

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

---

**Isuru Fernando**, Andreas Klöckner

March 2, 2021

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

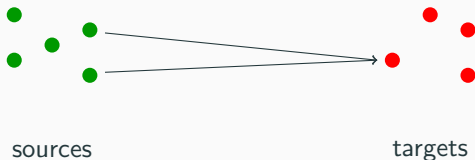
# N-body problem

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,

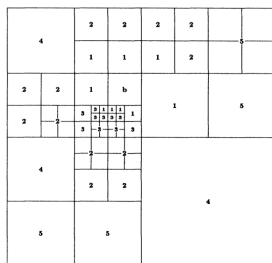
$$\psi(\mathbf{t}_i, \mathbf{s}_j) = \frac{1}{\text{dist}(\mathbf{t}_i, \mathbf{s}_j)}.$$



$n$  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 PDE BVPs with Integral equation methods via layer potentials.

$$\int_{\Gamma} G(x-y) \sigma_y dy.$$

# Taylor Series based FMM

Local expansion:

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

Multipole expansion:

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

# Taylor Series based FMM

Some Expansion Types:

- Expansions using separation of angular variables (Spherical harmonics, Fourier-bessel based)
- Linear Algebra (Eg: Kernel-independent FMM)
- Taylor series based (Cartesian) expansions

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

**Table 1:** Pros and cons of Taylor series based expansions

# Taylor Series based FMM

Goals:

- Find a way to reduce Taylor series cost.
- Use the Taylor series to automate FMM for "any" kernel

# Compressed Multipole Expansion

When  $\psi$  satisfies the Helmholtz equation,

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

Recall

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

From the PDE we have

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

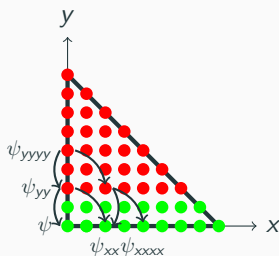


# Compressed Multipole Expansion

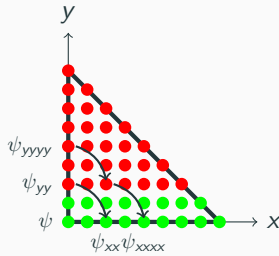
For Helmholtz equation we also have

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

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



Helmholtz 2D



Laplace 2D

All the coefficients represented by red dots get lumped into "green" coefficients.

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

# Compressed Local Expansion

Recall

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

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  $\mathcal{O}(p^{d-1})$ .

# Calculating derivatives for Local Expansion

Tausch (2003) proposes an algorithm which has an amortized  $\mathcal{O}(p)$  time assuming that the Green's function is radially symmetric.

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,

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

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

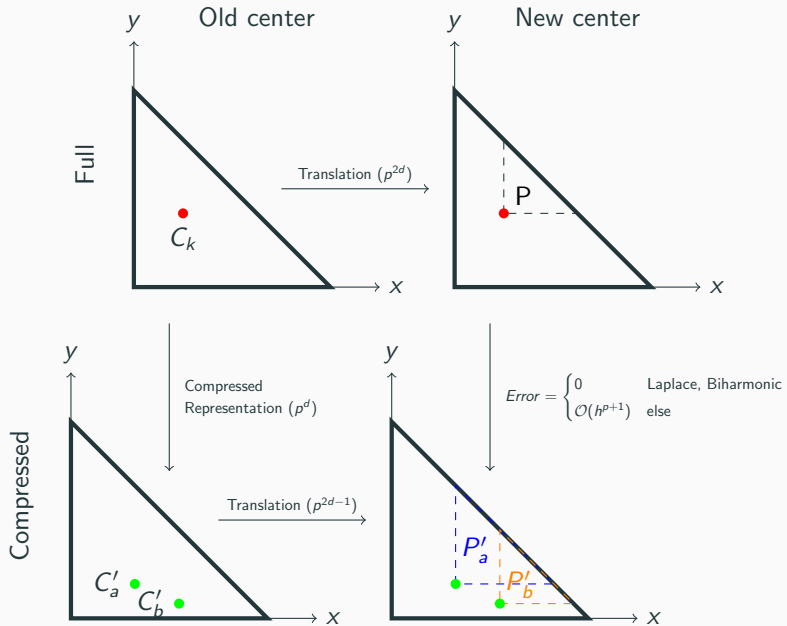
# Naive Multipole Translation

Let  $\mathbf{c}_1$  be the old center and  $\mathbf{c}$  be the new center. Then,

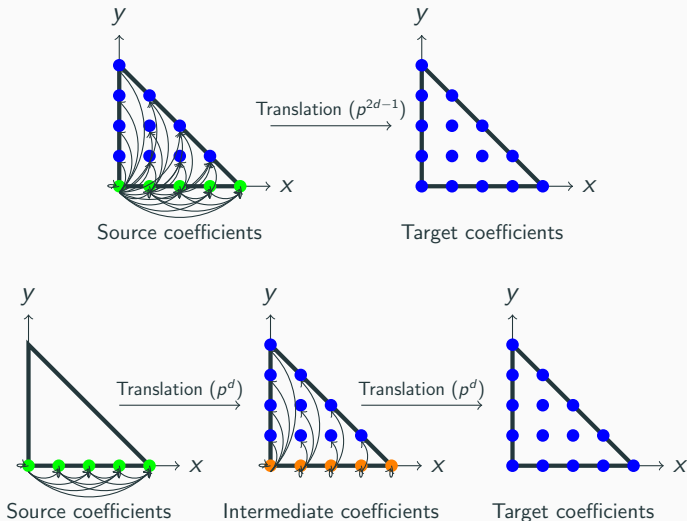
$$\begin{aligned}(\mathbf{s} - \mathbf{c})^k &= ((\mathbf{s} - \mathbf{c}_1) + (\mathbf{c}_1 - \mathbf{c}))^k \\&= \sum_{l \leq k} \binom{k}{l} (\mathbf{s} - \mathbf{c}_1)^l (\mathbf{c}_1 - \mathbf{c})^{k-l} \\&= \sum_{l \leq k} \beta_{k,l} (\mathbf{s} - \mathbf{c}_1)^l\end{aligned}$$

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

# Compressed Multipole Translation

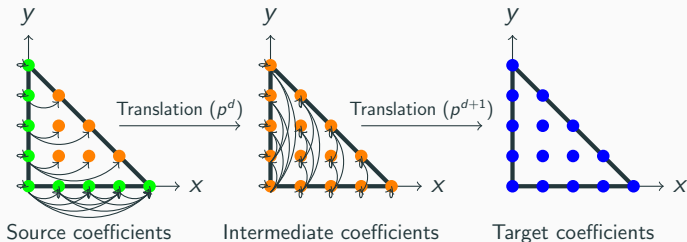


# Faster Compressed Multipole Translation

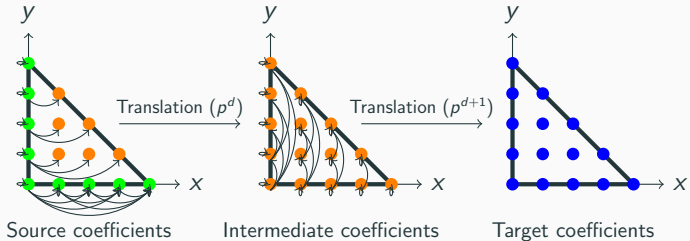


Note: For local to local translation, reverse all arrows.

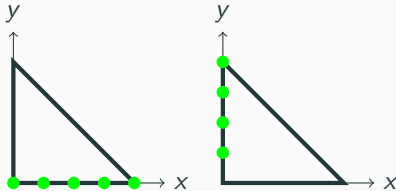
# Faster Compressed Multipole Translation



# Faster Compressed Multipole Translation



Divide the problem into 2 subproblems to keep the cost down to  $\mathcal{O}(p^d)$ .





# Compressed Multipole to Local Translation

From multipole expansion, we get,

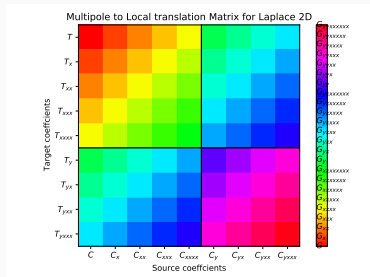
$$\psi(\mathbf{t}, \mathbf{s}) = \sum_{|\mathbf{m}| \leq k} \underbrace{\frac{D_{\mathbf{s}}^{\mathbf{m}} \psi(\mathbf{t}, \mathbf{s})|_{\mathbf{s}=\mathbf{c}}}{m!}}_{\text{depends on tgt/ctr}} \underbrace{(\mathbf{s} - \mathbf{c})^{\mathbf{m}}}_{\text{depends on src/ctr}}$$

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

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

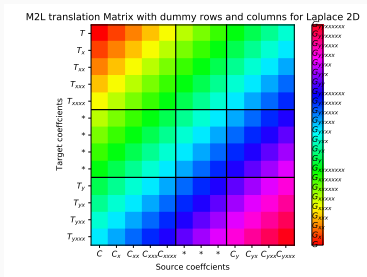
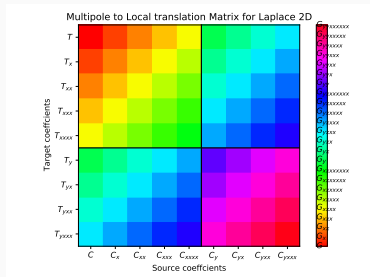
# Compressed Multipole to Local Translation

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	$p^3$	$p^3$	$p^6$	$p^6$	$p^6$
Improved Taylor Series	$p^3$	$p^3$	$p^4$	$p^3 \log(p)$	$p^4$
Compressed Taylor Series without fast derivatives	$p^3$	$p^3$	$p^3$	$p^2 \log(p)$	$p^3$
<b>Compressed Taylor Series with fast derivatives</b>	$p^2$	$p^3$	$p^3$	$p^2 \log(p)$	$p^3$
Spherical Harmonic Series	$p^2$	$p^2$	$p^2 \log(p)$	$p^2 \log(p)$	$p^2 \log(p)$

**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

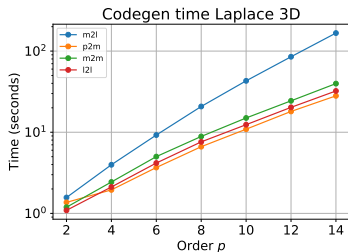
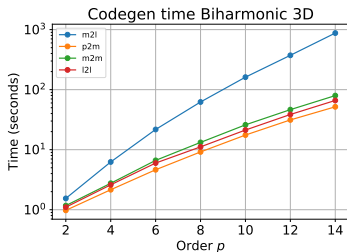
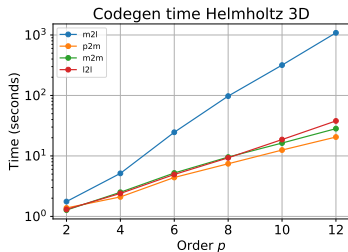
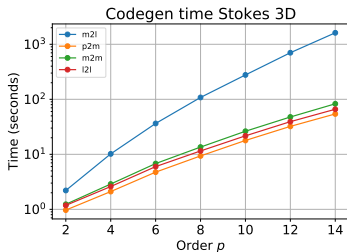
$$\begin{aligned}\mu \nabla^2 \mathbf{u} - \nabla p + \mathbf{f} &= \mathbf{0} \\ \nabla \cdot \mathbf{u} &= 0\end{aligned}$$

is done simply by giving the PDE as:

```
w = make_pde_syms(ndims=3, neqs=4)
mu = sym.Symbol("mu")
u = w[:3]
p = w[-1]
pdes = PDE(mu * laplacian(u) - grad(p), div(u))
```

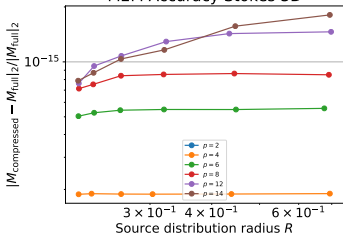
which generates code for the expansion, translations and evaluations.

# Code generation

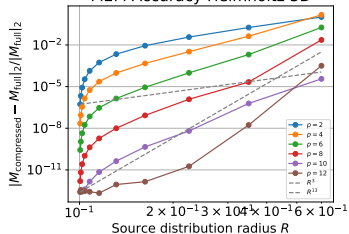


# Results - Error M2M

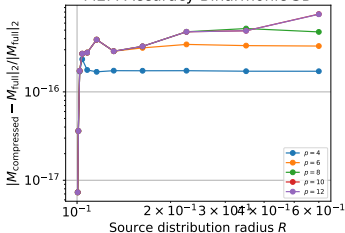
M2M Accuracy Stokes 3D



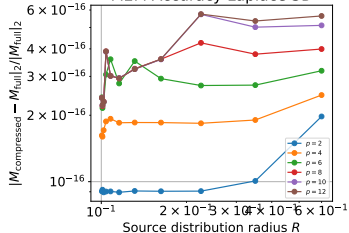
M2M Accuracy Helmholtz 3D



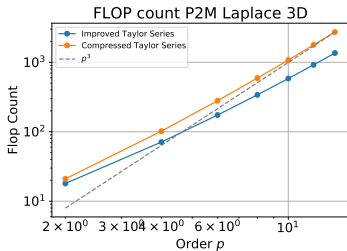
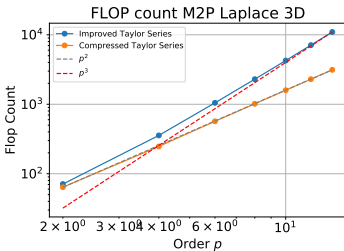
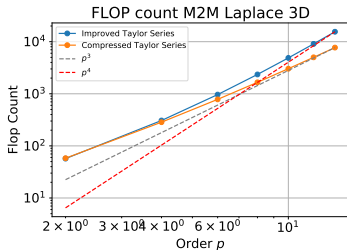
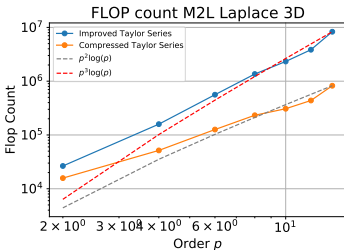
M2M Accuracy Biharmonic 3D



M2M Accuracy Laplace 3D



# Results - FLOP count





# Summary

- Kernel-generic method for elliptic constant coefficient linear PDEs with non-oscillatory kernels (for now).
- 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.

## Acknowledgements:

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