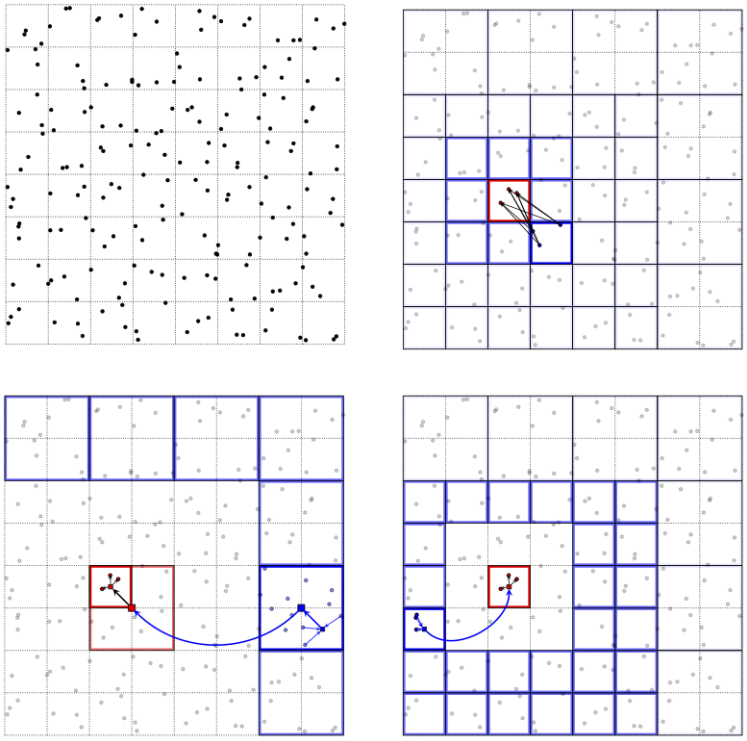
fmm2d Documentation

Release 1.0.0

Zydrunas Gimbutas Leslie Greengard Mike O'Neil
Manas Rachh Vladimir Rokhlin

Oct 24, 2023

CONTENTS



fmm2d is a set of libraries to compute N-body interactions governed by the Laplace and Helmholtz equations, to a specified precision, in three dimensions, on a multi-core shared-memory machine. The library is written in Fortran, wrappers will soon be available for C, MATLAB, and Python. As an example, given M arbitrary points $y_j \in \mathbb{R}^2$ with corresponding real numbers c_j, and N arbitrary points $x_j \in \mathbb{R}^2$, the Laplace FMM evaluates the N real numbers

$$u_\ell = \sum_{j=1}^{M} c_j \log \|x_\ell - y_j\| \,, \qquad \text{for } \ell = 1, 2, \ldots N \,. \tag{1}$$

The y_j can be interpreted as source locations, c_j as charge strengths, and u_ℓ as the resulting potential at target location x_ℓ.

Such N-body interactions are needed in many applications in science and engineering, including molecular dynamics, astrophysics, rheology, and the numerical solution of partial differential equations. The naive CPU effort to evaluate (1) is $O(NM)$. The FMM approximates (1) to a requested relative precision ϵ with linear effort $O((M+N)\log(1/\epsilon))$.

The FMM relies on compressing the interactions between well-separated clusters of source and target points at a hierarchy of scales using analytic outgoing, incoming, and plane-wave expansions of the interaction kernel and associated translation operators. This library is an improved version of the FMMLIB2D software, Copyright (C) 2010-2012: Leslie Greengard and Zydrunas Gimbutas, released under the BSD license. The major improvements are the following:

- Vectorization of the FMM, to apply the same kernel with the same source and target locations to multiple strength vectors
- A redesign of the adaptive tree data structure
- Diagonal form translation operators for high frequency Helmholtz problems

Note: For very small repeated problems (less than 1000 input and output points), users should also consider dense matrix-matrix multiplication using BLAS3 (eg DGEMM,ZGEMM).

CHAPTER

ONE

INSTALLATION

1.1 Obtaining fmm2d

The source code can be downloaded from https://github.com/flatironinstitute/fmm2d

1.2 Dependencies

This library is supported for unix/linux, Mac OSX, and Windows.

For the basic libraries

- Fortran compiler, such as `gfortran` packaged with GCC
- GNU make

1.3 Quick install instructions

Make sure you have dependencies installed, and *cd* into your fmm2d directory.

- For linux, run `make install`.
- For Mac OSX, run `cp make.inc.macos.gnu make.inc` followed by `make install`.
- For Windows, run `cp make.inc.windows.mingw make.inc` followed by `make install`

This should compile the static library in `lib-static/`, the dynamic library in `lib/` and copy the dynamic library to `$(HOME)/lib` on Linux, to `/usr/local/lib` on Mac OSX, and to `C:\lib` on Windows. The location of the default installation directory can be changed by running:

```
make install PREFIX=(INSTALL_DIR)
```

In order to link against the dynamic library, you will have to update the `PATH` environment variable on Windows, `LD_LIBRARY_PATH` environment variable on Linux and `DYLD_LIBRARY_PATH` environment variable on Mac OSX to the installation directory. You may then link to the FMM library using the `-lfmm2d` option.

Note: On MacOSX, /usr/local/lib is included by default in the DYLD_LIBRARY_PATH.

To verify successful compilation of the program, run `make test` which compiles some fortran test drivers in `test/` linked against the static library, after which it runs the test programs. The last 4 lines of the terminal output should be:

```
cat print_testreshelm.txt
Successfully completed 27 out of 27 tests in hfmm2d testing suite
Successfully completed 27 out of 27 tests in hfmm2d vec testing suite
rm print_testreshelm.txt
```

To verify successful installation of the program, and the correct setting for environment variables, run `make test-dyn` which compiles some fortran test drivers in `test/` linked against the dynamic library, after which it runs teh test prgram. The output ofshould be the same as above.

If `make test` fails, see more detailed instructions below.

If `make test-dyn` fails with an error about not finding `-llibfmm2d_dll` or `-lfmm2d` or `libfmm2d` make sure that the appropriate environment variables have been set. If it fails with other issues, see more detailed instructions below.

Type `make` to see a list of other build options (language interfaces, etc). Please see Fortran interfaces and look in `examples/` for sample drivers.

If there is an error in testing on a standard set-up, please file a bug report as a New Issue at https://github.com/flatironinstitute/fmm2d/issues

1.3.1 Custom library compilation options

In the (default) easy-to-install version, the library is compiled without using the optimized direct evaluation kernels.

In order to disable multi-threading, append `OMP=OFF` to the make task.

All of these different libraries are built with the same name, so you will have to move them to other locations, or build a 2nd copy of the repo, if you want to keep both versions.

You *must* do at least `make objclean` before changing to the openmp /fast direct kernel evaluation options.

1.3.2 Examples

- `make examples` to compile and run the examples for calling from Fortran.

The `examples` directory is a good place to see usage examples for Fortran. There are two sample Fortran drivers for Helmholtz FMMs, one which demonstrates the use of FMMs, and one which demonstrates the use of vectorized FMMs.

The sample Helmholtz drivers are `hfmm2d_example.f`, `hfmm2d_vec_example.f`, and `hfmm2d_legacy_example.f`. The corresponding makefiles are `hfmm2d_example.make`, `hfmm2d_vec_example.make`, and `hfmm2d_legacy_example.make`.

1.4 Tips for installing dependencies

1.4.1 On Ubuntu linux

On Ubuntu linux (assuming python3 as opposed to python):

```
sudo apt-get install make build-essential gfortran
```

1.4.2 On Fedora/CentOS linux

On a Fedora/CentOS linux system, these dependencies can be installed as follows:

```
sudo yum install make gcc gcc-c++ gcc-gfortran libgomp
```

1.4.3 On Mac OSX

First setup Homebrew as follows. If you don't have Xcode, install Command Line Tools by opening a terminal (from /Applications/Utilities/) and typing:

```
xcode-select --install
```

Then install Homebrew by pasting the installation command from https://brew.sh

Then do:

```
brew install gcc
```

1.4.4 On Windows

Download 64 bit mingw (available here). Follow the install instructions and append to the environment variable `PATH` the location of the bin directory of your mingw installation.

Download and install `make` for windows (Available here).

Download and install `git` for windows (Available here).

CHAPTER

TWO

DEFINITIONS

Let $x_j \in \mathbb{R}^2$, $i = 1, 2, \ldots N$, denote a collection of source locations and let $t_i \in \mathbb{R}^2$ denote a collection of target locations.

2.1 Helmholtz FMM

Let $c_j \in \mathbb{C}$, $j = 1, 2, \ldots N$, denote a collection of charge strengths, $v_j \in \mathbb{C}$, $j = 1, 2, \ldots N$, denote a collection of dipole strengths, and $d_j \in \mathbb{R}^2$, $j = 1, 2, \ldots N$, denote the corresponding dipole orientation vectors. Let $k \in \mathbb{C}$ denote the wave number or the Helmholtz parameter.

The Helmholtz FMM computes the potential $u(x)$ and the its gradient $\nabla u(x)$ given by

$$u(x) = \sum_{j=1}^{N} c_j H_0^{(1)}(k\|x - x_j\|) - v_j d_j \cdot \nabla H_0^{(1)}(k\|x - x_j\|)\,, \tag{2.1}$$

at the source and target locations, where $H_0^{(1)}$ is the Hankel function of the first kind of order 0. When $x = x_j$, the term corresponding to x_j is dropped from the sum.

2.2 Vectorized versions

The vectorized versions of the Helmholtz FMM, computes repeated FMMs for new charge and dipole strengths located at the same source locations, where the potential and its gradient are evaluated at the same set of target locations.

For example, let $c_{\ell,j} \in \mathbb{C}$, $j = 1, 2, \ldots N$, $\ell = 1, 2, \ldots n_d$ denote a collection of n_d charge strengths, and let $v_{\ell,j} \in \mathbb{C}$, $d_{\ell,j} \in \mathbb{R}^2$ denote a collection of n_d dipole strengths and orientation vectors. Then the vectorized Helmholtz FMM computes the potentials $u_\ell(x)$ and its gradients $\nabla u_\ell(x)$ defined by the formula

$$u_\ell(x) = \sum_{j=1}^{N} c_{\ell,j} H_0^{(1)}(k\|x - x_j\|) - v_{\ell,j} d_j \cdot \nabla H_0^{(1)}(k\|x - x_j\|)\,, \tag{2.2}$$

at the source and target locations.

Note: In double precision arithmetic, two numbers which are within machine precision of each other cannot be distinguished. In order to account for this, suppose that the sources and targets are contained in a cube with side length L, then for all x such that $\|x - x_j\| \leq L\varepsilon_{\text{mach}}$, the term corresponding to x_j is dropped from the sum. Here $\varepsilon_{\text{mach}} = 2^{-52}$ is machine precision.

CHAPTER

THREE

FORTRAN INTERFACES

3.1 Helmholtz FMM

The Helmholtz FMM evaluates the following potential, its gradient and its hessian

$$u(x) = \sum_{j=1}^{N} c_j H_0^{(1)}(\|x - x_j\|) - v_j d_j \cdot \nabla H_0^{(1)}(\|x - x_j\|) \,.$$

Here x_j are the source locations, c_j are the charge strengths, v_j are the dipole strengths, and d_j are the dipole orientation vectors. The collection of x at which the potential and its gradient are evaluated are referred to as the evalution points.

There are 27 different Fortran wrappers for the Helmholtz FMM to account for collection of evaluation points (sources only, targets only, sources+targets), interaction kernel (charges only, dipoles only, charges + dipoles), output request (potential, potential+gradient,potential+gradient+hessian).

For example, the subroutine to evaluate the potential and gradient, at a collection of targets t_i due to a collection of charges is:

```
hfmm2d_t_c_g
```

In general, the subroutine names take the following form:

```
hfmm2d_<eval-pts>_<int-ker>_<out>
```

- <eval-pts>: evaluation points. Collection of x where u and its gradient is to be evaluated
  - s: Evaluate u and its gradient at the source locations x_i
  - t: Evaluate u and its gradient at t_i, a collection of target locations specified by the user.
  - st: Evaluate u and its gradient at both source and target locations x_i and t_i.
- <int-ker>: kernel of interaction (type of sources present)
  - c: charges
  - d: dipoles
  - cd: charges + dipoles
- <out>: flag for evaluating potential or potential + gradient
  - p: on output only u is evaluated
  - g: on output both u and its gradient are evaluated

  - h: on output u, its gradient and its hessian are evaluated

These are all the single density routines. To get a vectorized version of any of the routines use:

```
<subroutine name>_vec
```

Note: For the vectorized subroutines, the charge strengths, dipole strengths, potentials, and gradients are interleaved as opposed to provided in a sequential manner. For example for three sets of charge strengths, they should be stored as $c_{1,1}, c_{2,1}, c_{3,1}, c_{1,2}, c_{2,2}, c_{3,2} \ldots c_{1,N}, c_{2,N}, c_{3,N}$.

Example drivers:

- `examples/hfmm2d_example.f`. The corresponding makefile is `examples/hfmm2d_example.make`
- `examples/hfmm2d_vec_example.f`. The corresponding makefile is `examples/hfmm2d_vec_example.make`

Back to top

3.1.1 List of interfaces

- Evaluation points: Sources
  - Interaction Type: Charges
    - Potential (*hfmm2d_s_c_p*)
    - Gradient (*hfmm2d_s_c_g*)
    - Hessian (*hfmm2d_s_c_h*)
  - Interaction Type: Dipoles
    - Potential (*hfmm2d_s_d_p*)
    - Gradient (*hfmm2d_s_d_g*)
    - Hessian (*hfmm2d_s_d_h*)
  - Interaction Type: Charges + Dipoles
    - Potential (*hfmm2d_s_cd_p*)
    - Gradient (*hfmm2d_s_cd_g*)
    - Hessian (*hfmm2d_s_cd_h*)
- Evaluation points: Targets
  - Interaction Type: Charges
    - Potential (*hfmm2d_t_c_p*)
    - Gradient (*hfmm2d_t_c_g*)
    - Hessian (*hfmm2d_t_c_h*)
  - Interaction Type: Dipoles
    - Potential (*hfmm2d_t_d_p*)
    - Gradient (*hfmm2d_t_d_g*)
    - Hessian (*hfmm2d_t_d_h*)

  - Interaction Type: Charges + Dipoles
    - Potential (*hfmm2d_t_cd_p*)
    - Gradient (*hfmm2d_t_cd_g*)
    - Hessian (*hfmm2d_t_cd_h*)
- Evaluation points: Sources + Targets
  - Interaction Type: Charges
    - Potential (*hfmm2d_st_c_p*)
    - Gradient (*hfmm2d_st_c_g*)
    - Hessian (*hfmm2d_st_c_h*)
  - Interaction Type: Dipoles
    - Potential (*hfmm2d_st_d_p*)
    - Gradient (*hfmm2d_st_d_g*)
    - Hessian (*hfmm2d_st_d_h*)
  - Interaction Type: Charges + Dipoles
    - Potential (*hfmm2d_st_cd_p*)
    - Gradient (*hfmm2d_st_cd_g*)
    - Hessian (*hfmm2d_st_cd_h*)

Back to top

hfmm2d_s_c_p

- Evaluation points: Sources
- Interaction kernel: Charges
- Outputs requested: Potential

```
subroutine hfmm2d_s_c_p(eps,zk,nsource,source,charge,pot,ier)
```

This subroutine evaluates the potential

$$u(x) = \sum_{j=1}^{N} c_j H_0^{(1)}(k\|x - x_j\|)$$

at the source locations $x = x_j$. When $x = x_j$, the term corresponding to x_j is dropped from the sum.

Input arguments:

- **eps: double precision** precision requested
- **zk: double complex** Helmholtz parameter, k
- **nsource: integer** Number of sources
- **source: double precision(3,nsource)** Source locations, x_j

- **charge: double complex(nsource)** Charge strengths, c_j

Output arguments:

- **pot: double complex(nsource)** Potential at source locations, $u(x_j)$
- **ier: integer** Error flag; ier=0 implies successful execution, and ier=4/8 implies insufficient memory

---

Vectorized version:

```
subroutine hfmm2d_s_c_p_vec(nd,eps,zk,nsource,source,charge,pot,ier)
```

This subroutine evaluates the potential

$$u_\ell(x) = \sum_{j=1}^{N} c_{\ell,j} H_0^{(1)}(k\|x - x_j\|)$$

at the source locations $x = x_j$. When $x = x_j$, the term corresponding to x_j is dropped from the sum.

Input arguments:

- **nd: integer** number of densities
- **charge: double complex(nd,nsource)** Charge strengths, $c_{\ell,j}$

Output arguments:

- **pot: double complex(nd,nsource)** Potential at source locations, $u_\ell(x_j)$
- **ier: integer** Error flag; ier=0 implies successful execution, and ier=4/8 implies insufficient memory

Back to Helmholtz FMM

Back to top

hfmm2d_s_c_g

- Evaluation points: Sources
- Interaction kernel: Charges
- Outputs requested: Potential and Gradient

```
subroutine hfmm2d_s_c_g(eps,zk,nsource,source,charge,pot,grad,ier)
```

This subroutine evaluates the potential and its gradient

$$u(x) = \sum_{j=1}^{N} c_j H_0^{(1)}(k\|x - x_j\|)$$

at the source locations $x = x_j$. When $x = x_j$, the term corresponding to x_j is dropped from the sum.

Input arguments:

- **eps: double precision** precision requested
- **zk: double complex** Helmholtz parameter, k
- **nsource: integer** Number of sources
- **source: double precision(3,nsource)** Source locations, x_j
- **charge: double complex(nsource)** Charge strengths, c_j

Output arguments:

- **pot: double complex(nsource)** Potential at source locations, $u(x_j)$
- **grad: double complex(2,nsource)** Gradient at source locations, $\nabla u(x_j)$
- **ier: integer** Error flag; ier=0 implies successful execution, and ier=4/8 implies insufficient memory

---

Vectorized version:

```
subroutine hfmm2d_s_c_g_vec(nd,eps,zk,nsource,source,charge,pot,grad,ier)
```

This subroutine evaluates the potential and its gradient

$$u_\ell(x) = \sum_{j=1}^{N} c_{\ell,j} H_0^{(1)}(k\|x - x_j\|)$$

at the source locations $x = x_j$. When $x = x_j$, the term corresponding to x_j is dropped from the sum.

Input arguments:

- **nd: integer** number of densities
- **charge: double complex(nd,nsource)** Charge strengths, $c_{\ell,j}$

Output arguments:

- **pot: double complex(nd,nsource)** Potential at source locations, $u_\ell(x_j)$
- **grad: double complex(nd,2,nsource)** Gradient at source locations, $\nabla u_\ell(x_j)$
- **ier: integer** Error flag; ier=0 implies successful execution, and ier=4/8 implies insufficient memory

Back to Helmholtz FMM

Back to top

hfmm2d_s_c_h

- Evaluation points: Sources
- Interaction kernel: Charges
- Outputs requested: Potential, Gradient and Hessian

```
subroutine hfmm2d_s_c_h(eps,zk,nsource,source,charge,pot,grad,hess,ier)
```

This subroutine evaluates the potential, its gradient, and its hessian

$$u(x) = \sum_{j=1}^{N} c_j H_0^{(1)}(k\|x - x_j\|)$$

at the source locations $x = x_j$. When $x = x_j$, the term corresponding to x_j is dropped from the sum.

Input arguments:

- **eps: double precision** precision requested
- **zk: double complex** Helmholtz parameter, k
- **nsource: integer** Number of sources
- **source: double precision(3,nsource)** Source locations, x_j
- **charge: double complex(nsource)** Charge strengths, c_j

Output arguments:

- **pot: double complex(nsource)** Potential at source locations, $u(x_j)$
- **grad: double complex(2,nsource)** Gradient at source locations, $\nabla u(x_j)$
- **hess: double complex(3,nsource)** Hessian at source locations, $\nabla\nabla u(x_j)$
- **ier: integer** Error flag; ier=0 implies successful execution, and ier=4/8 implies insufficient memory

---

Vectorized version:

```
subroutine hfmm2d_s_c_h_vec(nd,eps,zk,nsource,source,charge,pot,grad,hess,ier)
```

This subroutine evaluates the potential, its gradient, and its hessian

$$u_\ell(x) = \sum_{j=1}^{N} c_{\ell,j} H_0^{(1)}(k\|x - x_j\|)$$

at the source locations $x = x_j$. When $x = x_j$, the term corresponding to x_j is dropped from the sum.

Input arguments:

- **nd: integer** number of densities
- **charge: double complex(nd,nsource)** Charge strengths, $c_{\ell,j}$

Output arguments:

- **pot: double complex(nd,nsource)** Potential at source locations, $u_\ell(x_j)$
- **grad: double complex(nd,2,nsource)** Gradient at source locations, $\nabla u_\ell(x_j)$
- **hess: double complex(nd,3,nsource)** Hessian at source locations, $\nabla\nabla u_\ell(x_j)$
- **ier: integer** Error flag; ier=0 implies successful execution, and ier=4/8 implies insufficient memory

Back to Helmholtz FMM

Back to top

hfmm2d_s_d_p

- Evaluation points: Sources
- Interaction kernel: Dipoles
- Outputs requested: Potential

```
subroutine hfmm2d_s_d_p(eps,zk,nsource,source,dipstr,dipvec,pot,ier)
```

This subroutine evaluates the potential

$$u(x) = -\sum_{j=1}^{N} v_j d_j \cdot \nabla \left(H_0^{(1)}(k\|x - x_j\|) \right)$$

at the source locations $x = x_j$. When $x = x_j$, the term corresponding to x_j is dropped from the sum.

Input arguments:

- **eps: double precision** precision requested
- **zk: double complex** Helmholtz parameter, k
- **nsource: integer** Number of sources
- **source: double precision(3,nsource)** Source locations, x_j
- **dipstr: double complex(nsource)** Dipole strengths, v_j
- **dipvec: double precision(2,nsource)** Dipole orientation vectors, d_j

Output arguments:

- **pot: double complex(nsource)** Potential at source locations, $u(x_j)$
- **ier: integer** Error flag; ier=0 implies successful execution, and ier=4/8 implies insufficient memory

---

Vectorized version:

```
subroutine hfmm2d_s_d_p_vec(nd,eps,zk,nsource,source,dipstr,dipvec,pot,ier)
```

This subroutine evaluates the potential

$$u_\ell(x) = -\sum_{j=1}^{N} v_{\ell,j} d_{\ell,j} \cdot \nabla \left(H_0^{(1)}(k\|x - x_j\|) \right)$$

at the source locations $x = x_j$. When $x = x_j$, the term corresponding to x_j is dropped from the sum.

Input arguments:

- **nd: integer** number of densities
- **dipstr: double complex(nd,nsource)** Dipole strengths, $v_{\ell,j}$
- **dipvec: double precision(nd,2,nsource)** Dipole orientation vectors, $d_{\ell,j}$

Output arguments:

- **pot: double complex(nd,nsource)** Potential at source locations, $u_\ell(x_j)$
- **ier: integer** Error flag; ier=0 implies successful execution, and ier=4/8 implies insufficient memory

Back to Helmholtz FMM

Back to top

hfmm2d_s_d_g

- Evaluation points: Sources
- Interaction kernel: Dipoles
- Outputs requested: Potential and Gradient

```
subroutine hfmm2d_s_d_g(eps,zk,nsource,source,dipstr,dipvec,pot,grad,ier)
```

This subroutine evaluates the potential and its gradient

$$u(x) = -\sum_{j=1}^{N} v_j d_j \cdot \nabla \left(H_0^{(1)}(k\|x - x_j\|) \right)$$

at the source locations $x = x_j$. When $x = x_j$, the term corresponding to x_j is dropped from the sum.

Input arguments:

- **eps: double precision** precision requested
- **zk: double complex** Helmholtz parameter, k
- **nsource: integer** Number of sources
- **source: double precision(3,nsource)** Source locations, x_j
- **dipstr: double complex(nsource)** Dipole strengths, v_j
- **dipvec: double precision(2,nsource)** Dipole orientation vectors, d_j

Output arguments:

- **pot: double complex(nsource)** Potential at source locations, $u(x_j)$
- **grad: double complex(2,nsource)** Gradient at source locations, $\nabla u(x_j)$
- **ier: integer** Error flag; ier=0 implies successful execution, and ier=4/8 implies insufficient memory

---

Vectorized version:

```
subroutine hfmm2d_s_d_g_vec(nd,eps,zk,nsource,source,dipstr,dipvec,pot,grad,ier)
```

This subroutine evaluates the potential and its gradient

$$u_\ell(x) = -\sum_{j=1}^{N} v_{\ell,j} d_{\ell,j} \cdot \nabla \left(H_0^{(1)}(k\|x - x_j\|) \right)$$

at the source locations $x = x_j$. When $x = x_j$, the term corresponding to x_j is dropped from the sum.

Input arguments:

- **nd: integer** number of densities
- **dipstr: double complex(nd,nsource)** Dipole strengths, $v_{\ell,j}$
- **dipvec: double precision(nd,2,nsource)** Dipole orientation vectors, $d_{\ell,j}$

Output arguments:

- **pot: double complex(nd,nsource)** Potential at source locations, $u_\ell(x_j)$
- **grad: double complex(nd,2,nsource)** Gradient at source locations, $\nabla u_\ell(x_j)$
- **ier: integer** Error flag; ier=0 implies successful execution, and ier=4/8 implies insufficient memory

Back to Helmholtz FMM

Back to top

hfmm2d_s_d_h

- Evaluation points: Sources
- Interaction kernel: Dipoles
- Outputs requested: Potential, Gradient and Hessian

```
subroutine hfmm2d_s_d_h(eps,zk,nsource,source,dipstr,dipvec,pot,grad,hess,ier)
```

This subroutine evaluates the potential, its gradient, and its hessian

$$u(x) = -\sum_{j=1}^{N} v_j d_j \cdot \nabla \left(H_0^{(1)}(k\|x - x_j\|) \right)$$

at the source locations $x = x_j$. When $x = x_j$, the term corresponding to x_j is dropped from the sum.

Input arguments:

- **eps: double precision** precision requested
- **zk: double complex** Helmholtz parameter, k
- **nsource: integer** Number of sources
- **source: double precision(3,nsource)** Source locations, x_j
- **dipstr: double complex(nsource)** Dipole strengths, v_j
- **dipvec: double precision(2,nsource)** Dipole orientation vectors, d_j

Output arguments:

- **pot: double complex(nsource)** Potential at source locations, $u(x_j)$
- **grad: double complex(2,nsource)** Gradient at source locations, $\nabla u(x_j)$
- **hess: double complex(3,nsource)** Hessian at source locations, $\nabla\nabla u(x_j)$
- **ier: integer** Error flag; ier=0 implies successful execution, and ier=4/8 implies insufficient memory

Vectorized version:

```
subroutine hfmm2d_s_d_h_vec(nd,eps,zk,nsource,source,dipstr,dipvec,pot,grad,hess,ier)
```

This subroutine evaluates the potential, its gradient, and its hessian

$$u_\ell(x) = -\sum_{j=1}^{N} v_{\ell,j} d_{\ell,j} \cdot \nabla \left(H_0^{(1)}(k\|x - x_j\|) \right)$$

at the source locations $x = x_j$. When $x = x_j$, the term corresponding to x_j is dropped from the sum.

Input arguments:

- **nd: integer** number of densities
- **dipstr: double complex(nd,nsource)** Dipole strengths, $v_{\ell,j}$
- **dipvec: double precision(nd,2,nsource)** Dipole orientation vectors, $d_{\ell,j}$

Output arguments:

- **pot: double complex(nd,nsource)** Potential at source locations, $u_\ell(x_j)$
- **grad: double complex(nd,2,nsource)** Gradient at source locations, $\nabla u_\ell(x_j)$
- **hess: double complex(nd,3,nsource)** Hessian at source locations, $\nabla\nabla u_\ell(x_j)$
- **ier: integer** Error flag; ier=0 implies successful execution, and ier=4/8 implies insufficient memory

Back to Helmholtz FMM

Back to top

hfmm2d_s_cd_p

- Evaluation points: Sources
- Interaction kernel: Charges and Dipoles
- Outputs requested: Potential

```
subroutine hfmm2d_s_cd_p(eps,zk,nsource,source,charge,dipstr,dipvec,pot,ier)
```

This subroutine evaluates the potential

$$u(x) = \sum_{j=1}^{N} c_j H_0^{(1)}(k\|x - x_j\|) - v_j d_j \cdot \nabla \left(H_0^{(1)}(k\|x - x_j\|) \right)$$

at the source locations $x = x_j$. When $x = x_j$, the term corresponding to x_j is dropped from the sum.

Input arguments:

- **eps: double precision** precision requested

- **zk: double complex** Helmholtz parameter, k
- **nsource: integer** Number of sources
- **source: double precision(3,nsource)** Source locations, x_j
- **charge: double complex(nsource)** Charge strengths, c_j
- **dipstr: double complex(nsource)** Dipole strengths, v_j
- **dipvec: double precision(2,nsource)** Dipole orientation vectors, d_j

Output arguments:

- **pot: double complex(nsource)** Potential at source locations, $u(x_j)$
- **ier: integer** Error flag; ier=0 implies successful execution, and ier=4/8 implies insufficient memory

---

Vectorized version:

```
subroutine hfmm2d_s_cd_p_vec(nd,eps,zk,nsource,source,charge,dipstr,dipvec,pot,ier)
```

This subroutine evaluates the potential

$$u_\ell(x) = \sum_{j=1}^{N} c_{\ell,j} H_0^{(1)}(k\|x - x_j\|) - v_{\ell,j} d_{\ell,j} \cdot \nabla \left(H_0^{(1)}(k\|x - x_j\|) \right)$$

at the source locations $x = x_j$. When $x = x_j$, the term corresponding to x_j is dropped from the sum.

Input arguments:

- **nd: integer** number of densities
- **charge: double complex(nd,nsource)** Charge strengths, $c_{\ell,j}$
- **dipstr: double complex(nd,nsource)** Dipole strengths, $v_{\ell,j}$
- **dipvec: double precision(nd,2,nsource)** Dipole orientation vectors, $d_{\ell,j}$

Output arguments:

- **pot: double complex(nd,nsource)** Potential at source locations, $u_\ell(x_j)$
- **ier: integer** Error flag; ier=0 implies successful execution, and ier=4/8 implies insufficient memory

Back to Helmholtz FMM

Back to top

hfmm2d_s_cd_g

- Evaluation points: Sources
- Interaction kernel: Charges and Dipoles
- Outputs requested: Potential and Gradient

```
subroutine hfmm2d_s_cd_g(eps,zk,nsource,source,charge,dipstr,dipvec,pot,grad,ier)
```

This subroutine evaluates the potential and its gradient

$$u(x) = \sum_{j=1}^{N} c_j H_0^{(1)}(k\|x - x_j\|) - v_j d_j \cdot \nabla \left(H_0^{(1)}(k\|x - x_j\|) \right)$$

at the source locations $x = x_j$. When $x = x_j$, the term corresponding to x_j is dropped from the sum.

Input arguments:

- **eps: double precision** precision requested
- **zk: double complex** Helmholtz parameter, k
- **nsource: integer** Number of sources
- **source: double precision(3,nsource)** Source locations, x_j
- **charge: double complex(nsource)** Charge strengths, c_j
- **dipstr: double complex(nsource)** Dipole strengths, v_j
- **dipvec: double precision(2,nsource)** Dipole orientation vectors, d_j

Output arguments:

- **pot: double complex(nsource)** Potential at source locations, $u(x_j)$
- **grad: double complex(2,nsource)** Gradient at source locations, $\nabla u(x_j)$
- **ier: integer** Error flag; ier=0 implies successful execution, and ier=4/8 implies insufficient memory

---

Vectorized version:

```
subroutine hfmm2d_s_cd_g_vec(nd,eps,zk,nsource,source,charge,dipstr,dipvec,pot,grad,ier)
```

This subroutine evaluates the potential and its gradient

$$u_\ell(x) = \sum_{j=1}^{N} c_{\ell,j} H_0^{(1)}(k\|x - x_j\|) - v_{\ell,j} d_{\ell,j} \cdot \nabla \left(H_0^{(1)}(k\|x - x_j\|) \right)$$

at the source locations $x = x_j$. When $x = x_j$, the term corresponding to x_j is dropped from the sum.

Input arguments:

- **nd: integer** number of densities
- **charge: double complex(nd,nsource)** Charge strengths, $c_{\ell,j}$
- **dipstr: double complex(nd,nsource)** Dipole strengths, $v_{\ell,j}$
- **dipvec: double precision(nd,2,nsource)** Dipole orientation vectors, $d_{\ell,j}$

Output arguments:

- **pot: double complex(nd,nsource)** Potential at source locations, $u_\ell(x_j)$
- **grad: double complex(nd,2,nsource)** Gradient at source locations, $\nabla u_\ell(x_j)$
- **ier: integer** Error flag; ier=0 implies successful execution, and ier=4/8 implies insufficient memory

Back to Helmholtz FMM

Back to top

hfmm2d_s_cd_h

- Evaluation points: Sources
- Interaction kernel: Charges and Dipoles
- Outputs requested: Potential, Gradient and Hessian

```
subroutine hfmm2d_s_cd_h(eps,zk,nsource,source,charge,dipstr,dipvec,pot,grad,hess,ier)
```

This subroutine evaluates the potential, its gradient, and its hessian

$$u(x) = \sum_{j=1}^{N} c_j H_0^{(1)}(k\|x - x_j\|) - v_j d_j \cdot \nabla \left(H_0^{(1)}(k\|x - x_j\|) \right)$$

at the source locations $x = x_j$. When $x = x_j$, the term corresponding to x_j is dropped from the sum.

Input arguments:

- **eps: double precision** precision requested
- **zk: double complex** Helmholtz parameter, k
- **nsource: integer** Number of sources
- **source: double precision(3,nsource)** Source locations, x_j
- **charge: double complex(nsource)** Charge strengths, c_j
- **dipstr: double complex(nsource)** Dipole strengths, v_j
- **dipvec: double precision(2,nsource)** Dipole orientation vectors, d_j

Output arguments:

- **pot: double complex(nsource)** Potential at source locations, $u(x_j)$
- **grad: double complex(2,nsource)** Gradient at source locations, $\nabla u(x_j)$
- **hess: double complex(3,nsource)** Hessian at source locations, $\nabla\nabla u(x_j)$
- **ier: integer** Error flag; ier=0 implies successful execution, and ier=4/8 implies insufficient memory

---

Vectorized version:

```
subroutine hfmm2d_s_cd_h_vec(nd,eps,zk,nsource,source,charge,dipstr,dipvec,pot,grad,hess,
↪ier)
```

This subroutine evaluates the potential, its gradient, and its hessian

$$u_\ell(x) = \sum_{j=1}^{N} c_{\ell,j} H_0^{(1)}(k\|x - x_j\|) - v_{\ell,j} d_{\ell,j} \cdot \nabla \left(H_0^{(1)}(k\|x - x_j\|) \right)$$

at the source locations $x = x_j$. When $x = x_j$, the term corresponding to x_j is dropped from the sum.

Input arguments:

- **nd: integer** number of densities
- **charge: double complex(nd,nsource)** Charge strengths, $c_{\ell,j}$
- **dipstr: double complex(nd,nsource)** Dipole strengths, $v_{\ell,j}$
- **dipvec: double precision(nd,2,nsource)** Dipole orientation vectors, $d_{\ell,j}$

Output arguments:

- **pot: double complex(nd,nsource)** Potential at source locations, $u_\ell(x_j)$
- **grad: double complex(nd,2,nsource)** Gradient at source locations, $\nabla u_\ell(x_j)$
- **hess: double complex(nd,3,nsource)** Hessian at source locations, $\nabla\nabla u_\ell(x_j)$
- **ier: integer** Error flag; ier=0 implies successful execution, and ier=4/8 implies insufficient memory

Back to Helmholtz FMM

Back to top

hfmm2d_t_c_p

- Evaluation points: Targets
- Interaction kernel: Charges
- Outputs requested: Potential

```
subroutine hfmm2d_t_c_p(eps,zk,nsource,source,charge,ntarg,targ,pottarg,ier)
```

This subroutine evaluates the potential

$$u(x) = \sum_{j=1}^{N} c_j H_0^{(1)}(k\|x - x_j\|)$$

at the target locations $x = t_i$. When $x = x_j$, the term corresponding to x_j is dropped from the sum.

Input arguments:

- **eps: double precision** precision requested
- **zk: double complex** Helmholtz parameter, k
- **nsource: integer** Number of sources
- **source: double precision(3,nsource)** Source locations, x_j
- **charge: double complex(nsource)** Charge strengths, c_j
- **ntarg: integer** Number of targets
- **targ: double precision(3,ntarg)** Target locations, t_i

Output arguments:

- **pottarg: double complex(ntarg)** Potential at target locations, $u(t_i)$

- **ier: integer** Error flag; ier=0 implies successful execution, and ier=4/8 implies insufficient memory

---

Vectorized version:

```
subroutine hfmm2d_t_c_p_vec(nd,eps,zk,nsource,source,charge,ntarg,targ,pottarg,ier)
```

This subroutine evaluates the potential

$$u_\ell(x) = \sum_{j=1}^{N} c_{\ell,j} H_0^{(1)}(k\|x - x_j\|)$$

at the target locations $x = t_i$. When $x = x_j$, the term corresponding to x_j is dropped from the sum.

Input arguments:

- **nd: integer** number of densities
- **charge: double complex(nd,nsource)** Charge strengths, $c_{\ell,j}$

Output arguments:

- **pottarg: double complex(nd,ntarg)** Potential at target locations, $u_\ell(t_i)$
- **ier: integer** Error flag; ier=0 implies successful execution, and ier=4/8 implies insufficient memory

Back to Helmholtz FMM

Back to top

hfmm2d_t_c_g

- Evaluation points: Targets
- Interaction kernel: Charges
- Outputs requested: Potential and Gradient

```
subroutine hfmm2d_t_c_g(eps,zk,nsource,source,charge,ntarg,targ,pottarg,gradtarg,ier)
```

This subroutine evaluates the potential and its gradient

$$u(x) = \sum_{j=1}^{N} c_j H_0^{(1)}(k\|x - x_j\|)$$

at the target locations $x = t_i$. When $x = x_j$, the term corresponding to x_j is dropped from the sum.

Input arguments:

- **eps: double precision** precision requested
- **zk: double complex** Helmholtz parameter, k
- **nsource: integer** Number of sources

- **source: double precision(3,nsource)** Source locations, x_j
- **charge: double complex(nsource)** Charge strengths, c_j
- **ntarg: integer** Number of targets
- **targ: double precision(3,ntarg)** Target locations, t_i

Output arguments:

- **pottarg: double complex(ntarg)** Potential at target locations, $u(t_i)$
- **gradtarg: double complex(2,ntarg)** Gradient at target locations, $\nabla u(t_i)$
- **ier: integer** Error flag; ier=0 implies successful execution, and ier=4/8 implies insufficient memory

---

Vectorized version:

```
subroutine hfmm2d_t_c_g_vec(nd,eps,zk,nsource,source,charge,ntarg,targ,pottarg,gradtarg,
↪ier)
```

This subroutine evaluates the potential and its gradient

$$u_\ell(x) = \sum_{j=1}^{N} c_{\ell,j} H_0^{(1)}(k\|x - x_j\|)$$

at the target locations $x = t_i$. When $x = x_j$, the term corresponding to x_j is dropped from the sum.

Input arguments:

- **nd: integer** number of densities
- **charge: double complex(nd,nsource)** Charge strengths, $c_{\ell,j}$

Output arguments:

- **pottarg: double complex(nd,ntarg)** Potential at target locations, $u_\ell(t_i)$
- **gradtarg: double complex(nd,2,ntarg)** Gradient at target locations, $\nabla u_\ell(t_i)$
- **ier: integer** Error flag; ier=0 implies successful execution, and ier=4/8 implies insufficient memory

Back to Helmholtz FMM

Back to top

hfmm2d_t_c_h

- Evaluation points: Targets
- Interaction kernel: Charges
- Outputs requested: Potential, Gradient and Hessian

```
subroutine hfmm2d_t_c_h(eps,zk,nsource,source,charge,ntarg,targ,pottarg,gradtarg,
↪hesstarg,ier)
```

This subroutine evaluates the potential, its gradient, and its hessian

$$u(x) = \sum_{j=1}^{N} c_j H_0^{(1)}(k\|x - x_j\|)$$

at the target locations $x = t_i$. When $x = x_j$, the term corresponding to x_j is dropped from the sum.

Input arguments:

- **eps: double precision** precision requested
- **zk: double complex** Helmholtz parameter, k
- **nsource: integer** Number of sources
- **source: double precision(3,nsource)** Source locations, x_j
- **charge: double complex(nsource)** Charge strengths, c_j
- **ntarg: integer** Number of targets
- **targ: double precision(3,ntarg)** Target locations, t_i

Output arguments:

- **pottarg: double complex(ntarg)** Potential at target locations, $u(t_i)$
- **gradtarg: double complex(2,ntarg)** Gradient at target locations, $\nabla u(t_i)$
- **hesstarg: double complex(3,ntarg)** Hessian at target locations, $\nabla\nabla u(t_i)$
- **ier: integer** Error flag; ier=0 implies successful execution, and ier=4/8 implies insufficient memory

---

Vectorized version:

```
subroutine hfmm2d_t_c_h_vec(nd,eps,zk,nsource,source,charge,ntarg,targ,pottarg,gradtarg,
↪hesstarg,ier)
```

This subroutine evaluates the potential, its gradient, and its hessian

$$u_\ell(x) = \sum_{j=1}^{N} c_{\ell,j} H_0^{(1)}(k\|x - x_j\|)$$

at the target locations $x = t_i$. When $x = x_j$, the term corresponding to x_j is dropped from the sum.

Input arguments:

- **nd: integer** number of densities
- **charge: double complex(nd,nsource)** Charge strengths, $c_{\ell,j}$

Output arguments:

- **pottarg: double complex(nd,ntarg)** Potential at target locations, $u_\ell(t_i)$
- **gradtarg: double complex(nd,2,ntarg)** Gradient at target locations, $\nabla u_\ell(t_i)$
- **hesstarg: double complex(nd,3,ntarg)** Hessian at target locations, $\nabla u_\ell(t_i)$
- **ier: integer** Error flag; ier=0 implies successful execution, and ier=4/8 implies insufficient memory

Back to Helmholtz FMM

Back to top

hfmm2d_t_d_p

- Evaluation points: Targets
- Interaction kernel: Dipoles
- Outputs requested: Potential

```
subroutine hfmm2d_t_d_p(eps,zk,nsource,source,dipstr,dipvec,ntarg,targ,pottarg,ier)
```

This subroutine evaluates the potential

$$u(x) = -\sum_{j=1}^{N} v_j d_j \cdot \nabla \left(H_0^{(1)}(k\|x - x_j\|) \right)$$

at the target locations $x = t_i$. When $x = x_j$, the term corresponding to x_j is dropped from the sum.

Input arguments:

- **eps: double precision** precision requested
- **zk: double complex** Helmholtz parameter, k
- **nsource: integer** Number of sources
- **source: double precision(3,nsource)** Source locations, x_j
- **dipstr: double complex(nsource)** Dipole strengths, v_j
- **dipvec: double precision(2,nsource)** Dipole orientation vectors, d_j
- **ntarg: integer** Number of targets
- **targ: double precision(3,ntarg)** Target locations, t_i

Output arguments:

- **pottarg: double complex(ntarg)** Potential at target locations, $u(t_i)$
- **ier: integer** Error flag; ier=0 implies successful execution, and ier=4/8 implies insufficient memory

---

Vectorized version:

```
subroutine hfmm2d_t_d_p_vec(nd,eps,zk,nsource,source,dipstr,dipvec,ntarg,targ,pottarg,
↪ier)
```

This subroutine evaluates the potential

$$u_\ell(x) = -\sum_{j=1}^{N} v_{\ell,j} d_{\ell,j} \cdot \nabla \left(H_0^{(1)}(k\|x - x_j\|) \right)$$

at the target locations $x = t_i$. When $x = x_j$, the term corresponding to x_j is dropped from the sum.

Input arguments:

- **nd: integer** number of densities

- **dipstr: double complex(nd,nsource)** Dipole strengths, $v_{\ell,j}$
- **dipvec: double precision(nd,2,nsource)** Dipole orientation vectors, $d_{\ell,j}$

Output arguments:

- **pottarg: double complex(nd,ntarg)** Potential at target locations, $u_\ell(t_i)$
- **ier: integer** Error flag; ier=0 implies successful execution, and ier=4/8 implies insufficient memory

Back to Helmholtz FMM

Back to top

hfmm2d_t_d_g

- Evaluation points: Targets
- Interaction kernel: Dipoles
- Outputs requested: Potential and Gradient

```
subroutine hfmm2d_t_d_g(eps,zk,nsource,source,dipstr,dipvec,ntarg,targ,pottarg,gradtarg,
↪ier)
```

This subroutine evaluates the potential and its gradient

$$u(x) = -\sum_{j=1}^{N} v_j d_j \cdot \nabla \left(H_0^{(1)}(k\|x - x_j\|) \right)$$

at the target locations $x = t_i$. When $x = x_j$, the term corresponding to x_j is dropped from the sum.

Input arguments:

- **eps: double precision** precision requested
- **zk: double complex** Helmholtz parameter, k
- **nsource: integer** Number of sources
- **source: double precision(3,nsource)** Source locations, x_j
- **dipstr: double complex(nsource)** Dipole strengths, v_j
- **dipvec: double precision(2,nsource)** Dipole orientation vectors, d_j
- **ntarg: integer** Number of targets
- **targ: double precision(3,ntarg)** Target locations, t_i

Output arguments:

- **pottarg: double complex(ntarg)** Potential at target locations, $u(t_i)$
- **gradtarg: double complex(2,ntarg)** Gradient at target locations, $\nabla u(t_i)$
- **ier: integer** Error flag; ier=0 implies successful execution, and ier=4/8 implies insufficient memory

---

Vectorized version:

```
subroutine hfmm2d_t_d_g_vec(nd,eps,zk,nsource,source,dipstr,dipvec,ntarg,targ,pottarg,
↪gradtarg,ier)
```

This subroutine evaluates the potential and its gradient

$$u_\ell(x) = -\sum_{j=1}^{N} v_{\ell,j} d_{\ell,j} \cdot \nabla \left(H_0^{(1)}(k\|x - x_j\|) \right)$$

at the target locations $x = t_i$. When $x = x_j$, the term corresponding to x_j is dropped from the sum.

Input arguments:

- **nd: integer** number of densities
- **dipstr: double complex(nd,nsource)** Dipole strengths, $v_{\ell,j}$
- **dipvec: double precision(nd,2,nsource)** Dipole orientation vectors, $d_{\ell,j}$

Output arguments:

- **pottarg: double complex(nd,ntarg)** Potential at target locations, $u_\ell(t_i)$
- **gradtarg: double complex(nd,2,ntarg)** Gradient at target locations, $\nabla u_\ell(t_i)$
- **ier: integer** Error flag; ier=0 implies successful execution, and ier=4/8 implies insufficient memory

Back to Helmholtz FMM

Back to top

hfmm2d_t_d_h

- Evaluation points: Targets
- Interaction kernel: Dipoles
- Outputs requested: Potential, Gradient and Hessian

```
subroutine hfmm2d_t_d_h(eps,zk,nsource,source,dipstr,dipvec,ntarg,targ,pottarg,gradtarg,
↪hesstarg,ier)
```

This subroutine evaluates the potential, its gradient, and its hessian

$$u(x) = -\sum_{j=1}^{N} v_j d_j \cdot \nabla \left(H_0^{(1)}(k\|x - x_j\|) \right)$$

at the target locations $x = t_i$. When $x = x_j$, the term corresponding to x_j is dropped from the sum.

Input arguments:

- **eps: double precision** precision requested
- **zk: double complex** Helmholtz parameter, k
- **nsource: integer** Number of sources

- **source: double precision(3,nsource)** Source locations, x_j
- **dipstr: double complex(nsource)** Dipole strengths, v_j
- **dipvec: double precision(2,nsource)** Dipole orientation vectors, d_j
- **ntarg: integer** Number of targets
- **targ: double precision(3,ntarg)** Target locations, t_i

Output arguments:

- **pottarg: double complex(ntarg)** Potential at target locations, $u(t_i)$
- **gradtarg: double complex(2,ntarg)** Gradient at target locations, $\nabla u(t_i)$
- **hesstarg: double complex(3,ntarg)** Hessian at target locations, $\nabla\nabla u(t_i)$
- **ier: integer** Error flag; ier=0 implies successful execution, and ier=4/8 implies insufficient memory

---

Vectorized version:

```
subroutine hfmm2d_t_d_h_vec(nd,eps,zk,nsource,source,dipstr,dipvec,ntarg,targ,pottarg,
↪gradtarg,hesstarg,ier)
```

This subroutine evaluates the potential, its gradient, and its hessian

$$u_\ell(x) = -\sum_{j=1}^{N} v_{\ell,j} d_{\ell,j} \cdot \nabla \left(H_0^{(1)}(k\|x - x_j\|) \right)$$

at the target locations $x = t_i$. When $x = x_j$, the term corresponding to x_j is dropped from the sum.

Input arguments:

- **nd: integer** number of densities
- **dipstr: double complex(nd,nsource)** Dipole strengths, $v_{\ell,j}$
- **dipvec: double precision(nd,2,nsource)** Dipole orientation vectors, $d_{\ell,j}$

Output arguments:

- **pottarg: double complex(nd,ntarg)** Potential at target locations, $u_\ell(t_i)$
- **gradtarg: double complex(nd,2,ntarg)** Gradient at target locations, $\nabla u_\ell(t_i)$
- **hesstarg: double complex(nd,3,ntarg)** Hessian at target locations, $\nabla u_\ell(t_i)$
- **ier: integer** Error flag; ier=0 implies successful execution, and ier=4/8 implies insufficient memory

Back to Helmholtz FMM

Back to top

hfmm2d_t_cd_p

- Evaluation points: Targets
- Interaction kernel: Charges and Dipoles
- Outputs requested: Potential

```
subroutine hfmm2d_t_cd_p(eps,zk,nsource,source,charge,dipstr,dipvec,ntarg,targ,pottarg,
↪ier)
```

This subroutine evaluates the potential

$$u(x) = \sum_{j=1}^{N} c_j H_0^{(1)}(k\|x - x_j\|) - v_j d_j \cdot \nabla \left(H_0^{(1)}(k\|x - x_j\|) \right)$$

at the target locations $x = t_i$. When $x = x_j$, the term corresponding to x_j is dropped from the sum.

Input arguments:

- **eps: double precision** precision requested
- **zk: double complex** Helmholtz parameter, k
- **nsource: integer** Number of sources
- **source: double precision(3,nsource)** Source locations, x_j
- **charge: double complex(nsource)** Charge strengths, c_j
- **dipstr: double complex(nsource)** Dipole strengths, v_j
- **dipvec: double precision(2,nsource)** Dipole orientation vectors, d_j
- **ntarg: integer** Number of targets
- **targ: double precision(3,ntarg)** Target locations, t_i

Output arguments:

- **pottarg: double complex(ntarg)** Potential at target locations, $u(t_i)$
- **ier: integer** Error flag; ier=0 implies successful execution, and ier=4/8 implies insufficient memory

---

Vectorized version:

```
subroutine hfmm2d_t_cd_p_vec(nd,eps,zk,nsource,source,charge,dipstr,dipvec,ntarg,targ,
↪pottarg,ier)
```

This subroutine evaluates the potential

$$u_\ell(x) = \sum_{j=1}^{N} c_{\ell,j} H_0^{(1)}(k\|x - x_j\|) - v_{\ell,j} d_{\ell,j} \cdot \nabla \left(H_0^{(1)}(k\|x - x_j\|) \right)$$

at the target locations $x = t_i$. When $x = x_j$, the term corresponding to x_j is dropped from the sum.

Input arguments:

- **nd: integer** number of densities
- **charge: double complex(nd,nsource)** Charge strengths, $c_{\ell,j}$
- **dipstr: double complex(nd,nsource)** Dipole strengths, $v_{\ell,j}$
- **dipvec: double precision(nd,2,nsource)** Dipole orientation vectors, $d_{\ell,j}$

Output arguments:

- **pottarg: double complex(nd,ntarg)** Potential at target locations, $u_\ell(t_i)$
- **ier: integer** Error flag; ier=0 implies successful execution, and ier=4/8 implies insufficient memory

Back to Helmholtz FMM

Back to top

hfmm2d_t_cd_g

- Evaluation points: Targets
- Interaction kernel: Charges and Dipoles
- Outputs requested: Potential and Gradient

```
subroutine hfmm2d_t_cd_g(eps,zk,nsource,source,charge,dipstr,dipvec,ntarg,targ,pottarg,
↪gradtarg,ier)
```

This subroutine evaluates the potential and its gradient

$$u(x) = \sum_{j=1}^{N} c_j H_0^{(1)}(k\|x - x_j\|) - v_j d_j \cdot \nabla \left(H_0^{(1)}(k\|x - x_j\|) \right)$$

at the target locations $x = t_i$. When $x = x_j$, the term corresponding to x_j is dropped from the sum.

Input arguments:

- **eps: double precision** precision requested
- **zk: double complex** Helmholtz parameter, k
- **nsource: integer** Number of sources
- **source: double precision(3,nsource)** Source locations, x_j
- **charge: double complex(nsource)** Charge strengths, c_j
- **dipstr: double complex(nsource)** Dipole strengths, v_j
- **dipvec: double precision(2,nsource)** Dipole orientation vectors, d_j
- **ntarg: integer** Number of targets
- **targ: double precision(3,ntarg)** Target locations, t_i

Output arguments:

- **pottarg: double complex(ntarg)** Potential at target locations, $u(t_i)$
- **gradtarg: double complex(2,ntarg)** Gradient at target locations, $\nabla u(t_i)$
- **ier: integer** Error flag; ier=0 implies successful execution, and ier=4/8 implies insufficient memory

---

Vectorized version:

```
subroutine hfmm2d_t_cd_g_vec(nd,eps,zk,nsource,source,charge,dipstr,dipvec,ntarg,targ,
↪pottarg,gradtarg,ier)
```

This subroutine evaluates the potential and its gradient

$$u_\ell(x) = \sum_{j=1}^{N} c_{\ell,j} H_0^{(1)}(k\|x - x_j\|) - v_{\ell,j} d_{\ell,j} \cdot \nabla \left(H_0^{(1)}(k\|x - x_j\|) \right)$$

at the target locations $x = t_i$. When $x = x_j$, the term corresponding to x_j is dropped from the sum.

Input arguments:

- **nd: integer** number of densities
- **charge: double complex(nd,nsource)** Charge strengths, $c_{\ell,j}$
- **dipstr: double complex(nd,nsource)** Dipole strengths, $v_{\ell,j}$
- **dipvec: double precision(nd,2,nsource)** Dipole orientation vectors, $d_{\ell,j}$

Output arguments:

- **pottarg: double complex(nd,ntarg)** Potential at target locations, $u_\ell(t_i)$
- **gradtarg: double complex(nd,2,ntarg)** Gradient at target locations, $\nabla u_\ell(t_i)$
- **ier: integer** Error flag; ier=0 implies successful execution, and ier=4/8 implies insufficient memory

Back to Helmholtz FMM

Back to top

hfmm2d_t_cd_h

- Evaluation points: Targets
- Interaction kernel: Charges and Dipoles
- Outputs requested: Potential, Gradient and Hessian

```
subroutine hfmm2d_t_cd_h(eps,zk,nsource,source,charge,dipstr,dipvec,ntarg,targ,pottarg,
↪gradtarg,hesstarg,ier)
```

This subroutine evaluates the potential, its gradient, and its hessian

$$u(x) = \sum_{j=1}^{N} c_j H_0^{(1)}(k\|x - x_j\|) - v_j d_j \cdot \nabla \left(H_0^{(1)}(k\|x - x_j\|) \right)$$

at the target locations $x = t_i$. When $x = x_j$, the term corresponding to x_j is dropped from the sum.

Input arguments:

- **eps: double precision** precision requested
- **zk: double complex** Helmholtz parameter, k
- **nsource: integer** Number of sources
- **source: double precision(3,nsource)** Source locations, x_j
- **charge: double complex(nsource)** Charge strengths, c_j
- **dipstr: double complex(nsource)** Dipole strengths, v_j
- **dipvec: double precision(2,nsource)** Dipole orientation vectors, d_j
- **ntarg: integer** Number of targets
- **targ: double precision(3,ntarg)** Target locations, t_i

Output arguments:

- **pottarg: double complex(ntarg)** Potential at target locations, $u(t_i)$
- **gradtarg: double complex(2,ntarg)** Gradient at target locations, $\nabla u(t_i)$
- **hesstarg: double complex(3,ntarg)** Hessian at target locations, $\nabla\nabla u(t_i)$
- **ier: integer** Error flag; ier=0 implies successful execution, and ier=4/8 implies insufficient memory

---

Vectorized version:

```
subroutine hfmm2d_t_cd_h_vec(nd,eps,zk,nsource,source,charge,dipstr,dipvec,ntarg,targ,
↪pottarg,gradtarg,hesstarg,ier)
```

This subroutine evaluates the potential, its gradient, and its hessian

$$u_\ell(x) = \sum_{j=1}^{N} c_{\ell,j} H_0^{(1)}(k\|x - x_j\|) - v_{\ell,j} d_{\ell,j} \cdot \nabla \left(H_0^{(1)}(k\|x - x_j\|) \right)$$

at the target locations $x = t_i$. When $x = x_j$, the term corresponding to x_j is dropped from the sum.

Input arguments:

- **nd: integer** number of densities
- **charge: double complex(nd,nsource)** Charge strengths, $c_{\ell,j}$
- **dipstr: double complex(nd,nsource)** Dipole strengths, $v_{\ell,j}$
- **dipvec: double precision(nd,2,nsource)** Dipole orientation vectors, $d_{\ell,j}$

Output arguments:

- **pottarg: double complex(nd,ntarg)** Potential at target locations, $u_\ell(t_i)$
- **gradtarg: double complex(nd,2,ntarg)** Gradient at target locations, $\nabla u_\ell(t_i)$
- **hesstarg: double complex(nd,3,ntarg)** Hessian at target locations, $\nabla u_\ell(t_i)$
- **ier: integer** Error flag; ier=0 implies successful execution, and ier=4/8 implies insufficient memory

Back to Helmholtz FMM

Back to top

hfmm2d_st_c_p

- Evaluation points: Sources and Targets
- Interaction kernel: Charges
- Outputs requested: Potential

```
subroutine hfmm2d_st_c_p(eps,zk,nsource,source,charge,pot,ntarg,targ,pottarg,ier)
```

This subroutine evaluates the potential

$$u(x) = \sum_{j=1}^{N} c_j H_0^{(1)}(k\|x - x_j\|)$$

at the source and target locations $x = x_j, t_i$. When $x = x_j$, the term corresponding to x_j is dropped from the sum.

Input arguments:

- **eps: double precision** precision requested
- **zk: double complex** Helmholtz parameter, k
- **nsource: integer** Number of sources
- **source: double precision(3,nsource)** Source locations, x_j
- **charge: double complex(nsource)** Charge strengths, c_j
- **ntarg: integer** Number of targets
- **targ: double precision(3,ntarg)** Target locations, t_i

Output arguments:

- **pot: double complex(nsource)** Potential at source locations, $u(x_j)$
- **pottarg: double complex(ntarg)** Potential at target locations, $u(t_i)$
- **ier: integer** Error flag; ier=0 implies successful execution, and ier=4/8 implies insufficient memory

---

Vectorized version:

```
subroutine hfmm2d_st_c_p_vec(nd,eps,zk,nsource,source,charge,pot,ntarg,targ,pottarg,ier)
```

This subroutine evaluates the potential

$$u_\ell(x) = \sum_{j=1}^{N} c_{\ell,j} H_0^{(1)}(k\|x - x_j\|)$$

at the source and target locations $x = x_j, t_i$. When $x = x_j$, the term corresponding to x_j is dropped from the sum.

Input arguments:

- **nd: integer** number of densities
- **charge: double complex(nd,nsource)** Charge strengths, $c_{\ell,j}$

Output arguments:

- **pot: double complex(nd,nsource)** Potential at source locations, $u_\ell(x_j)$
- **pottarg: double complex(nd,ntarg)** Potential at target locations, $u_\ell(t_i)$
- **ier: integer** Error flag; ier=0 implies successful execution, and ier=4/8 implies insufficient memory

Back to Helmholtz FMM

Back to top

hfmm2d_st_c_g

- Evaluation points: Sources and Targets
- Interaction kernel: Charges
- Outputs requested: Potential and Gradient

```
subroutine hfmm2d_st_c_g(eps,zk,nsource,source,charge,pot,grad,ntarg,targ,pottarg,
↪gradtarg,ier)
```

This subroutine evaluates the potential and its gradient

$$u(x) = \sum_{j=1}^{N} c_j H_0^{(1)}(k\|x - x_j\|)$$

at the source and target locations $x = x_j, t_i$. When $x = x_j$, the term corresponding to x_j is dropped from the sum.

Input arguments:

- **eps: double precision** precision requested
- **zk: double complex** Helmholtz parameter, k
- **nsource: integer** Number of sources
- **source: double precision(3,nsource)** Source locations, x_j
- **charge: double complex(nsource)** Charge strengths, c_j
- **ntarg: integer** Number of targets
- **targ: double precision(3,ntarg)** Target locations, t_i

Output arguments:

- **pot: double complex(nsource)** Potential at source locations, $u(x_j)$
- **grad: double complex(2,nsource)** Gradient at source locations, $\nabla u(x_j)$
- **pottarg: double complex(ntarg)** Potential at target locations, $u(t_i)$

- **gradtarg: double complex(2,ntarg)** Gradient at target locations, $\nabla u(t_i)$
- **ier: integer** Error flag; ier=0 implies successful execution, and ier=4/8 implies insufficient memory

---

Vectorized version:

```
subroutine hfmm2d_st_c_g_vec(nd,eps,zk,nsource,source,charge,pot,grad,ntarg,targ,pottarg,
↪gradtarg,ier)
```

This subroutine evaluates the potential and its gradient

$$u_\ell(x) = \sum_{j=1}^{N} c_{\ell,j} H_0^{(1)}(k\|x - x_j\|)$$

at the source and target locations $x = x_j, t_i$. When $x = x_j$, the term corresponding to x_j is dropped from the sum.

Input arguments:

- **nd: integer** number of densities
- **charge: double complex(nd,nsource)** Charge strengths, $c_{\ell,j}$

Output arguments:

- **pot: double complex(nd,nsource)** Potential at source locations, $u_\ell(x_j)$
- **grad: double complex(nd,2,nsource)** Gradient at source locations, $\nabla u_\ell(x_j)$
- **pottarg: double complex(nd,ntarg)** Potential at target locations, $u_\ell(t_i)$
- **gradtarg: double complex(nd,2,ntarg)** Gradient at target locations, $\nabla u_\ell(t_i)$
- **ier: integer** Error flag; ier=0 implies successful execution, and ier=4/8 implies insufficient memory

Back to Helmholtz FMM

Back to top

hfmm2d_st_c_h

- Evaluation points: Sources and Targets
- Interaction kernel: Charges
- Outputs requested: Potential, Gradient and Hessian

```
subroutine hfmm2d_st_c_h(eps,zk,nsource,source,charge,pot,grad,hess,ntarg,targ,pottarg,
↪gradtarg,hesstarg,ier)
```

This subroutine evaluates the potential, its gradient, and its hessian

$$u(x) = \sum_{j=1}^{N} c_j H_0^{(1)}(k\|x - x_j\|)$$

at the source and target locations $x = x_j, t_i$. When $x = x_j$, the term corresponding to x_j is dropped from the sum.

Input arguments:

- **eps: double precision** precision requested
- **zk: double complex** Helmholtz parameter, k
- **nsource: integer** Number of sources
- **source: double precision(3,nsource)** Source locations, x_j
- **charge: double complex(nsource)** Charge strengths, c_j
- **ntarg: integer** Number of targets
- **targ: double precision(3,ntarg)** Target locations, t_i

Output arguments:

- **pot: double complex(nsource)** Potential at source locations, $u(x_j)$
- **grad: double complex(2,nsource)** Gradient at source locations, $\nabla u(x_j)$
- **hess: double complex(3,nsource)** Hessian at source locations, $\nabla\nabla u(x_j)$
- **pottarg: double complex(ntarg)** Potential at target locations, $u(t_i)$
- **gradtarg: double complex(2,ntarg)** Gradient at target locations, $\nabla u(t_i)$
- **hesstarg: double complex(3,ntarg)** Hessian at target locations, $\nabla\nabla u(t_i)$
- **ier: integer** Error flag; ier=0 implies successful execution, and ier=4/8 implies insufficient memory

---

Vectorized version:

```
subroutine hfmm2d_st_c_h_vec(nd,eps,zk,nsource,source,charge,pot,grad,hess,ntarg,targ,
↪pottarg,gradtarg,hesstarg,ier)
```

This subroutine evaluates the potential, its gradient, and its hessian

$$u_\ell(x) = \sum_{j=1}^{N} c_{\ell,j} H_0^{(1)}(k\|x - x_j\|)$$

at the source and target locations $x = x_j, t_i$. When $x = x_j$, the term corresponding to x_j is dropped from the sum.

Input arguments:

- **nd: integer** number of densities
- **charge: double complex(nd,nsource)** Charge strengths, $c_{\ell,j}$

Output arguments:

- **pot: double complex(nd,nsource)** Potential at source locations, $u_\ell(x_j)$
- **grad: double complex(nd,2,nsource)** Gradient at source locations, $\nabla u_\ell(x_j)$
- **hess: double complex(nd,3,nsource)** Hessian at source locations, $\nabla\nabla u_\ell(x_j)$
- **pottarg: double complex(nd,ntarg)** Potential at target locations, $u_\ell(t_i)$
- **gradtarg: double complex(nd,2,ntarg)** Gradient at target locations, $\nabla u_\ell(t_i)$

- **hesstarg: double complex(nd,3,ntarg)** Hessian at target locations, $\nabla u_\ell(t_i)$
- **ier: integer** Error flag; ier=0 implies successful execution, and ier=4/8 implies insufficient memory

Back to Helmholtz FMM

Back to top

hfmm2d_st_d_p

- Evaluation points: Sources and Targets
- Interaction kernel: Dipoles
- Outputs requested: Potential

```
subroutine hfmm2d_st_d_p(eps,zk,nsource,source,dipstr,dipvec,pot,ntarg,targ,pottarg,ier)
```

This subroutine evaluates the potential

$$u(x) = -\sum_{j=1}^{N} v_j d_j \cdot \nabla \left(H_0^{(1)}(k\|x - x_j\|) \right)$$

at the source and target locations $x = x_j, t_i$. When $x = x_j$, the term corresponding to x_j is dropped from the sum.

Input arguments:

- **eps: double precision** precision requested
- **zk: double complex** Helmholtz parameter, k
- **nsource: integer** Number of sources
- **source: double precision(3,nsource)** Source locations, x_j
- **dipstr: double complex(nsource)** Dipole strengths, v_j
- **dipvec: double precision(2,nsource)** Dipole orientation vectors, d_j
- **ntarg: integer** Number of targets
- **targ: double precision(3,ntarg)** Target locations, t_i

Output arguments:

- **pot: double complex(nsource)** Potential at source locations, $u(x_j)$
- **pottarg: double complex(ntarg)** Potential at target locations, $u(t_i)$
- **ier: integer** Error flag; ier=0 implies successful execution, and ier=4/8 implies insufficient memory

---

Vectorized version:

```
subroutine hfmm2d_st_d_p_vec(nd,eps,zk,nsource,source,dipstr,dipvec,pot,ntarg,targ,
↪pottarg,ier)
```

This subroutine evaluates the potential

$$u_\ell(x) = -\sum_{j=1}^{N} v_{\ell,j} d_{\ell,j} \cdot \nabla \left(H_0^{(1)}(k\|x - x_j\|) \right)$$

at the source and target locations $x = x_j, t_i$. When $x = x_j$, the term corresponding to x_j is dropped from the sum.

Input arguments:

- **nd: integer** number of densities
- **dipstr: double complex(nd,nsource)** Dipole strengths, $v_{\ell,j}$
- **dipvec: double precision(nd,2,nsource)** Dipole orientation vectors, $d_{\ell,j}$

Output arguments:

- **pot: double complex(nd,nsource)** Potential at source locations, $u_\ell(x_j)$
- **pottarg: double complex(nd,ntarg)** Potential at target locations, $u_\ell(t_i)$
- **ier: integer** Error flag; ier=0 implies successful execution, and ier=4/8 implies insufficient memory

Back to Helmholtz FMM

Back to top

hfmm2d_st_d_g

- Evaluation points: Sources and Targets
- Interaction kernel: Dipoles
- Outputs requested: Potential and Gradient

```
subroutine hfmm2d_st_d_g(eps,zk,nsource,source,dipstr,dipvec,pot,grad,ntarg,targ,pottarg,
↪gradtarg,ier)
```

This subroutine evaluates the potential and its gradient

$$u(x) = -\sum_{j=1}^{N} v_j d_j \cdot \nabla \left(H_0^{(1)}(k\|x - x_j\|) \right)$$

at the source and target locations $x = x_j, t_i$. When $x = x_j$, the term corresponding to x_j is dropped from the sum.

Input arguments:

- **eps: double precision** precision requested
- **zk: double complex** Helmholtz parameter, k
- **nsource: integer** Number of sources
- **source: double precision(3,nsource)** Source locations, x_j
- **dipstr: double complex(nsource)** Dipole strengths, v_j
- **dipvec: double precision(2,nsource)** Dipole orientation vectors, d_j

- **ntarg: integer** Number of targets
- **targ: double precision(3,ntarg)** Target locations, t_i

Output arguments:

- **pot: double complex(nsource)** Potential at source locations, $u(x_j)$
- **grad: double complex(2,nsource)** Gradient at source locations, $\nabla u(x_j)$
- **pottarg: double complex(ntarg)** Potential at target locations, $u(t_i)$
- **gradtarg: double complex(2,ntarg)** Gradient at target locations, $\nabla u(t_i)$
- **ier: integer** Error flag; ier=0 implies successful execution, and ier=4/8 implies insufficient memory

---

Vectorized version:

```
subroutine hfmm2d_st_d_g_vec(nd,eps,zk,nsource,source,dipstr,dipvec,pot,grad,ntarg,targ,
↪pottarg,gradtarg,ier)
```

This subroutine evaluates the potential and its gradient

$$u_\ell(x) = -\sum_{j=1}^{N} v_{\ell,j} d_{\ell,j} \cdot \nabla \left(H_0^{(1)}(k\|x - x_j\|) \right)$$

at the source and target locations $x = x_j, t_i$. When $x = x_j$, the term corresponding to x_j is dropped from the sum.

Input arguments:

- **nd: integer** number of densities
- **dipstr: double complex(nd,nsource)** Dipole strengths, $v_{\ell,j}$
- **dipvec: double precision(nd,2,nsource)** Dipole orientation vectors, $d_{\ell,j}$

Output arguments:

- **pot: double complex(nd,nsource)** Potential at source locations, $u_\ell(x_j)$
- **grad: double complex(nd,2,nsource)** Gradient at source locations, $\nabla u_\ell(x_j)$
- **pottarg: double complex(nd,ntarg)** Potential at target locations, $u_\ell(t_i)$
- **gradtarg: double complex(nd,2,ntarg)** Gradient at target locations, $\nabla u_\ell(t_i)$
- **ier: integer** Error flag; ier=0 implies successful execution, and ier=4/8 implies insufficient memory

Back to Helmholtz FMM

Back to top

hfmm2d_st_d_h

- Evaluation points: Sources and Targets
- Interaction kernel: Dipoles
- Outputs requested: Potential, Gradient and Hessian

```
subroutine hfmm2d_st_d_h(eps,zk,nsource,source,dipstr,dipvec,pot,grad,hess,ntarg,targ,
↪pottarg,gradtarg,hesstarg,ier)
```

This subroutine evaluates the potential, its gradient, and its hessian

$$u(x) = -\sum_{j=1}^{N} v_j d_j \cdot \nabla \left(H_0^{(1)}(k\|x - x_j\|) \right)$$

at the source and target locations $x = x_j, t_i$. When $x = x_j$, the term corresponding to x_j is dropped from the sum.

Input arguments:

- **eps: double precision** precision requested
- **zk: double complex** Helmholtz parameter, k
- **nsource: integer** Number of sources
- **source: double precision(3,nsource)** Source locations, x_j
- **dipstr: double complex(nsource)** Dipole strengths, v_j
- **dipvec: double precision(2,nsource)** Dipole orientation vectors, d_j
- **ntarg: integer** Number of targets
- **targ: double precision(3,ntarg)** Target locations, t_i

Output arguments:

- **pot: double complex(nsource)** Potential at source locations, $u(x_j)$
- **grad: double complex(2,nsource)** Gradient at source locations, $\nabla u(x_j)$
- **hess: double complex(3,nsource)** Hessian at source locations, $\nabla\nabla u(x_j)$
- **pottarg: double complex(ntarg)** Potential at target locations, $u(t_i)$
- **gradtarg: double complex(2,ntarg)** Gradient at target locations, $\nabla u(t_i)$
- **hesstarg: double complex(3,ntarg)** Hessian at target locations, $\nabla\nabla u(t_i)$
- **ier: integer** Error flag; ier=0 implies successful execution, and ier=4/8 implies insufficient memory

---

Vectorized version:

```
subroutine hfmm2d_st_d_h_vec(nd,eps,zk,nsource,source,dipstr,dipvec,pot,grad,hess,ntarg,
↪targ,pottarg,gradtarg,hesstarg,ier)
```

This subroutine evaluates the potential, its gradient, and its hessian

$$u_\ell(x) = -\sum_{j=1}^{N} v_{\ell,j} d_{\ell,j} \cdot \nabla \left(H_0^{(1)}(k\|x - x_j\|) \right)$$

at the source and target locations $x = x_j, t_i$. When $x = x_j$, the term corresponding to x_j is dropped from the sum.

Input arguments:

- **nd: integer** number of densities
- **dipstr: double complex(nd,nsource)** Dipole strengths, $v_{\ell,j}$
- **dipvec: double precision(nd,2,nsource)** Dipole orientation vectors, $d_{\ell,j}$

Output arguments:

- **pot: double complex(nd,nsource)** Potential at source locations, $u_\ell(x_j)$
- **grad: double complex(nd,2,nsource)** Gradient at source locations, $\nabla u_\ell(x_j)$
- **hess: double complex(nd,3,nsource)** Hessian at source locations, $\nabla\nabla u_\ell(x_j)$
- **pottarg: double complex(nd,ntarg)** Potential at target locations, $u_\ell(t_i)$
- **gradtarg: double complex(nd,2,ntarg)** Gradient at target locations, $\nabla u_\ell(t_i)$
- **hesstarg: double complex(nd,3,ntarg)** Hessian at target locations, $\nabla u_\ell(t_i)$
- **ier: integer** Error flag; ier=0 implies successful execution, and ier=4/8 implies insufficient memory

Back to Helmholtz FMM

Back to top

hfmm2d_st_cd_p

- Evaluation points: Sources and Targets
- Interaction kernel: Charges and Dipoles
- Outputs requested: Potential

```
subroutine hfmm2d_st_cd_p(eps,zk,nsource,source,charge,dipstr,dipvec,pot,ntarg,targ,
↪pottarg,ier)
```

This subroutine evaluates the potential

$$u(x) = \sum_{j=1}^{N} c_j H_0^{(1)}(k\|x - x_j\|) - v_j d_j \cdot \nabla \left(H_0^{(1)}(k\|x - x_j\|) \right)$$

at the source and target locations $x = x_j, t_i$. When $x = x_j$, the term corresponding to x_j is dropped from the sum.

Input arguments:

- **eps: double precision** precision requested
- **zk: double complex** Helmholtz parameter, k

- **nsource: integer** Number of sources
- **source: double precision(3,nsource)** Source locations, x_j
- **charge: double complex(nsource)** Charge strengths, c_j
- **dipstr: double complex(nsource)** Dipole strengths, v_j
- **dipvec: double precision(2,nsource)** Dipole orientation vectors, d_j
- **ntarg: integer** Number of targets
- **targ: double precision(3,ntarg)** Target locations, t_i

Output arguments:

- **pot: double complex(nsource)** Potential at source locations, $u(x_j)$
- **pottarg: double complex(ntarg)** Potential at target locations, $u(t_i)$
- **ier: integer** Error flag; ier=0 implies successful execution, and ier=4/8 implies insufficient memory

---

Vectorized version:

```
subroutine hfmm2d_st_cd_p_vec(nd,eps,zk,nsource,source,charge,dipstr,dipvec,pot,ntarg,
↪targ,pottarg,ier)
```

This subroutine evaluates the potential

$$u_\ell(x) = \sum_{j=1}^{N} c_{\ell,j} H_0^{(1)}(k\|x - x_j\|) - v_{\ell,j} d_{\ell,j} \cdot \nabla \left(H_0^{(1)}(k\|x - x_j\|) \right)$$

at the source and target locations $x = x_j, t_i$. When $x = x_j$, the term corresponding to x_j is dropped from the sum.

Input arguments:

- **nd: integer** number of densities
- **charge: double complex(nd,nsource)** Charge strengths, $c_{\ell,j}$
- **dipstr: double complex(nd,nsource)** Dipole strengths, $v_{\ell,j}$
- **dipvec: double precision(nd,2,nsource)** Dipole orientation vectors, $d_{\ell,j}$

Output arguments:

- **pot: double complex(nd,nsource)** Potential at source locations, $u_\ell(x_j)$
- **pottarg: double complex(nd,ntarg)** Potential at target locations, $u_\ell(t_i)$
- **ier: integer** Error flag; ier=0 implies successful execution, and ier=4/8 implies insufficient memory

Back to Helmholtz FMM

Back to top

hfmm2d_st_cd_g

- Evaluation points: Sources and Targets
- Interaction kernel: Charges and Dipoles
- Outputs requested: Potential and Gradient

```
subroutine hfmm2d_st_cd_g(eps,zk,nsource,source,charge,dipstr,dipvec,pot,grad,ntarg,targ,
↪pottarg,gradtarg,ier)
```

This subroutine evaluates the potential and its gradient

$$u(x) = \sum_{j=1}^{N} c_j H_0^{(1)}(k\|x - x_j\|) - v_j d_j \cdot \nabla \left(H_0^{(1)}(k\|x - x_j\|) \right)$$

at the source and target locations $x = x_j, t_i$. When $x = x_j$, the term corresponding to x_j is dropped from the sum.

Input arguments:

- **eps: double precision** precision requested
- **zk: double complex** Helmholtz parameter, k
- **nsource: integer** Number of sources
- **source: double precision(3,nsource)** Source locations, x_j
- **charge: double complex(nsource)** Charge strengths, c_j
- **dipstr: double complex(nsource)** Dipole strengths, v_j
- **dipvec: double precision(2,nsource)** Dipole orientation vectors, d_j
- **ntarg: integer** Number of targets
- **targ: double precision(3,ntarg)** Target locations, t_i

Output arguments:

- **pot: double complex(nsource)** Potential at source locations, $u(x_j)$
- **grad: double complex(2,nsource)** Gradient at source locations, $\nabla u(x_j)$
- **pottarg: double complex(ntarg)** Potential at target locations, $u(t_i)$
- **gradtarg: double complex(2,ntarg)** Gradient at target locations, $\nabla u(t_i)$
- **ier: integer** Error flag; ier=0 implies successful execution, and ier=4/8 implies insufficient memory

---

Vectorized version:

```
subroutine hfmm2d_st_cd_g_vec(nd,eps,zk,nsource,source,charge,dipstr,dipvec,pot,grad,
↪ntarg,targ,pottarg,gradtarg,ier)
```

This subroutine evaluates the potential and its gradient

$$u_\ell(x) = \sum_{j=1}^{N} c_{\ell,j} H_0^{(1)}(k\|x - x_j\|) - v_{\ell,j} d_{\ell,j} \cdot \nabla \left(H_0^{(1)}(k\|x - x_j\|) \right)$$

at the source and target locations $x = x_j, t_i$. When $x = x_j$, the term corresponding to x_j is dropped from the sum.

Input arguments:

- **nd: integer** number of densities
- **charge: double complex(nd,nsource)** Charge strengths, $c_{\ell,j}$
- **dipstr: double complex(nd,nsource)** Dipole strengths, $v_{\ell,j}$
- **dipvec: double precision(nd,2,nsource)** Dipole orientation vectors, $d_{\ell,j}$

Output arguments:

- **pot: double complex(nd,nsource)** Potential at source locations, $u_\ell(x_j)$
- **grad: double complex(nd,2,nsource)** Gradient at source locations, $\nabla u_\ell(x_j)$
- **pottarg: double complex(nd,ntarg)** Potential at target locations, $u_\ell(t_i)$
- **gradtarg: double complex(nd,2,ntarg)** Gradient at target locations, $\nabla u_\ell(t_i)$
- **ier: integer** Error flag; ier=0 implies successful execution, and ier=4/8 implies insufficient memory

Back to Helmholtz FMM

Back to top

hfmm2d_st_cd_h

- Evaluation points: Sources and Targets
- Interaction kernel: Charges and Dipoles
- Outputs requested: Potential, Gradient and Hessian

```
subroutine hfmm2d_st_cd_h(eps,zk,nsource,source,charge,dipstr,dipvec,pot,grad,hess,ntarg,
↪targ,pottarg,gradtarg,hesstarg,ier)
```

This subroutine evaluates the potential, its gradient, and its hessian

$$u(x) = \sum_{j=1}^{N} c_j H_0^{(1)}(k\|x - x_j\|) - v_j d_j \cdot \nabla \left(H_0^{(1)}(k\|x - x_j\|) \right)$$

at the source and target locations $x = x_j, t_i$. When $x = x_j$, the term corresponding to x_j is dropped from the sum.

Input arguments:

- **eps: double precision** precision requested
- **zk: double complex** Helmholtz parameter, k
- **nsource: integer** Number of sources

- **source: double precision(3,nsource)** Source locations, x_j
- **charge: double complex(nsource)** Charge strengths, c_j
- **dipstr: double complex(nsource)** Dipole strengths, v_j
- **dipvec: double precision(2,nsource)** Dipole orientation vectors, d_j
- **ntarg: integer** Number of targets
- **targ: double precision(3,ntarg)** Target locations, t_i

Output arguments:

- **pot: double complex(nsource)** Potential at source locations, $u(x_j)$
- **grad: double complex(2,nsource)** Gradient at source locations, $\nabla u(x_j)$
- **hess: double complex(3,nsource)** Hessian at source locations, $\nabla\nabla u(x_j)$
- **pottarg: double complex(ntarg)** Potential at target locations, $u(t_i)$
- **gradtarg: double complex(2,ntarg)** Gradient at target locations, $\nabla u(t_i)$
- **hesstarg: double complex(3,ntarg)** Hessian at target locations, $\nabla\nabla u(t_i)$
- **ier: integer** Error flag; ier=0 implies successful execution, and ier=4/8 implies insufficient memory

---

Vectorized version:

```
subroutine hfmm2d_st_cd_h_vec(nd,eps,zk,nsource,source,charge,dipstr,dipvec,pot,grad,
↪hess,ntarg,targ,pottarg,gradtarg,hesstarg,ier)
```

This subroutine evaluates the potential, its gradient, and its hessian

$$u_\ell(x) = \sum_{j=1}^{N} c_{\ell,j} H_0^{(1)}(k\|x - x_j\|) - v_{\ell,j} d_{\ell,j} \cdot \nabla \left(H_0^{(1)}(k\|x - x_j\|) \right)$$

at the source and target locations $x = x_j, t_i$. When $x = x_j$, the term corresponding to x_j is dropped from the sum.

Input arguments:

- **nd: integer** number of densities
- **charge: double complex(nd,nsource)** Charge strengths, $c_{\ell,j}$
- **dipstr: double complex(nd,nsource)** Dipole strengths, $v_{\ell,j}$
- **dipvec: double precision(nd,2,nsource)** Dipole orientation vectors, $d_{\ell,j}$

Output arguments:

- **pot: double complex(nd,nsource)** Potential at source locations, $u_\ell(x_j)$
- **grad: double complex(nd,2,nsource)** Gradient at source locations, $\nabla u_\ell(x_j)$
- **hess: double complex(nd,3,nsource)** Hessian at source locations, $\nabla\nabla u_\ell(x_j)$
- **pottarg: double complex(nd,ntarg)** Potential at target locations, $u_\ell(t_i)$
- **gradtarg: double complex(nd,2,ntarg)** Gradient at target locations, $\nabla u_\ell(t_i)$
- **hesstarg: double complex(nd,3,ntarg)** Hessian at target locations, $\nabla u_\ell(t_i)$

- **ier: integer** Error flag; ier=0 implies successful execution, and ier=4/8 implies insufficient memory

Back to Helmholtz FMM

Back to top