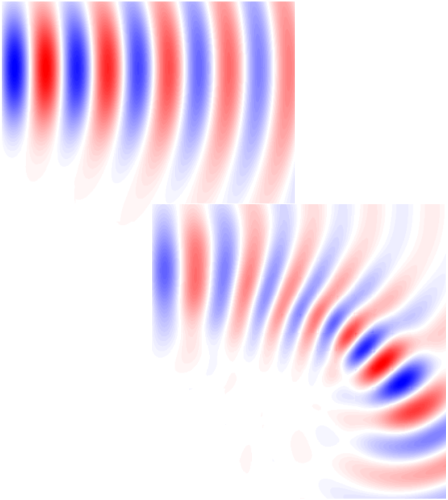


Fast Multipole Methods for Continuous Charge Densities

Travis Askham — New Jersey Institute of Technology
SIAM Conference on Image Science 2020 (IS20), via Zoom.




Joint work with

- Libin Lu (Flatiron Institute)
- Manas Rachh (Flatiron Institute)
- Alex Townsend (Cornell)
- Leslie Greengard (NYU)

This work was supported by the Flatiron Institute.

Joint work with

- Libin Lu (Flatiron Institute)
- Manas Rachh (Flatiron Institute)
- Alex Townsend (Cornell)
- Leslie Greengard (NYU)

`github.com/flatironinstitute/FMM3D`

`askham@njit.edu`

This work was supported by the Flatiron Institute.

Outline

Outline

- Example: scattering in variable media

Outline

- Example: scattering in variable media
- Box codes for volume integrals

Outline

- Example: scattering in variable media
- Box codes for volume integrals
  - Fast multipole method overview

Outline

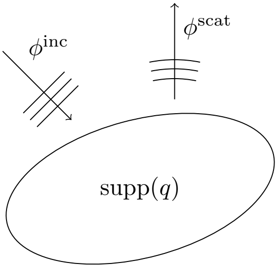
- Example: scattering in variable media
- Box codes for volume integrals
    - Fast multipole method overview
    - On-demand quadrature generation scheme

Outline

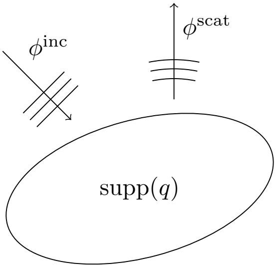
- Example: scattering in variable media
- Box codes for volume integrals
  - Fast multipole method overview
  - On-demand quadrature generation scheme
- Future work

Scattering in variable media

$$(\Delta + k^2(1+q(\mathbf{x})))\phi = 0$$



Scattering in variable media



$$(\Delta + k^2(1 + q(\mathbf{x})))\phi = 0$$

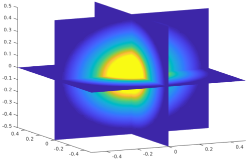
Setting $\phi = \phi^{\mathrm{inc}} + \phi^{\mathrm{scat}}$,

$$(\Delta+k^2(1+q(\mathbf{x})))\phi^{\mathrm{scat}} = -k^2 q(\mathbf{x})\phi^{\mathrm{inc}} \ .$$

- ϕ^{inc} is an *incident* wave which satisfies the constant coefficient Helmholtz equation (e.g. a plane wave)
- ϕ^{scat} is the *scattered* wave which satisfies an outgoing condition at infinity.

Scattering example[1]

- Let $q(\mathbf{x})$ correspond to an "Eaton" lens, which bends light through an angle

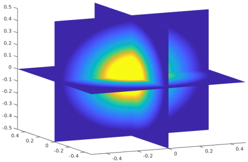


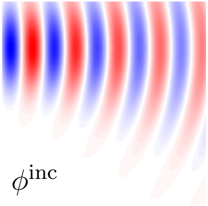
[1] Vico, Greengard, and Ferrando 2016; Danner and Leonhardt 2009.

Scattering example[1]

- Let $q(\mathbf{x})$ correspond to an "Eaton" lens, which bends light through an angle
- Let ϕ^{inc} be a Gaussian beam

$$\phi^{\text{inc}}(\mathbf{x}) = G_k(\mathbf{x}, \mathbf{z}) e^{-k/2} \;,$$
$$\mathbf{z} = (x_0 + i/2, y_0, z_0)$$





ϕ^{inc}

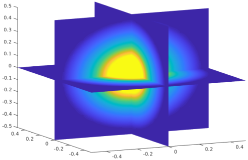
[1] Vico, Greengard, and Ferrando 2016; Danner and Leonhardt 2009.

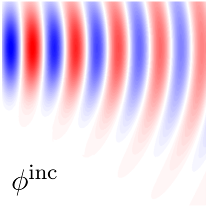
Scattering example[1]

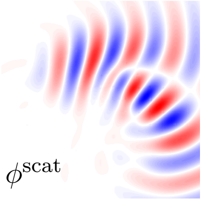
- Let $q(\mathbf{x})$ correspond to an "Eaton" lens, which bends light through an angle
- Let ϕ^{inc} be a Gaussian beam

$$\phi^{\text{inc}}(\mathbf{x}) = G_k(\mathbf{x}, \mathbf{z})e^{-k/2} \ ,$$
$$\mathbf{z} = (x_0 + i/2, y_0, z_0)$$

- Solve for scattered field







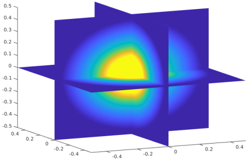
[1] Vico, Greengard, and Ferrando 2016; Danner and Leonhardt 2009.

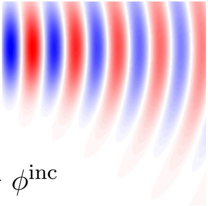
Scattering example[1]

- Let $q(\mathbf{x})$ correspond to an "Eaton" lens, which bends light through an angle
- Let ϕ^{inc} be a Gaussian beam

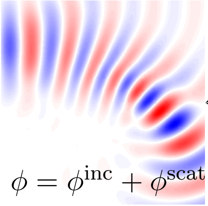
$$\phi^{\mathrm{inc}}(\mathbf{x}) = G_k(\mathbf{x}, \mathbf{z})e^{-k/2} \,,$$
$$\mathbf{z} = (x_0 + i/2, y_0, z_0)$$

- Solve for scattered field

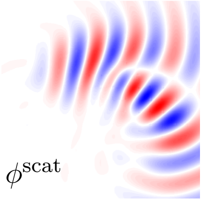




ϕ^{inc}



$\phi = \phi^{\mathrm{inc}} + \phi^{\mathrm{scat}}$



ϕ^{scat}

[1]Vico, Greengard, and Ferrando 2016; Danner and Leonhardt 2009.

Integral equation formulation

$$(\Delta + k^2(1 + q(\mathbf{x})))\phi^{\mathrm{scat}} = -k^2 q(\mathbf{x})\phi^{\mathrm{inc}}$$

Integral equation formulation

$$(\Delta + k^2(1+q(\mathbf{x})))\phi^{\text{scat}} = -k^2 q(\mathbf{x})\phi^{\text{inc}}$$

Represent ϕ^{scat} as a *volume integral*, i.e.

$$\phi^{\text{scat}}(\mathbf{x}) = V[\sigma](\mathbf{x}) = \int_\Omega G_k(\mathbf{x},\mathbf{y})\sigma(\mathbf{y})\,dv\ , \ \ G_k(\mathbf{x},\mathbf{y}) = \begin{cases} \frac{e^{ik|\mathbf{x}-\mathbf{y}|}}{4\pi|\mathbf{x}-\mathbf{y}|} \\ \\ \frac{iH_0^{(1)}(k|\mathbf{x}-\mathbf{y}|)}{4} \end{cases}$$

where $\operatorname{supp}(q) \subset \Omega$.

Integral equation formulation

$$(\Delta + k^2(1+q(\mathbf{x})))\phi^{\mathrm{scat}} = -k^2 q(\mathbf{x})\phi^{\mathrm{inc}}$$

Represent ϕ^{scat} as a *volume integral*, i.e.

$$\phi^{\mathrm{scat}}(\mathbf{x}) = V[\sigma](\mathbf{x}) = \int_\Omega G_k(\mathbf{x},\mathbf{y})\sigma(\mathbf{y})\,dv\ ,\ \ G_k(\mathbf{x},\mathbf{y}) = \begin{cases} \frac{e^{ik|\mathbf{x}-\mathbf{y}|}}{4\pi|\mathbf{x}-\mathbf{y}|} \\ \\ \frac{iH_0^{(1)}(k|\mathbf{x}-\mathbf{y}|)}{4} \end{cases}$$

where $\mathrm{supp}(q) \subset \Omega$. Then

$$\sigma(\mathbf{x}) + k^2 q(\mathbf{x}) V[\sigma](\mathbf{x}) = -k^2 q(\mathbf{x})\phi^{\mathrm{inc}}\ ,$$

which is a second-kind integral equation on $L^2(\Omega)$.

The need for speed

Solving

$$\sigma(\mathbf{x}) + k^2 q(\mathbf{x}) V[\sigma](\mathbf{x}) = -k^2 q(\mathbf{x}) \phi^{\mathrm{inc}}$$

- Apply quadrature to discretize the integral $V[\sigma]$. Resolving σ requires at least $O(k^d)$ nodes in $\mathbb{R}^d$.

The need for speed

Solving

$$\sigma(\mathbf{x}) + k^2 q(\mathbf{x}) V[\sigma](\mathbf{x}) = -k^2 q(\mathbf{x}) \phi^{\mathrm{inc}}$$

- Apply quadrature to discretize the integral $V[\sigma]$. Resolving σ requires at least $O(k^d)$ nodes in $\mathbb{R}^d$.
- Solve iteratively (e.g. GMRES or BICGstab)

The need for speed

Solving

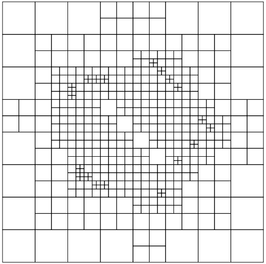
$$\sigma(\mathbf{x}) + k^2 q(\mathbf{x}) V[\sigma](\mathbf{x}) = -k^2 q(\mathbf{x}) \phi^{\mathrm{inc}}$$

- Apply quadrature to discretize the integral $V[\sigma]$. Resolving σ requires at least $O(k^d)$ nodes in $\mathbb{R}^d$.
- Solve iteratively (e.g. GMRES or BICGstab)
- Need a fast method for $V[\sigma]$, which is a dense operator.

Box codes[2]

$$V[\sigma](\mathbf{x}_i) = \int_\Omega G_k(\mathbf{x}_i, \mathbf{y})\sigma(\mathbf{y})\, dv$$

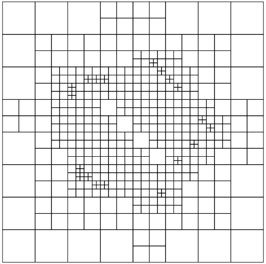
$\Omega = [-L/2, L/2]^d$



[2] Ethridge and Greengard 2001; Cheng, Huang, and Leiterman 2006; Langston, Greengard, and Zorin 2011; Malhotra and Biros 2015.

Box codes[2]

$\Omega = [-L/2, L/2]^d$



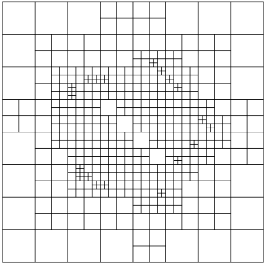
$$V[\sigma](\mathbf{x}_i) = \int_\Omega G_k(\mathbf{x}_i, \mathbf{y})\sigma(\mathbf{y})\, dv \;\approx\; \sum_{j=1}^{N_b} \int_{B_j} G_k(\mathbf{x}_i, \mathbf{y}) p_j(2(\mathbf{y} - \mathbf{y}_j)/L_j)\, dv$$

- N_b boxes, B_j, are leaves of a (balanced) quadtree/octree, which can be adaptively refined to capture small features

[2] Ethridge and Greengard 2001; Cheng, Huang, and Leiterman 2006; Langston, Greengard, and Zorin 2011; Malhotra and Biros 2015.

Box codes[2]

$\Omega = [-L/2, L/2]^d$



$$V[\sigma](\mathbf{x}_i) = \int_\Omega G_k(\mathbf{x}_i, \mathbf{y})\sigma(\mathbf{y})\,dv \;\approx$$

$$\sum_{j=1}^{N_b} \int_{B_j} G_k(\mathbf{x}_i, \mathbf{y}) p_j(2(\mathbf{y} - \mathbf{y}_j)/L_j)\,dv$$

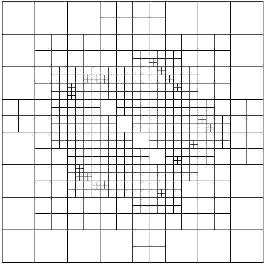
- N_b boxes, B_j, are leaves of a (balanced) quadtree/octree, which can be adaptively refined to capture small features
- $\sigma|_{B_j}(\mathbf{y}) \approx p_j(2(\mathbf{y} - \mathbf{y}_j)/L_j)$ with coefficients in

$$\mathcal{L}_M^{(d)} = \{P_{p_1}(y_1)P_{p_2}(y_2)\cdots P_{p_d}(y_d) \text{ s.t. } p_1{+}p_2{+}\cdots{+}p_d < M\}$$

[2] Ethridge and Greengard 2001; Cheng, Huang, and Leiterman 2006; Langston, Greengard, and Zorin 2011; Malhotra and Biros 2015.

Box codes[2]

$\Omega = [-L/2, L/2]^d$



$$V[\sigma](\mathbf{x}_i) = \int_\Omega G_k(\mathbf{x}_i, \mathbf{y})\sigma(\mathbf{y})\, dv \;\approx$$

$$\sum_{j=1}^{N_b} \int_{B_j} G_k(\mathbf{x}_i, \mathbf{y}) p_j(2(\mathbf{y} - \mathbf{y}_j)/L_j)\, dv$$

- N_b boxes, B_j, are leaves of a (balanced) quadtree/octree, which can be adaptively refined to capture small features
- $\sigma|_{B_j}(\mathbf{y}) \approx p_j(2(\mathbf{y} - \mathbf{y}_j)/L_j)$ with coefficients in

$$\mathcal{L}_M^{(d)} = \{P_{p_1}(y_1)P_{p_2}(y_2)\cdots P_{p_d}(y_d) \text{ s.t. } p_1+p_2+\cdots+p_d < M\}$$

- $\mathbf{x}_i$ are M^d scaled, tensor-product Legendre nodes on each leaf.

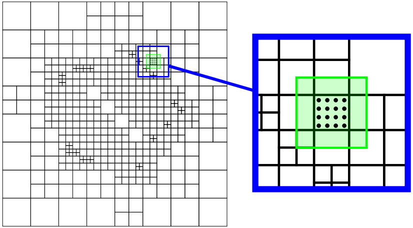
[2] Ethridge and Greengard 2001; Cheng, Huang, and Leiterman 2006; Langston, Greengard, and Zorin 2011; Malhotra and Biros 2015.

$$\sum_{j=1}^{N_b} \int_{B_j} G_k(\mathbf{x}_i, \mathbf{y}) p_j(\mathbf{y})\, dv$$

Naïve evaluation at all targets $\mathbf{x}_i$ costs $O(N_b^2)$.

$$\sum_{j=1}^{N_b} \int_{B_j} G_k(\mathbf{x}_i, \mathbf{y}) p_j(\mathbf{y})\, dv$$

Naïve evaluation at all targets $\mathbf{x}_i$ costs $O(N_b^2)$.



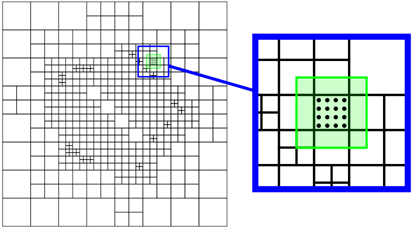
Outside of green box — smooth quadrature sufficient

$$\int_B G_k(\mathbf{x}, \mathbf{y}) \sigma(\mathbf{y})\, dv \approx$$

$$\underbrace{\sum_{l=1}^{N_p} G_k(\mathbf{x}, \mathbf{y}_l(B)) p[\sigma; B](\mathbf{y}_l(B)) w_l(B)}_{\text{“equivalent charges”}}$$

$$\sum_{j=1}^{N_b} \int_{B_j} G_k(\mathbf{x}_i, \mathbf{y}) p_j(\mathbf{y}) \, dv$$

Naïve evaluation at all targets $\mathbf{x}_i$ costs $O(N_b^2)$.



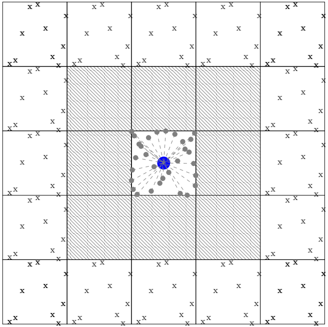
Outside of green box — smooth quadrature sufficient

$$\int_B G_k(\mathbf{x}, \mathbf{y}) \sigma(\mathbf{y}) \, dv \approx \underbrace{\sum_{l=1}^{N_p} G_k(\mathbf{x}, \mathbf{y}_l(B)) p[\sigma; B](\mathbf{y}_l(B)) w_l(B)}_{\text{“equivalent charges”}}$$

The FMM can compute the separated interactions for equivalent charges in $O(N_b \log(1/\epsilon))$.

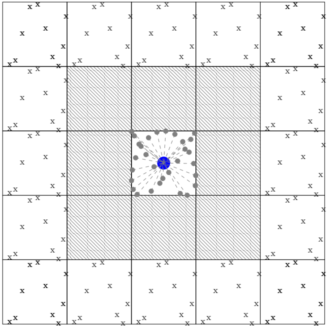
FMM basics (far field)

Multipole expansions for well-separated targets

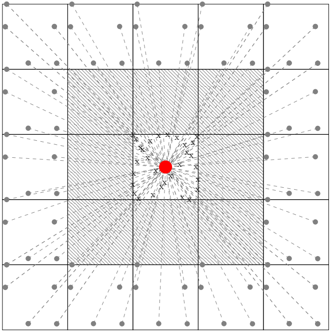


FMM basics (far field)

Multipole expansions for well-separated targets

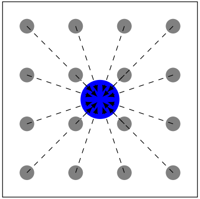


Local expansions for well-separated sources



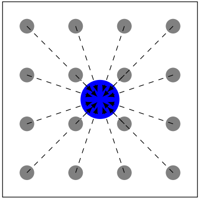
FMM basics (far field)

Form multipoles (leaves)

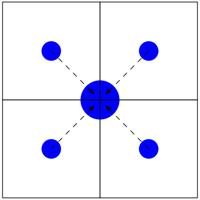


FMM basics (far field)

Form multipoles
(leaves)

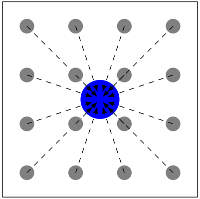


Merge multipoles
(upward pass)

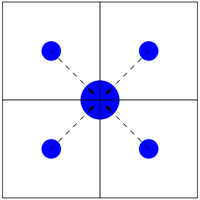


FMM basics (far field)

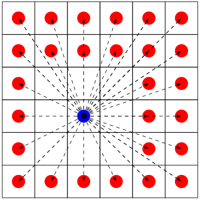
Form multipoles (leaves)



Merge multipoles (upward pass)

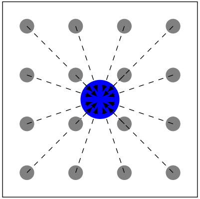


Multipole to local

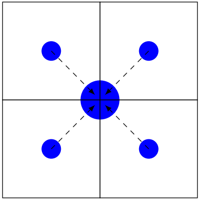


FMM basics (far field)

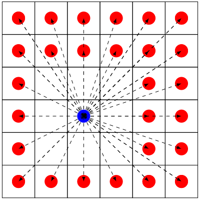
Form multipoles (leaves)



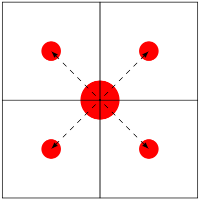
Merge multipoles (upward pass)



Multipole to local

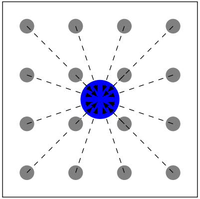


Local to local (downward pass)

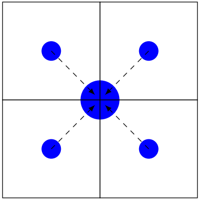


FMM basics (far field)

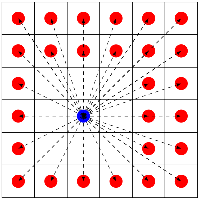
Form multipoles (leaves)



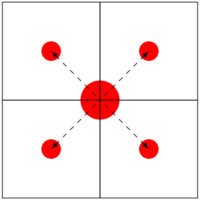
Merge multipoles (upward pass)



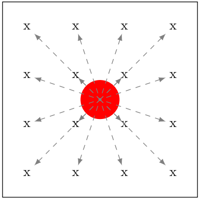
Multipole to local



Local to local (downward pass)



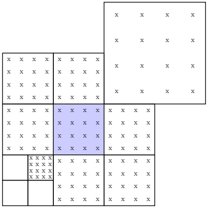
Evaluate local (leaves)



Local work in a box code

$$\int_{B_j} G_k(\mathbf{x},\mathbf{y}) p_j(2(\mathbf{y}-\mathbf{y}_j)/L_j)\, dv$$

These integrals on self and neighbors are weakly singular/ near singular and require special quadrature.

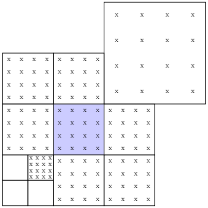


Local work in a box code

$$\int_{B_j} G_k(\mathbf{x},\mathbf{y})p_j(2(\mathbf{y}-\mathbf{y}_j)/L_j)\,dv$$

These integrals on self and neighbors are weakly singular/ near singular and require special quadrature.

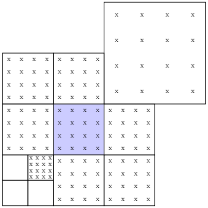
Simplifications



Local work in a box code

$$\int_{B_j} G_k(\mathbf{x},\mathbf{y}) p_j(2(\mathbf{y}-\mathbf{y}_j)/L_j)\, dv$$

These integrals on self and neighbors are weakly singular/ near singular and require special quadrature.



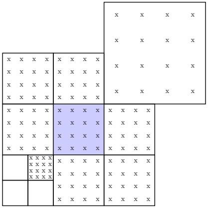
Simplifications

- Linearity: compute for basis and recombine

Local work in a box code

$$\int_{B_j} G_k(\mathbf{x}, \mathbf{y}) p_j(2(\mathbf{y} - \mathbf{y}_j)/L_j)\, dv$$

These integrals on self and neighbors are weakly singular/ near singular and require special quadrature.



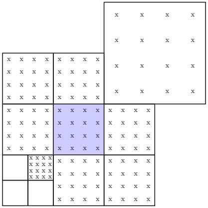
Simplifications

- Linearity: compute for basis and recombine
- Translation invariance and tree balance: relative target positions come from a small(ish), fixed set.

Local work in a box code

$$\int_{B_j} G_k(\mathbf{x}, \mathbf{y}) p_j(2(\mathbf{y} - \mathbf{y}_j)/L_j)\, dv$$

These integrals on self and neighbors are weakly singular/ near singular and require special quadrature.



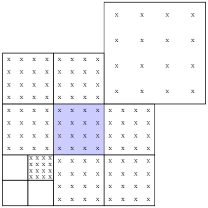
Simplifications

- Linearity: compute for basis and recombine
- Translation invariance and tree balance: relative target positions come from a small(ish), fixed set.
- G_k depends only on $|\mathbf{x} - \mathbf{y}|$ and tensor-product grids have many symmetries

Local work in a box code

$$\int_{B_j} G_k(\mathbf{x}, \mathbf{y}) p_j(2(\mathbf{y} - \mathbf{y}_j)/L_j)\, dv$$

These integrals on self and neighbors are weakly singular/ near singular and require special quadrature.



Simplifications

- Linearity: compute for basis and recombine
- Translation invariance and tree balance: relative target positions come from a small(ish), fixed set.
- G_k depends only on $|\mathbf{x} - \mathbf{y}|$ and tensor-product grids have many symmetries

Plan: precompute all possible interactions, reducing direct interaction calculations to mat-vecs

Box code vs point FMM

In a box code, can precompute and use mat-vecs for work that depends on source/target locations

Box code vs point FMM

In a box code, can precompute and use mat-vecs for work that depends on source/target locations

- direct interactions

Box code vs point FMM

In a box code, can precompute and use mat-vecs for work that depends on source/target locations

- direct interactions
- form multipole

Box code vs point FMM

In a box code, can precompute and use mat-vecs for work that depends on source/target locations

- direct interactions
- form multipole
- evaluate local

Box code vs point FMM

In a box code, can precompute and use mat-vecs for work that depends on source/target locations

- direct interactions
- form multipole
- evaluate local

These must be done for each configuration of points within a standard FMM.

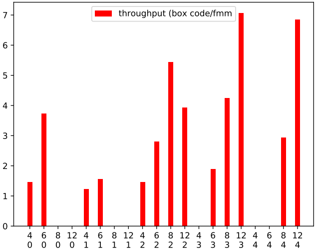
Box code vs point FMM

In a box code, can precompute and use mat-vecs for work that depends on source/target locations

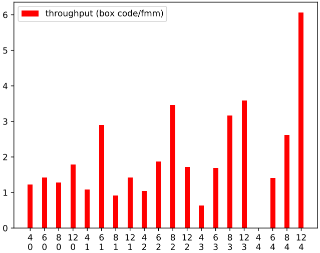
- direct interactions
- form multipole
- evaluate local

These must be done for each configuration of points within a standard FMM.

Less adaptive tree



Highly adaptive tree



Why optimize quadrature generation?

Local interaction tables could conceivably be computed offline once and for all (with some interpolation). Why worry about fast table generation?

Why optimize quadrature generation?

Local interaction tables could conceivably be computed offline once and for all (with some interpolation). Why worry about fast table generation?

- Storage considerations — would need many such tables

Why optimize quadrature generation?

Local interaction tables could conceivably be computed offline once and for all (with some interpolation). Why worry about fast table generation?

- Storage considerations — would need many such tables
  - k ranges over the complex numbers

Why optimize quadrature generation?

Local interaction tables could conceivably be computed offline once and for all (with some interpolation). Why worry about fast table generation?

- Storage considerations — would need many such tables
  - k ranges over the complex numbers
  - there are lots of interactions in 3D

Why optimize quadrature generation?

Local interaction tables could conceivably be computed offline once and for all (with some interpolation). Why worry about fast table generation?

- Storage considerations — would need many such tables
  - k ranges over the complex numbers
  - there are lots of interactions in 3D
  - tables of derivatives (2nd derivatives?)

Why optimize quadrature generation?

Local interaction tables could conceivably be computed offline once and for all (with some interpolation). Why worry about fast table generation?

- Storage considerations — would need many such tables
  - k ranges over the complex numbers
  - there are lots of interactions in 3D
  - tables of derivatives (2nd derivatives?)
- Flexibility — changes to discretization strategy require new tables

Why optimize quadrature generation?

Local interaction tables could conceivably be computed offline once and for all (with some interpolation). Why worry about fast table generation?

- Storage considerations — would need many such tables
  - k ranges over the complex numbers
  - there are lots of interactions in 3D
  - tables of derivatives (2nd derivatives?)
- Flexibility — changes to discretization strategy require new tables
  - Can easily change collocation points (e.g. 3D analogue of Padua points)

Why optimize quadrature generation?

Local interaction tables could conceivably be computed offline once and for all (with some interpolation). Why worry about fast table generation?

- Storage considerations — would need many such tables
  - k ranges over the complex numbers
  - there are lots of interactions in 3D
  - tables of derivatives (2nd derivatives?)
- Flexibility — changes to discretization strategy require new tables
  - Can easily change collocation points (e.g. 3D analogue of Padua points)
  - Elements that aren't quite cubes

Why optimize quadrature generation?

Local interaction tables could conceivably be computed offline once and for all (with some interpolation). Why worry about fast table generation?

- Storage considerations — would need many such tables
  - k ranges over the complex numbers
  - there are lots of interactions in 3D
  - tables of derivatives (2nd derivatives?)
- Flexibility — changes to discretization strategy require new tables
  - Can easily change collocation points (e.g. 3D analogue of Padua points)
  - Elements that aren't quite cubes
- Can still store table on a per-problem basis

Quadrature generation

$$\int_{(-1,1)^d} G_{k'}(\mathbf{x},\mathbf{y}) p_{\mathbf{p}}(\mathbf{y})\, dv\ , \quad p_{\mathbf{p}}(\mathbf{y}) = P_{p_1}(y_1)\cdots P_{p_d}(y_d)$$

[3] Greengard and Lee 1996.
[4] Greengard, O'Neil, et al. 2020

Quadrature generation

$$\int_{(-1,1)^d} G_{k'}(\mathbf{x},\mathbf{y}) p_{\mathbf{p}}(\mathbf{y})\, dv\,, \quad p_{\mathbf{p}}(\mathbf{y}) = P_{p_1}(y_1)\cdots P_{p_d}(y_d)$$

Idea[3]: Green's identity. If $\psi_{\mathbf{p}}$ is an "anti-Helmholtzian", i.e.

$$(\Delta + k'^2)\psi_{\mathbf{p}} = p_{\mathbf{p}}$$

then

$$\int_B G_{k'}(\mathbf{x},\mathbf{y}) p_{\mathbf{p}}(\mathbf{y})\, dv = \chi_B(\mathbf{x})\psi_{\mathbf{p}}(\mathbf{x}) + \underbrace{\int_{\partial B} G_{k'}(\mathbf{x},\mathbf{y})\partial_n \psi_{\mathbf{p}}(\mathbf{y}) - \partial_n G_{k'}(\mathbf{x},\mathbf{y})\psi_{\mathbf{p}}(\mathbf{y})\, da}_{\text{problem reduced to a surface integral}^4}\,.$$

[3] Greengard and Lee 1996.

[4] Greengard, O'Neil, et al. 2020

Anti-Helmholtzians

Goal: compute $\psi_{\mathbf{p}}$ satisfying

$$\max_{\mathbf{x}\in[-1,1]^d} |(\Delta + k'^2)\psi_{\mathbf{p}}(\mathbf{x}) - p_{\mathbf{p}}(\mathbf{x})| < \epsilon$$

Anti-Helmholtzians

Goal: compute $\psi_{\mathbf{p}}$ satisfying

$$\max_{\mathbf{x}\in[-1,1]^d} |(\Delta + k'^2)\psi_{\mathbf{p}}(\mathbf{x}) - p_{\mathbf{p}}(\mathbf{x})| < \epsilon$$

- ϵ near machine precision

Anti-Helmholtzians

Goal: compute $\psi_{\mathbf{p}}$ satisfying

$$\max_{\mathbf{x}\in[-1,1]^d} |(\Delta + k'^2)\psi_{\mathbf{p}}(\mathbf{x}) - p_{\mathbf{p}}(\mathbf{x})| < \epsilon$$

- ϵ near machine precision
- Stable and efficient formula for $\psi_{\mathbf{p}}$

Differentiation and integration on polynomials

Let $\mathcal{P}_M^{(d)} = \{ \text{ polynomials in } \mathbb{R}^d \text{ with total deg. } < M \}$.

Differentiation and integration on polynomials

Let $\mathcal{P}_M^{(d)} = \{$ polynomials in $\mathbb{R}^d$ with total deg. $< M\}$.

$$[\mathcal{D}p](t) = \frac{d}{dt}p(t)$$

Differentiation and integration on polynomials

Let $\mathcal{P}_M^{(d)} = \{$ polynomials in $\mathbb{R}^d$ with total deg. $< M\}$.

$$[\mathcal{D}p](t) = \frac{d}{dt}p(t)$$

- $\mathcal{D} : \mathcal{P}_M^{(1)} \to \mathcal{P}_{M-1}^{(1)}$
- $\|\mathcal{D}|_{\mathcal{P}_M^{(1)}}\|_{L^\infty[-1,1]} \approx M^2$
- $\|\mathcal{D}^l|_{\mathcal{P}_M^{(1)}}\|_{L^\infty[-1,1]} \approx M^{l+1}$, if $l \ll M$.
- $\mathcal{D}^M|_{\mathcal{P}_M^{(1)}} = 0$

Differentiation and integration on polynomials

Let $\mathcal{P}_M^{(d)} = \{$ polynomials in $\mathbb{R}^d$ with total deg. $< M\}$.

$$[\mathcal{D}p](t) = \frac{d}{dt}p(t)$$

- $\mathcal{D} : \mathcal{P}_M^{(1)} \to \mathcal{P}_{M-1}^{(1)}$
- $\|\mathcal{D}|_{\mathcal{P}_M^{(1)}}\|_{L^\infty[-1,1]} \approx M^2$
- $\|\mathcal{D}^l|_{\mathcal{P}_M^{(1)}}\|_{L^\infty[-1,1]} \approx M^{l+1}$,
  if $l \ll M$.
- $\mathcal{D}^M|_{\mathcal{P}_M^{(1)}} = 0$

Roughens, but nilpotent

Differentiation and integration on polynomials

Let $\mathcal{P}_M^{(d)} = \{$ polynomials in $\mathbb{R}^d$ with total deg. $< M\}$.

$$[\mathcal{D}p](t) = \frac{d}{dt}p(t) \qquad\qquad [\mathcal{I}p](t) = \int_{-1}^{t} p(s)\,ds\ .$$

- $\mathcal{D} : \mathcal{P}_M^{(1)} \to \mathcal{P}_{M-1}^{(1)}$
- $\|\mathcal{D}|_{\mathcal{P}_M^{(1)}}\|_{L^\infty[-1,1]} \approx M^2$
- $\|\mathcal{D}^l|_{\mathcal{P}_M^{(1)}}\|_{L^\infty[-1,1]} \approx M^{l+1}$, if $l \ll M$.
- $\mathcal{D}^M|_{\mathcal{P}_M^{(1)}} = 0$

Roughens, but nilpotent

Differentiation and integration on polynomials

Let $\mathcal{P}_M^{(d)} = \{$ polynomials in $\mathbb{R}^d$ with total deg. $< M\}$.

$$[\mathcal{D}p](t) = \frac{d}{dt}p(t) \qquad\qquad [\mathcal{I}p](t) = \int_{-1}^{t} p(s)\, ds\ .$$

- $\mathcal{D} : \mathcal{P}_M^{(1)} \to \mathcal{P}_{M-1}^{(1)}$
- $\|\mathcal{D}|_{\mathcal{P}_M^{(1)}}\|_{L^\infty[-1,1]} \approx M^2$
- $\|\mathcal{D}^l|_{\mathcal{P}_M^{(1)}}\|_{L^\infty[-1,1]} \approx M^{l+1}$, if $l \ll M$.
- $\mathcal{D}^M|_{\mathcal{P}_M^{(1)}} = 0$

- $\mathcal{I} : \mathcal{P}_M^{(1)} \to \mathcal{P}_{M+1}^{(1)}$
- $\|\mathcal{I}|_{\mathcal{P}_M^{(1)}}\|_{L^\infty[-1,1]} \approx 1$
- $\|\mathcal{I}^l|_{\mathcal{P}_M^{(1)}}\|_{L^\infty[-1,1]} \approx 1/l!$

Roughens, but nilpotent

Differentiation and integration on polynomials

Let $\mathcal{P}_M^{(d)} = \{$ polynomials in $\mathbb{R}^d$ with total deg. $< M\}$.

$$[\mathcal{D}p](t) = \frac{d}{dt}p(t)$$

- $\mathcal{D} : \mathcal{P}_M^{(1)} \to \mathcal{P}_{M-1}^{(1)}$
- $\|\mathcal{D}|_{\mathcal{P}_M^{(1)}}\|_{L^\infty[-1,1]} \approx M^2$
- $\|\mathcal{D}^l|_{\mathcal{P}_M^{(1)}}\|_{L^\infty[-1,1]} \approx M^{l+1}$, if $l \ll M$.
- $\mathcal{D}^M|_{\mathcal{P}_M^{(1)}} = 0$

Roughens, but nilpotent

$$[\mathcal{I}p](t) = \int_{-1}^t p(s)\,ds\ .$$

- $\mathcal{I} : \mathcal{P}_M^{(1)} \to \mathcal{P}_{M+1}^{(1)}$
- $\|\mathcal{I}|_{\mathcal{P}_M^{(1)}}\|_{L^\infty[-1,1]} \approx 1$
- $\|\mathcal{I}^l|_{\mathcal{P}_M^{(1)}}\|_{L^\infty[-1,1]} \approx 1/l!$

Smooths, but embiggens the set

Anti-Laplacian[5]

Let $p(\mathbf{y}) = P_{n_1}(y_1)P_{n_2}(y_2)\cdots P_{n_d}(y_d)$. Let $n_1 \geq n_2, \ldots, n_d$ and $m = n_2 + \cdots + n_d$. Set $\tilde{\Delta} = (\partial^2_{y_2} + \cdots + \partial^2_{y_d})$.

Observe

$$\tilde{\Delta} : \mathcal{P}_M^{(d-1)} \to \mathcal{P}_{M-2}^{(d-1)}$$

[5] Greengard and Lee 1996.

Anti-Laplacian[5]

Let $p(\mathbf{y}) = P_{n_1}(y_1)P_{n_2}(y_2)\cdots P_{n_d}(y_d)$. Let $n_1 \geq n_2, \ldots, n_d$ and $m = n_2 + \cdots + n_d$. Set $\tilde{\Delta} = (\partial_{y_2}^2 + \cdots + \partial_{y_d}^2)$.

Observe

$$\tilde{\Delta} : \mathcal{P}_M^{(d-1)} \to \mathcal{P}_{M-2}^{(d-1)}$$

$$\begin{aligned} \psi^{(1)} &= [\mathcal{I}^2 P_{n_1}](y_1)P_{n_2}(y_2)\cdots P_{n_d}(y_d) \\ \Delta\psi^{(1)} &= p + [\mathcal{I}^2 P_{n_1}](y_1)\tilde{\Delta}(P_{n_2}(y_2)\cdots P_{n_d}(y_d)) \end{aligned}$$

[5]Greengard and Lee 1996.

Anti-Laplacian[5]

Let $p(\mathbf{y}) = P_{n_1}(y_1)P_{n_2}(y_2)\cdots P_{n_d}(y_d)$. Let $n_1 \geq n_2, \ldots, n_d$ and $m = n_2 + \cdots + n_d$. Set $\tilde{\Delta} = (\partial^2_{y_2} + \cdots + \partial^2_{y_d})$.

Observe

$$\tilde{\Delta} : \mathcal{P}^{(d-1)}_{M} \to \mathcal{P}^{(d-1)}_{M-2}$$

$$\begin{aligned}
\psi^{(1)} &= [\mathcal{I}^2 P_{n_1}](y_1)P_{n_2}(y_2)\cdots P_{n_d}(y_d) \\
\Delta\psi^{(1)} &= p + [\mathcal{I}^2 P_{n_1}](y_1)\tilde{\Delta}(P_{n_2}(y_2)\cdots P_{n_d}(y_d)) \\
\psi^{(2)} &= [\mathcal{I}^2 P_{n_1}](y_1)P_{n_2}(y_2)\cdots P_{n_d}(y_d) - [\mathcal{I}^4 P_{n_1}](y_1)\tilde{\Delta}(P_{n_2}(y_2)\cdots P_{n_d}(y_d)) \\
\Delta\psi^{(2)} &= p - [\mathcal{I}^4 P_{n_1}](y_1)\tilde{\Delta}^2(P_{n_2}(y_2)\cdots P_{n_d}(y_d))
\end{aligned}$$

[5] Greengard and Lee 1996.

Anti-Laplacian[5]

Let $p(\mathbf{y}) = P_{n_1}(y_1)P_{n_2}(y_2)\cdots P_{n_d}(y_d)$. Let $n_1 \geq n_2, \dots, n_d$ and $m = n_2 + \cdots + n_d$. Set $\tilde{\Delta} = (\partial_{y_2}^2 + \cdots + \partial_{y_d}^2)$.

Observe

$$\tilde{\Delta} : \mathcal{P}_M^{(d-1)} \to \mathcal{P}_{M-2}^{(d-1)}$$

$$\begin{aligned}
\psi^{(1)} &= [\mathcal{I}^2 P_{n_1}](y_1)P_{n_2}(y_2)\cdots P_{n_d}(y_d) \\
\Delta\psi^{(1)} &= p + [\mathcal{I}^2 P_{n_1}](y_1)\tilde{\Delta}(P_{n_2}(y_2)\cdots P_{n_d}(y_d)) \\
\psi^{(2)} &= [\mathcal{I}^2 P_{n_1}](y_1)P_{n_2}(y_2)\cdots P_{n_d}(y_d) - [\mathcal{I}^4 P_{n_1}](y_1)\tilde{\Delta}(P_{n_2}(y_2)\cdots P_{n_d}(y_d)) \\
\Delta\psi^{(2)} &= p - [\mathcal{I}^4 P_{n_1}](y_1)\tilde{\Delta}^2(P_{n_2}(y_2)\cdots P_{n_d}(y_d)) \\
&\vdots \\
\Delta^{-1}p &:= \sum_{j=0}^{\lfloor m/2 \rfloor} (-1)^j [\mathcal{I}^{2j+2} P_{n_1}](y_1)\tilde{\Delta}^{2j}(P_{n_2}(y_2)\cdots P_{n_d}(y_d)) \in \mathcal{P}_{M+2}^{(d)}
\end{aligned}$$

[5] Greengard and Lee 1996.

Neumann series

Is $\Delta + k'^2|_{\mathcal{P}_M^{(d)}}$ a perturbation of Δ? Let $p \in \mathcal{P}_M^{(d)}$.

Neumann series

Is $\Delta + k'^2|_{\mathcal{P}_M^{(d)}}$ a perturbation of Δ? Let $p \in \mathcal{P}_M^{(d)}$.

$$\begin{aligned}(\Delta + k'^2)^{-1}p &= \Delta^{-1}(1 + k'^2\Delta^{-1})^{-1}p \\ &= \Delta^{-1}\sum_{j=0}^{\infty}(-1)^j k'^{2j}\Delta^{-j}p\end{aligned}$$

Neumann series

Is $\Delta + k'^2|_{\mathcal{P}_M^{(d)}}$ a perturbation of Δ? Let $p \in \mathcal{P}_M^{(d)}$.

$$\begin{aligned}(\Delta + k'^2)^{-1}p &= \Delta^{-1}(1 + k'^2\Delta^{-1})^{-1}p \\ &= \Delta^{-1}\sum_{j=0}^{\infty}(-1)^j k'^{2j}\Delta^{-j}p\end{aligned}$$

Neumann series

Is $\Delta + k'^2|_{\mathcal{P}_M^{(d)}}$ a perturbation of Δ? Let $p \in \mathcal{P}_M^{(d)}$.

$$\begin{aligned}(\Delta + k'^2)^{-1} p &= \Delta^{-1}(1 + k'^2 \Delta^{-1})^{-1} p \\ &= \Delta^{-1} \sum_{j=0}^{\infty} (-1)^j k'^{2j} \Delta^{-j} p\end{aligned}$$

Sum converges in $L^\infty[-1,1]$ for any p. Formula only good when $|k'|$ small.

Neumann series

Is $\Delta + k'^2|_{\mathcal{P}_M^{(d)}}$ a perturbation of k'^2? Let $p \in \mathcal{P}_M^{(d)}$.

Neumann series

Is $\Delta + k'^2|_{\mathcal{P}_M^{(d)}}$ a perturbation of k'^2? Let $p \in \mathcal{P}_M^{(d)}$.

$$\begin{aligned}(\Delta + k'^2)^{-1} p &= \frac{1}{k'^2} (\Delta / k'^2 + 1)^{-1} p \\ &= \frac{1}{k'^2} \sum_{j=0}^{\infty} (-1)^j \frac{\Delta^j p}{k'^{2j}}\end{aligned}$$

Neumann series

Is $\Delta + k'^2|_{\mathcal{P}_M^{(d)}}$ a perturbation of k'^2? Let $p \in \mathcal{P}_M^{(d)}$.

$$\begin{aligned}
(\Delta + k'^2)^{-1} p &= \frac{1}{k'^2} (\Delta / k'^2 + 1)^{-1} p \\
&= \frac{1}{k'^2} \sum_{j=0}^{\infty} (-1)^j \frac{\Delta^j p}{k'^{2j}} \\
&= \frac{1}{k'^2} \sum_{j=0}^{\lfloor M/2 \rfloor} (-1)^j \frac{\Delta^j p}{k'^{2j}}
\end{aligned}$$

Neumann series

Is $\Delta + k'^2|_{\mathcal{P}_M^{(d)}}$ a perturbation of k'^2? Let $p \in \mathcal{P}_M^{(d)}$.

$$\begin{aligned}
(\Delta + k'^2)^{-1} p &= \frac{1}{k'^2} (\Delta / k'^2 + 1)^{-1} p \\
&= \frac{1}{k'^2} \sum_{j=0}^{\infty} (-1)^j \frac{\Delta^j p}{k'^{2j}} \\
&= \frac{1}{k'^2} \sum_{j=0}^{\lfloor M/2 \rfloor} (-1)^j \frac{\Delta^j p}{k'^{2j}}
\end{aligned}$$

Formula only good when $|k'|$ large.

Have two anti-Helmholtzians:

$$\psi^{(1)} = \frac{1}{k'^2} \sum_{j=0}^{\lfloor M/2 \rfloor} (-1)^j \frac{\Delta^j p}{k'^{2j}} \,, \quad \psi^{(2)} = \Delta^{-1} \sum_{j=0}^{\infty} (-1)^j k'^{2j} \Delta^{-j} p$$

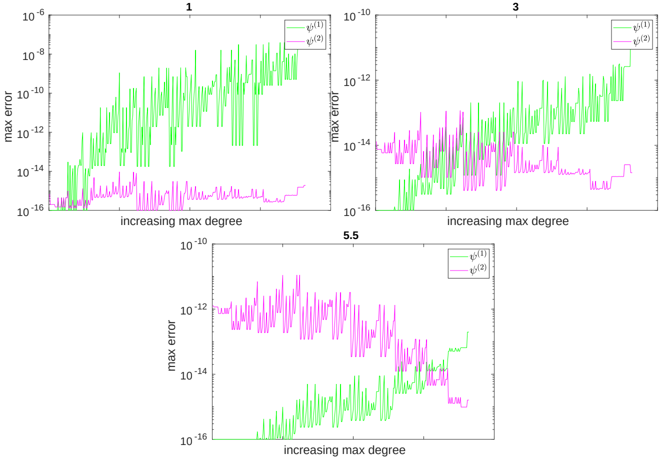
Are they good enough for all values $|k'|$?

Stability

- Test k' with $|k'| = 1, 3, 5.5$.
- Set cut-off for sum for $\psi^{(2)}$ very high.
- Plot error $\max|(\Delta + k'^2)\psi - p|$ (double precision)

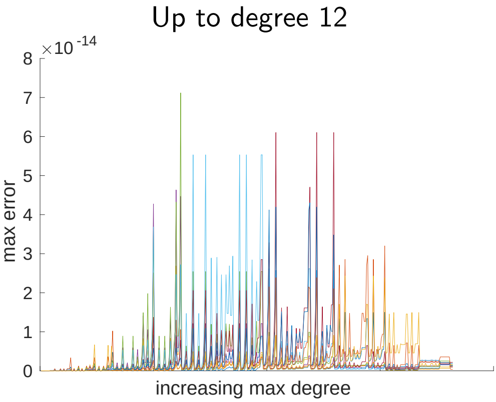
Stability

- Test k' with $|k'| = 1, 3, 5.5$.
- Set cut-off for sum for $\psi^{(2)}$ very high.
- Plot error $\max|(\Delta + k'^2)\psi - p|$ (double precision)



Stability

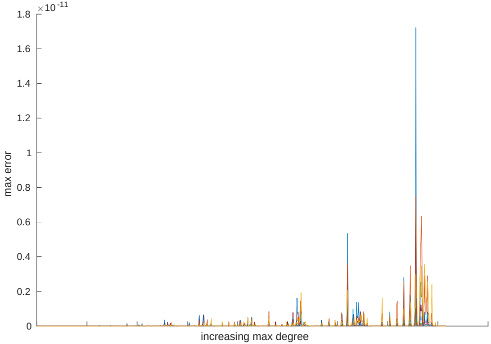
- Test k' with $|k'| = 1, 1.5, \ldots, 5.5$.
- Set cut-off for sum for $\psi^{(2)}$ very high.
- Plot best error $\max |(\Delta + k'^2)\psi - p|$ using either $\psi^{(1)}$ or $\psi^{(2)}$ (double precision)



Stability

- Test k' with $|k'| = 1, 1.5, \ldots, 5.5$.
- Set cut-off for sum for $\psi^{(2)}$ very high.
- Plot best error $\max |(\Delta + k'^2)\psi - p|$ using either $\psi^{(1)}$ or $\psi^{(2)}$ (double precision)

Up to degree 16



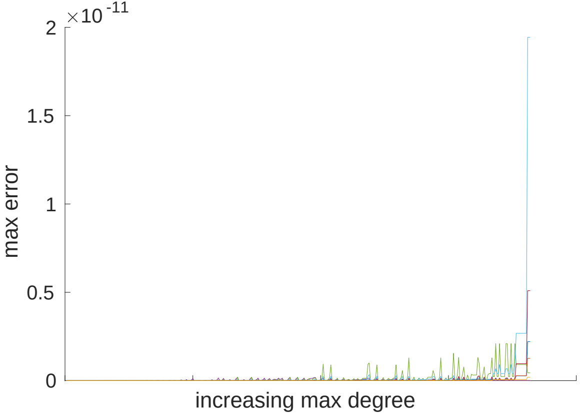
Efficiency

- Test k' with $|k'| = 1, 1.5, \ldots, 5.5$.
- Set cut-off for sum for $\psi^{(2)}$ at 16 terms.
- Plot best error using either $\psi^{(1)}$ or $\psi^{(2)}$ in double precision

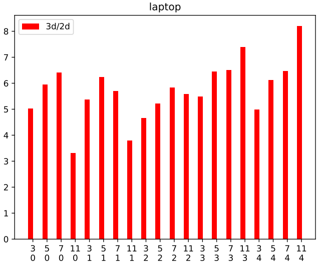
Efficiency

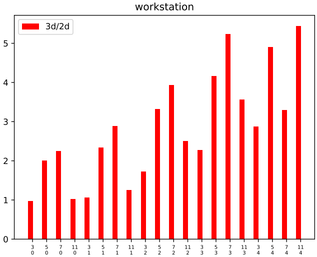
- Test k' with $|k'| = 1, 1.5, \ldots, 5.5$.
- Set cut-off for sum for $\psi^{(2)}$ at 16 terms.
- Plot best error using either $\psi^{(1)}$ or $\psi^{(2)}$ in double precision

Up to degree 12

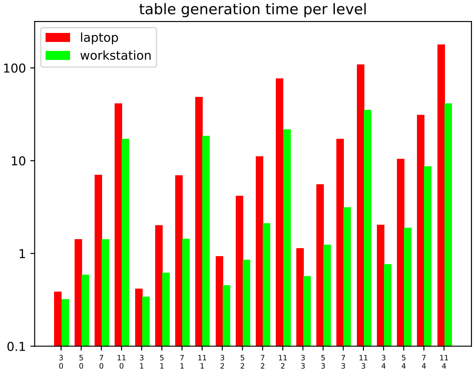


Compare to 3D adaptive integration





What does this do for us?



Future work

- Iteration count appears to be $O(k^2)$ for solving

$$\sigma + k^2 q V[\sigma] = -k^2 q \phi^{\mathrm{inc}}$$

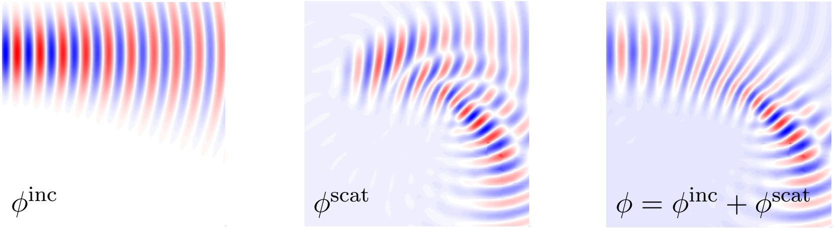
  Overall that's $O(k^5)$. Yikes! Experiment with preconditioning/domain decomposition strategies.

Future work

- Iteration count appears to be $O(k^2)$ for solving

$$\sigma + k^2 qV[\sigma] = -k^2 q\phi^{\mathrm{inc}}$$

  Overall that's $O(k^5)$. Yikes! Experiment with preconditioning/domain decomposition strategies.
- A posteriori adaptive refinement



Thank you.

References

- Hongwei Cheng, Jingfang Huang, and Terry Jo Leiterman. "An adaptive fast solver for the modified Helmholtz equation in two dimensions". In: *Journal of Computational Physics* 211.2 (2006), pp. 616–637.
- Aaron J Danner and Ulf Leonhardt. "Lossless design of an Eaton lens and invisible sphere by transformation optics with no bandwidth limitation". In: *International Quantum Electronics Conference*. Optical Society of America. 2009, JThC4.
- Frank Ethridge and Leslie Greengard. "A new fast-multipole accelerated Poisson solver in two dimensions". In: *SIAM Journal on Scientific Computing* 23.3 (2001), pp. 741–760.
- Leslie Greengard and June-Yub Lee. "A direct adaptive Poisson solver of arbitrary order accuracy". In: *Journal of Computational Physics* 125.2 (1996), pp. 415–424.

Leslie Greengard, Michael O'Neil, et al. "Fast multipole methods for evaluation of layer potentials with locally-corrected quadratures". In: *arXiv preprint arXiv:2006.02545* (2020).

Harper Langston, Leslie Greengard, and Denis Zorin. "A free-space adaptive FMM-based PDE solver in three dimensions". In: *Communications in Applied Mathematics and Computational Science* 6.1 (2011), pp. 79–122.

Dhairya Malhotra and George Biros. "PVFMM: A parallel kernel independent FMM for particle and volume potentials". In: *Communications in Computational Physics* 18.3 (2015), pp. 808–830.

Felipe Vico, Leslie Greengard, and Miguel Ferrando. "Fast convolution with free-space Green's functions". In: *Journal of Computational Physics* 323 (2016), pp. 191–203.