HPC Final Project

May 25, 2019

1 Algorithm description

Periodic Stokes potentials Given the Stokes equation, the Green's function for the velocity field for the free-space problem is given by the Oseen-Burgers tensor:

$$\mathbf{S}(\mathbf{x}) = \frac{\mathbf{1}}{|\mathbf{x}|} + \frac{\mathbf{x}\mathbf{x}}{|\mathbf{x}|^3}. \tag{1.1}$$

Now we consider a system of N point sources at location $\mathbf{x}_n$ with strength $\mathbf{f}_n$, in the periodic setting, the velocity field is given as

$$\mathbf{u}(\mathbf{x}) = \sum_{n=1}^{N} \sum_{\mathbf{p}} \mathbf{S}(\mathbf{x} - \mathbf{x_n} + \mathbf{p})\mathbf{f}_n, \tag{1.2}$$

where $\mathbf{p}$ form the discrete set $\{[iL_x \ jL_y \ kL_z] : (i,j,k) \in \mathbb{Z}^3\}$ and L_x, L_y and L_z are the periodic lengths in the three directions.

Ewald summation for Stokes Applying the idea of Ewald decomposition, the Eq.(1.2) can be split as following:

$$\mathbf{u}(\mathbf{x}_m) = \sum_{n=1}^{N} \sum_{\mathbf{p}} \mathbf{A}(\xi, \mathbf{x_m} - \mathbf{x_n} + \mathbf{p})\mathbf{f}_n + \frac{1}{V} \sum_{\mathbf{k} \neq 0} \mathbf{B}(\xi, \mathbf{k}) e^{-k^2/4\xi^2} \sum_{n=1}^{N} \mathbf{f}_n e^{-i\mathbf{k}\cdot(\mathbf{x}_m - \mathbf{x}_n)} - \mathbf{u}_{\text{self}} \tag{1.3}$$

where $\mathbf{k} \in \{[2\pi k_i/L_i] : k_i \in \mathbb{Z}, i = 1,2,3\}, k = |\mathbf{k}|, V = L_x L_y L_z$ and ξ is a positive constant known as the Ewald parameter. From the Eq.(1.3) we can see that the velocity field has been split into three parts: one sum in real space ($\mathbf{u}^R$), one sum in frequency domain ($\mathbf{u}^F$) and a self-contribution $\mathbf{u}_{\text{self}}$.

From the formulation by Hasimoto, we have

$$\mathbf{A}(\xi, \mathbf{x}) = 2\left(\frac{\xi e^{-\xi^2 r^2}}{\sqrt{\pi} r^2} + \frac{\text{erfc}(\xi r)}{2r^3}\right)(r^2\mathbf{I} + \mathbf{x}\mathbf{x}) - \frac{4\xi}{\sqrt{\pi}} e^{-\xi^2 r^2}\mathbf{I} \tag{1.4}$$

with $r = |\mathbf{X}|$ and

$$B(\xi, \mathbf{k}) = 8\pi \left(1 + \frac{k^2}{4\xi^2}\right)\frac{1}{k^4}(k^2\mathbf{I} - \mathbf{k}\mathbf{k}) \tag{1.5}$$

and

$$\mathbf{u}_{\text{self}}(\mathbf{x}_m) = \frac{4\xi}{\sqrt{\pi}} \mathbf{f}_m \tag{1.6}$$

Nonuniform fast Fourier transform The nonuniform discrete Fourier transform (NuFFT) of type 1 and 2 is defined as:

$$F(\mathbf{k}) = \frac{1}{N}\sum_{n=1}^{N} f_j e^{-i\mathbf{k}\cdot\mathbf{x}_n} \tag{1.7}$$

and

$$f(x_n) = \sum_{\mathbf{k}} F(\mathbf{k}) e^{i\mathbf{k}\cdot\mathbf{x}_n} \tag{1.8}$$

They can be computed efficiently by fast Gaussian gridding and FFT, we will introduce it in the later part.

Fast summation method in frequency domain Now we focus on the frequency domain part of the summation based on NuFFT:

$$\mathbf{u}^F(\mathbf{x}_m) = \frac{1}{V}\sum_{\mathbf{k}\neq 0} \mathbf{B}(\xi, \mathbf{k}) e^{-k^2/4\xi^2} \sum_{n=1}^{N} \mathbf{f}_n e^{-i\mathbf{k}\cdot(\mathbf{x}_m - \mathbf{x}_n)} \tag{1.9}$$

The formula can be split into

$$\mathbf{u}^F(\mathbf{x}_m) = \frac{1}{V}\sum_{\mathbf{k}\neq 0} \mathbf{B}(\xi, \mathbf{k}) e^{-k^2/4\xi^2} \left(\sum_{n=1}^{N} \mathbf{f}_n e^{i\mathbf{k}\cdot\mathbf{x}_n}\right) e^{-i\mathbf{k}\cdot\mathbf{x}_m} \tag{1.10}$$

which is one NuFFT (type 1) combined with scaling and another NuFFT (type 2).

Reformulation by Gaussian Gridding & FFT The basic idea is convolution by the Gaussian... leave it for later.

For NuFFT of type 1, introducing a free parameter η, we have that

$$\sum_{n=1}^{N} \mathbf{f}_n e^{i\mathbf{k}\cdot\mathbf{x}_m} = e^{\eta k^2/8\xi^2} \sum_{n=1}^{N} \mathbf{f}_n e^{-\eta k^2/8\xi^2} e^{-i(-\mathbf{k})\cdot\mathbf{x}_m} := e^{\eta k^2/8\xi^2} \hat{H}_{-\mathbf{k}} \tag{1.11}$$

while $\hat{H}_{\mathbf{k}}$ is the Fourier transform of

$$H(\mathbf{x}) = \left(\frac{2\xi^2}{\pi\eta}\right)^{3/2} \sum_{n=1}^{N} \mathbf{f}_n e^{-2\xi^2|\mathbf{x}-\mathbf{x}_n|_*^2/\eta} \tag{1.12}$$

where $|\cdot|_*$ denotes distance to closest periodic image.

For the part of type 2 NuFFT, under the same parameter η, first we can simplify

$$\mathbf{u}^F(\mathbf{x}_m) = \frac{1}{V} \sum_{\mathbf{k}\neq 0} \mathbf{B}(\xi,\mathbf{k}) e^{-k^2/4\xi^2} e^{\eta k^2/8\xi^2} \hat{H}_{-\mathbf{k}} e^{-i\mathbf{k}\cdot\mathbf{x}_m} \tag{1.13}$$

$$= \frac{1}{V} \sum_{\mathbf{k}\neq 0} \hat{\hat{H}}_{\mathbf{k}} e^{-\eta k^2/8\xi^2} e^{i\mathbf{k}\cdot\mathbf{x}_m} \tag{1.14}$$

with

$$\hat{\hat{H}}_{\mathbf{k}} := \mathbf{B}(\xi,-\mathbf{k}) \hat{H}_{\mathbf{k}} e^{-(1-\eta)k^2/4\xi^2} \tag{1.15}$$

Then we denote $\tilde{H}(\mathbf{x}_i)$ as the inverse Fourier transform of $\hat{\hat{H}}_{\mathbf{k}}$.

According to the convolution argument, we have that

$$\mathbf{u}^F(\mathbf{x}_m) = \left(\frac{2\xi^2}{\pi\eta}\right)^{3/2} \int_\Omega \tilde{H}(\mathbf{x}) e^{-2\xi^2|\mathbf{x}-\mathbf{x}_m|_*^2/\eta} \mathbf{dx} \tag{1.16}$$

$$\approx \frac{V}{N_{\text{grid}}} \left(\frac{2\xi^2}{\pi\eta}\right)^{3/2} \sum_{\mathbf{x}_{(i)} \text{ in equi-space grid}} \tilde{H}(\mathbf{x}_{(i)}) e^{-2\xi^2|\mathbf{x}_{(i)}-\mathbf{x}_m|_*^2/\eta} \tag{1.17}$$

The approximation integral is spectrally accurate since the integrand is periodic.

Now the only left problem is how to compute Eq.(1.12) and Eq.(1.17). The formula is known as **Gaussian gridding**.

Truncation error of Gaussian Gridding Since the Gaussian function has the infinite support, we have to truncate the Gaussian function to get the numerical value. Denote the grid size of real space (in one dimension) is h and number of grids in the support of truncated Gaussian as P, then the value of Gaussian at the truncated points is $e^{-\xi^2 P^2 h^2/2\eta}$. Let

$$\eta = \left(\frac{Ph\xi}{m}\right)^2 \tag{1.18}$$

we know the Gaussian at the truncated points is $e^{-m^2/2}$.

Moreover, we can bound the quadrature error from Eq.(1.12) and Eq.(1.17) by

$$E^Q \leq 4e^{-\frac{\pi^2 P^2}{2m^2 L^2}} + \text{erfc}(m/\sqrt{2}). \tag{1.19}$$

Note that by choosing the free parameter η as Eq.(1.18), we can decouple the quadrature error with the Ewald parameter ξ, which makes it easier to determine the parameters.

Fast Gaussian Gridding and Parallelization Denote N as the number of particles and M the number of layers in k-space, then if we compute Eq.(1.12) and Eq.(1.17) directly, the computation cost (each) is NM^3 exponential computation, $3NM^3$ multiplications and additions. Since exponential takes much more time than multiplication ($15 \sim 20$ time clocks than $1 \sim 2$ time clocks), a natural idea is to substitute exponential operation with multiplication. **The idea of fast Gaussian gridding is to pre-compute and store the exponential and only do necessary multiplications afterwards.**

Roughly speaking, take 1D situation for example, we have that ($x_{(i)} = ih, x_n = \tilde{x}_n + jh, 0 \leq \tilde{x}_n < h$)

$$e^{-\alpha|x_{(i)}-x_n|^2} = e^{-\alpha|ih-jh-\tilde{x}_n|^2} = e^{-\alpha((i-j)h)^2}\left(e^{2\alpha h\tilde{x}_n}\right)^i e^{-\alpha\tilde{x}_n^2} \tag{1.20}$$

All $e^{-\alpha((i-j)h)^2}, e^{2\alpha h\tilde{x}_n}, e^{-\alpha\tilde{x}_n^2}$ can be precomputed with only $\mathbf{2}(\mathbf{M}+\mathbf{N})$ exponential computations (also that much storage cost) and then we only need to do $2NM$ more multiplications to get all $e^{-\alpha|x_{(i)}-x_n|^2}$. For 3D situation we need $\mathbf{6}(\mathbf{M}+\mathbf{N})$ exponential computations and $2NM^3$ more multiplications. **As long as the exponential takes significantly more time than multiplication, the fast Gaussian gridding will improve the performance and it will benefit more with higher dimension.**

As for the parallelization, the parallelization of Eq.(1.17) is simple since we can compute the velocity filed $\mathbf{u}^F(\mathbf{x}_m)$ at each point source $\mathbf{x}_m$ separately , thus parallelly. Since usually we have a large number of source points and there's no communication between threads, it's a parallelization with high-efficiency.

However, the parallelization of Eq.(1.12) is not that ideal since (almost) every source point contributes to the grid points and it will need great memory cost if we still parallelize on the source points. Thus *instead we parallelize on the grid points to avoid the repeating IO for grid points.* To be more specific, for each source point we parallelize on the grid points that will be affected by the source (since we use a truncated Gaussian) and do the computation serially on the source points. We will see later it's not as efficient as the parallelization of Eq.(1.17), but it did reduce the computation time with more cores.

The Description of Algorithm The algorithm is conducted in the following way:

(1) Choose the free parameter ξ and η wisely and construct the uniform grid according to the problem.

(2) Evaluate $H(\mathbf{x})$ on the grid according to Eq.(1.12) with fast Gaussian gridding.

(3) Conduct FFT on $H(\mathbf{x})$ to get $\hat{H}_{\mathbf{k}}$.

(4) Apply the scaling according to Eq.(1.15) to get $\hat{\tilde{H}}_{\mathbf{k}}$.

(5) Conduct iFFT on $\hat{\tilde{H}}_{\mathbf{k}}$ to get $\tilde{H}(\mathbf{x})$.

(6) Evaluate the $\mathbf{u}^F(\mathbf{x}_m)$ according to Eq.(1.17) with fast Gaussian gridding.

(7) Evaluate the $\mathbf{u}^R(\mathbf{x}_m)$ in real space and get the final result.

Pseudo-code

Error estimation

$$E = E^F + E^Q + E^R \tag{1.21}$$

$$E^F \leq C_F e^{-\frac{M^2\pi^2}{4L\xi^2}} \quad \text{(truncation of k-space)} \tag{1.22}$$

$$E^R \leq C_R(\frac{1}{\xi^2} + \frac{p_\infty}{\xi})e^{-p_\infty^2\xi^2}. \quad \text{(truncation of real space)} \tag{1.23}$$

$$E^Q \leq 4e^{-\frac{\pi^2 P^2}{2m^2L^2}} + \text{erfc}(m/\sqrt{2}) \quad \text{(quadrature)} \tag{1.24}$$

For k-space, the error comes from two parts: the truncation error in the k-space E^F and the quadrature error E^Q to get the velocity fields. With the given parameters L, M, ξ, P, m, we have that the two error has the bound:

$$E^F \leq C_F e^{-\frac{M^2\pi^2}{4L\xi^2}}, \quad C_F \approx 1 \tag{1.25}$$

$$E^Q \leq 4e^{-\frac{\pi^2 P^2}{2m^2L^2}} + \text{erfc}(m/\sqrt{2}) \tag{1.26}$$

For the real space, the truncation error has the bound

$$E^R \leq C_R(\xi + \frac{p_\infty}{\xi})e^{-p_\infty^2\xi^2}, \quad C_R \approx 0.1. \tag{1.27}$$

We can see that all three parts of error decay exponentially with the corresponding parameter E^F with M, E^Q with P and E^R with p_∞). It's also been shown numerically.

Time complexity and speed up The total computational cost of the method is

$$\underbrace{O(NP^3)}_{\text{Gaussian gridding}} + \underbrace{O(M^3 \log(M^3))}_{\text{FFT + iFFT}} + \underbrace{O(M^3)}_{\text{Scaling on k-space}} + \underbrace{O(N^2 p_\infty^3)}_{\text{Real space}} + \underbrace{O(N)}_{self} \tag{1.28}$$

if no FFT acceleration, the cost is

$$\underbrace{O(N^2 M^3)}_{k-space} + \underbrace{O(N^2 p_\infty^3)}_{\text{Real space}} + \underbrace{O(N)}_{self} \tag{1.29}$$

Ewald parameter ξ trades off between real space and k-space!

- Larger $\xi \to$ larger M, smaller p_∞.
- Smaller $\xi \to$ smaller M, larger p_∞.

How we choose the parameters:

- **Balance the error**

- **Balance the time**

Weak scaling result

Strong scaling result

Information about the project Link of the project:

https://github.com/CecilMartin/Ewald_Summation

Team work:

- Zhe Chen: Write codes, presentation
- Guanchun Li: Develop algorithm into pseudo-code, run codes, write slides
- Together: Literature research

References

[1] Greengard, Leslie, and June-Yub Lee. "Accelerating the nonuniform fast Fourier transform." SIAM review 46.3 (2004): 443-454.

[2] Hasimoto, Hidenori. "On the periodic fundamental solutions of the Stokes equations and their application to viscous flow past a cubic array of spheres." Journal of Fluid Mechanics 5.2 (1959): 317-328.

[3] Lindbo, Dag, and Anna-Karin Tornberg. "Spectrally accurate fast summation for periodic Stokes potentials." Journal of Computational Physics 229.23 (2010): 8994-9010.

[4] Frigo, Matteo, and Steven G. Johnson. "The design and implementation of FFTW3." Proceedings of the IEEE 93.2 (2005): 216-231.

[5] https://developer.nvidia.com/cufft

[6] Dagum, Leonardo, and Ramesh Menon. "OpenMP: An industry-standard API for shared-memory programming." Computing in Science & Engineering 1 (1998): 46-55.