CS 6220 Lecture Notes

September 5

Sophia Novitzky

1 Fast Multipole Method

Instead of discussing the potential $\Phi(x_i)$ induced by the charges q_i located at coordinates x_i for $i = 1, \ldots, n$, we will consider the potential Φ in the form

$$\Phi(I_1) = k(I_1, I_1)q(I_1) + \sum_{j \in L_1^n} k(I_1, I_j)q(I_j) + \sum_{j \in L_1^f} k(I_1, I_j)q(I_j), \tag{1.1}$$

where the whole domain $\Omega \in \mathbb{R}^2$ is split into boxes Ω_j with I_j being the set of indexes of points located in domain Ω_j. The collection $\{I_j : j \in L_1^n\}$ denotes all the I_j's such that Ω_j's are **neighbor boxes** of Ω_1, and the collection $\{I_j : j \in L_1^f\}$ contains all I_j's that correspond to boxes **far** from box of I_1.

For simplicity we assume the domain Ω is square. An important part of the FMM is the splitting of the original domain into a tree of boxes (**quadtree**) so that the interactions between charges that are far apart could be approximated by the interactions of the boxes that they belong to. This decreases the computation time because naturally there are less boxes than individual points. For example, each box can easily contain 60 charges.

1.1 Quadtree

The quadtree is constructed through the following algorithm:

- Split Ω into 4 boxes
- Split each of those boxes into 4 boxes
- Repeat until no box has more than B points inside

Figure 1 shows a quadtree.

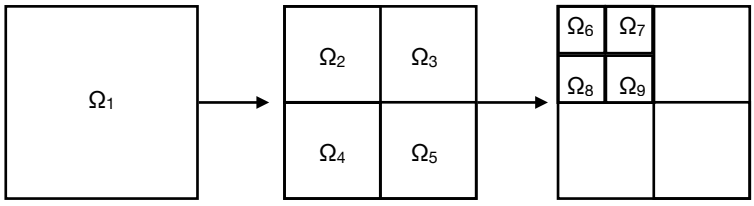

Figure 1: $\lambda = 10$, $\mu = 1$.

To compute the matrix vector product $\Phi = Kq$ efficiently we will first go up the tree starting from the leaves and then back down.

- **up:** Compute multipole expansion for each box:
  - leaf - directly
  - box with children - from the multipole expansion of its children. Multipole expansions give an estimate of the potential of the leafs.
- **down:**
  - Find local expansion for all boxes. Local expansions translate the
  - organized cleverly
  - compute local expansion for a leaf from $O(\log N)$ multipole expansion.

$$v^j = v^j + T^{L2L}_{j,\sigma} v^\sigma, \tag{1.2}$$

where $T_{j,i} \in \mathbb{R}^{pxp}$ and $w^i \in \mathbb{R}^p$ is the i'th multipole expansion. The operators L2L and M2M are translating multipole and local expansions to children/parent boxes, so that we can go from the multipole expansions of the leafs to the local expansions of bigger boxes.

1.2 Quadtree Vocabulary

There are particular phrases and words that are used to describe FMM fully and unambiguously. We will define them here and use them further throughout the notes:

- We build a quad-tree on Ω with **root** Ω_1, which is the domain that will be divided up into finer parts
- Same-sized boxes form a **level**
- The levels are denoted as $l = 0, 1, \ldots, L$; $l = 0$ is the root
- The boxes in the final level L are called **leafs**

For a given box Ω_j (not the root):

- it has a **parent** L^p_j, which is the box from the preceding level that contains Ω_j
- it has **children** L^c_j, which are the boxes in the next level that are contained in Ω_j; if our boxe is a leaf then obviously it doesn't have children
- it has **neighbors** L^n_j (boxes on the same level that touch Ω_j)
- it has an **interaction list** L^{int}_j - the set of all boxes σ such that σ and j are
  1. on the same level
  2. do not touch
  3. their parents are neighbors/ touch

1.3 FMM Implementation

We outline the implementation process of finding the potential Φ from Equation 1.1.
Given the points $\{X_j\}_{j=1}^N$ in domain $\Omega \in \mathbb{R}^2$

1. Build a quadtree on Ω such that no leaf has more than some chosen number of points P on it
2. Fix P - that's the expansion size based on desired accuracy. Specifically, if the desired accuracy is ϵ, then $P \sim \log(1/\epsilon)$. The cost of the algorithm will be proportional to P and is given by $O(PN)$ where N is the total number of points. This comes from the process of finding the multipole expansions as the value of P determines the size of the vector $w^s \in \mathbb{R}^P$.

Algorithm:

1. For each leaf j compute its multipole expansion
$$w^j = T_j^m q(I_j) \tag{1.3}$$

2. Loop over all boxes from fine to coarse, i.e. levels $L-1, L-2, \ldots, 0$. For each box, compute the multipole from L^c (its children):
$$w^j = \sum_{\sigma \in L_j^c} T_{j,\sigma}^{M2M} w^\sigma \tag{1.4}$$

3. Convert multipole to local (M2L) and loop over boxes:
$$v^j = \sum_{\sigma \in L_j^{int}} T_{j,\sigma}^{M2L} w^\sigma \tag{1.5}$$

4. Loop over the boxes from coarse to fine (root to leaf) $0, 1, \ldots, L$:
$$v^j = v^j + T_{j,\sigma}^{L2L} v^\sigma, \tag{1.6}$$
where σ is j's parent

5. For each leaf, add on your own contributions and those of your immediate neighbors
$$\Phi(I_j) = T_j^L v_j + k(I_j, I_i) q(I_j) + \sum_{\sigma \in L^n} k(I_j, I_\sigma) q(I_q). \tag{1.7}$$
T^{L2L}, T^{M2M}, and T^{M2L} depend only on differences between centers of boxes. Cost is $O(PN)$ to compute $\Phi = kq$ to accuracy ϵ and $P \sim \log(1/\epsilon)$.

2 Rank-Structured Matrices

Original application. One example of application of FMM/ rank-structured matrices comes from physics, where multitudes of particles exert forces on each other. Usually the forces depend on the distance between the particles. The closer the particles are to each other, the stronger is the force between them. With a sufficient distance, however, the force between a pair of particles becomes negligible. In order to compute these forces, the Fast Multipole Method is the go-to choice. In fact, the algorithm was developed specifically to study particle interaction in a many-particle system.

Common application. Another example from physics is the Laplace's equation. For a bounded domain D in space measured in $\mathbb{R}^2$ or $\mathbb{R}^3$ with a sufficiently smooth boundary ∂D, Laplace's equation on D with Dirichlet boundary condition is

$$-\Delta u = 0 \quad \text{on} \quad D \tag{2.8}$$
$$u = f \quad \text{on} \quad \partial D \tag{2.9}$$

We wish to find $u(x)$ in D.
We know the Green's function

$$G(x,y) = \begin{cases} \frac{1}{2\pi} \cdot \log(|x-y|), & d=2 \\ \frac{1}{4\pi} \cdot \frac{1}{|x-y|}, & d=3 \end{cases} \tag{2.10}$$

which allows us to determine $u(x)$ through the convolution

$$u(x) = \int_{\partial D} f(y) G(x,y) ds(y) \tag{2.11}$$

that incorporates the desired boundary condition. Noting the similarity between the Green's function $G(x, y)$ and the function $k(x, y)$ from Equation 1.1, we see that $u(x)$ is the continuous analogue of the potential function $\Phi(x)$. If we approximate $u(x)$ as

$$u(x) = \sum_{y_i} f(y_i)G(x, y_i) \tag{2.12}$$

we can apply the FMM to find the solution to Laplace's equation $u(x)$.

A good source to read is an article by Lexing Ying

http://web.stanford.edu/∼lexing/publication/Ying2009Chapter.pdf

where the author discusses several fast algorithms including FMM for boundary integral problems. This includes a deeper discussion of application to Laplace's equation as well as an application to Helmholtz equation.