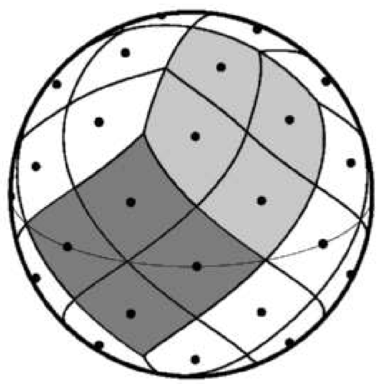
HEALPix Fortran Facility User Guidelines



Revision: Version 3.83; November 13, 2024

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

Abstract: This document describes the **HEALPix** Fortran stand-alone facilities

https://healpix.sourceforge.io
http://healpix.sf.net

TABLE OF CONTENTS

Using the HEALPix Fortran 90 facilities

Default file names and directories

For some applications, the **HEALPix** facilities require some precalculated input files describing the pixel window function and ring-based or pixel-based quadrature weights (shipped as `Healpix/data/pixel_window_n????.fits`, `Healpix/data/weight_ring_n?????.fits` and `Healpix/data/weight_pixel_n?????.fits` respectively).

By default, files with the very same name (generated by functions such as `get_healpix_pixel_window_file`) will be looked for into a list of directories (generated by `get_healpix_data_dir`) containing the current directory (`.`), the parent directory (`..`), `./data`, `../data`, `$HEALPIX` and `$HEALPIX/data` where `$HEALPIX` is a system variable defined as the full path to the **HEALPix** package (see the installation documentation). However, the user has the possibility to change both the name of those files and their location (with options like `windowfile`, `winfiledir`, `w8file`, or `w8filedir`)

Double/Single precision mode

Several facilities offer the option of switching at run time the precision of the internal variables and arrays and of the I/O data from single to double precision floating point reals. The following points should be noted:

- Facilities running in double precision mode can read indifferently single and double precision data files (and the same is true for single precision facilities). On the other hand, a double (resp. single) precision facility will only output double (single) precision files.
- Since the internal calculations sensitive to numerical round-off error (like the spherical harmonics recurrence) are *always* performed in double precision, switching to double precision mode
  - will have a limited impact on the output accuracy if the input file contains only single precision data,
  - is recommended if the input file contains double precision data, and the precision of the output is critical
  - will only increase a bit the execution time (by 15% in some of our tests), but it will almost double the memory consumption of the facility,
  - will obviously double the size of the output file(s).

Beam window function files

Several F90 and IDL applications (eg, alteralm, sky_ng_sim, smoothing, synfast, ialteralm, ismoothing, isynfast) accept, generally with the argument `beam_file` a circular (possibly non-gaussian) beam or smoothing window, which is described under the form of its *real* (polarized) Legendre window functions (see IDL's beam2bl) read from an external file. The file is either

- *highly recommended:* a FITS file containing a binary or ASCII table, with 1, 2 or 3 fields, respectively
  $b_T(\ell)$, $[b_E(\ell)$, $[b_B(\ell)]]$
- in IDL, is can also be an array of size $(\ell_{\max}+1, d)$ with d =1, 2 or 3, which will be automatically turned into a FITS file of the format described above using the **HEALPix**/IDL routine bl2fits.
- or, it can also be, a plain text file with 2, 3 or 4 columns, containing, on each line,
  ℓ, $b_T(\ell)$, $[b_E(\ell)$, $[b_B(\ell)]]$
  and in which lines starting with a # sign are ignored.

In any case, the multipole ℓ must take all integer values in $\{0,\ldots,\ell_{\max}\}$, with the assumption that the window functions vanish outside that range. b_T is the 'temperature' or intensity window function, while the optional b_E and b_B are respectively the electric (or gradient) and magnetic (or curl) components of polarization. If not provided, $b_B(\ell)$ takes the value of $b_E(\ell)$ which itself defaults to $b_T(\ell)$. With these window functions, the (polarized) map spherical harmonics coefficients and its power spectra are transformed according to

$$a^X_{\ell m} \longrightarrow a^X_{\ell m} b_X(\ell), \tag{1}$$
$$C^{XY}_{\ell m} \longrightarrow C^{XY}_{\ell m} b_X(\ell) b_Y(\ell), \tag{2}$$

with $X, Y \in \{T, E, B\}$.

Changes between releases 3.80 and 3.83

- Bug corrections in `input_map` and `read_fits_partial`,
- added a workaround for a bug detected in Apple-ARM-chips implementation of gfortran 11.
- Note that cfitsio 4.1.0 or higher is required for Fortran codes running on Apple's ARM chips

Older Changes

Version 3.80

- correction of bugs preventing the compilation with versions 10.* of `gfortran`,
- fixed bug affecting `map2gif` when compiled with versions 10.* of `gfortran` and `gcc`.

Version 3.70

- Addition of the subroutines `read_fits_partial`, and `write_fits_partial`, to read and write polarized or unpolarized map defined on a fraction of the sky.

Version 3.60

- Faster Spherical Harmonics Transforms in `anafast`, `smoothing`, and `synfast` thanks to the new `libsharp` library.

Version 3.50

- correction of a bug in `anafast` when `maskfile` or `theta_cut_deg` are used in combination with `iter_order`> 0,
- addition of `zbounds` in `alm2map`, `alm2map_der`, `alm2map_spin`, in order to simulate (faster) a signal on only a fraction of the sphere,
- improved support for version 18 and more of Intel C and F90 compilers in `configure` script,
- edition to fitstools.F90 allowing a proper compilation with g95.

Version 3.40

- The facilities `anafast` and `smoothing` now support pixel-based quadrature weights. Introduction of the supporting `nside2npweights`, `unfold_weightsfile`, `get_healpix_weight_file`, `get_healpix_pixel_weight_file`.
- The facilities `anafast`, `median_filter`, `smoothing`, and `ud_grade` test the value of the `POLCCONV` FITS keyword when reading a polarized map, and interpret the polarization accordingly, as described in the note on POLCCONV in The **HEALPix** Primer.
- `median_filter` faster by moving an internal array from heap to stack; does not crash anymore when dealing with empty data sets; slightly different output when median is computed over an even number of pixels $n \geq 100$, sorted in $[1, n]$, the output is $d(n/2)$ instead of $(d(n/2) + d(n/2 + 1))/2$ previously. Result remains $d(n/2 + 1)$ for odd n.

Version 3.31

- Bug correction in `input_map` routine for reading of polarized multi-HDU cut sky FITS files;
- Introduction of `winfiledir_*` and `windowfile_*` qualifiers in `alteralm` facility.

Version 3.30

- anafast now produces nine spectra (TT, EE, BB, TE, TB, EB, ET, BT and BE), instead of six previously, when analyzing two polarized maps
- CFITSIO version 3.20 (August 2009) or more now required

Version 3.20

- **HEALPix**-F90 routines and facilities can now also be compiled with the free Fortran95 compiler **g95** (www.g95.org)
- a separate `build` directory is used to store the objects, modules, ... produced during the compilation of the source codes
- improved handling of long FITS keywords, now producing FITS files fully compatible with the PyFITS and `Astropy` (https://www.astropy.org) `Python` libraries
- improved FITS file parsing in generate_beam, affecting the external $B(l)$ reading in the F90 facilities alteralm, synfast, sky_ng_sim, smoothing.

Version 3.11

- `libsharp` C routines used for Spherical Harmonics Transforms and introduced in **HEALPix** 3.10 can now be compiled with any `gcc` version.

Version 3.10

- all Fortran facilities now support most of `cfitsio`'s "Extended File Name Syntax" features, allowing the reading and processing of an arbitrary HDU and table column out of remote, compressed FITS files. For example, setting
  `infile = ftp://`*url/file.fits*`.gz[`*extn*`][col` *colname*`]`
  in anafast will download the FITS file *file.fits.gz* from *url*, uncompress it, open the HDU (extension) featuring keyword `EXTNAME=`*extn*, or the one with 1-based rank number *extn*, read the table column with `TTYPE*=`*colname* out of it and will analyze it.
  It is also possible to perform a remote anafast analysis of a Planck Legacy Archive (PLA) sky map named *map.fits* via the PLA AIO Subsystem by simply setting `infile=https://pla.esac.esa.int/pla/aio/product-action?MAP.MAP_ID=`*map.fits* as input map file.
- yet faster synfast, anafast, smoothing thanks to libsharp routines[1].
  Note that some gcc *versions (4.4.1 to 4.4.6) crash with an internal compiler error during compilation of* `libsharp`. The problem has been fixed in `gcc` 4.4.7, 4.5.*, 4.6.*, 4.7.* and newer versions and was not present in versions 4.2.* and 4.3.*.

Changes up to release 3.00

Version 3.00

- all *input* FITS files can now be compressed (with a `.gz`, `.Z`, `.z`, or `.zip` extension) and/or remotely located (with a `ftp://` or `http://` prefix). *Version 3.14 (March 2009) or newer of CFITSIO is required for **HEALPix** 3.0.*
- introduction of process_mask facility to compute the angular distance of valid pixels to the closest invalid pixels for a input binary mask,
- sky_ng_sim now allows the computation of the spatial derivatives of the non Gaussian map being produced, and the output of the $a_{\ell m}$ coefficients of that map,
- anafast now allows the pro/down-grading of the input mask to match the resolution of the map(s) being analyzed.

Version 2.20

- faster synfast, anafast, smoothing thanks to libpsht routines.
- most facilities can handle maps with $N_{\rm side} > 8192$, ie more than 805,306,368 pixels.

See "F90 Subroutines Overview" for details.

[1] To *revert* to the original F90 implementation of all these routines, the preprocessing variable `DONT_USE_SHARP` must be set during compilation.

Version 2.14

- In synfast facility, a numerical bug affecting the accuracy of the Stokes parameter derivatives $\partial X/\partial\theta$, $\partial^2 X/(\partial\theta\partial\phi\sin\theta)$, $\partial^2 X/\partial\theta^2$, for $X = Q, U$ has been corrected. See this appendix for details.

Version 2.10

- The anafast facility can now compute the cross-correlations of two different maps.
- The sky_ng_sim facility (Rocha et al, 2005), to produce non-Gaussian CMB temperature maps, has been added.

Version 2.00

- faster implementation of $a_{\ell m}$ related facilities, generalization of OpenMP parallelization, and availability of MPI parallelized routines (see mpi_* routines in **Fortran90 Subroutines Overview** document).
- introduction of alteralm facility to modify and/or rotate the spherical harmonics coefficients $a_{\ell m}$ and greater flexibility for constraining $a_{\ell m}$ in `synfast`
- single and double precision implementation of most facilities (see **Input and Output Precision** page 3)

alteralm

Location in HEALPix directory tree: src/f90/alteralm/alteralm.f90

This program can be used to modify a set of $a_{\ell m}$ spherical harmonics coefficients, as those extracted by anafast or simulated by synfast, before they are used as constraints on a synfast run. Currently the alterations possible are

- rotation (using Wigner matrices) of the $a_{\ell m}$ from the input coordinate system to any other standard astrophysical coordinate system. The resulting $a_{\ell m}$ can be used with e.g. synfast to generate a map in the new coordinate system.
- removal of the pixel and beam window functions of the input $a_{\ell m}$ (corresponding to the pixel size and beam shape of the map from which they were extracted) and implementation of an arbitrary pixel and beam window function.

$$a_{\ell m}^{\rm OUT} = a_{\ell m}^{\rm IN} \frac{B^{\rm OUT}(\ell) P^{\rm OUT}(\ell)}{B^{\rm IN}(\ell) P^{\rm IN}(\ell)}, \tag{3}$$

  where $P(\ell)$ is the pixel window function, and $B(\ell)$ is the beam window function (assuming a circular beam) or any other ℓ space filter (eg, Wiener filter). For an infinitely small pixel (or beam) one would have $P(\ell) = 1$ (resp. $B(\ell) = 1$) for any ℓ.

FORMAT

% alteralm [options] [parameter_file]

COMMAND LINE OPTIONS

`-d`
`--double` double precision mode (see Notes on double/single precision modes on page 3)

`-s`
`--single` single precision mode (default)

QUALIFIERS

infile_alms = Defines the FITS file from which to read the input $a_{\ell m}$.

outfile_alms = Defines the FITS file in which to write the altered $a_{\ell m}$.

fwhm_arcmin_in = Defines the FWHM size in arcminutes of the Gaussian beam present in the input $a_{\ell m}$. The output $a_{\ell m}$ will be corrected from it, see Eq. (3). (default= value of FWHM keyword in `infile_alms`).

beam_file_in = Defines the FITS file (see "Beam window function files" in introduction) describing the Legendre window function of the circular beam present in the input $a_{\ell m}$. The output $a_{\ell m}$ will be corrected from it, see Eq. (3). If set to an existing file name, it will override the `fhwm_arcmin_in` given above. (default= value of the BEAM_LEG keyword in `infile_alms`)

nlmax_out = Defines the maximum ℓ value to be used for the output $a_{\ell m}$s. (default= maximum ℓ of input $a_{\ell m}$ = value of MAX-LPOL keyword in `infile_alms`).

nsmax_in = If it can not be determined from the input file `infile_alms`, asks for the **HEALPix** resolution parameter $N_{\rm side}$ whose window function is applied to the input $a_{\ell m}$

nsmax_out = Defines the **HEALPix** resolution parameter $N_{\rm side}$ whose window function will be applied to the output $a_{\ell m}$. Could be set to 0 for infinitely small pixels, ie no pixel window function (default= same as input's $N_{\rm side}$).

fwhm_arcmin_out = Defines the FWHM size in arcminutes of the Gaussian beam to be applied to $a_{\ell m}$, see Eq. (3). (default= `fwhm_arcmin_in`).

beam_file_out = Defines the FITS file (see "Beam window function files" in introduction) describing the Legendre window function of the circular beam to be applied $a_{\ell m}$. If set to an existing file name, it will override the `fhwm_arcmin_out` given above. (default= " ")

coord_in = Defines astrophysical coordinates used to compute the input $a_{\ell m}$. Valid choices are 'G' = Galactic, 'E' = Ecliptic, 'C'/'Q' = Celestial = eQuatorial. (default = value of COORDSYS keyword read from input FITS file)

epoch_in =	Defines astronomical epoch of input coordinate system (default=2000)
coord_out =	Defines astrophysical coordinates into which to rotate the $a_{\ell m}$ (default = `coord_in`)
epoch_out =	Defines astronomical epoch of output coordinate system (default=`epoch_in`)
windowfile_in =	Defines the input filename from which to read the pixel window function parameterized by `nsmax_in` (default= pixel_window_n????.fits, see Notes on default files and directories on page 3)
winfiledir_in =	Defines the directory in which `windowfile_in` is located (default : see above).
windowfile_out =	Defines the input filename from which to read the pixel window function parameterized by `nsmax_out` (default= pixel_window_n????.fits, see above)
winfiledir_out =	Defines the directory in which `windowfile_out` is located (default : see above).

DESCRIPTION

Alteralm can modify temperature as well as polarisation $a_{\ell m}$. It will also modify the error on the $a_{\ell m}$ if those are provided. It works best if the input FITS file contains the relevant information on the beam size and shape, maximum multipoles, ...

DATASETS

The following datasets are involved in the **alteralm** processing.

Dataset	**Description**
/data/pixel_window_nxxxx.fits	Files containing pixel windows for various nsmax.

SUPPORT

This section lists those routines and facilities (including those *external* to the HEALPix distribution) which can assist in the utilisation of **alteralm**.

generate_beam — This **HEALPix** Fortran subroutine generates or reads the $B(\ell)$ window function(s) used in alteralm

anafast — This **HEALPix** Fortran facility can analyse a **HEALPix** map to extract the $a_{\ell m}$ that can be altered by alteralm.

synfast — This **HEALPix** facility can generate a **HEALPix** map from a power spectrum C_ℓ, with the possibility of including constraining $a_{\ell m}$ as those obtained with alteralm.

EXAMPLES: #1

```
alteralm
```

Alteralm runs in interactive mode, self-explanatory.

EXAMPLES: #2

```
alteralm filename
```

When 'filename' is present, alteralm enters the non-interactive mode and parses its inputs from the file 'filename'. This has the following structure: the first entry is a qualifier which announces to the parser which input immediately follows. If this input is omitted in the input file, the parser assumes the default value. If the equality sign is omitted, then the parser ignores the entry. In this way comments may also be included in the file. In this example, the file contains the following qualifiers:

```
infile_alms= alm.fits
nlmax_out= 512
fwhm_arcmin_out= 20.0
coord_out= G
outfile_alms= newalm.fits
```

Alteralm reads the $a_{\ell m}$ from 'alm.fits'. Since

```
nsmax_in
nsmax_out
fwhm_arcmin_in
beam_file_in
coord_in
epoch_in
epoch_out
windowfile_in
winfiledir_in
windowfile_out
winfiledir_out
```

have their default values, the pixel size will remain the same, the $a_{\ell m}$ will be corrected from its input beam (whatever it was, assuming the relevant information can be found), and a gaussian beam of 20.0 arcmin will be applied instead, the $a_{\ell m}$ will also be rotated from their original coordinate system (whatever it was, assuming the relevant information can be found) into Galactic coordinates, assuming a year 2000 epoch for both, and only the multipoles up to 512 will be written in 'newalm.fits'.

RELEASE NOTES

Revision 1: Initial release (**HEALPix** 2.00)

Messages

This section describes error messages generated by **alteralm**.

Message	Severity	Text
can not allocate memory for array xxx	Fatal	You do not have sufficient system resources to run this facility at the map resolution you required. Try a lower map resolution.
this is not a binary table		the fitsfile you have specified is not of the proper format
there are undefined values in the table!		the fitsfile you have specified is not of the proper format
the header in xxx is too long		the fitsfile you have specified is not of the proper format
XXX-keyword not found		the fitsfile you have specified is not of the proper format
found xxx in the file, expected:yyyy		the specified fitsfile does not contain the proper amount of data.
alteralm> no information found on input alms beam	Fatal	no information on the input beam was found, neither from parsing the FITS file header, nor from what the user provided.

anafast

Location in HEALPix directory tree: src/f90/anafast/anafast.f90

This program performs harmonic analysis of the **HEALPix** maps up to a user specified maximum spherical harmonic order $\ell_{\rm max}$. The integrals are computed on the whole sphere, unless the user chooses a provided option to excise from the input map(s) a simple, constant latitude, symmetric cut, and/or apply an arbitray cut read from an external file. Scalar, or scalar and tensor, spherical harmonic coefficients are evaluated from the map(s) if the input provides, respectively, only the temperature, or temperature and polarisation maps. The total operation count scales as $\mathcal{O}(N_{\rm pix}^{1/2}\ell_{\rm max}^2)$.

Anafast reads one (two) file(s) containing the map(s) and produces a file containing the temperature auto- (or cross-) power spectrum C_ℓ^{TT} and, if requested, also the polarisation power spectra C_ℓ^{EE}, C_ℓ^{BB}, C_ℓ^{TE}, C_ℓ^{TB}, C_ℓ^{EB} (as well as C_ℓ^{ET}, C_ℓ^{BT}, C_ℓ^{BE} if two maps are provided). The $a_{\ell m}$ coefficients computed during the execution also can be written to a (two) file(s) if requested.

Anafast executes an approximate, discrete point-set quadrature on a sphere sampled at the **HEALPix** pixel centers. Spherical harmonic transforms are computed using recurrence relations for Legendre polynomials on co-latitude, θ, and Fast Fourier Transforms on longitude, ϕ.

Anafast is provided with an option to use precomputed Legendre Polynomials; please note that since version 2.20 this will most likely reduce performance instead of increasing it.

Anafast permits two execution options which allow a significant improvement of accuracy of the approximate quadrature performed by this facility:

• Improved analyses using either the provided ring weights, which correct the quadrature on iso-latitude rings, or pixel-based weights which improve the quadrature on every pixels, and/or

• An iterative scheme using in succession several backward and forward harmonic transforms of the maps.

RECOMMENDATIONS FOR USERS

Execution of anafast requires a user to specify the maximum spherical harmonic order $\ell_{\max}$ up to which the harmonic decomposition of the input maps will be performed. Since there are no formal limits on parameter $\ell_{\max}$ enforced by anafast, the user should make his/her choices judiciously. Hereafter it is convenient to specify $\ell_{\max}$ in terms of the **HEALPix** map resolution parameter $N_{\rm side}$ (called *nsmax* in some other contexts). If the function to be analysed is strictly band-width limited, or nearly band-width limited (as in the case of a Gaussian beam smoothed signal discretized at a rate of a few pixels per beam area), it is sufficient to run anafast with $\ell_{\max} \approx 2{\cdot}N_{\rm side}$, with a very good C_ℓ error performance already in the raw (i.e. uncorrected quadrature) harmonic transform mode. If quadrature corrections are still desired in this case, it should be sufficient to use, at no extra cost in execution time, the ring-weighted quadrature scheme. This is the recommended mode of operation of anafast for essentially error and worry free typical applications, e.g. CPU-intensive Monte Carlo studies.
A new set of pixel-based quadrature weights was introduced in **HEALPix** 3.40. Pre-computed to inforce a (near) ideal integration of the spherical harmonics $Y_{\ell m}$ on the pixelized sphere (ie $\frac{4\pi}{N_{\rm pix}}\sum_p w(p)Y_{\ell m}(p) = \sqrt{4\pi}\delta_{\ell 0}\delta_{m0}$) for $|m| \le \ell \le 3N_{\rm side}$, they can be used to insure that the $a_{\ell m}$ and C_ℓ computed by anafast are perfectly accurate (almost to machine precision) without the need for iterations, but **only** for band-width limited input signal with $\ell_{\max} \le 1.5N_{\rm side}$.
If more aggressive attempts are undertaken to extract from a map the spectral coefficients at $\ell > 2 \cdot N_{\rm side}$ (for example, as in a possible case of an attempt to analyse an existing map, which was irreversibly binned at a suboptimal resolution) the following should be kept in mind:
• Spherical harmonics discretized using **HEALPix** (either sampled at pixel centers, or avaraged over pixel areas) form a linearly independent system up to $\ell_{\max} = 3{\cdot}N_{\rm side}-1$. Hence, the functions which are strictly band-width limited to $\ell_{\max} = 3{\cdot}N_{\rm side}-1$ can be fully spectrally resolved with anafast, albeit with integration errors in the uncorrected quadrature mode, which grow up to $\delta C_\ell \propto \epsilon{\cdot}C_\ell$, with $\epsilon < 0.1$, at the highest values of ℓ. These integration errors can be efficiently reduced using anafast in the iterative mode. Although this $\ell_{\max}$ range — $2 \cdot N_{\rm side} < \ell_{\max} < 3 \cdot N_{\rm side} - 1$ — is easily manageable with anafast used on strictly band-width limited functions, it should be used with caution in basic and automated applications, e.g. Monte Carlo simulations.
• As with any discrete Fourier transform, anafast application to functions which are not band-width limited results with aliasing of power, which can not be remedied. If the particular case of interest may result in such a band-width violation (i.e. there is significant power in the function at $\ell > 3 \cdot N_{\rm side} - 1$), the function should be smoothed before the application of anafast, or discretized and then analysed, on a refined **HEALPix** grid (with larger $N_{\rm side}$).
• REMEMBER: A peculiar property of the sphere, which usually surprises those whose intuition is built on experience with FFTs on a segment, or on a Euclidean

multidimensional domain, is the lack of a regular and uniform point-set at arbitrary resolution, and the resulting non-commutativity of the forward and backward discrete Fourier transforms on nearly-uniform point-sets, e.g. **HEALPix**. Hence, as in any case of attempting an extreme application of an off-the-shelf software, use caution and understand your problem well *before* executing anafast under such circumstances!

FORMAT % anafast [options] [parameter_file]

COMMAND LINE OPTIONS

`-d`
`--double` double precision mode (see Notes on double/single precision modes on page 3)

`-s`
`--single` single precision mode (default)

QUALIFIERS

infile = Defines the input map file. (default= map.fits) If not blank, the filename should *never* be put between quotes even if it contains symbols such as `&`, `[`, `]`, `?`, `=` which should be typed literally (ie, unprotected). For instance `infile = http://site/action?file.fits[2][col FLUX]` is just fine.

infile2 = Defines the 2nd input map file, to be cross-correlated with the first one. The 2 maps should match in resolution ($N_{\rm side}$) and coordinate. (default= '', only the auto-correlation of the first map will be computed)

outfile = Defines the output file with the power spectrum. If only one input map is provided, `outfile` will contains its auto-spectra, if 2 maps are provided, `outfile` will contain their cross-spectra. Note in particular that in the latter case, the $C_l^{T\times E}$ power spectrum will be build from the T field of the 1st (possibly polarized) map, and the E field of the second polarized map. (default= cl_out.fits)

simul_type = Defines which map(s) to analyse, 1=temperature only, 2=temperature AND polarisation. (default= 1)

nlmax = Defines the maximum ℓ value to be used. See the Recommendations for Users. (default= 64)

maskfile = Defines the FITS file containing the pixel mask(s) or weighting scheme(s) by which the map(s) read from `infile` will be multiplied before analysis. If the file contains several fields, the first one in which at least one pixel is non-zero will be used. This option can be used to, for instance, apply the WMAP Kp intensity mask to the data (see https://lambda.gsfc.nasa.gov), but it will *not* handle the WMAP composite mask correctly. Can be used in conjonction with `theta_cut_deg`. Masked or weighted pixels will be correctly accounted when performing the monopole and dipole regression.
Note: The mask's resolution ($N_{\rm side}$) and ordering can be different from the input map(s) one's, and the mask will be pro/down-graded and reordered to match the map. On the other hand, the mask and maps coordinates will always be presumed to match (ie, no attempt of rotation of the mask will be made). (default= '': no mask, all valid pixels are used)

theta_cut_deg = Defines the latitude (in degrees) of an optional, straight symmetric cut around the equator. Pixels located within that cut ($|b|$ <`theta_cut_deg`) are ignored. (default= 0°: no cut)

iter_order = Defines the maximum order of quadrature iteration to be used. (default=0, no iteration). For details, see the `map2alm_iterative` routine described in the "Fortran Subroutines" document. iter_order>0 can *not* be used together with won=2.

outfile_alms = Defines the name of the file containing the $a_{\ell m}$ coefficients of the first map which can be written optionally. (default= no entry — $a_{\ell m}$s are not written to a file)

outfile_alms2 = Defines the name of the file containing the $a_{\ell m}$ coefficients of the second map, if any, which can

	be written optionally. (default= no entry — $a_{\ell m}$s are not written to a file)
plmfile =	Defines the name for an input file containing precomputed Legendre polynomials $P_{\ell m}$. (default= no entry — anafast executes the recursive evaluation of $P_{\ell m}$s)
w8file =	Defines name for an input file containing ring-based or pixel-based weights (depending on the value of won) in the improved quadrature mode (default= no entry — see "Default file names and directories" in introduction)
w8filedir =	Gives the directory where the ring weight files are to be found (default= no entry — anafast searches the default directories, see "Default file names and directories" in introduction)
won =	Set this to 1 if ring-based quadrature weight files are to be used, or to 2 to use pixel-based weight files instead; otherwise set it to 0. (default= 0). won=2 can *not* be used together with iter_order> 0.
regression =	Sets the degree of the regression made on the input map before doing the power spectrum analysis. The regression is a minimal variance fit (assuming a uniform noise) made on valid (unflagged and unmasked) pixels, out of the symmetric cut (if any). In case of cut sky analysis, such a regression reduces the monopole and dipole leakage to higher ℓ's. 0 : no regression, does the $a\_\ell m$ analysis on the raw map 1 : removes the best fit monopole first 2 : removes the best fit monopole and dipole first default = 0.

DESCRIPTION

Anafast reads one or two binary FITS-files containing a **HEALPix** map. These files can each contain a temperature map or both temperature and polarisation (Q,U) maps. Anafast analyses the map(s) and makes an output ascii-FITS file containing the angular auto or cross power spectra C_ℓ^{TT}s (and C_ℓ^{EE}, C_ℓ^{BB}, C_ℓ^{TE}, C_ℓ^{TB} and C_ℓ^{EB} if specified, as well as C_ℓ^{ET}, C_ℓ^{BT} and C_ℓ^{BE} if two maps are provided). Here C_ℓ^{TE} is meant as the power spectrum built from the T field of the first (polarized) map, and the E field of the second polarized map, while it is the other way around for C_ℓ^{ET}. Anafast produces C_ℓs up to a specified maximum ℓ-value (see Recommendations for Users). If requested, the computed $a_{\ell m}$ coefficients can be written to a FITS file. This file can be used in the constrained realisation mode of synfast.

Anafast permits two execution modes that allow to improve the quadrature accuracy: (1) the ring weight corrected quadrature, and (2) the iterative scheme. Using the ring weights does not increase the execution time. The precomputed ring weights to be used for each **HEALPix** resolution $N_{\rm side}$ are provided in the `$HEALPIX/data` directory. The more sophisticated iterative scheme increases the accuracy more effectively than the weighted ring scheme, but its disadvantage is that the time for the analysis increases, 1 iteration takes 3 times as long, 2 iterations 5 times as long on so forth, since each order of iteration requires one more forward and backward transform.

The spherical harmonics evaluation uses a recurrence on associated Legendre polynomials $P_{\ell m}(\theta)$. This recurrence consumed most of the CPU time used by anafast up to version 2.15. We have therefore included an option to load precomputed values for the $P_{\ell m}(\theta)$ from a file generated by the **HEALPix** facility plmgen. Since the introduction of accelerated spherical harmonic transforms in **HEALPix** v2.20, this feature is obsolete and should no longer be used.

When dealing with polarized signal maps, the anafast behavior will depend on the value of the `POLCCONV` FITS keyword (see note on POLCCONV in The **HEALPix** Primer)

In version 3.50 a bug affecting previous versions of anafast has been fixed. (It occured previously when `iter_order`> 0 was used in conjonction with a `maskfile` and/or a restrictive `theta_cut_deg`, see `map2alm_iterative` for details). The result was correct when the mask (if any) was applied to the map prior to the anafast calling, or when no iteration was requested.

DATASETS

The following datasets are involved in the **anafast** processing.

Dataset	Description
data/weight_ring_n0xxxx.fits	Files containing ring weights for the anafast improved quadrature mode.

SUPPORT

This section lists those routines and facilities (including those *external* to the HEALPix distribution) which can assist in the utilisation of **anafast**.

synfast — This **HEALPix** facility can generate a map for analysis by anafast.

alteralm — This **HEALPix** Fortran facility can be used to modify the $a_{\ell m}$ extracted by anafast before they are used as constraints on a synfast run.

plmgen — This **HEALPix** facility can be used to generate precomputed Legendre polynomials.

EXAMPLES: #1

```
anafast
```

Anafast runs in interactive mode — self-explanatory.

EXAMPLES: #2

```
anafast filename
```

When 'filename' is present, anafast enters the non-interactive mode and parses its inputs from the file 'filename'. This has the following structure: the first entry is a qualifier which announces to the parser which input immediately follows. If this input is omitted in the input file, the parser assumes the default value. If the equality sign is omitted, then the parser ignores the entry. In this way comments may also be included in the file. In this example, the file contains the following qualifiers:

```
simul_type= 1
nlmax= 64
theta_cut_deg= 0
iter_order= 0
infile= map.fits
outfile= cl_out.fits
regression= 0
```

Anafast reads the map from *map.fits*, makes an analysis and produces C_l^Ts up to l=64. This powerspectrum is saved in the file *cl_out.fits*. No galactic cut is excised and no iterations are performed. As regressionis set to 0 (its default value) the map is analyzed as is, without prior best fit removal of the monopole nor the dipole.

Since

```
infile2
outfile_alms
outfile_alms2
w8file
w8filedir
plmfile
maskfile
```

were omitted, they take their default values (empty strings). This means that no file for precomputed Legendre polynomials is read, no second map is read, no mask is applied, and anafast does not save the $a_{\ell m}$ values from the analysis.

Also since

```
won
```

is not given, it takes its default value 0, which means that quadrature weights are not used.

RELEASE NOTES

Revision 1: Initial release (**HEALPix** 0.90)

Revision 2: Optional non-interactive operation. Proper FITS file support. Improved reccurence algorithm for $P_{\ell m}(\theta)$ which can compute to higher ℓ values. New functionality: arbitrary order of iterations, precomputed $P_{\ell m}$, dumping of $a_{\ell m}$. (**HEALPix** 1.00)

Revision 3: New functionality: possibility of removing the best fit monopole and dipole. New Parser. Can be linked to FFTW (**HEALPix** 1.20)

Revision 4: New functionality: addition of `maskfile` (**HEALPix** 2.0)

Revision 5: Bug correction: correct interaction of iterative scheme with masked pixels (**HEALPix** 2.01)

Revision 6: New functionality: cross-correlation of 2 maps; Correction of this documentation: the code expects `maskfile` and not `mask_file` (**HEALPix** 2.1)

Revision 7: Bug correction: now correctly supports mask pro/down-grading

Revision 8: Support for pixel-based quadrature weights when `won=2` (**HEALPix** 3.40)

Revision 9: Correction of a bug in `map2alm_iterative` (**HEALPix** 3.50)

Messages

This section describes error messages generated by **anafast**.

Message	Severity	Text
can not allocate memory for array xxx	Fatal	You do not have sufficient system resources to run this facility at the map resolution you required. Try a lower map resolution.

hotspot

Location in HEALPix directory tree: src/f90/hotspot/HotSpots.F90

This Fortran facility provides a means to find local extrema of a map in **HEALPix** format. It also serves to illustrate the use of the following parts of the **HEALPix** toolkit: fast neighbour and extrema finding in the nested scheme, in-place conversion between RING and NESTED pixel schemes

FORMAT

% hotspot

QUALIFIERS

infile =	Defines the input map file.
extrema_outfile =	Defines the output map file.
maxima_outfile =	Defines the output ascii list of maxima.
minima_outfile =	Defines the output ascii list of minima.

DESCRIPTION

hotspot reads a healpix map in FITS format and generates the following outputs: 1) a **HEALPix** map in FITS format which is zero everywhere, except at pixels which contain local extrema. These pixels have the same values as in the input map. 2) an ASCII file which lists the pixel numbers and values of maxima, and 3) an ASCII file which lists the pixel numbers and values of minima.

The facility can be used in both an interactive mode and a command mode, where command qualifiers are fed to the facility using an input file.

Note the following limitations: hotspot (and the toolkit neighbour finder which it uses) will only work on maps with $N_{\rm side} \geq 2$.

DATASETS

The following datasets are involved in the **hotspot** processing.

Dataset	Description
None required	

SUPPORT

This section lists those routines and facilities (including those *external* to the HEALPix distribution) which can assist in the utilisation of **hotspot**.

synfast	This **HEALPix** facility can generate a FITS format sky map to be input to hotspot.
map2gif	This **HEALPix** Fortran facility can be used to visualise the output map.
mollview	This **HEALPix** IDL facility can be used to visualise the output map.

EXAMPLES: #1

```
hotspot
```

hotspot runs in interactive mode.

EXAMPLES: #2

```
hotspot filename
```

When 'filename' is present, hotspot enters the non-interactive mode and parses its inputs from the file 'filename'. This has the following structure: the first entry is a qualifier which announces to the parser which input immediately follows. If this input is omitted in the input file, the parser assumes the default value shown below. If the equality sign is omitted, then the parser ignores the entry. In this way comments may also be included in the file. In this example, the file contains the following qualifiers:

```
infile= map.fits
extrema_outfile= pixlminmax.fits
maxima_outfile= maxima.dat
minima_outfile= minima.dat
```

hotspot reads in the map 'map.fits' and generates an output map with name 'pixlminmax.fits', and two ASCII files, 'maxima.dat' and 'minima.dat'.

RELEASE NOTES

Revision 1: Initial release (**HEALPix** 0.90)

Revision 2: Optional non-interactive operation. Proper FITS file support for input and output maps. (**HEALPix** 1.00)

Messages

This section describes error messages generated by **hotspot**.

Message	Severity	Text
can not allocate memory for array xxx	Fatal	You do not have sufficient system resources to run this facility at the map resolution you required. Try a lower map resolution.

map2gif

Location in HEALPix directory tree: src/f90/map2gif/map2gif.f90

This Fortran facility provides a means to generate a gif image from an input **HEALPix** sky map. It is intended to allow some primitive visualisation for those with limited or no access to IDL or python. It is also useful for image generation in a pipeline environment.

FORMAT

% map2gif -inp FITS_file -out GIF_file [options]

QUALIFIERS

-inp FITS_file	The file name of the input FITS sky map (**default:** none). In map2gif (and map2gif only) it may be necessary to put the file name between quotes if it contains symbols such as `&`, `[`, `]`, `?`, `=` or blanks
-out GIF_file	The file name of the output gif image (**default:** none). Prepending the output file name with `\!` (see example below) will allow the overwriting of the file if it already exists
-add offset	Real value to add to the signal before performing any other operation to it (like taking the logarithm etc.) (**default:** 0.0)
-ash flag	Logical to use the hyperbolic arc sine of the signal when plotting. Cannot be true when `-log` is true. (**default:** .false.)
-bar flag	Logical which determines whether a color bar is displayed (**default:** .false.).
-col table	The number of the color table to utilise (in [1,5]) (**default:** 5).
-hlp	Print on-line help for the facility.
-lat lat0	Latitude (Deg) of central point (**default:** 0)
-log flag	Logical to use the log of the signal when plotting (**default:** .false.)
-lon lon0	Longitude (Deg) of central point (**default:** 0)

-max maxval	Set the maximum value for the plotted signal (**default:** is to use the actual signal maximum).
-min minval	Set the minimum value for the plotted signal (**default:** is to use the actual signal minimum).
-mul factor	Real value to multiply the signal with directly after adding the offset (see above). (**default:** 1.0)
-pro projection	Select the projection scheme (among `mol` or `MOL` for Mollweide and `gno` or `GNO` for gnomic) (**default:** MOL).
-res reso	Resolution in projected plan in arcmin (only for gnomic projection) (**default:** 2)
-sig number	The identifier of the signal to plot: for a polarisation map, then the mapping is 1 = I; 2 = Q; 3 = U (**default:** 1).
-ttl title	A string specifying the title for the plot (see note on quotes for `-inp`) (**default:** none).
-xsz xsize	The x-dimension of the image in pixels (**default:** 800).

DESCRIPTION

map2gif reads in a **HEALPix** sky map in FITS format and generates an image in GIF format. map2gif allows the selection of the projection scheme (Mollweide for the full sky or Gnomonic for small patches of the sky), color table, color bar inclusion, linear or log scaling, maximum and minimum range for the plot and plot-title. Flagged, undefined and NaN-valued input pixels will take a grey color on output. The facility utilises a command-line interface.

DATASETS

The following datasets are involved in the **map2gif** processing.

Dataset	Description
None required	

SUPPORT

This section lists those routines and facilities (including those *external* to the HEALPix distribution) which can assist in the utilisation of **map2gif**.

display, open	a facility is required to view the gif image generated by map2gif (a browser can also be used).
synfast	This **HEALPix** facility will generate the FITS format sky map to be input to map2gif.

EXAMPLES: #1

```
map2gif  -inp planck100GHZ-LFI.fits \
         -out \!planck100GHZ-LFI.gif \
         -bar .true.  \
         -min -100 \
         -max 100 \
         -ttl 'Simulated Planck LFI Sky Map at 100GHz'
```

map2gif reads in the map 'planck100GHZ-LFI.fits' and outputs its Mollweide projection in a gif image with name 'planck100GHZ-LFI.gif' (overwriting it if necessary) in which the temperature scale has been set to lie between ± 100 (μK), a color bar has been drawn and the title 'Simulated Planck LFI Sky Map at 100GHz' appended to the image.

RELEASE NOTES

Revision 1: Initial release (**HEALPix** 1.00)

Messages

This section describes error messages generated by **map2gif**.

Message	Severity	Text
None at present		

median_filter

Location in HEALPix directory tree: src/f90/median_filter/median_filter.f90

This program produces the median filtered map of an input **HEALPix** map (polarised or unpolarised). The neighborhood on the which the median is computed is defined as a disk of user-defined radius

FORMAT

% median_filter [options] [parameter_file]

COMMAND LINE OPTIONS

`-d`

`--double` double precision mode (see Notes on double/single precision modes on page 3)

`-s`

`--single` single precision mode (default)

QUALIFIERS

simul_type = Either 1 or 2. If set to 1, only the temperature component of the input map will be filtered. If set to 2, all the Stokes components available in the input file will be filtered (default = 1)

infile = Name of the FITS file containing the map to be filtered (default = '', no default input file).

mf_radius_arcmin = Radius in arcmin of the disk over which the median is computed (default = $3\theta_{\rm pix}$ where $\theta_{\rm pix}$ is the input map pixel size).

fill_holes = If set to true, flagged pixels take for value the median of the valid pixels surrounding them (if any). Otherwise they are left unchanged. (default = .false.). Note that `y, yes,t, true, .true.` and 1 are interpreted as *true*, while `n, no, f, false, .false.` and 0 stand for *false*.

mffile = Name of the FITS file containing the median filtered map

DESCRIPTION

Median_Filter produces a median filtered map in which the value of each pixel is the median of the input map valid pixels found within a disk of given radius centered on that pixel. A pixel flagged as 'non-valid' in the input map can either be left unchanged or 'filled in' with the same scheme, if at least one valid pixel is found among its neighbors.
If the map is polarized, each of the three Stokes components is filtered separately.

SUPPORT

This section lists those routines and facilities (including those *external* to the HEALPix distribution) which can assist in the utilisation of **median_filter**.

anafast — This **HEALPix** Fortran facility can analyse a **HEALPix** map.

synfast — This **HEALPix** facility can generate a **HEALPix** map from a power spectrum C_ℓ.

EXAMPLES: #1

```
median_filter [option]
```

Median_Filter runs in interactive mode, self-explanatory.

EXAMPLES: #2

```
median_filter [option] filename
```

When 'filename' is present, median_filter enters the non-interactive mode and parses its inputs from the file 'filename'. This has the following structure: the first entry is a qualifier which announces to the parser which input immediately follows. If this input is omitted in the input file, the parser assumes the default value. If the equality sign is omitted, then the parser ignores the entry. In this way comments may also be included in the file. In this example, the file contains the following qualifiers:

```
simul_type= 1
infile= map.fits
mf_radius_arcmin= 20.0
mffile= med.fits
```

Median_Filter reads the sky map from 'map.fits'. Since `fill_holes` has its default value, ..., The median will be computed on a disk of 20 arcmin in radius, and the result will be written in 'med.fits'.

RELEASE NOTES

Revision 1: Initial release (**HEALPix** 2.00)

Messages

This section describes error messages generated by **median_filter**.

Message	**Severity**	**Text**
can not allocate memory for array xxx	Fatal	You do not have sufficient system resources to run this facility at the map resolution you required. Try a lower map resolution.
this is not a binary table		the fitsfile you have specified is not of the proper format
there are undefined values in the table!		the fitsfile you have specified is not of the proper format
the header in xxx is too long		the fitsfile you have specified is not of the proper format
XXX-keyword not found		the fitsfile you have specified is not of the proper format
found xxx in the file, expected:yyyy		the specified fitsfile does not contain the proper amount of data.

plmgen

Location in HEALPix directory tree: src/f90/plmgen/plmgen.f90

This program can be used to create a file containing the pre-computed values of the associated Legendre polynomials $P_{lm}(\theta)$ (and, if requested, of the tensor spherical harmonics) for faster execution of the **HEALPix** map analysis/synthesis. The map resolution parameter, *nsmax*, and the maximum value of the spherical harmonic order ℓ_{max} must be specified.

Note: Since the introduction of optimized spherical harmonic transforms in HEALPix v2.20, this code has become obsolete and should no longer be used.

FORMAT

% plmgen

QUALIFIERS

nsmax =	Defines the resolution parameter for the map to be analysed/synthesized with the precomputed harmonics. (default= 32)
nlmax =	Defines the ℓ_{max} value for the execution. (default= 64)
simul_type =	Defines whether only scalar, or scalar and tensor harmonics are to be precomputed, 1=scalar only, 2=scalar AND tensor. (default=1)
outfile =	Defines the name for the file that will contain the precomputed harmonics. (default='plm.fits')

DESCRIPTION

The recursion of Legendre polynomials and tensor harmonics during the analysis and synthesis of **HEALPix** maps can be time consuming. Especially when repetitive applications are desired there is no need to compute the recursions every time. For such applications the values of $P_{\ell m}(\theta)$ can be precomputed with plmgen and stored in a file. When using synfast or anafast this file can be read in to shorten the analysis/synthesis execution time.

The memory (and disc) consumption of plmgen is $8 N_\lambda N_p$ bytes, with $N_\lambda = \texttt{nsmax}(\texttt{nlmax}+1)(\texttt{nlmax}+2)$ and N_p is either 1 or 3, depending whether tensor harmonics are computed.

Currently an extra limitation $N_\lambda < 2^{31} = 2147483648$ also applies, corresponding to, eg, $\texttt{lmax} \leq 1446$ for $\texttt{nsmax} = 1024$.

DATASETS

The following datasets are involved in the **plmgen** processing.

Dataset	**Description**
None required	

SUPPORT

This section lists those routines and facilities (including those *external* to the HEALPix distribution) which can assist in the utilisation of **plmgen**.

synfast	This **HEALPix** facility can generate a map using precomputed harmonics made from plmgen.
anafast	This **HEALPix** facility can analyse a map using precomputed harmonics.
plm_gen	Fortran subroutine used to generate the harmonics

EXAMPLES: #1

```
plmgen
```

plmgen runs in interactive mode, self-explanatory.

EXAMPLES: #2

```
plmgen filename
```

When ‘filename’ is present, plmgen enters the non-interactive mode and parses its inputs from the file ‘filename’. This has the following structure: the first entry is a qualifier which announces to the parser which input immediately follows. If this input is omitted in the input file, the parser assumes the default value. If the equality sign is omitted, then the parser ignores the entry. In this way comments may also be included in the file. In this example, the file contains the following qualifiers:

```
simul_type= 1
nsmax= 32
nlmax= 86
outfile= plm.fits
```

Creates a binary FITS file called 'plm.fits' containing Legendre polynomials up to ℓ and m values of 86 for a nsmax= 32 map. Legendre polynomials for all ℓ and m values for each angle θ corresponding to all of the **HEALPix** pixel center rings will be created.

RELEASE NOTES

Revision 1: Initial release **HEALPix** 1.00

Messages

This section describes error messages generated by **plmgen**.

Message	Severity	Text
can not allocate memory for array xxx	Fatal	You do not have sufficient system resources to run this facility at the map resolution you required. Try a lower map resolution.
Error: these values of Nside and l_max ... are too large ...	Fatal	You are exceeding the limitation on Nside and l_max. Try a lower l_max.

process_mask

Location in HEALPix directory tree: **src/f90/process_mask/process_mask.F90**

This code can be used to modify a binary mask by removing small clusters of bad or invalid pixels (hereafter 'holes') and by computing the distance of each valid pixel to the closest invalid one, with the purpose of, for instance, defining a new apodized mask

FORMAT

% process_mask [parameter_file]

QUALIFIERS

mask_file =	Input binary mask FITS file
hole_min_size =	Minimal size (in pixels) of invalid regions to be kept (can be used together with hole_min_surf_arcmin2 below, the result will be the largest of the two). (**default:** 0)
hole_min_surf_arcmin2 =	Minimal surface area (in arcmin^2) of invalid regions to be kept (can be used together with hole_min_size above, the result will be the largest of the two). (**default:** 0.0)
filled_file =	Optional output FITS file to contain mask with filled-in small holes (as defined above). (**default:** ", no output file)
distance_file =	Optional output FITS file to contain angular distance (in radians) from valid pixel to the closest invalid one. (**default:** ", no output file)

DESCRIPTION

For a given input binary mask, in which pixels have either value 0 (=invalid) or 1 (=valid), this code produces a map containing for each valid pixel, its distance (in Radians, measured between pixel centers) to the closest invalid pixel.

This distance map can then be used to define an apodized mask.

Two pixels are considered adjacent if they have at least *one point* in common (eg, a pixel corner or a pixel side).

It is possible to treat small holes (=cluster of adjacent invalid pixels) as valid, by specifying a minimal number of pixels and/or minimal surface area (whichever is the largest), and the resulting new mask can be output.

The output FITS files have the same ordering as the input mask (even though the processing is done in NESTED ordering).

The algorithmic complexity of the distance calculation is expected to scale like $\propto N_{\rm pix}^{p} \propto N_{\rm side}^{2p}$ with p in $[1.5, 2]$ depending on the mask topology, even though the code has been optimized to reduce the number of calculations by a factor 10^2 to 10^3 compared to a naive implementation, and the most computationally intensive loops are parallelized with OpenMP. On a 3.06GHz Intel Core 2 Duo, the distances on a $N_{\rm side} = 512$ Galactic + Point sources mask can be computed in a few seconds, while a similar $N_{\rm side} = 2048$ mask takes a minute or less to process. For totally arbitrary masks though, the return times can be multiplied by as much as 10.

SUPPORT

This section lists those routines and facilities (including those *external* to the HEALPix distribution) which can assist in the utilisation of **process_mask**.

mollview	IDL routine to view the input and output masks and the angular distance map.
mask_tools	F90 module containing the routines `dist2holes_nest`, `fill_holes_nest`, `maskborder_nest`, `size_holes_nest` used in process_mask and described in the "Fortran Subroutines" document

EXAMPLES: #1

```
process_mask
```

process_mask runs in interactive mode, self-explanatory.

EXAMPLES: #2

```
process_mask filename
```

When 'filename' is present, process_mask enters the non-interactive mode and parses its inputs from the file 'filename'. This has the following structure: the first entry is a qualifier which announces to the parser which input immediately follows. If this input is omitted in the input file, the parser assumes the default value. If the equality sign is omitted, then the parser ignores the entry. In this way comments may also be included in the file. In this example, the file contains the following qualifiers:

```
mask_file= wmap_temperature_analysis_mask_r9_5yr_v3.fits
hole_min_size= 100
distance_file= !/tmp/dist_wmap.fits
```

process_mask computes the distance in Radians from each valid pixel to the closest invalid pixel for WMAP-5 mask 'wmap_temperature_analysis_mask_r9_5yr_v3.fits', ignoring the holes containing fewer than 100 pixels, and outputs the result in '/tmp/dist_wmap.fits'.

RELEASE NOTES

Revision 1: (Initial release **HEALPix** 3.00)

sky_ng_sim

Location in HEALPix directory tree: **src/f90/ngsims_full_sky/sky_ng_sim.F90**

This program can be used to create temperature **HEALPix** maps computed as realisations of random *Non-Gaussian* fields on a sphere (either even power of a Gaussian distribution, or Simple Harmonics Oscillator PDF, see Description section for details).

It is directly adapted from the NGSIMS code described in Rocha et al, MNRAS, 357, 1 (2005)

The operation count is dominated by a term scaling as $\mathcal{O}(N_{\rm pix}^{1/2}\ell_{\rm max}^2)$. The map angular power spectrum, resolution, Gaussian beam FWHM or arbitrary beam window and random seed for the simulation can be selected by the user.

FORMAT

% sky_ng_sim [parameter_file]

QUALIFIERS

simul_type = Defines the simulation type, 1=temperature map only, 3=temperature and its first spatial derivatives, 4=temperature and its first and second spatial derivatives. (default= 1).

infile = Defines the input power spectrum file, (default= HEALPIX/test/cl.fits).

outfile_alms = Defines the FITS file in which to output $a_{\ell m}$ used for the simulation (default= ‘’)

outfile = Defines the output map file, (default= test.fits).

nsmax = Defines the resolution of the map. (default= 32)

nlmax = Defines the maximum ℓ value to be used in the simulation. WARNING: ℓ_{max} can not exceed the value 4· `nsmax`, because the coefficients of the average Fourier pixel window functions are precomputed and provided up to this limit. (default= 2*`nsmax`)

fwhm_arcmin = Defines the FWHM size in arcminutes of the simulated Gaussian beam. (default= 0.0)

beam_file = Defines the FITS file describing the Legendre window function of the circular beam to be used for the simulation. If set to an existing file name, it will override the `fhwm_arcmin` given above. (default=‘’)

windowfile = Defines the input filename for the pixel smoothing windows (default= pixel_window_n????.fits, see Notes on default files and directories on page 3)

winfiledir = Defines the directory in which windowfile is located (default : see Notes on default files and directories on page 3)

iseed = Defines the seed of the pseudo-random sequence to be used for the generation of the non-gaussian white noise (default= 1)

plot = If sky_ng_sim was linked with the PGPLOT library during compilation, and if `plot` is set to (case unsensitive) `.true.`, `t`, `yes`, `y` or 1, then the histogram of the simulated non-gaussian is produced, overplotted with the theoretical PDF; the histogram of the final map pixel values, overplotted with a PDF of a gaussian of same mean and variance is subsequently produced. (default=.false.)

pdf_choice=1 Choice of non-Gaussian PDF to use: 1= Simple Harmonics oscillator (see Eq 4 below)

sigma0= RMS of oscillator ground state

na= Integer in {0, 20}. Number of α coefficients to be given (default=3). Note: analytical calculation of the PDF moments can only be done for `na`≤ 3.

alpha_1=, alpha_2=, ... Real values of α_i coefficients for i in $[1, \texttt{na}]$

pdf_choice=2 Choice of non-Gaussian PDF to use: 2=Power of a Gaussian (see Eq 5 below)

npower = Positive integer in {1,4} (default=1). The gaussian will be set to the power 2*`npower`.

DESCRIPTION

A random non-Gaussian white noise map is generated, using either

- a simple linear harmonic oscillator, where the PDF of the pixel temperature t is

$$\rho_{\rm SHO}(t) = |\psi_n|^2 = e^{-t^2/2\sigma_0^2} \left| \sum_{i=0}^{n} \alpha_i C_i H_i \left(\frac{t}{\sqrt{2}\sigma_0} \right) \right|^2 \qquad (4)$$

  where H_i are the Hermite polynomials, C_i their normalization constants, σ_0^2 the variance of the (Gaussian) ground state $|\psi_0|^2$, α_i for $i \geq 1$ are free parameters, while $\alpha_0 = (1 - \sum_i^n |\alpha_i|^2)^{1/2}$ is constrained;

- or, an even power of a gaussian PDF, where the temperature of pixel q is

$$t_q = g_q^{2P} \qquad (5)$$

  where g is a zero mean, unit variance Gaussian variable, and P is a user-defined positive integer.

The resulting map is analyzed into its $a_{\ell m}$ coefficients, which are then multiplied by the beam, pixel and spectrum window

$$a_{\ell m} \longrightarrow a_{\ell m} \left[C(l)\right]^{1/2} B(\ell) w_{\rm pix}(\ell)$$

The resulting $a_{\ell m}$ coefficients are turned back into a map, which is therefore non-gaussian, with an effective angular power spectrum $C(\ell)B(\ell)^2 w_{\rm pix}(\ell)^2$ (Rocha et al, 2005).
Notes: the code has been modified from the original NGSIMS package in several respects: the seed parameter is named `iseed` instead of `idum`, to be consistent with other **HEALPix** simulation codes; and the SHO generator has been dramatically sped up, without loss of accuracy. Moreover, just like in `synfast` facility, it is now possible to output the $a_{\ell m}$ coefficients being used (`outfile_alms` option), and the spatial derivatives of the final map can also be output (`simul_type` option).

DATASETS

The following datasets are involved in the **sky_ng_sim** processing.

Dataset	Description
/data/pixel_window_nxxxx.fits	Files containing pixel windows for various nsmax.

SUPPORT

This section lists those routines and facilities (including those *external* to the HEALPix distribution) which can assist in the utilisation of **sky_ng_sim**.

generate_beam
: This **HEALPix** Fortran subroutine generates or reads the $B(\ell)$ window function used in sky_ng_sim

map2gif
: This **HEALPix** Fortran facility can be used to visualise the output map.

mollview
: This **HEALPix** IDL facility can be used to visualise the output map.

anafast
: This **HEALPix** Fortran facility can analyse a **HEALPix** map and save the $a_{\ell m}$ and C_ℓs to be read by sky_ng_sim.

EXAMPLES: #1

```
sky_ng_sim
```

sky_ng_sim runs in interactive mode, self-explanatory.

EXAMPLES: #2

```
sky_ng_sim filename
```

When 'filename' is present, `sky_ng_sim` enters the non-interactive mode and parses its inputs from the file 'filename'. This has the following structure: the first entry is a qualifier which announces to the parser which input immediately follows. If this input is omitted in the input file, the parser assumes the default value. If the equality sign is omitted, then the parser ignores the entry. In this way comments may also be included in the file. In this example, the file contains the following qualifiers:

```
simul_type= 1
nsmax= 128
nlmax= 256
fwhm_arcmin= 30.0
infile= cl.fits
pdf_choice= 1
iseed= 1
na= 3
sigma0= 1.0
alpha_1= 0.0
alpha_2= 0.0
alpha_3= 0.2
outfile= !test_ngfs.fits
```

`sky_ng_sim` reads in the C_ℓ power spectrum in 'cl.fits' up to $\ell = 256$, and produces the map 'map.fits' which has $N_{\rm side} = 128$. The non-gaussian white noise was generated assuming a SHO PDF (see Eq 4 above) with $\sigma_0 = 1$ and $\alpha_i = (0, 0, 0.2)$. The map is convolved with a beam of FWHM 30.0 arcminutes. The iseed= 1 sets the random seed for the realisation. A different iseedwould have given a different realisation from the same power spectrum. And finally, since `simul_type= 1` only the map (and not its spatial derivatives) will be output.

Since

```
beam_file
windowfile
outfile_alms
```

were omitted, they take their default values (empty strings). This means respectively that no beam were read, that `sky_ng_sim` attempts to find the pixel window files in the default directories (see page 3), and that the $a_{\ell m}$ generated and used to produce the map were not output.

RELEASE NOTES

Revision 1: Initial release (**HEALPix** 2.10)

Messages

This section describes error messages generated by **sky_ng_sim**.

Message	Severity	Text
can not allocate memory for array xxx	Fatal	You do not have sufficient system resources to run this facility at the map resolution you required. Try a lower map resolution.
this is not a binary table		the fitsfile you have specified is not of the proper format
there are undefined values in the table!		the fitsfile you have specified is not of the proper format
the header in xxx is too long		the fitsfile you have specified is not of the proper format
XXX-keyword not found		the fitsfile you have specified is not of the proper format
found xxx in the file, expected:yyyy		the specified fitsfile does not contain the proper amount of data.

smoothing

Location in HEALPix directory tree: src/f90/smoothing/smoothing.f90

This program can be used to convolve a map with a gaussian beam. The input map can be given in RING or NESTED scheme and the smoothed map is written to a FITS file in the RING scheme.
NOTE: This automated facility is susceptible to problems with non-commutativity of discrete spherical harmonics transforms, described in the Recommendations for Users of the anafast facility. If very high accuracy of the results is required in the spectral regime of $\ell > 2 \cdot nsmax$, it is recommended to choose an iterative computation of the $a_{\ell m}$ coefficients.

FORMAT

% smoothing [options] [parameter_file]

COMMAND LINE OPTIONS

`-d`
`--double` double precision mode (see Notes on double/single precision modes on page 3)

`-s`
`--single` single precision mode (default)

QUALIFIERS

simul_type = Defines which map(s) to analyse, 1=temperature only, 2=temperature AND polarisation. (default= 1)

infile = Defines the filename for the FITS file containing the map to be smoothed. (default= 'map.fits')

nlmax = Defines the $\ell_{\max}$ value for the application. (default= 64)

iter_order = Defines the maximum order of quadrature iteration to be used. (default=0, no iteration). For details, see the `map2alm_iterative` routine described in the "Fortran Subroutines" document.

fwhm_arcmin = Defines the FWHM in arcminutes of the gaussian beam for the convolution. (default=10)

beam_file = Defines the FITS file describing the Legendre window function of the circular beam to be used for the simulation (see "Beam window function files" in introduction). If set to an existing file name, it will override the `fhwm_arcmin` given above. default=''

outfile = Defines the filename for the file that will contain the smoothed map. (default='map_smoothed.fits')

plmfile = Defines the name for an input file containing precomputed Legendre polynomials $P_{\ell m}$. (default= no entry — smoothing executes the recursive evaluation of $P_{\ell m}$s)

w8file = Defines name for an input file containing ring weights in the improved quadrature mode (default= no entry — the name is assumed to be 'weight_ring_n0xxxx.fits' where xxxx is nsmax)

w8filedir = Gives the directory where the weight files are to be found (default= no entry — smoothing searches in the default directories, see "Default file names and directories" in introduction)

won = Set this to 1 if ring-based quadrature weight files are to be used, or to 2 to use pixel-based weight files instead; otherwise set it to 0. (default= 0)

DESCRIPTION

A FITS file containing a **HEALPix** map in RING or NESTED scheme is read in. When dealing with polarized signal maps, the smoothing behavior will depend on the value of the `POLCCONV` FITS keyword (see note on POLCCONV in The **HEALPix** Primer). The map is analysed and smoothed in fourier space with a gaussian beam of a given FHWM. A new map is then synthesized using the smoothed $a_{\ell m}$ coefficients. For a more accurate application, an iteration of arbitrary order can be applied. The output map is stored in *the same scheme* as the input map.

DATASETS

The following datasets are involved in the **smoothing** processing.

Dataset	Description
data/weight_ring_n0xxxx.fits	Files containing ring weights for the smoothing improved quadrature mode.

SUPPORT

This section lists those routines and facilities (including those *external* to the HEALPix distribution) which can assist in the utilisation of **smoothing**.

generate_beam — This **HEALPix** Fortran subroutine generates or reads the $B(\ell)$ window function used in smoothing

map2gif — This **HEALPix** Fortran facility can be used to visualise the input and output maps of smoothing.

mollview — This **HEALPix** IDL facility can be used to visualise the input and output maps of smoothing.

synfast — This **HEALPix** facility can generate a map and also do the smoothing.

anafast — This **HEALPix** facility can analyse a smoothed map.

EXAMPLES: #1

```
smoothing
```

Smoothing runs in interactive mode, self-explanatory.

EXAMPLES: #2

```
smoothing filename
```

When 'filename' is present, smoothing enters the non-interactive mode and parses its inputs from the file 'filename'. This has the following structure: the first entry is a qualifier which announces to the parser which input immediately follows. If this input is omitted in the input file, the parser assumes the default value. If the equality sign is omitted, then the parser ignores the entry. In this way comments may also be included in the file. In this example, the file contains the following qualifiers:

```
simul_type= 1
nlmax= 64
infile= map.fits
outfile= map_smoothed.fits
fwhm_arcmin= 10.
iter_order= 1
```

smoothes the **HEALPix** temperature map contained in 'map.fits' with a 10 arcmin FWHM beam. The resulting map is saved in 'map_smoothed.fits'. The map analysis/synthesis was carried out using fourier coeffecients up to an ℓ value of 64. A first order iteration of the quadrature was performed.

RELEASE NOTES

Revision 1: Initial release (**HEALPix** 0.90)

Revision 2: Extension to polarization and arbitrary *circular* beams (**HEALPix** 1.20)

Revision 3: Support for pixel-based quadrature weights when won=2 (**HEALPix** 3.40)

Messages

This section describes error messages generated by **smoothing**.

Message	Severity	Text
can not allocate memory for array xxx	Fatal	You do not have sufficient system resources to run this facility at the map resolution you required. Try a lower map resolution.

synfast

Location in HEALPix directory tree: src/f90/synfast/synfast.f90

This program can be used to create **HEALPix** maps (temperature only or temperature and polarisation) computed as realisations of random Gaussian fields on a sphere characterized by the user provided theoretical power spectra, or as constrained realisations of such fields characterised by the user provided $a_{\ell m}$ coefficients and/or power spectra. Total operation count scales as $\mathcal{O}(N_{\rm pix}^{1/2}\ell_{\rm max}^2)$ where $N_{\rm pix}$ is the total number of pixels and $\ell_{\rm max}$ is the limiting spherical harmonics order. The map resolution, Gaussian beam FWHM, and random seed for the simulation can be selected by the user. Spherical harmonics are either generated using the recurrence relations during the execution of spectral synthesis, or precomputed and read in before the synthesis is executed. The latter is no longer recommended since it provides no acceleration since the introduction of optimized algorithms in **HEALPix** v2.20.

FORMAT

% synfast [options] [parameter_file]

COMMAND LINE OPTIONS

`-d`
`--double` double precision mode (see Notes on double/single precision modes on page 3)

`-s`
`--single` single precision mode (default)

QUALIFIERS

infile = Defines the input power spectrum file, (default= cl.fits). Note that `infile` is now optional : synfast can run even if only `almsfile` is provided.

outfile = Defines the output (RING ordered) map file, (default= map.fits). Note that `outfile` is now optional: if it set to '' (empty string), mo map is synthesized but the $a_{\ell m}$ generated can be output.

outfile_alms = Defines the FITS file in which to output $a_{\ell m}$ used for the simulation (default= '')

simul_type = Defines the simulation type, 1=temperature only (1 field), 2=temperature+polarisation (3 fields), 3=temperature and its first spatial derivatives (3 fields), 4=temperature and its first and second spatial derivatives (6 fields), 5=temperature and polarisation, and their first derivatives (9 fields), 6=same as 5 plus the second derivatives of (T,Q,U) (18 fields). (default= 1).

nsmax = Defines the resolution of the map. (default= 32)

nlmax = Defines the maximum ℓ value to be used in the simulation. WARNING: $\ell_{\rm max}$ can not exceed the value $4\cdot$ **nsmax**, because the coefficients of the average Fourier pixel window functions are precomputed and provided up to this limit. (default= 64)

iseed = Defines the random seed to be used for the generation of $a_{\ell m}$s from the power spectrum. (default= -1)

fwhm_arcmin = Defines the FWHM size in arcminutes of the simulated Gaussian beam. (default= 420.0)

beam_file = Defines the FITS file (see "Beam window function files" in introduction) describing the Legendre window function of the circular beam to be used for the simulation. If set to an existing file name, it will override the `fhwm_arcmin` given above. (default='')

almsfile = Defines the input filename for a file containing $a_{\ell m}$s for constrained realisations. (default= ''). If `apply_windows` is *false* those $a_{\ell m}$s are used as they are, without being multiplied by the beam or pixel window function (with the assumption that they already have the correct window functions). If `apply_windows` is *true*, the beam and pixel window functions chosen above are applied to the constraining $a_{\ell m}$ (with the assumption that those are free of beam and pixel window function). The code does not check the validity of these asumptions; if none is true, use the alteralm facility to modify or remove the window functions contained in the constraining $a_{\ell m}$.

apply_windows = Determines how the constraining $a_{\ell m}$ read from `almsfile` are treated with respect to window functions; see above for details. y, yes, t, true, .true. and 1 are considered as *true*, while n, no, f, false, .false. and 0 are considered as *false*, (default = .false.).

plmfile = Defines the input filename for a file containing precomputed Legendre polynomials $P_{\ell m}$. (default= '')

windowfile = Defines the input filename for the pixel smoothing windows (default= pixel_window_n????.fits, see below).

winfiledir = Defines the directory in which windowfile is located (default: see Notes on default files and directories on page 3).

DESCRIPTION

Synfast reads the power spectrum from a file in ascii FITS format. This can contain either just the temperature power spectrum C_ℓ^Ts or temperature and polarisation power spectra: C_ℓ^T, C_ℓ^E, C_ℓ^B and $C_\ell^{T\times E}$ (see Note 1, below). If `simul_type = 2` synfast generates Q and U maps as well as the temperature map. The output map(s) is (are) saved in a FITS file. The C_ℓs are used up to the specified $\ell_{\rm max}$, which can not exceed $4\times$ nsmax. If `simul_type = 3` or `4` the first derivatives of the temperature field or the first and second derivatives respectively are output as well as the temperature itself: $T(p)$, $(\partial T/\partial\theta, \partial T/\partial\phi/\sin\theta)$, $(\partial^2 T/\partial\theta^2, \partial^2 T/\partial\theta\partial\phi/\sin\theta,$ $\partial^2 T/\partial\phi^2/\sin^2\theta)$. If `simul_type = 5` or `6` the first derivatives of the (T,Q,U) fields or the first and second derivatives respectively are output as well as the field themself: $T(p)$, $Q(p)$, $U(p)$, $(\partial T/\partial\theta, \partial Q/\partial\theta, \partial U/\partial\theta; \partial T/\partial\phi/\sin\theta, \ldots)$, $(\partial^2 T/\partial\theta^2, \ldots; \partial^2 T/\partial\theta\partial\phi/\sin\theta, \ldots;\ \partial^2 T/\partial\phi^2/\sin^2\theta \ldots)$

The random sequence seed for generation of $a_{\ell m}$ from the power spectrum should be non-zero integer. If 0 is provided, a seed is generated randomly by the code, based on the current date and time. The map can be convolved with a gaussian beam for which a beamsize can be specified, or for an arbitrary *circular* beam for which the Legendre transform is provided. The map is automatically convolved with a pixel window function. These are stored in FITS files in the `healpix/data` directory. If synfast is not run in a directory which has these files, or from a directory which can reach these files by a '`../data/`' or '`./data/`' specification, the system variable `HEALPIX` is used to locate the main **HEALPix** directory and its `data` subdirectory is scanned. Failing this, the location of these files must be specified (using winfiledir). In the interactive mode this is requested only when necessary (see Notes on default directories on page 3).

If some of the $a_{\ell m}$ in the simulations are constrained eg. from observations, a FITS file with these $a_{\ell m}$ can be read. This FITS file contains the $a_{\ell m}$ for certain ℓ and m values and also the standard deviation for these $a_{\ell m}$. The sky realisation which synfast produces will be statistically consistent with the constraining $a_{\ell m}$.

The code can also be used to generate a set of $a_{\ell m}$ matching the input power spectra, beam size and pixel size with or without actually synthesizing the map. Those $a_{\ell m}$ can be used as an input (constraining $a_{\ell m}$) to another synfast run.

...

Spherical harmonics values in the synthesis are obtained from a recurrence on associated Legendre polynomials $P_{\ell m}(\theta)$. This recurrence consumed most of the CPU time used by synfast up to version 2.15. We have therefore included an option to load precomputed values for the $P_{\ell m}(\theta)$ from a file generated by the **HEALPix** facility plmgen. Since the introduction of accelerated spherical harmonic transforms in **HEALPix** v2.20, this feature is obsolete and should no longer be used.

synfast will issue a warning if the input FITS file for the power spectrum does not contain the keyword `POLNORM`. This keyword indicates that the convention used for polarization is consistent with CMBFAST (and consistent with **HEALPix** 1.2). See the **HEALPix** Primer for details on the polarization convention and the interface with CMBFAST. If the keyword is not found, *no attempt will be made* to renormalize the power spectrum. If the keyword is present, it will be inherited by the simulated map.

Note 1: to allow the generation of maps (and $a_{\ell m}$) with $C_\ell^{T\times B} \neq 0$ and/or $C_\ell^{E\times B} \neq 0$, see the subroutine `create_alm`.

DATASETS

The following datasets are involved in the **synfast** processing.

Dataset	Description
/data/pixel_window_nxxxx.fits	Files containing pixel windows for various nsmax.

SUPPORT

This section lists those routines and facilities (including those *external* to the HEALPix distribution) which can assist in the utilisation of **synfast**.

generate_beam — This **HEALPix** Fortran subroutine generates or reads the $B(\ell)$ window function used in synfast

map2gif — This **HEALPix** Fortran facility can be used to visualise the output map of synfast.

mollview	This **HEALPix** IDL facility can be used to visualise the output map of synfast.
alteralm	This **HEALPix** Fortran facility can be used to implement the beam and pixel window functions on the constraining $a_{\ell m}$s (`almsfile` file).
anafast	This **HEALPix** Fortran facility can analyse a **HEALPix** map and save the $a_{\ell m}$ and C_ℓs to be read by synfast.
plmgen	This **HEALPix** Fortran facility can be used to generate precomputed Legendre polynomials.

EXAMPLES: #1

```
synfast
```

Synfast runs in interactive mode, self-explanatory.

EXAMPLES: #2

```
synfast filename
```

When 'filename' is present, synfast enters the non-interactive mode and parses its inputs from the file 'filename'. This has the following structure: the first entry is a qualifier which announces to the parser which input immediately follows. If this input is omitted in the input file, the parser assumes the default value. If the equality sign is omitted, then the parser ignores the entry. In this way comments may also be included in the file. In this example, the file contains the following qualifiers:

```
simul_type= 1
nsmax= 32
nlmax= 64
iseed= -1
fwhm_arcmin= 420.0
infile= cl.fits
outfile= map.fits
```

Synfast reads in the C_ℓ power spectrum in 'cl.fits' up to $\ell = 64$, and produces the (RING ordered) map 'map.fits' which has $N_{\rm side} = 32$. The map is convolved with a beam of FWHM 420.0 arcminutes. The iseed$= -1$ sets the random seed for the realisation. A different iseed would have given a different realisation from the same power spectrum.
Since

```
outfile_alms
almsfile
apply_windows
plmfile
beam_file
windowfile
```

were omitted, they take their default values (empty strings). This means that no file for constrained realisation or precomputed Legendre polynomials are read, the $a_{\ell m}$ generated in the process are not output, and synfast attempts to find the pixel window files in the default directories (see Notes on default files and directories on page 3.

RELEASE NOTES

Revision 1: Initial release (**HEALPix** 0.90)

Revision 2: Optional non-interactive operation. Proper FITS file support. Improved reccurence algorithm for $P_{\ell m}(\theta)$ which can compute to higher ℓ values. Improved pixel windows averaged over actual HEALPix pixels. New functionality: constrained realisations, precomputed $P_{\ell m}$. (**HEALPix** 1.00)

Revision 3: New functionality: constrained realisations and pixel windows are now available for polarization as well. Arbitrary circular beams can be used. New parser (**HEALPix** 1.20)

Revision 4: New functionnality: the generated $a_{\ell m}$ can be output, and the map synthesis itself can be skipped. First and second derivatives of the temperature field can be produced on demand.

Revision 5: New functionnality: First and second derivatives of the Q and U Stokes field can be produced on demand.

Revision 6: Bug correction: corrected numerical errors on derivatives $\partial X/\partial\theta$, $\partial^2 X/(\partial\theta\partial\phi\sin\theta)$, $\partial^2 X/\partial\theta^2$, for $X = Q, U$. See this appendix for details. (**HEALPix** 2.14)

Messages

This section describes error messages generated by **synfast**.

Message	Severity	Text
can not allocate memory for array xxx	Fatal	You do not have sufficient system resources to run this facility at the map resolution you required. Try a lower map resolution.
this is not a binary table		the fitsfile you have specified is not of the proper format
there are undefined values in the table!		the fitsfile you have specified is not of the proper format
the header in xxx is too long		the fitsfile you have specified is not of the proper format
XXX-keyword not found		the fitsfile you have specified is not of the proper format
found xxx in the file, expected:yyyy		the specified fitsfile does not contain the proper amount of data.

ud_grade

Location in HEALPix directory tree: src/f90/ud_grade/ud_grade.f90

This program can upgrade or degrade the resolution of a **HEALPix** map.

FORMAT

% ud_grade [options] [parameter_file]

COMMAND LINE OPTIONS

`-d`
`--double` double precision mode (see Notes on double/single precision modes on page 3)

`-s`
`--single` single precision mode (default)

QUALIFIERS

nside_out = Defines the resolution parameter for the output map. (default= 64)

infile = Defines the name of the file containing the map to be up/degraded. (default='map.fits')

outfile = Defines the filename for the output up/degraded map. (default='outmap.fits')

DESCRIPTION

This facility transforms the resolution of an input **HEALPix** map. At each step of map resolution upgrade the four output map pixels nested in one pixel of the input map are given the values of the input pixel. At each step of map resolution degradation the four input map pixels nested in one output map pixel are averaged to produce the pixel value in the output map. **Caution** Beware that, at this stage, the parallel tranport of the polarization (Q and U Stokes vectors) that would be necessary to describe the change in local coordinates is **not** implemented.

DATASETS

The following datasets are involved in the **ud_grade** processing.

Dataset	Description
None required	

SUPPORT

This section lists those routines and facilities (including those *external* to the HEALPix distribution) which can assist in the utilisation of **ud_grade**.

mollview	IDL routine to view an up/downgraded map.
anafast	This **HEALPix** facility can analyse an up/downgraded map.

EXAMPLES: #1

```
ud_grade
```

ud_grade runs in interactive mode, self-explanatory.

EXAMPLES: #2

```
ud_grade filename
```

When ‘filename’ is present, ud_grade enters the non-interactive mode and parses its inputs from the file ‘filename’. This has the following structure: the first entry is a qualifier which announces to the parser which input immediately follows. If this input is omitted in the input file, the parser assumes the default value. If the equality sign is omitted, then the parser ignores the entry. In this way comments may also be included in the file. In this example, the file contains the following qualifiers:

```
nside_out= 64
infile= map.fits
outfile= outmap.fits
```

Ud_grade transforms the **HEALPix** map in ’map.fits’ to resolution $Nside = 64$, no matter what the input map resolution was. The up/downgraded map is stored in ’outmap.fits’.

RELEASE NOTES

Revision 1: (Initial release **HEALPix** 0.90)

Revision 2: Extension to multi-dimensional maps (**HEALPix** 1.20)

Messages

This section describes error messages generated by **ud_grade**.

Message	Severity	Text
can not allocate memory for array xxx	Fatal	You do not have sufficient system resources to run this facility at the map resolution you required. Try a lower map resolution.

Appendix

Bug Correction in synfast 2.14

Thanks to the routine alm2map_der, the Fortran90 synfast facility produces maps of I, Q, U Stokes parameters and their first and second spatial derivatives, starting from $C(\ell)$ or $a_{\ell m}$ coefficients. **A bug affecting the calculation of $\partial X/\partial\theta$, $\partial^2 X/\partial\theta^2$, $\partial^2 X/(\partial\theta\partial\varphi\sin(\theta))$, for $X = (Q, U)$ was detected in this routine and has been fixed in release 2.14 (March 2010).**

In what follows, the impact of this bug on the power spectra of the produced maps is quantified, so that users can assess how much their work could have been affected by this bug.

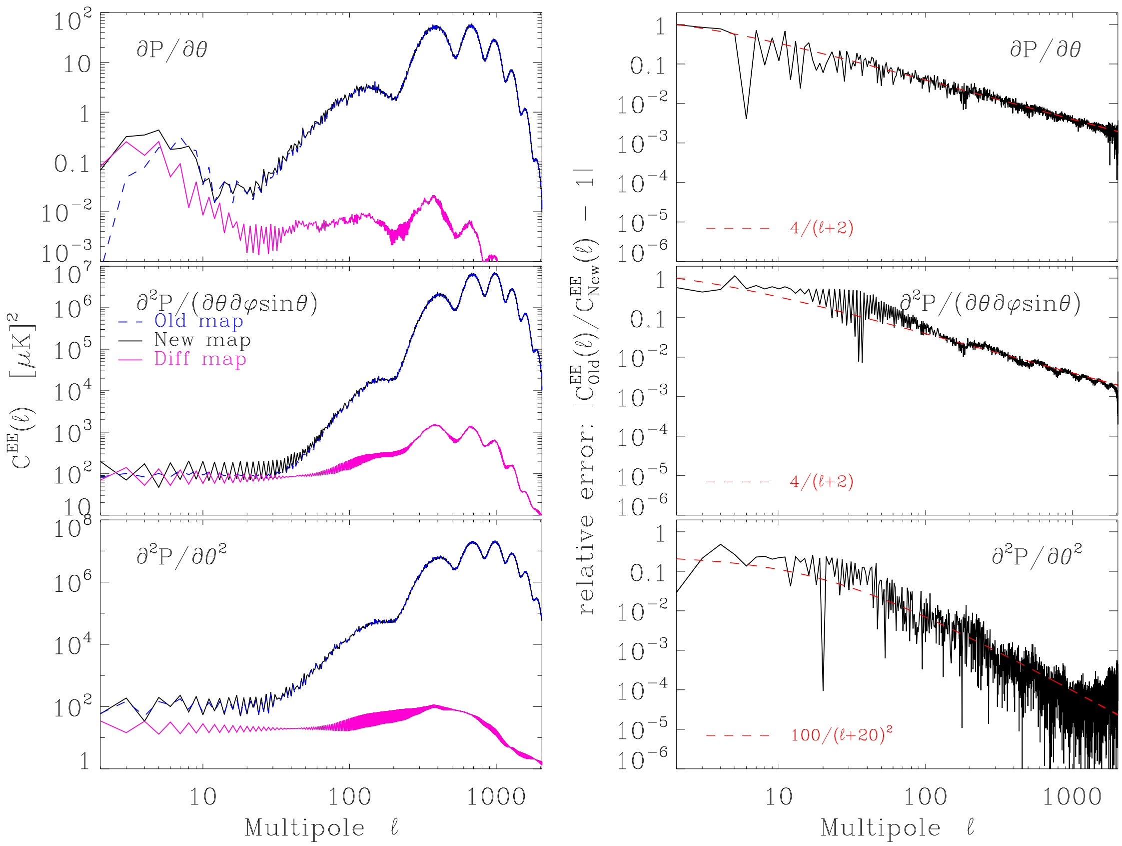


Figure 1: Left panels: comparison of the EE power spectra $C(\ell)$ computed on polarized maps derivatives generated by `synfast`-2.13a (Old maps, blue bashes), the bug corrected `synfast`-2.14 (New maps, black lines) and their differences (Diff maps, magenta lines). Note that what is plotted is $C(\ell)$, *not* the customary $\ell(\ell+1)C(\ell)/2\pi$. Right panels show respectively the relative error on the EE power spectrum of the old derivatives maps compared to that of the new maps. The red dashes show analytical fit to these errors.

In Figure 1 we show the polarization EE power spectrum of $N_{\rm side} = 1024$ maps in which the Stokes parameters (Q, U) have been replaced by, in turn, their derivatives $\partial(Q, U)/\partial\theta$, $\partial^2(Q, U)/\partial\theta^2$, $\partial^2(Q, U)/(\partial\theta\partial\varphi \sin\theta)$, for maps generated by either the version 2.13a of `synfast` or the corrected version 2.14, or the difference of the two set of maps. The input power spectra were those of WMAP-1yr Λ-CDM best fit model with a Gaussian beam FWHM of 10 arcmin. The power spectra were computed on the whole maps, except for 12 pixels around each pole that were masked out, because they get very bright in second order derivatives.

It can be seen that the relative effect of the computation error on the produced maps was large at low ℓ, at scales on which derivatives maps contain little power, but decreasing steadily with ℓ.

It should be stressed that the following quantities were *not* affected by the bug described above:

- the Stokes parameters themselves (I, Q, U),
- the intensity I and all its derivatives,
- the Laplacians $\Delta I, \Delta Q$ and ΔU, with $\Delta \equiv \left(\frac{\partial^2}{\partial\theta^2} + \cot\theta \frac{\partial}{\partial\theta} + \frac{\partial^2}{\sin^2\theta\partial\varphi^2}\right)$.