The Fast Fourier Transform

for\FFT.FOR – (without line numbers for\fftnn.for) cpp\four1.c -starts with data(1) "essentially" the code in Press- cpp\bfft.c – starts with data(0) -- The "best" implementation of the fft claims to be http://www.fftw.org/. They give permission to reproduce their manual which is fftw3.pdf. This version has a fair amount of overhead, but is not limited to powers of two.

Symmetric range.doc modifies the FFT to have a symmetric range..

The document SplineFT.DOC carries integration by parts to the fourth power resulting in $1/f^4$ convergence for a cubic spline.

```
      SUBROUTINE FFT(DATA,NN,ISIGN)
C  This is the Danielson and Lanczos implementation
C  of the fast Fourier transform as described in
C  Numerical Recipes, Press et al in section 12.2.
C  It has been tested by comparing with THE ORIGINAL
C  COOLEY-TUKEY TRANSFORM, which is a fortran 4
C  implementation of the same code.
C  TRANSFORM(K)=SUM(DATA(J)*EXP(ISIGN*
C 2PI*SQRT(-1)*(J-1)*(K-1)/NN)). SUMMED OVER ALL J
C  AND K FROM 1 TO NN. DATA IS IN A ONE-DIMENSIONAL
C COMPLEX ARRAY (I.E.,THE REAL AND IMAGINARY
C PARTS ARE ADJACENT IN STORAGE ,SUCH AS FORTRAN IV
C PLACES THEM) WHOSE LENGTH NN=2**K, K.GE.0 (IF
C  NECESSARY APPEND ZEROES TO THE DATA). ISIGN IS +1
C OR -1. IF A -1 TRANSFORM IS FOLLOWED BY A +1 ONE
C (OR A +1 BY A -1) THE ORIGINAL DATA REAPPEAR,
C MULTIPLIED BY NN. TRANSFORM VALUES ARE RETURNED IN
C ARRAY DATA, REPLACING THE INPUT.
```

To be specific

The C code implementation cpp\bfft.c returns

$$Data(m)=\sum_{k=0}^{N-1} data(k)\exp\left(j2\pi\frac{mk}{N}\right) \quad (1.1)$$

$$data(k)=\sum_{m=0}^{N-1} Data(m)\exp\left(-j2\pi\frac{mk}{N}\right)$$

The usual form for (1.1) has a Δ_t = T/N multiplying the first line and a Δ_f=1/T multiplying the second line to make the equations look like the Fourier integrals. Note that Δ_t times Δ_f = 1/N.

In a simplified Fortran impementation

```
Do I=1,256
  data(i)=exp(I-128)**2
Enddo
Call fft(data,256,1)
Call fft(data,256,-1)
Do I=1,256
   Data(i)=data(i)/256
Enddo
```

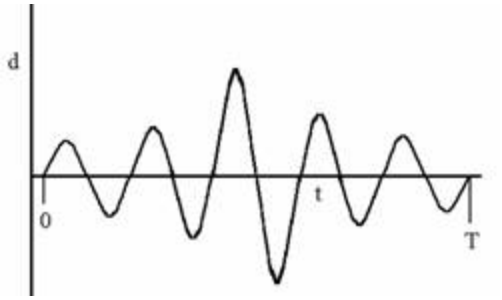
Returns the original data --- inefficient programming??

```
      REAL*8 WR,WI,WPR,WPI,WTEMP,THETA
      DIMENSION DATA(2*NN)
      N=2*NN
      J=1
      DO I=1,N,2
        IF(J.GT.I)THEN
          TEMPR=DATA(J)
          TEMPI=DATA(J+1)
          DATA(J)=DATA(I)
          DATA(J+1)=DATA(I+1)
          DATA(I)=TEMPR DATA(I+1)=TEMPI
        ENDIF
        M=N/2
1       IF((M.GE.2).AND.(J.GT.M))THEN
          J=J-M
          M=M/2
```

```
          GOTO 1
        ENDIF
        J=J+M
      ENDDO
C Here begins the Danielson-Lanczos section (outer
C loop executed Log2 (NN) times
      MMAX=2
2     IF(N.GT.MMAX)THEN
        ISTEP=2*MMAX
        THETA=6.28318530717959D0/(ISIGN*MMAX)
        WPR=-2*DSIN(0.5D0*THETA)**2
        WPI=DSIN(THETA)
        WR=1
        WI=0
        DO M=1,MMAX,2
          DO I=M,N,ISTEP
            J=I+MMAX
            TEMPR=WR*DATA(J)-WI*DATA(J+1)
            TEMPI=WR*DATA(J+1)+WI*DATA(J)
            DATA(J)=DATA(I)-TEMPR
            DATA(J+1)=DATA(I+1)-TEMPI
            DATA(I)=DATA(I)+TEMPR
            DATA(I+1)=DATA(I+1)+TEMPI
          ENDDO
          WTEMP=WR
          WR=WR*WPR-WI*WPI+WR
          WI=WI*WPR+WTEMP*WPI+WI
        ENDDO
        MMAX=ISTEP
        GOTO 2
      ENDIF
      RETURN
      END
```

Integral as a sum

$$H(f) \equiv \int_{-\infty}^{\infty} h(t) e^{j2\pi ft} dt \qquad (1.2)$$



If the h(t) is zero outside the range 0 < t < T this is equivalent to

$$H(f) \equiv \int_{0}^{T} h(t) e^{j2\pi ft} dt \qquad (1.3)$$

Divide the integral into N regions and estimate the value of the integrand by the value at the beginning of each region

$$t_k = \frac{kT}{N} \quad 0 \le k \le N-1 \qquad (1.4)$$

$$H(f) \cong \frac{T}{N} \sum_{k=0}^{N-1} h(t_k) e^{j2\pi ft_k} \qquad (1.5)$$

Next decide to evaluate this sum only at $f_m = \frac{m}{T}$ so that

$$H(f_m) \cong \frac{T}{N}\sum_{k=0}^{N-1} h(t_k) e^{j2\pi f_m t_k}$$
$$= \frac{T}{N}\sum_{k=0}^{N-1} h(t_k) e^{j2\pi \frac{mkT}{TN}} \qquad (1.6)$$
$$= \frac{T}{N}\sum_{k=0}^{N-1} h(t_k) e^{j2\pi \frac{mk}{N}}$$

This is the sum returned by the FFT.

Note that to decrease the spacing between frequencies that it is necessary to increase T. Increasing the number of data points, while holding T fixed gives more frequency values but not closer frequencies.

Periodicity

The m or k values returned are the N values from 0 to N-1

$$H(m+\ell N) = \frac{T}{N}\sum_{k=0}^{N-1} h(k)\exp\left(j2\pi\frac{mk}{N} + j2\pi\frac{\ell N k}{N}\right) \quad (1.7)$$
$$= \frac{T}{N}\sum_{k=0}^{N-1} h(k)\exp\left(j2\pi\frac{mk}{N}\right) = H(m)$$

This means that as a practical matter with $f = m/T$, that the value $H(f=-1/T) = H(f=(N-1)/T)$

With 512 points, the value of H(-1/T)=H(511)

Testing The FFT

Begin with

$$h(t) = \exp\left(-\left(\frac{t-t_0}{w}\right)^2\right) \qquad (2.1)$$
$$H(f) = \int_{-\infty}^{\infty} dt \exp\left(-\left(\frac{t-t_0}{w}\right)^2\right)\exp(j2\pi ft)$$

Complete the square in the integral

$$H(f) = \int_{-\infty}^{\infty} dt \exp\left(\frac{-1}{w^2}\left(t^2 - 2t_0 t + t_0^2 - j2\pi w^2 ft\right)\right)$$
$$= \int_{-\infty}^{\infty} dt \exp\left(\frac{-1}{w^2}\left(t^2 - 2t\left(j\pi w^2 f + t_0\right) + t_0^2\right)\right)$$
$$= \int_{-\infty}^{\infty} dt \exp\left(\frac{-1}{w^2}\left(\left[t - \left(j\pi w^2 f + t_0\right)\right]^2 + t_0^2 - \left(j\pi w^2 f + t_0\right)^2\right)\right) \qquad (2.2)$$
$$= \exp\left(-\frac{\pi^2 w^4 f^2 - 2j\pi w^2 f t_0}{w^2}\right)\int_{-\infty}^{\infty} dt \exp\left(\frac{1}{w^2}\left(\left[t - \left(j\pi w^2 f + t_0\right)\right]^2\right)\right)$$
$$= \exp\left(-\pi^2 w^2 f^2 + 2j\pi f t_0\right) w\int_{-\infty}^{\infty} dt \exp\left(t^2\right)$$
$$= w\sqrt{\pi}\exp\left(-\pi^2 w^2 f^2\right)\exp\left(2j\pi f t_0\right)$$

Define

$$w_f = \frac{1}{\pi w}$$

$$H(f) = \frac{1}{\sqrt{\pi} w_f}\exp\left(-\left(\frac{f}{w_f}\right)^2\right)\exp\left(2j\pi f t_0\right) \qquad (2.3)$$

Note that the width in frequency space is the 1/(πw) so that wide Gaussians in the time domain become narrow lines in the frequency domain.

Numbers

Let T=100, N=1024, t_0 = 10, w = 2

for\TFFT.FOR

```
      PARAMETER (N=1024)
      COMPLEX*8 DAT(N)
      DIMENSION DATA(2,N)
      EQUIVALENCE (DATA(1,1),DAT(1))
      COMPLEX*8 ANAL
      REAL*8 PI
      DATA PI/3.14159365359D0/
```

```
      OPEN(2,FILE='DAT.OUT')
      TM=100
      W=2
      HT=TM/N
      T0=10
      DO I=1,N
        T=HT*(I-1)
        ARG=(T-T0)/W
        ARG=ARG*ARG
        DAT(I)=0
        IF(ARG.LT.80.)DAT(I)=EXP(-ARG)
        WRITE(2,'(2G20.8)')T,REAL(DAT(I))
      ENDDO
      CLOSE(2)
      ISIGN=1
      CALL FFT(DATA,N,ISIGN)
      OPEN(1,FILE='RTRANS.OUT')
      OPEN(2,FILE='ITRANS.OUT')
      OPEN(3,FILE='ARTRANS.OUT')
      OPEN(4,FILE='AITRANS.OUT')
      DO M=1,N
        F=(M-1)/TM
        ARG=PI*W*F
        ARG=ARG*ARG
        ANAL=0
        IF(ARG.LT.80)ANAL=EXP(-ARG)
        ANAL=W*SQRT(PI)*ANAL*EXP((0,1)*2*PI*F*T0)
        WRITE(1,'(2G20.8)')F,HT*REAL(DAT(M))
        WRITE(2,'(2G20.8)')F,HT*IMAG(DAT(M))
        WRITE(3,'(2G20.8)')F,REAL(ANAL)
        WRITE(4,'(2G20.8)')F,IMAG(ANAL)
      ENDDO
      END
C$INCLUDE FFT
```

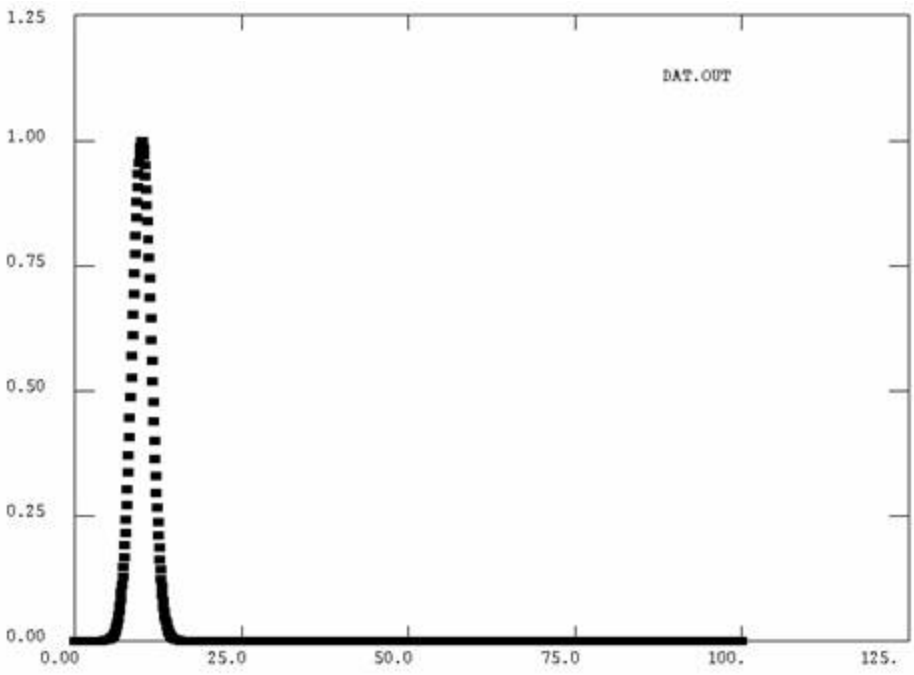


Figure 1 The original data

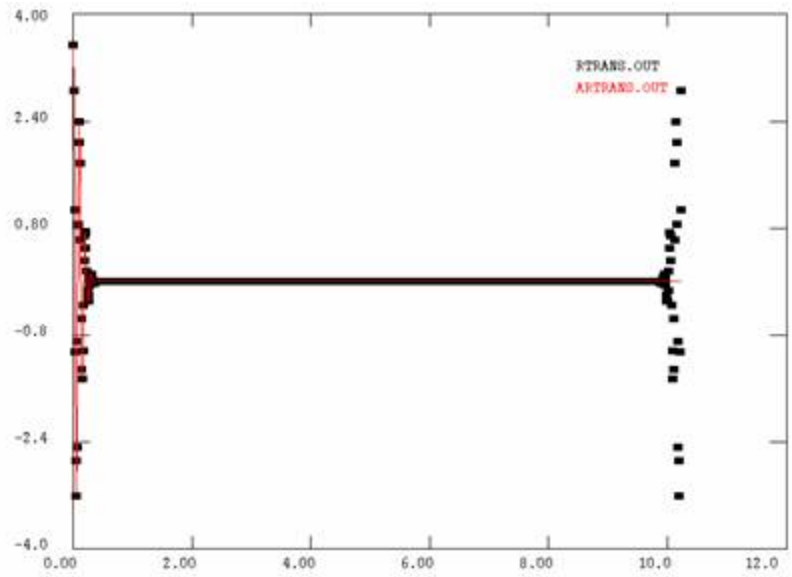
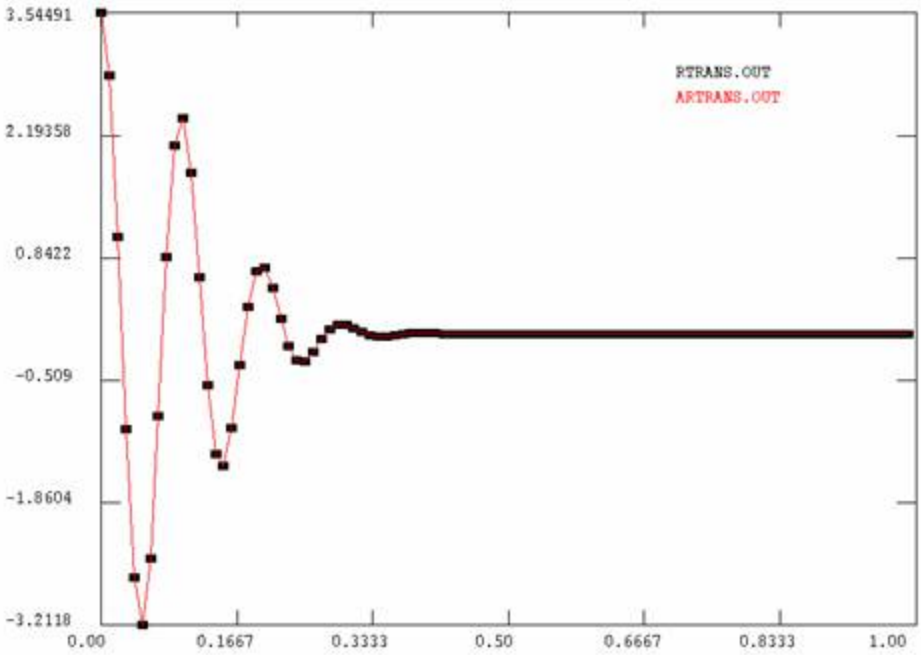


Figure 2 The complete real transform - black points - Analytical transform red lines.



Assignment

Redo the evaluations above with different values of T_0, TM and numbers of points. Plot the transform values from -N/(2T) to N/(2T). What happens if all of the points in the original data are placed inside the region where the Gaussian is large? What happens if there is only a single point inside the Gaussian? Plot the absolute value of the transform. Note that the width of the transform and of the data are in inverse ratio. Wide peaks transform into narrow peaks and vice versa. The location in the time domain is in the phase in the Fourier domain.