NAME

FFT2TR − Return in-place the direct or inverse fast Fourier transform of a matrix.

SYNOPSIS

CALL FFT2TR(LX,LY,NX,NY,INVDIR,SCALE,SHIFT, F,BX,BY, RC)

LX ≥ NX+1	is the INTEGER*4 first dimensioned size of F
LY ≥ NY+1	is the INTEGER*4 second dimensioned size of F
NX	is the INTEGER*4 number of rows used in F, from up left corner, a + power of 2
NY	is the INTEGER*4 number of cols used in F, from up left corner, a + power of 2
INVDIR	is INTEGER*4; +1 => transform the data, −1 => invert the transform
SCALE	is LOGICAL*4; T => scale output or assume input is scaled
SHIFT	is LOGICAL*4; T => frequency-shift result or assume input was frequency-shifted
F(LX,LY)	is the COMPLEX*16 matrix to be transformed or inverted
BX	is the REAL*8 upper limit of integration in the X direction
BY	is the REAL*8 upper limit of integration in the Y direction
RC	is the INTEGER*4 return code; see below

DESCRIPTION

First the routine sanity-checks its input parameters, sets RC accordingly, and returns without doing anything if RC is not 0. Next it invokes FFT repeatedly to transform the columns, transposes the result, invokes FFT repeatedly to transform the rows, and transposes the result. Then it finds the sampling intervals and new upper limits of integration. Finally it scales the output values if that has been requested, and if frequency-shifting is specified either shifts the forward transform or fixes up the inverse transform of the shifted input. Frequency shifting adds a row and column to F.

SEE ALSO

FFT2ST, which compute the 2-dimensional fast Fourier transform or inverse
DFT2, which computes the 2-dimensional discrete Fourier transform or inverse
FFT, used by this routine, computes the 1-dimensional fast Fourier transform or inverse
DFT, which computes the 1-dimensional discrete Fourier transform or inverse

DIAGNOSTICS

On output RC=0 if all went well, or certain bits are set to 1 if the following error conditions occur:

1 bit	1 => NX or NY is not a positive power of 2
2 bit	1 => INVDIR is not +1 or −1
4 bit	1 => BX or BY is not positive
8 bit	1 => NX or NY is not positive, or LX or LY is not big enough to fit F(NX,NY)

LINKAGE

gfortran source.f −L${HOME}/lib −lmisc

AUTHOR

Michael Kupferschmid

REFERENCE

[1] "Computing Fourier Transforms" by Michael Kupferschmid

EXAMPLE

```
      INTEGER*4 LX/5/,LY/5/,NX/4/,NY/4/
      LOGICAL*4 SCALE/.FALSE./,SHIFT/.FALSE./
      COMPLEX*16 F(5,5)/25*(0.0D0,0.0D0)/
      REAL*8 BX/3.5D0/,BY/3.5D0/
      INTEGER*4 RC
C
C     transform the pulse of [1, Section 7]
      F(2,2)=(2.5D0,0.D0)
      INVDIR=+1
      CALL FFT2TR(LX,LY,NX,NY,INVDIR,SCALE,SHIFT, F,BX,BY, RC)
      WRITE(6,900) RC
  900 FORMAT('RC=',I2)
      DO 1 I=1,NX
           WRITE(6,901) (F(I,J),J=1,NY)
  901      FORMAT(4('(',F5.2,',',F5.2,')'))
    1 CONTINUE
      INVDIR=-1
      CALL FFT2TR(LX,LY,NX,NY,INVDIR,SCALE,SHIFT, F,BX,BY, RC)
      WRITE(6,900) RC
      DO 2 I=1,NX
           WRITE(6,901) (F(I,J),J=1,NY)
    2 CONTINUE
      STOP
      END
```

This example produced the output below. The inverse recovers the input data exactly.

```
unix[1] a.out
RC= 0
( 2.50, 0.00)( 0.00, 2.50)(-2.50, 0.00)(-0.00,-2.50)
( 0.00, 2.50)(-2.50, 0.00)(-0.00,-2.50)( 2.50,-0.00)
(-2.50, 0.00)(-0.00,-2.50)( 2.50, 0.00)( 0.00, 2.50)
(-0.00,-2.50)( 2.50,-0.00)( 0.00, 2.50)(-2.50, 0.00)
RC= 0
( 0.00, 0.00)( 0.00, 0.00)( 0.00, 0.00)( 0.00, 0.00)
( 0.00, 0.00)( 2.50, 0.00)( 0.00, 0.00)( 0.00, 0.00)
( 0.00, 0.00)( 0.00, 0.00)( 0.00, 0.00)( 0.00, 0.00)
( 0.00, 0.00)( 0.00, 0.00)( 0.00, 0.00)( 0.00, 0.00)
unix[2]
```