CHarm

C Library to Work with Spherical Harmonics up to Almost Arbitrarily High Degrees

Blažej Bucha

Department of Theoretical Geodesy and Geoinformatics
Slovak University of Technology in Bratislava
blazej.bucha@stuba.sk

EGU General Assembly 2022, Vienna

Main Features

- FFT-based surface SHA and solid SHS

Main Features

- FFT-based surface SHA and solid SHS
- Stable up to high degrees (tens of thousands and beyond)

Main Features

- FFT-based surface SHA and solid SHS
- Stable up to high degrees (tens of thousands and beyond)
- Single, double and quadruple precision

Main Features

- FFT-based surface SHA and solid SHS
- Stable up to high degrees (tens of thousands and beyond)
- Single, double and quadruple precision
- Works with point and area-mean data values (both SHA and SHS)

Main Features

- FFT-based surface SHA and solid SHS
- Stable up to high degrees (tens of thousands and beyond)
- Single, double and quadruple precision
- Works with point and area-mean data values (both SHA and SHS)
- Integrates solid spherical harmonic expansions on band-limited undulated surfaces (*unique to CHarm*)

Main Features

- FFT-based surface SHA and solid SHS
- Stable up to high degrees (tens of thousands and beyond)
- Single, double and quadruple precision
- Works with point and area-mean data values (both SHA and SHS)
- Integrates solid spherical harmonic expansions on band-limited undulated surfaces (*unique to CHarm*)
- Computes Fourier coefficients of Legendre functions

Main Features

- FFT-based surface SHA and solid SHS
- Stable up to high degrees (tens of thousands and beyond)
- Single, double and quadruple precision
- Works with point and area-mean data values (both SHA and SHS)
- Integrates solid spherical harmonic expansions on band-limited undulated surfaces (*unique to CHarm*)
- Computes Fourier coefficients of Legendre functions
- Etcetera, etcetera

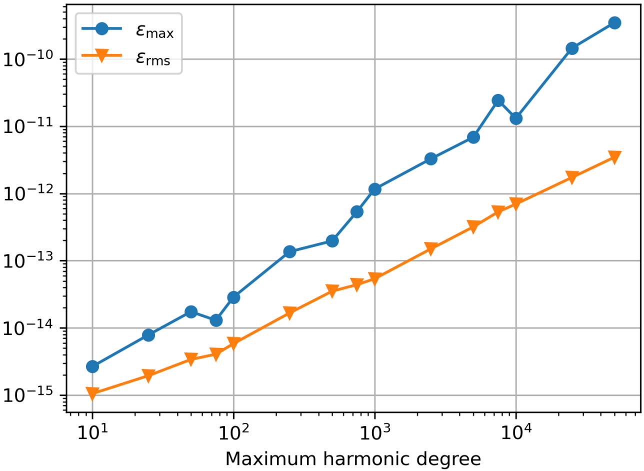
Main Features

- FFT-based surface SHA and solid SHS
- Stable up to high degrees (tens of thousands and beyond)
- Single, double and quadruple precision
- Works with point and area-mean data values (both SHA and SHS)
- Integrates solid spherical harmonic expansions on band-limited undulated surfaces (*unique to CHarm*)
- Computes Fourier coefficients of Legendre functions
- Etcetera, etcetera
- Discrete FFT by FFTW

Main Features

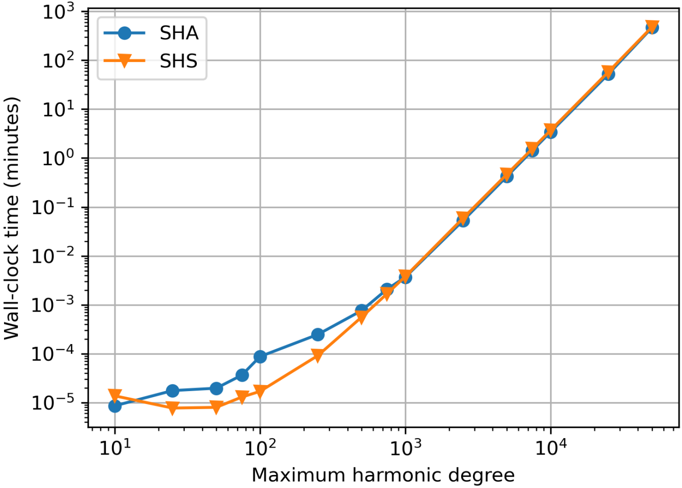
- FFT-based surface SHA and solid SHS
- Stable up to high degrees (tens of thousands and beyond)
- Single, double and quadruple precision
- Works with point and area-mean data values (both SHA and SHS)
- Integrates solid spherical harmonic expansions on band-limited undulated surfaces (*unique to CHarm*)
- Computes Fourier coefficients of Legendre functions
- Etcetera, etcetera
- Discrete FFT by FFTW
- OpenMP parallelization for shared-memory architectures

Accuracy (Double Precision)

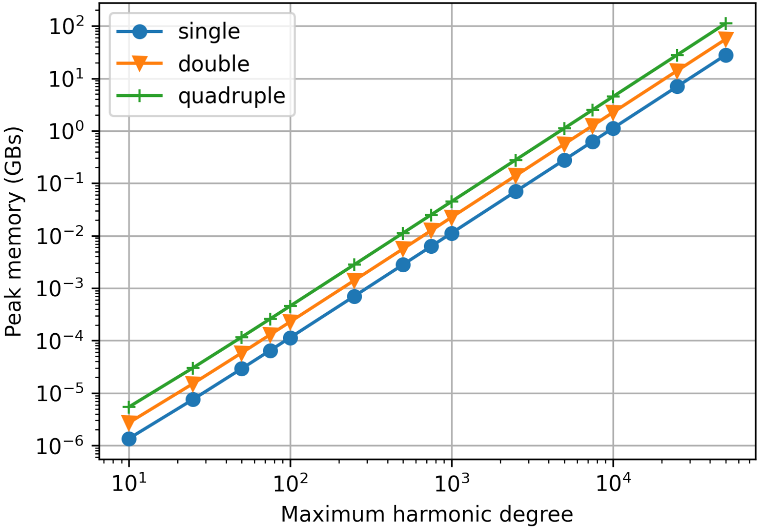


Speed (Double Precision)

Executed on a PC with Intel(R) Core(TM) i7-6800K CPU @ 3.40GHz. Compiled using GCC with parallelization enabled and the `-O3` and `-ffast-math` optimization flags. All 6 CPU cores were employed with hyperthreading enabled.



Memory Management



Tested Platforms

Architecture	Operating System	C Compiler
x86_64	Debian GNU/Linux 11 (bullseye)	GCC 10.2.1
		Clang 11.0.1-2
		ICC 2021.5.0
x86_64	Scientific Linux release 6.4 (Carbon)	GCC 7.2
		GCC 6.4
		GCC 5.4
		GCC 4.9.3
		GCC 4.8.4
		GCC 4.4.7
x86_64	Manjaro Linux	GCC 11.2.0
x86_64	FreeBSD 13.0-RELEASE	Clang 11.0.1
		GCC 10.3.0
x86_64	macOS Big Sur 11.4	Clang 12.0.5
		GCC 10.3.0
x86_64	Windows 10 (WSL, Debian (bullseye))	GCC 10.2.1

Source Code

- Source Code: `https://github.com/blazej-bucha/charm`
  Releases in `master`
  Development in `develop`

Source Code

- Source Code: `https://github.com/blazej-bucha/charm`
  Releases in `master`
  Development in `develop`
- Tarball and Zip Files of Releases:
  `https://github.com/blazej-bucha/charm/tags`

Source Code

- Source Code: `https://github.com/blazej-bucha/charm`
  Releases in `master`
  Development in `develop`
- Tarball and Zip Files of Releases:
  `https://github.com/blazej-bucha/charm/tags`
- Unrestrictive 3-clause BSD license

Documentation

https://blazej-bucha.github.io/charm/index.html

0.0.0

Search docs

charm_shc ***charm_shc_init**(unsigned long *nmax*, double *mu*, double *r*)

Allocates and initializes a `charm_shc` structure of spherical harmonic coefficients up to the degree `nmax`. All coefficients are initialized to zero and are associated with the scaling parameter `mu` and the radius of the reference sphere `r`.

On success, returned is a pointer to the `charm_shc` structure. On error, `NULL` is returned.

Warning

The `charm_shc` structure created by this function must be deallocated by calling `charm_shc_free`. The `free` function will *not* deallocate the memory and will lead to memory leaks.

Note

`r` must be greater than zero.

void **charm_shc_free**(charm_shc **shcs*)

Frees the memory associated with `shcs`. No operation is performed if `shcs` is `NULL`.

void **charm_shc_read_bin**(FILE **stream*, unsigned long *nmax*, charm_shc **shcs*, charm_err **err*)

Reads a `charm_shc` structure to `shcs` from a binary file pointed to by `stream`. The structure is loaded up to the maximum spherical harmonic degree `nmax`. The file is assumed to has been created by `charm_shc_write_bin` on the same architecture. Error reported by the function (if any) is written to `err`.

The input file is a binary representation of the `charm_shc` structure in the following order:

$$\mathrm{nmax2},\ \mu,\ R,\bar{C}_{0,0},\bar{C}_{1,0},\bar{C}_{2,0},\cdots,\bar{C}_{\mathrm{nmax2},0},\bar{C}_{1,1},\ \bar{C}_{2,1},\ \cdots,$$
$$\bar{C}_{\mathrm{nmax2},1},\ \bar{C}_{2,2},\bar{C}_{3,2},\ \cdots,\ \bar{C}_{\mathrm{nmax2},\mathrm{nmax2}},\ \bar{S}_{0,0},\ \bar{S}_{1,0},\ \bar{S}_{2,0},\ \cdots,$$
$$\bar{S}_{\mathrm{nmax2},0},\ \bar{S}_{1,1},\ \bar{S}_{2,1},\ \cdots,\bar{S}_{\mathrm{nmax2},1},\ \bar{S}_{2,2},\ \bar{S}_{3,2},\ \cdots,\bar{S}_{\mathrm{nmax2},\mathrm{nmax2}},$$

where `nmax2` is the maximum harmonic degree related to the `charm_shc` structure stored in the file,

$$\mu, R$$

are the scaling parameter of the coefficients and the associated radius of the reference sphere

Future Work

- Add the Condon–Shortley phase factor.

Future Work

- Add the Condon–Shortley phase factor.
- Add other normalization schemes.

Future Work

- Add the Condon–Shortley phase factor.
- Add other normalization schemes.
- Add polar optimization.

Future Work

- Add the Condon–Shortley phase factor.
- Add other normalization schemes.
- Add polar optimization.
- Add support for AVX, AVX2, AVX-512, etc. CPU instructions.

Future Work

- Add the Condon–Shortley phase factor.
- Add other normalization schemes.
- Add polar optimization.
- Add support for AVX, AVX2, AVX-512, etc. CPU instructions.
- Create a Python wrapper, probably using ctypes.

Future Work

- Add the Condon–Shortley phase factor.
- Add other normalization schemes.
- Add polar optimization.
- Add support for AVX, AVX2, AVX-512, etc. CPU instructions.
- Create a Python wrapper, probably using ctypes.
- Alternatively build CHarm with CMake on Windows.

How to make software attractive?

Get a logo!

Get a logo!

Thank you for your attention!