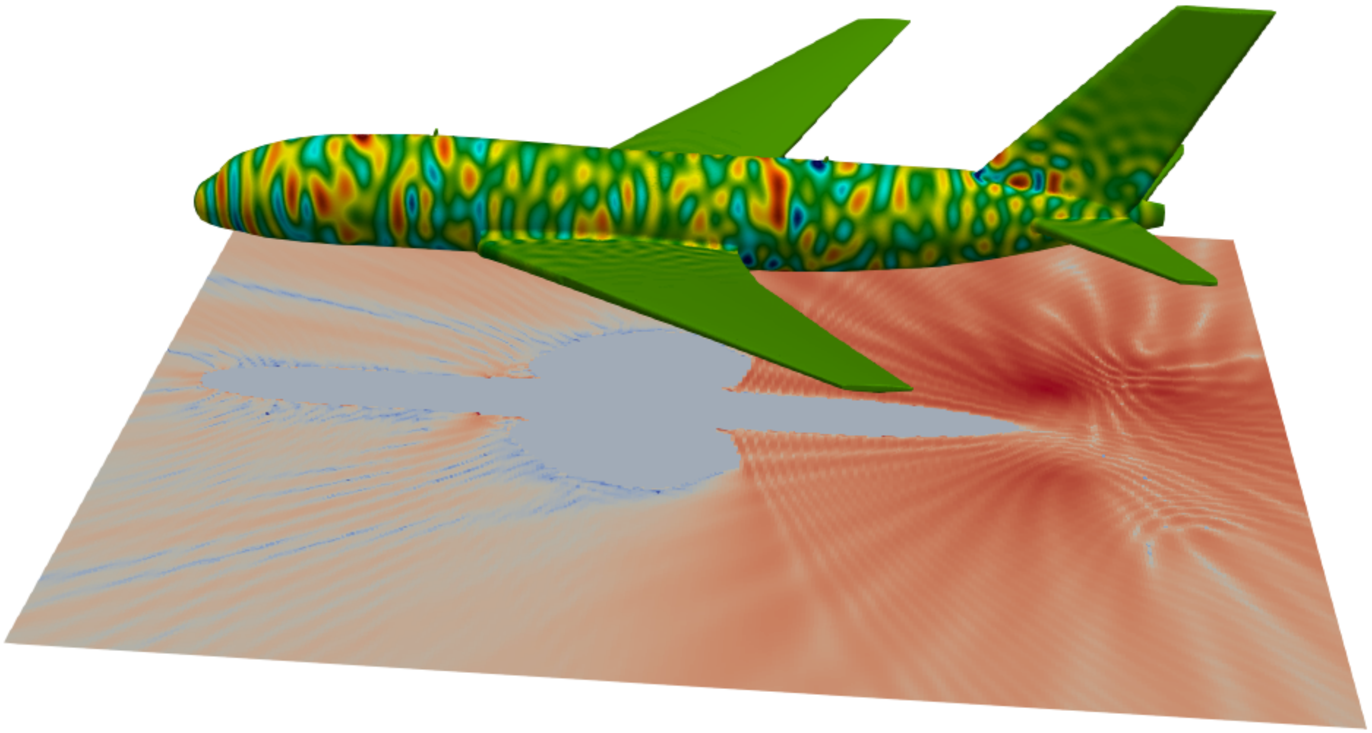
fmm3dbie - devel

fmm3dbie - a library for

- fmm-accelerated layer potential evaluators
- Iterative solvers using them
- robust locally corrected quadrature methods
- matrix entry generator for fast direct solvers
- Ability to interface with varied CAD formats
- Interfaces in Fortran/C/Python/MATLAB(?)

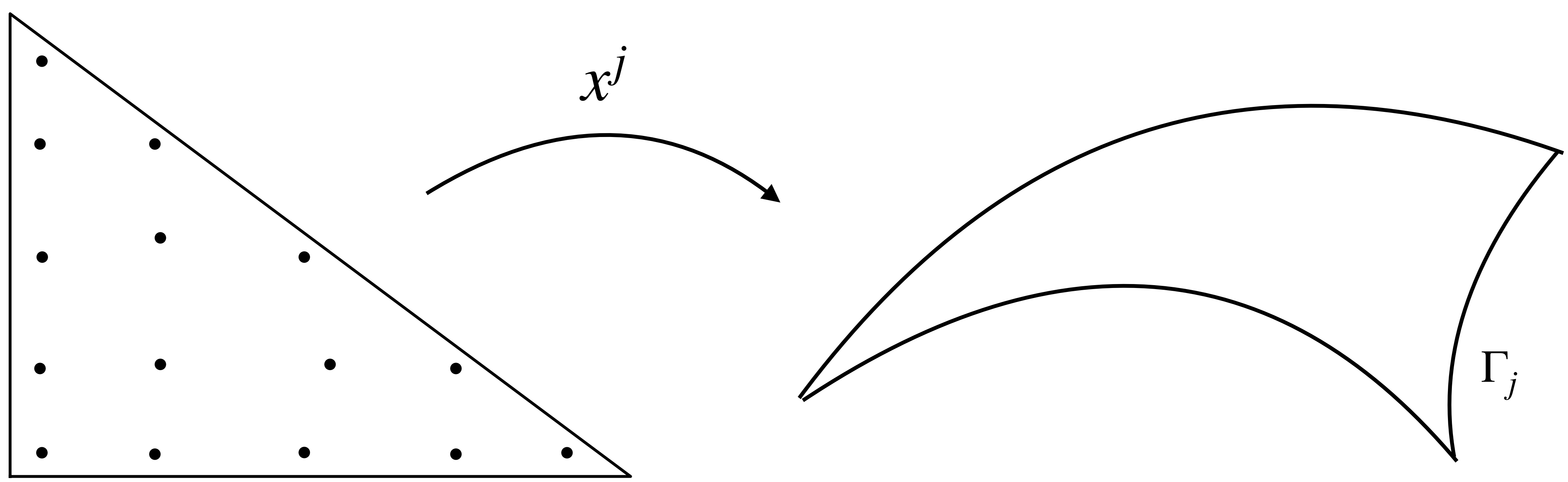
$$(\Delta + k^2)u = 0 \quad \textbf{in } \Omega$$
$$u = g \quad \textbf{on } \partial\Omega = \Gamma$$

$$u = \alpha S_k[\sigma] + \beta D_k[\sigma]$$
$$g = -\beta\sigma/2 + \alpha S_k[\sigma] + \beta D_k[\sigma]$$



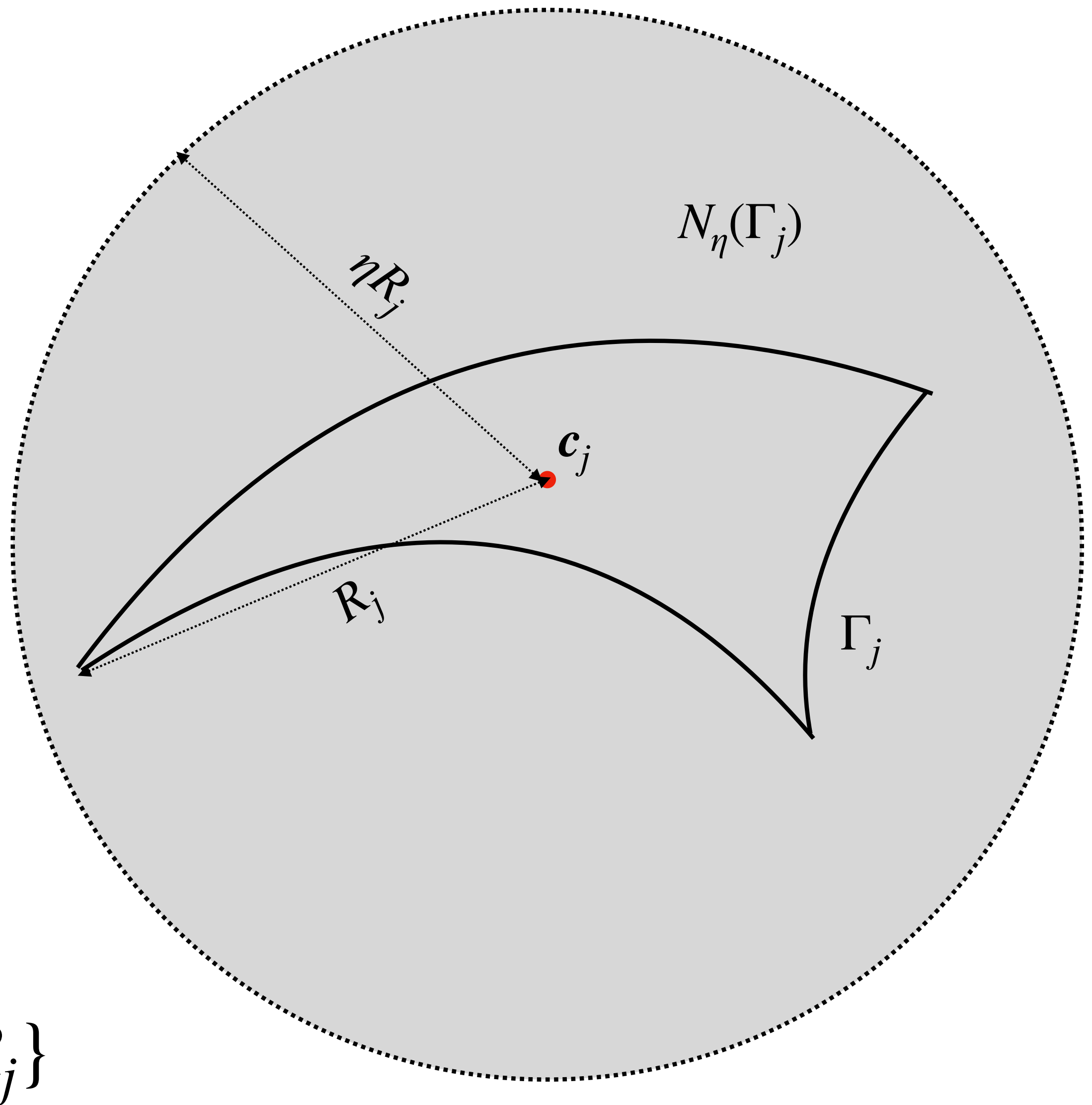
Surface representation

- $\Gamma = \cup_j \Gamma_j$
- Chart for $\Gamma_j = x^j$, i.e., $x^j : B \to \Gamma_j$
- Supported elements B:
  - $T_0 = \{(u, v) : u > 0, v > 0, u + v < 1\}$, basis on $T_0 = \{K_{nm}(u, v), n + m < p\}$, where K_{nm} are Koornwinder polynomials, and discretization nodes Vioreanu-Rokhlin nodes



Near-far quadrature split

- Centroid: $c_j = \int_{\Gamma_j} \boldsymbol{x} da$
- Bounding sphere radius
  $R_j = \min_R \{R : \Gamma_j \subset B_R(c_j)\}$
- η-scaled near-field of Γ_j:
  $N_\eta(\Gamma_j) = \{x : |x - c_j| \leq \eta R_j\}$
- $T_\eta(x)$: collection of patches for which x is in the near-field — dual of $N_\eta(\Gamma_j)$
  $T_\eta(x) = \{\Gamma_j : x \in N_\eta(\Gamma_j)\} = \{\Gamma_j : |x - c_j| \leq \eta R_j\}$



$$
\begin{aligned}
S[\sigma](x) &= \sum_j \int_{\Gamma_j} G(x,y)\sigma(y)da(y) \\
&= \sum_{\Gamma_j \in T_\eta(x)} \int_{\Gamma_j} G(x,y)\sigma(y)da(y) + \sum_{\Gamma_j \notin T_\eta(x)} \int_{\Gamma_j} G(x,y)\sigma(y)da(y) \\
&= \qquad S_{near}[\sigma](x) \qquad\qquad + \qquad S_{far}[\sigma](x)
\end{aligned}
$$

Precomputed in a target dependent manner

Target independent oversampled quadrature

Near quadrature

$$\int_{\Gamma_j} G(x,y)\sigma(y)da(y) = \int_B G(x,x^j(u,v))\sigma(u,v)J^j(u,v)dudv \qquad J^j(u,v) = \frac{\partial_u x^j \times \partial_v x^j}{|\partial_u x^j \times \partial_v x^j|}$$

σ^j_ℓ**,** $\ell = 1,2,\ldots n_b$ **denote the samples of the density on patch** Γ_j

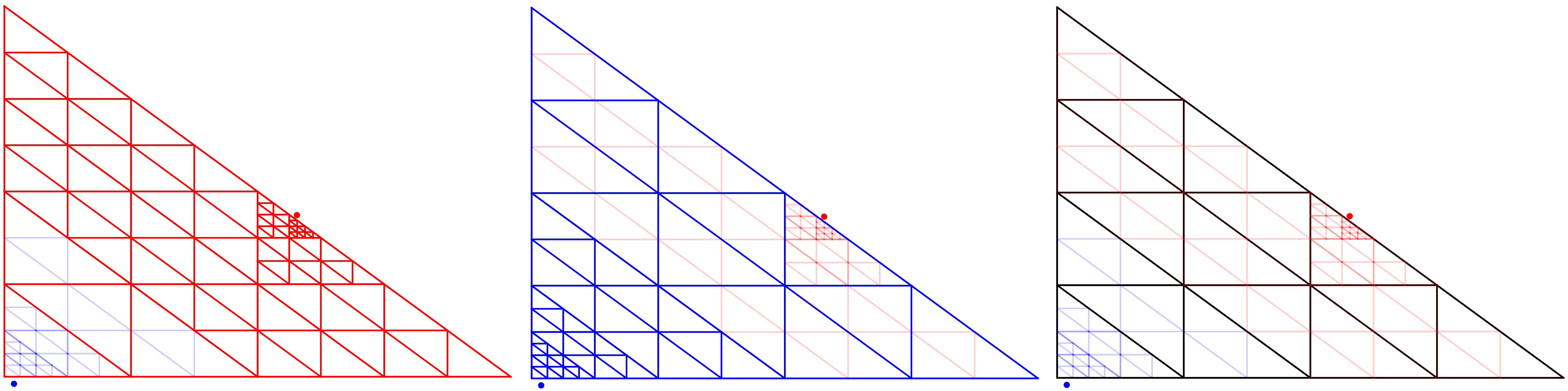
Find $a^j_\ell(x)$ **such that:** $\left| \sum_{j=1}^{n_b} a^j_\ell(x)\sigma^j_\ell - \int_B G(x,x^j(u,v))\sigma(u,v)J^j(u,v)\, dudv \right| < \varepsilon$

Suppose $P_j(u,v)$ **: orthogonal polynomials on** B

Instead compute: $b^j_\ell(x) = \int_B G(x,x^j)P_\ell(u,v)J^j(u,v)\, dudv \qquad \ell = 1,\ldots n_b$

$a^j_{1:n_b} = U \cdot b^j_{1:n_b}$ $\qquad$ $U \in \mathbb{R}^{n_b \times n_b}$ **is the coefficients to values matrix**

Near quadrature - optimizations



- Oversampled quadrature for $N_\eta(\Gamma_j) \backslash N_{\eta_1}(\Gamma_j)$ — avoids branching and can use BLAS routines for better performance. No need to suffer for high oversampling for all targets
- Request lower precision for adaptive integration for targets in $N_{\eta_1}(\Gamma_j)$

Far quadrature - oversampling estimation

- Identify 10 furthest targets in $N_\eta(\Gamma_j)$ (If $|N_\eta(\Gamma_j)| < 20$, then choose $|N_\eta(\Gamma_j)|/2$ targets from the list
- Add 15 (or 25 - $|N_\eta(\Gamma_j)|/2$ if $|N_\eta(\Gamma_j)| < 20$) randomly chosen targets on $\partial B_{\eta R_j}(c_j)$
- Let $F(\Gamma_j)$ denote the union of these two sets of targets
- Depending on the order (o) of the kernel (order = -1 for S, 0 for D, 1 for S'' and so on), consider the integrals $I_\ell[x] = \int_B \partial_n^{(o+1)} G(x, x^j(u,v)) P_\ell(u,v) J^j(u,v) du dv$
- Let $I_{\ell,q}[x]$ denote approximations to $I_\ell[x]$ computed using order q nodes on B
- Then q_j for patch Γ_j is given by

$$q_j = \min_q \left(\max_{\substack{0 \leq \ell < n_b \\ x \in F(\Gamma_j)}} |I_{\ell,q}[x] - I_{\ell,q+1}[x]| \right) < \varepsilon$$

Choice of η

- $\eta = 1.25$, if $p > 8$
- $\eta = 2$, if $4 < p \leq 8$
- $\eta = 2.75$, if $p \leq 4$

$p \backslash \eta$	1.25	2.00	2.75
2	24.1	7.45	5
3	10.6	4.67	3.26
4	8.76	2.8	2.8
6	3.92	2.14	1.4
8	2.39	1.25	1.25

(a) m

$p \backslash \eta$	1.25	2.00	2.75
2	23.6	65.6	137
3	50.4	134	276
4	82.3	222	458
6	173	466	961
8	298	798	1650

(b) α

- m: effective memory required per discretization node
- $\alpha = N_{over}/N$
- S_{near} : speed of quadrature generation in pts/sec/core
- S_{LP} : speed of layer potential evaluation in pts/sec/core

$p \backslash \eta$	1.25	2.00	2.75
2	6050	5280	4050
3	4820	3980	3240
4	3730	3490	2710
6	2070	1820	1590
8	1310	1180	956

(c) S_{NEAR}

$p \backslash \eta$	1.25	2.00	2.75
2	1200	2870	3790
3	2180	3440	3670
4	2210	4080	3460
6	3230	4270	3860
8	3740	4660	3500

(d) S_{LP}

Table 1: m,α,S_{NEAR}, and S_{LP} as a function of p and η

Performance results

$p \backslash \varepsilon$	$5 \cdot 10^{-3}$	$5 \cdot 10^{-4}$	$5 \cdot 10^{-7}$	$5 \cdot 10^{-10}$
2	1	1.62	5	9.58
3	0.701	1	3.26	7.15
4	0.6	1.03	2.8	4.77
6	0.714	0.87	2.14	3.15
8	0.778	1.17	2.39	2.53

(a) m

$p \backslash \varepsilon$	$5 \cdot 10^{-3}$	$5 \cdot 10^{-4}$	$5 \cdot 10^{-7}$	$5 \cdot 10^{-10}$
2	137	137	137	137
3	276	276	276	276
4	222	222	222	222
6	466	466	466	466
8	298	298	298	298

(b) α

$p \backslash \varepsilon$	$5 \cdot 10^{-3}$	$5 \cdot 10^{-4}$	$5 \cdot 10^{-7}$	$5 \cdot 10^{-10}$
2	10900	8800	4230	1290
3	7850	7810	3340	1170
4	8270	8410	3270	938
6	3260	3280	1810	773
8	2150	2100	1330	636

(c) S_{NEAR}

$p \backslash \varepsilon$	$5 \cdot 10^{-3}$	$5 \cdot 10^{-4}$	$5 \cdot 10^{-7}$	$5 \cdot 10^{-10}$
2	14000	10200	3710	1230
3	12900	10300	3670	1380
4	13300	10800	4110	2040
6	11600	10200	4160	2400
8	11400	8930	3730	2800

(d) S_{LP}

Table 2: m,α,s_1, and s_2 as a function of p and ε

- **m: effective memory required per discretization node**
- $\alpha = N_{over}/N$
- S_{near} **: speed of quadrature generation in pts/sec/core**
- S_{LP} **: speed of layer potential evaluation in pts/sec/core**

Order of convergence

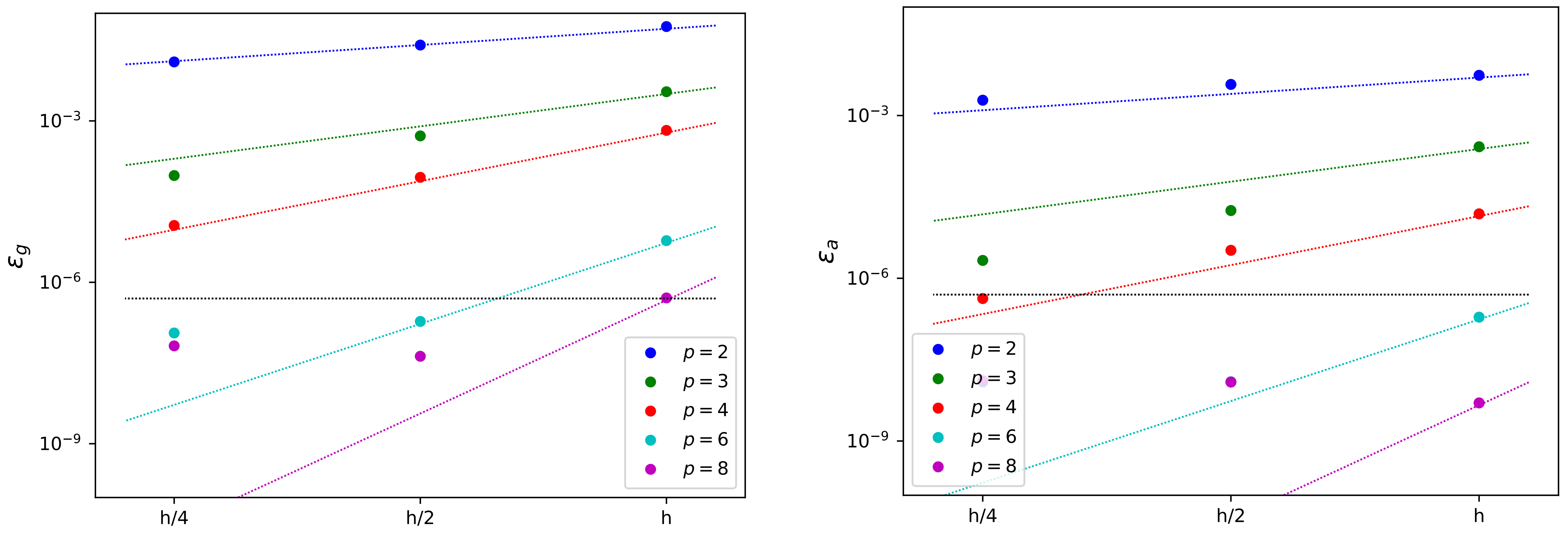
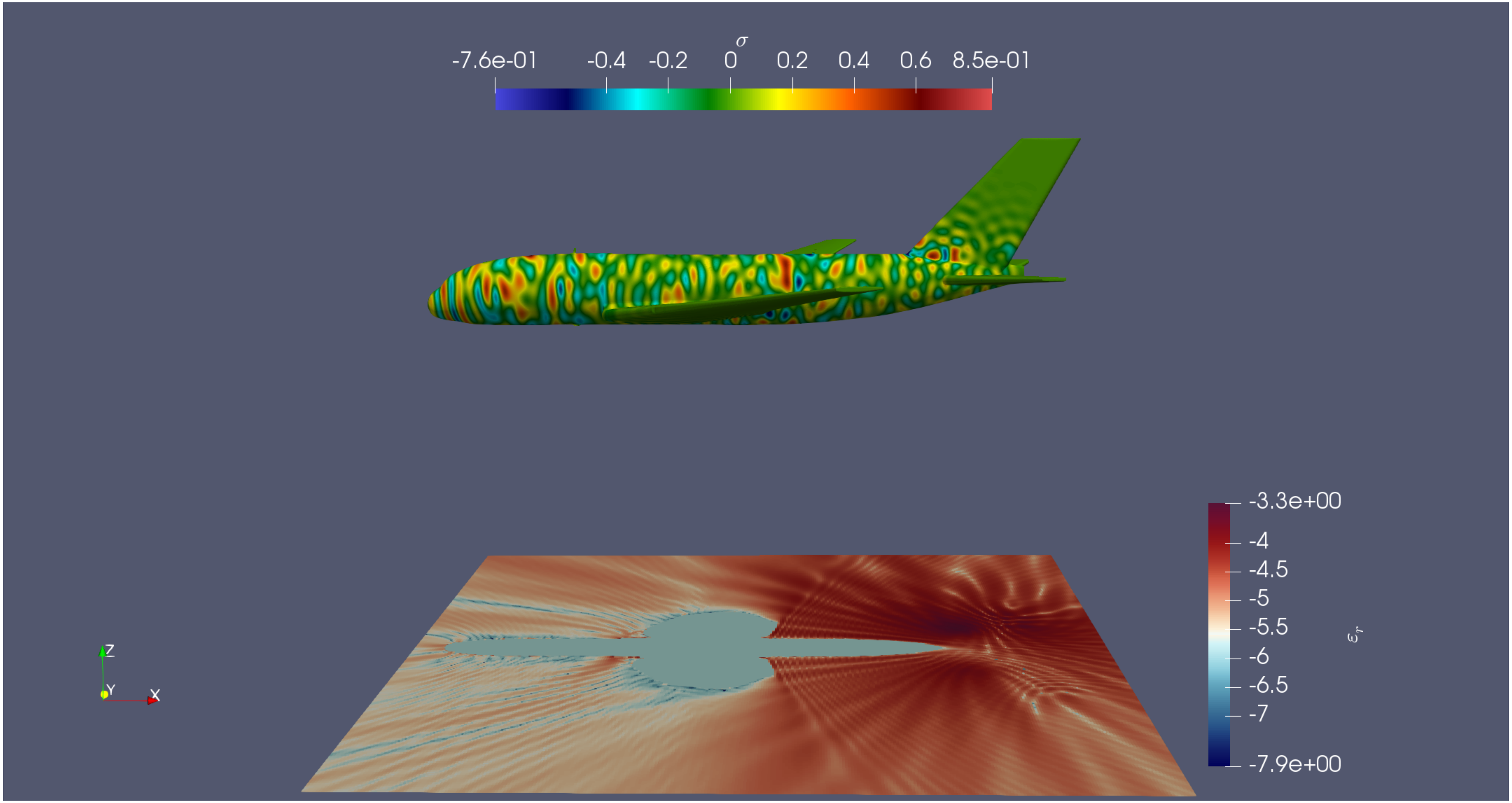


Figure 6: Left: Relative L^2 error in green's identity ε_g Right: relative L^∞ error in solution to integral equation ε_a. In both figures, the dashed colored lines are reference curves for h^{p-1} for the corresponding p. The dashed black line is a reference line for ε

Big example



σ
-7.6e-01
-0.4
-0.2
0
0.2
0.4
0.6
8.5e-01
-3.3e+00
-4
-4.5
-5
-5.5
-6
-6.5
-7
-7.9e+00
ε_r
Z
Y
X

Ways to get involved

- Request features at developer meetings
- Develop existing feature requests on the repo
  - Wrappers for other PDEs and representations
    - Maxwell: MFIE + CFIE + EFIE wrappers (#4)
    - Maxwell: NRCCIE (#7)
    - Maxwell: CSIE (#8)
    - Stokes : Combined field rep for velocity BVP (#9)
    - Stokes: Single layer for traction problem (#10)
    - Stokes: Mobility problem, combined field representation (#11)
    - Stokes: Mobility adjoint representation with single layer (#12)
  - Openmp optimizations for base routines
    - Cumulative sum (#2)
    - Convert (i,j) pairs to column sparse compressed format (#5)
  - Incorporate new quadrature schemes
    - Local QBX (#13)
    - Hedgehog (#14)
  - Incorporate new types of patches (B, associated discretization nodes, orthogonal polynomials)
    - Triangle T_0 with Gmsh nodes (#15)
  - I/O
    - Read in .msh files (#16)
    - Vtk files for plotting vector fields (#17)

List of open tasks
Developer process

Various structs in use

Geometry info (G):

- Npts: number of points
- Npatches: number of patches
- Norders(npatches): order of discretization of each patch
- iptype(npatches): type of patch, ipatch(i) = 1, triangular patch discretized using RV nodes as a map from the simplex (0,0),(1,0),(0,1)
- ixyzs(npatches + 1): location in array srcvals and srccoefs array where geometry info for patch i starts. Also ixyzs(i+1)-ixyzs(i) = number of points on patch
- srcvals(12,npts): x,y,z,dx/du,dy/du,dz/du,dx/dv,dy/dv,dz/dv, nx,ny,nz values
- srccoefs(9,npts): x,y,z,dx/du,dx/dv,dy/dv,dz/dv - koornwinder expansion coefficients

Oversampled geometry info (OG):

- Nptso: number of oversampled points
- Npatches: number of patches
- novers(npatches): order of oversampled discretization of each patch
- ixyzso(npatches + 1): location in array srcvalso array where geometry info for patch i starts. Also ixyzso(i+1)-ixyzso(i) = number of points oversampled points on patch
- srcvalso(12,nptso): x,y,z,dx/du,dy/du,dz/du,dx/dv,dy/dv,dz/dv, nx,ny,nz values
- ximats(nn): set of interpolation matrices to go from source patches to oversampled source patches ($nn = \sum_{j=1}^{\text{Npatches}} (ixyzso(j+1) - ixyzso(j)) \cdot (ixyzs(j+1) - ixyzs(j))$)
- ixmats(npatches) - location in ximats array where interpolation matrix for patch i starts

Target info (T):

- Ntarg: number of targets
- Ndtarg: leading dimension order for target arrays
- ipatch_id(ntarg): patch number if target is on surface, ipatch_id(i) = -1 if target is off-surface
- uvs_targ(2,ntarg): local u,v coordinates of targets on surface, irrelevant if target off surface
- targvals(ndtarg,ntarg): first three params must be target values

Kernel parameters (K):

- dpars_ker(ndd): real parameters
- zpars_ker(ndz): complex parameters
- ipars_ker(ndi): integer parameters

Near quadrature correction (N):

Stored in row-sparse compressed format as a list between targets and patches

- nnz - number of non-zero target-patch interactions
- col_ind(nnz) - list of patches corresponding to each target
- row_ptr(ntarg+1) - row_ptr(i) is the starting location in col_ind array where list of patches in the near field of target i start

if(row_ptr(i)<=j < row_ptr(i+1)), then target i, and patch (col_ind(j)) are in the near field of each other

- Nquad - number of non-zero entries in near-field quadrature array
- iquad(nnz) - iquad(i) is the location in the quadrature correction array where the matrix entries corresponding to the interaction in target i, and patch col_ind(j) start in wnear array
- wnear(nquad*ndimker) - near field quadrature correction array

Example: Consider the following matrix with 3 targets (rows) and 5 patches (columns), where $\times$ denotes a combination of patch and target which are handled through specialized corrections, and $-$ are the far-field targets

$$\begin{bmatrix} \times & - & - & - & \times \\ - & \times & - & \times & - \\ - & - & \times & \times & \times \end{bmatrix}$$

Then, for this example
row_ptr = [1,3,5,8]
col_ind = [1,5,2,4,3,4,5]

Quadrature parameters (QP):

- iquadtype - type of quadrature to use, current support for generalized gaussian quadrature for on patch targets + adaptive integration for rest of the targets
- r_0: radius of inner shell in the near field which is handled via adaptive integration, all targets in near field outside of r0 are handled via oversampled quadrature
- Internally set parameters not exposed to the user:
  - ε_{adap}: effective accuracy requested in adaptive integration
  - q_{order}: order of XG nodes used on each triangle in adaptive integration hierarchy
  - $n_{f,lev}$: number of levels of uniform refinement of standard simplex used in oversampled quadrature for targets outside sphere of radius r_0
  - $q_{order,f}$: order of XG nodes on each triangle for oversampled quadratures bit