

# **FMMYUK-U: Uniform Fast Multipole Method for the Yukawa Potential**

## **User's Guide and Programmer's Manual**

**Version Beta**

Jingfang Huang

CB# 3250, Department of Mathematics, University of North Carolina,  
Chapel Hill, NC 27599-3250.

\* Comments or requests concerning the FMMYUK-U program can be addressed to:

Dr. Jingfang Huang, Tel.: 01 919-962-9621; fax: 01 919-962-9345. Email:

huang@amath.unc.edu.

# **FMMYUK-U: Uniform Fast Multipole Method for the Yukawa Potential**

Contact information

Jingfang Huang (huang@amath.unc.edu)

Department of Mathematics, CB# 3250

University of North Carolina at Chapel Hill

Chapel Hill, NC 27599-3250

Copyright (c) 2008, Jingfang Huang.

**All Rights Reserved.**

This program is free software; you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation; either version 2 of the License, or (at your option) any later version.

This program is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

You should have received a copy of the GNU General Public License along with this program; if not, write to the Free Software Foundation, Inc., 675 Mass Ave, Cambridge, MA 02139, USA.

# CONTENTS

<b>Copyright</b>	<b>ii</b>
<b>1 Introduction</b>	<b>1</b>
1.1 What is Fast Multipole Method . . . . .	1
1.2 What is FMMYUK-U? . . . . .	3
1.3 Getting Started . . . . .	3
1.4 Recommended Reading . . . . .	4
<b>Part I: User's Guide</b>	<b>6</b>
<b>2 Methods</b>	<b>6</b>
2.1 A Brief History . . . . .	6
2.2 New version fast multipole method . . . . .	7
<b>3 Usage Examples</b>	<b>10</b>
3.1 A Sample Driver File . . . . .	10
<b>4 Using FMMYUK-U</b>	<b>11</b>
4.1 Compile and installation . . . . .	11
<b>Part II: Programmer's Reference Guide</b>	<b>12</b>
<b>5 Programmer's Notes</b>	<b>12</b>
5.1 Programming Language . . . . .	12
5.2 Special functions . . . . .	12

<b>6</b>	<b>Variables and Data Structure</b>	<b>13</b>
6.1	Important variables in header file . . . . .	13
6.2	Important variables in the code . . . . .	13
6.3	Memory Allocation . . . . .	13
<b>7</b>	<b>Important Subroutines</b>	<b>14</b>
<b>8</b>	<b>Frequently Asked Questions</b>	<b>15</b>
	<b>Acknowledgments</b>	<b>16</b>

# LIST OF FIGURES

# LIST OF TABLES

# 1. INTRODUCTION

## 1.1. What is Fast Multipole Method

There are many websites discussing the fast multipole methods, including the website by the authors of this manual (<http://www.fastmultipole.org/>). The following are from a simple google search.

### 1. From Wikipedia, the free encyclopedia

([http://en.wikipedia.org/wiki/Fast\\_Multipole\\_Method](http://en.wikipedia.org/wiki/Fast_Multipole_Method))

“The Fast Multipole Method (FMM) is a mathematical technique that was developed to speed up the calculation of long-ranged forces in the N-Body Problem. It does this by expanding the system Green’s Function using a Multipole Expansion, which allows one to group sources that lie close together and treat them as if they are a single source.

The FMM has also been applied in accelerating the iterative solver in the Method of Moments (MOM) as applied to computational electromagnetics problems. The FMM was first introduced in this manner by Greengard and Rokhlin [1] and is based on the multipole expansion of the vector Helmholtz equation. By treating the interactions between far-away basis functions using the FMM, the corresponding matrix elements do not need to explicitly stored, resulting in a tremendous savings in the required system memory. If the FMM is then applied in a hierarchical manner, it can improve the complexity matrix-vector product in an iterative solver from  $O(N^2)$  to  $O(N \log N)$ . This has expanded the area of applicability of the MOM to far greater problems than were previously possible.

The FMM, introduced by Rokhlin and Greengard, has been acclaimed as one of the greatest algorithms of the 20th century. The FMM algorithm dramatically reduces the complexity of matrix-vector multiplication involving a certain type of dense matrix,

which can arise out of many physical systems.

The FMM has also been applied for efficiently treating the Coulomb interaction in Hartree-Fock and Kohn-Sham Density Functional Theory calculations in quantum chemistry.”

**2. A short course on fast multipole methods,**  
([http://math.nyu.edu/faculty/greengar/shortcourse\\_fmm.pdf](http://math.nyu.edu/faculty/greengar/shortcourse_fmm.pdf))

**3. Citation from the Steele Prize,**  
(<http://www.ams.org/notices/200104/comm-steele.pdf>)

“For the paper by L. Greengard and V. Rokhlin A fast algorithm for particle simulations, J. Comput. Phys. 73, no. 2 (1987), 325348.

This is one of the most important papers in numerical analysis and scientific computing in the last two decades. This paper introduces an algorithm, the fast multipole method (FMM), that speeds up the computation of certain types of sums. It showed that several ideas in harmonic analysis, far field expansions, and multiscale analysis based on dyadic decompositions of space, together with some further innovations, such as so-called translation operators, could be combined to produce a practical algorithm that would make possible scientific and engineering computations that would have been impossible before. While the paper itself treats a very special case, it contains the fundamental ideas for a vast variety of generalizations and applications. The paper combines both the elegance and originality of the algorithm itself and the hard analysis of the proofs. These sums arise in a variety of applications, ranging from computational astronomy (computing the gravitational interaction of stars in a galaxy), to molecular dynamics (the Coulomb interaction of charges in a large molecule), to engineering computations of radar scattering, to anything related to solutions of the Laplace equation (vortex methods for incompressible flow). The FMM has been enormously influential not just because of its basic scientific applications, but also because it forms the basis of commercial software for electronic packaging analysis, semiconductor design, and electro-magnetic applications. It has brought previously intractable computational problems within reach.”



## 1.2. What is FMMYUK-U?

Although extremely useful, the FMM algorithms have been considered one of the most complicated to program in scientific computing, especially the new version of fast multipole method introduced by Greengard and Rokhlin in 1997.

Dr. Jingfang Huang started his work on the FMM since 1995 while he was a Ph.D. Student at the Courant Institute of Mathematical Sciences at the New York University, under the direction of Professor Leslie F. Greengard, one of the inventors of the FMM. With the help and encouragements from Professors Greengard and Rokhlin, Dr. Jingfang Huang has been working on the fast multipole methods for different potentials in the past, and has finished the preliminary version of both the uniform and adaptive FMMs for the Laplace, Yukawa, and low frequency Helmholtz solvers. These solvers are combined together and called the *FMM Suite*.

The FMMYUK-U is one of the solvers in the FMM Suite, it computes the screened Coulombic (Yukawa) interactions of N particles where the data structure is based on a uniform octree. This solver is open source under the GPL 2.0. Please see the included licence file for details.

The authors of this package hopes that this solver can be useful for research in scientific computing society. The code is still under active improvement. We welcome suggestions, bug reports, comments, and collaborating opportunities from our research community.

## 1.3. Getting Started

This code computes

$$\Phi_j = \sum_{i \neq j}^{NATOMS} q_i \cdot e^{-BETA \cdot r_{i,j}} / r_{i,j}$$

where  $r_{i,j} = |x_i - x_j|$  is the distance between the points  $x_i$  and  $x_j$ .

The main subroutine is the following

FMMYUK\_UNI(BETA,NATOMS,ZAT,CHARGE,POT,FIELD,NLEV,IER)

The input variables include

- BETA(real \*8): the inverse of the screening length.
- NATOMS(integer \*4): the total number of particles.
- ZAT(3,natoms) (real \*8): the location of each particle. Note that all particles must be located inside the unit box centered at (0,0,0).
- CHARGE(natoms) (real \*8): the charge each particle carries.
- NLEV (integer \*4): the number of levels in the octree structure. For the uniform code, nlev should be chosen such that the local interactions cost about the same time as the far field evaluations. Note that on a computer with 2G memory, NLEV can not be greater than 7.

The output variables are

- POT(natoms) (real \*8): the potential at each particle location.
- FIELD(3,natoms) (real \*8): the force field at each particle location.
- IER (integer \*4) : an index for error message.

For advanced users, check parm-uniyuk.h to change the accuracy of the algorithm. Currently only 3 digits (NTERMS=NLAMBS=9) and 6 digits (NTERMS=NLAMBS=18) accuracy are allowed.

The users are recommended to read the driver file for further information on how to use the code.

## 1.4. Recommended Reading

We recommend the advanced programmers the following papers for understanding the adaptive new version of fast multipole methods used in this package.

- Greengard, Leslie; Rokhlin, Vladimir, “A new version of the fast multipole method for the Laplace equation in three dimensions”. Acta numerica, 1997, 229–269, Acta Numer., 6, Cambridge Univ. Press, Cambridge, 1997.

- Cheng, H.; Greengard, L.; Rokhlin, V. “A fast adaptive multipole algorithm in three dimensions.” J. Comput. Phys. 155 (1999), no. 2, 468–498.
- Greengard, Leslie F.; Huang, Jingfang, “A new version of the fast multipole method for screened Coulomb interactions in three dimensions”. J. Comput. Phys. 180 (2002), no. 2, 642–658.

## 2. METHODS

In this chapter, we discuss the technical details of the algorithm.

### 2.1. A Brief History

Clearly, direct calculation of the  $N$ -body electrostatic particle screened Coulombic interactions requires  $O(N^2)$  operations. In the last twenty years or so, novel fast summation algorithms have been developed to reduce the cost to  $O(N \log N)$  or the asymptotically optimal  $O(N)$ . These algorithms include the hierarchical “tree code”,<sup>2,3</sup> fast Fourier transform (FFT) based algorithms such as the precorrected FFT (pFFT)<sup>16</sup> and the particle-mesh Ewald (PME) methods,<sup>5</sup> the hierarchical SVD method,<sup>11</sup> and FFT on multipoles.<sup>14,15</sup> Further improvements show that asymptotically optimal  $O(N)$  complexity can be achieved by using the wavelet techniques or the fast multipole method (FMM).<sup>8</sup> However, as revealed by previous numerical experiments, although asymptotically optimal, the original FMM<sup>8</sup> turns out to be less efficient for problem sizes of current interest when compared with the tree code and FFT based  $O(N \log N)$  techniques, due to the huge prefactor in  $O(N)$ .

In 1997, Greengard and Rokhlin introduced a new version of FMM<sup>9</sup> for the Laplace equation to further accelerate the performance of the FMM. Compared with the original FMM, the plane wave expansion based diagonal translation operators dramatically reduce the prefactor in the  $O(N)$  new version FMM, especially in three dimensions where a break-even point of approximately 600 for 6 digits precision is numerically observed. The package FMMYUK-U is based on the new version of FMM. Perhaps due to its complexity in theory and programming, we are unaware of any previous implementations of the new version FMM for the Yukawa potential.

## 2.2. New version fast multipole method

The original idea of FMM is to subdivide the summation system of  $N$  particles into hierarchical groups of particles, and the potentials produced by far-field particles for a given particle are approximated by using the multipole expansions (Figure 1a). The fundamental observation in the multipole expansion based methods is that the numerical rank of the far field interactions is relatively low and hence can be approximated by  $P$  terms (depending on the prescribed accuracy) of the so-called “multipole expansion”,

$$\Phi(R, \theta, \phi) = \sum_{i=1}^N q_i \cdot \frac{1}{|\vec{R} - \vec{\rho}_i|} \approx \sum_{n=0}^P \sum_{m=-n}^n M_n^m \frac{Y_n^m(\theta, \phi)}{R^{n+1}} \quad (2.1)$$

where the multipole coefficients,

$$M_n^m = 8 \sum_{i=1}^N q_i \cdot Y_n^{-m}(\alpha_i, \beta_i) \quad (2.2)$$

where the spherical harmonic function of order  $n$  and degree  $m$  is defined according to the formula,<sup>1</sup>

$$Y_n^m(\theta, \phi) = \sqrt{\frac{(2n+1)(n-|m|)!}{4\pi(n+|m|)!}} \cdot P_n^{|m|}(\cos\theta) e^{im\phi} \quad (2.3)$$

For the Debye-Hückel (screened Coulombic) interaction, a similar expansion can be written as follows,

$$\Phi(R, \theta, \phi) = \sum_{i=1}^N q_i \cdot \frac{e^{-\kappa|\vec{R} - \vec{\rho}_i|}}{|\vec{R} - \vec{\rho}_i|} \approx \sum_{n=0}^P \sum_{m=-n}^n M_n^m \cdot k_n(\kappa R) \cdot Y_n^m(\theta, \phi) \quad (2.4)$$

where the multipole coefficients,

$$M_n^m = 8\kappa \sum_{i=1}^N q_i \cdot i_n(\kappa \rho_i) \cdot Y_n^{-m}(\alpha_i, \beta_i) \quad (2.5)$$

where  $i_n(r)$  and  $k_n(r)$  are modified spherical Bessel and modified spherical Hankel functions respectively. The modified spherical Bessel and modified spherical Hankel

functions are defined in terms of the conventional Bessel function via,<sup>1</sup>

$$I_\nu(r) = i^{-\nu} J_\nu(ir), \quad (2.6)$$

$$K_\nu(r) = \frac{\pi}{2 \sin \nu \pi} [I_{-\nu}(r) - I_\nu(r)], \quad (2.7)$$

$$i_n(r) = \sqrt{\frac{\pi}{2r}} I_{n+1/2}(r), \quad (2.8)$$

$$k_n(r) = \sqrt{\frac{\pi}{2r}} K_{n+1/2}(r). \quad (2.9)$$

For arbitrary distributions of particles, a hierarchical oct-tree (in 3D) is generated so each particle is associated with different boxes at different levels, and a divide-and-conquer strategy is applied to account for the far field interactions at each level in the tree structure. In the “tree code” developed by Appel,<sup>2</sup> and Barnes and Hut,<sup>3</sup> as each particle interacts with 189 boxes in its “interaction list” through  $P$  terms of multipole expansions at each level and there are  $O(\log N)$  levels, the total amount of operations is approximately  $189P^2N \log N$ . The tree code was later improved by Greengard and Rokhlin in 1987.<sup>8</sup> In their original FMM, local expansions (under a different coordinate system) were introduced to accumulate information from the multipole expansions.

$$\Phi(R, \theta, \phi) = \sum_{i=1}^N q_i \cdot \frac{1}{|\vec{R} - \vec{\rho}_i|} \approx \sum_{n=0}^P \sum_{m=-n}^{m=n} L_n^m \cdot R^n Y_n^m(\theta, \phi) \quad (2.10)$$

where  $L_n^m$  are local expansion coefficients.

$$L_n^m = 8 \sum_{i=1}^N q_i \cdot \frac{Y_n^{-m}(\alpha_i, \beta_i)}{\rho_i^{n+1}} \quad (2.11)$$

For the screened Coulombic interaction, a similar expansion can be written as follows,

$$\Phi(R, \theta, \phi) = \sum_{i=1}^N q_i \cdot \frac{e^{-\kappa|\vec{R} - \vec{\rho}_i|}}{|\vec{R} - \vec{\rho}_i|} \approx \sum_{n=0}^P \sum_{m=-n}^{m=n} L_n^m \cdot i_n(\kappa R) \cdot Y_n^m(\theta, \phi) \quad (2.12)$$

where

$$L_n^m = 8\kappa \sum_{i=1}^N q_i k_n(\kappa \rho_i) \cdot Y_n^{-m}(\alpha_i, \beta_i), \quad (2.13)$$

As the particles only interact with boxes and other particles at the finest level, and information at higher levels is transferred using a combination of multipole and local

expansions, and the original FMM is asymptotically optimal  $O(N)$ . However, because the multipole to local translation requires prohibitive  $189P^4$  operations for each box, the huge prefactor makes the original FMM less competitive with the tree code and other FFT based methods. In 1997, a new version of FMM was introduced by Greengard and Rokhlin<sup>9</sup> for the Laplace equation. Compared with the original FMM, a plane wave expansion based diagonal translation operator is introduced and the original  $189P^4$  operations were reduced to  $40P^2 + 2P^3$ . In our algorithm, we adapt the new version of FMM for the screened Coulomb interactions. Preliminary numerical experiments show that the overall break even point of the new version FMM becomes 600 with 6-digit accuracy and about 400 for 3-digit.

## 3. USAGE EXAMPLES

### 3.1. A Sample Driver File

A sample driver file is provided in the package, check uniyukdriver.f

The code is also given in the following:

#### **A driver file for FMMYUK-U**

```
IMPLICIT NONE
INTEGER *4 NATOMS,NLEV,IER
PARAMETER (NATOMS=1000000)
REAL *8 BETA
REAL *8 ZAT(3,NATOMS),CHARGE(NATOMS)
REAL *8 POT(NATOMS),FIELD(3,NATOMS)
c
c—set up parameters.
c
  BETA=0.1D0
  NLEV=7
c
c—generate charges and their locations.
c
  CALL DUMMY(NATOMS,ZAT,CHARGE)
c
c—call fmm to calculate the potential and field.
c
  CALL FMMYUK_UNI(BETA,NATOMS,ZAT,CHARGE,POT,FIELD,NLEV,IER)
c
  STOP
END
```



## 4. USING FMMYUK-U

### 4.1. Compile and installation

After you download the package and extract it to your local computer, you will find the following directories on your computer.

- doc: contains the license, readme.txt, and this manual.
- source: the source files and makefile of the package.

For compiling, check the file **Makefile** for further information.

The package has been successfully compiled using the Intel® compiler for Linux, and the GNU®F95 compiler.

The compiled executable is named fmm, you may change it to anything you like.

## 5. PROGRAMMER'S NOTES

### 5.1. Programming Language

Most of the codes are in Fortran 77 style. We also use two commands (allocate and deallocate) from Fortran 90 and later versions for dynamical memory allocations.

### 5.2. Special functions

One function which may be machine dependent is the subroutine for get the current CPU clock information for timing purposes. The users should check their platform and compiler and write such a subroutine. Check second.f for details.

## 6. VARIABLES AND DATA STRUCTURE

### 6.1. Important variables in header file

Two important variables are defined in the header file `parm-yukuni.h`:

- `nterms`: the number of terms in the multipole and local expansions.
- `nlambs`: the number of terms in the exponential expansion.

Currently only 3 digits accuracy and 6 digits accuracy are allowed. We will add more options in later versions.

### 6.2. Important variables in the code

The user should provide the number of particles, their locations and charges. The number of particles is currently set by a parameter *NATOMS*.

A data printing package is provided (see `prini.f`) for outputting integer, real and complex type variables. The output unit is initialized by calling “`prini(num1, num2)`” where `num1` and `num2` are two unit numbers for output. If set to 0, then it will not output those data. See the file `main.f` for further information on calling `prini()`.

### 6.3. Memory Allocation

FMM requires  $O(N)$  memory to store the coefficients of the expansions and precomputed matrices for translation operators. There are two ways to define these variables.

1. The sizes of many variables are determined once *NTERMS* and *NLAMBS* are determined. These variables can be found at the header file `parm-yukuni.h`. Most of these variables are for precomputed matrices.
2. The sizes of the rest of the variables will depend on the number of boxes in the oct-tree structure. For these variables, integer pointers are created and three huge vectors (integer, real and complex) are allocated.

## 7. IMPORTANT SUBROUTINES

**uniyukdriver.f**: the main driver for FMMYUK-U.

**fmm yuk\_uni (fmmuniyuk.f)**: the main subroutine for the FMMYUK-U.

**yuniyuk (fmmuniyuk.f)**: this subroutine computes the far field contribution in the FMM.

**ynbdirect (fmmuniyuk.f)**: this subroutine computes the nearest neighbor direct interaction.

## 8. FREQUENTLY ASKED QUESTIONS

1. **Where can I download this package?** <http://www.fastmultipole.org>
2. ?

Any suggestions, bug reports, please let us know.

## Acknowledgments

We would like to express our gratitude to Prof. Leslie F. Greengard at the Courant Institute of Mathematical Sciences, Professor Vladimir Rokhlin at Yale, and members in their groups. Without their help and encouragements, this package is simply impossible.

We would like to thank our sponsors, DOE, NSF, and CTBP

# BIBLIOGRAPHY

- [1] M. Abramowitz, I. A. Stegun, Handbook of Mathematical Functions, Dover Publications, New York, 1965.
- [2] A. W. Appel, An efficient program for many-body simulations, *SIAM J. Sci. Stat. Comput.* 6 (1985) 85–103.
- [3] J. Barnes, P. Hut, A hierarchical  $O(n \log n)$  force-calculation algorithm, *Nature* 324 (4) (1986) 446 – 449.
- [4] A. H. Boschitsch, M. O. Fenley, W. K. Olson, A fast adaptive multipole algorithm for calculating screened Coulomb (Yukawa) interactions, *J. Comput. Phys.* 151 (1) (1999) 212–241.
- [5] T. Darden, D. York, L. Pedersen, Particle mesh ewald: An  $n \log(n)$  method for Ewald sums in large systems, *J. Chem. Phys.* 98 (12) (1993) 10089–10092.
- [6] P. Debye, E. Huckel, Zur theorie der elektrolyte, *Phys. Zeitschr.* 24 (1923) 185–206.
- [7] M. K. Gilson, A. Rashin, R. Fine, B. Honig, On the calculation of electrostatic interactions in proteins, *J. Mol. Biol.* 184 (3) (1985) 503–516.
- [8] L. Greengard, V. Rokhlin, A fast algorithm for particle simulations, *J. Comput. Phys.* 73 (2) (1987) 325–348.
- [9] L. Greengard, V. Rokhlin, A new version of the fast multipole method for the laplace equation in three dimensions, *Acta Numerica* 6 (1997) 229–269.
- [10] L. F. Greengard, J. F. Huang, A new version of the fast multipole method for screened coulomb interactions in three dimensions, *J. Comput. Phys.* 180 (2) (2002) 642–658.
- [11] S. Kapur, D. E. Long, IES3: Efficient electrostatic and electromagnetic simulation, *IEEE Comput. Sci. Eng.* 5 (4) (1998) 60–67.
- [12] S. S. Kuo, M. D. Altman, J. P. Bardhan, B. Tidor, J. K. White, Fast methods for simulation of biomolecule electrostatics, in: *ICCAD '02: Proceedings of the 2002 IEEE/ACM international conference on Computer-aided design*, ACM Press, New York, NY, USA, 2002.
- [13] B. Z. Lu, X. L. Cheng, J. F. Huang, J. A. McCammon, Order  $N$  algorithm for computation of electrostatic interactions in biomolecular systems, *Proc. Natl. Acad. Sci. U. S. A.* 103 (51) (2006) 19314–19319.

- [14] E. T. Ong, K. H. Lee, K. M. Lim, A fast algorithm for three-dimensional electrostatics analysis: fast fourier transform on multipoles (FFTM), *Int. J. Numer. Methods Eng.* 61 (5) (2004) 633–656.
- [15] E. T. Ong, K. M. Lim, K. H. Lee, H. P. Lee, A fast algorithm for three-dimensional potential fields calculation: fast Fourier transform on multipoles, *J. Comput. Phys.* 192 (1) (2003) 244–261.
- [16] J. R. Phillips, J. K. White, A precorrected-FFT method for electrostatic analysis of complicated 3-D structures, *IEEE Trans. Comput-Aided Des. Integr. Circuits Syst.* 16 (10) (1997) 1059–1072.