Fast Multipole Methods: Fundamentals & Applications

Ramani Duraiswami
Nail A. Gumerov

Week 1. Introduction.

- What are multipole methods and what is this course about.
- Problems from physics, mathematics, computer vision, statistics, etc. that can be solved with fast multipole methods.
- Historical review of the fast multipole method.
- Review of Complexity
- Homework.

Course Mechanics

- Background Needed
  - Linear Algebra
    - Matrices, Vectors, Linear Systems, LU Decomposition, Eigenvalue problem, SVD
  - Numerical Analysis
    - (Interpolation, Approximation, Finite Differences, Taylor Series, etc.)
  - Programming
    - Matlab, C/C++ and/or FORTRAN
- Participation essential !

Course Requirements

- Ideally
  - you have a problem in mind
  - would like to explore how FMM could be applied
- At the end of the course you will have a familiarity with
  - Application of FMM to different problems
  - Data structures for FMM, and analysis related to it.
- Course focus is on applying the methods to achieve solution.
  - Theory as needed to proceed

Homework

- Will try to have it every week
- Will not be excessive
- Essential for learning --- must do as opposed to just read.
- Homework handed out last class of a week.
- Due last class of next week
- Thanksgiving week no homework
  - (you can turn in the previous homework after thanksgiving)

Projects & Exams

- There will be a final project that will require you to implement an FMM algorithm in a field of your choice,
  - account for 20% of the grade.
- Project to be chosen around October 14.
  - Implementation (reimplementation) of an FMM algorithm (hints and help will be given)
- Exams
  - intermediate exam worth 10%, week of October 14
  - final exam worth 20%. Finals Week

Class web & Mailing List

- http://www.umiacs.umd.edu/~ramani/cmsc878r
- cmsc878r@umiacs.umd.edu
- Homework will be posted on web
  - No late homework (except by timely prior arrangement)
  - (no web or email submissions)
- Solutions will be posted after homework collected
- Links to papers etc.

Introductions

- Email addresses
- What are your interests?
- What do you want us to cover?
  - An outline is posted at

What is the Fast Multipole Method?

- An algorithm for achieving fast products of particular dense matrices with vectors
- Similar to the Fast Fourier Transform
  - Matrix entries are uniformly sampled complex exponentials
- For FMM, matrix entries are
  - Derived from particular functions
  - Functions satisfy known "translation" theorems
- Name is a bit unfortunate
  - What the heck is a multipole?
  - We will return to this …

Vectors and Matrices

d dimensional column vector $\mathbf{x}$ and its transpose

$$\mathbf{x} = \begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_d \end{pmatrix} \quad \text{and} \quad \mathbf{x}^t = (x_1 \; x_2 \; \ldots \; x_d)$$

- $n \times d$ dimensional matrix **M** and its transpose $\mathbf{M}^t$

$$\mathbf{M} = \begin{pmatrix} m_{11} & m_{12} & m_{13} & \ldots & m_{1d} \\ m_{21} & m_{22} & m_{23} & \ldots & m_{2d} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ m_{n1} & m_{n2} & m_{n3} & \ldots & m_{nd} \end{pmatrix} \text{ and}$$

$$\mathbf{M}^t = \begin{pmatrix} m_{11} & m_{21} & \ldots & m_{n1} \\ m_{12} & m_{22} & \ldots & m_{n2} \\ m_{13} & m_{23} & \ldots & m_{n3} \\ \vdots & \vdots & \ddots & \vdots \\ m_{1d} & m_{2d} & \ldots & m_{nd} \end{pmatrix}.$$

Matrix vector product

$s_1 = m_{11}\, x_1 + m_{12}\, x_2 + \ldots + m_{1d}\, x_d$

$s_2 = m_{21}\, x_1 + m_{22}\, x_2 + \ldots + m_{2d}\, x_d$

…

$s_n = m_{n1}\, x_1 + m_{n2}\, x_2 + \ldots + m_{nd}\, x_d$

- Matrix vector product is identical to a sum

  $s_i = \sum_{j=1}^{d} m_{ij}\, x_j$
- So algorithm for fast matrix vector products is also a fast summation algorithm

- d products and sums per line
- N lines
- Total Nd products and Nd sums to calculate N entries

Fast Fourier Transform

- If entries of matrix are

  $m_{kn} = e^{2\pi i k n}$ where k and n are integers
- Then complexity of matrix vector product goes from $O(N^2)$ to $O(N \log N)$
- Allowed several industries to develop …
- A crucial invention of the 2nd half of the 20th century
- (see numerical recipes www.nr.com)

Fast Multipole Methods (FMM)

- Introduced by Rokhlin & Greengard in 1987
- Called one of the 10 most significant advances in computing of the 20th century
- Speeds up matrix-vector products (sums) of a particular type

$$s(x_j) = \sum_{i=1}^{N} \alpha_i \phi(x_j - x_i), \quad \{s_j\} = [\Phi_{ji}]\{\alpha_i\}.$$

- Above sum requires $O(MN)$ operations.
- For a given precision ε the FMM achieves the evaluation in $O(M+N)$ operations.

- Can accelerate matrix vector products
  - Convert $O(N^2)$ to $O(N \log N)$
- However, can also accelerate linear system solution
  - Convert $O(N^3)$ to $O(kN \log N)$

Linear System Solution

- Solution of most problems in scientific computing reduces to solution of a linear system

$$A\,x = b$$

- For dense A, direct solution requires $O(N^3)$ operations
  - For large N (e.g. 10^7 – 10^8) this is prohibitive
- Iterative methods can achieve solutions in k steps
- Each step typically involves 2 matrix vector multiplications and thus solution time is

$$O(2\,k * \text{cost of matrix-vector multiplication})$$

- For general dense matrices this cost is $O(kN^2)$
- With FMM this cost becomes $O(kN)$ or $O(kN \log N)$
- If k is independent of N reduction is significant

Memory complexity

- Sometimes we are not able to fit a problem in available memory
  - Don't care how long solution takes, just if we can solve it
- To store a $N \times N$ matrix we need N^2 locations
  - 1 GB RAM = 1024^3 =1,073,741,824 bytes
  - => largest N is 32,768
- "Out of core" algorithms copy partial results to disk, and keep only necessary part of the matrix in memory
- FMM allows reduction of memory complexity as well
  - *Elements of the matrix required for the product can be generated as needed*

A very simple algorithm

- Not FMM, but has some key ideas
- Consider

$$S(x_i)=\sum_{j=1}^{N} \alpha_j (x_i - y_j)^2 \qquad i=1, \ldots ,M$$

- Naïve way to evaluate the sum will require MN operations
- Instead can write the sum as

$$S(x_i)=(\sum_{j=1}^{N} \alpha_j)x_i^2 + (\sum_{j=1}^{N} \alpha_j y_j^2) - 2x_i(\sum_{j=1}^{N} \alpha_j y_j)$$

  - Can evaluate each bracketed sum over j and evaluate an expression of the type

$$S(x_i)=\beta x_i^2 + \gamma - 2x_i\delta$$

  - Requires O(M+N) operations
- Key idea – use of analytical manipulation of series to achieve faster summation

---

Approximate evaluation

- FMM introduces another key idea or "philosophy"
  - In scientific computing we almost never seek exact answers
  - At best, "exact" means to "machine precision"
- So instead of solving the problem we can solve a "nearby" problem that gives "almost" the same answer
- If this "nearby" problem is much easier to solve, and we can bound the error analytically we are done.
- In the case of the FMM
  - Manipulate series to achieve approximate evaluation
  - Use analytical expression to bound the error
- FFT is exact … FMM can be arbitrarily accurate

Some FMM algorithms

- Molecular and stellar dynamics
  - Computation of force fields and dynamics
- Interpolation with Radial Basis Functions
- Solution of acoustical scattering problems
  - Helmholtz Equation
- Electromagnetic Wave scattering
  - Maxwell's equations
- Fluid Mechanics: Potential flow, vortex flow
  - Laplace/Poisson equations
- Fast nonuniform Fourier transform

Applications – I Interpolation

- Given a scattered data set with points and values $\{\mathbf{x}_i, f_i\}$
- Build a representation of the function f($\mathbf{x}$)
  - That satisfies f($\mathbf{x}_i$)=f_i
  - Can be evaluated at new points
- One approach use "radial-basis functions"

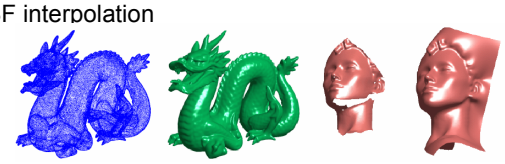
$$f(\mathbf{x})=\sum_i^N \alpha_i R(\mathbf{x}-\mathbf{x}_i) + p(\mathbf{x})$$

$$f_j=\sum_i^N \alpha_i R(\mathbf{x}_j-\mathbf{x}_i) + p(\mathbf{x}_j)$$

- Two problems
  - Determining α_i
  - Knowing α_i determine the product at many new points $\mathbf{x}_j$
- Both can be solved via FMM (Cherrie et al, 2001)

Applications 2

- RBF interpolation



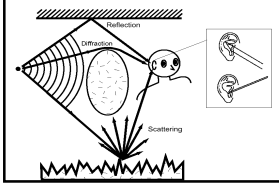
Cherrie et al 2001

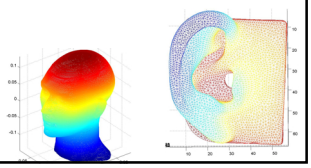
Applications 2

- Sound scattering off rooms and bodies
  - Need to know the scattering properties of the head and body (our interest)

$$\nabla^2 P + k^2 P = 0 \qquad \frac{\partial P}{\partial n} + i\sigma P = g \qquad \lim_{r\to\infty} r\left(\frac{\partial P}{\partial r} - ikP\right) = 0$$

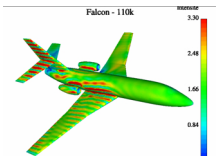
$$C(x)\,p(x) = \int_{\Gamma_y}\left[G(x,y;k)\frac{\partial\ p(y)}{\partial\ n_y} - \frac{\partial\ G(x,y;k)}{\partial\ n_y}p(y)\right]d\Gamma_y \qquad G(\mathbf{x},\mathbf{y}) = \frac{e^{ik|\mathbf{x}-\mathbf{y}|}}{4\pi|\mathbf{x}-\mathbf{y}|}$$





Applications: EM wave scattering

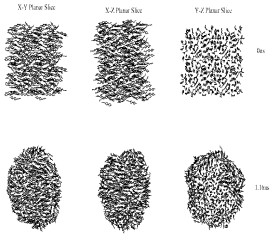
- Similar to acoustic scattering
- Send waves and measure scattered waves
- Attempt to figure out object from the measured waves
- Need to know "Radar cross-section"
- Many applications
  - Light scattering
  - Radar
  - Antenna design
  - ….



Darve 2001

Molecular and stellar dynamics

- Many particles distributed in space
- Particles exert a force on each other
- Simplest case force obeys an inverse-square law (gravity, coulombic interaction)



$$\frac{d\mathbf{x}_i}{dt} = F_i, \qquad F_i = \sum_{\substack{j=1\\ j\neq i}}^{N} q_i q_j \frac{(\mathbf{x}_i - \mathbf{x}_j)}{|\mathbf{x}_i - \mathbf{x}_j|^3}$$

Fluid mechanics

- Incompressible Navier Stokes Equation

$$\nabla \cdot \mathbf{u} = 0 \qquad \mathbf{u} = \nabla\phi + \nabla \times \mathbf{A}$$

$$\rho\left(\frac{\partial \mathbf{u}}{\partial t} + (\mathbf{u}\cdot\nabla)\mathbf{u}\right) + \nabla p = \mu\nabla^2\mathbf{u} \qquad \boldsymbol{\omega} = -\nabla^2\mathbf{A}$$

- Laplace equation for potential and Poisson equation for vorticity
- Solved via particle methods …