CONTENTS

ビジュアルヘルプ – データのインポートとエクスポート（1）

データのインポート

Igor のほとんどのユーザーは、他のプログラムで作成したファイルからデータを読み込んでウェーブを作成します。
ファイルをロードするプロセスでは、新しいウェーブが作成され、その中にファイルからのデータが保存されます。
オプションとして、新しいウェーブを作成する代わりに既存のウェーブを上書きすることもできます。
ウェーブは、数値またはテキストで、１～４の次元です。

Igor には、データファイルを読み込むためのさまざまなルーチンが用意されています。
数値データやテキストデータのファイルの形式は、すべてのプログラムで読み書きできる単一の形式はありません。

データ交換に使われるファイルは、テキストファイルとバイナリファイルの２つの大きなカテゴリに分類されます。
テキストファイルは通常、プログラム間のデータ交換に使われます。
テキストファイルと呼ばれていますが、数値データ、テキストデータ、またはその両方を格納することができます。
いずれの場合も、データはテキストエディターで読み取れるプレーンテキストとしてエンコードされています。
バイナリファイルには、通常、あるプログラムに固有の方法で効率的にエンコードされたデータが含まれており、テキストエディターでは表示できません。

データ交換の形式として世界的に広く受け入れられているものに最も近いものは「区切り記号付きテキスト」フォーマットです。
これは、改行文字（CR-MacOS）、ラインフィード文字（LR-Unix）、改行／ラインフィード（CRLF-Windows）で区切られた行と、タブまたはカンマで区切られた列からなる、数値またはテキストデータの行と列で構成されています。
タブまたはカンマは「区切り文字/記号（デリミター）」と呼ばれます。
CR、LF、CRLF の文字は「終端文字（ターミネーター）」と呼ばれます。
Igor は、ほとんどのプログラムで作成された区切り記号付きテキストファイルを読み取ることができます。

FORTRAN プログラムは、通常、固定長フィールドのテキストファイルを作成するため、このファイルでは、各データ列に固定バイト数が使われ、列間のパディングとしてスペースが使われます。
Load Fixed Field Text ルーチンは、これらのファイルを読み込むように設計されています。

テキストファイルは、任意のテキストエディターで作成、確認、編集できるので便利です。
Igor では、この目的のためにノートブックウィンドウを使うことができます。
テキストファイルに通常とは異なる形式のデータがある場合、読み込む前に手動で編集する必要があるかもしれません。

科学機器やカスタムプログラムによって生成されたテキストファイルには、通常、ファイルの先頭に「ヘッダー」情報が含まれている場合が多いです。
ヘッダーはデータブロックの一部ではありませんが、それに関連する情報を含んでいます。
Igor のテキスト読み込みルーチンは、ヘッダーではなくデータブロックを読み込むように設計されています。
Load General Text ルーチンは、通常、ヘッダーを自動的にスキップすることができます。
Load Delimited Text ルーチンと Load Fixed Field Text ルーチンは、データブロックがファイルの先頭にない場合、そのブロックがどこから始まるのかを指示する必要があります。

上級ユーザーであれば、Igor の文字列操作機能に加えて、Open、FReadLine、StrSearch、sscanf、Close の各コマンドを使って、ヘッダー内の情報を読み取り、解析するプロシージャを記述することができます。
Igor には、Load File Demo というサンプル Experiment が含まれており、このことを説明しています。

以下の表は、Igor で使うことができるデータロードルーチンと、その主な特徴をまとめたものです。

ファイル形式	説明
区切り記号付きテキスト	スプレッドシート、データベースプログラム、データ取得プログラム、テキストエディター、カスタムプログラムによって作成される。 これは、プログラム間でデータをやり取りするときに最も一般的に使われる形式。 フォーマット：`<data><delimiter><data><terminator>` 任意の行と列を持つ 1 つのデータブロックを含む。 列ラベルの行は任意。 数値、テキスト、日付、時刻、日付/時刻の列をロード可能。 1D ウェーブに列を 2D ウェーブにブロックをロード可能。 列の長さは、等しくても異なっていてもよい。 「区切り文字付きテキストファイルの読み込み」のセクションを参照。
固定長フィールドテキスト	FORTRAN プログラムによって作成される。 フォーマット：`<data><padding><data><padding><terminator>` 任意の行の列を持つ 1 つのデータブロックを含む。 各列は、パディングに使われるスペース文字を含め、固定バイト数で構成されている。 数値、テキスト、日付、時刻、日付/時刻の列をロード可能。 1D ウェーブに列を 2D ウェーブにブロックをロード可能。 列は通常、長さが同じだが、そうである必要はない。 「固定フィールドテキストファイルの読み込み」のセクションを参照。
一般的なテキスト	スプレッドシート、データベースプログラム、データ取得プログラム、テキストエディター、カスタムプログラムによって作成される。 フォーマット：`<number><white space><number><terminator>` 任意の行と列を持つ 1 つ以上のデータブロックを含む。 列ラベルの行は任意。 非数値テキスト、日付、時刻を含む列は処理できない。 1D ウェーブに列を 2D ウェーブにブロックをロード可能。 列の長さは等しい必要がある。 Load General Text ルーチンは、数値以外のヘッダーテキストを自動的にスキップ可能。 「一般的なテキストファイルの読み込み」のセクションを参照してください。
Igor テキスト	Igor、カスタムプログラムで作成される。 主に、カスタムプログラムから Igor にデータやコマンドを供給する手段として使われる。 フォーマット：「Igor テキストファイル形式」のセクションを参照。 数値、テキストデータをロード可能。 データを 1～4 次元のウェーブにロード可能。

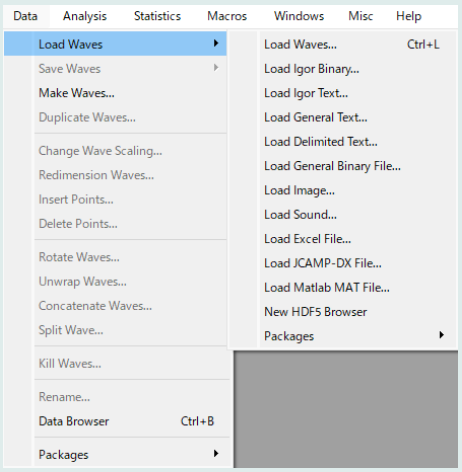
	任意の数のウェーブと列を含む、１つ以上のウェーブブロックを含む。 Igor の特別なキーワード、数値、コマンドで構成されている。 「Igor テキストファイルの読み込み」のセクションを参照。
Igor バイナリ	Igor、カスタムプログラムで作成される。 ウェーブデータを保存するために Igor で使われる。 各ファイルには、１～４次元の Igor ウェーブデータが含まれる。 フォーマット：Igor Technical Note #003 “Igor Binary Format” を参照。 「Igor バイナリデータの読み込み」のセクションを参照。
画像	さまざまなプログラムで作成される。 フォーマット：常にバイナリ。ファイル形式によってさまざま。 JPEG、PNG、TIFF、BMP、Sun Raster 画像ファイルをロード可能。 TIFF 画像スタックを含むデータをマトリックスウェーブにロード可能。 「画像ファイルの読み込み」のセクションを参照。
一般的なバイナリ	他のプログラムによって作成されたバイナリファイル。 バイナリファイルのフォーマットを理解していれば、データを読み込ませることが可能。 「一般的なバイナリファイルの読み込み」のセクションを参照。
Excel	.xls と .xlsx ファイル形式をサポート。 「Excel ファイルの読み込み」のセクションを参照。
HDF4	Igor エクステンションを有効にする必要がある。 ヘルプ HDF Loader XOP を参照。
HDF5	ヘルプ HDF5 in Igor Pro を参照。
Matlab	「MATLAB MAT ファイルの読み込み」のセクションを参照。
JCAMP-DX	JCAMP-DX 形式は主に赤外分光で使われる。 「JCAMP ファイルの読み込み」のセクションを参照。
音声	さまざまな音声ファイル形式をサポート。 「音声ファイルの読み込み」のセクションを参照。
TDMS	National Instruments の TDMS ファイルからデータをロード可能。 エクステンションを有効にする必要がある。 詳細は、ヘルプ TDM XOP を参照。
Nicolet WFT	古い Nicolet オシロスコープで作成されたデータをロード可能。 エクステンションを有効にする必要がある。 詳細は、ヘルプ NILoadWave XOP を参照。
SQL データベース	SQL データベースからデータをロード可能。

エクステンションを有効にする必要があり、データベースプログラミングの知識が必要。

「SQL データベースへのアクセス」のセクションを参照。

Load Waves サブメニュー

これらのルーチンはすべて、Data メニューの Load Waves サブメニューからアクセスできます。



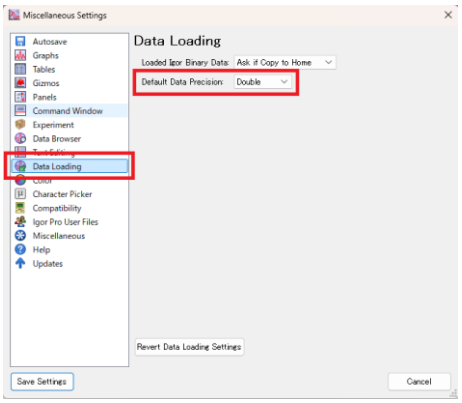
Load Waves : Load Waves ダイアログを開きます。
このダイアログでは、Igor バイナリウェーブファイル、Igor テキストファイル、区切り記号付きテキストファイル、一般テキストファイル、固定フィールドテキストファイルの読み込みのためのビルトインルーチンにアクセスでき、利用可能なすべてのオプションにアクセスできます。

Load Igor Binary / Load Igor Text / Load General Text / Load Delimited Text : Load Waves サブメニュー内のショートカットであり、デフォルトの設定で各ファイル読み込みルーチンにアクセスします。
使うことができるオプションを確認するには、Load Waves アイテムからアクセスすることをお勧めします。

Load Image : Load Image ダイアログにリンクしており、さまざまな種類の画像ファイルを読み込む機能を提供します。

固定フィールドのテキストを読み込むためのショートカット項目はありません。

Load General Text と Load Delimited Text によって作成される数値ウェーブの精度は、Miscellaneous Settings ダイアログ（Misc メニュー）の Data Loading セクションの Default Data Precision の設定によってコントロールされます。



行の終端記号

テキストの行の終わりを示す文字または文字列は、「行終端文字」または略して「終端文字」と呼ばれます。
異なるコンピューターシステムでは異なる終端文字が使われます。

Mac OS 9 ではキャリッジリターン文字（CR）が使われていました。

Unix では改行コード（LF）を使います。

Windows ではキャリッジリターンとラインフィードのシーケンス（CRLF）を使います。

ウェーブを読み込む時、Igor は 1 つの CR、1 つの LF、または CRLF を改行の終わりとして扱います。
これにより、変換なしで様々なコンピューター上のファイルサーバーからテキストデータを読み込むことが可能になります。

LoadWave のテキストのエンコーディング

このセクションの説明は、Load General Text、Load Delimited Text、Load Fixed Field Text、Load Igor Text を使ってテキストファイルを読み込む場合に適用されます。

ファイルがバイト指向のテキストエンコーディング（UTF-16 または UTF-32 以外のテキストエンコーディング）を使っていて、ファイルが数字または ASCII テキストのみで構成されている場合は、テキストエンコーディングについて心配する必要はありません。

ファイルが UTF-16 または UTF-32 を使っている場合、または非 ASCII テキストを含んでいる場合は、
LoadWave コマンドにファイルのテキストエンコードを指示する必要があります。
詳細は「LoadWave テキストエンコーディングの問題」のセクションを参照してください。

区切り記号付きテキストファイルの読み込み

区切り記号付きテキストファイルは、タブまたはカンマで区切られた値の行から構成され、行の末尾にはキャリッジリターン（CR）、ラインフィード（LF）、またはキャリッジリターン/ラインフィード（CRLF）のシーケンスが続きます。
オプションとして、列ラベルの行があるかもしれません。
Igor は、ファイル内の各列を個別の 1D ウェーブに読み込むことも、すべての列を単一の 2D ウェーブに読み込むこともできます。
ただし、すべてのデータが利用可能なメモリに格納できる必要がある、という条件を除いて、行数や列数に制限はありません。

区切り記号付きテキストファイルには、数字やテキストに加えて、日付、時刻、日付/時刻が含まれる場合があります。
Load Delimited Text ルーチンは、ファイル内の各列にどの形式が適切かを自動的に判断しようと試みます。
必要に応じて、この自動判断を上書きすることができます。

数値の列には、数値の他に NaN および [±]INF を含めることができます。
NaN は「Not a Number」の略で、数値の列の空白または欠損値を表す方法です。
INF は「無限大」を意味します。
数値または日付/時刻の列で、その列のフォーマットに従って解釈できないテキストを見つけた場合は、NaN として処理します。

いずれかの列で、データ文字が先行しない区切り文字（例えば、連続する 2 つのタブ）があった場合、これは欠損値とみなされ、ウェーブには空白が格納され、数値ウェーブの場合は、空白は NaN で表されます。
テキストウェーブでは、ゼロ文字を持つ要素で表現されます。

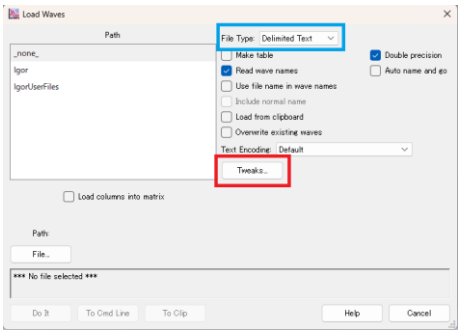
列のフォーマットの決定

Load Delimited Text ルーチンは、ロードされる各データ列のフォーマットを決定しなければなりません。
特定の列のフォーマットは、数値、日付、時間、日付/時間、またはテキストとすることができます。
テキスト列はテキストウェーブにロードされ、その他のタイプは数値ウェーブにロードされ、日付は 1904 年 1 月 1 日からの秒数としてあらわされます。

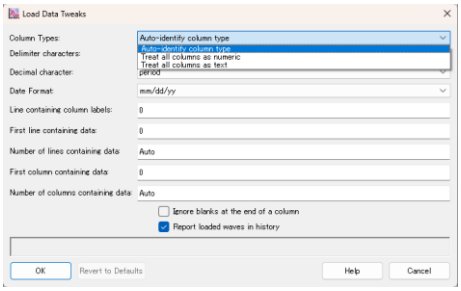
列のフォーマットを決定する方法は４つあります：

- 列タイプを自動的に特定
- すべての列を数値で処理
- すべての列をテキストで処理
- 各列のフォーマットを明示的に指定するために　LoadWave /B フラグを使用

これらの方法のうち最初の３つは、Load Waves ダイアログの Tweaks サブダイアログにある Column Types ポップアップメニューから選択できます。
/B フラグを使うには、LoadWave コマンドに手動でフラグを追加する必要があります。
これは通常、プロシージャで行われます。



「列タイプを自動的に特定（auto-identify column type）」の方法では、ファイルを調査して各列のフォーマットを決定しようと試みます。
これは、メニュー Data → Load Waves → Load Delimited Text を選択したときのデフォルトの方法です。
Igor は、各列の最初の空白ではない値を探し、その列の内容に基づいて判断します。
ほとんどの場合、自動識別が機能するため、他の方法を使う必要はありません。



「すべての列を数値で処理（treat all columns as numeric）」の方法では、すべての列を数値ウェーブにロードします。
データの一部が数値でない場合、出力ウェーブに NaN が出力されます。
後方互換性を保つため、コマンドラインまたはプロシージャから LoadWave/J コマンドを使う時には、これがデフォルトの方法となります。
「auto-identify column type」の方法を使うには、LoadWave/J/K=0 を使う必要があります。

「すべての列をテキストで処理（treat all columns as text）」の方法では、すべての列をテキストウェーブにロードします。
この方法は、テキストウェーブにファイルを読み込んで、文字列操作機能を使ってファイルを加工したいという希なケースで使えるかもしれません。

/B の方法についての詳細は、LoadWave コマンドのヘルプの Specifying Characteristics of Individual Columns を参照してください。

日付/時刻フォーマット

Load Delimited Text ルーチンは、多くのフォーマットの日付を処理できます。
いくつかの「標準」フォーマットがサポートされており、さらに、「カスタム」フォーマットを指定することもできます（「カスタムの日付フォーマット」のセクションを参照）。

標準の日付フォーマットとしては：

```
mm/dd/yy        (月/日/年)
mm/yy           (月/年)
dd/mm/yy        (日/月/年)
```

mm/dd/yy の代わりに dd/mm/yy フォーマットを使う時には、Tweak（微調整）を設定する必要があります。
「区切り文字付きテキストの調整」のセクションを参照してください。

スラッシュ（/）の代わりに、ダッシュ（-）やドット（.）を区切り文字として使うこともできます。

Igor は次の形式の時刻を扱うこともできます：

```
[+][-]hh:mm:ss [AM PM]        (時:分:秒)
[+][-]hh:mm:ss.ff [AM PM]     (時:分:[小数点付き]秒)
[+][-]hh:mm [AM PM]           (時:分)
[+][-]hhhh:mm:ss.ff           (時間:分:[小数点付き]秒)
```

Igor Pro 6.23 以降では、秒の小数点のドットの代わりにコロンも受け入れるようになりました。

最初の３つ名形式は、時間帯形式です。
最後の１つは経過時間です。経過時間では、時間は 0 から 9999 の範囲です。

年は２桁（99）または４桁（1999）で指定できます。
２桁の年が 00～39 の範囲の場合、2000～2039 として処理されます。
２桁の年が 40～99 の範囲の場合、1940～1999 として処理されます。

Load Delimited Text ルーチンは、これらの日付フォーマットのいずれか、単一スペースまたは文字「T」で始まり時刻フォーマットのいずれかで構成される日付/時刻も処理できます。
日付/時刻の値として、<日付><スペース><時刻>を読み込むには、区切り文字としてスペースを指定してはいけません。

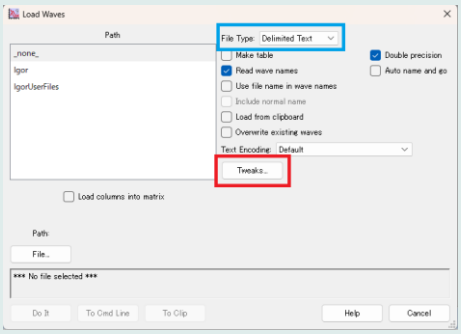
カスタムの日付フォーマット

データファイルに「標準」フォーマット以外の日付が含まれている場合、Load Delimited Text を使って、使用する日付フォーマットを正確に指定することができます。

これは、Load Waves ダイアログの Tweaks ボタンからアクセスできる Delimited Tweaks ダイアログを使って行います。

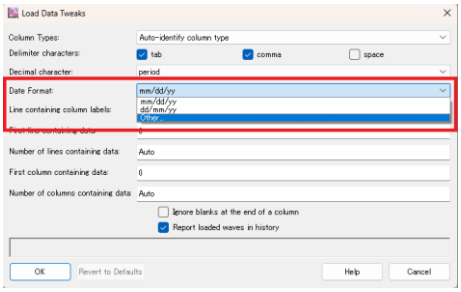
1. **メニュー Data → Load Waves → Load Waves を選択します。**

   File Type ポップアップメニューで Delimited Text を選択し、Tweaks ボタンをクリックします。

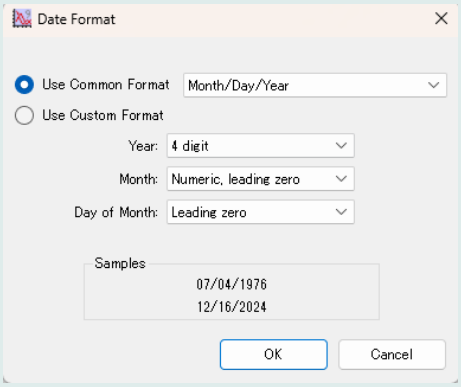


2. **Date Format ポップアップメニューから Other を選択します。**

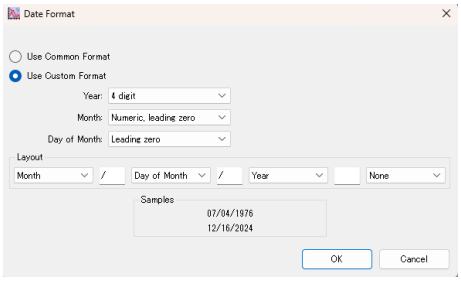
   これは Date Format ダイアログを開きます。



3. Use Common Format ラジオボタンを選択すると、一般フォーマットのポップアップメニューから選択できます。一般フォーマットを選択した後でも、２桁または４桁の年号を使うか、ゼロを先頭に付けるかどうかなど、フォーマットの細かな部分をコントロールすることができます。



4. ファイルの日付形式が一般フォーマットのいずれにも一致しない場合、Use Custom Format ラジオボタンを選択して、完全なカスタムフォーマットを使うことができます。
   まずは、使うフォーマットに最も近い一般フォーマットを選択し、次に Use Custom Format ボタンをクリックするのが最善です。
   それから、最終的なフォーマットに到達するために、細かな変更を加えることができます。



一般的なフォーマットまたは完全なカスタムフォーマットのどちらかを使う場合、指定するフォーマットはファイル内の日付と完全に一致している必要があります。

区切り記号付きテキストとしてデータを読み込むとき、「October 11, 1999」のようにカンマを含む日付形式を使う場合は、LoadWave コマンドでカンマが区切り記号として扱われないようにする必要があります。
これは Delimited Text Tweaks ダイアログを使って行うことができます。

991011 のように、すべて数字で構成される日付形式を読み込む場合は、LoadWave/B フラグを使って、そのデータが日付であることを指定する必要があります。
そうしないと、LoadWave はそれを通常の数値として処理します。
/B フラグはダイアログから生成することはできないため、コマンドラインから LoadWave コマンドを使う必要があります。

別の方法としては、/B フラグなしで LoadWave コマンドを生成するダイアログを使い、LoadWave コマンドが実行されたときに表示される Load Delimited Text ダイアログで、その列が日付列であることを指定します。

列ラベル

各列には、オプションで列ラベルを付けることができます。
1D ウェーブを読み込む時、ウェーブ名を読み込む場合、およびファイルが列ラベルを持つ場合は、列ラベルをウェーブ名として使います。
そうでない場合、Igor は自動的に wave0、wave1 といった形式のウェーブ名を生成します。

ラベル行のテキストがデータ（数値、日付、時刻、または日時）として解釈できない場合、またはテキストがシングルクオートまたはダブルクオートで引用されている場合、そのテキストを列ラベルとみなします。

2D ウェーブを読み込む時は、オプションとしてウェーブの列の次元を設定するために列ラベルを使います。
ウェーブ名は列ラベルからではなく、Igor によって自動的に割り当てられます。
必要であれば、読み込み後にウェーブの名前を変更することができます。

Igor は、フォームの行に列ラベルが出現することを期待しています。

```
<label><delimiter><label><delimiter>…<label><terminator>
```

<column label>は以下のいずれかの形式です：

`<label>` （クオートなしのラベル）

`"<label>"` （ダブルクオート付きのラベル）

`'<label>'` （シングルクオート付きのラベル）

デフォルトの区切り文字はタブとカンマです。
他の区切り文字を使うための調整方法もあります（「区切り付きテキストの調整」のセクションを参照）。

Igor は、列ラベルの行がファイルの先頭に表示されることを想定しています。
この想定が当てはまらない場合は、Tweaks [微調整]（「区切り付きテキストの調整」のセクションを参照）を使って指定することができます。

Igor は、必要に応じて、ファイル内の列ラベルを標準の命名規則に従って、有効なウェーブ名となるようにクリーンアップします。
クリーンアップ処理では、不正な文字をアンダースコアに変換し、長い名前を最大 255 バイトに切り詰めます。

区切り記号付きテキストの例

区切り記号付きテキストファイルに含まれるテキストの例を紹介します。
これらの例はタブ区切りです。

単純な区切り付きテキスト

```
ch0         ch1         ch2         ch3     （ラベル行はオプション）
2.97055     1.95692     1.00871     8.10685
3.09921     4.08008     1.00016     7.53136
3.18934     5.91134     1.04205     6.90194
```

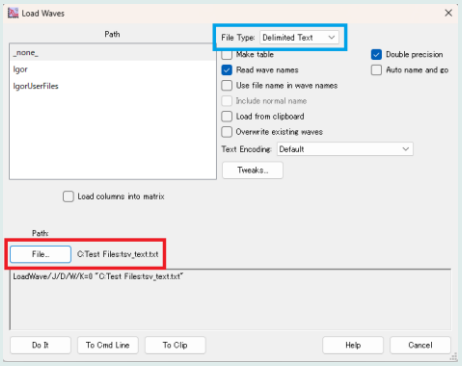
このテキストを読み込むと、３つのポイントを持つ４つのウェーブが作成されます。または、行列として読み込むように指定すると、３行４列の１つのウェーブが作成されます。

詳細な例は改めて記載しますが、簡単に結果を把握するために、上記のテキストをテキストファイルとして保存したところから動作を確認します。

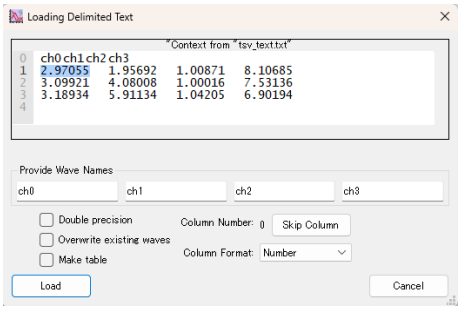
1. **メニュー Data → Load Waves → Load Waves を選択します。**

   File ボタンをクリックし、作成したテキストファイルを選択します。

   Do It をクリックします。

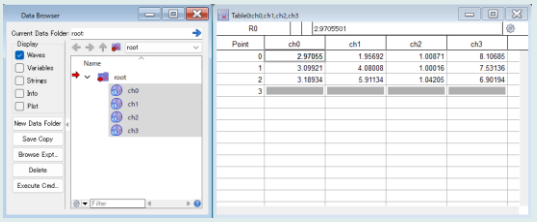


2. Loading Delimited Text ダイアログが開きます。



3. **まずはデフォルトの状態で Load ボタンをクリックします。**

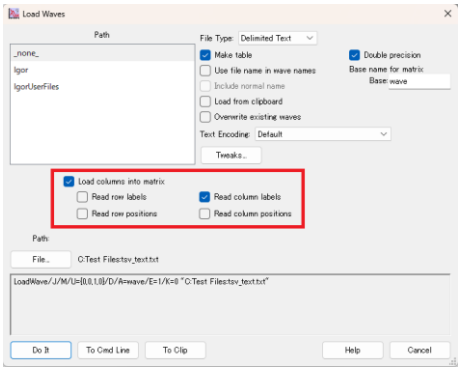
   ３つのポイントを持つ４つのウェーブが作成されます。



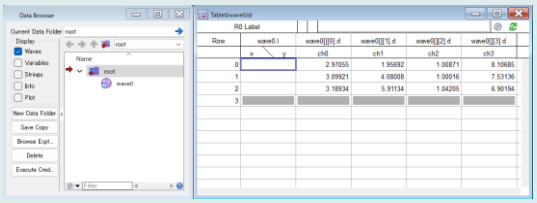
4. **もう一度新規の Experiment を作成して、Load Waves ダイアログを表示します。**

   今度は、Load columns into matrix チェックボックスにチェックを入れます。
   Read column labels にもチェックを入れます。

   Do It をクリックします。



5. ３行４列の１つのウェーブが作成されます。

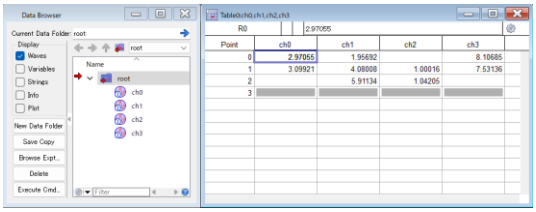


欠損値のある区切り付きテキスト

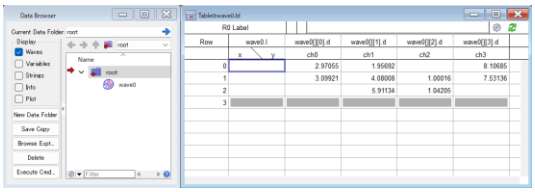
```
ch0             ch1             ch2             ch3     （ラベル行はオプション）
2.97055         1.95692                         8.10685
3.09921         4.08008         1.00016         7.53136
                5.91134         1.04205
```

このテキストを 1D ウェーブとして読み込むと、４つのウェーブが作成されます。
通常、各ウェーブには３つのポイントが含まれますが、列の末尾にある空白を無視するオプションがあります。
このオプションを使うと、ch0 と ch3 には２つのポイントが設定されます。
行列として読み込むと、0、2、３列目が空白の３行４列のウェーブが１つ作成されます。

３つのポイントを持つ４つのウェーブを作成した場合



３行４列の１つのウェーブを作成した場合

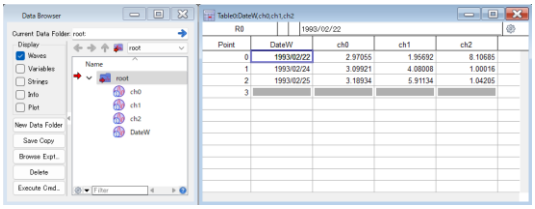


日付列を持つ区切り記号付きテキスト

```
Date            ch0             ch1             ch2     （ラベル行はオプション）
2/22/93         2.97055         1.95692         1.00871
2/24/93         3.09921         4.08008         1.00016
2/25/93         3.18934         5.91134         1.04205
```

このテキストを 1D ウェーブとして読み込むと、３つのポイントを持つ４つのウェーブが作成されます。
Igor は、最初の日付を、日付の保存システム（1904 年 1 月 1 日からの秒数）を使って適切な数値に変換します。
このデータは行列として読み込むには適していません。

３つのポイントを持つ４つのウェーブとなった結果



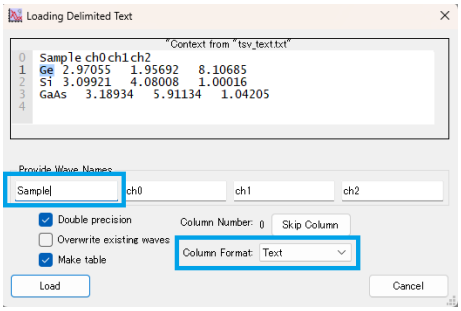
非数値列を持つ区切り記号付きテキスト

```
Sample          ch0             ch1             ch2     （ラベル行はオプション）
Ge              2.97055         1.95692         1.00871
Si              3.09921         4.08008         1.00016
GaAs            3.18934         5.91134         1.04205
```

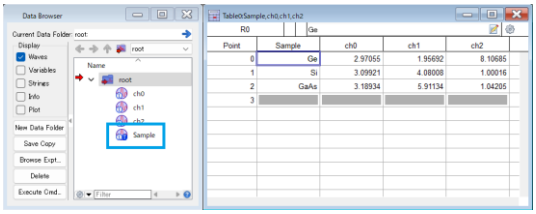
このテキストを 1D ウェーブとして読み込むと、３つのポイントを持つ４つのウェーブが作成されます。
最初のウェーブはテキストウェーブとなり、残りは数値となります。

また、これを 3x3 の単一行列として読み込み、行列の最初の行を列ラベル、最初の列を行ラベルとして扱うこともできます。
行列として読み込んだが、最初の列をラベルとして扱わなかった場合、数値ウェーブではなく、３行４列のテキストウェーブが作成されます。

最初の列がテキストウェーブとして認識された結果



３つのポイントを持つ４つのウェーブとなった結果
（最初のウェーブはテキストウェーブ）

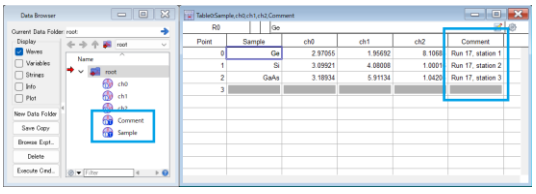


クオートで囲まれた文字列を持つ区切り記号付きテキスト

```
Sample          ch0             ch1             ch2             Comment
Ge              2.97055         1.95692         1.00871         "Run 17, station 1"
Si              3.09921         4.08008         1.00016         "Run 17, station 2"
GaAs            3.18934         5.91134         1.04205         "Run 17, station 3"
```

Igor Pro 8.0 以降、Load Delimited Text（LoadWave/J）は、ASCII のダブルクオートを、区切り文字を含む文字列を囲むものと認識します。
上記の場合、Comment 列にカンマを含むテキストが含まれています。
カンマは通常、区切り文字ですが、列のテキストがクオートで囲まれているため、LoadWave では区切り文字として扱われません。
詳細は「区切り文字付きテキストファイルにおけるクオート付き文字列」のセクションを参照してください。

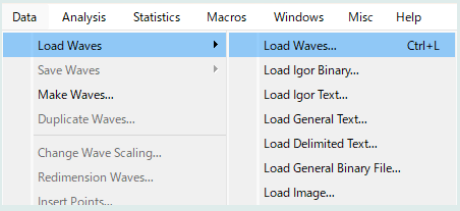
Comment 列がカンマを含めて格納された結果



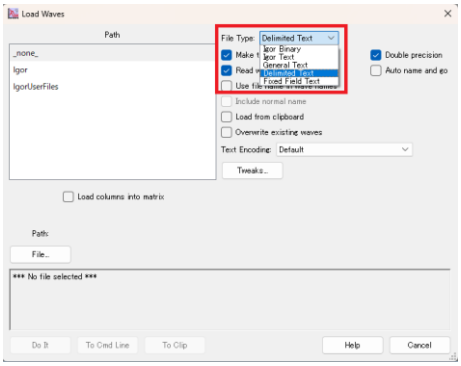
区切り記号付きテキストに対する Load Wave ダイアログ – 1D

ここまでのセクションですでにある程度、説明しましたが、区切り記号付きテキストファイルから 1D データを読み込む基本的なプロセスは次の通りです。

1. **メニュー Data → Load Waves → Load Waves を選択します。**

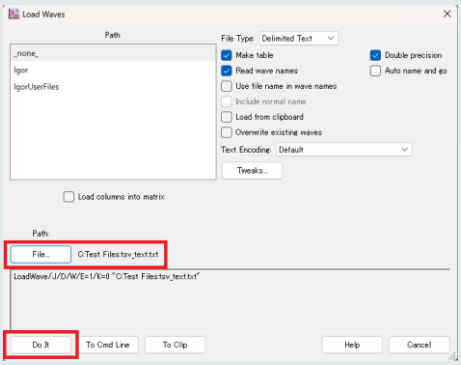


2. **File Type ポップアップメニューから Delimited Text を選択します。**



3. **File ボタンをクリックして、データを含むファイルを選択します。**

   Do It をクリックします。



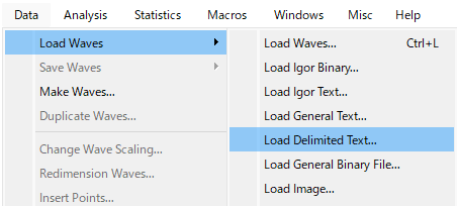
Do It をクリックすると、LoadWave コマンドが起動します。
Load Delimited Text ルーチンが次のステップで実行されます。

1. オプションで、列ラベルの行があるかどうかを決定する
2. 列の数を決定する
3. 各列のフォーマットを決定する（数値、テキスト、日付、時刻、日付/時刻）
4. オプションで、ウェーブの名前の確認または変更ができる別のダイアログを表示する
5. ウェーブを作成する
6. ウェーブにデータを読み込む

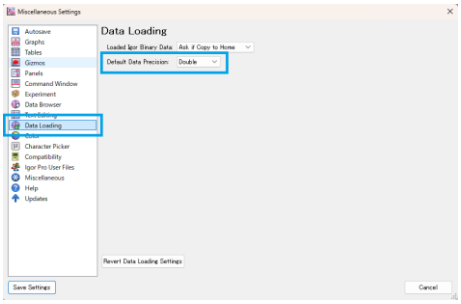
「Read wave names」オプションが有効な場合のみ、Igor がラベルの行を探します。
このオプションを有効にし、ラベルの行が見つかった場合、ファイルに含まれる列の数を決定します。
そうでなければ、ファイルの最初の行にあるデータ項目の数を数え、残りの行も同じ数の列があるものと想定します。

上記のステップ 3 では、Igor は各列の最初のデータ項目を調べ、各列のフォーマットを決定します。
与えられた列の最初の項目から決定したフォーマットを使って、その列の残りの項目をすべて解釈しようとします。

メニュー Data → Load Waves → Load Waves ではなく、Data → Load Waves → Load Delimited Text を選択すると、Open File ダイアログが表示され、直接ロードする区切り記号付きテキストファイルを選択できるようになります。
これは、Load Waves ダイアログをスキップし、デフォルトのオプションを使ってロードするショートカットです。
これは常に 1D ウェーブを読み込み、行列は読み込みません。



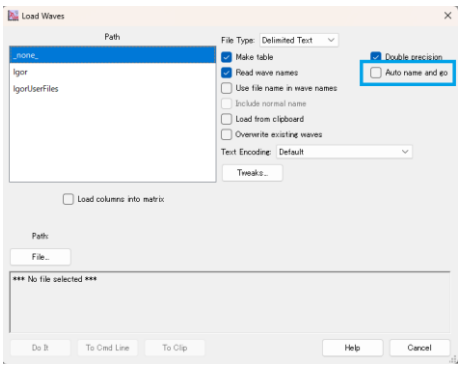
数値ウェーブの精度は、Miscellaneous Settings ダイアログ（Misc メニュー）の Data Loading セクションにある Default Data Precision 設定によってコントロールされます。
このショートカットを使う前に、Load Waves ダイアログで使うことができるオプションを確認してください。



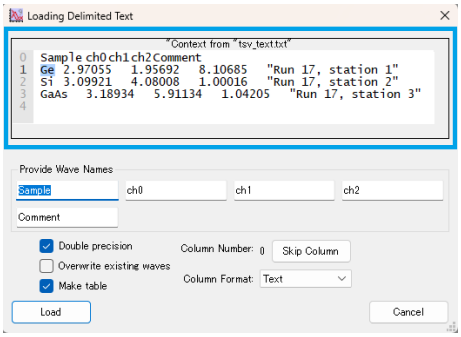
ウェーブ名の編集

Load Wave ダイアログの「Auto name & go」オプションは、プロシージャのコントロール下で 1D データを読み込む時に、すべてを自動したい場合に主に使われます。
1D データを手動で読み込む時には、通常、Auto name & go オプションは選択しないままにしておきます。

次に、Igor はウェーブ名を確認または変更できる Loading Delimited Text ダイアログを追加で表示します。



Loading Delimited Text ダイアログの Context 領域（上部）では、読み込もうとしている内容についてのフィードバックが表示されます。
ここではファイルは編集できません。
ファイルを編集したい場合は、ロードを中止し、Igor ノートブックとしてファイルを開くか、テキストエディターで開いてください。



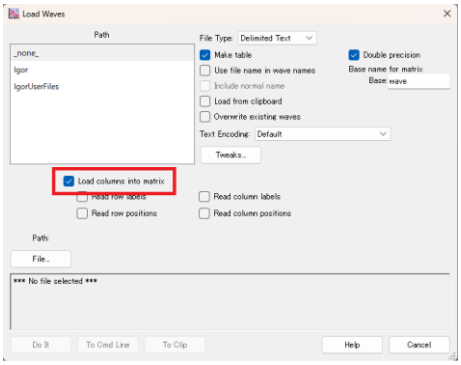
区切り記号付きテキストデータの読み込み後にスケーリングを設定する

1D の数値データが X 軸方向に等間隔で配置されている場合、ウェーブフォーム用に設計された多くのコマンドや機能を使うことができます。
ウェーブを読み込んだ後、Change Wave Scaling ダイアログ（メニュー Data → Change Wave Scaling）を使って、ウェーブの X スケールを設定する必要があります。

注記： 1D データが等間隔の場合、ウェーブの X スケーリングを設定することが非常に重要です。
Igor の多くのコマンドは、正しい結果を得るために X スケーリング情報に依存しています。

1D データが等間隔ではない場合は、XY ペアを使うため、X スケーリングを変更する必要はありません。
データの単位を設定するには、Change Wave Scaling を使うとよいでしょう。

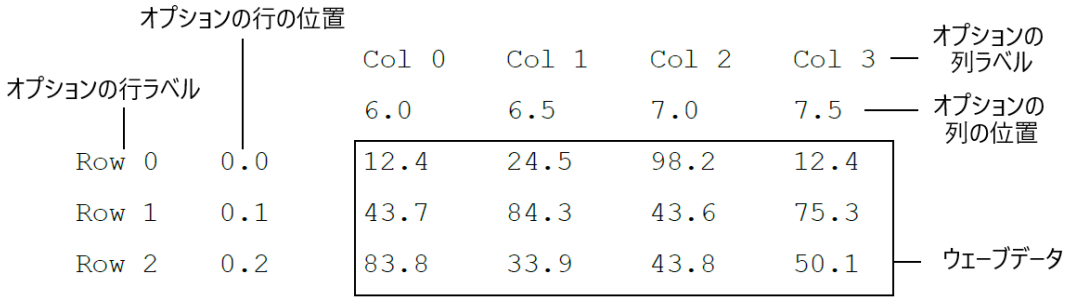
区切り記号付きテキストに対する Load Wave ダイアログ – 2D



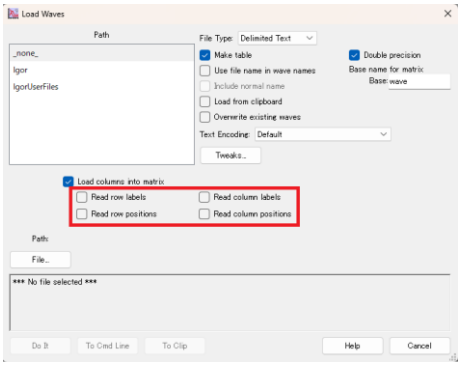
区切り記号付きテキストファイルを 2D データとして読み込むには、メニュー Data → Load Waves → Load Waves を選択します。
次に、Load Waves ダイアログで Load columns into matrix チェックボックスを選択します。

テキストファイルから行列（2D ウェーブ）を読み込むと、１つのウェーブが作成されます。
そのため、２番目のダイアログでウェーブ名を入力する必要はありません。
代わりに、指定したベースの名前に基づいてウェーブを自動的に命名します。
読み込み後、必要に応じてウェーブの名前を変更することができます。

行/列/位置のコントロールを理解するには、Igor の 2D 区切り記号付きテキストファイルの表示を理解する必要があります。



最もシンプルなケースでは、ファイルにはウェーブデータのみが含まれ、ラベルや位置情報は含まれません。
その場合、Load Waves ダイアログの４つのラベル/位置のチェックボックスをすべて選択解除することで、これを指定します。



2D のラベルと位置の詳細

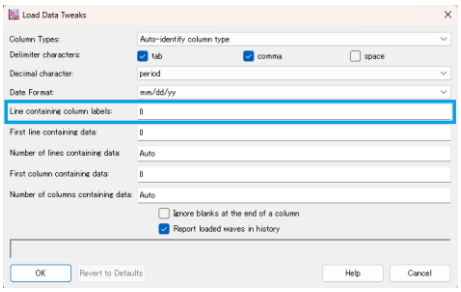
ファイルにラベルや位置情報がある場合は、Load Waves ダイアログの適切なチェックボックスを選択することで、それを指定します。
Igor は、Tweaks サブダイアログ（Load Waves ダイアログの Tweaks ボタン）を使って異なる指示をしていない限り、ファイルの最初の列に行ラベルが表示され、最初の行に列ラベルがあることを想定しています（「区切り記

号付きテキストの調整」のセクションを参照）。
ウェーブの次元ラベルに行/列ラベルを読み込みます（ヘルプ Multidimentional Waves を参照）。

列の位置は２つの方法で処理することができます。
ウェーブの次元スケーリングを設定するために使うか（位置が等間隔に配置されている場合）、位置に対して個別の 1D ウェーブを作成することもできます。
行ラベルの直後の列に行の位置があるか、ファイルに行ラベルが含まれていない場合は最初の列に行の位置があることを想定しています。
列の位置は、列ラベルの直後にあるか、ファイルに列ラベルが含まれていない場合は最初の行にあることを想定しています（Tweaks サブダイアログで異なる設定をする場合を除く）。

行位置ウェーブは、ファイルの行位置の列の数値を持つ 1D ウェーブです。
ウェーブ名は、ロードされた行列ウェーブの名前に、「RP_」（行位置/Row Position）と付けたものです。
列位置ウェーブは、ファイルの列位置のラインに含まれる数値を持つ 1D ウェーブです。
ウェーブ名は、ロードされた行列ウェーブの名前に、「CP_」（列位置/Column Position）と付けたものです。
（別々の 1D ウェーブまたは行列ウェーブの次元スケーリングへの）ロードが完了すると、行列を画像として表示する場合や、行列の輪郭を表示する場合に、行と列の位置情報を使うことができます。

ファイルにデータ、列ラベル、列位置の前にヘッダー情報が含まれている場合は、Tweaks（微調整）サブダイアログを使って、対象のデータがどこにあるかを指定する必要があります。
Line containing column labels 設定では、列ラベルを探す行を指定します。ファイルの最初の行は行 0 とみなされます。

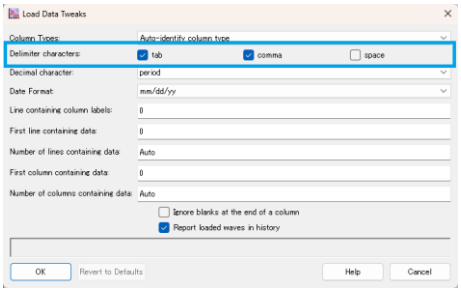


LoadWave に列の位置を読み取るように指示した場合、列ラベルの読み取りも指示したかどうかによって、２つの方法のうちの１つで、どの行にそれらが含まれるかを決定します。
LoadWave に列ラベルを読み取らせる場合、LoadWave は列ラベルの行の直後に列の位置の行が続くものと想定します。
LoadWave に列ラベルの読み取りを指示しない場合、LoadWave は列の位置が最初のデータ行の直前に位置していると仮定します。

区切り記号付きテキストファイルからテキストウェーブを読み込む

テキストウェーブにデータを読み込む時に、特殊文字に関連するいくつかの問題に対応する必要があるかもしれません。

デフォルトでは、カンマとタブ文字を区切り文字として扱い、１つの列と次の列を区切ります。
読み込むテキストに、区切り文字としてではなく値としてカンマやタブが含まれている可能性がある場合は、区切り文字を変更する必要があります。
これは、Load Delimited Text ダイアログの Tweaks サブダイアログを使って行うことができます。



Load Delimited Text 操作では、常に改行（CR）文字とラインフィード（LF）文字をテキストの行の終わりを示すものとして扱います。
これらの文字を値として使っているデータファイルはほとんどないと思います。

改行やラインフィードを値として読み込む必要があるという希なケースでは、エスケープシーケンスを使うことができます。
改行値を “\r”（クオートは不要）に、ラインフィード値を “\n” に置き換えます。
Igor はこれらを改行（CR）とラインフィード（LF）に変換し、テキストウェーブに適切な文字を格納します。

“\r” と “\n” に加えて、“\t” もタブ値に変換し、その他のエスケープシーケンスも変換します（ヘルプ Escape Sequences in String を参照）。
これらの変換により、まれに発生する可能性がある問題があります。
エスケープシーケンスとして処理したくない “\r”、“\n”、“\t” シーケンスを含むテキストを読み込む必要があるかもしれません。
これらを改行やタブに変換しないようにするには、“\\r”、“\\n”、“\\t” に置き換える必要があります。

Igor は、区切り記号付きテキストファイルからテキストウェーブにデータを読み込む時に、クオテーションマークを削除しません。
必要であれば、ファイルをノートブックとして開き、読み込み前に一括変換を行うか、読み込んだウェーブをテーブルに表示し、メニュー Edit→Replace を使って置換することができます。

区切り記号付きテキストファイル内のクオートで囲まれた文字

カンマ区切り値（CSV）テキストファイルは、カンマを区切り文字として、区切り記号付きテキストファイルとして読み込むことができます。
CSV テキストファイルにある次のようなテキストがあるとします。

```
1,London
2,Paris
3,Rome
```

CSV ファイルでは、項目を囲むためにダブルクオートが使われることがあります。
例えば、

```
1,”London”
2,”Paris”
3,”Rome”
```

Igor 6 と Igor 7 では、区切り記号付きテキストファイル内のダブルクオートは特別な処理はされませんでした。
したがって、２番目の例を読み込むと、1、2、3 を含むウェーブと “London”、”Paris”、”Rome” を含むテキストウェーブを作成します。
テキストウェーブにはダブルクオート文字が含まれます。

Igor 8 以降では、デフォルトで、区切り記号付きテキストの読み込みルーチンは、ASCII のダブルクオート文字を囲み文字として扱い、ウェーブには読み込まれません。
したがって、２番目の例を読み込むと、テキストウェーブには、London、Paris、Rome が含まれ、ダブルクオート文字はありません。

この機能は、クオートで囲まれた文字列にカンマが含まれている場合に特に便利です。
次の例を見てください：

```
1,”123 First Street, London, England”
2,”59 Rue Poncelet, Paris, France”
3,”Viale Europe 22, 00144 Rome, Italy”
```

Igor 8 以前では、ダブルクオートは特別な処理を受けず、カンマはデフォルトで区切り文字として扱われたため、このテキストを４つの列を含むものとして処理していました。
Igor 8 以降ではこれを２つの列として読み込み、３つの数値からなる数値ウェーブと、３つの住所からなるテキストウェーブを作成します。

初期のバージョンで確立された区切り付きテキストファイルの列名に関するルールにより、LoadWave/W フラグを使ってファイルに列名を含めるように指定すると、テキストがすべて数値であっても、LoadWave はクオートで囲まれたテキストを列名として解釈します。
例えば、LoadWave/W を使い、ファイルが次を含んでいる場合：

```
”1”,”2”,”3”
”4”,”5”,”6”
```

LoadWave 最初の行を列名として処理します。
しかし、LoadWave/W を使い、ファイルが次を含んでいる場合：

```
1,2,3
4,5,6
```

LoadWave は最初の行を列名ではなく、データとして処理します。
したがって、ファイルにクオートで囲まれた文字列が含まれている場合で、ファイルに列名が含まれていない場合は、/W フラグを省略する必要があります。

区切り記号付きテキストの Igor 7 との互換性

Igor 8 以降では、LoadWave/J はカンマ区切りファイルなどの区切り記号付きテキストファイル内のクオート付き文字列の処理が改善されました。
詳細は「区切り記号付きテキストファイル内のクオートで囲まれた文字」のセクションを参照してください。

副作用として、まれに Igor 8 以降では、Igor 7 と比較して特定のファイルに対して異なる結果が生成されることがあります。
例えば、Igor 7 は、このデータが２つの数値列を含むものとして自動的に識別します。

```
1;,2;
```

Igor 8 は、予期せぬセミコロンがあるため、これが２つのテキスト列を含むものとして自動的に識別されます。
この変更がファイルの読み込みに支障をきたす場合は、LoadWave/V フラグの loadFlags パラメーターの bit 4 を設定することで、Igor 8 以降を Igor 7 と同様に動作させることができます。
例えば、

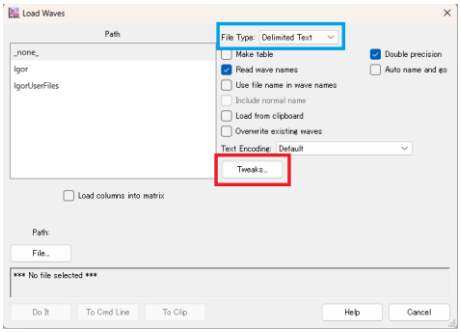
```
LoadWave/J/V={",", " ", 0, 16}              // 16 が bit 4 を設定することを意味する
```

これにより、Igor 8 のクオートで囲まれた文字列のサポートが無効になり、LoadWave の動作が Igor 7 により近くなります。

別の回避策としては、LoadWave に特定の列に対して特定のデータ形式を指定するという方法があります。
これは、Loading Delimited Text ダイアログの Column Format 使うか（「ウェーブ名の編集」のセクションを参照）、LoadWave/B フラグを使うことで実行できます。

区切り記号付きテキストの微調整（Tweaks）

区切り記号付きテキストファイルの基本的な形式には多くのバリエーションがあります。
バリエーションを使うファイルを読み込む必要がある場合に、処理を指示するための微調整（Tweak）ができるようになっています。
これを行うには、LoadWaves ダイアログの Tweaks ボタンを使います。



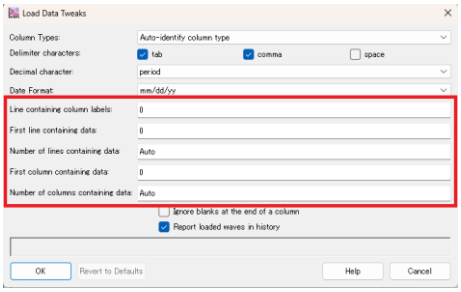
Tweaks ダイアログでは、スペース文字を区切り文字として指定できます。
他の区切り文字を指定するには LoadWave コマンドを使います。

スペースを区切り文字として許可する理由は、列の揃えにスペースを使っているファイルを読み込むことができるようにするためです。
これは、FORTRAN プログラムによって生成されるファイルの一般的な形式です。
通常、これらのファイルをロードするには、区切り記号付きテキスト（Delimited Text）用のローダーではなく、固定フィールドテキスト（Fixed Field Text）用のローダーを使う必要があります。
区切り記号付きテキストローダーを使う場合で、スペースを区切り記号として使える場合、Igor は連続するスペースを１つの区切り記号として扱います。
つまり、２つの連続したスペースは２つの連続したタブの場合と違い、値が欠けているということを示すものではありません。

区切り記号付きテキストファイルを読み込む時、デフォルトでは、Igor はファイルの最初の行に列ラベルまたはデータの最初の行のいずれかが含まれることを想定しています。
この想定に合わないファイルに対して、いくつかの調整を行うことができます。

Tweaks ダイアログ内の行と列は、ゼロから始まる番号が振られています。

Line containing column labels 調整を使うと、列ラベルをゼロ行以外に配置する行を指定することができます。
この設定と First line containing data 設定を併用すれば、ファイルの冒頭にある不要なデータをスキップするように指示することができます。



First line containing data、Number of lines containing data、First column containing data、number of columns containing data 設定は、ファイル内の任意の場所から任意のデータブロックを読み込むことができるように設計されています。
これは数百の列を持つファイルがあり、そのうちの一部にだけ関心がある場合に役立つかもしれません。

Number of lines containing data が auto または 0 に設定されている場合、ファイルの終わりに到達するまですべての行を読み込みます。
Number of columns containing data が auto または 0 に設定されている場合、Igor はファイルの最後の列に到達するまで全ての列を読み込みます。

Ignore blanks at the end of a column 設定の適切な使い方は、ファイルに保存されている 1D データの種類によって異なります。
ファイルにいくつかの類似した列が含まれている場合、例えば、デジタルオシロスコープからの４チャンネルのデータなど、おそらくファイル内のすべての列を同じ長さのウェーブに読み込みたいと思います。
特定の列の末尾に１つ以上の欠損値がある場合、欠損値を表す NaN をウェーブの対応するポイントに含める必要があります。
一方、ファイルに異なる列が多数含まれる場合、列の末尾にある空白ポイントを無視すると、結果として得られるウェーブの長さが等しくならないことがあります。
Ignore blanks at the end of a column を有効にすると、LoadWave は列の末尾の空白を 1D ウェーブに読み込まなくなります。
このオプションが有効になっていて、特定の列が空白のみの場合、そのウェーブは全く読み込まれません。

区切り記号付きテキストファイルでのトラブルシューティング

Load Delimited Text ルーチンによって作成されたウェーブは、テーブルを使って確認することができます。
期待通りの結果が得られない場合は、他の LoadWave オプションを試したり、Igor が処理できる形式になるまでテキストファイルを調べて編集することができます。
以下の点を覚えておいてください：

- Igor は、異なる設定を行わない限り、タブまたはカンマで区切られた数値、テキスト値、日付、時刻、または日付/時刻で構成されるファイルを想定しています。
- Igor は、Tweaks（微調整）で別の設定をしていない限り、列ラベルがある場合はファイルの最初の行にあることを想定しています。
  列ラベルも、Tweaks（微調整）で別の設定をしていない限り、タブまたはカンマで区切られていると想定しています。
  1D ウェーブの Read Wave Names オプション、または 2D ウェーブの Read Column Labels オプションを有効にしない限り、列ラベルの行を検索しません。
- Igor は、列ラベル行を調べるか、列ラベル行がない場合はデータの最初の行を調べて、ファイルの列数を決定します。

ファイルを調べるだけでは問題を特定できない場合は、次のトラブルシューティングのテクニックを試してください。

- ファイルの最初の何行かをテストファイルにコピーします。
- テストファイルをロードし、テーブルで読み込み結果のウェーブを調べます。
- テストファイルをノートブックで開きます。
- ファイルを編集して不具合を修正し、保存して再度ロードします。
  ノートブックとして開いている場合でも、区切り記号付きテキストとしてファイルを読み込むことができます。
  ノートブックに変更を加えた場合は、読み込む前に保存されていることを確認してください。
- 再度、読み込まれたウェーブを調べます。

このプロセスにより、通常、ファイルのどの部分に問題があるかが明らかになります。
ファイルの小さなサブセットで作業を行うと、試行錯誤の調査を素早く簡単に実行できます。