

CONTENTS

ビジュアルヘルプ - カーブフィッティング (2)	3
組み込みのカーブフィッティング関数	3
組み込みフィッティングの入力と出力	11
Curve Fitting ダイアログのタブ	13
全体のコントロール	13
Function and Data タブ	13
Data Options タブ	16
Coefficients タブ	17
Output Options タブ	17
残差の計算	18
自動トレースを使った残差	19
残差オートトレースの削除	20
自動ウェーブを用いた残差	20
明示的な残差ウェーブを用いた残差	20
新しいウェーブを用いた明示的な残差ウェーブ	21
フィッティング後の残差の計算	21
決定係数または R^2	22
R^2 値と原点を通るフィッティング	23
誤差の推定	24
信頼帯と係数の信頼区間	24
フィッティング後の信頼区間の計算	26
信頼帯ウェーブ	27
いくつかの統計	27
信頼帯と非線形関数	28
共分散行列	28
相関行列	29
識別可能性の問題	29
制約付きフィッティング	30
Curve Fitting ダイアログを使った制約	30
制約ウェーブを使った複雑な制約	31
制約式	31

等式制約	32
制約付きフィッティングの例.....	32
制約行列と制約ベクトル.....	34
制約付きカーブフィッティングにおける落とし穴	35
カーブフィッティングにおける NaN と INF	36
カーブフィッティング用の特殊変数	36
V_FitOptions.....	38
V_chisq	40
V_q	40
V_FitError と V_FitQuitReason	40
V_FitIterStart.....	41
S_Info.....	42
変数における誤差：直交距離回帰	42
スレッドセーフな ODR Fitting フィッティング	43
ODR フィッティングのための重み付けウェーブ	43
ODR 初期推定値	43
独立変数の調整を保持する	44
ODR フィッティングの結果	44
制約と ODR フィッティング	45
ODR フィッティングからの誤差推定.....	45
ODR フィッティングの例.....	45

ビジュアルヘルプ – カーブフィッティング（2）

組み込みのカーブフィッティング関数

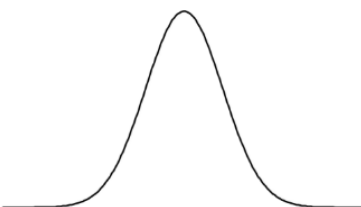
ほとんどの場合、自動推定を使えば良好な結果が得られます。

一部については、前のセクションで要約した内容に加えて追加の入力が必要です。

このセクションでは、有用と思われる場合に詳細を補足する、フィッティング関数に関する注記を記載します。

gauss

これはガウス分布のピークにフィッティングします。

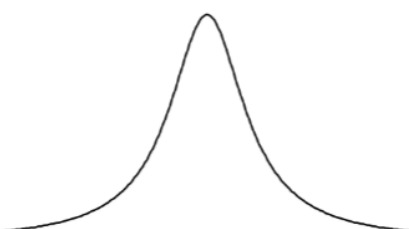
$$y_0 + A \exp \left[-\left(\frac{x - x_0}{width} \right)^2 \right]$$


width パラメータはピークの標準偏差の平方根 2 倍であることに注意してください。

これはガウス関数の w_i パラメータとは異なり、 w_i は単純に標準偏差です。

lor

これはローレンツ分布のピークにフィッティングします。

$$y_0 + \frac{A}{(x - x_0)^2 + B}$$


Voigt

Voigt（フォークト）ピーク（ローレンツ分布とガウス分布の畳み込み）にフィッティングします。

ローレンツ分布成分とガウス分布成分の幅の比率を変更することで、形状をローレンツ分布形状からガウス分布形状へと段階的に変化させることができます。

$$y_0 + \frac{2Area}{W_G} \sqrt{\frac{\ln(2)}{\pi}} \bullet Voigt \left[\frac{2\sqrt{\ln(2)}}{W_G} (x - x_0), Shape \bullet 2\sqrt{\ln(2)} \right]$$

Voigt 関数は正規化された Voigt ピーク形状関数です。

これは組み込み関数 VoigtFunc と同じです。

プリマルチプライヤーと Voigt 関数の引数により、ピーク形状が形成され、そのフィッティング係数は以下の通りです：

Area 垂直オフセットを除いたピーク下の領域

y0 垂直方向オフセット

W_G ガウシアン半値全幅 (FWHM)

Shape ローレンツ成分とガウス成分の比、 W_L/W_G

Voigt ピークの高さを計算する解析式は存在しません。

Voigt ピークの半値全幅 (FWHM) は近似として：

$$W_V = \frac{W_L}{2} + \sqrt{\frac{W_L^2}{4} + W_G^2}$$

ここで、 $W_L = \text{Shape} * W_G$ です。

形状の極値を持つピークのフィッティングは数値的不安定性の問題があるため推奨されません。
代わりに端点形状である Lor (形状 $\rightarrow\infty$) または Gauss (形状=0) でフィッティングしてください。

exp_XOffset

これは減衰する指数関数にフィッティングします。

$$y_0 + A \exp \left\{ \frac{-(x-x_0)}{\tau} \right\}$$

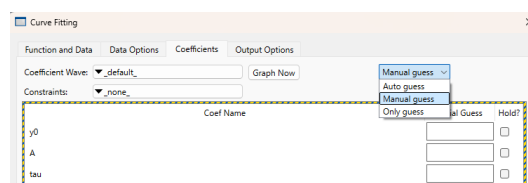

この式において、 x_0 は定数であり、フィッティング係数ではありません。
自動推定の生成時には、 x_0 はフィッティングデータの最初の X 値に設定されます。
これにより、浮動小数点の丸め誤差による問題が解消されます。

CurveFit コマンドで /K フラグを使って x_0 の値を設定できますが、自動値を受け入れることを推奨します。
入力データの初期 X 値から遠く離れた値を x_0 に設定すると、問題が発生することが確実です。

注記 フィッティング係数 τ は、exp 関数における等価係数の逆数です。これは実際には減衰定数であり、逆減衰定数ではありません。

自動推定は、指数関数的に増加する (負の τ) 場合には機能しません。

負の τ 値をフィッティングさせるには、Coefficients タブで Manual Guess を使うか、コマンドラインで CurveFit/G を実行してください。



dblexp_XOffset

2つの減衰する指数関数の和にフィッティングします。

$$y_0 + A_1 \exp \left\{ \frac{-(x-x_0)}{\tau_1} \right\} + A_2 \exp \left\{ \frac{-(x-x_0)}{\tau_2} \right\}$$


この式において、 x_0 は定数であり、フィッティング係数ではありません。
自動推定の生成時には、 x_0 はフィッティングデータ内の最小 X 値に設定されます。
これにより、浮動小数点の丸め誤差による問題が解消されます。

CurveFit コマンドの /K フラグを使って x0 の値を設定することも可能ですが、自動設定値を受け入れることを推奨します。

入力データの初期 X 値から大きく離れた値を x0 に設定すると、確実に問題が発生します。

注記 フィッティング係数 τ_1 と τ_2 は、dblexp 関数における対応する係数の逆数です。これらは実際の減衰定数であり、逆数ではありません。

指数関数の成長については、exp_XOffset の注記を参照してください。

また、振幅の符号が反対の場合には手動での推定も必要になります。

下記の dblexp_peak フィッティング関数も参照してください。

2つの減衰定数 (τ_1 と τ_2) が十分に異なる場合でなければ、正確な結果が得られない可能性があります。

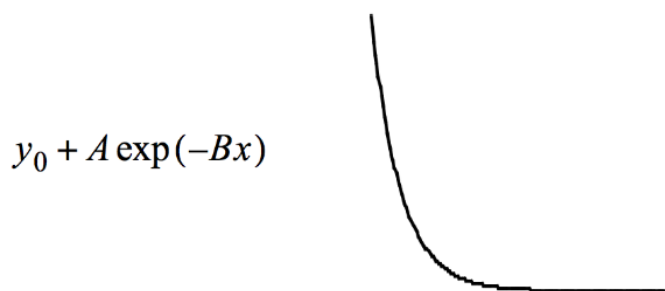
exp

減衰する指数関数にフィッティングします。

exp_XOffset と似ていますが、同等の堅牢性はありません。

下位互換性のために含まれています。

新規作業では exp_XOffset を使ってください。



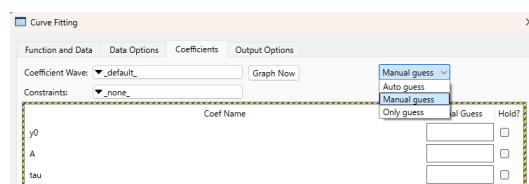
X 方向へのデータオフセットは A 値に変化をもたらすことに注意してください。

X 位置に依存しない結果を得るには exp_XOffset を使ってください。

注記 フィッティング係数 B は逆減衰定数です。

自動推定は指数関数的な増加（負の B）には機能しません。

負の B 値をフィッティングさせるには、Coefficients タブで Manual Guess を使うか、コマンドラインで CurveFit/G を実行してください。



浮動小数点演算のオーバーフローは、X オフセットが大きい指数関数をフィッティングする時に問題を引き起こします。

この問題は、時間の経過に伴う減衰をフィッティングする時に、時間がしばしば大きくなるため、頻繁に発生します。

最善の解決策は、exp_XOffset フィッティング関数を使うことです。

そうでない場合、そのようなデータをフィッティングするには、X 値をゼロ方向へオフセットし直す必要があります。

入力 X 値を変更することも可能ですが、通常はコピーで作業するのが最善です。

コマンドラインの Duplicate コマンド、または Data メニューの Duplicate Waves を使ってデータをコピーしてください。

XY ペアに対しては、コマンドラインで以下のコマンドを実行してください（これらのコマンドは、myXWave_copy という複製ウェーブを作成済みであることを前提としています）：

```
Variable xoffset = myXWave_copy[0]
myWave_copy[0] -= xoffset
```

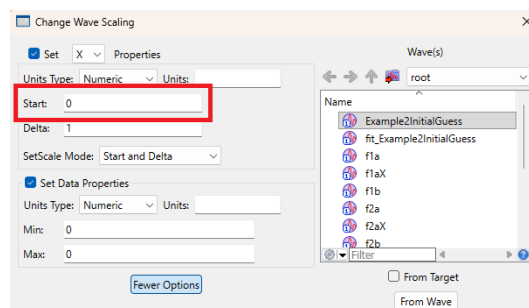
これらのコマンドは、ウェーブの先頭からデータをフィッティングすることを前提としています。部分データセットをフィッティングする場合は、[0] をフィッティングする最初のポイントの番号に置き換えてください。

グラフカーソルを使ってポイントを選択する場合は、[pcsr(A)] を代入してください。

これは、丸いカーソル（カーソル A）がデータの先頭をマークしていることを前提としています。

ウェーブフォームデータにフィッティングする場合（X Data メニューで _calculated_ を選択した場合）、データオフセットのためにウェーブスケーリングの x0 部分を設定する必要があります。

ウェーブ全体にフィッティングする場合は、Data メニューの Change Wave Scaling ダイアログを使い、スケーリングの x0 部分をゼロに設定してください。



グラフカーソルで選択した部分データにフィッティングする場合は、コマンドラインでスケーリングを変更する方が簡単です。

```
SetScale/P x leftx(myWave_copy)-xcscr(A), deltax(myWave_copy), myWave_copy
```

このコマンドは、データの先頭を丸いカーソル（カーソル A）でマークしたことを前提としています。

X オフセットを引くと、フィッティングにおける振幅係数が変化します。

多くの場合、関心のある係数は減衰定数（invTau）のみであり、振幅の変化は無視できます。

そうでない場合は、フィッティング終了後に正しい振幅を計算できます。

```
W_coef[1] = W_coef[1]*exp(W_coef[2]*xoffset)
```

ウェーブフォームデータをフィッティングする場合、xoffset の値は -leftx(myWave_copy) となります。

dblexp

減衰する指数関数の和をフィッティングさせます。

dblexp_XOffset と類似していますが、データが $x=0$ に十分近く開始しない場合、浮動小数点の丸め誤差の問題が生じます。

これは後方互換性のために含まれています。

新規作業では exp_XOffset を使ってください。

$$y_0 + A_1 \exp(-B_1 x) + A_2 \exp(-B_2 x)$$

X 方向へのデータオフセットは A1 および A2 に変化をもたらすことに注意してください。

X 位置に依存しない結果を得るには dblexp_XOffset を使ってください。

注記 フィッティング係数 B1 および B2 は逆減衰定数です。

指数関数の成長に関する注記を参照してください。

また、振幅の符号が反対の場合には手動での推定も必要になります。

下記の `dblexp_peak` フィッティング関数も参照してください。

2つの減衰定数（B1 と B2）が十分に異なる場合でなければ、正確な結果が得られない可能性があります。

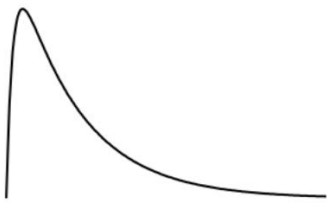
X オフセットが大きなデータにフィッティングすると、`exp` でフィッティングする場合と同様の問題が生じます。

最善の解決策は `dblexp_XOffset` フィッティング関数を使うことです。

また、上記の `exp` で説明したプロシージャと同様の手法で問題を解決することも可能です。

dblexp_peak

符号の異なる2つの指数項の和にフィッティングし、ピークを形成します。

$$y_0 + A \cdot \left\{ -\exp \left[\frac{-(x-x_0)}{\tau_{u1}} \right] + \exp \left[\frac{-(x-x_0)}{\tau_{u2}} \right] \right\}$$


ピークの位置は以下によって与えられます。

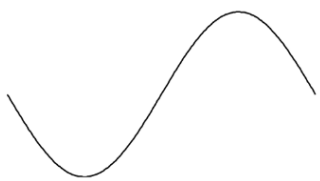
$$X_{peak} = \frac{\tau_{u1} \tau_{u2} \ln \left(\frac{\tau_{u1}}{\tau_{u2}} \right)}{\tau_{u1} - \tau_{u2}} + x_0$$

2項の振幅を個別にコントロールする必要がある場合は、`dblexp_XOffset` 関数を使ってください。

その場合、手動での推定が必要になります。

sin

これは正弦波にフィッティングします。

$$(y_0 + A \sin(fx + \phi))$$


ϕ はラジアンです。

度に変換するには、 $180/\pi$ を掛け算します。

正弦波フィッティングでは、正弦波のおおよその周波数を自動推定に伝える追加パラメータを使います。

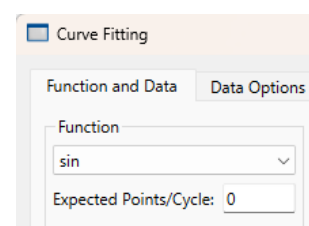
これは、1周期あたりのデータポイントのおおよその数として入力します。

Function メニューから `sin` を選択すると、1周期あたりの予想ポイント数を入力するボックスが表示されます。

6未満の数値を入力した場合、デフォルト値は7となります。

良好な結果を得るには、様々な値を試す必要があるかもしれません。

単純に手動での推定を使うことも考えられます。



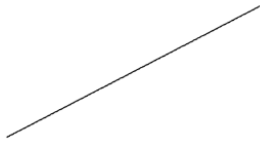
`sin` 関数の性質上、カーブフィッティングでは位相が 2π 異なるものは区別できません。

フィッティングを目的範囲内に収めようとするよりも、 $2n\pi$ を差し引く方がおそらく簡単です。

line

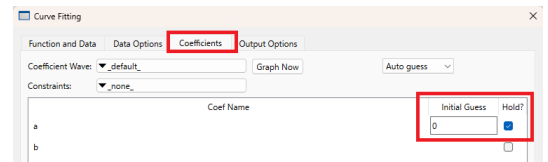
これはデータを通る直線にフィッティングします。

$$(a + bx)$$



手動での推定は一切不要です。

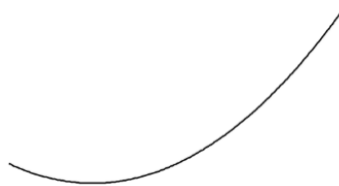
原点を通る直線をフィッティングしたい場合は、Coefficients タブで係数 a の Hold ボックスを選択し、Initial Guess をゼロに設定してください。



poly n

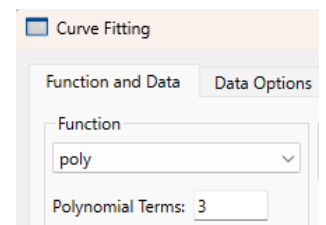
これは n 項の多項式、すなわち次数 n-1 の多項式にフィッティングします。

$$(K_0 + K_1x + K_2x^2 + \dots)$$



多項式フィッティングでは、多項式の次数を設定する追加パラメータが必要です。Function メニューから poly を選択すると、その値を入力するボックスが表示されます。

n の最小値は 1 で、これは Y 値の平均を取ることに相当します。
ほとんどのユーザーは 3（二次多項式）未満の値を必要としません。



多項式近似には手動での推定が一切不要です。

poly_XOffset n

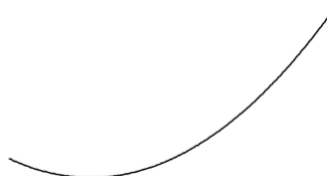
n 項の多項式、または次数 n-1 の多項式をフィッティングします。

定数 x0 は調整可能なフィッティング係数ではありません。

これにより、多項式を X 軸上の任意の位置に配置できます。

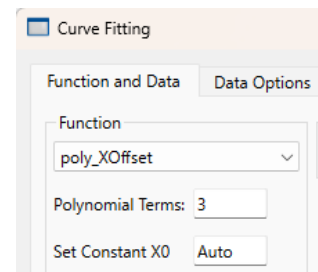
これは特に X 値が大きい場合、例えば日付/時刻データを扱う時に役立ちます。

$$(K_0 + K_1(x - x_0) + K_2(x - x_0)^2 + \dots)$$



poly_XOffset フィッティング関数は、多項式の次数と x0 の値を設定する追加パラメーターを受け取ります。

Function メニューから poly_XOffset を選択すると、これらの値を入力するボックスが表示されます。



n の最小値は 1 で、これは Y 値の平均を取ることに相当します。

ほとんどのユーザーは 3（二次多項式）未満の値を必要としません。

Set Constant X0 を自動的に設定すると、x0 はフィッティング対象データ内で検出された最小 X 値に設定されます。

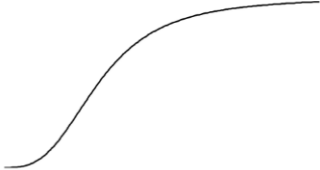
x0 は任意の値に設定可能です。

データセットの X 値から大きく離れた値を設定すると数値計算上の問題が発生します。

多項式フィッティングでは、手動での推定は一切不要です。

Hill Equation

これはヒルの方程式、すなわちシグモイド関数にフィッティングします。

$$base + \frac{(max - base)}{1 + \left(\frac{x_{1/2}}{x}\right)^{rate}}$$


係数 base は X が小さい時の y 値を設定し、max は X が大きい時の y 値を設定します。

rate は上昇速度を設定し、x1/2 は Y が (base+max)/2 となる X 値を設定します。

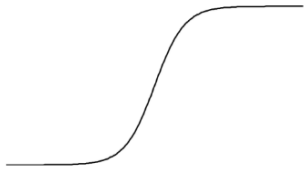
X 値は 0 より大きくなければならないことに注意してください。

X ≤ 0 のデータポイントを含めると、特異行列エラーが発生し、「The fitting function returned NaN for at least one X value. (フィッティング関数が少なくとも 1 つの X 値に対して NaN を返しました)」というメッセージが表示されます。

base と max 値を反転させれば、下降シグモイド関数を表現できます。

sigmoid

これはヒル方程式とは異なる形状を持つシグモイド関数をフィッティングします。

$$base + \frac{max}{1 + \exp\left(\frac{x_0 - x}{rate}\right)}$$


係数 base は X が小さい場合の Y 値を設定します。

X が大きい場合の Y 値は base+max となります。

x0 は Y が (base+max / 2) となる X 値を設定し、rate は上昇率を設定します。

rate が小さいほど上昇が速くなり、具体的には x=x0 における傾きは max / (4×rate) となります。

power

これはべき乗則にフィッティングします。

$$(y_0 + Ax^{\text{pow}})$$



これはフィッティングが困難な場合があります、特に $\text{pow} > 1$ または pow がゼロに近い場合には、適切な初期推定が必要となります。

X 値は 0 より大きくなければならないことに注意してください。

$X \leq 0$ のデータポイントを含めると、特異行列エラーが発生し、「"The fitting function returned NaN for at least one X value. (フィッティング関数が少なくとも 1 つの X 値に対して NaN を返しました)」というメッセージが表示されます。

lognormal

対数正規分布のピーク形状にフィッティングします。

この関数は対数 X 軸でプロットするとガウス分布となります。

$$y_0 + A \exp \left[- \left(\frac{\ln(x/x_0)}{\text{width}} \right)^2 \right]$$



係数 y_0 は基準値を、 A は振幅を、 x_0 は X 方向におけるピーク位置を、 width はピーク幅を設定します。

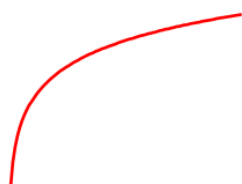
X 値は 0 より大きくなければならないことに注意してください。

$X \leq 0$ のデータポイントを含めると、特異行列エラーが発生し、「"The fitting function returned NaN for at least one X value. (フィッティング関数が少なくとも 1 つの X 値に対して NaN を返しました)」というメッセージが表示されます。

log

これは対数曲線にフィッティングします。

$$a + b \cdot \log_{10}(x)$$



$\log$ は初期推定の誤りを非常に寛容に扱います。

これは Igor Pro 9.0 で追加されました。

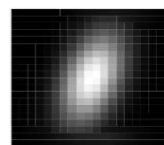
X 値は 0 より大きくなければならないことに注意してください。

$X \leq 0$ のデータポイントを含めると、特異行列エラーが発生し、「"The fitting function returned NaN for at least one X value. (フィッティング関数が少なくとも 1 つの X 値に対して NaN を返しました)」というメッセージが表示されます。

gauss2D

2次元でガウスピークをフィッティングします。

$$z_0 + A \exp \left[\frac{-1}{2(1 - \text{cor}^2)} \left(\left(\frac{x - x_0}{xwidth} \right)^2 + \left(\frac{y - y_0}{ywidth} \right)^2 - \frac{2\text{cor}(x - x_0)(y - y_0)}{xwidth \cdot ywidth} \right) \right]$$



相関係数 cor は交差相関の項であり、-1 から 1 の範囲でなければなりません（小さな図は cor=0.5 で作成）。この範囲は制約条件によって自動的に強制されます。

この項の値がゼロが適切であると分かっている場合は、この係数を固定できます。

cor をゼロに固定すると、通常はフィッティングがかなり高速化します。

ガウスフィッティング関数とは対照的に、xWidth と yWidth はピークの標準偏差です。

ガウス関数には相互相関パラメーター cor が含まれていないことに注意してください。

poly2D n

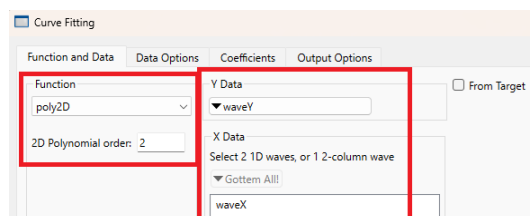
これは二次元において次数 n の多項式をフィッティングします。

$$(C_0 + C_1x + C_2y + C_3x^2 + C_4xy + C_5y^2 + \dots)$$



poly2D のフィッティングでは、多項式の次数を示す追加のパラメータが必要です。

関数メニューから poly2D を選択すると、その値を入力するボックスが表示されます。



最小値は 1 で、これは一次多項式、すなわち平面に相当します。

poly2D の係数ウェーブは、定数項 (K0) がポイントゼロに位置し、続くポイントには順に高次の項群が含まれます。

一次項は K1*x と K2*y の 2 つ、次に二次項が 3 つ、以下同様です。

項の総数は (N+1)(N+2)/2 であり、N は次数です。

Poly2D は手動での推定を一切必要としません。

組み込みフィッティングの入力と出力

カーブフィッティングには、様々な入力と出力を提供する複数の変数とウェーブが存在します。

通常は Curve Fitting ダイアログを使い、必要な項目はダイアログ上で明確に表示されます。

詳細な説明は本ヘルプファイルの様々な箇所で確認できます。

簡単な導入として、組み込み関数へのフィッティングに使われるウェーブと変数を一覧にした表を以下に示します。

ウェーブまたは変数	形式	用途
従属変数データウェーブ	入力	フィッティングする曲線の応答変数の測定値を含みます。しばしば「Y データ」と呼ばれます。

独立変数データウェーブ	入力	フィッティングする曲線の独立変数の測定値を含みます。 しばしば「X データ」と呼ばれます。
宛先ウェーブ	オプション出力	フィッティング中およびフィッティング後のグラフィカルな フィードバック用。 フィッティング中は、宛先ウェーブが継続的に更新され、現 在の係数で評価されたフィッティング関数が表示されます。
残差ウェーブ	オプション出力	データとモデルの差異。
重み付けウェーブ	オプション入力	個々の Y データポイントが、出力係数の探索にどれだけ寄 与するかをコントロールするために使われます。
システム変数 K0, K1, K2 ...	入力と出力	組み込みのフィッティング関数にのみ適用。 オプションでシステム変数から初期推定値を取得し、フィッ ティング終了時にそれらを更新します。
係数ウェーブ 更新	入力と出力	係数ウェーブから初期推定値を取得し、フィッティング中に し、最終係数を保持します。 追加オプションについては、CurveFit と FuncFit の参考資 料を参照してください。
イプシロンウェーブ	オプション入力	ユーザー定義のフィッティング関数のみに適用。 カーブフィッティングアルゴリズムが係数に対する偏微分を 計算するために使われます。
W_sigma	出力	このウェーブを生成し、その係数に対する誤差の推定値を格 納します。
W_fitConstants	出力	定数を含む組み込みのフィッティング関数を使ってフィッ ティングを実行した時に作成されます。 Igor はこのウェーブを作成し、フィッティング式で使われ る定数の値を保存します。 詳細は「定数を持つフィッティング」のセクションを参照し てください。 特定のフィッティング関数で使われる定数に関する注意事項 については、「組み込みカーブフィッティング関数」のセク ションを参照してください。
V_<xxx>	入力	カーブフィッティングアルゴリズムの動作を調整するために 設定できる、V_FitOptions などの特別な変数がいくつか存 在します。
V_<xxx>	出力	V_chisq や V_npnts などの複数の変数を作成し設定しま す。 これらはカーブフィッティングによって得られた様々な統計 量を含みます。
M_Covar	出力	オプションで「共分散行列」を含む行列ウェーブを生成しま す。 高度な統計量を生成するために使用できます。

その他のウェーブ オプション入力 ユーザーが提供した、または自動生成されたウェーブを出力し、信頼帯や予測帯を表示し、係数値に対する制約を指定します。

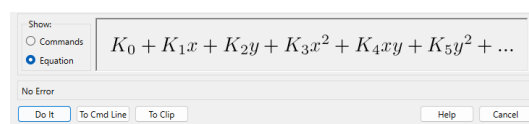
Curve Fitting ダイアログのタブ

このセクションでは、Curve Fitting ダイアログの各タブおよびメインペインにあるコントロールについて説明します。

全体のコントロール

ダイアログの下部にあるコントロールは常に利用可能です。

Commands ラジオボタンをクリックすると、Igor は方程式の代わりにダイアログで生成されたコマンドを表示します。



Function and Data タブ

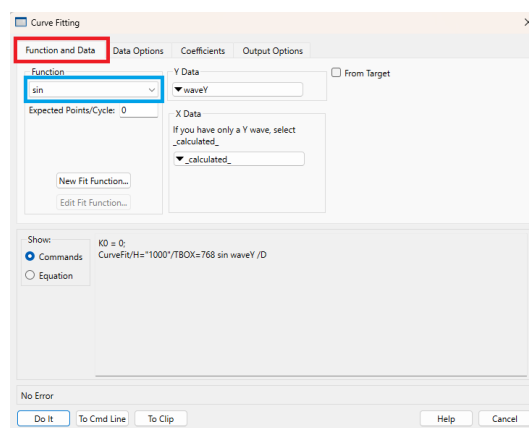
Function and Data タブは、Function メニューで選択した関数に応じて様々な表示になります。
組み込みの sin 関数が選択された場合の表示は右のようになります。

Function メニュー

主な目的は、フィッティングする機能を選択することです。

メニューには、メニューに表示される機能をコントロールする 2 つの項目もあります。

- ✓ Show Multivariate Functions
- Show Old-Style Functions



複数の独立変数の関数を含めるには、Show Multivariate Functions を選択してください。

FitFunc キーワードを持たない関数を表示するには、Show Old-Style Functions を選択してください。

FitFunc キーワードの詳細については、「ユーザー定義フィッティング関数」のセクションを参照してください。
バージョン 4 より前の Igor Pro で作成したフィッティング関数がある場合、この項目を選択する必要があるかもしれません。

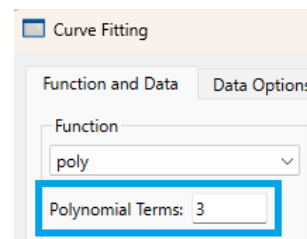
一部のフィッティング関数では、関数を選択した時に表示されるカスタマイズされた項目によって収集される追加情報が必要です。

Polynomial Terms

多項式をフィッティングさせる poly 関数を選択すると、この画面が表示されます。

項の数は次数より 1 大きいことに注意してください。

二次多項式の場合はこれを 3 に設定します。



Set Constant X0

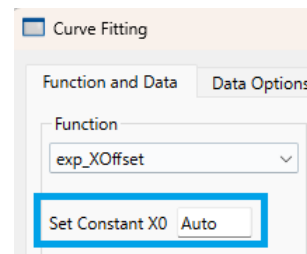
exp_XOffset、dblexp_XOffset、または poly_XOffset 関数を選択すると表示されます。

X0 は定数であり、X 値から差し引かれることで、フィッティングの X 範囲をゼロに近づけるために移動させます。

これにより、X がゼロから多少離れている場合でも、X 範囲で指数関数を評価する時に生じる数値上の問題が解消されます。

X0 を Auto に設定すると、入力データの最小 X 値に設定されます。

数値を設定すると、このデフォルト動作が上書きされます。



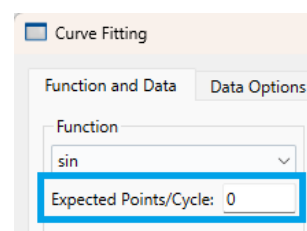
Expected Points/Cycle

sin 関数を選択した時に表示されます。

sin 関数の 1 周期におけるデータポイントのおおよその数を設定するために使います。

自動初期推定値が適切な推定値を導き出すのに役立ちます。

4 未満に設定された場合、Igor はデフォルト値である 7 を使います。



2D Polynomial Order

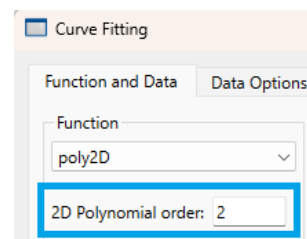
poly2D 関数を選択した時に表示されます。

二次元多項式（多変数関数）をフィッティングするための関数です（メニューに表示されるのは Show Multivariate Functions を選択した場合のみ）。

多項式の次数を設定します。

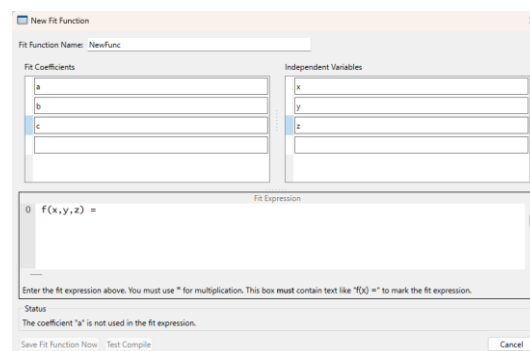
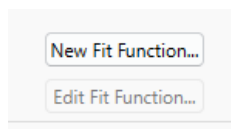
項の数ではありません。

特定の次数は X の項と Y の項、それらの交差項を含むため、項の数は $(N+1)(N+2)/2$ となります（N は次数）。



New Fit Function

このボタンをクリックすると、独自のフィッティング関数を定義できるダイアログが表示されます。



Y Data

フィッティングさせる従属変数データを含むウェーブを選択します。
Function メニューで多変量関数が選択されている場合、Y Data メニューには 1D ウェーブおよび関数が使う独立変数の数に一致する次元のウェーブが表示されます。

X Data

このエリアは、選択した機能と Y データウェーブによって変化します。

単変量関数の場合、X Data メニューのみが表示されます。
Y ウェーブのみがある場合は `_calculated_` を選択してください。
X 値は Y ウェーブの X スケーリングから取得されます。

X Data メニューには、Y ウェーブと同じポイント数のウェーブのみが表示されます。

多変量関数と 1D Y ウェーブを選択すると、X Data メニューの下にリストボックスが追加されます。
フィッティング関数で使う各独立変数に対して、1 つの X ウェーブを選択するか、適切な列数の 1 つの複数列ウェーブを選択する必要があります。
X ウェーブを選択すると、それらがリストに追加されます。
リスト内のウェーブの順序は重要です。
これは、関数の各独立変数にどのウェーブが割り当てられるかを決定します。

リストからウェーブを削除するには、ウェーブを選択して Backspace キーを押します。

多変量関数と多次元 Y ウェーブを選択すると、ウェーブが持つことができる各次元ごとに 1 つずつ、4 つの独立変数ウェーブのメニューが表示されます。

各メニューには、Y データウェーブの対応する次元に一致する長さのウェーブが表示されます。

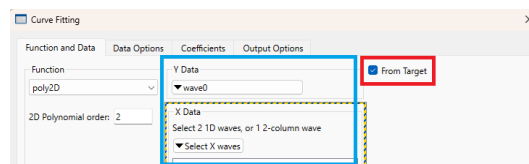
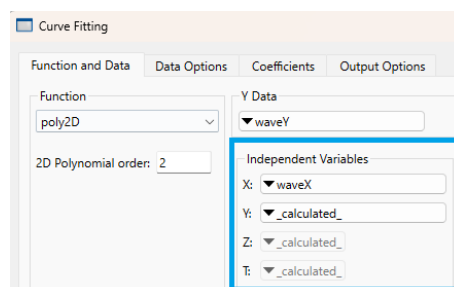
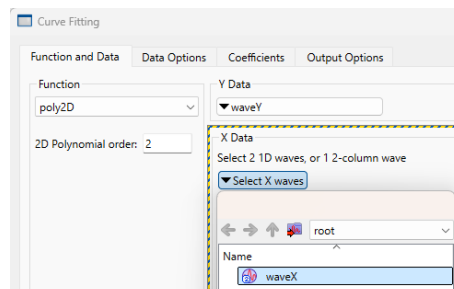
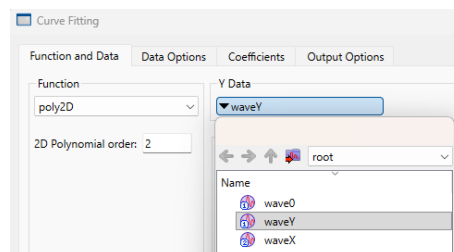
独立変数の値が Y ウェーブの次元スケーリングから得られる場合は、`_calculated_` を選択してください。

From Target

From Target を選択すると、Y Data および X Data メニューには、最前面のテーブルまたはグラフに表示されているウェーブのみが表示されます。

多数の実験のウェーブを扱う時に、メニュー内でのウェーブ検索を容易にします。

From Target チェックボックスをクリックすると、X データおよび Y データに適したウェーブの選択を試みます。



Data Options タブ

Range

Y データのサブセットにフィッティングさせる時の開始ポイントと終了ポイントのポイント番号を入力してください。

Cursors

最前面のウィンドウが Y データを表示するグラフであり、かつグラフカーソルが Y データトレース上に配置されている場合に使用可能です。

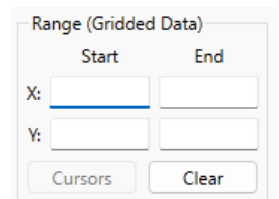
注記

データが Y ウェーブと X ウェーブの両方を使用し、X 値がランダムな順序で並んでいる場合、期待される結果は得られません。

Clear

このボタンをクリックすると、Start ボックスと End ボックスからテキストが削除されます。

多変量関数と多次元ウェーブを選択した場合、Range ボックスは変わります。ダイアログには、Y ウェーブの各次元に対応する Start ボックスと End ボックスが表示されます。



Weighting

重み付け値を含むウェーブを選択してください。

このメニューには、ポイント数と次元が Y ウェーブと一致するウェーブのみが表示されます。

詳細は「重み付け」のセクションを参照してください。

Wave Contains

重み付けウェーブに各 Y データポイントの標準偏差値が含まれている場合は、Standard Deviation を選択してください。

値が大きいほど、そのポイントがフィッティングに与える影響は小さくなります。

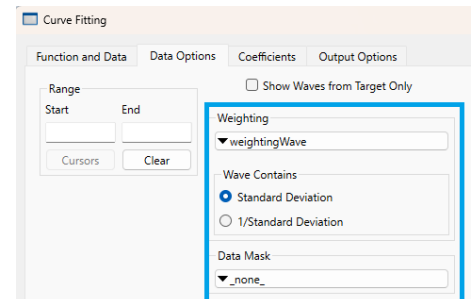
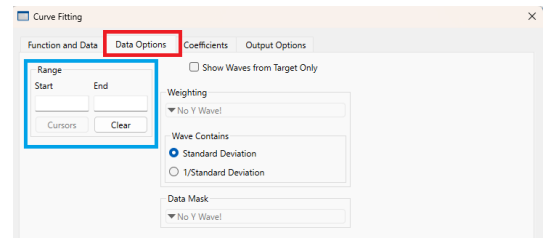
重み付けウェーブに標準偏差の逆数の値が含まれている場合は、1/Standard Deviation を選択してください。

値が大きいほど、そのポイントがフィッティングに与える影響が大きくなります。

Data Mask

1 と 0、または NaN を含むウェーブを選択し、どの Y データポイントをフィッティングに含めるかを指定します。このメニューには、Y ウェーブとデータポイント数および次元が一致するウェーブのみが表示されます。

1 は含めるデータポイントを示し、0 または NaN（数値ではない、または表の空白）は除外すべきポイントを示します。



Coefficients タブ

Coefficients タブは非常に複雑です。

その詳細な説明は、フィッティング方法に関する各セクションで詳細に説明しています。

「2つの便利な追加機能：係数の保持と残差の生成」「自動推定が機能しなかった場合」「ユーザー定義関数の係数タブ」「係数ウェーブ」の各セクションを参照してください。

Output Options タブ

Output Options タブには、フィッティングの結果の報告と表示をコントロールする設定が含まれています：

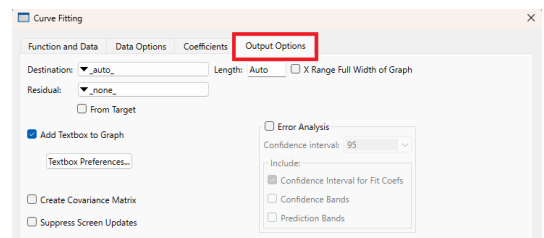
Destination

フィッティングからモデル値を受け取るウェーブを選択するか、_auto_ を選択するとモデル値用に等間隔の自動出力ウェーブを作成します。

各反復ごとに更新されるため、グラフ表示でフィッティングの進捗を追跡できます。

Destination メニューの詳細については「宛先ウェーブ」のセクションを参照してください。

上図に表示されている Length ボックスと、上記に表示されていない New Wave ボックスについても同様です。



X Range Full Width of Graph

グラフカーソルでフィッティング範囲を制限した場合、自動宛先ウェーブは選択範囲のみをカバーします。

このチェックボックスを選択すると、自動宛先ウェーブがグラフの全幅をカバーようになります。

Residual

ウェーブを選択すると、残差（モデルとデータの差）の計算値が表示されます。

残差の詳細とこのメニューで選択可能な各種設定については、「残差の計算」のセクションを参照してください。

Error Analysis

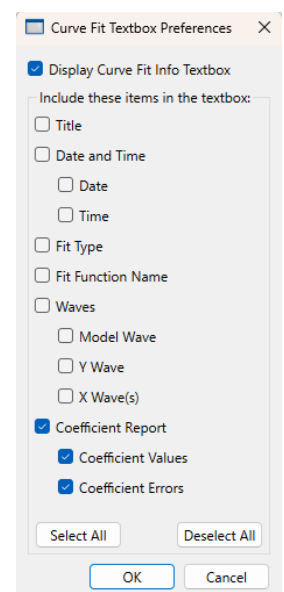
各種の統計的誤差分析を選択します。

詳細は「信頼区間および係数の信頼区間」のセクションを参照してください。

Add Textbox to Graph

選択すると、Y データを含むグラフにフィッティングに関する情報を表示するテキストボックスが追加されます。

Textbox Preferences ボタンをクリックすると、ダイアログが表示され、テキストボックスに含める各種情報を選択できます。



Create Covariance Matrix

このオプションを選択すると、ダイアログはフィッティング用の共分散行列を作成するコマンドを生成します。

共分散行列の詳細については、「共分散行列」のセクションを参照してください。

Suppress Screen Updates

このオプションを選択すると、フィッティングの進行中にグラフや表は更新されません。
これにより、特にフィッティングにコンタープロットや画像プロットが含まれる場合に、フィッティング処理を大幅に高速化できますが、フィッティング中のフィードバックは減少します。

残差の計算

残差とは、生データからフィッティング関数モデルを差し引いたときに残る値です。

理想的には、生データは既知の関数にランダムなノイズを加えたものと等しくなります。

データからこの関数を差し引いたとき、残る部分がノイズとなるはずですが、

これが当てはまらない場合、その関数は生データに適切にフィッティングしていないことになります。

右のグラフは、指数関数および二次関数（3 項多項式）にフィッティングさせた指数分布の生のデータを示しています。

指数関数によるフィッティングの残差はランダムな分布を示していますが、二次関数によるフィッティングの残差は、ランダムな分布の上にトレンドが重なっていることがわかります。

これは、二次関数がデータに対して適切なフィッティングではないことを示しています。

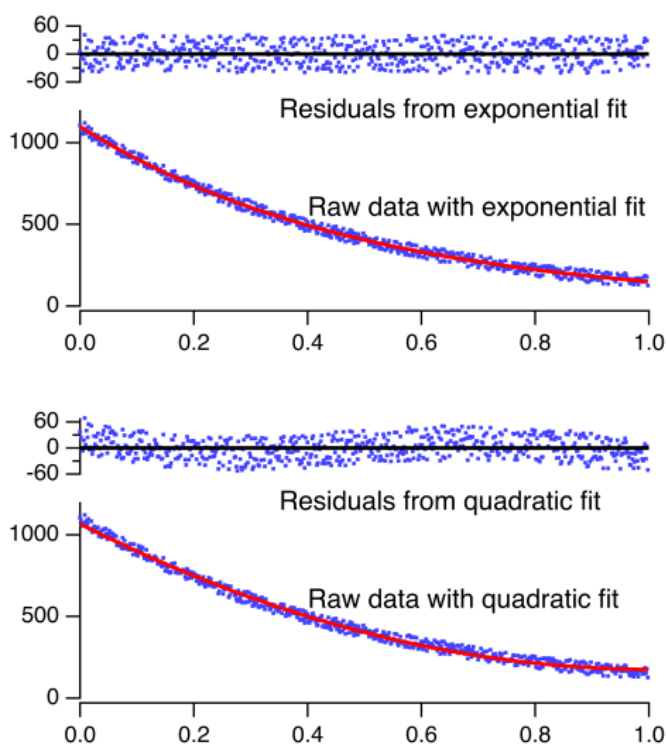
このようなグラフを作成する最も簡単な方法は、Curve Fitting ダイアログの Output Options タブにある Residual ポップアップメニューを使って自動的に生成させることです。

右のグラフは、表示を改善するための若干の微調整を加えつつ、この方法で作成されました。

残差はフィッティングの反復ごとに再計算されます。残差をグラフ上に表示すれば、フィッティングの進行に伴い残差が変化する様子を観察できます。

残差の計算を容易に行い、残差プロットをグラフに追加できるだけでなく、カーブフィッティング処理の一環として、残差生成に使われたウェーブの代入を履歴エリアに出力します。

```
•CurveFit line LineYData /X=LineXData /D /R
  fit_LineYData= W_coef[0]+W_coef[1]*x
  Res_LineYData= LineYData[p] - (W_coef[0]+W_coef[1]*LineXData[p])
  W_coef={-0.037971,2.9298}
  V_chisq= 18.25;V_npnts= 20;V_numNaNs= 0;V_numINFs= 0;
  V_startRow= 0;V_endRow= 19;V_q= 1;V_Rab= -0.879789;
  V_Pr= 0.956769;V_r2= 0.915408;
  W_sigma={0.474,0.21}
  Coefficient values ± one standard deviation
    a      =-0.037971 ± 0.474
    b      =2.9298 ± 0.21
```



この場合、「LineYData」というウェーブが直線モデルでフィッティングされました。

auto trace が宛先 (/D フラグ) と残差 (/R フラグ) の両方に選択されているため、fit_LineYData と Res_LineYData というウェーブが作成されました。

上記の3行目は、残差を計算するための式を示しています。

後で残差を再計算したい場合は、この行をコピーできます。

残差を別の方法で計算したい場合は、この行を編集できます。

「フィッティング後の残差の計算」のセクションを参照してください。

自動トレースを使った残差

Residual メニューから _auto trace_ を選択すると、残差ウェーブが自動的に作成され、Y データが最前面のグラフに表示されている場合、そのグラフに残差ウェーブが追加されます。

自動的に作成される残差ウェーブは、Y データウェーブの名前の先頭に Res_ を付けて命名されます。

結果の名前が長すぎる場合は、切り捨てられます。

その名前を持つウェーブがすでに存在する場合、そのウェーブが再利用され、ウェーブ内の既存の値は上書きされます。

ウェーブが存在しない場合、新規作成されたウェーブは自動的に NaN で埋められます。

残差ウェーブはデータウェーブと同じポイント数で作成され、各データポイントに対して1つの残差値が計算されます。

残差値は、実際にフィッティングに使われたデータポイントについてのみ保存されます。

つまり、データのサブ範囲にフィッティングする場合、あるいはマスクウェーブを使って一部のポイントをフィッティングから除外する場合、残差はフィッティングに使われたデータについてのみ保存され、他のポイントは影響を受けません。

これにより、自動残差オプションを使って、データの一部を順次フィッティングする区分フィッティングから残差を構築することができます。

また、Igor が古い値を残してしまう場合もあることを意味します。

これを防ぐには、残差ウェーブに任意の値を事前に入力してください。

同じデータを再度フィッティングすると、同じ残差ウェーブが使われます。

したがって、以前の結果を保持するには、残差ウェーブの名前の変更が必要です。

これは、Data メニューの Rename、またはコマンドラインの Rename コマンドを使って実行できます。

残差は、上記のような積み重ねグラフ上に表示されます。

これは、Y データを表示するために使われる軸を短縮し、Y データに使われる軸の上に新しい自由軸を配置することによって行われます。

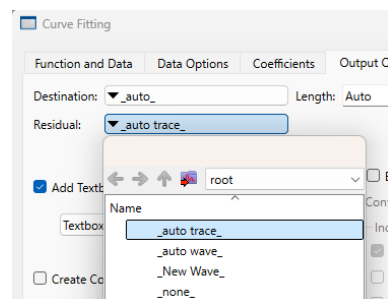
残差プロットは、データ軸を短縮したことで生じた空間を占めます。

生成される軸には、Y データ表示軸に由来する名前が付けられます。

Y データが標準軸に表示されている場合、Res_Left または Res_Right となります。

データ軸が名前付き自由軸の場合、自動生成される軸にはデータ軸の名前に Res_ を接頭辞として付けた名前が与えられます。

Igor は付加された残差プロットを美しく見せるために、Y データトレースの書式設定をコピーし、残差トレースを同じにすることで、どの残差トレースが特定の Y データトレースに対応するかを判別できるように努めます。



残差の軸の軸書式設定は、データトレースに使われる縦軸からコピーされ、残差軸はデータ軸の真上に配置されます。

邪魔になる可能性のある他の縦軸も、スペースを確保するために短縮されます。

自動書式設定は希望通りの結果にならない可能性があります、おそらく十分な精度で出力されるため、Modify Axis および Modify Trace Appearance ダイアログを使って書式設定を微調整できます。

詳細は Graphs (グラフ) のヘルプファイル、特に Creating Graphs With Multiple Axes (複数軸のグラフの作成) と Creating Stacked Plots (積み上げプロットの作成) を参照してください。

残差オートトレースの削除

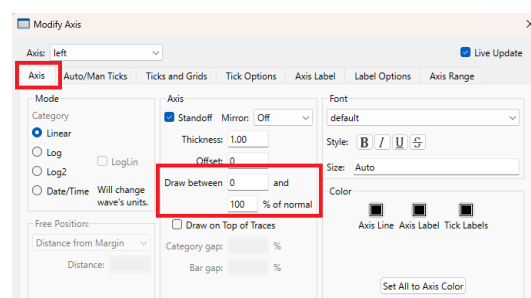
自動トレース残差プロットをグラフに追加すると、元の Y データをプロットするために使われる軸が変更されます。グラフから自動トレース残差を削除すると、残差軸が削除され、ほとんどの場合 Y データ軸は以前の状態に復元されます。

複雑なグラフでは、データ軸の復元が正しく行われない場合があります。

グラフを復元するには、Y データ軸をダブルクリックして Modify Axes ダイアログを表示し、Axis タブを選択します。

「Draw between ... and ... % of normal」というラベルの付いた 2 つの設定があります。

通常、正しい設定は 0 と 100 です。

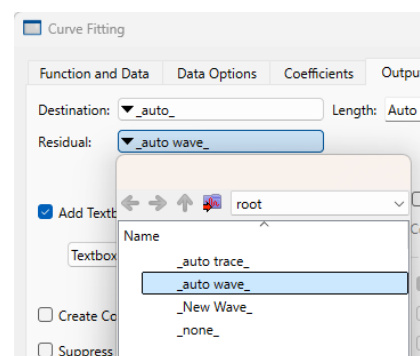


自動ウェーブを用いた残差

グラフの書式設定の変更は大幅なものであるため、自動残差ウェーブを作成して埋めることはしても、グラフに追加しないことをお勧めします。

これを実現するには、Curve Fitting ダイアログの Residual ポップアップメニューから _auto wave_ を選択してください。

カーブフィッティング処理によってウェーブが作成されたら、それをグラフに追加したり、その他の任意の方法で使用したりすることができます。



明示的な残差ウェーブを用いた残差

Residual ポップアップメニューからウェーブを選択すると、そのウェーブが残差の値で埋められます。

この場合、ウェーブは最前面のグラフに追加されません。

この手法を用いることで、書式設定を完全にコントロールしたグラフを作成したり、残差を他の目的に利用したりできます。

Y データウェーブと同じポイント数を持つウェーブのみがメニューに表示されます。

ダイアログでウェーブを作成したくない場合は、まず Data メニューの Duplicate Waves、または Duplicate コマンドを使って Y データウェーブを複製し、適切なウェーブを作成してください。

```
Duplicate yData, residuals_poly3
```

特にデータのサブ範囲にフィッティングする場合、ウェーブを NaN に設定することがしばしば有効です。

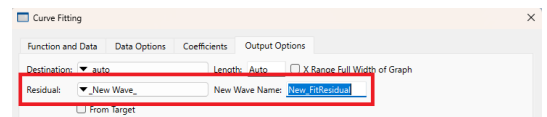
```
residuals_poly3=NaN
```

ウェーブを複製した後、通常はカーブフィッティングに使われる前にグラフや表に追加されます。

新しいウェーブを用いた明示的な残差ウェーブ

明示的な残差ウェーブを作成する最も簡単な方法は、Curve Fitting ダイアログで作成することです。

これを行うには、Residual メニューから `_New Wave_` を選択します。



新しいウェーブの名前を入力できるボックスが表示されます。

ダイアログはその後、カーブフィッティング開始前にウェーブを作成するために必要なコマンドを生成します。

この方法で作成されたウェーブはグラフに追加されません。

フィッティング終了後に、手動で追加する必要があります。

フィッティング後の残差の計算

フィッティング処理中に残差を計算するオーバーヘッドを回避したい場合があるかもしれません。

このセクションでは、自動ウェーブ生成に依存せずにカーブフィッティング後に残差を計算する方法を説明します。

このセクションの最初のほうにあるグラフと同様のグラフは、左自由軸を使って残差ウェーブを追加することで作成できます。

次に、Modify Axes ダイアログの Axis タブで、自由軸の距離を 0 に設定し、軸を通常の 80% から 100% の間で描画するように設定しました。

通常の左軸は 0~70%の範囲で描画されるように設定し、左軸と下軸の両方で軸のオフセット表示を無効化しました。

以下に、これを行うために使う代表的なコマンドを示します。

// サンプルデータの作成

```
Make/N=500 xData, yData
xData = x/500 + gnoise(1/1000)
yData = 100 + 1000*exp(-.005*x) + gnoise(20)
```

// 指数関数フィッティングを自動トレースで実行

```
CurveFit exp yData /X=xData /D
Rename fit_yData, fit_yData_exp
```

// 自動トレースウェーブにおける補間を使って指数残差を計算し、モデル値を取得

```
Duplicate yData, residuals_exp
residuals_exp = yData - fit_yData_exp(xData)
```

```
Untitled
0 // Make sample data
1 *Make/N=500 xData, yData
2 *xData = x/500 + gnoise(1/1000)
3 *yData = 100 + 1000*exp(-.005*x) + gnoise(20)
4
5 // Do exponential fit with auto-trace
6 CurveFit exp yData /X=xData /D
7 Fit converged properly
8 fit_yData= W_coef[0]+W_coef[1]*exp(-W_coef[2]*x)
9 W_coef={100.09, 997.39, 2.5038}
10 W_chisq= 200994; V_npnts= 500; V_numNaNs= 0; V_numINFs= 0;
11 V_startRow= 0; V_endRow= 499;
12 W_sigma={4.51, 3.95, 0.0316}
13 Coefficient values ± one standard deviation
14 y0 = 100.09 ± 4.51
15 A = 997.39 ± 3.95
16 invTau = 2.5038 ± 0.0316
17 *Rename fit_yData, fit_yData_exp
18
19 // Calculate exponential residuals using interpolation in
20 // the auto trace wave to get model values
21 *Duplicate yData, residuals_exp
22 residuals_exp = yData - fit_yData_exp(xData)
23
24 // Do polynomial fit with auto-trace
25 CurveFit poly 3, yData /X=xData /D
26 fit_yData= poly(W_coef, x)
27 W_coef={1049.4, -1835.5, 1003}
28 W_chisq= 318409; V_npnts= 500; V_numNaNs= 0; V_numINFs= 0;
29 V_startRow= 0; V_endRow= 499;
30 W_sigma={3.38, 15.7, 15.2}
31 Coefficient values ± one standard deviation
32 K0 = 1049.4 ± 3.38
33 K1 = -1835.5 ± 15.7
34 K2 = 1003 ± 15.2
35 *Rename fit_yData fit_yData_poly3
36
37 // Find polynomial residuals
38 *Duplicate yData, residuals_poly3
39 residuals_poly3 = yData - fit_yData_poly3(xData)
40
```

// 自動トレース付き多項式フィッティングを実行

```
CurveFit poly 3, yData /X=xData /D  
Rename fit_yData fit_yData_poly3
```

// 多項式の残差を求める

```
Duplicate yData, residuals_poly3  
residuals_poly3 = yData - fit_yData_poly3(xData)
```

自動トレースウェーブでの補間によるモデル値の計算は、十分な精度が得られない場合があります。

代わりに、実際のフィッティング式を用いて正確な値を計算できます。

フィッティングが終了すると、履歴にフィッティング式が出力されます。

少し編集すると、残差計算用のウェーブ代入を作成できます。

上記のフィッティングの履歴からの代入は次の通りです：

```
fit_yData= poly(W_coef,x)  
fit_yData= W_coef[0]+W_coef[1]*exp(-W_coef[2]*x)
```

これらを次のように残差計算に変換できます：

```
residuals_poly3 = yData - poly(W_coef,xData)  
residuals_exp = yData - (W_coef[0]+W_coef[1]*exp(-W_coef[2]*xData))
```

x を xData に置換したのは、x 値をテーブル形式でまとめたためです。

等間隔のデータをフィッティングしていた場合は、テーブル形式の x 値のウェーブは使わず、x をそのまま使っていたでしょう。

残差を計算するこの方法は、明示的な宛先ウェーブを作成して使う場合にも適用できます。

この場合、残差は単にデータと宛先ウェーブの差となります。

例えば、指数関数フィッティングと残差計算を以下のように実行できます。

```
Duplicate yData, yDataExpFit,residuals_exp  
// /D=wave を使った明示的な宛先ウェーブ  
CurveFit exp yData /X=xData /D=yDataExpFit  
residuals_exp = yData - yDataExpFit
```

決定係数または R^2

線形フィッティング ($y = a + bx$) を実行すると、Igor は「決定係数」とも呼ばれる「r-squared」を含む様々な統計情報を出力します。

r-squared の値は、自動的に作成される変数 V_{r2} に格納されます。

R^2 を計算する方法は2つあり、それらは解釈が微妙に異なります。

$$r^2 = 1 - \frac{SS_{res}}{SS_{tot}} \quad (1)$$

$$r^2 = \frac{SS_{reg}}{SS_{tot}} \quad (2)$$

ここで

$$SS_{tot} = \sum_{i=1}^n (y_i - \bar{y})^2 \quad (3 - \text{total sum of squares})$$

$$SS_{res} = \sum_{i=1}^n (y_i - \hat{y}_i)^2 \quad (4 - \text{residual sum of squares})$$

$$SS_{reg} = \sum_{i=1}^n (\hat{y}_i - \bar{y})^2 \quad (5 - \text{regression sum of squares})$$

ここで、 y_i は i 番目の Y データポイント、 $\hat{y}_i$ は i 番目の予測値またはモデル Y 値、 $\bar{y}$ は Y データの平均値です。

直線近似を行う場合、近似直線はデータの平均値を通ることが保証されていて、以下のことが確実です。

$$SS_{tot} = SS_{res} + SS_{reg} \quad (6)$$

従って、式 (1) と式 (2) は同等で、式 (2) は R^2 を「説明された分散」として解釈することを強調しています。つまり、これはデータ分散のうち、回帰直線によって「説明」される、あるいは考慮される部分です。

Igor が出力する R^2 値は、一般的に Excel や他のアプリケーションの値と一致します。

Igor はまた、自動生成変数 V_Pr を介してピアソン相関係数を出力します。

直線近似の場合、 $V_Pr^2 = V_r^2$ となります。

R^2 値と原点を通るフィッティング

原点を通る直線をフィッティングする場合、つまりフィッティング係数 a の値をゼロに設定して固定すると、状況は変わります。

これは通常、常識や理論上、対象量が $X=0$ でゼロでなければならない場合に実施されます。

原点を通る直線近似において、決定係数 (R^2) のようなものを計算する正しい方法について、合意は得られていません。

このような近似は、一般に平均データ値を通らず、式 (6) は成立しないため、式 (1) と式 (2) は同じ値を与えません。

当社としては R^2 を「説明された分散」と解釈することを好むため、原点を通る近似には式 (2) を用います。

これにはいくつかの帰結があります。

1 つは、Igor の R^2 が他のアプリケーション（特に Excel）と必ずしも一致しないことです。

もう 1 つは、原点を通る制約のない同じデータへの直線近似よりも、必ず小さく（つまり「精度が低い」と言われることもある）なります。

より驚くべき結果としては、決定係数が 1 を超える可能性があることです。

これは単純に解釈できます。

回帰直線の分散がデータの分散よりも大きいということです。

原点を通る回帰直線は、おそらく、あなたのデータに対して正当化されないでしょう。

原点を通る回帰直線について、決定係数 (R^2) の解釈に合意が得られておらず不確実なため、このような回帰分析において、決定係数を引用したり、適合度の指標として使ったりすることは推奨できません。

誤差の推定

Igor はカーブフィッティングの各係数について、推定誤差（標準偏差）を自動的に計算します。

カーブフィッティングを実行すると、W_sigma というウェーブが生成されます。

W_sigma の各ポイントは、フィッティングにおける対応する係数の推定誤差に設定されます。

推定誤差も、フィッティングの他の結果とともに履歴エリアに表示されます。

重み付けウェーブを指定しない場合、シグマ値は残差から推定されます。

これは、誤差が平均ゼロで分散が一定な正規分布に従うこと、およびフィッティング関数がデータを適切に記述していることを暗黙的に仮定しています。

係数とそのシグマ値は、同じ基礎データに対して（ただし毎回異なるノイズを加えて）同じフィッティングを無限回繰り返した後に各係数の平均と標準偏差を計算した場合に得られる値の推定値（通常は非常に優れた推定値）です。

信頼帯と係数の信頼区間

データに適合したモデルの不確実性を、グラフに信頼帯または予測帯を追加することで視覚的に表示できます。

これらは、モデルまたは測定データが一定の確率で収まる領域を示す曲線です。

信頼帯はモデルが収まる領域を示し、予測帯はモデルからの無作為標本に無作為誤差を加えた値が収まる領域を示します。

フィッティング係数の信頼区間も計算できます。

信頼区間とは、実際の係数が一定の確率で含まれると推定される範囲です。

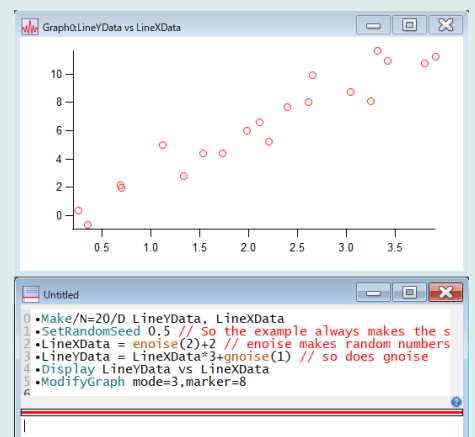
注記

多変量カーブフィッティングでは、信頼帯および予測帯は使用できません。

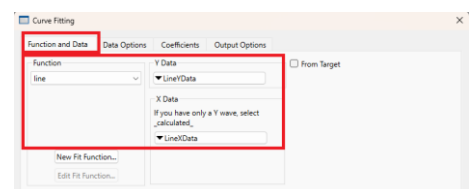
Curve Fitting ダイアログの Output Options タブにある Error Analysis セクションを使って、信頼帯と予測帯の表示、および係数の信頼区間の計算をコントロールします。

1. このヘルプファイルの冒頭にある直線フィッティングの例（A Simple Case -- Fitting to a Built-in Function: Line Fit を参照）を使います。

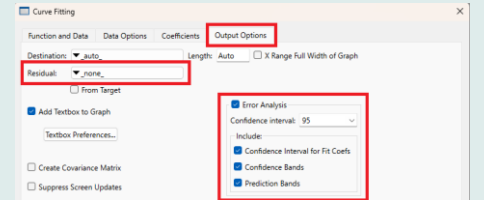
```
Make/N=20/D LineYData, LineXData
SetRandomSeed 0.5
LineXData = enoise(2)+2
LineYData = LineXData*3+gnoise(1)
Display LineYData vs LineXData
ModifyGraph mode=3,marker=8
```



2. Curve Fitting ダイアログの Function and Data タブで、Function を line に、Y Data を LineYData、X Data を LineXData に設定します。



3. Output Options タブで、Residual を `_none_` に、Confidence interval を 95%に設定し、3つの誤差分析オプションすべてを選択します。

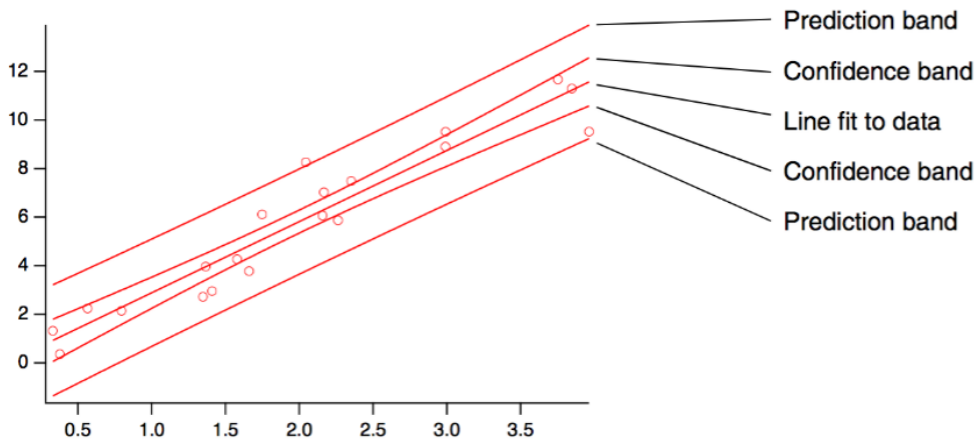
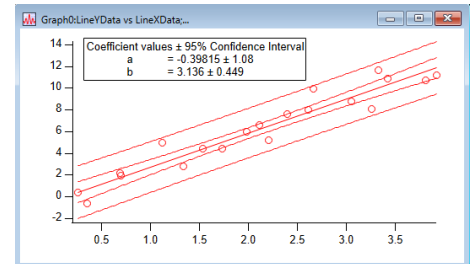


4. Do It をクリックすると、下の出力と右のグラフを生成します。

```

6 •CurveFit/TBOX=768 line LineYData /X=LineXData /D /F={0.95, 7}
7 fit_LineYData= W_coef[0]+W_coef[1]*x
8 W_coef={-0.39815,3.136}
9 V_chisq= 20.3201;V_npnts= 20;V_numNaNs= 0;V_numINFs= 0;
10 V_startRow= 0;V_endRow= 19;V_q= 1;V_Rab= -0.885758;
11 V_Pr= 0.960731;V_r2= 0.923004;
12 W_sigma={0.512,0.213}
13 Fit coefficient confidence intervals at 95.00% confidence level:
14 W_ParamConfidenceInterval={1.08,0.449,0.95}
15 Coefficient values ± 95% Confidence Interval
16 a = -0.39815 ± 1.08
17 b = 3.136 ± 0.449
18

```

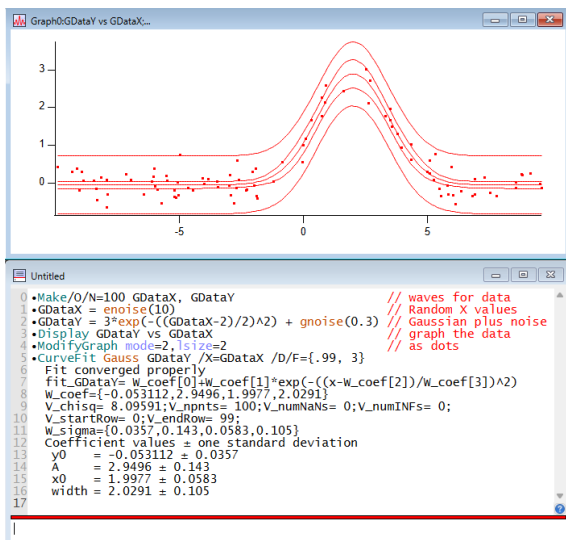


非線形関数でも同様の処理が可能です。非線形関数に対してはあくまで近似であることに注意してください。

```

Make/O/N=100 GDataX, GDataY // データ用のウェーブ
GDataX = enoise(10) // ランダムな x 値
GDataY = 3*exp(-((GDataX-2)/2)^2) + gnoise(0.3) // Gaussian パルスノイズ
Display GDataY vs GDataX // データのグラフ
ModifyGraph mode=2, lsize=2 // ドット表示
CurveFit Gauss GDataY /X=GDataX /D/F={.99, 3}

```



ダイアログでは、信頼帯に対して自動生成されたウェーブのみをサポートしています。

CurveFit および FuncFit コマンドでは、エラーバー形式の表示を含むその他のオプションを複数サポートしています。

詳細は CurveFit コマンドのヘルプを参照してください。

フィッティング後の信頼区間の計算

フィッティング終了後、W_sigma ウェーブの値を使ってフィッティング係数の信頼区間を計算できます。

これには StudentT 関数を使います。

カーブフィッティング後に履歴に記録される情報は、例えば次のようなものです。

```
Fit converged properly
fit_junk= f(coeffs,x)
coeffs={4.3039,1.9014}
V_chisq= 101695; V_npnts= 128; V_numNaNs= 0; V_numINFs= 0;
W_sigma={4.99,0.0679}
```

フィッティング係数の 95 パーセント信頼区間を計算し、その値を別のウェーブに保存するには、以下の行を実行します。

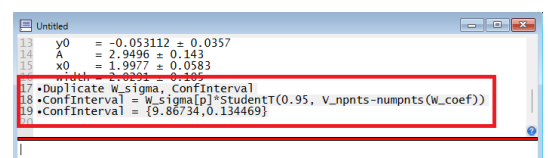
```
Duplicate W_sigma, ConfInterval
ConfInterval = W_sigma[p]*StudentT(0.95, V_npnts-numpts(W_coef))
```

もちろん、V_npnts-numpts(W_coef) の代わりに単に 126 と入力することもできますが、記述されたままの行は、どのフィッティングに対しても変更なしで動作します。

例でフィッティングを行った後にこれを実行した時の結果は次のようになりました。

```
ConfInterval = {9.86734,0.134469}
```

明らかに、W_coef[0] はゼロと有意な差がありません。



信頼帯ウェーブ

これらのオプションを選択した場合、カーブフィッティングにより、信頼帯と予測帯を表示するために必要な値を含む新しいウェーブが生成されます。

ウェーブの数と名前は、選択されたオプションと表示スタイルによって異なります。

上記の図のようなコンター帯の場合、ウェーブは2つ存在します：上側コンター用と下側コンター用です。

エラーバーを表示するにはウェーブは1つだけで十分です。

詳細は CurveFit コマンドのヘルプを参照してください。

いくつかの統計

信頼帯と予測帯の計算には、いくつかの統計的な仮定が伴います。

まず第一に、測定誤差が正規分布に従うと仮定されます。

正規性からの逸脱が実際の問題を引き起こすには、通常、かなり顕著なものでなければなりません。

重み付けウェーブを指定しない場合、誤差の分布は残差から推定されます。

この推定では、分布は正規分布であるだけでなく、平均がゼロの均一分布であると仮定されます。

つまり、誤差の分布はモデルを中心とし、誤差の標準偏差は独立変数のすべての値で同じです。

平均がゼロであるという仮定は、モデルが正しいことを要求します。つまり、測定データが真にモデルとランダムな正規誤差の和を表していると仮定します。

一部のデータセットは、誤差が均一であるという仮定では十分に特徴付けられません。

その場合、重み付けウェーブを指定することで誤差分布を明示する必要があります（「重み付け」のセクションを参照）。

この操作を行うと、誤差推定値がフィッティング係数の不確かさを決定するために使われ、したがって信頼帯の計算にも使われます。

信頼帯は、モデル係数と係数の推定不確かさにのみ依存し、重み付けウェーブによって提供される誤差推定値を常に考慮して計算されます。

一方、予測帯は、各ポイントにおける測定誤差の分布にも依存します。

ただし、これらの誤差は考慮されず、残差から推定された均一な測定誤差のみが使われます。

信頼帯および予測帯の計算は、予測モデル値の分散の推定値に基づいています。

$$V(\hat{Y}) = a^T C a$$
$$a = \delta F / \delta p|_x$$

ここで、 $\hat{Y}$ は独立変数 X の与えられた値におけるモデルの予測値、 a は独立変数の与えられた値で評価された係数に対するモデルの偏微分のベクトル、 C は共分散行列です。

$a^T C a$ 項が標本分散 σ^2 と乗算される例をよく見かけますが、これは共分散行列に含まれています。

信頼区間と予測区間は次のように計算されます

$$CI = t(n-p, 1-\alpha/2)[V(\hat{Y})]^{1/2} \quad \text{と} \quad PI = t(n-p, 1-\alpha/2)[\sigma^2 + V(\hat{Y})]^{1/2}$$

これらの式によって計算される量は、区間の大きさです。

これらはエラーバーに使われる値です。

これらの値は帯をコンターとして表示するために使われるウェーブを生成するために、モデル値 ($\hat{Y}$) に加算されます。

関数 $t(n-p, 1-\alpha/2)$ は、確率 $1-\alpha/2$ を持つ学生 t 分布上のポイントで、 σ^2 は標本分散です。予測区間の計算において、 σ^2 に用いる値は残差から推定された均一な値です。

これは、非均一な値を持つ重み付けウェーブを指定した場合、独立変数の任意の値に対する標本分散の正しい値に関する情報が存在しないため、正しくありません。

StudentT 関数を使って正しい予測区間を計算できます。

フィッティング関数のフィッティング係数に対する導関数の値が必要となります。

関数を微分して導関数を提供する別の関数を記述するか、数値近似を使用できます。

Igor は数値近似を使います。

信頼帯と非線形関数

厳密に言うと、上記の議論と式は、フィッティング係数に関して線形な関数に対してのみ正しいです。

その場合、ベクトル a は単に基底関数のベクトルです。

多項式フィッティングの場合、これは 1 、 x 、 x^2 などを意味します。

フィッティング関数が非線形の場合、この式はテイラー展開の線形項のみを使った近似から導かれます。

したがって、信頼帯と予測帯はあくまで近似値です。

この近似がどれほど不正確かは、関数に大きく依存するため断言できません。

共分散行列

Curve Fitting ダイアログの Output Options タブで Create Covariance Matrix チェックボックスを選択すると、カーブフィッティング係数の共分散行列が生成されます。

これは直線フィッティングを除く、すべてのフィッティングで利用可能です。

直線フィッティングでは特別な出力変数 V_{rab} が生成され、傾きと Y 切片の相関係数が得られます。

デフォルトでは (Curve Fitting ダイアログを使っている場合)、 N 行 N 列の行列ウェーブを生成します。

ここで N は係数の数です。

ウェーブの名前は M_Covar です。

このウェーブは行列演算で使用できます。

コマンドラインまたはユーザープロシージャから CurveFit、FuncFit、FuncFitMD 演算を使う場合は、 $/M=2$ フラグを使って行列ウェーブを生成してください。

もともと、カーブフィッティングでは各フィッティング係数ごとに 1 次元のウェーブが生成されていました。

これらのウェーブをすべて組み合わせることで共分散行列が構成されていました。

互換性のため、 $/M=1$ フラグを指定すると、現在も複数の 1 次元ウェーブ (W_Covarn という名前) が生成されます。

意図的にこの操作は行わないでください。

行列 $M_Covar[i][i]$ の対角要素は、パラメーター i の分散です。

分散はシグマの二乗であり、そのパラメーターの推定誤差の標準偏差です。

共分散行列の詳細な説明は、「Numerical Recipes in C 第 2 版」、685 ページおよび 15.5 節に記載されています。

相関行列

カーブフィッティング中に生成された共分散行列から相関行列を計算するには、以下のコマンドを使います。

```
Duplicate M_Covar, CorMat // CorMat の代わりに任意の名前を使用できます
CorMat = M_Covar[p][q]/sqrt(M_Covar[p][p]*M_Covar[q][q])
```

相関行列は共分散行列の正規化された形式です。

各要素は、2つのフィッティング係数間の相関を -1 から 1 までの数値として示します。

対応する要素が 1 の場合、2つの係数間の相関は完全です。

要素が -1 の場合、完全な逆相関であり、0 の場合、相関は存在しません。

相関行列の要素が 1 または -1 に非常に近い場合、そのフィッティングには注意が必要です。

これは「識別可能性」の問題を示している可能性があります。

つまり、フィッティングが2つのパラメーターを十分に区別できておらず、制約が不十分であることを意味します。この問題を回避するため、既存のパラメーターを組み合わせる新たなパラメーターでフィッティングを再記述できる場合があります。

識別可能性の問題

相関行列の対角線要素以外の値が 1 または -1 に近いことに加え、識別可能性の問題の兆候としては、収束に膨大な反復回数を要するフィッティングや、推定係数の誤差 (W_sigma ウェーブ) が不当に大きいフィッティングが挙げられます。

「識別可能性の問題」という表現は、2つ以上のフィッティング係数がトレードオフの関係にあるため、両方の値を同時に求めることがほぼ不可能となる状況を指します。

これらの係数は相関しており、一方のフィッティング係数を調整すると、ほぼ同等のフィッティング精度を得るための他方のフィッティング係数の値が見つかります。

相関が強すぎると、フィッティングアルゴリズムは進むべき方向を見失い、広範囲の係数値がほぼ同等に良好に見える係数空間を彷徨います。

つまり、カイ二乗空間の広い領域ではカイ二乗値の変動が極めて小さくなります。

通常の結果として、収束は見せかけるものの W_sigma の推定値が過大になるか、特異行列エラーが発生します。

誤差推定値が不当に大きい場合や、収束までに多くの反復を要する場合、相関行列を計算し、対角線から離れた値が 1 または -1 に近いものを探してください。

経験上、0.9 前後の値はおそらく問題ありません。

0.99 に近い値は疑わしいですが許容範囲内です。

0.999 前後の値はほぼ確実に問題の兆候です。

残念ながら、識別可能性の問題については、できることはほとんどありません。

これは、あなたのフィッティング関数の数学的特性です。

モデルによっては、係数空間において、2つの係数がフィッティングに類似した影響を与える領域が存在する場合があります。独立変数の範囲を拡大することで問題を緩和できることがあります。

また、係数がコントロールする特徴が非常に狭い範囲に限定されている場合もあり、サンプリング密度を高めることで問題を解決できることがあります。

制約付きフィッティング

係数の値をフィッティング関数に制限することが望ましい場合があります。

フィッティング関数は、数学的には問題ないものの物理的に不合理な係数値を許容することがあります。

また、係数値の特定の範囲が、関数値の特異点やコンピューター表現のオーバーフローを引き起こすほど大きな関数値といった数学的問題を引き起こす場合もあります。

このような場合、解を問題領域から外すために制約を適用することがしばしば望ましいことになります。

最終的な解にアクティブな制約が含まれない可能性もありますが、制約は解への到達過程で不良領域に迷い込んだことによるエラーでフィッティングが終了するのを防ぐ役割を果たします。

カーブフィッティングでは、フィッティング係数の任意の線形結合の値に対する制約をサポートします。

Curve Fitting ダイアログでは、個々の係数の値に対する制約をサポートします。

制約を適用するアルゴリズムはかなり寛容です。

初期推定値は制約領域内にある必要はありません（つまり、初期推定値は制約に違反しても構いません）。

ほとんどの場合、アルゴリズムは単にパラメーターを制約領域の境界上に移動させ、カーブフィッティングを続行します。

制約が矛盾している場合（カーブフィッティングの専門用語で「実現不可能」と呼ばれる状態）であっても、違反が深刻でなければ問題ありません。

フィッティングは単に「差を分割」し、実現不可能な制約間の妥協点となる係数を提供します。

組み込みの線形、多項式、および多項式 2D フィッティング関数には制約が利用できません。

これらのフィッティング関数に制約を適用するには、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。

Curve Fitting ダイアログを使った制約

「例 2 — ユーザー定義の簡易 2 次元ガウスフィッティング」のデータを使って説明します。

Curve Fitting ダイアログの Coefficients タブには、制約付きフィッティングを有効にするメニューが含まれています。

From Coefficient List を選択すると、係数リストの制約セクションが利用可能になります。

下限制約列に値を入力すると、対応する係数をその値より大きい値に制限するコマンドが生成されます。

上限制約の列に値を入力すると、対応する係数をその値より小さい値に制限します。

空欄のままにした場合は、いかなる制約も生成されません。

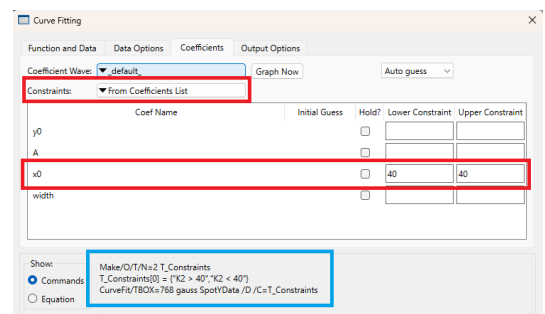
上記の数値は gauss 関数を選択して作成しました。

以下のコマンドがダイアログによって生成されます。

```
Make/O/T/N=2 T_Constraints
```

```
T_Constraints[0] = {"K2 > 40", "K2 < 40"}
```

```
CurveFit/TBOX=768 gauss SpotYData /D /C=T_Constraints
```



より複雑な制約も設定可能ですが、Curve Fitting ダイアログでは入力できません。
ダイアログを入力する前に、制約ウェーブを作成しておく必要があります。
その後、Constraints メニューからそのウェーブを選択してください。
制約ウェーブの構築方法については、以下のセクションをご覧ください。

制約ウェーブを使った複雑な制約

係数の線形結合の値を制約することは可能ですが、Curve Fitting ダイアログでは単純な制約のみをサポートしています。

Curve Fitting ダイアログに入る前に、制約付きの適切なテキストウェーブを作成できます。
Coefficients タブの Constraints メニューからウェーブを選択します。
コマンドラインで制約付きウェーブと共に CurveFit または FuncFit コマンドを使うこともできます。

テキストウェーブの各要素は、1つの制約式を保持します。
テキストウェーブを使うと、テーブル内の式を簡単に編集できます。
そうでない場合は、上記の例で2行目に示されているようなコマンドラインを使う必要があります。

制約式

制約式は任意に複雑にすることができ、フィッティング係数のいずれか、あるいはすべてを含むことができます。
各式には不等号（「<」、「<=」、「>」、「>=」）が含まれている必要があります。
式では、記号 Kn (K0、K1 など) が n 番目のフィッティング係数を表すために使われます。
これは Kn システム変数に似ていますが、制約式内では単なる記号的なプレースホルダーに過ぎません。
式には、Kn の任意の組み合わせの和、および Kn を乗算または除算する因子が含まれる場合があります。
因子は、Kn を一切含まない限り、非線形であっても任意に複雑であっても構いません。
Kn は関数呼び出しで使用できず、非線形式に関与することもできません。
以下に有効な制約式を示します。

```
K0 > 5
K1+K2 < numVar^2+2           // numVar はグローバル数値変数
K0/5 < 2*K1
(numVar+3)*K3 > K1+K2/(numVar-2)
log(numVar)*K3 > 5           // 非線形因子は K3 を含まない
```

以下は正しくありません。

```
K0*K1 > 5                   // K0*K1 は非線形
1/K1 < 4                   // これは非線形: K1 による除算
ln(K0) < 1                 // 関数のパラメーターとして K0 は使用不可
```

制約式が解析される時、K 値を乗算または除算する因子はリテラル文字列として抽出され、個別に評価されます。
したがって、<expression>*K0 または K0/<expression> という形式の場合、<expression> は単独で実行可能でなければなりません。

スレッドセーフな関数からのフィッティングには、制約式を含むテキストウェーブを使用できません。
「制約行列とベクトル」のセクションに記載されている方法を使う必要があります。

等式制約

フィッティング係数の値を特定の値に等しく制限したい場合があります。

制約アルゴリズムには等式制約の機能がありません。

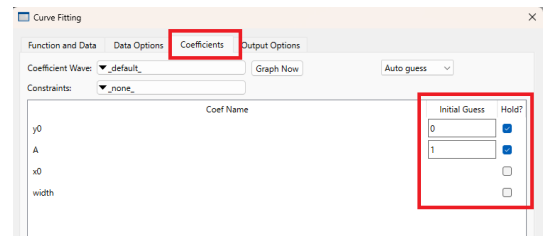
これを擬似的に実現する一つの方法は、フィッティング係数が特定の値より大きくかつ小さいことを要求する二つの制約を使うことです。

例えば、「 $K1 > 5$ 」と「 $K1 < 5$ 」は $K1$ が 5 に等しくなることを要求します。

1つのパラメーターを特定の値に固定する場合、この方法は最適ではありません。

パラメーターをホールドする方がはるかに効果的です。

Curve Fitting ダイアログでは、Coefficients タブの係数リストにある Hold ボックスを選択し、初期推定値列に値を入力してください。



コマンドラインでフィッティングを行う場合、

FuncFit/H=「01」...

と指定すると $K1$ を特定の値に固定します。

その値はフィッティングを開始する前に設定する必要があることに注意してください。

制約付きフィッティングの例

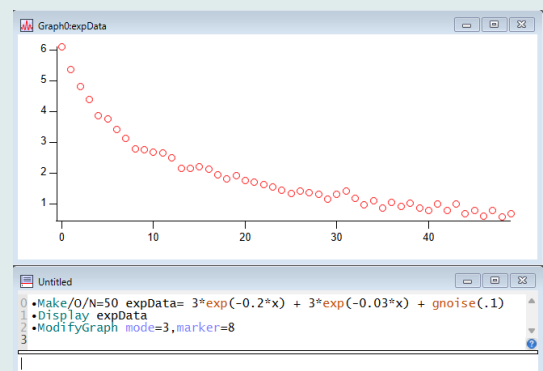
Igor ヘルプファイル内のサンプルコマンドは、該当する行を選択し Ctrl+Enter を押すことで実行できます。

この例は、2つの指数関数の和に適合させつつ、理論的知見によって課される可能性のある限界値以下に指数関数の振幅の和を制約するものです。

制約条件が複雑すぎて Curve Fitting ダイアログに入力できないため、コマンドラインを使います。

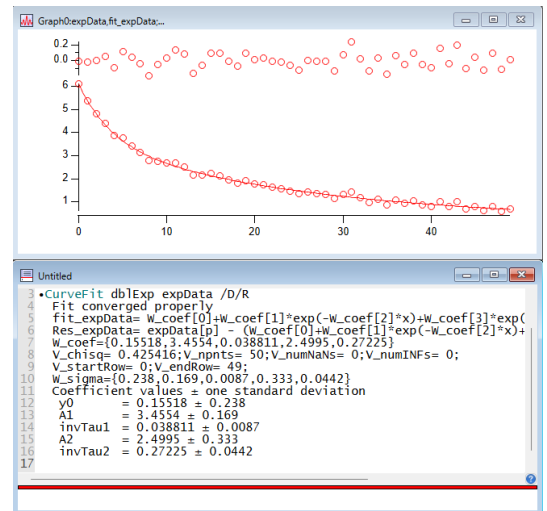
1. まず、データを作成し、グラフ化します。

```
Make/O/N=50 expData= 3*exp(-0.2*x) + 3*exp(-0.03*x) + gnoise(.1)
Display expData
ModifyGraph mode=3,marker=8
```



2. 制約なしでフィッティングを行います。

```
CurveFit dblExp expData /D/R
```



3. 次のコマンドは、個々の指数振幅の和に対する制限を実装する文字列「K1 + K3 < 5」を含む単一要素のテキストウェーブを生成します。

```
Make/O/T CTextWave={"K1 + K3 < 5"}
```

このヘルプファイルに記述できるよう、コマンドを使ってウェーブを作成します。

ウェーブの作成には Data メニューの Make Waves を使い、その後テーブルでウェーブを表示して式を編集する方が簡単かもしれません。

必ずテキストウェーブを作成してください。

ウェーブ内に空白行を残さないでください。

制約付きで再度フィッティングを実行します。

```
CurveFit dblExp expData /D/R/C=CtextWave
```

この場合、差はわずかとなります。

制約付きフィッティングのグラフでは、フィッティング線が標準カーブフィッティングと比較して、左端でわずかに低く、右端でわずかに高いことに注目してください。

この差は両端の残差値に反映されています。

制約付きカーブフィッティングの出力には、制約が使われた事実と、その制約が解において有効であったことを示す以下の行が含まれます。

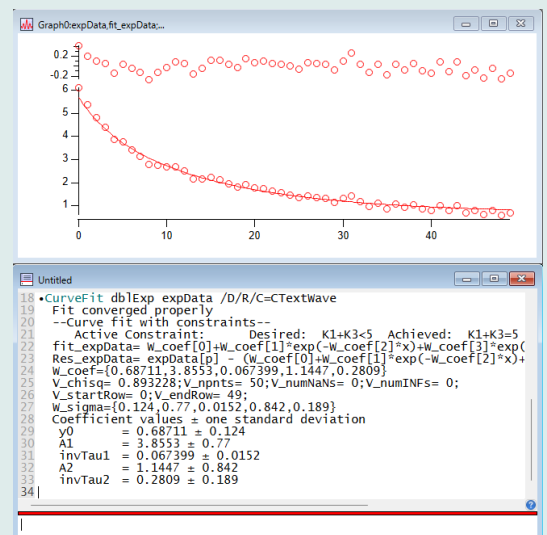
```
--Curve fit with constraints--
```

```
Active Constraint: Desired: K1+K3<5 Achieved: K1+K3=5
```

ほとんどの場合、これに近いメッセージが表示されます。

制約条件に矛盾がある場合、1つ以上の制約が違反する可能性があります。

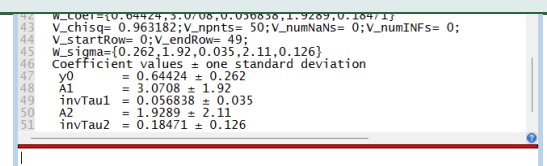
その場合、その事実に関するレポートが提供されます。



4. 以下のコマンドは、例にさらに 2 つの制約を追加します。

```
Make/O/T CTextWave={"K1 + K3 < 5", "K1 > 3.3",
"K3 > 2.2"}
```

```
CurveFit dblExp expData /D/R/C=CtextWave
```



新しい制約では、個々の振幅の値の合計が 5 より大きくなることを要求しつつ、同時に合計が 5 未満であることも要求します（したがって、これらは「実現不可能な制約」です）。

5. ほとんどの場合、テーブル内の制約ウェーブを編集して新しい制約を追加することになります。

Point	CTextWave
0	K1 + K3 < 5
1	K1 > 3.3
2	K3 > 2.2
3	

制約行列と制約ベクトル

制約付きフィッティングを行うと、制約式を解析し、制約を記述する行列とベクトルを構築します。

各制約式は解析され、次のような単純な式を形成します。

$$C_0K_0 + C_1K_1 + \dots \leq D$$

ここで、 K_i はフィッティング係数、 C_i と D は定数です。

制約は行列演算 $CK \leq D$ で表現でき、 C は定数の行列、 K はフィッティング係数ベクトル、 D は限界値のベクトルです。

この行列の次元は $N \times M$ で、 N は制約式の数を、 M はフィッティング係数の数を表します。

ほとんどの場合、行列の要素のほぼ全てはゼロです。

前の例では、 C 行列と D ベクトルは次の通りです。

C matrix	D vector
0 1 0 1 0	5
0 -1 0 0 0	-3.3
0 0 0 -1 0	-2.2

これはカーブフィッティングコードが内部で使う形式です。

必要に応じて、 C および D 定数を含む行列ウェーブおよび 1 次元ウェーブを構築できます。

制約付きフィッティングを要求するために `/C=textwave` を使う代わりに、`/C={matrix, 1Dwave}` を使ってください。

一般的に、与えられた制約条件に対して適切なウェーブを作成することは混乱を招き、エラーが発生しやすくなります。

ただし、スレッドセーフ関数からのカーブフィッティングに `textwave` メソッドを使うことは許可されていません（「制約とスレッドセーフ関数」のセクションを参照）。

幸い、Igor はテキストウェーブ内の式を解析する時に必要なウェーブを生成できます。

/C フラグを使って C 行列と D ベクトルを含むウェーブを作成します（このフラグは CurveFit コマンド名の直後に指定し、末尾に置かないことに注意してください）。

上記の例を実行済みであれば、以下のコマンドを実行してウェーブを構築し、テーブルに表示できます。

```
CurveFit/C dblExp expData /D/R/C=CTextWave
Edit M_FitConstraint, W_FitConstraint
```

C 行列は M_FitConstraint と命名され、D ベクトルは W_FitConstraint と命名されます（慣例により、行列名は「M_」で始まり、1 次元ウェーブ名は「W_」で始まります）。

The screenshot shows two windows. The top window, titled 'Table0:M_FitConstraint,W_FitConstraint', displays a table with 5 rows and 5 columns. The bottom window, titled 'Untitled', shows the command window with the following text:

```

53 •CurveFit/C dblExp expData /D/R/C=CTextWave
54 Fit converged properly
55 --Curve fit with constraints--
56 Constraint VIOLATED: Desired: K1>3.3 Achieved: K1=3.07075
57 Constraint VIOLATED: Desired: K3>2.2 Achieved: K3=1.9289
58 fit_expData= w_coef[0]+w_coef[1]*exp(-w_coef[2]*x)+w_coef[3]*exp(-w_
59 Res_expData= expData[p] - (w_coef[0]+w_coef[1]*exp(-w_coef[2]*x)+w_c
60 W_coef=[0.64424, 3.0708, 0.056838, 1.9289, 0.18471]
61 V_chisq= 0.963182; V_npts= 50; V_numNaNs= 0; V_numINFs= 0;
62 V_startRow= 0; V_endRow= 49;
63 W_sigma=[0.262, 1.92, 0.035, 2.11, 0.126]
64 Coefficient values ± one standard deviation
65 y0 = 0.64424 ± 0.262
66 A1 = 3.0708 ± 1.92
67 invTau1 = 0.056838 ± 0.035
68 A2 = 1.9289 ± 2.11
69 invTau2 = 0.18471 ± 0.126
70 •Edit M_FitConstraint, W_FitConstraint
71

```

スレッドセーフ関数における制約条件の指定に役立つことに加え、これらのウェーブは後で制約条件を確認するために使用できます。

次のコマンドは、生成されたフィッティング係数ウェーブ（W_coefs）に制約行列を乗算し、結果を以前に作成したテーブルに追加します。

```
MatrixMultiply M_FitConstraint, W_coef
AppendToTable M_Product
```

MatrixMultiply コマンドの結果は行列ウェーブ M_Product です。

制約が不可能なため制約違反が生じた結果、M_Product[1] と M_Product[2] の値が W_FitConstraint の対応する値よりも大きくなっていることに注意してください。

The screenshot shows a table with 5 rows and 5 columns. The first three columns are M_FitConstraint, and the last two are W_FitConstraint and M_product. The values in the M_product column are 0, 5, -3.3, -2.2, and -1.9289. The values in the W_FitConstraint column are 0, 3.07075, 1.9289, 0.18471, and 0.126. The values in the M_FitConstraint column are 0, 1, 2, 3, and 4. The values in the W_coef column are 0.64424, 3.0708, 0.056838, 1.9289, and 0.18471.

制約付きカーブフィッティングにおける落とし穴

制約境界における不正な値

制約付きカーブフィッティングにおける最も可能性の高い問題は、カーブフィッティング中に特異行列エラーを生成する制約境界上の値です。

例えば、多くのアプリケーションでは、指数関数前の乗数が正であること（この場合 $K_1 > 0$ ）を要求することが合理的です。

$$y = K_0 + K_1 e^{K_2 x}$$

制約式「 $K_1 > 0$ 」と記述するのは自然ですが、これは問題を引き起こす可能性があります。

反復処理で K_1 の負の値が試行されると、制約により K_1 は厳密にゼロに設定されます。

しかし K_1 がゼロの場合、関数は K_2 に依存せず、特異行列エラーが発生します。

解決策は、 K_1 が小さな正の数より大きくなることを要求する制約を使うことです。

例えば「 $K_1 > 0.1$ 」などです。

この限界値は、アプリケーションに合わせて調整する必要があります。

解が見つかった時点で制約が非活性であると予想しているのに、フィッティング結果で制約が活性だったと報告される場合は、限界値を下げ、フィッティングを再度実行できます。

深刻な制約の衝突

制約を適用する手法は、制約同士が矛盾する場合（実現不可能）でも解を見つけることができますが、矛盾が深刻すぎて手法が失敗する可能性があります。

この場合、特異行列エラーが発生します。

制約領域のフィッティングが悪い

制約条件によって許容される係数空間の領域がデータに著しく適合しない場合、特異行列エラーが生じる可能性があります。

初期推定値が制約領域から大きく外れている

通常、フィッティングの初期推定値が制約条件で許容される領域外にある場合、フィッティング係数は最初の反復で制約領域内に移行します。

ただし、初期推定値が制約領域からあまりにも離れている場合、制約条件の解が失敗する可能性があります。

これにより、通常の特異行列エラーが発生します。

保持されたパラメーターとの制約の競合

同じパラメーターに対してホールドと制約を同時に適用することはできません。

したがって、以下は許可されません。

```
Make/T CWave="K1 > 5"
```

```
FuncFit/H="01" myFunc, myCoefs, myData /C=CWave
```

カーブフィッティングにおける NaN と INF

カーブフィッティングは入力データ内の NaN と INF を無視します。

これは個々のデータ値をフィッティングから除外する便利な方法です。

より良い方法はデータマスクウェーブを使うことです（「マスクウェーブを使う」のセクションを参照）。

カーブフィッティング用の特殊変数

カーブフィッティングの入力（フィッティングの追加コントロール）と出力（追加の統計情報の提供）には、いくつかの特別な変数が使われます。

知識のあるユーザーは、入力変数を使ってフィッティングプロセスを微調整することができます。

ただし、通常、これは必要ありません。

一部の出力変数は、統計に精通したユーザーがカーブフィッティングの品質を評価するのに役立ちます。

入力変数を対話的に使うには、フィッティングを実行する前に、コマンドラインから Variable コマンドを使って作成してください。

出力変数のほとんどは、CurveFit または FuncFit コマンドによって自動的に作成されます。

以下に示すように、自動作成されないものもあります。

それらの変数が提供する情報を必要とする場合は、自身で作成する必要があります。

特に指定がない限り、出力変数は存在しない場合に作成されます。

それらのうちいくつかは履歴に出力されます。

プロシージャでカーブフィッティングを実行する場合、入力変数と出力変数の両方をローカル変数またはグローバル変数にすることができます。

通常、ローカル変数にすることをお勧めします。

ローカル変数の使用方法については、ヘルプ [Accessing Variables Used by Igor Operations](#) を参照してください。

プロシージャでは、出力専用の変数は常にローカル変数として作成されます。

コマンドラインまたは [Curve Fitting](#) ダイアログを使って対話的にカーブフィッティングを実行する場合、変数はグローバルになります。

複数のデータフォルダー（ヘルプ [Data Folders](#) で説明）を使う場合、入力変数と出力変数は現在のデータフォルダー内で検索または作成されることに留意する必要があります。

次の表は、すべての入力および出力の特殊変数を一覧表示しています。

それらのいくつかについては、以下でさらに詳しく説明します。

<u>変数</u>	<u>I/O</u>	<u>意味</u>
V_FitOptions	入力	カーブフィッティングのその他のオプションです。
V_FitTol	入力	通常、反復フィッティングは、反復ごとのカイ二乗値の減少率が 0.001 未満になると終了します。 V_FitTol というグローバル変数を作成し、1E-10 から 0.1 の間の値に設定すると、その値が終了許容誤差として使われます。 この範囲外の値は効果がありません。
V_tol	入力	（多項式フィッティングのみ）「特異値閾値」です。 詳細は下記の「多項式フィッティングに関する特記事項」のセクションを参照。
V_chisq	出力	フィッティングの度合いの尺度です。 各データポイントの標準誤差の逆数を含む重み付けウェーブを指定した場合にのみ絶対的な意味を持ちます。
V_q	出力	（線形フィッティングのみ）カイ二乗の信頼性の尺度です。 重み付けウェーブを指定した場合にのみ有効です。
V_siga, V_sigb	出力	（線形フィッティングのみ）直線近似 ($y = a + bx$) における切片係数 ($K0 = a$) と傾き係数 ($K1 = b$) の推定の不確かさです。
V_Rab	出力	（線形フィッティングのみ）a（切片、K0）の不確かさと b（傾き、K1）の不確かさとの間の相関係数です。
V_Pr	出力	（線形フィッティングのみ）線形相関係数 r（ピアソンの r と呼ばれる）です。 値が +1 または -1 の場合は完全相関を示し、値が 0 に近い場合は相関がないことを示します。
V_r2	出力	（線形フィッティングのみ）決定係数、通常は単に「R ² 」と呼ばれます。 詳細は「決定係数または R ² 」のセクションを参照してください。
V_npnts	出力	フィッティングされたポイントの数です。 重み付けウェーブを指定した場合、重み付けがゼロのポイントはこのカウント

に含まれません。

また、値が NaN または INF であるポイントも含まれません。

V_nterms	出力	フィッティングにおける係数の数です。
V_nheld	出力	フィッティング中に固定された係数の数です。
V_numNaNs	出力	フィッティング対象データ内の NaN 値の数です。 カーブフィッティング中は NaN は無視されます。
V_numINFs	出力	フィッティング対象データ内の INF 値の数です。 カーブフィッティング中は INF は無視されます。
V_FitError	I/O	フィッティング中のエラーからの回復を試みるために、プロシージャから使われます。
V_FitQuitReason	出力	非線形フィッティングが反復処理を停止した理由に関する追加情報を提供します。 注記： この変数は作成する必要があります。自動的に作成されません。
V_FitIterStart	出力	V_FitIterStart の使用は廃止されました。 代わりに一括フィッティング関数を使ってください。 詳細は「一括フィッティング関数」のセクションを参照してください。 反復処理が開始されたときに 1 に設定されます。 特定の反復処理において、ユーザー定義のフィッティング関数が初めて呼び出された時点を識別します。 注記： この変数は作成する必要があります。自動的に作成されません。
V_FitMaxIters	入力	フィッティングを停止する前に収束しない場合の最大パス数をコントロールします。 デフォルトは 40 です。 V_FitMaxIters は 0 より大きい任意の値に設定できます。 V_FitMaxIters が 1 未満の場合、デフォルトの 40 が使われます。
V_FitNumIters	出力	反復回数です。 注記： この変数は作成する必要があります。自動的に作成されません。
S_Info	出力	フィッティングに関する特定の情報を提供するキーと値のペアです。 注記： この変数は作成する必要があります。自動的に作成されません。

V_FitOptions

V_FitOptions という変数を作成し、その中の様々なビットを設定することで、フィッティングプロセスに対して呼び出せるオプションがいくつかあります。

V_FitOptions を 1 に設定するとビット 0 が設定され、2 に設定するとビット 1 が設定されます。

以下同様です。

ビット 0: オートトレースウェーブの X 方向スケーリングをコントロールする

V_FitOptions が存在し、ビット 0 が設定されている場合（変数 V_fitOptions=1）、かつ Y データウェーブが上段グラフにある場合、自動トレース先ウェーブの X 軸スケーリングは、グラフ上の適切な X 軸に一致するように設定されます。

これは、フィッティング対象の X データ範囲外で曲線を外挿したい場合に役立ちます。

これを行うより良い方法は、/X フラグを使うことです（パラメーターではなく、このフラグは CurveFit または FuncFit コマンドの直後、fit 関数名の前に指定します）。

詳細はコマンド CurveFit のヘルプを参照してください。

ビット 1: ロバストなフィッティング

外れ値を軽視する傾向がある偏差の二乗ではなく、絶対偏差の和を最小化するロバストなフィッティング手法が得られます。

これを行うには、V_FitOptions を作成し、ビット 1 を設定します（変数 V_fitOptions=2）。

警告： 推定誤差や相関行列の結果に対して、調整を試みることは一切行っていません。ご自身で対応してください。

警告： このビットを設定したら、そのまま放置しないでください。

警告： ビット 1 の設定は、直線、ポリゴン、またはポリゴン 2D のフィッティングには影響しません。

ビット 2: カーブフィッティング進行ウィンドウを非表示にする

通常、反復フィッティング中は進行状況を示すウィンドウが表示されます。

このウィンドウを表示したくない場合は、V_FitOptions を作成し、ビット 2 を設定してください（変数 V_fitOptions=4）。

大量のデータセットに対してバッチ処理でフィッティングを行う場合、これにより処理速度が若干向上する可能性があります。

これを行うより良い方法は、/W=2 フラグを使うことです。

詳細はコマンド CurveFit のヘルプを参照してください。

ビット 3: 反復を保存

カーブフィッティングが解に到達する（あるいは到達できない）までの経路を知ることは、役立つことがあります。

この情報を保存するには、V_FitOptions を作成し、ビット 3 を設定します（変数 V_FitOptions=8）。

これにより、各反復におけるフィッティング係数の値を含む M_iterates という行列ウェーブが生成されます。

この行列は、各反復ごとに 1 行、各フィッティング係数ごとに 1 列を持ちます。

最後の列には、各反復におけるカイ二乗値が格納されます。

ビット 4: 画面更新の抑制

/N=1 フラグを設定するのと同じように動作します。

詳細はコマンド CurveFit のヘルプを参照してください。

Igor Pro 7.0 で追加されました。

ビット 5: エラーのみ

設定すると、/O フラグの設定と同じ（推測のみ）ですが、FuncFit ではさらに係数セットの W_sigma ウェーブを計算し、オプションで共分散行列（/M フラグ）も計算します。

このビットを設定すると特異行列エラーが発生する可能性があります。

Igor Pro 7.0 で追加されました。

V_chisq

V_chisq はフィッティングの度合いの尺度です。
重み付けウェーブを指定した場合にのみ絶対的な意味を持ちます。
詳細は「重み付け」のセクションを参照してください。

V_q

V_q (直線フィッティングのみ) はカイ二乗の信頼性を測る指標です。
重み付けウェーブを指定した場合にのみ有効です。
これは以下の式で計算される量 q を表します。

$$q = \text{gammq}((N-2)/2, \text{chisq}/2)$$

ここで gammq は不完全ガンマ関数 $1-P(a,x)$ を表し、 N はデータポイントの数です。
 q が 0.1 以上の場合、フィッティングの度合いは信頼できます。
 q が 0.001 の場合、フィッティングの度合いは信頼できる可能性があります。
 q が 0.001 未満の場合、データに系統的誤差があるか、誤った関数でフィッティングしていることを示します。

V_FitError と V_FitQuitReason

カーブフィッティング中にエラーが発生すると、通常は実行中のユーザー定義プロシージャがすべて中止されます。

これにより、エラーからの回復を試みるプロシージャを記述することは不可能になります。
ただし、予測不可能な数学的状況から生じる特定の種類のエラーについては、中止を回避することは可能です。
これを実現するには、V_FitError という変数を作成し、フィッティングを実行する前にその値をゼロに設定します。

フィッティング中にエラーが発生すると、V_FitError のビット 0 が設定されます。
特定のエラーでは、V_FitError の他のビットも設定されます。

<u>エラー</u>	<u>設定ビット</u>
任意のエラー	0
特異行列	1
メモリ不足	2
関数が NaN または INF を返した	3
フィッティング関数の停止が要求された	4
再帰的カーブフィッティング	5

再帰的カーブフィッティングとは、何らかの理由で既に実行中のカーブフィッティングがある状態で、第二のカーブフィッティングが開始されたことを意味します。

これは、ユーザー定義のフィッティング関数がカーブフィッティングを試みた場合、またはフィッティングを行うボタン操作プロシージャが別のクリックに早すぎる反応をした場合に発生する可能性があります。

フィッティングがエラーなしで反復を停止する理由は複数あります。

非線形フィッティングが反復を停止した理由の詳細情報を取得するには、V_FitQuitReason という変数を作成します。

フィッティング後、V_FitQuitReason はフィッティングが正常に終了した場合は 0、反復制限に達した場合は 1、ユーザーがフィッティングを停止した場合は 2、カイ二乗値が減少しない反復の制限に達した場合は 3 になります。

ウェーブの欠落やフィッティングに十分なデータポイントがないといった他の種類のエラーは、プログラマーのミスである可能性が高いです。

V_FitError はこれらのエラーを捕捉しませんが、特殊関数 AbortOnRTE と Igor の try-catch-endtry 構文を使えば、必要に応じて中止を回避できます。

以下は、NaN のみを含む可能性のあるデータセットに対してカーブフィッティングを試みる関数の例です。

```
Function PreventCurveFitAbort()  
    Make/O test = NaN  
    try  
        CurveFit/N/Q line, test; AbortOnRTE  
    catch  
        if (V_AbortCode == -4)  
            Print "Error during curve fit:"  
            Variable cfError = GetRTErr(1) // エラーをクリアするために 1 にする  
            Print GetErrMsg(cfError,3)  
        endif  
    endtry  
End
```

この関数を実行すると、出力は次のようになります。

```
Error during curve fit:  
You must have at least as many data points as fit parameters.
```

GetRTErr の呼び出しにより、エラー警告は表示されません。

GetErrMsg を使ってエラーメッセージを取得し、そのメッセージを履歴エリアに出力することで、エラーがユーザーに報告されます。

V_FitIterStart

V_FitIterStart は、ユーザー定義関数がフィッティングルーチンが新しい反復を開始しようとしていることを知る手段を提供します。

この関数の本来の目的（現在では旧式）は、畳み込みを含むユーザー定義のフィッティング関数の効率的な計算を可能にすることでした。

そのような関数は、現在では一括フィッティング関数を使うべきです。

詳細は「一括フィッティング関数」のセクションを参照してください。

S_Info

CurveFit または Funcfit を呼び出す関数内で文字列変数を作成すると、Igor はその変数にフィッティングに関する情報を提供するキーワード・値ペアを格納します。

<u>キーワード</u>	<u>キーワードに続く情報</u>
DATE	フィッティングの日付
TIME	フィッティングの日の時刻
FUNCTION	フィッティング関数の名前
AUTODESTWAVE	/D パラメーターフラグを使って自動宛先指定ウェーブを要求した場合、このキーワードはウェーブの名前を示す
YDATA	Y データウェーブの名前
XDATA	X データのウェーブのカンマ区切りリスト、または X ウェーブが存在しない場合は _calculated_。ほとんどの場合、X ウェーブは 1 つ

StringByKey を使って文字列から情報を取得します。

keySepStr を「=」、listSepStr を「;」に設定する必要があります。

変数における誤差：直交距離回帰

モデルをデータにフィッティングさせる時、通常はすべての誤差が従属変数に存在し、独立変数は完全に既知である（すなわち、X は完全に設定され、Y は誤差を伴って測定される）と仮定されます。

この仮定は往々にして真実に近いもので、従属変数の誤差が独立変数の誤差よりもはるかに大きい限り、カーブフィッティングに大きな差異をもたらすことは通常ありません。

誤差が平均ゼロ、分散定数の正規分布に従い、かつモデルが正確である場合、標準の最小二乗法によるフィッティングは最尤解を与えます。

これは前述した手法です（「カーブフィッティングの概要」のセクションを参照）。

ただし、従属変数と独立変数の両方の誤差が同程度である場合もあります。

この状況は、変数誤差、測定誤差モデル、またはランダム回帰変数モデルなど、様々な名前で呼ばれます。

従属変数と独立変数で同様の誤差が生じうるモデルの例として、物体の軌跡を表面に沿ってフィッティングする場合が挙げられます。

関与する変数は、物体の位置に関する直交座標の測定値（様々な時点における）となります。

おそらく測定誤差は類似することになります。

というのも双方とも空間測定を伴うためです。

標準の最小二乗法や通常の最小二乗法を用いてこのようなデータをフィッティングすると、解にバイアスが生じる可能性があります。

この問題を解決するために、直交距離回帰（Orthogonal Distance Regression / ODR）を提供します。

ODR は、従属変数の二乗誤差の合計を最小化するのではなく、モデル係数と独立変数の値の両方を調整することで、データからフィッティング曲線までの直交距離を最小化します。

これは「全最小二乗法」または「縮小主軸（reduced major axis）回帰（RMA）」とも呼ばれます。

カーブフィッティングでは、Igor Pro は自由に利用可能な ODRPACK95 の修正版を使います。
CurveFit、FuncFit、FuncFitMD コマンドはすべて、/ODR フラグを使って ODR フィッティングを実行できます
（/ODR フラグの詳細については CurveFit コマンドのドキュメントを、ODRPACK95 による ODR フィッティングの実装については「カーブフィッティングの参考文献」のセクションを参照してください）。
当社の ODRPACK95 バージョンは、スレッドセーフ化および単一フィッティング中の自動マルチスレッディング処理を行うよう修正されています。

スレッドセーフな ODR Fitting フィッティング

Igor Pro 8.0 以降、ODR フィッティングはスレッドセーフです。
スレッドセーフなユーザー定義関数から ODR フィッティングを実行でき、Igor のプリエンプティブスレッドを使って複数の ODR フィッティングを同時に実行できます。

ODR フィッティングのための重み付けウェーブ

通常の最小二乗法によるフィッティングと同様に、ODR フィッティングにおいても誤差の予想大きさを示す重み付けウェーブを指定できます。

ただし ODR フィッティングでは従属変数と独立変数の両方に誤差が存在するため、両方の重み付けウェーブを受け付けます。

独立変数の重み付けウェーブを指定するには /XW フラグを使います。

重み付けウェーブを指定しない場合、誤差の分散は 1.0 と仮定されます。

従属変数と独立変数の誤差が真に同程度の大きさである場合には、この仮定が許容される可能性があります。

しかし従属変数と独立変数の大きさが大きく異なる場合、誤差の大きさも大きく異なる可能性が高く、適切なフィッティングには重み付けが不可欠です。

通常の最小二乗法では 1 つの重み付けウェーブのみが存在するのとは異なり、ODR フィッティングでは重みの絶対値だけでなく、X と Y の重みの相対的な大きさも重要となります。

ODR 初期推定値

係数が線形である通常の最小二乗法によるフィッティングは直接解くことができます。

初期推定値は不要です。

組み込みの線形、多項式、および 2 次元多項式曲線近似関数は係数が線形であり、初期推定値を必要としません。

係数が非線形である関数に対する通常の最小二乗法によるフィッティングは、開始点が必要な反復処理です。

つまり、フィッティング係数の初期推定値を指定する必要があります。

初期推定値に必要な精度は、フィッティングする関数とデータの品質に大きく依存します。

組み込みのフィッティング関数も、適切な初期推定値のセットを計算しようとしませんが、ユーザー定義のフィッティングでは、独自の初期推定値を指定する必要があります。

ODR フィッティングは、フィッティング方程式に非線形性を導入するため、フィッティング係数に対して線形であるフィッティング関数であっても、反復フィッティングと初期推定値が必要となります。

直線、多項式、および多項式 2 次フィッティング関数の場合、ODR フィッティングは初期推定値を得るために通常

最小二乗法を使います。

非線形の組み込みフィッティング関数については、フィッティング方法に関係なく、同じ初期推定値が使われます。

独立変数はフィッティング中に調整されるため、独立変数の調整には初期推定値も必要です。

初期推定値は、/XR フラグで指定された1つ以上のウェーブ（独立変数ごとに1つ）を通して Igor に送信されます。

X 残差ウェーブも出力されます。

「ODR フィッティングの結果」のセクションを参照してください。

/XR フラグが指定されていない場合、独立変数の値に対する調整の初期推測値は 0 に設定されます。

これは通常、適切な設定です。

ただし、フィッティング関数がほぼ垂直である領域では、フィッティングを成功させるために 0 以外の推測値が必要になる場合があります。

そのような状況の一例としては、特異点付近の領域が挙げられます。

独立変数の調整を保持する

場合によっては、独立変数の入力値の一部が正確（またはほぼ正確）であり、調整すべきではないと確信する理由があるかもしれません。

どの値を調整すべきでないかを指定するには、/XHLD フラグを使って、独立変数ごとに1つずつ、合計 X 個の保持ウェーブを指定します。

これらのウェーブは、調整すべき値に対応する 0、または保持すべき値に対応する 1 で埋める必要があります。

これは、フィッティング中にフィッティング係数を設定値でホールドする /H フラグに似ています。

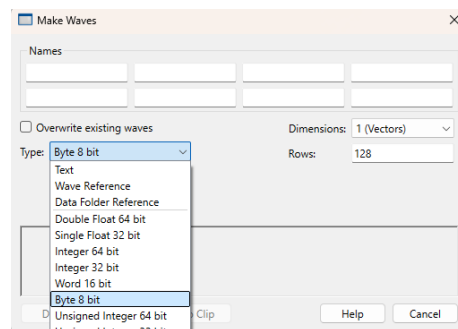
ただし、ODR フィッティングと独立変数の値の場合、ホールドは 1 と 0 の文字列ではなくウェーブで指定されます。

これは、膨大な数の 1 と 0 が必要になる可能性があるためです。

メモリを節約するために、ホールドにはバイトウェーブを使用できます。

Make Waves ダイアログで、Type メニューから Byte 8 Bit を選択します。

Make コマンドには /B フラグを使います。



ODR フィッティングの結果

通常の最小二乗法によるフィッティングでは、フィッティング係数を調整し、従属変数のモデル値を計算します。オプションで、モデルと従属変数データの差である残差を計算させることができます。

ODR フィッティングは、最小直交距離フィッティングを求める時に、フィッティング係数と独立変数の値の両方を調整します。

従属変数の残差に加えて、独立変数の残差を含むウェーブ、および独立変数の調整済み値を含むウェーブを計算して返すことができます。

独立変数の残差は、/XR フラグで指定されたウェーブを通じて返されます。

これらのウェーブの内容は、独立変数の調整に対する初期推定値の入力となるため、注意が必要です。

ほとんどの場合、フィッティング前にウェーブをゼロに設定する必要があります。

調整された独立変数の値は、/XD フラグで指定したウェーブに配置されます。

自動宛先ウェーブ (/D フラグ。「宛先ウェーブ」のセクションを参照) を設定した場合、結果は独立変数の等間隔の値セットにおけるモデル値を含むウェーブとなります。

モデル値を取得する特定のウェーブを指定することもできます (/D=wave)。

値は、フィッティングへの入力として指定した独立変数の値で計算されます。

ODR フィッティングの場合、モデルのグラフを作成するには、入力 X 値ではなく /XD フラグの出力値が適切な X ウェーブとなります。

制約と ODR フィッティング

最小二乗法によるフィッティング (/ODR=0) では、フィッティング係数を範囲内に保持する制約式を含むテキストウェーブを指定できます。

これらの式は、単純な境界制約（フィッティング係数の値を特定の値以上または以下にホールド）を適用したり、フィッティング係数の線形結合に境界を適用したり（例えば $a + b > 1$ を制約）するために使用できます。

ODR (/ODR=1 以上) を使ったフィッティングでは、単純な境界制約のみがサポートされます。

ODR フィッティングからの誤差推定

カーブフィッティングでは、出力にはフィッティング係数の誤差の推定値が含まれます。

これらの推定値は、解におけるカイ二乗面の線形化された二次近似から計算されます。

通常の最小二乗法による線形フィッティング (line、poly、poly2D フィッティング関数) の場合、カイ二乗面は実際には二次関数であり、測定誤差が平均ゼロで分散が一定な正規分布に従う場合、推定値は正確です。

フィッティング関数がフィッティング係数に関して非線形である場合、誤差推定値は近似値となります。

この近似の精度は、非線形性の性質に依存します。

ODR フィッティングでは、係数に関して線形であっても、フィッティング方程式自体が非線形性を導入します。

したがって、ODR フィッティングによる誤差推定は常に近似となります。

詳細については、「カーブフィッティングの参考文献」のセクションを参照してください。

ODR フィッティングの例

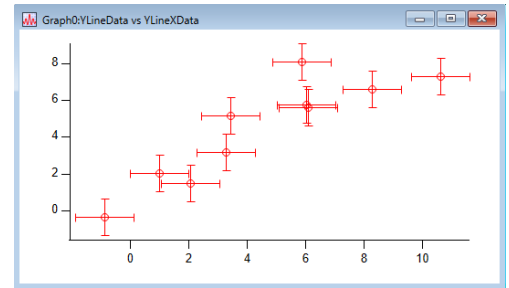
簡単な例：重み付けなしの直線フィッティングです。

これらのコマンドを実行すると、SetRandomSeed の呼び出しにより、gnoise が提供する「測定誤差」がここに示した例と同じになります。

```
SetRandomSeed 0.5           // 「ランダム」なデータが常に同じになるように...
Make/N=10 YLineData, YLineXData
YLineXData = p+gnoise(1)     // gnoise は x 値の誤差をシミュレートします
YLineData = p+gnoise(1)     // gnoise は y 値の誤差をシミュレートします
```


// 標準偏差誤差を示すエラーバー付きのきれいなグラフを作成します

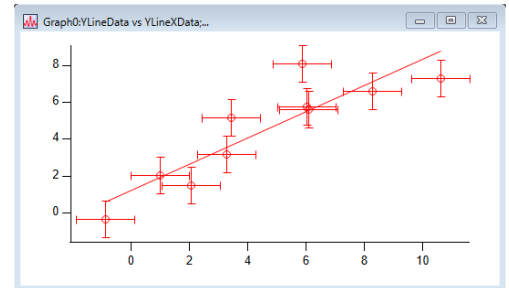
```
Display YLineData vs YLineXData
ModifyGraph mode=3,marker=8
ErrorBars YLineData XY,const=1,const=1
```



これで、データに対して線形フィッティングを実行する準備が整いました。

まず、標準的なカーブフィッティングを行います。

```
CurveFit line, YLineData/X=YLineXData/D
```



このコマンドは次の履歴レポートを生成します。

```
Untitled
8 •CurveFit line, YLineData/X=YLineXData/D
9   fit_YLineData= W_coef[0]+W_coef[1]*x
10  W_coef={1.2363,0.70802}
11  V_chisq= 14.3185;V_npnts= 10;V_numNaNs= 0;V_numINFs= 0;
12  V_startRow= 0;V_endRow= 9;V_q= 1;V_Rab= -0.812244;V_Pr= 0.889645;
13  V_r2= 0.791469;
14  W_sigma={0.725,0.128}
15  Coefficient values ± one standard deviation
16    a = 1.2363 ± 0.725
17    b = 0.70802 ± 0.128
18 |
```

次に、直交距離フィッティングをするために /ODR=2 を使います。

```
CurveFit/ODR=2 line, YLineData/X=YLineXData/D
```

これにより、次の結果が得られます。

```
Untitled
18 •CurveFit/ODR=2 line, YLineData/X=YLineXData/D
19   Fit converged properly
20   fit_YLineData= W_coef[0]+W_coef[1]*x
21   W_coef={0.93352,0.77406}
22   V_chisq= 9.24935;V_npnts= 10;V_numNaNs= 0;V_numINFs= 0;
23   V_startRow= 0;V_endRow= 9;
24   W_sigma={0.745,0.133}
25   Coefficient values ± one standard deviation
26     a = 0.93352 ± 0.745
27     b = 0.77406 ± 0.133
28 |
```

X 方向の補正値と Y 方向の残差の合計を出力します。

```
Duplicate/O YLineData, YLineDataXRes, YLineDataYRes
CurveFit/ODR=2 line, YLineData/X=YLineXData/D/XR=YLineDataXRes/R=YLineDataYRes
```

```

29 •CurveFit/ODR=2 line, YLineData/X=YLineXData/D/XR=YLineDataXRes/R=YLineDataYRes
30 Fit converged properly
31 fit_YLineData= W_coef[0]+W_coef[1]*x
32 YLineDataYRes= YLineData[p] - (W_coef[0]+W_coef[1]*YLineXData[p])
33 W_coef={0.93354,0.77406}
34 V_chisq= 9.24935;V_npnts= 10;V_numNaNs= 0;V_numINFs= 0;
35 V_startRow= 0;V_endRow= 9;
36 W_sigma={0.745,0.133}
37 Coefficient values ± one standard deviation
38 a = 0.93354 ± 0.745
39 b = 0.77406 ± 0.133
40|

```

誤差バーを使って残差を示すグラフを表示します。

Display YLineData vs YLineXData

ModifyGraph mode=3,marker=8

AppendToGraph fit_YLineData

ModifyGraph rgb(YLineData)=(0,0,65535)

ErrorBars YLineData BOX,wave=(YLineDataXRes,YLineDataXRes),wave=(YLineDataYRes,YLineDataYRes)

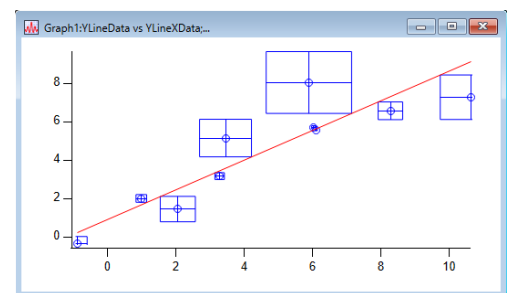
このグラフのボックスは誤差推定値ではなく、フィッティングからの残差を示しています。

つまり、データとフィッティングモデルとの差分です。

これは ODR フィッティングであるため、X 軸と Y 軸の両方に残差が存在します。

これを示す最も便利な方法はエラーバーです。

各ボックスの一角がモデル線に接していることに注目してください。



次の例では、Y 値と誤差が X 値と誤差に比べて小さい指数関数的なカーブフィッティングを行います。

カーブフィッティングの履歴レポートは、解の結果のみを表示するように編集されています。

まず、偽のデータとグラフを示します。

// 「ランダム」なデータが常に同じになるように...

SetRandomSeed 0.5

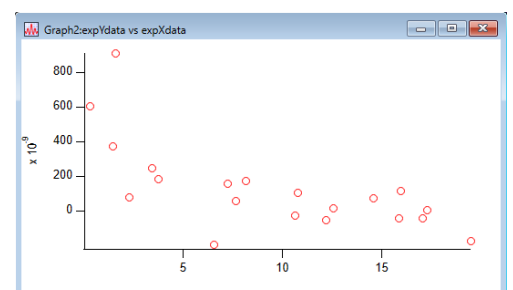
Make/D/O/N=20 expYdata, expXdata

expYdata = 1e-6*exp(-p/2)+gnoise(1e-7)

expXdata = p+gnoise(1)

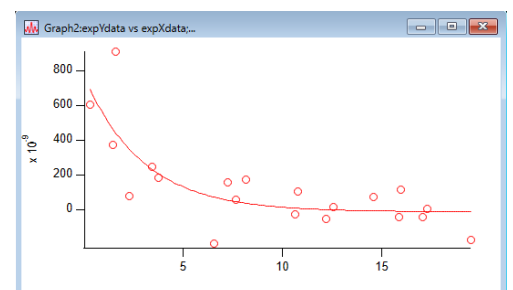
Display expYdata vs expXdata

ModifyGraph mode=3,marker=8



通常の指数関数的なフィッティングを行います。

CurveFit exp, expYdata/X=expXdata/D



```

51 •CurveFit exp, expYdata/X=expXdata/D
52 Fit converged properly
53 fit_expYdata= W_coef[0]+W_coef[1]*exp(-W_coef[2]*x)
54 W_coef={-9.5547e-09,7.8898e-07,0.33993}
55 V_chisq= 4.71e-13;V_npnts= 20;V_numNaNs= 0;V_numINFs= 0;
56 V_startRow= 0;V_endRow= 19;
57 W_sigma={5.83e-08,1.8e-07,0.155}
58 Coefficient values ± one standard deviation
59 y0 = -9.5547e-09 ± 5.83e-08
60 A = 7.8898e-07 ± 1.8e-07
61 invTau = 0.33993 ± 0.155
62

```

X と Y の残差付き、重み付けなしの ODR フィッティングを行います。

```

Duplicate/O expYdata, expYdataResY, expYdataResX
expYdataResY=0
expYdataResX=0
CurveFit/ODR=2 exp, expYdata/X=expXdata/D/R=expYdataResY/XR=expYdataResX

```

```

65 •CurveFit/ODR=2 exp, expYdata/X=expXdata/D/R=expYdataResY/XR=expYdataResX
66 Fit converged properly
67 fit_expYdata= W_coef[0]+W_coef[1]*exp(-W_coef[2]*x)
68 expYdataResY= expYdata[p] - (W_coef[0]+W_coef[1]*exp(-W_coef[2]*expXdata[p]))
69 W_coef={-9.638e-09,7.8884e-07,0.33968}
70 V_chisq= 4.71e-13;V_npnts= 20;V_numNaNs= 0;V_numINFs= 0;
71 V_startRow= 0;V_endRow= 19;
72 W_sigma={5.83e-08,1.8e-07,0.155}
73 Coefficient values ± one standard deviation
74 y0 = -9.638e-09 ± 5.83e-08
75 A = 7.8884e-07 ± 1.8e-07
76 invTau = 0.33968 ± 0.155
77

```

グラフを表示します。

```

Display /W=(137,197,532,405) expYdata vs expXdata
AppendToGraph fit_expYdata
ModifyGraph mode(expYdata)=3
ModifyGraph marker(expYdata)=8
ModifyGraph lSize(expYdata)=2
ModifyGraph rgb(expYdata)=(0,0,65535)
ErrorBars expYdata
BOX,wave=(expYdataResX,expYdataResX),wave=(expYdataResY,expYdataResY)

```

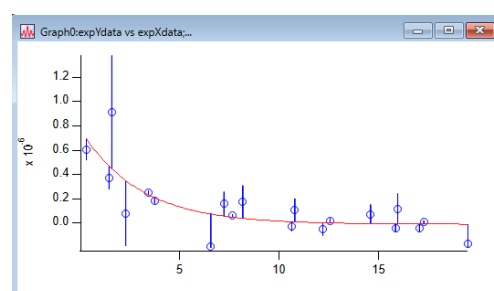
Y 値が X 値に比べて非常に小さいため、かつ Y の小さな誤差を反映する重み付けを使わなかったため、残差ボックスは細長い形状となります。

縦軸のグラフ目盛りが横軸と同じスケールであれば、ボックスはおおよそ正方形になります。

データラインはほぼ水平線に近くなります。

これを理解する一つの方法は、ODR メソッドが本質的に幾何学的な手法であることを思い出すことです。

この例では、グラフの縦軸の目盛りが非常に大きく拡大されていますが、ODR メソッドは拡大されていない目盛りで機能し、その場合、近似曲線に対する垂直線はほぼ完全に垂直になります。



これで適切な重み付けを追加できます。

gnoise() で「測定誤差」を追加したため、正しい重み付けを決定するのは簡単です。

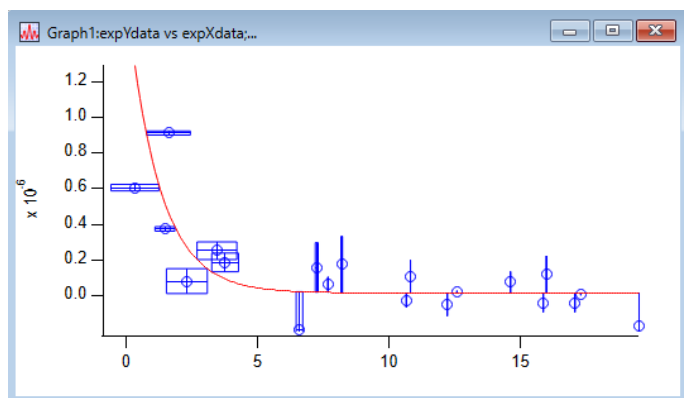
```
Duplicate/O expYdata, expYdataWY
```

```
expYdataWY=1e-7
```

```
Duplicate/O expYdata, expYdataWX
```

```
expYdataWX=1
```

```
CurveFit/ODR=2 exp, expYdata/X=expXdata/D/R=expYdataResY/XR=expYdataResX/W=expYdataWY/XW=expYdataWX/I=1
```



```
Untitled
182 •CurveFit/ODR=2 exp, expYdata/X=expXdata/D/R=expYdataResY/XR=expYdataResX/W=expYdataWY/XW=expYdataWX/I=1
183 Fit converged properly
184 fit_expYdata= W_coef[0]+W_coef[1]*exp(-W_coef[2]*x)
185 expYdataResY= expYdata[p] - (W_coef[0]+W_coef[1]*exp(-W_coef[2]*expXdata[p]))
186 W_coef={1.3827e-08,1.6787e-06,0.81163}
187 V_chisq= 20.2101;V_npnts= 20;V_numNaNs= 0;V_numINFs= 0;
188 V_startRow= 0;V_endRow= 19;
189 W_sigma={2.75e-08,1.49e-06,0.428}
190 Coefficient values ± one standard deviation
191 y0 = 1.3827e-08 ± 2.75e-08
192 A = 1.6787e-06 ± 1.49e-06
193 invTau = 0.81163 ± 0.428
194
```

ODR フィッティングにおける重み付けウェーブを解釈する一つの方法は、それらが幾何学的スケーリングを提供すると考えることです。

この例では、垂直方向の寸法は水平方向の約 107 分の 1 です。

重み付けウェーブによって垂直方向の次元がスケーリングされると、両方向の次元が類似し、フィッティングカーブからデータポイントへの垂直距離はもはや単なる垂直距離ではなくなります。