CONTENTS

サンプルエクスペリメント – FFT Swapping Demo

クイックノート

メニュー File → Example Experiments → Analysis → FFT Swapping Demo

この Experiment は、FFT（高速フーリエ変換）とスワップを説明するデモです。

基本的な FFT アルゴリズムには、データの原点が配列（ウェーブ）の境界にあるという暗黙の前提があります。
その結果、入力データのシフトは変換の位相シフトを引き起こします。
なぜこのようなことが起こるのかを理解するために、1D 関数　f(x)　のフーリエ変換を考えてみます。
これは、次の式で与えられます。

$$F(u) = \int f(x)e^{-i2\pi xt}\,dx$$

ここで f(x) を B だけオフセットすると、結果として得られる変換は線形乗法的位相シフトを持ちます。

$$F(u)e^{-i2\pi xt} = \int f(x-B)e^{-i2\pi xt}\,dx$$

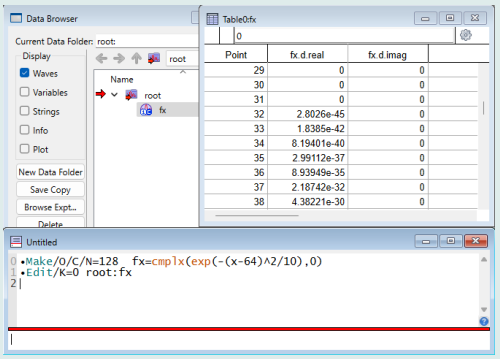
例 1 ：1D 複素信号

この例では、ガウス関数のフーリエ変換もガウス関数であることを念頭に置きながら、1D ガウス関数を変換します。
その幅は、ソースの幅に反比例します。

各例は新しいエクスペリメントを作成したところからコマンドごとに手順を確認します。

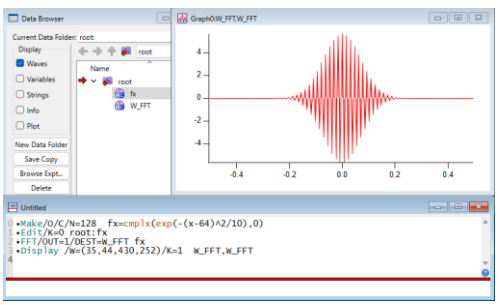
1. コマンドウィンドウで次を入力します。

```
Make/O/C/N=128 fx=cmplx(exp(-(x-64)^2/10),0)
```

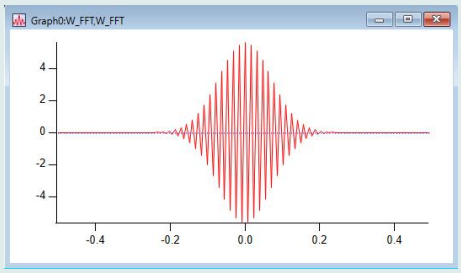


2. コマンドウィンドウで FFT を実行し、結果を表示します。

```
FFT/OUT=1/DEST=W_FFT fx
Display /W=(35,44,430,252)/K=1  W_FFT,W_FFT
```



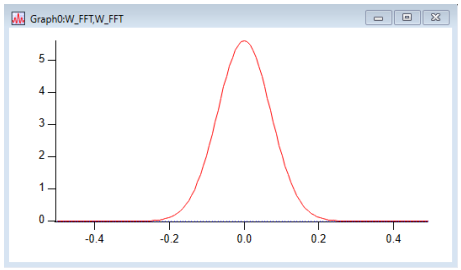
3. コマンドウィンドウで次を実行し、グラフを修正します。




```
ModifyGraph rgb(W_FFT#1)=(0,0,65535)
ModifyGraph cmplxMode(W_FFT)=1,cmplxMode(W_FFT#1)=2
```

その結果、位相によって変調されたガウス包絡線が得られます。

4. シフト定理を検証するには、次のように結果を共役位相で単純に掛けるだけです。




```
W_FFT*=exp(cmplx(0,2*pi*64*x))
```

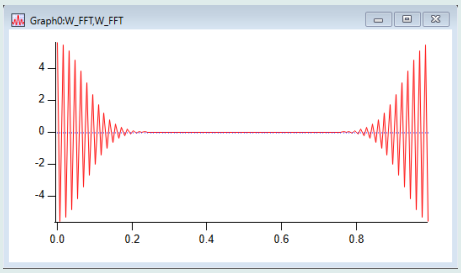
グラフが右図のように変わります。

この例では、複雑な入力がデフォルトで変換およびスワップされていることに注意してください（FFT/Z コマンドのヘルプを参照）。

例 2 ：スワップされた出力

デフォルトの入れ替え（またはローテーション）を行わずに、上記の例を繰り返すと、次のようになります。

1. コマンドウィンドウで次を実行します。

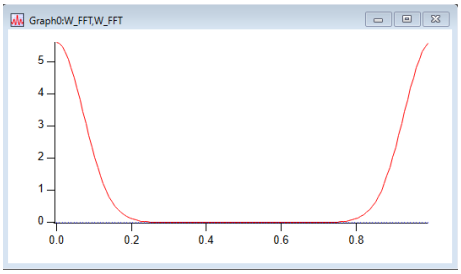



```
Make/O/C/N=128 fx=cmplx(exp(-(x-64)^2/10),0)
```

FFT を実行します。

```
FFT/OUT=1/DEST=W_FFT/Z fx
Display /W=(35,44,430,252)/K=1  W_FFT,W_FFT
ModifyGraph rgb(W_FFT#1)=(0,0,65535)
ModifyGraph
cmplxMode(W_FFT)=1,cmplxMode(W_FFT#1)=2
```

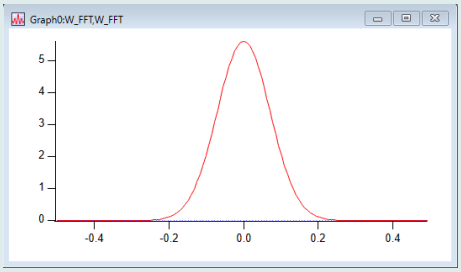
2. 前の手順と同様、コマンドウィンドウで次を実行します。




```
W_FFT*=exp(cmplx(0,2*pi*64*x))
```

グラフが右図のように変わります。

3. この場合、位相の補正を行った後、ガウス分布の中心に原点があり、負の周波数が配列の終わりまで回り込む変換が得られます。**次のコマンドを使って、それを見ることができます。**



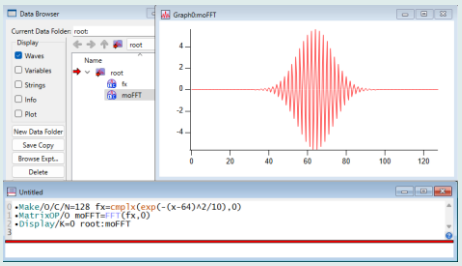

```
Rotate 64, W_FFT
```

例３：MatrixOP を使った FFT

MatrixOP はウェーブスケーリングを考慮しないため、位相による乗算はもう少し複雑になります。
単純な 1D FFT は次のように実行できます。

1. コマンドウィンドウで次を実行します。

```
Make/O/C/N=128 fx=cmplx(exp(-(x-64)^2/10),0)
```



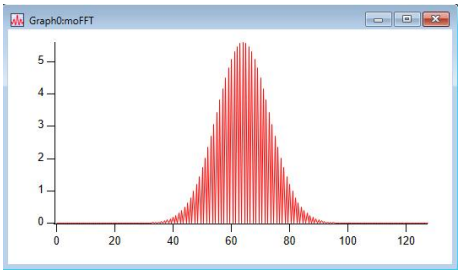
FFT を実行します。

```
MatrixOP/O moFFT=FFT(fx,0)
```

FFT 関数の第２引数として渡されるパラメーター 0 は、複素数出力と事後スワップ用のフラグが設定された順方向 FFT コマンドを実行することと同等です。

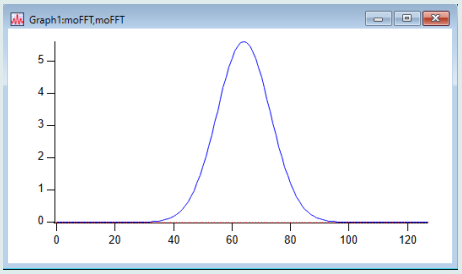
2. 位相補正を行うために、コマンドウィンドウで以下を実行します。

```
Variable/G start=-1/(2*DimDelta(fx,0))
Variable/G delta=1./(DimSize(fx,0)*DimDelta(fx,0))
moFFT*=exp(cmplx(0,2*pi*64*(start+p*delta)))
```



3. コマンドウィンドウで次を実行し、グラフを修正します。

```
Display /W=(35,44,430,252)/K=1  moFFT,moFFT
ModifyGraph rgb(moFFT)=(0,0,65535)
ModifyGraph
      cmplxMode(moFFT)=1,cmplxMode(moFFT#1)=2
```

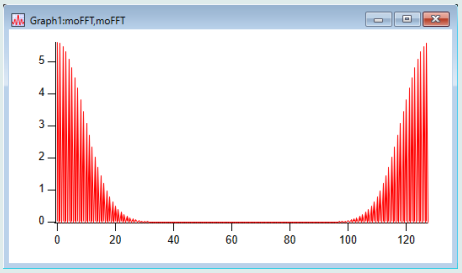


ここで、start はナイキスト周波数の負の値に相当し、delta は moFFT における FT ウェーブの周波数間隔です。
(start+p*delta) という表現は、例１の x に対応します。

例４：事後スワップをしない MatrixOP FFT

1. コマンドウィンドウで次を実行します。

```
Make/O/C/N=128 fx=cmplx(exp(-(x-64)^2/10),0)
```



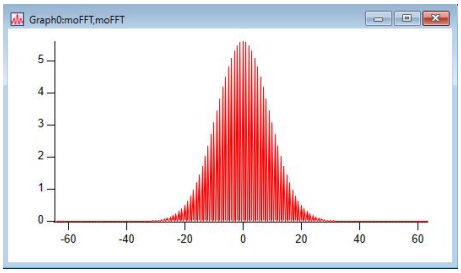
FFT/Z に相当する MatrixOP の関数は以下の通りです。

```
Variable/G start=-1/(2*DimDelta(fx,0))
Variable/G
      delta=1./(DimSize(fx,0)*DimDelta(fx,0))
MatrixOP/O moFFT=FFT(fx,2)
```

```
moFFT*=exp(cmplx(0,2*pi*64*(start+p*delta)))
Display/W=(35,44,430,252)/K=1  moFFT,moFFT
```

2. ここでも、表示を中央にシフトさせることができます。
コマンドウィンドウで次を実行し、グラフを修正します。

```
Rotate 64, moFFT
```



例５：逆フーリエ変換（IFFT）

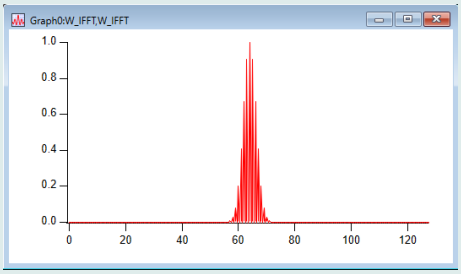
デフォルトの IFFT は、デフォルトの順方向 FFT を元の形に戻すように設計されています。

1. コマンドウィンドウで次を実行します。

```
Make/O/C/N=128 fx=cmplx(exp(-(x-64)^2/10),0)
```

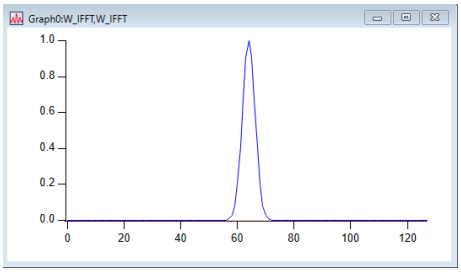
IFFT を実行します。

```
FFT/OUT=1/DEST=W_FFT fx
IFFT/C /DEST=W_IFFT W_FFT
Display /W=(35,44,430,252)/K=1  W_IFFT,W_IFFT
```



2. コマンドウィンドウで次を実行し、グラフを修正します。

```
ModifyGraph rgb(W_IFFT)=(0,0,65535)
ModifyGraph
      cmplxMode(W_IFFT)=1,cmplxMode(W_IFFT#1)=2
ModifyGraph mode(W_IFFT)=0
```



例６：畳み込み

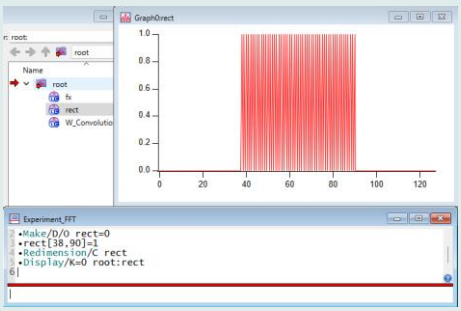
畳み込み演算では、線形、円形、非因果の３種類の畳み込みが可能です。
以下のコマンドを使って、３つすべてを試します。

1. コマンドウィンドウで次を実行します。

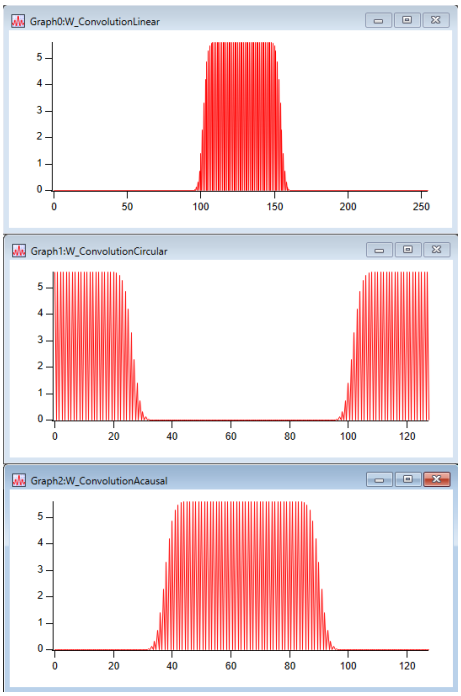
```
Make/O/C/N=128 fx=cmplx(exp(-(x-64)^2/10),0)
```

また、矩形関数ウェーブを作成します。

```
Make/D/O rect=0
rect[38,90]=1
Redimension/C rect
```



2. コマンドウィンドウで次を実行し、矩形関数を使った畳み込みを実行します。




```
Duplicate/O fx,W_ConvolutionLinear
Convolve rect, W_ConvolutionLinear

Duplicate/O fx,W_ConvolutionCircular
Convolve/C rect, W_ConvolutionCircular

Duplicate/O fx,W_ConvolutionAcausal
Convolve/A rect, W_ConvolutionAcausal
```

3. MatrixOP を使うと、FFT を利用して因果性のない円形の畳み込みを計算することができます。

```
MatrixOP/O
      circConvolution=IFFT(FFT(fx,0)*FFT(rect,0),0)
MatrixOP/O
      acausalConvolution=IFFT(FFT(fx,0)*FFT(rect,0),4)
```

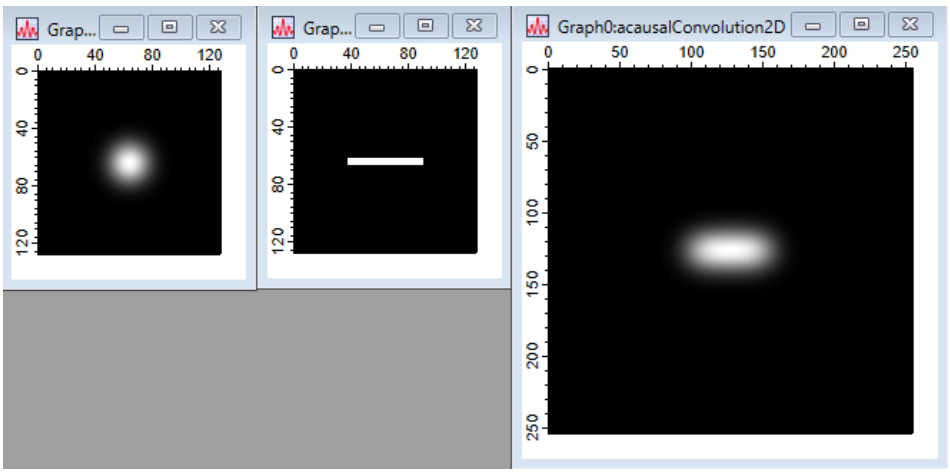
4. 最後の IFFT 関数に渡されたパラメーター 4 は、変換後に結果を交換するよう操作に指示します。あるいは、以下を実行することもできます。

```
MatrixOP/O circConvolution2=Convolve(fx,rect,0)
MatrixOP/O acausalConvolution2=Convolve(fx,rect,4)
```

例 7 ：2D 畳み込み

この例では、前の例で示した 1D の非因果畳み込みを 2D へ拡張します。
強調のため非対称な矩形を選択します。

```
Make/O/C/N=(128,128) rect2D=0
rect2D[38,90][62,66]=cmplx(1,0)
Make/O/C/N=(128,128) fx2d=cmplx(gauss(x,64,10,y,64,10),0)
MatrixOP/O acausalConvolution2D=Convolve(fx2d,rect2d,4)
NewImage/K=0 root:fx2d
NewImage/K=0 root:rect2D
NewImage/K=0 root:acausalConvolution2D
```

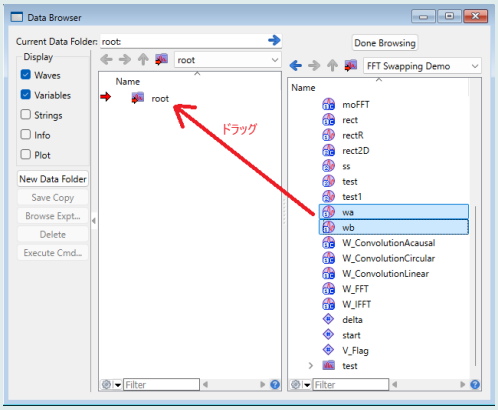


例８：MatrixOp の畳み込みと相関

この例では、以下の非対称入力 wa および wb を使った畳み込みと相関の結果を比較します。

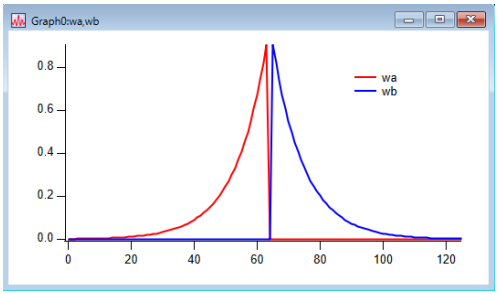
1. 新しいエクスペリメントを使う場合は、ウェーブはデモエクスペリメントから wa と wb をコピーして使います。

（Data Browser の Browse Expt ボタンで FFT Swapping Demo を選択して操作します）



2. **コマンドウィンドウで次を実行します。**

```
Display/K=0 wa,wb
ModifyGraph rgb(wb)=(0,0,65535)
ModifyGraph lsize(wa)=1.5
ModifyGraph lsize(wb)=1.5
Legend/C/N=text0/F=0/A=MC
```



3. **コマンドウィンドウで次を実行します。**

```
MatrixOP/O conv1d=Convolve(wa,wb,4)
MatrixOP/O corr1dwawb=correlate(wa,wb,4)
MatrixOP/O autoCorr=correlate(wa,wa,4)
Display/K=0 conv1d,corr1dwawb,autoCorr
ModifyGraph rgb(autoCorr)=(0,0,65535)
ModifyGraph rgb(corr1dwawb)=(0, 65535,0)
ModifyGraph lsize(conv1d)=1.5
ModifyGraph lsize(corr1dwawb)=1.5
ModifyGraph lsize(autoCorr)=1.5
Legend/C/N=text0/F=0/A=MC
```