# CONTENTS

**ビジュアルヘルプ – CurveFit** ..... 2

CurveFit コマンドのヘルプ ..... 2

# ビジュアルヘルプ – CurveFit

## CurveFit コマンドのヘルプ

**CurveFit** [ *flags* ] *fitType*, [*kwCWave=coefWaveName* ,] *waveName* [ *flag parameters* ]

CurveFit コマンドは、データにいくつかの組み込み関数のいずれかをフィッティングします（ユーザー定義のフィッティングについては、FuncFit コマンドを参照してください）。
CurveFit と組み込みのフィッティング関数を使ってフィッティングを行う場合、自動初期推定値が、ほとんどの場合、適切な出発点となります。

フィッティングの結果はウェーブとして返されます。
デフォルトでは W_coef です。
また、結果はシステム変数 K0、K1 … Kn にも格納されますが、システム変数の使用は制限されており、非推奨とされています。

ヘルプ Fitting with Complex-Valued Functions で説明しているように、ユーザー定義のフィッティング関数は複素数値を返すことができます。
複素数値のフィッティング関数を記述する時、複素係数ウェーブを必要とするように設定することも可能ですが、その場合はシステム変数はサポートされません。

kwCWave キーワードを使えば、W_coef の代わりに、係数ウェーブとして独自のウェーブを指定することができます。

CurveFit コマンドで指定されるウェーブは、ほぼすべて、Display コマンドがグラフ作成に使うのと同じサブレンジ（一部）構文を使って、より大きなウェーブのサブレンジとすることができます。
詳細は、以下の「ウェーブのサブレンジの詳細」のセクションを参照してください。

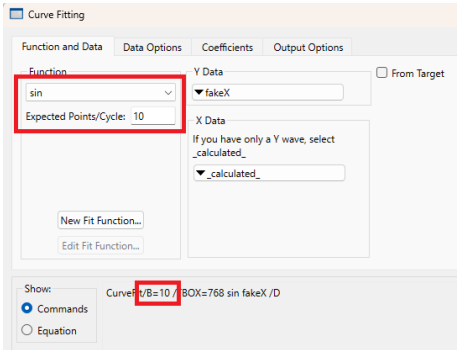
「カーブフィッティング」の詳細（Curve Fit ダイアログの使い方など）については、ヘルプ Curve Fitting を参照してください。
Curve Fit ダイアログで、すべての設定ができるわけではないことは覚えておいてください。
以降では、フラグ、パラメーターとダイアログの対応をいくつか画面で示しています。

CurveFit コマンドのパラメーターは、フラグ、フィッティングの種類、パラメーター（kwCWave=coefWaveName と waveName）、フラグパラメーターというカテゴリに分類されます。
以下の各セクションは、これらのカテゴリに対応しています。
なお、フラグはフィッティングの種類の前に記述し、フラグパラメーターは waveName の後に記述する必要があります。

**フラグ**

/B=*pointsPerCycle*

type が sin の場合に使います。*pointsPerCycle* は、正弦波の 1 周期あたりのデータポイント数の推定値です。これにより、フィッティングの初期推定値が得られます。推定した *pointsPerCycle* の値の前後で、いくつかの異なる値を試す必要がある場合があります。




/C

制約行列とベクトルを作成します。これは、/C=constraintSpec パラメーターを使って制約を指定する場合にのみ適用されます。

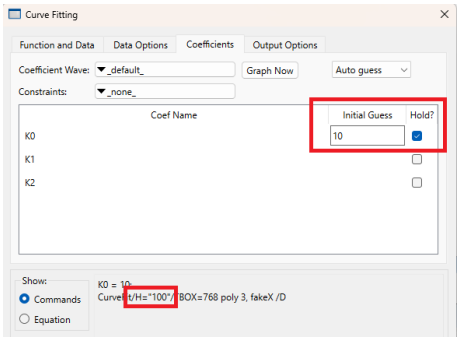
M_FitConstraint という行列と W_FitConstraint というベクトルが作成されます。詳細については、ヘルプ Fitting with Constraints を参照してください。

/G

変数 K0、K1…Kn の値を、フィッティングの初期推定値として使います。kwCWave キーワードで係数ウェーブを指定した場合、初期推定値は係数ウェーブから読み込まれます。

/H="*hhh...*"

一定に保つ係数を指定します。*h* が 1 の場合は係数が一定、0 の場合は係数が変動することを示します。
例：/H=「100」 と指定すると、K0 は一定に保たれ、K1 と K2 が変化します。




/K={*constants*}

特定のフィッティング関数において、定数（フィッティング係数ではない）の値を設定します。例えば、exp_XOffset 関数には X オフセット定数が含まれています。組み込み関数ではこの定数が自動的に設定されますが、このフラグを使うことで、その自動設定値を上書きすることができます。

*constants* は定数値のリストです（例：/K={1,2,3}）。このリストの長さは、選択したフィッティング関数で使われる定数の数と一致している必要があります。

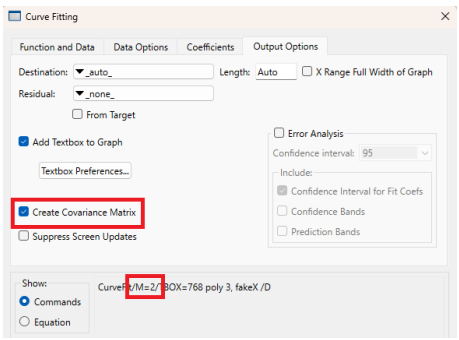
このフラグは現在、Curve Fit ダイアログではサポートされていません。To Cmd ボタンを使って、コマンドラインでこのフラグを追加してください。

/L=*destLen*

AutoTrace 機能によって生成されるウェーブの長さを設定します。つまり、宛先ウェーブを指定しない /D パラメーターに相当します（以下の「フラグパラメーター」セクションにある /D パラメーターを参照してください）。fit_waveName のウェーブの長さは、*destLen* に設定されます。このキーワードは、信頼区間および予測区間のために生成されるウェーブの長さも設定します。

/M=*doMat*

共分散行列を生成します。
*doMat*=2 の場合、共分散行列は M_Covar という名前の 2D ウェーブとして格納されます。
*doMat*=1 または指定がない場合、共分散行列は 1D ウェーブ CM_Kn として生成されます。ここで、n は 0（K0 の場合）から係数の数から 1 を引いた値までとなります。

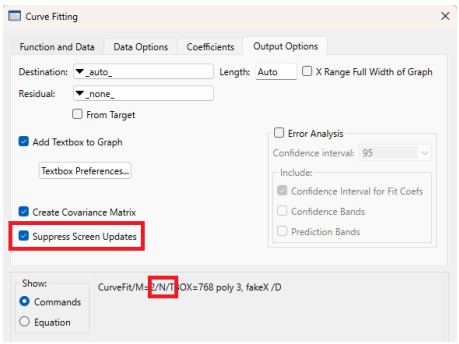



*doMat*=0 の場合、共分散行列は生成されません。*doMat*=1 は、Igor Pro の旧バージョンとの互換性のために含まれていますが、*doMat*=2 を使うことを推奨します。

CurveFit コマンドは、直線近似の共分散行列を生成しません。その代わりに、出力変数 V_rab を通じて相関係数を出力します。

/N[=*updateMode*]　　カーブフィッティング処理中のウィンドウの更新をコントロールします。反復計算のたびにウィンドウを更新すると、カーブフィッティングの処理速度が大幅に低下する可能性があります。

*updateMode*=0：各反復ごとに更新が行われるため、フィッティングの進捗状況を監視することができます。フィッティングが完了すると、更新が実行されます。




*updateMode*=1：各フィッティング反復後の画面更新を抑制します。/X フラグを使うと、X 範囲が正確であることを確認するために画面更新が行われます。/N は /N=1 と同じです。これがデフォルトの更新モードです。

*updateMode*=2：/X フラグを使う場合を含め、すべての更新を無効にします。ユーザー定義関数内で /X フラグを使う場合は、DoUpdate を呼び出してグラフが更新されていることを確認する必要があります。このモードを使うには、Igor Pro 9.0 以降が必要です。

詳細については、以下の「カーブフィッティング画面の更新」のセクションを参照してください。

/NTHR = *nthreads*　　このフラグは有効ですが、廃止予定であり、何の効果もありません。詳細については、ヘルプ Curve Fitting with Multiple Processors を参照してください。

/O　　初期推定値を生成するのみであり、実際のフィッティングは行いません。

/ODR=2 でない限り、このフラグは線形フィッティングタイプ（line、poly、poly_XOffset、poly2D）で使われた場合、無視されます。FuncFit コマンドでも、このフラグは無視されます。

/ODR=*fitMethod*　　フィッティングの方法を選択します。選択肢は以下の通りです：

*fitMethod*=0：　デフォルト。古いコードを使った Levenberg-Marquardt 法（最小二乗法）です。

*fitMethod*=1：　ODRPACK95 コードを使って実装された、信頼領域 Levenberg-Marquardt 法（通常の最小二乗法）です。「カーブフィッティングの参考文献」を参照してください。

*fitMethod*=2：　ODRPACK95 コードを用いて実装された、信頼領域 Levenberg-Marquardt 最小直交距離法です。この手法は、独立変数に測定誤差が存在する場合の推定に適していて、「変数誤差推定」、「ランダム回帰変数モデル」、または「測定誤差モデル」とも呼ばれます。

*fitMethod*=3：　暗黙的なフィッティングです。従属変数は指定されず、代わりに、すべての従属変数に対してフィッティング係数がゼロを返すように、フィッティング係数を調整しようと試みます。

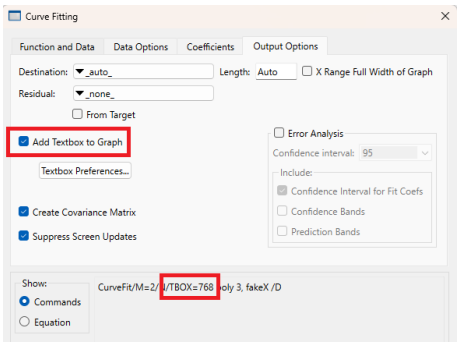
組み込みのフィッティング関数では、暗黙的関数フィッティングはほとんど役に立ちません。その代わりに、FuncFit 関数と、暗黙的関数フィッティング用に設計されたユーザー定義のフィッティング関数を使ってください。

Q[=*quiet*]

*quiet*=1 に設定すると、結果が履歴に出力されなくなります。/Q は /Q=1 と同じです。

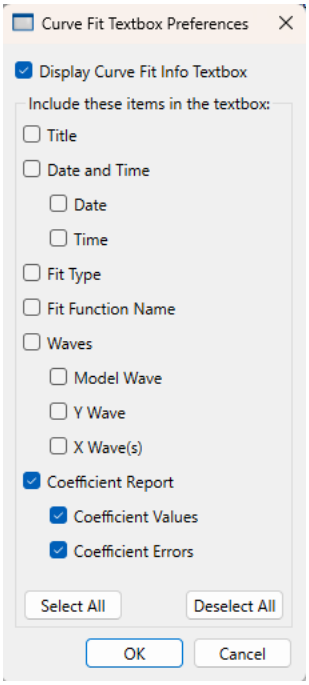
/TBOX=*textboxSpec*

最前面のグラフに推定データが表示されている場合、/TBOX フラグを指定すると、そのグラフに注釈が追加されます。すでに注釈が存在する場合は、それが更新されます。最前面のグラフ以外のグラフには影響しません。




このテキストボックスには、フィッティングに関するカスタマイズ可能な情報セットが含まれます。textboxSpec パラメーターは、テキストボックスに含めるさまざまな要素を選択するためのビットフィールドです：

| **textboxSpec** | **この要素を選択** |
|---|---|
| 1 | タイトル "Curve Fit Results" |
| 2 | 日付 |
| 4 | 時刻 |
| 8 | フィッティング手法（最小二乗法、ODR など） |
| 16 | フィッティング関数名 |
| 32 | モデルウェーブ、自動宛先ウェーブ（必要に応じてトレースの記号を含む） |
| 64 | Y ウェーブとトレースシンボル |
| 128 | X ウェーブ |
| 256 | 係数値レポート |
| 512 | 係数値レポートにエラーを含める |




必要な各部品の合計を算出し、それらをすべて含めるよう指定してください。

*textboxSpec* のデフォルト設定は、すべてのビットがオンになっています。

*textboxSpec* を 0 に設定すると、テキストボックスが削除されます。

/W=*wait*

カーブフィッティングの結果ウィンドウの動作を指定します。

*wait*=0：カーブフィッティングの結果ウィンドウを表示しますが、ユーザーが「OK」ボタンをクリックするのを待ちません。

*wait*=1：カーブフィッティングの結果ウィンドウを閉じる前に、ユーザーが「OK」ボタンをクリックするのを待ちます。これは、コマンドラインまたはダイアログからの実行時におけるデフォルトの動作です。

*wait*=2：カーブフィッティングの結果ウィンドウを一切表示しません。これは、プロシージャから実行されるカーブフィッティング処理のデフォルト設定です。

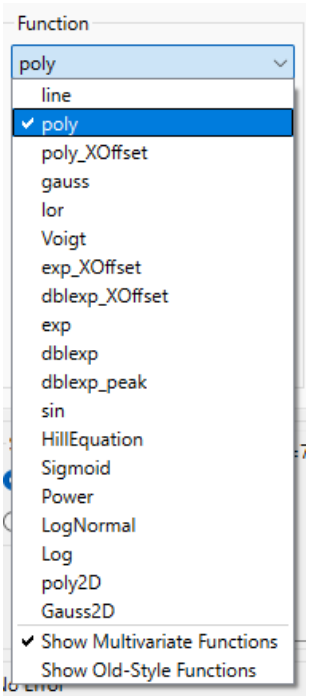
/X [=*fullRange*]

/X または /X=1 を指定すると、Y ウェーブが上部のグラフに表示されている場合、自動トレース先のウェーブの X 軸のスケールが、グラフ上の適切な X 軸に合わせて設定されます。これは、カーブフィッティングの対象となる X データ範囲の外側まで曲線を外挿したい場合に便利です。

/X を省略するか、/X=0 を指定すると、自動トレース先のウェーブの X 軸のスケーリングは、フィッティング対象の X 値の範囲に合わせて設定されます。

**フィッティングタイプ**

fitType は、組み込みのカーブフィッティング関数のタイプを指定します。




gauss

Gaussian（ガウス）ピーク：
y = K0+K1*exp(-((x-K2)/K3)^2)

lor

Lorentzian（ローレンツ）ピーク：
y = K0+K1/((x-K2)^2+K3)

Voigt

Lorentzian ピークと Gaussian ピークの畳み込みである Voigt（ヴォイクト）ピークにフィッティングします。Lorentzian ピーク成分と Gaussian ピーク成分の幅の比率を変えることで、形状を Lorentzian 形状から Gaussian 形状へと段階的に変化させることができます：
y = VoigtPeak(W_coef, x)

exp

指数関数：y = K0+K1*exp(-K2*x)

dblexp

二重指数関数：
y = K0+K1*exp(-K2*x)+K3*exp(-K4*x)

exp_XOffset

指数関数：y = K0+K1*exp(-(x-x0)/K2)

x0 は定数であり、デフォルトでは、近似に含まれる最小の X 値に設定されます。x0 を指定することで、exp 関数で発生しうる浮動小数点丸め誤差による問題を回避できます。

dblexp_XOffset

二重指数関数：y = K0+K1*exp(-(x-x0)/K2)+K3*exp(-(x-x0)/K4)

x0 は定数であり、デフォルトでは、近似に含まれる最小の X 値に設定されます。x0 を指定することで、exp 関数で発生しうる浮動小数点丸め誤差による問題を回避できます。

dblexp_peak

符号が逆の振幅を持つ二重指数関数が重なり、ピークを形成：
y = K0+K1*(exp(-(x-K4)/K2)+exp(-(x-K4)/K3)).

sin

正弦波：y = K0+K1*sin(K2*x+K3)

line

線形：y = K0+K1*x

poly *n*

多項式：y = K0+K1*x+K2*x^2+...

*n* は 0 より大きくなければなりません。*n* は項の数、あるいは次数に 1 を加えた値です。Igor Pro 9.0 以前では、*n* の最小許容値は 3 でした。

*n* に上限はありませんが、適度な値に抑えることを推奨します。浮動小数点数の切り捨て誤差により、事実上、明確に定義されていない上限が生じます。高次多項式を近似する場合は、X の範囲が [-1, 1] に近いことを確認するか、poly_XOffset フィッティングタイプを使ってください。

*n*=1 は Y 値の平均を求めることと同じで、*n*=2 はデータセットに直線をフィッティングさせますが、線形フィッティングタイプを使った場合に得られるような特別な統計情報は出力されません。

poly_XOffset *n*

多項式：y = K0+K1*(x-x0)+K2*(x-x0)^2+...

*n* は 0 より大きくなければなりません。*n* は項の数、あるいは次数に 1 を加えた値です。Igor Pro 9.0 以前では、*n* の最小許容値は 3 でした。

*n*=1 は Y 値の平均を求めることと同じで、*n*=2 はデータセットに直線をフィッティングさせますが、線形フィッティングタイプを使った場合に得られるような特別な統計情報は出力されません。

*n* に上限はありませんが、適度な値に抑えることを推奨します。poly_XOffset フィッティングタイプは poly フィッティングタイプよりも優れていますが、浮動小数点数の切り捨て誤差により、実用上は明確な上限が設けられています。

x0 は定数で、デフォルトでは、近似に含まれる最小の X 値に設定されます。x0 を指定することで、近似対象の X 値が原点から遠く離れている場合に生じる浮動小数点丸め誤差による問題を回避できます。ただし、それでも高次多項式では問題が生じる可能性があることに注意してください。

hillequation

Hill（ヒル）の方程式：K0+(K1-K0)*(x^K2/(1+(x^K2+K3^K2)))

これは S 字関数です。X の値は 0 より大きくなければなりません。

sigmoid

シグモイド：y = K0+K1/(1+exp(-(x-K2)/K3))

power

べき乗則：K0+K1*x^K2

X の値は 0 より大きくなければなりません。

lognormal

対数正規分布：0+K1*exp(-(ln(x/K2)/K3)^2)

X の値は 0 より大きくなければなりません。

log

対数（底 10）：K0+K1*log(x)

X の値は 0 より大きくなければなりません。

Log は Igor Pro 9.0 で追加されました。

gauss2D

２次元 Gaussian（ガウス）：
K0+K1*exp((-1/(2*(1-K6^2)))*(((x-K2)/K3)^2 + ((y-K4)/K5)^2 - (2*¥K6*(x-K2)*(y-K4)/(K3*K5))))

相互相関係数（K6）は、-1 から 1 の範囲内である必要があります。この係数は、自動的にその範囲内に収まるよう制約されます。相関がゼロであると確信している場合は、これをゼロに固定することで、推定処理を大幅に高速化できる可能性があります。

poly2D *n*　　２次元多項式：K0+K1*x+K2*y+K3*x^2+K4*xy+K5*y^2+...

*n* は多項式の次数です。次数 *n* までのすべての項が含まれ、交差項も含まれます。例えば、次数 3 の項には $x^3$、$x^2y$、$xy^2$、$y^3$ があります。

組み込みのフィッティング関数の詳細については、ヘルプ Built-in Curve Fitting Functions を参照してください。

**パラメーター**

kwCWave=*coefWaveName* は、オプションの係数ウェーブを指定します。
指定された場合、指定された係数ウェーブには、カーブフィッティングによって決定された最終的な係数が設定されます。
指定がない場合、W_coef という名前のウェーブが作成され、カーブフィッティングによって決定された最終的な係数が設定されます。

kwCWave=*coefWaveName* を使い、かつ /G フラグを指定した場合、初期推定値は指定された係数ウェーブから取得されます。

*waveName* は、選択した関数タイプにフィッティングさせる従属変数のデータを含むウェーブです。
独立変数が１つだけの関数の場合、従属変数のデータはしばしば「Y データ」と呼ばれます。
ウェーブ名の後に (startX,endX) を指定することで、ウェーブの一部サブレンジにフィッティングすることができます。
構文の説明には記載されていませんが、ウェーブ名の後に [startP,endP] を指定することで、ポイント単位でサブレンジを指定することも可能です。
カーブフィッティングにおけるウェーブの一部サブレンジの詳細については、以下の「ウェーブの一部サブレンジの詳細」のセクションを参照してください。

２次元フィッティング関数（gauss2D または poly2D）のいずれかを使う場合、*waveName* には行列ウェーブを指定するか、/X フラグを使って X ウェーブのリストを指定する必要があります。

**フラグパラメーター**

これらのフラグパラメーターは、*waveName* の後に指定する必要があります。

/A=*appendResid*　　*appendResid*=1 に設定すると、自動生成された残差がグラフに追加され、*appendResid*=0 に設定すると追加されません。*appendResid*=0 の場合、ウェーブは生成され残差値で塗りつぶされますが、グラフには追加されません。デフォルトは /A=1 です。

/AD[=*doAutoDest*]　　*doAutoDest* が 1 の場合、/D のみを指定した場合と同じになります。/AD は /AD=1 と同じです。

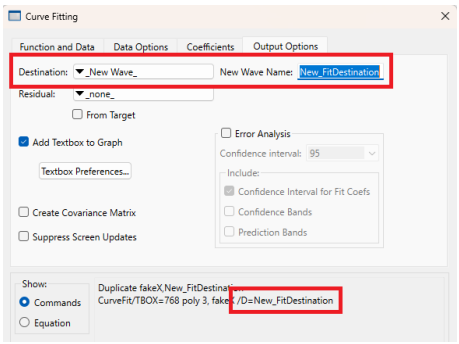
/AR=*doAutoResid*　　*doAutoResid* が 1 の場合、/R のみを指定した場合と同じになります。/AR は /AR=1 と同じです。

/C=*constraintSpec*　　カーブフィッティングの時に線形制約を適用します。制約は、制約式を含むテキストウェーブ（/C=*textWaveName*）の形式、または適切な行列とベクトル（/C={*constraintMatrix*, *constraintVector*}）の形式で指定できます。
詳細は、ヘルプ Fitting with Constraints を参照してください。

**注記：**
組み込みの line、poly、poly2D フィッティング関数では、制約条件を設定することはできません。これらのフィッティング関数に制約条件を適用するには、ユーザー定義のフィッティング関数を作成する必要があります。

/D[=*destwaveName*]　　*destwaveName* は、フィッティングの結果得られた式に基づいて評価されます。*destwaveName* は *waveName* と同じ長さでなければなりません。

/D のみを指定した場合、自動的に名前が付けられたウェーブが作成されます。名前は *waveName* を基に、「fit_」を接頭辞として付けられます。この自動的に名前が付けられたウェーブは、*waveName* が最前面のグラフにプロットされている場合、必要に応じてそのグラフに追加されます。




自動的に名前が付けられたウェーブの X 軸のスケールは、フィッティングに使われた x データの範囲に基づいて設定されます。

デフォルトでは、自動的に作成されるウェーブの長さは 200 ポイント（直線フィッティングの場合は 2 ポイント）です。これは /L フラグを使って変更できます。

*waveName* が対数スケールの X 軸上に表示される 1D ウェーブである場合、Igor Pro は値が指数関数的に間隔を空けて配置された X ウェーブも作成します。その名前は *waveName* を基に、「fitX_」を接頭辞として付けられます。

複素数を返すユーザー定義関数にフィッティングする場合、結果のウェーブも複素数になります。

/F={*confLevel*, *confType* [, *confStyleKey* [, *waveName* ...]]}

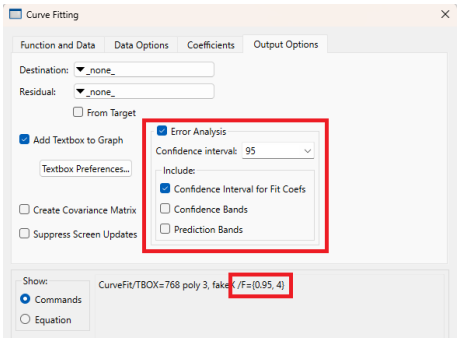
信頼水準 *confLevel* に対する信頼区間を計算します。*confLevel* の値は 0 から 1 の間でなければならず、これは 0% から 100% の信頼水準に対応します。

*confType* は、何を計算するかを選択します：

1： モデルの信頼区間

2： モデルの予測区間

4： フィッティング係数の信頼区間




これらの値を合計することで、複数のオプションを選択できます。つまり、信頼区間とフィッティング係数の信頼区間の両方を選択するには、*confType* を 5 に設定します。

信頼区間と予測区間は、指定した信頼水準に沿ったウェーブとして表示することも（*confStyleKey* に「Contour」を指定）、エラーバーとして表示することもできます（*confStyleKey* に「ErrorBar」を指定）。デフォルトは Contour です。

ウェーブが指定されていない場合、結果を含むウェーブが自動的に生成され、最前面のグラフに追加されます（最前面のグラフに推定データが含まれている場合）。信頼区間のウェーブに関する詳細は、以下をご覧ください。

**注記：**
多変量カーブフィッティングでは、信頼区間と予測区間を使用することはできません。

/I[=*weightType*]

*weightType* が 1 の場合、重み付けウェーブ（後述の /W パラメーターを参照）には標準偏差が含まれます。*weightType* が 0 の場合、重み付けウェーブには標準偏差の逆数が含まれます。/I パラメーターが指定されていない場合、デフォルトは /I=0 となります。

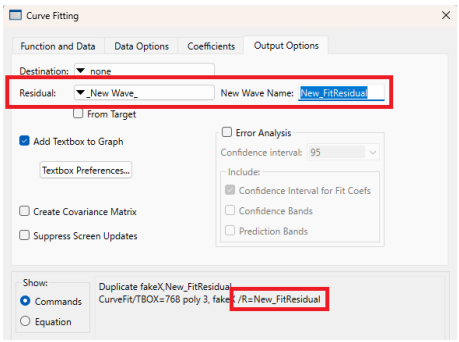
/M=*maskWaveName*

*maskWaveName* という名前のウェーブを使って、フィッティングの対象となるポイントを選択することを指定します。マスクウェーブは、ポイント数と次元において、従属変数のウェーブと一致している必要があります。マスクウェーブのポイントをゼロまたは NaN（表では空白）に設定すると、そのポイントはフィッティングの対象から除外されます。複素数ウェーブをフィッティングする場合でも、マスクウェーブは実数でなければなりません。

/R[=*residwaveName*]

*residwaveName* の各要素は、データ値からモデル値を差し引くことで算出されます。*residwaveName* の長さは *waveName* と同じでなければなりません。

/R のみが指定された場合、*waveName* と同じポイント数を持つ、自動的に名前が付けられたウェーブが作成されます。名前は *waveName* を基に、「Res_」を接頭辞として付けられます。新しいウェーブを作成する必要がある場合は、自動的に NaN で埋められますが、ウェーブがすでに存在する場合は、そのまま再利用されます。




フィッティング中は、実際にフィッティングの対象となるポイントについてのみ残差が計算されます。サブレンジ（一部）に対してフィッティングを行う場合や、マスクウェーブを使って特定のポイントに対してフィッティングを行う場合は、それらのポイントのみが保存されます。詳細については、ヘルプ Residuals Using Auto Trace を参照してください。

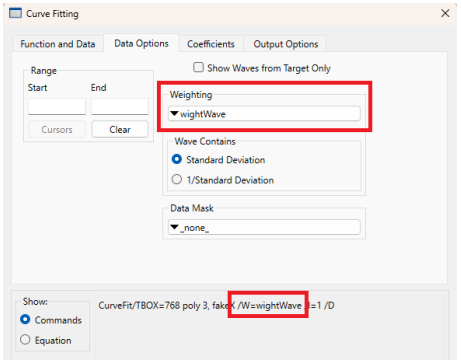
*waveName* が前面のグラフにプロットされている場合、自動的に作成された残差ウェーブは（必要に応じて）そのグラフに追加されます。残差ウェーブは、Y データのプロットに使われている縦軸の名前の先頭に「Res_」を付加して命名された、新しい自由軸に追加されます。可能な限り、新しい自由軸は見やすくフォーマットされます。

フィッティング対象のデータを含むグラフの書式設定が非常に複雑な場合、残差をグラフに自動的に追加したくないことがあるかもしれません。その場合は、前述の通り /A=0 を使ってください。

複素数を返すユーザー定義関数にフィッティングを行う場合、残差のウェーブも複素数になります。

/W=*wghtwaveName*

*wghtwaveName* には、フィッティング中に適用される重み値が含まれており、*waveName* と同じ長さでなければなりません。これらの重み値は、標準誤差の逆数、または標準誤差そのもののいずれかです。詳細については、上記の /I パラメーターを参照してください。




複素数を返すユーザー定義関数にフィッティングさせる場合、重み付けウェーブも複素数でなければなりません。

/X=*xwaveName*

フィッティングさせるデータの X 値は *xwaveName* から取得されます。*xwaveName* は *waveName* と同じ長さでなければなりません。

２次元フィッティング関数のいずれかにフィットを行う場合で、*waveName* が列ウェーブであるとき、*xwaveName* は X 次元の独立変数データを提供します。この場合、*xwaveName* は、*waveName* と同じ行数を持つ１次元ウェーブを指定する必要があります。

複素数値のフィッティング関数にフィッティングさせる場合、X ウェーブは、X 入力またはフィッティング関数で宣言されたパラメーターと整合する実数または複素数でなければなりません。

/X={*xwave1*, *xwave2*}

*waveName* が 1D ウェーブの場合、２次元フィッティング関数のいずれかにフィッティングさせます。*xwave1* と *xwave2* は、*waveName* と同じ長さでなければなりません。

/Y=*ywaveName*

*waveName* が行列ウェーブである場合、2D フィッティング関数のいずれかを使ってフィッティングを行います。*ywaveName* は、*waveName* の列数と同じ長さの 1D ウェーブでなければなりません。

/NWOK

ユーザー定義関数でのみ使用可能です。このオプションが指定されている場合、特定のウェーブを null ウェーブ参照に設定することができます。通常、CurveFit に null ウェーブ参照を渡すとエラーとして扱われます。/NWOK を使うことで、null ウェーブ参照はエラーではなく、対応するフラグを無視すべきであることを CurveFit コマンドに指示することになります。これにより、オプションのウェーブを指定して CurveFit を呼び出す関数コードを記述しやすくなります。

影響を受けるウェーブは、X ウェーブ（/X）、ウェイトウェーブ（/W）、マスクウェーブ（/M）、制約テキストウェーブ（/C）です。宛先ウェーブ（/D=wave）と残差ウェーブ（/R=wave）も影響を受けますが、/D と /R が「自動宛先指定を行う」と「自動残差指定を行う」という二重の意味を持つため、状況はより複雑になります。/AR および /AD を参照してください。

このオプションが必要ない場合は、このフラグを含めない方がよいでしょう。というのも、ミスや実行時の状況によって予期せずウェーブが欠落した場合、有用なエラーメッセージが表示されなくなるためです。

**注記：**
正しく動作させるには、このフラグをコマンドの最後に指定する必要があります。

**非ゼロ /ODR のフラグパラメーター**

/XW=*xWeightWave*
/XW={*xWeight1*, *xWeight2*}

/ODR=2 または 3 のみ。

*xWeightWave* には独立変数の重み値が含まれており、*waveName* と同じ長さでなければなりません。poly2D や Gauss2D などの多変量フィッティング関数でフィッティングを行う場合、2番目の形式を使って、各独立変数に対して重みウェーブを指定する必要があります。

重み付けの値は、標準誤差の逆数、または標準誤差そのもののいずれかです。標準誤差と標準誤差の逆数のどちらを選択する場合でも、/W と /XW の両方で同じ値を使う必要があります。詳細については、上記の /I フラグを参照してください。

複素数値のフィッティング関数にフィッティングさせる場合、X 重み付けウェーブは、フィッティング関数で宣言された X パラメーターと整合するように、実数または複素数でなければなりません。

/XHLD=*holdWave*
/XHLD={*holdWave1*, *holdWave2*}

/ODR=2 または 3 のみ。

直交距離回帰において、独立変数の値を固定するためのウェーブです。ウェーブは入力 X データと一致している必要があります。ウェーブの要素に「1」が入力されると、対応する X の値が固定されます。

/CMAG=*scaleWave*　解におけるフィッティング係数の予想されるスケールを示すウェーブを指定します。係数ごとに予想値の桁数が大きく異なる場合、これによりフィッティングの効率と精度が向上することがあります。

/XD=*xDestWave*
/XD={*xDestWave1*, *xDestWave2*}

/ODR=2 または 3 のみ。

最小直交距離回帰において、独立変数の推定値を受け取るウェーブを指定します。

複素数値のフィッティング関数にフィッティングさせる場合、X の宛先ウェーブは、フィッティング関数で宣言された X パラメーターと整合するように、実数または複素数でなければなりません。

/XR=*xResidWave*
/XR={*xResidWave1*, *xResidWave2*}

/ODR=2 または 3 のみ。

最小直交距離回帰において、独立変数の推定値と初期値との差分を受け取るウェーブを指定します。つまり、これらのウェーブには X の残差が格納されます。

複素数値のフィッティング関数にフィッティングさせる場合、X 残差ウェーブは、フィッティング関数で宣言された X パラメーターと整合するように、実数または複素数でなければなりません。

/PNLT=*penalty*　/ODR=3 のみ。

暗黙的 ODR フィッティングのペナルティパラメーターの初期値を設定します。ペナルティが 0 以下の場合、ODRPACK のデフォルト値が使われます。

ペナルティパラメーターは、残差の二乗和の最小化と、暗黙的方程式の充足とのバランスをコントロールします。通常、これを設定する必要はありません。ほとんどの場合、デフォルト設定で十分です。

**注意：**
このフラグは上級ユーザーのみが使ってください。ペナルティパラメーターの設定を誤ると、収束に失敗する可能性があります。

/ODRT={*chiSqTol*, *paramTol*}

0 以外の場合、/ODR です。ODR フィッティングの収束許容誤差を設定します。*chiSqTol* は、残差の二乗和に対する相対収束許容誤差です。*paramTol* は、フィッティングパラメーターに対する相対収束許容誤差です。いずれかの値が 0 以下の場合、ODRPACK のデフォルト値が使われます（*chiSqTol* には sqrt(machine epsilon)、*paramTol* には (machine epsilon)^(2/3)）。通常、デフォルトよりも厳密な、あるいは緩やかな収束基準が必要な場合を除き、これらを設定する必要はありません。

**注意：**
このフラグは上級ユーザーのみを対象としています。許容誤差を厳しくしすぎると収束しなくなる恐れがあり、逆に緩すぎると不正確な結果が生じる可能性があります。

**詳細**

kwCWave キーワードを使って係数ウェーブを指定しない限り、CurveFit は、ユーザーによる初期推定値（/G）が指定された場合、Kn システム変数から初期推定値を取得します。
kwCWave キーワードを使って係数ウェーブを指定しない限り、最終的なカーブフィッティングパラメーターは、ウェーブ W_coef に書き込まれます。

以前のバージョンの Igor Pro との互換性を保つため、パラメーターはシステム変数 Kn にも保存されます。
これにより、混乱を招く可能性があります。
W_coef を出力係数、Kn を上書きされる入力係数として考えることを推奨します。

その他の出力ウェーブには、M_Covar（/M フラグを参照）、M_FitConstraint と W_FitConstraint（/C パラメーターとヘルプ Fitting with Constraints を参照）、W_sigma があります。
/F パラメーターを使ってフィッティング係数の信頼区間を選択した場合、フィッティング係数の信頼区間を含む W_ParamConfidenceInterval というウェーブが生成されます。

CurveFit は、その他のカーブフィッティングの統計情報を、名前が「V_」で始まる変数に格納します。
また、CurveFit は、その動作を変更するために使用できる特定の V_ 変数を参照します。
詳細については、ヘルプ Curve Fitting と Special Variables for Curve Fitting を参照してください。

/ODR=nonzero を指定して推定を行う場合、制約条件の設定は単純な「境界制約」に限定されます。
つまり、フィッティング係数が特定の値より大きいか小さいかといった制約を設定することができます。
複数のフィッティング係数の組み合わせによる制約は、/ODR=0 を指定した場合にのみサポートされます。
制約条件の入力方法は同様で、「K0 > 1」のような式を使います。

**ウェーブのサブレンジの詳細**

CurveFit で指定するウェーブは、ほとんどの場合、あるウェーブのサブレンジ（一部）とすることができます。
ウェーブのサブレンジの構文は、Display コマンドの場合と同じです。
詳細については、ヘルプ Subrange Display Syntax を参照してください。
ただし、Display コマンドでは、範囲（その次元からの複数の要素）を指定できるのは 1 次元のみです。
CurveFit に多次元ウェーブが適している場合は、複数の次元に対して範囲を指定することができます。

一部のウェーブは、他のウェーブと同じポイント数でなければなりません。
例えば、１次元の Y ウェーブは、どの X ウェーブとも同じポイント数でなければなりません。
したがって、X ウェーブにサブレンジを使う場合、そのサブレンジのポイント数は、Y ウェーブで使われているポイント数と一致しなければなりません（この規則に関する複雑な点については、次の「サブレンジの後方互換性」のセクションを参照）。

ウェーブのサブレンジの一般的な用途としては、すべてのデータに加え、残差やモデル値を１つの複数列ウェーブにまとめることが挙げられます。
単変量フィッティングの場合は、X と Y のウェーブに加え、結果（モデル）ウェーブと残差ウェーブが必要になることがあります。
これらすべてを、４列のウェーブを使って実現できます。
例えば：

```
Make/D/N=(100, 4) Data
... 0 列目を X データで、1 列目を Y データで埋める ...
CurveFit poly 3, Data[][1] /X=Data[][0]/D=Data[][2]/R=Data[][3]
```

なお、すべてのウェーブは１つの複数列ウェーブから抽出された完全な列であるため、データポイントの数は必ず同じになります。

X ウェーブ（*xwaveName* または {*xwave1*, *xwave2*, ...}）、重み付けウェーブ（*wghtwaveName*）、マスクウェーブ（*maskWaveName*）、宛先ウェーブ（*destwaveName*）、残差ウェーブ（*residwaveName*）に使われるポイント数は、Y ウェーブ（*waveName*）に使われるポイント数と同じでなければなりません。
独自の信頼区間ウェーブ（/F フラグ）を指定する場合、それらは Y ウェーブと一致している必要があります。
信頼区間ウェーブではサブレンジを使うことはできません。
/ODR を 0 以外に設定する場合、X 重み、ホールド、宛先、残差ウェーブは Y ウェーブと一致している必要があります。

各ウェーブの合計ポイント数が他のウェーブと一致する必要はありません。
指定されたサブレンジ内のポイント数だけが一致していれば大丈夫です。

単変量フィッティング関数（ほぼすべてのフィッティングタイプを含む）にフィッティングさせる場合、Y ウェーブは実質的に１次元でなければなりません。
つまり、Y ウェーブは１次元ウェーブであるか、または使われるデータを１次元化するサブレンジを持つ必要があります。
例えば：

```
Make/N=(100,100) Ydata              // 2D ウェーブ
CurveFit gauss Ydata[][0]           // OK - １列が１次元
CurveFit gauss Ydata[2][]           // OK - １行が１次元
CurveFit gauss Ydata                // not OK - Ydata が２次元
CurveFit gauss Ydata[][0,1]         // not OK - ２列が 2D サブレンジを作成
```

多変量関数（poly2D または Gauss2D）をフィッティングする時、Y データを１次元にするか２次元にするかを選択できます。
１次元の場合、XYZ（または Y,X1,X2）のトリプレットをフィッティングする必要があります。
その場合、１次元の Y ウェーブと２つの１次元の X ウェーブ、あるいは複数列のウェーブから２つの列を指定する必要があります。
例えば：

以下は OK です：

```
Make/N=(100,3) myData
CurveFit Gauss2D myData[][0] /X={myData[][1],myData[][2]}
CurveFit Gauss2D myData[][0] /X=myData[][1,2]
```

以下は OK ではありません：

```
// 1D X ウェーブを持つ 2D Y ウェーブ
CurveFit Gauss2D myData /X={myData[][1],myData[][2]}
// X 列が多すぎる
CurveFit Gauss2D myData[][0] /X=myData
```

2D Y ウェーブを使う場合、X1 と X2 のデータは、グリッドの位置とウェーブの X と Y インデックスのスケーリングから取得することができ、１次元ウェーブやウェーブのサブレンジ（一部）を使ってグリッドの X1 と X2 の位置を指定することもできます。

```
Make/N=(20,30) yData
CurveFit Gauss2D yData       //  OK - 2D Y データ、スケーリングからの X1 と X2
Make/N=20 x1Data
Make/N=30 x2Data
// OK: ２次元の有効な Y データ（1D の X と Y フラグと一致）
CurveFit Gauss2D yData[0,9][0,19] /X=x1Data[0,9]/Y=x2data[10,29]
// OK: 有効な 2D Y データ
Make/N=(10,20,3) Y data
CurveFit Gauss2D yData[][][0]
```

もちろん、考えられる組み合わせは数多くあり、すべてを挙げるのは不可能です。

**サブレンジの後方互換性**

従来、Y ウェーブにはサブレンジが存在する場合がありました。
このサブレンジは、他のすべてのウェーブにも同様に適用されます。
下位互換性を確保するため、Y ウェーブにのみサブレンジを使い、他のウェーブにサブレンジがない場合、それらのウェーブは以下のいずれかを満たす必要があります：

1) Y ウェーブのポイント総数と同じポイント総数である場合、Y ウェーブのサブレンジが適用される

または

2) Y ウェーブのサブレンジと同じポイント総数である場合

さらに、Y ウェーブには括弧で囲んだサブレンジを指定して、そのサブレンジが Y ウェーブのスケーリングされたインデックス（X スケーリング）を指すことを示すことができます。
括弧を使って X 範囲を指定する場合は、従来のサブレンジのルールに従う必要があります。
つまり、すべてのウェーブのポイント数が同じでなければなりません。
サブレンジは Y ウェーブにのみ指定可能です。
Y ウェーブのサブレンジは、他のすべてのウェーブにも適用されます。

**信頼区間の詳細**

/F={...} パラメーターをウェーブの名前を指定せずに使うと、信頼区間および予測区間が自動的に生成されます。
１～４つのウェーブが生成されますが、*confKind* と *confStyle* の設定に応じて、１～４つのウェーブの名前を自分で指定することも可能です。

/F={*confLevel*, *confKind*, *confStyle*} によって自動生成されるウェーブ：

| **confKind** | **confStyle** | **取得するもの** | **自動ウェーブ名** |
|---|---|---|---|
| 1 | "Contour" | 上側と下側の信頼区間 | UC_*dataName* / LC_*dataName* |
| 2 | "Contour" | 上側と下側の予測区間 | UP_*dataName* / LP_*dataName* |

| | | | |
|---|---|---|---|
| 3 | "Contour" | 上側と下側の信頼区間と予測区間 | UC_*dataName* / LC_*dataName*<br>UP_*dataName* / LP_*dataName* |
| 1 | "ErrorBar" | 信頼区間ウェーブ | CI_*dataName* |
| 2 | "ErrorBar" | 予測区間ウェーブ | PI_*dataName* |
| 3 | "ErrorBar" | 信頼区間と予測区間ウェーブ | CI_*dataName*<br>PI_*dataName* |

なお、係数の信頼区間も計算したい場合は、*confKind* に 4 を加える必要があります。

データウェーブが最前面のグラフに表示されている場合、コンターウェーブはトレースとして最前面のグラフに追加されます。
ウェーブ名については、*dataName* が、フィッティングの Y データを含むウェーブの名前で置き換えられます。

/F={*confLevel*, *confKind*, *confStyle*, *wave*, *wave*...} で指定する必要があるウェーブ：

| **confKind** | **confStyle** | **指定するもの** |
|---|---|---|
| 1 | "Contour" | 上下の信頼区間コンターを受け取る 2 つのウェーブ |
| 2 | "Contour" | 上下の予測区間コンターを受け取る 2 つのウェーブ |
| 3 | "Contour" | 上下の信頼区間コンターと上下の予測区間コンターを受け取る 4 つのウェーブ |
| 1 | "ErrorBar" | 信頼区間の幅の値を取得する 1 つのウェーブ |
| 2 | "ErrorBar" | 予測区間の幅の値を取得する 1 つのウェーブ |
| 3 | "ErrorBar" | 信頼区間と予測区間の幅の値を取得する 2 つのウェーブ |

指定するウェーブは、従属変数のデータウェーブと同じ数のデータポイントを含む必要があります。
幅の間隔は、入力データの X 値に基づいて計算されます。
これらのウェーブはグラフに自動的に追加されることはありません。
コンターウェーブをトレースとして表示するか、エラーバーウェーブを使ってモデルフィッティングウェーブにエラーバーを作成することが想定されています。

**残差の詳細**

残差は、フィッティングに含まれる *waveName* の要素に対応する要素についてのみ計算されます。
したがって、複数のステップで実行された区分フィッティングの残差を自動的に計算することができます。

グラフに Y データが表示されている場合、自動残差ウェーブが最前面のグラフに追加されます。
これは、Y データの表示に使われている軸の真上に配置された新しい自由軸に追加され、積み重ねグラフが作成されます。
新しい軸のためのスペースを確保するため、他の軸は必要に応じて縮小されます。
軸の書式設定は後で変更できます。
ヘルプ Creating Stacked Plots を参照してください。

Igor Pro は、見栄えの良い積み上げグラフを作成するためにかなりの手間をかけるものの、グラフの書式設定に対する変更が、場合によっては望ましくないこともあります。
/A=0 を指定すると、グラフへの自動追加を抑制できます。
残差のウェーブは自動的に作成され、残差値で塗りつぶされますが、グラフには追加されません。

**カーブフィッティング画面の更新**

画面の更新では、前回の更新以降に変更があったデータが表示されているウィンドウが再描画されます。更新にはかなりの時間がかかる場合があるため、/N フラグを使うことで、カーブフィッティングの実行中に更新をコントロールすることができます。

Igor Pro では従来、カーブフィッティングの反復処理のたびに画面を更新していたため、フィッティング結果を示すグラフには、各反復処理の後に最新の暫定解が表示されていました。
これはデフォルトの動作であり、/N=0 と同等です。

プロセッサの処理速度が向上したことで、反復計算のたびに更新を行うことの有用性は低下しました。というのも、ほとんどの場合、反復計算間の間隔は短く、フィッティング全体が短時間で完了するためです。
そのため、Igor Pro 7.0 以降では、デフォルト設定を変更し、フィッティングの終了時のみ更新を行うようにしました。
これは /N=1 に相当します。

画面の更新を抑制することには、いくつかの潜在的な落とし穴があります。

/X フラグを使って、Igor Pro にグラフの X 軸全範囲にわたってフィッティング曲線を外挿するよう指示し、かつプロシージャ内で X 軸の範囲を変更する場合、Igor Pro が X 軸範囲の変更を確定できるように、CurveFit または FuncFit を呼び出す前に DoUpdate を呼び出す必要があります。

ユーザー定義関数から CurveFit または FuncFit を呼び出し、自動フィッティング機能を有効にする /D フラグを使った場合、呼び出しチェーン内のすべての実行中の関数が戻るまで、フィッティング曲線は更新されません。
関数が戻る前に、フィッティング結果がフィッティングウェーブを表示するグラフに反映されることが重要な場合は、DoUpdate を呼び出す必要があります。

**参考（ヘルプ）**

Curve Fitting、Special Variables for Curve Fitting、Fitting with Constraints
Accessing Variables Used by Igor Operations

ユーザー指定関数へのフィッティングについては、FuncFit コマンドを参照してください。
多変量ユーザー指定関数へのフィッティングについては、FuncFit と FuncFitMD を参照してください。
信頼区間と予測区間の詳細については、ヘルプ Confidence Bands and Coefficient Confidence Intervals を参照してください。