Interpolation in high dimensions: Non-intrusive reduced order modeling

Akil Narayan[1]

[1] Department of Mathematics
University of Massachusetts Dartmouth

June 7, 2013
GR-ROM @ Caltech
Pasadena, CA

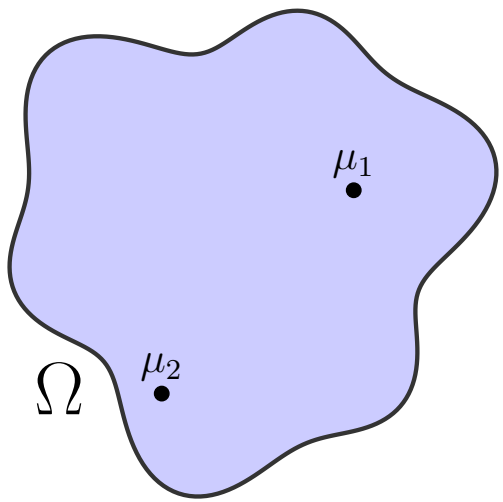
Parameterized functions

Problems of interest are often functions that depend both on space x and a parametric variable μ.

Let $x \in D \subseteq \mathbb{R}^p$ be a space-like variable ($p = 1, 2, 3$) and $\mu \in \Omega \subseteq \mathbb{R}^d$ a parameter ($d \geq 1$).

If $u = u(x; \mu)$, an approximation $u_N \simeq u$ is usually formed via some combined spatial (x) discretization and parametric (μ) discretization.

The whole game: compute u_N.
For each μ evaluating $u(x; \mu)$ is expensive.
The main goal: approximate $u(x; \mu)$ with as few parametric degrees of freedom as possible.
In particular, use *only* μ-point-evaluation information.



The main ideas

Why is this important?

- need $u(x;\mu)$ for numerous values of μ
- for a given μ, need fast query of $u(x;\mu)$
- want μ-moment information about $u(x;\mu)$

The major points and discussions in this talk:

- Interpolatory ("non-intrusive") methods can perform on par with projective ("intrusive") methods
- Non-adaptive interpolatory methods: single-dimension fundamentals and high-dimensional techniques
- Adaptive interpolatory methods: optimal approximation spaces and reconstructions

Themes throughout: greedy schemes, pivoted linear algebra routines

General setup

We are concerned with the standard linear approximation techniques:

$$u_N(x;\mu) = \sum_{m,n=1}^{N} c_n b_n(\mu) v_n(x) = \sum_{n=1}^{N} C_n(\mu) v_n(x)$$

The coefficients $C_n(\mu)$ and the basis v_n determine the approximation.

Some notation throughout:

- V approximates x
- V_n subspace of V with basis v_n
- B approximates μ
- B_n subspace of B with basis b_n

Generally, simulation tools are developed to evaluate the following map:

$$\mu \mapsto u(x;\mu), \qquad \mu \text{ fixed} \tag{1}$$

This limited information about $u(x;\mu)$ constrains our knowledge.

Intrusive methods

One approach: with some preconceived basis $v_n(x)$, $b_n(\mu)$ in a Hilbert space $V \times B$, construct

$$u_N(x;\mu) = \sum_{m,n=1}^{N} c_{m,n} b_m(\mu) v_n(x),$$

and ask that $u_N = \text{proj}_{V_N \times B_N} u$.
(Or appropriate residual formulation for DE.)

Determining the approximation coefficients $c_{m,n}$ requires information

$$\langle u(x;\mu), v_n(x) b_m(\mu) \rangle_{V \times B},$$

but we can only evaluate the map $u(x;\mu)$ for a fixed μ.

Thus we require data beyond what (1) can provide, so a rewrite of existing simulation tools is necessary: intrusive.

Non-intrusive methods

A second approach: with some preconceived basis $v_n(x)$, $b_n(\mu)$, construct

$$u_N(x;\mu) = \sum_{m,n=1}^{N} c_{m,n} b_m(\mu) v_n(x),$$

and ask that $u_N(\cdot;\mu_n) = \text{proj}_{V_N} u(\cdot;\mu_n)$ for some chosen nodes μ_n.

Note: In principle no Hilbertian structure on V necessary.

This is an interpolatory approach; the only data we need is $u(x;\mu_n)$ at the sites μ_n.
Thus we can use the existing simulation tools: non-intrusive

A short, sweet example

For concreteness, consider an elliptic problem

$$-\frac{d}{dx}\left(\kappa(x,\mu)\frac{du(x;\mu)}{dx}\right) = f(x;\mu),$$

with $x \subset \mathbb{R}$ and $\mu \in [-1,1]^8 \subset \mathbb{R}^8$. The diffusion coefficient is given by

$$\kappa(x;\mu) = 1 + \sum_{j=1}^{8} \frac{1}{\pi j^2}\cos(2\pi j x)\mu_j,$$

We seek to approximate $u(x;\mu)$: In this case, a non-intrusive method can perform comparably to an intrusive method.

A classical ("intrusive") approach

Consider a single variable (x, μ) and perform a Galerkin (FEM-like) approximation: Find $u_N \in V_N \times B_N$ such that

$$\left\langle -\left(\kappa(x;\mu)u_N'(x;\mu)\right)', v(x;\mu)\right\rangle = \langle f(x;\mu), v(x;\mu)\rangle ,$$

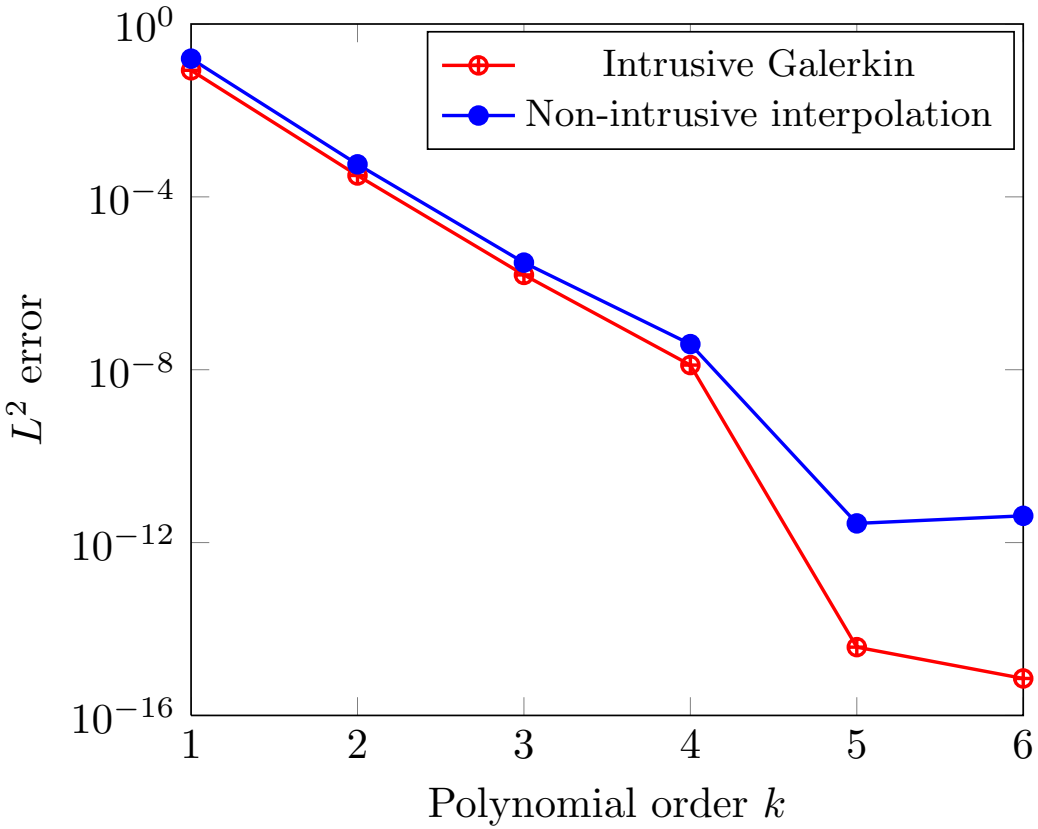
$\forall\, v \in V_N \times B_N$.
This is intrusive: we require projective, non-interpolatory information about $u(\mu)$.

The non-intrusive interpolatory approach: select some parametric locations μ_n, $n = 1, \ldots, N$, and find $u_N \in V_N \times B_N$ such that

$$u_N(x;\mu_n) = u(x;\mu_n), \qquad \forall\, \mu_n$$

Convergence

Intrusive methods are generally more accurate, but are more expensive.



Galerkin-FEM intrusive $k = 6$ solution requires a linear solve of size $\sim 10^5$

The non-intrusive interpolatory $k = 6$ approach requires ~ 3000 linear solves of size ~ 30.

Non-adaptive interpolation topics

Interpolation:

$$u(\cdot;\mu_n) = u_N(\cdot;\mu_n), \qquad n = 1,\ldots,N.$$

Choice of μ_n is the next subject under discussion.

- Lagrange interpolation and Lebesgue constants
- polynomials: one-dimensional grids
- higher dimensions: tensorizations, sparse grids
- greedy, 'unstructured' methods: Fekete and Leja points

Non-adaptive interpolation

For non-adaptive interpolation, *both* the basis and the coefficients are chosen independent of the function u:

$$u(x;\mu) \simeq u_N = \sum_{n=1}^{N} c_n(\mu) u(x;\mu_n)$$

Both the parametric locations μ_n and the parametric dependence $c_n(\mu)$ are free to be chosen. In order to intelligently choose μ_n, we specify a basis for c_n:

$$c_n(\mu) = \sum_{m=1}^{N} c_{n,m} b_m(\mu), \qquad n = 1, \ldots, N.$$

Realistically, the b_n are selected from a standard μ-approximation set, e.g. polynomials, trigonometric functions, wavelets, splines, etc.

Lagrange interpolation

A clearer way to see what is happening: solve for $c_{n,m}$ so that u_N interpolates u at μ_n. Then:

$$C_n(\mu) = \sum_{m=1}^{N} c_{n,m} b_m(\mu) = \ell_n(\mu),$$

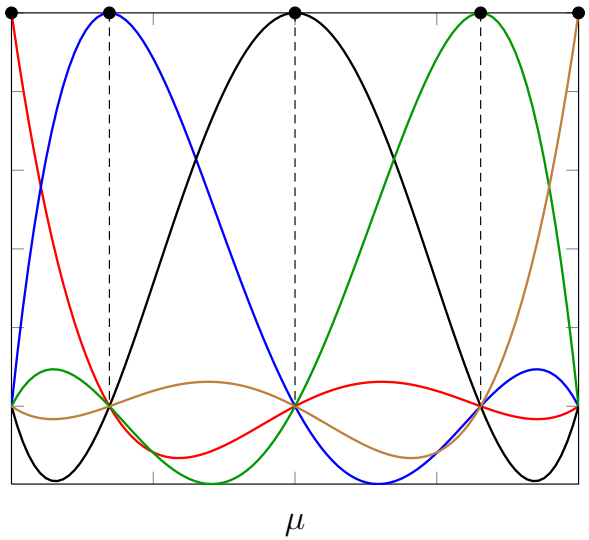
with ℓ_n the cardinal Lagrange interpolant of the b_n at the sites μ_n. I.e.:

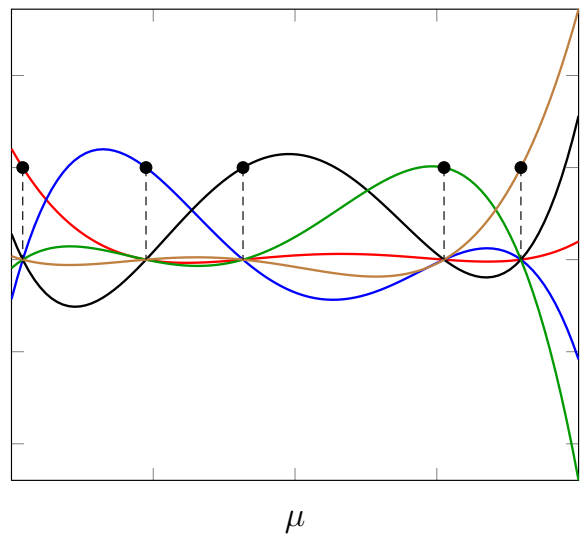
$$\ell_n(\mu_m) = \delta_{n,m}, \qquad n = 1, \ldots, N$$

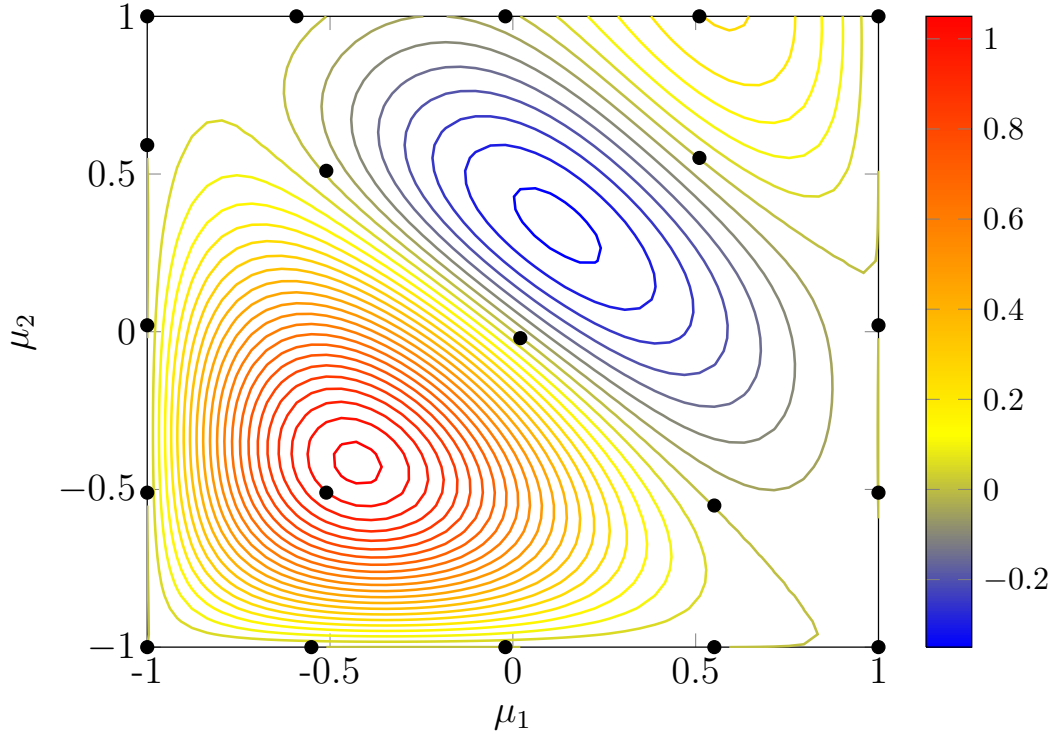
$\ell_n(\mu)$ determines "how much" information from $u(\cdot;\mu_n)$ contributes to reconstruction at μ.

These can be constructed without the data from u. This process is 100% *independent* from u.

Lagrange interpolation







Interpolation Error

Error estimates from classical spatial interpolation may be augmented here.

$$\sup_{\mu\in\Omega} \|u(\cdot;\mu) - u_N(\cdot;\mu)\|_{V_N} \leq \sup_{\mu\in\Omega} \left\|u(\cdot;\mu) - \mathrm{proj}_{V_N} u(\cdot;\mu)\right\| + (1+\Lambda) d\left[\mathrm{proj}_{V_N} u(\cdot;\mu), V_N \times B_N\right]$$

Some of the terms are optimal so we cannot do better. They depend only on the approximation spaces V_N and B_N.

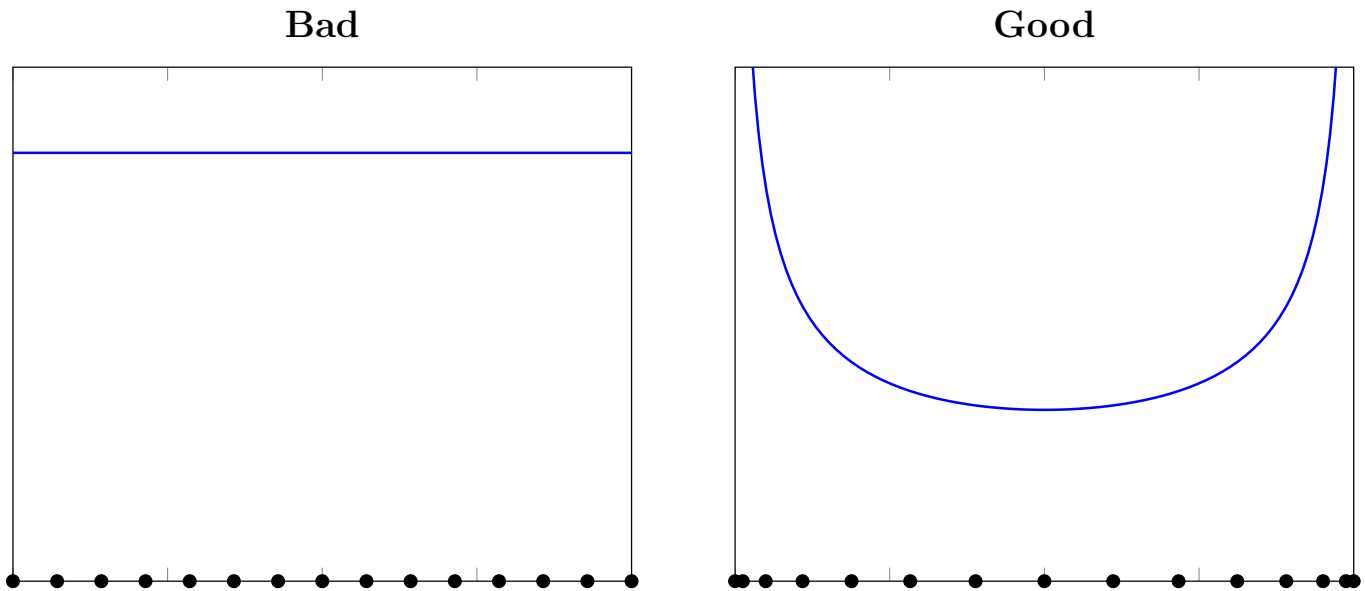
But there is a penalty for interpolation; the "Lebesgue constant". With the approximation space fixed, it depends only on the choice of interpolation nodes.

One central question in the formulation of non-adaptive interpolation methods: how to choose μ_n to minimize Λ?

Polynomial interpolation

For concreteness, consider polynomials. Some one-dimensional intuition:

- the Lebesgue constant for any nodal array is unbounded in N
- equispaced nodes are bad -- exponentially growing Λ
- arcsine-distributed nodes are good -- logarithmically growing Λ



Univariate polynomial interpolation

Great, so what set do I use for interpolation?

- computing Lebesgue-optimal nodes is *hard*
- there are easily computable, "good enough" sets: Chebyshev-type, Gauss-type, Clenshaw-Curtis
- these nodal sets are effectively explicit

But there's more to the story: We want error estimates and refinement capabilities.

One easy solution is hierarchical computations $\longrightarrow$ need nested nodal sequences.

	Equidistant	Gauss	Nested Gauss	Fekete
Accurate?	No	✓	✓	✓
Nested?	Sort of	No*	✓	No
Generation?	✓	✓	Involved	Involved*

Can 1D inform multi-D?

Much of the univariate theory does not extend into the multivariate case:

- computing Lagrange interpolating functions has a subtle complication
- polynomial degree $\neq$ number of nodes N
- $\dim \Pi_k^d = \begin{pmatrix} k+d \\ k \end{pmatrix} \sim k^d$
- no direct analogues of Chebyshev or Gauss constructions
- good, explicit constructions on general geometry $\rightarrow ???$

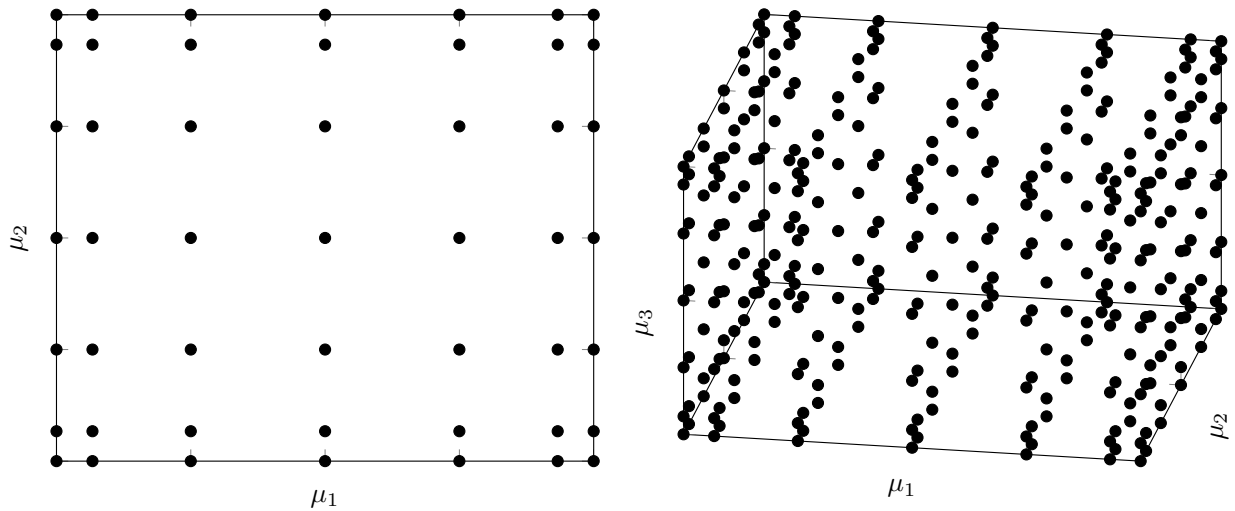
Two standard approaches for extending 1D rules into multi-D: tensorization and sparse grids.

Tensorizing one dimension

If $\Omega = \Omega_1 \times \Omega_2 \times \cdots \times \Omega_d$, use a tensor product rule. Let $\mathcal{M}^d = \left(\mu_1^d, \ldots, \mu_M^d\right)$ be a univariate interpolation set on interval Ω_d. Then the full set is formed as

$$\mathcal{M} = \mathcal{M}^1 \times \mathcal{M}^2 \times \cdots \mathcal{M}^d$$

If each set $\mathcal{M}^d$ has M points, then total number of points is $N = M^d$.

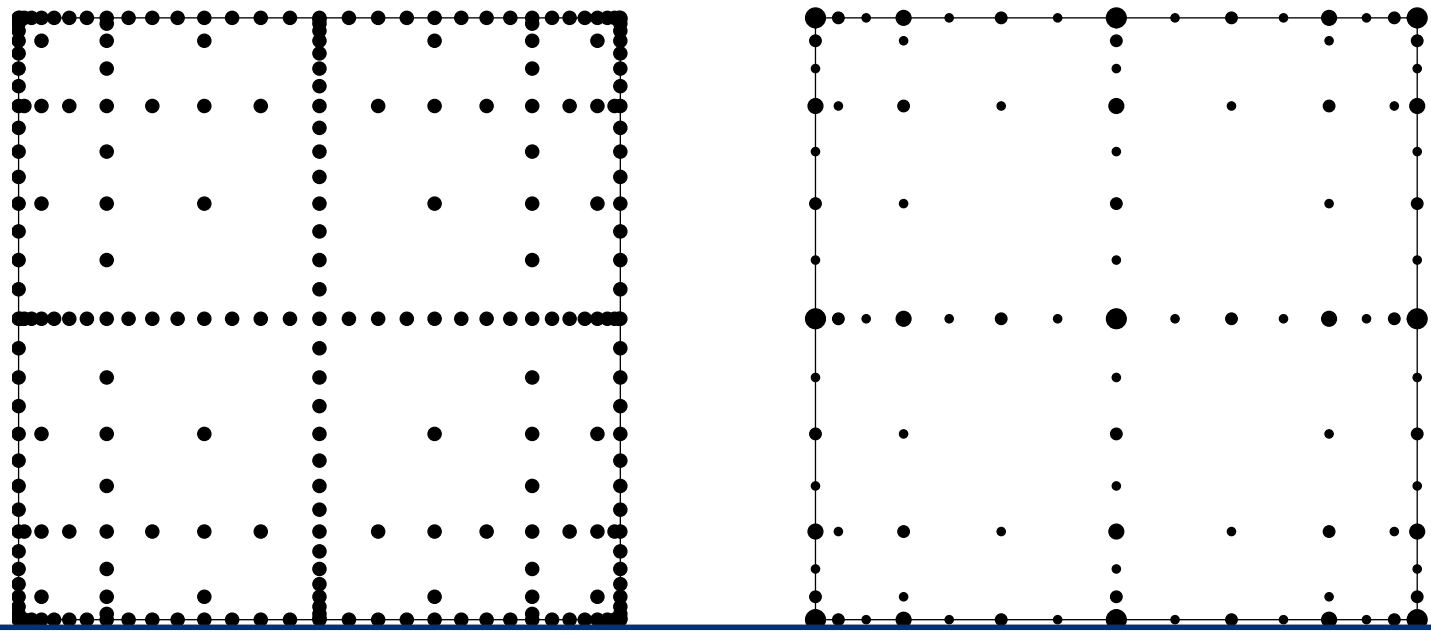


If each univariate rule is "good", then this construction solves the problem of finding a good set ... at the expense of large cardinality.

Sparse grids

To combat the large cardinality of tensor product grids: sparse grids. Define an *array* of univariate nodes for each dimension: a "level" ℓ univariate grid in dimension d: $\mathcal{M}_\ell^d$. The full set is defined as

$$\mathcal{M} = \bigcup_{\substack{\alpha \in \mathbb{N}_0^d \\ |\alpha| \leq \ell}} \left(\bigtimes_{k=1}^{d} \mathcal{M}_{\alpha_k}^k \right)$$



Sparse grids

Sparse grids are very popular:

- delay the curse of dimensionality
- dimensionally adaptive
- are straightforward and explicitly generated
- can generate sparse quadrature rules
- use of nested univariate rules yields nested sparse grids
- hierarchical levels allows dimension-adaptive approximation
- global approximations or local approximations can be used

Nontensorial domains

No silver bullets in general geometries -- but there are some constructive methods for unstructured global interpolation.

One mathematically appealing nodal set: a *Fekete* set.
To solve for the k'th cardinal interpolating function $\ell_k(\mu)$, the following linear system must be solved

$$\sum_{m=1}^{N} A_{n,m} c_{m,k} = \delta_{n,k}, \qquad A_{n,m} = b_m(\mu_n) \longrightarrow \qquad Ac = e_k.$$

A depends on the basis b_n and the nodes μ_n. For a fixed space B_N the nodal set that *maximizes* the determinant of A is a set of *Fekete* nodes:

$$(\widehat{\mu}_1, \ldots, \widehat{\mu}_N) = \underset{(\mu_1,\ldots,\mu_N)\subset\Omega}{\arg\max} \; |\det A(\mu_1, \ldots, \mu_N)|$$

Why Fekete nodes?

Fekete nodes guarantee an at-worst linear growth of Λ. (Usually, the growth is logarithmic.)
Univariate, bounded domain: Fekete $\equiv$ Legendre-Gauss-Lobatto nodes, hence explicitly constructible.

But, multivariate Fekete points are *hard* to construct, require global (dim-N) optimization.
When global optimization is hard, greedy schemes shine: optimize determinantal volume by adding nodes one-at-a-time:

$$\mu_{n+1} = \arg\max_{\mu\in\Omega} \operatorname{vol}_{n+1}\left(b(\mu_1),\ldots,b(\mu_n),b(\mu)\right)$$

with $\dim b(\mu) = N$.
These are *Approximate Fekete Points* (AFP).

Leja Points

AFP are great -- but they are not nested: a size-N AFP set has (almost) no common nodes with a size-$(N+1)$ AFP set.

A second greedy approximation to AFP can produce a nested sequence: Leja points. With Leja points, we also greedily maximize the determinant. The difference: the approximation space is also greedily enlarged.

$$\mu_{n+1} = \arg\max_{\mu\in\Omega} \operatorname{vol}_{n+1}\left(b_{1:(n+1)}(\mu_1), \ldots, b_{1:(n+1)}(\mu_n), b_{1:(n+1)}(\mu)\right)$$

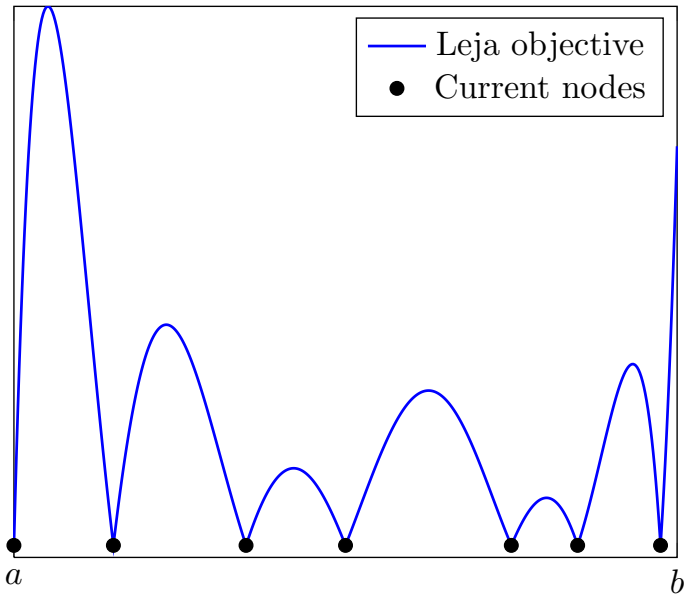
with $\dim b_{1:(n+1)}(\mu) = n+1$.

Leja Points

Example: Let $b_n(\mu) = \mu^{n-1}$. Then μ_{n+1} is defined as

$$\mu_{n+1} = \arg\max_{\mu} \prod_{k=1}^{n} |\mu - \mu_k| ,$$

with $\dim b(\mu) = n + 1$.

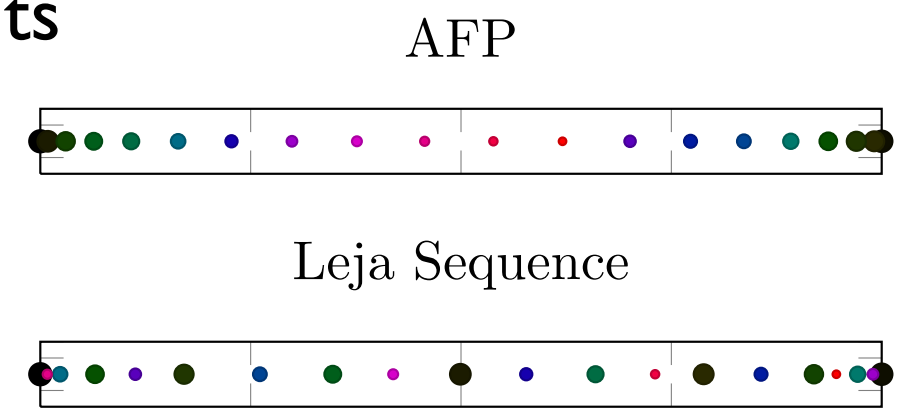


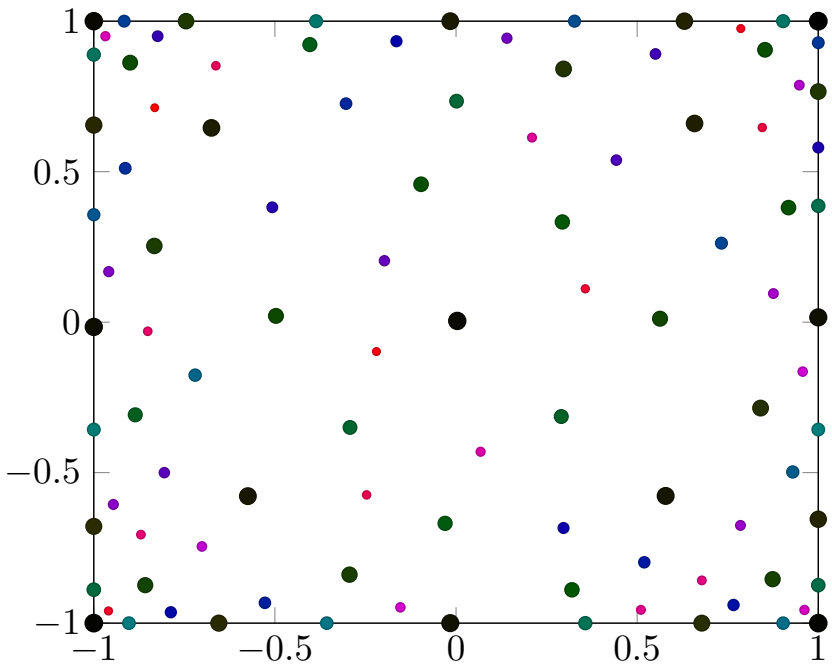
For future reference: The Leja objective is formed via interpolation:

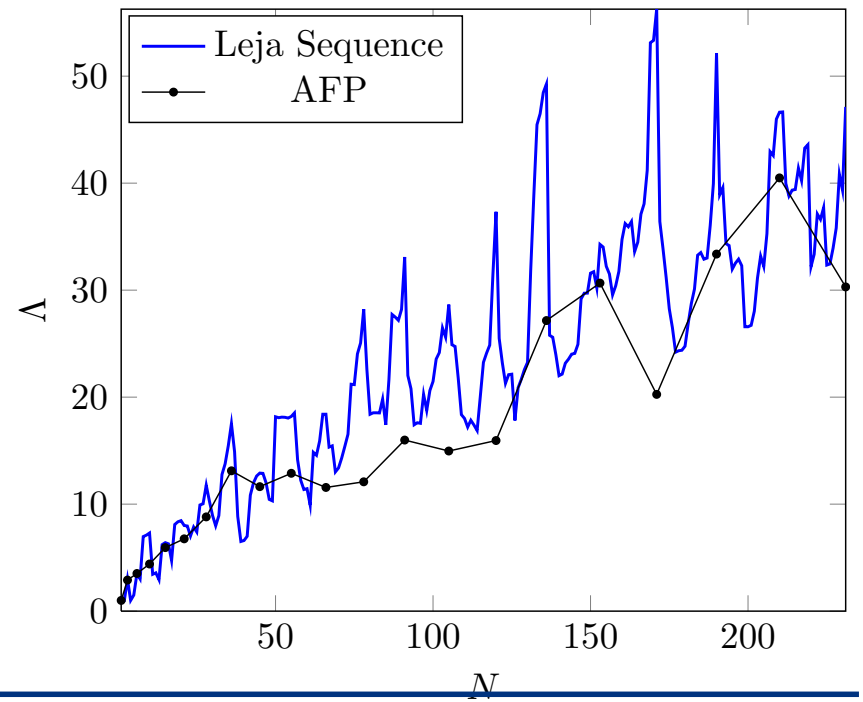
$$\mu_{n+1} = \arg\max_{\mu \in \Omega} |b_{n+1}(\mu) - \mathcal{I}_n b_{n+1}(\mu)| ,$$

where $\mathcal{I}_n$ interpolates at $\mu_1, \ldots, \mu_n$ using $b_1, \ldots, b_n$.

AFP vs Leja points







Is this the right thing to do?

Do AFP and Leja Sequences produce anything meaningful?

- Leja Sequences and AFP are *asymptotically Fekete*
- Leja Sequences and AFP have empirical measures (cf. histogram plot) converge to the *pluripotential equilibrium distribution*
- The interpolants using the above sets converge for analytic functions
- The above are *necessary* conditions for subexponentially-growing Lebesgue constant

None of the above guarantees a good Lebesgue constant -- but usually Λ is quite good.

This is all wonderful -- are AFP and Leja sequences computable?

Discrete AFP

Recall greedy AFP optimization:

$$\mu_{n+1} = \underset{\mu \in \Omega}{\arg\max}\, \mathrm{vol}_{n+1}\left(b(\mu_1), \ldots, b(\mu_n), b(\mu)\right)$$

In optimization algorithms, its standard to replace continuous optimization with discrete candidate sets: $\Omega \leftarrow \{\nu_1, \ldots, \nu_M\}$

For Fekete nodes:

$$A^T = \begin{pmatrix} | & | & & | \\ b(\nu_1) & b(\nu_2) & \cdots & b(\nu_M) \\ | & | & & | \end{pmatrix} \longrightarrow$$

Column pivoted QR factorization greedily maximizes volume spanned by length-N vectors

Discrete Leja points

Recall iterative Leja optimization:

$$\mu_{n+1} = \arg\max_{\mu\in\Omega} = \arg\max_{\mu} \prod_{k=1}^{n} |\mu - \mu_k| ,$$

In optimization algorithms, its standard to replace continuous optimization with discrete candidate sets: $\Omega \leftarrow \{\nu_1, \ldots, \nu_M\}$

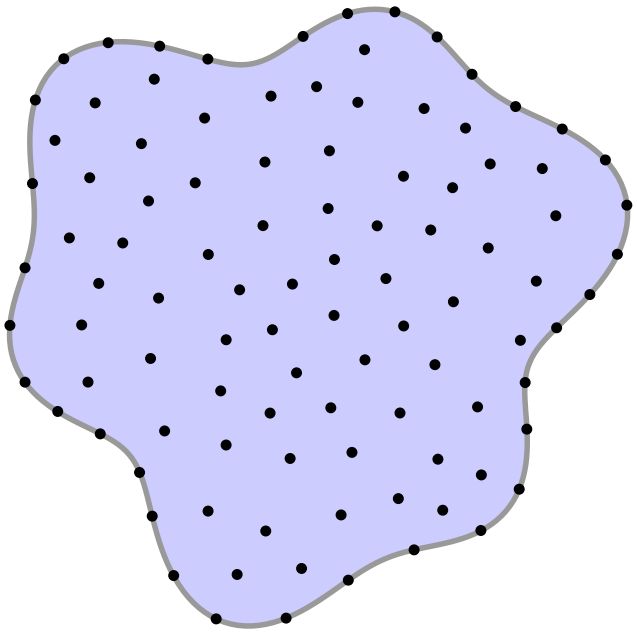
For Leja sequences:

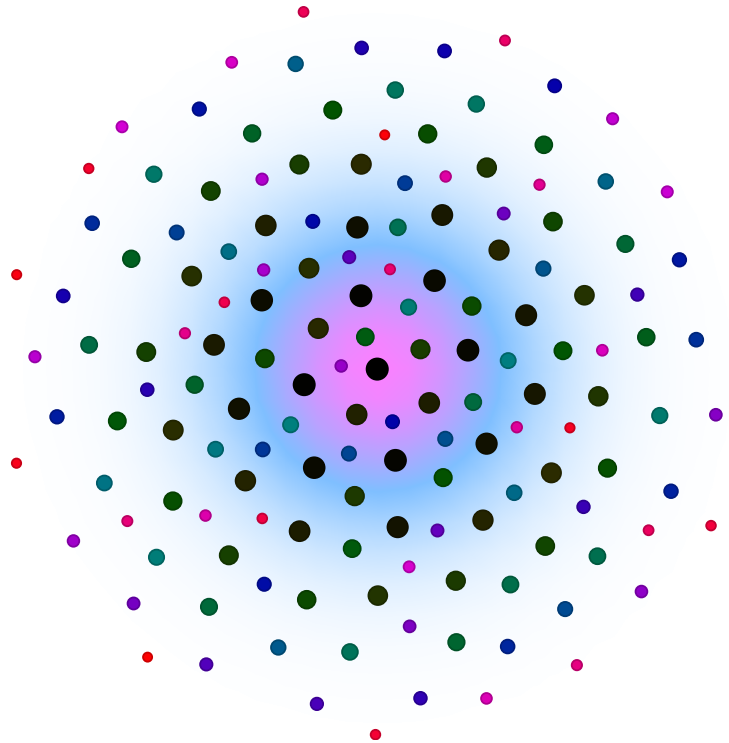
$$A = \begin{pmatrix} — & b(\nu_1)^T & — \\ — & b(\nu_2)^T & — \\ & \vdots & \\ — & b(\nu_M)^T & — \end{pmatrix} \longrightarrow$$

Partial (row) pivoted *LU* factorization greedily maximizes volume spanned spanned by length-*n* vectors

Discrete AFP, DLP

The point: generating discrete AFP and DLP is *very* easy, and can be done with any basis, and in arbitrary geometries.





Nonadaptive interpolation

Attempt to approximate $u(x;\mu)$:

$$u(x;\mu) \simeq u_N(x;\mu) = \sum_{n=1}^{N} C_n(\mu) u(x;\mu_n).$$

With some μ-approximation space B_N prescribed, we choose a "good" interpolation set μ_n.
The C_n are cardinal Lagrange interpolants that *a priori* prescribe parametric variations of u_N.

- polynomials: Gauss-type (Chebyshev) nodes, tensorizations, sparse grid constructions, Fekete nodes, Leja sequences
- AFP and DLP formulations are a greedy procedure, implemented with *LU* and *QR* factorizations
- All can be done without any knowledge of u

Adaptive approximations

Recall the goal is approximation of $u(x;\mu)$ with a linear sum of snapshots

$$u(x;\mu) \simeq u_N(x;\mu) = \sum_{n=1}^{N} c_n(\mu) u(x;\mu_n)$$

Adaptive methods: the μ_n (hence the basis $u(x;\mu_n)$) and the reconstruction coefficients $c_n(\mu)$ depend on the data u.

In adaptive scenarios, we change our point of view, noting that we seek to approximate a functional manifold

$$\mathcal{U} = \{u(x;\mu) | \mu \in \Omega\}$$

Non-adaptive scenarios: we choose an approximation space, the "best" error we can hope for depends on μ-regularity of u.

The path from nonadaptive to adaptive

$$u(x;\mu) \simeq u_N(x;\mu) = \sum_{n=1}^{N} c_n(\mu) u(x;\mu_n)$$

$$V_N = \operatorname{span}\{u(x;\mu_1), \ldots, u(x;\mu_N)\},$$

$$B_N = \operatorname{span}\{c_1(\mu), \ldots, c_N(\mu)\} = \operatorname{span}\{b_1(\mu), \ldots, b_N(\mu)\}$$

In non-adaptive methods:

1. pick parametric basis B_N
2. pick points μ_n
3. look at u
4. define approximation space V_N

In adaptive methods:

1. look at u
2. pick space V_N and points μ_n
3. pick parametric basis B_N

The N width

In adaptive scenarios: do not directly appeal to μ-smoothness of u with respect to μ. Instead, focus abstractly on the "best" possible dimension-N approximation space:

$$d_N(\mathcal{M}) = \inf_{\substack{V_N \subset V \\ \dim V_N = N}} \sup_{\mu \in \Omega} \|u(x;\mu) - u_N(x;\mu)\|,$$

where

$$u_N(x;\mu) = \mathrm{proj}_{V_N} u(x;\mu)$$

Computing the infimizing space $\widehat{V}_N$ is usually intractable, but the N width d_N provides a yard stick for evaluating realistic computational methods.

If we could compute the N-width....

Let u_n be some orthonormal basis for $\widehat{V}_N$. Then the "best" thing to do is use the projection onto V_N.

This prescription defines the μ-variation for us:

$$u_N(x,\mu) = \mathrm{proj}_{V_N} u(x;\mu) = \sum_{n=1}^{N} \widehat{c}_n(\mu) u_n(x), \quad \widehat{c}_n(\mu) = \langle u(x,\mu), u_n(x) \rangle$$

This does *not* necessarily interpolate u for any μ.
It is adaptive: $\widehat{c}_n(\mu)$ depends on u.

Note that the major difficulty above is identification of the approximation space $\widehat{V}_N$.
But since we cannot solve the global optimization problem, how to identify an approximation space?

Greedy approximations

First restrict search: instead of searching ambient space, search only on manifold $\mathcal{U}$.

To identify V_N, use a greedy approach over $\mathcal{M}$:

$$\mu_{n+1} = \arg\max_{\mu\in\Omega} \|u(\cdot,\mu) - P_n u(\cdot,\mu)\|, \quad V_{n+1} = \operatorname{span}\{u(\cdot,\mu_1),\ldots,u(\cdot,\mu_{n+1})\}$$

$$u_N = P_N u$$

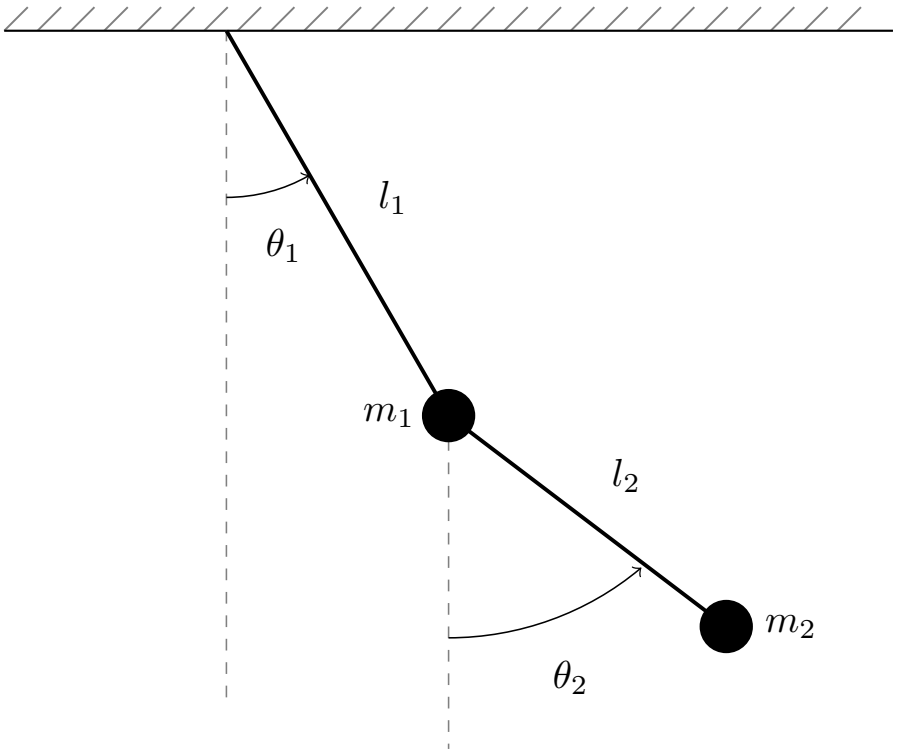
where P_n is the orthogonal projector onto V_n.

Note: now this *is* an interpolatory approach:

- $u_N(\cdot,\mu_n) = u(\cdot,\mu_n)\ \forall n$
- $C_n(\mu)$ are cardinal Lagrange interpolants: $C_n(\mu_m) = \delta_{n,m}$ (But the C_n depend on u!)

The above approach is the skeleton for the Reduced Basis Method (RBM).

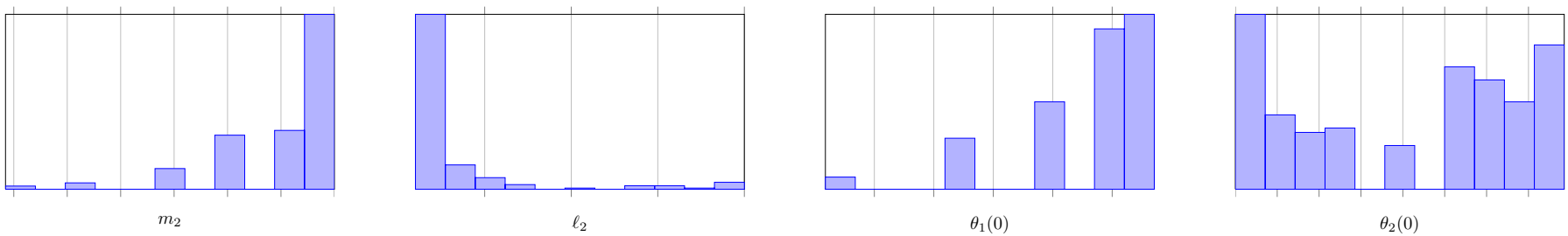
Double pendulum



Set $m_1 = \ell_1 = 1$
Unknown parameters m_2, ℓ_2, $\theta_1(0)$, $\theta_2(0)$

Use trajectory of $\theta_1(t)$ for $0 \le t \le 15$ to form reduced basis.

Choose 200 basis elements using RBM.



On discrete candidate sets: reduced basis search $\equiv$ (pivoted) QR

To EIM and beyond

One complaint about RBM is that (spatial-)projective information about snapshots are required. Can we relax this?

Take another look at the RBM reconstruction: assume $u(x;\mu_n)$ are orthonormal,

$$u_N(x;\mu) = \sum_{n=1}^{N} c_n(\mu) u(x,\mu_n),$$
$$c_n(\mu) = \langle u(x,\mu), u(x,\mu_n)\rangle = U_n^*\left[u(x,\mu)\right],$$

where U_n^* is the Riesz representor for $u(x;\mu_n)$: $U_n^*[v] = \langle v, u(x;\mu_n)\rangle$ for all $v \in V$.
The linear functional U_n^* determines what information we need from u to perform reconstruction at μ.

We can change U_n^* to any convenient functional we like.
Let's change it to point-evaluation.

Empirical Interpolation (EIM)

Assume that some basis $v_1, \ldots, v_N$ is specified for the approximation space V_N. Instead of projecting onto V_N, choose a different approximation:

Given $u(\cdot, \mu)$, determine how to choose approximant from V_N from δ_{x_1}, δ_{x_2}, . . ., δ_{x_N}, where $x_n \in D$.

Choosing x_n: as usual, optimization of N-point configuration is optimal, but difficult. Greedy algorithms to the rescue:

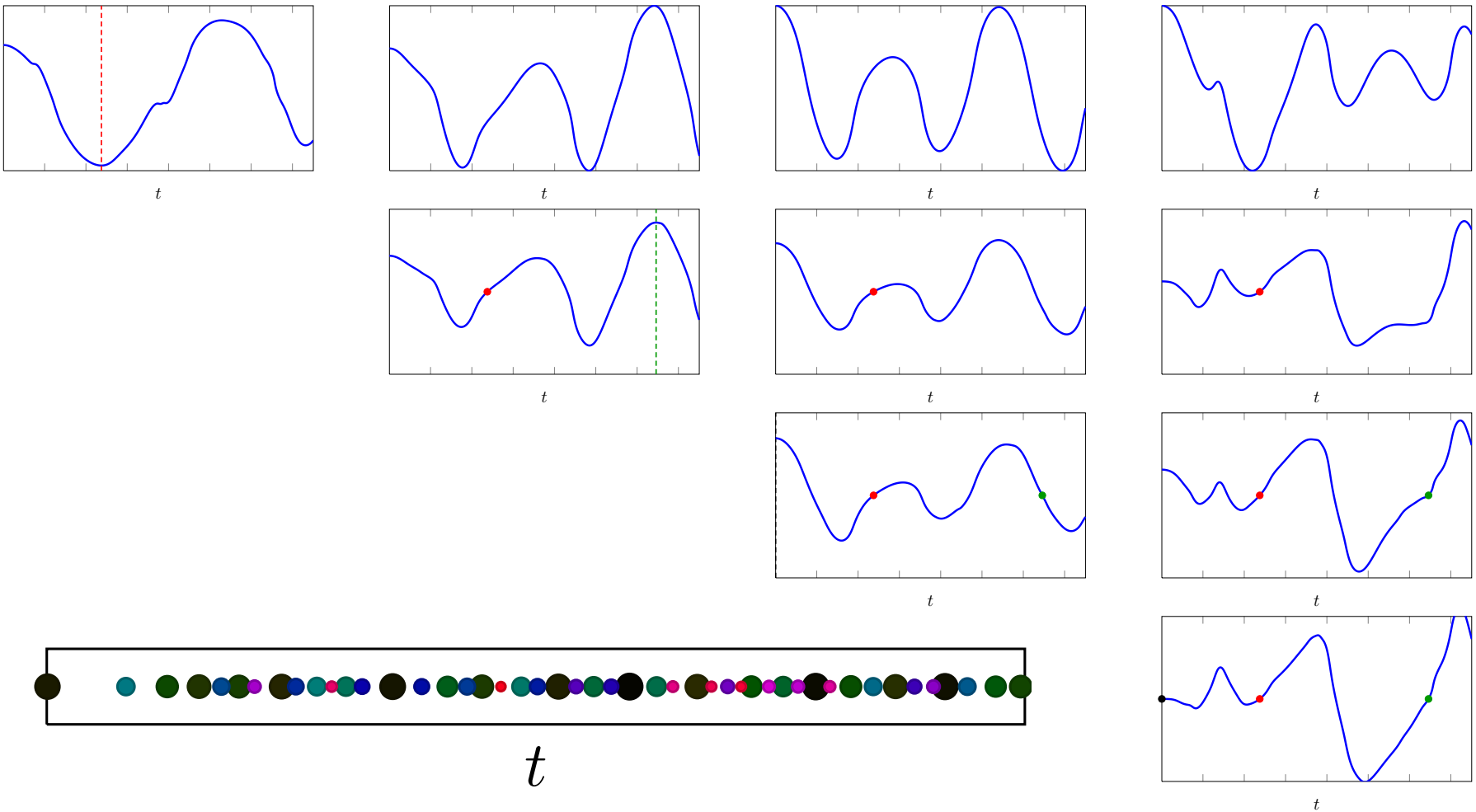
$$x_{n+1} = \arg\max_{x \in D} \left| u_{n+1}(x) - \mathcal{I}_n u_{n+1}(x) \right|,$$

where $\mathcal{I}_n : V \to V_n$ is an interpolation operator: $\mathcal{I}_n v$ interpolates at $x_1, \ldots, x_n$ with $v_1, \ldots, v_n$.

This is just a Leja sequence (in space)

Empirical Interpolation (EIM)

Ok...for a discrete spatial candidate set: "discrete EIM" (DEIM)



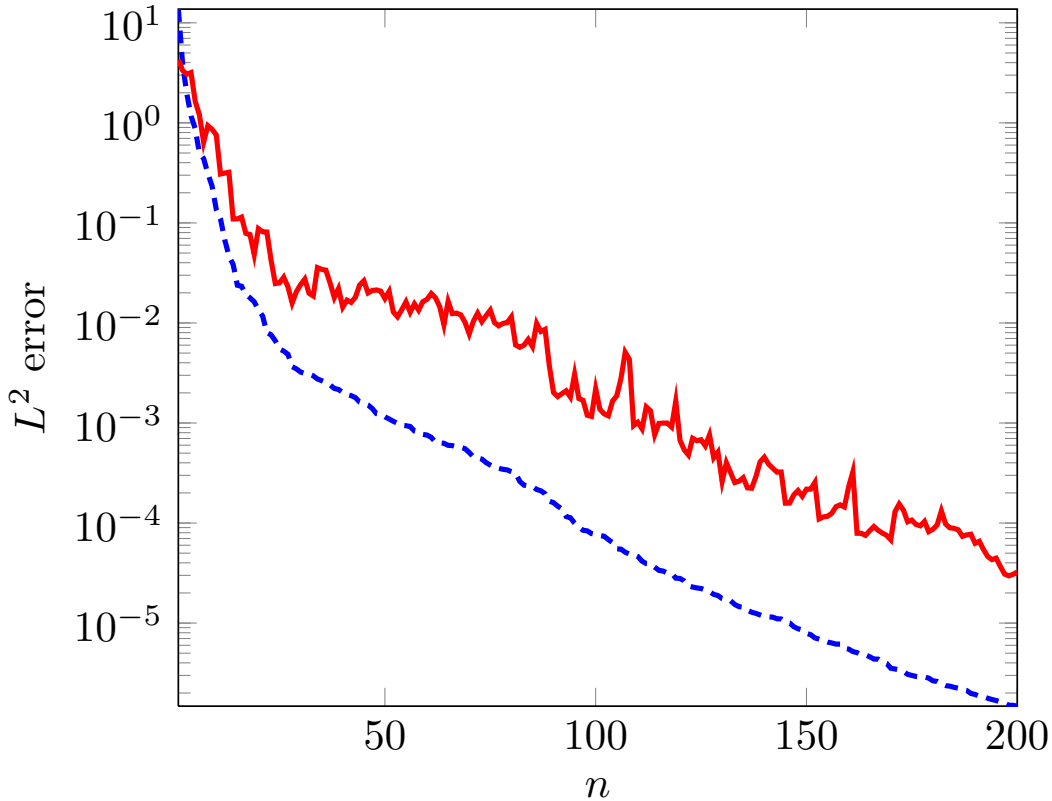
Discrete versions

Again, DEIM is just a pivoted *LU* factorization (a discrete Leja sequence) once we have a reduced basis.

In order to compute Lagrange interpolants $C_n(\mu)$, we *only* need $u(x_n, \mu)$.

So in this example, we can form an RBM approximation: requires spatial inner product information

And a DEIM approximation: requires only point-evaluations at x_n.



Approximation methods for parameterized functions

With respect to the parameter μ, interpolatory methods are non-intrusive, only requiring interrogation of legacy simulation codes.

Broadly speaking, there are non-adaptive (linear) approximation methods, and adaptive (nonlinear) approximation methods.

Non-adaptive methods are simple to implement:

- choosing a basis and a (poised) collection of nodes yields a Lagrange interpolation formulation
- reconstruction coefficients explicitly computed without data
- "Holy grail": balance of curse of dimensionality and blessing of smoothness. Error $\sim \mathcal{O}\left(N^{-s/d}\right)$
- Hermite-type (gradient) interpolation, least-squares, minimum-norm etc. are simple generalizations of similar procedures

Adaptive approximation methods

Non-adaptive methods usually perform poorly compared to adaptive methods.

Adaptive methods are generally harder:

- interpolation nodes, basis functions, *and* reconstruction conditions may depend on data
- cardinal Lagrange interpolants depend on functionals of the data
- greedy schemes are among the few computationally feasible approaches
- special cases: PCA, RBM, EIM (discrete: SVD, QR, LU)
- "Holy grail": error decay commensurate with n width.

Adaptive approximation methods

Non-adaptive methods usually perform poorly compared to adaptive methods.

Adaptive methods are generally harder:

- interpolation nodes, basis functions, *and* reconstruction conditions may depend on data
- cardinal Lagrange interpolants depend on functionals of the data
- greedy schemes are among the few computationally feasible approaches
- special cases: PCA, RBM, EIM (discrete: SVD, QR, LU)
- "Holy grail": error decay commensurate with n width.

Thank you!