Accelerating potential evaluation over unstructured meshes in two dimensions

Zewen Shen, Kirill Serkh

University of Toronto

July, 2022

Introduction

Definition

The volume potential u is defined as

$$u(x) = \iint_{\Omega} K(x,y) f(y) \, \mathrm{d}y,$$

where K is a Green's function, and f is a given density function.

Why is it interesting?

- Convert an inhomogeneous second order linear elliptic PDE to a homogeneous one.
- Then, the boundary integral equation-based algorithms can handle the homogeneous PDEs nicely.
- Volume integral equations.

Difficulties

$$u(x) = \iint_{\Omega} K(x,y) f(y)\,\mathrm{d}y$$

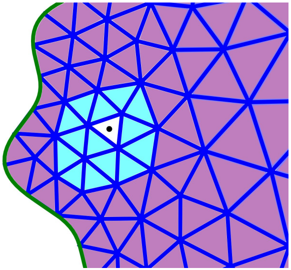
1. The Green's function K has a singularity.
2. Linear time complexity w.r.t. the number of quadrature nodes and targets.
3. The integration domain Ω can be complicated.

Singularity of the integrand

$$u(x) = \iint_\Omega K(x,y)f(y)\,\mathrm{d}y$$

Given a target location $x \in \mathbb{R}^2$, divide the integration domain Ω into three parts: $\Omega_{far}, \Omega_{near}, \Omega_{self}$, such that

$$u(x) = \iint_{\Omega_{far}} K(x,y)f(y)\,\mathrm{d}y + \iint_{\Omega_{near}} K(x,y)f(y)\,\mathrm{d}y + \iint_{\Omega_{self}} K(x,y)f(y)\,\mathrm{d}y$$



Singularity of the integrand

$$u(x) = \iint_{\Omega_{far}} K(x,y)f(y)\,\mathrm{d}y + \iint_{\Omega_{near}} K(x,y)f(y)\,\mathrm{d}y + \iint_{\Omega_{self}} K(x,y)f(y)\,\mathrm{d}y$$

- **Over Ω_{far} (far field interactions)**: smooth integrand.
  Solution: a standard quadrature rule is enough.
  Referred to as "*far field quadrature rule*".
- **Over Ω_{near} (near field interactions)**: nearly-singular integrand.
  Solution: adaptively subdivide the domain.
- **Over Ω_{self} (self-interactions)**: singular integrand.
  Solution: specialized quadrature rules [Bremer & Gimbutas (2012)].

The details of the evaluation algorithms are off-topic.

Takeaway: these three integrals can be computed accurately.

$\mathcal{O}(n)$ volume potential evaluation algorithm

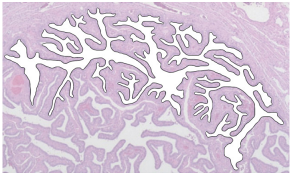
Given a list of target locations $\{x_i\}_{i=1,\ldots,n}$ in the domain Ω,

- the far field interactions can be computed in $\mathcal{O}(n)$ operations for all x_i by the FMM;
- for each x_i, the associated near and self-interaction computational costs is $\mathcal{O}(1)$, since the number of mesh elements inside its near field is a constant.

Takeaway: the volume potential can be evaluated in $\mathcal{O}(n)$ operations without any pain.

Complicated integration domain Ω

Suppose that the domain Ω is complicated, e.g., [Guo et. al. (2020)]



A natural idea: triangulate it.
Problem: mesh elements around the boundary are curved.

- Cannot treat these curved elements as triangles.
- Only have access to quadrature rules for standard shapes.
- Can construct a well-conditioned mapping from a triangle to an arbitrary curved element (so-called blending function).

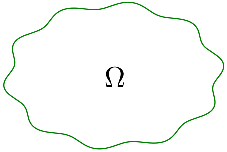
$$\Longrightarrow$$

Change of variables.

Experimental results

$$u(x) = \iint_\Omega \log(\|x-y\|) f(y)\,\mathrm{d}y$$

- The density f is the sum of two Gaussians.
- The error tolerance is 10^{-14}.
- Run on a laptop, without any parallelization.



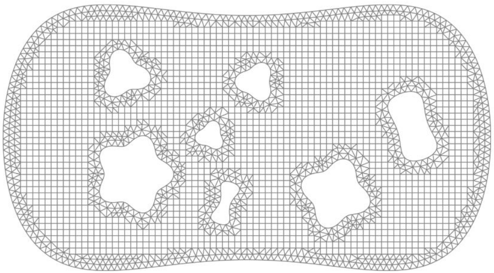
N_{tgt}	T_{far}	T_{near}	E_{abs}
$52K$	0.63	9.07	2.42×10^{-12}
$230K$	3.15	33.1	4.13×10^{-13}
$964K$	13.3	123	4.54×10^{-13}

Can we accelerate the near interaction computations?

Special case: when the domain Ω is regular... No problem!

- rectangular mesh + translation invariance of the Green's function = "box code"! (super fast near and self-interaction computations.)

What about the general case?



[Anderson & Zhu & Veerapaneni (2022)]

Can we accelerate the near interaction computations?

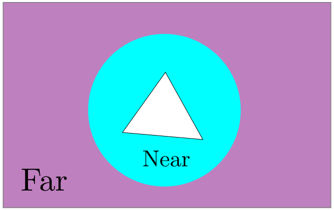
- Even though the unstructured mesh only makes up a small proportion of the whole mesh, the near interaction computations over them are still the bottleneck of the whole algorithm.
- Furthermore, the performance of “embedded box code” deteriorates as the ratio of boundary length to domain area increases.
- Additionally, when one computes surface potentials, the box code is not applicable at all, since the use of unstructured meshes is in general necessary.
- Therefore, the potential evaluation over unstructured meshes is a problem of great importance.

Rethinking the time cost

- The cost of far field interaction computations depends on the total number of sources and targets, and the error tolerance.
- For each target, the cost of near field interaction computations depends on the number of adaptive subdivisions that has to be done over the mesh elements that are "near" the target.

The conventional definition of "near"

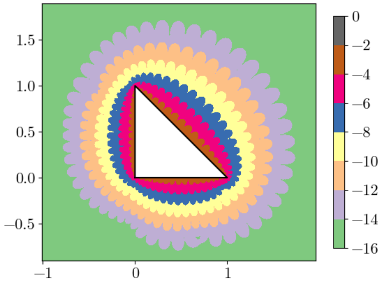
Given a mesh element Δ, a target location $x \in \mathbb{R}^2 \setminus \Delta$ is near Δ if it's in the interior of a circle centered at Δ with radius being the radius of Δ times some factor.



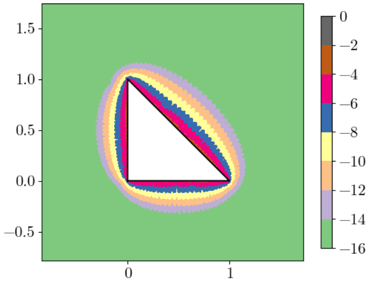
Rethinking the time cost

The real definition of "near"

Given a mesh element Δ, a target location $x \in \mathbb{R}^2 \setminus \Delta$ is near Δ if the volume potential $u(x)$ generated over Δ cannot be approximated to the desired error tolerance by the far field quadrature rule.



(a) Order 20



(b) Order 40

Rethinking the time cost

Smaller near field

$$\Longrightarrow$$

Fewer near field interaction computations

In the low accuracy regime, the near field is small even for far field quadrature rules of moderate order.

In the high accuracy regime, can we just use an extremely high order far field quadrature rule to minimize the number of near interaction computations?

Caveat I: increased far field interaction computational cost

The total number of quadrature nodes increases as we increase the order of the far field quadrature rule.
More specifically, $2\times$order $\Rightarrow$ $4\times$quadrature nodes.

The far field interaction computations take $\mathcal{O}(N_{\text{quad}} + N_{\text{interp}})$ operations.
Thus, $2\times$order $\Rightarrow \mathcal{O}(4N_{\text{quad}} + N_{\text{interp}})$ operations.

The far field interaction computational cost is proportional to the size of the near field. Roughly speaking,
$2\times$order $\Rightarrow \frac{1}{2}\times$size of the near field $\Rightarrow \frac{1}{2}\times$near field interactions.

The near field interaction computations are offloaded onto the far field interaction computations. But is it scalable?

Caveat I: increased far field interaction computational cost

Fact I: The FMM cost is often dominated by the number of interpolation nodes (targets), rather than the number of quadrature nodes (sources).

- The length of a 20th order interpolation scheme is 231.
- The length of a 20th order quadrature rule is 78.
- The far field interaction computational cost only increases by 69% when the quadrature order increases from 20 to 40.

Fact II: there exist highly-optimized parallel FMM libraries.

- These libraries are designed to have high arithmetic intensity and parallel efficiency.
- While the near field interaction computations are embarrassingly parallelizable, it involves many branch instructions. Furthermore, it requires expertise to write high performance parallel programs.

Caveat II: practical estimation of the near field

Now it doesn't seem to be a bad idea to use a very high order far field quadrature rule.

But the offloading technique is impractical, unless we know precisely the geometry of the near field.

Caveat II: practical estimation of the near field

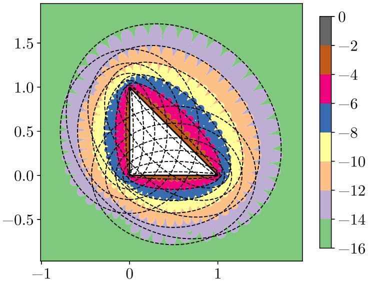
Common strategy for a precise estimation of the near field:
(practical) quadrature error estimate.

Related work:

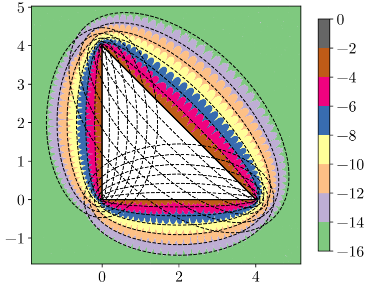
- **1-D Layer potential**: [Barnett (2014)], [af Klinteberg & Tornberg (2017)].
- **Surface potential** (off-surface eval.): [af Klinteberg & Sorgentone & Tornberg (2022)].
- **Surface potential** (on-surface eval.): open question.
- **Volume potential**: we made some progress.

Preview of our estimation

For each error tolerance (e.g., $10^{-14}, 10^{-12}, \dots$) , we place one Bernstein ellipse located at each edge of the triangle. The union of the three ellipses is our estimated error contour.



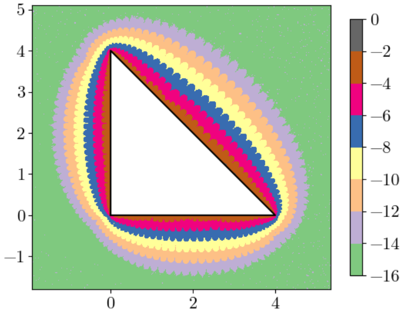
(a) Order 20



(b) Order 40

Derivation of the estimation

Motivation: Bernstein ellipses!



Derivation of the estimation

Setup: Let $x_0 \in \mathbb{R}^2 \setminus \Delta$ be arbitrary. Assume that x_0 is in the far field of Δ, i.e.,

$$\left|\sum_{i=1}^{M} w_i \log\|x_0 - y_i\| f(y_i) - \iint_\Delta \log\|x_0 - y\| f(y)\,\mathrm{d}A_y\right| < \varepsilon,$$

where $\{(y_i, w_i)\}_{i=1,\ldots,M}$ denotes a quadrature rule of order N.
We first expand the integrand into Koornwinder polynomials

$$\log\|x_0 - y\| f(y) = \sum_{n=0}^{\infty}\sum_{m=0}^{n} a_{mn} K_{mn}(y).$$

The formula above becomes

$$\left|\sum_{i=1}^{M} w_i \sum_{n=0}^{\infty}\sum_{m=0}^{n} a_{mn} K_{mn}(y_i) - \iint_\Delta \sum_{n=0}^{\infty}\sum_{m=0}^{n} a_{mn} K_{mn}(y)\,\mathrm{d}A_y\right| < \varepsilon.$$

Derivation of the estimation

$$\left|\sum_{i=1}^{M} w_i \sum_{n=0}^{\infty}\sum_{m=0}^{n} a_{mn}K_{mn}(y_i) - \iint_{\Delta} \sum_{n=0}^{\infty}\sum_{m=0}^{n} a_{mn}K_{mn}(y)\,\mathrm{d}A_y\right| < \varepsilon$$

Since the quadrature rule integrates all polynomials up to order N exactly, and $\iint_{\Delta} K_{mn}(y)\,\mathrm{d}A_y = 0$ for $m+n>0$, we have

$$\left|\sum_{i=1}^{M} w_i \cdot R_N(y_i)\right| < \varepsilon,$$

where

$$R_N(y) := \sum_{n=0}^{\infty}\sum_{m=0}^{n} a_{mn}K_{mn}(y) - \sum_{n=0}^{N}\sum_{m=0}^{N-n} a_{mn}K_{mn}(y).$$

Intuitively, it's saying that if we apply the quadrature rule to the "remainder" of the integrand, we should get a tiny number.

Derivation of the estimation

$$\left|\sum_{i=1}^{M} w_i \cdot R_N(y_i)\right| < \varepsilon$$

As R_N is a general function, it's generally true that the above inequality holds iff for all i,

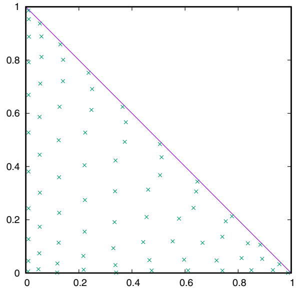
$$|R_N(y_i)| < \frac{\varepsilon}{w_i}.$$

This criterion is very discrete...Can we make it continuous?

Derivation of the estimation

$$|R_N(y_i)| < \frac{\varepsilon}{w_i}$$

Let's take a look at the layout of the quadrature nodes $\{y_i\}$ when the order of the quadrature $N = 20$. Generally speaking, $\sim N/2$ nodes cluster around the each edge of the triangle



We first consider the edge at the bottom.

Derivation of the estimation

$$|R_N(y_i)| < \frac{\varepsilon}{w_i}$$

Denote the $\sim N/2$ nodes next to the edge at the bottom by $\{\widetilde{y}_i\}$. Define w to be the largest quadrature weight corresponding to these nodes. Then,

$$|R_N(\widetilde{y}_i)| < \frac{\varepsilon}{w}$$

is trivially satisfied.

By continuity,

$$\left\| R_N|_{[0,1]\times\{0\}} \right\|_{L^\infty} < C_N \cdot \frac{\varepsilon}{w},$$

where C_N is a constant that accounts for the usage of continuity.

The appearance of C_N seems to be sketchy...

Derivation of the estimate

$$\left\| R_N|_{[0,1]\times\{0\}} \right\|_{L^\infty} < C_N \cdot \frac{\varepsilon}{w}$$

It's equivalent to say that the restriction of the integrand $\log \|x_0 - y\| f(y)$ onto the bottom edge can be approximated by a Nth order 1-D polynomial up to an error of $C_N \cdot \frac{\varepsilon}{w}$.

The 1-D approximation theory tells us that the integrand must be analytic inside a Bernstein ellipse with a certain parameter to make this hold. Therefore, x_0 must be outside of that Bernstein ellipse, if x_0 is in the "far field" of the triangle.

We then apply this argument to the remaining two edges of the triangle to get two more Bernstein ellipses. The near field must contain the union of these three Bernstein ellipses.

Derivation of the estimate

A lower bound is not enough. Roughly speaking, if we know the integrand is both analytic and **bounded** inside a Bernstein ellipse, we can bound its approximation error.

Given x_0 outside the union of three Bernstein ellipses, the integrand $\log \|x_0 - y\| f(y)$ is analytic, but not bounded inside the ellipses.

But the singularity of the logarithmic kernel is very weak.

Create 3 Bernstein ellipses that are $\sim \epsilon$ larger than the original ones

$$\Longrightarrow$$

require x_0 to be outside of these three new ellipses

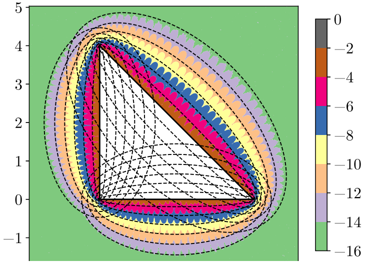
$$\Longrightarrow$$

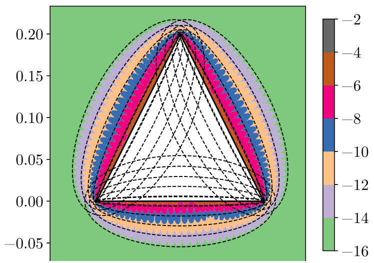
the integrand is bounded and analytic inside the original ellipses

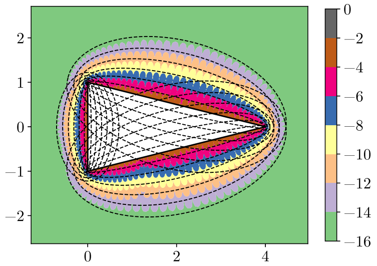
$$\Longrightarrow$$

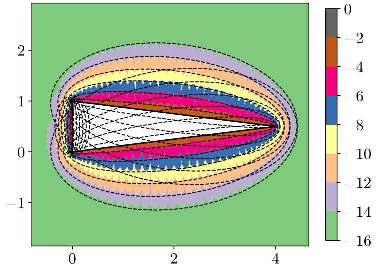
an upper bound of the near field is obtained.

Demo: $N = 40$, various sizes and aspect ratios

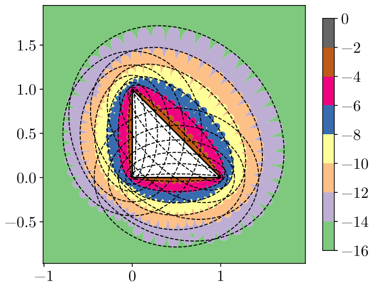


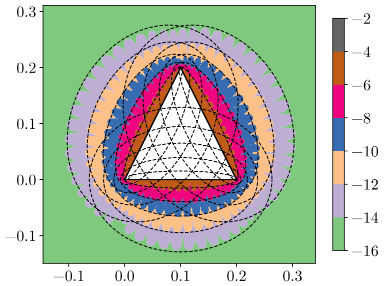






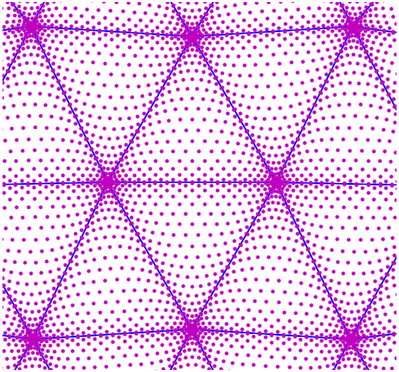
Demo: $N = 20$, various sizes and aspect ratios





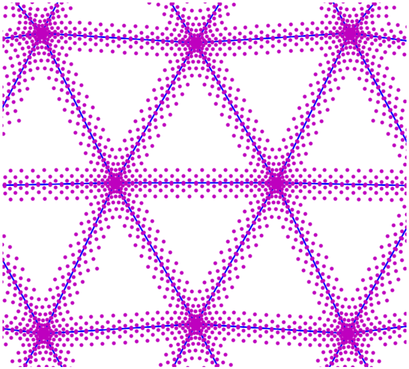
As long as the density function f can be well-resolved by the quadrature rule, our estimation works well for triangles of various sizes and aspect ratios, and for quadrature rules of various orders.

Demo: reduction in near interaction computations



Originally, almost all of the target points require several near interaction computations.

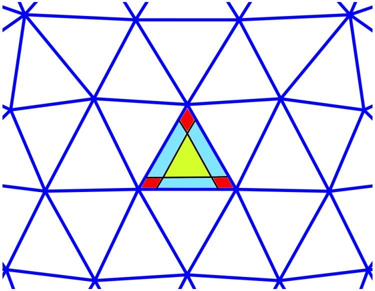
Demo: reduction in near interaction computations



With the error tolerance set to be 10^{-10} and the use of a 50-th order far field quadrature rule, the targets located near the centroid of each triangle require no near field interaction computations.

Non-uniform near interaction computational costs

Let's think about the near interaction computational costs one more time...



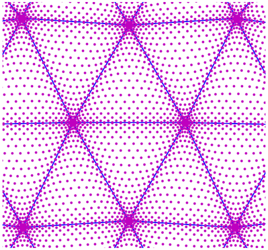
- Green: Free (w/ a high order far field quadrature rule);
- Blue: Subdivision over ~ 1 neighboring triangle;
- Red: Subdivisions over ~ 5 neighboring triangles.

Conclusion: the cost depends highly on the relative position of the target.

Why is the green region more privileged?

Reducing near interaction computations via meshes

In the setting of volume potential interpolation, the target nodes cluster near the vertices of the mesh.



We can create a quadrature mesh, such that the regions with high density of targets are located near "green regions" in that mesh.

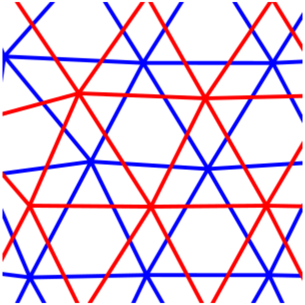
$$\Longrightarrow$$

The quadrature mesh should be staggered to the interpolation mesh.

$$\Longrightarrow$$

The majority of targets are free of near interaction computations.

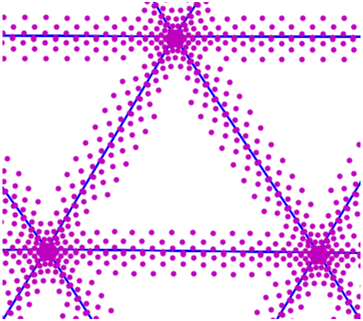
An example of the staggered meshes

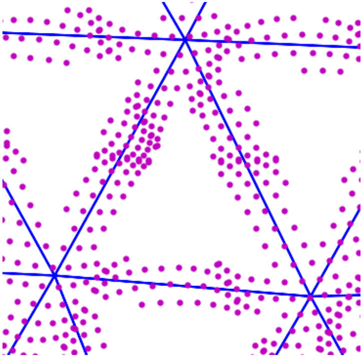


- One can formulate the problem of constructing a staggered mesh as a maximal independent set problem.
- In practice, even a mesh that's moderately different from the original mesh can substantially reduce the near interaction computations.

Demo: effect of staggered mesh

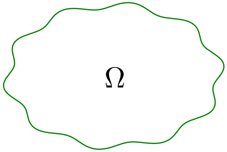
Left: without a staggered mesh; Right: after the use of a staggered mesh.





Numerical experiment (fine-tuned baseline)

We evaluate the logarithmic volume potential at all the discretization nodes over Ω, with the density being the sum of two Gaussian functions, and the error tolerance being 10^{-10}.



h_0	$T_{\mathcal{F}}^0$	$T_{\mathcal{F}}^*$	$T_{\mathcal{N}}^0$	$T_{\mathcal{N}}^*$	$\frac{T_{\mathcal{N}}^*}{T_{\mathcal{N}}^0}$
0.2	0.55	0.84	3.59	1.15	31.9%
0.1	2.75	4.10	13.3	3.83	28.8%
0.05	11.7	17.3	48.8	12.3	25.2%

With no parallelization and an imperfect staggered mesh:
1.5x far interaction costs, 0.25x near interaction costs.

With a highly optimized parallel FMM library and a nice staggered mesh...

Generalization and conclusion

One could create multiple staggered meshes to make even more targets free of near interactions.

The ideas presented in this talk are kernel-independent, and can be applied to the surface and the 3-D volume cases.

One of the few applications of very high order quadrature rules in high dimensions...

Takeaway: The near field interactions over an unstructured mesh are costly to evaluate. In theory, however, most of them can be transformed into far field interactions. All you need is

1. a reasonably high order quadrature rule;
2. an educated selection of the mesh.

Acknowledgement

We sincerely thank James Bremer for his helpful advice and for our informative conversations.

Registration and travel support for this presentation was provided by the Society for Industrial and Applied Mathematics.