快速多级子算法 (FMM) 介绍和实现

原创 www.cae-sim.com 多物理场仿真技术



国内对FMM (Fast Multipole Method)的介绍都比较复杂,涉及大量的计算公式。

本文试图用最简单的语言介绍FMM的原理和实现,并介绍使用C++开发的快速多级算法模块,用户可自定义核函数,收敛标准,截断系数等参数,可用于实际工程,后续会介绍FMM与边界元方法(Boundary Element Method)/ 矩量法(Method of Moment)结合解决大规模声场,电磁场问题。

在前面简单介绍了快速多级算法 FMM。快速多级子算法能加快解决非对称满秩矩阵,扩展了矩量法,边界元等方法的应用规模,使其能解决较大规模的实际工程问题。

以下是电磁计算软件FEKO中关于多层快速多级算法(MultiLevelFMM)的介绍,是对FMM的一种改讲:

https://www.feko.info/product-detail/numerical methods/mlfmm

1. 概述

FMM算法的提出来源于多粒子系统相互作用的势场计算,比如带电粒子或者天体之间引力等。以静电场为例,空间中N个带点离子构成的系统,第i个离子所在位置的静电势 A(xi)表示为:

$$A(x_i) = \sum_{j=1 \atop \text{minimizer}}^{N} \frac{m_j}{r_{ij}}$$

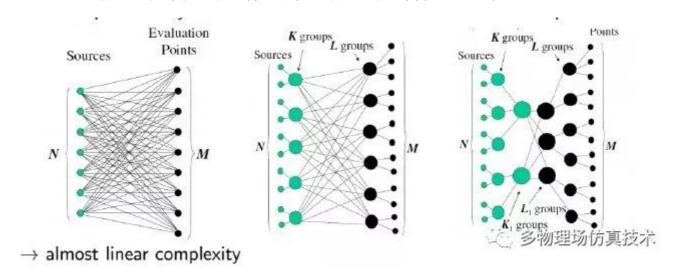
其中

1. i不等于j;

- 2. xi是第i个粒子所在的坐标;
- 3. mj是第j个粒子所占的权重,与带电量呈正比;
- 4. rij是第i个粒子和第j个粒子之间的距离;

按照常规计算方法,对N个粒子实现求和问题,计算量达到O(N^2);

在BEM MOM 等数值计算中,一次这样的求和也就是一次矩阵和向量的乘法迭代。

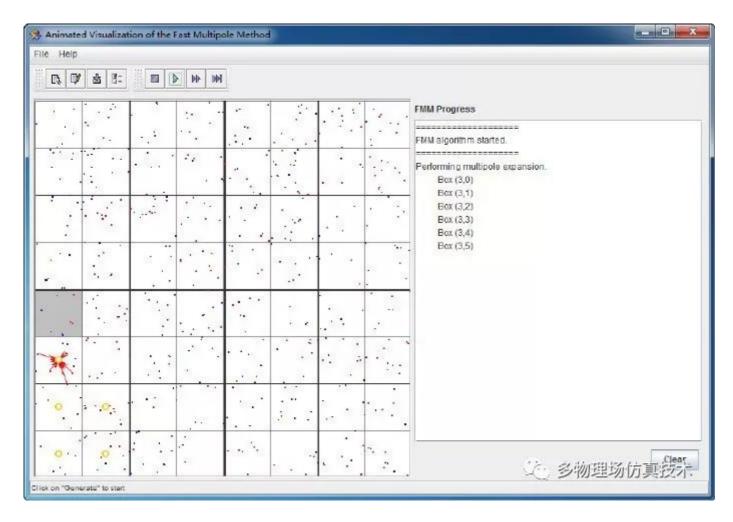


FMM的实现基本思想是以树形结构为基础,通过多级展开和局部展开,把原对象进行分层分组,将 N*N的关系转换为少数组对象之间的关系(如上图),从而减少计算量。该算法实现的核心是如何 把每个对象归纳到一组对象中,这个主要是通过动态树结构来实现的。

计算过程如下:

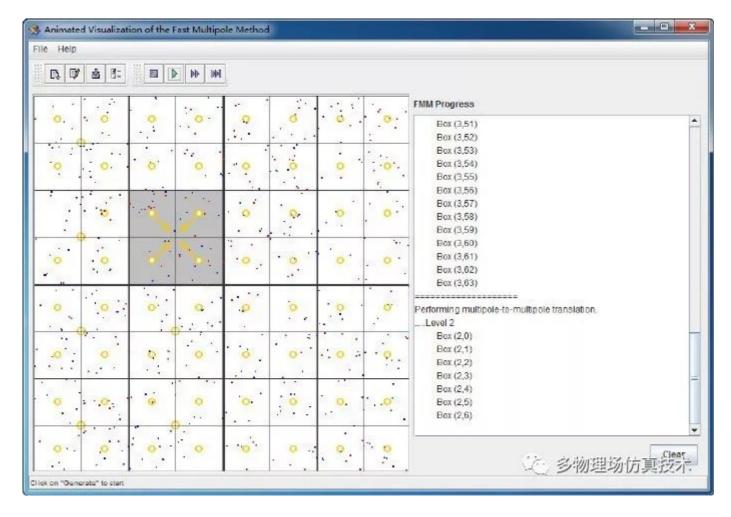
1. 多级展开:

多级展开将叶子节点内(每个小方块内)所有多级展开系数累加,即可以得到该叶子节点的多级展开系数,展开节点为叶子节点的中心。



2.多级-多级转换 (Level-2)

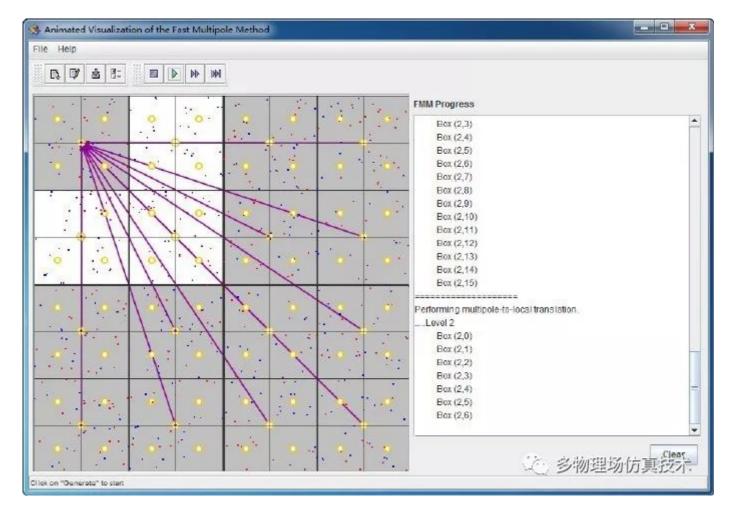
将源点和目标点系数进行转换



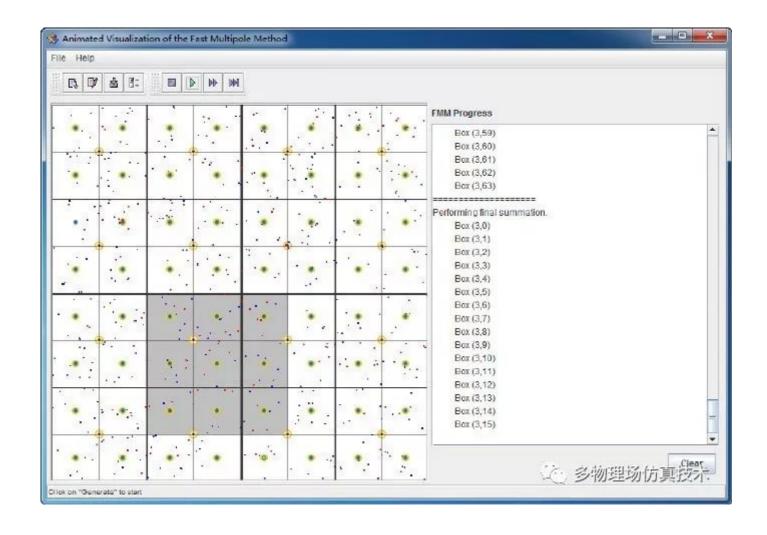
3.多级-局部转换 (Level-2)

利用多级展开向局部展开系数的传递关系式,将该节点的"相互作用表"(紫色直线连接区域部分即为左上角点的相互作用表)中所有节点的多级展开系数传递,并累加到该节点的局部展开系数,局部展开点是该点的中心

4.局部-局部转换 (Level-2)



- 5. 多级-局部转换 (Level-3)
- 6. 最终求和



2. 实现

FMM基础模块,该模块采用C++开发,使用面向对象思想,定义了基础的核函数,并实现了几种简单的核函数,用户可以继承定义自己的核函数,从而应用到流体,电磁,声场等各种需要计算非对称满秩矩阵的数值计算方法中。

基本函数和数据结构:

主要函数:

CreateFMMTree //递归的方法构建节点,生成四叉树或者八叉树结构;

L2L/Local2Local //局部到局部转换;

M2L/Moment2Local //多级到局部转换;

M2M/Moment2Moment //多级到多级转换;

UpwardPass //树结构向上遍历;

DownwardPass //树结构向下遍历;

下图显示了采用Laplace核函数,一次迭代即矩阵*向量,FMM和高斯迭代所用的时间比较,可以看出FMM所用时间与N几乎呈线性,而且精度很高。考虑到求解线性方程组时需要计算N次,在N

达到几万以后,高斯迭代法在PC上基本求不出结果。

-4	A	В	C	D	E	
1	N 值	FMM(秒)	高斯迭代(秒)	FMM计算误差	层数	
2	1000	7. 857	0.599	4.30E-05	2	
3	2000	10.568	2. 379	3.66E-05	2	
4	5000	11.403	14. 892	3.43E-05	2	
5	10000	13.007	69. 856	3.24E-05	2	
6	20000	19. 382	237.64	3.16E-05	2	
7	30000	30.06	541.574	3.14E-05	2	
8	40000	45. 21	961. 239	3.11E-05	2	
9	50000	65. 579	1437.07	3.11E-05	2	
10						
11		CPU时间比较				
12	1600					
13	1400	400				
14	1200	1200				
15	1000	1000				
16 17		800				
18	40,000	600				
19						
20	400					
21	200	100 Miles				
22	0	00-0				
23		0 10000	20000 30000 40	50000 50000	60000	
24						
25		→ FMM(秒) → 高斯迭代(秒) 多物理场仿真技术				
26						