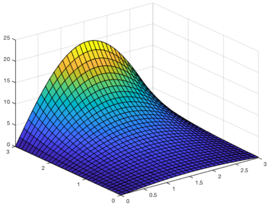
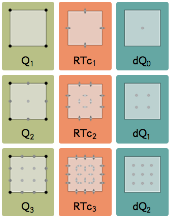
Basis construction techniques for serendipity-type spaces

Andrew Gillette
University of Arizona

joint work with
Tyler Kloefkorn, National Academy of Sciences
Victoria Sanders, University of Arizona





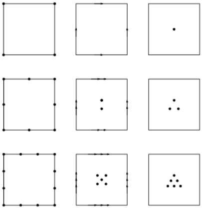
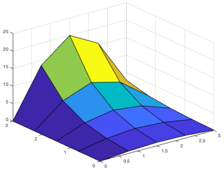
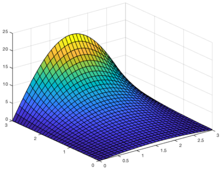


Table of Contents

Outline



$\ell = 4$



$\ell = 32$

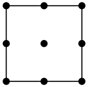
Finite element method for $\Delta u = 0$.

Boundary data: $\sin(x)\ e^y$

Domain: $[0,3]^2$, with $\ell \times \ell$ square grid.

Code: MATLAB

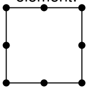
Quadratic tensor product element:



$\mathcal{Q}_2^-$

ℓ	DoFs	$\lvert\lvert u - u_h \rvert\rvert_2$	ratio	order	$\lvert\lvert \nabla u - \nabla u_h \rvert\rvert_2$	ratio	order
2	25	4.2029e-01			1.9410e+00		
4	81	5.7476e-02	7.313	2.870	5.0683e-01	3.830	1.937
8	289	7.3802e-03	7.788	2.961	1.2823e-01	3.952	1.983
16	1089	9.2909e-04	7.943	2.990	3.2157e-02	3.988	1.996
32	**4225**	1.1635e-04	7.986	**2.997**	8.0455e-03	3.997	**1.999**

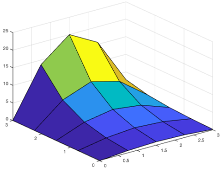
Quadratic serendipity element:



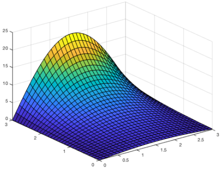
$\mathcal{S}_2^-$

ℓ	DoFs	$\lvert\lvert u - u_h \rvert\rvert_2$	ratio	order	$\lvert\lvert \nabla u - \nabla u_h \rvert\rvert_2$	ratio	order
2	21	5.6921e-01	0.000	0.000	2.4006e+00	0.000	0.000
4	65	6.0711e-02	9.376	3.229	5.3156e-01	4.516	2.175
8	225	7.4447e-03	8.155	3.028	1.2947e-01	4.106	2.038
16	833	9.3040e-04	8.002	3.000	3.2221e-02	4.018	2.007
32	**3201**	1.1637e-04	7.995	**2.999**	8.0491e-03	4.003	**2.001**

The original “serendipity” phenomenon



$\ell = 4$



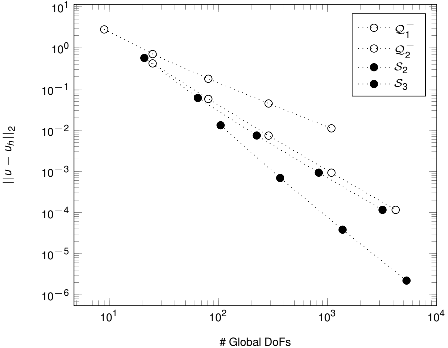
$\ell = 32$

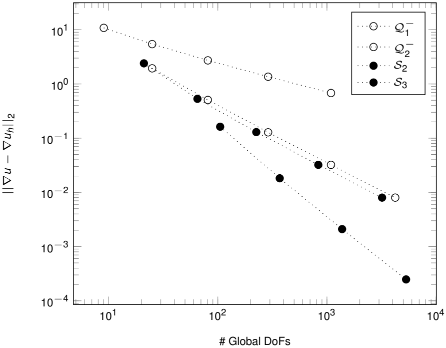
Finite element method for $\Delta u = 0$.

Boundary data: $\sin(x)\ e^y$

Domain: $[0,3]^2$, with $\ell \times \ell$ square grid.

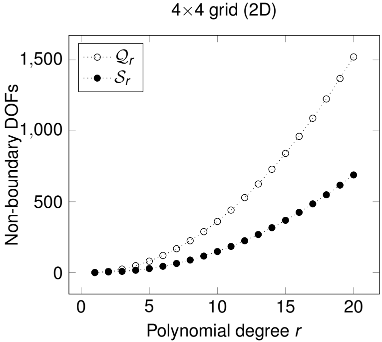
Code: MATLAB

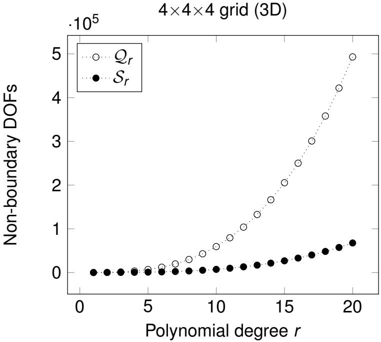




How much of a savings in DoFs can we get for large r?

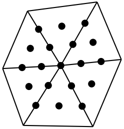
Serendipity per-element DoF savings grow with r

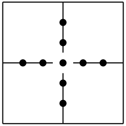




- $\rightarrow$ DoFs per $\mathcal{Q}_r^-$ (scalar) element in dim $n = (r+1)^n$
- $\rightarrow$ DoFs per $\mathcal{S}_r$ (scalar) element in dim $n = \mathcal{O}(r^n/n!)$
- $\rightarrow$ In 2D, for large r, $\mathcal{Q}_r$ has $\approx$ 2 times as many DoFs per element as $\mathcal{S}_r$
- $\rightarrow$ **In 3D**, for large r, $\mathcal{Q}_r$ has $\approx$ **5.8 times** as many DoFs per element as $\mathcal{S}_r$, including **more than 2 times** as many DoFs shared between elements!

Additional potential savings for solvers

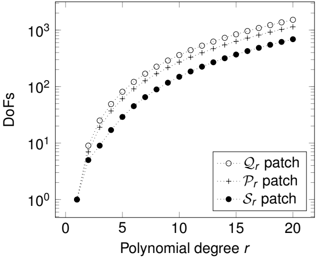




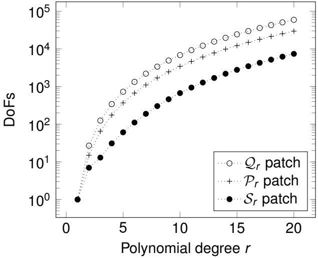
Patch-based solvers depend on a stencil of DoFs around each vertex in a mesh. Stencils for $\mathcal{P}_3$ on a triangular mesh and $\mathcal{S}_3$ on a quad mesh are shown.

$\hookrightarrow$ from a proposal with Rob Kirby (Baylor U.); currently under review

2D patches (6 triangles vs. 4 quads)



3D patches (24 tets vs. 8 hexes)



Ex: In 3D, a $\mathcal{Q}_5$ patch has $\approx$ **12 times** the number of DoFs as a $\mathcal{S}_5$ patch

$\Longrightarrow$ a quadratic order complexity solver with $\mathcal{Q}_5$ patches would have $\approx$ **144 times** longer run times than one with $\mathcal{S}_5$ patches!

Takeaway: robustly implementing serendipity elements should allow significant reduction in computational cost with no loss in order of accuracy.

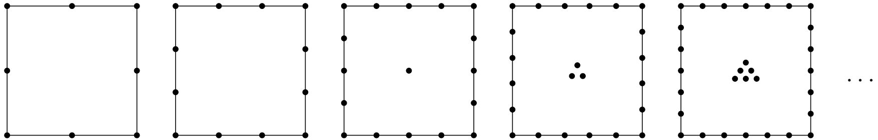
Outline

1. Serendipity methods: a review of their potential
2. Recent mathematical advances in serendipity theory
3. POEMS techniques for generic quad and hex elements
4. Explicit bases and implementation plans

Two key insights from Arnold and Awanou

→ Scalar serendipity elements exist for any order $r \geq 1$ in any dimension $n \geq 2$.

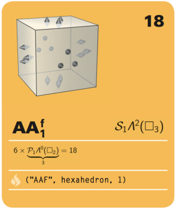
ARNOLD, AWANOU "The serendipity family of finite elements ", *Foundations of Computational Mathematics*, 2011



$r = 2$ $\quad$ $r = 3$ $\quad$ $r = 4$ $\quad$ $r = 5$ $\quad$ $r = 6$ $\quad$ ...

→ Scalar serendipity elements are part of a family of finite element differential forms.

ARNOLD, AWANOU "Finite element differential forms on cubical meshes", *Mathematics of Computation*, 2013

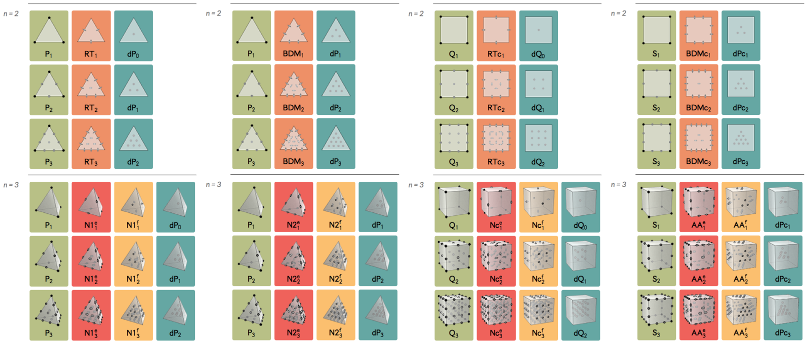


Ex: $\mathcal{S}_1\Lambda^2(\square_3)$ is an element for

$$
\begin{array}{lcl}
r = 1 & \rightarrow & \text{linear order of error decay} \\
k = 2 & \rightarrow & \text{conformity in } \Lambda^2(\mathbb{R}^3) \rightsquigarrow H(\text{div}) \\
n = 3 & \rightarrow & \text{domains in } \mathbb{R}^3
\end{array}
$$

The ‘Periodic Table of the Finite Elements’

ARNOLD, LOGG, “Periodic table of the finite elements,” *SIAM News*, 2014.

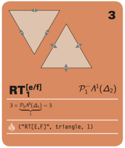


Classification of many common conforming finite element types.

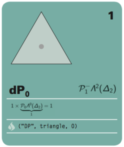
$n \rightarrow$ Domains in $\mathbb{R}^2$ (top half) and in $\mathbb{R}^3$ (bottom half)
$r \rightarrow$ Order $1, 2, 3$ of error decay (going down columns)
$k \rightarrow$ Conformity type $k = 0, \ldots, n$ (going across a row)

Geometry types: Simplices (left half) and cubes (right half).

Method selection and cochain complexes

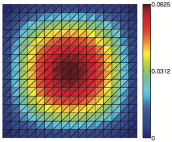


$\subset H(\mathrm{div})$



$\subset L^2$

$\Rightarrow$



Provably stable method

converges to
$u = x(1-x)y(1-y)$

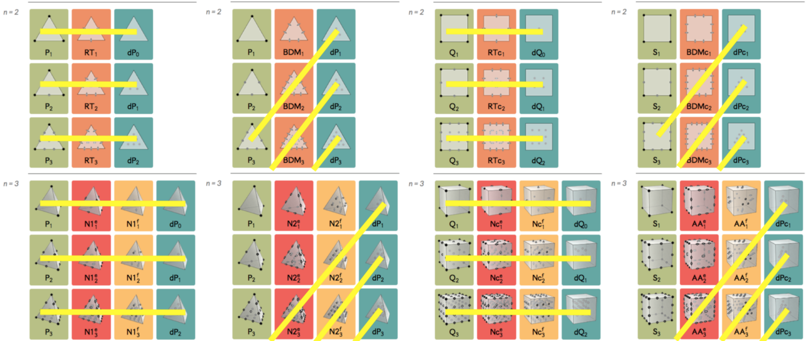
Stable pairs of elements for mixed Hodge-Laplacian problems are found by choosing consecutive spaces in compatible discretizations of the L^2 deRham Diagram.

$$H^1 \xrightarrow[\text{grad}]{\nabla} H(\mathrm{curl}) \xrightarrow[\text{curl}]{\nabla\times} H(\mathrm{div}) \xrightarrow[\text{div}]{\nabla\cdot} L^2$$

	H^1	$H(\mathrm{curl})$	$H(\mathrm{div})$	L^2
vector Poisson	σ	μ		
Maxwell's eqn's		h	b	
Darcy / Poisson			$\mathbf{u}$	p

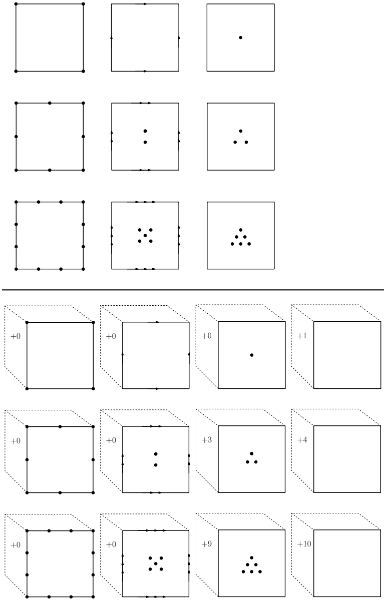
Stable pairs are found from consecutive entries in a cochain complex.

Exact cochain complexes found in the table



$\mathcal{P}_r^-\Lambda^k$ $\mathcal{P}_r\Lambda^k$ $\mathcal{Q}_r^-\Lambda^k$ $\mathcal{S}_r\Lambda^k$

- Cochain complexes occur either horizontally or diagonally in the table as shown.
- Methods can be chosen from $\mathcal{P}$ or $\mathcal{P}^-$ (simplices) and $\mathcal{Q}^-$ or $\mathcal{S}$ (cubes).
- Mysteriously, the DoF count for mixed methods from the $\mathcal{P}^-$ spaces is smaller than those from the $\mathcal{P}$ spaces, while the opposite is true for $\mathcal{Q}^-$ and $\mathcal{S}$ spaces.



A new column for the PToFE:
the **trimmed serendipity** elements.

$\boxed{\mathcal{S}_r^- \Lambda^k(\square_n)}$ denotes
approximation order r,
subset of k-form space $\Lambda^k(\Omega)$,
use on meshes of n-dim'l cubes.

Defined for any $n \geq 1$, $0 \leq k \leq n$, $r \geq 1$

Identical or analogous properties to all the other colummns in the table.

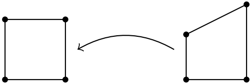
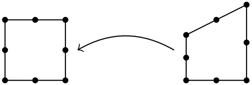
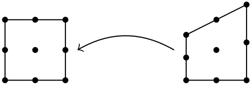
Computational advantage:
Fewer DoFs for mixed methods than both tensor product and serendipity counterparts.

G., Kloefkorn "Trimmed Serendipity Finite Element Differential Forms." *Mathematics of Computation*, to appear, 2019.

Outline

Correct usage on unstructured quad/hex meshes

Quadratic serendipity elements, mapped non-affinely, are only expected to converge at the rate of *linear* elements.

	reference → physical	$\|\|u - u_h\|\|_{L^2}$	$\|\|\nabla(u - u_h)\|\|_{L^2}$
linear		$O(h^2)$	$O(h)$
quadratic serendipity		$O(h^2)$	$O(h)$
quadratic tensor prod.		$O(h^3)$	$O(h^2)$

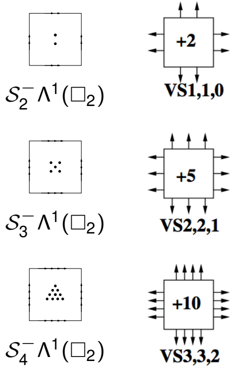
Similar problems for all elements in the serendipity families!

ARNOLD, BOFFI, FALK, "Approximation by Quadrilateral Finite Elements," *Mathematics of Computation*, 2002

ARNOLD, BOFFI, FALK, "Quadrilateral H(div) Finite Elements," *SINUM*, 2005.

ARNOLD, BOFFI, BONIZZONI, "Finite element differential forms on curvilinear cubic meshes," *Numer. Math.*, 2014

One way out: use VEM serendipity!



$\rightarrow$ The VEM serendipity spaces $VEMS^f_{r,r,r-1}$ on quads have the same degree of freedom counts as the trimmed serendipity spaces $\mathcal{S}^-_{r+1}\Lambda^1(\square_2)$

$\rightarrow$ Similar equivalences hold between other VEM serendipity spaces and other (trimmed) serendipity spaces.

$\rightarrow$ Going the VEM route means giving up on local basis functions.

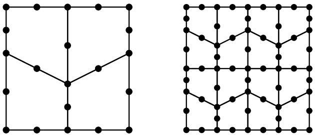
Beirão da Veiga, Brezzi, Marini, Russo "Serendipity face and edge VEM spaces" *Rendiconti Lincei-Matematica e Applicazioni*, 2017.

Another way out: basis functions on physical elements

→ Define basis functions ψ_{ij} on physical elements:

$$u_h = I_q u := \sum_{i=1}^{n} u(\mathbf{v}_i)\psi_{ii} + u\left(\frac{\mathbf{v}_i + \mathbf{v}_{i+1}}{2}\right)\psi_{i(i+1)}$$

→ Hard to generalize and compute beyond quadratic order



Non-affine bilinear mapping

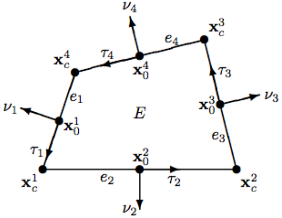
	$\|\|u - u_h\|\|_{L^2}$		$\|\|\nabla(u - u_h)\|\|_{L^2}$	
n	error	rate	error	rate
2	5.0e-2		6.2e-1	
4	6.7e-3	2.9	1.8e-1	1.8
8	9.7e-4	2.8	5.9e-2	1.6
16	1.6e-4	2.6	2.3e-2	1.4
32	3.3e-5	2.3	1.0e-2	1.2
64	7.4e-6	2.1	4.96e-3	1.1

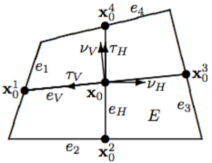
Physical element basis functions:

	$\|\|u - u_h\|\|_{L^2}$		$\|\|\nabla(u - u_h)\|\|_{L^2}$	
n	error	rate	error	rate
2	2.34e-3		2.22e-2	
4	3.03e-4	2.95	6.10e-3	1.87
8	3.87e-5	2.97	1.59e-3	1.94
16	4.88e-6	2.99	4.04e-4	1.97
32	6.13e-7	3.00	1.02e-4	1.99
64	7.67e-8	3.00	2.56e-5	1.99

RAND, G., BAJAJ "Quadratic Serendipity Finite Elements on Polygons Using Generalized Barycentric Coordinates." *Mathematics of Computation* , 83:290, 2014.

"Half-and-half": the Arbogast-Correa technique





A finite element space on a general quadrilateral is built in two parts:

- Apply Piola mapping to functions associated to boundary of reference element.
- Define functions on the physical element corresponding to interior degrees of freedom in a way that ensures relevant polynomial approximation properties.

ARBOGAST, CORREA "Two families of H(div) mixed finite elements on quadrilaterals of minimal dimension," *SIAM J. Numerical Analysis*, 2016

Outline

Building a computational basis

dx	dy	dz
$-yz$	xz	0
0	$-xz$	xy
yz	xz	xy
$2xy$	x^2	0
$2xz$	0	x^2
y^2	$2xy$	0
0	$2yz$	y^2
z^2	0	$2xz$
0	z^2	$2yz$
$2xyz$	x^2z	x^2y
y^2z	$2xyz$	xy^2
yz^2	xz^2	$2xyz$

Goal: find a computational basis for $\mathcal{S}_1\Lambda^1(\square_3)$:

- Must be $H(\text{curl})$-conforming
- Must have 24 functions, 2 associated to each edge of cube
- Must recover constant and linear approx. on each edge
- The approximation space contains:

  (1) Any polynomial coefficient of at most linear order:
  $\{1, x, y, z\} \times \{dx, dy, dz\} \ \rightarrow$ 12 forms

  (2) Certain forms with quadratic or cubic order coefficients shown in table at left $\rightarrow$ 12 forms

- For constants, use “obvious” functions:

$$\{(y \pm 1)(z \pm 1)dx, \ \ (x \pm 1)(z \pm 1)dy, \ \ (x \pm 1)(y \pm 1)dz\}$$

e.g. $(y+1)(z+1)dx$ evaluates to zero on every edge ***except*** $\{y = 1, z = 1\}$ where it is $\equiv 4 \rightarrow$ constant approx.

Also, $(y+1)(z+1)dx$ can be written as a linear combo, by using the first three forms at left to get the $yz\ dx$ term

Building a computational basis

dx	dy	dz
$-yz$	xz	0
0	$-xz$	xy
yz	xz	xy
$2xy$	x^2	0
$2xz$	0	x^2
y^2	$2xy$	0
0	$2yz$	y^2
z^2	0	$2xz$
0	z^2	$2yz$
$2xyz$	x^2z	x^2y
y^2z	$2xyz$	xy^2
yz^2	xz^2	$2xyz$

- For constant approx on edges, we used:

$$\{(y\pm 1)(z\pm 1)dx,\ \ (x\pm 1)(z\pm 1)dy,\ \ (x\pm 1)(y\pm 1)dz\}$$

- Guess for linear approx on edges:

$$\{x(y\pm 1)(z\pm 1)dx,\ \ y(x\pm 1)(z\pm 1)dy,\ \ z(x\pm 1)(y\pm 1)dz\}$$

e.g. $x(y+1)(z+1)dx$ evaluates to $4x$ on $\{y=1, z=1\}$.

- Unfortunately: $\boxed{x(y+1)(z+1)dx \notin \mathcal{S}_1\Lambda(\square_3)!}$

Why? $x(y+1)(z+1)dx = (xyz+xy+xz+x)dx$

but $xyz\ dx$ *only appears with other cubic order coefficients!*

- Remedy: add dy and dz terms that vanish on all edges.

Building a computational basis

dx	dy	dz
$-yz$	xz	0
0	$-xz$	xy
yz	xz	xy
$2xy$	x^2	0
$2xz$	0	x^2
y^2	$2xy$	0
0	$2yz$	y^2
z^2	0	$2xz$
0	z^2	$2yz$
$2xyz$	x^2z	x^2y
y^2z	$2xyz$	xy^2
yz^2	xz^2	$2xyz$

Computational basis element associated to $\{y = 1, z = 1\}$:

$$\boxed{2x(y+1)(z+1)\,dx + (z+1)(x^2-1)\,dy + (y+1)(x^2-1)\,dz}$$

✓ Evaluates to $4x$ on $\{y = 1, z = 1\}$ (linear approx.)

✓ Evaluates to 0 on all other edges

✓ Belongs to the space $\mathcal{S}_1\Lambda(\square_3)$:

$$\begin{array}{rcrcrl}
2xyz\,dx & + & x^2z\,dy & + & x^2y\,dz & \\
2xy\,dx & + & x^2\,dy & + & 0\,dz & \\
2xz\,dx & + & 0\,dy & + & x^2\,dz & \\
2x\,dx & + & (-z-1)dy & + & (-y-1)dz & \leftarrow \text{linear order} \\
\hline
\end{array}$$

$\hookrightarrow$ summation and factoring yields the desired form)

There are 11 other such functions, one per edge. We have:

$$\begin{array}{lcccc}
\mathcal{S}_1\Lambda(\square_3) & = & \underbrace{E_0\Lambda^1(\square_3)}_{\text{"obvious" basis for constant approx}} & \oplus & \underbrace{\tilde{E}_1\Lambda^1(\square_3)}_{\text{modified basis for linear approx}} \\
\dim\ 24 & = & 12 & + & 12
\end{array}$$

A complete table of computational bases

$n=3$	$k=0$	$k=1$	$k=2$	$k=3$
$\mathcal{S}_r\Lambda^k$	$V\Lambda^0(\square_3)$	$\emptyset$	$\emptyset$	$\emptyset$
	$\bigoplus_{i=0}^{r-2} E_i\Lambda^0(\square_3)$	$\bigoplus_{i=0}^{r-1} E_i\Lambda^1(\square_3) \oplus \tilde{E}_r\Lambda^1(\square_3)$	$\emptyset$	$\emptyset$
	$\bigoplus_{i=4}^{r} F_i\Lambda^0(\square_3)$	$\bigoplus_{i=2}^{r-1} F_i\Lambda^1(\square_3) \oplus \hat{F}_r\Lambda^1(\square_3)$	$\bigoplus_{i=0}^{r-1} F_i\Lambda^2(\square_3) \oplus \tilde{F}_r\Lambda^2(\square_3)$	$\emptyset$
	$\bigoplus_{i=6}^{r} I_i\Lambda^0(\square_3)$	$\bigoplus_{i=4}^{r} I_i\Lambda^1(\square_3)$	$\bigoplus_{i=2}^{r} I_i\Lambda^2(\square_3)$	$\bigoplus_{i=2}^{r} I_i\Lambda^3(\square_3)$
$\mathcal{S}_r^-\Lambda^k$	$V\Lambda^0(\square_3)$	$\emptyset$	$\emptyset$	$\emptyset$
	$\bigoplus_{i=0}^{r-2} E_i\Lambda^0(\square_3)$	$\bigoplus_{i=0}^{r-1} E_i\Lambda^1(\square_3)$	$\emptyset$	$\emptyset$
	$\bigoplus_{i=4}^{r} F_i\Lambda^0(\square_3)$	$\bigoplus_{i=2}^{r-1} F_i\Lambda^1(\square_3) \oplus \tilde{F}_r\Lambda^1(\square_3)$	$\bigoplus_{i=0}^{r-1} F_i\Lambda^2(\square_3)$	$\emptyset$
	$\bigoplus_{i=6}^{r} I_i\Lambda^0(\square_3)$	$\bigoplus_{i=4}^{r-1} I_i\Lambda^1(\square_3) \oplus \tilde{I}_r\Lambda^1(\square_3)$	$\bigoplus_{i=2}^{r-1} I_i\Lambda^2(\square_3) \oplus \tilde{I}_r\Lambda^2(\square_3)$	$\bigoplus_{i=2}^{r-1} I_i\Lambda^3(\square_3)$

G., Kloefkorn, Sanders "Computational serendipity and tensor product finite element differential forms."
SMAI J. Computational Mathematics, to appear, 2019.

Open source finite element software packages I

deal.II

→ open source C++ program library for adaptive FEM, in development since 1998

→ designed to support quad/hex meshes and *h*/*p* adaptivity

→ data structures are well-documented but not easy to introduce new element types without in-depth knowledge of the code

FEniCS PROJECT Firedrake

→ FEM toolkits that use Unified Form Language to define a weak form and create local assembly kernels

→ FEniCS passes kernels to DOLFIN's C++ libraries and PETSc to do solves

→ Firedrake creates intermediate data structures that are then passed to parallel schedulers, including notions like "`dofs`" and "`interior facet`" that more easily accommodate extensibility

ALNÆS ET AL. "The FEniCS Project Version 1.5" *Archive of Numerical Software*, 2015

RATHGEBER ET AL. "Firedrake: automating the finite element method by composing abstractions" *ACM Transactions on Mathematical Software*, 2016.

BANGERTH ET AL. "The `deal.ii` Library, Version 8.4," *Journal of Numerical Mathematics*, 2016

First pass at Firedrake implementation

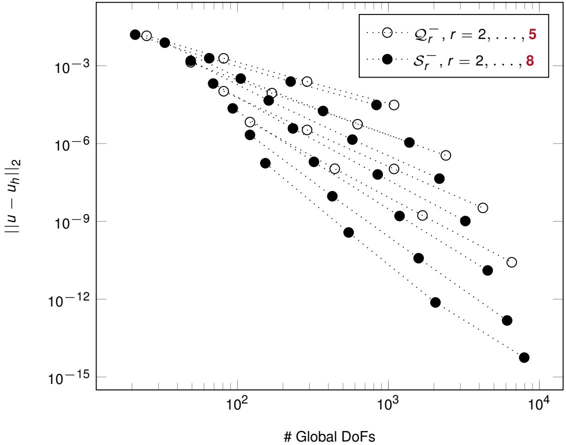
→ Scalar-valued, 2D square elements only (so far!)

→ Replaced "monomial" parts of basis with Legendre polynomials.

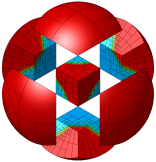
→ Laplace problem with boundary data: $\cos(\pi x)\cos(\pi y)$

→ Domain: $[0, 1]^2$, with $\ell \times \ell$ square grid.

→ Code: Firedrake, with Krylov solver options



Open source finite element software packages II



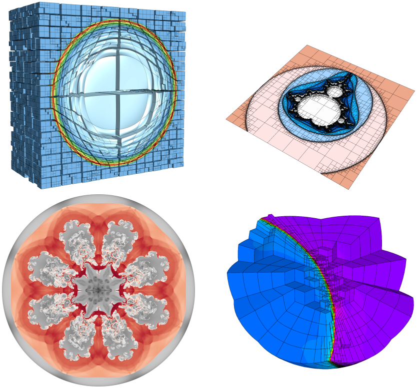
MFEM:

Modular
Finite
Element
Methods library

→ "free, lightweight, scalable C++ library for FE methods," developed at Lawrence Livermore National Labs since 2010

→ emphasis placed on high-order methods, parallelizability, and support for variety of techniques

→ supports lab missions in studies of hydrodynamics, magnetostatics, fusion, turbulence, etc.



Pictures from `mfem.org/gallery`

I will be working with the MFEM team at LLNL this summer to (begin to) implement serendipity elements in their package!

ANDERSON ET AL. "MFEM: A Modular Finite Element Methods Library," `mfem.org`, LLNL, 2010–2019

Acknowledgments

Merci **to the organizers for the invitation!**

Collaborators on this work

Rob Kirby	Baylor University
Tyler Kloefkorn	National Academies Program Officer, Math (*former postdoc*)
Victoria Sanders	U. Arizona (*undergrad math major*)

Research Funding

Supported in part by the National Science Foundation grant DMS-1522289.

Slides and Pre-prints

`http://math.arizona.edu/~agillette/`