

# HEALPix C Subroutines Overview



Revision: Version 2.11; November 13, 2008

Prepared by: Eric Hivon, Anthony J. Banday, Matthias Bartelmann, Frode K. Hansen, Krzysztof M. Górski, Martin Reinecke and Benjamin D. Wandelt

Abstract: This document is an overview of the **HEALPix** C subroutines.

## Contents

Conventions . . . . .	3
Compilation and Installation . . . . .	3
Usage . . . . .	3
Note on the C routines . . . . .	3
ang2vec . . . . .	4
get_fits_size . . . . .	5
nside2npix . . . . .	7
pix2xxx, ang2xxx, vec2xxx, nest2ring, ring2nest . . . . .	8
read_healpix_map . . . . .	11
vec2ang . . . . .	12
write_healpix_map . . . . .	13

## Conventions

Here we list some conventions which are used in this document.

---

$N_{\text{side}}$	<b>HEALPix</b> resolution parameter — see the <b>HEALPix</b> Primer.
$\theta$	The polar angle or colatitude on the sphere, ranging from 0 at the North Pole to $\pi$ at the South Pole.
$\phi$	The azimuthal angle on the sphere, $\phi \in [0, 2\pi[$ .

## Compilation and Installation

A tentative compilation and installation script is provided in `src/C/doinstall.s`. If it does not work, you can try editing the `src/C/subs/Makefile` by hand.

## Usage

To use in your 'c' code, include the line

```
#include "chealpix.h"
```

in your code and link with something like

```
gcc -o myprog myprog.c -I<incdir> -L<libdir> -lchealpix
```

where `|incdir|` is where you've installed the '.h' files and `|libdir|` is where you've installed the libraries (See the header of the 'subs/Makefile').

You will also need the 'cfitsio' library. See <http://heasarc.gsfc.nasa.gov/docs/software/fitsio/>

## Note on the C routines

This small set of C routines is provided as a start up kit to users wanting to link the **HEALPix** routines with some other languages (C, C++, IDL, perl, ...), and it was actually mainly provided by various users (see individual routines for details). As for the rest of the **HEALPix** package, all interested persons are welcome to contribute to this effort.

## ang2vec

---

Location in HEALPix directory tree: `src/C/subs/ang2vec.c`

Routine to convert the position angles  $(\theta, \phi)$  of a point on the sphere into its 3D position vector  $(x, y, z)$  with  $x = \sin \theta \cos \phi$ ,  $y = \sin \theta \sin \phi$ ,  $z = \cos \theta$ .

---

**FORMAT**                      `void vec2ang(double theta, double phi, double *vector);`

---

### ARGUMENTS

name & dimensionality	kind	in/out	description
theta	double	IN	colatitude in radians measured southward from north pole (in $[0, \pi]$ ).
phi	double	IN	longitude in radians measured eastward (in $[0, 2\pi]$ ).
vector(3)	double	OUT	three dimensional cartesian position vector $(x, y, z)$ . The north pole is $(0, 0, 1)$

---

### RELATED ROUTINES

This section lists the routines related to **ang2vec**.

vec2ang	converts the 3D position vector of point into its position angles on the sphere.
---------	--

---

# get\_fits\_size

---

**Location in HEALPix directory tree:** src/C/subs/get\_fits\_size.c

This routine reads the number of pixels, the resolution parameter and the pixel ordering of a FITS file containing a **HEALPix** map.

---

**FORMAT**                    long get\_fits\_size(char \*filename, long \*nside, char \*ordering)

---

## ARGUMENTS

---

name&dimensionality	kind	in/out	description
get_fits_size	long	OUT	number of pixels the FITS file
filename	char	IN	filename of the FITS-file containing the <b>HEALPix</b> map.
ordering	char	OUT	pixel ordering, either 'RING' or 'NESTED'
nside	long	OUT	Healpix resolution parameter Nside

---

## EXAMPLE:

---

```
long npix, nside ;
char file[180]=''map.fits'' ;
char order[10] ;
npix= get_fits_size(file, &nside, order)
```

Returns in npix the number of pixel in the file 'map.fits', and read in nside or order its resolution parameter or ordering scheme

---

## RELATED ROUTINES

This section lists the routines related to **get\_fits\_size**.

read_healpix_map	subroutine to read <b>HEALPix</b> maps
write_healpix_map	subroutine to write <b>HEALPix</b> maps

---

# nside2npix

---

**Location in HEALPix directory tree:** `src/C/subs/nside2npix.c`

Function to provide the number of pixels  $N_{\text{pix}}$  over the full sky corresponding to resolution parameter  $N_{\text{side}}$ .

---

**FORMAT**                      `long nside2npix(const long nside)`

---

## ARGUMENTS

name&dimensionality	kind	in/out	description
nside	long	IN	the $N_{\text{side}}$ parameter of the map.
nside2npix	long	OUT	returns the number of pixels $N_{\text{pix}}$ of the map $N_{\text{pix}} = 12N_{\text{side}}^2$ .

---

## EXAMPLE:

```
npix= nside2npix(256);
```

Returns the pixel the number of **HEALPix** pixels (786432) for the resolution parameter 256.

---

# pix2xxx, ang2xxx, vec2xxx, nest2ring, ring2nest

Location in HEALPix directory tree: `src/C/subs/*.c`

These subroutines can be used to convert between pixel number in the **HEALPix** map and  $(\theta, \phi)$  coordinates on the sphere. This is only a subset of the routines equivalent in Fortran90 or in IDL.

Note: These routines are based on the translation of the original F77 routines to C++ and then to C, by Reza Ansari (ansari@lal.in2p3.fr), Alex Kim (akim@lilys.lbl.gov), Guy Le Meur (lemeur@lal.in2p3.fr), Benoit Revenu (revenu@iap.fr) and Ken Ganga (kmg@ipac.caltech.edu).

## ARGUMENTS

name & dimensionality	type	in/out	description
nside	long	IN	$N_{side}$ parameter for the <b>HEALPix</b> map.
ipnest	long	—	pixel identification number in NESTED scheme over the range $\{0, N_{pix} - 1\}$ .
ipring	long	—	pixel identification number in RING scheme over the range $\{0, N_{pix} - 1\}$ .
theta	double	—	colatitude in radians measured southward from north pole in $[0, \pi]$ .
phi	double	—	longitude in radians, measured eastward in $[0, 2\pi]$ .
vector	double	—	3D cartesian position vector $(x, y, z)$ . The north pole is $(0, 0, 1)$ . An output vector is normalised to unity.



---

**ROUTINES:**

```
void pix2ang_ring(long nside, long ipring, double *theta, double *phi);
```

renders *theta* and *phi* coordinates of the nominal pixel center given the pixel number *ipring* and a map resolution parameter *nside*.

```
void pix2vec_ring(long nside, long ipring, double *vector);
```

renders cartesian vector coordinates of the nominal pixel center given the pixel number *ipring* and a map resolution parameter *nside*. Optionally renders cartesian vector coordinates of the considered pixel four vertices.

```
void ang2pix_ring(long nside, double theta, double phi, long *ipring);
```

renders the pixel number *ipring* for a pixel which, given the map resolution parameter *nside*, contains the point on the sphere at angular coordinates *theta* and *phi*.

```
void vec2pix_ring(long nside, double *vector, long *ipring);
```

renders the pixel number *ipring* for a pixel which, given the map resolution parameter *nside*, contains the point on the sphere at cartesian coordinates *vector*.

```
void pix2ang_nest(long nside, long ipnest, double *theta, double *phi);
```

renders *theta* and *phi* coordinates of the nominal pixel center given the pixel number *ipnest* and a map resolution parameter *nside*.

```
void pix2vec_nest(long nside, long ipnest, double *vector);
```

renders cartesian vector coordinates of the nominal pixel center given the pixel number *ipnest* and a map resolution parameter *nside*. Optionally renders cartesian vector coordinates of the considered pixel four vertices.

```
void ang2pix_nest(long nside, double theta, double phi, long *ipnest);
```

renders the pixel number *ipnest* for a pixel which, given the map resolution parameter *nside*, contains the point on the sphere at angular coordinates *theta* and *phi*.

```
void vec2pix_nest(long nside, double *vector, long *ipnest)
```

renders the pixel number *ipnest* for a pixel which, given the map resolution parameter *nside*, contains the point on the sphere at cartesian coordinates *vector*.

```
void nest2ring(long nside, long ipnest, long *ipring);
```

performs conversion from NESTED to RING pixel number.

```
void ring2nest(long nside, long ipring, long *ipnest);
```

performs conversion from RING to NESTED pixel number.

---

## MODULES & ROUTINES

This section lists the modules and routines used by **pix2xxx**, **ang2xxx**, **vec2xxx**, **nest2ring**, **ring2nest**.

mk_pix2xy, mk_xy2pix	routines used in the conversion between pixel values and “cartesian” coordinates on the Healpix face.
----------------------	---

---

## RELATED ROUTINES

This section lists the routines related to **pix2xxx**, **ang2xxx**, **vec2xxx**, **nest2ring**, **ring2nest**.

ang2vec	convert $(\theta, \phi)$ spherical coordinates into $(x, y, z)$ cartesian coordinates.
vec2ang	convert $(x, y, z)$ cartesian coordinates into $(\theta, \phi)$ spherical coordinates.

---

# read\_healpix\_map

Location in HEALPix directory tree: `src/C/subs/read_healpix_map.c`

This routine reads a full sky **HEALPix** map from a FITS file

---

**FORMAT**                      float    \*read\_healpix\_map(char    \*infile,    long  
                                 \*nside, char \*coordsys, char \*ordering)

---

## ARGUMENTS

name&dimensionality	kind	in/out	description
read_healpix_map	float	OUT	array containing the map read from the file
infile	char	IN	FITS file containing a full sky to be read
nside	long	OUT	<b>HEALPix</b> resolution parameter of the map
coordsys	char	OUT	astronomical coordinate system of pixelisation (either 'C', 'E' or 'G' standing respectively for Celestial=equatorial, Ecliptic or Galactic)
ordering	char	OUT	<b>HEALPix</b> pixel ordering (either 'RING' or 'NESTED')

---

## RELATED ROUTINES

This section lists the routines related to **read\_healpix\_map**.

anafast	executable that reads a <b>HEALPix</b> map and analyses it.
synfast	executable that generate full sky <b>HEALPix</b> maps
write_healpix_map	subroutine to write <b>HEALPix</b> maps
get_fits_size	subroutine to determine the size of a map

---

## vec2ang

Location in HEALPix directory tree: `src/C/subs/vec2ang.c`

Routine to convert the 3D position vector  $(x, y, z)$  of point into its position angles  $(\theta, \phi)$  on the sphere with  $x = \sin \theta \cos \phi$ ,  $y = \sin \theta \sin \phi$ ,  $z = \cos \theta$ .

---

**FORMAT**                      `void vec2ang(double *vector, double *theta, double *phi);`

---

## ARGUMENTS

name & dimensionality	kind	in/out	description
vector(3)	double	IN	three dimensional cartesian position vector $(x, y, z)$ . The north pole is $(0, 0, 1)$
theta	double	OUT	colatitude in radians measured southward from north pole (in $[0, \pi]$ ).
phi	double	OUT	longitude in radians measured eastward (in $[0, 2\pi]$ ).

## RELATED ROUTINES

This section lists the routines related to **vec2ang**.

ang2vec	converts the position angles of a point on the sphere into its 3D position vector.
---------	--

---

# write\_healpix\_map

Location in HEALPix directory tree: `src/C/subs/write_healpix_map.c`

This routine writes a full sky **HEALPix** map into a FITS file

---

**FORMAT**                      `int write_healpix_map( float *signal, long nside, char *filename, char nest, char *coordsys)`

---

## ARGUMENTS

name&dimensionality	kind	in/out	description
write_healpix_map	int	OUT	returns a non zero value in case of error
signal	float	IN	full sky map to be written
nside	long	IN	<b>HEALPix</b> resolution parameter of the map (the map should have $12 * nside * nside$ pixels).
filename	char	IN	FITS file in which to write the full sky map
nest	char	IN	flag specifying the <b>HEALPix</b> pixel ordering of the map. 0: 'RING' and 1: 'NESTED'
coordsys	char	IN	astronomical coordinate system of map (must be either 'C', 'E' or 'G' standing respectively for Celestial=equatorial, Ecliptic or Galactic)

---

## RELATED ROUTINES

This section lists the routines related to **write\_healpix\_map**.

anafast	executable that reads a <b>HEALPix</b> map and analyses it.
synfast	executable that generate full sky <b>HEALPix</b> maps
read_healpix_map	subroutine to read <b>HEALPix</b> maps
get_fits_size	subroutine to determine the size of a map

---