2D boundary integral equations and the Nyström method

Alex Barnett[1] and **Fruzsina Agocs**[1]

Computational Tools 2024 BIE workshop. Day 1, 6/10/24

[1]Center for Computational Mathematics, Flatiron Institute, Simons Foundation

Integral equations on 1D interval

Given: i) function $\sigma(t)$ defined on interval $[0, 2\pi)$, periodic: $\sigma(2\pi) = \sigma(0)$, etc

ii) "kernel" function $k(t, s)$ defined on square $[0, 2\pi)^2$,

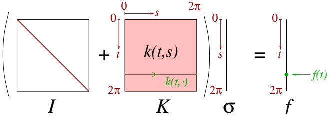
Integral *operator* K acts on σ to give another function $K\sigma$:

$$(K\sigma)(t) := \int_0^{2\pi} k(t, s)\sigma(s)ds, \quad t \in [0, 2\pi)$$

continuous analog of matrix-vector prod. $A\boldsymbol{x}$

Integral *equation*: $(I + K)\sigma = f$, ie

$$\sigma(t) + \int_0^{2\pi} k(t, s)\sigma(s)ds = f(t), \quad t \in [0, 2\pi)$$



analog of lin. sys. $A\boldsymbol{x} = \boldsymbol{b}$

Fredholm "second kind" (due to presence of I, otherwise called "first kind")

If kernel continuous, or "weakly" singular (integrable), K is *compact*:

- eigenvalues $(K\phi_k = \lambda_k\phi_k)$ discrete, with $\lim_{k\to\infty} \lambda_k = 0$

  unless some $\lambda_k = -1$, 2nd kind IE has at most one soln: Nul $(I + K) = \{0\}$

- Nul $(I + K) = \{0\} \;\Rightarrow\;$ *existence* of solution for *any* f Fredholm Alternative

  like square matrix (finite-dim), recall: uniqueness $\Rightarrow$ consistent for any RHS

Contrast 1st kind IE $K\sigma = f$ is ill-posed problem (unstable)!

Nyström discretization of 2nd kind IE on interval

Simplest quadrature for periodic funcs: periodic trapezoid rule (PTR)

$$\int_0^{2\pi} f(t)dt \approx \sum_{j=1}^N \frac{2\pi}{N} f\left(\frac{2\pi j}{N}\right) = \sum_{j=1}^N w_j f(t_j)$$

w_j=weights, t_j=nodes

For f smooth, superalgebraically convergent ("spectral"): error $= \mathcal{O}(N^{-p})$ for any p

Apply quad to integral in 2nd kind IE:

call the resulting approx soln $\tilde{\sigma}$

$$\tilde{\sigma}(t) + \sum_{j=1}^N k(t, t_j) w_j \tilde{\sigma}(t_j) = f(t), \quad t \in [0, 2\pi) \qquad (*)$$

Holds for all t; in particular, holds at each t_i, $i = 1, \ldots, N$, giving:

$$\sigma_i + \sum_{j=1}^N k(t_i, t_j) w_j \sigma_j = f(t_i), \quad i = 1, \ldots, N$$

where $\sigma_j := \tilde{\sigma}(t_j)$

Write as: $A\boldsymbol{\sigma} = \boldsymbol{f}$

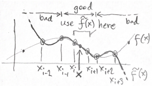
$N \times N$ lin sys, entries $a_{ij} = \delta_{ij} + k(t_i, t_j) w_j$, and $f_j := f(t_j)$

solve? dense direct $\mathcal{O}(N^3)$; dense iter. $\mathcal{O}(N^2)$; fast iter. $\approx \mathcal{O}(N)$; fast direct $\approx \mathcal{O}(N^{(d+1)/2})$

Why 2nd kind? eigs(A) accumulate only at $+1$, iterative fast conv.

Sometimes for BIE (eg, far-field eval), node values $\{\sigma_j\}_{j=1}^N$ suffice.
If not, interpolate from them to any $t \in [0, 2\pi)$. Two approaches:

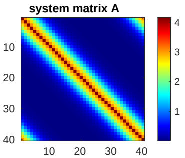
- either: rearrange (*) to give $\tilde{\sigma}(t) = \ldots$, called "Nyström interpolant" (rare)
- or (common): use local high-order Lagrange (or global spectral) interpolation:

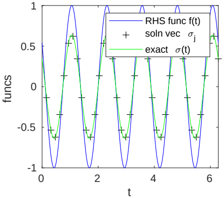


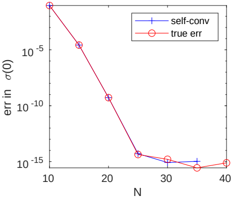
Demo Nyström on interval (1D)

day1/code/nyst1d_demos.m

```
kfun = @(t,s) exp(3*cos(t-s));                    % smooth convolutional kernel, periodic domain [0,2pi)
ffun = @(t) cos(5*t+1);                           % smooth data (RHS) func
N = 30;                                           % number of unknowns: should study convergence as N grows...
t = 2*pi/N*(1:N); w = 2*pi/N*ones(1,N);          % PTR nodes and weights, row vecs
A = eye(N) + bsxfun(kfun,t',t)*diag(w);           % identity plus fill k(t_i,t_j)w_j for i,j=1..N
rhs = ffun(t');                                   % col vec
sigmaj = A\rhs;                                   % dense direct square solve (pivoted LU), gives col vec
```







"self-convergence": use $N=40$ as "true"

f and k smooth

$\Rightarrow$ σ smooth

$\Rightarrow$ spectral conv?

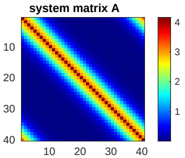
Thm. (Anselone, Kress,...): error at node values (and Nyström interpolant) same order as that of quadrature rule applied to integrand $k(t,\cdot)\sigma$.

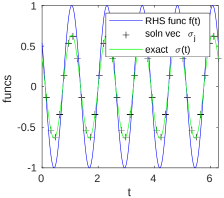
FLATIRON INSTITUTE

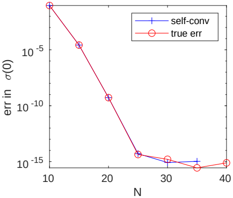
Demo Nyström on interval (1D)

day1/code/nyst1d_demos.m

```
kfun = @(t,s) exp(3*cos(t-s));           % smooth convolutional kernel, periodic domain [0,2pi)
ffun = @(t) cos(5*t+1);                  % smooth data (RHS) func
N = 30;                                  % number of unknowns: should study convergence as N grows...
t = 2*pi/N*(1:N); w = 2*pi/N*ones(1,N);  % PTR nodes and weights, row vecs
A = eye(N) + bsxfun(kfun,t',t)*diag(w);  % identity plus fill k(t_i,t_j)w_j for i,j=1..N
rhs = ffun(t');                          % col vec
sigmaj = A\rhs;                          % dense direct square solve (pivoted LU), gives col vec
```







"self-convergence": use N=40 as "true"

f and k smooth

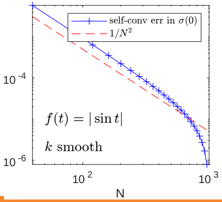
$\Rightarrow$ σ smooth

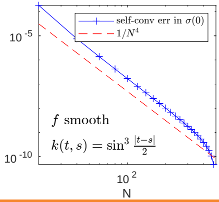
$\Rightarrow$ spectral conv?

Thm. (Anselone, Kress,...): error at node values (and Nyström interpolant) same order as that of quadrature rule applied to integrand $k(t,\cdot)\sigma$.

- Then, f or k nonsmooth?

worse (here *algebraic*) convergence using plain PTR rule:





Qu: why does order appear to improve at end?

Fundamental solution in $\mathbb{R}^2$

Eg PDE: Poisson eqn $\Delta u = g$ $\quad$ $\Delta := (\partial/\partial x_1)^2 + (\partial/\partial x_2)^2$ Laplacian

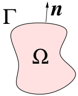
Notation: $\boldsymbol{x} := (x_1, x_2) \in \mathbb{R}^2$ is a point. This frees up $\boldsymbol{y} \in \mathbb{R}^2$ as another point (not y-coord!)

Not well-posed prob. unless add BC! BIEs are good for *homogeneous* PDEs (driving $g \equiv 0$)

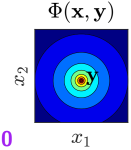
Eg well-posed* BVP:

$$\Delta u = 0 \text{ in } \Omega \qquad \text{PDE } (u \text{ harmonic})$$
$$u = f \text{ on } \Gamma \qquad \text{Dirichlet BC}$$

*exists, unique, continuous w.r.t. data



Laplace fundamental soln: $\Phi(\boldsymbol{x}, \boldsymbol{y}) = \frac{1}{2\pi} \log \frac{1}{r}$ $\quad$ where $r := \|\boldsymbol{x} - \boldsymbol{y}\|$



obeys $-\Delta_{\boldsymbol{x}} \Phi = -\Delta_{\boldsymbol{y}} \Phi = \delta_{\boldsymbol{x}}$ $\quad$ Φ harmonic except unit point-mass at $\mathbf{0}$

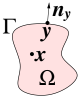
Normal $\boldsymbol{n}$ points outwards, $\|\boldsymbol{n}\| = 1$ $\quad$ normal deriv. notation $u_n := \boldsymbol{n} \cdot \nabla u$

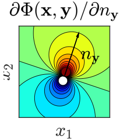
Green's 2nd identity: $\int_\Gamma v u_n - v_n u \, ds = \int_\Omega v \Delta u - (\Delta v) u \, d\boldsymbol{y}$ $\quad$ calculus

warm-up: set $u =$ BVP soln, $v \equiv 1$, G2I becomes $\int_\Gamma u_n ds \; - 0 = 0 - 0$: so u has zero *flux*
more fun: fix "target" $\boldsymbol{x} \in \Omega$, let $v = \Phi(\boldsymbol{x}, \cdot)$, G2I gives:

Green's representation formula:

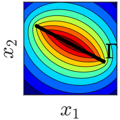
$$\int_\Gamma \Phi(\boldsymbol{x}, \boldsymbol{y}) u_n(\boldsymbol{y}) - \frac{\partial \Phi(\boldsymbol{x}, \boldsymbol{y})}{\partial \boldsymbol{n}_{\boldsymbol{y}}} u(\boldsymbol{y}) \, ds_{\boldsymbol{y}} = u(\boldsymbol{x}) \quad \text{for } \boldsymbol{x} \in \Omega$$

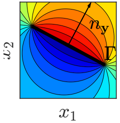




Gets soln from "Cauchy data" $(u, u_n)|_\Gamma$
also versions for Helmholtz, Stokes, Maxwell,...

Layer potentials and their jump relations





Representations of harmonic functions off a curve Γ: "density" σ

Single-layer potential $(\mathcal{S}\sigma)(\boldsymbol{x}) := \int_\Gamma \Phi(\boldsymbol{x},\boldsymbol{y})\sigma(\boldsymbol{y})ds_{\boldsymbol{y}}$ charge sheet

Double-layer potential $(\mathcal{D}\sigma)(\boldsymbol{x}) := \int_\Gamma \frac{\partial\Phi(\boldsymbol{x},\boldsymbol{y})}{\partial \boldsymbol{n_y}}\sigma(\boldsymbol{y})ds_{\boldsymbol{y}}$ dipole sheet

interior (-) / exterior (+) limits:

$$u^\pm(\boldsymbol{x}) := \lim_{h\to 0^+} u(\boldsymbol{x} \pm h\boldsymbol{n_x})$$
$$u_n^\pm(\boldsymbol{x}) := \lim_{h\to 0^+} \boldsymbol{n_x}\cdot\nabla u(\boldsymbol{x} \pm h\boldsymbol{n_x})$$

Jump relations:

$(\mathcal{S}\sigma)^\pm =$	$S\sigma$	S (Roman font) means *restriction* of $\mathcal{S}$ to Γ: a bdry int. op.
$(\mathcal{D}\sigma)^\pm =$	$(D \pm I/2)\sigma$	jump in potential equal to σ; D restriction to Γ in P.V. sense
$(\mathcal{S}\sigma)_n^\pm =$	$(D^T \mp I/2)\sigma$	jump in normal derivative
$(\mathcal{D}\sigma)_n^\pm =$	$T\sigma$	T hypersingular, kernel $\partial^2\Phi(\boldsymbol{x},\boldsymbol{y})/\partial\boldsymbol{n_x}\partial\boldsymbol{n_y} \sim 1/r^2$

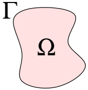
- D smooth kernel on smooth Γ, while S always log (weakly) singular

Recap GRF in LP notation: u harmonic in Ω $\Rightarrow$ $\mathcal{S}u_n^- - \mathcal{D}u^- = u$ in Ω

Converting BVP to BIE and solving

Say wish to solve interior Dirichlet Laplace BVP:

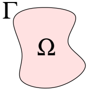
$$\Delta u = 0 \text{ in } \Omega \qquad \text{PDE}$$
$$u^- = f \text{ on } \Gamma \qquad \text{BC}$$



Converting BVP to BIE and solving

Say wish to solve interior Dirichlet Laplace BVP:

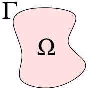
$$\Delta u = 0 \text{ in } \Omega \qquad \text{PDE}$$
$$u^- = f \text{ on } \Gamma \qquad \text{BC}$$



Pick **representation**: $u = \mathcal{D}\sigma$, look up its **JR** for BC: $u^- = (D - I/2)\sigma$

Converting BVP to BIE and solving

Say wish to solve interior Dirichlet Laplace BVP: $\Delta u = 0$ in Ω PDE
$u^- = f$ on Γ BC



Pick **representation**: $u = \mathcal{D}\sigma$, look up its **JR** for BC: $u^- = (D - I/2)\sigma$

Insert the BC to get BIE: $(I - 2D)\sigma = -2f$ scaled to 2nd kind form

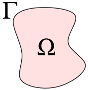
This shows: let σ solve BIE, then $u = \mathcal{D}\sigma$ solves BVP (i.e., no spurious solns)

But how know *all* solns u of this form?

Converting BVP to BIE and solving

Say wish to solve interior Dirichlet Laplace BVP: $\Delta u = 0$ in Ω PDE

$u^- = f$ on Γ BC



Pick **representation**: $u = \mathcal{D}\sigma$, look up its **JR** for BC: $u^- = (D - I/2)\sigma$

Insert the BC to get BIE: $(I - 2D)\sigma = -2f$ scaled to 2nd kind form

This shows: let σ solve BIE, then $u = \mathcal{D}\sigma$ solves BVP (i.e., no spurious solns)

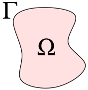
But how know *all* solns u of this form? Fred. Alt.: BIE *has* soln $\forall f$! BVP & BIE equivalent ☺

(had we picked $u = \mathcal{S}\sigma$, would get 1st kind, poorly conditioned but can have its uses)

Converting BVP to BIE and solving

Say wish to solve interior Dirichlet Laplace BVP: $\Delta u = 0$ in Ω PDE

$u^- = f$ on Γ BC



Pick **representation**: $u = \mathcal{D}\sigma$, look up its **JR** for BC: $u^- = (D - I/2)\sigma$

Insert the BC to get BIE: $(I - 2D)\sigma = -2f$ scaled to 2nd kind form

This shows: let σ solve BIE, then $u = \mathcal{D}\sigma$ solves BVP (i.e., no spurious solns)

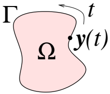
But how know *all* solns u of this form? Fred. Alt.: BIE *has* soln $\forall f$! BVP & BIE equivalent ☺

(had we picked $u = \mathcal{S}\sigma$, would get 1st kind, poorly conditioned but can have its uses)

Above BIE expressed on Γ using arc-length measure ds. Usually not how Γ described...

Converting BVP to BIE and solving

Say wish to solve interior Dirichlet Laplace BVP: $\Delta u = 0$ in Ω PDE
$u^- = f$ on Γ BC



Pick **representation**: $u = \mathcal{D}\sigma$, look up its **JR** for BC: $u^- = (D - I/2)\sigma$

Insert the BC to get BIE: $(I - 2D)\sigma = -2f$ scaled to 2nd kind form

This shows: let σ solve BIE, then $u = \mathcal{D}\sigma$ solves BVP (i.e., no spurious solns)

But how know *all* solns u of this form? Fred. Alt.: BIE *has* soln $\forall f$! BVP & BIE equivalent ☺

(had we picked $u = \mathcal{S}\sigma$, would get 1st kind, poorly conditioned but can have its uses)

Above BIE expressed on Γ using arc-length measure ds. Usually not how Γ described...

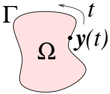
Parameterize the bdry $\boldsymbol{y}(t)$ $\boldsymbol{y} : \mathbb{R} \to \mathbb{R}^2$, 2π-periodic, $\Gamma = \{\boldsymbol{y}(t) : t \in [0, 2\pi)\}$

change variable $ds_{\boldsymbol{y}} = \|\boldsymbol{y}'(t)\| dt$ abuse notation $\sigma(t) = \sigma(\boldsymbol{y}(t))$

Converting BVP to BIE and solving

Say wish to solve interior Dirichlet Laplace BVP: $\Delta u = 0$ in Ω PDE

$u^- = f$ on Γ BC



Pick **representation**: $u = \mathcal{D}\sigma$, look up its **JR** for BC: $u^- = (D - I/2)\sigma$

Insert the BC to get BIE: $(I - 2D)\sigma = -2f$ scaled to 2nd kind form

This shows: let σ solve BIE, then $u = \mathcal{D}\sigma$ solves BVP (i.e., no spurious solns)

But how know *all* solns u of this form? Fred. Alt.: BIE *has* soln $\forall f$! BVP & BIE equivalent ☺

(had we picked $u = \mathcal{S}\sigma$, would get 1st kind, poorly conditioned but can have its uses)

Above BIE expressed on Γ using arc-length measure ds. Usually not how Γ described...

Parameterize the bdry $\boldsymbol{y}(t)$ $\boldsymbol{y} : \mathbb{R} \to \mathbb{R}^2$, 2π-periodic, $\Gamma = \{\boldsymbol{y}(t) : t \in [0, 2\pi)\}$

change variable $ds_{\boldsymbol{y}} = \|\boldsymbol{y}'(t)\| dt$ abuse notation $\sigma(t) = \sigma(\boldsymbol{y}(t))$

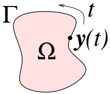
Get 1D IE: $\sigma(t) - 2\int_0^{2\pi} \frac{\partial \Phi(\boldsymbol{y}(t), \boldsymbol{y}(s))}{\partial \boldsymbol{n}_{\boldsymbol{y}(s)}} \sigma(s) \|\boldsymbol{y}'(s)\| ds = -2f(t), \quad t \in [0, 2\pi)$

familiar form $(I + K)\sigma = -2f$, with kernel $k(s,t) = \frac{-2}{2\pi} \frac{\boldsymbol{n}_{\boldsymbol{y}(s)} \cdot (\boldsymbol{y}(t) - \boldsymbol{y}(s))}{\|\boldsymbol{y}(t) - \boldsymbol{y}(s)\|^2} \|\boldsymbol{y}'(s)\|$

formula on diagonal: $k(t,t) = \lim_{s \to t} k(t,s) = \kappa(t)/2\pi$, κ curvature of Γ (check!)

Converting BVP to BIE and solving

Say wish to solve interior Dirichlet Laplace BVP: $\Delta u = 0$ in Ω PDE
$u^- = f$ on Γ BC



Pick **representation**: $u = \mathcal{D}\sigma$, look up its **JR** for BC: $u^- = (D - I/2)\sigma$

Insert the BC to get BIE: $(I - 2D)\sigma = -2f$ scaled to 2nd kind form

This shows: let σ solve BIE, then $u = \mathcal{D}\sigma$ solves BVP (i.e., no spurious solns)

But how know *all* solns u of this form? Fred. Alt.: BIE *has* soln $\forall f$! BVP & BIE equivalent ☺

(had we picked $u = \mathcal{S}\sigma$, would get 1st kind, poorly conditioned but can have its uses)

Above BIE expressed on Γ using arc-length measure ds. Usually not how Γ described...

Parameterize the bdry $\boldsymbol{y}(t)$ $\boldsymbol{y} : \mathbb{R} \to \mathbb{R}^2$, 2π-periodic, $\Gamma = \{\boldsymbol{y}(t) : t \in [0, 2\pi)\}$

change variable $ds_{\boldsymbol{y}} = \|\boldsymbol{y}'(t)\| dt$ abuse notation $\sigma(t) = \sigma(\boldsymbol{y}(t))$

Get 1D IE: $\sigma(t) - 2\int_0^{2\pi} \frac{\partial\Phi(\boldsymbol{y}(t),\boldsymbol{y}(s))}{\partial \boldsymbol{n}_{\boldsymbol{y}(s)}} \sigma(s) \|\boldsymbol{y}'(s)\| ds = -2f(t), \quad t \in [0, 2\pi)$

familiar form $(I + K)\sigma = -2f$, with kernel $k(s,t) = \frac{-2}{2\pi} \frac{\boldsymbol{n}_{\boldsymbol{y}(s)} \cdot (\boldsymbol{y}(t) - \boldsymbol{y}(s))}{\|\boldsymbol{y}(t) - \boldsymbol{y}(s)\|^2} \|\boldsymbol{y}'(s)\|$

formula on diagonal: $k(t,t) = \lim_{s \to t} k(t,s) = \kappa(t)/2\pi$, κ curvature of Γ (check!)

Now Nyström discretize with PTR, solve lin. sys. for $\boldsymbol{\sigma} := \{\sigma_j\}_{j=1}^N$

Finally evaluate soln: $u(\boldsymbol{x}) = (\mathcal{D}\sigma)(\boldsymbol{x}) \overset{\text{PTR}}{\approx} \sum_{j=1}^N \frac{\boldsymbol{n}_{\boldsymbol{y}(t_j)} \cdot (\boldsymbol{x} - \boldsymbol{y}(t_j))}{2\pi \|\boldsymbol{x} - \boldsymbol{y}(t_j)\|^2} \|\boldsymbol{y}'(t_j)\| w_j \sigma_j$

Interior Laplace Dirichlet BVP solve demo

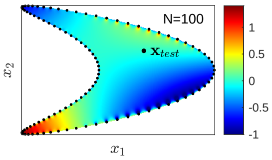
demo_lapintdir.m

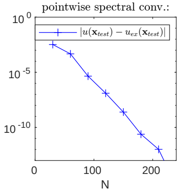
```
a=0.7; b=1.0;                                                      % shape params (note a=1,b=0 unit circle)
Y = @(t) [a*cos(t)+b*cos(2*t); sin(t)];                            % kite parameterization y(t)
Yp = @(t) [-a*sin(t)-2*b*sin(2*t); cos(t)];                        % y', analytic
Ypp = @(t) [-a*cos(t)-4*b*cos(2*t); -sin(t)];                      % y'', analytic
N = 100;
t = 2*pi/N*(1:N); w = 2*pi/N*ones(1,N);                            % PTR nodes & weights
y = Y(t);                                                          % bdry nodes, 2-by-N
n = [0 1;-1 0]*Yp(t); speed = sqrt(sum(n.^2,1)); n = n./speed;     % bdry normals
kappa = -sum(Ypp(t) .* n,1)./speed.^2;                             % bdry curvatures
r1 = y(1,:)'-y(1,:); r2 = y(2,:)'-y(2,:);                          % matrix of r=x-y (two vec cmpnts)
A = (-1/pi)*(n(1,:).*r1 + n(2,:).*r2) ./ (r1.^2+r2.^2);            % off-diag (-1/pi) n.r/r^2
A(diagind(A)) = kappa/(2*pi);                                      % overwrite diag elements
A = eye(N) + A*diag(speed.*w);                                     % note Id gets no "speed weights"
uex = @(x) ([1 0]*x) .* ([0 1]*x-0.3);                             % test u(x) = x_1(x_2-0.3), not symmetric!
f = @(t) uex(Y(t));                                                % read off its Dirichlet data
rhs = -2*f(t)';
sigma = A\rhs;                                                     % solve. Leave u = D.sigma eval to reader
```

Interior Laplace Dirichlet BVP solve demo

demo_lapintdir.m

```
a=0.7; b=1.0;                                                        % shape params (note a=1,b=0 unit circle)
Y = @(t) [a*cos(t)+b*cos(2*t); sin(t)];                              % kite parameterization y(t)
Yp = @(t) [-a*sin(t)-2*b*sin(2*t); cos(t)];                          % y', analytic
Ypp = @(t) [-a*cos(t)-4*b*cos(2*t); -sin(t)];                        % y'', analytic
N = 100;
t = 2*pi/N*(1:N); w = 2*pi/N*ones(1,N);                              % PTR nodes & weights
y = Y(t);                                                            % bdry nodes, 2-by-N
n = [0 1;-1 0]*Yp(t); speed = sqrt(sum(n.^2,1)); n = n./speed;       % bdry normals
kappa = -sum(Ypp(t) .* n,1)./speed.^2;                               % bdry curvatures
r1 = y(1,:)'-y(1,:); r2 = y(2,:)'-y(2,:);                            % matrix of r=x-y (two vec cmpnts)
A = (-1/pi)*(n(1,:).*r1 + n(2,:).*r2) ./ (r1.^2+r2.^2);              % off-diag (-1/pi) n.r/r^2
A(diagind(A)) = kappa/(2*pi);                                        % overwrite diag elements
A = eye(N) + A*diag(speed.*w);                                       % note Id gets no "speed weights"
uex = @(x) ([1 0]*x) .* ([0 1]*x-0.3);                               % test u(x) = x_1(x_2-0.3), not symmetric!
f = @(t) uex(Y(t));                                                  % read off its Dirichlet data
rhs = -2*f(t)';
sigma = A\rhs;                                                       % solve. Leave u = D.sigma eval to reader
```

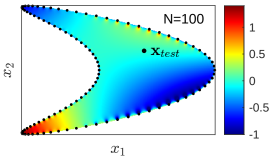


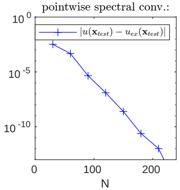


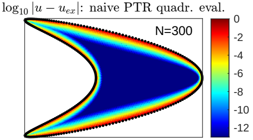
Interior Laplace Dirichlet BVP solve demo

demo_lapintdir.m

```
a=0.7; b=1.0;                                          % shape params (note a=1,b=0 unit circle)
Y = @(t) [a*cos(t)+b*cos(2*t); sin(t)];                % kite parameterization y(t)
Yp = @(t) [-a*sin(t)-2*b*sin(2*t); cos(t)];            % y', analytic
Ypp = @(t) [-a*cos(t)-4*b*cos(2*t); -sin(t)];          % y'', analytic
N = 100;
t = 2*pi/N*(1:N); w = 2*pi/N*ones(1,N);                % PTR nodes & weights
y = Y(t);                                              % bdry nodes, 2-by-N
n = [0 1;-1 0]*Yp(t); speed = sqrt(sum(n.^2,1)); n = n./speed;   % bdry normals
kappa = -sum(Ypp(t) .* n,1)./speed.^2;                 % bdry curvatures
r1 = y(1,:)'-y(1,:); r2 = y(2,:)'-y(2,:);              % matrix of r=x-y (two vec cmpnts)
A = (-1/pi)*(n(1,:).*r1 + n(2,:).*r2) ./ (r1.^2+r2.^2);  % off-diag (-1/pi) n.r/r^2
A(diagind(A)) = kappa/(2*pi);                          % overwrite diag elements
A = eye(N) + A*diag(speed.*w);                         % note Id gets no "speed weights"
uex = @(x) ([1 0]*x) .* ([0 1]*x-0.3);                 % test u(x) = x_1(x_2-0.3), not symmetric!
f = @(t) uex(Y(t));                                    % read off its Dirichlet data
rhs = -2*f(t)';
sigma = A\rhs;                                         % solve. Leave u = D.sigma eval to reader
```





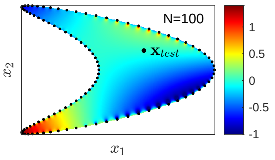


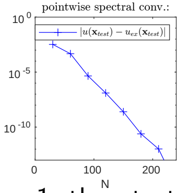
"5h" rule:
special eval.
quadratures
can fix near Γ
(Helsing, QBX...)

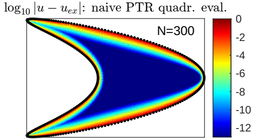
Interior Laplace Dirichlet BVP solve demo

demo_lapintdir.m

```
a=0.7; b=1.0;                                           % shape params (note a=1,b=0 unit circle)
Y = @(t) [a*cos(t)+b*cos(2*t); sin(t)];                 % kite parameterization y(t)
Yp = @(t) [-a*sin(t)-2*b*sin(2*t); cos(t)];             % y', analytic
Ypp = @(t) [-a*cos(t)-4*b*cos(2*t); -sin(t)];           % y'', analytic
N = 100;
t = 2*pi/N*(1:N); w = 2*pi/N*ones(1,N);                 % PTR nodes & weights
y = Y(t);                                               % bdry nodes, 2-by-N
n = [0 1;-1 0]*Yp(t); speed = sqrt(sum(n.^2,1)); n = n./speed;  % bdry normals
kappa = -sum(Ypp(t) .* n,1)./speed.^2;                  % bdry curvatures
r1 = y(1,:)'-y(1,:); r2 = y(2,:)'-y(2,:);               % matrix of r=x-y (two vec cmpnts)
A = (-1/pi)*(n(1,:).*r1 + n(2,:).*r2) ./ (r1.^2+r2.^2); % off-diag (-1/pi) n.r/r^2
A(diagind(A)) = kappa/(2*pi);                           % overwrite diag elements
A = eye(N) + A*diag(speed.*w);                          % note Id gets no "speed weights"
uex = @(x) ([1 0]*x) .* ([0 1]*x-0.3);                  % test u(x) = x_1(x_2-0.3), not symmetric!
f = @(t) uex(Y(t));                                     % read off its Dirichlet data
rhs = -2*f(t)';
sigma = A\rhs;                                          % solve. Leave u = D.sigma eval to reader
```







"5h" rule:
special eval.
quadratures
can fix near Γ
(Helsing, QBX...)

Debug: $\sigma \equiv -1 \;\Rightarrow\; u \equiv 1$, then test data from (generic!) soln u, and...

1. check/plot $\boldsymbol{n}$, κ. First test unit circle!
2. check Nyström matrix smooth at diag (*before* add I)

FLATIRON INSTITUTE

Indirect vs direct formulations

using Laplace interior Dirichlet BVP

So far "indirect" BIE: pick representation (eg $u = \mathcal{D}\sigma$), get BIE from JRs

Alternative is "direct": take limit of GRF on Γ, rearrange to get BIE:

GRF $u = \mathcal{S}u^- - \mathcal{D}u_n^- \quad \xrightarrow{\text{JRs}} \quad u_n^- = (D^T + I/2)u_n^- - Tu^- \quad \xrightarrow{\text{BC}} \quad (D^T - I/2)u_n^- = Tf$

Solve BIE for u_n^-, eval u via GRF (= two layer potential evals)

Notice BIO $(D^T - I/2)$ **adjoint** of that for indirect $(D - I/2)$ general fact

spectra, hence iterative convergence rates, the same

Compare pros and cons:

indirect BIE	direct BIE
unknown density (unphysical)	unknown is physical
RHS is plain data	RHS needs BIO apply to data
eval the representation (may be simpler)	eval the GRF

- technical differences for domains with corners (Hoskins–Rachh...)

Generalizations: Exterior, Neumann

some laplace aspects, confusions w/ BVP, briefly

notation for exterior domain

subtlety of decay in 2D

For int Neu or ext Dir, need kernel $k(t, s) + 1$

3. Neumann BC case: contour lines orthog to Γ

Helmholtz

$(\Delta + \kappa^2)u = 0$ arises from scalar wave equation $u_{tt} - \Delta u = 0$
κ "wavenumber"; wavelength $\lambda = 2\pi/\kappa$
Also used for 2D Maxwell (z-invar); TE vs TM
Dirichlet BC = PEC in TE (check)
Scattering formalism:
$u^{\text{tot}} = u^{\text{inc}} + u$.
BVP for u (3 lines: PDE, BC, SRC)
Solve BVP for u via PTR + Nyström

4. debug BVP with known data from a radiative soln sources inside Ω

Helmholtz transmission BVP

refractive indices in Ω vs exterior
DECIDE IF TRANSMISSION
Matching? (more effort, needs $\mathcal{S}\sigma + \mathcal{D}\tau \ldots$)

Helmholtz

fund sol,

Dirichlet demo, plots only

(don't demo CFIE since requires S w/ log-singularity). Demo DLP only,

see $1/N^3$ conv if use naive PTR with correct diag limit (see M126 HW?)

Getting spectral-acc Nyström for log-singular kernels: beyond today.

eg kernel-split or product quadratures (Kress, Helsing,...)

close-eval: kernel-split, QBX, etc.

see libraries: chunkie, BIE2D, etc

More debug ideas

TO DISCARD

Other tests:

5. Test SLP & DLP evaluators via GRF for any harmonic u in Ω

Recap

*** TO EDIT WITH FRUZSINA:

BIE involves several steps: write out yourself + try HW exer. in repo

- we covered basics for one smooth curve in 2D

  code yourself, if only to understand what libraries do better!
- Interior/exterior, Laplace/Helmholtz/etc, somewhat simple changes

  spectral conv once you have good log-singular kernel quadrature
- Fancier quadratures needed for singular kernels and/or close eval
- Transmission BVPs may need twice the unknowns
- Nyström is not only way to discretize: Galerkin, collocation

  but: simplest and no less accurate

Resources

Many numerical analysis (mathematics heavy). Somewhat accessible:

- *Linear Integral Equations*, R. Kress, (1999, Springer). Ch. 6 & 12.
- *The Numerical Solution of Integral Equations of the Second Kind*, K. E. Atkinson, (1997, CUP).

Fewer on the practical/tutorial side, few with modern devels:

- "High-order accurate methods for Nyström discretization of integral equations on smooth curves in the plane", S Hao, AH Barnett, PG Martinsson, P Young. *Adv. Comput. Math.* **40**, 245–272 (2014).

focuses on quadrature for logarithmic singularities, eg SLP, Helmholtz

- `https://users.flatironinstitute.org/~ahb/BIE/`
- `https://github.com/ahbarnett/BIEbook` in progress...

FLATIRON INSTITUTE