Automatic Synthesis of Low Complexity Translation Operators for the Fast Multipole Method

Isuru Fernando, Andreas Klöckner

March 1, 2021

Outline

- Quick introduction to Taylor series based Fast Multipole Method
- Compressed Taylor Series based expansions and translations
- Results - accuracy and time complexity

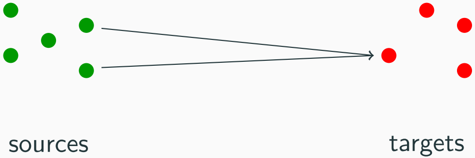
N-body problem

Let $(\mathbf{s}_j)_{j=1}^n$ be sources and $(\mathbf{t}_i)_{i=1}^n$ be targets. Potential at target $\mathbf{t}_i$ is the sum of all potentials from the sources $\mathbf{s}_j$ given by,

$$\sum_j \psi(\mathbf{t}_i, \mathbf{s}_j).$$

For example,

$$\psi(\mathbf{t}_i, \mathbf{s}_j) = \frac{1}{\mathrm{dist}(\mathbf{t}_i, \mathbf{s}_j)}.$$



n sources and n targets $\implies \mathcal{O}(n^2)$ cost.

Fast Multipole Method

Algorithm by Greengard and Rokhlin (1987) to compute the potentials in $\mathcal{O}(n)$ time.

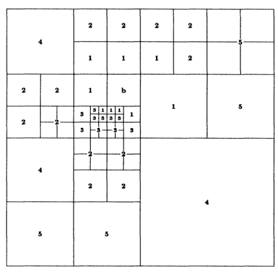


Figure 1: Carrier et al, 1988

Useful for solving PDEs with Integral equation methods.

$$\int G(x-y)\sigma_y dy.$$

Taylor Series based FMM

Local expansion:

$$\psi(\mathbf{t},\mathbf{s}) = \sum_{|m|\le k} \underbrace{\frac{D_{\mathbf{t}}^m \psi(\mathbf{t},\mathbf{s})\Big|_{\mathbf{t}=\mathbf{c}}}{m!}}_{\text{depends on src/ctr}} \underbrace{(\mathbf{t}-\mathbf{c})^m}_{\text{depends on tgt/ctr}}$$

Multipole expansion:

$$\psi(\mathbf{t},\mathbf{s}) = \sum_{|m|\le k} \underbrace{\frac{D_{\mathbf{s}}^m \psi(\mathbf{t},\mathbf{s})\Big|_{\mathbf{s}=\mathbf{c}}}{m!}}_{\text{depends on tgt/ctr}} \underbrace{(\mathbf{s}-\mathbf{c})^m}_{\text{depends on src/ctr}}$$

Taylor Series based FMM

Expansion Types:

- Special purpose expansions (Spherical harmonics, Fourier-bessel based)
- Linear Algebra (Eg: Kernel-independent FMM)
- Taylor series based expansions

Pros	Cons
- Easily tractable symbolically for any kernel	- Expansions $\mathrm{O}(p^3)$ compared to $\mathrm{O}(p^2)$
	- Translations $\mathrm{O}(p^6)$ compared to $\mathrm{O}(p^2 \log(p))$
	- Stability issues

Table 1: Pros and cons of Taylor series based expansions

Compressed Multipole Expansion

When ψ satisfies the Helmholtz equation,

$$\psi_{xx} + \psi_{yy} + \kappa^2 \psi = 0.$$

Recall

$$\psi(\mathbf{t}, \mathbf{s}) = \sum_{|m| \le p} \underbrace{\frac{D_{\mathbf{s}}^m \psi(\mathbf{t}, \mathbf{s})\Big|_{\mathbf{s}=\mathbf{c}}}{m!}}_{\text{depends on tgt/ctr}} \underbrace{(\mathbf{s} - \mathbf{c})^m}_{\text{depends on src/ctr}}$$

From the PDE we have

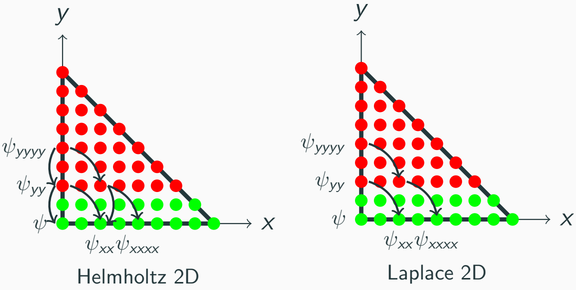
$$\begin{aligned} c_1\psi_{xx} + c_2\psi_{yy} + c_3\psi &= c_1\psi_{xx} + c_2(-\psi_{xx} - \kappa^2\psi) + c_3\psi \\ &= (c_1 - c_2)\psi_{xx} + 0\psi_{yy} + \psi(c_3 - \kappa^2 c_2). \end{aligned}$$

Compressed Multipole Expansion

For Helmholtz equation we also have

$$\psi_{xxyy} + \psi_{yyyy} + \kappa^2 \psi_{yy} = 0,$$

$$\psi_{xxxx} + \psi_{xxyy} + \kappa^2 \psi_{xx} = 0.$$



All the coefficients represented by red dots get zeroed.

Count of expansion coefficients go from $\mathcal{O}(p^d)$ to $\mathcal{O}(p^{d-1})$.

Compressed Local Expansion

Recall

$$\psi(\mathbf{t},\mathbf{s}) = \sum_{|m|\le p} \underbrace{\frac{D_{\mathbf{t}}^{m}\psi(\mathbf{t},\mathbf{s})\Big|_{\mathbf{t}=\mathbf{c}}}{m!}}_{\text{depends on src/ctr}} \underbrace{(\mathbf{t}-\mathbf{c})^{m}}_{\text{depends on tgt/ctr}}$$

Out of $\mathcal{O}(p^d)$ coefficients, only $\mathcal{O}(p^{d-1})$ are independent.

This makes the number of terms of a local expansion to be $\mathcal{O}(p^{d-1})$.

Calculating derivatives for Local Expansion

Tausch (2003) proposes an algorithm which has an amortized $\mathcal{O}(p)$ time.

We found several formulae to calculate these in amortized $\mathcal{O}(1)$ time.

For Laplace 3D

$$
\begin{aligned}
r^2 \frac{\partial^{n+m+l}}{\partial x^n y^m z^l}\left(\frac{1}{r}\right) = &-(2n-1)x\frac{\partial^{n+m-1}}{\partial x^{n-1} y^m z^l}\left(\frac{1}{r}\right) - (n-1)^2\frac{\partial^{n+m-2}}{\partial x^{n-2} y^m z^l}\left(\frac{1}{r}\right) - 2my\frac{\partial^{n+m-1}}{\partial x^n y^{m-1} z^l}\left(\frac{1}{r}\right) \\
&- m(m-1)\frac{\partial^{n+m-2}}{\partial x^n y^{m-2} z^l}\left(\frac{1}{r}\right) - 2lz\frac{\partial^{n+m-1}}{\partial x^n y^m z^{l-1}}\left(\frac{1}{r}\right) - l(l-1)\frac{\partial^{n+m-2}}{\partial x^n y^m z^{l-2}}\left(\frac{1}{r}\right)
\end{aligned}
$$

.

For Biharmonic 2D,

$$
\begin{aligned}
r^2 \frac{\partial^{n+m}}{\partial x^n y^m}\left(r^2\log(r)\right) = &-2(n-2)x\frac{\partial^{n+m-1}}{\partial x^{n-1} y^m}\left(r^2\log(r)\right) - (n-1)(n-4)\frac{\partial^{n+m-2}}{\partial x^{n-2} y^m}\left(r^2\log(r)\right) \\
&- 2my\frac{\partial^{n+m-1}}{\partial x^n y^{m-1}}\left(r^2\log(r)\right) - m(m-1)\frac{\partial^{n+m-2}}{\partial x^n y^{m-2}}\left(r^2\log(r)\right).
\end{aligned}
$$

This reduces the cost of P2L from $\mathcal{O}(p^d)$ to $\mathcal{O}(p^{d-1})$.

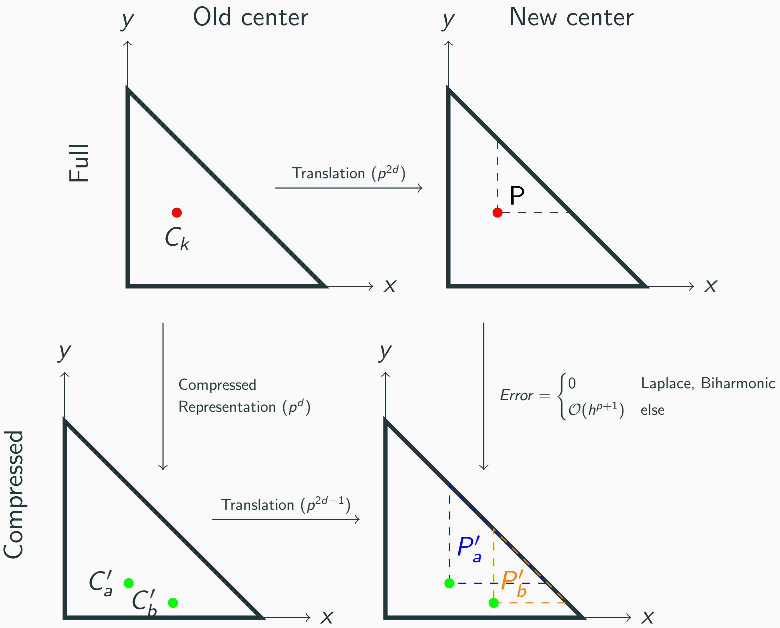
Naive Multipole Translation

Let c_1 be the old center and c be the new center. Then,

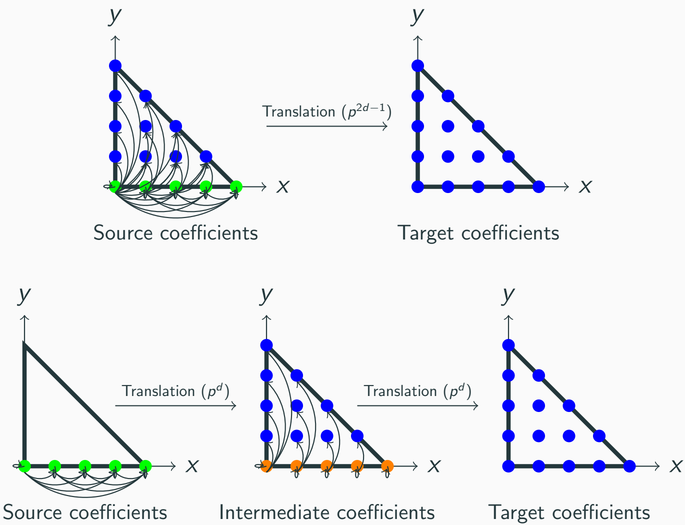
$$\begin{aligned}
(\mathbf{s}-\mathbf{c})^k &= \left((\mathbf{s}-\mathbf{c}_1)+(\mathbf{c}_1-\mathbf{c})\right)^k \\
&= \sum_{l\leq k} \binom{k}{l} (\mathbf{s}-\mathbf{c}_1)^l (\mathbf{c}_1-\mathbf{c})^{k-l} \\
&= \sum_{l\leq k} \beta_{k,l} (\mathbf{s}-\mathbf{c}_1)^l
\end{aligned}$$

Cost: $\mathcal{O}(p^{2d})$.

Compressed Multipole Translation

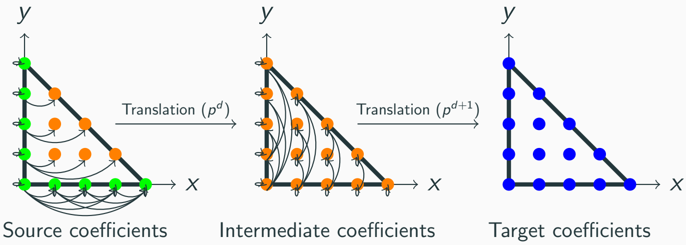


Faster Compressed Multipole Translation

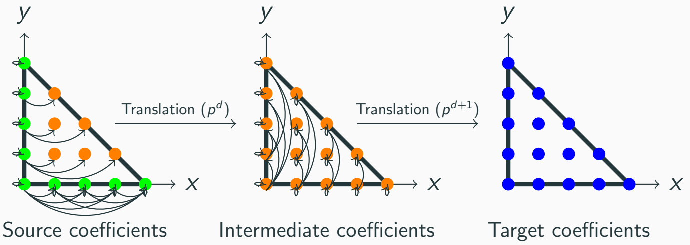


Note: For local to local translation, reverse all arrows.

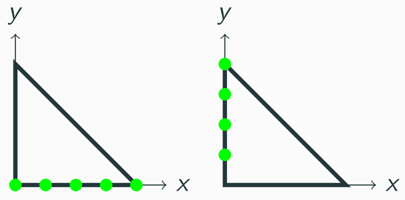
Faster Compressed Multipole Translation



Faster Compressed Multipole Translation



Divide the problem into 2 subproblems



Compressed Multipole to Local Translation

From multipole expansion, we get,

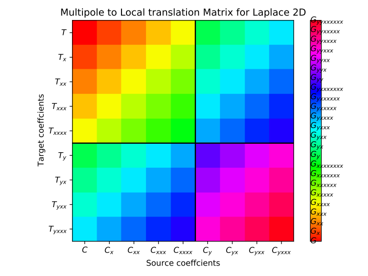
$$\psi(\mathbf{t},\mathbf{s}) = \sum_{|m|\le k} \underbrace{\frac{D_{\mathbf{s}}^{m}\psi(\mathbf{t},\mathbf{s})\Big|_{\mathbf{s}=\mathbf{c}}}{m!}}_{\text{depends on tgt/ctr}} \underbrace{(\mathbf{s}-\mathbf{c})^{m}}_{\text{depends on src/ctr}}$$

To translate this multipole expansion to a local expansion, we need to get the derivatives of the above expression and evalaute at new center.

Cost: $\mathcal{O}(p^{2d-2})$.

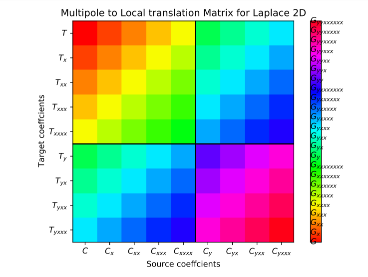
Compressed Multipole to Local Translation

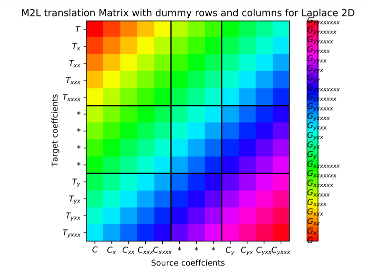
Multipole to local translation matrix is a block Toeplitz matrix of smaller Toeplitz matrices.



Compressed Multipole to Local Translation

Multipole to local translation matrix is a block Toeplitz matrix of smaller Toeplitz matrices.





Use an FFT to do the translation similar to Greengard (1988).

Cost depends on number of dummy rows:

- $\mathcal{O}(p^{d-1}\log(p))$ for elliptic PDEs
- $\mathcal{O}(p^{d}\log(p))$ for other PDEs

Time complexities

	P2L/M2P	P2M/L2P	M2M	M2L	L2L
Taylor Series	$\mathbf{p}^3$	$\mathbf{p}^3$	$\mathbf{p}^6$	$\mathbf{p}^6$	$\mathbf{p}^6$
Improved Taylor Series	$\mathbf{p}^3$	$\mathbf{p}^3$	$\mathbf{p}^4$	$\mathbf{p}^3 \log(\mathbf{p})$	$\mathbf{p}^6$
Compressed Taylor Series without fast derivatives	$\mathbf{p}^3$	$\mathbf{p}^3$	$\mathbf{p}^3$	$\mathbf{p}^2 \log(\mathbf{p})$	$\mathbf{p}^3$
Compressed Taylor Series with fast derivatives	$\mathbf{p}^2$	$\mathbf{p}^3$	$\mathbf{p}^3$	$\mathbf{p}^2 \log(\mathbf{p})$	$\mathbf{p}^3$
Spherical Harmonic Series	$\mathbf{p}^2$	$\mathbf{p}^2$	$\mathbf{p}^2 \log(\mathbf{p})$	$\mathbf{p}^2 \log(\mathbf{p})$	$\mathbf{p}^2 \log(\mathbf{p})$

Table 2: Time complexities for expansions, translations and evaluations

All operations are exact except for M2M in Compressed Taylor and M2L operations with FFT.

Code generation

With Compressed Taylor generating code for Stokes

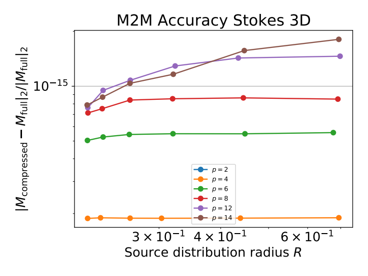
$$\mu\nabla^2\mathbf{u} - \boldsymbol{\nabla} p + \mathbf{f} = \mathbf{0}$$
$$\boldsymbol{\nabla}\cdot\mathbf{u} = 0$$

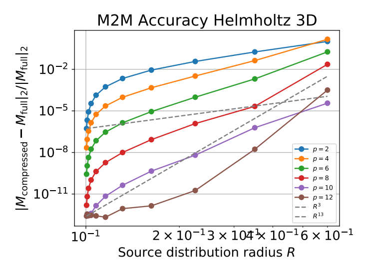
is done simply by giving the PDE as,

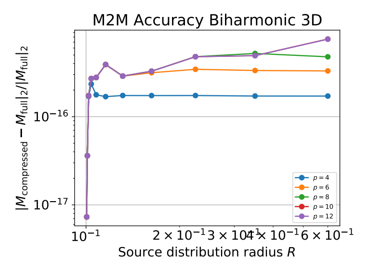
```
w = make_pde_syms(dim, dim+1)
mu = sym.Symbol("mu")
u = w[:dim]
p = w[-1]
pdes = PDE(mu * laplacian(u) - grad(p), div(u))
```

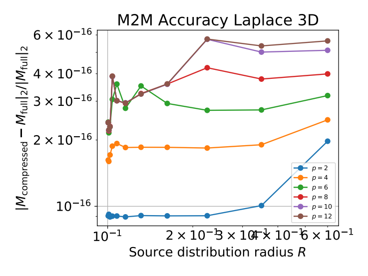
which generates code for the expansion, translations and evaluations.

Results - Error M2M

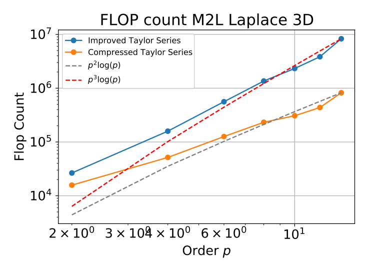


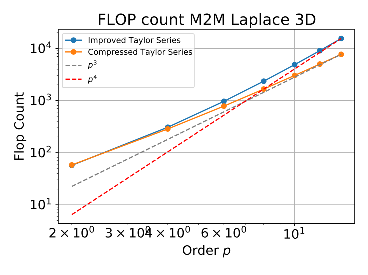


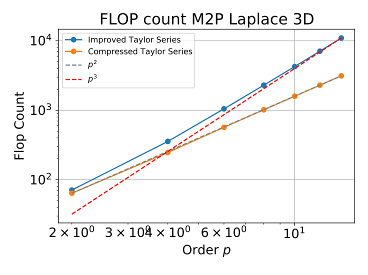


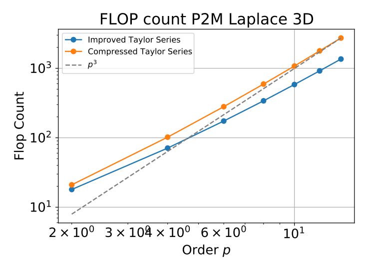


Results - FLOP count









Summary

- Kernel generic method for elliptic constant coefficient linear PDEs.
- Only needs the PDE and the Green's function for the PDE.
- Asymptotically better than full Taylor Series in
  - Number of FLOPs
  - Storage
- Next goal: A fast Stokes solver on a GPU.

Ackowledgements:

- NSF grants 19-11019 and 16-54756
- SIAM travel grant