Arguments

data	The array to search
min_location	Position of the minimum value
mask	If set, only consider locations where mask==True Allowed:bool array Default: null

Returns
min

See Also
min, max, max built-ins, range_with_location

1.5.11 fftserver - Tool
Package utility
Module mathematics

FFTs and related operations
include "fftserver.g"

Constructors

fftserver	Construct an fftserver tool

Functions

autocorr	Auto-correlate an array
complexfft	Complex in-place Fourier transform.
convolve	Convolve two arrays.
crosscorr	Cross-correlate two arrays
done	Delete the fftserver tool
mfft	mutiple Fourier transforms.
realtocomplexfft	Real to complex forward Fourier transform.
shift	Shift an array a non-integral number of pixels.

Description

This tool does Fourier transforms and related operations like cross and auto-correlations and frequency domain based shifting. The computation is

performed in a pre-compiled executable. It is not interpreted by Glish.
Throughout this module the following definitions will be used for the Fourier transform. A *forward* transform is one where the exponent is negative and no scaling is done of the result. ie., given a 1-dimensional array, $x[t]$, the Fourier transform, X[f], is given by[1]:

$$X[f] = \sum_{t=-N/2}^{t=N/2-1} x[t] \exp(\frac{-2\pi i f t}{N})$$

A *backward* transform is one where the exponent is positive and the result is scaled by $\frac{1}{N}$. ie., given a 1-dimensional array, $X[f]$, the Fourier transform, x[t], is given by:

$$x[t] = \frac{1}{N} \sum_{f=-N/2}^{f=N/2-1} X[f] \exp(\frac{2\pi i f t}{N})$$

The transforms described above follow the radio-astronomical convention of having the origin of the transform as the centre pixel. This means that if the vector $x[t]$ is symmetric about the centre pixel then the result of a forward transform will also be a real vector. The centre pixel is the one at the $(N+1)/2$ location, using integer division, with the first element at index 1.
The length of the Fourier transform is not restricted to a power of two. Any length transform can be done.
Similarly the input data can be an N-dimensional array and N-dimensional transforms will be performed.

Example

To use the functions in this tool we have to firstly (1) load the definition of the fftserver tool and (2) construct a server tool that will actually perform the computations.

```
include 'fftserver.g'     # 1
server := fftserver()     # 2
```

Step 2 above starts the 'numerics' process if it is not already running.
After creating the server a full complex to complex in-place FFT can be performed:

```
server.complexfft(a, 1)     # forward (+1) fft on complex array "a"
server.complexfft(a, -1)    # reverse (-1) fft on complex array "a"
```

If you know that you have real-valued arrays, you can perform a real to complex FFT. Here the FFT is not done in place, so the original array is preserved:

[1] For clarity I have assumed that $x[t]$ & $X[f]$ have indices between $\frac{-N}{2}$ and $\frac{N}{2}-1$ although in practice the indices will be between 1 and N.

```
b := server.realtocomplexfft(a)
```

Note that the output array has the same shape as the input array so that half of the output array is the conjugate of the other half.
FFT-based convolution is available using the convolve function.

```
c := server.convolve(a,b)
```

The input arrays must have the the same number of dimensions, but they do not have to be the same shape, the padding is handled for you. The output array is the size of the larger of the input arrays. These convolutions are "linear" ie., the effects of the convolution at one edge do not reflect onto the opposite side of the array.
Auto and cross-correlations of real arrays can also be done.

```
c := server.autocorr(a)
c := server.crosscorr(a,b)
```

For cross-correlations, the arrays must have the same shape. The correlations are circular, meaning that effects near one edge of the array reflect to the opposite side. If this is not what you want, you should zero-pad the arrays first.
The finally the fftserver tool can do an an FFT-based shift of an array. For example, if `a` is a 1-dimensional array, then:

```
b := server.shift(a, 3.5)
```

would shift the array 3.5 pixels towards the end of the array. This is equivalent to a band-limited, or sinc interpolation of the array.
Once you have finished doing transforms you can shut down the numerics process, assuming nothing else needs it, using the done function.

```
server.done()
```

See Also
fftserverdemo fftservertest

fftserver - Constructor

Package	**utility**
Module	**mathematics**
Tool	**fftserver**

Construct an fftserver tool

Synopsis

fftserver()

Description

Use this constructor to construct an fftserver tool. If it is not already running this this will start the numerics process on your computer. You can construct as many fftserver tools as you like and they all share the same numerics process.

Returns
An fftserver tool

Example
Look at the example for the mfft function.

---

fftserver.complexfft - Function

Package	**utility**
Module	**mathematics**
Tool	**fftserver**

Complex in-place Fourier transform.

Synopsis
complexfft(a, dir)

Arguments

a	Complex array to transform
dir	Direction of the transform Allowed:+1 forward, -1 reverse

Returns
Nothing

Description
This function does a complex N-dimensional Fourier transform of the supplied

array. The transformed array is the same shape as the input array and is returned in the same variable.

---

fftserver.mfft - Function

Package	**utility**
Module	**mathematics**
Tool	**fftserver**

mutiple Fourier transforms.

Synopsis
mfft(a, axes, forward)

Arguments

a	Complex array to transform
axes	axes to transform Allowed:Vector of Booleans
forward	Direction of the transform Allowed:T=forward, F=reverse Default: T

Returns
Complex array the same size as the input array

Description
This function does a complex Fourier transform along user selected axes of an N-dimensional array. Transforms are done along the axes where the axes vector is True. The axes vector must have the same number of elements as the number of dimensions in the input array.
Backward transforms are specified by setting the forward argument to False. The default is to do forward transforms.
The returned (transformed) array is the same shape as the input array.

Example

```
include 'fftserver.g'
server := fftserver();
```

```
a := array(0,3,4);
a[,] := 1;                   # set all elements to 1
b1 := server.mfft(a, [T,F]); # transform along axis 1
b2 := server.mfft(a, [F,T]); # transform along axis 2
b := server.mfft(a, [T,T]);  # transform along both axes, ie., a 2-D transform.
```

---

fftserver.realtocomplexfft - Function

Package	**utility**
Module	**mathematics**
Tool	**fftserver**

Real to complex forward Fourier transform.

Synopsis
realtocomplexfft(a)

Arguments

a	Any real array

Returns
complex array

Description
This function does a real-to-complex N-dimensional forward Fourier transform of the supplied real array. The transformed array is returned as a complex, Hermition array. It is the same shape as the input array.

---

fftserver.convolve - Function

Package	**utility**
Module	**mathematics**
Tool	**fftserver**

Convolve two arrays.

Synopsis
convolve(a, b)

Arguments

a	Real array
b	Real array

Returns
Real array

Description
Convolve the two input arrays. The arrays do not have to be the same size, but must have the same number of dimensions. The output array is the size of the larger of the input arrays.

---

fftserver.crosscorr - Function

Package	**utility**
Module	**mathematics**
Tool	**fftserver**

Cross-correlate two arrays

Synopsis
crosscorr(a, b)

Arguments

a	Real array
b	Real array

Returns
Real array

Description
Cross correlate two real arrays. The arrays must be the same size. At present

this is a circular correlation — if this is not what you want you must pad the arrays yourself.

---

fftserver.autocorr - Function

Package	**utility**
Module	**mathematics**
Tool	**fftserver**

Auto-correlate an array

Synopsis
autocorr(a)

Arguments

a	Real array

Returns
Real array

Description
Correlate a real with itself. At present this is a circular correlation — if this is not what you want you must pad the array yourself.

---

fftserver.shift - Function

Package	**utility**
Module	**mathematics**
Tool	**fftserver**

Shift an array a non-integral number of pixels.

Synopsis
shift(a, shift)

Arguments

a	Real array
shift	How much to shift on each axis. >0 means towards the end of the array, <0 means towards the beginning of the array.

Returns
Real array

Description

The units of the shift, which need not be integral, are pixels. The length of `shift` must be the same as the number of dimensions of `a`. So, if `a` is a 1-dimensional array, then `shift` is a scalar.
This function shifts an array by; Fourier transforming it, multiplying each element of the transform by a complex phase factor, inverse Fourier transforming the result.
Multiplication by a linear phase gradient in the spectral domain corresponds to a shift in the image domain, so this process will shift the original array. If the shift is a non-integral number of pixels then a band-limited, or sinc interpolation is done.

---

fftserver.done - Function

Package	**utility**
Module	**mathematics**
Tool	**fftserver**

Delete the fftserver tool

Synopsis
done()

Description
The done function frees up all the memory associated with a fftserver tool. After calling this function the fftserver tool cannot be used. This function will shut down the server process if no other tools are using the numerics server.

Returns
T

Example

```
include 'fftserver.g'
server := fftserver();
# ... manipulate the server as you see fit ...
server.done(); # shut down the fftserver tool
```

1.5.12 fftserverdemo - Function

Package utility
Module mathematics

Demonstrate the use of functions in fftserver

Synopsis
fftserverdemo()
include "fftserver.g"

Description
Demonstrate the use of the functions in fftserver. Inputs and results are sent to the logger.

See Also
fftserver fftservertest

Returns
Bool: T for success

1.5.13 fftservertest - Function

Package utility
Module mathematics

Test the fftserver class