# CONTENTS

# ビジュアルヘルプ – 分析（1）

## 分析

Igor は多様な分析機能を備えています。
ここでは、カーブフィッティング、信号処理、関数解析、画像処理以外の分析について説明します。
これらは別項で扱われています。

## 多次元ウェーブの分析

Igor Pro の分析操作の多くは 1 次元データを対象としています。
ただし、Igor Pro には多次元データの分析のための以下の機能が含まれています。

- 多次元ウェーブの代入
- 行列数学演算
- MatrixOp 演算
- 疎行列
- 多次元 FFT
- 2D および 3D 画像処理演算
- 2D および 3D 補間演算と関数

これらのトピックの一部は、ヘルプ Multidimensional Waves と Image Processing で議論されています。

1 次元データ専用に設計された分析コマンドは数多く存在します。
これらのコマンドのダイアログには多次元ウェーブは表示されません。
コマンドラインや Igor プロシージャから多次元ウェーブに対してこれらのコマンドを呼び出すと、Igor は多次元ウェーブを 1 次元ウェーブとして扱います。
例えば、Histogram コマンドは、n 行 m 列の 2 次元ウェーブを、n*m 行の 1 次元ウェーブとして扱います。
場合によっては（WaveStats など）、このコマンドが役立ちます。
他の場合には、まったく意味をなさないでしょう。

## ウェーブフォーム vs XY データ

Igor はウェーブフォームデータの処理に高度に適応しています。
ウェーブフォームでは、データ値は X 次元で等間隔に配置されます。
ヘルプ The Waveform Model of Data を参照してください。

データが等間隔で配置されている場合、SetScale コマンドを使って間隔を指定します。
これは非常に重要です。
組み込みの分析コマンドや関数のほとんどが、正しく動作するためにこの間隔を知る必要があるためです。

データが等間隔でない場合は、XY ペアとして扱うことができます。
これについては、ヘルプ The XY Model of Data で説明しています。
Igor の分析コマンドと関数の中には、XY ペアを直接処理できないものもあります。
これらを使うには、XY ペアをウェーブフォーム表現にするか、組み込みルーチンを基盤とする Igor プロシージャを使う必要があります。

## XY データをウェーブフォームに変換

XY データを分析するには、均一間隔のウェーブフォーム表現に変換して分析することが最良の方法である場合があります。
ほとんどの解析コマンドはウェーブフォームデータの方が容易です。
FFT などのコマンドはウェーブフォームデータでしか実行できません。
多くの場合、XY データセットはほぼ均一間隔であるため、ウェーブフォーム変換後は極めて近い近似となります。

実際、他のプログラムからインポートされた XY データには、Igor では全く不要な X ウェーブが含まれていることがよくあります。
なぜなら、X ウェーブの値は実際には単純な「系列」（2.2、2.4、2.6、2.8 など、規則的な間隔で定義された値）であるため、ウェーブフォームへの変換は、SetScale（または Change Wave Scaling ダイアログ）を使って Y データウェーブに正しい X スケーリングを割り当てるだけの簡単な作業です。

```
SetScale/P x, xWave[0], xWave[1]-xWave[0], yWave
```

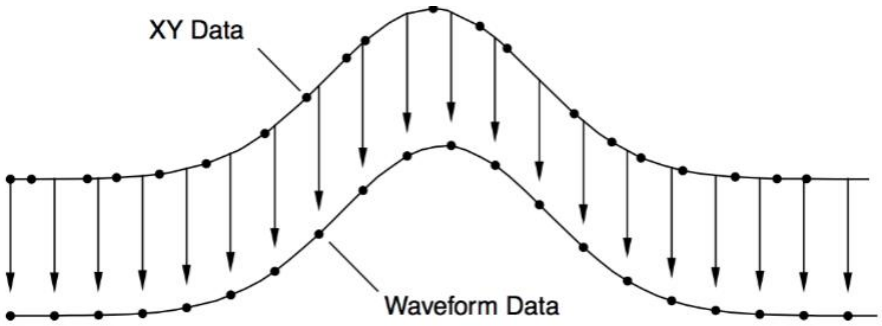
この X ウェーブは不要であり、破棄できます。

```
KillWaves/Z xWave
```

XY Pair to Waveform パネルは、X ウェーブに連続データが含まれていると検出された時に、Y ウェーブの X スケーリングを設定するために使用できます。
詳細は「XY Pair to Waveform パネルを使う」のセクションを参照してください。

X ウェーブが連続ウェーブでない場合、XY データのウェーブフォームを作成するには補間を使う必要があります。
補間は、XY ペアを等間隔でサンプリングすることでウェーブフォームを生成します。

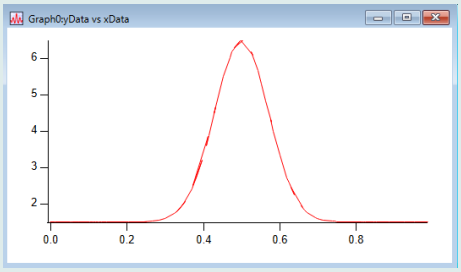
下図は、上側の曲線を定義する XY ペアが、下側の曲線を定義する等間隔のウェーブフォームを計算するために補間される方法を示しています。
各矢印は補間されたウェーブフォーム値を示しています。




Igor はこの補間を行うための 3 つのツールを提供します。
XY Pair to Waveform パネル、組み込みの interp 関数、および Interpolate2 コマンドです。
サンプル XY データを使ってこれらのツールを説明します。

**1. 新しいエクスペリメントを作成して、以下のコマンドでサンプルデータを作成し、グラフに表示します。**

```
Make/N=100 xData = .01*x + gnoise(.01)
Make/N=100 yData = 1.5 + 5*exp(-((xData-.5)/.1)^2)
Display yData vs xData
```




これによりガウス形状が生成されます。
XY ペアの x ウェーブにはノイズが含まれているため、X 次元におけるデータ間隔は均一ではありません。

**2.** x データはおおむね 0 から 1.0 の範囲ですが、ノイズが含まれているため単調増加とは限りません。つまり、あるポイントから次のポイントへ進むにつれて、x データは通常増加しますが、特定のポイントでは減少する場合もあります。
この問題はデータを並べ替えることで修正できます。
次を実行します。

```
Sort xData, xData, yData
```

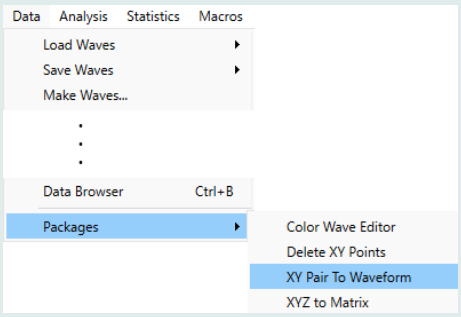
このコマンドは xData ウェーブをソートキーとして使い、xData と yData の両方をソートします。
これにより、あるポイントから次のポイントへと進むにつれて xData が常に増加するようにします。

## XY Pair to Waveform パネルを使う

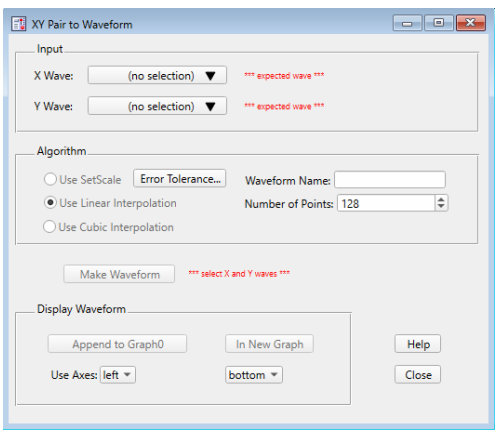
XY Pair to Waveform パネルは、X ウェーブのデータに対する自動解析に基づく SetScale または Interpolate2 コマンドを使って XY データからウェーブフォームを生成します。

必要なステップは次の通りです。

**1. メニュー Data→Packages から XY Pair to Waveform を選択します。**

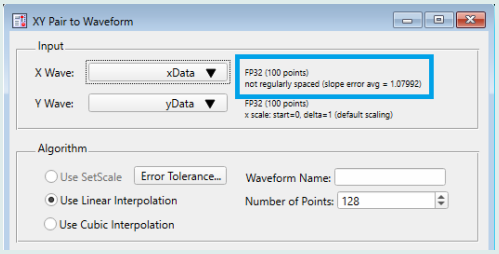



**2.** パネルが表示されます。

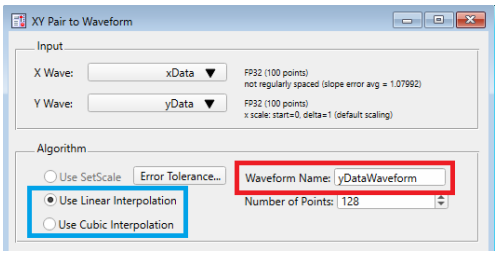


**3. ポップアップメニューで X ウェーブと Y ウェーブ（xData と yData）を選択します。**

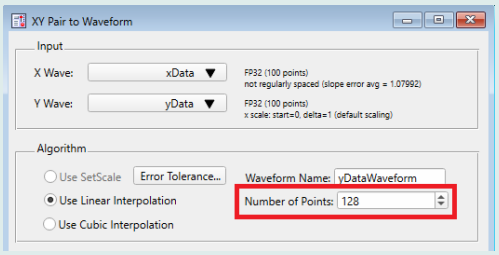
この例の xData ウェーブを解析すると「等間隔でない（例えば、勾配誤差平均=0.52...）」と判定され、SetScale 関数は yData をウェーブフォームに変換するのに適していません。



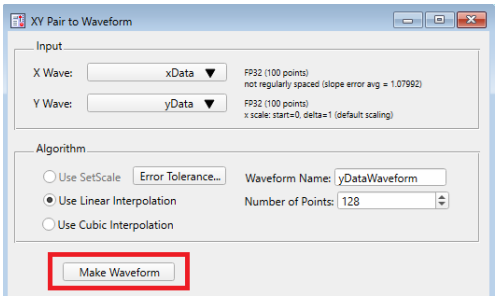

**4.** Use Interpolate が選択されていると、出力用のウェーブフォーム名が必要です。
**有効なウェーブフォーム名を入力してください。**




**5. 出力ポイント数を設定します。**
ウェーブの長さとほぼ同数を使うのが最初の試行として適しています。
元の形状への忠実度が不十分な場合は、後でより大きな数値を選択できます。
適切な数値は X 値の不均一さに依存します。
不均一さが大きいほど多くのポイントを使ってください。

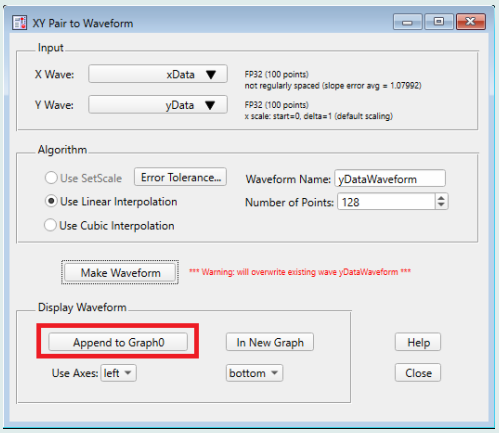



**6. Make Waveform ボタンをクリックします。**




**7.** データの XY 表現とウェーブフォーム表現を比較するには、ウェーブフォームを XY ペアを表示するグラフに追加します。

**グラフを最前面のグラフにした後、「Append to <グラフ名>」ボタンをクリックします。**



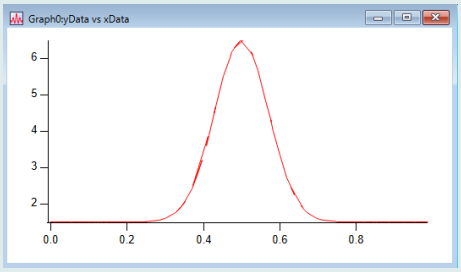

**8.** Number of Points を修正し、Make Waveform をクリックすると、以前に作成したウェーブフォームをその場で上書きできます。

## Interp 関数を使う

ガウス関数のウェーブフォーム版を作成するには、interp 関数を使用できます。
必要な手順は以下の通りです。

**1. 新しいエクスペリメントを作成して、以下のコマンドでサンプルデータを作成し、グラフに表示します。**

```
Make/N=100 xData = .01*x + gnoise(.01)
Make/N=100 yData = 1.5 + 5*exp(-((xData-.5)/.1)^2)
Display yData vs xData
Duplicate yData, wData
```

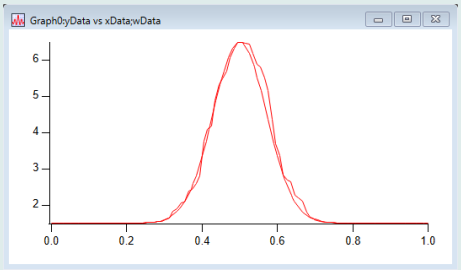



**2. SetScale コマンドを使って、ウェーブフォームにおける X 値の範囲を定義します。**

```
SetScale/I x 0, 1, wData
```

**3. ウェーブフォーム表示と XY 表示を比較するため、ウェーブフォームをグラフに追加します。**

```
AppendToGraph wData
```

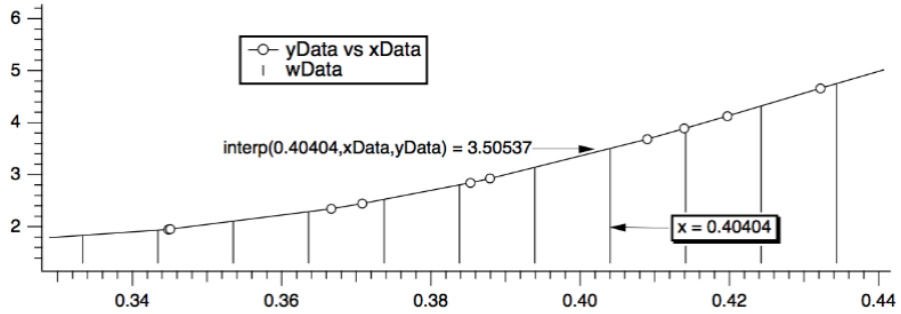



これらのコマンドが何をしているのか、詳しく見てみます。

まず、yData を複製し、新しいウェーブフォーム wData を作成しました。
複製（Duplicate）を使ったため、wData は yData と同じポイント数になります。
異なるポイント数のウェーブフォームを作成することも可能です。
その場合は、Duplicate（複製）ではなく Make（作成）コマンドを使います。

SetScale コマンドは、wData ウェーブフォームの X 方向スケーリングを設定します。
この例では、wData の X 値を 0 から 1.0 まで（1.0 を含む）に設定しています。
これにより、ウェーブフォーム表示は X 方向（0 から 1.0）に等間隔で 100 個の値を含むことになります。

最後のステップでは、ウェーブフォーム代入を使って wData のデータ値を設定します。
この代入は、wData の各ポイントについて右辺の式を 1 回評価します。
各評価において、x は 0 から 1.0 までの異なる値を取ります。
interp 関数は、x における曲線 yData vs xData の値を返します。
例えば、x=.40404（wData の 40 番目のポイント）は XY 曲線の 2 ポイント間に位置します。
interp 関数はそれらの値の間を線形補間し、データ値を 3.50537 と推定します。



これらの計算を Igor プロシージャにまとめ、任意の XY ペアのウェーブフォームバージョンを作成できるようにします。

```
Function XYToWave1(xWave, yWave, wWaveName, numPoints)
        Wave/D xWave                            // XY ペアの X ウェーブ
        Wave/D yWave                            // XY ペアの Y ウェーブ
        String wWaveName                        // 新しいウェーブフォームウェーブの名前
        Variable numPoints                      // ウェーブフォームのポイント数

        Make/O/N=(numPoints) $wWaveName         // ウェーブフォームの作成
        Wave wWave= $wWaveName
        WaveStats/Q xWave                       // x 座標の範囲を見つける
        SetScale/I x V_min, V_max, wWave        // ウェーブに X スケーリングを設定
        wWave = interp(x, xWave, yWave)         // 補間を実行
End
```

この関数は WaveStats コマンドを使って、XY ペアの X 範囲を求めます。
WaveStats は変数 V_min と V_max（その他も）を作成します。
詳細はヘルプ Accessing Variables Used by Igor Operations を参照してください。

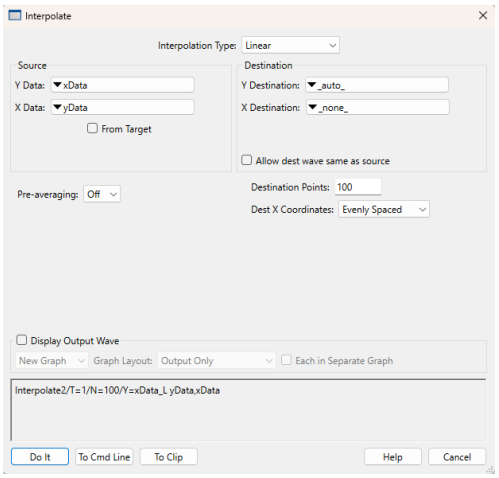
この関数は、入力ウェーブが既にソート済みであることを前提としています。
ソート処理を省略したのは、ソートが副作用となるためです。
また、プロシージャには非自明な副作用を持たせないことが望ましいためです。

WaveMetrics が提供する XYToWave1 関数を使うには、「XY Pair To Waveform」プロシージャファイルをインクルードしてください。
プロシージャファイルのインクルード方法については、ヘルプ The Include Statement を参照してください。

入力データに空白（NaN）が含まれている場合、interp 関数はウェーブフォームにも空白を生成します。
次のセクションで説明する Interpolate2 コマンドは、データの欠損部分を補間し、ウェーブフォームに空白を生成しません。

## Interpolate2 コマンドを使う

Interpolate2 コマンドは、線形補間だけでなく、３次スプラインおよび平滑化スプライン補間も提供します。
さらに、入力のソートを必要とせず、自動的に目的のウェーブフォームを作成し、その X スケーリングを設定することができます。
また、対話的に簡単に使用できるダイアログも備わっています。




前のセクションの XY サンプルデータで使うには、Analysis→Interpolate を選択し、右のようにダイアログを設定します。

Y Destination で _auto_ を選択すると、入力 Y Data ウェーブの名前に「_L」を付加して宛先ウェーブフォームが自動的に命名されます。
X Destination で _none_ を選択すると、Interpolate に新しい XY ペアを作成するのではなく、入力された XY ペアからウェーブフォームを作成するよう指示します。

以下は、interp 関数ではなく Interpolate2 コマンドを使う XYToWave1 関数の書き換えです。

```
Function XYToWave2(xWave, yWave, wWaveName, numPoints)
        Wave xWave                          // XY ペアの X ウェーブ
```

```
        Wave yWave                      // XY ペアの Y ウェーブ
        String wWaveName                // 新しいウェーブフォームウェーブの名前
        Variable numPoints              // ウェーブフォームのポイント数

        Interpolate2/T=1/N=(numPoints)/Y=$wWaveName xWave, yWave
End
```

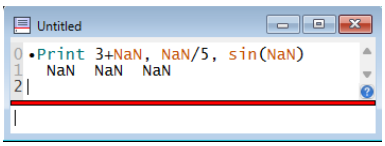
入力データ内の空白は無視されます。

Interpolate2 の詳細については、「Interpolate2 コマンド」のセクションを参照してください。

## 欠損値の取り扱い

Igor では、欠損値は「数値ではない（Not a Number）」を意味する NaN 値で表されます。
欠損値は「空白」とも呼ばれます。
これは、表の中で空白のセルとして表示されるためです。

NaN が任意の値と算術演算されると、結果は NaN となります。
これを確認するには、次のコマンドを試してください。




```
Print 3+NaN, NaN/5, sin(NaN)
```

定義上、NaN は何にも等しくありません。
したがって、この記述の条件：

```
if (myValue == NaN)
```

は常に偽となります。

回避策として NumType 関数を使います。

```
if (NumType(myValue) == 2)   // これは NaN か？
```

NaN 値の詳細については、ヘルプ NaNs, INFs and Missing Values も参照してください。

Igor のいくつかのルーチンは、欠損値を無視することで処理します。
CurveFit がその一例です。
他のルーチンは、欠損値が存在する場合に予期しない結果を生成する可能性があります。
例としては、FFT 演算や面積関数、平均関数などが挙げられます。

欠損値に対処するためのいくつかの戦略を以下に示します。

### 欠損値を別の値で置き換える

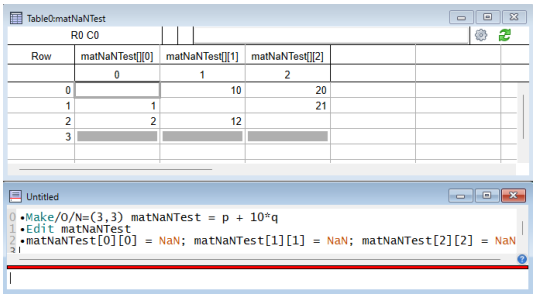
次の記述でウェーブ内の NaN を置換できます。

```
wave0 = NumType(wave0)==2 ? 0:wave0          // NaN をゼロに置換
```

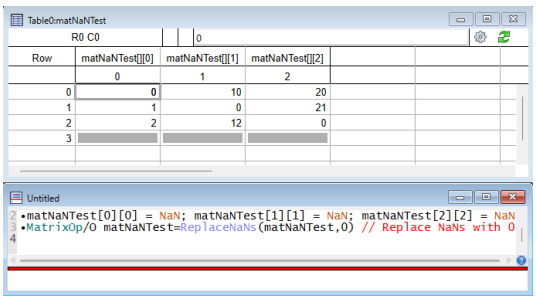
「:?」演算子の意味が不明な場合は、ヘルプ「? :」を参照してください。

多次元ウェーブの場合、MatrixOp を使って NaN を置換できます。
例えば：

```
Make/O/N=(3,3) matNaNTest = p + 10*q
Edit matNaNTest
matNaNTest[0][0] = NaN; matNaNTest[1][1] = NaN;
matNaNTest[2][2] = NaN
```



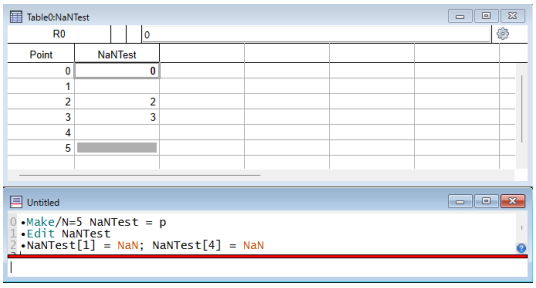

```
MatrixOp/O matNaNTest=ReplaceNaNs(matNaNTest,0)
                    // NaN をゼロに置換
```



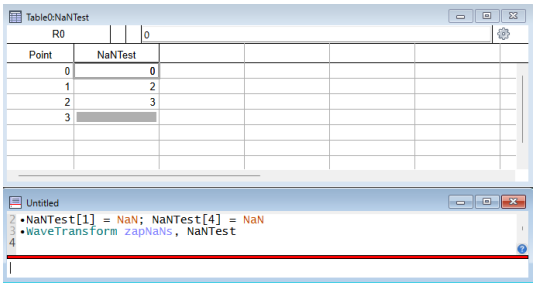

## 欠損値を除去する

1D ウェーブの場合、WaveTransform の zapNaNs を使って NaN を除去できます。
例えば：

```
Make/N=5 NaNTest = p
Edit NaNTest
NaNTest[1] = NaN; NaNTest[4] = NaN
```




```
WaveTransform zapNaNs, NaNTest
```




X ウェーブまたは Y ウェーブのいずれかに NaN が現れた場合、XY ペアから NaN を除去する組み込みコマンドは存在しません。
ただし、Help→Windows→WM Procedures Index からアクセス可能な「Remove Points」WaveMetrics プロシージャファイル内の RemoveNaNsXY プロシージャを使ってこれを実行できます。

多次元ウェーブから NaN を除去するコマンドは存在しません。
各 NaN が出現した行全体と列全体を削除する必要があるためです。

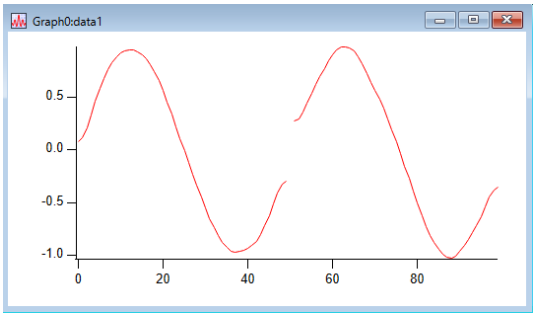
## データの不備を回避する

多くの解析ルーチンはデータのサブレンジ（一部）で動作可能です。
多くの場合、欠損値を含むデータ領域を単に回避すれば十分です。

他のケースでは、データのサブセットを抽出し、それを処理した後、変更したデータを元のウェーブに戻すことも可能です。

次は抽出・修正・置換の例です（Smooth が NaN を適切に処理しているにもかかわらず）。

```
Make/N=100 data1= sin(P/8)+gnoise(.05); data1[50]= NaN
Display data1
// 最初のセットの処理開始
Duplicate/R=[0,49] data1,tmpdata1
Smooth 5,tmpdata1
// 修正したデータを書き戻す
data1[0,49]= tmpdata1[P]
// ２番目のセットの処理開始
Duplicate/O/R=[51,] data1,tmpdata1
Smooth 5,tmpdata1
data1[51,]= tmpdata1[P-51]
KillWaves tmpdata1
```




## 欠損データを補間値で置き換える

NaN データ値を、Smooth、Loess、または Interpolate2 コマンドを使って平滑化または補間した値に置き換えることで、NaN を許容しないコマンドを実行する前に NaN データ値を置換できます。

## Interpolate2 コマンドを使って欠損データを補間する

ソースポイントと同じ数のポイントを宛先ポイントに使うことで、他のデータを変更せずに NaN を置き換えることができます。

ウェーブフォームデータがある場合、データを複製し、元のデータと同じ数のポイントを使って線形補間を実行します。
例えば、100 個のデータポイントがあると仮定します。

```
Duplicate data1,data1a
Interpolate/T=1/N=100/Y=data1a data1
```

XY データがある場合、Interpolate2 コマンドは入力 x 値を出力 X ウェーブに含める機能を備えています。
例えば：

```
Duplicate data1, yData1, xData1
xData1 = x
Display yData1 vs xData1
Interpolate2/T=1/N=100/I/Y=yData1a/X=xData1a xData1,yData1
```

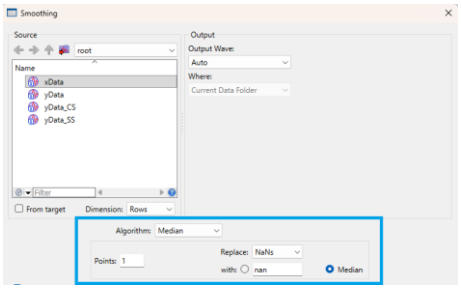
データに対してコマンドを実行した後、変更したデータを元のウェーブに戻しつつ、元の欠損値を維持したい場合、以下のようなウェーブ代入を使用できます。

```
yData1 = (numtype(yData1) == 0) ? yData1 : yData1a
```

この手法は、Loess と Smooth コマンドによって生成された補間結果を使って適用することも可能です。

### 中央値（Median）平滑化を使って欠損データを補間する

Smoothing ダイアログを使うと、各 NaN を周囲の値の中央値で置き換えることができます。




アルゴリズムで Median を選択し、Replace ポップアップから NaNs を選択し、with: ラジオボタンで Median を選択します。
中央値計算に使う周囲のポイントの数を入力します（奇数が最適です）。

NaN を上書きするか、結果で新しいウェーブフォームを作成するかを選択できます。
Smoothing ダイアログは次のようなコマンドを生成します。

```
Duplicate/O data1,data1_smth;DelayUpdate
Smooth/M=(NaN) 5, data1_smth
```

## 補間

Igor Pro には、様々な用途向けに設計された複数の補間ツールが用意されています。
これらを以下の表にまとめます。

| **データ** | **コマンド／関数** | **補間手法** |
|---|---|---|
| 1D ウェーブ | ウェーブ代入（例：val=wave(x)） | 線形 |
| 1D ウェーブ | Smooth | 移動平均、平均、二項式、Savitsky-Golay |
| 1D XY ウェーブ | interp | 線形 |
| 1D 単体 or XY ウェーブ | Interpolate2 コマンド | 線形、３次スプライン、平滑化スプライン |
| 1D 単体 or 2D 単体 or XY | Loess | 局所加重回帰 |
| トリプレット XYZ ウェーブ | ImageInterpolate | Voronoi |
| 1D X, Y, Z ウェーブ | メニュー Data→Packages→XYZ to Matrix | Voronoi |
| 1D X, Y, Z ウェーブ | Loess | 局所加重回帰 |
| 2D ウェーブ | ImageInterpolate | Bilinear、スプライン、Kriging、Voronoi |
| 2D ウェーブ | Interp2D | Bilinear |
| 2D ウェーブ | SphericalInterpolate | Voronoi |
| 3D ウェーブ | Interp3d | extractSurface、Trilinear |
| | Interp3DPath | extractSurface、Trilinear |
| | ImageTransform | extractSurface、Trilinear |
| 3D 散布データ | Interpolate3D | Barycentric |

| | | |
|---|---|---|
| 4D ウェーブ | Interp4D | 4D 線形 |
| | Interp4dPath | 4D 線形 |

この表に記載されているすべての補間手法は、２つの共通のステップで構成されています。
最初のステップでは、補間位置に最も近いデータポイントを特定し、２番目のステップでは、隣接する値とその相対的な近接度を用いて補間値を計算します。
具体的な詳細については、個々のコマンドまたは関数のドキュメントを参照してください。

### 補間のデモエクスペリメント

Smooth Curve Through Noise（File→Example Experiments→Feature Demos）

## Interpolate2 コマンド

Interpolate2 コマンドは、1D ウェーブフォームデータおよび XY データに対して、線形補間、３次スプライン補間、平滑化スプライン補間を実行します。
３次スプライン補間は書籍「Numerical Recipes in C」のルーチンに基づいています。
平滑化スプラインは書籍「Smoothing by Spline Functions」、Christian H. Reinsch、Numerische Mathematic 10、177-183 (1967) に基づいています。

Igor 7 より前では、Interpolate2 は Interpolate XOP の一部として実装されていました。
現在は組み込み関数となっています。

Interpolate XOP は、構文が若干異なる古いコマンドである Interpolate も実装していました。
Interpolate コマンドを使っていた場合は、Interpolate2 の使用に切り替えることを推奨します。

線形補間の主な用途は、XY ペアのウェーブを１つのウェーブに変換し、等間隔の X 値で Y 値を含むようにすることです。
これにより、FFT など等間隔データが必要な Igor コマンドを利用できるようになります。

３次スプライン補間は、任意の XY データに滑らかな曲線を当てはめるのに最も役立ちます。
結果として得られる曲線には、元のデータとは無関係な特徴が含まれる可能性があるため、３次スプラインを美的目的ではなく分析目的で使う時には注意が必要です。

線形スプライン補間と３次スプライン補間の両方は、出力曲線をすべてのデータポイントを通るように制約されるため、データポイントの数が少ない場合に最も効果を発揮します。
平滑化スプラインはこの制約を持たないため、大規模でノイズの多いデータセットでも良好に機能します。

Interpolate2 コマンドには「事前平均化」と呼ばれる機能があり、データポイント数が非常に多い場合に使用できます。
事前平均化は、平滑化スプラインをサポートする前に、ノイズの多い大規模データセットに３次スプラインを適用する方法として Interpolate2 に追加されました。
現在では、事前平均化の代わりに平滑化スプラインを試すことを推奨します。

Smooth Curve Through Noise サンプルエクスペリメント（File→Example Experiments→Feature Demos）は、スプライン補間を説明しています。

### スプライン補間の例

Interpolate2 について完全に議論する前に、まず簡単な例を見てみます。

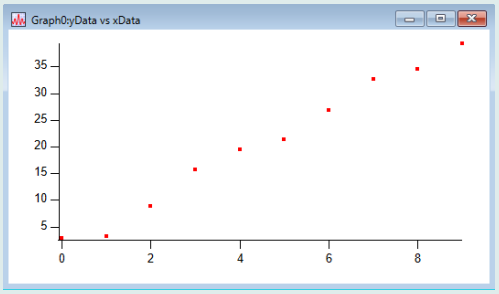
**注記**　　ヘルプの内のコマンドは、コマンドを選択して、Control+Enter キーを押すことで実行できます。

まず、以下のコマンドを使ってサンプルの XY データを作成します。

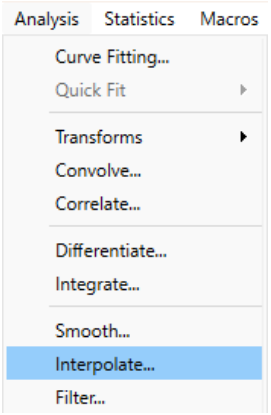
```
Make/N=10 xData, yData                          // ソースデータを作成
xData = p; yData = 3 + 4*xData + gnoise(2)      // サンプルデータを作成
Display yData vs xData                          // グラフを作成
Modify mode=2, lsize=3                          // ソースデータをドットで表示
```

**1. 新しいエクスペリメントを作成して、コマンドウィンドウで次を実行します。**

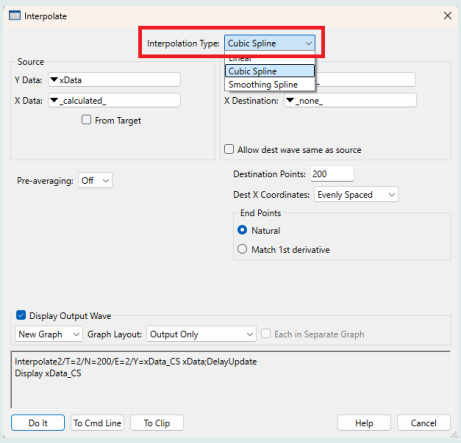
```
// ソースデータを作成
Make/N=10 xData, yData
// サンプルデータを作成
xData = p; yData = 3 + 4*xData + gnoise(2)
// グラフを作成
Display yData vs xData
// ソースデータをドットで表示
Modify mode=2, lsize=3
```




**2. 次に、Analysis→Interpolate を選択して、Interpolate ダイアログを表示します。**

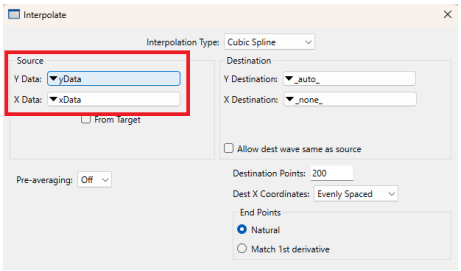



**3. Interpolation Type を Cubic Spline に設定します。**



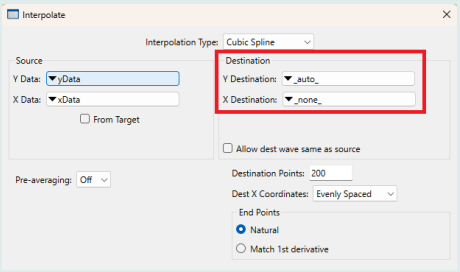

**4. Y Data ポップアップメニューから yData を選択します。**

**X Data ポップアップメニューから xData を選択します。**



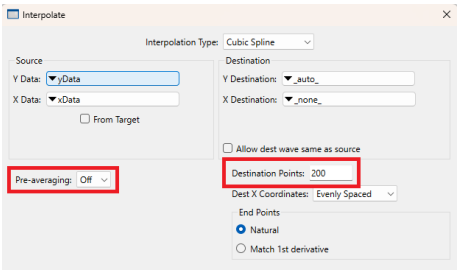
**5.** **Y Destination ポップアップメニューから _auto_ を選択します。**

**X Destination ポップアップメニューから _none_ を選択します。**

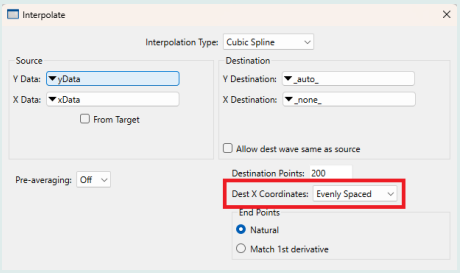



**6.** **Destination Points に 200 を入力します。**

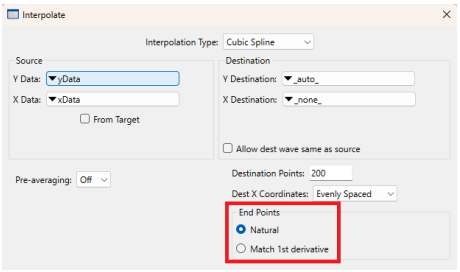
**Pre-averaging ポップアップメニューを Off にします。**




**7.** **Dest X Coords ポップアップメニューから Evenly Spaced を選択します。**

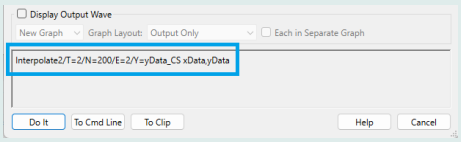



**8.** **End Points セクションで Natural をクリックします。**




**9.** **ダイアログが以下のコマンドを生成したことに注目してください。**

```
Interpolate2/T=2/N=200/E=2/Y=yData_CS xData, yData
```

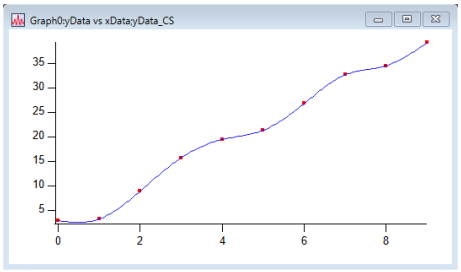



これは、Interpolate2 が yData を Y ウェーブとして、xData を X ウェーブとして使い、yData_CS を宛先ウェーブとして生成することを示しています。
_CS は「cubic spline」を意味します。

**10.** **補間を実行するために「Do It」ボタンをクリックします。**

**次に、次のコマンドを使って yData_CS　宛先ウェーブをグラフに追加します。**

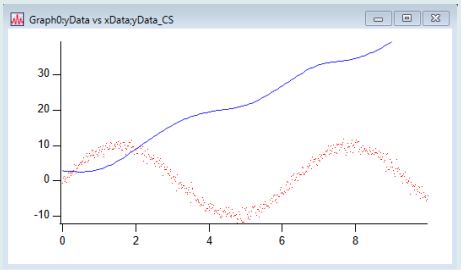
```
AppendToGraph yData_CS; Modify
                rgb(yData_CS)=(0,0,65535)
```




**11.** 入力データポイントの数を増やして試してみます。

**以下を実行します。**

```
Redimension/N=500 xData, yData
// サンプルデータを生成
xData = p/50; yData = 10*sin(xData) + gnoise(1.0)
Modify lsize(yData)=1              // より小さなドット
```

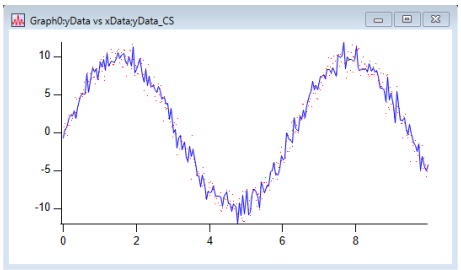



**12.** **次に、Analysis→Interpolate を選択して、Interpolate ダイアログを呼び出します。**

すべての設定は、前の手順で設定したままの状態であるはずです。

**Do It ボタンをクリックします。**

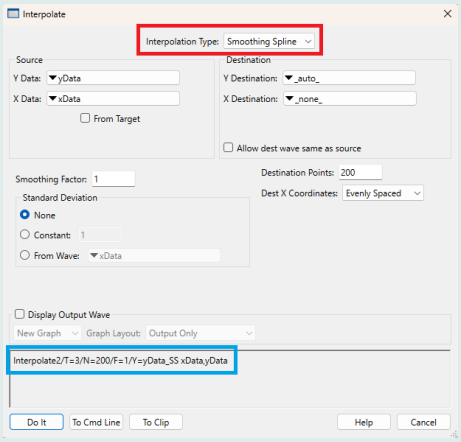
結果として得られる３次スプラインは、入力データポイントすべてを通過しようとする点に注意してください。
これは通常、望ましい結果ではありません。




**13.** **再度 Analysis→Interpolate を選択し、以下の変更を行います。**

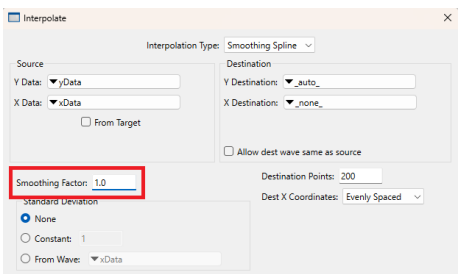
**Interpolation Type ポップアップメニューから、Smoothing Spline を選択します。**

生成されたコマンドが yData_SS という名前のウェーブを参照していることに注意してください。
これが出力ウェーブとなります。

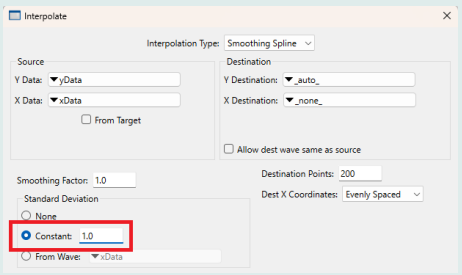



**14.** **Smoothing factor（平滑化係数）に 1.0 を入力してください。**

これは通常、開始値として適切な値です。

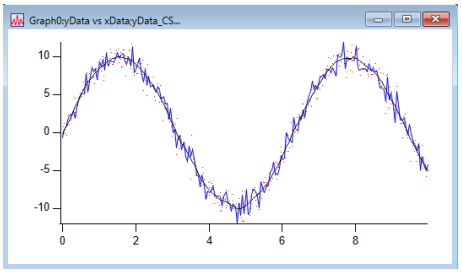


**15. Standard Deviation セクションで、Constant ラジオボタンをクリックし、標準偏差値として 1.0 を入力します。**




1.0 が正しい値である理由は、上記の `gnoise(1.0)` 項の結果として、データに標準偏差 1.0 のノイズが含まれていることがわかっているためです。

**16. Do It ボタンをクリックして、補間を実行します。**




**次のコマンドを使って、yData_SS 宛先ウェーブをグラフに追加します。**

```
AppendToGraph yData_SS; Modify rgb(yData_SS)=(0,0,0)
```

平滑化スプラインが、ノイズの多い大規模なデータセットに美しい曲線を描いていることに注目してください。
必要に応じてグラフウィンドウを拡大し、この様子を確認してください。

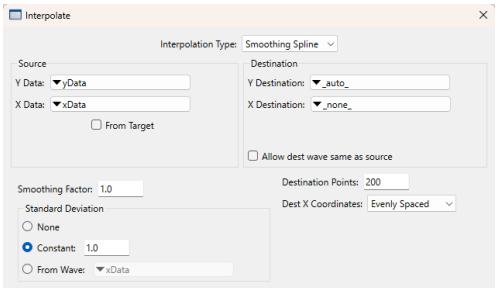
Smoothing Spline は、平滑化係数パラメーターまたは標準偏差パラメーターのいずれかを使って調整できます。
ノイズの標準偏差が不明な場合は、平滑化係数を 1.0 に設定したまま、異なる標準偏差値を試してください。
通常、妥当な値を見つけるのはそれほど難しくありません。

## Interpolate ダイアログ

メニュー Analysis→Interpolate を選択すると、Interpolate ダイアログが表示されます。
このダイアログでは、目的の補間タイプ、ソースウェーブ、出力ウェーブ、および出力ウェーブにおける補間ポイント数を選択できます。
このダイアログは Interpolate2 コマンドを生成します。
このコマンドは実行、クリップボードへのコピー、またはコマンドラインへのコピーが可能です。

Interpolation Type ポップアップメニューから、Linear、Cubic Spline、または Smoothing Spline を選択します。
Cubic Spline は、入力データセットが小さい場合に適しています。
Smoothing Spline は、入力データセットが大きくノイズが多い場合に適しています。

Cubic Spline を選択すると、Pre-averaging ポップアップメニューが表示されます。
事前平均化機能はほとんど必要なくなり、推奨されません。
事前平均化付き Cubic Spline の代わりに、Smoothing Spline を使ってください。




Smoothing Spline を選択すると、Smoothing Factor（平滑化係数）項目と Standard Deviant（標準偏差）コントロールが表示されます。
通常、Smoothing Factor は 1.0 に設定し、Standard Deviant の設定には定数モードを使うのが最適です。
次に、Y データのノイズの標準偏差の推定値を入力する必要があります。
その後、データに満足のいく平滑なスプラインが得られるまで、標準偏差のさまざまな値を試してみてください。
詳細は「Smoothing Spline のパラメーター」のセクションを参照してください。

Y Data と X Data ポップアップメニューから、補間したいデータを定義するウェーブを選択します。
X Data ポップアップからウェーブを選択した場合、Interpolate2 は XY 曲線をソースデータとして使います。
この曲線は、Y データウェーブの内容と X データウェーブの内容によって定義されます。
X Data ポップアップから _calculated_ を選択した場合、Y データウェーブの X 値と Y 値がソースデータとして使われます。

From Target チェックボックスをクリックすると、Source と Destination のセクションのポップアップメニューには、ターゲットのグラフまたはテーブル内のウェーブのみが表示されます。

X ウェーブと Y ウェーブは、同じ数のデータポイントを持つ必要があります。
データタイプが同じである必要はありません。

X と Y のソースデータは、Interpolate2 を使う前にソートする必要はありません。
必要に応じて、Interpolate2 は補間処理を行う前に、入力データのコピーをソートします。

ソースデータ内の NaN（欠損値）および INF（無限大値）は無視されます。
X 値または Y 値が NaN または INF であるポイントは、存在しないものとして扱われます。

目的のウェーブに配置したいポイント数を Destination Points ボックスに入力してください。
通常、200 ポイントで十分です。

Y Destination と X Destination のポップアップメニューから、補間結果を含むウェーブフォームを選択します。
ほとんどの場合、Y Destination ウェーブフォームには _auto_ を、X Destination ウェーブフォームには _none_ を選択します。
これにより、出力ウェーブフォームが得られます。
あまり一般的ではない用途に便利なその他のオプションについては、以降の段落で説明します。

Y Destination ポップアップから _auto_ を選択すると、Interpolate2 は Y 出力データを、Y データウェーブの名前にサフィックスを付けた名前のウェーブに入れます。
サフィックスは、線形（Linear）補間では「_L」、3次スプライン（Cubic Spline）補間では「_CS」、平滑化スプライン（Smoothing Spline）補間では「_SS」です。
例えば、Y データウェーブが「yData」という名前であれば、デフォルトの Y 出力ウェーブは「yData_L」、「yData_CS」、または「yData_SS」という名前になります。

X 出力ウェーブに _none_ を選択した場合、Interpolate2 は Y 出力データを Y 出力ウェーブに配置し、Y 出力ウェーブの X 方向スケーリングを X 出力データを表現するように設定します。

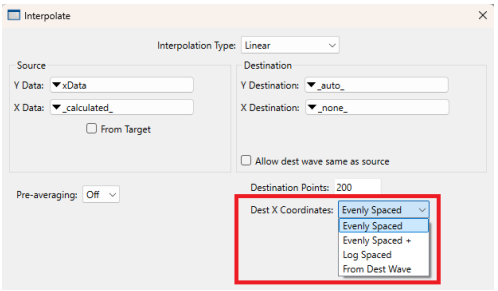
X Destination ポップアップから _auto_ を選択すると、Interpolate2 は X 出力データを、X データウェーブの名前から適切なサフィックスを付けた名前のウェーブに入れます。
X データウェーブが 「xData」 の場合、X 宛先ウェーブは「xData_L」、「xData_CS」または「xData_SS」 になります。
X ウェーブが存在しない場合、X 出力ウェーブ名は Y 出力ウェーブ名に文字「x」を追加して生成されます。
例えば、Y 出力ウェーブが「yData_CS」の場合、X 出力ウェーブは「yData_CSx」となります。

X および Y の宛先ウェーブの両方について、ウェーブが既に存在する場合、Interpolate2 はそれを上書きします。
ウェーブがまだ存在しない場合、Interpolate2 はそれを作成します。

Interpolate2 が呼び出された時点で既に存在している場合を除き、宛先ウェーブは倍精度になります。
この場合、Interpolate2 は単精度宛先ウェーブを単精度のまま残します。
その他の精度については、Interpolate2 は宛先ウェーブを倍精度に変更します。

Dest X Coordinates ポップアップメニューでは、補間を行う X 位置をコントロールできます。
通常は Evenly Spaced を選択してください。
これにより、X 入力値の範囲で等間隔に補間値が生成されます。

Evenly Spaced+ 設定は、Interpolate2 によって出力 X 値に入力 X 値がすべて含まれることを保証する点を除き、Evenly Spaced 設定と同じです。
通常、これは必要ありません。
X 宛先ウェーブに _none_ を選択した場合、このモードは使用できません。

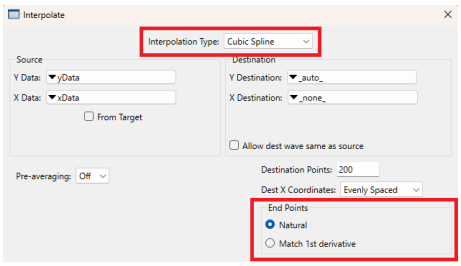



Log Spaced（対数間隔）設定は、対数軸上で出力を等間隔に表示します。
このモードでは、入力 X データの非正値は無視されます。
X ウェーブに _none_ を選択した場合、このモードは利用できません。
代替方法については「指数データの補間」のセクションを参照してください。

From Dest Wave 設定は、宛先ウェーブフォームの X 座標から出力 X 値を取得します。
宛先のポイント設定は無視されます。
例えば、入力データの一部を通してスプラインを取得するためにこれを使用できます。
このモードの補間を行う前に、宛先ウェーブフォームを作成する必要があります。
宛先がウェーブフォームの場合は、SetScale コマンドを使ってウェーブフォームの X 値を定義してください。
Interpolate2 はこれらの X 値で出力を計算します。
宛先が XY ペアの場合は、X ウェーブの値を設定します。
Interpolate2 はこれらの値をソートしたバージョンを作成し、これらの値で出力を計算します。
X ウェーブが元から逆順ソートされていた場合、Interpolate2 は出力を逆順にします。

End Points ラジオボタンは Cubic Spline にのみ適用されます。
これらは、ソースウェーブの最初と最後の区間における宛先ウェーブをコントロールします。

Natural は、宛先ウェーブの最初と最後のポイントにおいて、スプラインの 2 次微分をゼロに強制します。
Match 1st Derivative は、スプラインの傾きが、最初の入力ポイントと 2 番目の入力ポイントの間、および最後の入力ポイントと最後から 2 番目の入力ポイントの間に引かれた直線と一致するよう強制します。
ほとんどの場合、これらの選択肢のどれを使っても大きな違いはありません。




## 指数データの補間

指数関数的なプロセスから生じるデータなど、桁が違う値を含むデータを対数軸に対してプロットすることが一般的です。
このようなデータから補間データセットを作成する場合、元のデータそのものではなく、元のデータの対数を補間するのが最善であることが多いです。

Interpolate2 Log Demo エクスペリメント（フォルダー Igor Pro Folder→Examples→Feature Demos [メニューからは選択できません]）では、そのような補間を行う方法を示しています。
以下はそのデモエクスペリメントからの抜粋です：

```
Function DoLogYInterpolate2(xInput, yInput, numOutputPoints, xOutput, yOutput)
        Wave xInput, yInput
        int numOutputPoints
        Wave& xOutput                   // 出力ウェーブの参照
        Wave& yOutput                   // 出力ウェーブの参照

        // 補間で使う log(yInput) を作成
        Duplicate/FREE yInput, logYInput        // 自由ウェーブは関数を抜けるときに自動的に Kill される
        logYInput = log(yInput)
```

```
        String xOutName = NameOfWave(xInput) + "_interp"
        String yOutName = NameOfWave(yInput) + "_interp"
        Interpolate2 /T=1 /I=0 /N=(numOutputPoints) /X=$xOutName /Y=$yOutName xInput, logYInput

        WAVE xOutput = $xOutName
        WAVE yOutput = $yOutName

        // yOutput は log(yInput) に基づいて設定される。元のスケールに変換して戻す
        yOutput = 10^yOutput
End
```

## Smoothing Spline（平滑化スプライン）アルゴリズム

Smoothing Spline アルゴリズムは、Christian H. Reinsch 著「Numerische Mathematik 10」の「Smoothing by Spline Functions」に基づいています。
これは次を最小化します。

$$\int_{x_0}^{x_n} g''(x)^2 dx$$

これは次のようなすべての関数 g(x) ついてです。

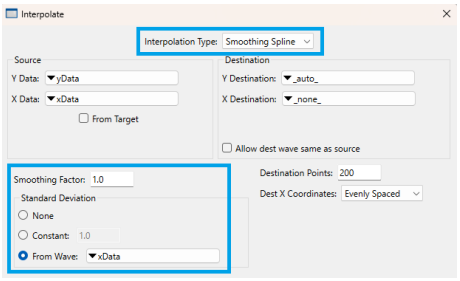
$$\sum_{i=0}^{n} \left( \frac{g(x_i) - y_i}{\sigma_i} \right)^2 \leq S$$

ここで、$g(x_i)$ は特定のポイントにおける平滑化スプラインの値、$y_i$ はそのポイントにおける Y データ、$\sigma_i$ はそのポイントの標準偏差、S は平滑化係数です。

## Smoothing Spline パラメーター

Smoothing Spline コマンドには、標準偏差パラメーターと平滑化係数パラメーターが必要です。
標準偏差パラメーターは、Y データのノイズの標準偏差を適切に推定できる値であるべきです。
平滑化係数は、標準偏差の推定値が正確であると仮定した場合、名目上 1.0 に近い値であるべきです。

Interpolate ダイアログの Standard Deviation セクションでは、標準偏差パラメーターに対して次の 3 つのオプションから選択できます：None、Constant、From Wave。




None を選択した場合、Interpolate2 は Y データの振幅の 0.05 倍という任意の標準偏差推定値を使います。
その後、平滑化スプラインが得られるまで平滑化係数パラメーターを調整できます。
平滑化係数 1.0 から開始してください。
この方法は推奨されません。

Constant を選択した場合、ノイズの標準偏差の推定値を入力できます。
Interpolate2 はこの値を Y データの各データポイントの標準偏差として使います。
推定値が適切であれば、平滑化係数を 1.0 前後に設定することで、データを通る滑らかな曲線を得られます。
最初の試行がうまくいかない場合は、平滑化係数を 1.0 のままにしておき、標準偏差の推定値を再設定してみてください。
ほとんどのデータタイプにおいて、これが推奨される方法です。

From Wave を選択した場合、Interpolate2 は指定されたウェーブ内の各ポイントが、Y データ内の対応するポイントに対する推定標準偏差を含むものと見なします。
適切なウェーブがある場合にこの方法を使ってください。

## Interpolate2 の事前平均化機能

Linear または Cubic Spline 補間は、入力データポイントすべてを通過します。
大規模でノイズの多いデータセットの場合、これは望ましい結果ではないでしょう。
代わりに Smoothing Spline を使ってください。

Interpolate2 に平滑化スプラインが実装される前は、入力データの減算版を補間するために Cubic Spline を使うことを推奨していました。
事前平均化機能はこの操作を容易にするために設計されました。

Interpolate2 が Smoothing Spline をサポートするようになったため、事前平均化機能は不要となりました。
ただし、後方互換性のため引き続きサポートしています。

事前平均化をオンにすると、Interpolate2 は入力データの一時的なコピーを作成し、それをデシメーションによってノードと呼ばれるより少ないポイント数に減らします。
Interpolate2 は通常、データの最初と最後にノードを追加します。
最後に、これらのノードを通して補間を行います。

## 同一またはほぼ同一の X 値

このセクションでは、ほとんどのユーザーには関係のない特異なケースについて議論します。

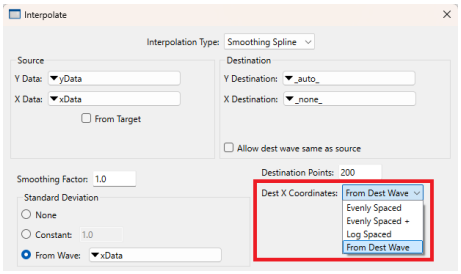
X 値が同一の 2 つのポイントを含む入力データは、補間アルゴリズムが予期しない結果を生成する原因となる場合があります。
これを回避するため、Interpolate2 は X 値がほぼ同一の 2 つ以上のデータポイントに遭遇した場合、補間を行う前にそれらを 1 つの値に平均化します。
この動作は事前平均化機能とは別に処理されます。
この処理は、Dest X Coordinates モードが Log Spaced または From Dest Wave の場合を除き、3 次スプラインおよび平滑化スプラインに対して行われます。線形補間では行われません。

X において、2 つのポイントがほぼ同一と見なされるのは、それらの間の X における差分（dx）が、名目上の dx の 0.01 倍未満である場合です。
名目上の dx は、入力データの X スパンを入力データポイント数で割って計算されます。

## 宛先ウェーブから宛先 X 座標を取得

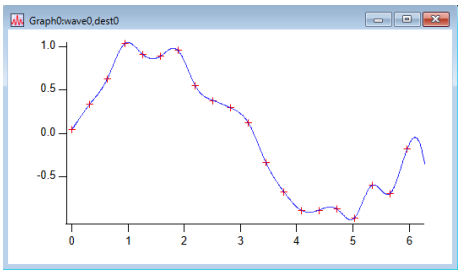
このモードは、Interpolate ダイアログの Dest X Coordinates ポップアップメニューから From Dest Wave を選択するか、Interpolate2 /I=3 フラグを使うと有効になります。
このモードでは、出力ポイントの数は宛先ウェーブによって決定され、/N フラグは無視されます。

X From Dest モードでは、補間が行われるポイントは宛先ウェーブフォームによって決定されます。
宛先がウェーブフォームの場合、その X 値で補間が行われます。
宛先が XY ペアの場合、X 宛先ウェーブフォームに保存されているデータ値で補間が行われます。




ウェーブフォームを宛先として使う例を以下に示します。

```
Make /O /N=20 wave0      // ソースデータを生成
SetScale x 0, 2*PI, wave0
wave0 = sin(x) + gnoise(.1)
Display wave0
ModifyGraph mode=3
Make /O /N=1000 dest0    // 宛先ウェーブフォームを生成
SetScale x 0, 2*PI, dest0
AppendToGraph dest0
ModifyGraph rgb(dest0)=(0,0,65535)
// 3次スプライン補間を実行
Interpolate2 /T=2 /I=3 /Y=dest0 wave0
```

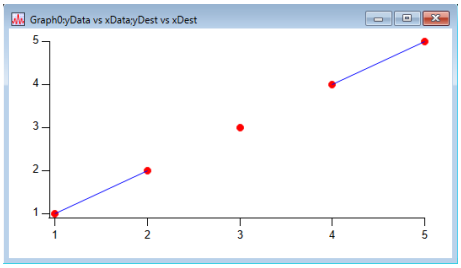



宛先が XY ペアの場合、Interpolate2 はこれらの値をソートしたバージョンを作成し、X ウェーブの値で出力を計算します。
X ウェーブが最初の値と最後の値を調べることで逆順ソートされていた場合、Interpolate2 は補間後に出力を逆転させ、元の順序を復元します。

XY ペアを宛先とする X From Dest モードでは、X ウェーブに NaN が含まれる場合があります。
この場合、Interpolate2 は XY ペアの内部コピーを作成し、X 宛先が NaN のポイントを削除して補間を行い、結果を宛先の XY ペアにコピーします。
この最後のステップで、Interpolate2 は NaN を X 宛先ウェーブの元の位置に復元します。

以下は、X 宛先ウェーブに NaN を含む XY 宛先の例です。

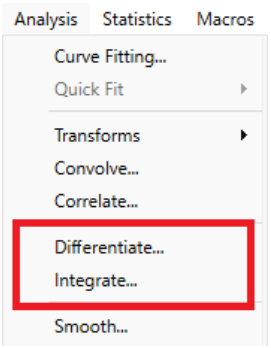
```
// ソースデータを生成
Make/O xData={1,2,3,4,5}, yData={1,2,3,4,5}
Display yData vs xData
// 宛先 XY ペアを生成
Make/O xDest={1,2,NaN,4,5}, yDest={0,0,0,0,0}
ModifyGraph mode=3,marker=19
AppendToGraph yDest vs xDest
ModifyGraph rgb(yDest)=(0,0,65535)
// 線形補間を実行
Interpolate2 /T=1 /I=3 /Y=yDest /X=xDest xData, yData
```




## 微分と積分

微分コマンドと積分コマンドは、１次元ウェーブフォームと XY データに対して操作を行うためのいくつかのアルゴリズムを提供します。
これらの操作は、元のデータを置き換えるか、結果から新しいウェーブを作成することができます。
これらの操作を使う最も簡単な方法は、Analysis メニューから利用可能なダイアログを使うことです。

ほとんどのアプリケーションでは、台形積分と中心差分微分が適切な手法です。
ただし、XY データで操作する場合、アルゴリズムによって X ウェーブにおけるポイント数の要件が異なります。
ダイアログに X ウェーブが表示されない場合は、別のアルゴリズムを選択するか、ヘルプボタンをクリックして要件を確認してください。




ウェーブフォームデータの操作時には X スケーリングが考慮されます。
/P フラグを使ってこれを無効にできます。
SetScale コマンドを使って、Y データウェーブフォームの X スケーリングを定義できます。

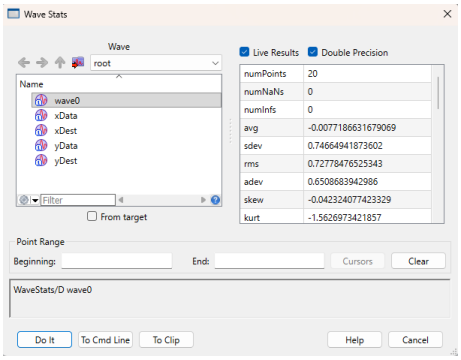
これらのコマンドは 1 次元のみを対象としますが、/DIM フラグを使うことで行列（あるいはそれ以上の次元）の行または列を対象に動作させることができます。

積分コマンドは、数値積分を用いてウェーブを置換または生成します。
曲線の下の面積を求めるには、「面積と平均」のセクションを参照してください。

## 面積と平均

Igor を使って、ウェーブの面積と平均値をいくつかの方法で計算できます。

平均値を計算する最も簡単な方法は、おそらく Statistics メニューの Wave Stats ダイアログを使うことです。
ウェーブを選択し、X 範囲を入力（または現在のカーソル位置を使用）し、Do It をクリックすると、Igor はいくつかの統計結果を履歴エリアに出力します。
その中には、平均値である V_avg が含まれます。
これは Igor の mean 関数が返す値と同じですが、mean 関数は他の統計量を計算しないため高速です。
mean 関数は、指定範囲内のデータに NaN が含まれると NaN を返します。
一方、WaveStats コマンドはこのような欠損値を無視します。

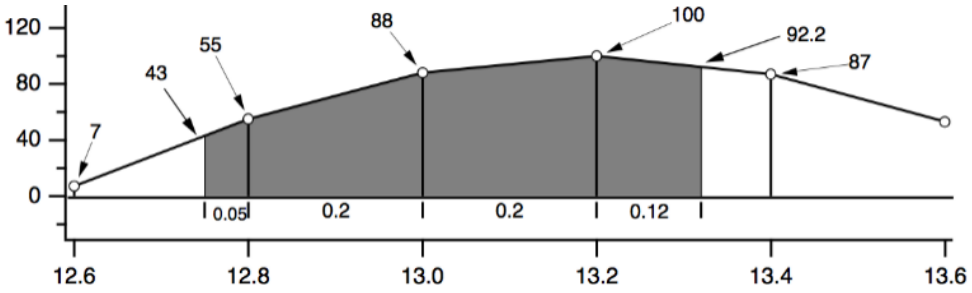



WaveStats と mean 関数は、平均を計算するために同じ方法を使います。
指定された X 範囲内のウェーブフォーム値を見つけ、それらを合計し、値の数で割ります。
X 範囲は、組み合わせる値を選択するだけの役割を果たします。
範囲は最も近いポイント番号に丸められます。

イベントの発生回数など、離散値を記述するデータを考える場合、平均値の計算には WaveStats または mean 関数を使うべきです。
イベントの総数（ある種の面積）は、sum 関数を使うと最も簡単に計算できます。
また、WaveStats の出力である V_avg と V_npnts を乗算することで簡単に計算することも可能です。

データが、サンプリングされたオーディオ信号などの連続的なプロセスのサンプル表現である場合は、平均を計算するには faverage 関数を使い、面積を計算するには area 関数を使ってください。
これら 2 つの関数は、台形積分と同じ線形補間スキームを使って、データポイント間のウェーブフォーム値を推定します。
X 範囲は、最も近いポイントに丸められません。
部分的な X 間隔は、この線形補間によって計算に含まれます。

下の図は、表示されたデータに対して各関数が実行する計算を示しています。
数値 43 と 92.2 は線形補間による推定値です。
この図は区間 (12.75,13.32) における面積関数、平均関数、平均値関数を比較しています。




```
WaveStats/R=(12.75,13.32) wave
                        V_avg = (55+88+100+87)/4 = 82.5

mean(wave,12.75,13.32)
                        = (55+88+100+87)/4 = 82.5

area(wave,12.75,13.32) =
                        0.05·(43+55)/2              // 最初の台形
                        + 0.20·(55+88)/2            // ２番目の台形
                        + 0.20·(88+100)/2           // ３番目の台形
                        + 0.12·(100+92.2)/2         // ４番目の台形
                              = 47.082

faverage(wave,12.75,13.32)  = area(wave,12.75,13.32) / (13.32-12.75)
                        = 47.082/0.57 = 82.6
```

ウェーブの X 方向スケーリングの影響を受けるのは面積関数のみであることに注意してください。
faverage は、面積をその面積が乗算されたのと同じ X 範囲で除算することで、X 方向スケーリングの影響を除去します。

これらの関数の問題点は、指定されたデータ範囲に欠損値（NaN）が含まれている場合に使用できないことです。
詳細は「欠損値の扱い」のセクションを参照してください。

## X の範囲と平均、平均値、面積関数

mean、faverage、area 関数の X 範囲入力はオプションです。
したがって、ウェーブ全体を含める場合、範囲を指定する必要はありません。

```
Make/N=10 wave=2; Edit wave.xy              // 0 から 9 までの X 範囲
Print area(wave)                            // X の全範囲
18
```

プログラミングにおいて、範囲がウェーブの端を超えているかどうかを判断するのが不便な場合があります。
幸いなことに、これらの関数はウェーブの端を超えた X 範囲も受け付けます。

```
Print area(wave, 0, 9)                      // X の全範囲
18
```

ウェーブの端を超えた範囲を評価する式を使用できます。

```
Print leftx(wave),rightx(wave)
0  10
Print area(wave,leftx(wave),rightx(wave))          // X の全範囲
18
```

または、±∞ の X 範囲でも可能です。

```
Print area(wave, -Inf, Inf)                    // 世界の全 x 範囲
18
```

## ウェーブの区間の平均値を求める

「分析プログラミング」のセクションでは、指定した長さのウェーブセグメントの平均値を求める機能を説明しています。
これは各セグメントの平均値を格納する新しいウェーブを作成します。

## XY データの面積

XY ペアのウェーブに含まれるデータ領域の面積を計算するには、areaXY 関数を使います。
faverage 関数の XY バージョンも存在します。
詳細は faverageXY コマンドのヘルプを参照してください。

さらに、AreaXYBetweenCursors WaveMetrics プロシージャファイルを使用できます。
このファイルには AreaXYBetweenCursors と AreaXYBetweenCursorsLessBase プロシージャが含まれています。
プロシージャファイルの読み込み方法については、ヘルプ WaveMetrics Procedures Folder を参照してください。
情報パネルとカーソルを使って、面積を計算する X 範囲を指定します。
AreaXYBetweenCursorsLessBase は、カーソル間の直線である単純な台形ベースラインを除去します。

# ウェーブの統計

WaveStats コマンドは、ウェーブに関連する様々な記述統計量を計算し、コマンドウィンドウの履歴エリアに出力します。
また、統計量を一連の特殊変数またはウェーブに保存するため、プロシージャからアクセスできます。

出力される統計量と対応する特殊変数は以下の通りです。

| | |
|---|---|
| V_npnts | NaN または INF の値を持つポイントを除く範囲内のポイントの数 |
| V_numNans | NaN の数 |
| V_numINFs | INF の数 |
| V_avg | データ値の平均 |
| V_sum | データ値の合計 |
| V_sdev | データ値の標準偏差 $\sigma = \sqrt{\frac{\sum (Y_i - V\_avg)^2}{V\_npnts - 1}}$<br>"Variance" は V_sdev2 |
| V_sem | 平均値の標準誤差 $sem = \frac{\sigma}{\sqrt{V\_npnts}}$ |

| | |
|---|---|
| V_rms | Y 値の RMS（二乗平均平方根） $=\sqrt{\frac{1}{V\_npnts}\sum Y_i^2}$ |
| V_adev | 平均偏差 $=\frac{1}{V\_npnts}\sum_{i=0}^{V\_npnts-1}\left\|Y_i-\bar{Y}\right\|$ |
| V_skew | 歪度 $=\frac{1}{V\_npnts}\sum_{i=0}^{V\_npnts-1}\left(\frac{Y_i-\bar{Y}}{\sigma}\right)^3$ |
| V_kurt | 尖度 $=\left(\frac{1}{V\_npnts}\sum_{i=0}^{V\_npnts-1}\left(\frac{Y_i-\bar{Y}}{\sigma}\right)^4\right)-3$ |
| V_minloc | X データ値の最小値の位置 |
| V_min | 最小データ値 |
| V_maxloc | X データ値の最大値の位置 |
| V_max | 最大データ値 |
| V_minRowLoc | 最小データ値を含む行 |
| V_maxRowLoc | 最大データ値を含む行 |
| V_minColLoc | 最小データ値を含む列（2D 以上のウェーブ） |
| V_maxColLoc | 最大データ値を含む列（2D 以上のウェーブ） |
| V_minLayerLoc | 最小データ値を含むレイヤー（3D 以上のウェーブ） |
| V_maxLayerLoc | 最大データ値を含むレイヤー（3D 以上のウェーブ） |
| V_minChunkLoc | 最小データ値を含むチャンク（4D ウェーブのみ） |
| V_maxChunkLoc | 最大データ値を含むチャンク（4D ウェーブのみ） |
| V_startRow | 統計計算に含まれる最初の行の非スケーリングインデックス |
| V_endRow | 統計計算に含まれる最後の行の非スケーリングインデックス |
| V_startCol | 統計計算に含まれる最初の列の非スケーリングインデックス。/RMD が使われている場合にのみ設定される |
| V_endCol | 統計計算に含まれる最後の列の非スケーリングインデックス。/RMD が使われている場合にのみ設定される |
| V_startLayer | 統計計算に含まれる最初のレイヤーの非スケーリングインデックス。/RMD が使われている場合にのみ設定される |
| V_endLayer | 統計計算に含まれる最後のレイヤーの非スケーリングインデックス。/RMD が使われている場合にのみ設定される |
| V_startChunk | 統計計算に含まれる最初のチャンクの非スケーリングインデックス。/RMD が使われている場合にのみ設定される |

| | |
|---|---|
| V_endChunk | 統計計算に含まれる最後のチャンクの非スケーリングインデックス。/RMD が使われている場合にのみ設定される |

WaveStats コマンドを使うには、Statistics メニューから Wave Stats を選択します。

Igor は平均、標準偏差、RMS（二乗平均平方根）、最小値、最大値の計算において NaN と INF を無視します。
NaN は数学的に定義された意味を持たない計算から生じます。
また、欠損値を表すためにも使われます。
INF は有限値を持たない数学的演算から生じます。

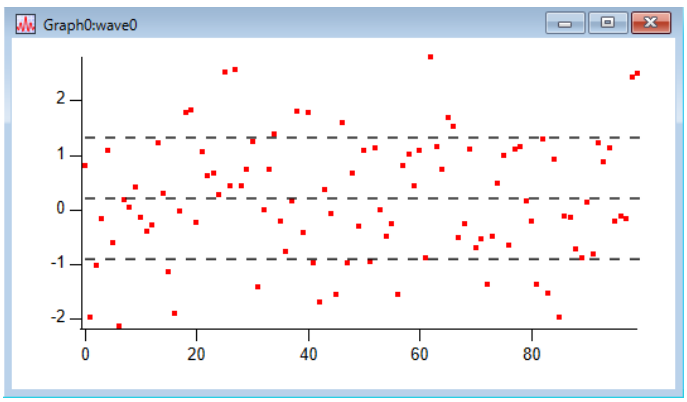
次のプロシージャは WaveStats の使い方を示しています。
ソースウェーブの平均値と標準偏差を表示します（ソースウェーブは最前面のグラフに表示されているものと仮定します）。
平均値と標準偏差を示す線を描画します。

```
Function ShowAvgStdDev(source)
        Wave source                     // ソースウェーブフォーム

        Variable left=leftx(source),right=rightx(source)       // ソースの x 範囲
        WaveStats/Q source
        SetDrawLayer/K ProgFront
        SetDrawEnv xcoord=bottom,ycoord=left,dash= 7
        DrawLine left, V_avg, right, V_avg                     // 平均を表示
        SetDrawEnv xcoord=bottom,ycoord=left,dash= 7
        DrawLine left, V_avg+V_sdev, right, V_avg+V_sdev       // +標準偏差を表示
        SetDrawEnv xcoord=bottom,ycoord=left,dash= 7
        DrawLine left, V_avg-V_sdev, right, V_avg-V_sdev       // -標準偏差を表示
        SetDrawLayer UserFront
End
```

以下のコマンドを使って、この関数を試すことができます。

```
Make/N=100 wave0 = gnoise(1)
Display wave0; ModifyGraph mode(wave0)=2, lsize(wave0)=3
ShowAvgStdDev(wave0)
```




複素数ウェーブに対して WaveStats を使う場合、ウェーブの実部、虚部、振幅、位相について、上記と同じ統計量を計算するように選択できます。
デフォルトでは、WaveStats はウェーブの実部に対する統計量のみを計算します。
他の成分に対する統計量を計算する場合、このコマンドは結果を多次元ウェーブ M_WaveStats に保存します。

大量のデータを処理しており、計算速度が懸念される場合、計算を一次モーメントに制限する /M フラグを活用できる可能性があります。

2D または 3D ウェーブを扱っており、任意の形状の領域に対する統計量を計算したい場合は、ROI ウェーブを用いた ImageStats コマンドを使うべきです。