Subject: Re: FFT in 1 dimension
Posted by hradilv.nospam on Fri, 07 Dec 2001 19:05:36 GMT
View Forum Message <> Reply to Message

Here's what I came up with. Any comments?

```
function ft1d, data, dim, inverse=inverse

  ds = size(data)
  ndim = ds[0]
  nlines = n_elements(data)/ds[dim]

  dims = lindgen(ndim)+1
  dims[dim-1] = -1
  porder = sort(dims)

  fdata = transpose(data,porder)
  fdata = reform(fdata,ds[dim],nlines)

  for i=0, nlines-1 do fdata[*,i] = fft(fdata[*,i],inverse=inverse)

  porder = (ds[1:ndim])[sort(dims)]
  fdata = reform(fdata,porder)
  fdata = transpose(fdata,sort(porder))

  return, fdata
end
```

Subject: Re: FFT in 1 dimension
Posted by hradilv.nospam on Fri, 07 Dec 2001 20:10:03 GMT
View Forum Message <> Reply to Message

OOPS - I think this one works:

```
function ft1d, data, dim, inverse=inverse

  ds = size(data)
  ndim = ds[0]
  nlines = n_elements(data)/ds[dim]

  dims = lindgen(ndim)+1
  dims[dim-1] = -1
  porder = sort(dims)

  fdata = transpose(data,porder)
  fdata = reform(fdata,ds[dim],nlines)
```

```
  for i=0, nlines-1 do fdata[*,i] = fft(fdata[*,i],inverse=inverse)

  dims[0] = dim
  porder = sort(dims)
  fdata = reform(fdata,(ds[1:ndim])[porder])
  fdata = transpose(fdata,porder)

  return, fdata
end
```

On Fri, 07 Dec 2001 19:05:36 GMT, hradilv.nospam@yahoo.com wrote:

```
> Here's what I came up with.  Any comments?
>
> function ft1d, data, dim, inverse=inverse
>
>   ds = size(data)
>   ndim = ds[0]
>   nlines = n_elements(data)/ds[dim]
>
>   dims = lindgen(ndim)+1
>   dims[dim-1] = -1
>   porder = sort(dims)
>
>   fdata = transpose(data,porder)
>   fdata = reform(fdata,ds[dim],nlines)
>
>   for i=0, nlines-1 do fdata[*,i] = fft(fdata[*,i],inverse=inverse)
>
>   porder = (ds[1:ndim])[sort(dims)]
>   fdata = reform(fdata,porder)
>   fdata = transpose(fdata,sort(porder))
>
>   return, fdata
> end
>
```

Subject: Re: FFT in 1 dimension
Posted by Wayne Landsman on Fri, 07 Dec 2001 21:41:10 GMT

View Forum Message <> Reply to Message

hradilv.nospam@yahoo.com wrote:

> Here's what I came up with. Any comments?

I haven't examined your code in detail, but there is one line where the

speed of the program can be improved significantly:

Instead of writing

```
 for  i=0, nlines-1 do fdata[*,i] = fft(fdata[*,i],inverse=inverse)
```

write

```
 for i=0, nlines-1 do fdata[0,i] = fft(fdata[*,i],inverse=inverse)
```

I'm actually not sure why this speed things up. I suppose by writing fdata[0,i] on the left hand side, you've directly specified a starting location to place the computation, but if you instead write fdata[*,i] then IDL first extracts a temporary variable, assigns it to the value of the computation, and then writes the temporary variable back into fdata. Still, I would think that the compiler could recognize that these two notations are equivalent.

--Wayne Landsman landsman@mpb.gsfc.nasa.gov

Subject: Re: FFT in 1 dimension
Posted by Paul Woodford on Sat, 08 Dec 2001 04:09:27 GMT
View Forum Message <> Reply to Message

From the What's New in IDL 5.5 document at
<http://www.rsinc.com/idl/whatsnew.cfm>:

New Dimension-specific Transforming for FFT

Paul
