GammaLib - Action #1731

Feature # 1729 (Closed): Add support to smooth sky maps

Use low-level FFT classes to implement GSkyMap smoothing

03/03/2016 10:46 PM - Knödlseder Jürgen

Status:	Closed	**Start date:**	03/03/2016
Priority:	Normal	**Due date:**	
Assigned To:	Knödlseder Jürgen	**% Done:**	100%
Category:		**Estimated time:**	0.00 hour
Target version:	1.5.0		

Description

The low-level FFT classes should be used to implement GSkyMap smoothing

Related issues:

Related to GammaLib - Feature # 1768: Investigate whether we can interface Ga...	**In Progress**	**04/18/2016**

History

#1 - 06/21/2016 10:01 PM - Knödlseder Jürgen

- Target version set to 1.2.0

#2 - 10/07/2016 10:17 PM - Knödlseder Jürgen

One possibility is to implement a GFft2d class for performing a 2-dimensional fast fourier transform. The class would store the fourier transform coefficients, allow operations, and provide methods for forward and backward transformations. A possible use case could look like this (this makes use of a ndarray, see #1768):

```
GNdarray a(10,5);
GNdarray b(10,5);
GFft2d fa(a);
GFft2d fb;
fb.forward(b);
GFft2d fc = fa * fb;
GNdarray c = fc.backward();
```

#3 - 10/07/2016 10:17 PM - Knödlseder Jürgen

- Related to Feature #1768: Investigate whether we can interface GammaLib with NumPy added

#4 - 10/16/2016 11:43 PM - Knödlseder Jürgen

- Status changed from New to In Progress

- Assigned To set to Knödlseder Jürgen

- % Done changed from 0 to 70

I implemented the FFT from the GNU Scientific Library. Two classes have been added:

- GFft which performs a FFT on a n-dimensional array of type GNdarray
- GFftWavetable which is a helper class for GFft that contains the trigonometric coefficients for a factorisation

The class so far operates only on 1-dimensional arrays since the GSL does not provide support for more-dimensional arrays. Implementation of such support should however not be too complicated.

#5 - 10/17/2016 12:45 AM - Knödlseder Jürgen

Here a Fortran code of a 2D FFT:

```
C
C Transform X lines of C array
c
c  On 10 May 2010, the index IW was modified.
c
      IW = 2 * L + INT ( LOG ( REAL ( L ) ) ) + 5

      CALL CFFTMF(L, 1, M, LDIM, C, (L-1) + LDIM*(M-1) +1,
     1     WSAVE(IW), 2*M + INT(LOG(REAL(M))) + 4,
     2     WORK, 2*L*M, IER1)
      IF (IER1 .NE. 0) THEN
        IER = 20
        CALL XERFFT ('CFFT2F',-5)
        GO TO 100
      ENDIF
C
C Transform Y lines of C array
C
      IW = 1
      CALL CFFTMF (M, LDIM, L, 1, C, (M-1)*LDIM + L,
     1     WSAVE(IW), 2*L + INT(LOG(REAL(L))) + 4,
     2     WORK, 2*M*L, IER1)
      IF (IER1 .NE. 0) THEN
        IER = 20
        CALL XERFFT ('CFFT2F',-5)
      ENDIF
```

with

- L is the number of elements in the first dimension
- M is the number of elements in the second dimension
- LDIM is the number of elements in the first dimension and corresponds to the stride

The CFFTMF is shown below and calls the 1D function CMFM1F:

```
      SUBROUTINE CFFTMF (LOT, JUMP, N, INC, C, LENC, WSAVE, LENSAV,
     1                   WORK, LENWRK, IER)
C
      INTEGER  LOT, JUMP, N, INC, LENC, LENSAV, LENWRK, IER
      COMPLEX  C(LENC)
      REAL     WSAVE(LENSAV)     ,WORK(LENWRK)
      LOGICAL  XERCON
C
      IW1 = N+N+1
      CALL CMFM1F (LOT,JUMP,N,INC,C,WORK,WSAVE,WSAVE(IW1),
     1                        WSAVE(IW1+1))
      RETURN
      END
```

with

- LOT is the number of sequences to be transformed
- JUMP is the integer increment of the first elements of two consecutive sequences

- N is the integer length of each sequence to be transformed
- INC is the integer increment of two consecutive elements within the same sequence

```
                N-1
C(L*JUMP+J*INC+1) = SUM C(L*JUMP+K*INC+1)*EXP(-I*J*K*2*PI/N)
                K=0
where I=SQRT(-1).
J=0,...,N-1
L=0,...,LOT-1
```

#6 - 10/17/2016 03:56 PM - Knödlseder Jürgen

- % Done changed from 70 to 90

The GFft class now also supports 2-dimensional arrays.

Code has been merged into devel.

What remains is the implementation of the operators, and the usage of the GFft class for map smoothing in GSkyMap.

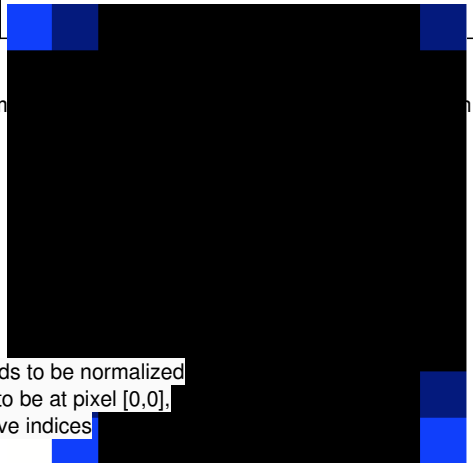
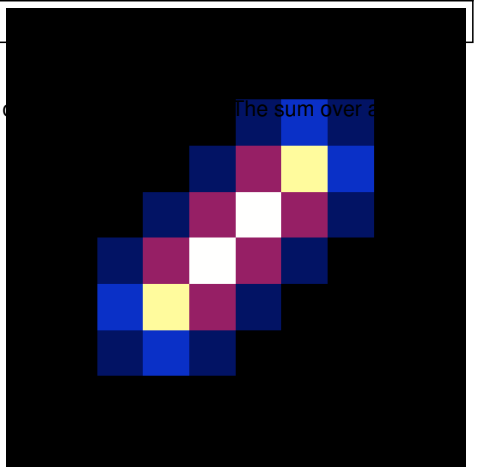
#7 - 10/17/2016 04:57 PM - Knödlseder Jürgen

- File input.png added

- File kernel.png added

- File smoothed.png added

Here a test sequence to illustrate that smoothing of a 2-dimensional image works (this also illustrates how the smoothing kernel needs to be aligned):

Input image	Kernel	Smoothed imaged
		

A m The sum over a
i

```
    # Allocate and set 2-d array
    array = gammalib.GNdarray(10, 10)
    for i in range(3,7):
        array[i,i] = 1.0
    ref = array.sum()

    # Allocate and set 2-d kernel. The kernel needs to be normalized
    # to unity and the centre of the kernel needs to be at pixel [0,0],
    # and it needs to be wraped around to negative indices
    kernel = gammalib.GNdarray(10, 10)
    kernel[0,0] = 0.4
    # 0.4
    kernel[0,1] = 0.1
    kernel[1,0] = 0.1
    kernel[0,9] = 0.1
    kernel[9,0] = 0.1
    # 0.2
    kernel[1,1] = 0.05
    kernel[9,1] = 0.05
    kernel[9,9] = 0.05
```

```
    kernel[1,9] = 0.05

    # Smooth 2-d array using FFT
    fft_array  = gammalib.GFft(array)
    fft_kernel = gammalib.GFft(kernel)
    fft_smooth = fft_array * fft_kernel

    # Backtransform
    smooth = fft_smooth.backward()

    # Test sum
    sum = smooth.sum()
    self.test_value(sum, ref)

    # Store in sky map
    map = gammalib.GSkyMap('CAR','CEL',0.0,0.0,-1.0,1.0,10,10)
    for iy in range(10):
        for ix in range(10):
            map[ix+iy*10] = smooth[ix,iy]
    map.save('test_fft.fits', True)
```

#8 - 03/03/2017 10:23 AM - Knödlseder Jürgen

- Target version changed from 1.2.0 to 1.3.0

#9 - 06/06/2017 10:25 PM - Knödlseder Jürgen

- Target version changed from 1.3.0 to 1.4.0

#10 - 07/31/2017 11:10 PM - Knödlseder Jürgen

- Target version changed from 1.4.0 to 1.5.0

#11 - 10/17/2017 05:14 PM - Knödlseder Jürgen

- Status changed from In Progress to Closed

- % Done changed from 90 to 100

The GSkyMap::smooth() method was added to accomplish the job.

So far the method supports smoothing using a uniform disk kernel and smoothing using a Gaussian kernel.

Files

input.png	19.4 KB	10/17/2016	Knödlseder Jürgen
kernel.png	19.4 KB	10/17/2016	Knödlseder Jürgen
smoothed.png	19.8 KB	10/17/2016	Knödlseder Jürgen