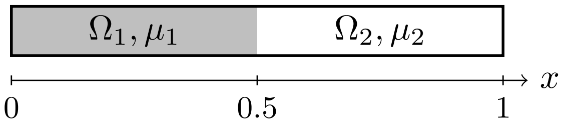
We will consider reduced-basis approximation for problems in which the parameter-dependent bilinear form $a(s, s'; \mu)$ exhibits affine structure in its parameter dependence. That is, we will assume that $a(s, s'; \mu)$ can be written as follows:

$$a(s, s'; \mu) = \sum_{q=1}^{Q} \theta_q(\mu) a_q(s, s') \tag{2.10}$$

for some scalar functions $\theta_q : \mathcal{D} \to \mathbb{R}$, for $q = 1, 2, \ldots, Q$, and *parameter-independent* bilinear forms $a_q(s, s')$. We also would generally like eq. (2.10) to be true for a small number of terms Q.

2.2.1. Example problem with affine parameter dependence

Consider the solution of the heat equation on a bar of length 1 made of two materials, where each material has its own conductivity:



Then, the weak form of the heat equation is

$$\mu_1 \int_0^{0.5} s' \frac{\partial^2 s}{\partial x^2} + \mu_2 \int_{0.5}^{1} s' \frac{\partial^2 s}{\partial x^2} = \int_0^1 \mathfrak{f} s', \tag{2.11}$$

which can be re-written as

$$a(s, s'; \mu) = \sum_{q=1}^{2} \theta_q(\mu) a_q(s, s') = f(s'), \tag{2.12}$$

where $\theta_1(\mu) = \mu_1$ and $\theta_2(\mu) = \mu_2$, and

$$a_1(s, s') = -\mu_1 \int_0^{0.5} \frac{\partial s}{\partial x} \frac{\partial s'}{\partial x}, \qquad a_2(s, s') = -\mu_2 \int_{0.5}^{1} \frac{\partial s}{\partial x} \frac{\partial s'}{\partial x}. \tag{2.13}$$

2.2.2. Finite element solution of affine problems

Approximating the state s in the span of the finite element basis as in eq. (2.5) and enforcing the Galerkin orthogonality condition in the affine weak form eq. (2.12) yields:

$$\sum_{q=1}^{2} \theta_q(\mu) a_q(s_{\rm FE}, \phi_i) = f(\phi_i), \qquad i = 1, 2, \dots, N. \tag{2.14}$$

Expanding the FE approximation and invoking bilinearity of the forms a_q yields

$$\sum_{j=1}^{N} \sum_{q=1}^{2} \theta_q(\mu) \mathbf{s}_j a_q(\phi_j, \phi_i) = f(\phi_i), \qquad i = 1, 2, \dots, N. \tag{2.15}$$

We can define $\mathbf{f} \in \mathbb{R}^N$ as before, and define the parameter-independent matrices $\mathbf{A}_q \in \mathbb{R}^{N \times N}$ as:

$$\mathbf{A}_q = \begin{bmatrix} a_q(\phi_1, \phi_1) & a_q(\phi_2, \phi_1) & \cdots & a_q(\phi_N, \phi_1) \\ a_q(\phi_1, \phi_2) & a_q(\phi_2, \phi_2) & \cdots & a_q(\phi_N, \phi_2) \\ \vdots & \vdots & \ddots & \vdots \\ a_q(\phi_1, \phi_N) & a_q(\phi_2, \phi_N) & \cdots & a_q(\phi_N, \phi_N). \end{bmatrix} \tag{2.16}$$

Then, eq. (2.15) yields

$$\mathbf{A}(\mu)\mathbf{s} = \left(\sum_{q=1}^{Q} \theta_q(\mu) \mathbf{A}_q \right) \mathbf{s} = \mathbf{f}. \tag{2.17}$$

2.3. Reduced-basis method

The reduced-basis method approximates the finite solution $s_{\rm FE}$ not in the full N-dimensional finite element space, but in a reduced subspace. Reduced-basis methods exploit the affine structure of the parameter dependence to yield computationally efficient approximations.

2.3.1. Reduced-basis approximation (usual formulation)

We denote by $\{u_i\}_{i=1}^r$ a reduced basis of size r. For now we assume we have the basis given to us in order to illustrate how the reduced-basis approximation exploits the affine parametric dependence for computational efficiency. We will discuss the greedy procedure for building the basis in the next section.

The reduced-basis state $s_{\text{RB}}(\mu)$ approximates the finite element state $s_{\text{FE}}(\mu)$ in the span of this reduced basis:

$$s_{\text{RB}}(\mu) = \sum_{i=1}^{r} \hat{s}_i(\mu) u_i. \tag{2.18}$$

The reduced-basis model is obtained by substituting eq. (2.18) into the weak form and enforcing a Galerkin orthogonality condition:

$$a(s_{\text{RB}}(\mu), u_i; \mu) = f(u_i), \qquad i = 1, 2, \ldots, r. \tag{2.19}$$

Again, eq. (2.19) defines a system of r equations, which we can re-write using bilinearity of $a(\cdot, \cdot; \mu)$ as follows:

$$\sum_{j=1}^{r} \hat{s}_j(\mu) a(u_j, u_i\,; \mu) = f(u_i), \qquad i = 1, 2, \ldots, r. \tag{2.20}$$

Now, we invoke the assumption of affine parameter dependence to expand the bilinear form $a(\cdot, \cdot; \mu)$:

$$\sum_{j=1}^{r} \sum_{q=1}^{Q} \hat{s}_j(\mu) \theta_q(\mu) a_q(u_j, u_i) = f(u_i), \qquad i = 1, 2, \ldots, r. \tag{2.21}$$

We can collect the $f(u_i)$ into a vector $\hat{\mathbf{f}} \in \mathbb{R}^r$ and the $a_q(u_j, u_i\,)$ into matrices $\hat{\mathbf{A}}_q \in \mathbb{R}^{r \times r}$:

$$\hat{\mathbf{A}}_q = \begin{bmatrix} a_q(u_1, u_1) & a_q(u_2, u_1) & \cdots & a_q(u_r, u_1) \\ a_q(u_1, u_2) & a_q(u_2, u_2) & \cdots & a_q(u_r, u_2) \\ \vdots & \vdots & \ddots & \vdots \\ a_q(u_1, u_r) & a_q(u_2, u_r) & \cdots & a_q(u_r, u_r). \end{bmatrix}, \qquad \hat{\mathbf{f}} = \begin{bmatrix} f(u_1) \\ f(u_2) \\ \vdots \\ f(u_r) \end{bmatrix}. \tag{2.22}$$

The basis $\{u_i\}_{i=1}^r$ is computed in the offline phase, and the parameter-independent reduced operators $\hat{\mathbf{A}}_q$, as well as the reduced forcing $\hat{\mathbf{f}}$, can also be computed in the offline phase. Then, during the online phase, given a new parameter μ, we can evaluate the following efficient r-dimensional model:

$$\left(\sum_{q=1}^{Q} \theta_q(\mu) \hat{\mathbf{A}}_q \right) \hat{\mathbf{s}} = \hat{\mathbf{A}}(\mu) \hat{\mathbf{s}} = \hat{\mathbf{f}}, \tag{2.23}$$

where $\hat{\mathbf{s}} \in \mathbb{R}^r$ is the reduced state containing coefficients of reduced basis functions $\{u_i\}_{i=1}^r$. Note that the assembly of $\hat{\mathbf{A}}(\mu)$ is efficient because it requires only the evaluation of Q scalar functions and summing over Q matrices in $\mathbb{R}^{r \times r}$.

2.3.2. Reduced-basis approximation (an experiment with a discrete formulation)

Suppose we are given a basis $\mathbf{U}_r \in \mathbb{R}^{N \times r}$. The reduced basis state approximation is given by

$$\mathbf{s}_{\text{RB}}(\mu) = \sum_{i=1}^{r} \hat{\mathbf{s}}_i(\mu) \mathbf{u}_i = \mathbf{U}_r \hat{\mathbf{s}}, \tag{2.24}$$

where $\hat{\mathbf{s}} \in \mathbb{R}^r$ is the vector of coefficients, as in our original formulation of POD. Then, the reduced model is obtained by substituting eq. (2.24) into the discrete finite element equations eq. (2.17) as follows:

$$\mathbf{A}(\mu)\mathbf{s}_{\mathrm{RB}} = \left(\sum_{q=1}^{Q} \theta_q(\mu) \mathbf{A}_q \right) \mathbf{s}_{\mathrm{RB}} = \mathbf{f}, \tag{2.25}$$

and enforcing Galerkin orthogonality to the basis $\mathbf{U}_r$:

$$\mathbf{U}_r^\top \mathbf{A}(\mu)\mathbf{s}_{\mathrm{RB}} = \mathbf{U}_r^\top \left(\sum_{q=1}^{Q} \theta_q(\mu) \mathbf{A}_q \right) \mathbf{s}_{\mathrm{RB}} = \mathbf{U}_r^\top \mathbf{f}, \tag{2.26}$$

which can be re-written using eq. (2.24) as

$$\left(\sum_{q=1}^{Q} \theta_q(\mu) \mathbf{U}_r^\top \mathbf{A}_q \mathbf{U}_r \right) \hat{\mathbf{s}} = \mathbf{U}_r^\top \mathbf{f}. \tag{2.27}$$

Defining $\hat{\mathbf{A}}_q = \mathbf{U}_r^\top \mathbf{A}_q \mathbf{U}_r$ for $q = 1, 2, \ldots, Q$, and $\hat{\mathbf{f}} = \mathbf{U}_r^\top \mathbf{f}$, we get

$$\left(\sum_{q=1}^{Q} \theta_q(\mu) \hat{\mathbf{A}}_q \right) \hat{\mathbf{s}} = \hat{\mathbf{f}}, \tag{2.28}$$

which is the same placed we arrived from the continuous formulation (analogous to the discrete and continuous POD formulations).

2.3.3. Greedy approach to building the basis

Until now we have taken the reduced basis $\{u_i\}_{i=1}^r$ as given. In the reduced-basis approach for parameter dependent problems, the basis functions are (more or less) taken to be snapshots of the state solution at different parameters. The more or less part is that in general we will apply a Gram-Schmidt orthogonalization procedure to the snapshots to obtain basis vectors in order to improve the conditioning of the basis.

The reduced-basis approach is different from the POD approach in that the POD approach generally assumes the availability of a large number of snapshots, and so we take the dominant POD modes of the data to be our basis. In the reduced-basis context, we do not assume we have any data, so we have to choose carefully which r parameter values we want to evaluate the full model at to obtain snapshots. So the question is, how do we choose the r parameters at which take snapshots to make the reduced basis?

The greedy approach uses a heuristic indicator of error to choose basis functions until a specified tolerance on the error indicator is met. Essentially, the greedy approach initializes the basis with a single snapshot at an initial parameter, and then iteratively finds the parameter in $\mathcal{D}$ with the highest error indicator and adds the snapshot at that parameter to the basis. Algorithm 1 summarizes the Greedy approach to building the basis.

In the reduced-basis method, the error indicator $\Delta^r(\mu)$ that is usually employed is an *a posteriori* error bound.

Algorithm 1 Greedy algorithm

```
0: Inputs: training parameter sample $M = \{\mu_m\}$, error indicator $\Delta^r(\mu)$, error tolerance $\tau$
   Initialize: $r = 1$, $u_1 = s_{\rm FE}(\mu_1)/\|s_{\rm FE}(\mu_1)\|_X$
   while $\max_{\mu\in M} \Delta^r(\mu) \geq \tau$ do
      $\mu_{\rm add} = \arg\max_{\mu\in M} \Delta^r(\mu)$
      $u_{r+1} = \texttt{gramschmidt}\big(s_{\rm FE}(\mu_{\rm add}), \{u_i\}_{i=1}^r\big)$
      $r \leftarrow r + 1$
   end while
```

2.4. A posteriori error estimation (usual formulation)

2.4.1. Norms and other sundries

Definition 2.1. *The parameter-dependent* energy inner product *and* energy norm *are given by:*

$$\langle s, s'\rangle_\mu = a(s, s'; \mu), \qquad \|s\|_\mu = a(s, s; \mu). \tag{2.29}$$

Definition 2.2. *The parameter-independent* X-inner product *and induced* X-norm *are given by selecting some reference parameter* $\bar\mu \in \mathcal{D}$*:*

$$\langle s, s'\rangle_X = a(s, s'; \bar\mu), \qquad \|s\|_X = a(s, s; \bar\mu). \tag{2.30}$$

Theorem 2.1. *Assume* $a(s, s'; \mu)$ *is a symmetric bilinear form with affine parameter dependence, and additionally assume that the associated parameter-dependent functions* $\theta_q(\mu)$ *satisfy* $\theta_q(\mu) > 0$ *for all* $\mu \in \mathcal{D}$*. Define* $\alpha_{\rm LB}(\mu)$ *as follows:*

$$\alpha_{\rm LB}(\mu) = \min_{q=1,2,\ldots,Q} \frac{\theta_q(\mu)}{\theta_q\bar\mu}. \tag{2.31}$$

Then, $\alpha_{\rm LB}$ *is a coercivity lower bound for* $a(s, s'; \mu)$*, that is,*

$$0 < \alpha_{\rm LB}(\mu) \leq \inf_{s\in X} \frac{a(s, s; \mu)}{\|s\|_X^2}, \qquad \forall \mu \in D. \tag{2.32}$$

Proof. Omitting for now, it is exactly analogous to the proof below in the discrete case. □

Definition 2.3. *The quantity of interest* $y(\mu)$ *is called a* compliant output *for the weak form given by*

$$a(s, s'; \mu) = f(s') \quad \forall s' \in X \tag{2.33}$$

if $y(\mu) = f(s(\mu))$.

Additionally, here is a review of some functional analysis.

Definition 2.4. *Given a Hilbert space* X*, its dual space* X' *is the space of all bounded linear functionals on* X.

Theorem 2.2. ***Riesz representation theorem.*** *Given any* $\psi \in X'$*, there exists a* $\hat\psi \in X$ *such that*

$$\psi(s) = \left\langle s, \hat\psi \right\rangle_X \quad \forall s \in X. \tag{2.34}$$

Additionally,

$$\|\psi\|_{X'} \equiv \sup_{s\in X} \frac{\psi(s)}{\|s\|}_X = \|\hat\psi\|_X. \tag{2.35}$$

2.4.2. Error bounds

Recall the following equations for our finite element truth and reduced-basis approximation:

$$a(s_{\mathrm{FE}}(\mu), s'; \mu) = f(s'), \qquad \forall s' \in X, \tag{2.36}$$

$$a(s_{\mathrm{RB}}(\mu), s'; \mu) = f(s'), \qquad \forall s' \in X_r \subset X. \tag{2.37}$$

The residual of the reduced-basis approximation is defined to be

$$r(s'; \mu) = f(s') - a(s_{\mathrm{RB}}(\mu), s'; \mu), \qquad \forall s' \in X. \tag{2.38}$$

Thus, subtracting eq. (2.37) from eq. (2.36), and using the definition eq. (2.38), we obtain the *error-residual relationship*:

$$a(s_{\mathrm{FE}}(\mu) - s_{\mathrm{RB}}(\mu), s'; \mu) = a(e_{\mathrm{RB}}(\mu), s'; \mu) = r(s'; \mu), \qquad \forall s' \in X. \tag{2.39}$$

We wish to bound the error $e_{\mathrm{RB}}(\mu) = s_{\mathrm{FE}}(\mu) - s_{\mathrm{RB}}(\mu)$ in the X- and energy norms. To do this, we invoke the Riesz representation theorem to say that there exists an $\hat{e}(\mu) \in X$ satisfying

$$r(s'; \mu) = \left\langle \hat{e}(\mu), s' \right\rangle_X \qquad \forall s' \in X. \tag{2.40}$$

Thus, eq. (2.39) becomes

$$a(e_{\mathrm{RB}}(\mu), s'; \mu) = \left\langle \hat{e}(\mu), s' \right\rangle_X, \forall s' \in X. \tag{2.41}$$

We now let $s' = e_{\mathrm{RB}}(\mu)$ – we can do this because $e_{\mathrm{RB}}(\mu) \in X$ – to obtain

$$a(e_{\mathrm{RB}}(\mu), e_{\mathrm{RB}}(\mu); \mu) = \langle \hat{e}(\mu), e_{\mathrm{RB}}(\mu) \rangle_X, \forall s' \in X. \tag{2.42}$$

But the left-hand side of eq. (2.42) can be lower-bounded by Theorem 2.1 as follows:

$$\alpha_{\mathrm{LB}}(\mu) \| e_{\mathrm{RB}}(\mu) \|_X^2 \leq a(e_{\mathrm{RB}}(\mu), e_{\mathrm{RB}}(\mu); \mu), \tag{2.43}$$

and the right-hand side of eq. (2.42) can be upper-bounded by the Cauchy-Schwarz inequality, so we get

$$\alpha_{\mathrm{LB}}(\mu) \| e_{\mathrm{RB}}(\mu) \|_X^2 \leq \| \hat{e}(\mu) \|_X \| e_{\mathrm{RB}}(\mu) \|_X. \tag{2.44}$$

Re-arranging the above yields our first result.

Theorem 2.3. *A bound on the X-norm of the reduced-basis error:*

$$\| e_{\mathrm{RB}}(\mu) \|_X \leq \frac{\| \hat{e}(\mu) \|_X}{\alpha_{\mathrm{LB}}(\mu)}. \tag{2.45}$$

Proof. See derivation preceding. □

Here are two additional results.

Theorem 2.4. *A bound on the energy norm of the reduced-basis error:*

$$\| e_{\mathrm{RB}}(\mu) \|_\mu \leq \frac{\| \hat{e}(\mu) \|_X}{\sqrt{\alpha_{\mathrm{LB}}(\mu)}}. \tag{2.46}$$

Proof. Starting from eq. (2.42), note that $a(e_{\mathrm{RB}}(\mu), e_{\mathrm{RB}}(\mu); \mu) = \|e_{\mathrm{RB}}(\mu)\|_\mu^2$, so that

$$\|e_{\mathrm{RB}}(\mu)\|_\mu^2 \leq \langle \hat{e}(\mu), e_{\mathrm{RB}}(\mu)\rangle_X \leq \|\hat{e}(\mu)\|_X \|e_{\mathrm{RB}}(\mu)\|_X, \tag{2.47}$$

where the second inequality comes from Cauchy-Schwarz. Then, using the result in Theorem 2.3, we have

$$\|e_{\mathrm{RB}}(\mu)\|_\mu^2 \leq \|\hat{e}(\mu)\|_X \frac{\|\hat{e}(\mu)\|_X}{\alpha_{\mathrm{LB}}(\mu)}. \tag{2.48}$$

Taking square roots of both sides yields the result. □

Theorem 2.5. *A bound on the error in a compliant output $y(\mu)$:*

$$|y_{\mathrm{FE}}(\mu) - y_{\mathrm{RB}}(\mu)| \leq \frac{\|\hat{e}(\mu)\|_X^2}{\alpha_{\mathrm{LB}}(\mu)}. \tag{2.49}$$

Proof. Note that since y is compliant, $y_{\mathrm{FE}}(\mu) = f(s_{\mathrm{FE}}(\mu))$ and $y_{\mathrm{RB}}(\mu) = f(s_{\mathrm{RB}}(\mu))$. Then,

$$|y_{\mathrm{FE}}(\mu) - y_{\mathrm{RB}}(\mu)| = f(e_{\mathrm{RB}}(\mu)) = a(s_{\mathrm{FE}}(\mu), e_{\mathrm{RB}}(\mu); \mu), \tag{2.50}$$

where the second equality comes from the weak form eq. (2.36). Then, because $s_{\mathrm{FE}}(\mu) = s_{\mathrm{RB}}(\mu) + e_{\mathrm{RB}}(\mu)$, and noting that the reduced-basis error is orthogonal to $s_{\mathrm{RB}}(\mu)$ due to Galerkin orthogonality, we have

$$f(e_{\mathrm{RB}}(\mu)) = a(e_{\mathrm{RB}}(\mu), e_{\mathrm{RB}}(\mu); \mu) = \|e_{\mathrm{RB}}(\mu)\|_\mu^2. \tag{2.51}$$

Invoking Theorem 2.4 yields the result. □

2.4.3. Efficient computation of error bounds

See below for computation in discretized space... I'll update this with the continuous formulation later, but it's cleaner in discrete space.

2.5. A posteriori error estimation (experiment in discrete formulation)

We start by defining discrete analogues of the norms and conditions developed in section 2.4.1.

Definition 2.5. *The parameter-dependent* energy inner product *and* energy norm *are given by*

$$\left\langle \mathbf{s}, \mathbf{s}' \right\rangle_\mu = \mathbf{s}^\top \mathbf{A}(\mu)\mathbf{s}, , \qquad \|\mathbf{s}\|_\mu = \mathbf{s}^\top \mathbf{A}(\mu)\mathbf{s}. \tag{2.52}$$

Definition 2.6. *The parameter-independent X-*inner product *and X-*norm *are given by*

$$\left\langle \mathbf{s}, \mathbf{s}' \right\rangle_{\mathbf{X}} = \mathbf{s}^\top \mathbf{X}\mathbf{s}, \qquad \|\mathbf{s}\|_{\mathbf{X}} = \mathbf{s}^\top \mathbf{X}\mathbf{s}, \tag{2.53}$$

where $\mathbf{X} = \mathbf{A}(\bar{\mu})$ for some fixed reference parameter $\bar{\mu}$.

Definition 2.7. *The quantity of interest $y(\mu)$ is called a* compliant *output if*

$$y(\mu) = \mathbf{f}^\top \mathbf{s}(\mu). \tag{2.54}$$

Theorem 2.6. *Let $\alpha_{\rm LB}(\mu)$ be given by the 'min-θ' method as follows:*

$$\alpha_{\rm LB}(\mu) = \min_{q=1,2,\ldots,Q} \frac{\theta_q(\mu)}{\theta_q(\bar\mu)}. \tag{2.55}$$

Then, $\alpha_{\rm LB}(\mu) \leq \inf_{\mathbf{s}\in\mathbb{R}^N} \frac{\mathbf{s}^\top \mathbf{A}(\mu)\mathbf{s}}{\|\mathbf{s}\|^2_{\mathbf{X}}}$.

Proof. Note that

$$\inf_{\mathbf{s}\in\mathbb{R}^N} \frac{\mathbf{s}^\top \mathbf{A}(\mu)\mathbf{s}}{\|\mathbf{s}\|^2_{\mathbf{X}}} = \inf_{\mathbf{s}\in\mathbb{R}^N} \frac{\mathbf{s}^\top \mathbf{A}(\mu)\mathbf{s}}{\mathbf{s}^\top \mathbf{A}(\bar\mu)\mathbf{s}} = \inf_{\mathbf{s}\in\mathbb{R}^N} \frac{\sum_{q=1}^Q \theta_q(\mu)\mathbf{s}^\top \mathbf{A}_q\mathbf{s}}{\sum_{q=1}^Q \theta_q(\bar\mu)\mathbf{s}^\top \mathbf{A}_q\mathbf{s}}. \tag{2.56}$$

Also note that:

$$\sum_{q=1}^Q \theta_q(\mu)\mathbf{s}^\top \mathbf{A}_q\mathbf{s} = \sum_{q=1}^Q \frac{\theta_q(\bar\mu)}{\theta_q(\bar\mu)}\theta_q(\mu)\mathbf{s}^\top \mathbf{A}_q\mathbf{s} \geq \alpha_{\rm LB}(\mu)\sum_{q=1}^Q \theta_q(\bar\mu)\mathbf{s}^\top \mathbf{A}_q\mathbf{s}. \tag{2.57}$$

Combining the two expressions above yields

$$\frac{\alpha_{\rm LB}(\mu)\sum_{q=1}^Q \theta_q(\bar\mu)\mathbf{s}^\top \mathbf{A}_q\mathbf{s}}{\sum_{q=1}^Q \theta_q(\bar\mu)\mathbf{s}^\top \mathbf{A}_q\mathbf{s}} = \alpha_{\rm LB}(\mu) \leq \inf_{\mathbf{s}\in\mathbb{R}^N} \frac{\mathbf{s}^\top \mathbf{A}(\mu)\mathbf{s}}{\|\mathbf{s}\|^2_{\mathbf{X}}}. \tag{2.58}$$

□

The Riesz representation theorem (RRT) holds for our discrete Hilbert space because it holds for all Hilbert spaces. In fact, it is perhaps usefully illustrative to apply the RRT to our discrete space as follows – note that our Hilbert space is $\mathbb{R}^N$ with the weighted X-inner product.

Theorem 2.7. ***Riesz representation theorem for $\mathbb{R}^N$ with inner product matrix X.*** *For any linear function $c : \mathbb{R}^N \to \mathbb{R}$, there exists an element $\hat{\mathbf{c}} \in \mathbb{R}^N$ satisfying*

$$c(\mathbf{s}) = \langle \mathbf{s}, \hat{\mathbf{c}}\rangle_{\mathbf{X}}, \qquad \forall \mathbf{s} \in \mathbb{R}^N, \tag{2.59}$$

and the dual norm of c is given by

$$\|c\|_{\mathbf{X}'} \equiv \sup_{\mathbf{s}\in\mathbb{R}^N} \frac{c(\mathbf{s})}{\|\mathbf{s}\|_{\mathbf{X}}} = \|\hat{\mathbf{c}}\|_{\mathbf{X}}. \tag{2.60}$$

Proof. I say this is illustrative because the proof is. Note that all linear functions of vectors in $\mathbb{R}^N$ can be written as

$$c(\mathbf{s}) = \sum_{i=1}^N \mathbf{g}_i\mathbf{s}_i = \mathbf{g}^\top\mathbf{s}. \tag{2.61}$$

for some coefficient vector $\mathbf{g} \in \mathbb{R}^N$. Let $\hat{\mathbf{c}} = \mathbf{X}^{-1}\mathbf{g}$. Then, for any $\mathbf{s} \in \mathbb{R}^N$

$$\langle \mathbf{s}, \hat{\mathbf{c}}\rangle_{\mathbf{X}} = \mathbf{s}^\top\mathbf{X}\hat{\mathbf{c}} = \mathbf{s}^\top\mathbf{X}\mathbf{X}^{-1}\mathbf{g} = c(\mathbf{s}). \tag{2.62}$$

The proof of the second part of the theorem, eq. (2.60), is left as an exercise to the motivated reader. □

2.5.1. Error bounds

The residual of the reduced basis approximation is given by

$$\mathbf{r}(\mu) = \mathbf{f} - \mathbf{A}(\mu)\mathbf{s}_{\mathrm{RB}}(\mu). \tag{2.63}$$

Recall that the finite element solution satisfies $\mathbf{f} - \mathbf{A}(\mu)\mathbf{s}_{\mathrm{FE}} = 0$. Note then that

$$\mathbf{A}(\mu)\mathbf{e}_{\mathrm{RB}}(\mu) = \mathbf{r}(\mu). \tag{2.64}$$

This is the error-residual relationship in the discrete setting. We are now going to invoke the Riesz representation theorem, noting that $c(\mathbf{s};\mu) = \mathbf{r}(\mu)^\top\mathbf{s}$ is a linear function. Thus, there must exist some $\hat{\mathbf{e}}(\mu) \in \mathbb{R}^N$ satisfying:

$$\mathbf{e}_{\mathrm{RB}}(\mu)^\top\mathbf{A}(\mu)^\top\mathbf{s} = \mathbf{r}(\mu)^\top\mathbf{s} = \hat{\mathbf{e}}(\mu)^\top\mathbf{s}(\mu), \qquad \forall \mathbf{s} \in \mathbb{R}^N. \tag{2.65}$$

Letting $\mathbf{s} = \mathbf{e}_{\mathrm{RB}}(\mu)$, we obtain (because $\mathbf{A}$ is symmetric):

$$\mathbf{e}_{\mathrm{RB}}(\mu)^\top\mathbf{A}(\mu)\mathbf{e}_{\mathrm{RB}}(\mu) = \hat{\mathbf{e}}(\mu)^\top\mathbf{e}_{\mathrm{RB}}(\mu). \tag{2.66}$$

Invoking Theorem 2.6 to lower-bound the left-hand side of the above, and using Cauchy-Schwarz to upper-bound the right-hand side, we have

$$\alpha_{\mathrm{LB}}(\mu)\|\mathbf{e}_{\mathrm{RB}}(\mu)\|_{\mathbf{X}}^2 \leq \|\hat{\mathbf{e}}(\mu)\|_{\mathbf{X}}\|\mathbf{e}_{\mathrm{RB}}(\mu)\|_{\mathbf{X}}. \tag{2.67}$$

Re-arranging yields a result analogous to that of Theorem 2.3:

$$\|\mathbf{e}_{\mathrm{RB}}(\mu)\|_{\mathbf{X}} \leq \frac{\|\hat{\mathbf{e}}(\mu)\|_{\mathbf{X}}}{\alpha_{\mathrm{LB}}(\mu)}. \tag{2.68}$$

The following analogues of Theorem 2.4 and Theorem 2.5 are obtained similar to their continuous counterparts:

$$\|\mathbf{e}_{\mathrm{RB}}(\mu)\|_{\mu} \leq \frac{\|\hat{\mathbf{e}}(\mu)\|_{\mathbf{X}}}{\sqrt{\alpha_{\mathrm{LB}}(\mu)}}, \tag{2.69}$$

and

$$|y_{\mathrm{FE}}(\mu) - y_{\mathrm{RB}}(\mu)| \leq \frac{\|\hat{\mathbf{e}}(\mu)\|_{\mathbf{X}}^2}{\alpha_{\mathrm{LB}}(\mu)}. \tag{2.70}$$

2.5.2. Efficient computation of the error bounds

We have already provided an efficient way to compute $\alpha_{\mathrm{LB}}(\mu)$ in our definition. Let's talk about $\hat{\mathbf{e}}(\mu)$, the Riesz representer of the function $c(\mathbf{s};\mu) = \mathbf{r}(\mu)^\top\mathbf{s}$. From the proof of Theorem 2.7, we can easily see that

$$\hat{\mathbf{e}}(\mu) = \mathbf{X}^{-1}\,\mathbf{r}(\mu). \tag{2.71}$$

Let's look at $\mathbf{r}(\mu)$ more closely. We can expand eq. (2.63) using the affine structure to obtain:

$$\mathbf{r}(\mu) = \mathbf{f} - \sum_{q=1}^{Q}\theta_q(\mu)\mathbf{A}_q\mathbf{U}_r\hat{\mathbf{s}}(\mu). \tag{2.72}$$

We can separate the above expression into a parameter-dependent and parameter-independent part: define $\mathbf{H}$ and $\xi(\mu)$ as

$$\mathbf{H} = \begin{pmatrix} \mathbf{f} & -\mathbf{A}_1\mathbf{U}_r & -\mathbf{A}_2\mathbf{U}_r & \cdots & -\mathbf{A}_Q\mathbf{U}_r \end{pmatrix} \in \mathbb{R}^{N\times(1+rQ)}, \tag{2.73}$$

$$\xi(\mu) = \begin{pmatrix} 1 & \theta_1(\mu)\hat{\mathbf{s}}(\mu)^\top & \theta_2(\mu)\hat{\mathbf{s}}(\mu)^\top & \cdots & \theta_Q(\mu)\hat{\mathbf{s}}(\mu)^\top \end{pmatrix}^\top \in \mathbb{R}^{1+rQ}, \tag{2.74}$$

and note that $\mathbf{r}(\mu) = \mathbf{H}\xi(\mu)$. Using this fact and eq. (2.71), we have

$$\|\hat{\mathbf{e}}(\mu)\|_{\mathbf{X}} = \hat{\mathbf{e}}(\mu)^\top\mathbf{X}\hat{\mathbf{e}}(\mu) = \mathbf{r}(\mu)^\top\mathbf{X}^{-1}\mathbf{X}\mathbf{X}^{-1}\mathbf{r}(\mu) = \xi(\mu)^\top\mathbf{H}^\top\mathbf{X}^{-1}\mathbf{H}\xi(\mu). \tag{2.75}$$

Note that $\xi(\mu)$ depends on the parameter and the coefficients of the reduced basis solution, and must therefore be computed online. However, the cost to compute ξ only requires a reduced-basis solve (eq. (2.28)), and is thus an efficient online computation with cost independent of N. Then, define $\hat{\mathbf{G}} = \mathbf{H}^\top\mathbf{X}^{-1}\mathbf{H}$, and note that $\hat{\mathbf{G}} \in \mathbb{R}^{(1+rQ)\times(1+rQ)}$ depends only on the basis functions, not on the parameter, so it can be pre-computed offline and stored to evaluate

$$\|\hat{\mathbf{e}}(\mu)\|_{\mathbf{X}} = \xi(\mu)^\top\hat{\mathbf{G}}\xi(\mu) \tag{2.76}$$

efficiently during the online phase.

3. Balancing

3.1. Preliminaries

We consider the following standard linear time-invariant (LTI) dynamical system:

$$\dot{\mathbf{x}} = \mathbf{A}\mathbf{x} + \mathbf{B}u(t) \tag{3.1a}$$

$$y = \mathbf{C}\mathbf{x}, \tag{3.1b}$$

with $\mathbf{x} \in \mathbb{R}^N$, $\mathbf{A} \in \mathbb{R}^{N\times N}$, $\mathbf{B} \in \mathbb{R}^N$, and $\mathbf{C}^\top \in \mathbb{R}^N$. Such a system has reachability Gramian $\mathbf{P} \in \mathbb{R}^{N\times N}$ and observability Gramian $\mathbf{Q} \in \mathbb{R}^{N\times N}$ given by

$$\mathbf{P} = \int_0^\infty e^{\mathbf{A}t}\mathbf{B}\mathbf{B}^\top e^{\mathbf{A}^\top t}\,\mathrm{d}t, \qquad \mathbf{Q} = \int_0^\infty e^{\mathbf{A}^\top t}\mathbf{C}^\top\mathbf{C}e^{\mathbf{A}t}\,\mathrm{d}t. \tag{3.2}$$

For LTI systems, the reachability and observability energies are given by the Gramians, respectively, as

$$\|\mathbf{x}\|_{\mathbf{P}^{-1}} = \mathbf{x}^\top\mathbf{P}^{-1}\mathbf{x}, \qquad \|\mathbf{x}\|_{\mathbf{Q}} = \mathbf{x}^\top\mathbf{Q}\mathbf{x}. \tag{3.3}$$

Note that the Gramians solve the following dual Lyapunov equations:

$$\mathbf{A}\mathbf{P} + \mathbf{P}\mathbf{A}^\top = -\mathbf{B}\mathbf{B}^\top, \qquad \mathbf{A}^\top\mathbf{Q} + \mathbf{Q}\mathbf{A} = -\mathbf{C}^\top\mathbf{C}. \tag{3.4}$$

This latter definition allows the Gramians to efficiently computed without having to take the infinite integrals.

Recall that a system is *reachable* if $\mathbf{P}$ has full rank and *observable* if $\mathbf{Q}$ has full rank. A system is *minimal* if and only if it is reachable and observable.

3.2. Balancing

3.2.1. Motivation

Recall that

- the reachability energy of a state is the minimum energy required to steer the system from a zero initial condition to that state, and
- the observability energy of a state is the integrated energy of the output y over all time when the system starts at the state and evolves with no input.

These energy definitions lead us to consider reducing the state dimension in by defining a basis in a way that preserves states that (i) contribute most to the observability energy:

$$\mathbf{u}_1 = \arg\max_{\|\mathbf{x}\|=1} \|\mathbf{x}\|_{\mathbf{Q}}, \qquad \mathbf{u}_i = \arg\max_{\substack{\|\mathbf{x}\|=1 \\ \mathbf{x}\notin\operatorname{span}(\{\mathbf{u}_j\}_{j=1}^{i-1})}} \|\mathbf{x}\|_{\mathbf{Q}}, \quad i = 2,\ldots,r, \tag{3.5}$$

or (ii) require the least energy to reach:

$$\mathbf{u}_1 = \arg\min_{\|\mathbf{x}\|=1} \|\mathbf{x}\|_{\mathbf{P}^{-1}}, \qquad \mathbf{u}_i = \arg \min_{\substack{\|\mathbf{x}\|=1 \\ \mathbf{x}\notin \mathrm{span}(\{\mathbf{u}_j\}_{j=1}^{i-1})}} \|\mathbf{x}\|_{\mathbf{P}^{-1}}, \quad i = 2,\ldots,r. \tag{3.6}$$

However note that the two bases eqs. (3.5) and (3.6) are generally not the same! This motivates the use an objective function that balances considerations of observability and reachability, as follows:

$$\mathbf{u}_1 = \arg\max_{\|\mathbf{x}\|=1} \frac{\|\mathbf{x}\|_{\mathbf{Q}}}{\|\mathbf{x}\|_{\mathbf{P}^{-1}}}, \qquad \mathbf{u}_i = \arg \max_{\substack{\|\mathbf{x}\|=1 \\ \mathbf{x}\notin \mathrm{span}(\{\mathbf{u}_j\}_{j=1}^{i-1})}} \frac{\|\mathbf{x}\|_{\mathbf{Q}}}{\|\mathbf{x}\|_{\mathbf{P}^{-1}}}, \quad i = 2,\ldots,r. \tag{3.7}$$

The quantity

$$\frac{\|\mathbf{x}\|_{\mathbf{Q}}}{\|\mathbf{x}\|_{\mathbf{P}^{-1}}} = \frac{\mathbf{x}^\top \mathbf{Q}\mathbf{x}}{\mathbf{x}^\top \mathbf{P}^{-1}\mathbf{x}} \tag{3.8}$$

is called a generalized *Rayleigh quotient*. The directions which maximize this Rayleigh quotient, given in eq. (3.7), are the dominant eigenvectors of $\mathbf{PQ}$. That is, they satisfy

$$\mathbf{PQ}\mathbf{u}_i = \delta_i^2 \mathbf{u}_i, \qquad i = 1,2,\ldots,N, \tag{3.9}$$

where $\delta_i > 0$ are the Hankel singular values. Equivalently, $(\mathbf{u}_i, \delta_i^2)$ are the generalized eigenvector-eigenvalues pairs of the pencil $(\mathbf{Q}, \mathbf{P}^{-1})$, i.e.,

$$\mathbf{Q}\mathbf{u}_i = \delta_i^2 \mathbf{P}^{-1}\mathbf{u}_i. \tag{3.10}$$

We can think of the generalized eigenvalue problem eq. (3.10) as looking for directions in which the action of $\mathbf{Q}$ is the same (up to a multiplicative constant) as the action of $\mathbf{P}^{-1}$ on that direction (compare to typical eigenvalue problems where we look for directions where the action of a matrix on a vector is the vector itself up to a multiplicative constant).

3.2.2. The balancing transformation

The balancing transformation refers to a coordinate transformation of $\mathbf{x}$ into the coordinates defined by the generalized eigenvectors of $(\mathbf{P}, \mathbf{Q}^{-1})$. This transformation provides a way to compute the balanced basis.

To start, note that the Gramians $\mathbf{P}, \mathbf{Q}$ are basis-dependent. That is, if we apply an invertible transformation matrix $\mathbf{T} \in \mathbb{R}^{N\times N}$ to obtain a transformed state $\tilde{\mathbf{x}} = \mathbf{T}\mathbf{x}$, we obtain the following transformed dynamical system:

$$\dot{\tilde{\mathbf{x}}} = \mathbf{TAT}^{-1}\tilde{\mathbf{x}} + \mathbf{TB}u(t) = \tilde{\mathbf{A}}\tilde{\mathbf{x}} + \tilde{\mathbf{B}}u(t) \tag{3.11a}$$

$$y = \mathbf{CT}^{-1}\tilde{\mathbf{x}} = \tilde{\mathbf{C}}\tilde{\mathbf{x}}, \tag{3.11b}$$

where the transformed system matrices are given by $\tilde{\mathbf{A}} = \mathbf{TAT}^{-1}$, $\tilde{\mathbf{B}} = \mathbf{TB}$, and $\tilde{\mathbf{C}} = \mathbf{CT}^{-1}$. Note that the input-output map is invariant to this kind of similarity transformation.[1]

Lemma 3.1. *The reachability and observability Gramians of the transformed system eq.* (3.11)*, denoted* $\tilde{\mathbf{P}}$ *and* $\tilde{\mathbf{Q}}$*, respectively, are given in terms of the Gramians of the original system eq.* (3.1) *as follows:*

$$\tilde{\mathbf{P}} = \mathbf{TPT}^\top, \qquad \tilde{\mathbf{Q}} = \mathbf{T}^{-\top}\mathbf{QT}^{-1}. \tag{3.12}$$

[1]If you decide to consult other sources, you will find that some of them flip the definitions of $\mathbf{T}$ and its inverse, so tread carefully.

Proof. Note that the transformed Gramians satisfy

$$\tilde{\mathbf{A}}\tilde{\mathbf{P}} + \tilde{\mathbf{P}}\tilde{\mathbf{A}}^\top = -\tilde{\mathbf{B}}\tilde{\mathbf{B}}^\top, \qquad \tilde{\mathbf{A}}^\top\tilde{\mathbf{Q}} + \tilde{\mathbf{Q}}\tilde{\mathbf{A}} = -\tilde{\mathbf{C}}^\top\tilde{\mathbf{C}}. \tag{3.13}$$

Expanding eq. (3.13) using the definitions of the transformed system matrices yields

$$\mathbf{TAT}^{-1}\tilde{\mathbf{P}} + \tilde{\mathbf{P}}\mathbf{T}^{-\top}\mathbf{A}^\top\mathbf{T}^\top = -\mathbf{TBB}^\top\mathbf{T}^\top, \tag{3.14}$$
$$\mathbf{T}^{-\top}\mathbf{A}^\top\mathbf{T}^\top\tilde{\mathbf{Q}} + \tilde{\mathbf{Q}}\mathbf{TAT}^{-1} = -\mathbf{T}^{-\top}\mathbf{C}^\top\mathbf{CT}^{-1}. \tag{3.15}$$

Using eq. (3.4) in eq. (3.14), we have

$$\mathbf{TAT}^{-1}\tilde{\mathbf{P}} + \tilde{\mathbf{P}}\mathbf{T}^{-\top}\mathbf{A}^\top\mathbf{T}^\top = \mathbf{T}(\mathbf{AP} + \mathbf{PA}^\top)\mathbf{T}^\top = \mathbf{TAPT}^\top + \mathbf{TPA}^\top\mathbf{T}^\top, \tag{3.16}$$

which is true if $\mathbf{T}^{-1}\tilde{\mathbf{P}} = \mathbf{PT}^\top$, or equivalently if $\tilde{\mathbf{P}} = \mathbf{TPT}^\top$. Similarly, using eq. (3.4) in eq. (3.15), we have

$$\mathbf{T}^{-\top}\mathbf{A}^\top\mathbf{T}^\top\tilde{\mathbf{Q}} + \tilde{\mathbf{Q}}\mathbf{TAT}^{-1} = \mathbf{T}^{-\top}(\mathbf{A}^\top\mathbf{Q} + \mathbf{QA})\mathbf{T}^{-1}, \tag{3.17}$$

which is true if $\mathbf{T}^\top\tilde{\mathbf{Q}} = \mathbf{QT}^{-1}$, or equivalently if $\tilde{\mathbf{Q}} = \mathbf{T}^{-\top}\mathbf{QT}^{-1}$. □

Definition 3.1. *The realization* $(\mathbf{A}, \mathbf{B}, \mathbf{C})$ *of an asymptotically stable system is a* balanced realization *if it has reachability and observability Gramians satisfying* $\mathbf{P} = \mathbf{Q} = \mathrm{diag}(\delta_1, ..., \delta_N)$.

Theorem 3.1. *Let* $(\mathbf{A}, \mathbf{B}, \mathbf{C})$ *be asymptotically stable, controllable and observable. Let* $\mathbf{P} = \mathbf{RR}^\top$, $\mathbf{Q} = \mathbf{LL}^\top$ *be the Cholesky factorizations of the Gramians, and let* $\mathbf{L}^\top\mathbf{R} = \mathbf{U\Sigma V}^\top$ *be the singular value decomposition of the product* $\mathbf{L}^\top\mathbf{R}$. *Then,* $\mathbf{T}$ *given by*

$$\mathbf{T} = \mathbf{\Sigma}^{-\frac{1}{2}}\mathbf{U}^\top\mathbf{L}^\top \qquad \mathbf{T}^{-1} = \mathbf{RV\Sigma}^{-\frac{1}{2}} \tag{3.18}$$

is a balancing transformation, *i.e., it is invertible and the system* $(\widetilde{\mathbf{A}}, \widetilde{\mathbf{B}}, \widetilde{\mathbf{C}})$ *given by eq.* (3.11) *is balanced, i.e., the Gramians of the transformed system satisfy* $\widetilde{\mathbf{P}} = \widetilde{\mathbf{Q}} = \mathrm{diag}(\delta_1, \ldots, \delta_N)$.

Proof. In-class exercise: First note that

$$\mathbf{T}^{-1}\mathbf{T} = \mathbf{RV\Sigma}^{-\frac{1}{2}}\mathbf{\Sigma}^{-\frac{1}{2}}\mathbf{U}^\top\mathbf{L}^\top = \mathbf{RV\Sigma}^{-1}\mathbf{U}^\top\mathbf{L}^\top = \mathbf{R}(\mathbf{L}^\top\mathbf{R})^{-1}\mathbf{L}^\top = \mathbf{RR}^{-1}\mathbf{L}^{-\top}\mathbf{L}^\top = I, \tag{3.19}$$
$$\mathbf{TT}^{-1} = \mathbf{\Sigma}^{-\frac{1}{2}}\mathbf{U}^\top\mathbf{L}^\top\mathbf{RV\Sigma}^{-\frac{1}{2}} = \mathbf{\Sigma}^{-\frac{1}{2}}\mathbf{U}^\top\mathbf{U\Sigma V}^\top\mathbf{V\Sigma}^{-\frac{1}{2}} = \mathbf{\Sigma}^{-\frac{1}{2}}\mathbf{\Sigma\Sigma}^{-\frac{1}{2}} = I, \tag{3.20}$$

so $\mathbf{T}^{-1}$ is indeed the inverse of $\mathbf{T}$ as we have claimed. Now, using the definition of $\tilde{\mathbf{P}}$, we have,

$$\begin{aligned}\tilde{\mathbf{P}} = \mathbf{TPT}^\top &= \mathbf{\Sigma}^{-\frac{1}{2}}\mathbf{U}^\top\mathbf{L}^\top\mathbf{RR}^\top\mathbf{LU\Sigma}^{-\frac{1}{2}} = \mathbf{\Sigma}^{-\frac{1}{2}}\mathbf{U}^\top\mathbf{U\Sigma V}^\top\mathbf{V\Sigma U}^\top\mathbf{U\Sigma}^{-\frac{1}{2}} \\ &= \mathbf{\Sigma}^{-\frac{1}{2}}\mathbf{\Sigma}^2\mathbf{\Sigma}^{-\frac{1}{2}} = \mathbf{\Sigma},\end{aligned} \tag{3.21}$$

and similarly, for $\tilde{\mathbf{Q}}$ we have

$$\begin{aligned}\tilde{\mathbf{Q}} = \mathbf{T}^{-\top}\mathbf{QT}^{-1} &= \mathbf{\Sigma}^{-\frac{1}{2}}\mathbf{V}^\top\mathbf{R}^\top\mathbf{LL}^\top\mathbf{RV\Sigma}^{-\frac{1}{2}} = \mathbf{\Sigma}^{-\frac{1}{2}}\mathbf{V}^\top\mathbf{V\Sigma U}^\top\mathbf{U\Sigma V}^\top\mathbf{V\Sigma}^{-\frac{1}{2}} \\ &= \mathbf{\Sigma}^{-\frac{1}{2}}\mathbf{\Sigma}^2\mathbf{\Sigma}^{-\frac{1}{2}} = \mathbf{\Sigma}.\end{aligned} \tag{3.22}$$

Let $\mathbf{e}_j$ denote the unit vector where $e_i = 1$, for $i = j$, and $e_i = 0$ otherwise. Note that because $\tilde{\mathbf{Q}}, \tilde{\mathbf{P}}$ are diagonal, we have $\tilde{\mathbf{Q}}\mathbf{e}_j = \sigma_j\mathbf{e}_j = \sigma_j^2\tilde{\mathbf{P}}^{-1}\mathbf{e}_j$, so

$$\mathbf{T}^{-\top}\mathbf{QT}^{-1}\mathbf{e}_j = \sigma_j^2\mathbf{T}^{-\top}\mathbf{P}^{-1}\mathbf{T}^{-1}\mathbf{e}_j, \tag{3.23}$$
$$\mathbf{QT}^{-1}\mathbf{e}_j = \sigma_j^2\mathbf{P}^{-1}\mathbf{T}^{-1}\mathbf{e}_j. \tag{3.24}$$

That is, note that $(\sigma_j^2, \mathbf{u}_j)$ where $\mathbf{u}_j = \mathbf{T}^{-1}\mathbf{e}_j$ are the generalized eigenpairs of eq. (3.10) and thus $\sigma_j = \delta_j$, and the generalized eigenvector $\mathbf{u}_j$ is the j-th column of $\mathbf{T}^{-1}$. □

Note that the last part of the proof is essentially showing that generalized eigenvectors of the transformed Gramians $(\tilde{\mathbf{Q}}, \tilde{\mathbf{P}}^{-1})$ satisfy

$$\tilde{\mathbf{Q}}\tilde{\mathbf{u}}_i = \delta_i^2 \tilde{\mathbf{P}}^{-1}\tilde{\mathbf{u}}_i, \tag{3.25}$$

which, since $\tilde{\mathbf{Q}} = \tilde{\mathbf{P}} = \mathrm{diag}(\delta_1, \ldots, \delta_N)$, is satisfied for any unit vector that is all 0s except for one entry that is a 1. Thus, this transformation does indeed move us into the coordinate system defined by the generalized eigenvectors of eq. (3.10).

3.3. Model reduction via balanced truncation

3.3.1. Method

Balanced truncation does exactly what it sounds like, which is to say it transforms to a balanced realization and then truncates to the upper-left blocks of the system. Given a balancing transformation $\mathbf{T}$, let us write our transformed system matrices (used in eq. (3.11)) in blocks:

$$\tilde{\mathbf{A}} = \begin{pmatrix} \tilde{\mathbf{A}}_{11} & \tilde{\mathbf{A}}_{12} \\ \tilde{\mathbf{A}}_{21} & \tilde{\mathbf{A}}_{22} \end{pmatrix}, \qquad \tilde{\mathbf{B}} = \begin{pmatrix} \tilde{\mathbf{B}}_1 \\ \tilde{\mathbf{B}}_2 \end{pmatrix}, \qquad \tilde{\mathbf{C}} = \begin{pmatrix} \tilde{\mathbf{C}}_1 & \tilde{\mathbf{C}}_2 \end{pmatrix}, \tag{3.26}$$

where the blocks of $\mathbf{A}$ have sizes $\tilde{\mathbf{A}}_{11} \in \mathbb{R}^{r\times r}$, $\tilde{\mathbf{A}}_{12} \in \mathbb{R}^{r\times(N-r)}$, $\tilde{\mathbf{A}}_{21} \in \mathbb{R}^{(N-r)\times r}$, $\tilde{\mathbf{A}}_{22} \in \mathbb{R}^{(N-r)\times(N-r)}$, and the blocks of $\tilde{\mathbf{B}}$ and $\tilde{\mathbf{C}}$ have sizes $\tilde{\mathbf{B}}_1, \tilde{\mathbf{C}}_1^\top \in \mathbb{R}^r$ and $\tilde{\mathbf{B}}_2, \tilde{\mathbf{C}}_2^\top \in \mathbb{R}^{(N-r)}$. Then, the balanced truncation reduced model is given by taking

$$\hat{\mathbf{A}} = \tilde{\mathbf{A}}_{11}, \quad \hat{\mathbf{B}} = \tilde{\mathbf{B}}_1, \quad \hat{\mathbf{C}} = \tilde{\mathbf{C}}_1, \tag{3.27}$$

so we have a reduced LTI system:

$$\dot{\hat{\mathbf{x}}} = \hat{\mathbf{A}}\hat{\mathbf{x}} + \hat{\mathbf{B}}u(t), \tag{3.28a}$$

$$\hat{y} = \hat{\mathbf{C}}\hat{\mathbf{x}}. \tag{3.28b}$$

Alternatively, we can get the reduced matrices directly by letting $\mathbf{U}_r, \mathbf{V}_r$, respectively, denote the first r columns of $\mathbf{U}, \mathbf{V}$ respectively, and let $\mathbf{\Sigma}_r$ denote the upper-left r-by-r block of $\mathbf{\Sigma}$. Then, let

$$\tilde{\mathbf{W}}_r^\top = \mathbf{\Sigma}_r^{-\frac{1}{2}}\mathbf{U}_r^\top\mathbf{L}^\top, \qquad \mathbf{W}_r = \mathbf{R}\mathbf{V}_r\mathbf{\Sigma}_r^{-\frac{1}{2}}. \tag{3.29}$$

Note that $\tilde{\mathbf{W}}_r^\top$ contains the first r rows of $\mathbf{T}$ and $\mathbf{W}_r$ has the first r columns of $\mathbf{T}^{-1}$. That is, $\tilde{\mathbf{W}}_r^\top$ converts a state $\mathbf{x}$ to a balanced truncated state $\hat{\mathbf{x}}$, and $\mathbf{W}_r$ reverses that. Then,

$$\hat{\mathbf{A}} = \tilde{\mathbf{W}}_r^\top\mathbf{A}\mathbf{W}_r, \qquad \hat{\mathbf{B}} = \tilde{\mathbf{W}}_r^\top\mathbf{B}, \qquad \hat{\mathbf{C}} = \mathbf{C}\mathbf{W}_r. \tag{3.30}$$

Note that this is an example of an *oblique* (Petrov-Galerkin) projection.

3.4. Properties

Theorem 3.2. *Let $(\mathbf{A}, \mathbf{B}, \mathbf{C})$ be a stable and minimal system, and suppose $(\hat{\mathbf{A}}, \hat{\mathbf{B}}, \hat{\mathbf{C}})$ is a reduced model obtained by balancing and truncating for some r satisfying $\delta_r > \delta_{r+1}$. Then, the reduced model is asymptotically stable, minimal and balanced with Gramians $\hat{\mathbf{P}} = \hat{\mathbf{Q}} = \mathrm{diag}(\delta_1, \ldots, \delta_r) =: \mathbf{\Sigma}_r$.*

Proof. The proof of preserving stability is somewhat technical and can be found in Antoulas.[2] Assuming that asymptotic stability is preserved, the proof about the balanced reduced model having balanced Gramians is more enlightening, so I might put it on the homework.

□

Theorem 3.3. *Let* $(\mathbf{A}, \mathbf{B}, \mathbf{C})$ *be asymptotically stable and balanced with reachability Gramian* $\mathbf{P}$ *and observability Gramian* $\mathbf{Q}$ *and let* $G(s) = \mathbf{C}(sI - \mathbf{A})^1\mathbf{B}$ *be the transfer function. Let the singular values be ordered,* $\delta_r > \delta_{r+1} = \ldots = \delta_n$. *Let* $(\hat{\mathbf{A}}, \hat{\mathbf{B}}, \hat{\mathbf{C}})$ *be the r-th order reduced model obtained with BT, with transfer function* $\hat{G}(s) = \hat{\mathbf{C}}(sI - \hat{\mathbf{A}})^{-1}\hat{\mathbf{B}}$. *Then, we have:*

$$\|G - \hat{G}\|_{\mathcal{H}_\infty} \leqslant 2\delta_{r+1}, \tag{3.31}$$

independent of the multiplicity of δ_{r+1}.

Corollary 3.1. *Let* $(\mathbf{A}, \mathbf{B}, \mathbf{C})$ *be asymptotically stable and balanced with reachability Gramian and observability Gramian* $\mathbf{P} = \mathbf{Q} = \operatorname{diag}(\delta_1 I_{s1}, \delta_2 I_{s2}, \ldots, \delta_r I_{sr})$*(*$\delta$ *could be repeated), where* $\delta_1 > \delta_2 > \ldots > \delta_r \geqslant 0$. *Let* $(\hat{\mathbf{A}}, \hat{\mathbf{B}}, \hat{\mathbf{C}})$ *be the balanced reduced model,* $r = s_1 + s_2 + \ldots + s_l$ *for some* $l \geqslant r$. *Then, we have:*

$$\|G - \hat{G}\|_{\mathcal{H}_\infty} \leqslant \sum_{j=l+1}^{k} 2\delta_j. \tag{3.32}$$

3.5. Balancing extensions

3.5.1. Goal-oriented dimension reduction for linear inverse problems

The inference-for-prediction (IFP) method[3] extends the idea of balanced truncation to create a goal-oriented model reduction framework in an inference setting. We summarize that approach here, highlighting its connections to balanced truncation.

Problem formulation and motivation Let $\mu \in \mathbb{R}^p$ be a parameter that defines a system of interest, and let $\mathbf{O}_m \in \mathbb{R}^{q\times p}$ be a linear experiment measurement operator so that measurements $m \in \mathbb{R}^q$ are obtained as follows:

$$m(\mu) = \mathbf{O}_m\mu. \tag{3.33}$$

For example, $\mathbf{O}_m$ could be the composition of a PDE operator and an observation operator – i.e., solve the PDE at μ and then measure at sensor locations. Usually, we assume p is large, and that $p \gg q$, for example, if μ represents the discretization of a spatially varying parameter *field* and we can only measure a few different locations in space.

Given a value of μ, the *forward problem* is solve for $m(\mu)$. The *inverse problem* is to solve for μ given measurements m_{data}. The inverse problem is usually an ill-posed problem, because there are many more parameter values q to infer than we have measurements p. Traditional approaches to solving the inverse problem usually solve a regularized least-squares minimization:

$$\mu^* = \arg\min_{\mu\in\mathbb{R}^p} \|m_{\text{data}} - \mathbf{O}_m\mu\|^2 + \|\Gamma\mu\|^2, \tag{3.34}$$

[2]Antoulas, A.C., 2005. Approximation of large-scale dynamical systems. Society for Industrial and Applied Mathematics.

[3]Lieberman, C. and Willcox, K. Goal-oriented inference: Approach, linear theory, and application to advection-diffusion , SIAM Review , Vol. 55, No. 3, pp. 493-519, 2013. (SIGEST award)

where $\Gamma \in \mathbb{R}^{p\times p}$ is some Tikhonov regularization weighting. When the forward operator $\mathbf{O}_m$ involves the solution of a PDE, eq. (3.34) is a *PDE-constrained optimization.* PDE-constrained optimizations can be quite costly to solve because every step of a gradient-based solver requires solution of the PDE, and the number of required steps explodes as the dimension p increases (the curse of dimensionality). Thus, we would like to reduce the dimension of eq. (3.34).

Dimension reduction Rather than searching all of $\mathbb{R}^p$ for the right μ in eq. (3.34), we would like to search a low-dimensional subspace of $\mathbb{R}^p$. But what is the right subspace? The IFP method makes a balancing-analogous choice by observing that usually it is not μ itself that is of interest but some prediction based on μ:

$$y(\mu) = \mathbf{O}_p\mu, \tag{3.35}$$

where $\mathbf{O}_p$ is a prediction operator. For example, in a carbon capture application, we may attempt to infer the underground porosity and peremeability fields, and use these to calculate the CO_2 leakage rate. The dimension of the prediction is typically also much less than p, let's assume $y(\mu) \in \mathbb{R}^q$ (same dimension as the measurements) for simplicity.

Thus, the IFP method finds the dominant eigenspaces of the measurement Gramian $\mathbf{O}_m^\top\mathbf{O}_m$ and the prediction Gramian $\mathbf{O}_p^\top\mathbf{O}_p$.

3.5.2. Low-rank posterior updates in Bayesian inverse problems

We now present the work of Spantini et al.[4] on low rank Bayesian posterior updates.

Linear Bayesian inverse problem A standard linear Bayesian inference formulation is given by the following measurement model

$$\mathbf{m} = \mathbf{G}\mathbf{x} + \epsilon, \tag{3.36}$$

where $\mathbf{G} \in \mathbb{R}^{m\times n}$ is the linear forward operator mapping the state to outputs, and $\epsilon \sim \mathcal{N}(0, \mathbf{\Gamma}_{\text{obs}})$ represents additive Gaußian noise, where $\mathbf{\Gamma}_{\text{obs}} \in \mathbb{R}^{m\times m}$.

The goal of Bayesian inference is to infer $\mathbf{x}$ from $\mathbf{m}$. We take a Gaußian prior,

$$\mathbf{x} \sim \mathcal{N}(\mu_{\text{prior}}, \mathbf{\Gamma}_{\text{prior}}) \tag{3.37}$$

with mean $\mu_{\text{prior}} \in \mathbb{R}^n$ and covariance $\mathbf{\Gamma}_{\text{prior}} \in \mathbb{R}^{n\times n}$. The Gaußian prior, combined with our linear Gaußian measurement model (eq. (3.36)), yields the following Gaußian likelihood:

$$\mathbf{m} \mid \mathbf{x} \sim \mathcal{N}(\mathbf{G}\mathbf{x}, \mathbf{\Gamma}_{\text{obs}}). \tag{3.38}$$

where $\mathbf{m} \in \mathbb{R}^n$, $\mathbf{G} \in \mathbb{R}^{n\times n}$, and $\mathbf{\Gamma}_{\text{obs}} \in \mathbb{R}^{n\times n}$ are given as follows: Then, the Bayesian posterior distribution for such a prior and likelihood is again Gaußian:

$$\mathbf{x} \mid \mathbf{m} \sim \mathcal{N}(\mu_{\text{pos}}, \mathbf{\Gamma}_{\text{pos}}), \tag{3.39}$$

where

$$\mu_{\text{pos}} = \mathbf{\Gamma}_{\text{pos}}\mathbf{G}^\top\mathbf{\Gamma}_{\text{obs}}^{-1}\mathbf{m}, \qquad \mathbf{\Gamma}_{\text{pos}} = \left(\mathbf{H} + \mathbf{\Gamma}_{\text{prior}}^{-1}\right)^{-1}, \tag{3.40}$$

where $\mathbf{H} = \mathbf{G}^\top\mathbf{\Gamma}_{\text{obs}}^{-1}\mathbf{G} \in \mathbb{R}^{n\times n}$ is called the Fisher information matrix.

[4] Alessio Spantini, Antti Solonen, Tiangang Cui, James Martin, Luis Tenorio, and Youssef Marzouk, *Optimal Low-rank Approximations of Bayesian Linear Inverse Problems*, SIAM Journal on Scientific Computing 37 (2015), no. 6, A2451–A2487.

Posterior approximation by low-rank updates to the prior Spantini considers the approximation of $\mathbf{\Gamma}_{\text{pos}}$ within the following class of rank-r negative semidefinite updates to $\mathbf{\Gamma}_{\text{prior}}$:

$$\mathcal{M}_r =: \left\{ \mathbf{M} \in \mathbb{R}^{n\times n} \quad \middle| \quad \mathbf{M} = \mathbf{\Gamma}_{\text{prior}} - KK^T \succ 0 \;\text{ with }\; \mathsf{rank}(K) \leq r \right\}. \tag{3.41}$$

The distance between the true posterior covariance and an approximation within $\mathcal{M}_r$ is measured by the Förstner metric between symmetric positive semidefinite matrices $A, B \in \mathbb{R}^{n\times n}$:

$$d_{\mathcal{F}}(A,B) = \sum_{i=1}^{d} \ln^2(\sigma_i), \tag{3.42}$$

where σ_i are the generalized eigenvalues of the pencil (A, B). The Förstner distance is invariant to similarity transformation, and $d_{\mathcal{F}}(A,B) = d_{\mathcal{F}}(A^{-1}, B^{-1})$. The optimal approximation $\hat{\mathbf{\Gamma}}_{\text{pos}}$ to $\mathbf{\Gamma}_{\text{pos}}$ within the class $\mathcal{M}_r$,

$$\hat{\mathbf{\Gamma}}_{\text{pos}} = \arg \min_{M\in\mathcal{M}_r} d_{\mathcal{F}}\left(\mathbf{\Gamma}_{\text{pos}}, M\right) \tag{3.43}$$

is then given by Spantini as:

$$\hat{\mathbf{\Gamma}}_{\text{pos}} = \mathbf{\Gamma}_{\text{prior}} - K_* K_*^\top \quad \text{for} \quad K_* K_*^\top = \sum_{i=1}^{r} \frac{\delta_i^2}{1+\delta_i^2} v_i v_i^\top, \tag{3.44}$$

where (δ_i^2, v_i) are the generalized eigenvalue-eigenvector pairs of the pencil $(\mathbf{H}, \mathbf{\Gamma}_{\text{prior}}^{-1})$.

4. Interpolatory methods

4.1. Set-up

We consider the following single-input single-output (SISO) linear dynamical system:

$$\mathbf{E}\dot{\mathbf{x}} = \mathbf{A}\mathbf{x} + \mathbf{b}u(t) \tag{4.1a}$$

$$y = \mathbf{c}\mathbf{x}, \tag{4.1b}$$

where $\mathbf{x} \in \mathbb{R}^N$, $\mathbf{E}, \mathbf{A} \in \mathbb{R}^{N\times N}$, and $\mathbf{b}, \mathbf{c}^\top \in \mathbb{R}^N$. We assume $\mathbf{E}$ is non-singular, so that eq. (4.1a) is a system of ODEs, not DAEs. This system has the following scalar transfer function:

$$G(s) = \mathbf{c}(s\mathbf{E} - \mathbf{A})^{-1}\mathbf{b}, \tag{4.2}$$

which is a *rational* function, i.e., we can write $G(s) = \frac{p(s)}{q(s)}$, where p and q are polynomial in s.
Model reduction via rational interpolation seeks to identify a reduced model:

$$\hat{\mathbf{E}}\dot{\hat{\mathbf{x}}} = \hat{\mathbf{A}}\hat{\mathbf{x}} + \hat{\mathbf{b}}u(t) \tag{4.3a}$$

$$\hat{y} = \hat{\mathbf{c}}\hat{\mathbf{x}}, \tag{4.3b}$$

where $\hat{\mathbf{x}} \in \mathbb{R}^r$, $\hat{\mathbf{E}}, \hat{\mathbf{A}} \in \mathbb{R}^{r\times r}$, and $\hat{\mathbf{b}}, \hat{\mathbf{c}}^\top \in \mathbb{R}^r$, such that the reduced transfer function from u to $\hat{y}$, given by

$$\hat{G}(s) = \hat{\mathbf{c}}(s\hat{\mathbf{E}} - \hat{\mathbf{A}})^{-1}\hat{\mathbf{b}}, \tag{4.4}$$

interpolates the full transfer function $G(s)$ at selected values of s.

4.2. Model reduction via interpolation

Let $\{\sigma_i\}_{i=1}^r \in \mathbb{C}$ and $\{\mu_i\}_{i=1}^r \in \mathbb{C}$ be selected interpolation frequencies. That is, we want $G(\sigma_i) = \hat{G}(\sigma_i)$ for all i, and $G(\mu_i) = \hat{G}(\mu_i)$ for all i. This means that the reduced model has the same response to inputs with frequencies given by σ_i, μ_i. We define basis vectors as follows:

$$\mathbf{v}_i = (\sigma_i\mathbf{E} - \mathbf{A})^{-1}\mathbf{b}, \qquad \mathbf{w}_i = (\mu_i\mathbf{E} - \mathbf{A})^{-\top}\mathbf{c}^\top, \tag{4.5}$$

and use these vectors to define left and right bases

$$\mathbf{V}_r = \begin{pmatrix}\mathbf{v}_1 & \cdots & \mathbf{v}_r\end{pmatrix}, \qquad \mathbf{W}_r = \begin{pmatrix}\mathbf{w}_1 & \cdots & \mathbf{w}_r\end{pmatrix}. \tag{4.6}$$

Then, we define the reduced model operators as follows:

$$\hat{\mathbf{E}} = \mathbf{W}_r^\top\mathbf{E}\mathbf{V}_r, \quad \hat{\mathbf{A}} = \mathbf{W}_r^\top\mathbf{A}\mathbf{V}_r, \quad \hat{\mathbf{b}} = \mathbf{W}_r^\top\mathbf{b}, \quad \hat{\mathbf{c}} = \mathbf{c}\mathbf{V}_r. \tag{4.7}$$

Theorem 4.1. *Given $\{\sigma_i\}_{i=1}^r \in \mathbb{C}$ and $\{\mu_i\}_{i=1}^r \in \mathbb{C}$, let the reduced model be defined as in eqs.* (4.5) *to* (4.7)*. Then,*

$$G(\sigma_i) = \hat{G}(\sigma_i), \qquad i = 1, 2, \dots, r, \tag{4.8}$$

$$G(\mu_i) = \hat{G}(\mu_i), \qquad i = 1, 2, \dots, r, \tag{4.9}$$

$$G'(\sigma_j) = \hat{G}(\sigma_j), \qquad \textit{if } \sigma_j = \mu_j. \tag{4.10}$$

Proof. I'll just prove the first to interpolation conditions for the transfer functions themselves. The proof of Hermite interpolation is a bit messier and can be found in Antoulas. Essentially, we define two projections $\mathbf{P}(s)$ and $\mathbf{Q}(s)$ as follows:

$$\mathbf{P}(s) = \mathbf{V}_r(s\hat{\mathbf{E}} - \hat{\mathbf{A}})^{-1}\mathbf{W}_r(s\mathbf{E} - \mathbf{A}) \tag{4.11}$$

$$\mathbf{Q}(s) = (s\mathbf{E} - \mathbf{A})\mathbf{P}(s)(s\mathbf{E} - \mathbf{A})^{-1} = (s\mathbf{E} - \mathbf{A})\mathbf{V}_r(s\hat{\mathbf{E}} - \hat{\mathbf{A}})^{-1}\mathbf{W}_r^\top. \tag{4.12}$$

It is easy to verify that $\mathbf{P}(s)\mathbf{P}(s) = \mathbf{P}(s)$ and $\mathbf{Q}(s)\mathbf{Q}(s) = \mathbf{Q}(s)$, so these are indeed projections. Additionally, verify that

$$\mathbf{P}(s)\mathbf{V}_r = \mathbf{V}_r, \qquad \text{and} \qquad \mathbf{W}_r^\top\mathbf{Q} = \mathbf{W}_r^\top. \tag{4.13}$$

Now, observe that the difference in the transfer functions can be written as follows:

$$G(s) - \hat{G}(s) = \mathbf{c}(s\mathbf{E} - \mathbf{A})^{-1}(\mathbf{I} - \mathbf{Q})(s\mathbf{E} - \mathbf{A})(\mathbf{I} - \mathbf{P})(s\mathbf{E} - \mathbf{A})^{-1}\mathbf{b}. \tag{4.14}$$

(The easiest way for me to see this is to write out the right-hand side of the above using the definition of $\mathbf{Q}$ in terms of $\mathbf{P}$, and then cancel/simplify to get $G(s) - \hat{G}(s)$.) Then, note that if we take $s = \sigma_i$ in eq. (4.14), the last two terms are

$$(\mathbf{I} - \mathbf{P})(\sigma_i\mathbf{E} - \mathbf{A})^{-1}\mathbf{b} = (\mathbf{I} - \mathbf{P})\mathbf{v}_i = 0, \tag{4.15}$$

because $\mathbf{v}_i$ is in the nullspace of $(\mathbf{I} - \mathbf{P})$, so $G(\sigma_i) - \hat{G}(\sigma_i) = 0$ and the interpolation condition is satisfied. Similarly, note that if we take $s = \mu_i$, the first two terms in eq. (4.14) are

$$\mathbf{c}(\mu_i\mathbf{E} - \mathbf{A})^{-1}(\mathbf{I} - \mathbf{Q}) = \mathbf{w}_i^\top(\mathbf{I} - \mathbf{Q}) = 0, \tag{4.16}$$

because $\mathbf{w}_i^\top$ is in the left nullspace of $(\mathbf{I} - \mathbf{Q})$, so $G(\mu_i) - \hat{G}(\mu_i) = 0$ and the interpolation condition is satisfied.

To prove the Hermite interpolation condition, we need to do some Taylor expansion. See Antoulas, because it's a bit messy for this first pass at notes. add this proof later □

add 2D example

Corollary 4.1. *Given left and right bases $\mathbf{W}_r$ and $\mathbf{V}_r$, let the reduced model be defined as in eq. (4.7). Then, we have:*

1. *if $(\sigma\mathbf{E} - \mathbf{A})^{-1}\mathbf{b} \in range(\mathbf{V}_r)$, then $G(\sigma) = \hat{G}(\sigma)$,*
2. *if $(\mu\mathbf{E} - \mathbf{A})^{-\top}\mathbf{c}^\top \in range(\mathbf{W}_r)$, the $G(\mu) = \hat{G}(\mu)$,*
3. *if both of the above conditions are true for $\sigma = \mu$, then $G'(\sigma) = \hat{G}'(\sigma)$.*

This corollary tells us that we need not specifically construct the bases out of the basis vectors given by eq. (4.5) – in fact, in general it is a good idea to orthogonalize the basis vectors in eq. (4.5) using a QR algorithm or similar.

We can actually construct left and right bases to interpolate the transfer function at higher-order derivatives as well.

Theorem 4.2. *Let $\mathbf{W}_r$ and $\mathbf{V}_r$ be left and right bases which define a reduced model as in eq. (4.7). Then:*

1. *if $\left((\sigma\mathbf{E} - \mathbf{A})^{-1}\mathbf{E}\right)^{j-1}(\sigma\mathbf{E} - \mathbf{A})^{-1}\mathbf{b} \in range(\mathbf{V}_r)$ for $j = 1, 2, \dots, n$, then*

$$G^{(\ell)}(\sigma) = \hat{G}^{(\ell)}(\sigma), \qquad \ell = 0, 1, \dots, n-1, \tag{4.17}$$

2. *if* $\left((\mu\mathbf{E}-\mathbf{A})^{-\top}\mathbf{E}^\top\right)^{j-1}(\mu\mathbf{E}-\mathbf{A})^{-\top}\mathbf{c}^\top \in$ *range*$(\mathbf{W}_r)$ *for* $j=1,2,\ldots,m$, *then*

$$G^{(\ell)}(\mu)=\hat{G}^{(\ell)}(\mu), \qquad \ell=0,1,\ldots,m-1, \tag{4.18}$$

3. *if* $\sigma=\mu$, *and both of the above conditions hold, then*

$$G^{(\ell)}(\sigma)=\hat{G}^{(\ell)}(\sigma), \qquad \ell=0,1,\ldots,m+n-1. \tag{4.19}$$

4.3. $\mathcal{H}_2$-optimal interpolation

Until now we have assumed that the interpolation frequencies σ_i, μ_i are given to us, and we have not talked about how we choose points. Interpolation can be tricky business – one may recall from the more familiar territory of polynomial interpolation that a polynomial interpolant may match the function of interest very well near the interpolation points but lead to arbitrarily high error far away from the interpolation points. In polynomial interpolation, we have the Chebyshev interpolation points as a good practice. Fortunately, for rational interpolation we also have some well-accepted strategies.

In this section we introduce a strategy from $\mathcal{H}_2$-optimal interpolation. That is, we want to choose our interpolation points σ_i, μ_i such that $\hat{G}$ satisfies

$$\hat{G}=\min_{\dim(G_r)=r}\|G-\hat{G}\|_{\mathcal{H}_2}. \tag{4.20}$$

Recall that $\|y(t)-\hat{y}(t)\|_{L^\infty}\leq\|G-\hat{G}\|_{\mathcal{H}_2}\|u(t)\|_{L^2}$, so by minimizing eq. (4.20) we are seeking the reduced model whose output is closest to the full output in the L^∞ norm. Unfortunately, 4.20 is non-convex, so finding a globally optimum will be computationally expensive. Instead, we will seek interpolation points that lead to *local optimality.*

We derive conditions for local optimality under the assumption that the reduced and full models are both stable and have only simple poles (no repeated poles). The theory applies to the general case, but this assumption will simplify the exposition.

Lemma 4.1. *From [7]: Suppose* $G(s)$ *has simple poles* $\lambda_1,\ldots,\lambda_n$ *in the open left-half plane, and* $H(s)$ *has simple poles* $\mu_1,\ldots,\mu_m$ *in the open left-half plane. Then,* $G(s)$ *and* $H(s)$ *can be written in the following pole-residue expansions:*

$$G(s)=\sum_{i=1}^{n}\frac{g_i}{s-\lambda_i}, \qquad\qquad H(s)=\sum_{i=1}^{m}\frac{h_i}{s-\mu_i}, \tag{4.21}$$

and the $\mathcal{H}_2$*-inner product can be written as*

$$\langle G,H\rangle_{\mathcal{H}_2}\equiv\frac{1}{2\pi}\int_{-\infty}^{\infty}G(-j\omega)H(j\omega)\,d\omega=\sum_{i=1}^{m}G(-\mu_i)h_i=\sum_{i=1}^{n}H(-\lambda_i)g_i. \tag{4.22}$$

Note that this lets us write the squared $\mathcal{H}_2$-norm of a transfer function as

$$\|G\|_{\mathcal{H}_2}^2=\sum_{i=1}^{N}g_iG(-\lambda_i)=\sum_{i,j=1}^{N}\frac{g_ig_j}{-\lambda_i-\lambda_j}, \tag{4.23}$$

and that

$$G'(s) = \sum_{i=1}^{n} -\frac{g_i}{(s-\lambda_i)^2}. \tag{4.24}$$

Thus, we can re-write our objective function

$$\|G-\hat{G}\|_{\mathcal{H}_2}^2 = \left\langle G-\hat{G}, G-\hat{G}\right\rangle_{\mathcal{H}_2} = \langle G, G\rangle_{\mathcal{H}_2} - \left\langle G, \hat{G}\right\rangle_{\mathcal{H}_2} - \left\langle \hat{G}, G\right\rangle_{\mathcal{H}_2} + \left\langle \hat{G}, \hat{G}\right\rangle_{\mathcal{H}_2} \tag{4.25}$$

$$= \|G\|_{\mathcal{H}_2}^2 - 2\sum_{i=1}^{r} G(-\hat{\lambda}_i)\hat{g}_i + \sum_{i,j=1}^{r} \frac{\hat{g}_i\hat{g}_j}{-\hat{\lambda}_i - \hat{\lambda}_j}, \tag{4.26}$$

where $\hat{g}_i, \hat{\lambda}_i$ are the residues and poles of the reduced transfer function. We will now derive first-order optimality conditions in terms of $\hat{g}_i$ and λ_i:

$$\frac{\partial}{\partial \hat{g}_k}\|G-\hat{G}\|_{\mathcal{H}_2}^2 = 0 = -2G(-\hat{\lambda}_k) + \sum_{j=1, j\neq k}^{r} \frac{\hat{g}_j}{-\hat{\lambda}_k - \hat{\lambda}_j} + \sum_{i=1, i\neq k}^{r} \frac{\hat{g}_i}{-\hat{\lambda}_i - \hat{\lambda}_k} + 2\frac{\hat{g}_k}{-\lambda_k - \lambda_k}$$

$$= -2G(-\hat{\lambda}_k) + 2\sum_{i=1}^{r} \frac{\hat{g}_k\hat{g}_k}{-\hat{\lambda}_k - \hat{\lambda}_i} = -2G(-\hat{\lambda}_k) + 2\hat{G}(-\hat{\lambda}_k), \tag{4.27}$$

and

$$\frac{\partial}{\partial \hat{\lambda}_k}\|G-\hat{G}\|_{\mathcal{H}_2}^2 = 0 = -2\hat{g}_k G'(-\hat{\lambda}_k) + \sum_{j=1, j\neq k}^{r} \frac{\hat{g}_k\hat{g}_j}{(-\hat{\lambda}_k - \hat{\lambda}_j)^2} + \sum_{i=1, i\neq k}^{r} \frac{\hat{g}_i\hat{g}_k}{(-\hat{\lambda}_i - \hat{\lambda}_k)^2} + \frac{2\hat{g}_k^2}{(-\lambda_k - \lambda_k)^2}$$

$$= -2\hat{g}_k G'(-\hat{\lambda}_k) + 2\sum_{i=1}^{r} \frac{\hat{g}_k\hat{g}_i}{(-\hat{\lambda}_k - \hat{\lambda}_i)^2} = 2\hat{g}_k G'(-\hat{\lambda}_k) + 2\hat{g}_k\hat{G}'(-\hat{\lambda}_k). \tag{4.28}$$

Thus, the first-order optimality conditions are satisifed if

$$G(-\hat{\lambda}_k) = \hat{G}(-\hat{\lambda}_k) \tag{4.29}$$

$$G'(-\hat{\lambda}_k) = \hat{G}'(-\hat{\lambda}_k), \tag{4.30}$$

or, in other words, if the interpolation conditions are satisfied at the opposites of the poles of the reduced model. So if we know the poles of the reduced model, we can do this easily using Theorem 4.1.

One might recall that first-order optimality conditions do not guarantee that we have a true local minimum – it could be a maximum or a saddle point. See [7] for alternate optimality conditions that give us a local minimum. But these first-order necessary conditions are the basis for the fixed point iteration we present next, and in practice whether we have a minimum appears to be more a theoretical concern than a practical one. More discussion to follow.

4.4. Iterative Rational Krylov Algorithm

We have seen that interpolating at the opposite of the reduced model poles, $-\hat{\lambda}_k$, $k = 1, 2, \ldots, r$, provides necessary conditions for local $\mathcal{H}_2$-optimality of our reduced model. But clearly, the reduced model poles depend on the interpolation points. This leads to the *Iterative Rational Krylov Algorithm* (IRKA).

Algorithm 2 Iterative Rational Krylov Algorithm (IRKA) for SISO systems

```
1: Initalize interpolation frequencies $\{\sigma_i\}_{i=1}^r$ and compute interpolatory reduced operators
   $\hat{\mathbf{E}}, \hat{\mathbf{A}}, \hat{\mathbf{b}}, \hat{\mathbf{c}}$.
2: while (not converged) do
3:     Update interpolation frequencies $\sigma_i \leftarrow -\texttt{eig}(\hat{\mathbf{E}}, \hat{\mathbf{A}})$
4:     Update reduced operators $\hat{\mathbf{E}}, \hat{\mathbf{A}}, \hat{\mathbf{b}}, \hat{\mathbf{c}}$ with new interpolation frequencies
5: end while
6: return reduced operators $\hat{\mathbf{E}}$, $\hat{\mathbf{A}}$, $\hat{\mathbf{b}}$, $\hat{\mathbf{c}}$.
```

IRKA exploits the fact that the r poles of the reduced model are generalized eigenvalues of $(\hat{\mathbf{E}}, \hat{\mathbf{A}})$, i.e., they satisfy $\hat{\lambda}_k \hat{\mathbf{E}}\mathbf{u}_k = \hat{\mathbf{A}}\mathbf{u}_k$. Because the dimension r of the reduced model is small, computing the generalized eigenvalues of the system can be done cheaply.

In its SISO form, IRKA is a fixed point iteration, and its convergence can be understood from that perspective. That is, when it does converge, the rate is at most linear. In practice, for SISO systems and small MIMO systems, IRKA converges fast enough that it has been used very widely for model reduction via rational interpolation. Note that it is possible for IRKA to converge to a maximum or saddle point, but in many cases these are repellent fixed points, whereas local minima are usually attractive fixed points (proved for certain classes of systems). Thus, if IRKA converges to a point we can be reasonably confident that we have found a local minimum.

Various other algorithms for determining the optimal interpolation points have been proposed, including a realization-independent variant of IRKA, descent algorithms, and greedy algorithms.

4.5. Tangential interpolation for MIMO systems

Consider now the multi-input multi-output (MIMO) system

$$\mathbf{E}\dot{\mathbf{x}} = \mathbf{A}\mathbf{x} + \mathbf{B}u(t), \tag{4.31a}$$

$$y = \mathbf{C}\mathbf{x}, \tag{4.31b}$$

where $\mathbf{B} \in \mathbb{R}^{N\times m}$ and $\mathbf{C} \in \mathbb{R}^{p\times N}$. The transfer function for this system is now an $p \times m$ matrix of transfer functions:

$$\mathbf{G}(s) = \mathbf{C}(s\mathbf{E} - \mathbf{A})^{-1}\mathbf{B}. \tag{4.32}$$

How do we interpolate a MIMO system? While matrix interpolation (i.e., interpolating so that the reduced transfer function matches $\mathbf{G}(s)$ at every entry), this can lead to large reduced models because we need a basis function for every input/output/interpolation point. Tangential interpolation is an alternative to matrix interpolation. In tangential interpolation, we let

- $\mu_i \in \mathbb{C}$ be left interpolation points for $i = 1, \dots, r$ with corresponding left tangential directions $\tilde{\mathbf{c}}_i \in \mathbb{C}^p$ for $i = 1, \dots, r$
- $\sigma_i \in \mathbb{C}$ be right interpolation points for $i = 1, \dots, r$ with corresponding right tangential directions $\tilde{\mathbf{b}}_i \in \mathbb{C}^m$ for $i = 1, \dots, r$,

and seek a reduced model defined by operators $\hat{\mathbf{E}}$, $\hat{\mathbf{A}}$, $\hat{\mathbf{B}}$, $\hat{\mathbf{C}}$ such that the reduced transfer function $\hat{\mathbf{G}}(s)$ satisfies

$$\tilde{\mathbf{c}}_i^\top \hat{\mathbf{G}}(\mu_i) = \tilde{\mathbf{c}}_i^\top \mathbf{G}(\mu_i) \quad \text{for} \quad i = 1, \dots, r, \tag{4.33}$$

$$\hat{\mathbf{G}}(\sigma_j)\tilde{\mathbf{b}}_j = \mathbf{G}(\sigma_j)\tilde{\mathbf{b}}_j \quad \text{for} \quad j = 1, \dots, r. \tag{4.34}$$

Theorem 4.3. *Let $\sigma, \mu \in \mathbb{C}$ and let $\sigma\mathbf{E} - \mathbf{A}$, $\sigma\hat{\mathbf{E}} - \hat{\mathbf{A}}$, $\mu\mathbf{E} - \mathbf{A}$, $\mu\hat{\mathbf{E}} - \hat{\mathbf{A}}$ be invertible. Moreover, let $\tilde{\mathbf{b}}_i \in \mathbb{C}^m$ and $\tilde{\mathbf{c}}_i \in \mathbb{C}^p$ for $i = 1, \dots, r$ be nontrivial tangential directions, then,*

- *If $(\sigma\mathbf{E} - \mathbf{A})^{-1}\mathbf{B}\tilde{\mathbf{b}} \in range(\mathbf{V}_r)$ then $\mathbf{G}(\sigma)\mathbf{b} = \hat{\mathbf{G}}(\sigma)\mathbf{b}$.*
- *If $\tilde{\mathbf{c}}^\top \mathbf{C}[(\mu\mathbf{E} - \mathbf{A})^{-1}]^\top \in range(\mathbf{W}_r)$ then $\tilde{\mathbf{c}}^\top \mathbf{G}(\mu) = \tilde{\mathbf{c}}^\top \hat{\mathbf{G}}(\mu)$.*
- *If $\mu = \sigma$, and the above two conditions hold, then, $\tilde{\mathbf{c}}^\top \mathbf{G}'(\sigma)\tilde{\mathbf{b}} = \tilde{\mathbf{c}}^\top \hat{\mathbf{G}}'(\sigma)\tilde{\mathbf{b}}$.*

The $\mathcal{H}_2$-optimality conditions can be modified for the MIMO case and IRKA can also be modified for the MIMO case. Details can be found in [1].

5. Nonlinear model reduction

Until this point we have mostly considered model reduction for linear systems. Now, we will discuss nonlinear model reduction, a very active area of research. Nonlinear systems are ubiquitous in application, and are very diverse in their behavior. Many model reduction methods have been developed and successfully applied to nonlinear systems, and we will present some of the most popular ones here, but keep in mind as we move forward that there is generally very little that can be guaranteed about the practical performance of these methods on any particular problem.

5.1. Hyper-reduction

The first group of methods we will consider are categorized as methods for *hyper-reduction.* This name comes from the fact that the methods are typically used in conjunction with POD, and involve an additional layer of approximation/reduction beyond the POD state approximation.

To see this, we consider the following high-dimensional model:

$$\dot{\mathbf{s}} = \mathbf{A}\mathbf{s} + \mathbf{f}(\mathbf{s}), \tag{5.1}$$

where $\mathbf{s} \in \mathbb{R}^N$ is the state, $\mathbf{A} \in \mathbb{R}^{N\times N}$ is some linear operator, and $\mathbf{f} : \mathbb{R}^N \to \mathbb{R}^N$ is some high-dimensional nonlinear function. For example, such a system may arise from discretizing a convection-diffusion-*reaction* equation, where the convection and diffusion terms may be modeled by the linear operator $\mathbf{A}$ and the nonlinear reaction term (perhaps of Arrhenius type) is modeled by the function $\mathbf{f}$.

Recall that in POD, we collect snapshots of the state, and define the POD basis of size r to be the leading left singular vectors of the snapshot matrix:

$$\begin{pmatrix}\mathbf{s}_1 & \mathbf{s}_2 & \cdots & \mathbf{s}_K\end{pmatrix} = \mathbf{S} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^\top. \tag{5.2}$$

We denote the POD basis of size r by $\mathbf{U}_r$, and approximate the state as $\mathbf{s}_{\rm POD} = \mathbf{U}_r\hat{\mathbf{s}}$. Enforcing Galerkin orthogonality yields the following reduced model:

$$\dot{\hat{\mathbf{s}}} = \mathbf{U}_r^\top\mathbf{A}\mathbf{U}_r\hat{\mathbf{s}} + \mathbf{U}_r^\top\mathbf{f}(\mathbf{U}_r\hat{\mathbf{s}}) = \hat{\mathbf{A}}\hat{\mathbf{s}} + \mathbf{U}_r^\top\mathbf{f}(\mathbf{U}_r\hat{\mathbf{s}}). \tag{5.3}$$

The linear matrix operator $\hat{\mathbf{A}} = \mathbf{U}_r^\top\mathbf{A}\mathbf{U}_r$ can be pre-computed in an offline phase to yield an $r\times r$ operator that be efficiently used in an online phase to evolve the r-dimensional reduced state. However, the reduced nonlinear term suffers from two high-dimensional *computational bottlenecks*:

1. the need to reconstruct $\mathbf{s}_{\rm POD} = \mathbf{U}_r\hat{\mathbf{s}} \in \mathbb{R}^N$, and
2. the need to evaluate $\mathbf{f} : \mathbb{R}^N \to \mathbb{R}^N$.

Methods for hyper-reduction seek to circumvent these computational bottlenecks by introducing an additional approximation for this nonlinear term.

5.2. DEIM

Our first stop on our tour of methods for nonlinear model reduction is DEIM, which stands for Discrete Empirical Interpolation Method. DEIM was proposed in 2010 [4], DEIM and its variants are the current state of the art in nonlinear model reduction. DEIM is attractive because like POD, it can be applied to any nonlinear term – however, like POD, its performance is very problem dependent.

5.2.1. Approximation

In DEIM, we will define a DEIM basis of size ρ, denoted $\mathbf{\Phi}_\rho$, which is basically a POD basis *for the nonlinear term.* That is, let $\mathbf{f}_1, \mathbf{f}_2, \ldots, \mathbf{f}_K \in \mathbb{R}^N$ denote snapshots of the nonlinearity $\mathbf{f}$ – these are usually defined such that $\mathbf{f}_i = \mathbf{f}(\mathbf{s}_i)$, but this doesn't have to hold. Then, we take the SVD of the nonlinear snapshots,

$$\begin{pmatrix}\mathbf{f}_1 & \mathbf{f}_2 & \cdots & \mathbf{f}_K\end{pmatrix} = \mathbf{F} = \mathbf{\Phi}\mathbf{\Xi}\mathbf{\Psi}^\top, \tag{5.4}$$

and define $\mathbf{\Phi}_\rho$ to be the ρ leading left singular vectors of $\mathbf{F}$. We will approximate the nonlinear term $\mathbf{f}$ within the span of the DEIM basis:

$$\mathbf{f}_{\text{DEIM}}(\hat{\mathbf{s}}) = \mathbf{\Phi}_\rho \mathbf{c}(\hat{\mathbf{s}}), \tag{5.5}$$

where $\mathbf{c}$ are the coefficients of the DEIM basis of our approximation, which depend on the current reduced state $\hat{\mathbf{s}}$. Clearly, since $\rho < N$, we must have $\mathbf{f}_{\text{DEIM}}(\hat{\mathbf{s}}) \neq \mathbf{f}(\mathbf{U}_r\hat{\mathbf{s}})$. One might be tempted to use the orthogonal projection of $\mathbf{f}$ onto the DEIM subspace - the problem with that is that we cannot compute it without computing the full $\mathbf{f}$.

The main idea of DEIM is that we will choose coefficients $\mathbf{c}(\hat{\mathbf{s}})$ so that $\mathbf{f}_{\text{DEIM}}(\hat{\mathbf{s}})$ matches $\mathbf{f}(\mathbf{U}_r\hat{\mathbf{s}})$ *at ρ specified interpolation points.*

Let $\mathbf{P} \in \mathbb{R}^{N\times\rho}$ denote a *selection matrix*, whose columns $\mathbf{p}_i$ are all 0s except for a single 1. For example, if we let $\rho = 3$ and $N = 5$, then the matrix

$$\mathbf{P} = \begin{pmatrix}1 & 0 & 0\\ 0 & 1 & 0\\ 0 & 0 & 0\\ 0 & 0 & 1\\ 0 & 0 & 0\end{pmatrix}, \quad \mathbf{P}^\top = \begin{pmatrix}1 & 0 & 0 & 0 & 0\\ 0 & 1 & 0 & 0 & 0\\ 0 & 0 & 0 & 1 & 0\end{pmatrix} \tag{5.6}$$

is such that $\mathbf{P}^\top\mathbf{f}$ will yield a vector in $\mathbb{R}^3$ whose three entries are the first, second, and fourth entries of $\mathbf{f} \in \mathbb{R}^5$. Given a selection matrix $\mathbf{P}$, we want to choose $\mathbf{c}(\hat{\mathbf{s}})$ such that $\mathbf{P}^\top\mathbf{f}_{\text{DEIM}}(\hat{\mathbf{s}}) = \mathbf{P}^\top\mathbf{f}(\mathbf{U}_r\hat{\mathbf{s}})$. Thus, we have

$$\mathbf{P}^\top\mathbf{\Phi}_\rho\mathbf{c}(\hat{\mathbf{s}}) = \mathbf{P}^\top\mathbf{f}(\mathbf{U}_r\hat{\mathbf{s}}) \tag{5.7}$$

$$\mathbf{c}(\hat{\mathbf{s}}) = (\mathbf{P}^\top\mathbf{\Phi}_\rho)^{-1}\mathbf{P}^\top\mathbf{f}(\mathbf{U}_r\hat{\mathbf{s}}). \tag{5.8}$$

Thus, with eq. (5.5), our $\mathbf{f}_{\text{DEIM}}$ approximation is given by

$$\mathbf{f}_{\text{DEIM}}(\hat{\mathbf{s}}) = \mathbf{\Phi}_\rho(\mathbf{P}^\top\mathbf{\Phi}_\rho)^{-1}\mathbf{P}^\top\mathbf{f}(\mathbf{U}_r\hat{\mathbf{s}}), \tag{5.9}$$

and the POD-DEIM reduced model is given by

$$\dot{\hat{\mathbf{s}}} = \hat{\mathbf{A}}\hat{\mathbf{s}} + \mathbf{U}_r^\top\mathbf{f}_{\text{DEIM}}(\hat{\mathbf{s}}). \tag{5.10}$$

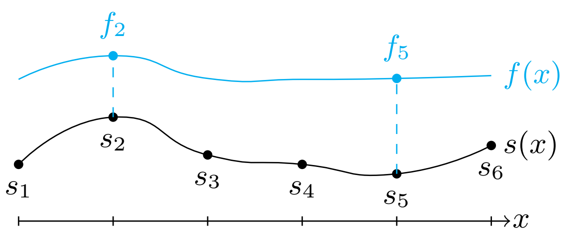


Figure 5.1.: DEIM pointwise nonlinear dependence sketch for $N = 6$, $\rho = 2$.

5.2.2. Cost savings

For pointwise f

We consider first a special case, where $\mathbf{f}(\mathbf{s})$ has 'pointwise' dependence on $\mathbf{s}$, i.e., where

$$\mathbf{s} = \begin{pmatrix} s_1 \\ s_2 \\ \vdots \\ s_N \end{pmatrix}, \qquad \mathbf{f}(\mathbf{s}) = \begin{pmatrix} f_1(s_1) \\ f_2(s_2) \\ \vdots \\ f_N(s_N) \end{pmatrix}. \tag{5.11}$$

This type of local nonlinearity is common in reaction modeling, where the nonlinear reaction depends on the concentrations of chemical species at a given point. This pointwise dependence allows us circumvent both computational bottlenecks of nonlinear POD. To see this, consider the sketch in Figure 5.1, where $N = 6$, $\rho = 2$, and the selection matrix $\mathbf{P}$ selects rows 2 and 5. To evaluate $\mathbf{P}^\top\mathbf{f}$ at these rows, we need only evaluate f_2 and f_5, which only require us to reconstruct s_2 and s_5, not the whole nonlinear state. Formally, we can define the reduced nonlinearity $\mathbf{f}_\rho : \mathbb{R}^\rho \to \mathbb{R}^\rho$ as follows:

$$\mathbf{f}_\rho(\mathbf{P}^\top\mathbf{U}_r\hat{\mathbf{s}}) = \mathbf{P}^\top\mathbf{f}(\mathbf{P}^\top\mathbf{U}_r\hat{\mathbf{s}}) \tag{5.12}$$

and eq. (5.10) becomes

$$\dot{\hat{\mathbf{s}}} = \hat{\mathbf{A}}\hat{\mathbf{s}} + \mathbf{U}_r^\top\mathbf{\Phi}_\rho(\mathbf{P}^\top\mathbf{\Phi}_\rho)^{-1}\mathbf{f}_\rho(\mathbf{P}^\top\mathbf{U}_r\hat{\mathbf{s}}), \tag{5.13}$$

where the product $\mathbf{U}_r^\top\mathbf{\Phi}_\rho(\mathbf{P}^\top\mathbf{\Phi}_\rho)^{-1} \in \mathbb{R}^{r\times\rho}$ can be pre-computed offline, the matrix $\mathbf{P}^\top\mathbf{U}_r \in \mathbb{R}^{\rho\times r}$ can be pre-computed offline, and and evaluating $\mathbf{f}_\rho$ is a low-dimensional nonlinear function evaluation, leading to cost savings.

Cost savings for more general f

What about more general $\mathbf{f}$, where each entry of $\mathbf{f}$ may depend on any or all of the entries of $\mathbf{s}$? This latter case, where $\mathbf{f}$ has truly global dependence, is problematic for DEIM, because we cannot circumvent the first computational bottleneck of state reconstruction. Fortunately, however, many nonlinearities arising in application do not exhibit this global nonlinear dependence — think of nonlinearities that involve derivatives of the state. For standard discretization methods, each entry of $\mathbf{f}$ will only depend on a few entries of $\mathbf{s}$ specified by the stencil used in the discretization.

For example, consider the sketch in Figure 5.2, where a 2-point stencil is used to discretize the local nonlinearity. In contrast to the setting of Figure 5.1, to evaluate $\mathbf{P}^\top\mathbf{f}$ at rows 2 and 5, we have to reconstruct s_1 and s_2 to evaluate f_2, and s_4 and s_5 are needed to evaluate s_5. Thus an m-point stencil will generally require that we reconstruct $m\rho$ entries of $\mathbf{s}$ to evaluate $\mathbf{P}^\top\mathbf{f}$. We

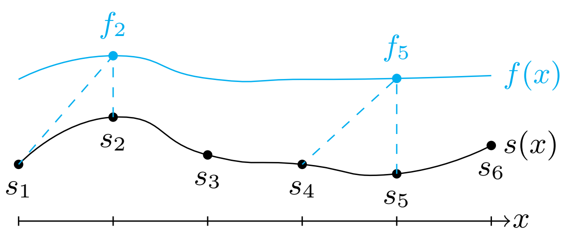


Figure 5.2.: DEIM local nonlinear dependence sketch for $N = 6$, $\rho = 2$.

still come out ahead if $m\rho \ll N$, which happens in application for large N and nonlinearities with very sparse Jacobians. The original DEIM paper [4] provides some more detail about how DEIM for nonlinearities with sparse Jacobians can be efficiently implemented using a particular sparse matrix representation.

5.2.3. DEIM error

Let's consider the error between $\mathbf{f}(\mathbf{s})$ and its DEIM approximant $\mathbf{f}_{\text{DEIM}}(\mathbf{s})$ in the 2-norm:

$$\|\mathbf{f}(\mathbf{s}) - \mathbf{f}_{\text{DEIM}}(\mathbf{s})\|_2. \tag{5.14}$$

Let $\mathbf{f}_* = \mathbf{\Phi}_\rho \mathbf{\Phi}_\rho^\top \mathbf{f}$ denote the orthogonal projection of $\mathbf{f}$ onto the DEIM basis - note that this is the optimal approximation in the 2-norm within the space spanned by the DEIM basis. Note that

$$\mathbf{\Pi} = \mathbf{\Phi}_\rho (\mathbf{P}^\top \mathbf{\Phi}_\rho)^{-1} \mathbf{P}^\top \tag{5.15}$$

is a projection operator onto the range of $\mathbf{\Phi}_\rho$, i.e., $\mathbf{\Pi}^2 = \mathbf{\Pi}$ and $\mathbf{\Pi}\mathbf{\Phi}_\rho = \mathbf{\Phi}_\rho$. Thus, we can write (dropping the $\mathbf{s}$ dependency for shorthand)

$$\mathbf{f}_{\text{DEIM}} = \mathbf{\Pi}\mathbf{f} = \mathbf{\Pi}(\mathbf{f} - \mathbf{f}_* + \mathbf{f}_*) = \mathbf{\Pi}(\mathbf{f} - \mathbf{f}_*) + \mathbf{f}_*. \tag{5.16}$$

Thus,

$$\mathbf{f} - \mathbf{f}_{\text{DEIM}} = \mathbf{f} - \mathbf{f}_* - \mathbf{\Pi}(\mathbf{f} - \mathbf{f}_*) = (\mathbf{I} - \mathbf{\Pi})(\mathbf{f} - \mathbf{f}_*), \tag{5.17}$$

so

$$\|\mathbf{f} - \mathbf{f}_{\text{DEIM}}\|_2 \leq \|\mathbf{I} - \mathbf{\Pi}\|_2 \|\mathbf{f} - \mathbf{f}_*\|_2 = \|\mathbf{I} - \mathbf{\Pi}\|_2 \|(\mathbf{I} - \mathbf{\Phi}_\rho \mathbf{\Phi}_\rho^\top)\mathbf{f}\|_2. \tag{5.18}$$

Note that $\|\mathbf{I} - \mathbf{\Pi}\|_2 = \|\mathbf{\Pi}\|_2$ for any projector that is not zero or the identity, so $\|\mathbf{I} - \mathbf{\Pi}\|_2 \leq \|\mathbf{\Phi}_\rho (\mathbf{P}^\top \mathbf{\Phi}_\rho)^{-1} \mathbf{P}^\top\|_2 \leq \|(\mathbf{P}^\top \mathbf{\Phi}_\rho)^{-1}\|_2$, so we have

$$\|\mathbf{f} - \mathbf{f}_{\text{DEIM}}\|_2 \leq \|(\mathbf{P}^\top \mathbf{\Phi}_\rho)^{-1}\|_2 \|(\mathbf{I} - \mathbf{\Phi}_\rho \mathbf{\Phi}_\rho^\top)\mathbf{f}\|_2. \tag{5.19}$$

5.2.4. DEIM algorithm and bound

Our error analysis of the previous subsection clearly shows that the DEIM error depends on the choice of interpolation points represented by $\mathbf{P}$. The original DEIM paper provides a greedy algorithm for choosing the interpolation points. This DEIM algorithm (Algorithm 3) starts by choosing the entry that maximizes $(\mathbf{P}^\top \mathbf{\Phi}_\rho)$ for a DEIM basis size of $\rho = 1$, motivated by eq. (5.19), and then iteratively chooses the next interpolation index to add as the one that

Algorithm 3 DEIM algorithm for selecting interpolation points

```
0: Input: DEIM basis $\mathbf{\Phi}_\rho \in \mathbb{R}^{N\times\rho}$
1: $p_1 = \texttt{argmax}_i(\texttt{abs}(\mathbf{\Phi}_\rho(i,1)))$
2: $\mathbf{P} = \mathbf{e}_{p_1}$
3: for $r = 2$ to $\rho$ do
4:     $\beta = (\mathbf{I} - \mathbf{\Phi}_{:,1:r-1}(\mathbf{P}^\top\mathbf{\Phi}_{:,1:r-1})^{-1}\mathbf{P}^\top)\mathbf{\Phi}_{:,r}$
5:     $p_r = \texttt{argmax}_i(\texttt{abs}(\beta_i))$
6:     $\mathbf{P} = [\mathbf{P}, \mathbf{e}_{p_r}]$
7: end for
8: return P
```

yields the highest $(\mathbf{I} - \mathbf{\Pi})\mathbf{\Phi}_{:,r}$. This is a greedy strategy because it sets the current highest interpolation error for the next basis function to 0.

The original DEIM paper provides a bound on the factor $\|(\mathbf{P}^\top\mathbf{\Phi}_\rho)^{-1}\|_2$ in the DEIM error eq. (5.19) based on Algorithm 3:

$$\|(\mathbf{P}^\top\mathbf{\Phi}_\rho)^{-1}\|_2 \leq (1+\sqrt{2N})^{\rho-1}\|\phi_1\|_\infty^{-1}. \tag{5.20}$$

This is a terrible bound! For even a modest number of basis functions ρ, this factor grows exponentially. Fortunately, the bound tends to be overly conservative/pessimistic (although one can construct examples for which the error does indeed grow this fast). In practice, since $(\mathbf{P}^\top\mathbf{\Phi}_\rho)^{-1} \in \mathbb{R}^{\rho\times\rho}$ we can just compute this norm to get an a posteriori bound, which tends to be much tighter.

5.3. QDEIM

A variant of DEIM called QDEIM was introduced in 2016 [5]. In QDEIM, the interpolatory projection is defined in the same way as in DEIM, but the choice of interpolation points is given by computing a QR factorization with permutations of $\mathbf{\Phi}_\rho^\top$. The QDEIM algorithm leads to the following improved bound on the factor $\|(\mathbf{P}^\top\mathbf{\Phi}_\rho)^{-1}\|_2$:

$$\|(\mathbf{P}^\top\mathbf{\Phi}_\rho)^{-1}\|_2 \leq \sqrt{1+\rho(N-\rho)}. \tag{5.21}$$

This is much better because the bound only grows linearly in the number of interpolation points/DEIM basis size. It is still more pessimistic than just computing $\|(\mathbf{P}^\top\mathbf{\Phi}_\rho)^{-1}\|_2$ *a posteriori*, but is more useful as an *a priori* error bound than the original DEIM bound in eq. (5.20).

Why does this work? At a high-level, we can motivate QDEIM by noting that $\|\mathbf{P}^\top\mathbf{\Phi}_\rho\|_2 = 1$, and so $\|(\mathbf{P}^\top\mathbf{\Phi}_\rho)^{-1}\|_2 = \texttt{cond}(\mathbf{P}^\top\mathbf{\Phi}_\rho)$. Thus, to prevent the factor $\|\mathbf{P}^\top\mathbf{\Phi}_\rho\|_2$ in the DEIM error from growing too much, we want to control the conditioning of $\mathbf{P}^\top\mathbf{\Phi}_\rho$. This is exactly what QR with column pivoting seeks to do. That is, QR with column pivoting seeks a permuted matrix decomposition:

$$\mathbf{\Phi}_\rho^\top\begin{pmatrix}\mathbf{P} & \mathbf{P}'\end{pmatrix} = \mathbf{QR} = \begin{pmatrix} | & & | \\ \mathbf{q}_1 & \cdots & \mathbf{q}_\rho \\ | & & | \end{pmatrix}\begin{pmatrix} r_{11} & r_{12} & \cdots & r_{1\rho} & * & \cdots & * \\ & r_{22} & \cdots & r_{2\rho} & * & \cdots & * \\ & & \ddots & \vdots & * & \cdots & * \\ & & & r_{\rho\rho} & * & \cdots & * \end{pmatrix} = \mathbf{Q}\begin{pmatrix}\mathbf{R} & \mathbf{R}'\end{pmatrix}$$

with $\mathbf{q}_i$ orthonormal, and $\begin{pmatrix}\mathbf{P} & \mathbf{P}'\end{pmatrix}$ a permutation such that $\mathbf{R}$ has good conditioning. This leads to $\mathbf{QR} = \mathbf{\Phi}_\rho^\top\mathbf{P}$ with good conditioning. We can therefore use efficient QR implementations to scalably compute $\mathbf{P}$. In practice, QDEIM has a similar performance to DEIM, but is more efficient to compute (and rank-revealing QR algorithms which return the permutation matrix have been scalably implemented for large-scale systems).

Algorithm 4 QDEIM algorithm for selecting interpolation points

```
0: Input: DEIM basis $\mathbf{\Phi}_\rho \in \mathbb{R}^{N\times\rho}$
1: $[\mathbf{Q}, \mathbf{R}, \mathbf{P}] = \texttt{qr}(\mathbf{\Phi}_\rho^\top)$
2: return P
```

5.4. DEIM extensions

We note that (Q)DEIM involves two different approximations:

1. linear approximation within the DEIM subspace, and
2. interpolatory projection onto the DEIM subspace.

There are some clear limitations of these approximation approaches:

- complicated nonlinearities may not have a low-dimensional DEIM subspace that captures most of the nonlinear energy, and
- interpolation is notoriously unstable.

These limitations suggest extensions of DEIM to address these issues: adaptive/local DEIM bases that are low-dimensional even for complicated nonlinearities, and oversampling/regression approaches that are more robust than interpolation.

5.4.1. LDEIM

5.4.2. ADEIM

5.4.3. ODEIM

It is known that interpolatory methods tend to suffer when the interpolation data are corrupted by noise. In [8], the authors provide a probablistic analysis showing that the QDEIM error actually increases as ρ increases when the data are noisy. To remedy this problem, the authors propose an strategy that selects $s > \rho$ points in the nonlinearity and seeks the projection within the DEIM subspace that best fits the s points in a least-squares *regression.* That is, let $\mathbf{O} \in \mathbb{R}^{N\times s}$ be the *oversampled* selection operator, and the oversampled regressive projection is:

$$\mathbf{\Pi_O} = \mathbf{\Phi}_\rho(\mathbf{O}^\top\mathbf{\Phi}_\rho)^\dagger\mathbf{O}^\top, \tag{5.22}$$

where $\cdot^\dagger$ denote the Moore-Penrose pseudo-inverse. Multiple strategies for selecting the extra $s-\rho$ points exist, including random sampling, using the DEIM algorithm (Algorithm 3), using leverage scores, and the GappyPOD+E strategy proposed in [8]. In [8], it is shown that all of these strategies can limit the instability in the presence of noise, although some have lower error.

How many extra points should be used? Numerical experiments in [8] suggest that the error decays at a rate of $\sqrt{1/s}$ to the error level of the noiseless data.

connection to MPE/gappy POD

6. Data-driven methods

A. Generalized Eigenvalue Problems

Bibliography

[1] Athanasios Constantinos Antoulas, Christopher Andrew Beattie, and Serkan Güğercin. *Interpolatory methods for model reduction.* SIAM, 2020.

[2] Peter Benner, Mario Ohlberger, Albert Cohen, and Karen Willcox. *Model reduction and approximation: Theory and algorithms.* SIAM, 2017.

[3] Gal Berkooz, Philip Holmes, and John L Lumley. The proper orthogonal decomposition in the analysis of turbulent flows. *Annual Review of Fluid Mechanics*, 25(1):539–575, 1993.

[4] Saifon Chaturantabut and Danny C Sorensen. Nonlinear model reduction via discrete empirical interpolation. *SIAM Journal on Scientific Computing*, 32(5):2737–2764, 2010.

[5] Zlatko Drmac and Serkan Gugercin. A new selection operator for the discrete empirical interpolation method—improved a priori error bound and extensions. *SIAM Journal on Scientific Computing*, 38(2):A631–A648, 2016.

[6] Kunihiko Taira et al. Modal analysis of fluid flows: An overview. *AIAA Journal*, 55(12):4013–4041, 2017.

[7] Serkan Gugercin, Athanasios C Antoulas, and Christopher Beattie. H_2 model reduction for large-scale linear dynamical systems. *SIAM journal on matrix analysis and applications*, 30(2):609–638, 2008.

[8] B. Peherstorfer, Z. Drmac, and S. Gugercin. Stability of discrete empirical interpolation and gappy proper orthogonal decomposition with randomized and deterministic sampling points. *SIAM Journal on Scientific Computing*, 2020.