Fast construction of efficient cut cell quadratures

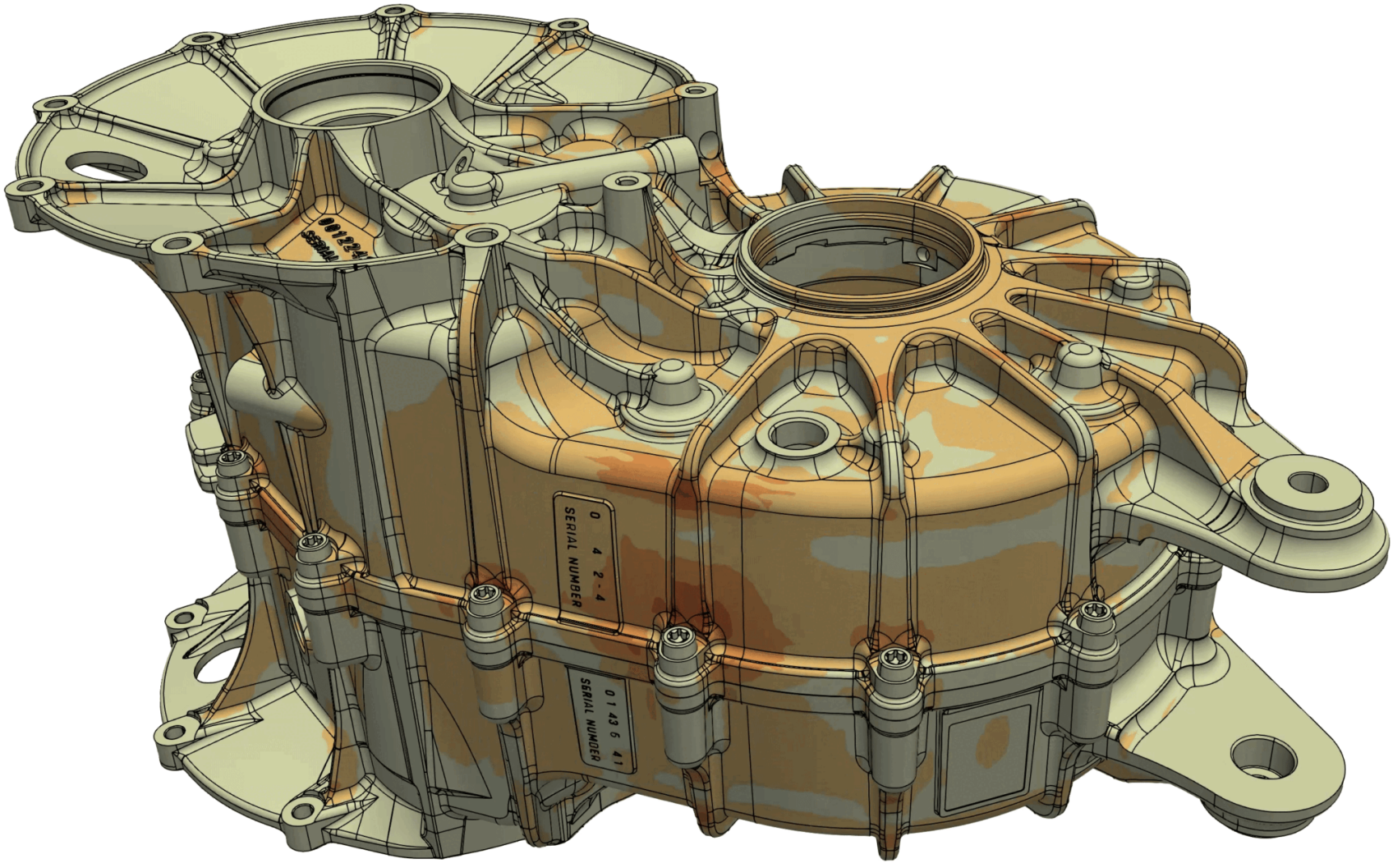
Samuel F. Potter

Robert McNeel & Associates
Seattle, WA

Who am I?

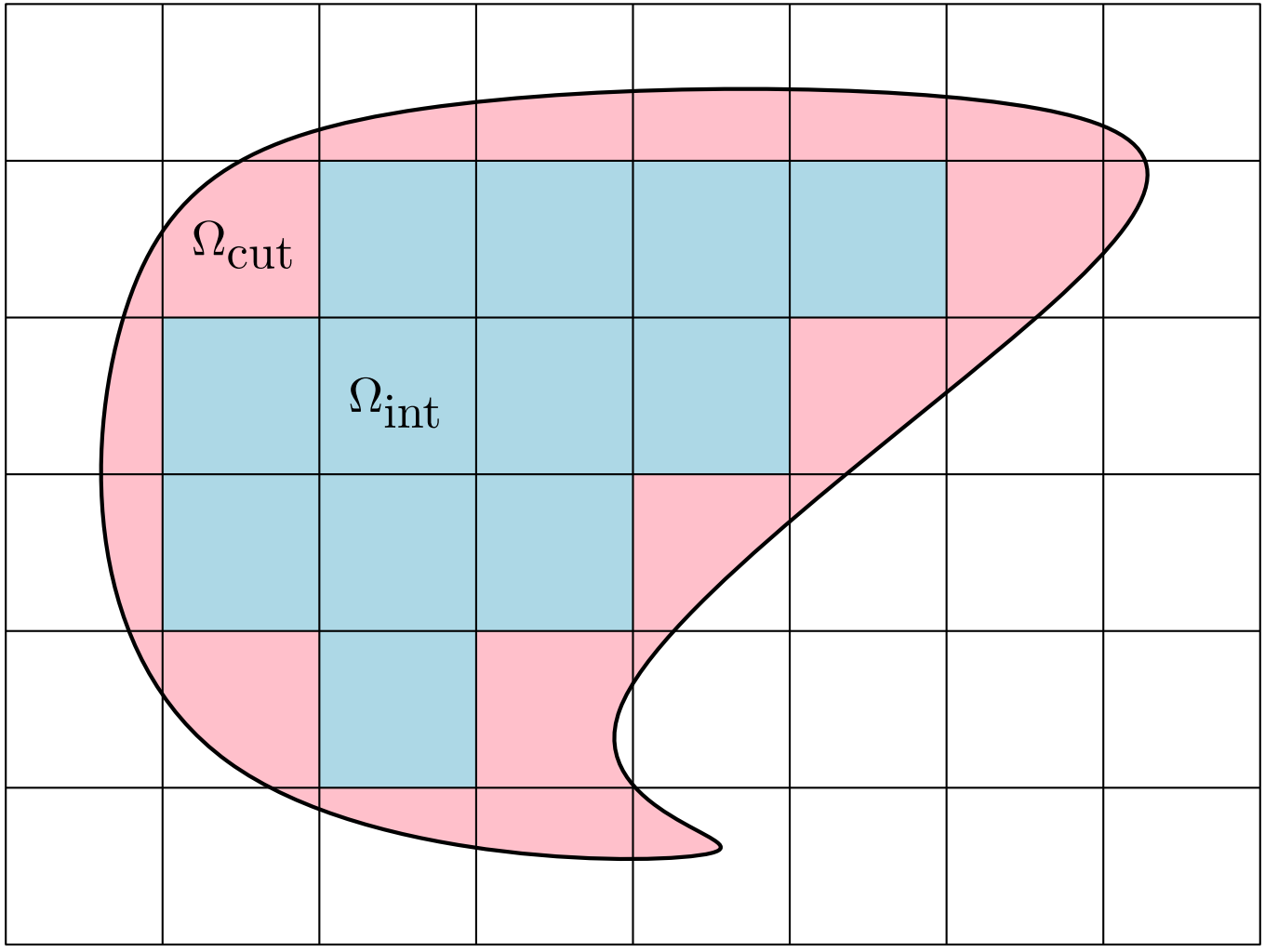
- Me: https://sampotter.github.io
- Currently employed by Robert McNeel & Associates (makers of **Rhinoceros 3D** (www.rhino3d.com)
- Working on modernizing their CAD kernel: faster, more accurate, more robust...
- **Previously**: research scientist at Coreform, LLC (**this talk**)
- **Previously**: CI at NYU w/ Leslie Greengard
- **Previously**: CS PhD from UMD w/ Masha Cameron

Coreform



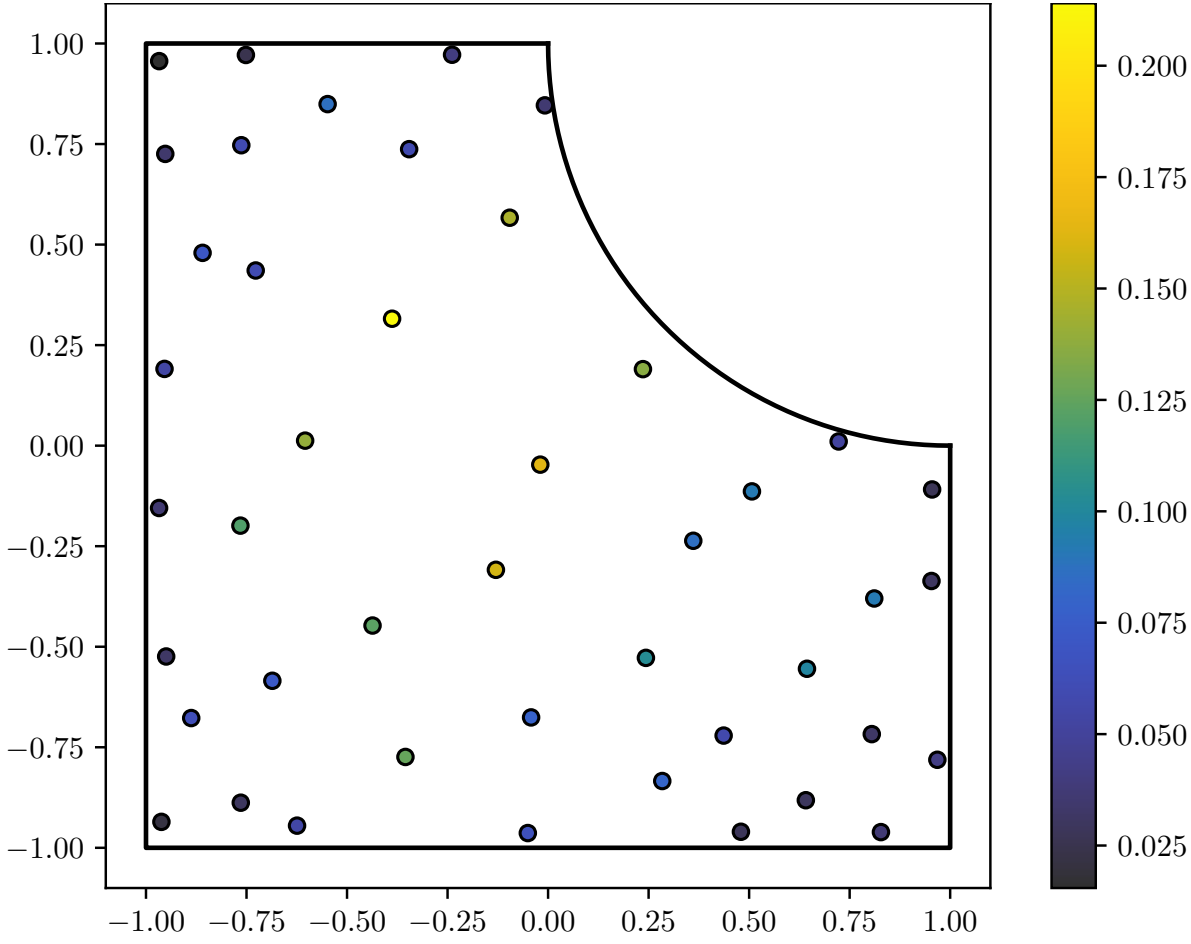
What's this talk about?

Useful algorithms for on-the-fly construction of "Gaussian"-ish cut cell quadratures...



What's this talk about?

Useful algorithms for on-the-fly construction of "Gaussian"-ish cut cell quadratures...



The beast

- Choose basis $\{\phi_\alpha\}$ (α a multi-index) for multivariate space of polynomials $\mathbb{P}$
- For order n quadrature, find weights and nodes satisfying:

$$\sum_{i=1}^{n} w_i \phi_\alpha(x_i) = \int_\Omega \phi_\alpha, \qquad \forall \alpha$$

- **Nonlinear, underdetermined**: can solve with Gauss-Newton with good initial guess
- **Linear** if node positions are fixed (**but how to choose nodes?**)

The efficiency of a quadrature rule

Definition

Number of **degrees of freedom** of $\mathcal{Q}_n$ is:

$$\mathrm{dof}(\mathcal{Q}_n) = (d+1)n \qquad \text{(for } w_i \text{ and } x_i\text{)}$$

and the **efficiency** of $\mathcal{Q}_n$ by:

$$\mathrm{eff}(\mathcal{Q}_n) = \frac{\dim \mathbb{P}_{m,d}}{\mathrm{dof}(\mathcal{Q}_n)}$$

where $0 \leq \mathrm{eff}(\mathcal{Q}_n) \leq 1$.

- Gauss-Legendre: $\mathrm{eff} = 1$
- Clenshaw-Curtis: $\mathrm{eff} = 1/2$
- Tensor Gauss-Legendre: $\mathrm{eff} \sim 2^d/(d+1)! \sim 0$ as $n \to \infty$

Existence and stability of positive quadratures

Theorem (Tchakaloff)

For $\Omega \subseteq \mathbb{R}^d$ compact, exists ***positive*** *quadrature with order not greater than $m = \dim \mathbb{P}$ with nodes in Ω.*

Theorem (Huybrechs 2009)

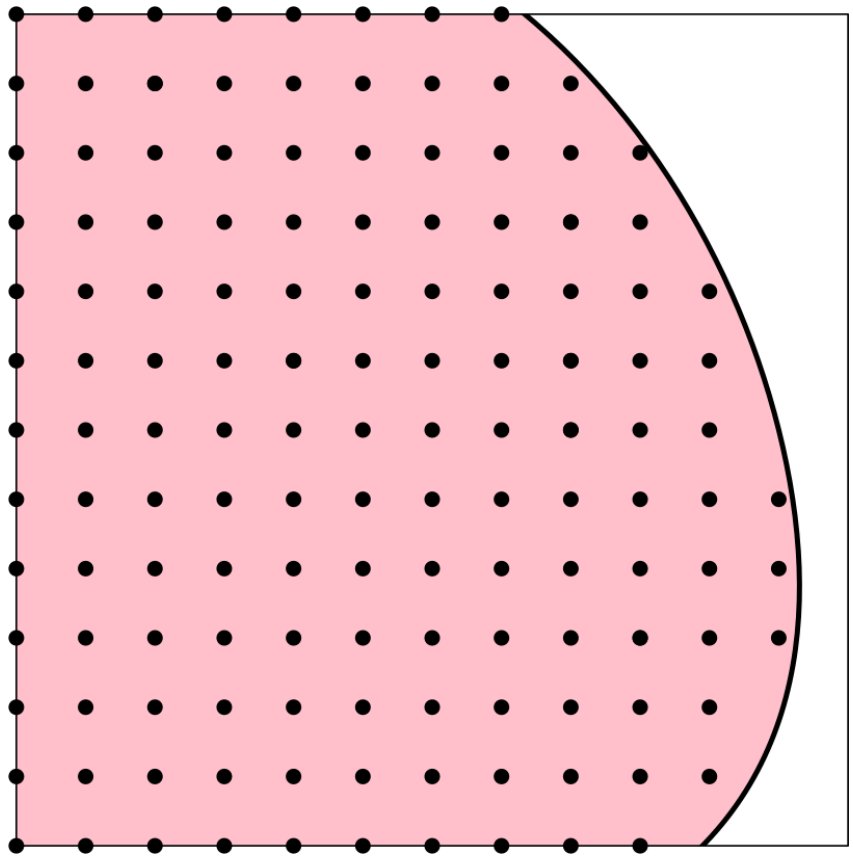
Let $\kappa(\mathcal{Q}_n) = \sum_i |w_i|$, let f a function be integrated, and $\tilde{f}$ an approximation of f with $|f - \tilde{f}| \leq \epsilon$ pointwise. Then:

$$\left| \sum_i w_i f(x_i) - \sum_i w_i \tilde{f}(x_i) \right| \leq \epsilon \kappa(\mathcal{Q}_n).$$

Strictly larger *if $\mathcal{Q}_n$ isn't positive.*

Discretizing the cut cell

1. Map cut cell bounding box onto $[-1, 1]^d$
2. Discretize the box into a uniform, isotropic grid with grid spacing $h > 0$
3. Discard all points that lie strictly outside Ω



Discretizing the cut cell

- Nodes: $x_1, \ldots, x_n$
- Basis matrix for moment-matching equations:

$$V = \begin{bmatrix} \phi_1(x_1) & \cdots & \phi_m(x_1) \\ \vdots & \ddots & \vdots \\ \phi_1(x_n) & \cdots & \phi_m(x_n) \end{bmatrix} \tag{1}$$

- Moment vector:

$$I \in \mathbb{R}^m \quad \text{st} \quad I_\alpha = \int_\Omega \phi_\alpha(x) dx \tag{2}$$

- Linear moment-matching (**underdetermined**):

$$V^\top w = I \tag{3}$$

Quadrature toolbox: overview

Goal: using our oversampled grid, find sparse, nonnegative solution, subset of positive weights. Lots of approaches:

- Least squares
- Steinitz elimination
- Nonnegative least squares
- Basis pursuit (won't cover, equivalent to NNLS)
- Xiao-Gimbutas elimination
- Ryu-Boyd (linear prog.)

Quadrature toolbox: least squares rules

Least squares quadrature:

- Solve:

$$\text{minimize} \quad \frac{1}{2}\|V^{\mathsf{T}} w - I\|_2^2$$

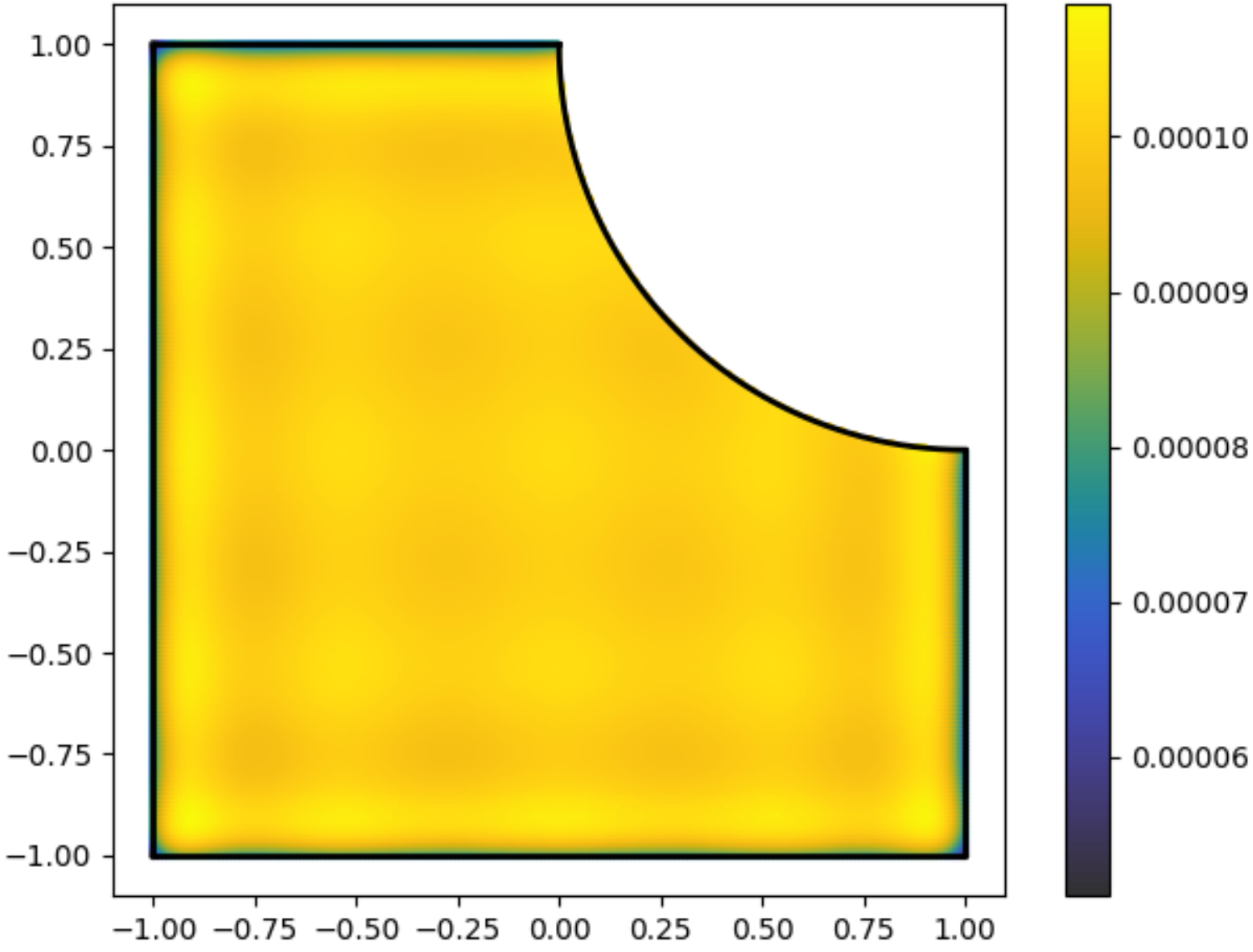
  using LU decomposition or QR.

Theorem (Existence: Huybrechs 1D, me *n*D)

There exists a positive integer n_0 such that for $n \geq n_0$ the resulting least squares quadrature is positive.

Problem: $w > 0$ at all nodes.

Quadrature toolbox: least squares rules



(For 32,121 points...)

Quadrature toolbox: Steinitz elimination

Can we remove unnecessary quadrature nodes?
Yes, look in the nullspace of V^T to find shift Δw so that:

$$V^\mathsf{T}(w + \Delta w) = I, \qquad w + \Delta w \geq 0. \tag{4}$$

Theorem (Steinitz)

Let $\mathcal{Q}_n$ be a positive quadrature that integrates $\mathbb{P}$ on Ω such that $n \geq m = \dim \mathbb{P}$. Then there exists a positive quadrature of order m w/ subset of nodes of $\mathcal{Q}_n$.

Proof.

Proof gives an algorithm now called Steinitz elimination. □

Consequence: positive least squares rule exists $\Longrightarrow$ order m rule exists, proof is constructive.

Quadrature toolbox: NNLS rules

Nonnegative least squares quadrature:

- Solve:

$$\begin{aligned} \text{minimize} \quad & \frac{1}{2}\|V^{\mathsf{T}} w - I\|_2^2 \\ \text{subject to} \quad & w \geq 0 \end{aligned}$$

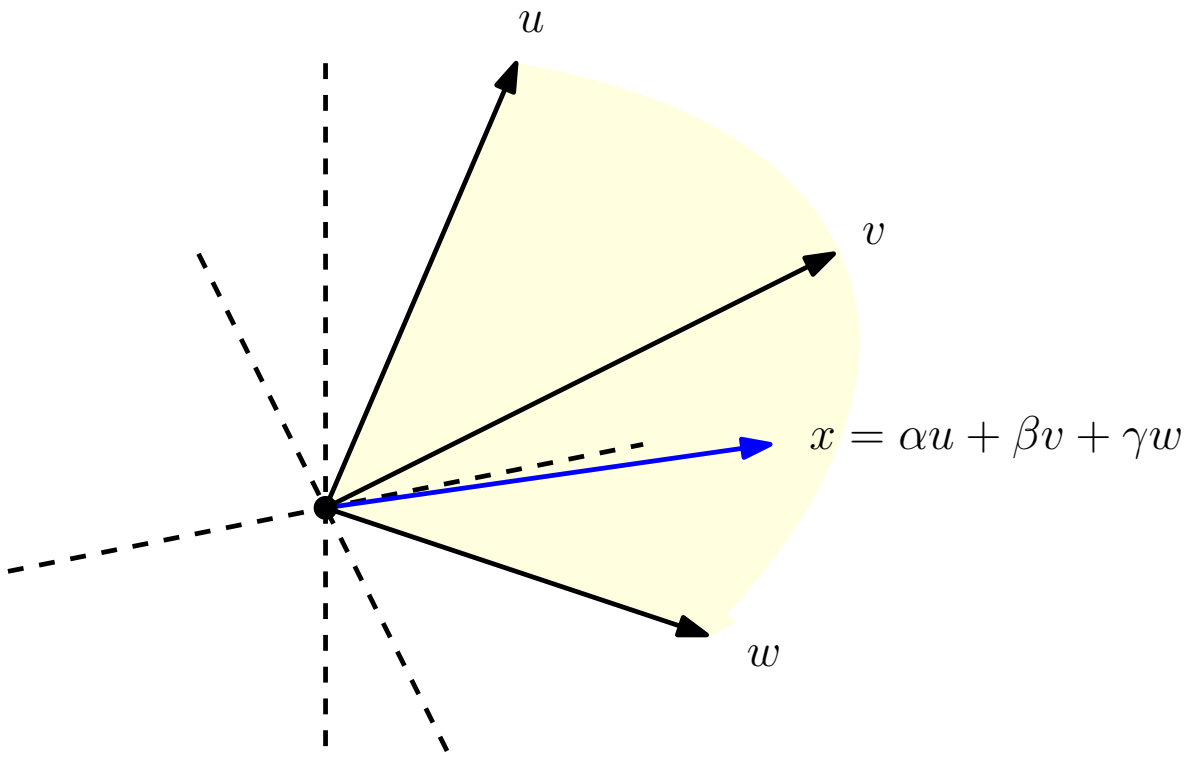
Undeterdetermined convex optimization problem, can have infinite number of solutions.

Can always find m-sparse solution, but solution may not have zero residual

Quadrature toolbox: NNLS rules

Theorem (Carathéodory)

If $x \in cone(A) \subseteq \mathbb{R}^m$, then x can be written as the conical combination of m points in A.



Note: $V^\top w = I$ is a conic combination!

Quadrature toolbox: NNLS rules

Problem: solution biased by algorithm when nonunique, need to use right algorithm.

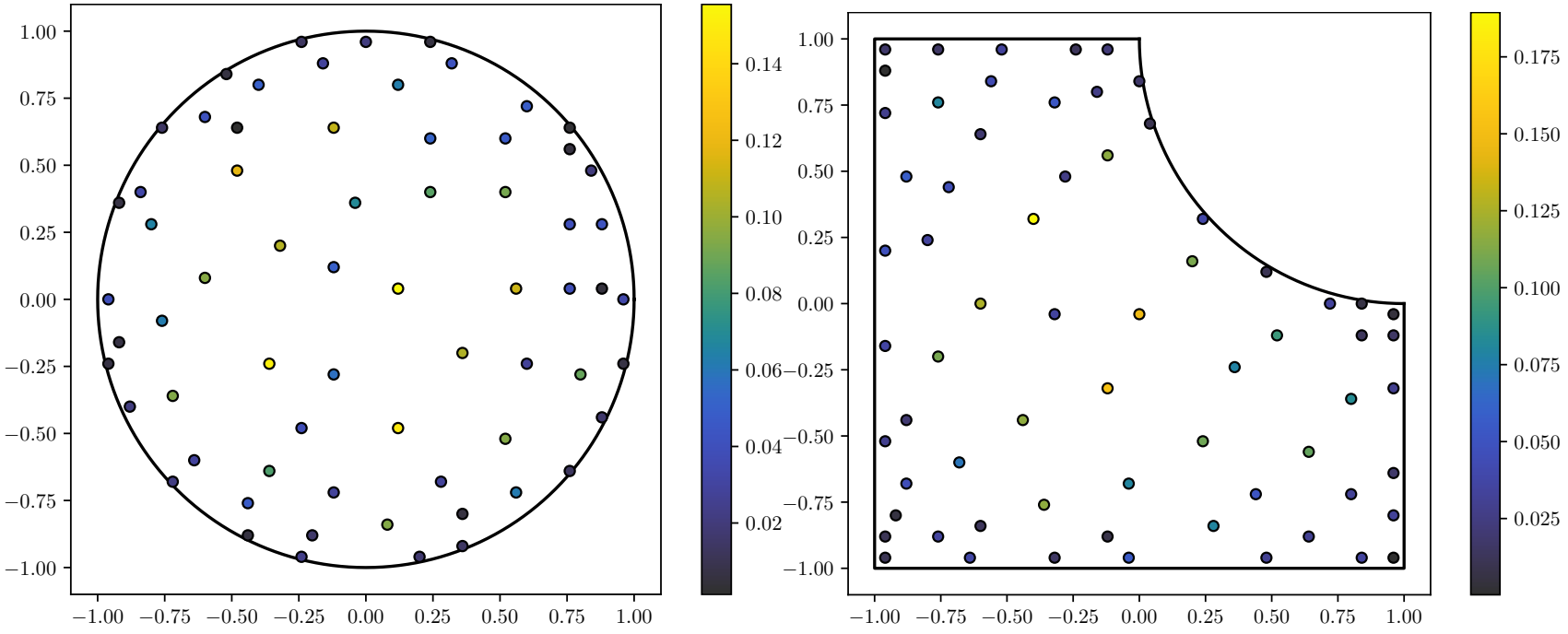
Algorithm NNLS:

1. Use the Lawson-Hanson algorithm (i.e. primal active set method for quadratic program with linear inequality constraints) to solve NNLS.
2. Return the subset of weights and nodes for which $w_i > 0$.

Typically relaxes one quadrature node at a time, converging to an extreme point on the feasible set.

Quadrature toolbox: NNLS rules

What does this look like?



Quadrature toolbox: Xiao-Gimbutas elimination

A concern: In general, we expect the solution of either NNLS or BP to be M-sparse. This is only achieves:

$$\mathrm{eff}(\mathcal{Q}) = \frac{1}{d+1}. \tag{5}$$

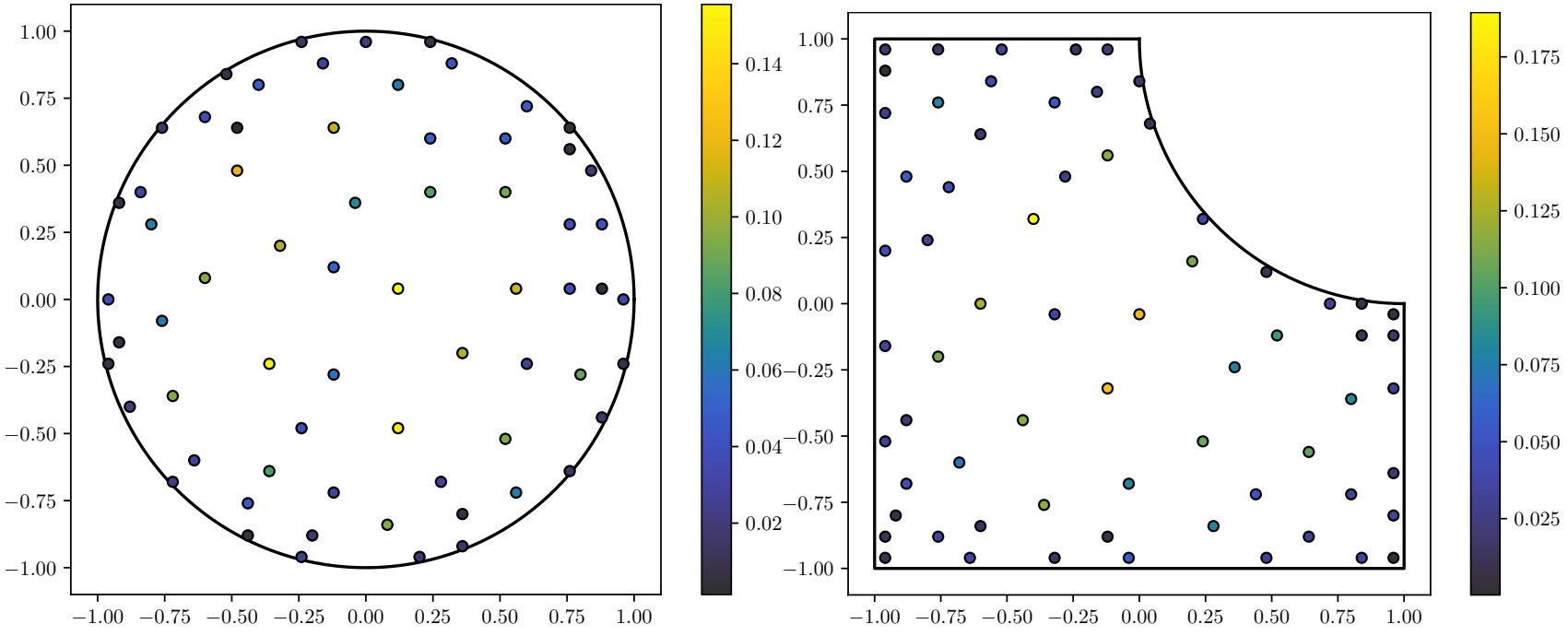
Nonlinear node elimination: Xiao and Gimbutas propose the following method for removing nodes from a quadrature which differs from Steinitz' approach.

Algorithm XG:

1. Compute $\sigma_j = w_j \sum_\alpha \phi_\alpha(x_j)^2$ for each j.
2. Delete node j from the quadrature.
3. Run Gauss-Newton on the result.

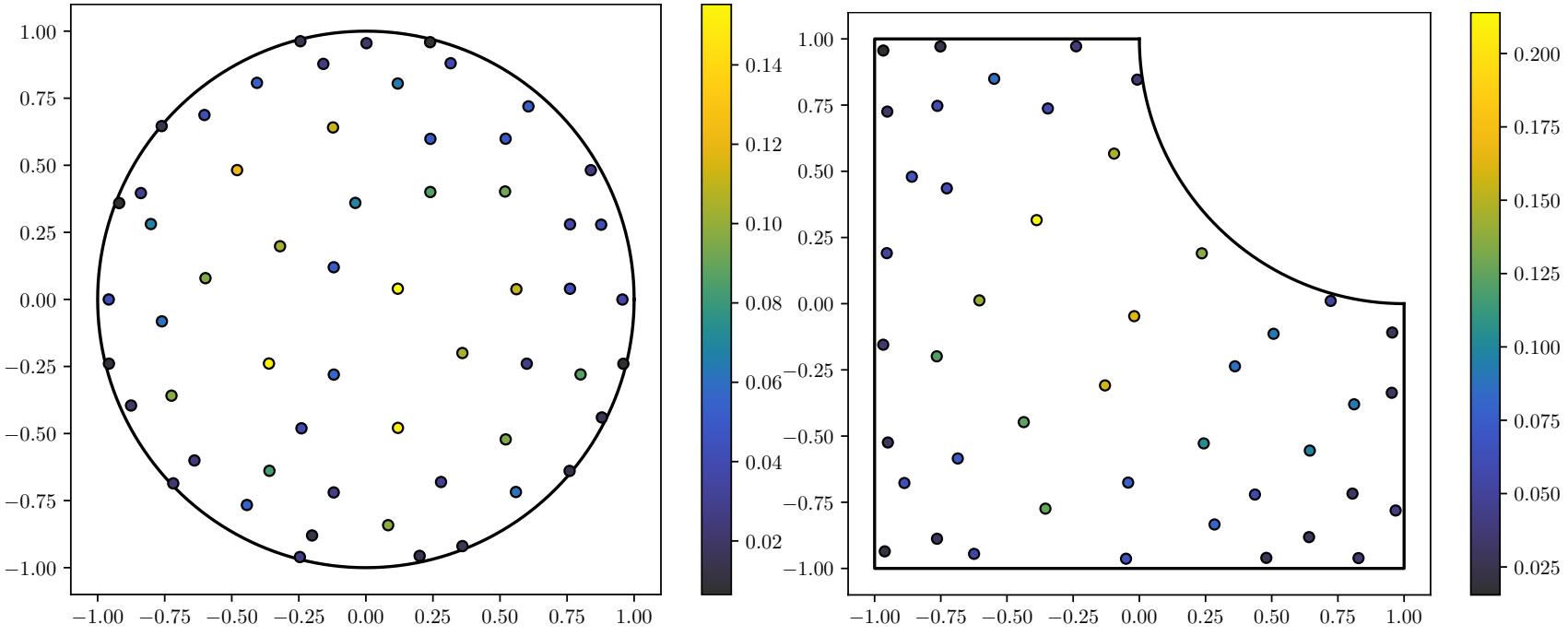
Quadrature toolbox: Xiao-Gimbutas elimination

What does this look like?



Quadrature toolbox: Xiao-Gimbutas elimination

What does this look like?



Quadrature toolbox: Ryu-Boyd

Ryu and Boyd consider the semi-infinite linear program:

$$\begin{aligned} \text{minimize}_{\mu} \quad & \int_{\Omega} r d\mu \\ \text{subject to} \quad & \int_{\Omega} \phi_\alpha d\mu = \int_{\Omega} \phi_\alpha(x) dx \qquad \forall \alpha \\ & \mu \geq 0 \end{aligned}$$

where optimization is over **all nonnegative measures supported on** Ω.

Crucial: function r is the "residual": free for the user to choose.

Ryu and Boyd prove that on $[-1, 1]$, with $r(x) = x^{2n}$ and $\phi_i(x) = x^i$ $(0 \leq i < 2n)$, Gaussian-Legendre quadrature is the unique sol'n of this problem.

Quadrature toolbox: Ryu-Boyd

Lay down uniform grid and discretize to get the LP:

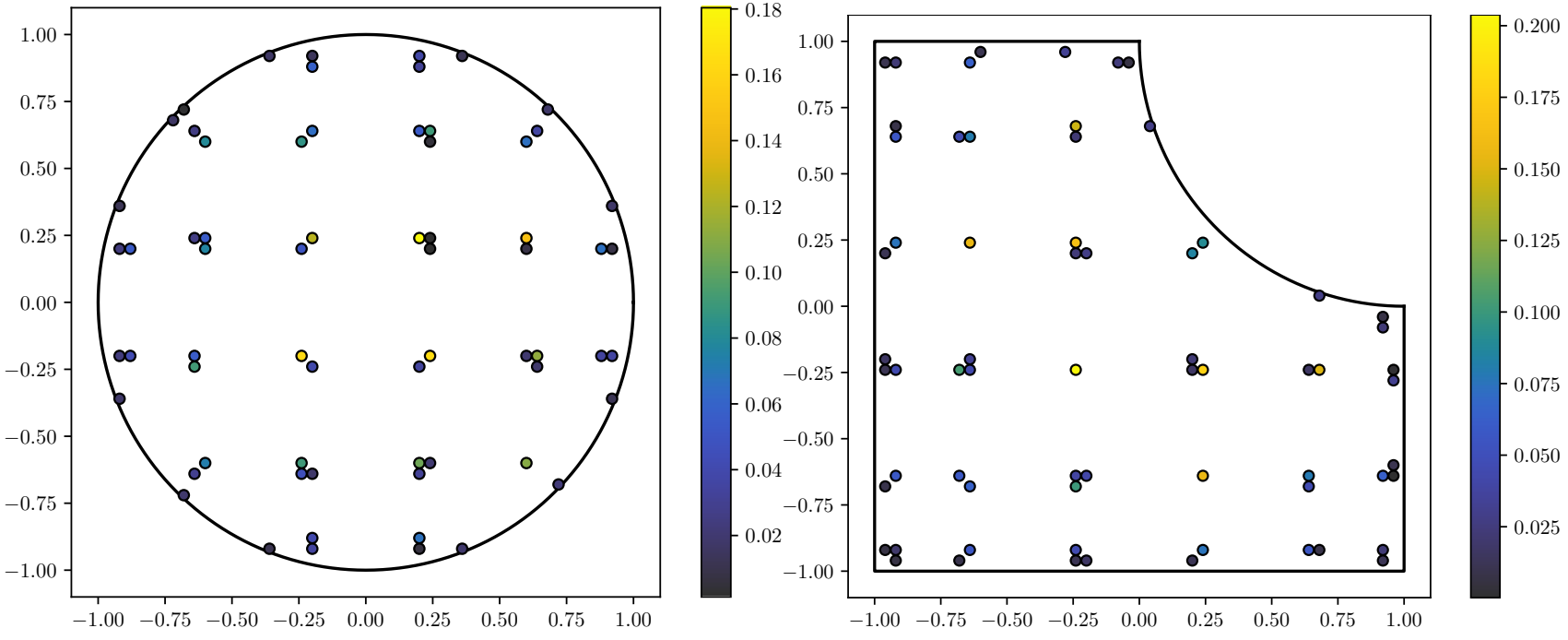
$$\begin{array}{lll} \text{minimize} & r^{\mathsf{T}} w & \\ \text{subject to} & V^{\mathsf{T}} w = I & \text{(RB)} \\ & w \geq 0. & \end{array}$$

where r is the "residual", a parameter.

We use $r(x, y) = x^{2n} + y^{2n}$ throughout, but choice of residual needs more work.

Quadrature toolbox: Ryu-Boyd

What happens if we throw RB at a black box LP solver?



Again, only produces eff$(\mathcal{Q}) = 1/(d+1)$ quadrature... but look at the nodes...

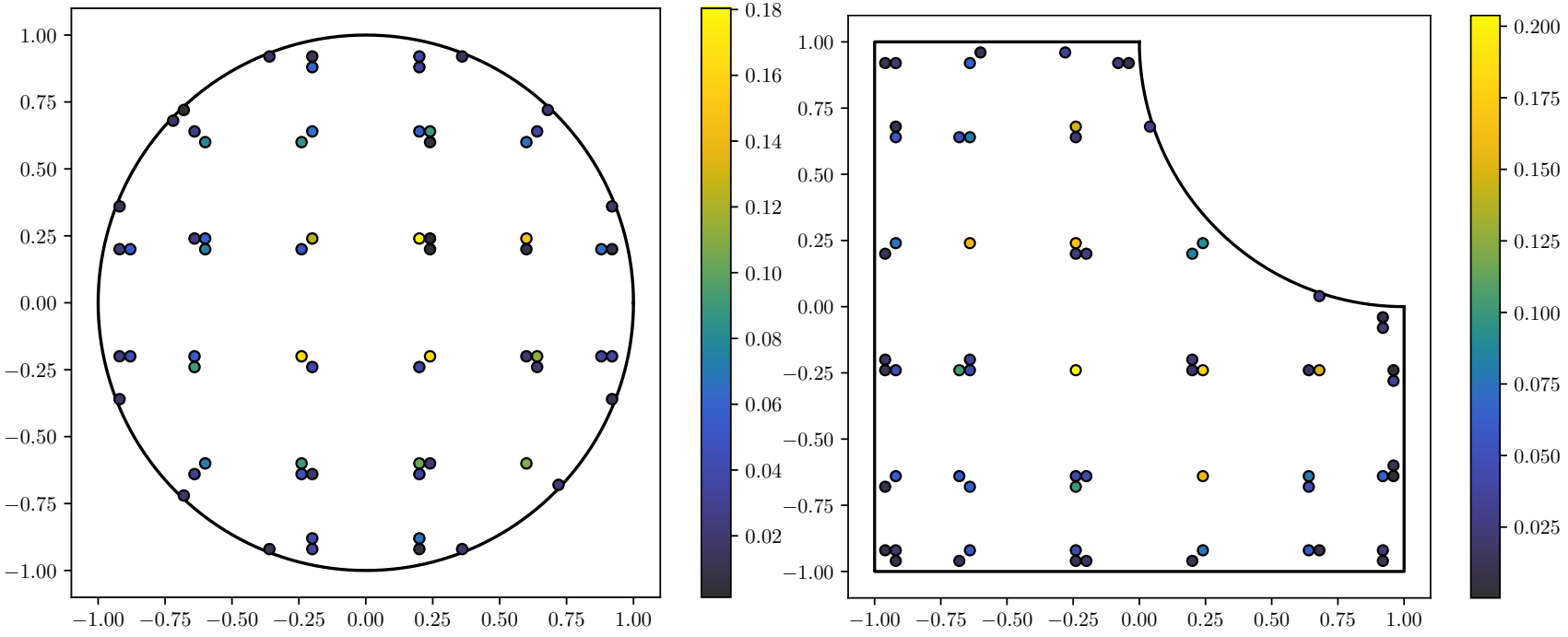
Quadrature toolbox: Ryu-Boyd

Algorithm RB:

- Solve RB using black-box LP solver.
- Identify clusters of points.
- Replace each cluster with a single quadrature point:
  - Let the new weight w_{clust} be the sum of the weights.
  - Let the new node be the convex combination of the old nodes using the normalized quadrature weights for each point, i.e. $x_{\text{clust}} = \sum_{j \in \text{clust}} (w_j / w_{\text{clust}}) x_j$.
- Optimize the "clustered" quadrature using Gauss-Newton.

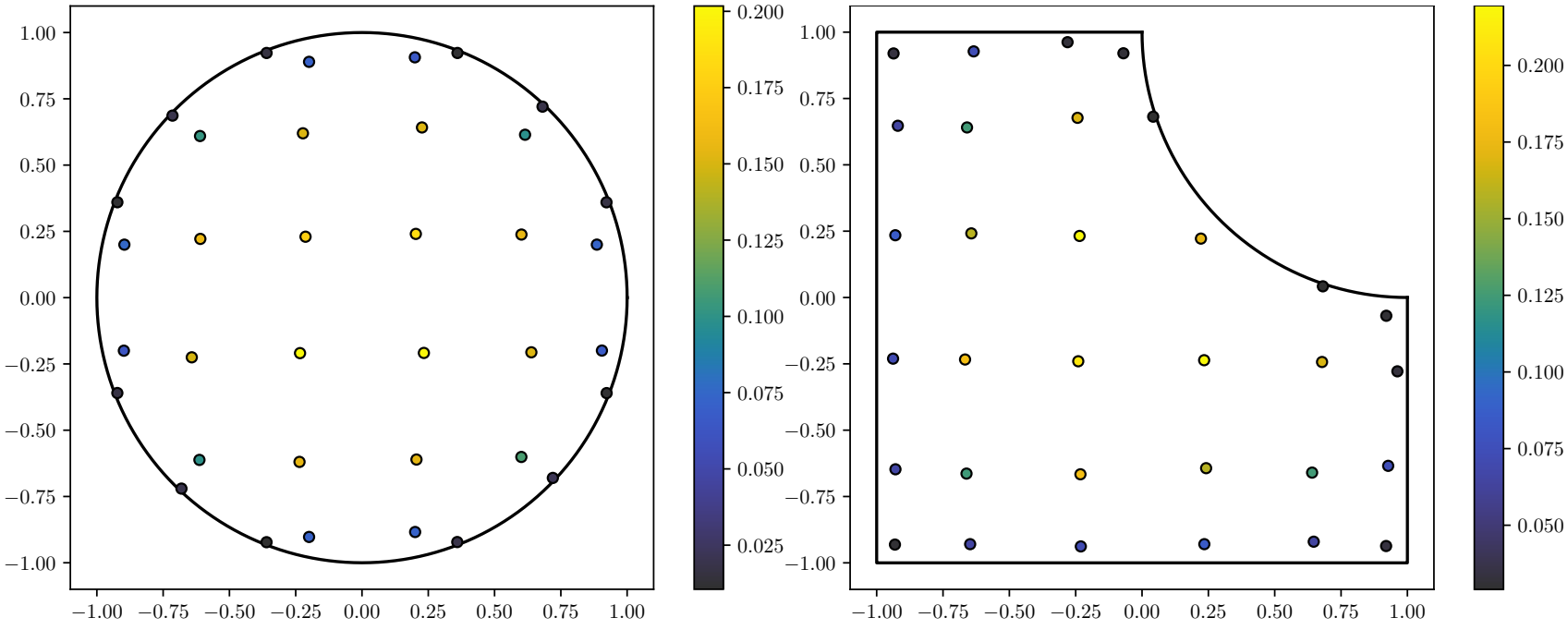
Quadrature toolbox: Ryu-Boyd

What does this look like?



Quadrature toolbox: Ryu-Boyd

What does this look like?



Quadrature toolbox: primal-dual RB

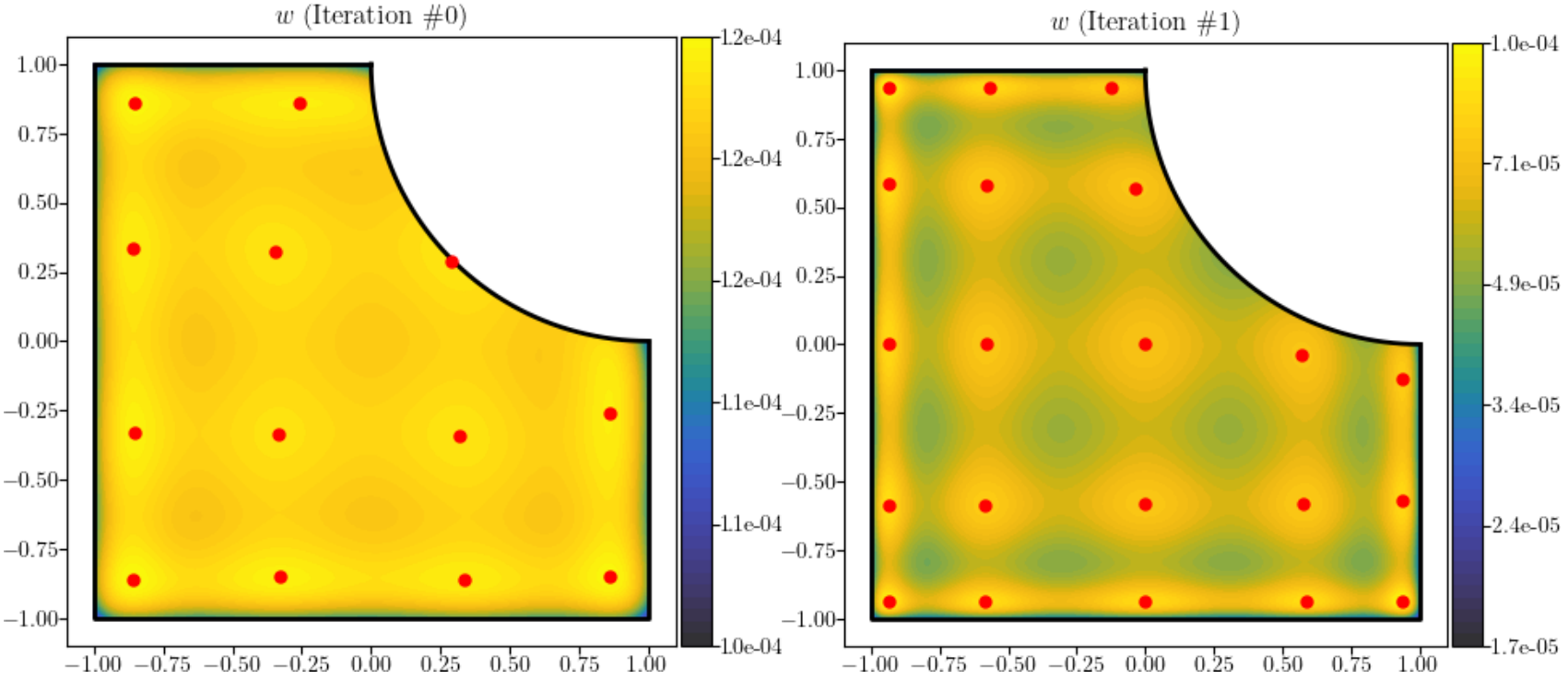
What if we look under the hood of our LP solver? Two standard approaches: **simplex** and **interior point**.

An interior point method for solving Ryu-Boyd is a **curious choice**, because by construction it produces iterates which have w_i for every i at each step!

But the result converges pointwise to an M-sparse solution...?

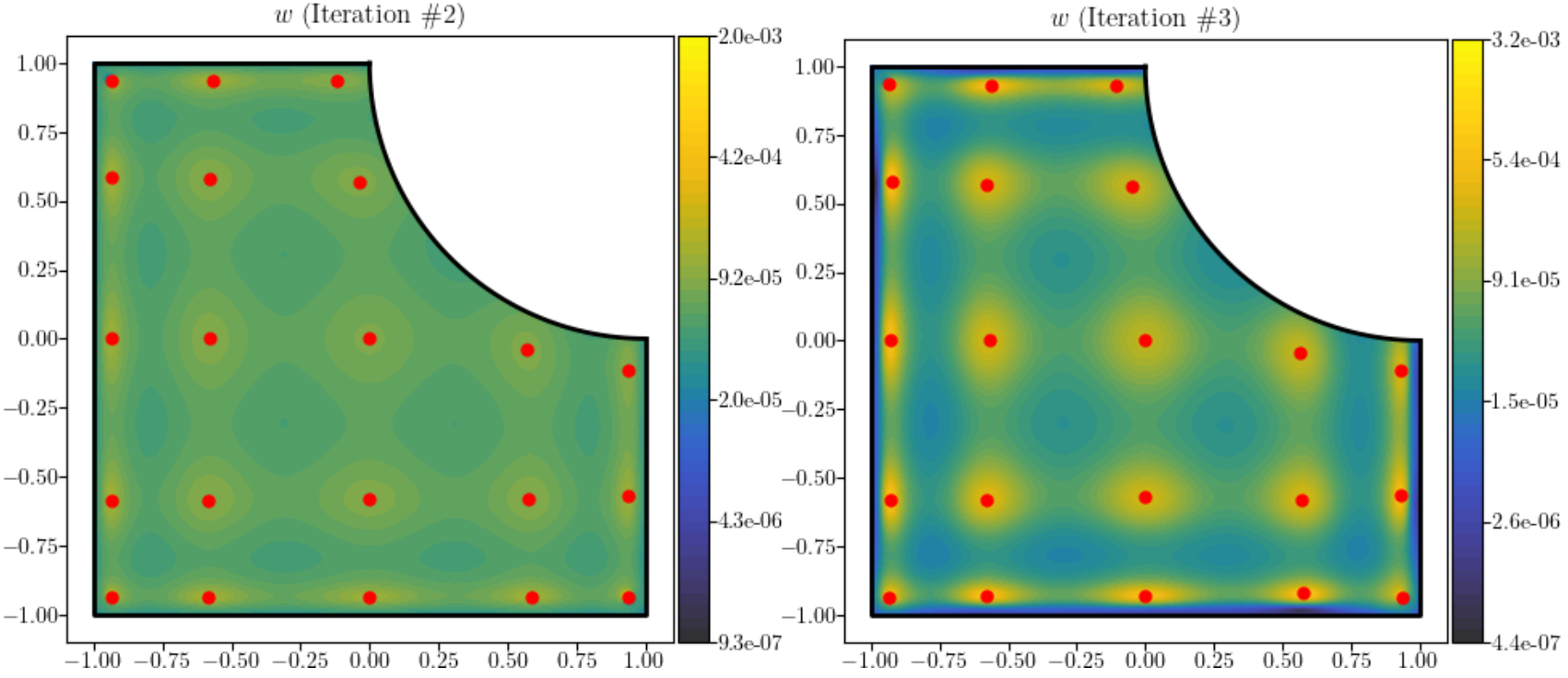
Quadrature toolbox: primal-dual RB

It works! And after only a single step. Just need to find local optima and re-optimize using Gauss-Newton after selection:

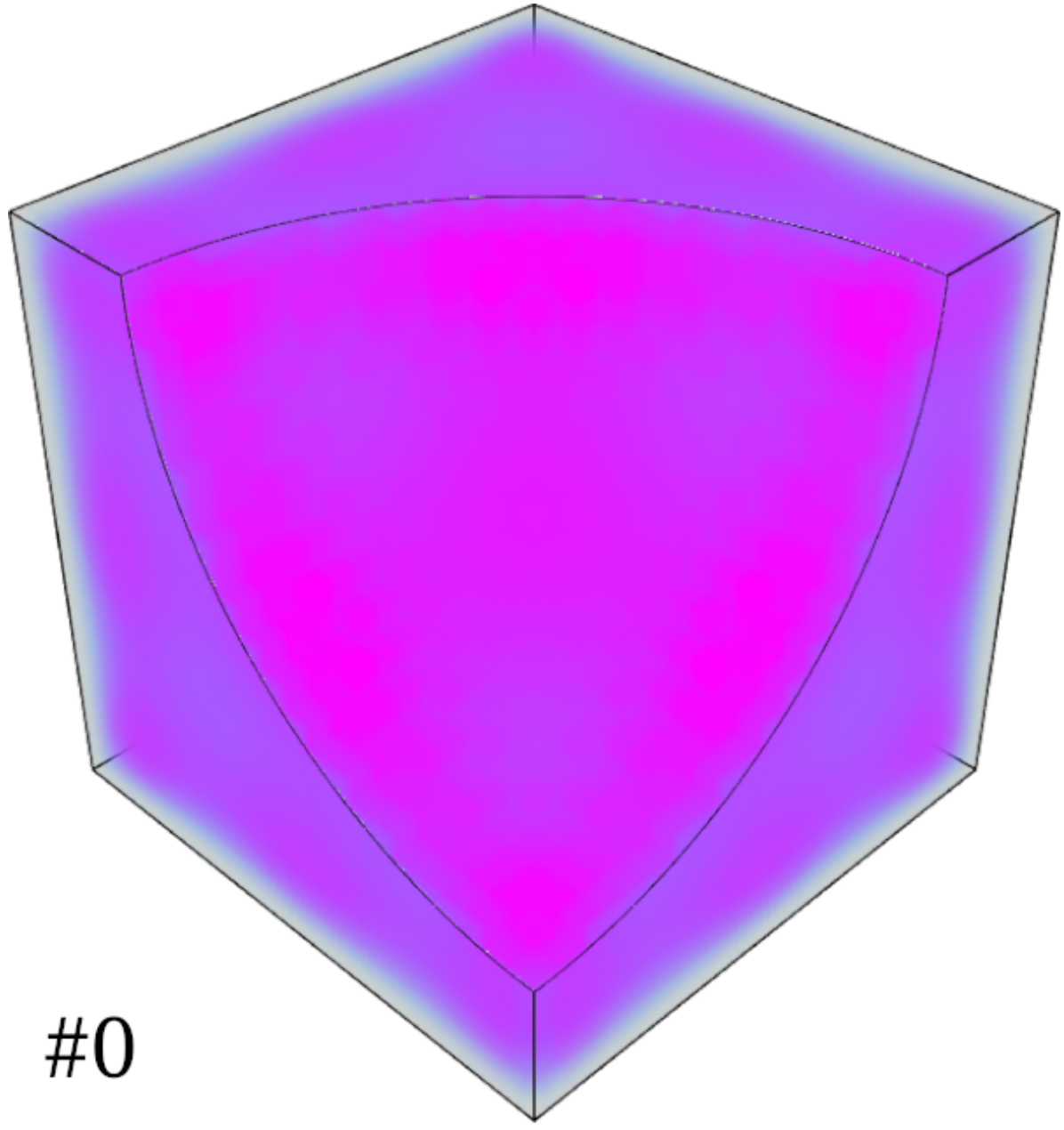


Quadrature toolbox: primal-dual RB

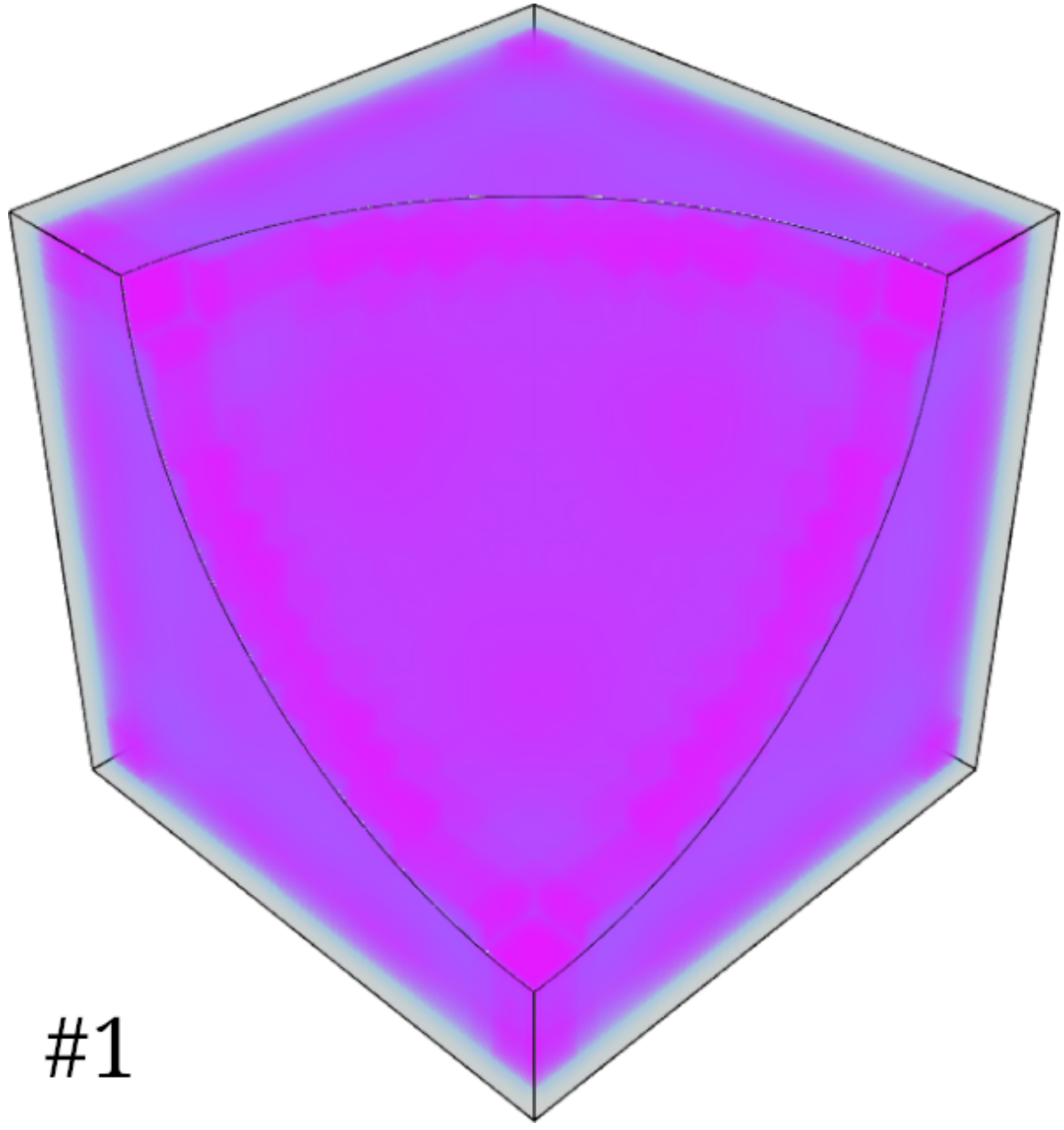
It works! And after only a single step. Just need to find local optima and re-optimize using Gauss-Newton after selection:



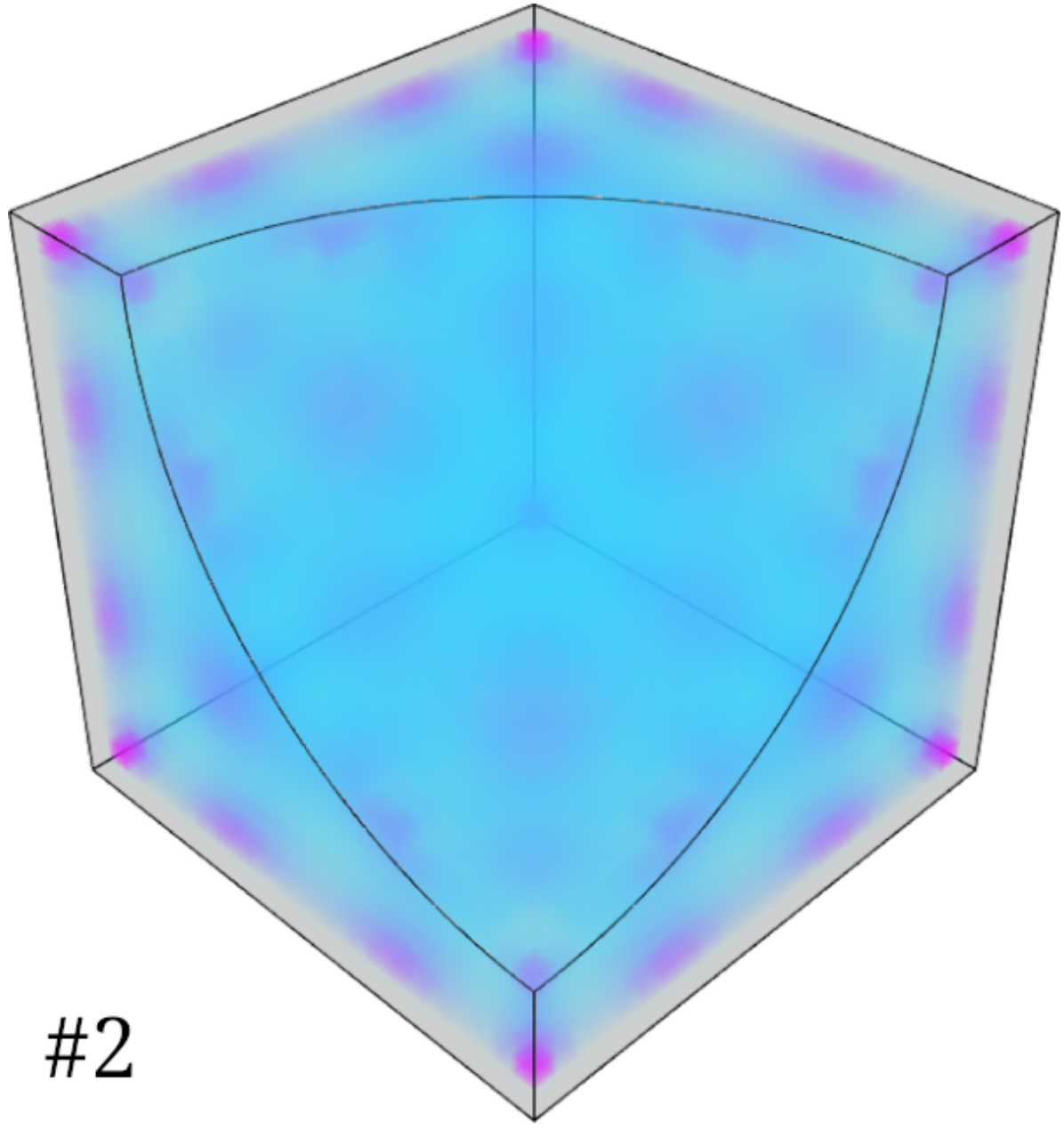
Quadrature toolbox: primal-dual RB



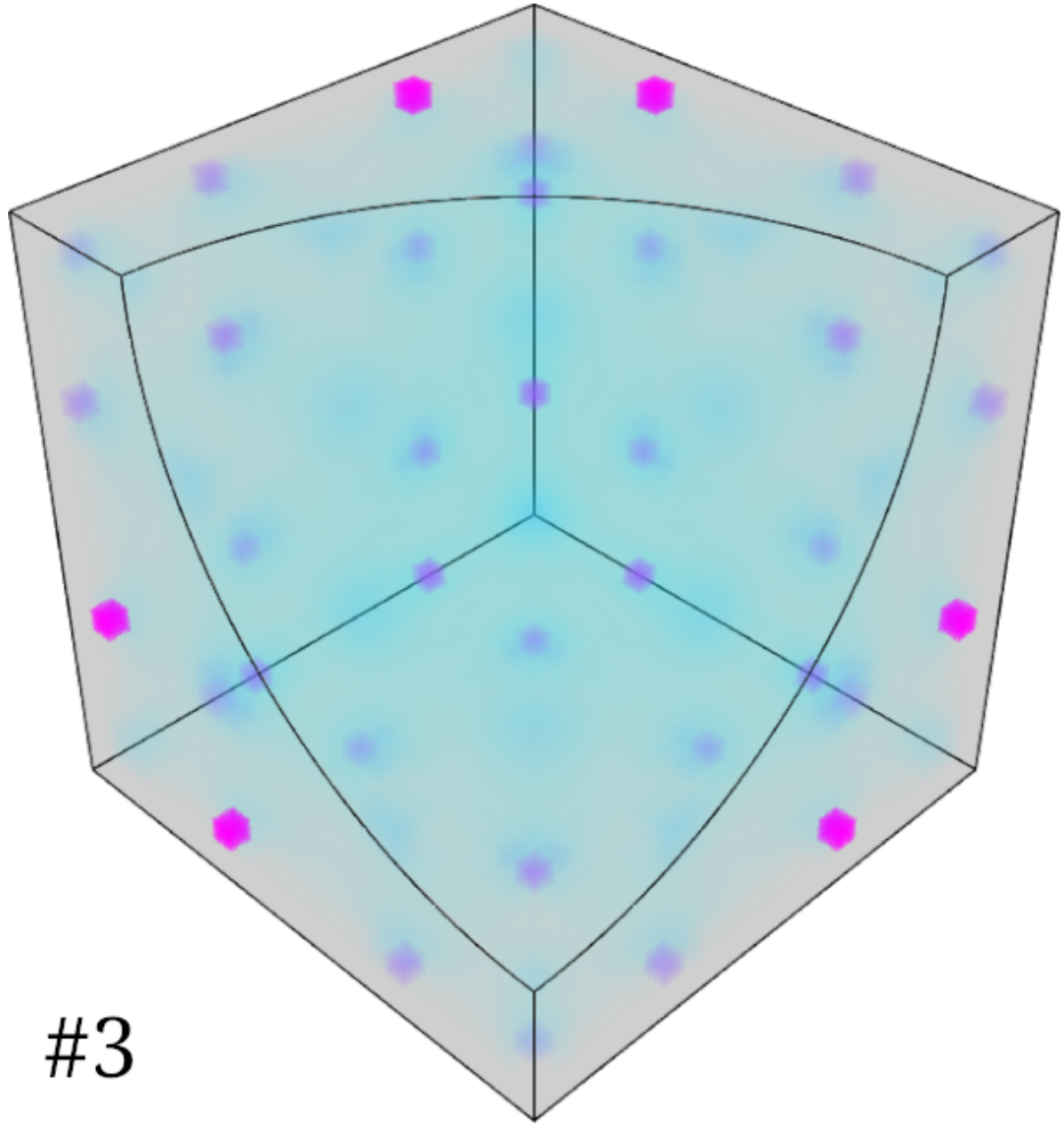
Quadrature toolbox: primal-dual RB



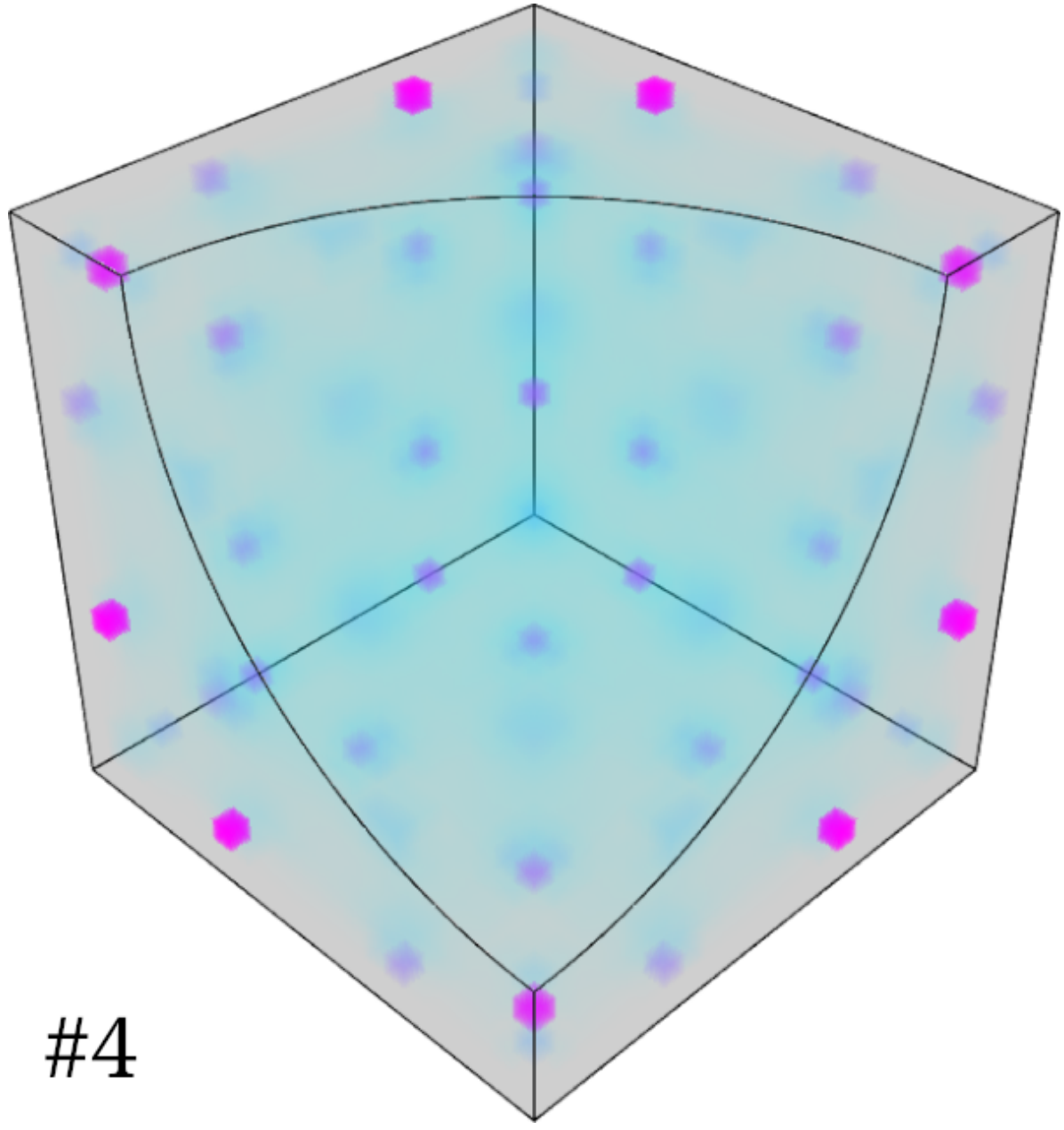
Quadrature toolbox: primal-dual RB



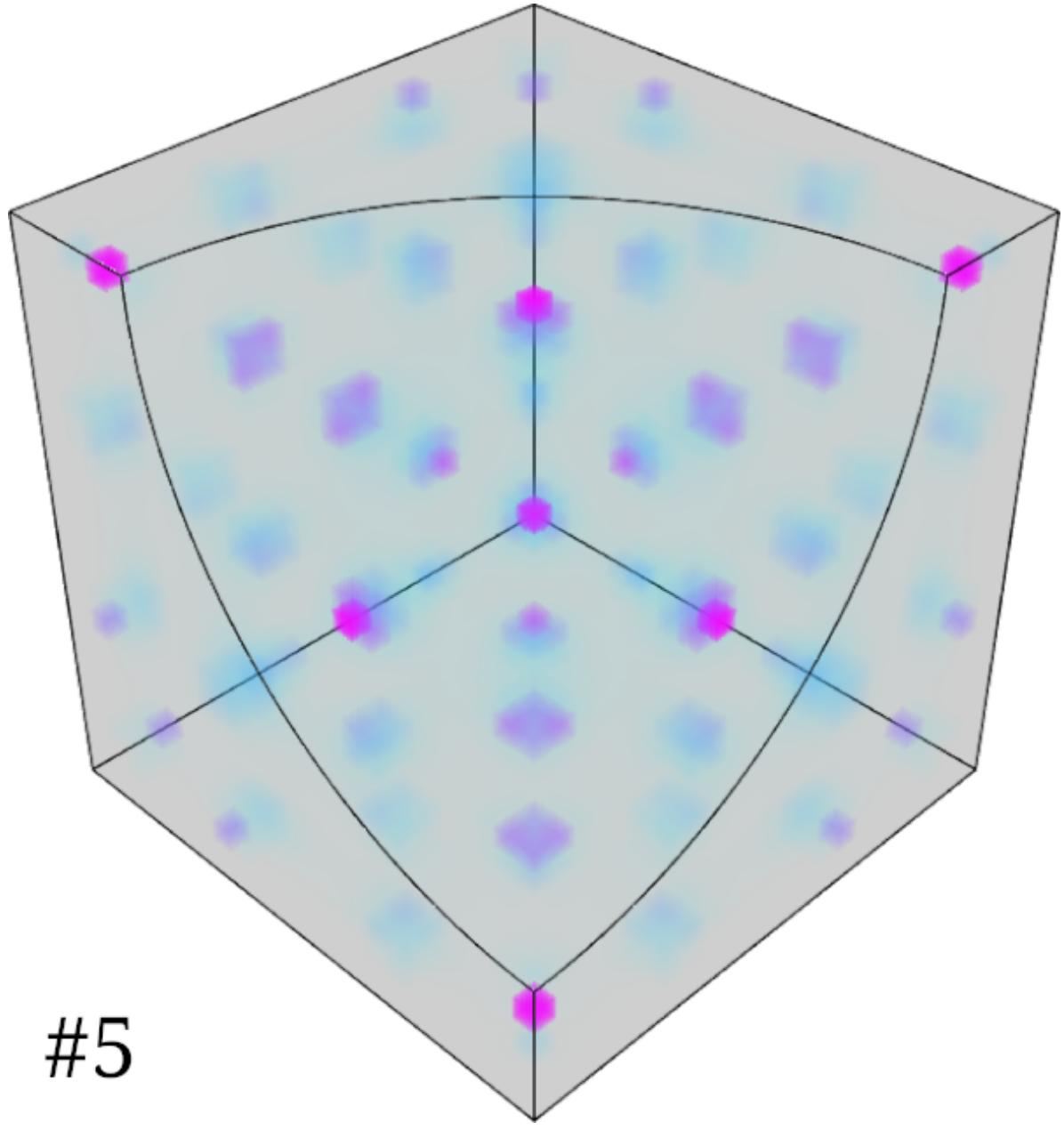
Quadrature toolbox: primal-dual RB



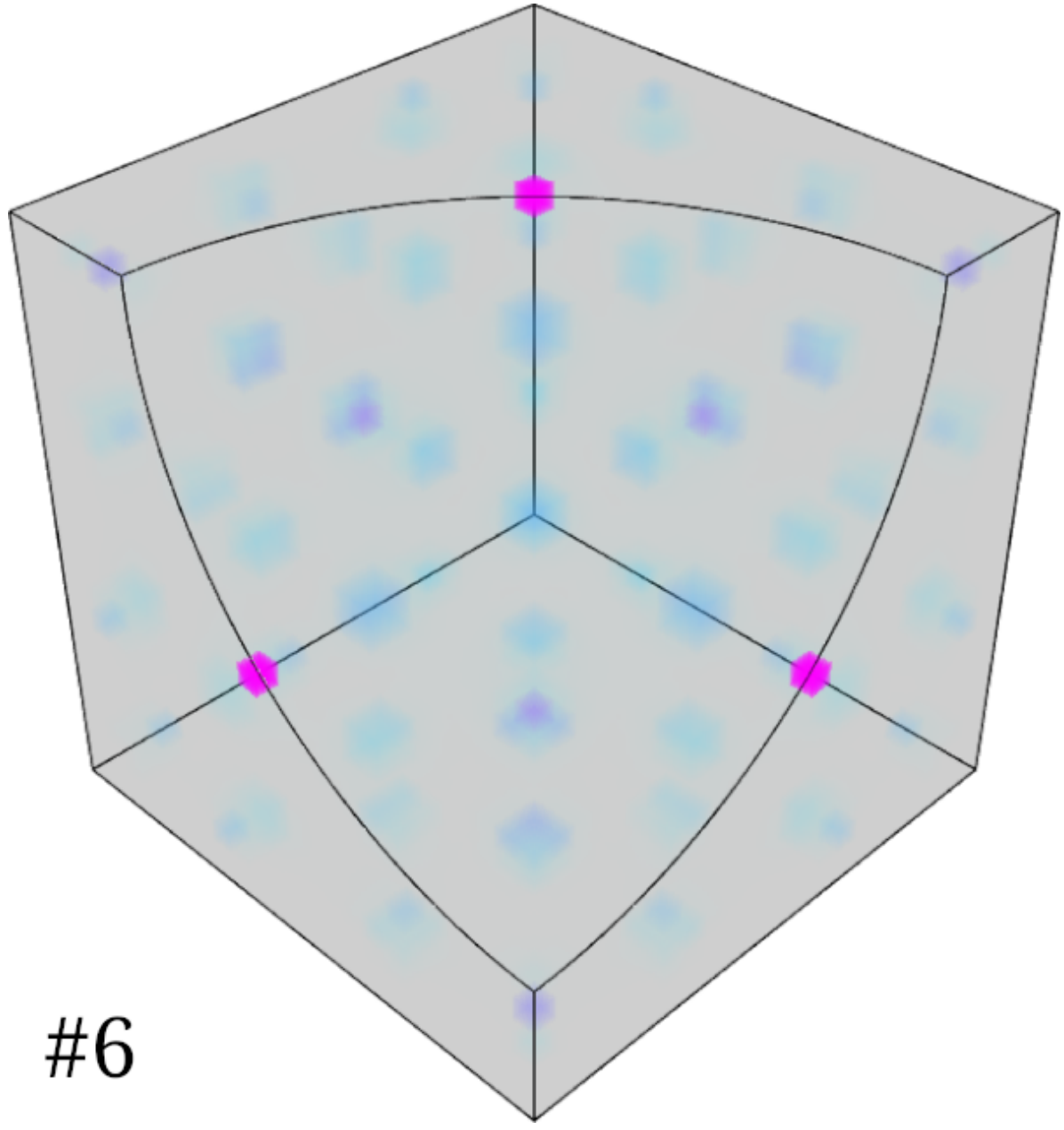
Quadrature toolbox: primal-dual RB



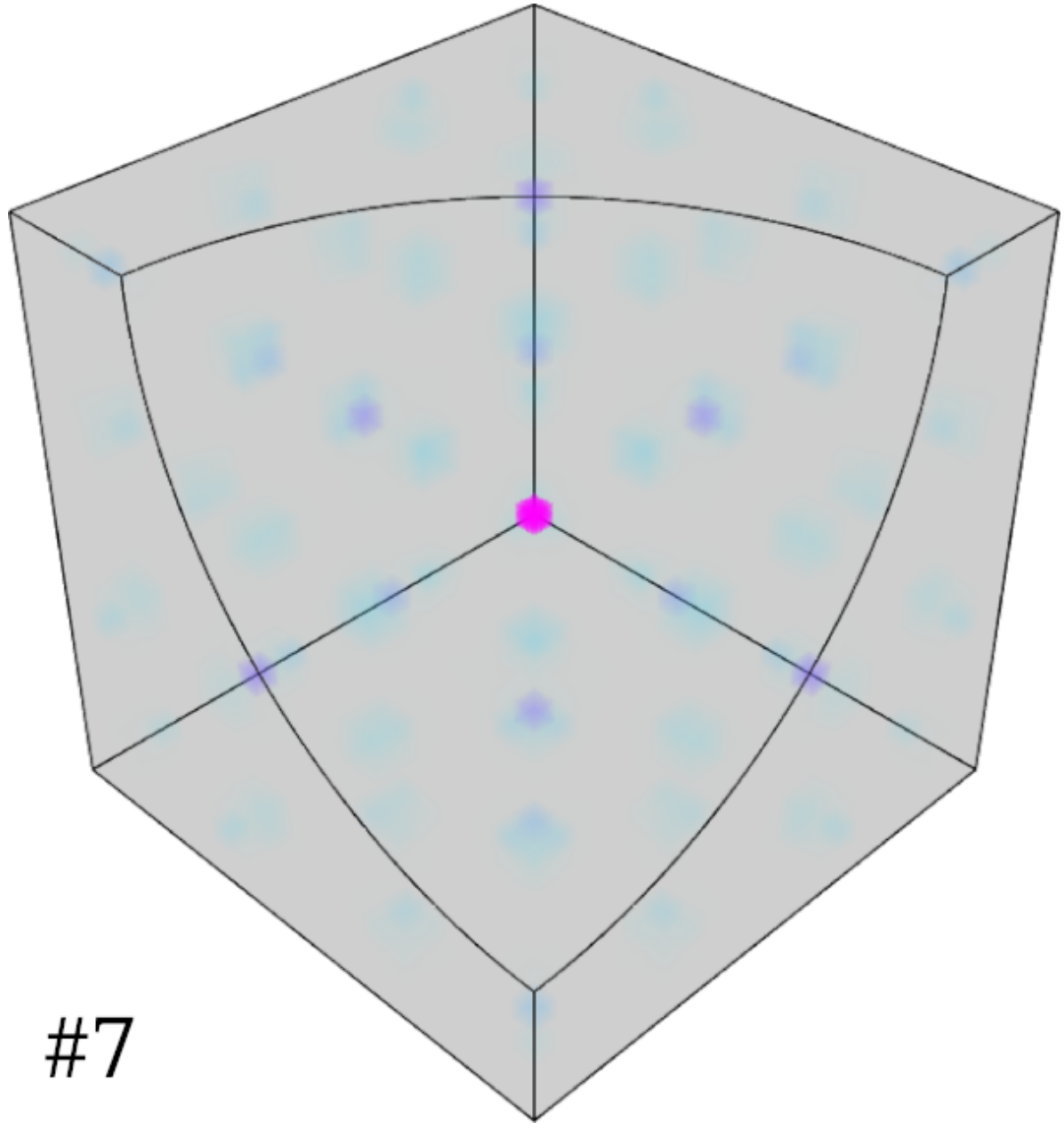
Quadrature toolbox: primal-dual RB



Quadrature toolbox: primal-dual RB

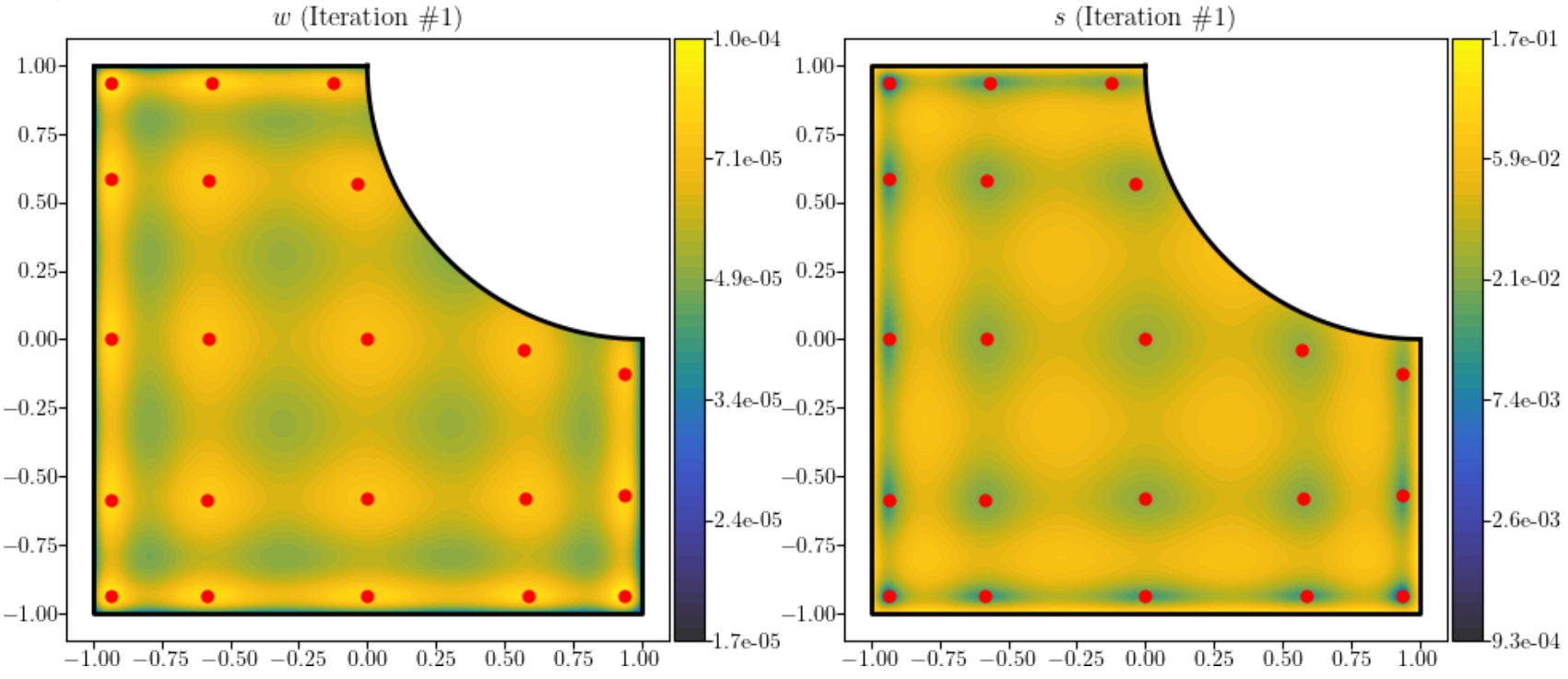


Quadrature toolbox: primal-dual RB



Quadrature toolbox: primal-dual RB

What about the dual variables? (Left: primal variable, right: dual variable)



Quadrature toolbox: dual asymptotic RB

What *are* the dual variables? Explicitly, to initialize the primal-dual iteration, we compute:

$$w^{(0)} \leftarrow V(V^{\mathsf{T}}V)^{-1}I \qquad \text{(initialize w/ Alg. LS)}$$
$$\lambda^{(0)} \leftarrow (V^{\mathsf{T}}V)^{-1}V^{\mathsf{T}}r \qquad \text{(compute Lagrange mult.)}$$
$$s^{(0)} \leftarrow r - V\lambda^{(0)} \qquad \text{(compute slack var.)}$$

Observations:

- The Lagrange multiplier $\lambda^{(0)}$ gives the coefficients in the ϕ_α basis of the projection of r onto $\mathbb{P}_{m,d}$ with respect to the uniform discrete measure supported on our sampling grid.
- Hence, the slack variable $s^{(0)}$ is the grid function which is the vector rejection of r—i.e., the part of r not in $\mathbb{P}_{m,d}$ with respect to the discrete measure.

Quadrature toolbox: dual asymptotic RB

Does this make sense? Yes. On the interval $[-1, 1]$ and with respect to the L^2 inner product, if we "vector reject" the monomial x^{2n} from $1, x, \ldots, x^{2n-1}$, we get the Legendre polynomial P_{2n}. Furthermore, the locations of the local minima of P_{2n} are asymptotic to the zeros of P_n (i.e., the order n Gauss-Legendre abscissae) (see e.g. Szëgo).

Some evidence: on the interval $[-1, 1]$, Gauss-Newton appears to always converge starting from the local minima of P_{2n}.

Fast primal/dual algorithms

Fast primal Ryu-Boyd:

- Run primal-dual interior point method for one or two steps.
- Find local maxima of w.
- Pull out those nodes; solve for weights using LS or NNLS.
- Optimize result w/ Gauss-Newton.

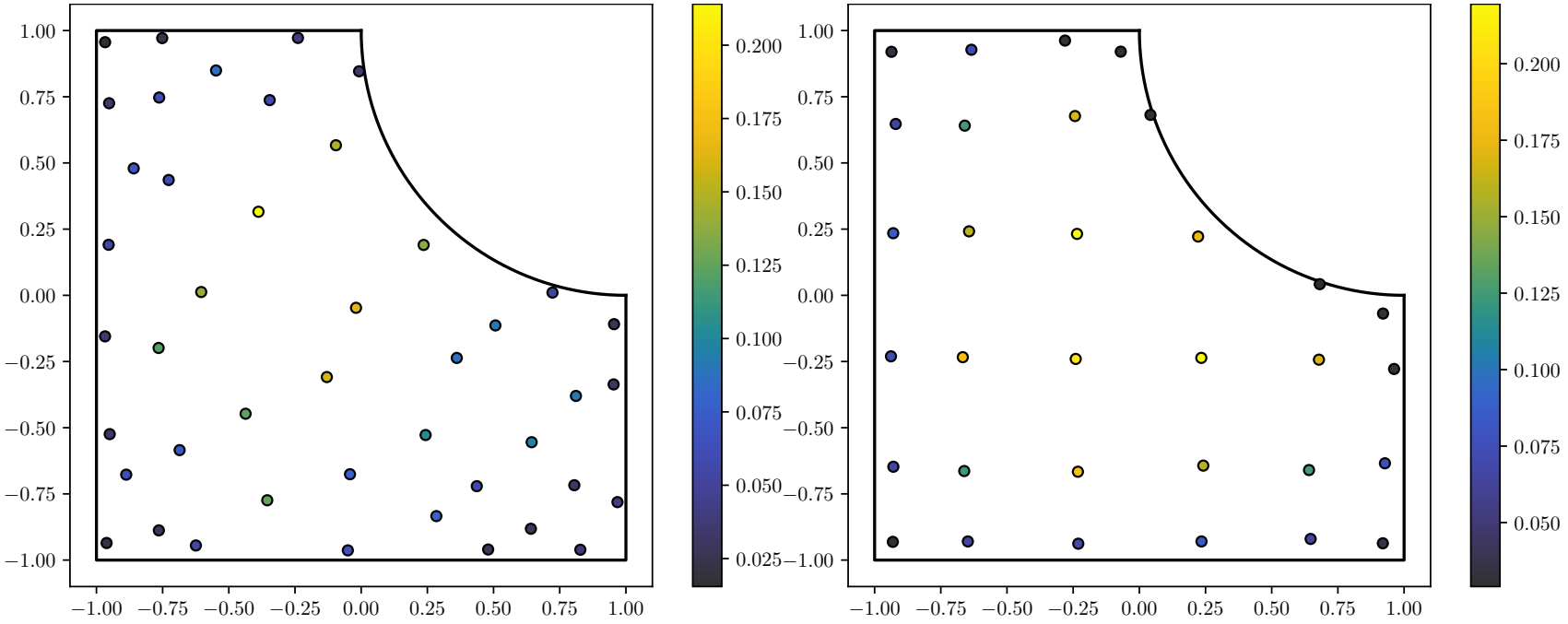
Fast dual Ryu-Boyd:

- Compute $s = r - V(V^\mathsf{T}V)^{-1}V^\mathsf{T}r$.
- Find local minima of s.
- Same deal as before.

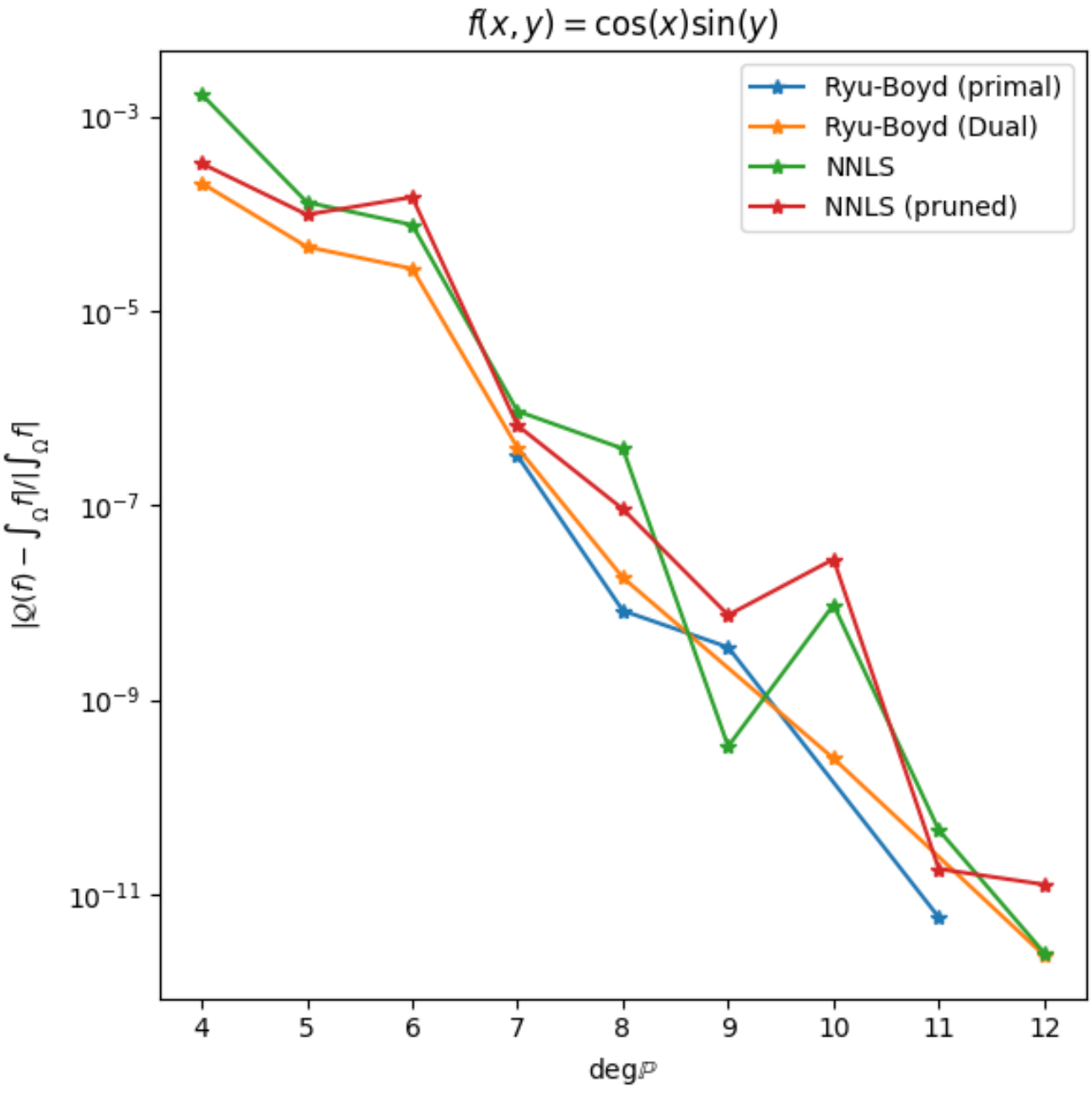
Both of these methods are heuristic!

Numerical results

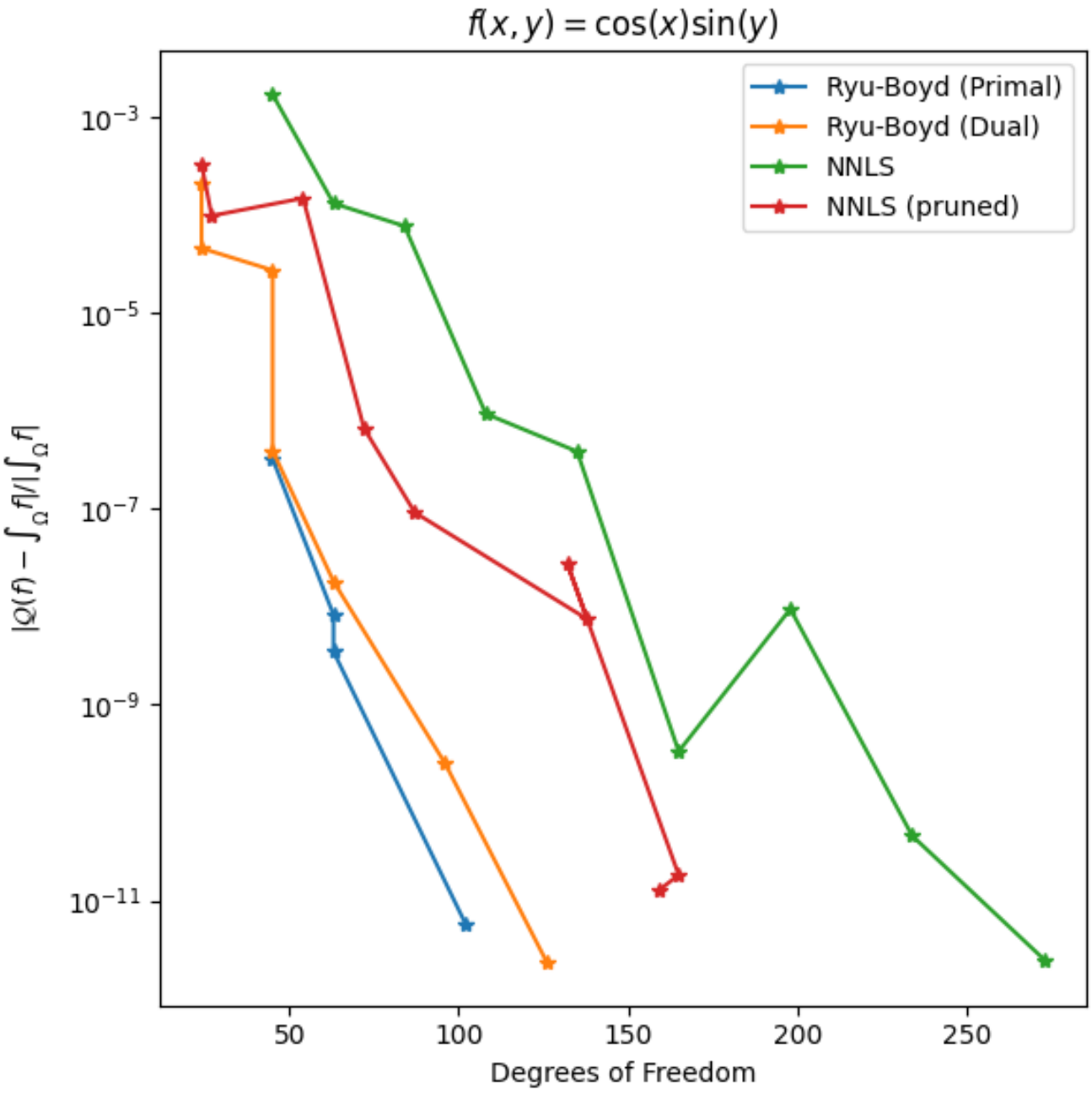
Vary degree, integrate different functions on one of our test domains, compare DOFs, accuracy, time to build rule



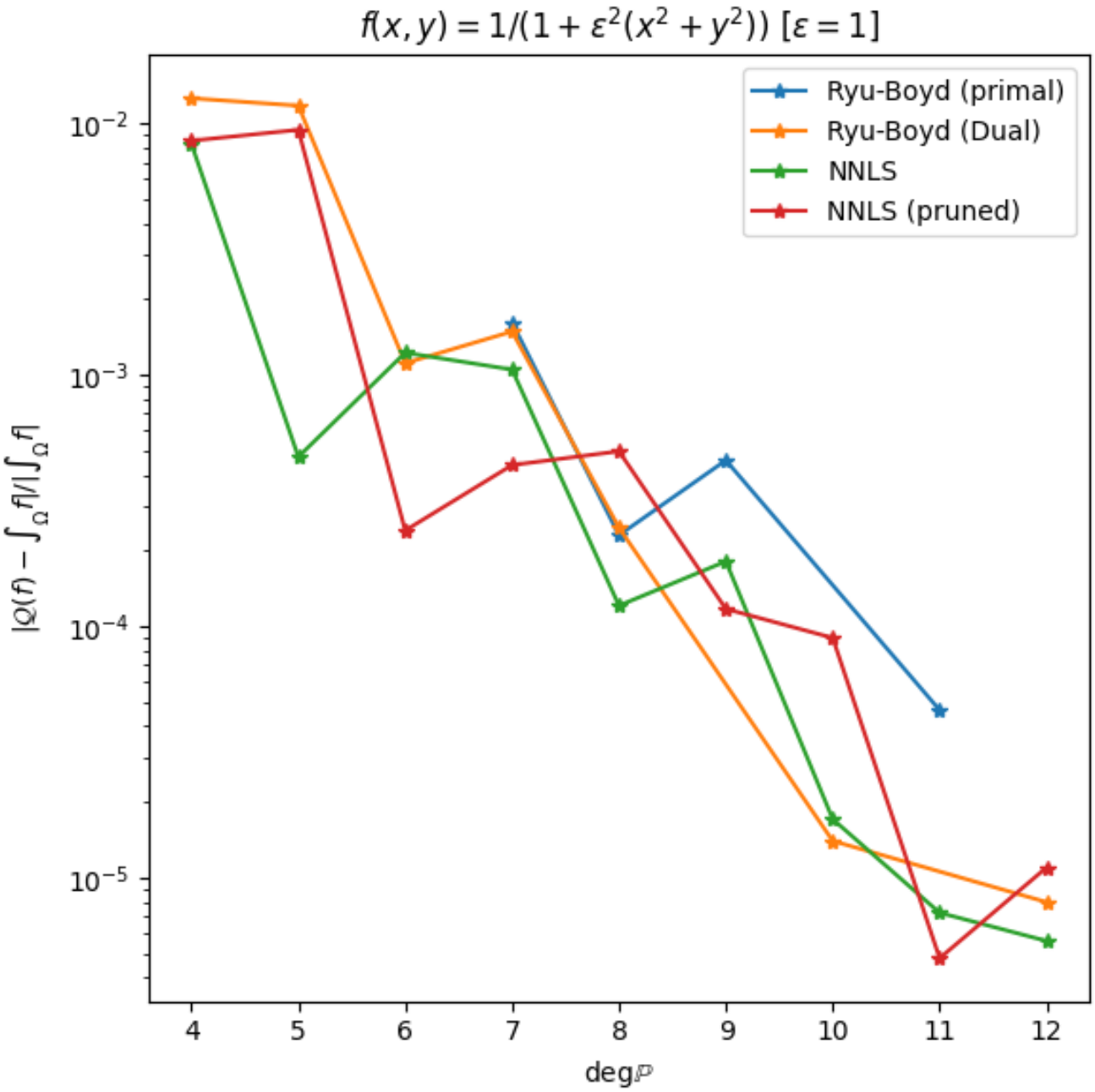
Numerical results



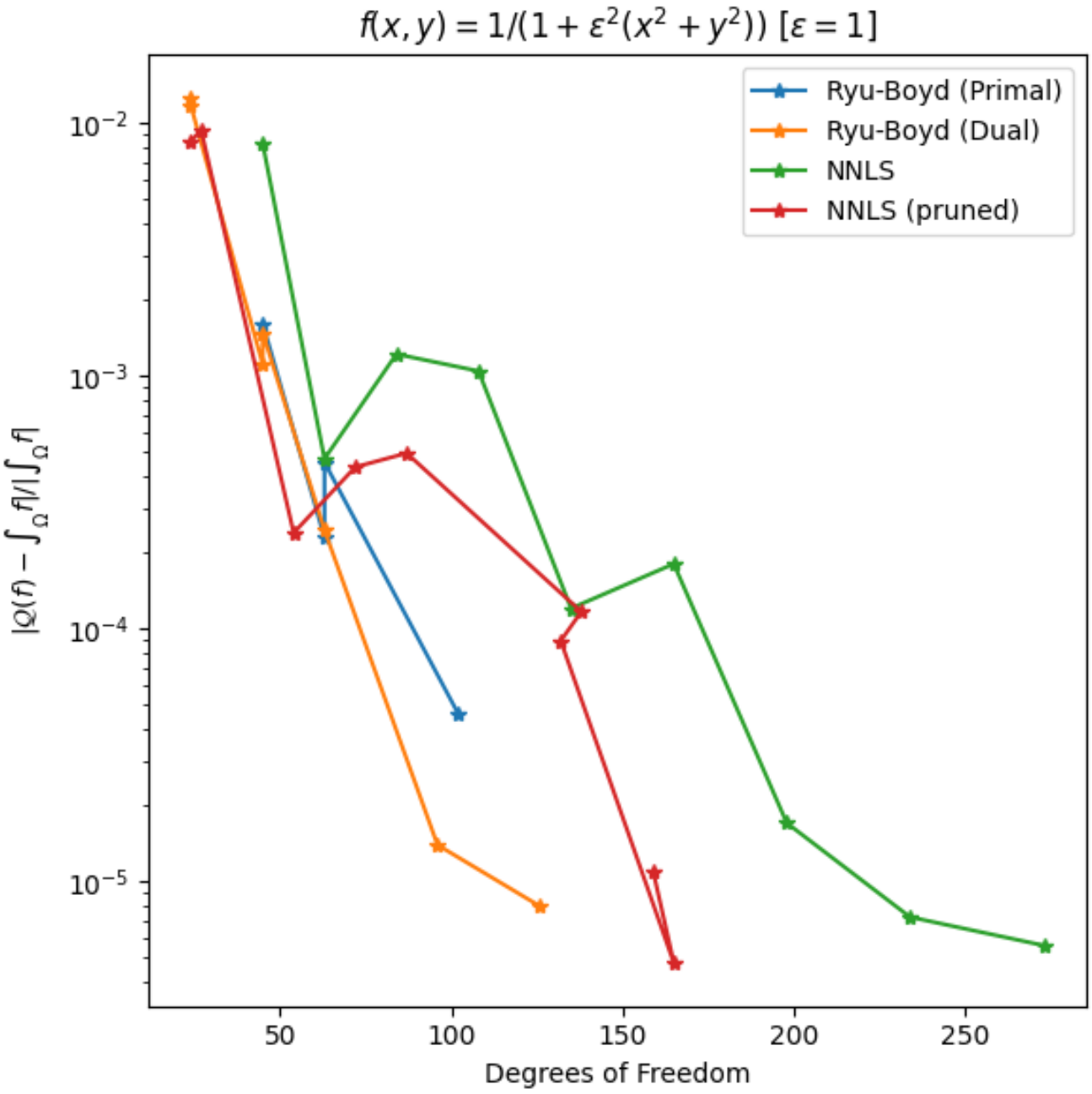
Numerical results



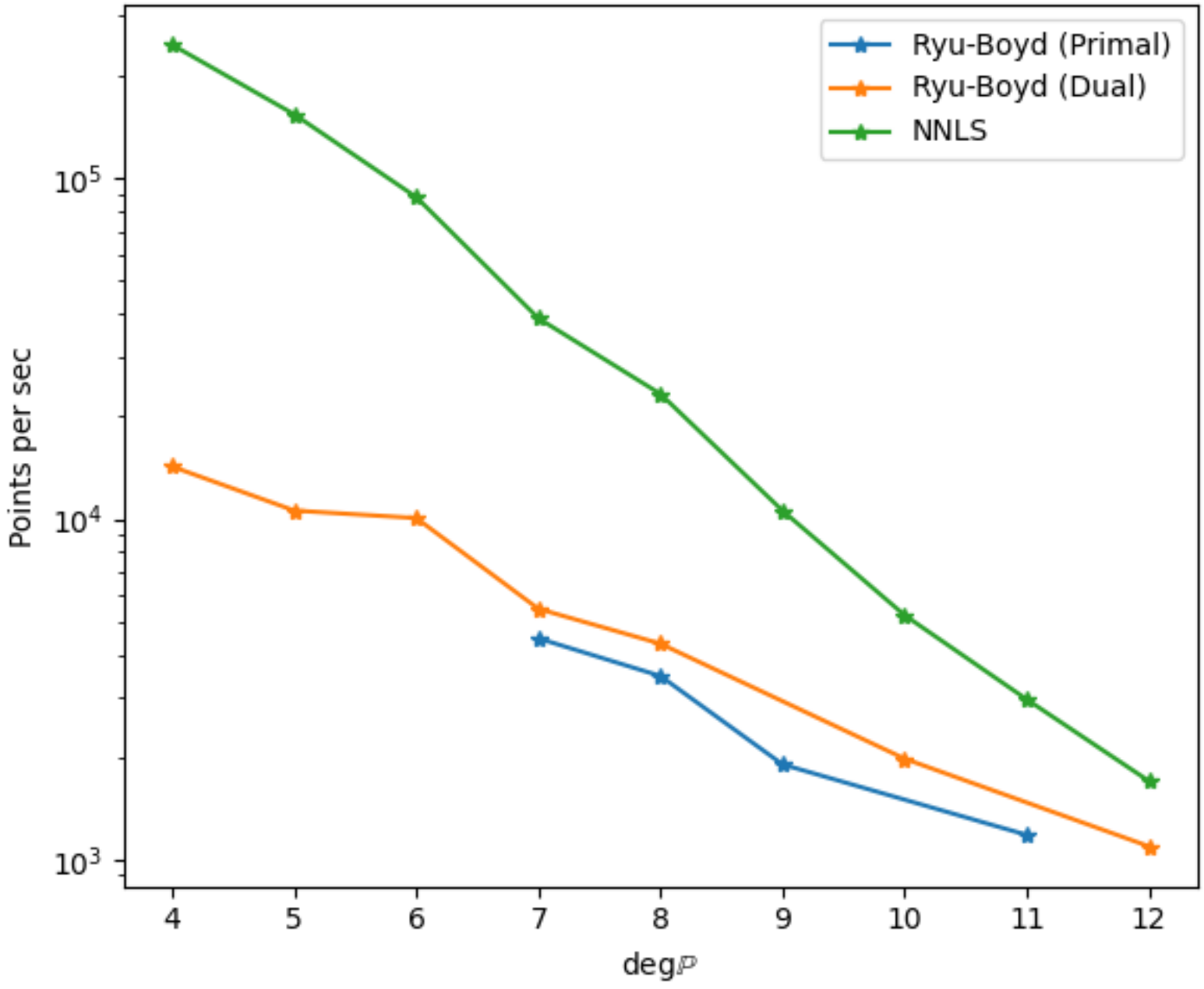
Numerical results



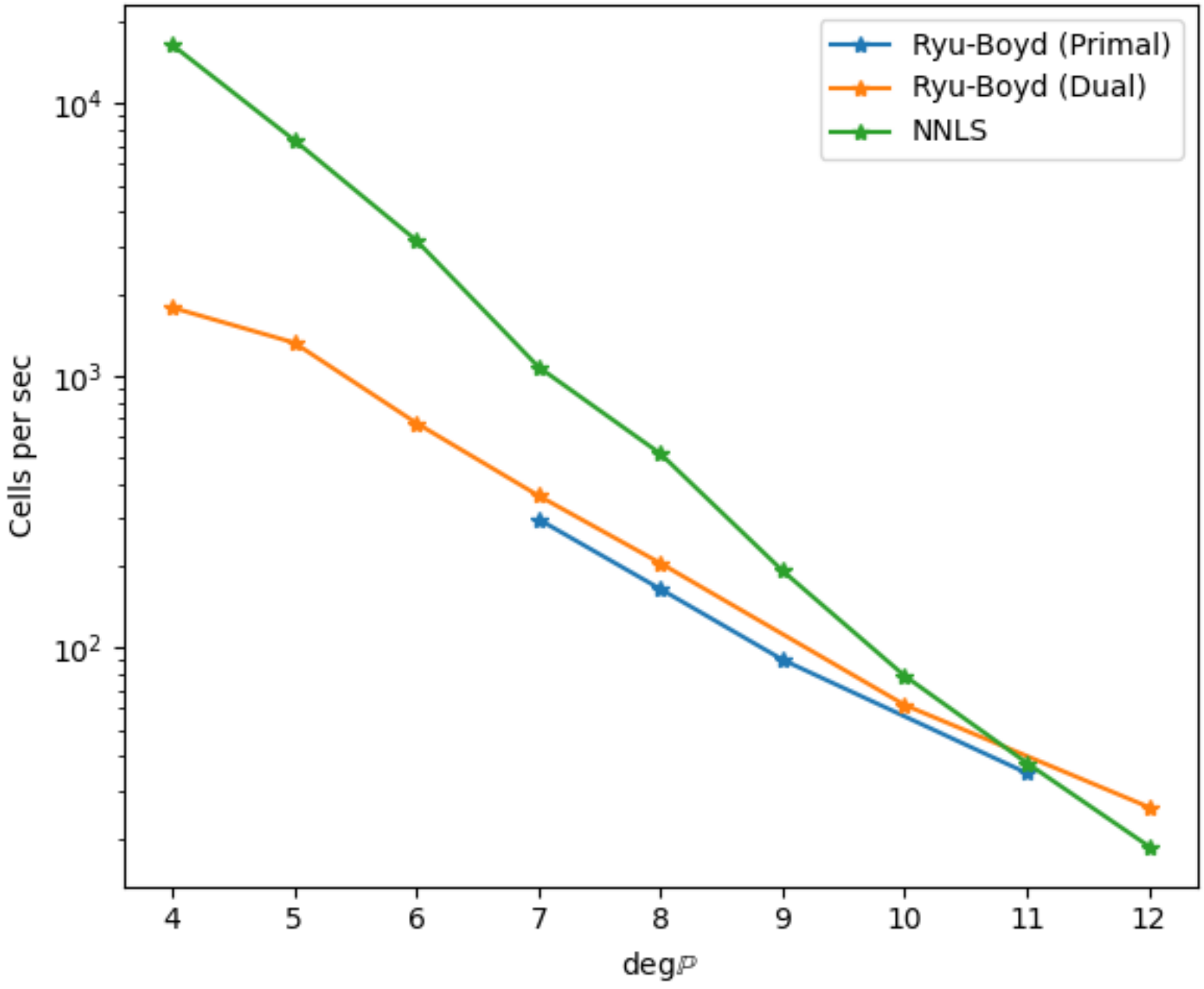
Numerical results



Numerical results

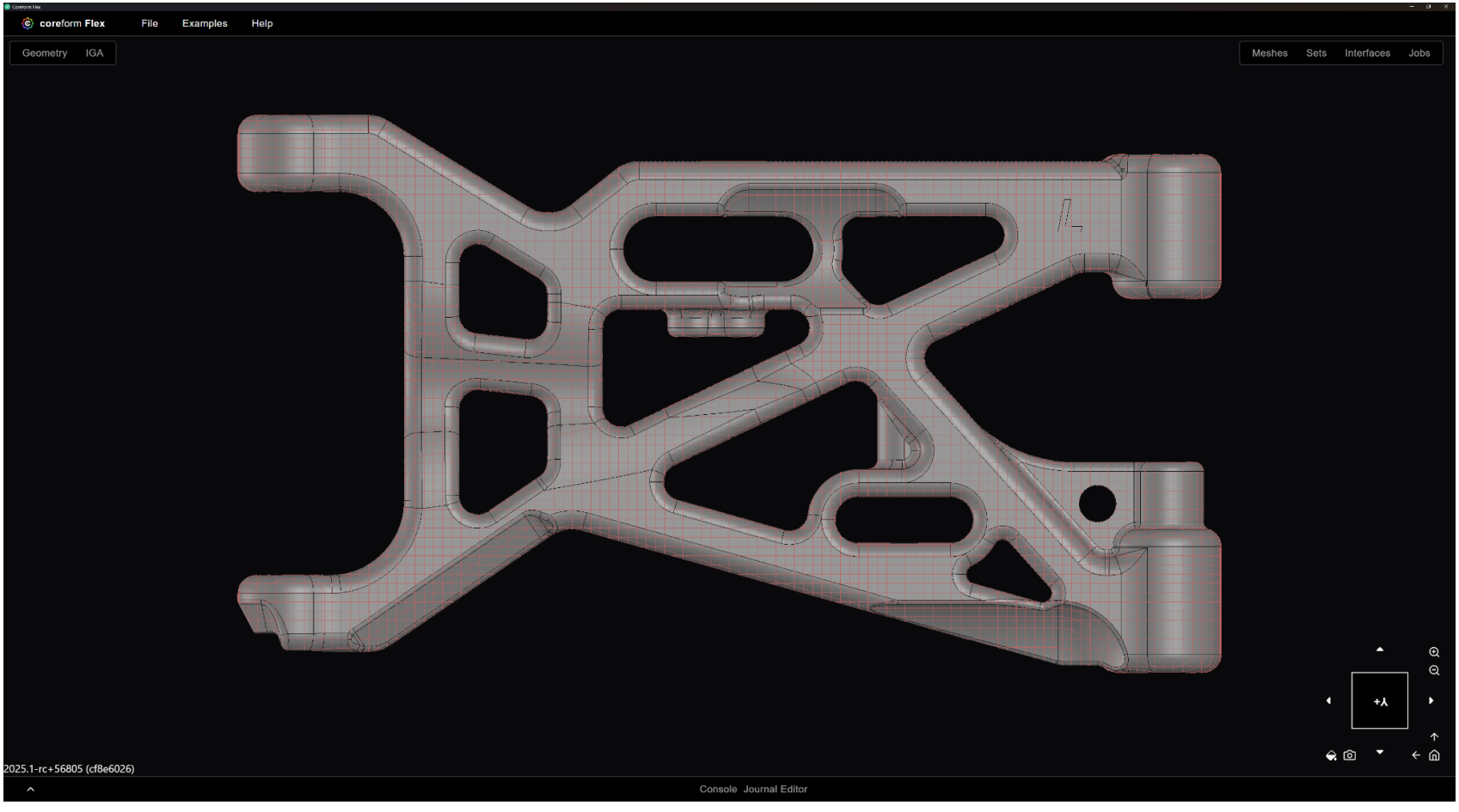


Numerical results

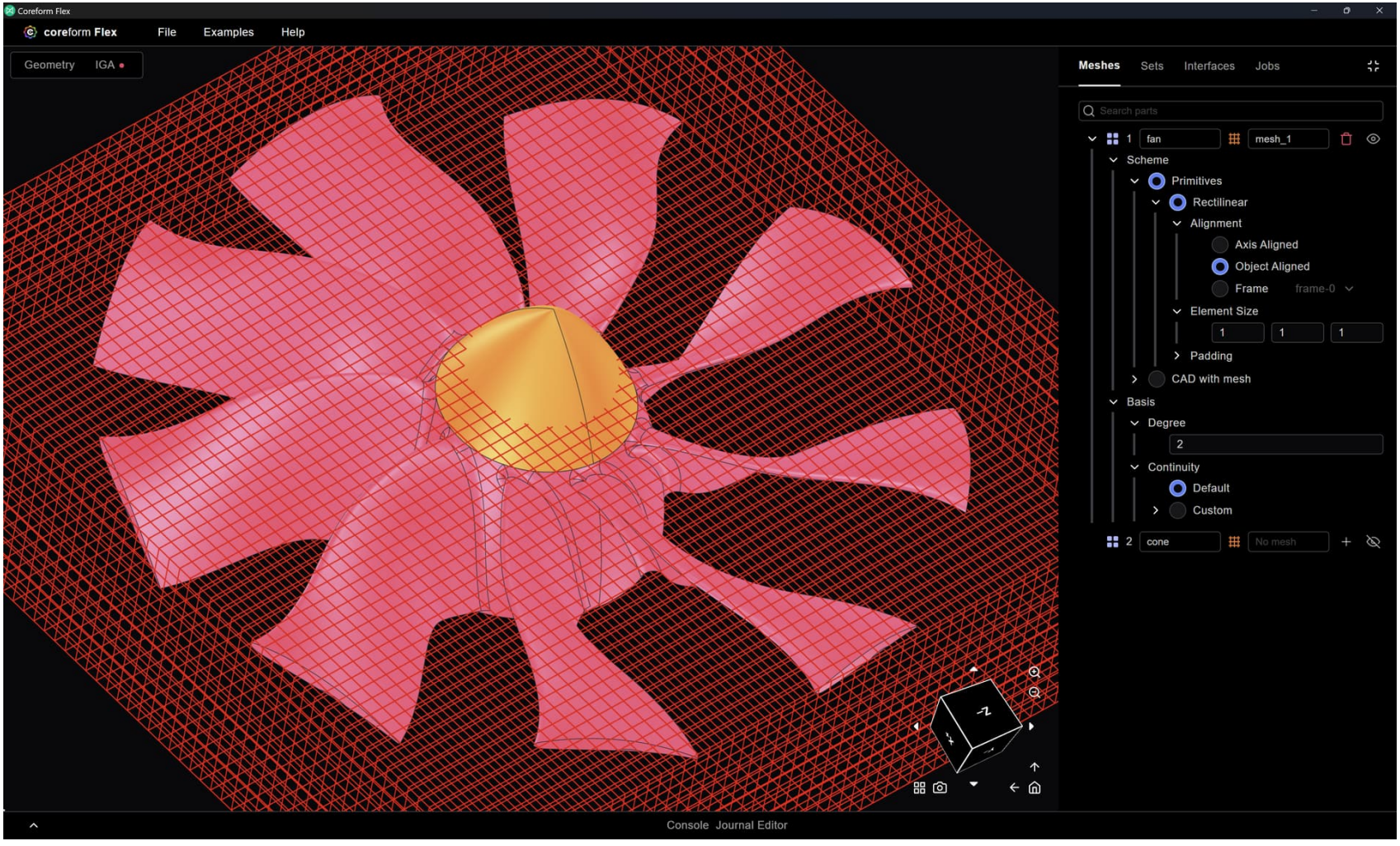


Use in Coreform Flex

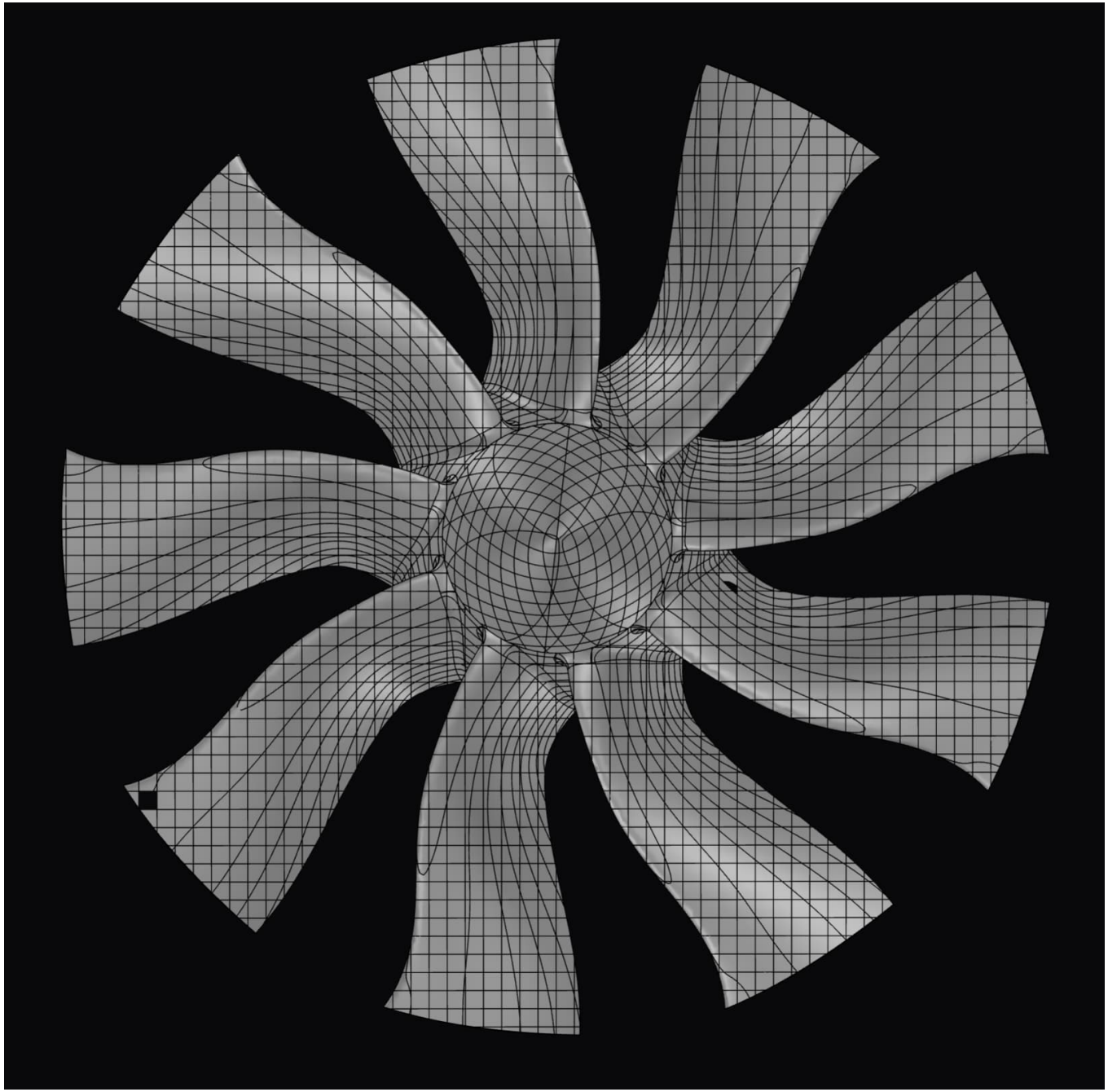
10x runtime speedup, 14x memory reduction



Use in Coreform Flex



Use in Coreform Flex



Conclusion

Thoughts:

- **Key takeway** is that there are many options to compute excellent quadrature rules on unstructured domains, allowing us to avoid mapping, meshing, decomposition, whatever.
- Lots of interesting things to try and to prove going forward to put this on firmer footing.
- The **most expensive** part of this process is setting up the moment vector I. Doing this quickly and accurately for a 3D simulation is “simple” but has a nontrivial implementation.

Paper:

- “Fast construction of efficient cut cell quadratures”, Samuel F. Potter, link to preprint available on my website https://sampotter.github.io.